

# UNIVERSIDAD DE TARAPACÁ



## FACULTAD DE INGENIERÍA

Departamento de Ingeniería en Computación e Informática.

# Sistema antirrobo con notificación y aviso inteligente



**Autor(es):** Renato Almeyda.

Jeany Aravena

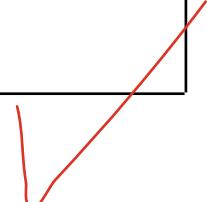
Bastián Cruz.

Josue Sucso.

**Académico:** Diego Aracena.

## Historial de Cambios.

Fecha	Versión	Descripción	Autor(es)
14/10	0.1	Versión preliminar del formato	Renato Almeyda Jeany Aravena Bastián Cruz Josue Sucso
28/10	1.0	Desarrollo de requerimientos solicitados	Bastián Cruz Josue Sucso
22/11	2.0	Integración de los modelos de diseño(caso de uso general, diagrama de clases y secuencia) y la sección de herramientas y técnicas utilizadas	Bastián Cruz Josue Sucso
24/11	2.1	Errores corregidos e integración de las secciones faltantes “conclusiones” y “referencias”	Bastián Cruz Josue Sucso



## Tabla de contenidos.

<b>Historial de Cambios.</b>	<b>2</b>
<b>Tabla de contenidos.</b>	<b>3</b>
<b>Índice de tablas</b>	<b>4</b>
<b>Índice de figuras</b>	<b>4</b>
<b>I. Panorama General</b>	<b>5</b>
1.0 Resumen del proyecto:	5
1.1 Propósito del proyecto:	5
1.2 Alcances del proyecto:	5
1.3 Objetivos específicos:	6
1.4 Suposiciones y restricciones:	6
1.4.1 Suposiciones:	6
1.4.2 Restricciones:	7
1.5 Entregables del proyecto:	7
<b>II. Organización del proyecto</b>	<b>8</b>
2.1 Personal y entidades externas	8
2.2 Roles y responsabilidades	8
2.3 Mecanismos de Comunicación	9
<b>III. Planificación de los procesos de gestión.</b>	<b>10</b>
3.1 Planificación inicial del proyecto	10
3.1.1 Planificación de estimaciones	10
3.1.2 Planificación de Recursos Humanos.	10
3.2 Lista de actividades (carta Gantt)	12
3.2.1 Actividades de trabajo	12
3.2.2 Asignación de tiempo	13
3.3 Planificación de la gestión de riesgos.	14
3.4 Poster Asociado.	15
5.0 Planificación de procesos técnicos	16
5.1 Modelo de procesos	16
5.1.1 Requisitos funcionales y no funcionales	16
5.1.1.1 Tabla de requerimientos funcionales	16
5.1.1.2 Tabla de requerimientos no funcionales	17
5.1.2 Diagrama de caso de uso general	18
5.1.3 Descripción de la arquitectura	18
5.1.4 Interfaz de usuario	19
5.1.5 Diagrama de clases	20
5.1.6 Diagrama de secuencias	20
5.1.7 Descripción de la arquitectura con respecto a los modelos	21
5.2 Herramientas y técnicas	22
5.2.1 Herramientas:	22
5.2.1 Técnicas utilizadas:	23
6. Conclusiones	24

7. Referencias	25
Referencias de Hardware y Componentes	25
Referencias de Software y Desarrollo	25



## Índice de tablas

Tabla 1: Entregables del proyecto	7
Tabla 2: Planificación de estimaciones	11
Tabla 3: Planificación de recursos humanos	11
Tabla 4: Planificación de recursos totales	12
Tabla 5: Planificación de gestión de riesgos.	16
Tabla 6: Requisitos funcionales.	18
Tabla 7: Requisitos no funcionales.	18

## Índice de figuras

Figura 1: Carta Gantt	12
Figura 2: Poster asociado.	15
Figura 3: Diagrama de casos de uso general	18
Figura 4: Diagrama de la arquitectura.	18
Figura 5: Boceto de Interfaz de usuario	19
Figura 6: Diagrama de Clases.	20
Figura 7: Diagrama de Secuencias	21



## I. Panorama General

### 1.0 Resumen del proyecto:

En la actualidad, el robo de vehículos representa un problema constante que afecta la seguridad y economía de los propietarios. Este proyecto propone una solución tecnológica basada en tecnología IoT mediante el desarrollo de un sistema de notificación inteligente que permite detectar y reportar de forma automática la conducción no autorizada de un vehículo, es decir, informar donde efectivamente el vehículo fue sustraído del dueño

Este sistema se implementa sobre una Raspberry Pi 4, la cual integra un acelerómetro, una cámara para lectura de códigos QR dinámicos, un display LCD para su interfaz, además de unas luces LED de manera complementaria y una conexión con la aplicación móvil del propietario. A través de esta infraestructura, se busca ofrecer un método moderno de autenticación vehicular y alerta inmediata, contribuyendo a la prevención del robo y al fortalecimiento de la seguridad automotriz.

### 1.1 Propósito del proyecto:

Proveer a los propietarios de vehículos una herramienta activa e inmediata para mitigar el riesgo de robo, transformando la seguridad automotriz de un enfoque reactivo a uno preventivo y de respuesta en tiempo real mediante la integración de la tecnología IoT.

### 1.2 Alcances del proyecto:

- **Hardware:** Raspberry Pi 4, sensor de cámara, sensor acelerómetro, módulo de red, display LCD, luces LED complementarias.

- Software: app móvil, lectura QR, base de datos y sistema de notificaciones.
- Simulación VR: entorno Unity para visualización del contexto real situado.



### 1.3 Objetivos específicos:

- Detectar movimiento vehicular mediante sensor acelerómetro.
- Implementar autenticación QR dinámica con cámara.
- Enviar notificaciones al propietario del vehículo.
- Crear simulación VR del sistema en el vehículo.



### 1.4 Suposiciones y restricciones:

#### 1.4.1 Suposiciones:

- El propietario del vehículo dispone de un teléfono con conexión a Internet y la aplicación instalada.
- La cámara de la Raspberry Pi 4 puede leer el código QR dinámico en condiciones normales de luz.
- El acelerómetro detecta correctamente el movimiento del vehículo sin error alguno.
- El sistema cuenta con una fuente de energía estable, sea batería o conectada por cable.
- El entorno de simulación en Unity representa de forma adecuada el contexto real.



#### 1.4.2 Restricciones:

- El desarrollo debe completarse dentro de los tiempos solicitados.
- Solo se cuenta con una Raspberry Pi 4 y sensores limitados.
- El sistema depende de conexión Wi-Fi; no incluye comunicación móvil.
- El presupuesto es limitado.



#### 1.5 Entregables del proyecto:

Revisión	Entregables	AS	AQ	ET
Parte del 1er informe	Formulación del Proyecto Corregido	X		
Parte del 2do informe	Diagramas de diseño, herramientas y técnicas utilizadas, errores del 1er informe corregidos	X		

Tabla 1: Entregables del proyecto



## II. Organización del proyecto

### 2.1 Personal y entidades externas

- Jefe del proyecto: Renato Almeyda
- Programador(es): Renato Almeyda , Jeany Aravena
- Diseñador: Bastián Cruz
- Ensamblador: Josue Sucso
- Documentador: Bastián Cruz

### 2.2 Roles y responsabilidades

- **Jefe del proyecto:** Encargado de coordinar y supervisar el correcto avance de todos los procesos que componen el desarrollo del proyecto. Es representante del equipo de trabajo ante los profesores encargados del ramo y el resto de los equipos.
- **Diseñador:** Se encarga de diseñar la interfaz de usuario y la experiencia de usuario de la *aplicación móvil de alerta*. Es responsable de crear el flujo de notificaciones que recibe el propietario.
- **Programador(es):** Encargados de escribir el código que se ejecuta en el Raspberry Pi 4. Su trabajo es asegurar que el dispositivo pueda leer correctamente la información de los sensores, tomar las decisiones de seguridad (como verificar la velocidad y el código QR), y coordinar todas las partes del sistema para que funcionen de manera sincronizada.
- **Ensamblador:** Encargado de la preparación física y el montaje del sistema. Sus responsabilidades incluyen: ensamblar el Raspberry Pi 4 con sus componentes (caja o chasis), cablear y conectar correctamente todos los sensores (velocidad, cámara) y periféricos al microcontrolador, y asegurar que la instalación del hardware sea funcional dentro del vehículo.
- **Documentador:** Responsable de la gestión integral de la información del proyecto. Esto incluye la elaboración de informes de avance y bitácoras, así como la creación y mantenimiento de la Wiki para documentación técnica. Es el encargado de administrar la Carta Gantt dentro de la plataforma Redmine para el seguimiento y control del proyecto.

## 2.3 Mecanismos de Comunicación

**Canales internos:** correo institucional, grupo de WhatsApp, Discord.

**Documentación compartida:** Google Drive y **GitHub** (repositorio del proyecto).

### Comunicaciones y Estándares Técnicos:

#### Lenguajes de Programación:

- **Python:** Para la lógica del sistema en la Raspberry Pi 4 y el procesamiento de datos del acelerómetro y la cámara.
- **Kotlin:** Para el desarrollo de la aplicación móvil nativa (Android).



#### Lenguajes de Interfaz y Datos:

- **JSON (JavaScript Object Notation):** Como estándar de intercambio de datos para la comunicación entre el dispositivo IoT (Raspberry Pi) y la aplicación móvil.



#### Comunicaciones para XR (Realidad Extendida):

- **C#:** Utilizado en el motor Unity para la creación de una maqueta virtual del proyecto, la cual será desplegada en un dispositivo Meta Quest 3.



### III. Planificación de los procesos de gestión.

#### 3.1 Planificación inicial del proyecto

##### 3.1.1 Planificación de estimaciones

Producto	Cantidad	Costo por unidad	Costo Total
Notebook(Uso)	4	\$50.000	\$200.000
Raspberry PI 4	1	\$90.000	\$90.000
Sensor Camara	1	\$5.000	\$5.000
Sensor acelerómetro	1	\$5.000	\$5.000
Grove LCD RGB Backlight	1	\$15.000	\$15.000
Tarjeta SD	1	\$13.000	\$13.000
Total			\$328.000

Tabla 2: Planificación de estimaciones



##### 3.1.2 Planificación de Recursos Humanos.

Roles	Tarifa x Hora
Jefe de proyecto	\$12.000
Programador	\$10.000
Diseñador	\$8.500
Documentador	\$5.000
Ensamblador	\$6.000

Tabla 3: Planificación de recursos humanos

## Proyecto II

Miembro	Rol	Hora x mes	Meses de utilidad	Resultado	Pago Final
Renato Almeyda	Jefe de proyecto	40	4	\$1.920.000	\$3.520.000
	Programador	40	4	\$1.600.000	
Bastián Cruz	Diseñador	40	2	\$680.000	\$1.480.000
	Documentador	40	4	\$800.000	
Josue Sucso	Documentador	40	4	\$800.000	\$1.280.000
	Ensamblador	40	2	\$480.000	
Jeany Aravena	Programador	40	4	\$1.600.000	\$1.600.000
Total					\$7.880.000

Tabla 4: Planificación de recursos totales

Costo total: \$8.208.000



### 3.2 Lista de actividades (carta Gantt)

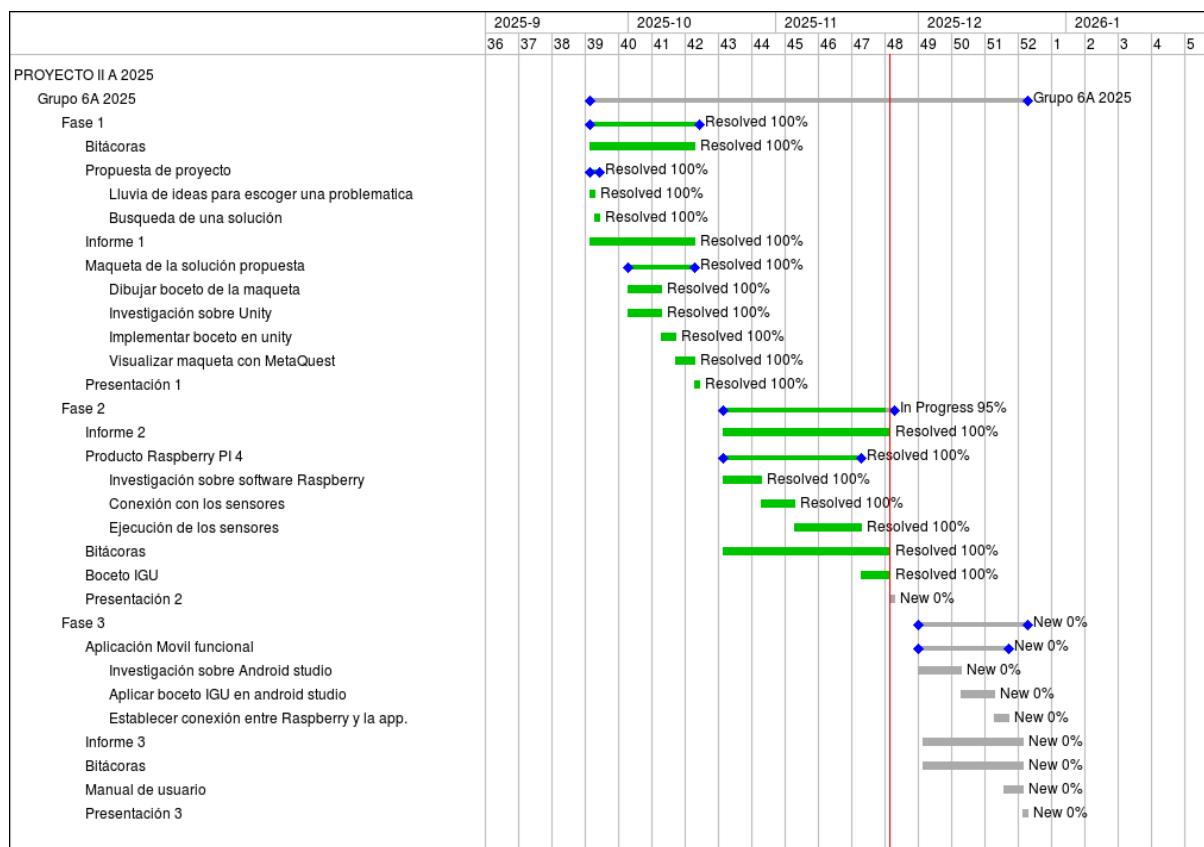


Figura 1: Carta Gantt

#### 3.2.1 Actividades de trabajo

##### 1) Planificación del proyecto:

- Revisión del formato y requisitos del plan de proyecto.
- Definición del problema, propósito y objetivos generales.
- Asignación de roles y responsabilidades dentro del equipo.
- Identificación de los recursos materiales (Raspberry Pi, sensores, cámara, etc.).
- Elaboración del panorama general, suposiciones, restricciones y entregables.
- Estimación de tiempos y costos iniciales del proyecto.

## 2) Ejecución y desarrollo del proyecto:

- **Análisis:** investigar el funcionamiento de los sensores y componentes; definir los requerimientos funcionales y técnicos del sistema.
- **Diseño:** desarrollar el modelo de simulación del contexto en Unity y posteriormente, crear la arquitectura general del sistema, diseñar las interfaces de usuario (app móvil y display LCD)
- **Implementación:** programar la lógica de lectura del QR y acelerómetro en la Raspberry Pi 4; integrar la comunicación con la aplicación móvil y la base de datos; ensamblar físicamente los componentes.
- **Pruebas:** realizar pruebas unitarias, de comunicación y de integración del sistema; validar el correcto funcionamiento de las notificaciones y la autenticación dinámica.

## 3) Cierre del proyecto:

- Elaborar los manuales de usuario e instalación.
- Redactar la documentación técnica final.
- Preparar la presentación del proyecto y la entrega del prototipo.
- Firmar el acta de conformidad final del proyecto.

### 3.2.2 Asignación de tiempo

- Planificación del proyecto: 3 semanas
- Ejecución y desarrollo del proyecto: 11 semanas
- Cierre de proyecto: 2 semanas

### 3.3 Planificación de la gestión de riesgos.

- Niveles de impacto:

- Catastrófico
- Crítico
- Marginal
- Despreciable

Riesgo	Probabilidad de ocurrencia	Nivel de impacto	Acción Remedial
Retraso en la entrega de componentes (sensores, cámara, cables).	70%	2	Reasignar tareas de software mientras se espera el hardware. Avanzar en simulación y documentación.
Fallo en la compatibilidad de librerías entre sensores Grove y Raspberry Pi 4.	60%	2	Buscar alternativas compatibles o adaptar código con librerías Python (ej. smbus, OpenCV, grovepi).
Error en la lectura del QR por condiciones de luz o enfoque.	30%	2	Implementar prueba de iluminación adicional con LED blanco o ajustar contraste por software.
Fallas en la conexión Wi-Fi durante las pruebas.	15%	4	Utilizar red local o conexión directa entre Raspberry y smartphone.
Problemas de programación en la app móvil o en la comunicación con Raspberry.	50%	2	Realizar pruebas modulares (API y comunicación). Dividir tareas por submódulos.
Dificultad del equipo para coordinar horarios o tareas.	20%	3	Planificar reuniones semanales y utilizar Google Drive y WhatsApp para actualizaciones rápidas.
Sobrecarga académica o ausencia de un integrante clave.	20%	3	Reasignar tareas temporalmente y mantener documentación actualizada para continuidad del trabajo.
Problemas de Raspberry y sensores por motivos accidentales	20%	1	Manejar con cuidado el dispositivo Raspberry y cuidar que los sensores no se quemen.
Deriva o calibración incorrecta del acelerómetro	50%	3	Establecer una rutina de calibración inicial del sensor y

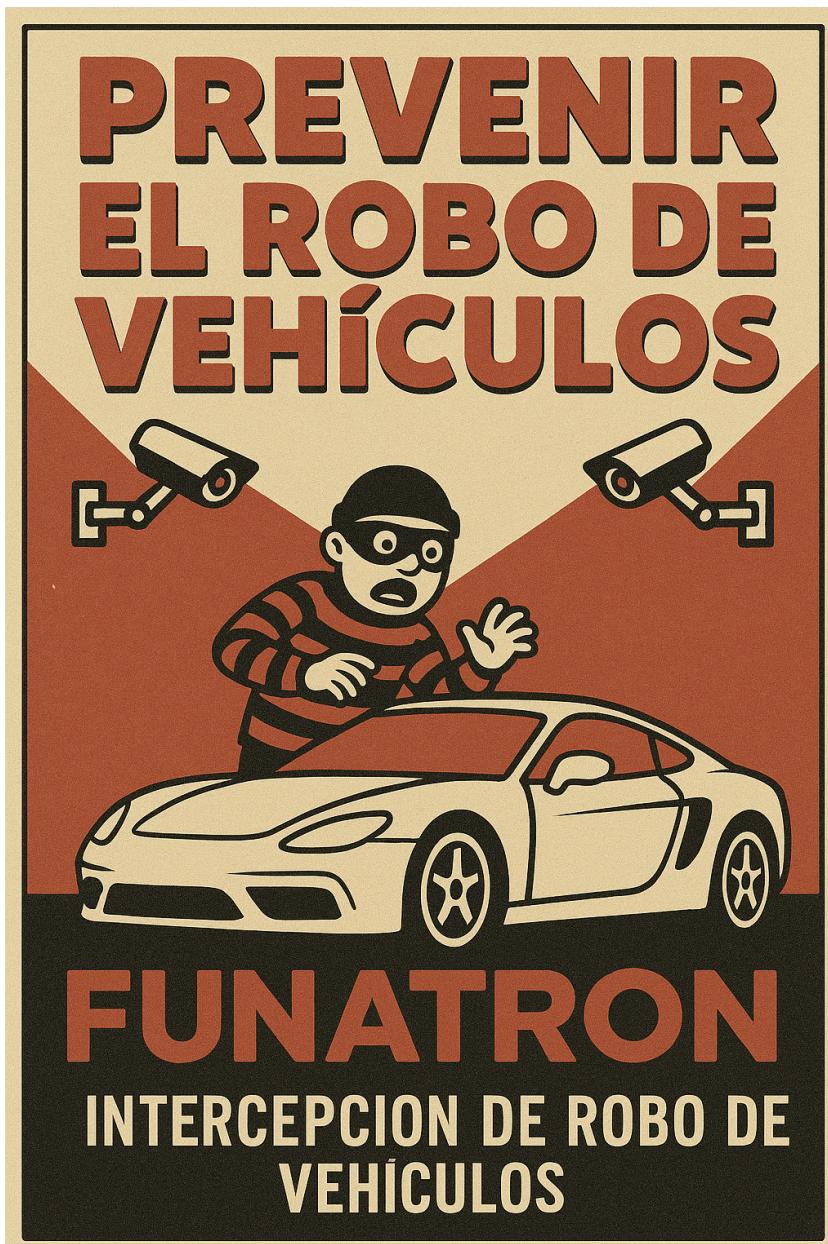
## Proyecto II

			aplicar filtros digitales (ej. Filtro Complementario o Kalman) por software para suavizar las lecturas.
Problemas de seguridad en la transmisión de datos (IoT)	40%	4	Implementar cifrado (SSL/TLS) en la comunicación entre el dispositivo (Raspberry Pi) y la aplicación móvil para proteger la información.

Tabla 5: Planificación de gestión de riesgos.



3.4 Poster Asociado.



se ve bien, no muestra mucho el proyecto.. debe mostrar más detalles

Figura 2: Poster asociado.

## 5.0 Planificación de procesos técnicos

### 5.1 Modelo de procesos

#### 5.1.1 Requisitos funcionales y no funcionales

##### 5.1.1.1 Tabla de requerimientos funcionales

Requisitos funcionales	
<b>RF 1</b>	El sistema debe generar un código QR dinámico que cambie periódicamente para permitir la autenticación del propietario del vehículo.
<b>RF 2</b>	El sistema debe capturar el QR mediante la cámara de la Raspberry Pi para validar al usuario autorizado.
<b>RF 3</b>	El sistema debe verificar el QR escaneado contra el código generado y determinar si es válido o no.
<b>RF 4</b>	El sistema debe permitir al usuario autenticado desactivar la alerta desde la aplicación móvil con el código QR.
<b>RF 5</b>	El sistema debe leer los valores del acelerómetro (GY-6500/9250) para detectar la aceleración del vehículo, permitiendo identificar que hubo arranque y que está en movimiento.
<b>RF 6</b>	El sistema debe enviar una notificación en tiempo real a la aplicación móvil del propietario cuando se detecte movimiento no autorizado al no escanear el QR.
<b>RF 7</b>	El sistema debe permitir al usuario, tras confirmar un evento, enviar los datos a carabineros indicando fecha, hora, modelo del vehículo y una imagen de la persona arribada en el vehículo.
<b>RF 8</b>	El sistema debe almacenar las alertas confirmadas en un registro histórico accesible desde la app.
<b>RF 9</b>	El sistema debe activar un display LCD dependiendo del estado en que se encuentre.
<b>RF 10</b>	El sistema debe permitir registrar información del vehículo (modelo, patente, color, detalles adicionales)

Tabla 6: Requisitos funcionales.

con estos requisitos, el proyecto promete... bien

### 5.1.1.2 Tabla de requerimientos no funcionales

	<b>Requisitos no funcionales</b>
<b>RNF 1</b>	El sistema debe utilizar cifrado para todas las comunicaciones entre la Raspberry Pi y la aplicación móvil para proteger el código QR y la información sensible.
<b>RNF 2</b>	El tiempo de respuesta desde la captura del QR (RF 2) hasta la validación y determinación de si es válido (RF 3) no debe exceder los 2 segundos.
<b>RNF 3</b>	El sistema debe ser capaz de mantener un registro histórico (RF 8) de al menos 6 meses de alertas.
<b>RNF 4</b>	El código del sistema debe ser modular y estar bien documentado para permitir que un nuevo desarrollador pueda entender y modificar la lógica.
<b>RNF 5</b>	El sistema debe ser actualizable de forma remota (OTA - Over-The-Air) para el software de la Raspberry Pi sin requerir acceso físico al vehículo.
<b>RNF 6</b>	El consumo de energía del sistema, cuando está en modo de espera (monitoreando el acelerómetro), debe ser mínimo para no descargar la batería del vehículo.
<b>RNF 7</b>	El tiempo de envío de la notificación de movimiento no autorizado a la aplicación móvil no debe superar los 5 segundos desde que se detecta el movimiento.
<b>RNF 8</b>	La interfaz de usuario (UI) de la aplicación móvil debe ser intuitiva y de fácil navegación.

Tabla 7: Requisitos no funcionales.



### 5.1.2 Diagrama de caso de uso general

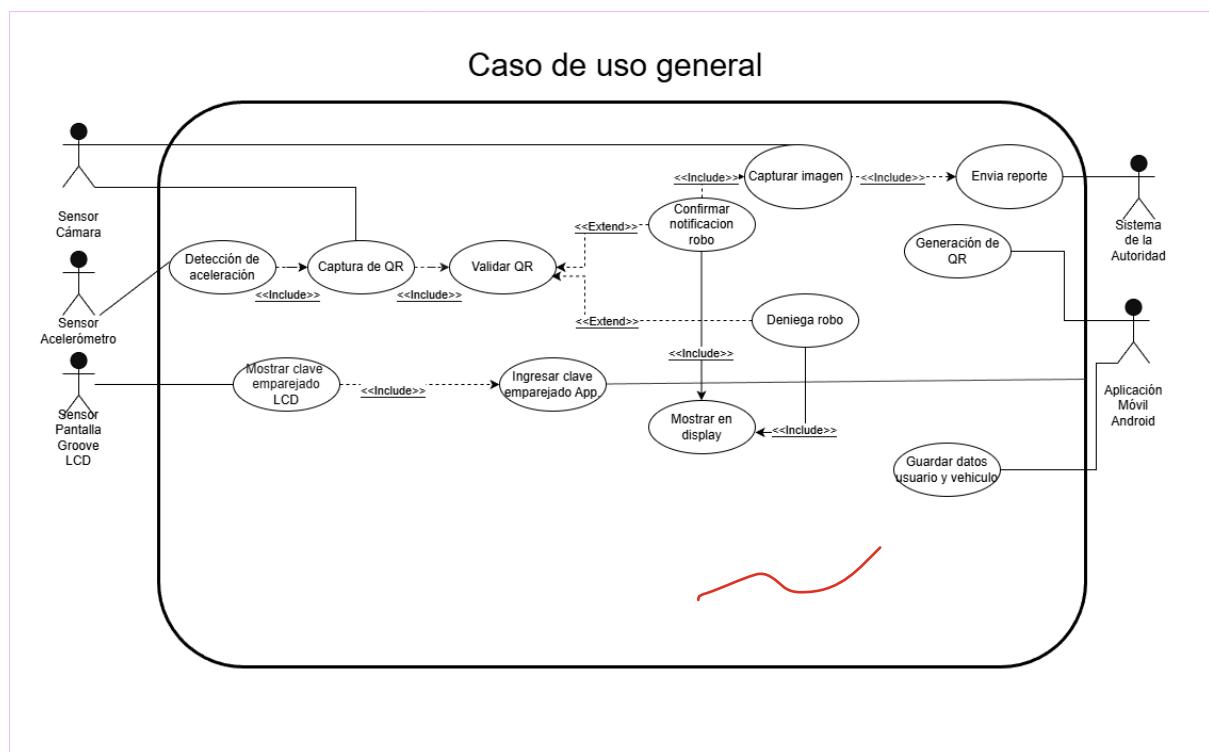


Figura 3: Diagrama de casos de uso general

### 5.1.3 Descripción de la arquitectura

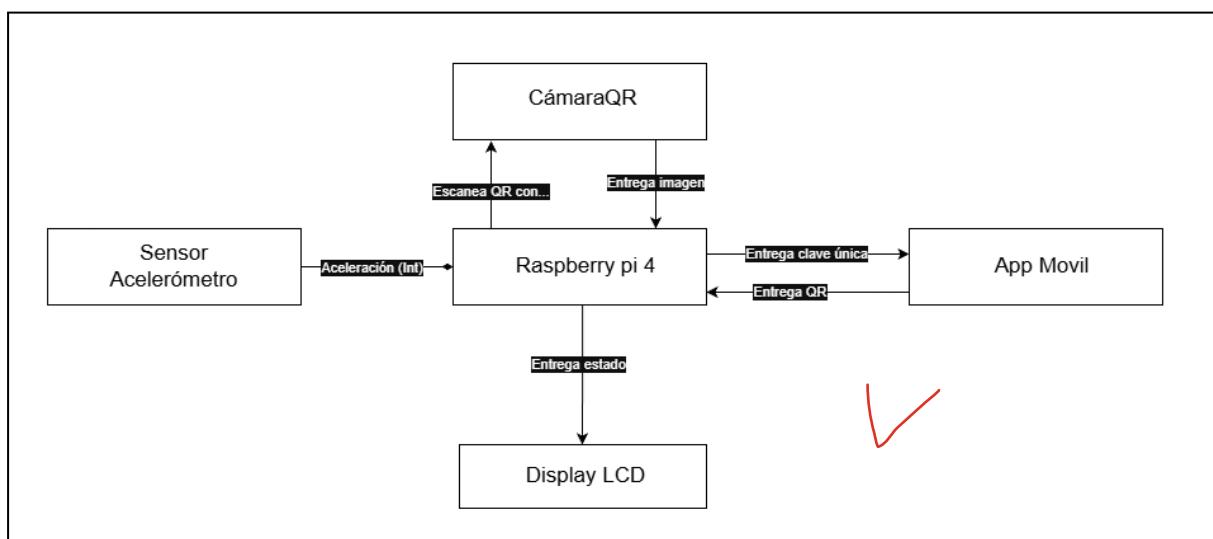
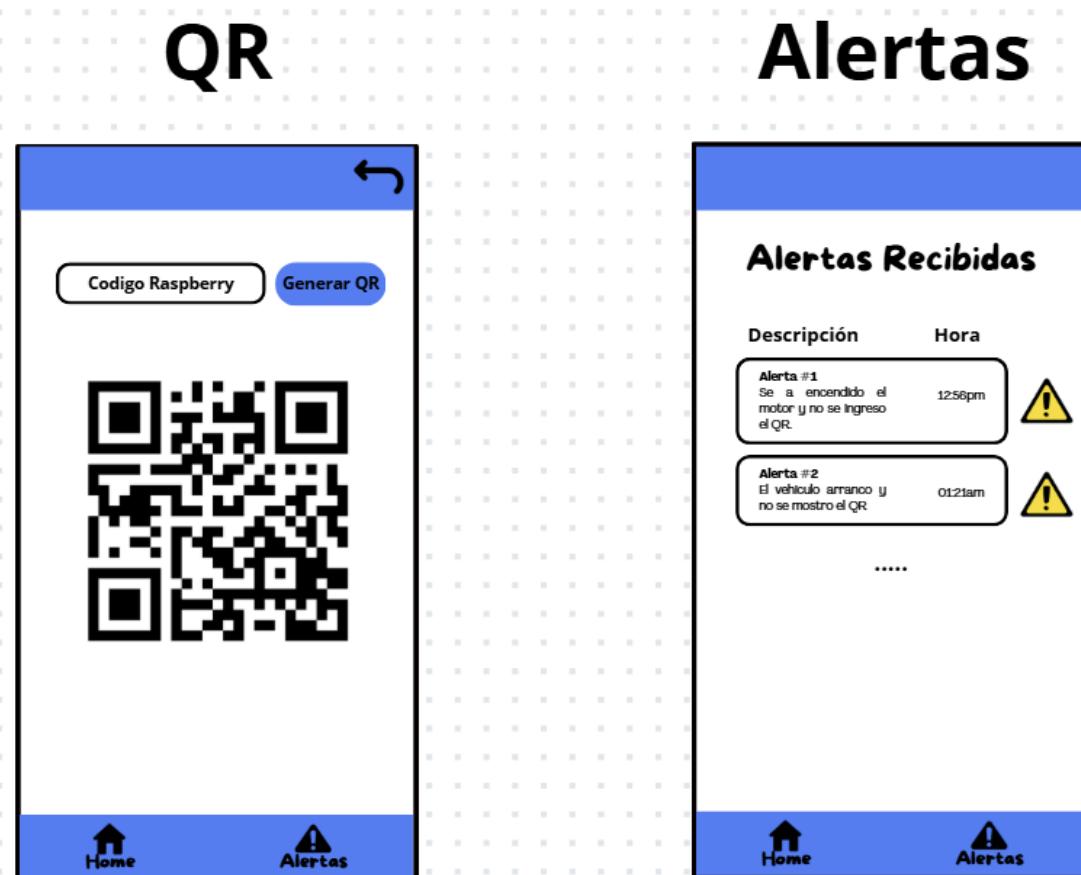
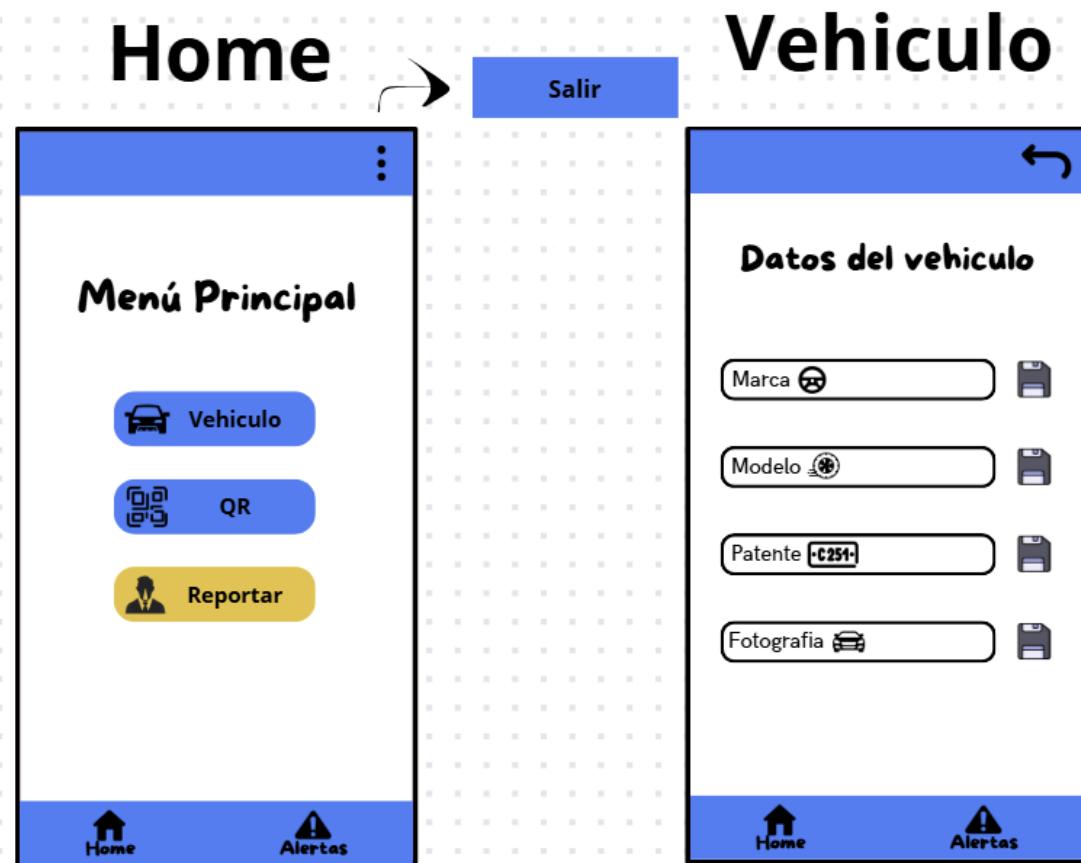


Figura 4: Diagrama de la arquitectura.

falta detecta robo

#### **5.1.4 Interfaz de usuario**



## Proyecto II

Figura 5: Boceto de Interfaz de usuario

### 5.1.5 Diagrama de clases

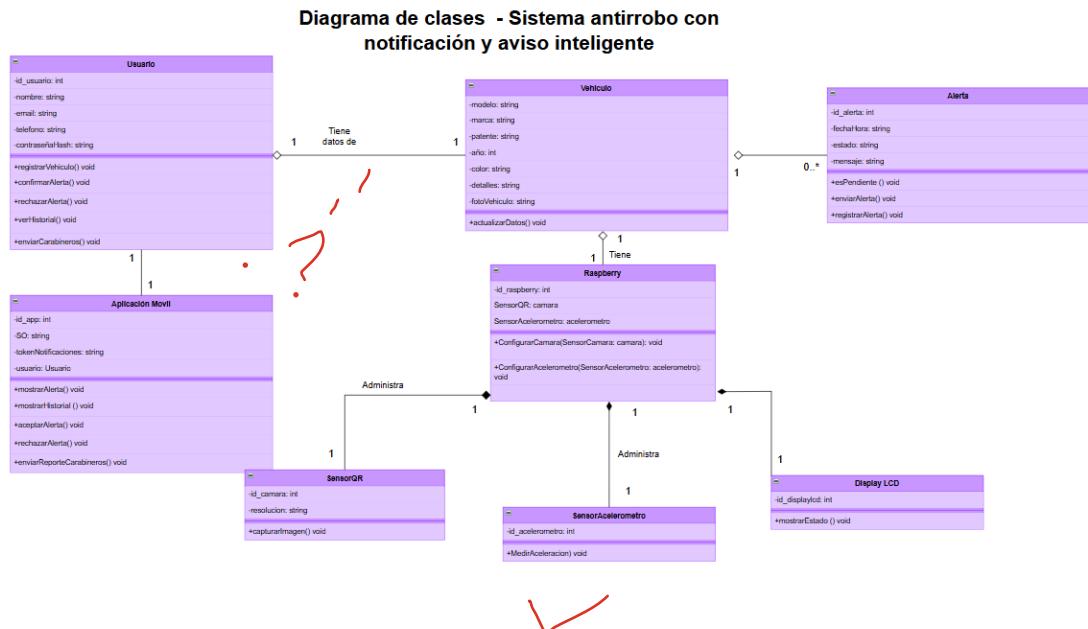


Figura 6: Diagrama de Clases.

### 5.1.6 Diagrama de secuencias

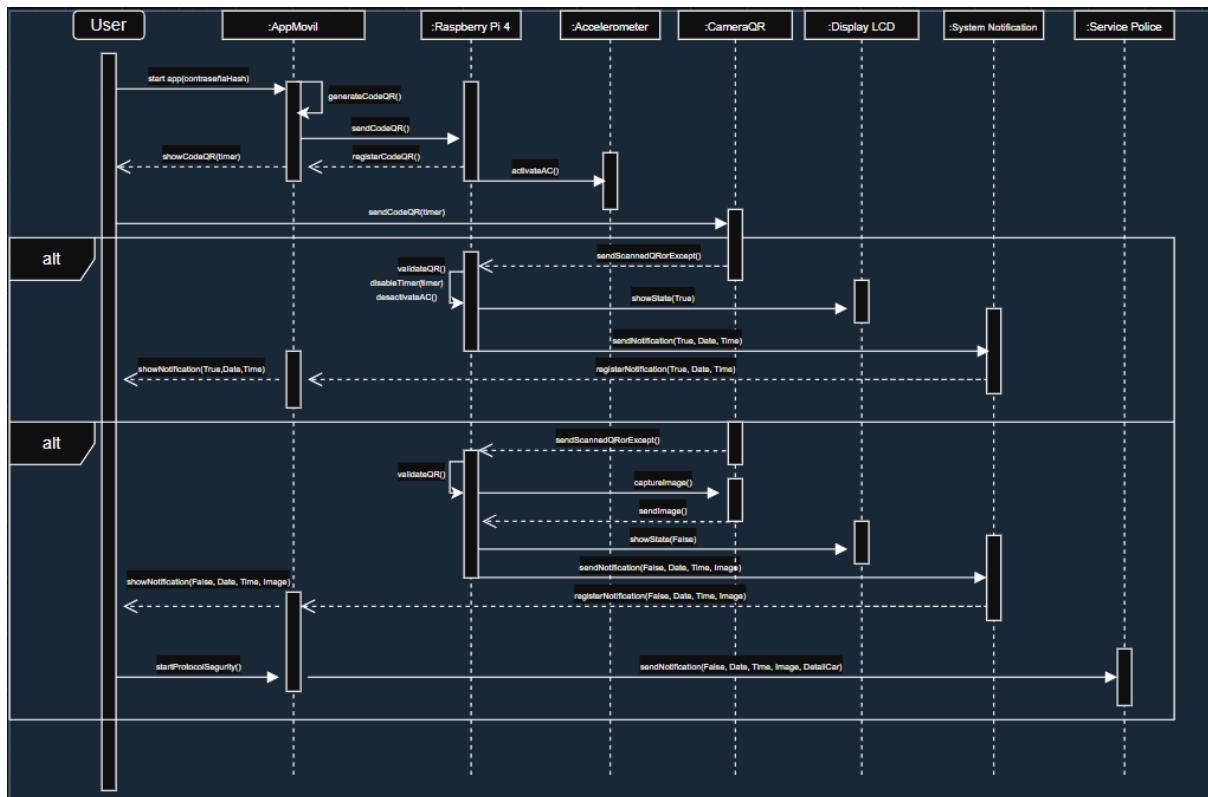


Figura 7: Diagrama de Secuencias



### 5.1.7 Descripción de la arquitectura con respecto a los modelos

- **Modelo de caso de uso general:**

El diagrama de caso de uso general demuestra cómo la arquitectura del sistema integra los distintos componentes físicos y lógicos. Es así como se representan los sensores y actuadores del vehículo en este caso, acelerómetro, cámara y pantalla LCD, los cuales interactúan directamente con la Raspberry Pi como sistema central. Estos dispositivos permiten ejecutar funcionalidades como la detección de aceleración, la captura de QR, la validación del código y la alerta visual mediante display en base a la validación.

Por otro lado se ubican los actores externos, la aplicación móvil, utilizada por el dueño para autenticarse, recibir notificaciones y confirmar un posible robo, entre otras funcionalidades, y el Sistema de la Autoridad, que recibe el reporte cuando el usuario confirma la alerta. Esto evidencia la comunicación



entre el componente físico del sistema central y los sistemas externos mediante la app móvil.



- **Modelo de diagrama de clases:**

En el diagrama de clases se representa la estructura interna del sistema según la arquitectura propuesta. Las clases principales se dividen entre los elementos del vehículo (Raspberry, sensores y display) y los componentes externos como el Usuario y la Aplicación Móvil. La clase Raspberry aparece como el centro de la arquitectura, ya que administra la cámara (SensorQR), el acelerómetro y el display LCD, tal como ocurre en el funcionamiento real del sistema.

La Aplicación Móvil se relaciona directamente con el Usuario y tiene métodos para mostrar alertas, ver el historial y enviar reporte a Carabineros. Esto refleja la parte lógica de la arquitectura donde la app sirve como puente entre el dueño y la Raspberry Pi.

La clase Vehículo contiene los datos del auto y se asocia al Usuario, mientras que la clase Alerta representa la información que se genera cuando ocurre un movimiento de aceleración. La relación entre Alerta, Vehículo y Usuario sigue el flujo real del sistema, la Raspberry detecta un movimiento en base al sensor del acelerómetro, genera una alerta y la app del usuario decide si confirmar o no.

- **Modelo de diagrama de secuencia:**

El diagrama de secuencia representa cómo la arquitectura funciona en los dos escenarios principales del sistema. Primero se muestra el inicio, donde la Raspberry Pi deja los sensores activos, el acelerómetro comienza a detectar movimiento y la cámara queda lista para capturar el código QR.

El primer caso corresponde al ingreso normal del dueño. El acelerómetro detecta la aceleración del vehículo, lo que activa un temporizador mientras el propietario muestra su código QR a la cámara. La Raspberry valida el QR y, si es correcto, la alerta se desactiva y el display LCD muestra el acceso autorizado. Aquí se observa la coordinación entre la aplicación móvil, la Raspberry Pi, el acelerómetro y la cámara.

El segundo caso representa el escenario de intrusión. El acelerómetro detecta el movimiento igual que antes, pero como no se presenta un QR válido, la Raspberry activa la cámara para capturar una imagen del

sospechoso y envía una notificación al dueño mediante la aplicación móvil. Si el usuario confirma que se trata de un robo, la app envía los datos al sistema de la autoridad. Esto muestra cómo la arquitectura reacciona automáticamente usando los sensores, la Raspberry Pi y la app, enlazándose finalmente con un servicio externo.

## 5.2 Herramientas y técnicas

### 5.2.1 Herramientas:

Durante el desarrollo del proyecto se utilizaron distintas herramientas tanto de software como de hardware:

- **Visual Studio Code:** editor principal para la programación y organización del código.
- **Redmine:** gestión de tareas y seguimiento del avance del proyecto.
- **Google Document:** Elaboración del documento y bitácora.
- **Draw.io:** creación de los diagramas (casos de uso, clases, secuencia, contexto).
- **Canva:** diseño de interfaz de usuario y elementos visuales del informe y presentación.
- **Sensores y módulos físicos:** acelerómetro, cámara, display LCD y demás componentes utilizados en la Raspberry.
- **Sistema Operativo Raspberry Pi OS:** plataforma donde se ejecutará la lógica del sistema.

✓

### 5.2.1 Técnicas utilizadas:

- **Dividir para conquistar:** Se separaron los módulos del sistema para facilitar el desarrollo.
- **Iteración incremental:** Cada parte del proyecto se fue avanzando en etapas, revisando antes de pasar a la siguiente.
- **Validación por escenarios:** Se analizaron los dos flujos principales (dueño e intruso) para comprobar coherencia.
- **Prototipado temprano:** Se diseñaron los modelos antes de pasar a la implementación real.
- **Modularización:** Cada componente (sensores, cámara, app móvil, alertas) se trató como módulo independiente.
- **Pruebas por componente:** Cada módulo se considerará individualmente en la integración (acelerómetro, QR, LCD).



## 6. Conclusiones

Esta primera fase se ha completado exitosamente con la definición integral de la propuesta del proyecto. Se estableció la problemática central y se diseñó la solución tecnológica, que consiste en un sistema de detección inteligente para anticipar y prevenir el robo vehicular. Paralelamente, se consolidaron los cimientos de la gestión del proyecto, lo que incluyó la asignación formal de roles (Jefe de Proyecto, Programadores, Diseñador, Ensamblador y Documentador), la estimación de costos y tiempos, y la determinación de los objetivos. La conclusión de esta etapa garantiza que el proyecto cuenta con la estructura organizativa y el marco técnico necesarios para abordar las siguientes fases de ejecución y desarrollo.

La segunda fase se centró exitosamente en la validación técnica y la estructuración del diseño del sistema. Se llevó a cabo una exhaustiva investigación sobre la compatibilidad y ejecución de los componentes centrales, incluyendo la Raspberry Pi 4, los sensores (cámara y acelerómetro), y los lenguajes de programación requeridos (Python, Kotlin, C#). Esto permitió asegurar la base tecnológica necesaria para la implementación. Además, se definieron los modelos de diseño esenciales para comunicar la solución, como el diagrama de casos de uso, el diagrama de clases y los diagramas de secuencia, complementados por el boceto de la Interfaz Gráfica de Usuario (IGU) de la aplicación móvil, elemento clave para la recepción de notificaciones de alerta. Finalmente, la creación formal de los

## Proyecto II

requerimientos funcionales y no funcionales del proyecto garantiza una guía precisa para el desarrollo del software y la verificación de la calidad del producto final.



Buen informe, con algunos detalles, revisen los diagramas y los sentidos de las flechas  
En general si se ve consistente su desarrollo, su posters es muy general .. rehacer

## 7. Referencias

### Referencias de Hardware y Componentes

- Raspberry Pi 4  
<https://www.raspberrypi.com/documentation/computers/raspberry-pi.html>
- Página donde se compró el sensor acelerómetro  
<https://www.mechatronicstore.cl/sensor-acelerometro-y-giroscopio-gy-6500-mpu-6500/>
- Grove - LCD RGB  
[https://wiki.seeedstudio.com/es/Grove-LCD\\_RGB\\_Backlight/](https://wiki.seeedstudio.com/es/Grove-LCD_RGB_Backlight/)
- Sensores con Raspberry PI 4  
<https://www.circuitbasics.com/what-is-an-accelerometer/>

### Referencias de Software y Desarrollo

- Meta Quest 3  
<https://www.meta.com/es-es/help/quest/509273027107091/>
- Software del Raspberry PI 4  
<https://www.raspberrypi.com/documentation/computers/getting-started.html>
- Unity  
<https://docs.unity3d.com/es/530/Manual/UnityManual.html>

Referencias fuera de norma IEEE Ingeniare