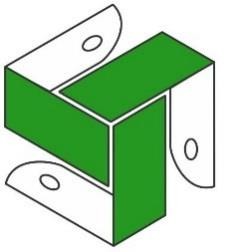
TECNOLOGICO DE ESTUDIOS SUPERIORES DE COACALCO



|  |
| --- |
| **TESCoPark QR** |
| Ingeniería en Sistemas Computacionales |
| Especialidad |
| Desarrollo de software |
| Presenta(n): |
| Landon Donovan Delint Alvidrez Christian Manuel Monroy Patiño  José Manuel Rodríguez Hernández |
| Grupo: 3812 |
| Coacalco de Berriozábal, 12/05/2025 |

# Introducción

Actualmente, el control de acceso vehicular en el estacionamiento del Tecnológico de Estudios Superiores de Coacalco (TESCo) se realiza mediante marbetes físicos, los cuales son susceptibles a extravíos, falsificaciones o mal uso. Este sistema no permite un monitoreo en tiempo real ni un registro eficiente de entradas y salidas, lo que genera desorganización, pérdida de tiempo y falta de control.

Con el objetivo de modernizar este proceso, surge el proyecto **TESCoPark QR**, una aplicación móvil que digitaliza el acceso al estacionamiento mediante códigos QR únicos vinculados a los datos del alumno y su vehículo. Además, permite a los guardias verificar el acceso desde sus dispositivos móviles y a los administradores monitorear la ocupación del estacionamiento. Esta solución no solo mejora la seguridad y eficiencia, sino que también proporciona una plataforma escalable para futuras integraciones tecnológicas como sensores IoT.

# Objetivos

# **Objetivo General:** Desarrollar una aplicación móvil que reemplace el sistema de marbetes físicos del estacionamiento del TESCo mediante la implementación de códigos QR, facilitando el control automatizado de acceso y la gestión de espacios disponibles en tiempo real.

# **Objetivos Específicos:**

# Diseñar una interfaz intuitiva para alumnos y guardias.

# Implementar la generación y escaneo de códigos QR seguros.

# Crear una base de datos centralizada en la nube con Firebase.

# Registrar automáticamente las entradas y salidas de los vehículos.

# Monitorear la disponibilidad del estacionamiento en tiempo real.

# Facilitar la creación de reportes para la administración del campus.

# Justificación

# El sistema actual basado en marbetes impresos no responde a las necesidades modernas de seguridad, agilidad y control. Es fácil de vulnerar y no proporciona una trazabilidad confiable de los accesos. Además, el registro manual de entradas y salidas es propenso a errores y consume tiempo valioso del personal de vigilancia.

# TESCoPark QR representa una solución innovadora y factible al transformar el proceso en una experiencia completamente digital. Al utilizar códigos QR y almacenamiento en la nube, se garantiza un control eficiente, seguro y automatizado. La app también permite escalar el proyecto a futuras versiones con reconocimiento facial, sensores de ocupación o integración con sistemas institucionales, alineándose con los principios de una universidad inteligente y tecnológicamente preparada.

# Descripción

La aplicación constará de dos módulos principales:

1. App para Alumnos:

* Registro de vehículos (placas, modelo, foto).
* Generación de un QR único vinculado a sus datos.
* Visualización de lugares disponibles en tiempo real.

1. App para Guardias:

* Escaneo de QR para verificar acceso.
* Asignación manual/automática de espacios.
* Actualización del estado de ocupación del estacionamiento.

Además, se incluirá un panel administrativo para monitorear el uso del

estacionamiento y generar reportes.

**Funcionalidades:**

**Para Alumnos:**

**Registro de vehículos:**

* Subir datos del auto/moto (placas, modelo, foto).
* Validación por parte de administradores (o mediante IA básica si se escala el

proyecto).

**Generación de código QR:**

* Estático (vinculado a matrícula y datos del vehículo).
* Dinámico (opcional, para mayor seguridad).

**Consulta de disponibilidad:**

* Mapa interactivo o lista de espacios libres/ocupados.
* Notificación si el estacionamiento está lleno.

**Historial de accesos:**

* Registro de entradas y salidas.

**Para Guardias:**

1. **Lector de QR:**

* Escaneo desde una tablet o celular.
* Visualización de datos del alumno y vehículo.

1. **Asignación de espacios:**

* Manual: El guardia indica qué lugar ocupa.
* Automática (avanzado): La app sugiere el primer lugar libre.

1. **Actualización de estado:**

* Marcar lugares como ocupados/libres.

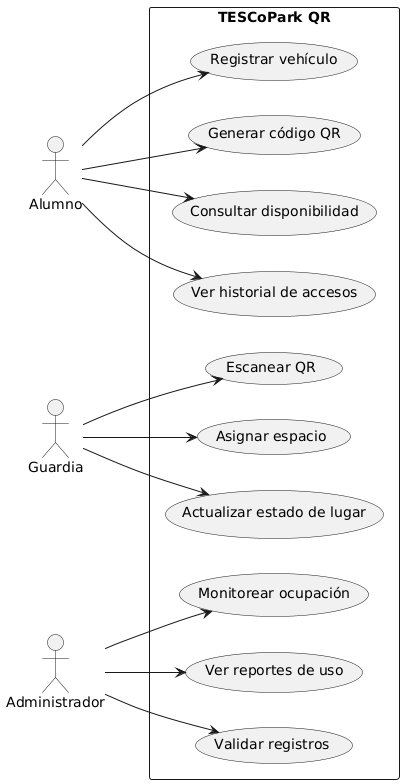
**Para Administradores:**

Panel de control:

* Reportes de ocupación (horas pico, días más congestionados).
* Detección de vehículos no registrados.

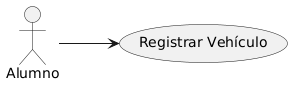
# Casos de uso

# **General**



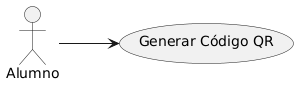
**Caso de Uso 1: Registrar Vehículo**

* **Actor principal:** Alumno
* **Descripción:** El alumno registra los datos de su vehículo, incluyendo placas, modelo y una fotografía.
* **Precondiciones:** El alumno debe haber iniciado sesión en la app.
* **Flujo principal:**
  1. El alumno accede a la opción "Registrar vehículo".
  2. Ingresa las placas, modelo y sube una foto del vehículo.
  3. La app guarda la información en Firebase.
  4. Se notifica al alumno que el vehículo fue registrado correctamente.
* **Flujos alternos:**
  1. 1a. Si los datos son inválidos o faltan campos, se muestra un mensaje de error.
* **Postcondiciones:** El vehículo queda registrado en la base de datos para su validación y uso posterior.

****

**Caso de Uso 2: Generar Código QR**

* **Actor principal:** Alumno
* **Descripción:** El alumno genera un código QR único vinculado a sus datos y vehículo.
* **Precondiciones:** El vehículo debe estar registrado y validado.
* **Flujo principal:**
  1. El alumno accede a la opción "Generar QR".
  2. La app genera un código QR con la matrícula, ID del vehículo y fecha.
  3. El código QR se muestra en pantalla y se guarda localmente.
* **Flujos alternos:**
  1. 2a. Si no hay vehículo validado, se muestra una advertencia.
* **Postcondiciones**: El código QR queda disponible para ser escaneado por los guardias.



### **Caso de Uso 3: Consultar Disponibilidad**

* **Actor principal:** Alumno
* **Descripción:** El alumno consulta los lugares disponibles del estacionamiento.
* **Precondiciones:** Debe tener sesión activa.
* **Flujo principal:**
  1. El alumno accede a la sección "Disponibilidad".
  2. Se carga el mapa de espacios libres/ocupados desde Firebase o Google Sheets.
  3. Se muestra visualmente la información (colores o lista).
* **Postcondiciones:** El alumno visualiza la disponibilidad actual del estacionamiento.



### **Caso de Uso 4: Ver Historial de Accesos**

* **Actor principal:** Alumno
* **Descripción:** Consulta el historial de entradas y salidas de su vehículo.
* **Precondiciones:** Debe haber registros en la base de datos.
* **Flujo principal:**
  1. El alumno abre el historial.
  2. Se muestra una lista con fechas, horas y estado (entrada/salida).
* **Postcondiciones:** El historial se presenta de manera ordenada al usuario.



### **Caso de Uso 5: Escanear QR**

* **Actor principal:** Guardia
* **Descripción:** El guardia escanea el QR del alumno al ingresar o salir.
* **Precondiciones:** El guardia debe tener sesión activa en su app.
* **Flujo principal:**
  1. El guardia abre el lector QR.
  2. Escanea el código del alumno.
  3. Se muestra la información del alumno y vehículo.
* **Flujos alternos:**
  1. 2a. Si el QR no es válido o está duplicado, se bloquea el acceso.
* **Postcondiciones:** Se registra el evento (entrada o salida) en la base de datos.



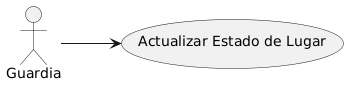
### **Caso de Uso 6: Asignar Espacio**

* **Actor principal:** Guardia
* **Descripción:** Se asigna un lugar de estacionamiento al vehículo.
* **Precondiciones:** QR escaneado correctamente.
* **Flujo principal:**
  1. El guardia selecciona manualmente un lugar disponible.
  2. (Opción avanzada: la app sugiere uno automáticamente).
  3. Se guarda la asignación en la base de datos.
* **Postcondiciones:** El lugar queda registrado como ocupado.



### **Caso de Uso 7: Actualizar Estado de Lugar**

* **Actor principal:** Guardia
* **Descripción:** El guardia puede liberar o marcar como ocupado un espacio.
* **Flujo principal:**
  1. Accede a la vista de lugares.
  2. Cambia el estado de un lugar específico.
* **Postcondiciones:** El mapa y la base de datos se actualizan.



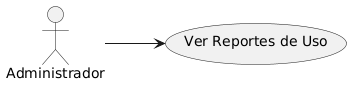
### **Caso de Uso 8: Monitorear Ocupación**

* **Actor principal:** Administrador
* **Descripción:** El panel muestra en tiempo real la ocupación del estacionamiento.
* **Precondiciones:** El sistema debe estar actualizado por los guardias.
* **Flujo principal:**
  1. Accede al panel administrativo.
  2. Visualiza estadísticas y mapa de ocupación.
* **Postcondiciones:** Se genera una vista global del uso actual del estacionamiento.



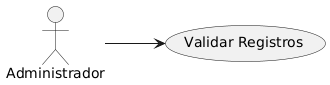
### **Caso de Uso 9: Ver Reportes de Uso**

* **Actor principal:** Administrador
* **Descripción:** Permite ver reportes sobre accesos, horarios pico y ocupación.
* **Flujo principal:**
  1. Selecciona un rango de fechas.
  2. Visualiza o descarga el reporte en formato tabla o gráfico.
* **Postcondiciones:** El administrador obtiene datos útiles para tomar decisiones.

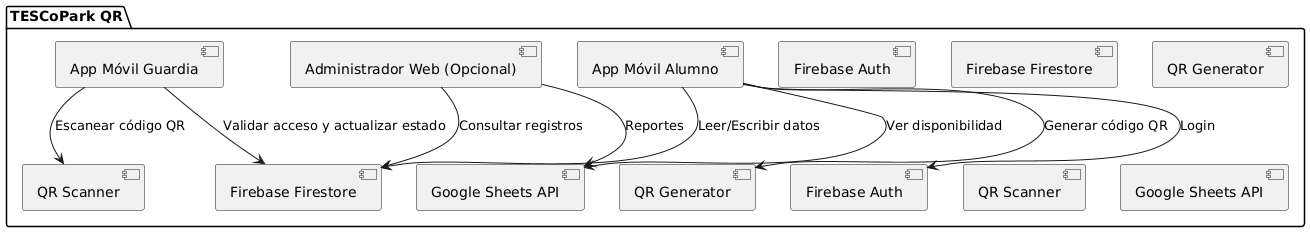


### **Caso de Uso 10: Validar Registros**

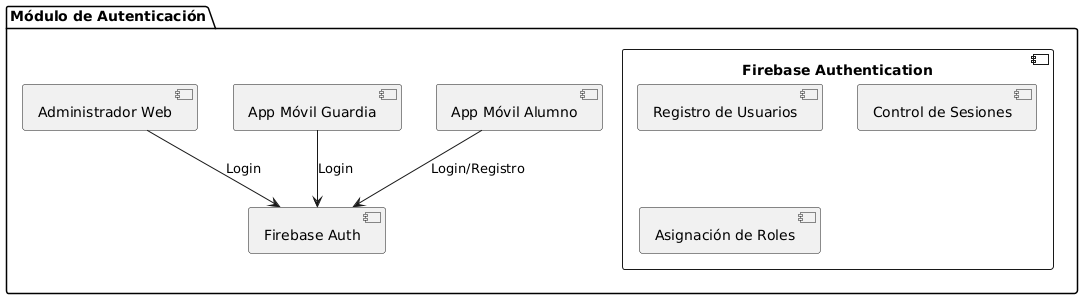
* **Actor principal:** Administrador
* **Descripción:** Valida o rechaza el registro de vehículos enviados por alumnos.
* **Flujo principal:**
  1. Accede a la lista de registros pendientes.
  2. Revisa la información.
  3. Acepta o rechaza el registro.
* **Postcondiciones:** El alumno es notificado y su QR se activa si fue aceptado.



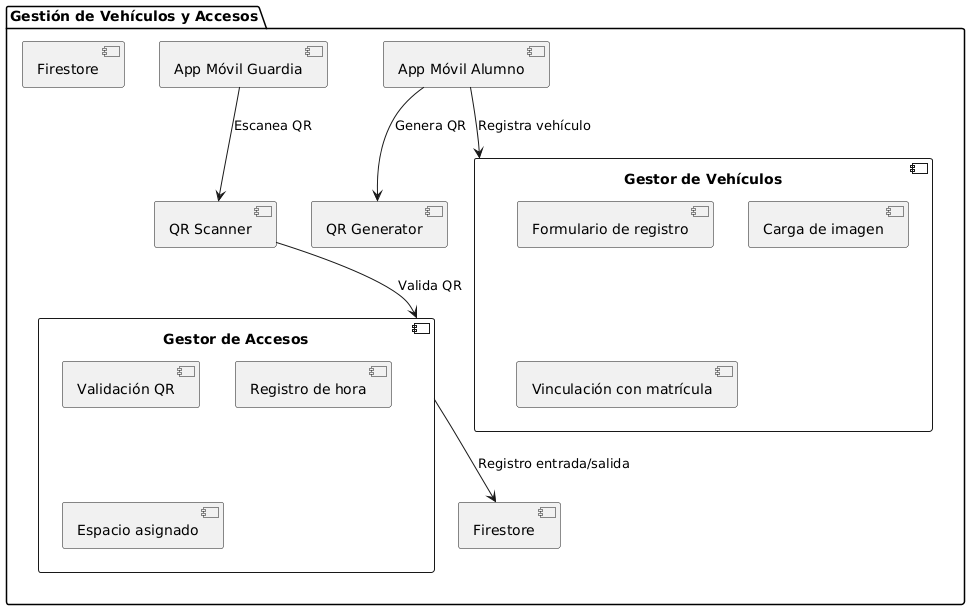
**Diagrama UML de Componentes – TESCoPark QR (General)**



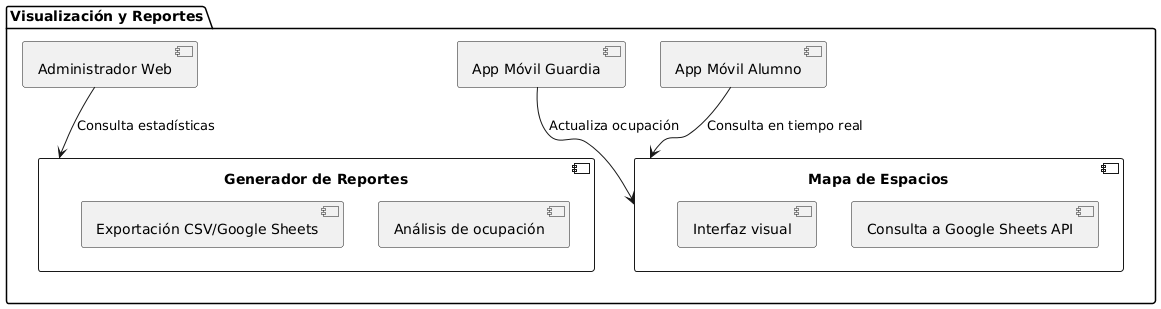
**Diagrama UML de Componentes – Autenticación**



**Diagrama UML de Componentes – Gestión de Vehículos y Accesos**



**Diagrama UML de Componentes – Visualización y Reportes**

****

# Fichas técnicas

**FICHA TÉCNICA 1 – App Móvil Alumno**

| **Elemento** | **Descripción** |
| --- | --- |
| **Nombre del módulo** | App Móvil - Alumno |
| **Objetivo** | Permitir a los alumnos registrar su vehículo, generar QR y consultar disponibilidad. |
| **Entradas** | Datos del vehículo (placas, modelo, imagen), credenciales del usuario. |
| **Salidas** | Código QR generado, visualización de espacios disponibles. |
| **Tecnologías utilizadas** | Flutter, Firebase Auth, Firestore, qr\_flutter, Google Sheets API |
| **Interfaz** | Pantallas móviles (registro, QR, mapa de espacios) |
| **Validaciones** | Datos obligatorios, formato de placas, existencia de cuenta registrada |
| **Responsable** | Equipo de desarrollo móvil |

**FICHA TÉCNICA 2 – App Móvil Guardia**

| **Elemento** | **Descripción** |
| --- | --- |
| **Nombre del módulo** | App Móvil - Guardia |
| **Objetivo** | Validar el acceso al estacionamiento mediante escaneo QR y asignación de espacios. |
| **Entradas** | Código QR escaneado, credenciales del guardia |
| **Salidas** | Confirmación de acceso, registro de entrada/salida, estado de espacios |
| **Tecnologías utilizadas** | Flutter, Firebase Auth, Firestore, mobile\_scanner, Google Sheets API |
| **Interfaz** | Pantallas móviles (escanear QR, asignar espacio, historial) |
| **Validaciones** | QR válido, usuario activo, espacio disponible |
| **Responsable** | Equipo de desarrollo móvil |

**FICHA TÉCNICA 3 – Backend / Base de Datos**

| **Elemento** | **Descripción** |
| --- | --- |
| **Nombre del módulo** | Backend Firebase (Firestore y Storage) |
| **Objetivo** | Almacenar y recuperar información de usuarios, vehículos y registros. |
| **Entradas** | Datos del vehículo, accesos, espacio ocupado |
| **Salidas** | Respuestas a consultas, actualización de datos |
| **Tecnologías utilizadas** | Firebase Firestore (NoSQL), Firebase Storage |
| **Estructura de datos** | Colecciones: usuarios, vehículos, accesos, espacios |
| **Validaciones** | Seguridad mediante reglas de Firestore |
| **Responsable** | Backend Developer / Administrador Firebase |

**FICHA TÉCNICA 4 – Módulo de Autenticación**

| **Elemento** | **Descripción** |
| --- | --- |
| **Nombre del módulo** | Firebase Authentication |
| **Objetivo** | Controlar el acceso seguro de usuarios al sistema |
| **Entradas** | Correo electrónico y contraseña |
| **Salidas** | Sesión iniciada, token de autenticación |
| **Tecnologías utilizadas** | Firebase Auth |
| **Validaciones** | Email válido, credenciales correctas |
| **Tipos de usuarios** | Alumno, Guardia, Administrador |
| **Responsable** | Backend Developer / Administrador Firebase |

**FICHA TÉCNICA 5 – Módulo de Código QR**

| **Elemento** | **Descripción** |
| --- | --- |
| **Nombre del módulo** | Generación y escaneo de códigos QR |
| **Objetivo** | Generar QR único para cada vehículo y escanearlo para validar acceso |
| **Entradas** | Datos del usuario y del vehículo |
| **Salidas** | Código QR estático o dinámico |
| **Tecnologías utilizadas** | qr\_flutter, mobile\_scanner |
| **Interacción** | App Alumno (genera QR), App Guardia (escanea QR) |
| **Validaciones** | Integridad del código QR, verificación en base de datos |
| **Responsable** | Equipo de desarrollo móvil |

**FICHA TÉCNICA 6 – Panel Administrativo**

| **Elemento** | **Descripción** |
| --- | --- |
| **Nombre del módulo** | Panel de Control Administrativo |
| **Objetivo** | Monitorear accesos, revisar registros, generar reportes |
| **Entradas** | Consultas desde interfaz web |
| **Salidas** | Reportes, estadísticas, validación de registros |
| **Tecnologías utilizadas** | Google Sheets API, Firebase Firestore, interfaz web (opcional con Flutter Web) |
| **Interacción** | Solo administradores |
| **Validaciones** | Acceso restringido mediante roles |
| **Responsable** | Administrador del sistema |