

# Universidad Peruana de Ciencias Aplicadas

---

INFORME DEL TRABAJO 1 (TP)



**Curso:** Diseño de Experimentos de Ingeniería de Software

**Sección:** 1ASI0732

**Profesor:** Julio Manuel Noriega Melendez

**Carrera:** Nombre de la carrera

**Ciclo:** 2025-01

**Startup:** Netvia

**Producto:** HomeyPark

Integrantes:

Nombre	Código
Sebastian Cachis Gonzales	u202210846
Adriano Sebastian Cruz Palomino	u202210697
Amner Levi Llamo Sanchez	u20221c376
Marcelo Fabian Garro Vega	u20201c410
Lucio Heli Yen Cerna	u202213143

Mayo del 2025

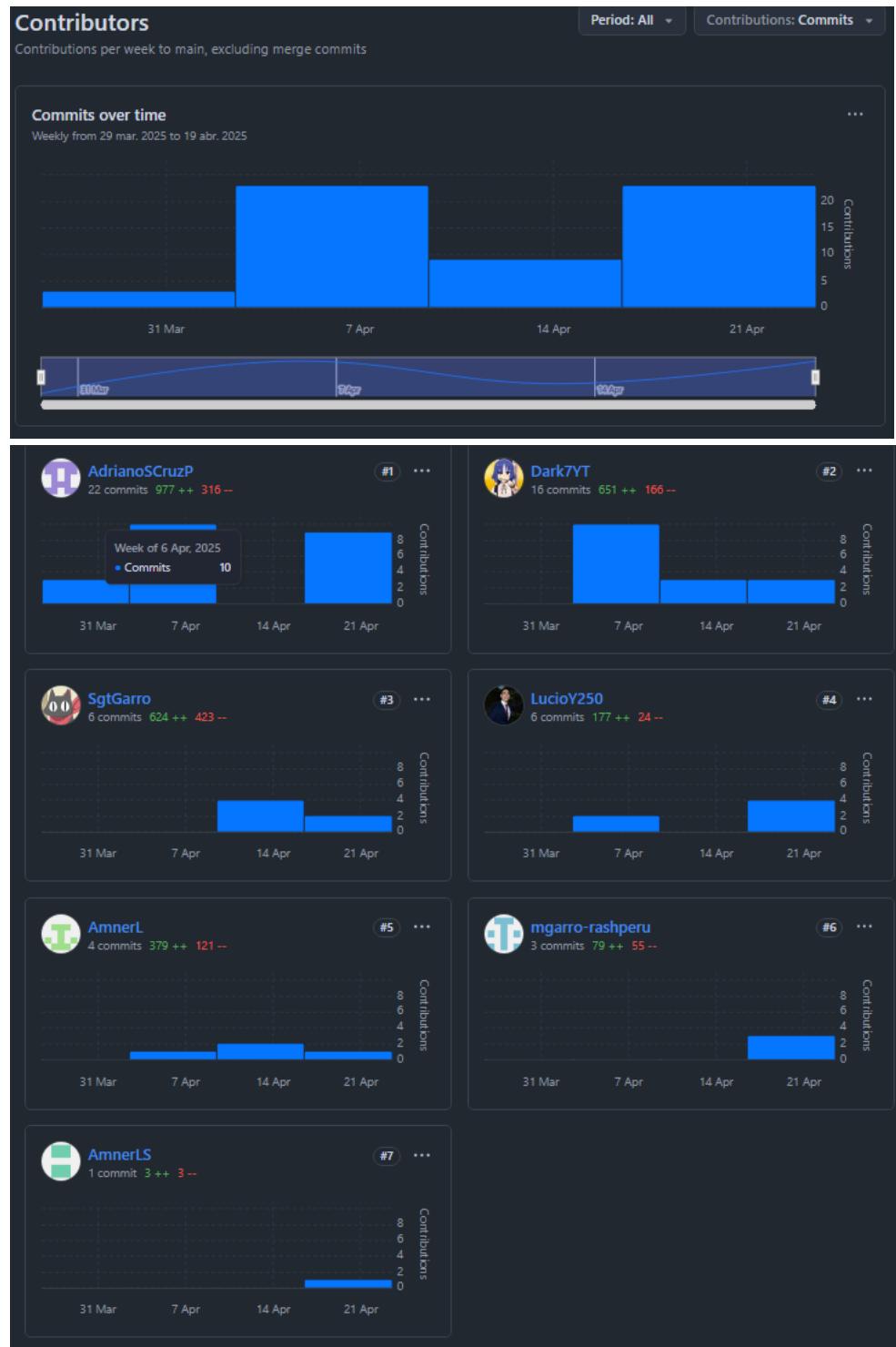
Registro de versiones del informe

Versión	Fecha	Autor	Descripción
1.0	05/04/2025	Adriano Cruz	Creación de la estructura del informe
1.1	06/04/2025	Sebastian Cachis	Desarrollo del capítulo I
1.2	06/04/2025	Sebastian Cachis	Desarrollo del capítulo V
1.3	19/04/2025	Adriano Cruz	Desarrollo del capítulo IV
1.4	07/04/2025	Amner Llamo	Desarrollo del capítulo II
1.5	22/04/2025	Marcelo Garro	Desarrollo del capítulo II
1.6	24/04/2025	Lucio Yen	Desarrollo del capítulo IV
2.0	13/05/2025	Equipo Netvia	Desarrollo del capítulo V y VI

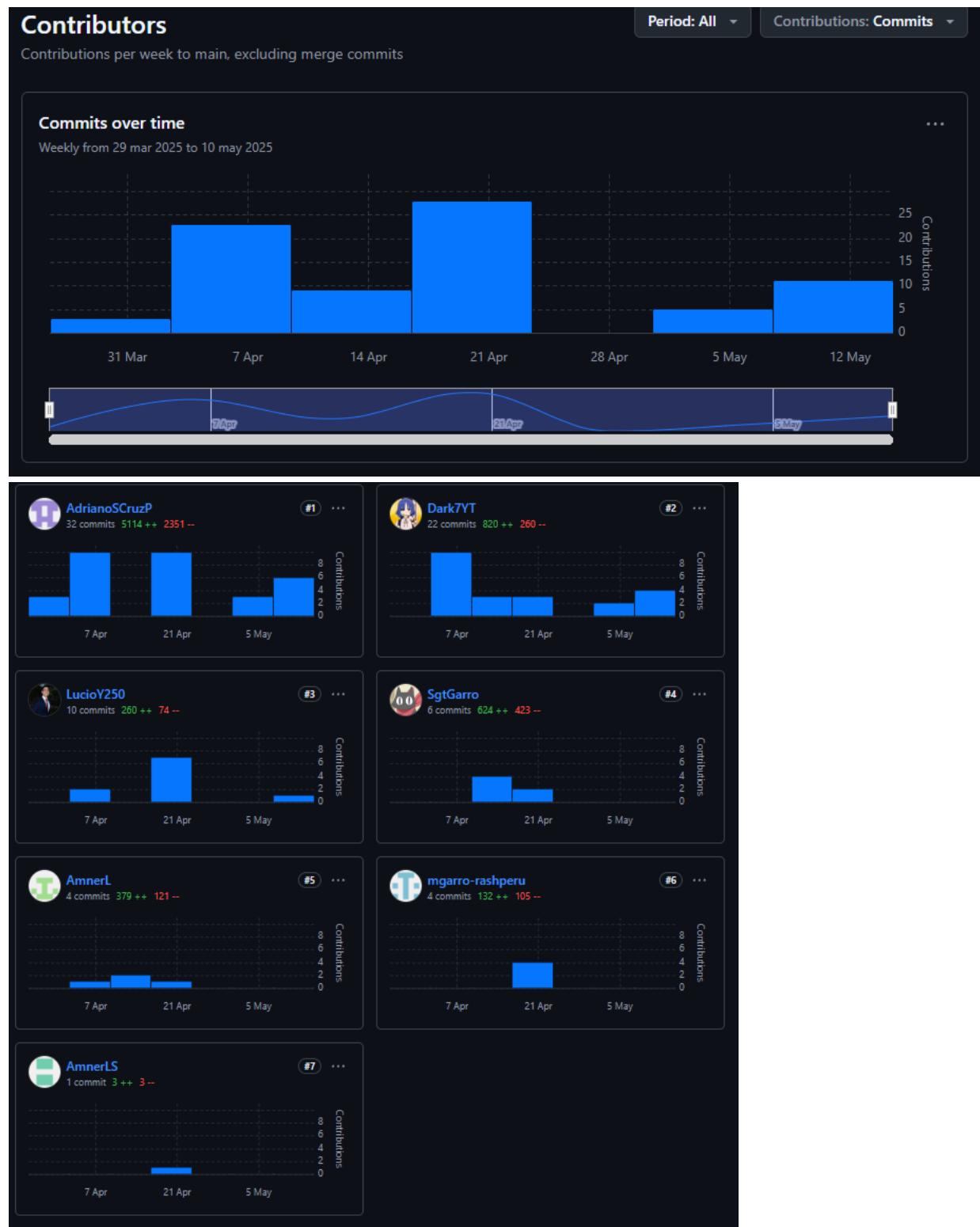
Project Report Collaboration Insights

## TB1

Para el desarrollo del informe perteneciente a la entrega TB1, se dividió la implementación de secciones de la siguiente forma para cada integrante del equipo:

**TP1**

Para el desarrollo del informe perteneciente a la entrega TP1, se dividió la implementación de secciones de la siguiente forma para cada integrante del equipo:



## Tabla de Contenidos

### Student Outcome

#### Capítulo I: Introducción

- 1.1. Startup Profile
  - 1.1.1. Descripción de la Startup
  - 1.1.2. Perfiles de integrantes del equipo
- 1.2. Solution Profile
  - 1.2.1. Antecedentes y problemática
  - 1.2.2. Lean UX Process
    - 1.2.2.1. Lean UX Problem Statements
    - 1.2.2.2. Lean UX Assumptions
    - 1.2.2.3. Lean UX Hypothesis Statements
    - 1.2.2.4. Lean UX Canvas

- 1.3. Segmentos objetivo

## Capítulo II: Requirements Elicitation & Analysis

- 2.1. Competidores
  - 2.1.1. Análisis competitivo
  - 2.1.2. Estrategias y tácticas frente a competidores
- 2.2. Entrevistas
  - 2.2.1. Diseño de entrevistas
  - 2.2.2. Registro de entrevistas
  - 2.2.3. Análisis de entrevistas
- 2.3. Needfinding
  - 2.3.1. User Personas
  - 2.3.2. User Task Matrix
  - 2.3.3. User Journey Mapping
  - 2.3.4. Empathy Mapping
  - 2.3.5. As-is Scenario Mapping
- 2.4. Ubiquitous Language

## Capítulo III: Requirements Specification

- 3.1. To-Be Scenario Mapping
- 3.2. User Stories
- 3.3. Product Backlog
- 3.4. Impact Mapping

## Capítulo IV: Product Design

- 4.1. Style Guidelines
  - 4.1.1. General Style Guidelines
  - 4.1.2. Web Style Guidelines
  - 4.1.3. Mobile Style Guidelines
    - 4.1.3.1. iOS Mobile Style Guidelines
    - 4.1.3.2. Android Mobile Style Guidelines
- 4.2. Information Architecture
  - 4.2.1. Organization Systems
  - 4.2.2. Labeling Systems
  - 4.2.3. SEO Tags and Meta Tags
  - 4.2.4. Searching Systems
  - 4.2.5. Navigation Systems
- 4.3. Landing Page UI Design
  - 4.3.1. Landing Page Wireframe
  - 4.3.2. Landing Page Mock-up
- 4.4. Mobile Applications UX/UI Design
  - 4.4.1. Mobile Applications Wireframes
  - 4.4.2. Mobile Applications Wireflow Diagrams
  - 4.4.3. Mobile Applications Mock-ups
  - 4.4.4. Mobile Applications User Flow Diagrams
- 4.5. Mobile Applications Prototyping
  - 4.5.1. Android Mobile Applications Prototyping
  - 4.5.2. iOS Mobile Applications Prototyping
- 4.6. Web Applications UX/UI Design
  - 4.6.1. Web Applications Wireframes
  - 4.6.2. Web Applications Wireflow Diagrams
  - 4.6.3. Web Applications Mock-ups
  - 4.6.4. Web Applications User Flow Diagrams
- 4.7. Web Applications Prototyping
- 4.8. Domain-Driven Software Architecture
  - 4.8.1. Software Architecture Context Diagram
  - 4.8.2. Software Architecture Container Diagrams
  - 4.8.3. Software Architecture Components Diagrams
- 4.9. Software Object-Oriented Design
  - 4.9.1. Class Diagrams
  - 4.9.2. Class Dictionary
- 4.10. Database Design
  - 4.10.1. Relational/Non-Relational Database Diagram

## Capítulo V: Product Implementation

- 5.1. Software Configuration Management
  - 5.1.1. Software Development Environment Configuration
  - 5.1.2. Source Code Management

- 5.1.3. Source Code Style Guide & Conventions
- 5.1.4. Software Deployment Configuration
- 5.2. Product Implementation & Deployment
  - 5.2.1. Sprint Backlogs
  - 5.2.2. Implemented Landing Page Evidence
  - 5.2.3. Implemented Frontend-Web Application Evidence
  - 5.2.4. Acuerdo de Servicio - SaaS
  - 5.2.5. Implemented Native-Mobile Application Evidence
  - 5.2.6. Implemented RESTful API and/or Serverless Backend Evidence
  - 5.2.7. RESTful API documentation
  - 5.2.8. Team Collaboration Insights
- 5.3. Video About-the-Product

## Capítulo VI: Product Verification & Validation

- 6.1. Testing Suites & Validation
  - 6.1.1. Core Entities Unit Tests
  - 6.1.2. Core Integration Tests
  - 6.1.3. Core Behavior-Driven Development
  - 6.1.4. Core System Tests
- 6.2. Static testing & Verification
  - 6.2.1. Static Code Analysis
    - 6.2.1.1. Coding standard & Code conventions
    - 6.2.1.2. Code Quality & Code Security
  - 6.2.2. Reviews
- 6.3. Validation Interviews
  - 6.3.1. Diseño de Entrevistas
  - 6.3.2. Registro de Entrevistas
  - 6.3.3. Evaluaciones según heurísticas
- 6.4. Auditoría de Experiencias de Usuario
  - 6.4.1. Auditoría realizada
    - 6.4.1.1. Información del grupo auditado
    - 6.4.1.2. Cronograma de auditoría realizada
    - 6.4.1.3. Contenido de auditoría realizada
  - 6.4.2. Auditoría recibida
    - 6.4.2.1. Información del grupo auditor
    - 6.4.2.2. Cronograma de auditoría recibida
    - 6.4.2.3. Contenido de auditoría recibida
    - 6.4.2.4. Resumen de modificaciones para subsanar hallazgos

## Capítulo VII: DevOps Practices

- 7.1. Continuous Integration
  - 7.1.1. Tools and Practices
  - 7.1.2. Build & Test Suite Pipeline Components
- 7.2. Continuous Delivery
  - 7.2.1. Tools and Practices
  - 7.2.2. Stages Deployment Pipeline Components
- 7.3. Continuous Deployment
  - 7.3.1. Tools and Practices
  - 7.3.2. Production Deployment Pipeline Components
- 7.4. Continuous Deployment
  - 7.4.1. Tools and Practices
  - 7.4.2. Monitoring Pipeline Components
  - 7.4.3. Alerting Pipeline Components
  - 7.4.4. Notification Pipeline Components

## Capítulo VIII: Experiment-Driven Development

- 8.1. Experiment Planning
  - 8.1.1. As-Is Summary
  - 8.1.2. Raw Material: Assumptions, Knowledge Gaps, Ideas, Claims
  - 8.1.3. Experiment-Ready Questions
  - 8.1.4. Question Backlog
  - 8.1.5. Experiment Cards

- 8.2. Experiment Design
  - 8.2.1. Hypotheses
  - 8.2.2. Measures
  - 8.2.3. Conditions
  - 8.2.4. Scale Calculations and Decisions
  - 8.2.5. Methods Selection
  - 8.2.6. Data Analytics: Goals, KPIs and Metrics Selection
  - 8.2.7. Web and Mobile Tracking Plan
- 8.3. Experimentation
  - 8.3.1. To-Be User Stories
  - 8.3.2. To-Be Product Backlog
- Conclusiones, Bibliografía y Anexos

## Student Outcome

---

Student Outcome

Criterio específico	Acciones Realizadas	Conclusiones

Criterio específico	Acciones Realizadas	Conclusiones
	<p>Sebastian Nicolas Cachis Gonzales</p> <p><b>TB1</b></p> <p>Para esta entrega desarollé el capítulo 1, hice entrevistas y también los diseños para la landing page.</p> <p><b>TP1</b></p> <p>Participé en la refactorización del backend y llevé a cabo pruebas unitarias e integrales para asegurar el correcto funcionamiento y calidad del sistema.</p>	
	<p>Amner Levi Llamo Sánchez</p> <p><b>TB1</b></p> <p>En el análisis competitivo, evalué las prácticas éticas de las soluciones existentes y documenté aquellas que debemos adoptar o mejorar en HomeyPark.</p> <p><b>TP1</b></p> <p>Apoyé en la refactorización del backend y ejecuté pruebas unitarias e integrales siguiendo criterios de calidad y responsabilidad profesional.</p>	
Reconoce responsabilidad ética y profesional en situaciones de ingeniería de software	<p>Marcelo Garro</p> <p><b>TB1</b></p> <p>Planifiqué y recopilé requisitos del proyecto en base a análisis previos de nuestro segmento objetivo para satisfacer sus necesidades y demandas.</p> <p><b>TP1</b></p> <p>Me encargué de la refactorización del frontend y verifiqué su funcionamiento mediante pruebas funcionales que garanticen una experiencia amigable y coherente.</p>	<p><b>TB1</b></p> <p>Designamos tareas a cada integrante para optimizar el tiempo de trabajo.</p> <p><b>TP1</b></p> <p>Dividimos las responsabilidades considerando las fortalezas de cada miembro, logrando una ejecución ordenada, ética y eficaz del proyecto.</p>
	<p>Adriano Sebastian Cruz Palomino</p> <p><b>TB1</b></p> <p>Me encargué del desarrollo de los diagramas de arquitectura del sistema utilizando el modelo C4, asegurando que la estructura técnica promoviera una implementación ética, segura y comprensible para todos los stakeholders.</p> <p><b>TP1</b></p> <p>Implementé pruebas unitarias e integrales, y configuré flujos de CI/CD en el backend para reforzar las prácticas DevOps y garantizar un desarrollo continuo y controlado.</p>	
	<p>Lucio Heli Yen Cerna</p> <p><b>TB1</b></p> <p>Me encargué del desarrollo de los style guidelines tanto para la landing page, aplicativo móvil en Android e iOS, y en la plataforma web, asegurando un diseño ético, de fácil entendimiento y para todos los usuarios.</p> <p><b>TP1</b></p> <p>Llevé a cabo la refactorización de la aplicación móvil, orientando el diseño hacia una mejor accesibilidad y usabilidad para todos los usuarios.</p>	

Criterio específico	Acciones Realizadas	Conclusiones
	<p>Sebastian Nicolas Cachis Gonzales</p> <p><b>TB1</b> Esto me ha permitido tener una mejor visión de los límites y objetivos de nuestro proyecto, así como conocer las inquietudes de nuestros segmentos objetivos.</p> <p><b>TP1</b> Al realizar las pruebas y refactorizar el backend, comprendí cómo estas tareas contribuyen a construir soluciones que respondan adecuadamente a las necesidades del entorno urbano y tecnológico.</p>	
Emite juicios informados considerando el impacto de las soluciones de ingeniería de software en contextos globales, económicos, ambientales y sociales	<p>Amner Levi Llamo Sánchez</p> <p><b>TB1</b> Analicé el impacto potencial de HomeyPark en el contexto urbano, evaluando cómo la plataforma podría afectar la movilidad, el uso eficiente de espacios y la economía local.</p> <p><b>TP1</b> Durante el proceso técnico, evalué cómo los cambios en la estructura del backend pueden repercutir directamente en la escalabilidad, seguridad y adaptabilidad del sistema frente a diversos contextos sociales.</p>	
	<p>Marcelo Garro</p> <p><b>TB1</b> Planificando el proyecto, obtuve una perspectiva más clara sobre los alcances y objetivos del proyecto, así como un mayor entendimiento de las inquietudes y expectativas de nuestros segmentos objetivo.</p> <p><b>TP1</b> A través de la mejora del frontend, pude observar cómo una buena interfaz puede facilitar la inclusión tecnológica y aportar valor en distintos entornos sociales y económicos.</p>	<p><b>TB1</b> Hemos enfocado las habilidades de cada integrante en las áreas de desarrollo que mejor dominen para una mejor línea de trabajo.</p> <p><b>TP1</b> Reflexionamos sobre cómo cada mejora técnica tiene implicancias reales en el entorno social y urbano, y buscamos que nuestras decisiones promuevan una solución que responda a dichas realidades.</p>
	<p>Adriano Sebastian Cruz Palomino</p> <p><b>TB1</b> Al definir la arquitectura C4, consideré cómo cada componente afectaría a nivel de escalabilidad, costos operativos y su adecuación a contextos urbanos sostenibles, para que la solución pueda adaptarse a diferentes realidades sociales y económicas.</p> <p><b>TP1</b> Al configurar los procesos de integración continua y realizar pruebas, pude confirmar que una arquitectura sólida favorece no solo la eficiencia del sistema, sino también su sostenibilidad y adaptabilidad a largo plazo.</p>	
	<p>Lucio Heli Yen Cerna</p> <p><b>TB1</b> Diseñando el proyecto, pude entender de forma más clara la perspectiva que necesitan nuestros segmentos objetivos para que este proyecto tenga un impacto positivo en su rutina diaria.</p> <p><b>TP1</b> Gracias a la refactorización de la app móvil, comprendí cómo decisiones de diseño bien pensadas pueden tener un efecto positivo en la vida diaria de nuestros usuarios y en su interacción con la tecnología.</p>	

## Capítulo I: Introducción

### 1.1. Startup Profile

#### 1.1.1. Descripción de la Startup

Nuestro equipo Netvia ha sido creado con el propósito de solucionar la preocupación por la escasez de espacios de estacionamiento en entornos urbanos. La congestión del tráfico y la ineficiente búsqueda de estacionamientos genera una gran frustración para el conductor. HomeyPark propone revolucionar la forma en que las personas encuentran y utilizan espacios de estacionamiento. Desarrollaremos una aplicación que ofrecerá una interfaz intuitiva, permitiendo a los usuarios buscar, reservar y pagar por estacionamientos de manera sencilla y rápida.

## Misión

Facilitar el intercambio equitativo de habilidades y servicios entre personas, empoderando a las comunidades para crear valor compartido sin intermediación monetaria, promoviendo así una economía colaborativa basada en el talento y las conexiones humanas.

## Visión

Ser la plataforma líder global en intercambio de servicios peer-to-peer, transformando la manera en que las personas intercambian valor, fomentando una sociedad más colaborativa donde el acceso a servicios y conocimientos no esté limitado por restricciones económicas sino potenciado por el talento colectivo de la comunidad.

### 1.1.2. Perfiles de integrantes del equipo

Descripción de los perfiles de los integrantes del equipo	Foto del integrante
<p>Mi nombre es Soy un estudiante de 22 años con interés en el desarrollo web y móvil. Disfruto aprender nuevas tecnologías y colaborar en proyectos donde pueda aportar con mis conocimientos. Me motiva compartir experiencias con otros y crecer junto a mi equipo en cada desafío.</p>	
<p>Mi nombre es <b>Amner Levi Llamo Sánchez</b>, soy estudiante del séptimo ciclo de ingeniería de software en la UPC. Me gusta jugar fútbol y videojuegos, por eso estoy constantemente investigando sobre nuevas tecnologías. Soy responsable con los trabajos que se me asignan; además soy tolerante y me adapto a las circunstancias del equipo.</p>	
<p>Mi nombre es <b>Sebastian Nicolas Cachis Gonzales</b>, soy estudiante de séptimo ciclo de ingeniería de software en la UPC. Me considero una persona proactiva, organizada, meticulosa y muy enfocada en mis estudios, tanto grupales como individuales. Tengo facilidad para entender y ejemplificar los distintos temas que vemos, teniendo soltura para explicar.</p>	
<p>Mi nombre es <b>Adriano Sebastian Cruz Palomino</b>, tengo 20 años, soy alumno de Ingeniería de Software en la UPC, actualmente estoy cursando el 7mo ciclo. Soy una persona curiosa, responsable, y comprometida con mis estudios, siempre busco aprender más y mejorar mis habilidades.</p>	
<p>Mi nombre es <b>Lucio Heli Yen Cerna</b>, soy estudiante del séptimo ciclo de la carrera de Ingeniería de Software en la UPC. Soy una persona proactiva y organizada que se esmera en construir productos de calidad innovadores. Me apasiona mucho trabajar en equipo, debatir y compartir una misma motivación debido a que siento que aprendo de mis propios compañeros y mejoro como profesional. Por otro lado, mis hobbies son el gimnasio, la música y los videojuegos los cuales me permiten llevar un estilo de vida balanceado y saludable.</p>	

## 1.2. Solution Profile

### 1.2.1. Antecedentes y problemática

En la actualidad, la congestión vehicular ha ido en aumento debido al crecimiento constante de la población y al incremento del uso de vehículos. Este problema se ve agravado por la ineficiencia en la búsqueda de espacios de estacionamiento, lo que ha generado una experiencia frustrante para los conductores a nivel mundial. A su vez, aquellos que poseen espacios de estacionamiento sin utilizar en áreas urbanas se enfrentan al desafío de no contar con una plataforma eficaz para rentabilizar estos recursos.

#### What?

Nuestro startup ha identificado como problemática principal la escasez de espacios de estacionamiento en entornos urbanos y la complejidad que representa actualmente encontrar un lugar para estacionar. Esto se debe a la elevada demanda de vehículos para las actividades cotidianas.

#### When?

Esta preocupación ha ido en aumento a lo largo del tiempo, ya que las ciudades han visto crecer su población, lo que ha resultado en un mayor número de vehículos en circulación. En los últimos años, la congestión del tráfico y la dificultad para encontrar estacionamiento se han vuelto problemas más urgentes.

#### Where?

El problema se presenta principalmente en áreas urbanas densamente pobladas a nivel global, donde el espacio es limitado y la demanda de estacionamiento es alta.

#### Who?

Los conductores son los principales afectados por este problema, ya que se enfrentan a dificultades para encontrar estacionamientos convenientes. Además, los propietarios de espacios de estacionamiento se ven en desventaja frente a grandes empresas del sector, lo que complica la promoción y alquiler de sus servicios.

#### Why?

La causa principal radica en la insuficiencia de espacios de estacionamiento disponibles en áreas urbanas, lo que intensifica la congestión vehicular y dificulta que los conductores encuentren un lugar para estacionar en el momento oportuno.

#### How?

El problema ocurre cuando la congestión del tráfico, combinada con la falta de espacios de estacionamiento, impide que la población pueda estacionar de manera eficiente y conveniente.

## How much?

Este problema afecta de manera notable a Lima, la capital, donde según un estudio realizado por la ONG Luz Ámbar en 2016, existe una carencia de aproximadamente 45,000 espacios de estacionamiento en cinco distritos. Sin embargo, esta cifra es insuficiente en comparación con la cantidad de vehículos en la ciudad, que alcanza aproximadamente 1 millón 800,000 unidades. Esta discrepancia entre la cantidad de vehículos y la disponibilidad de estacionamientos contribuye significativamente a la congestión vehicular y al desafío constante de encontrar un lugar adecuado para estacionar.

### 1.2.2. Lean UX Process

#### 1.2.2.1. Lean UX Problem Statements

El desarrollo de HomeyPark sigue la metodología Lean UX, que combina el pensamiento Lean Startup con el diseño centrado en el usuario. Este enfoque nos permite validar rápidamente nuestras hipótesis de valor y crear soluciones adaptadas a las necesidades reales de los usuarios con el mínimo de recursos.

##### **Problem Statement 1: Usuarios de parking**

En la actualidad, muchos ciudadanos de las zonas urbanas del Perú requieren de un vehículo motorizado para realizar tramos largos de viaje. Debido a su alta demanda, el diario "El Comercio" (2024) revela que el 51% de conductores de la ciudad de Lima consideran necesario incrementar la cantidad de espacios para estacionar.

Hemos encontrado que los conductores presentan dificultades para encontrar espacios de estacionamiento disponibles, tomando un aproximado de 10 horas totales al mes para conseguir uno.

¿Cómo podemos buscar o proporcionar más espacios de estacionamiento para los usuarios de forma eficiente y rápida para sus vehículos motorizados?

##### **Problem Statement 2: Anfitriones**

Nuestra aplicación permitirá a los usuarios poder promocionar sus garajes para obtener una fuente de ingresos adicional sin interrupción en sus actividades del día a día. Esto favorecerá a las comunidades para la reducción de la congestión vehicular.

Hemos encontrado que muchos propietarios no están dispuestos a ofrecer su garaje en alquiler debido al anonimato de los clientes, lo cual genera una sensación de inseguridad en estas entidades.

¿Cómo podemos implementar un sistema de seguridad que vele por la integridad, bienes y bienes inmobiliarios de nuestros usuarios de la aplicación?

### 1.2.2.2. Lean UX Assumptions

#### **Business Assumptions**

1. Creo que mis usuarios necesitan una mejor opción de encontrar estacionamientos y, de ser posible, reservarlos.
2. Estas necesidades se pueden resolver con una aplicación móvil que les permita a los conductores reservar en los garajes de la ciudad, debido a que la mayoría del tiempo se encuentran disponibles.
3. Mis clientes iniciales son las personas que cuenten con un vehículo y que tengan dificultades de encontrar algún estacionamiento disponible.
4. El valor #1 que el cliente requiere de mi servicio es encontrar y reservar espacios para estacionar en un corto periodo de tiempo de forma segura.
5. Voy a adquirir a mis clientes a través de estrategias de marketing en diversas redes sociales, mostrando todos los beneficios que da nuestra aplicación móvil.
6. Mi competencia en el mercado serán las empresas que se dedican a ofrecer sus servicios de estacionamiento.
7. Los venceremos debido a que ofrecemos a los usuarios poder generar ingresos de manera pasiva al rentar sus garajes como estacionamiento.
8. Mis mayores riesgos del producto es no encontrar una manera de brindar seguridad a los conductores como a los propietarios de los garajes.
9. Resolveremos esto con la incorporación de un sistema que se encargue de validar los parámetros de seguridad de los conductores y los garajes en alquiler para ofrecer una mejor seguridad al público.

#### **User Assumptions**

##### *Usuarios de parking*

**¿Quién es el usuario?** Conductores que en su día a día necesitan encontrar estacionamiento para sus vehículos.

**¿Dónde encaja nuestro servicio?** Nuestro servicio encaja tanto para su trabajo como para sus actividades diarias.

**¿Qué problema tiene nuestro servicio y cómo se resuelve?** El problema es sobre la posible inseguridad del usuario al alquilar en una cochera de cualquier persona desconocida. Se puede resolver mediante un sistema de filtros que garanticen al usuario la seguridad de la cochera y contar con bases legales para la publicación del producto.

**¿Cuándo y cómo es usado nuestro producto?** Nuestro producto será usado mayormente cuando el usuario necesite encontrar algún estacionamiento para realizar cualquier actividad. La aplicación móvil se podrá usar como un sistema de búsqueda y reserva de cocheras.

##### *Anfitriones*

**¿Quién es el usuario?** Personas con garajes que desean poner en alquiler para generar ingresos.

**¿Dónde encaja nuestro servicio?** Nuestro servicio encaja en su vida porque pueden poner en alquiler sus garajes mientras realizan cualquier actividad.

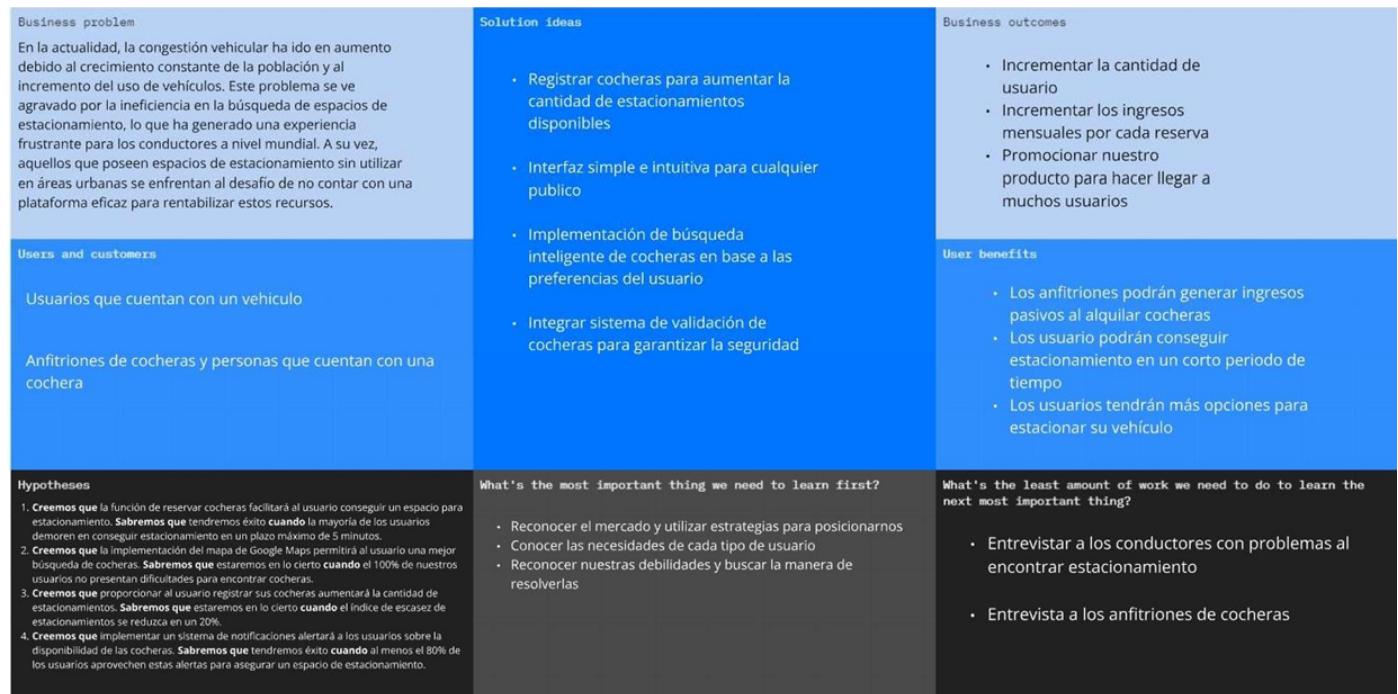
**¿Qué problema tiene nuestro servicio y cómo se resuelve?** El problema será el proceso para poder registrar sus cocheras, debido a que puede llegar a ser confuso o tedioso para el anfitrión. Lo podemos resolver mediante capacitación sobre el proceso de filtros para facilitar al usuario el registro.

**¿Cuándo y cómo es usado nuestro producto?** Nuestra aplicación es usada principalmente cuando el anfitrión dispone de cualquier garaje disponible que desee poner en alquiler para generar ingresos. Nuestro producto es usado como un gestor de cocheras sobre reservas, estados, recibos, etc.

#### 1.2.2.3. Lean UX Hypothesis Statements

1. Creemos que la función de reservar cocheras facilitará al usuario conseguir un espacio para estacionamiento. Sabremos que tendremos éxito cuando la mayoría de los usuarios demoren en conseguir estacionamiento en un plazo máximo de 5 minutos.
2. Creemos que la implementación del mapa de Google Maps permitirá al usuario una mejor búsqueda de cocheras. Sabremos que estaremos en lo cierto cuando el 100% de nuestros usuarios no presentan dificultades para encontrar cocheras.
3. Creemos que proporcionar al usuario registrar sus cocheras aumentará la cantidad de estacionamientos. Sabremos que estaremos en lo cierto cuando el índice de escasez de estacionamientos se reduzca en un 20%.
4. Creemos que implementar un sistema de notificaciones alertará a los usuarios sobre la disponibilidad de las cocheras. Sabremos que tendremos éxito cuando al menos el 80% de los usuarios aprovechen estas alertas para asegurar un espacio de estacionamiento.

#### 1.2.2.4. Lean UX Canvas



### 1.3. Segmentos objetivo

Nuestro segmento objetivo está compuesto por dos usuarios:

**Usuarios de parking:** Persona que busca alguna solución para su necesidad que es buscar un estacionamiento en entornos urbanos para su vehículo

**Anfitrión:** Propietario de una vivienda que cuente con una cochera privada que busque sacar provecho de forma efectiva.

## Capítulo II: Requirements Elicitation & Analysis

### 2.1. Competidores

#### 2.1.1. Análisis competitivo

Nuestros competidores principalmente vendrían a ser otras aplicaciones que se encargan del servicio de estacionamiento por celular. Algunos de nuestros competidores potenciales vendrían a ser:

1. EasyPark: Es una app popular que permite a los usuarios encontrar y pagar por estacionamientos en diversas ciudades. EasyPark se integra con sistemas de estacionamiento inteligente, ofreciendo información en tiempo real sobre la disponibilidad de espacios, y cuenta con opciones de pago automatizado y reconocimiento de placas. La app también permite reservar espacios con anticipación y está disponible en varios países, facilitando la movilidad urbana a nivel internacional.
2. Parkopedia: Esta aplicación ofrece una amplia base de datos de estacionamientos, incluyendo garajes privados y públicos. Parkopedia también proporciona predicciones sobre la disponibilidad de espacios y se integra con sistemas de navegación en vehículos. Además, ofrece información detallada sobre tarifas y horarios, lo que ayuda a los usuarios a tomar decisiones más informadas al buscar estacionamiento. Su cobertura incluye más de 15,000 ciudades en todo el mundo.
3. Wayleadr: Se centra en la optimización de espacios de estacionamiento en áreas urbanas y corporativas. Ofrece características como reservas anticipadas y gestión de estacionamiento para empresas, utilizando geolocalización para mejorar la eficiencia. También permite a las empresas monitorear y gestionar sus

plazas de estacionamiento, reduciendo costos operativos y mejorando la experiencia de los empleados. Wayleadr ha sido adoptada por importantes corporaciones globales para optimizar el uso de sus instalaciones.

## 2.1.2. Estrategias y tácticas frente a competidores

**Competitive Analysis Landscape**

¿Por qué llevar a cabo este análisis?		Un análisis competitivo ayuda a identificar oportunidades y mejorar su propuesta de valor para mantenerse relevante en el mercado.		
Perfil	Overview	HomeyPark	EasyPark	Parkopedia
Perfíl	Ventaja competitiva ¿Qué valor ofrece a los clientes?	Una aplicación que conecta conductores con garajes privados disponibles, permitiéndoles reservar espacios de estacionamiento fácilmente.	Una app popular que ayuda a encontrar, reservar y pagar estacionamientos fácilmente.	Una base de datos global que ofrece información detallada sobre estacionamientos y disponibilidad.
Perfíl de Marketing	Mercado objetivo	Conductores Propietarios de garajes	Conductores urbanos Viajeros frecuentes	Conductores Empresas de navegación vehicular
	Estrategias de marketing	Marketing en redes sociales, marketing de contenido	Marketing en redes sociales, marketing de contenido	Marketing en redes sociales, marketing de contenido
Perfíl de Producto	Productos & Servicios	Reserva de garajes a través de la app, integración con Google Maps para la localización de espacios, y un sistema de seguridad para transacciones.	App de reserva de estacionamiento, pagos automatizados, y notificaciones de disponibilidad en tiempo real.	Base de datos de estacionamientos, predicciones de disponibilidad, e integración con sistemas de navegación.
	Precios & Costos	El precio varía según el servicio que adquiere el usuario.	El precio varía según el servicio que adquiere el usuario.	El precio varía según el servicio que adquiere el usuario.
	Canales de distribución (Web y/o Móvil)	Aplicación móvil y web.	Aplicación móvil y web.	Aplicación móvil y web.
Análisis SWOT	Fortalezas	Proporciona una solución innovadora para la falta de estacionamientos en áreas urbanas al aprovechar los garajes privados, lo que también beneficia a las comunidades locales.	Amplia adopción y fácil integración con sistemas de pago y navegación, lo que facilita su uso en diversas ciudades.	Ofrece la base de datos de estacionamientos más extensa y detallada a nivel global, con información en tiempo real.
	Debilidades	La seguridad tanto para los conductores como para los propietarios de garajes puede ser una preocupación, lo que podría afectar la adopción.	Puede enfrentar dificultades en regiones donde la infraestructura de estacionamiento inteligente es limitada.	Su dependencia de datos de terceros puede afectar la precisión y actualización de la información.
	Oportunidades	Expansión a nuevas ciudades y la posibilidad de integrar tecnologías avanzadas de seguridad y autenticación para aumentar la confianza de los usuarios.	Expandir su servicio a más ciudades y regiones que aún no cuentan con soluciones avanzadas de estacionamiento.	Integrar tecnologías avanzadas como inteligencia artificial para mejorar la precisión de las predicciones de disponibilidad de estacionamiento.
	Amenazas	Competencia de aplicaciones similares que podrían ofrecer características de seguridad o precios más competitivos, así como	La creciente competencia de nuevas aplicaciones de estacionamiento con funcionalidades más	Los cambios en las regulaciones de datos y privacidad pueden limitar el acceso a la información
				La dependencia de un nicho específico como el mercado corporativo puede limitar su crecimiento frente a

la posible resistencia de los usuarios a confiar en garajes privados. avanzadas podría reducir su cuota de mercado. necesaria para mantener su base de datos actualizada. soluciones más generalistas.

## 2.2. Entrevistas

### 2.2.1. Diseño de entrevistas

Para obtener una comprensión profunda de las necesidades y expectativas de los usuarios potenciales de HomeyPark, se diseñarán entrevistas estructuradas con preguntas específicas para diferentes segmentos de usuarios. A continuación, se presentan las preguntas para cada grupo:

#### Preguntas Generales

- ¿Cuál es su nombre y apellido completo?
- ¿Cuál es su edad?
- ¿En qué distrito o lugar reside?
- ¿Cuál es su ocupación?

#### Preguntas para el usuario de Parking

- ¿Con qué tipo de vehículo dispone? ¿Carro ligero, camioneta o moto?
- ¿Con qué frecuencia usa su(s) vehículo(s)?
- Hoy en día, es complicado encontrar un estacionamiento disponible. ¿Cuál cree que es la razón por la que sucede esto?
- ¿Cuáles serían las horas donde es más difícil encontrar estacionamientos?
- ¿Cuánto tiempo demora aproximadamente en encontrar estacionamiento?
- ¿Estaría dispuesto a usar como estacionamiento los garajes de las viviendas? ¿Cuáles serían sus motivos?
- ¿Cuáles serían los requisitos mínimos para asegurar la integridad del usuario y su vehículo?
- Si este servicio llega a ser seguro para sus clientes, ¿cree que sería útil para los ciudadanos de las zonas urbanas del Perú? ¿En qué los beneficiaría?
- ¿Confiaría usted en una aplicación móvil que facilite la búsqueda de estos tipos de estacionamientos?

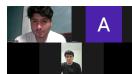
#### Preguntas para el anfitrión o host

- ¿Con cuántos garajes o cocheras cuenta en su hogar?
- ¿Cuál es el tamaño promedio de los garajes en su hogar?
- ¿En qué momentos de su día a día sus garajes se encuentran desocupados?
- ¿Cuenta con algún sistema de seguridad en su cochera?
- ¿Ha llegado a considerar la posibilidad de alquilar sus garajes en algún momento? ¿Por qué?
- ¿Cuáles serían los requisitos mínimos para asegurar la integridad del usuario y su bienes, ya sea inmobiliario o no?
- ¿Le interesaría tener una aplicación móvil que permita promocionar sus garajes como estacionamiento y generar ingresos, sin interrumpir sus actividades diarias?

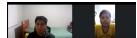
### 2.2.2. Registro de entrevistas

#### Usuarios de parking

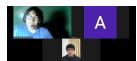
##### Registro de entrevistas

<b>Nombre entrevistado</b>	Walther Cachay
<b>Edad</b>	25 años
<b>Departamento</b>	Lima, Perú
 <b>A</b>	Walter Cachay, un estudiante de ingeniería mecatrónica de 25 años, enfrenta dificultades diarias para encontrar estacionamiento con su auto en Lima, e como Surco o cerca de su universidad, e incluso una o dos horas en el centro, una situación que atribuye al aumento del parque automotor. Aunque rea consideraría esta opción si existieran garantías como comprobantes de empresa, reglas claras y vigilancia 24/7. Walter ve una aplicación para buscar est encontrar lugares disponibles de manera eficiente, siempre y cuando garantice la seguridad del vehículo mediante un sistema de recomendaciones y ub
Duración entrevista 00:00-06:00	<a href="https://upcedupe-my.sharepoint.com/:v/g/personal/u20221c376_upc_edu_pe/EUWwVDeGIMxOirf7aLw_VKoBfk-Q-X3iBSyjW7t8n4CoA?e=czxNW6">https://upcedupe-my.sharepoint.com/:v/g/personal/u20221c376_upc_edu_pe/EUWwVDeGIMxOirf7aLw_VKoBfk-Q-X3iBSyjW7t8n4CoA?e=czxNW6</a>
<b>Nombre entrevistado</b>	Augusto Granados
<b>Edad</b>	20 años
<b>Departamento</b>	Lima Magdalena
 <b>B</b>	Augusto Granados es un joven que cuentan con un carro, pero se le dificulta la posibilidad de encontrar espacio donde puede dejarlo en lugares por los encontrar uno hasta por casi 20 minutos, siento esto una gran pérdida de tiempo para él. El contar con una aplicación que le permita saber en qué luga fue que menciono el que poder valorar estos espacios y conocer sus tamaños serian cruciales para el usuario que busca reservar uno de esos espacios. a gente pueda confiar en que puede dejar ahí sus vehículos.ión
Duración entrevista 00:00-08:57	<a href="https://upcedupe-my.sharepoint.com/:v/g/personal/u202210846_upc_edu_pe/EfwuXreiPk9Ak-akWrpZlVUBYeZe630_bahsL-sYD3QmHw?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOijTdHJIYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOjTaGFyZURpYWxvZy1MaW5rlwiicmVmZXJyYWx">https://upcedupe-my.sharepoint.com/:v/g/personal/u202210846_upc_edu_pe/EfwuXreiPk9Ak-akWrpZlVUBYeZe630_bahsL-sYD3QmHw?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOijTdHJIYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOjTaGFyZURpYWxvZy1MaW5rlwiicmVmZXJyYWx</a>
<b>Nombre</b>	Edu Arturo Antayhua

**entrevistado**

<b>Edad</b>	23 años
<b>Departamento</b>	Lima, San Miguel
 El entrevistado usa su vehículo con frecuencia y enfrenta dificultades para encontrar estacionamiento, especialmente en horas pico. Señala que la conge hallar un espacio. Ve con buenos ojos la idea de una app web o móvil que permita alquilar cocheras de viviendas, siempre que garantice seguridad. Con accesibles, cercanas y seguras.	
Duración entrevista 00:00-06:03	<a href="#">entrevista-edu.mp4</a>
<b>Nombre entrevistado</b>	Fabrizio Buleje
<b>Edad</b>	20 años
<b>Departamento</b>	Santiago de Surco
 Descripción Fabrizio Buleje Alfaro, de 20 años, vive en Surco y estudia Medicina en la Universidad Científica del Sur. Aunque tiene una moto, actualmente se transpo tráfico en Lima y la dificultad para encontrar estacionamiento, especialmente en zonas céntricas como el Centro Cívico, son desafíos frecuentes. Está dis considera útil una aplicación móvil que facilite la búsqueda de estacionamientos seguros y actualizados.	
Duración entrevista 00:00-07:50	<a href="https://upcedupe-my.sharepoint.com/:v/g/personal/u202210697_upc_edu_pe/ETqMzVRTXmNJoKJiOZrbD3cB8QXeMq1xaryOh3PwfP0aRQ?e=OIbOln&amp;nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAIoijTdhJIYw1XZWJBcHAIlCJyZWZlcnJhbFZpZXciOijTaGFyZURpYWxvZy1MaW5rlwiwcm">https://upcedupe-my.sharepoint.com/:v/g/personal/u202210697_upc_edu_pe/ETqMzVRTXmNJoKJiOZrbD3cB8QXeMq1xaryOh3PwfP0aRQ?e=OIbOln&amp;nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAIoijTdhJIYw1XZWJBcHAIlCJyZWZlcnJhbFZpZXciOijTaGFyZURpYWxvZy1MaW5rlwiwcm</a>
<b>Nombre entrevistado</b>	Miguel Carpio
<b>Edad</b>	20
<b>Departamento</b>	Lima, Chorrillos
 Miguel Carpio es un estudiante de 20 años de la carrera de ingeniería de Software en la UPC. Él reside en el distrito de chorrillos y tiene un kia picanto c estacionamiento cuando sale a divertirse con sus amigos debido a que a menudo sale a lugares que él no ha manejado nunca. Por otro lado, afirma que impacto positivo en su rutina diaria, sin embargo, agrega que estos espacios de estacionamiento deben cumplir con requisitos cómo una buena ubicaci	
Duración entrevista 00:00-06:21	<a href="https://upcedupe-my.sharepoint.com/:v/g/personal/u202213143_upc_edu_pe/EcGG9i8Sum1Anzi9nYESXvQB3jfs2Z2vpaNGWXIXTuhMA?e=naqm0O&amp;nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAIoijTdhJIYw1XZWJBcHAIlCJyZWZlcnJhbFZpZXciOijTaGFyZURpYWxvZy1MaW5rlwiwcm">https://upcedupe-my.sharepoint.com/:v/g/personal/u202213143_upc_edu_pe/EcGG9i8Sum1Anzi9nYESXvQB3jfs2Z2vpaNGWXIXTuhMA?e=naqm0O&amp;nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAIoijTdhJIYw1XZWJBcHAIlCJyZWZlcnJhbFZpZXciOijTaGFyZURpYWxvZy1MaW5rlwiwcm</a>

**Anfitriones****Registro de entrevistas**

<b>Nombre entrevistado</b>	Henry Sanchez
<b>Edad</b>	36 años
<b>Departamento</b>	Lima, Perú
 Henry Sánchez, un psicólogo de 36 años residente en San Isidro, posee dos espacios de estacionamiento en su cochera, uno de los cuales, de aproximac Reconociendo la dificultad para encontrar estacionamiento en su zona, Henry ha considerado alquilar este espacio infratilizado, pero la falta de un siste requeriría verificación de identidad y del vehículo, un sistema de calificaciones, alguna forma de garantía o seguro, términos claros, y como medida adic HomeyPark que le permita monetizar este espacio de manera eficiente, segura y sin complicaciones, optimizando sus recursos y generando ingresos pa	
Duración entrevista 00:00-05:00	<a href="https://upcedupe-my.sharepoint.com/:v/g/personal/u20221c376_upc_edu_pe/EW40jWzkohErF8MoKSlyElBy-fT5CM8i_D9YgqGlxFeOA?e=g8VpYy">https://upcedupe-my.sharepoint.com/:v/g/personal/u20221c376_upc_edu_pe/EW40jWzkohErF8MoKSlyElBy-fT5CM8i_D9YgqGlxFeOA?e=g8VpYy</a>
<b>Nombre entrevistado</b>	Carla Cachis Gonzales
<b>Edad</b>	27 años
<b>Departamento</b>	Lima Magdalena
 Carla Cachis es una señorita que cuenta con una propiedad con estacionamiento, pero no con un carro propio, por lo que durante la semana suele estar de HomeyPark le pareció una idea muy buena para poder contar con ingreso extra de los que ya posee sería muy útil, además que como el estacionami mucho énfasis fue que el usuario que buscar dejar su vehículo tiene que dejar documentación oficial que valide quien es la persona, posibles antecedentes este conductor no presenta ser un problema.	
Duración entrevista 00:00-09:13	<a href="https://upcedupe-my.sharepoint.com/:v/g/personal/u202210846_upc_edu_pe/ERvpgREHqlEm_px0TzKoABe-aAytpTQTglcoAfuuAJQ?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAIoijTdhJIYw1XZWJBcHAIlCJyZWZlcnJhbFZpZXciOijTaGFyZURpYWxvZy1MaW5rlwiwcmVmZXJyYWx">https://upcedupe-my.sharepoint.com/:v/g/personal/u202210846_upc_edu_pe/ERvpgREHqlEm_px0TzKoABe-aAytpTQTglcoAfuuAJQ?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAIoijTdhJIYw1XZWJBcHAIlCJyZWZlcnJhbFZpZXciOijTaGFyZURpYWxvZy1MaW5rlwiwcmVmZXJyYWx</a> Entrevista N°5: <a href="https://upcedupe-my.sharepoint.com/:v/g/personal/u20221c376_upc_edu_pe/EbiUMHvfuqtNu7hVWBK2HD8BNYoY4tEjRtNIN3tiEoemmi">https://upcedupe-my.sharepoint.com/:v/g/personal/u20221c376_upc_edu_pe/EbiUMHvfuqtNu7hVWBK2HD8BNYoY4tEjRtNIN3tiEoemmi</a>
<b>Nombre</b>	Rodrigo Tornero Loayza

**entrevistado**

<b>Edad</b>	21 años
<b>Departamento</b>	Lima, Surco
	El entrevistado dispone de un garaje con espacio para un vehículo y ocasionalmente una motocicleta, el cual permanece desocupado por períodos intermitentes. Considera atractivo el estacionamiento y ve en ello una forma práctica de aprovechar un recurso infratilizado.
Duración entrevista	<a href="#">entrevista-rodrigo.mp4</a>
00:00:04:35	
<b>Nombre entrevistado</b>	Carla Cruz
<b>Edad</b>	30 años
<b>Departamento</b>	Cusco, Cusco
	Descripción Carla Jackdel Cruz Palomino, de 30 años, vive en Cusco en un edificio con cocheras por departamento. Cuenta con dos cocheras: una pequeña vehículo, por lo que sus cocheras suelen estar desocupadas y a veces las presta a vecinos o amistades. Los garajes tienen cámaras de seguridad y alarma que sería una buena fuente de ingreso adicional. Señala que para hacerlo se necesitaría información detallada del usuario y respeto por las normas del edificio alquilar sus cocheras de forma práctica y sin afectar sus actividades diarias.
Duración entrevista	<a href="https://upcedupe-my.sharepoint.com/:v/g/personal/u202210697_upc_edu_pe/EVkwUQwS6yRCudGFUP9IYkYBdD-sjfGGK55ZiScS9zMmfg?e=AGO7Kq&amp;nav=eyJyZWZlcnjhbEluZm8iOnsicmVmZXJyYWxBcHAIoiJtdHJIYW1XZWJBcHAIICjyZWZlcnjhbFZpZXciOijTaGFyZURpYWxvZy1MaW5rlwiwcm00:00:04:56">https://upcedupe-my.sharepoint.com/:v/g/personal/u202210697_upc_edu_pe/EVkwUQwS6yRCudGFUP9IYkYBdD-sjfGGK55ZiScS9zMmfg?e=AGO7Kq&amp;nav=eyJyZWZlcnjhbEluZm8iOnsicmVmZXJyYWxBcHAIoiJtdHJIYW1XZWJBcHAIICjyZWZlcnjhbFZpZXciOijTaGFyZURpYWxvZy1MaW5rlwiwcm00:00:04:56</a>
00:00:04:56	
<b>Nombre entrevistado</b>	Joaquín Diaz
<b>Edad</b>	23
<b>Departamento</b>	Lima, Chorrillos
	Joaquín Diaz, joven de 23 años, se desempeña como ingeniero civil. Él cuenta con dos espacios de estacionamiento, el cuál uno alquila. Para Joaquín, Hacer alquileres de manera eficiente y segura ya que actualmente sigue en búsqueda de otra persona que desee alquilar un espacio de estacionamiento. Él resalta situaciones que pondrían su integridad en riesgo.
Duración entrevista	<a href="https://upcedupe-my.sharepoint.com/:v/g/personal/u202213143_upc_edu_pe/EeRv75Hk65JJrqUzWZ-zincBbL95Z3EjtUQh7k0_iPeCYw?e=p7F8u8&amp;nav=eyJyZWZlcnjhbEluZm8iOnsicmVmZXJyYWxBcHAIoiJtdHJIYW1XZWJBcHAIICjyZWZlcnjhbFZpZXciOijTaGFyZURpYWxvZy1MaW5rlwiwcm00:00:03:36">https://upcedupe-my.sharepoint.com/:v/g/personal/u202213143_upc_edu_pe/EeRv75Hk65JJrqUzWZ-zincBbL95Z3EjtUQh7k0_iPeCYw?e=p7F8u8&amp;nav=eyJyZWZlcnjhbEluZm8iOnsicmVmZXJyYWxBcHAIoiJtdHJIYW1XZWJBcHAIICjyZWZlcnjhbFZpZXciOijTaGFyZURpYWxvZy1MaW5rlwiwcm00:00:03:36</a>
00:00:03:36	

## 2.2.3. Análisis de entrevistas

**Segmento 1: Usuarios de parking**

- El 100% de los entrevistados están de acuerdo que la falta o escasez de estacionamientos disponibles se debe a la alta demanda de vehículos motorizados de las zonas urbanas.
- El 100% de los entrevistados presentan dificultades para encontrar estacionamiento principalmente en los tiempos de inicio de trabajo o estudio. Como ejemplo, de 7AM a 10AM o de 2PM a 4PM.
- En la búsqueda de estacionamientos, indican tomar un aproximado de 20 a 30 minutos para conseguir un espacio disponible.
- El 100% del segmento están conformes con la idea de estacionar sus vehículos en garajes de las viviendas, siempre y cuando este cuente con medidas de seguridad.

**Segmento 2: Anfitriones**

- El 100% de los entrevistados mencionan estar interesados en la posibilidad de poner en alquiler su garaje como estacionamiento para afrontar el problema principal, escasez de estacionamientos, y generar ingresos adicionales.
- Un porcentaje de los entrevistados indican que sus espacios de estacionamientos suelen estar desocupados por las mañanas e incluso por las tardes, debido a sus actividades diarias.
- De acuerdo con los entrevistados, al promocionar sus servicios de garaje como estacionamiento a personas desconocidas piden conseguir acceso a la información de estos usuarios y contar con un sistema de calificación para el servicio y el cliente.

## 2.3. Needfinding

### 2.3.1. User Personas

A continuación, se presentan los perfiles de usuario para los dos segmentos objetivo de HomeyPark: conductores y anfitriones.

**User Persona: Usuario de Parking**

PERSONA: Luis Arturo

NAME	MARKET SIZE	TYPE
Luis Arturo	 75 %	Seeker



**Demographic**

♂ Male      36 years  
📍 Lima, Perú  
Single

**Goals**

- Encontrar espacios de estacionamiento de forma rápida y segura.
- Reducir el tiempo que pasa buscando un lugar para estacionar su vehículo.

**Background**

Luis es un conductor habitual en la ciudad de Lima, donde el tráfico y la falta de estacionamientos son problemas diarios. Trabaja como consultor independiente y necesita desplazarse constantemente entre reuniones y oficinas. Luis dedica, en promedio, 30 minutos al día buscando un lugar para estacionar, lo que afecta su productividad y le genera estrés. Busca una solución que le permita encontrar y reservar espacios de estacionamiento de manera eficiente.

**Motivations**

- Optimizar su tiempo evitando largas búsquedas de estacionamiento.
- Garantizar la seguridad de su vehículo al estacionar en garajes confiables.
- Usar una plataforma que le permita conocer la disponibilidad de espacios en tiempo real.

**Frustrations**

- Pérdida de tiempo buscando estacionamiento en zonas congestionadas.
- Inseguridad al dejar su vehículo en cocheras desconocidas.
- Inconsistencia en la disponibilidad de espacios anunciados en la vía pública.

**UXPRESSIA**  
This persona was built in [uxpressia.com](https://uxpressia.com)

**User Persona: Anfitrío**

PERSONA: Jose Perez

NAME	MARKET SIZE	TYPE
<b>Jose Perez</b>	 65 %	Host
	<h3>Goals</h3> <ul style="list-style-type: none"> <li>• Obtener ingresos adicionales alquilando su cochera sin comprometer su seguridad ni la de sus bienes.</li> <li>• Encontrar una manera sencilla y confiable de gestionar el alquiler de su cochera a través de una plataforma digital.</li> </ul>	
<h3>Demographic</h3> <p><input checked="" type="radio"/> Male      45 years</p> <p><input type="checkbox"/> San Borja, Lima, Perú</p> <p><input type="checkbox"/> Married</p>	<h3>Background</h3> <p>Jose es un profesional de 45 años que vive en San Borja, Lima. Posee una cochera que rara vez utiliza, ya que se traslada principalmente en transporte público. Con la idea de generar ingresos extra, ha considerado alquilar su cochera, pero le preocupa la seguridad y la posibilidad de tener problemas con inquilinos desconocidos. Jose busca una plataforma que le ofrezca la tranquilidad de un proceso seguro y fácil para alquilar su espacio sin involucrarse demasiado.</p>	
	<h3>Motivations</h3> <ul style="list-style-type: none"> <li>• Maximizar el uso de su cochera vacía generando ingresos pasivos.</li> <li>• Sentirse seguro al alquilar su espacio a personas verificadas y confiables.</li> <li>• Usar una aplicación que le permita gestionar las reservas sin complicaciones.</li> </ul>	<h3>Frustrations</h3> <ul style="list-style-type: none"> <li>• Usar una aplicación que le permita gestionar las reservas sin complicaciones.</li> <li>• Falta de control sobre quién utiliza su cochera y cómo se cuida durante su uso.</li> </ul>

**UXPRESSIA**

This persona was built in [uxpressia.com](https://uxpressia.com)

### 2.3.2. User Task Matrix

A continuación, se presenta el User Task Matrix para los segmentos de conductores y anfitriones, destacando las tareas que realizan para cumplir sus objetivos en la plataforma HomeyPark.

#### Segmento Jovenes Universitario

Actividades	Luis Arturo	
	Frecuencia	Importancia
Buscar espacios de estacionamiento disponibles en tiempo real	Con frecuencia	Alta
Reservar un espacio de estacionamiento	Con frecuencia	Alta
Evaluar la seguridad del estacionamiento antes de reservar	A veces	Alta
Cancelar una reserva de estacionamiento	Rara vez	Media
Pagar el estacionamiento a través de la plataforma	Con frecuencia	Alta
Calificar el servicio de estacionamiento	A veces	Media
Recibir alertas sobre la disponibilidad de espacios	Con frecuencia	Alta

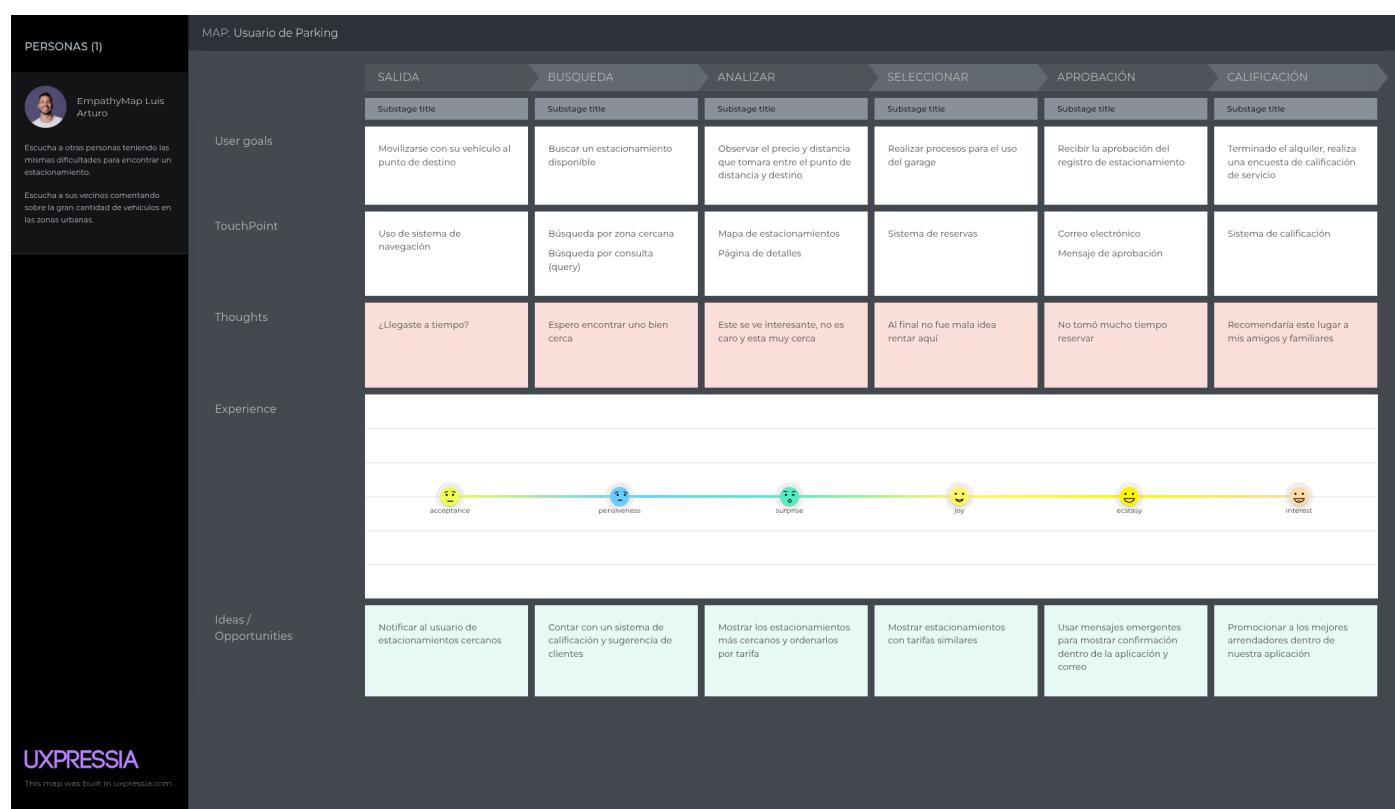
## Segmento Jovenes Freelancer

Actividades	Jose Perez	
	Frecuencia	Importancia
Publicar disponibilidad de su cochera en la plataforma	A veces	Alta
Verificar la identidad de los usuarios que desean alquilar la cochera	A veces	Alta
Confirmar la reserva de la cochera	Con frecuencia	Alta
Recibir pagos por el alquiler de la cochera	Con frecuencia	Alta
Gestionar cancelaciones de reservas	Rara vez	Media
Comunicarse con los inquilinos a través de la plataforma	A veces	Media
Actualizar la información de su cochera en la plataforma	Rara vez	Media

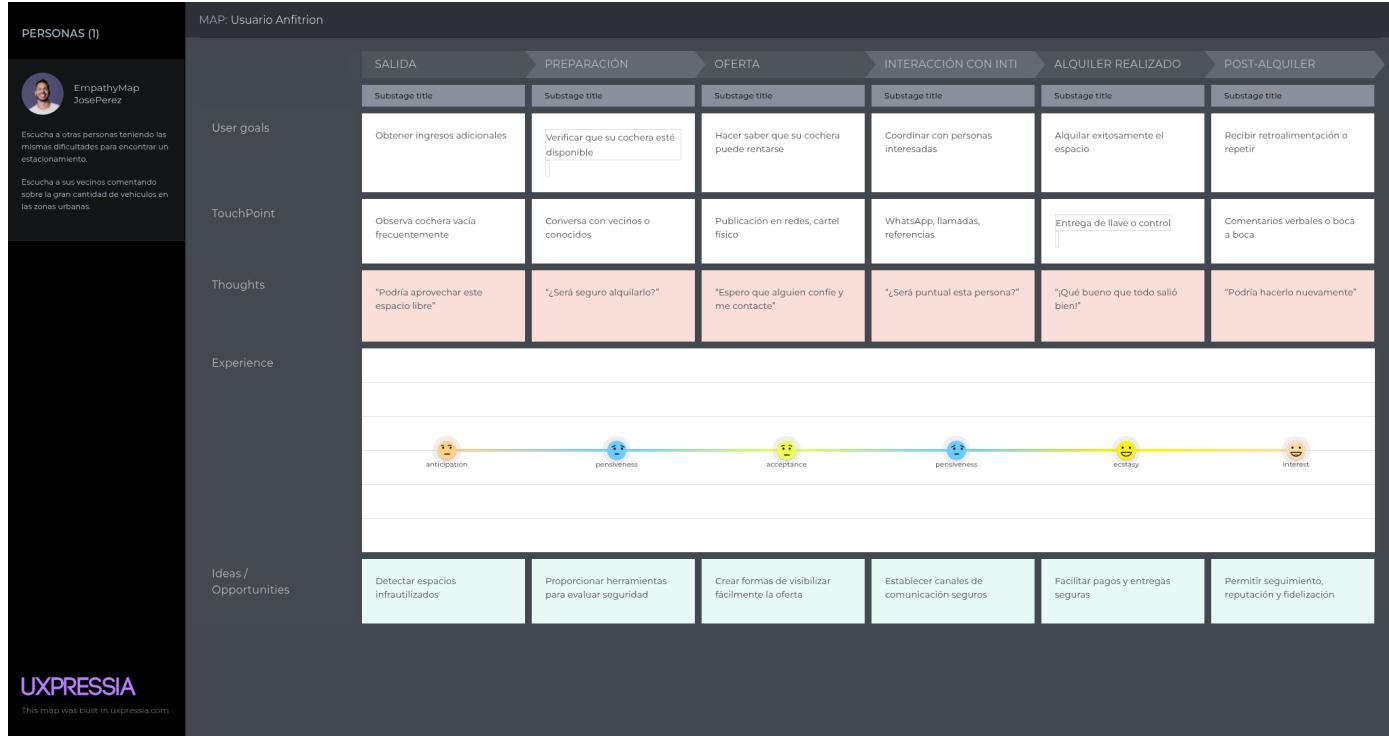
### 2.3.3. User Journey Mapping

A continuación, se presentan los User Journey Mapping para los segmentos de conductores y anfitriones, ilustrando sus experiencias al interactuar con HomeyPark.

## Segmento Usuario de Parking



## Segmento Anfitrion



### 2.3.4. Empathy Mapping

Aquí se muestran los Empathy Map para los segmentos de jóvenes universitarios y freelancers, ofreciendo una visión detallada de sus pensamientos, sentimientos y necesidades.

#### Usuario de Parking

PERSONA: Luis Arturo

**1.WHO are we empathizing with?**

Luis Arturo es una persona de la ciudad de Lima con bastantes actividades en diferentes sitios.

**7.What do they THINK and FEEL?**

*“Piensa y siente que debe haber más formas de estacionamiento en estas zonas.”*

**2.What do they need to DO?**

Encontrar un estacionamiento libre, seguro y de preferencia cercano al punto de trabajo, gimnasio, etc.

**6.What do they HEAR?**

- Escucha a otras personas teniendo las mismas dificultades para encontrar un estacionamiento.
- Escucha a sus vecinos comentando sobre la gran cantidad de vehículos en las zonas urbanas.

**3.What do they SEE?**

- Observa una gran demanda de vehículos en las ciudades urbanas.
- Observa congestiones vehiculares muy seguido.
- Observa la cantidad de estacionamientos incrementa pero no llega a ser suficiente.

**5.What do they DO?**

Tiene que realizar una búsqueda exhaustiva donde toma largos períodos de tiempo durante todos los días con la esperanza de conseguir un estacionamiento cercano.

**PAINS**

- Tiempo perdido por búsquedas
- Ansiedad por no encontrar puestos libres
- Estrés por tráficos continuos
- Dejar su coche en la calle (inseguridad)

**GAINS**

- Necesidad de más estacionamientos
- Sentir seguridad, tanto vehículo como usuario
- Tarifas no elevadas de manera exagerada

**4.What do they SAY?**

*“Tienen que incrementar la cantidad de estacionamientos en estas zonas. A este ritmo los vehículos los dejaremos en los carriles.”*

**UXPRESSIA**This persona was built in [uxpressia.com](https://uxpressia.com)**Anfitrion**

PERSONA: Jose Perez

**1.WHO are we empathizing with?**

Jose Perez es una persona con garaje que usualmente suele estar desocupado.

**7.What do they THINK and FEEL?**

“  
Piensa en cómo poder promocionar su servicio sin tener la inseguridad de resultar afectado negativamente.”

**2.What do they need to DO?**

Encontrar una manera de aprovechar su garaje cuando este desocupado.

**6.What do they HEAR?**

Escucha que muchas personas estarían dispuestas a pagar por tener un espacio para su vehículo, incluso si fuera de una vivienda.

**3.What do they SEE?**

Él ve que hay más vehículos en las ciudades y menos estacionamientos libres.

**5.What do they DO?**

Se encuentra buscando cómo promocionar su garaje como servicio de estacionamiento, publicando posts y repartiendo folletos.

**PAINS**

Miedo a alquilar una cochera a personas extrañas.

**GAINS**

Ingresos adicionales.

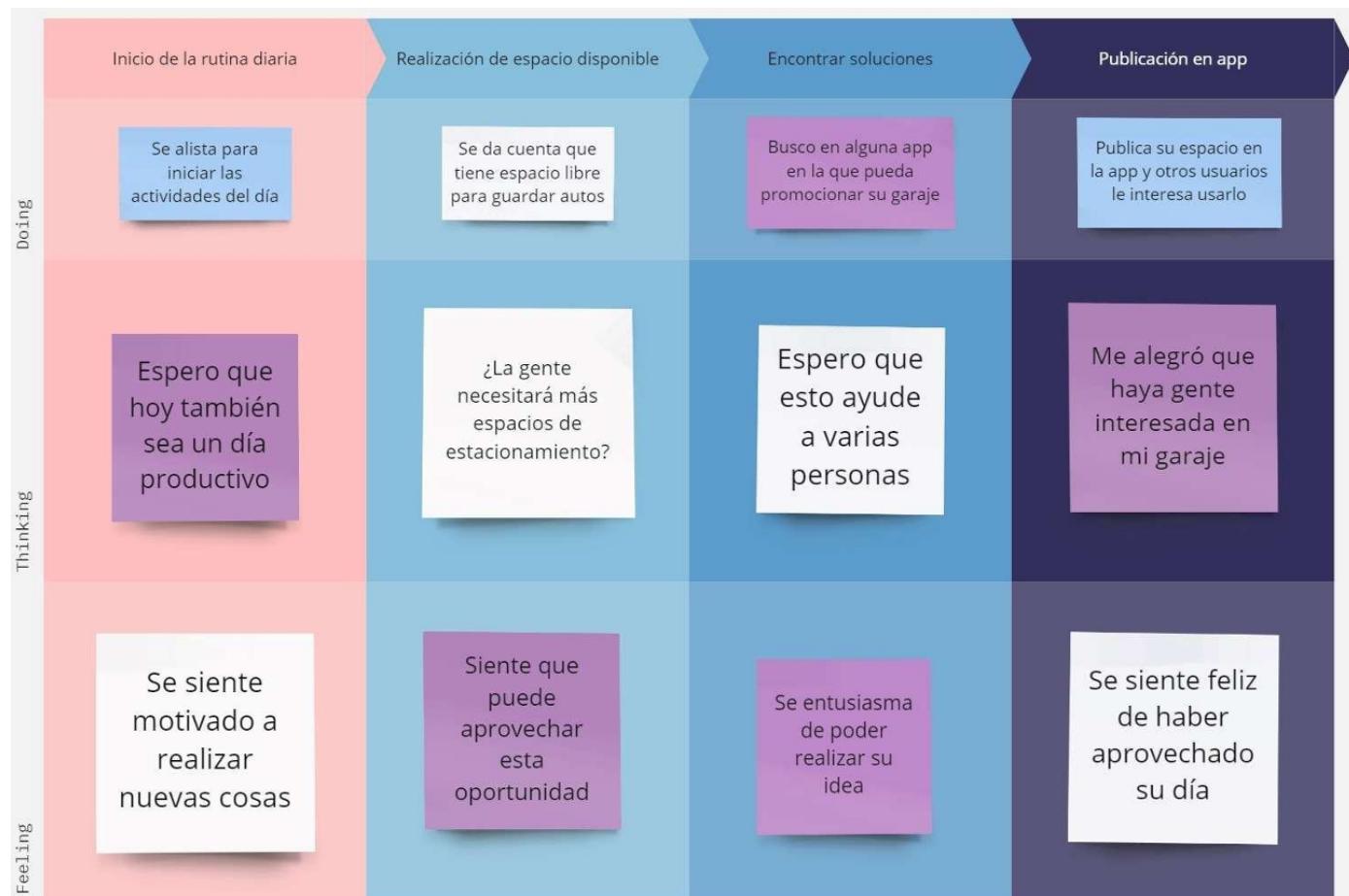
**4.What do they SAY?**

“  
Podría aprovechar en alquilar mi garaje para generar un ingreso adicional.”

**UXPRESSIA**This persona was built in [uxpressia.com](https://uxpressia.com)**2.3.5. As-is Scenario Mapping**

El As-Is Scenario Mapping describe la experiencia actual de los conductores al buscar estacionamiento y los propietarios de cocheras al intentar rentarlas. Los conductores pierden tiempo y combustible buscando espacios disponibles en zonas congestionadas, mientras que los propietarios tienen garajes con frecuencia baja o nula de uso que podrían generar ingresos adicionales. Este mapeo identifica los puntos de fricción y oportunidades de mejora que HomeyPark busca resolver, como la falta de visibilidad de espacios disponibles, inquietudes sobre seguridad y la ausencia de un sistema estructurado para conectar ambos segmentos del mercado de manera eficiente.

**Usuario de Parking**

**Anfitrion****2.4. Ubiquitous Language(Cambiar)**

El Ubiquitous Language es un conjunto de términos y conceptos que comparten los desarrolladores y los expertos del dominio (en este caso, los usuarios de HomeyPark) para describir el sistema. Su objetivo es reducir la ambigüedad y facilitar la comunicación entre todos los involucrados en el proyecto. A continuación, se presenta un Ubiquitous Language inicial para HomeyPark:

- **Usuario de parking:** Persona que busca una solución para su necesidad de encontrar un estacionamiento en entornos urbanos para su vehículo.
- **Anfitrión:** Propietario de una vivienda que cuenta con una cochera privada y busca obtener provecho económico de ella.
- **Estacionamiento:** Espacio donde un vehículo puede ser aparcado. Dentro de la aplicación, un estacionamiento tendrá atributos como ubicación, precio, disponibilidad, tamaño, reseñas, horarios, teléfono de servicio y descripción.
- **Reserva:** Acción de asegurar un espacio de estacionamiento para un tiempo determinado. Los usuarios de parking podrán realizar reservas, y los anfitriones podrán confirmarlas.
- **Cochera privada:** Un garaje o espacio de estacionamiento perteneciente a una vivienda particular, que un anfitrión puede ofrecer en alquiler.
- **Búsqueda:** Funcionalidad que permite a los usuarios de parking encontrar estacionamientos disponibles según diferentes criterios como ubicación, precio, y disponibilidad.
- **Pago:** Proceso mediante el cual el usuario de parking abona el costo de la reserva del estacionamiento.
- **Calificación:** Sistema que permite a los usuarios de parking evaluar el servicio de estacionamiento y a los anfitriones.
- **Reseña:** Comentario o valoración escrita por un usuario sobre su experiencia con un estacionamiento o un anfitrión.

## Capítulo III: Requirements Specification

### 3.1. To-Be Scenario Mapping

A continuación, mostraremos la experiencia ideal de los usuarios al usar la aplicación HomeyPark para encontrar y reservar espacios de estacionamiento. Los conductores pueden ubicar rápidamente lugares disponibles cerca de su destino, mientras que los propietarios de cocheras pueden rentarlas de forma segura cuando no las utilizan.

#### Usuario de Parking

Phases	Registro en la plataforma	Exploración de opciones de parking	Reserva de espacio	Llegada y uso del espacio de parking	Retraolimentación y recomendaciones
Doing	El usuario accede al sitio web o a la aplicación de HomeyPark	Navega por las diferentes categorías de estacionamientos disponibles en HomeyPark.	Selecciona una cochera que se ajuste a sus preferencias y necesidades.	Se dirige a la ubicación del parking y sigue las instrucciones proporcionadas por HomeyPark.	Después de utilizar el servicio, proporciona comentarios y calificaciones sobre la experiencia.
	Completa el formulario de registro con su información personal y de contacto.	Lee descripciones, revisa imágenes de las cocheras y precios.	Escoge la fecha y hora de uso, y confirma la reserva a través de HomeyPark.	Utiliza el espacio reservado para estacionar su vehículo.	Comparte sus experiencias y recomienda HomeyPark a amigos y familiares.
Thinking	¿Será fácil y seguro registrarme en esta plataforma?	¿Qué tipo de espacio de parking necesito según mi vehículo?	¿Es esta opción de parking la más conveniente en términos de ubicación y costo?	¿Qué tan fácil será encontrar el lugar y acceder a la cochera?	¿Cómo puedo expresar adecuadamente mi satisfacción o sugerencias sobre la experiencia?
	¿Qué tipo de opciones de parking ofrece HomeyPark y en qué zonas?	¿Cuáles son las características de seguridad de cada opción?	¿Cómo será la experiencia al llegar y acceder al espacio reservado?	¿El espacio cumple con las expectativas de seguridad y comodidad?	¿Qué aspectos destaco al recomendar HomeyPark a otras personas?
Feeling	Satisfacción al encontrar un proceso de registro intuitivo y rápido.	Curiosidad por conocer más detalles sobre las ubicaciones y características de los espacios disponibles.	Anticipación por tener un lugar seguro y conveniente donde dejar su vehículo.	Alegría al encontrar un lugar seguro y conveniente para estacionar.	Plenitud al compartir experiencias positivas y fomentar la participación en la plataforma.
	Confianza en la seguridad y reputación de HomeyPark como plataforma de parking.	Satisfacción al encontrar una opción que se ajuste a sus necesidades de parking.	Tranquilidad al haber asegurado una reserva según sus horarios y necesidades.	Gratitud por la facilidad del proceso de acceso y uso del espacio.	Compromiso por contribuir a mejorar la calidad y la oferta de espacios de parking en HomeyPark.

#### Anfitrión

Phases	Registro en la plataforma	Publicación de la cochera	Gestión de reservas	Recepción y soporte al usuario	Retraoimentación y recomendaciones
Doing	El anfitrión accede al sitio web o a la aplicación de HomeyPark.	Navega por las opciones de publicación y selecciona el plan más adecuado para su cochera.	Recibe notificaciones de reservas y revisa la información del usuario que alquilará su cochera.	Recibe al usuario, le da acceso a la cochera y le proporciona cualquier información necesaria.	Proporciona comentarios sobre la experiencia del usuario y recibe feedback de su parte.
	Completa el formulario de registro con su información personal y detalles de la cochera que desea listar.	Proporciona una descripción detallada, sube fotos y establece el precio de la cochera.	Confirma la reserva y se prepara para la llegada del usuario.	Está disponible para resolver cualquier duda o inconveniente que el usuario pueda tener durante su estancia.	Revisa las calificaciones y ajusta su oferta según las recomendaciones recibidas.
Thinking	¿Será fácil y seguro registrar mi cochera en esta plataforma?	¿Qué detalles debo destacar para hacer mi cochera más atractiva?	¿Cómo puedo asegurarme de que mi cochera esté lista y en condiciones para el usuario?	¿Qué puedo hacer para que el usuario tenga una experiencia positiva?	¿Cómo puedo mejorar la experiencia para futuros usuarios?
	¿Qué tan demandado será mi espacio de parking en HomeyPark?	¿Cuál es el precio competitivo para mi cochera en comparación con otras en la zona?	¿Cómo será la comunicación con el usuario durante el proceso?	¿Cómo asegurarme de que el proceso de entrada y salida sea fluido?	¿Qué aspectos positivos destacan en los comentarios recibidos?
Feeling	Satisfacción al encontrar un proceso de registro intuitivo y rápido.	Emoción por la posibilidad de generar ingresos adicionales al listar su cochera.	Tranquilidad al saber que el proceso de reserva es claro y transparente.	Orgullo al poder ofrecer un espacio bien mantenido y seguro.	Plenitud al recibir buenas calificaciones y saber que su cochera es bien valorada.
	Confianza en que HomeyPark es una plataforma segura y confiable para listar mi espacio de parking.	Satisfacción al ver su cochera publicada de manera profesional y atractiva.	Anticipación por recibir al usuario y proporcionar un buen servicio.	Satisfacción al ver que el usuario está contento con el servicio.	Compromiso por seguir mejorando su servicio y mantener una alta reputación en HomeyPark.

### 3.2. User Stories

Las User Stories y Épicas permiten descomponer y organizar las funcionalidades del sistema desde la perspectiva del usuario. Las épicas agrupan funcionalidades clave, mientras que las user stories detallan necesidades específicas que guían el desarrollo de la aplicación.

#### Epics

Epic ID	Nombre	Descripción
EP1	Landing Page	Pantalla principal de la aplicación donde se presenta la propuesta de valor, beneficios y navegación general.
EP2	Autenticación y sesión	Funcionalidades relacionadas al registro, inicio y cierre de sesión de los usuarios.
EP3	Gestión de estacionamientos	Visualización, búsqueda, publicación y edición de cocheras disponibles para arrendar.
EP4	Gestión de reservas	Permitir que los usuarios arrendatarios puedan reservar, cancelar o consultar reservas.
EP5	Seguimiento de servicios	Acceso a historial, servicios en curso y próximos tanto para arrendadores como arrendatarios.
EP6	Control de reservas activas	Proceso de aprobación, finalización y validación de reservas realizadas.
EP7	Gestión de vehículos	Permitir que los usuarios registren, editen o eliminen los vehículos vinculados a su cuenta.
EP8	Perfil del usuario	Visualización y actualización de la información personal del usuario.

#### User Stories

Epic / Story ID	Título	Descripción	Relacionado con (Epic ID)	Criterios de Aceptación (Given-When-Then)
US-01	Ver portada	Como usuario, quiero ver una portada para entender de qué trata la plataforma	EP1	<ul style="list-style-type: none"> <li>1. Dado que el usuario visita la página de inicio, cuando la página carga completamente, entonces ve la portada con título y subtítulo.</li> <li>2. Dado que el usuario ingresa desde dispositivos diversos, cuando se carga la página, entonces la portada se adapta correctamente a cada pantalla.</li> </ul>

Epic / Story ID	Título	Descripción	Relacionado con (Epic ID)	Criterios de Aceptación (Given-When-Then)
US-02	Ver beneficios	Como usuario, quiero ver los beneficios para entender por qué me conviene	EP1	<ul style="list-style-type: none"> <li>1. Dado que el usuario explora la landing page, cuando llega a la sección de beneficios, entonces se muestran al menos tres beneficios destacados.</li> <li>2. Dado que el usuario consulta la sección de beneficios, cuando se presenta el contenido, entonces la información es clara y fácilmente comprensible.</li> </ul>
US-03	Ver ejemplos de servicios	Como usuario, quiero ver ejemplos de servicios para imaginar su uso	EP1	<ul style="list-style-type: none"> <li>1. Dado que el usuario accede a la sección de ejemplos, cuando la página carga, entonces se muestran al menos tres tarjetas descriptivas.</li> <li>2. Dado que el usuario interactúa con las tarjetas, cuando selecciona una, entonces se muestra información ampliada sobre el servicio.</li> </ul>
US-04	Ver precios	Como usuario, quiero ver una sección de precios para conocer los planes	EP1	<ul style="list-style-type: none"> <li>1. Dado que el usuario busca información de precios, cuando accede a la sección, entonces se muestran los planes organizados en una tabla o listado.</li> <li>2. Dado que el usuario compara planes, cuando revisa la sección de precios, entonces se detallan características y valores de cada plan.</li> </ul>
US-05	Ver testimonios	Como usuario, quiero ver la sección de testimonios para una mayor referencia	EP1	<ul style="list-style-type: none"> <li>1. Dado que el usuario accede a la sección de testimonios, cuando la página carga, entonces se visualizan al menos tres testimonios con nombre e imagen.</li> <li>2. Dado que el usuario desplaza la vista, cuando revisa la sección, entonces los testimonios se muestran en un formato legible y organizado.</li> </ul>
US-06	Ver barra de navegación	Como usuario, quiero contar con una barra de navegación para facilitar la búsqueda	EP1	<ul style="list-style-type: none"> <li>1. Dado que el usuario se encuentra en cualquier sección, cuando observa la interfaz, entonces la barra de navegación es visible.</li> <li>2. Dado que el usuario interactúa con la barra, cuando selecciona un enlace, entonces es redirigido sin errores a la sección correspondiente.</li> </ul>
US-07	Ver equipo	Como usuario, quiero ver la sección del equipo para conocer su propósito o historia	EP1	<ul style="list-style-type: none"> <li>1. Dado que el usuario quiere conocer al equipo, cuando ingresa a esa sección, entonces se muestran fotos, nombres y roles de cada miembro.</li> <li>2. Dado que el usuario revisa la historia del equipo, cuando visualiza la sección, entonces se aprecia una breve descripción o propósito de cada integrante.</li> </ul>
US-08	Ver footer	Como usuario, quiero un footer para acceder a información adicional	EP1	<ul style="list-style-type: none"> <li>1. Dado que el usuario llega al final de la página, cuando la misma carga, entonces el footer es visible con enlaces a contacto y políticas.</li> <li>2. Dado que el usuario explora el footer, cuando revisa su contenido, entonces la información adicional se presenta de forma organizada.</li> </ul>
US-09	Registrarse	Como usuario, quiero registrarme con correo, contraseña y datos de perfil para crear una cuenta	EP2	<ul style="list-style-type: none"> <li>1. Dado que el usuario quiere crear una cuenta, cuando completa todos los campos requeridos, entonces se crea la cuenta y se notifica al usuario.</li> <li>2. Dado que el usuario ingresa un correo duplicado, cuando intenta registrarse, entonces se muestra un mensaje de error pertinente.</li> </ul>
US-10	Iniciar sesión	Como usuario, quiero iniciar sesión con correo y contraseña para acceder a mi cuenta	EP2	<ul style="list-style-type: none"> <li>1. Dado que el usuario tiene una cuenta, cuando introduce credenciales correctas, entonces inicia sesión y es redirigido al dashboard.</li> <li>2. Dado que el usuario introduce credenciales erróneas, cuando intenta iniciar sesión, entonces se muestra un mensaje de error sin permitir el acceso.</li> </ul>
US-11	Cerrar sesión	Como usuario, quiero poder cerrar sesión para proteger mi información	EP2	<ul style="list-style-type: none"> <li>1. Dado que el usuario está autenticado, cuando selecciona la opción de cerrar sesión, entonces se termina la sesión y se redirige al login.</li> <li>2. Dado que el usuario cierra sesión, cuando intenta acceder a una sección protegida, entonces se redirige al login.</li> </ul>
US-12	Búsqueda de estacionamientos por dirección	Como arrendatario quiero ingresar una dirección para encontrar los espacios de estacionamiento en el mapa	EP3	<ul style="list-style-type: none"> <li>1. Dado que el usuario ingresa una dirección válida, cuando presiona buscar, entonces se muestran los estacionamientos disponibles en esa zona.</li> <li>2. Dado que el usuario ingresa una dirección no válida, cuando presiona buscar, entonces se muestra un mensaje de error.</li> </ul>

Epic / Story ID	Título	Descripción	Relacionado con (Epic ID)	Criterios de Aceptación (Given-When-Then)
US-13	Visualizar estacionamientos en mapa	Como arrendatario, quiero visualizar los estacionamientos en un mapa para tener una mayor referencia	EP3	1. Dado que hay estacionamientos disponibles, cuando se carga el mapa, entonces se muestran marcadores en la ubicación correspondiente. 2. Dado que no hay estacionamientos, cuando se carga el mapa, entonces se muestra un mensaje indicando que no hay resultados.
US-14	Ver espacios cercanos	Como arrendatario, quiero ver los espacios cercanos en modo de lista para ver todos los espacios en mi zona	EP3	1. Dado que el usuario permite el acceso a su ubicación, cuando entra a la sección de lista, entonces se muestran los espacios cercanos. 2. Dado que el usuario no da permisos de ubicación, cuando entra a la sección de lista, entonces se muestra una advertencia.
US-15	Ver detalle de estacionamiento	Como arrendatario, quiero visualizar los detalles del estacionamiento para encontrar el mejor puesto	EP3	1. Dado que el usuario selecciona un estacionamiento, cuando se muestra el detalle, entonces debe contener la información completa: dirección, precio, horario y disponibilidad. 2. Dado que no se puede cargar el detalle, cuando el usuario selecciona, entonces se muestra un error.
US-16	Ver calificación del estacionamiento	Como arrendatario, quiero visualizar la calificación del estacionamiento para evitar fraudes	EP3	1. Dado que el estacionamiento tiene calificaciones, cuando el usuario visualiza el detalle, entonces se muestra un promedio de calificaciones con comentarios. 2. Dado que no hay calificaciones, cuando se visualiza el detalle, entonces se muestra un mensaje indicando "Sin reseñas aún".
US-17	Vista previa del lugar	Como arrendatario, quiero visualizar el entorno del estacionamiento para reconocer el lugar	EP3	1. Dado que el estacionamiento tiene imágenes, cuando el usuario entra al detalle, entonces se muestran las imágenes del entorno. 2. Dado que no hay imágenes disponibles, cuando se accede al detalle, entonces se muestra un mensaje informativo.
US-18	Registrar cochera	Como arrendador, quiero registrar mi cochera como estacionamiento para generar ingresos	EP3	1. Dado que el arrendador llena todos los campos requeridos, cuando presiona registrar, entonces la cochera se guarda exitosamente. 2. Dado que falta información, cuando intenta registrar, entonces se muestra un mensaje de validación.
US-19	Ver mis cocheras registradas	Como arrendador, quiero visualizar mis cocheras registradas para gestionarlas	EP3	1. Dado que el arrendador tiene cocheras registradas, cuando accede a la sección, entonces se muestran en una lista con sus detalles. 2. Dado que no tiene cocheras, cuando accede, entonces se muestra un mensaje indicando que aún no ha registrado ninguna.
US-20	Actualizar cochera	Como arrendador, quiero modificar mis cocheras registradas	EP3	1. Dado que el arrendador selecciona una cochera, cuando modifica los datos y guarda, entonces los cambios se actualizan correctamente. 2. Dado que no llena un campo obligatorio, cuando intenta guardar, entonces se muestra un mensaje de error.
US-21	Eliminar cochera	Como arrendador, quiero eliminar mi cochera	EP3	1. Dado que el arrendador desea eliminar una cochera, cuando confirma la acción, entonces se elimina de la lista. 2. Dado que cancela la eliminación, cuando se le pide confirmación, entonces no se ejecuta ninguna acción.
US-22	Solicitar reserva	Como arrendatario, quiero solicitar la reserva de un estacionamiento	EP4	1. Dado que el usuario llena la información requerida, cuando presiona reservar, entonces se envía la solicitud al arrendador. 2. Dado que la información está incompleta, cuando intenta reservar, entonces se muestra una advertencia.
US-23	Seleccionar horario de reserva	Como arrendatario, quiero indicar el horario en la solicitud	EP4	1. Dado que el usuario elige un rango de tiempo, cuando lo confirma, entonces el horario queda registrado en la reserva. 2. Dado que no selecciona un horario, cuando intenta continuar, entonces se muestra un mensaje de error.
US-24	Seleccionar vehículo para reserva	Como arrendatario, quiero indicar el vehículo a usar	EP4	1. Dado que el usuario tiene vehículos registrados, cuando selecciona uno, entonces se asocia con la reserva. 2. Dado que no tiene vehículos, cuando accede a la selección, entonces se muestra una opción para registrar uno.
US-25	Cancelar solicitud de reserva	Como arrendatario, quiero cancelar la reserva	EP4	1. Dado que el usuario ya reservó, cuando presiona cancelar y confirma, entonces se elimina la solicitud. 2. Dado que decide no cancelar, cuando rechaza la confirmación, entonces no se realiza ninguna acción.

Epic / Story ID	Título	Descripción	Relacionado con (Epic ID)	Criterios de Aceptación (Given-When-Then)
US-26	Rechazar solicitud de reserva	Como arrendador, quiero rechazar las solicitudes de reserva	EP4	1. Dado que recibe una solicitud, cuando presiona rechazar y confirma, entonces el arrendatario recibe una notificación de rechazo.
US-27	Aprobar solicitud de reserva	Como arrendador, quiero aprobar las solicitudes de reserva	EP4	1. Dado que recibe una solicitud, cuando presiona aprobar, entonces el arrendatario recibe la confirmación y los detalles de la reserva.
US-28	Ver próximas reservas	Como arrendatario, quiero visualizar mis próximas reservas	EP5	1. Dado que el usuario tiene reservas futuras, cuando accede a la sección, entonces se muestran en orden cronológico. 2. Dado que no tiene reservas, cuando entra a la sección, entonces se muestra un mensaje informativo.
US-29	Ver reservas en curso	Como arrendatario, quiero visualizar mis reservas en curso	EP5	1. Dado que el usuario tiene reservas activas, cuando entra a la sección, entonces se listan las reservas con su estado. 2. Dado que no tiene reservas en curso, cuando accede, entonces se muestra un mensaje correspondiente.
US-30	Ver historial de reservas	Como arrendatario, quiero visualizar mi historial de reservas	EP5	1. Dado que el usuario ha realizado reservas, cuando entra a historial, entonces se muestran las reservas pasadas con fecha y lugar. 2. Dado que no hay historial, se muestra un mensaje indicando que aún no ha reservado.
US-31	Ver detalle de reserva	Como arrendatario, quiero visualizar el detalle de mi reserva	EP5	1. Dado que el usuario selecciona una reserva, cuando se accede al detalle, entonces se muestra el horario, cochera, estado y datos del propietario.
US-32	Ver próximos servicios	Como arrendador, quiero visualizar mis próximos servicios	EP5	1. Dado que hay servicios próximos, cuando accede a la sección, entonces se listan con su fecha y datos del arrendatario. 2. Dado que no hay próximos servicios, se muestra un mensaje indicándolo.
US-33	Ver servicios en curso	Como arrendador, quiero visualizar mis servicios en curso	EP5	1. Dado que hay reservas activas, cuando accede a la sección, entonces se muestran con los horarios y placas de los vehículos.
US-34	Ver historial de servicios	Como arrendador, quiero visualizar mi historial de servicios realizados	EP5	1. Dado que ha tenido reservas anteriores, cuando entra al historial, entonces se muestran las fechas, cocheras y arrendatarios.
US-35	Ver detalle del servicio	Como arrendador, quiero visualizar el detalle mi servicio	EP5	1. Dado que el arrendador selecciona un servicio, cuando accede al detalle, entonces se muestra toda la información relevante del mismo.
US-36	Iniciar proceso de reserva	Como arrendador, quiero iniciar el proceso de reserva	EP6	1. Dado que un arrendatario ha enviado una solicitud, cuando el arrendador la aprueba, entonces inicia el proceso y se notifica al arrendatario.
US-37	Finalizar reserva	Como arrendatario, quiero dar por finalizado la reserva	EP6	1. Dado que el usuario ha utilizado el estacionamiento, cuando presiona finalizar, entonces el sistema actualiza el estado a 'finalizado'.
US-38	Subir comprobante de pago	Como arrendatario, quiero subir mi comprobante de pago	EP6	1. Dado que el usuario ha efectuado el pago, cuando sube el comprobante, entonces se almacena y se notifica al arrendador para su validación. 2. Dado que el comprobante no es válido, cuando se intenta subir, entonces se muestra un mensaje de error.
US-39	Ver vehículos registrados	Como usuario, quiero ver mis vehículos registrados	EP7	1. Dado que el usuario tiene vehículos registrados, cuando entra a la sección, entonces se muestran en formato de lista. 2. Dado que no tiene vehículos registrados, se muestra un mensaje indicándolo.
US-40	Añadir vehículo	Como usuario, quiero añadir mi vehículo en la aplicación	EP7	1. Dado que el usuario completa los datos del vehículo, cuando presiona guardar, entonces se registra y aparece en la lista de vehículos. 2. Dado que falta un campo obligatorio, el sistema muestra un mensaje de error.
US-41	Editar vehículo	Como usuario, quiero editar la información de mi vehículo	EP7	1. Dado que el usuario edita los datos del vehículo, cuando guarda, entonces se actualiza correctamente.
US-42	Eliminar vehículo	Como usuario, quiero eliminar mi vehículo en la aplicación	EP7	1. Dado que el usuario desea eliminar un vehículo, cuando confirma la acción, entonces se elimina de la lista. 2. Dado que cancela la acción, entonces no se realiza ningún cambio.
US-43	Ver perfil	Como usuario, quiero visualizar mi perfil	EP8	1. Dado que el usuario accede a la sección de perfil, cuando carga la vista, entonces se muestran sus datos personales actualizados.

Epic / Story ID	Título	Descripción	Relacionado con (Epic ID)	Criterios de Aceptación (Given-When-Then)
US-44	Actualizar perfil	Como usuario, quiero editar mi perfil	EP8	1. Dado que el usuario edita sus datos, cuando guarda los cambios, entonces se actualiza la información correctamente. 2. Dado que falta algún campo obligatorio, entonces se muestra un mensaje de error.
TS-01	Implementar sistema de navegación	Como desarrollador, quiero configurar un sistema de rutas nombradas para moverme entre las diferentes vistas (registro, login, listado, mapa, perfil, etc.).	EP2	1. Dado que el usuario interactúa con botones de navegación, cuando los presiona, entonces es redirigido a la pantalla correspondiente. 2. Dado que se ingresa una ruta no válida, cuando se intenta acceder, entonces se muestra una pantalla de error manejada.
TS-02	Configurar control de estado global	Como desarrollador, quiero implementar un sistema de manejo de estado para mantener sincronizados datos entre pantallas (usuario, vehículos, reservas, etc.).	EP2	1. Dado que se actualiza un estado compartido, cuando un componente lo modifica, entonces los demás componentes reflejan el cambio. 2. Dado que se cierra sesión, cuando el usuario regresa a una vista anterior, entonces los datos deben estar limpios.
TS-03	Implementar validación de formularios	Como desarrollador, quiero agregar validaciones a los formularios (login, registro, creación de cochera) para asegurar que los datos ingresados sean correctos.	EP2	1. Dado que el usuario completa un formulario, cuando un campo requerido está vacío, entonces se muestra un mensaje de validación. 2. Dado que todos los campos son válidos, cuando se envía el formulario, entonces continúa el proceso sin errores.
TS-04	Desarrollar lógica de geolocalización	Como desarrollador, quiero acceder a la ubicación actual del usuario para mostrar estacionamientos cercanos en tiempo real.	EP3	1. Dado que el usuario permite acceso a su ubicación, cuando entra a la vista de búsqueda, entonces se obtiene su ubicación actual. 2. Dado que el usuario deniega el permiso, cuando entra a la vista, entonces se muestra un mensaje alternativo o una vista de fallback.
TS-05	Integrar Google Maps en la vista de búsqueda	Como desarrollador, quiero integrar un mapa para mostrar los estacionamientos disponibles según la ubicación del usuario.	EP3	1. Dado que hay estacionamientos disponibles, cuando se carga el mapa, entonces se muestran los marcadores. 2. Dado que no hay estacionamientos disponibles, cuando se carga el mapa, entonces se muestra un mensaje indicándolo.
TS-06	Implementar CRUD de cocheras	Como desarrollador, quiero desarrollar las funciones para crear, leer, actualizar y eliminar cocheras registradas por el arrendador.	EP3	1. Dado que el usuario arrendador accede a la sección de cocheras, cuando añade una nueva, entonces esta se guarda correctamente. 2. Dado que edita o elimina una cochera, entonces los cambios se reflejan inmediatamente en su listado.
TS-07	Implementar CRUD de vehículos	Como desarrollador, quiero permitir que los usuarios puedan añadir, visualizar, editar y eliminar la información de sus vehículos.	EP7	1. Dado que el usuario desea registrar su vehículo, cuando completa los campos obligatorios, entonces se guarda y aparece en la lista. 2. Dado que el usuario edita o elimina un vehículo, entonces el sistema refleja los cambios correctamente.
TS-08	Implementar CRUD de reservas	Como desarrollador, quiero desarrollar la funcionalidad para que los arrendatarios puedan registrar, cancelar y consultar sus reservas, y los arrendadores puedan gestionarlas.	EP4	1. Dado que el usuario crea una reserva, cuando se valida la información, entonces esta se registra en el sistema. 2. Dado que el usuario desea cancelar una reserva, cuando confirma la acción, entonces se elimina correctamente.
TS-09	CRUD de perfil de usuario	Como desarrollador, quiero implementar la edición y visualización del perfil del usuario para que pueda modificar sus datos personales.	EP8	1. Dado que el usuario accede a su perfil, cuando actualiza su información, entonces esta se guarda correctamente. 2. Dado que se dejan campos requeridos vacíos, entonces se muestra una advertencia de validación.

### 3.3. Product Backlog

El Product Backlog es una lista priorizada de funcionalidades, mejoras y requerimientos que guían el desarrollo del producto. Incluye todas las épicas, user stories y technical stories, sirviendo como base para la planificación y evolución continua de la aplicación.

Orden	ID	Título	Descripción	Épica	Story Points
1	TS-05	Integrar Google Maps en la vista de búsqueda	Como desarrollador, quiero integrar Google Maps para visualizar estacionamientos disponibles.	EP3	8
2	TS-04	Desarrollar lógica de geolocalización	Como desarrollador, quiero acceder a la ubicación del usuario para mostrar estacionamientos cercanos.	EP3	5
3	US-13	Visualizar estacionamientos en mapa	Como arrendatario, quiero visualizar estacionamientos en el mapa.	EP3	5
4	US-12	Búsqueda de estacionamientos por dirección	Como arrendatario, quiero buscar por dirección para encontrar estacionamientos.	EP3	5

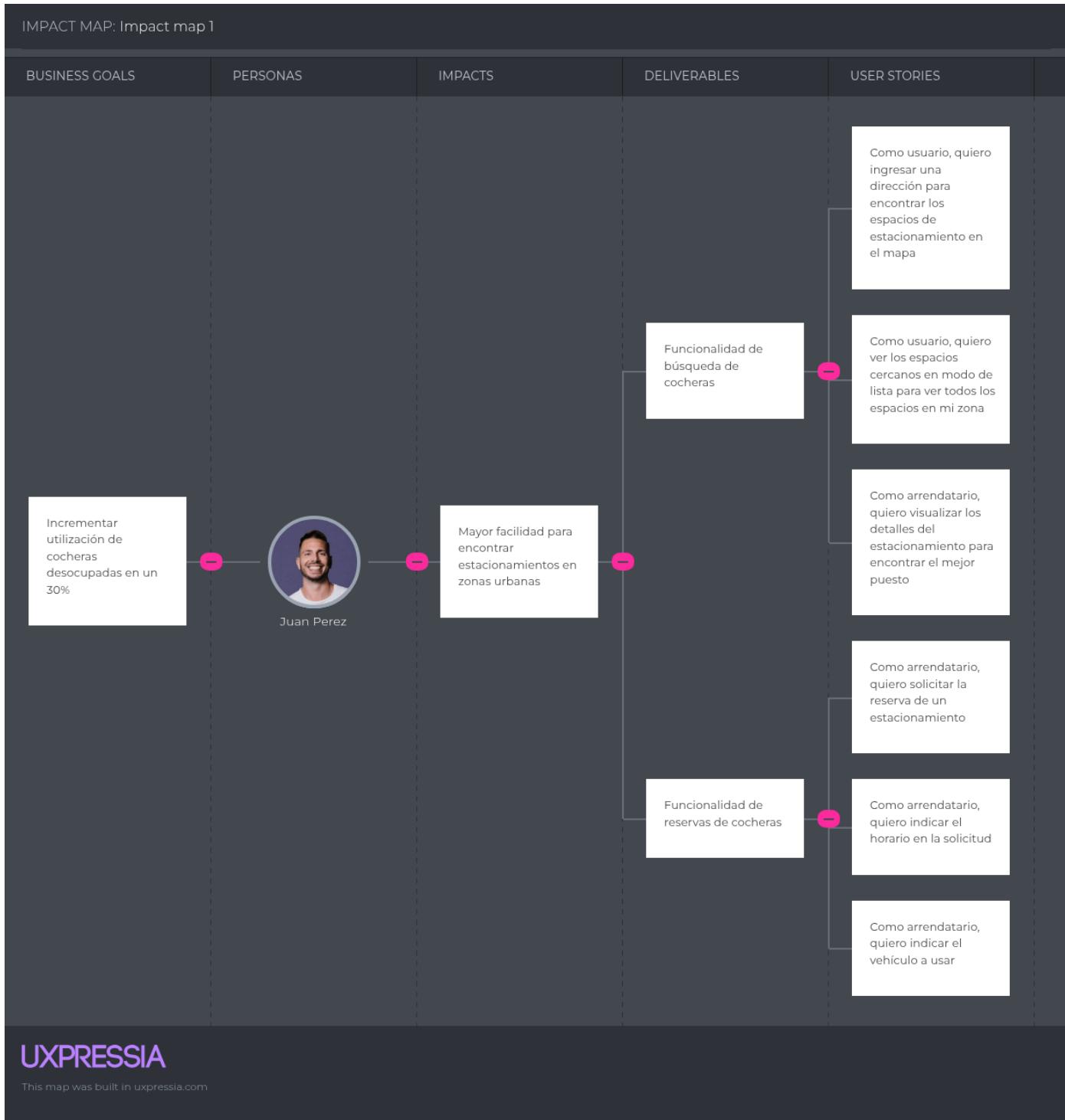
Orden	ID	Título	Descripción	Épica	Story Points
5	TS-06	Implementar CRUD de cocheras	Como desarrollador, quiero implementar funciones CRUD para cocheras registradas.	EP3	5
6	US-22	Solicitar reserva	Como arrendatario, quiero reservar un estacionamiento.	EP4	5
7	TS-08	Implementar CRUD de reservas	Como desarrollador, quiero implementar la funcionalidad CRUD para reservas.	EP4	5
8	TS-02	Configurar control de estado global	Como desarrollador, quiero implementar un sistema de manejo de estado para sincronizar datos.	EP2	5
9	US-15	Ver detalle de estacionamiento	Como arrendatario, quiero ver los detalles del estacionamiento.	EP3	3
10	US-23	Seleccionar horario de reserva	Como arrendatario, quiero seleccionar un horario para la reserva.	EP4	3
11	TS-01	Implementar sistema de navegación	Como desarrollador, quiero configurar un sistema de rutas nombradas para moverme entre vistas.	EP2	3
12	TS-03	Implementar validación de formularios	Como desarrollador, quiero agregar validaciones a los formularios para asegurar datos válidos.	EP2	3
13	US-09	Registrarse	Como usuario, quiero registrarme para crear una cuenta.	EP2	3
14	US-18	Registrar cochera	Como arrendador, quiero registrar mi cochera para generar ingresos.	EP3	3
15	US-20	Actualizar cochera	Como arrendador, quiero actualizar la información de mi cochera.	EP3	3
16	US-21	Eliminar cochera	Como arrendador, quiero eliminar una cochera registrada.	EP3	3
17	TS-07	Implementar CRUD de vehículos	Como desarrollador, quiero desarrollar funciones CRUD para vehículos del usuario.	EP7	3
18	TS-09	CRUD de perfil de usuario	Como desarrollador, quiero permitir la edición y visualización del perfil de usuario.	EP8	3
19	US-38	Subir comprobante de pago	Como arrendatario, quiero subir el comprobante de pago.	EP6	3
20	US-16	Ver calificación del estacionamiento	Como arrendatario, quiero ver calificaciones para evitar fraudes.	EP3	2
21	US-24	Seleccionar vehículo para reserva	Como arrendatario, quiero seleccionar un vehículo para la reserva.	EP4	2
22	US-25	Cancelar solicitud de reserva	Como arrendatario, quiero cancelar una reserva solicitada.	EP4	2
23	US-26	Rechazar solicitud de reserva	Como arrendador, quiero rechazar una solicitud de reserva.	EP4	2
24	US-27	Aprobar solicitud de reserva	Como arrendador, quiero aprobar una solicitud de reserva.	EP4	2
25	US-10	Iniciar sesión	Como usuario, quiero iniciar sesión con mi correo y contraseña.	EP2	2
26	US-17	Vista previa del lugar	Como arrendatario, quiero una vista previa del entorno del estacionamiento.	EP3	2
27	US-14	Ver espacios cercanos	Como arrendatario, quiero ver espacios cercanos en lista.	EP3	2
28	US-19	Ver mis cocheras registradas	Como arrendador, quiero ver mis cocheras registradas.	EP3	2
29	US-40	Añadir vehículo	Como usuario, quiero añadir un vehículo.	EP7	2
30	US-41	Editar vehículo	Como usuario, quiero editar mi vehículo.	EP7	2
31	US-42	Eliminar vehículo	Como usuario, quiero eliminar un vehículo.	EP7	2

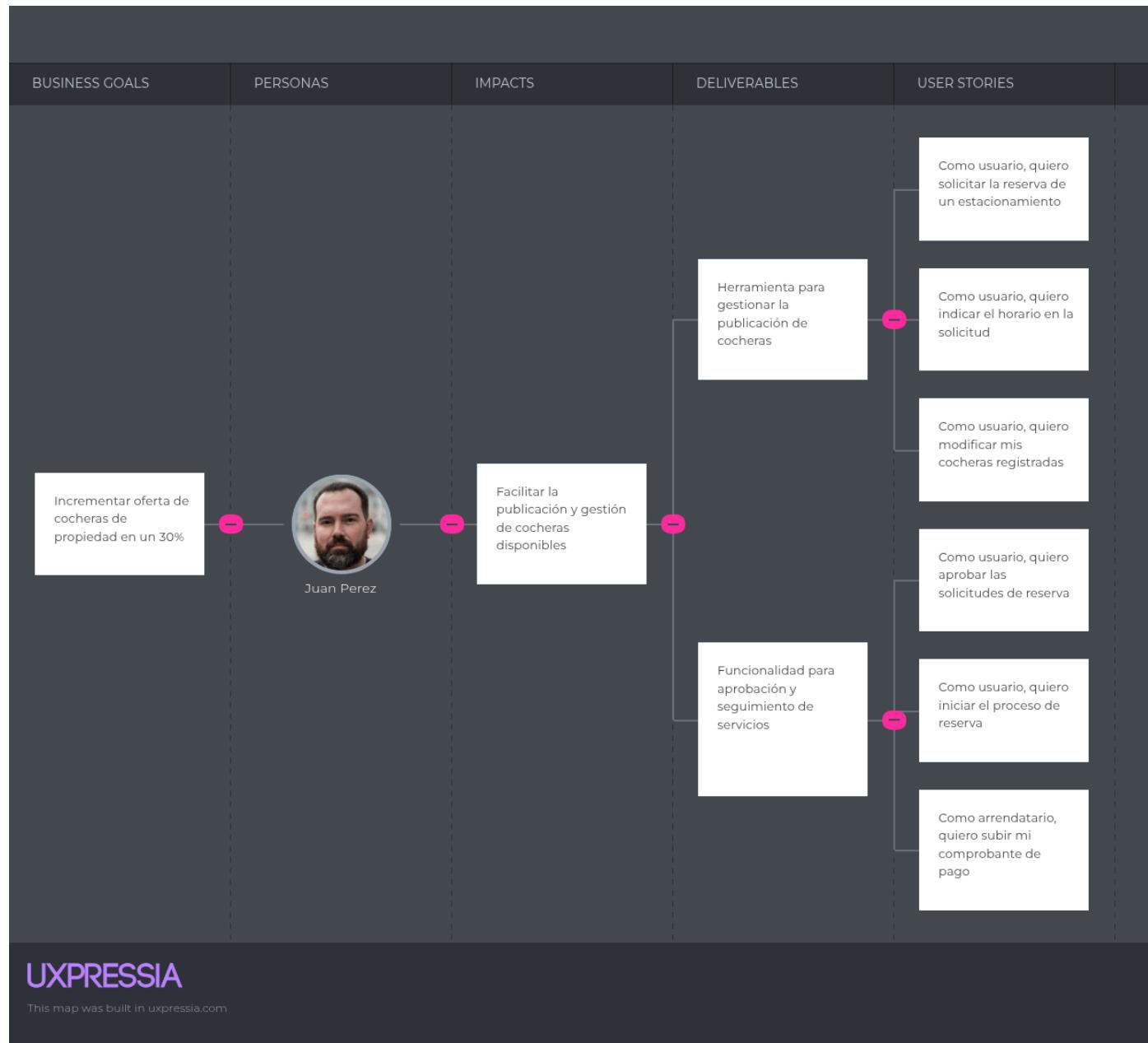
Orden	ID	Título	Descripción	Épica	Story Points
32	US-44	Actualizar perfil	Como usuario, quiero actualizar mi perfil.	EP8	2
33	US-03	Ver ejemplos de servicios	Como usuario, quiero ver ejemplos de servicios para imaginar su uso.	EP1	2
34	US-04	Ver precios	Como usuario, quiero ver una sección de precios para conocer los planes.	EP1	2
35	US-28	Ver próximas reservas	Como arrendatario, quiero ver mis próximas reservas.	EP5	2
36	US-29	Ver reservas en curso	Como arrendatario, quiero ver reservas en curso.	EP5	2
37	US-30	Ver historial de reservas	Como arrendatario, quiero ver el historial de mis reservas.	EP5	2
38	US-31	Ver detalle de reserva	Como arrendatario, quiero ver el detalle de una reserva.	EP5	2
39	US-32	Ver próximos servicios	Como arrendador, quiero ver mis próximos servicios.	EP5	2
40	US-33	Ver servicios en curso	Como arrendador, quiero ver servicios en curso.	EP5	2
41	US-34	Ver historial de servicios	Como arrendador, quiero ver historial de servicios.	EP5	2
42	US-35	Ver detalle del servicio	Como arrendador, quiero ver el detalle del servicio.	EP5	2
43	US-36	Iniciar proceso de reserva	Como arrendador, quiero iniciar el proceso de reserva.	EP6	2
44	US-37	Finalizar reserva	Como arrendatario, quiero finalizar la reserva.	EP6	2
45	US-11	Cerrar sesión	Como usuario, quiero cerrar sesión para proteger mi información.	EP2	1
46	US-39	Ver vehículos registrados	Como usuario, quiero ver mis vehículos registrados.	EP7	1
47	US-43	Ver perfil	Como usuario, quiero ver mi perfil.	EP8	1
48	US-01	Ver portada	Como usuario, quiero ver una portada para entender de qué trata la plataforma.	EP1	1
49	US-02	Ver beneficios	Como usuario, quiero ver los beneficios para entender por qué me conviene.	EP1	1
50	US-05	Ver testimonios	Como usuario, quiero ver testimonios para tener referencias.	EP1	1
51	US-06	Ver barra de navegación	Como usuario, quiero una barra de navegación para facilitar la búsqueda.	EP1	1
52	US-07	Ver equipo	Como usuario, quiero ver al equipo para conocer su propósito.	EP1	1
53	US-08	Ver footer	Como usuario, quiero un footer para acceder a información adicional.	EP1	1

### 3.4. Impact Mapping

El Impact Mapping es una técnica de planificación estratégica y visualización que nos ayuda a trazar el camino desde los objetivos del negocio hasta las entregas concretas. En el contexto de HomeyPark, utilizamos esta herramienta para conectar nuestros objetivos principales con los actores clave, sus comportamientos y las acciones específicas que necesitamos implementar.

#### Usuario de Parking

**Anfitrion**



## Capítulo IV: Product Design

### 4.1. Style Guidelines

En esta parte, presentaremos el plan de diseños, estilos y estéticas que hemos elaborado para nuestra página web y aplicativo móvil, con el objetivo de garantizar que nuestros usuarios disfruten de una interfaz amigable y fácil de utilizar. Para lograrlo, hemos optado por emplear elementos visuales que sean claramente visibles y estéticamente agradables, además de identificar una serie de restricciones para evitar incluir elementos gráficos discordantes o poco llamativos.

#### 4.1.1. General Style Guidelines

**Branding:** Nuestro logo para HomeyPark presenta la silueta estilizada de un auto, representando la facilidad y conveniencia del estacionamiento. Debajo del auto, se encuentran las letras "HomeyPark" en una tipografía moderna y clara, que asegura legibilidad y fácil reconocimiento. Este diseño simboliza la accesibilidad y la confiabilidad que ofrecemos a través de la aplicación, reflejando seguridad y profesionalismo, valores fundamentales de HomeyPark.



**Typography:**\* Para HomeyPark, se seleccionaron dos tipografías que trabajan en conjunto para crear una experiencia visual moderna y accesible. La tipografía principal es Rubik, que aporta un estilo geométrico y contemporáneo, ideal para títulos y encabezados, asegurando un impacto visual fuerte. La tipografía secundaria es Mulish, utilizada para el cuerpo de texto y descripciones, gracias a su diseño limpio y sencillo que mejora la legibilidad en dispositivos móviles y de escritorio. Juntas, estas tipografías aseguran que toda la información relevante sea clara y fácil de leer para los usuarios.

## Typography

### Rubik

Usually used for content like titles, subtitles and heading.  
9 font weights available

### Mulish

Usually used for content like paragraphs and additional information.  
9 font weights available

### DESKTOP BREAKPOINT

## Heading 1

## Rubik 60/72

## Heading 2

## Rubik 48/64

## Heading 3

## Rubik 40/48

### Subtitle 1

### Rubik 24/40

### Subtitle 2

### Rubik 20/24

### Body 1 (Bold)

### Mulish 18/28

### Body 1 (Regular)

### Mulish 18/28

### Body 2 (Bold)

### Mulish 16/24

### Body 2 (Regular)

### Mulish 16/24

**Colores:** Los colores seleccionados para HomeyPark están diseñados para ofrecer una experiencia visualmente agradable y coherente, al mismo tiempo que transmiten sensaciones de confianza y frescura. El tono **#3C4E67** es un azul oscuro que aporta profesionalismo y estabilidad, mientras que el **#6CD391**, un verde vibrante, introduce un toque de frescura y modernidad. Juntos, estos colores crean una atmósfera de serenidad y dinamismo, reflejando los valores de accesibilidad y confianza de nuestra marca.



#3C4E67

#6cd391

**Spacing:** Hemos adoptado un espaciado base de 4px para los márgenes y el padding en todos los elementos, garantizando una presentación consistente y bien alineada en toda la interfaz. Este enfoque contribuye a una estructura visual limpia y ordenada, lo que facilita la navegación y mejora la experiencia del usuario tanto en dispositivos móviles como en la web.

### 4.1.2. Web Style Guidelines

Para la versión web de HomeyPark, se ha priorizado una disposición responsive que garantice una experiencia fluida sin importar el tamaño de la pantalla. Se utilizan diseños basados en rejillas (grid systems) que permiten adaptar los contenidos de manera flexible. Los botones presentan esquinas suavemente redondeadas (8px de radio) y efectos de hover que consisten en un ligero sombreado y cambio de color para indicar interactividad sin ser intrusivos. Los íconos empleados siguen una línea visual uniforme, con un tamaño estándar de 24px, permitiendo una rápida identificación de acciones y secciones. Además, se han incorporado transiciones suaves para cambios de estado (como desplegables o cambios de pestaña), mejorando la percepción de fluidez en la interfaz.

### 4.1.3. Mobile Style Guidelines

#### 4.1.3.1. iOS Mobile Style Guidelines

En la versión para dispositivos iOS, HomeyPark adopta las convenciones de diseño propias del ecosistema Apple, como el uso de botones de navegación tipo "back" en la parte superior izquierda y el aprovechamiento del gesto de deslizamiento para retornar a pantallas anteriores. La interfaz sigue el patrón Human Interface Guidelines de Apple, lo que se refleja en la fluidez de las animaciones, la tipografía ajustada a San Francisco (cuando es posible), y la integración con funcionalidades como el modo oscuro. Los menús y ventanas emergentes presentan bordes redondeados y un fondo translúcido que se adapta al contexto visual del sistema operativo.

#### 4.1.3.2. Android Mobile Style Guidelines

Para la versión Android, se siguen las Material Design Guidelines, priorizando la claridad visual, la jerarquía de información y la retroalimentación visual al usuario. Se utilizan componentes nativos como el AppBar, Floating Action Buttons (FAB), y Snackbars para acciones rápidas y notificaciones no intrusivas. La paleta de colores respeta los tonos definidos por la identidad de HomeyPark, integrándose con la interfaz del sistema. Se han optimizado las vistas para múltiples resoluciones y densidades de pantalla, garantizando una presentación coherente en diversos dispositivos Android. Además, se respeta la navegación por gestos introducida en versiones recientes del sistema.

## 4.2. Information Architecture

Centrados en el objetivo de HomeyPark, buscamos ofrecer una interfaz amigable e intuitiva que inspire confianza y seguridad a los usuarios. Dado que nuestra plataforma facilita la búsqueda y reserva de espacios de estacionamiento, estas características deben prevalecer en toda la experiencia de usuario. Un componente crucial para lograrlo es la arquitectura de información, diseñada para guiar a los usuarios a través de la plataforma de manera eficiente. A continuación, se detalla el plan de arquitectura de información implementado en HomeyPark.

1. Página de Inicio:

**Home**

Sección que resalta el valor principal de HomeyPark, además explica brevemente cómo la plataforma facilita el proceso de encontrar y gestionar estacionamientos. Esta sección invita a los usuarios a explorar las ventajas de la plataforma y los anima a registrarse con el botón presente que los redirige a la página de registro.

#### **Why Choose Us?**

Sección que resalta los beneficios y características clave de HomeyPark, explicando por qué los usuarios deben elegir nuestra plataforma para encontrar estacionamiento.

#### **How It Works:**

Descripción clara y concisa del funcionamiento de la plataforma, explicando los pasos simples para registrarse, buscar y reservar un lugar de estacionamiento.

#### **Testimonials:**

Sección que presenta opiniones y experiencias de usuarios que ya han utilizado HomeyPark, destacando la facilidad y comodidad que ofrece la plataforma.

#### **Pricing:**

Información sobre los planes de precios de HomeyPark, proporcionando detalles transparentes y claros para que los usuarios sepan lo que pueden esperar.

#### **Formulario de Contacto:**

Al final de la página, se incluye un formulario de contacto para que los usuarios puedan enviar consultas o pedir asistencia.

#### 2. Registro:

#### **Registro de Usuarios:**

Formulario que permite a los usuarios crear una cuenta en HomeyPark, solicitando información básica como nombre, correo electrónico y una contraseña segura.

#### **Inicio de Sesión:**

Opción para que los usuarios ya registrados puedan acceder a su cuenta introduciendo su correo electrónico y contraseña.

#### **4.2.1. Organization Systems**

El sistema de organización en HomeyPark está diseñado para brindar una experiencia clara y eficiente, permitiendo a los usuarios navegar fácilmente por la plataforma. Nuestro enfoque está centrado en facilitar el acceso a la información y las funcionalidades clave, lo que permite a los usuarios registrarse, conocer los beneficios, y contactar rápidamente con nosotros.

##### 1. Categorización de la Información:

- Why Choose Us?

Presenta los beneficios clave de la plataforma para los usuarios, como facilidad de uso, seguridad y conveniencia.

- How It Works

Categorizada en pasos simples para que los usuarios puedan entender rápidamente el funcionamiento del proceso de búsqueda y reserva de estacionamientos.

##### 2. Filtros y Búsqueda:

- Filtros

Aunque no se muestran opciones de búsqueda avanzadas en la landing page, los usuarios podrán aplicar filtros en la plataforma principal para ajustar sus preferencias, como ubicación y disponibilidad de espacios de estacionamiento.

#### **4.2.2. Labeling Systems**

Para el contenido, se ha priorizado la claridad y brevedad en los textos, enfocándonos en destacar los beneficios clave de la plataforma para los usuarios. El diseño de los botones sigue un estilo minimalista con colores principales y bordes redondeados.

En cuanto a los íconos, se emplean elementos visuales sencillos y los colores del sistema de diseño del equipo para facilitar la navegación y comprensión rápida.

#### **4.2.3. SEO Tags and Meta Tags**

Para asegurar una correcta indexación y visibilidad en los motores de búsqueda, es fundamental la implementación estratégica de SEO Tags y Meta Tags en la estructura HTML de nuestras páginas web. La siguiente imagen ilustra la configuración de estos elementos clave.

```

<meta charset="UTF-8" />
<meta http-equiv="X-UA-Compatible" content="IE=edge" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<meta name="robots" content="index, follow" />
<link rel="shortcut icon" href="./images/myicon.png" type="image/x-icon" />
<title>HomeyPark | Reserve Your Parking</title>
<meta
  name="keywords"
  content="parking, homes, hosts, free, hourly rates, generate income,
  vehicles"
/>
<meta
  name="description"
  content="Find your spot, hassle-free. Over 3,000 users already trust
  HomeyPark to find their perfect parking spot effortlessly, exploring new ways
  to park and move around the city without the stress of searching for space."
/>

```

#### 4.2.4. Searching Systems

El sistema de búsqueda en HomeyPark facilitará a los usuarios encontrar la información que necesitan para utilizar la plataforma, centrándose principalmente en cómo funciona el sistema, precios y opiniones de otros usuarios.

##### **Búsqueda por Características de Servicio:**

Los usuarios podrán buscar información sobre cómo funciona la plataforma, las ventajas de elegir HomeyPark, y testimonios de usuarios.

##### **Búsqueda por Precio:**

Los usuarios podrán encontrar fácilmente la sección de precios para obtener información detallada sobre las tarifas.

#### 4.2.5. Navigation Systems

La navegación en HomeyPark ha sido diseñada para ser intuitiva, guiada y consistente a lo largo de toda la experiencia del usuario, tanto en la Landing Page como en las aplicaciones móviles y web. El objetivo es facilitar el cumplimiento de tareas como la búsqueda, reserva o publicación de estacionamientos, sin fricciones ni pérdidas de contexto.

##### **Landing Page**

En la Landing Page, se implementan elementos visuales y patrones de navegación claros que permiten a los usuarios explorar rápidamente la propuesta de valor. Se incluyen:

- Barra de navegación fija que permite moverse entre secciones como beneficios, precios, testimonios y contacto.
- Botones de acción visibles ("Regístrate", "Reserva ahora") que redirigen a las vistas correspondientes según el perfil del usuario.
- Scroll guiado verticalmente, con secciones jerarquizadas para facilitar la lectura secuencial del contenido.

##### **Aplicaciones móviles y web**

En las plataformas principales del producto, se utilizan rutas nombradas y una estructura de navegación jerárquica y contextual:

- Menú de navegación lateral o inferior (según la plataforma) para acceder rápidamente a vistas clave como Inicio, Mis reservas, Mis vehículos, Mi perfil, Buscar estacionamientos, entre otros.
- Navegación a través de botones contextuales, como "Reservar", "Ver detalles", "Editar perfil", que llevan al usuario a pantallas específicas dentro de su flujo actual.
- Integración con Google Maps que permite navegación basada en geolocalización para visualizar y seleccionar espacios cercanos.
- Uso de breadcrumbs y retrocesos claros, especialmente en la versión web, para no perder el hilo de navegación.

##### **Técnicas y acciones clave**

- Uso de flujos de usuario (user flows) previamente definidos para asegurar que cada perfil (guest y host) pueda alcanzar sus objetivos con el mínimo de pasos.
- Incorporación de redirecciones automáticas post-registro/login hacia los dashboards personalizados según el rol.
- Inclusión de notificaciones y alertas para guiar acciones pendientes (como confirmar reservas o completar registros).

Esta arquitectura de navegación está diseñada para minimizar la carga cognitiva, mantener a los usuarios orientados y maximizar la eficiencia en la interacción con la plataforma.

## 4.3. Landing Page UI Design

### 4.3.1. Landing Page Wireframe

The wireframe illustrates the layout of a landing page:

- Header:** Features a logo on the left and a navigation bar with links: Home, Why choose us?, How it works?, Testimonials, Pricing, Sign in, and Sign up.
- Hero Section:** Contains the main title "Find your spot, hassle-free", a subtitle "We are a platform that simplifies exploring new transportation options, promoting sustainable mobility and environmental care.", and a "Get started" button.
- Placeholder Image:** A large, light gray rectangular area containing a dark gray square with a white mountain icon.
- Text Placeholder:** A light gray box containing placeholder text: "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque accumsan velit sem, vehicula sollicitudin tortor imperdiet et. Curabitur blandit mollis diam, imperdiet porttitor dolor tempus quis."
- Section: Why choose us?** This section contains three items, each consisting of a small image, a title, and a text block:
  - Image:** A small gray square with a white mountain icon.
  - Title:**  **Lorem ipsum**
  - Text:** Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque accumsan velit sem.
  - Image:** A small gray square with a white mountain icon.
  - Title:**  **Lorem ipsum**
  - Text:** Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque accumsan velit sem.
  - Image:** A small gray square with a white mountain icon.
  - Title:**  **Lorem ipsum**
  - Text:** Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque accumsan velit sem.

# How it works?

**01**

## **Lorem ipsum**

Start by searching in your area. HomeyPark uses your location to show available parking spots nearby, making it easy to find the most convenient options. Simply allow location access, and our app will do the rest, guiding you to the best spots closest to you.



**02**

## **Lorem ipsum**

Select your parking spot easily. After searching in your area, HomeyPark will present you with available options. Choose the spot that best fits your needs and preferences, and secure it with just a few taps. Finding the perfect parking spot has never been simpler!



**03**

## **Lorem ipsum**

Reserve your parking spot with ease. Once you've selected your preferred location, confirm your reservation in just a few clicks. HomeyPark ensures that your chosen spot is secured and ready for you when you arrive.

## Our testimonials



John Doe  
Avenue, street

★ 4.5

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque accumsan velit sem, vehicula sollicitudin tortor imperdiet et.



John Doe  
Avenue, street

★ 4.5

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque accumsan velit sem, vehicula sollicitudin tortor imperdiet et.



John Doe  
Avenue, street

★ 4.5

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque accumsan velit sem, vehicula sollicitudin tortor imperdiet et.

## Pricing

### Drivers



**Some pricing term**  
Some pricing definitions  
of a term



**Some pricing term**  
Some pricing definitions  
of a term



**Some pricing term**  
Some pricing definitions  
of a term

### Hosters



**Some pricing term**  
Some pricing definitions  
of a term

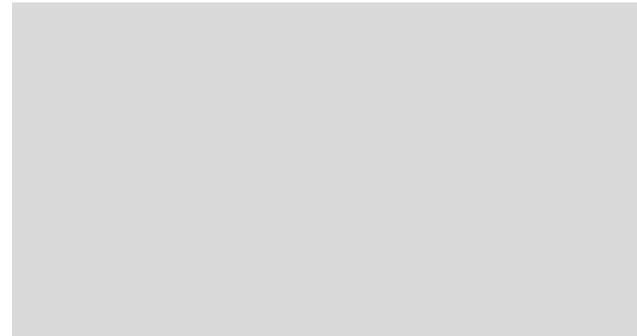


**Some pricing term**  
Some pricing definitions  
of a term



**Some pricing term**  
Some pricing definitions  
of a term

## About the product



**Contact us**

Lore ipsum dolor sit amet, consectetur adipiscing elit. Cras varius mauris quis nisi faucibus venenatis. Donec pretium non turpis eget rutrum. Suspendisse porta at ipsum.

Name

Email address

Message

**Submit**



LOGO



Copyright © 2024 by HomeyPark, Inc.  
All rights reserved

### Contact us

Saloverry Avenue 123

999-999-999

homeypark@mail.com

### Company

About HomeyPark

For Business

Careers

### Account

Create account

Sign in

iOS app

Android App

### About us

Help center

Privacy & terms

# Find your spot, hassle-free

We are a platform that simplifies exploring new transportation options, promoting sustainable mobility and environmental care.

[Get started](#)

Over 3,000 users already trust HomeyPark to find their perfect parking spot effortlessly, exploring new ways to park and move around the city without the stress of searching for space.

## Why Choose Us?



### Easy to use

Our app quickly finds parking spots, letting you spend less time searching and more time enjoying your journey.



### Access from Any Device

Access HomeyPark from any device—phone, tablet, or computer—to manage your parking spots anytime, anywhere.



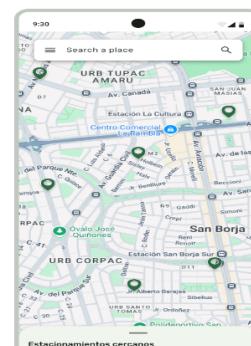
### Quick and Easy Reservation

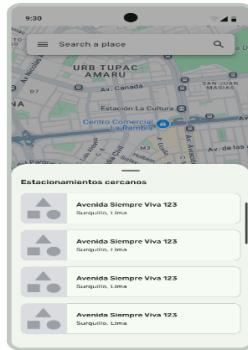
With HomeyPark, quickly book parking ahead of time to guarantee a spot when you need it.

## How it works?

### 01 Search in your area

Start by searching in your area. HomeyPark uses your location to show available parking spots nearby, making it easy to find the most convenient options.





## 02 Choose a parking spot

Select your parking spot easily. After searching in your area, HomeyPark will present you with available options. Choose the one that fits your needs and preferences, and secure it with just a few taps.

## 03 Reserve your parking spot

Reserve your parking spot with ease. Once you've selected your preferred location, confirm your reservation in just a few clicks. HomeyPark ensures that your chosen spot is secured and ready for you when you arrive.



## Our testimonials



**Maria Perez**  
Lima, Peru

★ 5.0

"Excelente servicio! Los estacionamientos son seguros y el personal es muy amable. Recomiendo esta aplicación a todos los propietarios de viviendas en Lima."



**Carlos Rodriguez**  
Arequipa, Peru

★ 4.5

"Muy útil para encontrar estacionamientos cerca de mi casa. La aplicación es fácil de usar y los precios son razonables. ¡Gracias!"

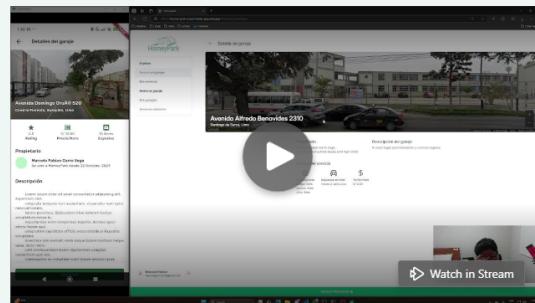


**Ana Gomez**  
Trujillo, Peru

★ 4.0

"Me encanta esta aplicación. Siempre encuentro estacionamientos disponibles y el proceso de reserva es muy sencillo."

## About the product



## Pricing

### Drivers



#### Choose the service that fits your convenience.

Find the perfect option for you with just a few clicks.



#### Many rates, many garages

Choose from a wide variety of available parking spots.



#### Fast and secure payments

Payment gateway verified by multiple organizations.

### Hosts



#### Set the rate for your parking spot

Customize your pricing and manage your space effortlessly.



#### Daily earnings

Generate daily passive and/or active income with our app.



#### Instant payment transfers

Direct communication with financial institutions.

### Contact us

Our team is always here to help and ensure you have the best experience possible. Contact us anytime, and we'll get back to you as soon as we can!






© Copyright © 2025 by  
HomeyPark, Inc.  
All rights reserved

#### Contact Us

Salaverry Avenue 123

999 999 999

homeypark@gmail.com

#### Company

About HomeyPark

For Business

Careers

#### Account

Create account

Sign in

iOS app

Android app

#### About Us

Help center

Privacy & terms

## 4.4. Mobile Applications UX/UI Design

### 4.4.1. Mobile Applications Wireframes

# Registro

Email

Contraseña

Repetir contraseña

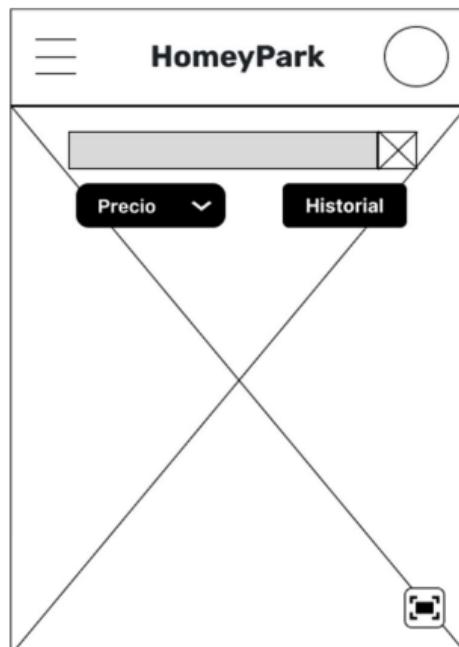
**Registrate**

# Inicio de sesión

Email

Contraseña

**Ingresar**



**HomeyPark**

Precio ▾ Historial

**Estacionamientos cercanos:**

- 
- 
-

**Historial de reservas**

Reservas:

- Cancelar reserva

Reservas pasadas:

-



Nombre de estacionamiento



Descripción

Precio  
199.00

Calificación →  
4.2

**Reservar ahora**



**Reseñas**

4.2

Nombre de estacionamiento

Ordenar por:

Comentario:

Comentario:

Comentario:



**Registro de cochera**

Dirección del espacio de renta

Longitud

Ancho

Altura

Hora de inicio

Hora de cierre

Precio por hora

Número de teléfono

Espacios disponibles

Descripción



**Registrar cochera**

**Editar perfil**

Cambiar imagen

**Nombre**

**Email**

**Contraseña**

**Método de pago**  
**Añadir tarjeta**

**Añadir tarjeta**

**Número de tarjeta**

**Fecha de vencimiento**

**Código de seguridad**

**Guardar cambios**

**Guardar cambios**

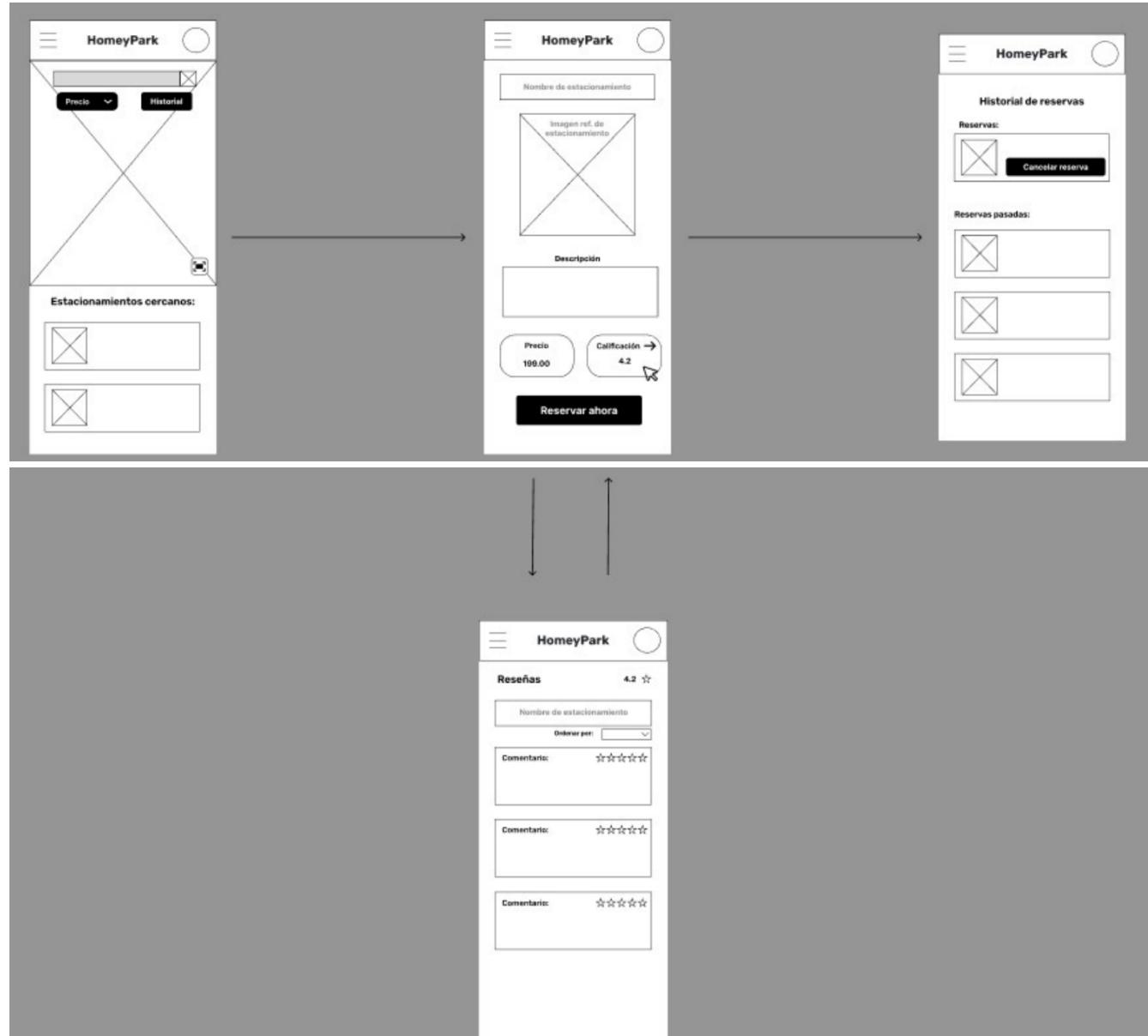
#### 4.4.2. Mobile Applications Wireflow Diagrams

**User goal:** El usuario se registra en la plataforma e inicia sesión en la aplicación con sus datos.



Descripción: Al iniciar la aplicación, el usuario se encuentra con el apartado de registro de cuenta, donde podrá registrarse con su email y contraseña. Cuando le de al botón de registrarse, le mandará al apartado de inicio de sesión, donde podrá ingresar sus datos anteriormente añadidos y al ser las correctas, podrá acceder a la página principal de la plataforma.

**User goal:** Usuario desea hacer la búsqueda y reserva de un espacio de estacionamiento



**Descripción:** El usuario al encontrarse en la página principal, podrá visualizar y buscar estacionamientos de su preferencia. Al seleccionar un estacionamiento, verá el detalle de este, aquí el usuario contará con dos opciones, ver los comentarios y calificación o seleccionar la opción de reservar ahora. El usuario al ingresar a la sección de calificación visualizará los comentarios que ha obtenido dicho estacionamiento, así como también la calificación en general del mismo. Por otro lado, luego de seleccionar la opción de reservar, el estacionamiento habrá quedado apartado y podrá revisarlo en la opción de historial del menú principal, donde también podrá cancelar su reserva o revisar sus reservas pasadas.

#### 4.4.3. Mobile Applications Mock-ups

## Inicio de sesión

Email —  
example@mail.com

Contraseña —  
\*\*\*\*\* 

[¿Olvidó su contraseña?](#)

**Ingresar**

[Crear cuenta](#)

## Crear cuenta

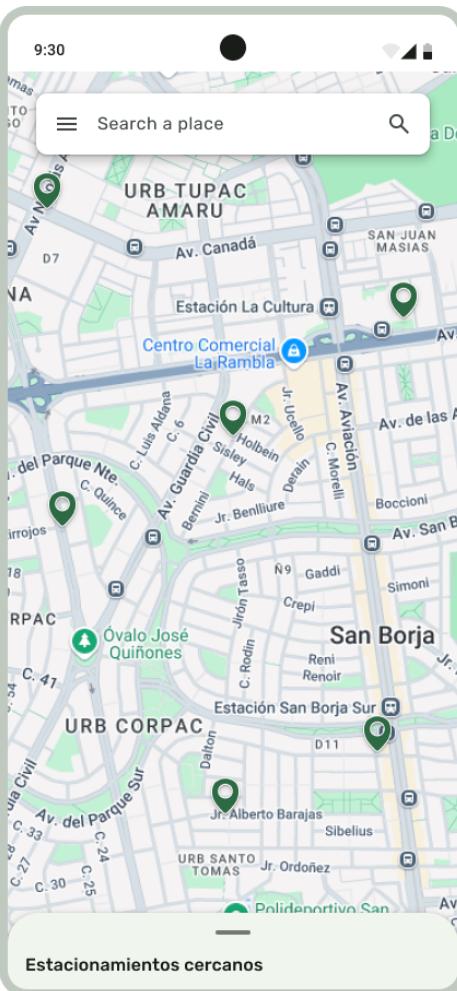
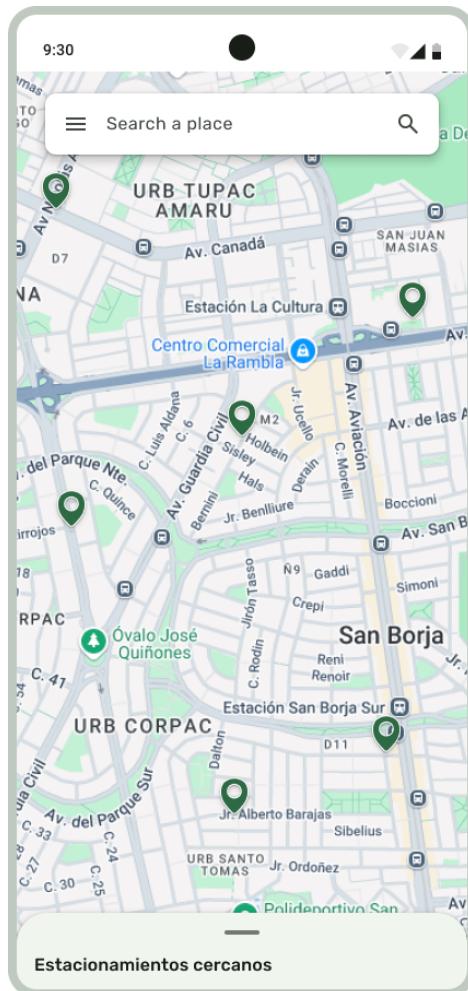
Email —  
example@mail.com

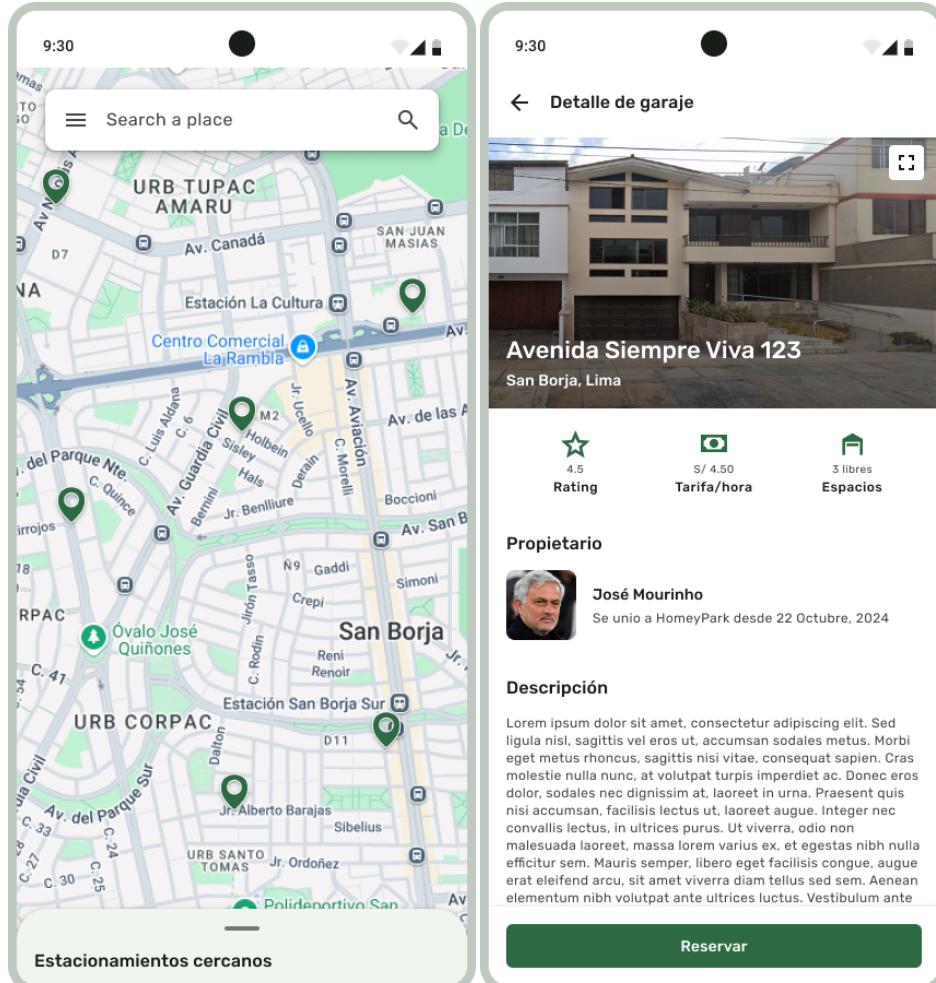
Contraseña —  
\*\*\*\*\* 

Repetir contraseña —  
\*\*\*\*\* 

[¿Tiene una cuenta? Inicie sesión](#)

**Registrarse**





9:30

**Checkout**

**Horario**

Fecha — 11/24/2024

MM/DD/YYYY

Hora inicio — 03:00 PM      Hora fin — 03:30 PM

**Medio de pago**

\*\*\*\* \*4885 MARCELO FABIAN GARRO VEGA

**Vehículo**

B521C31 Toyota Corolla

**Resumen**

Tarifa total S/ 2.25

**Pagar y reservar**

9:30

**Checkout**

**Horario**

Fecha — 11/24/2024

MM/DD/YYYY

Hora inicio — 03:00 PM      Hora fin — 03:30 PM

**Medio de pago**

\*\*\*\* \*4885 MARCELO FABIAN GARRO VEGA

**Vehículo**

B521C31 Toyota Corolla

**Resumen**

Tarifa total S/ 2.25

**Pagar y reservar**

9:30

← Mis reservas

En progreso En camino Pasado

Avenida Siempre viva #1000001 Pendiente  
Desde 03:00PM - 12/12/2024  
Hasta 03:30PM - 12/12/2024

Avenida Siempre viva #1000001 Aprobado  
Desde 03:00PM - 12/12/2024  
Hasta 03:30PM - 12/12/2024

Avenida Siempre viva #1000001 Aprobado  
Desde 03:00PM - 12/12/2024  
Hasta 03:30PM - 12/12/2024

Avenida Siempre viva #1000001 Aprobado  
Desde 03:00PM - 12/12/2024  
Hasta 03:30PM - 12/12/2024

Avenida Siempre viva #1000001 Pendiente

9:30

← Mis reservas

En progreso En camino Pasado

Avenida Siempre viva #1000001 Completado  
Desde 03:00PM - 12/12/2024  
Hasta 03:30PM - 12/12/2024

Avenida Siempre viva #1000001 Completado  
Desde 03:00PM - 12/12/2024  
Hasta 03:30PM - 12/12/2024

Avenida Siempre viva #1000001 Cancelado  
Desde 03:00PM - 12/12/2024  
Hasta 03:30PM - 12/12/2024

Avenida Siempre viva #1000001 Completado  
Desde 03:00PM - 12/12/2024  
Hasta 03:30PM - 12/12/2024

Avenida Siempre viva #1000001 Cancelado

9:30

← Detalles de reserva Aprobado



**Avenida Siempre Viva 123**  
San Borja, Lima

**Horario**

**Creado:**  
22 de Noviembre del 2024

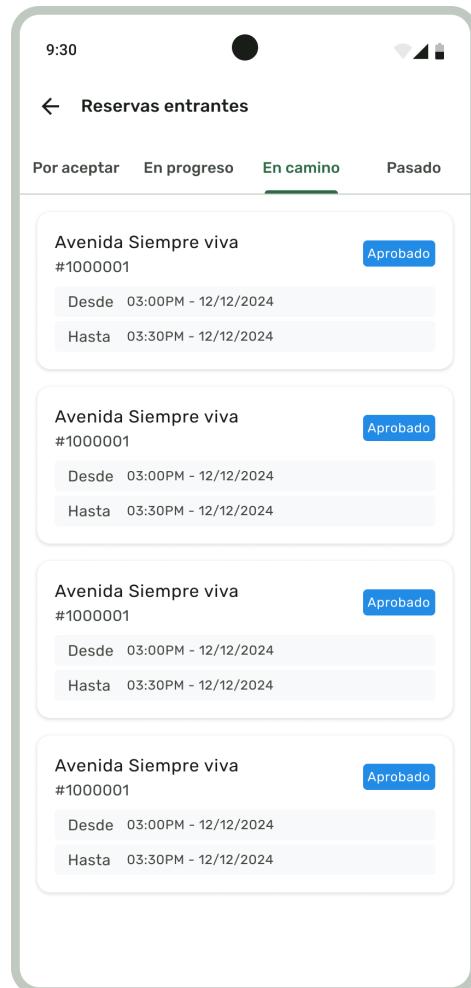
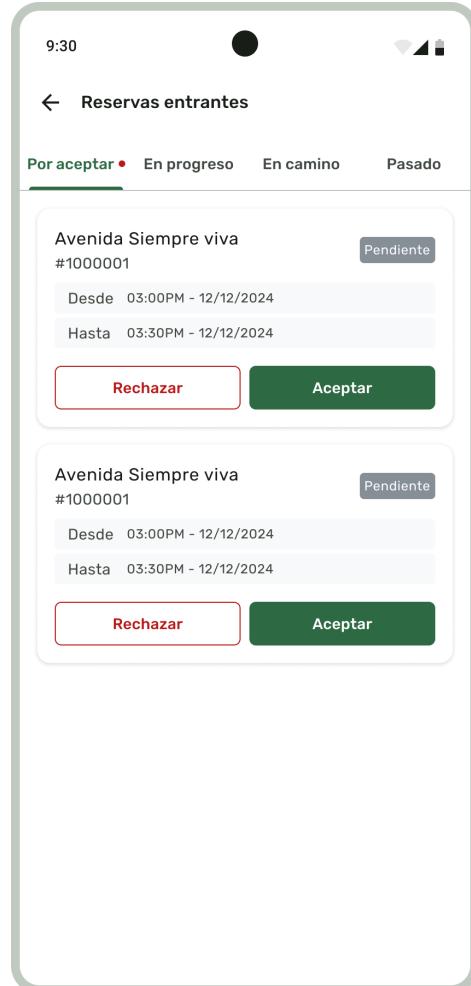
**Horario de reserva:**  
Desde: 03:00PM - 12/12/2024  
Hasta: 03:00PM - 12/12/2024

**Información adicional**

 **B521C31**  
Toyota Corolla

 **\*\*\*\*\* \*4885**  
MARCELO FABIAN GARRO VEGA





9:30

← Reservas entrantes

Pendiente

Avenida Siempre Viva 123  
San Borja, Lima

**Horario**

**Creado:**  
22 de Noviembre del 2024

**Horario de reserva:**  
Desde: 03:00PM - 12/12/2024  
Hasta: 03:00PM - 12/12/2024

**Información adicional**

B521C31  
Toyota Corolla

\*\*\*\* \* \* \* \* 4885  
MARCELO FABIAN GARRO VEGA

9:30

← Reservas entrantes

Aprobado

Avenida Siempre Viva 123  
San Borja, Lima

**Horario**

**Creado:**  
22 de Noviembre del 2024

**Horario de reserva:**  
Desde: 03:00PM - 12/12/2024  
Hasta: 03:00PM - 12/12/2024

**Información adicional**

B521C31  
Toyota Corolla

\*\*\*\* \* \* \* \* 4885  
MARCELO FABIAN GARRO VEGA

9:30

← Tus garages

Avenida Siempre Viva 123  
Surquillo, Lima

Borrar Edit

Avenida Siempre Viva 123  
Surquillo, Lima

Borrar Edit

**+ Agregar**

9:30

← Registra tu cochera

**Ubicación**

Dirección  
Avenida Siempre Viva 123

**Dimensions**

Espacios disponibles  
2

Altura  
12.5

Longitud  
4

Ancho  
6

**Horario**

Lunes 11:00 AM - 09:00pm

Martes 11:00 AM - 09:00pm

**Agregar cochera**

9:30

← Registra tu cochera

**Ubicación**

Dirección  
Avenida Siempre Viva 123



9:30

← Registra tu cochera

**Ubicación**

Dirección  
Avenida Siempre Viva 123

**Agrega un horario**

Día  
Jueves

Hora inicio  
06:00 AM

Hora fin  
09:00 PM

Cancelar Agregar

Altura  
12.5

Longitud  
4

Ancho  
6

En metros En metros En metros

**Horario**

Lunes 11:00 AM - 09:00pm X Pencil

Martes 11:00 AM - 09:00pm X Pencil

Miercoles 11:00 AM - 09:00pm X Pencil

(+) Agregar horario

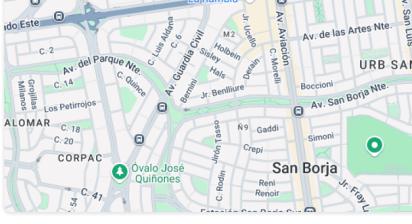
Agregar cochera

9:30

← Registra tu cochera

**Ubicación**

Dirección  
Avenida Siempre Viva 123



**Dimensiones**

Espacios disponibles  
2

Altura  
12.5

Longitud  
4

Ancho  
6

En metros En metros En metros

**Horario**

Lunes 11:00 AM - 09:00pm X Pencil

Martes 11:00 AM - 09:00pm X Pencil

Miercoles 11:00 AM - 09:00pm X Pencil

(+) Agregar horario

Agregar cochera

The screenshots show the user flow for registering a parking space (cochera) in the HoemyPark app.

**Screenshot 1 (Top Left):** Shows the registration screen for a parking space at "Avenida Siempre Viva 123". It includes a map of the area around San Borja, dimensions (2 spaces, 12.5m x 4m x 6m), and a weekly availability slot from 11:00 AM to 09:00 PM on Monday. A green "Agregar cochera" button is at the bottom.

**Screenshot 2 (Top Right):** Shows the same registration screen for the same location and dimensions, but with a different weekly availability slot from 11:00 AM to 09:00 PM on Monday. A green "Agregar cochera" button is at the bottom.

**Screenshot 3 (Bottom Left):** Shows the registration screen for the same location and dimensions, but with a different weekly availability slot from 11:00 AM to 09:00 PM on Monday. A green "Agregar cochera" button is at the bottom.

**Screenshot 4 (Bottom Right):** Shows the vehicle management screen ("Vehiculos") with an "Agregar" button. It lists a Toyota Corolla with license plate B123123. A modal window titled "Agrega un horario" (Add a schedule) is open, showing a dropdown for "Dia" (Day) set to "Jueves" (Thursday), and two time inputs for "Hora inicio" (Start time) at 06:00 AM and "Hora fin" (End time) at 09:00 PM. Buttons for "Cancelar" (Cancel) and "Agregar" (Add) are at the bottom of the modal.

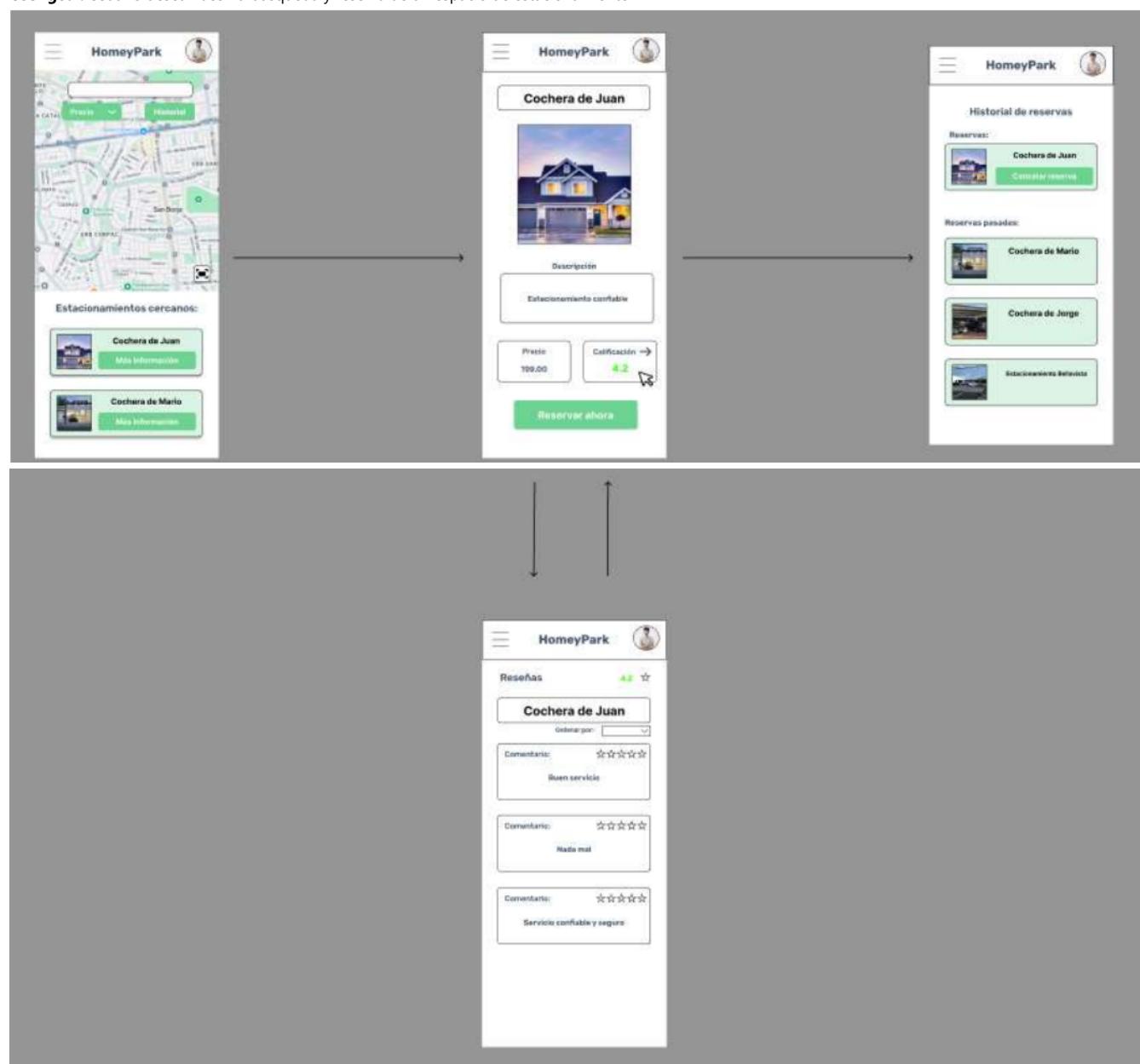
#### 4.4.4. Mobile Applications User Flow Diagrams

**User goal:** El usuario se registra en la plataforma e inicia sesión en la aplicación con sus datos.



**Descripción:** Al iniciar la aplicación, el usuario se encuentra con el apartado de registro de cuenta, donde podrá registrarse con su email y contraseña. Cuando le de al botón de registrarse, le mandará al apartado de inicio de sesión, donde podrá ingresar sus datos anteriormente añadidos y al ser las correctas, podrá acceder a la página principal de la plataforma.

**User goal:** Usuario desea hacer la búsqueda y reserva de un espacio de estacionamiento

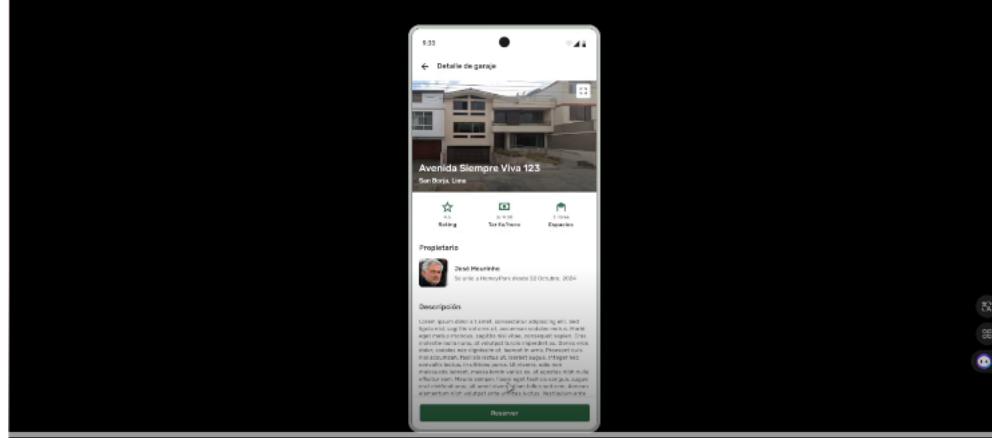


**Descripción:** El usuario al encontrarse en la página principal, podrá visualizar y buscar estacionamientos de su preferencia. Al seleccionar un estacionamiento, verá el

detalle de este, aquí el usuario contará con dos opciones, ver los comentarios y calificación o seleccionar la opción de reservar ahora. El usuario al ingresar a la sección de calificación visualizará los comentarios que ha obtenido dicho estacionamiento, así como también la calificación en general del mismo. Por otro lado, luego de seleccionar la opción de reservar, el estacionamiento quedará apartado y podrá revisarlo en la opción de historial del menú principal, donde también podrá cancelar su reserva o revisar sus reservas pasadas.

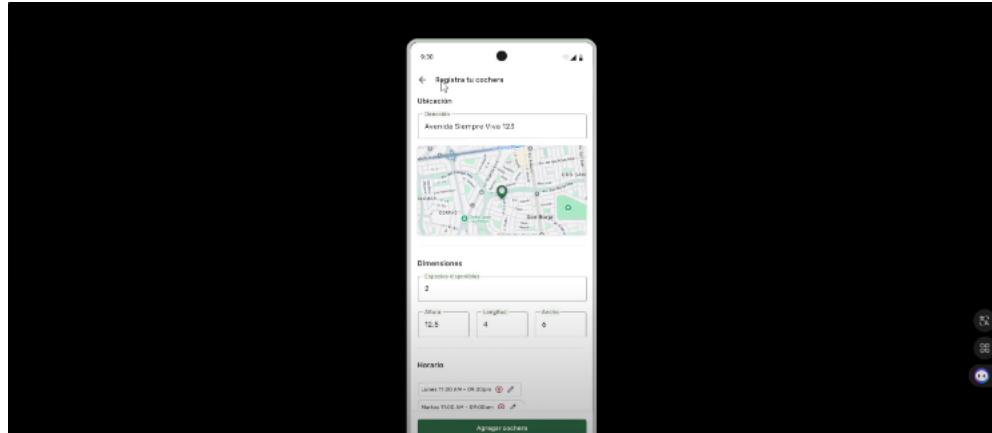
## 4.5. Mobile Applications Prototyping

### 4.5.1. Android Mobile Applications Prototyping



[link del video del prototipo](#)

### 4.5.2. iOS Mobile Applications Prototyping



[link del video del prototipo](#)

## 4.6. Web Applications UX/UI Design

### 4.6.1. Web Applications Wireframes

## Inicia sesión

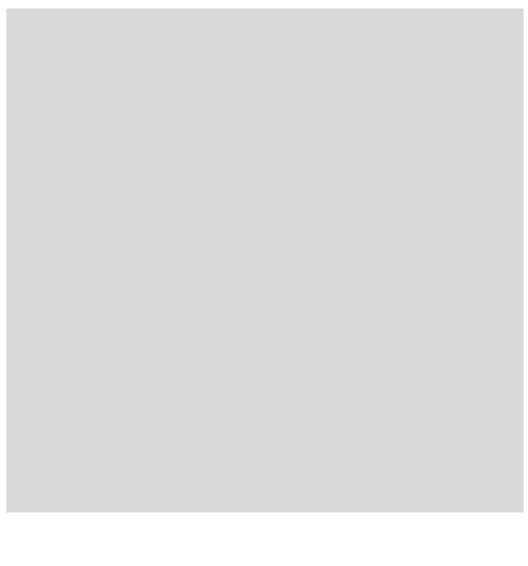
Ingresá a tu cuenta y empieza a reservar estacionamientos.

Email

Contraseña

[Iniciar sesión](#)

[¿No tienes una cuenta? Regístrate Aquí](#)



## Crear una cuenta nueva

Tu vehículo seguro, tu espacio garantizado: Encuentra el lugar perfecto para estacionar

Email

Nombres

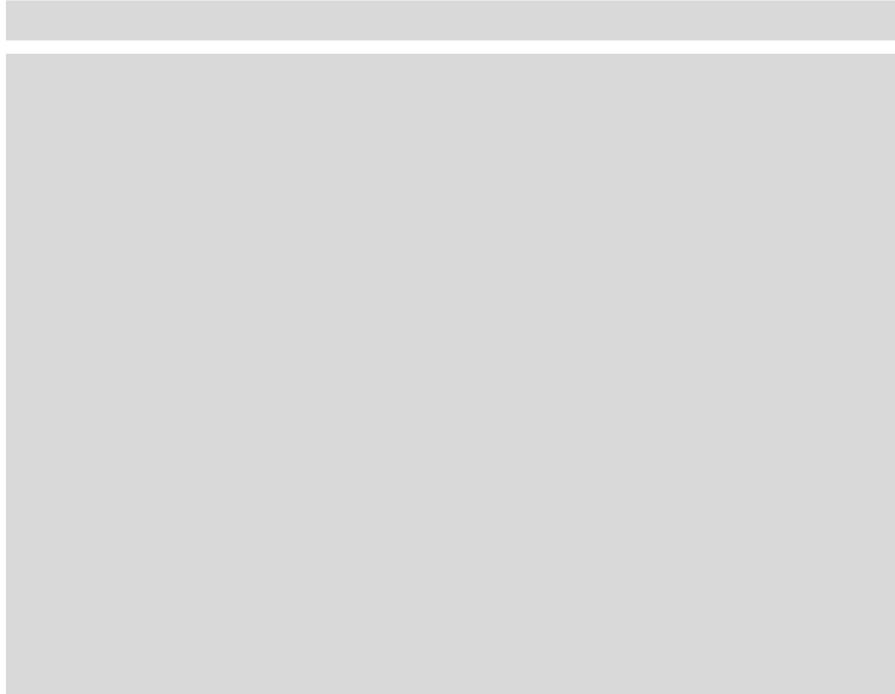
Apellidos

Contraseña

Repetir Contraseña

Crear cuenta

[¿Ya tienes una cuenta? Inicia Sesión](#)



Encuentra tu garaje

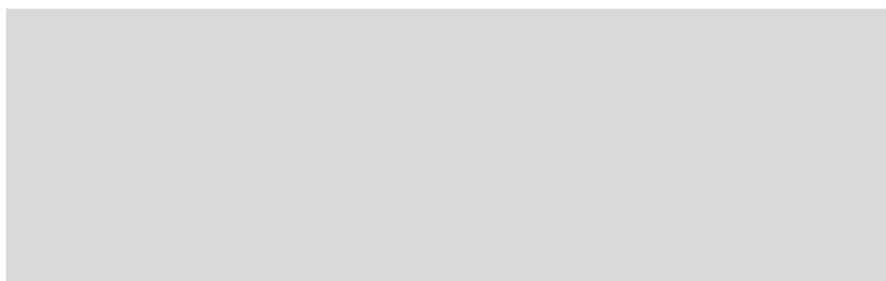
**Explora**

- Buscar un garage
- Mis reservas

**Renta un garaje**

- Mis garajes
- Reservas entrantes

usuario  
correo@test.com



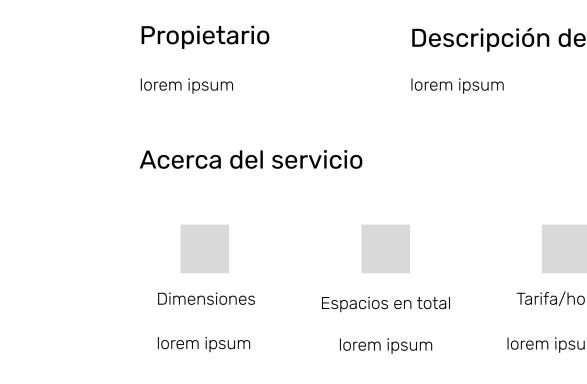
Detalle de garaje

**Explora**

- Buscar un garage
- Mis reservas

**Renta un garaje**

- Mis garajes
- Reservas entrantes



Propietario	Descripción del garaje
lorem ipsum	lorem ipsum

Acerca del servicio

Dimensiones	Espacios en total	Tarifa/hora
lorem ipsum	lorem ipsum	lorem ipsum

usuario  
correo@test.com



usuario  
correo@test.com



usuario  
correo@test.com

**Explora**

- Buscar un garaje
- Mis reservas

**Renta un garaje**

- Mis garajes
- Reservas entrantes

**Registra tu garaje**

Ubicación

Dimensions

Ancho (m)	Largo (m)	Alto (m)

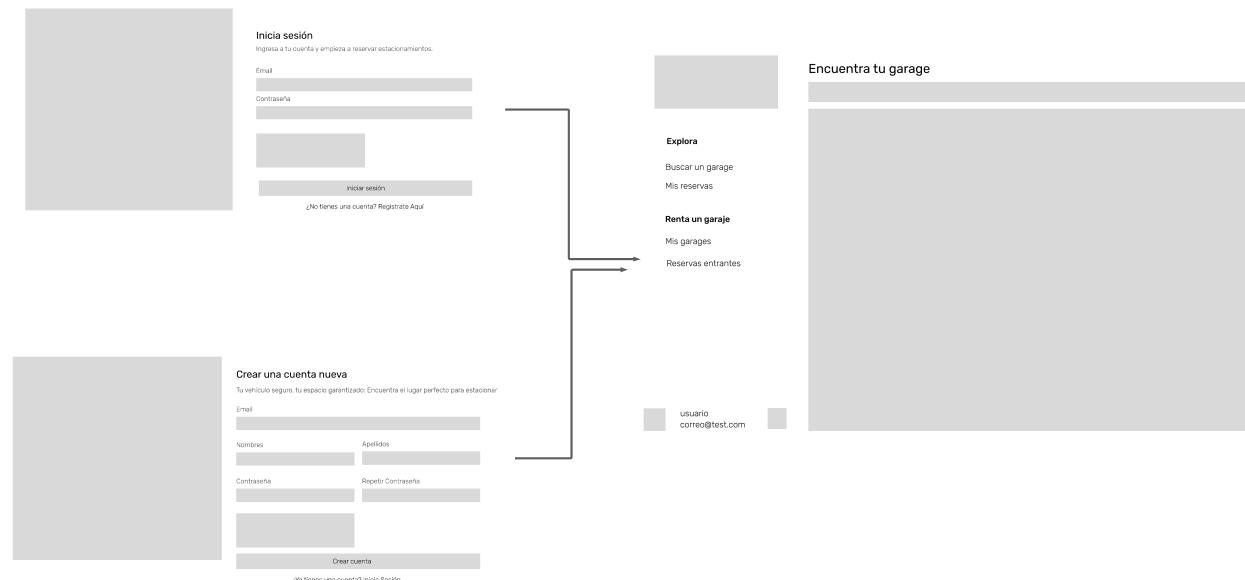
Descripción

usuario  
correo@test.com

**agregar**

#### 4.6.2. Web Applications Wireflow Diagrams

**User goal:** El usuario se registra en la plataforma e inicia sesión en la aplicación con sus datos.



Descripción: Al iniciar la aplicación, el usuario se encuentra con el apartado de registro de cuenta, donde podrá registrarse con su email y contraseña. Cuando le de al botón de registrarse, le mandará al apartado de inicio de sesión, donde podrá ingresar sus datos anteriormente añadidos y al ser las correctas, podrá acceder a la página principal de la plataforma.

**User goal:** Usuario desea hacer la búsqueda y reserva de un espacio de estacionamiento



Descripción: El usuario al encontrarse en la página principal, podrá visualizar y buscar estacionamientos de su preferencia. Al seleccionar un estacionamiento, verá el detalle de este, aquí el usuario contará con dos opciones, ver los comentarios y calificación o seleccionar la opción de reservar ahora. El usuario al ingresar a la sección de calificación visualizará los comentarios que ha obtenido dicho estacionamiento, así como también la calificación en general del mismo. Por otro lado, luego de seleccionar la opción de reservar, el estacionamiento habrá quedado apartado y podrá revisarlo en la opción de historial del menú principal, donde también podrá cancelar su reserva o revisar sus reservas pasadas.

#### 4.6.3. Web Applications Mock-ups



**Inicia sesión**

Ingresa a tu cuenta y empieza a reservas estacionamientos.

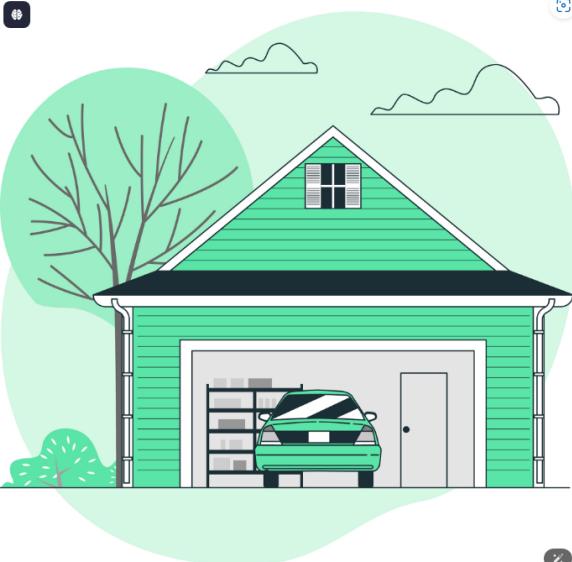
Email

Contraseña  [Olvidaste tu contraseña?](#)

I'm not a robot reCAPTCHA  
Privacy - Terms

**Iniciar sesión**

[¿No tienes una cuenta? Regístrate aquí](#)



**Crear una cuenta nueva**

Tu vehículo seguro, tu espacio garantizado: Encuentra el lugar perfecto para estacionar.

Email

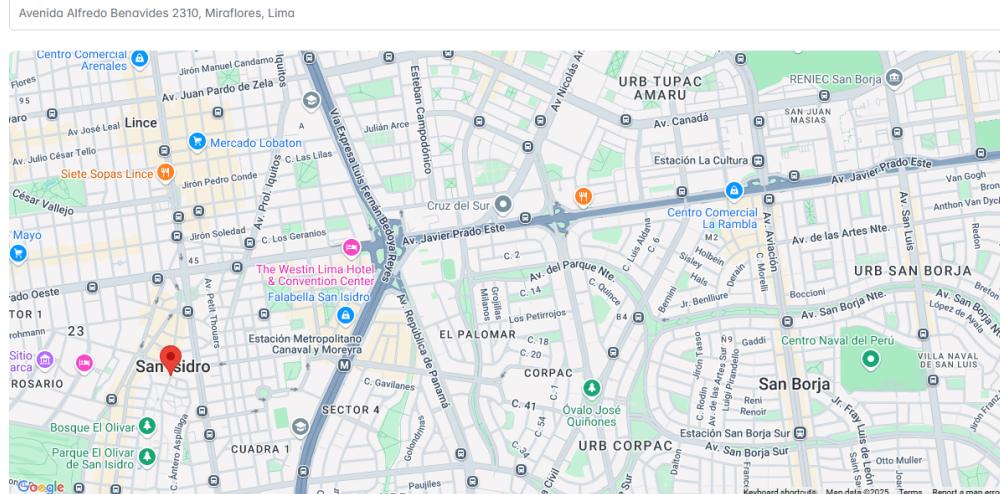
Nombres  Apellidos

Contraseña  Repetir contraseña

I'm not a robot reCAPTCHA  
Privacy - Terms

**Crear cuenta**

[¿Ya tienes una cuenta? Iniciar sesión](#)



**Encuentra tu garaje**

Avenida Alfredo Benavides 2310, Miraflores, Lima

**Explora**

- Buscar un garaje
- Mis reservas
- Renta un garaje
- Mis garajes
- Reservas entrantes

Lucio Heli [lucloyen@gmail.com](mailto:lucloyen@gmail.com)

**Mostrar información ▲**

 Detalle de garaje



**Av. Pardo y Aliaga 640**  
San Isidro, Lima

**Explora**

- Buscar un garage
- Mis reservas
- Renta un garaje**
- Mis garajes
- Reservas entrantes

**Propietario**

Daniel Chirinos  
Se unió a HomeyPark desde el 01 April 2025

**Descripción del garaje**

Secure parking space in the heart of the city

**Acerca del servicio**

Dimensiones  
Largo: 22m  
Ancho: 20m  
Alto: 35m



Espacios en total  
Hasta 10 vehículos



Tarifa/Hora  
S/ 80.00

Lucio Heli  
lucioyen1@gmail.com



YourReservationsPage

**Explora**

- Buscar un garage
- Mis reservas
- Renta un garaje**
- Mis garajes
- Reservas entrantes

Lucio Heli  
lucioyen1@gmail.com



Ocultar información ▾

© 2025 HomeyPark. Todos los derechos reservados.

[Facebook](#) [Twitter](#) [Instagram](#)

**Mis estacionamientos guardados**

+ Agregar

**Explora**

- Buscar un garage
- Mis reservas
- Renta un garaje**
- Mis garajes
- Reservas entrantes

Lucio Heli  
lucioyen1@gmail.com



**HomeyPark**

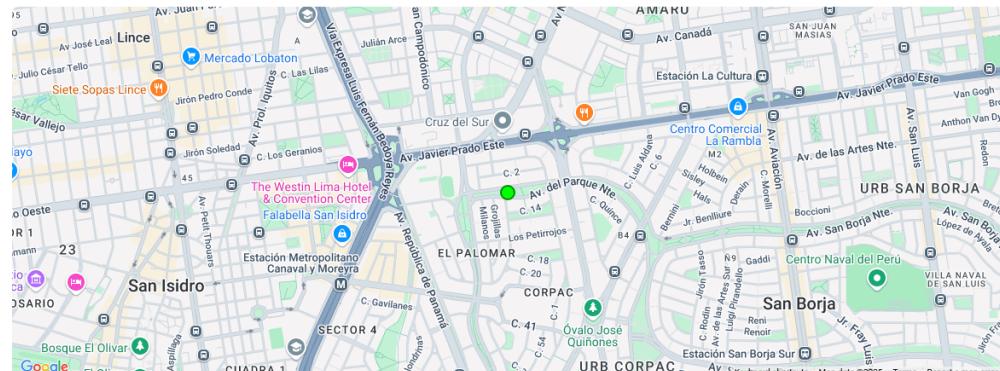
**Explora**

- Buscar un garaje
- Mis reservas
- Renta un garaje**
- Mis garajes
- Reservas entrantes

### Registra tu garage

Ubicación

Dirección



Dimensions

Espacios disponibles	Ancho (m)	Largo (m)	Alto (m)
0 unidad(es)	0 m	0 m	0 m

**Mis estacionamientos guardados**



test

 Borrar
 Editar

Lucio Heli  
lucioyen1@gmail.com

 Agregar

#### 4.6.4. Web Applications User Flow Diagrams

**User goal:** El usuario se registra en la plataforma e inicia sesión en la aplicación con sus datos.

**Create a new account**

Tu vehículo seguro, tu espacio garantizado: Encuentra el lugar perfecto para estacionar.

Email:

Nombres: \_\_\_\_\_ Apellidos: \_\_\_\_\_

Contraseña: \_\_\_\_\_ Repetir contraseña: \_\_\_\_\_

I'm not a robot

**Crear cuenta**

¿No tienes una cuenta? [Iniciar sesión](#)

**Iniciar sesión**

Ingresá o tu cuenta y empieza a reservas estacionamientos.

Email: \_\_\_\_\_

Contraseña: \_\_\_\_\_

I'm not a robot

**Iniciar sesión**

¿No tienes una cuenta? [Regístrate aquí](#)

**Descripción:** Al iniciar la aplicación web, el usuario se encuentra con el apartado de registro de cuenta, donde podrá registrarse con su email y contraseña. Cuando le de al botón de registrarse, le mandará al apartado de inicio de sesión, donde podrá ingresar sus datos anteriormente añadidos y al ser las correctas, podrá acceder a la página principal de la plataforma.

**User goal:** Usuario desea hacer la búsqueda y reserva de un espacio de estacionamiento

**Encuentra tu garaje**

Avenida Alfredo Benavides 2330, Miraflores, Lima

**Explora**

- Buscar un garaje
- Mis reservas
- Renta un garaje
- Mis garajes
- Reservas entradas

Lucio Hall lucoyer@gmail.com

**Detalle del garaje**

**HomeyPark**

**Explora**

- Buscar un garaje
- Mis reservas
- Renta un garaje
- Mis garajes
- Reservas entradas

**Propietario**  
Daniel Chirinos  
Se une a Homeypark desde el 01 April 2025

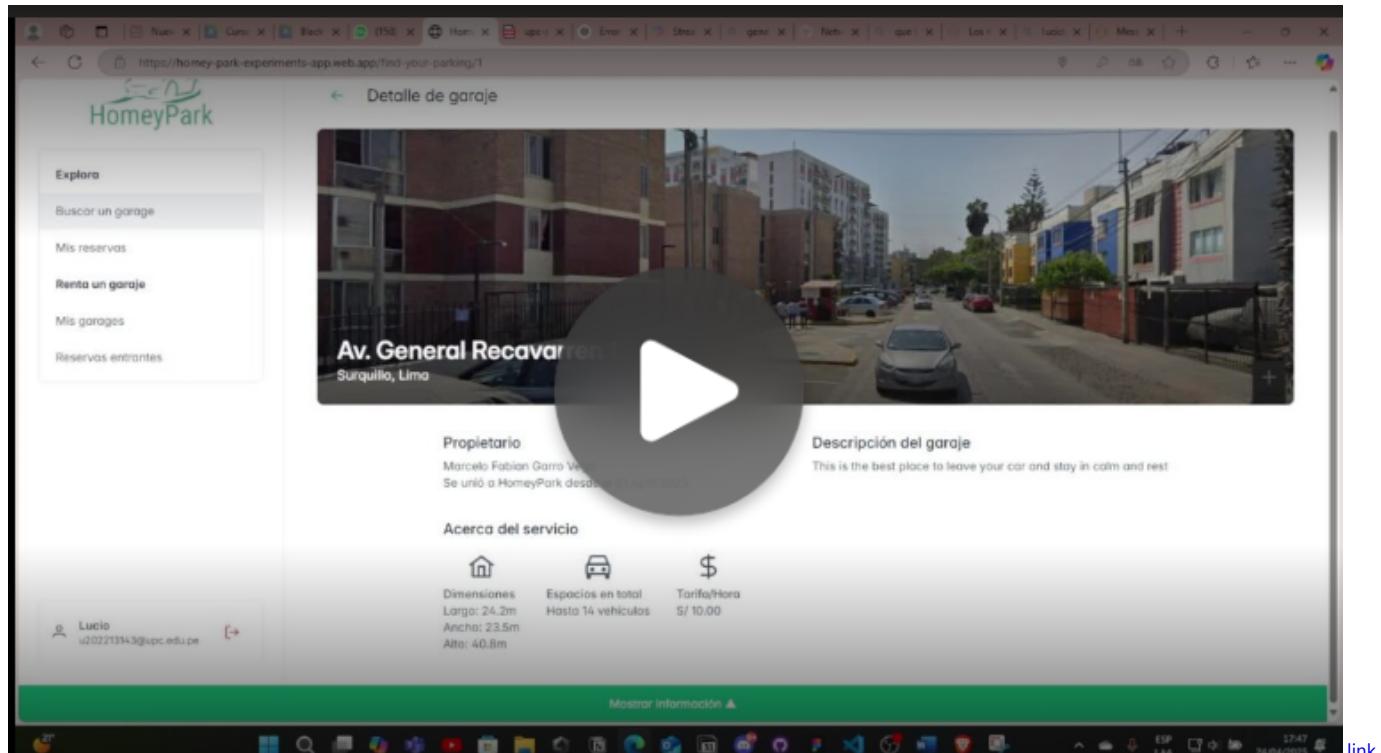
**Descripción del garaje**  
Secure parking space in the heart of the city

**Acero del servicio**

	Dimensiones
	Largo: 22m Ancho: 20m Altura: 3.5m
	Estacionamiento en total: Hasta 10 vehículos Tarifa/Hora: \$/ 80.00

**Descripción:** El usuario al encontrarse en la página principal, podrá visualizar y buscar estacionamientos de su preferencia. Al seleccionar un estacionamiento, verá el detalle de este, aquí el usuario contará con dos opciones, ver los comentarios y calificación o seleccionar la opción de reservar ahora. El usuario al ingresar a la sección de calificación visualizará los comentarios que ha obtenido dicho estacionamiento, así como también la calificación en general del mismo. Por otro lado, luego de seleccionar la opción de reservar, el estacionamiento quedará apartado y podrá revisarlo en la opción de historial del menú principal, donde también podrá cancelar su reserva o revisar sus reservas pasadas.

#### 4.7. Web Applications Prototyping

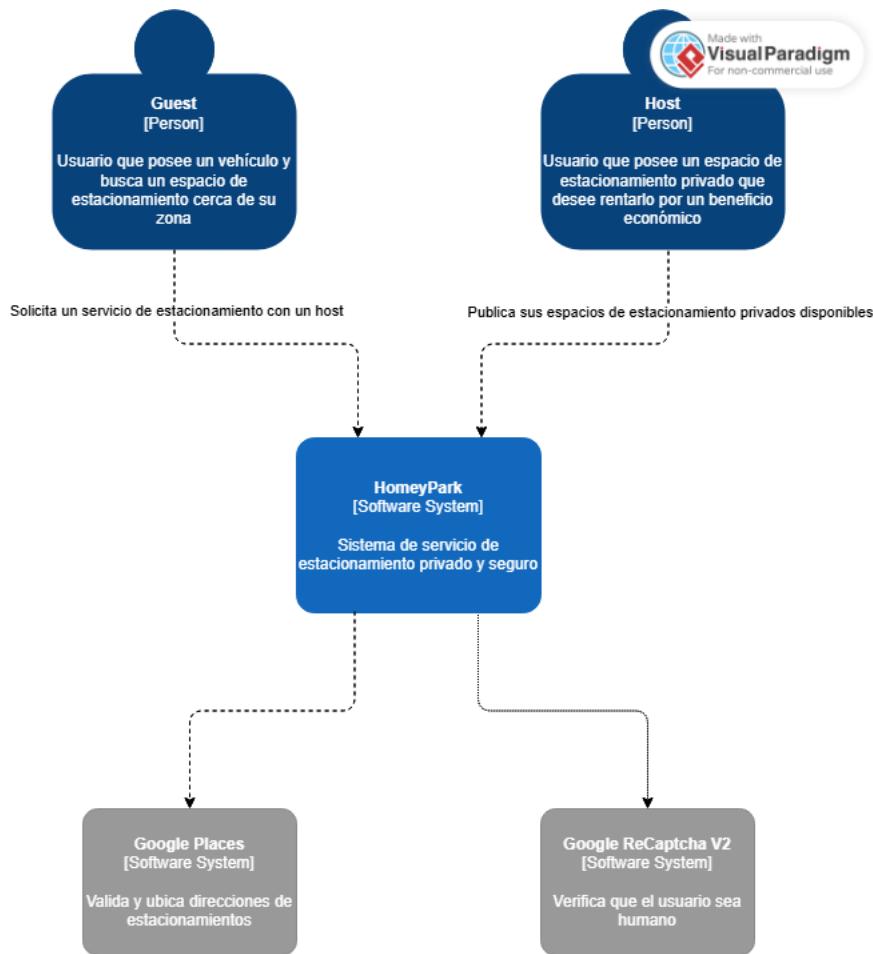


del video del prototipo

## 4.8. Domain-Driven Software Architecture

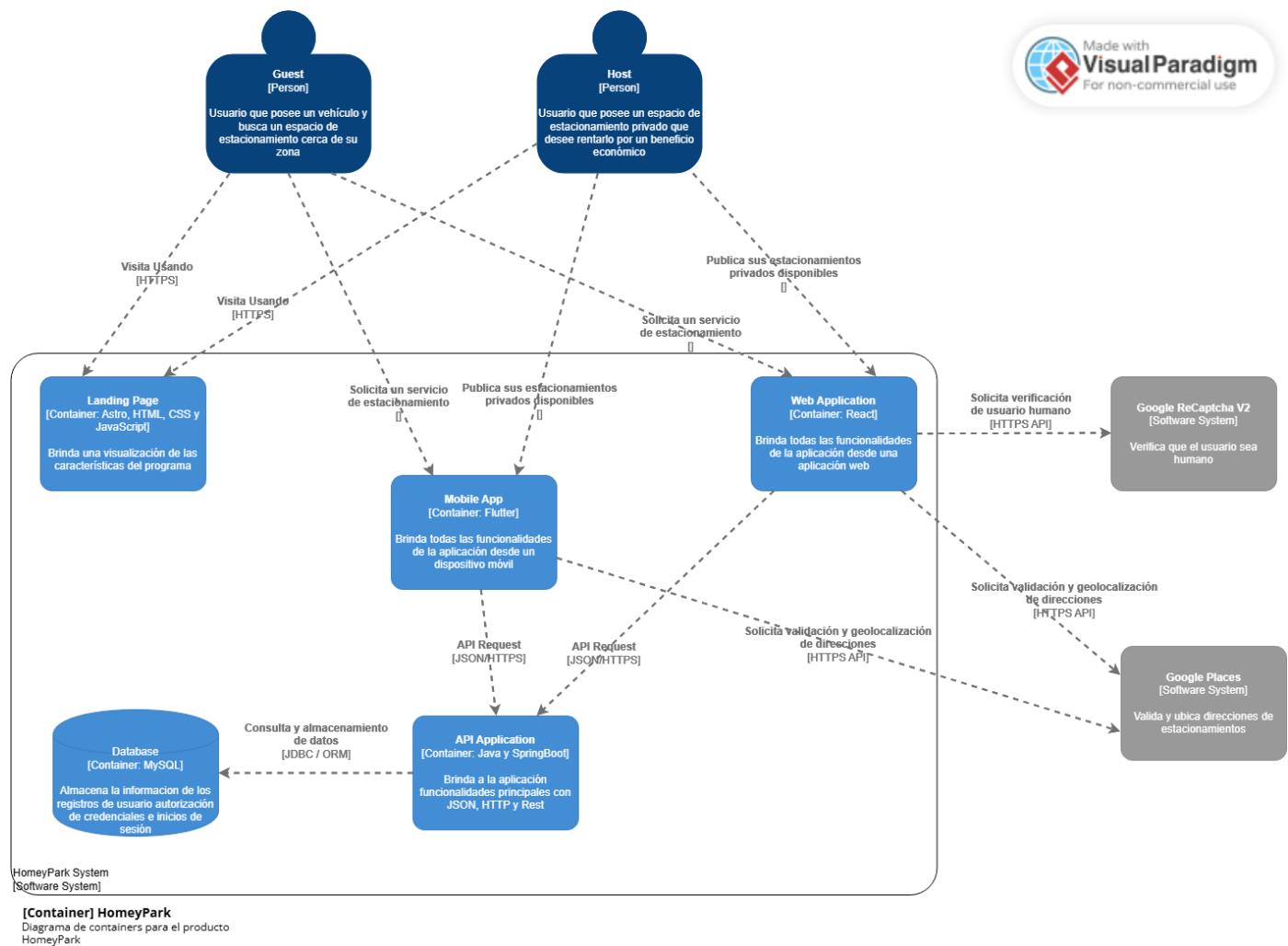
### 4.8.1. Software Architecture Context Diagram

Este diagrama contextualiza el sistema HomeyPark en su entorno, mostrando cómo interactúan los actores principales (Guest y Host) con el sistema central y qué servicios externos (Google Places y Google ReCaptcha V2) son utilizados para brindar funcionalidades complementarias como geolocalización y verificación de usuarios humanos.



#### 4.8.2. Software Architecture Container Diagrams

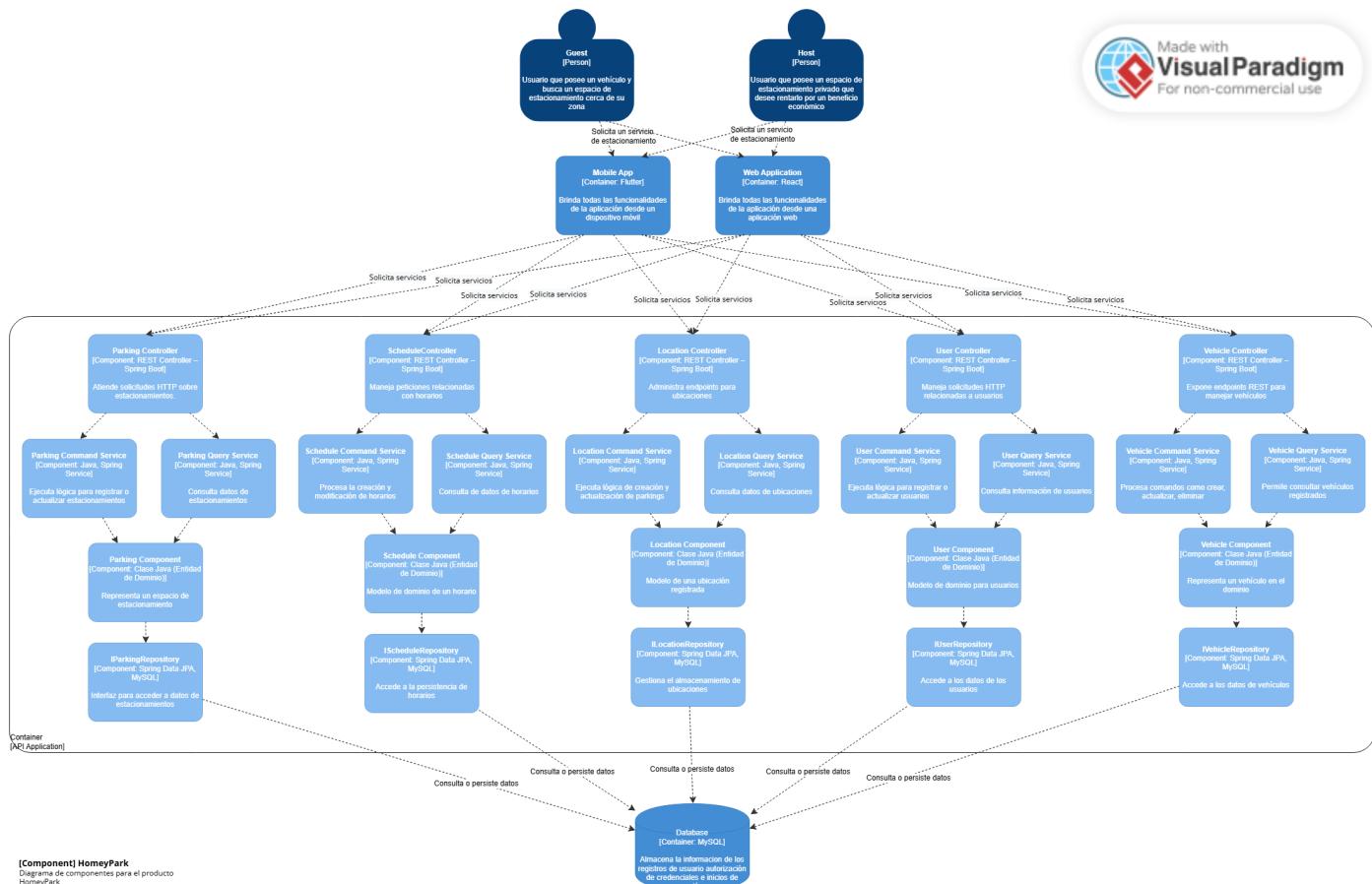
El diagrama de contenedores descompone el sistema en sus principales contenedores de ejecución. Se ilustran los distintos frontends (Mobile App y Web Application), la lógica de backend central implementada en Spring Boot, y los sistemas de almacenamiento y servicios externos. Se enfatiza el rol de la API Application como núcleo funcional del sistema.



#### 4.8.3. Software Architecture Components Diagrams

Este diagrama muestra la estructura interna de la API Application, organizada según Bounded Contexts definidos por el modelo de dominio. Cada contexto encapsula sus propios componentes siguiendo una Clean Architecture, donde se separan claramente las responsabilidades entre controladores, servicios de aplicación, modelos del dominio y repositorios.

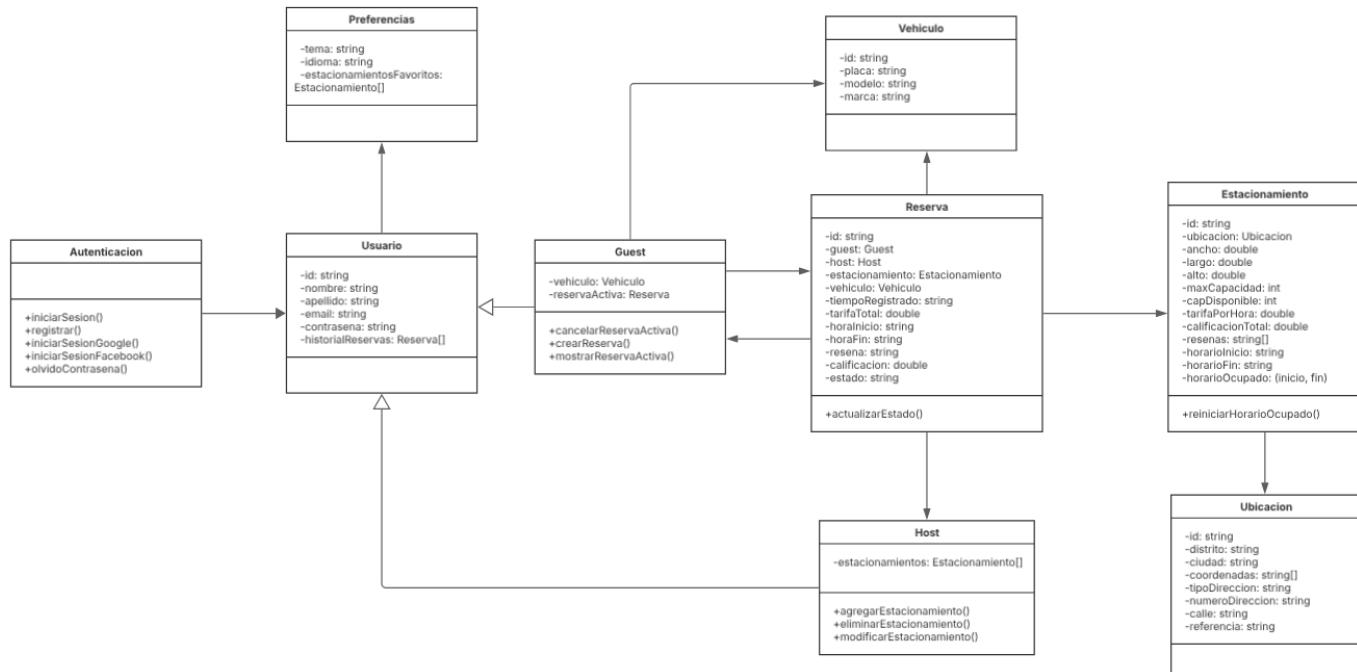
Además, se implementa el patrón CQRS (Command Query Responsibility Segregation), dividiendo explícitamente la lógica de lectura (Query Services) y escritura (Command Services), lo que mejora la escalabilidad y claridad del sistema. Esta arquitectura permite una evolución modular, mayor mantenibilidad y prepara el sistema para una futura transición a microservicios si fuera necesario.



## 4.9. Software Object-Oriented Design

### 4.9.1. Class Diagrams

El siguiente diagrama muestra las clases principales del sistema HomeyPark, incluyendo sus atributos, métodos y relaciones. Este modelo fue construido a partir de los escenarios del dominio y la funcionalidad esperada por los usuarios del sistema.



### 4.9.2. Class Dictionary

A continuación se presenta el diccionario de clases con los atributos, tipos y descripciones de las clases más relevantes del modelo.

#### Clase Usuario

Atributo	Tipo	Descripción
id	String	Código para el usuario
nombre	String	Nombre del usuario
apellido	String	Apellido del usuario
email	String	Correo del usuario
contrasena	String	Contraseña del usuario
historialReservas	Reserva[]	Lista de reservas hechas por el usuario

#### Clase Preferencias

Atributo	Tipo	Descripción
tema	String	Tema de visualización
idioma	String	Idioma preferido
estacionamientosFavoritos	Estacionamiento[]	Estacionamientos favoritos

#### Clase Vehículo

Atributo	Tipo	Descripción
id	String	Identificador del vehículo
placa	String	Placa del vehículo
modelo	String	Modelo del vehículo
marca	String	Marca del vehículo

#### Clase Guest

Atributo	Tipo	Descripción
vehiculo	Vehiculo	Vehículo asignado
reservaActiva	Reserva	Reserva actualmente activa

#### Clase Host

Atributo	Tipo	Descripción
estacionamientos	Estacionamiento[]	Estacionamientos administrados

#### Clase Reserva

Atributo	Tipo	Descripción
id	String	Identificador de la reserva
guest	Guest	Usuario que reserva
host	Host	Propietario del espacio
estacionamiento	Estacionamiento	Espacio reservado
vehiculo	Vehiculo	Vehículo asociado
tiempoRegistrado	String	Hora de registro
tarifaTotal	double	Costo total
horainicio	String	Inicio de la reserva
horaFin	String	Fin de la reserva
resena	String	Reseña del usuario
calificacion	double	Puntaje otorgado
estado	String	Estado de la reserva

#### Clase Estacionamiento

Atributo	Tipo	Descripción
id	String	Identificador del espacio
ubicacion	Ubicacion	Ubicación física

Atributo	Tipo	Descripción
ancho	double	Ancho del espacio
largo	double	Largo del espacio
alto	double	Altura del espacio
maxCapacidad	int	Capacidad máxima
capDisponible	int	Capacidad disponible
tarifaPorHora	double	Precio por hora
calificacionTotal	double	Promedio de calificaciones
resenas	String[]	Lista de reseñas
horarioInicio	String	Inicio de disponibilidad
horarioFin	String	Fin de disponibilidad
horarioOcupado	(inicio, fin)	Horas ocupadas

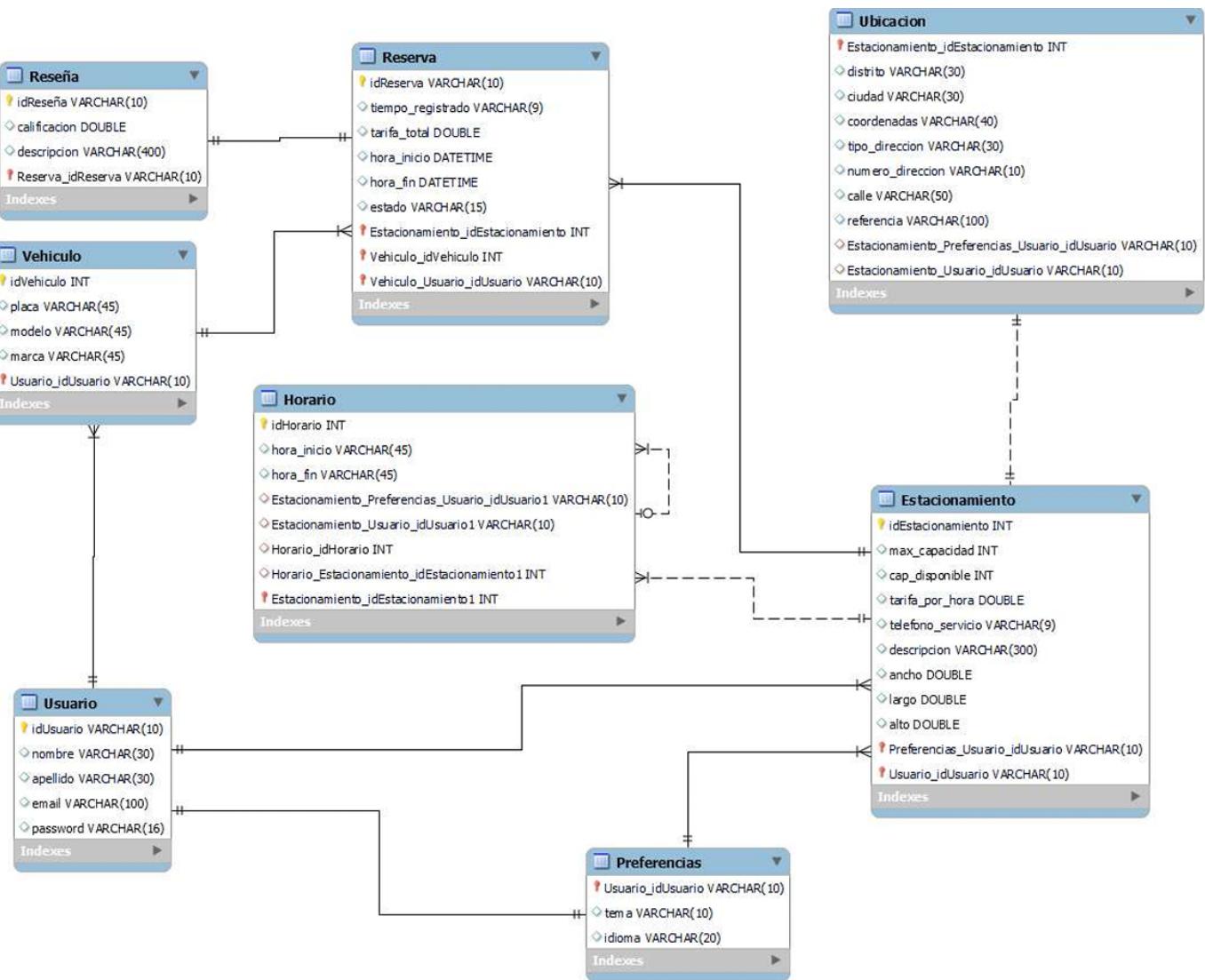
### Clase Ubicacion

Atributo	Tipo	Descripción
id	String	Identificador de ubicación
distrito	String	Distrito
ciudad	String	Ciudad
coordenadas	String[]	Latitud y longitud
tipoDireccion	String	Tipo de vía
numeroDireccion	String	Número del inmueble
calle	String	Nombre de la calle
referencia	String	Referencia adicional

## 4.10. Database Design

### 4.10.1. Relational/Non-Relational Database Diagram

Este diagrama representa el modelo lógico relacional de HomeyPark, diseñado para reflejar las entidades clave del dominio y sus relaciones. Cada tabla está normalizada y vinculada mediante claves primarias y foráneas que garantizan la integridad referencial del sistema. La base de datos incluye entidades como Usuario, Vehiculo, Estacionamiento, Reserva, Ubicacion, Horario, Reseña y Preferencias, cada una con un propósito específico para cubrir las operaciones de registro, reserva y evaluación dentro del sistema.



## Capítulo V: Product Implementation

### 5.1. Software Configuration Management

#### 5.1.1. Software Development Environment Configuration

Project management

**Discord.** Utilizamos esta aplicación como medio de comunicación para mantener una comunicación abierta entre el equipo y debido a su facilidad de uso para realizar llamadas, compartir pantalla y dividir por salas para una mejor organización.

**Trello.** Usamos esta herramienta de gestión de tareas por tarjetas debido a su facilidad y mostrar visibilidad del progreso en el desarrollo de actividades del equipo por integrantes. **WhatsApp.** Principal medio de comunicación para mantenernos conectados por medio de mensajes de texto, imágenes y videos. Esta herramienta es utilizada con mayor frecuencia.

Requirements Management

Trello. Aplicación web y/o móvil para organizar los requerimientos a llevar durante los sprints del proyecto. También se utiliza para realizar seguimiento de las tareas o requerimientos de los integrantes.

Software Development

**Android Studio.** IDE de desarrollo para aplicaciones móviles de múltiples plataformas soportado por el equipo de Google y basado en IntelliJ IDEA.

**Figma.** Herramienta de desarrollo de interfaces para usuario (UI). Principal aplicación para desarrollar las interfaces y vistas de nuestros productos landing page y aplicación móvil.

**IntelliJ IDEA.** IDE de desarrollo diseñado principalmente en soluciones software basadas en el lenguaje de programación Java. Esta herramienta la usamos para desarrollar servicios API REST con el framework de Spring Boot.

**WebStorm.** IDE completo que incluye herramientas para crear, editar y depurar código en JavaScript, HTML, CSS, y otras tecnologías web.

Software Deployment

Google Play Console. La plataforma de Google permite subir aplicaciones móviles en producción dentro del catálogo de Play Store.

### 5.1.2. Source Code Management

Para llevar a cabo una mejor gestión y desarrollo en equipo trabajaremos bajo la metodología de Git Flow y convenciones para los commits, la cual se basa en desarrollo de tareas y requerimientos en ramas especiales sin afectar la rama principal de nuestro repositorio ni el progreso de los demás desarrolladores.

#### Flujo de trabajo en Git

##### Rama principal

- Rama donde se encontrará el código puesto en producción, listo para el uso de los usuarios finales.

##### Rama de desarrollo

- Rama donde estarán los últimos desarrollos de requerimientos listos para llevar a producción.

##### Ramas auxiliares

- Ramas de *features*: En esta rama se llevarán a cabo los requerimientos asignados por integrante sin repercutir en sus desarrollos.
- Ramas de *hotfix*: En estas ramas se encargan principalmente de resolver bugs o incidencias en caliente.
- Ramas de *release*: En estas ramas se lleva a cabo el último paso donde se validará y resolverá posibles incidencias antes de subir a producción el desarrollo.

#### Convenciones de commits

Nuestro equipo de desarrolladores trabajará bajo las buenas prácticas de los commits para facilitar la redacción e identificación del impacto en nuestro producto software.

`<type>(scope): <description>`

**type:** Campo obligatorio donde se especifica el tipo de cambio realizado. Los tipos de commits son los siguientes:

- **feat**: Introduce una nueva funcionalidad en el código.
- **fix**: Soluciona un error en el código.
- **style**: Modifica aspectos estéticos del proyecto, como formatear los archivos.
- **refactor**: Mejora el código sin añadir nuevas funcionalidades. Esto puede incluir la implementación de buenas prácticas.
- **docs**: Realiza cambios en la documentación sin impactar las funcionalidades del proyecto.
- **build**: Ajusta la configuración del proyecto, como agregar, eliminar o modificar dependencias.

**scope:** Campo opcional que indica el alcance del commit, incluyendo los identificadores de historias de usuario o requisitos.

**description:** Campo obligatorio que ofrece un resumen breve del commit, en inglés y comenzando con un verbo en infinitivo.

#### Convenciones para versionamiento de lanzamientos

Para el desarrollo del proyecto, se utilizará el modelo de versionamiento "Semantic Versioning 2.0.0" (<https://semver.org/>). Los commits y tags de los lanzamientos seguirán el siguiente formato:

- Release vX.Y.Z

Donde:

- X: Indica un cambio de versión mayor (MAJOR). Es usado principalmente para realizar cambios significativos con respecto a versiones anteriores.
- Y: Indica un cambio de versión menor (MINOR). Este tipo de lanzamiento incluye funcionalidades adicionales o mejoras al producto final.
- Z: Indica la versión del parche (PATCH). No alteran las funcionalidades del producto, solo se encarga de corregir errores.

### 5.1.3. Source Code Style Guide & Conventions

En la siguiente sección estaremos detallando las nomenclaturas para los siguientes lenguajes de programación y frameworks a utilizar a lo largo del proyecto.

- Gherkin
- Las especificaciones deben ser claras y fáciles de leer.
- Evitar el uso de términos técnicos para asegurar una mejor comprensión entre los colaboradores.
- Utilizar las palabras clave Given, When, Then, And, y But para describir el comportamiento del sistema.
- Evitar la redundancia en la descripción de los escenarios.
- Java
- Uso de camelCase para nombre variables y métodos
- Uso de PascalCase para nombrar clases, interfaces, record y otras estructuras.
- Importaciones explícitas (evitar importaciones con \*)
- Estructura de clases, organizar y agrupar los atributos, constructores y métodos.
- Usar indentación de dos espacios.
- La longitud de una línea no debe superar los 100 caracteres.

### 5.1.4. Software Deployment Configuration

Para el despliegue de nuestro servicio API REST usaremos la plataforma de alojamiento Render a través de contenedores de Docker para subir nuestras aplicaciones ubicadas en nuestro repositorio de GitHub.

Enlace al repositorio: <https://github.com/NetviaOrganization/HomeyPark-Web-Service>

- Como primer paso, crearemos nuestro proyecto en un nuevo repositorio de GitHub llamado "HomeyPark-Web-Service"

The screenshot shows the GitHub repository 'HomeyPark-Web-Service'. At the top, there are navigation links: Code, Issues, Pull requests, Actions, Projects, Security, Insights, and Settings. Below the header, the repository name 'HomeyPark-Web-Service' is displayed with a green icon, followed by 'Public'. On the right, there are 'Edit Pins' and 'Watch' buttons. The main area shows the 'main' branch with 2 branches and 0 tags. A search bar 'Go to file' and a 't' button are on the right. Below the search bar, a green 'Add file' button and a 'Code' dropdown are visible. The commit history lists 2 commits from 'AdrianoSCruzP':

- feat(user): add email field to User entity (04d337e · 3 days ago)
- feat(iam,shared): add bounded contexts for IAM and Shared ... (3 days ago)
- feat(user): add email field to User entity (3 days ago)
- feat(iam,shared): add bounded contexts for IAM and Shared ... (3 days ago)
- feat(iam,shared): add bounded contexts for IAM and Shared ... (3 days ago)
- feat(iam,shared): add bounded contexts for IAM and Shared ... (3 days ago)
- feat(iam,shared): add bounded contexts for IAM and Shared ... (3 days ago)
- pom.xml (3 days ago)

- Seguido de ello, ingresamos a la plataforma de Render y crearemos un nuevo servicio web.

The screenshot shows the Render platform interface. At the top, there is a search bar 'Search ^ K' and a '+ New' button. To the right, a sidebar lists service types:

- Static Site
- Web Service
- Private Service
- Background Worker
- Cron Job

---

- Postgres
- Key Value

---

- Project
- Blueprint

- Private Services**: Web app hosted on a private network, accessible only from your other Render services.  
New Private Service →
- Background W**: Long-lived services that handle tasks, usually from a job.  
New Worker →

- Configuraremos el nuevo servicio conectándolo con nuestro repositorio de GitHub e indicaremos trabajar con el lenguaje Docker, lo que nos permitirá usar contenedores con Java y Spring Boot.

## You are deploying a Web Service

[Source Code](#)

 NetviaOrganization / HomeyPark-Web-Service • 3d ago [Edit](#)

---

**Name**  
A unique name for your web service.

**HomeyPark-Web-Service**

**Project** Optional  
Add this web service to a project once it's created.

 Create a new project to add this to?

You don't have any projects in this workspace. Projects allow you to group resources into environments so you can better manage related resources.

[+ Create a project](#)

---

**Language**  
Docker

**Branch**  
The Git branch to build and deploy.

main

**Region**  
Your services in the same region can communicate over a private network.

Oregon (US West)

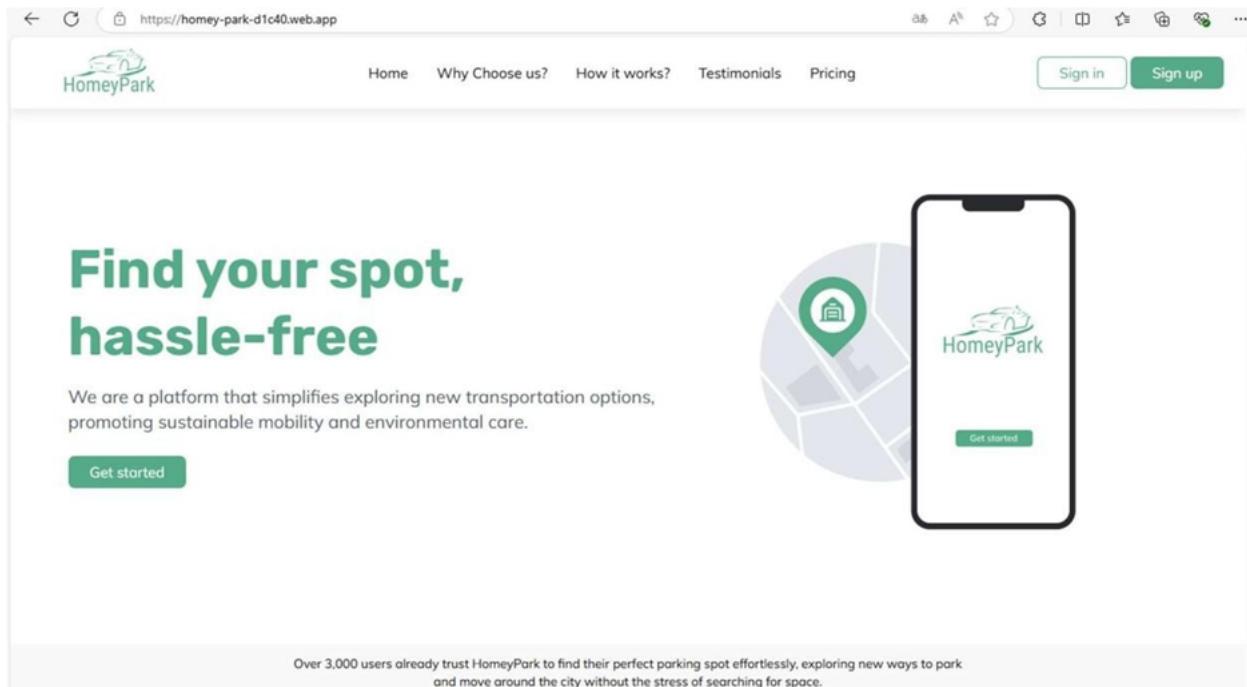
## 5.2. Product Implementation & Deployment

### 5.2.1. Sprint Backlogs

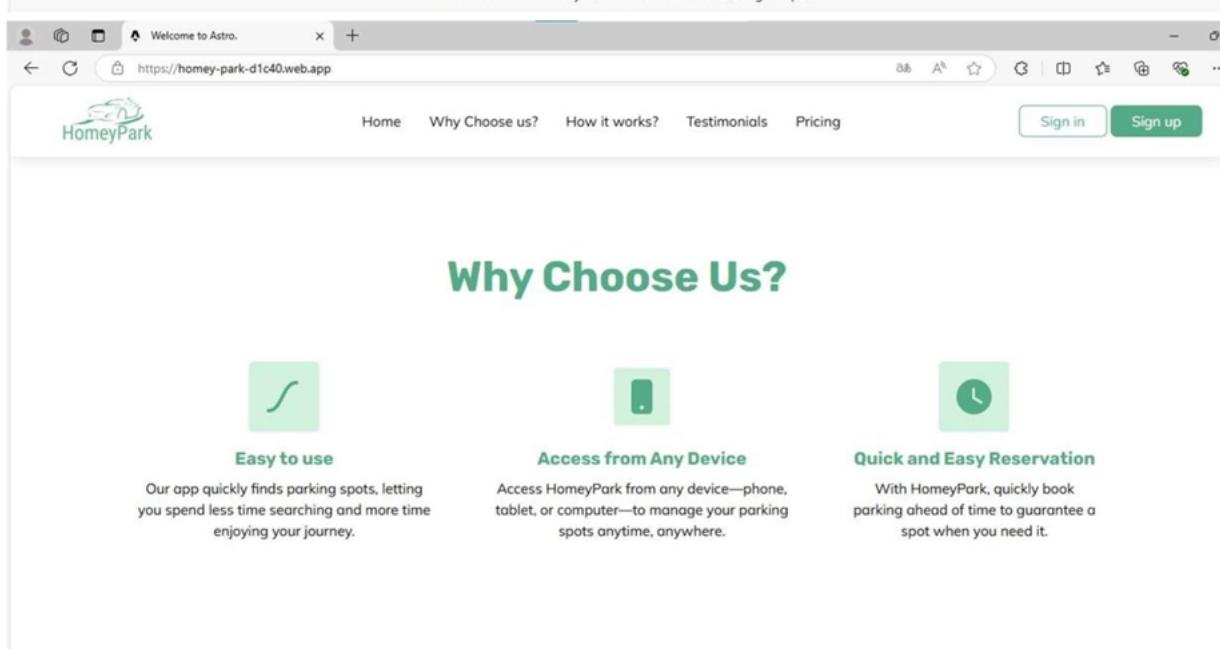
ID	Title	Title	Task ID	Title	Description	Estimation (Hours)	Assigned To	Status (To-do, In Process, To Review, Done, Cancelled)
US-	Integrar mapa de Google Maps	TASK-001				2	Sebastian Cachis	Done
US-	Búsqueda de estacionamiento por dirección	TASK-002				2	Adriano Cruz	Done
US-	Usar ubicación de dispositivo para buscar en mapa	TASK-003				1	Amner Llamo	Done
US-	Ver espacios de estacionamientos cercanos	TASK-004				2	Marcelo Garro	Done
US-	Visualizar detalles de estacionamiento	TASK-005				2	Lucio Yen	Done
US-	Visualizar el entorno del estacionamiento	TASK-006				2	Adriano Cruz	Done
US-	Registrar una cochera	TASK-007				3	Amner Llamo	Done
US-	Visualizar cocheras registradas	TASK-008				2	Marcelo Garro	Done
US-	Modificar cochera registrada	TASK-009				2	Lucio Yen	Done
US-	Eliminar la cochera	TASK-010				1	Sebastian Cachis	Done

### 5.2.2. Implemented Landing Page Evidence

Como resultado del primer sprint, se presenta el despliegue de la Landing Page



The screenshot shows the HomeyPark homepage. At the top, there's a navigation bar with links for Home, Why Choose us?, How it works?, Testimonials, Pricing, Sign in, and Sign up. The main heading is "Find your spot, hassle-free". Below it, a subtext reads: "We are a platform that simplifies exploring new transportation options, promoting sustainable mobility and environmental care." A prominent green "Get started" button is located below the subtext. To the right, there's a graphic of a smartphone displaying the HomeyPark app interface, which includes a map with a parking spot marked and a "Get started" button.



The screenshot shows the "Why Choose Us?" section of the HomeyPark website. It features three highlighted features: "Easy to use" (represented by a green square icon with a white line), "Access from Any Device" (represented by a green square icon with a white smartphone), and "Quick and Easy Reservation" (represented by a green square icon with a white clock). Each feature has a brief description below it.

Easy to use	Access from Any Device	Quick and Easy Reservation
Our app quickly finds parking spots, letting you spend less time searching and more time enjoying your journey.	Access HomeyPark from any device—phone, tablet, or computer—to manage your parking spots anytime, anywhere.	With HomeyPark, quickly book parking ahead of time to guarantee a spot when you need it.

The screenshot shows a web browser window with the URL <https://homey-park-d1c40.web.app>. The page title is "Welcome to Astro.". The main content area features the "HomeyPark" logo and navigation links for "Home", "Why Choose us?", "How it works?", "Testimonials", and "Pricing". There are also "Sign in" and "Sign up" buttons. The main heading is "How it works?".

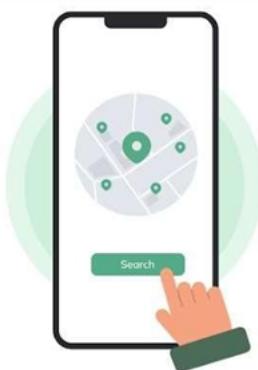
## 01 Search in your area

Start by searching in your area. HomeyPark uses your location to show available parking spots nearby, making it easy to find the most convenient options.



## 02 Choose a parking spot

Select your parking spot easily. After searching in your area, HomeyPark will present you with available options. Choose the one that fits your needs and preferences, and secure it with just a few taps.



## 03 Reserve your parking spot

Reserve your parking spot with ease. Once you've selected your preferred location, confirm your reservation in just a few clicks. HomeyPark ensures that your chosen spot is secured and ready for you when you arrive.



**Maria Perez**  
Lima, Peru ★ 5.0

"Excelente servicio! Los estacionamientos son seguros y el personal es muy amable. Recomiendo esta aplicación a todos los propietarios de viviendas en Lima."

**Carlos Rodriguez**  
Arequipa, Peru ★ 4.5

"Muy útil para encontrar estacionamientos cerca de mi casa. La aplicación es fácil de usar y los precios son razonables. ¡Gracias!"

**Ana Gomez**  
Trujillo, Peru ★ 4.0

"Me encanta esta aplicación. Siempre encuentro estacionamientos disponibles y el proceso de reserva es muy sencillo."

**Pricing**

**Drivers****Hosts****5.2.3. Implemented Frontend-Web Application Evidence**

**Inicio sesión**  
Ingresa o crea una cuenta y empieza a reservar estacionamientos.

**Crear una cuenta nueva**  
Tu vehículo seguro, tu espacio perfecto. Encuentra el lugar perfecto para estacionar.

**Encuentra tu garaje**  
Avda. Miraflores 2310, Miraflores, Lima

**Datos de garaje**  
Av. General Recavarren 1300, San Isidro, Lima

**5.2.4. Acuerdo de Servicio - SaaS**

El presente Acuerdo de Servicio (en adelante, el "Acuerdo") regula los términos y condiciones aplicables al uso de la plataforma HomeyPark, un servicio de software como servicio (SaaS) ofrecido por Netvia. Al acceder y utilizar los servicios provistos en la plataforma web y móvil de HomeyPark, el usuario acepta quedar legalmente vinculado por los términos de este Acuerdo.

## 1. Alcance del Servicio

HomeyPark es una solución SaaS que permite a conductores (en adelante, "Usuarios de Parking") buscar, reservar y pagar por espacios de estacionamiento ofrecidos por terceros (en adelante, "Anfitriones") a través de una interfaz digital. Asimismo, permite a los Anfitriones registrar y gestionar la disponibilidad de sus cocheras privadas.

## 2. Derechos y Obligaciones del Usuario

- Los usuarios se comprometen a hacer uso del sistema de manera diligente, lícita y conforme a los fines para los cuales fue diseñado.
- El usuario garantiza que la información proporcionada durante el registro y uso del servicio es veraz, completa y actualizada.
- Los usuarios deberán respetar las condiciones específicas de uso y las normas internas establecidas por los Anfitriones respecto a sus cocheras.

## 3. Responsabilidades del Proveedor (Netvia)

- Netvia se compromete a mantener la disponibilidad del servicio, salvo casos de mantenimiento programado, fuerza mayor o fallas técnicas imprevistas.
- El proveedor no garantiza la disponibilidad de estacionamientos ni se responsabiliza por pérdidas, daños o robos ocurridos en el uso físico de los espacios.
- Se brindará soporte técnico razonable mediante canales digitales para incidencias con el uso de la plataforma.

## 4. Seguridad y Protección de Datos

- Toda la información personal será tratada conforme a las normas vigentes en materia de protección de datos personales en el Perú.
- HomeyPark implementa mecanismos de autenticación, encriptación y validación de usuarios para preservar la integridad del sistema y de sus usuarios.

## 5. Limitación de Responsabilidad

- HomeyPark actúa como un intermediario entre usuarios y anfitriones, por lo que no se hace responsable por el cumplimiento de los acuerdos entre ellos.
- El uso del servicio se realiza bajo riesgo propio del usuario. HomeyPark no asume responsabilidad por daños indirectos, incidentales o consecuentes.

## 6. Modificaciones al Servicio

- Netvia se reserva el derecho de actualizar, modificar o suspender funcionalidades de HomeyPark previa notificación a los usuarios a través de los canales registrados.

## 7. Terminación

- Cualquiera de las partes podrá dar por terminado el uso del servicio en cualquier momento. La terminación no exime del cumplimiento de obligaciones previamente contraídas.

## 8. Legislación Aplicable

- Este Acuerdo se regirá por las leyes de la República del Perú. Ante cualquier controversia, las partes se someten a la jurisdicción de los tribunales del distrito judicial de Lima Metropolitana.

**Términos y Condiciones**

El presente Acuerdo de Servicio (en adelante, el "**Acuerdo**") regula los términos y condiciones aplicables al uso de la plataforma HomeyPark, un servicio de software como servicio (SaaS) ofrecido por Netvia. Al acceder y utilizar los servicios provistos en la plataforma web y móvil de HomeyPark, el usuario acepta quedar legalmente vinculado por los términos de este Acuerdo.

**1. Alcance del Servicio**

HomeyPark es una solución SaaS que permite a conductores (en adelante, "**Usuarios de Parking**") buscar, reservar y pagar por espacios de estacionamiento ofrecidos por terceros (en adelante, "**Anfitriones**") a través de una interfaz digital. Asimismo, permite a los Anfitriones registrar y gestionar la disponibilidad de sus cocheras privadas.

**2. Derechos y Obligaciones del Usuario**

- Los usuarios se comprometen a hacer uso del sistema de manera diligente, lícita y conforme a los fines para los cuales fue diseñado.
- El usuario garantiza que la información proporcionada durante el registro y uso del servicio es veraz, completa y actualizada.
- Los usuarios deberán respetar las condiciones específicas de uso y las normas internas establecidas por los Anfitriones respecto a sus cocheras.

**3. Responsabilidades del Proveedor (Netvia)**

- Netvia se compromete a mantener la disponibilidad del servicio, salvo casos de mantenimiento programado, fuerza mayor o fallas técnicas imprevistas.
- El proveedor no garantiza la disponibilidad de estacionamientos ni se responsabiliza por pérdidas, daños o robos ocurridos en el uso físico de los espacios.
- Se brindará soporte técnico razonable mediante canales digitales para incidencias con el uso de la plataforma.

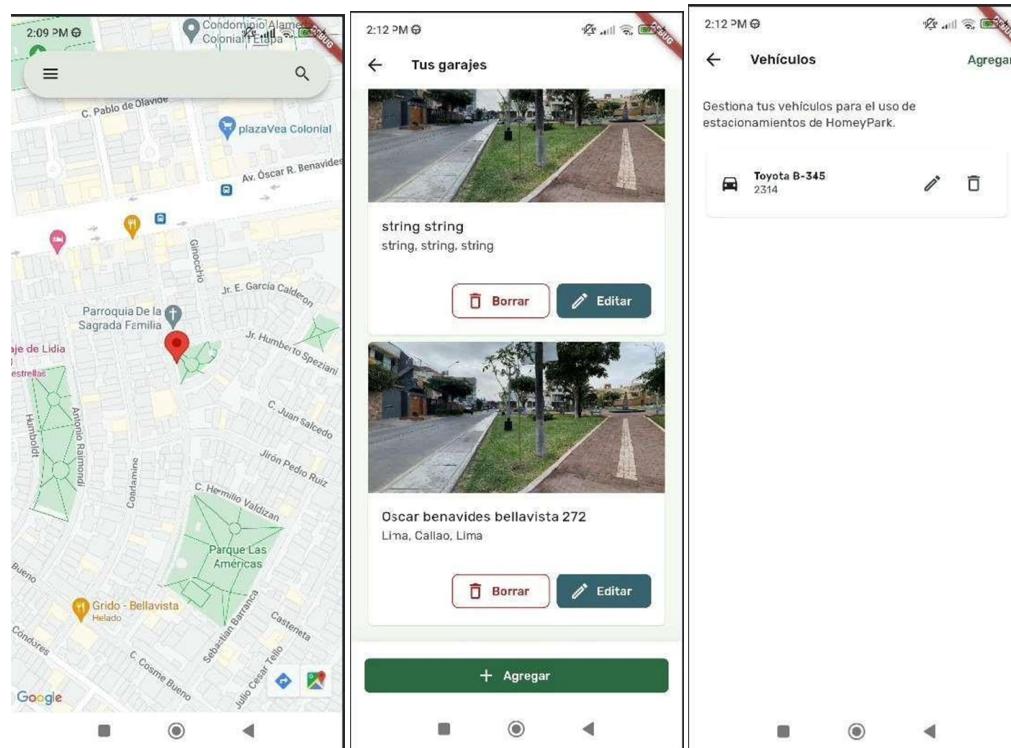
**4. Seguridad y Protección de Datos**

- Toda la información personal será tratada conforme a las normas vigentes en materia de protección de datos personales en el Perú.

Link: <https://homey-park-experiments.web.app/terms-conditions>

## 5.2.5. Implemented Native-Mobile Application Evidence

A continuación, se presentan las pantallas de nuestra aplicación móvil:



## 5.2.6. Implemented RESTful API and/or Serverless Backend Evidence

### Crear un usuario

HoneyPark / Crear un usuario

POST http://localhost:8080/users

Params Authorization Headers (8) Body Scripts Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```

1 {
2   "firstName": "Nombre",
3   "lastName": "Apellido",
4   "email": "correo@example.com",
5   "password": "contraseña"
6 }
```

Body Cookies Headers (8) Test Results

{ } JSON D Preview Visualize

```

1 {
2   "id": 6,
3   "name": null,
4   "lastName": "Apellido",
5   "email": "correo@example.com",
6   "password": "contraseña",
7   "dateCreated": "2025-04-23T13:51:07.7525886",
8   "vehicles": [],
9   "cards": []
10 }
```

### Obtener todos los usuarios

The screenshot shows two separate Postman requests:

- GET Ver usuarios**: URL: http://localhost:8080/users. The request body is a JSON array of user objects. One object is highlighted:
 

```

1 [
2   {
3     "id": 1,
4     "name": "Marcelo Fabian",
5     "lastName": "Gazzo Vega",
6     "email": "marcelogarroi37@gmail.com",
7     "password": "admin123",
8     "dateCreated": "2025-04-18T15:31:52.601419",
9     "vehicles": [],
10    "cards": []
11  },
12  [
13    {
14      "id": 2,
15      "name": "Daniel",
16      "lastName": "Chirinos",
17      "email": "daniel@gmail.com",
18      "password": "admin123",
19      "dateCreated": "2025-04-18T15:31:52.64658",
20      "vehicles": [],
21      "cards": []
22    },
23    [
24      {
25        "id": 3,
26        "name": "George",
27        "lastName": "Aliaga",
28        "email": "george@gmail.com",
29        "password": "admin123",
30        "dateCreated": "2025-04-18T15:31:52.650579",
31        "vehicles": [],
32        "cards": []
33      },
34      {
35        "id": 4,
36        "name": "Vitorio",
37        "lastName": "Eduardo",
38        "email": "marcelogarroi37@gmail.com",
39        "password": "admin123",
40        "dateCreated": "2025-04-18T15:31:52.653578",
41      }
42  ]
43 ]
      
```
- PUT Editar usuario**: URL: http://localhost:8080/users/2. The request body is a JSON object with updated user information:
 

```

1 {
2   "firstName": "Carla",
3   "lastName": "Cachis",
4   "email": "nuevoCorre@example.com"
5 }
      
```

## Crear un parking

The screenshot shows a single Postman request:

- POST Crear Parking**: URL: http://localhost:8080/vehicles/create. The request body is a JSON object representing a parking lot:
 

```

1 {
2   "userId": 1,
3   "pricePerHour": 20.0,
4   "pricePerDay": 100.0,
5   "description": "Descripción del parking",
6   "address": "Dirección",
7   "city": "Ciudad",
8   "state": "Estado",
9   "latitude": -12.107520,
10  "longitude": -77.024848
11 }
      
```

## Obtener todos los parkings

The screenshot shows two panels of the Postman application. The left panel displays a GET request to 'http://localhost:8080/vehicles' with a JSON body containing a single parking entry. The right panel shows a PUT request to 'http://localhost:8080/parking/1' with a JSON body containing updated values for price and description.

**Left Panel (GET Request):**

```

1 {
2   "userId": 1,
3   "pricePerHour": 28.0,
4   "pricePerDay": 180.0,
5   "description": "Descripción del parking",
6   "address": "Dirección",
7   "city": "Ciudad",
8   "state": "Estado",
9   "latitude": -12.107520,
10  "longitude": -77.024040
11 }

```

**Right Panel (PUT Request):**

```

1 {
2   "id": 1,
3   "width": null,
4   "length": null,
5   "height": null,
6   "price": null,
7   "phone": null,
8   "space": null,
9   "description": "Nueva descripción",
10  "location": {
11    "id": 1,
12    "address": "Av. General Recavarren",
13    "numDirection": "1300",
14    "street": "Surquillo",
15    "district": "Lima",
16    "city": "Lima",
17    "latitude": -12.10752,
18    "longitude": -77.02404
19  },
20  "schedules": []
21 }

```

### Eliminar un parking

The screenshot shows a DELETE request to 'http://localhost:8080/parking/delete/1' with a JSON body containing the same parking entry as the previous update request.

**Body:**

```

1 {
2   "pricePerHour": 25.0,
3   "pricePerDay": 120.0,
4   "description": "Nueva descripción"
5 }

```

**Response:**

```

1 Parking with given id successfully deleted

```

### Obtener todas las reservaciones

```

GET http://localhost:8080/reservations
Params Authorization Headers (8) Body Scripts Settings
Body: raw
1: {
2:   "duration": 2,
3:   "price": 35.0,
4:   "startTime": "2023-10-01T10:00:00",
5:   "endTime": "2023-10-01T12:00:00",
6:   "userId": 1,
7:   "parkingId": 2
8: }

GET http://localhost:8080/reservations/2
Params Authorization Headers (8) Body Scripts Settings
Body: raw
1: {
2:   "duration": 2,
3:   "price": 35.0,
4:   "startTime": "2023-10-01T10:00:00",
5:   "endTime": "2023-10-01T12:00:00",
6:   "userId": 1,
7:   "parkingId": 2
8: }

```

### 5.2.7 RESTful API documentation

En esta sección se presentan los endpoints desarrollados en el presente sprint y se adjuntan capturas de las acciones CRUD realizadas con OpenAPI. Dentro del alcance del sprint, se han desarrollado los bounded contexts de User Management, Parking Management, Reservation Management, Schedule Management, Location Management, y Vehicle Management.

Controller	Endpoint	HTTP Method	Description
Schedule	/schedule/{id}	PUT	
Schedule	/schedule	GET	
Schedule	/schedule	POST	
Parking	/parking/{id}	PUT	
Parking	/parking	GET	
Parking	/parking	POST	
Parking	/parking/{id}/details	GET	
Parking	/parking/delete/{id}	DELETE	
Location	/location/{id}	PUT	
Location	/location	GET	
vehicle-controller	/vehicles/{id}	PUT	
vehicle-controller	/vehicles/{id}	GET	
vehicle-controller	/vehicles	GET	
vehicle-controller	/vehicles	POST	
vehicle-controller	/vehicles/delete/{id}	DELETE	
user-controller	/users/{id}	PUT	
user-controller	/users/{id}	GET	
user-controller	/users	GET	
user-controller	/users	POST	
user-controller	/users/delete/{id}	DELETE	
reservation-controller	/reservations/{id}	PUT	
reservation-controller	/reservations/{id}	PUT	
reservation-controller	/reservations/{id}/status	PUT	
reservation-controller	/reservations	GET	
reservation-controller	/reservations	POST	
reservation-controller	/reservations/{id}/details	GET	
reservation-controller	/reservations/upcoming	GET	
reservation-controller	/reservations/past	GET	
reservation-controller	/reservations/inProgress	GET	
card-controller	/cards	GET	
card-controller	/cards	POST	
card-controller	/cards/delete/{id}	DELETE	

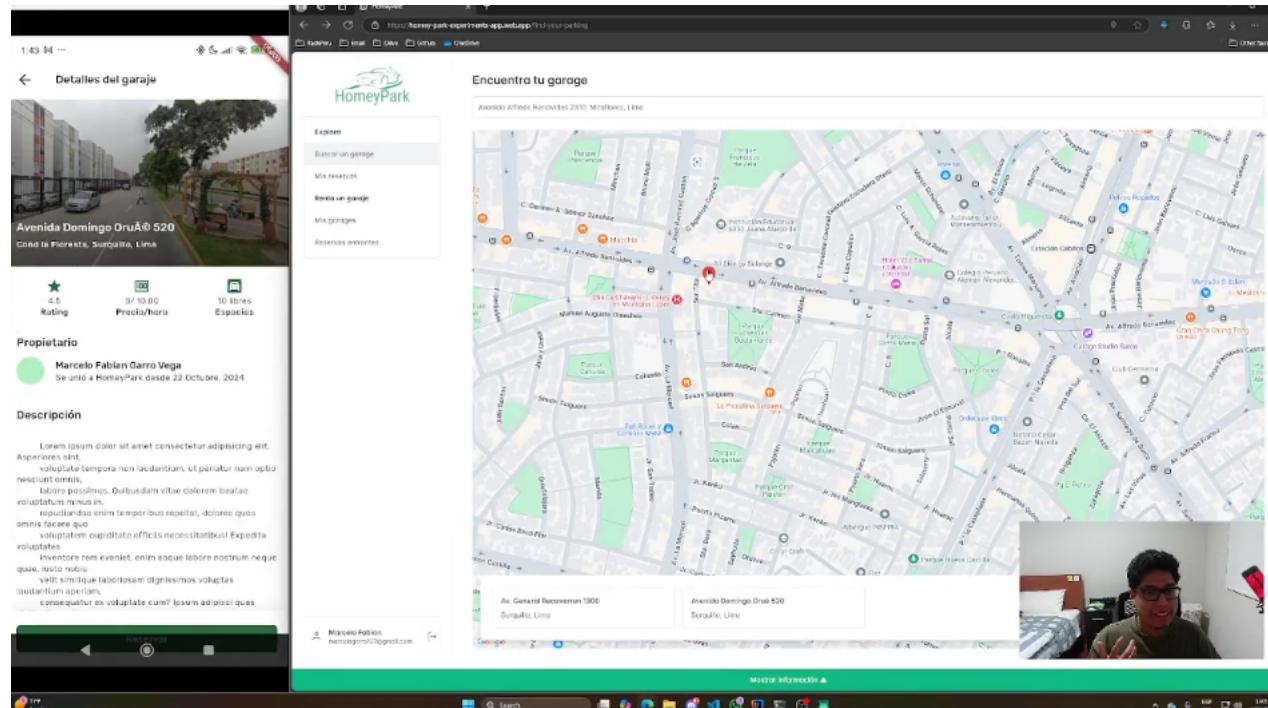
## 5.2.8. Team Collaboration Insights

Nombre	Actividad
Sebastian Cachis Gonzales	Implementacion y desarrollo de Backend
Adriano Sebastian Cruz Palomino	Implementacion y desarrollo de la App Web y Landing Page
Amner Levi Llamo Sanchez	Implementacion y desarrollo de Backend
Marcelo Fabian Garro Vega	Implementacion y desarrollo de la App Web y Mobile App
Lucio Heli Yen Cerna	Implementacion y desarrollo de la App Web y Mobile App

Se realizaron los commits por una persona, pero eso desestima el aporte evidenciado entre los miembros de grupo.

## 5.3. Video About-the-Product

### Video del Producto



## Capítulo VI: Product Verification & Validation

### 6.1. Testing Suites & Validation

#### 6.1.1. Core Entities Unit Tests

Los Core Entities Unit Tests son cruciales en el desarrollo de software para asegurar la calidad y el funcionamiento adecuado de las entidades centrales, lo que ayuda a prevenir errores y simplifica el mantenimiento del código.

User Service Test

The screenshot shows a Java development environment with the following details:

- Project Structure:** The project contains several modules like transformers, reservations, shared, vehicles, and HomeparkApplication. The test module contains java and resources sub-modules. Under java, there's a com.homepark.web\_service package with core and entities unit tests. The entities unit tests folder contains RoleTest, UserTest, and VehicleTest.
- Code Editor:** The UserTest.java file is open. It imports org.junit.jupiter.api.BeforeEach, org.junit.jupiter.api.Test, java.util.List, static org.junit.jupiter.api.Assertions.\*, and static org.mockito.Mockito.mock. The class is annotated with @Test and @BeforeEach. It has methods for setting password, email, and constructor with fields and command.
- Run Tab:** The UserTest tab is selected. The output shows the test ran for 821ms and passed 5 tests out of 5 total. It also displays some Mockito and Java agent warnings.

```

import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;
import java.util.List;
import static org.junit.jupiter.api.Assertions.*;
import static org.mockito.Mockito.mock;

/**
 * This class represents the unit tests for the User aggregate.
 */
public class UserTest {
    private Role role; 3 usages
    private SignUpCommand command; 17 usages

    @BeforeEach
    void setup() {
        ...
    }
}

UserTest (com.homepark.web_service.core.entities.unit.tests)
  ✓ testSetPassword()
  ✓ testGetEmail()
  ✓ testConstructorWithFields()
  ✓ testConstructorWithCommand()
  ✓ testSetUsername()

Process finished with exit code 0
  
```

Role Service Test

**Project**

- > transformers
- > reservations
- > shared
- > vehicles
- HomeparkApplication
- > resources
- test
  - java
    - com.homepark.web\_service
      - core
        - entities.unit.tests
          - RoleTest
          - UserTest
          - VehicleTest**
      - integration.tests
        - ProfileControllerIntegrationTest
        - VehicleControllerIntegrationTest

**RoleTest.java**

```

package com.homepark.web_service.core.entities.unit.tests;

import com.homepark.web_service.iam.domain.model.entities.Role;
import com.homepark.web_service.iam.domain.model.valueobjects.Roles;
import org.junit.jupiter.api.Test;

import java.util.List;

import static org.junit.jupiter.api.Assertions.*;

class RoleTest { Dark7YT

    @Test Dark7YT
    void testNoArgsConstructor() {
        Role role = new Role();
        assertNull(role.getId());
        assertEquals("", role.getName());
    }
}

```

**Run**

RoleTest (com.homepark.web_service.core.entities.unit.tests)	21ms	8 tests passed 8 tests total, 21 ms
testValidateRoleSetReturnsDefaultIfEmpty()	19ms	*C:\Program Files\Java\jdk-23\bin\java.exe* ...
testNoArgsConstructor()	1ms	
testToRoleFromName()	1ms	
testValidateRoleSetReturnsSameIfNotEmpty()		
testConstructorWithEnum()		
testGetStringName()		
testAllArgsConstructor()		
testGetDefaultRole()		

Process finished with exit code 0

## Vehicle Service Test

**Project**

- > transformers
- > reservations
- > shared
- > vehicles
- HomeparkApplication
- > resources
- test
  - java
    - com.homepark.web\_service
      - core
        - entities.unit.tests
          - RoleTest
          - UserTest
          - VehicleTest**
      - integration.tests
        - ProfileControllerIntegrationTest
        - VehicleControllerIntegrationTest

**VehicleTest.java**

```

package com.homepark.web_service.core.entities.unit.tests;

import com.homepark.web_service.vehicles.domain.model.aggregates.Vehicle;
import com.homepark.web_service.vehicles.domain.model.commands.CreateVehicleCommand;
import com.homepark.web_service.vehicles.domain.model.commands.UpdateVehicleCommand;
import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.*;

class VehicleTest { Dark7YT

    @Test Dark7YT
    void testConstructorWithFields() {
        Vehicle vehicle = new Vehicle(licensePlate: "ABC123", model: "Model X");
        assertNull(vehicle.getId());
        assertEquals("ABC123", vehicle.getLicensePlate());
    }
}

```

**Run**

VehicleTest (com.homepark.web_service.core.entities.unit.tests)	26 ms	3 tests passed 3 tests total, 26 ms
testUpdatedVehicle()	23 ms	*C:\Program Files\Java\jdk-23\bin\java.exe* ...
testConstructorWithCreateVehicleCommand()	3 ms	
testConstructorWithFields()		

Process finished with exit code 0

## Location Service Test

The screenshot shows an IDE interface with two main panes. The left pane displays the project structure under 'com.homeypark.web\_service'. It includes packages for profiles, reservations, shared, vehicles, and HomeyparkApplication. Below these are 'resources' and 'test' directories. The 'test' directory contains a 'java' package with sub-packages 'core' and 'entities.unit.tests'. Inside 'entities.unit.tests', several test classes are listed: LocationTest (selected), ParkingTest, ProfileTest, ReservationTest, RoleTest, ScheduleTest, UserTest, and VehicleTest. There are also 'integration.tests' and 'HomeyparkApplicationTests' files. The right pane shows the code for 'LocationTest'. The code defines a class 'LocationTest' with a private field 'mockLocation'. It includes an annotation '@BeforeEach' and a method 'setUp()' that initializes 'mockLocation' as a mock object. Below this is a test method '@Test' named 'testUpdateLocation()' which creates an 'UpdateLocationCommand' object with various parameters (locationId, address, numDirection, street, district, city, latitude, longitude) and then calls the 'updateLocation' method on 'mockLocation'. The test concludes with an assertion. At the bottom of the right pane, the output of the test run is shown, indicating 2 tests passed in 991ms. The command used to run the test is 'C:\Users\user\.jdks\openjdk-23.0.1\bin\java.exe ...'. The output also includes several warning messages related to Mockito's self-attaching mechanism and Java agent loading.

```

9
10  class LocationTest { ▲ AdrianoSCruzP
11
12      private Location mockLocation; 17 usages
13
14      @BeforeEach ▲ AdrianoSCruzP
15      void setUp() {
16          mockLocation = mock(Location.class);
17      }
18
19      @Test ▲ AdrianoSCruzP
20      void testUpdateLocation() {
21          // Arrange
22          UpdateLocationCommand updateCommand = new UpdateLocationCommand(
23              locationId: 1L,
24              address: "Test Address",
25              numDirection: "123",
26              street: "Test Street",
27              district: "Test District",
28              city: "Test City",
29              latitude: 12.34,
30              longitude: 56.78
31          );
32
33          // Act
34          mockLocation.updateLocation(updateCommand);
35
36          // Assert

```

Run LocationTest x

LocationTest (com.homeypark.web\_service) 991 ms

- ✓ testSettersAndGetters() 989 ms
- ✓ testUpdateLocation() 2 ms

Tests passed: 2 of 2 tests – 991 ms

C:\Users\user\.jdks\openjdk-23.0.1\bin\java.exe ...

Mockito is currently self-attaching to enable the inline-mock-maker. This will no longer work in future releases of the JUnit framework. Please use the --enable-junit-platform argument to enable the platform-based runner instead.

WARNING: A Java agent has been loaded dynamically (C:\Users\user\.m2\repository\net\bytebuddy\byte-buddy-agent\1.15.11\byte-buddy-agent-1.15.11.jar). This may cause issues with the Java Virtual Machine.

WARNING: If a serviceability tool is in use, please run with -XX:+EnableDynamicAgentLoading to hide this warning.

WARNING: If a serviceability tool is not in use, please run with -Djdk.instrument.traceUsage for more information.

WARNING: Dynamic loading of agents will be disallowed by default in a future release.

OpenJDK 64-Bit Server VM warning: Sharing is only supported for boot loader classes because bootstrap classpath has been explicitly specified.

Process finished with exit code 0

## Parking Service Test

The screenshot shows an IDE interface with two main panes. The left pane is the 'Project' view, displaying the file structure of the 'HomeparkApplication' project. The right pane is the 'ParkingTest.java' code editor and the 'Run' tool window.

**Project View:**

- src
  - main
    - com.homepark.infrastructure.persistence.jpa.repositories
    - com.homepark.interfaces
    - com.homepark.profiles
    - com.homepark.reservations
    - com.homepark.shared
    - com.homepark.vehicles
    - com.homepark.HomeparkApplication
  - test
    - java
      - com.homepark.web\_service
        - core
          - entities.unit.tests
            - ParkingTest
  - target

**ParkingTest.java Code:**

```

12 class ParkingTest { @AdrianoSCruzP
31     void testConstructorWithCommand() {
48         assertEquals(expected: "Test Address", parking.getLocation().getAddress());
49     }
50
51     @Test @AdrianoSCruzP
52     void testSettersAndGetters() {
53         // Arrange
54         Parking parking = new Parking();
55
56         // Act
57         parking.setWidth(2.5);
58         parking.setLength(5.0);
59         parking.setHeight(3.0);
60         parking.setPrice(10.0);
61         parking.setPhone("123456789");
62         parking.setSpace(1);
63         parking.setDescription("Test Description");
64
65         // Assert
66         assertEquals(expected: 2.5, parking.getWidth());
67         assertEquals(expected: 5.0, parking.getLength());
68         assertEquals(expected: 3.0, parking.getHeight());
69         assertEquals(expected: 10.0, parking.getPrice());
70         assertEquals(expected: "123456789", parking.getPhone());
71         assertEquals(expected: 1, parking.getSpace());
72         assertEquals(expected: "Test Description", parking.getDescription());
73     }

```

**Run Session:**

Test Method	Time
testSetAndGetLocation()	892ms
testSettersAndGetters()	1ms
testConstructorWithCommand()	2ms

Tests passed: 3 of 3 tests – 895 ms

C:\Users\user\.jdks\openjdk-23.0.1\bin\java.exe ...

Mockito is currently self-attaching to enable the inline-mock-maker. This will no longer work in future releases of the

WARNING: A Java agent has been loaded dynamically (C:\Users\user\.m2\repository\net\bytebuddy\byte-buddy-agent\1.15.11\b

WARNING: If a serviceability tool is in use, please run with -XX:+EnableDynamicAgentLoading to hide this warning

WARNING: If a serviceability tool is not in use, please run with -Djdk.instrument.traceUsage for more information

WARNING: Dynamic loading of agents will be disallowed by default in a future release

OpenJDK 64-Bit Server VM warning: Sharing is only supported for boot loader classes because bootstrap classpath has been

Process finished with exit code 0

Schedule Service Test

The screenshot shows an IDE interface with the following details:

- Project View:** Shows the project structure with packages like `infrastructure.persistence.jpa.repositories`, `interfaces`, `reservations`, `shared`, `vehicles`, `HomeparkApplication`, `resources`, and `test`. The `test/java/com.homepark.web_service/core/entities/unit.tests` package is expanded, and `ScheduleTest` is selected.
- ScheduleTest.java Content:**

```

1 package com.homepark.web_service.core.entities.unit.tests;
2
3 > import ...
4
5 class ScheduleTest { ✎ AdrianoSCruzP
6
7     private CreateScheduleCommand createCommand; 3 usages
8     private UpdateScheduleCommand updateCommand; 4 usages
9     private Schedule mockSchedule; 9 usages
10
11    @BeforeEach ✎ AdrianoSCruzP
12    void setUp() {
13        // Arrange: Initialize commands and mock
14        createCommand = new CreateScheduleCommand(parkingId: 1L, day: "MONDAY",
15            LocalTime.of(hour: 8, minute: 0),
16            LocalTime.of(hour: 18, minute: 0));
17
18        updateCommand = new UpdateScheduleCommand(scheduledId: 1L, day: "TUESDAY",
19            LocalTime.of(hour: 9, minute: 0),
20            LocalTime.of(hour: 17, minute: 0));
21
22        mockSchedule = mock(Schedule.class);
23    }
24
25    @Test ✎ AdrianoSCruzP
26    void testConstructorWithCommand() {
27        // Arrange
28        CreateScheduleCommand command = createCommand;
29
30        // Act
31        Schedule result = command.create();
32
33        // Assert
34        assertEquals("MONDAY", result.getDay());
35        assertEquals(8, result.getStartTime().getHour());
36        assertEquals(18, result.getEndTime().getHour());
37    }
38

```
- Run Tab:** Shows the test `ScheduleTest` running. The output indicates 3 tests passed in 930ms.
- Terminal Output:**

```

C:\Users\user\.jdks\openjdk-23.0.1\bin\java.exe ...
Mockito is currently self-attaching to enable the inline-mock-maker. This will no longer work in future releases of Mockito.
WARNING: A Java agent has been loaded dynamically (C:\Users\user\.m2\repository\net\bytebuddy\byte-buddy-agent\1.13.1\byte-buddy-agent-1.13.1.jar).
WARNING: If a serviceability tool is in use, please run with -XX:+EnableDynamicAgentLoading to hide this warning.
WARNING: If a serviceability tool is not in use, please run with -Djdk.instrument.traceUsage for more information.
WARNING: Dynamic loading of agents will be disallowed by default in a future release.
OpenJDK 64-Bit Server VM warning: Sharing is only supported for boot loader classes because bootstrap classpath has been modified.

```

Reservation Service Test

The screenshot shows an IDE interface with the following details:

- Project View:** Shows the project structure under 'test/java/com.homeypark.web\_service/core/entities.unit.tests'. The file 'ReservationTest.java' is selected.
- Code Editor:** Displays the content of 'ReservationTest.java'. The code includes imports, class definitions, and a setup method. It uses Mockito annotations like @Mock and @BeforeEach.
- Run View:** Shows the test results for 'ReservationTest'. It lists four tests: 'testConstructorWithCreateCommand', 'testUpdatedStatus()', 'testUpdatedReservation()', and 'testDefaultConstructor()'. All tests passed in 1 second and 41 milliseconds.
- Output:** Shows log messages from Mockito and Java agents. It includes warnings about self-attaching and dynamic loading, and a note about OpenJDK sharing support.

```

package com.homeypark.web_service.core.entities.unit.tests;
import ...;

public class ReservationTest { ... }

private CreateReservationCommand createCommand; 13 usages
private UpdateReservationCommand updateCommand; 7 usages
private UpdateStatusCommand updateStatusCommand; 3 usages

private final Long hostId = 1L; 2 usages
private final Long guestId = 2L; 2 usages
private final Long parkingId = 3L; 2 usages
private final Long vehicleId = 4L; 2 usages
private final LocalDate reservationDate = LocalDate.now(); 4 usages
private final LocalTime startTime = LocalTime.of( hour: 10, minute: 0); 4 usages
private final LocalTime endTime = LocalTime.of( hour: 12, minute: 0); 4 usages
private final Integer hoursRegistered = 2; 4 usages
private final Double totalFare = 20.0; 4 usages

@BeforeEach
void setUp() {
    // Mock CreateReservationCommand
    createCommand = mock(CreateReservationCommand.class);
    when(createCommand.hoursRegistered()).thenReturn(hoursRegistered);
    when(createCommand.totalFare()).thenReturn(totalFare);
    when(createCommand.reservationDate()).thenReturn(reservationDate);
    when(createCommand.startTime()).thenReturn(startTime);
}

Tests passed: 4 of 4 tests – 1sec 41ms
C:\Users\user\.jdks\openjdk-23.0.1\bin\java.exe ...
Mockito is currently self-attaching to enable the inline-mock-maker. This will no longer work in future release
WARNING: A Java agent has been loaded dynamically (C:\Users\user\.m2\repository\net\bytebuddy\byte-buddy-agent\
WARNING: If a serviceability tool is in use, please run with -XX:+EnableDynamicAgentLoading to hide this warning
WARNING: If a serviceability tool is not in use, please run with -Djdk.instrument.traceUsage for more information
WARNING: Dynamic loading of agents will be disallowed by default in a future release
OpenJDK 64-Bit Server VM warning: Sharing is only supported for boot loader classes because bootstrap classpath
Process finished with exit code 0

```

Profile Service Test

The screenshot shows an IDE interface with the following details:

- Project View:** Shows the project structure under "Project". It includes:
  - infrastructure.persistence.jpa.repositories
  - interfaces
  - profiles
  - reservations
  - shared
  - vehicles
  - HomeyparkApplication
  - resources
  - test (selected)
  - java (selected)
  - com.homeypark.web\_service (selected)
  - core (selected)
  - entities.unit.tests (selected)
  - ProfileTest (selected)
  - LocationTest
  - ParkingTest
  - ReservationTest
  - RoleTest
  - ScheduleTest
  - UserTest
  - VehicleTest
  - integration.tests
  - HomeyparkApplicationTests
  - resources
- Code Editor:** The active file is `ProfileTest.java`. The code contains two test methods: `testConstructorWithCreateCommand()` and `testUpdateProfile()`.
- Run Tab:** Shows the results of the last run. It lists three tests under `ProfileTest`:
  - testConstructorWithCreateCommand() (914 ms)
  - testSettersAndGetters() (43 ms)
  - testUpdateProfile() (2 ms)
 The total time is 959 ms. All tests passed.
- Output Tab:** Displays log messages from the Java runtime environment. It includes:
  - C:\Users\user\.jdks\openjdk-23.0.1\bin\java.exe ...
  - Mockito is currently self-attaching to enable the inline-mock-maker. This will no longer work in future
  - WARNING: A Java agent has been loaded dynamically (C:\Users\user\.m2\repository\net\bytebuddy\byte-budd
  - WARNING: If a serviceability tool is in use, please run with -XX:+EnableDynamicAgentLoading to hide thi
  - WARNING: If a serviceability tool is not in use, please run with -Djdk.instrument.traceUsage for more i
  - WARNING: Dynamic loading of agents will be disallowed by default in a future release
  - OpenJDK 64-Bit Server VM warning: Sharing is only supported for boot loader classes because bootstrap c
 The process finished with exit code 0.

### 6.1.2. Core Integration Tests

Las Core Integration Tests resultan esenciales para verificar la correcta interacción de los controladores con otros componentes del sistema, tales como servicios y bases de datos. Al poner a prueba escenarios de error, estas pruebas aseguran que el sistema gestione apropiadamente situaciones inesperadas y responda con los códigos de estado precisos. Esto se traduce en una mejor experiencia para el usuario, simplifica la depuración y contribuye a la creación de software robusto y de alta calidad.

#### Profile Controller Test

The screenshot shows a Java IDE interface with the following details:

- Project View:** Shows the project structure under "Project". It includes "shared", "vehicles", "HomeyparkApplication", "resources", "test" (containing "java" which has "com.homeypark.web\_service" and "core" with "entities.unit.tests" containing "RoleTest", "UserTest", and "VehicleTest"), "integration.tests" containing "ProfileControllerIntegrationTest" and "VehicleControllerIntegrationTest", and "HomeyparkApplicationTests".
- Code Editor:** The active file is `ProfileControllerIntegrationTest.java`. The code is as follows:

```

1 package com.homeypark.web_service.core.integration.tests;
2
3 import com.homeypark.web_service.profiles.domain.model.Profile;
4 import com.homeypark.web_service.profiles.domain.model.User;
5 import com.homeypark.web_service.profiles.domain.model.Vehicle;
6 import com.homeypark.web_service.profiles.domain.service.ProfileService;
7 import com.homeypark.web_service.profiles.domain.service.UserService;
8 import com.homeypark.web_service.profiles.interfaces.ProfileRepository;
9 import com.homeypark.web_service.profiles.interfaces.UserRepository;
10 import com.homeypark.web_service.profiles.interfaces.VehicleRepository;
11 import com.homeypark.web_service.profiles.interfaces.UserService;
12 import org.junit.jupiter.api.BeforeEach;
13 import org.junit.jupiter.api.Test;
14 import org.mockito.ArgumentMatchers;
15 import org.mockito.Mockito;
16 import org.springframework.http.HttpStatus;

```

- Run Tab:** The "ProfileControllerIntegrationTest" is selected for execution. The results show:
  - 2 tests passed
  - 2 tests total, 801 ms
  - testGetProfileByIdSuccess() took 796 ms
  - testCreateProfileSuccess() took 5 ms
  - Output log lines include: "C:\Program Files\Java\jdk-23\bin\java.exe" ...", "Mockito is currently self-attaching to enable the inline-m...", "WARNING: A Java agent has been loaded dynamically (C:\User...", "WARNING: If a serviceability tool is in use, please run wi...", "WARNING: If a serviceability tool is not in use, please run...", "WARNING: Dynamic loading of agents will be disallowed by d...", "Java HotSpot(TM) 64-Bit Server VM warning: Sharing is only..."
  - Final message: "Process finished with exit code 0"

Vehicle Controller Test

The screenshot shows an IDE interface with the following details:

- Project View:** Shows the project structure with packages like `shared`, `vehicles`, `HomeparkApplication`, `resources`, and `test` (containing `java` and `resources.Features`). A file named `VehicleControllerIntegrationTest.java` is selected.
- Code Editor:** Displays the source code for `VehicleControllerIntegrationTest.java`. The code imports Mockito, ArgumentMatchers, HttpStatus, ResponseEntity, Optional, and Assertions from JUnit Jupiter. It defines a class `VehicleControllerIntegrationTest` with a constructor `setUp()` and an annotation `@BeforeEach`.
- Run View:** Shows the execution results for the selected test class:
  - Tests:** `VehicleControllerIntegrationTest` (com.homepark.web\_service.core.inte...), `testGetVehicleByIdSuccess()`, and `testCreateVehicleSuccess()`.
  - Time:** Total execution time was 780 ms.
  - Status:** All tests passed (2 tests total).
  - Output:** Includes log messages from Mockito and Java HotSpot VM, indicating self-attachment and dynamic loading of agents.
  - Exit Code:** Process finished with exit code 0.

Location Controller Test

The screenshot shows a Java project structure on the left and a code editor on the right.

**Project View:**

- vehicles
- HomeyparkApplication
- resources
- test
  - java
    - com.homeypark.web\_service
      - core
      - entities.unit.tests
      - integration.tests
        - LocationControllerIntegrationTest (selected)
        - ParkingControllerIntegrationTest
        - ProfileControllerIntegrationTest
        - ReservationControllerIntegrationTest
        - ScheduleControllerIntegrationTest
        - VehicleControllerIntegrationTest
      - HomeyparkApplicationTests
  - resources
  - target
    - .dockerignore
    - .env.development
    - .env.production
    - .gitattributes
    - .gitignore
    - docker-compose.yml
    - Dockerfile

**Code Editor (LocationControllerIntegrationTest.java):**

```
1 package com.homeypark.web_service.core.integration.tests;
2
3 import ...
4
5 public class LocationControllerIntegrationTest { ✘ AdrianoSCruzP
6
7     private LocationCommandService locationCommandService; 3 usages
8     private LocationQueryService locationQueryService; 3 usages
9     private LocationController locationController; 3 usages
10
11     @BeforeEach ✘ AdrianoSCruzP
12     void setUp() {
13         locationCommandService = Mockito.mock(LocationCommandService.class);
14         locationQueryService = Mockito.mock(LocationQueryService.class);
15         locationController = new LocationController(locationCommandService, locationQueryService);
16     }
17
18     @Test ✘ AdrianoSCruzP
19     void testUpdateLocationSuccess() {
20         // Arrange
21         Long locationId = 1L;
22         UpdateLocationResource resource = new UpdateLocationResource(
23             address: "Av. Siempre Viva", numDirection: "742", street: "Springfield St",
24             district: "Springfield", city: "Illinois", latitude: -12.0464, longitude: -77.0464
25         );
26
27         Location updatedLocation = new Location(
28             locationId,
29         );
30     }
31
32     @Test ✘ AdrianoSCruzP
33     void testGetAllLocationsSuccess() {
34         // Arrange
35         List<Location> locations = new ArrayList<>();
36
37         // Act
38         List<Location> result = locationController.getAllLocations();
39
40         // Assert
41         assertEquals(locations, result);
42     }
43
44     @Test ✘ AdrianoSCruzP
45     void testGetLocationSuccess() {
46         // Arrange
47         Location location = new Location();
48
49         // Act
50         Location result = locationController.getLocation(1L);
51
52         // Assert
53         assertEquals(location, result);
54     }
55
56     @Test ✘ AdrianoSCruzP
57     void testDeleteLocationSuccess() {
58         // Arrange
59         Location location = new Location();
60
61         // Act
62         boolean result = locationController.deleteLocation(1L);
63
64         // Assert
65         assertTrue(result);
66     }
67
68     @Test ✘ AdrianoSCruzP
69     void testUpdateLocationFailure() {
70         // Arrange
71         Long locationId = 1L;
72         UpdateLocationResource resource = new UpdateLocationResource(
73             address: "Av. Siempre Viva", numDirection: "742", street: "Springfield St",
74             district: "Springfield", city: "Illinois", latitude: -12.0464, longitude: -77.0464
75         );
76
77         Location updatedLocation = new Location(
78             locationId,
79         );
80
81         // Act
82         Exception exception = assertThrows(IllegalArgumentException.class, () ->
83             locationController.updateLocation(resource));
84
85         // Assert
86         assertEquals("Location ID must be provided", exception.getMessage());
87     }
88
89     @Test ✘ AdrianoSCruzP
90     void testGetAllLocationsFailure() {
91         // Arrange
92         List<Location> locations = new ArrayList<>();
93
94         // Act
95         List<Location> result = locationController.getAllLocations();
96
97         // Assert
98         assertEquals(locations, result);
99     }
100
101    @Test ✘ AdrianoSCruzP
102    void testGetLocationFailure() {
103        // Arrange
104        Location location = new Location();
105
106        // Act
107        Location result = locationController.getLocation(1L);
108
109        // Assert
110        assertEquals(location, result);
111    }
112
113    @Test ✘ AdrianoSCruzP
114    void testDeleteLocationFailure() {
115        // Arrange
116        Location location = new Location();
117
118        // Act
119        boolean result = locationController.deleteLocation(1L);
120
121        // Assert
122        assertFalse(result);
123    }
124
125    @Test ✘ AdrianoSCruzP
126    void testUpdateLocationFailureWithNullAddress() {
127        // Arrange
128        Long locationId = 1L;
129        UpdateLocationResource resource = new UpdateLocationResource(
130            address: null, numDirection: "742", street: "Springfield St",
131            district: "Springfield", city: "Illinois", latitude: -12.0464, longitude: -77.0464
132        );
133
134        Location updatedLocation = new Location(
135            locationId,
136        );
137
138        // Act
139        Exception exception = assertThrows(IllegalArgumentException.class, () ->
140            locationController.updateLocation(resource));
141
142        // Assert
143        assertEquals("Address cannot be null", exception.getMessage());
144    }
145
146    @Test ✘ AdrianoSCruzP
147    void testGetAllLocationsFailureWithEmptyList() {
148        // Arrange
149        List<Location> locations = new ArrayList<>();
150
151        // Act
152        List<Location> result = locationController.getAllLocations();
153
154        // Assert
155        assertEquals(locations, result);
156    }
157
158    @Test ✘ AdrianoSCruzP
159    void testGetLocationFailureWithEmptyList() {
160        // Arrange
161        Location location = new Location();
162
163        // Act
164        Location result = locationController.getLocation(1L);
165
166        // Assert
167        assertEquals(location, result);
168    }
169
170    @Test ✘ AdrianoSCruzP
171    void testDeleteLocationFailureWithEmptyList() {
172        // Arrange
173        Location location = new Location();
174
175        // Act
176        boolean result = locationController.deleteLocation(1L);
177
178        // Assert
179        assertFalse(result);
180    }
181
182    @Test ✘ AdrianoSCruzP
183    void testUpdateLocationFailureWithEmptyList() {
184        // Arrange
185        Long locationId = 1L;
186        UpdateLocationResource resource = new UpdateLocationResource(
187            address: "Av. Siempre Viva", numDirection: "742", street: "Springfield St",
188            district: "Springfield", city: "Illinois", latitude: -12.0464, longitude: -77.0464
189        );
190
191        Location updatedLocation = new Location(
192            locationId,
193        );
194
195        // Act
196        Exception exception = assertThrows(IllegalArgumentException.class, () ->
197            locationController.updateLocation(resource));
198
199        // Assert
200        assertEquals("Location ID must be provided", exception.getMessage());
201    }
202
203    @Test ✘ AdrianoSCruzP
204    void testGetAllLocationsFailureWithEmptyListAndNullAddress() {
205        // Arrange
206        List<Location> locations = new ArrayList<>();
207
208        // Act
209        List<Location> result = locationController.getAllLocations();
210
211        // Assert
212        assertEquals(locations, result);
213    }
214
215    @Test ✘ AdrianoSCruzP
216    void testGetLocationFailureWithEmptyListAndNullAddress() {
217        // Arrange
218        Location location = new Location();
219
220        // Act
221        Location result = locationController.getLocation(1L);
222
223        // Assert
224        assertEquals(location, result);
225    }
226
227    @Test ✘ AdrianoSCruzP
228    void testDeleteLocationFailureWithEmptyListAndNullAddress() {
229        // Arrange
230        Location location = new Location();
231
232        // Act
233        boolean result = locationController.deleteLocation(1L);
234
235        // Assert
236        assertFalse(result);
237    }
238
239    @Test ✘ AdrianoSCruzP
240    void testUpdateLocationFailureWithEmptyListAndNullAddress() {
241        // Arrange
242        Long locationId = 1L;
243        UpdateLocationResource resource = new UpdateLocationResource(
244            address: null, numDirection: "742", street: "Springfield St",
245            district: "Springfield", city: "Illinois", latitude: -12.0464, longitude: -77.0464
246        );
247
248        Location updatedLocation = new Location(
249            locationId,
250        );
251
252        // Act
253        Exception exception = assertThrows(IllegalArgumentException.class, () ->
254            locationController.updateLocation(resource));
255
256        // Assert
257        assertEquals("Address cannot be null", exception.getMessage());
258    }
259
260    @Test ✘ AdrianoSCruzP
261    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCity() {
262        // Arrange
263        List<Location> locations = new ArrayList<>();
264
265        // Act
266        List<Location> result = locationController.getAllLocations();
267
268        // Assert
269        assertEquals(locations, result);
270    }
271
272    @Test ✘ AdrianoSCruzP
273    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCity() {
274        // Arrange
275        Location location = new Location();
276
277        // Act
278        Location result = locationController.getLocation(1L);
279
280        // Assert
281        assertEquals(location, result);
282    }
283
284    @Test ✘ AdrianoSCruzP
285    void testDeleteLocationFailureWithEmptyListAndNullAddressAndNullCity() {
286        // Arrange
287        Location location = new Location();
288
289        // Act
290        boolean result = locationController.deleteLocation(1L);
291
292        // Assert
293        assertFalse(result);
294    }
295
296    @Test ✘ AdrianoSCruzP
297    void testUpdateLocationFailureWithEmptyListAndNullAddressAndNullCity() {
298        // Arrange
299        Long locationId = 1L;
300        UpdateLocationResource resource = new UpdateLocationResource(
301            address: null, numDirection: "742", street: "Springfield St",
302            district: "Springfield", city: null, latitude: -12.0464, longitude: -77.0464
303        );
304
305        Location updatedLocation = new Location(
306            locationId,
307        );
308
309        // Act
310        Exception exception = assertThrows(IllegalArgumentException.class, () ->
311            locationController.updateLocation(resource));
312
313        // Assert
314        assertEquals("Address cannot be null", exception.getMessage());
315    }
316
317    @Test ✘ AdrianoSCruzP
318    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrict() {
319        // Arrange
320        List<Location> locations = new ArrayList<>();
321
322        // Act
323        List<Location> result = locationController.getAllLocations();
324
325        // Assert
326        assertEquals(locations, result);
327    }
328
329    @Test ✘ AdrianoSCruzP
330    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrict() {
331        // Arrange
332        Location location = new Location();
333
334        // Act
335        Location result = locationController.getLocation(1L);
336
337        // Assert
338        assertEquals(location, result);
339    }
340
341    @Test ✘ AdrianoSCruzP
342    void testDeleteLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrict() {
343        // Arrange
344        Location location = new Location();
345
346        // Act
347        boolean result = locationController.deleteLocation(1L);
348
349        // Assert
350        assertFalse(result);
351    }
352
353    @Test ✘ AdrianoSCruzP
354    void testUpdateLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrict() {
355        // Arrange
356        Long locationId = 1L;
357        UpdateLocationResource resource = new UpdateLocationResource(
358            address: null, numDirection: "742", street: "Springfield St",
359            district: null, city: null, latitude: -12.0464, longitude: -77.0464
360        );
361
362        Location updatedLocation = new Location(
363            locationId,
364        );
365
366        // Act
367        Exception exception = assertThrows(IllegalArgumentException.class, () ->
368            locationController.updateLocation(resource));
369
370        // Assert
371        assertEquals("Address cannot be null", exception.getMessage());
372    }
373
374    @Test ✘ AdrianoSCruzP
375    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreet() {
376        // Arrange
377        List<Location> locations = new ArrayList<>();
378
379        // Act
380        List<Location> result = locationController.getAllLocations();
381
382        // Assert
383        assertEquals(locations, result);
384    }
385
386    @Test ✘ AdrianoSCruzP
387    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreet() {
388        // Arrange
389        Location location = new Location();
390
391        // Act
392        Location result = locationController.getLocation(1L);
393
394        // Assert
395        assertEquals(location, result);
396    }
397
398    @Test ✘ AdrianoSCruzP
399    void testDeleteLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreet() {
400        // Arrange
401        Location location = new Location();
402
403        // Act
404        boolean result = locationController.deleteLocation(1L);
405
406        // Assert
407        assertFalse(result);
408    }
409
410    @Test ✘ AdrianoSCruzP
411    void testUpdateLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreet() {
412        // Arrange
413        Long locationId = 1L;
414        UpdateLocationResource resource = new UpdateLocationResource(
415            address: null, numDirection: "742", street: null,
416            district: null, city: null, latitude: -12.0464, longitude: -77.0464
417        );
418
419        Location updatedLocation = new Location(
420            locationId,
421        );
422
423        // Act
424        Exception exception = assertThrows(IllegalArgumentException.class, () ->
425            locationController.updateLocation(resource));
426
427        // Assert
428        assertEquals("Address cannot be null", exception.getMessage());
429    }
430
431    @Test ✘ AdrianoSCruzP
432    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirection() {
433        // Arrange
434        List<Location> locations = new ArrayList<>();
435
436        // Act
437        List<Location> result = locationController.getAllLocations();
438
439        // Assert
440        assertEquals(locations, result);
441    }
442
443    @Test ✘ AdrianoSCruzP
444    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirection() {
445        // Arrange
446        Location location = new Location();
447
448        // Act
449        Location result = locationController.getLocation(1L);
450
451        // Assert
452        assertEquals(location, result);
453    }
454
455    @Test ✘ AdrianoSCruzP
456    void testDeleteLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirection() {
457        // Arrange
458        Location location = new Location();
459
460        // Act
461        boolean result = locationController.deleteLocation(1L);
462
463        // Assert
464        assertFalse(result);
465    }
466
467    @Test ✘ AdrianoSCruzP
468    void testUpdateLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirection() {
469        // Arrange
470        Long locationId = 1L;
471        UpdateLocationResource resource = new UpdateLocationResource(
472            address: null, numDirection: null, street: null,
473            district: null, city: null, latitude: -12.0464, longitude: -77.0464
474        );
475
476        Location updatedLocation = new Location(
477            locationId,
478        );
479
480        // Act
481        Exception exception = assertThrows(IllegalArgumentException.class, () ->
482            locationController.updateLocation(resource));
483
484        // Assert
485        assertEquals("Address cannot be null", exception.getMessage());
486    }
487
488    @Test ✘ AdrianoSCruzP
489    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitude() {
490        // Arrange
491        List<Location> locations = new ArrayList<>();
492
493        // Act
494        List<Location> result = locationController.getAllLocations();
495
496        // Assert
497        assertEquals(locations, result);
498    }
499
500    @Test ✘ AdrianoSCruzP
501    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitude() {
502        // Arrange
503        Location location = new Location();
504
505        // Act
506        Location result = locationController.getLocation(1L);
507
508        // Assert
509        assertEquals(location, result);
510    }
511
512    @Test ✘ AdrianoSCruzP
513    void testDeleteLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitude() {
514        // Arrange
515        Location location = new Location();
516
517        // Act
518        boolean result = locationController.deleteLocation(1L);
519
520        // Assert
521        assertFalse(result);
522    }
523
524    @Test ✘ AdrianoSCruzP
525    void testUpdateLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitude() {
526        // Arrange
527        Long locationId = 1L;
528        UpdateLocationResource resource = new UpdateLocationResource(
529            address: null, numDirection: null, street: null,
530            district: null, city: null, latitude: null, longitude: -77.0464
531        );
532
533        Location updatedLocation = new Location(
534            locationId,
535        );
536
537        // Act
538        Exception exception = assertThrows(IllegalArgumentException.class, () ->
539            locationController.updateLocation(resource));
540
541        // Assert
542        assertEquals("Address cannot be null", exception.getMessage());
543    }
544
545    @Test ✘ AdrianoSCruzP
546    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLongitude() {
547        // Arrange
548        List<Location> locations = new ArrayList<>();
549
550        // Act
551        List<Location> result = locationController.getAllLocations();
552
553        // Assert
554        assertEquals(locations, result);
555    }
556
557    @Test ✘ AdrianoSCruzP
558    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLongitude() {
559        // Arrange
560        Location location = new Location();
561
562        // Act
563        Location result = locationController.getLocation(1L);
564
565        // Assert
566        assertEquals(location, result);
567    }
568
569    @Test ✘ AdrianoSCruzP
570    void testDeleteLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLongitude() {
571        // Arrange
572        Location location = new Location();
573
574        // Act
575        boolean result = locationController.deleteLocation(1L);
576
577        // Assert
578        assertFalse(result);
579    }
580
581    @Test ✘ AdrianoSCruzP
582    void testUpdateLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLongitude() {
583        // Arrange
584        Long locationId = 1L;
585        UpdateLocationResource resource = new UpdateLocationResource(
586            address: null, numDirection: null, street: null,
587            district: null, city: null, latitude: -12.0464, longitude: null
588        );
589
590        Location updatedLocation = new Location(
591            locationId,
592        );
593
594        // Act
595        Exception exception = assertThrows(IllegalArgumentException.class, () ->
596            locationController.updateLocation(resource));
597
598        // Assert
599        assertEquals("Address cannot be null", exception.getMessage());
600    }
601
602    @Test ✘ AdrianoSCruzP
603    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitude() {
604        // Arrange
605        List<Location> locations = new ArrayList<>();
606
607        // Act
608        List<Location> result = locationController.getAllLocations();
609
610        // Assert
611        assertEquals(locations, result);
612    }
613
614    @Test ✘ AdrianoSCruzP
615    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitude() {
616        // Arrange
617        Location location = new Location();
618
619        // Act
620        Location result = locationController.getLocation(1L);
621
622        // Assert
623        assertEquals(location, result);
624    }
625
626    @Test ✘ AdrianoSCruzP
627    void testDeleteLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitude() {
628        // Arrange
629        Location location = new Location();
630
631        // Act
632        boolean result = locationController.deleteLocation(1L);
633
634        // Assert
635        assertFalse(result);
636    }
637
638    @Test ✘ AdrianoSCruzP
639    void testUpdateLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitude() {
640        // Arrange
641        Long locationId = 1L;
642        UpdateLocationResource resource = new UpdateLocationResource(
643            address: null, numDirection: null, street: null,
644            district: null, city: null, latitude: null, longitude: null
645        );
646
647        Location updatedLocation = new Location(
648            locationId,
649        );
650
651        // Act
652        Exception exception = assertThrows(IllegalArgumentException.class, () ->
653            locationController.updateLocation(resource));
654
655        // Assert
656        assertEquals("Address cannot be null", exception.getMessage());
657    }
658
659    @Test ✘ AdrianoSCruzP
660    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddress() {
661        // Arrange
662        List<Location> locations = new ArrayList<>();
663
664        // Act
665        List<Location> result = locationController.getAllLocations();
666
667        // Assert
668        assertEquals(locations, result);
669    }
670
671    @Test ✘ AdrianoSCruzP
672    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddress() {
673        // Arrange
674        Location location = new Location();
675
676        // Act
677        Location result = locationController.getLocation(1L);
678
679        // Assert
680        assertEquals(location, result);
681    }
682
683    @Test ✘ AdrianoSCruzP
684    void testDeleteLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddress() {
685        // Arrange
686        Location location = new Location();
687
688        // Act
689        boolean result = locationController.deleteLocation(1L);
690
691        // Assert
692        assertFalse(result);
693    }
694
695    @Test ✘ AdrianoSCruzP
696    void testUpdateLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddress() {
697        // Arrange
698        Long locationId = 1L;
699        UpdateLocationResource resource = new UpdateLocationResource(
700            address: null, numDirection: null, street: null,
701            district: null, city: null, latitude: null, longitude: null
702        );
703
704        Location updatedLocation = new Location(
705            locationId,
706        );
707
708        // Act
709        Exception exception = assertThrows(IllegalArgumentException.class, () ->
710            locationController.updateLocation(resource));
711
712        // Assert
713        assertEquals("Address cannot be null", exception.getMessage());
714    }
715
716    @Test ✘ AdrianoSCruzP
717    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirection() {
718        // Arrange
719        List<Location> locations = new ArrayList<>();
720
721        // Act
722        List<Location> result = locationController.getAllLocations();
723
724        // Assert
725        assertEquals(locations, result);
726    }
727
728    @Test ✘ AdrianoSCruzP
729    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirection() {
730        // Arrange
731        Location location = new Location();
732
733        // Act
734        Location result = locationController.getLocation(1L);
735
736        // Assert
737        assertEquals(location, result);
738    }
739
740    @Test ✘ AdrianoSCruzP
741    void testDeleteLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirection() {
742        // Arrange
743        Location location = new Location();
744
745        // Act
746        boolean result = locationController.deleteLocation(1L);
747
748        // Assert
749        assertFalse(result);
750    }
751
752    @Test ✘ AdrianoSCruzP
753    void testUpdateLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirection() {
754        // Arrange
755        Long locationId = 1L;
756        UpdateLocationResource resource = new UpdateLocationResource(
757            address: null, numDirection: null, street: null,
758            district: null, city: null, latitude: null, longitude: null
759        );
760
761        Location updatedLocation = new Location(
762            locationId,
763        );
764
765        // Act
766        Exception exception = assertThrows(IllegalArgumentException.class, () ->
767            locationController.updateLocation(resource));
768
769        // Assert
770        assertEquals("Address cannot be null", exception.getMessage());
771    }
772
773    @Test ✘ AdrianoSCruzP
774    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitude() {
775        // Arrange
776        List<Location> locations = new ArrayList<>();
777
778        // Act
779        List<Location> result = locationController.getAllLocations();
780
781        // Assert
782        assertEquals(locations, result);
783    }
784
785    @Test ✘ AdrianoSCruzP
786    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitude() {
787        // Arrange
788        Location location = new Location();
789
790        // Act
791        Location result = locationController.getLocation(1L);
792
793        // Assert
794        assertEquals(location, result);
795    }
796
797    @Test ✘ AdrianoSCruzP
798    void testDeleteLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitude() {
799        // Arrange
800        Location location = new Location();
801
802        // Act
803        boolean result = locationController.deleteLocation(1L);
804
805        // Assert
806        assertFalse(result);
807    }
808
809    @Test ✘ AdrianoSCruzP
810    void testUpdateLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitude() {
811        // Arrange
812        Long locationId = 1L;
813        UpdateLocationResource resource = new UpdateLocationResource(
814            address: null, numDirection: null, street: null,
815            district: null, city: null, latitude: null, longitude: null
816        );
817
818        Location updatedLocation = new Location(
819            locationId,
820        );
821
822        // Act
823        Exception exception = assertThrows(IllegalArgumentException.class, () ->
824            locationController.updateLocation(resource));
825
826        // Assert
827        assertEquals("Address cannot be null", exception.getMessage());
828    }
829
830    @Test ✘ AdrianoSCruzP
831    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitude() {
832        // Arrange
833        List<Location> locations = new ArrayList<>();
834
835        // Act
836        List<Location> result = locationController.getAllLocations();
837
838        // Assert
839        assertEquals(locations, result);
840    }
841
842    @Test ✘ AdrianoSCruzP
843    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitude() {
844        // Arrange
845        Location location = new Location();
846
847        // Act
848        Location result = locationController.getLocation(1L);
849
850        // Assert
851        assertEquals(location, result);
852    }
853
854    @Test ✘ AdrianoSCruzP
855    void testDeleteLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitude() {
856        // Arrange
857        Location location = new Location();
858
859        // Act
860        boolean result = locationController.deleteLocation(1L);
861
862        // Assert
863        assertFalse(result);
864    }
865
866    @Test ✘ AdrianoSCruzP
867    void testUpdateLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitude() {
868        // Arrange
869        Long locationId = 1L;
870        UpdateLocationResource resource = new UpdateLocationResource(
871            address: null, numDirection: null, street: null,
872            district: null, city: null, latitude: null, longitude: null
873        );
874
875        Location updatedLocation = new Location(
876            locationId,
877        );
878
879        // Act
880        Exception exception = assertThrows(IllegalArgumentException.class, () ->
881            locationController.updateLocation(resource));
882
883        // Assert
884        assertEquals("Address cannot be null", exception.getMessage());
885    }
886
887    @Test ✘ AdrianoSCruzP
888    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddress() {
889        // Arrange
890        List<Location> locations = new ArrayList<>();
891
892        // Act
893        List<Location> result = locationController.getAllLocations();
894
895        // Assert
896        assertEquals(locations, result);
897    }
898
899    @Test ✘ AdrianoSCruzP
900    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddress() {
901        // Arrange
902        Location location = new Location();
903
904        // Act
905        Location result = locationController.getLocation(1L);
906
907        // Assert
908        assertEquals(location, result);
909    }
910
911    @Test ✘ AdrianoSCruzP
912    void testDeleteLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddress() {
913        // Arrange
914        Location location = new Location();
915
916        // Act
917        boolean result = locationController.deleteLocation(1L);
918
919        // Assert
920        assertFalse(result);
921    }
922
923    @Test ✘ AdrianoSCruzP
924    void testUpdateLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddress() {
925        // Arrange
926        Long locationId = 1L;
927        UpdateLocationResource resource = new UpdateLocationResource(
928            address: null, numDirection: null, street: null,
929            district: null, city: null, latitude: null, longitude: null
930        );
931
932        Location updatedLocation = new Location(
933            locationId,
934        );
935
936        // Act
937        Exception exception = assertThrows(IllegalArgumentException.class, () ->
938            locationController.updateLocation(resource));
939
940        // Assert
941        assertEquals("Address cannot be null", exception.getMessage());
942    }
943
944    @Test ✘ AdrianoSCruzP
945    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirection() {
946        // Arrange
947        List<Location> locations = new ArrayList<>();
948
949        // Act
950        List<Location> result = locationController.getAllLocations();
951
952        // Assert
953        assertEquals(locations, result);
954    }
955
956    @Test ✘ AdrianoSCruzP
957    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirection() {
958        // Arrange
959        Location location = new Location();
960
961        // Act
962        Location result = locationController.getLocation(1L);
963
964        // Assert
965        assertEquals(location, result);
966    }
967
968    @Test ✘ AdrianoSCruzP
969    void testDeleteLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirection() {
970        // Arrange
971        Location location = new Location();
972
973        // Act
974        boolean result = locationController.deleteLocation(1L);
975
976        // Assert
977        assertFalse(result);
978    }
979
980    @Test ✘ AdrianoSCruzP
981    void testUpdateLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirection() {
982        // Arrange
983        Long locationId = 1L;
984        UpdateLocationResource resource = new UpdateLocationResource(
985            address: null, numDirection: null, street: null,
986            district: null, city: null, latitude: null, longitude: null
987        );
988
989        Location updatedLocation = new Location(
990            locationId,
991        );
992
993        // Act
994        Exception exception = assertThrows(IllegalArgumentException.class, () ->
995            locationController.updateLocation(resource));
996
997        // Assert
998        assertEquals("Address cannot be null", exception.getMessage());
999    }
1000
1001    @Test ✘ AdrianoSCruzP
1002    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullAddress() {
1003        // Arrange
1004        List<Location> locations = new ArrayList<>();
1005
1006        // Act
1007        List<Location> result = locationController.getAllLocations();
1008
1009        // Assert
1010        assertEquals(locations, result);
1011    }
1012
1013    @Test ✘ AdrianoSCruzP
1014    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullAddress() {
1015        // Arrange
1016        Location location = new Location();
1017
1018        // Act
1019        Location result = locationController.getLocation(1L);
1020
1021        // Assert
1022        assertEquals(location, result);
1023    }
1024
1025    @Test ✘ AdrianoSCruzP
1026    void testDeleteLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullAddress() {
1027        // Arrange
1028        Location location = new Location();
1029
1030        // Act
1031        boolean result = locationController.deleteLocation(1L);
1032
1033        // Assert
1034        assertFalse(result);
1035    }
1036
1037    @Test ✘ AdrianoSCruzP
1038    void testUpdateLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullAddress() {
1039        // Arrange
1040        Long locationId = 1L;
1041        UpdateLocationResource resource = new UpdateLocationResource(
1042            address: null, numDirection: null, street: null,
1043            district: null, city: null, latitude: null, longitude: null
1044        );
1045
1046        Location updatedLocation = new Location(
1047            locationId,
1048        );
1049
1050        // Act
1051        Exception exception = assertThrows(IllegalArgumentException.class, () ->
1052            locationController.updateLocation(resource));
1053
1054        // Assert
1055        assertEquals("Address cannot be null", exception.getMessage());
1056    }
1057
1058    @Test ✘ AdrianoSCruzP
1059    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullAddressAndNullNumDirection() {
1060        // Arrange
1061        List<Location> locations = new ArrayList<>();
1062
1063        // Act
1064        List<Location> result = locationController.getAllLocations();
1065
1066        // Assert
1067        assertEquals(locations, result);
1068    }
1069
1070    @Test ✘ AdrianoSCruzP
1071    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullAddressAndNullNumDirection() {
1072        // Arrange
1073        Location location = new Location();
1074
1075        // Act
1076        Location result = locationController.getLocation(1L);
1077
1078        // Assert
1079        assertEquals(location, result);
1080    }
1081
1082    @Test ✘ AdrianoSCruzP
1083    void testDeleteLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullAddressAndNullNumDirection() {
1084        // Arrange
1085        Location location = new Location();
1086
1087        // Act
1088        boolean result = locationController.deleteLocation(1L);
1089
1090        // Assert
1091        assertFalse(result);
1092    }
1093
1094    @Test ✘ AdrianoSCruzP
1095    void testUpdateLocationFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullAddressAndNullNumDirection() {
1096        // Arrange
1097        Long locationId = 1L;
1098        UpdateLocationResource resource = new UpdateLocationResource(
1099            address: null, numDirection: null, street: null,
1100            district: null, city: null, latitude: null, longitude: null
1101        );
1102
1103        Location updatedLocation = new Location(
1104            locationId,
1105        );
1106
1107        // Act
1108        Exception exception = assertThrows(IllegalArgumentException.class, () ->
1109            locationController.updateLocation(resource));
1110
1111        // Assert
1112        assertEquals("Address cannot be null", exception.getMessage());
1113    }
1114
1115    @Test ✘ AdrianoSCruzP
1116    void testGetAllLocationsFailureWithEmptyListAndNullAddressAndNullCityAndNullDistrictAndNullStreetAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullLatitudeAndNullLongitudeAndNullAddressAndNullNumDirectionAndNullAddressAndNullNumDirectionAndNullAddress() {
1117        // Arrange
1118        List<Location> locations = new ArrayList<>();
1119
1120        // Act
1121        List<Location> result = locationController.getAllLocations();
1122
1123        // Assert
1124        assertEquals(locations, result);
1125    }
1126
1127    @Test ✘ AdrianoSCruzP
1128    void testGetLocationFailureWithEmptyListAndNullAddressAndNullCity
```

Parking Controller Test

The screenshot shows an IDE interface with a project tree on the left and a code editor on the right.

**Project Tree:**

- vehicles
- HomeparkApplication
- resources
- test
  - java
    - @com.homepark.web\_service.core.entities.unit.tests
    - integration.tests
      - LocationControllerIntegrationTest
      - ParkingControllerIntegrationTest
      - ProfileControllerIntegrationTest
      - ReservationControllerIntegrationTest
      - ScheduleControllerIntegrationTest
      - VehicleControllerIntegrationTest
  - HomeparkApplicationTests
- target
- .dockerignore
- .env.development
- .env.production
- .gitattributes
- .gitignore
- docker-compose.yml
- Dockerfile

**Code Editor (ParkingControllerIntegrationTest.java):**

```

1 package com.homepark.web_service.core.integration.tests;
2
3 import com.homepark.web_service.parkings.domain.model.aggregates.Parking;
4 import com.homepark.web_service.parkings.domain.model.commands.CreateParkingCommand;
5 import com.homepark.web_service.parkings.domain.model.entities.Location;
6 import com.homepark.web_service.parkings.domain.model.queries.GetAllParkingQuery;
7 import com.homepark.web_service.parkings.domain.model.valueobjects.ProfileId;
8 import com.homepark.web_service.parkings.domain.services.ParkingCommandService;
9 import com.homepark.web_service.parkings.domain.services.ParkingQueryService;
10 import com.homepark.web_service.parkings.interfaces.rest.ParkingController;
11 import com.homepark.web_service.parkings.interfaces.rest.resources.CreateParkingResource;
12 import com.homepark.web_service.parkings.interfaces.rest.resources.ParkingResource;
13 import com.homepark.web_service.parkings.interfaces.rest.transform.ParkingResourceFromE;
14 import org.junit.jupiter.api.BeforeEach;
15 import org.junit.jupiter.api.Test;
16 import org.mockito.ArgumentMatchers;
17 import org.mockito.Mockito;
18 import org.springframework.http.HttpStatus;
19 import org.springframework.http.ResponseEntity;
20
21 import java.util.Collections;
22 import java.util.Optional;
23
24 import static org.junit.jupiter.api.Assertions.*;
25
26
27 public class ParkingControllerIntegrationTest { ▾ AdrianoSCruzP
28
29     private ParkingCommandService parkingCommandService; 3 usages
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51

```

**Run Tab (ParkingControllerIntegrationTest):**

Tests passed: 2 of 2 tests – 938 ms

- testCreateParkingSuccess() 935 ms
- testGetAllParkingSuccess() 3 ms

C:\Users\user\.jdks\openjdk-23.0.1\bin\java.exe ...  
Mockito is currently self-attaching to enable the inline-mock-maker. This will no longer work in future releases of the JDK. Please add Mockito as a dependency.  
WARNING: A Java agent has been loaded dynamically (C:\Users\user\.m2\repository\net.bytebuddy\byte-buddy-agent\1.15.11\byte-buddy-agent-1.15.11-byte-buddy-agent-1.15.11.jar).  
WARNING: If a serviceability tool is in use, please run with -XX:+EnableDynamicAgentLoading to hide this warning.  
WARNING: If a serviceability tool is not in use, please run with -Djdk.instrument.traceUsage for more information.  
WARNING: Dynamic loading of agents will be disallowed by default in a future release.  
OpenJDK 64-Bit Server VM warning: Sharing is only supported for boot loader classes because bootstrap classpath has been appended.

Process finished with exit code 0

## Reservation Controller Test

The screenshot shows an IDE interface with a project tree on the left and a code editor on the right.

**Project Tree:**

- vehicles
- HomeparkApplication
- resources
- test
  - java
    - @com.homepark.web\_service.core.entities.unit.tests
    - integration.tests
      - LocationControllerIntegrationTest
      - ParkingControllerIntegrationTest
      - ProfileControllerIntegrationTest
      - ReservationControllerIntegrationTest
      - ScheduleControllerIntegrationTest
      - VehicleControllerIntegrationTest
  - HomeparkApplicationTests
- target
- .dockerignore
- .env.development

**Code Editor (ReservationControllerIntegrationTest.java):**

```

1 package com.homepark.web_service.core.integration.tests;
2
3 import ...
4
5 public class ReservationControllerIntegrationTest { ▾ Amer Llamo Sánchez
6     private ReservationCommandService reservationCommandService; 10 usages
7     private ReservationQueryService reservationQueryService; 19 usages
8     private ReservationController reservationController; 16 usages
9
10
11     @BeforeEach ▾ Amer Llamo Sánchez
12     void setUp() {
13         reservationCommandService = Mockito.mock(ReservationCommandService.class);
14         reservationQueryService = Mockito.mock(ReservationQueryService.class);
15         reservationController = new ReservationController(reservationCommandService, reservationQueryService);
16     }
17
18     private Reservation createMockReservation(Long id, Status status, LocalDate date, LocalTime startTime,
19                                             LocalTime endTime, Double totalFare, int hoursRegistered,
20                                             Long hostId, Long guestId, Long parkingId, Long vehicleId) {
21
22         Reservation reservation = new Reservation();
23         reservation.setId(id);
24         reservation.setStatus(status);
25     }
26
27
28
29
30
31
32
33
34
35
36
37
38
39
39
40
41
42
43
44
45
46
47
48
49
50
51

```

**Run Tab (ReservationControllerIntegrationTest):**

Tests passed: 15 of 15 tests – 1 sec 64 ms

C:\Users\user\.jdks\openjdk-23.0.1\bin\java.exe ...  
Mockito is currently self-attaching to enable the inline-mock-maker. This will no longer work in future releases of the JDK. Please add Mockito as a dependency.  
WARNING: A Java agent has been loaded dynamically (C:\Users\user\.m2\repository\net.bytebuddy\byte-buddy-agent\1.15.11\byte-buddy-agent-1.15.11-byte-buddy-agent-1.15.11.jar).  
WARNING: If a serviceability tool is in use, please run with -XX:+EnableDynamicAgentLoading to hide this warning.  
WARNING: If a serviceability tool is not in use, please run with -Djdk.instrument.traceUsage for more information.  
WARNING: Dynamic loading of agents will be disallowed by default in a future release.  
OpenJDK 64-Bit Server VM warning: Sharing is only supported for boot loader classes because bootstrap classpath has been appended.

Process finished with exit code 0

## Schedule Controller Test

The screenshot shows a Java-based IDE interface. On the left, the project structure is displayed under 'Project' (Alt+1). It includes a '.mvn' folder, a 'src' folder containing 'main' and 'test' sub-folders. 'main/java/com.homeypark.web\_service' contains 'iam', 'parkings', 'profiles', 'reservations', 'shared', and 'vehicles' packages, along with a 'HomeyparkApplication' class. 'test/java/com.homeypark.web\_service' contains 'core', 'entities.unit.tests', and 'integration.tests' packages, which include 'LocationControllerIntegrationTest', 'ParkingControllerIntegrationTest', 'ProfileControllerIntegrationTest', 'ReservationControllerIntegrationTest', 'ScheduleControllerIntegrationTest', and 'VehicleControllerIntegrationTest' classes. The right side of the interface shows the code editor for 'ScheduleControllerIntegrationTest.java'. The code defines a test class 'ScheduleControllerIntegrationTest' with a constructor that injects 'ScheduleCommandService', 'ScheduleQueryService', and 'ScheduleController'. It includes a setup method using Mockito to mock these services and a test method 'testCreateScheduleSuccess' that arranges a 'CreateScheduleResource' object with specific parameters (parkingId: 1L, day: "MONDAY", LocalTime.of(hour: 8, minute: 0), LocalTime.of(hour: 18, minute: 0)) and creates a 'Parking' object with id 1L.

```
package com.homeypark.web_service.core.integration.tests;

import ...;

public class ScheduleControllerIntegrationTest { AdrianoSCruzP

    private ScheduleCommandService scheduleCommandService; 4 usages
    private ScheduleQueryService scheduleQueryService; 3 usages
    private ScheduleController scheduleController; 4 usages

    @BeforeEach AdrianoSCruzP
    void setUp() {
        scheduleCommandService = Mockito.mock(ScheduleCommandService.class);
        scheduleQueryService = Mockito.mock(ScheduleQueryService.class);
        scheduleController = new ScheduleController(scheduleCommandService, scheduleQueryService);
    }

    @Test AdrianoSCruzP
    void testCreateScheduleSuccess() {
        // Arrange
        CreateScheduleResource resource = new CreateScheduleResource(
            parkingId: 1L, day: "MONDAY", LocalTime.of(hour: 8, minute: 0), LocalTime.of(hour: 18, minute: 0)
        );

        Parking parking = new Parking();
        parking.setId(1L);
    }
}
```

### 6.1.3. Core Behavior-Driven Development

The screenshot shows a Java test project structure in a code editor. The project tree includes a 'test' folder containing a 'java' folder with 'com.homeypark.web\_service' and 'core' packages, and a 'resources.Features' folder containing several feature files: US44EditarPerfil.feature, US09RegistroUsuario.feature, US10IniciarSesion.feature, US11CerrarSesion.feature, US39VisualizarVehiculos.feature, US40RegistrarVehiculo.feature, US41EditarVehiculo.feature, US42EliminarVehiculo.feature, and US43VisualizarPerfil.feature.

The code editor displays the content of the 'US09RegistroUsuario.feature' file:

```
1 ► Feature: US09 Registro de usuario
  Como usuario
    quiero registrarme con correo, contraseña y datos de perfil
    para crear una cuenta.

  Scenario Outline: Registro exitoso de usuario
    Given el usuario ingresa su <correo>, <contraseña> y <datos de perfil> válidos
    When envía la solicitud de registro
    Then el sistema crea una nueva cuenta y retorna el perfil del usuario

  Examples:
    | correo           | contraseña | datos de perfil |
    | user@email.com | pass123   | nombre: Juan, apellido: Pérez |

  Scenario Outline: Registro fallido por datos inválidos
    Given el usuario ingresa un <correo> o <contraseña> inválidos
    When envía la solicitud de registro
    Then el sistema retorna un <mensaje de error>

  Examples:
    | correo           | contraseña | mensaje de error |
    | user@email.com | pass123   | "Correo electrónico inválido" |
    | user@email.com |            | "La contraseña es obligatoria" |
```

```

US10IniciarSesion.feature ×
1 ► Feature: US10 Iniciar sesión
2   Como usuario
3     quiero iniciar sesión con correo y contraseña
4     para acceder a mi cuenta.
5
6 ► Scenario Outline: Inicio de sesión exitoso
7   Given el usuario ingresa su <correo> y <contraseña> válidos
8   When envía la solicitud de inicio de sesión
9   Then el sistema autentica al usuario y retorna un <token de acceso>
10
11 Examples:
12   | correo          | contraseña | token de acceso |
13   | user@email.com | pass123    | "eyJhbGciOi..." |
14
15 ► Scenario Outline: Fallo al iniciar sesión por credenciales inválidas
16   Given el usuario ingresa un <correo> o <contraseña> inválidos
17   When envía la solicitud de inicio de sesión
18   Then el sistema retorna un <mensaje de error>
19
20 Examples:
21   | correo          | contraseña | mensaje de error |
22   | user@email.com | wrongpass   | "Credenciales inválidas" |
23   | bad@email.com  | pass123    | "Usuario no encontrado" |

```

#### 6.1.4. Core System Tests

US-09	Registrarse	Given el usuario está en la página de registro, mWhen completa todos los campos requeridos y presiona "Registrarse", Then se crea la cuenta y se redirige al login. Given el usuario ingresa un correo ya registrado, When intenta crear la cuenta, Then se muestra un mensaje de error.
-------	-------------	--

Project: HomeyPark US Testing

Executing	Command	Target
US-09	double click	name=password
	click	name=password
	click	name=password
	double click	name=password
	click	name=password
	click	name=repeatPassword
	type	name=repeatPassword
	click	id=termsAndConditions
	run script	window.scrollTo(0,0)
	select frame	index=1
	click	css=.recaptcha-checkbox-border
	select frame	relative=parent
	click	css=.p-button
	mouse over	css=.p-ink
	mouse out	css=.p-ink

**US-10 Iniciar sesión** Given el usuario tiene una cuenta registrada, When ingresa credenciales válidas y presiona "Iniciar sesión", Then accede a su dashboard.  
 Given el usuario Ingresa credenciales incorrectas, When Intenta iniciar sesión, Then se muestra un mensaje de error y no se permite el acceso.

Project: HomeyPark US Testing 

Executing	Command	Target
US-10	1 open /login	
	2 set window size 945x1060	
	3 type id=email	
	4 type name=password	
	5 click id=email	
	6 click id=email	
	7 double click id=email	
	8 click name=password	
	9 click name=password	
	10 click name=password	
	11 double click name=password	
	12 select frame index=0	
	13 click css=.recaptcha-checkbox-border	
	14 select frame relative=parent	
	15 click css=.p-button	
	16 mouse over css=.p-ink	

Runs: 0 Failures: 0

Command  

Target  

Value

Description

**US-12 Buscar estacionamientos por dirección** Given el usuario ingresa una dirección válida, When presiona el botón de buscar, Then se muestran las cocheras disponibles en esa zona. Given el usuario ingresa una dirección inválida, When intenta buscar, Then se muestra un mensaje de error.

Project: HomeyPark US Testing 

Tests	Command	Target
US-09	1 open /find-your-parking	
US-10	2 set window size 1936x1096	
US-12	3 click linkText=Buscar un garage	
US-13	4 mouse over linkText=Buscar un garage	
	5 mouse out linkText=Buscar un garage	
	6 click css=.p-inputtext	
	7 type css=.p-inputtext	
	8 mouse up css=.gm-style > div > div:nth-child(2)	

**US-13 Visualizar estacionamientos en mapa** Given hay estacionamientos disponibles When el usuario accede al mapa, Then se muestran los marcadores en la ubicación correcta. Given no hay estacionamientos, When el usuario accede al mapa, Then se muestra un mensaje indicando "Sin resultados".

## Project: HomeyPark US Testing

Tests	+		Command	Target
US-09			1 open	/find-your-parking
US-10			2 set window size	1936x1096
US-12			3 click	css=.yNHHyP-marker-view:nth-child(1)
US-13				

**US-15 Ver detalle de cochera** Given el usuario selecciona un estacionamiento, When accede al detalle, Then ve información completa: ubicación, precio, horario, calificación. Given hay un error al cargar el detalle, When intenta acceder, Then se muestra un mensaje de error.

Tests	+	Command	Target
US-15*		1 open	/find-your-parking
US-18*		2 set window size	1680x1060
US-19*		3 click	css=.yNHHyP-marker-view:nth-child(1)
US-40*		4 mouse down	css=.p-button-outlined > .p-button-label
		5 mouse up	css=.p-ink-active
		6 click	css=.p-button-outlined
		7 mouse over	css=.p-ink-active

**US-18 Registrar cochera** Given el anfitrión llena todos los campos requeridos, When presiona "Registrar cochera", Then la cochera se guarda exitosamente. Given falta información, When intenta registrar, Then se muestra un mensaje de validación.

## Project: HomeyPark US Testing2\*

Tests	+	Command
US-15*		1 open
US-18*		2 set window size
US-19*		3 click
US-40*		4 mouse down
		5 mouse up
		6 click
		7 mouse over
		8 click
		9 type
		10 mouse up
		11 close

**US-19 Ver cocheras registradas** Given el anfitrión tiene cocheras registradas, When accede a la sección de cocheras, Then se muestra el listado con sus detalles. Given el anfitrión no tiene cocheras, When accede a la sección, Then se muestra un mensaje indicando que no hay cocheras registradas.

Project: **HomeyPark US Testing2\***

The screenshot shows a test runner interface with the following details:

- Tests:** A dropdown menu with a '+' icon.
- Search tests...**: A search bar with a magnifying glass icon.
- URL:** http://localhost:5173
- Test List:**
  - US-15\*
  - US-18\*
  - US-19\*** (highlighted in blue)
  - US-40\*
- Command List:**

Index	Command
1	open
2	set window size
3	click

**US- 40 Añadir vehículo Given el usuario completa los campos del vehículo, When presiona “Guardar”, Then el vehículo se registra y aparece en la lista. Given falta un campo obligatorio, When intenta guardar, Then se muestra un mensaje de error.**

Project: **HomeyPark US Testing2\***

The screenshot shows a detailed view of a test case (US-40) with the following details:

- Executing:** A dropdown menu with a '+' icon.
- URL:** http://localhost:5173
- Test:** US-40\*
- Command Table:**

Index	Command	Target	Value
1	open	/find-your-parking	
2	set window size	945x1060	
3	click	linkText=Mis vehículos	
4	mouse down	css=.mt-4 .p-button-label	
5	mouse up	css=.p-ink-active	
6	click	css=.mt-4 > .p-button	
7	mouse over	css=.p-ink-active	
8	mouse out	css=.p-ink-active	
9	click	id=brand	
10	type	id=brand	Toyota
11	click	id=licensePlate	
12	type	id=licensePlate	ABC123
13	click	id=model	
14	type	id=model	Corolla
15	mouse down	css=w-full:nth-child(1) > .p-button-label	
16	mouse up	css=.p-ink-active	

## Capítulo VII: DevOps Practices

### 6.2. Static testing & Verification

#### 6.2.1. Static Code Analysis

##### 6.2.1.1. Coding standard & Code conventions

##### 6.2.1.2. Code Quality & Code Security

#### 6.2.2. Reviews

### 6.3. Validation Interviews

#### 6.3.1. Diseño de Entrevistas

#### 6.3.2. Registro de Entrevistas

#### 6.3.3. Evaluaciones según heurísticas

### 6.4. Auditoría de Experiencias de Usuario

## 6.4.1. Auditoría realizada

### 6.4.1.1. Información del grupo auditado

### 6.4.1.2. Cronograma de auditoría realizada

### 6.4.1.3. Contenido de auditoría realizada

## 6.4.2. Auditoría recibida

### 6.4.2.1. Información del grupo auditor

### 6.4.2.2. Cronograma de auditoría recibida

### 6.4.2.3. Contenido de auditoría recibida

### 6.4.2.4. Resumen de modificaciones para subsanar hallazgos

## 7.1. Continuous Integration

### 7.1.1. Tools and Practices

Para implementar la Integración Continua (CI), utilizamos GitHub Actions, una herramienta de automatización integrada en GitHub que permite construir, probar y verificar automáticamente cada push o pull request a la rama main

#### Tools:

- GitHub Actions: Es la herramienta principal para la automatización del pipeline de integración. Permite ejecutar tareas como compilación, pruebas y ejecución de workflows en cada evento del repositorio.
- Docker: Se emplea para contenerizar la aplicación backend, garantizando que el entorno de ejecución sea consistente desde el desarrollo hasta la producción.
- Render: Plataforma utilizada para desplegar automáticamente la aplicación Spring Boot en la nube mediante la opción webhooks.

#### Practices:

- Disparadores por rama (main): Cada commit a esta rama activa el pipeline.
- Compilación automática: El código es construido dentro del pipeline usando Maven.
- Pruebas automatizadas: Se ejecutan pruebas unitarias para validar la estabilidad del sistema.
- Contenerización: El backend se empaqueta en una imagen Docker preparada para producción.
- Despliegue automatizado (hook): Una vez validado el pipeline, se envía una solicitud al webhook de Render para iniciar el despliegue.

### 7.1.2. Build & Test Suite Pipeline Components

El pipeline de Integración Continua implementado en GitHub Actions se estructura en varias etapas que garantizan que cada cambio en el código sea correctamente validado antes de su despliegue. Estas etapas permiten detectar errores tempranamente, asegurar la calidad del software y mantener el proyecto en un estado siempre desplegable.

#### Componentes del Pipeline del Backend:

##### 1. Checkout del código fuente:

- El pipeline comienza con la acción de checkout, la cual descarga el contenido del repositorio para que las siguientes tareas puedan ejecutarse sobre la versión más reciente del código.

##### 2. Configuración del entorno:

- Se configura el entorno de ejecución instalando Java 17 (distribución Temurin) y preparando el entorno para compilar el proyecto.

##### 3. Construcción del proyecto (build):

- Se ejecuta mvn clean install, lo que permite compilar el código fuente y resolver todas las dependencias declaradas en el pom.xml.
- En esta etapa también se generan los artefactos necesarios para el despliegue (por ejemplo, el jar final).

##### 4. Ejecución de pruebas:

- Se ejecuta mvn test, lo que dispara las pruebas unitarias integradas en el proyecto.
- Esta fase es crítica para verificar que los cambios realizados no introduzcan fallos en la lógica de negocio ni rompan funcionalidades existentes.

##### 5. Notificación de despliegue (solo en rama main):

- Si el commit pertenece a la rama main y todas las fases anteriores se completan exitosamente, se ejecuta un curl al webhook de Render (Render Deploy Hook) para iniciar automáticamente el despliegue.

## 7.2. Continuous Delivery

### 7.2.1. Tools and Practices

El objetivo de la Entrega Continua (Continuous Delivery) es garantizar que la aplicación esté siempre en un estado desplegable y validado, de modo que pueda ser publicada en producción de manera rápida y segura con una simple aprobación manual. Para lograrlo, se integraron herramientas que permiten automatizar todo el proceso hasta la etapa previa al despliegue final.

#### Tools:

- GitHub Actions: Es la herramienta central del pipeline, donde se automatizan las etapas de construcción, pruebas y despliegue condicional. Está configurado para detectar cambios en las ramas main y develop, permitiendo un flujo de validación constante.
- Docker: Utilizado para contenerizar la aplicación backend desarrollada en Spring Boot. Garantiza un entorno de ejecución coherente entre desarrollo, pruebas y producción, reduciendo riesgos por diferencias de configuración.
- Render Deploy Hook: Se configura un webhook de Render que permite disparar el despliegue desde GitHub Actions. En el caso de Entrega Continua, este webhook puede activarse manualmente o de forma controlada desde un entorno intermedio como develop.

#### Practices:

- Feature Branching: Cada nueva funcionalidad se desarrolla en una rama independiente. Luego de pasar pruebas y revisión de código, estas ramas se fusionan en develop (entorno de staging) o en main (producción).
- Despliegue Automatizado: En la rama main, el pipeline activa automáticamente el webhook de Render, lo que produce un despliegue inmediato al entorno de producción una vez que se validan los cambios. Esto asegura que toda versión fusionada en main pase directamente a producción sin intervención manual.
- Separación de entornos (.env): Se utilizaron archivos .env para separar claramente las configuraciones de desarrollo y producción, lo que facilita validaciones intermedias antes de desplegar versiones finales.

### 7.2.2. Stages Deployment Pipeline Components

#### Integración Continua (CI):

Cada vez que se realiza un commit o pull request hacia la rama develop o main, GitHub Actions ejecuta automáticamente un pipeline que compila el proyecto con Maven, corre las pruebas y valida la construcción. Este paso garantiza que el código esté siempre en un estado desplegable.

#### Validación previa al despliegue:

La validación se lleva a cabo mediante la ejecución del build y pruebas en un entorno aislado definido por el contenedor Docker. Esto permite asegurar que la aplicación sea coherente y funcional antes de cualquier despliegue.

#### Despliegue automático a producción (main):

Cuando los cambios se fusionan a la rama main, el pipeline ejecuta un paso adicional que activa un Webhook de Render. Esto produce un despliegue inmediato de la nueva versión de la aplicación backend (Spring Boot), de forma completamente automatizada.

#### Separación de entornos:

Se han definido configuraciones diferenciadas mediante variables de entorno en archivos .env, lo que permite manejar distintos entornos (por ejemplo, desarrollo y producción) sin modificar el código fuente.

#### Monitoreo y observabilidad (integrado en Render):

Render proporciona monitoreo básico de la aplicación desplegada. Si el despliegue falla o hay errores de inicialización, estos quedan registrados automáticamente en el dashboard, permitiendo su revisión y solución.

## 7.3. Continuous Deployment

### 7.3.1. Tools and Practices

El objetivo de Continuous Deployment (CD) es que cada cambio validado automáticamente se despliegue directamente al entorno de producción sin intervención manual, siempre y cuando pase todas las pruebas previas definidas.

#### Tools:

- GitHub Actions: Utilizado como motor de automatización del pipeline CI/CD. Permite detectar cambios en la rama main y ejecutar una serie de pasos automatizados, entre ellos el despliegue continuo.
- Docker: Se utiliza para contenerizar la aplicación backend (Java Spring Boot). Esto garantiza que el entorno de ejecución en producción sea idéntico al entorno en el que se realizan las pruebas, reduciendo el riesgo de errores por diferencias de configuración.
- Render: Plataforma encargada de ejecutar el entorno de producción. Se configuró para que despliegue automáticamente la nueva versión de la aplicación cada vez que recibe una señal (deploy hook) desde GitHub Actions.

#### Practices:

- Despliegue basado en commits (commit-based deployment): Cada vez que se realiza un push a la rama main, GitHub Actions ejecuta el flujo completo: construcción, pruebas, y si todo es exitoso, activa el hook de despliegue en Render.
- Pipeline completamente automatizado: No hay intervención manual entre la validación del código y el despliegue final. Esto permite entregas más rápidas, frecuentes y confiables al entorno productivo.
- Rollback manual supervisado: Si bien el despliegue es automático, en caso de fallos se debe hacer un rollback manual en Render, seleccionando una versión anterior. Esto permite control frente a errores críticos sin automatizar retrocesos que puedan agravar fallas si no son correctamente evaluadas.

### 7.3.2. Production Deployment Pipeline Components

El pipeline de despliegue a producción está completamente automatizado y se ejecuta cada vez que se realiza un push en la rama main. El objetivo es que la nueva versión de la aplicación backend llegue al entorno productivo sin intervención manual, garantizando agilidad y confiabilidad.

#### Componentes del pipeline del Backend:

1. Activación automática por GitHub Actions: ante un push a main, se inicia el pipeline.
2. Compilación y pruebas: el código se construye con Maven y se ejecutan pruebas unitarias.
3. Despliegue con Render Deploy Hook: si todo es exitoso, GitHub Actions dispara un webhook que comunica a Render que debe desplegar la nueva versión.
4. Ejecución en contenedor: Render construye la imagen de la app y la ejecuta dentro de un contenedor Docker.
5. Monitoreo automático: Render monitorea la aplicación desplegada, reiniciando si detecta fallos.

## 7.4. Continuous Monitoring

### 7.4.1. Tools and Practices

### 7.4.2. Monitoring Pipeline Components

### 7.4.3. Alerting Pipeline Components

### 7.4.4. Notification Pipeline Components

## 8.1. Experiment Planning

### 8.1.1. As-Is Summary

### 8.1.2. Raw Material: Assumptions, Knowledge Gaps, Ideas, Claims

### 8.1.3. Experiment-Ready Questions

### 8.1.4. Question Backlog

### 8.1.5. Experiment Cards

## 8.2. Experiment Design

### 8.2.1. Hypotheses

### 8.2.2. Measures

### 8.2.3. Conditions

### 8.2.4. Scale Calculations and Decisions

### 8.2.5. Methods Selection

### 8.2.6. Data Analytics: Goals, KPIs and Metrics Selection

### 8.2.7. Web and Mobile Tracking Plan

## 8.3. Experimentation

### 8.3.1. To-Be User Stories

### 8.3.2. To-Be Product Backlog

## Conclusiones, Bibliografía y Anexos

---

- Conclusiones

Definir la viabilidad y el enfoque del proyecto HomeyPark como solución a la escasez de estacionamientos urbanos. Comprender las necesidades de conductores y propietarios, los principales usuarios. Analizar la competencia para identificar oportunidades de diferenciación. Diseñar la arquitectura, UI y UX de la aplicación móvil y web. Adoptar la metodología Lean UX para un desarrollo iterativo y centrado en el usuario. Organizar el equipo y asignar roles para una gestión eficiente del proyecto. Identificar desafíos y riesgos iniciales para planificar estrategias de mitigación. En resumen, esta fase inicial establece una base sólida para el desarrollo de HomeyPark, demostrando la capacidad del equipo para abordar el problema, entender a los usuarios y diseñar una solución tecnológica viable.

Durante esta segunda fase del proyecto, se consolidaron avances clave en la implementación del producto. Se refactorizaron los módulos del frontend y backend, siguiendo principios de arquitectura limpia y el patrón CQRS, lo cual permitió una mejor organización de la lógica de negocio y una mayor escalabilidad del sistema. Un aspecto importante fue la realización de pruebas a diferentes niveles: se implementaron pruebas unitarias, integrales y funcionales, cubriendo tanto los servicios del backend como los flujos principales del sistema, garantizando así la robustez del producto ante distintos escenarios de uso. Por otro lado, se configuraron pipelines de integración y despliegue continuo (CI/CD) mediante GitHub Actions y Render, lo cual aseguró la entrega automática y confiable de cada nueva versión del backend en producción. En síntesis, la segunda etapa del proyecto consolidó la transición de la propuesta conceptual a un producto funcional.

Se completaron desarrollos clave del frontend, backend y landing page, se validaron funcionalidades mediante pruebas unitarias, integrales y funcionales, y se implementaron pipelines CI/CD que garantizan la calidad y continuidad del despliegue. Estos avances fortalecen la viabilidad técnica del sistema y preparan el terreno para una siguiente fase de validación con usuarios reales.

- **Bibliografía**

El Comercio. (2024). Congestión vehicular en Lima: La ciudad más lenta de América Latina con solo 14.5 km/h en hora punta. <https://elcomercio.pe/lima/congestion-vehicular-en-lima-la-ciudad-mas-lenta-de-america-latina-con-solo-145-kmh-en-hora-punta-trafico-caos-vehicular-ultimas-noticia/>

- **Anexos**

Video del TP: [https://upcedupe-my.sharepoint.com/:g/personal/u202210846\\_upc\\_edu\\_pe/EWvS3mC9b-FAgdP1\\_0rQLJcBEzgi-kD3iAUFPG98dw2vXg?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOjPbmVEcmI2ZUZvcKJ1c2luZXNzliwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IlldlYilsInJlZmVycmFsTW9kZSI6InZpZXciLCJyZWZlcnJhbFZpZXciOjJNeUZpbGVzTGlua0NvcHkifX0&e=QgTiZB](https://upcedupe-my.sharepoint.com/:g/personal/u202210846_upc_edu_pe/EWvS3mC9b-FAgdP1_0rQLJcBEzgi-kD3iAUFPG98dw2vXg?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOjPbmVEcmI2ZUZvcKJ1c2luZXNzliwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IlldlYilsInJlZmVycmFsTW9kZSI6InZpZXciLCJyZWZlcnJhbFZpZXciOjJNeUZpbGVzTGlua0NvcHkifX0&e=QgTiZB)

Landing Page: <https://homey-park-experiments.web.app/>

Web App: <https://homey-park-experiments-app.web.app/login>

Organización en GitHub: <https://github.com/orgs/NetviaOrganization/repositories>

Trello: <https://trello.com/b/TghZeTMj/product-backlog-v2>