**Proyecto de Título**

**Capstone**

**Sección 003-D**

**Caso Bomberos de Talcahuano**

**Profesor:**

Patricio Soto

***Integrantes:***

Benjamín Alvarez

Daniel Gonzales

***Carrera:***

Ingeniería en Informática

***Sede:***

Duoc UC, Sede San Joaquín

índice

[Introducción 3](#_Toc208682619)

[Fundamentación 4](#_Toc208682620)

[Problema identificado 4](#_Toc208682621)

[Solución propuesta 4](#_Toc208682622)

[Beneficiarios 5](#_Toc208682623)

[Alcance 6](#_Toc208682624)

[Objetivos 7](#_Toc208682625)

[Objetivo general 7](#_Toc208682626)

[Objetivos específicos 7](#_Toc208682627)

[Metodología 8](#_Toc208682628)

[Roles definidos 8](#_Toc208682629)

[Artefactos principales 9](#_Toc208682630)

[Estructura de iteraciones 9](#_Toc208682631)

[Requerimientos 10](#_Toc208682632)

[Requerimientos Funcionales: 10](#_Toc208682633)

[Requerimientos no funcionales: 11](#_Toc208682634)

[Diagramas (DAS) 12](#_Toc208682635)

[Diagrama de casos de uso: 12](#_Toc208682636)

[Diagrama Entidad-Relación (DER) 13](#_Toc208682637)

[Diagrama de Clases: 14](#_Toc208682638)

[Tecnologías 15](#_Toc208682639)

[Costos del Proyecto 17](#_Toc208682640)

[Riesgos 18](#_Toc208682641)

[Pruebas 18](#_Toc208682642)

[Backlog del Sistema 19](#_Toc208682643)

[**1. Requerimientos Funcionales (RF)** 19](#_Toc208682644)

[**2. Requerimientos No Funcionales (RNF)** 20](#_Toc208682645)

[Plan de Trabajo 22](#_Toc208682646)

[Carta Gantt 23](#_Toc208682647)

# Introducción

El adecuado funcionamiento de los vehículos de emergencia es crítico para garantizar la respuesta oportuna del Cuerpo de Bomberos de Talcahuano ante situaciones que ponen en riesgo la vida y la seguridad de la comunidad. No obstante, la gestión de mantenciones preventivas y correctivas no siempre se realiza de forma sistemática, lo que provoca retrasos operativos, incremento de costos y, en ocasiones, la indisponibilidad de unidades en momentos clave. La falta de un sistema web centralizado que organice y consolide la información de la flota dificulta la toma de decisiones oportunas y eficientes.

Para abordar esta necesidad, el proyecto propone el diseño e implementación de un sistema web para el control y seguimiento de mantenciones de vehículos de emergencia. La solución permitirá registrar datos técnicos, programar y monitorear mantenciones, generar alertas automáticas sobre tareas pendientes y emitir reportes confiables. Con ello, se optimiza la gestión interna y se mejora directamente la calidad del servicio prestado a la comunidad.

Este informe corresponde a la planificación inicial que guiará el desarrollo. Presenta la fundamentación, el alcance, los objetivos generales y específicos, la metodología seleccionada, los requerimientos identificados (funcionales y no funcionales), los diagramas preliminares (casos de uso, entidad–relación y clases), la definición tecnológica, una estimación de costos, el plan de trabajo con su diagrama de Gantt y los criterios de pruebas a aplicar. Estos elementos constituyen la hoja de ruta para las siguientes etapas del proyecto.

# Fundamentación

### Problema identificado

El Cuerpo de Bomberos de Talcahuano enfrenta dificultades significativas en el control y la gestión de las mantenciones de su flota de vehículos de emergencia. La falta de un sistema web centralizado para registrar, organizar y monitorear la información técnica y operativa de cada unidad provoca dispersión de datos y pérdida de trazabilidad respecto de intervenciones pasadas, fallas recurrentes, fechas programadas y costos asociados

Como consecuencia, se producen impactos operativos y de seguridad:

* Retrasos en la ejecución de mantenciones preventivas y correctivas, con riesgo de indisponibilidad de unidades críticas en momentos clave.
* Aumento de costos por intervenciones tardías, duplicidades y compras no planificadas de insumos.
* Toma de decisiones con información incompleta o desactualizada, al carecer de un historial consolidado e indicadores oportunos.
* Mayor exposición al riesgo para el personal y la comunidad, debido a menor disponibilidad operativa y fiabilidad de los vehículos ante emergencias.

La situación descrita se refiere a la gestión de mantenciones de la flota vehicular, abarcando el registro histórico de intervenciones, la programación de tareas, el control de alertas y la visibilidad de costos por vehículo, todo ello bajo un único sistema web.

### Solución propuesta

Para resolver las brechas detectadas en la gestión de la flota, se propone el diseño e implementación de un sistema web que centralice la información de los vehículos de emergencia del Cuerpo de Bomberos de Talcahuano y permita controlar de forma trazable las mantenciones preventivas y correctivas.

Arquitectura y tecnologías.

El sistema se implementará con Angular 17 (SPA) y SCSS en el frontend, integrado a Firebase en el backend: Autenticación (control de acceso por roles con Custom Claims), Cloud Firestore (almacenamiento de datos e índices), Cloud Functions con Node 20 (reglas de negocio, generación de alertas, procesos programados), Cloud Storage (adjuntos) y FCM Web Push (notificaciones en navegador). Para avisos por correo se utilizará una función transaccional (p. ej., Nodemailer) disparada desde Functions.

Capacidades clave del sistema:

* Gestión de vehículos: registro, edición y consulta del perfil técnico por unidad (patente única, datos técnicos, estado operativo) y visualización de historial. Regla de negocio: no eliminar vehículos con mantenciones asociadas (RESTRICT o baja lógica).
* Gestión de mantenciones: documentación de mantenciones preventivas y correctivas con fecha, tipo, trabajo realizado, responsable, estado (programada, en curso, realizada, vencida), kilometraje e insumos.
* Alertas y notificaciones: generación automática de alertas por mantenciones próximas o vencidas, con umbrales configurables (tiempo y/o kilometraje). Envío por correo y notificaciones en la web (banner/toast y push de navegador vía FCM).
* Historial y consulta: historial consolidado por vehículo con búsqueda por patente y filtros por fecha, tipo y estado; indicadores básicos para decisiones oportunas.
* Usuarios y roles: autenticación centralizada y autorización por roles (administrador, supervisor/mecánico, conductor) mediante Claims y Reglas de Firestore/Storage (principio de mínimo privilegio).
* Adjuntos: carga y visualización de boletas, facturas y fotos (PDF/JPG/PNG) asociadas a cada mantención, con metadatos y control de descarga.
* Costos: registro de costos por intervención (repuestos y mano de obra) y consolidación por vehículo y período.
* Reportes y exportación: reportes configurables sobre estado de flota, frecuencia/tipo de mantenciones y costos, con exportación a PDF y Excel respetando los filtros aplicados.

Atributos de calidad (síntesis).

Seguridad: Auth + Claims + Reglas; TLS; auditoría de acciones sensibles. Rendimiento: índices compuestos y consultas paginadas.

Disponibilidad: despliegue en Hosting de Firebase y monitoreo básico; ventanas de mantención comunicadas.

Mantenibilidad: código modular, pruebas unitarias/integraciones integradas en cada iteración y documentación técnica.

## Beneficiarios

Beneficiarios directos: Los principales beneficiarios son los integrantes del Cuerpo de Bomberos de Talcahuano involucrados en la administración, supervisión y operación de la flota de vehículos de emergencia: responsables de flota, jefaturas de compañía/guardia, supervisores/mecánicos y conductores. El sistema web les proporciona información centralizada y actualizada sobre el estado de cada unidad, su historial de mantenciones, alertas próximas/vencidas y costos asociados, facilitando la planificación, la coordinación de intervenciones y la toma de decisiones con trazabilidad.

Beneficiarios indirectos: La comunidad de Talcahuano y zonas aledañas se beneficia al mejorar la disponibilidad operativa de los vehículos y, con ello, los tiempos de respuesta ante emergencias. La gestión más eficiente y auditable de la flota contribuye a la seguridad y al bienestar de la población.

# Alcance

El proyecto contempla el diseño e implementación de un sistema web para la gestión de mantenciones de la flota del Cuerpo de Bomberos de Talcahuano, con las siguientes capacidades:

Gestión de vehículos:

* Registro, edición y consulta del perfil técnico (patente única, marca, modelo, año, estado operativo, observaciones).
* Historial por unidad (intervenciones, alertas, costos).
* Regla de negocio: no eliminar vehículos con mantenciones asociadas (baja lógica o RESTRICT).

Gestión de mantenciones (preventivas y correctivas):

* Registro con fecha, tipo de trabajo, responsable, estado (programada, en curso, realizada, vencida), kilometraje, observaciones e insumos.
* Adjuntos por mantención (boletas, facturas, fotos: PDF/JPG/PNG).
* Costos por intervención (repuestos, mano de obra) y consolidado por vehículo/período.

Alertas y notificaciones:

* Generación automática de alertas por mantenciones próximas o vencidas, con umbrales configurables (tiempo y/o kilometraje).
* Notificación por correo electrónico y en la interfaz web (banner/toast y, si aplica, push de navegador).

Historial, búsqueda y filtrado:

* Historial consolidado por vehículo:
* Búsqueda por patente y filtros por rango de fechas, tipo y estado.

Reportes y exportación:

* Reportes configurables sobre estado de flota, frecuencia/tipo de mantenciones y costos.
* Exportación a PDF y Excel, respetando los filtros aplicados.

Usuarios y roles:

* Autenticación y control de acceso por rol (administrador, supervisor/mecánico, conductor).
* Trazabilidad de acciones relevantes (auditoría básica).

Parámetros del sistema:

* Administración de catálogos (tipos/estados de mantención) y umbrales de alertas.

3.2 Exclusiones (fuera de alcance):

* Telemetría en tiempo real, geolocalización, despacho o gestión de incidentes.
* Inventario de bodega completo e integración contable/ERP (solo registro de insumos a nivel de mantención).
* Integraciones externas (GPS, SAP, etc.) distintas de correo/FCM web push.
* Offline completo; el uso requiere conexión a internet.

3.3 Supuestos:

* Existencia de un listado oficial de vehículos y acceso a información histórica mínima para carga inicial.
* Cuentas de correo institucionales disponibles para envío de notificaciones.
* Política de roles y responsables definida por la institución.
* Usuarios con navegadores modernos actualizados y conectividad a internet.

3.4 Restricciones:

* Plataforma web optimizada para escritorio (≥ 1366×768); compatibilidad con las últimas 2 versiones de Chrome/Edge/Firefox.
* Backend sobre Firebase (Authentication, Firestore, Functions, Storage, FCM); consumo bajo políticas del plan con bolsón anual definido.
* Seguridad regida por autenticación y reglas de acceso; resguardo de datos conforme a lineamientos internos de la institución.

# Objetivos

### Objetivo general

Diseñar e implementar un sistema web que permita al Cuerpo de Bomberos de Talcahuano gestionar, controlar y dar seguimiento a las mantenciones de su flota de vehículos de emergencia, centralizando la información operativa y mejorando la disponibilidad de las unidades mediante alertas oportunas, historial trazable y reportes confiables para la toma de decisiones.

### Objetivos específicos

* O1. Gestión de vehículos: habilitar el registro, edición y consulta del perfil técnico por unidad con patente única y visualización del historial asociado.
* O2. Gestión de mantenciones: registrar mantenciones preventivas y correctivas con fecha, tipo, responsable, estado (programada, en curso, realizada, vencida), kilometraje, observaciones e insumos.
* O3. Alertas y notificaciones: implementar alertas automáticas por mantenciones próximas o vencidas, con umbrales configurables (tiempo y/o kilometraje), notificando por correo y en la interfaz web.
* O4. Historial, búsqueda y filtrado: disponer de historial consolidado por vehículo con búsqueda por patente y filtros por rango de fechas, tipo y estado.
* O5. Adjuntos: permitir la carga y visualización de boletas, facturas y fotos (PDF/JPG/PNG) asociadas a cada mantención.
* O6. Costos: registrar costos por intervención (repuestos y mano de obra) y consolidarlos por vehículo y período.
* O7. Reportes y exportación: generar reportes configurables sobre estado de flota, frecuencia/tipo de mantenciones y costos, con exportación a PDF y Excel respetando los filtros.
* O8. Usuarios y roles: asegurar autenticación y control de acceso por rol (administrador, supervisor/mecánico, conductor) mediante reglas de autorización.
* O9. Parámetros del sistema: administrar catálogos (tipos/estados de mantención) y umbrales de alertas.

# Metodología

Para el desarrollo del proyecto se adoptará un Proceso Iterativo–Incremental sin cliente, lo que permite avanzar en el sistema mediante ciclos cortos de construcción y validación interna. Esta metodología se ajusta a un contexto académico, donde no existe un cliente real, pero sí la necesidad de generar entregables funcionales y verificables en plazos definidos.

Cada iteración tendrá una duración fija de 2 semanas, orientada a construir, probar y documentar un conjunto de funcionalidades específicas. Este enfoque asegura entregables constantes y verificables en plazos cortos.

## Roles definidos

* Jefe de Proyecto: planifica iteraciones, controla plazos/costos, gestiona riesgos y calidad, coordina revisiones internas.
* Desarrolladores (Dev): implementa frontend (Angular 17/SCSS) e integra backend (Firebase: Auth/Claims, Firestore/índices, Functions Node 20, Storage, FCM Web Push).
* QA Interno: diseña y ejecuta pruebas dentro de cada iteración (unitarias, reglas de seguridad, caja negra, usabilidad y rendimiento).
* Docente Revisor: valida entregables parciales y da retroalimentación.

## Artefactos principales

* Backlog del sistema: lista priorizada de funcionalidades/tareas con criterios de aceptación y dependencias.
* Plan de Iteración: alcance comprometido, objetivos, criterios de salida, riesgos/impedimentos.
* Registro de Pruebas: evidencias de pruebas unitarias, de integración y de usabilidad realizadas en cada iteración.
* Matriz de Trazabilidad: vincula requerimientos con casos de uso, pruebas y entregables para asegurar consistencia.

## Estructura de iteraciones

* Iteración 0 – planificación: Selección de ítems desde el Backlog; definición de objetivos y criterios de salida de la iteración.
* Iteración 1 – Análisis y diseño: actualización focalizada de DAS/DER/Clases y maquetas de UI afectadas (solo lo impactado).
* Iteración 2 – Implementación: desarrollo del incremento conforme a estándares (lint/formato, convenciones de commits, revisiones de código).
* Iteración 3 - Pruebas: Unitarias (Angular/Node) y reglas de seguridad (Firestore/Storage en emuladores).
* Iteración 4 – Integración y revisión interna: demo del incremento; registro de observaciones del docente revisor.
* Iteración 5 - Cierre: correcciones finales del alcance comprometido, defectos críticos = 0, documentación y Backlog actualizados; planificación de la siguiente iteración.

Con este enfoque, el proyecto garantiza un avance controlado y verificable, integrando pruebas y documentación dentro de cada iteración, lo que asegura la calidad de los entregables y la coherencia con los requerimientos definidos.

# Requerimientos

### Requerimientos Funcionales:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Título** | **Descripción** | **Actor principal** | **Prioridad** | **Criterio de aceptación** |
| **RF1** | **Gestión de vehículos** | **Registrar, consultar y actualizar el perfil técnico de cada unidad (patente, marca, modelo, año, kilometraje, estado operativo y características). Unicidad de patente.** | **Administrador / Supervisor** | **Alta** | **Dado un usuario con permisos, cuando registra un vehículo con campos obligatorios, entonces el sistema lo guarda y lo muestra en el listado con patente única.** |
| **RF2** | **Gestión de mantenciones** | **Registrar mantenciones preventivas y correctivas, con fecha, tipo, trabajo realizado, observaciones, responsable e insumos.** | **Supervisor / Usuario estándar** | **Alta** | **Dado un vehículo, cuando se documenta una mantención con campos mínimos, entonces queda asociada y visible en su historial.** |
| **RF3** | **Alertas automáticas** | **Generar alertas por mantenciones próximas y vencidas, con umbrales configurables; notificar por correo y en la app.** | **Administrador / Supervisor** | **Alta** | **Dado un plan de mantención, cuando se supera el umbral configurado, entonces el sistema registra la alerta y notifica al actor correspondiente.** |
| **RF4** | **Historial por vehículo** | **Mostrar historial consolidado de intervenciones por unidad, con filtros por fechas, tipo y estado.** | **Supervisor / Usuario estándar** | **Alta** | **Dado un vehículo, cuando se accede a su ficha, entonces es posible visualizar y filtrar todas las mantenciones registradas.** |
| **RF5** | **Usuarios y roles** | **Autenticar usuarios y administrar roles (administrador, supervisor, usuario estándar) con permisos diferenciados por módulo/acción.** | **Administrador** | **Alta** | **Dado un nuevo usuario, cuando se le asigna un rol, entonces sus permisos quedan aplicados de inmediato en menús y acciones.** |
| **RF6** | **Adjuntos de mantención** | **Subir y visualizar documentos asociados a una mantención (boletas, facturas, fotos) en formatos comunes (PDF/JPG/PNG).** | **Supervisor / Usuario estándar** | **Media** | **Dado un registro de mantención, cuando se adjunta un archivo válido, entonces queda asociado, descargable y visible en el detalle.** |
| **RF7** | **Reportes y exportación** | **Generar reportes configurables sobre estado de flota, frecuencia/tipo de mantenciones y costos. Exportación a PDF y Excel.** | **Administrador / Supervisor** | **Alta** | **Dado un conjunto de filtros, cuando se genera un reporte, entonces se muestra en pantalla y se habilita la exportación en PDF y Excel.** |
| **RF8** | **Registro de costos** | **Registrar costos por intervención (repuestos, mano de obra) y consolidar totales por vehículo y período.** | **Supervisor / Administrador** | **Media** | **Dado un registro de mantención, cuando se ingresan costos, entonces el sistema actualiza los totales por unidad y permite consultarlos por período.** |
| **RF9** | **Búsqueda y filtrado** | **Buscar por patente y filtrar vehículos/mantenciones por fecha, tipo y estado.** | **Todos los roles autenticados** | **Alta** | **Dada una cadena de búsqueda o filtros, cuando se ejecuta la consulta, entonces la lista muestra únicamente los resultados coincidentes.** |

### Requerimientos no funcionales:

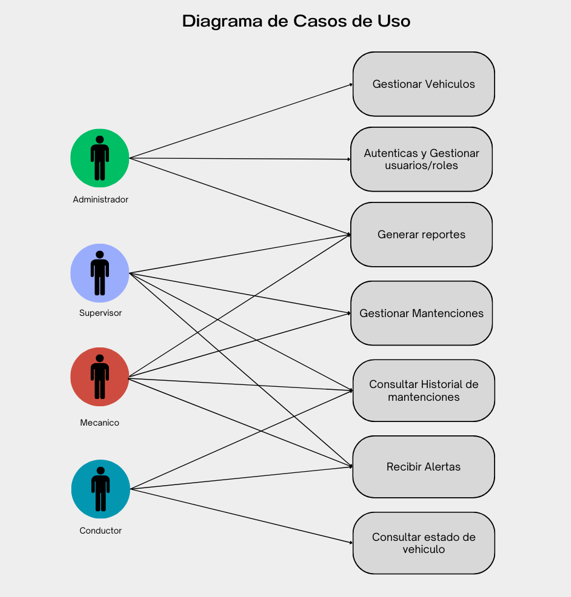
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Categoría | Enunciado | Métrica/Objetivo | Método de verificación | Prioridad |
| RNF1 | Usabilidad | Interfaz sencilla e intuitiva para perfiles no técnicos. | SUS ≥ 75; tasa de error < 2% en tareas críticas; 100% de componentes con (WCAG 2.0). | Pruebas de usabilidad (n≈5), checklist de accesibilidad, SUS. | Alta |
| RNF2 | Disponibilidad | Acceso vía web con conexión a internet. | SLO ≥ 99.0% de disponibilidad mensual (horario operativo); mantenciones preavisadas ≥ 24 h. | Monitoreo de uptime y registro de incidentes. | Alta |
| RNF3 | Seguridad | Acceso autenticado y control de permisos por rol. | TLS ≥ 1.2; contraseñas ≥ 8 con complejidad; bloqueo 5 intentos/15 min; auditoría de acciones clave; Reglas Firestore/Storage (mínimo privilegio). | Revisión de configuración, pruebas por rol, checklist OWASP; tests de reglas (emuladores). | Alta |
| RNF4 | Rendimiento (backend) | Operación estable con carga concurrente. | 50 usuarios concurrentes con p95 API < 1.5 s; degradación < 10% vs. carga base. | Pruebas de carga/estrés (JMeter/Gatling). | Alta |
| RNF5 | Escalabilidad | Crecer en usuarios/módulos sin rediseños mayores. | Duplicar usuarios concurrentes sin cambios arquitectónicos; modularidad validado. | Carga incremental; revisión de arquitectura. | Media |
| RNF6 | Compatibilidad | Funcionamiento en navegadores de escritorio modernos. | Últimas 2 versiones de Chrome/Edge/Firefox (desktop); resolución mínima 1366×768. | Matriz de pruebas cross-browser. | Alta |
| RNF7 | Mantenibilidad | Código legible, modular y documentado. | Cobertura unit tests ≥ 60% en módulos críticos; linter sin errores; documentación de API/README al día. | Reportes de cobertura, linter, revisión de docs. | Media |
| RNF8 | Respaldo y recuperación | Copia de seguridad y restauración de datos. | Backup diario, retención 30 días; RPO ≤ 24 h; RTO ≤ 4 h; prueba de restauración mensual. | Evidencias de jobs y simulacros de restauración. | Alta |
| RNF9 | Portabilidad/Despliegue | Despliegue reproducible en distintos entornos. | Guía de despliegue para Firebase Hosting y un entorno alternativo (hosting básico o cloud); variables de entorno parametrizadas. | Despliegue de prueba en ambos entornos. | Media |
| RNF10 | Rendimiento (frontend) | Respuesta ágil de la interfaz web. | LCP < 4 s y TTI < 5 s (throttling estándar en desktop); p95 de navegación < 3 s en vistas principales. | Auditoría Lighthouse y mediciones en equipo real. | Alta |



# Diagramas (DAS)

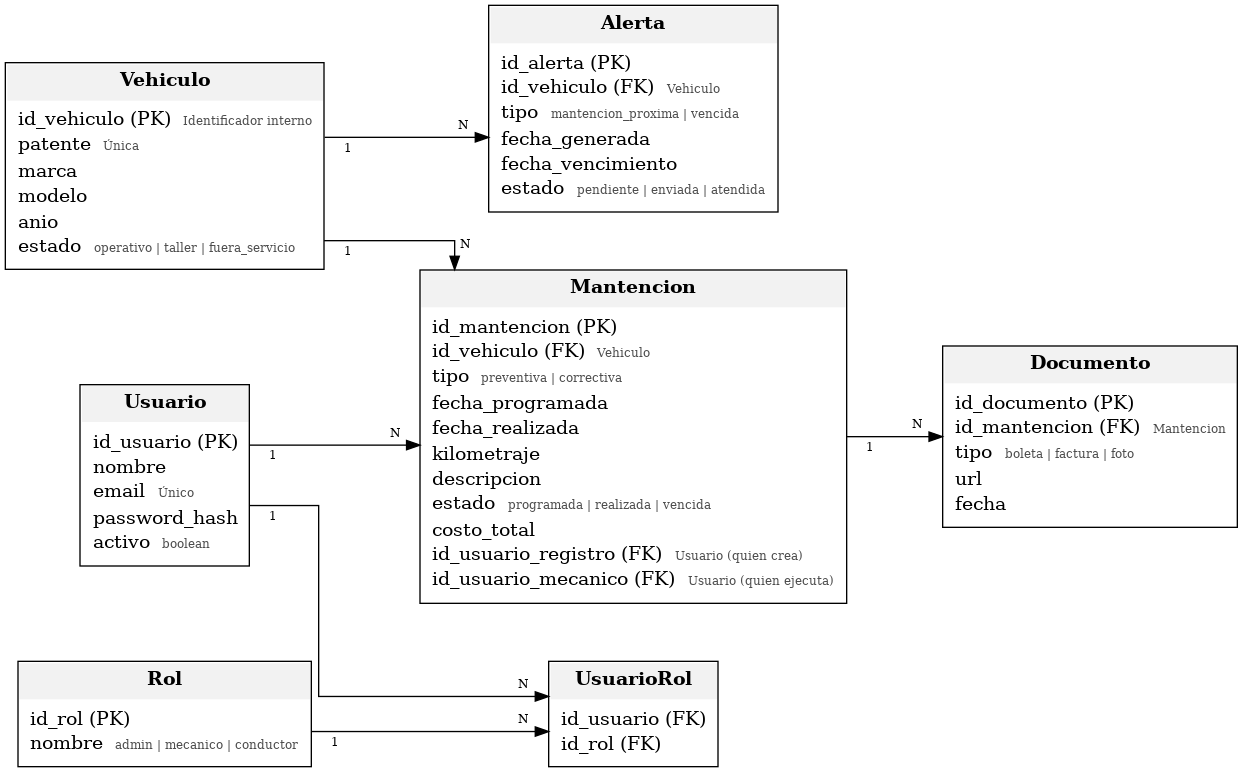
### Diagrama de casos de uso:

El diagrama modela las interacciones entre los actores Administrador, Supervisor (con la especialización Mecánico) y Conductor con el Sistema de Gestión de Mantenciones de Vehículos de Emergencia (SGMVE). Se representan las capacidades clave alineadas a los requerimientos: gestión de vehículos (RF1), gestión de mantenciones (RF2) —con adjuntos (RF6) y costos (RF8) como extensiones del registro—, consulta del historial con búsqueda y filtrado (RF4–RF9), generación y exportación de reportes (RF7), recepción de alertas (RF3), consulta de estado del vehículo (RF10) y autenticación/gestión de usuarios y roles (RF5). Las relaciones <<include>> y <<extend>> explicitan dependencias funcionales y variantes, mientras que el límite de sistema delimita claramente el alcance. Este diagrama asegura trazabilidad directa con los RF definidos y ofrece una visión comprensible del comportamiento esperado desde la perspectiva de los usuarios.



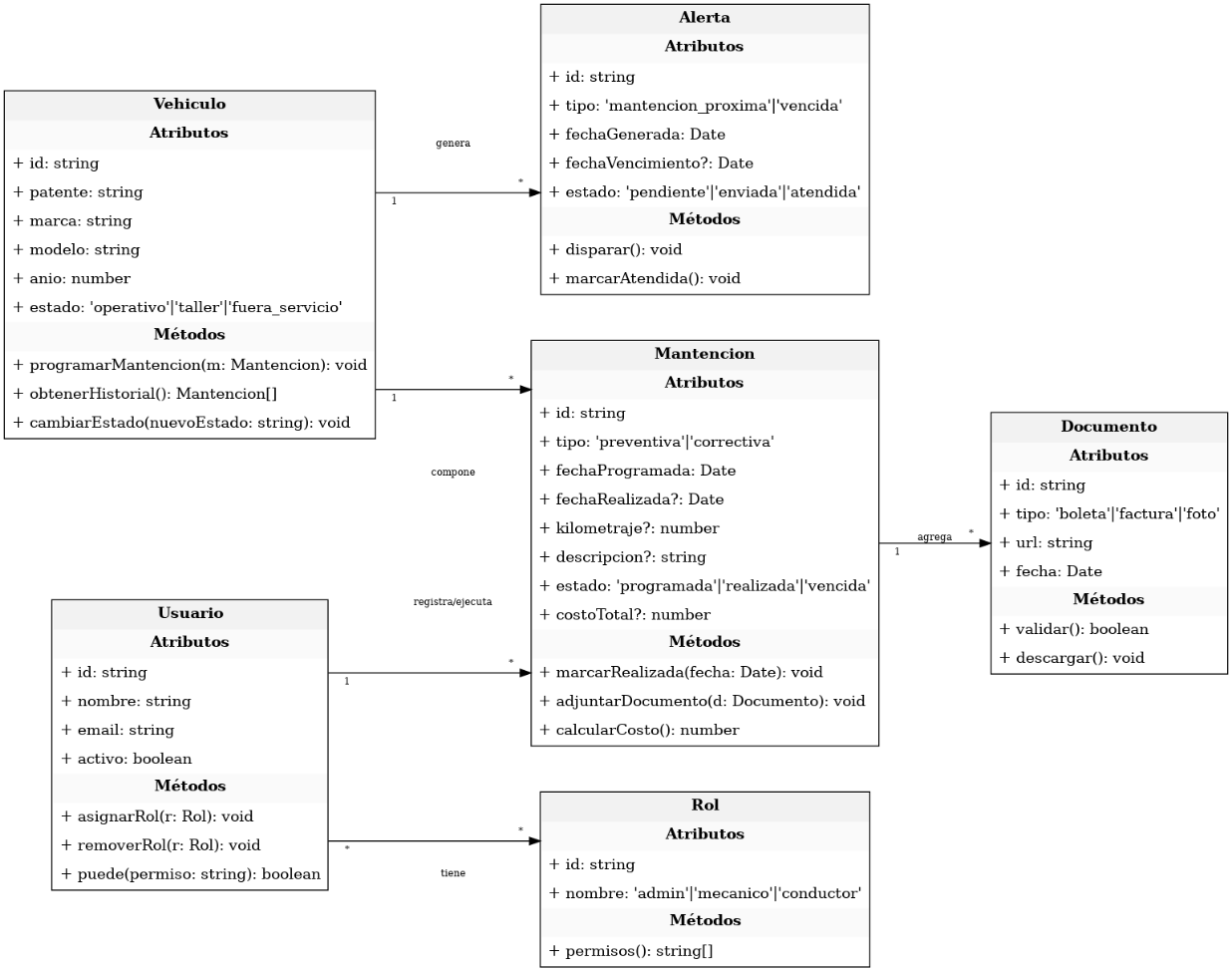
### Diagrama Entidad-Relación (DER)

El modelo de datos del sistema se compone de las entidades Vehículo, Mantención, Documento, Alerta, Usuario, Rol y UsuarioRol. Se establecen relaciones 1–N entre Vehículo–Mantención y Mantención–Documento, 1–N entre Vehículo–Alerta, y M–N entre Usuario–Rol, resuelta mediante la tabla intermedia UsuarioRol. Las claves primarias son numéricas; se definen restricciones de unicidad sobre la patente de Vehículo y el correo electrónico de Usuario. Los atributos de tipo y estado en Mantención y Alerta se gobiernan mediante dominios controlados, lo que garantiza la integridad semántica. El diseño incorpora políticas de borrado que preservan el historial operativo (ON DELETE RESTRICT en Vehículo–Mantención; ON DELETE CASCADE en Mantención–Documento) y admite el registro de adjuntos y costos por intervención, asegurando la trazabilidad de cada acción. Adicionalmente, se contemplan índices específicos para acelerar consultas por patente, fechas, estado y vehículo, en concordancia con los requisitos de rendimiento y generación de reportes definidos. En conjunto, este DER proporciona una base normalizada, consistente y escalable, alineada con los requerimientos funcionales (RF1–RF10) y con los no funcionales de seguridad, rendimiento y mantenibilidad.



### Diagrama de Clases:

El diagrama de clases refleja cómo estructuramos el sistema desde un enfoque orientado a objetos. En él definimos las clases principales, sus atributos, métodos y las relaciones entre ellas. Este modelo es clave porque marca la transición entre el análisis y la implementación, mostrando cómo se llevarán a cabo las funcionalidades en el código. En este proyecto, el diagrama de clases nos asegura que la lógica de negocio, como la gestión de mantenciones, documentos, alertas y roles de usuario, esté bien modelada y lista para ser implementada en el desarrollo.



# Tecnologías

Arquitectura General:

* SPA Web (Angular 17)
* Objetivo: entrega rápida, bajo costo operativo, seguridad y escalabilidad.

Frontend Web:

* Angular 17 (standalone, routing, guards, reactive forms).
* SCSS (arquitectura modular; variables/mixins).
* UI: Angular Material (theming con SCSS).
* Reportes en cliente: pdfmake / jsPDF (PDF) y SheetJS/xlsx (Excel).

Backend / Integración (Cloud Functions v2, Node 20):

* SDKs Firebase: firebase-admin, firebase-functions (v2).
* Auth con Custom Claims (roles) + Firestore/Storage Rules.
* Cloud Firestore (NoSQL) con índices compuestos para consultas y filtros.
* Cloud Storage para comprobantes con metadatos.
* Cloud Scheduler para jobs (alertas próximas/vencidas).
* FCM para notificaciones push.
* Librerías de Node (recomendadas y compatibles con Node 20):
  + HTTP y seguridad (en onRequest con Express, si aplica): express, cors, helmet, hpp.
  + Validación y sanitización: zod (o ajv), validator, sanitize-html.
  + Utilidades: uuid, mime-types (o file-type), date-fns / dayjs.
  + Reportes server-side (lotes): pdfmake / pdf-lib (PDF), exceljs / xlsx (Excel).
  + Correo: nodemailer (para avisos por email si corresponde).
  + Logs: pino o winston.
  + Rate limiting (si expones endpoints HTTP): express-rate-limit o rate-limiter-flexible.
  + Config: dotenv solo en emuladores; en producción usar Config/vars de Functions.

Observabilidad y seguridad:

* Backups programados de Firestore/Storage (RPO ≤ 24 h).
* Rules basadas en Claims (mínimo privilegio).
* CORS y HTTPS en endpoints; validación de entrada (DTO con zod/ajv).

DevOps / Entornos:

* GitHub (ramas : main, develop, feature/\*).
* CI/CD con GitHub Actions (Node 20): build, tests y depliage a Firebase Hosting / Play Console.
* Versionado semántico y feature toggles para módulos nuevos.

Pruebas:

* Unitarias: Jest / testing de Angular; coverage en módulos críticos.
* Caja negra: validación manual de flujos clave (crear mantención, generar alerta, exportar reporte).
* Rules: firebase/rules-unit-testing.
* Calidad Web: Lighthouse (performance y accesibilidad).

Justificación RF/RNF:

* RF1–RF4, RF8–RF9: Firestore + índices → historiales, consultas y filtrados eficientes.
* RF3: Scheduler + Functions + FCM/nodemailer (correo) → alertas.
* RF6: Storage + Rules por rol/propiedad → adjuntos seguros.
* RF7: pdfmake/jsPDF y SheetJS (cliente) + exceljs/pdfmake (lotes en Functions).
* RNF3: Auth + Claims + Rules + helmet/cors/zod → seguridad.
* RNF4/RNF10: Functions v2 Node 20 ligeras, caché local y medición p95.
* RNF8: backups programados Firestore/Storage y pruebas de restauración.

# Costos del Proyecto

El siguiente presupuesto refleja los recursos mínimos para desarrollar e implementar un sistema web de gestión de mantenciones para la Compañía de Bomberos de Talcahuano. Aunque el proyecto es académico, los costos son representativos y viables en un escenario real.

Recursos tecnológicos:

* Dominio .cl: $12.000 CLP/año
* Hosting (Firebase Hosting): $0 CLP en uso académico controlado; se reserva bolsón anual para escalamientos menores.
* Servicios en la nube (Firebase – Firestore, Functions, Storage, FCM): $50.000 CLP/año (bolsón para cuotas de uso fuera del nivel gratuito).
* Herramientas de diseño/gestión: Figma (licencia básica) $36.000 CLP/año; Trello/Jira (plan gratuito).

Recursos Humanos:

* Desarrollo: 2 desarrolladores × 400 h c/u = 800 h × $7.000 CLP = $5.600.000 CLP
* Pruebas y revisión: 40 h × $7.000 CLP = $280.000 CLP

Otros Gastos:

* Comunicación y reuniones (internet, llamadas): $30.000 CLP
* Imprevistos (10% del subtotal): $600.800 CLP (cálculo sobre el subtotal sin imprevistos)

|  |  |
| --- | --- |
| **Rubro** | **Costo estimado** |
| Dominio + Hosting (Firebase) | $12.000 + $0 |
| Servicios en la nube (Firebase) | $50.000 |
| Herramientas de diseño/gestión | $36.000 |
| Horas de desarrollo | $5.600.000 |
| Pruebas y revisión | $280.000 |
| Comunicación y reuniones | $30.000 |
| Imprevistos (10%) | $600.800 |
| Total estimado | $6.608.800 |

# Riesgos

En todo proyecto de desarrollo tecnológico existen factores que pueden afectar el cumplimiento de los objetivos, los plazos y la calidad final.

A continuación, se detallan los riesgos principales del sistema de gestión de mantenciones de vehículos de emergencia para la Compañía de Bomberos de Talcahuano, junto con su impacto y las medidas de control.

En este proyecto solo se identificaron algunos riesgos generales, ya que el enfoque principal es cumplir con los tiempos y entregables definidos:

* **Retrasos por carga académica:** al tener otras asignaturas como la práctica profesional y evaluaciones en paralelo, existe la posibilidad de que se retrase alguna entrega parcial.
* **Problemas técnicos con Firebase:** al trabajar con la versión gratuita, pueden surgir limitaciones o cambios en el servicio que dificulten el desarrollo.
* **Errores de seguridad en pruebas:** una mala configuración de los roles o accesos podría generar fallas en la autenticación durante las pruebas.
* **Costos adicionales por imprevistos técnicos**: Eventual necesidad de integrar librerías externas, servicios de terceros o ampliación de recursos en la nube.

Estos riesgos se irán controlando dentro de cada iteración para que no afecten el avance general del sistema.

# Pruebas

Las pruebas no se realizan como una etapa aparte, sino que se repiten dentro de cada ciclo de desarrollo. En cada iteración se harán los siguientes tipos de pruebas:

* **Unitarias:** para revisar que los métodos y funciones del sistema funcionen de forma correcta.
* **De integración:** para confirmar que los distintos módulos trabajen bien entre sí.
* **De usabilidad y accesibilidad:** para comprobar que la interfaz sea simple de usar y cumpla con estándares básicos de accesibilidad.
* **De rendimiento:** para medir que los tiempos de respuesta se mantengan dentro de lo esperado (ejemplo: consultas rápidas y navegación fluida).
* **De caja negra**: verificación de las funcionalidades clave desde la perspectiva del usuario (ej.: registro de vehículo, mantención, generación de alerta, exportación de reportes).

De esta manera, cada ciclo entrega un avance probado y funcional, evitando acumular errores al final del proyecto.

# Backlog del Sistema

**Proyecto:** Sistema de Control y Seguimiento de Mantención de Vehículos de Emergencia  
 **Metodología:** Iterativo–Incremental sin cliente

### **1. Requerimientos Funcionales (RF)**

**RF1. Gestión de vehículos**

* RF1.1 Registrar vehículos con patente (única), marca, modelo, año, estado operativo y observaciones.
* RF1.2 Editar datos de vehículos.
* RF1.3 No eliminar físicamente vehículos con mantenciones asociadas (baja lógica / RESTRICT).
* RF1.4 Consultar ficha técnica e historial consolidado por vehículo.

**RF2. Gestión de mantenciones**

* RF2.1 Registrar mantenciones preventivas y correctivas con fecha, tipo, trabajo realizado, responsable, estado e insumos.
* RF2.2 Editar/anular mantenciones solo si no están realizadas ni con costos liquidados.
* RF2.3 Mantener historial completo por vehículo.

**RF3. Alertas y notificaciones**

* RF3.1 Generar alertas por mantenciones próximas o vencidas (tiempo/kilometraje).
* RF3.2 Notificar por correo y push (FCM) a roles involucrados.

**RF4. Historial y consulta**

* RF4.1 Visualizar historial de intervenciones con filtros por fecha, tipo, estado y responsable.

**RF5. Gestión de usuarios y roles**

* RF5.1 Autenticación con Firebase Auth.
* RF5.2 Control de acceso por roles (administrador, supervisor/mecánico, conductor).
* RF5.3 Actualización de perfil y recuperación de contraseña.

**RF6. Adjuntos de mantención**

* RF6.1 Subir documentos asociados (boletas, facturas, fotos) en PDF/JPG/PNG.
* RF6.2 Visualizar y descargar documentos con control de acceso.

**RF7. Reportes y exportación**

* RF7.1 Generar reportes de flota, frecuencia y costos.
* RF7.2 Exportar a PDF y Excel con filtros aplicados.

**RF8. Registro de costos**

* RF8.1 Registrar costos por intervención (repuestos, mano de obra).
* RF8.2 Consolidar totales por vehículo y período.

**RF9. Búsqueda y filtrado**

* RF9.1 Búsqueda por patente.
* RF9.2 Filtrar vehículos/mantenciones por fecha, tipo y estado.

**RF10. Movilidad**

### **2. Requerimientos No Funcionales (RNF)**

**RNF1. Seguridad**

* Autenticación/autorización por roles (Auth + Claims + Reglas Firestore/Storage).
* TLS ≥ 1.2, auditoría de acciones sensibles.
* Hashing seguro en contraseñas (Firebase Auth).
* Validación de entrada y protección de endpoints.

**RNF2. Usabilidad y accesibilidad**

* Interfaz intuitiva; SUS ≥ 75.
* Error en tareas críticas < 2%.
* Cumplimiento WCAG 2.0.

**RNF3. Rendimiento**

* Backend: 50 usuarios concurrentes; p95 API < 1.5 s.
* Frontend: LCP < 4 s; TTI < 5 s; navegación < 3 s.

**RNF4. Disponibilidad**

* SLO ≥ 99% mensual en horario operativo.
* Mantenciones programadas preavisadas ≥ 24 h.

**RNF5. Escalabilidad**

* Escalar al doble de usuarios concurrentes sin rediseños críticos.

**RNF6. Compatibilidad**

* Últimas 2 versiones de Chrome/Edge/Firefox.
* Android 10+; iOS 14+ (opcional).

**RNF7. Mantenibilidad**

* Cobertura de pruebas unitarias ≥ 60%.
* Código sin errores de linter.
* Documentación de API actualizada.

**RNF8. Respaldo y recuperación**

* Backups diarios, retención 30 días.
* RPO ≤ 24 h, RTO ≤ 4 h, prueba mensual de restauración.

**RNF9. Portabilidad / Despliegue**

* Guía de despliegue en Firebase Hosting.
* Variables de entorno parametrizadas.

# Plan de Trabajo

El proyecto se gestionará con un **proceso iterativo–incremental**, donde en cada iteración se desarrolla un conjunto de funcionalidades, se realizan pruebas internas y se documenta el avance. Cada iteración tendrá una duración fija de 2 semanas.

**Iteraciones definidas**

|  |  |  |  |
| --- | --- | --- | --- |
| Iteración | Duración estimada | Actividades principales (alcance) | Entregables / Criterio de salida |
| Iteración 0 | 2 semanas | Plan y setup: Backlog priorizado; DAS/DER/Clases final; repos y estructura, Emuladores Firebase; Reglas mínimas de Firestore/Storage; índices iniciales; Plan de pruebas | Backlog “Ready”; entornos y pipelines operativos; reglas mínimas funcionando; plan de pruebas y DoD publicados. |
| Iteración 1 | 2 semanas | RF1 Vehículos (ABM con patente única, ficha e historial básico). RF5 Usuarios/Roles (Auth + Claims; navegación por permisos). Estilo base (SCSS/Material). | Módulo Vehículos funcional + login/roles aplicados; unitarias y pruebas de reglas aprobadas; defectos críticos = 0. |
| Iteración 2 | 2 semanas | RF2 Mantenciones (preventivas/correctivas, estados, kilometraje, insumos). RF3 Alertas (Scheduler+Functions; correo y web push). RF10 Parámetros (umbrales y catálogos). | Pruebas de caja negra de alta/edición/alerta aprobadas. |
| Iteración 3 | 2 semanas | RF6 Adjuntos (Storage + metadatos + reglas). RF8 Costos (repuestos/mano de obra; totales por unidad/período). RF9 Búsqueda/Filtrado (patente, fecha, tipo, estado). | Adjuntos seguros; costos consolidados; búsqueda/filtrado operativos; pruebas de caja negra correspondientes aprobadas. |
| Iteración 4 | 2 semanas | RF7 Reportes y exportación (estado de flota, mantenciones, costos; PDF/Excel). Accesibilidad en vistas principales; mejoras de UX. | Reportes configurables con exportación fiel a filtros; SUS ≥ 75; checklist de accesibilidad aprobado. |
| Iteración 5 | 2 semanas | Integración y hardening: rendimiento, seguridad (rules/roles), backup/restore, estabilización e interna (docente revisor). | Métricas de rendimiento cumplidas; simulacro de restauración exitoso; UAT aprobada; defectos críticos = 0. |
| Fase de Cierre | 2 semanas | Pruebas finales puntuales, documentación técnica y manual de usuario; empaquetado de evidencias; presentación y defensa. | Informe final, manuales, artefactos/evidencias, presentación y acta de cierre. |

# Carta Gantt

|  |  |  |  |
| --- | --- | --- | --- |
| Iteración | Fechas | Objetivo principal | Entregables |
| 0. Configuración inicial | 08/09 – 19/09 | Backlog priorizado, diagramas finales, setup de repos y reglas mínimas | Entorno operativo, backlog listo, plan de pruebas |
| 1. Vehículos + Usuarios/Roles | 22/09 – 03/10 | Login/autenticación, gestión de roles, módulo de vehículos con historial básico | Módulo de vehículos funcional, roles aplicados, pruebas de caja negra |
| 2. Mantenciones + Alertas + Parámetros | 06/10 – 17/10 | Registro de mantenciones, estados, alertas automáticas y notificaciones | Flujos de mantención y alertas probados |
| 3. Adjuntos + Costos + Búsquedas | 20/10 – 31/10 | Subida de documentos, consolidado de costos y filtros de búsqueda | Adjuntos seguros, costos consolidados, búsquedas operativas |
| 4. Reportes + Accesibilidad | 03/11 – 14/11 | Reportes configurables y exportables, mejoras de UX y accesibilidad (WCAG 2.0) | Reportes en PDF/Excel, validación de accesibilidad |
| 5. Integración + Seguridad + Rendimiento | 17/11 – 28/11 | Integración final de módulos, pruebas de seguridad y rendimiento | Sistema integrado, validación de seguridad y rendimiento |
| 6. Cierre y Defensa | 01/12 – 12/12 | Pruebas finales, manual de usuario, documentación final, preparación y defensa | Informe final, manual, presentación y defensa |

Gráfico, Escala de tiempo

El contenido generado por IA puede ser incorrecto.El proyecto se gestionará mediante un proceso iterativo–incremental, con iteraciones de 2 semanas hábiles. En cada iteración se desarrolla un conjunto de funcionalidades, se aplican pruebas de caja negra, usabilidad y seguridad, y se documenta el avance.