



**UNCUYO**  
UNIVERSIDAD  
NACIONAL DE CUYO



**FACULTAD  
DE INGENIERÍA**

## Ingeniería del Software II

### GUÍA INFORME

### **PROYECTO “NTM CONSORCIO”**

(Trabajo Práctico N°1)



#### GRUPO (NTM):

- Ignacio Coppede – Legajo: 14215
- Mauro Sorbello – Legajo: 14006
- Tomás Rando – Legajo: 14004

#### PROFESORES:

- Lucias Cortez.
- Leandro Spadaro

#### AÑO:

- 2024

<https://github.com/NTMConsortio/ProyectoConsortio>

## **ÍNDICE.**

### **1. Metodología de desarrollo.**

### **2. Desarrollo de Software.**

Iteración N°1.

Iteración N°2.

Iteración N°3.

Iteración N°4.

### **3. Propuesta de Mejora.**

### **4. Propuesta de refactorización.**

### **5. Material de Exposición en Conferencia.**

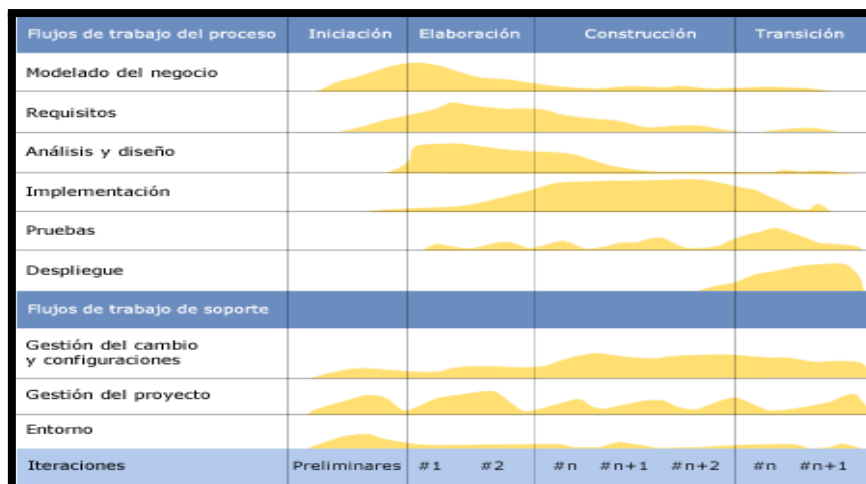
### **6. Referencias.**

### **7. Anexos.**

## 1. Metodología de desarrollo.

La metodología que utilizamos en el trabajo práctico fue RUP (proceso unificado racional), la cual, como hemos visto, es una metodología iterativa que tiene 4 etapas:

- **Iniciación:** Se define el alcance y objetivos del proyecto a desarrollar. En esta etapa básicamente se captan los requisitos principales del sistema y se intenta comprender el negocio. Además se pueden realizar algunas estimaciones preliminares de tiempo y costo de desarrollo
- **Elaboración:** En esta fase se profundiza más en los requisitos ya que para cuando esta etapa termine se debe tener una idea más clara del diseño del sistema y un plan detallado de desarrollo. En esta etapa incluso es posible empezar con un mínimo de codificación.
- **Construcción:** Se codifica el software y se realizan algunas pruebas de acuerdo con lo establecido en los requisitos. Se asegura la funcionalidad del producto.
- **Transición:** En esta fase se intenta llevar el producto al entorno de producción, se realizan algunas pruebas en entornos similares al de producción y se valida el software.



## 2. Desarrollo de Software.

### Iteración 1

## Iniciación

### Modelado de Negocio.

#### Especificación del proyecto.

El proyecto consiste en una web app para la administración del cobro de las expensas de un consorcio. El mismo debe notificar por whatsapp a los habitantes todos los meses informando del valor de las expensas junto a un link de pago. Además, una vez realizado y cargado el pago, se debe enviar por email el recibo correspondiente. Por último, se debe poder gestionar las multas y expensas adeudadas de los departamentos del consorcio.

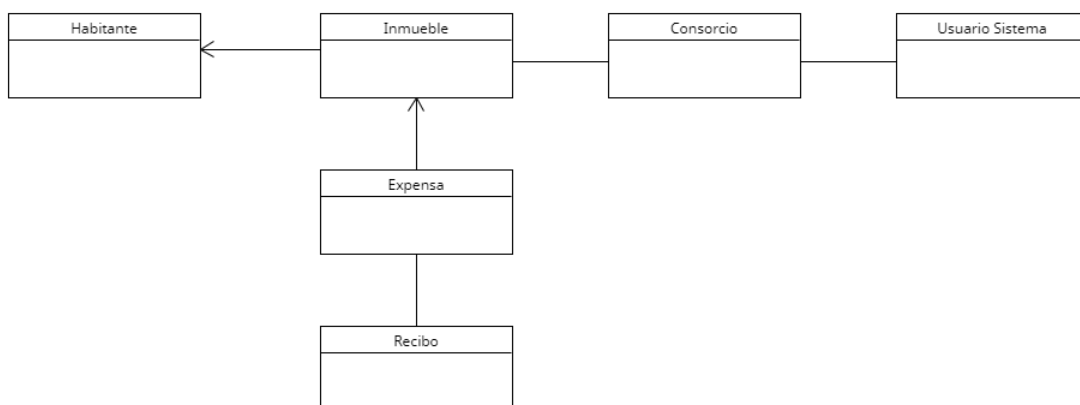
#### Entrevista con el cliente.

Algunas de las preguntas y respuestas que fueron realizadas al cliente son:

- ¿Todos los inquilinos pagan las mismas expensas? Sí, todos pagan las mismas
- ¿Cada persona del personal administrativo tiene su propio usuario y contraseña? Sí, cada uno tendrá su propio usuario y contraseña
- ¿Hay algún usuario con privilegios especiales? Por ahora no va a haber ningún usuario con privilegios especiales.
- ¿Dónde se alojará el servidor? –
- ¿Qué pasa si los habitantes no poseen whatsapp o email? Deberán hacer el proceso físicamente.

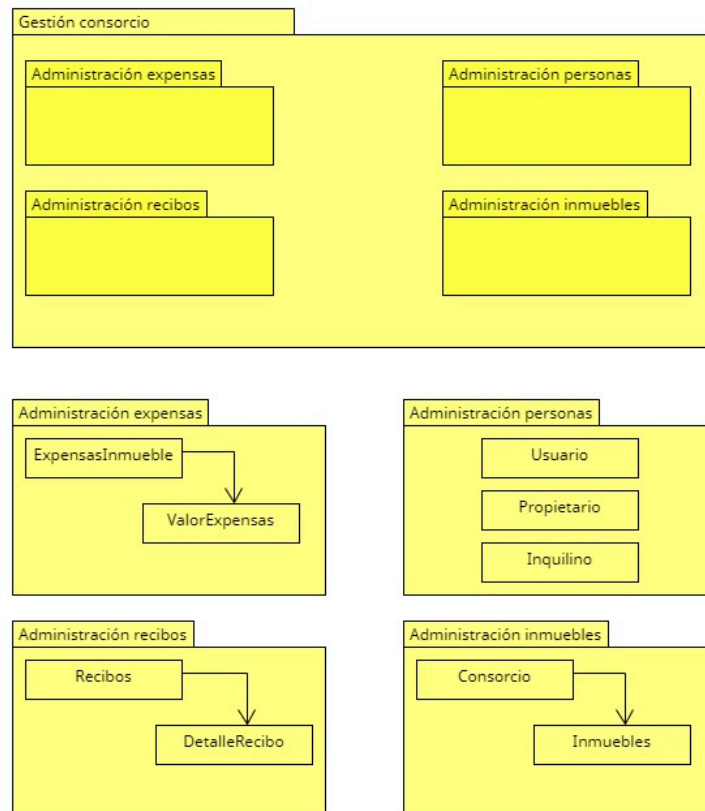
#### Diagrama de clases de negocio.

Se realizó el siguiente diagrama correspondiente al modelado del negocio.



#### Modelado de paquetes de dominio.

Se realizó el diagrama de paquetes **del dominio**. Este diagrama tiene el objetivo de agrupar clases relacionadas para lograr una comprensión óptima del dominio del problema. Nos ayuda a mejorar nuestra visión de las relaciones y la organización de las posibles clases del sistema.



## Requisitos.

Para la etapa de iniciación se buscaron los primeros requisitos funcionales de la aplicación. Es decir, no se pensaron “todos” los que iba a tener el sistema final, sino que solamente se identificaron los más críticos y más importantes para el cliente.

### Requisitos funcionales.

La aplicación debe cumplir los siguientes requisitos funcionales, o, en su defecto, dejarlos para una siguiente etapa.

- Notificar por whatsapp las expensas mensuales automáticamente.
- Registrar pagos.
- Generar recibos de pago y enviarlos por email automáticamente.
- Tener registro de multas.
- Tener registro de deudas.
- Modificar los valores de las expensas.

### Diagramas de caso de uso.

Se realizó la tabla de casos de uso con las prioridades correspondientes. Nuevamente, fueron realizados teniendo en cuenta los requisitos funcionales identificados en esta etapa y en la siguiente será expandida.

<b>Crítica</b>	<b>Media</b>	<b>Baja</b>
<ul style="list-style-type: none"> <li>- Generar expensa mensual por departamento</li> <li>- Consultar expensas</li> <li>- Notificar expensas</li> <li>- Registrar pago</li> <li>- Generar recibo</li> <li>- Notificar recibo</li> </ul>	<ul style="list-style-type: none"> <li>- ABM administrativo</li> <li>- ABM habitante</li> <li>- ABM inmueble</li> <li>- ABM expensas</li> </ul>	<ul style="list-style-type: none"> <li>- ABM multa</li> <li>- Loguear</li> </ul>

## Gestión de Proyecto.

### Viabilidad.

El proyecto propuesto es viable tanto desde el punto de vista técnico como económico. El equipo de desarrollo cuenta con los conocimientos y habilidades necesarias para llevarlo a cabo con éxito. Además, al tratarse de un desarrollo de menor escala, resulta rentable para el cliente, ya que será propietario del producto mediante un único pago inicial, evitando así la necesidad de incurrir en costos mensuales por el uso de aplicaciones comerciales similares, que además suelen ofrecer menos opciones de personalización.

### Estimación de tiempos de desarrollo e implementación.

Se estimó un total de 38 días hábiles para el desarrollo e implementación de la aplicación web del consorcio. El proyecto inició el 5 de agosto y está previsto que finalice el 27 de septiembre. Durante este periodo se llevarán a cabo todas las fases del proyecto, desde el diseño y desarrollo hasta las pruebas y el despliegue final en producción.

La planificación se basó en un cronograma detallado que distribuye las tareas entre los distintos equipos de desarrollo, asegurando que se cumplan los hitos establecidos dentro del plazo previsto.

### Estimación de riesgos.

Para la estimación de tiempos y costos, se evaluaron diversos factores de riesgo que podrían afectar el desarrollo del proyecto. Entre ellos, se contempló la posibilidad de bajas por enfermedad de algún miembro del equipo, lo que podría ocasionar retrasos en las fechas de entrega previamente acordadas. Asimismo, se consideró el tiempo adicional que podría ser necesario para resolver posibles complicaciones en la integración de APIs con las que el equipo no está familiarizado, lo que podría generar demoras en el desarrollo si no se gestionan adecuadamente.

### Estimación de costos.

La estimación de costos para el desarrollo e implementación de la aplicación web se realizó tomando en cuenta los honorarios de desarrolladores de aplicaciones full stack, basándose en las recomendaciones del **Consejo Profesional de Ciencias Informáticas de Córdoba (CPCI)**. Esta estimación incluye los costos asociados al trabajo de los desarrolladores, así como otros gastos necesarios para completar el proyecto, como licencias de software y costos operativos.

Para una visión detallada y específica de los costos, se remite al **Anexo 1**, donde se encuentran desglosados todos los elementos y cálculos utilizados en la estimación.

### **Forma de realización del proyecto. Comprar/construir.**

El proyecto se realizará mediante la construcción del mismo. No se utilizará ningún enlatado ya que el sistema que se realizó era “simple” y los integrantes del equipo se capacitaron durante el proceso del mismo para poder completar el desarrollo satisfactoriamente.

### **Presupuesto.**

El presupuesto asociado al desarrollo e implementación de la aplicación web del consorcio están detalladamente desglosados en el **Anexo 1**. Este anexo proporciona un desglose exhaustivo de los gastos involucrados, incluyendo recursos humanos, software, hardware y otros gastos operativos.

### **Seguimiento Avance Proyecto.**

El seguimiento del avance del proyecto se llevará a cabo mediante una serie de reuniones semanales con el cliente, en las cuales se comprobarán los avances iterativos. Estas reuniones se han acordado mutuamente para los **días viernes**, en el horario de **17:00 a 20:00 horas**.

Durante estas sesiones, se presentarán los progresos alcanzados, se discutirán los entregables y se abordarán cualquier inquietud o ajuste necesario. Este enfoque permitirá una comunicación continua y efectiva, asegurando que el proyecto se mantenga alineado con los requisitos del cliente y se realicen ajustes oportunos según sea necesario.

## **Entorno**

### **Arquitectura**

La aplicación se desarrolló basándose en capas, estas son vista, controlador, lógica del negocio, servicio y acceso a datos. Seguir este enfoque nos permite lograr una división de responsabilidades, facilitar el mantenimiento y la escalabilidad del sistema. Además, es muy útil para permitir la reutilización del código en proyectos futuros. Se utilizaron diferentes tecnologías para lograr esto. Una de ellas es Java Enterprise Edition, la cual soporta muy bien este modelo.

### **Tecnología**

Para el desarrollo del proyecto se utilizaron diferentes tecnologías:

- Lenguaje de programación: Java Enterprise Edition
- Kit desarrollo Java: JDK versión 7u60
- Entorno ejecución Java: JRE versión 7u60
- Base de datos relacional: MySQL versión 5.0
- IDE para desarrollo: Netbeans 8.0.2
- Framework para interfaces: JavaServer Faces
- Biblioteca componentes JSF: Primefaces

Estas tecnologías nos brindaron un entorno de desarrollo óptimo que nos permitió desarrollar la aplicación objetivo. Esto es debido a su extensa trayectoria y a su compatibilidad con los patrones utilizados.

### **Base de datos**

La base de datos utilizada fue MySql en su versión 5.0. Además, para el gráfico del DER (Diagrama de entidad relación) se utilizó la herramienta MySQL Workbench. Se eligió esta herramienta en específico ya que es compatible con Java EE y JPA. Además, la versión utilizada es gratuita, por lo que reduce el coste de desarrollo.

### **Tipo de software**

El tipo de software desarrollado es una aplicación web, esto es debido a que el cliente solicitó específicamente este tipo de desarrollo. Además, el mercado actual está enfocado actualmente en aplicaciones web, dejando atrás a las de escritorio. Para la realización de la interfaz visual se utilizó el framework JavaServer Faces.

### **Herramienta de comunicación del equipo**

Para la comunicación entre los miembros del equipo de desarrollo se utilizaron principalmente 3 herramientas: Github para el traspaso de código y la programación en conjunto, Google Meet para las reuniones ocasionales y Whatsapp para la comunicación continua.

### **Herramienta para la captura de requerimientos**

Para la captura de requerimientos el cliente brindó un documento por escrito con distintas funcionalidades que necesitaba del software a desarrollar. Sin embargo, para complementar esto, se realizaron una serie de entrevistas con el cliente a lo largo del proceso de desarrollo, tanto para capturar nuevos requerimientos como para la clarificación de los mismos.

### **Herramienta de prototipado de interfaces gráficas UI**

Para el prototipado de interfaces gráficas se utilizó la herramienta Balsamiq, esta nos permitió realizar prototipos simples pero útiles para poder lograr una mejor comprensión de los requerimientos del sistema.

### **Herramienta de modelado**

Las herramientas que se utilizaron para realizar el modelado fueron principalmente Miro, draw.io y UMLletino. Estas permiten diagramar fluidamente los modelos visuales. Además, Miro y draw.io incluyen herramientas para trabajar cooperativamente a la hora de realizarlos.

### **Herramienta de codificación**

La herramienta de codificación que se utilizó principalmente fue Netbeans 8.0.2. En ella se escribió casi todo el código realizado, sin embargo, también se hizo uso de Visual Studio Code, el cual nos permitió una más eficiente y veloz refactorización del código. Por último, otra herramienta que fue de utilidad fue el versionador de código Git junto a Github.

### **Herramienta de pruebas de software**

Para el desarrollo del proyecto no se utilizó ninguna herramienta especializada en pruebas. Las pruebas realizadas fueron hechas manualmente utilizando la interfaz gráfica del sistema.

### **Herramienta de gestión de proyecto**

No se hizo uso de ninguna herramienta especializada específicamente en la gestión de proyectos. Sin embargo, se utilizó whatsapp para suplir esta función. Es decir, el progreso



individual y las tareas faltantes iban siendo comunicadas por este medio a medida que el proyecto iba avanzando.

## **Elaboración**

En esta fase nos enfocamos en refinar y detallar los requisitos y el entendimiento del negocio que adquirimos en la fase de iniciación.

### **Requisitos.**

Para esta etapa los requisitos funcionales fueron ampliados debido al mejor entendimiento de las necesidades del cliente. Esto nos permitió además ampliar la tabla de casos de uso y poder realizar nuestro diagrama junto a los escenarios de los casos de uso críticos.

### **Requisitos funcionales.**

La aplicación debe cumplir los siguientes requisitos funcionales, o, en su defecto, dejarlos para una siguiente etapa.

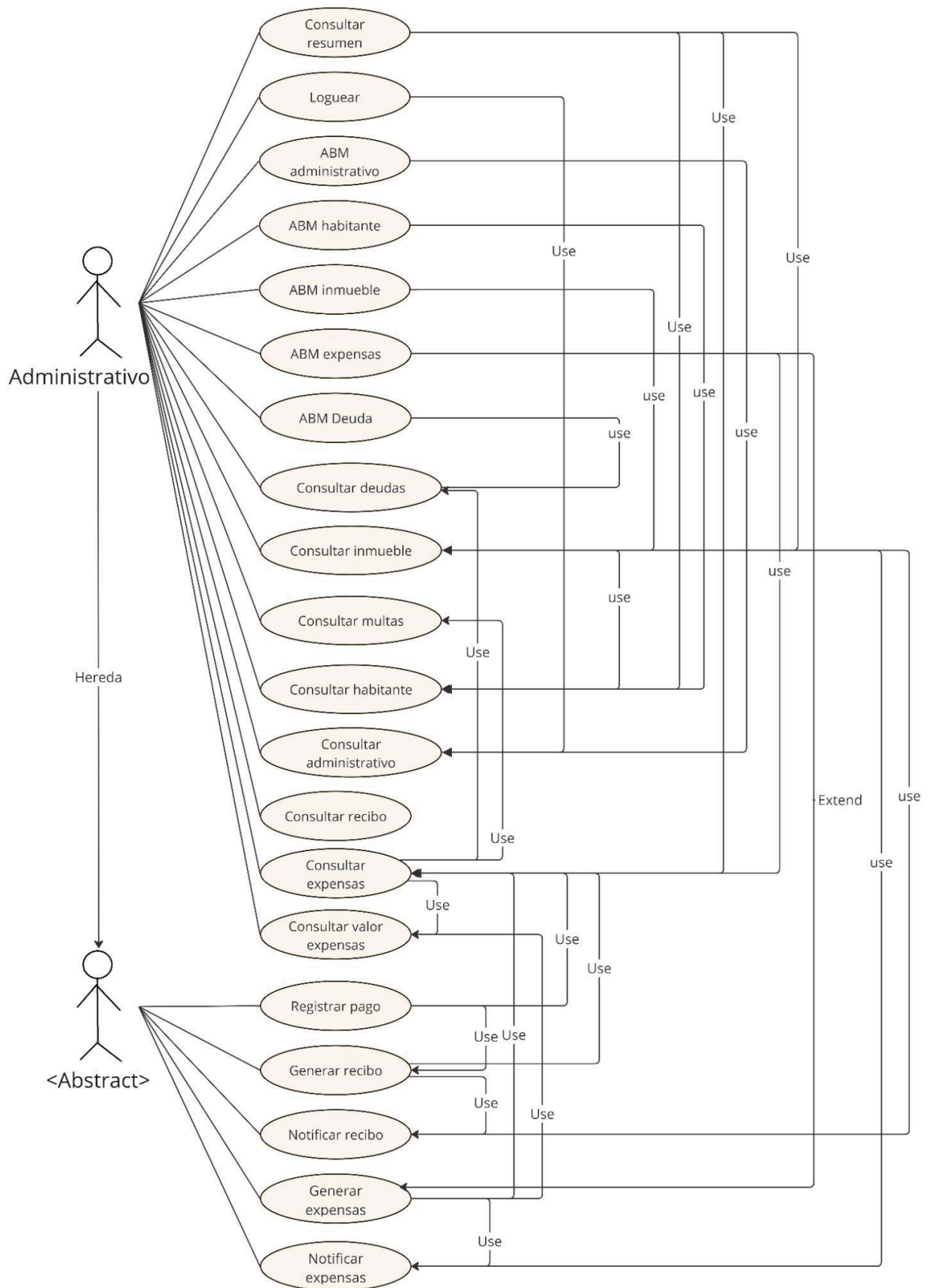
- Notificar por whatsapp las expensas mensuales automáticamente.
- Registrar pagos.
- Generar recibos de pago y enviarlos por email automáticamente.
- Tener registro de multas.
- Tener registro de deudas.
- Modificar los valores de las expensas.
- Mostrar un resumen de las expensas de los departamentos.
- Dar de alta departamentos, habitantes, deudas, multas y expensas.
- Dar de baja departamentos, habitantes, deudas, multas y expensas.
- Modificar departamentos, habitantes, deudas, multas y expensas.

### **Diagrama de casos de uso.**

Se elaboró una tabla con los casos de uso separados según su prioridad incorporando más casos de uso identificados.

<b>Crítica</b>	<b>Media</b>	<b>Baja</b>
- Generar expensa mensual por departamento - Consultar expensas - Notificar expensas - Registrar pago - Generar recibo - Notificar recibo	- ABM administrativo - ABM habitante - ABM inmueble - ABM expensas	- ABM multa - Loguear - Consultar deudas - Consultar multas - Consultar habitante - Consultar administrativo - Consultar recibo - Consultar expensas - Consultar valor expensas

Posteriormente, se elaboró el diagrama de casos de uso con los respectivos actores. Cabe aclarar que hasta el momento no se ha realizado el empaquetamiento de estos.



## Escenarios casos de uso.

Se elaboraron los escenarios de casos de uso para los que se habían designado como críticos.

Generar expensas mensuales por departamento	
Descripción	Se generan las expensas para cada departamento correspondiente al mes actual
Requisitos especiales	Tener conexión a internet
Precondiciones	Tener usuario con privilegios administrativos. Tener datos de alta los departamentos. Tener datos de alta los habitantes. Tener datos de alta los valores de las expensas
Poscondiciones	Se generaran todas las expensas y quedarán listas para ser enviadas
Activación	El usuario selecciona "Generar expensas"
Flujo principal	1- El usuario ingresa a la opción "Generar expensas". 2- El sistema llama al caso de uso "Consultar valor expensas". 3- El sistema solicita confirmación mostrando el monto actual de las expensas. 4- El usuario acepta. 5- El sistema llama al caso de uso "Consultar expensas" con lo que genera las expensas mensuales de todos los departamentos. 6- El sistema muestra un resumen de los departamentos junto a las expensas generadas. 7- El usuario selecciona "Enviar expensas" 8- El sistema solicita confirmación. 9- El usuario acepta. 10- El sistema llama al caso de uso "Notificar expensas".
Flujo alternativo	3.1- El usuario cancela 3.1.1- Fin del caso de uso  6.1- El usuario cancela 6.1.1- Fin del caso de uso  8.1- El usuario cancela 8.1.1- Se vuelve al paso 5
Puntos de extensión	3.2- El usuario selecciona "Modificar expensas" 3.2.1- El sistema llama al caso de uso "Modificar expensas"

Consultar expensas	
Descripción	Se consultan las expensas de un departamento específico
Requisitos especiales	Tener conexión a internet
Precondiciones	Tener usuario con privilegios administrativos. Tener datos de alta los departamentos. Tener datos de alta los habitantes. Tener datos de alta los valores de las expensas Tener generadas las expensas
Poscondiciones	Se mostrarán las expensas del departamento seleccionado.
Activación	El usuario selecciona "Consultar expensas"
Flujo principal	1- El usuario ingresa a la opción "Consultar expensas". 2- El sistema solicita un código único de departamento. 3- El usuario ingresa el código del departamento. 4- El sistema llama al caso de uso "Consultar inmueble" y verifica la existencia del departamento ingresado. 5- El sistema llama al caso de uso "Consultar valor expensas" 6- El sistema llama al caso de uso "Consultar deudas" 7- El sistema llama al caso de uso "Consultar multas" 8- El sistema muestra las expensas totales del departamento junto con las deudas y multas. 9- Fin del caso de uso.
Flujo alternativo	2.1- El usuario cancela. 2.1.1- Fin del caso de uso.  4.1 – El sistema verifica que el código ingresado no existe. 4.1.1 – Se vuelve al paso 2.

### Notificar recibo: Caso de uso específico del equipo

	Notificar recibo
Descripción	Se notifica el recibo del pago de las expensas por habitante, por departamento
Requisitos especiales	Tener conexión a internet
Precondiciones	Tener usuario con privilegios administrativos. Tener dados de alta los departamentos. Tener dados de alta los habitantes. Tener dados de alta los valores de las expensas. Haber enviado las expensas Haber generado el recibo
Poscondiciones	Se enviará el recibo de la expensa
Activación	El usuario selecciona "Notificar recibo"
Flujo principal	1- El usuario ingresa a la opción "Notificar recibo". 2- El sistema va al caso de uso "Buscar recibo" 3- El sistema verifica que el recibo posea detalles 4- El sistema genera el archivo del recibo 5- El sistema va al caso de uso "Buscar cuenta correo" 6- El sistema envía el recibo. 7- Fin del caso de uso
Flujo alternativo	2.1- El sistema verifica que el recibo no existe y muestra un error 2.2- Fin del caso de uso  3.1- El sistema verifica que el recibo no posee detalles y muestra un error 3.2- Fin del caso de uso  4.1- El sistema verifica que hubo un error al generar el archivo y muestra un error 4.2- Fin del caso de uso  5.1- El sistema verifica que la cuenta de correo no se encontró y muestra un error 5.2- Fin del caso de uso  6.1- El sistema verifica que hubo un error al enviar el recibo y muestra un error 6.2- Fin del caso de uso

### Notificar expensas: Caso de uso específico del equipo

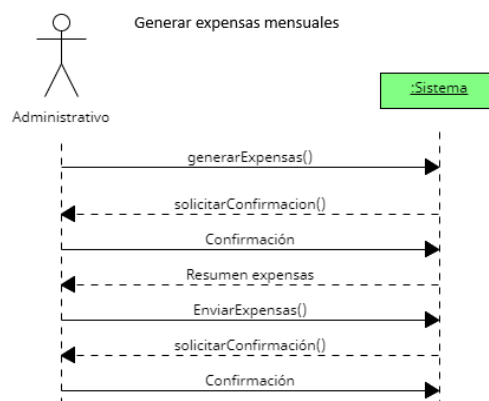
	Notificar expensas
Descripción	Se notifica el valor de la expensa a pagar por el habitante, por departamento
Requisitos especiales	Tener conexión a internet
Precondiciones	Tener usuario con privilegios administrativos. Tener dados de alta los departamentos. Tener dados de alta los habitantes. Tener dados de alta los valores de las expensas. Tener generadas las expensas
Poscondiciones	Se enviaran las expensas a pagar
Activación	El usuario selecciona "Notificar expensas"
Flujo principal	1- El usuario ingresa a la opción "Notificar expensas". 2- El sistema va al caso de uso "Buscar expensa del inmueble" 3- El sistema envía al destinatario el valor de las expensas junto con información del cliente y el departamento. 4- El sistema muestra confirmación. 5- Fin de caso de uso
Flujo alternativo	2.1- El sistema no encuentra una expensa para el inmueble y muestra un error 2.2- Fin del caso de uso  3.1- El sistema verifica que el envío fallo y muestra mensaje de error. 3.2- Fin de caso de uso

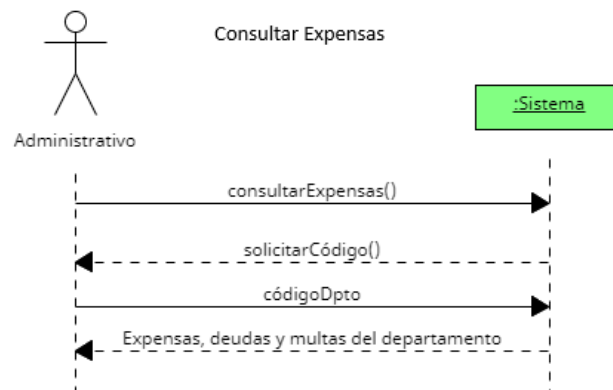
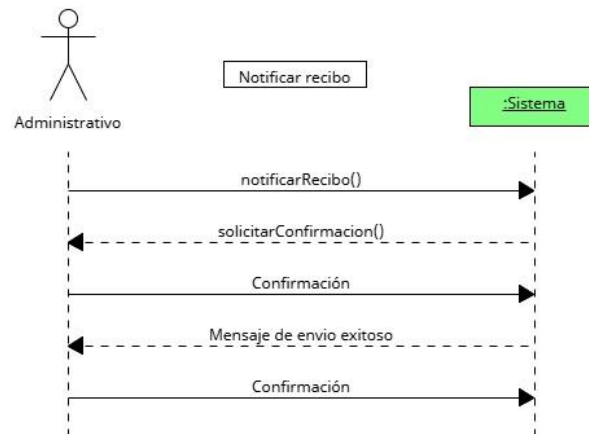
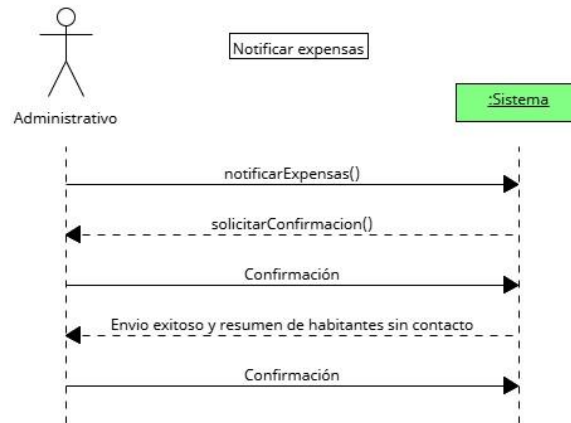
	Registrar pago
Descripción	Se guardara un registro del pago que se realizo para un departamento.
Requisitos especiales	Tener conexión a internet. Tener constancia del pago de la exensa.
Precondiciones	Tener usuario con privilegios administrativos. Tener dados de alta los departamentos. Tener dados de alta los habitantes. Tener dados de alta los valores de las expensas.
Poscondiciones	Se guardara un registro del pago que se realizo para un departamento.
Activación	El usuario selecciona "Registrar pago"
Flujo principal	1- El usuario ingresa a la opción "Registrar Pago". 2- El sistema solicita un código único de departamento. 3- El usuario ingresa el código del departamento. 4- El sistema verifica la existencia del departamento ingresado. 5- El sistema llama al caso de uso "Consultar expensas" 6- El sistema muestra las expensas totales del departamento, junto con las deudas y multas. 7- El sistema pide que guarde un comprobante del pago. 8- El usuario ingresa el comprobante de pago. 9- El sistema llama al caso de uso "Generar recibo"
Flujo alternativo	2.1- El usuario cancela. 2.1.1- Fin del caso de uso.  4.1 – El sistema verifica que el código ingresado no existe. 4.1.1 – Se vuelve al paso 2.

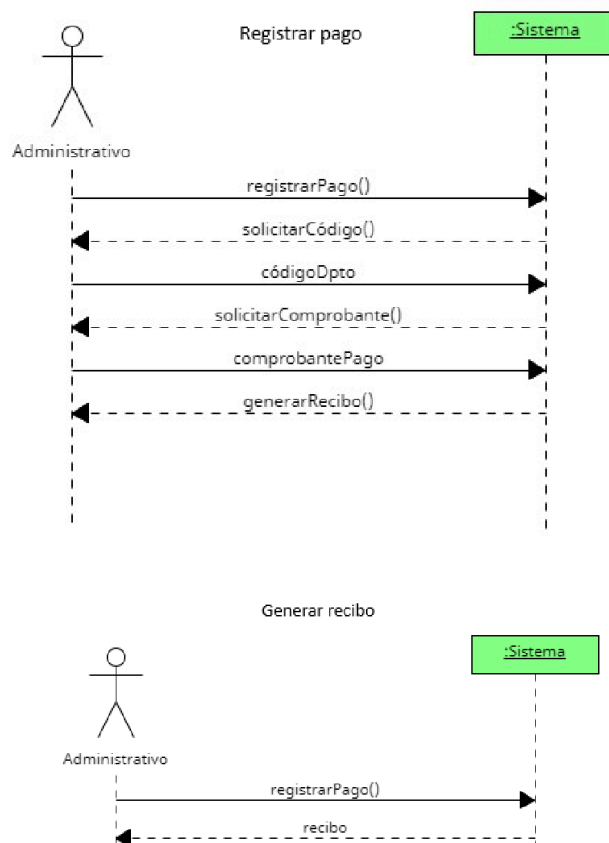
	Generar recibo
Descripción	Se genera un recibo del pago de las expensas.
Requisitos especiales	Tener conexión a internet.
Precondiciones	Tener el pago registrado. Tener usuario con privilegios administrativos. Tener dados de alta los departamentos. Tener dados de alta los habitantes. Tener dados de alta los valores de las expensas.
Poscondiciones	Se devuelve un recibo de la expensa pagada.
Activación	El usuario registro el pago.
Flujo principal	1- El usuario registro el pago 2- El sistema llama al caso de uso "Consultar expensas". 3- El sistema genera un recibo del pago de las expensas. 4- Fin del caso de uso.
Flujo alternativo	

## Diagrama de secuencia del sistema.

Para cada escenario se realizó el correspondiente diagrama de secuencia. Estos diagramas fueron la primera aproximación para lograr el correcto entendimiento de las necesidades del cliente.



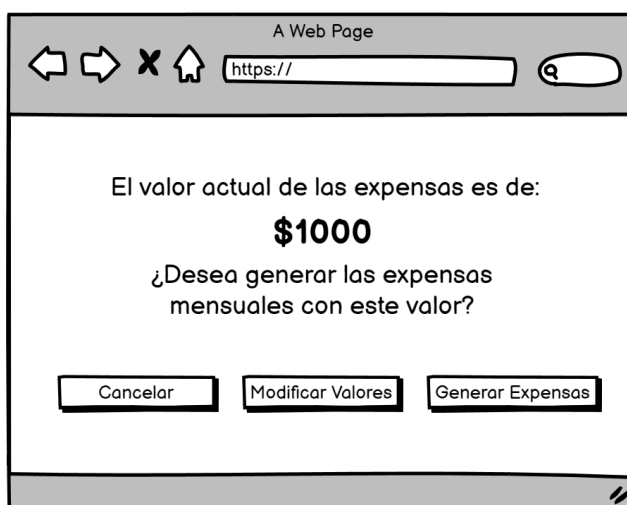




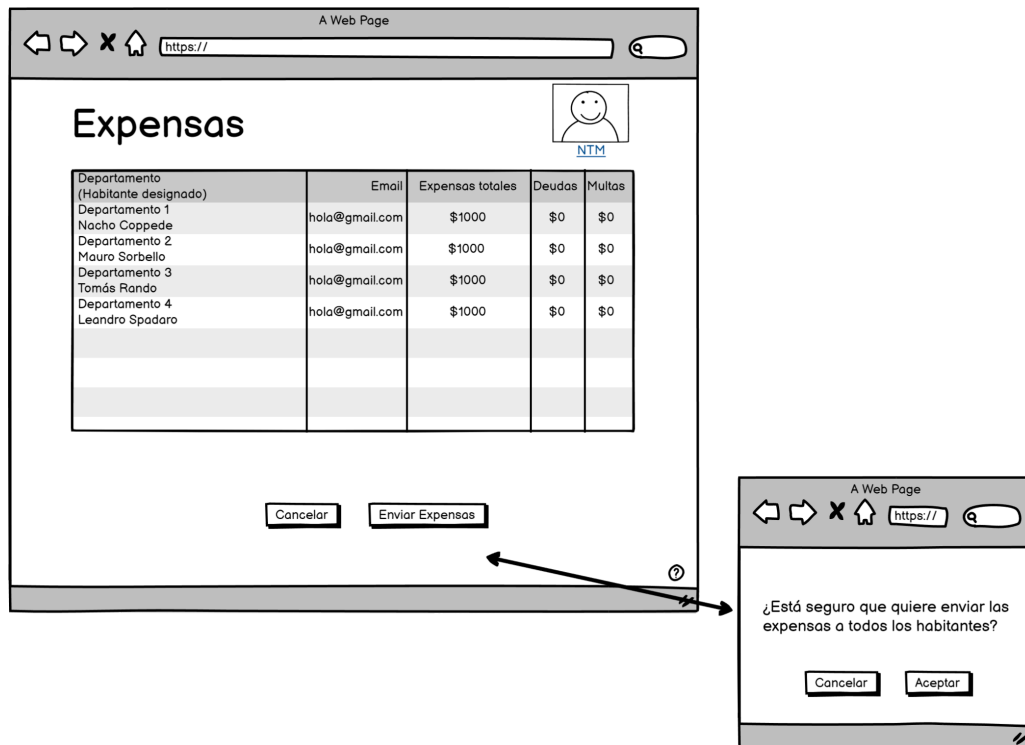
### Prototipado de pantallas.

Además, se realizó el prototipado de las correspondientes pantallas. Estos prototipos nos ayudan a validar los requisitos del cliente, es decir, nos ayudan a disminuir la ambigüedad que pudiese existir luego del planteo de necesidades de este. Además, nos ayudan a detectar rápidamente problemas de usabilidad. Por otro lado, es más entendible para el cliente observar una pantalla “real”, por lo que unificamos ambos “lenguajes” (el del equipo de desarrollo y el cliente) en uno solo.

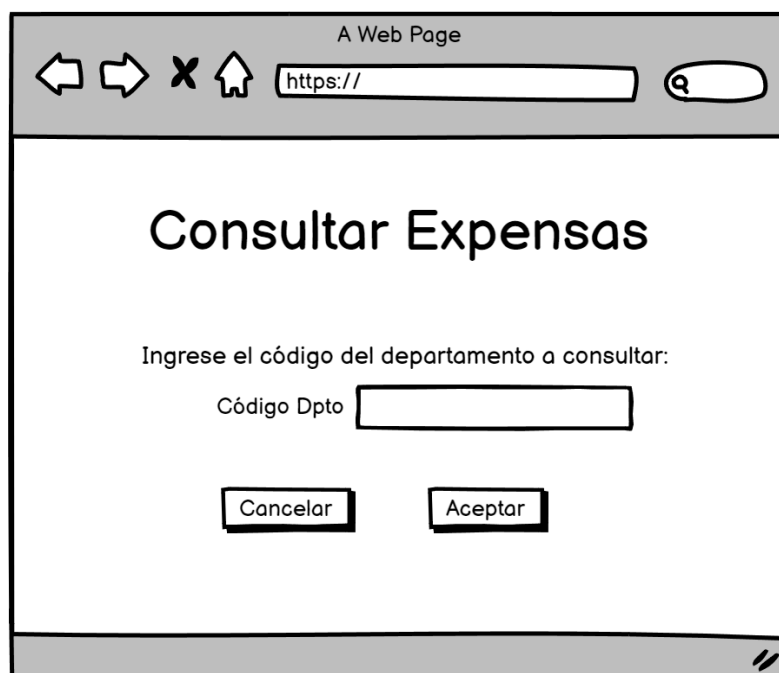
### Generar expensas mensuales







Consultar expensas.






A Web Page

https://

# Expensas



NTM

Departamento (Habitante designado)	DNI designado	Email	Expensas totales	Deudas	Multas
Departamento 1 Nacho Coppede	45131313	hola@gmail.co	\$1000	\$0	\$0


Cerrar

## Notificar expensa.

A Web Page

http://

# Expensas



NTM

Departamento (Habitante designado)	Email	Expensas totales	Deudas	Multas
Departamento 1 Nacho Coppede	hola@gmail.com	\$1000	\$0	\$0
Departamento 2 Mauro Sorbello	hola@gmail.com	\$1000	\$0	\$0
Departamento 3 Tomás Rando	hola@gmail.com	\$1000	\$0	\$0
Departamento 4 Leandro Spadaro	hola@gmail.com	\$1000	\$0	\$0
Departamento 1 Juan Hernandez	-	\$1000	\$0	\$0
Departamento 2 José Díaz	-	\$1000	\$0	\$0

Cancelar Enviar Expensas

A Web Page

http://


¿Está seguro que quiere enviar las expensas a todos los habitantes?

Cancelar Aceptar

A Web Page

http://

# Habitantes sin contactar



NTM

Departamento (Habitante designado)	Email	Expensas totales	Deudas	Multas
Departamento 1 Juan Hernandez	-	\$1000	\$0	\$0
Departamento 2 José Díaz	-	\$1000	\$0	\$0


Aceptar

## Notificar recibo.

A Web Page

http://

# Recibos



NTM

Departamento (Habitante designado)	Email	Pago	Deudas	Multas
Departamento 1 Nacho Coppede	hola@gmail.com	\$1000	\$0	\$0
Departamento 2 Mauro Sorbello	hola@gmail.com	\$1000	\$0	\$0
Departamento 3 Tomás Rando	hola@gmail.com	\$1000	\$0	\$0
Departamento 4 Leandro Spadaro	hola@gmail.com	\$1000	\$0	\$0
Departamento 1 Juan Hernandez	-	\$1000	\$0	\$0
Departamento 2 José Díaz	-	\$1000	\$0	\$0

Cancelar Enviar Recibo

A Web Page

http://


¿Está seguro que quiere enviar los recibos a todos los habitantes?

Cancelar Aceptar

A Web Page

http://

# Habitantes sin contactar



NTM

Departamento (Habitante designado)	Email	Pago	Deudas	Multas
Departamento 1 Juan Hernandez	-	\$1000	\$0	\$0
Departamento 2 José Díaz	-	\$1000	\$0	\$0

Aceptar

### Requisitos no funcionales.

Se tuvo en cuenta algunos requisitos no funcionales a tener en cuenta para el desarrollo de la aplicación.

Facilidad de uso	<ul style="list-style-type: none"><li>● Interfaz intuitiva y fácil de utilizar.</li><li>● La aplicación debe ser sencilla de utilizar para lograr facilitar la capacitación lo máximo posible.</li></ul>
Fiabilidad	<ul style="list-style-type: none"><li>● Se debe garantizar la integridad de los datos que se almacenen, sin pérdidas de estos.</li><li>● Copias de seguridad de los datos.</li></ul>
Rendimiento	<ul style="list-style-type: none"><li>● Tiempo de respuesta decente y acorde a la operación ejecutada.</li></ul>
Soporte	<ul style="list-style-type: none"><li>● Se resolverán preguntas acerca del uso del sistema y se ofrecerá una sesión de capacitación para aprender acerca del uso del sistema.</li><li>● Se proveerán videos pequeños para capacitar sobre el uso del sistema</li></ul>
Implementación	<ul style="list-style-type: none"><li>● Está previsto que funcione en cualquier navegador web de cualquier dispositivo.</li><li>● Se desplegará en un servicio de host como Don Web.</li></ul>
Interfaz	<ul style="list-style-type: none"><li>● Interfaz sencilla que permita comprender fácilmente las funcionalidades sin mayor explicación.</li></ul>
Operaciones	<ul style="list-style-type: none"><li>● No está previsto un periodo de mantenimiento posterior al desarrollo, sin embargo, se puede solicitar con un costo adicional.</li></ul>

Empaquetamiento	<ul style="list-style-type: none"> <li>• Los componentes se colocarán organizados en paquetes siguiendo el modelo vista-controlador.</li> </ul>
Legales	<ul style="list-style-type: none"> <li>• Se utilizan lenguajes y herramientas con licencias gratuitas para el desarrollo.</li> </ul>

## **Análisis y diseño.**

### **Diagrama de clases de diseño.**

El diagrama de clases de diseño nos permite describir los componentes principales del sistema y cómo se relacionan entre sí. Es decir, es una representación visual de la arquitectura del sistema y su correcta elaboración nos permite programar con mayor velocidad y fluidez, ya que para programar las clases es posible seguir de guía el diagrama. Este fue modificado varias veces en el proceso de su elaboración. Primero se hizo una primera aproximación entre los miembros del grupo y luego se unificó con el resto de los equipos de desarrollo. Sin embargo, el diagrama de clases fue actualizado con especialmente algunos métodos que se adaptaban más al estilo de programación del presente equipo. Esta versión puede ser encontrada en el siguiente link:

<https://drive.google.com/file/d/1sNDbdTXw3bsgitSeNrlBosnDWWs8-qXf/view?usp=sharing>

### **Diagrama de secuencia de diseño.**

Los diagramas de secuencia de diseño nos permiten visualizar la interacción entre las distintas partes del sistema (objetos, etc). Se enfoca no solo en representar la estructura interna de los componentes del software, si no que también de la lógica interna de los procesos. Estos diagramas fueron realizados únicamente para los casos de uso que les fueron asignados al equipo. Estos se tratan de notificar recibo y notificar expensas. Ambos diagramas quedaron demasiado extensos a causa de la gran interacción de las distintas partes del sistema, por ello, para permitir una mejor visualización, se adjunta el link a cada uno de ellos.

Notificar expensas:

[https://drive.google.com/file/d/1JO7EVAw\\_uX9EkU6ea5fRnb66-DiJ5TXc/view?usp=sharing](https://drive.google.com/file/d/1JO7EVAw_uX9EkU6ea5fRnb66-DiJ5TXc/view?usp=sharing)

Notificar recibos:

[https://drive.google.com/file/d/1PSA8-nMcsjDPJP5qtaVaGoSJX\\_3\\_3fPQ/view?usp=sharing](https://drive.google.com/file/d/1PSA8-nMcsjDPJP5qtaVaGoSJX_3_3fPQ/view?usp=sharing)

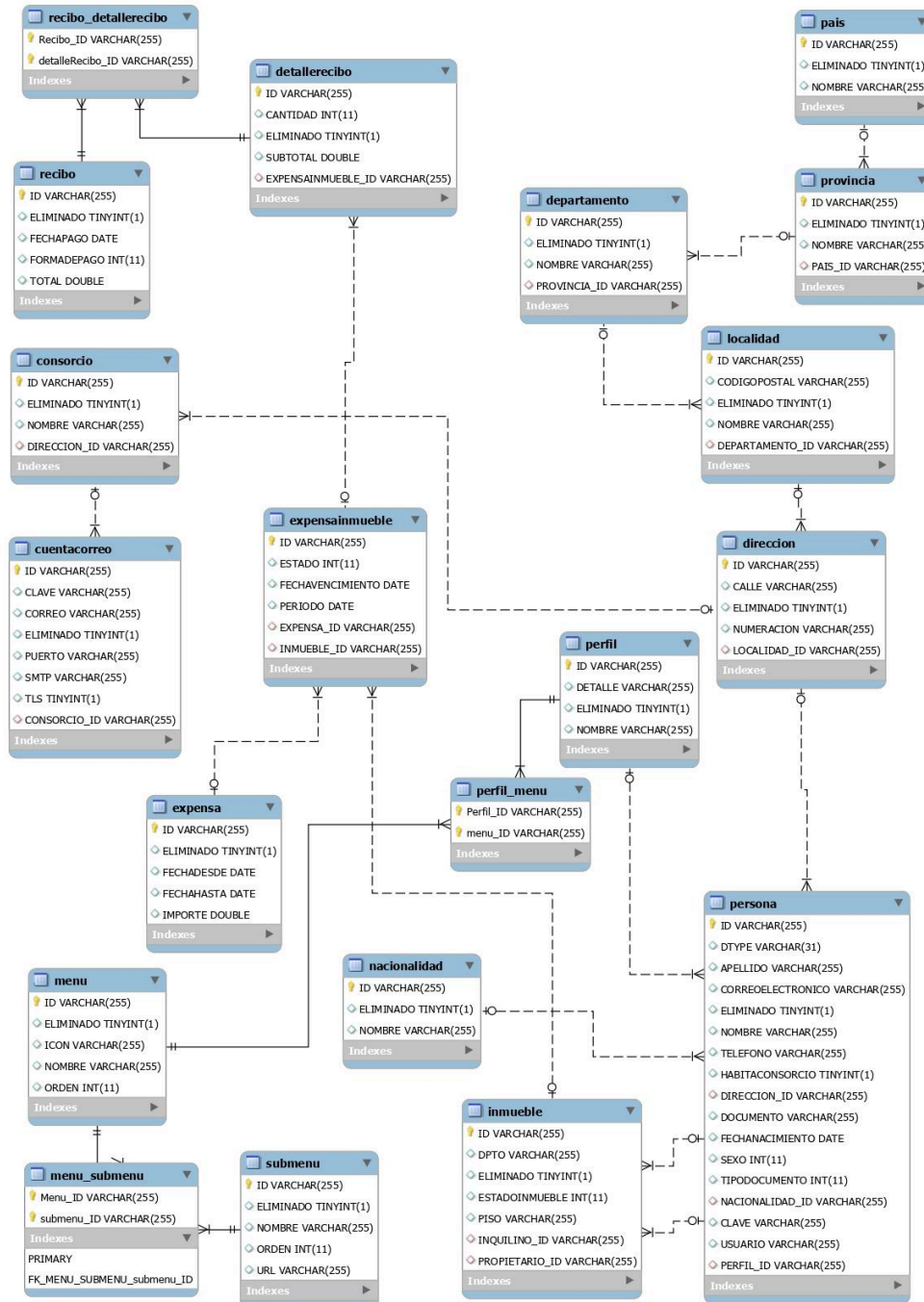
### **Diagrama de paquetes de diseño.**

El diagrama de paquetes de diseño también nos ayuda a observar la arquitectura interna del sistema y las dependencias entre paquetes. Además, ya que nosotros estuvimos trabajando en capas, nos permite observar de manera más visual esa separación existente entre ellas. Para la realización de este, no se colocaron los métodos o atributos ya que la visualización de estos puede darse en el diagrama de clases de diseño y se buscaba hacer énfasis en los paquetes. Nuevamente, por la gran cantidad de clases utilizadas (servicio, entidades, daos, etc), el diagrama de paquetes de diseño quedó con una gran extensión, por ese motivo se adjunta el link correspondiente:

<https://drive.google.com/file/d/1Zrh3ZYN1DGQu2xSF5YucWifjUdIksmon/view?usp=sharing>

## Diagrama Entidad Relación Base de Datos.

El diagrama de entidad relación nos permite observar la estructura lógica de la base de datos y las relaciones entre las diferentes bases de datos. Este diagrama es importante también ya que nos permite ver de una manera más visual como el ORM mapea los objetos creados en Java a entidades de la base de datos. El diagrama fue realizado utilizando la aplicación MySQL Workbench compatible con la versión de MySQL utilizada.



## **Construcción**

Esta etapa trata principalmente de la implementación de las distintas partes del sistema para lograr un incremento funcional del mismo. Es decir, esta etapa trata principalmente de codificar todo lo que se logró del análisis de los requisitos en las anteriores etapas. Recordemos que si bien la codificación pudo ser empezada mínimamente en la etapa de elaboración, es esta en la cual será el foco principal.

### **Implementación.**

#### **Framework de trabajo.**

El desarrollo se realizó utilizando el **IDE NetBeans 8.0.2**, en conjunto con **JDK 7u60** y **JRE 7u60**, lo que garantiza un entorno de desarrollo compatible y eficiente. NetBeans fue elegido por su capacidad para soportar proyectos basados en Java EE y su facilidad para gestionar bases de datos relacionales.

El **JDK 7u60** se utilizó para compilar y depurar el código fuente, mientras que el **JRE 7u60** permitió la ejecución estable de la aplicación en los entornos de producción, asegurando compatibilidad con la infraestructura existente.

Además, se emplearon **GitHub** y **Git** como herramientas de control de versiones, lo que facilitó la colaboración entre los desarrolladores. A través de GitHub, se gestionaron los cambios en el código, se realizó un seguimiento eficiente de las distintas versiones y se garantizaron buenas prácticas en la integración continua.

#### **Base de Datos.**

Para la gestión de datos de la aplicación, se utilizó **MySQL versión 5.0** como sistema de bases de datos e **Hibernate** como framework ORM. MySQL fue seleccionado por su rendimiento y escalabilidad, mientras que Hibernate facilitó la interacción entre la aplicación y la base de datos, al permitir manejar datos mediante objetos Java sin necesidad de escribir SQL manualmente.

#### **Codificación.**

Se codificó el proyecto utilizando el modelo vista controlador. Básicamente, se divide la lógica de la aplicación en 3 capas principales: el modelo, la vista y el controlador. Sin embargo, la capa de modelo se subdivide en 3 capas más: el dominio, el acceso a datos y la lógica del negocio. A la hora de interactuar con la aplicación los datos van saltando de capa en capa. Para implementarlo, se utilizó Java Enterprise Edition con JPA. El código elaborado por los integrantes del grupo puede ser observado en el repositorio de github utilizado:

<https://github.com/NTMConsorcio/ProyectoConsorcio>

## **Pruebas**

### **Pruebas unitarias.**

Se realizaron pruebas unitarias manuales a medida que se iba desarrollando cada pequeña funcionalidad de la aplicación. Básicamente se testeó cada función realizando una pequeña interfaz gráfica que sirviera para insertar y visualizar los datos.

### **Pruebas de interfaz.**

Se realizaron también pruebas de interfaz manuales. Es decir, luego de definir los casos de prueba, cada integrante del equipo realizó manualmente esos casos en la interfaz para comprobar que el resultado fuese el esperado. Además, se comprobó que la navegabilidad fuese cómoda para el usuario.

## **Gestión del Cambio y Configuración.**

Múltiples cambios fueron necesarios a la hora de realizar el proyecto. Estos fueron derivados por múltiples causas, principalmente la adquisición de nuevos conocimientos a lo largo del cursado, pero también derivado de la misma programación. Es decir, a la hora de programar nos encontramos con que la implementación del diagrama de clases podía ser realizado de otra manera diferente, por lo que tuvimos que realizar modificaciones. Otro motivo es la aparición de métodos no previstos para evitar malas prácticas, por ejemplo, cuando se realizaban las validaciones siempre eran iguales, por ello se tuvo que crear el método verificar() en cada clase, incorporando modificaciones a los diagramas. Por último, una gran fuente de cambios fue el cliente, quién fue incorporando o eliminando requisitos a lo largo del desarrollo, por lo que para tener en cuenta estos cambios se tuvo que modificar varios diagramas anteriormente realizados.

### **Modificación de artefactos**

A medida que se fueron realizando cambios en la implementación o en el diseño, por requerimientos cambiantes o por mejoras en la estructura, se fueron actualizando los diagramas realizados, especialmente el diagrama de clases y los de secuencia. Además, el código fue actualizado para ajustarse a estos cambios.

## **Transición**

Esta etapa trata básicamente de garantizar que el sistema esté listo para su implementación y uso real. Esto incluye validar el software y hacer la entrega en un entorno de producción. A causa de la falta de herramientas, no se pudo desarrollar completamente esta etapa, ya que, por ejemplo, no se tuvo un entorno de producción en el que fuese posible realizar pruebas o un entorno para realizar el despliegue del mismo. Sin embargo, para intentar suplir esto, se realizaron pruebas manuales en entornos de desarrollo locales.

## **3. Propuesta de mejora**

### **Referencia de mercado.**

En el mercado actual existen múltiples competidores con productos dirigidos al mismo nicho. Entre ellos, destaca APro Expensas, parte del Sistema APro, que está compuesto por un conjunto de soluciones orientadas a la administración de consorcios, barrios privados, clubes

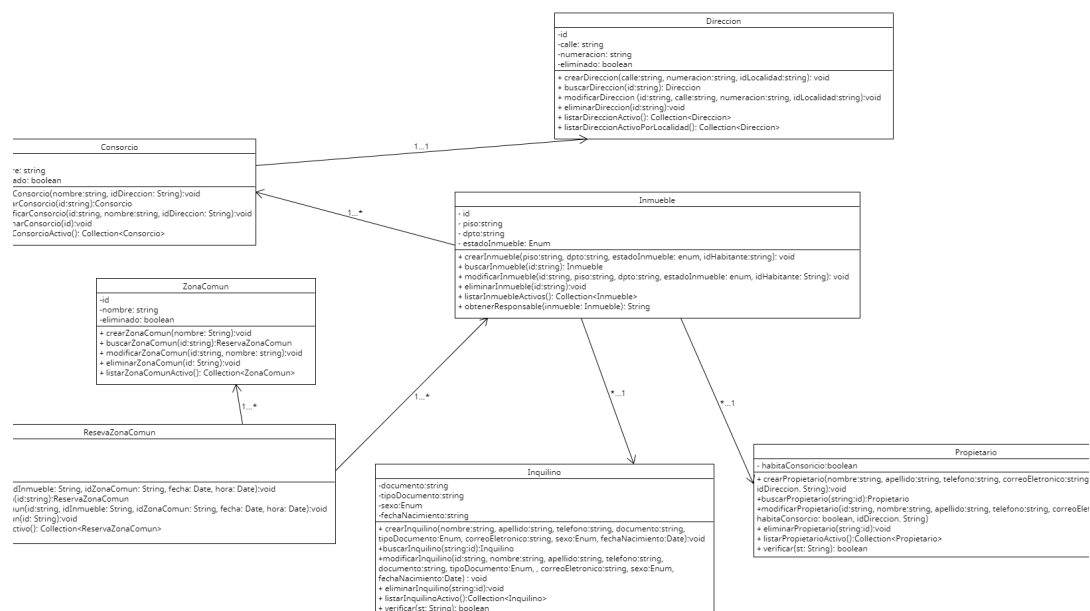
de campo y cualquier organización que requiera la distribución de gastos entre varios copropietarios o socios.

APro Expensas es el programa principal de este sistema, permitiendo la gestión integral de consorcios, incluyendo la administración de propietarios, proveedores y cuentas corrientes. Además, facilita la realización de liquidaciones y la generación de informes. Uno de los aspectos más valorados del sistema es su capacidad de personalización, permitiendo modificar plantillas, boletas y recibos, lo que otorga flexibilidad para adaptar el diseño a las necesidades específicas de cada consorcio.

### Propuesta de mejora.

La propuesta de mejora sugerida por el equipo es la posibilidad de gestión (reserva) de los espacios de uso compartido en el consorcio (Zoom de eventos, churrasqueras, entre otros). Se elaboró una modificación del diagrama de clases presentado para poder implementarla.

### Modificación del diagrama de clases



El diagrama dicta ciertas pautas a seguir en la implementación. Dado a que no existe una relación entre la zonaComun y el Consorcio, las zonas comunes serán validadas como que pertenecen al consorcio por la relación entre estas y los Inmuebles.

## 4. Propuesta de refactorización

### Vista

Para la vista la propuesta de refactorización sería innovar con nuevos elementos visuales para aumentar la estética de la aplicación. Actualmente el sistema cuenta con una vista agradable y tiene utilidad, sin embargo, sí existe la posibilidad de mejorar las características visuales de este.

### Controlador

Nuestra propuesta de refactorización para los controladores, especialmente para los de la vista de listas, incluye el desarrollo de un controlador genérico. Es decir, actualmente los

controladores hacen cosas prácticamente iguales, por lo que sería posible crear uno genérico que resolviera las funcionalidades de los actuales. Además, esto nos ahorraría una gran cantidad de código ya que nos solucionaría el problema de la repetición.

### **Modelo**

Nuevamente, nuestra propuesta sería la construcción de clases de servicio genéricas. Esto es debido a que la mayoría de las actuales realizan los mismos métodos (Crear, modificar, buscar, eliminar, listar), por lo que sería posible la generalización. De la misma manera que para el controlador, esto nos ahorraría mucho código y solucionaría el problema de la repetición.

### **Acceso a datos**

Para la capa de acceso a datos sucede lo mismo que para las dos anteriores. Es decir, los métodos de los DAOs se repiten en todos, por lo que nuestra propuesta sería hacer un solo DAO genérico con la posibilidad de crear nuevos solo para funcionalidades específicas. Un ejemplo de este último podría ser `buscarInmuebleExpensaPorInmuebleExpensa()`. Podemos ver que esta manera sería una forma mucho más eficiente y limpia de implementar el proyecto

## **5. Material de exposición en conferencia**

### **Porcentaje avance del software**

El software desarrollado hasta el momento de la presentación se encuentra con un porcentaje de avance de aproximadamente **80%** sin contar las extensiones del software que fueron dejadas pendientes para otras etapas. Para la fase actual del desarrollo del proyecto restaría mejorar la interfaz visual, mejorar el funcionamiento de los recibos y agregar la gestión de usuarios con los respectivos perfiles.

### **Powerpoint presentación**

Link al powerpoint utilizado durante la presentación:

## **6. Referencias**

1. Oracle. Técnica de Sun Java Enterprise System  
<https://docs.oracle.com/cd/E19528-01/820-0888/6ncjkpn6/index.html>
2. Primefaces.  
<https://www.primefaces.org/documentation/>
3. UMLetino.  
<https://www.umletino.com/>
4. Apuntes de la cátedra
5. Sistema APro  
<https://www.glsistemas.com.ar/>



## 7. Anexo

### 1. Presupuesto del sistema

<https://drive.google.com/file/d/13LdZwZcBalsGIH3aXb6ygV5K7RKm1Kp4/view?usp=sharing>

### 2. Diagrama de paquetes de diseño

<https://drive.google.com/file/d/1Zrh3ZYN1DGQu2xSF5YucWifjUdlksmon/view?usp=sharing>

### 3. Diagrama de clases de diseño

<https://drive.google.com/file/d/1sNDbdTXw3bsgitSeNrIBosnDWWs8-qXf/view?usp=sharing>

### 4. Diagrama de secuencia de diseño de notificar recibo

[https://drive.google.com/file/d/1PSA8-nMcsjDPJP5qtaVaGoSJX\\_3\\_3fPQ/view?usp=sharing](https://drive.google.com/file/d/1PSA8-nMcsjDPJP5qtaVaGoSJX_3_3fPQ/view?usp=sharing)

### 5. Diagrama de secuencia de diseño de notificar expensas

[https://drive.google.com/file/d/1JO7EVAw\\_uX9EkU6ea5fRnb66-DiJ5TXc/view?usp=sharing](https://drive.google.com/file/d/1JO7EVAw_uX9EkU6ea5fRnb66-DiJ5TXc/view?usp=sharing)