

UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

CARRERA DE SOFTWARE

Tema:

SISTEMA WEB Y MÓVIL PARA LA SISTEMATIZACIÓN DE LA GESTIÓN ADMINISTRATIVA Y DE ASAMBLEAS DEL CONJUNTO HABITACIONAL "EL PORTAL DE LA VIÑA"

Trabajo de titulación modalidad Proyecto de investigación, presentado previo a la obtención del título de Ingeniero de Software

ÁREA: Software

LÍNEA DE INVESTIGACION: Desarrollo de software

AUTOR: Alex Ignacio Tigselema Pacheco

TUTOR: Ing. Leonardo David Torres Valverde, Mg.

Ambato - Ecuador

agosto - 2024

APROBACIÓN DEL TUTOR

En calidad de tutor del trabajo de titulación con el tema: SISTEMA WEB Y MÓVIL PARA LA SISTEMATIZACIÓN DE LA GESTIÓN ADMINISTRATIVA Y DE ASAMBLEAS DEL CONJUNTO HABITACIONAL “EL PORTAL DE LA VIÑA”, desarrollado bajo la modalidad Proyecto de Investigación, por el señor Alex Ignacio Tigselema Pacheco, estudiante de la Carrera de Software, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que el estudiante ha sido tutorado durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 17 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.3 del instructivo del reglamento referido.

Ambato, agosto 2024.



Ing. Leonardo David Torres Valverde, Mg.

TUTOR

AUTORÍA

El presente trabajo de titulación con el tema: SISTEMA WEB Y MÓVIL PARA LA SISTEMATIZACIÓN DE LA GESTIÓN ADMINISTRATIVA Y DE ASAMBLEAS DEL CONJUNTO HABITACIONAL "EL PORTAL DE LA VIÑA" es absolutamente original, auténtico y personal y ha observado los preceptos establecidos en la Disposición General Quinta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, agosto 2024.



Alex Ignacio Tigselema Pacheco

C.C. 1803834371

AUTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato para que reproduzca total o parcialmente este trabajo de titulación dentro de las regulaciones legales e institucionales correspondientes. Además, cedo todos mis derechos de autor a favor de la institución con el propósito de su difusión pública, por lo tanto, autorizo su publicación en el repositorio virtual institucional como un documento disponible para la lectura y uso con fines académicos e investigativos de acuerdo con la Disposición General Cuarta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato.

Ambato, agosto 2024.



Alex Ignacio Tigselema Pacheco

C.C. 1803834371

AUTOR

APROBACIÓN DEL TRIBUNAL DE GRADO

En calidad de par calificador del informe final del trabajo de titulación presentado por el señor Alex Ignacio Tigselema Pacheco, estudiante de la Carrera de Software, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado SISTEMA WEB Y MÓVIL PARA LA SISTEMATIZACIÓN DE LA GESTIÓN ADMINISTRATIVA Y DE ASAMBLEAS DEL CONJUNTO HABITACIONAL "EL PORTAL DE LA VIÑA" nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 19 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.4 del instructivo del reglamento referido. Para cuya constancia suscribimos, conjuntamente con la señora Presidente del Tribunal.

Ambato, agosto 2024.

Firmado electrónicamente por:
ELSA PILAR URRUTIA URRUTIA

Ing. Elsa Pilar Urrutia Urrutia, Mg.
PRESIDENTE DEL TRIBUNAL

Firmado electrónicamente por:
EDISON HOMERO ALVAREZ MAYORGA

Ing. Edison Homero Alvarez Mayorga Mg.
PROFESOR CALIFICADOR

Firmado electrónicamente por:
EDWIN HERNANDO BUENANO VALENCIA

Ing. Edwin Hernando Buenaño Valencia Mg.
PROFESOR CALIFICADOR

DEDICATORIA

A mi madre Patricia por ser mi pilar fundamental en mi vida y el mótivo principal de este gran logro, a mi padre Ignacio por ser quien me apoyo en tiempos difíciles e indicarme como seguir adelante, a mi hermana Adriana por enseñarme a nunca rendirme, a mi pareja Paola por ser mi compañera de vida y apoyarme en todo momento, y a mis amigos por ser parte de mi etapa universitaria y brindarme su amistad y apoyo.

Alex Ignacio Tigselema Pacheco

AGRADECIMIENTOS

Quiero empezar agradeciendo a mi madre por haberme brindado la oportunidad de seguir con mis estudios. A toda mi familia por siempre ser un gran apoyo en momentos difíciles. A mi pareja por estar ahi cuando más la he necesitado y ser mi compañera. A mis amigos por esa gran amistad que ha conllevado a grandes momentos. A todos mis profesores por haberme brindado sus conocimientos y experiencias. Y a mi tutor por haberme guiado en este proceso de titulación.

Alex Ignacio Tigselema Pacheco

ÍNDICE GENERAL DE CONTENIDOS

PORTADA i

APROBACIÓN DEL TUTOR ii

AUTORÍA iii

DERECHOS DE AUTOR iv

APROBACIÓN DEL TRIBUNAL DE GRADO v

DEDICATORIA vi

AGRADECIMIENTO vii

ÍNDICE GENERAL DE CONTENIDOS x

ÍNDICE DE TABLAS xi

ÍNDICE DE FIGURAS xvi

ÍNDICE DE ANEXOS xvii

ABSTRACT xix

CAPÍTULO I. MARCO TEÓRICO 1

1.1 Tema de investigación 1

1.1.1 Planteamiento del problema 1

1.1.2 Antecedentes investigativos 2

1.2 Fundamentación teórica 6

1.2.1 Condominio 6

1.2.2 Gestión administrativa 6

1.2.3 Modelo de gestión 7

1.2.4 Ley de propiedad horizontal 7

1.2.5 Sistema web 7

1.2.6 Sistema móvil 8

1.2.7 API 8

1.2.8 Gestor de bases de datos 8
1.2.9 Lenguajes de programación 9
1.2.10 Framework 10
1.2.11 GIT 12
1.2.12 Diagramas de diseño 12
1.2.13 Patrones de arquitectura de software 13
1.2.14 Metodologías de desarrollo de software 13
1.2.15 Modelo de aceptación tecnológica (TAM) 14
1.2.16 Figma 14
1.3 Objetivo general 15
1.4 Objetivos específicos 15

CAPÍTULO II. METODOLOGÍA **16**

2.1 Materiales 16
2.2 Métodos 16
2.2.1 Modalidad de la investigación 16
2.2.2 Población y muestra 16
2.2.3 Recolección de información 17
2.2.4 Procesamiento y análisis de datos 29

CAPÍTULO III. RESULTADOS Y DISCUSIÓN **31**

3.1 Análisis de herramientas de desarrollo 31
3.1.1 Análisis y selección de frameworks 31
3.2 Definición de la metodología de desarrollo de software 36
3.2.1 Comparación entre metodologías tradicionales y metodologías ágiles . 36
3.3 Cálculo de la distancia Haversine 39
3.4 Análisis del proceso actual 40
3.5 Desarrollo de la propuesta 48
3.5.1 Planificación de requerimientos 49
3.5.2 Diseño de usuario 54
3.5.3 Construcción 91
3.5.4 Transición 144
3.6 Aplicación de los cuestionarios TAM 149
3.6.1 Modelo TAM 150

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES **155**
4.1 Conclusiones 155
4.2 Recomendaciones 156
REFERENCIAS BIBLIOGRÁFICAS **161**
ANEXOS **162**

ÍNDICE DE TABLAS

1 Entrevista a la directiva del conjunto habitacional “El Portal de la Viña” . 18
2 Resultados de la pregunta uno 21
3 Resultados de la pregunta dos 22
4 Resultados de la pregunta tres 23
5 Resultados de la pregunta cuatro 24
6 Resultados de la pregunta cinco 25
7 Resultados de la pregunta seis 26
8 Resultados de la pregunta siete 26
9 Resultados de la pregunta ocho 27
10 Resultados de la pregunta nueve 28
11 Características de los framework front-end Angular, React y Vue 31
12 Ventajas y desventajas de los framework front-end Angular, React y Vue . 32
13 Comparación de los frameworks: Métricas y puntuajes 33
14 Comparación entre gestores SQL, NoSQL y NewSQL 35
15 Comparación entre metodologías ágiles y metodologías tradicionales . . . 37
16 Comparación de las metodologías de desarrollo ágiles 38
17 Descripción de los usuarios identificados en la interacción con el sistema web administrativo 49
18 Descripción de los usuarios identificados en la interacción con la aplicación móvil 50
19 Identificación de los requerimientos del sistema 50
20 Plan de trabajo de iteraciones 53
21 Pruebas de aceptación - Primera iteración 144
22 Pruebas de aceptación - Segunda iteración 144
23 Pruebas de aceptación - Tercera iteración 145
24 Resultados PU/PEOU-TAM web 151
25 Resultados PU/PEOU-TAM móvil 153

ÍNDICE DE FIGURAS

1 Resultados de la encuesta en la pregunta uno 21
2 Resultados de la encuesta en la pregunta dos 22
3 Resultados de la encuesta en la pregunta tres 23
4 Resultados de la encuesta en la pregunta cuatro 24
5 Resultados de la encuesta en la pregunta cinco 25
6 Resultados de la encuesta en la pregunta seis 26
7 Resultados de la encuesta en la pregunta siete 27
8 Resultados de la encuesta en la pregunta ocho 28
9 Resultados de la encuesta en la pregunta nueve 29
10 Diagrama de flujo de procesos actual de administración de parqueaderos (Zona azul). 43
11 Diagrama de flujo de procesos actual de administración de convocatorias. 45
12 Diagrama de flujo de procesos actual del pago de obligaciones financieras. 46
13 Diagrama de flujo de procesos actual del pago de obligaciones financieras. 47
14 Diagrama de flujo de procesos actual de multas. 48
15 Diagrama de flujo de procesos sistematizado prupuesto de administración de parqueaderos (Zona azul). 55
16 Diagrama de flujo de procesos sistematizado prupuesto de convocatorias . 58
17 Diagrama de flujo de procesos sistematizado prupuesto de pago de obligaciones financieras mensuales 60
18 Diagrama de flujo de procesos sistematizado prupuesto de pago a proveedores 61
19 Diagrama de flujo de procesos sistematizado prupuesto de multas 62
20 Arquitectura del sistema 63
21 Prototipo de inicio de sesión 64
22 Prototipo de la vista del la barra de navegación 65
23 Prototipo de la vista del dashboard de administración 66
24 Prototipo de la vista del dashboard de presidencia 66
25 Prototipo de la vista del dashboard de tesorería 67
26 Prototipo de la vista de la administración de usuarios 68
27 Prototipo de la vista del fomulario para edición o creación de usuarios . . 69
28 Prototipo de la vista de la configuración de la geolocalización 70
29 Prototipo de la vista de administración de residencias 71

30 Prototipo de la vista de administración de parqueaderos 72
31 Prototipo de la vista de administración de tipos de parqueaderos 72
32 Prototipo de la vista del formulario de cambio de residencia asociada a un parqueadero 73
33 Prototipo de la vista del formulario en la selección de residencia 74
34 Prototipo de la vista de la administración de convocatorias 75
35 Prototipo de la vista del formulario de de edición o creación de convocatorias 75
36 Prototipo de la vista de vista previa a la descarga de la convocatoria . . . 76
37 Prototipo de la vista de administración de asistencias en asambleas 76
38 Prototipo de la vista de administración de votaciones en asambleas 77
39 Prototipo de la vista de administración de tipos de ingresos casuales y mensuales 78
40 Prototipo de la vista del formulario de creación y edición de tipos de ingresos casuales y mensuales 78
41 Prototipo de la vista de administración de tipos de multas 79
42 Prototipo de la vista de administración de ingresos mensuales 80
43 Prototipo de la vista del formulario de creación y edición de ingresos mensuales 80
44 Prototipo de la vista del recibo de cobro generado por el sistema 81
45 Prototipo de la vista de la administración de multas 82
46 Prototipo de la vista del formulario de creación y edición de ingresos multas 82
47 Prototipo de la vista de la administración de egresos 83
48 Prototipo de la vista de la administración de tipos de egresos 84
49 Prototipo de la vista de la administración de proveedores 84
50 Prototipo de la vista del formulario de creación y edición de egresos . . . 85
51 Prototipo de la vista del inicio de sesión 86
52 Prototipo de la vista de la barra de navegación 86
53 Prototipo de la vista de eventos próximos 87
54 Prototipo de la vista detallada de un evento 88
55 Prototipo de la vista de asambleas del día actual 89
56 Prototipo de la vista del registro de asistencia en asambleas 89
57 Prototipo de la vista del perfil 90
58 Prototipo de la vista del registro de voto en asambleas 90
59 Archivo *pom.xml* con las dependencias Maven de la API 91

60 Código de configuración de propiedades de la API 92
61 Código de configuración de seguridad de la API 93
62 Código de la clase abstracta base para las demas entidades AbstractEntity 95
63 Código de la entidad Income 96
64 Código de el repositorio para para entidad Income 97
65 Diagrama entidad-relación de la base de datos (Parte 1) 98
66 Diagrama entidad-relación de la base de datos (Parte 2) 99
67 Código de configuración del almacenamiento de archivos en S3 100
68 Código del servicio de carga y eliminación del archivo S3 101
69 Código de configuración del motor de plantillas HTML para los correos electrónicos 102
70 Código de modelo para mapearlo en la plantilla HTML 102
71 Código del método de envio de correo electrónico para recuperar la contraseña 102
72 Código de configuración de Firebase para el envío de Notificaciones Push 103
73 Código del servicio de las Notificaciones Push 104
74 Código del cálculo de la distancia de Haversine en Java 105
75 Código de método de registro de asistencia en la asamblea 105
76 Código del controlador de usuarios 106
77 Archivo *package.json* con las dependencias del sistema web 107
78 Archivo *app.config.ts* con la configuración de la aplicación 108
79 Archivo *app-routes.ts* con la configuración de las rutas de la aplicación . 109
80 Código del inicio de sesión en el frontend 110
81 Código del método de autenticación en el API 111
82 Página de inicio de sesión 111
83 Código del interceptor de peticiones HTTP en el sistema web 112
84 Endpoint para obtener todos los usuarios 113
85 Página de gestión de usuarios 113
86 Endpoint para crear un usuario 113
87 Formulario de creación y edición de usuarios 114
88 Método de actualización de la geolocalización en el API 114
89 Página de gestión de geolocalización 115
90 Endpoint para actualizar el usuario representante de una residencia 115
91 Formulario de actualización de usuario representante de una residencia . . 116

92 Método de actualización de la casa asociada a un parqueadero en la API . 116
93 Página de gestión de parqueaderos 117
94 Formulario de actualización de la casa asociada a un parqueadero 117
95 Endpoint para crear un incidente 118
96 Página de gestión de incidentes 118
97 Formulario de creación y edición de incidentes 118
98 Reporte de incidente 119
99 Método de creación de una convocatoria y envio de notificación push en la API 120
100 Página de gestión de convocatorias 120
101 Formulario de creación y edición de convocatorias 121
102 Vista previa a la descarga del comunidado de la convocatoria en PDF . . 121
103 Método del registro de asistencia manual en la API 122
104 Página de gestión de asistencia en asambleas 123
105 Reporte de inasistencias en la asamblea 123
106 Endpoint para crear una pregunta de asamblea 124
107 Endpoint para obtener todas las preguntas de asamblea 124
108 Endpoint para habilitar o deshabilitar una pregunta de asamblea 124
109 Lista de preguntas de asamblea 124
110 Listado detallado de votaciones de una pregunta de asamblea 125
111 Listado de los ingresos de tipo mensual 125
112 Endpoint para listar los ingresos mensuales 126
113 Formulario de creación y edición de ingresos mensuales 126
114 Endpoint para guardar un ingreso mensual 126
115 Listado de multas 127
116 Endpoint para listar las multas 127
117 Formulario de creación y edición de multas 128
118 Listado de los egresos 128
119 Endpoint para listar los egresos 129
120 Formulario de creación y edición de egresos 129
121 Endpoint para guardar un ingreso mensual 130
122 Dependencias de la application móvil hecha con Ionic 131
123 Configuración de la URL de la API en la aplicación móvil 132
124 Codigo del archivo de configuración principal de la aplicación móvil . . . 132

125 Codigo del servicio de Notificaciones Push en la aplicación móvil 133
126 Ubicación del archivo *google.services.json* 134
127 Código del archivo *variables.gradle* con la referencia al plugin de Firebase 134
128 Código del archivo *manifest.xml* con los permisos necesarios 135
129 Página de inicio de sesión 135
130 Método para obtener las convocatorias próximas 136
131 Página de las próximas asambleas 136
132 Detalle de una asamblea 137
133 Página de el estado de las obligaciones financieras 138
134 Endpoints para obtener las obligaciones financieras de un usuario 138
135 Página de los últimos pagos realizados por el usuario 139
136 Página de las multas sin pagar 139
137 Código para registrar la asistencia en la asamblea 141
138 Endpoint para registrar la asistencia en la asamblea 142
139 Página de asistencia en la asamblea 142
140 Método para registrar el voto en la asamblea 143
141 Página de las votaciones de la asamblea 143
142 Administrador de VPC en AWS 147
143 Administrador de EC2 en AWS 147
144 Código de copia de recursos y ejecución del API en EC2 147
145 Zonas hospedadas de Route 53 en AWS 148
146 Certificados SSL generados con certbot 148
147 Estado del servicio web Nginx en la instancia EC2 149
148 Archivo de configuración de Nginx en la instancia EC2 149
149 Resultados del PU de la encuesta TAM al sistema web 151
150 Resultados del PEOU de la encuesta TAM al sistema web 151
151 Resultados del PU de la encuesta TAM a la aplicación móvil 152
152 Resultados del PEOU de la encuesta TAM a la aplicación móvil 153

ÍNDICE DE ANEXOS

A Guía de entrevista a la directiva del conjunto habitacional “El Portal de la Viña” 162
B Encuesta de necesidades a los residentes del conjunto habitacional “El Portal de la Viña” 163
C Encuesta TAM del sistema web administrativo 165
D Encuesta TAM de la aplicación móvil 166
E Dependencias de la API 167

RESUMEN EJECUTIVO

El conjunto habitacional “El Portal de la Viña” consta de 303 residencias, lo cual dificulta el control de los procesos administrativos por parte de la directiva, dando lugar a desafíos en la gestión de los parqueaderos, convocatorias y en tesorería, ya que estos procesos se realizan de manera manual.

El presente proyecto implementa un sistema de gestión web junto con una aplicación móvil para sistematizar los procesos administrativos y financieros del conjunto habitacional. Además, se utiliza la geolocalización como mecanismo de verificación de la asistencia y las votaciones en las asambleas. Para ello, se emplea la fórmula de Haversine para calcular con precisión la distancia entre la ubicación del residente y el punto de la asamblea, asegurando así que solo los asistentes presentes puedan participar en las votaciones.

El sistema de gestión del conjunto habitacional incluye módulos para la administración de propietarios, residencias, pagos, convocatorias, guardianía y parqueaderos. Se emplearon tecnologías modernas como los frameworks Angular e Ionic para el cliente web y móvil, respectivamente. Para el desarrollo de la interfaz de programación de aplicaciones (API), se utilizó el framework Spring Boot junto con PostgreSQL como gestor de base de datos.

La validación del sistema implementado se realizó aplicando el modelo de aceptación de tecnología (TAM) a los residentes del conjunto habitacional, teniendo como resultado una aceptación positiva del sistema propuesto. Se espera que la implementación de este sistema mejore la eficiencia y la transparencia en la administración del conjunto habitacional.

Palabras clave: Sistematización, conjunto habitacional, Framework, geolocalización, Angular, Ionic, Spring boot.

ABSTRACT

The “El Portal de la Viña” housing complex consists of 303 residences, which makes it difficult for the board of directors to control the administrative processes, giving rise to challenges in the management of parking spaces, calls, and in treasury, since these processes are carried out manually.

This project implements a web management system along with a mobile application to systematize the administrative and financial processes of the housing complex. In addition, geolocation is used as a mechanism to verify attendance and voting at assemblies. To do this, the Haversine formula is used to accurately calculate the distance between the resident’s location and the assembly point, thus ensuring that only those present can participate in the votes.

The housing complex management system includes modules for the administration of owners, residences, payments, calls, guardianship, and parking. Modern technologies such as Angular and Ionic frameworks were used for the web and mobile client, respectively. For the development of the application programming interface (API), the Spring Boot framework was used along with PostgreSQL as a database manager.

The validation of the implemented system was carried out by applying the technology acceptance model (TAM) to the residents of the housing complex, resulting in a positive acceptance of the proposed system. The implementation of this system is expected to improve efficiency and transparency in the administration of the housing complex.

Keywords: Systematization, housing complex, Framework, geolocation, Angular, Ionic, Spring boot.

CAPÍTULO I. MARCO TEÓRICO

1.1 Tema de investigación

SISTEMA WEB Y MÓVIL PARA LA SISTEMATIZACIÓN DE LA GESTIÓN ADMINISTRATIVA Y DE ASAMBLEAS DEL CONJUNTO HABITACIONAL "EL PORTAL DE LA VIÑA"

1.1.1 Planteamiento del problema

En [1] el autor menciona que los avances tecnológicos han sido muy relevantes en la sociedad actual, debido a que han permitido la optimización de procesos y la automatización de tareas, lo cual ha facilitado a las empresas y organizaciones ser más eficientes y productivas. En el caso administrativo, las herramientas tecnológicas ofrecen soluciones automatizadas mediante sistemas computacionales que reemplazan a los sistemas manuales tradicionales.

En [2], menciona que Ambato según el censo del 2022 realizado por el Instituto Nacional de Estadística y Censos(INEC) se ha registrado un total de 153948 viviendas particulares y 234 viviendas colectivas con un promedio de 3.18 habitantes por vivienda. En estos casos, según [3] en su Artículo 11 de la Ley de Propiedad horizontal los copropietarios tienen derecho a establecer sus propias normativas y reglamentos internos con la finalidad de poder crear modelos de gestión internos. En este sentido la gestión de los condóminios o departamentos se realizan mediante sistemas manuales lo cual al ser conjuntos residenciales son muy ineficientes debido a la gran cantidad de viviendas que suelen tener. Por consiguiente se hace necesario la implementación de herramientas tecnológicas que permitan agilizar y automatizar los procesos administrativos y financieros de los condominios o departamentos

En [4], menciona que el conjunto Habitacional "El Portal de la Viña" pese a que cuenta con un reglamento de gestión, el cual se rige por la ley de propiedad horizontal, y con un modelo de gestión administrativo, carece de una herramienta tecnológica que automatice estos procesos. En este sentido la administración de la gerencia se realiza en gran medida mediante sistemas de hojas de cálculo en línea y físicos, lo cual dificulta

la gestión de los recursos del conjunto habitacional. En el caso de las asambleas donde se aprueban o rechazan normativas mediante votaciones es necesario tener un control de los asistentes el cual es realizado actualmente de manera escrita.

De lo anteriormente expuesto la siguiente investigación se realizará en el conjunto habitacional "El Portal de la Viña" ubicado en la ciudad de Ambato, provincia de Tungurahua, Ecuador en el periodo académico marzo 2024 - agosto 2024.

1.1.2 Antecedentes investigativos

Para el desarrollo de esta investigación es necesario realizar una exploración de trabajos previos relacionados con el problema en los cuales se debe identificar las similitudes y diferencias con el presente proyecto. En este sentido de la búsqueda realizada se obtuvieron los siguientes trabajos previos:

En [5] se desarrollaron herramientas tecnológicas web y móvil para la administración de un condominio ubicado en la ciudad de Quito llamado "Torrez de Aranjuez". En dicho estudio se tuvo como objetivo centralizar y optimizar la información para la generación de reportes y mejorar la comunicación entre los usuarios del condominio y los administradores. El sistema web fue desarrollado en el framework .Net usando el lenguaje de programación C# en conjunto con HyperText Markup Language(HTML), Cascading Style Sheets(CSS3) y Javascript. En cuanto a los procesos administrativos cubiertos por el sistema web se encuentran módulos de gestion de pagos, gestión de zonas comunales, reportes finacieros de estados de cuenta, generación de recibos de pago y asistencias de los trabajadores de mantenimiento del conjunto. Mientras que el sistema móvil fue desarrolado unicamente para dispositivos Android utilizando el lenguaje de programación Java y cuenta con los módulos de gestion de pagos y notificaciones. Como metodología para el desarrollo del sistema se utilizó Extreme Programming (XP) debido a la flexibilidad de dicha métodología para el manejo de roles dentro de el ciclo de vida del proyecto y se usó como gestor de bases de datos Microsfot SQL Server.

En [6] se implementó un sistema de gestión administrativa para el condominio "Puertas de Alcalá" cuyo objetivo fue crear una plataforma informativa para los residentes de manera que toda la información sea transparente centrada específicamente en el

control de pagos. En este sentido el sistema web fue desarrollado en Hypertext Preprocessor(PHP), JavaScript y SQL Server como base de datos sin el uso de ningun tipo de framework. El sistema unicamente cuenta con procesos de gestión de pagos y reportes de pagos. La metodología utilizada fue la metodología de desarrollo de software iterativa.

En [7] se propone un prototipo de modelado de software utilizando el protocolo de intercambio de información Simple Object Access Protocol (SOAP). En este estudio se tomó como población a 129 residentes a los cuales se les aplicó una encuesta inicial para determinar las necesidades a cumplir mínimas por un sistema de votaciones de asambleas comunales. En donde se pudo evidenciar que las asambleas tienen una duración excesiva y se tiene la necesidad de optar por votaciones electrónicas debido a que el sistema manual presenta deficiencias en términos de transparencia. En este sentido el prototipo propuesto sugiere que se implementen roles a los usuarios para el manejo de seguridad, un módulo para la gestión de contabilidad, un módulo para la gestion legal, un módulo para la gestión de las asambleas y un módulo para la administración de los bienes comunales. La métodología utilizada para este caso de estudio fue Rational Unified Process(RUP).

En [8] se desarrolló un sistema web para la gestión de presupuestos en el Edificio Condominio Aquamar en Peru ubicado en el distrito La perla, Callao. En dicho estudio de tipo experimental se contó con una poblacion de 204 residentes y se utilizó un enfoque cuantitativo para la recolección de datos. El objetivo principal del estudio fue determinar la influencia del sistema web a los procesos de gestion del presupuesto. Teniendo en cuenta los indicadores de indice de ingresos de montos corrientes, liquidez y tiempo promedio de respuesta, con los cuales se busco analizar los efectos del sistema en los procesos. En este sentido el sistema propuesto fue desarrollado en PHP y MySQL sin el uso de ningún framework de desarrollo web, y se utilizó la metodología RUP. El sistema unicamente cuenta con módulos relacionados a la gestion econónomica más no a otros procesos administratrativos tales como incidentes, buzón de sugerencia, zonas comunales, parqueaderos y asambleas. Los resultados obtenidos fueron que el sistema web propuesto permitió mejorar los procesos de gestión del presupuesto en el Edificio Condominio Aquamar.

En [9] se desarrollaron sistemas para el control de asistencias de personal a través de reconocimiento facial y geolocalización. En este estudio se desarrollo una aplicación

móvil llamada SICAP cuya finalidad es la de registrar la asistencia de los empleados mediante el reconocimiento facil y su ubicación en tiempo real en un area de 50 metros de cualquiera de las areas de trabajo designadas por la empresa Agro Rural. Cabe mencionar que esta aplicación no requiere de internet para su uso puesto que utiliza un paradagima de programación reactiva lo cual permite que la aplicación envie sus peticiones cuando el dispositivo tenga acceso a internet. La Interfaz de Programación de Aplicaciones (API) desarrollada en este estudio permite exponer el servicio de asistencia a cualquier otro sistema desarrollado por la empresa Agro Rural. Por otro lado la aplicación web tiene la finalidad de la visualización de las asistencias por parte de los empleados, gestionar las incidencias laborales y generar reportes. Para el desarrollo de la aplicación móvil se utilizó el lenguaje de programación Kotlin junto con las librerias RxJava, EventBus y Retrofit. Mientras que para la aplicacion web se utilizo el framework de desarrollo Angular y para la API se utilizo el framework de JavaScript Express.

En [10] se desarrolló un sistema web alojado en la nube de Microsoft Azure para mejorar el control de los procesos administrativos y financieros del condominio “Solar del Rio”.En dicho estudio se contó con una población de 82 propietarios de viviendas. Para el desarrollo del sistema se utilizó el framework .Net Core y SQL Server como gestor de bases de datos. En cuanto a la metodología de desarollo se utilizó SCRUM la cual permitión tener un mejor control de los procesos a seguir durante el desarollo del sistema. Adicionalmente se aplicó la norma ISO/IEC 25010 para la evaluación de la calidad del sistema. El sistema cuenta con módulos de gestión de pagos, gestión de parqueaderos, impresión de recivos de pago y reportes de pagos. Para la validación del sistema se utilizó el Cuestionario de usabilidad de sistemas informáticos (CSUQ) y el métodos estadisticos de 2 variables para obtener la relación entre las preguntas del la encuesta.

En [11] se desarrolló un prototipo de software para la gestión de votaciones en las asambleas de condominios. En este estudio el prototipo desarrollado cuenta con un módulo de seguridad para el registro de usuarios y permisos. Ademas de el módulo de votaciones en el cual se pueden colocar observaciones adicionales al voto. Para el desarrollo de este prototipo se utilizo el framewwork de JavaScript Node.js junto con el motor de bases de datos SQL Server. La métodología utilizada fue Xtreme Programming (XP). Para la toma de información se utilizaron dos cuestionarios en

10 conjuntos residenciales, el primero a los usuarios residentes y el segundo a los administradores de los condominios. En dichos cuestionarios se evidenció la necesidad de tener un sistema informático para realizar las votaciones en asambleas comunales.

En [12] se implementó un sistema web para optimizar la gestión de actividades y eventos en el condominio "Los Jardines". En este estudio se tuvo una poblacion de 266 residentes. El sistema cuenta con las funciones de poder registrar eventos o actividades comunales, con un módulo de buzon de sugerencias que a su vez funciona como un foro de objetos perdidos y un módulo de reservas de espacios comunales. El sistema no cuenta con funcionalidades relacionadas con la gestion financiera de un conjunto residencial. Para desarrollar este sistema web se utilizó PHP, JavaScript, CSS, HTML y Bootstrap. La metodología utilizada fue la metodología de desarrollo de software en cascada y como gestor de bases de datos se uso MYSQL.

En [13] se desarrolló un sistema de información web para el condominio "jardines de Aramburú" ubicado en Perú. En dicho estudio se tuvo como poblacion a 550 propietarios, de los cuales se tomaron dos muestras. La primera muestra fueroin los 3 miembros de la junta de propietarios y la segunda 100 propietarios seleccionados de manera aleatoria. La investigación se centro en los procesos de gestión administrativa tales como pagos, mantenimientos de zonas comunales, y reportes financieros. para el levantamiento de información se aplicaron encuetas a los 100 propietarios lo cual les permitió medir de forma cuantitativa y estadisticamente. El sistema web se desarrollo en PHP y MySQL sin el uso de ningún framework de desarrollo web.

En [14] se implementó un sistema para mejorar los procesos de asistencia mediante la aplicación de geolocalización y reconocimiento facial. Para este estudio de investigación se utilizó un enfoque cuantitativo de diseño pre-experimental. En el sistema se implemento un modulo de administración de empleados en donde se registra la información personal y una foto del empleado para validar el reconocimiento facial. Para validar la asistencia tambien se colocan las coordenadas geograficas de las sucursales de la empresa en donde mediante el uso del Sistema de Posicionamiento Global(GPS) del dispositivo móvil se valida la asistencia del empleado. El sistema utiliza la herramienta de desarrollador de Google Maps API.

En [15] se desarrolló una aplicación web y móvil para el registro de asistencia con geolocalización de los empleados de la empresa proveedora de Internet SISCOM en

Latacunga. En este estudio se utilizaron encuestas y entrevistas para el levantamiento de información e identificar las necesidades. En este sentido para la aplicación web se utilizó Laravel 7 como framework de desarrollo. Mientras que para la aplicación móvil se utilizó el framework de desarrollo IONIC basado en Angula. Las métodolgías empleadas son XP y Mobil-D respectivamente. Como resultado de este proyecto los autores indican que se pudo optimizar el control de asistencia de los empleados asi como la veracidad de la misma.

En [16] se desarrolló un software multiplataforma para la optimización del modelo de gestión que posee el condominio "Conjunto habitacional oriental". En este estudio se demostró que el sistema con un nivel de confianza del 95% y un margen de error del 5% optimizó el modelo de gestión. En este proyecto se utilizó una base de datos PostgreSQL junto con los frameworks Spring Boot, React.js y React Native. La capa de negocio se enmarcó en la arquitectura REST. Por último la metodlogía utilizada fue SCRUM.

1.2 Fundamentación teórica

1.2.1 Condominio

Es un conjunto de viviendas que comparten áreas comunes y servicios. En [5,6,10,16], los autores mencionan que un condominio o copropiedad es un lugar, ya sea un conjunto de viviendas o terrenos, donde conviven propiedades compartidas por todos los dueños y propiedades exclusivas de cada propietario. En [17], los autores mencionan que un condominio es un desarrollo inmobiliario caracterizado por estar conformado por varios edificios o unidades de vivienda.

1.2.2 Gestión administrativa

Es el proceso de planificación, organización, dirección y control de los recursos de una organización con el fin de alcanzar sus objetivos. En [8,13,16], los autores mencionan que es necesario desarrollar una práctica de gestión democrática y participativa para la toma de decisiones en las asambleas de condominios.

1.2.3 Modelo de gestión

Es un conjunto de procesos que permiten la gestión de una organización. En [7, 8, 16], los autores mencionan que un modelo de gestión permite imitar o reproducir un arquetipo de gestión que ha sido exitoso en otras organizaciones.

1.2.4 Ley de propiedad horizontal

Es una ley que regula la convivencia de los propietarios de un condominio. En [3],su artículo 11 menciona "El Reglamento General de esta Ley establecerá un capítulo especial para precisar los derechos y obligaciones recíprocos de los copropietarios. Los propietarios de los diversos pisos, departamentos o locales, podrán constituir una sociedad que tenga a su cargo la administración de los mismos. Si no lo hicieren, deberán dictar un reglamento interno acorde con el Reglamento General". Su artículo 12 menciona "El Reglamento Interno de Copropiedad contendrá las normas sobre administración y conservación de los bienes comunes, funciones que correspondan a la Asamblea de los Copropietarios, facultades y obligaciones y forma de elección del administrador, distribución de las cuotas de administración entre los copropietarios y todo lo que converge a los intereses de los copropietarios y al mantenimiento y conservación del edificio."

1.2.5 Sistema web

Es un sistema de información que se encuentra alojado en un servidor web y que puede ser accedido mediante un navegador. En [5,9–11], los autores indican que la utilización de un sistema web permite gestionar más rápido la información y procesos de una organización residencial. En [11, 14, 16, 17], los autores mencionan que los sistemas web poseen la característica de poder manipular información desde cualquier parte del mundo, siempre y cuando se tenga acceso a internet.

- ***HTML5.*** Es un lenguaje de marcado de hipertexto, el cual define los contenidos que formarán parte de la página web. En [15], los autores mencionan que este lenguaje es útil para describir sintácticamente el contenido de una página web.

- ***CSS3.*** Es un lenguaje de hojas de estilo en cascada, el cual permite definir la presentación de los documentos HTML. En [15], los autores mencionan que es fundamental estilar el diseño acorde al prototipo diseñado por lo cual es necesario el uso de este lenguaje.

1.2.6 Sistema móvil

Es un sistema de información que se encuentra alojado en un dispositivo móvil. En [5, 9, 15], los autores mencionan que un sistema móvil tiene características similares a un sistema web con la ventaja de la portabilidad que ofrecen los dispositivos móviles.

Android

Es un sistema operativo de código abierto para dispositivos móviles, el cual es desarrollado por Google. En [5], los autores señalan que según estadísticas, este sistema operativo es el más utilizado a nivel nacional en un 82.2%.

1.2.7 API

Es una interfaz de programación de aplicaciones, la cual permite la comunicación entre dos aplicaciones de software. En [9, 16], los autores menciona que la implementación de una API permite la comunicación entre el sistema web y el sistema móvil.

Google Maps API

Es una API de Google que permite la integración de mapas en aplicaciones web y móviles. En [14, 15], los autores mencionan que esta API permite localizar geográficamente a los usuarios de una aplicación web o móvil.

1.2.8 Gestor de bases de datos

Un gestor de bases de datos es una aplicación o software diseñado para crear, administrar, manipular y gestionar datos en una base de datos.

Mysql es un sistema de gestión de bases de datos relacional mantenido por Oracle. En [8], los autores mencionan que este gestor de base de datos trabaja perfectamente bien con sistemas web y es fácil de implementar. En [15], los autores indican que

este gestor de base de datos posee una herramienta visual fácil de utilizar en la cual se pueden crear tablas, relaciones y consultas.

SQL Server es un sistema de gestión de bases de datos relacional desarrollado por Microsoft. En [5], los autores mencionan que este motor de bases de datos fue seleccionado debido a que es apto para dar soluciones de comercio electrónico y a la facilidad de implementación. PostgreSQL es un sistema de gestión de bases de datos relacional de código abierto. En [16], los autores indican que este gestor es compatible con el estándar SQL y posee las características de tener control de concurrencia e integridad transaccional.

1.2.9 Lenguajes de programación

Un lenguaje de programación es un lenguaje formal que permite a un programador especificar de manera precisa sobre un conjunto de instrucciones para que una computadora pueda producir un resultado.

PHP

Es un lenguaje de programación de código abierto, el cual es utilizado para el desarrollo de aplicaciones web. En [8, 15], el autor señala que este lenguaje tiene la facilidad de poder incrustar código PHP en código HTML, lo cual permite que el desarrollo de aplicaciones web sea más rápido y sencillo. En [5, 15], los autores mencionan que este lenguaje es muy utilizado en el desarrollo de software, dado que es muy robusto.

C#

Es un lenguaje de programación basado en objetos desarrollado por Microsoft. En [5], los autores mencionan que este lenguaje es uno de los más utilizados, debido a que tiene la característica de poder crear sistemas multiplataforma.

Java

Es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems. En [5, 16], los autores mencionan que este lenguaje es muy utilizado en el desarrollo de aplicaciones web y móviles, puesto que es multiplataforma.

1.2.10 Framework

Un framework es un conjunto de librerías y herramientas que permiten el desarrollo de aplicaciones de manera más rápida y sencilla.

CodeIgniter

Es un framework de PHP utilizado para el desarrollo web óptimo para aplicaciones que se ejecutan en un hosting compartido y configurados de manera diferente. En [15] los autores mencionan que este marco de desarrollo agiliza el proceso de desarrollo de aplicaciones web.

ASP.NET

Es un framework de código abierto para el desarrollo de aplicaciones web, el cual es desarrollado por Microsoft. En [5, 10], los autores mencionan que este framework permite el desarrollo de aplicaciones web de forma rápida y sencilla, debido a que cuenta con una gran cantidad de librerías y componentes que facilitan el desarrollo de aplicaciones web.

Express

Es un framework de código abierto para el desarrollo de aplicaciones web, el cual es desarrollado por la comunidad de Node.js. En [9], el autor menciona que este framework brinda un conjunto de características para las aplicaciones web y móviles la cual destacan su función como middleware.

Angular

Es un framework de desarrollo de aplicaciones web creado por Google que permite construir aplicaciones robustas y dinámicas del lado del cliente utilizando TypeScript y una arquitectura basada en componentes. En [9], el autor indica que este framework brinda una alta velocidad y buen rendimiento debido a que convierte las plantillas HTML en código optimizado para JavaScript. En [15], mencionan que este framework permite la creación de aplicaciones móviles, ya que posee un lenguaje adaptable.

React Js

React.js es una biblioteca de JavaScript utilizada para construir interfaces de usuario interactivas y dinámicas. Desarrollada por Facebook, se centra en la creación de componentes reutilizables que representan diferentes partes de la interfaz de usuario.

En [16], los autores mencionan que esta librería cuenta con una gran cantidad de librerías construidas por la comunidad, las cuales permiten la creación de aplicaciones web y móviles.

React native

Es un framework de desarrollo de aplicaciones móviles que utiliza JavaScript y React para crear aplicaciones nativas para iOS y Android con un único código base. En [16], los autores mencionan que esta herramienta permite la creación de aplicaciones nativas sin comprometer la experiencia del usuario debido a que brinda un set básico de componentes visuales.

Ionic

Es un framework de desarrollo de aplicaciones móviles híbridas que utiliza tecnologías web como HTML, CSS y JavaScript para crear aplicaciones multiplataforma. En [15], los autores indican que esta herramienta optimiza el consumo de recursos para la funcionalidad de una aplicación móvil.

Spring Boot

Es un marco de trabajo o framework de desarrollo de aplicaciones en Java que facilita la creación de aplicaciones robustas de manera rápida y sencilla. En [16], los autores mencionan que este marco de desarrollo está basado en el modelo MVC, el cual permite el desarrollo y despliegue de servicios REST.

Retrofit

Es una cliente de servicios REST para Android y Java, el cual es desarrollado por Square. En [9], el autor menciona que esta librería permite realizar peticiones HTTP, gestionar los parámetros y transformar la respuesta en un objeto Java.

RxJava

Es una librería de Java para la programación reactiva basada mediante el uso de observables. En [9], el autor menciona que esta biblioteca brinda una alternativa al uso de Thread y AsyncTask, dado que permite la ejecución de tareas en segundo plano de forma sencilla y asíncronas, además tiene una integración óptima con Retrofit.

Bootstrap

Es una librería de código abierto para el desarrollo de aplicaciones web, la cual permite el desarrollo de aplicaciones web responsivas. En [6, 12], los autores recalcan que esta

librería ofrece componentes que nos permíten interactuar con el usuario al instante, debido a que se encuentran predefinidos.

1.2.11 GIT

Es un sistema de control de versiones distribuido de código abierto, el cual permite el desarrollo de software de forma colaborativa. En [12], el autor menciona que este sistema de control de versiones permite guardar el estado de los archivos de un proyecto en repositorio.

1.2.12 Diagramas de diseño

Un diagrama de diseño es una representación gráfica de un sistema de información, el cual permite visualizar los componentes de un sistema y sus relaciones.

Diagrama de clases

Es un diagrama de estructura estática orientado completamente a la programación orientada a objetos. En [9], los autores mencionan que este diagrama representa la estructura de un sistema, mostrando las clases del sistema, sus atributos y sus relaciones.

ADM-Archimate

Es un lenguaje de modelado de arquitectura empresarial, el cual permite la representación de la arquitectura de una organización. En [7], los autores mencionan que este lenguaje de modelado ofrece los conceptos suficientes para poder modelar una arquitectura empresarial. Sin embargo tambíen indican que no ofrece ninguna metodología para el desarrollo de software.

Ninja Mockup

Es una herramienta de diseño de prototipos de software en línea. En [12], el autor menciona que esta herramienta permite la creación de prototipos de software de forma rápida y sencilla, debido a que cuenta con una gran cantidad de componentes predefinidos.

1.2.13 Patrones de arquitectura de software

Un patrón de arquitectura de software es un patrón de diseño que permite la creación de una arquitectura de software.

Patrón Modelo-Vista-Controlador(MVC)
Es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones. En [9], el autor indica que esta herramienta permite mostrar el acceso a los recursos del sistema gráficamente.

Patrón Model-View-ViewModel(MVVM)
Es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones. En [9], el autor menciona que este patrón de arquitectura es uno de los mejores para el desarrollo móvil debido a que nos permite separar de forma limpia la lógica de presentación y la lógica de negocio.

Representational State Transfer(REST)
Es un estilo de arquitectura de software para sistemas hipermedia distribuidos. En [16], los autores mencionan que este estilo de arquitectura permite la comunicación entre sistemas de información de forma sencilla y rápida gracias a los verbos HTTP GET, POST, PUT y DELETE.

1.2.14 Metodologías de desarrollo de software

Una metodología de desarrollo de software es un conjunto de pasos, procedimientos, técnicas y herramientas que permiten el desarrollo de software de forma eficiente.

Metodología RUP
Es una metodología de desarrollo de software iterativa e incremental, la cual tiene como objetivo asegurar la producción de software de alta y mayor calidad. En [8], los autores mencionan que esta metodología propone mayor interacción con el cliente, por lo tanto, se puede tener un mayor control sobre el proyecto de forma general.

Metodología XP

Es una metodología de desarrollo de software ágil, la cual tiene como objetivo principal la satisfacción del cliente mediante la entrega temprana y continua de software. En [5], los autores mencionan que se implementó esta metodología debido al desarrollo rápido en términos de tiempo y ajustes de requisitos a lo largo del desarrollo del mismo que se puedan llevar a cabo.

Metodología SCRUM

Es una metodología ágil que ayuda a los equipos a estructurar y gestionar el trabajo mediante un conjunto de valores, principios y prácticas En [10], el autor que esta metodología permite tratar de mejor maneras situaciones imprevisibles y resolver problemas complejos adaptándose a los cambios que se puedan presentar durante el desarrollo del proyecto. En [14], indican que la metodología SCRUM proporciona una vision global del proyecto a desarrollar mediante un cronograma de actividades.

1.2.15 Modelo de aceptación tecnológica (TAM)

Este modelo pretende determinar si una tecnología será aceptada o no por los usuarios basandose en los supuestos de que la percepción de utilidad y facilidad de uso de una tecnología determinan la actitud de un usuario hacia el uso de dicha tecnología. En [18], los autores mencionan que este modelo se debe evaluar en dos dimensiones: la percepción de utilidad y la percepción de facilidad de uso. La percepción de utilidad se refiere al grado en que una persona cree que el uso de una tecnología en particular mejorará su desempeño laboral. La percepción de facilidad de uso se refiere al grado en que una persona cree que el uso de una tecnología en particular será libre de esfuerzo.

1.2.16 Figma

Es una herramienta de diseño de interfaces de usuario basada en la nube. En [19], los autores mencionan que esta herramienta permite el diseño de interfaces de usuario de forma colaborativa, lo cual permite que varias personas puedan trabajar en un mismo proyecto de forma simultanea. Además, esta herramienta permite la creación de prototipos de software de forma rápida y sencilla, debido a que cuenta con una gran cantidad de plugins.

De todos los artículos y tesis revisados previamente, se destaca el uso de PHP en la creación de sistemas web junto con la librería de componentes Bootstrap, mientras que para el desarrollo de aplicaciones móviles se tiene como preferencía el desarrollo para sistemas operativos Android mediante el uso de Kotlin como lenguaje de programación. En cuanto a la georreferenciación es indiscutible el uso de GoogleMaps API para obtencion de la referencia geográfica y el uso de los servicios del mismo. En arquitecturas de software destaca MVC debído a que muchos frameworks y libreiras lo implementan nativamente. Por último la métodología de desarrollo de software más utilizada es SCRUM debido a la flexibilidad que ofrece para el desarrollo de proyectos de software.

1.3 Objetivo general

Implantar un sistema web y móvil mediante frameworks de desarrollo para la sistematización de la gestión administrativa y asambleas del conjunto habitacional “El Portal de la Viña”, analizando la satisfacción general del usuario.

1.4 Objetivos específicos

- Analizar frameworks para el desarrollo web y móvil considerando sus capacidades específicas para la sistematización de la gestión administrativa y asambleas.
- Desarrollar un sistema web y móvil para la gestión administrativa y asambleas con registro de asistencia usando geolocalización.
- Evaluar el sistema web y móvil, utilizando pruebas específicas de usabilidad para determinar la experiencia del usuario en su interacción con el sistema.

CAPÍTULO II. METODOLOGÍA

2.1 Materiales

Para la recolección de información se utilizarán encuestas y entrevistas mediante el uso de cuestionarios. Las encuestas serán aplicadas a los residentes del conjunto habitacional "El Portal de la Viña" para determinar necesidades (Véase Anexo B) y posteriormente medir la usabilidad de los sistemas informáticos propuestos en la presente investigación (Véase Anexos C y D). y las entrevistas serán aplicadas a los miembros de la gerencia del conjunto habitacional "El Portal de la Viña" para obtener información técnica(Véase Anexo A).

2.2 Métodos

2.2.1 Modalidad de la investigación

En la siguiente sección se detalla la modalidad de la investigación que se aplicará al desarrollo de este proyecto.

Investigación bibliográfica

Debido a que se requiere de una exploración previa de investigaciones relacionadas para poder identificar las similitudes y diferencias con el presente proyecto, se utilizará la investigación bibliográfica.

Investigación cuantitativa

Debido a que se aplicarán encuestas para la recolección de datos que posteriormente serán analizados mediante métodos estadísticos, se utilizará la investigación cuantitativa.

2.2.2 Población y muestra

Para el sistema web administrativo se considerará como población a los 4 miembros de la directiva del conjunto habitacional "El Portal de la Viña".

Para el sistema móvil se considerará como población a un solo propietario de cada residencia del conjunto habitacional “El Portal de la Viña”, teniendo un total de 303 propietarios. Del cual se tomará la siguiente muestra:

Muestra

- n = tamaño de la muestra.
- N = tamaño de la población (303).
- Z = nivel de confianza (1.96).
- d = desviación estándar (0.5).
- e = error de estimación máximo aceptable (0.09).

$$n = \frac{N * Z^2 * d * (1 - d)}{e^2 * (N - 1) + Z^2 * d * (1 - d)} \tag{1}$$

$$n = \frac{303 * 1.96^2 * 0.5 * (1 - 0.5)}{0.09^2 * (303 - 1) + 1.96^2 * 0.5 * (1 - 0.5)} \tag{2}$$

$$n = 85.22 \tag{3}$$

De las 303 casas con un nivel de confianza del 95% y un error de estimación del 9% se obtiene una muestra de 85 casas.

2.2.3 Recolección de información

La información se recolectó mediante el uso de encuestas y entrevistas. Las encuestas fueron aplicadas a los propietarios del conjunto habitacional “El Portal de la Viña” para determinar necesidades y posteriormente medir la usabilidad de los sistemas informáticos propuestos en la presente investigación. Las entrevistas fueron aplicadas a los miembros de la directiva del conjunto habitacional “El Portal de la Viña” para obtener información técnica y legal del conjunto habitacional.

Las siguientes respuestas son interpretaciones de las entrevistas realizadas a los miembros de la directiva del conjunto habitacional “El Portal de la Viña” por lo cual no son transcripciones literales de las respuestas dadas por los entrevistados.

Tabla 1. Entrevista a la directiva del conjunto habitacional “El Portal de la Viña”

N. Pregunta	Respuesta	Observación
1.	Las funciones de la directiva son mayormente administrativas y económicas en donde la presidencia(presidente y vicepresidente) se encargan de gestionar las multas, los eventos sociales y de asambleas, extender comunicados, gestionar las normativas y gestionar los parqueaderos. La tesorería se encarga de gestionar los ingresos y egresos del conjunto mediante el uso de hojas de cálculo como Excel, en donde los ingresos están dados por los pagos que se realizan por parte de los propietarios tales como las alícuotas, los parqueaderos y las multas. Los egresos, por otro lado, están dados por los pagos de los servicios que se tiene contratado para el condominio tales como la limpieza, jardinería, guardias y otros servicios adicionales. También se encarga de extender los recibos de pagos los cuales pueden ser pagados en efectivo o mediante una transferencia bancaria. Además, en tesorería se lleva el control de los horarios de cortes de jardín y de limpieza de áreas comunes. En secretaría se encarga de elaborar las actas de asamblea, informes de asistencia en las asambleas e invitaciones a los eventos sociales.	Se evidencia una gran carga de trabajo en tesorería, ya que es únicamente una persona la que debe atender a las 303 casas. También el hecho de tener todos los registros de ingresos y egresos en hojas de cálculo son un problema para la directiva, porque dificulta la búsqueda de information y la generación de reportes. Adicionalmente, el presidente para poder gestionar los parqueaderos se basa en un croquis de los mismos para poder identificarlos, lo cual es un proceso lento y tedioso.

2.	El presidente tiene como función extra realizar mínimo una ronda de revisiones de las áreas comunes del conjunto para encontrar fallas y poder realizar mejoras. La tesorera tiene como iniciativa estar al pendiente de las novedades que ocurran en guardianía. La secretaria tuvo como iniciativa realizar un seguimiento de las actividades diarias que realizan la directiva.	Estas funciones adicionales serán agregadas al sistema para que se introduzcan en el modelo de gestión del conjunto habitacional.
3.	La presidencia entrega certificados de no adeudamiento a los propietarios que lo soliciten, comunicados de eventos y asambleas y notificaciones de multas. La tesorería entrega los recibos de pagos, reporte de cartera de los parqueaderos de zona azul y emisión de listados de suspension de corte de jardín. La secretaría entrega las actas de asamblea y los informes de asistencia.	Se evidenció la necesidad de la directiva de poder digitalizar y generar gran parte de los documentos mencionados
4.	Presentar el físico cédula y papeleta de votación junto con sus respetivas copias, Predio o escrituras de la casa y firmar una acta de compromiso.	Este proceso al depender de documentos físicos los cuales no pueden ser enviados por medios digitales se mantendrán de la misma forma, sin embargo, se sistematizara el proceso para asignar el cambio de propietario de cada parqueadero.
5.	Se utiliza un croquis que está en guardianía y también se usa como guía la numeración de la casa que tiene en frente.	Esta forma que tienen de identificar no es eficiente, ya que no todas las casas del conjunto tienen domicilios en frente lo cual se optará por colocar un código de enumeración en cada parqueadero dependiendo de a que grupo pertenezcan.
6.	A todos los servicios que se quieren contratar se solicita una proforma y de acuerdo a los planes que ofrezcan se elige el que más se ajuste a las necesidades del conjunto.	Los servicios que se contratan se registran en una hoja de cálculo siendo así un proceso manual y tedioso, por lo cual se optará por digitalizar este proceso.

7.	No se lleva contabilidad como tal sino que se lleva un registro de ingresos y egresos. Los reportes varían dependiendo de la necesidad de la directiva, ya que no se tiene un reporte fijo salvo la rendición de cuentas que se realiza en las asambleas.	Se solicitó por parte de la directiva establecer unos reportes fijos que se generen automáticamente y que sean de fácil acceso.
8.	Se utiliza la pizarra de guardianía para emitir comunicados y también se envían mensajes por WhatsApp.	La directiva menciona que estos medios no son eficientes debido a que si el comunicado es enviado por WhatsApp muchos propietarios no lo reciben, ya que los grupos se llenan de mensajes que no son relevantes y discusiones entre vecinos. Por otro lado, la pizarra de guardianía no suele ser vista por todos los propietarios, debido a que han existido quejas que cuesta mucho leer lo que se escribe en ella.
9.	El registro de asistencias se hace en una hoja impresa en la cual se coloca el nombre del propietario o inquilino, el número de casa y la firma. Mientras que para el conteo de votos se realiza mediante el conteo de manos alzadas.	Se evidencia una necesidad clara de poder agilizar el proceso de registro de asistencia y el conteo de votaciones.
10.	Se le multa al residente por su inasistencia injustificada, ya que se le dan tres días para justificarla.	Ocasionalmente, suelen faltar más de veinte propietarios a las asambleas y se debe multar a todos ellos, lo cual es un proceso tedioso debido a que se les debe notificar uno por uno y esto consume mucho tiempo.
11.	Actualmente el registro de pagos y la generación de recibos son manuales y esto ha ocasionado quejas en los propietarios, ya que no se les entrega el recibo de pago a tiempo. Durante las asambleas la asistencia lleva más de media hora en concluirse.	La necesidad de poder sistematizar los pagos y la generación de recibos es evidente por parte de la directiva, así como también la necesidad de agilizar el proceso de registro de asistencia.

12.	El buzón de sugerencias es muy poco utilizado por los propietarios.	Las quejas las reciben a sus números de teléfono personal y esto les ocasiona que a menudo que no puedan atender todos o se les pase por alto alguno de estos mensajes.
13.	Casi toda la información que se maneja en la directiva están en libros escritos a mano y en hojas de cálculo.	Estos libros y hojas de cálculo se han ido pasando de directiva en directiva y ocasionalmente se pierden o se dañan y en los peores casos han provocado malos entendidos

a. Resultados de la encuesta aplicada a los residentes del conjunto habitacional "El Portal de la Viña"

Pregunta 1: ¿Cree usted que el proceso actual para obtener la información de las obligaciones financieras (pagos, multas, etc.) de su domicilio es eficiente?

Tabla 2. Resultados de la pregunta uno

Indicador	Frecuencia	Porcentaje
Totalmente de acuerdo	47	31.5%
De acuerdo	61	40.9%
Ni de acuerdo ni en desacuerdo	23	15.4%
En desacuerdo	13	8.7%
Totalmente en desacuerdo	5	3.5%
Suma total	149	100%

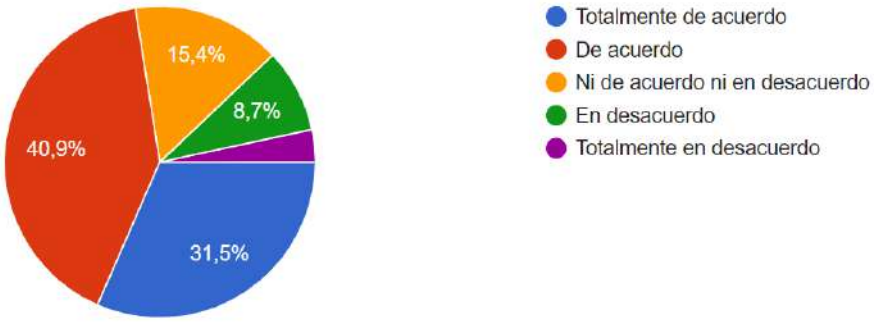


Figura 1. Resultados de la encuesta en la pregunta uno

Análisis e interpretación:

Los resultados de la pregunta uno muestran que el 31.5% de los propietarios del conjunto habitacional "El Portal de la Viña" están totalmente de acuerdo con el proceso

actual para obtener la información de las obligaciones financieras de su domicilio, el 40.9% de los propietarios están de acuerdo con el proceso actual, el 15.4% de los propietarios ni están de acuerdo ni en desacuerdo con el proceso actual, el 8.7% de los propietarios están en desacuerdo con el proceso actual y por último el 3.5% de los propietarios están totalmente en desacuerdo con el proceso actual. Tomando en cuenta los resultados obtenidos en la encuesta, se puede observar que el 72.4% de los propietarios del conjunto habitacional consideran que el proceso actual para obtener la información de las obligaciones financieras de su domicilio es eficiente, mientras que, el 27.6% de los propietarios consideran que el proceso actual no es eficiente y tomando en consideración las entrevistas realizadas a tesorería se concluye que se puede agilizar el proceso de consulta de información mediante el uso de una aplicación móvil.

Pregunta 2: ¿Con que frecuencia solicita la información de las obligaciones financieras (pagos, multas, etc.) de su domicilio a tesorería?

Tabla 3. Resultados de la pregunta dos

Indicador	Frecuencia	Porcentaje
Por lo menos una vez al mes	135	90.6%
Dos veces al mes	5	3.4%
Tres veces al mes	5	3.4%
Cuatro veces al mes	0	0%
Más de cuatro veces al mes	4	2.7%
Suma total	149	100%

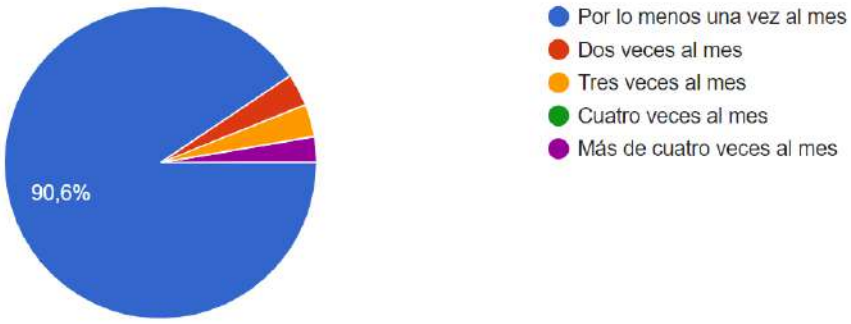


Figura 2. Resultados de la encuesta en la pregunta dos

Análisis e interpretación:

Los resultados de la pregunta dos evidencian que el 90.6% de los propietarios del conjunto habitacional "El Portal de la Viña" solicitan por lo menos una vez al mes la información de las obligaciones financieras de su domicilio a tesorería, el 3.4% de los propietarios solicitan dos veces al mes, el 3.4% de los propietarios solicitan tres veces

al mes, el 0% de los propietarios solicitan cuatro veces al mes y por último el 2.7% de los propietarios solicitan más de cuatro veces al mes. Por tanto, se puede observar que existe una demanda considerable de solicitudes a tesorería de manera mensual, lo cual junto con la entrevista realizada a tesorería se concluye que se puede facilitar el proceso de consulta de información mediante el uso de una aplicación móvil.

Pregunta 3: ¿Cree usted que el proceso actual para el registro de asistencias durante la asamblea es eficiente en términos de tiempo?

Tabla 4. Resultados de la pregunta tres

Indicador	Frecuencia	Porcentaje
Totalmente de acuerdo	29	19.5%
De acuerdo	55	36.9%
Ni de acuerdo ni en desacuerdo	34	22.8%
En desacuerdo	17	11.4%
Totalmente en desacuerdo	14	9.4%
Suma total	149	100%

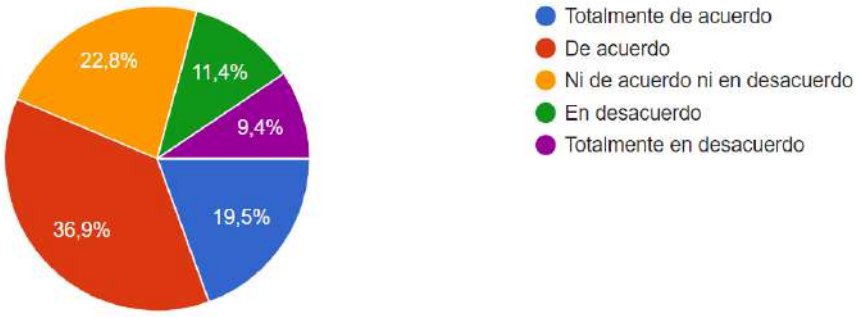


Figura 3. Resultados de la encuesta en la pregunta tres

Análisis e interpretación:

Los resultados de la pregunta tres muestran que el 19.5% de los propietarios del conjunto habitacional "El Portal de la Viña" están totalmente de acuerdo con el proceso actual para el registro de asistencias durante la asamblea, el 36.9% de los propietarios están de acuerdo con el proceso actual, el 22.8% de los propietarios ni están de acuerdo ni en desacuerdo con el proceso actual, el 11.4% de los propietarios están en desacuerdo con el proceso actual y por último el 9.4% de los propietarios están totalmente en desacuerdo con el proceso actual. Por lo tanto, se observa que el 56.4% de los propietarios del conjunto habitacional consideran que el proceso actual para el registro de asistencias durante la asamblea es eficiente en términos de tiempo, mientras que el 43.6% de los propietarios consideran que el proceso actual no es eficiente y tomando

en consideración las entrevistas realizadas a presidencia se concluye que se puede optimizar el proceso de registro de asistencias mediante el uso de sistemas informáticos.

Pregunta 4: ¿Considera usted que el conteo actual de las votaciones durante las asambleas es transparente y eficiente?

Tabla 5. Resultados de la pregunta cuatro

Indicador	Frecuencia	Porcentaje
Totalmente de acuerdo	50	33.6%
De acuerdo	57	38.3%
Ni de acuerdo ni en desacuerdo	31	20.8%
En desacuerdo	5	3.4%
Totalmente en desacuerdo	6	4%
Suma total	149	100%

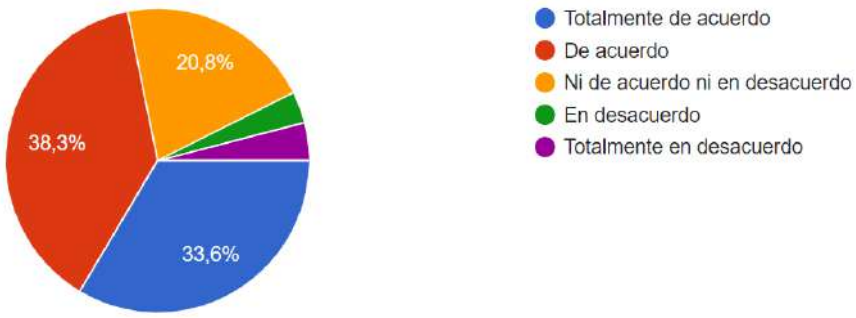


Figura 4. Resultados de la encuesta en la pregunta cuatro

Análisis e interpretación:

Los resultados de la pregunta cuatro muestran que el 33.6% de los propietarios del conjunto habitacional "El Portal de la Viña" están totalmente de acuerdo con el conteo actual de las votaciones durante las asambleas, el 38.3% de los propietarios están de acuerdo con el conteo actual, el 20.8% de los propietarios ni están de acuerdo ni en desacuerdo con el conteo actual, el 3.4% de los propietarios están en desacuerdo con el conteo actual y por último el 4% de los propietarios están totalmente en desacuerdo con el conteo actual. Por lo cual, se puede observar que el 71.9% de los propietarios del conjunto habitacional consideran que el conteo actual de las votaciones durante las asambleas es transparente y eficiente, mientras que el 28.1% de los propietarios consideran que el conteo actual no es transparente y eficiente y tomando en consideración las entrevistas realizadas a presidencia en donde se expuso que la duración de las mismas suelen ser de entre veinte a treinta minutos, se concluye que se puede sistematizar el proceso de conteo de votaciones mediante el uso de sistemas

informáticos para reducir y cumplir así con los tiempos establecidos de duración de las asambleas.

Pregunta 5: ¿Ha experimentado retrasos en la entrega de los comprobantes de pagos?

Tabla 6. Resultados de la pregunta cinco

Indicador	Frecuencia	Porcentaje
Si	62	41.6%
No	87	58.4%
Suma total	149	100%

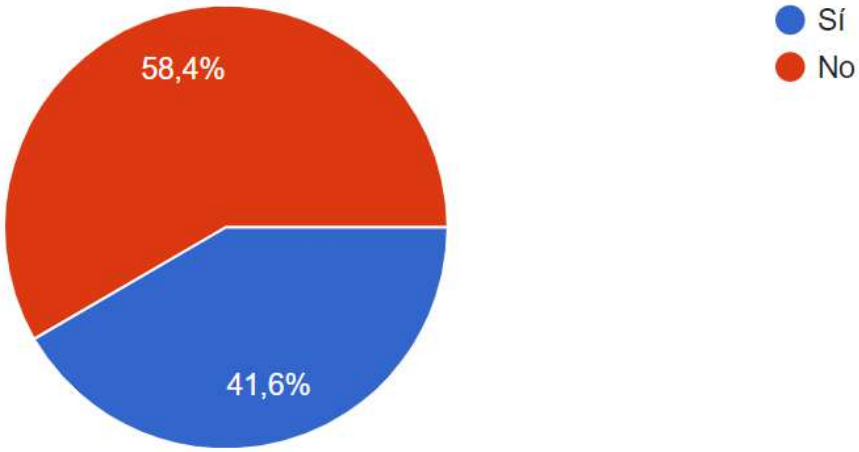


Figura 5. Resultados de la encuesta en la pregunta cinco

Análisis e interpretación:

Los resultados de la pregunta cinco muestran que el 41.6% de los propietarios del conjunto habitacional "El Portal de la Viña" han experimentado retrasos en la entrega de los comprobantes de pagos, mientras que el 58.4% de los propietarios no han experimentado retrasos en la entrega de los comprobantes de pagos. Por tanto, se evidencia que un sistema informático administrativo puede ser de gran ayuda para la directiva del conjunto habitacional, ya que se puede sistematizar el proceso de entrega de comprobantes de pagos y evitar retrasos en la entrega de los mismos.

Pregunta 6: ¿Con qué frecuencia olvida que el pago de alícuotas de su domicilio esta por vencer?

Tabla 7. Resultados de la pregunta seis

Indicador	Frecuencia	Porcentaje
Muy frecuentemente	21	14.1%
Frecuentemente	20	13.4%
Ocasionalmente	31	20.8%
Raramente	36	24.2%
Nunca	41	27.5%
Suma total	149	100%

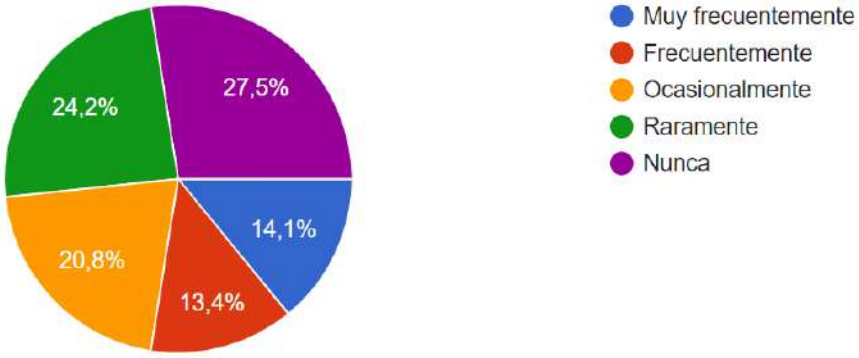


Figura 6. Resultados de la encuesta en la pregunta seis

Análisis e interpretación:

Los resultados de la pregunta seis muestran que el 14.1% de los propietarios del conjunto habitacional "El Portal de la Viña" olvidan muy frecuentemente que el pago de alícuotas de su domicilio están por vencer, el 13.4% de los propietarios olvidan frecuentemente, el 20.8% de los propietarios olvidan ocasionalmente, el 24.2% de los propietarios olvidan raramente y por último el 27.5% de los propietarios nunca olvidan que el pago de alícuotas. Para mejorar estos resultados se puede implementar recordatorios de pagos por vencer mediante una aplicación móvil, ya que se evidencia que el 48.3% de los propietarios lo olvidan muy frecuentemente, frecuentemente u ocasionalmente.

Pregunta 7: ¿Con qué frecuencia ha utilizado el buzón de quejas/sugerencias?

Tabla 8. Resultados de la pregunta siete

Indicador	Frecuencia	Porcentaje
Nunca lo he usado	142	95.3%
Al menos una vez al mes	6	4%
Dos veces al mes	0	0%
Tres veces al mes	0	0%
Más de tres veces al mes	1	0.7%
Suma total	149	100%

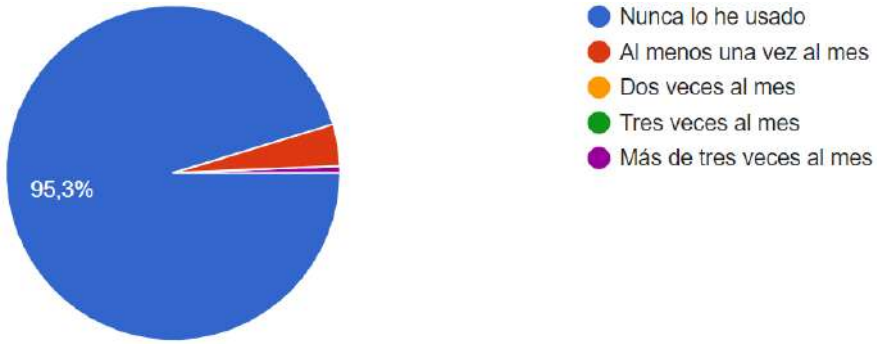


Figura 7. Resultados de la encuesta en la pregunta siete

Análisis e interpretación:

Los resultados de la pregunta siete muestran que el 95.3% de los propietarios del conjunto habitacional "El Portal de la Viña" nunca han utilizado el buzón de quejas/sugerencias, el 4% de los propietarios lo han utilizado al menos una vez al mes, el 0% de los propietarios lo han utilizado dos veces al mes, el 0% de los propietarios lo han utilizado tres veces al mes y por último el 0.7% de los propietarios lo han utilizado más de tres veces al mes. Por tanto, se evidencia que el buzón de quejas/sugerencias no es utilizado por los propietarios del conjunto habitacional, lo cual se puede mejorar mediante una aplicación móvil que permita a los propietarios acceder a la información de una manera más sencilla y eficiente.

Pregunta 8: ¿Considera usted que los medios actuales (WhatsApp/pizarrón en la garita) que utiliza la directiva para publicar comunicados son eficientes?

Tabla 9. Resultados de la pregunta ocho

Indicador	Frecuencia	Porcentaje
Totalmente de acuerdo	58	38.9%
De acuerdo	68	45.6%
Ni de acuerdo ni en desacuerdo	12	8.1%
En desacuerdo	4	2.7%
Totalmente en desacuerdo	7	4.7%
Suma total	149	100%

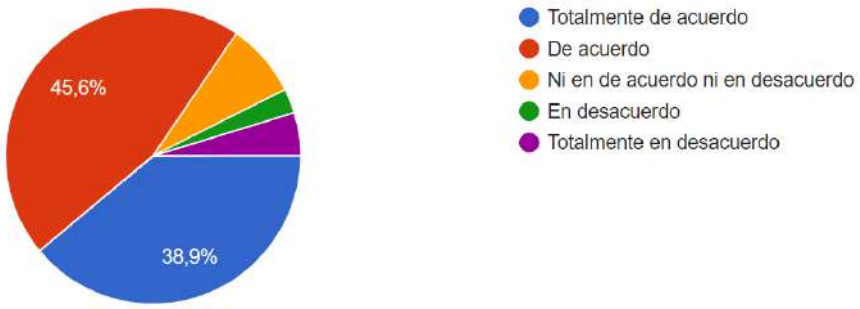


Figura 8. Resultados de la encuesta en la pregunta ocho

Análisis e interpretación:

Los resultados de la pregunta ocho muestran que el 38.9% de los propietarios del conjunto habitacional "El Portal de la Viña" están totalmente de acuerdo con los medios actuales que utiliza la directiva para publicar comunicados, el 45.6% de los propietarios están de acuerdo con los medios actuales, el 8.1% de los propietarios ni están de acuerdo ni en desacuerdo con los medios actuales, el 2.7% de los propietarios están en desacuerdo con los medios actuales y por último el 4.7% de los propietarios están totalmente en desacuerdo con los medios actuales. Para mejorar estos resultados se puede implementar un sistema informático que permita a la directiva publicar comunicados de una manera más centralizada, ya que tomando en consideración las entrevistas realizadas a la directiva el uso de WhatsApp no es apto para la publicación de comunicados por el mal uso que le dan los demás residentes y el pizarrón en la garita no es visible para todos los propietarios.

Pregunta 9: ¿Preferiría utilizar una aplicación móvil que le permita revisar la información de las obligaciones financieras de su domicilio, el registro de su asistencia y voto durante las asambleas, se le envíen notificaciones recordándole los pagos por vencer o que ha sido multado por alguna infracción?

Tabla 10. Resultados de la pregunta nueve

Indicador	Frecuencia	Porcentaje
Si	127	85.2%
No	22	14.8%
Suma total	149	100%

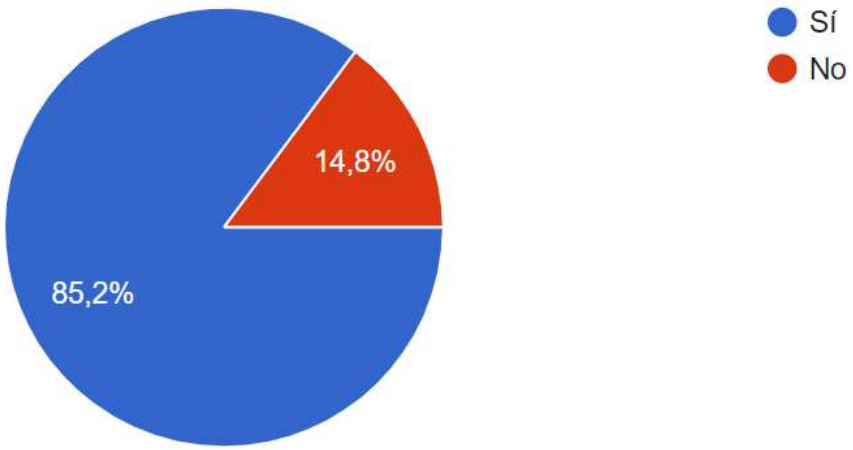


Figura 9. Resultados de la encuesta en la pregunta nueve

Análisis e interpretación:

Los resultados de la pregunta nueve muestran que el 85.2% de los propietarios del conjunto habitacional "El Portal de la Viña" preferirían utilizar una aplicación móvil que les permita revisar la información de las obligaciones financieras de su domicilio, el registro de su asistencia y voto durante las asambleas, se les envíen notificaciones recordándoles los pagos por vencer o que han sido multados por alguna infracción, mientras que el 14.8% de los propietarios no preferirían utilizar una aplicación móvil.

2.2.4 Procesamiento y análisis de datos

Con la información recolectada a través de las entrevistas y las encuestas se concluyó que:

- Actualmente la directiva lleva mucho de sus procesos de forma manual o en hojas de cálculo que han ocasionado atrasos y una alta carga de trabajo para todos los miembros de la misma. Los procesos administrativos actualmente funcionan, pero no son eficientes. Cada uno de los miembros de la directiva tienen sus funciones específicas, pero no cuentan con un sistema que les permita realizarlos de una manera más sencilla, puesto que todos ellos tienen empleos y no pueden dedicarle todo el tiempo que quisieran al conjunto habitacional.
- En presidencia se lleva a cabo el control y manejo de los parqueaderos de una manera manual y muy tediosa, debido a que deben de dibujar un croquis manual de los parqueaderos y actualizar las hojas de excel con los datos de los

propietarios. El tiempo empleado en las asambleas es demasiado extenso, por el motivo de que se debe hacer el registro de asistencia por cada residente de manera manual en una hoja impresa.

- En tesorería se maneja la información económica del conjunto en hojas de cálculo, teniendo en cuenta que son 303 casas y 275 parqueaderos en total, se convierte en un problema muy alto en cuanto a la cantidad de información que se debe manejar y el tiempo que se debe emplear en la generación de reportes. Además lo anteriormente mencionado provoca atrasos en la entrega de recibos de pagos.

- En secretaría se lleva la generación de gran parte de actas y convocatorias mediante el uso de Word o Canva, en donde se identificó que estos documentos son generados de forma manual y para emitirlos se utiliza WhatsApp. Como consecuencia de esto los propietarios no suelen enterarse de las convocatorias ya que el grupo de WhatsApp esta en gran parte del tiempo con mensajes irrelevantes y discusiones entre vecinos.

- De las encuestas realizadas a los propietarios se pudo identificar que la mayoría están conformes con la gestión de la directiva, sin embargo, se evidenció que los atrasos en las entregas de recibos de pagos es un problema bastante común. También existe un poco uso del buzón de sugerencias, ya que los propietarios prefieren comunicarse directamente con los miembros de la directiva. Y por último existe una necesidad alta de poder tener una aplicación móvil que les permite consultar información de su domicilio y notificar de eventos importantes.

CAPÍTULO III. RESULTADOS Y DISCUSIÓN

3.1 Análisis de herramientas de desarrollo

En la siguiente sección se detalla el análisis de las herramientas de desarrollo que se utilizarán para la implementación del sistema de gestión de conjunto habitacional tanto frameworks como herramientas y metodologías de desarrollo que mejor se adapten a las necesidades del proyecto.

3.1.1 Análisis y selección de frameworks

Dado que el sistema administrativo para la directiva será un sistema web y el sistema para los propietarios e inquilinos será un sistema móvil, se debe seleccionar un framework de desarrollo web, un framework de desarrollo móvil, un framework de desarrollo de APIs, un gestor de bases de datos y herramientas para la geolocalización.

a. Framework de desarrollo web

En primera instancia se debe evaluar las características de los frameworks de desarrollo web más utilizados en la actualidad, como Angular, React y Vue, para seleccionar el que mejor se adapte a las necesidades del proyecto. Para lo cual en [20] el autor muestra una tabla de comparación entre las características de los frameworks de desarrollo web Angular, React y Vue que se detallan a continuación.

Tabla 11. Características de los framework front-end Angular, React y Vue [20]

Característica	Angular	React	Vue
Tipo	Framework	Biblioteca	Framework
Año de lanzamiento	2016	2013	2014
Lenguaje de programación	TypeScript	JavaScript	JavaScript
Tamaño del ecosistema	Grande	Grande	Mediano
Enlace de datos bidireccional	Sí	Sí	Sí
Arquitectura de componentes	Sí	Sí	Sí

DOM Virtual	No	Sí	Sí
Curva de aprendizaje	Relativamente pronunciada	Moderada	Moderada
Rendimiento	Bueno	Muy Bueno	Bueno
Soporte comunitario	Fuerte	Fuerte	Fuerte
Casos de uso	Aplicaciones grandes y complejas	Varias escalas	Varias escalas

En la Tabla 12 analizan también las ventajas y desventajas de los frameworks de desarrollo web Angular, React y Vue. En donde el autor [20] realiza una extracción de las ventajas y desventajas de los frameworks los cuales serán detallados a continuación.

Tabla 12. Ventajas y desventajas de los framework front-end Angular, React y Vue [20]

Framework	Ventajas	Desventajas
Angular	▪ Potente ▪ Vinculación de datos bidireccional ▪ Arquitectura de componentes ▪ Enrutamiento y navegación ▪ Desarrollo multiplataforma	▪ Curva de aprendizaje pronunciada ▪ Grande y complejo ▪ Problemas de rendimiento ▪ No es optimo para proyectos pequeños ▪ Escalabilidad adecuada para proyectos grandes
React	▪ Alto rendimiento ▪ Desarrollo por componentes ▪ Comunidad activa ▪ Ecosistema rico	▪ Curva de aprendizaje alta para principiantes ▪ Sintaxis JSX ▪ Enfoque solo en la capa visible
Vue	▪ Fácil de aprender ▪ Marco incremental ▪ Responsive data binding ▪ Desarrollo por componentes ▪ Ecosistema rico	▪ Ecosistema pequeño comparado con Angular y React ▪ Menos soporte comunitario en comparación con Angular y React ▪ Menos consistente que Angular

Después de analizar las características, ventajas y desventajas de los frameworks de desarrollo web Angular, React y Vue, se selecciona el framework de desarrollo web Angular, ya que es el que mejor se adapta a las necesidades del proyecto debido a que proporciona un framework muy completo con una arquitectura ya predefinida

además de tener una variedad de herramientas y librerías que facilitan el desarrollo de aplicaciones web y teniendo en cuenta que posee un framework de desarrollo móvil llamado Ionic que facilita la creación de aplicaciones móviles multiplataforma.

b. Framework de desarrollo móvil

Como framework de desarrollo móvil se selecciona el framework de desarrollo móvil Ionic, ya que en el estudio comparativo en [21], el autor muestra una Tabla 13 de resultados en donde se tiene en cuenta diversos factores de rendimiento, seguridad, facilidad de acceso al hardware, uso de CPU, uso de memoria, entre otros factores, en donde se evidencia que el framework de desarrollo móvil Ionic obtiene un puntaje de 81 puntos, React Native 85 puntos y Flutter 79 puntos.

Tabla 13. Comparación de los frameworks: Métricas y puntuajes [21]

Métrica	Ionic	React Native	Flutter	NativeScript	MAUI	ReactJs
Plataformas objetivo	10	10	10	6	8	10
Acceso al hardware	13	16	16	14	14	16
Funciones específicas de la plataforma	18	17	18	18	16	12
Distribución de canales	4	4	4	4	4	1
Testeo	8	8	8	4	8	8
Monetización	6	6	6	6	3	6
Integración personalizada de código	2	2	2	2	2	0
Seguridad	8	6	6	8	6	6
Uso del CPU	3	6	2	1	5	4
Uso de memoria	3	6	2	1	5	4
Tamaño de la aplicación	6	4	5	2	1	3
Total	**81**	**85**	**79**	**66**	**72**	**70**

Por lo cual teniendo en consideración de que Ionic es un framework que en puede trabajar con Angular el cual también se utilizará para el desarrollo web y los resultados obtenidos del estudio revisado en donde se evidencia que Ionic tiene un buen

rendimiento en las métricas medidas siendo únicamente superada por React Native por 4 puntos se elige el framework de desarrollo móvil Ionic.

c. Framework de desarrollo back-end

Dado que el proyecto será tanto web como móvil se realizará un estudio comparativo de frameworks que faciliten la creación de una API para comunicar tanto interfaz de usuario web como la interfaz de usuario móvil con la base de datos.

Como lenguaje de programación se selecciona Java, ya que en [22], el autor menciona que java es un lenguaje muy robusto y seguro, orientado a objetos, con una amplia comunidad, además el autor menciona que es más eficiente que lenguajes de programación muy usados como .NET en términos de tiempo de respuesta y utilización de recursos.

En [23], el autor realizó un estudio comparativo entre los frameworks de desarrollo back-end Spring Boot, Django y Express, en dicho estudio se les realizaron pruebas de rendimiento enviando dieciséis mil peticiones HTTP simultáneas de tipo GET, POST, PUT y DELETE, en donde se evidenció la superioridad de Spring Boot ya que el autor menciona que éste framework supera al resto debido a que implementa mejores mecanismos de mejoras de rendimiento extraídos de Spring Framework, por lo cual se selecciona este framework para el desarrollo back-end.

d. Herramienta de geolocalización

Para la obtención de las ubicaciones en tiempo real se usara el plugin de geolocalización que viene integrado en Ionic, el cual esta provisto de métodos simples para obtener la ubicación actual y poder hacerle un seguimiento en tiempo real usando el GPS del dispositivo móvil [24].

Por otro lado, se usara la API de Google Maps para la visualización de las ubicaciones en un mapa, ya que esta API proporciona una variedad de herramientas para la visualización de mapas y la obtención de ubicaciones, además de funciones en la cuales nos permite representar áreas mediante la configuración de polígonos y círculos [25].

e. Gestor de bases de datos

Las bases de datos son un componente fundamental en el desarrollo de aplicaciones, por lo cual se debe seleccionar un gestor de bases de datos que se adapte a las necesidades del proyecto, en donde se debe tener en cuenta la escalabilidad, la seguridad, la disponibilidad y la facilidad de uso.

Tabla 14. Comparación entre gestores SQL, NoSQL y NewSQL [26]

Descripción	Relacional	No Relacional	NewSQL
Modelo de datos	Normaliza los datos en tablas conformadas por filas y columnas. Define estrictamente relación entre tablas	Proporciona una variedad de modelos de datos, como pares clave/valor, documentos y gráficos.	Estructura de datos flexible en donde es posible combinar la transaccionalidad y alta redundancia.
Cargas de trabajo óptima	Están diseñadas para aplicaciones de procesamiento de transacciones online (OLTP) y procesamiento analítico online (OLAP).	Las bases de datos de búsqueda NoSQL están diseñadas para hacer análisis sobre datos semiestructurados.	Alto rendimiento para cargas de trabajo OLTP/ OLAP.
Escalabilidad	Escalabilidad vertical, crecimiento de la cantidad de nodos de almacenamiento depende de la estructura tecnológica física.	Escalabilidad horizontal y se distribuye la carga por todos los nodos.	Escalabilidad horizontal proporcionando un alta rendimiento en una amplia gama de plataformas.
Propiedades ACID	Las bases de datos relacional ofrecen propiedades de atomicidad, coherencia, aislamiento y durabilidad (ACID).	Las bases de datos NoSQL hacen concesiones al flexibilizar algunas de las propiedades ACID para un modelo de datos más flexible.	Mantiene las propiedades de ACID de un sistema de base de datos tradicional o relacional.
Tolerancia a fallos	Fallo en el nodo generalmente hará fallar la consulta.	Configurados para que la perdida de algunos nodos no interrumpa funcionamiento global.	Crash Recovery: Las bases de datos NewSQL tienen un mecanismo que les permite recuperar datos y pasar a un estado coherente.

Una vez analizadas las características de los gestores descritos en la Tabla 14, se selecciona el gestor de bases de datos relacional, ya que son gestores en los cuales los datos están normalizados en tablas y se define estrictamente la relación entre tablas, además de que es un gestor de bases de datos que se adapta a las necesidades del proyecto, ya que Spring Boot tiene soporte para el ORM Hibernate que facilita la conexión y el manejo de la los datos.

Como sistema de gestor de bases de datos relacional se elige PostgreSQL, ya que en [27], el autor realiza un análisis de los gestores de bases de datos PostgreSQL y MySQL obteniendo como resultado que PostgeSQL tiene un mejor resultado en términos de consumo de CPU y memoria sobre MySQL, sin embargo en los procesos CRUD(Create, Read, Update, Delete) de eliminar y consultar imágenes MySQL tiene un mejor rendimiento, por lo cual teniendo en cuenta que no se realizarán almacenamiento de imágenes en la base de datos se selecciona PostgreSQL.

3.2 Definición de la metodología de desarrollo de software

En la siguiente sección se detalla la metodología de desarrollo de software que se utilizará para la implementación del sistema de gestión de conjunto habitacional. La metodología de desarrollo de software es un enfoque estructurado para la creación de sistemas de software. Existen dos tipos de metodologías de desarrollo de software: metodologías tradicionales y metodologías ágiles. A continuación, se mostrará una comparación entre ambas metodologías y se justificará la elección de la metodología seleccionada para el desarrollo del sistema de gestión de conjunto habitacional.

3.2.1 Comparación entre metodologías tradicionales y metodologías ágiles

En la tabla 15 se muestra una comparación entre las metodologías ágiles y las metodologías tradicionales. En donde se puede evidenciar que las metodologías ágiles se enfocan más en la adaptación de posibles cambios, en proyectos pequeños y con un equipo de trabajo pequeño junto con una buena colaboración entre el equipo de desarrollo y el cliente. Por otro lado las metodologías tradicionales se evidencia su utilidad de aplicación en proyectos grandes, con un equipo de trabajo grande y con una planificación previa amplia. Por el análisis previo se llegó a la conclusión que para

llevar este proyecto a cabo se utilizará una metodología ágil, ya que se cuenta con un equipo de trabajo pequeño, con entregas continuas al cliente y con la posibilidad de adaptarse a los cambios que se presenten durante el desarrollo del sistema. En [28] se extrajo la siguiente tabla que compara las metodologías ágiles con las metodologías tradicionales.

Tabla 15. Comparación entre metodologías ágiles y metodologías tradicionales traducido de [28]

Característica	Metodologías Ágiles	Metodologías Tradicionales
Enfoque	Adaptativo	Predictivo
Medición de éxito	Valor de negocio	Conforme al plan
Tamaño del proyecto	Pequeño	Grande
Estilo de gestión	Descentralizado	Autocrático
Perspectiva de cambio	Adaptable	Sostenible
Cultura	Liderazgo - Colaboración	Ordenar - Controlar
Documentación	Baja	Alta
Énfasis	Orientado al cliente	Orientado al proceso
Ciclos	Numerosos	Limitados
Domínio	Impredecible/Exploratorio	Predecible
Planificación previa	Mínima	Amplia
Retorno de la inversión	Principio del proyecto	Fin del proyecto
Tamaño del equipo	Pequeño	Grande

Por otro lado para escoger la metodología ágil que se utilizará en el desarrollo del sistema de gestión de conjunto habitacional se realizó un análisis de las metodologías ágiles más utilizadas en la actualidad detalladas a continuación en la siguiente tabla. En [29] se extrajo la siguiente tabla que compara algunas de las metodologías ágiles más utilizadas en la actualidad.

Tabla 16. Comparación de las metodologías de desarrollo ágiles [29]

Criterio	XP	Lean	RAD	Kanban
Enfoque	Iterativo e incremental	Iterativo e incremental	Prototipado	Continuo
Principios	Integración continua, programación en pares, desarrollo basado en pruebas, comentarios de los clientes	Centrarse en el valor, eliminar desperdicios, flujo, mejora continua, respeto por las personas	Desarrollo rápido, participación del usuario, desarrollo iterativo, creación de prototipos	Visualización del flujo de trabajo, limitación del trabajo en progreso, mejora continua
Tamaño del equipo	3–5	2–3	2–3	No definido
Tamaño del proyecto	Pequeños y medianos con requisitos bien definidos	Efectivo para proyectos grandes con requisitos complejos	Pequeños y medianos con requisitos cambiantes	Pequeños, medianos y grandes
Ventajas	Calidad y comunicación	Velocidad y flexibilidad	Costo y tiempo	Mejor flujo de trabajo
Simplicidad	Se busca la simplicidad en el código, en el diseño y en la solución de problemas	Se busca eliminar desperdicio y complejidad innecesaria para mejorar la eficiencia	Se enfoca en desarrollar soluciones simples y rápidas, evitando excesos de diseño	Busca eliminar desperdicio, simplificar procesos y claridad de flujo de trabajo
Entrega de software	Frecuente y regular	Entrega incremental	Entrega rápida	Entrega continua
Planificación	Planificación continua	Planificación y modelado	Planificación rápida y flexible	Planificación continua y visual

Del análisis de la Tabla 16 se llegó a la conclusión de que la metodología ágil que se utilizará en el desarrollo del proyecto será RAD, ya que se ajusta a las necesidades del proyecto, debido a que el equipo de desarrollo es pequeño, el proyecto es de tamaño mediano y se cuenta con tiempo limitado para la entrega del sistema. Además, sus características de desarrollo rápido junto con la entrega de prototipos funcionales y participación del usuario, permitirá una mayor adaptabilidad a los cambios que se presenten durante el desarrollo del sistema.

3.3 Cálculo de la distancia Haversine

El cálculo de la distancia entre dos puntos geográficos es una tarea común en la programación de aplicaciones que requieren la ubicación de un usuario o la ubicación de un lugar específico. En el caso de la aplicación móvil, se requiere conocer si el usuario se encuentra dentro de un rango de distancia de el lugar de reuniones en las asambleas de conjunto residencial.

En [30] el autor menciona que la fórmula de Haversine es una fórmula utilizada para calcular la distancia entre dos puntos en la superficie de una esfera, en este caso, la Tierra. Además, menciona que esta fórmula es precisa en el cálculo numérico incluso a distancias pequeñas, lo cual es ideal debido a que la precisión es un factor importante en la aplicación móvil, ya que será determinante para la verificación de la ubicación de los residentes.

La fórmula de Haversine se define como:

- R: Radio de la Tierra (aproximadamente 6371 km o 6371e3 metros).
- lat1, lon1: Latitud y longitud del primer punto en grados.
- lat2, lon2: Latitud y longitud del segundo punto en grados.
- $dLat$: Diferencia de latitud en radianes.
- $dLon$: Diferencia de longitud en radianes.
- a: Fórmula de Haversine.
- c: Ángulo central en radianes.
- d: Distancia entre los dos puntos en metros.

$$
\begin{aligned}
\mathrm{R} &= 6371e3 \\
\mathrm{dLat} &= \mathrm{lat2} - \mathrm{lat1} \\
\mathrm{dLon} &= \mathrm{lon2} - \mathrm{lon1} \\
a &= \sin^2\left(\frac{dLat}{2}\right) + \cos(\mathrm{lat1}) \cdot \cos(\mathrm{lat2}) \cdot \sin^2\left(\frac{dLon}{2}\right) \\
c &= 2 \cdot \mathrm{atan2}\left(\sqrt{a}, \sqrt{1-a}\right) \\
d &= R \cdot c
\end{aligned}
\tag{4}
$$

3.4 Análisis del proceso actual

Para poder determinar el flujo del sistema se deben analizar los procesos actuales del conjunto habitacional, para ello se debe identificar los actividades que se llevan a cabo en el conjunto habitacional, los actores que intervienen en las mismas y los resultados que se obtienen. Estos procesos fueron identificados en base a los resultados de las entrevistas y encuestas realizadas en el capítulo anterior.

A continuación se muestran los procesos identificados en el conjunto habitacional:

- Proceso de administración de parqueaderos. Proceso en el cual previamente revisada la documentación entregada de manera física del propietario o inquilino se le proporcionará un parqueadero de zona azul o particular dependiendo del tipo de parqueadero solicitado y siempre y cuando se cumpla con los requisitos establecidos en el reglamento del conjunto habitacional.
- Proceso de administración de eventos sociales. Proceso en el cual se lleva a cabo la organización de eventos sociales en el conjunto habitacional.
- Proceso de administración de guardianía. Proceso en el cual se lleva a cabo la administración de la guardianía del conjunto habitacional junto con las actividades que se realizan en la misma y los incidentes que se presentan.
- Proceso de administración de convocatorias. Proceso en el cual se lleva a cabo la organización de convocatorias en el conjunto habitacional, en donde en el caso de ser asambleas se registra la asistencia de los propietarios o inquilinos y se lleva a cabo la votación de las propuestas.

- Proceso de administración financiera. Proceso en el cual se administran los recursos financieros del conjunto habitacional tanto los ingresos como los egresos, multas y los proveedores.
- Proceso de administración de áreas comunales. Proceso en el cual se lleva a cabo la gestión del mantenimiento de las áreas comunales del conjunto habitacional tales como la limpieza y cortes de césped.
- Proceso de administración de la bitácora de la directiva. Proceso en el cual se lleva a cabo el registro de las actividades realizadas por la directiva del conjunto habitacional.
- Proceso de generación de documentos. Proceso en el cual se lleva a cabo la generación de documentos para los propietarios o inquilinos del conjunto habitacional.
- Proceso de consulta obligaciones financieras. Proceso en el cual se lleva a cabo la consulta de información de las obligaciones financieras de los propietarios o inquilinos del conjunto habitacional.
- Proceso de registro de asistencias en asambleas. Proceso en el cual cada propietario o inquilino del conjunto habitacional registra su asistencia a las asambleas.
- Proceso de votación en asambleas. Proceso en el cual únicamente los propietarios del conjunto habitacional realizan la votación de los temas a tratar en las asambleas.

De los procesos mencionados anteriormente se establecieron cuatro procesos principales los cuales serán detallados a continuación:

- Proceso de administración de parqueaderos (Zona azul).

    1. El propietario o inquilino debe realizar una solicitud via telefónica o presencial al presidente de la directiva.
    2. Se verifica que el propietario o inquilino cumpla con los requisitos establecidos en el reglamento de parqueaderos.
        (a) Si cumple con los requisitos se procede al siguiente proceso.

(b) Si no cumple con los requisitos se finaliza el proceso.

3. El presidente solicita enviar la documentación requerida para la asignación de un parqueadero.
   (a) Si entrega la documentación completa se procede al siguiente proceso.
   (b) Si no entrega la documentación completa se finaliza el proceso.
4. El inquilino debe realizar el pago por adelantado de 10$.
5. Se verifica el pago realizado por el inquilino.
   (a) Si el pago es válido se procede al siguiente proceso.
   (b) Si el pago no es válido se finaliza el proceso.
6. Se registra el pago en el libro de ingresos.
7. Se genera la orden de pago.
8. Se registra al propietario o inquilino en el libro de parqueaderos.
9. Se genera una carta de compromiso.
10. Se entrega la orden de pago físicamente al inquilino y se le toma una foto como respaldo.

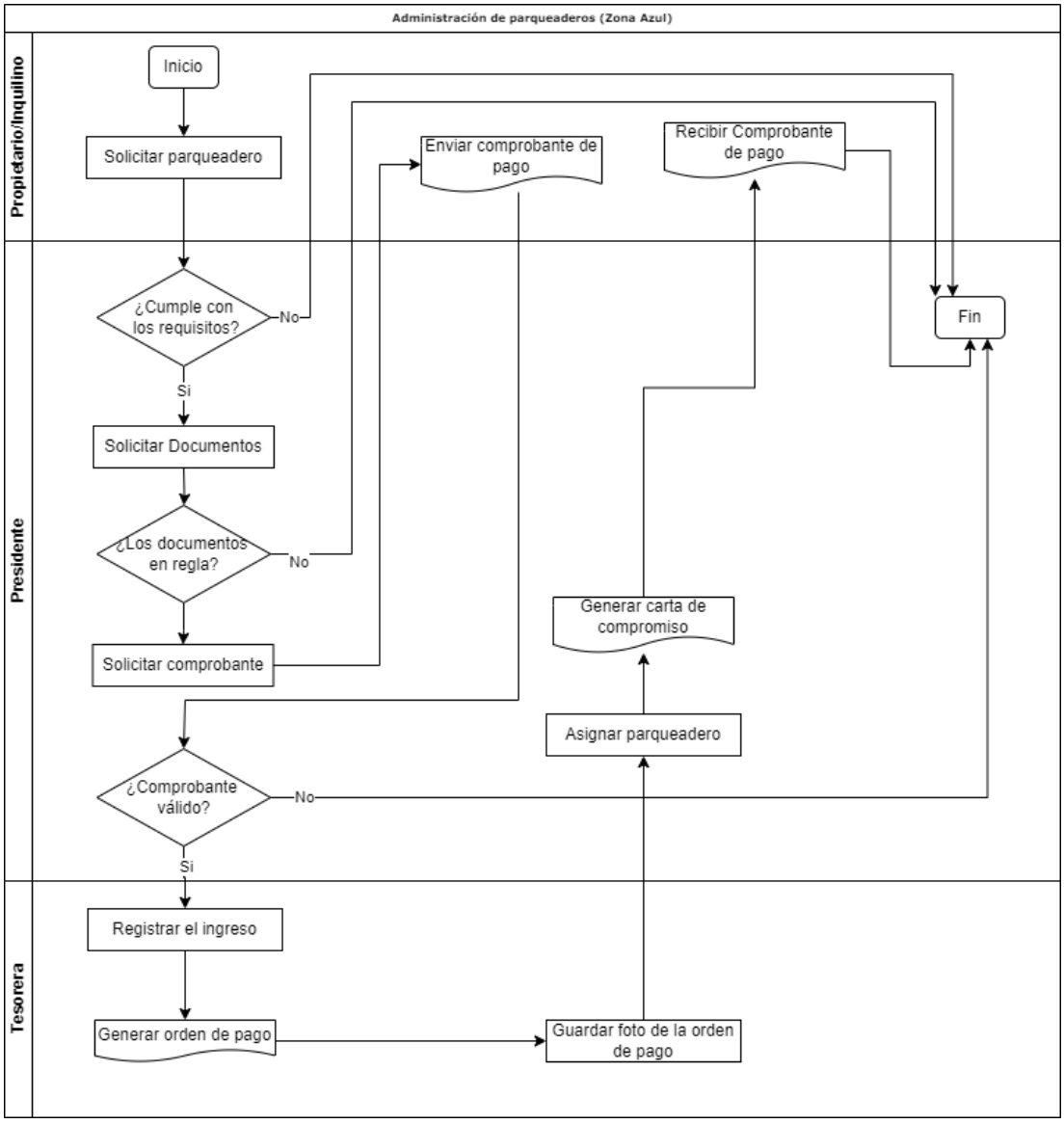


Figura 10. Diagrama de flujo de procesos actual de administración de parqueaderos (Zona azul).

- Proceso de administración de convocatorias.

    1. La directiva del conjunto se reúne para definir la fecha de la convocatoria.
    2. Se crea la publicación de la convocatoria y se comparte en los medios de comunicación del conjunto habitacional.
    3. Se reenvía la publicación como recordatoria una semana antes de darse la convocatoria solo si es de asambleas.
    4. Se lleva a cabo la convocatoria.
        - Si es una asamblea se procede al siguiente proceso.

- Si es una reunión o sesión de directiva se salta al proceso 6.

5. Se registra la asistencia de los propietarios o inquilinos.
6. Se tratan los temas de la convocatoria.
7. Se propone una votación de los temas a tratar.
    - Si existe una votación se procede al siguiente proceso.
    - Si no existe una votación se salta al proceso 9.
8. Se lleva a cabo el conteo de votos a mano alzada únicamente de propietarios.
9. Se notifica a los propietarios el resultado de la votación.
10. Se finaliza la convocatoria.
11. Se genera el informe de asistencia.
12. Se genera el acta de la convocatoria.
    - Si es una asamblea se procede al siguiente proceso.
    - Si es una reunión o sesión de directiva se finaliza el proceso.
13. Se envía el informe de asistencia a los residentes y se registra la respectiva multa.
14. El propietario o inquilino puede presentar una justificación por la inasistencia en las siguientes 24 horas.
15. Se verifica la justificación presentada.
    - Si la justificación es válida se procede a eliminar la multa y termina el proceso.
    - Si la justificación no es válida la multa se mantiene y termina el proceso.

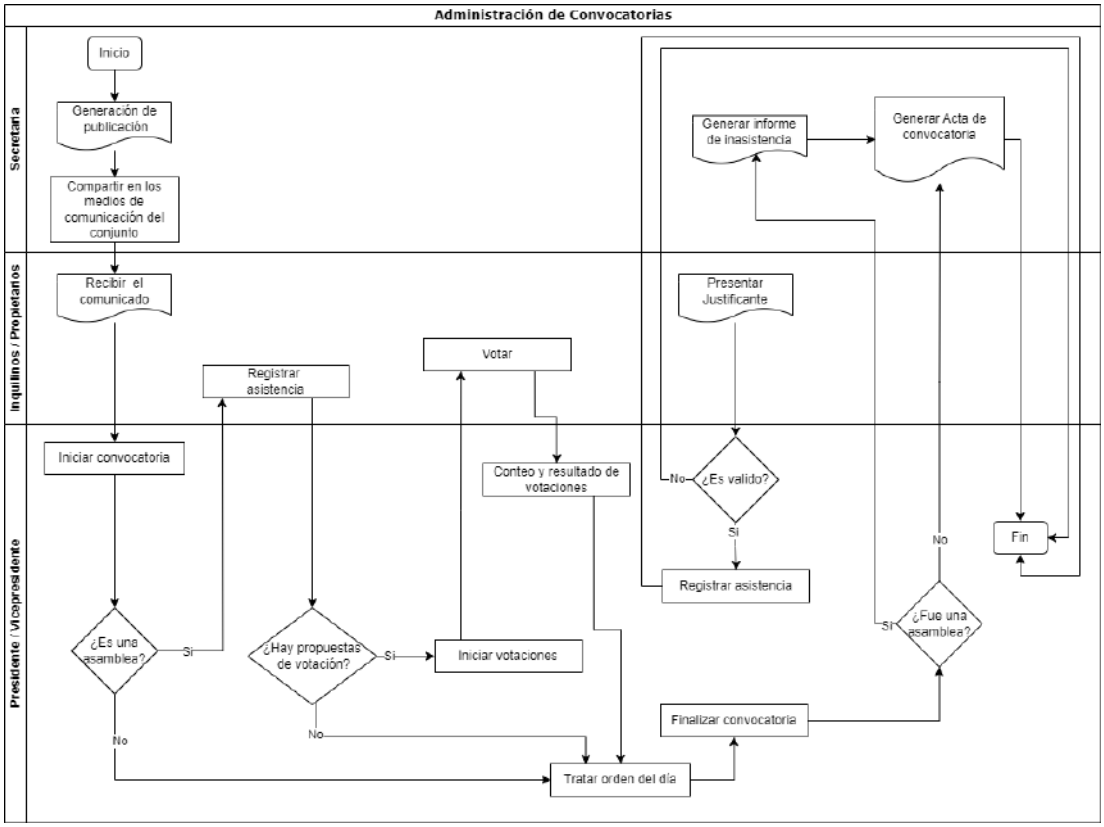


Figura 11. Diagrama de flujo de procesos actual de administración de convocatorias.

- Proceso de administración financiera.

1. Pagos de obligaciones financieras mensuales

(a) El inquilino realiza el pago de la mensualidad de sus obligaciones financieras.

(b) El inquilino envía el comprobante de pago a la tesorera.

(c) La tesorera verifica el comprobante de pago.

- Si el comprobante es válido se procede al siguiente proceso.
- Si el comprobante no es válido se finaliza el proceso.

(d) La tesorera revisa la última fecha de pago del inquilino.

(e) Se registra el pago en el libro de ingresos.

(f) Se genera una orden de pago.

(g) La tesorera entrega la orden de pago a guardianía pára que se le entregue al inquilino.

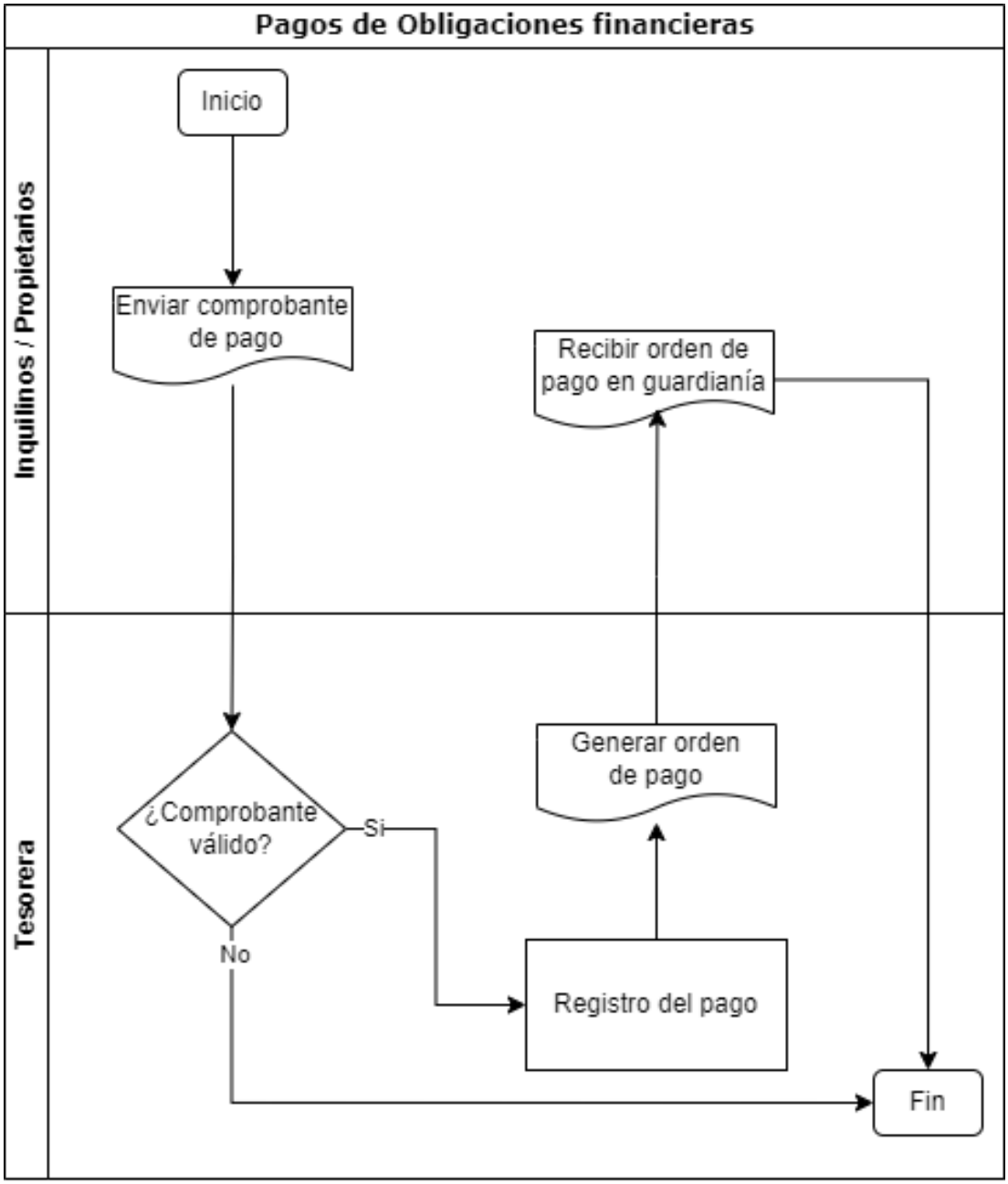


Figura 12. Diagrama de flujo de procesos actual del pago de obligaciones financieras.

2. Pagos a proveedores

   (a) El proveedor envía la factura a la tesorera.

   (b) La tesorera verifica la factura.

   - Si la factura es válida se procede al siguiente proceso.
   - Si la factura no es válida se finaliza el proceso.

   (c) La tesorera registra la factura en el libro de egresos.

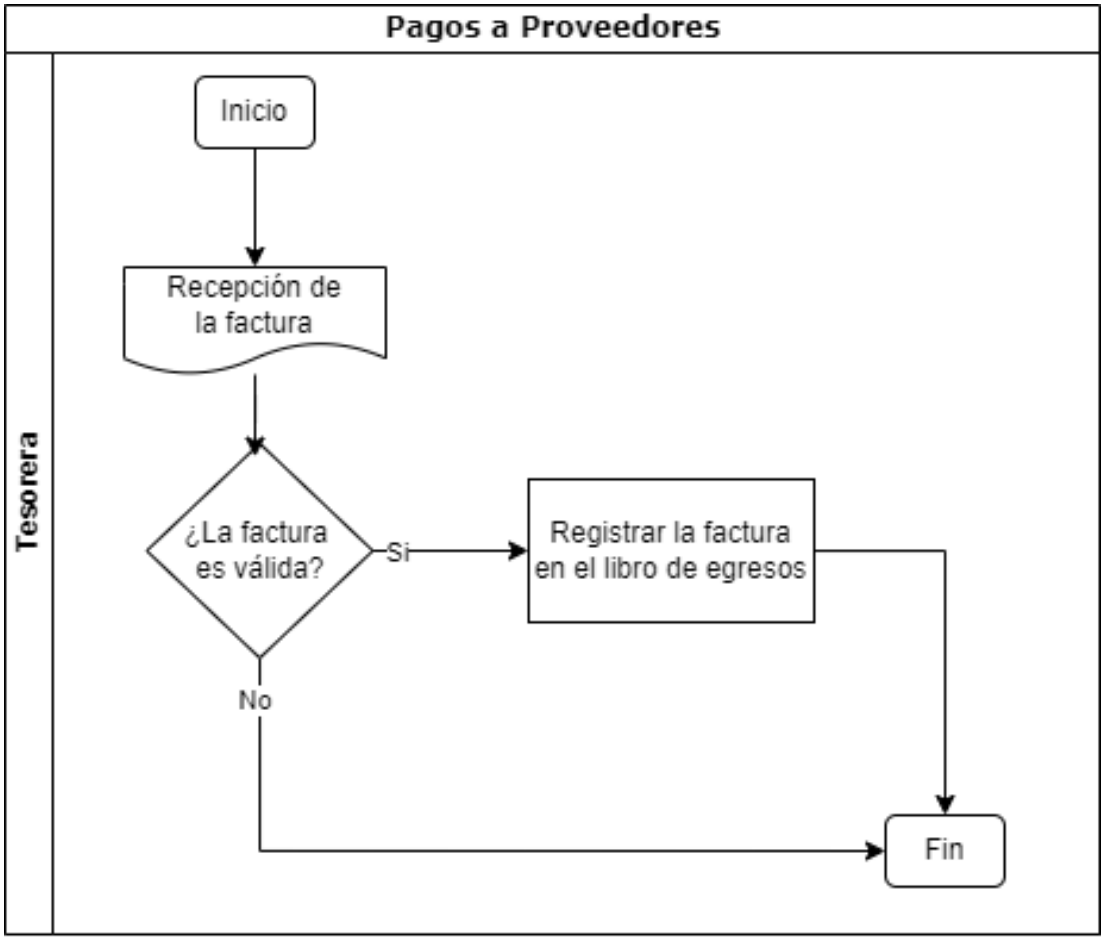


Figura 13. Diagrama de flujo de procesos actual del pago de obligaciones financieras.

3. Multas

   (a) Se registra la multa en el libro de multas.

   (b) Se guardan mediante fotos en los teléfonos de los directivos las infracciones cometidas por los propietarios o inquilinos.

   (c) Se le notifica al propietario o inquilino la multa.

   (d) El propietario o inquilino envía el comprobante de pago de la multa.

   (e) La tesorera verifica el comprobante de pago.

      - Si el comprobante es válido se procede al siguiente proceso.
      - Si el comprobante no es válido se mantiene la multa y se finaliza el proceso.

   (f) Se registra el pago de la multa en el libro de ingresos.

   (g) Se genera una orden de pago.

   (h) La tesorera entrega la orden de pago a guardianía para que se le entregue al propietario o inquilino.

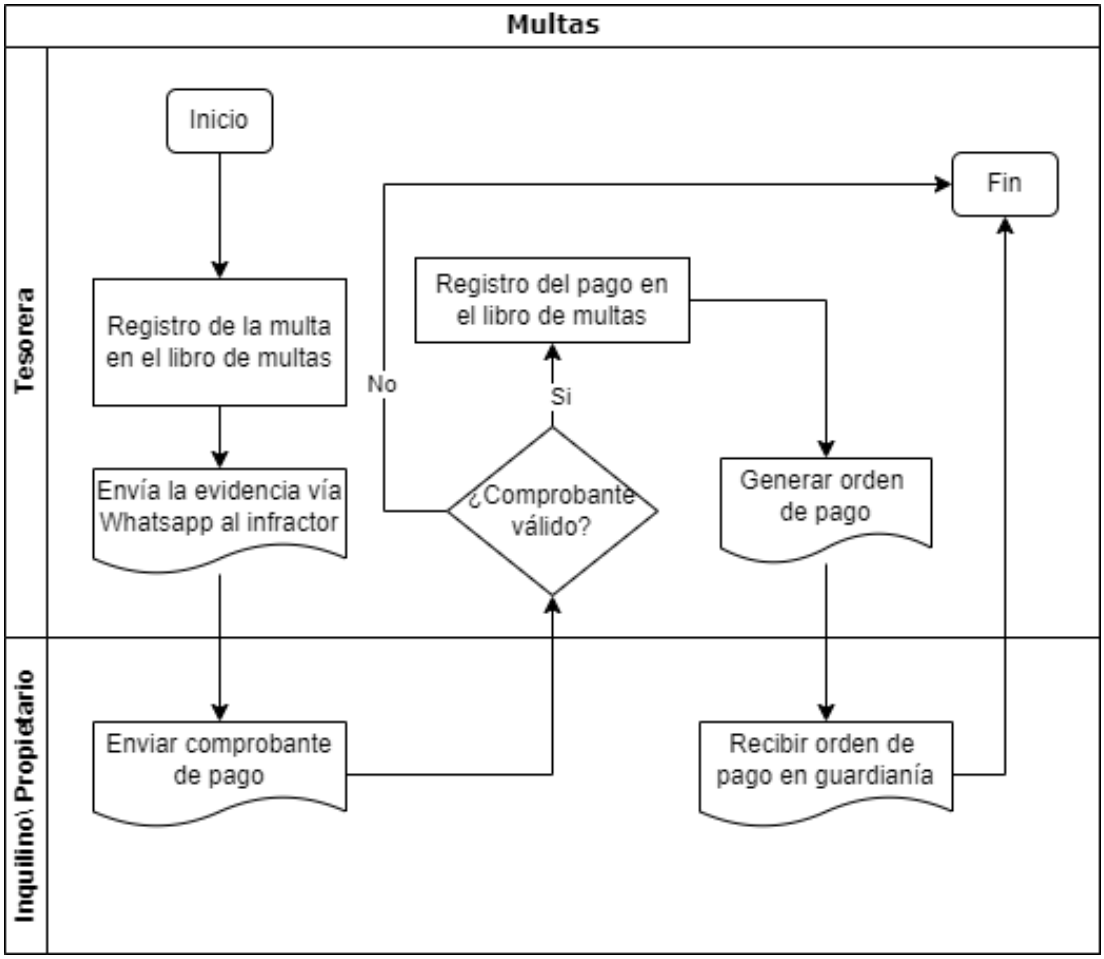


Figura 14. Diagrama de flujo de procesos actual de multas.

3.5 Desarrollo de la propuesta

Para el desarrollo de la propuesta se aplicó la metodología RAD la cual según [31], la autora menciona que esta metodología ayuda al desarrollo rápido de aplicaciones de una manera rápida y económica enfocada principalmente a empresas con baja disponibilidad de recursos y tiempo.

La metodología RAD se basa en cuatro fases principales:

- Planificación de requerimientos

  En esta primera fase se identifican los requerimientos del sistema para satisfacer las necesidades del cliente y se establece el alcance del proyecto.

- Diseño de usuario

  Una vez identificados los requerimientos se crean modelos de diseño previos a

la construcción del sistema los cuales son presentados a los usuarios para recibir retroalimentación.

- Construcción

  En esta fase se construye el sistema de acuerdo a los modelos de diseño previamente aprobados, mediante la codificación y pruebas del sistema.

- Transición

  La fase final en la que se realiza la entrega del sistema levantado en un entorno de producción real en donde se realizarán pruebas finales y se capacitará a los usuarios finales.

3.5.1 Planificación de requerimientos

En esta primera etapa se realizó un análisis de la información recopilada en las entrevistas, encuestas y en la situación actual descrita anteriormente.
En las siguientes tablas se detallan los usuarios identificados en la interacción con el sistema web y la aplicación móvil. En la Tabla 19 se detallan los requerimientos del proyecto identificados por usuarios.

Tabla 17. Descripción de los usuarios identificados en la interacción con el sistema web administrativo

Usuario	**Descripción**
Administrador	Usuario encargado de la administración de usuarios y la configuración de la geolocalización.
Presidente	Usuario encargado de la administración de parqueaderos, residencias, guardianía, convocatorias
Vicepresidente	Usuario encargado de la administración de residencias, guardianía y convocatorias
Tesorero	Usuario encargado de la administración de los ingresos, multas y egresos
Secretario	Usuario encargado de la administración de los eventos sociales y convocatorias

Tabla 18. Descripción de los usuarios identificados en la interacción con la aplicación móvil

Usuario	**Descripción**
Propietario	Usuario encargado de registrar la asistencia en asambleas y votación, visualización de las obligaciones financieras relacionadas con su residencia o sus residencias
Inquilino	Usuario encargado de registrar la asistencia en asambleas y visualización de las obligaciones financieras relacionadas con su residencia o sus residencias

Tabla 19. Identificación de los requerimientos del sistema

ID	**Requerimiento**	**Descripción**	**Prioridad**	**Riesgo**
Todos los usuarios				
R1	Iniciar sesión	El usuario podrá iniciar sesión mediante su autenticación ingresando sus credenciales: cédula y contraseña	Alta	Alto
R2	Cerrar sesión	El usuario podrá finalizar la sesión en cualquier momento	Alta	Bajo
R3	Recuperar contraseña	El usuario en caso de olvidar la contraseña podrá solicitar una contraseña nueva autogenerada por el sistema y enviada a su correo electrónico	Media	Bajo
R4	Cambiar contraseña	El usuario podrá cambiar su contraseña en cualquier momento	Media	Bajo
Administrador				
R5	Gestionar usuarios	El administrador podrá visualizar, registrar, inhabilitar o editar la información y roles de los demás usuarios	Alta	Alto
R6	Gestionar roles	El administrador podrá visualizar o editar la descripción de los roles existentes en el sistema	Media	Bajo
R7	Gestionar pasajes	El administrador podrá visualizar o editar la descripción de los pasajes existentes en el sistema	Media	Bajo
R8	Gestionar geolocalización	El administrador podrá editar la ubicación de las coordenadas y el radio de aceptación mediante la visualización de un mapa del lugar en donde se darán lugar las asambleas	Alta	Alto

Presidente				
R9	Gestionar estacionamientos	El presidente podrá visualizar, eliminar o actualizar la residencia asociada a cada parqueadero. También podrá visualizar o editar la descripción de los tipos de parqueaderos existentes	Alta	Alto
Presidente y Vicepresidente				
R10	Gestionar residencias	El presidente o vicepresidente podrán visualizar, eliminar o actualizar el inquilino o propietario de cada residencia del conjunto	Alta	Alto
R11	Gestionar guardias	El presidente o vicepresidente podrán visualizar, registrar, editar, o inhabilitar a los guardias de seguridad	Alta	Alto
R12	Gestionar actividades de guardianía	El presidente o vicepresidente podrá visualizar, registrar, editar, o eliminar las actividades de guardianía. También podrá cambiar el estado de cada actividad	Alta	Alto
R13	Gestionar los tipos de incidentes	El presidente o vicepresidente podrán visualizar, registrar, editar, o inhabilitar los tipos de incidentes	Alta	Alto
R14	Gestionar incidentes	El presidente o vicepresidente podrán visualizar, crear, editar, o eliminar los incidentes reportados por los guardías, asi como el cambio del estado del incidente	Alta	Alto
R15	Gestionar convocatorias	El presidente o vicepresidente podrá visualizar, descargar, registrar, editar, finalizar o eliminar las convocatorias,	Alta	Alto
R16	Gestionar asistencias de las asambleas	Las convocatorias de tipo asamblea son las únicas que poseerán registro de asistencias, de tal manera que el presidente o vicepresidente podrán realizar el registro manual de las asistencias, asi como descargar un reporte de inasistentes	Alta	Alto
R17	Gestionar votaciones de las asambleas	Las convocatorias de tipo asamblea son las únicas que poseerán votaciones, de tal manera que el presidente o vicepresidente podrán visualizar, editar, habilitar el voto o eliminar de cada pregunta propuesta a votación, asi como también poder visualizar la información de votación de cada votante	Alta	Alto

Tesorero				
R18	Gestionar ingresos mensuales	Posterior a la entrega física o digital del comprobante de pago por parte del residente la tesorera procede a registrar los datos del pago junto con el comprobante teniendo en cuenta el último mes de pago y hasta que mes está abonando el residente y posteriormente se envía al correo electrónico un respaldo del registro del pago, adicional a esto también puede visualizar, editar la información así como actualizar el comprobante de pago o eliminar el registro del pago	Alta	Alto
R19	Gestionar ingresos casuales	Posterior a la entrega física o digital del comprobante de pago por parte del residente la tesorera procede a registrar los datos del pago junto con el comprobante y posteriormente se envía al correo electrónico un respaldo del registro del pago, adicional a esto también puede visualizar, editar la información así como actualizar el comprobante de pago o eliminar el registro del pago	Alta	Alto
R20	Gestionar multas	La tesorera podrá visualizar, crear, editar o eliminar las multas. Los residentes deben presentar de manera física o digital el comprobante del pago de la multa por el monto indicado y posterior a su revisión se procede a subir el comprobante y a actualizar el estado del pago de la multa, posteriormente se envía al correo electrónico un respaldo del registro del pago	Alta	Alto
Secretario				
R21	Gestionar eventos sociales	El secretario podrá visualizar, registrar, editar, o eliminar los eventos sociales.	Media	Bajo
R22	Subir actas	El secretario podrá subir actas de las convocatorias	Media	Bajo
Propietario e Inquilino				
R23	Visualizar el calendario	El propietario o inquilino podrán visualizar el calendario de eventos sociales y asambleas próximas	Media	Bajo
R24	Visualizar estado de las obligaciones financieras	El propietario o inquilino podrán visualizar el estado financiero de sus obligaciones financieras de todas sus residencias o parqueaderos	Media	Bajo
R25	Visualizar asamblea del día	El propietario o inquilino podrán visualizar la asamblea únicamente si se da en ese día	Alta	Bajo

R26	Registrar asistencia	El propietario o inquilino podrán registrar su asistencia siempre que se encuentre dentro del rango de geolocalización registrado por el administrador del sistema	Alta	Alto
Propietario				
R27	Registrar voto	El propietario podrá registrar su voto en las preguntas que se encuentren habilitadas para su voto siempre que se encuentre dentro del rango de geolocalización registrado por el administrador del sistema y tenga registrada la asistencia	Alta	Alto

Una vez definidos los requerimientos del sistema se define el plan de trabajo en cada iteración para el desarrollo de los sistemas.

Tabla 20. Plan de trabajo de iteraciones

N° Iteración	**N°**	**ID**	**Requerimiento**	**Tiempo estimado**	
				Horas	**dias**
Iteración 1	1	R1	Iniciar sesión	6	1
	2	R2	Cerrar sesión	1	1
	3	R3	Recuperar contraseña	2	1
	4	R4	Cambiar contraseña	1	1
	5	R5	Gestionar usuarios	10	2
	6	R6	Gestionar roles	3	1
	7	R7	Gestionar pasajes	3	1
	8	R8	Gestionar geolocalización	6	1
Iteración 2	9	R9	Gestionar estacionamientos	16	2
	10	R10	Gestionar residencias	8	1
	11	R11	Gestionar guardias	6	1
	12	R12	Gestionar actividades de guardianía	8	1
	13	R13	Gestionar los tipos de incidentes	4	1
	14	R14	Gestionar incidentes	8	1
	15	R15	Gestionar convocatorias	16	2
	16	R16	Gestionar asistencias de las asambleas	8	1
	17	R17	Gestionar votaciones de las asambleas	10	2

Iteración 3	18	R18	Gestionar ingresos mensuales	16	2
	19	R19	Gestionar ingresos casuales	8	1
	20	R20	Gestionar multas	14	2
	21	R21	Gestionar eventos sociales	8	1
	22	R22	Subir actas	2	1
	23	R23	Visualizar el calendario	4	1
	24	R24	Visualizar estado de las obligaciones financieras	12	2
	25	R25	Visualizar asamblea del día	4	1
	26	R26	Registrar asistencia	8	1
	27	R27	Registrar voto	8	1

3.5.2 Diseño de usuario

a. Análisis de los procesos sistematizados propuestos

A continuación se detallaran los procesos sistematizados para la administración de parqueaderos, convocatorias y administración financiera.

- Proceso de administración de parqueaderos (Zona azul).

1. Se verifica que el propietario o inquilino este registrado en el sistema.
   (a) Si está registrado se procede al siguiente proceso.
   (b) Si no está registrado se le notifica al administrador para su registro en el sistema y se repite el proceso.
2. El presidente solicita enviar la documentación requerida para la asignación de un parqueadero.
   (a) Si entrega la documentación completa se procede al siguiente proceso.
   (b) Si no entrega la documentación completa se finaliza el proceso.
3. El presidente muestra en el sistema el mapa de parqueaderos disponibles.
4. El propietario o inquilino debe abonar diez dólares por la asignación del parqueadero y enviar el comprobante de pago a la tesorera.

5. La tesorera verifica el comprobante de pago de diez dólares por la asignación del parqueadero.
6. Se registra el pago en los ingresos mensuales en el sistema.
7. Se sube el comprobante de pago al sistema.
8. Se envía por correo electrónico el respaldo de pago al propietario o inquilino.
9. El presidente asigna el parqueadero al domicilio relacionado al propietario o inquilino.
10. El sistema genera una carta de compromiso.

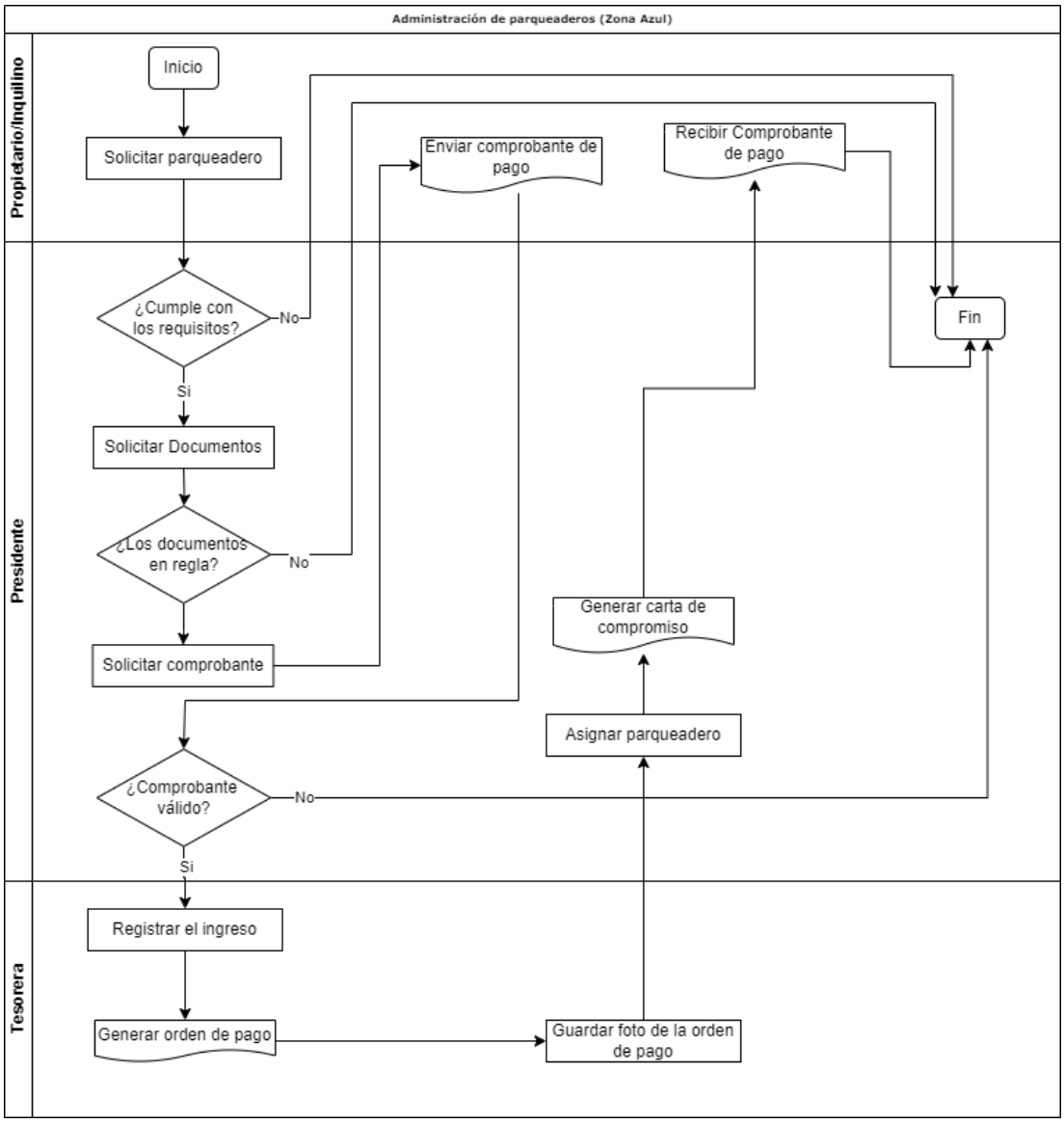


Figura 15. Diagrama de flujo de procesos sistematizado prupuesto de administración de parqueaderos (Zona azul).

- Proceso de administración de convocatorias.

    1. La directiva del conjunto se reúne para definir la fecha de la convocatoria.
    2. Se registra en el sistema la convocatoria y se notifica a los propietarios o inquilinos mediante la aplicación móvil.
    3. Se lleva a cabo la convocatoria.
        - Si es una asamblea se procede al siguiente proceso.
        - Si es una reunión o sesión de directiva se salta al proceso 6.
    4. Los inquilinos o propietarios registran su asistencia en la aplicación móvil.
        (a) Si no esta registrado en el sistema el presidente o vicepresidente registra su asistencia de su domicilio al que representa.
        (b) Si esta registrado en el sistema se procede al siguiente proceso.
    5. Se verifica la ubicación del propietario o inquilino mediante la geolocalización.
        - Si se encuentra en el rango de geolocalización se registra la asistencia y se procede al siguiente proceso.
        - Si no se encuentra en el rango de geolocalización no se registra la asistencia.
    6. Se tratan los temas de la convocatoria.
    7. Se propone una votación de los temas a tratar.
        - Si existe una votación se procede al siguiente proceso.
        - Si no existe una votación se salta al proceso 14.
    8. Se registra la votación en el sistema.
    9. Se habilita el voto a los propietarios o inquilinos.
    10. El sistema verifica que el usuario que va a votar sea propietario.
        - Si es propietario se procede al siguiente proceso.
        - Si no es propietario no se registra el voto.
    11. Los propietarios registran su voto en la aplicación móvil.
    12. Se verifica la ubicación del propietario mediante la geolocalización.
        - Si se encuentra en el rango de geolocalización se registra el voto y se procede al siguiente proceso.

- Si no se encuentra en el rango de geolocalización no se registra el voto.

13. Se muestra el resultado de la votación desde el sistema.
14. Se finaliza la convocatoria.
15. Se sube el acta de la convocatoria al sistema.
    - Si es una asamblea se procede al siguiente proceso.
    - Si es una reunión o sesión de directiva se finaliza el proceso.
16. Se genera el informe de inasistencia de los propietarios o inquilinos desde el sistema.
17. Se envía el informe de asistencia a los residentes y se registra la respectiva multa en el sistema.
18. El propietario o inquilino puede presentar una justificación por la inasistencia en las siguientes 24 horas.
19. Se verifica la justificación presentada.
    - Si la justificación es válida se procede a eliminar la multa del sistema y se finaliza el proceso.
    - Si la justificación no es válida la multa se mantiene y termina el proceso.

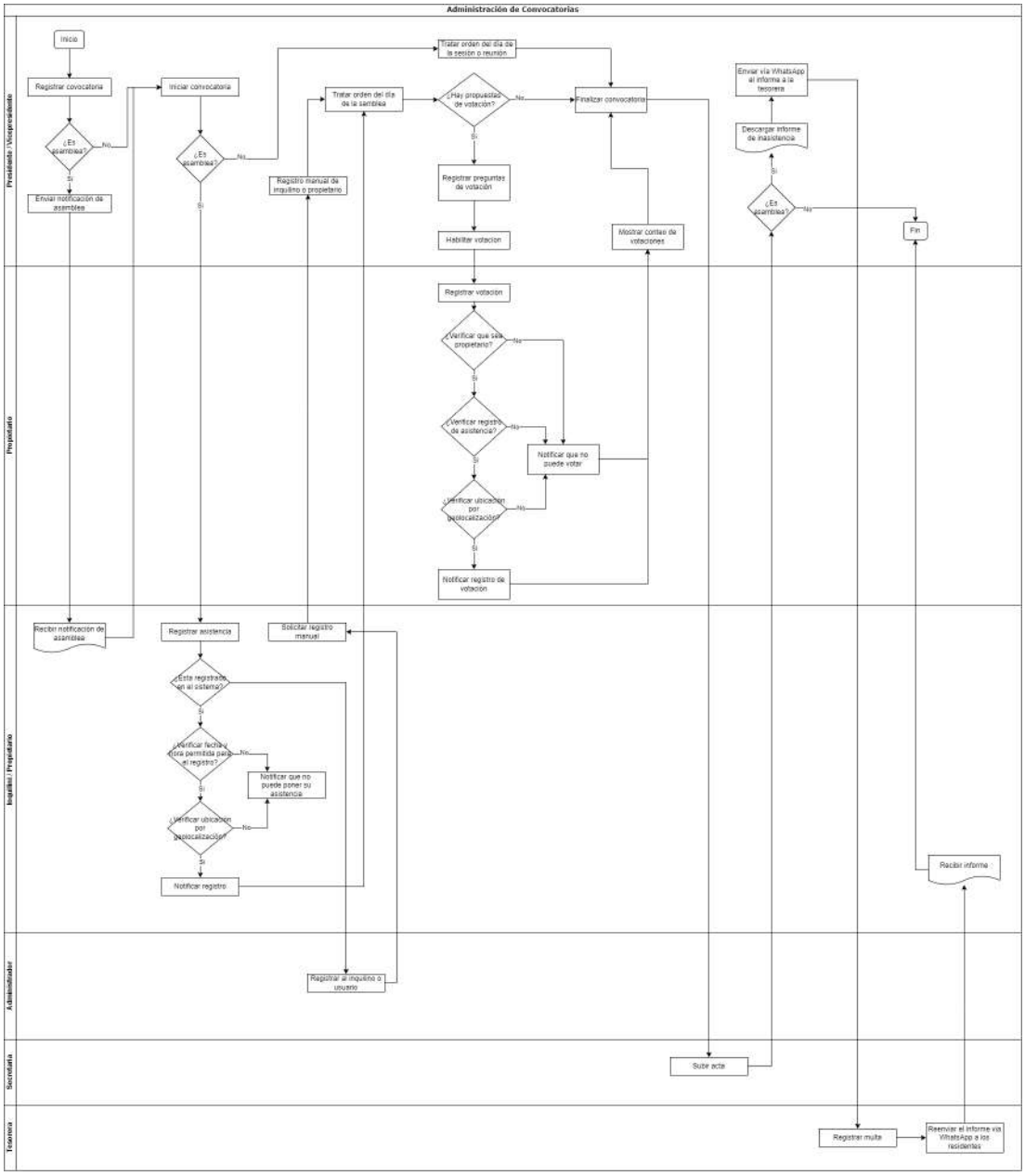


Figura 16. Diagrama de flujo de procesos sistematizado prupuesto de convocatorias

- Proceso de administración financiera.

1. Pagos de obligaciones financieras mensuales

   (a) El inquilino realiza el pago de la mensualidad de sus obligaciones financieras.

   (b) El inquilino envía el comprobante de pago a la tesorera.

   (c) La tesorera verifica el comprobante de pago.

   - Si el comprobante es válido se procede al siguiente proceso.

- Si el comprobante no es válido se notifíca via WhatsApp y se finaliza el proceso.

(d) La tesorera selecciona la residencia del inquilino o propietario.

(e) El sistema verifica hasta que mes abonó el inquilino en su anterior pago.

- Si el inquilino tiene abonos anteriores el sistema coloca el mes siguiente de manera automática y se procede al siguiente paso.
- Si el inquilino no tiene abonos se procede al siguiente paso.

(f) Se registra el pago en el sistema.

(g) Se sube el comprobante de pago al sistema.

(h) Se genera una orden de pago.

(i) Se envía por correo electrónico el respaldo de pago al inquilino o propietario.

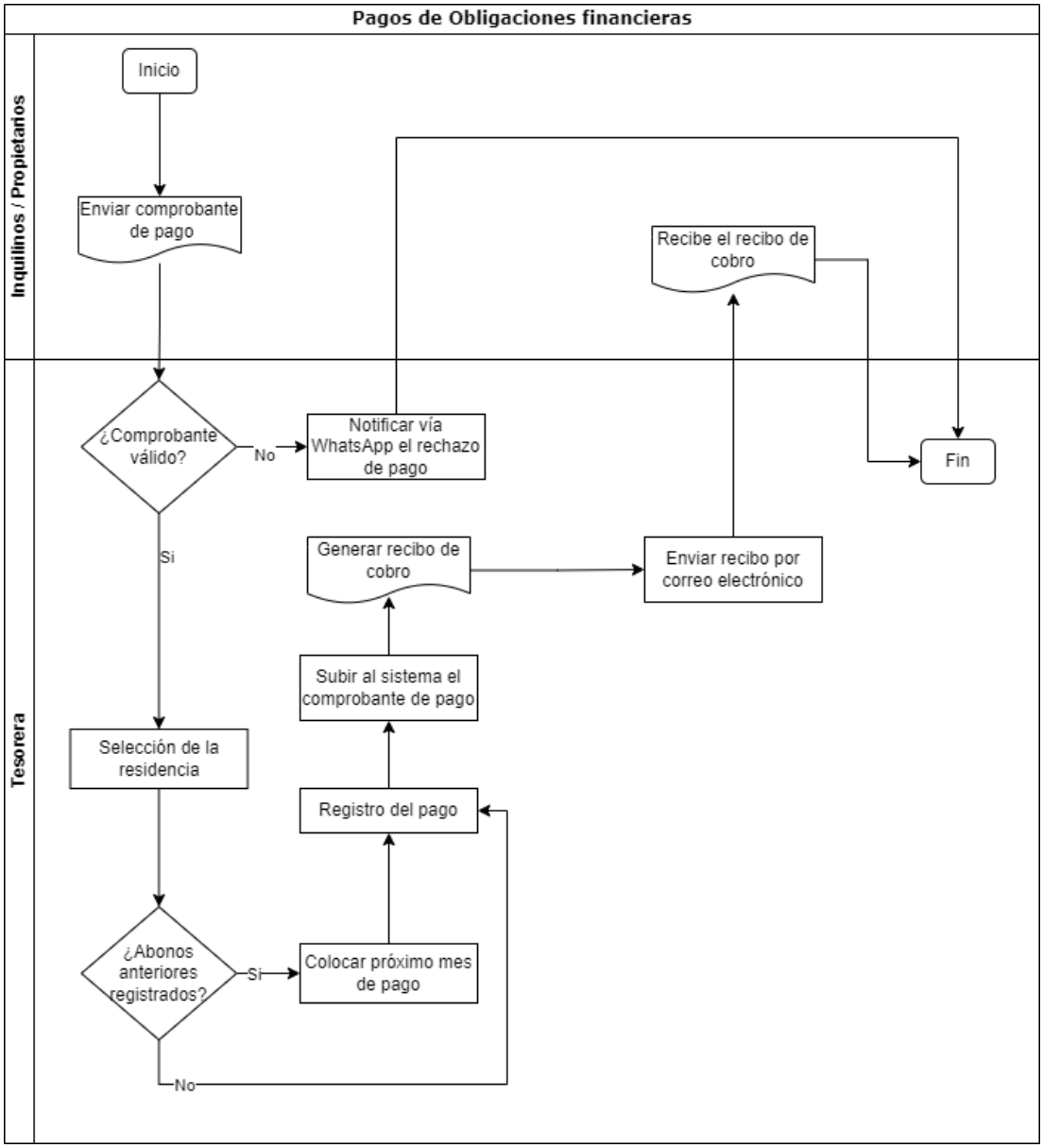


Figura 17. Diagrama de flujo de procesos sistematizado prupuesto de pago de obligaciones financieras mensuales

2. Pagos a proveedores

   (a) El proveedor envía la factura a la tesorera.

   (b) La tesorera verifica la factura.

   - Si la factura es válida se procede al siguiente proceso.
   - Si la factura no es válida se finaliza el proceso.

   (c) La tesorera verifica si el proveedor está registrado en el sistema.

   - Si el proveedor está registrado se procede al siguiente proceso.
   - Si el proveedor no está registrado se le registra en el sistema y se

repite el proceso.

(d) La tesorera registra la factura en el sistema.

(e) Se sube la factura al sistema.

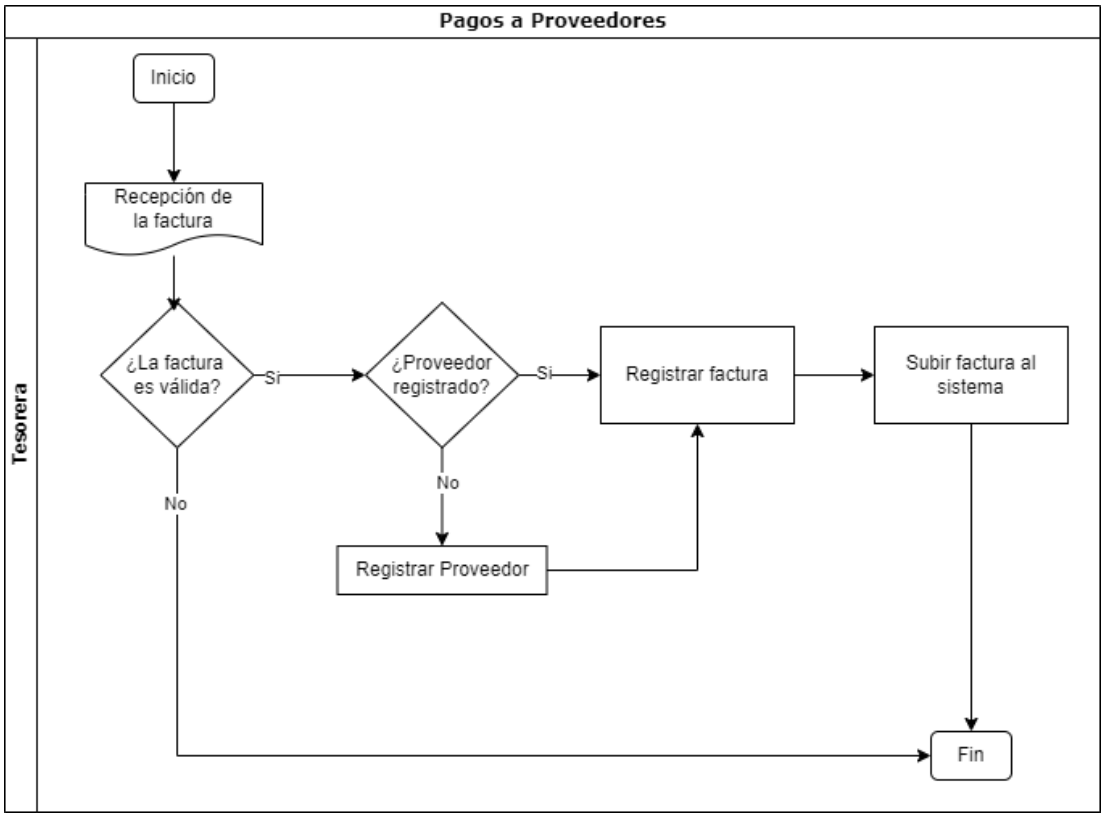


Figura 18. Diagrama de flujo de procesos sistematizado prupuesto de pago a proveedores

3. Multas

(a) Se registra la multa en el sistema.

(b) Se suben evidencias de la multa al sistema.

(c) Se le notifica al propietario o inquilino la multa mediante la aplicación móvil.

(d) El propietario o inquilino envía el comprobante de pago de la multa.

(e) La tesorera verifica el comprobante de pago.

- Si el comprobante es válido se procede al siguiente proceso.
- Si el comprobante no es válido se mantiene la multa, se le notifica por la aplicación móvil el rechazo del pago y se finaliza el proceso.

(f) Se actualiza el estado de la multa en el sistema.

(g) Se genera una orden de pago.

(h) El sistema envía por correo electrónico el respaldo de pago al propietario o inquilino.

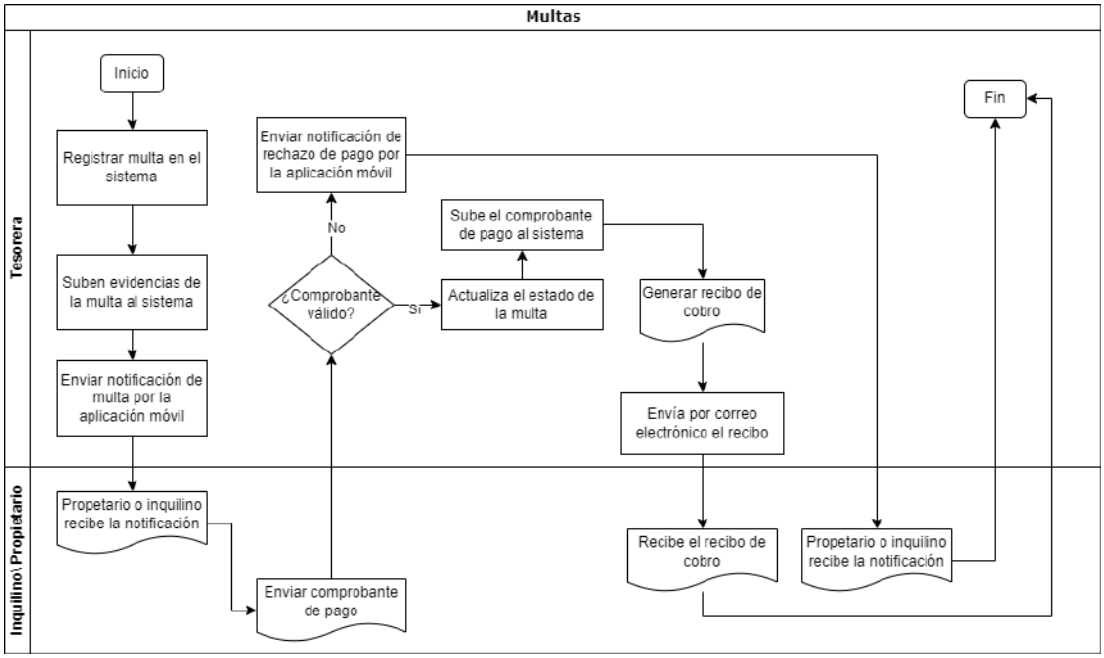


Figura 19. Diagrama de flujo de procesos sistematizado prupuesto de multas

b. Arquitectura

Para el desarrollo de la propuesta se utilizó una arquitectura cliente-servidor. El cliente permite a los usuarios interactuar con el sistema mediante la aplicación móvil y el sistema web. En donde las personas que son parte de la directiva tendrán acceso al sistema web y los propietarios o inquilinos accederán únicamente a la aplicación móvil. Por otro lado la API que estará alojada en el servidor se encargara de procesar las peticiones de los clientes provenientes de la aplicación móvil y el sistema web, en donde se proveerán de servicios necesarios para satisfacer los requerimientos tal y como se muestra en la Figura 20. Además de ofrecernos la conexión a la base de datos para la persistencia de los datos. La base de datos la cual también se encontrará alojada en el servidor, nos permitirá almacenar y servir la información necesaria para el funcionamiento del sistema. Por último la API se conectará a los servicios de almacenamiento de archivos de AWS S3 para la carga de archivos necesarios para el sistema.

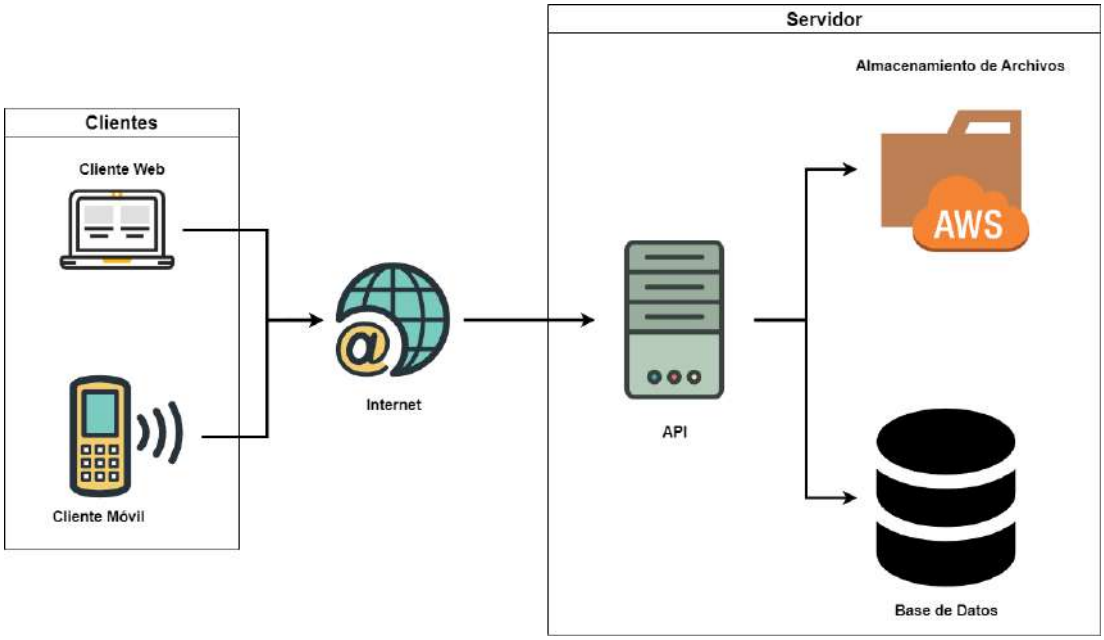


Figura 20. Arquitectura del sistema

c. Prototipado

Con la finalidad de poder presentar a los usuarios finales una vista previa de cómo se verán los sistemas y también como parte de la fase de la metodología RAD se realizaron prototipos de las interfaces de usuario del sistema web y la aplicación móvil.

Prototipo web

En la Figura 21 se muestra la vista de la pantalla de inicio de sesión web de los usuarios miembros de la directiva en donde el usuario podrá ingresar sus credenciales para acceder al sistema y en el caso de ser necesario recuperar su contraseña a través de su correo electrónico.

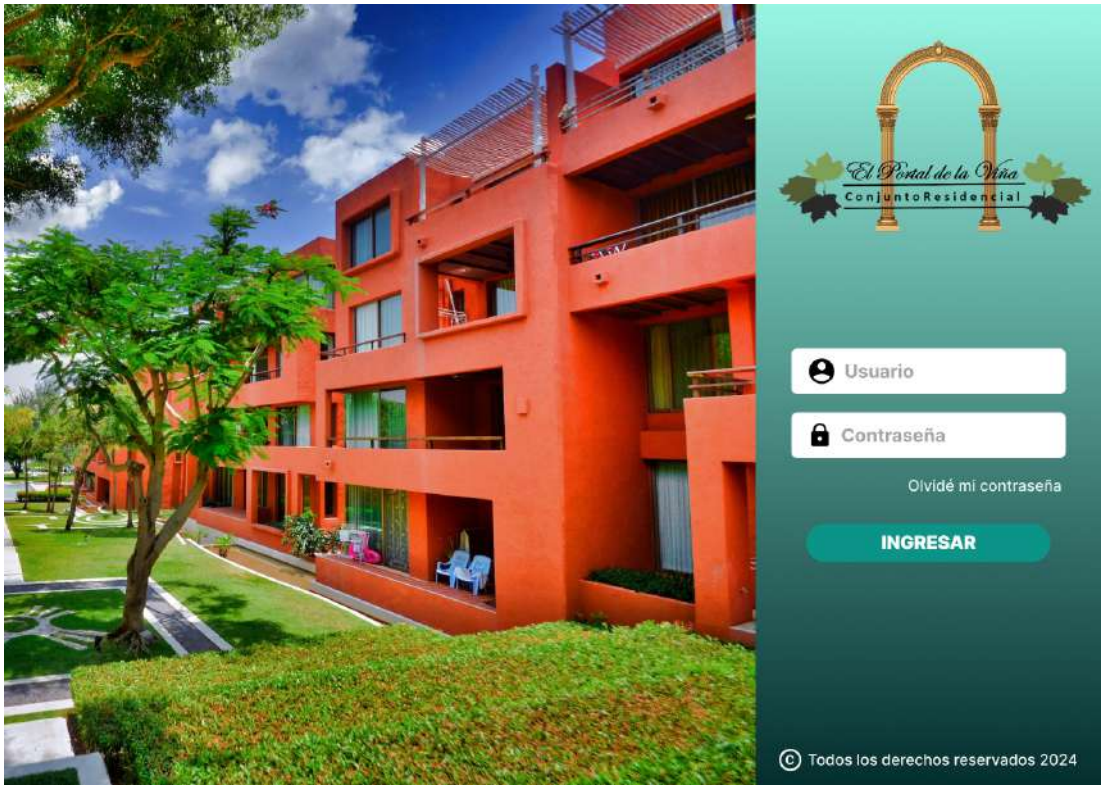


Figura 21. Prototipo de inicio de sesión

En la Figura 22 se muestra la vista de la barra de navegación del sistema web en donde se visualizan las opciones de menú disponibles para los usuarios miembros de la directiva y en la parte inferior de la misma se muestra el nombre del usuario que ha iniciado sesión asi como la opción de cerrar sesión.

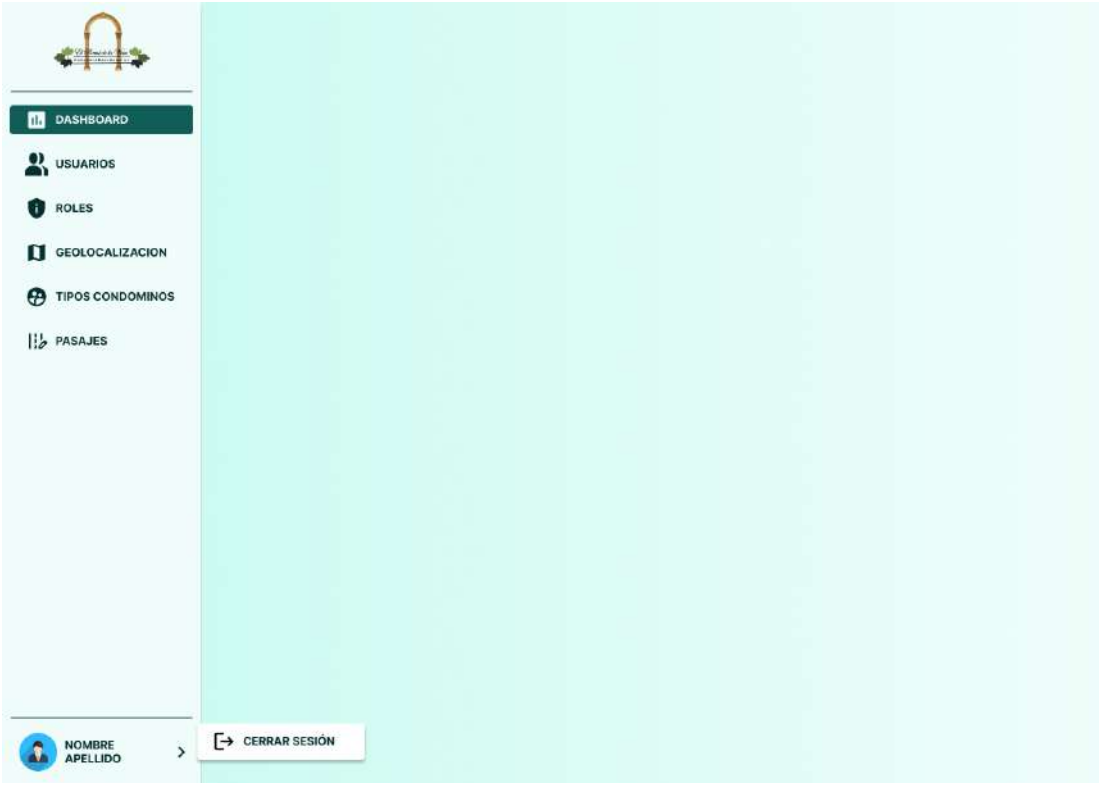


Figura 22. Prototipo de la vista del la barra de navegación

Una vez iniciada la sesión se mostrara una vista de dashboard en donde se visualizará información relevante para los usuarios miembros de la directiva dependiendo de quien haya iniciado sesión, en el caso de la Figura 23 se muestra la vista del dashboard para el administrador, en la Figura 24 se muestra la vista del dashboard para presidente y vicepresidente y por último en la Figura 25 se muestra la vista del dashboard para la tesorera.



Figura 23. Prototipo de la vista del dashboard de administración

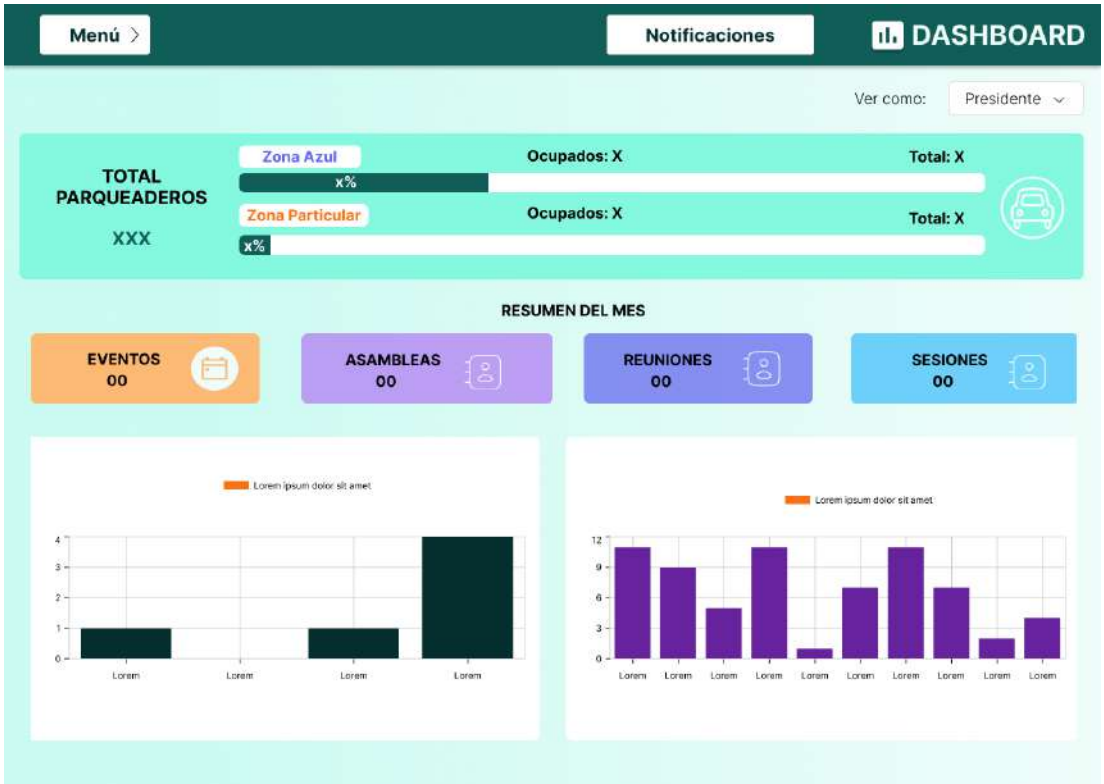


Figura 24. Prototipo de la vista del dashboard de presidencia



Figura 25. Prototipo de la vista del dashboard de tesorería

En la Figura 26 se muestra la vista de la administración de usuarios en donde el administrador podrá visualizar, registrar, inhabilitar o editar la información y roles de los demás usuarios.

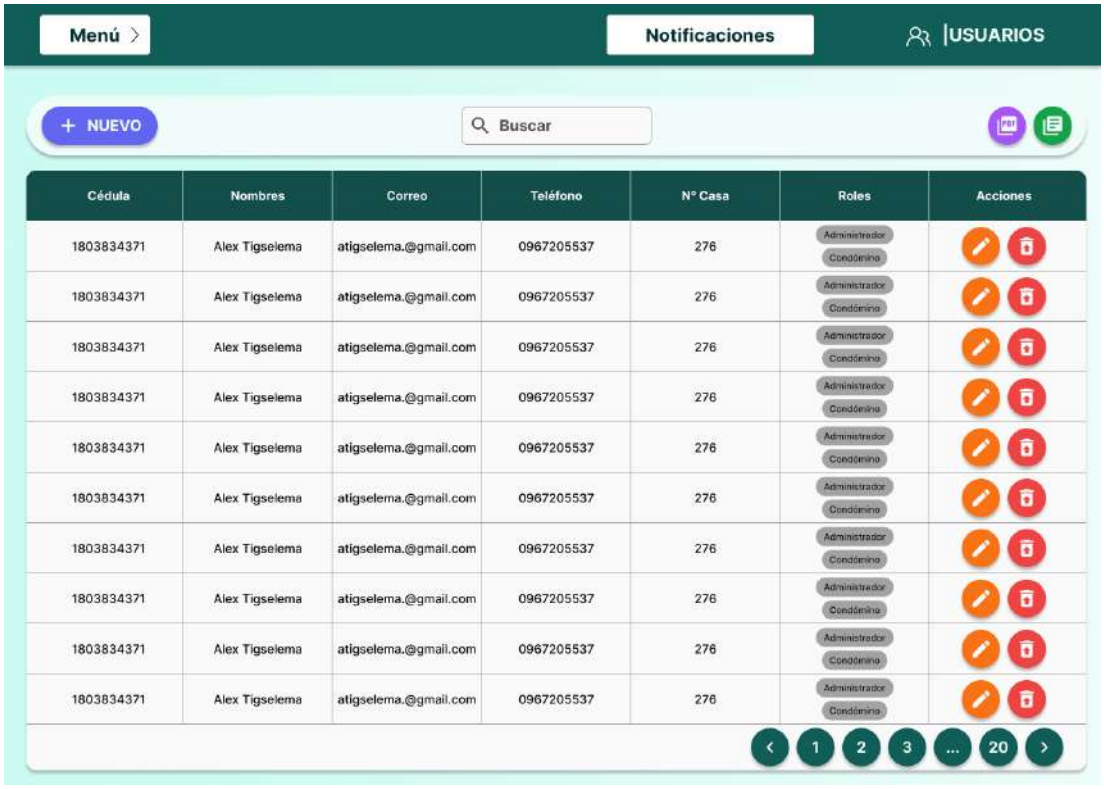


Figura 26. Prototipo de la vista de la administración de usuarios

En la Figura 27 se muestra la vista del formulario para edición o creación de usuarios, en donde el administrador podrá ingresar la información necesaría y seleccionar el rol que cumplirá el usuario a ingresar o editar.

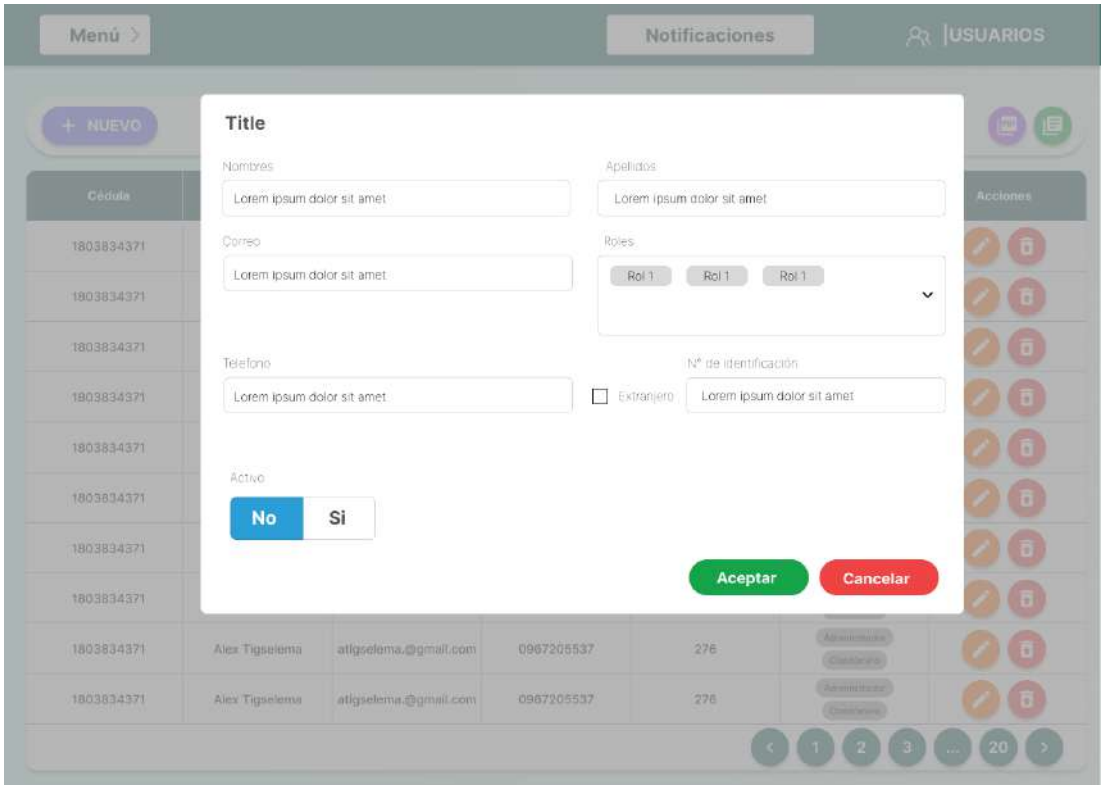


Figura 27. Prototipo de la vista del fomulario para edición o creación de usuarios

En la Figura 28 se muestra la vista de la configuración de la geolocalización en donde el administrador podrá editar la ubicación de las coordenadas y el radio de aceptación mediante la visualización de un mapa del lugar en donde se darán lugar las asambleas.

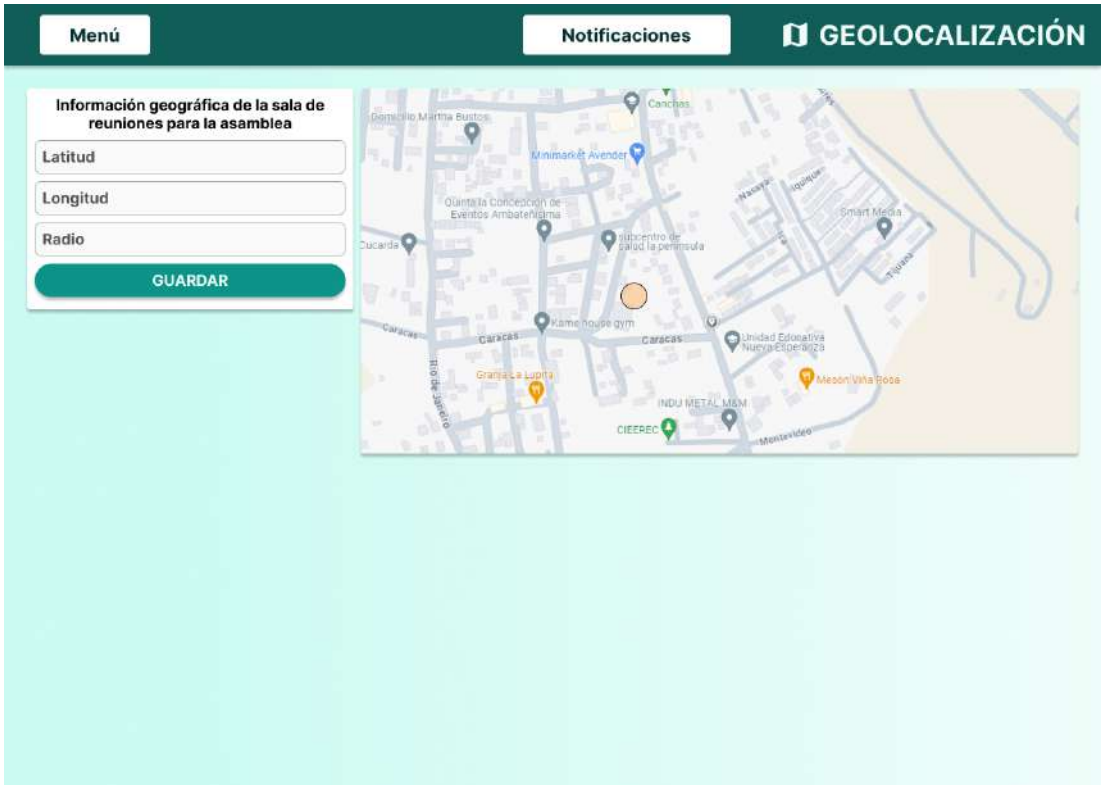


Figura 28. Prototipo de la vista de la configuración de la geolocalización

En la Figura 29 se muestra la vista de la administración de residencias en donde el presidente o vicepresidente podrán visualizar o desplegar el formulario de actualización de residente de cada residencia.

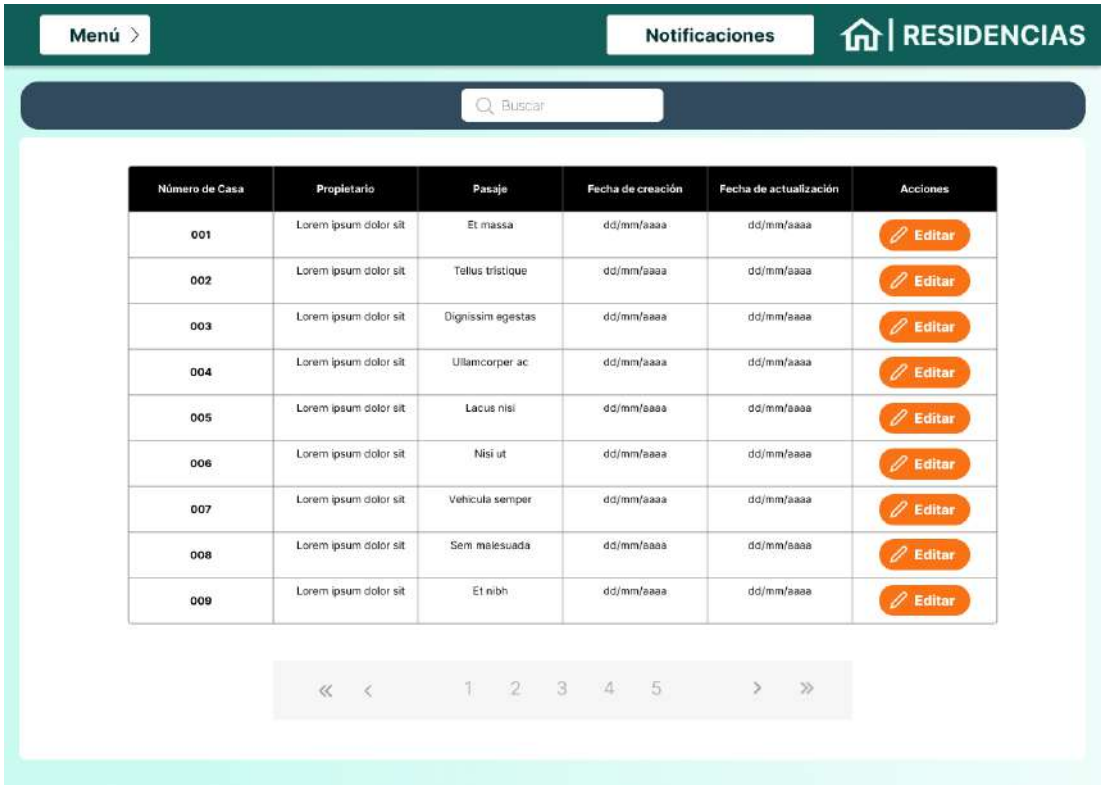


Figura 29. Prototipo de la vista de administración de residencias

En la Figura 30 se muestra la vista de la administración de parqueaderos, en donde mediante la representación de un mapa del condominio el presidente puede visualizar los grupos de parqueaderos por colores en donde los amarillos representan a los parqueaderos partículares y los azules los de zona azul. Por cada grupo se desplegara una vista emergente de los parqueaderos que forman parte de ese grupo junto con su código de numeración y su estado, en donde el presidente podrá seleccionar un parqueadero y desplegar el formulario de cambio de residencia asociada a ese parqueadero. En la parte superior se muestra una barra de navegación para poder cambiar a la vista de visualización de los tipos de parqueaderos Figura 31.

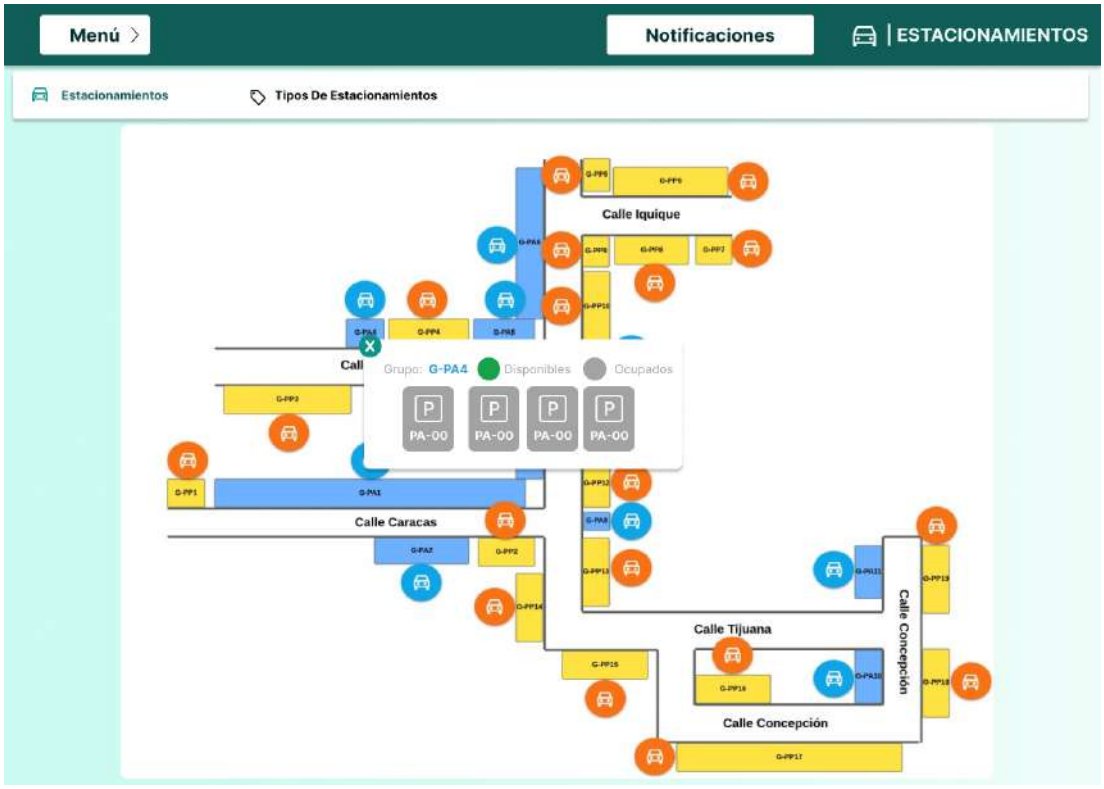


Figura 30. Prototipo de la vista de administración de parqueaderos

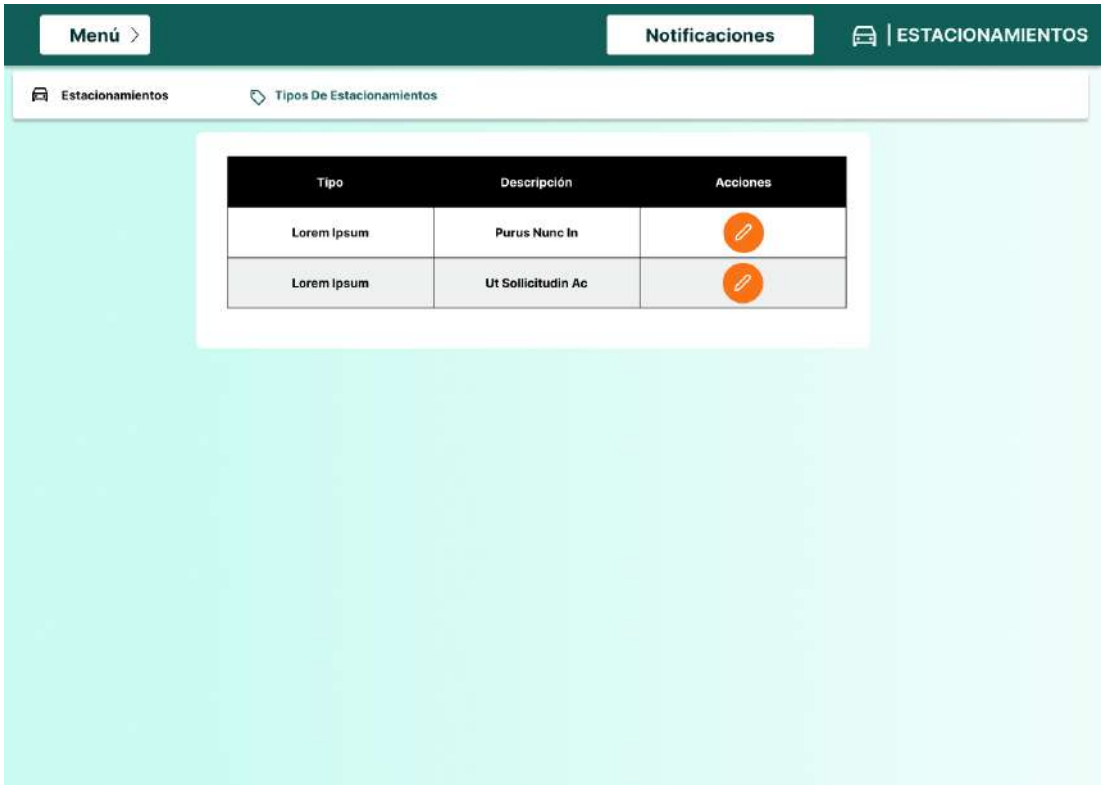


Figura 31. Prototipo de la vista de administración de tipos de parqueaderos

En la Figura 32 se muestra la vista del formulario de cambio de residencia asociada a

un parqueadero, en donde el presidente podrá seleccionar un residente mediante una tabla de búsqueda Figura 33 y seleccionar la residencia que se asociará al parqueadero.

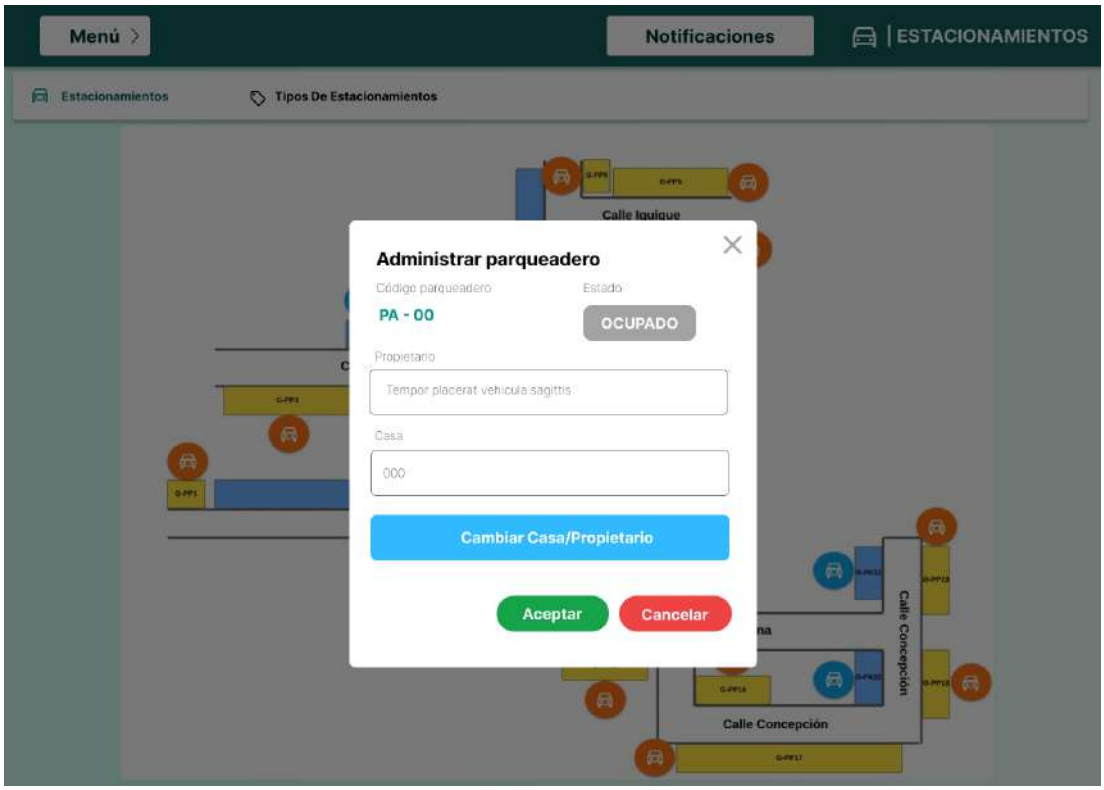


Figura 32. Prototipo de la vista del formulario de cambio de residencia asociada a un parqueadero

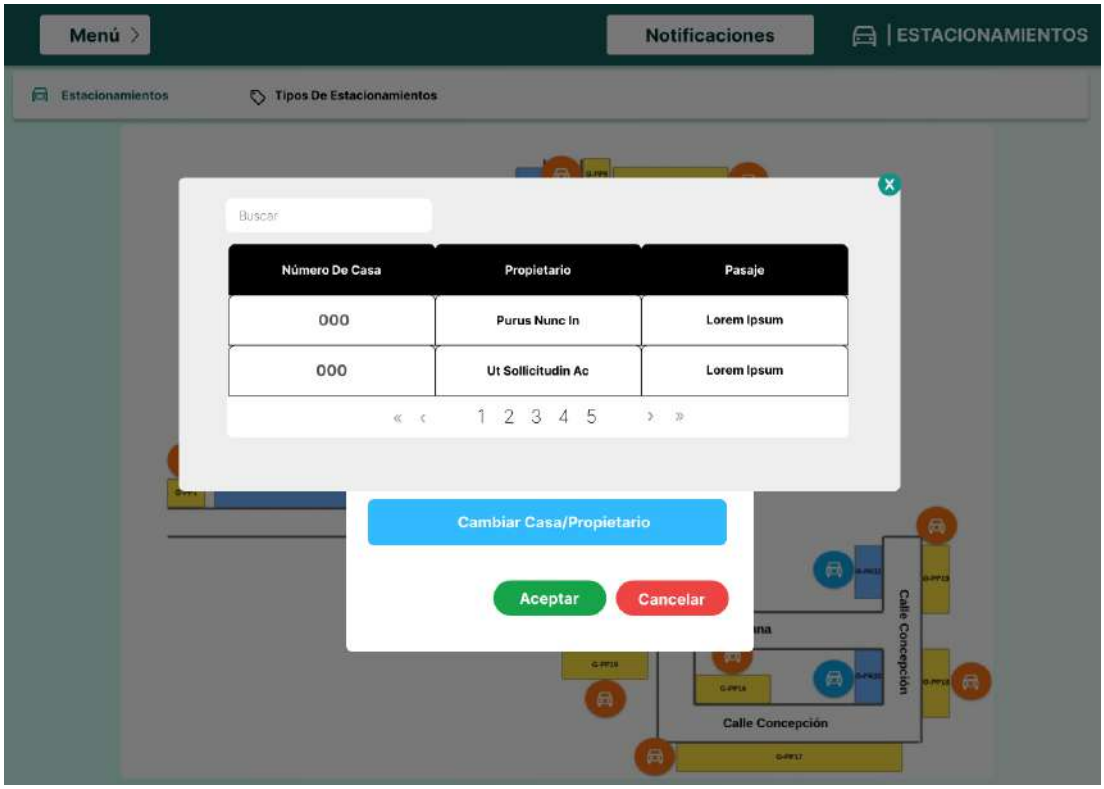


Figura 33. Prototipo de la vista del formulario en la selección de residencia

En la Figura 34 se muestra la vista de la administración de convocatorias en donde el presidente o vicepresidente podrán visualizar las convocatorias junto con la información relevante de cada una, registrar o editar una convocatoria mediante un formulario Figura 35, además de poder desplegar un menú de opciones en donde se podrá visualizar un pista previa de la convocatoria para ser descargada en formato PDF Figura 36, finalizar la convocatoria, administrar la asistencia Figura 37 o administrar las votaciones en el caso de ser una asamblea Figura 38.

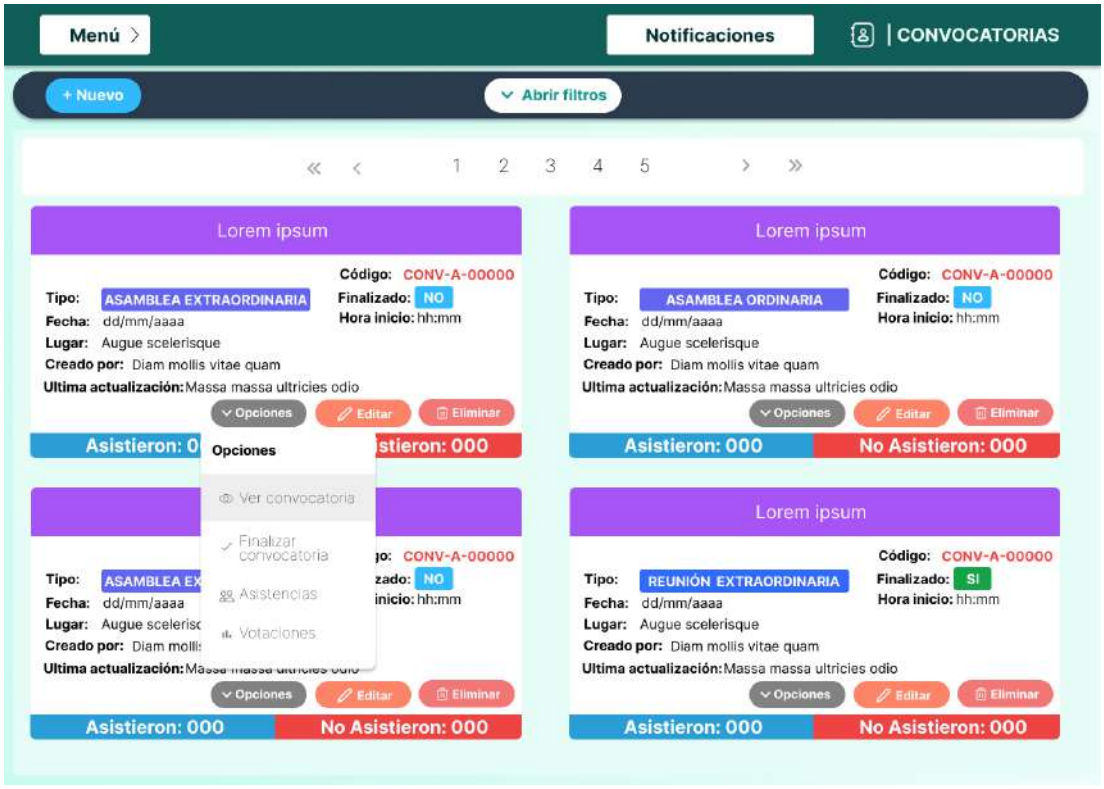


Figura 34. Prototipo de la vista de la administración de convocatorias

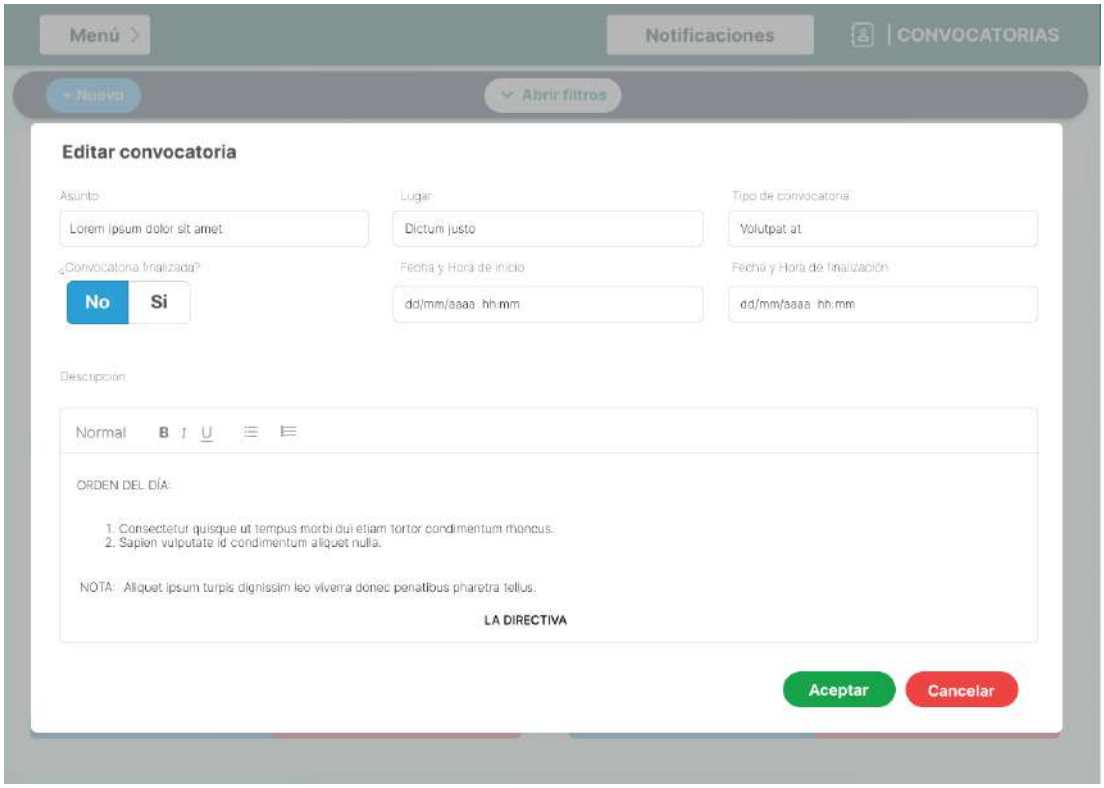


Figura 35. Prototipo de la vista del formulario de de edición o creación de convocatorias

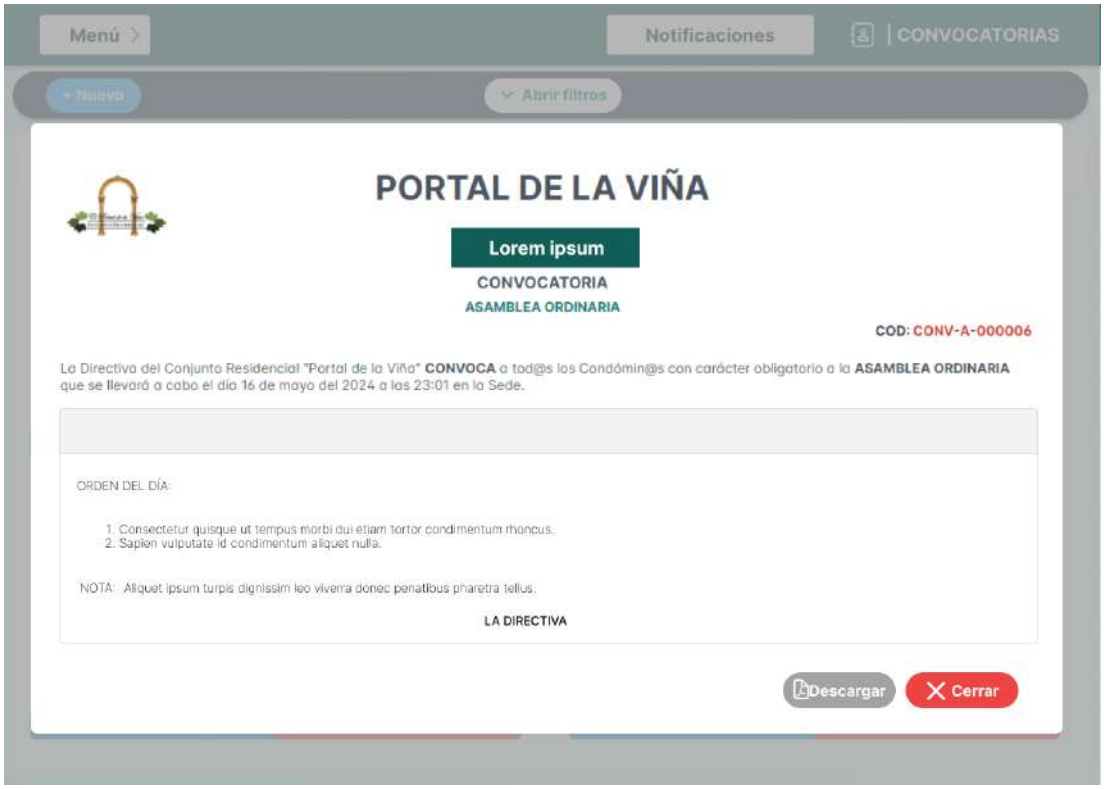


Figura 36. Prototipo de la vista de vista previa a la descarga de la convocatoria

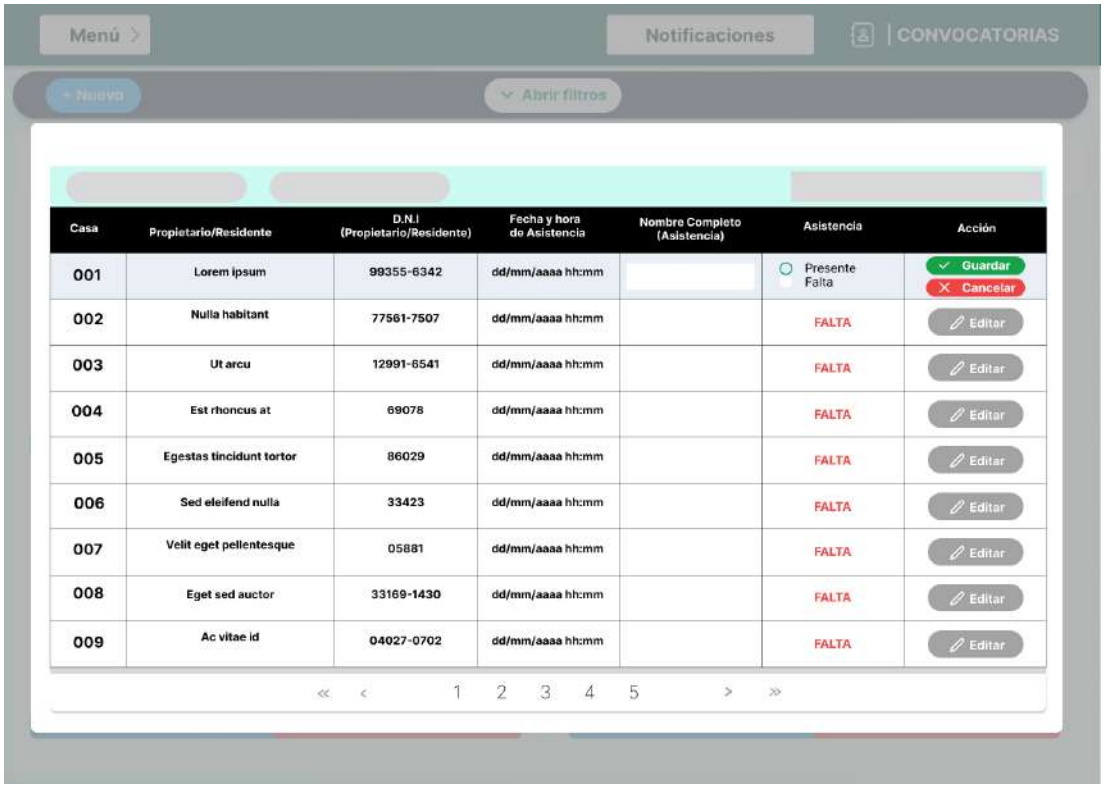


Figura 37. Prototipo de la vista de administración de asistencias en asambleas



Figura 38. Prototipo de la vista de administración de votaciones en asambleas

En tesorería existen ingresos por tres conceptos los cuales son los ingresos casuales, mensuales y las multas, en la Figura 39 se muestra la vista de la administración de los tipos de ingresos mensuales y casuales con las todas las opciones de administración como la creación o edición de cada uno de los registros Figura 40 y en la Figura 41 se muestra la vista de la administración de los tipos de multas en donde también se visualizan todas las opciones de administración.

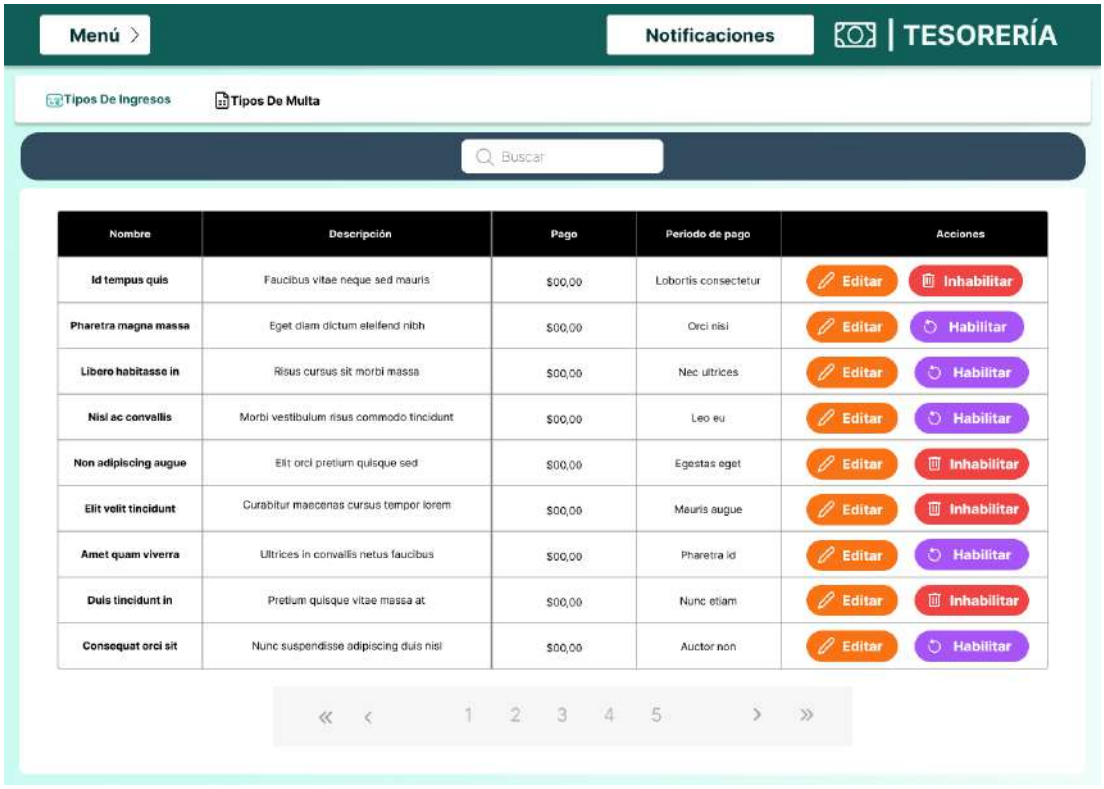


Figura 39. Prototipo de la vista de administración de tipos de ingresos casuales y mensuales

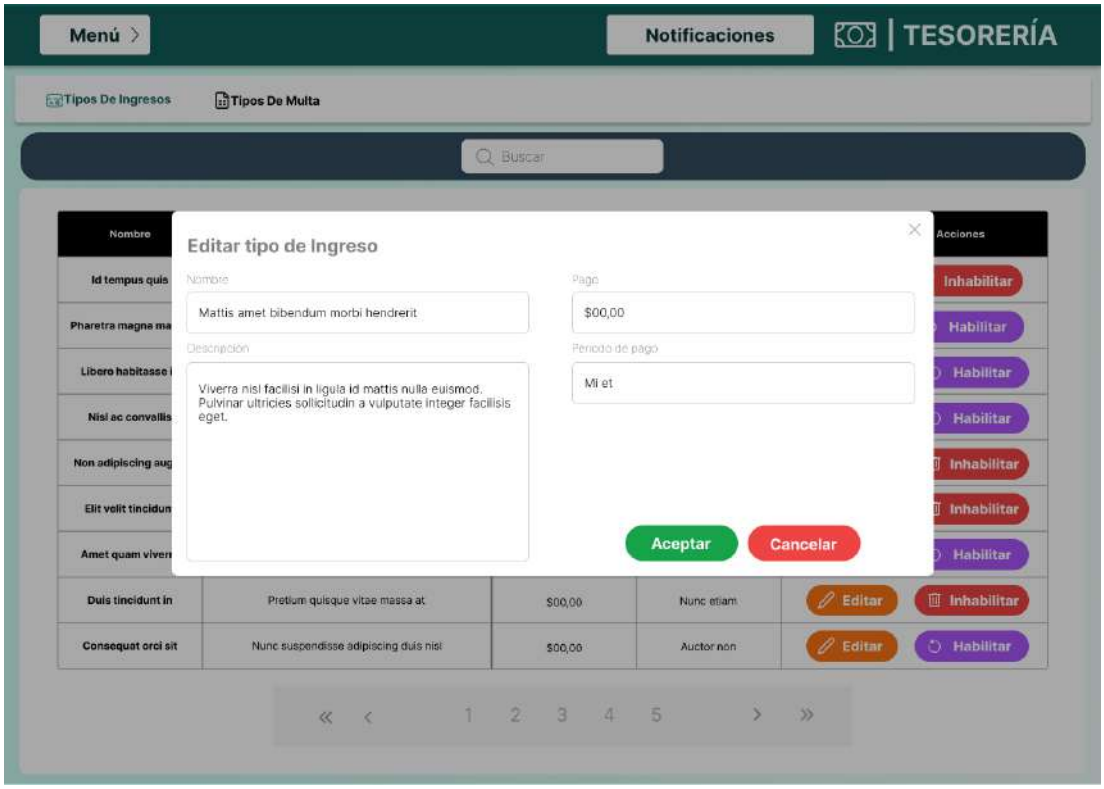


Figura 40. Prototipo de la vista del formulario de creación y edición de tipos de ingresos casuales y mensuales

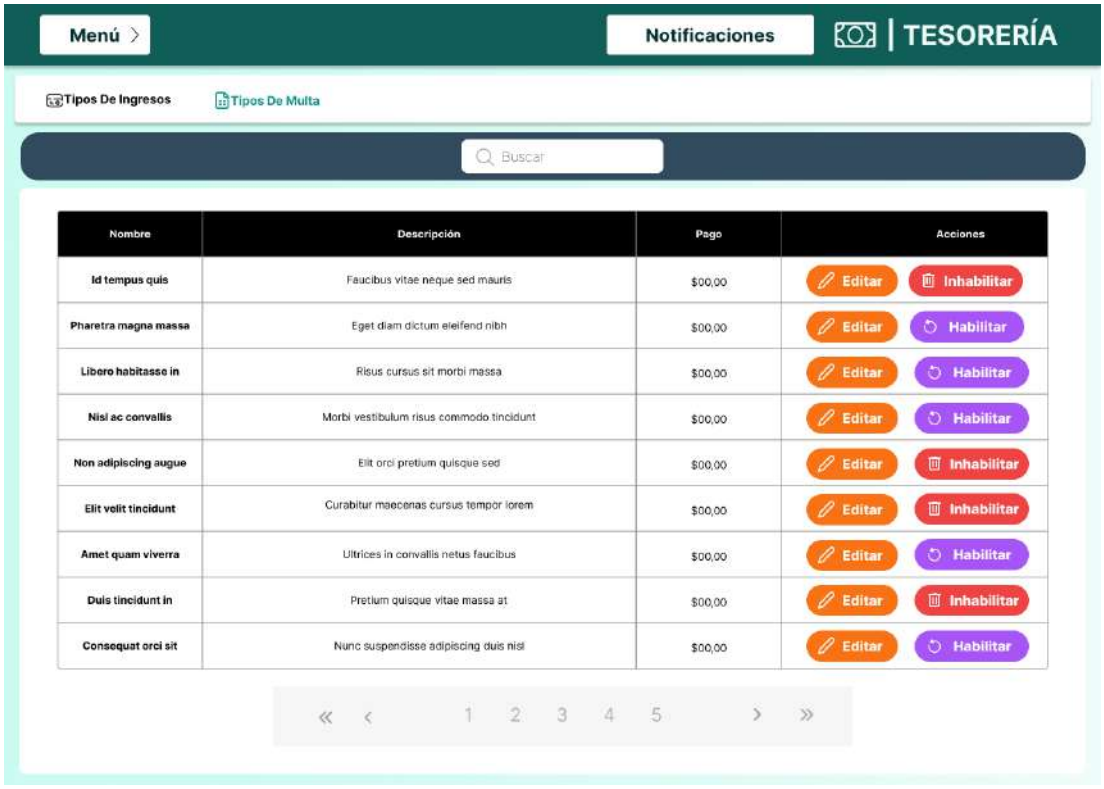


Figura 41. Prototipo de la vista de administración de tipos de multas

Tal como se muestra en la Figura 42 en los ingresos mensuales se visualizan los registros de los pagos realizados por los propietarios o inquilinos de alícuotas o parqueaderos azules junto con una imagen del comprobante de pago por parte de los mismos, para la creación o edición de los registros se despliega un formulario Figura 43 en donde se debe seleccionar la casa a la que se le va a realizar el registro del pago, en donde se colocará de forma automática el próximo mes a pagar y se podrá colocar hasta que fecha se realiza el pago, además de subir el comprobante de pago. Por último en la Figura 44 se muestra la vista del recibo de cobro de los ingresos mensuales la cual puede ser descargada en formato PDF para su firma y envío al propietario o inquilino en el caso de ser necesario.

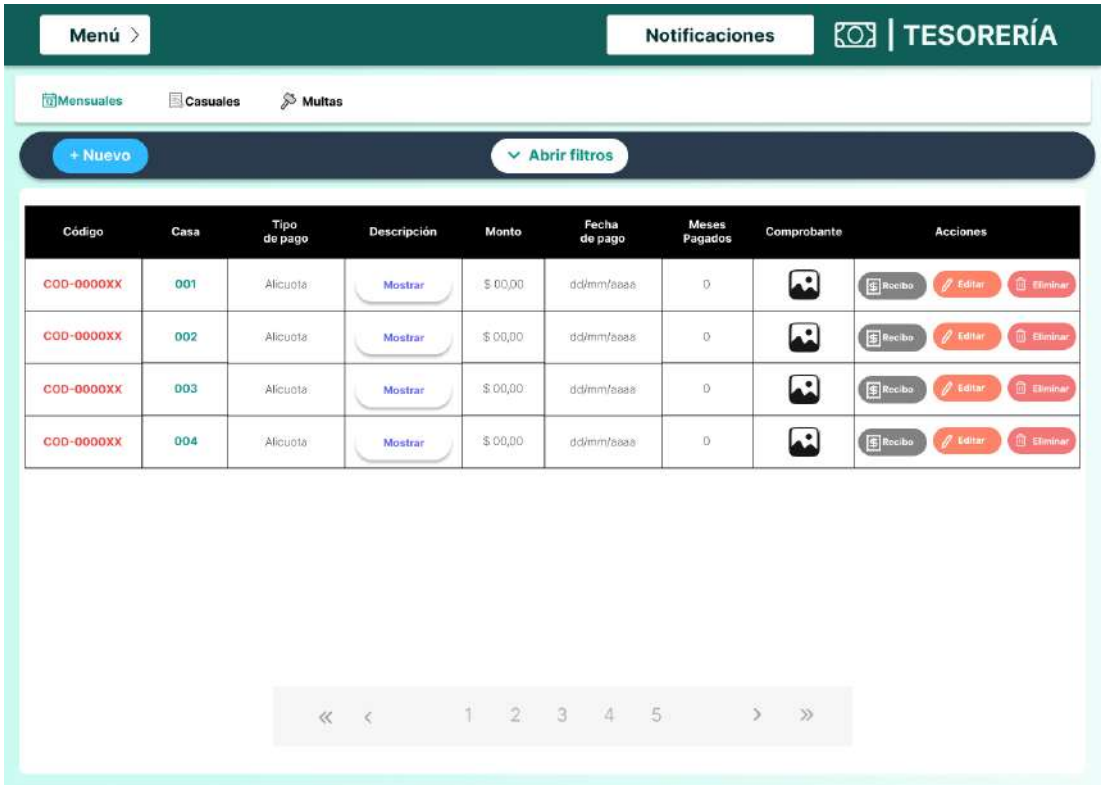


Figura 42. Prototipo de la vista de administración de ingresos mensuales

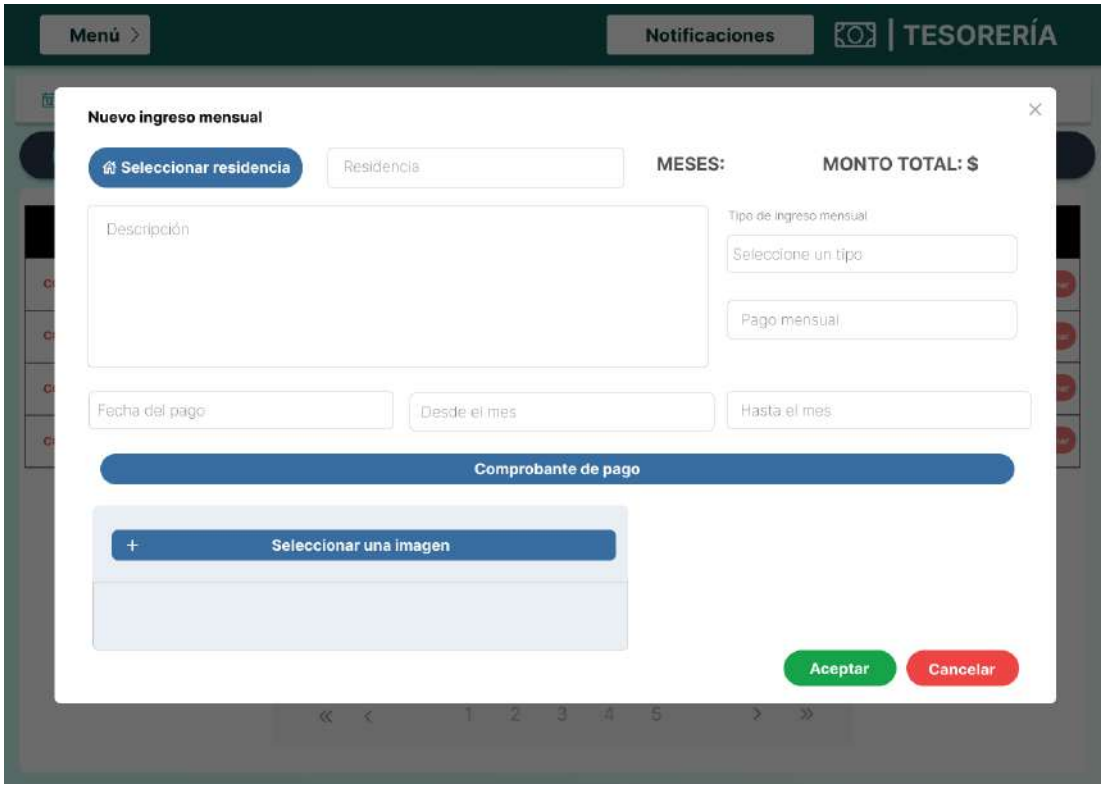


Figura 43. Prototipo de la vista del formulario de creación y edición de ingresos mensuales

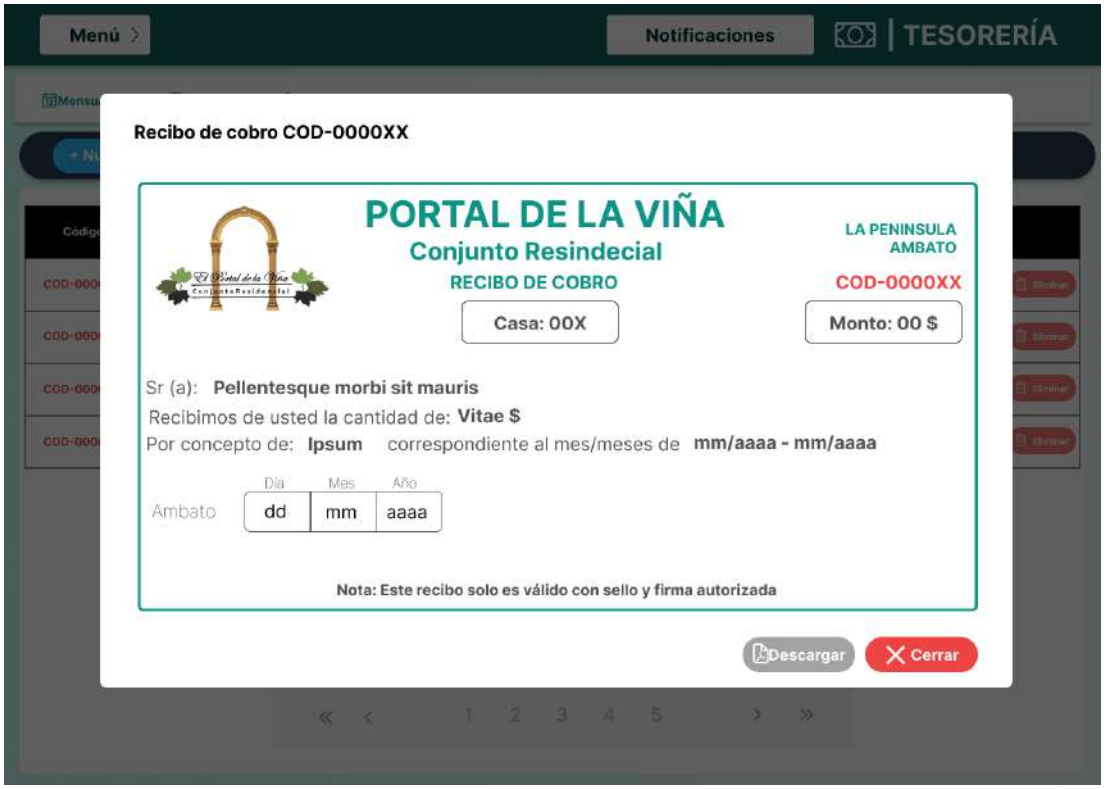


Figura 44. Prototipo de la vista del recibo de cobro generado por el sistema

En el caso de las múltas, son administradas de una forma muy similar a los pagos mensuales salvo que no tienen un rango de fechas de pago, en la Figura 45 se muestra el listado de registros de pagos de multas en los cuales se puede revisar el estado de las mismas para verificar si están pagadas o no, las evidencias de la multa y el comprobante de pago por parte del residente. Para crear o editar un registro se despliega un formulario Figura 46 en donde se selecciona la residencia que será multada, el tipo de multa, junto con la fecha de la infracción, opcionalmente se pueden subir las evidencias y el comprobante de pago si en el caso de que sea una multa que ya se ha pagado para posteriormente generar un recibo de cobro de multa en formato PDF que ya se mostró en el proceso anterior Figura 44.

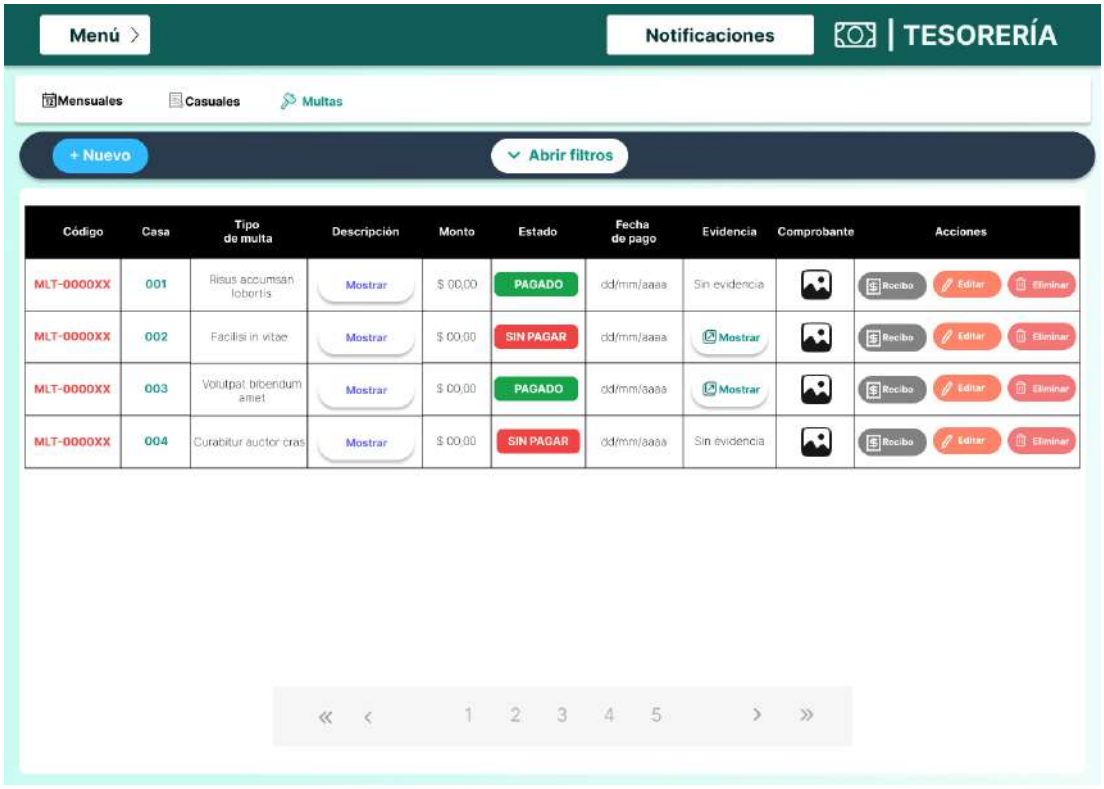


Figura 45. Prototipo de la vista de la administración de multas

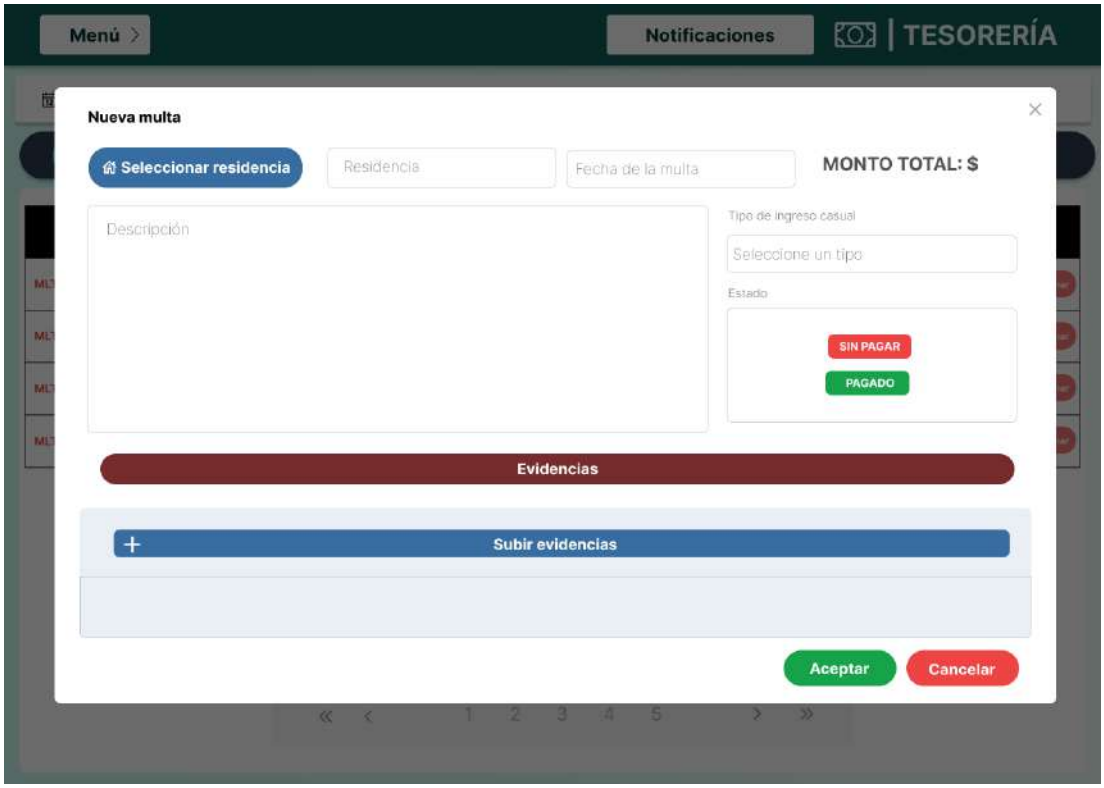


Figura 46. Prototipo de la vista del formulario de creación y edición de ingresos multas

Para el registro de egresos en la Figura 47 se muestra la vista administradora de los egresos en los cuales se pueden visualizar, registrar o editar los registros de los pagos a proveedores o eliminarlos en caso de ser necesario. Posteriormente a este proceso se tienen la administración de los tipos de egresos Figura 48 y también la administración de los proveedores Figura 49 en donde en ambos casos se pueden visualizar, registrar, editar o inhabilitar. Continuando con los egresos en la Figura 50 se muestra la vista del formulario para edición o registro en donde se necesita seleccionar el proveedor, el tipo de egreso, la fecha de la factura, el monto registrado en la factura, la factura o comprobante expendida por el proveedor y una descripción del egreso.

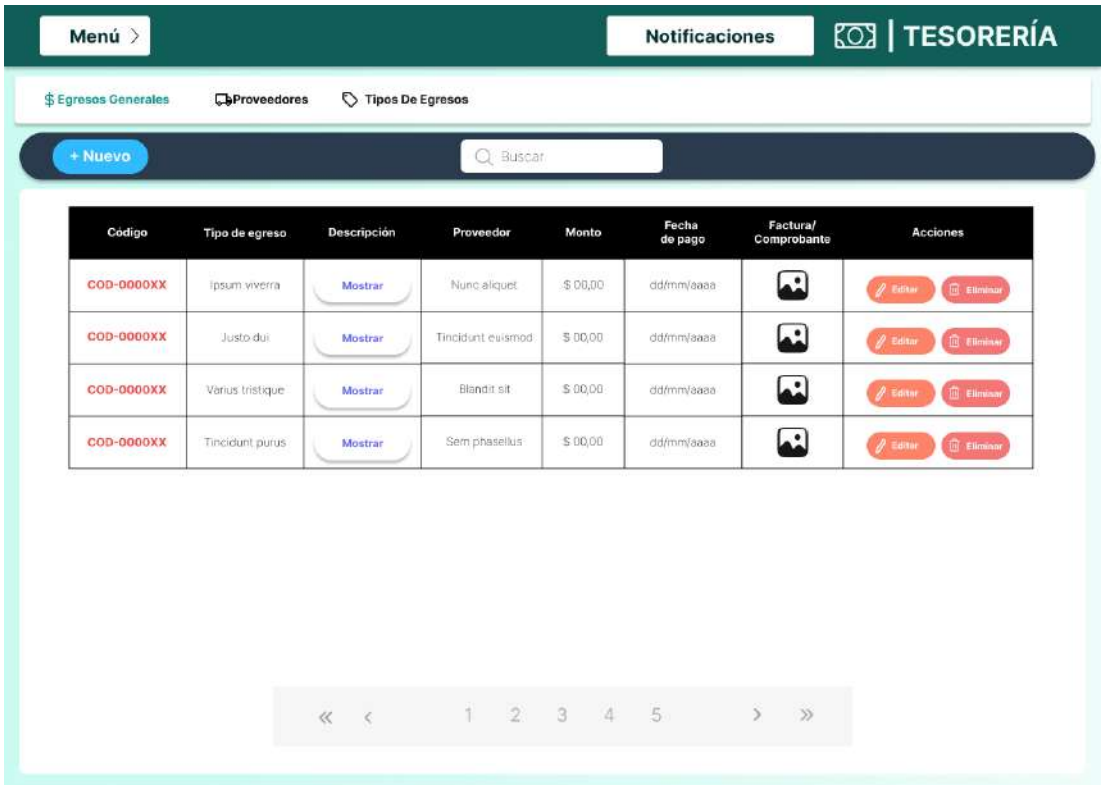


Figura 47. Prototipo de la vista de la administración de egresos

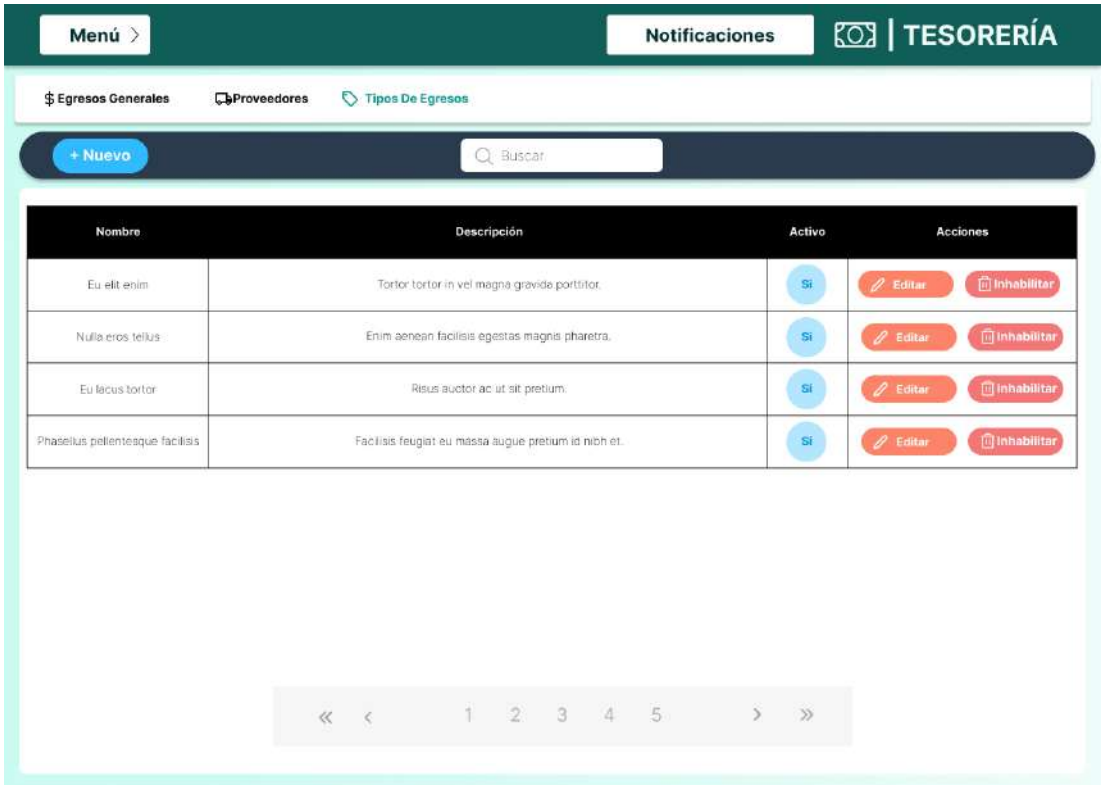


Figura 48. Prototipo de la vista de la administración de tipos de egresos

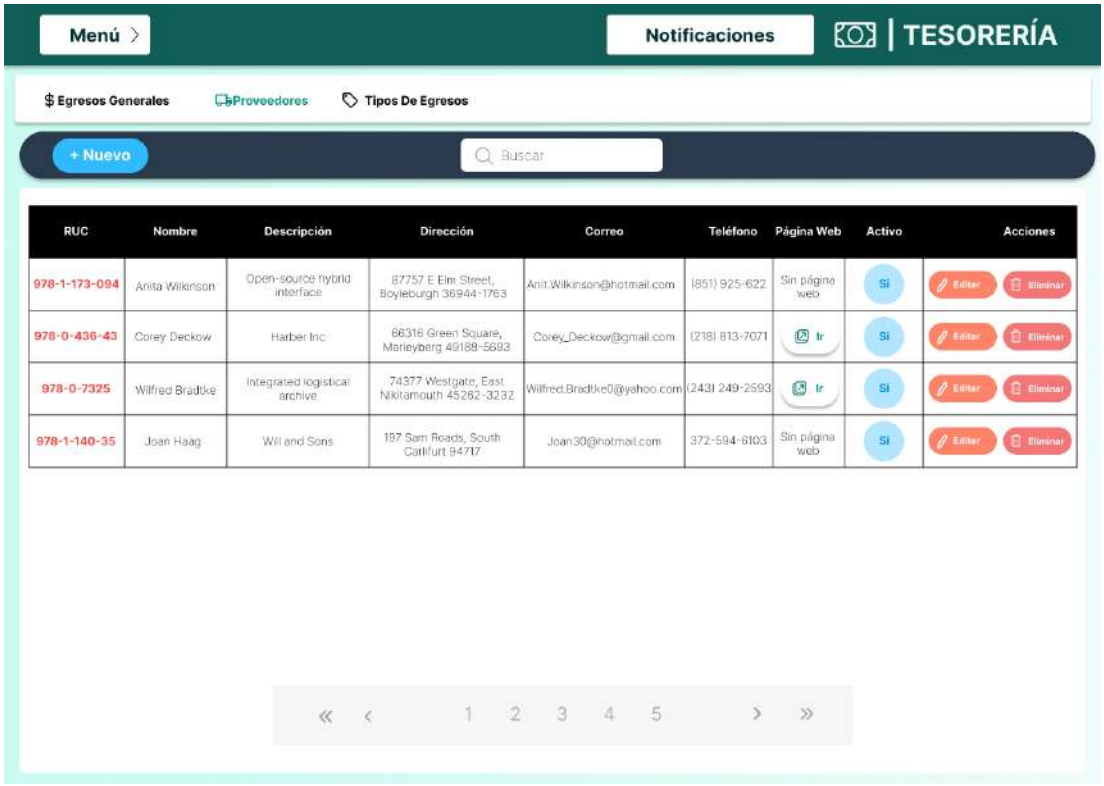


Figura 49. Prototipo de la vista de la administración de proveedores

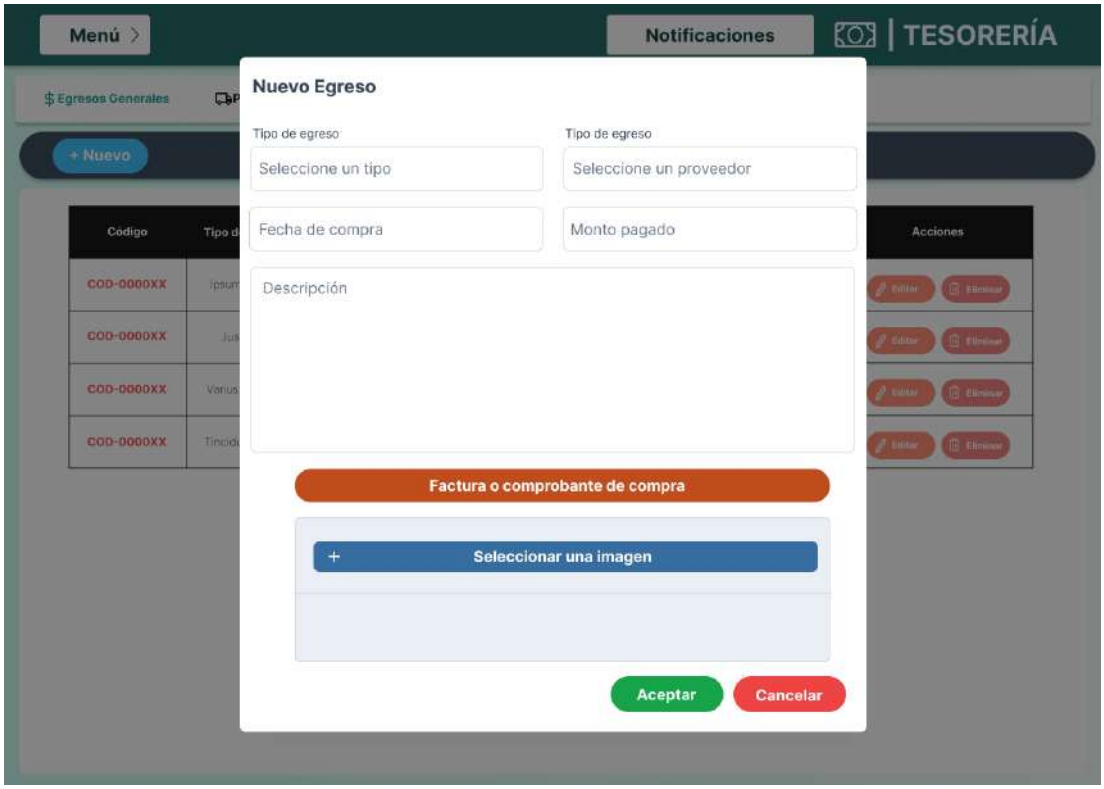


Figura 50. Prototipo de la vista del formulario de creación y edición de egresos

Prototipo móvil

Para el inicio de sesión en la aplicación móvil tan como se muestra en la Figura 51 el usuario debe proporcionar sus credenciales de acceso, además de también contar con la opción de poder recuperar la contraseña mediante su correo electrónico.



Figura 51. Prototipo de la vista del inicio de sesión

En la parte inferior se puede ver la barra de navegación con las opciones del menú de la aplicación Figura 52 en donde se visualizan las opciones de eventos, hogar, asambleas y perfil.

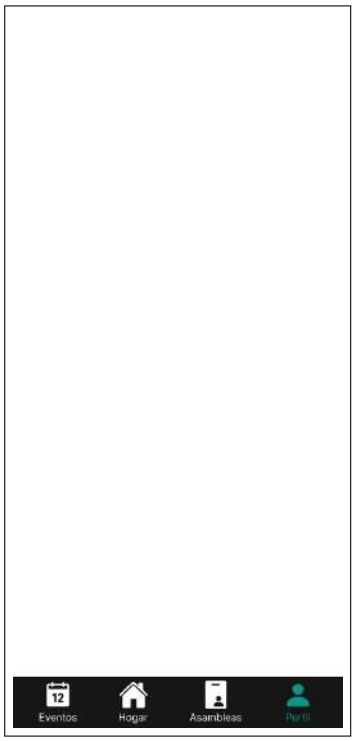


Figura 52. Prototipo de la vista de la barra de navegación

Al iniciar sesión como primera vista se mostrarán los eventos próximos a realizarse en el condominio Figura 53los cuales pueden ser eventos sociales o eventos de asambleas, adicionalmente se podrá ver de una forma más detallada la información de cada evento como se muestra en la Figura 54 en donde se visualiza la fecha, hora, lugar y descripción del evento.



Figura 53. Prototipo de la vista de eventos próximos



Figura 54. Prototipo de la vista detallada de un evento

En la sección de asambleas se mostrará únicamente la asamblea que se dará lugar en el día actual Figura 55, en donde se podrá visualizar la información más relevante como la fecha de inicio, la ficha límite pára poder registrar la asistencia, el lugar en donde se dará la asamblea y la opción de poder registrar o votar en la asamblea.

En el caso de querer registrar la asistencia se mostrará la o las residencias asociadas al usuario que ha iniciado sesión en donde tendrá que registrar su asistencia por cada una de sus residencias como se muestra en la Figura 56. También se mostrarán en la parte superior indicaciones a tomar en cuenta para el registro de asistencia en la asamblea.

Por otro lado en el caso de que se quiera votar en la asamblea se mostrarán las preguntas a votar habilitadas por el presidente o vicepresidente en donde el usuario podrá decidir si esta a favor o en contra de la pregunta tal como se muestra en la Figura 57. Al igual que en el registro también para las votaciones se mostrarán indicaciones en la parte superior.



Figura 55. Prototipo de la vista de asambleas del día actual

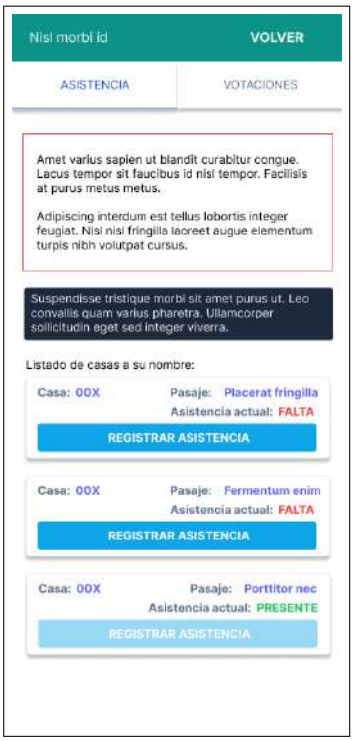


Figura 56. Prototipo de la vista del registro de asistencia en asambleas



Figura 57. Prototipo de la vista del perfil

Por último el usuario podrá visualizar su perfil en donde podrá ver su información personal además de las opciones de cierre de sesión y cambio de contraseña Figura 58.



Figura 58. Prototipo de la vista del registro de voto en asambleas

3.5.3 Construcción

En la siguiente sección se detalla la construcción de la propuesta de los sistemas de administración de condominios, en donde se detallan los procesos que se siguieron para el desarrollo de los sistemas.

a. Backend

- ***Dependencias de la API.*** Las dependencias necesarias para el sistema de la API se utilizó Maven como gestor de dependencias. En la Figura 59 se muestra el archivo *pom.xml* en donde se establecierón las dependencias necesarias para el desarrollo de la API. El listado de todas las dependencias usadas en la API se detallan en el Anexo E.

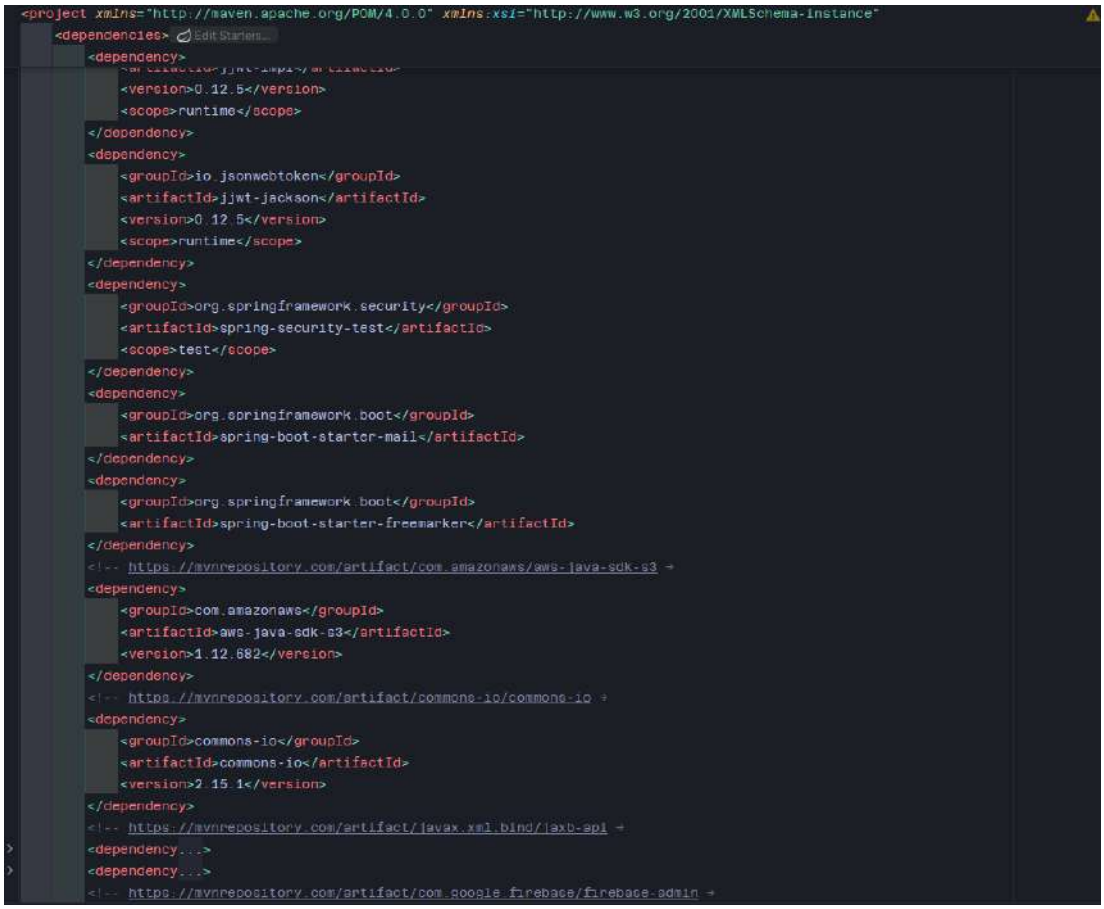


Figura 59. Archivo *pom.xml* con las dependencias Maven de la API

- ***Configuración de las propiedades de la API.*** Para la configuración de las propiedades de la API se utilizó el archivo *application.properties* en donde se establecieron las propiedades de conexión a la base de datos, el puerto en el que se ejecutará la API, la configuración del envío de mensajes via Simple Mail

Transfer Protocol (SMTP), el archivo de configuración de Firebase y las variables de configuración de los servicios de almacenamiento de archivos de Amazon Web Services (AWS) Amazon Simple Storage Service (S3) como se muestra en la Figura 60.

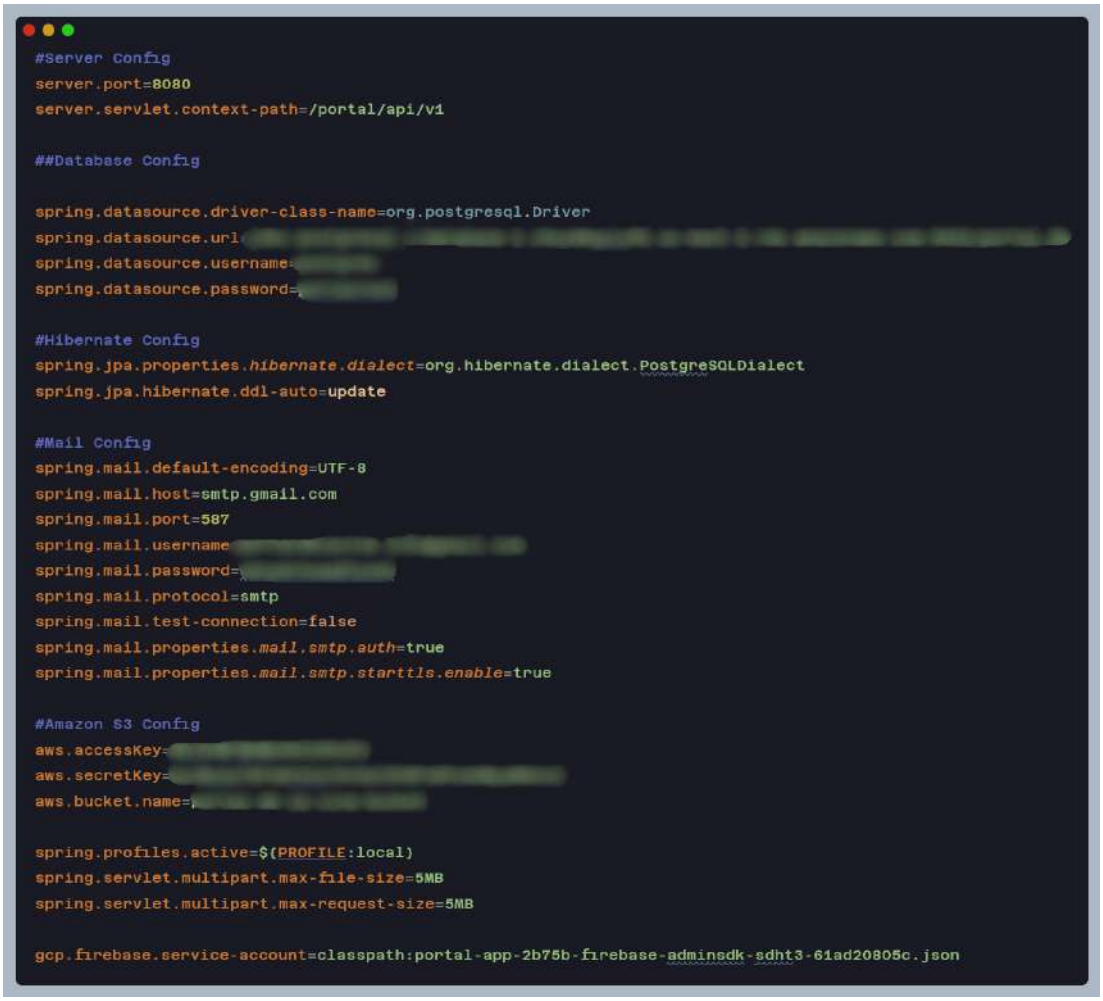
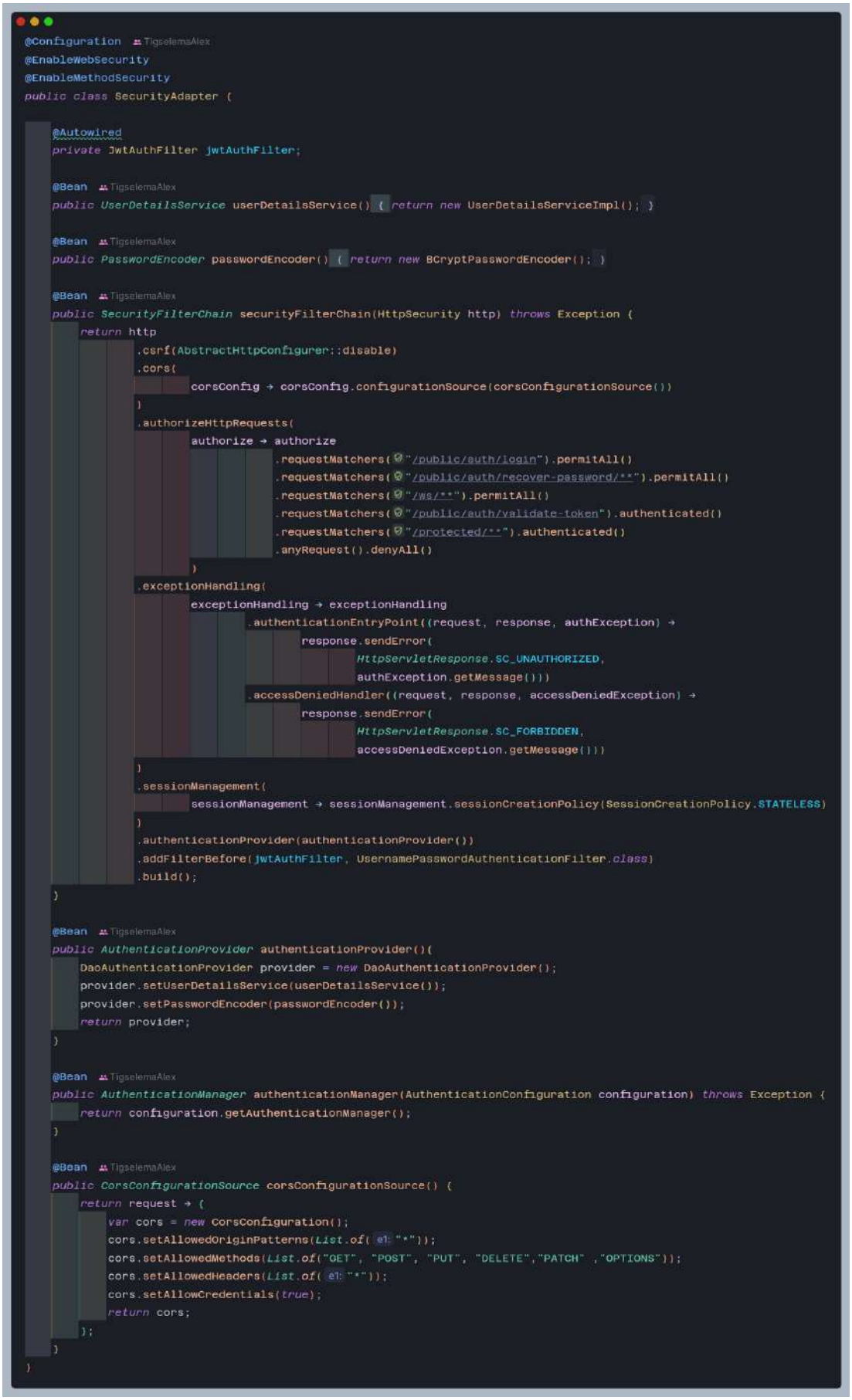


Figura 60. Código de configuración de propiedades de la API

- ***Configuración de la seguridad de la API.*** Como parte de la configuración de la seguridad de la API se utilizó Spring Security en donde se establecierón las reglas de seguridad para el acceso a los puntos de entrada expuestos en la APi mediante el uso de Json Web Token (JWT) como mecanismo de autenticación y autorización. Adicionalmente se configurarón los Cross-Origin Resource Sharing (CORS) para permitir el acceso a la API desde la aplicación móvil y el sistema web como se muestran en la Figura 61.



```java
@Configuration  TigselemaAlex
@EnableWebSecurity
@EnableMethodSecurity
public class SecurityAdapter {

    @Autowired
    private JwtAuthFilter jwtAuthFilter;

    @Bean  TigselemaAlex
    public UserDetailsService userDetailsService() { return new UserDetailsServiceImpl(); }

    @Bean  TigselemaAlex
    public PasswordEncoder passwordEncoder() { return new BCryptPasswordEncoder(); }

    @Bean  TigselemaAlex
    public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
        return http
                .csrf(AbstractHttpConfigurer::disable)
                .cors(
                        corsConfig -> corsConfig.configurationSource(corsConfigurationSource())
                )
                .authorizeHttpRequests(
                        authorize -> authorize
                                .requestMatchers("/public/auth/login").permitAll()
                                .requestMatchers("/public/auth/recover-password/**").permitAll()
                                .requestMatchers("/ws/**").permitAll()
                                .requestMatchers("/public/auth/validate-token").authenticated()
                                .requestMatchers("/protected/**").authenticated()
                                .anyRequest().denyAll()
                        )
                .exceptionHandling(
                        exceptionHandling -> exceptionHandling
                                .authenticationEntryPoint((request, response, authException) ->
                                        response.sendError(
                                                HttpServletResponse.SC_UNAUTHORIZED,
                                                authException.getMessage()))
                                .accessDeniedHandler((request, response, accessDeniedException) ->
                                        response.sendError(
                                                HttpServletResponse.SC_FORBIDDEN,
                                                accessDeniedException.getMessage()))
                )
                .sessionManagement(
                        sessionManagement -> sessionManagement.sessionCreationPolicy(SessionCreationPolicy.STATELESS)
                )
                .authenticationProvider(authenticationProvider())
                .addFilterBefore(jwtAuthFilter, UsernamePasswordAuthenticationFilter.class)
                .build();
    }

    @Bean  TigselemaAlex
    public AuthenticationProvider authenticationProvider(){
        DaoAuthenticationProvider provider = new DaoAuthenticationProvider();
        provider.setUserDetailsService(userDetailsService());
        provider.setPasswordEncoder(passwordEncoder());
        return provider;
    }

    @Bean  TigselemaAlex
    public AuthenticationManager authenticationManager(AuthenticationConfiguration configuration) throws Exception {
        return configuration.getAuthenticationManager();
    }

    @Bean  TigselemaAlex
    public CorsConfigurationSource corsConfigurationSource() {
        return request -> {
            var cors = new CorsConfiguration();
            cors.setAllowedOriginPatterns(List.of( e1: "*"));
            cors.setAllowedMethods(List.of("GET", "POST", "PUT", "DELETE","PATCH" ,"OPTIONS"));
            cors.setAllowedHeaders(List.of( e1: "*"));
            cors.setAllowCredentials(true);
            return cors;
        };
    }
}
```

Figura 61. Código de configuración de seguridad de la API

- ***Acceso y persistencia de datos.*** Para el acceso y persistencia de datos se utilizó Spring Data JPA. Spring Data JPA es un proyecto de Spring que facilita la creación de tablas en la base de datos y la realización de operaciones CRUD (Create, Read, Update, Delete) sobre las mismas. Para lograr esto Spring Data JPA hace uso del mapeo objeto-relacional (ORM) de Hibernate.

Para la creación de las entidades se útiliza la anotación @Entity para indicarle a Spring que esa clase será mapeada a una tabla en la base de datos. En la Figura 62 se muestra una clase abstracta AbstractEntity que contiene los atributos comunes a todas las entidades como el id, la fecha de creación, la fecha de actualización y si está activo o no. En la Figura 63 se muestra una entidad con herencia de la claseAbstractEntity, en donde se establecen los atributos y las relaciones con otras entidades para que sean mapeadas en la base de datos.

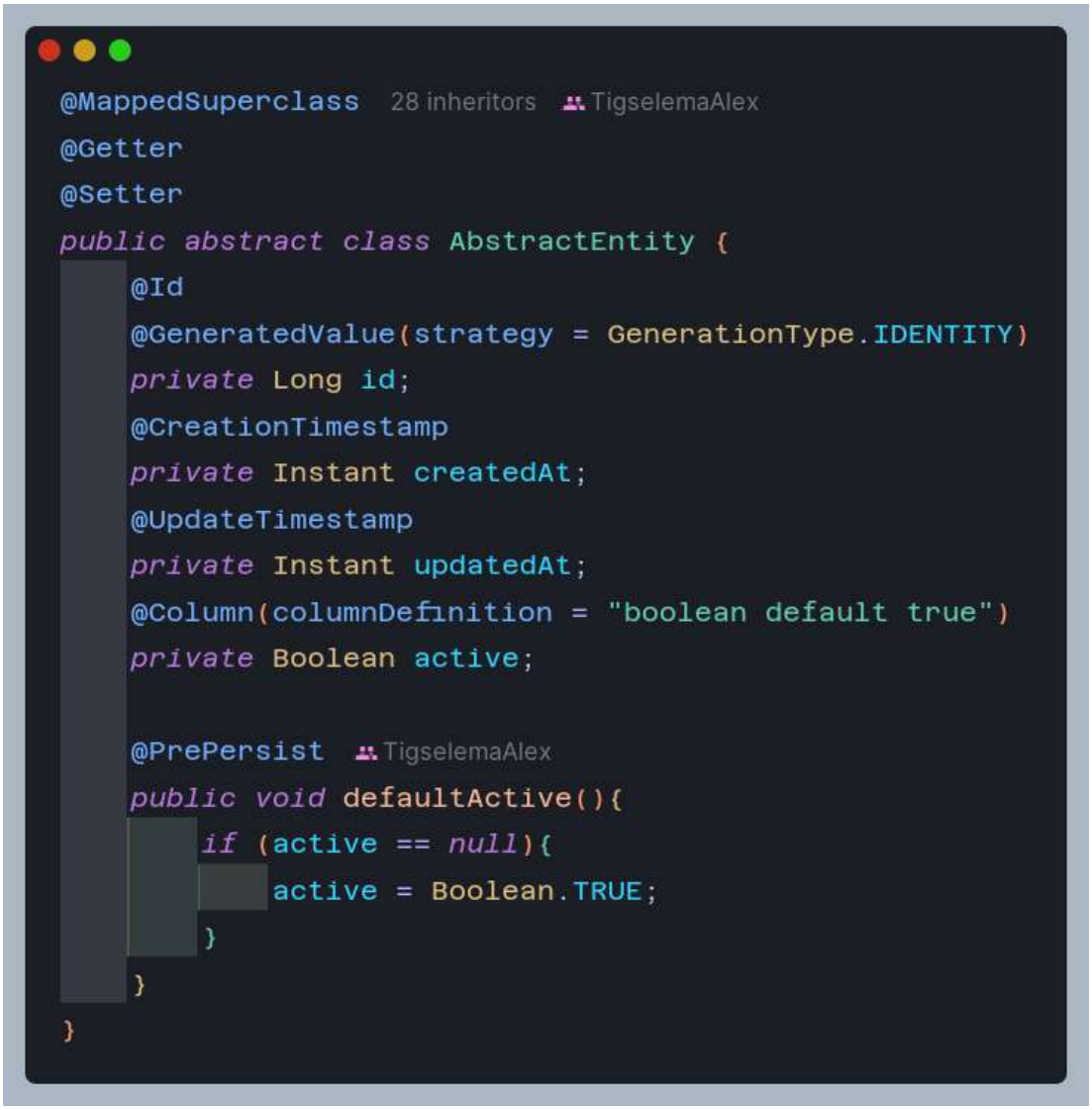


Figura 62. Código de la clase abstracta base para las demas entidades AbstractEntity

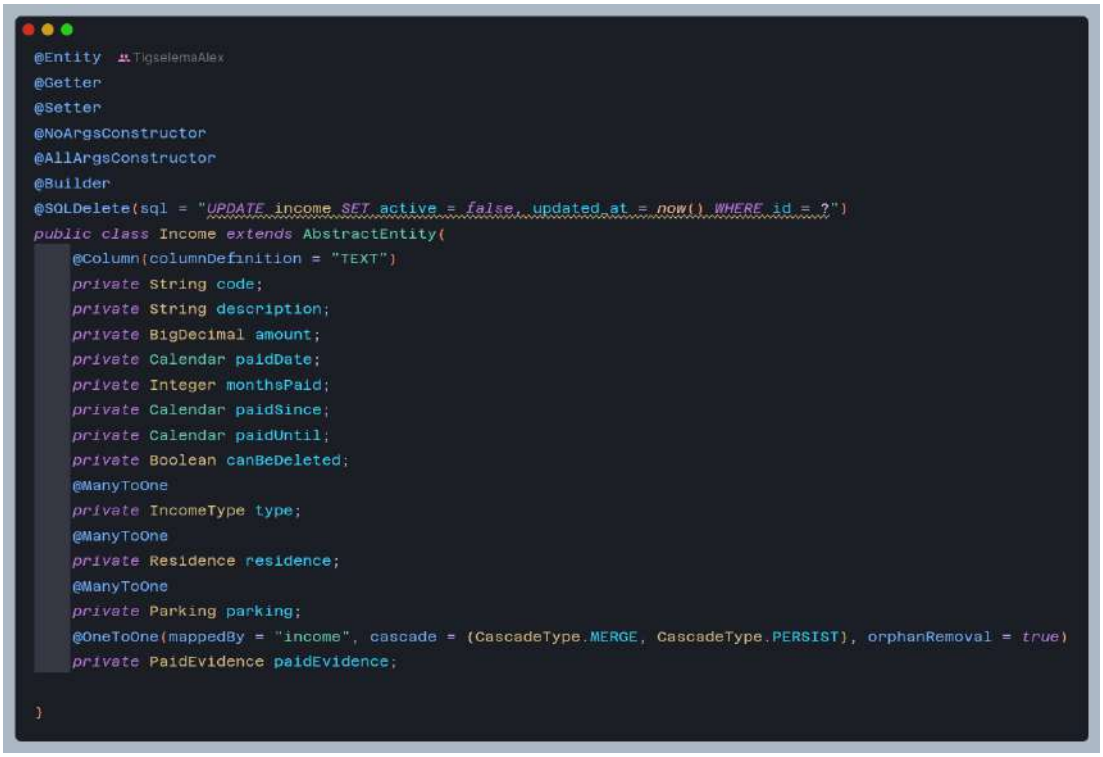


Figura 63. Código de la entidad Income

Para los procesos CRUD se utilizaron repositorios que extienden de la interfaz JpaRepository de Spring Data tal como se muestra en la Figura 64, en donde se definen los métodos necesarios para realizar las operaciones de creación, lectura, actualización y eliminación de los registros en la base de datos además de los métodos personalizados que sean necesarios.

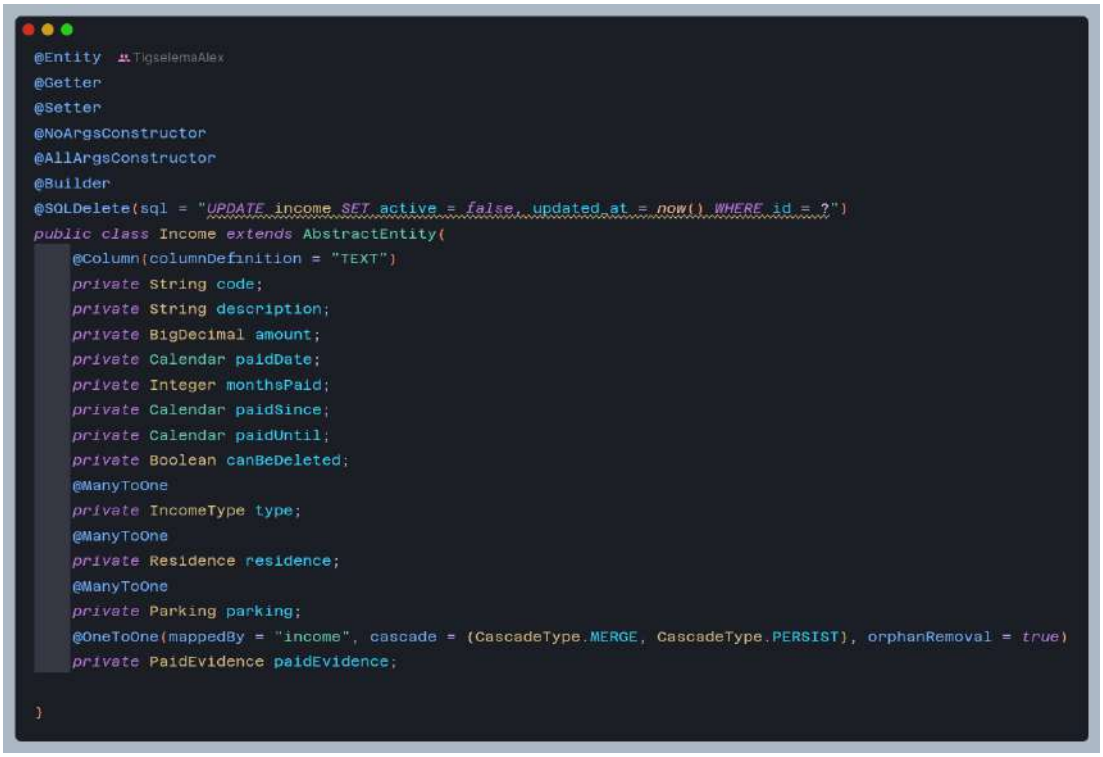


Figura 64. Código de el repositorio para para entidad Income

Posteriormente a la creación de todas las entidades se generó el siguiente diagrama entidad-relación como se muestra en las Figuras 65 y 66.

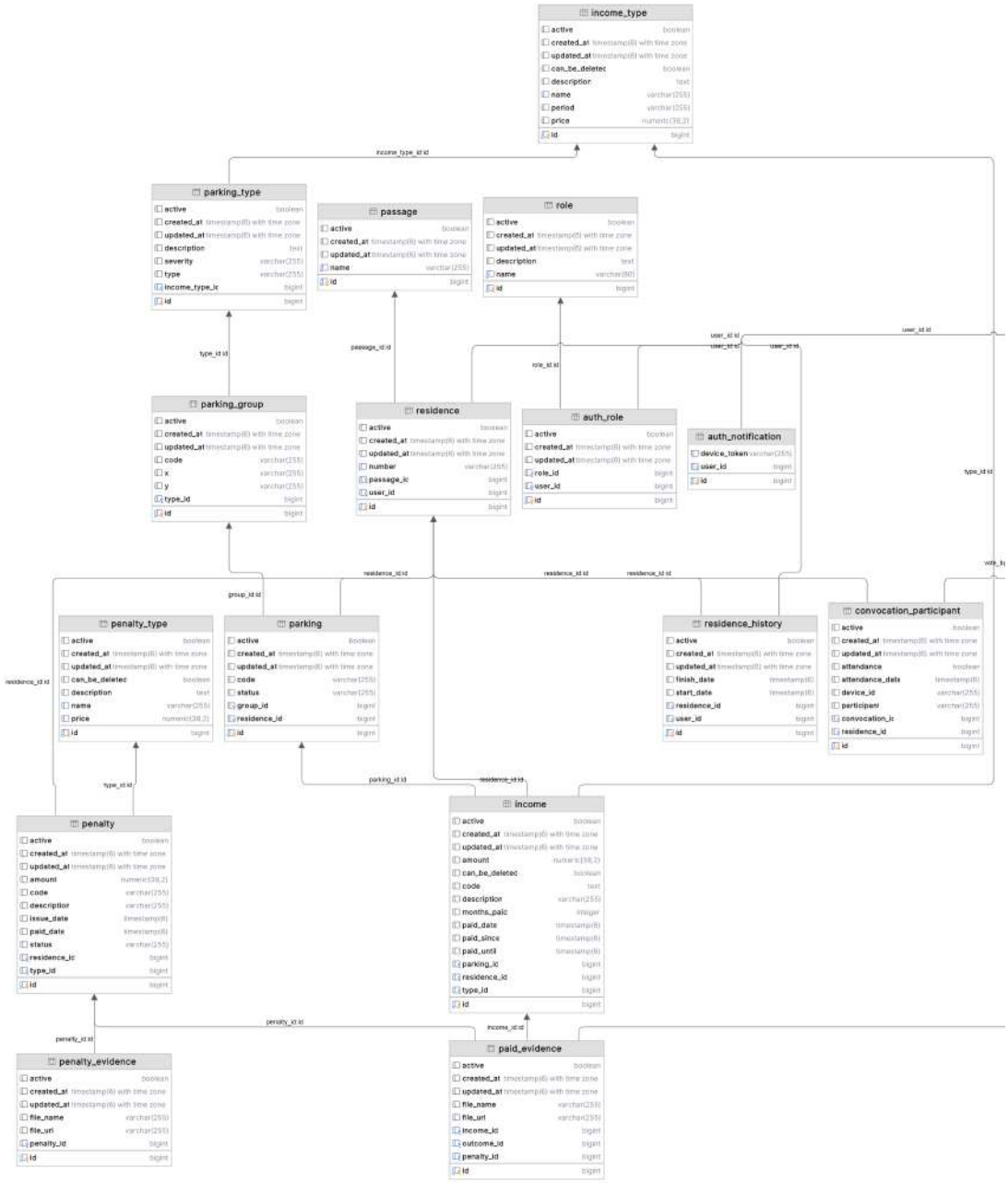


Figura 65. Diagrama entidad-relación de la base de datos (Parte 1)

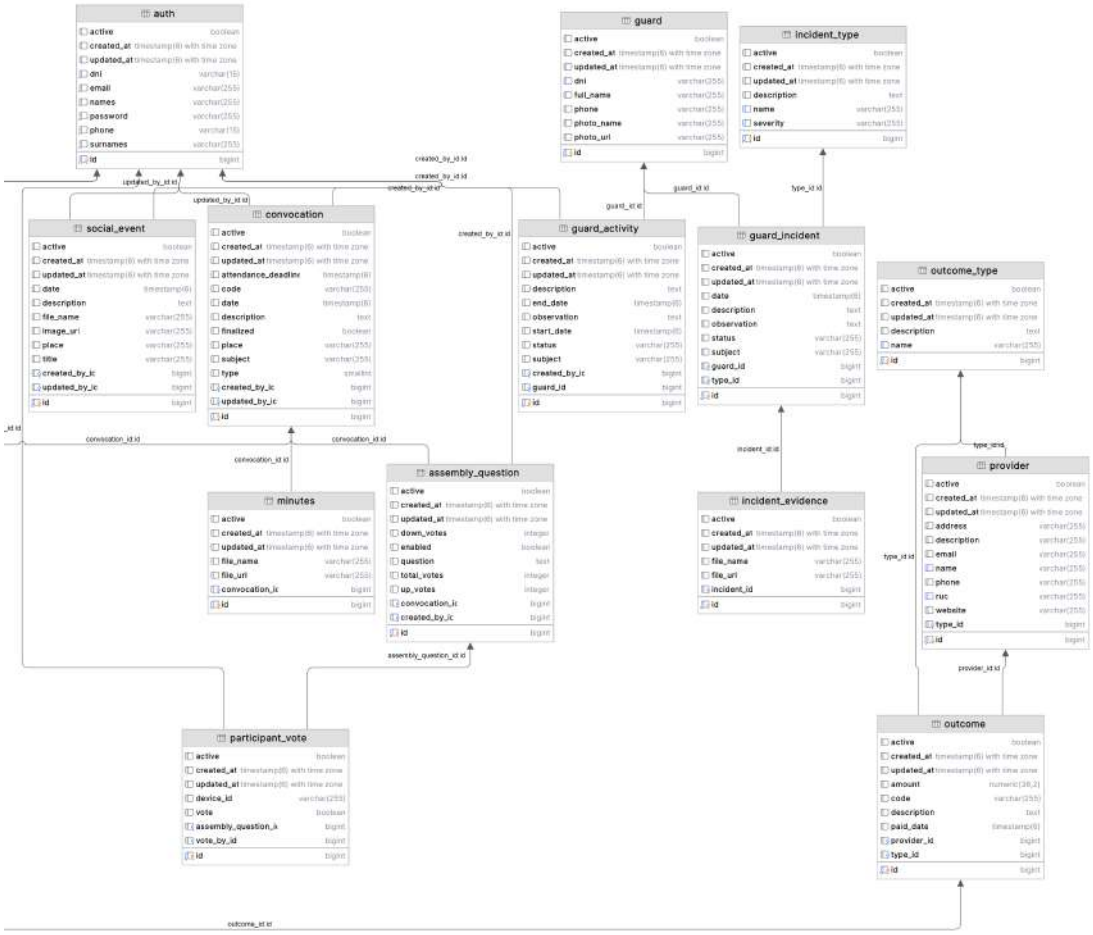


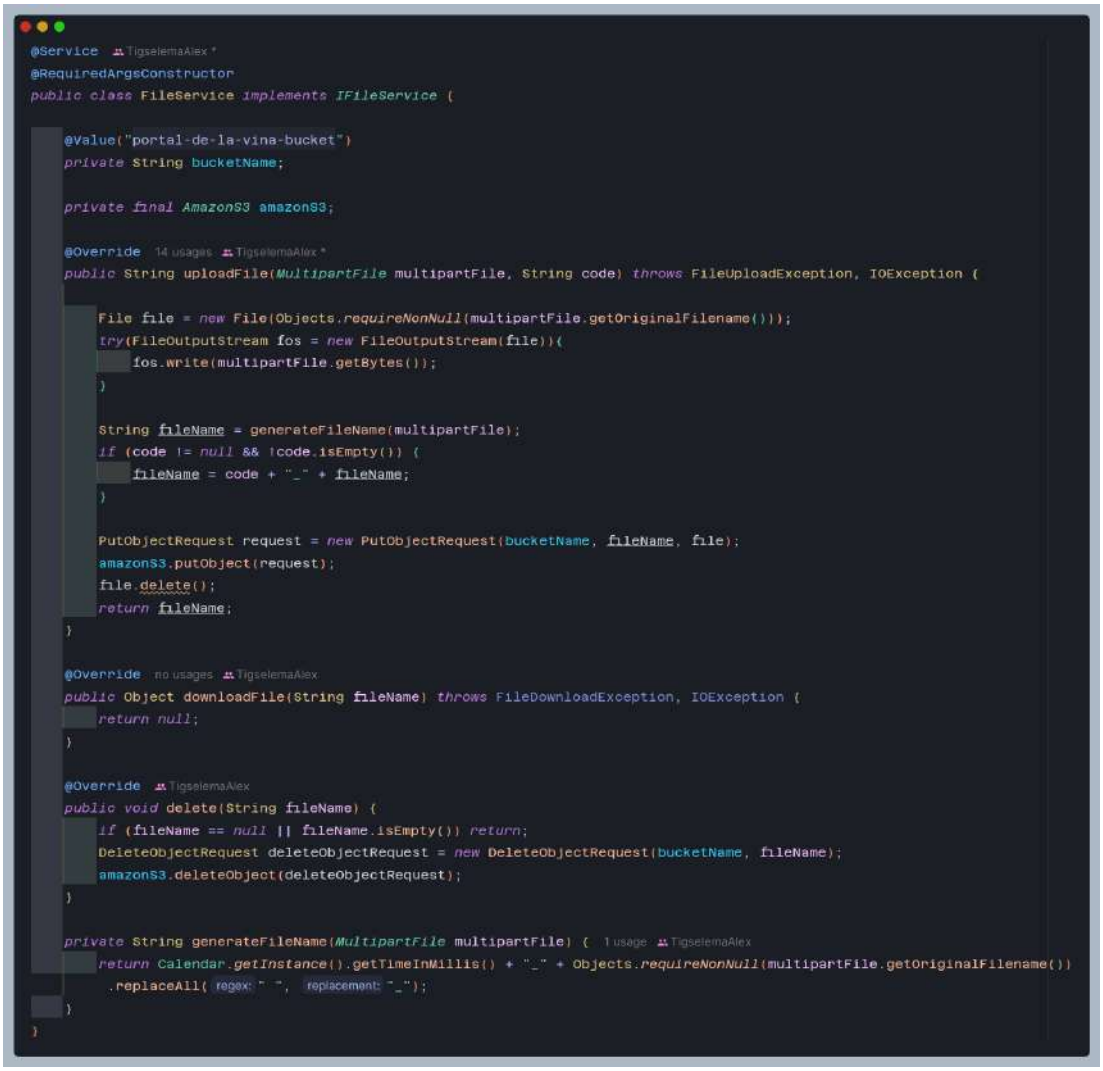
Figura 66. Diagrama entidad-relación de la base de datos (Parte 2)

- ***Configuración y uso del almacenamiento de archivos en S3.*** Para poder usar los servicios de almacenamiento de AWS S3 se creo una componente o bean de configuración en donde se establecieron las credenciales de acceso al usuario de AWS y la región en donde se encuentra alojado el bucket de almacenamiento de archivos como se muestra en la Figura 67.



Figura 67. Código de configuración del almacenamiento de archivos en S3

Para poder realizar la carga y eliminación de archivos en el bucket de almacenamiento como se ve en la Figura 68, se creó un servicio en donde se establecieron los métodos de carga en donde se reciben como parámetros el código y el archivo a cargar, en este método se genera un nombre único para el archivo y se procede a crear una petición de carga al bucket a través del método putObject de la instancia del bean inyectado AmazonS3. Por otro lado el método de eliminación recibe como parámetro el nombre del archivo a eliminar y se procede a realizar la eliminación a través del bean inyectado AmazonS3.



```java
@Service  TigselemaAlex *
@RequiredArgsConstructor
public class FileService implements IFileService {

    @Value("portal-de-la-vina-bucket")
    private String bucketName;

    private final AmazonS3 amazonS3;

    @Override  14 usages  TigselemaAlex *
    public String uploadFile(MultipartFile multipartFile, String code) throws FileUploadException, IOException {

        File file = new File(Objects.requireNonNull(multipartFile.getOriginalFilename()));
        try(FileOutputStream fos = new FileOutputStream(file)){
            fos.write(multipartFile.getBytes());
        }

        String fileName = generateFileName(multipartFile);
        if (code != null && !code.isEmpty()) {
            fileName = code + "_" + fileName;
        }

        PutObjectRequest request = new PutObjectRequest(bucketName, fileName, file);
        amazonS3.putObject(request);
        file.delete();
        return fileName;
    }

    @Override  no usages  TigselemaAlex
    public Object downloadFile(String fileName) throws FileDownloadException, IOException {
        return null;
    }

    @Override  TigselemaAlex
    public void delete(String fileName) {
        if (fileName == null || fileName.isEmpty()) return;
        DeleteObjectRequest deleteObjectRequest = new DeleteObjectRequest(bucketName, fileName);
        amazonS3.deleteObject(deleteObjectRequest);
    }

    private String generateFileName(MultipartFile multipartFile) {  1 usage  TigselemaAlex
        return Calendar.getInstance().getTimeInMillis() + "_" + Objects.requireNonNull(multipartFile.getOriginalFilename())
                .replaceAll( regex: " ",  replacement: "_");
    }
}
```

Figura 68. Código del servicio de carga y eliminación del archivo S3

- ***Configuración del motor de plantillas HTML FreeMarker y envío de correos electrónicos.*** Se debe crear un componente o bean para configurar el creador de plantillas HTML para los correos electrónicos en este caso usando FreeMarker como motor de plantillas tal como se muestra en la Figura 69.



Figura 69. Código de configuración del motor de plantillas HTML para los correos electrónicos

Para el envío de correos electrónicos se creo un servicio en el cual se establecen los métodos necesarios para los envíos de correos. Se necesita crear un modelo para que éste sea mapeado en la plantilla HTML de FreeMarker, en la Figura 70 se muestra un ejemplo de modelo y en la Figura 71 se muestra el uso de éste modelo en un método y el envío de un correo electrónico a un usuario en particular.



Figura 70. Código de modelo para mapearlo en la plantilla HTML



Figura 71. Código del método de envio de correo electrónico para recuperar la contraseña

- ***Configuración de Firebase para el envío de Notificaciones Push y servicio de envió de notificaciones.*** Para poder enviar notificaciones push a los dispositivos móviles se debe configurar Firebase Cloud Messaging (FCM) en la API, para ello se debe crear un componente o bean de configuración en donde se establecerán las credenciales de acceso a Firebase como se muestra en la Figura 72, dichas credenciales se obtienen de un archivo Json que se descarga desde la consola de Firebase para posteriormente referenciar su ubicación en el archivo de configuración de propiedades como se mostró en la Figura 60.

```java
@Configuration  TigselemaAlex
@EnableConfigurationProperties(FirebaseProperties.class)
@RequiredArgsConstructor
public class FirebaseConfiguration {

    private final FirebaseProperties firebaseProperties;

    @Bean  TigselemaAlex
    public GoogleCredentials googleCredentials(){
        try {
            if (firebaseProperties.getServiceAccount() != null) {
                try(InputStream is = firebaseProperties.getServiceAccount().getInputStream()){
                    return GoogleCredentials.fromStream(is);
                }
            }else{
                return GoogleCredentials.getApplicationDefault();
            }
        }catch (IOException ex){
            throw new RuntimeException("Error al cargar las credenciales de Firebase", ex);
        }
    }

    @Bean  TigselemaAlex
    public FirebaseApp firebaseApp(GoogleCredentials googleCredentials){

            FirebaseOptions options = FirebaseOptions.builder()
                    .setCredentials(googleCredentials)
                    .build();
            if (FirebaseApp.getApps().isEmpty()) {
                return FirebaseApp.initializeApp(options);
            }
            return FirebaseApp.getInstance();

    }

    @Bean  TigselemaAlex
    public FirebaseMessaging firebaseMessaging(FirebaseApp firebaseApp){
        return FirebaseMessaging.getInstance(firebaseApp);
    }
}
```

Figura 72. Código de configuración de Firebase para el envío de Notificaciones Push

En la Figura 73 se muestra como se crea un servicio para el envío de notificaciones

push y como se actualiza el token de cada dispositivos móvil en la base de datos para poder enviar notificaciones a un dispositivo en particular.



```java
@Service
@RequiredArgsConstructor
public class PushNotificationService implements IPushNotificationService {

    private final FirebaseMessaging firebaseMessaging;
    private final IUserService userService;
    private final AuthNotificationRepository authNotificationRepository;

    @Override
    public void sendPushNotification(Long userId, String title, String message, Map<String, String> data) throws
     FirebaseMessagingException {
        AuthNotification authNotification = findAuthNotificationByUserId(userId);
        if(authNotification == null){
            return;
        }
        Message messageObj = Message.builder()
                .putAllData(data)
                .setToken(authNotification.getDeviceToken())
                .setNotification(Notification.builder()
                        .setTitle(title)
                        .setBody(message)
                        .build()
                )
                .build();
        firebaseMessaging.send(messageObj);
    }

    @Override
    public void sendPushNotificationAllClients(String title, String message, Map<String, String> data) throws
     FirebaseMessagingException {
        List<AuthNotification> authNotifications = authNotificationRepository.findAllByDeviceTokenNotNull();
        if (authNotifications.isEmpty()) {
            return;
        }
        MulticastMessage multicastMessage = MulticastMessage.builder()
                .putAllData(data)
                .setNotification(Notification.builder()
                        .setTitle(title)
                        .setBody(message)
                        .build()
                )
                .addAllTokens(authNotifications.stream().map(AuthNotification::getDeviceToken).toList())
                .build();

        firebaseMessaging.sendMulticast(multicastMessage);
    }


    public void updateDeviceToken(String deviceToken, Long userId){
        AuthNotification authNotification = findAuthNotificationByUserId(userId);
        if(authNotification == null){
            User user = userService.findById(userId);
            authNotification = new AuthNotification();
            authNotification.setUser(user);
        }
        if (deviceToken.equals(authNotification.getDeviceToken())) {
            return;
        }
        authNotification.setDeviceToken(deviceToken);
        authNotificationRepository.save(authNotification);
    }

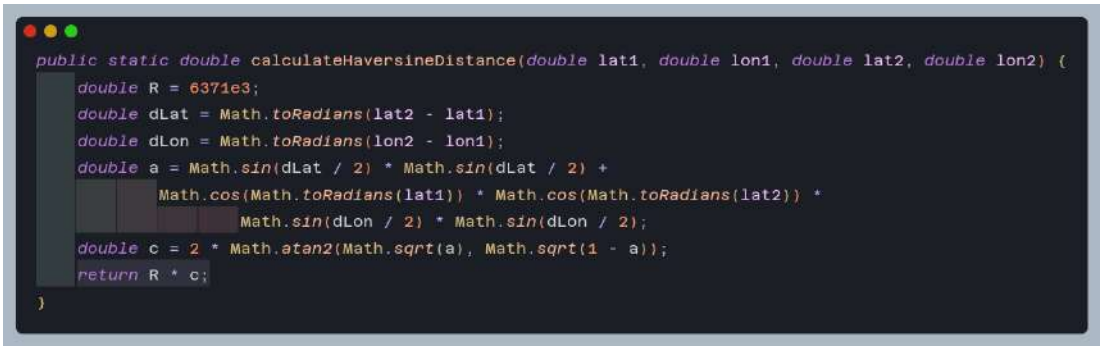
    @Override
    public void deleteDeviceToken(Long userId) {
        AuthNotification authNotification = findAuthNotificationByUserId(userId);
        if(authNotification == null){
            return;
        }
        authNotificationRepository.delete(authNotification);
    }

    private AuthNotification findAuthNotificationByUserId(Long userId){
        return authNotificationRepository.findByUserId(userId).orElse( other: null);
    }
}
```

Figura 73. Código del servicio de las Notificaciones Push

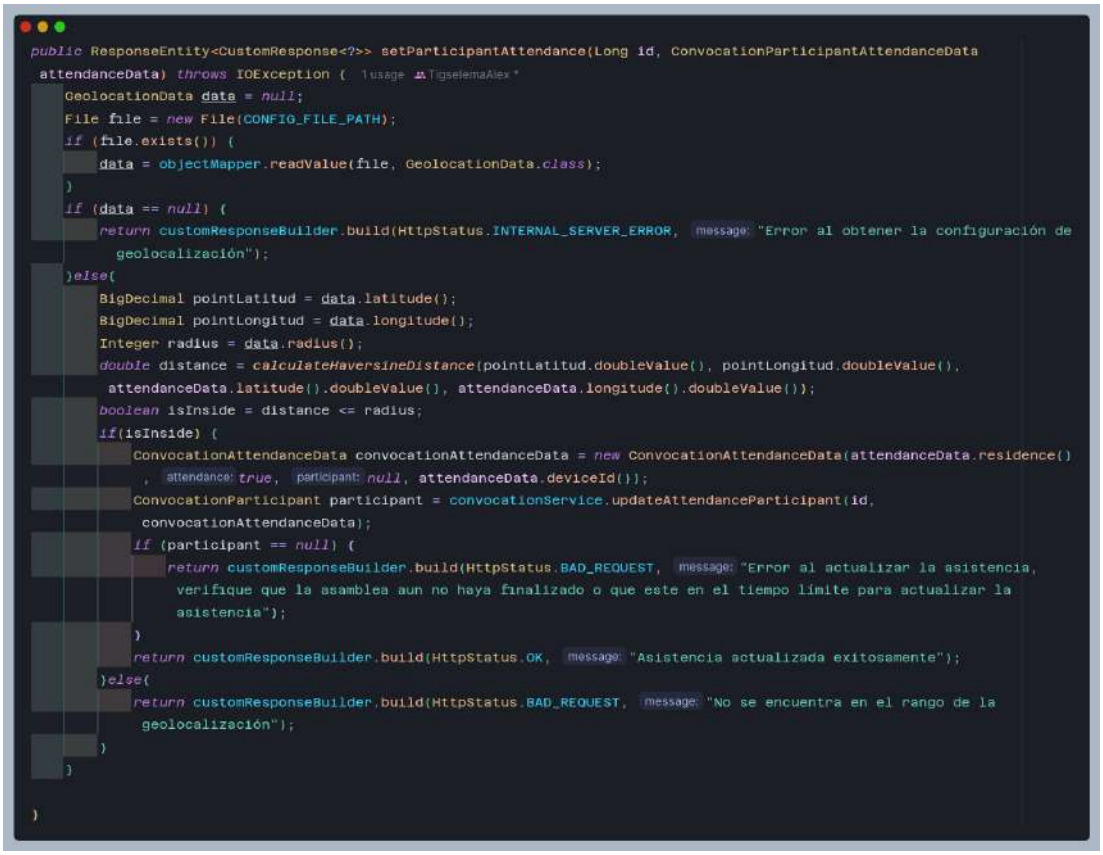
- ***Cálculo de la distancia de Haversine.*** Como parte de la funcionalidad de la geolocalización se debe calcular la distancia entre dos puntos geográficos para poder

determinar posteriormente si se encuentra dentro de un radio de aceptación. Para este cálculo se necesitan las coordenadas del punto de referencia las cuales son almacenadas en un archivo Json en el servidor, y las coordenadas del punto a calcular la distancia. Posteriormente se aplica la formula de Haversine como se muestra en la Figura 74 y se obtiene la distancia en metros. Por último se compara la distancia obtenida con el radio de aceptación para determinar si el punto se encuentra dentro del rango o no tal como se ve aplicada en el registro de asistencia en la asamblea en la Figura 75.



```java
public static double calculateHaversineDistance(double lat1, double lon1, double lat2, double lon2) {
    double R = 6371e3;
    double dLat = Math.toRadians(lat2 - lat1);
    double dLon = Math.toRadians(lon2 - lon1);
    double a = Math.sin(dLat / 2) * Math.sin(dLat / 2) +
            Math.cos(Math.toRadians(lat1)) * Math.cos(Math.toRadians(lat2)) *
                    Math.sin(dLon / 2) * Math.sin(dLon / 2);
    double c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));
    return R * c;
}
```

Figura 74. Código del cálculo de la distancia de Haversine en Java

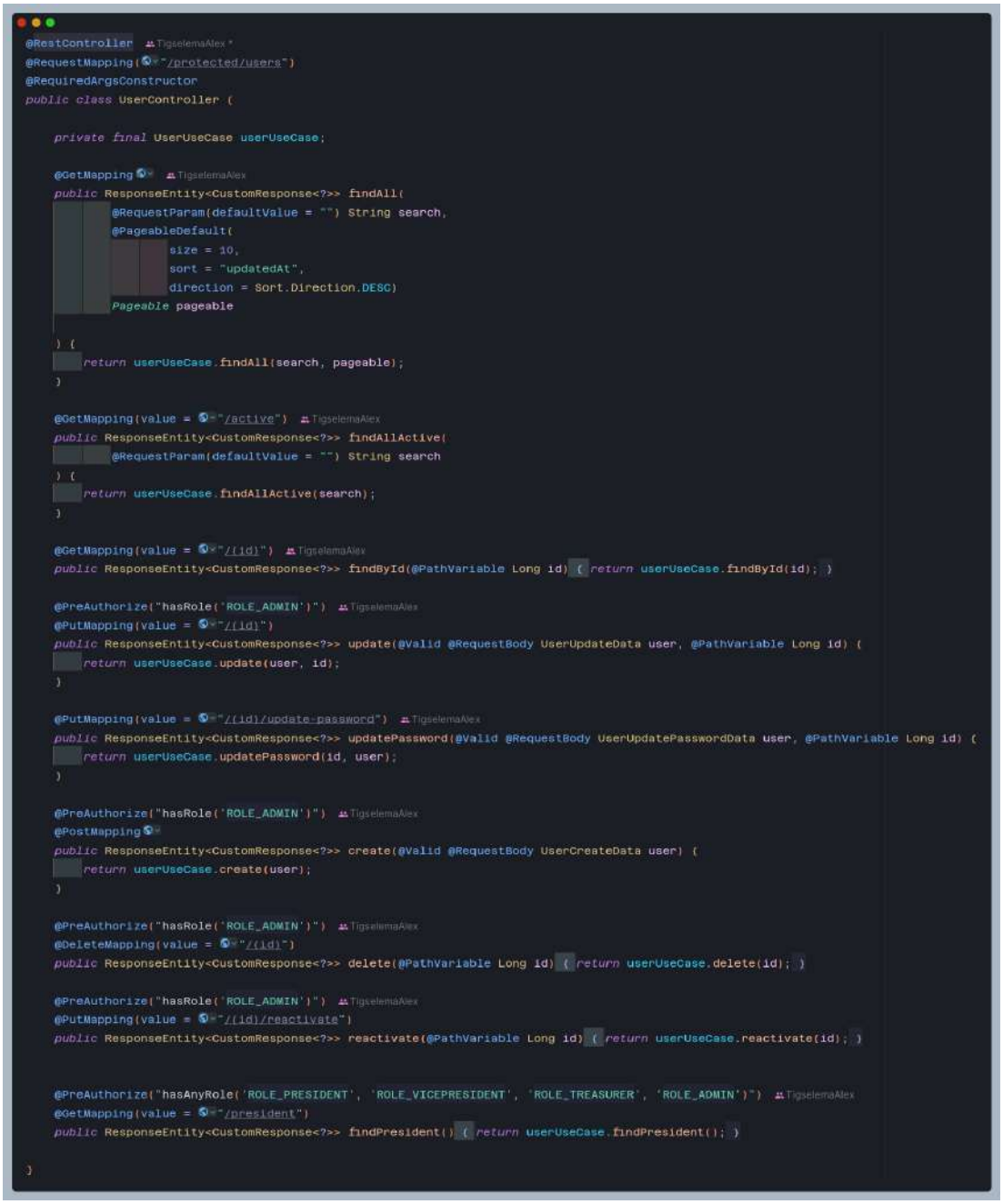


```java
public ResponseEntity<CustomResponse<?>> setParticipantAttendance(Long id, ConvocationParticipantAttendanceData
 attendanceData) throws IOException {  1 usage  Tigselema Alex *
    GeolocationData data = null;
    File file = new File(CONFIG_FILE_PATH);
    if (file.exists()) {
        data = objectMapper.readValue(file, GeolocationData.class);
    }
    if (data == null) {
        return customResponseBuilder.build(HttpStatus.INTERNAL_SERVER_ERROR,  message: "Error al obtener la configuración de
          geolocalización");
    }else{
        BigDecimal pointLatitud = data.latitude();
        BigDecimal pointLongitud = data.longitude();
        Integer radius = data.radius();
        double distance = calculateHaversineDistance(pointLatitud.doubleValue(), pointLongitud.doubleValue(),
         attendanceData.latitude().doubleValue(), attendanceData.longitude().doubleValue());
        boolean isInside = distance <= radius;
        if(isInside) {
            ConvocationAttendanceData convocationAttendanceData = new ConvocationAttendanceData(attendanceData.residence()
              ,  attendance: true,  participant: null, attendanceData.deviceId());
            ConvocationParticipant participant = convocationService.updateAttendanceParticipant(id,
             convocationAttendanceData);
            if (participant == null) {
                return customResponseBuilder.build(HttpStatus.BAD_REQUEST,  message: "Error al actualizar la asistencia,
                  verifique que la asamblea aun no haya finalizado o que este en el tiempo límite para actualizar la
                  asistencia");
            }
            return customResponseBuilder.build(HttpStatus.OK,  message: "Asistencia actualizada exitosamente");
        }else{
            return customResponseBuilder.build(HttpStatus.BAD_REQUEST,  message: "No se encuentra en el rango de la
             geolocalización");
        }
    }

}
```

Figura 75. Código de método de registro de asistencia en la asamblea

- ***Controladores.*** Para poder exponer los puntos de salida o endpoints de la aplicación se deben crear controladores en donde se establecen las rutas y los métodos que se ejecutarán según el tipo de petición que se realice y validando los roles de los usuarios que realizan la petición tal como se muestra en la Figura 76.



```java
@RestController
@RequestMapping("/protected/users")
@RequiredArgsConstructor
public class UserController {

    private final UserUseCase userUseCase;

    @GetMapping
    public ResponseEntity<CustomResponse<?>> findAll(
            @RequestParam(defaultValue = "") String search,
            @PageableDefault(
                    size = 10,
                    sort = "updatedAt",
                    direction = Sort.Direction.DESC)
            Pageable pageable

    ) {
        return userUseCase.findAll(search, pageable);
    }

    @GetMapping(value = "/active")
    public ResponseEntity<CustomResponse<?>> findAllActive(
            @RequestParam(defaultValue = "") String search
    ) {
        return userUseCase.findAllActive(search);
    }

    @GetMapping(value = "/{id}")
    public ResponseEntity<CustomResponse<?>> findById(@PathVariable Long id) { return userUseCase.findById(id); }

    @PreAuthorize("hasRole('ROLE_ADMIN')")
    @PutMapping(value = "/{id}")
    public ResponseEntity<CustomResponse<?>> update(@Valid @RequestBody UserUpdateData user, @PathVariable Long id) {
        return userUseCase.update(user, id);
    }

    @PutMapping(value = "/{id}/update-password")
    public ResponseEntity<CustomResponse<?>> updatePassword(@Valid @RequestBody UserUpdatePasswordData user, @PathVariable Long id) {
        return userUseCase.updatePassword(id, user);
    }

    @PreAuthorize("hasRole('ROLE_ADMIN')")
    @PostMapping
    public ResponseEntity<CustomResponse<?>> create(@Valid @RequestBody UserCreateData user) {
        return userUseCase.create(user);
    }

    @PreAuthorize("hasRole('ROLE_ADMIN')")
    @DeleteMapping(value = "/{id}")
    public ResponseEntity<CustomResponse<?>> delete(@PathVariable Long id) { return userUseCase.delete(id); }

    @PreAuthorize("hasRole('ROLE_ADMIN')")
    @PutMapping(value = "/{id}/reactivate")
    public ResponseEntity<CustomResponse<?>> reactivate(@PathVariable Long id) { return userUseCase.reactivate(id); }


    @PreAuthorize("hasAnyRole('ROLE_PRESIDENT', 'ROLE_VICEPRESIDENT', 'ROLE_TREASURER', 'ROLE_ADMIN')")
    @GetMapping(value = "/president")
    public ResponseEntity<CustomResponse<?>> findPresident() { return userUseCase.findPresident(); }

}
```

Figura 76. Código del controlador de usuarios

b. Frontend

Sistema web Administrativo

- ***Dependencias del sistema.*** Para el desarrollo del sistema web Administrativo el cual se construyó con Angular se necesito instalar dependencias mediante el gestor de paquetes npm con el comando *npm install <dependencia>*. Estas dependencias se encuentran en el archivo *package.json* como se muestra en la Figura 77. Entre ellas se encuentran PrimeNg, PrimeFlex, PrimeIcons, quill, chart.js, ngx-print, entre otros.

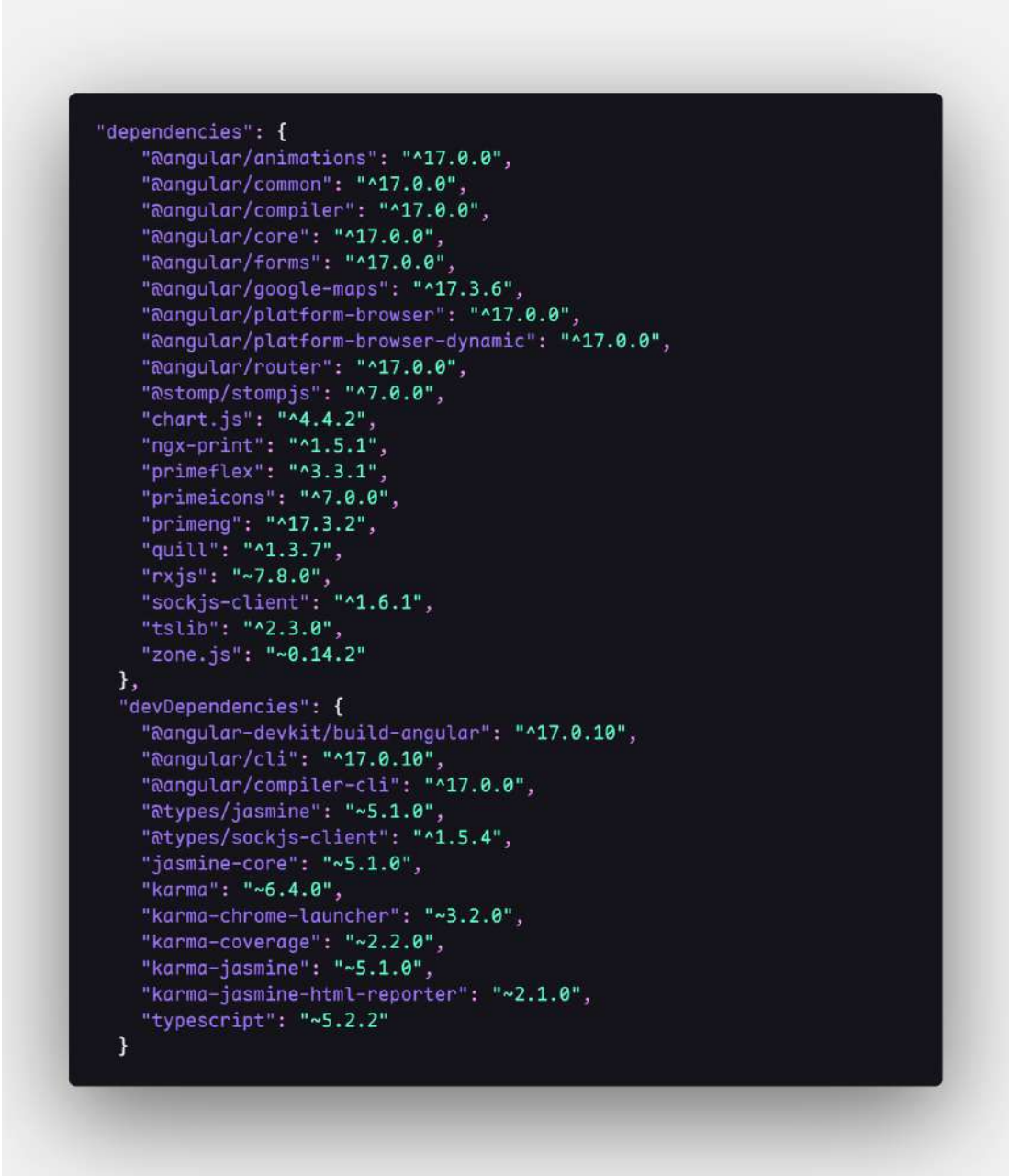


Figura 77. Archivo *package.json* con las dependencias del sistema web

- ***Configuración de la aplicación.*** Angular nos provee un archivo principal llamado *app.config.ts* en donde se establecen los proveedores de servicios tales como el servicio de animaciones, el de rutas, el servicio para realizar peticiones HTTP, el servicio de

localización y el servicio de optimización de imágenes como se muestra en la Figura 78.



```
import { ApplicationConfig, LOCALE_ID } from '@angular/core';
import { provideRouter } from '@angular/router';
import { routes } from './app.routes';
import { provideAnimations } from '@angular/platform-browser/animations';
import { provideHttpClient, withInterceptors } from
'@angular/common/http';
import { authInterceptorInterceptor } from
'./core/interceptors/auth-interceptor.interceptor';
import { IMAGE_CONFIG, registerLocaleData } from '@angular/common';
import es from '@angular/common/locales/es';

registerLocaleData(es);

export let appConfig: ApplicationConfig;
appConfig = {
  providers: [
    provideRouter(routes),
    provideAnimations(),
    provideHttpClient(withInterceptors([authInterceptorInterceptor])),
    {
      provide: IMAGE_CONFIG,
      useValue: {
        disableImageSizeWarning: true,
        disableImageLazyLoadWarning: true,
      },
    },
    {
      provide: LOCALE_ID,
      useValue: 'es-EC',
    },
  ],
};
```

Figura 78. Archivo *app.config.ts* con la configuración de la aplicación

Adicionalmente es necesario configurar las rutas de la aplicación en el archivo *app-routes.ts* en donde se establecen las rutas de los componentes que se mostrarán en la aplicación como se muestra en la Figura 79.



Figura 79. Archivo *app-routes.ts* con la configuración de las rutas de la aplicación

- ***Inicio de sesión.*** En la Figura 80 se muestra el código del login en donde se tomarán las credenciales del formulario de inicio de sesión y se enviarán al servicio de autenticación para que éste realize la solicitud a la API y así el controlador de autenticación pueda validar las credenciales a través del método de autenticación definido Figura 81 en donde se válidan las credenciales y en el caso de ser válidas se generará un token JWT que será enviado al cliente web para que este lo almacene en el sesión storage y pueda ser utilizado en las demás peticiones mediante el interceptador de peticiones HTTP definido en el sistema web Figura 83.



```
login() {
    if (this.loginForm.valid) {
      const loginRequest: AuthLoginData = this.loginForm.getRawValue();
      this.authService
        .login(loginRequest)
        .pipe()
        .subscribe({
          next: (jwt) => {
            this.storageService.saveAuth(jwt);
            this.router.navigate(['/home']);
          },
          error: (error) => {
            this.storageService.clean();
            if (error.status === 401 || error.status === 404) {
              this.messageService.add({
                severity: 'error',
                summary: 'Error',
                detail: 'Usuario o contraseña incorrectos',
              });
            } else {
              this.messageService.add({
                severity: 'error',
                summary: 'Error',
                detail: 'Error al iniciar sesión',
              });
            }
          },
        });
    } else {
      Object.values(this.loginForm.controls).forEach((control) => {
        if (control.invalid) {
          control.markAsDirty();
          control.updateValueAndValidity({ onlySelf: true });
        }
      });
    }
  }
```

Figura 80. Código del inicio de sesión en el frontend

```java
public ResponseEntity<JwtResponse> login(AuthLoginData authLoginData) {  1 usa
    Authentication authentication = authenticationManager
            .authenticate(
                    new UsernamePasswordAuthenticationToken(
                            authLoginData.dni(),
                            authLoginData.password()
                    )
            );
    if (authentication.isAuthenticated()) {
        JwtResponse response = new JwtResponse(
                (CustomUserDetails)authentication.getPrincipal(),
                jwtProvider.generateToken(authLoginData.dni())
        );
        return ResponseEntity.ok(response);
    }else{
        throw new UsernameNotFoundException("Invalid username or password");
    }
}
```

Figura 81. Código del método de autenticación en el API



Figura 82. Página de inicio de sesión



```
export const authInterceptorInterceptor: HttpInterceptorFn = (req, next)
⇒ {
  let jwt = inject(StorageService).getAuth();
  if (jwt) {
    req = req.clone(
      {
        setHeaders: {
          Authorization: `Bearer ${jwt.token}`
        }
      }
    );
  }
  return next(req);
};
```

Figura 83. Código del interceptor de peticiones HTTP en el sistema web

- ***Gestión de usuarios.*** Los usuarios del condominio únicamente podrán ser registrado por el administrador para lo cual se debe ingresar los datos personales proporcionados por el usuario y se debe seleccionar el rol que tendrá el usuario dentro del condominio. Ésta información será enviada a la API en un cuerpo de tipo JSON para que ésta pueda ser procesada y almacenada en la base de datos. Una vez registrado el usuario podrá ser visualizado en la tabla de usuarios en donde se podrá editar o eliminar el usuario tal como se muestra en la Figura 85.



Figura 84. Endpoint para obtener todos los usuarios

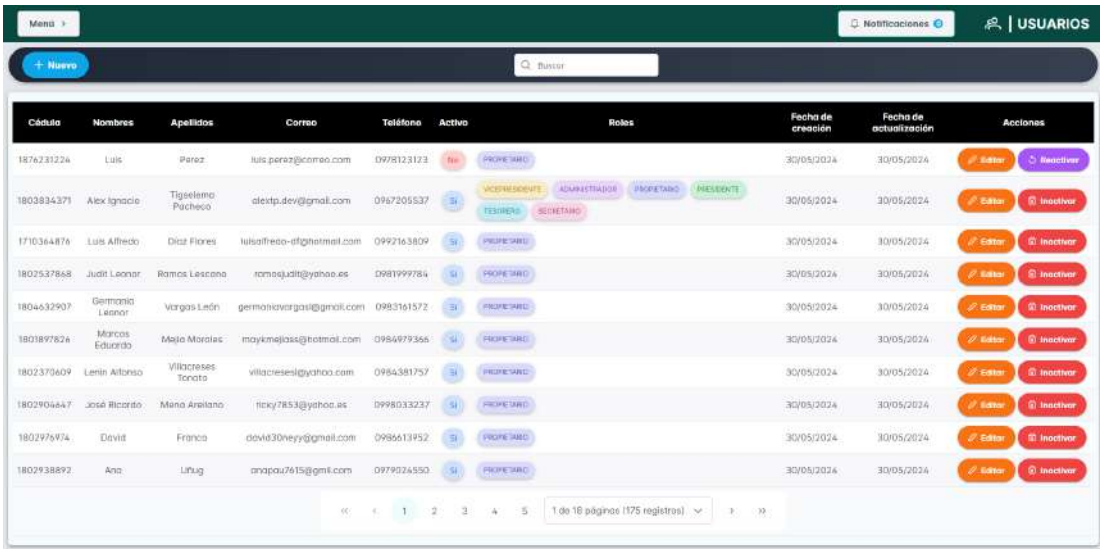


Figura 85. Página de gestión de usuarios



Figura 86. Endpoint para crear un usuario

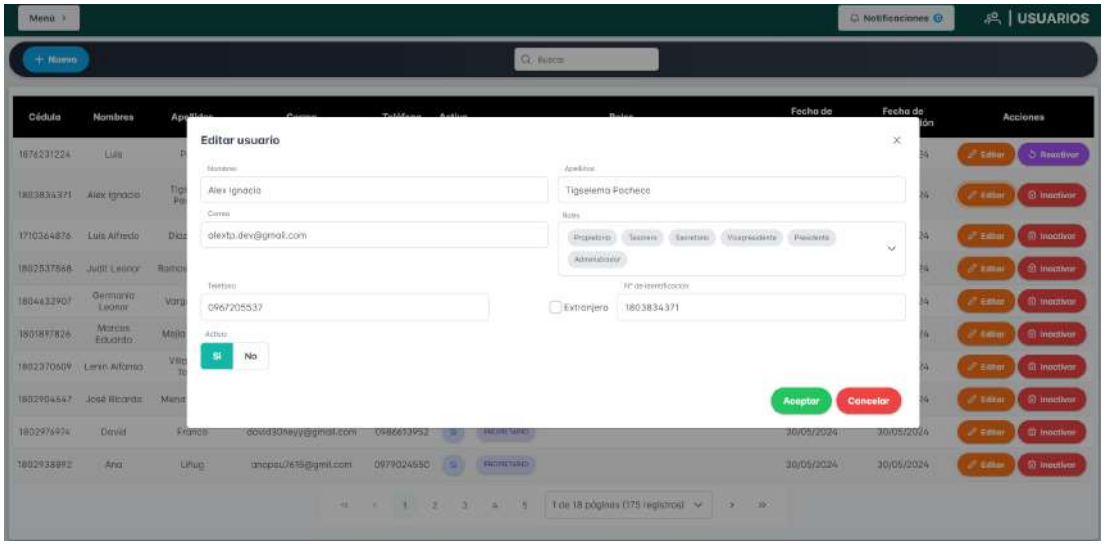


Figura 87. Formulario de creación y edición de usuarios

- ***Gestión de geolocalización.*** El usuario administrador podrá actualizar el punto de referencia para la geolocalización de la asamblea en donde se podrá visualizar un mapa con la ubicación actual y una circunferencia representado el radio de aceptación. Para la modificación de éste punto de referencia se debe enviar la latitud, la longitud y el radio de aceptación en un cuerpo de tipo JSON a la API en donde se procesara ésta información leyendo el archivo JSON del servidor y modificando sus valores tal como se muestra en la Figura 88.

```
public ResponseEntity<CustomResponse<?>> updateConfig(GeolocationData data) throws IOException {  1 usage   TigselemaAlex *
    File file = new File(CONFIG_FILE_PATH);
    objectMapper.writeValue(file, data);
    simpMessagingTemplate.convertAndSend( destination: "/topic/notification", new NotificationResponse( title: "Geolocalización",
      message: "Se ha modificado el punto de reuniones",  user: null));
    return customResponseBuilder.build(HttpStatus.OK,  message: "Configuración actualizada correctamente");
}
```

Figura 88. Método de actualización de la geolocalización en el API

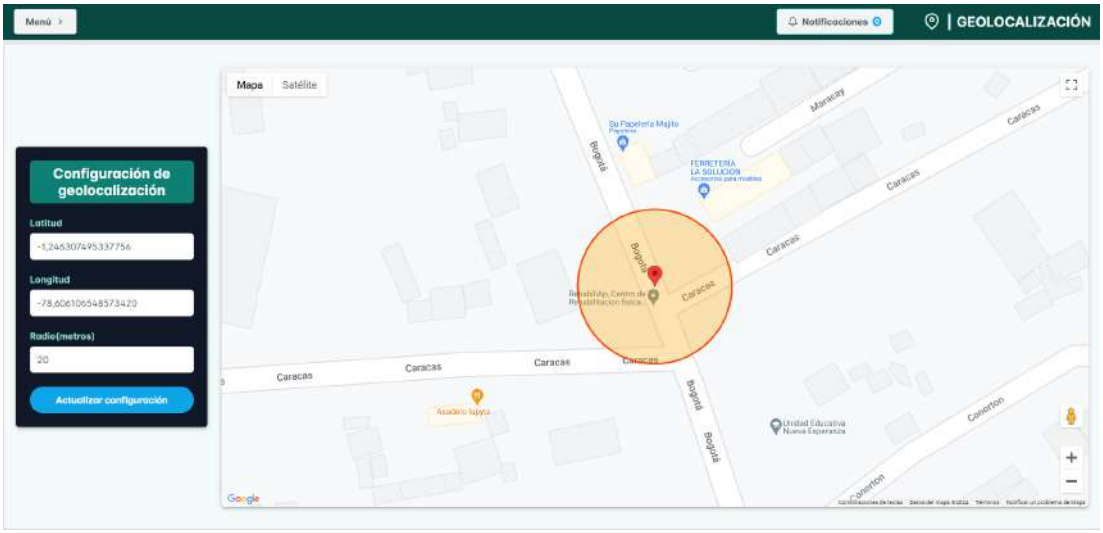


Figura 89. Página de gestión de geolocalización

- ***Gestión de residencias.*** El presidente o vicepresidente podrá registrar en las residencias el usuario representante de la misma para ello se debe enviar a la API el id del usuario representante y el id de la residencia en un cuerpo de tipo JSON. La API procesará esta información y actualizará el usuario representante de la residencia tal como se muestra en la Figura 90.

```
@PreAuthorize("hasAnyRole('ROLE_ADMIN', 'ROLE_PRESIDENT', 'ROLE_VICE_PRESIDENT')")
@PutMapping("/{id}")
public ResponseEntity<CustomResponse<?>> updateResidence(
        @PathVariable Long id,
        @Valid @RequestBody ResidenceUpdateData residence) {
    return residenceUseCase.update(id, residence);
}
```

Figura 90. Endpoint para actualizar el usuario representante de una residencia

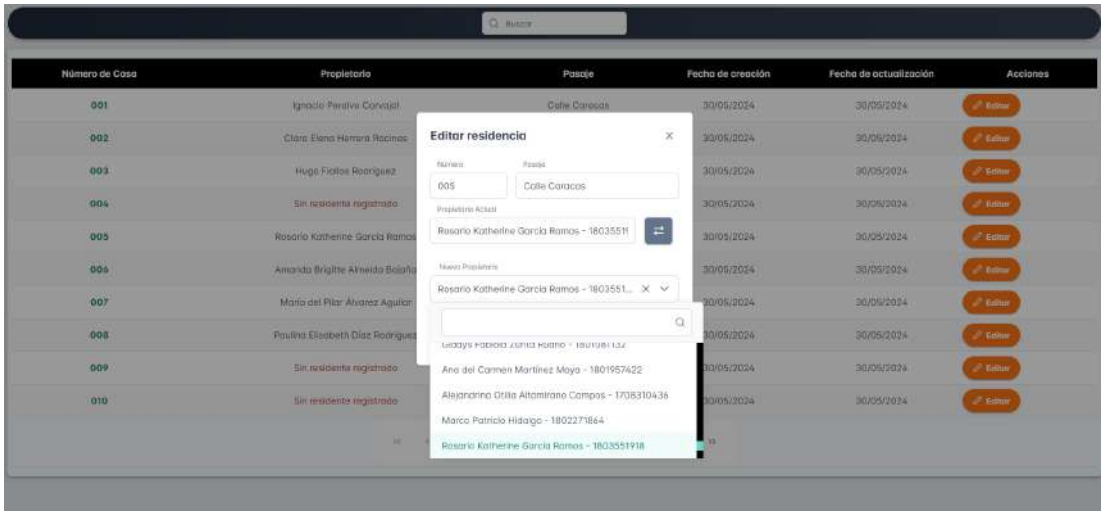


Figura 91. Formulario de actualización de usuario representante de una residencia

- ***Gestión de parqueaderos.*** Como parte de la gestión de parqueaderos el presidente visualizará un mapa con la ubicación de los parqueaderos en grupos de dos colores en donde los azules representan a los de zona azul y los amarillos a los partículares. Para poder la casa asociada a un parqueadero se debe seleccionar el parqueadero de uno de los grupos y seleccionar la casa a la que se le desea asignar el parqueadero, de esta manera se enviarán a la API el id de la casa y el id del parqueadero. La API procesará la solicitud y actualizará la casa asociada a ese parqueadero tal como se muestra en la Figura 92.

```
@Override  TigselemaAlex
public Parking update(ParkingUpdateData data, Long id) {
    Parking parking = findById(id);
    if (data.residence() == null) {
        parking.setResidence(null);
        parking.setStatus(ParkingStatus.AVAILABLE);
    }else{
        parking.setResidence(residenceService.findById(data.residence()));
        parking.setStatus(ParkingStatus.OCCUPIED);
    }
    return parkingRepository.save(parking);
}
```

Figura 92. Método de actualización de la casa asociada a un parqueadero en la API

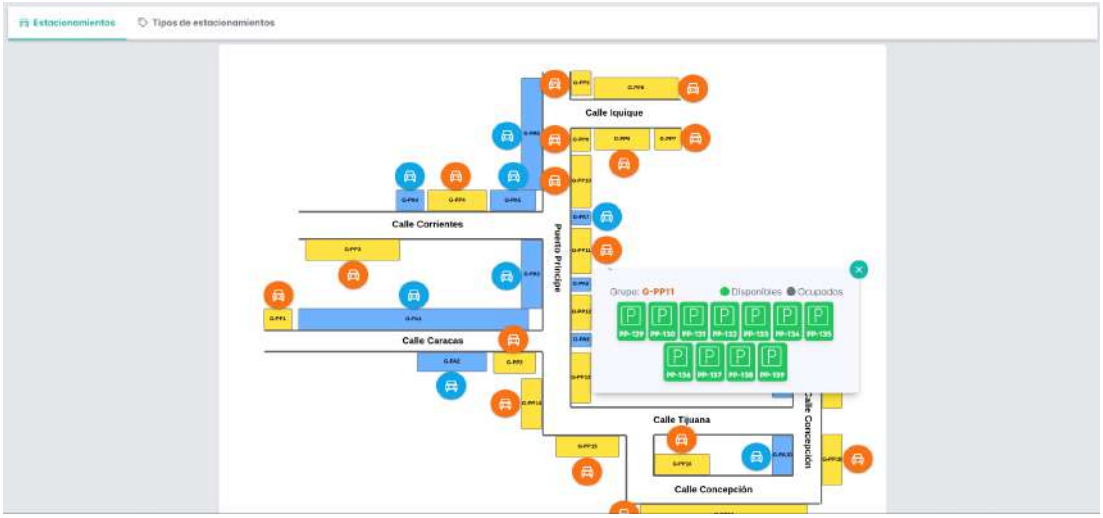


Figura 93. Página de gestión de parqueaderos

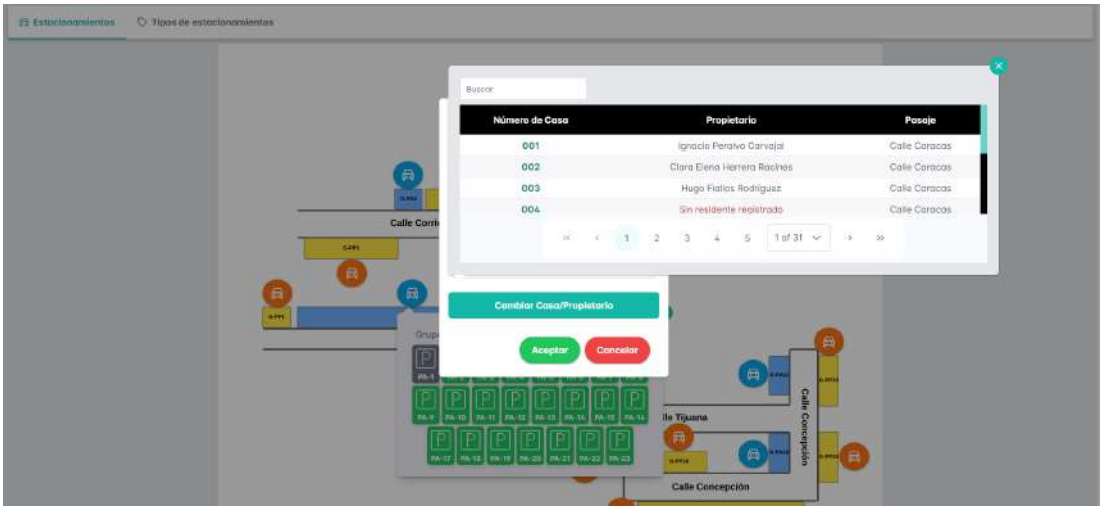


Figura 94. Formulario de actualización de la casa asociada a un parqueadero

- ***Gestión de incidentes.*** El control de los incidentes reportados por los guardías serán gestionados por el presidente y vicepresidente los cuales se deberá de proporcionar la información reportada por el guardía y en el caso de ser necesario subir evidencias del incidente. Para ello se debe enviar el id del guardía que reportó el incidente, el asunto de incidente, una descripción, el tipo de incidente y las evidencias en formato form-data a la API tal como se muestra en la Figura 95. Estos incidentes registrados podrán ser actualizados en el caso de que se lleguen a solucionar o queden inconclusos para poder generar un reporte del incidente tal como se muestra en la Figura 98.



Figura 95. Endpoint para crear un incidente

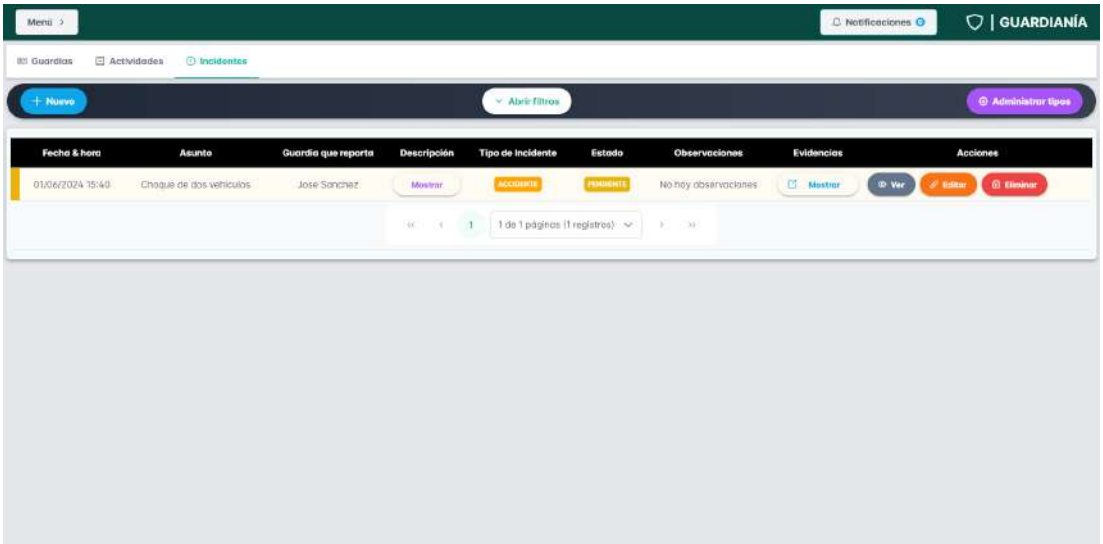


Figura 96. Página de gestión de incidentes

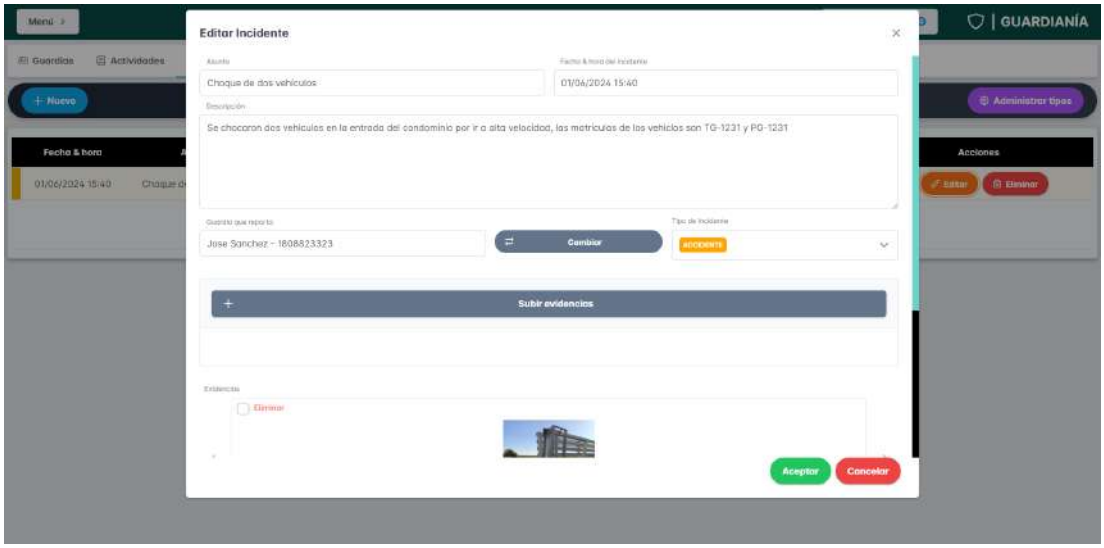
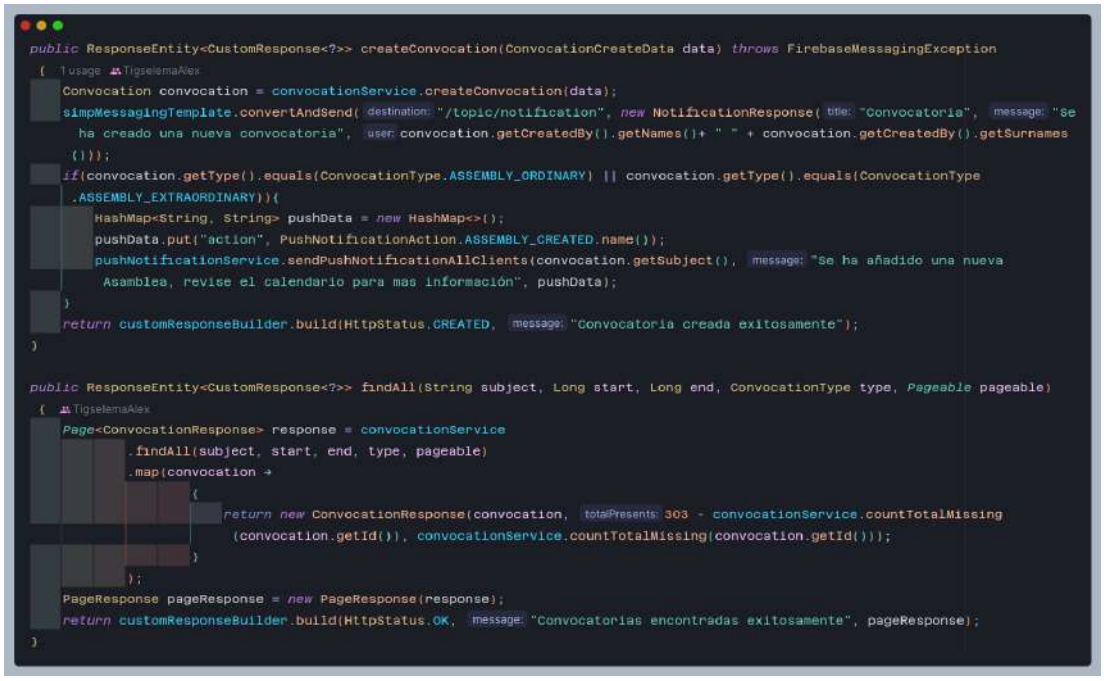


Figura 97. Formulario de creación y edición de incidentes



Figura 98. Reporte de incidente

- ***Gestión de convocatorias.*** Para el registro de convocatorias se debe establecer la fecha de inicio, de que tipo de convocatoria es ya que dependiendo de si es una asamblea se podrá establecer la fecha y hora límite para el registro de asistencia por parte de los residentes mediante el uso de la aplicación móvil, el asunto de la convocatoria y una descripción a cerca de los temas o actividades a tratar en la convocatoria. Esta información será enviada a la API en formato JSON para que ésta pueda ser registrada y notificar a los demás residentes solo en el caso de ser una asamblea tal como se muestra en la Figura 99. Adicionalmente se puede visualizar la convocatoria para ser descargada en formato PDF para poder ser compartida en los grupos de WhatsApp tal como se muestra en la Figura 102.



```
public ResponseEntity<CustomResponse<?>> createConvocation(ConvocationCreateData data) throws FirebaseMessagingException
{  1 usage  TigselemaAlex
    Convocation convocation = convocationService.createConvocation(data);
    simpMessagingTemplate.convertAndSend( destination: "/topic/notification", new NotificationResponse( title: "Convocatoria",  message: "Se
      ha creado una nueva convocatoria",  user: convocation.getCreatedBy().getNames()+ " " + convocation.getCreatedBy().getSurnames
     ()));
    if(convocation.getType().equals(ConvocationType.ASSEMBLY_ORDINARY) || convocation.getType().equals(ConvocationType
     .ASSEMBLY_EXTRAORDINARY)){
        HashMap<String, String> pushData = new HashMap<>();
        pushData.put("action", PushNotificationAction.ASSEMBLY_CREATED.name());
        pushNotificationService.sendPushNotificationAllClients(convocation.getSubject(),  message: "Se ha añadido una nueva
          Asamblea, revise el calendario para mas información", pushData);
    }
    return customResponseBuilder.build(HttpStatus.CREATED,  message: "Convocatoria creada exitosamente");
}

public ResponseEntity<CustomResponse<?>> findAll(String subject, Long start, Long end, ConvocationType type, Pageable pageable)
{  TigselemaAlex
    Page<ConvocationResponse> response = convocationService
            .findAll(subject, start, end, type, pageable)
            .map(convocation →
                    {
                        return new ConvocationResponse(convocation,  totalPresents: 303 - convocationService.countTotalMissing
                          (convocation.getId()), convocationService.countTotalMissing(convocation.getId()));
                    }
            );
    PageResponse pageResponse = new PageResponse(response);
    return customResponseBuilder.build(HttpStatus.OK,  message: "Convocatorias encontradas exitosamente", pageResponse);
}
```

Figura 99. Método de creación de una convocatoria y envio de notificación push en la API

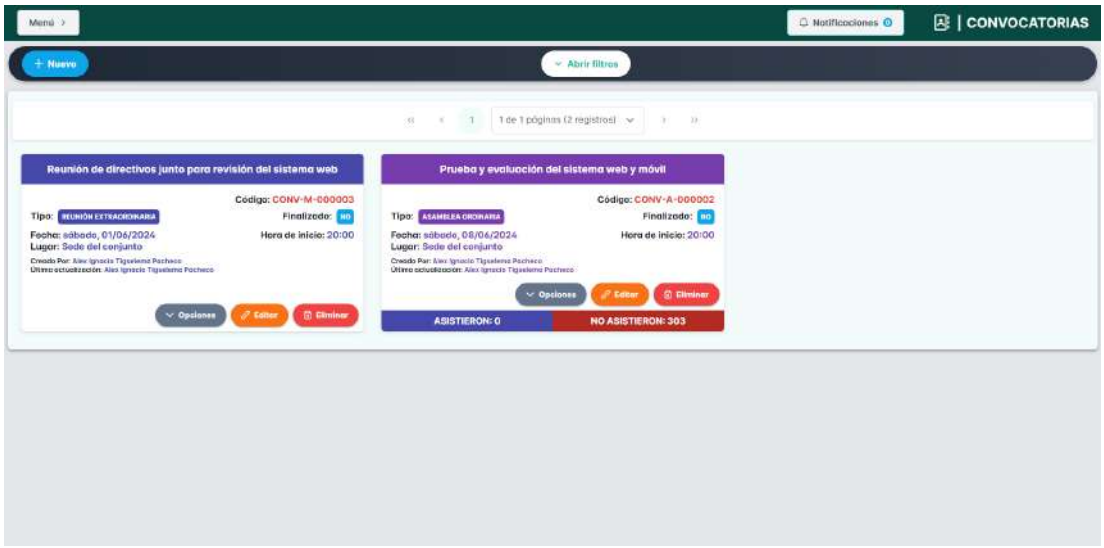


Figura 100. Página de gestión de convocatorias

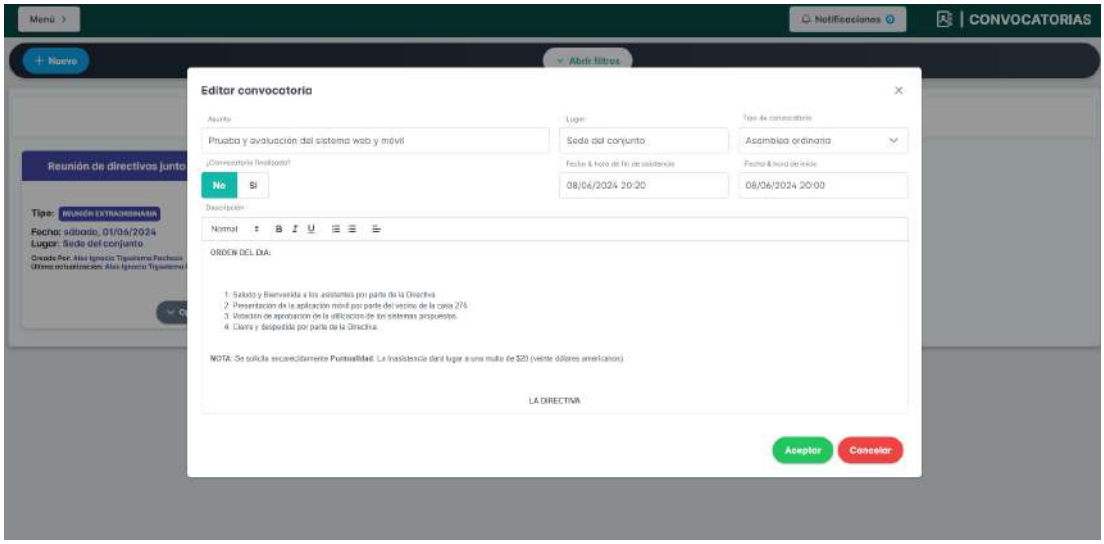


Figura 101. Formulario de creación y edición de convocatorias

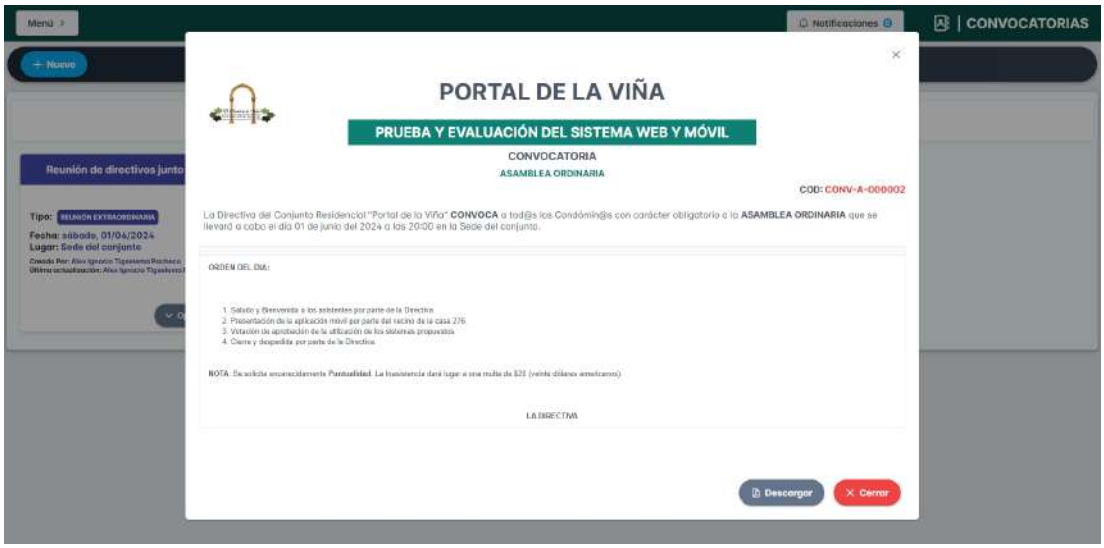
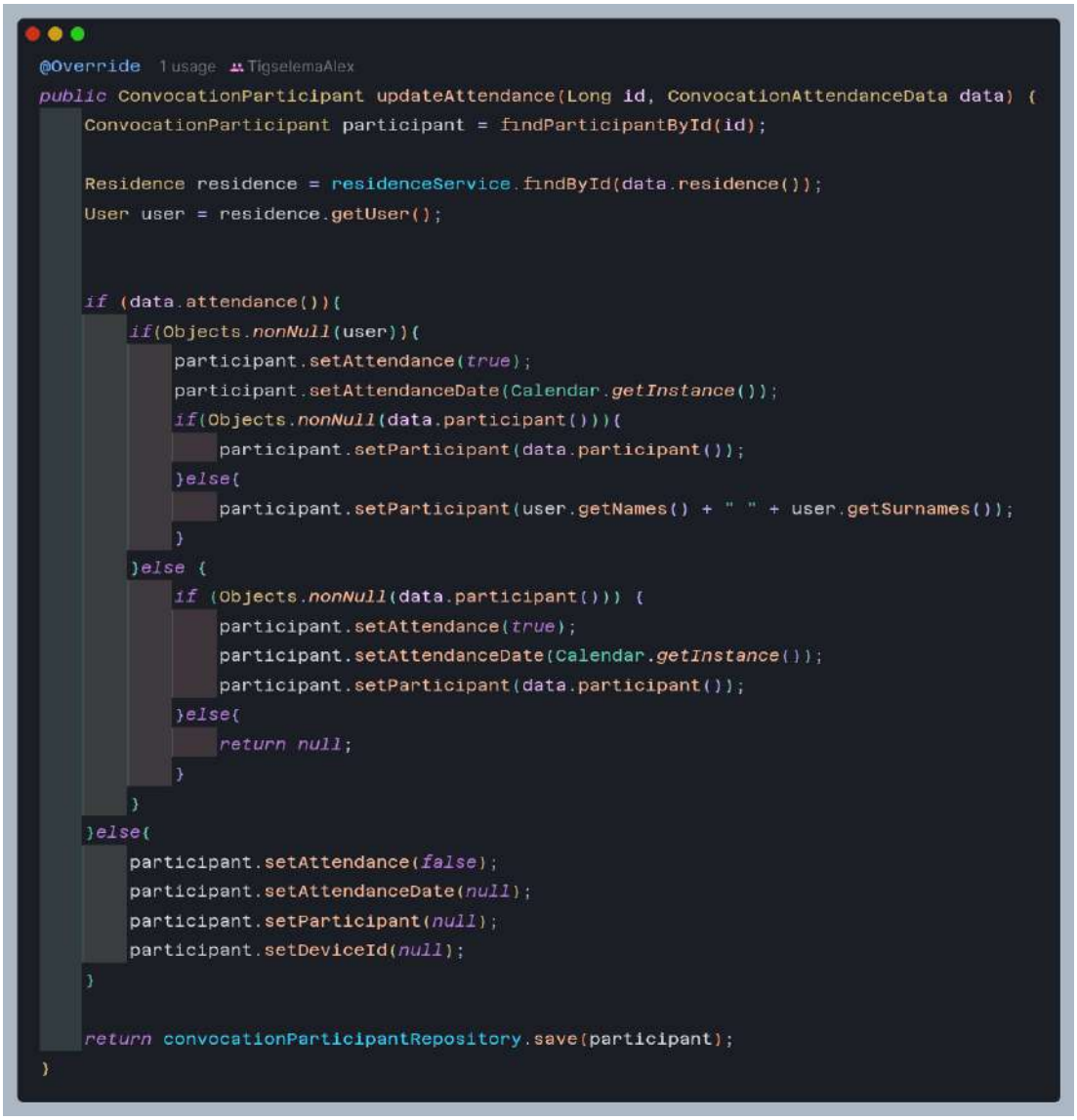


Figura 102. Vista previa a la descarga del comunidado de la convocatoria en PDF

- ***Gestión de asistencias en asambleas.*** Para las convocatorias que son asambleas tendrán habilitada la opción de poder administrar las asistencias, en donde se listarán todas las residencias junto con la información de los residentes asociados a cada una de ellas. Para el registro de asistencia manual se selecciona la casa y se coloca su estado de asistencia y en el caso de ser necesario se puede registrar el nombre de la persona que asiste representando esa residencia sin ser la persona asociada a ella. A la API se le envía el id de la convocatoria, el id de la residencia, el estado de la asistencia y opcionalmente el nombre de la persona que asiste tal como se muestra en la Figura 103. Adicionalmente se puede obtener un reporte de las casas ausentes para poder realizar su descarga e informar a los residentes por los grupos de comunicación tal como se

muestra en la Figura 105.



```
@Override  1 usage  TigselemaAlex
public ConvocationParticipant updateAttendance(Long id, ConvocationAttendanceData data) {
    ConvocationParticipant participant = findParticipantById(id);

    Residence residence = residenceService.findById(data.residence());
    User user = residence.getUser();


    if (data.attendance()){
        if(Objects.nonNull(user)){
            participant.setAttendance(true);
            participant.setAttendanceDate(Calendar.getInstance());
            if(Objects.nonNull(data.participant())){
                participant.setParticipant(data.participant());
            }else{
                participant.setParticipant(user.getNames() + " " + user.getSurnames());
            }
        }else {
            if (Objects.nonNull(data.participant())) {
                participant.setAttendance(true);
                participant.setAttendanceDate(Calendar.getInstance());
                participant.setParticipant(data.participant());
            }else{
                return null;
            }
        }
    }else{
        participant.setAttendance(false);
        participant.setAttendanceDate(null);
        participant.setParticipant(null);
        participant.setDeviceId(null);
    }

    return convocationParticipantRepository.save(participant);
}
```

Figura 103. Método del registro de asistencia manual en la API

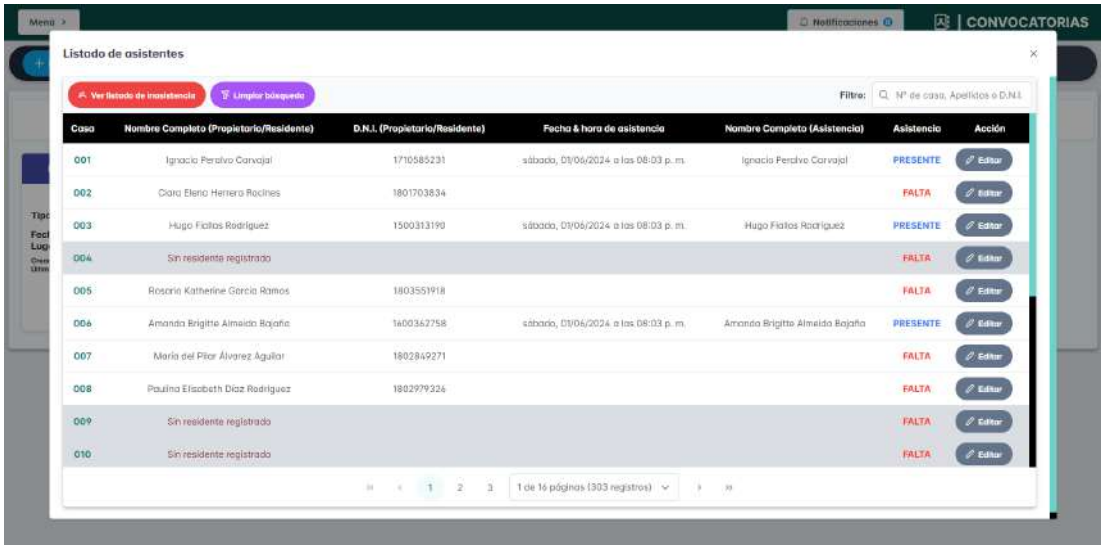


Figura 104. Página de gestión de asistencia en asambleas



Figura 105. Reporte de inasistencias en la asamblea

- ***Gestión de votación en asambleas.*** Para las convocatorias que son asambleas se pueden registrar preguntas para que sean votadas por los residentes, en donde se registrará las preguntas para lo cual solo es necesario proporcionar el enunciado de la pregunta y enviarla a la API en formato JSON como se muestra en la Figura 106. Una vez registrada la pregunta se debe habilitar para que los residentes puedan realizar la votación a través de la aplicación móvil, adicionalmente se podrán ver más detalladamente los votos de cada pregunta 110, para lo cual se le debe enviar a la API el id de la pregunta por la ruta tal como se muestra en la Figura 108.

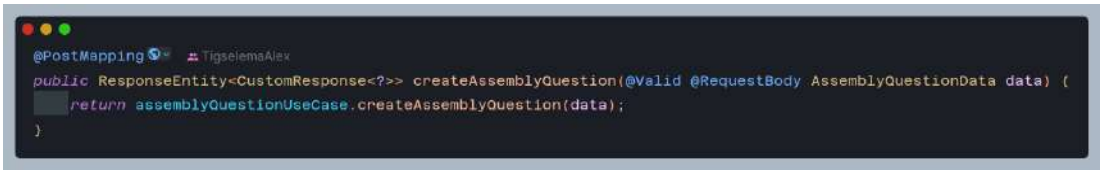


Figura 106. Endpoint para crear una pregunta de asamblea



Figura 107. Endpoint para obtener todas las preguntas de asamblea



Figura 108. Endpoint para habilitar o deshabilitar una pregunta de asamblea

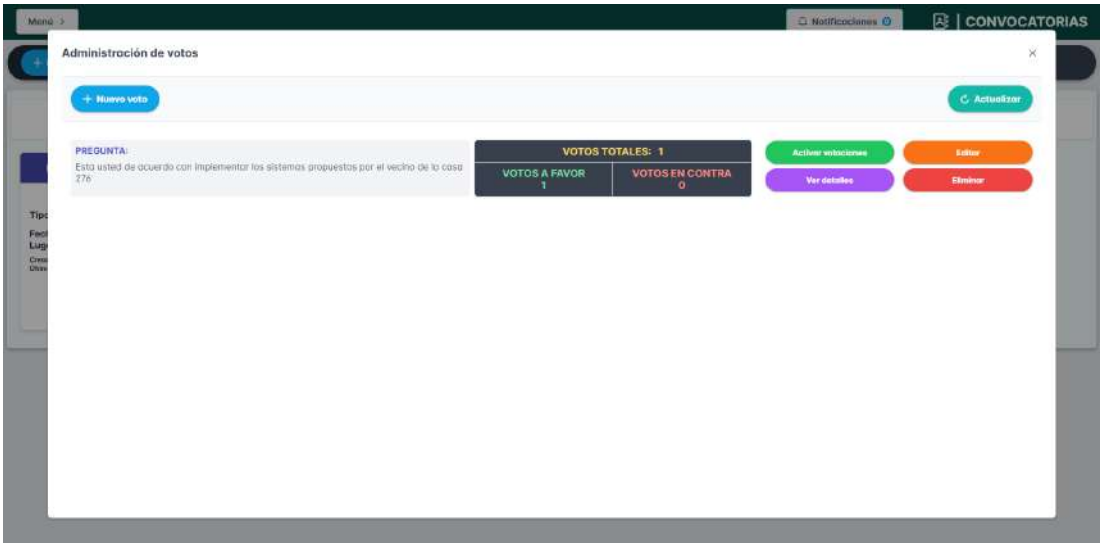


Figura 109. Lista de preguntas de asamblea

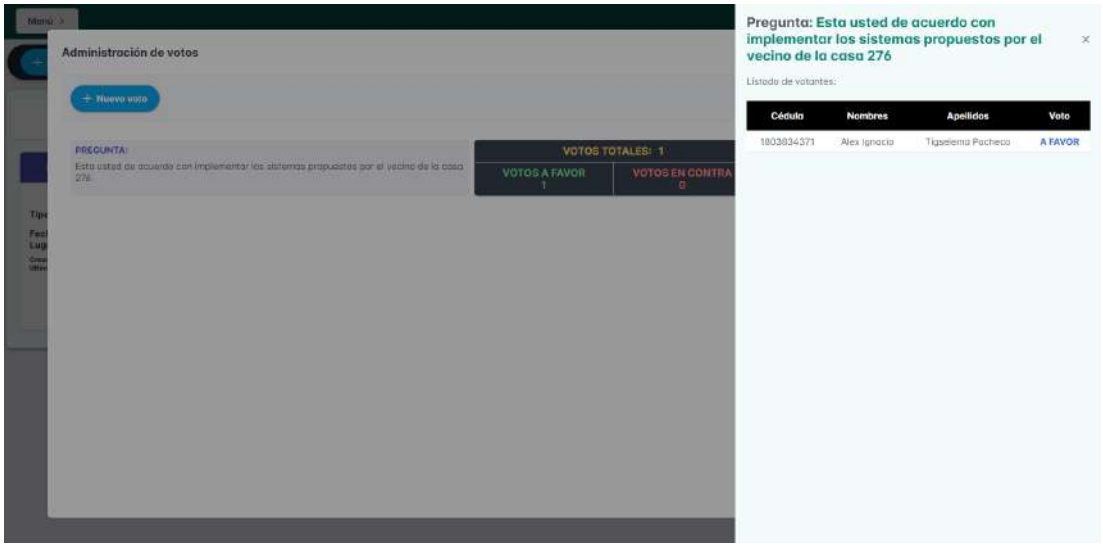


Figura 110. Listado detallado de votaciones de una pregunta de asamblea

- ***Gestión de ingresos mensuales.*** Para el registro de un ingreso mensual se debe proporcionar el o los meses de abono, la residencia a la que se le registra el pago junto con un comprobante de pago en formato de imagen, la fecha del pago y el monto se calcula automáticamente según el tipo de pago mensual tal y como se muestra en la Figura 113. Esta información será enviada a la API en formato form-data para que ésta pueda ser procesada y almacenada en la base de datos tal como se muestra en la Figura 114.

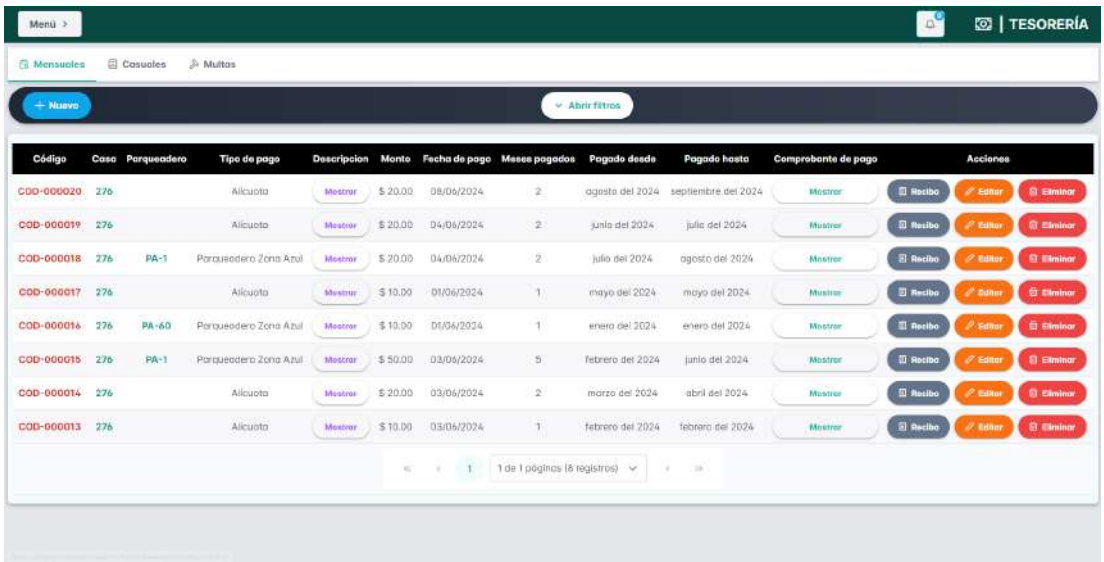


Figura 111. Listado de los ingresos de tipo mensual



Figura 112. Endpoint para listar los ingresos mensuales

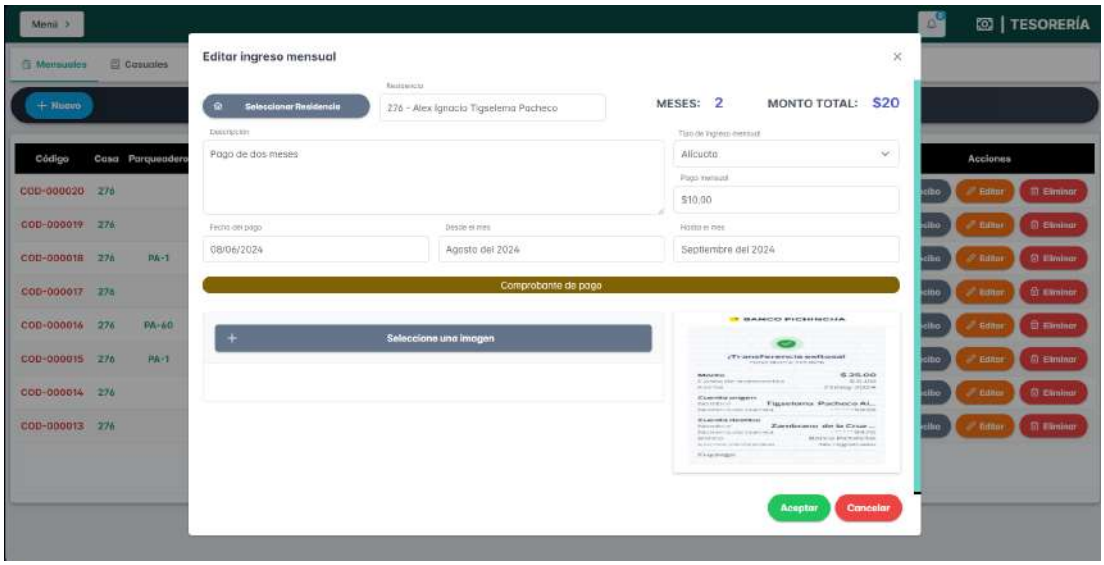


Figura 113. Formulario de creación y edición de ingresos mensuales



Figura 114. Endpoint para guardar un ingreso mensual

- ***Gestión de multas.*** Para el registro de multas se realiza un proceso similar al de los ingresos mensuales, salvo por la diferencia que este pago no tiene meses de abono ya que es un pago único, se debe proporcionar la residencia a que se le registra la multa,

la cual puede estar o no pagada, junto con su comprobante en el caso de ser pagad, y adicionalmente se pueden adjuntar evidencias tal y como se muestra en la Figura 117. Esta información será enviada a la API en formato form-data para que ésta pueda ser procesada y almacenada en la base de datos.

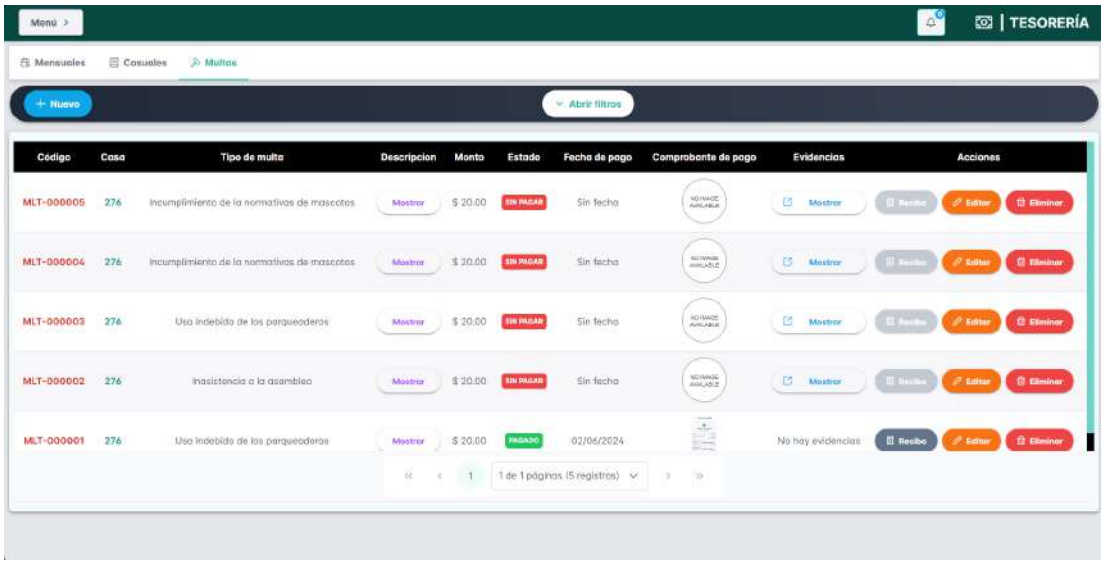


Figura 115. Listado de multas

```java
@GetMapping("/active")  TigselemaAlex
public ResponseEntity<CustomResponse<?>> findAllActive(
        @RequestParam(required = false) Long type,
        @RequestParam(required = false) Long from,
        @RequestParam(required = false) Long to,
        @RequestParam(required = false) Long residence,
        @RequestParam(required = false) PaidStatus status,
        @PageableDefault(
                size = 20,
                sort = "createdAt",
                direction = Sort.Direction.DESC)
        Pageable pageable) {
    return penaltyUseCase.findAllActive(type, from, to, residence, status, pageable);
}
```

Figura 116. Endpoint para listar las multas

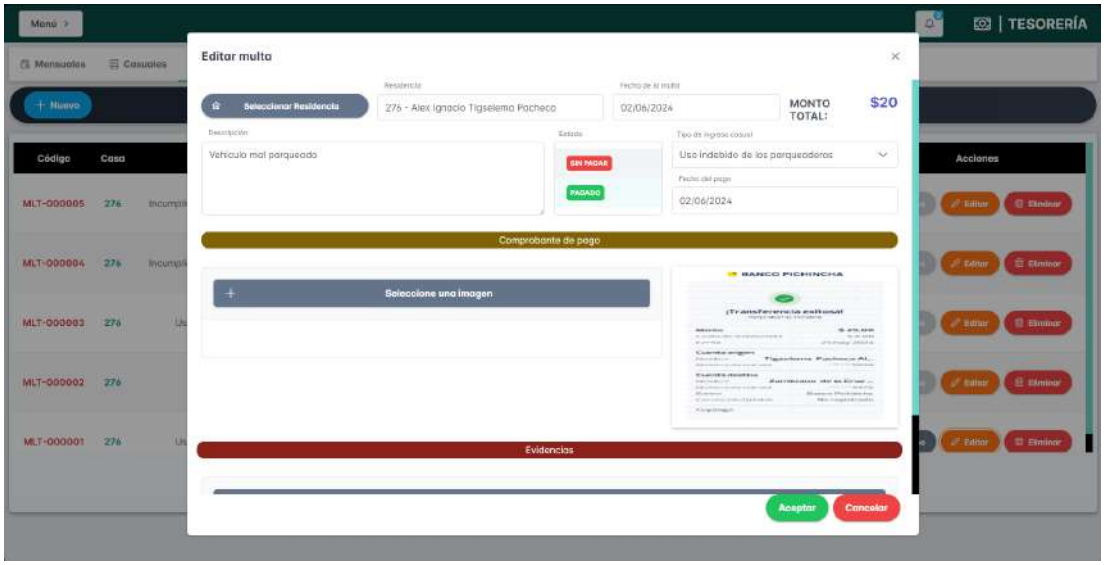


Figura 117. Formulario de creación y edición de multas

- ***Gestión de egresos.*** El registro de egresos se realizan proporcionando el concepto del egreso, la fecha en la que se realizó el egreso, el monto del egreso y el comprobante de pago en formato de imagen o en formato PDF, junto con el proveedor del servicio o producto adquirido y el tipo de egreso tal y como se muestra en la Figura 120. La información es enviado a la API en formato form-data para que ésta pueda ser procesada y almacenada en la base de datos tal como se muestra en la Figura 121.

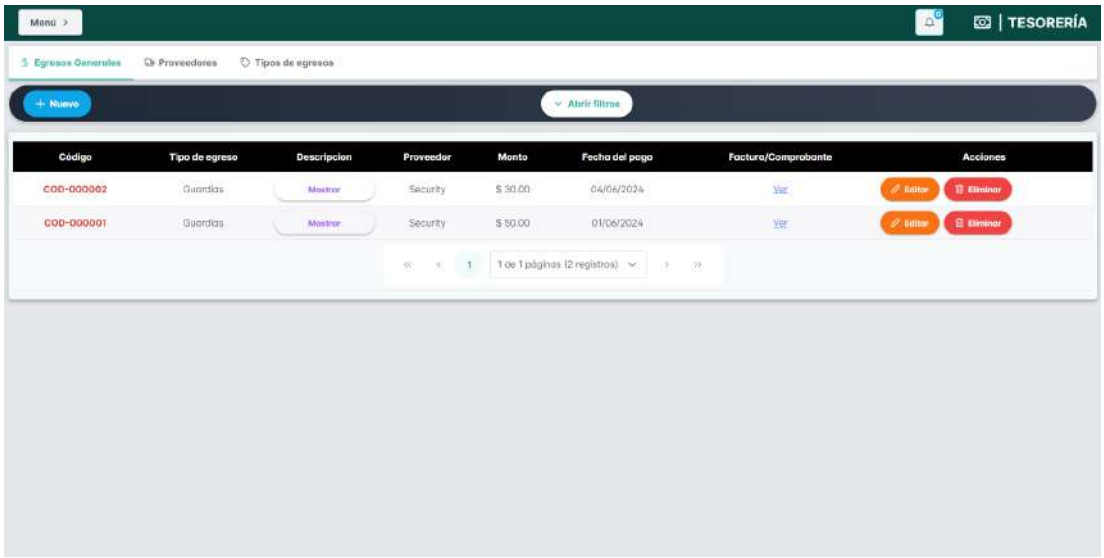


Figura 118. Listado de los egresos



Figura 119. Endpoint para listar los egresos

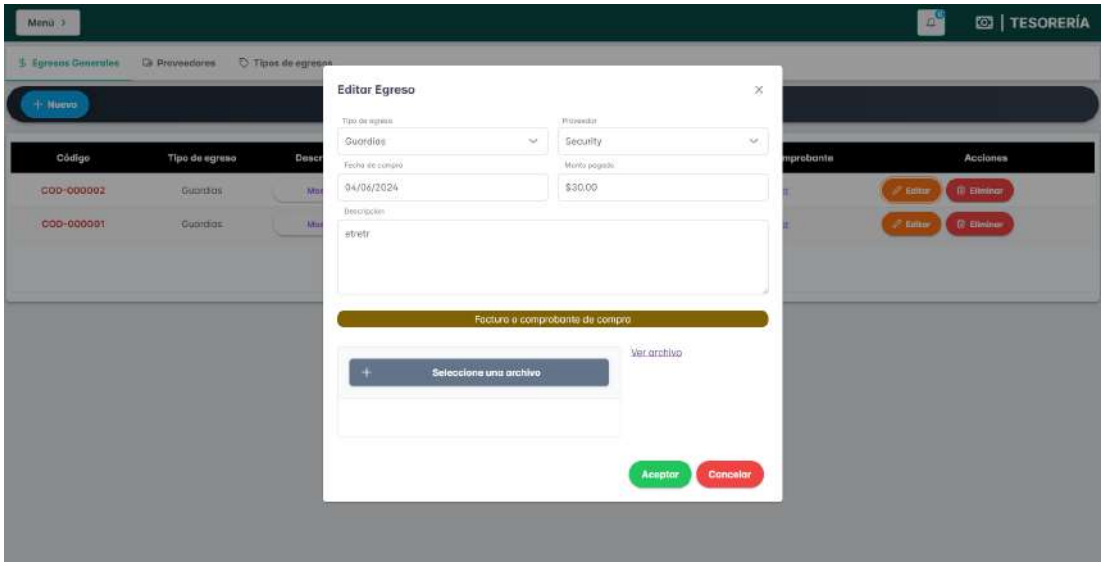


Figura 120. Formulario de creación y edición de egresos

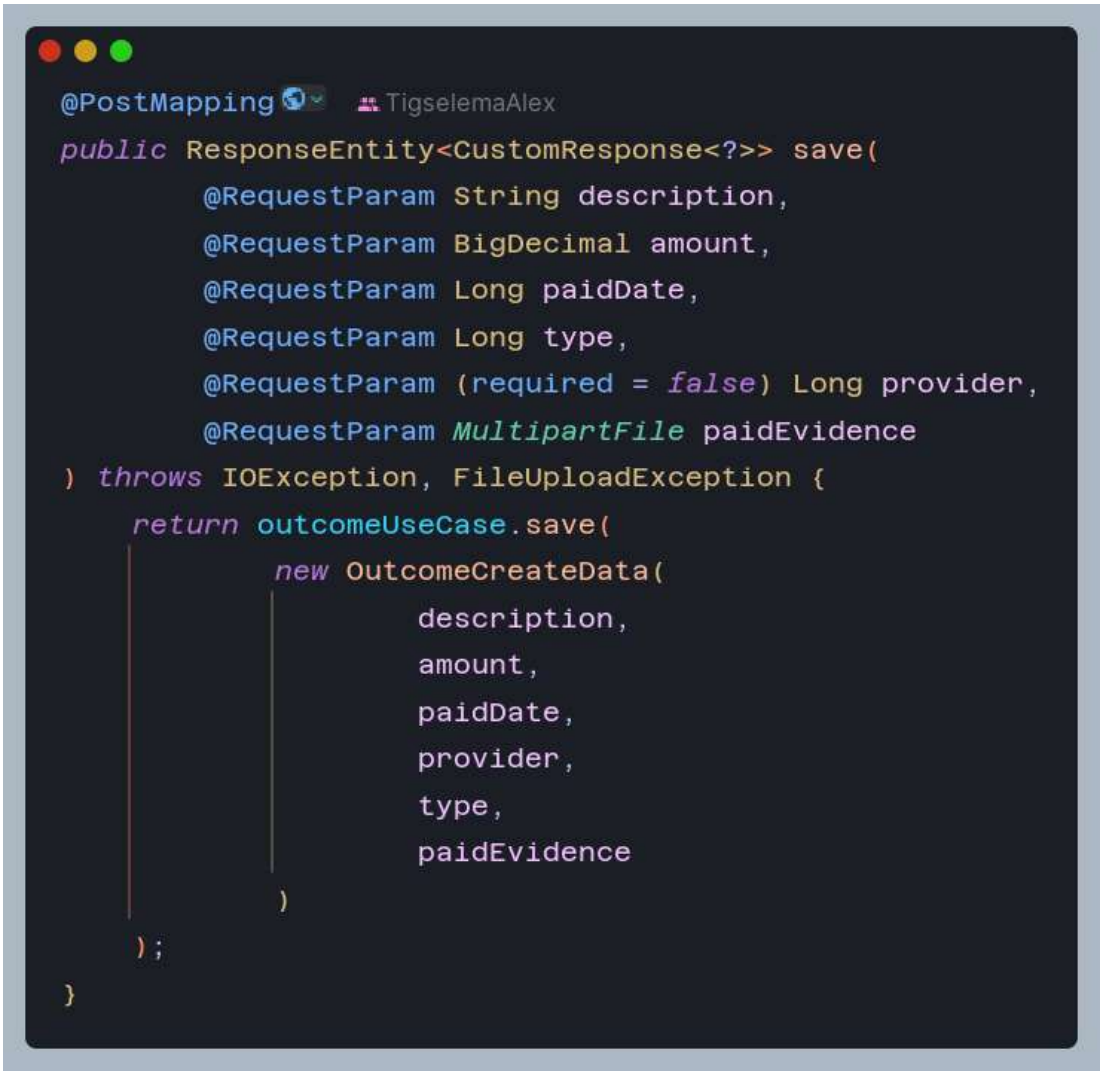


Figura 121. Endpoint para guardar un ingreso mensual

Aplicación Móvil

- ***Dependencias de la aplicación.*** Para desarrollar la aplicación móvil se utilizarón algunas dependencias extras de Capacitor como lo son el plugin de geolocalización, el plugin de notificaciones push y el plugin de dispositivos para poder obtener la información del dispositivo en donde se instala la aplicación como se pueden observar en la Figura 122. También se utilizó TailwindCss como librería de estilos para el diseño de la aplicación.



```
"dependencies": {
    "@angular/animations": "^17.0.2",
    "@angular/common": "^17.0.2",
    "@angular/compiler": "^17.0.2",
    "@angular/core": "^17.0.2",
    "@angular/forms": "^17.0.2",
    "@angular/platform-browser": "^17.0.2",
    "@angular/platform-browser-dynamic": "^17.0.2",
    "@angular/router": "^17.0.2",
    "@capacitor/android": "6.0.0",
    "@capacitor/app": "6.0.0",
    "@capacitor/core": "6.0.0",
    "@capacitor/device": "^6.0.0",
    "@capacitor/geolocation": "^6.0.0",
    "@capacitor/haptics": "6.0.0",
    "@capacitor/keyboard": "6.0.0",
    "@capacitor/push-notifications": "^6.0.0",
    "@capacitor/status-bar": "6.0.0",
    "@ionic/angular": "^8.0.0",
    "@ionic/storage-angular": "^4.0.0",
    "ionicons": "^7.2.1",
    "rxjs": "~7.8.0",
    "tslib": "^2.3.0",
    "zone.js": "~0.14.2"
  },
  "devDependencies": {
    "@angular-devkit/build-angular": "^17.0.0",
    "@angular-eslint/builder": "^17.0.0",
    "@angular-eslint/eslint-plugin": "^17.0.0",
    "@angular-eslint/eslint-plugin-template": "^17.0.0",
    "@angular-eslint/schematics": "^17.0.0",
    "@angular-eslint/template-parser": "^17.0.0",
    "@angular/cli": "^17.0.0",
    "@angular/compiler-cli": "^17.0.2",
    "@angular/language-service": "^17.0.2",
    "@capacitor/cli": "6.0.0",
    "@ionic/angular-toolkit": "^11.0.1",
    "@types/jasmine": "~5.1.0",
    "@typescript-eslint/eslint-plugin": "^6.0.0",
    "@typescript-eslint/parser": "^6.0.0",
    "autoprefixer": "^10.4.19",
    "eslint": "^8.57.0",
    "eslint-plugin-import": "^2.29.1",
    "eslint-plugin-jsdoc": "^48.2.1",
    "eslint-plugin-prefer-arrow": "1.2.2",
    "jasmine-core": "~5.1.0",
    "jasmine-spec-reporter": "~5.0.0",
    "karma": "~6.4.0",
    "karma-chrome-launcher": "~3.2.0",
    "karma-coverage": "~2.2.0",
    "karma-jasmine": "~5.1.0",
    "karma-jasmine-html-reporter": "~2.1.0",
    "postcss": "^8.4.38",
    "tailwindcss": "^3.4.3",
    "typescript": "~5.2.2"
  },
```

Figura 122. Dependencias de la applicación móvil hecha con Ionic

- ***Configuración de la aplicación.*** Se debe configurar la aplicación móvil para que se puedan realizar las peticiones a la API, para ello se debe establecer la URL de la API en el archivo *environment.ts* como se muestra en la Figura 123, y adicionalmente

configurar el archivo *main.ts* para colocar los proveedores de peticiones HTTP, el proveedor de enrutamiento y los servicios de almacenamiento.

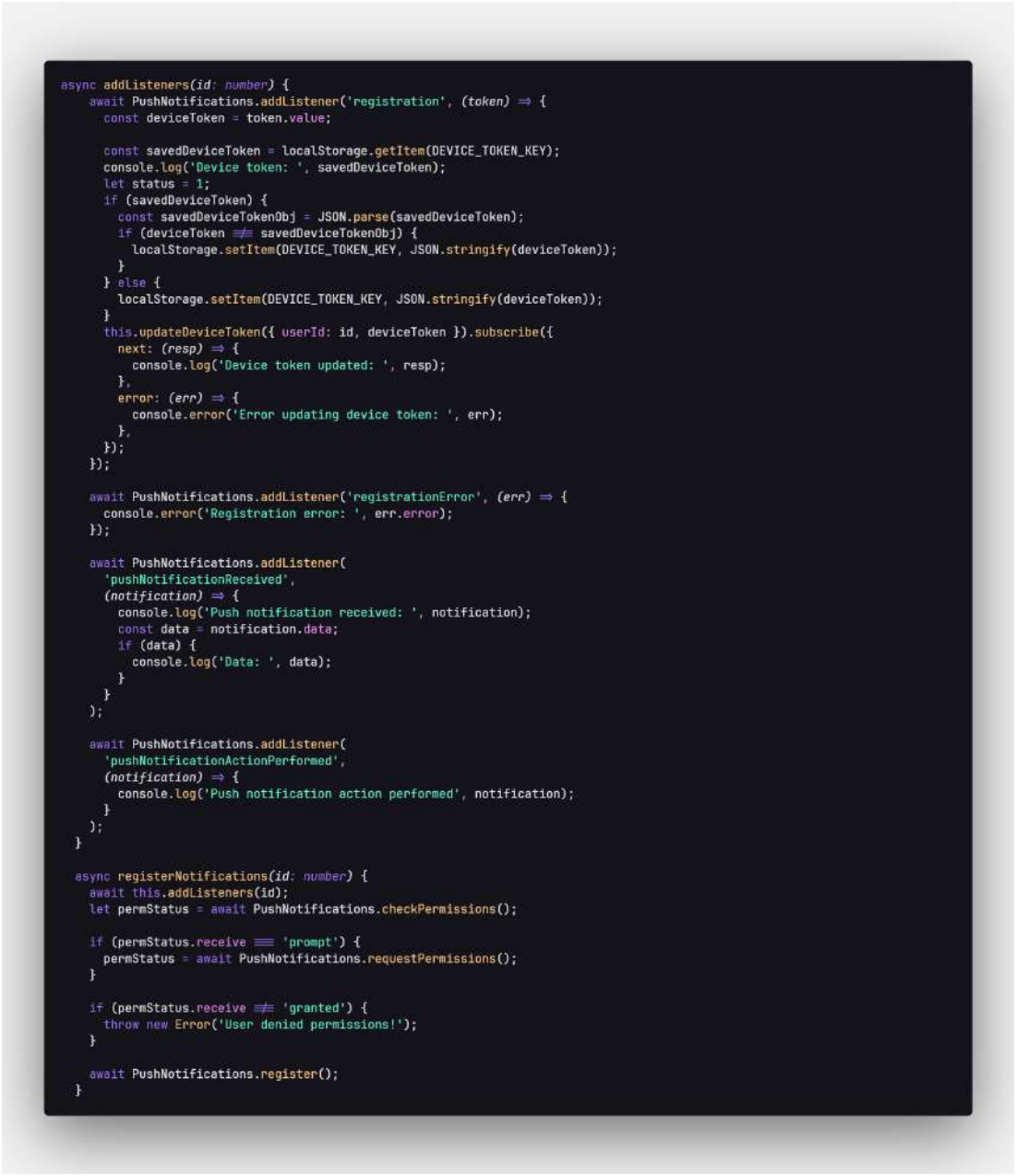
```
export const environment = {
  production: true,
  API_URL: 'https://www.api.portaldelavinia.com/portal/api/v1',
};
```

Figura 123. Configuración de la URL de la API en la aplicación móvil

```
bootstrapApplication(AppComponent, {
  providers: [
    { provide: RouteReuseStrategy, useClass: IonicRouteStrategy },
    provideIonicAngular(),
    provideRouter(routes),
    provideHttpClient(withInterceptors([authInterceptorInterceptor]))
,
    {
      provide: LOCALE_ID,
      useValue: 'es-EC',
    },
    StorageService,
    Storage,
  ],
});
```

Figura 124. Codigo del archivo de configuración principal de la aplicación móvil

También es importante configurar Firebase para el envío de Notificaciones Push al dispositivo, para ello se debe crear un servicio en donde se establecerán el registro y los métodos de escucha de las notificaciones tal como se muestra en la Figura 125. Adicionalmente es necesario copiar el archivo JSON *google.services.json* de Firebase en la carpeta *android/app/src* de la aplicación móvil y agregar en el archivo *variables.gradle* la referencia al plugin de Firebase como se muestra en la Figura 127.



```
async addListeners(id: number) {
    await PushNotifications.addListener('registration', (token) => {
      const deviceToken = token.value;

      const savedDeviceToken = localStorage.getItem(DEVICE_TOKEN_KEY);
      console.log('Device token: ', savedDeviceToken);
      let status = 1;
      if (savedDeviceToken) {
        const savedDeviceTokenObj = JSON.parse(savedDeviceToken);
        if (deviceToken !== savedDeviceTokenObj) {
          localStorage.setItem(DEVICE_TOKEN_KEY, JSON.stringify(deviceToken));
        }
      } else {
        localStorage.setItem(DEVICE_TOKEN_KEY, JSON.stringify(deviceToken));
      }
      this.updateDeviceToken({ userId: id, deviceToken }).subscribe({
        next: (resp) => {
          console.log('Device token updated: ', resp);
        },
        error: (err) => {
          console.error('Error updating device token: ', err);
        },
      });
    });

    await PushNotifications.addListener('registrationError', (err) => {
      console.error('Registration error: ', err.error);
    });

    await PushNotifications.addListener(
      'pushNotificationReceived',
      (notification) => {
        console.log('Push notification received: ', notification);
        const data = notification.data;
        if (data) {
          console.log('Data: ', data);
        }
      }
    );

    await PushNotifications.addListener(
      'pushNotificationActionPerformed',
      (notification) => {
        console.log('Push notification action performed', notification);
      }
    );
  }

  async registerNotifications(id: number) {
    await this.addListeners(id);
    let permStatus = await PushNotifications.checkPermissions();

    if (permStatus.receive === 'prompt') {
      permStatus = await PushNotifications.requestPermissions();
    }

    if (permStatus.receive !== 'granted') {
      throw new Error('User denied permissions!');
    }

    await PushNotifications.register();
  }
```

Figura 125. Codigo del servicio de Notificaciones Push en la aplicación móvil

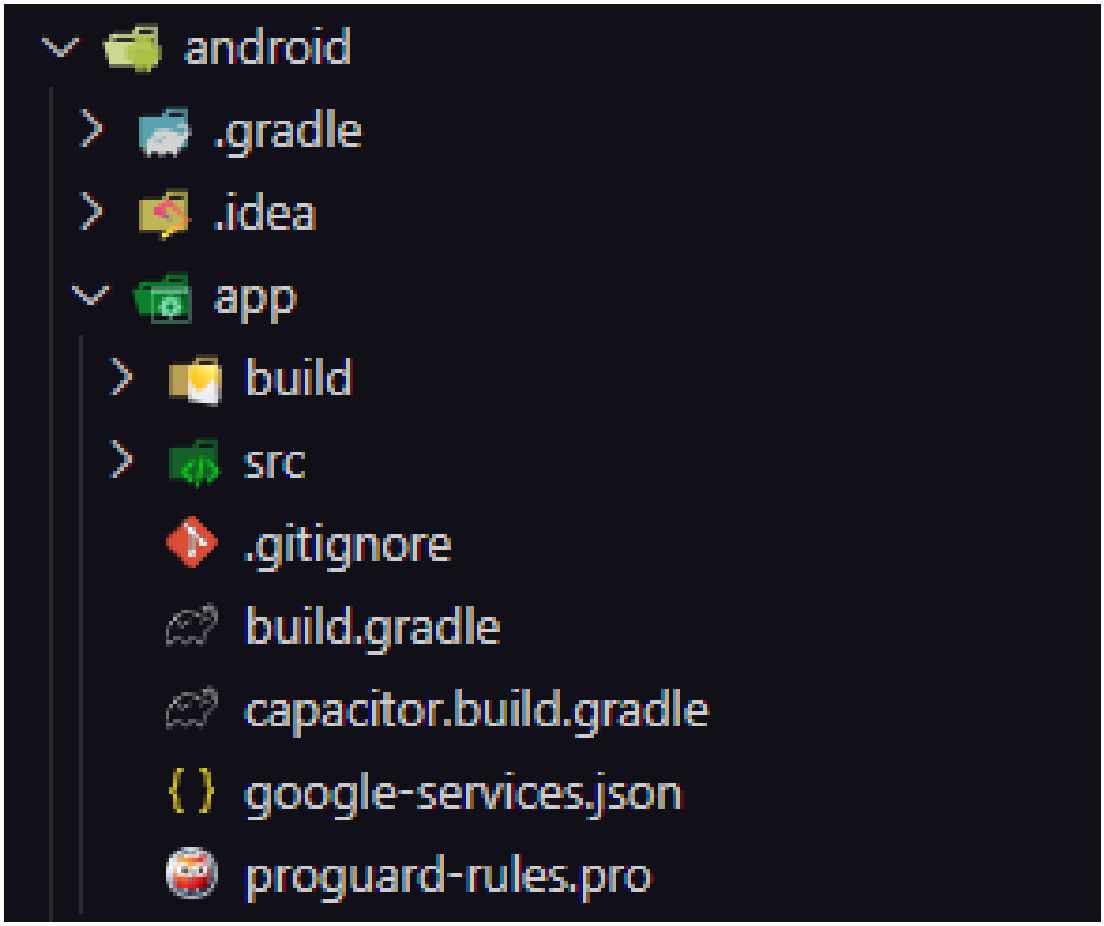


Figura 126. Ubicación del archivo *google.services.json*

```
ext {
    minSdkVersion = 22
    compileSdkVersion = 34
    targetSdkVersion = 34
    androidxActivityVersion = '1.8.0'
    androidxAppCompatVersion = '1.6.1'
    androidxCoordinatorLayoutVersion = '1.2.0'
    androidxCoreVersion = '1.12.0'
    androidxFragmentVersion = '1.6.2'
    coreSplashScreenVersion = '1.0.1'
    androidxWebkitVersion = '1.9.0'
    junitVersion = '4.13.2'
    androidxJunitVersion = '1.1.5'
    androidxEspressoCoreVersion = '3.5.1'
    cordovaAndroidVersion = '10.1.1'
    firebaseMessagingVersion = '23.3.1'
}
```

Figura 127. Código del archivo *variables.gradle* con la referencia al plugin de Firebase

Finalmente en el archivo *manifest.xml* se debe establecer los permisos necesarios para

el uso de la geolocalización y notificaciones push como se muestra en la Figura 128.

```xml
<?xml version='1.0' encoding='utf-8'?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android">
    <application android:allowBackup="true" android:icon="@mipmap/ic_launcher" android:label="@string/app_name" android:roundIcon=
"@mipmap/ic_launcher_round" android:supportsRtl="true" android:theme="@style/AppTheme" android:usesCleartextTraffic="true">
        <activity android:configChanges="orientation|keyboardHidden|keyboard|screenSize|locale|smallestScreenSize|screenLayout|uiMode"
android:exported="true" android:label="@string/title_activity_main" android:launchMode="singleTask" android:name=".MainActivity"
android:screenOrientation="portrait" android:theme="@style/AppTheme.NoActionBarLaunch">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <provider android:authorities="${applicationId}.fileprovider" android:exported="false" android:grantUriPermissions="true"
android:name="androidx.core.content.FileProvider">
            <meta-data android:name="android.support.FILE_PROVIDER_PATHS" android:resource="@xml/file_paths" />
        </provider>
        <meta-data android:name="com.google.firebase.messaging.default_notification_icon" android:resource="@mipmap/ic_launcher" />
    </application>
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-feature android:name="android.hardware.location.gps" />
</manifest>
```

Figura 128. Código del archivo *manifest.xml* con los permisos necesarios

- ***Inicio de sesión.*** Para el inicio de sesión el usuario debe proporcionar sus credenciales las cuales serán enviadas a la API para que sean válidadas y en el caso de ser correctas generar un JWT que será almacenando de manera local en el dispositivo móvil para ser utilizado en las demás peticiones a través de un interceptor.



Figura 129. Página de inicio de sesión

- ***Visualizar calendario.*** Los usuarios podrán visualizar los eventos próximos al dia actual y podrán ver detalladamente cada evento. Para ello se toma la fecha actual desde la API y se evalúan todos las convocatorias que sean de tipo asamblea y esta es enviada como respuesta para obtener los eventos próximos a la fecha actual tal como se muestra en la Figura 130.



```
@Override  1 usage  TigselemaAlex
public Page<Convocation> findAllByActiveIsTrueAndTypeInAndDateGreaterThanEqual(Pageable pageable) {
    Calendar now = Calendar.getInstance();
    now.set(Calendar.HOUR_OF_DAY, 0);
    now.set(Calendar.MINUTE, 0);
    now.set(Calendar.SECOND, 0);
    List<ConvocationType> types = List.of(ConvocationType.ASSEMBLY_EXTRAORDINARY, ConvocationType.ASSEMBLY_ORDINARY);
    return convocationRepository.findAllByActiveIsTrueAndTypeInAndDateGreaterThanEqual(types, now, pageable);
}
```

Figura 130. Método para obtener las convocatorias próximas



Figura 131. Página de las próximas asambleas



Figura 132. Detalle de una asamblea

- ***Visualizar estado de las obligaciones financieras.*** En la Figura 134 se muestra el código de los endpoints de la API para obtener las obligaciones financieras de un usuario en particular, en donde se obtiene el id del usuario que ha iniciado sesión y se envía en la ruta de las peticiónes a la API y esta devuelve un conjunto de listados en las cuales se encuentran las los últimos diez pagos realizados por el usuario tanto en multas como en mensualidades, las multas sin pagar, y el estado por residencia del usuario en el cual se encuentran el pago de alícuotas y el pago de parqueaderos azules en el caso de que éste posea alguno tal como se muestra en la Figura 133.



Figura 133. Página de el estado de las obligaciones financieras

```java
@GetMapping("/status/{id}")  TigselemaAlex
public ResponseEntity<CustomResponse<?>> getFinancialObligationsStatus(@PathVariable Long id) {
    return financialObligationsUseCase.getFinancialObligations(id);
}

@GetMapping("/{id}")  TigselemaAlex
public ResponseEntity<CustomResponse<?>> getFinancialObligations(@PathVariable Long id) {
    return financialObligationsUseCase.getFinancialObligationsStatus(id);
}
```

Figura 134. Endpoints para obtener las obligaciones financieras de un usuario



Figura 135. Página de los últimos pagos realizados por el usuario



Figura 136. Página de las multas sin pagar

- ***Registrar asistencia.*** Como se muestra en la Figura 137 el se envía la ubicación actual del dispositivo móvil y el id del usuario que ha iniciado sesión y en un cuerpo la latitud, longitud, la residencia a la que esta registrando como asistente y un el identificador del dispositivo móvil para poder registrar la asistencia en la asamblea. La API validará la ubicación a través del calculo de Haversine y en el caso de que la distancia sea menor al radio de aceptación se procederá a registrar la asistencia en la asamblea tal como se muestra en la Figura 139.



```
const coordinates = await Geolocation.getCurrentPosition();

      const deviceId = await Device.getId();

      const data: ConvocationParticipantAttendanceData = {
        latitude: coordinates.coords.latitude,
        longitude: coordinates.coords.longitude,
        residence: asistant.residence.id,
        deviceId: deviceId.identifier,
      };

      this.convocationService
        .updateParticipantAttendance(asistant.id, data)
        .subscribe({
          next: (response) => {
            toast.message = response.message;
            toast.cssClass = 'success';
            setTimeout(() => {
              loading.dismiss();
              toast.present();
            }, 300);
            this.getParticipants();
          },
          error: (error) => {
            if (error.error) {
              toast.message = error.error.message;
            } else {
              toast.message =

'Error al registrar asistencia. Intente nuevamente.';
            }
            toast.duration = 3000;
            toast.color = 'danger';
            setTimeout(() => {
              loading.dismiss();
              toast.present();
            }, 300);
          },
        });
```

Figura 137. Código para registrar la asistencia en la asamblea



Figura 138. Endpoint para registrar la asistencia en la asamblea

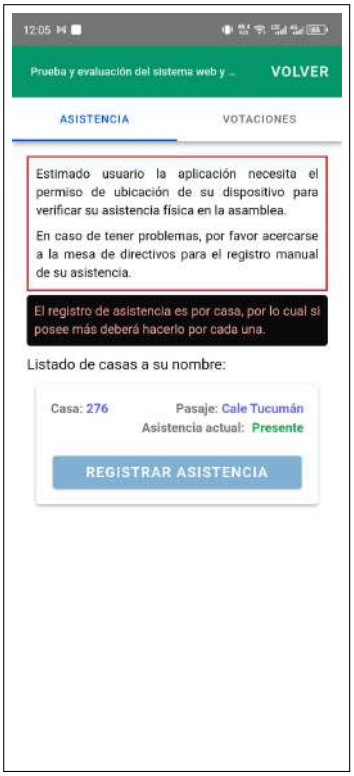


Figura 139. Página de asistencia en la asamblea

- ***Registrar voto.*** Como se muestra en la Figura 140, el registro del voto se realiza enviando el id del usuario, el id de la pregunta, el voto, el identificador del dispositivo móvil y la ubicación actual del dispositivo móvil para poder registrar el voto en la pregunta de la asamblea de una manera muy similar a como se realiza en el registro de la asistencia. En este caso la API validará que el usuario que esta votando no sea un inquilino y que tenga colocada la asistencia en la asamblea para poder registrar el voto tal como se muestra en la Figura 141.

```java
public ResponseEntity<CustomResponse<?>> updateVote(ParticipantVoteData voteData) throws IOException {  1 usage  TigselemaAlex

    GeolocationData data = null;
    File file = new File(CONFIG_FILE_PATH);
    if (file.exists()) {
        data = objectMapper.readValue(file, GeolocationData.class);
    }
    if (data == null) {
        return customResponseBuilder.build(HttpStatus.INTERNAL_SERVER_ERROR, message: "Error al obtener la configuración de
         geolocalización");
    }else{
        BigDecimal pointLatitud = data.latitude();
        BigDecimal pointLongitud = data.longitude();
        Integer radius = data.radius();
        double distance = calculateHaversineDistance(pointLatitud.doubleValue(), pointLongitud.doubleValue(), voteData
         .geolocation().latitude(), voteData.geolocation().longitude());
        boolean isInside = distance <= radius;
        if(isInside) {
            AssemblyQuestion assemblyQuestion = assemblyQuestionService.updateVote(voteData);
            if (assemblyQuestion == null) {
                return customResponseBuilder.build(HttpStatus.BAD_REQUEST, message: "No puede votar en esta pregunta de la
                 asamblea");
            }
            AssemblyQuestionResponse response = new AssemblyQuestionResponse(assemblyQuestion);
            return customResponseBuilder.build(HttpStatus.OK, message: "Voto actualizado con éxito", response);
        }else{
            return customResponseBuilder.build(HttpStatus.BAD_REQUEST, message: "No se encuentra en el rango de la
             geolocalización");
        }
    }

}
```

Figura 140. Método para registrar el voto en la asamblea

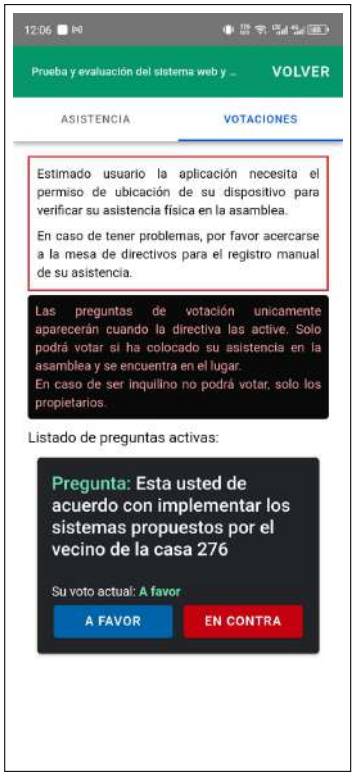


Figura 141. Página de las votaciones de la asamblea

3.5.4 Transición

La transición de la aplicación a producción se realizó en dos fases, la primera fase fue la de pruebas de aceptación en donde se validaron los criterios de aceptación definidos en las tablas a continuación con las cuales los usuarios validarán las diferentes funcionalidades tanto de la aplicación móvil como de la web para que de esta manera se asegure el cumplimiento de los requisitos definidos en la primera fase de la metodología y la segunda fase fue la de despliegue en producción en donde se realizó la configuración de los servidores y se subió la aplicación a producción.

Tabla 21. Pruebas de aceptación - Primera iteración

N.	Criterio	Estado	Usuario
1	Al iniciar sesión la página redirige a la sección principal	Aprobado	Administrador
2	Al cerrar sesión la página redirige a la sección de inicio de sesión	Aprobado	Administrador
3	Al recuperar la contraseña, el sistema envía una nueva contraseña válida por el correo electrónico	Aprobado	Administrador
4	Visualización de los usuarios junto con su información personal, así como sus roles y su estado	Aprobado	Administrador
5	Comprobar las funciones de crear, editar o inhabilitar usuarios	Aprobado	Administrador
6	Comprobar la función de actualizar las coordenadas así como el radio de aceptación de la geolocalización	Aprobado	Administrador
7	Comprobar la función de actualizar el usuario representante de una residencia	Aprobado	Administrador

Tabla 22. Pruebas de aceptación - Segunda iteración

N.	Criterio	Estado	Usuario
1	Visualización de los diferentes parqueaderos agrupados por colores según su tipo en un mapa que represente las calles del conjunto habitacional	Aprobado	Presidente
2	Comprobar la funcionalidad de actualización de residencia asociada con cada parqueadero	Aprobado	Presidente
3	Visualización de todas las residencias del conjunto habitacional	Aprobado	Presidente

4	Comprobar la función de actualizar al usuario representante de cada residencia	Aprobado	Presidente
5	Comprobar las funciones de crear, editar o inhabilitar de guardias	Aprobado	Vicepresidente
6	Comprobar las funciones de crear, editar o eliminar las actividades de guardianía	Aprobado	Vicepresidente
7	Comprobar las funciones de crear, editar o eliminar los incidentes de guardianía	Aprobado	Presidente
8	Al crear o editar una incidencia las imágenes se sube correctamente al sistema	Aprobado	Presidente
9	Comprobar las funciones de crear, editar o eliminar las convocatorias	Aprobado	Presidente
10	En las asambleas se visualiza todo el listado de casas junto con sus representantes en el caso de tener	Aprobado	Presidente
11	Comprobar la función de actualizar el estado de asistencia de una casa en la asamblea	Aprobado	Presidente
12	Comprobar la función de registrar la asistencia de una casa desde la aplicación móvil	Aprobado	Presidente
13	En las asambleas comprobar las funciones de crear, editar, eliminar las preguntas de las votaciones	Aprobado	Presidente
14	Comprobar la habilitación a votación de las preguntas	Aprobado	Presidente
15	Comprobar el registro del voto por parte de los asistentes a través de la aplicación móvil	Aprobado	Presidente

Tabla 23. Pruebas de aceptación - Tercera iteración

N.	Criterio	Estado	Usuario
1	Visualización de los ingresos mensuales como casuales junto con los meses pagados, fechas de pago, número de casa y comprobante de pago	Aprobado	Tesorera
2	Comprobar las funcionalidades de crear, editar o eliminar ingresos mensuales y casuales	Aprobado	Tesorera
3	Visualización de las multas junto con el monto a pagar, el estado del pago, número de casa, evidencias y comprobante de pago	Aprobado	Tesorera
4	Comprobar las funciones de crear, editar o eliminar multas	Aprobado	Tesorera
5	Al crear o editar una multa el sistema guarda correctamente comprobantes de pago y evidencias	Aprobado	Tesorera
6	Comprobar las funciones de crear, editar, eliminar o descarga en PDF los eventos sociales	Aprobado	Secretaria

7	Al crear o editar un evento social con imagen se sube correctamente al sistema	Aprobado	Secretaria
8	Al iniciar sesión en la aplicación móvil se redirige a la vista principal	Aprobado	Propietario/Inquilino
9	Se visualizan los eventos sociales y asambleas próximas	Aprobado	Propietario/Inquilino
10	Se visualiza el estado financiero de las casas asociadas a cada usuario	Aprobado	Propietario/Inquilino
11	Durante la asamblea la asistencia es validad y registrada	Aprobado	Propietario/Inquilino
12	Durante la asamblea las preguntas a votar son visualizadas	Aprobado	Propietario
13	Durante la asamblea se registra y valida el voto a cada pregunta	Aprobado	Propietario
14	Durante la asamblea se visualiza el voto actual de cada pregunta	Aprobado	Propietario

Para poder implantar el sistema en producción se realizo la configuración de una instancia Elastic Compute Cloud (EC2) en AWS en donde se instaló un servidor Amazon-Linux para el despliegue de la aplicación web y un servidor Tomcat para el despliegue de la API. También se requirió de un dominio para configurar los certificados (SSL) y poder habilitar el protocolo HTTPS en la aplicación web y la API.

En primer lugar es necesario crear un red virtual en AWS usando Virtual Private Cloud (VPC) para poder configurar las instancias EC2 y los grupos de seguridad. A continuación se creó una instancia EC2 con Amazon-Linux y mediante la consola de AWS se accedió a la instancia para poder copiar el ejecutable .jar de la API desarrollada en Spring Boot. Una vez creado el build del sistema web se realizó la misma operación de copiar el contenido dentro del servidor EC2. Adicionalmente se se utilizo Route 53 para la configuración del dominio y poder asignar el redireccionamiento del dominio adquirido a la IP pública de la instancia EC2. Finalmente en la instancia de EC2 usando la consola de AWS se instalo Nginx para poder configurar el proxy inverso y redirigir el tráfico del puerto 80 al puerto 8080 en donde se encuentra la API y el tráfico del puerto 443 al puerto 80 en donde se encuentra la aplicación web, y también se configurarón los certificados SSL para habilitar el protocolo HTTPS con certbot.



Figura 142. Administrador de VPC en AWS

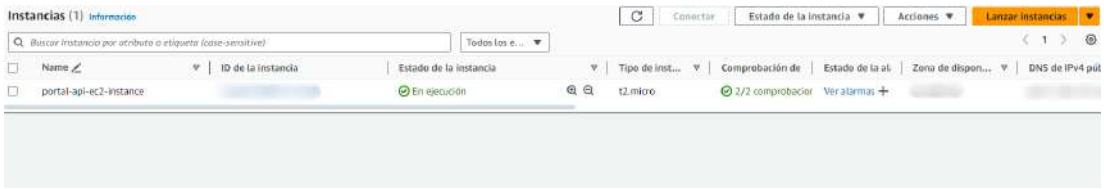


Figura 143. Administrador de EC2 en AWS



Figura 144. Código de copia de recursos y ejecución del API en EC2

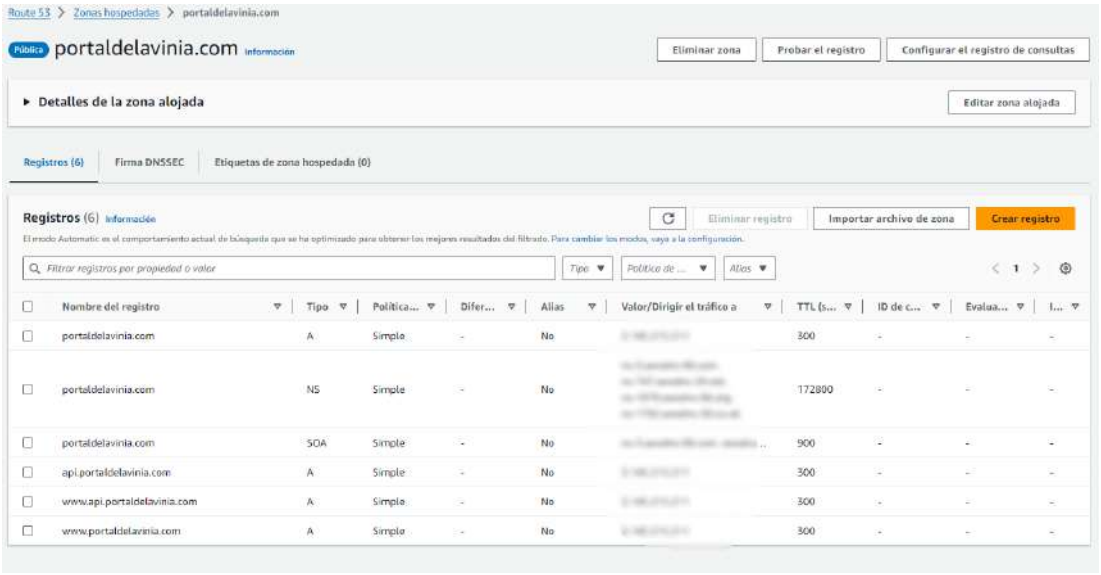


Figura 145. Zonas hospedadas de Route 53 en AWS

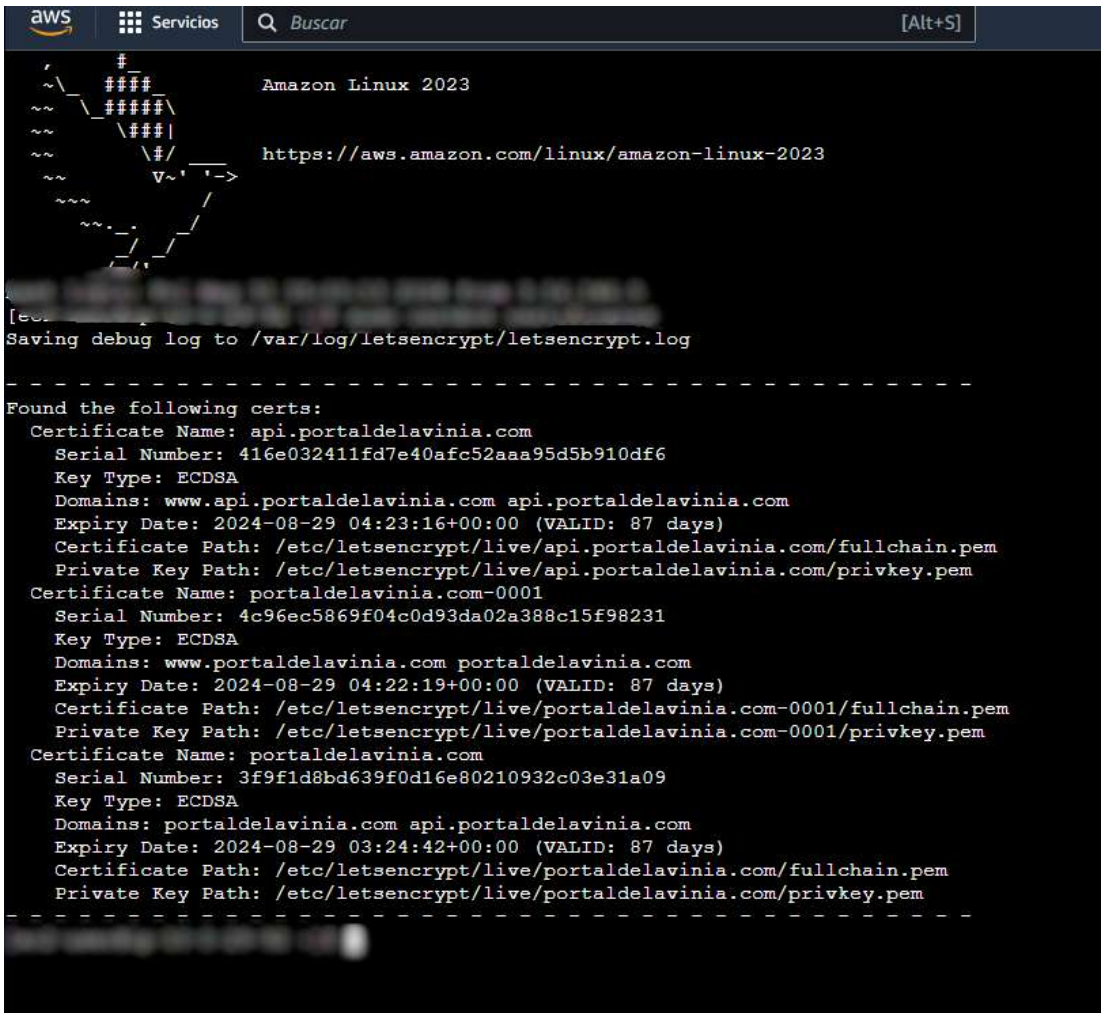


Figura 146. Certificados SSL generados con certbot



Figura 147. Estado del servicio web Nginx en la instancia EC2

```
http {
    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    access_log  /var/log/nginx/access.log  main;

    sendfile            on;
    tcp_nopush          on;
    keepalive_timeout   65;
    types_hash_max_size 4096;

    include             /etc/nginx/mime.types;
    default_type        application/octet-stream;

    include /etc/nginx/conf.d/*.conf;

    server {
        server_name api.portaldelavinia.com www.api.portaldelavinia.com;

        location / {
            proxy_pass http://localhost:8080;
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto $scheme;
        }

    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/api.portaldelavinia.com/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/api.portaldelavinia.com/privkey.pem; # managed by Certbot
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot
```

Figura 148. Archivo de configuración de Nginx en la instancia EC2

3.6 Aplicación de los cuestionarios TAM

Una vez terminado el ciclo de vida de la aplicación con un resultado "exitoso" en las pruebas de aceptación en donde se evaluaron el cumplimiento de los requerimientos y la funcionalidad de la aplicación se procedió a aplicar los cuestionarios de Modelo de Aceptación de Tecnología (TAM) para poder evaluar la percepción de los usuarios sobre la utilidad y facilidad de uso de la aplicación en cada una de las aplicaciones desarrolladas.

3.6.1 Modelo TAM

Según los autores en [32] describen al modelo TAM como un modelo el cual se enfoca en analizar las tecnologías de la información y establecer la relación con los factores que condicionan la actitud del usuario hacia una tecnología nueva y su uso.

a. Carácterísiticas principales de TAM)

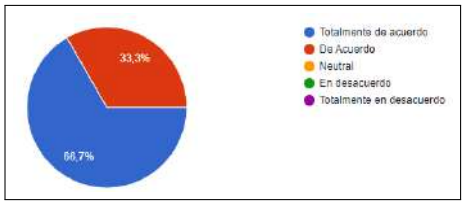
- ***Utilidad percibida (PU).*** En [33] se menciona que la utilidad percibida hace referencia al grado en el que un usuario cree que el uso de un sistema específico mejorará su desempeño en el trabajo.]

- ***Facilidad de uso percibida (PEOU).*** Por otro lado en [32] menciona que la facilidad de uso percibida hace referencia al grado en el que un usuario piensa que el uso de una herramienta tecnológica le llevá a un esfuerzo mínimo.

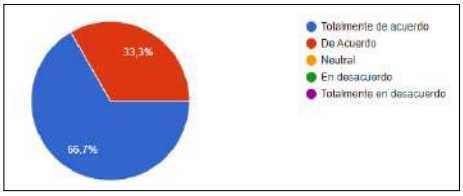
b. Resultados del modelo TAM

En esta sección se presentan los resultados obtenidos de la aplicación de los cuestionarios TAM a los directivos del conjunto habitacional y a los residentes del mismo. Ambos cuestionarios se aplicarón a través de formularios en línea y se encuentran en los Anexos C1 y D1.

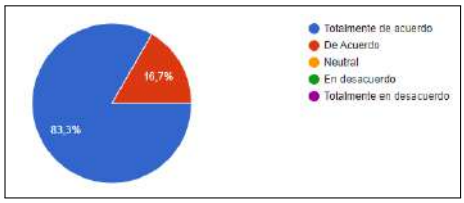
- ***Resultados del cuestionario TAM al sistema web.*** Los resultados recogidos de las encuestas aplicadas a los cuatro miembros de la directiva y dos miembros representantes de los pasajes del conjunto habitacional se muestran en las Figuras 149 y 150.



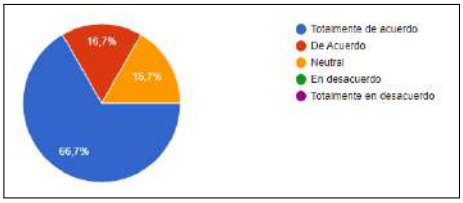
(a) Resultados PU1



(b) Resultados PU2

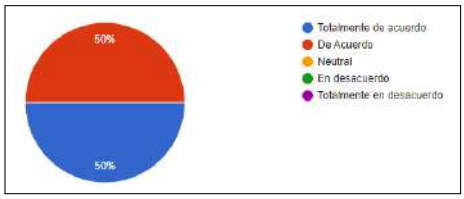


(c) Resultados PU3

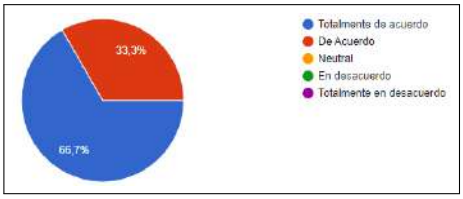


(d) Resultados PU4

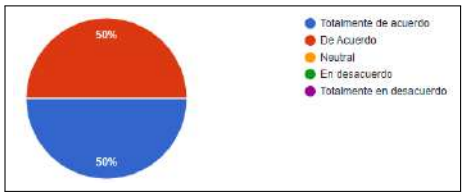
Figura 149. Resultados del PU de la encuesta TAM al sistema web



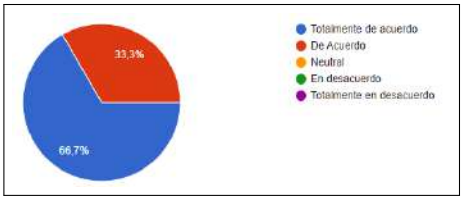
(a) Resultados PEOU1



(b) Resultados PEOU2



(c) Resultados PEOU3



(d) Resultados PEOU4

Figura 150. Resultados del PEOU de la encuesta TAM al sistema web

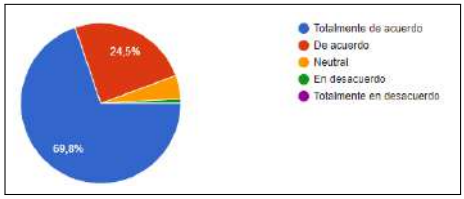
Tabla 24. Resultados PU/PEOU-TAM web

Resultados Utilidad percibida (PU)						
Código	**Totalmente en desacuerdo**	**En desacuerdo**	**Neutral**	**De acuerdo**	**Totalmente de acuerdo**	**Media**
PU1	0	0	0	1.332	3.335	4.667
PU2	0	0	0	1.332	3.335	4.667
PU3	0	0	0	0.668	4.165	4.833
PU4	0	0	0.501	0.668	3.335	4.504
Media Total (PU)						4.6678
Resultados Facilidad de uso percibida (PEOU)						
PEOU1	0	0	0	2	2.5	4.5
PEOU2	0	0	0	1.332	3.335	4.667

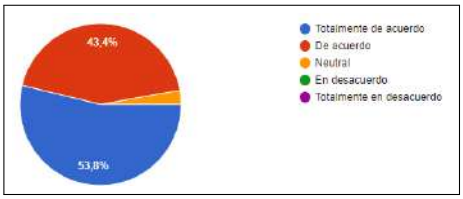
PEOU3	0	0	0	2	2.5	4.5
PEOU4	0	0	0	1.332	3.335	4.667
Media Total (PEOU)						4.5835

Basándonos en los datos recabados dela encuesta TAM del sistema web, donde la utilidad percibida promedio alcanzó 4.6678 y la facilidad de uso percibida fue de 4.5835, es claro que los usuarios de la directiva del conjunto habitacional valoran altamente tanto la utilidad como la facilidad de uso de la aplicación web.

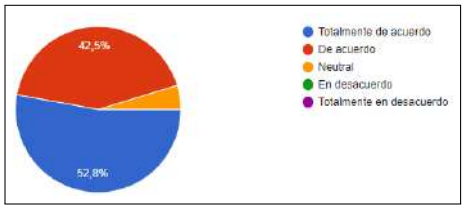
- ***Resultados del cuestionario TAM a la aplicación móvil.*** Los resultados recogidos de las encuestas aplicadas a ciento seis residentes tanto propietarios como inquilinos del conjunto habitacional se muestran en las Figuras 151 y 152.



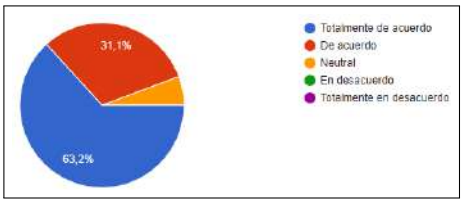
(a) Resultados PU1



(b) Resultados PU2

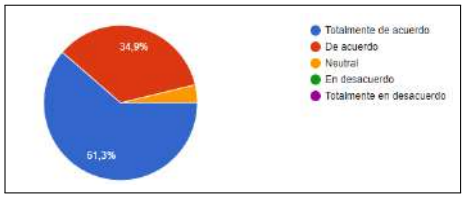


(c) Resultados PU3

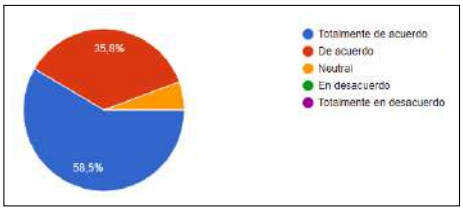


(d) Resultados PU4

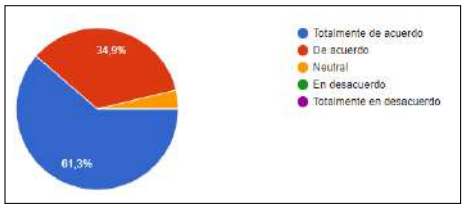
Figura 151. Resultados del PU de la encuesta TAM a la aplicación móvil



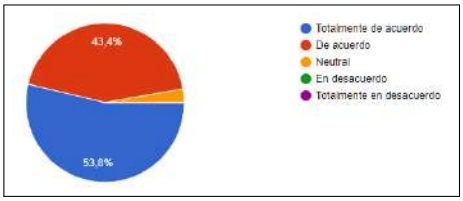
(a) Resultados PEOU1



(b) Resultados PEOU2



(c) Resultados PEOU3



(d) Resultados PEOU4

Figura 152. Resultados del PEOU de la encuesta TAM a la aplicación móvil

Tabla 25. Resultados PU/PEOU-TAM móvil

Resultados Utilidad percibida (PU)						
Código	**Totalmente en desacuerdo**	**En desacuerdo**	**Neutral**	**De acuerdo**	**Totalmente de acuerdo**	**Media**
PU1	0	0.018	0.141	0.98	3.49	4.629
PU2	0	0	0.084	1.736	2.69	4.51
PU3	0	0	0.141	1.7	2.64	4.481
PU4	0	0	0.171	1.244	3.065	4.48
Media Total (PU)						4.525

Resultados Facilidad de uso percibida (PEOU)						
PEOU1	0	0	0.114	1.396	3.065	4.575
PEOU2	0	0	0.171	1.432	2.926	4.529
PEOU3	0	0	0.114	1.396	3.065	4.575
PEOU4	0	0	0.084	1.736	2.69	4.51
Media Total (PEOU)						4.54725

Con los resultados recopilados de la encuesta TAM sobre la aplicación móvil, donde la utilidad percibida promedio fue de 4.525 y la facilidad de uso percibida fue de 4.54725, es evidente que los residentes del conjunto habitacional valoran altamente la utilidad y la facilidad de uso de la aplicación móvil. Estos datos sugieren claramente que la aplicación es considerada como una herramienta valiosa y fácil de utilizar por parte de los usuarios del complejo residencial.

Por lo tanto se concluye que el proyecto de investigación ha sido un éxito en términos de la percepción de los usuarios sobre la utilidad y facilidad de uso de la aplicación web y móvil desarrollada para la gestión de un conjunto habitacional.

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

El presente trabajo de titulación se desarrolló con el objetivo de proponer una solución tecnológica que permita mejorar la gestión de la administración del Conjunto habitacional "El Portal de la Viña", a través de la implementación de un sistema web y móvil que permita a la directiva y a los residentes gestionar de una manera más ágil y cómoda los procesos administrativos, así como proporcionar un acceso más rápido a la información del conjunto habitacional.

- Con la ayuda de las entrevistas y encuestas se pudo identificar los problemas principales que enfrentaba la directiva del conjunto habitacional, con lo cual se logró proponer una solución tecnológica que permita mejorar la gestión y el acceso a la información. La entrevista con el presidente de la directiva fue en mayor parte la que permitió identificar los problemas administrativos, tales como la deficiencia en la toma de asistencia en las asambleas, la dificultad de acceder a la información de las obligaciones financieras de los residentes y la mala organización de la información de los parqueaderos. Con estos problemas analizados se pudo evidenciar la necesidad de tener un sistema que permita gestionar de manera eficiente la información del conjunto habitacional.

- El uso de frameworks de desarrollo ha demostrado que son herramientas que permiten acelerar el proceso de desarrollo de software, ya que proveen de un marco de trabajo ya definido con el cual se puede agilizar de forma considerable el desarrollo de cualquier software, además de que existe una gran cantidad de documentación y una comunidad activa que puede ayudar a resolver problemas que se presenten durante el desarrollo. Sin embargo, la elección del framework adecuado dependerá de las necesidades del proyecto y de la experiencia de las personas involucradas en el desarrollo.

- La metodología de desarrollo de software debe ser seleccionada de acuerdo a la naturaleza del proyecto, ya que depende directamente de los requerimientos y del equipo de trabajo. En el caso partícular de este proyecto se pudo observar que el uso de una metodología ágil como RAD permitió tener un desarrollo rápido

y con la participación activa de los usuarios, lo cual permitió tener un producto final que cumplió con las necesidades y expectativas del cliente.

- La implementación de las validaciones mediante el uso de la geolocalización junto con el cálculo de la distancia de Haversiné demostraron ser unas herramientas útiles para verificar la ubicación de los residentes durante las asambleas. Esta estrategia ha tenido un impacto positivo en la confiabilidad del proceso de asistencias y votaciones, previniendo el mal uso de la aplicación y garantizando la autenticidad de los datos durante las sesiones de asamblea.
- El uso de encuestas TAM permitieron evaluar la percepción de los usuarios sobre la aplicación móvil y web, lo cual permitió identificar los aspectos positivos y negativos de la aplicación.

4.2 Recomendaciones

- Se recomienda que la directiva del conjunto habitacional “El Portal de la Viña” realice una capacitación a los residentes sobre el uso de la aplicación móvil y web, con el fin de que puedan aprovechar al máximo las funcionalidades que ofrece el sistema.
- Se sugiere llevar a cabo una concientización a los residentes por parte de la directiva, para que se utilicen las herramientas tecnológicas de manera responsable, ya que el uso innecesario puede conllevar a un incremento de costos en los servicios implementados de AWS.
- Dada la importancía de la información que se maneja en el sistema, se recomienda realizar respaldos de manera periódica de la base de datos, con el fin de evitar la pérdida de información en caso de que ocurra algún fallo en el sistema.
- En consideración de la información que se maneja en el sistema, se sugiere una integración con el sistema actual de control de tarjetas de acceso, con el fin de tener centralizada toda la información y evitar el uso de múltiples sistemas.

- AWS provee una gran cantidad de servicios por lo cual se recomienda tener un conocimiento mínimo a cerca de esta plataforma, ya que el uso de servicios innecesarios puede conllevar a una mayor inversion de recursos y, por lo tanto, a un incremento en los costos.

REFERENCIAS BIBLIOGRÁFICAS

[1] Carla, Moran, "DESARROLLO DE UNA APLICACIÓN WEB DE GESTIÓN ADMINISTRATIVA PARA LA COOPERATIVA DE TRANSPORTE "EXPRESO MILAGRO" EN EL CANTÓN MILAGRO," *UNIVERSIDAD ESTATAL DE MILAGRO*, 2020.

[2] INEC. (2022) Censo de Población y 2022. [En línea]. Disponible en: https://censoecuador.ecudatanalytics.com/

[3] C. Nacional, "LEY DE PROPIEDAD HORIZONTAL, Art.11," 2013.

[4] F. Fajardo, Azucena, "MODELO DE GESTIÓN ADMINISTRATIVO - FINANCIERA, PARA EL CONJUNTO HABITACIONAL EL PORTAL DE LA VIÑA DE LA CIUDAD DE AMBATO, PARA OPTIMIZAR LOS RECURSOS Y TOMA DE DECISIONES," *ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO*, 2013.

[5] H. Ortega y A. Tamayo, "Análisis, diseño e implementación de una aplicación web para la administración de un condominio, además de una aplicación móvil para el envío de notificaciones," Trabajo de Titulación, Universidad Politecnica Salesiana, Jul. 2017.

[6] S. Pinto y A. Fuentes, "Sistema de gestión administrativo para Condominio Puertas de Alcalá," Trabajo de Titulación, Universidad del Bío-Bío, 2017.

[7] J. Nieto y L. Prada, "Prototipo de Aplicación Web para la Sistematización del Proceso de Asamblea General de Copropietarios en Propiedad Horizontal Mediante Servicios SOAP," Trabajo de Titulación, Universidad Distrital Francisco Jose de Caldas, May 2018.

[8] M. Moscayza y T. Alejandro, "Sistema web para la gestión de presupuestos en el Edificio Condominio Aquamar S.A.C., 2018."

[9] J. Leonardo, "Mejora del control de asistencia de personal a través de un sistema de información con reconocimiento facial geolocalizado en Agro Rural," Trabajo de Titulación, Universidad Tecnologica del Peru, Peru, 2019.

[10] C. Moreira, “DESARROLLO DE UN SISTEMA WEB PARA EL FORTALECIMIENTO DE LOS PROCESOS DE GESTIÓN ADMINISTRATIVA Y FINANCIERA DEL CONDOMINIO SOLAR DEL RÍO DE LA CIUDAD DE IBARRA UTILIZANDO MICROSOFT AZURE CORONADO MOREIRA CRISTOPHER GEOVANNY 2019-04-09 PREGRADO INGENIERO EN SISTEMAS COMPUTACIONALES MSC. DIEGO TREJO,” Trabajo de Titulación, Universidad Tecnica del Norte, Abr. 2019.

[11] G. López y H. Sanchez, “PROTOTIPO DE SOFTWARE WEB PARA LA REALIZACIÓN DE VOTACIONES,” Trabajo de Titulación, Universidad Distrital Francisco Jose de Caldas, Nov. 2020.

[12] A. Ortega, “Prototipo de un sistema Web para la gestión de actividades e inconvenientes por medio de herramientas Open Source para el Condominio los Jardines.” Trabajo de Titulación, Universidad de Guayaquil, Oct. 2020.

[13] Y. Portugal, “Sistema de Información para la Administración de los Servicios que se Presta a los Propietarios del Condominio Jardines de Aramburu 2, Lima, 2021,” Trabajo de Titulación, Universidad Privada Telesup, Peru, 2022.

[14] E. Castro y O. Eche, “Reconocimiento facial y geolocalización para el control de asistencias en una empresa de textiles - Piura 2023,” Trabajo de Titulación, Universidad Cesar Vallejo, Lima - Peru, 2023.

[15] J. Iza y W. Rojas, “Desarrollo de un ERP para la empresa Siscom módulo: control de asistencia.” Trabajo de Titulación, Universidad Técnica de Cotopaxi, Latacunga, Ago. 2023.

[16] A. Montalvo y D. Paredes, “Desarrollo de un sistema software multiplataforma basado en un modelo de gestión administrativa enmarcado en la arquitectura REST y REDUX para la optimización de la administración del conjunto habitacional ”Oriental”.” Trabajo de Titulación, Universidad de las Fuerzas Armadas, Latacunga, Ene. 2023.

[17] L. Bravo, L. Orjuela, y E. Bonilla, “SISTEMA DE INFORMACION WEB PARA PROPIEDAD HORIZONTAL “SIPHO”,” Trabajo de Titulación, Corporación Universitaria Minuto de Dios, 2015.

[18] D. Puello, Plinio y Scholoborg, Francisco, “Modelo de aceptación tecnológica (tam) en el laboratorio de física iii basado en internet de las cosas en el programa de ingeniería de sistemas de la universidad de cartagena, colombia,” *Universidad de Cartagena*, 2020.

[19] R. Martín y G. Ollé, “Agilizando los cambios de ui-ux sobre el ambiente productivo mediante figma,” *Universidad Nacional de la Plata*, 2020.

[20] L. Wu, “Desarrollo de una aplicación para el tracking de animales domésticos,” vol. 18, no. 1, p. ix. [En línea]. Disponible en: http://polipapers.upv.es/index.php/IA/article/view/3293

[21] M. Ahmad, “ANALYSIS OF CROSS PLATFORM MOBILE APPLICATION DEVELOPMENT FRAMEWORKS.” [En línea]. Disponible en: https://www.researchgate.net/profile/Marghoob-Ahmad/publication/372679769_ANALYSIS_OF_CROSS_PLATFORM_MOBILE_APPLICATION_DEVELOPMENT_FRAMEWORKS/links/64c2a96bcda2775c03c9e1d1/ANALYSIS-OF-CROSS-PLATFORM-MOBILE-APPLICATION-DEVELOPMENT-FRAMEWORKS.pdf

[22] F. Zhang, G. Sun, B. Zheng, y L. Dong, “Design and implementation of energy management system based on spring boot framework,” vol. 12, no. 11, p. 457. [En línea]. Disponible en: https://www.mdpi.com/2078-2489/12/11/457

[23] D. Choma, K. Chwaleba, y M. Dzieńkowski, “THE EFFICIENCY AND RELIABILITY OF BACKEND TECHNOLOGIES: EXPRESS, DJANGO, AND SPRING BOOT,” vol. 13, no. 4, pp. 73–78. [En línea]. Disponible en: https://ph.pollub.pl/index.php/iapgos/article/view/4279

[24] I. Framework. Geolocation. [En línea]. Disponible en: https://ionicframework.com/docs/native/geolocation

[25] Google. Api de maps javascript. [En línea]. Disponible en: https://developers.google.com/maps/documentation/javascript/reference/polygon?hl=es-419

[26] K. Lasluisa, M. A. G. Saltos, P. F. B. Egas, y R. M. Toasa, “Evaluación del desempeño en tiempos de respuesta para bases de datos SQL, NoSQL y NewSQL.”

[27] J. J. León Soberón, “ANÁLISIS COMPARATIVO DE SISTEMAS GESTORES DE BASES DE DATOS POSTGRESQL y MYSQL EN PROCESOS CRUD,” thesis, Universidad Señor de Sipán. [En línea]. Disponible en: https://repositorio.uss.edu.pe/bitstream/handle/20.500.12802/7012/Le%c3%b3n%20Sober%c3%b3n%20Jenner%20Jes%c3%bas.pdf?sequence=1&isAllowed=y

[28] A. Z. Islam y D. A. Ferworn, “A comparison between agile and traditional software development methodologies,” *Global Journal of Computer Science and Technology*, pp. 7–42. [En línea]. Disponible en: https://computerresearch.org/index.php/computer/article/view/1987

[29] L. J. Barriga Sánchez, “SISTEMA DE GESTIÓN DE HISTORIAS CLÍNICAS y TELEMEDICINA ORIENTADO AL DEPARTAMENTO MÉDICO GAD MUNICIPALIDAD DE AMBATO,” thesis, Universidad Técnica de Ambato. [En línea]. Disponible en: https://repositorio.uta.edu.ec/jspui/bitstream/123456789/39584/1/t2401so.pdf

[30] O. Camino Costa, “Desarrollo de una aplicación de realidad aumentada sobre android para el apuntamiento de los nodos en el telescopio de neutrinos antares,” thesis, Universidad Politecnica de Valencia. [En línea]. Disponible en: https://riunet.upv.es/handle/10251/18444

[31] J. A. Bonilla Cadena, “Desarrollo de una plataforma web para recorridos virtuales 360° mediante la metodología RAD. caso grupo inmobiliario horizonte de la ciudad de riobamba,” thesis, Universidad Nacional del Chimborazo.

[32] M. Tapia León, F. Peñaherrera Larenas, y M. Cedillo Fajardo, “Comparación de los LMS moodle y CourseSites de blackboard usando el modelo de aceptación tecnológica TAM,” vol. 8, no. 16, pp. 78–85. [En línea]. Disponible en: https://www.redalyc.org/pdf/5826/582663856010.pdf

[33] L. A. Y. Varela, L. A. R. Tovar, y J. Chaparro, “Modelo de aceptación tecnológica (TAM): un estudio de la influencia de la cultura nacional y del perfil del usuario en el uso de las TIC,” vol. 20.

ANEXOS

Anexo A. Guía de entrevista a la directiva del conjunto habitacional “El Portal de la Viña”

Tabla A1. Guía de entrevista a la directiva del conjunto habitacional “El Portal de la Viña”

N.	Pregunta	Respuesta	Observación
1	¿Cuáles son sus funciones administrativas o financieras como parte de la directiva?		
2	¿Cómo miembro de la directiva tiene alguna función extra que la realice de forma extraordinaria como parte de su iniciativa?		
3	¿Qué reportes, documentos o certificados entrega como parte de directiva y con qué información?		
4	¿Cuál es el proceso actual para la adquisición de un parqueadero tanto particular como de zona azul y en el caso de requerir alguna información o documento, existe la posibilidad de poder entregar alguno de manera digital?		
5	¿De que manera se lleva el registro de los parqueaderos?		
6	¿Cuáles es el proceso actual para la adquisición de servicios de proveedores (guardianía, internet, jardinería, etc.)?		
7	¿De qué manera se lleva la contabilidad del conjunto habitacional y que reportes se suelen presentar y con qué información?		
8	¿De que manera se comunica a los residentes del condominio eventos sociales o convocatorias de asambleas?		
9	¿Cuál es el proceso actual para el registro de asistencia y las votaciones en las asambleas?		
10	¿Cuándo una residente falta a una reunión de asamblea que penalizaciones o multas recibe?		
11	¿Qué problemas o inconvenientes han tenido con los procesos administrativos o financieros actuales?		
12	¿Actualmente cómo funciona el buzón de quejas/sugerencias?		
13	¿Dónde manejan la información de los pagos (multas, alícuotas, parqueaderos, etc.), asambleas, eventos sociales, pagos a proveedores y demás información importante?		

Anexo B. Encuesta de necesidades a los residentes del conjunto habitacional “El Portal de la Viña”

Tabla B1. Encuesta de necesidades a los residentes del conjunto habitacional “El Portal de la Viña”

Pregunta	**Respuesta**
¿Cree usted que el proceso actual para obtener la información de las obligaciones financieras (pagos, multas, etc.) de su domicilio es eficiente?	▪ Totalmente de acuerdo ▪ De acuerdo ▪ Ni de acuerdo ni en desacuerdo ▪ En desacuerdo ▪ Totalmente en desacuerdo
¿Con que frecuencia solicita la información de las obligaciones financieras (pagos, multas, etc.) de su domicilio a tesorería?	▪ Por lo menos una vez al mes ▪ Dos veces al mes ▪ Tres veces al mes ▪ Cuatro veces al mes ▪ Más de cuatro veces al mes
¿Cree usted que el proceso actual para el registro de asistencias durante la asamblea es eficiente en términos de tiempo?	▪ Totalmente de acuerdo ▪ De acuerdo ▪ Ni de acuerdo ni en desacuerdo ▪ En desacuerdo ▪ Totalmente en desacuerdo
¿Considera usted que el conteo actual de las votaciones durante las asambleas es transparente y eficiente?	▪ Totalmente de acuerdo ▪ De acuerdo ▪ Ni de acuerdo ni en desacuerdo ▪ En desacuerdo ▪ Totalmente en desacuerdo
¿Ha experimentado retrasos en la entrega de los comprobantes de pagos?	▪ Si ▪ No

¿Con que frecuencia olvida que el pago de alícuotas de su domicilio esta por vencer?	▪ Muy frecuentemente ▪ Frecuentemente ▪ Ocasionalmente ▪ Raramente ▪ Nunca
¿Con que frecuencia ha utilizado el buzón de quejas/sugerencias?	▪ Nunca lo he usado ▪ Al menos una vez al mes ▪ Dos veces al mes ▪ Tres veces al mes ▪ Más de tres veces al mes
¿Considera usted que los medios actuales (whatsapp/pizarrón en la garita) que utiliza la directiva para publicar comunicados son eficientes?	▪ Totalmente de acuerdo ▪ De acuerdo ▪ Ni de acuerdo ni en desacuerdo ▪ En desacuerdo ▪ Totalmente en desacuerdo
¿Preferiría utilizar una aplicación móvil que le permita revisar la información de las obligaciones financieras de su domicilio, el registro de su asistencia y voto durante las asambleas, se le envíen notificaciones recordándole los pagos por vencer o que ha sido multado por alguna infracción?	▪ Si ▪ No

Anexo C. Encuesta TAM del sistema web administrativo

Tabla C1. Encuesta de aceptación TAM #1

ENCUESTA DE ACEPTACIÓN TAM SISTEMA WEB				
En una escala de Likert indique su grado de aceptación con las siguientes ponderaciones:				
1	2	3	4	5
Totalmente en desacuerdo	En desacuerdo	Neutral	De acuerdo	Totalmente de acuerdo
PU	**Utilidad Percibida**			
PU1	El sistema facilita las tareas de gestión del conjunto residencial			
PU2	Mejora la eficiencia en las operaciones diarias del conjunto residencial			
PU3	Proporciona información relevante para la toma de decisiones			
PU4	Mejora la calidad de vida/comodidad para los residentes			
PEOU	**Facilidad de Uso Percibida**			
PEOU1	Es fácil aprender a utilizar el sistema de gestión residencial			
PEOU2	La navegación por el sistema es clara y lógica			
PEOU3	El diseño de la interfaz es intuitivo y amigable			
PEOU4	Es cómodo interactuar desde diferentes navegadores web			

Anexo D. Encuesta TAM de la aplicación móvil

Tabla D1. Encuesta de aceptación TAM #2

ENCUESTA DE ACEPTACIÓN TAM APLICACIÓN MÓVIL				
En una escala de Likert indique su grado de aceptación con las siguientes ponderaciones:				
1	2	3	4	5
Totalmente en desacuerdo	En desacuerdo	Neutral	De acuerdo	Totalmente de acuerdo
PU	**Utilidad Percibida**			
PU1	La aplicación ofrece información relevante sobre próximos eventos en el conjunto residencial			
PU2	Las notificaciones de pagos son claras y fáciles de comprender en la aplicación.			
PU3	La opción de votaciones dentro de la aplicación es útil y accesible.			
PU4	La aplicación mejora la interacción y la comunicación entre los residentes.			
PEOU	**Facilidad de Uso Percibida**			
PEOU1	Es sencillo navegar por la aplicación para encontrar la información deseada.			
PEOU2	El diseño de la aplicación es intuitivo y fácil de comprender.			
PEOU3	La aplicación ofrece una experiencia cómoda al interactuar desde distintos dispositivos móviles.			
PEOU4	La función de votación en la aplicación es fácil de usar y comprender.			

Anexo E. Dependencias de la API



```
<dependencies> Edit Starters...
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-validation</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <dependency>
        <groupId>org.postgresql</groupId>
        <artifactId>postgresql</artifactId>
        <scope>runtime</scope>
    </dependency>
    <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <optional>true</optional>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-security</artifactId>
    </dependency>
    <dependency>
        <groupId>io.jsonwebtoken</groupId>
        <artifactId>jjwt-api</artifactId>
        <version>0.12.5</version>
    </dependency>
    <dependency>
        <groupId>io.jsonwebtoken</groupId>
        <artifactId>jjwt-impl</artifactId>
        <version>0.12.5</version>
        <scope>runtime</scope>
    </dependency>
```


<dependency>
<groupId>io.jsonwebtoken</groupId>
<artifactId>jjwt-jackson</artifactId>
<version>0.12.5</version>
<scope>runtime</scope>
</dependency>
<dependency>
<groupId>org.springframework.security</groupId>
<artifactId>spring-security-test</artifactId>
<scope>test</scope>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-mail</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-freemarker</artifactId>
</dependency>
<!-- https://mvnrepository.com/artifact/com.amazonaws/aws-java-sdk-s3 →
<dependency>
<groupId>com.amazonaws</groupId>
<artifactId>aws-java-sdk-s3</artifactId>
<version>1.12.682</version>
</dependency>
<!-- https://mvnrepository.com/artifact/commons-io/commons-io →
<dependency>
<groupId>commons-io</groupId>
<artifactId>commons-io</artifactId>
<version>2.15.1</version>
</dependency>
<!-- https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api →
<dependency>
<groupId>javax.xml.bind</groupId>
<artifactId>jaxb-api</artifactId>
<version>2.3.1</version>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-websocket</artifactId>
</dependency>
<!-- https://mvnrepository.com/artifact/com.google.firebase/firebase-admin →
<dependency>
<groupId>com.google.firebase</groupId>
<artifactId>firebase-admin</artifactId>
<version>9.2.0</version>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
</dependencies>