



# **SCAV**

## **CAPSTONE\_006V**

### **Informe avances**

### **Proyecto**

**Integrantes**

Nibaldo Quezada  
Cristian Arroyo  
Nicolas Peña

**Docente**

Fabián Alcántara

## Contenido

### Contenido

Introducción .....	3
Descripción del proyecto APT .....	4
Requerimientos de Alto Nivel: .....	4
Historias de Usuario y Product Backlog - Proyecto TechApps SCAV .....	6
Objetivos Específicos - Proyecto TechApps SCAV .....	8
Diagrama de arquitectura .....	10
Planificación .....	11
Avances del proyecto .....	12
Ajustes del proyecto.....	13
Metodología: Corrección hacia Scrum .....	13
Cronograma del Proyecto: Corrección de Sprints .....	14
Desarrollo de Aplicaciones: División entre Aplicación Administrativa y de Residentes .....	15
Implementación de Servicios en la Nube y Migración de la Base de Datos: .....	16
Uso de PostgreSQL para la Gestión de la Base de Datos .....	17
Evidencias de avance.....	18
Módulo Lector de patentes:.....	18
Módulo API Rest:.....	21
Módulo Flutter: .....	22
Modelo Entidad Relación: .....	27
Proyecciones Futuras del Proyecto TechApps - SCAV.....	28

## Introducción

El proyecto TechApps - SCAV tiene como objetivo desarrollar un sistema integral de control de acceso vehicular para el condominio Vista Parque 1 y 2. Esta iniciativa surge a partir de las dificultades que enfrenta el condominio en la gestión manual del acceso vehicular, lo que ocasiona congestión en los puntos de entrada y salida, además de vulnerabilidades de seguridad ante posibles robos. El sistema propuesto permitirá autorizar, monitorear y restringir el acceso de vehículos de manera automática, empleando tecnologías como reconocimiento de patentes, bases de datos, y aplicaciones móviles. Este proyecto permitirá a los estudiantes aplicar competencias clave de su perfil profesional, como la gestión de proyectos, el desarrollo de software y la implementación de soluciones informáticas para problemas reales, asegurando una experiencia práctica y acorde a los estándares de la industria.

## Descripción del proyecto APT

El objetivo del proyecto es desarrollar un sistema que permita mejorar la gestión y seguridad del acceso vehicular en el condominio. El sistema deberá registrar, autorizar y monitorear el ingreso y salida de vehículos, asegurando un control eficiente y seguro.

### Requerimientos de Alto Nivel:

Uso Administradores y Personal de Seguridad:

- Manejo de Registro de Residentes:
  - Registro y actualización de la información vehicular y personal de los residentes.
  - Administración de permisos de acceso para los vehículos registrados de cada residente.
- Manejo de Registro de Visitas:
  - Registro anticipado de visitas por parte de los residentes o por personal administrativo.
  - Autorización o denegación de acceso a las visitas, basada en políticas definidas por el recinto.
  - Notificación automática a los residentes sobre la llegada de una visita.
- Autorización de Vehículos Comerciales:
  - Registro de solicitudes de acceso para vehículos comerciales (ej. entregas de encomiendas, servicios de comida, servicios de transporte).
  - Revisión, aprobación o rechazo de dichas solicitudes por parte del personal autorizado.
- Ingreso de Vehículos Emergencia:
  - El conserje autoriza la entrada de forma manual en caso de vehículos que correspondan a Bomberos, ambulancia, policiales.
- Monitoreo y Reportes de Acceso:
  - Monitoreo en tiempo real de entradas y salidas de vehículos, mediante bitácora y reportería.
  - Generación de reportes de acceso diarios, semanales o mensuales, que incluyan datos sobre todos los ingresos, con filtros por tipo de vehículo, residente, visita, etc.
- Integración con API Externa para Validación Vehicular:
  - Validación automática de los datos vehiculares (marca, modelo, propietario) mediante una API externa, asegurando que solo vehículos registrados puedan acceder.
- Aplicación automática de multas:

- El sistema deberá ser capaz de identificar cuando una visita permanece por más de las horas autorizadas (Definido por el condominio), y aplicar la multa correspondiente.

#### Uso Residentes:

- Solicitud de Acceso para Visitas:
  - App móvil donde los residentes pueden registrar visitas anticipadamente, proporcionando la información del vehículo.
  - Recepción de notificaciones y actualizaciones sobre el estado de la visita (aprobada, rechazada).
- Solicitud de Acceso para Servicios Comerciales:
  - Validación por parte del residente receptor del servicio, si se requiere.
- Recepción de Notificaciones y Alertas:
  - Envío de notificaciones automáticas a los residentes cuando una visita registrada llega al recinto.
  - Actualizaciones periódicas sobre políticas de acceso y seguridad del recinto.
- Consulta de Historial de Accesos:
  - Acceso a un historial personal de entradas y salidas, tanto de los vehículos del residente como de las visitas autorizadas, y servicios externos, disponible a través de la app.

#### Arquitectura del Sistema:

- Lector de patente: Python OpenCV
- Base de Datos: PostgreSQL
- Api Rest: Spring Boot
- App Móvil: Flutter
- Api Externa: Auth ApiKey
- Nube: GCP (Google Cloud Platform)

Este sistema permitirá una gestión eficaz del acceso vehicular, mejorando la seguridad y el control en recintos residenciales, y garantizando una experiencia fluida tanto para los residentes como para los visitantes y proveedores.

# Historias de Usuario y Product Backlog - Proyecto TechApps SCAV

## 1. Historias de Usuario

Las historias de usuario son esenciales para definir los requisitos del sistema y asegurar que las funcionalidades desarrolladas se alineen con las necesidades de los usuarios finales. A continuación, se detallan las historias de usuario priorizadas para la aplicación móvil, enfocadas en la mejora del acceso vehicular en el condominio Vista Parque.

### 01. Registro de Vehículos de Residentes

- **Descripción:** Como **residente**, quiero que mi vehículo esté registrado y pueda ingresar **automáticamente** al condominio para evitar la congestión en la entrada durante las horas punta.
- **Prioridad: Alta**
- **Justificación:** Esta funcionalidad es crítica para agilizar el flujo vehicular y reducir los tiempos de espera en la entrada, mejorando la experiencia diaria de los residentes.

### 02. Registro de Vehículos de Visitas

- **Descripción:** Como **residente**, quiero poder registrar visitas anticipadamente para que puedan ingresar **sin demoras**, en cualquier horario.
- **Prioridad: Alta**
- **Justificación:** Facilitar el acceso de visitas previamente autorizadas mejora la seguridad y eficiencia, permitiendo a los residentes gestionar sus visitas de forma autónoma.

### 03. Historial de Visitas

- **Descripción:** Como **residente**, quiero poder consultar el historial de mis entradas y salidas, así como de las visitas al condominio, para controlar mis movimientos vehiculares y **evitar multas**.
- **Prioridad: Media**
- **Justificación:** Esta función permite a los residentes tener un mayor control sobre el acceso de sus vehículos y visitas, contribuyendo a la transparencia y a la prevención de infracciones.

## 2. Product Backlog

El Product Backlog define las funcionalidades y módulos clave que el sistema debe incluir para optimizar el control de acceso vehicular y la seguridad en el condominio Vista Parque. A continuación, se describen las funcionalidades priorizadas para el equipo de desarrollo.

### 01. Monitoreo en Tiempo Real

- **Descripción:** Como **personal de seguridad (Portería)**, quiero monitorear en tiempo real el flujo vehicular en la entrada para **agilizar el acceso** y atender incidentes en situaciones de emergencia.

- **Prioridad: Alta**
- **Justificación:** Esta funcionalidad es esencial para que el personal de seguridad tenga una visión clara del acceso vehicular y pueda tomar decisiones rápidas en caso de emergencias.

## 02. Gestor de Multas

- **Descripción:** Como **personal de seguridad (Portería)**, quiero que el sistema gestione y registre automáticamente las **multas** por visitas que exceden el tiempo autorizado.
- **Prioridad: Media**
- **Justificación:** Automatizar el control de multas ayuda a reducir la carga administrativa y asegura que las reglas de acceso sean cumplidas.

## 03. Consulta de Vehículos Comerciales

- **Descripción:** Como **personal de seguridad (Portería)**, quiero poder consultar rápidamente el acceso de **vehículos comerciales** (Uber, correos, pedidos) para darles acceso de manera rápida al condominio.
- **Prioridad: Media**
- **Justificación:** Optimiza el acceso de vehículos de servicio, mejorando la eficiencia sin comprometer la seguridad del condominio.

## 04. Control de Acceso para Emergencias

- **Descripción:** Como **personal de seguridad (Portería)**, quiero tener el control total del acceso en situaciones de emergencia para **garantizar la seguridad** de los residentes y el condominio.
- **Prioridad: Alta**
- **Justificación:** Esta funcionalidad es crítica para asegurar un **acceso controlado** durante emergencias, evitando que personas no autorizadas ingresen en momentos de crisis.

El desarrollo de las funcionalidades descritas en las historias de usuario y el backlog tiene un enfoque claro en mejorar la **eficiencia del acceso vehicular**, garantizar la **seguridad de los residentes** y facilitar la labor del **personal de seguridad**. Al priorizar las necesidades tanto de los residentes como del personal administrativo, se busca crear un sistema integral y escalable que responda a los desafíos del condominio Vista Parque.

Este informe servirá como guía para el equipo de desarrollo durante los siguientes sprints, asegurando que las funcionalidades más críticas sean implementadas de manera eficiente y alineada con las necesidades del cliente.

# Objetivos Específicos - Proyecto TechApps SCAV

## 1. Desarrollo del Lector de Patentes

- **Descripción:** Implementar un **lector automático de patentes** utilizando **Python** y una cámara, que capture imágenes en tiempo real para verificar el ingreso de vehículos registrados. El sistema se conecta al backend en la nube para validar la información.
- **Motivo del Cambio:** Originalmente, este módulo funcionaba en un entorno local. Ahora, se ha integrado con la **API REST en la nube** para validar los datos de las patentes de manera más eficiente.
- **Estado Actual:** El lector de patentes está completado y optimizado para funcionar con el servicio en la nube, utilizando **Google Kubernetes Engine (GKE)** para garantizar la escalabilidad.

## 2. Modelamiento de la Base de Datos

- **Descripción:** Diseñar y gestionar una **base de datos** en **Neon Tech PostgreSQL** que almacene los registros de residentes, visitas y vehículos comerciales, permitiendo la consulta y actualización en tiempo real desde la aplicación móvil y la API REST.
- **Motivo del Cambio:** Anteriormente, la base de datos se ejecutaba en un entorno local (localhost). Se ha migrado a un **entorno en la nube** para mejorar su **accesibilidad, disponibilidad y seguridad**.
- **Estado Actual:** La base de datos está totalmente funcional en Neon Tech PostgreSQL, integrándose con el backend mediante la API REST.

## 3. Desarrollo de la API REST (Spring Boot)

- **Descripción:** Crear una **API REST** en **Spring Boot**, que gestione los permisos de acceso vehicular y se comunique con la base de datos en la nube para validar la información vehicular. La API está desplegada en **Google Kubernetes Engine (GKE)** para mejorar su escalabilidad y disponibilidad.
- **Motivo del Cambio:** Originalmente, la API se desarrollaba en un entorno local. Ahora, está desplegada en **GCP** con **Kubernetes**, lo que permite un acceso seguro y escalable, eliminando la necesidad de utilizar localhost.
- **Estado Actual:** El servicio de autenticación (msautenticar) y otros endpoints de la API están activos en la nube y configurados para manejar un alto volumen de solicitudes.

## 4. Desarrollo de la Aplicación Móvil (Flutter)

- **Descripción:** Crear una **aplicación móvil** en **Flutter** que permita a los residentes registrar visitas, recibir notificaciones en tiempo real y gestionar su historial de accesos.
- **Motivo del Cambio:** En lugar de desarrollar aplicaciones separadas para administración y residentes, se ha unificado en **una sola aplicación móvil** que adapta las funcionalidades en función del rol del usuario. La app ahora se conecta directamente al backend en la nube para mejorar la eficiencia y la accesibilidad.



- **Estado Actual:** El desarrollo está en curso, con un enfoque en la integración con la API REST para manejar el registro y gestión de visitas en tiempo real.

## 5. Monitoreo en Tiempo Real

- **Descripción:** Desplegar un sistema de monitoreo que permita al personal de seguridad ver en tiempo real el flujo de vehículos que ingresan y salen del condominio, generando informes periódicos basados en los datos recopilados.
- **Motivo del Cambio:** Gracias a la migración a la nube y al uso de **Kubernetes**, ahora se puede escalar el sistema de monitoreo automáticamente para manejar picos de tráfico, asegurando un acceso sin interrupciones.
- **Estado Actual:** El módulo de monitoreo se está integrando con los microservicios desplegados en GKE para ofrecer una visualización en tiempo real.

## 6. Seguridad y Protección de Datos

- **Descripción:** Garantizar que el sistema cumpla con **estándares de seguridad**, protegiendo los datos de los usuarios y la **confidencialidad de la información** vehicular mediante la implementación de políticas de seguridad en **GCP**.
- **Motivo del Cambio:** Con la migración a **Google Cloud Platform**, se han implementado medidas de seguridad avanzadas, incluyendo **encriptación de datos en tránsito y en reposo**, así como la gestión de accesos mediante políticas IAM (Identity and Access Management).
- **Estado Actual:** La seguridad del sistema está configurada para cumplir con los **estándares de la industria**, asegurando que solo usuarios autorizados accedan a la información.

Los ajustes realizados en la arquitectura del proyecto han permitido una **mayor escalabilidad, disponibilidad y seguridad** del sistema **TechApps SCAV**. La migración a la nube y el uso de **Kubernetes** para el despliegue del backend aseguran un rendimiento óptimo y una fácil administración, alineándose con los objetivos de optimizar el acceso vehicular en el condominio Vista Parque.

Estos cambios mejoran significativamente la eficiencia operativa del sistema, lo que permite a los residentes y al personal de seguridad acceder a las funcionalidades críticas de manera fluida y segura.



[illegible]

## Avances del proyecto

Actualmente nos encontramos en la fase de desarrollo de las aplicaciones móviles en flutter, a continuación, se adjunta un cuadro resumen con el estado de cada tarea realizada durante el proyecto.

Actividades y Estado de Avance Materiales:

Actividad	Estado	Dificultades	Materiales/Recursos Físicos	Lenguajes/Entorno
Lector de Patentes	Completado	Dificultades para hacer pruebas en vivo con la cámara en la calle debido a condiciones climáticas y tráfico	Cámara de video, PC	Python, OpenCV
Diseño de Base de Datos	Completado	N/A	PC, Neon Tech PostgreSQL	PostgreSQL
API REST	En curso	N/A	PC, GCP (Google Cloud Platform) API server	Java, Spring Boot
Aplicación Móvil en Flutter	En curso	N/A	PC, smartphone para pruebas	Flutter, Dart
Pruebas de Integración	No iniciado	Coordinación para pruebas con cámara en tiempo real en un entorno controlado	Cámara de video, PC, red local para la integración	Python, Java, SQL, Flutter

## Ajustes del proyecto

### Metodología: Corrección hacia Scrum

Desde el inicio del proyecto TechApps - SCAV, la metodología Scrum fue la propuesta y establecida como la guía para gestionar el desarrollo. Sin embargo, durante las primeras fases de implementación, por un error del equipo, las actividades comenzaron a gestionarse bajo un enfoque más tradicional. Este desvío se debió a la organización de tareas como bloques independientes, con fechas de entrega fijas y poca interacción entre los módulos en desarrollo (lector de patentes, API REST, base de datos y aplicación móvil).

Al identificar este error, se realizó un ajuste inmediato para volver al enfoque ágil Scrum, lo cual permitió:

1. Recuperar la planificación iterativa e incremental: Se organizaron los trabajos en los sprints, lo que garantiza mayor flexibilidad para incorporar cambios según las necesidades.
2. Mejor coordinación y colaboración: Las reuniones diarias (daily meetings) se reintrodujeron para alinear al equipo y resolver bloqueos rápidamente.
3. Entregas parciales frecuentes: Se retomaron las entregas incrementales en cada sprint, lo que permitió presentar avances funcionales, como el lector de patentes y la integración de la API con la base de datos.
4. Retroalimentación continua: Se fomenta la interacción constante con los involucrados para garantizar que cada módulo se desarrollará conforme a las expectativas y requerimientos del proyecto.

Este ajuste ha sido crucial para recuperar la dinámica ágil que se había perdido. El equipo se alineó nuevamente con la metodología Scrum, permitiendo una mejor sincronización entre los módulos, optimizando los tiempos de desarrollo y asegurando que el proyecto avance según lo planeado.

## Cronograma del Proyecto: Corrección de Sprints

Inicialmente, el proyecto TechApps - SCAV se organizó en un número diferente de sprints, lo cual no permitía una planificación adecuada para la implementación de todos los módulos. Tras una revisión del cronograma, se realizó una corrección en la cantidad y duración de los sprints, con el fin de optimizar el desarrollo y cumplir con los objetivos planteados.

### Nuevo Cronograma:

- Total de Sprints: 9 Sprints.
- Duración de cada Sprint: 2 semanas cada uno.
- Distribución por Módulos:
  1. Gestión de Proyecto y Backlog: Sprint 1 (S1) y (S2), Sprint 2 (S3) y (S4).
  2. Desarrollo del Sistema de Reconocimiento de Patentes: Sprint 3 (S5) y (S6).
  3. Desarrollo del Modelo de Base de Datos: Sprint 4 (S7) y (S7).
  4. Implementación de API REST: Sprint 5 (S9) y (S10).
  5. Aplicación Móvil en Flutter: Se ajustó a dos sprints completos: Sprint 6 (S11) y (S11), Sprint 7 (S13) y (S14).
  6. Pruebas de Calidad e Integración: Sprint 8 (S15) y (S16).
  7. Marcha Blanca y Capacitación: Sprint 9 (S17) y (S18).

### Impacto del Ajuste:

- Redistribución del Tiempo: Las fechas se ajustaron para que los módulos más críticos tengan más tiempo asignado, como el desarrollo de la aplicación móvil en Flutter, que se desarrollará a lo largo de 2 sprints completos.
- Optimización del Flujo: El resto de los módulos se desarrollarán en sprints únicos para evitar la fragmentación del trabajo.

## Desarrollo de Aplicaciones: División entre Aplicación Administrativa y de Residentes

### Unificación en una Única App con Roles Administrativos y Residentes

Durante la fase inicial de planificación del proyecto TechApps - SCAV, se propuso desarrollar dos aplicaciones independientes: una para administradores y otra para residentes. Sin embargo, a medida que avanzamos en el desarrollo y tras una revisión del flujo y la usabilidad, identificamos que la división generaba redundancia y aumentaba la carga de mantenimiento del sistema.

**Decisión de Cambio: Unificación en una Única Aplicación con Roles Dinámicos**

En lugar de gestionar dos aplicaciones separadas, decidimos integrar todas las funcionalidades en una única aplicación móvil. La nueva estrategia simplifica el desarrollo, facilita el mantenimiento y mejora la experiencia del usuario al unificar las operaciones en una sola plataforma.

Cómo funciona la nueva estructura:

- Al iniciar sesión, la aplicación detecta el tipo de cuenta (Administrador o Residente) basado en las credenciales del usuario.
- Dependiendo del rol del usuario, se derivan a diferentes pantallas y funcionalidades:
  - Administradores:
    - Gestión de residentes, visitas y vehículos.
    - Acceso a reportes y monitoreo en tiempo real de accesos vehiculares.
    - Administración de permisos de entrada y salida.
    - Automatización y gestión de multas.
  - Residentes:
    - Registro anticipado de visitas.
    - Notificaciones en tiempo real sobre la llegada de visitas.
    - Visualización del historial de accesos personales.

### Ventajas del Enfoque Unificado

- A. Reducción de Complejidad: Al unificar ambas aplicaciones en una sola, se eliminan redundancias y se simplifica la administración del sistema.
- B. Flexibilidad en el Uso de Roles: La aplicación ahora se adapta dinámicamente al rol del usuario, proporcionando un flujo intuitivo y centralizado.
- C. Mantenimiento Simplificado: Facilita la gestión de actualizaciones, ya que cualquier mejora se refleja en una sola app.
- D. Experiencia del Usuario Mejorada: Los usuarios (administradores y residentes) pueden acceder a todas sus funcionalidades desde una única plataforma sin tener que cambiar de aplicación.

## Implementación de Servicios en la Nube y Migración de la Base de Datos:

Contexto del Cambio: Durante las fases iniciales del proyecto TechApps - SCAV, el sistema estaba configurado para operar en un entorno local, lo que generaba ciertas limitaciones en cuanto al acceso y escalabilidad. Para mejorar el rendimiento y garantizar un acceso más robusto y seguro, se decidió migrar la aplicación y la base de datos a un entorno en la nube utilizando Google Cloud Platform (GCP).

Detalles de la Implementación:

1. Migración de la Base de Datos:
  - La base de datos originalmente desarrollada en SQL Server se migro a cloud utilizando la herramienta NeonTech, la cual te brinda una instancia cloud en aws para alojar bases de datos de hasta 512mb de manera gratuita.
  - Esta migración permite un acceso más rápido, confiable y centralizado a la información de residentes, visitas y vehículos desde cualquier lugar.
  - Ventajas:
    - Elimina la dependencia de un entorno localhost, mejorando la disponibilidad y accesibilidad.
    - Aumenta la seguridad al aprovechar las políticas de seguridad y encriptación.
    - Facilita el escalado de la base de datos según la demanda del sistema.
2. Despliegue de todos los microservicios en GCP:
  - Los cuatro microservicios que componen SCAV de momento han sido desplegado en la nube usando Google Kubernetes Engine (GKE).
  - Esto permite un despliegue automatizado y escalable del sistema de autenticación, que es crucial para gestionar los accesos tanto de administradores como de residentes.
  - Ventajas:
    - Mejora la escalabilidad y disponibilidad del servicio de autenticación y gestión de vehículos, residentes y visitas.
    - Facilita la gestión de cargas en función del número de usuarios concurrentes.
    - Reduce el tiempo de inactividad al permitir actualizaciones en caliente sin afectar a los usuarios.

Impacto de estos Cambios en el Proyecto

1. Eliminación de Dependencias Locales:
  - Con la migración a la nube, se elimina la necesidad de ejecutar servicios en localhost, lo que simplifica la integración y despliegue de la aplicación en entornos de producción.



## Uso de PostgreSQL para la Gestión de la Base de Datos

Contexto del Cambio: Para optimizar el almacenamiento, la consulta y la escalabilidad de los datos del sistema, se tomó la decisión de implementar PostgreSQL como el motor de base de datos principal. Esto ha permitido un enfoque más moderno y eficiente para la gestión de datos del proyecto TechApps - SCAV.

- Detalles de la Implementación:
  - A. Migración a PostgreSQL en Neon Tech: La base de datos ha sido migrada a PostgreSQL, aprovechando las ventajas Neon Tech para asegurar un acceso rápido y seguro. PostgreSQL se seleccionó por su capacidad para gestionar grandes volúmenes de datos y su compatibilidad con consultas avanzadas y transacciones ACID.
    - Alta Disponibilidad: La configuración en la nube garantiza la disponibilidad continua del servicio, incluso en caso de fallos en el hardware.
    - Despliegue Continuo y Backup Automático: permite copias de seguridad automatizadas, lo que facilita la recuperación ante desastres y actualizaciones sin interrupciones.
    - Seguridad Mejorada: Soporta encriptación de datos en tránsito y en reposo, garantizando la confidencialidad de la información.

### Impacto de estos Cambios en el Proyecto

- Eliminación de Dependencias Locales:  
Con la migración a PostgreSQL en Cloud, se elimina la necesidad de ejecutar bases de datos en entornos locales, simplificando el despliegue y la integración en producción.
- Mejora en la Escalabilidad y Disponibilidad:  
El sistema ahora puede escalar dinámicamente según la carga, lo que es crucial para un proyecto que optimiza el acceso vehicular en un condominio de gran tamaño como Vista Parque.
- Facilita la Colaboración y el Desarrollo:  
Al centralizar la base de datos en la nube, todos los miembros del equipo pueden acceder al entorno de desarrollo y producción de manera remota, lo que agiliza las pruebas y el desarrollo colaborativo.

## Evidencias de avance

A continuación, se adjuntarán algunos prints de pantalla correspondientes a los avances de los distintos módulos desarrollados durante el proyecto.

### Módulo Lector de patentes:



Ilustración 1 - Implementación lector de patentes

A continuación, describiremos el funcionamiento del sistema de control de acceso vehicular desarrollado para optimizar la seguridad en recintos privados, tales como condominios, oficinas o estacionamientos. El sistema permite el acceso automatizado de vehículos autorizados mediante la detección de patentes (placas de vehículos), y registra tanto la entrada como la salida en una bitácora a través de una API. Adicionalmente, se integra un servomotor controlado por un Arduino para manejar una barrera que se eleva automáticamente al detectar una patente válida.

### Descripción General del Sistema

El sistema consta de varios componentes que interactúan entre sí:

1. **Cámara:** Captura en tiempo real las imágenes del vehículo al acercarse al punto de control.
2. **Lector de Patentes (Python):** Detecta la patente del vehículo a partir del flujo de video y realiza consultas a una API para verificar si el vehículo está registrado.
3. **API (Backend):** Se encarga de manejar la información sobre los vehículos autorizados y de registrar los accesos en una base de datos.

4. **Control de Barrera (Arduino):** Utiliza un servomotor para controlar la apertura y cierre de una barrera física, activada por comandos enviados desde el sistema Python.

## Funcionamiento del Sistema

### 1. Flujo de Entrada de Vehículos

El proceso de entrada al recinto sigue estos pasos:

1. **Captura de Imagen:** La cámara captura continuamente imágenes en tiempo real.
2. **Detección de la Patente:**
  - Utilizando la biblioteca OpenCV en Python, se detecta la presencia de una patente en el flujo de video.
  - El número de la patente se extrae utilizando técnicas de reconocimiento óptico de caracteres (OCR).
3. **Consulta a la API:**
  - El número de la patente detectada se envía a un endpoint específico (/api/v2/vehiculo/patente/{patente}) para verificar si el vehículo está registrado.
  - La API devuelve información sobre el vehículo, incluyendo su estado (residente, visita, etc.).
4. **Registro en la Bitácora:**
  - Si el vehículo está registrado, se crea un nuevo registro de entrada en la bitácora mediante una solicitud POST a la API (/api/v2/bitacora).
  - La bitácora almacena la hora de entrada y deja el campo de salida como null hasta que el vehículo salga.
5. **Activación de la Barrera:**
  - Si la patente es válida, se envía un comando al Arduino a través de la comunicación serial para que el servomotor eleve la barrera.

### 2. Flujo de Salida de Vehículos

El proceso de salida sigue una lógica similar, con algunas diferencias:

1. **Captura y Detección:**
  - La cámara detecta la patente del vehículo cuando intenta salir del recinto.
2. **Verificación y Actualización en la Bitácora:**

- El número de la patente se consulta en la API para obtener el ID del registro de entrada correspondiente.
- Se envía una solicitud PUT a la API (/api/v2/bitacora/salida/{vehiculoid}) para actualizar el campo fechaout con la hora de salida.

### 3. **Cálculo del Tiempo de Permanencia:**

- Una vez registrado el horario de salida, se calcula la duración total del tiempo de permanencia del vehículo en el recinto.

### 4. **Apertura de la Barrera para Salida:**

- Al confirmar que el registro ha sido actualizado correctamente, se envía un comando al Arduino para abrir la barrera y permitir la salida del vehículo.

## **Conexión con el Arduino para Control de Barrera**

El sistema utiliza un Arduino conectado a un servomotor que controla la barrera. El Arduino se comunica con el sistema Python mediante un puerto serial (COM6, en este caso). La integración se realiza de la siguiente forma:

### 1. **Inicialización:**

- El Arduino se configura para escuchar comandos desde el puerto serial.

### 2. **Comando de Apertura:**

- Cuando se detecta una patente válida, se envía el comando subir al Arduino, el cual activa el servomotor para elevar la barrera.

### 3. **Cierre Automático:**

- Después de un breve intervalo (5 segundos), el Arduino baja automáticamente la barrera.

## Módulo API Rest:

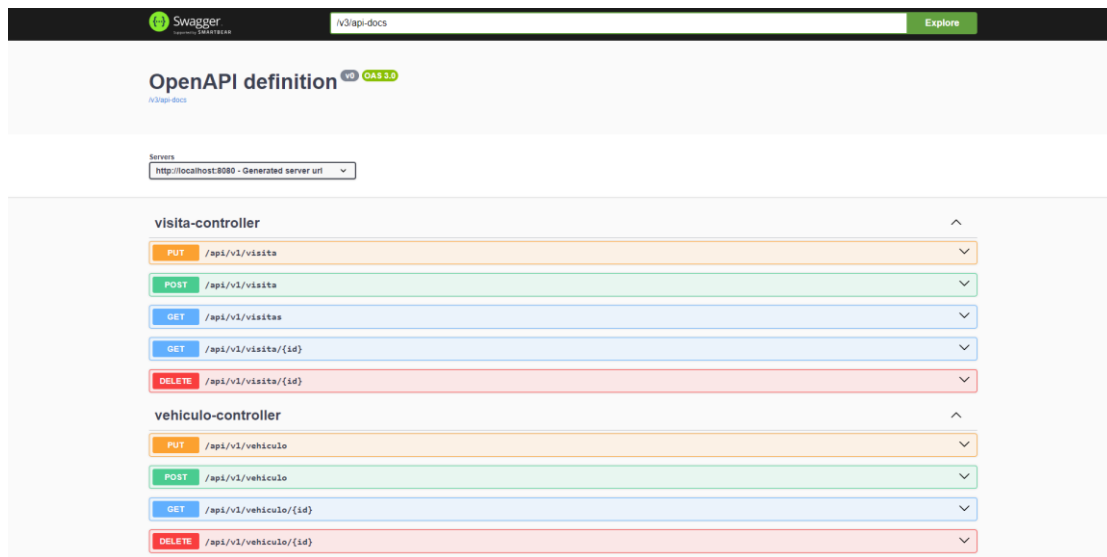
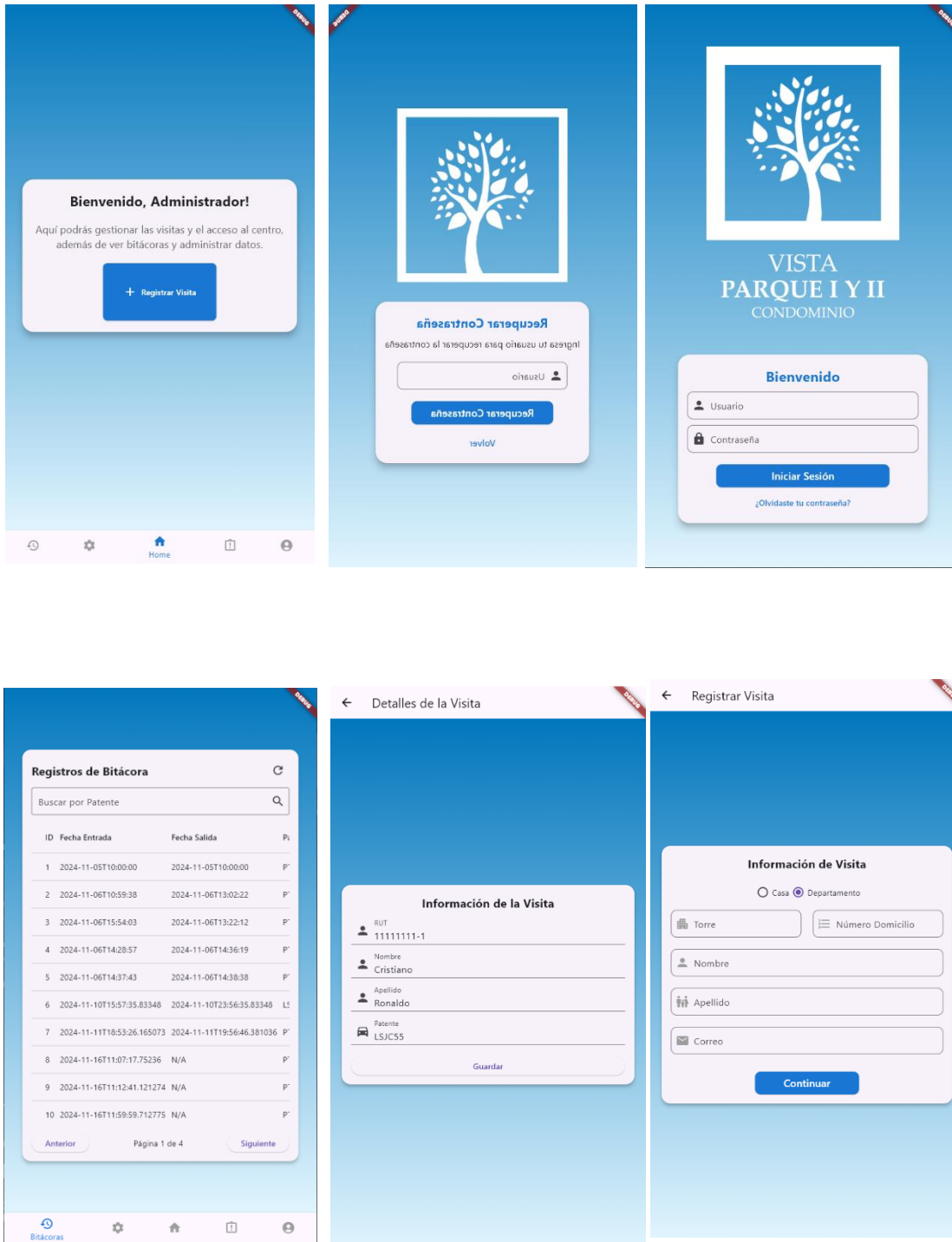


Ilustración 2 - Implementación API Rest

## Módulo Flutter:



← Administrar Residentes

Back

### Registrar Nuevo Residente

Nombre

Apellido

RUT













Correo

Torre

Departamento

Guardar Residente

### Lista de Residentes

<b>Ana Torres</b> RUT: 34567890-2	 
<b>Carlos Fuentes</b> RUT: 45678901-3	 
<b>Lucía Mendoza</b> RUT: 56789012-4	 
<b>Luis Martínez</b> RUT: 67890123-5	 
<b>Elena Rojas</b> RUT: 78901234-6	 
<b>Jorge Hernández</b>	 

### Mantenedores

-  Administrar Residentes
-  Administrar Visitas
-  Administrar Vehículos

Mantenedores

← Administrar Vehículos

### Registrar Nuevo Vehículo

Patente

Marca













Modelo

Color

Año

Guardar Vehículo

### Lista de Vehículos

<b>Honda Civic</b> Patente: DEF5678	 
<b>Ford Focus</b> Patente: GHI9012	 
<b>Chevrolet Cruze</b> Patente: JKL3456	 
<b>Nissan Sentra</b> Patente: MNO7890	 
<b>Toyota Corolla</b> Patente: PQL21	 
<b>HYUNDAI ACCENT</b> Patente: LSI55	 

← Administrar Visitas

### Registrar Nueva Visita

Nombre















Apellido

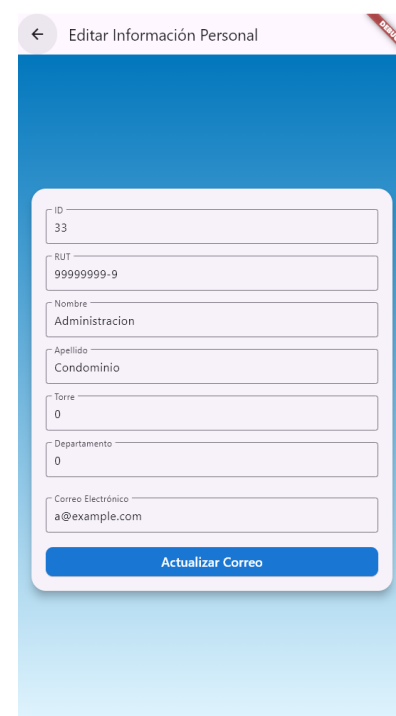
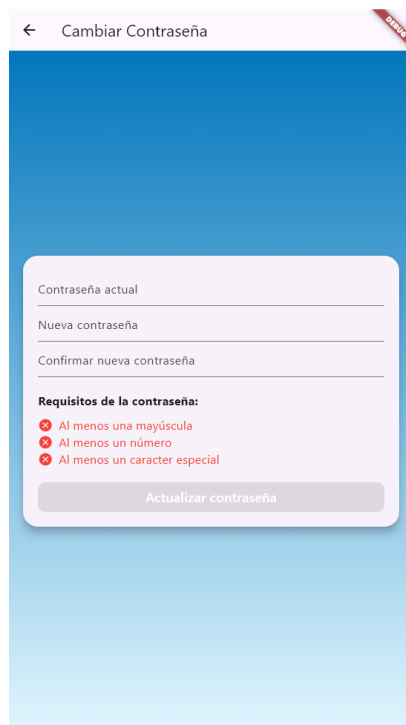
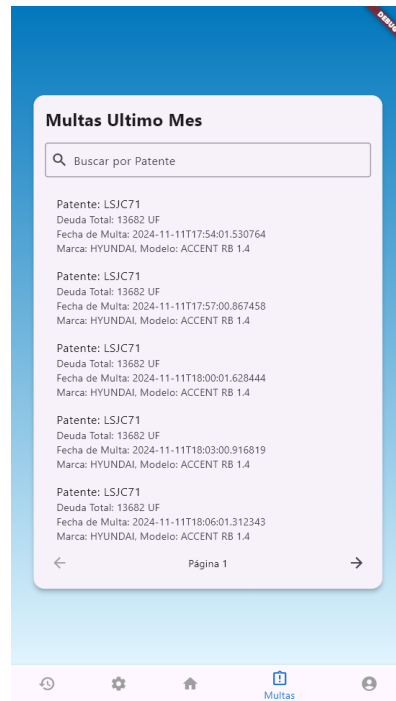
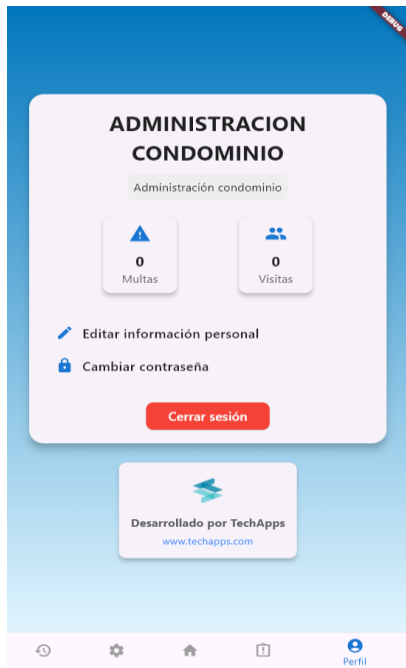
RUT

Motivo de Visita

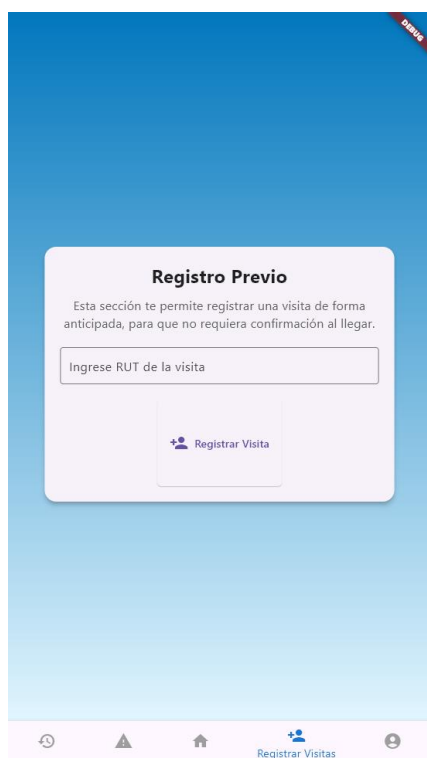
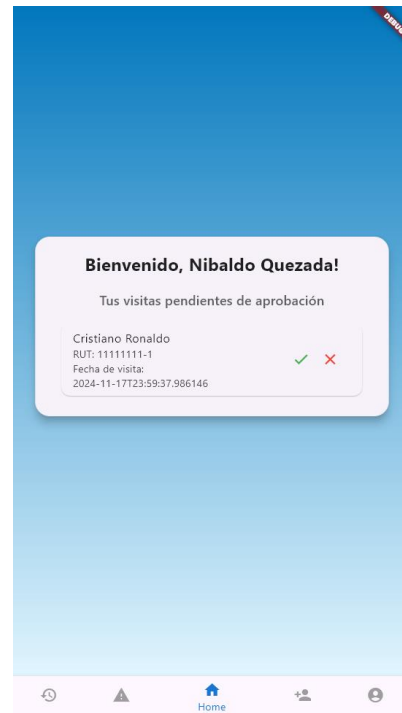
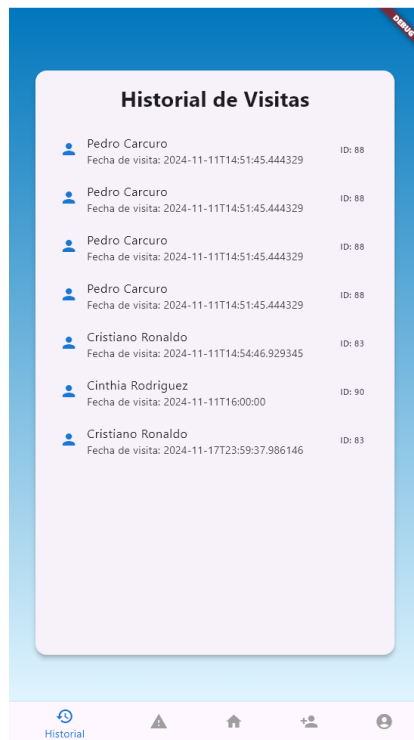
Guardar Visita

### Lista de Visitas

<b>Cristiano Ronaldo</b> RUT: 11111111-1	 
<b>Pedro Carcuro</b> RUT: 11111111-3	 
<b>Cinthia Rodríguez</b> RUT: 18976933-4	 
<b>Maria Garay</b> RUT: 10786226-9	 
<b>Diego Maradonna</b> RUT: 22222222-3	 
<b>Cristian Arroyo</b> RUT: 55555555-5	 
<b>Fabian Alcantara</b> RUT: 77777777-7	 







NIBALDO QUEZADA

Torre 1, Departamento 102

0

Multas

0

Visitas


✎

Editar información personal

🔒

Cambiar contraseña

Cerrar sesión



Desarrollado por TechApps

www.techapps.com

🕒

⚠️

🏠

👤

👤

Perfil

←

Registrar Visita

Visita Encontrada

Nombre

Cristiano

Apellido

Ronaldo

Patente del Vehículo

LSJC55

📅

Visita: 2024-11-18 12:02 AM

Guardar Visita

←

Cambiar Contraseña

Contraseña actual

Nueva contraseña

Confirmar nueva contraseña

Requisitos de la contraseña:

✖

Al menos una mayúscula

✖

Al menos un número

✖

Al menos un caracter especial

Actualizar contraseña

←

Editar Información Personal

ID

2

RUT

87654321-0

Nombre

Nibaldo

Apellido

Quezada

Torre

1

Departamento

102

Correo Electrónico

nib.quezada@duocuc.cl

Actualizar Correo

## Modelo Entidad Relación:

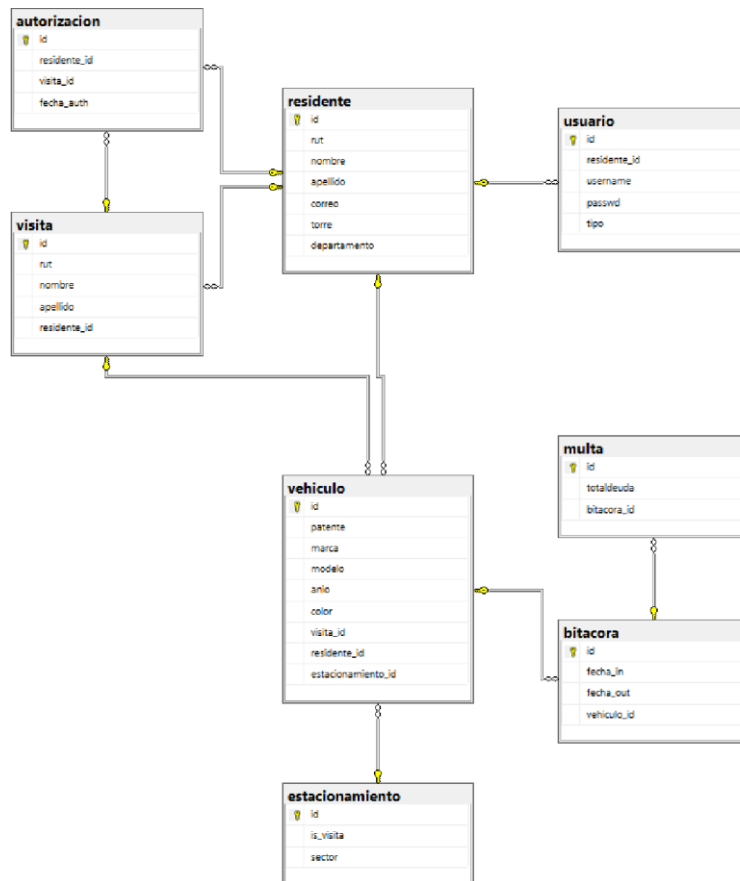


Ilustración 3 – MER

## Proyecciones Futuras del Proyecto TechApps - SCAV

### 1. Sistema de Pago para Multas

- **Descripción:** Se planifica desarrollar un **módulo de pago** que permita a los residentes **pagar multas** de forma rápida y segura a través de la aplicación móvil.
- **Características del Sistema:**
  - **Integración con Plataformas de Pago:** Permitir el uso de tarjetas de crédito/débito y otros métodos de pago electrónicos.
  - **Notificaciones Automáticas:** Envío de **recordatorios automáticos** a los residentes sobre multas pendientes y fechas límite para su pago.
  - **Historial de Pagos:** Los usuarios podrán ver un **historial de pagos** y descargas de recibos desde la aplicación.

### 2. Módulo de Gestión de Estacionamientos

- **Descripción:** Implementar un **módulo para la gestión de estacionamientos**, permitiendo a los residentes y visitantes **reservar y gestionar espacios de estacionamiento** dentro del condominio.
- **Características Clave:**
  - **Reservas en Tiempo Real:** Permitir a los residentes **reservar estacionamientos** desde la aplicación con antelación, evitando conflictos por espacio durante las horas pico.
  - **Monitoreo de Espacios Disponibles:** Utilizar sensores y datos en tiempo real para **monitorear la disponibilidad** de espacios de estacionamiento.
  - **Asignación de Espacios para Visitas:** Gestionar la asignación de **espacios específicos para visitas**, mejorando el control y seguridad del condominio.