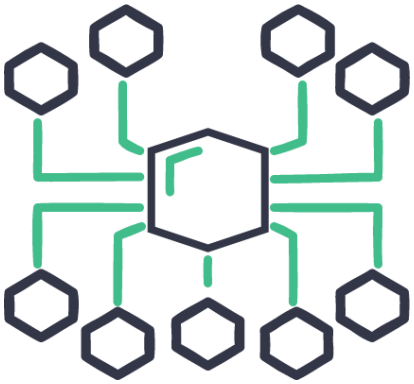
Proyecto Final

Ingeniería en Sistemas de Información



Accelerate

Sistema generador automático de código e infraestructura

Autores:

Peña Altare, Pablo Roberto – 45173 – DNI 42712757

Fernández Valenzuela, Valentín Manuel – 45091 – DNI 42505935

Rivero Zuin, Maximiliano Tomás – 43742 – DNI 41230862

Directores del trabajo:

Vázquez, Alejandro

Moralejo, Raúl

Manino, Gustavo

Lemos, Gustavo

Casas, Malena

Año académico 2022

Mendoza, 08 de Noviembre del 2022

Biblioteca del Departamento de Ingeniería en Sistemas de Información

Universidad Tecnológica Nacional

Facultad Regional Mendoza

S_____/_____D

Nos dirigimos con el fin de realizar la entrega del trabajo final de la carrera Ingeniería en Sistemas de Información.

El proyecto llevado a cabo como trabajo final de la carrera es "Accelerate - Sistema generador automático de código e infraestructura".

La documentación presentada es la carpeta completa que incluye: Desarrollo de un sistema de información real, Planificación de proyectos de sistemas, Trabajo Práctico Integrador "Dirección de Proyectos de Sistemas", Trabajo Práctico Integrador "Gerenciamiento de Sistemas", Diagrama de tiempos, Diagrama de tiempos de planificación de Implementación, Manual de usuario, Metamodelo accelerateMLI, Metamodelo accelerateMLS y Metamodelo accelerateMLC.

Sin otro particular, nos despedimos cordialmente.

Rivero Zuin, Maximiliano Tomás
43742

Fernández Valenzuela, Valentín Manuel
45091

Peña Altare, Pablo Roberto
45173

Resumen

Accelerate consiste en la implementación de una herramienta grafica de modelado de sistemas que permite utilizar el diagrama diseñado para la generación automática de código y archivos de configuración de infraestructura en la nube.

El proyecto utiliza como base los contenidos del Desarrollo de Software Dirigido Por Modelos (MDD, Model Driven Development) y surge de acuerdo con el problema detectado en diversos proyectos de software, los cuales pierden valioso tiempo de desarrollo con tareas de configuración repetitivas en la etapa inicial del proyecto y frecuentemente no se adhieren a patrones o estándares de la industria. A su vez, se requiere conocimientos en una amplia variedad de tecnologías y profesionales que logren integrarlas para comenzar con el desarrollo. En general se observa que en un proyecto de desarrollo convencional el 40% del tiempo se dedica a la programación el cual solo entre el 5% y 15% se dedica a la lógica del negocio y el resto del tiempo preparando la infraestructura para soportar la aplicación. Esta se divide en manejo de errores y seguridad, debugging y programación de código de infraestructura.

Por estas razones el objetivo de Accelerate es proveer una herramienta que asista a los desarrolladores en la etapa inicial del proyecto mediante modelos que unifiquen el diseño de una arquitectura de microservicios en la nube, permitiendo elegir distintos parámetros luego utilizados por la generación automática tales como el lenguaje de programación de cada microservicio, el proveedor cloud de cada ambiente de la arquitectura, entre otros. Luego, las personas involucradas en el software a desarrollar utilizan los archivos de configuración y el código generado como punto de partida sobre el cual se obtiene la ventaja de poder concentrarse únicamente en la lógica de negocio y sin preocuparse por la infraestructura del proyecto.

Palabras Clave

Metamodelo, Generador de código, Microservicio, Infraestructura como código, Aprovisionamiento en la nube, Desarrollo dirigido por Modelos.

Contenido

Índice de Figuras

Índice de Tablas

Anexos

Desarrollo de un sistema de información real

Sistema generador automático de código e infraestructura

Definición de requerimientos

Sistema generador automático de código e infraestructura

Definición de Requerimientos

Amplication

1) Relevamiento General

1.1) De la Organización

Amplication [1] es una herramienta de desarrollo de código abierto. Ayuda a los desarrolladores profesionales de Node.js a desarrollar aplicaciones de calidad sin perder tiempo en tareas de codificación repetitivas.

Amplication genera automáticamente aplicaciones back-end creadas con TypeScript y Node.js, y un cliente creado con React.

Esta organización se centra en 2 filosofías principales: el Código abierto y el Low Code que busca reducir la necesidad de escribir código repetitivo o también conocido como código boilerplate.

Al ser de Código Abierto la plataforma busca ser mantenida totalmente por la comunidad. Actualmente recibe contribuciones de más de 60 desarrolladores alrededor de todo el mundo.

La compañía tiene su sede en Tel Aviv-Yafo, Tel Aviv, Israel.

Visión

“Nuestra visión es crear una plataforma que permita a los desarrolladores profesionales crear aplicaciones comerciales y ampliar las capacidades de la plataforma, con el poder de colaboración y transparencia de la comunidad de código abierto.” (https://amplication.com/ , 2022)

Esta plataforma está directamente dirigida a desarrolladores y equipos de desarrollo que buscan reducir la necesidad de escribir código repetitivo. Permitiendo generar entidades y roles que serán la base del Backend junto a las conexiones a la base de datos. También genera una interfaz utilizando React de Administración de Entidades.

1.2) Funciones detectadas a nivel general y relaciones con otros Sistemas y Entidades

Crear Proyecto

Permite crear un nuevo proyecto vacío en el que se podrá configurar Roles, entidades, repositorios.

Descargar Código

Una vez diseñados y configurados los componentes se utiliza esta funcionalidad para generar el código y descargar todos los archivos automáticamente.

Diseño de Entidades

Permite diseñar entidades del modelo, agregando las entidades al modelo. En estas se gestionan los permisos para distintos roles y se configuran los distintos atributos y relaciones.

Diseño de Roles

Se crean distintos roles con un nombre y descripción, los cuales después se utilizan para dar permisos para cada entidad.

Login al Sistema

El sistema requiere una cuenta para iniciar sesión. Esto se hace a través de una cuenta de GitHub.

Sincronizar Repositorio

Se sincroniza directamente a un repositorio remoto de GitHub [16] por lo que en vez de descargar el código se puede hacer pull request directamente al repositorio.

Subida a GitHub

Una vez que se diseñaron los componentes se utiliza esta funcionalidad para subir el código generado directamente al repositorio Sincronizado.

Por lo que se analizó en las funciones anteriores se detecta que la aplicación interactúa con 2 entidades:

Usuario: En base a como está diseñada la aplicación este usuario al que está dirigido será un desarrollador, diseñador de sistemas, analista funcional, o cualquier individuo o equipo que busque generar código.

GitHub: Se comunica con GitHub en 3 instancias, para generar/verificar una cuenta para iniciar sesión, al vincular el proyecto en el que se está trabajando con un repositorio y por último al momento de realizar commits al repositorio remoto.

A continuación, se presenta un diagrama de las relaciones y las interacciones.

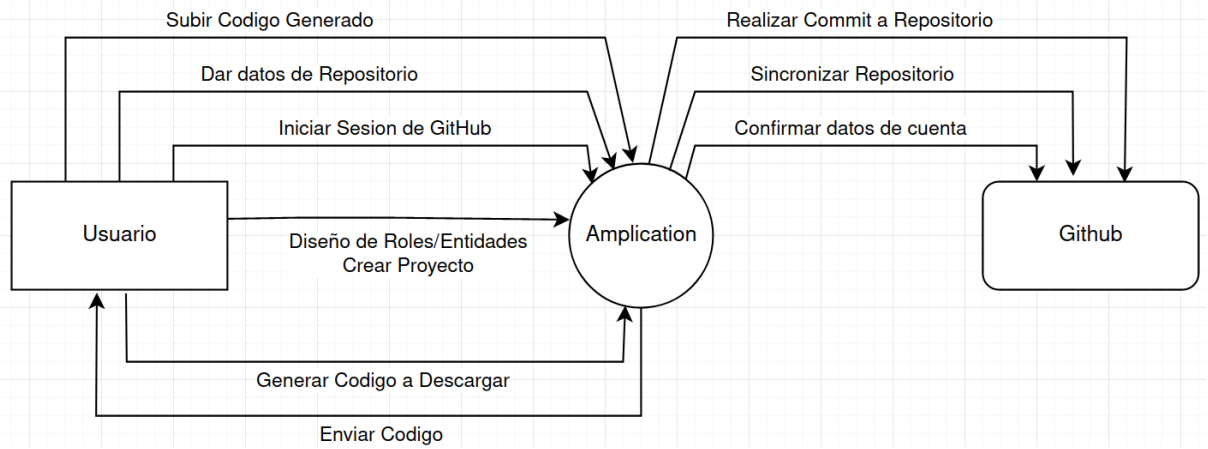


Figura 1 - Diagramas de relaciones e interacciones de Amplication

1.3) Tecnología de Información

Las tecnologías que utiliza para la generación de código son:

Para el lado del Servidor:

- NestJS - Un framework de Node.js [17] progresivo para crear aplicaciones del lado del servidor eficientes, confiables y escalables.
- Prisma - ORM para NodeJS y TypeScript
- PostgreSQL - Base de Datos SQL
- Passport - Autenticación simple y discreta para Node.js
- GraphQL - Lenguaje de Query para APIs
- Swagger UI - Documentación visual para API REST basada en la especificación OpenAPI.
- Jest - Framework de testing para javascript
- Docker - Plataforma de Contenedores.

Para el lado del Cliente

- ReactJS - A JavaScript library for building user interfaces. Libreria Javascript para construir interfaces de usuario.
- Axios - Cliente HTTP para Javascript.

También utilizan otras tecnologías dentro de su web app como:

- GitHub: Herramienta de Versionado de Código
- Bootstrap: Biblioteca multiplataforma de código abierto para diseño de sitios y aplicaciones web.

2) Relevamiento detallado y análisis del Sistema

2.1) Detalle, explicación y documentación detallada de todas las funciones seleccionadas.

Crear Proyecto

Amplication permite crear un número ilimitado de proyectos, esto se hace a través del botón de New App+.

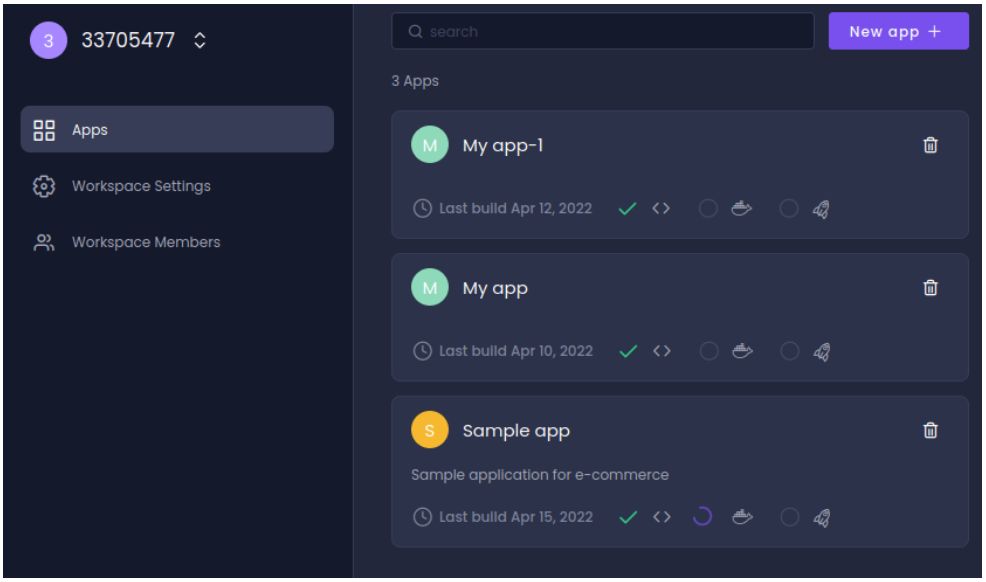


Figura 2 - Interfaz de proyectos creados Amplication

En la interfaz de creación de proyecto se puede crear un proyecto vacio, crear un proyecto base o subir uno utilizando un schema XML desde Excel.

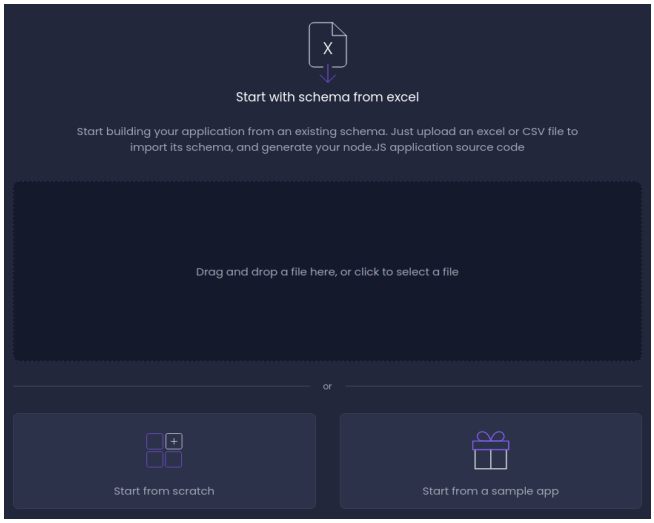


Figura 3 - Interfaz crear proyecto vacío o importar Excel

La diferencia principal entre estas opciones es si tendrá o no entidades y roles ya creados.

Una vez creado se puede ver en la ventana con todos los proyectos, el nuevo proyecto con un nombre por defecto y una fecha de creación.

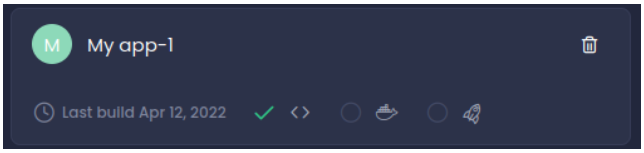


Figura 4 - Interfaz nuevo proyecto creado

Descargar Código

En la barra inferior de la aplicación se encuentran varias opciones, entre ellas la de descarga código indicada con una flecha.

Figura 5 - Interfaz descargar código

Este comienza el proceso de descarga del código al dispositivo donde se esté realizando.

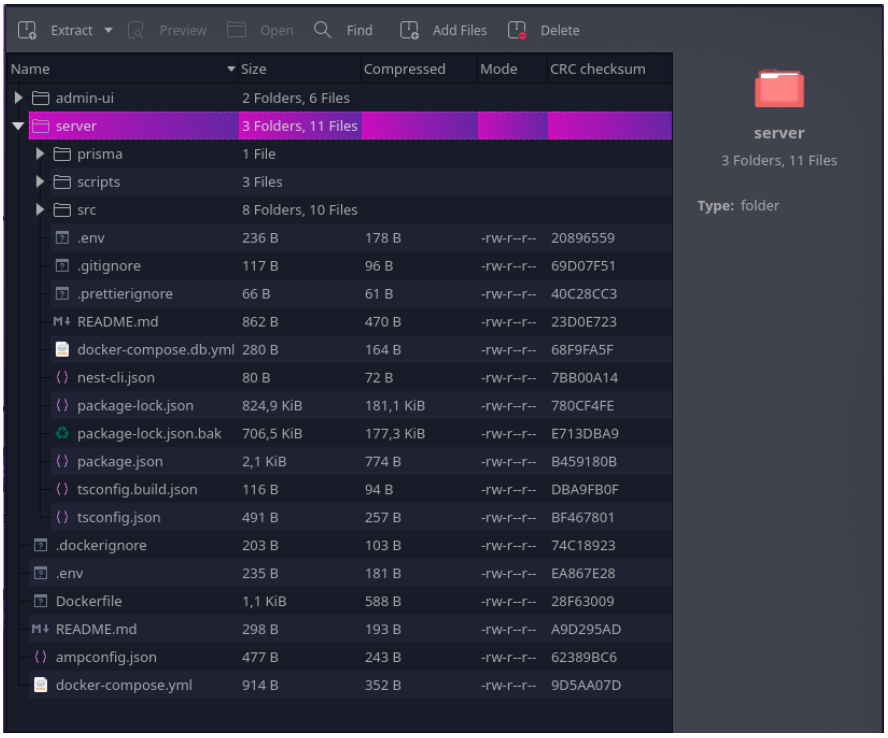


Figura 6 - Interfaz proceso de descarga de código

Se puede observar que descarga un archivo comprimido con todos los archivos.

Diseño de Entidades

En la interfaz de entidades nos encontramos con lo siguiente.

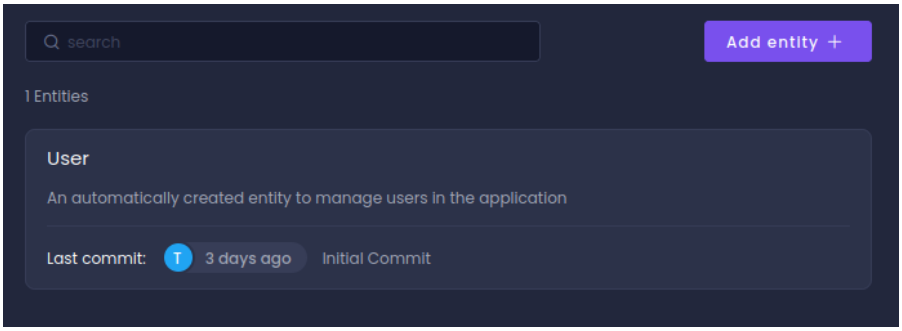


Figura 7 - Interfaz diseño de entidades

En esta pantalla se pueden crear nuevas entidades o modificar entidades existentes.

Si seleccionamos crear una nueva entidad nos encontramos con una ventana emergente para dar un nombre a esta nueva entidad. Una vez creada se agrega a la lista de entidades.

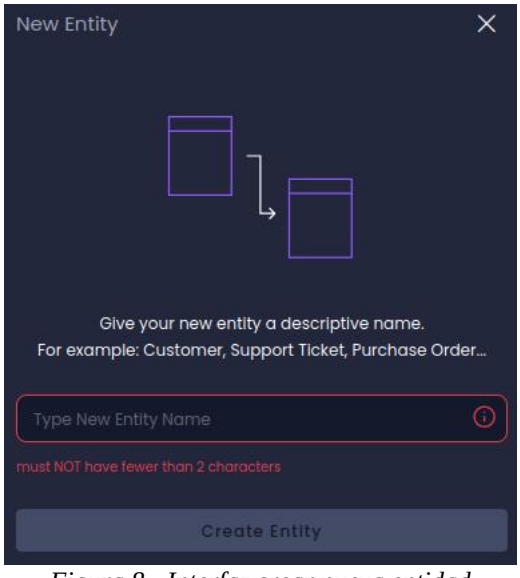


Figura 8 - Interfaz crear nueva entidad

Si se selecciona una entidad nos encontramos con una ventana de modificación en la cual podemos realizar acciones como agregar atributos y configurarlos, dar permisos específicos, etc.

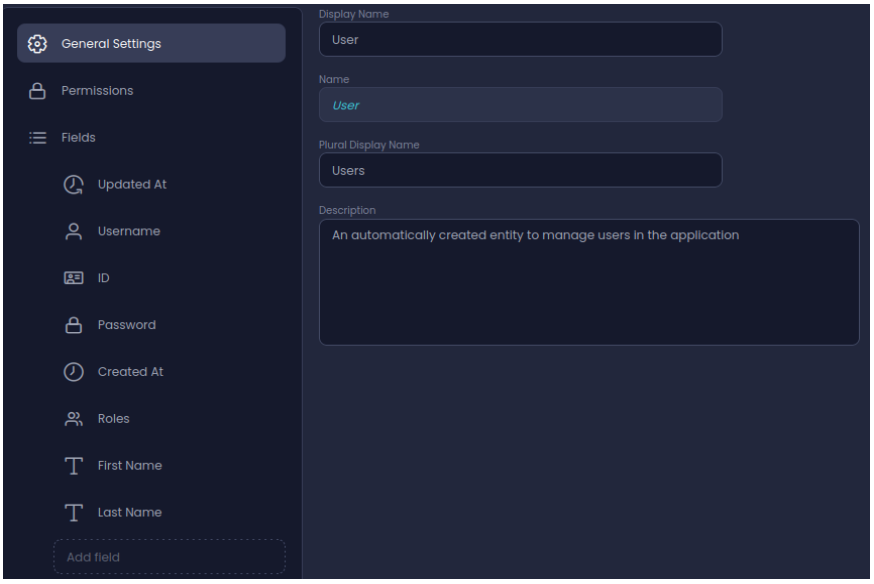


Figura 9 - Interfaz modificar configuraciones generales de una entidad

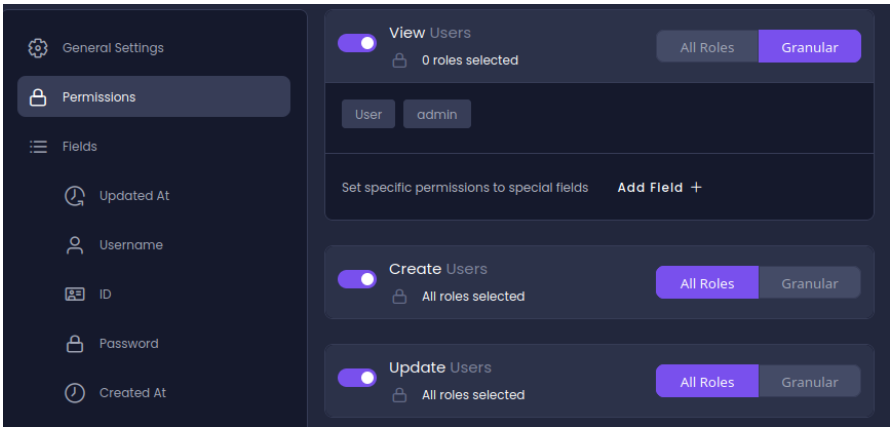


Figura 10 - Interfaz modificar permisos

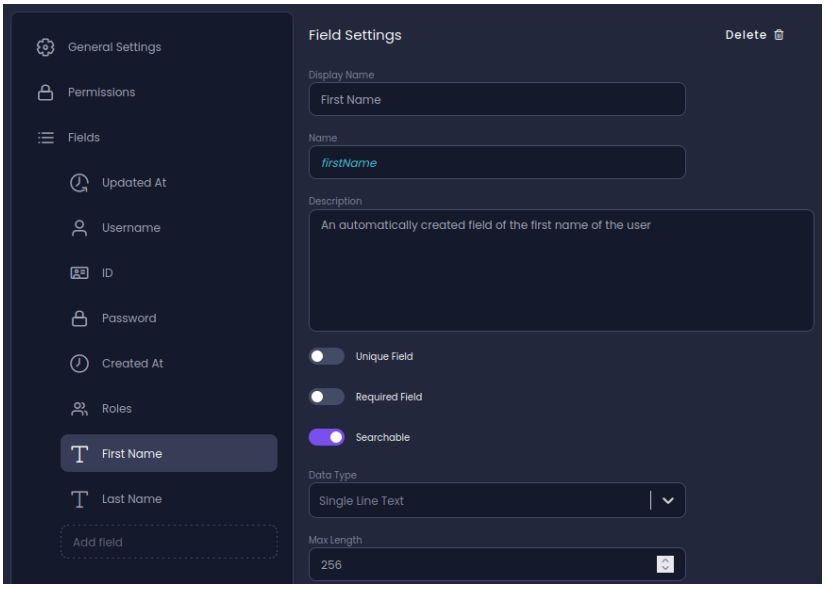


Figura 11 - Interfaz configuración de campos

Diseño de Roles

Como se aprecia en la figura anterior los permisos se asignan a roles, estos roles se crean en la ventana de gestión de roles.

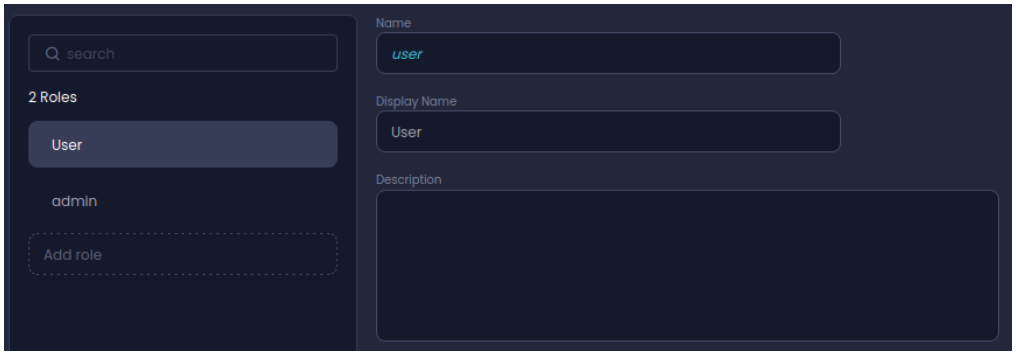


Figura 12 - Interfaz diseño de roles

Esta permite agregar roles nuevos, y modificar los existentes ya se cambiando su nombre o descripción.

Login al Sistema

Amplication no permite iniciar sesión con un mail y contraseña o a través de servicios de Google o Facebook. Para iniciar sesión en la aplicación se debe hacer a través de una cuenta de GitHub.

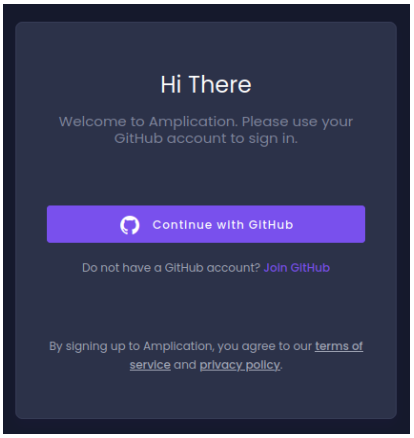


Figura 13 - Interfaz Login al sistema

Una Vez se presiona el botón “Continue with GitHub” la aplicación se comunica con GitHub y verifica la cuenta permitiendo así iniciar sesión en amplication.

Sincronizar Repositorio

Dentro de la ventana principal del proyecto se encuentra una funcionalidad de sincronización con un repositorio de GitHub.

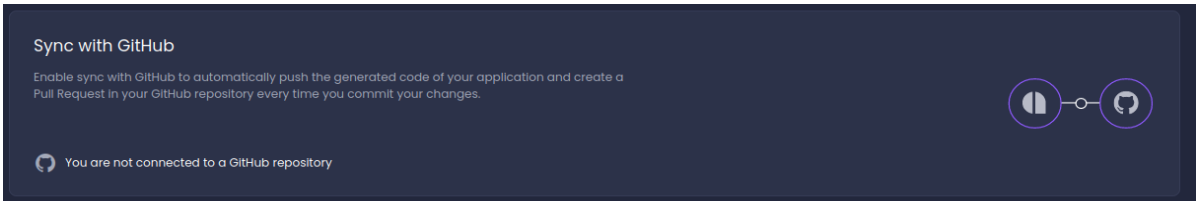


Figura 14 - Interfaz sincronizar repositorio

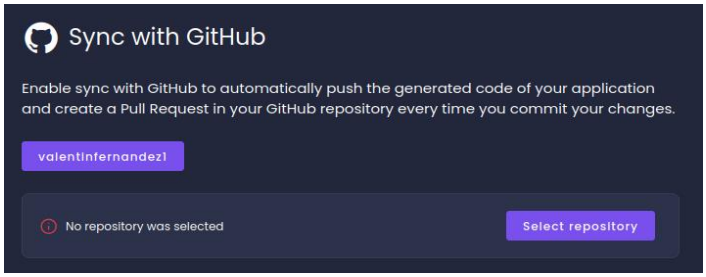


Figura 15 - Interfaz repositorio sincronizado

En esta opción de vincular repositorios, primero se debe dar permisos a amplication desde GitHub.

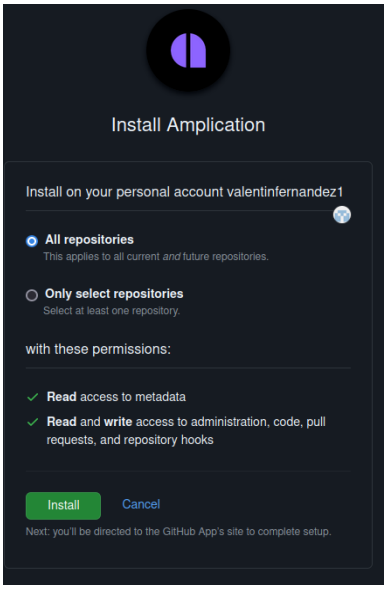


Figura 16 - Interfaz permisos al repositorio

Luego de hacerlo se puede seleccionar entre la lista de repositorios el que se desea vincular.

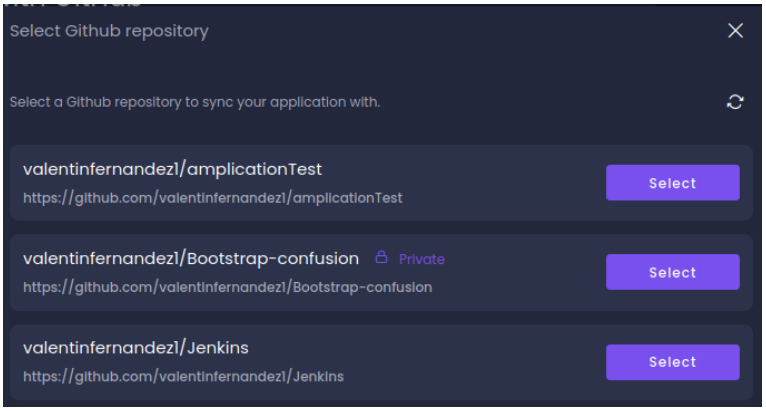


Figura 17 - Interfaz seleccionar repositorio

Subida a GitHub

Una vez que se diseñaron los componentes se utiliza esta funcionalidad para subir el código generado directamente al repositorio Sincronizado.

Lo primero que se destaca de esta funcionalidad es que a medida que se hacen cambios en el proyecto en la barra lateral derecha se muestra el listado de cambios y se permite generar un commit para GitHub de estos cambios.

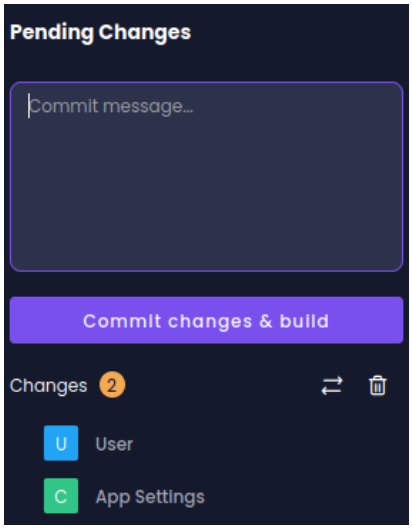


Figura 18 - Interfaz commit al repositorio

Una vez hecho esto en la barra inferior hay un botón que permite hacer un pull request a GitHub. Luego de realizar este pull request el botón despliega un nuevo texto “Open GitHub” que redirige al panel de pull requests del repositorio vinculado, en el cual ya se encuentra el pull request.

Figura 19 - Interfaz botón pull request

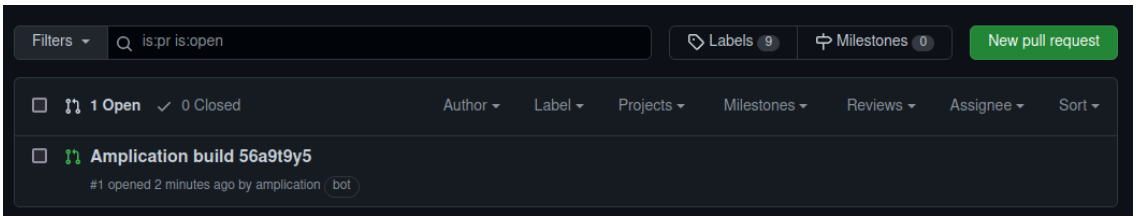


Figura 20 - Interfaz ver pull request en repositorio

2.2) Modelo lógico del Sistema actual

Tras un análisis de la aplicación tenemos el siguiente diagrama de caso de usos y de entidades.

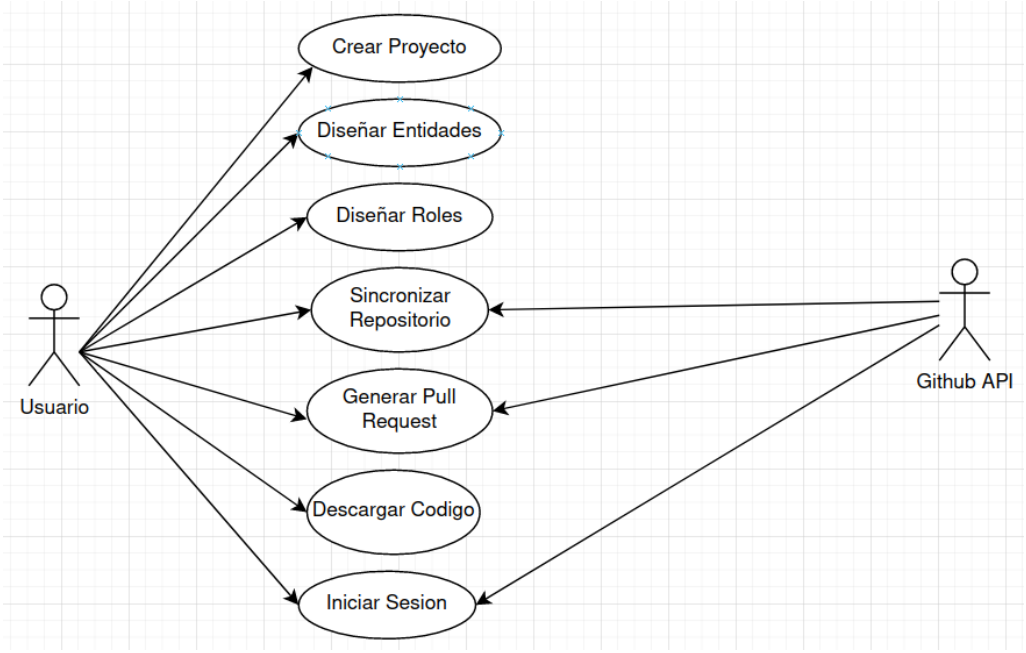


Figura 21 - Diagrama de casos de uso del sistema Amplication

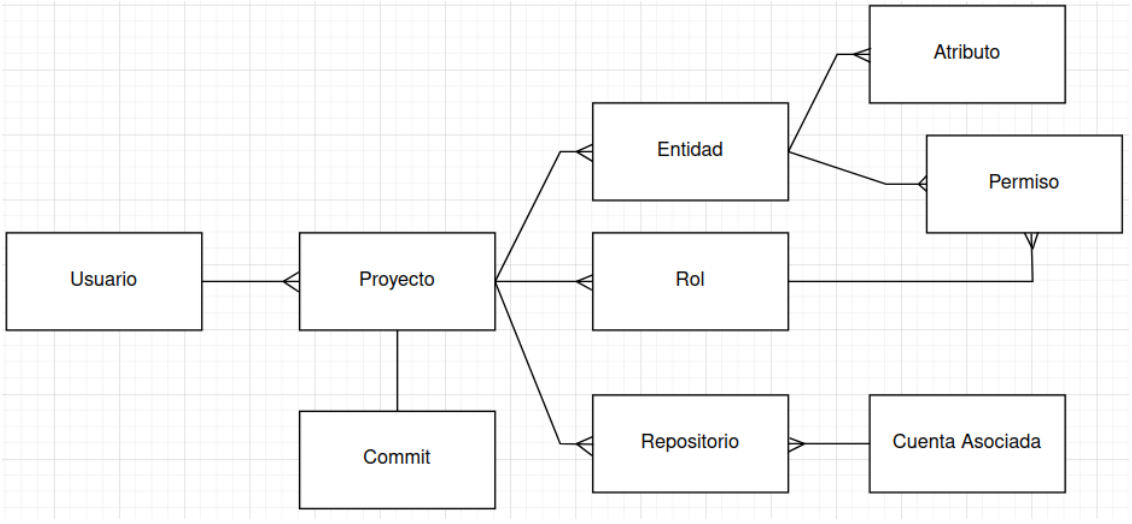


Figura 22 - Diagrama de entidades del sistema Amplication

2.3) Problemas y necesidades detectados en las funciones relevadas en detalle y en su entorno organizacional

Funcionalidad: Iniciar Sesión

Problema 1: Solo se permite crear proyectos si se tiene una cuenta previa en GitHub.

Necesidad 1: Permitir crear una cuenta propia en la herramienta o brindar otras opciones de Login. Ej: Facebook, Google, etc.

Funcionalidad: Crear Proyecto

Problema 1: Si bien se proveen opciones de creación, la opción de creación por schema de Excel es poco intuitiva ya que no se explica como debe ser este archivo de Excel, y por otro lado la creación de un archivo en blanco no permite ninguna configuración inicial por lo que exige indagar en las opciones de la interfaz para encontrar configuraciones importantes de esta.

Necesidad 1: Debe dar una mejor explicación para el proceso de creación y brindar de manera más intuitiva configuraciones básicas del proyecto.

Funcionalidad: Diseño de Entidades

Problema 1: Visualizar las relaciones entre entidades se vuelve confuso y poco intuitivo.

Problema 2: No se permite crear herencias entre entidades por lo que esto lleva a la utilización de malas prácticas dependiendo los requisitos del proyecto.

Necesidad 1: Utilizar una implementación de la interfaz que permita visualizar las relaciones entre entidades de manera más completa.

Problemas y necesidades generales del entorno

Problema 1: La aplicación está limitada a la generación de código específico para NodeJS implementando el framework NestJS.

Problema 2: No hay forma de compartir el proyecto o permitir la colaboración entre individuos en un mismo proyecto, limitando así la versatilidad de la herramienta.

Problema 3: Si bien permite subir el código a la nube brindando así un servicio de infraestructura, la única nube a la que brinda soporte el código generado es a la nube propia de amplication, lo que lo hace inviable para proyectos que necesitan una infraestructura más confiable.

Problema 4: La única base de datos para la que está preparado el código generado es PostgreSQL [8], una base de datos SQL.

Problema 5: El código generado solo es viable en el caso de generar sistemas pequeños ya que a medida que la cantidad de entidades aumenta se vuelve más complejo visualizar las relaciones entre entidades. Se necesita una interfaz más intuitiva de modelado.

Problema 6: En caso de que se quiera aplicar una arquitectura de microservicios en un sistema la única forma de hacerlos es generando cada microservicio por separado y separando los archivos útiles.

Vemto

1) Relevamiento general

1.1) De la organización.

Vemto [2] es un estudio completo y generador de código Laravel.

La herramienta está enfocada en ahorrar tiempo de desarrollo al generar todos los archivos de la aplicación y la API, desde el modelo y los controladores hasta las vistas CRUD.

Vemto genera código que es, al mismo tiempo: limpio, probado (TDD), sencillo y altamente extensible.

La aplicación es de paga, pero tiene prueba gratis por tiempo limitado (7 días).

La sede se encuentra en Araçatuba, Brasil.

1.2) Funciones detectadas a nivel general y relaciones con otros Sistemas y Entidades

Crear proyecto.

Permite crear un nuevo proyecto en el que se podrá configurar el nombre del proyecto, versión de Laravel para generar el código y configuración de la base de datos a utilizar.

Importar proyecto.

Permite importar un proyecto Vemto creado anteriormente.

Modelar esquema.

Permite modelar el esquema de la aplicación a generar, agregando entidades al modelo. En estas se gestionan los atributos y relaciones con otras entidades.

Crear CRUD.

Permite crear y configurar los CRUD que se quieren generar para la aplicación.

Configurar proyecto.

Permite editar las configuraciones del proyecto en general, como por ejemplo, modos de nomenclatura de controladores, estrategia de asignación masiva, rutas, validaciones y autenticación.

También permite editar la configuración de las vistas, cambiar la carpeta y el espacio de nombres para generar los Modelos, los Controladores y los metadatos del proyecto.

Configurar la generación del código.

Permite seleccionar de qué módulos se quiere generar el código, por ejemplo: si se quiere generar los Controladores, Pruebas o Vistas de un módulo específico o no.

Generar el código.

Permite generar el código para el esquema anteriormente modelado y las configuraciones seleccionadas. Este automáticamente se guarda en el dispositivo que se está utilizando, en la carpeta seleccionada en la configuración del proyecto.

Ejecutar proyecto.

Permite ejecutar localmente el proyecto generado en el dispositivo que se está utilizando.

Verificar licencia.

Permite verificar la licencia del producto obtenida para extender el tiempo de uso del software, recibir actualizaciones y soporte. La verificación se realiza a través del correo electrónico ingresado.

Según las funciones relevadas, la aplicación interactúa en todo momento con una entidad:

Usuario: El usuario al que está dirigida la aplicación será un desarrollador de software, diseñador de sistemas, analista funcional, o cualquier individuo que busque generar código.

A continuación, se presenta un diagrama de las relaciones y las interacciones.

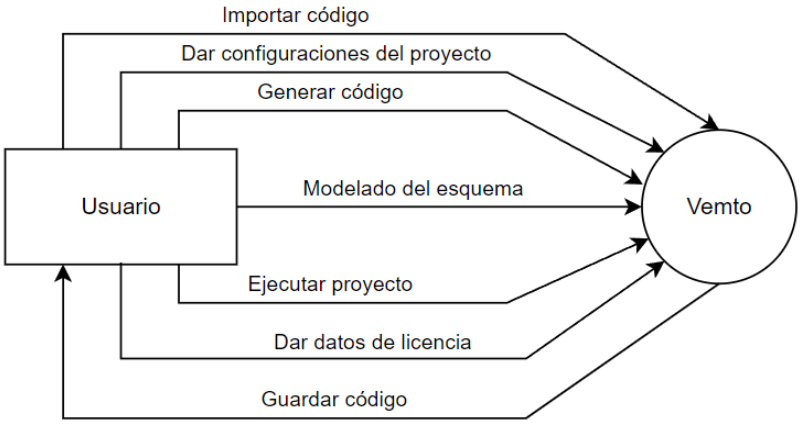


Figura 23 - Diagrama de interacciones del sistema Vemto

1.3) Tecnología de Información.

Al ser un software privado con licencia de uso paga, el código fuente se encuentra oculto por lo que no se ha podido relevar información sobre las tecnologías con las que fue construido. Luego de analizar la documentación de la aplicación se lograron detectar las siguientes tecnologías:

- Electron
- Laravel
- Vue
- GitHub
- PHP
- Composer 2
- NodeJS

Tener en cuenta que Vemto podría utilizar otras tecnologías extra, pero al ser herramienta propietaria no podemos acceder a más información sobre esta.

Utilizando el complemento para Firefox “What Runs” podemos detectar que para su página web utiliza

- Nginx 1.14.0
- Bootstrap
- CloudFlare

2) Relevamiento detallado y análisis del Sistema

2.1) Detalle, explicación y documentación detallada de todas las funciones seleccionadas.

Crear proyecto

Permite crear un nuevo proyecto vacío donde vamos a generar el código de la aplicación.

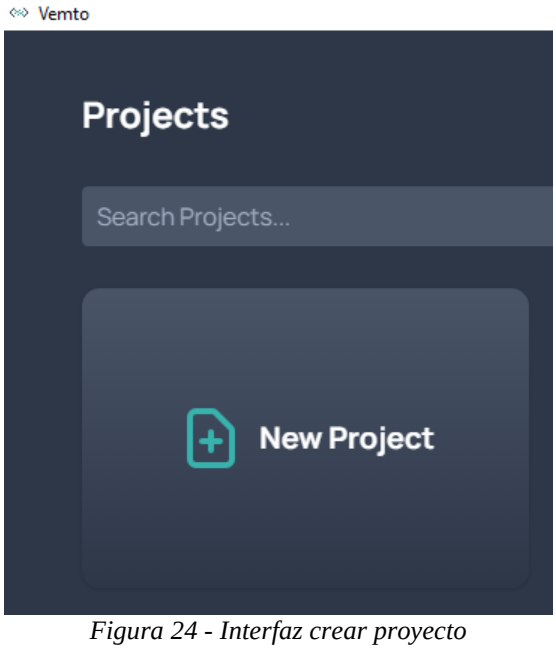


Figura 24 - Interfaz crear proyecto

En él, se podrá configurar el nombre del proyecto, configuración del entorno, así como versión de laravel para generar el código y configuración de las vistas.

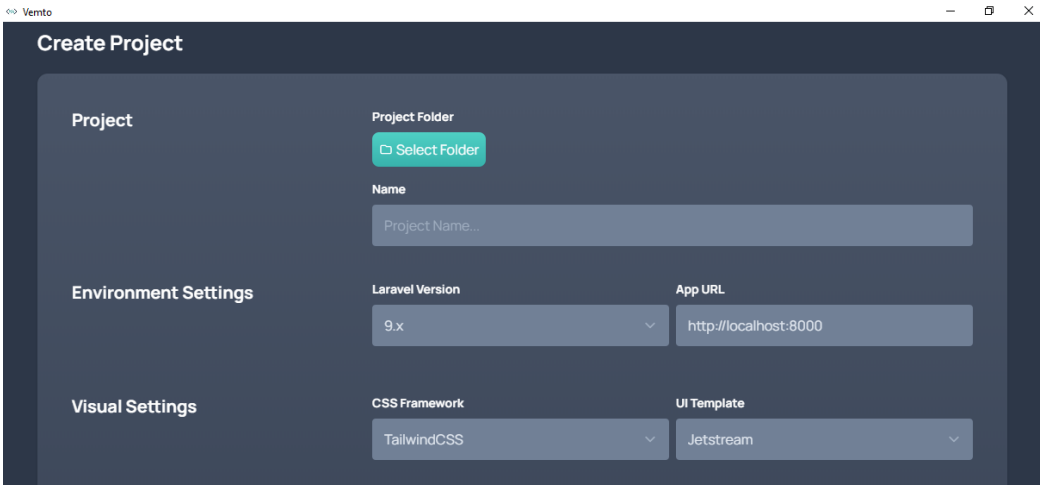


Figura 25 - Interfaz de configuraciones iniciales del proyecto

También se configura la conexión a la base de datos que se va a utilizar.

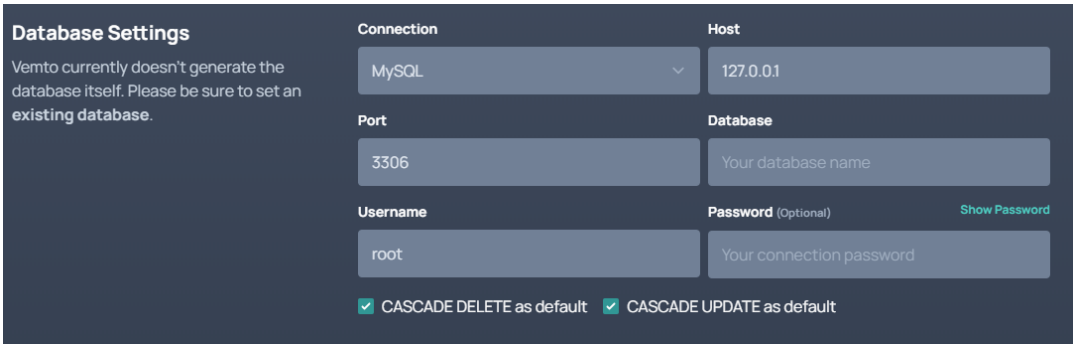


Figura 26 - Interfaz de conexión a la base de datos

Importar proyecto.

Permite importar un proyecto Vemto creado anteriormente para reutilizar esquemas planteados y modificarlo.

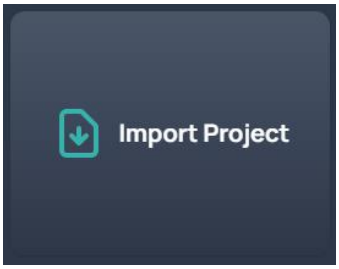


Figura 27 - Interfaz botón importar proyecto

Modelar esquema.

Permite modelar el esquema de la aplicación a generar. En este, automáticamente se genera la entidad User.

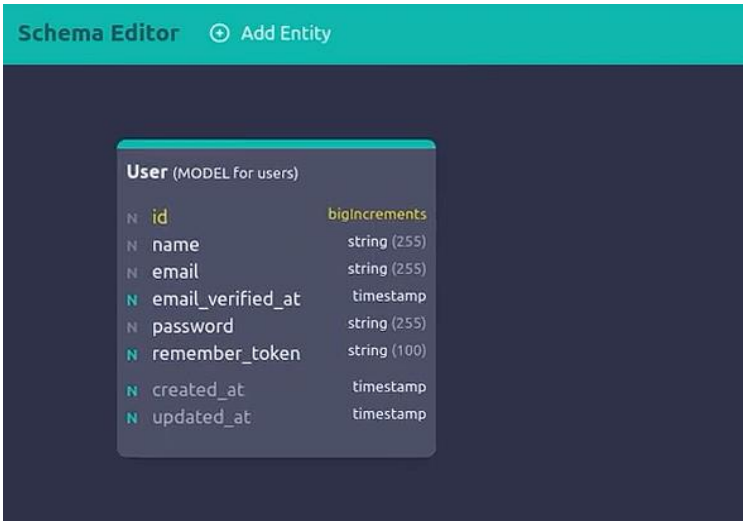


Figura 28 - Interfaz entidad User

Una ventana desplegable permite gestionar los atributos y relaciones con otras entidades.

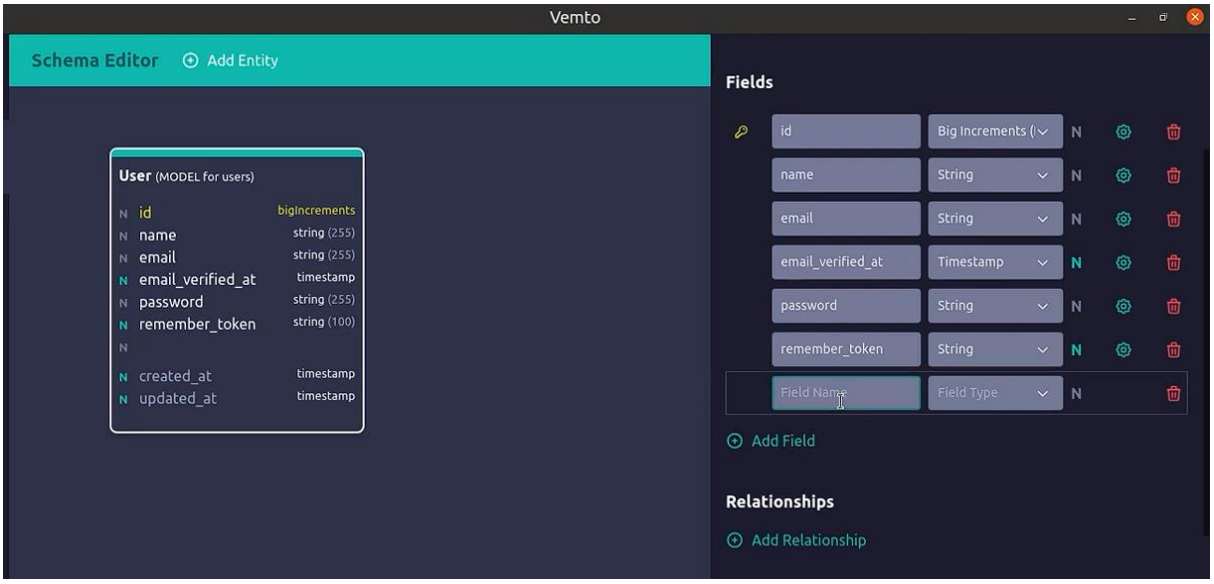


Figura 29 - Interfaz gestión de atributos y relaciones

También es posible generar otras entidades para completar el modelado de la aplicación a generar.

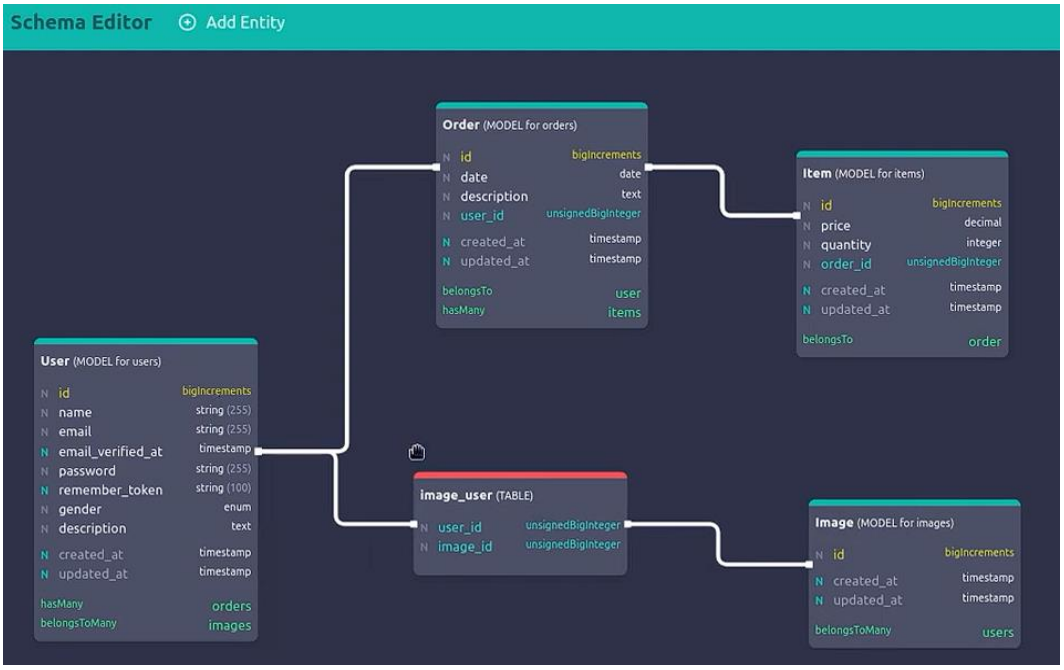


Figura 30 - Interfaz modelado del esquema

Crear CRUD.

Permite crear y configurar los CRUD que se quieren generar para la aplicación.

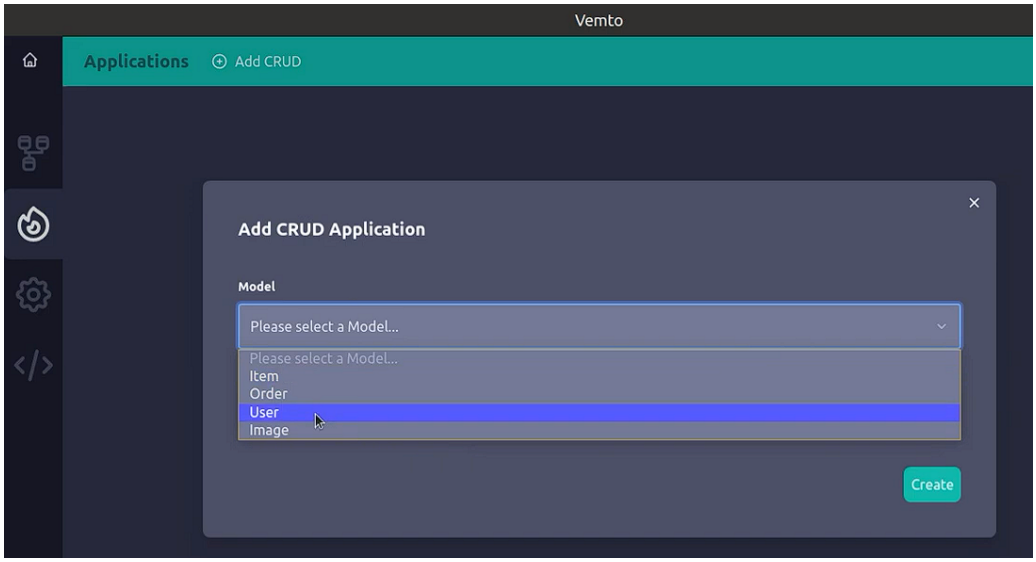


Figura 31 - Interfaz de gestión del CRUD

En este caso seleccionamos User, y podremos editar los formularios, validaciones y otras configuraciones.

En el caso de los formularios permite configurar las vistas, sus valores por defecto y eventos.

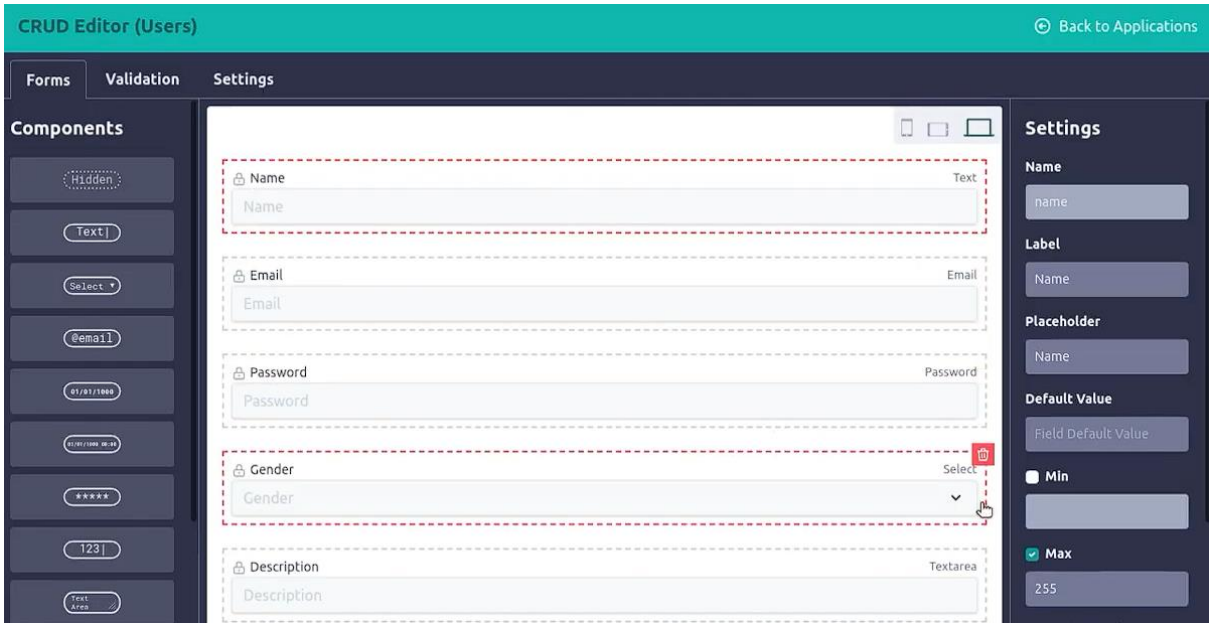


Figura 32 - Interfaz de configuración de formularios

Permite también configurar las validaciones para los CRUD.

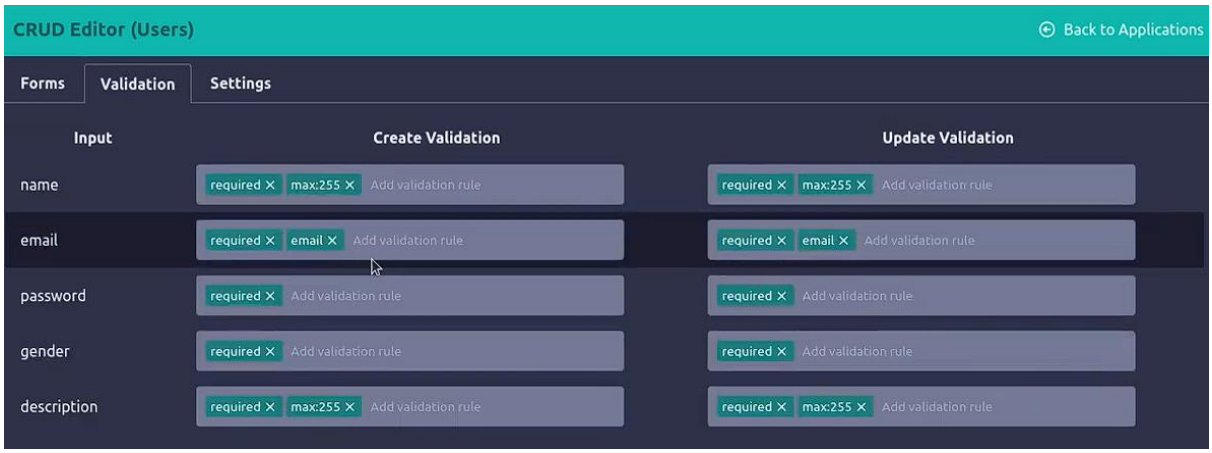


Figura 33 - Interfaz de configuración de validaciones

Configurar proyecto.

Permite editar las configuraciones del proyecto en general, como por ejemplo, modos de nomenclatura de controladores, estrategia de asignación masiva, rutas, validaciones y autenticación.

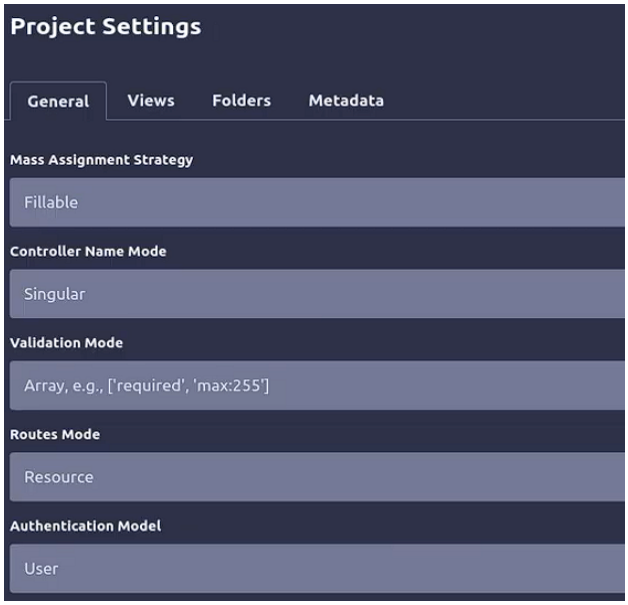


Figura 34 - Interfaz de configuraciones del proyecto

También permite editar la configuración de las vistas.

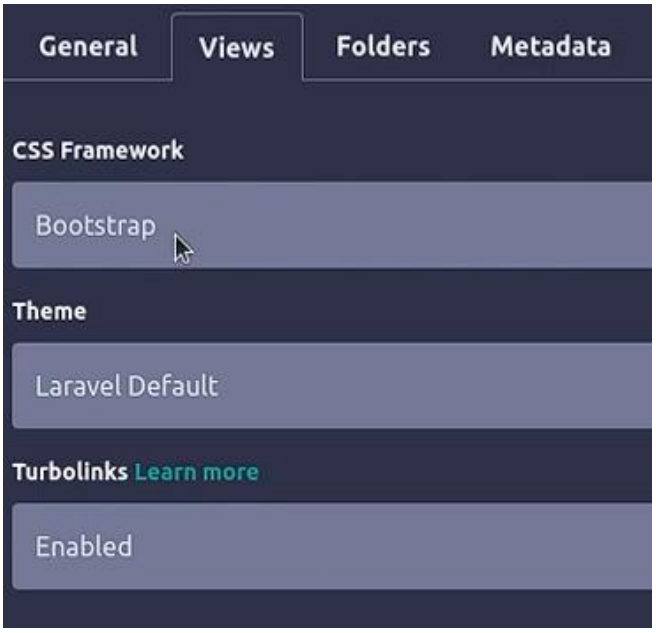


Figura 35 - Interfaz de configuración de vistas

Cambiar la carpeta y nombres para generar los Modelos y los Controladores.

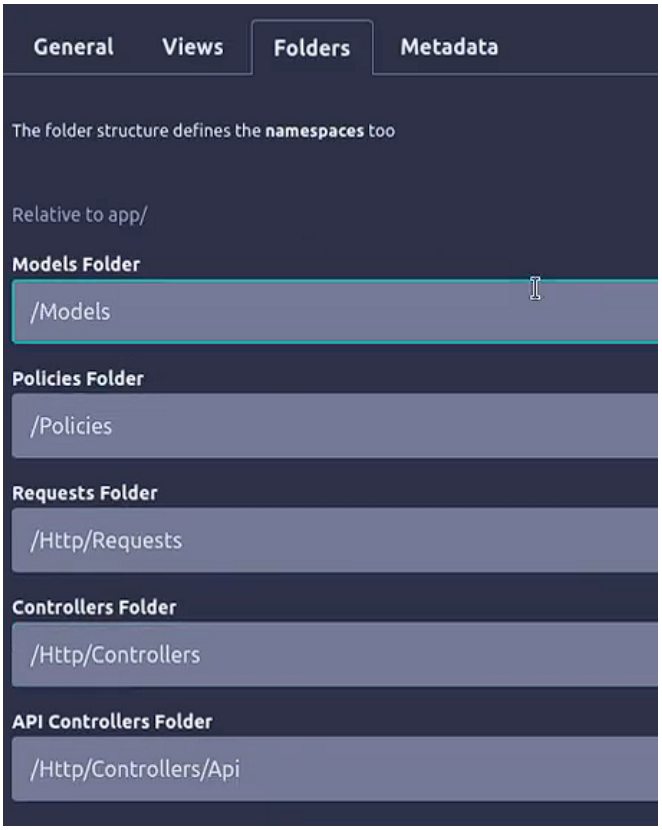


Figura 36 - Interfaz de configuración de estructura de carpetas

Configurar la generación del código.

Permite, específicamente para cada módulo, seleccionar qué sección de código se quiere generar. Por ejemplo: si se quiere generar los Controladores o Vistas de un módulo específico o no.

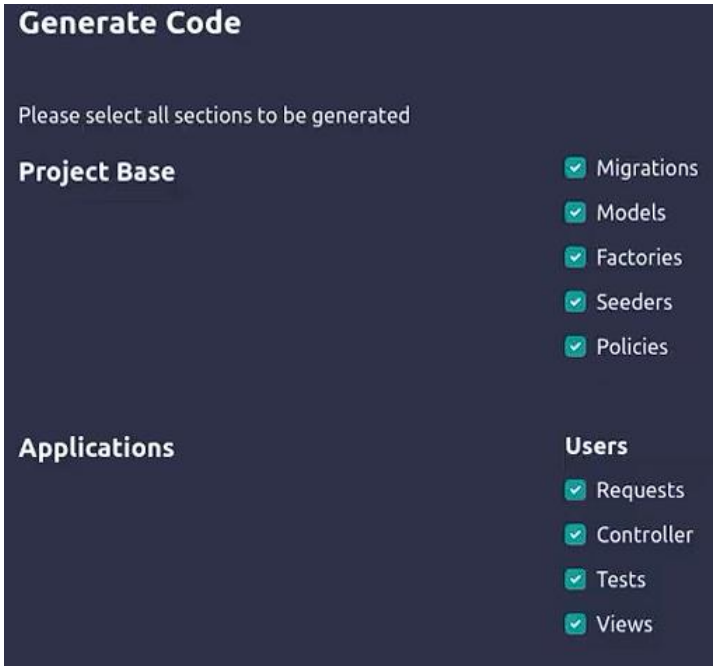


Figura 37 - Interfaz configuración de generación de código

Generar el código.

Esta función permite generar el código para el esquema anteriormente modelado y las configuraciones seleccionadas.

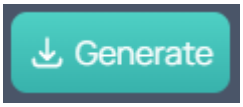


Figura 38 - Interfaz botón generar código

Automáticamente se guarda el código en el dispositivo que se está utilizando, en la carpeta seleccionada en la configuración del proyecto.

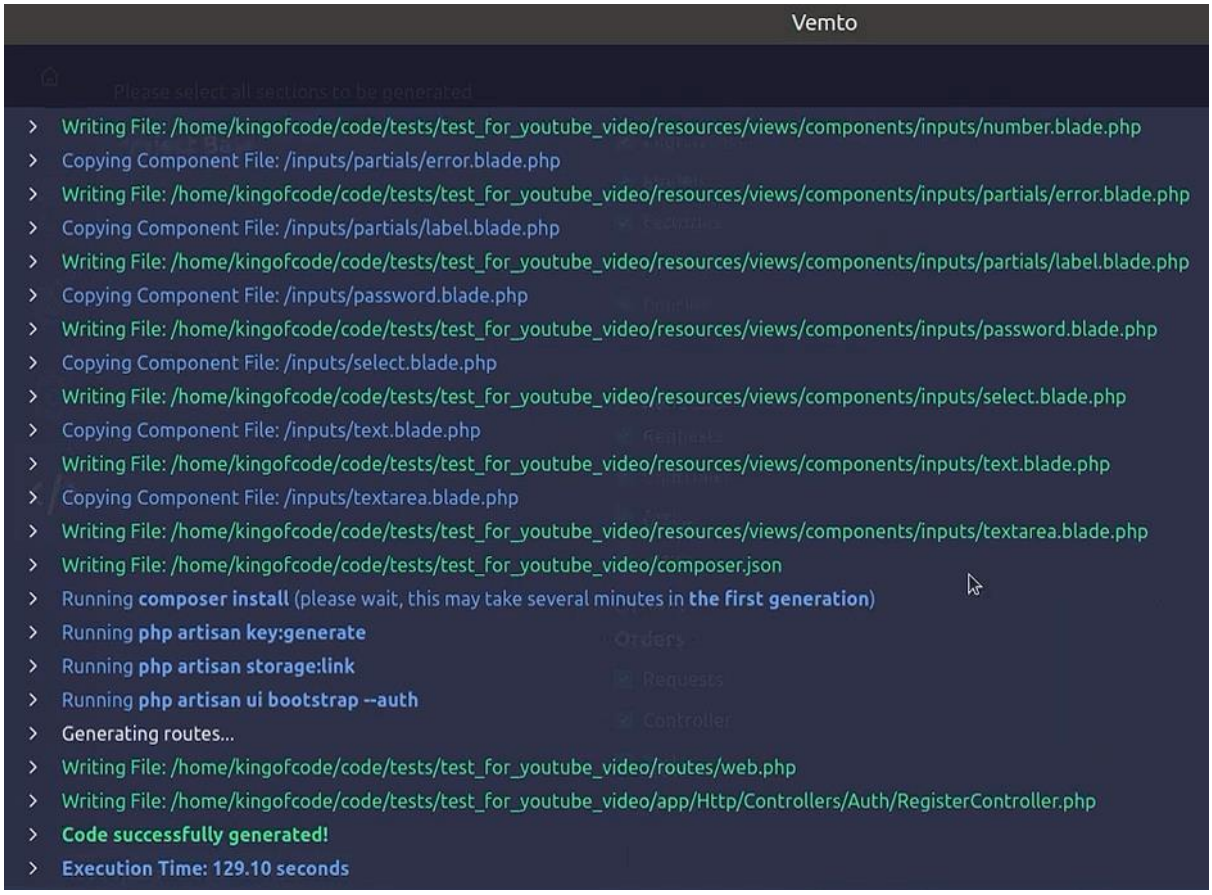


Figura 39 - Interfaz de generación del código

Ejecutar proyecto.

Permite ejecutar localmente el proyecto generado en el dispositivo.

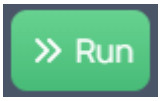


Figura 40 - Interfaz botón de ejecución del proyecto

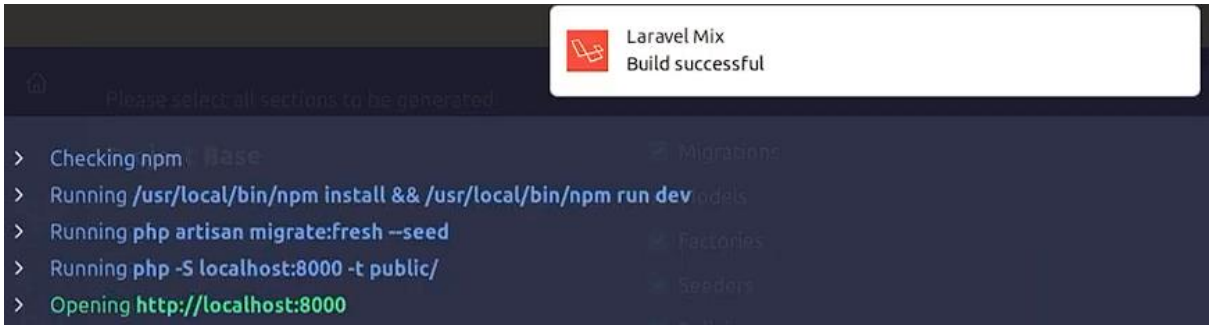


Figura 41 - Interfaz de ejecución del proyecto

El proyecto ejecutado se abre en un navegador para su visualización.

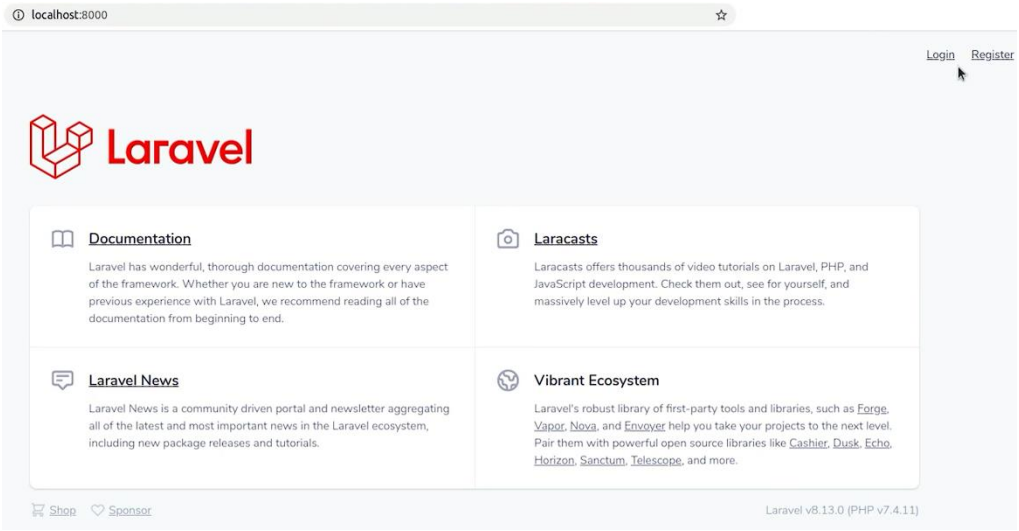


Figura 42 - Interfaz proyecto ejecutado en el dispositivo

Verificar licencia.

Permite verificar la licencia del producto obtenida para extender el tiempo de uso del software, recibir actualizaciones y soporte. La verificación se realiza a través del correo electrónico ingresado.

Figura 43 - Interfaz verificar licencia

2.2) Modelo lógico del Sistema actual

Detallamos el modelo lógico del sistema a través de diagrama de caso de uso.

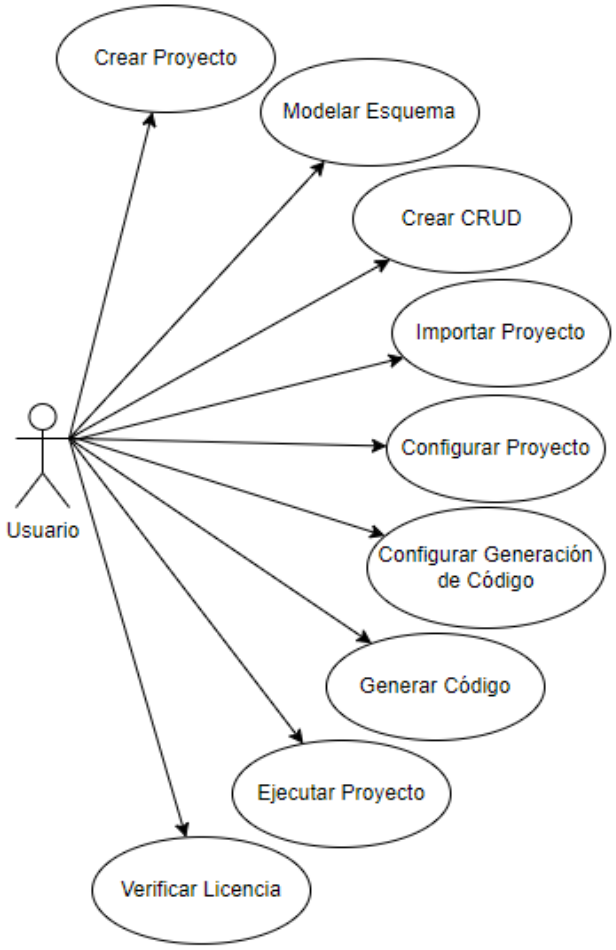


Figura 44 - Diagrama de casos de uso del sistema Vemto

2.3) Problemas y necesidades detectados en las funciones relevadas en detalle y en su entorno organizacional

Problemas y necesidades en general:

Problema 1: La aplicación se ejecuta localmente en el dispositivo y no permite iniciar sesión para guardar configuraciones o proyectos.

Problema 2: No es posible compartir el proyecto o trabajar colaborativamente con un equipo de desarrollo.

Problema 3: No maneja roles dentro de la aplicación por lo que cualquier usuario puede editar el proyecto y generar código fuente.

Problema 4: Los modelos del código son generados únicamente para bases de datos SQL.

Problema 5: En caso de querer aplicar arquitectura de microservicios en el sistema a generar, la única forma de hacerlos es generando cada microservicio por separado y separando los archivos útiles.

Necesidad 1: Que el usuario pueda crear una cuenta personal en la aplicación donde guardar configuraciones y proyectos para reutilizar en otros dispositivos.

Necesidad 2: Permitir exportar el código directamente a GitHub u otro sistema de control de versiones.

Necesidad 3: Permitir generar código para bases de datos NoSQL.

Funcionalidad: Generar código.

Problema 1: Generación del código limitada a un solo lenguaje/framework (PHP/Laravel).

Necesidad 1: Permitir generar código para más lenguajes dándole flexibilidad al usuario.

Funcionalidad: Importar proyecto.

Problema 1: Solo permite importar el proyecto localmente desde los archivos fuentes.

Necesidad 1: Permitir importar el proyecto desde un repositorio externo.

JHipster

1) Relevamiento general

1.1) De la organización

JHipster [3] es un generador de aplicaciones gratuito y de código abierto que se utiliza para desarrollar rápidamente aplicaciones web modernas y microservicios utilizando Angular o React y Spring Framework.

También tiene soporte para aplicaciones móviles utilizando ionic y React Native, para despliegue utiliza herramientas en la nube como son Docker [10] y kubernetes [11]. Por último tiene soporte para plataformas Cloud como AWS, Azure, Google Cloud, etc.

Misión

Su misión es generar aplicaciones y arquitecturas de microservicio modernas y completas que cumplan con los siguientes objetivos:

- Tecnología del lado del servidor robusta, de alto rendimiento y con una excelente cobertura de pruebas.

- Una interfaz moderna, responsiva y elegante utilizando las mejores tecnologías del mercado.
- Un flujo de trabajo poderoso para construir aplicaciones con Webpack, Maven o incluso Gradle.
- Una arquitectura de microservicios resiliente que siga principios de gestión nativos en la nube.
- Infraestructura como código para desplegar de manera fácil y rápida en la nube.

Es una herramienta muy usada entre grandes compañías como lo son Google, adobe, siemens, Carrefour, Heroku, hsbc, etc.

1.2) Funciones detectadas a nivel general y relaciones con otros Sistemas y Entidades

Crear proyecto

Permite crear un nuevo proyecto mediante la selección de parámetros que sirve de esquema inicial para la posterior definición de entidades. En esta función se detalla el tipo de proyecto, el lenguaje a utilizar, la base de datos a utilizar, entre otros.

Definir entidades

Permite definir las entidades a utilizar en el proyecto (o en el microservicio, en el caso de crear un proyecto distribuido) mediante JDL Studio, una interfaz gráfica guiada por código.

Descargar archivo de entidades (JDL)

Permite la descarga del archivo de especificación .jdl que contiene la definición de las entidades.

Generar código

Esta acción realiza la generación automática de código según los parámetros definidos en la creación del proyecto a partir de las entidades definidas. La entrada a este proceso es el archivo .jdl y la salida es el código generado automáticamente.

Generar archivos de configuración de despliegue

Esta acción genera los archivos de configuración necesarios para el despliegue de nuestro proyecto en el proveedor cloud especificado. Luego, el usuario puede aplicar los archivos para realizar el despliegue.

Generar archivos de CI/CD

Esta acción genera los archivos necesarios de CI/CD (Integración Continua/Despliegue continuo) con algunos parámetros a definir.

JHipster solo se comunica con el usuario que utiliza la herramienta y no tiene ninguna interacción con otros sistemas, a continuación se presenta un diagrama de esta interacción.

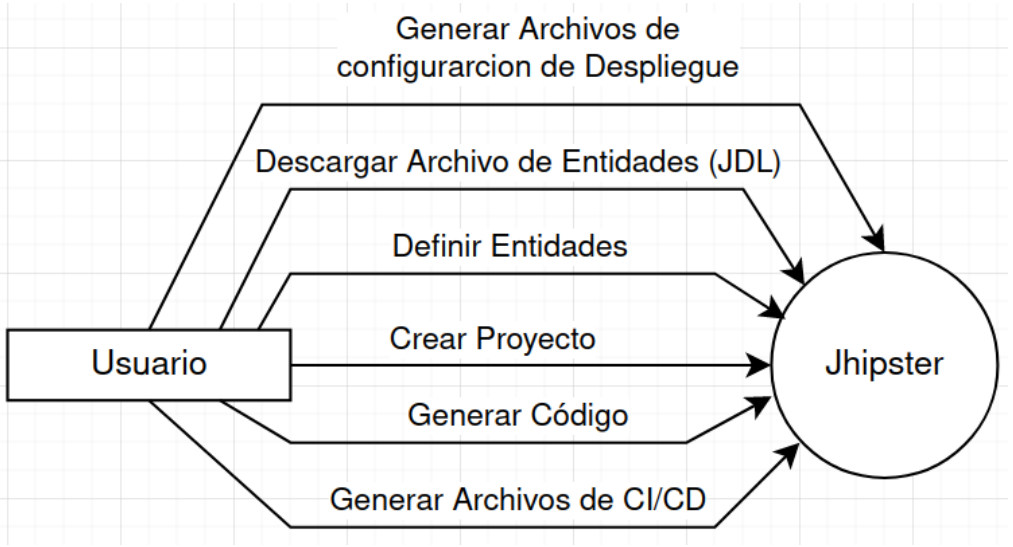


Figura 45 - Diagrama de interacción del sistema JHipster

1.3) Tecnología de Información

Utilizando el plugin de Firefox “What Runs” podemos detectar que las tecnologías que utiliza son:

- Angular
- JQuery
- Bootstrap
- Es ejecutado en un servidor Ubuntu
- El servidor corre Apache
- CodeMirror
- GitHub

También al analizar el GitHub de la herramienta podemos encontrar que está codificada utilizando:

- Java
- Javascript
- Vue
- Typescript
- AWS
- NodeJs
- Docker

A continuación, también enumeramos las tecnologías que permite generar código, ya que JHipster cuenta con una amplia variedad de tecnologías que puede generar, se mencionan las principales.

Para el lado del Cliente:

- React
- Angular
- Vue
- Bootstrap
- Jest

Para el lado del servidor:

- Para la API:
  - SpringBoot
  - Micronaut
- Bases de Datos

- MySQL
- MongoDB
- Redis
- Oracle
- PostgreSQL

- Herramientas de Despliegue:
  - Docker
  - Kubernetes
  - Google Cloud
  - AWS
  - Heroku
- Opciones de integración Continua
  - Jenkins
  - GitLab
  - GitHub Workflows
  - CircleCI

2) Relevamiento detallado y análisis del Sistema

2.1) Detalle, explicación y documentación detallada de todas las funciones seleccionadas.

Crear Proyecto

Para crear un proyecto, una vez que la herramienta JHipster esté instalada, se debe ejecutar el comando JHipster para iniciar la creación del proyecto. El comando muestra la

dirección en la que el proyecto será creado (carpeta donde se ejecutó el comando JHipster) y los siguientes parámetros a completar:

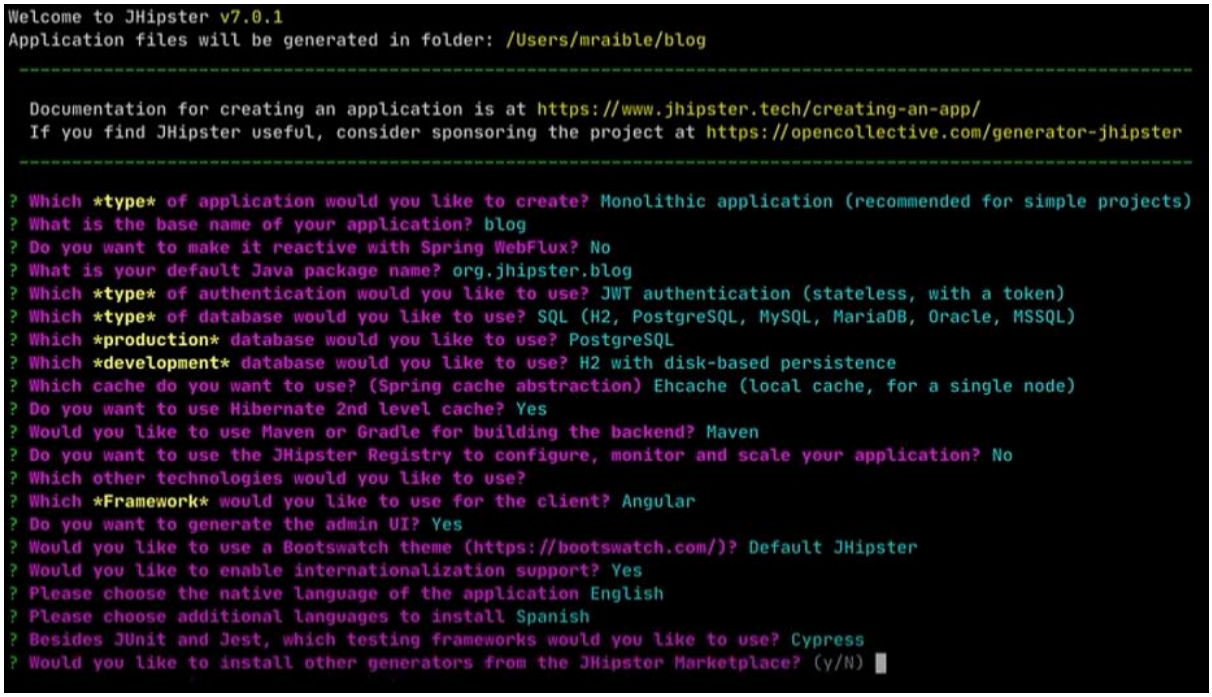


Figura 46 - Interfaz de creación de proyecto

A continuación, se describen los parámetros posibles del proyecto y las opciones disponibles para cada uno de ellos:

- Tipo de aplicación. Opciones: Aplicación monolítica/Aplicación de microservicios.
- Nombre de la aplicación.
- Aplicación reactiva con Spring WebFlux. Opciones. Si/No
- Nombre del paquete Java por defecto.
- Tipo de autenticación a utilizar. Opciones: JWT/OAuth 2.0/HTTP Session.

- Tipo de base de datos. Opciones: SQL/MongoDB/Cassandra/Neo4j/Sin base de datos. Si se elige utilizar alguna de las bases de datos mencionadas, se observan los siguientes parámetros:
    - Base de datos a utilizar en producción. Opciones: PostgreSQL/MySQL/ MariaDB/Oracle/Microsoft SQL Server.
    - Base de datos a utilizar en desarrollo. Opciones: PostgreSQL/H2 con persistencia en memoria/H2 con persistencia en disco.

Los tipos de bases de datos mostrados para producción y desarrollo dependen del tipo de base de datos elegido.

- Caché a utilizar. Opciones: Ehcache/Caffeine/Hazelcast/Infinispan/Memcached/Redis.
- Hibernate como caché de segundo nivel. Opciones: Si/No
- Tecnología para la generación del empaquetado en el backend. Opciones: Maven/Gradle.
- Registro de JHipster para configurar, monitorear y escalar la aplicación. Opciones: Si/No.
- Otras tecnologías a utilizar. Opciones: Elasticsearch como motor de búsqueda/Websockets con Spring WebSocket/Apache Kafka como agente de comunicación asíncrona/OpenAPI Generator.
- Framework a utilizar para el cliente(frontend).Opciones: Angular/React/Vue/Sin cliente.
- Generar interfaz gráfica de administración. Opciones: Si/No.

- Tema de Bootswatch. Opciones: Default JHipster/Cerulean/Cosmo/Cyborg/Darkly/Flatly/Journal. Cada uno de los temas se encuentra detallado en https://bootswatch.com.
- Habilitar soporte para internacionalización. Opciones: Si/No.
- Lenguaje de aplicación. Opciones: Inglés/Estonio/Farsi/Finés/Francés/Alemán/Español.
- Lenguajes adicionales a instalar. Opciones: Todos los idiomas.
- Frameworks de prueba a utilizar además de JUnit y Jest. Opciones: Cypress/Protractor/Gatling/Cucumber
- Instalar otros generadores mediante JHipster Marketplace. Opciones: Si/N

Al completar cada parámetro, JHipster crea automáticamente el esqueleto de la aplicación según lo indicado. Sin embargo, el proyecto se crea sin entidades, ya que las mismas se especifican en otra etapa.

Definir entidades

La definición de entidades permite la especificación de las entidades de nuestro proyecto. Para ello, se utiliza JDL-Studio, una herramienta de interfaz gráfica guiada por código que permite definir las distintas clases y sus relaciones a través de JDL (JHipster Domain Language), el lenguaje específico de dominio de JHipster. Al ingresar a JDL Studio, se observa lo siguiente:

Figura 47 - Interfaz de definición de entidades

A la izquierda, se observa el código escrito en JDL. En el mismo, se detallan las entidades (en este caso Blog, Post y Tag) con sus respectivos atributos. Además, para cada atributo se especifica un tipo de dato y se puede agregar una validación como, por ejemplo, la longitud mínima requerida para un campo String.

Luego, las relaciones se definen mediante la palabra relationship seguido de la multiplicidad de la relación (relationship ManyToOne, por ejemplo). Dentro de cada bloque relationship, se indican todas las relaciones del mismo tipo. En este caso, existen dos relaciones Muchos a Uno:

- Relación Blog a User
- Relación Post a Blog.

Descargar archivo de entidades (JDL)

Esta acción nos permite descargar el archivo .jdl que contiene toda la especificación de entidades en el lenguaje específico de dominio de JHipster. Para descargarlo, debemos hacer click en el quinto elemento (flecha apuntando hacia abajo) de la barra de elementos de JDL Studio:

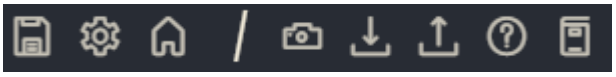


Figura 48 - Interfaz descarga archivo de entidades

Generar código

Esta acción realiza la generación automática de código según los parámetros definidos en la creación del proyecto a partir de las entidades definidas. Para generar el código de un archivo .jdl para un determinado proyecto, se debe realizar lo siguiente:

1. Abrir una consola de comandos en la carpeta de proyectos.
2. Copiar el archivo .jdl a la carpeta de proyectos.
3. Ejecutar el archivo JHipster jdl <nombre-archivo>.jdl

Luego, se observa el código generado.

Generar archivos de configuración de despliegue

Esta acción genera los archivos de configuración necesarios para el despliegue de nuestro proyecto en el proveedor cloud especificado. Para generar los archivos de configuración de despliegue, se debe ejecutar el comando JHipster <proveedor-cloud> en la carpeta del proyecto. A continuación, se muestra un ejemplo:

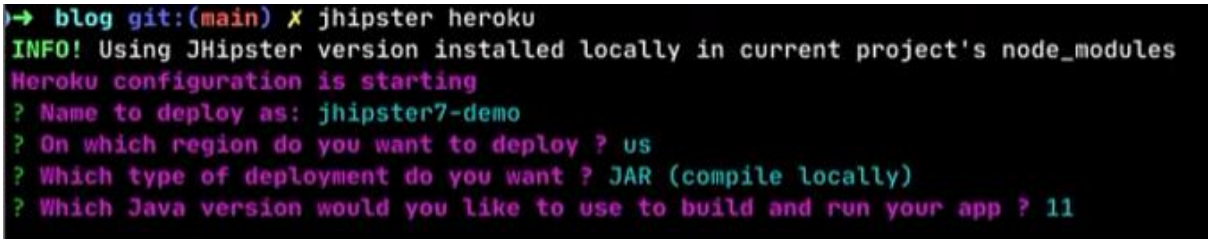


Figura 49 - Interfaz generar archivos configuración de despliegue

En este ejemplo, se generan los archivos de configuración para el proveedor cloud Heroku. Podemos observar los siguientes parámetros:

- Nombre del despliegue.
- Región de despliegue. Opciones: US(Estados Unidos)/EU(Europa)
- Tipo de despliegue. Opciones: JAR(compilar localmente)/Git(Compilar en Heroku).
- Versión de Java. Opciones: 1.8/11/12/13/14

Cada proveedor cloud tiene su configuración propia. Los proveedores cloud disponibles son:

- Amazon Web Services
- Microsoft Azure
- Google Cloud
- Cloud Captain

Generar archivos de CI/CD

Esta acción genera los archivos necesarios de CI/CD (Integración Continua/Despliegue continuo) con algunos parámetros a definir. Para generar los archivos de CI/CD, se debe ejecutar el comando JHipster ci-cd en la carpeta del proyecto. A continuación, se muestra un ejemplo:

Figura 50 - Interfaz generar archivos CI/CD

En este ejemplo se generan los archivos de ci-cd para GitHub Actions [12] y su integración con Heroku. Los parámetros a configurar son los siguientes:

- Pipeline de CI/CD a generar. Opciones:Github Actions/Jenkins/Azure/GitLab/Travis/Circle.
- Tareas o integraciones a incluir. Opciones:
  - Desplegar a Artifactory
  - Analizar código con Sonar
  - Empaquetar y publicar imagen con Docker
  - Escanear vulnerabilidades de seguridad con Snyk.
  - Desplegar a Heroku.
  - Habilitar el dashboard de Cypress.
- Nombre de la aplicación Heroku. En caso de haber generado los archivos de configuración de despliegue previamente, este campo se autocompleta.

2.2) Modelo lógico del Sistema actual

Detallamos el modelo lógico del sistema a través de diagrama de caso de uso.

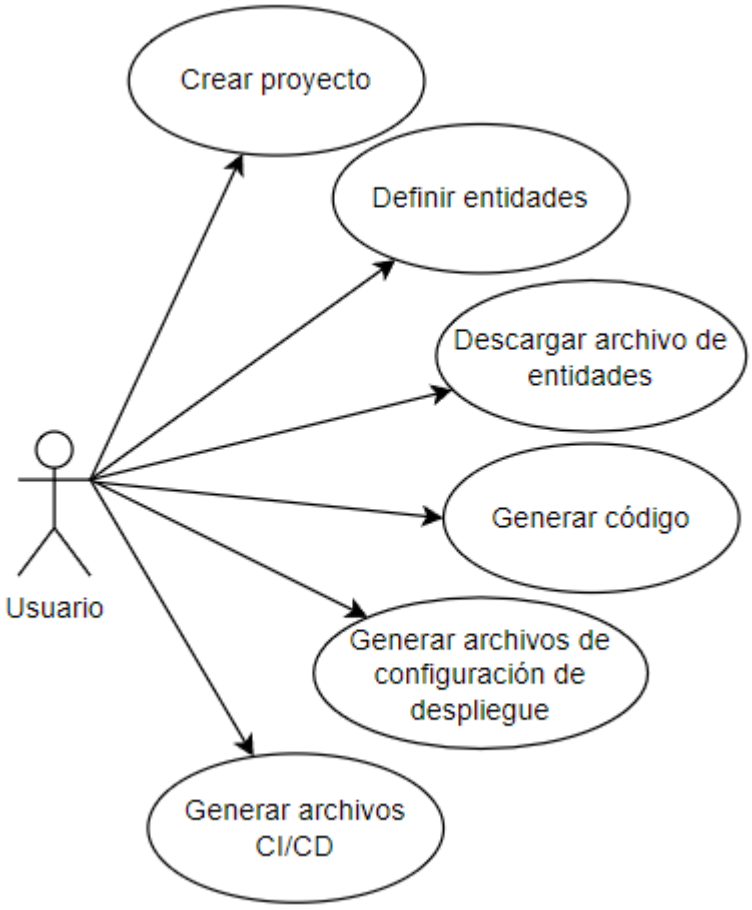


Figura 51 - Diagrama de casos de uso del sistema JHipster

2.3) Problemas y necesidades detectados en las funciones relevadas en detalle y en su entorno organizacional

Problemas asociados a la generación de código

JHipster tiene un grave problema con respecto a la generación de código y es que está solamente orientado a Java. JHipster no permite generar microservicios en distintos lenguajes (Ruby on Rails, por ejemplo). Este es un grave problema ya que, si el desarrollador desea obtener el código de un microservicio para otro lenguaje distinto de Java, no puede hacerlo.

Problemas asociados a la definición de entidades

Con respecto a la definición de entidades, JHipster tiene dos problemas:

- Definición de entidades a partir de JDL, un lenguaje específico de JHipster. Las entidades pueden ser solamente definidas a través del lenguaje específico de dominio JDL. Esto es un grave problema, ya que no sigue ningún estándar y el desarrollador debe aprender su funcionamiento para definir entidades.
- Necesidad de descarga del archivo de entidades para la generación automática. Para realizar la generación automática basada en las entidades definidas, el usuario debe descargarse el archivo y ejecutar un comando en una interfaz de línea de comandos. Esta forma es incómoda para el desarrollador, quien seguramente preferiría apretar un botón y generar el código en vez de manejar archivos.

Problemas asociados al uso de la herramienta en general

El principal problema de la herramienta es su usabilidad. JHipster solamente puede ser utilizado mediante una interfaz de línea de comandos. Para realizar la generación automática, ya sea de código o de infraestructura, el usuario debe primero instalar la herramienta JHipster en el PATH de su computadora y luego aprender los comandos de la misma.

Fogg

1) Relevamiento General

1.1) De la Organización

Fogg [4] es un generador de código que estandariza la configuración de la infraestructura y facilita el manejo de espacios de trabajo de Terraform [9].

Está diseñado bajo los principios de transparencia y Convención por sobre configuración.

Convención por sobre Configuración: En vez de enfocarse en las variaciones que se pueden hacer en la configuración que terminan llevando a soluciones muy similares, Fogg se concentra en estandarizar ciertos principios para así organizar el código Terraform en base a estos

Transparencia: Se encarga de generar el código, una vez generado el usuario puede ver todos los detalles de este y encargarse de administrar la infraestructura.

Otro aspecto Importante de Fogg es que se enfoca en mantener las mejores prácticas utilizadas actualmente en el mercado para Terraform para así reducir el riesgo operacional y mejorar la colaboración.

1.2) Funciones detectadas a nivel general y relaciones con otros Sistemas y Entidades

Inicializar Fogg.

Inicializar un nuevo repositorio para usar Fogg

Crear Componente.

Modificar archivo de configuración fogg.yml para realizar cambios en la estructura de la infraestructura

Aplicar Cambios.

Aplicar modelo Definido en el archivo de configuración fogg.yml y generar código.

Al ser una herramienta de consola Fogg sólo interactúa con el usuario que utiliza la consola de comandos y con el repositorio local de Git. A continuación, se observa un diagrama de estas interacciones.

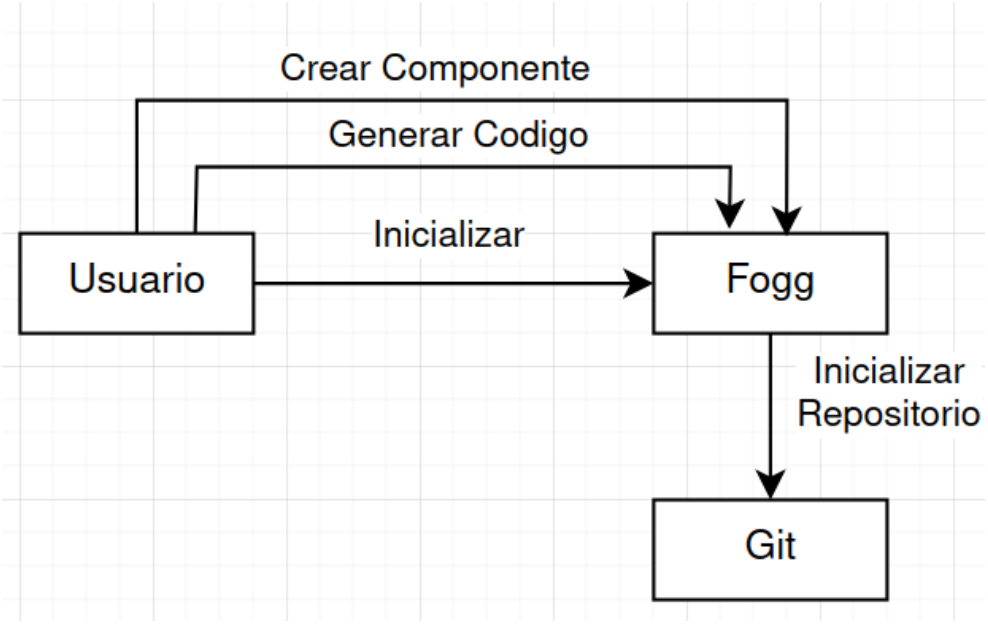


Figura 52 - Diagrama de interacción del sistema Fogg

1.3) Tecnología de Información

Ya que la herramienta no tiene interfaz gráfica, sino que es controlada completamente desde la consola de comandos las tecnologías que utiliza no son muchas. Estas son:

- Terraform: Para la definición de la infraestructura como código
- Go: Como principal lenguaje de programación utilizado

- GitHub: Para el versionado de código
- AWS: Como plataforma en la nube para implementar la infraestructura
- Git: Para el versionado de código local
- Brew: Herramienta de gestión de dependencias.

2) Relevamiento detallado y análisis del Sistema

2.1) Detalle, explicación y documentación detallada de todas las funciones seleccionadas.

Inicializar Fogg.

Para inicializar Fogg primero se debe generar un git en la carpeta en la que se desea utilizar Fogg.

```
mkdir terraform_project
cd terraform_project
git init
```

Figura 53 - Inicializar Fogg

Una vez en la carpeta en donde se piensa crear el proyecto de Fogg hay que ejecutar el comando Fogg init.

```
$ fogg init
fogg init
project name?: acme
aws region?: us-west-2
infra bucket name?: acme-infra
infra dynamo table name?: acme-dynamo
auth profile?: acme-auth
owner?: infra@acme.example
```

Figura 54 - Crear proyecto

En este hay que completar algunas preguntas que permiten generar el archivo fogg.yml. Este archivo es el que aloja toda la configuración necesaria para la generación de código.

Una vez se responden las preguntas se crea el archivo de configuración fogg.yml que tiene el siguiente código.

```yaml
defaults:
    backend:
        bucket: acme-infra
        dynamodb_table: acme-dynamo
        profile: acme-auth
        region: us-west-2
    owner: infra@acme.example
    project: acme
    providers:
        aws:
            account_id: "000"
            profile: acme-auth
            region: us-west-2
            version: 1.27.0
    terraform_version: 0.12.5
accounts: {}
envs:
    staging: {}
modules: {}
version: 2
```

Figura 55 - Archivo de configuración fogg.yml

Crear Componente.

Dentro del archivo de configuración fogg.yml se crean los componentes que necesita la configuración de Terraform a definir. A continuación, se listan los distintos componentes que se pueden crear dentro de su propio staging environment con su respectivo código.

```
    terraform_version: 0.12.5
accounts: {}
envs:
    staging:
        components:
            vpc:
                module_source: "github.com/scholzj/terraform-aws-vpc"
modules: {}
version: 2
```

Figura 56 - vpc component

```
envs:
    staging:
        components:
            vpc:
                module_source: "github.com/scholzj/terraform-aws-vpc"
            database: {}
modules: {}
version: 2
```

Figura 57 - database component

```
            vpc:
                module_source: "github.com/scholzj/terraform-aws-vpc"
            database: {}
            server: {}
modules: {}
version: 2
```

Figura 58 - server component

Aplicar Cambios.

Genera un código nuevo a partir de los cambios realizados al archivo de configuración base fogg.yml. Esto se hace utilizando el comando Fogg apply. Este muestra una vez ejecutado el árbol con todas las carpetas generadas.

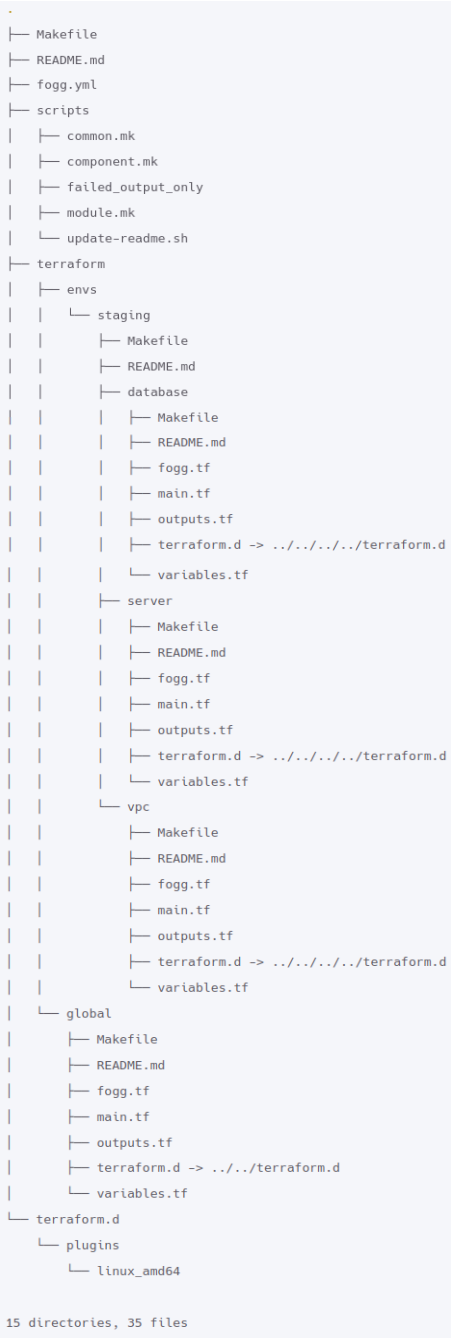


Figura 59 - Árbol con las carpetas generadas

Automáticamente se crearon todo lo necesario para la infraestructura definida en el archivo de configuración.

2.2) Modelo lógico del Sistema actual

Detallamos el modelo lógico del sistema a través de diagrama de caso de uso.

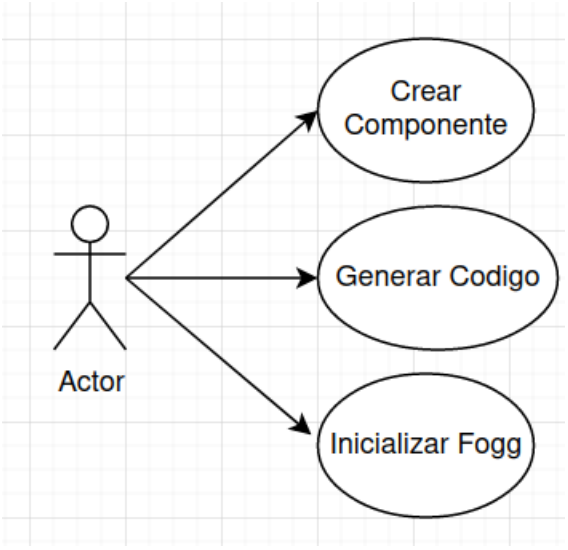


Figura 60 - Diagrama de casos de uso del sistema Fogg

2.3) Problemas y necesidades detectados en las funciones relevadas en detalle y en su entorno organizacional

Problemas asociados a la definición de la configuración

Si bien modificar un archivo de configuración no es muy complejo, es algo poco intuitivo y puede llegar a ser confuso. Sería conveniente proveer una herramienta gráfica para la definición de los ambientes, los componentes, y la información específica de la configuración.

Problemas con la herramienta

El principal problema de la herramienta es su limitación ya que no es muy versátil, esta solo permite la generación de infraestructura con Terraform y utilizando AWS como servicio en la nube lo que limita mucho el uso de esta.

Problemas de Documentación

La Documentación provista por Fogg para el uso de la herramienta puede ser útil para el uso básico de la herramienta, pero si se quiere indagar más profundo en su utilización la documentación no es demasiado útil.

DhiWise

1) Relevamiento General

1.1) De la Organización

DhiWise [5] es una plataforma web que se centra en la automatización del proceso de desarrollo de software, generando código de aplicaciones móviles, backend y frontend.

Visión

Construir la plataforma de programación más inteligente del mundo.

Cultura

DhiSutra. En DhiWise [5], estamos constantemente iterando, resolviendo problemas y trabajando juntos para solucionar los problemas del día a día de los desarrolladores. Por eso es importante que nuestra plantilla tenga esa chispa única para identificar los problemas a los que se enfrentan los desarrolladores. Tenemos una cultura de empleados increíble y única y hemos acuñado el término "DhiSutra" para la misma. Descubra de primera mano cómo es realmente trabajar aquí y conozca nuestros valores DhiSutra. (https://www.dhiwise.com/, 2022)

1.2) Funciones detectadas a nivel general y relaciones con otros Sistemas y Entidades

Para el desarrollo de este informe, vamos a centrarnos en las funcionalidades de generación de código backend.

Crear aplicación.

Permite crear una nueva aplicación en Node.js o Laravel donde se podrá configurar el tipo de arquitectura y el tipo de base de datos a utilizar.

Iniciar sesión.

Permite iniciar sesión en la aplicación con una cuenta de Google o GitHub.

Crear modelo de datos.

Permite crear y configurar el modelo de datos a utilizar para generar el código. También permite usar plantillas de una biblioteca.

Configurar roles de acceso.

Permite crear y configurar distintos roles para el acceso dentro de la plataforma a generar.

Configurar API

Permite configurar el CRUD, Rutas y documentación de la API a generar.

Configurar autenticación.

Permite configurar el módulo de autenticación y personalizar la seguridad del proyecto.

Generar aplicación.

Permite generar la aplicación con la configuración establecida con las funcionalidades anteriormente descriptas.

Descargar código fuente.

Descarga el código fuente de la aplicación generada en un archivo comprimido zip.

Sincronizar repositorio.

Permite sincronizar una cuenta GitHub o GitLab con un repositorio creado vacío.

Según lo analizado en las funcionalidades relevadas, se observa que la aplicación se relaciona con 3 entidades:

Usuario: El usuario al que está dirigido será un desarrollador, diseñador de sistemas, o cualquier individuo o equipo que busque generar código.

GitHub: Se comunica con GitHub para iniciar sesión, al vincular el proyecto en el que se está trabajando con un repositorio y por último al momento de realizar commits al repositorio remoto.

Google: Se comunica con Google para confirmar credenciales de inicio de sesión.

A continuación, se presenta un diagrama de las relaciones y las interacciones.

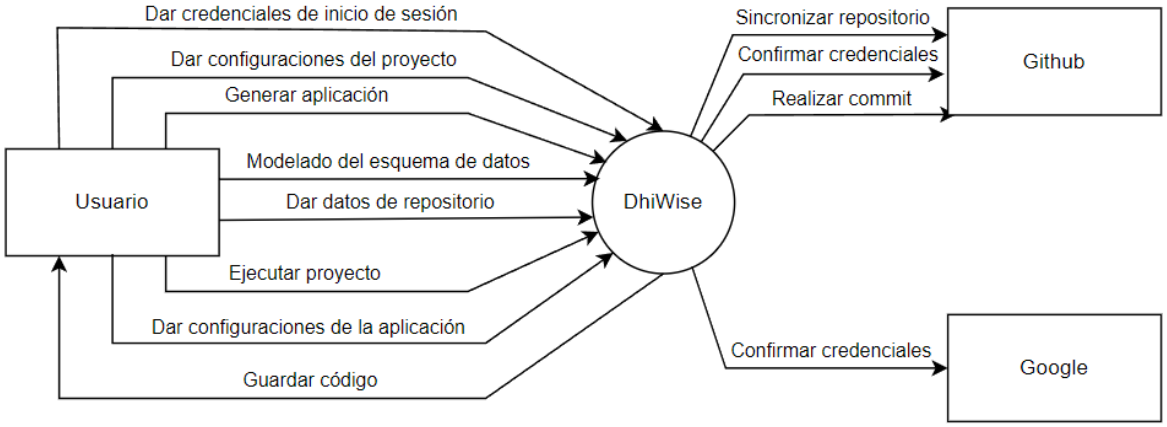


Figura 61 - Diagrama de interacciones del sistema DhiWise

1.3) Tecnología de Información

Utilizando el plugin de Firefox “Wappalyzer” se puede observar que se utilizan las siguientes tecnologías:

- React
- Babel
- AWS
- Tailwind CSS
- Lodash
- Sentry
- Amazon Cloudfront
- Emotion
- Express

A continuación, también enumeramos las tecnologías que permiten generar código.

Para el lado del Cliente:

- React

Para aplicaciones móviles:

- Dart
- Kotlin
- Swift

Para el lado del servidor:

- Node.js
- Laravel
- Bases de Datos:
  - SQL Server
  - MongoDB
  - MySQL
  - PostgreSQL

2) Relevamiento detallado y análisis del Sistema

2.1) Detalle, explicación y documentación detallada de todas las funciones seleccionadas.

Detalle, explicación y documentación detallada de todas las funciones seleccionadas

Crear aplicación nueva.

Permite crear una aplicación nueva, seleccionando la arquitectura que puede ser MVC o MVVM/Clean Code, y seleccionando el tipo de base de datos que se va a utilizar para generar el código.

También permite seleccionar si se desea crear una aplicación en blanco o desde una plantilla.

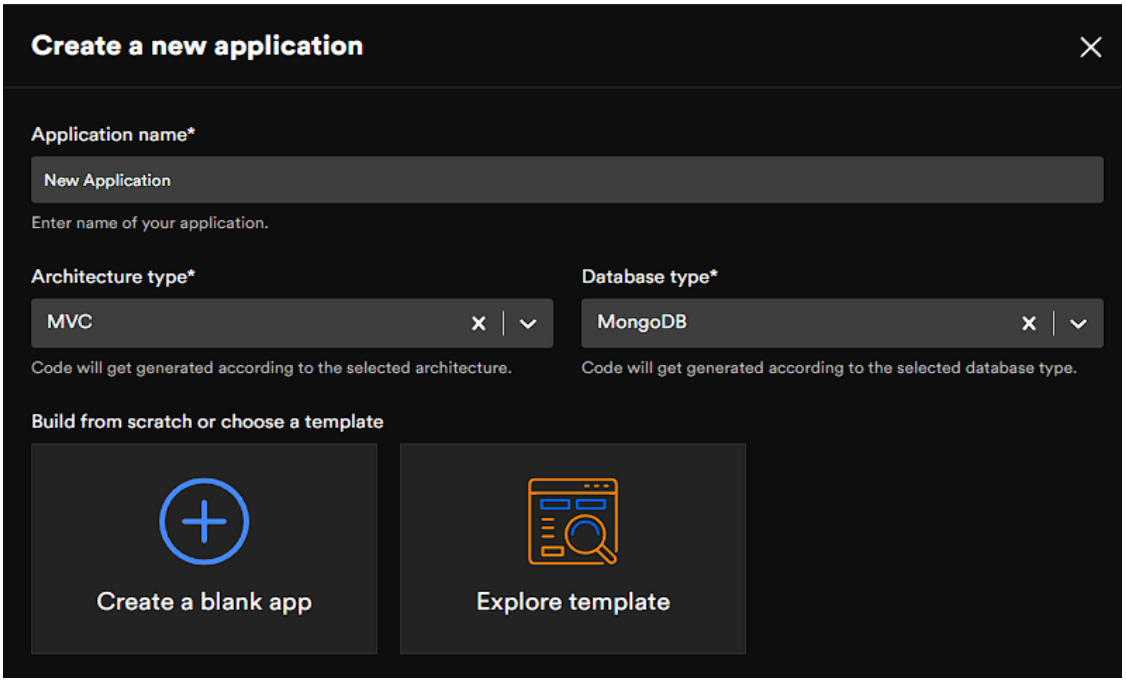


Figura 62 - Interfaz crear aplicación nueva

Iniciar sesión.

Permite iniciar sesión en la aplicación con una cuenta de Google o GitHub. El usuario puede tener varias aplicaciones para generar el código en su cuenta.

Figura 63 - Interfaz iniciar sesión

Crear modelo de datos.

Permite crear y configurar el modelo de datos a utilizar para generar el código. En esta pantalla podremos configurar el esquema, establecer los atributos, el tipo de dato, el valor por defecto y otras configuraciones.

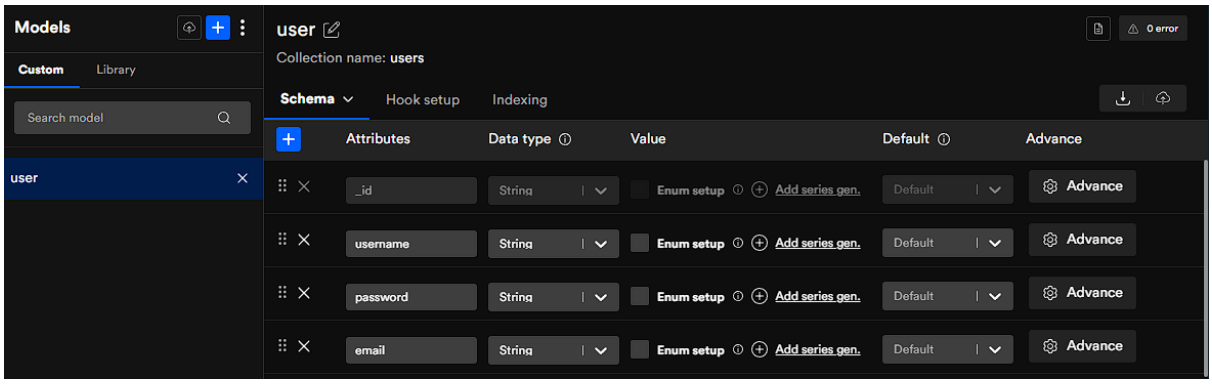


Figura 64 - Interfaz crear modelo de datos

También, permite importar plantillas de una biblioteca.

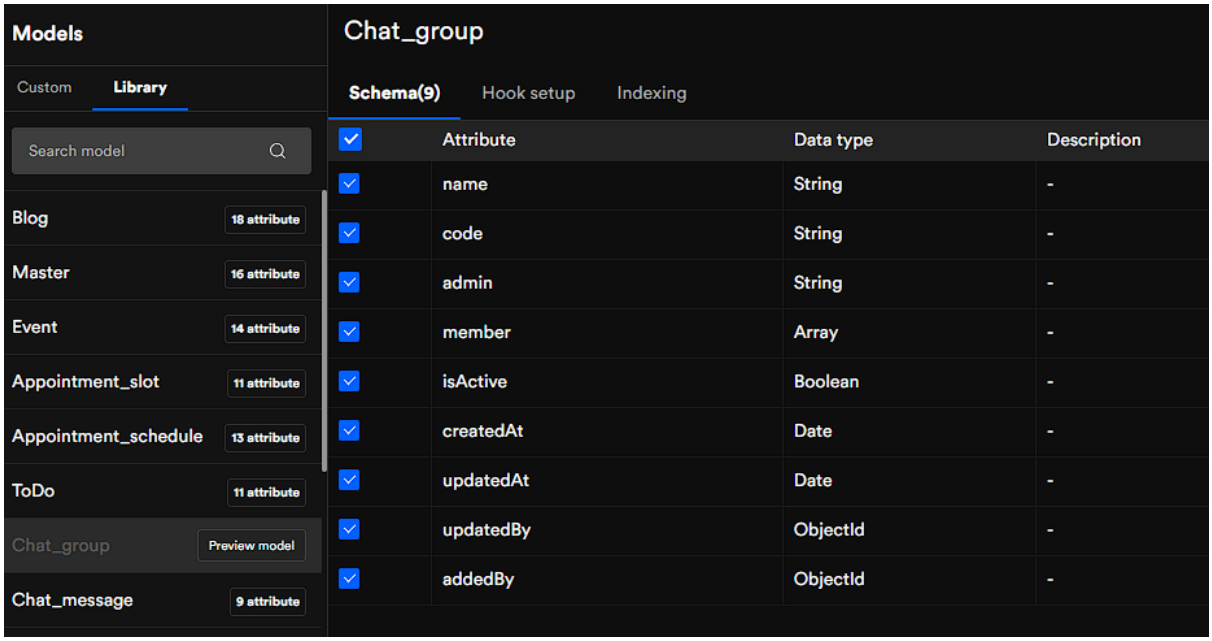


Figura 65 - Interfaz importar plantillas

Configurar roles de acceso.

Permite crear y configurar distintos roles para el acceso dentro de la plataforma a generar.

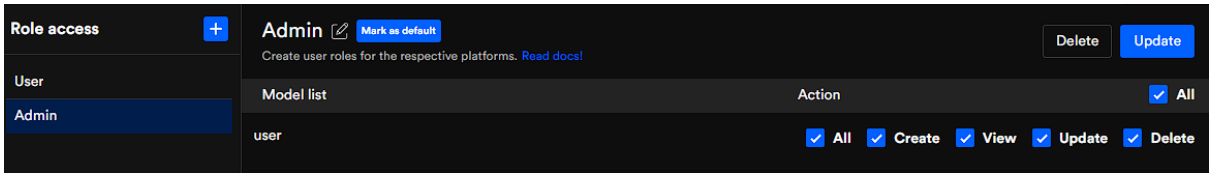


Figura 66 - Interfaz configurar roles de acceso

Configurar API

Permite configurar el CRUD, Rutas y documentación de la API a generar.

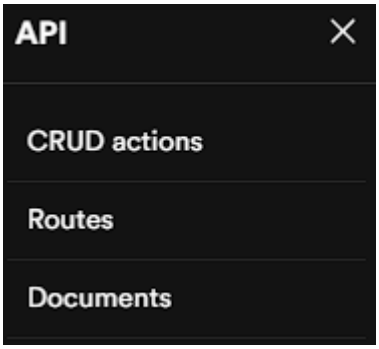


Figura 67 - Interfaz configurar API

La herramienta permite configurar las operaciones CRUD en función del modelo para la plataforma requerida.

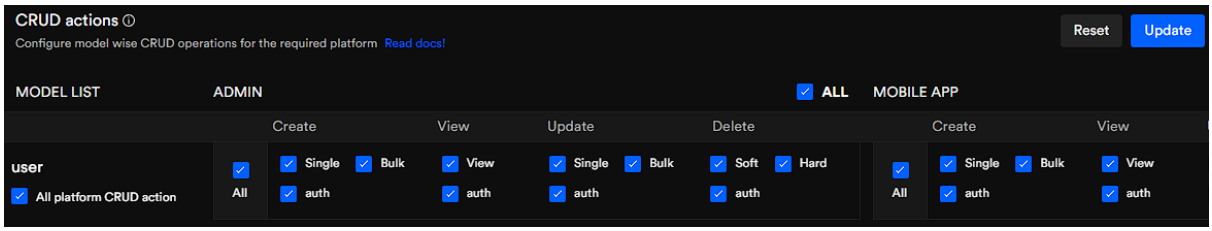


Figura 68 - Interfaz configurar operaciones CRUD

También permite configurar y crear nuevas rutas, además de las generadas automáticamente por la herramienta.

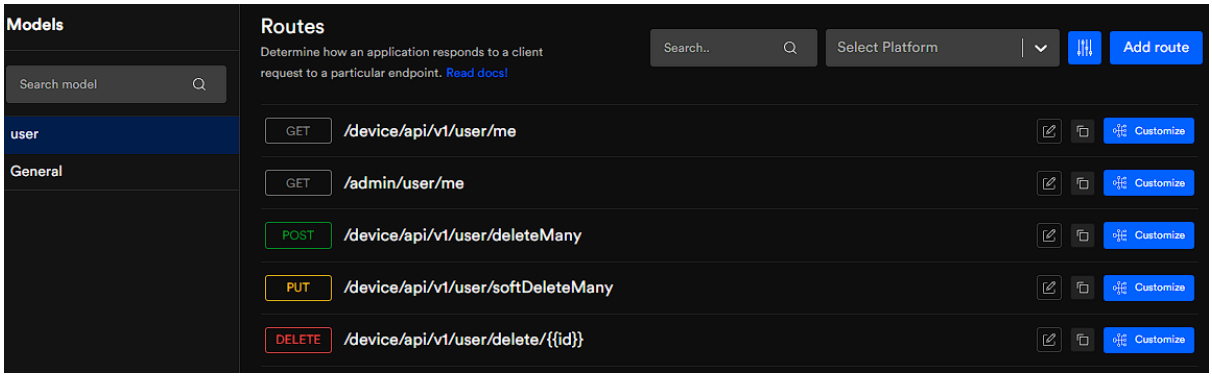


Figura 69 - Interfaz configurar rutas

Finalmente, una vez construida la aplicación requerida, permite acceder a la documentación de la API generada con sus rutas, así como también descargar una colección de Postman.

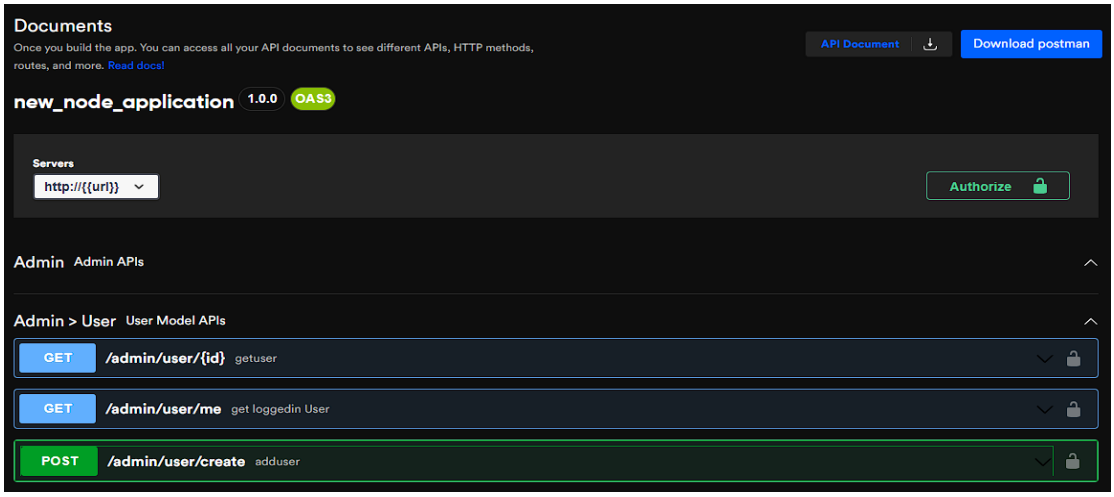


Figura 70 - Interfaz documentación de la API

Configurar autenticación:

Permite configurar el módulo de autenticación y personalizar la seguridad del proyecto. Se puede seleccionar la configuración que se requiera por cada modelo y seleccionar si se requiere verificación en dos pasos.

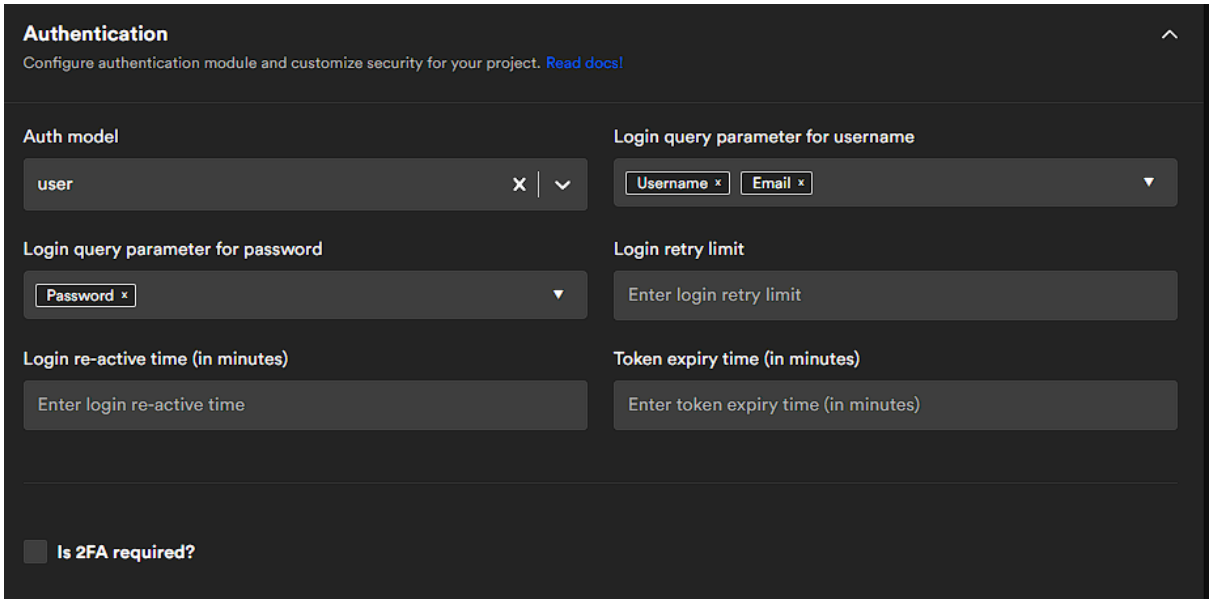


Figura 71 - Interfaz configurar autenticación

Generar aplicación:

Permite generar la aplicación con la configuración establecida con las funcionalidades anteriormente descriptas.

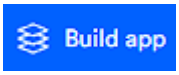


Figura 72 - Interfaz botón generar aplicación

Es posible elegir entre tres opciones, generar y desplegar la aplicación en un entorno aislado para pruebas, directamente a producción o solo generar el código.

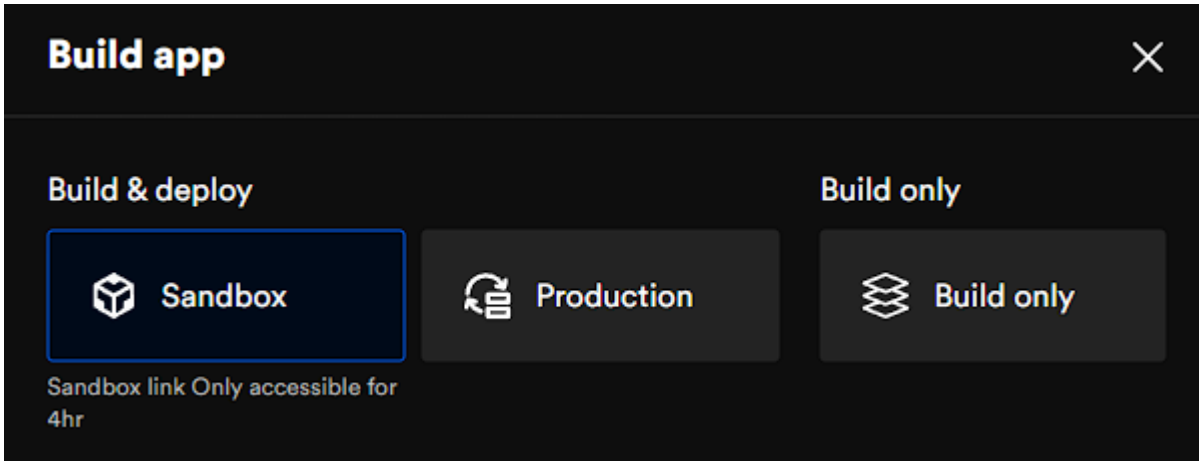


Figura 73 - Interfaz opciones de generación de aplicación

Al generar el código se abre una ventana en el navegador que permite ver el código generado.

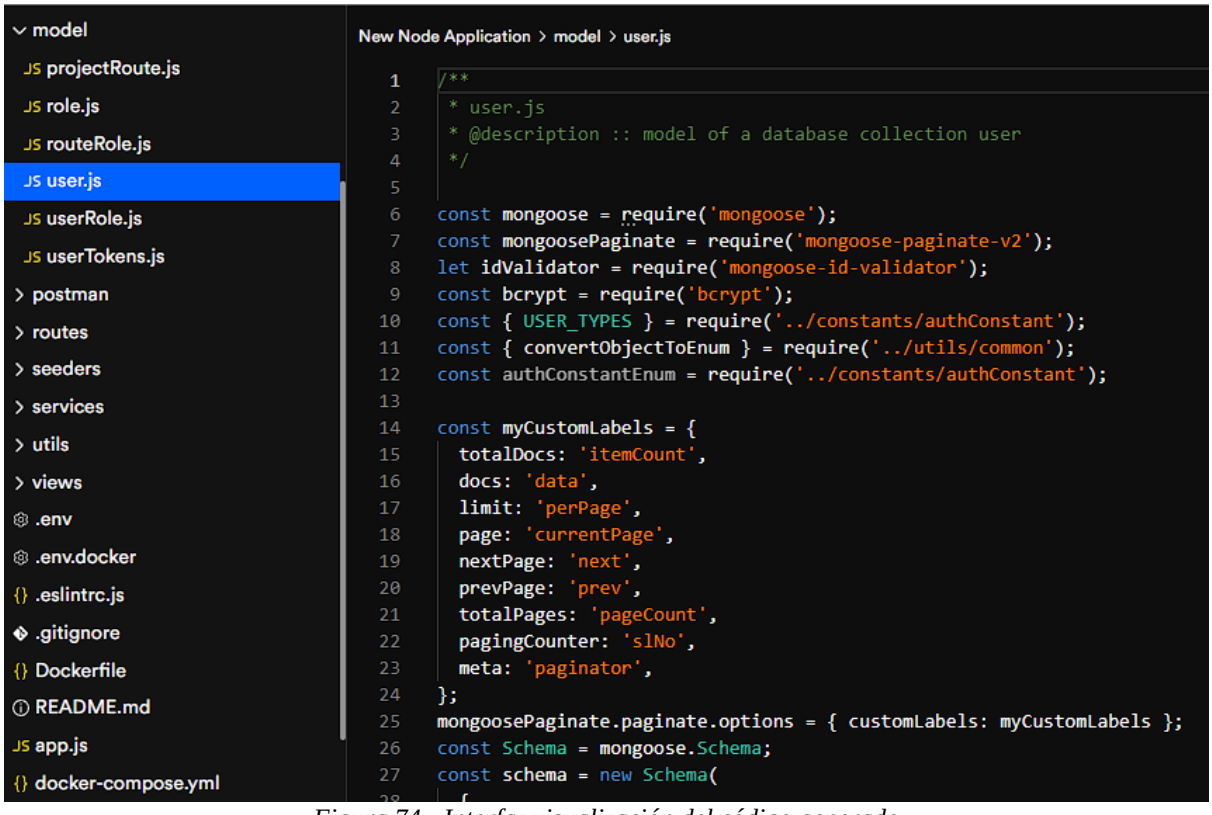


Figura 74 - Interfaz visualización del código generado

Descargar código fuente.

Guarda el código fuente de la aplicación generada en un archivo comprimido zip en el dispositivo utilizado.

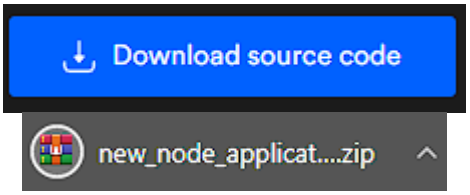


Figura 75 - Botón descarga del código fuente generado

Sincronizar Repositorio.

Permite sincronizar una cuenta en GitHub o GitLab.

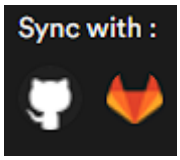


Figura 76 - Interfaz sincronizar repositorio 1

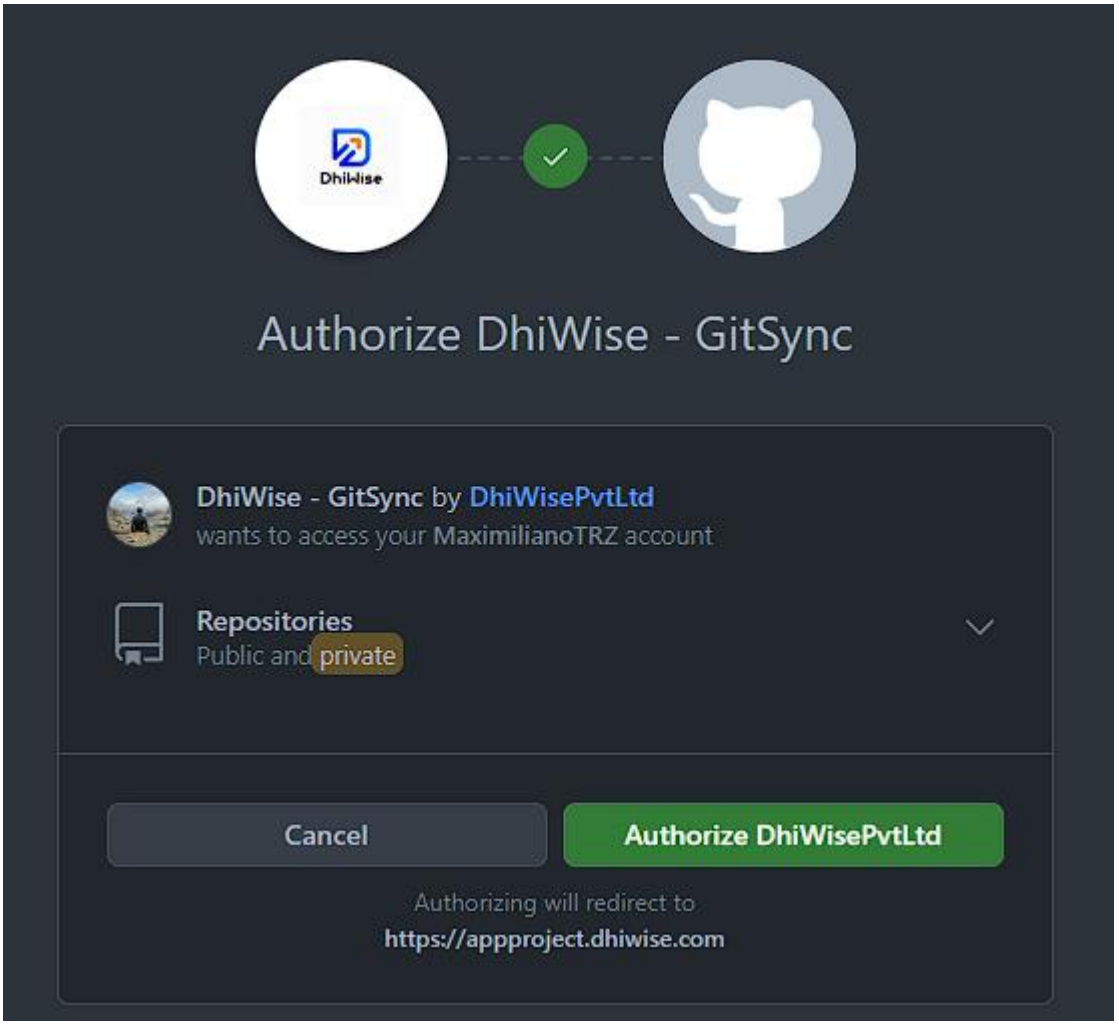


Figura 77 - Interfaz sincronizar repositorio 2

Esta funcionalidad provee un acceso directo para crear el repositorio vacío.

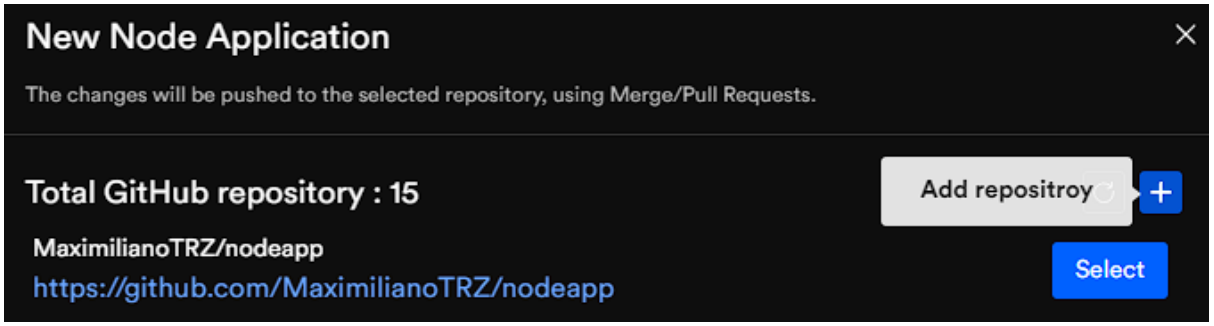


Figura 78 - Interfaz crear repositorio vacío

Finalmente permite cambiar el repositorio o realizar commits desde la interfaz de la aplicación.

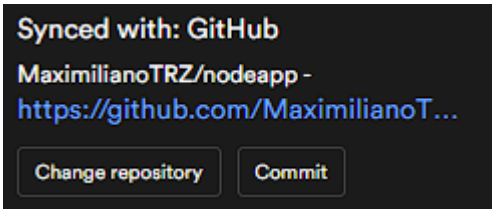


Figura 79 - Interfaz cambiar repositorio o realizar commit

2.2) Modelo lógico del Sistema actual

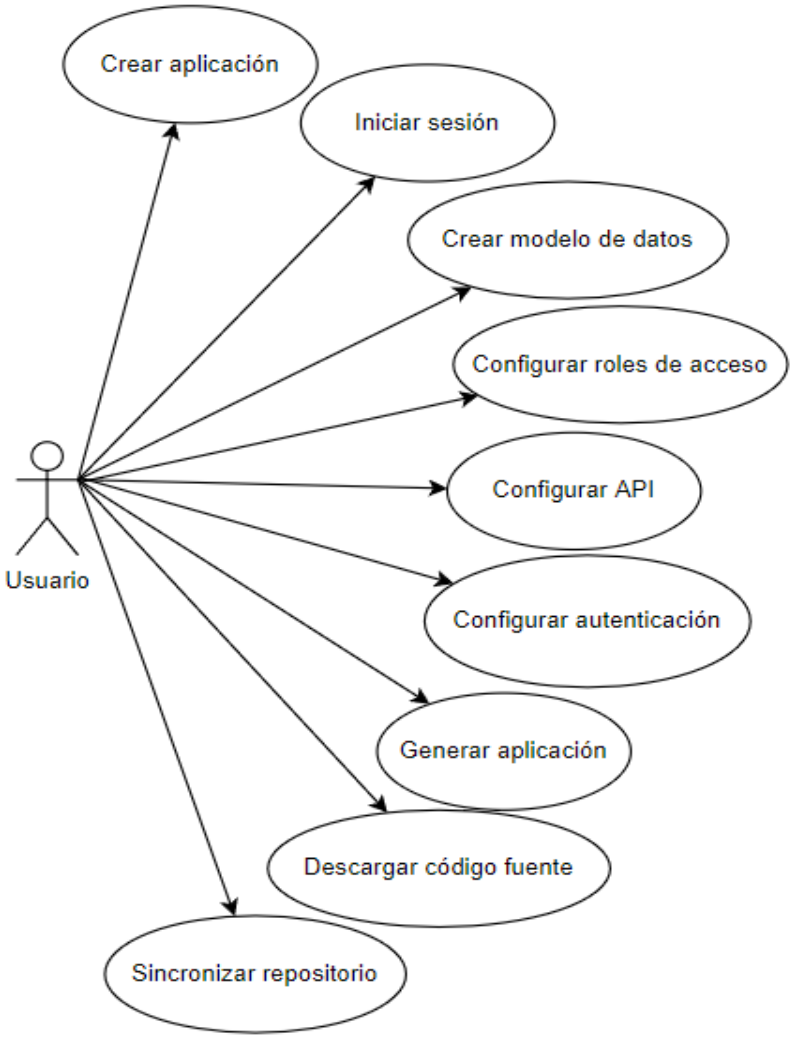


Figura 80 - Diagrama de casos de uso del sistema DhiWise

2.3) Problemas y necesidades detectados en las funciones relevadas en detalle y en su entorno organizacional

Funcionalidad: Generar aplicación.

Problema 1: Generación del código limitada solo a dos tecnologías backend como son Laravel y Node.js.

Necesidad 1: Permitir generar código para más lenguajes dándole flexibilidad al usuario.

Problemas y necesidades en general:

Problema 1: No es posible generar aplicaciones con arquitectura de microservicios. En caso de querer aplicar arquitectura de microservicios en el sistema, se debe realizar generando cada microservicio por separado y separando los archivos útiles.

Problema 2: No permite importar un proyecto anteriormente creado para continuar modelando en el sistema.

Problema 3: No permite la generación e integración de infraestructura con Terraform u otra herramienta para la aplicación generada.

	Amplication	Vemto	JHipster	Fogg	DhiWise
Autenticación	x				x
Es colaborativa	x				
Genera Infraestructura			x	x	
Interfaz Gráfica	x	x			x
Genera Integración Continua			x		
Tiene Herramienta de Modelado		x			
Permite Múltiples Bases de Datos		x	x		
Arquitectura de microservicio para código generado	x		x		

Tabla 1 - Comparativa de Herramientas Relevadas

2.4) Objetivos y alcances preliminares del nuevo Sistema

Objetivos preliminares del sistema

El sistema busca ofrecer una herramienta que genere código y archivos de configuración de infraestructura, basándose en modelos que unifiquen el diseño de una arquitectura de microservicios en la nube con el diseño orientado a objetos de todos los microservicios, permitiendo elegir distintos parámetros para la generación automática tales como el lenguaje de programación de cada microservicio, el proveedor cloud de cada ambiente de la arquitectura, entre otros.

Alcances preliminares del sistema

El usuario utilizará la herramienta a través de un canvas o lienzo en conjunto con una barra de elementos. Para ello, arrastrará los distintos elementos desde la barra hacia el canvas y cada uno de ellos tendrá distintas propiedades para su configuración.

Los modelos que se pueden realizar con la herramienta están divididos en dos segmentos:

- **Modelo de Arquitectura de Microservicios.** El usuario podrá modelar distintos clústeres de distintos proveedores en la nube (Amazon Web Services, por ejemplo) y dentro de cada uno de ellos, crear ambientes que contengan microservicios. Dichos ambientes permiten organizar el clúster en distintos segmentos, agrupando microservicios por algún criterio especificado. Además, podrá especificar los mecanismos de comunicación entre microservicios (mediante peticiones HTTP, por ejemplo), el lenguaje de cada microservicio, la base de datos a utilizar, entre otras cosas.

- **Modelo de Datos de un Microservicio.** Una vez creado un microservicio en el modelo anterior, el usuario podrá adentrarse en el mismo para modelar su funcionamiento interno. Para ello, podrá crear algo similar a un diagrama de clases de UML, creando clases y estableciendo relaciones (incluso podrá establecer una relación con una clase de otro microservicio). Además, el usuario podrá especificar las rutas de acceso REST al microservicio, indicando un path, el tipo de método HTTP que utilice y eligiendo algún método de protección a la ruta.

Una vez que se hayan realizado los modelos, el usuario podrá utilizar los mismos para generar código e infraestructura de forma automática. A continuación, se especifica lo que el usuario podrá obtener:

- **Generación Automática de Infraestructura.** Utilizando como entrada el modelo de la arquitectura de microservicios, el usuario podrá obtener lo siguiente:
    - Archivos de integración continua con la herramienta seleccionada.
    - Manifiestos de configuración de Kubernetes para cada clúster.
    - Archivo Dockerfile para crear una imagen de cada microservicio.
    - Archivos de configuración de Terraform para desplegar automáticamente cada clúster en el proveedor seleccionado.
- **Generación Automática de Código:** Utilizando como entrada el modelo de datos, para cada microservicio el usuario podrá obtener el código en el lenguaje seleccionado de acuerdo con lo siguiente:
    - Conexión a la base de datos especificada.
    - Conexión a servicios de autenticación y autorización de la forma especificada.

- Configuración e implementación de rutas de Alta, Baja y Modificación para las entidades y relaciones especificadas.
- Configuración para las rutas especificadas por el usuario (que deben ser implementadas por el usuario de acuerdo al negocio).

Diseño

Sistema generador automático de código e infraestructura

Diseño

1) Objetivos y alcances definitivos del nuevo Sistema.

Objetivos definitivos del sistema

El sistema busca ofrecer una herramienta que genere código y archivos de configuración de infraestructura, basándose en modelos que unifiquen el diseño de una arquitectura de microservicios en la nube con el diseño orientado a objetos de todos los microservicios, permitiendo elegir distintos parámetros para la generación automática tales como el lenguaje de programación de cada microservicio, el proveedor cloud de cada ambiente de la arquitectura, entre otros.

El administrador del sistema tiene la capacidad de visualizar reportes sobre la utilización de la herramienta a través de gráficos detallados.

Alcances definitivos del sistema

El usuario utilizará la herramienta a través de un canvas o lienzo en conjunto con una barra de elementos. Para ello, arrastrará los distintos elementos desde la barra hacia el canvas y cada uno de ellos tendrá distintas propiedades para su configuración.

Los modelos que se pueden realizar con la herramienta están divididos en dos módulos:

- **Módulo de Ambientes y Microservicios.** El usuario podrá modelar distintos clústers de distintos proveedores en la nube (Amazon Web Services, por ejemplo) y dentro de cada uno de ellos, crear ambientes que contengan microservicios. Dichos ambientes permiten organizar el clúster en distintos segmentos, agrupando microservicios por algún criterio especificado. Además, podrá especificar los mecanismos de comunicación entre

microservicios (mediante peticiones HTTP, por ejemplo), el lenguaje de cada microservicio, la base de datos a utilizar, entre otras cosas.

- **Módulo de Entidades y Rutas.** Una vez creado un microservicio en el modelo anterior, el usuario podrá adentrarse en el mismo para modelar su funcionamiento interno. Para ello, podrá crear algo similar a un diagrama de clases de UML, creando clases y estableciendo relaciones (incluso podrá establecer una relación con una clase de otro microservicio). Además, el usuario podrá especificar las rutas de acceso REST al microservicio, indicando un path, el tipo de método HTTP que utilice y eligiendo algún método de protección a la ruta.

Una vez que se hayan realizado los modelos, el usuario puede utilizar los mismos para generar código e infraestructura de forma automática. A continuación, se especifica lo que el usuario podrá obtener de los dos módulos utilizados en la generación automática:

- **Módulo de Generación de Infraestructura.** Utilizando como entrada el modelo de la arquitectura de microservicios, el usuario podrá obtener lo siguiente:
    - Archivos de integración continua con la herramienta seleccionada.
    - Manifiestos de configuración de Kubernetes para cada clúster.
    - Archivo Dockerfile para crear una imagen de cada microservicio.
    - Archivos de configuración de Terraform para desplegar automáticamente cada clúster en el proveedor seleccionado.
- **Módulo de Generación de Código Backend.** Utilizando como entrada el modelo de datos, para cada microservicio el usuario podrá obtener el código en el lenguaje seleccionado de acuerdo con lo siguiente:

- Conexión a la base de datos especificada.
- Conexión a servicios de autenticación y autorización de la forma especificada.
- Configuración e implementación de rutas de Alta, Baja y Modificación para las entidades y relaciones especificadas.
- Configuración para las rutas especificadas por el usuario (que deben ser implementadas por el usuario de acuerdo al negocio).

Junto con la creación del código en base al modelo diseñado se generan informes que son enviados a un sistema de recolección de reportes para la posterior visualización del administrador

- **Módulo de Reportes.** Utilizando como entrada el modelo de datos se extrae información del diagrama. Esta información no contiene detalles sobre el producto diseñado por el usuario, como por ejemplo la arquitectura y nombre de entidades. Los detalles extraídos sólo detallan cómo y cuánto se utiliza la herramienta para conocer mejor el estado de la misma.

  El reporte es enviado a un servicio a cargo de guardarlo en la base de datos.

  Un usuario administrador puede visualizar esta información a través de un dashboard interactivo que consulta los datos en el servicio de reportes.

2) Modelo funcional

Historias de Usuario

El diseño de las funcionalidades de la herramienta de modelado está implementado a través de historias de usuario. Estas se dividen en:

- Épicas: Módulo del sistema, correspondiente en este caso con una vista de la herramienta o un artefacto a generar código.
- Historia de Usuario
- Criterios de aceptación: Presentes en cada historia de usuario. Son los criterios que se utilizan para corroborar el funcionamiento esperado.

A continuación se encuentran las historias de usuario separadas por el color respectivo a la épica de la que provienen y la pantalla a la que están relacionadas dentro de la interfaz.

Módulo de Ambientes y Microservicios		
US-001 Crear Ambiente		
Descripción	Como usuario quiero crear un ambiente de aprovisionamiento para incluirlo dentro del modelo.	
Criterios de Aceptación	Creación de Ambiente	
	Contexto	La vista de modelado de microservicios está disponible
	Evento	Cuando el usuario arrastra el ambiente dentro de la vista de modelado
	Resultado Esperado	El ambiente es creado y visible en la vista de modelado de microservicios con valores por defecto

Tabla 2 - US-001 Crear Ambiente

Módulo de Ambientes y Microservicios
US-002 Modificar Ambiente

Descripción	Como usuario quiero modificar las características de un ambiente	
Criterios de Aceptación	Modificación de ambiente	
	Contexto	El ambiente existe
	Evento	Cuando el usuario presiona el ambiente Ingresa los valores correspondientes al proveedor específico. ● Nombre: String ● ProjectId: Int ● Región: String
	Resultado Esperado	Se provee una vista con los datos disponibles de completar, una vez completados se guardan
	Ambiente con mismo nombre	
	Contexto	Hay ambientes creados
	Evento	Cuando el usuario intenta modificar el ambiente con el mismo nombre de otro ambiente
	Resultado Esperado	Se advierte sobre el error y no se permite la creación hasta cambiar el valor
	Datos de Ambiente Incompleto	
	Contexto	El ambiente existe
	Evento	Cuando el usuario intenta guardar la modificación con datos incompletos
	Resultado Esperado	No se permite guardar la modificación

Tabla 3 - US-002 Modificar Ambiente

Módulo de Ambientes y Microservicios

US-003 Eliminar Ambiente		
Descripción	Como usuario quiero eliminar un ambiente previamente creado para que deje de ser parte del modelo	
Criterios de Aceptación	Eliminación de Ambiente	
	Contexto	Hay amiente/s en la vista de modelado.
	Evento	Cuando el usuario selecciona la opción de eliminar
	Resultado Esperado	El ambiente se elimina de la vista sin eliminar los microservicios dentro

Tabla 4 - US-003 Eliminar Ambiente

Módulo de Ambientes y Microservicios		
US-004 Crear Configuración de Kubernetes del Ambiente		
Descripción	Como usuario quiero crear una nueva configuración de kubernetes para un ambiente específico para detallar las características de aprovisionamiento deseadas	
Criterios de Aceptación	Creación Kubernetes Config del ambiente	
	Contexto	Hay ambientes creados
	Evento	Cuando el usuario selecciona crear configuración kubernetes
	Resultado Esperado	La configuración es creada exitosamente con valores por defecto

Tabla 5 - US-004 Crear Configuración de Kubernetes del Ambiente

Módulo de Ambientes y Microservicios

US-005 Modificar Configuración de Kubernetes del Ambiente		
Descripción	Como usuario quiero modificar la configuración del aprovisionamiento con kubernetes de un ambiente específico para cambiar sus características	
Criterios de Aceptación	Modificación de Kubernetes Config del Ambiente	
	Contexto	La configuración de kubernetes está creada
	Evento	Cuando el usuario selecciona la configuración kubernetes Ingresa los valores correspondientes a la configuración para el proveedor: ● PoolName: String ● MachineType: String ● nodeMinCount: Integer ● nodeMaxCount: Integer ● nodeDiskSize: String
	Resultado Esperado	Se permite la modificación de la configuración de kubernetes
	Datos de configuración en NULL	
	Contexto	La configuración de kubernetes está creada
	Evento	Cuando el usuario modifica los campos y deja algunos vacíos
	Resultado Esperado	No se permite la modificación y se advierte al usuario sobre el error

Tabla 6 - US-005 Modificar Configuración de Kubernetes del Ambiente

Módulo de Ambientes y Microservicios	
US-006 Eliminar Configuración Kubernetes del Ambiente	
Descripción	Como usuario quiero eliminar la configuración de kubernetes de un ambiente

Criterios de Aceptación	Eliminación de Kubernetes Config.	
	Contexto	La configuración de kubernetes del ambiente se encuentra creada
	Evento	Cuando el usuario selecciona eliminar la configuración de kubernetes
	Resultado Esperado	La configuración es eliminada del ambiente

Tabla 7 - US-006 Eliminar Configuración Kubernetes del Ambiente

Módulo de Ambientes y Microservicios		
US-007 Crear Configuración de Kubernetes del Microservicio		
Descripción	Como usuario quiero crear una configuración kubernetes de un microservicio para detallar las características de aprovisionamiento de este.	
Criterios de Aceptación	Creación Kubernetes Config del microservicio	
	Contexto	Hay microservicios creados
	Evento	Cuando el usuario selecciona crear configuración kubernetes
	Resultado Esperado	La configuración es creada exitosamente con valores por defecto

Tabla 8 - US-007 Crear Configuración de Kubernetes del Microservicio

Módulo de Ambientes y Microservicios	
US-008 Modificar Configuración de Kubernetes del Microservicio	
Descripción	Como usuario quiero modificar la configuración del aprovisionamiento con kubernetes de un ambiente específico para cambiar sus características

Criterios de Aceptación	Modificación de Kubernetes Config del microservicio	
	Contexto	La configuración de kubernetes está creada
	Evento	Cuando el usuario selecciona la configuración kubernetes. Ingresa los valores correspondientes a la configuración para el microservicio: ● replicas: Integer ● memoryRequest: Integer ● cpuRequest: Integer ● memoryLimit: Integer ● cpuLimit: Integer
	Resultado Esperado	Se realiza la modificación de la configuración de kubernetes para el microservicio.
	Datos de configuración en NULL	
	Contexto	La configuración de kubernetes está creada
	Evento	Cuando el usuario modifica los campos y deja algunos vacíos
	Resultado Esperado	No se permite la modificación y se advierte al usuario sobre el error

Tabla 9 - US-008 Modificar Configuración de Kubernetes del Microservicio

Módulo de Ambientes y Microservicios		
US-009 Eliminar Configuración Kubernetes del microservicio		
Descripción	Como usuario quiero modificar la configuración kubernetes de un microservicio para detallar cambios en esta	
Criterios de Aceptación	Eliminación de Kubernetes Config. del Microservicio	
	Contexto	La configuración de kubernetes del microservicio se encuentra

		creada
	Evento	Cuando el usuario selecciona eliminar la configuración de kubernetes
	Resultado Esperado	La configuración es eliminada del microservicio

Tabla 10 - US-009 Eliminar Configuración Kubernetes del microservicio

Módulo de Ambientes y Microservicios		
US-0010 Crear Microservicio Backend		
Descripción	Como usuario quiero crear un microservicio backend dentro de un ambiente para detallar las características de este.	
Criterios de Aceptación	Creación de Microservicio	
	Contexto	Está disponible la creación de microservicios backend
	Evento	Cuando el usuario arrastra el microservicio dentro de un ambiente
	Resultado Esperado	El microservicio se crea y es visible en la vista de modelado de microservicios
	Creación fuera de un ambiente	
	Contexto	Está disponible la creación de microservicios backend
	Evento	Cuando el usuario arrastra el microservicio fuera de un ambiente en la vista de modelado
	Resultado Esperado	El microservicio no se crea

Tabla 11 - US-0010 Crear Microservicio Backend

Módulo de Ambientes y Microservicios		
US-0011 Modificar Microservicio Backend		
Descripción	Como usuario quiero modificar las características base de un microservicio	
Criterios de Aceptación	Modificación de Microservicio	
	Contexto	El microservicio está creado.
	Evento	Cuando se selecciona el microservicio Ingresa los valores correspondientes a la configuración para el microservicio: •
	Resultado Esperado	Se permite la modificación del microservicio Backend
	Microservicio con el mismo nombre	
	Contexto	Hay Microservicios creados.
	Evento	Cuando el usuario modifica el microservicio colocando el mismo nombre que otro microservicio
	Resultado Esperado	Se advierte sobre el error y no se permite la modificación hasta cambiar el valor

Tabla 12 - US-0011 Modificar Microservicio Backend

Módulo de Ambientes y Microservicios

US-012 Cambiar de Ambiente Microservicio Backend		
Descripción	Como usuario quiero cambiar el ambiente en el que está presente un microservicio para que sea parte de otro ambiente de aprovisionamiento.	
Criterios de Aceptación	Cambio de ambiente de un microservicio	
	Contexto	Mas de 1 ambiente se encuentra creado y el microservicio existe
	Evento	Cuando se arrastra el microservicio a otro ambiente
	Resultado Esperado	El microservicio se vuelve parte del nuevo ambiente

Tabla 13 - US-012 Cambiar de Ambiente Microservicio Backend

Módulo de Ambientes y Microservicios		
US-013 Eliminar Microservicio Backend		
Descripción	Como usuario quiero eliminar un microservicio backend para que deje de ser parte del modelo.	
Criterios de Aceptación	Eliminación de Microservicio	
	Contexto	El microservicio está creado.
	Evento	Cuando el usuario presiona eliminar Microservicio Backend
	Resultado Esperado	El microservicio se elimina y ya no es visible en la vista de microservicios

Tabla 14 - US-013 Eliminar Microservicio Backend

Módulo de Ambientes y Microservicios

US-014 Ventana Capacidades		
Descripción	Como usuario quiero poder acceder a una vista en la que se observen las capacidades creadas en el microservicio y me permite seleccionar nuevas	
Criterios de Aceptación	Abrir Vista de Capacidades	
	Contexto	Hay Microservicios creados
	Evento	Al presionar el botón de capacidades de un microservicio
	Resultado Esperado	Se muestra una ventana con todas las capacidades creadas en el microservicio

Tabla 15 - US-014 Ventana Capacidades

Módulo de Ambientes y Microservicios		
US-015 Agregar Capacidades		
Descripción	Como usuario quiero agregar capacidades al microservicio	
Criterios de Aceptación	Seleccionar Capacidad	
	Contexto	El microservicio está creado
	Evento	Al seleccionar una capacidad a crear, se ingresan los datos correspondientes a la capacidad. Ej: Capacidad de Bases de datos NoSQL ● nombre: String ● username: String ● password: String ● host: String ● databaseName: String
	Resultado Esperado	Se crea la capacidad exitosamente

	Datos Incompletos	
	Contexto	Se está creando una capacidad del microservicio
	Evento	Cuando el usuario intenta guardar la capacidad con valores incompletos
	Resultado Esperado	No se permite la creación
	Agregar Capacidad	
	Contexto	El microservicio está creado
	Evento	Cuando se guarda la capacidad a agregar
	Resultado Esperado	Se crea la capacidad correspondiente con los datos indicados

Tabla 16 - US-015 Agregar Capacidades

Módulo de Ambientes y Microservicios		
US-016 Modificar Capacidad		
Descripción	Como usuario quiero modificar la capacidad de base de datos para cambiar los detalles de su configuración.	
Criterios de Aceptación	Modificar Capacidad	
	Contexto	El microservicio y sus capacidades se encuentran creados y el usuario está en la ventana de capacidades
	Evento	Al seleccionar una capacidad a crear, se ingresan los datos correspondientes a la capacidad. Ej: Capacidad de Bases de datos NoSQL

		• nombre: String • username: String • password: String • host: String • databaseName: String
	Resultado Esperado	Se modifica la capacidad exitosamente.
	Datos de capacidad incompletos	
	Contexto	El microservicio y las capacidades se encuentran creados
	Evento	Cuando el usuario intenta guardar la modificación de los datos con valores incompletos
	Resultado Esperado	No se permite la modificación y se advierte al usuario sobre el error

Tabla 17 - US-016 Modificar Capacidad

Módulo de Ambientes y Microservicios		
US-017 Eliminar Capacidad/es		
Descripción	Como usuario quiero eliminar 1 o varias capacidades presentes en un microservicio.	
Criterios de Aceptación	Eliminar Capacidades	
	Contexto	Hay capacidades de un microservicio creadas
	Evento	Cuando el usuario presiona eliminar
	Resultado Esperado	Se muestra una lista de las capacidades y se permite seleccionar, una vez seleccionadas se eliminan.

Tabla 18 - US-017 Eliminar Capacidad/es

Módulo de Entidades y Rutas		
US-101 Vista de Modelado de Microservicio		
Descripción	Como usuario quiero acceder a una vista de configuración del microservicio para detallar las entidades y Namespaces que serán parte de este	
Criterios de Aceptación	Vista de Microservicio	
	Contexto	El Microservicio se encuentra creado
	Evento	Cuando el usuario presiona el botón de vista de microservicio
	Resultado Esperado	Se redirige a la vista de modelado de microservicio la cual solo tiene creado un entityPackage

Tabla 19 - US-101 Vista de Modelado de Microservicio

Módulo de Entidades y Rutas		
US-102 Crear REST namespace		
Descripción	Como usuario quiero crear un REST Namespace en un microservicio.	
Criterios de Aceptación	Creación de Rest Namespace	
	Contexto	La vista de modelado del microservicio está disponible
	Evento	Cuando el usuario arrastra el Rest namespace a la vista de modelado de microservicio
	Resultado Esperado	El Rest Namespace es creado vacío (sin Rest Resources dentro) y con valores por defecto

Tabla 20 - US-102 Crear REST namespace

Módulo de Entidades y Rutas

US-103 Modificar REST namespace		
Descripción	Como usuario quiero modificar las características de un REST Namespace.	
Criterios de Aceptación	Modificación de Rest Namespace	
	Contexto	El namespace a modificar debe estar creado
	Evento	Cuando el usuario presiona el Rest namespace
	Resultado Esperado	Se presenta un formulario con los datos que se pueden modificar
	Modificación de Atributo	
	Contexto	El namespace a modificar debe estar creado
	Evento	Cuando el usuario presiona el atributo e ingresa los valores para los campos del REST Namespace, en este caso los atributos son: • nombre: String
	Resultado Esperado	El campo del atributo permite la modificación con un nuevo valor
	Rest Namespace con el mismo nombre	
	Contexto	Hay Rest namespaces creados
	Evento	Cuando el usuario modifica el Rest Namespace con un nombre ya presente en otro Rest namespace
	Resultado Esperado	Se advierte sobre el error y no se permite la creación hasta cambiar el valor

Tabla 21 - US-103 Modificar REST namespace

Módulo de Entidades y Rutas

US-104 Eliminar REST namespace		
Descripción	Como usuario quiero eliminar un REST namespace de un microservicio para que no sea parte de este.	
Criterios de Aceptación	Eliminación de Rest Namespace	
	Contexto	El namespace a eliminar se encuentra creado
	Evento	Cuando el usuario presiona la opción de eliminar
	Resultado Esperado	Se elimina el Rest Namespace y los Rest Resources contenidos en el

Tabla 22 - US-104 Eliminar REST namespace

Módulo de Entidades y Rutas		
US-105 Crear REST resource		
Descripción	Como usuario quiero crear un REST Resource en un namespace.	
Criterios de Aceptación	Creación de Rest Resource	
	Contexto	Hay Rest namespaces creados
	Evento	Cuando el usuario arrastra el Rest Resource dentro de un Rest Namespace
	Resultado Esperado	Se crea un Rest Resource dentro del Rest Namespace vacío (sin Rest Routes dentro)

Tabla 23 - US-105 Crear REST resource

Módulo de Entidades y Rutas
US-106 Modificar REST resource

Descripción	Como usuario quiero modificar las características de un REST Resource.	
Criterios de Aceptación	Modificación de Rest Resource	
	Contexto	El resource a modificar debe estar creado
	Evento	Cuando el usuario presiona el Rest resource
	Resultado Esperado	Se presenta un formulario con los datos que se pueden modificar
	Modificación de Atributo	
	Contexto	El resource a modificar debe estar creado
	Evento	Cuando el usuario presiona el atributo e ingresa los valores para los campos del REST resource, en este caso los atributos son: • nombre: String
	Resultado Esperado	El campo del atributo permite la modificación con un nuevo valor.
	Rest resources con el mismo nombre	
	Contexto	Hay Rest resources creados
	Evento	Cuando el usuario modifica el Rest resource con un nombre ya presente en otro Rest resource
	Resultado Esperado	Se advierte sobre el error y no se permite la creación hasta cambiar el valor

Tabla 24 - US-106 Modificar REST resource

Módulo de Entidades y Rutas
US-107 Eliminar REST resource

Descripción	Como usuario quiero eliminar un REST resource de un namespace para que no sea parte de este.	
Criterios de Aceptación	Eliminación de Rest resource	
	Contexto	El Rest resource a eliminar se encuentra creado
	Evento	Cuando el usuario presiona la opción de eliminar
	Resultado Esperado	Se elimina el Rest resource y los Rest routes contenidos en el

Tabla 25 - US-107 Eliminar REST resource

Módulo de Entidades y Rutas		
US-108 Crear REST route		
Descripción	Como usuario quiero crear un Rest Route en un Resource para detallar las características de este.	
Criterios de Aceptación	Creación de Rest route	
	Contexto	Hay Rest resources creados
	Evento	Cuando el usuario arrastra el Rest route dentro de un Rest resource
	Resultado Esperado	Se crea un Rest Route dentro del Rest Resource

Tabla 26 - US-108 Crear REST route

Módulo de Entidades y Rutas
US-109 Modificar REST route

Descripción	Como usuario quiero modificar las características de un REST Route.	
Criterios de Aceptación	Modificación de Rest Route	
	Contexto	El Rest Route a modificar debe estar creado
	Evento	Cuando el usuario presiona el REST Route
	Resultado Esperado	Se presenta un formulario con los datos que se pueden modificar
	Modificación de Atributo	
	Contexto	El Rest Route a modificar debe estar creado
	Evento	Cuando el usuario presiona el atributo e ingresa los valores para los campos del REST resource, en este caso los atributos son: • pathExtension: String • parameters: String • httpMethod: String • requiresAuth: Boolean
	Resultado Esperado	El campo del atributo permite la modificación con un nuevo valor.
	Rest Route con el mismo nombre	
	Contexto	Hay Rest Routes creados dentro del Rest Resource
	Evento	Cuando el usuario modifica el Rest route con un nombre ya presente en otro Rest route dentro del resource.
	Resultado Esperado	Se advierte sobre el error y no se permite la creación hasta cambiar el valor

Tabla 27 - US-109 Modificar REST route

Módulo de Entidades y Rutas

US-110 Eliminar REST Route		
Descripción	Como usuario quiero eliminar un REST route de un Rest Resource para que no sea parte de este.	
Criterios de Aceptación	Eliminación de Rest resource	
	Contexto	El Rest resource a eliminar se encuentra creado
	Evento	Cuando el usuario presiona la opción de eliminar
	Resultado Esperado	Se elimina el Rest route del Rest Resource

Tabla 28 - US-110 Eliminar REST Route

Módulo de Entidades y Rutas		
US-112 Crear Clase		
Descripción	Como usuario quiero crear una clase dentro del microservicio para detallar sus características.	
Criterios de Aceptación	Creación de Clase	
	Contexto	La vista de modelado de microservicio está disponible con un entityPackage
	Evento	Cuando el usuario arrastra una nueva clase dentro del entityPackage
	Resultado Esperado	La clase es creada sin ningún atributo ni relación y con valores por defecto

Tabla 29 - US-112 Crear Clase

Módulo de Entidades y Rutas		
US-113 Modificar Clase		
Descripción	Como usuario quiero modificar una clase para cambiar sus características.	
Criterios de Aceptación	Modificación de Clase	
	Contexto	La clase se encuentra creada
	Evento	Cuando el usuario presiona la clase y modifica las valores correspondientes a la clase: • hasRoutes: Boolean • isAbstract: Boolean
	Resultado Esperado	La clase es modificada exitosamente.
	Clase con mismo nombre ya existe	
	Contexto	Existen clases creadas en el microservicio
	Evento	Cuando el usuario modifica una clase aplicando el mismo nombre que otra clase presente en el microservicio
	Resultado Esperado	Se advierte sobre el error y no se permite la modificación hasta cambiar el valor

Tabla 30 - US-113 Modificar Clase

Módulo de Entidades y Rutas	
US-114 Eliminar Clase	
Descripción	Como usuario quiero eliminar una clase del microservicio.
Criterios de	Eliminación de Clase

Aceptación	Contexto	La clase a eliminar se encuentra creada
	Evento	Cuando el usuario presiona eliminar clase
	Resultado Esperado	La clase es eliminada del entityPackage y ya no es visible en la vista de modelado del microservicio

Tabla 31 - US-114 Eliminar Clase

Módulo de Entidades y Rutas		
US-115 Agregar Atributo		
Descripción	Como usuario quiero crear un atributo en una clase para ampliar la información de la clase.	
Criterios de Aceptación	Creación de Atributo	
	Contexto	La clase se encuentra creada
	Evento	Cuando el usuario arrastra el atributo dentro de una clase e indica: • name: String • type: String • required: Boolean • isUnique: Boolean
	Resultado Esperado	El atributo es creado dentro de la clase con los valores indicados y es visible dentro de la clase

Tabla 32 - US-115 Agregar Atributo

Módulo de Entidades y Rutas
US-116 Modificar Atributo

Descripción	Como usuario quiero modificar las características de un atributo.	
Criterios de Aceptación	Modificación de Atributo	
	Contexto	El atributo de la clase se encuentra creado.
	Evento	Cuando el usuario presiona el atributo y modifica los valores: • name: String • type: String • required: Boolean • isUnique: Boolean
	Resultado Esperado	El Atributo es modificado exitosamente.
	Atributo con mismo nombre ya existe	
	Contexto	Existen atributos creados en la clase.
	Evento	Cuando el usuario modifica un atributo aplicando el mismo nombre que otro atributo presente en la clase.
	Resultado Esperado	Se advierte sobre el error y no se permite la modificación hasta cambiar el valor.
	Tipo de atributo no especificado	
	Contexto	La clase se encuentra creada y se está modificando el atributo.
	Evento	Cuando el usuario intenta guardar los cambios del atributo sin especificar el tipo del atributo.
	Resultado Esperado	Se advierte sobre el error y no se permite la hasta agregar el tipo.

Tabla 33 - US-116 Modificar Atributo

Módulo de Entidades y Rutas

US-117 Eliminar Atributo		
Descripción	Como usuario quiero eliminar un atributo para que deje de ser parte de la clase.	
Criterios de Aceptación	Eliminación de Atributo	
	Contexto	El atributo se encuentra creado
	Evento	Cuando el usuario presiona eliminar atributo
	Resultado Esperado	El atributo es eliminado de la clase y ya no es visible en la vista de modelado del microservicio

Tabla 34 - US-117 Eliminar Atributo

Módulo de Entidades y Rutas		
US-118 Crear Relación Herencia		
Descripción	Como usuario quiero crear una herencia entre 2 clases para indicar esta característica en el modelo.	
Criterios de Aceptación	Creación de Relación de Herencia	
	Contexto	Hay más de una clase creada en la vista de modelado del microservicio
	Evento	Cuando el usuario conecta las clases arrastrando la herencia e indica: • nombre:String
	Resultado Esperado	La herencia es creada y es visible en la vista de modelado del microservicio.
	Intento de crear múltiples herencias	

	Contexto	Hay varias clases creadas y la clase a agregar herencia ya cuenta con una herencia
	Evento	Cuando el usuario intenta crear una herencia que parte de una clase que ya contiene una herencia
	Resultado Esperado	La herencia no se crea y se indica el error "Las clases con múltiple herencia no son soportadas por el modelo"
	Destino de Herencia no indicado	
	Contexto	Hay más de una clase creada en la vista de modelado del microservicio
	Evento	Cuando el usuario intenta crear una herencia sin indicar el destino
	Resultado Esperado	La herencia no es creada

Tabla 35 - US-118 Crear Relación Herencia

Módulo de Entidades y Rutas		
US-119 Modificar Relación Herencia		
Descripción	Como usuario quiero modificar las características de una relación de herencia.	
Criterios de Aceptación	Cambiar clase destino	
	Contexto	La relación de herencia entre clases se encuentra creado
	Evento	Cuando el usuario arrastra el destino de la relación de herencia a otra clase
	Resultado Esperado	La relación de herencia se modifica apuntando a la nueva clase destino
	Destino de Herencia no indicado	

	Contexto	Hay más de una clase creada en la vista de modelado del microservicio, y hay relaciones de herencia creadas.
	Evento	Cuando el usuario intenta cambiar el destino de la herencia sin indicar la clase destino.
	Resultado Esperado	La herencia no es modificada.

Tabla 36 - US-119 Modificar Relación Herencia

Módulo de Entidades y Rutas		
US-120 Eliminar Relación Herencia		
Descripción	Como usuario quiero eliminar una relación de herencia para que no sea parte del modelo.	
Criterios de Aceptación	Eliminación de Relación de Herencia	
	Contexto	La Herencia entre las clases se encuentra creada
	Evento	Cuando el usuario presiona eliminar herencia
	Resultado Esperado	La herencia entre las clases se elimina de la vista de modelado del microservicio

Tabla 37 - US-120 Eliminar Relación Herencia

Módulo de Entidades y Rutas	
US-121 Crear Relación Direccional	
Descripción	Como usuario quiero crear una relación direccional entre clases dentro del mismo modelo o entre entidades de distintos microservicios.

Criterios de Aceptación	Creación de Relación Direccional dentro del mismo microservicio	
	Contexto	Hay al menos una clase creada en la vista de modelado del microservicio
	Evento	Cuando el usuario conecta las clase(puede ser entre la misma clase) arrastrando la relación direccional e indica: ● nombre: String ● DirectionalRelationType: String ● required: Boolean
	Resultado Esperado	La relación es creada con los valores indicados y se puede observar en la vista de modelado del microservicio.
	Creación de Relación Direccional entre clases de distintos microservicios	
	Contexto	Hay al menos 2 microservicios creados, con al menos una clase creada dentro
	Evento	Cuando el usuario arrastra la relación direccional externa e indica: ● nombre: String ● DirectionalRelationType: String ● required: Boolean
	Resultado Esperado	Se pide indicar el microservicio y clase dentro de este y se crea la relación direccional.
	Destino de Relación no indicado	
	Contexto	Hay al menos una clase creada en la vista de modelado del microservicio
	Evento	Cuando el usuario intenta crear una relación sin indicar el destino
	Resultado Esperado	La relación no es creada

Tabla 38 - US-121 Crear Relación Direccional

<table>
<tr><th colspan="3">Módulo de Entidades y Rutas</th></tr>
<tr><td colspan="3">US-122 Modificar Relación Direccional</td></tr>
<tr><td>Descripción</td><td colspan="2">Como usuario quiero modificar las características de una relación direccional.</td></tr>
<tr><td rowspan="10">Criterios de Aceptación</td><td colspan="2">Modificación de Relación Direccional</td></tr>
<tr><td>Contexto</td><td>La relación direccional entre clases se encuentra creada.</td></tr>
<tr><td>Evento</td><td>Cuando el usuario presiona la relación direccional y completa el formulario con los datos:
nombre: String
DirectionalRelationType: String
required: Boolean</td></tr>
<tr><td>Resultado Esperado</td><td>La Relación direccional es modificada exitosamente.</td></tr>
<tr><td colspan="2">Relación Direccional con nombre repetido entre 2 clases</td></tr>
<tr><td>Contexto</td><td>Hay clases creadas y con una relación direccional entre ellas</td></tr>
<tr><td>Evento</td><td>Cuando el usuario intenta modificar una relación entre estas 2 clases colocando el mismo nombre que una relación que ya existe entre estas clases.</td></tr>
<tr><td>Resultado Esperado</td><td>La relación no es modificada y se indica el error "La Relación Direccional entre clases no puede tener el mismo nombre que una relación ya existente entre ellas"</td></tr>
<tr><td colspan="2">Destino de Relación Direccional no indicado</td></tr>
<tr><td>Contexto</td><td>Hay al menos una clase creada en la vista de modelado del microservicio y hay relaciones direccionales creadas.</td></tr>
<tr><td>Evento</td><td>Cuando el usuario intenta cambiar el destino de la Relación Direccional sin indicar la clase destino.</td></tr>
</table>

	Resultado Esperado	La Relación Direccional no es modificada.

Tabla 39 - US-122 Modificar Relación Direccional

Módulo de Entidades y Rutas		
US-123 Eliminar Relación Direccional		
Descripción	Como usuario quiero eliminar una relación direccional para que deje desvincular las clases.	
Criterios de Aceptación	Eliminación de Relación de Herencia	
	Contexto	La relación direccional entre clases se encuentra creada
	Evento	Cuando el usuario presiona eliminar la relación direccional
	Resultado Esperado	La relación entre la/s clase/s se elimina de la vista de modelado del microservicio

Tabla 40 - US-123 Eliminar Relación Direccional

Módulo de Generación de Código Backend		
US-201 Generar código Node - Express		
Descripción	Como usuario quiero generar microservicios con Node con el Framework Express en base a los aspectos detallados en el modelo	
Criterios de Aceptación	Generación de Microservicio Node con Express	
	Contexto	Los microservicios se encuentran cargados con los datos,

		capacidades, y modelo de entidades y rutas completos
	Evento	Cuando el usuario presiona el botón de generar código
	Resultado Esperado	Se generan las carpetas con los distintos microservicios que contiene la configuración de cada microservicio

Tabla 41 - US-201 Generar código Node - Express

Módulo de Generación de Código Backend		
US-202 Generar Código Python		
Descripción	Como usuario quiero generar microservicios con Python en base a los aspectos detallados en el modelo	
Criterios de Aceptación	Generación de Microservicio Python	
	Contexto	Los microservicios se encuentran cargados con los datos, capacidades, y modelo de entidades y rutas completos
	Evento	Cuando el usuario presiona el botón de generar código
	Resultado Esperado	Se generan las carpetas con los distintos microservicios que contiene la configuración de cada microservicio

Tabla 42 - US-202 Generar Código Python

Módulo de generación de Código Infraestructura		
US-301 Generación de Integración Continua		
Descripción	Como usuario quiero generar el código de integración continua con los aspectos detallados en el modelo.	
Criterios de Aceptación	Generación de Integración Continua	
	Contexto	Los atributos correspondientes a la integración continua se encuentran indicados en la vista de modelado

	Evento	Cuando el usuario presiona el botón de generar código
	Resultado Esperado	El código de Integración continua es creado en un paquete dentro del archivo generado de todo el proyecto

Tabla 43 - US-301 Generación de Integración Continua

Módulo de generación de Código Infraestructura		
US-302 Generación de configuración de Kubernetes		
Descripción	Como usuario quiero generar el código de configuración de kubernetes de los ambientes y microservicios especificados en el modelo.	
Criterios de Aceptación	Generación de configuración de Kubernetes	
	Contexto	Las configuraciones de kubernetes del ambiente y de cada microservicio se encuentran creadas en la vista de modelado
	Evento	Cuando el usuario presiona el botón de generar código
	Resultado Esperado	El código de Kubernetes es creado en un paquete dentro del archivo generado de todo el proyecto.

Tabla 44 - US-302 Generación de configuración de Kubernetes

Módulo de generación de Código Infraestructura		
US-303 Generación de Docker File		
Descripción	Como usuario quiero generar el código de los archivos de docker para los microservicios detallados en el modelo.	
Criterios de Aceptación	Generación de Dockerfile	
	Contexto	Los atributos necesarios para la generación del dockerFile se encuentran indicados en la vista de modelado
	Evento	Cuando el usuario presiona el botón de generar código

	Resultado Esperado	Los DockerFiles es creado en un paquete dentro del archivo generado de todo el proyecto

Tabla 45 - US-303 Generación de Docker File

Módulo de generación de Código Infraestructura		
US-304 Generación de Integración Continua		
Descripción	Como usuario quiero generar el código de integración continua con los aspectos detallados en el modelo.	
Criterios de Aceptación	Generación de Archivos de Aprovisionamiento Automático con terraform	
	Contexto	Los atributos correspondientes a los ambientes se encuentran creados en la vista de modelado
	Evento	Cuando el usuario presiona el boton de generar codigo
	Resultado Esperado	El Código de Terraform es generado en un paquete dentro del archivo generado de todo el proyecto.

Tabla 46 - US-304 Generación de Integración Continua

Módulo de Reportes		
US-401 Generación de Reporte		
Descripción	Como administrador quiero que al generar un proyecto se cree un reporte indicando información sobre el modelo diagramado y es enviado al servidor de reportes. Solo se debe incluir información sobre el uso de la herramienta.	
Criterios de Aceptación	Generación de Reporte	
	Contexto	Los atributos del modelo se encuentran indicados en la vista de modelado.
	Evento	Cuando el usuario presiona el botón de generar código.

	Resultado Esperado	El reporte es generado dentro del proyecto en la misma carpeta que los archivos de generación para que el usuario pueda corroborar que la información enviada no contiene datos sensibles del proyecto.
	Envío de reporte	
	Contexto	El reporte se encuentra generado.
	Evento	Al completarse el proceso de generación.
	Resultado Esperado	El reporte es enviado al servicio de recolección de informes. No es necesario corroborar si se recibió correctamente.

Tabla 47 - US-401 Generación de Reporte

Módulo de Reportes		
US-402 Guardar Reporte		
Descripción	Como administrador quiero almacenar los reportes recibidos para analizarlos posteriormente.	
Criterios de Aceptación	Recepción de Reporte correcto.	
	Contexto	La conexión con la base de datos está establecida.
	Evento	Al recibir una petición a la ruta de creación de reportes.
	Resultado Esperado	Se agrega la fecha de creación y se guarda en la base de datos exitosamente.
	Formato de Reporte erróneo	
	Contexto	La conexión con la base de datos está establecida.
	Evento	Al recibir una petición a la ruta de creación de reportes con datos que no coinciden con el formato de reporte establecido.

	Resultado Esperado	Se agrega la fecha de creación y se guarda en la base de datos exitosamente.

Tabla 48 - US-402 Guardar Reporte

Módulo de Reportes		
US-403 Estadística de Ambientes		
Descripción	Como administrador quiero visualizar información sobre cuánto se está utilizando cada ambiente de la herramienta filtrando por rango de fecha.	
Criterios de Aceptación	Visualización de Estadísticas de ambientes	
	Contexto	Hay reportes cargados en el servicio de recolección y el administrador se encuentra logueado en la plataforma de visualización de dashboard.
	Evento	Al visualizar el dashboard e indicar un rango de fecha.
	Resultado Esperado	Se muestra un diagrama de barras con la cantidad de veces que se ha utilizado cada ambiente para el rango de fecha indicado.
	Cambio de rango de fecha	
	Contexto	Hay reportes cargados en el servicio de recolección y el administrador se encuentra logueado en la plataforma de visualización de dashboard.
	Evento	Al cambiar el rango de fechas.
	Resultado Esperado	Se llama al sistema de recolección de reportes, se obtienen los nuevos datos para el nuevo rango de fecha indicado y se actualiza el gráfico de barras.
	Rango de fecha invalido	
	Contexto	El administrador está logueado en la plataforma de visualización de dashboard.

	Evento	Al indicar un rango de fecha invalido
	Resultado Esperado	El servicio de recolección de informes responde con un error indicando que se debe corregir el rango de fechas.
	Datos no encontrados o vacíos	
	Contexto	No hay registros cargados o con fechas que no coinciden con el rango de fecha indicado.
	Evento	Al visualizar el dashboard.
	Resultado Esperado	El gráfico se presenta vacío.

Tabla 49 - US-403 Estadística de Ambientes

Módulo de Reportes		
US-404 Estadística de Uso		
Descripción	Como administrador quiero visualizar cuantas generaciones se realizan históricamente para analizar el estado de uso del producto.	
Criterios de Aceptación	Visualización de Estadísticas de uso	
	Contexto	Hay reportes cargados en el servicio de recolección y el administrador se encuentra logueado en la plataforma de visualización de dashboard.
	Evento	Al visualizar el dashboard e indicar un rango de fecha.
	Resultado Esperado	Se muestra un diagrama histórico con la cantidad de generaciones para cada día dentro del rango de fechas indicado.
	Cambio de rango de fecha	
	Contexto	Hay reportes cargados en el servicio de recolección y el administrador se encuentra logueado en la plataforma de

		visualización de dashboard.
	Evento	Al cambiar el rango de fechas.
	Resultado Esperado	Se llama al sistema de recolección de reportes, se obtienen los nuevos datos para el nuevo rango de fecha indicado y se actualiza el gráfico del histórico.
	Rango de fecha invalido	
	Contexto	El administrador está logueado en la plataforma de visualización de dashboard.
	Evento	Al indicar un rango de fecha invalido.
	Resultado Esperado	El servicio de recolección de informes responde con un error indicando que se debe corregir el rango de fechas.
	Datos no encontrados o vacíos	
	Contexto	No hay registros cargados o con fechas que no coinciden con el rango de fecha indicado.
	Evento	Al visualizar el dashboard.
	Resultado Esperado	El gráfico se presenta vacío.

Tabla 50 - US-403 Estadística de Uso

Módulo de Reportes	
US-405 Estadística de Lenguajes	
Descripción	Como administrador quiero visualizar información sobre cuánto se está utilizando cada lenguaje de la herramienta.
Criterios de	Visualización de Estadísticas de Lenguajes

Aceptación	Contexto	Hay reportes cargados en el servicio de recolección y el administrador se encuentra logueado en la plataforma de visualización de dashboard.
	Evento	Al visualizar el dashboard e indicar un rango de fecha.
	Resultado Esperado	Se muestra un diagrama de torta con el porcentaje y cantidad de veces que se a utilizado cada lenguaje/framework para el rango de fecha indicado.
	Cambio de rango de fecha	
	Contexto	Hay reportes cargados en el servicio de recolección y el administrador se encuentra logueado en la plataforma de visualización de dashboard.
	Evento	Al cambiar el rango de fechas.
	Resultado Esperado	Se llama al sistema de recolección de reportes, se obtienen los nuevos datos para el nuevo rango de fecha indicado y se actualiza el gráfico de torta.
	Rango de fecha invalido	
	Contexto	El administrador está logueado en la plataforma de visualización de dashboard.
	Evento	Al indicar un rango de fecha invalido.
	Resultado Esperado	El servicio de recolección de informes responde con un error indicando que se debe corregir el rango de fechas.
	Datos no encontrados o vacíos	
	Contexto	No hay registros cargados o con fechas que no coinciden con el rango de fecha indicado.
	Evento	Al visualizar el dashboard.

	Resultado Esperado	El gráfico se presenta vacío.

Tabla 51 - US-405 Estadística de Lenguajes

Módulo de Reportes		
US-406 Estadística de Capacidad de microservicio		
Descripción	Como administrador quiero visualizar información sobre cuánto se está utilizando cada capacidad de microservicio de la herramienta.	
Criterios de Aceptación	Visualización de Estadísticas de Capacidades	
	Contexto	Hay reportes cargados en el servicio de recolección y el administrador se encuentra logueado en la plataforma de visualización de dashboard.
	Evento	Al visualizar el dashboard e indicar un rango de fecha.
	Resultado Esperado	Se muestra un diagrama de barras con la cantidad de veces que se ha utilizado cada capacidad de microservicio para el rango de fecha indicado.
	Cambio de rango de fecha	
	Contexto	Hay reportes cargados en el servicio de recolección y el administrador se encuentra logueado en la plataforma de visualización de dashboard.
	Evento	Al cambiar el rango de fechas.
	Resultado Esperado	Se llama al sistema de recolección de reportes, se obtienen los nuevos datos para el nuevo rango de fecha indicado y se actualiza el gráfico de barra.
	Rango de fecha invalido	
	Contexto	El administrador está logueado en la plataforma de visualización de dashboard.

	Evento	Al indicar un rango de fecha invalido.
	Resultado Esperado	El servicio de recolección de informes responde con un error indicando que se debe corregir el rango de fechas.
	Datos no encontrados o vacíos	
	Contexto	No hay registros cargados o con fechas que no coinciden con el rango de fecha indicado.
	Evento	Al visualizar el dashboard.
	Resultado Esperado	El gráfico se presenta vacío.

Tabla 52 - US-406 Estadística de Capacidad de microservicio

Módulo de Reportes		
US-407 Estadística de Entidades		
Descripción	Cómo administrar quiero visualizar información sobre el promedio de atributos y relaciones para las entidades.	
Criterios de Aceptación	Visualización de Estadísticas de Entidades	
	Contexto	Hay reportes cargados en el servicio de recolección y el administrador se encuentra logueado en la plataforma de visualización de dashboard.
	Evento	Al visualizar el dashboard e indicar un rango de fecha.
	Resultado Esperado	Se muestra un diagrama de barras con el promedio de atributos y relaciones de las entidades para el rango de fecha indicado.
	Cambio de rango de fecha	
	Contexto	Hay reportes cargados en el servicio de recolección y el administrador se encuentra logueado en la plataforma de

		visualización de dashboard.
	Evento	Al cambiar el rango de fechas.
	Resultado Esperado	Se llama al sistema de recolección de reportes, se obtienen los nuevos datos para el nuevo rango de fecha indicado y se actualiza el gráfico de barra.
	Rango de fecha invalido	
	Contexto	El administrador está logueado en la plataforma de visualización de dashboard.
	Evento	Al indicar un rango de fecha invalido.
	Resultado Esperado	El servicio de recolección de informes responde con un error indicando que se debe corregir el rango de fechas.
	Datos no encontrados o vacíos	
	Contexto	No hay registros cargados o con fechas que no coinciden con el rango de fecha indicado.
	Evento	Al visualizar el dashboard.
	Resultado Esperado	El gráfico se presenta vacío.

Tabla 53 - US-407 Estadística de Entidades

Módulo de Reportes	
US-408 Estadística de Generación	
Descripción	Como administrador quiero visualizar la cantidad de usuarios, cantidad de generaciones y cantidad de generaciones por proyecto para analizar cuánto se utiliza la herramienta.
Criterios de	Visualización de Estadísticas de Generación

Aceptación	Contexto	Hay reportes cargados en el servicio de recolección y el administrador se encuentra logueado en la plataforma de visualización de dashboard.
	Evento	Al visualizar el dashboard e indicar un rango de fecha.
	Resultado Esperado	Se muestra una tabla con la cantidad de usuarios, el promedio de generaciones por usuario y generaciones por proyecto para el rango de fecha indicado.
	Cambio de rango de fecha	
	Contexto	Hay reportes cargados en el servicio de recolección y el administrador se encuentra logueado en la plataforma de visualización de dashboard.
	Evento	Al cambiar el rango de fechas.
	Resultado Esperado	Se llama al sistema de recolección de reportes, se obtienen los nuevos datos para el nuevo rango de fecha indicado y se actualiza la tabla.
	Rango de fecha invalido	
	Contexto	El administrador está logueado en la plataforma de visualización de dashboard.
	Evento	Al indicar un rango de fecha invalido.
	Resultado Esperado	El servicio de recolección de informes responde con un error indicando que se debe corregir el rango de fechas.
	Datos no encontrados o vacíos	
	Contexto	No hay registros cargados o con fechas que no coinciden con el rango de fecha indicado.
	Evento	Al visualizar el dashboard.

	Resultado Esperado	El gráfico se presenta vacío.

Tabla 54 - US-408 Estadística de Generación

3) Pantallas y reportes.

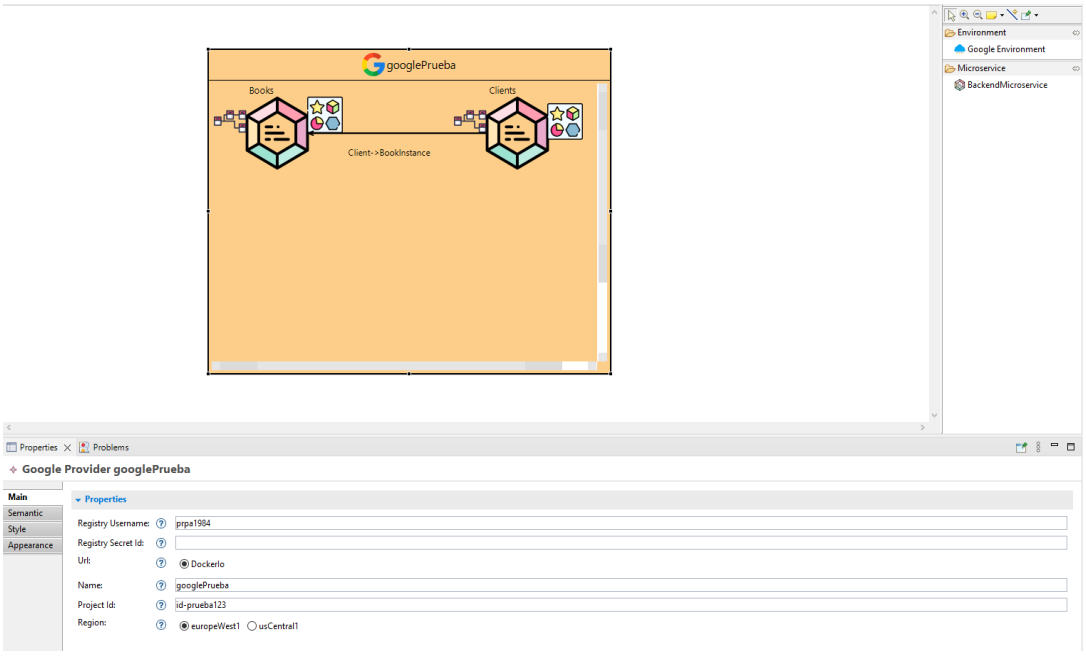


Figura 81 - ABM Ambiente (US-001, US-002, US-003)

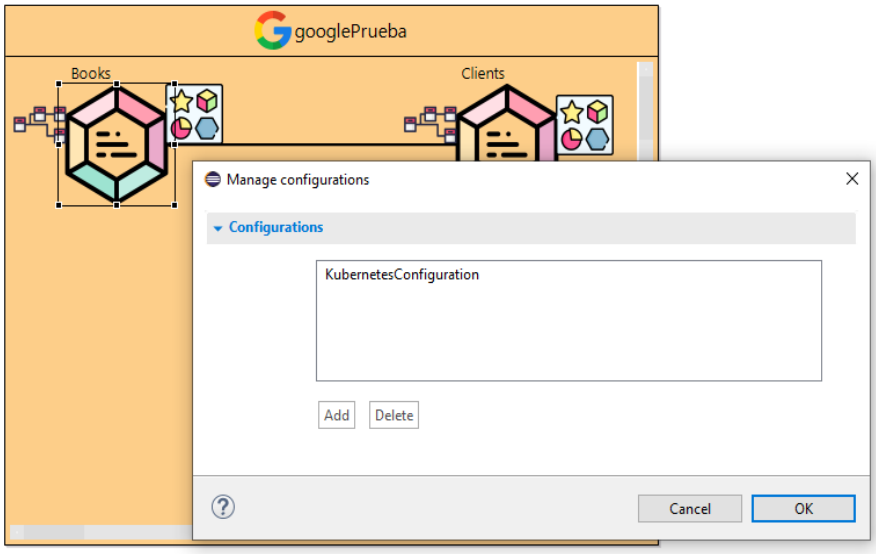


Figura 82 – ABM Configuración Kubernetes del Ambiente (US-004, US-005, US-006)

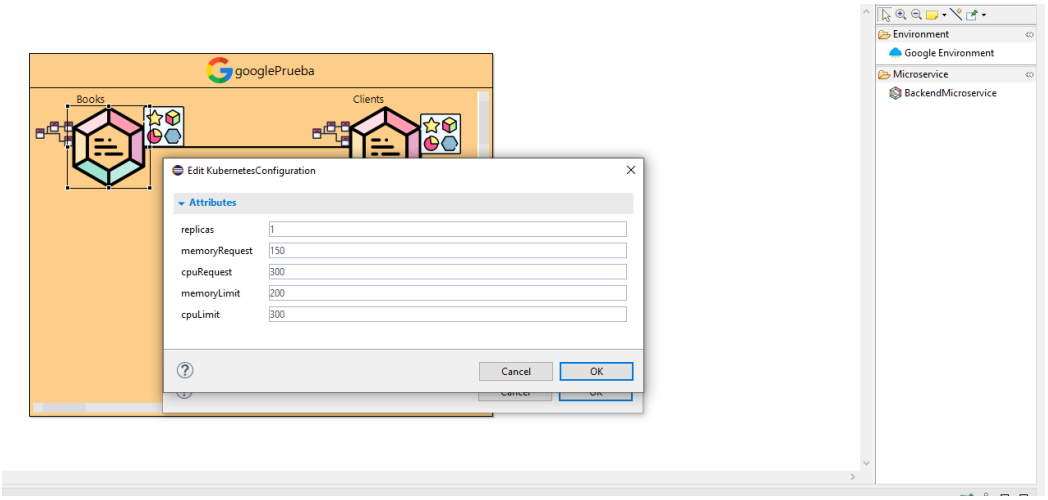


Figura 83 - ABM Configuración de Kubernetes del Microservicio (US-007, US-008, US-009)

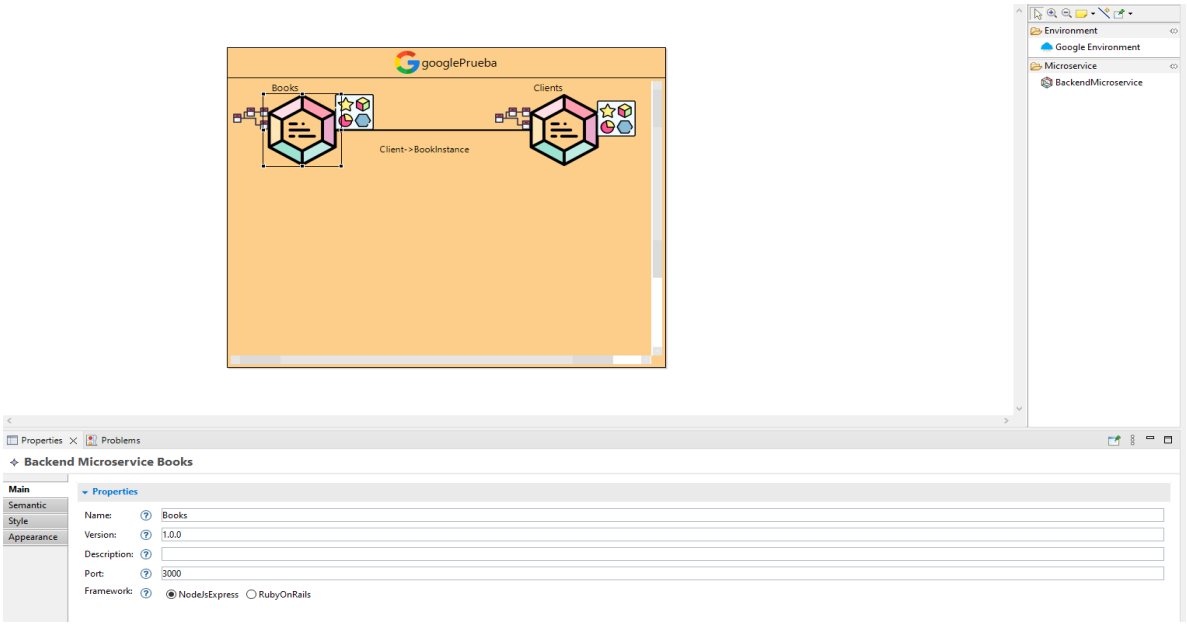


Figura 84 - ABM Microservicio Backend (US-0010, US-0011, US-0012)

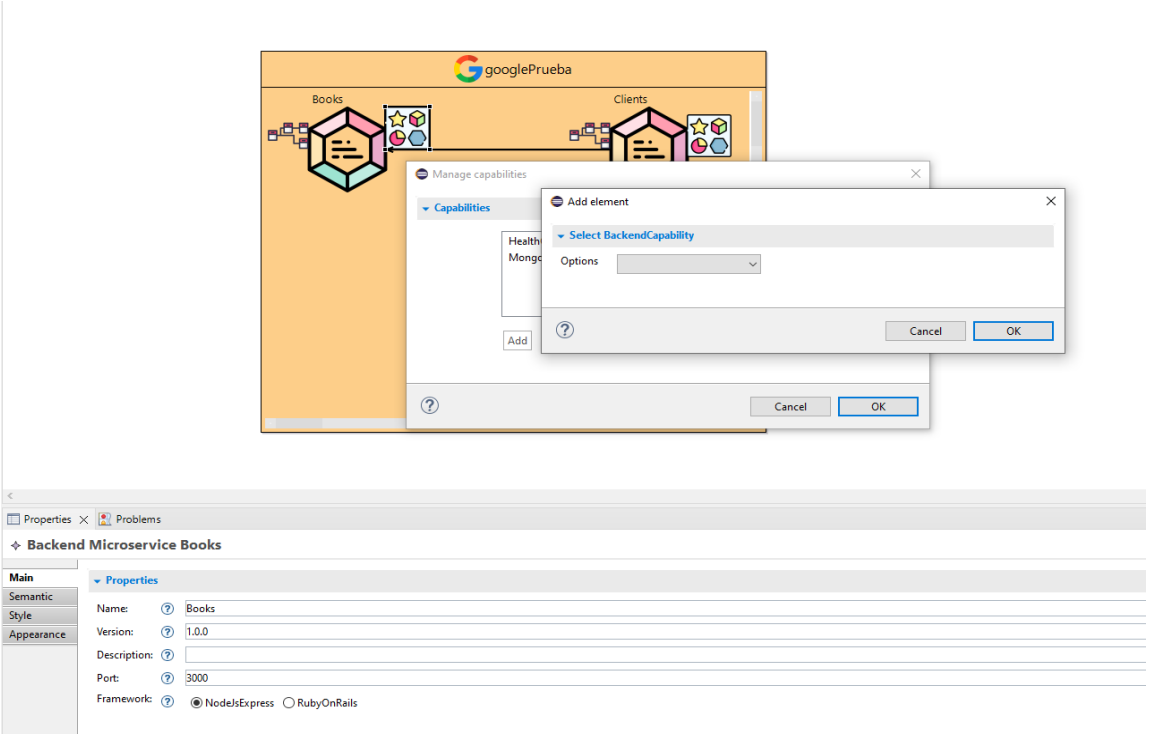


Figura 85 - Ventana Capacidades (US-014), Agregar capacidades (US-015)

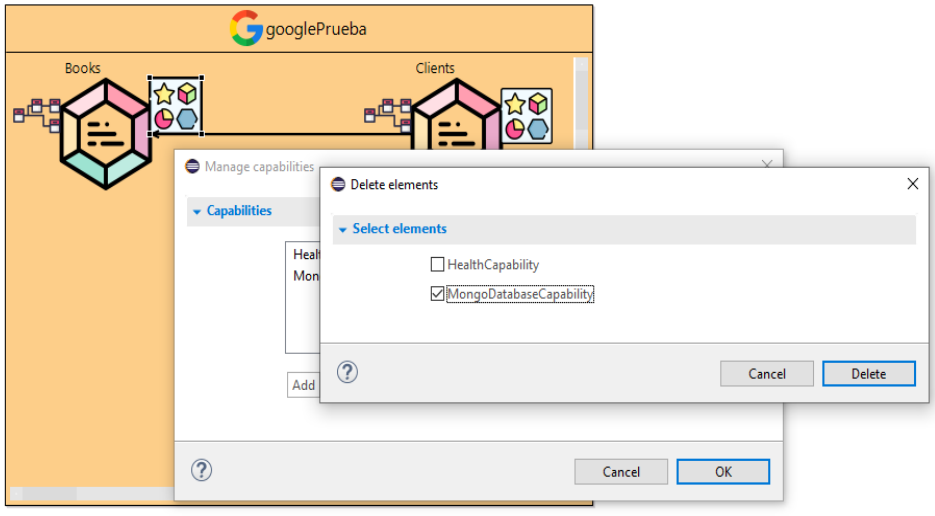


Figura 86 - Modificar Capacidad (US-016), Eliminar Capacidad/es (US-017).

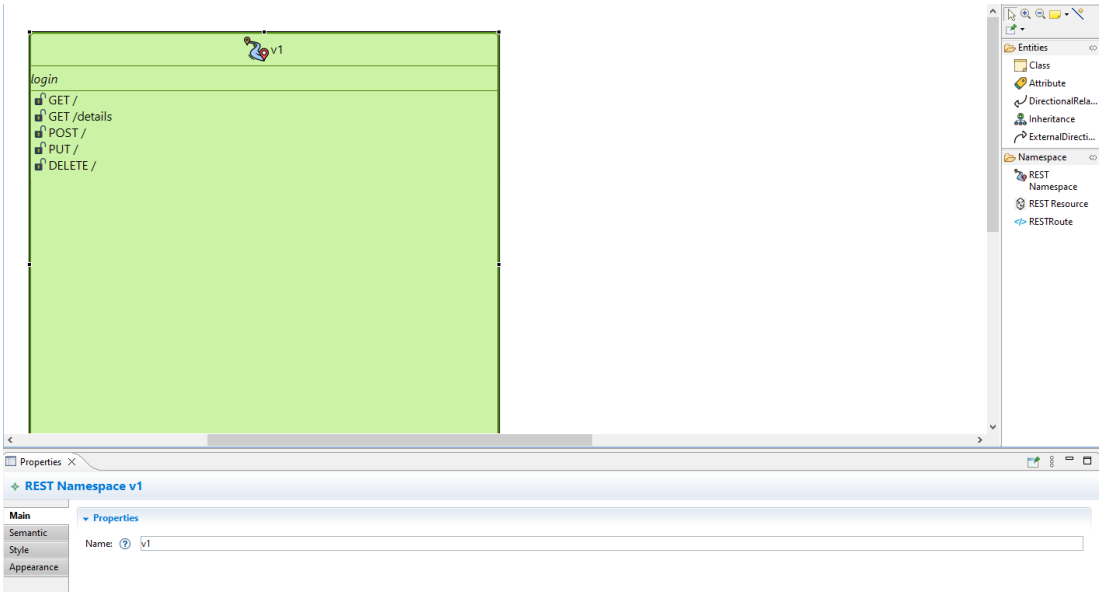


Figura 87 – Vista de Modelado de Microservicio (US-101) y ABM REST namespace (US-102, US-103, US-104)

Figura 88 - ABM REST resource (US-105, US-106, US-107)

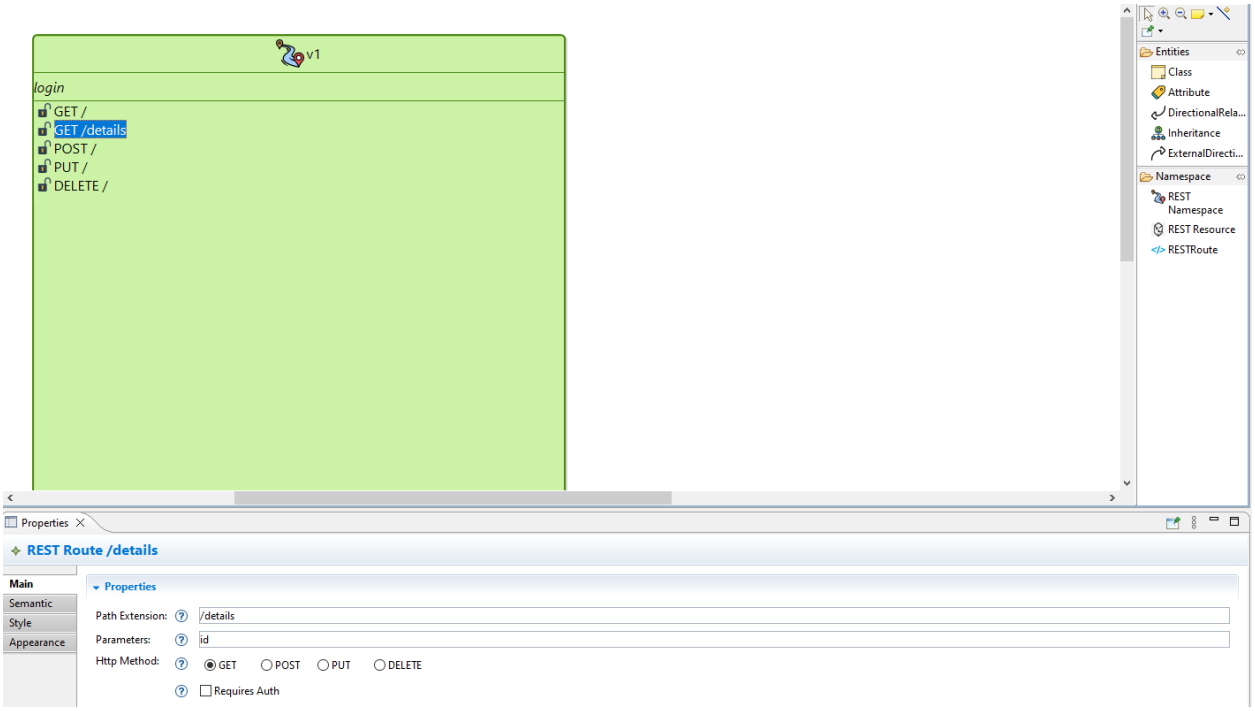


Figura 89 - ABM REST Route (US-108, US-109, US-110)

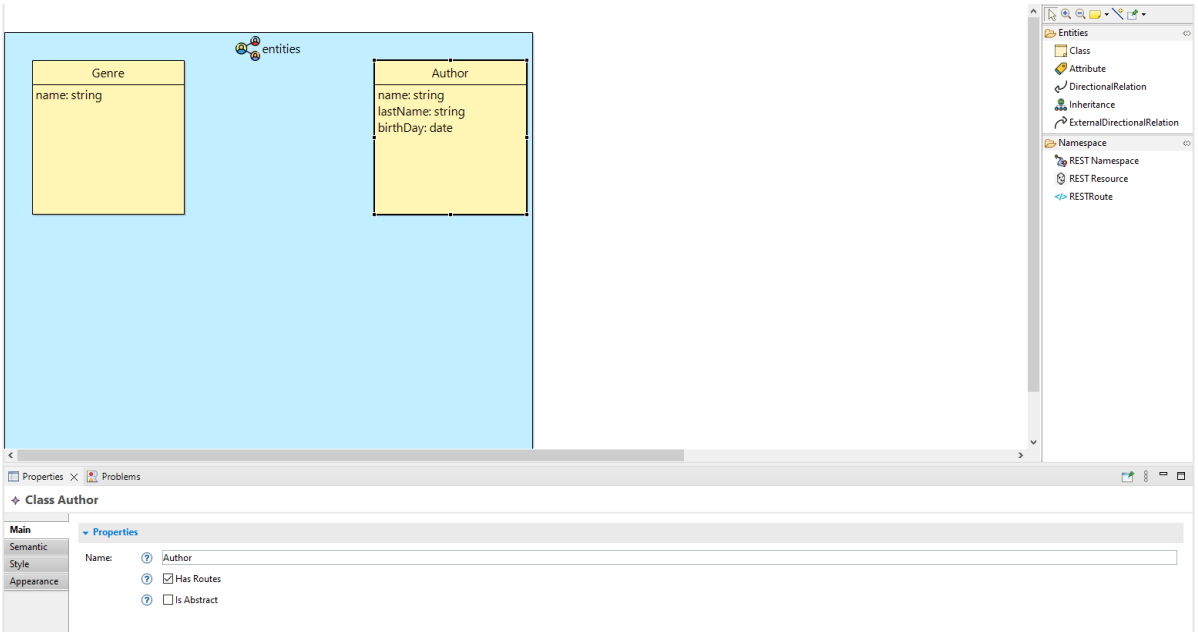


Figura 90 - ABM Clase (US-112, US-113, US-114)

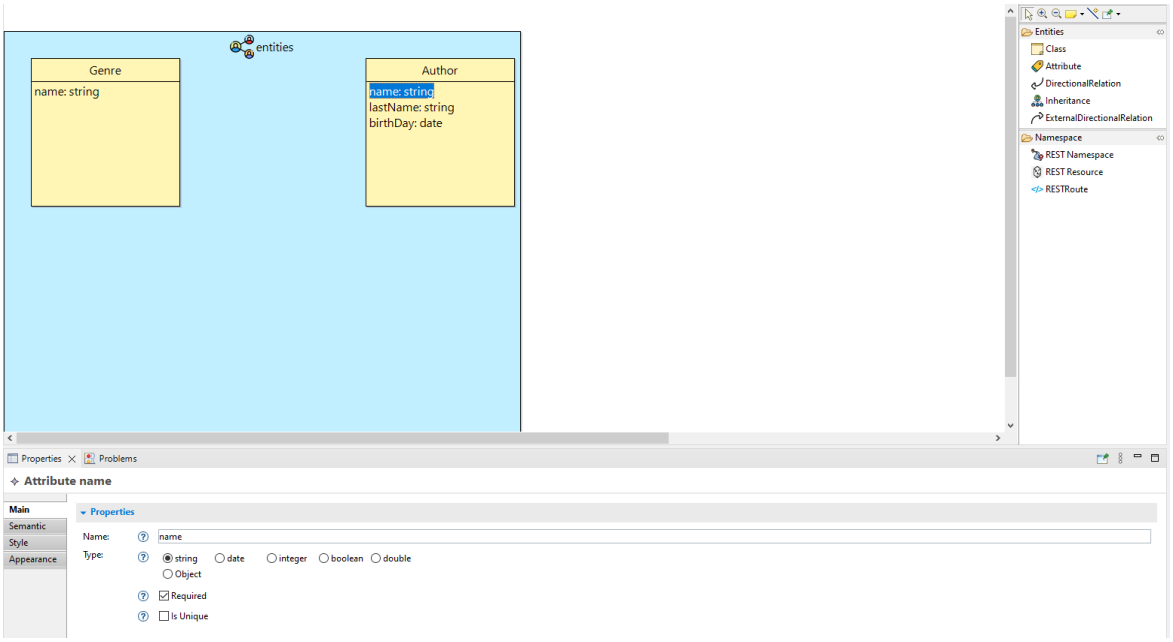


Figura 91 - ABM Atributo (US-115, US-116, US-117)

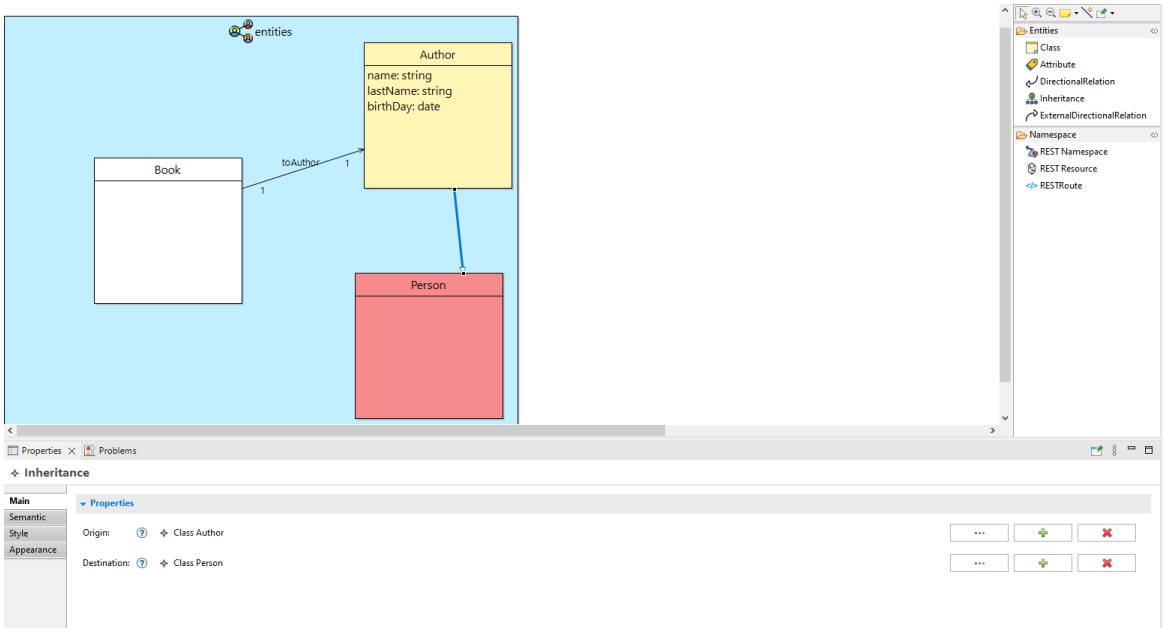


Figura 92 - ABM Relación Herencia (US-118, US-119, US-120)

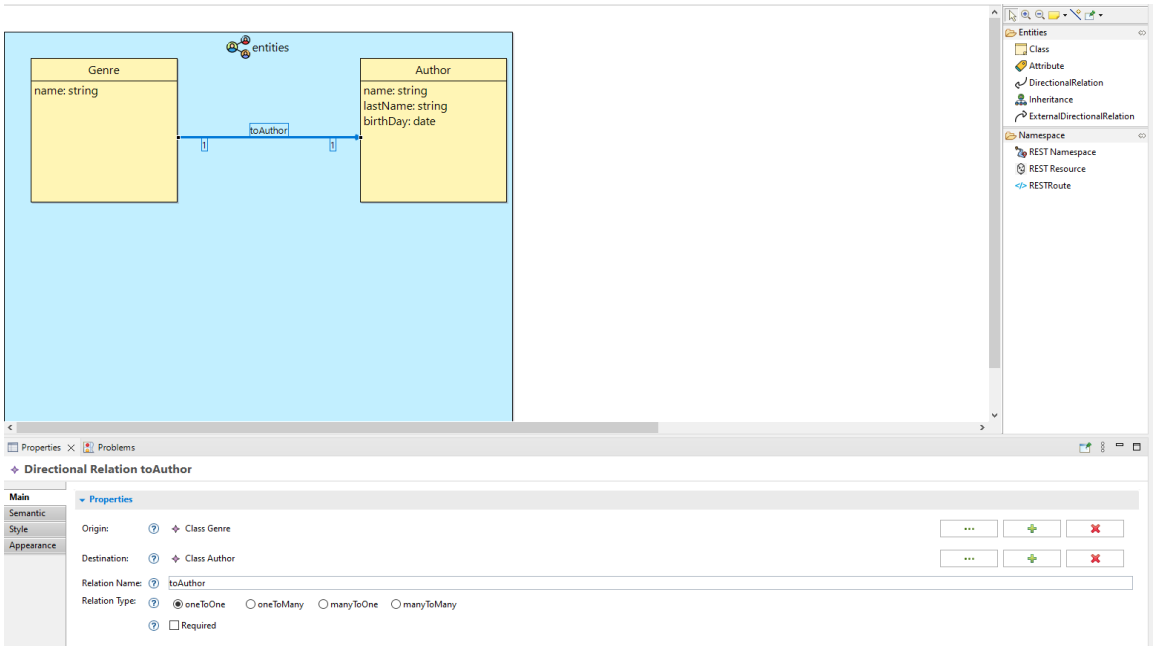


Figura 93 - Crear, Modificar Relación Direccional (US-121, US-122)

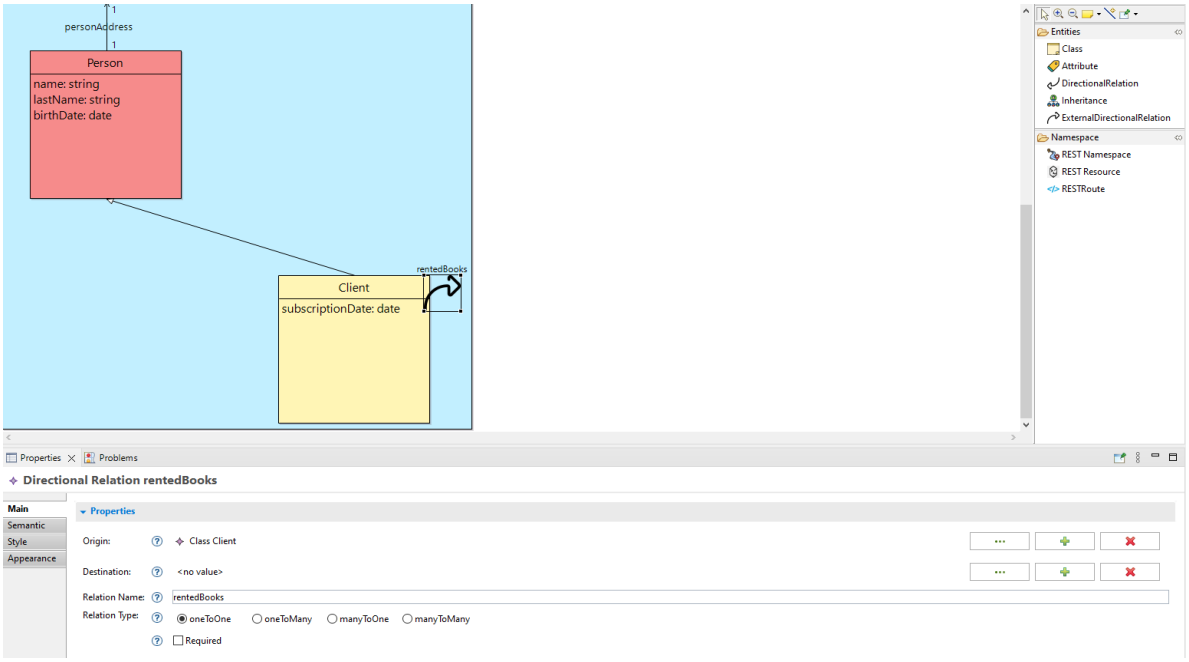


Figura 94 - Eliminar Relación Direccional (US-123)

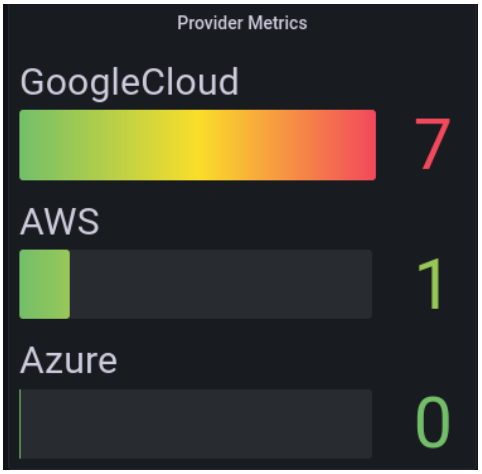


Figura 95 - Estadística de Ambientes (US-403)

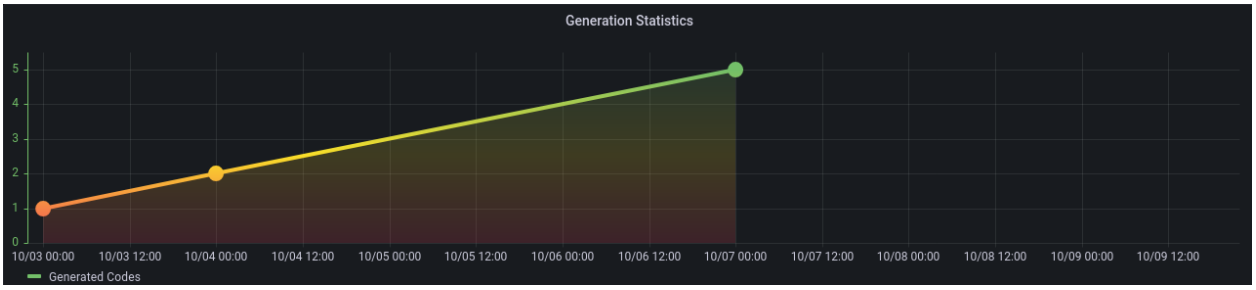


Figura 96 - Estadística de Uso (US-404)

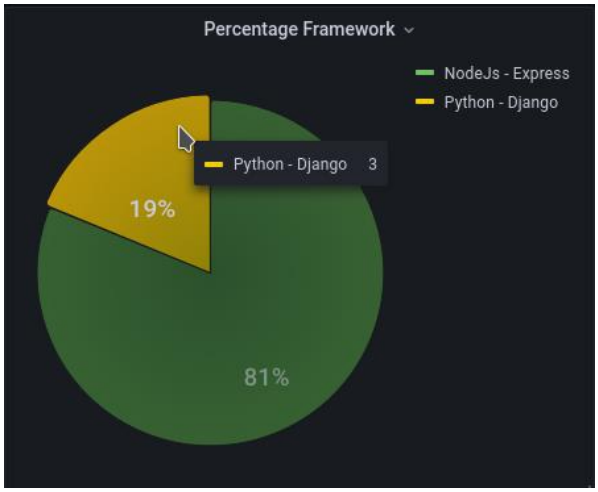


Figura 97 - Estadística de Lenguajes (US-405)

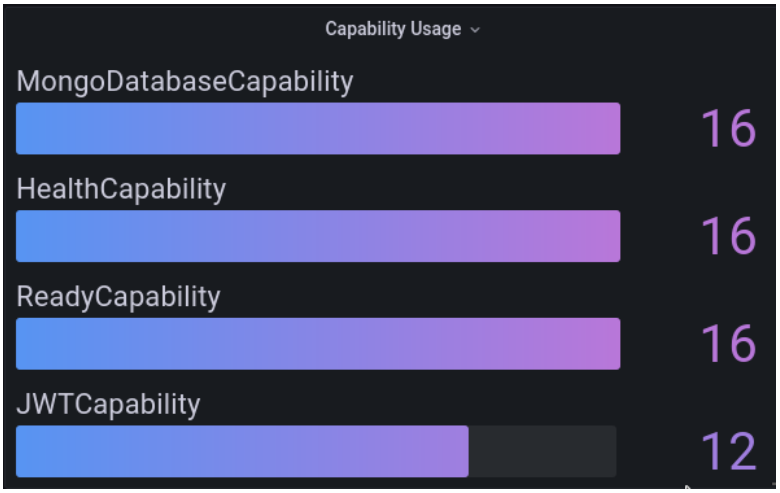


Figura 98 - Estadística de Capacidad de microservicio (US-406)

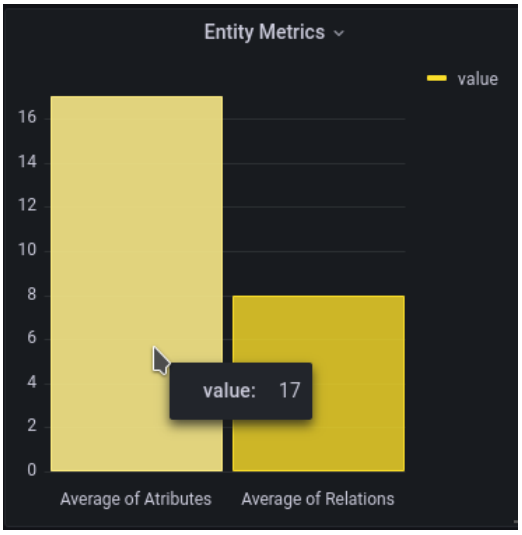


Figura 99 - Estadística de Entidades (US-407)

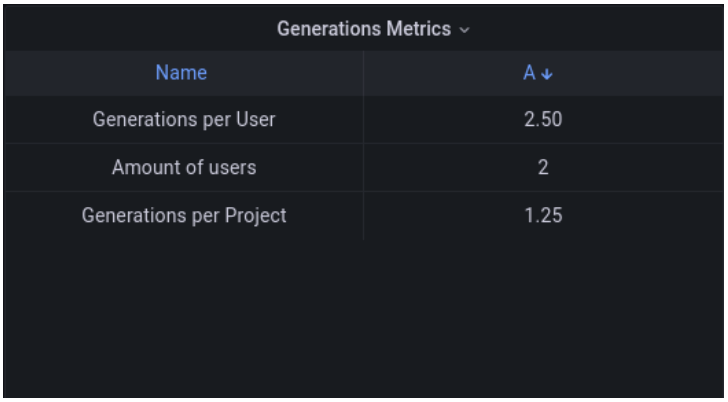


Figura 100 - Estadística de Generación (US-408)

4) Modelo de datos.

El modelo de datos de Accelerate se divide en tres componentes:

- **Componente de infraestructura:** Modela todo lo correspondiente al aprovisionamiento de clústeres en la nube.
- **Componente de microservicios:** Define el comportamiento del modelado respecto a la arquitectura de microservicios.
- **Componente de código:** Especifica la información del funcionamiento interno de cada microservicio, utilizando diagramas de clases y definiendo rutas.

Modelo de datos: Infraestructura

El componente de infraestructura contiene las siguientes clases y atributos:

- **ProvidersDiagram**: Clase raíz del metamodelo. Representa un conjunto de proveedores en la nube.
- **Provider**: Representa un proveedor en la nube mediante una clase abstracta, es decir, cada proveedor tendrá su implementación.
- **GoogleProvider**: Implementación del proveedor Google.
    - projectId: Identificador del proyecto en el que se va a realizar el despliegue.
    - region: Región de despliegue.
- **Configuration**: Representación de las configuraciones para los proveedores en la nube.

- **KubernetesConfiguration**: Representación de la configuración necesaria para un clúster de Kubernetes en un proveedor.
- **GoogleKubernetesConfiguration**: Implementación de KubernetesConfig para GKE (Google Kubernetes Engine).
    - poolName: Nombre del grupo de nodos a crear.
    - nodeMachineType: Tipo de máquina que los nodos utilizarán.
    - nodeMinCount: Mínima cantidad de nodos que tendrá el grupo de nodos.
    - nodeMaxCount: Máxima cantidad de nodos que tendrá el grupo de nodos.
    - nodeDiskSize: Tamaño de disco de los nodos (medido en GB).

Ver “Anexo 4: accelerateMLI”

Modelo de datos: Microservicios

El componente de microservicios contiene las siguientes clases y atributos:

- **Environment**: Representa un ambiente compuesto de microservicios.
    - name: Nombre del ambiente de microservicios.
- **Microservice**: Representa un microservicio de cualquier tipo.
    - name: Nombre del microservicio.
    - version: Versión del microservicio.
    - description: Descripción del microservicio
- **Relation**: Representa una relación entre microservicios. Está relacionada con un microservicio destino y con un microservicio origen.
- **Configuration**: Representa un elemento de configuración para un microservicio.

- **KubernetesConfiguration**: Implementación de Configuration. Representa la configuración de un Deployment de Kubernetes para un microservicio.
    - replicas: Cantidad de instancias de dicho microservicio que se ejecutarán en simultáneo. Valor por defecto: 1.
    - memoryRequest: Cantidad de memoria mínima que necesita el microservicio (medida en MB). Valor por defecto: 256.
    - cpuRequest: Cantidad de CPU mínima que necesita el microservicio (medida en MHz). Valor por defecto: 500.
    - memoryLimit: Cantidad de memoria máxima que puede solicitar el microservicio (medida en MB). Valor por defecto: 512.
    - cpuyLimit: Cantidad de CPU máxima que puede solicitar el microservicio (medida en MHz). Valor por defecto: 1000.
- **FunctionalMicroservice**: Representa un microservicio de tipo Funcional, es decir, cuya funcionalidad es de utilidad para el usuario.
    - framework: Framework utilizado para la generación automática de código.
- **BackendMicroservice**: Representa un microservicio funcional backend, es decir, que actúa como servidor.
- **BackendCapability**: Representa una capacidad de un microservicio funcional backend.
- **DatabaseCapability**: Capacidad de un microservicio para el acceso a una base de datos.
- **NoSQLDatabaseCapability**: Categoría NoSQL para una capacidad Database.

- **MongoDatabaseCapability**: Implementación de la capacidad de acceso a una base de datos NoSQL para mongoDB [13].
    - username: Nombre de usuario para acceder a la base de datos.
    - password: Contraseña para acceder a la base de datos.
    - host: URL en la que se encuentra la base de datos.
    - databaseName: Nombre de la base de datos a la que se quiere acceder.
- **RoutingCapability**: Capacidad de un microservicio con respecto a su routeo.
- **ReadyCapability**: Implementación de la capacidad de routeo que permite realizar verificaciones a un microservicio con respecto a su disponibilidad, logrando la distribución de tráfico entre microservicios disponibles solamente.
    - readyRoute: Ruta de verificación. Valor por defecto: ready.
    - initialDelay: Demora inicial para el comienzo de las verificaciones (medida en segundos). Valor por defecto: 15.
    - period: Frecuencia de verificación (medida en segundos). Valor por defecto: 5.
- **HealthCapability**: Implementación de la capacidad de routeo que permite realizar verificaciones a un microservicio con respecto a su salud, reiniciando aquellos microservicios inutilizados.
    - healthRoute: Ruta de verificación. Valor por defecto: health.
    - initialDelay: Demora inicial para el comienzo de las verificaciones (medida en segundos). Valor por defecto: 15.

- period: Frecuencia de verificación (medida en segundos). Valor por defecto: 5.

Ver “Anexo 5: accelerateMLS”

Modelo de datos: Código

El componente de código está compuesto por las siguientes clases y atributos:

- **Diagram:** Clase raíz del metamodelo. Representa un diagrama de clases y una implementación de un mecanismo de comunicación.
- **Namespace**: Un namespace es un agrupamiento de recursos utilizados para la comunicación.
  - name: Nombre del namespace.
- **RESTNamespace:** Implementación REST de namespace. Agrupa recursos REST.
- **RESTResource:** Grupo de rutas de acceso al microservicio utilizando REST.
  - name**:** Nombre del recurso.
- **RESTRoute:** Ruta o endpoint utilizada para acceder al microservicio. La generación automática correspondiente creará solamente la estructura, dejando la lógica contenida sin implementación.
  - httpMethod**:** Método de acceso HTTP. Valor por defecto = GET.
  - pathExtension**:** Extensión del path establecido por el recurso. Valor por defecto = /.

  - requiresAuth**:** Define si el usuario debe estar autenticado o no para poder realizar una llamada a la ruta. Valor por defecto = Falso.
  - parameters**:** Parámetros de ruta separados por punto y coma.
- **EntityPackage:** Paquete de entidades compuesto por clasificadores.
  - name**:** Nombre del paquete.
- **Classifier**: Implementación abstracta que refiere a clases o interfaces UML.
  - name: Nombre del clasificador.
- **Class:** Implementación de clasificador representando a una clase. Contiene atributos.
  - isAbstract**:** Define si la clase es abstracta o no. Valor por defecto = Falso.
  - hasRoutes**:** Especifica si se generarán o no rutas de Alta, Baja o Modificación para la clase. Valor por defecto = Verdadero.
- **Attribute:** Atributo de una clase.
  - type**:** Tipo de dato.
  - required**:** Especifica si el atributo debe estar presente o no al momento de guardar una instancia en base de datos. Valor por defecto = Verdadero.
  - name**:** Nombre del atributo.
  - isUnique**:** Define si el valor del atributo puede repetirse en una tabla en base de datos, es decir, si debe ser único. Valor por defecto = Falso.
- **Relation**: Representa una relación entre clases.
- **Inheritance:** Representa una relación de herencia entre dos clases.
- **DirectionalRelation:** Hace referencia a una relación direccional entre dos clases.

- relationName**:** Nombre del atributo en la clase origen que hace referencia a dicha relación.
- required**:** Define si la instancia origen debe estar relacionada a la instancia destino al momento de guardar la primera en base de datos.
- directionalRelationType: Multiplicidad de la relación. Valor por defecto: OneToOne.

Ver “Anexo 6: accelerateMLC”

Conexión de componentes

Cabe aclarar que, para que los distintos modelos funcionen de manera conjunta, se los debe conectar de alguna manera. La conexión está realizada de la siguiente manera:

- **Conexión infraestructura – microservicios:** La clase Provider del modelo de infraestructura hereda de la clase Environment del modelo de microservicios, logrando que un proveedor contenga microservicios en el modelo.
- **Conexión microservicios – código:** La clase BackendMicroservice del modelo de microservicios hereda de la clase Diagram del modelo de código, permitiendo el modelado interno a través de clases y mecanismos REST.

Modelo de Datos de sistema de Reportes

Como se detalló previamente el reporte es creado por Accelerate durante el proceso de generación de código y enviado al sistema de reportes. Este es un objeto de tipo JSON y es almacenado de esa forma en MongoDB por lo que el modelo de datos del reporte es de tipo documento. A continuación, se detalla esta estructura.

```
{
    "dateGenerated": "Date",    //Fecha del reporte
    "macAddress": "String",     //Dirección MAC del dispositivo
    "projectName": "String",    //Nombre del proyecto que generó el reporte
    "environments": [           //Ambientes modelados
        {
            "environmentName": "String" //Nombre de cada ambiente
        }
    ],
    "backendMicroservices": [   //Microservicios modelados
        {
            "framework": "String",  //Framework de generación
            "capabilities": [       //Capacidades del microservicio
                "typeCapability": "String"  //Tipo de capacidad
            ],
            "amountEntities": "number", //Cantidad de entidades del microservicio
            "amountRoutes": "number"    //Cantidad de rutas del microservicio
        }
    ],
    "entities": [   //Entidades modeladas
        {
            "amountAttributes": "number",   //Cantidad de atributos
            "amountInternalRelations": "number",    //Cantidad de relaciones internas
            "amountExternalRelations": "number"     //Cantidad de relaciones externas
        }
    ]
}
```

Figura 101 - Estructura JSON de reporte

Desarrollo e Implementación

Sistema generador automático de código e infraestructura

Desarrollo e Implementación

Programación y Documentación

En esta sección se explica el código fuente de las plantillas de generación de código de Acceleo [6], en particular la funcionalidad de generación de entidades del sistema en NodeJS, correspondiente a la historia de usuario “Generar código Node - Express”.

El desarrollo de las plantillas de generación automática se desarrolla completamente en el lenguaje Acceleo el cual es una tecnología open source de plantillas de generación de código. Los archivos generados son archivos de nodeJS implementando mongoDB [Figura 1 - Tecnologías de codificación] como base de datos para la persistencia.

Figura 102 - Tecnologías de codificación

Código base para la generación

Similar a como al desarrollar un código de una interfaz gráfica Frontend se debe partir de un diseño en una herramienta gráfica como Figma [15] que le muestra al programador como se debe ver el producto final. Al crear plantillas de generación de código, se parte de un código ejemplo que permite ver con qué debe contar el resultado generado. Este tiene todos los elementos que el programador utiliza posteriormente para escribir la plantilla.

Al escribir la plantilla el desarrollador busca los elementos del código a generar que deben ser dinámicos y cambiar respecto del modelo y cuáles elementos deben ser iguales sin importar el diagrama realizado en la herramienta gráfica.

A continuación se ve uno de estos ejemplos de código utilizados para el desarrollo [Figura 2 - Ejemplo de código para generación].

Este es un ejemplo de cómo se debe ver parte de un controlador para un proyecto en NodeJS. Por un lado se puede observar que los métodos como getAll, getOneById y saveOne deben ser iguales para cualquier entidad ya sea un libro, una factura, etc.

Por otro lado, la entidad a la que se hace referencia dentro del método debe ser distinta dependiendo de las que el usuario cree en el modelador.

Por último el programador debe tener en cuenta que los archivos generados se deben crear correctamente en la carpeta adecuada. Por ejemplo, un controlador debe crearse en la carpeta Controllers del microservicio apropiado.

Figura 103 - Ejemplo de código para generación

Estructura de Código

Para el desarrollo se utilizó una convención al momento del nombrado y organización de los archivos [Figura 3 - Diagrama de Convención de Archivos], esto con el objetivo de mantener la estructura del proyecto ordenada, sencilla de entender y fácil de mantener y escalar en el tiempo al agregar nuevos lenguajes/tecnologías. Esta convención se utiliza a modo de estándar en el proyecto para el desarrollo actual y a futuro. A continuación se encuentra un diagrama ejemplificando la estructura.

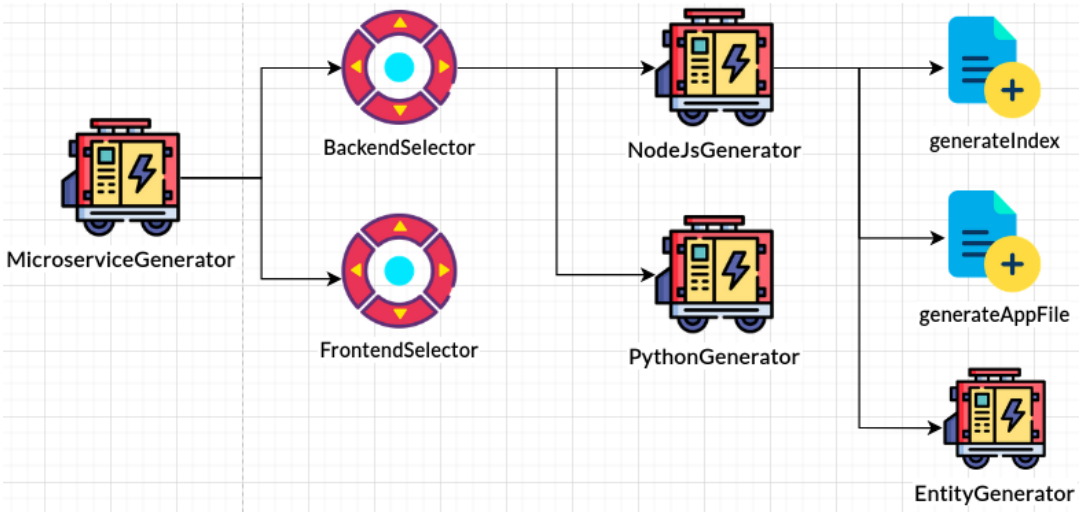


Figura 104 - Diagrama de Convención de Archivos

Convenciones

- generateXX: Los archivos que adoptan la convención “generate” son los encargados de la generación de archivos, contienen la lógica necesaria para generar el código asignado con los detalles especificados en el modelo.

  Ej: generate Index genera el archivo “index.js” del microservicio de NodeJs

- XXSelector: Los archivos que adoptan la convención “Selector” son los encargados de seleccionar módulos/tecnologías que generan código. No incluyen lógica de generación de archivos.

  Ej: backendSelector llama al módulo correspondiente a la tecnología seleccionada para generar el microservicio backend, si se selecciona NodeJS el módulo llamado es NodeJSGenerator.

- XXGenerator: Los archivos que adoptan este nombre cumplen el rol de llamar a los módulos encargados de generar archivos o seleccionar tecnologías. No incluye ninguna lógica sobre generación de código, ya que contiene todas las llamadas a funciones necesarias para generar los archivos de un módulo específico.

  Ej: NodeJSGenerator es el encargado de llamar las funciones necesarias para generar el código del microservicio en NodeJS.

Generación de entidades

La funcionalidad “Generar código Node - Express” involucra la creación automática de todo el contenido del microservicio de node incluyendo configuración de servidor, archivos de dependencias, docker, autenticación, rutas del microservicio y entidades del sistema. Ya que mostrar las plantillas para generar todo el microservicio es demasiado extenso y requiere mostrar demasiado código, se analiza la porción de código que consideramos que aporta más valor a la documentación, la cual es la generación automática de entidades en NodeJS [Figura 4 - Estructura de EntityGeneration].

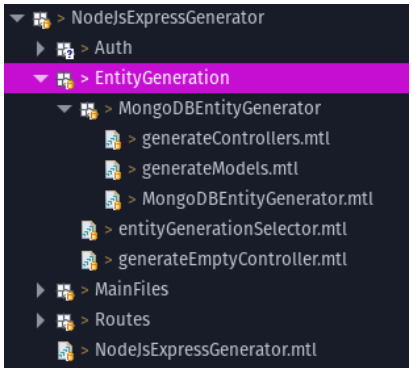


Figura 105 - Estructura de EntityGeneration

La funcionalidad consiste en obtener las entidades del diagrama y generar los modelos y controladores correspondientes. Estos se generan dependiendo la capacidad de base de datos seleccionada en el modelo los cuales establecen la estructura y el ORM (Object Relation Mapper) que se utiliza para mapear la información a datos con la estructura necesaria para guardar en la base de datos.

Es importante destacar que cada archivo contiene un solo método definido el cual se declara de la forma:

```
[template public nombreFuncion(parámetro1: Tipo)] ... comportamiento [/template]
```

Una vez declarados se pueden importar desde cualquier archivo y son activados a través de:

```
[nombreFuncion(Parametro1, Parametro2)/]
```

Código de la funcionalidad EntityGeneration

EntityGeneratorSelector

Es el código encargado de seleccionar el tipo de generador que se encargará de las entidades [Figura 5 - Código EntityGenerator], dependiendo de la capacidad de base de datos seleccionada ya que la estructura de cada uno será distinta.

Este recorre la información del modelo y verifica el tipo de capacidad de base de datos y genera solo aquellas que no sean abstractas.

```
[comment
    Para la generacion de entidades se debe seleccionar la base de datos
    que los almacenara. Por lo que la funcion "entityGenerationSelector"
    encontrada a continuación es la encarga de esto.
/]

[template public entityGenerationSelector(microservice: BackendMicroservice, environment: Provider)]
[comment
    El selector esta a cargo de direccionar al generador
    correspondiente a la base de datos seleccionada
/]

[comment "MongoDB Selector" /]
[if (microservice.contained_capabilities -> selectByKind(DatabaseCapability)->size() > 0)]
    [if (microservice.contained_capabilities -> selectByKind(DatabaseCapability)->first().oclIsTypeOf(MongoDatabaseCapability) <> false)]

        [comment Se itera por cada entidad NO ABSTRACTA del microservicio y se la genera/]
        [for (class : Class | microservice.oclAsType(Diagram).contained_entityPackages -> first().contained_classifiers.oclAsSet() -> selectByType(Class) -> select(entity |
entity.isAbstract = false
        ))]
            [comment Si se selecciona MongoDB se dirige a esta funcion/]
            [MongoDBEntityGenerator(
                class,
                microservice,
                environment
            )/]

        [/for]

    [/if]
[else]

[comment
    Si no hay capacidad de base de datos se generan controladores
    con funcionalidad a implementar por el desarrollador
/]
[for (class : Class | microservice.oclAsType(Diagram).contained_entityPackages -> first().contained_classifiers.oclAsSet() -> selectByType(Class) -> select(entity | entity.isAbstract =
false
))]
    [generateEmptyController(class, microservice, environment)/]
[/for]
[/if]
[/template]
```

Figura 106 - Código EntityGenerator

Observar que si no se selecciona capacidad de base de datos se llama a la función generateEmptyController.

GenerateEmptyController

Cuando no hay base de datos se generan los controladores para cada entidad no abstracta [Figura 6 - Código GenerateEmptyController]. En este caso al no conocer la implementación cada método de los controladores devuelve un mensaje indicando `Route —[Nombre de Ruta]—- for [Nombre de entidad] not Implemented`. El programador debe implementarlo posteriormente según las necesidades del proyecto.

```
[comment encoding = UTF-8 /]
[module generateEmptyController('http://www.example.org/accelerateMLI','http://www.example.org/accelerateMLS', 'http://www.example.org/accelerateMLC')]


[comment En caso de que no se seleccione una base de datos se generan controlladores vacios
         para ser implementados posteriormente por el desarrollador./]
[template public generateEmptyController(entity: Classifier, microservice: BackendMicroservice, environment: Provider)]

[comment Se genera el controlador en la ruta "./ambiente1/Microservices/microservicio1/src/controllers/ejemploController.js"/]
[file ((environment.oclAsType(Environment).name + '/Microservices/'+microservice.oclAsType(Microservice).name+'/src/controllers/'+entity.name+'Controller.js'), false, 'UTF-8'), false,
'UTF-8')]

[comment Se proveen los metodos de acceso getAll, getOneById, saveOne, updateOneById, deleteOneById
         pero su unica funcionalidad es indicar que aun no han sido implementados. /]
export default {
  getAll: async (req, res, next) => {
    res.status(200).json({msg: "Route --getAll-- for [entity.name/] not Implemented"});
  },

  getOneById: async (req, res, next) => {
    res.status(200).json({msg: "Route --getOneById-- for [entity.name/] not Implemented"});
  },

  saveOne: async (req, res, next) => {
    res.status(200).json({msg: "Route --saveOne-- for [entity.name/] not Implemented"});
  },

  updateOneById: async (req, res, next) => {
    res.status(200).json({msg: "Route --updateOneById-- for [entity.name/] not Implemented"});
  },

  deleteOneById: async (req, res, next) => {
    res.status(200).json({msg: "Route --deleteOneById-- for [entity.name/] not Implemented"});
  },
};

[/file]
[/template]
```

Figura 107 - Código GenerateEmptyController

MongoDBEntityGenerator

Si la Capacidad de base de Datos es MongoDB entonces se utiliza el módulo MongoDBEntityGenerator para generar las entidades [Figura 7 - Código MongoDBEntityGenerator], este se encarga de llamar a los generadores de archivos que necesita que en este caso son “generateModels” y “generateControllers”.

```
[comment encoding = UTF-8 /]
[module MongoDBEntityGenerator('http://www.example.org/accelerateMLI','http://www.example.org/accelerateMLS', 'http://www.example.org/accelerateMLC')]

[import NodeJsExpressGenerator::EntityGeneration::MongoDBEntityGenerator::generateModels /]
[import NodeJsExpressGenerator::EntityGeneration::MongoDBEntityGenerator::generateControllers /]

[comment Generacion de entidades para MongoDB /]
[template public MongoDBEntityGenerator(entity: Class, microservice: BackendMicroservice, environment: Provider)]

[comment
    Para MongoDB se requiere 2 modulos:
        -Models: Encargados de definir la estructura con la que se guarda en la base de datos.
        -Controllers: Encargados de brindar metodos de acceso y modificacion de datos.
/]

[generateModels(entity, microservice, environment)/]

[comment
    Los Controllers solo son generados si la entidad fue marcada con el atributo "hasRoutes"
 /]
[if (entity.hasRoutes = true)]
    [generateControllers(entity.oclAsType(Classifier), microservice, environment)/]
[/if]


[/template]
```

Figura 108 - Código MongoDBEntityGenerator

GenerateModels

Se encarga de crear los archivos que contienen la definición de la entidad con sus atributos y entidades para su posterior acceso y creación en una base de datos MongoDB [Figura

8-9 - Código GenerateModels 1-2]. Para cada entidad toma todos sus atributos y relaciones y también todos los atributos y relaciones de su padre en caso de tener una herencia.

```
[comment encoding = UTF-8 /]
[module generateModels('http://www.example.org/accelerateMLI','http://www.example.org/accelerateMLS', 'http://www.example.org/accelerateMLC')]

[template public generateModels(entity: Class, microservice: BackendMicroservice, environment: Provider)]
[comment
    Se genera el modelo en la ruta "./ambiente1/Microservices/microservicio1/src/Models/Ejemplo.js"
 /]
[file ((environment.oclAsType(Environment).name + '/Microservices/'+microservice.oclAsType(Microservice).name+'/src/Models/'+entity.oclAsType(Classifier).name.toUpperFirst()+'.js'),
false, 'UTF-8')]
import mongoose from "mongoose";

[comment
    El archivo debe definir la estructura de la entidad para guardar posteriormente en la base de datos.

    Debe indicar todos los atributos y relaciones propias de la entidad y en caso de ser necesario de la
    entidad abstracta de la que hereda.
 /]
const [entity.oclAsType(Classifier).name/]Schema = new mongoose.Schema({

[comment Primero se crean los atributos y relaciones de la clase a la que hereda si la hay /]
[if ((entity.contained_relations -> selectByType(Inheritance) -> isEmpty()) <> true)]
    [for (inheritanceAttributes : Attribute | entity.contained_relations -> selectByType(Inheritance) -> first().destination.oclAsType(Class).contained_attributes)]
    [inheritanceAttributes.name/]: {
        type: [inheritanceAttributes.type.toString().toUpperFirst()/],
        required: [inheritanceAttributes.required.toString()/],
        unique: [inheritanceAttributes.isUnique.toString()/]
    },
    [/for]
    [if ((entity.contained_relations -> selectByType(Inheritance) -> first().destination.oclAsType(Class).contained_relations -> selectByType(DirectionalRelation) -> isEmpty()) <> true)]
        [for (relation : DirectionalRelation | entity.contained_relations -> selectByType(Inheritance) -> first().destination.oclAsType(Class).contained_relations ->
selectByType(DirectionalRelation))]
            [if (environment.oclAsType(Environment).contained_microservices->selectByType(BackendMicroservice)->select(m1 | m1 <> microservice)->select(m2 |
m2.oclAsType(Diagram).contained_entityPackages.contained_classifiers->includes(relation.oclAsType(DirectionalRelation).destination))->size()>0)]
    [relation.relationName/]: {
        type: String,
        required: [relation.required.toString()/],
        ref: "[relation.oclAsType(DirectionalRelation).destination.oclAsType(Class).name.toUpperFirst()/]"
    },
            [else]
    [relation.relationName/]: {
        type: mongoose.Schema.Types.ObjectId,
        required: [relation.required.toString()/],
    },
            [/if]
        [/for]
    [/if]
[/if]
```

Figura 109 - Código GenerateModels 1

```
[comment generacion de los atributos y relaciones propios de la entidad /]
[for (entityAttribute : Attribute | entity.contained_attributes)]
    [entityAttribute.name/]: {
        type: [entityAttribute.type.toString().toUpperFirst()/],
        required: [entityAttribute.required.toString()/],
        unique: [entityAttribute.required.toString()/]
    },
[/for]
[if ((entity.contained_relations -> selectByType(DirectionalRelation) -> isEmpty()) <> true)]
    [for (relation : DirectionalRelation | entity.contained_relations -> selectByType(DirectionalRelation))]
        [if (environment.oclAsType(Environment).contained_microservices->selectByType(BackendMicroservice)->select(m1 | m1 <> microservice)->select(m2 |
m2.oclAsType(Diagram).contained_entityPackages.contained_classifiers->selectByType(Class)->includes(relation.oclAsType(DirectionalRelation).destination))->size()>0)]
    [relation.relationName/]: {
        type: String,
        required: [relation.required.toString()/],
        ref: "[relation.oclAsType(DirectionalRelation).destination.oclAsType(Class).name.toUpperFirst()/]"
    },
        [else]
    [relation.relationName/]: {
        type: mongoose.Schema.Types.ObjectId ,
        required: [relation.required.toString()/],
    },
        [/if]
    [/for]
[/if]
});

export default mongoose.model("[entity.oclAsType(Classifier).name.toUpperFirst()/]", [entity.oclAsType(Classifier).name.toUpperFirst()/]Schema);

[/file]
[/template]
```

Figura 110 - Código GenerateModels 2

GenerateControllers

Se encarga de generar un archivo por entidad el cual contiene todos los métodos para crear, buscar, modificar y borrar entidades al comunicarse con el microservicio a la ruta correspondiente (Ej: “http://microservicio1/api/entities/cliente/getAll”). Debe crear las llamadas a las bases de datos correspondientes y en caso de que una entidad tenga relaciones externas debe agregar sus llamadas en el método correspondiente [Figura 10-12 - Código GenerateControllers 1-3].

```
[template public generateControllers(entity: Classifier, microservice: BackendMicroservice, environment: Provider)]

[comment
    Se genera el controlador en la ruta "./ambiente1/Microservices/microservicio1/src/controllers/ejemploController.js"
 /]
[file ((environment.oclAsType(Environment).name + '/Microservices/'+microservice.oclAsType(Microservice).name+'/src/controllers/'+entity.name+'Controller.js'), false, 'UTF-8')]
import [entity.name.toUpperFirst()/] from "../Models/[entity.name.toUpperFirst()/].js";

[comment Si alguna entidad requiere comunicacion a otro microservicio se importan los modulos necesarios. /]
[if (environment.oclAsType(Environment).contained_microservices->selectByType(BackendMicroservice)->select(m1 | m1 <> microservice)->select(m2 |
microservice.oclAsType(Diagram).contained_entityPackages.oclAsType(EntityPackage).contained_classifiers.oclAsType(Class).contained_relations.destination->any(d |
m2.oclAsType(Diagram).contained_entityPackages.contained_classifiers->includes(d))->size() > 0)->size() > 0)]
import axios from "axios";
import dotenv from "dotenv";
dotenv.config();
[/if]

[comment
    Para cada entidad se generan 5 metodos: getAll, getOneById, saveOne, updateOne, deleteOneById.
    Si se requiere traer informacion de sus relaciones tambien se agrega al metodo la logica para
    realizarlo.
 /]

export default {
  getAll: async (req, res, next) => {
    try {
      let list_[entity.name.toLowerFirst()/] = await [entity.name.toUpperFirst()/].find();
      res.status(200).json(list_[entity.name.toLowerFirst()/]);
    } catch (err) {
      console.log(err);
      res.status(500).json(err);
    }
  },
```

Figura 111 - Código GenerateControllers 1

```

  [comment
    Al buscar un elemento tambien se busca la informacion a las entidades relacionadas
    ya sea dentro del mismo microservicio o microservicios externos.
  /]
  getOneById: async (req, res, next) => {
    const _id = req.params.id;
    try {
      let obtained[entity.name.toUpperFirst()/] = await [entity.name.toUpperFirst()/].findById(_id)
    [for (microserviceRelation : Class | environment.oclAsType(Environment).contained_microservices->selectByType(BackendMicroservice)->select(m1 | m1 =
microservice).oclAsType(Diagram).contained_entityPackages.oclAsType(EntityPackage).contained_classifiers->selectByType(Class)->select(c | entity.oclAsType(Class).contained_relations ->
selectByType(DirectionalRelation).destination->includes(c))]
            .populate("[microserviceRelation.oclAsType(Classifier).name/]")
    [/for]
    [for (microserviceRelation : Class | environment.oclAsType(Environment).contained_microservices->selectByType(BackendMicroservice)->select(m1 | m1 =
microservice).oclAsType(Diagram).contained_entityPackages.oclAsType(EntityPackage).contained_classifiers->selectByType(Class)->select(c | entity.oclAsType(Class).contained_relations ->
selectByType(Inheritance).destination.contained_relations -> selectByType(DirectionalRelation).destination->includes(c))]
            .populate("[microserviceRelation.oclAsType(Classifier).name/]")
    [/for]
            .exec().lean();

    [for (relationToEntity : DirectionalRelation | entity.oclAsType(Class).contained_relations->select(r | not
microservice.oclAsType(Diagram).contained_entityPackages.contained_classifiers->includes(r.destination)).oclAsType(DirectionalRelation))]
            obtained[entity.name.toUpperFirst()/].[relationToEntity.relationName/] = await axios.get(
                process.env.[environment.oclAsType(Environment).contained_microservices->select(m | m.oclAsType(Diagram).contained_entityPackages.contained_classifiers->any(c |
c.oclAsType(Class) = relationToEntity.eGet('destination'))->size() > 0)->first().name.toUpper()/]_MICROSERVICE_URL + "/api/entity/[relationToEntity.eGet('destination').eGet('name')/]"+
[relationName/]
            )
            .then((responseData) => responseData.data)
            .catch((e) => res.send(e));

    [/for]
      res.status(200).json(obtained[entity.name.toUpperFirst()/]);
    } catch (err) {
      res.status(500).json(err);
    }
  },

```

Figura 112 - Código GenerateControllers 2

```
  saveOne: async (req, res, next) => {
    const req[entity.name.toUpperFirst()/] = req.body.[entity.name/];

    try {
      const new[entity.name.toUpperFirst()/] = await [entity.name.toUpperFirst()/].create(req[entity.name.toUpperFirst()/]);
      res.json(new[entity.name.toUpperFirst()/]);
    } catch (err) {
      res.status(500).json(err);
    }
  },

  updateOneById: async (req, res, next) => {
    const _id = req.params.id;
    const req[entity.name.toUpperFirst()/] = req.body.[entity.name/];

    try {
      let result = await [entity.name.toUpperFirst()/].updateOne({ _id }, req[entity.name.toUpperFirst()/]);
      res.json(result);
    } catch (err) {
      console.log(err);
      res.status(500).json(err);
    }
  },

  deleteOneById: async (req, res, next) => {
    const _id = req.params.id;
    try {
      let result = await [entity.name.toUpperFirst()/].deleteOne({ _id });
      res.json(result);
    } catch (err) {
      res.status(500).json(err);
    }
  },

};
[/file]
[/template]
```

Figura 113 - Código GenerateControllers 3

Documentación de la Funcionalidad

Las funcionalidades son documentadas en una plantilla la cual especifica la épica de la que proviene, su historia de usuario, los casos de prueba, etc [Tabla 1 - Tabla Documentación Generar Código NodeJS].

A continuación tenemos un ejemplo para la funcionalidad anteriormente mencionada.

Generación de Entidades en NodeJS - Express		
Epica	Módulo de Generación de Código Backend	
Caso de Uso	US-201 Generar código Node - Express	
Inicio Programación	15 de agosto, 2022	
Finalización Programación	29 de agosto, 2022	
Programador/es	Fernández, Valentín	
Casos de Prueba		
Generación de controlador vacío cuando no existe una	Fecha de Prueba	29 de octubre, 2022
	Tipo de Prueba	Generación Automática

capacidad de base de datos	Resultado	Fallo
	Usuario que lo probó	Fernández Valentín
Relaciones a Microservicio externo en Modelo de Entidad	Fecha de Prueba	29 de octubre, 2022
	Tipo de Prueba	Generación Automática
	Resultado	Éxito
	Usuario que lo probó	Fernández Valentín
Llamada a microservicio externo en controlador	Fecha de Prueba	29 de octubre, 2022
	Tipo de Prueba	Generación Automática
	Resultado	Éxito
	Usuario que lo probó	Fernández Valentín

Tabla 55 - Tabla Documentación Generar Código NodeJS

Planificación de capacitación

Objetivo

El objetivo principal del plan de capacitación es proporcionar los conocimientos apropiados para que los usuarios aprendan en el menor tiempo posible a utilizar de forma eficaz el sistema Accelerate para una pronta implementación en el entorno productivo de su proyecto u organización.

Objetivos específicos de la capacitación

Se listan a continuación los objetivos que se pretenden lograr con la capacitación:

- Implementar fácilmente el sistema en los procesos productivos de la organización.
- Brindar el conocimiento necesario sobre el sistema para comenzar a operarlo.
- Evitar errores futuros ocasionados por desconocimiento del sistema.
- Fomentar el uso y recomendación del sistema a nuevos usuarios.
- Resolver dudas planteadas por los usuarios respecto a alcances y casos de uso del sistema.

Destinatarios

Los destinatarios considerados para la capacitación son:

- Usuarios Finales: Estos serán arquitectos de software que lo utilizarán para crear aplicaciones a demanda.
- Administradores: Serán capacitados fundamentalmente en la instalación, reporting y puesta en marcha del sistema.

Condiciones previas de los participantes

Los usuarios a capacitar en el uso del sistema deben tener conocimientos mínimos sobre desarrollo de software, modelado UML y arquitecturas de microservicios con despliegue en un ambiente cloud. En el caso del administrador, también debe tener conocimientos previos en la herramienta Grafana.

Métodos de capacitación

Se crearán distintas secciones para profundizar en cada situación. Se distinguen los métodos de capacitación entre usuarios finales y administradores.

- Usuarios finales: La sección inicial para capacitar a los usuarios finales es “Getting Started” con pasos a seguir utilizando el software. Luego, una sección llamada “Elements” donde se describen cada uno de los "objetos" que puede crear el usuario en el sistema y una sección de “FAQs” (Frequently Asked Questions) donde se puede evacuar dudas comunes al utilizarlo. Finalmente, una sección de capacitación llamada “Working Methodology” donde se describe la forma de trabajo que se debe adaptar al utilizar la herramienta.
- Administradores: Primero, una sección para capacitar a los administradores donde se explican los pasos a seguir para la instalación del sistema, con recomendaciones para la puesta en marcha, y requisitos necesarios para su funcionamiento utilizando el manual de usuario. Luego, una sección llamada “Reporting” donde se explica los reportes que brinda la herramienta en base a las generaciones de código realizadas.

Temas

Usuarios Finales

Los usuarios finales del sistema Accelerate serán arquitectos de software que quieran obtener el boilerplate code de una aplicación de una manera rápida para acortar tiempo de desarrollo, por lo que esta capacitación será, por un lado concisa para que se pueda modelar y obtener el código en un corto periodo de tiempo, y por otro lado, con recursos adicionales para aquellos que necesiten resolver dudas o requieran un mayor nivel de entendimiento del sistema.

Por lo anteriormente mencionado se ha diseñado:

Sección "Getting Started"

En esta sección se detallan los pasos a seguir para comenzar a utilizar el sistema adentrándose en la configuración y modelado. Se dan indicaciones a seguir desde la configuración inicial hasta obtener el código generado para una aplicación de ejemplo. Se tiene en cuenta la explicación de cada sección configurable en el entorno y su alcance para la generación de código a medida. Dentro de esta sección se proveen los siguientes videos descriptivos para comenzar a utilizar la herramienta:

1. Uso de la herramienta de modelado: https://youtu.be/qNvLHrZda3A
2. Resultados de la generación automática: https://youtu.be/icHCFJjnhvE

Sección "Elements"

Dentro de esta sección se describen cada uno de los "objetos" que puede crear el usuario con la descripción detallada de sus atributos. Ejemplo: Objeto Microservice, permite configurar

su nombre, su versión, su descripción y el puerto que va a exponer en su formato API REST. Esta sección, surge como complemento a la sección de introducción “Getting Started” y en esta, se detalla el sistema con mayor profundidad.

Sección “FAQs”

En esta sección se han preparado preguntas frecuentes para la resolución de dudas de usuarios o para aquellos que quieran aprender más sobre el software.

- ¿Cómo crear un ambiente?
- ¿Cómo crear un microservicio?
- ¿Cómo configurar las capacidades del microservicio?
- ¿Cómo modelar las clases de un microservicio?
- ¿Cómo configurar las rutas de un microservicio?
- ¿Cuáles son los alcances del sistema?

Además, se contemplan dudas futuras de los usuarios para su incorporación a la sección.

Sección “Working Methodology”

En esta sección se describe la metodología de trabajo que se debe adaptar al utilizar la herramienta. Esto ocurre ya que el ciclo de vida de desarrollo de software se modifica al utilizar la herramienta. Luego de que el arquitecto de software modela el sistema requerido en la herramienta y genera el código, este se transfiere a los desarrolladores los cuales continúan el desarrollo pero enfocado en la lógica del negocio, ya que el código inicial y repetitivo lo dispone Accelerate.

Recursos

- Una computadora con Windows.
- Un analista funcional.

Actividades para plan capacitación de usuarios finales

Actividad	Tiempo (días)	Personal Encargado
Elaboración del plan de capacitación de usuarios finales.	2	Analista Funcional 3
Elaboración de la sección y videos de “Getting Started”.	1	Analista Funcional 3
Elaboración de la sección “Elements”	2	Analista Funcional 3
Elaboración de la sección “FAQs”.	1	Analista Funcional 3
Elaboración de la sección “Working Methodology”	1	Analista Funcional 3

Tabla 56 - Actividades para plan capacitación de usuarios finales

Administradores

Sección de instalación del sistema.

Dentro de la sección de capacitación a los administradores, se detallan cuáles son los requisitos para la instalación, los pasos para su descarga desde la web e instalación, hasta su puesta en marcha para comenzar a utilizarlo, utilizando el manual de usuario.

Sección "Reporting"

En la sección de Reporting se explican los reportes que brinda la herramienta, utilizando Grafana para su presentación gráfica, en base a las generaciones de código realizadas.

Recursos

- Una computadora con Windows.
- Un analista funcional.
- Un desarrollador frontend.
- Manual de usuario.
- Servidor host página web para descargar el sistema.
- Página web.
- Internet.

Actividades para plan capacitación de administradores

Actividad	Tiempo (días)	Personal Encargado
Elaboración del plan de capacitación para administradores.	2	Analista Funcional 3
Elaboración de la sección de instalación del sistema.	2	Analista Funcional 3
Elaboración de la sección "Reporting"	1	Analista Funcional 3
Elaboración página web y despliegue en internet.	2	Desarrollador Frontend

Tabla 57 - Actividades para plan capacitación de administradores

Diagrama de tiempos

Ver “Anexo N°1: Diagrama de tiempos”

Planificación, ejecución y documentación de pruebas

Objetivos generales

El objetivo principal es evaluar el funcionamiento del sistema mediante la aplicación de diferentes casos de prueba, verificando que se cumple con lo especificado en la etapa de Diseño y evitando futuros errores de la aplicación al momento de su despliegue.

Alcances

El análisis del sistema se realiza en base a cuatro tipos de prueba:

- **Pruebas de aceptación,** que verifican si el sistema cumple con los requisitos esperados y especificados en las historias de usuario.
- **Pruebas de generación automática,** que analizan si el sistema genera correctamente el código y la infraestructura en base a los modelos realizados por el usuario.
- **Pruebas de validación de ingreso de datos,** que evalúan si el sistema controla correctamente los datos ingresados por el usuario, para evitar errores al momento de la generación automática.
- **Pruebas de carga,** que comprueban si el sistema soporta la creación de modelos de gran tamaño.

Pruebas de aceptación

CP001 – Borrado de una capacidad de un microservicio	
Tipo de prueba	Prueba de aceptación
Objetivo	Comprobar que, dado que un usuario eligió una capacidad en el menú “Delete” del apartado “Manage capabilities” de un microservicio, la misma fue eliminada del modelo.
Usuario	Modelador
Módulos involucrados	Módulo de Ambientes y Microservicios

<table>
<tr><td>Precondición</td><td colspan="2">Debe existir un proyecto creado.
Debe existir un ambiente de Google.
Dentro de dicho ambiente, debe existir un microservicio.
El microservicio debe contener una capacidad.</td></tr>
<tr><td>Datos de prueba</td><td colspan="2">Capacidad utilizada = HealthCapability. Valores de la instancia: healthRoute = health, period = 5, initialDelay = 15.</td></tr>
<tr><td>Actor</td><td>Usuario</td><td>Sistema</td></tr>
<tr><td rowspan="4">Pasos</td><td></td><td>El sistema muestra un lienzo en blanco junto con una paleta de elementos que permite el modelado mediante el mecanismo “arrastrar y soltar”. Además, muestra la vista de microservicios junto con la pestaña “Properties” que permite configurar cada elemento individualmente</td></tr>
<tr><td>El usuario realiza doble clic sobre el ícono del microservicio existente.</td><td></td></tr>
<tr><td></td><td>El sistema muestra el apartado “Manage Capabilities”, que contiene únicamente la capacidad de salud existente. Además, muestra los botones “Add” y “Delete” para agregar y borrar capacidades, respectivamente.</td></tr>
<tr><td>El usuario selecciona el botón “Delete”.</td><td></td></tr>
</table>

		El sistema muestra una lista de capacidades, cada una con un botón de selección múltiple, que permite elegir las capacidades a ser borradas (en este caso, solamente muestra la capacidad de salud). Además, muestra los botones “Delete” y “Cancel”.
	El usuario selecciona el elemento HealthCapability y presiona el botón “Delete” para confirmar el borrado.	
Resultado esperado	Al seleccionar la capacidad deseada en el apartado “Delete” y confirmar con el botón ”Delete”, el sistema debe borrar la capacidad de salud del modelado.	
Resultado obtenido	El sistema no realiza ninguna acción.	
Acciones correctivas	Corregir el código de la herramienta gráfica para que borre la capacidad seleccionada en el apartado “Delete” del menú “Manage capabilities”.	

Tabla 58 - Borrado de una capacidad de un microservicio

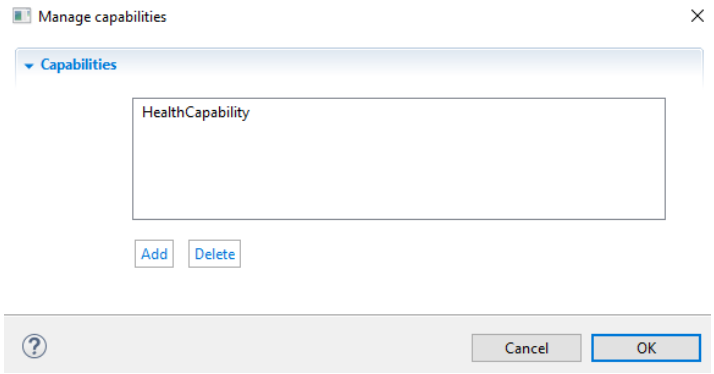


Figura 114. CP001 – Borrado de una capacidad de un microservicio. Menú que muestra las capacidades de un microservicio (Paso 1).

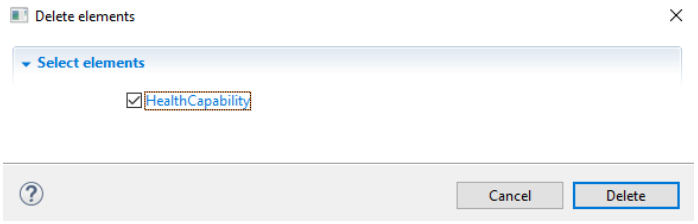


Figura 115. CP001 – Borrado de una capacidad de un microservicio. Menú que permite seleccionar las capacidades a borrar (Paso 2).

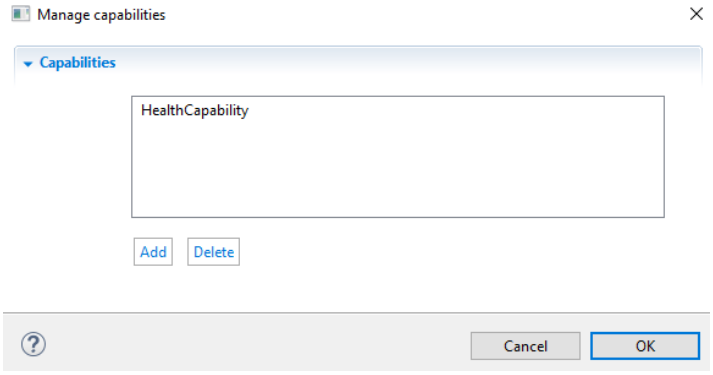


Figura 116. CP001 – Borrado de una capacidad de un microservicio. Resultado luego de seleccionar la capacidad de salud para su borrado (Paso 3).

CP002 – Creación de una relación externa		
Tipo de prueba	Prueba de aceptación	
Objetivo	Comprobar que un usuario puede crear una relación externa a través del elemento “ExternalDirectionalRelation”.	
Usuario	Modelador	
Módulos involucrados	Módulo de Entidades y Rutas	
Precondición	• Debe existir un proyecto creado. • Debe existir un ambiente de Google. • Dentro de dicho ambiente, deben existir dos microservicios. • Dentro de ambos microservicios debe existir una clase como mínimo.	
Datos de prueba	Nombre del ambiente de Google: AmbientePrueba Nombre del primer microservicio: Personas. Nombre del segundo microservicio: Facturación. Nombre de la primera clase: Persona. Nombre de la segunda clase: Factura.	
Actor	**Usuario**	**Sistema**

Pasos		El sistema muestra un lienzo en blanco junto con una paleta de elementos que permite el modelado mediante el mecanismo "arrastrar y soltar". Además, muestra la vista de microservicios junto con la pestaña "Properties" que permite configurar cada elemento individualmente.
	El usuario realiza doble clic sobre el ícono  del microservicio existente	
		El sistema cambia a la vista de entidades, donde se observa un el paquete "entities".
	El usuario arrastra el elemento "ExternalDirectionalRelation" desde la paleta a la clase Persona, para realizar la relación externa.	
		El sistema muestra un menú interactivo que permite la selección de la clase destino, navegando de la siguiente manera: 1. Selección del ambiente contenedor. 2. Selección del microservicio contenedor. 3. Selección de la clase destino.

		Además, se muestran los botones “OK”y “Cancel” para confirmar la operación o cancelarla, respectivamente.
	El usuario realiza la navegación con el menú interactivo de la siguiente manera: 1. Selección del ambiente “Ambiente Prueba”. 2. Selección del microservicio “Facturación”. 3. Selección de la clase “Factura”.	
Resultado esperado	Al seleccionar la clase Factura, se espera que se cree una relación unidireccional desde la clase Persona a la clase Factura. En el modelado, la relación se ve representada con una flecha que sale de la clase Persona y llega a la clase Factura.	
Resultado obtenido	El sistema crea la relación entre la clase Persona y la clase Factura y muestra la relación correctamente.	
Acciones correctivas	N/A	

Tabla 59 - Creación de una relación externa

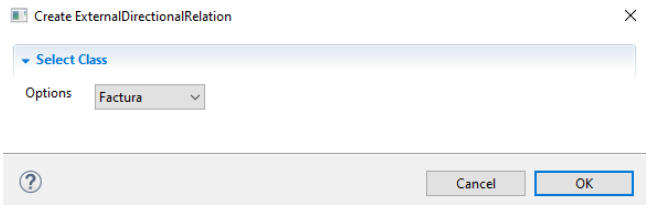


Figura 117. CP002 – Creación de una relación externa. Menú de navegación hacia la clase Factura.

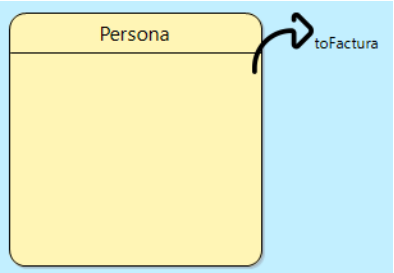


Figura 118. CP002 – Creación de una relación externa. Clase Persona (origen) al seleccionar el botón "OK".

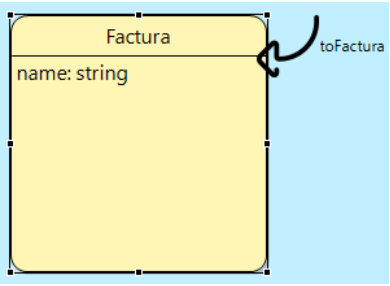


Figura 119. CP002 – Creación de una relación externa. Clase Factura (destino) al seleccionar el botón "OK".

CP003 – Creación de una relación de herencia		
Tipo de prueba	Prueba de aceptación	
Objetivo	Comprobar que al seleccionar una clase origen y luego una clase destino con el elemento "Inheritance", el sistema debe crear una relación de herencia entre la clase origen y la clase destino con una flecha sin relleno.	
Usuario	Modelador	
Módulos involucrados	Módulo de Entidades y Rutas	
Precondición	• Debe existir un proyecto creado. • Debe existir un ambiente de Google y contener un microservicio. • Dentro del microservicio deben existir dos clases.	
Datos de prueba	Nombre de la primera clase: Persona. Nombre de la segunda clase: Alumno..	
Actor	**Usuario**	**Sistema**
		El sistema muestra un lienzo en blanco junto con una paleta de elementos que permite el modelado mediante el mecanismo

Pasos		"arrastrar y soltar". Además, muestra la vista de microservicios junto con la pestaña "Properties" que permite configurar cada elemento individualmente.
	El usuario realiza doble clic sobre el ícono del microservicio existente	
		El sistema cambia a la vista de entidades, donde se observa un el paquete "entities".
	El usuario arrastra el elemento "Inheritance" desde la paleta y selecciona la clase Alumno y luego la clase Persona.	
Resultado esperado	Al seleccionar la clase Alumno y luego la clase Persona con el elemento "Inheritance", el sistema debe crear una relación de herencia entre la clase Alumno (origen) y la clase Persona (destino) con una flecha sin relleno.	
Resultado obtenido	El sistema no realiza ninguna acción.	
Acciones correctivas	Corregir el elemento "Inheritance", para que cree una relación de herencia entre las dos clases seleccionadas, comportándose como clase origen o destino dependiendo de la selección del usuario, siendo la clase origen la primera en ser seleccionada.	

Tabla 60 - Creación de una relación de herencia

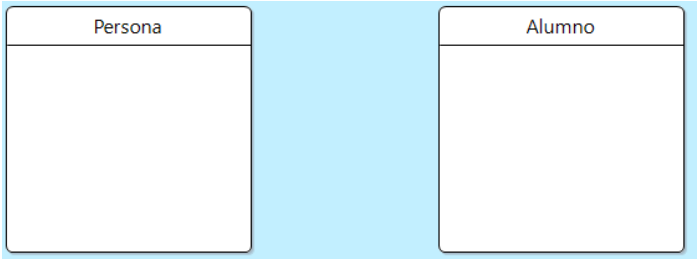


Figura 120. CP003 – Creación de una relación de herencia. Resultado obtenido.

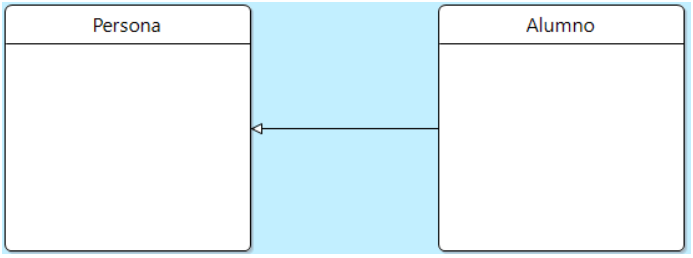


Figura 121. CP003 – Creación de una relación de herencia. Resultado esperado.

Pruebas de generación automática

CP004 – Generación de controlador vacío cuando no existe una capacidad de base de datos en el microservicio		
Tipo de prueba	Prueba de generación automática	
Objetivo	Comprobar que, dado que un microservicio no tiene una capacidad de base de datos configurada, los controladores generados no tengan implementación ABM.	
Usuario	Modelador	
Módulos involucrados	Módulo de Generación de Código Backend	
Precondición	• Debe existir un proyecto creado. • Debe existir un ambiente de Google. • Dentro de dicho ambiente, debe existir un microservicio. • El microservicio no debe contener una capacidad de base de datos configurada. • El microservicio debe contener una o más entidades con el atributo hasRoutes igual a verdadero.	
Datos de prueba		
Actor	**Usuario**	**Sistema**
		El sistema muestra un lienzo en blanco junto con una paleta de elementos que permite el modelado

Pasos		mediante el mecanismo "arrastrar y soltar". Además, muestra la vista de microservicios junto con la pestaña "Properties" que permite configurar cada elemento individualmente.
	El usuario selecciona el botón "Generate Code" y lo arrastra hacia el lienzo.	
		El sistema muestra un diálogo preguntándole al usuario si desea continuar. Para ello, se muestran dos botones "OK" y "Cancel", en el caso de que el usuario se arrepienta de la generación automática.
	El usuario presiona el botón "OK".	
		El sistema genera el código y la infraestructura a partir del modelo realizado.
Resultado esperado	Al generar automáticamente el código, se espera que los controladores de cada entidad (ubicados en la carpeta controllers del microservicio) no tengan implementación de ABM, ya que no existe una capacidad de base de datos configurada.	
Resultado obtenido	El sistema crea la carpeta controllers y cada uno de los controladores se crea con implementación de ABM, como si el microservicio tuviese una capacidad de MongoDB configurada.	
Acciones correctivas	Corregir el código de generación automática para que genere controladores sin implementación (es decir, únicamente con la cabecera de cada método) en el caso de que no tengan capacidad de base de datos configurada.	

Tabla 61 - Generación de controlador vacío cuando no existe una capacidad de base de datos en el microservicio

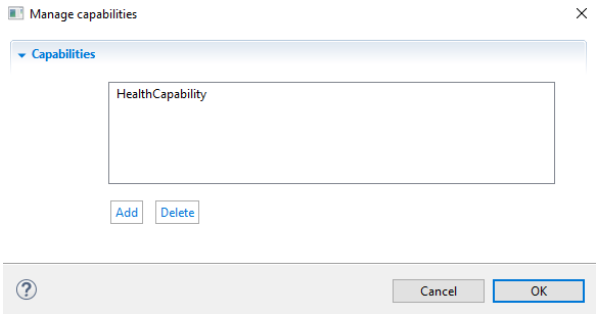


Figura 122. CP004 – Generación de controlador vacío cuando no existe una capacidad de base de datos en el microservicio. Capacidades del microservicio. Se puede observar que únicamente tiene configurada una capacidad de salud.

```
import Client from "../Models/Client.js";

import axios from "axios";


export default {    //Controladores con implementación ABM
  getAll: async (req, res, next) => {
    try {
      let list_client = await Client.find();
      res.status(200).json(list_client);
    } catch (err) {
      console.log(err);
      res.status(500).json(err);
    }
  },
```

Figura 123. CP004 – Generación de controlador vacío cuando no existe una capacidad de base de datos en el microservicio. Controlador generado automáticamente para la clase Client. Se puede observar que el mismo posee un controlador con implementación ABM.

CP005 – Relaciones a microservicio externo en modelo de entidad	
Tipo de prueba	Prueba de generación automática
Objetivo	Comprobar que, dado que una clase tiene una relación a otra que reside en un microservicio externo, se cree un atributo en su modelo que representa a dicha relación.

Usuario	Modelador	
Módulos involucrados	Módulo de Entidades y Rutas Módulo de Generación de Código Backend	
Precondición	• Debe existir un proyecto creado. • Debe existir un ambiente de Google. • Dentro de dicho ambiente, deben existir dos microservicios, uno llamado Books y otro llamado Clientes. • El microservicio Clientes debe tener una capacidad de MongoDB configurada. • Dentro del microservicio Book, debe existir una clase llamada BookInstance. • Dentro del microservicio Clientes, debe existir una clase llamada Client. • La clase Client debe tener una relación saliente con destino igual a BookInstance y nombre igual a rentedBook.	
Datos de prueba		
Actor	**Usuario**	**Sistema**
Pasos		El sistema muestra un lienzo en blanco junto con una paleta de elementos que permite el modelado mediante el mecanismo “arrastrar y soltar”. Además, muestra la vista de microservicios junto con la pestaña “Properties” que permite configurar cada elemento individualmente.
	El usuario selecciona el botón “Generate Code” y lo arrastra hacia el lienzo.	
		El sistema muestra un diálogo preguntándole al usuario si desea continuar. Para ello, se muestran dos botones “OK” y “Cancel”, en el caso de que el usuario se arrepienta de la generación automática.

	El usuario presiona el botón “OK”.	
		El sistema genera el código y la infraestructura a partir del modelo realizado.
Resultado esperado	Al generar automáticamente el código, se espera que exista un archivo “Client.js”, ubicado dentro de la carpeta models del microservicio Clientes. Dicho archivo, debe tener un atributo rentedBook de tipo String que represente la relación a la clase BookInstance del microservicio Books.	
Resultado obtenido	El sistema crea el archivo “Client.js” correctamente.	
Acciones correctivas	N/A	

Tabla 62 - Relaciones a microservicio externo en modelo de entidad

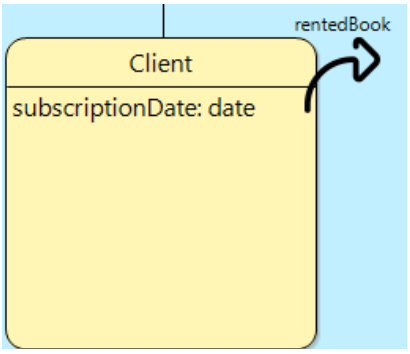


Figura 124. CP005 - Relaciones a microservicio externo en modelo de entidad. Clase Client con relación saliente.

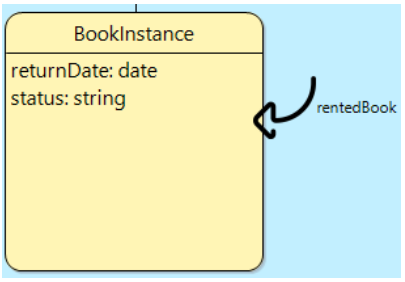


Figura 125. CP005 - Relaciones a microservicio externo en modelo de entidad. Clase BookInstance con relación saliente.

```
subscriptionDate: { //Atributo propio de la clase Client
    type: Date,
    required: true,
    unique: true
},

rentedBook: { //Atributo que referencia a la relación
    type: String,
    required: true,
    ref: "BookInstance"
},
```

Figura 126. CP005 - Relaciones a microservicio externo en modelo de entidad. Atributo que representa la relación en el archivo Client.js.

CP006 – Llamada a microservicio externo en controlador	
Tipo de prueba	Prueba de generación automática
Objetivo	Comprobar que, dado que una clase tiene una relación a otra que reside en un microservicio externo, se realice una llamada a este cuando se desee buscar una instancia de esta clase.
Usuario	Modelador
Módulos involucrados	Módulo de Entidades y Rutas Módulo de Generación de Código Backend
Precondición	• Debe existir un proyecto creado. • Debe existir un ambiente de Google. • Dentro de dicho ambiente, deben existir dos microservicios, uno llamado Books y otro llamado Clientes. • El microservicio Clientes debe tener una capacidad de MongoDB configurada. • Dentro del microservicio Book, debe existir una clase llamada BookInstance.

	• Dentro del microservicio Clientes, debe existir una clase llamada Client. • La clase Client debe tener una relación saliente con destino igual a BookInstance y nombre igual a rentedBook.	
Datos de prueba		
Actor	**Usuario**	**Sistema**
Pasos		El sistema muestra un lienzo en blanco junto con una paleta de elementos que permite el modelado mediante el mecanismo "arrastrar y soltar". Además, muestra la vista de microservicios junto con la pestaña "Properties" que permite configurar cada elemento individualmente.
	El usuario selecciona el botón "Generate Code" y lo arrastra hacia el lienzo.	
		El sistema muestra un diálogo preguntándole al usuario si desea continuar. Para ello, se muestran dos botones "OK" y "Cancel", en el caso de que el usuario se arrepienta de la generación automática.
	El usuario presiona el botón "OK".	
		El sistema genera el código y la infraestructura a partir del modelo realizado.

Resultado esperado	Al generar automáticamente el código, se espera que exista un archivo que represente al controlador de clientes denominado “ClientController.js”, ubicado dentro de la carpeta controllers del microservicio Clientes. Dicho archivo, debe tener la implementación del método getOneById, que permite buscar una entidad por su identificador. Esta función debe buscar la entidad de clientes en la base de datos y realizar una llamada al microservicio Books para buscar los libros asociados a la clase Client.
Resultado obtenido	El sistema crea el archivo “ClientController.js” correctamente.
Acciones correctivas	N/A

Tabla 63 - Llamada a microservicio externo en controlador

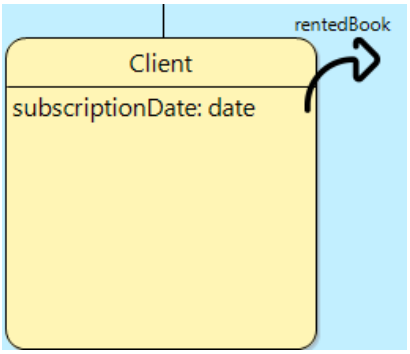


Figura 127. CP007 - Llamada a microservicio externo en controlador. Clase Client con relación saliente.

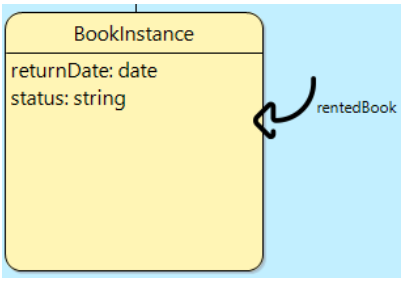


Figura 128. CP007 - Llamada a microservicio externo en controlador. Clase BookInstance con relación saliente.

```
getOneById: async (req, res, next) => {
  const _id = req.params.id;
  try {
    let obtainedClient = await Client.findById(_id) //Búsqueda de la instancia de cliente
    .populate("Address")
    .exec().lean();

    obtainedClient.rentedBook = await axios.get( //Llamada al microservicio Books
      process.env.BOOKS_MICROSERVICE_URL + "/api/entity/BookInstance"+rentedBook
    )
    .then((responseData) => responseData.data)
    .catch((e) => res.send(e));

    res.status(200).json(obtainedClient);
  } catch (err) {
    res.status(500).json(err);
  }
},
```

Figura 129. CP007 - Llamada a microservicio externo en controlador. Implementación del método getOneById con llamada al microservicio externo.

Pruebas de validación de ingreso de datos

CP007 – Ambiente de Google sin nombre		
Tipo de prueba	Prueba de validación de ingreso de datos.	
Objetivo	Corroborar que ningún ambiente de Google existente pueda no tener nombre.	
Usuario	Modelador	
Módulos involucrados	Módulo de Ambientes y Microservicios	
Precondición	• Debe existir un proyecto creado.	
Datos de prueba		
Actor	**Usuario**	**Sistema**
Pasos		El sistema muestra un lienzo en blanco junto con una paleta de elementos que permite el modelado mediante el mecanismo “arrastrar y soltar”. Además, muestra la vista de microservicios junto con la pestaña “Properties” que permite configurar cada elemento individualmente.

	El usuario crea un elemento Google Environment.	
	El usuario realiza clic derecho sobre el lienzo en blanco y selecciona la opción “Validate Diagram”.	
		El sistema valida los datos ingresados y muestra un mensaje de error.
Resultado esperado	Al crear un ambiente de Google sin nombre y validar el diagrama, el sistema debe mostrar el mensaje de error “Environment’s name cannot be null”.	
Resultado obtenido	El sistema muestra el mensaje mencionado previamente pero adicionalmente muestra el mensaje “Microservice’s name cannot be null”.	
Acciones correctivas	Reconfigurar la validación que verifica que el nombre de un microservicio no sea nulo, ya que está validando al elemento incorrecto.	

Tabla 64 - Ambiente de Google sin nombre

Figura 130. CP-007. Ambiente de Google sin nombre

CP008 – Dos clases con el mismo nombre dentro de un mismo microservicio	
Tipo de prueba	Prueba de validación de ingreso de datos.
Objetivo	Corroborar que no puedan existir dos clases con el mismo nombre dentro del mismo microservicio.
Usuario	Modelador
Módulos involucrados	Módulo de Entidades y Rutas
Precondición	• Debe existir un proyecto creado. • Dentro del proyecto, debe existir un GoogleEnvironment.

	• Dentro del elemento GoogleEnvironment, debe existir un BackendMicroservice. • El usuario debe haber navegado hacia el interior del elemento BackendMicroservice.	
Datos de prueba	Nombre de las clases = “Persona”	
Actor	**Usuario**	**Sistema**
Pasos		El sistema muestra un lienzo en blanco junto con una paleta de elementos que permite el modelado mediante el mecanismo “arrastrar y soltar”. Además, muestra la vista de microservicios junto con la pestaña “Properties” que permite configurar cada elemento individualmente.
	El usuario realiza doble clic sobre el ícono del microservicio existente.	
		El sistema cambia a la vista de entidades, donde se observa un el paquete “entities”.
	El usuario crea dos clases en el paquete “entities”, arrastrando el elemento Class hacia el lienzo.	

	El usuario realiza clic derecho sobre el lienzo en blanco y selecciona la opción “Validate Diagram”.	
		El sistema valida los datos ingresados y muestra un mensaje de error.
Resultado esperado	Al crear dos clases con el mismo nombre dentro de un microservicio, el sistema muestra el mensaje “Two or more classifiers with the same name”.	
Resultado obtenido	El sistema muestra el mensaje “Two or more classifiers with the same name”.	
Acciones correctivas	N/A	

Tabla 65 - Dos clases con el mismo nombre dentro de un mismo microservicio

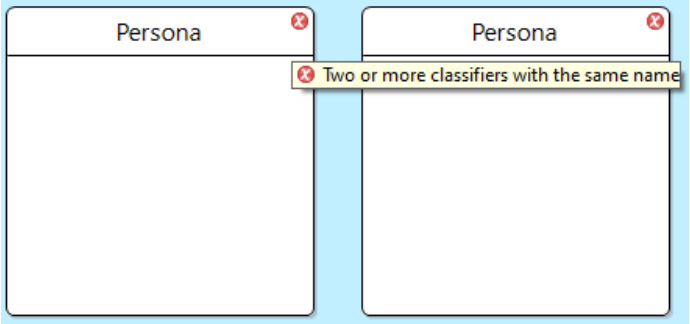


Figura 131. CP-008. Dos clases con mismo nombre dentro de un mismo microservicio

CP009 – Configuración de Kubernetes de un ambiente de Google con cantidad máxima de nodos menor a la cantidad mínima	
Tipo de prueba	Prueba de validación de ingreso de datos.
Objetivo	Corroborar que una configuración de Kubernetes de un ambiente de Google no pueda tener su cantidad máxima de nodos menor que la cantidad mínima especificada.
Usuario	Modelador
Módulos involucrados	Módulo de Ambientes y Microservicios
Precondición	• Debe existir un proyecto creado. • Dentro del proyecto, debe existir un GoogleEnvironment.
Datos de prueba	Cantidad máxima de nodos = 3 Cantidad mínima de nodos = 4

Actor	Usuario	Sistema
Pasos		El sistema muestra un lienzo en blanco junto con una paleta de elementos que permite el modelado mediante el mecanismo “arrastrar y soltar”. Además, muestra la vista de microservicios junto con la pestaña “Properties” que permite configurar cada elemento individualmente.
	El usuario realiza doble clic sobre el elemento GoogleEnvironment.	
		El sistema muestra un diálogo con las configuraciones del ambiente. El diálogo posee un botón “OK” para cerrarlo y aplicar los cambios.
	El usuario realiza doble clic sobre la configuración “GoogleKubernetesConfiguration”.	
		El sistema muestra un diálogo con la configuración de Kubernetes del ambiente. El diálogo posee un botón “OK” para cerrarlo y aplicar los cambios.

	El usuario ingresa la cantidad mínima de nodos en el atributo nodeMinCount y la cantidad máxima en el atributo nodeMaxCount. Luego, cierra los diálogos de configuración presionando los botones "OK" de cada uno de ellos.	
	El usuario realiza clic derecho sobre el lienzo en blanco y selecciona la opción "Validate Diagram".	
		El sistema valida los datos ingresados y muestra un mensaje de error.
Resultado esperado	Al crear una configuración de un ambiente de Google con una cantidad máxima de nodos menor que la cantidad mínima especificada, el sistema muestra el mensaje "GoogleKubernetesConfig's nodeMaxCount must be greater than nodeMinCount".	
Resultado obtenido	El sistema no muestra ningún mensaje de error.	
Acciones correctivas	Agregar una validación que muestre un mensaje de error cuando el atributo nodeMaxCount sea menor que nodeMinCount dentro de un elemento de configuración de Kubernetes de un GoogleEnvironment.	

Tabla 66 - Configuración de Kubernetes de un ambiente de Google con cantidad máxima de nodos menor a la cantidad mínima

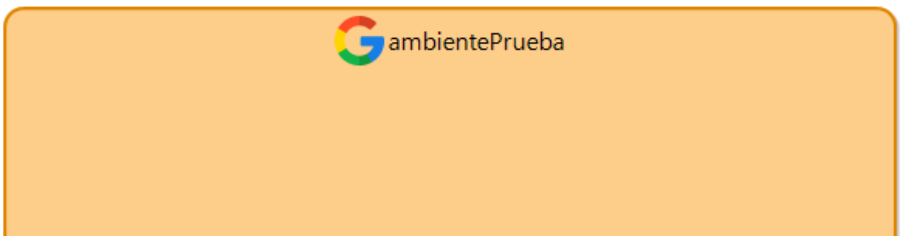


Figura 132. CP-009. Configuración de Kubernetes de un ambiente de Google con cantidad máxima de nodos menor a la cantidad mínima. El mensaje de error debería mostrarse sobre la esquina superior izquierda.

Pruebas de carga

CP010 – Ambientes de Google de gran tamaño		
Tipo de prueba	Prueba de carga.	
Objetivo	Corroborar que el sistema soporta la creación de un único ambiente de Google con una gran cantidad de microservicios. Generación automática del código para cada uno de ellos en un tiempo esperado no mayor a 25 segundos.	
Usuario	Modelador	
Módulos involucrados	Módulo de Ambientes y Microservicios Módulo de Generación de Infraestructura Módulo de Generación de Código Backend	
Precondición	• Debe existir un proyecto creado. • Dentro del proyecto, debe existir un GoogleEnvironment. • Dentro de dicho ambiente, deben existir 25 microservicios, cada uno de ellos con 4 entidades.	
Datos de prueba		
Actor	**Usuario**	**Sistema**
Pasos		El sistema muestra un lienzo en blanco junto con una paleta de elementos que permite el modelado mediante el mecanismo “arrastrar y soltar”. Además, muestra la vista de microservicios junto con la pestaña “Properties” que permite configurar cada elemento individualmente.
	El usuario selecciona el botón “Generate Code” y lo arrastra hacia el lienzo.	
		El sistema muestra un diálogo preguntándole al usuario si desea continuar. Para ello, se muestran dos botones “OK” y “Cancel”, en el caso de que el usuario se arrepienta de la generación automática.

	El usuario presiona el botón “OK”.	
		El sistema genera el código y la infraestructura a partir del modelo realizado.
Resultado esperado	Al crear un ambiente con una gran cantidad de microservicios, se espera que el sistema genere automáticamente el código para cada uno de ellos en un tiempo no mayor a 25 segundos.	
Resultado obtenido	El sistema genera el código y la infraestructura en 15 segundos.	
Acciones correctivas	N/A	

Tabla 67 - Ambientes de Google de gran tamaño

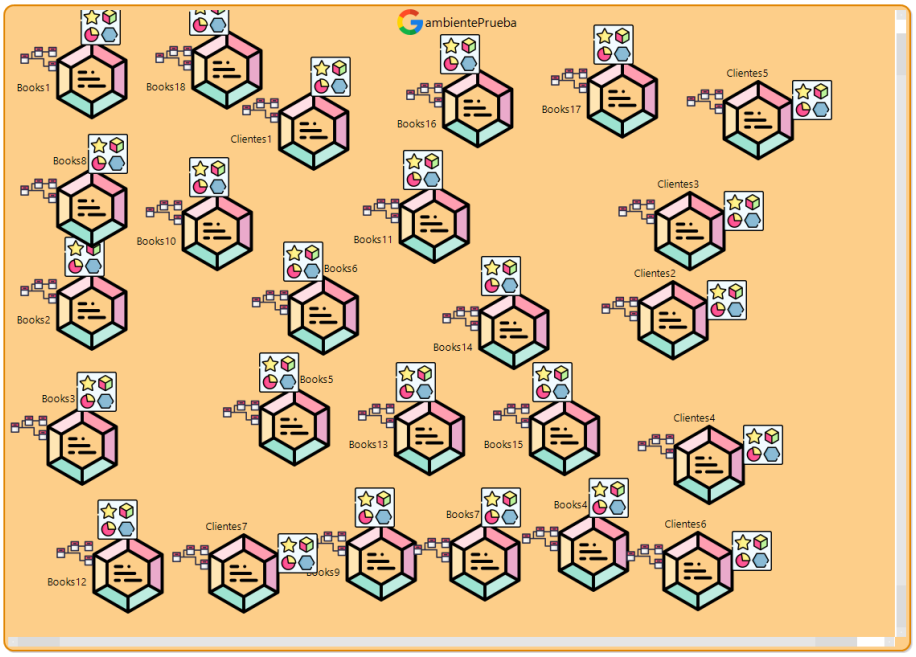


Figura 133. CP010. Ambientes de Google de gran tamaño. Vista de microservicios.

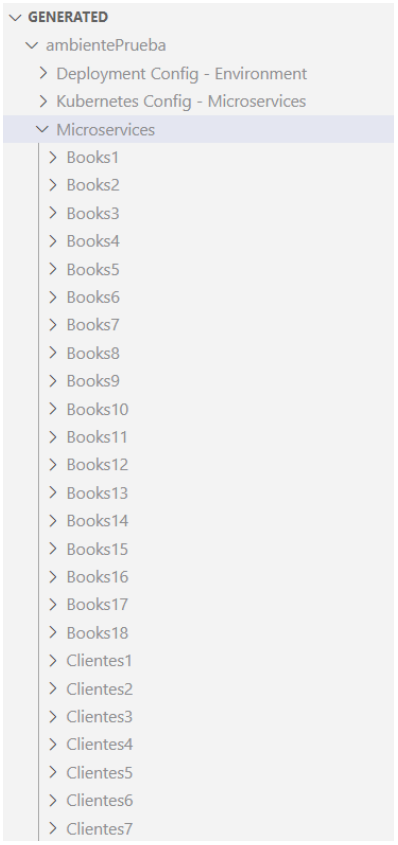


Figura 134. CP010. Ambientes de Google de gran tamaño. Resultado de la generación automática.

CP011 – Gran cantidad de ambientes de Google		
Tipo de prueba	Prueba de carga.	
Objetivo	Corroborar que el sistema soporta la creación de una gran cantidad de ambientes de Google. Generación automática en un tiempo no mayor a 15 segundos.	
Usuario	Modelador	
Módulos involucrados	Módulo de Ambientes y Microservicios Módulo de Generación de Infraestructura	
Precondición	• Debe existir un proyecto creado. • Dentro del proyecto, deben existir 15 ambientes de Google, cada uno de ellos con dos microservicios.	
Datos de prueba		
Actor	**Usuario**	**Sistema**
		El sistema muestra un lienzo en blanco junto con una paleta de elementos que permite el modelado

Pasos		mediante el mecanismo “arrastrar y soltar”. Además, muestra la vista de microservicios junto con la pestaña “Properties” que permite configurar cada elemento individualmente.
	El usuario selecciona el botón “Generate Code” y lo arrastra hacia el lienzo.	
		El sistema muestra un diálogo preguntándole al usuario si desea continuar. Para ello, se muestran dos botones “OK” y “Cancel”, en el caso de que el usuario se arrepienta de la generación automática.
	El usuario presiona el botón “OK”.	
		El sistema genera el código y la infraestructura a partir del modelo realizado.
Resultado esperado	Al crear una gran cantidad de ambientes, se espera que se genere automáticamente la infraestructura y el código para cada uno de ellos en un tiempo no mayor a 15 segundos.	
Resultado obtenido	El sistema genera el código y la infraestructura en 12 segundos.	
Acciones correctivas	N/A	

Tabla 68 - Gran cantidad de ambientes de Google

Figura 135. CP011 – Gran cantidad de ambientes de Google. Vista de microservicios

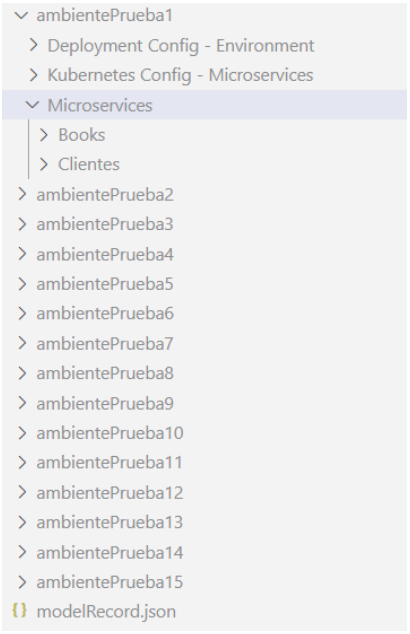


Figura 136. CP011 – Gran cantidad de ambientes de Google. Resultado de la generación automática.

CP012 – Gran cantidad de clases dentro de un microservicio		
Tipo de prueba	Prueba de carga.	
Objetivo	Corroborar que el sistema soporta la creación de una gran cantidad de clases dentro de un mismo microservicio. Generación automática en un tiempo de 15 segundos como máximo.	
Usuario	Modelador	
Módulos involucrados	Módulo de Entidades y Rutas Módulo de Generación de Código Backend	
Precondición	• Debe existir un proyecto creado. • Dentro del proyecto, debe existir un elemento GoogleEnvironment que contenga un microservicio. • Dicho microservicio, debe contener 100 clases.	
Datos de prueba		
Actor	**Usuario**	**Sistema**
Pasos		El sistema muestra un lienzo en blanco junto con una paleta de elementos que permite el modelado mediante el mecanismo “arrastrar y soltar”. Además, muestra la vista

		de microservicios junto con la pestaña “Properties” que permite configurar cada elemento individualmente.
	El usuario selecciona el botón “Generate Code” y lo arrastra hacia el lienzo.	
		El sistema muestra un diálogo preguntándole al usuario si desea continuar. Para ello, se muestran dos botones “OK” y “Cancel”, en el caso de que el usuario se arrepienta de la generación automática.
	El usuario presiona el botón “OK”.	
		El sistema genera el código y la infraestructura a partir del modelo realizado.
Resultado esperado	Al crear una gran cantidad de clases dentro del mismo microservicio, se espera que la generación automática tarde 15 segundos como máximo.	
Resultado obtenido	El sistema genera el código y la infraestructura en 5 segundos.	
Acciones correctivas	N/A	

Tabla 69 - Gran cantidad de clases dentro de un microservicio

Figura 137. CP012 – Gran cantidad de clases dentro de un microservicio. Vista de entidades.

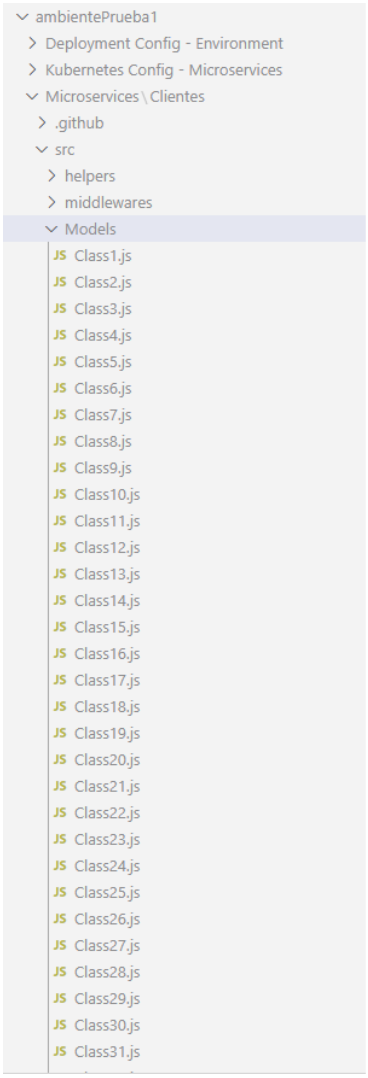


Figura 138. CP012 – Gran cantidad de clases dentro de un microservicio. Resultado de la generación automática.

La imagen muestra únicamente hasta la clase número 31, pero el resultado sigue hasta la clase número 100.

Manual de usuario del Sistema completo

Ver “Anexo N°3: Manual de usuario”

Planificación de implementación del Sistema

Objetivo

El objetivo principal del plan de implementación es colocar en producción la herramienta Accelerate y poder hacer uso de todas las funcionalidades descritas en la etapa de diseño.

Método de conversión

El método de conversión a utilizar es el de prueba piloto por etapas. El mismo, consiste en utilizar la herramienta Accelerate en un proyecto en particular (denominado “proyecto piloto”) y generar automáticamente los microservicios más pequeños y sencillos. Si al finalizar el proyecto, se observa que el resultado de utilizar Accelerate es bueno, el próximo proyecto utilizará la herramienta, pero no sólo para los microservicios más pequeños y sencillos, sino que también se usará para generar aquellos módulos grandes y complejos.

Recursos involucrados

Para llevar a cabo la implementación utilizando el método de conversión mencionado, es necesario contar con el apoyo de un equipo de trabajo, que será quien lleve a cabo el proyecto piloto y hará uso de la herramienta.

Además, el equipo de desarrollo de Accelerate estará disponible para dar soporte en caso de presentarse algún inconveniente y se encargará de la implementación del sistema.

Distribución de roles

Los recursos involucrados están segmentados de la siguiente manera:

- **Equipo de trabajo**: Equipo de trabajo que utilizará la herramienta en un proyecto piloto.
- **Líder de proyecto**: Líder del equipo de trabajo mencionado.
- **DevOps**: Miembro del equipo creador de Accelerate que se encargará del despliegue de la herramienta.

Actividades del plan de implementación

Para colocar Accelerate en producción, es necesario realizar las siguientes actividades:

- **Definición del equipo de trabajo**:
    - **Formación del equipo**: Se forma el equipo de trabajo que utilizará Accelerate.
    - **Distribución de roles**: Se distribuyen los roles dentro del equipo (desarrolladores, líder de proyecto, etc.)
    - **Distribución de tareas**: Se distribuyen las tareas para cada uno de los roles. Cabe aclarar que la mayoría de los desarrolladores codifica de manera convencional, mientras que el resto utiliza Accelerate como herramienta de generación. Además, la infraestructura para los módulos desarrollados manualmente es creada por el DevOps del equipo de trabajo.
    - **Capacitación**: Se capacita al equipo de trabajo en el uso de Accelerate como herramienta de modelado y generación automática. Además, el líder de proyecto se capacita en la herramienta de generación de reportes.
- **Implementación del módulo de reporting**:
    - **Creación de una base de datos en la nube**: Se crea una base de datos MongoDB en la nube utilizado Mongo Atlas. La creación se realiza mediante una interfaz

gráfica ingresando a la página https://www.mongodb.com/es e ingresando con la cuenta de Gmail de Accelerate. Luego, se procede a la creación de una base de datos seleccionando el botón *"Create Database"*.

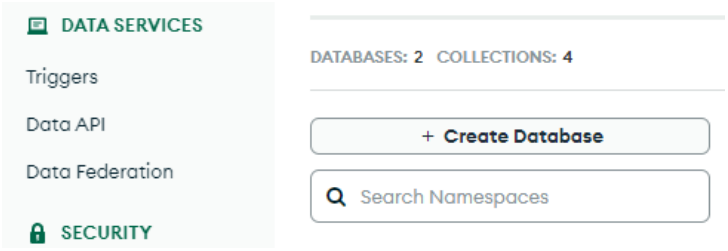


Figura 139. Creación de una base de datos.

Al hacerlo, se despliega un menú donde se puede ingresar el nombre de la base de datos y de la colección o tabla en cuestión.

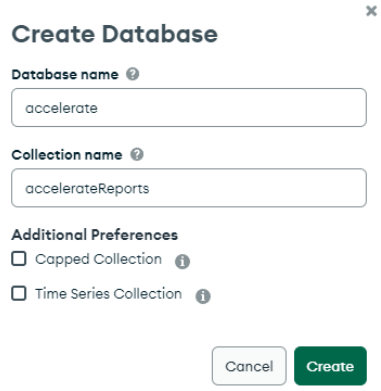


Figura 140. Creación de la base de datos de reporting.

Debido a que MongoDB es una base de datos no relacional, no es necesaria la creación de tablas, sino que el microservicio de reportes puede almacenar la información sin una estructura definida.

- **Despliegue del módulo de reporting en la nube**: Se despliega el módulo de reporting en la nube utilizando Heroku, enlazándolo con la base de datos creada previamente (es decir, enviándole la dirección de la base de datos al módulo de

reporting mediante una variable de entorno). Además, se despliega el visualizador del módulo en Grafana Cloud, permitiendo su acceso a través de Internet.

- **Implementación del módulo de modelado y generación automática**:
    - **Aseguramiento de requisitos mínimos**: Se aseguran los requisitos mínimos (sistema operativo Windows, por ejemplo) y se realizan correcciones, en caso de ser necesario, para cada uno de los puestos de trabajo que utilizará Accelerate.
    - **Instalación del módulo**: Se instala la herramienta de modelado y generación automática en cada puesto de trabajo, permitiendo su ejecución a través de un archivo ejecutable. Este paso se describe en la sección 2.1.1 “Instalación del sistema” del manual de usuarios de Accelerate.
    - **Creación del proyecto de modelado**: Se crea un proyecto de modelado que permitirá al equipo de trabajo realizar el diseño del sistema en el mismo.
    - **Backup del proyecto creado**: Se realiza el backup del proyecto de modelado, permitiendo su recuperación en caso de que se presente algún inconveniente. En caso de ser necesario, la recuperación puede aplicarse simplemente copiando el proyecto resguardado en la herramienta.
    - **Aplicación de generación automática al proyecto creado**: Se realiza la generación automática a partir del diseño del sistema modelado.
- **Integración de código e infraestructura**:
    - **Integración de los microservicios generados con aquellos desarrollados manualmente**: Se realiza la integración entre los microservicios generados por Accelerate y aquellos codificados de manera manual.

- **Integración de la infraestructura generada con la infraestructura creada por el DevOps**: Se realiza la integración entre la infraestructura generada por Accelerate y la creada de forma manual.

- **Evaluación de resultados**: El equipo de trabajo se reúne para evaluar la utilidad de la herramienta y el desempeño de esta en el flujo de trabajo, detectando qué otros servicios podrían haber sido generados por la herramienta. En esta actividad, el líder de proyecto accede al módulo de reporting para visualizar métricas que ayuden en la evaluación.
- **Planteamiento de la siguiente etapa**: El líder de proyecto define la aplicación de la herramienta en la siguiente etapa basándose en la evaluación de resultados. Si la evaluación da resultados positivos, Accelerate se utilizará para una mayor cantidad de microservicios en el siguiente proyecto. Este proceso se repite de manera iterativa hasta que se implemente de manera total en todos los proyectos.
- **Implementación de Accelerate en todos los proyectos**: Se incorpora Accelerate de manera total al flujo de trabajo, desarrollando todos los microservicios y la infraestructura a partir de la generación inicial que provee la herramienta.

Duración estimada

El plan de implementación tiene una duración estimada de 45 días, tomando como inicio la formación del equipo de trabajo y como fin la implementación de Accelerate en todos los proyectos.

Diagrama de tiempos

Ver “Anexo N° 2: Diagrama de tiempos de planificación de implementación”

Planificación de proyectos de sistemas

Sistema generador automático de código e infraestructura

Planificación de proyectos de sistemas

Capítulo I: Actividades

Definición y descripción de actividades

El proyecto está dividido en 4 etapas:

- **Etapa de planificación:** En esta etapa se definen las actividades a realizar y los tiempos que demora cada una de ellas. Además, se especifica el equipo de trabajo y se asigna cada tarea a los integrantes de este. Finalmente, se definen los métodos de comunicación formal, avances, retroalimentación y gestión de la configuración.
- **Etapa de definición de requerimientos**: En esta etapa se analizan herramientas similares a la del proyecto, mostrando pros y contras de cada una, la tecnología que utilizan, entre otras cosas. Esta información relevada será de utilidad para el diseño del sistema.
- **Etapa de diseño**: En esta etapa se definen todas las funcionalidades del proyecto, se diseñan los modelos de datos y los módulos a implementar.
- **Etapa de desarrollo e implementación**: En esta etapa se codifica lo especificado en la etapa de diseño y se crea un plan de capacitación a usuarios y un plan de implementación del sistema.

A continuación, se especifican las actividades de cada etapa junto con una breve descripción:

Planificación

- **Actividades:** Corresponde al Capítulo I de la etapa de planificación.
    - **Definición y descripción de actividades:** Definir las actividades a realizar a lo largo de todo el proyecto. Para cada actividad se especifica una descripción a modo de resumen. Además, se especifican las fechas límite de presentación o revisión.
    - **Creación del diagrama de tiempos:** Ubicar las actividades en el tiempo de forma que dicha combinación cumpla con las fechas límite de presentación o revisión.
- **Organización para la ejecución del proyecto:** Corresponde al Capítulo II de la etapa de planificación.
    - **Definición del equipo de trabajo:** Definir los puestos necesarios para llevar a cabo las tareas descriptas. Para cada puesto se especifica una descripción mínima, un perfil y sus responsabilidades.
    - **Descripción de las funciones principales de los miembros del equipo:** Distribuir los perfiles a cada miembro del equipo de trabajo de acuerdo a los roles que cada uno debe cumplir a lo largo del desarrollo del proyecto.
    - **Asignación de tareas a cada miembro del equipo:** Asignar cada tarea a los puestos de trabajo. Los puestos asignados no deben tener sobrecarga de tareas, es decir, no puede una persona estar trabajando en más de una tarea.
    - **Especificación de métodos de comunicación formal, control de avance y retroalimentación:** Definir la forma de comunicación del equipo, los

mecanismos de control de avance y el estilo de retroalimentación con el que se trabajará en el proyecto.

- **Especificación de la Gestión de Configuración del Software:** Especificar las herramientas y los métodos utilizados para la gestión de la configuración del software tanto para artefactos como para el código a desarrollar.

- **Estudio de factibilidad:** Corresponde al Capítulo III de la etapa de planificación.
  - **Realización del diagrama de recursos:** Realizar el diagrama de recursos del proyecto.
  - **Realización del análisis de factibilidad:** Realizar el análisis de factibilidad del proyecto.
  - **Definición de costos desagregados por recursos con periodicidad mensual:** Establecer un costo para cada recurso. Luego, calcular el costo mensual para cada uno de ellos según la cantidad de horas de trabajo.
  - **Realización del análisis de riesgos:** Realizar el análisis de riesgos del proyecto.
  - **Realización del análisis de impacto ambiental:** Realizar el análisis de impacto ambiental del proyecto.

Definición de requerimientos

- Relevamiento de herramienta Amplication
  - Relevamiento general de Amplication: Realizar el relevamiento general de la herramienta Amplication. En el mismo se debe describir la información, las funciones detectadas a nivel general y sus relaciones con otros sistemas y la tecnología que utiliza.

- Relevamiento detallado de Amplication: Realizar el relevamiento detallado de la herramienta Amplication. En el mismo se debe explicar con detalle todas las funciones descritas en el relevamiento general, realizar un modelo lógico del sistema y detectar problemas y necesidades en las funciones relevadas.

- Relevamiento de herramienta Vemto
  - Relevamiento general de Vemto: Realizar el relevamiento general de la herramienta Vemto. En el mismo se debe describir la información, las funciones detectadas a nivel general y sus relaciones con otros sistemas y la tecnología que utiliza.
  - Relevamiento detallado de Vemto: Realizar el relevamiento detallado de la herramienta Vemto. En el mismo se debe explicar con detalle todas las funciones descritas en el relevamiento general, realizar un modelo lógico del sistema y detectar problemas y necesidades en las funciones relevadas.
- Relevamiento de herramienta JHipster
  - Relevamiento general de JHipster: Realizar el relevamiento general de la herramienta JHipster. En el mismo se debe describir la información, las funciones detectadas a nivel general y sus relaciones con otros sistemas y la tecnología que utiliza.
  - Relevamiento detallado de JHipster: Realizar el relevamiento detallado de la herramienta JHipster. En el mismo se debe explicar con detalle todas las

funciones descritas en el relevamiento general, realizar un modelo lógico del sistema y detectar problemas y necesidades en las funciones relevadas.

- Relevamiento de herramienta Fogg
    - Relevamiento general de Fogg: Realizar el relevamiento general de la herramienta Fogg. En el mismo se debe describir la información, las funciones detectadas a nivel general y sus relaciones con otros sistemas y la tecnología que utiliza.
    - Relevamiento detallado de Fogg: Realizar el relevamiento detallado de la herramienta Fogg. En el mismo se debe explicar con detalle todas las funciones descritas en el relevamiento general, realizar un modelo lógico del sistema y detectar problemas y necesidades en las funciones relevadas.
- Relevamiento de herramienta DhiWise
    - Relevamiento general de DhiWise: Realizar el relevamiento general de la herramienta DhiWise. En el mismo se debe describir la información, las funciones detectadas a nivel general y sus relaciones con otros sistemas y la tecnología que utiliza.
    - Relevamiento detallado de DhiWise: Realizar el relevamiento detallado de la herramienta DhiWise. En el mismo se debe explicar con detalle todas las funciones descritas en el relevamiento general, realizar un modelo lógico del sistema y detectar problemas y necesidades en las funciones relevadas.
- **Definición de objetivos y alcances preliminares del sistema:** Definir los objetivos y los alcances del sistema a nivel preliminar. Estos objetivos y alcances serán profundizados más adelante.

Diseño

- **Definición del sistema y creación de historias de usuario**: Profundizar los objetivos y alcances del sistema preliminares descritos previamente y se crean de las historias de usuario.
- **Creación de ejemplos base para la generación automática de código en Node.js**: Crear ejemplos de ABM (Alta, Baja y Modificación) que cubren todas las necesidades descritas en las historias de usuario en Node.js. La salida de esta tarea es un proyecto de código que permita la ejecución en modo servidor API REST que reciba peticiones y responda de acuerdo con la funcionalidad ABM para modelos de ejemplo.
- **Investigación sobre mecanismos de autenticación y autorización:** Realizar la investigación sobre la posibilidad de implementar mecanismos de autenticación y autorización a determinadas rutas de un microservicio.
- **Creación de ejemplos base para la generación automática de infraestructura**: Crear ejemplos que cubren todas las necesidades descritas en las historias de usuario respecto a infraestructura. La salida de esta tarea es la siguiente:
    - Ejemplo de Dockerfile para creación de imagen Docker para un proyecto de ejemplo.
    - Ejemplo de manifiestos de Kubernetes para un ecosistema de microservicios de ejemplo.
    - Ejemplo de CI/CD genérica con GitHub Actions.

- Ejemplo de Terraform aplicado a GKE para un ecosistema de microservicios de ejemplo.

- **Creación de metamodelo para la generación de código:** Crear el metamodelo de base que utilizarán las plantillas de generación automática y la visualización de código.
- **Creación de metamodelo para la generación de microservicios e infraestructura:** Crear el metamodelo de base que utilizarán las plantillas de generación automática y la visualización de microservicios e infraestructura.
- **Definición de pantallas del sistema:** Definir las pantallas que utilizará el usuario para realizar modelos que servirán de entrada a los módulos de generación automática.

Desarrollo

- **Creación de plantilla de Acceleo para la generación de código en Node.js:** Crear la plantilla en Acceleo para la generación automática de código en Node.js.
- **Creación de plantilla de Acceleo para la generación de infraestructura:** Crear la plantilla en Acceleo para la generación automática de la infraestructura de microservicios.
- **Creación de ejemplos base para la generación automática de código en Python**: Crear ejemplos de ABM (Alta, Baja y Modificación) que cubren todas las necesidades descritas en las historias de usuario utilizando Python [18]. La salida de esta tarea es un proyecto de código que permita la ejecución en modo servidor API REST que reciba peticiones y responda de acuerdo con la funcionalidad ABM para modelos de ejemplo.
- **Creación de plantilla de Acceleo para la generación de código en Python:** Crear la plantilla en Acceleo para la generación automática de código en Python.

- **Creación del módulo de reporting:** Crear el módulo encargado de la visualización de métricas de uso de la herramienta.
- **Creación de visualización en Sirius para la generación de código:** Crear la visualización de las pantallas del sistema definidas en Sirius [7] con respecto a la generación de código.
- **Creación de visualización en Sirius para la generación de infraestructura:** Crear la visualización de las pantallas del sistema definidas en Sirius con respecto a la generación de infraestructura de microservicios.
- **Investigación e implementación de Sirius Web:** Investigar la creación de un servicio frontend basado en los metamodelos existentes que permita ejecutar la visualización de Sirius dentro de un contenedor Docker, es decir, por fuera del entorno de Eclipse. Si el resultado de la investigación es exitoso, se procede a migrar la visualización de Sirius a un entorno web dentro de un contenedor Docker que produzca un archivo XMI como salida. Dicho archivo XMI contendrá el modelo realizado por el usuario.
- **Investigación para la creación de un plugin de Eclipse:** Investigar la creación de un plugin de Eclipse que permite realizar una unión entre los metamodelos, los archivos de generación automática y la herramienta gráfica.
- **Creación de un plugin de Eclipse:** Crear un plugin de Eclipse a partir de la investigación realizada y de los metamodelos, plantillas y herramienta desarrollada, para poder ejecutar el sistema a través de un archivo ejecutable que permita su portabilidad.

- **Planificación, ejecución y documentación de pruebas de aceptación:** Ejecutar pruebas de aceptación al sistema y documentar su resultado, verificando que lo implementado corresponde con el diseño.
- **Planificación, ejecución y documentación de pruebas de validación de ingreso de datos:** Ejecutar pruebas de ingreso de datos al sistema y documentar su resultado, verificando que el usuario no pueda ingresar datos inconsistentes.
- **Planificación, ejecución y documentación de pruebas de carga:** Ejecutar pruebas de ingreso de datos al sistema y documentar su resultado, verificando que el sistema responda correctamente ante modelos de gran tamaño.
- **Planificación, ejecución y documentación de pruebas de generación automática:** Ejecutar pruebas de generación automática y documentar su resultado, verificando que el sistema genere correctamente los archivos de código y configuración respecto a los datos ingresados por el usuario.
- **Planificación de capacitación**: Crear un plan de capacitación a los usuarios del sistema.
- **Creación de manual de usuario**: Crear un manual de usuario del sistema que incluya normas y procedimientos que reflejen los cambios necesarios para la implementación del Sistema y su entorno de funcionamiento.
- **Planificación de implementación**: Definir un plan de implementación del sistema desarrollado.

Diagrama de tiempos

Ver ”Anexo N°1: Diagrama de tiempos”

Capítulo II: Organización para la ejecución del proyecto

El proyecto se lleva a cabo utilizando la metodología tradicional de proyectos, separando el proyecto en etapas secuenciales con fechas límite coincidentes con las fechas de entrega de avances a la cátedra. Sin embargo, se usan las historias de usuario como herramienta para la definición de requerimientos en lugar de especificaciones de casos de uso.

Equipo de trabajo

Para llevar a cabo las tareas mencionadas en el Capítulo I, es necesario contar con personal que cumpla con los siguientes roles y cantidades:

- **Coordinador**: Es el rol de Peña, Pablo.
- **Analista funcional**:
    - **Analista funcional 1**: Es el rol de Peña, Pablo.
    - **Analista funcional 2**: Es el rol de Fernández, Valentín.
    - **Analista funcional 3**: Es el rol de Rivero, Maximiliano.
- **Especialista en generación automática de código en Node.js**: Es el rol de Fernández, Valentín.
- **Especialista en generación automática de código en Python**: Es el rol de Rivero, Maximiliano.
- **Especialista en generación automática de infraestructura**: Es el rol de Peña, Pablo.
- **Arquitecto de software**:
    - **Arquitecto de software 1**: Es el rol de Peña, Pablo.
    - **Arquitecto de software 2**: Es el rol de Fernández, Valentín.

  - **Arquitecto de software 3**: Es el rol de Rivero, Maximiliano.

- **Desarrollador de plantillas de generación automática**:
  - **Desarrollador de plantillas de generación automática 1**: Es el rol de Rivero, Maximiliano.
  - **Desarrollador de plantillas de generación automática 2**: Es el rol de Fernández, Valentín.
- **Desarrollador Frontend**: Es el rol de Peña, Pablo.
- **Desarrollador de Business Intelligence**: Es el rol de Fernández, Valentín.

A continuación, mostramos una descripción de los puestos mencionados:

Coordinador	
Su rol es el de gerenciar el proyecto para lograr los resultados esperados en tiempo y forma mediante la aplicación de técnicas de liderazgo y resolución de conflictos. Además, debe brindar apoyo a los demás integrantes del equipo y gestionar los recursos a aplicar de manera eficiente	
Perfil	Responsabilidades
• Conocimiento de metodología tradicional de desarrollo de software. • Conocimiento de técnicas de liderazgo.	• Motivar y apoyar al equipo de trabajo. • Gestionar reuniones de equipo. • Planificar actividades y gestionar los recursos necesarios.

• Conocimiento de técnicas de resolución de conflictos. • Capacidad de trabajo en equipo y buen manejo de relaciones interpersonales • Proactividad. • Capacidad de organización y planificación. • Formación: Carrera de grado en Ingeniería en Sistemas de Información, Licenciatura en Sistemas o afín.	• Elaborar informes de avances del proyecto. • Realizar seguimiento de cada fase e hito del proyecto. • Asegurar la comunicación en el equipo de trabajo. • Resolver posibles conflictos del equipo de trabajo. • Crear un plan de capacitación de usuarios. • Crear un plan de implementación del proyecto.

Tabla 70 - Descripción Coordinador

Analista funcional	
Su función principal es la de realizar tareas de relevamiento y análisis de los sistemas informáticos.	
Perfil	Responsabilidades
• Conocimiento de metodología tradicional de desarrollo de software. • Capacidad analítica.	• Entender, establecer y formalizar los requerimientos del cliente. • Creación de historias de usuario. • Creación de manuales de usuario.

• Capacidad de trabajar en equipo con buen manejo de las relaciones interpersonales. • Capacidad de negociación, flexibilidad y proactividad. • Formación: Analista en Sistemas o afín.	• Asegurar la correspondencia entre las necesidades y expectativas del cliente y el alcance del proyecto.

Tabla 71 - Descripción Analista Funcional

Especialista en generación automática de código en Node.js	
Su función principal es la de crear ejemplos para la generación automática de código en Node.js.	
Perfil	Responsabilidades
• Conocimiento de MDE (Ingeniería dirigida por Modelos). • Conocimiento de Node.js • Conocimiento de buenas prácticas de desarrollo de software • Conocimiento de programación orientada a objetos. • Responsabilidad y compromiso.	• Creación de ejemplos para la generación automática de código en Node.js. • Asegurar que los ejemplos creados cubran las necesidades descriptas en las historias de usuario.

Tabla 72 - Descripción Especialista en generación automática de código en Node.js

Especialista en generación automática de código en Python	
Su función principal es la de crear ejemplos para la generación automática de código en Python.	
Perfil	Responsabilidades
• Conocimiento de MDE (Ingeniería dirigida por Modelos). • Conocimiento de Python • Conocimiento de buenas prácticas de desarrollo de software • Conocimiento de programación orientada a objetos. • Responsabilidad y compromiso.	• Creación de ejemplos para la generación automática de código en Python. • Asegurar que los ejemplos creados cubran las necesidades descriptas en las historias de usuario.

Tabla 73 - Descripción Especialista generación automática de código en Python

Especialista en generación automática de infraestructura	
Su función principal es la de crear ejemplos para la generación automática de infraestructura.	
Perfil	Responsabilidades
• Conocimiento de MDE (Ingeniería dirigida por Modelos). • Conocimiento de Docker.	• Creación de ejemplos para la generación automática de infraestructura.

• Conocimiento de GitHub Actions. • Conocimiento de Kubernetes. • Conocimiento de Terraform. • Conocimiento de GKE (Google Kubernetes Engine). • Conocimiento de EKS (Elastic Kubernetes Service) de AWS. • Conocimiento de AKS (Azure Kubernetes Service). • Conocimientos intermedios de despliegue de un clúster en la nube. • Responsabilidad y compromiso. • Conocimiento de arquitecturas de microservicios.	• Asegurar que los ejemplos creados cubran las necesidades descriptas en las historias de usuario.

Tabla 74 - Descripción Especialista en generación automática de infraestructura

Arquitecto de software	
Es el responsable tanto de la definición y el diseño de la arquitectura del proyecto como de los modelos de datos de cada microservicio y la interacción entre los mismos.	
Perfil	Responsabilidades

• Conocimientos en arquitecturas de microservicios REST con comunicación por interfaz JSON. • Conocimiento de metodología tradicional de desarrollo de software. • Conocimientos generales de programación con Express en Node.js • Conocimientos generales de programación con TypeScript en React.js • Conocimiento de patrones para el desarrollo de los modelos de datos. • Conocimiento general de funcionamiento de bases de datos NoSQL. • Buena comunicación. • Capacidad de trabajo en equipo. • Capacidad analítica y de resolución de problemas complejos. • Creatividad. • Formación: Carrera de grado en Ingeniería en Sistemas de	• Diseño y definición de la arquitectura de microservicios del proyecto. • Definición de la tecnología a utilizar. • Diseño del modelo de datos de cada microservicio. • Diseño de las rutas de acceso a cada microservicio. • Colaborar en la creación del plan de implementación del proyecto. • Definición de la interacción entre microservicios.

Información, Licenciatura en Sistemas o afín.	

Tabla 75 - Descripción Arquitecto de software

Desarrollador de plantillas de generación automática	
Se encarga del desarrollo de plantillas en Acceleo que generan archivos de código e infraestructura de manera automática.	
Perfil	Responsabilidades
• Conocimientos avanzados de Acceleo Query Language. • Conocimientos básicos de programación en Node.js y Python. • Conocimientos básicos de metamodelado con Eclipse. • Conocimiento de buenas prácticas de desarrollo de plantillas. • Capacidad de resolución de problemas. • Responsabilidad y compromiso. • Capacidad para trabajar en equipo.	• Desarrollar plantillas en Acceleo que generen código en Node.js de forma automática basándose en los ejemplos desarrollados por un especialista. • Desarrollar plantillas en Acceleo que generen código en Python de forma automática basándose en los ejemplos desarrollados por un especialista. • Desarrollar plantillas en Acceleo que generen los archivos de infraestructura de forma automática basándose en los ejemplos desarrollados por un especialista. • Realizar la conexión entre los metamodelos y las plantillas de generación automática.

Tabla 76 - Descripción Desarrollador de plantillas de generación automática

Desarrollador Frontend	
Se encarga del desarrollo de la interfaz gráfica de usuario de la herramienta.	
Perfil	Responsabilidades
• Conocimientos avanzados de Acceleo Query Language. • Conocimientos avanzados de Sirius. • Conocimientos básicos de metamodelado con Eclipse. • Conocimiento de buenas prácticas de desarrollo de visualizaciones en Sirius. • Capacidad de resolución de problemas. • Responsabilidad y compromiso. • Capacidad para trabajar en equipo.	• Desarrollar la interfaz gráfica para el desarrollo automático de código definida en las pantallas de usuario. • Desarrollar la interfaz gráfica para el desarrollo automático de infraestructura definida en las pantallas de usuario. • Asegurar la usabilidad del sistema. • Realizar la conexión entre los metamodelos y las visualizaciones.

Tabla 77 - Descripción Desarrollador Frontend

Desarrollador de Business Intelligence	
Se encarga del desarrollo del módulo de reporting, que permite la visualización de métricas de uso de la herramienta.	
Perfil	Responsabilidades

• Conocimientos avanzados de Grafana • Conocimientos avanzados de bases de datos en la nube. • Conocimientos básicos de HTTP. • Capacidad de resolución de problemas. • Capacidad analítica. • Responsabilidad y compromiso. • Capacidad para trabajar en equipo.	• Desarrollar la visualización de las métricas de uso utilizando Grafana. • Diseñar e implementar la base de datos en la nube que permite la recopilación de métricas. • Definir el mecanismo de recopilación de métricas.

Tabla 78 - Descripción Desarrollador de Business Intelligence

Funciones principales de los miembros del equipo de trabajo

Al analizar los puestos necesarios para el proyecto, se observa que se necesita personal para los puestos mencionados pero el equipo de trabajo es solamente de tres personas. Por lo tanto, la distribución de puestos dentro del equipo de trabajo es la siguiente:

	Fernández, Valentín	Rivero, Maximiliano	Peña, Pablo
Coordinador			X
Analista funcional	X	X	X
Especialista en generación automática de código en Node.js	X		
Especialista en generación automática de código en Python		X	

Especialista en generación automática de infraestructura			X
Arquitecto de software	X	X	X
Desarrollador de plantillas de generación automática	X	X	
Desarrollador Frontend			X
Desarrollador de Business Intelligence	X		

Tabla 79 - Distribución de puestos dentro del equipo de trabajo

Métodos de comunicación, control de avance y retroalimentación

WhatsApp Messenger

Figura 141 - WhatsApp Messenger

WhatsApp Messenger es una aplicación de mensajería instantánea que permite enviar y recibir mensajes mediante Internet, creación de grupos de múltiples usuarios, canales de difusión, entre otras funciones.

Forma de uso de la herramienta: La comunicación principal del equipo de trabajo es a través de un chat de grupo en WhatsApp. En dicho grupo, cada miembro tiene la posibilidad de

comunicar inconvenientes con la tarea que está desarrollando, solicitar ayuda a otro miembro del equipo, entre otros.

Además, el equipo de trabajo realiza una reunión semanal donde se discuten los siguientes temas:

- Avance de cada miembro en la tarea que está realizando y revisión general por parte del equipo.
- Posibles inconvenientes en el desarrollo de las tareas.
- Necesidad de ajuste de la planificación.
- Metas por cumplir que serán revisadas la siguiente semana.

Trello

Figura 142 - Trello

Trello es un software de administración de proyectos con interfaz web y con cliente para iOS y Android para organizar proyectos. Esta herramienta permite la creación de dashboards con listas para agrupar tarjetas. De esta forma, el usuario puede crear listas personalizadas y agrupar tarjetas por un criterio determinado.

Forma de uso de la herramienta: En Trello, se crea un espacio de trabajo con 5 listas que cada uno de los integrantes del equipo puede editar. Cada una de estas listas agrupan distintas tarjetas que detallan título, descripción y fecha con los propósitos que se detallan a continuación:

- **Lista Completar hoy**: Esta lista posee tarjetas con fecha límite del día actual. La descripción de la tarjeta marca una meta que debe cumplirse hoy.
- **Lista Por hacer:** Esta lista posee tarjetas ordenadas por su fecha límite. Dichas tarjetas representan metas que deben cumplirse antes de su fecha límite.
- **Lista Entregas:** Esta lista posee tarjetas con las fechas límite de entrega a la cátedra. Cada tarjeta representa un hito propuesto por la cátedra de Proyecto Final.
- **Lista Atrasadas:** Esta lista posee todas las tarjetas retrasadas, es decir, aquellas que no han sido terminadas antes de su fecha límite.
- **Lista Reuniones:** Esta lista posee tarjetas con cada una de las reuniones realizadas semanalmente con los temas tratados en la misma
- **Lista Recursos Extra:** Esta lista posee tarjetas que hacen referencia a recursos que puedan ser de utilidad para algún miembro del equipo. Un ejemplo de un recurso extra es información o un enlace que sirva para el desarrollo de alguna tarea.

Gestión de Configuración del Software

GitHub

Figura 143 - GitHub

GitHub es una plataforma de desarrollo colaborativo utilizada para el alojamiento de proyectos que utilicen el sistema de control de versiones Git. Para trabajar con GitHub, cada persona que desee tener acceso a un repositorio debe registrarse en la plataforma. Luego, el líder de proyectos o coordinador del equipo se encarga de la creación de un repositorio y entrega acceso con distintos permisos a los colaboradores. Lo ideal, es que existan múltiples ramas para el trabajo de múltiples colaboradores de forma tal que cada colaborador realice su trabajo en una rama y luego el coordinador del equipo se encargue de la unión de cada rama a una rama central.

Cuando una persona realiza un avance, sincroniza su repositorio local de Git con el repositorio de GitHub (en línea) y el mismo queda registrado en un historial visible para los colaboradores.

Forma de uso de la herramienta: Alojamiento de repositorios Git.

Git

Figura 144 - Git

Git es una herramienta de control de versiones de código libre que permite el trabajo con ramas de forma distribuida. De esta manera, quien tenga acceso a un repositorio de Git puede descargar una copia, modificarla, y luego sincronizarla con el repositorio central. Este manejo distribuido del control de versiones permite a distintos colaboradores avanzar en el mismo proyecto y es de gran utilidad para nuestro equipo de trabajo. Además, el trabajo con ramas permite la división del trabajo dentro del repositorio. La forma más usual de trabajar con Git es mediante la instalación en una computadora y su posterior interacción con su consola de comandos.

Forma de uso de la herramienta: Cada módulo en desarrollo (metamodelos, plantillas Acceleo, visualización en Sirius, por ejemplo) cuenta con un repositorio de GitHub al que los miembros del equipo tienen acceso mediante invitación. Dichos repositorios se organizan de acuerdo con las siguientes ramas:

- **Rama master:** La rama master tiene la última versión de producción. El código en esta rama es siempre estable.
- **Rama develop:** La rama develop es el espacio de trabajo del desarrollador y tiene código no necesariamente estable. Una vez que el desarrollador completa su trabajo, lo comunica al equipo y los cambios realizados son enviados a la rama master.

En el caso de que un desarrollador no pueda cumplir con su tarea y necesite ayuda de otro desarrollador, se crearán dos ramas a partir de la rama develop, una con el nombre de cada desarrollador conteniendo el trabajo de cada uno.

Google Drive

Figura 145 - Drive

Google Drive es un servicio de alojamiento de archivos propiedad de Google Inc. que permite el guardado de elementos en la nube mediante el uso de una cuenta de email de Google. Además, cada persona puede crear carpetas y compartirlas a otros usuarios con distintos permisos (permiso de lectura y escritura, por ejemplo) según sea necesario.

Este otorgamiento de permisos permite que múltiples personas alojen su trabajo en una misma carpeta y puedan observar el trabajo de los demás. Si una persona tiene acceso a un archivo, también tiene la posibilidad de ver un historial de cambios del archivo.

Otra gran utilidad de esta herramienta es la portabilidad. Google Drive puede accederse mediante aplicaciones móviles, aplicaciones de computadoras de escritorio e incluso desde un buscador web.

Forma de uso de la herramienta: Esta herramienta es utilizada para almacenar artefactos de diseño de software y otros elementos. Los artefactos correspondientes a historias de usuario o diagramas de clases son versionados a través de una carpeta compartida llamada Artefactos en Google Drive a la que todos los miembros del equipo tienen acceso. Existe una

carpeta para cada tipo de artefacto que contiene las versiones de este diferenciadas por un número incremental. A continuación, se muestra un ejemplo de la estructura de los directorios:

- **Historias de usuario** (carpeta general de artefacto):
  - **Versión 1 (**carpeta de versión)
  - **Versión 2 (**carpeta de versión)

Cada vez que un miembro del equipo realiza una modificación a la última versión de un artefacto debe realizar los siguientes pasos:

1. Crear una carpeta con el número de la última versión más uno.
2. Dentro de la carpeta creada, guardar el artefacto modificado.
3. Crear un archivo .txt donde se indiquen los principales cambios realizados.

Lo mismo sucede para el versionado de otros elementos tales como los ejemplos para la generación automática de código. Sin embargo, en este último caso los distintos tipos de elementos están agrupados en una carpeta llamada Otros.

Capítulo III: Factibilidad.

Diagrama de Recursos

Se presenta un diagrama de recursos donde se puede observar el nombre del recurso y nivel de asignación de cada uno.

Name	Max. Units	Standard Rate	Accrue At	Base Calendar
Coordinador	100%	$1300.00/hour	Prorated	Standard
Analista funcional 1	100%	$800.00/hour	Prorated	Standard
Analista funcional 2	100%	$800.00/hour	Prorated	Standard
Analista funcional 3	100%	$800.00/hour	Prorated	Standard
Especialista en generación automática de código en Node.js	100%	$950.00/hour	Prorated	Standard
Especialista en generación automática de código en Python	100%	$950.00/hour	Prorated	Standard
Especialista en generación automática de infraestructura	100%	$100.00/hour	Prorated	Standard
Arquitecto de software 1	100%	$1200.00/hour	Prorated	Standard
Arquitecto de software 2	100%	$1200.00/hour	Prorated	Standard
Arquitecto de software 3	100%	$1200.00/hour	Prorated	Standard
Desarrollador de plantillas de generación automática 1	100%	$970.00/hour	Prorated	Standard
Desarrollador de plantillas de generación automática 2	100%	$970.00/hour	Prorated	Standard
Desarrollador frontend	100%	$750.00/hour	Prorated	Standard
Desarrollador de Business Intelligence	100%	$950.00/hour	Prorated	Standard

Figura 146- Diagrama de Recursos extraído de Project Libre [14]

Nombre recurso	Tipo	Unidades Máx	Tasa Estándar	Devengado en	Calendario Base
Coordinador	Trabajo	100%	$1.300	Prorrateado	Standard
Analista Funcional 1	Trabajo	100%	$800	Prorrateado	Standard
Analista Funcional 2	Trabajo	100%	$800	Prorrateado	Standard
Analista Funcional 3	Trabajo	100%	$800	Prorrateado	Standard
Especialista en generación automática de código en Node.js	Trabajo	100%	$950	Prorrateado	Standard
Especialista en generación automática de código en Python	Trabajo	100%	$950	Prorrateado	Standard
Especialista en generación automática de infraestructura	Trabajo	100%	$1000	Prorrateado	Standard

Arquitecto de software 1	Trabajo	100%	$1200	Prorrateado	Standard
Arquitecto de software 2	Trabajo	100%	$1200	Prorrateado	Standard
Arquitecto de software 3	Trabajo	100%	$1200	Prorrateado	Standard
Desarrollador de plantillas de generación automática 1	Trabajo	100%	$970	Prorrateado	Standard
Desarrollador de plantillas de generación automática 2	Trabajo	100%	$970	Prorrateado	Standard
Desarrollador Frontend	Trabajo	100%	$750	Prorrateado	Standard
Desarrollador de Business Intelligence	Trabajo	100%	$950	Prorrateado	Standard

Tabla 80 - Diagrama de Recursos

Conclusión diagrama de recursos.

Como se puede observar, la distribución de tareas del proyecto se ha realizado de modo que no haya sobrecarga en los recursos. Esto nos permite obtener el mejor rendimiento de cada recurso a lo largo del proyecto.

Análisis de Factibilidad

Factibilidad Operativa

Tras el relevamiento realizado, se propone crear un sistema de aprovisionamiento automático en la nube mediante la ingeniería dirigida por modelos para la mejora de la calidad de software, llamado Accelerate.

Accelerate presentará una interfaz amigable, del estilo drag and drop, para el desarrollo de la arquitectura de microservicios con la posibilidad de configurar la forma de creación y configuración de los componentes que conforman la arquitectura, tales como los ambientes para cada proveedor y las capacidades de cada microservicio. Para cada uno de ellos, permite configurar su definición interna a través de la Programación Orientada a Objetos (POO) mediante un modelo UML y las rutas de acceso HTTP que se exponen con sus mecanismos de autenticación y autorización asociados.

El proyecto se lleva a cabo utilizando la metodología tradicional de proyectos, separando el proyecto en etapas secuenciales con fechas límite. Sin embargo, se usan las historias de usuario como herramienta para la definición de requerimientos en lugar de especificaciones de casos de uso. Todos los miembros conocen su rol en el equipo y el trabajo a realizar para hacerlo de la manera más eficiente durante el desarrollo del proyecto. El Coordinador es quien lleva a cabo la supervisión y control de los avances del proyecto, así como también la resolución de conflictos para cumplir con la calidad y plazos establecidos.

El personal necesario para el proyecto es: 1 Coordinador, 3 analistas funcionales, 1 Especialista en generación automática de código en Node.js, 1 Especialista en generación automática de código en Python, 1 Especialista en generación automática de infraestructura, 3 arquitectos de software, 2 Desarrollador de plantillas de generación automática y 1 desarrollador Frontend.

Se provee un plan de capacitación y manual de usuario del sistema que incluya normas y procedimientos que reflejen las acciones necesarias para la implementación del Sistema y su

entorno de funcionamiento, aunque el sistema está destinado a desarrolladores de software por lo que no se requiere una capacitación extensa para su uso.

El desarrollo de este sistema soluciona el problema que afronta el diseño e implementación de una arquitectura de microservicios con aprovisionamiento en la nube, la integración entre los distintos microservicios y la infraestructura subyacente, así como la heterogeneidad de las formas de diseño de una arquitectura. Como resultado, facilita a los miembros del equipo involucrados en el diseño e implementación del proyecto, realizar una correcta integración entre todos los elementos de la arquitectura.

Por otro lado, para la integración continua y el aprovisionamiento de la infraestructura en la nube, soluciona tareas repetitivas como la configuración inicial y optimiza su realización, además de solucionar la necesidad de personal con un alto grado de capacitación para configurar todo el entorno.

También, desde la perspectiva de los microservicios, optimiza la implementación de cada uno de ellos en un lenguaje de programación definido y su comunicación dentro de su entorno ya que es una tarea que consume valioso tiempo y es propensa a errores, sin mencionar la alta dependencia entre la aplicación de buenas prácticas de desarrollo y la experiencia del desarrollador.

Conclusión factibilidad operativa

Como conclusión, la cobertura por parte del sistema de estas necesidades, motiva y facilita la aceptación por parte de los usuarios ya que ahorra tiempo de implementación y configuración, soluciona la necesidad de contar con personal con alto grado de capacitación en

las tecnologías utilizadas, evita tareas repetitivas de desarrollo y configuración, y facilita la integración entre microservicios.

Factibilidad Técnica

Con la finalidad de resolver las necesidades detectadas en el relevamiento e implementar el sistema, se realiza un análisis de factibilidad técnica.

El sistema plantea automatizar la codificación y despliegue de la arquitectura de microservicios una vez ya haya sido modelada y configurada por el usuario dentro del sistema. Esto se realizará a través de plantillas de generación de código mediante la ingeniería dirigida por modelos, utilizando plantillas generadoras en Node.js y el paradigma “Infraestructura como código” (IaC), mediante archivos de configuración de Terraform.

La información que se ingresará manualmente al sistema por parte del usuario es: modelado y configuración de los componentes que conforman la arquitectura, tales como los ambientes para cada proveedor y las capacidades de cada microservicio. Para cada uno de ellos, se podrá configurar su definición interna mediante UML y las rutas de acceso HTTP, con sus mecanismos de autenticación y autorización asociados.

El uso del sistema será a través de una computadora personal, ingresando a la web para realizar el modelado y configuración de la aplicación a generar por Accelerate.

Como tecnologías utilizadas para el desarrollo del sistema tenemos a React para implementar la interfaz visual en una web, junto con la implementación de Sirius web para el modelado del esquema, Node.js para el backend, Docker y Kubernetes para el ecosistema de

microservicios, GitHub Actions para la implementación de CI/CD y Terraform para implementar Infraestructura como código. La seguridad del sistema estará implementada mediante Auth0 utilizando el protocolo de autorización OAuth 2.0, para el aseguramiento en las rutas en los microservicios.

Los metamodelos están planteados genéricamente con la flexibilidad de ampliar a nuevas tecnologías en el futuro, como pueden ser plantillas de generación automática en Python, ampliar los proveedores en la nube como Azure o AWS.

Tanto los metamodelos como las plantillas de generación automática utilizan Eclipse Modeling Framework como marco de trabajo, utilizando el metamodelo de guía para todo el desarrollo. Esta herramienta permite la instalación de plugins, permitiendo realizar transformaciones modelo a texto (M2T) mediante Acceleo y una representación gráfica con Sirius.

Aspecto	Necesidad	Análisis	Factible
Tipos de datos	Los tipos de datos ingresados por el usuario son de tipo texto y numérico.	Eclipse Modeling Framework soporta tipos de dato texto y numérico.	Sí

Nivel de automatización de las funciones.	Aumentar el grado de automatización para la implementación de nuevas aplicaciones de microservicios en la nube mediante el aprovisionamiento automático utilizando el paradigma IaC.	El sistema genera código para microservicios junto con su configuración de despliegue, basándose en la información ingresada por el usuario a través de la representación gráfica.	Sí
Crecimiento funcional y flexibilidad para nuevas TI	Escalamiento para necesidades de usuarios y nuevas tecnologías de información.	Los metamodelos están planteados genéricamente con la posibilidad de ampliar a nuevas tecnologías en el futuro.	Sí
Metodología de desarrollo	El equipo debe contar con conocimientos de metodología tradicional de desarrollo con historias de usuario.	Cada integrante del equipo conoce la metodología tradicional y el diseño del modelo funcional a través de historias de usuario.	Sí
Seguridad	Se necesita autenticación para usuarios y seguridad en las rutas.	La seguridad del sistema estará implementada mediante Auth0 utilizando el protocolo de autorización OAuth 2.0 para las rutas de acceso del código generado.	Sí

Integración con otros Sistemas y otras TI internas y externas.	Se requiere que el sistema se pueda integrar con otros sistemas y otras TI internas y externas.	El sistema se implementará utilizando el paradigma cliente-servidor, permitiendo la integración con otros sistemas y TI a través de una interfaz REST.	Sí
Testing	Se requiere la ejecución de pruebas de generación automática, de validación de ingreso de datos, de carga y de aceptación del sistema.	El sistema se prueba a través de la ejecución y documentación de pruebas de generación automática, de aceptación, de carga y de validación de ingreso de datos.	Sí

Tabla 81 - Factibilidad Técnica

Conclusión factibilidad técnica

De acuerdo con el análisis realizado, el sistema es factible de implementar utilizando Eclipse Modeling Framework con sus correspondientes plugins (Acceleo y Sirius) y permitiendo la conexión con el usuario y otras tecnologías mediante el paradigma cliente-servidor basado en una interfaz REST. Esto se logra mediante personal capacitado para cada actividad de implementación en el proyecto.

Factibilidad Legal

Se establecerá un contrato de términos y condiciones de uso en la plataforma del sistema detallando los aspectos legales que incumben la actividad del usuario en la misma una vez implementada, y que este último debe aceptar previo a su acceso al sistema. Este debe estar redactado con alto nivel de detalle para que no resulte ambiguo o confuso.

El desarrollo del sistema se plantea utilizando tecnologías open source aplicando la licencia GNU General Public License v3 y, por lo tanto, se plantea que el usuario debe aceptar los términos y condiciones de uso.

Desarrollar el sistema mediante tecnologías open source permite a los usuarios usar el código fuente, estudiarlo, modificarlo y distribuirlo ya sea mediante una copia exacta del programa o en versiones modificadas.

Condiciones de uso y permisos:

- El sistema puede ser distribuido.
- El sistema puede ser modificado.
- El sistema puede ser utilizado para uso comercial.
- El sistema puede ser modificado y utilizado para uso personal.
- Se debe publicar el código fuente del sistema modificado al ser distribuido.
- Las modificaciones que se realicen deben publicarse con la licencia actual heredada.
- Las modificaciones que se realicen al sistema deben documentarse.
- Esta explícitamente establecido que no se proveen garantías.

- La limitación de la responsabilidad de la licencia establece que en ningún caso podrá, un tercero que modifique o distribuya el sistema, hacer a usted responsable de daño alguno, siendo este daño genera, especial, ocasional o derivado que surja por el uso del sistema.

Conclusión factibilidad legal

Como resultado del análisis de factibilidad legal realizado, se concluye que el proyecto es legalmente factible ya que cumple con los requisitos que permiten su desarrollo y distribución mediante las licencias mencionadas anteriormente, así como también solicitando al usuario que acepte los términos y condiciones de uso del sistema.

Factibilidad Económica

Los costos a tener en cuenta para el desarrollo del proyecto se detallan a continuación:

- Google Cloud Platform: El nivel gratuito de GKE proporciona $74.40 en créditos mensuales por cuenta de facturación. Luego de esto, la tarifa por administración de clústeres de $0.10 por clúster por hora. Para el alcance del desarrollo del proyecto se utilizará el crédito gratuito.
- Auth0: Sin costo hasta 7.000 usuarios activos e inicios de sesión ilimitados. Luego los costos van desde 23 usd a 228 usd por mes. Durante el desarrollo del proyecto y hasta que se superen los 7.000 usuarios, se utilizará la versión sin costo de este servicio.

Es necesario considerar los costos para el equipo de desarrollo según el trabajo realizado y los equipos informáticos utilizados en el desarrollo, como son computadoras, internet y la oficina para el equipo de trabajo, como se puede ver en la sección “Costos desagregados por recurso con periodicidad mensual”.

Además, las tecnologías a utilizar durante el desarrollo son gratuitas, por lo que no se perciben costos en este factor.

Presupuesto Económico									
Costos	Mar	Abr	May	Jun	Jul	Ago	Sep	Oct	Nov
Computadora de Desarrollo 1	$168.000/6								
Computadora de Desarrollo 2	$168.000/6								
Computadora de Desarrollo 3	$168.000/6								
Oficina de Trabajo	$23.000/1	$23.000/1	$23.000/1	$23.000/1	$23.000/1	$23.000/1	$23.000/1	$23.000/1	$23.000/1
Internet	$2.000/1	$2.000/1	$2.000/1	$2.000/1	$2.000/1	$2.000/1	$2.000/1	$2.000/1	$2.000/1
Recursos Humanos	$84.000/1	$508.800/1	$654.400/1	$585.120/1	$564.160/1	$639.680/1	$531.120/1	$414.960/1	$88.000/1

Total Proyecto	$ 4.799.240

Tabla 82 - Presupuesto Económico

Conclusión factibilidad económica

Tras detallar los costos de tecnología, recursos e infraestructura necesarios para el desarrollo del proyecto, se determina que el proyecto es factible económicamente ya que no se presentan conflictos para abordarlos.

Factibilidad Financiera

Presupuesto Financiero									
Egresos	Mar	Abr	May	Jun	Jul	Ago	Sep	Oct	Nov
Computadora de Desarrollo 1	$28.000/6	$28.000/6	$28.000/6	$28.000/6	$28.000/6	$28.000/6			
Computadora de Desarrollo 2	$28.000/6	$28.000/6	$28.000/6	$28.000/6	$28.000/6	$28.000/6			
Computadora de Desarrollo 3	$28.000/6	$28.000/6	$28.000/6	$28.000/6	$28.000/6	$28.000/6			

Oficina de Trabajo	$23.000/1	$23.000/1	$23.000/1	$23.000/1	$23.000/1	$23.000/1	$23.000/1	$23.000/1	$23.000/1
Internet	$2.000/1	$2.000/1	$2.000/1	$2.000/1	$2.000/1	$2.000/1	$2.000/1	$2.000/1	$2.000/1
Recursos Humanos	$84.000/1	$508.800/1	$654.400/1	$585.120/1	$564.160/1	$639.680/1	$531.120/1	$414.960/1	$88.000/1
Total Mensual	$ 193.000	$ 617.800	$ 763.400	$ 694.120	$ 673.160	$ 748.680	$ 556.120	$ 439.960	$ 113.000

Tabla 83 - Presupuesto Financiero

Conclusión Factibilidad Financiera

Tras detallar los egresos que tendrá el proyecto mes a mes, se calcularon los egresos mensuales y se puede observar que hay una inversión inicial de computadoras de desarrollo que se financia a 6 meses, recursos de infraestructura contemplados en todo el desarrollo del proyecto y recursos humanos que los egresos varían dependiendo del personal necesario para cada tarea en cada etapa del proyecto. Se concluye que el proyecto es factible al tener contemplados los egresos monetarios necesarios.

Conclusión del Análisis de Factibilidad

Habiendo contemplado un análisis detallado del desarrollo proyecto en los aspectos operativo, técnico, legal, económico y financiero, y determinar la factibilidad de este en cada caso, se concluye que el proyecto es factible en su totalidad.

Costos desagregados por recursos con periodicidad mensual

Marzo			
Recursos Humanos			
Nombre recurso	Horas	Precio por hora	Subtotal
Coordinador	24	$1.100,00	$26.400
Analista Funcional 1	24	$800	$19.200
Analista Funcional 2	24	$800	$19.200
Analista Funcional 3	24	$800	$19.200
Total Recursos Humanos			$84.000
Recursos de infraestructura			
Nombre recurso	Cantidad	Precio	Subtotal
Computadora de desarrollo (Cuota 1)	3	$28.000,00	$84.000,00
Oficina de trabajo	1	$23.000,00	$23.000,00
Internet	1	$2.000,00	$2.000,00
Total Recursos de infraestructura			$109.000,00
TOTAL			$193.000

Tabla 84 - Costos Marzo

Abril			
Recursos Humanos			
Nombre recurso	Horas	Precio por hora	Subtotal
Coordinador	96	$1.100,00	$105.600
Analista Funcional 1	136	$800	$108.800
Analista Funcional 2	120	$800	$96.000
Analista Funcional 3	152	$800	$121.600
Arquitecto de software 1	32	$1.200	$38.400
Arquitecto de software 2	32	$1.200	$38.400
Total Recursos Humanos			$508.800
Recursos de infraestructura			
Nombre recurso	Cantidad	Precio	Subtotal
Computadora de desarrollo (Cuota 2)	3	$28.000,00	$84.000,00
Oficina de trabajo	1	$23.000,00	$23.000,00
Internet	1	$2.000,00	$2.000,00

Total Recursos de infraestructura	$109.000,00
TOTAL	$617.800

Tabla 85 - Costos Abril

Mayo			
Recursos Humanos			
Nombre recurso	Horas	Precio por hora	Subtotal
Coordinador	128	$1.100,00	$140.800
Analista Funcional 3	168	$800	$134.400
Especialista en generación automática de infraestructura	24	$1.000	$24.000
Arquitecto de software 1	152	$1.200	$182.400
Arquitecto de software 2	136	$1.200	$163.200
Arquitecto de software 3	8	$1.200	$9.600
Total Recursos Humanos			$654.400
Recursos de infraestructura			
Nombre recurso	Cantidad	Precio	Subtotal

Computadora de desarrollo (Cuota 3)	3	$28.000,00	$84.000,00
Oficina de trabajo	1	$23.000,00	$23.000,00
Internet	1	$2.000,00	$2.000,00
Total Recursos de infraestructura			$109.000,00
TOTAL			$763.400

Tabla 86 - Costos Mayo

Junio			
Recursos Humanos			
Nombre recurso	Horas	Precio por hora	Subtotal
Coordinador	96	$1.100,00	$105.600
Analista Funcional 3	104	$800	$83.200
Especialista en generación automática de código en Node.js	8	$950	$7.600
Especialista en generación automática de código en Python	8	$950	$7.600

Especialista en generación automática de infraestructura	24	$1.000	$24.000
Arquitecto de software 1	56	$1.200	$67.200
Arquitecto de software 2	32	$1.200	$38.400
Arquitecto de software 3	72	$1.200	$86.400
Desarrollador de plantillas de generación automática 2	96	$970	$93.120
Desarrollador frontend	96	$750	$72.000
Total Recursos Humanos			$585.120
Recursos de infraestructura			
Nombre recurso	Cantidad	Precio	Subtotal
Computadora de desarrollo (Cuota 4)	3	$28.000,00	$84.000,00
Oficina de trabajo	1	$23.000,00	$23.000,00
Internet	1	$2.000,00	$2.000,00
Total Recursos de infraestructura			$109.000,00
TOTAL			$694.120

Tabla 87 - Costos Junio

Julio			
Recursos Humanos			
Nombre recurso	Horas	Precio por hora	Subtotal
Coordinador	128	$1.100,00	$140.800
Analista Funcional 3	168	$800	$134.400
Desarrollador de plantillas de generación automática 2	168	$970	$162.960
Desarrollador frontend	168	$750	$126.000
Total Recursos Humanos			$564.160
Recursos de infraestructura			
Nombre recurso	Cantidad	Precio	Subtotal
Computadora de desarrollo (Cuota 5)	3	$28.000,00	$84.000,00
Oficina de trabajo	1	$23.000,00	$23.000,00
Internet	1	$2.000,00	$2.000,00
Total Recursos de infraestructura			$109.000,00

TOTAL	$673.160

Tabla 88 - Costos Julio

Agosto			
Recursos Humanos			
Nombre recurso	Horas	Precio por hora	Subtotal
Coordinador	96	$1.100,00	$105.600
Analista Funcional 3	184	$800	$147.200
Especialista en generación automática de código en Python	80	$950	$76.000
Arquitecto de software 2	160	$1.200	$192.000
Desarrollador de plantillas de generación automática 1	80	$970	$77.600
Desarrollador de plantillas de generación automática 2	24	$970	$23.280
Desarrollador frontend	24	$750	$18.000
Total Recursos Humanos			$639.680
Recursos de infraestructura			
Nombre recurso	Cantidad	Precio	Subtotal

Computadora de desarrollo (Cuota 6)	3	$28.000,00	$84.000,00
Oficina de trabajo	1	$23.000,00	$23.000,00
Internet	1	$2.000,00	$2.000,00
Total Recursos de infraestructura			$109.000,00
TOTAL			$748.680

Tabla 89 - Costos Agosto

Septiembre			
Recursos Humanos			
Nombre recurso	Horas	Precio por hora	Subtotal
Coordinador	104	$1.100,00	$114.400
Analista Funcional 3	176	$800	$140.800
Arquitecto de software 2	120	$1.200	$144.000
Desarrollador de plantillas de generación automática 1	80	$970	$77.600
Desarrollador de plantillas de generación automática 2	56	$970	$54.320

Desarrollador de Business Intelligence	40	$950	$38.000
Total Recursos Humanos			$569.120
Recursos de infraestructura			
Nombre recurso	Cantidad	Precio	Subtotal
Oficina de trabajo	1	$23.000,00	$23.000,00
Internet	1	$2.000,00	$2.000,00
Total Recursos de infraestructura			$25.000,00
TOTAL			$594.120

Tabla 90 - Costos Septiembre

Octubre			
Recursos Humanos			
Nombre recurso	Horas	Precio por hora	Subtotal
Coordinador	136	$1.100,00	$149.600
Analista Funcional 3	128	$800	$102.400
Desarrollador de plantillas de generación automática 2	168	$970	$162.960

Total Recursos Humanos			$414.960
Recursos de infraestructura			
Nombre recurso	Cantidad	Precio	Subtotal
Oficina de trabajo	1	$23.000,00	$23.000,00
Internet	1	$2.000,00	$2.000,00
Total Recursos de infraestructura			$25.000,00
TOTAL			$439.960

Tabla 91 - Costos Octubre

Noviembre			
Recursos Humanos			
Nombre recurso	Horas	Precio por hora	Subtotal
Coordinador	80	$1.100,00	$88.000
Total Recursos Humanos			$88.000
Recursos de infraestructura			
Nombre recurso	Cantidad	Precio	Subtotal
Oficina de trabajo	1	$23.000,00	$23.000,00
Internet	1	$2.000,00	$2.000,00

Total Recursos de infraestructura	$25.000,00
TOTAL	$113.000

Tabla 92 - Costos Noviembre

Costo Total Proyecto			
Mes	Recursos Humanos	Recursos de Infraestructura	Total
Marzo	$84.000	$109.000	$193.000
Abril	$508.800	$109.000	$617.800
Mayo	$654.400	$109.000	$763.400
Junio	$585.120	$109.000	$694.120
Julio	$564.160	$109.000	$673.160
Agosto	$639.680	$109.000	$748.680
Septiembre	$531.120	$25.000	$594.120
Octubre	$414.960	$25.000	$439.960
Noviembre	$88.000	$25.000	$113.000
TOTAL	$4.070.240	$729.000	$4.837.240

Tabla 93 - Costo total del proyecto

Análisis de Riesgos

A continuación, se detallará y realizará un análisis de los riesgos que pueden aparecer en el proyecto, así como también una evaluación de su impacto y las medidas preventivas a aplicar en cada caso.

Riesgos que pueden aparecer en el proyecto:

Constantes cambios y alcance del proyecto en continuo crecimiento: Como consecuencia de este riesgo puede suceder que el alcance quede poco claro en los integrantes del equipo y retrasos y cambios en la ejecución del proyecto.

Falta de experiencia con el dominio del negocio: Al tener poca experiencia en el dominio del negocio, se puede recaer en un mal análisis y definición de los requerimientos del sistema.

Falta de comunicación en el equipo: En caso de que ocurra falta de comunicación en el equipo las consecuencias pueden ser baja productividad, escaso control de avances e incumplimiento en la planificación y objetivos poco claros en el desarrollo.

Falta de conocimiento técnico de las tecnologías a utilizar: Como consecuencia de esto se necesitará ajustar la planificación para la inclusión de actividades de investigación, en caso de que excedan el plazo ya establecido y pueden ocurrir retrasos en los tiempos del proyecto.

Estimación inadecuada de tiempos de ejecución de las actividades: Al estimar en forma errónea los tiempos puede haber aumento de costos y demoras en el proyecto.

Estimación de costos inadecuada: Si se realiza una mala estimación de costos habría una alta posibilidad de no culminar el proyecto, incumplimiento en pagos e imposibilidad de adquirir recursos necesarios para el proyecto.

Extravío de documentación del proyecto: Como consecuencia, este riesgo ocasionaría retraso en la ejecución de actividades y aumento de costos del proyecto.

Fallos en el diseño del sistema: Al diseñar incorrectamente el sistema se tendría un aumento de errores en etapas posteriores, habría que rediseñar en etapas tardías del proyecto, aumento de costos y retraso del proyecto.

Cambios o bajas en el equipo de trabajo: Las turbulencias en el desarrollo del proyecto pueden ocasionar bajas de personal y esto repercutirá ocasionando retrasos en la planificación del proyecto, baja motivación del equipo, cambios en la planificación y asignación de recursos.

Fallos en hardware y conectividad: Imprevistos como fallas de hardware y conectividad podrían ocasionar tiempo perdido durante etapas de trabajo, pérdida de información y cambios en la asignación de recursos.

Fallas en el versionado de documentación y código: El versionado es crucial en el desarrollo del proyecto, por esto que al haber fallas podría ocurrir en pérdida de avances e inconsistencias en etapas de desarrollo y por lo tanto perjudicar en gran medida.

Evaluación de Riesgos

Para la evaluación de riesgos se establece la probabilidad de ocurrencia y el nivel de impacto para cada riesgo. La evaluación está realizada mediante el método directo.

Escala de probabilidad de ocurrencia e impacto:

Impacto	
Alto	5
Medio	3
Bajo	1

Tabla 94 - Escala de impacto

Probabilidad de ocurrencia	
Alta (71% - 100%)	5
Media (41% - 70%)	3
Baja (0% - 40%)	1

Tabla 95 - Escala probabilidad de ocurrencia

Tabla de evaluación de riesgos

Descripción	Probabilidad de ocurrencia	Impacto	Riesgo
Constantes cambios y alcance del proyecto en continuo crecimiento.	3	5	15
Estimación inadecuada de tiempos de ejecución de las actividades.	3	5	15
Estimación de costos inadecuada.	3	5	15
Fallos en el diseño del sistema.	3	5	15

Falta de experiencia con el dominio del negocio.	3	5	15
Falta de conocimiento técnico de las tecnologías a utilizar.	3	3	9
Falta de comunicación en el equipo.	1	5	5
Cambios o bajas en el equipo de trabajo.	1	5	5
Extravío de documentación del proyecto.	1	3	3
Fallas en el versionado de documentación y código.	1	5	5
Fallos en hardware y conectividad.	3	1	3

Tabla 96 - Evaluación de riesgos

Escala Categorización de los riesgos

Clasificación de los riesgos	
Riesgo Bajo	1 a 5
Riesgo Medio	6 a 10
Riesgo Alto	11 a 15

Tabla 97 – Escala de categorización de los riesgos

Categorización de los riesgos

Descripción	Riesgo	Categoría

Constantes cambios y alcance del proyecto en continuo crecimiento.	15	Alto
Estimación inadecuada de tiempos de ejecución de las actividades.	15	Alto
Estimación de costos inadecuada.	15	Alto
Fallos en el diseño del sistema.	15	Alto
Falta de experiencia con el dominio del negocio.	15	Alto
Falta de conocimiento técnico de las tecnologías a utilizar.	9	Medio
Falta de comunicación en el equipo.	5	Bajo
Cambios o bajas en el equipo de trabajo.	5	Bajo
Extravío de documentación del proyecto.	3	Bajo
Fallas en el versionado de documentación y código.	5	Bajo
Fallos en hardware y conectividad.	3	Bajo

Tabla 98 - Categorización de los riesgos

Conclusión del análisis de riesgo y medidas preventivas.

Tras detallar y evaluar los riesgos que pueden aparecer en el proyecto, se establecen medidas preventivas para los categorizados como Medio o Alto, con el fin de reducir el impacto o la probabilidad de ocurrencia de estos. No se establecen medidas preventivas para los categorizados como Bajos ya que no representan una gran amenaza y no suponen grandes esfuerzos a la hora de mitigarlos. Además, para la ubicación de las actividades en el tiempo se tienen en cuenta las medidas correctivas de los riesgos, estableciendo tiempos de holgura implícitos en la planificación que permiten su aplicación.

Riesgo	Medida preventiva	Medidas correctivas
Constantes cambios y alcance del proyecto en continuo crecimiento	Número 1: Reunión mensual para seguimiento de cambios.	- Reasignación de recursos para abarcar nuevos alcances.
Estimación inadecuada de tiempos de ejecución de las actividades.	Número 2: Revisión de tiempos con un experto.	- Reasignación de tiempos para actividades con estimaciones erróneas.
Estimación de costos inadecuada.	Número 3: Revisión de recursos a utilizar en el proyecto.	- Reasignación de recursos para minimizar costos. - Solicitar préstamo en caso de no poder afrontar nuevos costos.

	Número 4: Revisión de costos con un experto	
Fallos en el diseño del sistema.	Número 5: Revisión del uso patrones de diseño, seguimiento de normas y estándares que garanticen la calidad de esta etapa.	- Revisión y cambios en la aplicación patrones de diseño y estándares con expertos. - Realizar historias de usuario que abarquen problemas de diseño.
Falta de experiencia con el dominio del negocio.	Número 6: Revisión del relevamiento del negocio, incluyendo sistemas similares que se encuentren en funcionamiento.	- Relevamiento exhaustivo y detenido del dominio del negocio.
Falta de conocimiento técnico de las tecnologías a utilizar.	Número 7: Considerar periodos de investigación necesarios para tecnologías desconocidas.	- Prolongar períodos de investigación y consultar con personal capacitado.

Tabla 99 - Medidas preventivas de los riesgos

Análisis de Impacto Ambiental

Para realizar el análisis del impacto ambiental del proyecto, se destaca en la siguiente tabla cada uno de los ítems destacables en este aspecto, su explicación, y si el impacto es positivo o negativa.

Impacto	Tipo de Impacto	Detalles
Fomentar el uso de	Positivo para el	Ya que el código generado por Accelerate busca

tecnologías en la nube	medio ambiente	ser implementado en plataformas cloud, se fomenta el uso de las mismas por las industrias. Esto es beneficioso ya que estas han demostrado generar una huella de carbono mucho menor que la producida al montar los servicios on premise.
Reducción de recursos para la codificación inicial del proyecto	Positivo para el medio ambiente y para la sociedad	Al reducir los tiempos de desarrollo y la cantidad de equipos que se necesitan para la programación inicial del proyecto se logra ahorrar energía y almacenamiento en los dispositivos que se utilizan para el desarrollo al no necesitar tantas computadoras escribiendo y almacenando código, así como también reducción de smog y uso de combustible para transporte del personal encargado de esta etapa.
Creación de nuevas oportunidades laborales y de investigación	Positivo para la sociedad	Al ser una herramienta nueva y que permite un flujo de trabajo totalmente distinto para los equipos de desarrollo permite la creación de oportunidades laborales para aquellos profesionales que dominan la herramienta.

Facilita la inserción de personas no relacionadas en el mundo del desarrollo de software	Positivo para la sociedad	Al acelerar procesos de configuración inicial y evitar programación repetitiva, facilita la inserción de personas en el mundo del desarrollo de software al fomentar el desarrollo de más software en menos tiempo.
Uso de energía eléctrica para el mantenimiento de los servidores	Negativo en el medio Ambiente	Debido a que Accelerate se ejecuta en un servidor on premise, se genera un consumo de energía continuo para mantener el servidor y su refrigeración, ya que este debe estar activo continuamente incluso cuando la actividad en el mismo es baja.
Fomenta el desarrollo de tecnologías open source	Positivo para la sociedad	El desarrollo del sistema se realiza utilizando tecnologías open source por lo que estimula el crecimiento de la comunidad, el desarrollo de tecnología y nuevas soluciones en industria IT para la sociedad.

Tabla 100 - Aspectos de impacto ambiental

Para la evaluación del impacto ambiental se presenta una matriz con dos componentes principales: Por un lado las acciones que generan impacto, colocadas en las columnas de la matriz, y por el otro, las variables o componentes ambientales que representan el tipo de impacto, colocadas en las filas, seleccionadas en el estudio como aquellas más representativas del ambiente del proyecto.

	Fomentar el uso de tecnologías en la nube	Reducción de recursos para la codificación inicial del proyecto	Creación de nuevas oportunidades laborales y de investigación	Facilita la inserción de personas no relacionadas en el mundo del desarrollo de software	Uso de energía eléctrica para el mantenimiento de los servidores	Fomenta el desarrollo de tecnologías open source
Energía	**Alto**	**Alto**	Nulo	Nulo	**Medio**	Nulo
Social	**Bajo**	**Bajo**	**Alto**	**Alto**	Nulo	**Alto**
Auditivo	Nulo	**Medio**	Nulo	Nulo	Nulo	Nulo
Aire	Nulo	**Alto**	Nulo	Nulo	Nulo	Nulo

Suelo	Nulo	**Alto**	Nulo	Nulo	Nulo	Nulo

Tabla 101 - evaluación de cada factor de impacto ambiental

Referencias y escalas utilizadas

Referencias y Escalas			
SIGNO	Positivo (+)	Negativo (-)	
MAGNITUD	Alta (3)	Media (2)	Baja (1)
ALCANCE	Global (3)	Local (2)	Restringido (1)
PERSISTENCIA	Alta (3)	Media (2)	Baja (1)

Tabla 102 - Referencias y escalas de impacto ambiental

Matriz detallando el impacto de las acciones

	Fomentar el uso de tecnologías en la nube			
	Signo	Magnitud	Alcance	Persistencia
Energía		Alta	Local	Alta

	+	3	2	3
Social	+	Baja	Global	Media
		1	3	2

Tabla 103 - Matriz de impacto Fomentar el uso de tecnologías en la nube

	Reducción de recursos informáticos para la codificación inicial del proyecto.			
	Signo	Magnitud	Alcance	Persistencia
Energía	+	Alta	Global	Media
		3	3	2
Social	+	Baja	Global	Baja
		1	3	1
Auditivo	+	Media	Global	Media
		2	3	2
Aire		Alta	Global	Media

	+	3	3	2
Suelo	+	Alta	Global	Media
		3	3	2

Tabla 104 - Matriz de impacto Reducción de recursos informáticos para la codificación inicial del proyecto

	Creación de nuevas oportunidades laborales y de investigación			
	Signo	Magnitud	Alcance	Persistencia
Social	+	Alta	Global	Alta
		3	3	3

Tabla 105 - Matriz de impacto Creación de nuevas oportunidades laborales y de investigación

	Facilita la inserción de personas no relacionadas en el mundo del desarrollo de software			
	Signo	Magnitud	Alcance	Persistencia
Social		Alta	Global	Media

	+	3	3	2

Tabla 106 - Matriz de impacto Facilita la inserción de personas no relacionadas en el mundo del desarrollo de software

	Uso de energía eléctrica para el mantenimiento de los servidores			
	Signo	Magnitud	Alcance	Persistencia
Energía	-	Medio	Local	Media
		2	2	2

Tabla 107 - Matriz de impacto Uso de energía eléctrica para el mantenimiento de los servidores

	Fomenta el desarrollo de tecnologías open source			
	Signo	Magnitud	Alcance	Persistencia
Social	+	Alta	Global	Media
		3	3	2

Tabla 108 - Matriz de impacto Fomenta el desarrollo de tecnologías open source

Conclusión Análisis de Impacto Ambiental

Como conclusión vemos que el desarrollo e implementación del proyecto genera un impacto ambiental positivo, ya sea tanto en la sociedad aumentando puestos de trabajo y de investigación como en el medio ambiente mediante la reducción de recursos en el proyecto desarrollado por el sistema, y por lo tanto reduciendo la contaminación ambiental.

Trabajo Práctico Integrador

"Dirección de Proyectos de Sistemas"

Trabajo Práctico Integrador "Dirección de Proyectos de Sistemas"

1. Ordenar del 1 al 15 según la importancia (en el puesto N°1 la de mayor importancia) que le otorga a cada una de las funciones que deberías realizar como Jefe (o Director) de Proyecto, con una breve explicación de cada una.

1) Formulación del proyecto: Identificar los objetivos y alcances del proyecto, así como las tareas necesarias para llevarlo a cabo con los recursos disponibles. De esta manera, el director de proyectos define la forma en que se va a solucionar la problemática a la que apuntan los objetivos, especificando las actividades de la solución junto con su alcance.

2) Toma de decisiones: Realizar una elección entre distintas alternativas para resolver una situación particular que afecta el ciclo de vida del proyecto. Es importante que las decisiones tomadas estén alineadas con los objetivos del proyecto.

3) Administración eficiente de recursos y gestión de presupuestos: Gestionar correctamente los recursos disponibles para llevar a cabo las tareas definidas en la formulación del proyecto. La administración ineficiente de recursos ocasiona altos costos para el proyecto y, en casos extremos, puede llevar a la imposibilidad de terminarlo debido a la limitación de recursos.

4) Planificación y gestión de la planificación: Ubicar las actividades del proyecto en el tiempo, estableciendo su duración, fecha de inicio, entre otros parámetros. Además, es necesario que el director del proyecto ajuste la planificación con el pasar del tiempo, para poder conocer su estado y fecha de finalización en todo momento.

5) Asignación de tareas y recursos: Establecer una relación entre la tarea que se debe llevar a cabo con el/los recurso/s que se utilizarán para ello. En esta función, es importante que la formulación sea correcta y completa, ya que tanto las actividades como los recursos especificados son necesarios para poder llevar a cabo la asignación.

6) Gestión de riesgos: Identificar vulnerabilidades y amenazas que se convierten en riesgos para el proyecto, determinando su impacto y especificando tratamientos para reducirlos. Estos riesgos pueden aparecer en cualquier etapa y, si no se realiza alguna acción para reducirlos, puede causar un alto impacto en el ciclo de vida del proyecto. También es importante que se modifique la planificación de acuerdo con la aplicación de medidas preventivas, ya que ellas forman parte de las actividades del proyecto.

7) Aplicación de técnicas y métricas de estimación de tiempo y esfuerzo y evaluación inicial del proyecto: Evaluar las tareas que se llevarán a cabo utilizando técnicas y métricas para estimar el tiempo y el esfuerzo necesario para realizar el proyecto. Los resultados de estas técnicas y métricas permiten tomar mejores decisiones relativas al seguimiento del proyecto, asegurando que el desarrollo de las actividades va por el camino correcto.

8) Diseño y ejecución de acciones para el logro de equipos equilibrados y efectivos: Lograr un ambiente de trabajo en equipo para llevar a cabo las tareas definidas con productividad. Es importante recalcar que, si el director de proyectos no promueve el trabajo en equipo efectivo, el rendimiento del equipo de trabajo será mucho menor y lo más probable es que se generen numerosos conflictos que obstaculicen el curso del proyecto.

9) Supervisión y control de cumplimiento: Supervisar el desarrollo de las tareas y controlar su cumplimiento de acuerdo con la planificación para poder cumplir con los objetivos propuestos. La supervisión permite apoyar al equipo y buscar soluciones en conjunto para cualquier problemática que se presente, aumentando la productividad. Es importante que el control de cumplimiento esté acompañado por técnicas de motivación, logrando que el equipo se esfuerce para llegar a las metas planteadas.

10) Aplicación de retroalimentación y resolución de conflictos: Brindar soporte al equipo mediante constante comunicación y aplicación de técnicas de resolución de conflictos para eliminar los problemas que obstaculizan el desarrollo de las tareas. El director de proyectos debe asegurar el respeto y el buen trato en todo el ciclo de vida del proyecto, evitando conflictos irreconciliables entre los miembros del equipo que se traducen automáticamente a una baja en la productividad.

11) Ejercicio de distintos tipos de liderazgo: Guiar a las personas hacia el cumplimiento de los objetivos, aplicando distintos tipos de liderazgo dependiendo del equipo de trabajo y logrando influir en el equipo de trabajo para que sus integrantes se interesen en el logro de los objetivos propuestos. El interés de las personas del proyecto genera una mejora en la productividad.

12) Aplicación de diferentes estilos y técnicas de comunicación interpersonal: Lograr un buen intercambio de información con las personas del equipo mediante la aplicación de distintos estilos de comunicación, dependiendo del individuo con quien se desea establecer la comunicación. Es importante recalcar que la falta de comunicación es una de las principales

causas de generación de conflictos durante el trabajo en equipo. Por ello, el director debe esforzarse en crear un buen flujo de información para todo el proyecto.

13) Análisis de las personas, diseño y aplicación de técnicas de motivación individual: Evaluar las características de las personas del equipo y, en base a ellas, aplicar técnicas de motivación individual que influyan en su comportamiento y se traduzca en una mejora de resultados. Un ejemplo de esta función es asignar tareas de investigación a aquellos miembros con una personalidad curiosa, motivándolos para el desarrollo de las actividades.

14) Liderazgo de diseño de planes de testing, capacitación, implementación, manuales de usuario y documentación: Guiar el diseño de planes de testing, planes de capacitación, manuales de usuario y documentación necesaria para el proyecto, aportando una visión global del proyecto que sea de utilidad para la determinación de estos.

15) Generación de informes iniciales, parciales y finales: Crear reportes del proyecto al comienzo, a lo largo del desarrollo y al finalizar el proyecto para que las partes interesadas estén al tanto de su avance, logrando el apoyo necesario para llegar a los objetivos con la menor cantidad de obstáculos posibles.

2. Cuáles son las 5 principales funciones que cumplirá durante la fase anterior a la ejecución del proyecto, el “Jefe (o Director) de Proyecto” (pueden repetirse con las del punto anterior).

Las 5 principales funciones que el director del proyecto cumplirá durante la fase anterior a la ejecución del proyecto son:

1) Formular el proyecto.

2) Administración eficiente de recursos y gestión de presupuestos.

3) Planificar.

4) Asignar tareas y recursos.

5) Realizar el estudio de factibilidad.

3. Cuáles son las 5 principales funciones que cumplirá durante la fase de ejecución del proyecto, el "Jefe (o Director) de Proyecto" (pueden repetirse con las del punto 1).

Las 5 principales funciones que el director del proyecto cumplirá durante la fase de ejecución del proyecto son:

1) Toma de decisiones.

2) Diseñar y ejecutar acciones para el logro de equipos equilibrados y efectivos.

3) Supervisión y control de cumplimiento.

4) Aplicar retroalimentación y resolución de conflictos.

5) Realizar la evaluación del proyecto mediante diferentes técnicas.

4. Si los obligaran a incorporar al equipo del Proyecto a 2 personas, en qué momento los incorporaría, en cuál puesto y perfil y qué actividades les asignaría.

Si tuviésemos que incorporar al equipo a 2 personas, las incorporaríamos al finalizar la etapa de desarrollo en el puesto de Analista Funcional.

Analista funcional
Su función principal es la de realizar tareas de relevamiento y análisis de los sistemas informáticos.
Perfil
Requerimientos objetivos: ● Edad entre 20 y 35 años. ● Tener computadora portátil para desarrollar las actividades. Características personales generales: ● Habilidad analítica. ● Proactividad. ● Iniciativa.

- Confiabilidad.
- Orden y disciplina personal.
- Integridad.

Conducta y actitudes:

- Capacidad de trabajar en equipo con buen manejo de las relaciones interpersonales.
- Capacidad de resolución de problemas.

Características personales especiales:

- Autonomía.
- Adaptabilidad y flexibilidad.
- Colaboración.

Nivel educativo específico:

- Formación: Analista en Sistemas o afín.

Habilidades:

- Conocimiento de metodología tradicional de desarrollo de software.

● Experiencia laboral comprobada de al menos un año en puestos similares.

Tabla 109 – Puesto Analista Funcional

La asignación de actividades sería de la siguiente manera:

- Persona 1: Le asignaríamos la creación de un plan de capacitación a usuarios.
- Persona 2: Le asignaríamos la creación de un manual de usuarios.

Seleccionamos esas dos actividades porque, de todas las actividades del proyecto, son las que menos conocimientos técnicos requieren. Las personas por incorporar podrían llevarlas a cabo sin problemas luego de una breve inducción al proyecto. Además, estas tareas no tienen una alta complejidad, pero sí demandan tiempo, por lo que las personas que inicialmente fueron asignadas a estas actividades podrían hacer otra actividad.

5. Decidir qué estilo de liderazgo se deberá utilizar durante la ejecución del Proyecto, con la fundamentación correspondiente. Recordamos que los estilos de liderazgo pueden ser:

LIBRE: Cuando se dispone de personas en el equipo de trabajo que tienen alto grado de preparación, capacidad y responsabilidad.

DEMOCRÁTICA: Cuando se intenta lograr el tratamiento participativo de todos los temas, situaciones y llegar a decisiones por consenso.

AUTOCRÁTICA: Cuando por diferentes motivos, no se puede aplicar ninguna de las anteriores y se necesitan tomar y ejecutar decisiones rápidas.

Durante la ejecución del proyecto se utilizará el estilo libre, ya que todos los miembros del equipo son altamente responsables y tienen los conocimientos técnicos necesarios para llevar las actividades sin problemas. Además, poseen una alta capacidad de resolución de conflictos, en caso de que estos aparezcan, por lo que pueden tomar decisiones de forma autónoma (siempre que el problema no sea demasiado complejo) sin la necesidad de intervención del resto del equipo de trabajo.

6. Detallar los principales 10 riesgos que pueden aparecer en el proyecto, cuáles serían sus consecuencias y qué impacto tendrían esas consecuencias. Además, detallar cuáles son las medidas preventivas para cada uno de los riesgos. Recordamos que las medidas preventivas tienen como objetivo reducir la probabilidad de ocurrencia de cada riesgo o reducir el impacto que produciría cada riesgo.

Riesgo	Consecuencias	Impacto	Medidas preventivas
Constantes cambios y alcance del proyecto en continuo crecimiento.	Alcances poco claros en los integrantes del equipo. Retrasos y cambios en la planificación del proyecto.	Alto.	Realizar reuniones de seguimiento y control de avances ante cambios en los requisitos. Relevamiento detallado para una definición de requisitos precisa.
Falta de experiencia con el dominio del negocio.	Mal análisis y definición de los requerimientos del sistema.	Alto.	Realizar relevamiento detallado del negocio, incluyendo sistemas similares

			que se encuentren en funcionamiento.
Falta de comunicación en el equipo.	Baja productividad. Escaso control de avances e incumplimiento en la planificación. Objetivos poco claros en el equipo.	Medio.	Utilizar herramientas de gestión de proyectos, comunicación y control de avances para mejorar la calidad del manejo de información en el proyecto.
Falta de conocimiento técnico de las tecnologías a utilizar.	Necesidad de ajustar la planificación para la inclusión de actividades de investigación. Retraso en los tiempos del proyecto.	Bajo.	Considerar periodos de investigación necesarios para tecnologías desconocidas.
Estimación inadecuada de tiempos de ejecución de las actividades.	Aumento de costos del proyecto. Demoras en el proyecto.	Alto.	Realizar reuniones con frecuencia para comunicar cualquier cambio o inconveniente con el desarrollo del proyecto. Estimar las tareas en conjunto con expertos.
Estimación de costos inadecuada.	Alta posibilidad de no culminar el proyecto. Incumplimiento en pagos. Imposibilidad de adquirir recursos necesarios para el proyecto.	Alto.	Definición detallada en mano de obra y recursos a utilizar en el proyecto. Estimación de costos a realizar por profesional con experiencia o amplio conocimiento del negocio.

Extravío de documentación del proyecto.	Retraso en la ejecución de actividades. Aumento de costos del proyecto.	Medio.	Realizar backup automatizado de la información periódicamente.
Fallos en el diseño del sistema.	Aumento de errores en etapas posteriores. Rediseñar en etapas tardías del proyecto. Aumento de costos y retraso del proyecto.	Alto.	Utilizar patrones de diseño, seguimiento de normas y estándares que garanticen la calidad de esta etapa.
Cambios o bajas en el equipo de trabajo.	Retrasos en la planificación del proyecto. Cambios en la planificación y asignación de recursos. Baja en la motivación del equipo.	Alto.	Prever un plan de capacitación para la integración de nuevos miembros al equipo de trabajo. Asignar tareas con tecnologías adecuadas al proyecto y con curva de aprendizaje rápida en caso de adquirir un reemplazo.
Fallos en hardware y conectividad.	Tiempo perdido durante etapas de trabajo. Pérdida de información. Cambios en la asignación de recursos.	Bajo.	Realizar respaldo de la información periódicamente. Disponer de equipos de respaldo en caso de bajas. Mantenimiento en equipos electrónicos.

Tabla 110 – Principales riesgos

7. Decidir cuál enfoque de resolución de conflictos aplicará en supuestas situaciones (que también detallará) que se le puedan presentar durante el proyecto. Si tuviera que aplicar los conceptos de negociación, cuáles aspectos consideraría.

Situaciones de conflicto que se pueden presentar

- Conflicto entre compañeros de equipo:

En este caso utilizaría 2 enfoques, por un lado, evasivo para que los compañeros se calmen, y una vez estén más tranquilos y posteriormente un enfoque de colaboración para arreglar las relaciones entre los individuos y poder seguir con el proyecto sin contar con una tensión constante y falta de cooperación entre estos compañeros.

- Desacuerdos respecto a aspectos del diseño del sistema:

Principalmente se aplicará un enfoque de colaboración ya que en el planeamiento de un sistema es muy común estar en desacuerdo en ciertos aspectos de este, pero al colaborar se puede llegar a una solución integradora que permita solucionarlo teniendo en cuenta los puntos de cada parte. Se puede lograr un acuerdo entre partes.

- Un individuo no se dispone o no le da la prioridad necesaria al desarrollo del proyecto:

Dependiendo la situación personal de esta persona se pueden tomar 2 enfoques: De arreglo para conseguir un acuerdo temporal que nos permita avanzar con el proyecto o si la persona no muestra signos de querer colaborar un enfoque más agresivo puede ser útil para lograr colaboración en un momento crítico del proyecto.

En todos los casos los aspectos más importantes a tener en cuenta a la hora de negociar serán

- Definir previamente los temas a tratar.
- Recopilar la mayor cantidad posible de información sobre la otra parte involucrada.
- Generar una lista de posibles soluciones que podrían hacerse.
- Eliminar comportamientos negativos como interrumpir, gritar, amenazar, etc.
- Hacer muchas preguntas al otro, para conocer mejor sus necesidades e intereses.
- No Interrumpir a la persona mientras se expresa.

8. Detallar al menos 5 técnicas de motivación que utilizará durante el proyecto (indicando si se trata de técnicas de motivación positiva o negativa), y detallar en qué tipos de situaciones sería necesario aplicar cada una.

- Ajustar las tareas al Perfil de la persona - (Motivación Positiva):

Esta motivación la utilizamos en todo el desarrollo del proyecto seleccionando las tareas acordes a la afinidad que tienen los integrantes con la misma. Por Ejemplo: Las tareas de codificación se le asignan a la persona que le entusiasme y disfrute escribir el código.

- Hacer partícipe a los integrantes en decisiones Importantes - (Motivación Positiva):

Una cuestión que nos parece importante es evitar que las decisiones sean tomadas por solo algunos integrantes del grupo y los otros se dediquen solamente a la elaboración de lo decidido. Esto genera descontento en estas personas por lo que las decisiones clave para el desarrollo del proyecto serán discutidas por todo el grupo, buscando la opinión de los integrantes. De esta manera los integrantes se sienten parte fundamental del proyecto logrando una mayor motivación.

- Dar flexibilidad en la asignación de tareas - (Motivación Negativa):

Nos encontraremos con situaciones en las que no todos los participantes del grupo tengan la misma libertad de tiempos para realizar actividades relacionadas al proyecto ya sea por razones personales o de trabajo/estudio. Para este caso se permitirá cierta flexibilidad en la cantidad de tareas que se le asigna a esa persona en ese momento dado.

- Proveer un espacio de buena y libre comunicación - (Motivación Negativa):

Los participantes del proyecto deben sentirse libres de opinar y discutir temas intrínsecos al proyecto sin preocuparse de poder ser agredidos o juzgados por los demás integrantes. Por esta razón se va a incentivar a mantener una buena comunicación saludable en el grupo de trabajo.

- Reconocer la importancia del trabajo realizado - (Motivación Positiva):

Se buscará reconocer los aportes de los integrantes y sus trabajos realizados para asi lograr que se sientan motivados a realizar más tareas de importancia. Principalmente se implementará en momentos en los que se tengan que realizar tareas largas/tediosas/difíciles de esta manera se mantendrá motivado a continuarla al recibir reconocimiento por esto.

Algo que se puede notar al analizar las técnicas de motivación que se usarán durante la elaboración del proyecto es que la gran mayoría de técnicas de motivación no van a una situación puntual sino a mantener un equipo motivado durante todo el ciclo de vida de desarrollo del proyecto, que como equipo nos parece un aspecto extremadamente importante.

9. Describir el método de conversión del Sistema (para pasar del sistema actual al nuevo, por ej. directo, paralelo, por etapas, piloto o alguna combinación de ellos), con todas las actividades a realizar. Se debe registrar en este punto no sólo el método y las actividades sino también la justificación correspondiente al máximo nivel de detalle.

Debido a que el proyecto de Accelerate es una herramienta que puede utilizar cualquier tipo de cliente y no una empresa en particular que se debe adaptar el método de conversión lo decidirá el usuario o empresa que lo desee implementar. De todas formas, hemos planteado a continuación la forma de conversión que consideramos conveniente para aquellas compañías que deseen incorporar el producto de Accelerate a su flujo de trabajo.

Método de Conversión por Etapas en proyecto Piloto

Consideramos que la mejor forma de pasar desde la codificación manual de toda la infraestructura, el código de microservicios, y el aprovisionamiento en la nube a la generación automática mediante el uso de Accelerate es a través de una transición por etapas principalmente para empresas grandes que desarrollan varios productos de software. Estos pueden realizar una transición inicial en un proyecto piloto en el que se utilice la herramienta a la par de la codificación manual tradicional.

La razón principal para esto es que la compañía puede adaptarse al uso de la herramienta y realmente ver en qué casos de uso le puede sacar el mayor provecho. Por ello, la transición inicialmente se daría generando los microservicios más sencillos mediante la herramienta y los más robustos a través del método tradicional. A medida que la compañía gane experiencia y se sienta más cómoda trabajando con la herramienta generará más y más parte de la infraestructura con la herramienta hasta finalmente hacer el 95% de la generación con la misma.

Debido a que hay ciertas partes o casos de uso complejos que no son óptimos a generarse de manera automática y es preferible hacerlos manual, Accelerate probablemente no reemplace completamente a la forma tradicional. Sin embargo, al hacer la transición por etapas, la compañía puede detectar mucho mejor en qué casos de uso específicos es preferible la generación automática y cuando no.

Actividades

- Configuración del servidor y despliegue de Accelerate: Se realiza la configuración inicial de un servidor en la nube y se despliega Accelerate en el mismo, permitiendo el acceso a través de una interfaz web.

- Configuración de terminales de trabajo: Se debe instalar un navegador web en cada terminal de trabajo para que los futuros usuarios del sistema puedan acceder a él.

- Capacitación: Inicialmente se debe capacitar a las personas en el uso de la herramienta para que sea utilizada de forma correcta y se le pueda sacar el mayor provecho a la misma.

- División inicial de generación automática: Luego de la etapa de diseño del sistema y antes de empezar a codificarlo, se deberá decidir qué parte de este se generará automáticamente con Accelerate.

- Formación de equipos: Se deben formar equipos de trabajo que trabajarán en conjunto, algunos generando el código manual de las partes más robustas y los otros generando automáticamente los demás microservicios/servicios cloud.

- Distribución de tareas: Se debe distribuir correctamente las tareas de cada equipo.

- Distribución de roles: Se deben asignar los roles dentro del equipo que utiliza Accelerate (Administrador, Desarrollador, etc.).

- Desarrollo, generación y conexión con los servicios creados: El equipo que trabaje con Accelerate utilizara la herramienta para generar los modelos del sistema a generar. Una vez terminada la actividad, se genera el código de los microservicios e infraestructura y se procede a conectarlos con los servicios que fueron generados manualmente.

- Evaluación de resultados: Una vez finalizada la actividad anterior, el equipo se reunirá para evaluar la utilidad de la herramienta y el desempeño de esta en el flujo de trabajo en equipo, detectando así qué otros servicios podrían haber sido generados con la herramienta.

- Planteamiento de la siguiente etapa: Con la etapa finalizada se decidirá si se expandirá el uso de la herramienta dándole mayor responsabilidad al generar un porcentaje más grande del sistema. Luego de esto, se repiten las actividades hasta integrar completamente la herramienta en el flujo de trabajo del equipo de desarrollo.

- Implementación de Accelerate en todos los proyectos: Finalmente, si se observa un buen funcionamiento en el proyecto piloto, se puede propagar el flujo de trabajo con Accelerate a los demás proyectos de la organización.

Trabajo Práctico Integrador

“Gerenciamiento de Sistemas”

Trabajo Práctico Integrador "Gerenciamiento de Sistemas"

1) La empresa está por construir un edificio nuevo de Data Center. Para ello está nivelando el terreno donde construirá el edificio, en una sola planta, de 500 m2. Detallar principales recomendaciones generales "técnicas y de seguridad física" para el Data Center, tanto para la fase de construcción del edificio como para toda la infraestructura, amoblamientos e instalaciones que sean necesarias.

A continuación, se enumeran las recomendaciones para la construcción del Data Center:

- Ubicar los equipos informáticos de tal manera que no tengan contacto con humedad (ubicando el Data Center lejos de los baños, por ejemplo).
- Conectar dichos equipos al cableado de corriente limpia para evitar inconvenientes eléctricos. Para ello, se puede utilizar un estabilizador para transformar la línea a corriente limpia.
- Asegurar que la ubicación de los equipos permite la conexión con los proveedores de Internet y telefonía.
- Ubicar el Data Center en un lugar que minimice las posibles vulnerabilidades que puedan afectar a los equipos (e.g. no colocar en un subsuelo, ya que sería una de las primeras ubicaciones en sufrir daños por inundación).
- Permitir el acceso solamente al personal que intervenga físicamente en el Data Center (personal de limpieza, personal de informática, etc.) mediante cerraduras eléctricas con control de acceso.

- Realizar una adecuada gestión de las llaves (de racks, por ejemplo) del Data Center mediante un tablero organizador rotulado.

- Definir un plan de recuperación ante desastres que permita restablecer las operaciones del Data Center lo antes posible.

- Implementar un mecanismo de recuperación que permita restablecer la información ante posibles pérdidas y alojar los dispositivos de backup en cajas de seguridad ignífugas que eviten una posible pérdida.

- Colocar un dispositivo UPS (Uninterruptible Power Supply) que asegure el servicio eléctrico ante posibles cortes de luz. Es importante que dicho dispositivo esté en una ubicación distinta a la del Data Center, para evitar que una misma vulnerabilidad afecte el suministro de energía y al UPS.

Figura 147 - Fotografías de un UPS. Nótese las múltiples salidas para alimentar dispositivos electrónicos.

Hay tres tipos de UPS ordenados por eficacia de manera ascendente:

- Standby: Recurre a una batería interna separada del suministro ante un corte de electricidad. El problema con este tipo de UPS es que se pierden unos milisegundos al cambiar a la batería interna

- Línea interactiva: Similar al UPS Standby pero con el agregado de un regulador de voltaje que permite corregir fluctuaciones.
- Online: Utiliza una batería directamente conectada al suministro, permitiendo el cambio a batería automáticamente, resolviendo el problema de las dos categorías mencionadas.

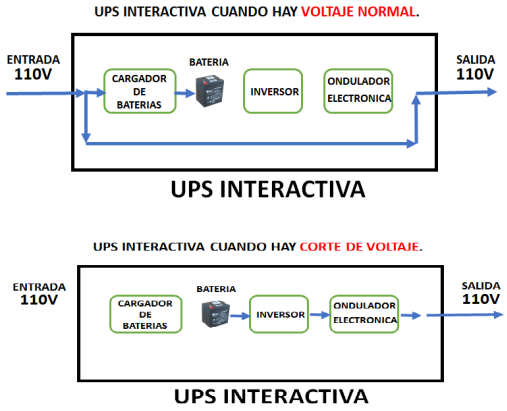


Figura 148 - Ejemplo de funcionamiento de UPS con línea interactiva.

Lo ideal es colocar un dispositivo UPS online, para evitar esos milisegundos sin energía que pueden dañar a los equipos informáticos. Sin embargo, estos equipos son de alto costo, por lo que, si el presupuesto es limitado, no existe otra opción más que acceder a dispositivos UPS más económicos.

- Acompañar el dispositivo UPS con un grupo electrógeno que permita suministrar energía eléctrica en caso de cortes de luz de alta duración. Los grupos electrógenos pueden ser alimentados con diferentes combustibles, tales como gasoil, gas o nafta, siendo estos últimos los más eficientes y con menor impacto sobre el medio ambiente pero con un mayor precio.

Figura 149 - Fotografía de un grupo electrógeno.

- Utilizar un grupo electrógeno con arranque eléctrico automático, ya que comienza a funcionar automáticamente ante un corte de energía eléctrica y evita el agotamiento de la batería UPS. En caso de no ser posible la compra de dicho dispositivo, contar con personal que ante un corte de electricidad se dirija rápidamente a la sala donde se ubica el grupo electrógeno con arranque manual lo antes posible.
- Instalar sistemas de monitoreo remoto con racks inteligentes que permitan controlar el estado del Data Center a distancia. A continuación, se describen las distintas herramientas que se pueden utilizar para el monitoreo de racks:
    - Sensores de temperatura, humedad y presión de aire, para evitar inconvenientes climáticos.
    - Sensores de puerta de bastidor, para evitar trabajos no autorizados.
    - Detectores de vibraciones, para monitorear las posibles vibraciones que podrían dañar los equipos tales como terremotos o trabajos de construcción.
    - Sensores de energía, para verificar el suministro eléctrico de los equipos del rack y evitar pérdidas por fluctuaciones en el servicio (e.g. variaciones en el voltaje).

Figura 150 - Ejemplo de rack inteligente con sensores integrados y monitoreo remoto.

- Implementar mecanismos de seguridad (cámaras de vigilancia, alarmas, personal de seguridad, etc.) para evitar el robo o hurto de los equipos, ya que muchos de ellos pueden ser de alto valor.
- Realizar la construcción de la infraestructura con materiales que permitan la aislación sonora y térmica para evitar daños en los equipos informáticos.
- Diseñar el Data Center teniendo en cuenta la refrigeración y distribución del aire (diseñando pasillos bien separados para permitir el paso del aire, sensores de temperatura, sensores de humedad, etc.). En el caso de la refrigeración, lo ideal es instalar un sistema de refrigeración líquida basado en grandes contenedores de agua fría que es bombeada a través de tuberías en contacto con los racks. Sin embargo, este sistema es muy costoso o directamente inviable para algunas compañías, por lo que se puede optar por una opción menos eficiente como un sistema de refrigeración por aire acondicionado.

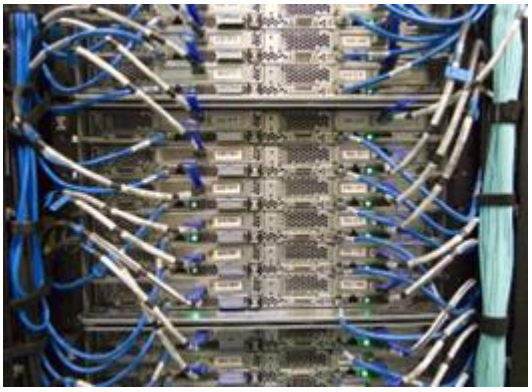


Figura 151 - Refrigeración líquida en un Data Center. Nótese las tuberías con líquido refrigerante que ingresan a cada servidor.

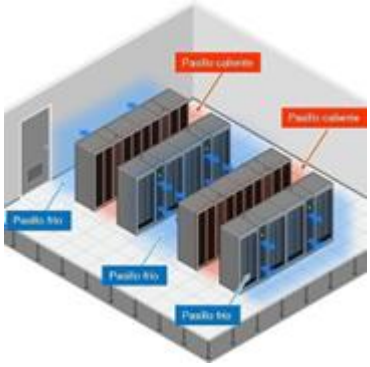


Figura 152 - Ejemplo de distribución del aire en un Data Center

- Contar con sistemas automáticos de extinción de incendios por gas que detectan y extinguen los posibles incendios antes de que el fuego evolucione. Estos sistemas funcionan a través de válvulas o activadas mediante sensores de humedad y temperatura, y que dispersan el agente gaseoso de forma homogénea por la sala protegida. A continuación, se describen las partes del sistema de extinción:
    - Almacenamiento mediante cilindros que contienen el gas.
    - Red de tuberías que permiten la distribución del agente.
    - Difusores de descarga combinados con sensores de temperatura y humedad que permiten la liberación del gas para extinguir el incendio.

Además, es importante contar con extinguidores manuales de clase C (para incendios generados por equipamiento eléctrico), ya que son de gran utilidad en caso de emergencia.

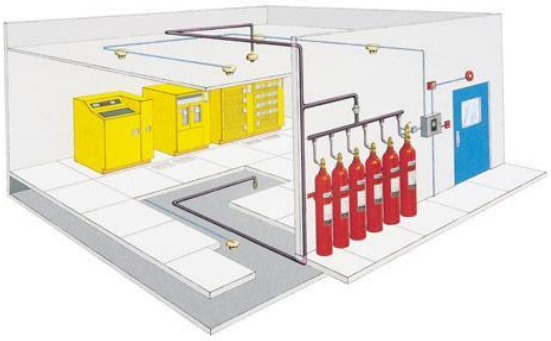


Figura 153 - Sistema automático de extinción de incendios por gas (almacenado en tubos de color rojo).

2) Si consideramos que trabajan, como mínimo, dos personas en cada una de las áreas detalladas, cuál es el tipo de estructura organizativa mostrada en el organigrama. Además, podría explicar cuáles otros tipos de estructuras organizativas podrían utilizarse.

La estructura organizativa mostrada en el organigrama es jerárquica departamental.

Además, podrían utilizarse las siguientes estructuras organizativas.

- Por función: En esta forma de departamentalización, las personas que se dedican a una actividad funcional, por ejemplo, gestión de datos o administración, se agrupan en una unidad. Esta estructura organizativa, favorece la especialización, capacitación y un buen control desde la cima.
- Territorial o geográfica: En caso de que la empresa opere en regiones geográficas extensas, podría utilizarse la estructura organizativa territorial.
- Por cliente: Dado el caso de que la empresa fuera por ejemplo una consultora y tenga múltiples clientes, podría utilizarse la estructura organizativa por cliente. Esta se da

cuando las actividades de la empresa, en favor de sus clientes, son puestas bajo la responsabilidad de un jefe. Estos clientes tienen necesidades bien diferenciadas.

- Por producto: Esta estructura organizativa se utiliza en empresas grandes y podría utilizarse en caso de desarrollar múltiples productos dentro de la empresa. Permite delegar a un ejecutivo divisional, autoridad sobre las funciones relativas a un producto o línea de producto. Esto permite establecer costos con mayor facilidad para cada caso y poner atención sobre cada línea productiva específica.
- Por proceso o equipo: Dentro de la estructura organizacional por proceso o equipo, se propone la unión de personal y materiales en un mismo punto, para el cumplimiento de un proceso en particular. Esta estructura simplifica la capacitación ya que se utilizan habilidades especiales dentro del área de implementación.
- Matricial: La estructura organizativa matricial podría utilizarse en caso de llevar a cabo múltiples proyectos en simultáneo, donde permite gestionar las distintas áreas de la empresa que se involucran y sus recursos. En esta estructura, cada empleado depende tanto de un gerente funcional como de un gerente de proyecto.

3) Detallar y explicar como mínimo seis servicios que brinde el área seleccionada (sea interna o externa a la empresa).

El área elegida es la de Nuevos Proyectos de TI:

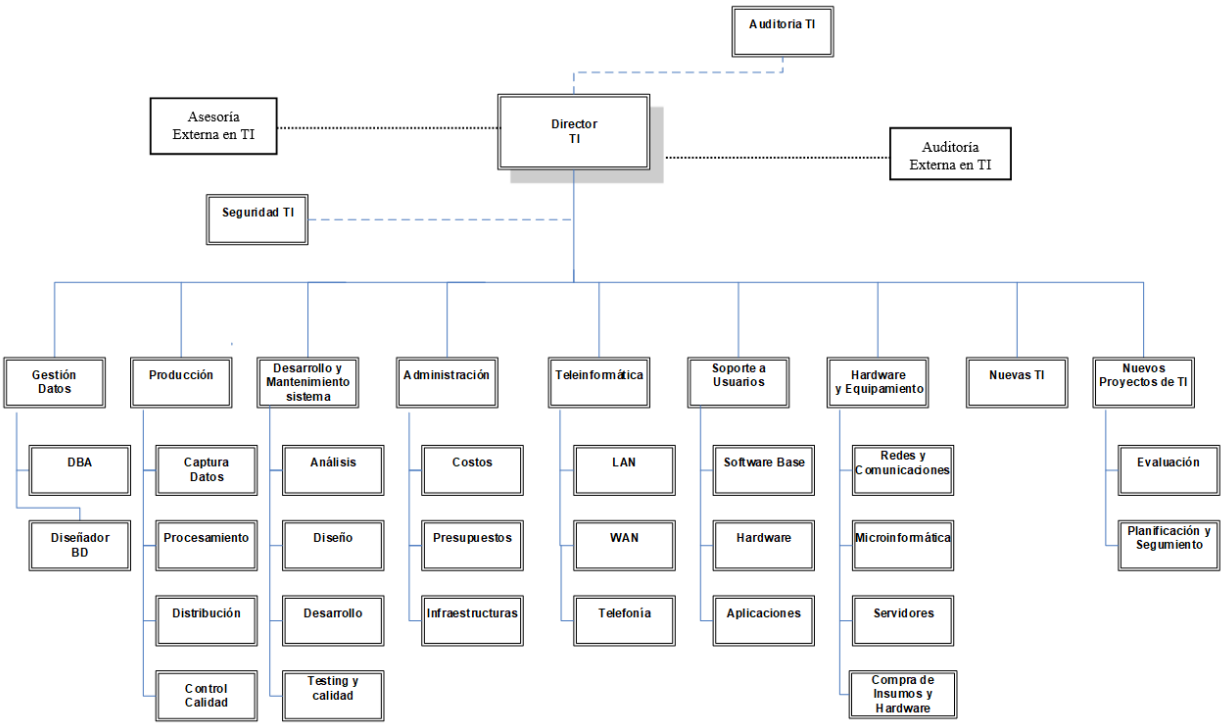


Figura 154 - Área nuevos proyectos de TI

El área mencionada brinda los siguientes servicios:

1. Evaluación de proyectos de T.I. necesarios dentro de la empresa:

Es necesario que las inversiones en proyectos de TI se administren estratégicamente y se enfoquen en agregar valor.

La evaluación de proyectos de T.I. comprende el establecimiento de criterios para identificar aquellas ideas que puedan concretarse desde aspectos técnicos, económicos y financieros, para su aceptación, rechazo o reformulación del proyecto.

2. Gestión de recursos para nuevos proyectos:

Es importante llevar a cabo una buena gestión de los recursos ya que es uno de los principales factores de éxito de los proyectos. Las iniciativas que no cuentan con los recursos necesarios en el momento indicado, tienen altas probabilidades de que ocurran problemas para alcanzar los límites de tiempo acordados, que se comprometa la calidad del proyecto y quedarse sin fondos para la concreción del mismo.

Dentro de este servicio se pueden destacar actividades como:

- Planificación de la gestión de recursos.
- Estimación de los recursos para las actividades.
- Contratación o adquisición de recursos.
- Desarrollo del equipo del proyecto.
- Gestión del equipo.
- Control de recursos.

3. Presupuestación para proyectos de TI:

El presupuesto de un proyecto es un plan en el que se detalla cuánto se gastará, cuándo y para qué. Al crear un presupuesto con anticipación y se usa para controlar el gasto a lo largo de tu proyecto, se reduce el riesgo de quedarse sin recursos o excederse en costos.

A medida que el proyecto avanza, se puede utilizar el presupuesto para comparar el gasto real con el gasto presupuestado y mitigar los costos adicionales que puedan surgir.

Teniendo como base los objetivos del proyecto, alcance, tiempo, entregables y recursos necesarios se procede a realizar el presupuesto óptimo para el proyecto, estimando costos de las

actividades a desarrollar, comparando presupuestos de proyectos similares, considerando diferentes escenarios y documentando en que parte del presupuesto se emplea para cada actividad.

4. Planificación de nuevos proyectos:

Dentro del servicio de planificación de nuevos proyectos de T.I. se deben identificar los objetivos y alcances de este, especificar los entregables y actividades necesarias para concretarlos.

De acuerdo a los objetivos y actividades a desarrollar, se debe diseñar un cronograma, por ejemplo, empleando un diagrama de Gantt, identificando las dependencias entre sus actividades.

Además, esta área se encarga de realizar una evaluación de riesgos y análisis de factibilidad del proyecto.

5. Seguimiento para proyectos que se encuentran en ejecución dentro de la empresa:

El área puede llevar el seguimiento y control de los proyectos mediante un tablero de comando, para posteriormente tomar medidas correctivas en la implementación del mismo, y en favor de que se cumplan los plazos y costos establecidos.

6. Coordinaciones generales entre proyectos de T.I.:

La coordinación entre proyectos de T.I. agiliza el flujo de trabajo de sus tareas y la colaboración entre estos. El coordinador de proyectos es el responsable de mitigar los obstáculos que afectan a los equipos de proyectos. También, debe coordinar y optimizar el uso de recursos dentro de la

empresa y entre los distintos proyectos de T.I., supervisar el cronograma del proyecto y asegurarte de que el trabajo se realice según lo programado.

4) Con ejemplos del área seleccionada, explique las características de un equipo de trabajo efectivo y un equipo de trabajo equilibrado.

Equipos de Trabajo Efectivo:

- Libre expresión de todos los miembros:

Los miembros del equipo tienen la libertad de proponer nuevos proyectos que consideran convenientes de implementar en la compañía y a su vez pueden expresar su opinión libremente sobre las implementaciones que se están llevando a cabo en caso de que no se sientan conformes o no estén de acuerdo con la planificación o desarrollo de los mismos.

- Principio del trabajo en conjunto:

El equipo se organiza para investigar diferentes tecnologías que se puedan implementar para el problema que se necesita solucionar u optimizar, o el proyecto a implementar de esta manera se obtiene una mirada más global de las opciones disponibles. De esa manera se obtiene un aprovechamiento mayor de la creatividad y cualidades del equipo

Se distribuyen las tareas en equipos encargados de cada sección del proceso (pre evaluación, presupuestación, planificación y seguimiento).

- Todos están dispuestos a asumir riesgos:

En este caso se da desde 2 aspectos. Primero el equipo asume el riesgo de invertir tiempo evaluando la opción de implementar ciertos proyectos que no se conoce con certeza si serán efectivos.

Por otro lado, al momento de la implementación el equipo asume el riesgo de implementar una nueva tecnología en base a su investigación realizada sobre el producto/proceso y a la gestión de riesgos planificada.

- Hay objetivos comunes y metas claras bien arraigados en todos los miembros: Para el caso del área de nuevos proyectos de TI esto se da desde el aspecto de que los integrantes se encuentran comprometidos con los proyectos a implementar. Al momento de la preevaluación del mismo se establecen los objetivos y metas del proyecto en base a las cuales se rige el equipo
- Buena relación de los miembros con otros integrantes de otros proyectos y otras áreas, para aprovechar las experiencias ajenas y poner en valor las propias:

  Es algo principal en el área de nuevos proyectos de TI mantener una estrecha relación con las demás áreas ya que es la principal manera de conocer si el trabajo e investigación que realizan para la mejora de la compañía es efectivo.

  Esta área lo realiza principalmente en la etapa de seguimiento de los nuevos proyectos de TI, en los cuales obtienen retroalimentación constante sobre el proyecto realizado, las fallas que tuvo y las mejoras que se pueden implementar en base a las falencias detectadas.

Equipos de Trabajo Equilibrados:

- Cantidad de integrantes, de acuerdo con recomendaciones de alcance de control del líder:

El líder del equipo se encarga de distribuir a los integrantes de manera que una porción de ellos se encargue de investigar sobre el proyecto a implementar, desarrollos similares y la factibilidad del mismo, otra porción del equipo planea el proceso de implementación de la misma en la empresa y por último un equipo desarrolla un presupuesto en base a la información recibida por los demás equipos.

- Disponibilidad de tiempo: Se requiere que los participantes del equipo estén presentes durante sus horas de trabajo para el momento de la investigación y estos deben estar presente de manera imprescindible al momento de la implementación de las tecnologías.
- Roles (orientado a la tarea, orientado a la relación, etc.): El Rol está principalmente orientado a la tarea de investigación.
- Personalidad (introvertido, extrovertido, agresivo, sumiso, solitario, etc.): Si bien no se requiere una personalidad extrovertida para realizar la labor es una cualidad preferible para la misma ya que se necesitan buenas habilidades sociales para comunicarse con las áreas afines y posibles proveedores de servicio.
- Ingenio, creatividad, generación de ideas, inquietudes, nuevos proyectos, etc.:

Los participantes de esta área deben ser creativos, tener habilidad para resolver problemas y principalmente deben estar predispuestos a la investigación constante. Debido a que las tecnologías de hoy en día avanzan a pasos agigantados deben poder conocer e investigar estos avances para plantear proyectos que implementen las tecnologías óptimas para la tarea.

- Competencias técnicas y nivel de capacitación:

- Preferentemente deben tener capacitación en análisis y diseño de sistemas o afín.
- Conocimientos en herramientas de gestión de tiempos y de proyectos
- Conocimientos en metodologías de desarrollo de software.

5) Analizar la aplicación del "Coaching Eficaz" en el área seleccionada. O sea, de qué forma relevaría la situación del personal y cuáles acciones realizaría Ud. como jefe del área seleccionada para poder aplicar correctamente el coaching.

Para la aplicación del Coaching Eficaz se dividirá en distintos aspectos:

- Proveer posibilidades de crecimiento personal para el equipo mediante un plan de carrera y cursos/capacitaciones que permitan a los integrantes del área crecer e incrementar sus conocimientos. Esto logra incrementar no solo el nivel de capacitación general del área, también incrementa la motivación de los individuos de la misma.
- En el caso de los nuevos integrantes del área el jefe de nuevos Proyectos de TI realizará un seguimiento y los acompañará durante el proceso de adaptación en el área mediante reuniones cada 3 o 4 días laborales, en estas reuniones estarán los nuevos integrantes y las personas del equipo de trabajo con mayor experiencia del área. Durante este proceso se podrá resolver dudas e inquietudes de los nuevos integrantes.
- Se escucharán las propuestas de proyectos de parte de cada integrante en reuniones mensuales del área de manera que estos puedan participar activamente en el proceso de selección de proyectos, y será trabajo del jefe de área escuchar y evaluar estos para dirigirlos en la dirección correcta en caso de ser factible.

- Por último, el jefe de área debe tener en cuenta problemas o aflicciones de los integrantes del área para distribuir la carga de trabajo de forma que estos individuos puedan cumplir con sus tareas en base a el nivel de productividad actual de cada uno. Para esto el jefe de área tiene un canal de comunicación abierto por el cual los integrantes pueden avisar sobre problemas y a se pone a disposición de los empleados del área para realizar reuniones con estas personas cuando sea necesario.

6) Analizar la aplicación de "Retroalimentación a 360°" en el área seleccionada. O sea, cuáles serían todas fuentes de información y acciones que Ud. aplicaría como jefe del área seleccionada para poder aplicar correctamente la retroalimentación a 360°, para mejorar su propia gestión a cargo del área.

Debido a que el área de Nuevas Proyectos de TI está en estrecho contacto con el resto de las áreas ya que se encarga de investigar/evaluar/implementar herramientas para estas, lo más apropiado sería que el jefe del Área busque retroalimentación sobre la gestión y operación del departamento mediante la comunicación a las áreas a las que les brinda soporte.

El jefe de nuevos proyectos de TI entonces obtiene información desde fuentes internas, a través de conversaciones con los jefes de otras áreas a las cuales se les realizó alguna implementación de proyecto. De esta forma puede conocer la posición del área respecto del resto de la empresa.

En caso de proyectos externos a la empresa el jefe se comunica con los clientes de manera semanal o mensual dependiendo el proyecto para obtener información del funcionamiento de los proyectos implementados o el progreso observado desde la perspectiva de los clientes para un proyecto en desarrollo.

Por último, el jefe del área prepara encuestas que se realizan a la gente involucrada en los proyectos en la cual se realizan las siguientes preguntas:

- ¿Qué resultados tuvo la nueva herramienta implementada?
- ¿En cuánto varió la eficiencia del departamento luego de implementar el proyecto?
- En caso de que el proyecto lo pidiera un área específica ¿Cuánto fue el tiempo de respuesta del área desde el momento que se realizó una petición para evaluar el proyecto?
- ¿Qué complicaciones o dificultades se tuvieron al momento de la implementación y adaptación al nuevo sistema?
- ¿Cuál fue el aumento/reducción de costos?
- ¿Cómo califican la ayuda brindada por los participantes de la implementación?
- ¿En cuánto varía el proceso de implementación en base a la planificación realizada por el área?

Con esta información, el jefe de área puede determinar la eficiencia y eficacia de las soluciones brindadas por el equipo de Nuevas Proyectos. Esto lo llevará a aplicar medidas correctivas de ser necesario en caso de encontrar fallas o falencias en el proceso del área, logrando de esta manera una mejora continua del departamento.

7) Detallar las funciones que podría tener un Tablero de Comandos del área seleccionada y el diseño de la pantalla principal del mismo.

Mediante este tablero de comando se podrá analizar las desviaciones, permitiendo que las personas dentro del área conozcan la situación actual y realicen proactivamente las acciones correctivas necesarias a tiempo.

En el tablero de comandos del área de nuevos proyectos de T.I., de acuerdo a sus funciones, podríamos ver por ejemplo:

- Validación y filtrado de los datos a incorporar de cada fuente, realizando el chequeo de datos nulos o inconsistentes e incorporación de los datos validados desde las bases de datos OLTP de cada proyecto específico del área de nuevos proyectos de T.I. para su presentación en el tablero, como puede ser el indicador de tareas completadas o el beneficio neto por proyecto.

- Construcción de gateways e interfaces de equipos y otros sistemas desde los D.W. teniendo un selector interactivo para variar los KPI que se observan por proyecto, obteniendo datos desde los sistemas particulares de cada caso y presentándose en el tablero.

- Automatización de los procesos de incorporación de datos, realizando pipelines de incorporación de datos desde las bases de datos de origen de cada proyecto, fundamentalmente para lograr un tablero de comandos que permita tomar decisiones proactivamente basadas en datos actualizados, como por ejemplo la ingesta automática de datos para conocer el porcentaje de retorno de inversión actual por proyecto y su evolución en el tiempo o los impedimentos nuevos los proyectos.

- Parametrización amplia de los módulos de incorporación de datos de fuentes externas para hacer transparente la ampliación del tablero. Esto permite poder escalar el tablero y añadir indicadores nuevos como por ejemplo en la parametrización realizada para seleccionar el

proyecto que se desea ver en el tablero. En adelante podría utilizarse esta parametrización para añadir nuevos KPIs por proyecto.

- Funciones automáticas de extracción, gestión, organización, explotación, relación y proyección de datos, al tomar datos desde las bases de datos transaccionales de cada proyecto e integrarlas en el tablero de comandos, como es el caso del indicador de proyectos completados por mes o resolución de impedimentos de avance, utilizando como fuentes todos los proyectos que lleva el área de nuevos proyectos de T.I.

- Parametrización de las reglas control, valores objetivo, información por excepción, alertas, detalles. Esto podemos verlo mediante el indicador del porcentaje de beneficio neto alcanzado en relación al estimado por mes por proyecto y su alerta establecida si esta baja de cierto valor objetivo dentro del área. También, podemos verlo en el indicador de tareas completadas sobre tareas objetivo por proyecto por mes y alertas si estos valores están lejos de completarse o con retrasos.

- Simulación de decisiones, implementando un indicador que muestre el retorno de inversión estimado para los próximos meses por proyecto, en base a las decisiones planteadas.

- Adaptación y utilización de las herramientas “EIS” para la facilidad de presentación, acceso y navegación. Podemos verlo implementado mediante Grafana u otra herramienta de Business Intelligence y el diseño de un tablero que cumpla con los KPI establecidos por el área como se puede ver en el mockup diseñado en el presente informe.

- Indicador de proyectos comenzados en el mes, respecto de valores objetivos establecidos en el área, como por ejemplo “comenzar más de X proyectos por mes”.

- Indicador del porcentaje de resolución de impedimentos de avance por mes, como indicador de una buena gestión de proyectos.

- Indicador del porcentaje de proyectos atrasados en el mes y su variación dinámica ante la resolución de los conflictos que ocurran.

- Funciones automáticas de incorporación de datos y parametrización de los proyectos en curso para establecer métricas para seguimiento de los proyectos que se implementaron, como puede ser retorno de inversión por mes por proyecto en una gráfica y su comparación con los valores objetivos establecidos en la planificación del proyecto. Además, puede plantearse una estimación a meses futuros e interactividad para cambiar de proyecto y su granularidad, ya sea a nivel de mes o semana.

Figura 155 - Tablero de comandos para el área

8) Elaborar una estrategia de mejora del área seleccionada, que contenga como mínimo 20 actividades a realizar en los próximos 2 años, distribuidas según el momento de ejecución (por ej. con cronograma mensual). La estrategia tiene que estar orientada a mejorar día a día la calidad en la gestión del área, por ej. mejorar el rendimiento del personal, mejorar los resultados, apoyar a los objetivos de la empresa u organización, tener una adecuada relación con otras áreas, eficiencia, generación proactiva, reducción de errores, mejoramiento de relaciones interpersonales, satisfacción continua de los Clientes internos y externos, potenciar fortalezas, aprovechar oportunidades, reducir debilidades y estar preparado para las amenazas, etc.

A continuación, se enumeran las actividades de mejora según un cronograma mensual para la mejora en la calidad de gestión del área:

- Mes 1: Analizar los procedimientos de inicio de proyectos con especialistas (planificación, presupuestación, firma de contratos, etc.) para evitar posibles demoras temporales causadas por tareas innecesarias.

- Mes 2: Implementar un sistema de capacitaciones mensuales respecto a nuevas tecnologías del mercado para la evaluación de futuros proyectos.

- Mes 3: Mejorar la comunicación con el área de Nuevas TI a través de reuniones semanales, para obtener una fuente de ayuda en caso de que se presente un posible proyecto que utilice una tecnología desconocida para el personal del área.

- Mes 4: Incorporar un tablero de comando que permita visualizar indicadores útiles para el área tales como la cantidad de proyectos en curso, la cantidad de proyectos que finalizan por mes, la entrada de nuevos proyectos por mes, etc.

- Mes 5: Crear un plan de actuación ante posibles desviaciones al llevar a cabo los procedimientos de inicio de proyectos.

- Mes 6: Capacitar al personal del área en técnicas de coaching eficaz y equipos efectivos para aumentar la productividad del área.

- Mes 7: Estudiar los problemas de proyectos pasados para buscar fallas en los procesos actuales y evitar inconvenientes a futuro.

- Mes 8: Encuestar a los equipos de trabajo y stakeholders de proyectos finalizados con el propósito de recibir retroalimentación respecto al seguimiento por parte del área.

- Mes 9: Incorporar un procedimiento de recibo y análisis de sugerencias para incorporar posibles mejoras a los procesos internos.

- Mes 10: Crear un equipo interdepartamental con el área de Nuevas TI, a fines de realizar la evaluación, presupuestación, planificación y seguimiento de proyectos basados en nuevas tecnologías.

- Mes 11: Implementar un sistema informático que elimine (o reduzca lo más posible) las tareas en papel y agilice los procesos del área. Es importante que la implementación se acompañe con tareas de capacitación respecto al funcionamiento del sistema.

- Mes 12: Establecer un plan de atracción de jóvenes profesionales al área para incorporar talento joven y promover su aporte de ideas innovadoras para mejorar los procesos del área.

- Mes 13: Programar actividades recreativas periódicas fuera del trabajo para mejorar el clima laboral del área (e.g. un partido de fútbol a la salida del trabajo los viernes).

- Mes 14: Implementar un sistema de trabajo híbrido (en las actividades que no necesitan la presencialidad del equipo) que permita hacer uso de los beneficios de la virtualidad (e.g. mejor aprovechamiento de los recursos, aumento en la motivación de los empleados, etc.).

- Mes 15: Elaborar un plan de auditoría semestral para monitorear el funcionamiento del área y realizar posibles correcciones.

- Mes 16: Realizar un análisis de debilidades y fortalezas del área para buscar nuevas oportunidades de mejora.

- Mes 17: Implementar una vía de comunicación rápida con clientes externos para brindar una mejor atención que impacte en el inicio de futuros proyectos a través de un email que replique los mensajes entrantes al personal correspondiente.

- Mes 18: Aplicar un mecanismo de motivación positiva que se traduzca en un aumento de la productividad, analizando y enriqueciendo las tareas repetitivas de los procesos.

- Mes 19: Mejorar las habilidades del personal del área en inglés para mejorar su entendimiento respecto a las nuevas tecnologías del mercado a través de clases in-company.

- Mes 20: Contratar a un equipo de reingeniería para rediseñar radicalmente aquellos procesos que no funcionen correctamente de acuerdo con los indicadores del tablero de comandos.

- Mes 21: Revisar los recursos informáticos utilizados en el área y actualizar los equipos obsoletos para mejorar la productividad.

- Mes 22: Elaborar un cronograma de charlas informativas, donde los miembros del área tendrán la posibilidad de explicar un tema relacionado al área en 10 o 15 minutos y así mejorar la capacitación del personal.

- Mes 23: Implementar un esquema de trabajo utilizando la suite de Microsoft (mensajería por Teams, emails mediante Outlook, etc.), que permita ordenar y agilizar la comunicación interna del área.

- Mes 24: Incorporar una casilla de comentarios, donde los miembros del área puedan realizar sugerencias sobre el sistema informático implementado en el mes 11. Además, esta casilla también puede ser utilizada para que los miembros con experiencia brinden soluciones a problemas frecuentes.

Conclusiones

En el presente trabajo se puede observar que mediante la aplicación de Accelerate dentro de proyectos de desarrollo de software, se logran excelentes resultados no sólo relacionados a la reducción del tiempo de desarrollo en la etapa inicial del proyecto, sino que también respecto al marco de trabajo de buenas prácticas que la generación automática establece como guía para el resto del proyecto.

A su vez brinda a analistas y arquitectos de software un marco de trabajo distinto centrado alrededor de la herramienta y provee al equipo un nuevo artefacto de documentación para el proyecto que, gracias a su funcionalidad de generación automática, asegura que no quedará desactualizada. Por último, permite a equipos inexpertos o sin personal especializado realizar despliegues en entornos cloud de manera sencilla al proveer todas las herramientas necesarias mediante el código de aprovisionamiento de infraestructura.

Como proyección al futuro, la expansión de este trabajo será la ampliación hacia nuevas áreas como el testing, microservicios frontend, y soporte a otras tecnologías como Amazon Web Services en el caso de los proveedores en la nube y/o generación automática en otros lenguajes como Java, Ruby, entre otros.

Bibliografía y Referencias

[1] Amplication: https://amplication.com/

[2] Vemto: https://vemto.app/

[3] JHipster: https://www.jhipster.tech/

[4] Fogg: https://github.com/chanzuckerberg/fogg

[5] DhiWise: https://www.dhiwise.com/

[6] Acceleo: https://www.eclipse.org/acceleo/

[7] Sirius: https://www.eclipse.org/sirius/

[8] PostgreSQL: https://www.postgresql.org/

[9] Terraform: https://www.terraform.io/

[10] Docker: https://www.docker.com/

[11] Kubernetes: https://kubernetes.io/es/

[12] Github Actions: https://docs.github.com/es/actions

[13] MongoDB: https://www.mongodb.com/es

[14] Project Libre: https://www.projectlibre.com/

[15] Figma: https://www.figma.com

[16] Github: https://github.com

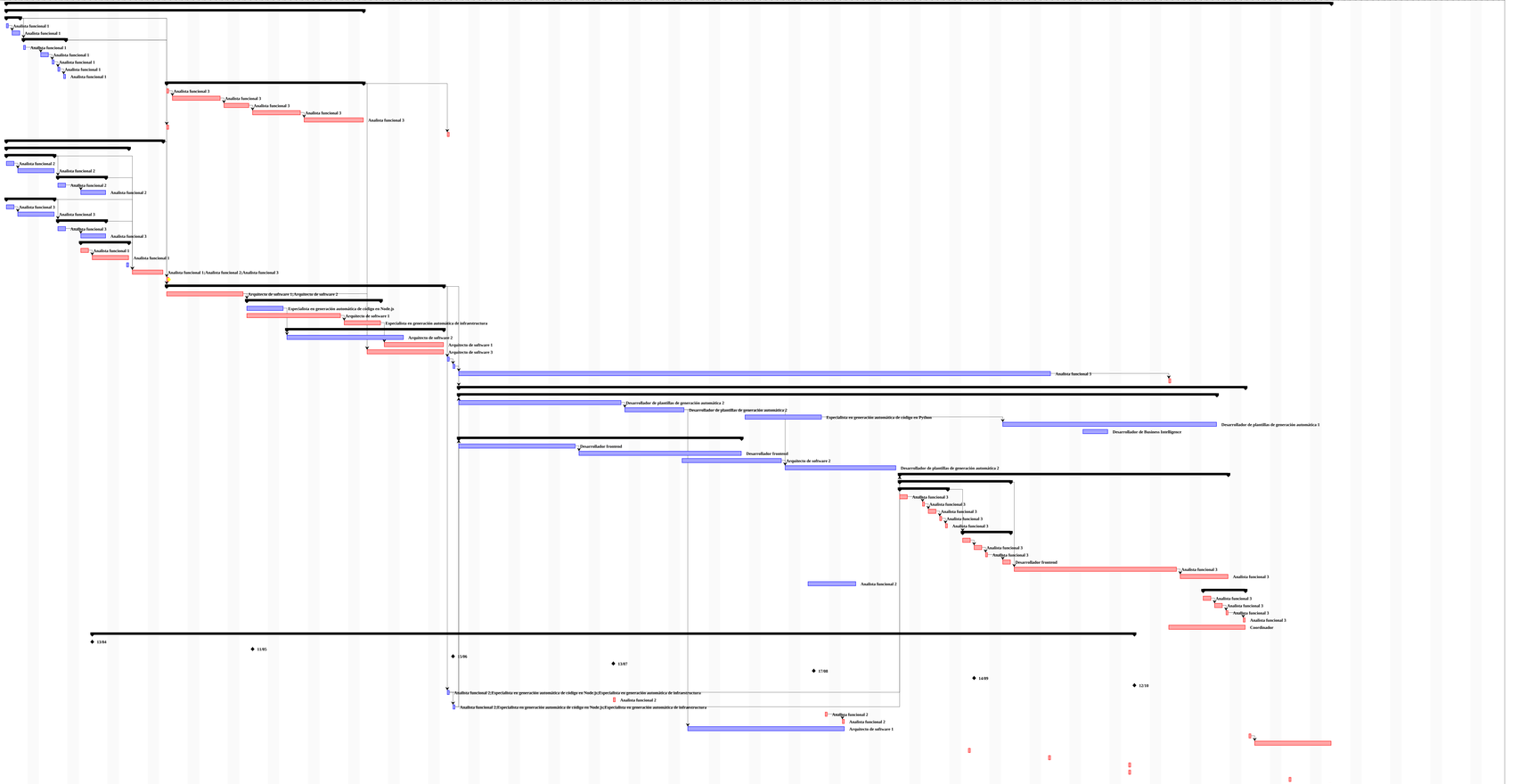
[17] NodeJS: https://nodejs.org/es/about/

[18] Python: https://www.python.org/

Anexos

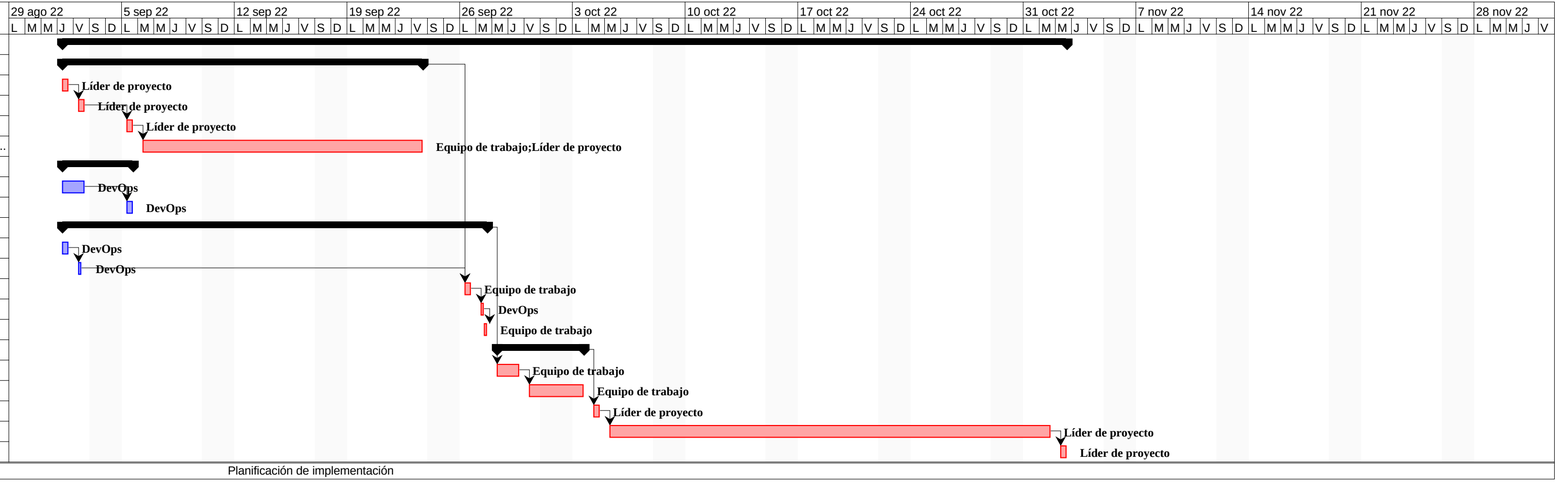
Anexo 1 – Diagrama de tiempos

		Nombre	Duracion	Inicio	Terminado	Predece...	Nombres del Recurso
1		**Accelerate - Proyecto Final 2022**	**166 days**	**29/03/22 08:00**	**15/11/22 17:00**		
2		**Planificación**	**45 days**	**29/03/22 08:00**	**30/05/22 17:00**		
3		**Actividades**	**3 days**	**29/03/22 08:00**	**31/03/22 17:00**		
4		Definición y descripción de actividades	1 day	29/03/22 08:00	29/03/22 17:00		Analista funcional 1
5		Creación del diagrama de tiempos	2 days	30/03/22 08:00	31/03/22 17:00	4	Analista funcional 1
6		**Organización para la ejecución del proyecto**	**6 days**	**01/04/22 08:00**	**08/04/22 17:00**	**3**	
7		Definición del equipo de trabajo	1 day	01/04/22 08:00	01/04/22 17:00		Analista funcional 1
8		Descripción de las funciones principales de los miembros del equipo	2 days	04/04/22 08:00	05/04/22 17:00	7	Analista funcional 1
9		Asignación de tareas a cada miembro del equipo	1 day	06/04/22 08:00	06/04/22 17:00	8	Analista funcional 1
10		Especificación de métodos de comunicación formal, control de avance y retroalimentación	1 day	07/04/22 08:00	07/04/22 17:00	9	Analista funcional 1
11		Especificación de la Gestión de Configuración del Software	1 day	08/04/22 08:00	08/04/22 17:00	10	Analista funcional 1
12		**Estudio de factibilidad**	**25 days**	**26/04/22 08:00**	**30/05/22 17:00**		
13		Realización del diagrama de recursos	1 day	26/04/22 08:00	26/04/22 17:00		Analista funcional 3
14		Realización del análisis de factibilidad	7 days	27/04/22 08:00	05/05/22 17:00	13	Analista funcional 3
15		Definición de costos desagregados por recursos con periodicidad mensual	3 days	06/05/22 08:00	10/05/22 17:00	14	Analista funcional 3
16		Realización del análisis de riesgos	7 days	11/05/22 08:00	19/05/22 17:00	15	Analista funcional 3
17		Realización del análisis de impacto ambiental	7 days	20/05/22 08:00	30/05/22 17:00	16	Analista funcional 3
18		Entrega de etapa de Planificación: Capítulo I y II	1 day	26/04/22 08:00	26/04/22 17:00	3;6	
19		Entrega de etapa de Planificación: Capítulo III	1 day	14/06/22 08:00	14/06/22 17:00	12	
20		**Definición de requerimientos**	**20 days**	**29/03/22 08:00**	**25/04/22 17:00**		
21		**Relevamiento y análisis de herramientas similares**	**16 days**	**29/03/22 08:00**	**19/04/22 17:00**		
22		**Relevamiento de herramienta Amplication**	**7 days**	**29/03/22 08:00**	**06/04/22 17:00**		
23		Relevamiento general de Amplication	2 days	29/03/22 08:00	30/03/22 17:00		Analista funcional 2
24		Relevamiento detallado de Amplication	5 days	31/03/22 08:00	06/04/22 17:00	23	Analista funcional 2
25		**Relevamiento de herramienta Ventó**	**7 days**	**07/04/22 08:00**	**15/04/22 17:00**	**22**	
26		Relevamiento general de Ventó	2 days	07/04/22 08:00	08/04/22 17:00		Analista funcional 2
27		Relevamiento detallado de Ventó	5 days	11/04/22 08:00	15/04/22 17:00	26	Analista funcional 2
28		**Relevamiento de herramienta Jhipster**	**7 days**	**29/03/22 08:00**	**06/04/22 17:00**		
29		Relevamiento general de Jhipster	2 days	29/03/22 08:00	30/03/22 17:00		Analista funcional 3
30		Relevamiento detallado de Jhipster	5 days	31/03/22 08:00	06/04/22 17:00	29	Analista funcional 3
31		**Relevamiento de herramienta D**	**7 days**	**07/04/22 08:00**	**15/04/22 17:00**	**28**	
32		Relevamiento general de D	2 days	07/04/22 08:00	08/04/22 17:00		Analista funcional 3
33		Relevamiento detallado de D	5 days	11/04/22 08:00	15/04/22 17:00	32	Analista funcional 3
34		**Relevamiento de herramienta E**	**7 days**	**11/04/22 08:00**	**19/04/22 17:00**		
35		Relevamiento general de E	2 days	11/04/22 08:00	12/04/22 17:00		Analista funcional 1
36		Relevamiento detallado de E	5 days	13/04/22 08:00	19/04/22 17:00	35	Analista funcional 1
37		Trabajo práctico integrador "Dirección de Proyectos de Sistemas"	1 day	19/04/22 08:00	19/04/22 17:00		
38		Definición de objetivos y alcances preliminares del sistema	4 days	20/04/22 08:00	25/04/22 17:00	25;28;...	Analista funcional 1;Analista funcional 2;Analista funcional 3
39		Entrega de etapa de definición de requerimientos	1 day	26/04/22 08:00	26/04/22 17:00	20	
40		**Diseño**	**35 days**	**26/04/22 08:00**	**13/06/22 17:00**	**3;6;20**	
41		Definición del sistema y creación de historias de usuario	10 days	26/04/22 08:00	09/05/22 17:00		Arquitecto de software 1;Arquitecto de software 2
42		**Creación de ejemplos**	**18 days**	**10/05/22 08:00**	**02/06/22 17:00**	**41**	
43		Creación de ejemplos base para la generación automática de código en Node.js	5 days	10/05/22 08:00	16/05/22 17:00		Especialista en generación automática de código en Node.js
44		Investigación sobre mecanismos de autenticación y autorización	13 days	10/05/22 08:00	26/05/22 17:00		Arquitecto de software 1
45		Creación de ejemplos base para la generación automática de infraestructura	5 days	27/05/22 08:00	02/06/22 17:00	44	Especialista en generación automática de infraestructura
46		**Creación de metamodelos**	**20 days**	**17/05/22 08:00**	**13/06/22 17:00**		
47		Creación de metamodelo para la generación de código	15 days	17/05/22 08:00	06/06/22 17:00	43	Arquitecto de software 2
48		Creación de metamodelo para la generación de microservicios e infraestructura	7 days	03/06/22 08:00	13/06/22 17:00	45	Arquitecto de software 1
49		Definición de pantallas del sistema	10 days	31/05/22 08:00	13/06/22 17:00	12;41	Arquitecto de software 3
50		Entrega de etapa de diseño	1 day	14/06/22 08:00	14/06/22 17:00	40	
51		Inicio de diseño de papers para Congreso CONAISI	1 day	15/06/22 08:00	15/06/22 17:00	50	
52		Elaboración paper CoNaISI	74 days	16/06/22 08:00	27/09/22 17:00	51	Analista funcional 3
53		Entrega final paper CoNaISI	1 day	18/10/22 08:00	18/10/22 17:00	52	
54		**Desarrollo e implementación**	**98 days**	**16/06/22 08:00**	**31/10/22 17:00**	**43**	
55		**Creación de plantillas de Acceleo**	**95 days**	**16/06/22 08:00**	**24/10/22 17:00**	**96;98**	
56		Creación de plantilla de Acceleo para la generación de código en Node.js	21 days	16/06/22 08:00	14/07/22 17:00		Desarrollador de plantillas de generación automática 2
57		Creación de plantilla de Acceleo para la generación de infraestructura	7 days	15/07/22 08:00	25/07/22 17:00	56	Desarrollador de plantillas de generación automática 2
58		Creación de ejemplos base para la generación automática de código en Python	10 days	05/08/22 08:00	18/08/22 17:00		Especialista en generación automática de código en Python
59		Creación de plantilla de Acceleo para la generación de código en Python	28 days	19/09/22 08:00	26/10/22 17:00	58	Desarrollador de plantillas de generación automática 1
60		Creación del módulo de reporting	5 days	03/10/22 08:00	07/10/22 17:00		Desarrollador de Business Intelligence
61		**Creación de visualización en Sirius**	**36 days**	**16/08/22 08:00**	**04/10/22 17:00**	**96;98**	
62		Creación de visualización en Sirius para la generación de código	15 days	16/08/22 08:00	06/09/22 17:00		Desarrollador frontend
63		Creación de visualización en Sirius para la generación de infraestructura	21 days	07/07/22 08:00	04/08/22 17:00	62	Desarrollador frontend
64		Investigación para la creación de un plugin de Eclipse	14 days	15/07/22 08:00	11/08/22 17:00		Arquitecto de software 2
65		Creación del plugin	14 days	12/08/22 08:00	31/08/22 17:00	57;64	Desarrollador de plantillas de generación automática 2
66		**Capacitación e implementación**	**42 days**	**01/09/22 08:00**	**28/10/22 17:00**	**96;98**	
67		**Planificación de capacitación**	**14 days**	**01/09/22 08:00**	**20/09/22 17:00**		
68		**Plan de capacitación - Usuarios Finales**	**7 days**	**01/09/22 08:00**	**09/09/22 17:00**		
69		Elaboración del plan de capacitación para usuarios finales	2 days	01/09/22 08:00	02/09/22 17:00		Analista funcional 3
70		Elaboración de la sección "Getting Started"	1 day	05/09/22 08:00	05/09/22 17:00	69	Analista funcional 3
71		Elaboración de la sección "Elements"	2 days	06/09/22 08:00	07/09/22 17:00	70	Analista funcional 3
72		Elaboración de la sección "FAQs"	1 day	08/09/22 08:00	08/09/22 17:00	71	Analista funcional 3
73		Elaboración de la sección "Working Methodology"	1 day	09/09/22 08:00	09/09/22 17:00	72	Analista funcional 3
74		**Plan de capacitación - Administradores**	**7 days**	**12/09/22 08:00**	**20/09/22 17:00**	**68**	
75		Elaboración del plan de capacitación para administradores	2 days	12/09/22 08:00	13/09/22 17:00		
76		Elaboración de la sección de instalación del sistema	2 days	14/09/22 08:00	15/09/22 17:00	75	Analista funcional 3
77		Elaboración de la sección "Reporting"	1 day	16/09/22 08:00	16/09/22 17:00	76	Analista funcional 3
78		Elaboración página web y despliegue en internet	2 days	19/09/22 08:00	20/09/22 17:00	77	Desarrollador frontend
79		Creación de un manual de usuarios	21 days	21/09/22 08:00	19/10/22 17:00	67	Analista funcional 3
80		Planificación de implementación del sistema	7 days	20/10/22 08:00	28/10/22 17:00	79	Analista funcional 3
81		Trabajo práctico integrador "Cierre/cambio de Sistemas"	7 days	16/08/22 08:00	24/08/22 17:00		Analista funcional 2
82		**Planificación, ejecución y documentación de pruebas**	**6 days**	**24/10/22 08:00**	**31/10/22 17:00**		
83		Pruebas de aceptación	2 days	24/10/22 08:00	25/10/22 17:00		Analista funcional 3
84		Pruebas de validación de ingreso de datos	2 days	26/10/22 08:00	27/10/22 17:00	83	Analista funcional 3
85		Pruebas de generación automática	1 day	28/10/22 08:00	28/10/22 17:00	84	Analista funcional 3
86		Pruebas de carga	1 day	31/10/22 08:00	31/10/22 17:00	85	Analista funcional 3
87		Correcciones del sistema	10 days	18/10/22 08:00	31/10/22 17:00		Coordinador
88		**Medidas preventivas**	**130 days**	**13/04/22 08:00**	**12/10/22 08:00**		
89		Tratamiento de riesgos 1: Reunión mensual para seguimiento de cambios - Abril.	0 days	13/04/22 08:00	13/04/22 08:00		Analista funcional 3;Coordinador
90		Tratamiento de riesgos 1: Reunión mensual para seguimiento de cambios - Mayo.	0 days	11/05/22 08:00	11/05/22 08:00		Analista funcional 3;Coordinador
91		Tratamiento de riesgos 1: Reunión mensual para seguimiento de cambios - Junio.	0 days	15/06/22 08:00	15/06/22 08:00		Analista funcional 3;Coordinador
92		Tratamiento de riesgos 1: Reunión mensual para seguimiento de cambios - Julio.	0 days	13/07/22 08:00	13/07/22 08:00		Analista funcional 3;Coordinador
93		Tratamiento de riesgos 1: Reunión mensual para seguimiento de cambios - Agosto.	0 days	17/08/22 08:00	17/08/22 08:00		Analista funcional 3;Coordinador
94		Tratamiento de riesgos 1: Reunión mensual para seguimiento de cambios - Septiembre.	0 days	14/09/22 08:00	14/09/22 08:00		Analista funcional 3;Coordinador
95		Tratamiento de riesgos 1: Reunión mensual para seguimiento de cambios - Octubre.	0 days	12/10/22 08:00	12/10/22 08:00		Analista funcional 3;Coordinador
96		Tratamiento de riesgos 2: Revisión de tiempos con un experto.	1 day	14/06/22 08:00	14/06/22 17:00	40	Analista funcional 2;Especialista en generación automática de código en Node.js;...
97		Tratamiento de riesgos 3: Revisión de recursos a utilizar en el proyecto.	1 day	13/07/22 08:00	13/07/22 17:00		Analista funcional 2
98		Tratamiento de riesgos 4: Revisión de costos con un experto.	1 day	15/06/22 08:00	15/06/22 17:00	96	Analista funcional 2;Especialista en generación automática de código en Node.js;...
99		Tratamiento de riesgos 5: Revision del uso de patrones, normas y estándares.	1 day	19/08/22 08:00	19/08/22 17:00		Analista funcional 2
100		Tratamiento de riesgos 6: Revisión del relevamiento del negocio y sistemas similares.	1 day	22/08/22 08:00	22/08/22 17:00	99	Analista funcional 2
101		Tratamiento de riesgos 7: Investigación de Sirius Web.	20 days	26/07/22 08:00	22/08/22 17:00	57	Arquitecto de software 1
102		Entrega de etapa de desarrollo e implementacion	1 day	01/11/22 08:00	01/11/22 17:00		
103		Ajuste de detalles para presentación final	10 days	02/11/22 08:00	15/11/22 17:00	102	
104		Primera demo del sistema	1 day	13/09/22 08:00	13/09/22 17:00		
105		Primera revisión de poster	1 day	27/09/22 08:00	27/09/22 17:00		
106		Segunda revisión de poster	1 day	11/10/22 08:00	11/10/22 17:00		
107		Segunda demo del sistema	1 day	11/10/22 08:00	11/10/22 17:00		
108		Tercera demo del sistema y ensayo de exposición	1 day	08/11/22 08:00	08/11/22 17:00		
109		Exposición Anual de Proyectos de Sistemas	1 day	15/11/22 08:00	15/11/22 17:00		



Anexo 2 – Diagrama de tiempos de planificación de implementación

	Nombre	Duracion	Predecesores	Nombres del Recurso
1	**Planificacion de implementacion**	**45 days**		
2	**Definicion del equipo de trabajo**	**17 days**		
3	Formacion del equipo	1 day		Líder de proyecto
4	Distribucion de roles	1 day	3	Líder de proyecto
5	Distribucion de tareas	1 day	4	Líder de proyecto
6	Capacitacion	14 days	5	Equipo de trabajo;Líder de p...
7	**Implementacion del modulo de reporting**	**3 days**		
8	Creacion de una base de datos en la nube	2 days		DevOps
9	Despliegue del modulo de reporting en la nube	1 day	8	DevOps
10	**Implementacion del modulo de modelado y generacion automatica**	**19 days**		
11	Aseguramiento de requisitos minimos	1 day		DevOps
12	Instalacion del modulo	0,5 days	11	DevOps
13	Creacion del proyecto de modelado	1 day	2;12	Equipo de trabajo
14	Backup del proyecto creado	0,5 days	13	DevOps
15	Aplicacion de generacion automatica al proyecto creado	0,5 days	14	Equipo de trabajo
16	**Integracion de codigo e infraestructura**	**4 days**		
17	Integracion de los microservicios generados con aquellos desarrollados manualmente	2 days	10	Equipo de trabajo
18	Integracion de la infraestructura generada con la infraestructura creada por el DevOps	2 days	17	Equipo de trabajo
19	Evaluacion de resultados	1 day	16	Líder de proyecto
20	Planteamiento de la siguiente etapa	20 days	19	Líder de proyecto
21	Implementacion de Accelerate en todos los proyectos	1 day	20	Líder de proyecto



Planificación de implementación

Anexo 3 – Manual de usuario

Contenido

Figuras

1. Sección General

1.1 Introducción

1.1.1 Breve descripción de Accelerate

Accelerate es una herramienta que busca generar archivos de código y de configuración de infraestructura a través de una herramienta de modelado. La misma, está basada en modelos integrales que unifican el diseño de una arquitectura de microservicios en la nube con el diseño orientado a objetos de cada uno de ellos, y utiliza múltiples parámetros de generación automática tales como el lenguaje de cada servicio.

El uso de Accelerate tiene los siguientes beneficios:

- Elimina tareas de codificación repetitivas.
- Genera el código de microservicios en múltiples frameworks mediante estándares de la industria y buenas prácticas de desarrollo.
- Integra automáticamente la arquitectura distribuida modelada.
- Permite dedicarle mayor tiempo al desarrollo de la lógica de negocio.
- Genera la configuración de aprovisionamiento en la nube para la fácil puesta en marcha del sistema planteado.
- Brinda una útil herramienta de modelado a los diseñadores del sistema.
- Documenta el proyecto mediante modelos y diagramas.

1.1.2 Ingeniería dirigida por modelos

La Ingeniería Dirigida por Modelos o MDE (Model Driven Engineering) es un paradigma de ingeniería de software que se centra en la creación y explotación de modelos de dominio (personalizados para una aplicación particular).

Accelerate está basado en este paradigma, ya que utiliza un modelo de dominio que se utiliza como entrada para la generación automática de código e infraestructura.

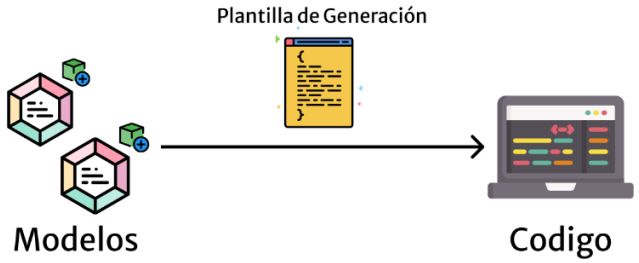


Figura 1. Proceso de generación automática de código

La generación automática utiliza plantillas de generación predefinidas, que utilizan como entrada el modelo diseñado a través de una herramienta gráfica y generan la salida en forma de archivos de código e infraestructura relevantes para el sistema modelado.

1.1.3 Arquitecturas de microservicios y su implementación con Kubernetes

Una arquitectura de microservicios consta de una colección de servicios autónomos independientes que se comunican entre sí. Normalmente, estos microservicios se empaquetan en un contenedor y se agrupan en ambientes de microservicios que pueden estar desplegados en un proveedor en la nube como Google Cloud.

Usualmente, el despliegue de una arquitectura de microservicios se realiza utilizando Kubernetes, una tecnología que permite automatizar la implementación, el escalado y la administración de contenedores mediante archivos de configuración denominados manifiestos.

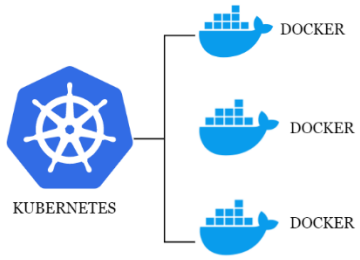


Figura 2. Orquestación de microservicios empaquetados en contenedores Docker.

Los proveedores en la nube se han adaptado a la creciente tendencia de Kubernetes y la mayoría de ellos provee una implementación de esta tecnología, permitiendo el despliegue de un clúster Kubernetes en la nube. Por ejemplo, Google Cloud provee el producto GKE (Google Kubernetes Engine).

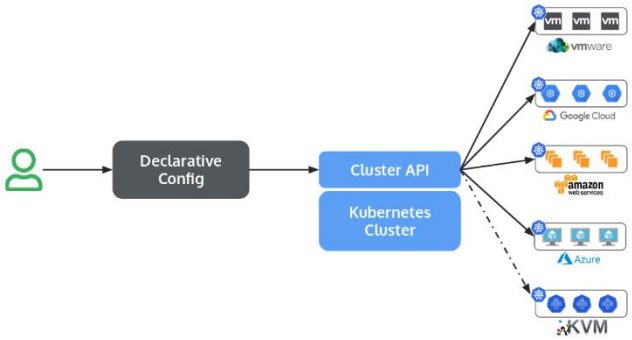


Figura 3. Despliegue de clústeres en diferentes proveedores en la nube a través de una configuración declarativa, propia de los manifiestos de Kubernetes.

De esta manera, cada ambiente de microservicios se convierte en un clúster de Kubernetes que luego se despliega en la nube.

1.1.4 Infraestructura como Código

En muchas oportunidades, la gestión de una arquitectura de microservicios en la nube se vuelve muy compleja, ya sea por las características propias de ella o por su tamaño creciente. Debido a ello, migrar una arquitectura de un proveedor a otro, es una tarea muy complicada si no se cuenta con las herramientas adecuadas.

Para solucionar este problema, aparece el paradigma IaC (Infrastructure as Code), que permite administrar el despliegue en la nube a través de archivos de configuración utilizando herramientas para tal fin como Terraform.

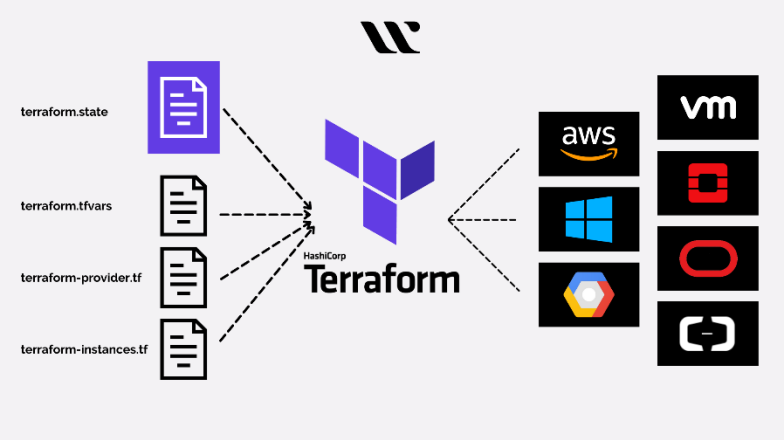


Figura 4. Despliegue en la nube a través de archivos de configuración con Terraform.

De esta manera, con los archivos de configuración, se puede realizar el aprovisionamiento de recursos en la nube de manera automática utilizando la Interfaz de Línea de Comandos.

2. Sección para el administrador

2.1 Implementación de Accelerate

2.1.1 Instalación del sistema

La instalación de Accelerate requiere como mínimo una computadora con acceso a Internet y con el sistema operativo Windows. A continuación, se enumeran los pasos necesarios para instalar Accelerate:

1. Ingresar a la página www.accelerate.com.ar y descargar la herramienta. Para ello, es necesario aceptar los términos y condiciones respecto a la recopilación de métricas de modelado.
2. Crear una carpeta y descomprimir el archivo descargado.
3. Realizar doble click sobre el archivo Accelerate.exe.
4. Elegir la carpeta workspace como espacio de trabajo y marcar la opción "*Use this as the default and do not ask again*".
5. Seleccionar el botón "*Launch*".

Una vez finalizados los pasos enumerados, se abrirá la interfaz de usuario de Accelerate y cada vez que se desee utilizar nuevamente, solamente será necesario realizar doble click sobre el archivo Accelerate.exe.

2.1.2 Backup de modelos

Cada modelo realizado se guarda en una instancia XMI, es decir, en un archivo con extensión .xmi. Debido a ello, con el simple hecho de guardar el archivo mencionado, el modelo estaría resguardado. A continuación, se describe el procedimiento para realizar backup con guardado en Google Drive:

1. Realizar clic derecho en el proyecto que contiene la instancia y seleccionar la opción "*Show in Explorer*".

2. Seleccionar el archivo de modelado con extensión .xmi.
3. Subir el archivo a una carpeta de Google Drive.

Cabe aclarar que este procedimiento no funciona solamente para realizar backup utilizando Google Drive ya que, si se desea utilizar otra herramienta, simplemente es necesario modificar el tercer paso.

Este procedimiento (al igual que la recuperación de modelos) es de respaldo total, es decir, almacena completamente el modelo XMI y luego lo recupera también en su totalidad.

El usuario administrador puede acceder a este procedimiento durante la ejecución de Accelerate a través del botón Backup/Restore. A continuación, se muestra la salida al arrastrar el botón a lienzo en blanco.

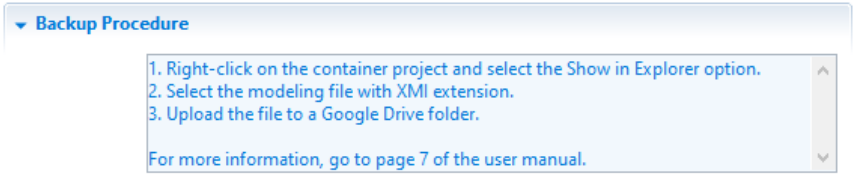


Figura 5. Procedimiento de backup a través del botón Backup/Restore.

2.1.3 Recuperación de modelos

A continuación, se describen los pasos para realizar la recuperación de modelos utilizando Google Drive:

1. Descargar el archivo de modelado con extensión .xmi de la carpeta de backup.
2. Realizar clic derecho en el proyecto de modelado que actuará de contenedor de la instancia y seleccionar la opción "*Show in Explorer*".
3. Mover el archivo descargado a la carpeta abierta en el paso anterior.
4. Realizar clic derecho en el proyecto de modelado y seleccionar la opción "*Refresh*".

El resultado de aplicar el procedimiento es un proyecto de modelado con su modelo en forma de instancia XMI, por lo que sólo es necesario crear una nueva representación a partir del mismo [página 19, paso 7].

Como se mencionó en la sección 2.1.2 “Backup de modelos”, la recuperación del modelo es total.

El usuario administrador puede acceder a este procedimiento durante la ejecución de Accelerate a través del botón Backup/Restore. A continuación, se muestra la salida al arrastrar el botón a lienzo en blanco.

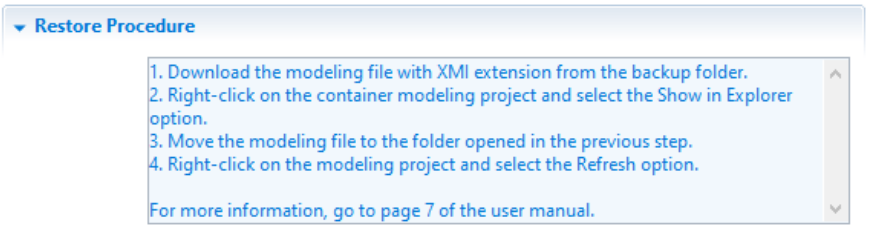


Figura 6. Procedimiento de recuperación a través del botón Backup/Restore.

2.2 Visualización de reportes

Cada vez que un usuario realiza la generación de código e infraestructura, la herramienta envía métricas de modelado de manera automática a una base de datos. La misma, permita la realización de consultas a través de HTTP y se utiliza en conjunto con Grafana, un software libre que permite la visualización de indicadores útiles para el negocio.

2.2.1 Instalación de Grafana

Para utilizar Grafana, es necesario descargar su instalador desde https://grafana.com/grafana/download y ejecutarlo para su instalación.

Una vez instalado, Grafana actúa como un servicio que se ejecuta en el puerto 3000 de nuestro host local, permitiendo su acceso desde un navegador con la dirección http://localhost:3000.

2.2.2 Instalación de Infinity para la consulta de datos

Al ingresar a Grafana, es necesario instalar el plugin Infinity para la consulta de los datos. Para ello, se debe ingresar al apartado *Configuration* y colocar "*Infinity*" en el buscador de plugins. A continuación, se observa la siguiente pantalla:

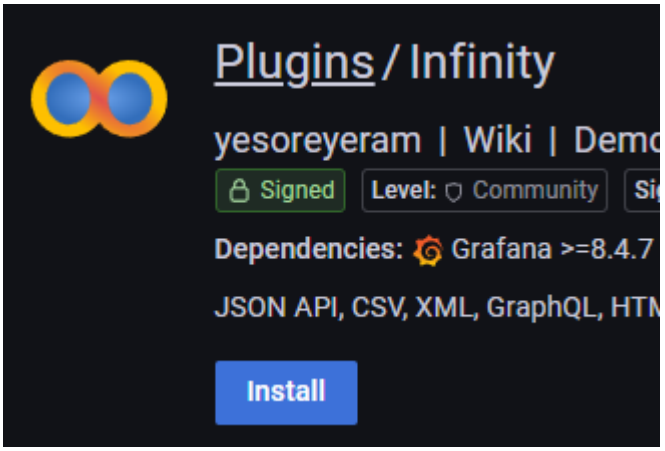


Figura 7. Instalación del plugin Infinity.

Luego, al presionar el botón Install se observa una pantalla que confirma la instalación de Infinity.

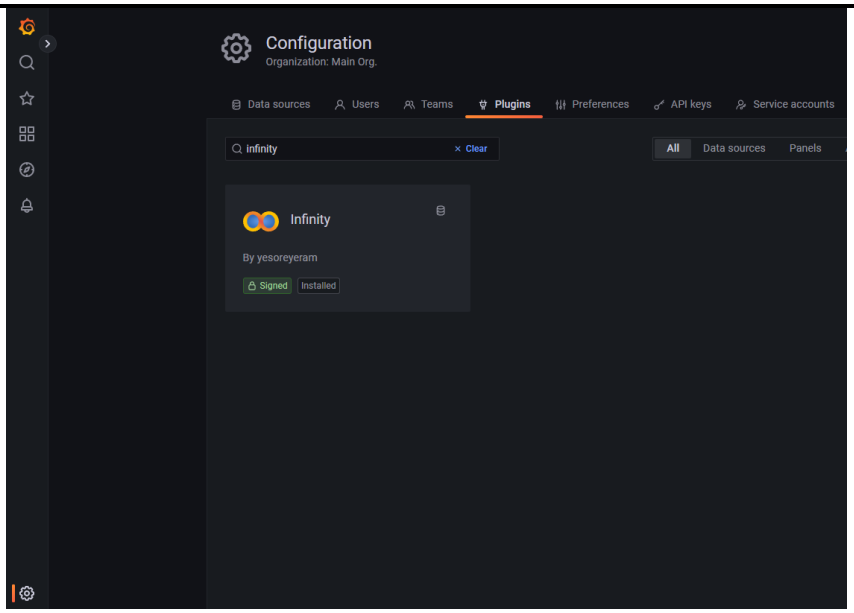


Figura 8. Confirmación de instalación de Infinity

2.2.3 Importación del dashboard

Grafana permite exportar sus dashboards en un formato JSON para luego importarlos a través de un menú gráfico. Ya que el dashboard de Accelerate está en formato JSON, únicamente es necesario importarlo desde el menú *Dashboard* con la opción *Import, que permite la carga de este desde el explorador de archivos.* Luego, se debe cargar el archivo JSON y configurar sus opciones como se muestran a continuación y presionar el botón Import.

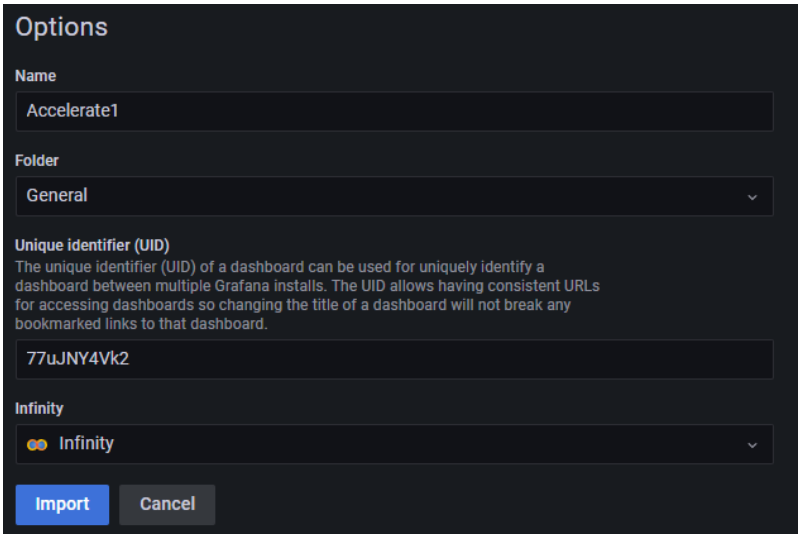


Figura 9. Opciones de importación del dashboard Accelerate.

2.2.4 Uso del dashboard

Una vez cargado el dashboard de Accelerate, se puede acceder al mismo a través de la opción *Browse* ubicada dentro del menú *Dashboard*, que muestra en pantalla todos los dashboards cargados en la herramienta y permite abrirlos con un simple clic sobre su nombre.

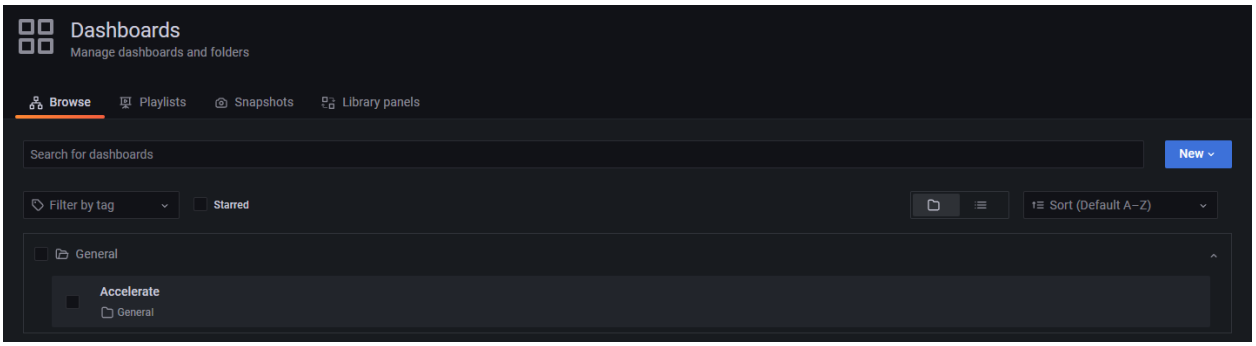


Figura 10. Búsqueda del dashboard Accelerate.

Al realizar clic sobre el dashboard Accelerate, se abre una vista como la que se muestra a continuación:

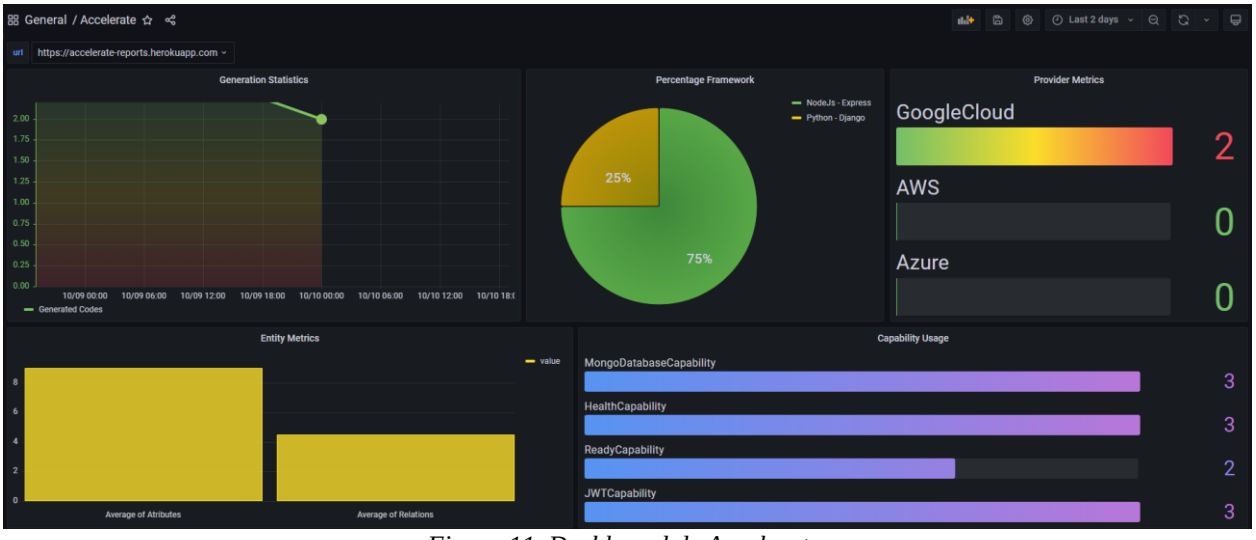


Figura 11. Dashboard de Accelerate.

Este dashboard muestra 5 indicadores clave para el negocio:

- **Generation Statistics:** Gráfico de tendencia que muestra la cantidad de generaciones automáticas en el tiempo.
- **Percentage Framework:** Porcentaje de generaciones automáticas respecto a cada lenguaje y framework de desarrollo.
- **Provider Metrics:** Cantidad de clústeres generados automáticamente para cada proveedor.
- **Entity Metrics:** Promedio de atributos y de relaciones por cada entidad modelada.
- **Capability Usage:** Comparación en cantidad de las diferentes capacidades utilizadas para la generación automática.

Además, el dashboard se refrezca de manera dinámica mediante la parametrización de rangos de fechas. Al realizar clic sobre el botón  ubicado en la esquina superior derecha, se despliega un menú con rangos de fechas seleccionables. Al seleccionar un rango de fecha, se observa cómo el dashboard se refresca en su totalidad, mostrando únicamente la información pertinente al rango elegido.

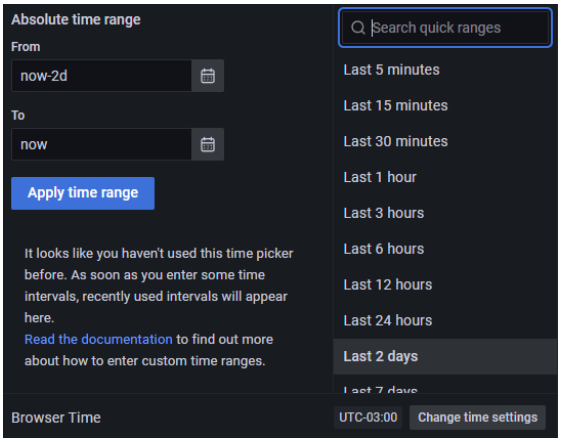


Figura 12. Rango de fechas seleccionables.

3. Sección para el modelador

3.1 Getting Started

Para comenzar a utilizar la herramienta, se recomienda ver los siguientes videos descriptivos:

1. **Uso de la herramienta de modelado**: https://youtu.be/qNvLHrZda3A
2. **Resultados de la generación automática**: https://youtu.be/icHCFJjnhvE

3.2 Componentes de Accelerate

3.2.1 Metamodelos e instancias XMI

El núcleo de Accelerate está conformado por tres metamodelos, AccelerateMLC, AccelerateMLS y AccelerateMLI. Cada uno de estos, contiene la estructura necesaria para almacenar la información del modelo en su nivel correspondiente (e.g. el metamodelo AccelerateMLS dirige el metamodelado únicamente respecto a la arquitectura de microservicios) y están conectados en forma de cascada, siendo el metamodelo AccelerateMLI la cabeza de la estructura.

Esta interacción entre los metamodelos es invisible a los usuarios gracias a las instancias XMI, que permiten crear un objeto de los tres metamodelos. Esta instancia tiene el mismo significado que un objeto UML, ya que en ella se almacena la información que el usuario modela a través de la herramienta gráfica. Debido a ello, es fundamental conservar esta instancia XMI como un archivo importante del proyecto, ya que la herramienta gráfica basa su representación en la información que guarda la misma.

3.2.2 Herramienta gráfica

Accelerate cuenta con una interfaz gráfica que permite modelar una arquitectura de microservicios y su correspondiente implementación interna a través del Paradigma Orientado a Objetos.

Esta interfaz gráfica cuenta con 4 componentes:

- **Paleta de elementos:** Contiene los elementos que pueden ser creados gráficamente.
- **Lienzo:** Permite la creación de elementos mediante un mecanismo de arrastre de elementos desde la paleta hacia el lienzo.
- **Vista de propiedades:** Permite configurar las propiedades de un elemento a través de campos de entrada de texto.
- **Explorador de proyectos:** Permite visualizar los proyectos creados y navegar entre diferentes modelos.

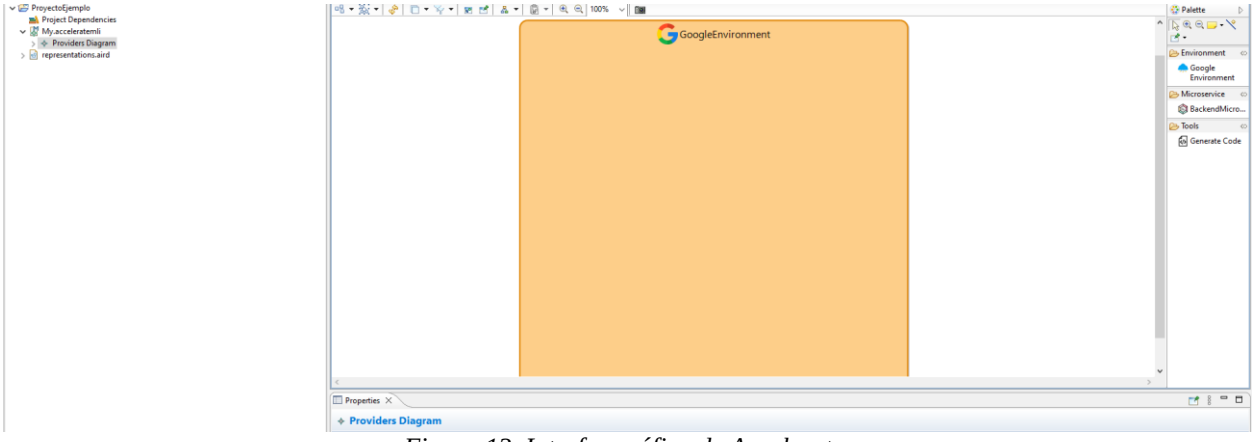


Figura 13. Interfaz gráfica de Accelerate.

Cada proyecto se crea en el explorador de la siguiente manera:

1. Seleccionar la opción File > New > Other > Modeling Project.
2. Colocar el nombre del proyecto.

3. Seleccionar la opción "*Finish"*.

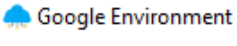
Además, cada proyecto puede alojar múltiples modelos. Los mismos, se crean de la siguiente manera:

1. Realizar clic derecho sobre el proyecto de modelado y elegir la opción New > Other > AccelerateMLI Model.
2. Colocar el nombre del modelo.
3. Seleccionar la opción *"Next"*.
4. Seleccionar el elemento ProvidersDiagram como *"Model Object"*.
5. Presionar la opción *"Finish"*.
6. Desplegar la instancia XMI creada.
7. Crear una nueva representación a partir de la instancia. Para ello, realizar clic derecho sobre el elemento ProvidersDiagram y elegir la opción New Representation > Other.
8. Elegir el elemento *"Microservices Diagram"*.

3.2.2.1 Representación de Microservicios

La representación de microservicios es una de las dos vistas de la herramienta gráfica, que permite el modelado de ambientes de microservicios y su configuración de despliegue para el aprovisionamiento automático en la nube.

3.2.2.1.1 Creación de un elemento Google Environment

Un elemento Google Environment representa un ambiente de microservicios alojado en Google Cloud a través de GKE (Google Kubernetes Engine) y se crea arrastrando el elemento 

desde la paleta hacia el lienzo.

Figura 14. Google Environment.

Una vez creado, se pueden configurar las siguientes propiedades al seleccionarlo:

- **Secret Id:** Usuario del registro de contenedores utilizado por los microservicios contenidos. Este valor es utilizado tanto por la integración continua como por los manifiestos de Kubernetes para la gestión de las imágenes de contenedores.
- **Url:** Registro de contenedores a utilizar. Valor por defecto: Docker.io.
- **Name:** Nombre del ambiente de Google. No pueden existir dos ambientes con el mismo nombre y este debe ser distinto de nulo.

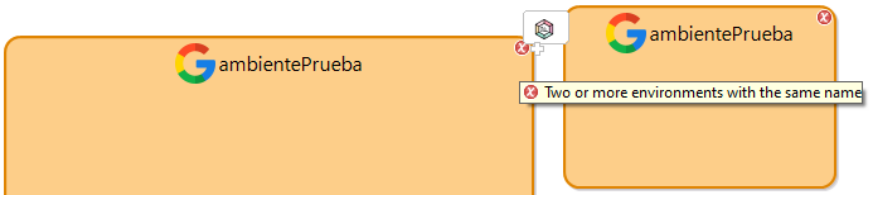


Figura 15. Mensaje de error para cuando existen dos ambientes con el mismo nombre.

Figura 16. Mensaje de error cuando el nombre de un ambiente es nulo.

- **Project Id:** Id del Proyecto de Google utilizado para el aprovisionamiento automático. No pueden existir dos ambientes con el mismo projectId y este debe ser distinto de nulo.

Figura 17. Mensaje de error cuando existe un ambiente con projectId nulo.

Figura 18. Mensaje de error cuando existen dos ambientes con el mismo projectId.

- **Region:** Región de despliegue del proyecto. Valor por defecto: southamerica-west-1.

Además, si se realiza doble clic sobre el elemento Google Environment, se observa el siguiente diálogo de configuraciones:

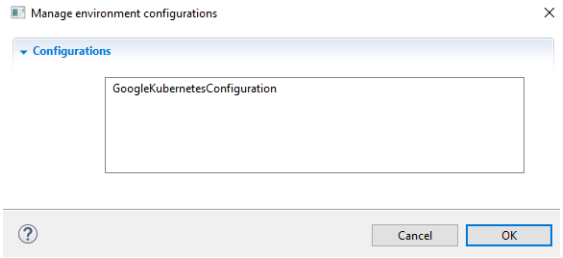


Figura 19. Configuración de un Google Environment.

Este diálogo permite al usuario editar internamente al ambiente de Google Cloud. En la primera versión de Accelerate, se puede parametrizar su configuración de Kubernetes.

3.2.2.1.1.1 Configuración de Kubernetes para un ambiente de Google

Cada ambiente de Google puede ser parametrizado respecto a su configuración de Kubernetes a través del elemento GoogleKubernetesConfiguration. Este elemento puede editarse al realizar doble clic sobre un elemento Google Environment y permite la siguiente configuración:

- **poolName:** Nombre del pool de nodos utilizado en el aprovisionamiento automático. Valor por defecto: defaultPool.
- **nodeMachineType:** Tipo de nodo utilizado por el pool. Valor por defecto: e2Medium.
- **nodeMinCount:** Cantidad mínima de nodos del pool. Valor por defecto: 1
- **nodeMaxCount:** Cantidad máxima de nodos del pool. Valor por defecto: 2. Este valor debe ser mayor a la cantidad mínima de nodos.

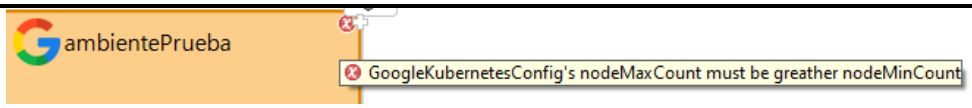


Figura 20. Mensaje de error cuando nodeMaxCount es menor a nodeMinCount.

- **nodeDiskSize:** Tamaño del disco de cada medido en GB. Valor por defecto: 10 GB. Este valor debe ser mayor o igual a 10.

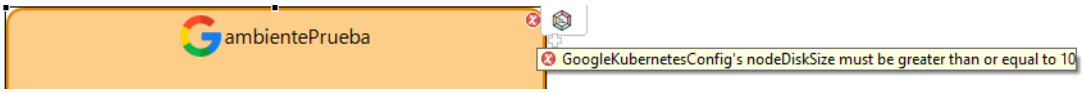


Figura 21. Mensaje de error cuando el tamaño de disco es menor a 10 GB.

3.2.2.1.2 Creación de un elemento BackendMicroservice

Un elemento BackendMicroservice representa un microservicio backend contenido en un ambiente y se crea arrastrando el elemento BackendMicroservice desde la paleta hacia dicho ambiente.

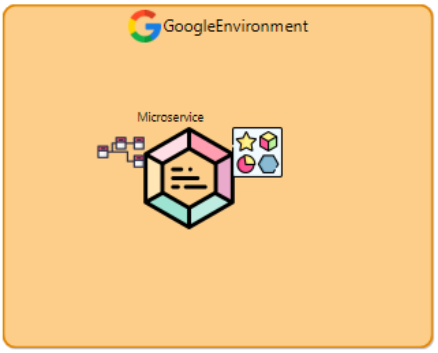


Figura 22. BackendMicroservice dentro de un GoogleEnvironment.

Una vez creado, se pueden configurar las siguientes propiedades al seleccionarlo:

- **name:** Nombre del microservicio. No pueden existir dos microservicios con el mismo nombre y este debe ser distinto de nulo.

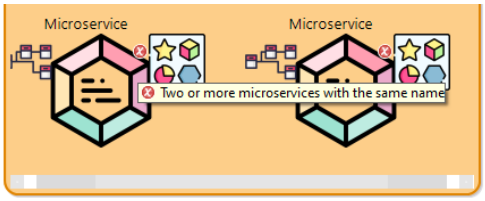


Figura 23. Mensaje de error cuando existen dos microservicios con el mismo nombre.

Figura 24. Mensaje de error cuando el nombre de un microservicio es nulo.

- **version:** Versión del microservicio.
- **description:** Descripción del microservicio.
- **port:** Puerto que expondrá el microservicio a través de una API. Valor por defecto: 8000.
- **framework:** Framework y lenguaje utilizado para la generación automática de código del microservicio.

Además, cada BackendMicroservice posee 3 aspectos que deben ser definidos: configuraciones, capacidades y modelado interno. Los aspectos mencionados, se describen a continuación.

3.2.2.1.2.1 Configuraciones de un BackendMicroservice

Cada BackendMicroservice necesita configuración propia de la infraestructura en la que será desplegado, por lo al hacer doble clic sobre el microservicio, se observa el siguiente diálogo de configuraciones.

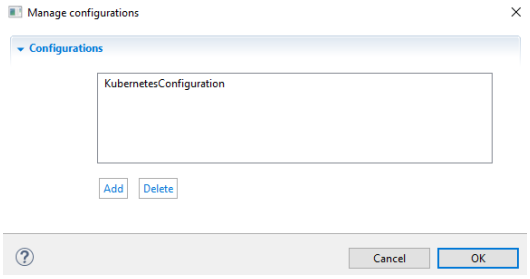


Figura 25. Configuración de un BackendMicroservice.

Este diálogo permite al usuario editar la configuración del microservicio. En la primera versión de Accelerate, se puede parametrizar su configuración de Kubernetes.

Al realizar doble clic sobre el elemento KubernetesConfiguration, se puede parametrizar lo siguiente:

- **replicas:** Cantidad de réplicas del microservicio que Kubernetes utilizará. Valor por defecto: 1. La cantidad de replicas no puede ser un valor nulo.

Figura 26. Mensaje de error cuando existe un microservicio con réplicas igual a null.

- **memoryRequest:** Cantidad mínima de memoria que se le brindará al microservicio. Valor por defecto: 256 MB.
- **cpuRequest:** Cantidad mínima de CPU que se le brindará al microservicio. Valor por defecto: 512 MHz.
- **memoryLimit:** Cantidad máxima de memoria que se le brindará al microservicio. Valor por defecto: 512 MB.
- **cpuLimit:** Cantidad máxima de CPU que se le brindará al microservicio. Valor por defecto: 1 GHz.

3.2.2.1.2.2 Capacidades de un BackendMicroservice

Cada BackendMicroservice podrá tener diferentes funcionalidades adicionales que denominaremos **capacidades**. Al hacer doble clic sobre el elemento  , se observa el siguiente diálogo de capacidades.

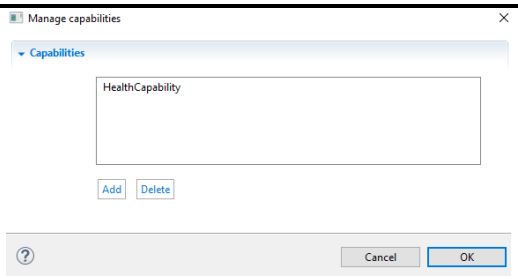


Figura 27. Capacidades de un BackendMicroservice.

Este diálogo muestra las capacidades actuales del microservicio (en este caso, posee la capacidad HealthCapability) y permite añadir y eliminar capacidades a través de los botones Add y Delete respectivamente.

Las capacidades están clasificadas por categorías y son las siguientes:

- **DatabaseCapability:** Permiten la conexión a una base de datos. Existen dos categorías de capacidades, SQL y NoSQL. Sin embargo, en la primera versión de Accelerate está implementada la capacidad NoSQL para realizar la conexión a MongoDB.
    - **MongoDatabaseCapability:** Permite la conexión a una base de datos MongoDB y requiere la siguiente configuración:
        - **username:** Usuario de la base de datos.
        - **password**: Contraseña de la base de datos.
        - **host**: Host donde se ubica la base de datos.
        - **databaseName**: Nombre de la base de datos.
- **AuthenticationCapability:** Permiten asegurar las rutas que expone el microservicio a través de un mecanismo de autenticación predefinido. En la primera versión de Accelerate se permite la autenticación de Accelerate a través de JWT (JSON Web Token).

- **JWTCapability:** Permite asegurar rutas a través del mecanismo JWT y requiere la siguiente configuración:
  - **secret:** Llave utilizada para el mecanismo JWT. Valor por defecto: secret.
  - **expiresIn:** Tiempo de expiración de la llave. Valor por defecto: 1800000.

- **RoutingCapability:** Permiten agregar funcionalidades relacionadas a sus rutas de acceso. Existen dos tipos de RoutingCapability:
  - **HealthCapability:** Permite agregar al microservicio la funcionalidad de chequeo de salud de manera automática, revisando el estado del microservicio a través de Kubernetes. Requiere la siguiente configuración:
    - **healthRoute:** Ruta que expondrá el microservicio para su chequeo de salud. Valor por defecto: health. El valor de esta ruta no puede comenzar con el carácter "/".

Figura 28. Mensaje de error cuando healthRoute comienza con /.

- **period:** Período de tiempo que indica cada cuántos segundos se revisará la salud del microservicio. Valor por defecto: 5 segundos. El período debe ser mayor a 0.

Figura 29. Mensaje de error cuando period no es mayor a 0.

- **initialDelay:** Período de tiempo que indica cuánto tiempo se esperará desde el arranque del microservicio hasta el comienzo de su chequeo periódico. Valor por defecto: 15 segundos.

- **ReadyCapability:** Permite agregar al microservicio la funcionalidad de chequear si el microservicio está listo u ocupado, revisando su estado a través de Kubernetes. Requiere la siguiente configuración:
  - **readyRoute:** Ruta que expondrá el microservicio para su chequeo. Valor por defecto: ready. El valor de esta ruta no puede comenzar con el carácter "/".

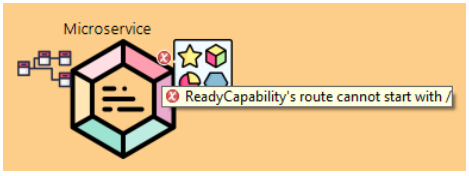


Figura 30. Mensaje de error cuando readyRoute comienza con /.

  - **period:** Período de tiempo que indica cada cuántos segundos se revisará la salud del microservicio. Valor por defecto: 5 segundos. El período debe ser mayor a 0.

Figura 31. Mensaje de error cuando el valor de period no es mayor a 0.

  - **initialDelay:** Período de tiempo que indica cuánto tiempo se esperará desde el arranque del microservicio hasta el comienzo de su chequeo periódico. Valor por defecto: 15 segundos.

3.2.2.1.2.3 Modelado interno de un BackendMicroservice

Cada elemento BackendMicroservice permite su modelado interno a través de un diagrama de clases, siguiendo una metodología de modelado similar a UML utilizando el paradigma orientado a objetos. Para modelar cada microservicio internamente, se debe hacer doble clic sobre el elemento y la herramienta gráfica cambia automáticamente a la **Representación de Clases”**.

3.2.2.2 Representación de Clases

La representación de clases es la segunda vista de la herramienta gráfica y permite modelar internamente cada uno de los microservicios, utilizando el paradigma de orientación a objetos y agregando propiedades específicas para la comunicación REST.

3.2.2.2.1 Creación de un elemento Class

Un elemento Class representa una clase que será contenida dentro del paquete “entities” y se crea arrastrando el elemento Class a dicho paquete.

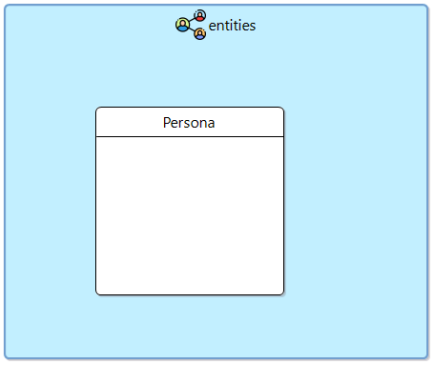


Figura 32. Elemento Class dentro del paquete entities.

Cada clase requiere la siguiente configuración:

- **name:** Nombre de la clase. No pueden existir dos clases con el mismo nombre y este debe ser distinto de null.

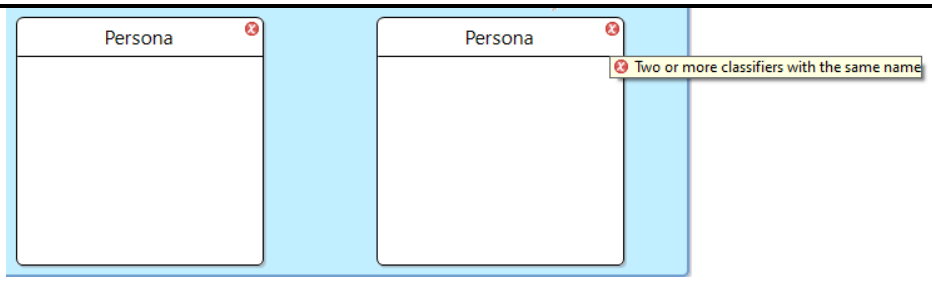


Figura 33. Mensaje de error cuando existen dos clases con el mismo nombre.

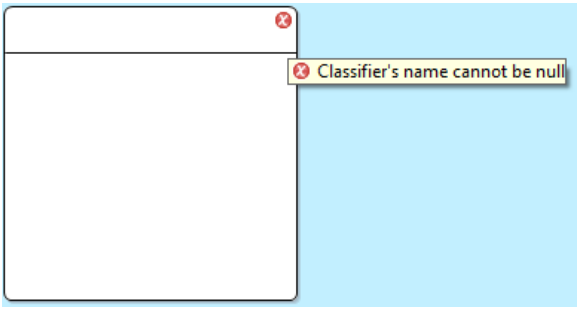


Figura 34. Mensaje de error cuando existe una clase con nombre nulo.

- **hasRoutes:** Atributo booleano que define la creación de rutas de ABM (Alta, Baja y Modificación) para la clase, en caso de ser verdadero. Valor por defecto: Falso.
- **isAbstract:** Atributo booleano que especifica si la clase es abstracta. Valor por defecto: Falso.

3.2.2.2.2 Creación de un elemento Attribute

Un elemento Attribute representa un atributo de una clase y se crea arrastrando el elemento Attribute a la clase contenedora.

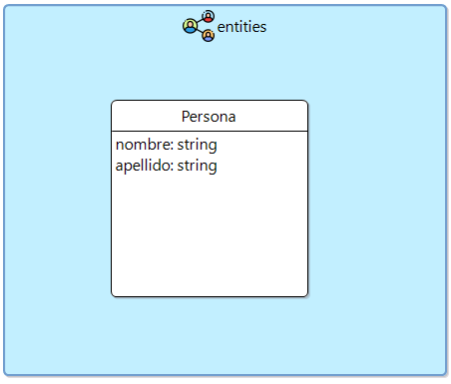


Figura 35. Elemento Class con dos atributos.

Cada atributo requiere la siguiente configuración:

- **name:** Nombre del atributo. No pueden existir dos atributos con el mismo nombre dentro de una clase y este debe ser distinto de nulo.

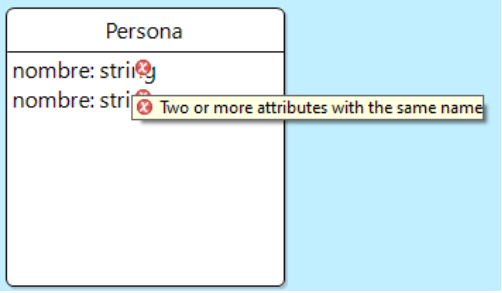


Figura 36. Mensaje de error para cuando existen dos atributos con el mismo nombre.

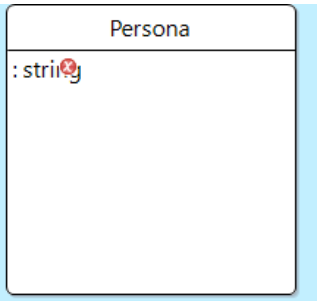


Figura 37. Mensaje de error cuando existe un atributo con nombre nulo.

- **type:** Tipo de dato del atributo. Valor por defecto: String.
- **required:** Atributo booleano que define si el atributo es de llenado obligatorio o no. Valor por defecto: Verdadero.
- **unique:** Atributo booleano que define si el valor del atributo será único en la base de datos. Valor por defecto: Falso.

3.2.2.2.3 Creación de un elemento DirectionalRelation

Un elemento DirectionalRelation representa una relación unidireccional entre clases y se crea arrastrando el elemento DirectionalRelation hacia el lienzo, seleccionando primero la clase origen y luego la clase destino.

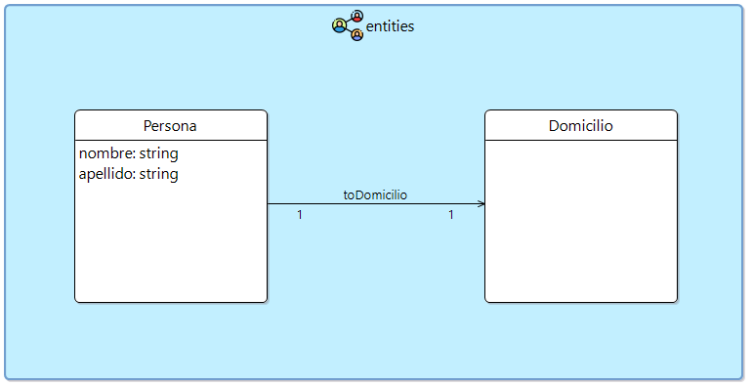


Figura 38. Relación unidireccional entre dos clases.

Cada relación unidireccional requiere la siguiente configuración:

- **relationName:** Nombre de la relación. Este valor es representa el rol de la relación y no pueden existir dos relaciones con el mismo rol dentro de una misma clase. Este valor también debe ser distinto de null.

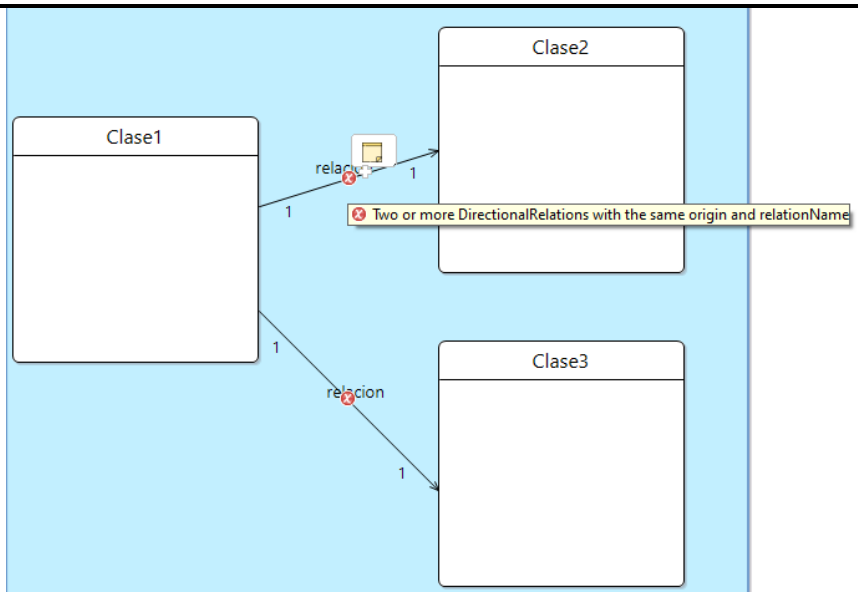


Figura 39. Mensaje de error cuando existen dos relaciones con el mismo nombre dentro de una clase.

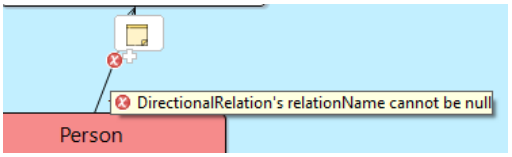


Figura 40. Mensaje de error cuando el nombre de una relación es nulo.

- **relationType:** Multiplicidad de la relación. Valor por defecto: OneToOne.
- **required:** Atributo booleano que define la obligatoriedad de la relación dentro de la base de datos. Valor por defecto: Verdadero.

3.2.2.2.4 Creación de un elemento ExternalDirectionalRelation

Un elemento DirectionalRelation representa una relación unidireccional entre dos clases ubicadas en diferentes microservicios, y se crea arrastrando el elemento ExternalDirectionalRelation hacia el lienzo, seleccionando la clase origen. Al hacerlo, se abre un diálogo que permite encontrar la clase destino navegando a través de los ambientes y microservicios presentes en la vista superior.

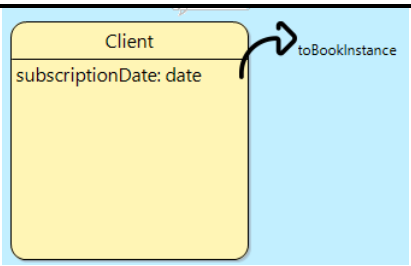


Figura 41. Clase con relación externa saliente.

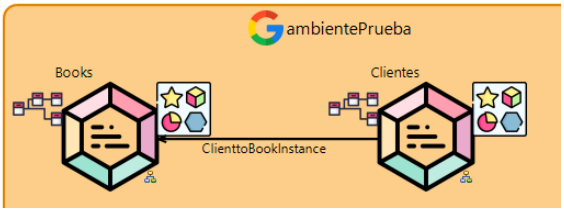


Figura 42. Vista de microservicios de una relación externa.

Este elemento requiere la misma configuración que el elemento DirectionalRelation.

3.2.2.2.5 Creación de un elemento Inheritance

Un elemento Inheritance representa una relación de herencia entre clases y se crea arrastrando el elemento Inheritance hacia el lienzo, seleccionando primero la clase origen (que hereda las propiedades) y luego la clase destino, es decir, la clase padre.

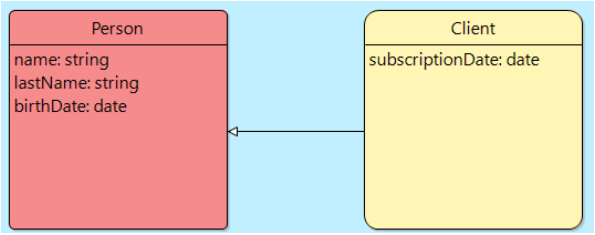
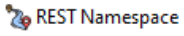


Figura 43. Relación de herencia entre dos clases.

Este elemento no requiere configuración adicional.

3.2.2.2.6 Creación de un elemento RESTNamespace

Cada microservicio tiene rutas adicionales a las generadas automáticamente de ABM (a través del atributo hasRoutes del elemento Class), por lo que un RESTNamespace es una manera de representar todas aquellas rutas adicionales que serán implementadas por el desarrollador y

que cumplen una funcionalidad propia de la lógica de negocio. Este elemento puede ser visto como el paquete entities pero que es contenedor de rutas de acceso REST y se crea arrastrando el elemento 

al lienzo.

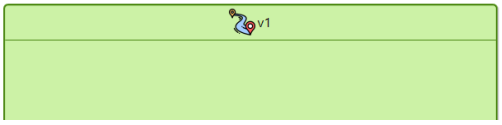


Figura 44. RESTNamespace con nombre igual a v1.

Para este elemento, se puede configurar su nombre, el cual debe ser no nulo y diferente para todos los RESTNamespaces dentro de un microservicio.

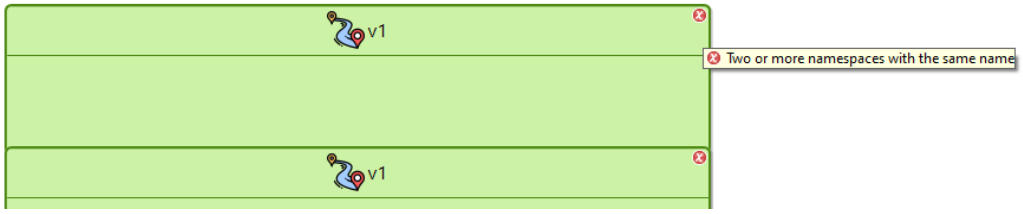


Figura 45. Mensaje de error para cuando existen dos namespaces con el mismo nombre.

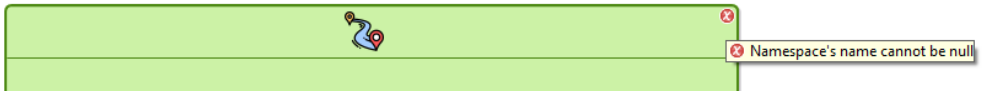
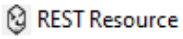


Figura 46. Mensaje de error para cuando existe un namespace con nombre nulo.

3.2.2.2.7 Creación de un elemento RESTResource

Dentro del elemento RESTNamespace, se pueden crear RESTResources, representando grupos de rutas REST agrupadas por algún concepto. Estos recursos luego serán utilizados para la generación automática de rutas y se crean arrastrando el elemento 

a un RESTNamespace.

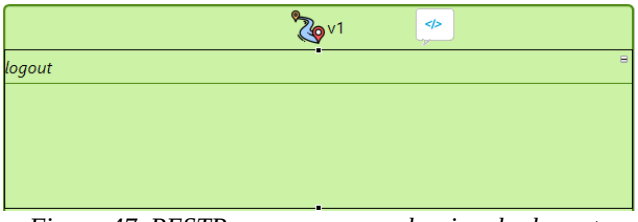


Figura 47. RESTResource con nombre igual a logout.

Para este elemento, se puede configurar su nombre, el cual debe ser no nulo y diferente para todos los RESTResources dentro de un RESTNamespace.

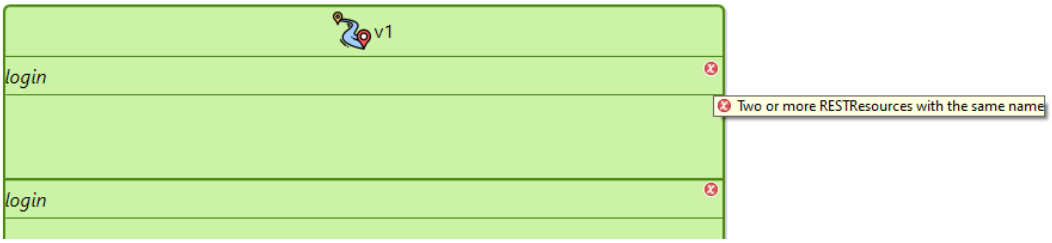


Figura 48. Mensaje de error cuando existen dos recursos con el mismo nombre dentro del mismo namespace.

Figura 49. Mensaje de error cuando existe un recurso con nombre nulo.

3.2.2.2.8 Creación de un elemento RESTRoute

Finalmente, arrastrando el elemento RESTRoute a un RESTResource, podemos crear una ruta de acceso REST, representando un endpoint que el microservicio contenedor expondrá al exterior. Este elemento se crea arrastrando una </> RESTRoute hacia un RESTResource.

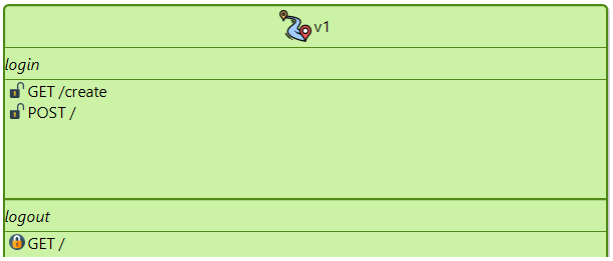


Figura 50. Ejemplo de RESTRoutes dentro de dos RESTResources diferentes.

Para cada RESTRoute, se requiere la siguiente configuración:

- **pathExtension:** Valor de la ruta de acceso. Valor por defecto: "/". No pueden haber dos rutas con el mismo path en el mismo RESTResource y este valor debe ser distinto de null.

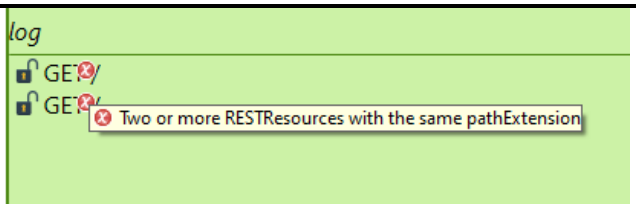


Figura 51. Mensaje de error para cuando existen dos o más rutas con el mismo pathExtension.

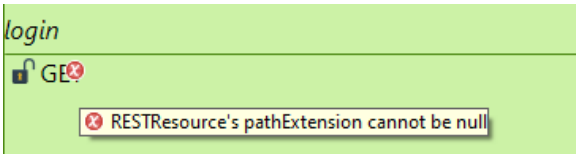


Figura 52. Mensaje de error para cuando existe una ruta con pathExtension nulo.

- **parameters:** Parámetros de la ruta separados por ";".
- **httpMethod:** Método HTTP utilizado por la ruta. Valor por defecto: GET.
- **requiresAuth:** Atributo booleano que define si el usuario debe estar o no autenticado antes de realizar la petición. En caso de estar marcado como verdadero, el ícono de la ruta cambia de una cruz roja a una tilde verde. Valor por defecto: Falso.

```
localhost:8000/v1/login/create
```

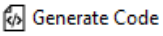
Figura 53. Ruta final mapeada a través de un RESTNamespace llamado v1, contenedor del RESTResource login que a su vez contiene la RESTRoute con pathExtension igual a /create.

3.2.2.3 Validaciones del diagrama

El usuario puede validar su modelo para verificar su consistencia haciendo uso de las validaciones del diagrama. Para ello, debe realizar clic derecho sobre el lienzo y seleccionar la opción Validate Diagram para ejecutar todas las validaciones.

Aquellos elementos que tengan errores (e.g. dos clases con el mismo nombre), mostrarán una cruz roja con un mensaje de error. Es importante tener en cuenta que la herramienta permitirá la generación automática sin tener en cuenta los mensajes de error, ya que los mismos son para advertir al usuario de posibles inconsistencias de modelado.

3.2.3 Motor de generación automática

Una vez que el usuario finaliza su modelo, puede generar el código correspondiente utilizando la herramienta gráfica a través del elemento 

de la vista de microservicios.

La salida generada se genera utilizando la Ingeniería por Modelos, tal cual se describe en la Figura 1, y está estructurada en cinco niveles:

- **Generación de infraestructura con Terraform**
- **Generación de manifiestos de Kubernetes.**
- **Generación de archivo Dockerfile.**
- **Generación de integración continua con Github Actions.**
- **Generación de código en Node.js y Python.**

A continuación, se describe cada nivel de generación y los pasos que debe seguir el usuario para su despliegue.

3.2.3.1 Generación de infraestructura con Terraform

El motor de generación crea automáticamente los archivos main.tf y variables.tf utilizando Terraform, permitiendo el aprovisionamiento automático en un cluster en la nube utilizando GKE (Google Kubernetes Engine). Estos archivos utilizan la información del elemento GoogleEnvironment y de su configuración de Kubernetes.

Para aprovisionarse de recursos utilizando GKE y estos archivos generados automáticamente, es necesario instalar gcloud y Terraform y ejecutar los siguientes comandos:

1. gcloud init
2. terraform init

3. terraform apply

3.2.3.2 Generación de manifiestos de Kubernetes

Para cada microservicio, el motor de generación automática crea su deployment y su service correspondiente a Kubernetes, permitiendo la orquestación de sus respectivos contenedores. Estos archivos son parametrizados a través de los datos de cada microservicio y de su configuración de Kubernetes.

Además, para permitir el ingreso al ambiente, se genera un archivo Ingress que redistribuye el tráfico entrante a cada módulo.

Una vez instalada la herramienta kubectl, es necesario ejecutar el comando <***kubectl apply -f . –recursive***> sobre el directorio de Kubernetes para aplicar los manifiestos en el cluster.

3.2.3.3 Generación de archivo Dockerfile

Para cada microservicio, el motor de generación automática crea el archivo Dockerfile, permitiendo empaquetar cada módulo en una imagen de Docker.

Para hacerlo localmente, es necesario tener instalado Docker y ejecutar el comando <***docker build -t {usuarioDeDocker}/{nombreDelMicroservicio}:{versionDelMicroservicio} .***> sobre el archivo, reemplazando las palabras entre corchetes por sus respectivos valores.

3.2.3.4 Generación de integración continua con Github Actions

El motor de generación automática crea un archivo de Github Actions, que permite el empaquetado de cada microservicio en una imagen Docker y su posterior push a Docker Hub, evitando la necesidad de ejecutar el comando docker build de manera local.

Para ello, es necesario cargar en el repositorio de Github de cada microservicio los siguientes secrets:

- DOCKER_LOGIN_USERNAME: Usuario de Docker Hub.
- DOCKER_LOGIN_PASSWORD: Contraseña de Docker Hub.

3.2.3.5 Generación de código en Node.js y Python

El motor de generación automática genera automáticamente, para cada microservicio, el código correspondiente en el lenguaje seleccionado (Node.js o Python).

El código generado tiene las siguientes características:

- Funcionamiento como API que expone el puerto indicado.
- Conexión a base de datos según los datos indicados (MongoDB, por ejemplo).
- Alta, Baja y Modificación de las entidades modeladas a través de controladores REST.
- Comunicación entre microservicios en caso de modelar relaciones externas entre módulos.
- Estructura de rutas de negocio (a través de RESTNamespaces) generadas automáticamente y protegidas mediante JWT (JSON Web Token).

3.3 Metodología de trabajo con Accelerate

Al implementar la herramienta Accelerate al marco de trabajo, se observa que la metodología cambia, ya que genera código e infraestructura que tradicionalmente crea el desarrollador y el DevOps respectivamente. A continuación, se realiza un análisis de los cambios metodológicos desde los tres roles más afectados por los cambios.

3.3.1 Cambio desde el punto de vista del arquitecto de software

En la nueva metodología, el arquitecto de software posee una sola herramienta que le permite modelar la arquitectura del sistema de manera integral, permitiendo diagramar un sistema distribuido a través de microservicios. Debido a ello, el arquitecto de software utilizará Accelerate como principal herramienta, y generará el código y la infraestructura correspondiente a los modelos realizados.

3.3.2 Cambio desde el punto de vista de los desarrolladores

Tradicionalmente, el desarrollador recibía los modelos del arquitecto de software y comenzaba la programación desde cero. Sin embargo, en esta nueva metodología, el desarrollador recibe la estructura de código inicial, por lo que de ahora en más solamente deberá codificar lo respectivo a la lógica de negocio.

3.3.3 Cambio desde el punto de vista del DevOps

En esta nueva metodología, el DevOps podrá realizar el aprovisionamiento automático en la nube con la infraestructura generada, por lo que no será necesario que cree una infraestructura ajustada a los modelos realizados, ya que esta se genera de manera automática. Sin embargo, el trabajo del DevOps seguirá siendo útil, permitiendo la gestión de la infraestructura aprovisionada y el ajuste de detalles específicos que no hayan sido generados.

3.4 FAQs

- ¿Qué puedo hacer si no aparece la representación gráfica de mi instancia XMI?

  En este caso, es posible que, al cerrar la herramienta gráfica, la representación se haya dañado, por lo que se debe crear un nuevo Modeling Project y copiar la instancia XMI al mismo.

- ¿Qué significa la cruz roja en un elemento de modelado?

La cruz roja en un elemento de modelado significa un error en el diagrama que puede traducirse a un problema en la generación automática. Para solucionarlo, es necesario solucionar el error indicado en el mensaje y validar el diagrama nuevamente. Un ejemplo de mensaje de error podría ser “No pueden existir dos clases con el mismo nombre” ya que, al realizar la generación automática, se generarían dos archivos con el mismo nombre.

- ¿Cómo elimino un elemento del diagrama?

  Para eliminar un elemento, existen dos opciones:

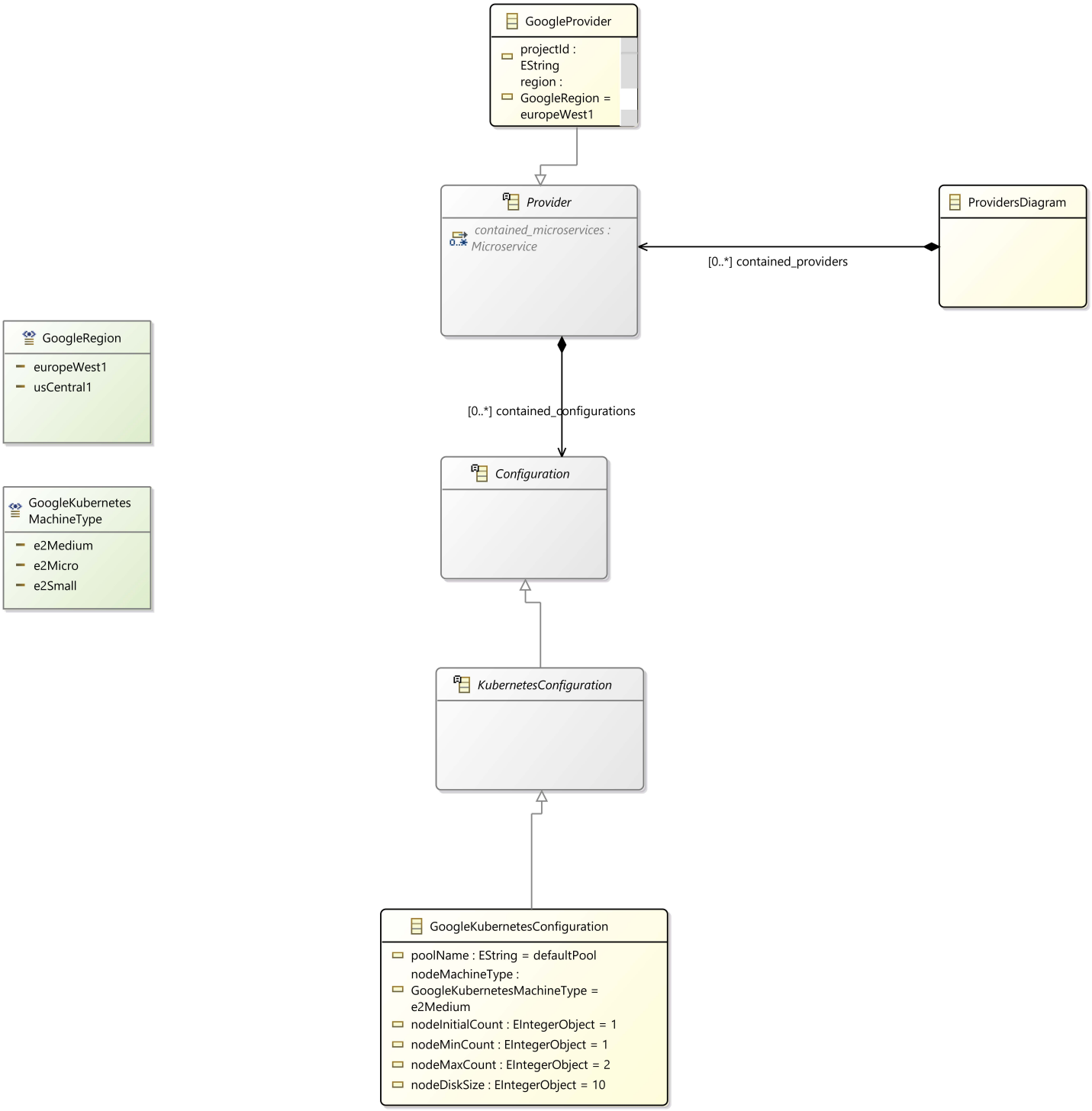
  - **Opción 1:** Realizar clic derecho sobre el elemento y elegir la opción “Delete”.
  - **Opción 2:** Seleccionar el elemento y presionar la tecla Suprimir.

- ¿Cómo ajusto el tamaño de un elemento gráfico?

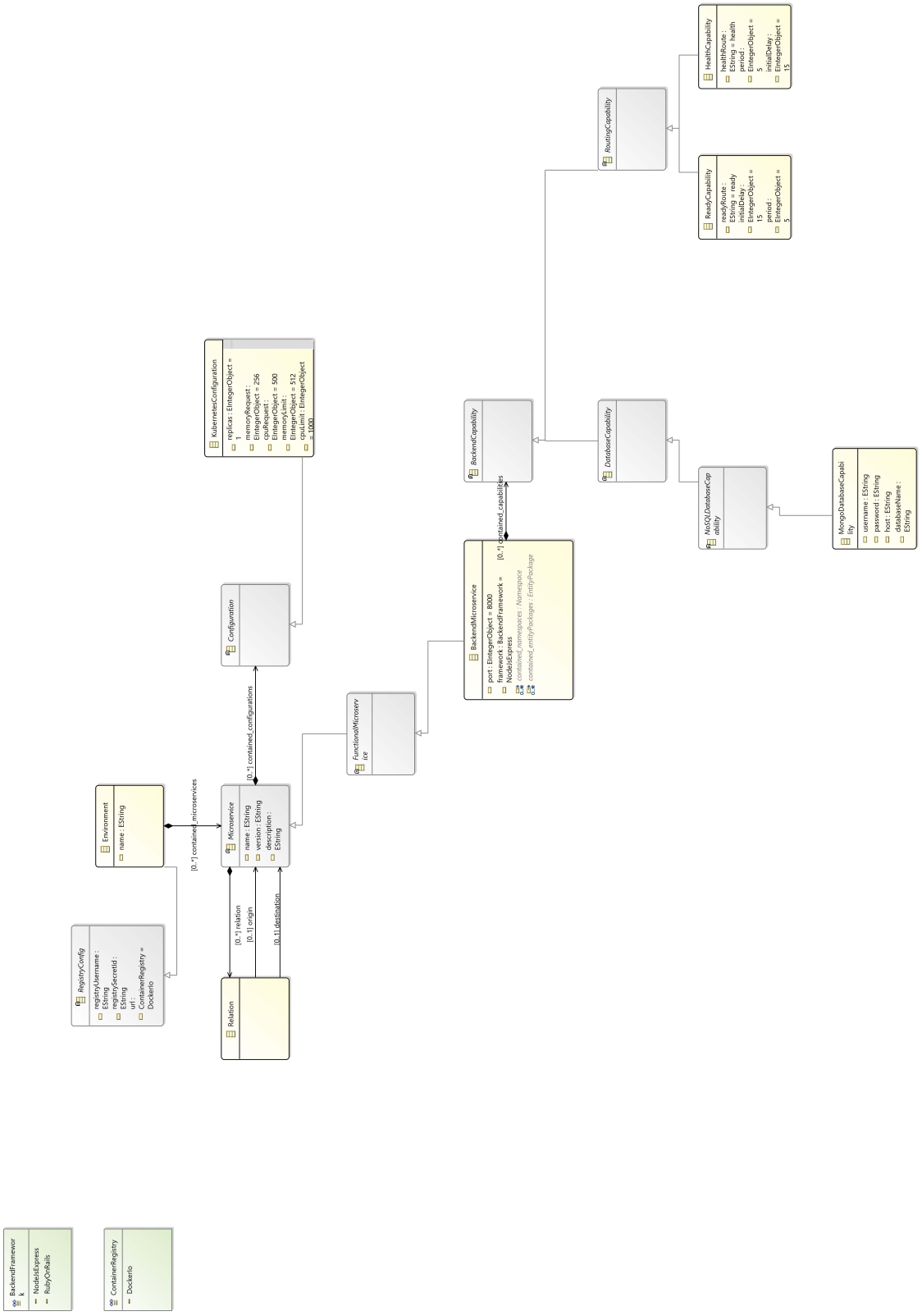
  El tamaño de un elemento del diagrama puede ser ajustado a través de los pequeños cuadrados que se muestran en sus vértices y aristas al seleccionarlo. Un ejemplo de este procedimiento es el siguiente:

  1. Seleccionar el elemento gráfico.
  2. Seleccionar uno de los cuadrados observados y mantener el clic presionado.
  3. Arrastrar el cursor para modificar el tamaño del elemento.

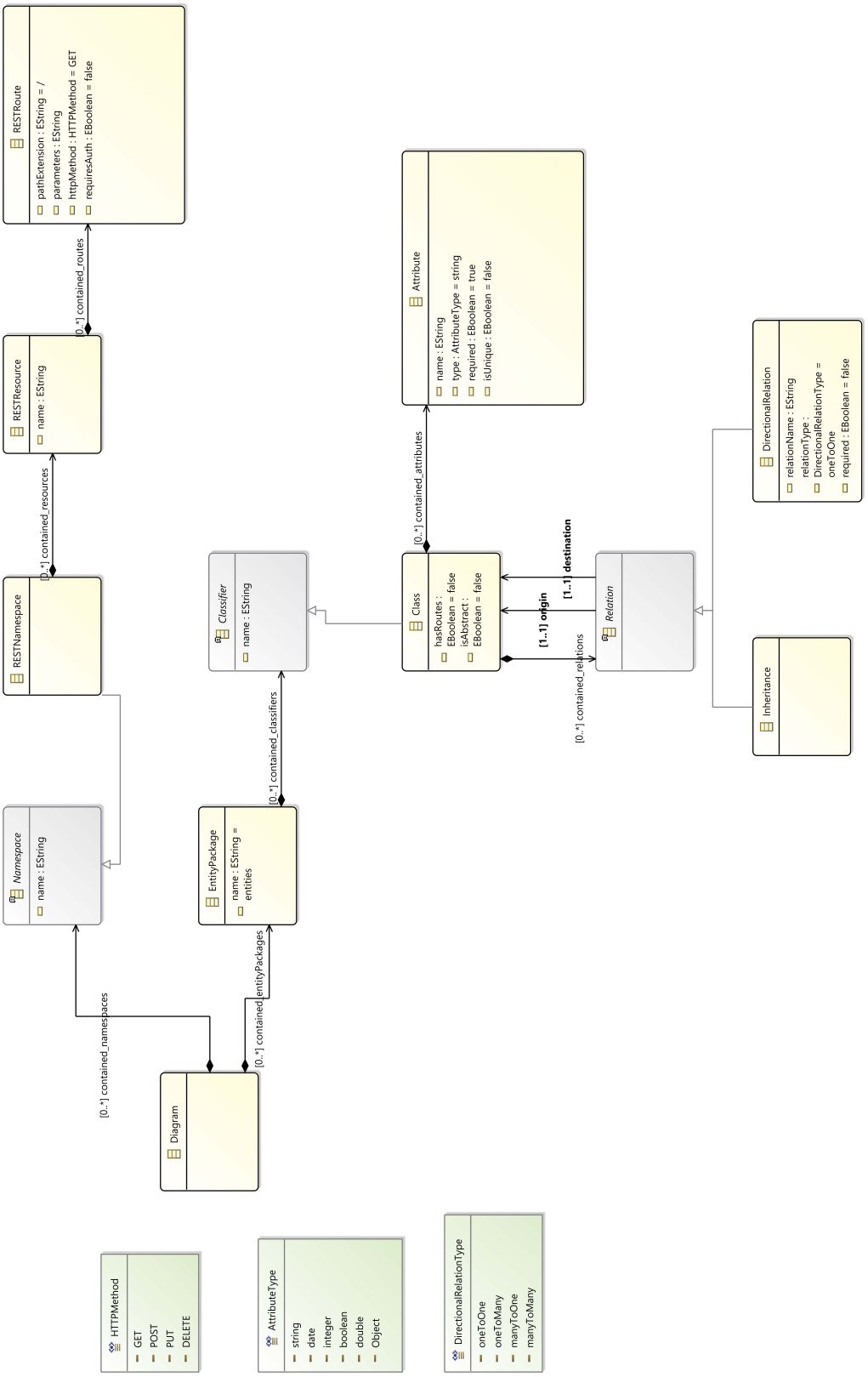
Anexo 4 – accelerateMLI


GoogleProvider
projectId : EString
region : GoogleRegion = europeWest1
Provider
contained_microservices : Microservice
0..*
ProvidersDiagram
[0..*] contained_providers
GoogleRegion
europeWest1
usCentral1
[0..*] contained_configurations
Configuration
GoogleKubernetes MachineType
e2Medium
e2Micro
e2Small
KubernetesConfiguration
GoogleKubernetesConfiguration
poolName : EString = defaultPool
nodeMachineType : GoogleKubernetesMachineType = e2Medium
nodeInitialCount : EIntegerObject = 1
nodeMinCount : EIntegerObject = 1
nodeMaxCount : EIntegerObject = 2
nodeDiskSize : EIntegerObject = 10

Anexo 5 – accelerateMLS


HealthCapability
healthRoute : EString = health
period : EIntegerObject = 5
initialDelay : EIntegerObject = 15
RoutingCapability
ReadyCapability
readyRoute : EString = ready
initialDelay : EIntegerObject = 15
period : EIntegerObject = 5
KubernetesConfiguration
replicas : EIntegerObject = 1
memoryRequest : EIntegerObject = 256
cpuRequest : EIntegerObject = 500
memoryLimit : EIntegerObject = 512
cpuLimit : EIntegerObject = 1000
BackendCapability
DatabaseCapability
NoSQLDatabaseCapability
MongoDatabaseCapability
username : EString
password : EString
host : EString
databaseName : EString
Configuration
BackendMicroservice
port : EIntegerObject = 8000
framework : BackendFramework = NodeJsExpress
contained_namespaces : Namespace
contained_entityPackages : EntityPackage
[0..1] contained_capabilities
FunctionalMicroservice
[0..1] contained_configurations
Environment
name : EString
[0..*] contained_microservices
Microservice
name : EString
version : EString
description : EString
[0..*] relation
[0..1] origin
[0..1] destination
Relation
RegistryConfig
registryUsername : EString
registrySecretId : EString
url :
ContainerRegistry = DockerHo
BackendFramework
NodeJsExpress
RubyOnRails
ContainerRegistry
DockerHo

Anexo 6 - accelerateMLC


RESTRoute
pathExtension : EString = /
parameters : EString
httpMethod : HTTPMethod = GET
requiresAuth : EBoolean = false
[0..*] contained_routes
RESTResource
name : EString
[0..*] contained_resources
RESTNamespace
Namespace
name : EString
Classifier
name : EString
Class
hasRoutes : EBoolean = false
isAbstract : EBoolean = false
[0..*] contained_attributes
Attribute
name : EString
type : AttributeType = string
required : EBoolean = true
isUnique : EBoolean = false
[1..1] destination
[1..1] origin
Relation
[0..*] contained_relations
DirectionalRelation
relationName : EString
relationType : DirectionalRelationType = oneToOne
required : EBoolean = false
Inheritance
[0..*] contained_classifiers
EntityPackage
name : EString = entities
[0..*] contained_entityPackages
[0..*] contained_namespaces
Diagram
HTTPMethod
GET
POST
PUT
DELETE
AttributeType
string
date
integer
boolean
double
Object
DirectionalRelationType
oneToOne
oneToMany
manyToOne
manyToMany