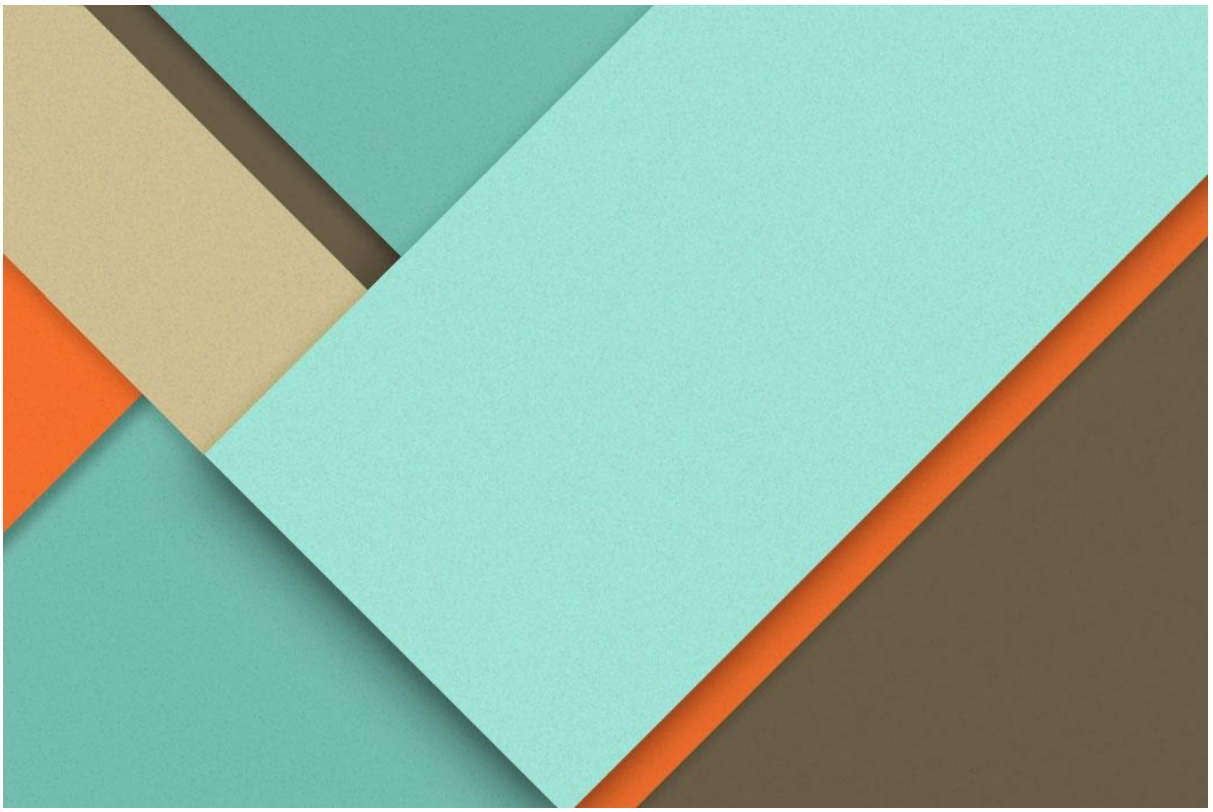


Universidad Tecnológica Nacional
Facultad Regional Santa Fe
Ingeniería en Sistemas de Información

–

Actualización del soporte tecnológico en una empresa de servicios funerarios

Cabrera Juan Ignacio - LU 23180 - juanignaciocabrera@live.com
Fedele Fausto - LU 23333 - faustofedele2013@gmail.com

–

Director: Mg. Ing. Vrancken Lisandro

Año 2021

Índice

1. Introducción

1.1 Temática del proyecto

El presente proyecto se realizó con el fin de cubrir las necesidades tecnológicas y de gestión de la información de la empresa de servicios funerarios “Cochería Zurdo”, ubicada en la ciudad de Bovril, Entre Ríos. La propuesta se enfocó en revisar los procesos de negocio y renovar a su vez la tecnología utilizada para el soporte de la operación, específicamente, el hardware utilizado y el sistema de información empleado para el registro de asociados, pagos, zonas, cobradores, deudas, comprobantes, planes, tarifas, coberturas, servicios, entre otros.

La organización cliente es una empresa familiar que desarrolla sus actividades desde el mes de marzo de 1991. El sistema actual que da soporte a la operación fue adquirido en el año 1994, con el objetivo de lograr una automatización en el registro de operaciones manuales llevadas a cabo por la organización en ese tiempo.

La problemática origen se centra entonces en las dificultades de la empresa para resolver aspectos de la operación y en la prestación de servicios con el soporte tecnológico vigente, en un negocio funerario que ha ido experimentando cambios a través del tiempo. En lo que respecta al soporte tecnológico, el proyecto se fundamenta debido a la obsolescencia del hardware de impresión utilizado y del sistema actual, a través de sus falencias técnicas (por ejemplo, en aspectos de seguridad y auditabilidad), así como también en aspectos de usabilidad que dificultan su adaptación a los cambios de la compañía y su entorno. Estas particularidades han generado a través del paso del tiempo una mayor complejidad, poca flexibilidad, dificultades en el uso y en la capacitación de nuevos recursos.

Asimismo, el proyecto se basó en procesos comunes del negocio, con el fin de generar una solución que cumpla con las necesidades básicas de gestión de todo tipo de organizaciones funerarias.

El negocio en el cual se desarrolló el presente proyecto presenta una terminología específica que vale la pena mencionar para acompañar la lectura del presente documento, por lo que se define a continuación un glosario.

1.2 Glosario

- Adherente: persona aportante, por lo general de vínculo familiar con el socio, que abona un fragmento de la cuota societaria y que posteriormente puede adquirir titularidad de un plan ante una posible baja del socio.
- Asociado: socio o adherente.
- Cobrador: persona encargada de repartir comprobantes y recibir pagos por parte de los asociados en una determinada zona.
- Cochería: sinónimo de funeraria.
- Comprobante: documento que no sirve como comprobante legal propiamente dicho, sino un objeto físico que describa el importe con su respectiva justificación.
- Plan: referencia el tipo de contrato que determina la metodología de pago de los asociados.

  Existen los tipos:

  - Individual (conformado por sólo el socio): abona un determinado precio determinado por la tarifa asociada.
  - Titular y adherente (conformado por un socio y un adherente): se abona un precio base determinado por la tarifa asociada tanto por parte del socio como del adherente.
  - Familiar (conformado por un socio y más de un adherente): se abona un precio base y además, dependiendo la edad del adherente se determina su respectiva tarifa. Menores de 18 no abonan.

- Servicio: Conjunto de actividades extra a los procesos de la funeraria en sí, pero que aporta gran valor en la organización.
- Socio: persona referente de un plan, es a quien el mismo ofrece cobertura y sobre quien recaen los servicios y seguros.
- Usuario: persona integrante del personal administrativo de la cochería.
- Tarifa: identificador de un conjunto de precios, generalmente compuestos por un precio base y además un precio por cada intervalo de edad de 5 años que será determinante para determinar el precio de un plan.
- Zona: corresponde a una región no necesariamente geográfica delimitada, sino que le sirva de referencia al usuario para determinar un área que comprenda cierto conjunto de asociados.

1.3 Contexto y problemática

La problemática principal puede comenzar a analizarse entendiendo a la organización. La cochería se trata de un negocio familiar, con una operación conocida pero que carece de optimización en sus procesos y con pocas perspectivas de crecimiento, agregando además que cubre una fracción de mercado, podría decirse una funeraria de nicho. Esto muestra una visión no orientada a los procesos y a los beneficios de un adecuado soporte tecnológico, que la empresa ha mantenido durante su trayectoria.

Si bien actualmente el cliente cuenta con un sistema, éste quedó obsoleto, sin mantenimiento y se utilizan muy pocas funcionalidades del total que contiene: su foco está en la utilización del listado de socios para realizar la impresión de comprobantes, siendo estos únicamente demostrativos ya que sólo contienen una cantidad reducida de datos como la mínima información de los integrantes de un plan y una breve descripción de la cochería.

Además, posee escasa amigabilidad y usabilidad lo cual genera en el usuario mayor resistencia a utilizar todas las funcionalidades que el sistema ofrece.

Un punto donde el cliente pone mayor foco es en el gasto irrelevante a la hora de la impresión de comprobantes de pago. El proceso actual se lleva a cabo con una impresora matricial de marca EPSON y modelo Action Printer 2000, la cual oprime una cinta de tinta sobre la hoja para registrar la impresión, simulando el funcionamiento de una máquina de escribir. Dada su antigüedad, el costo y el tiempo es muy elevado.

La solución diseñada para dar respuesta a esto fue materializada en un sistema con tecnología innovadora totalmente adaptable a la nueva impresora láser marca HP y modelo Pro 102w adquirida por la organización, donde se reduce notablemente los tiempos y costos a la hora de generar los comprobantes. Además, el producto ataca las debilidades y problemáticas que el sistema actual posee, las cuales se listan a continuación:

- La dificultad para navegar entre los módulos que componen el sistema es alta, ya que solo permite el uso de teclas o comandos y complica a los usuarios acostumbrados a utilizar el mouse.
- Para imprimir comprobantes, el único filtro que se permite es el rango por número de socios. Por lo que para realizar una impresión de todos los comprobantes correspondientes, es necesario saber con exactitud la cantidad total de socios por lo que obliga al usuario a realizar un listado previo sólo con el fin de saber esta incógnita para poder realizar la impresión de todos los resúmenes mensuales.
- Para la presentación del resultado de una consulta, se visualiza en pantalla una tabla que es desplegable únicamente por unidades de una línea por vez (presionando flecha hacia abajo) y no está paginada. Además, el formato de exportación cuenta con una única opción que es un archivo de extensión txt con los datos de dicha tabla.

- El uso de localidades se limita a un identificador y un nombre, no se lleva registro de datos útiles como provincias, códigos postales, etc. dato necesario para la generación de estadísticas.
- Para la búsqueda y obtención de un socio en el listado el único filtro que existe es por apellido y nombre, limitando búsquedas amplias de socios con las mismas características que no sean las mencionadas. Ejemplo: todos los socios nacidos en cierto año, o de la misma edad, o con cierto estado, etc.
- La organización de los módulos o formas de acceder a las funcionalidades a través de secciones en ciertas ocasiones no es intuitiva u óptima, por ejemplo para eliminar una zona era necesario ir a la sección de “Otros”, luego “Zona” y finalmente “Eliminar zona”.

1.4 Alcance del sistema

El alcance de la solución se resume en los siguientes módulos principales:

- Alta, baja, modificación y listado de socios, adherentes, cobradores, zonas, planes, tarifas y sus rangos, servicios, entre otros.
- Manejo financiero. Impresión de comprobantes y registros de pagos por cada uno de los titulares. Venta de servicios extras.
- Algoritmo de cobertura. Desarrollo de un algoritmo de predicción para proponer una fecha de cobertura teniendo en cuenta diferentes factores.
- Analytics. El sistema posee un conjunto de módulos que consumen datos para ofrecer al usuario y poder así tomar mejores decisiones.
- Auditoría. Cada inserción, modificación y baja quedará registrado en la base de datos de modo tal de tener trazabilidad de dichas acciones.

1.5 Organización del informe

El informe está dividido en tres secciones principales, sumado a un bloque de posibles extensiones, una conclusión y el anexo donde se detallan diagramas y especificaciones de casos de uso.

En primer lugar, se encuentra la información referida a los modelos de procesos que describen las actividades principales y secundarias de la organización, sumado a las mejoras que el equipo propuso para una mejor eficiencia en dichas tareas.

Luego se detallan los conceptos de la ingeniería de software utilizados para construir una base teórica que se utilizó para poder generar tanto el informe como un producto de calidad.

Seguidamente se define la documentación del proyecto donde describimos con mayor detalle la metodología de desarrollo de software utilizada, información precisa de cada una de las etapas de desarrollo, funcionalidades, capturas de pantallas del sistema y toda la definición de requerimientos, casos de uso y la arquitectura llevada a cabo con sus diferentes vistas, entre otros.

En la última parte, se presenta la sección de extensibilidad, donde se detallan posibles extensiones del aplicativo construído, un bloque de conclusión donde se indica lo aprendido en el desarrollo del proyecto y el anexo donde se encuentran los diagramas y especificaciones de los casos de uso.

2. Modelo de procesos

2.1 Introducción

En esta sección, se detallan cada uno de los procesos que en conjunto forman el núcleo de la operación de la organización. También, se describen las actividades que fueron mejoradas luego del relevamiento a través del mecanismo de revisión del sistema obsoleto que maneja actualmente la empresa.

El proceso de renovación fue el principal foco a la hora de relevar los procesos de negocios claves de la organización e identificar puntos de inflexión para mejorar cada una de las actividades y ganar eficiencia.

Además, fue necesario realizar un conjunto de entrevistas con el cliente con la finalidad de poder recopilar la mayor cantidad de información y sus propuestas para la mejora de los mismos.

2.2 Procesos de negocio principales

A continuación, se especifican los procesos principales de la organización que han sido identificados. Además, se detallan las potenciales mejoras que se llevan a cabo en el diseño del nuevo sistema para mejorar la eficiencia de los mismos.

2.2.1 Socios y Adherentes

El proceso de vinculación de nuevas personas a la organización es uno de los flujos más importantes, ya que es donde se generarán nuevos ingresos. Es por esto que el equipo puso mucho foco en dichas actividades, realizando numerosas entrevistas donde se refinaban cuestiones puntuales y observando detalladamente el sistema anterior para ver las potenciales mejoras.

Como idea principal se diseñó un formulario donde el usuario final pueda cargar la información necesaria de manera correcta pero sin perder velocidad, ya

que si se encontraba con el cliente, la idea es demorarlo lo menos posible y evitar pérdidas de tiempo. Los datos solicitados es información que un usuario puede responder rápidamente.

Una vez completa la información básica, el usuario asignará al cliente a una zona, donde será relacionado, indirectamente, a un cobrador para que el mismo pueda agregarlo a su lista de clientes a cobrar. Además, el usuario final consultará si desea agregar adherentes o no, para verificar a qué plan agregar al mismo.

Luego de esto, se le pedirá un informe médico de patologías preexistentes, ya que gracias a esta información, el sistema deberá proponer una fecha de comienzo de cobertura para todos los nuevos socios. Seguidamente, se cargará información de los adherentes, si existiesen.

Otro punto importante a destacar, es que es posible agregar adherentes a un socio titular no solo en el proceso de alta del socio mismo, sino en cualquier momento. Esto aporta gran flexibilidad.

Además, el usuario final tiene la posibilidad de consultar un padrón tanto de socios como de adherentes, donde en primera instancia, se visualiza la información más relevante de los mismos, como por ejemplo, si el mismo tiene cobertura o no. Luego, a través de un botón simplificado, el mismo puede modificar datos personales, ampliar la información o simplemente dar de baja. Los casos posibles de baja son: fallecimiento, renuncia o morosidad, donde este último es propuesto por el sistema automáticamente cuando el socio posee tres o más comprobantes sin abonar, permitiéndole al usuario final la opción de dar de baja al cliente.

A continuación, se detalla un diagrama de actividad del proceso de alta de un socio y sus adherentes, cuyo objetivo es registrar nuevos clientes a la organización:

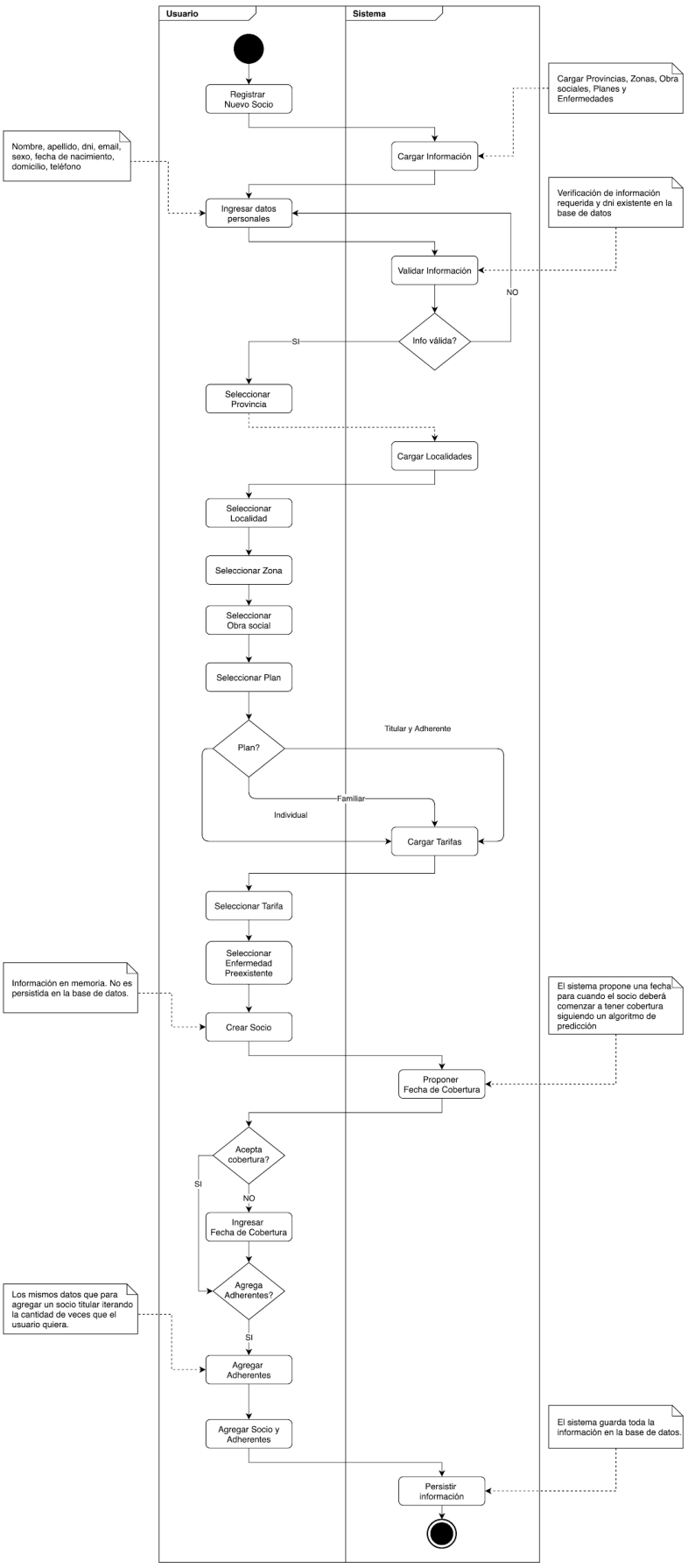


Figura 1. Alta de socios y adherentes

2.2.2 Planes y Tarifas

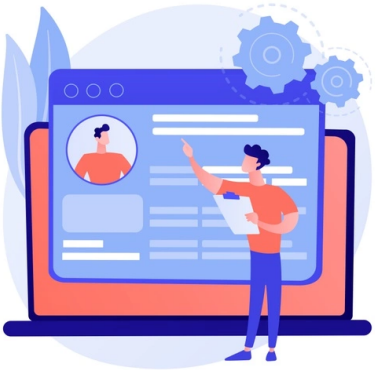
A través del proceso de revisión, se detectó que el sistema antiguo manejaba cinco tipos de planes:

1. Individual
2. Titular y Adherente
3. Familiar
4. Familiar con Obra Social
5. Jubilados

Luego de una revisión por completo de la plantilla de socios se observó que ninguno tenía asignado los planes 4 y 5, es por esto, que el nuevo sistema solamente contempla los primeros tres planes, sin perder flexibilidad evitando la redundancia de información.

Por lo tanto, el nuevo sistema cuenta con tres planes:

1. Individual
2. Titular y Adherente
3. Familiar



El primer plan es para aquel cliente que quiere asociarse solo y no posee pareja ni hijos para agregar. El siguiente, se adapta completamente a una pareja o un par de personas con algún vínculo. Y por último, el plan Familiar es aquel que se acomoda a un grupo familiar.

La principal diferencia además de la cantidad de personas que pueden estar en el plan es el manejo de las *tarifas* de cada uno.

El plan Familiar tiene un “precio base” que abarca al titular del grupo sumado al primer adherente, luego el valor del resto de los adherentes quedará determinado por diferentes rangos de edades. La suma del precio base más cada

uno de los precios de los adherentes es el valor final del comprobante a generar mensualmente.

En cambio, los planes restantes solamente manejan el rango de edades sin tener en cuenta un precio base, es decir, depende de la edad del titular y de su adherente o del titular solo para ver a qué rango se adapta y cuál es el valor del mismo. El rango de edades va de 0 años a 105 años, con un intervalo de 5 años, es decir, el rango 1 va de 0 a 5, el rango 2 de 6 a 10, etc.

Al dar de alta una nueva tarifa, se deberá agregar a qué plan pertenece y los precios para cada rango de edad.

En comparación al sistema anterior, el mismo no discrimina las tarifas por planes, generando un gran desorden, ya que el usuario no sabía qué tarifa seleccionar para un determinado plan. Con la relación entre tarifas y planes se mejoró la eficiencia y la usabilidad del sistema por parte del usuario final.

2.2.3 Cobertura

En vista de la necesidad de determinar una dinámica para el cálculo de una fecha estimada de comienzo de cobertura de los servicios, se optó por investigar las métricas a nivel nacional de mortalidad y sus respectivas causas. Es por esto que se extrajeron los siguientes datos desde el sitio web de INDEC[1]:

Mortalidad / Año	2013	2014	2015	2016	2017
Enfermedad sistema circulatorio	94.099	92.190	96.252	101.928	97.219
Tumores	60.294	60.791	62.621	62.625	62.731
Enfermedad sistema respiratorio	52.522	54.250	56.901	65.182	64.869
Enfermedad infecciosa y parasitaria	14.282	14.534	13.565	13.897	14.198
Diabetes	8.045	8.201	9.223	9.599	8.893
Enferemedad sistema urinario	10.966	11.664	12.698	13.135	12.897

Tabla 1. Cantidad de muertes por enfermedad por año

[1] Indicadores de salud. Indec. https://www.indec.gob.ar/indec/web/Nivel4-Tema-4-32-94

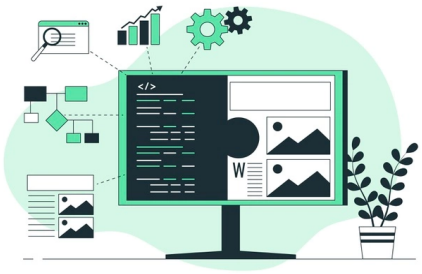
Mortalidad (miles) / Año	2012	2013	2014	2015	2016	2017
<1	11,10	10,80	10,60	9,70	9,70	9,30
1 a 4	0,50	0,50	0,40	0,40	0,40	0,40
5 a 14	0,20	0,20	0,20	0,20	0,20	0,20
15 a 24	1,00	0,90	0,90	0,90	0,80	0,80
25 a 34	1,10	1,10	1,10	1,10	1,10	1,10
35 a 44	1,80	1,80	1,80	1,70	1,80	1,70
45 a 54	4,40	4,40	4,20	4,20	4,30	4,00
55 a 64	10,90	10,90	10,60	10,70	11,00	10,20
65 a 74	24,50	24,60	24,10	24,30	25,30	24,10
> 75	88,30	88,90	87,80	88,60	92,60	88,20

Tabla 2. Promedio de muerte por edad por año

Dado que la información recabada era demasiado antigua, se decidió realizar un pronóstico de tan sólo un año para no insertar demasiado margen de error ya que es un proceso sujeto a incertidumbres, con el fin de tener una percepción un poco más cercana a la realidad.

Un pronóstico es una predicción de lo que ocurrirá en el futuro. Existen diferentes tipos:

- Pronósticos de largo y corto plazo: en este caso puntual se considera de corto plazo.
- Micro y macro pronósticos: dado el dominio del problema de la mortalidad nacional, en esta ocasión se trata de un micro pronóstico
- Pronósticos cualitativos y cuantitativos: se prefirió usar un método cuantitativo para tener la mayor precisión posible en el cálculo de los valores para componer la fórmula ya que hacen uso de técnicas estadísticas.



Debido a que, como se detalló anteriormente, se trata de un micro pronóstico a corto plazo cuantitativo, se hará uso de un método de análisis de serie de tiempo conocido como metodología de alisado exponencial ajustado[2], ya que no existen tendencias uniformes ni estacionales en las tasas de mortalidad nacional, la cual es en general fluctuante.

Se muestra un ejemplo a continuación a modo de resumen con los valores de las variables α y β siguientes:

α	0,75
β	0,30

Tabla 3. Coeficientes alisado exponencial ajustado

Una vez obtenidos todos los valores correspondientes al año 2018, tanto para edades como para enfermedades, se procede a calcular el promedio de cada característica (enfermedad o edad) para conocer la influencia de cada una sobre el total.

Mortalidad / Año	< 1 Año	Alisado Exponencial	Ajuste	Final
2012	11,10	11,10	0,00	110,10
2013	10,80	11,10	0,00	11,10
2014	10,60	10,88	-0,07	10,81
2015	9,70	10,67	-0,11	10,56
2016	9,70	9,94	-0,29	9,65
2017	9,30	9,76	-0,26	9,50
2018	9,13	9,45	-0,29	9,13

Tabla 4. Promedio de cada característica

[2] Russell & Taylor (2011). Operations Management Creating Value Along the Supply Chain, 7th ed. *Forecasting*.

Por consiguiente y con el afán de obtener valores más pequeños y representativos pero con la misma semántica, se realizó un cálculo del porcentaje de cada característica. Y en última instancia para reducir la brecha entre cada enfermedad o edad, se realizó el cálculo de una media móvil de dos en dos; resultando de esta manera estas dos tablas:

Mortalidad / Año	2013	2014	2015	2016	2017	2018	Promedio	%	Media móvil
Sistema circulatorio	94.099	92.190	96.252	101.928	97.219	98.578,66	96.711,11	38,15	38,15
Tumores	60.294	60.791	62.621	62.625	62.731	63.056,66	62.019,77	24,46	31,31
Sistema respiratorio	52.522	54.250	56.901	65.185	64.869	66.719,69	60.074,45	23,70	27,50
Infecciosas o parasitarias	14.282	14.534	13.565	13.897	14.198	14.125,99	14.100,33	5,56	16,53
Diabetes	8.045	8.201	9.223	9.599	8.893	9.136,65	8.849,61	3,49	10,01
Sistema urinario	10.966	11.664	12.698	13.135	12.897	9.136,65	11.749,44	4,63	7,32
Total	326.197	325.539	333.407	352.992	341.688	-	253.504,72	-	-

Tabla 5. Cálculo de porcentaje y media móvil por enfermedad

Mortalidad	2012	2013	2014	2015	2016	2017	2018	Promedio	%	Media móvil
< 1 Año	11,10	10,80	10,60	9,70	9,70	9,30	9,13	10,05	7,04	7,04
1 a 4	0,50	0,50	0,40	0,40	0,40	0,40	0,39	0,43	0,30	3,67
5 a 14	0,20	0,20	0,20	0,20	0,20	0,20	0,20	0,20	0,14	1,90
15 a 24	1,00	0,90	0,90	0,90	0,80	0,80	0,78	0,87	0,61	1,26
25 a 34	1,10	1,10	1,10	1,10	1,10	1,00	1,00	1,07	0,75	1,00
35 a 44	1,80	1,80	1,80	1,70	1,80	1,70	1,70	1,76	1,23	1,12
45 a 54	4,40	4,40	4,20	4,20	4,30	4,00	4,00	4,21	2,95	2,03
55 a 64	10,90	10,90	10,60	10,70	11,00	10,20	10,25	10,65	7,46	4,75
65 a 74	24,50	24,60	24,10	24,30	25,30	24,10	24,26	24,45	17,13	10,94
> 74	88,30	88,90	87,80	88,60	92,60	88,20	88,96	89,05	62,39	36,66
								142,74		

Tabla 6. Cálculo de porcentaje y media móvil por edad

Finalmente, para culminar con un cálculo estimativo de una razonable fecha de cobertura, se genera una fórmula compuesta de la siguiente manera:

$$y_{DIAS} = K[(W_A \times F_A) + (W_E \times F_E)]$$

Donde, y es la cantidad de días a partir de los cuales se recomienda comenzar la fecha de cobertura, K es una constante de seguro adicional, W_A es el peso correspondiente a los años del potencial asociado, F_A es el valor de la media móvil correspondiente a las tablas de referencia anteriores acorde a la edad del asociado, W_E es el peso correspondiente a la enfermedad del potencial asociado, F_E es el valor de la media móvil correspondiente a las tablas de referencia anteriores acorde a la enfermedad del asociado.

Se determinaron los siguientes valores de pesos: $W_A = 1,25$; $W_E = 1,5$. Esto se debió a que, como se puede apreciar en los valores arrojados en las tablas anteriores, la cantidad de defunciones es mayor en proporción a la enfermedad de los pacientes y no así a la edad de cada uno. Y en base a una puesta en común con el cliente y las variables que arrojaban valores más cercanos a la realidad, se decidió utilizar el siguiente valor de constante: $K = 8$.

2.2.4 Comprobantes, saldos y pagos

Otro de los procesos claves de la organización es el estado de cuenta y generación de comprobantes, ya que es un conjunto de actividades que forman parte de los ingresos de dinero a la empresa.

Cada socio, es decir, el titular de un plan mantendrá una variable denominada “saldo” donde se decrementará cada vez que se genere un nuevo comprobante e incrementará cada vez que el socio realice un pago. Esta variable no afecta a los adherentes, ya que solo se necesita tener relación con el titular de un plan.

Dicha información es una métrica que el usuario podrá consumir viendo el estado de cuenta de un titular y podrá tomar determinaciones al respecto.

Mensualmente, o cuando el usuario disponga, se generará el listado de comprobantes a través de un proceso manual. Esta actividad construirá un archivo en formato *.pdf donde se observarán todos los comprobantes de todos los socios activos. El monto del mismo quedará determinado por la tarifa que el mismo tenga relacionada y es acá donde se verifica el rango de edades de los adherentes descrito en el proceso de negocio anterior.

Luego el usuario podrá imprimir dicha información o simplemente recortar el comprobante que desee y enviarlo por algún canal electrónico al cliente.

El sistema anterior era demasiado acotado en este proceso, es decir, el usuario para realizar la impresión de los comprobantes de los usuarios activos primero debía imprimir en una impresora antigua el listado de los titulares y luego ingresar ese rango por pantalla para imprimir los comprobantes. Con la nueva funcionalidad, solo se necesita ejecutar una acción, ganando eficiencia, flexibilidad y reduciendo el coste notablemente, ya que el papel que la impresora demandaba ya no existe en el mercado y era muy costoso conseguirlo. Ahora el mismo podrá imprimir la información con impresoras láser y modernas o simplemente enviarla por email para contribuir al cuidado del medio ambiente.

Para informar el pago, el usuario deberá seleccionar el socio e informar que comprobante es pagado. Además, es posible consultar los últimos comprobantes generados e informar el pago directamente con un click. Una vez que se registre el pago, se deberá indicar el *cobrador* que realizó el mismo con el objetivo de obtener métricas y poblar de información la sección de estadísticas para tomar acciones a futuro que aporten nuevo valor a la organización.

2.3 Procesos de negocio secundarios

2.3.1 Obras sociales

Al dar de alta un usuario titular de un plan o un adherente, se requiere agregar la información de que si el mismo está cubierto o no por una obra social.

El usuario final del sistema tiene la posibilidad de dar de alta una serie de obras sociales para luego seleccionar las mismas a través de un componente y asignarle así la relación con un nuevo cliente. Esta información deberá ser cargada por el administrador del sistema, es decir, el usuario final del mismo, con el objetivo de darle flexibilidad y escalabilidad a este módulo, cargando solamente la información más comúnmente usada, existiendo la posibilidad de poblar con nueva información en un futuro.

Uno de los puntos más relevantes a la hora de dar de alta una obra social es verificar si la misma cuenta con seguro de sepelio o si el plan del cliente tiene dicho seguro, ya que esta información es importante para verificar si un posible sepelio está cubierto o no y evitar temas económicos con los familiar en tal situación.

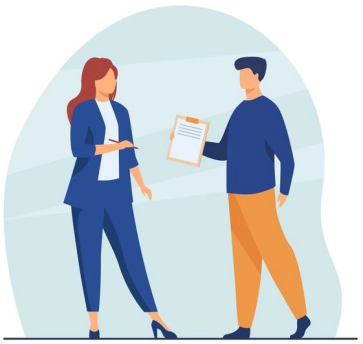
2.3.2 Zonas y cobradores

Una forma de segmentar los clientes es dividir los mismos por zonas, con el objetivo de crear grupos para obtener métricas. Cada zona tiene asignado uno o más cobradores, donde estos son los encargados de generar el cobro a cada uno de los socios afectados por dicha zona. Además, es posible que un cobrador esté asignado a más de una zona.

Gracias a este proceso de negocio, se facilita la lectura de la información de los movimientos contables, sabiendo cuánto ingreso genera un determinado cobrador o cuanto en una zona y poder tomar medidas al respecto, ya sea, bonificar

al cobrador que más ingresos generó en tal mes o segmentar una zona en dos para descongestionar la misma.

2.3.3 Estados de un cliente



Al dar de alta un cliente, el sistema propone una fecha de cobertura siguiendo el algoritmo de predicción descrito anteriormente. El primer estado que se le asigna a un cliente al ingresar al sistema es “Alta”, independientemente de que si tiene o no cobertura inmediata, ya que este punto no indica un cambio de estado.

Luego, un cliente puede pasar a ser dado de “Baja” por tres motivos:

1. Renuncia: Es cuando el socio indica que quiere dejar de abonar su cuota mensual. En este punto se le consulta si los adherentes quieren seguir con la cobertura o si quieren dar de baja el grupo familiar, si existiese.

2. Fallecimiento: Similar al punto anterior. Se le consulta a los adherentes si quieren seguir abonando su plan o no.

3. Morosidad: Si un socio no abona su cuota por tres meses, es decir, no se informó un pago de los últimos tres comprobantes, el sistema propone la baja del mismo permitiendo al usuario final aceptar la opción, ya que es necesario esa flexibilidad y no dar de baja el cliente directamente.

2.3.4 Servicios

Además de la administración y comercialización de planes para servicios de sepelio, otro de los procesos que impactan directamente en los flujos monetarios de la organización es la venta de servicios.

Esta entidad abarca a aquel potencial cliente que no está asociado a ningún plan, es decir, no tiene relación alguna con la organización pero se le presentó la

urgencia de utilizar uno de sus servicios. Por lo tanto, la empresa también ofrece la comercialización de los servicios que ya están incluidos para un cliente asociado.

A continuación, se detallan los tipos de servicios actualmente disponibles en la organización:

1. Servicio completo. Incluye: ataúd de calidad media, sala velatoria con azafata y capilla ardiente, traslado con coche fúnebre y porta corona al cementerio, coche acompañamiento, permiso municipal para inhumación.
2. Servicio semicompleto. Incluye: Ataúd de calidad baja, sala velatoria con azafata y capilla ardiente, traslado con coche fúnebre y porta corona al cementerio, permiso municipal para inhumación.
3. Servicio básico. Incluye: Ataúd de calidad baja, sala velatoria con azafata, traslado con coche fúnebre y permiso municipal para inhumación.

Asimismo, es posible la venta de cada uno de los elementos que componen los servicios. Es decir, es posible comercializar un traslado o un ataúd en cualquiera de sus calidades.

Cabe destacar que existen dos tipos de traslado:

1. Traslado al cementerio: Prestación incluída en todos los servicios.
2. Traslado del fallecido: Prestación extra que se comercializa por kilómetro recorrido. Un ejemplo es cuando una persona fallece en un determinado punto geográfico y es necesario el traslado a su ciudad natal para la inhumación.

3. Conceptos de la Ingeniería de Software

Con el fin de introducir a la ejecución del presente proyecto, se describen a continuación, conceptos que provienen de la ingeniería de software que fueron aplicados para materializar con éxito un producto de calidad, investigando y resolviendo los diferentes aspectos para una mejor solución del problema.

3.1 Proceso de desarrollo

Es necesario definir un proceso ya que es fundamental organizar y estructurar las actividades, además, mejora la calidad del software y aumenta la velocidad con que se desarrolla, definiendo un enfoque. Por lo tanto, se logrará un marco de trabajo ganando estabilidad, organización y control en todo el ciclo de desarrollo.

En primer lugar, se detallan los puntos teóricos para seleccionar una metodología de desarrollo que se adecue a nuestro proyecto.

Standish Group[3] es una firma internacional independiente que se dedica a la investigación IT donde uno de sus principales focos es la generación de informes sobre proyectos de implementación de sistemas de información. Uno de sus informes describe en porcentajes, porque los proyectos fallan o son exitosos.

Estos informes nos indican que es importante hacer foco en la especificación de los requerimientos, detallando y refinando cada punto, además de involucrar al cliente en todas las etapas, de manera que el mismo sea un partícipe de todo el ciclo de desarrollo con la finalidad de evitar cometer los errores causados por malentendidos.

[3] The Standish Group: https://www.standishgroup.com/

A continuación, se detalla la información obtenida del Chaos Report.

¿Por qué fallan los proyectos?

Motivo	Porcentaje (%)
Requerimientos incompletos	13,10
Pobre inclusión de los usuarios	12,40
Planificación y estrategia	10,60
Expectativas no realistas	9,90
Falta de soporte gerencial	9,30
Requerimientos cambiantes	8,70
Recursos insuficientes	8,10
Requerimientos dejan de ser necesarios	7,50
Pobre manejo de IT	6,20

Tabla 7. Motivos de falla de los proyectos

¿Por qué los proyectos son exitosos?

Motivo	Porcentaje (%)
Usuario involucrados	16,00
Soporte gerencial	14,00
Requerimientos claros	13,00
Planeamientos apropiado	10,00
Expectativas realistas	8,00
Hitos/objetivos pequeños	7,70
Recursos apropiados	7,20
Metodología y estrategia	5,30
Visión y objetivos claros	3,00

Tabla 8. Motivos de éxito de los proyectos

3.2 Metodología ágil

El desarrollo ágil de software es un concepto sumamente demandado en la industria en los últimos tiempos, teniendo como objetivo entregar al cliente un producto o parte del mismo con funcionalidades desarrolladas de manera rápida y eficiente.

En este proyecto se utiliza un enfoque iterativo e incremental[4], con amplia comunicación con el cliente, generando una mayor satisfacción en el mismo, ya que vive en tiempo real todo el proceso. Con esto se logra poder abarcar las necesidades prioritarias que tenga el cliente y entregarle un conjunto de funciones de un producto que atienden dichas necesidades en un lapso de tiempo corto, sin perder calidad en el software. Por lo tanto, se mejora la calidad, ya que fomenta el enfoque proactivo de cada desarrollador buscando la excelencia en el producto.

Además, la motivación de los trabajadores aumenta ya que generalmente son equipos autogestionados, donde se le da mucho foco a la capacidad creativa e innovadora, sumado a que se apunta mucho al trabajo colaborativo.

Las métricas que se consiguen en cuestión de tiempos, costos y rendimiento son más reales que en proyectos tradicionales, debido a la división de pequeños equipos y fases donde se ve con más consistencia lo que se está realizando. Por lo tanto, se mejora el control y la capacidad de predicción, reduciendo así los costos en el proyecto.[5]

[4] Roger S. Pressman (2011). Ingeniería del software: un enfoque práctico. *Modelos de procesos.*
[5] Wesley Clark (2020). Metodología ágil.

3.3 Aplicación web

Este tipo de software se desarrolla siguiendo un conjunto de lenguajes y estándares que lo hace soportable por navegadores de cualquier dispositivo que corra un sistema operativo moderno. La gran ventaja de esto es que cualquier usuario en cualquier parte del mundo podrá utilizar la aplicación y tener acceso a los datos del sistema sin necesidad de tener que instalar un programa o aplicación.

3.4 Cobertura de código

La cobertura de código es una métrica representada porcentualmente que entrega información acerca de cuánto código fuente ha sido comprobado a través de pruebas unitarias. Por lo tanto, una línea de código es cubierta si dicha línea ha sido ejecutada en la realización de un test. Si el porcentaje es relativamente alto, se puede tener una buena métrica del correcto funcionamiento de la aplicación.

Además, este proceso es útil para detectar y eliminar código innecesario, ya que es un código que no se ejecuta.

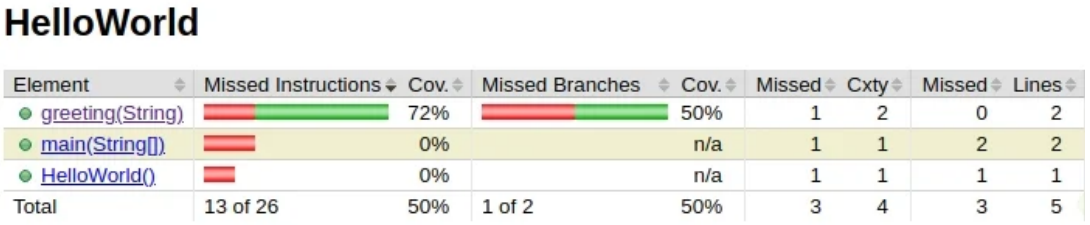
Una cobertura del 100% no garantiza la ausencia de errores o bugs, esto simplemente nos indica que el código supera las pruebas en su totalidad.

JACOCO[6] es una herramienta que analiza la cobertura de código y arroja un reporte de cuanto código está cubierto por test unitarios, sumado a la gran facilidad de lectura y de evitar tener que hacer un seguimiento de lo que se está probando.

Esta herramienta arroja un informe en formato html donde se puede navegar fácilmente entre paquetes, clases y métodos.

El reporte es el siguiente:

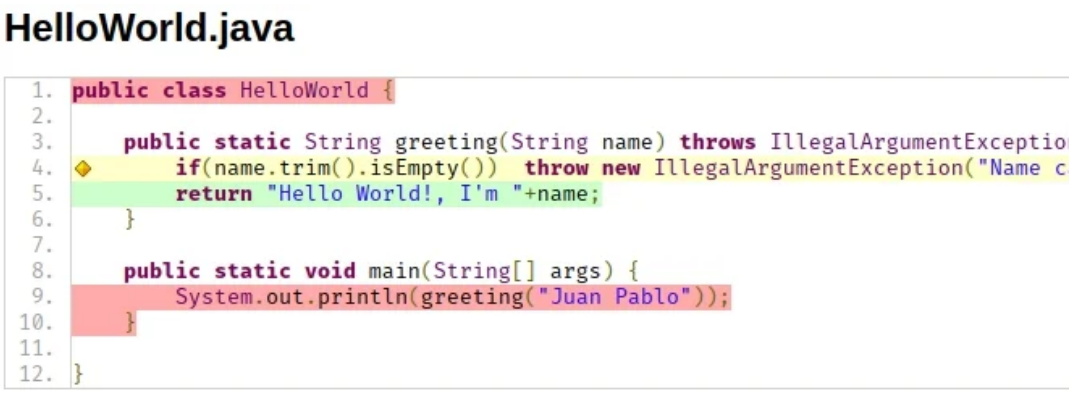
[6] Jacoco. https://www.eclemma.org/jacoco/



HelloWorld

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines
greeting(String)		72%		50%	1	2	0	2
main(String[])		0%		n/a	1	1	2	2
HelloWorld()		0%		n/a	1	1	1	1
Total	13 of 26	50%	1 of 2	50%	3	4	3	5

Figura 2. Interfaz librería Jacoco



HelloWorld.java

```
1.  public class HelloWorld {
2.
3.      public static String greeting(String name) throws IllegalArgumentExceptio
4.          if(name.trim().isEmpty())  throw new IllegalArgumentException("Name c
5.          return "Hello World!, I'm "+name;
6.      }
7.
8.      public static void main(String[] args) {
9.          System.out.println(greeting("Juan Pablo"));
10.     }
11.
12. }
```

Figura 3. Ejemplo de código cubierto

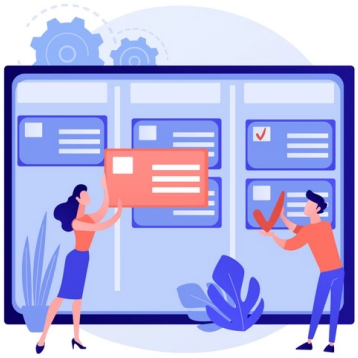
La herramienta permite acceder hasta llegar a los métodos donde se obtienen las métricas del código cubierto y del que faltó cubrir. Ingresando por paquete y llegando a una clase, se puede visualizar el porcentaje cubierto del código correspondiente, mostrando en rojo lo que falta testear y en verde lo que se cubrió correctamente. Idealmente en un proyecto de este tamaño se busca cubrir un 80% del código.

En conclusión, JaCoCo es de fácil integración, para todo el ciclo de vida de un proyecto y muy poco intrusivo; resultando en una herramienta útil para mejorar la calidad del producto.

3.5 Casos de uso

Existen diferentes metodologías para desarrollar software, cada una enfocada en las necesidades que se tengan. También existen diferentes etapas pero, independientemente de la metodología que se esté utilizando, la comunicación

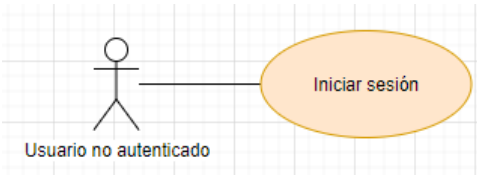
con el cliente y el relevamiento de los requerimientos siempre está presente, ya que es fundamental para definir las funcionalidades que se esperan del producto.



Existen numerosas formas de representar estas funcionalidades, cada una con una perspectiva distinta. En el diagrama de casos de uso[7] se manifiesta qué hará el sistema pero no cómo lo hará. Es parte del conjunto de diagramas de comportamiento, ya que se describen las acciones que el sistema debe manejar desde el punto de vista del usuario.

Los diagramas de caso de uso modelan la funcionalidad del sistema utilizando actores y casos de uso. Los actores son disparadores de acciones del sistema, en cambio, los casos de uso, son servicios o funciones provistas por el sistema. Por lo tanto, son los actores quienes consumen las funciones que se ofrecen en los casos de uso.

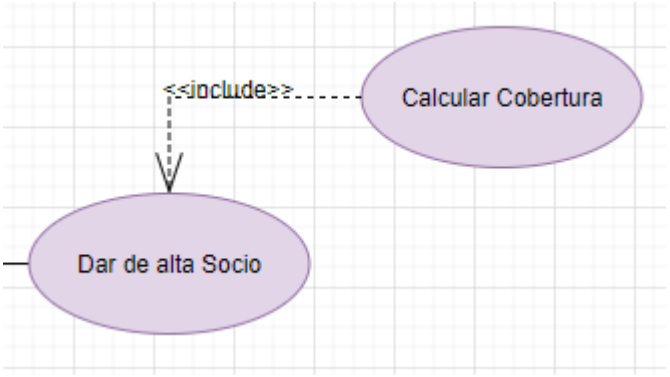
Si bien es posible delimitar el sistema a través de un recuadro, a continuación se muestra un ejemplo de un actor con una funcionalidad básica del sistema: Iniciar sesión.



Como se puede observar, el caso de uso (funcionalidad) se representa con un óvalo, con un nombre descriptivo pero no muy largo. El objetivo es que describa la acción.

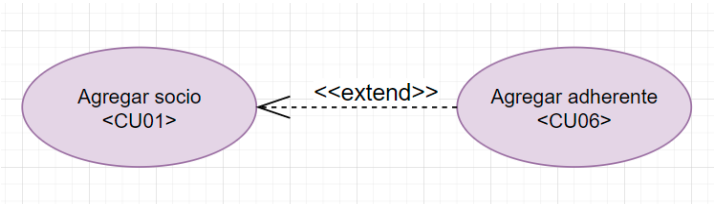
El actor es quien ejecuta ese caso de uso. La relación se representa a través de una línea que une ambos componentes. Además, es posible generar relaciones entre casos de uso. Estas líneas deben estar etiquetadas con *"include"* o *"extend"* . Para el primer punto, es cuando el caso de uso que incluye al otro, se ejecuta siempre que el primero ocurra. En cambio, cuando un caso de uso se extiende de otro, puede que se ejecute como no.

[7] Roger S. Pressman (2011). Ingeniería del software: un enfoque práctico. *Compresión de los requerimientos.*



En este ejemplo, se observa cómo la funcionalidad de calcular cobertura se ejecuta siempre a la hora de dar de alta un socio, ya que es una operación que se requiere siempre que se necesite registrar un nuevo socio al sistema.

En cambio, al dar de alta un nuevo socio, el usuario puede optar por dar de alta los adherentes en ese momento o ejecutar dicha operación en un futuro.



A continuación, se muestra la plantilla utilizada para la definición de casos de uso:

Nombre		**ID**	CU#
Descripción			
Actores	➔		
Prioridad	◉ Alta ◎ Media ◎ Baja	**Complejidad**	◎ Alta ◉ Media ◎ Baja
Precondiciones	➔		
Postcondiciones	Éxito ➔	Fracaso ➔	
Flujo Principal		**Flujo Alternativo**	
Flujo de excepciones	➔		
Asociaciones de inclusión	➔		
Asociaciones de extensión	➔		
Requerimientos especiales	➔		
Observaciones			

Figura 4. Plantilla de especificación de caso de uso

3.6 Versionado

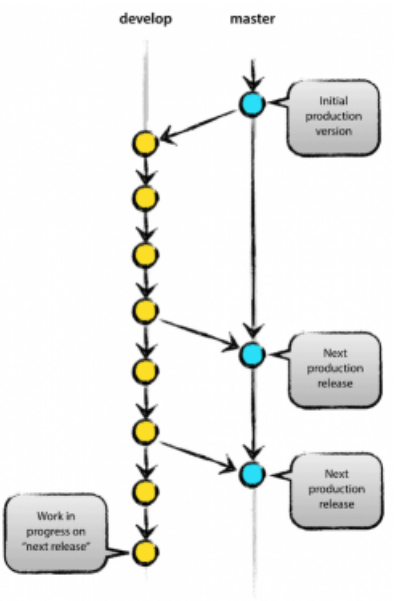
Apuntando a aumentar la velocidad de entrega y disminuir los errores humanos en el manejo de ramas, se decidió que, como sistema de control de versiones[8], se utilice GIT ya que dicha herramienta se adapta completamente a la metodología y complejidad del proyecto.

A continuación se enumeran los diferentes conceptos a utilizar en el siguiente apartado:

1. Rama/branch: Es simplemente un apuntador a una versión de código que se puede ir modificando por cada uno de los desarrolladores. Por lo general, la rama “master” es la inicial y a partir de allí nacen todas las ramificaciones.

2. Commit: Confirmación de un conjunto de cambios hechos por un desarrollador. Posee una estructura especial que se explicará más adelante.

Para que el flujo de trabajo funcione correctamente, se iniciaron en principio, las dos ramas principales.

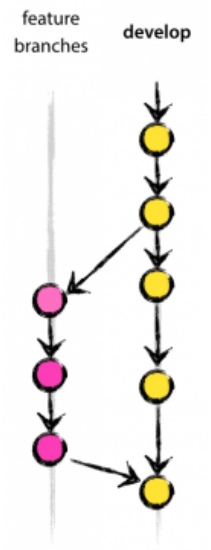


La rama **master** debe contener una versión estable, es decir, todo el conjunto de commits que están preparados para subir a producción. Por lo tanto, cada vez que se incorpora un nuevo commit modificando la versión del producto, se obtendrá un nuevo paquete listo para desplegar en producción.

En la rama **develop** se tiene el código donde se encuentra todo el conjunto de funcionalidades que cada desarrollador va aportando y es donde se localiza el conjunto de commits que conformarán la próxima versión planificada.

[8] Roger S. Pressman (2011). Ingeniería del software: un enfoque práctico. *Control de versión.*

Además de las ramas ya detalladas, para una mejor identificación y un mejor control del repositorio se generan las siguiente ramas:



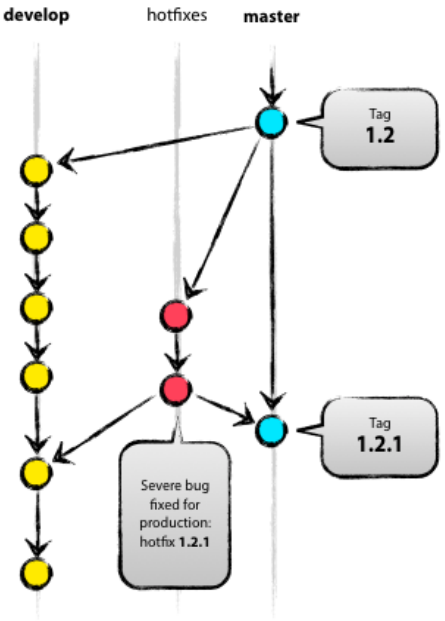
Feature: cada nueva funcionalidad, un nuevo ABM, una nueva configuración en el código, una nueva interfaz del usuario o el agregado de un campo de texto deberá manejarse a través de estas ramas.

El flujo se origina siempre desde la rama **develop** donde la nueva rama tendrá el prefijo **feature** seguido de la característica en sí que tendrá el desarrollo: **feature/alta-cobrador.**

Luego de que se complete el desarrollo, se incorpora el conjunto de commits nuevamente a la rama develop.

Una vez que dicha funcionalidad fue desplegada en producción, se elimina la rama para mantener el orden en el repositorio.

Hotfix: cada vez que se presente un error o bug en producción se deberá generar una rama de este tipo partiendo desde la rama master.



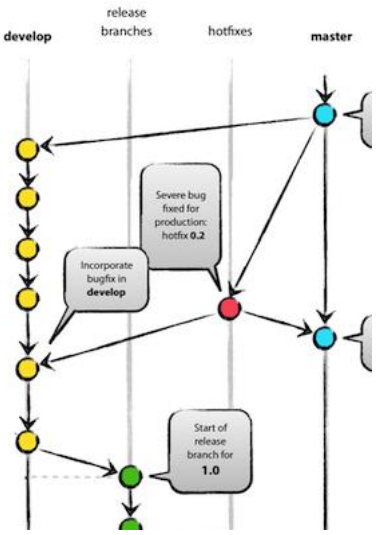
La estructura del nombre que tendrá será la siguiente: **hotfix/error-alta-prestador**, es decir, el prefijo hotfix sumado a una descripción del bug que se presentó.

Luego de que se solucione y la prueba local sea satisfactoria, se incorpora dicho conjunto de *commits* a la rama master nuevamente para que la solución sea desplegada en producción cuanto antes y evitar así generar problemas más grandes. Finalmente se elimina la rama anteriormente creada.

Notar que un hotfix es consecuencia de un bug, es decir, un error en el sistema ya desplegado y en uso por el cliente, este tipo de arreglos generalmente son de urgencia y los que más se deberían evitar.

Una vez que se logra una versión estable en la rama de desarrollo con las funcionalidades del incremento desarrolladas, se deberá armar una rama Release.

Release: rama con el conjunto de commits de una versión preliminar preparada en un principio para salir a producción. En esta rama es posible detectar defectos que deben ser solucionados antes de incorporar a la rama master.



Una vez realizado el testing por todo el equipo de todos los circuitos que componen este incremento, se mergea a la rama master y se obtiene una nueva versión del producto.

Para una mejor organización del producto que se va liberando al cliente, se optó por utilizar un sistema de versionado semántico de la siguiente manera:

Se partió de un entregable con una versión que comienza con 0.0.1 en la rama master, indicando en la posición de **major** que no es un producto preparado para entregar al cliente.

Luego de que se comienza con el desarrollo, es decir, se libera la rama **develop** el producto itera su posición **major** y se reinicia el resto de las posiciones: *1.0.0*, además de agregar el tag final **SNAPSHOT**, indicando que es una versión en desarrollo.

Resulta entonces: *1.0.0-SNAPSHOT*.

Cada vez que se genere una rama de **feature/***, no se itera la versión del producto en sí, tratando de conservar la versión del producto en las posiciones de **major, minor y patch** la misma hasta liberar una versión estable.

Una vez que se tiene una versión potencial para salir a producción, se genera la rama **release/***. En esta instancia, el producto deja de estar en desarrollo y se le cambia el tag final por el de **release candidate**: *1.0.0-RC1*. Como esta versión del producto puede contener defectos, como la versión de desarrollo está al mismo nivel, cada vez que se desee arreglar uno solamente se genera el cambio en la rama de **develop** y se mergea nuevamente a la rama de release, iterando el RC: *1.0.0-RC2*.

Al pasar la etapa de testing por todo el equipo, se libera la versión *1.0.0* a **master**.

Cada vez que se necesite generar un hotfix, se generará la rama desde master y se iterará la última posición de la versión, es decir, el **patch**: *1.0.1.*

Luego de tener la versión del producto 1.0.0 desplegada en producción, se itera la versión de **develop** para dejarla preparada para la nueva versión del producto: *1.1.0-SNAPSHOT.*

Por lo tanto y siguiendo el flujo, la próxima versión del producto que saldrá a producción será la versión *1.1.0*.

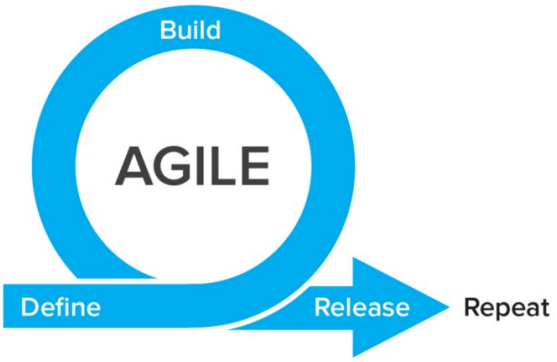
4. Documentación del proyecto

Se presenta a continuación el detalle correspondiente a cada una de las diferentes fases del proceso de elaboración del software.

4.1 Metodología de desarrollo

La metodología utilizada contiene en gran parte el enfoque ágil, siempre apuntando a entregar funcionalidades al cliente lo más pronto posible y poder tener una retroalimentación en corto tiempo.

Basándonos en el Chaos Report explicado anteriormente, todo el proceso quedó determinado haciendo hincapié en la calidad de los requerimientos, es decir, generar un fuerte foco en la etapa de análisis para llegar al diseño con requerimientos claros y concisos, optando por llevar a cabo muchas entrevistas con el cliente, incluyendo a este en todo el proceso.



Al comienzo de cada incremento, se realiza una reunión de planificación donde se dividen los casos de uso a desarrollar. Esta actividad es denominada *planning.* Durante la etapa de planificación se decidió organizar los requerimientos en tres iteraciones, por lo tanto, el equipo de desarrolladores se reunió en tres ocasiones para diagramar y dividir el conjunto de funcionalidades a cargo de cada desarrollador para ser implementadas en dicho incremento.

Además, los desarrolladores participaban de reuniones diarias para contar sus avances, despejar dudas y resolver problemas que pueden existir y generar trabas. Participando así de la ceremonia conocida como *daily*.

Luego, al finalizar cada incremento, es decir, las seis etapas a detallar a continuación, el equipo se reúne con el cliente para corroborar que la funcionalidad aplicada sea correcta y obtener una retroalimentación.

Por último, los integrantes del equipo discuten el resultado del finalizado sprint, analizando el funcionamiento del equipo y el cumplimiento de lo pactado. Este tipo de reuniones es conocido como *retrospectiva*.

Además, cabe aclarar que fue necesario la implementación de un incremento 0 no planificado, donde se realizaron actividades primordiales para poder comenzar a desarrollar las funcionalidades, entre estas se encuentran: relevamiento de requerimientos, refinamiento y especificación de casos de uso, diagrama de casos de uso, modelo de datos, diagrama de entidad relación y primera versión de la arquitectura.

4.2 Planificación de incrementos

A continuación, se presentan las estimaciones realizadas en la etapa de planificación. En principio, todas las horas estimadas para las etapas de análisis y diseño de cada uno de los incrementos planificados, fueron dedicadas a utilizarse al incremento 0.

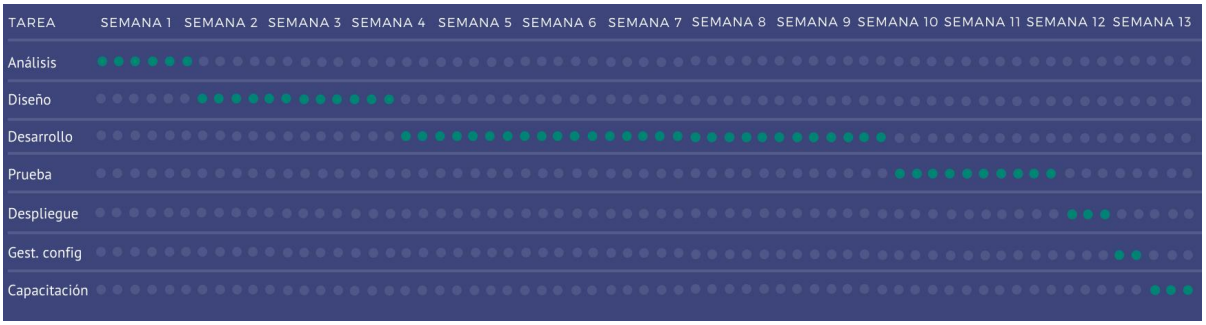


Figura 6. Estimación incremento 1

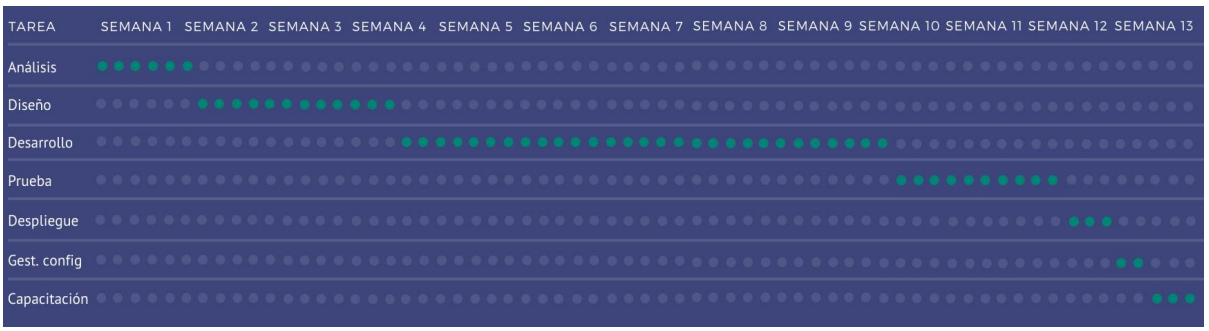


Figura 7. Estimación incremento 2

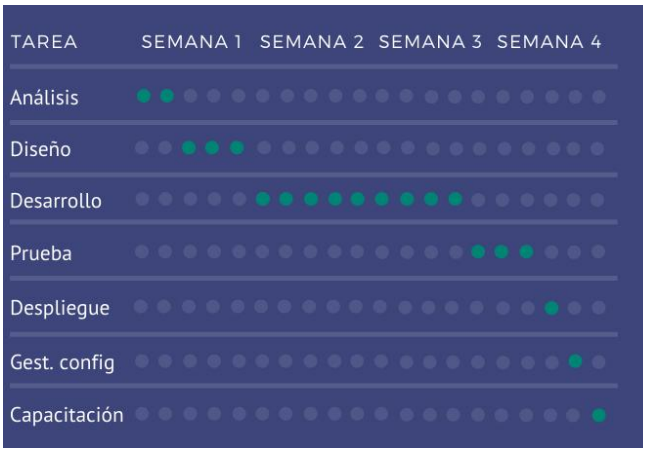


Figura 8. Estimación incremento

Incremento 0

Como se detalló con anterioridad, todas las actividades de las etapas de análisis y diseño fueron realizadas en este incremento, ya que se optó por generar una lista de requerimientos temprana para poder construir un modelo de datos consistente y los diferentes diagramas junto con la arquitectura antes de comenzar el desarrollo del producto.

Por lo tanto, en dicho incremento se estimó un total de 41 días en lo cual se produjo un corrimiento del 120% concretando dichas tareas en 90 días (18 semanas contando días laborales), ya que el desconocimiento del negocio fue mayor al esperado, esto abarca la coordinación de entrevistas con el cliente y la obsoleta tecnología del sistema anterior.

A continuación, se detallan las tareas realizadas en el incremento 0:

	Semanas																	
Tarea	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**
Relevamiento de requerimientos																		
Refinamiento de requerimientos																		
Diagrama de casos de uso																		
Especificación de casos de uso																		
Modelo de datos																		
Diagrama de entidad-relación																		
Arquitectura																		

Tabla 11. Registro de tareas del incremento 0

En varias de las tareas hubo una fuerte relación por lo cual no fue posible realizarlas en paralelo. Además, en una primera instancia se había indicado que cada desarrollador trabajaría 5 horas al día, tiempo que se vio afectado por cuestiones laborales que surgieron durante la construcción del producto. Por lo tanto, cada desarrollador dedicó en promedio un total de 3 horas por día, concretando así un total de 450 horas para el incremento 0.

En el resto de los incrementos, se destinó la estimación planificada en la etapa de desarrollo, prueba, despliegue, gestión de la configuración y capacitación, ya que se desarrollaron componentes puntuales del producto, con su posterior despliegue y validación del cliente, iniciando un proceso de capacitación al mismo sobre el conjunto de funcionalidades desarrolladas en dicho incremento.

Incremento 1

Luego de generar la primera versión de cada uno de los componentes del incremento 0, se comenzó con el desarrollo del código, esto abarca instalación de dependencias, configuración de entornos de desarrollo, gestión del sistema de versionado, entre otras.

En dicho incremento, se estimaron un total de 47 días en el cual por cuestiones ya detalladas anteriormente se generó un corrimiento del 100% consumiendo un total de 94 días, es decir, 19 semanas.

A continuación, se detallan los requerimientos desarrollados y el tiempo dedicado:

	Semanas																	
Tarea	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Req. Funcional 1																			
Req. Funcional 2																			
Req. Funcional 3																			
Req. Funcional 4																			
Req. Funcional 5																			
Req. Funcional 6																			

Tabla 12. Registro de tareas del incremento 1

En dicho incremento se desarrollaron los módulos que abarcan gran parte de los procesos principales, apuntando siempre a entregar una primera versión del producto con el mayor valor posible para el cliente.

Incremento 2

En el incremento 2 se continuó con el desarrollo del producto. En la planificación se había estimado 47 días la cual sufrió un corrimiento del 90% presentando el conjunto de funcionalidades en un total de 89 días, es decir, 18 semanas. En la siguiente tabla se detalla cada uno de los requerimientos:

	Semanas																
Tarea	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Req. Funcional 7																		
Req. Funcional 10																		
Req. Funcional 13																		
Req. Funcional 14																		
Req. Funcional 16																		
Req. Funcional 24																		

Tabla 13. Registro de tareas del incremento 2

En este incremento se completaron los requerimientos de los procesos de negocio principales y en gran parte los flujos del negocio tildados como secundarios.

Incremento 3

La estimación para este incremento en la planificación fue de 15 días, donde se presentó un desplazamiento del 110% completando las tareas en un total de 32 días, es decir, aproximadamente 7 semanas.

En la siguiente tabla, se presenta en detalle las tareas del incremento 3:

	Semanas						
Tarea	1	2	3	4	5	6	7
Req. Funcional 17							
Req. Funcional 21							
Req. Funcional 22							
Req. Funcional 25							
Req. Funcional 26							

Tabla 14. Registro de tareas del incremento 3

A continuación se describen cada una de las etapas que se llevaron a cabo de todo el ciclo. Como se describió con anterioridad, gran parte de la etapa de

análisis y diseño fue desarrollada en el incremento 0, pero igualmente en el resto de los incrementos fue necesario alguna modificación, por ejemplo, un cambio presentado por el cliente o una mejora del modelo de datos. El resto de las etapas fueron completadas en los incrementos 1, 2 y 3.

4.3 Análisis

Las actividades que se realizaron en la etapa de análisis fueron el relevamiento y refinamiento de los requerimientos, información importante para poder diagramar el alcance del producto y definir las necesidades del cliente. A continuación se detallan ambas etapas:

4.3.1 Relevamiento

Para la captación de requerimientos, se realizó un conjunto de entrevistas con el cliente para poder lograr así que él mismo informe detalladamente cómo son los procesos de negocio de la organización y poder conocer cómo trabaja el negocio en cada una de sus aristas.

La comunicación con el cliente fue a través de reuniones virtuales y, en gran parte, asistencia a las oficinas de la organización ya que algunos procesos muy puntuales requerían una explicación más clara y detallada. Para refinar estos detalles, también asistían empleados de la empresa que conocen y trabajan a menudo estos procesos.

Otro componente clave para el relevamiento fue el proceso de revisión del sistema actual que maneja la organización. Leer, investigar y abstraer flujos de trabajo de este aplicativo de escritorio, fue clave para un relevamiento más limpio de los procesos de negocio.

Gracias a este proceso, se logró obtener un conjunto de requerimientos no funcionales o atributos de calidad que el usuario quería mejorar, tales como la

usabilidad, performance, seguridad y a su vez, también mejorar la mantenibilidad, escalabilidad y robustez, entre otros.

Figura 9. Información e interfaz del sistema anterior

4.3.2 Refinamiento

Luego del relevamiento y de estudiar los flujos a través del proceso de revisión del sistema actual, se llevó a cabo el refinamiento de la información obtenida, teniendo como resultado una lista de requerimientos funcionales y no funcionales que se detallan a continuación.

4.3.3 Requerimientos funcionales

ID	Requerimiento
RF1	El sistema permitirá agregar, actualizar, eliminar y consultar socios.
RF2	El sistema permitirá agregar, actualizar, eliminar y consultar adherentes. Cada adherente está asignado a un socio.
RF3	El sistema permitirá consultar planes.

RF4	El sistema permitirá agregar, eliminar y consultar tarifas. Información requerida para dar de alta un socio.
RF5	Para el cálculo de los valores del comprobante de cada asociado, el sistema permitirá calcular una tarifa bajo ciertas condiciones y generar los comprobantes de pago.
RF6	El sistema permitirá agregar, actualizar, eliminar y consultar cobradores.
RF7	El sistema permitirá agregar, eliminar y consultar zonas. A cada zona, se le asignará uno o varios cobradores.
RF10	Cuando se genere el alta de un socio, existirá un algoritmo para calcular el tiempo de comienzo de cobertura.
RF13	El sistema permitirá agregar, eliminar y consultar obras sociales.
RF14	El sistema permitirá llevar registro del saldo y comprobantes, permitiendo la visualización para cada uno de los clientes.
RF16	El sistema permitirá informar un pago modificando el estado del comprobante pagado y el saldo del socio.
RF17	El sistema generará un backup automáticamente todas las semanas en una hora específica.
RF21	El sistema permitirá generar el reporte de estado y cobertura de socios y adherentes.
RF22	El sistema permitirá generar un reporte de comprobantes emitidos y pagos realizados para el mes actual.
RF24	El sistema permitirá visualizar diferentes métricas: Mes con mayor recaudación, cantidad de socios activos, pagos faltantes en el mes, cantidad de socios morosos, cantidad de socios activos.
RF25	El sistema permitirá el alta, baja y consulta de servicios.
RF26	El sistema permitirá la comercialización de los servicios, es decir, generar una venta o dar de baja la misma.

Tabla 9. Requerimientos implementados

Además, como parte de las tareas de revisión de los procesos y del alcance de la solución, se resolvió junto al cliente que los siguientes requerimientos no sean llevados a cabo en el desarrollo, motivados por diferentes argumentos, los cuales se exponen posteriormente.

ID	Requerimiento
RF8	El sistema permitirá agregar, actualizar, eliminar y consultar defunciones.
RF9	Ante la defunción o baja de un socio, el sistema permitirá que un adherente pase a ocupar la posición de socio. Siempre y cuando se respeten las restricciones del plan.
RF11	El sistema permitirá realizar la impresión de los comprobantes de pago del mes de cada socio.
RF12	El sistema permitirá agregar, actualizar, eliminar y consultar sucursales.
RF15	El sistema permitirá informar el pago de un comprobante.
RF18	El sistema permitirá realizar búsquedas de clientes con diferentes filtros: estado, cobertura, edad, plan, tarifa, dni, apellido, nombre.
RF19	El sistema permitirá generar un reporte de defunción de un socio o adherente para información parroquial.

Tabla 10. Requerimientos no implementados

- El RF8 no se llevó a cabo ya que en la funcionalidad de dar de baja un socio es posible indicar que el mismo falleció interpretando un alta de una defunción.
- El RF9 no se desarrolló ya que se puede conseguir la misma funcionalidad a través del flujo de dar de alta un nuevo socio con cobertura inmediata, consiguiendo el mismo resultado de este requerimiento funcional.
- El RF11 se encuentra embebido dentro del RF5, es decir, en la generación de los comprobantes ya se genera la impresión.
- El RF12 se decidió no desarrollar ya que, si bien la empresa cuenta con diferentes sucursales, en diferentes entrevistas con el cliente, se constató que es mejor llevar toda la información centralizada con el objetivo de obtener mejores métricas a nivel macro y poder tomar decisiones como toda una organización.
- El RF15 fue fusionado con el RF16 logrando reducir las tareas consiguiendo las mismas funcionalidades.
- El RF18 ya fue incorporado en la funcionalidad de consulta de cada entidad nombrada en la descripción.

- El RF19 no fue llevado a cabo ya que es una información que podría pasarse mediante otros medios.

4.3.4 Requerimientos no funcionales

La siguiente lista se definió siguiendo el criterio de atributos de calidad propuestos en el apartado de Arquitectura.

ID	Requerimiento
RNF1	El sistema deberá permitir la posterior integración con una aplicación móvil.
RNF2	El sistema debe negar conexiones entrantes desde redes ajenas a la red local del cliente.
RNF3	El sistema no debe exceder los quince segundos en la generación de los reportes.
RNF4	El sistema debe ser lo suficientemente intuitivo para que el trámite de una alta de socio o adherente no exceda los 5 minutos, una baja los 2 minutos y una modificación 5 minutos.
RNF5	El sistema debe permitir realizar un cambio mínimo (cambio de interfaces) en menos de 5 horas laborales, un cambio medio (aquellos que requieran modificación de lógica) en menos de 7 días hábiles, y un cambio significativo (aquellos por encima de los anteriores) en menos de un mes.
RNF7	El sistema no debe perder disponibilidad en más del 1% de los casos.
RNF8	En el caso de que el sistema falle, debe reponerse en el 90% de los casos advirtiendo al usuario y sin intervenir en el flujo.
RNF9	El sistema debe permitir al usuario revertir acciones en el transcurrir de como mínimo 5 segundos en el 80% de los casos.
RNF10	El sistema debe ser capaz de funcionar sin problemas en el 100% de las plataformas y Sistemas Operativos del mercado.

Tabla 11. Requerimientos no funcionales implementados

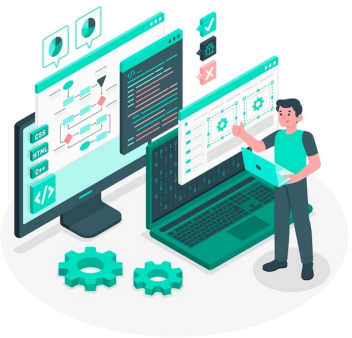
Además, por cuestiones de revisión del alcance del producto con el cliente, se decidió no llevar a cabo el RNF6, considerando que dicha funcionalidad no aportaba gran valor al producto.

ID	Requerimiento
RNF6	El sistema debe permitir al usuario la modificación de colores en los componentes principales, como Toolbars, backgrounds.

Tabla 12. Requerimientos no funcionales no implementados

4.4 Diseño

4.4.1 Requerimientos y casos de uso



Se llevó a cabo un proceso de diseño donde se diagramaron los diferentes requerimientos teniendo como resultado un conjunto de casos de uso con el alcance necesario para cumplir con los procesos de negocios principales y secundarios de la organización. Así mismo, cada caso de uso fue detallado y especificado siguiendo una plantilla previamente definida.

En dicha plantilla, explicada en la sección de CU se describió el funcionamiento que debería desprenderse del aplicativo para que cumpla con el requerimiento especificado.

La generación del diagrama de caso de uso fue una actividad única, realizada en el incremento 0 previamente detallado.

La serie de pasos que se utilizaron para generar un diagrama apuntando a la calidad, escalabilidad y modificabilidad, son los siguientes:

1. Identificación de los actores: Al comenzar toda actividad de caso de uso, es necesario identificar todos los actores encargados de disparar las funcionalidades del sistema.

2. Identificación de acciones: Las funcionalidades que se desean representar fueron descritas en los requerimientos funcionales, en este paso, las mismas

son llevadas al diagrama a través de óvalos y relacionadas con el actor que disparará la misma.

3. Priorización de funciones: Es necesario dividir este conjunto de requerimientos en las tres iteraciones identificadas en la etapa de planificación del proyecto.

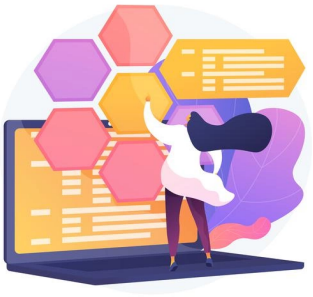
4. Especificación de caso de uso: Una vez identificadas las tareas a realizar, se deberá realizar el detalle de la especificación de los casos de uso que entren en la iteración siguiendo la plantilla previamente definida.

Luego, siguiendo cada especificación, el desarrollador tendrá todas las definiciones para construir la funcionalidad asignada.

Se adjunta el diagrama de casos de uso y la especificacion de cada uno en la sección de Anexos.

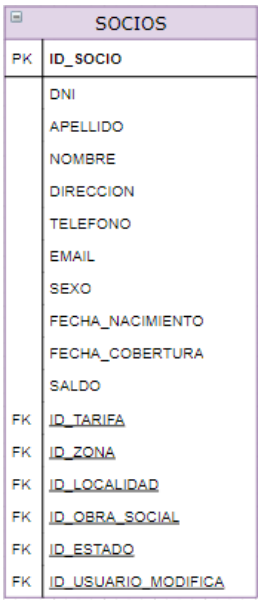
4.4.2 Modelo de datos

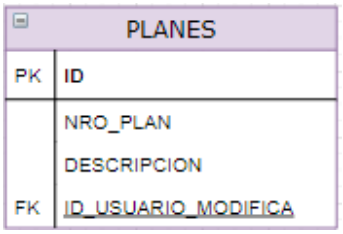
El modelo de datos es una forma de organizar y estructurar con la finalidad de introducir estos conceptos en la base de datos y lograr consistencia en la información.



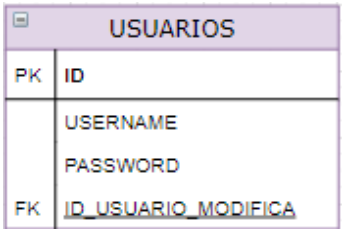
Una de las formas de representar esta información es a través de un diagrama de entidad-relación, o también conocido como “DER” donde se muestran las relaciones de las diferentes entidades (modelos) en un sistema. Son diagramas clave para ganar calidad en la base de datos y son muy utilizados en ingeniería de software. Además, por la naturaleza del proyecto y al utilizar una base de datos relacional, este tipo de diagrama se adapta completamente.

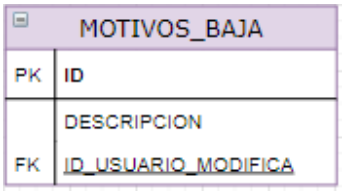
A continuación se describen los modelos utilizados y sus atributos relacionados:

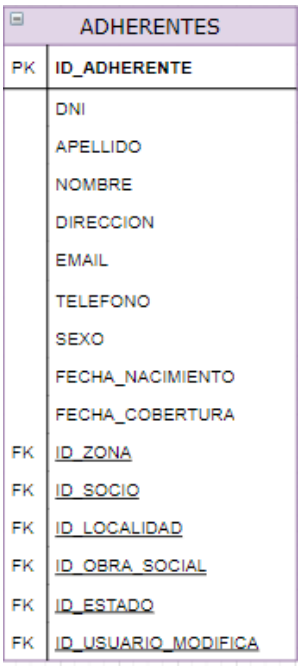

SOCIOS
PK ID_SOCIO
DNI
APELLIDO
NOMBRE
DIRECCION
TELEFONO
EMAIL
SEXO
FECHA_NACIMIENTO
FECHA_COBERTURA
SALDO
FK ID_TARIFA
FK ID_ZONA
FK ID_LOCALIDAD
FK ID_OBRA_SOCIAL
FK ID_ESTADO
FK ID_USUARIO_MODIFICA


PLANES
PK ID
NRO_PLAN
DESCRIPCION
FK ID_USUARIO_MODIFICA

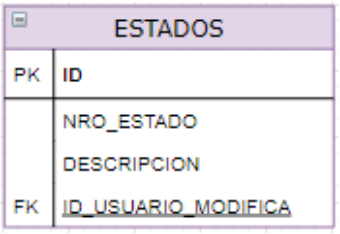

TARIFAS
PK ID
NRO_TARIFA
DESCRIPCION
VALOR
FK ID_PLAN
FK ID_USUARIO_MODIFICA

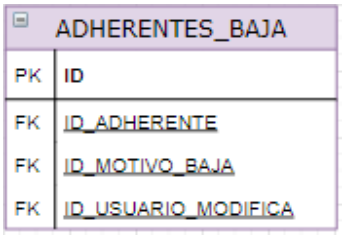

USUARIOS
PK ID
USERNAME
PASSWORD
FK ID_USUARIO_MODIFICA


MOTIVOS_BAJA
PK ID
DESCRIPCION
FK ID_USUARIO_MODIFICA

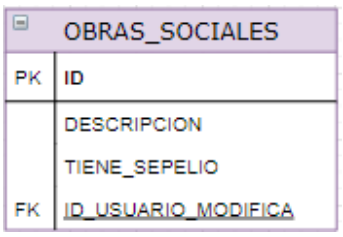

ADHERENTES
PK ID_ADHERENTE
DNI
APELLIDO
NOMBRE
DIRECCION
EMAIL
TELEFONO
SEXO
FECHA_NACIMIENTO
FECHA_COBERTURA
FK ID_ZONA
FK ID_SOCIO
FK ID_LOCALIDAD
FK ID_OBRA_SOCIAL
FK ID_ESTADO
FK ID_USUARIO_MODIFICA

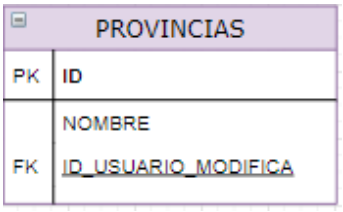

RANGOS_TARIFAS
PK ID
EDAD_DESDE
EDAD_HASTA
VALOR
FK ID_TARIFA
FK ID_USUARIO_MODIFICA


ESTADOS
PK ID
NRO_ESTADO
DESCRIPCION
FK ID_USUARIO_MODIFICA

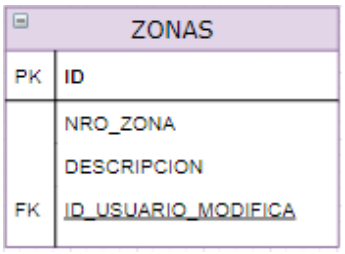

ADHERENTES_BAJA
PK ID
FK ID_ADHERENTE
FK ID_MOTIVO_BAJA
FK ID_USUARIO_MODIFICA

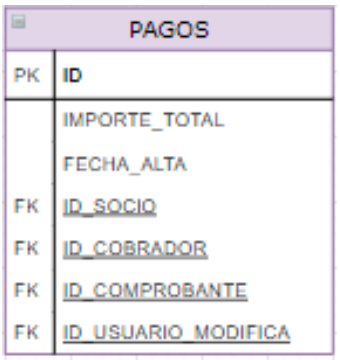

SOCIOS_BAJA
PK ID
FK ID_SOCIO
FK ID_MOTIVO_BAJA
FK ID_USUARIO_MODIFICA

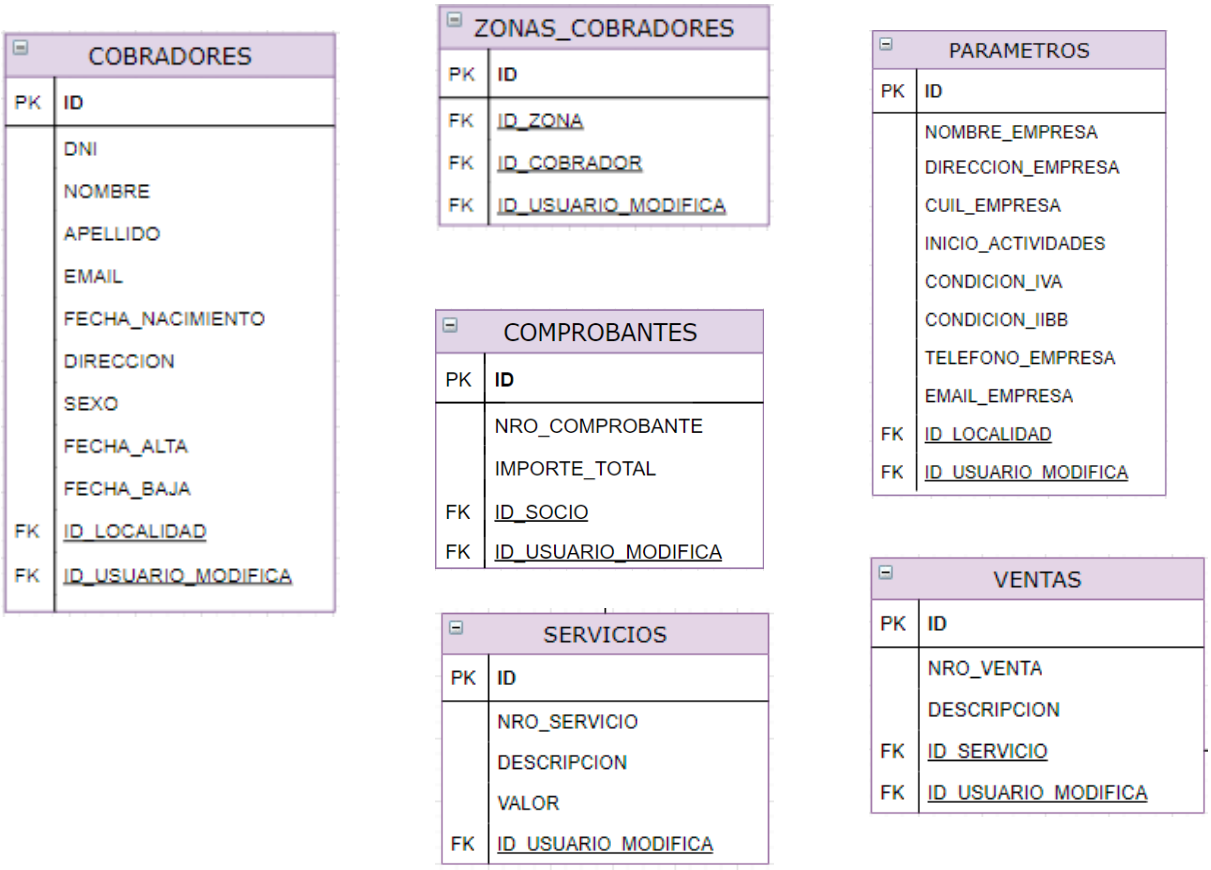

OBRAS_SOCIALES
PK ID
DESCRIPCION
TIENE_SEPELIO
FK ID_USUARIO_MODIFICA


PROVINCIAS
PK ID
NOMBRE
FK ID_USUARIO_MODIFICA


LOCALIDADES
PK ID
NRO_LOCALIDAD
NOMBRE
CODIGO_POSTAL
FK ID_PROVINCIA
FK ID_USUARIO_MODIFICA


ZONAS
PK ID
NRO_ZONA
DESCRIPCION
FK ID_USUARIO_MODIFICA


PAGOS
PK ID
IMPORTE_TOTAL
FECHA_ALTA
FK ID_SOCIO
FK ID_COBRADOR
FK ID_COMPROBANTE
FK ID_USUARIO_MODIFICA



En el diagrama de entidad-relación se puede ver detalladamente el modelo con todas sus relaciones.

4.4.3 Herramientas de documentación

Para el manejo de la documentación técnica más vinculada al código y principalmente para el backend, se utilizó Swagger.

Como se describió anteriormente, todo el backend está armado basado en servicios, es decir, una API que atiende solicitudes que el cliente (frontend) realiza. Para poder documentar toda esta información y llevar un documento ordenado y consistente, Swagger nos ofrece un conjunto de reglas, herramientas y especificaciones que nos

ayuda a cumplir con esto. Además, se logra escalabilidad en la documentación, modificabilidad y un documento actualizado constantemente.

A continuación, se puede observar como swagger[9] documenta y facilita información a través de una interfaz de usuario amigable:

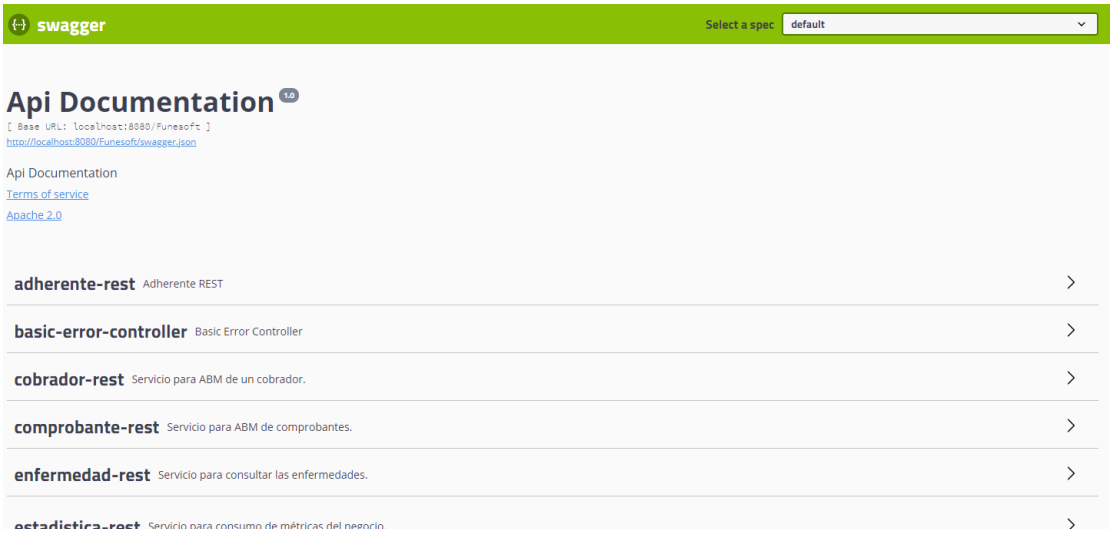


Figura 10. Interfaz de Swagger

A su vez, es posible realizar consultas HTTP a cualquier servicio a través de dicha interfaz.

Luego, para complementar el resto de los documentos, se utilizó app.diagrams.net. Diagrams es un aplicativo libre y gratuito disponible en la plataforma Google que permite realizar diferentes diagramas para documentar un software y compartir los mismos en tiempo real, desde un diagrama de entidad-relación hasta el diseño de la arquitectura logrando una visualización agradable.

También se utilizaron las herramientas Docs y Sheets de Google para la organización de documentación, cálculos, entre otros.

[9] Swagger. https://swagger.io/

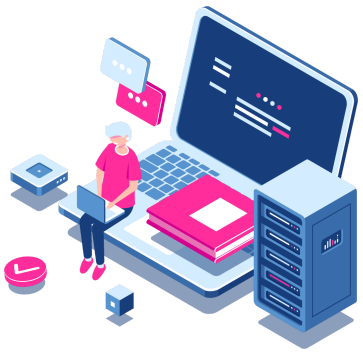
4.5 Desarrollo

Las serie de actividades llevadas a cabo en esta etapa están relacionadas con la tecnología en sí y el porqué de la utilización de las mismas, además de la entrega de una funcionalidad lista para realizar las pruebas.

4.5.1 Aplicación web

Luego de la revisión de los requerimientos en la etapa de análisis y su posterior diseño, se decidió que el producto debería ser una aplicación web, ya que se adapta por completo a la problemática, es decir, entregar un producto por partes siguiendo las iteraciones planteadas y apuntando a las futuras extensiones, haciendo un sistema totalmente escalable.

Es por esto que ante futuras actualizaciones, no será necesario una reinstalación en todos los dispositivos donde el cliente utilice el aplicativo, es decir, el mismo verá reflejado los cambios inmediatamente. Este es un punto que beneficia mucho este tipo de aplicativos, sobre todo cuando se apoyan sobre metodologías ágiles.



Siguiendo un estándar en común, la aplicación en alto nivel está desarrollada en dos partes: el Backend y el Frontend.

El Backend son todos los procesos que corren la lógica del negocio y el acceso a datos. En cambio, el Frontend es la interfaz con la que el usuario interactúa.

Gracias a esta división, es posible escalar completamente el aplicativo. Es decir que los procesos realizados en el backend construido con una arquitectura orientada a servicios pueden ser consumidos por otro sistema, como por ejemplo, una futura aplicación móvil.

Además, para la construcción del backend se siguió un estándar denominado MVC. Este estándar o patrón de diseño se sigue para lograr una separación en diferentes componentes que, en conjunto, manejan toda la lógica del

negocio, logrando así, una alta cohesión, es decir, cada componente con una responsabilidad única y un bajo acoplamiento, esto significa que cada componente trabaje de forma independiente conociendo lo mínimo e indispensable del otro. Gracias a esto, es posible la actualización o reemplazo de cada componente sin un gran costo, logrando así que la modificabilidad y escalabilidad del software sea lo suficientemente alta.

Sumado a esto, se trabaja también siguiendo un patrón de diseño denominado BFF. Para seguir este estándar, es necesario que el backend se adapte por completo al frontend y lograr así que la lógica entregue la información que la interfaz del usuario demande. Es decir, que el backend lea y procese de la base de datos sólo la información que el frontend requiera, logrando así que únicamente se entreguen datos esenciales ganando seguridad e integridad de los datos, entre otros.

Retomando la estructura del backend, este expone servicios REST para que el frontend los consuma. REST es un estilo arquitectónico utilizado para la comunicación entre sistemas distribuidos que se apoya sobre el protocolo HTTP, consiguiendo así poder utilizar los verbos más comunes: GET, POST, PUT, DELETE. Muchos sistemas heredados utilizan SOAP como alternativa a REST ya que este último surgió más tarde. Igualmente, son escenarios distintos y no se pueden comparar totalmente. SOAP es un protocolo y como tal, posee requerimientos específicos, en cambio, REST es un estilo que llegó para implementaciones más flexibles. Por lo tanto, los servicios REST son más ligeros.

Debido a eso, es posible crear una arquitectura orientada a servicios, donde cada conjunto de componentes que complementan un módulo, realiza una operación única.

Todo este conjunto de módulos que contiene procedimientos consumibles, conforma una API. En nuestro caso, el backend en completo corresponde a una API,

donde se pueden consumir servicios tanto para crear un socio así como también consultar el estado de un comprobante. Para los servicios se siguió una nomenclatura intuitiva donde el usuario o desarrollador que necesite hacer uso de algún servicio, entienda su finalidad sólo observando su nombre.

Gracias a que la API en cuestión está conformada por RESTful Web Services, toda la transferencia de información entre el backend y el frontend viaja en formato JSON (JavaScript Object Notation). Este formato permite una lectura de datos intuitiva, más entendible, y además los archivos son de menor tamaño y la velocidad de transferencia consecuentemente mayor.

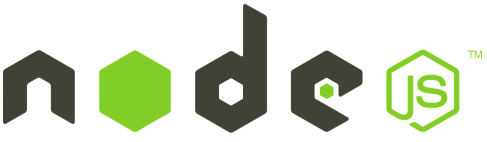
4.5.2 Tecnología

Al comenzar a definir qué herramientas y tecnologías deberían utilizarse para una mejor implementación del sistema, se presentó un fuerte foco en la documentación, soporte bibliográfico y la experiencia de los desarrolladores.

Al procesar toda esta información, se decide, en principio, qué tecnología utilizar en el backend.

4.5.3 Backend

Existen en el mercado actual varias ramas que dan soporte en su totalidad a la funcionalidad que el sistema debería entregar, entre las más comunes se encuentran: NodeJS, .NET, Java.



NodeJS está diseñado para dar soporte a aplicaciones escalables. Salió al mercado en 2009 y es de código abierto. Utiliza el lenguaje JavaScript y se ejecuta del lado del servidor. Además, es asíncrono, posee una arquitectura orientada a eventos y está basado en el motor V8 de Google. Es muy utilizado en el mercado hoy en día, hay mucho soporte bibliográfico, muchos blogs donde dan respuestas a preguntas comunes de

la tecnología pero ninguno de los desarrolladores tenía la experiencia suficiente como para utilizar Node en el backend del producto.

Otra alternativa que se analizó fue .NET, más precisamente el sucesor de .NET Framework: .NET Core con utilización de lenguaje C#.

La mayor diferencia entre .NET Framework y .NET Core es que en este último lo que buscó Microsoft es que sea multiplataforma y open source. Las aplicaciones desarrolladas en esta tecnología poseen gran escalabilidad y buena performance. La primera versión se lanzó en noviembre de 2014 y por lo cual actualmente se siguen entregando versiones mejoradas de la misma.

Si bien .NET Core se adapta completamente, está respaldado por Microsoft y hay mucho contenido bibliográfico, ninguno de los desarrolladores había trabajado en proyectos que involucren esta tecnología.

Por último se analizó Java. Este es un lenguaje antiguo con un soporte bibliográfico muy extenso en internet. Nació en 1995 y, desde entonces, cientos de miles de dispositivos corren esta tecnología en su sistema operativo. Con Java y su rama para el desarrollo de aplicaciones web, se logra conseguir escalabilidad y una arquitectura que se adapta completamente a nuestro aplicativo.

Si bien todas las herramientas poseen grandes características y se pueden conseguir resultados similares, Java, al ser muy antiguo, gratis y libre, se enseña en las universidades y se consiguen cursos totalmente gratuitos, es por esto, que cualquier persona que comience a desempeñar su tarea en el desarrollo, es muy probable que conozca dicho lenguaje.

Haciendo referencia al último punto de los tres que se tenían en cuenta para la elección de la tecnología a correr en el backend, todos los desarrolladores poseen experiencia tanto personal como laboral en Java[10]. Es por esto, que se decidió

utilizar dicho lenguaje para la construcción de los diferentes componentes que componen el aplicativo.

Sumado a esto, existen cientos de frameworks que se adaptan completamente al lenguaje y diferentes servidores que corren este tipo de aplicativo.

4.5.4 Framework de desarrollo

Para lograr una estructura conceptual y tecnológica, que aporte artefactos comúnmente utilizados en el mercado, con gran soporte y organización tanto del proyecto pero especialmente en el código, se decidió utilizar Spring Boot Framework[11].

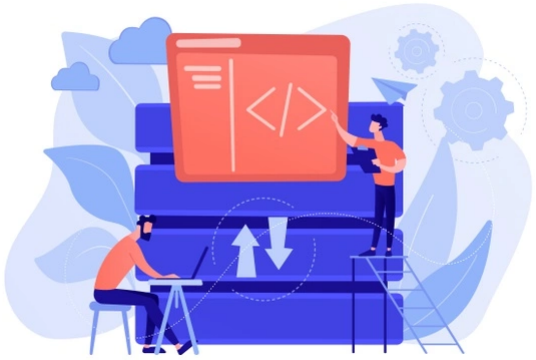
Este marco de trabajo muy popular en el lenguaje Java se utiliza para crear código de alto rendimiento, liviano y reutilizable, teniendo como finalidad estandarizar, agilizar y resolver los problemas que normalmente se presentan a la hora de desarrollar, ofreciendo como elemento clave el soporte de infraestructura a nivel aplicación, desde la configuración del software hasta su despliegue en diferentes tipos de servidores.

La principal ventaja es que gran porcentaje del esfuerzo del equipo de desarrollo se centra solamente en resolver la lógica de negocio, dejando todos los procesos de configuración al framework.

[10] Lenguaje Java. https://www.java.com/
[11] Spring framework. https://spring.io/

4.5.5 Base de datos

Toda la información que la aplicación crea, obtiene, actualiza y borra está alojada en una base de datos de tipo relacional.



Se utiliza una estructura de tipo relacional ya que los datos, por su naturaleza, pueden organizarse en tablas formalmente descritas. Toda la información se integra mediante identificadores, es decir, por ejemplo, la relación entre un socio y sus comprobantes es mediante un campo identificador y es esto la gran diferencia a una base de datos no relacional. Sumado a que son más robustas, mayor capacidad de almacenamiento, menos vulnerables ante fallas y utilizan lenguaje SQL.

La base de datos es una parte esencial de este aplicativo, ya que es donde se aloja toda la información; el sistema sólo se encarga de manipularla.

Para una correcta interacción y gestión de estos datos, es indispensable contar con un gestor de base de datos, donde los más utilizados son: SQL Server, MySQL, Oracle Database, DB2, entre otros.

Todos estos cuentan con aspectos importantes que cualquier gestor debe manejar:

1. SQL: Es el lenguaje de consulta estructurado que se utiliza comúnmente en este tipo de base de datos.

2. Integridad de los datos: Todas las bases de datos relacionales utilizan restricciones para una correcta integridad de los datos.

3. Transacciones: Es posible ejecutar una o muchas instrucciones SQL en una sola unidad lógica de trabajo.

4. Conformidad con ACID: Además, todas las transacciones deben ser atómicas, coherentes, aisladas y duraderas.

Para una correcta elección de qué motor utilizar dentro de nuestro aplicativo, se basó principalmente en la experiencia que cada desarrollador tenía manipulando cada uno de los gestores y en el precio correspondiente, ya que el resto de las características de cada uno no difieren en grandes aspectos.

Siguiendo esto, se decidió utilizar MySQL como motor de base de datos en nuestro producto.

MySQL[12] es un sistema de administración de base de datos relacionales de código abierto, posee una gran demanda en el mercado y es utilizado por grandes compañías, lo cual hace que exista gran cantidad de información y documentación sobre esta herramienta.

La conexión contra la base de datos la hace nuestro backend, siguiendo una arquitectura de capas. La instalación del motor de base de datos estará en el mismo servidor donde se ejecuta la lógica de negocio, por lo tanto se puede pensar en un modelo cliente-servidor, donde el cliente es el backend y el servidor nuestro gestor de base de datos.

Para pruebas locales, cada desarrollador posee un aplicativo denominado XAMPP[13] que levanta una instancia de MySQL localmente. La contra de esto es que en el ambiente local/desarrollo las base de datos de cada desarrollador no está sincronizada, pero acelera notablemente la velocidad en el que se puede entregar un requerimiento.

[12] MySQL. https://www.mysql.com/
[13] XAMPP. https://www.apachefriends.org/

Para el ambiente de testing, se genera una instancia tanto del gestor como de la aplicación en la nube a través de un servidor free para que se puedan realizar pruebas de integración correctamente.

En el ambiente productivo, la base de datos y el servidor de la aplicación corren en una misma infraestructura para mejorar la performance, seguridad e integridad de la información.

4.5.6 Servidor de aplicación

El servidor de aplicación es el sistema encargado de correr el aplicativo, tanto la parte del backend como la parte del frontend.

Reutilizando XAMPP, que es empleado para la instancia de MySQL, también trabaja levantando una instancia de un servidor http denominado Apache, utilizado con la finalidad de poder establecer una conexión entre un servidor y el navegador cliente. Dentro de este aplicativo, también se inicia una instancia de Tomcat, donde el mismo es un contenedor, utilizado ampliamente en el mercado para la compilación y ejecución de productos desarrollados en java, aunque también es posible ejecutar otro tipo de tecnología. Apache y Tomcat deben trabajar en conjunto.

4.5.7 Frontend

Para completar el stack de tecnología de nuestro producto, se decidió que del lado del cliente corra un framework muy demandado en el mercado y con mucha documentación: Angular[14].

La elección de dicha tecnología se basó en la experiencia de los desarrolladores. Sumado a esto, Angular se adapta completamente a los procesos de negocio que se necesitan implementar, separa el backend del frontend sumando gran escalabilidad, evita el código repetitivo y mantiene una estructura ordenada siguiendo el patrón BFF, asegurando rapidez en el desarrollo y en las entregas.

[14] Angular. https://angular.io/

Uno de los puntos más importantes de este lenguaje es que maneja un concepto denominado “Programación Reactiva”, donde manipula automáticamente el contenido que se muestra en la página, es decir, cualquier alteración en una variable se visualiza al instante en la pantalla.

4.5.8 Proceso de desarrollo

Dada la información que nos brinda el Chaos Report, se logra la definición de un proceso que nos enfoca en la organización y el control necesario para poder entregarle al cliente un producto de calidad.

A continuación, se detalla las etapas a seguir a la hora de desarrollar una funcionalidad:

Una vez que se especifican detalladamente cada uno de los requerimientos por cada incremento, generar los casos de uso, organizar las tareas e implementar el modelo de datos, se comienza a desarrollar los componentes del producto siguiendo las tecnologías previamente detalladas.

Al enfocar el proceso en un modo iterativo, el desarrollo de cada funcionalidad seguía la siguiente línea:

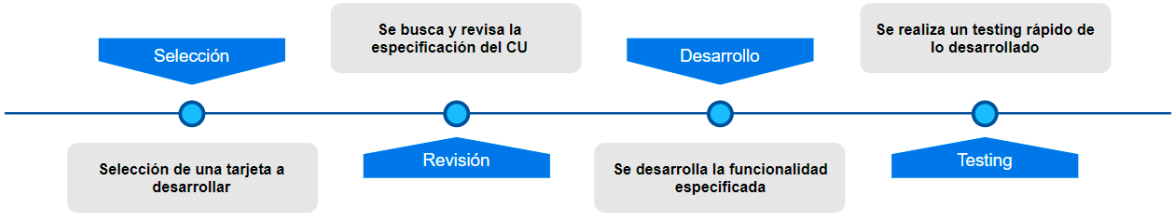


Figura 11. Diagrama de proceso de desarrollo

Al comenzar dicho proceso, se le asigna una tarea al desarrollador, luego, el mismo busca y revisa la especificación de caso de uso y comienza a desarrollar con las tecnologías especificadas. Al completar la tarea, realiza una prueba rápida de los desarrollado y mueve la carta a *Testing* para que el otro desarrollador realice el testing cruzado.

4.5.9 Estructura de los commits

A la hora de versionar el software y como se detalló en los conceptos de la ingeniería de software, el equipo se apoyó en la herramienta GIT.

A continuación se detalla una lista de prefijos para realizar un commit con el objetivo de llevar un control de nuevas características, arreglos de defectos u otros.

Tipologías aceptadas:

- **build**: Cambios que afectan las dependencias externas.
- **docs**: Cambios en la documentación.
- **feat**: Una nueva funcionalidad.
- **fix**: Solución de un bug.
- **perf**: Código que mejora el rendimiento.
- **refactor**: Organización de código.
- **style**: Cambios que mejoran la visualización del código.
- **test**: Test faltantes o corregir test existentes.
- **revert**: Un revert de un commit.

Ejemplos:

- **fix:** git commit -a -m "fix: <CU12> - se arregla el problema xxx"
- **feature:** git commit -a -m "feat: <CU12> - se implementa xxx"
- **documentación:** git commit -a -m "docs: <CU15> - solución del der"

4.5.10 Reportes

Para el diseño y generación de reportes se utilizó una herramienta muy conocida en el mercado denominada Jasper Reports[15]. Dicha librería tiene la función de diseñar informes lo suficientemente amigables para el usuario final, con la posibilidad de incorporar gráficos para mejorar la lectura, consumiendo la información directamente desde la base de datos.

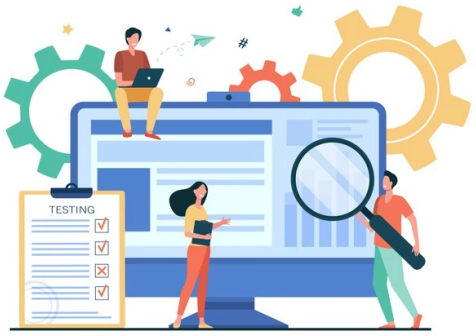
[15] Jaspersoft. https://www.jaspersoft.com/

Jasper Reports es el sistema de generación de reportes de código abierto más utilizado en el mundo donde sus principales características son:

- Informes en PDF, XML, HTML, CSV, entre otros.
- Estructuras de datos y gráficos variados.
- Múltiples fuentes de datos
- Generación de reportes de alta complejidad.

4.6 Pruebas

Cada vez que un desarrollador daba por terminado un módulo pasando por todas las etapas que concluían con un test unitario, el otro desarrollador era el encargado de realizar una prueba que contemple tanto la unidad desarrollada en sí así como también una prueba de integración.



Por lo tanto, en todo el ciclo de vida se contaba con diferentes tipos de pruebas[16]: Pruebas unitarias de las funcionalidades desarrolladas con una posterior prueba de integración, pruebas de los procesos de negocio a través de un testing cruzado y pruebas betas o de aceptación del cliente interactuando directamente con el mismo.

4.6.1 Pruebas unitarias

Como se comentó anteriormente, todo el proceso de pruebas unitarias se desarrolló en el backend, es decir, probar funcionalidades que abarcan la lógica del negocio en sí y no probar la interfaz del usuario con esta modalidad.

Para todo este procedimiento, se utilizó el framework JUnit, el cual brinda un conjunto de librerías que permiten la ejecución controlada de diferentes clases y poder evaluar si el comportamiento de estas es el esperado, es decir, ante un valor de entrada corroborar si el retorno es correcto.

[16] Roger S. Pressman (2011). Ingeniería del software: un enfoque práctico. *Estrategias de prueba de software.*

Uno de los puntos donde se hizo gran foco a la hora de desarrollar las pruebas, es que cada uno de los test debe cumplir con el principio de FIRST.

- **F**ast: Ejecución rápida que no conlleve grandes tiempos.
- **I**solated: Independientes del resto de las pruebas.
- **R**epeatable: Que se pueda repetir en el tiempo.
- **S**elf-validating: Cada prueba se deberá validar a sí misma si es correcta.
- **T**imely: Desarrollar los test en el momento adecuado.

Además, para aportar mayor dinamismo a las pruebas, se utilizó una librería llamada Mockito que nos permite simular comportamiento, aislar dependencias de otras clases y solo testear la funcionalidad concreta que existe en el código.

Si bien el desarrollo estuvo pensado en que cada módulo posea un bajo acoplamiento, las pruebas de integración de dicho módulo se basan en tomar el resto de los módulos con los que interactúa como una caja negra, esto ayudó a identificar que cada módulo posee una baja responsabilidad y poder definir una alta cohesión en cada uno. Todo esto se pensó de esta manera ya que la arquitectura así lo demandaba, es decir, una arquitectura de servicios donde cada uno posee una responsabilidad bien definida.

En la aplicación, se evalúa la cobertura del código que corre en el backend ya que se cree necesario corroborar la lógica de negocio y obtener una métrica de cuanto código está cubierto. Hacer este seguimiento de manera manual se torna imposible, por lo tanto, la elección de una herramienta es una actividad fundamental en este proceso.

A continuación se muestra la cobertura en cada uno de los paquetes utilizando Jacoco:

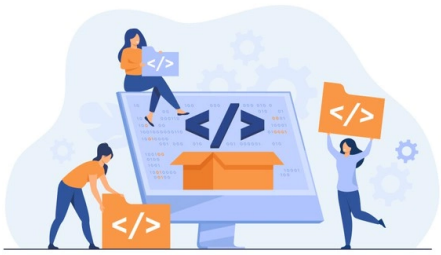
Funesoft

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
com.funesoft.model		41%		24%	436	649	571	938	204	411	1	25
com.funesoft.controller		24%		20%	109	149	420	549	56	91	0	17
com.funesoft.rest		14%		0%	48	75	206	247	47	74	0	18
com.funesoft.dto		29%		16%	206	304	200	306	201	298	17	26
com.funesoft		37%		0%	12	22	29	52	8	18	0	5
com.funesoft.utilities		51%		50%	15	31	33	64	13	28	1	8
com.funesoft.repository		6%		n/a	7	8	14	15	7	8	0	1
com.funesoft.config		100%		n/a	0	3	0	3	0	3	0	1
Total	7.069 of 10.370	31%	472 of 618	23%	833	1.241	1.473	2.174	536	931	19	101

Figura 12. Cobertura del producto

4.6.2 Pruebas cruzadas

Como se comentó en la etapa de diseño, la asignación de tareas se manejaba de manera dinámica y el desarrollo de las mismas era totalmente autónomo, es decir, el desarrollador llevaba a cabo la construcción de la pieza correspondiente siguiendo su total entendimiento del requerimiento especificado en el caso de uso.



El producto en sí estaba compuesto por varios procesos de negocios no comunes y requerían de una exhaustiva validación, cada vez que se terminaba de integrar una nueva funcionalidad al producto, la misma era probada por el otro desarrollador, logrando que lo entendido por ambos sea exactamente igual.

Mediante este tipo de prueba, se logró mejorar la robustez del producto antes de ser liberado al cliente para su posterior aceptación.

4.6.3 Pruebas de aceptación

Luego de cada incremento, se liberaba una versión al usuario final o cliente. En esta instancia, se realizaron diferentes pruebas de usabilidad y aceptación, apuntando a corroborar si los procesos de negocios desarrollados entregaban el resultado esperado por el negocio, ya que es el cliente quién tiene mayor conocimiento de esto.

Este proceso seguía la naturaleza iterativa del ciclo de vida del desarrollo del producto, ya que cada vez que se liberaba un incremento con un conjunto de funcionalidades lista para probar, se realizaba una instalación en la máquina local del cliente para que, posteriormente, el mismo pruebe las piezas construidas.

En este ciclo de pruebas, se lograba captar información de usabilidad del usuario, es decir, si había flujos que al mismo se le dificultaba operar. Por lo tanto en

esta etapa, se probó también la interfaz del usuario, visualizando si la misma cumplía con las expectativas del cliente y del proceso de negocio en sí.

4.7 Despliegue

Cada vez que se completaba una iteración, se sumaban nuevas funcionalidades que quedaban disponibles para ser probadas por el cliente.

A nivel implementación, se decidió, en principio, desplegar el aplicativo en un servidor local del cliente, dejando todo preparado para una migración futura a la nube en un corto plazo. Esta migración no será tan costosa por como se pensó la arquitectura del sistema en sí.

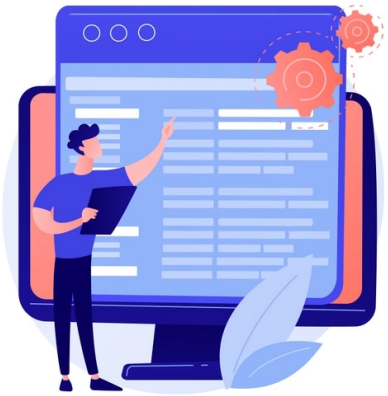
Para realizar las pruebas betas, también se optó por un despliegue en un servidor local en la máquina del cliente, logrando que el mismo pruebe los diferentes circuitos y comience a familiarizarse con el sistema.

El despliegue se llevó a cabo levantado un servidor Tomcat donde se desplegó el archivo *.WAR del backend como una aplicación de spring boot y la aplicación de angular para el frontend. Para este mecanismo, se utilizó un aplicativo denominado “XAMPP”, que además de los despliegue, es posible dar de alta una instancia de MySQL, que se utilizó como base de datos.

Gracias a este tipo de despliegue, se logró mejorar tanto la performance como la seguridad de la información, evitando que la misma se encuentre en la nube, reduciendo el costo de abonar una plataforma que no aporte significativas funcionalidades ya que no es necesario acceder a la información del sistema desde una red ajena a la local. Para futuras extensiones, se analizará llevar todo el producto a una plataforma en la nube y poder consumir la información de diferentes sistemas y/o dispositivos.

4.7.1 Ambientes

Como se comentó anteriormente, todo el desarrollo del proyecto consta de tres ambientes: desarrollo, testing y producción.



Desarrollo es el ambiente que utiliza cada desarrollador para construir las piezas que demande el requerimiento en el que está trabajando. En este también realiza sus pruebas unitarias, para controlar que el componente que desarrolló funcione correctamente. A nivel servidor, ejecuta en su máquina local los mismos programas que se instalarán en la máquina del cliente, comenzando a probar esa integración también.

A nivel versionado, se cuenta con una rama *develop* que es donde los desarrolladores vuelcan sus cambios locales.

Luego de que la prueba unitaria concluye correctamente, se suben los cambios al ambiente de testing donde se realizan las pruebas de integración, es decir, probar como el componente construído se relaciona con el resto de los módulos. Este despliegue se realiza en una de las máquinas de los desarrolladores, donde ambos prueban cruzadamente las funcionalidades desarrolladas.

Para el manejo del versionado en esta instancia, se mergea la rama *develop* a una rama de tipo *release/**.

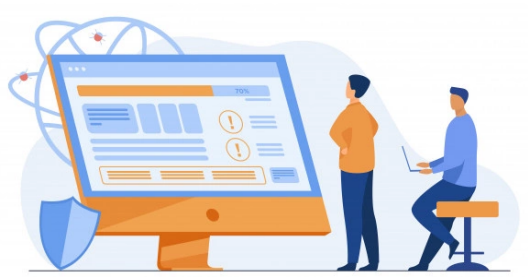
Cuando ya se consigue una línea base del producto, se sube todo el paquete al ambiente de Producción, es decir, se mergea el contenido de la rama *release/** a la rama *master*. Este ambiente es liberado al cliente, es decir, se despliega en el servidor local del cliente para comenzar las pruebas betas. Toda la información y el aplicativo deben funcionar de manera consistente y acorde a los deseos del usuario.

4.8 Gestión de la configuración

Todo el conjunto de actividades descritas anteriormente, es decir, el proceso de análisis, diseño, desarrollo, pruebas y despliegue generan un resultado que se puede dividir en tres amplias categorías:

1. Producto de software (código fuente como ejecutable)
2. Documentos (técnicos y de usuario)
3. Estructura de datos (base de datos y de conocimientos)

Todo elemento que compone esta información producida se denomina, en conjunto, configuración del software.



La configuración del software[17] es la única representación tangible del sistema, por lo tanto, cuando se desarrolla el producto final, se debe controlar para mantener una correcta precisión, una actualización constante y una consistencia adecuada.

Un proceso vital en el desarrollo de software son los cambios y se generan por tres motivos principales:

1. Cambios por partes del cliente
2. Cambios en el enfoque técnico del desarrollo
3. Cambios en el enfoque del proyecto

A medida que pasa el tiempo y se conoce más lo que se quiere lograr, ocurren estas modificaciones. Por lo tanto, es fundamental una correcta gestión con el objetivo de minimizar los costos.

Los cambios se producen en cualquier momento y por diferentes motivos, ya sean, revisiones, nuevas ideas, etc. y en cualquier fase, es decir, en el diseño, en el desarrollo o en las pruebas.

[17] Roger S. Pressman (2011). Ingeniería del software: un enfoque práctico. *Administración de la configuración de software.*

Por lo tanto, la gestión de la configuración del software es un conjunto de actividades dedicadas a gestionar los cambios durante todo el ciclo de vida, ganando calidad en todas las fases del proceso.

A continuación, se diagramó el proceso de petición de un cambio y su debida aprobación o denegación:

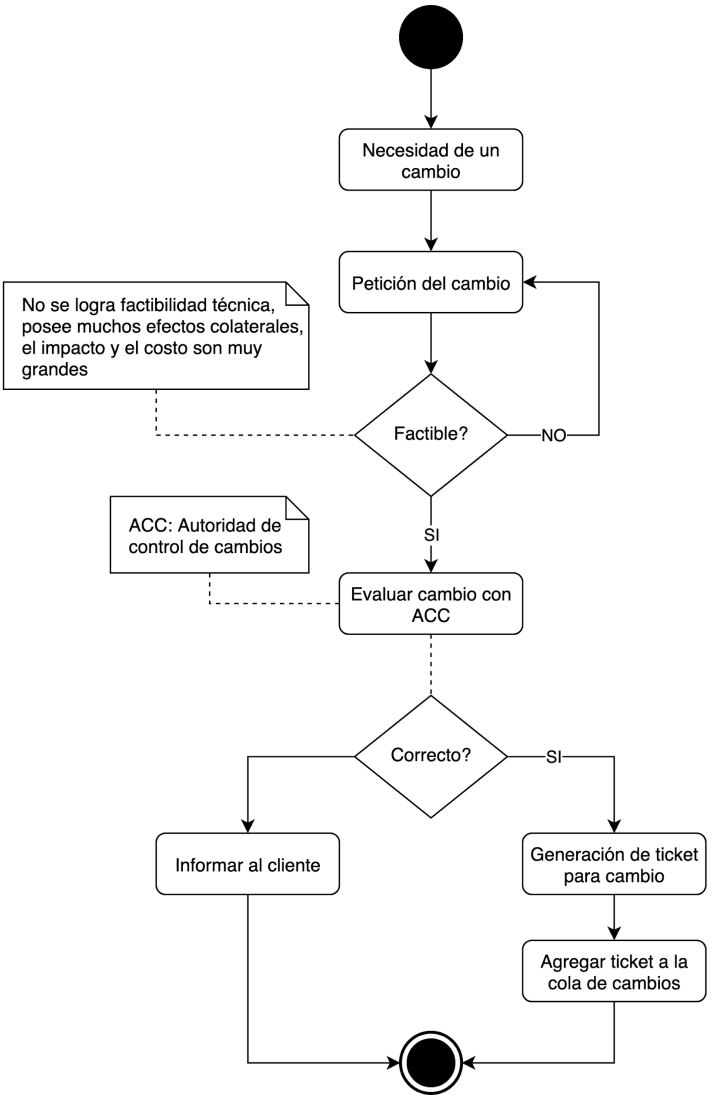


Figura 13. Proceso de petición de un cambio

Como el equipo actual es reducido, la ACC estaba compuesta por los dos desarrolladores. La primera actividad es presentar un informe de costo beneficio y el tiempo que demanda en realizar el cambio, previamente revisando si era técnicamente factible.

Si la petición es rechazada, se realiza una entrevista con el cliente explicando detalladamente el motivo. Si la petición se acepta, se genera un ticket en el tablero kanban con todas las especificaciones técnicas refinadas.

4.9 Gestión del proyecto

4.9.1 Gestión de riesgos

A continuación, se presenta el conjunto de disparadores y la estrategia utilizada sumado al plan de contingencia, ambos propuestos en la planificación, sumado a una descripción de los sucesos ocurridos durante el desarrollo.

ID	Disparador	Estrategia
R1	Un integrante del equipo obtiene un parte médico que requiere reposo absoluto por 24 horas o más.	Mitigar
R2	Un integrante del equipo sufre un accidente que afecta a su habilidad de trabajar por más de un mes.	Mitigar
R3	La situación económica del cliente lo obliga a dar marcha atrás con el producto.	Mitigar/Aceptar
R4	Las proyecciones de progreso de trabajo demuestran fechas de entregas posteriores a la estimada.	Mitigar
R5	Los integrantes del equipo realizan más de tres CU en los que exceden sus horas estimadas y realizan al menos una búsqueda sobre funcionalidades de la tecnología utilizada.	Mitigar
R6	Un integrante decrementa su productividad (horas estimadas por CU/horas de trabajo por CU) a menos del 20% de sus mejores números.	Mitigar
R7	Los integrantes del equipo demoran más de 1 (una) hora de reloj en cualquier tipo de decisión que se requiera.	Mitigar
R8	La estructura inicial del proyecto es ineficiente causando que se deba rediseñar por completo nuevamente.	Mitigar
R9	Un integrante del equipo excede las horas estimadas de una tarea debido a demasiadas validaciones, mientras que los restantes priorizan la performance en CU del mismo módulo.	Mitigar

R10	El sistema requiere una restructuración de su arquitectura que lleva más de 8hs laborales.	Mitigar
R11	Los CU son redactados con ambigüedades o con requerimientos diferentes a los del usuario, de tal manera que se debe realizar cambios sobre los casos de uso que requieren volver a estimar horas de trabajo o deshacer trabajo realizado de 10 horas o más.	Mitigar
R12	El cliente no participa del proyecto ni se lo puede contactar en más de tres ocasiones por mes.	Mitigar
R13	Las reuniones diarias del equipo superan los 15 minutos, o las semanales de planificación los 30 minutos en más de tres veces por mes.	Eliminar

Tabla 13. Riesgos y estrategias

ID	Plan de gestión
R1	Se recuperarán las horas perdidas en los próximos días.
R2	Recurso con conocimientos similares ocupa el/los puesto/s.
R3	Se trata de explicar nuevamente que el proyecto vale la pena y la amortización será cubierta pronto. Se evalúa la oportunidad de brindar un plan de financiación que motive al cliente.
R4	Se evalúa nuevamente los costos para generar una nueva entrega, o se plantea la reducción del alcance del producto al cliente.
R5	Se opta por capacitar al personal mientras participan de la ejecución del proyecto.
R6	Se modifica la metodología de trabajo para aumentar la motivación.
R7	Se intenta investigar sobre el tema en otras fuentes, o, en caso de ser necesario, se consulta con un profesional del tema para cerrar la incertidumbre lo antes posible.
R8	Se reúnen los integrantes del equipo para consolidar ideas y mejorar el diseño.
R9	Se busca impartir el mismo objetivo para poder agilizar el proceso y llegar con las entregas. Además, se verifica que las especificaciones de requerimientos no sean ambiguas.

R10	Se intenta contratar a los servicios y componentes necesarios para suplantar los erróneos y que el desarrollo o despliegue de la aplicación pueda realizarse de manera adecuada.
R11	Mediante una reunión del equipo se intenta comprobar la efectividad de la situación y, de ser necesario, se consulta al cliente.
R12	Se intenta resolver todo lo posible con autonomía y luego del tercer intento de contacto con el usuario sin tener éxito, se realiza una visita presencial.
R13	Se realiza una reunión haciendo uso de cronómetros y controlando que el contenido de la misma sea relacionado e indispensable.

Tabla 14. Plan de contingencia

En la siguiente sección, se presentan los riesgos que se materializaron y los que no, con el plan de mitigación que se llevó a cabo.

Situación ante riesgos

ID	Disparador	Estrategia
R1	Un integrante del equipo obtiene un parte médico que requiere reposo absoluto por 24 horas o más.	Mitigar

R1: El equipo no tuvo problemas de salud que conlleven a no poder trabajar por más de 24 horas. Además, ninguno sufrió accidentes.

ID	Disparador	Estrategia
R2	Ningún desarrollador sufrió accidentes durante el ciclo de vida de desarrollo del producto.	Mitigar

R2: Ningún desarrollador sufrió accidentes durante el ciclo de vida de desarrollo del producto.

ID	Disparador	Estrategia
R3	La situación económica del cliente lo obliga a dar marcha atrás con el producto.	Mitigar/Aceptar

R3: La situación económica del cliente está comprometida en parte por la pandemia, afectando en gran medida sus finanzas. Por lo tanto, se propuso que él nos brindara los conocimientos del negocio y el equipo implementaría el sistema sin costos. Si bien se siguió en gran medida el plan de gestión, se optó por entender la situación y continuar con el proyecto.

ID	Disparador	Estrategia
R4	Las proyecciones de progreso de trabajo demuestran fechas de entregas posteriores a la estimada.	Mitigar

R4: En cuestión de tiempos, la pandemia afectó en gran medida el desempeño del equipo, ya que los mismos debieron reubicarse, sumado al nivel de estrés que conlleva dicha situación. Por lo tanto, en los requerimientos se puede observar una reducción del alcance para tratar de acercarse a las fechas estimadas.

ID	Disparador	Estrategia
R5	Los integrantes del equipo realizan más de tres CU en los que exceden sus horas estimadas y realizan al menos una búsqueda sobre funcionalidades de la tecnología utilizada.	Mitigar

R5: Todo el equipo estuvo capacitado para poder lograr un desarrollo correcto de todas las funcionalidades. Solamente se vieron afectados los tiempos con errores que surgieron en las etapas de testing o requerimientos no claros descritos por el cliente. Por lo tanto, no se necesitó capacitar al personal.

ID	Disparador	Estrategia
R6	Un integrante decrementa su productividad (horas estimadas por CU/horas de trabajo por CU) a menos del 20% de sus mejores números.	Mitigar

R6: La motivación del personal se vió afectada en porcentajes relativamente bajos, es decir, complementar el desarrollo del proyecto con las horas laborales de cada uno sumado al estrés de la pandemia redujo un poco la productividad, pero no lo necesario como para aplicar el plan de mitigación.

ID	Disparador	Estrategia
R7	Los integrantes del equipo demoran más de 1 (una) hora de reloj en cualquier tipo de decisión que se requiera.	Mitigar

R7: Los desarrolladores cuentan con gran capacidad de autonomía y es por esto que todas las decisiones que se tomaron fueron en tiempo y forma.

ID	Disparador	Estrategia
R8	La estructura inicial del proyecto es ineficiente causando que se deba rediseñar por completo nuevamente.	Mitigar

R8: En modelo de datos se fue refinando en cada iteración siguiendo la metodología planteada, logrando un modelo consistente sin tener que ejecutar un plan para lograr rediseñarlo desde 0.

ID	Disparador	Estrategia
R9	Un integrante del equipo excede las horas estimadas de una tarea debido a demasiadas validaciones, mientras que los restantes priorizan la performance en CU del mismo módulo.	Mitigar

R9: Se logró conseguir un conjunto de especificaciones de requerimientos claras y concisas, por lo tanto, el objetivo de los desarrolladores estuvo alineado a entregar un producto de calidad tratando de cumplir con las fechas.

ID	Disparador	Estrategia
R10	El sistema requiere una restructuración de su arquitectura que lleva más de 8hs laborales.	Mitigar

R10: Se utilizaron componentes que todos los desarrolladores conocen por completo, es por esto que no fue necesaria una reestructuración de la arquitectura.

ID	Disparador	Estrategia
R11	Los CU son redactados con ambigüedades o con requerimientos diferentes a los del usuario, de tal manera que se debe realizar cambios sobre los casos de uso que requieren volver a estimar horas de trabajo o deshacer trabajo realizado de 10 horas o más.	Mitigar

R11: Los casos de uso fueron redactados con información previamente refinada con el cliente, si bien surgieron dudas, todas pudieron ser aclaradas con el equipo de desarrollo o mediante una simple comunicación con el cliente, por lo tanto, ninguna de estas tareas demandó más de 10 horas.

ID	Disparador	Estrategia
R12	El cliente no participa del proyecto ni se lo puede contactar en más de tres ocasiones por mes.	Mitigar

R12: El cliente estuvo predispuesto en todo el ciclo de vida del desarrollo del producto, aceptando entrevistas y comunicaciones para aclarar dudas.

ID	Disparador	Estrategia
R13	Las reuniones diarias del equipo superan los 15 minutos, o las semanales de planificación los 30 minutos en más de tres veces por mes.	Eliminar

R13: Si bien en algunas ocasiones no se cumplieron las reuniones diarias, no fue necesario cronometrar los encuentros, ya que era tiempo necesario para aclarar dudas que bloqueen el desarrollo de alguna funcionalidad.

4.9.2 Organización de tareas

Para una mejor organización de la división de tareas entre los miembros del equipo, se detallaron con una mayor granularidad dichos casos de uso en un tablero de tipo Kanban. Cada tarjeta fue asignada a un desarrollador donde el mismo iba comentando un seguimiento de la funcionalidad que iba construyendo, es decir, en qué estado se encontraba.

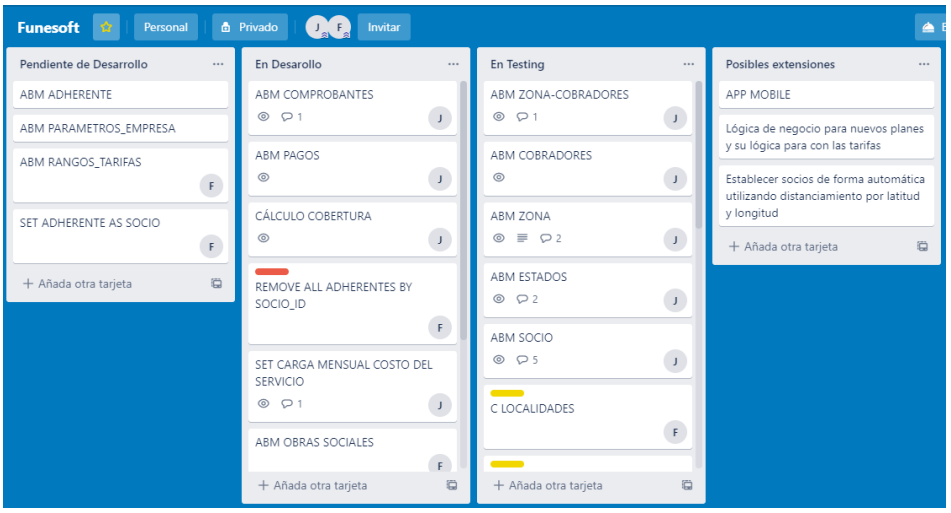


Figura 14. Tablero trello

Como se puede observar, cada tarjeta contenía una funcionalidad específica, además de un conjunto de actividades detalladas por el desarrollador para conocer el estado de la misma, cuándo fue realizada y si ya estaba apta para comenzar a realizar el proceso de testing cruzado.

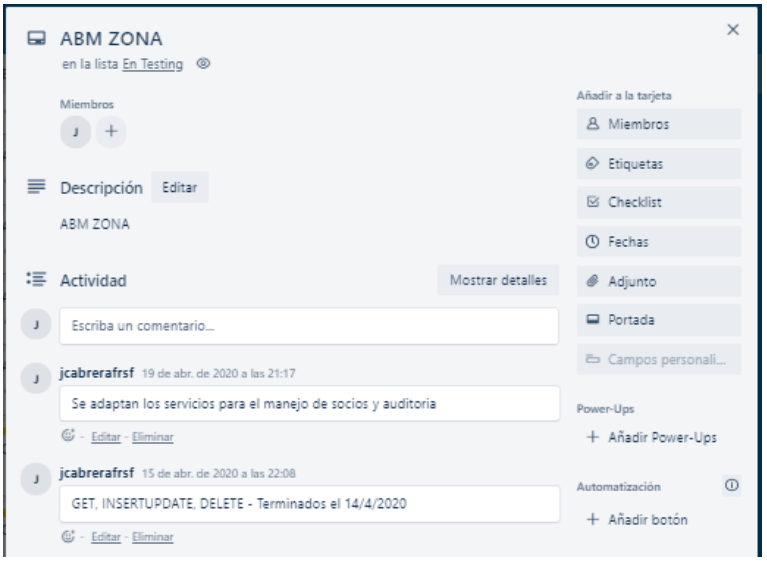
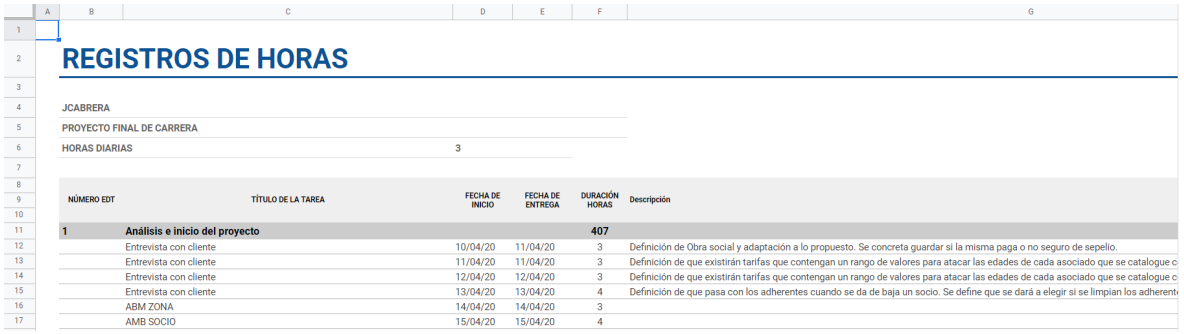


Figura 15. Información detallada de un requerimiento en trello

A su vez, el desarrollador podía dejar comentarios a tener en cuenta para la fase de prueba o de integración con otros módulos.

Este proceso se repetía de manera de seguir con los incrementos detallados para poder brindarle al cliente una entrega lo antes posible, obtener su retroalimentación y así evitar defectos futuros en la integración de los módulos restantes.

Para una mejor organización interna del trabajo diario que realizaba cada desarrollador, se llevó a cabo un registro de horas donde se detalla que requerimiento construyó. Es decir, un registro de la tarea realizada, fecha de inicio y fin así como también cuántas horas trabajó en el día y un detalle de lo que realizó. A continuación, se muestra el documento de registro de horas:



REGISTROS DE HORAS

JCABRERA

PROYECTO FINAL DE CARRERA

HORAS DIARIAS 3

NÚMERO EDT	TÍTULO DE LA TAREA	FECHA DE INICIO	FECHA DE ENTREGA	DURACIÓN HORAS	Descripción
1	**Análisis e inicio del proyecto**			407	
	Entrevista con cliente	10/04/20	11/04/20	3	Definición de Obra social y adaptación a lo propuesto. Se concreta guardar si la misma paga o no seguro de sepelio.
	Entrevista con cliente	11/04/20	11/04/20	3	Definición de que existirán tarifas que contengan un rango de valores para atacar las edades de cada asociado que se catalogue c
	Entrevista con cliente	12/04/20	12/04/20	3	Definición de que existirán tarifas que contengan un rango de valores para atacar las edades de cada asociado que se catalogue c
	Entrevista con cliente	13/04/20	13/04/20	4	Definición de que pasa con los adherentes cuando se da de baja un socio. Se define que se dará a elegir si se limpian los adherent
	ABM ZONA	14/04/20	14/04/20	3	
	AMB SOCIO	15/04/20	15/04/20	4	

Figura 16. Documento de registro de horas

En primer lugar, se detalló un backlog de las tareas a realizar además de las posibles extensiones, toda esta información se consiguió luego del relevamiento de los requerimientos y refinados de los mismos a través de las especificaciones de casos de uso.

Luego, ya definido el primer incremento, se llevaron las tareas puntuales afectadas al mismo, atomizando las mismas en actividades más puntuales con el fin de conseguir asignaciones a cada desarrollador e ir atacando un caso de uso por parte.

4.10 Arquitectura

La arquitectura de software[18] es la estructura del mismo, es decir, un conjunto de elementos que pueden ser objetos del software en sí, un proceso, una librería, una base de datos, etc, y las relaciones entre ellos. Además, el comportamiento de cada uno de estos elementos es parte de la arquitectura.

Todo sistema tiene una arquitectura, ya que todo software puede mostrarse como composición de elementos. Sin embargo, los sistemas deben estar conformados por más de una estructura.

Sumado a esto, es importante definir la arquitectura basándose en los atributos de calidad que se esperan, es decir, reflejar la solución de diseño teniendo en cuenta los requerimientos de calidad.

A continuación, se detalla un cuadro donde se especifican los diferentes atributos de calidad[19] que se tuvieron en cuenta para generar la arquitectura planteada:

#	Atributo de calidad	Descripción	Justificación
1	Seguridad	Habilidad de poseer diferentes mecanismos de protección en diferentes niveles.	Al contener información sensible es imprescindible que usuarios no autorizados no tengan acceso a los datos
2	Integridad	Propiedad que busca mantener los datos libres de modificaciones no autorizadas.	Es importante que la información en este sistema sea consistente y confiable.
3	Escalabilidad	Habilidad de un sistema para reaccionar y adaptarse al crecimiento de información.	Para hacer frente a un aumento masivo en el volumen de datos.

4	Usabilidad	Facilidad con la que el usuario puede utilizar el sistema.	Es importante ofrecer una experiencia amigable y agradable al usuario con el fin de que el mismo utilice y confíe en todos los módulos.
5	Performance	Medida de la eficiencia en el uso de recursos del sistema ejecutándose.	Es importante que el sistema funcione con alto rendimiento para lograr mostrar la información rápida para la toma de decisiones.

Tabla 15. Principales atributos de calidad

Gracias a esto, se decide seguir una arquitectura orientada a servicios, apuntando a desarrollar una serie de pequeños módulos que se ejecuten de forma autónoma y se comuniquen entre sí, logrando gran escalabilidad y performance.

Además, siguiendo esta idea, es posible aplicar diferentes técnicas a cada módulo para el cifrado de la información, ganando así seguridad e integridad en datos sensibles.

A continuación, se presentan diferentes vistas de la arquitectura del sistema, donde se logra ver una representación de un conjunto de elementos y las relaciones entre estos:

[18] Len Bass, Paul Clements, Rick Kazman (2003). Software Architecture in Practice. *What Is Software Architecture?*
[19] Len Bass, Paul Clements, Rick Kazman (2003). Software Architecture in Practice. *Understanding Quality Attributes.*

4.10.1 Vista de instalación

Documentar una arquitectura es documentar las vistas relevantes y luego agregar documentación que se aplica a más de una vista.

La vista de instalación hace referencia a cómo está estructurado el sistema de archivos del proyecto en sí.

Cómo se utilizó un estilo arquitectónico orientado a servicios, en principio se tiene un servidor donde está toda la lógica del negocio, es decir, el BFF (Java server, en la imagen).

Además, también se cuenta con otro servidor donde está instalado el conjunto de interfaces que consume el usuario (Angular server, en la imagen).

Esta información le ofrece al desarrollador la estructura del proyecto en la que va a generar nuevos módulos y poder seguir un estándar previamente definido.

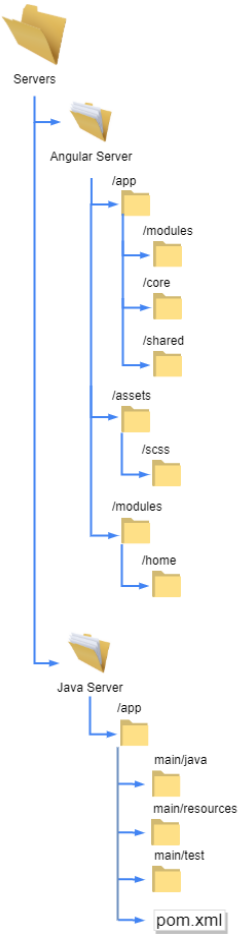


Figura 17. Vista de instalación

4.10.2 Vista de componentes

La vista de componentes define entidades consistentes que tiene alguna presencia en tiempo de ejecución, la interacción entre ellas y una posible agrupación de componentes donde se pueda representar como un todo. Generalmente las interacciones son llevadas a cabo utilizando alguna infraestructura, tales como framework, por ejemplo.

En el siguiente diagrama se logra observar los componentes principales de nuestro producto y las posibles interacciones entre ellos:

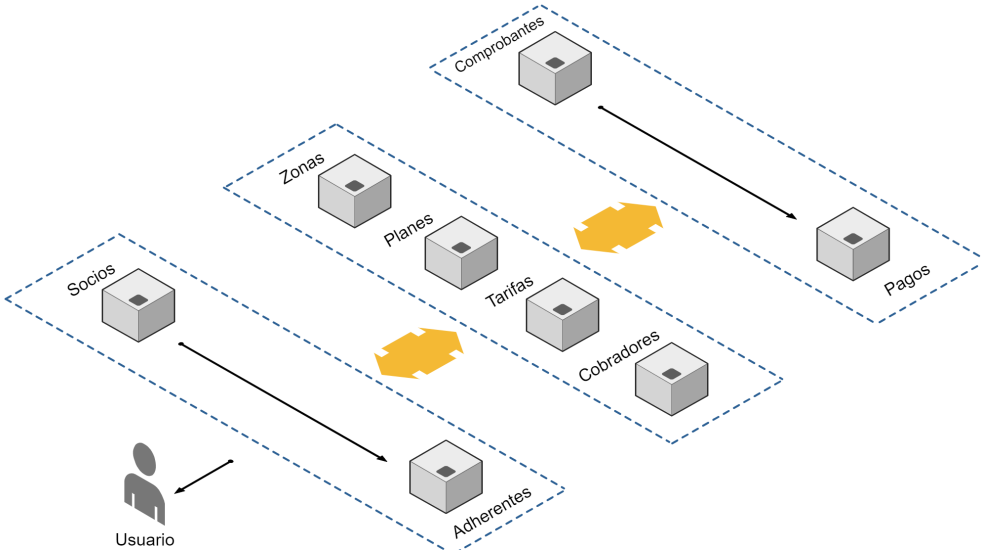


Figura 18. Vista de componentes

Como se observa en nuestra vista, cada componente debe tener un nombre y una posible relación con otro, es decir, que componente puede disparar una interacción o información de otro. Si bien se observa que el camino más común de la interacción del usuario con el sistema es contra los componentes de socios y adherentes, es posible que, a través de estos, se comunique o consuma información de Tarifas o Pagos.

En esta vista se logra observar como el sistema puede ser diagramado en diferentes componentes, cada uno con una responsabilidad única haciendo referencia a la alta cohesión y una relación entre los mismos lo más baja posible, es decir, bajo acoplamiento.

4.10.3 Vista de despliegue

Si bien el despliegue de un software es un proceso con una serie de actividades que culminando las mismas hacen un producto disponible y listo para su entrega, la vista de despliegue brinda información de cómo y dónde está desplegado dicho sistema.

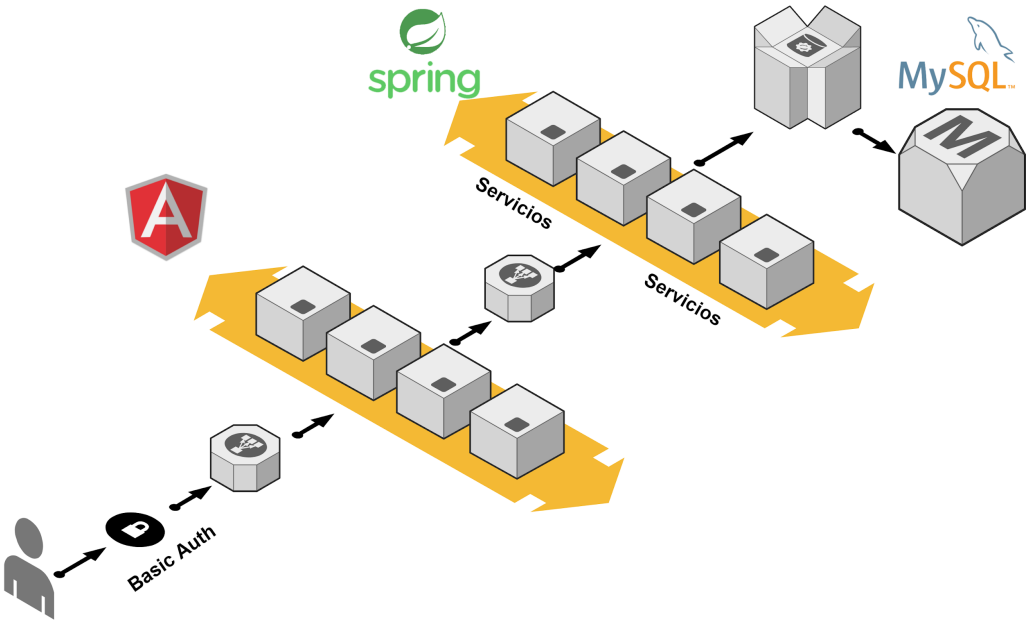


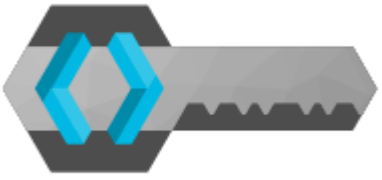
Figura 19. Vista de despliegue

En dicha vista, se observa la interacción de los componentes a un nivel más alto. El proceso, luego de que un usuario pasa el proceso de login correctamente, un orquestador determina que servicio del frontend se quiere consumir, por consiguiente, se determina que microservicio utilizar, donde estos mismos se comunican con el motor de la base de datos para acceder a la información.

4.11 Seguridad

Teniendo en cuenta que gran parte del diseño tanto del producto como de la arquitectura fue basado en la seguridad como primer atributo de calidad, fue necesario la incorporación de un módulo de autenticación al sistema.

Para implementar dicho módulo se realizó una comparativa con las plataformas más utilizadas en el mercado: Keycloak, Auth0 y Spring Security.



Keycloak es una plataforma de código abierto que permite SSO (inicio de sesión único) sumado a la gestión de identidad, siguiendo un marco de políticas y tecnologías para garantizar que solo los usuarios

permitidos tengan el acceso a los recursos. Si bien es un producto muy potente, no existe gran documentación en internet y la experiencia de los desarrolladores en el manejo de dicha plataforma era nula.

Luego, se analizó Auth0. Una de las principales ventajas de dicha plataforma es que posee una autorización adaptable y es muy fácil de instalar ya que cuenta con gran documentación en la web. En sí lo que provee es un sistema de autenticación simple y externo al sistema que lo implementa, generando gran modificabilidad y escalabilidad. Si bien contiene una implementación totalmente gratuíta, ninguno de los desarrolladores contaba con una experiencia en dicha tecnología que aporte mayor valor que implementar la opción que ya nos brindaba el framework que se utiliza en el backend.

Retomando el punto anterior, Spring como framework nos ofrece una librería totalmente adaptable a nuestro backend: Spring Security.

Spring Security nos brinda un marco de control de acceso y autenticación potente y altamente personalizable. Es un estándar de facto en las aplicaciones que utilizan Spring como framework para el backend, proporcionando autenticación y autorización para las aplicaciones Java. Además, existe suficiente documentación en internet como para destrabar cualquier inconveniente que se presente durante su implementación.

Por lo tanto, para la protección de nuestros datos y una correcta autenticación de usuarios se decidió la implementación de Spring Security, ya que los desarrolladores contaban con experiencia en dicha librería.

4.12 Auditabilidad

Apuntando siempre a mejorar la seguridad del sistema y considerando que este aspecto era uno de los puntos de mejora a abordar, se desarrolló un proceso de auditoría para evitar que quede información inconsistente en la base de datos.

Se diseñaron un conjunto de disparadores en la instancia de la base de datos que se ejecutan cada vez que se genera un operación sobre cualquier tabla, es decir, si se inserta un nuevo dato, se presenta la actualización de alguna fila o la eliminación de algún registro, se ejecuta una función que ingresa un nuevo registro en la tabla auditora de dicha entidad. Por ejemplo, si se presenta una inserción en la tabla de socios, la función a ejecutarse generará un nuevo registro sobre la tabla socios_audit.

Gracias a esto, es posible asegurarse que cualquier acción sobre los datos puede examinarse exhaustivamente a fin de establecer las responsabilidades reales de la operación.

4.13 Funcionalidades y comparativas

A continuación, se detallan las funcionalidades principales del sistema y cómo fueron sus mejoras a través del proceso de renovación, comparando el anterior sistema con la nueva implementación.

4.13.1 Acceso al sistema

Una de las primeras funcionalidades que difieren del sistema anterior es el módulo de autenticación de usuarios. Dicha funcionalidad aumenta la seguridad restringiendo el acceso a usuarios no deseados en la organización o empleados que no tengan permisos.

En la siguiente interfaz se puede visualizar la pantalla que se presenta al usuario cuando quiere ingresar por primera vez al sistema:

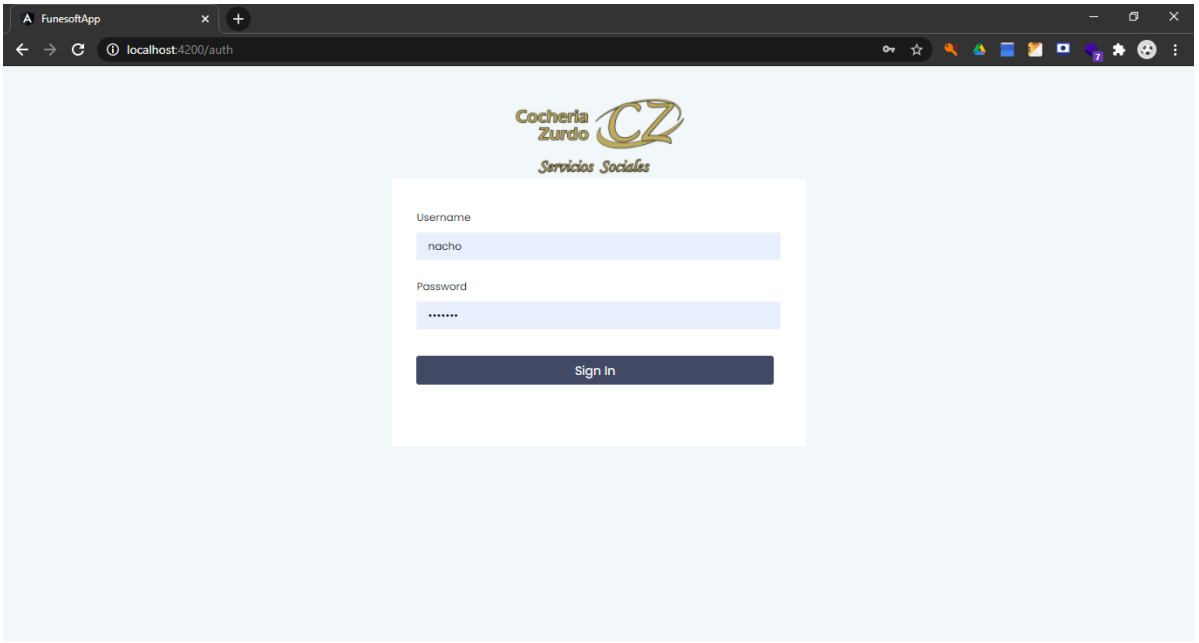


Figura 20. Interfaz de autenticación de usuario.

Este punto se destaca sobre el sistema anterior, ya que el mismo no tenía un módulo de autenticación, dejando mucha información sensible accesible a cualquier persona que tenga acceso a la computadora.

4.13.2 Gestión de socios y adherentes

Retomando la descripción de los procesos de negocios detallados anteriormente, en la siguiente imagen se observa la información que obtiene el usuario al consultar el listado de socios a través del menú.

En la primera pantalla se muestran datos relevantes que puede consultar el usuario rápidamente, como por ejemplo, si un socio tiene cobertura o no, sumado a un conjunto de acciones: Detalles, Modificar y Dar de baja.

Además, es posible exportar dicho padrón en formato .pdf o .xls a través de los botones ubicados en la parte superior derecha de la tabla.

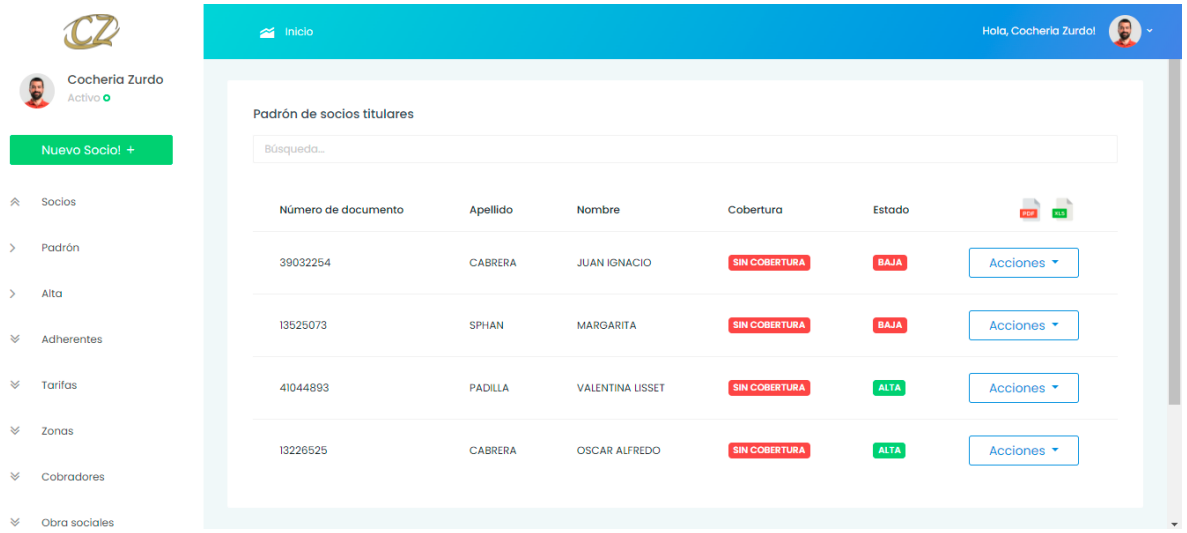


Figura 21. Padrón de socios

El objetivo principal fue mejorar la usabilidad y la amigabilidad. En la siguiente imagen, se puede observar el listado de socios del sistema anterior a modo comparativo.

Figura 22. Padrón de socios - Sistema anterior

La información que se visualiza no aporta mucha relevancia. Un dato importante es si tiene cobertura o no, que es información que debería estar

accesible rápidamente, sumado a la imposibilidad de manejar el sistema con el mouse.

Si el usuario ingresa a Detalles, se le presentará la siguiente pantalla, que tiene como objetivo mostrar toda la información que existe en la base de datos de un determinado cliente.

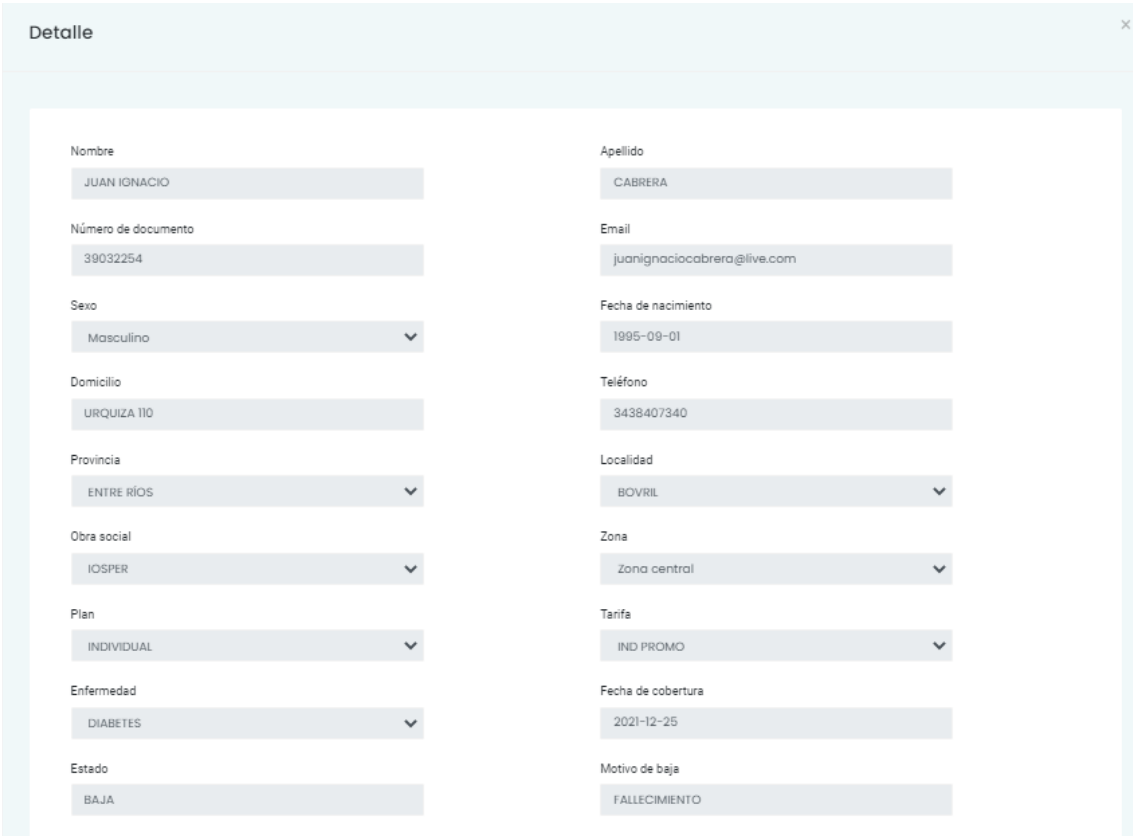


Figura 23. Detalle de un socio

Si se desea dar de baja un socio, se presenta el siguiente modal, donde se debe indicar el motivo de la baja.

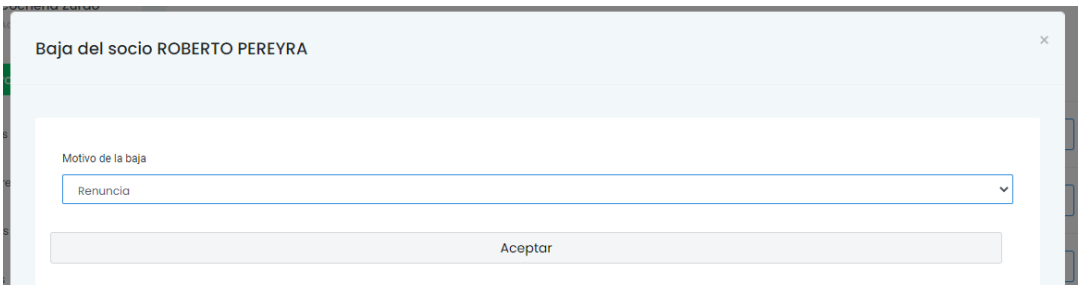


Figura 24. Motivo de baja de un socio

En comparación, al querer dar de baja un socio anterior, se presenta un formulario donde el usuario debe primero filtrar por zona, información irrelevante a la hora de dar de baja un cliente:

Figura 25. Baja de un socio - Sistema anterior

A su vez, si el usuario desea dar de alta un nuevo socio, deberá ingresar a Alta y completar la información del siguiente formulario:

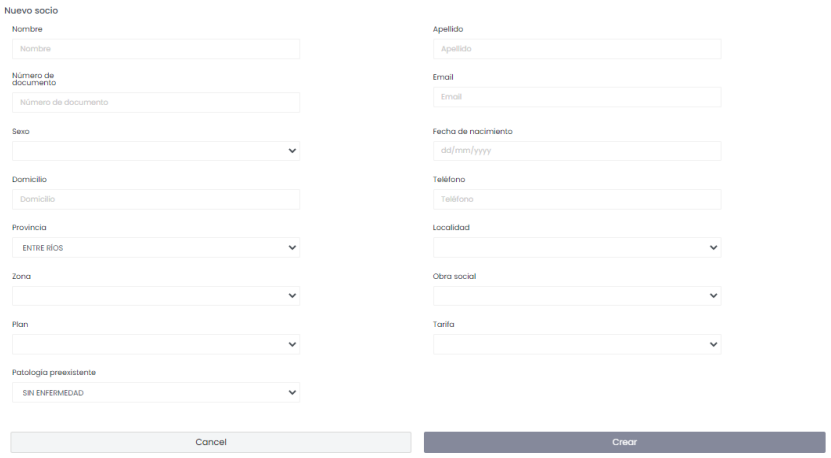


Figura 26. Formulario de alta de socio

Comparado con el formulario de alta del sistema anterior, se puede observar la amigabilidad y la usabilidad de la interfaz, sumando listas desplegables, calendarios para elección de fechas, validaciones, entre otros.

Figura 27. Formulario de alta de socio - Sistema anterior

Luego, para la gestión de adherentes, la dinámica y el conjunto de interfaces es similar.

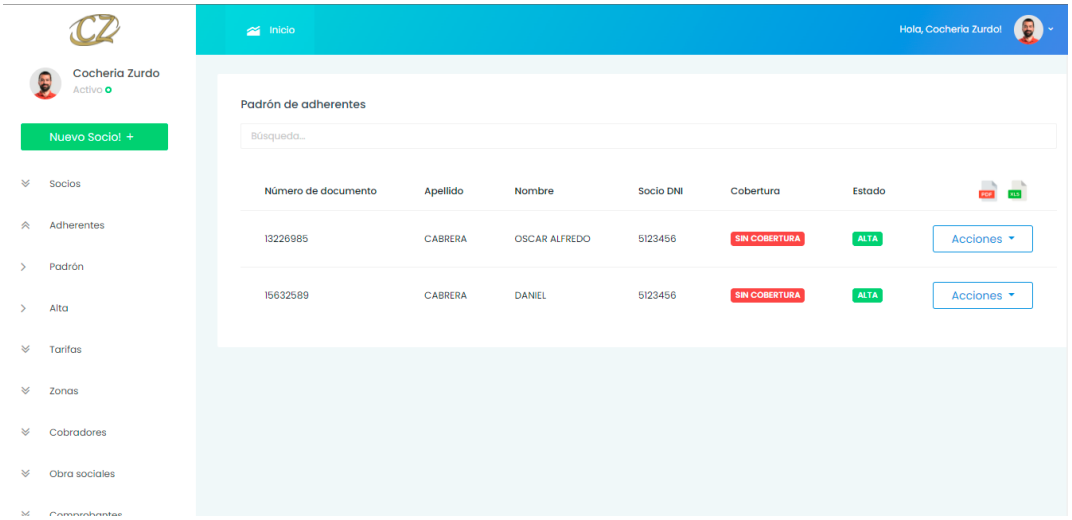


Figura 28. Padrón de adherentes

4.13.3 Gestión de planes y tarifas

Como se comentó anteriormente, la cantidad de planes se redujo de cinco a tres, ya que se consiguen las características necesarias para manejar a los clientes de manera óptima. Por lo tanto, esta información permanece fija en el sistema, sin la

posibilidad de agregar un nuevo plan por parte del usuario, conllevando a que esta información esté dentro de una lista desplegable al dar de alta un socio.

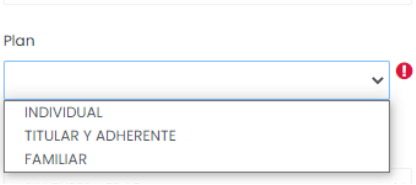


Figura 29. Lista desplegable de planes

En el sistema anterior, se manejaban cinco planes, en lo cual, los últimos dos, tenían las mismas características que el plan “Familiar”. Además, tiene la funcionalidad de poder editar y agregar nuevos planes, agregando flexibilidad. En cambio, en la nueva solución se traslada esa flexibilidad a que el usuario pueda agregar las tarifas que desee a cada uno de los planes, dejando estático el conjunto de planes.

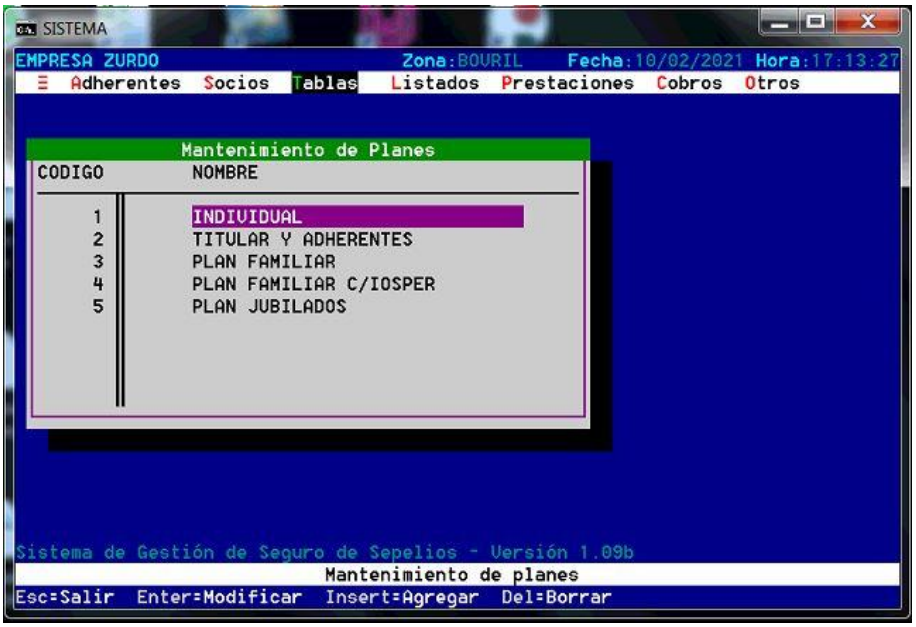


Figura 30. Padrón de planes - Sistema anterior

Luego, es posible para cada uno de estos, hacer la gestión de las tarifas, es decir, agregar una nueva tarifa para un plan específico, modificar, consultar los rangos y dar de baja.

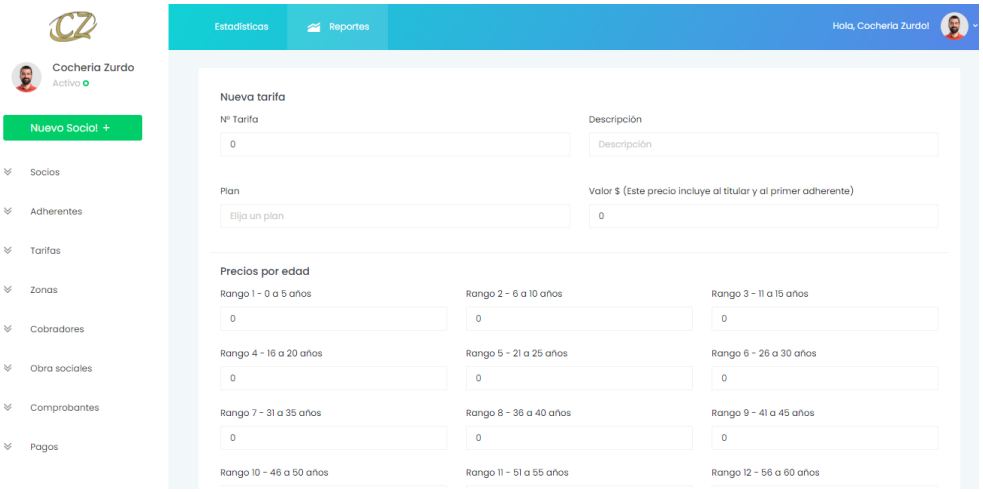


Figura 31. Formulario de alta de una tarifa

Al dar de alta una tarifa, es necesario agregar un número para que el usuario pueda utilizarlo como un acceso rápido a la misma, una breve descripción, el plan a la cual va a pertenecer, el precio base, donde el mismo solamente es necesario cuando el plan es “Familiar” y los precios por rango de edades que va desde los 0 años hasta los 105 años.

Comparando con el sistema anterior, las tarifas no tenían una relación directa con el plan, por lo que al dar de alta una tarifa, la interfaz solicitaba al usuario información que no era relevante para todos los planes.

Figura 32. Alta y modificación de tarifas - Sistema anterior

Además, es posible consultar toda la información acerca de las mismas así como también dar de baja una o más tarifas.

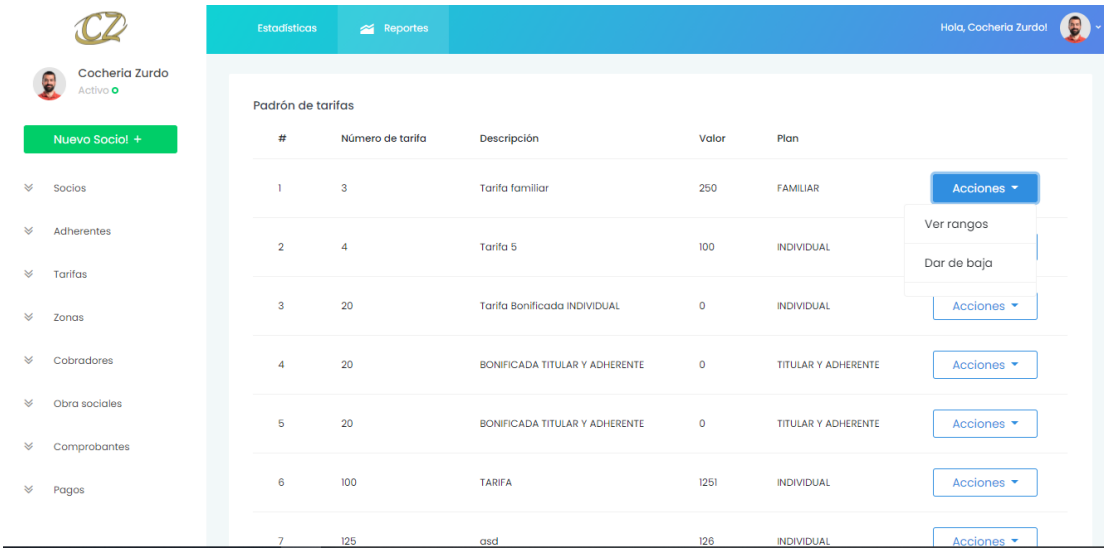


Figura 33. Padrón de tarifas y posibles acciones

Si el usuario desea recuperar la información de los rangos, se le presenta la siguiente tabla:

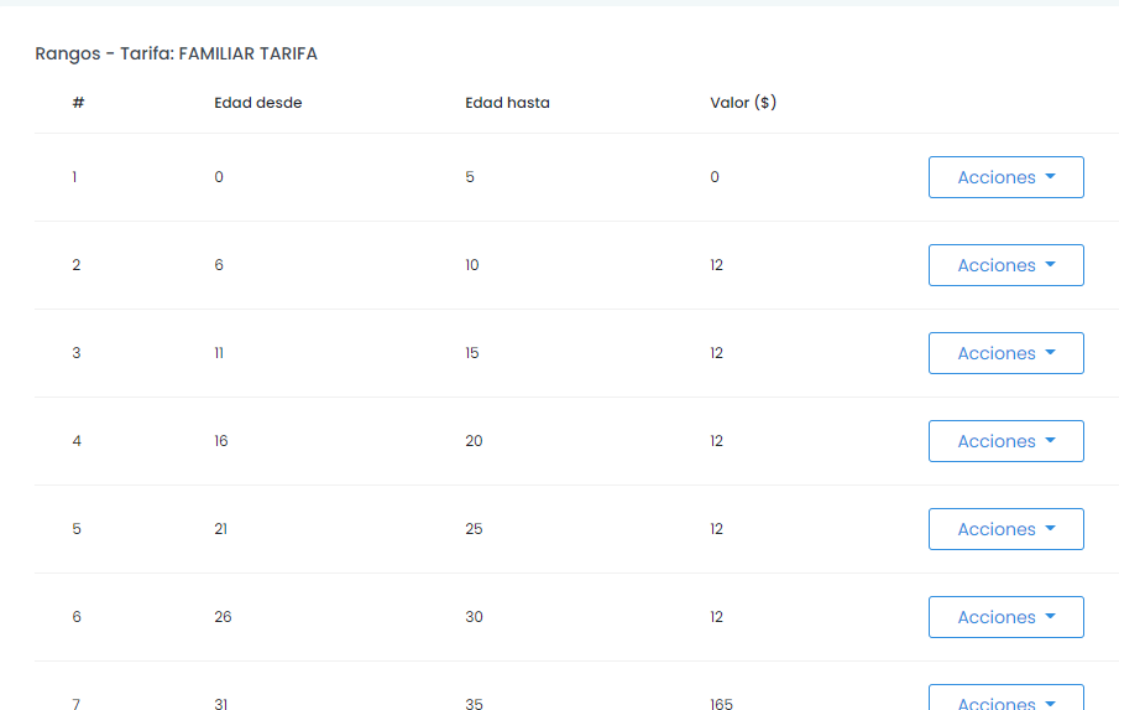


Figura 34. Tablas de rangos de una tarifa

Con la posibilidad de generar una modificación directa a cada campo sin tener que generar una nueva tarifa.

4.13.4 Gestión de obra social

Al dar de alta un socio y sus adherentes, es posible indicar si los mismos tienen o no obra social. A continuación, se muestra la pantalla donde se refleja toda la información de los registros cargados previamente por el usuario.

Los datos relevantes, en principio, son la descripción y si tiene seguro de sepelio, para corroborar si el cliente está cubierto en ese aspecto o no.

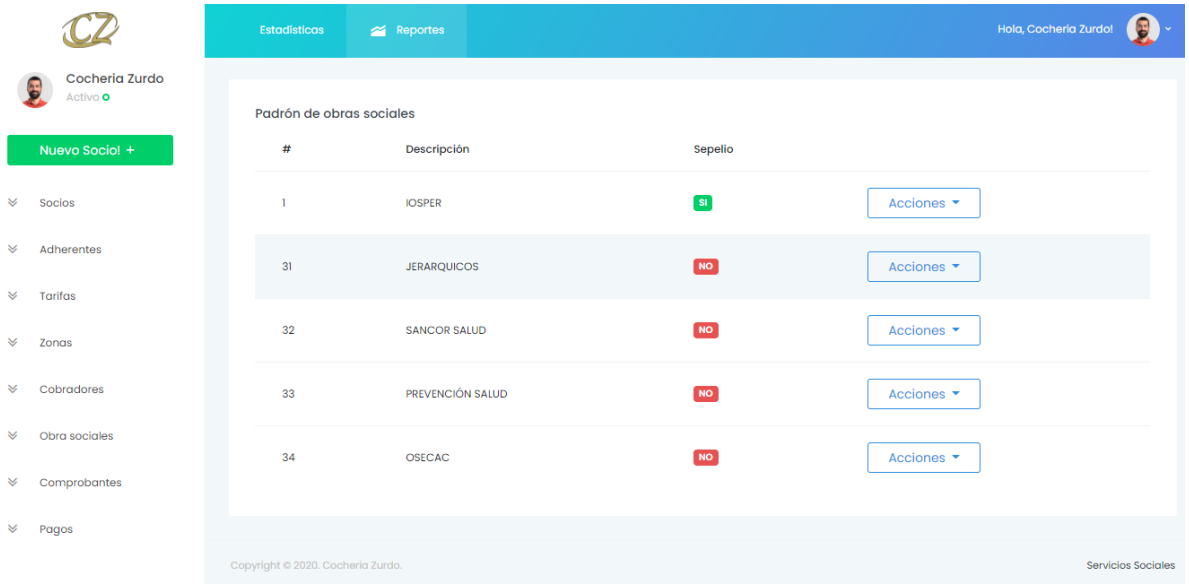


Figura 35. Padrón de obras sociales

Por lo tanto, para dar de alta una nueva obra social, solamente el usuario debe indicar una descripción, que generalmente es el nombre de la misma, sumado a la información de que si la misma tiene seguro de sepelio o no.

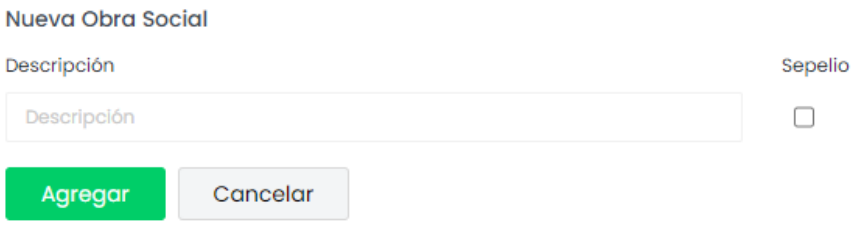


Figura 36. Formulario de alta de obra social

En comparación al sistema anterior, en el padrón que se puede visualizar, solamente se muestra el nombre de la misma y se presenta mucha dificultad para, por ejemplo, saber si la misma tiene cobertura de sepelio.

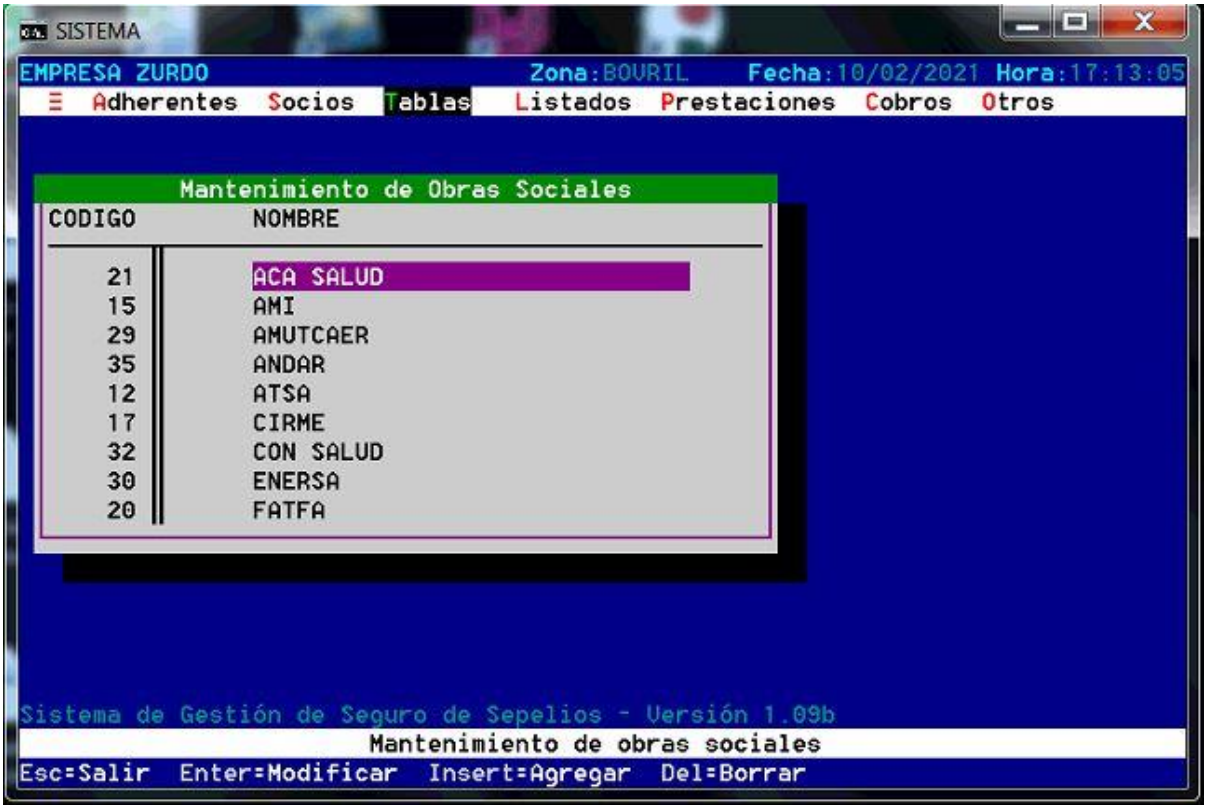


Figura 37. Mantenimiento de obras sociales - Sistema anterior

4.13.5 Gestión de zona y cobradores

Con el objetivo de poder segmentar los clientes en diferentes zonas y que cada una de las mismas mantenga un conjunto de cobradores, se desarrolló el siguiente módulo, donde con esta información, además de poder visualizar y llevar un control, adhiere diferentes datos para recopilación de métricas y mejorar las tomas de decisiones.

A continuación, se puede visualizar el padrón de zonas con la posibilidad de consultar qué cobradores pertenecen a la misma.

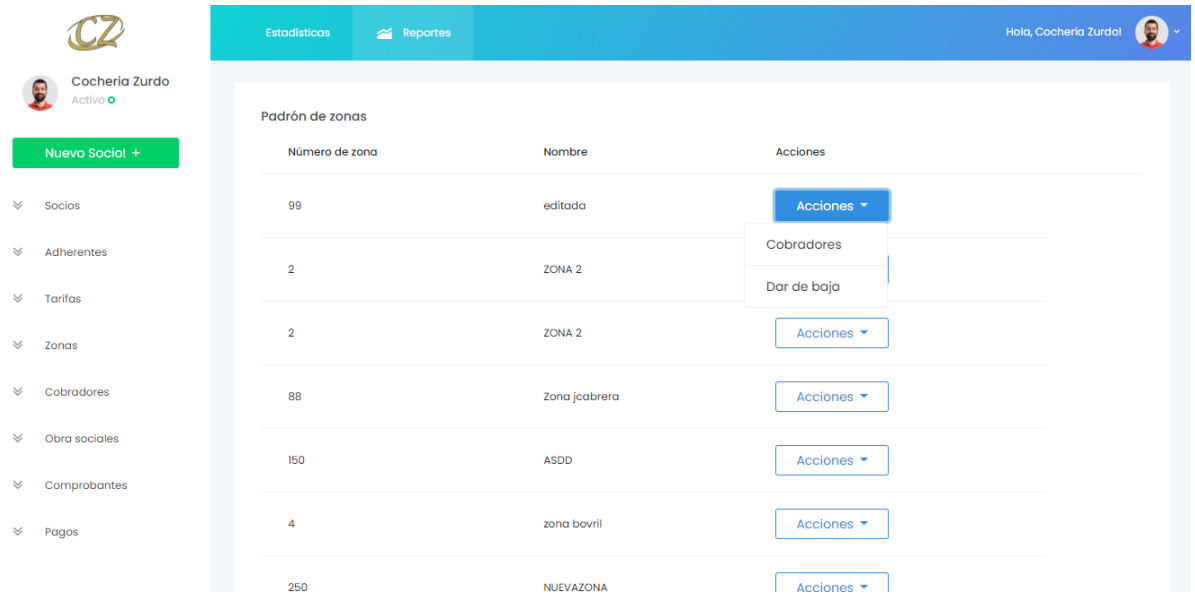


Figura 38. Padrón de zonas

Además, al dar de alta esta información, se presenta una pantalla donde debe elegir un número de zona y un número, sumado al conjunto de cobradores que estará afectado por la misma:

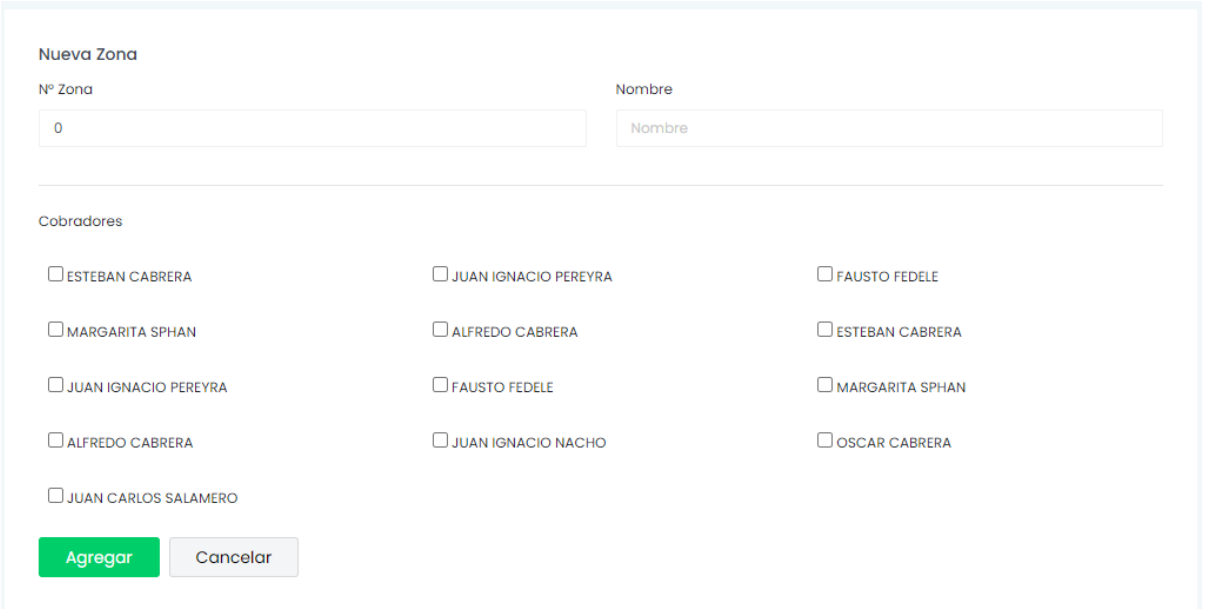


Figura 39. Formulario de alta de zona

Se puede recuperar el padrón de cobradores, donde también existe la posibilidad de buscar información más específica de los mismos:

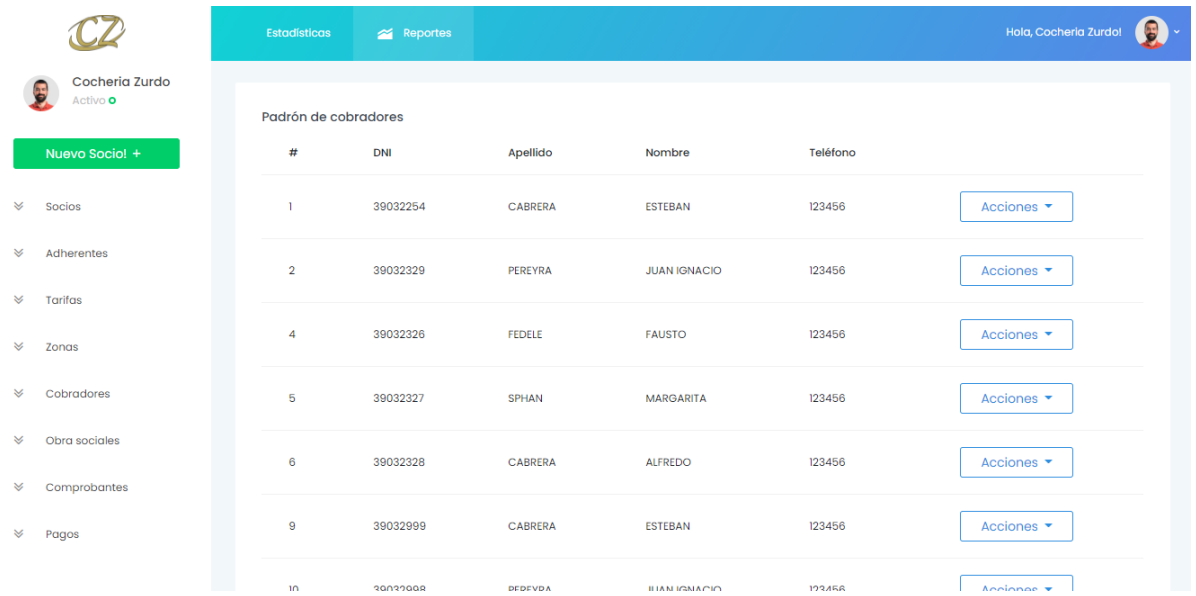


Figura 40. Padrón de cobradores

Luego, si el usuario desea realizar un alta de un nuevo cobrador, al mismo se le presentará la siguiente interfaz:

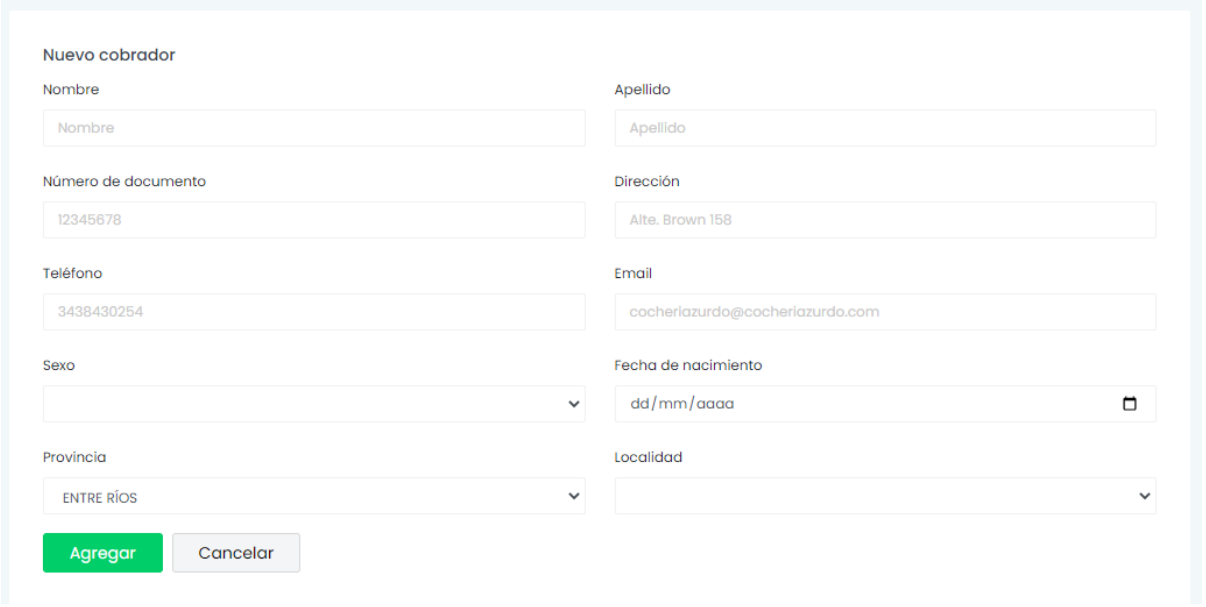


Figura 41. Formulario de alta de cobrador

Realizando una comparación con el sistema anterior, no existía la posibilidad de asignar más de un cobrador a una zona, además, se presentaba una serie de

dificultades para navegar desde el módulo de zona al de cobrador y viceversa, es decir, no se relacionaban directamente ambos conceptos.

En la siguiente imagen, se puede observar el padrón de zonas con un único cobrador asignado:

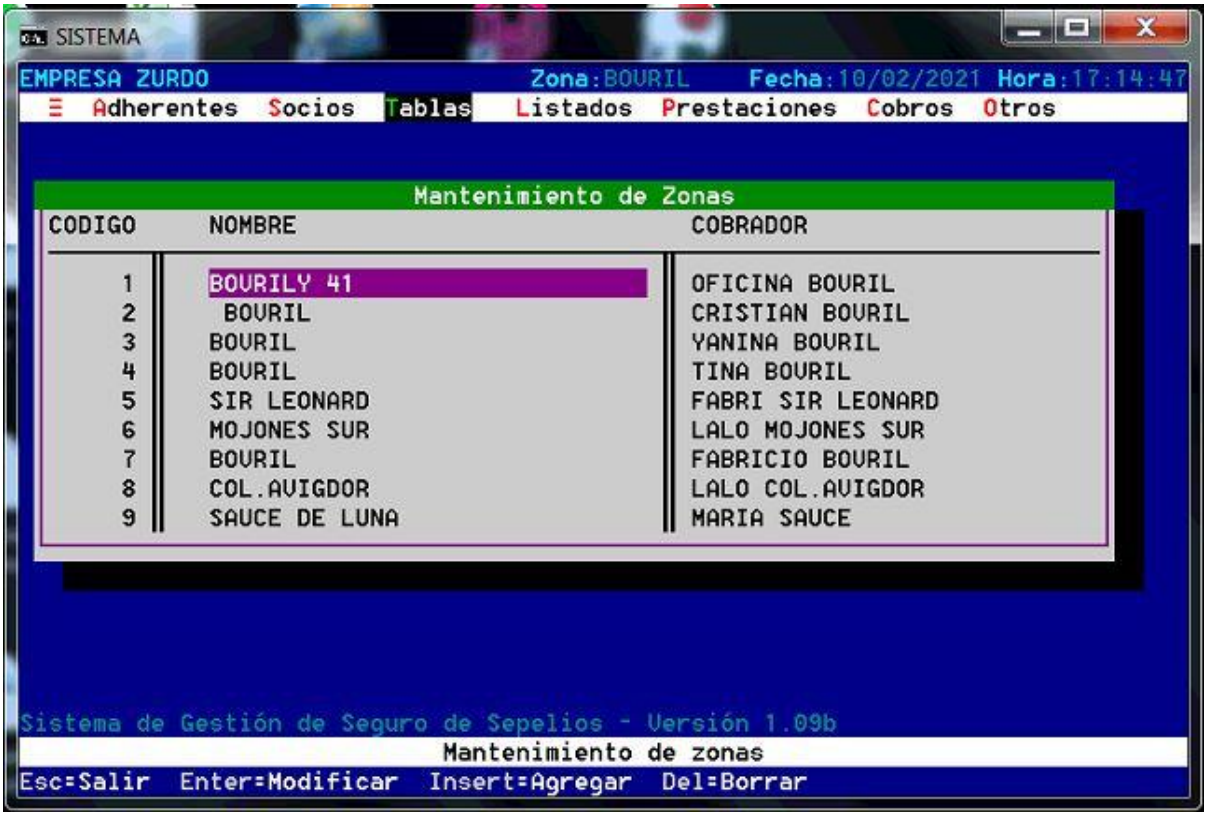


Figura 42. Padrón de zonas - Sistema anterior

Como se puede observar, se presenta un desorden a la hora de asignar más de un cobrador a una zona, es decir, el usuario repetía la información y generaba un desorden en la administración.

Además, al navegar nuevamente por el menú, se puede visualizar el padrón de cobradores, donde el mismo muestra una escasa información:

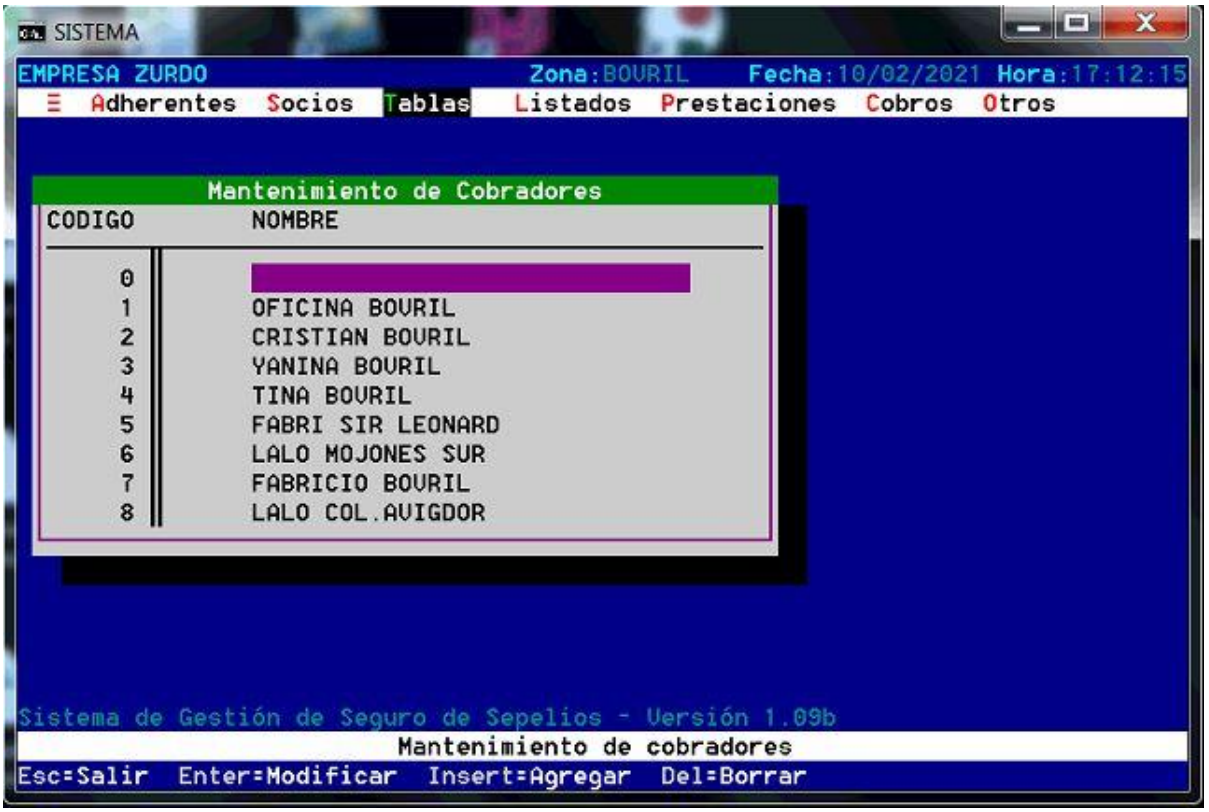


Figura 43. Padrón de cobradores - Sistema anterior

4.13.6 Gestión de saldo, comprobantes y pagos

Haciendo referencia a uno de los procesos de negocios principales de la organización, ya que es donde se administra el ingreso de dinero, se crearon diferentes componentes con el objetivo de llevar una correcta gestión de las finanzas de la empresa.

La primera información importante del socio es el saldo, información que se decrementa e incrementa cada vez que se genera un comprobante que afecta al mismo o cada vez que se genera un pago, respectivamente.

En la siguiente interfaz, se puede observar una lista desplegable donde el usuario podrá ir tipeando letras del nombre o apellido para luego seleccionar el socio buscado y mostrar la información principal, como su dni, plan, tarifa y el estado del mismo, sumado al saldo y a los comprobantes generados para el mismo.

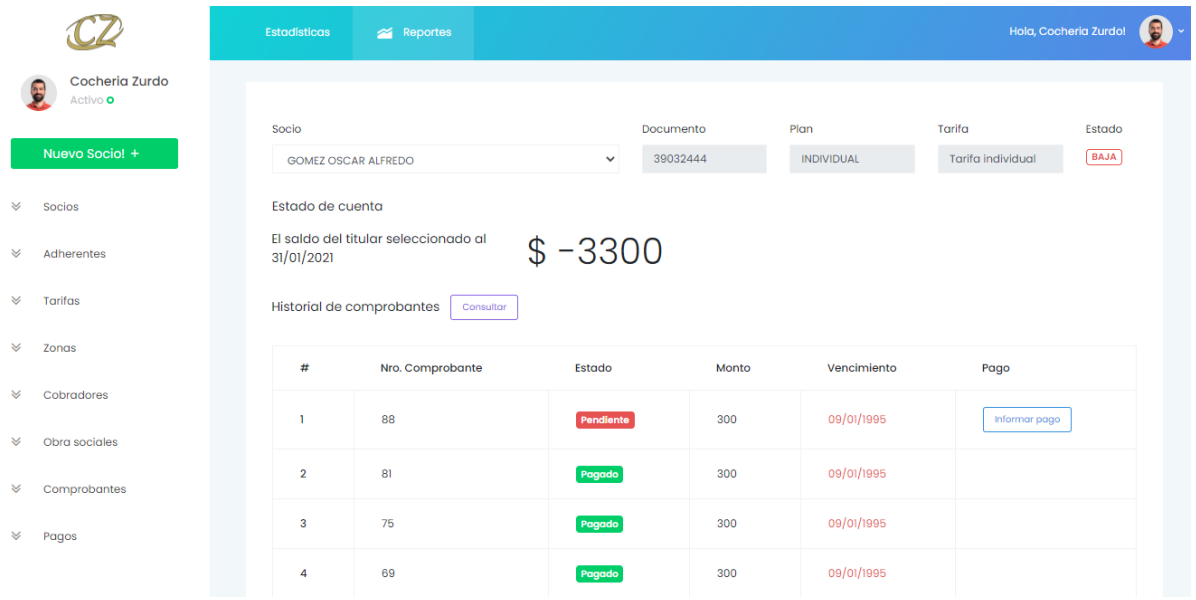


Figura 44. Estado de cuenta de un socio

Además, para cada uno de los comprobantes generados se puede observar información relevante y la posibilidad de informar el pago del mismo, donde se decrementará automáticamente el saldo del titular.

Luego, en el siguiente componente, el usuario mensualmente generará los comprobantes, acción que realizará un decremento del saldo en cada uno de los socios titulares que estén dados de alta. Además, se visualiza una tabla en la parte inferior donde se muestran todos los comprobantes generadores y su estado.

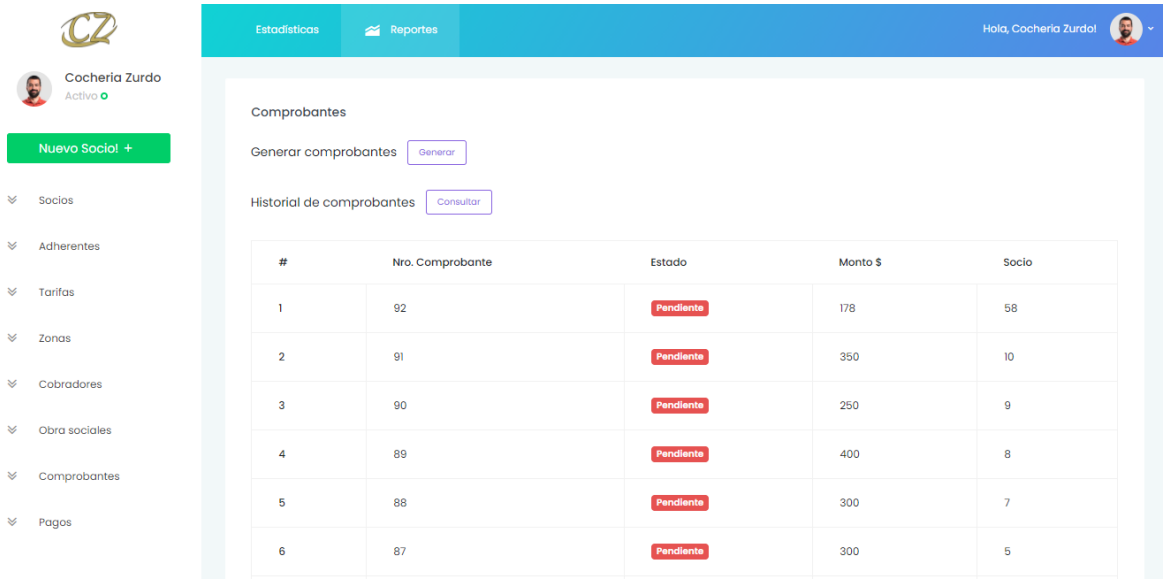


Figura 45. Interfaz de generación de comprobantes

Comparando con el sistema anterior, existe una sección de “Cobros” donde en la misma solamente se podía visualizar la cantidad de dinero que debía ingresar de un cobrador.

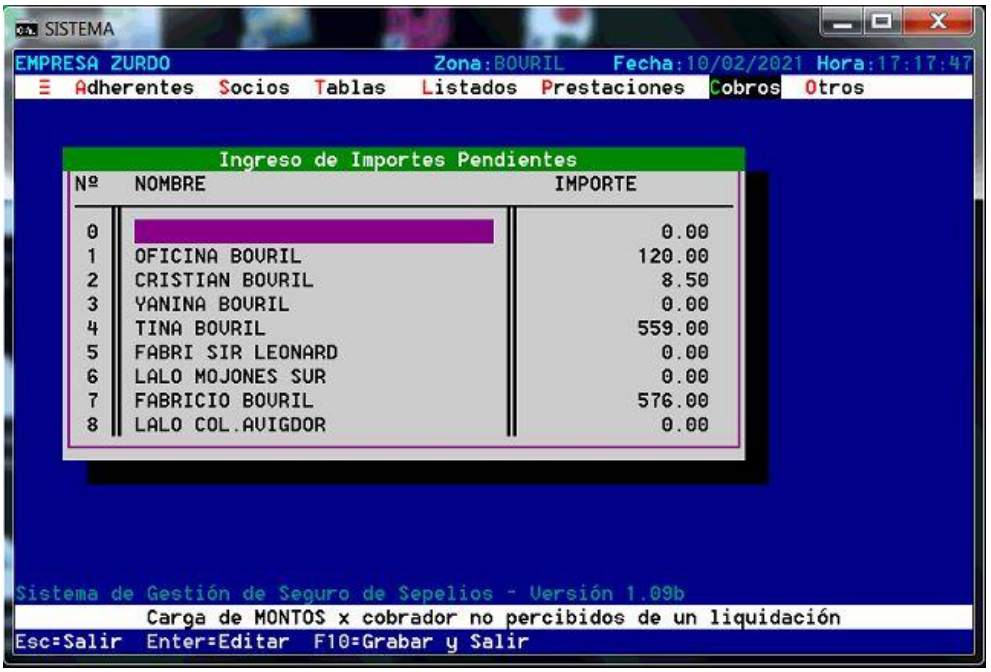


Figura 46. Ingresos pendientes - Sistema anterior

El proceso de generación de comprobantes de pago del sistema anterior era poco eficiente y muy costoso, ya que no generaba un documento de manera digital y debía imprimirse en su totalidad con una impresora antigua lo cual generaba un excesivo costo, sumado al elevado mantenimiento.

A continuación, se presentan ambos comprobantes:

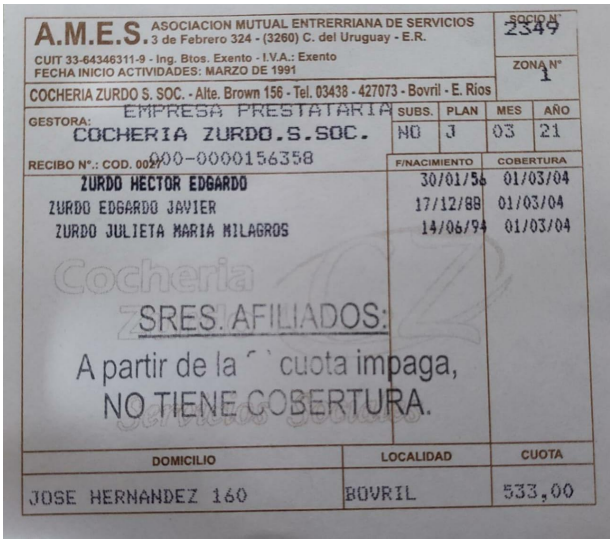


Figura 47. Comprobante de pago - Sistema anterior

Cocheria Zurdo - Servicios Sociales | Bovril - Entre Ríos
Tel.: 03438-427073 | Alte. Brown 156

Fecha de emisión: 31/01/2021
Comprobante de pago n°: 00000097

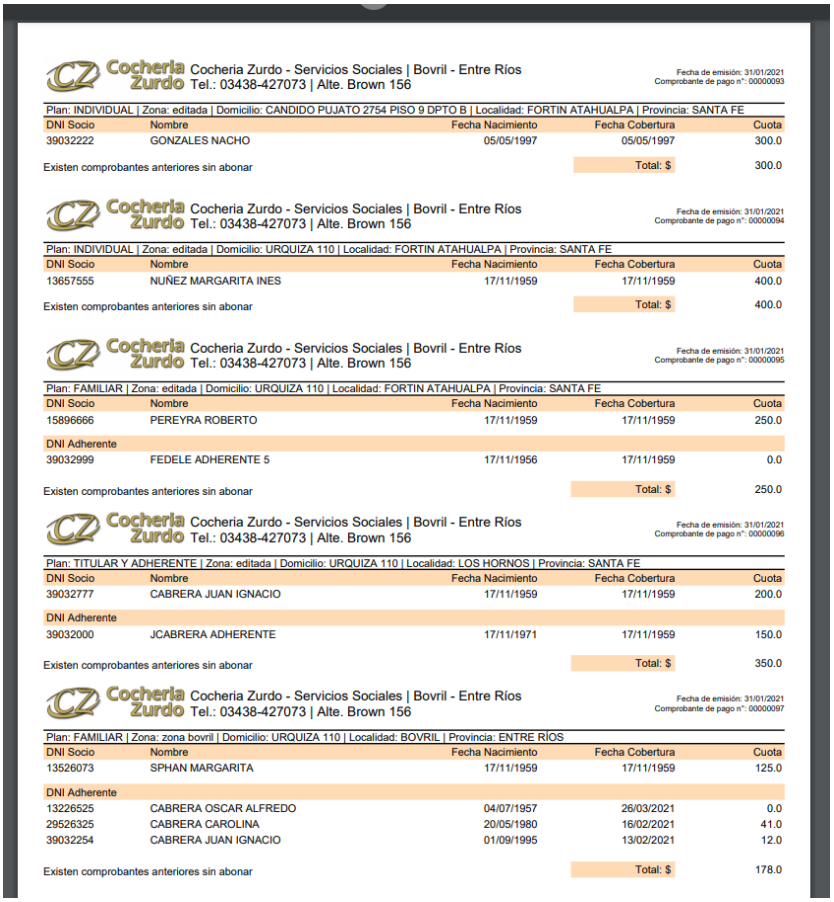
Plan: FAMILIAR | Zona: zona bovril | Domicilio: URQUIZA 110 | Localidad: BOVRIL | Provincia: ENTRE RÍOS

DNI Socio	Nombre	Fecha Nacimiento	Fecha Cobertura	Cuota
13526073	SPHAN MARGARITA	17/11/1959	17/11/1959	125.0
DNI Adherente				
13226525	CABRERA OSCAR ALFREDO	04/07/1957	26/03/2021	0.0
29526325	CABRERA CAROLINA	20/05/1980	16/02/2021	41.0
39032254	CABRERA JUAN IGNACIO	01/09/1995	13/02/2021	12.0
Existen comprobantes anteriores sin abonar			Total: $	178.0

Figura 48. Comprobante de pago - Sistema actual

Como se puede observar, la información está mejor estructurada, indicando información solamente relevante para el cliente que es a quién está destinado el documento. Además, mejora notablemente el diseño, se genera un documento digital con la posibilidad de enviar cada uno via email si el cliente lo requiere.

Además, es posible imprimir una cantidad de entre 5 a 8 comprobantes por hoja A4, reduciendo notablemente el costo, sumado a que puede emplearse mediante cualquier impresora existente en el mercado.



Cocheria Zurdo - Servicios Sociales | Bovril - Entre Ríos
Tel.: 03438-427073 | Alte. Brown 156

Fecha de emisión: 31/01/2021
Comprobante de pago n°: 00000093

Plan: INDIVIDUAL | Zona: editada | Domicilio: CANDIDO PUJATO 2754 PISO 9 DPTO B | Localidad: FORTIN ATAHUALPA | Provincia: SANTA FE

DNI Socio	Nombre	Fecha Nacimiento	Fecha Cobertura	Cuota
39032222	GONZALES NACHO	05/05/1997	05/05/1997	300.0
Existen comprobantes anteriores sin abonar			Total: $	300.0

Cocheria Zurdo - Servicios Sociales | Bovril - Entre Ríos
Tel.: 03438-427073 | Alte. Brown 156

Fecha de emisión: 31/01/2021
Comprobante de pago n°: 00000094

Plan: INDIVIDUAL | Zona: editada | Domicilio: URQUIZA 110 | Localidad: FORTIN ATAHUALPA | Provincia: SANTA FE

DNI Socio	Nombre	Fecha Nacimiento	Fecha Cobertura	Cuota
13657555	NUÑEZ MARGARITA INES	17/11/1959	17/11/1959	400.0
Existen comprobantes anteriores sin abonar			Total: $	400.0

Cocheria Zurdo - Servicios Sociales | Bovril - Entre Ríos
Tel.: 03438-427073 | Alte. Brown 156

Fecha de emisión: 31/01/2021
Comprobante de pago n°: 00000095

Plan: FAMILIAR | Zona: editada | Domicilio: URQUIZA 110 | Localidad: FORTIN ATAHUALPA | Provincia: SANTA FE

DNI Socio	Nombre	Fecha Nacimiento	Fecha Cobertura	Cuota
15896666	PEREYRA ROBERTO	17/11/1959	17/11/1959	250.0
DNI Adherente				
39032999	FEDELE ADHERENTE 5	17/11/1956	17/11/1959	0.0
Existen comprobantes anteriores sin abonar			Total: $	250.0

Cocheria Zurdo - Servicios Sociales | Bovril - Entre Ríos
Tel.: 03438-427073 | Alte. Brown 156

Fecha de emisión: 31/01/2021
Comprobante de pago n°: 00000096

Plan: TITULAR Y ADHERENTE | Zona: editada | Domicilio: URQUIZA 110 | Localidad: LOS HORNOS | Provincia: SANTA FE

DNI Socio	Nombre	Fecha Nacimiento	Fecha Cobertura	Cuota
39032777	CABRERA JUAN IGNACIO	17/11/1959	17/11/1959	200.0
DNI Adherente				
39032000	JCABRERA ADHERENTE	17/11/1971	17/11/1959	150.0
Existen comprobantes anteriores sin abonar			Total: $	350.0

Cocheria Zurdo - Servicios Sociales | Bovril - Entre Ríos
Tel.: 03438-427073 | Alte. Brown 156

Fecha de emisión: 31/01/2021
Comprobante de pago n°: 00000097

Plan: FAMILIAR | Zona: zona bovril | Domicilio: URQUIZA 110 | Localidad: BOVRIL | Provincia: ENTRE RÍOS

DNI Socio	Nombre	Fecha Nacimiento	Fecha Cobertura	Cuota
13526073	SPHAN MARGARITA	17/11/1959	17/11/1959	125.0
DNI Adherente				
13226525	CABRERA OSCAR ALFREDO	04/07/1957	26/03/2021	0.0
29526325	CABRERA CAROLINA	20/05/1980	16/02/2021	41.0
39032254	CABRERA JUAN IGNACIO	01/09/1995	13/02/2021	12.0
Existen comprobantes anteriores sin abonar			Total: $	178.0

Figura 49. PDF comprobantes de pago

4.13.7 Gestión de servicios

Para complementar todos los procesos de negocios de la organización y poder nuclear el gran porcentaje de los ingresos, el usuario dispondrá de un módulo para dar de alta y consultar los servicios extras al flujo principal, es decir, servicios aparte de los cuales ya cuenta el cliente al estar asociado a un plan.

El la siguiente imagen, se observa el formulario para dar de alta un servicio:

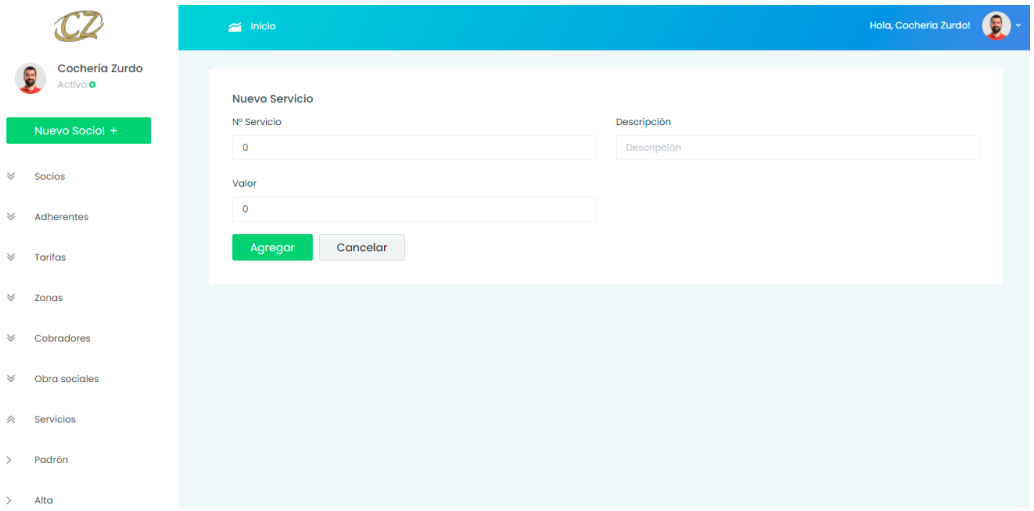


Figura 50. Formulario alta de un servicio

Además, como se detalló anteriormente, el usuario dispondrá de una pantalla donde podrá visualizar los servicios dados de alta y poder dar de baja aquel que ya no concuerde con el valor informado o ya no se usa:

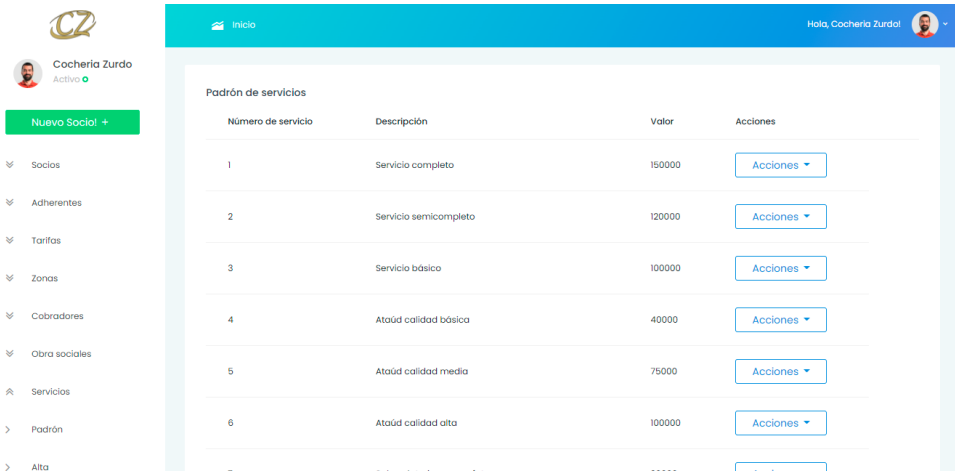


Figura 51. Padrón de servicios

4.13.8 Gestión de ventas

Para que los servicios entren en el circuito financiero de la organización, se desarrolló un módulo donde el usuario podrá dar de alta nuevas ventas seleccionando un servicio previamente definido.

En el mismo deberá asignar un número de venta a modo informativo o relacionado con algún otro parámetro dentro de la empresa, una descripción de la misma y la elección del servicio en el selector final.

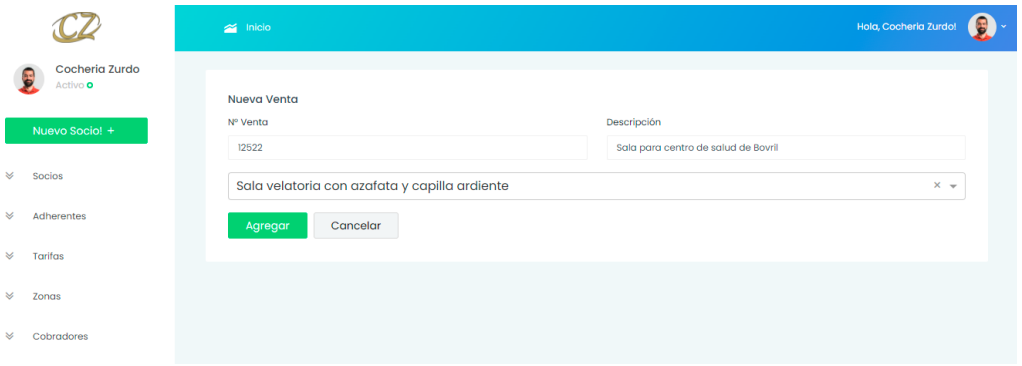


Figura 52. Formulario alta de una venta

Además, para llevar un control exacto de dicha entidad, se dispone de un componente donde se pueden visualizar todas las ventas realizadas mostrando el servicio, el valor y la fecha con la posibilidad de eliminar aquellas que fueron cargadas erróneamente.

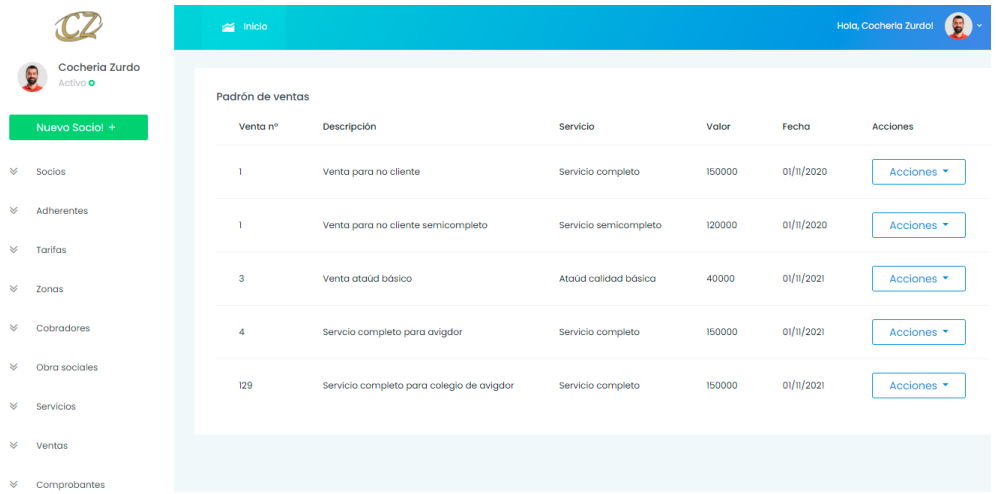


Figura 53. Padrón de ventas

4.13.9 Estadísticas del negocio

Las métricas son un módulo relevante en dicho sistema, ya que es información necesaria para una correcta toma de decisiones.

Al iniciar sesión, se presenta la interfaz principal de la aplicación, donde se visualizan cuatro cuadros principales que muestran los datos más relevantes de las áreas del negocio más importantes: Finanzas y Clientes.

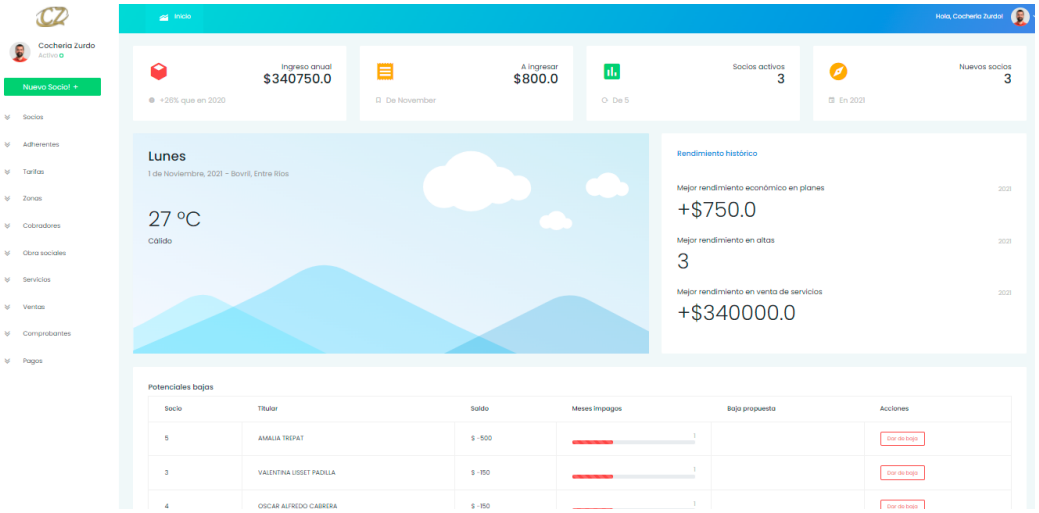


Figura 54. Portada del sistema - Métricas

En el primero, se puede observar cuánto dinero ha ingresado en la organización durante el año actual, con esta información, el usuario puede ver una comparación porcentual, si existiese, con el año anterior para comprobar que la decisiones a largo plazo de las finanzas son las correctas.

Por lo tanto, si existe información del año anterior, el sistema entregará dicha comparativa porcentualmente:

Figura 55. Métrica de comparación anual

En el siguiente cuadro, se ofrece una información más a corto/mediano plazo: dinero a ingresar en el mes actual, es decir, los comprobantes generados que aún no han sido abonados.

Figura 56. Métrica de futuro ingreso mensual

Con esta información, el usuario podrá tener un estimativo de cuánto dinero ingresará en la organización antes de que finalice el mes. Este dato tiene muchos objetivos, uno de los principales es brindarle cuanto egreso podrá generar teniendo en cuenta el ingreso, evitando un déficit.

En las siguientes dos métricas, el usuario podrá visualizar información acerca de los clientes registrados en el sistema.

En el tercer cuadro, se pueden observar los socios activos del total de socios registrados. Con este número, el usuario podrá corroborar en el listado de clientes el motivo de baja principal y luego poder tomar una decisión en base a dicho dato.

Figura 57. Métrica de socios activos

En el cuarto y último cuadro, se visualiza la cantidad de nuevos socios registrados en el año actual.

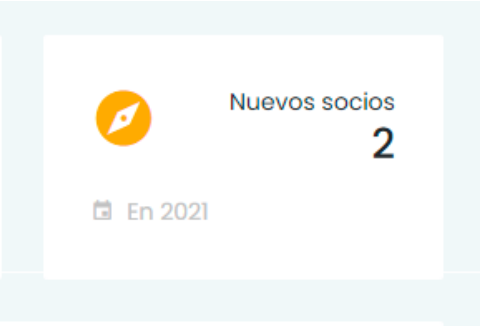


Figura 58. Métrica de nuevos socios

Con este dato, se puede corroborar si se cumplió con el objetivo anual que se impone la organización de reclutar nuevos clientes.

En la siguiente sección, se encuentran dos componentes: Uno que indica la fecha y el clima, diseño que se construyó para ganar amigabilidad y, en el siguiente, se muestra información histórica del negocio.

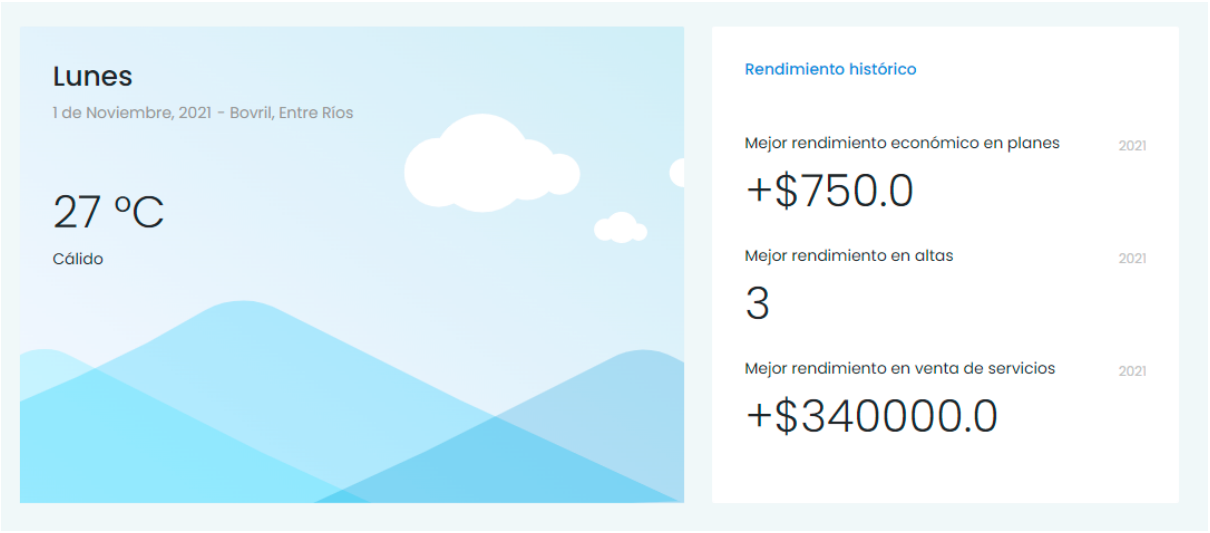


Figura 59. Fecha, clima y rendimientos históricos

Como se puede observar, se visualizan dos datos históricos de métricas relevantes al negocio. Además del valor obtenido, se muestra el año en el que se consiguió el mejor rendimiento.

A su vez, en la parte inferior de la portada se puede visualizar una tabla donde se muestran los clientes que poseen la mayor cantidad de comprobantes sin abonar.

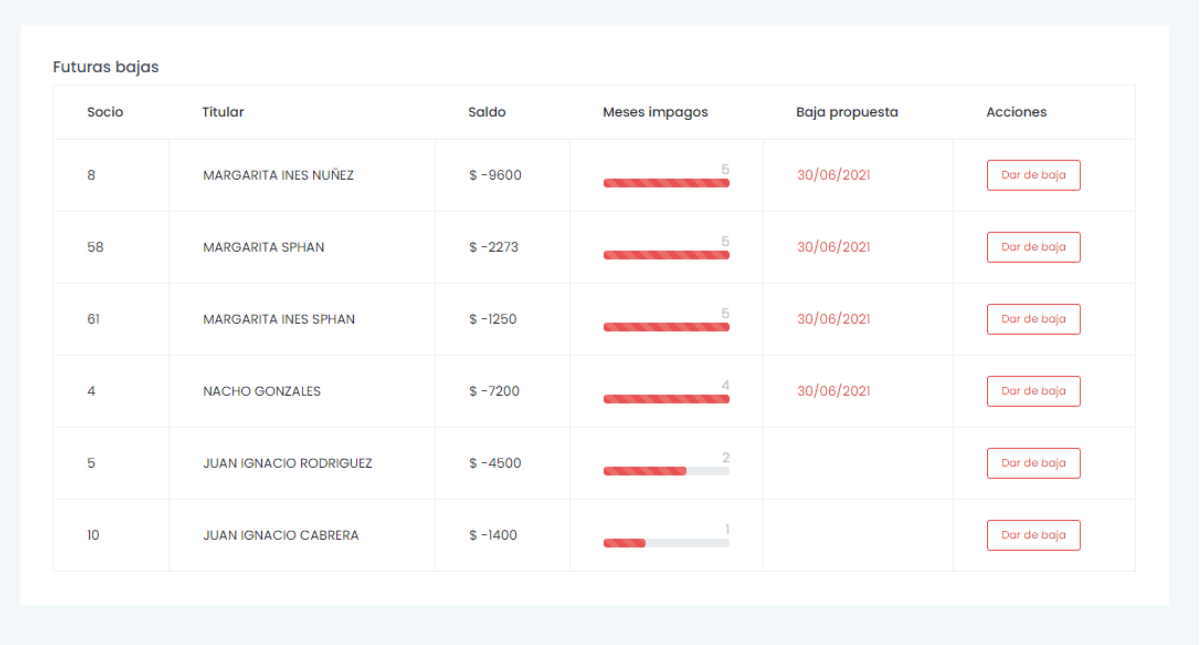


Figura 60. Tabla de potenciales bajas por deuda

Siguiendo la idea tradicional del negocio, si un cliente posee una deuda de tres meses o más, el mismo será dado de baja por morosidad.

Para aumentar la flexibilidad y tener en cuenta a clientes que abonan varios meses juntos ya que viven en zonas alejadas de una oficina y es una gran dificultad llegar a una, es el usuario quién deberá realizar la acción de dar de baja a los clientes morosos.

Por lo tanto, en la tabla se muestra el saldo que el cliente adeuda y cuantos meses o comprobantes posee sin pagar, sumando a una fecha propuesta de baja, que es la suma de tres meses a la fecha de emisión del primer comprobante adeudado.

Reportes

Además, para sumar información que siga ayudando a la toma de decisiones, el usuario podrá generar reportes asociados a métricas más utilizadas en el negocio:

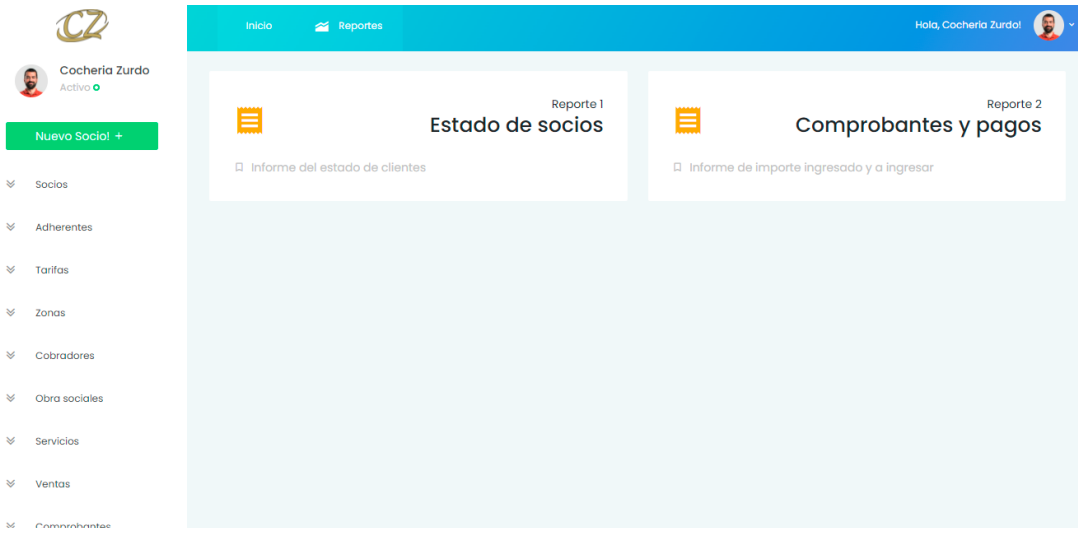
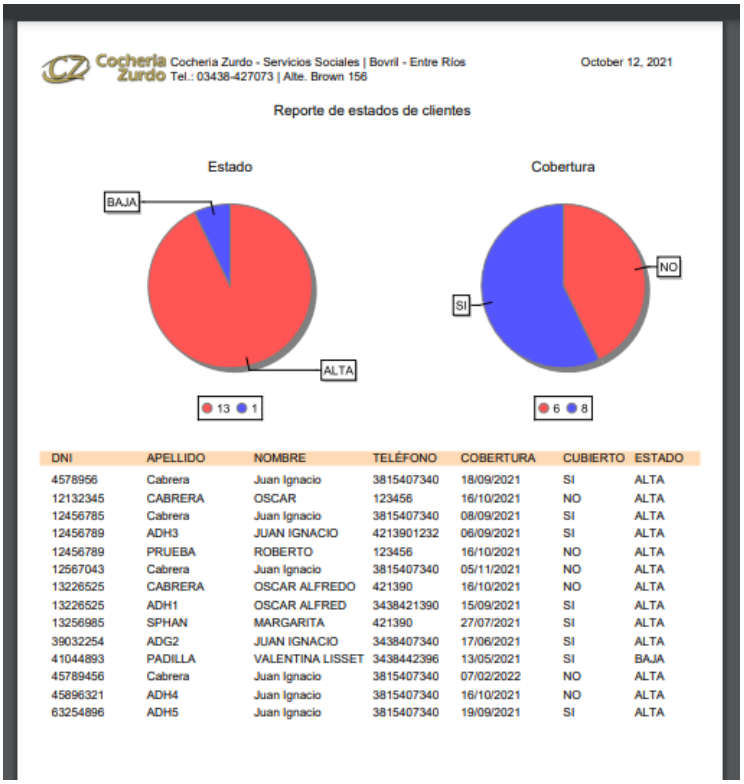


Figura 61. Interfaz de reportes

En primer lugar, es posible construir un reporte donde se muestre el estado de los clientes, es decir, la cantidad de socios y adherentes dados de alta y aquellos que estén dados de baja sumado a si ya poseen cobertura o todavía no se alcanzó la fecha requerida. En la siguiente imagen se puede observar el informe como resultado:



DNI	APELLIDO	NOMBRE	TELÉFONO	COBERTURA	CUBIERTO	ESTADO
4578956	Cabrera	Juan Ignacio	3815407340	18/09/2021	SI	ALTA
12132345	CABRERA	OSCAR	123456	16/10/2021	NO	ALTA
12456785	Cabrera	Juan Ignacio	3815407340	08/09/2021	SI	ALTA
12456789	ADH3	JUAN IGNACIO	4213901232	06/09/2021	SI	ALTA
12456789	PRUEBA	ROBERTO	123456	16/10/2021	NO	ALTA
12567043	Cabrera	Juan Ignacio	3815407340	05/11/2021	NO	ALTA
13226525	CABRERA	OSCAR ALFREDO	421390	16/10/2021	NO	ALTA
13226525	ADH1	OSCAR ALFRED	3438421390	15/09/2021	SI	ALTA
13256985	SPHAN	MARGARITA	421390	27/07/2021	SI	ALTA
39032254	ADG2	JUAN IGNACIO	3438407340	17/06/2021	SI	ALTA
41044893	PADILLA	VALENTINA LISSET	3438442396	13/05/2021	SI	BAJA
45789456	Cabrera	Juan Ignacio	3815407340	07/02/2022	NO	ALTA
45896321	ADH4	Juan Ignacio	3815407340	16/10/2021	NO	ALTA
63254896	ADH5	Juan Ignacio	3815407340	19/09/2021	SI	ALTA

Figura 62. Reporte de estados de clientes

Dada esta información, es posible obtener una medida porcentual o proporcional del estado de los clientes dado el total de los mismos.

En un enfoque más financiero, es posible generar un reporte donde se indiquen el total de comprobantes emitidos en el corriente mes visualizando proporcionalmente cuantos fueron abonados y cuantos no, sumado al total de dinero que ingresó y falta por ingresar.

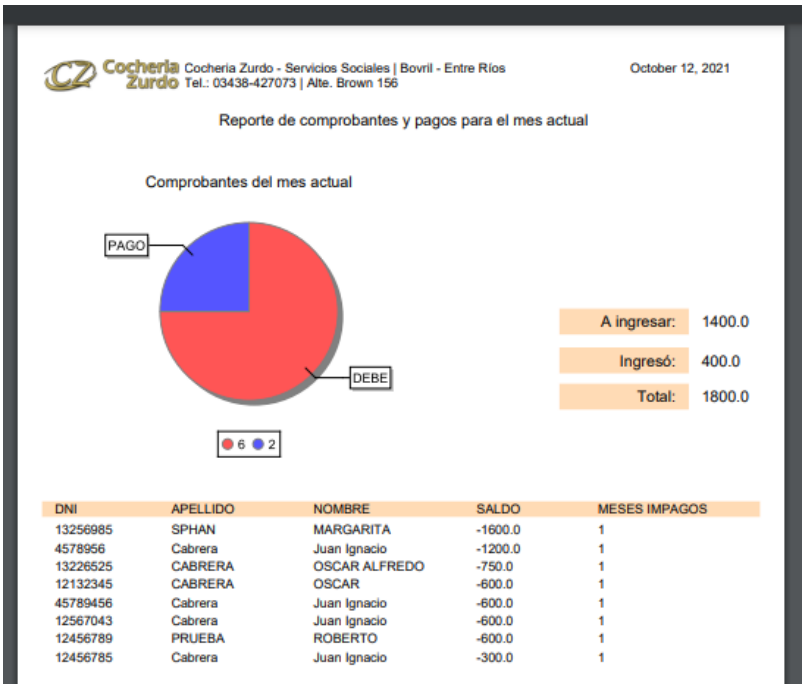


Figura 63. Reporte de comprobantes y pagos del mes actual

Se utilizaron gráficos circulares para mostrar la información, mejorando la visualización de las proporciones. Además, toda la información se representa en la tabla final que muestra con mayor detalles los valores obtenidos, brindando la posibilidad de conocer los deudores que presenten un saldo mayor a cubrir.

4.14 Validaciones en formularios

Las validaciones son un elemento clave en todo formulario para evitar la inconsistencia de datos. Cuando se introduce información al sistema, existe una gran posibilidad de que el usuario final la ingrese erróneamente, es por esto que es necesario generar validaciones en cada uno de los campos de los formularios.

A continuación, se presentan algunas de las validaciones:

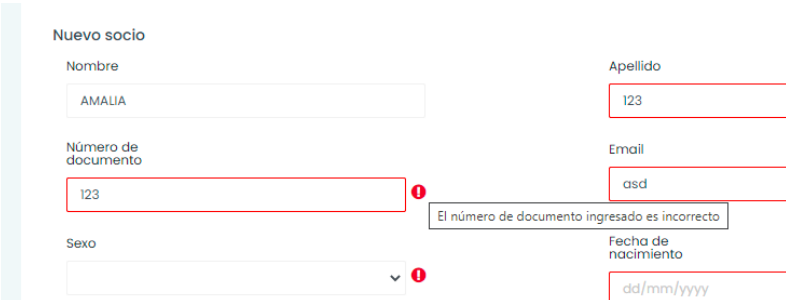


Figura 64. Mensaje de error por validación en campo numérico

Como se puede observar, un número de documento en Argentina cuenta con un máximo de 8 dígitos y un mínimo de 7, por lo tanto cuando el usuario ingrese un número que no cumpla con estas restricciones, se le mostrará un mensaje de error. Además, si el mismo ingresa letras o deja en blanco el campo, también aparecerá el error para que el usuario revise dicha información.

En los campos que contengan email, es requerido que se ingrese el símbolo arroba (@) seguido de un punto.

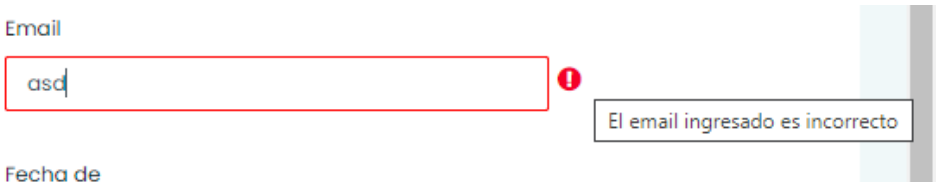


Figura 65. Mensaje de error por validación en email

Cabe aclarar que todos los campos que son requeridos en el modelo de datos, renderizan el mensaje de error cuando el usuario deja en blanco el mismo.

4.15 Encriptación de información sensible

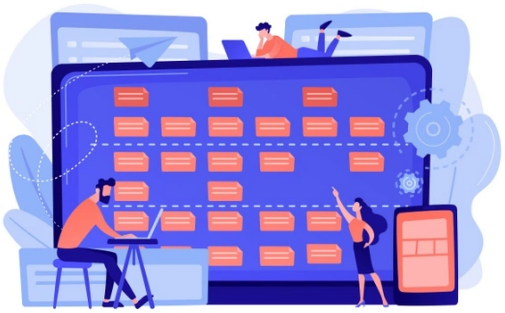
La finalidad de encriptar la información sensible es mejorar y proteger la confidencialidad de los datos de clientes para evitar que puedan visualizarse sin

problemas ante un ataque o un acceso indebido, apuntando a mejorar la seguridad del aplicativo, como se decidió en la definición de los atributos de calidad.

Toda esta característica se apoya sobre la Ley de Protección de datos personales, n° 25.326, que indica que se debe garantizar el derecho al honor y a la intimidad de las personas. Las definiciones que son alcanzadas por dicha ley son las siguientes:

1. Datos personales: Información de personas físicas.
2. Datos sensibles: Datos que revelan origen racial, político, filosófico, cuestiones de salud o vida sexual.
3. Banco de datos: Conjunto de datos destinados al procesamiento.
4. Tratamientos de datos: Operaciones y procedimientos sistemáticos.

Apoyándonos en este conjunto de definiciones, se puede determinar que nuestra aplicación interactúa en su totalidad con información sensible, por lo cual es correcto ocultar los datos que podrían generar problemas si los mismos son accedidos.



En principio el alcance de esta característica es proteger los datos de enfermedades o patologías preexistentes, que es un dato necesario en el negocio para generar un cálculo de cobertura correcto.

Si bien el mejor escenario sería encriptar todos los datos personales del usuario, por una cuestión de alcance, se redujo a ocultar la información que se detalló previamente.

Técnicamente, se utilizó un patrón de diseño denominado Converter, que proporciona una forma fácil y genérica de convertir datos de manera bidireccional, permitiendo una implementación rápida y limpia.

En principio, se define una semilla que se utilizará para encriptar el atributo de la clase que estén anotadas. En cada atributo de la clase que se desea cifrar, se debe especificar la anotación @Convert indicando la clase a convertir previamente definida. Esto hará que cada vez que viaje información hacía la base de datos, se

ejecutará el convertidor con la semilla dada y persistirá una cadena imposible de entender ante un usuario malintencionado que acceda a la base de datos. Así mismo, cuando el aplicativo recupera información de la base de datos, primero verifica si el atributo fue previamente encriptado, si es así, ejecuta el proceso inverso con la semilla y entrega el dato sin cifrar.

Por lo tanto, tanto las patologías preexistentes como las defunciones son cadenas ilegibles en la base de datos que ante un acceso indebido por un usuario malintencionado, no podrá descifrar la información. Gracias a esta característica, se aumenta notablemente la seguridad de la información e integridad de los datos, entregando un sistema más robusto.

[20] Ley 25.326. Protección de datos personales. http://servicios.infoleg.gob.ar/

5. Extensibilidad

El alcance original de la solución fue revisado de modo tal de cumplir con los procesos de negocios más centrales de la organización, es decir, funcionalidades para la administración de asociados, manejo de comprobantes y pagos de los mismos entre otros, pero la empresa en sí posee otros procesos que, si bien no fueron diagramados en este sistema, es totalmente factible agregarlos en entregas futuras.

En principio, uno de los mayores ingresos como atracción de nuevos socios son los consultorios médicos. La sede central de la organización cuenta con diferentes consultorios donde asisten especialistas médicos de diferentes partes de la provincia en el cual los asociados cuentan con una bonificación para asistir a consultas y estudios con los mismos. Es por esto que una propuesta en agenda es poder sumarle al sistema un módulo que maneje los turnos y que verifique si es asociado o no para generarle una bonificación, así como también que cada médico lleve una historia clínica para que la empresa contenga esa información también y pueda ser compartida entre los diferentes especialistas. Esto también genera una base de información para diferentes métricas que se mostrarían en la página principal del sistema.

Otro de los procesos importantes y de gran auge en estos últimos tiempos es el traslado de pacientes a diferentes entidades médicas de la provincia para la realización de diferentes estudios.

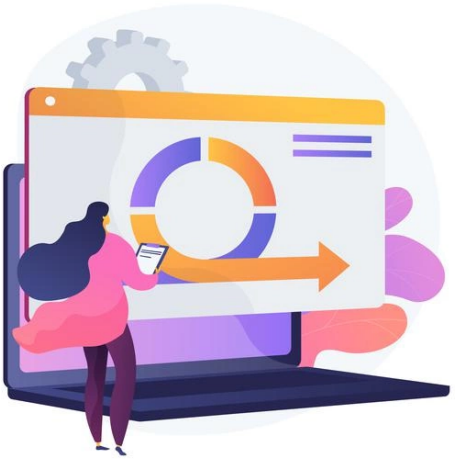
La empresa cuenta con una flota de vehículos capacitados para el traslados de pacientes, por ejemplo, oncológicos para realizar quimioterapia o que deban llevar a cabo un estudio que no sea posible en la ciudad. Es por esto que se podría llevar registros de todos los asociados y no asociados que se trasladaron y el motivo

de su movilización para luego brindar esta información al usuario para una mejor toma de decisiones.

Haciendo hincapié en la curva de utilización de dispositivos móviles se pensó en la realización de una aplicación móvil para que el asociado pueda consultar su estado ante la empresa, es decir, si adeuda, si está activo o no por tal motivo, si posee o no cobertura, así como también sacar turnos para los consultorios médicos sin tener que comunicarse con la organización para que luego mediante una notificación se le recuerde la asistencia al médico.

Este aplicativo también contaría con un módulo para los especialistas que acuden a los consultorios para notificarles cuántos pacientes deberá atender en el día y quienes, con el objetivo de recordarle si debe llevar algún estudio ya que generalmente provienen de otras ciudades de la provincia.

A su vez, se pensó en contar con la implementación de una pasarela de pago mediante la aplicación móvil para facilitar al asociado abonar mediante tarjetas de crédito sin tener que acudir a la sede.



Otro de los puntos importantes y que se utilizan comúnmente en prepagas y otros organismos relacionados con la salud es poder brindarle al asociado una credencial. De este modo la empresa podría generar convenios con diferentes organizaciones para brindarles descuentos a los asociados. Es por esto que se piensa en desarrollar un módulo que se ocupe de la generación digital de credenciales y envío por cualquier medio electrónico y hasta una posible impresión para que el asociado la lleve físicamente con el.

Además, se notó que gran porcentaje de los muertos se deben a enfermedades cardíacas, gracias a esta información, se podría extender que todos

los clientes con este tipo de enfermedad sean notificados una vez por mes para un control en los propios consultorios de la organización o si ya asistió a los consultorios y fue medicado, notificar periódicamente para que no olvide de tomar la medicación recetada.

6. Conclusiones

Ya culminado todo el proceso de análisis, diseño, desarrollo e implementación, se concluye que se logró materializar un producto de calidad, mejorando ampliamente todos los procesos y generando un mayor orden de la información para una mejor toma de decisiones.

Haciendo hincapié en todo el proceso de renovación, se consiguió un aplicativo más moderno y amigable que el sistema anterior que manejaba la organización, mejorando ampliamente la usabilidad, amigabilidad, seguridad y escalabilidad.

Las contadas entrevistas con el cliente fueron importantes para mejorar la comunicación, el relevamiento de requerimientos y el poder de negociación de las partes. Si bien por la pandemia que atraviesa gran parte del mundo, la asistencia a las oficinas fue en pocas ocasiones, donde igualmente se trató de estar en contacto directo con él y que sea una parte más de todo el ciclo de vida del desarrollo del aplicativo.

La pandemia fue un condimento importante en todo el proceso de desarrollo. A nivel organización, se necesitaba mejorar la administración y el control ya que el aumento de muertes fue superior al de años anteriores. A nivel equipo, se había diagramado una reunión semanal para ver los avances, donde este proceso sufrió un cambio sobre la marcha, evitando realizar la misma. Si bien no todo fue negativo por las restricciones que existieron, ya que se logró un producto de calidad con muy pocas reuniones presenciales.

El crecimiento como profesionales fue alto, ya que se aplicó todo el conjunto de herramientas que nos brindó el cursado durante estos años y poder generar un producto utilizado en el mercado actual, es muy satisfactorio.

7. Soporte bibliográfico

La mayoría de las fuentes se refieren a foros de programación tales como documentaciones oficiales de frameworks, plantillas de interfaces y foros de especialistas en software. Además, se ha aprovechado del sistema anterior para obtener las principales ideas del negocio en sí. Para la documentación y diseño arquitectónico se ha revisado la bibliografía otorgada por los diferentes cursos del campus virtual. Se ha investigado de manera adicional las métricas utilizadas para determinar indicadores importantes en las funerarias o empresas aseguradoras de la salud.

Principalmente, los contenidos teóricos referidos al negocio, se han obtenido a partir de la comunicación directa con el cliente.

Cabe aclarar que la mayoría de los conocimientos sobre las tecnologías, ya se conocían de antemano debido a la experiencia laboral de los respectivos responsables del presente proyecto.

Se listan a continuación fuentes bibliográficas del contexto descripto anteriormente:

- https://spring.io/projects/spring-boot
- https://angular.io/docs
- https://ng-bootstrap.github.io
- https://stackoverflow.com/
- https://www.argentina.gob.ar/salud
- https://www.eclemma.org/jacoco/
- https://www.usability.gov/how-to-and-tools/methods/use-cases.html
- https://normas-apa.org/
- Ingeniería del software: un enfoque práctico. Roger S. Pressman
- Ingeniería del Software 7ma edición. Ian Sommerville

8. Anexo

8.1 Diagramas

8.1.1 Diagramas de Casos de Uso

Figura 66. Usuario no autenticado

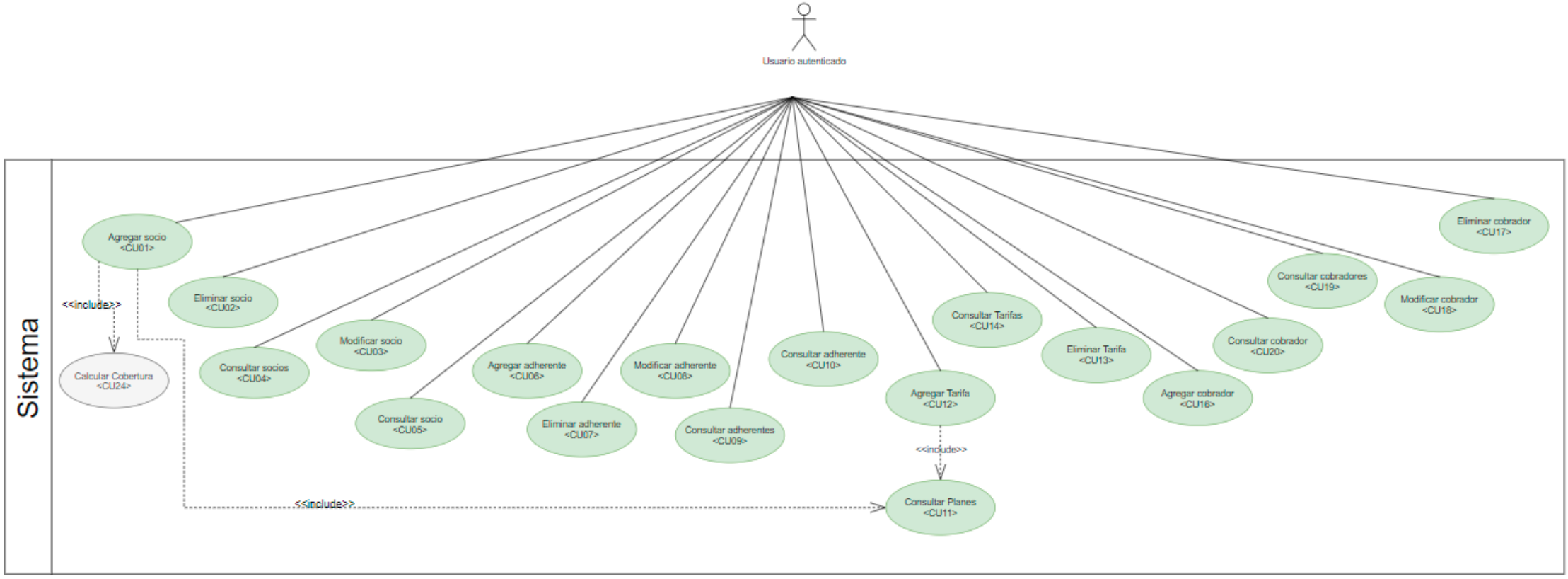


Figura 67. Usuario autenticado - Incremento 1

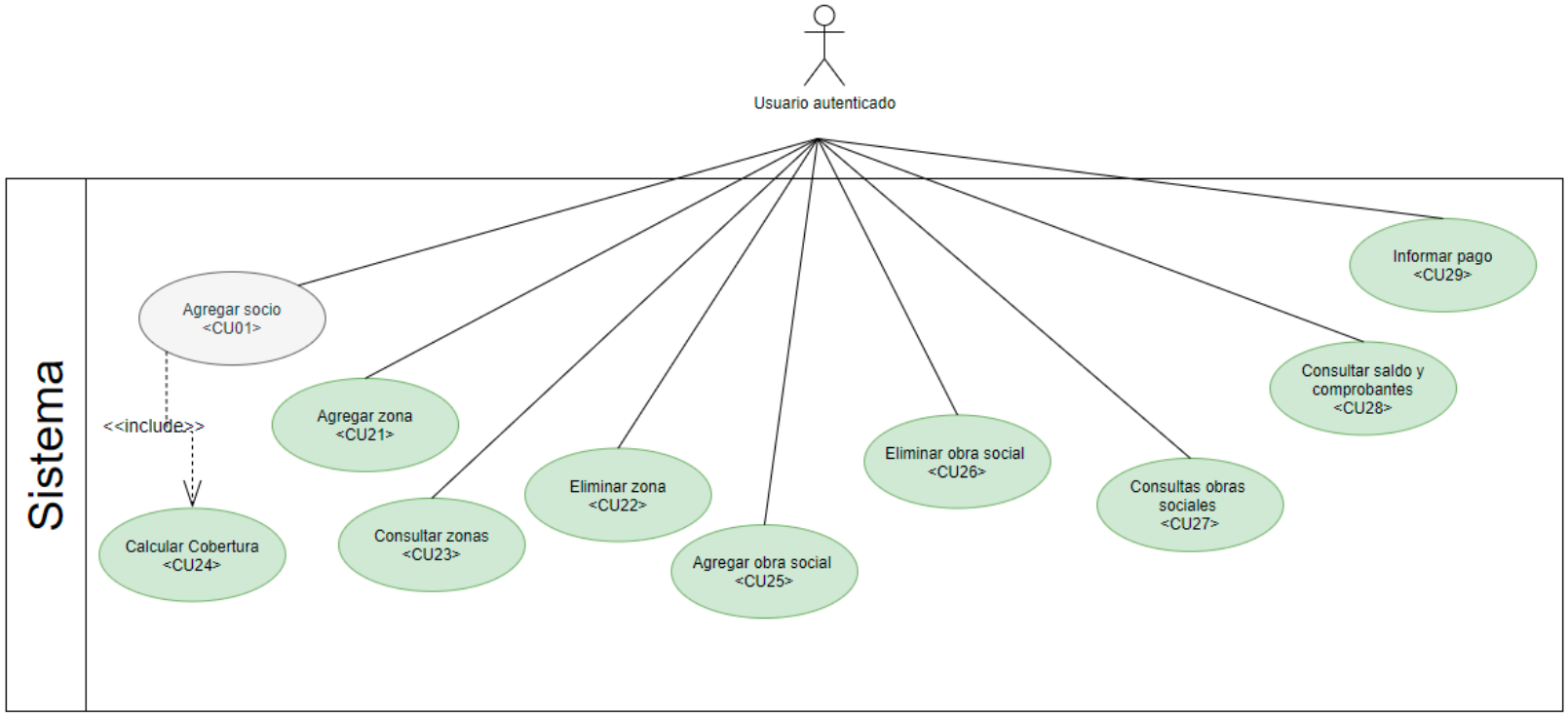


Figura 68. Usuario autenticado - Incremento 2

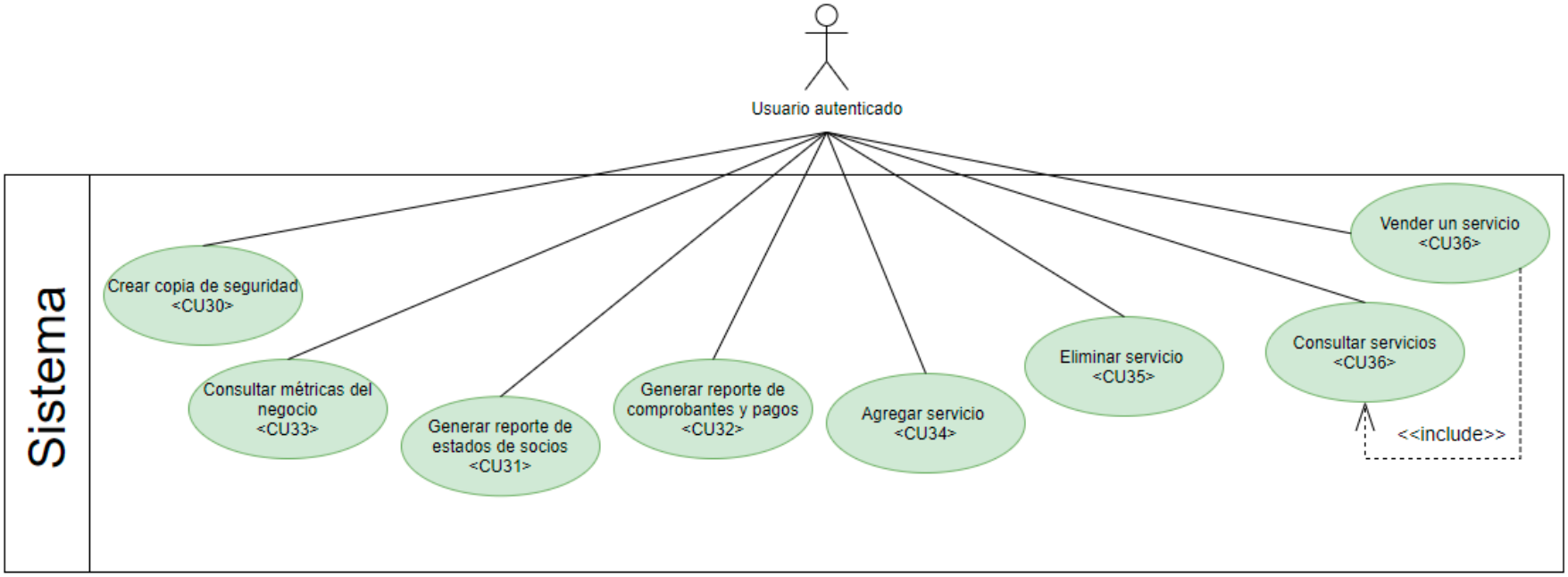
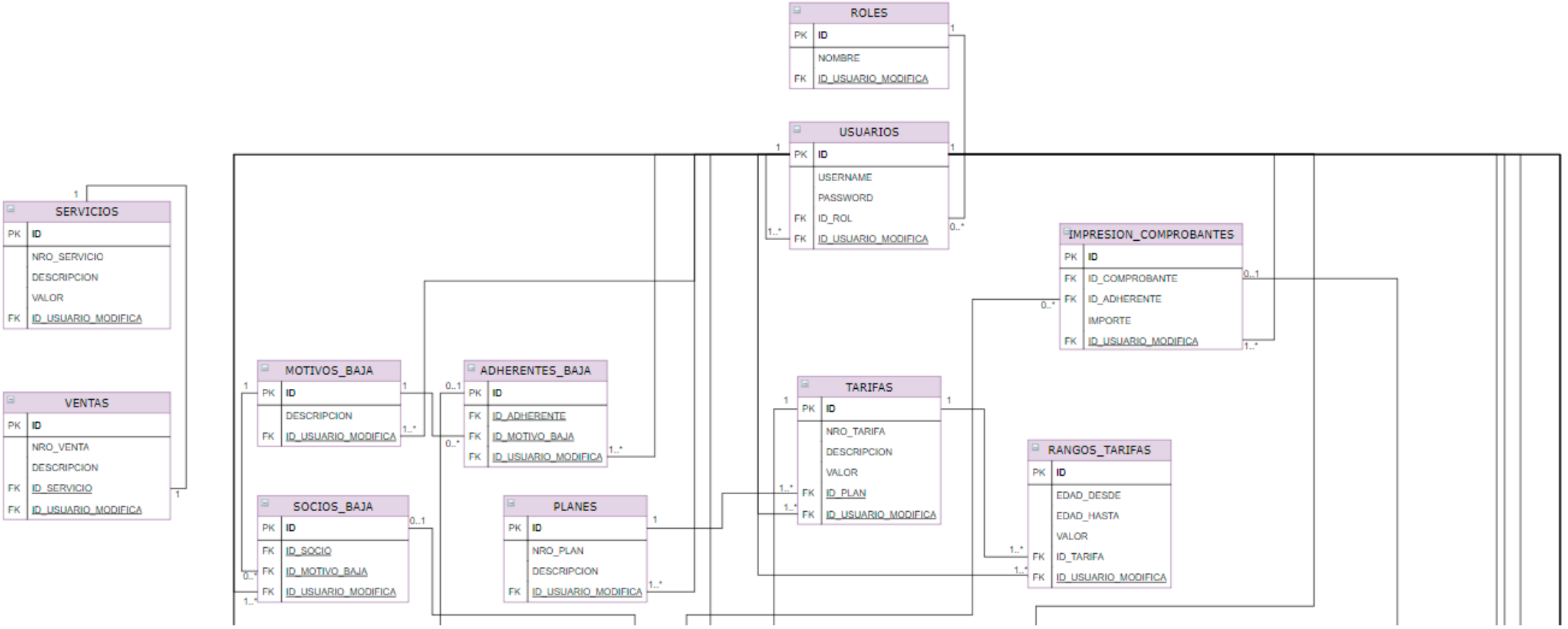


Figura 69. Usuario autenticado - Incremento 3

8.1.2 Diagrama de Entidad-Relación



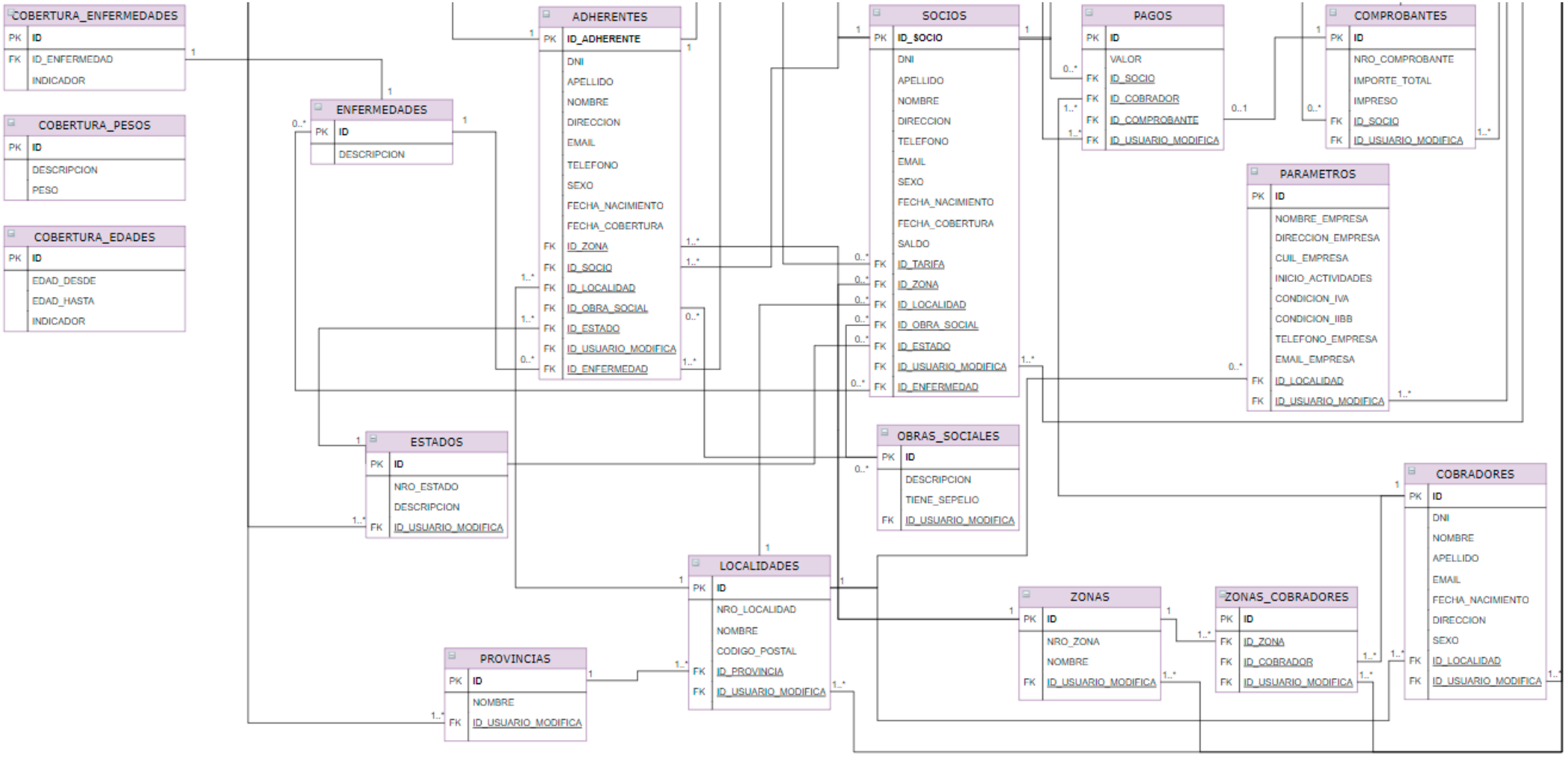


Figura 70. Diagrama de entidad - relación

8.2 Especificación de casos de uso

A continuación, se detalla la especificación de cada caso de uso siguiendo el modelo planteado.

8.2.1 Requerimientos Funcional 1

El sistema permitirá agregar, actualizar, eliminar y consultar socios.

<table>
<tr><td>Nombre</td><td colspan="2">Agregar socio</td><td>ID</td><td>CU01</td></tr>
<tr><td>Descripción</td><td colspan="4">El sistema permitirá agregar un socio.</td></tr>
<tr><td>Actores</td><td colspan="4">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◉ Alta ◎ Media ◎ Baja</td><td>Complejidad</td><td>◎ Alta ◉ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="4">➔ No existe socio que represente a la persona que se intenta agregar.</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="2">Fracaso</td></tr>
<tr><td colspan="2">➔ Socio almacenado en base de datos con los datos especificados.</td><td colspan="2">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="2">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario escoge la opción “Agregar” en la lista desplegable de socio.</td><td colspan="2"></td></tr>
<tr><td colspan="3">2. Se completa la totalidad de estos campos:
● DNI *
● Nombres *
● Apellidos *
● Fecha de nacimiento *
● Número de celular *
● Correo electrónico
● Obra social *
● Plan *
● Tarifa *
● Sexo *
● Zona *
● Localidad *
● Domicilio *
● Adherente (#REF: CU7)</td><td colspan="2">2.1. El usuario desea cancelar la operación.
2.1.1 Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.</td></tr>
<tr><td colspan="3">3. El usuario presiona el botón “Aceptar”</td><td colspan="2">3.1. El usuario desea cancelar la operación.
3.1.1 Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.</td></tr>
</table>

4. El sistema valida que todos los campos obligatorios (*) han sido ingresados y que el valor de cada uno de los campos no vacíos (requeridos o no) sea correcto.	**4.1.** Ya existe un socio con el DNI ingresado. **4.1.1** Se visualiza un mensaje de error describiendo explícitamente que ese socio ya existe en la base de datos. **4.1.2** Se retorna al paso **2**. **4.2.** Un campo no corresponde al tipo y formato esperado. **4.2.1** Se visualiza un mensaje de error describiendo explícitamente cuál es el campo involucrado en el error y cuál es la causa. **4.2.2** Se retorna al paso **2**.
Flujo de excepciones	➔ NullPointerException: Se detiene el flujo y se detalla cuál es el objeto sin instanciar en el sistema mediante un mensaje de error. Se retorna el flujo al paso **2**.
Asociaciones de inclusión	➔ CU11, CU14, CU23, CU24, CU27
Asociaciones de extensión	➔ CU06
Requerimientos especiales	➔
Observaciones	

Nombre	Eliminar socio	**ID**	CU02
Descripción	El sistema permitirá eliminar un socio. Es decir, presentar la baja de un socio.		
Actores	➔ Usuario final		
Prioridad	◉ Alta ◎ Media ◎ Baja	**Complejidad**	◎ Alta ◉ Media ◎ Baja
Precondiciones	➔ Socio existente con los datos especificados.		
Postcondiciones	Éxito: ➔ Eliminación lógica del socio.		Fracaso: ➔ Visualización de un mensaje de error que describe la causa del fallo.

Flujo Principal	**Flujo Alternativo**
3. El usuario selecciona la opción “Eliminar” en el padrón de socios.	
4. Se solicita la elección en una lista desplegable del motivo. ● Motivo: ○ Fallecimiento ○ Morosidad ○ Renuncia	
5. El usuario presiona el botón “Eliminar”.	**3.1.** El usuario desea cancelar la operación. **3.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
6. El sistema presenta un mensaje de confirmación para eliminar el socio indicado.	
7. El usuario escoge la opción “Aceptar”.	**5.1.** El usuario desea cancelar la operación. **5.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
8. El sistema presenta un mensaje de eliminación exitosa.	**6.1.** El usuario desea cancelar la operación. **6.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.

Flujo de excepciones	➔ NullPointerException: Se detiene el flujo y se detalla cuál es el objeto sin instanciar en el sistema mediante un mensaje de error. Se finaliza el flujo.
Asociaciones de inclusión	➔
Asociaciones de extensión	➔

Requerimientos especiales	➔
Observaciones	

Nombre	Modificar socio	**ID**	CU03
Descripción	El sistema permitirá editar la información de un socio.		
Actores	➔ Usuario final		
Prioridad	◉ Alta ◎ Media ◎ Baja	**Complejidad**	◎ Alta ◉ Media ◎ Baja
Precondiciones	➔ Socio existente con los datos especificados.		
Postcondiciones	Éxito: ➔ Socio con datos actualizados en base de datos.	Fracaso: ➔ Visualización de un mensaje de error que describe la causa del fallo.	

Flujo Principal	**Flujo Alternativo**
1. El usuario selecciona la opción “Modificar” de un socio del listado.	
2. Se presenta una pantalla con la información actual del socio.	
3. Se completa la totalidad de estos campos: ● DNI * ● Nombres * ● Apellidos * ● Fecha de nacimiento * ● Número de teléfono * ● Correo electrónico ● Obra social * ● Plan * ● Tarifa * ● Sexo * ● Zona * ● Localidad * ● Domicilio *	**3.1.** El usuario desea cancelar la operación. **3.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
4. El usuario presiona el botón “Aceptar”.	**4.1.** El usuario desea cancelar la operación. **4.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
5. El sistema valida que todos los campos obligatorios (*) han sido ingresados y que el valor de cada uno de los campos no vacíos (requeridos o no) sea correcto.	**5.1.** Ya existe un socio con el DNI ingresado. **5.1.1** Se visualiza un mensaje de error describiendo explícitamente que ese socio ya existe en la base de datos. **5.1.2** Se retorna al paso **2**.

	5.2. Un campo no corresponde al tipo y formato esperado. **5.2.1** Se visualiza un mensaje de error describiendo explícitamente cuál es el campo involucrado en el error y cuál es la causa. **5.1.2** Se retorna al paso **2**.
Flujo de excepciones	➔ NullPointerException: Se detiene el flujo y se detalla cuál es el objeto sin instanciar en el sistema mediante un mensaje de error. Se retorna el flujo al paso **2**.
Asociaciones de inclusión	➔ CU11, CU14, CU23, CU27
Asociaciones de extensión	➔
Requerimientos especiales	➔
Observaciones	

<table>
<tr><td>Nombre</td><td colspan="3">Consultar socios</td><td>ID</td><td>CU04</td></tr>
<tr><td>Descripción</td><td colspan="5">El sistema permitirá buscar y listar socios.</td></tr>
<tr><td>Actores</td><td colspan="5">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◉ Alta ◎ Media ◎ Baja</td><td>Complejidad</td><td colspan="2">◎ Alta ◉ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="5">➔</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="3">Fracaso</td></tr>
<tr><td colspan="2">➔ Visualización de una lista de socios.</td><td colspan="3">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="3">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario selecciona la opción “Listar socios”.</td><td colspan="3"></td></tr>
<tr><td colspan="3">2. El sistema presenta una tabla de todos los socios con la siguiente información:
● DNI
● Apellido
● Nombre
● Cobertura
● Estado</td><td colspan="3">2.1. No existen socios.
2.1.1 Se muestra un mensaje en el dashboard “No existen socios.”. Finaliza el CU.</td></tr>
<tr><td colspan="3">3. El sistema permite la opción de filtrar por DNI, Nombre y Apellido</td><td colspan="3"></td></tr>
<tr><td colspan="3">4. El usuario selecciona una acción sobre un socio:
a. Información (#REF CU05)
b. Dar de baja (#REF CU02)
c. Modificar (#REF CU03)</td><td colspan="3"></td></tr>
<tr><td>Flujo de excepciones</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="5">➔ CU02, CU03, CU05</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="5">➔</td></tr>
<tr><td>Observaciones</td><td colspan="5"></td></tr>
</table>

<table>
<tr><td>Nombre</td><td colspan="3">Consultar socio</td><td>ID</td><td>CU05</td></tr>
<tr><td>Descripción</td><td colspan="5">El sistema permitirá brindar información específica de un socio.</td></tr>
<tr><td>Actores</td><td colspan="5">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◉ Alta ◎ Media ◎ Baja</td><td>Complejidad</td><td colspan="2">◎ Alta ◉ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="5">➔ Socio existente en la base de datos.</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="3">Fracaso</td></tr>
<tr><td colspan="2">➔ Visualización de una ficha con toda la información de un socio.</td><td colspan="3">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="3">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El sistema presenta una ficha con toda la información del socio.</td><td colspan="3">1.1. No existe socio.
1.1.1 Se muestra un mensaje en el dashboard “No existen socios.”. Finaliza el CU.</td></tr>
<tr><td colspan="3">2. Se visualiza la siguiente información del socio:
● ID socio
● DNI
● Nombres
● Apellidos
● Fecha de nacimiento
● Fecha de cobertura
● Número de teléfono
● Correo electrónico
● Obra social
● Plan
● Tarifa
● Sexo
● Zona
● Localidad
● Domicilio</td><td colspan="3"></td></tr>
<tr><td>Flujo de excepciones</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="5">➔</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="5">➔</td></tr>
<tr><td>Observaciones</td><td colspan="5"></td></tr>
</table>

8.2.2 Requerimientos Funcional 2

El sistema permitirá agregar, actualizar, eliminar y consultar adherentes. Cada adherente está asignado a un socio.

Nombre	Agregar adherente	**ID**	CU06
Descripción	El sistema permitirá agregar un adherente y relacionarlo con un socio.		
Actores	➔ Usuario final		
Prioridad	◉ Alta ◎ Media ◎ Baja	**Complejidad**	◎ Alta ◉ Media ◎ Baja
Precondiciones	➔ Existencia de un socio para relacionarse con la nueva entidad.		
Postcondiciones	Éxito: ➔ Adherente almacenado en base de datos y relacionado con un socio.	Fracaso: ➔ Visualización de un mensaje de error que describe la causa del fallo.	

Flujo Principal	Flujo Alternativo
1. El usuario selecciona la opción “Agregar” en la sección adherente.	
2. Se completa la totalidad de estos campos: ● DNI * ● Nombres * ● Apellidos * ● Fecha de nacimiento * ● Número de celular * ● Correo electrónico ● Obra social * ● Sexo * ● Zona * ● Localidad * ● Domicilio *	**2.1.** El usuario desea cancelar la operación. **2.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
3. El usuario presiona el botón “Aceptar”	**3.1.** El usuario desea cancelar la operación. **3.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
4. El sistema valida que todos los campos obligatorios (*) han sido ingresados y que el valor de cada uno de los campos no vacíos (requeridos o no) sea correcto.	**4.1.** Ya existe un adherente con el DNI ingresado. **4.1.1** Se visualiza un mensaje de error describiendo explícitamente que ese socio ya existe en la base de datos.

	4.1.2 Se retorna al paso **2**. **4.2.** Un campo no corresponde al tipo y formato esperado. **4.2.1** Se visualiza un mensaje de error describiendo explícitamente cuál es el campo involucrado en el error y cuál es la causa. **4.2.2** Se retorna al paso **2**.
Flujo de excepciones	➔ NullPointerException: Se detiene el flujo y se detalla cuál es el objeto sin instanciar en el sistema mediante un mensaje de error. Se retorna el flujo al paso **2**.
Asociaciones de inclusión	➔ CU23, CU27
Asociaciones de extensión	➔ CU04
Requerimientos especiales	➔

Nombre	Eliminar adherente	**ID**	CU07
Descripción	El sistema permitirá eliminar un adherente..		
Actores	➔ Usuario final		
Prioridad	◉ Alta ◎ Media ◎ Baja	**Complejidad**	◎ Alta ◉ Media ◎ Baja
Precondiciones	➔ Adherente existente con los datos especificados.		
Postcondiciones	Éxito: ➔ Eliminación lógica del adherente..		Fracaso: ➔ Visualización de un mensaje de error que describe la causa del fallo.

Flujo Principal	Flujo Alternativo
1. El usuario selecciona la opción “Eliminar” en el listado de adherentes.	
2. Se solicita la inserción de los siguientes campos: ● Fecha: por defecto fecha actual. ● Motivo: ○ Fallecimiento ○ Morosidad ○ Renuncia	
3. El usuario presiona el botón “Eliminar”.	**3.1.** El usuario desea cancelar la operación. **3.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
4. El sistema presenta un mensaje de confirmación para eliminar el adherente indicado.	
5. El usuario escoge la opción “Aceptar”.	**5.1.** El usuario desea cancelar la operación. **5.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
6. El sistema presenta un mensaje de eliminación exitosa.	**6.1.** El usuario desea cancelar la operación. **6.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.

Flujo de excepciones	➔ NullPointerException: Se detiene el flujo y se detalla cuál es el objeto sin instanciar en el sistema mediante un mensaje de error. Se finaliza el flujo.
Asociaciones de inclusión	➔
Asociaciones de extensión	➔

Requerimientos especiales	➔
Observaciones	

Nombre	Modificar adherente	**ID**	CU08
Descripción	El sistema permitirá editar la información de un adherente.		
Actores	➔ Usuario final		
Prioridad	◉ Alta ◎ Media ◎ Baja	**Complejidad**	◎ Alta ◉ Media ◎ Baja
Precondiciones	➔ Adherente existente con los datos especificados.		
Postcondiciones	Éxito: ➔ Adherente con datos actualizados en base de datos.	Fracaso: ➔ Visualización de un mensaje de error que describe la causa del fallo.	

Flujo Principal	**Flujo Alternativo**
1. El usuario selecciona la opción “Modificar” de un adherente del listado.	
2. Se presenta una pantalla con la información actual del adherente.	
3. Se completa la totalidad de estos campos: ● DNI ***** ● Nombres ***** ● Apellidos ***** ● Fecha de nacimiento ***** ● Número de teléfono ***** ● Correo electrónico ● Obra social ***** ● Sexo ***** ● Zona ***** ● Localidad ***** ● Domicilio *****	**3.1.** El usuario desea cancelar la operación. **3.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
4. El usuario presiona el botón “Aceptar”.	**4.1.** El usuario desea cancelar la operación. **4.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
5. El sistema valida que todos los campos obligatorios (*****) han sido ingresados y que el valor de cada uno de los campos no vacíos (requeridos o no) sea correcto.	**5.1.** Ya existe un socio con el DNI ingresado. **5.1.1** Se visualiza un mensaje de error describiendo explícitamente que ese socio ya existe en la base de datos. **5.1.2** Se retorna al paso **2**. **5.2.** Un campo no corresponde al tipo y formato esperado.

	5.2.1 Se visualiza un mensaje de error describiendo explícitamente cuál es el campo involucrado en el error y cuál es la causa. **5.1.2** Se retorna al paso **2**.
Flujo de excepciones	➔ NullPointerException: Se detiene el flujo y se detalla cuál es el objeto sin instanciar en el sistema mediante un mensaje de error. Se retorna el flujo al paso **2**.
Asociaciones de inclusión	➔ CU23, CU27
Asociaciones de extensión	➔
Requerimientos especiales	➔
Observaciones	

<table>
<tr><td>Nombre</td><td colspan="3">Consultar adherentes</td><td>ID</td><td>CU09</td></tr>
<tr><td>Descripción</td><td colspan="5">El sistema permitirá buscar y listar adherentes.</td></tr>
<tr><td>Actores</td><td colspan="5">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◉ Alta ◎ Media ◎ Baja</td><td>Complejidad</td><td colspan="2">◎ Alta ◉ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="5">➔</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="3">Fracaso</td></tr>
<tr><td colspan="2">➔ Visualización de una lista representativa de los adherentes.</td><td colspan="3">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="3">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario selecciona la opción “Listar adherentes”.</td><td colspan="3"></td></tr>
<tr><td colspan="3">2. El sistema presenta una tabla de todos los adherentes con la siguiente información:
● DNI
● Apellido
● Nombre
● Socio DNI
● Cobertura
● Estado</td><td colspan="3">2.1. No existen adherentes.
2.1.1 Se muestra un mensaje en el dashboard “No existen adherentes.”. Finaliza el CU.</td></tr>
<tr><td colspan="3">3. El sistema permite la opción de filtrar por DNI, Nombre y Apellido</td><td colspan="3"></td></tr>
<tr><td colspan="3">4. El usuario selecciona una acción sobre un adherente:
a. Información (#REF CU10)
b. Dar de baja (#REF CU07)
c. Modificar (#REF CU08)</td><td colspan="3"></td></tr>
<tr><td>Flujo de excepciones</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="5">➔ CU07, CU08, CU10</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="5">➔</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="5">➔</td></tr>
<tr><td>Observaciones</td><td colspan="5"></td></tr>
</table>

Nombre	Consultar adherente	**ID**	CU10
Descripción	El sistema permitirá brindar información específica de un adherente.		
Actores	➔ Usuario final		
Prioridad	◉ Alta ◎ Media ◎ Baja	**Complejidad**	◎ Alta ◉ Media ◎ Baja
Precondiciones	➔		
Postcondiciones	Éxito: ➔ Visualización de una ficha con toda la información de un adherente.	Fracaso: ➔	

Flujo Principal	**Flujo Alternativo**
1. El sistema presenta una ficha con toda la información del adherente seleccionado.	
2. Se visualiza la siguiente información del socio: ● ID adherente ● ID socio ● DNI ● Nombres ● Apellidos ● Fecha de nacimiento ● Fecha de cobertura ● Número de teléfono ● Correo electrónico ● Obra social ● Sexo ● Zona ● Localidad ● Domicilio	

Flujo de excepciones	➔
Asociaciones de inclusión	➔
Asociaciones de extensión	➔
Requerimientos especiales	➔
Observaciones	➔

8.2.3 Requerimientos Funcional 3

El sistema permitirá consultar planes.

Nombre	Consultar planes	**ID**	CU11
Descripción	El sistema permitirá listar planes.		
Actores	➔ Usuario final		
Prioridad	◉ Alta ◎ Media ◎ Baja	**Complejidad**	◎ Alta ◉ Media ◎ Baja
Precondiciones	➔		
Postcondiciones	Éxito: ➔ Visualización de una lista representativa de los planes.		Fracaso: ➔ Visualización de un mensaje de error que describe la causa del fallo.

Flujo Principal	Flujo Alternativo
1. El sistema solicita la información de los planes	
2. El sistema presenta una lista de planes identificada por: ● Nombre	**2.1.** No existen planes. **2.1.1** Se muestra un mensaje en el dashboard “No existen planes.”. Finaliza el CU.

Flujo de excepciones	➔
Asociaciones de inclusión	➔
Asociaciones de extensión	➔
Requerimientos especiales	➔
Requerimientos especiales	➔
Observaciones	

8.2.4 Requerimientos Funcional 4

El sistema permitirá agregar, eliminar y consultar tarifas.

<table>
<tr><td>Nombre</td><td colspan="3">Agregar tarifa</td><td>ID</td><td>CU12</td></tr>
<tr><td>Descripción</td><td colspan="5">El sistema permitirá agregar una tarifa.</td></tr>
<tr><td>Actores</td><td colspan="5">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◉ Alta ◎ Media ◎ Baja</td><td>Complejidad</td><td colspan="2">◎ Alta ◉ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="5"></td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="3">Fracaso</td></tr>
<tr><td colspan="2">➔ Tarifa almacenada en base de datos.</td><td colspan="3">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="3">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario escoge la opción “Alta” en la lista desplegable de tarifa.</td><td colspan="3"></td></tr>
<tr><td colspan="3">2. Se completa la totalidad de estos campos:
● N° tarifa *
● Descripción *
● Plan *
● Valor *
● Rangos *</td><td colspan="3">2.1. El usuario desea cancelar la operación.
2.1.1 Se presiona el botón “Cancelar” o se regresa a la página anterior.</td></tr>
<tr><td colspan="3">3. El usuario presiona el botón “Aceptar”</td><td colspan="3">3.1. El usuario desea cancelar la operación.
3.1.1 Se presiona el botón “Cancelar” o se regresa a la página anterior.</td></tr>
<tr><td colspan="3">4. El sistema valida que todos los campos obligatorios (*) han sido ingresados y que el valor de cada uno de los no vacíos (requeridos o no) sea correcto.</td><td colspan="3">4.1. Un campo no corresponde al tipo y formato esperado.
4.1.1 Se visualiza un mensaje de error describiendo explícitamente cuál es el campo involucrado en el error y cuál es la causa.
4.1.2 Se retorna al paso 2.</td></tr>
<tr><td>Flujo de excepciones</td><td colspan="5">➔ NullPointerException: Se detiene el flujo y se detalla cuál es el objeto sin instanciar en el sistema mediante un mensaje de error. Se retorna el flujo al paso 2.</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="5">➔ CU11</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="5">➔</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="5">➔</td></tr>
</table>

Observaciones	

Nombre	Eliminar tarifa			**ID**	CU13
Descripción	El sistema permitirá eliminar una tarifa.				
Actores	➔ Usuario final				
Prioridad	◉ Alta ◎ Media ◎ Baja		**Complejidad**	◎ Alta ◉ Media ◎ Baja	
Precondiciones	➔ Tarifa existente con los datos especificados.				
Postcondiciones	Éxito		Fracaso		
	➔ Eliminación lógica de la tarifa.		➔ Visualización de un mensaje de error que describe la causa del fallo.		

Flujo Principal	**Flujo Alternativo**
1. El usuario selecciona la opción “Eliminar” en el padrón de tarifas.	
2. El sistema presenta un mensaje de confirmación para eliminar la tarifa indicada.	
3. El usuario escoge la opción “Aceptar”.	**3.1.** El usuario desea cancelar la operación. **3.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
4. El estado de la tarifa es modificado en la base de datos indicando que ya no existe. El sistema presenta un mensaje de eliminación exitosa.	

Flujo de excepciones	➔ NullPointerException: Se detiene el flujo y se detalla cuál es el objeto sin instanciar en el sistema mediante un mensaje de error. Se finaliza el flujo.
Asociaciones de inclusión	➔
Asociaciones de extensión	➔
Requerimientos especiales	➔
Observaciones	

<table>
<tr><td>Nombre</td><td colspan="3">Consultar tarifas</td><td>ID</td><td>CU14</td></tr>
<tr><td>Descripción</td><td colspan="5">El sistema permitirá buscar y listar tarifas.</td></tr>
<tr><td>Actores</td><td colspan="5">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◉ Alta ◎ Media ◎ Baja</td><td>Complejidad</td><td colspan="2">◎ Alta ◉ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="5">➔</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="3">Fracaso</td></tr>
<tr><td colspan="2">➔ Visualización de una lista representativa de las tarifas.</td><td colspan="3">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="3">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario selecciona la opción “Padrón” en el menú de Tarifas.</td><td colspan="3"></td></tr>
<tr><td colspan="3">2. El sistema presenta una tabla de todos los adherentes con la siguiente información:
● N° Tarifa
● Descripción
● Valor
● Plan</td><td colspan="3">2.1. No existen tarifas.
2.1.1 Se muestra un mensaje en el dashboard “No existen tarifas.”. Finaliza el CU.</td></tr>
<tr><td colspan="3">3. El sistema permite la opción de filtrar por N°, descripción y plan.</td><td colspan="3"></td></tr>
<tr><td colspan="3">4. El usuario selecciona una acción sobre una tarifa:
d. Ver rangos
e. Dar de baja (#REF CU13)</td><td colspan="3"></td></tr>
<tr><td>Flujo de excepciones</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="5">➔ CU13</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="5">➔</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="5">➔</td></tr>
<tr><td>Observaciones</td><td colspan="5"></td></tr>
</table>

8.2.5 Requerimientos Funcional 5

Para el cálculo de los valores del comprobante de cada asociado, el sistema permitirá calcular una tarifa bajo ciertas condiciones y generar los comprobantes de pago.

Nombre	Generar comprobantes de pago		**ID**	CU15
Descripción	Para el cálculo de los valores del comprobante de cada asociado, el sistema permitirá calcular una tarifa bajo ciertas condiciones.			
Actores	➔ Usuario final			
Prioridad	◎ Alta ⊙ Media ◎ Baja	**Complejidad**	◎ Alta ⊙ Media ◎ Baja	
Precondiciones	➔			
Postcondiciones	Éxito	Fracaso		
	➔ PDF listo para imprimir y entregar como comprobante de pago a los socios.	➔ Se frena el flujo de impresión y se solicita un reintento.		

Flujo Principal	**Flujo Alternativo**
1. El usuario selecciona la opción “Consultar” en el menú de comprobantes	
2. El usuario selecciona el botón de “Generar”.	**2.1.** No existen socios. **2.1.1** Se muestra un mensaje en el dashboard “No existen socios.”. Finaliza el CU.
3. El sistema realiza el cálculo del comprobante de cada socio: **3.1** El sistema busca la tarifa del socio. **3.2** Si no tiene adherentes, se entrega el valor de la tarifa. Si tiene adherentes: **3.2.1** Se busca los rangos de la tarifa del socio y se itera por cada adherente recuperando la edad del mismo y sumando el rango en el que se encuentre el mismo.	**3.1.** Si se presenta un error de lógica, se muestra un mensaje en el dashboard “No fue posible generar los comprobantes. Reintente.”. Finaliza el CU.
4. El sistema genera un pdf con todos los comprobantes.	

Flujo de excepciones	➔
Asociaciones de inclusión	➔ CU04, CU09
Asociaciones de extensión	➔
Requerimientos especiales	➔
Observaciones	

8.2.6 Requerimientos Funcional 6

El sistema permitirá agregar, actualizar, eliminar y consultar cobradores.

<table>
<tr><td>Nombre</td><td colspan="2">Agregar cobrador</td><td>ID</td><td>CU16</td></tr>
<tr><td>Descripción</td><td colspan="4">El sistema permitirá agregar un cobrador.</td></tr>
<tr><td>Actores</td><td colspan="4">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◉ Alta ◎ Media ◎ Baja</td><td>Complejidad</td><td>◎ Alta ◉ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="4">➔ No existe cobrador que represente a la persona que se intenta agregar.</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="2">Fracaso</td></tr>
<tr><td colspan="2">➔ Cobrador almacenado en base de datos con los datos especificados.</td><td colspan="2">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><th colspan="3">Flujo Principal</th><th colspan="2">Flujo Alternativo</th></tr>
<tr><td colspan="3">1. El usuario escoge la opción “Alta” en la lista desplegable de cobradores.</td><td colspan="2"></td></tr>
<tr><td colspan="3">2. Se completa la totalidad de estos campos:
● DNI *
● Nombres *
● Apellidos *
● Fecha de nacimiento *
● Número de celular *
● Correo electrónico *
● Sexo *
● Localidad *
● Domicilio *</td><td colspan="2">2.1. El usuario desea cancelar la operación.
2.1.1 Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.</td></tr>
<tr><td colspan="3">3. El usuario presiona el botón “Aceptar”</td><td colspan="2">3.1. El usuario desea cancelar la operación.
3.1.1 Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.</td></tr>
<tr><td colspan="3">4. El sistema valida que todos los campos obligatorios (*) han sido ingresados y que el valor de cada uno de los no vacíos (requeridos o no) sea correcto.</td><td colspan="2">4.1. Ya existe un cobrador con el DNI ingresado.
4.1.1. Se visualiza un mensaje de error describiendo explícitamente que ese cobrador ya existe en la base de datos.
4.1.2. Se retorna al paso 2.
4.2. Un campo no corresponde al tipo y formato esperado.</td></tr>
</table>

	4.2.1. Se visualiza un mensaje de error describiendo explícitamente cuál es el campo involucrado en el error y cuál es la causa. **4.2.2.** Se retorna al paso **2**.
Flujo de excepciones	➔ NullPointerException: Se detiene el flujo y se detalla cuál es el objeto sin instanciar en el sistema mediante un mensaje de error. Se retorna el flujo al paso **2**.
Asociaciones de inclusión	➔
Asociaciones de extensión	➔
Requerimientos especiales	➔
Observaciones	

<table>
<tr><td>Nombre</td><td colspan="2">Eliminar cobrador</td><td>ID</td><td>CU17</td></tr>
<tr><td>Descripción</td><td colspan="4">El sistema permitirá eliminar un cobrador.</td></tr>
<tr><td>Actores</td><td colspan="4">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◉ Alta ◎ Media ◎ Baja</td><td>Complejidad</td><td>◎ Alta ◉ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="4">➔ Cobrador existente con los datos especificados.</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="2">Fracaso</td></tr>
<tr><td colspan="2">➔ Eliminación lógica del cobrador.</td><td colspan="2">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="2">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario selecciona la opción “Eliminar” en el listado de cobradores.</td><td colspan="2"></td></tr>
<tr><td colspan="3">2. El sistema presenta un mensaje de confirmación para eliminar al cobrador indicado.</td><td colspan="2"></td></tr>
<tr><td colspan="3">3. El usuario escoge la opción “Aceptar”.</td><td colspan="2">3.1. El usuario desea cancelar la operación.
3.1.1 Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.</td></tr>
<tr><td colspan="3">4. El sistema modifica el cobrador en la base de datos para que a partir de ese momento deje de existir y presenta un mensaje de eliminación exitosa.</td><td colspan="2"></td></tr>
<tr><td>Flujo de excepciones</td><td colspan="4">➔ NullPointerException: Se detiene el flujo y se detalla cuál es el objeto sin instanciar en el sistema mediante un mensaje de error. Se finaliza el flujo.</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="4">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="4">➔</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="4">➔</td></tr>
<tr><td>Observaciones</td><td colspan="4"></td></tr>
</table>

<table>
<tr><td>Nombre</td><td colspan="3">Modificar cobrador</td><td>ID</td><td>CU18</td></tr>
<tr><td>Descripción</td><td colspan="5">El sistema permitirá editar la información de un cobrador.</td></tr>
<tr><td>Actores</td><td colspan="5">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◉ Alta ◎ Media ◎ Baja</td><td>Complejidad</td><td colspan="2">◎ Alta ◉ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="5">➔ Cobrador existente con los datos especificados.</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="3">Fracaso</td></tr>
<tr><td colspan="2">➔ Cobrador con datos actualizados en base de datos.</td><td colspan="3">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="3">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario selecciona la opción “Modificar” de un cobrador del listado.</td><td colspan="3"></td></tr>
<tr><td colspan="3">2. Se presenta una pantalla con la información actual del cobrador.</td><td colspan="3"></td></tr>
<tr><td colspan="3">3. Se completan los siguientes campos:
● DNI *
● Nombres *
● Apellidos *
● Fecha de nacimiento *
● Número de celular *
● Correo electrónico *
● Sexo *
● Localidad *
● Domicilio *</td><td colspan="3">3.1. El usuario desea cancelar la operación.
3.1.1 Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.</td></tr>
<tr><td colspan="3">4. El usuario presiona el botón “Aceptar”.</td><td colspan="3">4.1. El usuario desea cancelar la operación.
4.1.1 Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.</td></tr>
<tr><td colspan="3">5. El sistema valida que todos los campos obligatorios (*) han sido ingresados y que el valor de cada uno de los campos no vacíos (requeridos o no) sea correcto.</td><td colspan="3">5.1. Ya existe un cobrador con el DNI ingresado.
5.1.1 Se visualiza un mensaje de error describiendo explícitamente que ese cobrador ya existe en la base de datos.
5.1.2 Se retorna al paso 2.
5.2. Un campo no corresponde al tipo y formato esperado.
5.2.1 Se visualiza un mensaje de error describiendo explícitamente cuál</td></tr>
</table>

	es el campo involucrado en el error y cuál es la causa. **5.1.2** Se retorna al paso **2**.
Flujo de excepciones	➔ NullPointerException: Se detiene el flujo y se detalla cuál es el objeto sin instanciar en el sistema mediante un mensaje de error. Se retorna el flujo al paso **2**.
Asociaciones de inclusión	➔
Asociaciones de extensión	➔
Requerimientos especiales	➔
Observaciones	

<table>
<tr><td>Nombre</td><td colspan="2">Consultar cobradores</td><td>ID</td><td>CU19</td></tr>
<tr><td>Descripción</td><td colspan="4">El sistema permitirá buscar y listar cobradores.</td></tr>
<tr><td>Actores</td><td colspan="4">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◎ Alta ◉ Media ◎ Baja</td><td>Complejidad</td><td>◎ Alta ◉ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="4">➔</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="2">Fracaso</td></tr>
<tr><td colspan="2">➔ Visualización de una lista de cobradores.</td><td colspan="2">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="2">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario selecciona la opción “Padrón” en el menú de Cobradores.</td><td colspan="2"></td></tr>
<tr><td colspan="3">2. El sistema presenta una tabla de todos los cobradores con la siguiente información:
● ID cobrador
● DNI
● Apellido
● Nombre
● Teléfono</td><td colspan="2">2.1. No existen cobradores.
2.1.1 Se muestra el mensaje“No existen cobradores.”. Finaliza el CU.</td></tr>
<tr><td colspan="3">3. El usuario selecciona una acción sobre un cobrador:
a. Modificar (#REF CU18)
b. Dar de baja (#REF CU17)
c. Zonas asignadas (#REF CU23)</td><td colspan="2">3.1. El usuario presiona el botón “Editar” de un determinado plan de la grilla.
3.1.1 Se invoca al CU14.
3.2. El usuario presiona el botón “Eliminar” de un determinado plan de la grilla.
3.2.1 Se invoca al CU13.</td></tr>
<tr><td>Flujo de excepciones</td><td colspan="4">➔</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="4">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="4">➔ CU17, CU18, CU20</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="4">➔</td></tr>
<tr><td>Observaciones</td><td colspan="4"></td></tr>
</table>

<table>
<tr><td>Nombre</td><td colspan="2">Consultar cobrador</td><td>ID</td><td>CU20</td></tr>
<tr><td>Descripción</td><td colspan="4">El sistema permitirá visualizar el detalle de un cobrador</td></tr>
<tr><td>Actores</td><td colspan="4">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◎ Alta ◉ Media ◎ Baja</td><td>Complejidad</td><td>◎ Alta ◉ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="4">➔</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="2">Fracaso</td></tr>
<tr><td colspan="2">➔ Visualización de los detalles correspondientes a un cobrador.</td><td colspan="2">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="2">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario selecciona un cobrador del listado.</td><td colspan="2"></td></tr>
<tr><td colspan="3">2. El sistema visualiza la siguiente información:
● ID cobrador
● DNI
● Nombres
● Apellidos
● Fecha de nacimiento
● Número de celular
● Correo electrónico
● Sexo
● Localidad
● Domicilio</td><td colspan="2"></td></tr>
<tr><td>Flujo de excepciones</td><td colspan="4">➔</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="4">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="4">➔</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="4">➔</td></tr>
<tr><td>Observaciones</td><td colspan="4"></td></tr>
</table>

8.2.7 Requerimientos Funcional 7

El sistema permitirá agregar, actualizar, eliminar y consultar zonas. A cada zona, se le asignará uno o más cobradores.

<table>
<tr><td>Nombre</td><td colspan="3">Agregar zona</td><td>ID</td><td>CU21</td></tr>
<tr><td>Descripción</td><td colspan="5">El sistema permitirá agregar una zona.</td></tr>
<tr><td>Actores</td><td colspan="5">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◉ Alta ◎ Media ◎ Baja</td><td>Complejidad</td><td colspan="2">◎ Alta ◉ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="5"></td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="3">Éxito</td><td colspan="2">Fracaso</td></tr>
<tr><td colspan="3">➔ Zona almacenada en base de datos con los datos especificados.</td><td colspan="2">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="3">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario escoge la opción “Alta” en la lista desplegable de zonas.</td><td colspan="3"></td></tr>
<tr><td colspan="3">2. Se completa la totalidad de estos campos:
● N° Zona *
● Nombre *
● Cobradores *</td><td colspan="3"></td></tr>
<tr><td colspan="3">3. El usuario presiona el botón “Aceptar”</td><td colspan="3">3.1. El usuario desea cancelar la operación.
3.1.1 Se presiona el botón “Cancelar” o se regresa a la página anterior.</td></tr>
<tr><td colspan="3">4. El sistema valida que todos los campos obligatorios (*) han sido ingresados y que el valor de cada uno de los no vacíos (requeridos o no) sea correcto.</td><td colspan="3">4.1. Un campo no corresponde al tipo y formato esperado.
4.1.1 Se visualiza un mensaje de error describiendo explícitamente cuál es el campo involucrado en el error y cuál es la causa.
4.1.2 Se retorna al paso 2.</td></tr>
<tr><td>Flujo de excepciones</td><td colspan="5">➔ NullPointerException: Se detiene el flujo y se detalla cuál es el objeto sin instanciar en el sistema mediante un mensaje de error. Se retorna el flujo al paso 2.</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="5">➔</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="5">➔</td></tr>
</table>

Observaciones	

Nombre	Eliminar zona			**ID**	CU22
Descripción	El sistema permitirá eliminar una zona.				
Actores	➔ Usuario final				
Prioridad	◉ Alta ◎ Media ◎ Baja		**Complejidad**	◎ Alta ◉ Media ◎ Baja	
Precondiciones	➔ Zona existente con los datos especificados.				
Postcondiciones	Éxito		Fracaso		
	➔ Eliminación lógica de la zona.		➔ Visualización de un mensaje de error que describe la causa del fallo.		

Flujo Principal	**Flujo Alternativo**
1. El usuario selecciona la opción “dar de baja” en el listado de zonas.	
2. El sistema presenta un mensaje de confirmación para eliminar la zona indicada.	
3. El usuario escoge la opción “Aceptar”.	**3.1.** El usuario desea cancelar la operación. **3.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
4. El sistema modifica la zona en la base de datos para que, a partir de ese momento, deje de existir y presenta un mensaje de eliminación exitosa.	

Flujo de excepciones	➔ NullPointerException: Se detiene el flujo y se detalla cuál es el objeto sin instanciar en el sistema mediante un mensaje de error. Se finaliza el flujo.
Asociaciones de inclusión	➔
Asociaciones de extensión	➔
Requerimientos especiales	➔
Observaciones	

<table>
<tr><td>Nombre</td><td colspan="2">Consultar zonas</td><td>ID</td><td>CU23</td></tr>
<tr><td>Descripción</td><td colspan="4">El sistema permitirá buscar y listar zonas.</td></tr>
<tr><td>Actores</td><td colspan="4">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◎ Alta ◉ Media ◎ Baja</td><td>Complejidad</td><td>◎ Alta ◉ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="4">➔</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="2">Fracaso</td></tr>
<tr><td colspan="2">➔ Visualización de una lista de las zonas.</td><td colspan="2">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="2">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario selecciona la opción “Padrón” del menú de Zonas.</td><td colspan="2"></td></tr>
<tr><td colspan="3">2. El sistema presenta una tabla de todas las zonas con la siguiente información:
● N° Zona
● Nombre
● Cobradores</td><td colspan="2">2.1. No existen zonas.
2.1.1 Se muestra el siguiente mensaje: “No existen zonas”. Finaliza el CU.</td></tr>
<tr><td>Flujo de excepciones</td><td colspan="4">➔</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="4">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="4">➔ CU22</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="4">➔</td></tr>
<tr><td>Observaciones</td><td colspan="4"></td></tr>
</table>

8.2.8 Requerimientos Funcional 10

Cuando se genere el alta de un socio, existirá una función para calcular el tiempo de comienzo de cobertura.

<table>
<tr><td>Nombre</td><td colspan="2">Calcular cobertura</td><td>ID</td><td>CU24</td></tr>
<tr><td>Descripción</td><td colspan="4">Cuando se genere el alta de un socio, existirá una función para calcular el tiempo de comienzo de cobertura.</td></tr>
<tr><td>Actores</td><td colspan="4">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td>◎ Alta ⊙ Media ◎ Baja</td><td>Complejidad</td><td colspan="2">⊙ Alta ◎ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="4">➔ Formulario de alta de socio/adherente completo</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td>Éxito</td><td colspan="3">Fracaso</td></tr>
<tr><td>➔ Fecha estimada de cobertura presentada por el algoritmo.</td><td colspan="3">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="2">Flujo Principal</td><td colspan="3">Flujo Alternativo</td></tr>
<tr><td colspan="2">1. El sistema toma como entrada la fecha de nacimiento y dispara un algoritmo de pronóstico.</td><td colspan="3"></td></tr>
<tr><td colspan="2">2. El sistema entrega una fecha propuesta que es información de salida del algoritmo.</td><td colspan="3">2.1. La fecha de nacimiento es mayor a la fecha de hoy.
2.1.1 Se presenta un mensaje de error y el algoritmo no se ejecuta. Finaliza el CU.
2.2. Hubo un error de búsqueda o procesamiento de información en el algoritmo. Finaliza el CU.</td></tr>
<tr><td>Flujo de excepciones</td><td colspan="4">➔</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="4">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="4">➔</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="4">➔</td></tr>
<tr><td>Observaciones</td><td colspan="4"></td></tr>
</table>

8.2.9 Requerimientos Funcional 13

El sistema permitirá agregar, actualizar, eliminar y consultar obras sociales.

<table>
<tr><td>Nombre</td><td colspan="3">Agregar obra social</td><td>ID</td><td>CU25</td></tr>
<tr><td>Descripción</td><td colspan="5">El sistema permitirá agregar una obra social (OS).</td></tr>
<tr><td>Actores</td><td colspan="5">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◎ Alta ◎ Media ◉ Baja</td><td>Complejidad</td><td colspan="2">◎ Alta ◎ Media ◉ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="5"></td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="3">Fracaso</td></tr>
<tr><td colspan="2">➔ OS almacenada en base de datos con los datos especificados.</td><td colspan="3">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="3">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario escoge la opción “Alta” en la lista desplegable de obras sociales.</td><td colspan="3"></td></tr>
<tr><td colspan="3">2. Se completa la totalidad de estos campos:
● Descripción *
● Sepelio * (si cubre los gastos de fallecimiento)</td><td colspan="3"></td></tr>
<tr><td colspan="3">3. El usuario presiona el botón “Aceptar”</td><td colspan="3">3.1. El usuario desea cancelar la operación.
3.1.1 Se presiona el botón “Cancelar” o se regresa a la página anterior.</td></tr>
<tr><td colspan="3">4. El sistema valida que todos los campos obligatorios (*) han sido ingresados y que el valor de cada uno de los no vacíos (requeridos o no) sea correcto.</td><td colspan="3">4.1. Un campo no corresponde al tipo y formato esperado.
4.1.1 Se visualiza un mensaje de error describiendo explícitamente cuál es el campo involucrado en el error y cuál es la causa.
4.1.2 Se retorna al paso 2.</td></tr>
<tr><td>Flujo de excepciones</td><td colspan="5">➔ NullPointerException: Se detiene el flujo y se detalla cuál es el objeto sin instanciar en el sistema mediante un mensaje de error. Se retorna el flujo al paso 2.</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="5">➔</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="5">➔</td></tr>
<tr><td>Observaciones</td><td colspan="5"></td></tr>
</table>

Nombre	Eliminar obra social	**ID**	CU26
Descripción	El sistema permitirá eliminar una obra social.		
Actores	➔ Usuario final		
Prioridad	◉ Alta ◎ Media ◎ Baja	**Complejidad**	◎ Alta ◉ Media ◎ Baja
Precondiciones	➔ Zona existente con los datos especificados.		
Postcondiciones	Éxito: ➔ Eliminación lógica de la zona.	Fracaso: ➔ Visualización de un mensaje de error que describe la causa del fallo.	

Flujo Principal	**Flujo Alternativo**
1. El usuario selecciona la opción “dar de baja” en el listado de obras sociales.	
2. El sistema presenta un mensaje de confirmación para eliminar la obra social indicada.	
3. El usuario escoge la opción “Aceptar”.	**3.1.** El usuario desea cancelar la operación. **3.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
4. El sistema modifica la obra social en la base de datos para que, a partir de ese momento, deje de existir y presenta un mensaje de eliminación exitosa.	

Flujo de excepciones	➔ NullPointerException: Se detiene el flujo y se detalla cuál es el objeto sin instanciar en el sistema mediante un mensaje de error. Se finaliza el flujo.
Asociaciones de inclusión	➔
Asociaciones de extensión	➔
Requerimientos especiales	➔
Observaciones	

<table>
<tr><td>Nombre</td><td colspan="3">Consultar obras sociales</td><td>ID</td><td>CU27</td></tr>
<tr><td>Descripción</td><td colspan="5">El sistema permitirá buscar y listar obras sociales.</td></tr>
<tr><td>Actores</td><td colspan="5">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◎ Alta ⊙ Media ◎ Baja</td><td>Complejidad</td><td colspan="2">◎ Alta ⊙ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="5">➔</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="3">Éxito</td><td colspan="2">Fracaso</td></tr>
<tr><td colspan="3">➔ Visualización de una lista representativa de las obras sociales que cumplan con los filtros deseados.</td><td colspan="2">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="3">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario selecciona la opción “Padrón” del menú de Obras sociales.</td><td colspan="3"></td></tr>
<tr><td colspan="3">2. El sistema presenta una tabla de todas las OS con la siguiente información:
● N° OS
● Descripción
● Sepelio</td><td colspan="3">2.1. No existen OS.
2.1.1 Se muestra el mensaje “No existen obras sociales”. Finaliza el CU.</td></tr>
<tr><td colspan="3">3. El sistema presenta la opción de dar de baja una OS (#REF CU26)</td><td colspan="3"></td></tr>
<tr><td>Flujo de excepciones</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="5">➔ CU26</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="5">➔</td></tr>
<tr><td>Observaciones</td><td colspan="5"></td></tr>
</table>

8.2.10 Requerimientos Funcional 14

El sistema permitirá llevar registro del saldo y comprobantes, permitiendo la visualización para cada uno de los clientes.

<table>
<tr><td>Nombre</td><td colspan="2">Consultar saldo y comprobantes</td><td>ID</td><td>CU28</td></tr>
<tr><td>Descripción</td><td colspan="4">El sistema permitirá llevar seguimiento de movimientos de comprobantes de un socio y su saldo actual</td></tr>
<tr><td>Actores</td><td colspan="4">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◎ Alta ⊙ Media ◎ Baja</td><td>Complejidad</td><td>◎ Alta ⊙ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="4">➔</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="2">Fracaso</td></tr>
<tr><td colspan="2">➔ Se muestran los comprobantes pagados y no pagados de un socio junto a su saldo.</td><td colspan="2">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="2">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario ingresá a “Informar” en el menú de Pagos.</td><td colspan="2"></td></tr>
<tr><td colspan="3">2. El usuario selecciona el socio a consultar.</td><td colspan="2"></td></tr>
<tr><td colspan="3">3. El sistema muestra:
● DNI
● Plan
● Tarifa
● Estado
● Saldo</td><td colspan="2"></td></tr>
<tr><td colspan="3">4. El usuario puede consultar el historial de comprobantes.
4.1 Si lo consulta, se muestra el listado de comprobantes asociado al socio con el estado actual de los mismos.
4.2 El usuario puede informar el pago de un comprobante dado (#REF CU29)</td><td colspan="2"></td></tr>
<tr><td>Flujo de excepciones</td><td colspan="4">➔</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="4">➔</td></tr>
</table>

Asociaciones de extensión	➔ CU29
Requerimientos especiales	➔
Observaciones	

8.2.11 Requerimientos Funcional 16

El sistema permitirá informar un pago modificando el estado del comprobante pagado y el saldo del socio.

Nombre	Informar pago	**ID**	CU29
Descripción	El sistema permitirá registrar el pago de un comprobante dado asociado a un socio.		
Actores	➔ Usuario final		
Prioridad	◎ Alta ⊙ Media ◎ Baja	**Complejidad**	◎ Alta ⊙ Media ◎ Baja
Precondiciones	➔ Comprobante no pagado.		
Postcondiciones	Éxito: ➔ Se informa el pago y decrementa el saldo de un socio.	Fracaso: ➔ Visualización de un mensaje de error que describe la causa del fallo.	

Flujo Principal	**Flujo Alternativo**
1. El usuario selecciona la opción de “Informar pago” en un comprobante de la tabla de comprobantes de un socio.	
2. El usuario selecciona el cobrador que realizó el cobro.	**2.1.** El usuario desea cancelar la operación. **2.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
3. El sistema muestra un mensaje de éxito y se decrementa el saldo del socio.	**3.1.** No fue posible informar el pago por error del sistema. Se muestra un mensaje de error descriptivo. Finaliza el CU.

Flujo de excepciones	➔
Asociaciones de inclusión	➔
Asociaciones de extensión	➔
Requerimientos especiales	➔
Observaciones	

8.2.12 Requerimientos Funcional 17

El sistema generará un backup automáticamente todas las semanas en una hora específica.

Nombre	Crear copia de seguridad	**ID**	CU30
Descripción	El sistema generará un backup automáticamente todas las semanas en una hora específica de la base de datos.		
Actores	➔ Usuario final		
Prioridad	◎ Alta ⊙ Media ◎ Baja	**Complejidad**	◎ Alta ⊙ Media ◎ Baja
Precondiciones	➔ Gestor de backup de BD instalado		
Postcondiciones	Éxito: ➔ Respaldo almacenado en disco externo.	Fracaso: ➔ Visualización de mensaje de error con la descripción de la causa del fallo.	

Flujo Principal	Flujo Alternativo
1. El usuario habilita la configuración de respaldo automático.	
2. Se solicita el horario deseado para que se lleve a cabo el respaldo.	
3. La base de datos del servidor es duplicada en un disco externo en un archivo cuyo nombre incluya la fecha.	**3.1.** Ocurre un error en la ejecución del respaldo. **3.1.1.** Se escribe en un archivo de log indicando el detalle del fallo. **3.1.2.** Se almacena en la base de datos un flag indicando que no pudo realizarse el back up para que en la próxima ejecución del programa se visualice un mensaje advirtiendo la situación.

Flujo de excepciones	➔
Asociaciones de inclusión	➔
Asociaciones de extensión	➔
Requerimientos especiales	➔
Observaciones	

8.2.13 Requerimientos Funcional 21

El sistema permitirá generar el reporte de estado y cobertura de socios y adherentes.

<table>
<tr><td>Nombre</td><td colspan="3">Reporte de estado de socios</td><td>ID</td><td>CU31</td></tr>
<tr><td>Descripción</td><td colspan="5">El sistema permitirá generar el reporte de estado y cobertura de socios y adherentes.</td></tr>
<tr><td>Actores</td><td colspan="5">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◎ Alta ⊙ Media ◎ Baja</td><td>Complejidad</td><td colspan="2">◎ Alta ⊙ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="5">➔</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="3">Fracaso</td></tr>
<tr><td colspan="2">➔ Reporte generado correctamente y visualizado en pantalla.</td><td colspan="3">➔ Visualización de mensaje de error con la descripción de la causa del fallo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="3">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario ingresa a la sección de reportes</td><td colspan="3"></td></tr>
<tr><td colspan="3">2. El usuario ingresa a generar el reporte de “estados de clientes”</td><td colspan="3"></td></tr>
<tr><td colspan="3">3. El sistema genera el reporte y lo presenta en la pantalla del usuario final permitiendo la opción de imprimir.</td><td colspan="3">3.1. No fue posible informar el pago por error del sistema. Se muestra un mensaje de error descriptivo. Finaliza el CU.</td></tr>
<tr><td>Flujo de excepciones</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="5">➔</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="5">➔</td></tr>
<tr><td>Observaciones</td><td colspan="5"></td></tr>
</table>

8.2.14 Requerimientos Funcional 22

El sistema permitirá generar un reporte de comprobantes emitidos y pagos realizados para el mes actual.

Nombre	Reporte de comprobantes y pagos del mes actual	**ID**	CU32
Descripción	El sistema permitirá generar un reporte de comprobantes emitidos y pagos realizados para el mes actual.		
Actores	➔ Usuario final		
Prioridad	◎ Alta ⊙ Media ◎ Baja	**Complejidad**	◎ Alta ⊙ Media ◎ Baja
Precondiciones	➔		
Postcondiciones	Éxito: ➔ Reporte generado correctamente y visualizado en pantalla.		Fracaso: ➔ Visualización de mensaje de error con la descripción de la causa del fallo.

Flujo Principal	Flujo Alternativo
1. El usuario ingresa a la sección de reportes	
2. El usuario ingresa a generar el reporte de “comprobantes y pagos del mes actual”	
3. El sistema genera el reporte y lo presenta en la pantalla del usuario final permitiendo la opción de imprimir.	**3.1.** No fue posible informar el pago por error del sistema. Se muestra un mensaje de error descriptivo. Finaliza el CU.

Flujo de excepciones	➔
Asociaciones de inclusión	➔
Asociaciones de extensión	➔
Requerimientos especiales	➔
Observaciones	

8.2.15 Requerimientos Funcional 24

El sistema permitirá visualizar diferentes métricas para la toma de decisiones tales como: Mes con mayor recaudación, cantidad de socios activos, pagos faltantes de impactar en el mes, cantidad de socios morosos, cantidad de socios activos.

<table>
<tr><td>Nombre</td><td colspan="2">Visualización de métricas</td><td>ID</td><td>CU33</td></tr>
<tr><td>Descripción</td><td colspan="4">El sistema permitirá visualizar diferentes métricas del negocio ofreciendo al usuario información para la toma de decisiones.</td></tr>
<tr><td>Actores</td><td colspan="4">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◎ Alta ⊙ Media ◎ Baja</td><td>Complejidad</td><td>◎ Alta ⊙ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="4">➔ Datos para los cálculos</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="2">Fracaso</td></tr>
<tr><td colspan="2">➔ Visualización de métricas para mejorar la toma de decisiones.</td><td colspan="2">➔ Mensaje de error descriptivo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="2">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El sistema realizará el cálculo de las siguientes métricas:
a. Ingreso Anual
b. Monto a ingresar en el mes
c. Socios activos
d. Nuevos socios
e. Rendimiento histórico en:
i. Económico
ii. En altas
f. Potenciales bajas</td><td colspan="2"></td></tr>
<tr><td colspan="3">2. Además, se deberán visualizar dichos valores en la página principal del aplicativo, comparando con el del mes o año anterior.</td><td colspan="2">2.1. No hay datos para mostrar. Se informa el valor 0 en cada campo y finaliza el CU.</td></tr>
<tr><td>Flujo de excepciones</td><td colspan="4">➔</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="4">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="4">➔</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="4">➔</td></tr>
</table>

Observaciones	

8.2.16 Requerimientos Funcional 25

El sistema permitirá el alta, baja y consulta de servicios.

Nombre	Alta de un servicio	**ID**	CU34
Descripción	El sistema permitirá realizar el alta de un servicio		
Actores	➔ Usuario final		
Prioridad	◎ Alta ◉ Media ◎ Baja	**Complejidad**	◎ Alta ◉ Media ◎ Baja
Precondiciones	➔		
Postcondiciones	Éxito: ➔ Servicio almacenado en la base de datos.	Fracaso: ➔ Mensaje de error descriptivo.	

Flujo Principal	Flujo Alternativo
1. El usuario escoge la opción “Agregar” en la lista desplegable de servicios.	
2. Se completa la totalidad de estos campos: ● Nro de servicio ● Nombre ***** ● Descripción ***** ● Valor *	**2.1.** El usuario desea cancelar la operación. **2.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
3. El usuario presiona el botón “Aceptar”	**3.1.** El usuario desea cancelar la operación. **3.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.

Flujo de excepciones	➔
Asociaciones de inclusión	➔
Asociaciones de extensión	➔
Requerimientos especiales	➔
Observaciones	

<table>
<tr><td>Nombre</td><td colspan="3">Baja de un servicio</td><td>ID</td><td>CU35</td></tr>
<tr><td>Descripción</td><td colspan="5">El sistema permitirá dar de baja un servicio</td></tr>
<tr><td>Actores</td><td colspan="5">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◎ Alta ◉ Media ◎ Baja</td><td>Complejidad</td><td colspan="2">◎ Alta ◉ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="5">➔</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="3">Fracaso</td></tr>
<tr><td colspan="2">➔ Servicio eliminado de la base de datos.</td><td colspan="3">➔ Mensaje de error descriptivo.</td></tr>
<tr><td colspan="3">Flujo Principal</td><td colspan="3">Flujo Alternativo</td></tr>
<tr><td colspan="3">1. El usuario selecciona la opción “dar de baja” en el listado de servicios.</td><td colspan="3"></td></tr>
<tr><td colspan="3">2. El sistema presenta un mensaje de confirmación para eliminar el servicio indicado.</td><td colspan="3"></td></tr>
<tr><td colspan="3">3. El usuario escoge la opción “Aceptar”.</td><td colspan="3">3.1. El usuario desea cancelar la operación.
3.1.1 Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.</td></tr>
<tr><td>Flujo de excepciones</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="5">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="5">➔</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="5">➔</td></tr>
<tr><td>Observaciones</td><td colspan="5"></td></tr>
</table>

<table>
<tr><td>Nombre</td><td colspan="2">Consultar servicios</td><td>ID</td><td>CU36</td></tr>
<tr><td>Descripción</td><td colspan="4">El sistema permitirá buscar y listar servicios.</td></tr>
<tr><td>Actores</td><td colspan="4">➔ Usuario final</td></tr>
<tr><td>Prioridad</td><td colspan="2">◎ Alta ◉ Media ◎ Baja</td><td>Complejidad</td><td>◎ Alta ◉ Media ◎ Baja</td></tr>
<tr><td>Precondiciones</td><td colspan="4">➔</td></tr>
<tr><td rowspan="2">Postcondiciones</td><td colspan="2">Éxito</td><td colspan="2">Fracaso</td></tr>
<tr><td colspan="2">➔ Visualización de un listado de servicios.</td><td colspan="2">➔ Visualización de un mensaje de error que describe la causa del fallo.</td></tr>
<tr><td colspan="2">Flujo Principal</td><td colspan="3">Flujo Alternativo</td></tr>
<tr><td colspan="2">1. El usuario selecciona la opción “Padrón” del menú de Servicios.</td><td colspan="3"></td></tr>
<tr><td colspan="2">2. El sistema presenta una tabla de todos los servicios con la siguiente información:
● N° Zona
● Nombre
● Cobradores</td><td colspan="3">2.1. No existen servicios.
2.1.1 Se muestra el siguiente mensaje: “No existen servicios”. Finaliza el CU.</td></tr>
<tr><td>Flujo de excepciones</td><td colspan="4">➔</td></tr>
<tr><td>Asociaciones de inclusión</td><td colspan="4">➔</td></tr>
<tr><td>Asociaciones de extensión</td><td colspan="4">➔ CU22</td></tr>
<tr><td>Requerimientos especiales</td><td colspan="4">➔</td></tr>
<tr><td>Observaciones</td><td colspan="4"></td></tr>
</table>

8.2.17 Requerimientos Funcional 26

El sistema permitirá la comercialización de los servicios.

Nombre	Venta de un servicio		**ID**	CU37
Descripción	El sistema permitirá realizar la venta de un servicio			
Actores	➔ Usuario final			
Prioridad	◎ Alta ⊙ Media ◎ Baja	**Complejidad**	◎ Alta ⊙ Media	◎ Baja
Precondiciones	➔			
Postcondiciones	Éxito	Fracaso		
	➔ Venta almacenada en la base de datos.	➔ Mensaje de error descriptivo.		

Flujo Principal	**Flujo Alternativo**
1. El usuario escoge la opción “Venta” en la lista desplegable de servicios.	
2. Se selecciona el tipo de servicio a vender.	**2.1.** El usuario desea cancelar la operación. **2.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.
3. El usuario presiona el botón “Aceptar”	**3.1.** El usuario desea cancelar la operación. **3.1.1** Se presiona el botón “Cancelar” o se regresa a la página anterior. Finaliza el CU.

Flujo de excepciones	➔
Asociaciones de inclusión	➔
Asociaciones de extensión	➔
Requerimientos especiales	➔
Observaciones	