

Informe de Trabajo Final

Universidad Peruana de Ciencias Aplicadas



Ingeniería de Software

Fundamentos de Arquitectura de software - 6327

Docente: Jorge Luis Delgado Vite

Startup: TinkuyTech

Producto: Ñango

Team members:

Nombre	Código
Gamarra Vega, Anderson Jose	u202016154
Nanfuñay Liza, Pedro Jesús	u202215462
Quijandria Araneda Vicente	U201822697
Hallasi Saravia, Miguel Angel	U202312391
Valera Garcés, Samuel Ignacio	U202111952

Ciclo 2025-02

Registro de versiones del informe

Versión	Fecha	Autor	Descripción de Modificación
TB1	06/09/2025	TinkuyTech	Redacción de los capítulos: Capítulo I: Introducción Capítulo II: Requirements & Analysis Capítulo III: Requirements Specification
TB2	24/09/2025	TinkuyTech	Redacción de los capítulos: Capítulo IV: Product Architecture Design

Student Outcomes

Criterio específico	Acciones realizadas	Conclusiones
Actualiza conceptos y conocimientos necesarios para su desarrollo profesional y en especial para su proyecto en soluciones de software.	Gamarra Vega, Anderson José TB1 <ul style="list-style-type: none"> - Redacté el Lean UX Canvas, relacionando hipótesis y supuestos con los objetivos del producto, lo que me permitió afianzar mis conocimientos en metodologías ágiles de diseño. - Contribuí en la construcción del Product Backlog, aplicando técnicas de priorización que me ayudaron a fortalecer mis competencias en gestión de requisitos y planificación de software. 	TB1 <p>Cada integrante actualizó y aplicó conceptos fundamentales en Lean UX, análisis de usuarios, especificación de requisitos y principios de arquitectura de software. El equipo en conjunto demostró que la aplicación práctica de estos conocimientos fortaleció sus competencias profesionales y mejoró la calidad del proyecto.</p>

Criterio específico	Acciones realizadas	Conclusiones
		de la ingeniería de software, tales como Domain-Driven Design, Attribute-Driven Design, microservicios y patrones arquitectónicos. Con ello garantizamos que nuestra plataforma responda adecuadamente a requisitos funcionales y no funcionales.
	TB2 <ul style="list-style-type: none">- Aplicué tácticas de arquitectura y definí Architectural Drivers para el Capítulo IV, fortaleciendo mis conocimientos en diseño escalable y mantenable. TB3 <ul style="list-style-type: none">- Implementé mejoras en el <i>dockerizado</i> y <i>deploy</i> del entorno productivo, aplicando prácticas avanzadas de CI/CD para optimizar la integración y el despliegue automatizado.- Documenté el flujo de despliegue continuo, reforzando mis conocimientos en contenedorización, orquestación y monitoreo de microservicios. TP <ul style="list-style-type: none">- Realicé el <i>deploy</i> y <i>dockerización</i> de los tres microservicios del backend.- Desarrollé la <i>landing page</i> del proyecto y gestioné su <i>deploy</i> en el entorno productivo.	TB3 Durante esta etapa, el equipo profundizó en la implementación técnica y la optimización de los módulos del sistema. Se reforzaron conocimientos en integración continua, despliegue automatizado, comunicación asíncrona, seguridad de acceso y diseño web moderno. Esto consolidó las habilidades prácticas de los integrantes en entornos de desarrollo reales y escalables.
	Nanfuñay Liza, Pedro Jesús TB1 <ul style="list-style-type: none">- Redacté puntos esenciales para el desarrollo del presente proyecto como el Lean UX Assumptions, Empathy Maps y User Stories, aplicando conocimientos adquiridos de manera eficiente y clara, y buenas prácticas que me permitieron actualizar mis conocimientos y fortalecer mis competencias en el desarrollo de software. TB2 <ul style="list-style-type: none">- Redacté principios y enfoques arquitectónicos, seleccionando estilos y patrones que mejor se alinean con las necesidades del proyecto. TB3 <ul style="list-style-type: none">- Implementé y validé el módulo de <i>mensajería interna</i> del sistema, aplicando principios de comunicación asíncrona y manejo de colas.- Profundicé mis conocimientos en APIs REST y WebSockets, integrando de manera eficiente la mensajería con el backend. TP <ul style="list-style-type: none">- Desarrollé y ejecuté <i>testing funcional</i> y <i>pruebas unitarias</i> para los microservicios y el frontend.- Participé en la gestión y supervisión de las <i>code guidelines</i>, asegurando consistencia en la base de código.	TP El equipo consolidó las etapas finales del proyecto mediante la implementación, despliegue y aseguramiento de la calidad del producto. Samuel y Anderson lideraron los procesos de desarrollo y despliegue, mientras que Pedro y Miguel fortalecieron la calidad técnica mediante pruebas y lineamientos de código. Esto evidenció una comprensión integral del ciclo de vida del software, desde el desarrollo hasta la puesta en producción.
	Quijandria Araneda, Vicente TB1 <ul style="list-style-type: none">- Desarrollé la sección de antecedentes y problemática, aplicando herramientas de análisis para contextualizar el proyecto.- Participé en la redacción de User Stories, lo que me permitió reforzar mi capacidad de transformar hallazgos en especificaciones claras y accionables. TB2 <ul style="list-style-type: none">- Elaboré el Context Diagram y los ViewPoints Diagram para representar la arquitectura del sistema. TB3 <ul style="list-style-type: none">- Desarrollé el módulo de <i>reportes</i> del sistema, implementando consultas y visualizaciones dinámicas que facilitan la toma de decisiones.- Aplicué técnicas de optimización de consultas y generación de reportes en tiempo real, reforzando mis conocimientos en bases de datos y presentación de datos. TP <ul style="list-style-type: none">- Desarrollé y ejecuté <i>testing funcional</i> y <i>pruebas unitarias</i> para los microservicios y el frontend.	

Criterio específico	Acciones realizadas	Conclusiones
	<p>Hallasi Saravia, Miguel Ángel</p> <p>TB1</p> <ul style="list-style-type: none"> - Redacté la sección de segmentos objetivo, lo cual me permitió aplicar conceptos de segmentación y análisis de usuarios. - Participé activamente en entrevistas y en la construcción del Impact Map, afianzando mis conocimientos en la identificación de métricas de valor y validación de usuarios. <p>TB2</p> <ul style="list-style-type: none"> - Documenté funcionalidades principales y Quality Attribute Scenarios, relacionando requisitos con la arquitectura. <p>TB3</p> <ul style="list-style-type: none"> - Diseñé y desarrollé la <i>landing page final</i> del proyecto, aplicando buenas prácticas de diseño web responsive y optimización de rendimiento. - Profundicé mis conocimientos en frameworks frontend y experiencia de usuario, asegurando coherencia visual y técnica con el resto del sistema. <p>TP</p> <ul style="list-style-type: none"> - Ejecuté pruebas unitarias y de integración para garantizar la calidad del código. - Colaboré en el <i>code guidelines management</i>, reforzando la estandarización de buenas prácticas dentro del equipo. 	
	<p>Valera Garcés, Samuel Ignacio</p> <p>TB1</p> <ul style="list-style-type: none"> - Redacté la descripción de la startup y documenté escenarios As-Is y To-Be, lo que me ayudó a profundizar en el análisis de procesos y a fortalecer mis competencias en modelado de escenarios. <p>TB2</p> <ul style="list-style-type: none"> - Diseñé el diagrama de base de datos relacional/no relacional, definí Design Patterns y dirigí las ADD Iterations del Capítulo IV. <p>TB3</p> <ul style="list-style-type: none"> - Realicé la <i>modificación y optimización de los módulos IAM (Identity and Access Management) y Ride</i>, mejorando la seguridad y eficiencia del sistema. - Apliqué buenas prácticas en autenticación, autorización y manejo de sesiones, fortaleciendo mis conocimientos en gestión de identidades y arquitectura segura. <p>TP</p> <ul style="list-style-type: none"> - Desarrollé la <i>aplicación web completa</i>, incluyendo su <i>deploy</i> y la implementación de los <i>tres microservicios del backend</i>. 	
Reconoce la necesidad del aprendizaje permanente para el desempeño profesional y el desarrollo de proyectos en soluciones de software.	<p>Gamarra Vega, Anderson José</p> <p>TB1</p> <ul style="list-style-type: none"> - Al elaborar el Lean UX Canvas y priorizar el backlog, comprendí la importancia de actualizar de manera constante mis conocimientos en análisis de producto y planificación, reconociendo que el aprendizaje continuo es clave para mejorar mi desempeño profesional. <p>TB2</p> <ul style="list-style-type: none"> - Al definir tácticas y drivers de arquitectura en el Capítulo IV, reforcé la importancia de seguir aprendiendo para tomar mejores decisiones de diseño. <p>TB3</p> <ul style="list-style-type: none"> - Al optimizar procesos de <i>dockerización y deploy</i>, comprendí que la actualización constante en herramientas de automatización y despliegue es esencial para mantener entornos productivos eficientes y seguros. <p>TP</p> <ul style="list-style-type: none"> - Al realizar el <i>deploy, dockerización y la landing page</i>, 	<p>TB1</p> <p>El equipo reconoció que el aprendizaje permanente es un pilar para el crecimiento profesional y para la mejora continua en el desarrollo de proyectos de software. La experiencia les permitió reafirmar la importancia de seguir investigando, aplicando nuevas técnicas y adaptando buenas prácticas a futuros retos.</p> <p>TB2</p> <p>Reconocemos que las arquitecturas de software están en constante evolución, por lo que es esencial mantener un aprendizaje continuo para asegurar la calidad de soluciones de software, así como en nuestro crecimiento profesional.</p> <p>TB3</p> <p>El equipo reforzó su compromiso con la actualización continua, aplicando nuevos conocimientos en comunicación entre servicios, seguridad, despliegue automatizado y diseño web. Esta etapa consolidó la madurez técnica y la capacidad de adaptación de cada integrante frente a nuevas tecnologías y desafíos.</p>

Criterio específico	Acciones realizadas	Conclusiones TP
	<p>comprendí que el aprendizaje continuo es esencial para mantenerme actualizado en tecnologías de despliegue y entornos productivos.</p> <p>Nanfuñay Liza, Pedro Jesús</p> <p>TB1</p> <ul style="list-style-type: none"> - Al redactar este informe comprendí la importancia de actualizar constantemente los conocimientos que adquiero, ya que es un aspecto fundamental para mejorar mis capacidades, desempeño profesional y en el desarrollo de proyectos de soluciones de software. <p>TB2</p> <ul style="list-style-type: none"> - Al seleccionar estilos y patrones arquitectónicos, confirmé la necesidad de aprendizaje continuo para identificar buenas prácticas de arquitectura. <p>TB3</p> <ul style="list-style-type: none"> - Al implementar el módulo de <i>mensajes</i>, comprendí la importancia de seguir aprendiendo sobre comunicación entre servicios y manejo eficiente de datos en tiempo real. <p>TP</p> <ul style="list-style-type: none"> - Durante la ejecución de <i>testing</i> y <i>pruebas unitarias</i>, reafirmé la importancia del aprendizaje constante en técnicas de aseguramiento de la calidad de software. 	<p>En esta fase final, el equipo reafirmó la importancia de continuar aprendiendo sobre herramientas de despliegue, pruebas, automatización y buenas prácticas de desarrollo. El aprendizaje permanente permitió al grupo culminar el proyecto con una solución funcional, escalable y técnicamente sólida.</p>
	<p>Quijandria Araneda, Vicente</p> <p>TB1</p> <ul style="list-style-type: none"> - Al redactar los antecedentes y problemática y participar en la definición de historias de usuario, entendí que debo mantenerme actualizado en técnicas de análisis y documentación de requisitos para elevar mi desempeño en futuros proyectos. <p>TB2</p> <ul style="list-style-type: none"> - Al elaborar diagramas de contexto y vistas arquitectónicas, reconocí que el aprendizaje permanente es esencial para mejorar mis capacidades de diseño. <p>TB3</p> <ul style="list-style-type: none"> - Al desarrollar el módulo de <i>reportes</i>, reafirmé la necesidad de seguir aprendiendo sobre análisis de datos, optimización de consultas y visualización de información. <p>TP</p> <ul style="list-style-type: none"> - Al ejecutar <i>testing</i>, <i>pruebas unitarias</i> y participar en la gestión de <i>code guidelines</i>, comprendí la importancia de la actualización constante en herramientas de QA y control de calidad. 	
	<p>Hallasi Saravia, Miguel Ángel</p> <p>TB1</p> <ul style="list-style-type: none"> - Al elaborar los segmentos objetivo, realizar entrevistas y desarrollar el Impact Map, comprendí que el aprendizaje permanente es esencial para perfeccionar mis competencias en validación de usuarios y métricas de impacto. <p>TB2</p> <ul style="list-style-type: none"> - Al documentar escenarios de atributos de calidad y diagramas de arquitectura, confirmé la necesidad de seguir aprendiendo para abordar sistemas complejos. <p>TB3</p> <ul style="list-style-type: none"> - Al diseñar la <i>landing page</i> final, entendí que debo continuar aprendiendo sobre usabilidad, accesibilidad y nuevas tendencias de diseño web para mejorar la experiencia del usuario. <p>TP</p> <ul style="list-style-type: none"> - Al ejecutar <i>testing</i>, <i>pruebas unitarias</i> y participar en la gestión de <i>code guidelines</i>, comprendí la importancia de la actualización constante en herramientas de QA y 	

Criterio específico	Acciones realizadas	Conclusiones
	control de calidad.	
	Valera Garcés, Samuel Ignacio	
TB1	- Al redactar la descripción de la startup y los escenarios As-Is/To-Be, entendí la necesidad de fortalecer constantemente mis conocimientos en mapeo de procesos y comunicación de resultados, reconociendo que el aprendizaje continuo es esencial para mi desarrollo profesional.	
TB2	- Al diseñar la base de datos y dirigir las ADD Iterations, reafirmé la importancia del aprendizaje constante para optimizar la arquitectura de software.	
TB3	- Al modificar los módulos <i>IAM</i> y <i>Ride</i> , reforcé mi compromiso con el aprendizaje continuo en seguridad, autenticación y optimización de servicios backend.	
TP	- Al desarrollar la <i>aplicación web</i> y los <i>microservicios backend</i> con su <i>deploy</i> , reforcé mis conocimientos en despliegue y orquestación de servicios, reconociendo la necesidad de mantener un aprendizaje continuo en tecnologías modernas de desarrollo web.	

Index

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
- 2.2. 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

Capítulo III: Requirements Specification

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

Capítulo IV: Product Architecture Design

- 4.1. Design Concepts, ViewPoints & ER Diagrams
 - 4.1.1. Principles Statements
 - 4.1.2. Approaches Statements, Architectural Styles & Patterns
 - 4.1.3. Context Diagram
 - 4.1.4. Approach-Driven ViewPoints Diagrams
 - 4.1.5. Relational/Non-Relational Database Diagram
 - 4.1.6. Design Patterns
 - 4.1.7. Tactics

- 4.1.8. Design Purpose
- 4.1.9. Primary Functionality (Primary User Stories)
- 4.1.10. Quality Attribute Scenarios
- 4.1.11. Constraints
- 4.1.12. Architectural Concerns
- 4.2. Architectural Drivers
- 4.3. ADD Iterations (Attribute-Driven Design)
 - 4.3.1. Iteration N:
 - 4.3.1.1. Architectural Design Backlog N
 - 4.3.1.2. Establish Iteration Goal by Selecting Drivers
 - 4.3.1.3. Choose System Elements to Refine
 - 4.3.1.4. Choose Design Concepts to Satisfy Drivers
 - 4.3.1.5. Instantiate Architectural Elements, Allocate Responsibilities & Define Interfaces
 - 4.3.1.6. Sketch Views (C4 & UML) and Record Design Decisions
 - 4.3.1.7. Analyze Current Design and Review Iteration Goal (Kanban Board)

Capítulo V: Product Implementation, Validation & Deployment

- 5.1. Testing Suites & General Patterns
 - 5.1.1. Backend Application Core Testing Suite
 - 5.1.2. Pattern-Based Backend Applications
 - 5.1.3. Pattern-Based Custom Software Library
 - 5.1.4. Framework Pattern-Driven Refactoring Report
- 5.2. Software Configuration Management
 - 5.2.1. Software Development Environment Configuration
 - 5.2.2. Source Code Management
 - 5.2.3. Source Code Style Guide & Conventions
 - 5.2.4. Software Deployment Configuration
- 5.3. Microservices Implementation
 - 5.3.1. Sprint 1:
 - 5.3.1.1. Sprint Backlog 1
 - 5.3.1.2. Development Evidence for Sprint Review
 - 5.3.1.3. Testing Suite Evidence for Sprint Review
 - 5.3.1.4. Execution Evidence for Sprint Review
 - 5.3.1.5. Microservices Documentation Evidence for Sprint Review
 - 5.3.1.6. Software Deployment Evidence for Sprint Review
 - 5.3.1.7. Team Collaboration Insights during Sprint
 - 5.3.1.8. Kanban Board

Capítulo I: Introducción

1.1. Startup Profile

En esta sección se describen los detalles del problema que buscamos resolver.

Se detalla el perfil de la startup, el mercado objetivo, y se presentan la misión y visión.

Asimismo, se expone una visión atractiva del equipo, resaltando su potencial en función de las habilidades y capacidades de cada integrante.

1.1.1. Descripción de la Startup

TinkuyTech es una startup peruana dedicada a fortalecer la seguridad en el transporte de los estudiantes. Nuestro producto principal, ÑanGo, es una aplicación web diseñada específicamente para la comunidad universitaria. A través de esta plataforma, los estudiantes que cuentan con movilidad propia pueden conectarse con quienes necesitan transporte hacia la universidad, facilitando la coordinación de rutas, el reparto de gastos y la optimización del tiempo de viaje. De esta forma, promovemos una alternativa más segura, accesible y eficiente para la vida estudiantil.

La aplicación ofrece herramientas interactivas que permiten visualizar rutas compartidas, coordinar horarios, gestionar costos y acceder a los perfiles de conductores y pasajeros, generando confianza y transparencia. Al integrar estas funciones, ÑanGo se convierte en una solución práctica y confiable para estudiantes y familiares que priorizan seguridad y comodidad en sus traslados diarios.

Misión

Nuestra misión es transformar la movilidad estudiantil en el Perú mediante una solución tecnológica que facilite el acceso a un transporte compartido, seguro, económico y solidario. Queremos impulsar la seguridad de los estudiantes y contribuir a un futuro más sostenible, conectado y confiable.

Visión

Nuestra visión es consolidarnos como la startup referente en seguridad de transporte universitario en el Perú. Aspiramos a construir una comunidad donde los estudiantes puedan acceder de manera sencilla a un sistema de movilidad eficiente, confiable y colaborativo.

1.1.2. Perfiles de integrantes del equipo

- **Anderson Jose Gamarra Vega:**

Descripción:

Mi nombre es Anderson Jose Gamarra Vega, tengo 25 años , estudiante de Ingeniería de Software. Desde siempre he sentido una gran pasión por la tecnología, pero me decanto especialmente por el desarrollo de software, pues me fascina aprender nuevos lenguajes de programación, diseñar soluciones digitales y afrontar retos mediante código. Esta inclinación hacia el software fue lo que me motivó a elegir esta carrera, y actualmente estoy profundizando en áreas como backend, arquitectura de software, metodologías agile.



- **Pedro Jesús Nanfuñay Liza:**

Descripción:

Mi nombre es Pedro Jesús Nanfuñay Liza, tengo 20 años y soy estudiante de la carrera de Ingeniería de Software. Me considero una persona creativa, responsable, perseverante y siempre dispuesto a trabajar en equipo. Tengo conocimientos en varios lenguajes de programación como C++, Java y Python; en el desarrollo web

con frameworks Angular y Primevue, y en base de datos relacionales y no relacionales como SQL y MongoDB. Espero aportar de manera positiva al equipo y cumplir con los objetivos establecidos.



- **Vicente Quijandria Araneda:**

Descripción:

Mi nombre es Vicente Quijandria Araneda, estudio la carrera de Ingeniería de Software en la UPC. Me gusta el fútbol, surf, buceo y la tecnología. Tengo conocimientos en lenguajes de programación como Java y Python; en el desarrollo web con frameworks React y Angular, y en base de datos relacionales y no relacionales como SQL y MongoDB. Soy una persona que siempre busca aprender nuevos conocimientos que me lleven a convertirme en un gran profesional y así poder cumplir mis metas cada día.



- **Samuel Ignacio Valera Garces:**

Descripción:

Hola, mi nombre es Samuel, estudiante de la carrera de Ingeniería de software. Me considero una persona responsable, empática y con adaptación rápida al trabajo en equipo. Cuento con conocimiento en diversos lenguajes de programación. Mi objetivo a futuro es utilizar la tecnología para el desarrollo de aplicaciones que necesite la sociedad en el día a día. En mis tiempos libres, me gusta practicar guitarra y leer.



- **Miguel Hallasi Saravia:**

Descripción:

Soy Miguel Hallasi, estudiante del sexto ciclo de la carrera de Ingeniería de Software. Me gusta el aprendizaje continuo y adquirir nuevas experiencias.

**1.2. Solution Profile**

En esta sección se detallan los segmentos de descripción de nuestra solución de software, sus características de valor y las estrategias de monetización.

Product Name:

Optamos por el nombre Ñango porque proviene de la palabra quechua "ñan", que significa camino, y "go", que remite a la acción de ir o avanzar. Esta combinación transmite la esencia de la aplicación: facilitar el desplazamiento de los estudiantes de manera segura, práctica y colaborativa. El nombre refleja cercanía con la cultura peruana y, al mismo tiempo, proyecta modernidad y dinamismo, valores que nuestra plataforma busca ofrecer a sus usuarios.

1.2.1 Antecedentes y problemática

En los últimos años, la situación del transporte en el Perú se ha visto marcada por altos niveles de inseguridad que afectan tanto a pasajeros como a conductores. Este contexto resulta especialmente delicado para los estudiantes universitarios, quienes suelen ser víctimas de robos y asaltos en los alrededores de sus centros de estudio. Asimismo, la presencia de transporte informal y la práctica de cobro de "cupos" a choferes limitan la disponibilidad de servicios confiables, generando incertidumbre y restringiendo la movilidad diaria.

Frente a esta realidad, surge la necesidad de una alternativa tecnológica que garantice un traslado más seguro y accesible para los estudiantes. Es en este escenario que se plantea ÑanGo, una plataforma de movilidad compartida enfocada en la comunidad universitaria, la cual busca conectar a estudiantes con vehículo propio con aquellos que requieren transporte. Con esta solución, se busca optimizar costos, reducir riesgos asociados a la inseguridad y generar confianza a través de perfiles verificados y medidas de protección.

Para analizar la problemática, se empleó la técnica de las 5 "W" y 2 "H":

WHAT – ¿Cuál es el problema? Los servicios de transporte actuales presentan deficiencias en términos de seguridad y confianza. Los estudiantes se enfrentan a delitos frecuentes como robos, asaltos y acoso, lo que genera un ambiente de vulnerabilidad. Existe una carencia de opciones que combinen economía y seguridad en los traslados hacia las universidades.

WHEN – ¿Cuándo ocurre el problema? La problemática se presenta de forma cotidiana, sobre todo en horas punta, coincidiendo con los horarios de ingreso y salida de las universidades. Estos momentos de congestión son aprovechados por delincuentes en zonas cercanas a los centros educativos.

WHERE – ¿Dónde surge el problema y dónde se usa el producto? El problema es más visible en las calles de Lima Metropolitana y otras ciudades universitarias donde predominan los taxis informales y la inseguridad. Los estudiantes podrán usar ÑanGo desde sus casas, universidades u otros espacios con conexión a internet, para coordinar viajes hacia sus centros de estudio y compartir rutas con compañeros de confianza.

WHO – ¿Quiénes están involucrados? Los principales afectados son los estudiantes universitarios, aunque también se incluye a familiares que pueden actuar como choferes de confianza. Los usuarios de la plataforma serán tanto pasajeros que buscan un transporte seguro, como estudiantes que ofrecen sus vehículos para compartir gastos y mejorar la movilidad.

WHY – ¿Por qué ocurre el problema? Las causas principales son la informalidad en el servicio de transporte, la falta de regulación eficiente y la limitada presencia de sistemas tecnológicos que garanticen la seguridad de los pasajeros. Además, los altos costos del transporte privado llevan a los estudiantes a optar por alternativas más riesgosas.

HOW – ¿Cómo se utiliza la solución? ÑanGo será utilizada en situaciones de alta demanda de transporte, especialmente en horas punta. Los estudiantes coordinarán viajes compartidos, reducirán gastos al dividir costos y contarán con perfiles verificados que les brinden mayor confianza y seguridad en el servicio.

HOW MUCH – Sustento estadístico De acuerdo con datos de la Policía Nacional del Perú y reportes del Ministerio del Interior, Lima Metropolitana registra elevados índices de robos y asaltos en el transporte público e informal. Estas cifras evidencian la urgencia de implementar soluciones seguras e innovadoras para la comunidad estudiantil, reforzando la necesidad de una plataforma como ÑanGo.

1.2.2 Lean UX Process.

1.2.2.1. Lean UX Problem Statements

Nuestro servicio de movilidad compartida para estudiantes universitarios fue diseñado para garantizar un desplazamiento que garantice la seguridad, accesibilidad y economía de los estudiantes.

Hemos observado que el sistema de transporte público e informal actual no cumple con estos objetivos, lo que está causando altos niveles de inseguridad y sentimientos negativos como temor y estrés en los estudiantes universitarios, quienes se ven expuestos diariamente a robos, asaltos y extorsiones.

Actualmente, muchos estudiantes optan por alternativas riesgosas o costosas, lo que incrementa su exposición a la delincuencia, reduce su libertad de movilización y afecta negativamente su calidad de vida y rendimiento académico.

¿Cómo podríamos mejorar nuestro servicio de movilidad compartida para que los estudiantes universitarios tengan una solución efectiva para conectar con compañeros verificados y optimizar sus rutas diarias garantizando la confianza y la protección en sus desplazamientos?

1.2.2.2. Lean UX Assumptions

Business Assumptions:

1. Creemos que nuestros clientes necesitan una aplicación que les permita movilizarse de forma segura, confiable y económica, compartiendo viajes con otros estudiantes verificados.
2. Estas necesidades pueden cubrirse mediante una aplicación que integre validación de usuarios, rutas seguras, y un sistema de reparto de gastos entre pasajeros.
3. Nuestros clientes iniciales serán estudiantes universitarios que cuenten o no con vehículo propio, y que necesiten transportarse de manera segura.
4. El valor más importante que se espera de nuestro servicio es la seguridad durante los traslados, seguida de la accesibilidad económica y confianza entre usuarios.

5. Nuestros clientes también pueden obtener beneficios adicionales como la reducción de costos al compartir viajes, y mayor confianza y tranquilidad al trasladarse con compañeros verificados y seguimiento de su trayecto en tiempo real.
6. Planeamos adquirir la mayoría de nuestros usuarios mediante campañas en redes sociales y eventos académicos.
7. Nuestra competencia directa son los servicios de taxi tradicionales y aplicaciones de movilidad, pero estas no garantizan seguridad ni conexión exclusiva entre estudiantes.
8. Superaremos a la competencia ofreciendo una comunidad cerrada de usuarios verificados, opciones de viaje compartido y evaluación de rutas seguras.
9. El mayor riesgo de nuestro producto es la desconfianza inicial en el uso compartido del transporte con desconocidos.
10. Este riesgo será mitigado mediante perfiles verificados, calificaciones por viaje y soporte de emergencia en la aplicación.
11. La confianza en la plataforma aumentará con el tiempo gracias a la validación de usuarios y experiencias positivas.
12. La satisfacción del cliente también se podrá mejorar mediante la integración de un sistema de soporte.

User Assumptions:

1. ¿Quién es el usuario?

Estudiantes universitarios (como pasajeros o conductores) que necesitan trasladarse hacia sus instituciones educativas de forma segura y económica.

2. ¿Dónde encaja nuestro producto en su vida?

En su rutina diaria, cuando deben movilizarse a las clases universitarias, especialmente en horarios de alto riesgo.

3. ¿Qué problemas tiene nuestro producto y cómo se pueden resolver?

Al tratarse de un nuevo sistema, los usuarios podrían tener dudas sobre la seguridad y confiabilidad del servicio. Para resolverlo, se harán campañas informativas, se promoverán testimonios, y se establecerán protocolos de seguridad visibles desde el primer uso.

4. ¿Cuándo y cómo es usado nuestro producto?

NanGo será utilizado antes de clases para coordinar viajes, compartir rutas y gastos. La plataforma permitirá programar viajes con antelación y coordinar trayectos similares con compañeros.

5. ¿Qué características son importantes?

Verificación de identidad, calificación de usuarios, sistema de pago compartido, coordinar horarios de viajes y rutas seguras, y registro de detalle del trayecto.

6. ¿Cómo debe verse nuestro producto y cómo debe comportarse?

Debe tener una interfaz juvenil, amigable y confiable, con diseño intuitivo y botones de acción rápida. Debe responder con fluidez y ofrecer información clara sobre seguridad y confiabilidad del viaje.

Features:

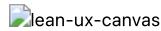
- Visualización de rutas y horarios disponibles para facilitar la planificación de los viajes.
- Notificaciones instantáneas sobre llegada, cancelaciones o cambios.
- Sistema de verificación.
- Administración eficiente de vehículos.
- Chat interno entre usuarios verificados para coordinar viajes.
- Calificaciones y comentarios entre pasajeros y conductores.
- Historial de viajes y registro de gastos compartidos.
- Soporte técnico constante y actualizaciones de sistema para mejorar el rendimiento y la seguridad.

1.2.2.3. Lean UX Hypothesis Statements

- Creemos que los estudiantes podrán optimizar su tiempo y reducir costos de transporte si se implementa una funcionalidad que les permita coordinar viajes en grupo hacia la universidad a través de una plataforma que les conecte con otros estudiantes de su misma universidad. Sabremos que hemos tenido éxito cuando el 30% de los estudiantes utilicen regularmente la plataforma para coordinar viajes y que el 25% reporte una disminución en sus gastos de transporte mensual.
- Creemos que los estudiantes reducirán su huella de carbono y contribuirán a la sostenibilidad si proporcionamos una opción fácil de coordinar viajes en vehículos compartidos, disminuyendo el número de autos individuales en las rutas comunes hacia las universidades. Sabremos que hemos tenido éxito cuando el 20% de los estudiantes reporten que han preferido la opción de compartir vehículo por encima de usar transporte público o privado.
- Creemos que los estudiantes podrán mejorar la accesibilidad y seguridad en sus traslados si ofrecemos un sistema de calificación y seguimiento de viajes compartidos, lo que proporcionará mayor confianza y control. Sabremos que hemos tenido éxito cuando al menos el 15% de los usuarios reporten mayor confianza y comodidad al utilizar la plataforma, y que el 20% mencionen que el sistema de calificación ha mejorado su experiencia de viaje.
- Creemos que los estudiantes estarán más dispuestos a utilizar la plataforma si esta ofrece perfiles verificados y filtros de coincidencia por universidad o facultad. Sabremos que hemos tenido éxito cuando al menos el 60% de los usuarios activen la verificación de identidad y el 70% prefiera viajar con contactos verificados de su misma institución.
- Creemos que la adopción aumentará si se integra un sistema de notificaciones para viajes programados. Sabremos que hemos tenido éxito cuando el 80% de los trayectos se confirmen con más de 6 horas de anticipación y se reduzca en un 30% la tasa de cancelaciones de último minuto.

- Creemos que la plataforma será más atractiva si permite compartir gastos automáticamente mediante pagos digitales entre los pasajeros. Sabremos que hemos tenido éxito cuando al menos el 50% de los viajes registrados utilicen el sistema de pago integrado y el 85% de los usuarios lo califiquen como fácil y seguro.

1.2.2.4. Lean UX Canvas



1.3. Segmentos Objetivo

Segmento 1: Estudiantes Conductores

Este segmento está compuesto por estudiantes universitarios que poseen un vehículo y buscan optimizar sus gastos de transporte.

- Características Demográficas:**

- Edad:** 18 años a más.
- Ubicación:** Zonas urbanas de Perú, principalmente Lima.
- Nivel Educativo:** Estudiantes universitarios.

Segmento 2: Estudiantes Pasajeros

Son estudiantes que no tienen un vehículo propio y necesitan una alternativa de transporte segura y económica para sus traslados a la universidad.

- Características Demográficas:**

- Edad:** 16 años a más.
- Ubicación:** Zonas urbanas de Perú.
- Nivel Educativo:** Estudiantes universitarios y de institutos superiores.

Capítulo II: Requirements Elicitation & Analysis

2.1. Competidores

Competitive Analysis Landscape

¿Por qué llevar a cabo este análisis?	Escriba en el recuadro la pregunta que busca responder o el objetivo de este análisis.				
	Para poder comprender mejor el panorama competitivo en el mercado de servicios de transporte para alumnos en Lima, Perú. Identificando fortalezas, debilidades, oportunidades y amenazas de mi startup y sus competidores.				
Nuestro Producto / Competidores	ÑanGo	GoLadies	Hoop carpool	BlaBlaCar	
Perfil	Overview	Aplicación web diseñada específicamente para los estudiantes. ÑanGo conecta a estudiantes que cuentan con movilidad propia con aquellos que buscan transporte para ir a la universidad.	Go Ladies Perú se dedica a realizar traslados exclusivos a mujeres, niños, personas con discapacidad certificada y adultos mayores con identificación.	Hoop Carpool es una empresa de carpooling para empresas. Ofrece a tus empleados y alumnos la opción de compartir el coche en su día a día.	BlaBlaCar es una comunidad de usuarios basada en la confianza que conecta a conductores con asientos vacíos y pasajeros que se dirigen a un mismo lugar, para que viajen juntos y compartan el costo.
	Ventaja competitiva	Movilización de forma segura, confiable y económica además de compartir viajes solo con estudiantes verificados.	Transporte exclusivo de mujeres para mujeres, brindando seguridad y confianza en las usuarias.	Enfoque más orientado a empresas como también hacer un impacto al tratar de reducir el uso de vehículos.	Brinda viajes a precios bajos, reconocimiento mediante identificaciones y reservas fáciles
Perfil de Marketing	Mercado objetivo	Estudiantes de Universidades en todo Lima.	Todas las mujeres de Lima.	Personal empresarial.	Todo público en general.
	Estrategias de marketing	Promoción y publicidad en redes sociales y anuncios	Promoción en redes sociales, anuncios y colaboraciones con influencers.	Publicidad en redes sociales y anuncios.	Publicidad en páginas web.
Perfil de Producto	Productos & Servicios	Brinda servicio de transporte a estudiantes, utilizando vehículos personales.	Brinda servicio de transporte de solo mujeres para mujeres.	Brinda servicio de transporte compartido a personal empresarial.	Brinda viajes en auto personal y bus compartido a todo público.
	Precios & Costos	Tarifa sujeta al conductor con previa coordinación.	Tarifas sujetas al trayecto.	Tarifa sujeta a contrato	Tarifa sujeta al trayecto.
Canales de	Aplicación web.	Sitio web.	Aplicación móvil y sitio	Aplicación móvil y sitio web.	

	distribución (Web y/o Móvil)		web.	
Análisis SWOT	Fortalezas	Seguridad y rapidez al brindar servicio de transportes a estudiantes.	Servicio de transporte solo para mujeres.	Especializado en servicios de transporte empresarial.
	Debilidades	Dependencia de la disponibilidad de los usuarios.	Poca disponibilidad en horario nocturno.	Abarca solo empresas y costos altos
	Oportunidades	Mejora la seguridad del transporte de los estudiantes.	Mejora la seguridad para las mujeres al transportarse en taxi.	Reducción de autos del personal empresarial.
	Amenazas	Competencia con servicios de taxi en general.	Competencia con servicios de taxi en general.	Cantidad mínima de contratos.
				Servicio de transporte para todo público
				Poca publicidad y costos altos.
				Mejora la seguridad y la confianza al tomar un transporte de viaje extenso.
				Competencia de otros servicios similares.

2.1.2 Estrategia y tácticas frente a competidores

1. Servicio especializado en la comunidad universitaria:

- Exclusividad y confianza:** A diferencia de apps masivas como BlaBlaCar, en ÑanGo solo viajan con otros estudiantes. La verificación con correo institucional crea un círculo de confianza que nos diferencia.

2. Precios pensados para estudiantes:

- Costos compartidos, no tarifas:** Promovemos un modelo donde los conductores comparten gastos, no buscan lucrar. Esto nos hace más económicos que las alternativas.
- Flexibilidad:** Ofreceremos descuentos para grupos y precios más bajos en horas de poca demanda para adaptarnos al bolsillo del estudiante.

3. Una red de confianza, no solo de viajes:

- La confianza es nuestro pilar:** Nuestra ventaja frente a los taxis es la seguridad. Perfiles completos, reseñas entre compañeros y la opción de compartir tu ruta en tiempo real son la base.
- Fidelización y comunidad:** Recompensaremos a los mejores conductores y pasajeros, y organizaremos eventos para fortalecer los lazos entre usuarios.

2.2. Entrevistas

2.2.1 Diseño de Entrevistas

Objetivos

Recoger información sobre las necesidades, expectativas y posibles preocupaciones de los estudiantes y familiares que estarían interesados de ofrecer servicios de transporte (como un taxi compartido).

Preguntas

Estudiante con vehículo

Datos básicos

- ¿Nombre, Edad y carrera?
- ¿Conduces a la universidad? ¿Con qué frecuencia?

Transporte compartido

- ¿Has pensado en llevar a otros estudiantes?
- ¿Te incomodaría compartir tu auto? ¿Por qué?

Sobre la app

- ¿Qué funciones te parecen esenciales?
- ¿Qué medidas de seguridad te darían confianza?
- ¿Aceptarías compartir tu ruta o horario?
- ¿Preferirías elegir a los pasajeros o que sea automático?
- ¿Qué tipo de pago prefieres?

Cierre

- ¿Probarías la app cuando esté disponible?
- ¿Te gustaría participar en futuras pruebas?

Estudiante sin vehículo

Datos básicos

- ¿Edad y carrera?
- ¿Cómo llegas a la universidad normalmente?
- ¿Cuánto te toma el trayecto?

Interés en el servicio

- ¿Te interesaría usar un servicio de transporte compartido con otros estudiantes?
- ¿Qué te haría sentir más seguro al usarlo?

Sobre la app

- ¿Qué funciones te gustaría que tenga?
- ¿Qué tipo de pago preferirías?
- ¿Preferirías elegir al conductor o que sea automático?

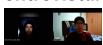
Cierre

- ¿Probarías la app cuando este disponible?
- ¿Te gustaría participar en futuras encuestas o pruebas?

2.2.2 Registro de Entrevistas**Entrevistas para el Segmento Objetivo 1 - Estudiante con vehículo:**

Entrevista	2	Nombre	Ariana Martinez
Edad	24	Distrito	Santiago de Surco, Lima
Captura de la entrevista: 	Ariana es una estudiante universitaria de la UPC, propietaria de un vehículo privado, que se desplaza frecuentemente desde su casa hasta la universidad. Ella considera que compartir su vehículo con otros estudiantes es una buena idea para reducir los costos de transporte y hacer nuevas amistades. Sin embargo, tiene preocupaciones sobre la seguridad personal y la verificación de identidad, por lo que considera importante que una plataforma como la nuestra presente medidas de seguridad estrictas que verifiquen la identidad de cada pasajero y conductor.		
URL de la grabación	Link a la entrevista		
Timing	00:00 – fin		

Entrevistas para el Segmento Objetivo 2 - Estudiante sin vehículo:

Entrevista	1	Nombre	Álvaro Urbina
Edad	21	Distrito	Comas
Captura de la entrevista: 	Álvaro Urbina es un estudiante universitario de 21 años que estudia la carrera de Ingeniería Ambiental en la Universidad Nacional Tecnológica del Sur. Él transporta informales y formales con un tiempo aproximado de 3 horas. Considera que una aplicación con servicio de transporte compartido con otros conocer a más estudiantes. Además, sugiere la implementación de funciones de seguridad para poder compartir su ubicación y ruta de viaje asignado a pagos por servicios como Yape por su practicidad; y prefiere que el sistema elija a los conductores automáticamente para conocer nuevas personas. El compartido que ofrece Ñango para transportarse a su centro de estudios y está dispuesto a colaborar en futuras pruebas y encuestas de la misma.		
URL de la grabación	https://upcedupe-my.sharepoint.com/:v/g/personal/u202215462_upc_edu_pe/EYC-VVwDoY9Orw2tatR9QxQB11ebyRm_vkseBp2bTEIU-w-e=X5KQgN&nnav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAIoiJTdHJYIW1XZWJBcHAIcJyZWZlcnJhbFZpZXciOijTaGFyZURpYWxvZy1MaW5rl		
Timing	00:00 - 3:55		

2.2.3. Análisis de entrevistas**Segmento objetivo: estudiante con vehículo**

Los estudiantes con auto propio consideraron que una plataforma para compartir su vehículo con otros compañeros sería beneficiosa tanto para **reducir costos de transporte** como para **aprovechar mejor el tiempo** y conocer nuevas personas.

Destacaron que la **seguridad** es un aspecto clave, valorando funciones como la **geolocalización en tiempo real** y la **validación de perfiles**, ya que brindan confianza al ofrecer cupos a otros usuarios.

También mencionaron que una **compensación justa por viaje** y una **interfaz sencilla** serían elementos importantes para motivar su participación como conductores.

Segmento objetivo: estudiante sin vehículo

Los estudiantes sin vehículo propio resaltaron la importancia de contar con una aplicación que permita **seguir la ubicación del vehículo en tiempo real** y **conocer la calificación de los conductores**, lo que les genera mayor confianza y sensación de seguridad.

Prefieren **métodos de pago digitales** como **Yape o Plin** por su facilidad y rapidez, y ven el servicio como una buena alternativa para **acortar sus largos tiempos de traslado y socializar con otros estudiantes**.

En general, mostraron interés en la propuesta y disposición para participar en **pruebas piloto**, aportando así a la mejora del sistema.

2.3. Needfinding

2.3.1. User Persons

PERSONA: Ana Guevara

NAME	TYPE
Ana Guevara	Rational



Skills or Scale

Conducción

Beginner Intermediate Expert

Tecnología

Beginner Intermediate Expert

Comunicación

Beginner Intermediate Expert

Demography

Female 19 years

Lima, Perú

Single

Estudiante

Quote

Necesito saber que puedo confiar en quien me lleve a clase, y que la coordinación sea rápida

Goals

- Asegurar un viaje seguro y económico a la universidad.

Background

- Estudiante sin vehículo y con limitaciones económicas para contratar transporte privado.
- Activa en redes sociales y en grupos estudiantiles.

Frustrations

- Falta de información sobre el origen y reputación del conductor.
- Dificultades en la coordinación sin una herramienta centralizada.

Motivations

- Buscar opciones de transporte confiables y puntuales.

Expectations

- Una plataforma fácil de usar.
- Soporte técnico eficiente.

Needs

- Sistema transparente de evaluación entre usuarios.
- Calendario de rutas y notificaciones.

UXPRESSIA

This persona was built in uxpressia.com

Estudiante conductor

PERSONA: Carlos Saavedra

NAME	TYPE
Carlos Saavedra	Rational



Skills or Scale

Conducción

Beginner Intermediate Expert

Tecnología

Beginner Intermediate Expert

Comunicación

Beginner Intermediate Expert

Demography

♂ Male 21 years

Lima, Perú

Single

Estudiante

Quote

“Quiero sacarle el máximo partido a mi auto, pero sin comprometer mi seguridad y tiempo”

Goals

- Optimizar el uso de su automóvil para reducir gastos y aprovechar el trayecto.
- Contribuir a la sostenibilidad ambiental.

Background

- Estudiante universitario con vehículo propio.
- Posee un vehículo propio.

Frustrations

- Inseguridad al desconocer la reputación de los pasajeros.
- Problemas de coordinación de horarios.

Motivations

- Reducir gastos personales.
- Aprovechar el tiempo de traslado para socializar y conocer compañeros.

Expectations

- Una plataforma fácil de usar.
- Soporte técnico eficiente.

Needs

- Plataforma segura y confiable.
- Herramienta de verificación de perfiles y ratings.

UXPRESSIA
This persona was built in uxpressia.com

PERSONA: Javier Morales

NAME	TYPE
Javier Morales	Rational
	<p>Quote</p> <p>“Quiero ayudar asegurando que todos tengan un medio de transporte confiable, sin complicaciones adicionales.”</p>
Skills or Scale	<p>Goals</p> <ul style="list-style-type: none"> Compartir gastos de manera efectiva y optimizar tiempos sin afectar su agenda laboral.
Conducción	<p>Background</p> <ul style="list-style-type: none"> Padre de estudiante que decide brindar transporte ocasional a otros compañeros de sus hijos. Tiene experiencia en conducción y busca contribuir a la comunidad estudiantil.
<input type="range" value="1"/> Beginner Intermediate Expert	
Tecnología	<p>Frustrations</p> <ul style="list-style-type: none"> Poca información sobre la confiabilidad de los usuarios. Dificultades para coordinar los horarios con el ritmo universitario.
<input type="range" value="1"/> Beginner Intermediate Expert	
Comunicación	<p>Motivations</p> <ul style="list-style-type: none"> Garantizar la seguridad del trayecto para su hijo y otros estudiantes.
<input type="range" value="1"/> Beginner Intermediate Expert	
Demography	<p>Expectations</p> <ul style="list-style-type: none"> Una plataforma fácil de usar. Soporte técnico eficiente.
<input checked="" type="radio"/> Male 45 years	<p>Needs</p> <ul style="list-style-type: none"> Herramientas para comunicación directa y coordinación de horarios. Funciones de notificación y feedback para mejorar la experiencia en cada viaje.
<input checked="" type="radio"/> Lima, Perú	
Married	
Padre de familia	

UXPRESSIA

This persona was built in uxpressia.com

2.3.2. User Task Matrix

El User Task Matrix identifica las tareas que cada arquetipo debe realizar para alcanzar sus objetivos, sin confundir actividades o funcionalidades específicas de la aplicación. Se consideran la frecuencia y la importancia de cada tarea para cada User Persona.

Table

Tarea / Actividad	Estudiante-Conductor		Estudiante-Pasajero		Familiar-Conductor	
	Frec.	Importancia	Frec.	Importancia	Frec.	Importancia
Buscar rutas disponibles	Alta	Alta	Alta	Alta	Media	Media
Publicar disponibilidad de viaje	Alta	Alta	-	-	Alta	Alta
Coordinar horarios y puntos de encuentro	Alta	Alta	Alta	Alta	Alta	Alta
Confirmar reserva o viaje	Media	Alta	Alta	Alta	Media	Media
Revisar y gestionar perfiles/reseñas	Media	Alta	Alta	Alta	Alta	Alta
Compartir detalles del trayecto	Alta	Media	Media	Alta	Media	Meida

2.3.3 User Journey Mapping

2.3.4. Empathy Mapping

Estudiante Conductor:

PERSONA: Ariana Martínez

1.WHO are we empathizing with?

- Persona: Estudiante universitario con vehículo propio que está dispuesto a ofrecer servicio de movilidad.
- Situación: Busca optimizar el uso de su auto, ahorrar en gasolina y estacionamiento, y al mismo tiempo ayudar y conocer a nuevas personas.

7.What do they THINK and FEEL?

- “
- *¿Será seguro llevar a alguien que apenas conozco?*
 - *No quiero perder tiempo organizando viajes por mensajes.*
 - *Si coordino con pasajeros confiables, me ahorro gastos y viajo más tranquilo.*
- ”

2.What do they need to DO?

- Difundir su disponibilidad de asientos y rutas de manera sencilla.
- Coordinar horarios y puntos de recogida con posibles pasajeros.
- Verificar la confiabilidad de los pasajeros.

6.What do they HEAR?

- Peticiones constantes de compañeros que necesitan transporte.
- Quejas de otros conductores sobre la impuntualidad de algunos pasajeros.
- Sugerencias de amigos sobre cómo compartir gastos y establecer reglas claras.
- Opiniones de familiares preocupados por la seguridad al llevar personas poco conocidas.

**3.What do they SEE?**

- Estudiantes solicitando transporte por aplicaciones de servicios o en transporte público.
- Falta de una plataforma segura que facilite la publicación de rutas de viaje.
- Aplicaciones de movilidad masivas que no garantizan confianza entre compañeros de estudio.

5.What do they DO?

- Publican en grupos de WhatsApp, Facebook o foros internos de la universidad sobre su disponibilidad.
- Coordinan con varios interesados para coordinar horarios y lugares de encuentro.
- Reciben pagos en efectivo o a través de servicios de transferencia.

PAINS

- Inseguridad al aceptar pasajeros desconocidos.
- Dificultad para coordinar horarios y puntos de encuentro.
- Pagos informales que pueden causar conflictos o incumplimientos.
- Pérdida de tiempo gestionando viajes de forma manual.

GAINS

- Ahorro en gasolina y estacionamiento gracias a la contribución de pasajeros.
- Posibilidad de conocer más personas.
- Organización clara de horarios, rutas y pagos desde una sola plataforma.
- Sentimiento de satisfacción al apoyar a compañeros universitarios.

4.What do they SAY?

- “
- *Quiero compartir mi carro, pero no me siento seguro.*
 - *Necesito coordinar los viajes de forma más rápida y clara.*
 - *Ojalá hubiera un sistema de calificaciones para saber a quién llevo.*
 - *Quiero ahorrar en gasolina y estacionamiento sin complicaciones.*
- ”

UXPRESSIA

This persona was built in upressoia.com

Estudiante Pasajero:

PERSONA: Álvaro Urbina

1.WHO are we empathizing with?

- Persona: Estudiante universitario sin vehículo propio.
- Situación: Necesita llegar a la universidad de manera segura y económica, pero depende de terceros (conductores particulares o transporte público).

7.What do they THINK and FEEL?

- “
- *Me preocupa no llegar a tiempo a mis clases.*
 - *Deseo evitar la mayor cantidad de tráfico posible.*
 - *Me siento estresado por la inseguridad en el transporte público.*
- ”

2.What do they need to DO?

- Usar transporte público o conseguir un conductor confiable y económico que lo transporte a su centro de estudios.
- Revisar opciones en base a rutas y tiempo estimado.
- Verificar la reputación y calificación del conductor, en caso de ser un transporte privado.

6.What do they HEAR?

- Compañeros aquejados por su situación para llegar a tiempo a la universidad.
- Compañeros que están dispuestos a llevar a sus amigos en su vehículo.
- Comentarios sobre costos injustos o conductores poco confiables

**3.What do they SEE?**

- Compañeros que buscan un sistema de transporte más coordinado.
- Falta de una plataforma centralizada para conectar estudiantes que cuentan con transporte propio y aquellos que buscan este tipo de transporte.

5.What do they DO?

- Consultan con familiares y amigos si conocen a alguien con vehículo disponible.
- Utiliza el transporte público con la ruta más óptima.
- Compara costos y tiempos estimados para llegar con antelación a su centro de estudios.
- Revisa reseñas o referencias de compañeros, si existen.

PAINS

- Falta de un sistema de transporte seguro y económico
- Riesgo de llegar tarde a su centro de estudios.
- Estar en constante estrés por el tiempo invertido durante el transporte.

GAINS

- Poder compartir gastos y acceder a un servicio más económico que un taxi convencional.
- Reducir el estrés de llegar tarde si se coordina bien.
- Conocer nuevos compañeros y socializar durante el trayecto.

4.What do they SAY?

- “
- *Necesito encontrar un transporte que garantice mi seguridad y economía.*
 - *No quiero llegar tarde a mis clases por rutas poco óptimas tomadas por el transporte el cual suelo usar.*
 - *Deseo una plataforma que me permita coordinar mi transporte rápidamente.*
- ”

UXPRESSIA

This persona was built in upressoia.com

2.3.5. As-is Scenario Mapping

Estudiante pasajero

Estudiante - Pasajero

Scenario Mapping

Steps	Descubrimiento	Búsqueda de Opciones	Join	Coordinación y Reserva	Viaje	Feedback
Doing	Revisa redes sociales, recibe recomendaciones de amigos y nota la falta de transporte.	Explora grupos en WhatsApp, Facebook y foros internos para identificar posibles rutas.	Examina perfiles, comparaciones y evaluaciones, pero duda al comprometerse formalmente.	Contacta al conductor, intercambia mensajes para confirmar detalles como punto y horario.	Se reúne con el conductor en el punto acordado y realiza el trayecto hacia la universidad.	Comparte su experiencia de forma informal (chat, boca a boca o redes sociales).
	"Necesito encontrar rápidamente un transporte seguro para llegar a clases."	"Encontraré un conductor confiable en medio de tanta información dispersa?"	"Puedo confiar en ese conductor sin un sistema de verificación real?"	"Espero que la coordinación fluya y no se presenten contratiempos."	"Llegaré a tiempo y sin inconvenientes."	"Debería recomendar esta opción o advertir a otros sobre los problemas."
	Ansiedad, urgencia y cierta preocupación.	Incertidumbre combinada con esperanza.	Duda e inseguridad; existe reticencia a comprometerse.	Nerviosismo al esperar confirmación; alivio leve al concretar.	Preocupación al inicio, alivio al comprobar que el viaje transcurre bien.	Satisfacción si el viaje fue exitoso o frustración en caso contrario.

Estudiante conductor

Estudiante - Conductor

Scenario Mapping

Steps	Descubrimiento	Búsqueda de Opciones	Join	Coordinación y Reserva	Viaje	Feedback
Doing	Toma conciencia de la oportunidad al escuchar solicitudes de pasajeros y revisa la tendencia de compra vehículo.	Publica su disponibilidad en redes y grupos internos, especificando horarios y ruta.	Invita a los interesados a confirmar su participación, pero respuestas limitadas o tardías	Se comunica con los interesados, verifica datos y afina detalles de punto y hora mediante llamadas o mensajes	Conduce el trayecto siguiendo la ruta establecida, interactuando con los pasajeros en el proceso	Solicita opiniones de los pasajeros y anota sugerencias o inconvenientes experimentados
	"Puedo ahorrar y ayudar a mis compañeros, pero ¿será seguro?"	"Debo comunicar bien mi oferta para atraer a los interesados."	"Por qué es tan difícil conseguir compromisos? ¿No confían en mí?"	"Espero que todos cumplan con lo acordado para evitar retrasos."	"Espero que el viaje transcurra sin contratiempos y que todos sean puntuales."	"Con un buen feedback, podré mejorar y generar mayor confianza para futuros viajes."
	Motivación mezclada con cautela y cierto temor.	Entusiasmo, aunque con preocupación por la seguridad	Frustración e inseguridad por la falta de compromiso formal	Nerviosismo al ultimar los detalles, con alivio una vez confirmados.	Alivio al comprobar una coordinación exitosa, pero siempre con vigilancia	Satisfacción por el viaje exitoso o inquietud si surgen problemas

Capítulo III: Requirements Specification

3.1. To-be Scenario Mapping

Estudiante pasajero

Steps	Descubrimiento	Búsqueda de Opciones	Join	Coordinación y Reserva	Viaje	Feedback
Doing	Abre la app Ñango y descubre rutas sugeridas de estudiantes con auto cerca.	Filtrar opciones según horario, ruta y calificaciones de conductores.	Revisa perfiles verificados, lee reseñas y selecciona una opción con confianza.	Reserva el viaje con clic, recibe confirmación automática y accede al chat integrado.	Llega al punto de encuentro gracias al mapa, viaja con seguimiento en tiempo real.	Califica el conductor desde la app dejando un comentario visible para otros usuarios.
	"Qué útil! Hay opciones pensadas para estudiantes como yo."	"Este sistema me da confianza, veo opiniones y rutas exactas."	"Parece seguro, todos están verificados y hay calificaciones."	"Todo está claro, ya tengo confirmación y contacto directo!"	"El trayecto está siendo cómodo, puedo ver la ruta en el mapa."	"¡Buena experiencia! Voy a dejar una reseña positiva."
	Curiosidad y optimismo.	Confianza, empoderamiento.	Seguridad y tranquilidad.	Alivio, organización.	Relajación, comodidad.	Gratitud, pertenencia a una comunidad.

Estudiante conductor

Steps	Descubrimiento	Búsqueda de Opciones	Join	Coordinación y Reserva	Viaje	Feedback
Doing	Se registra en ÑanGo como conductor, recibe notificaciones de otros pasajeros que coinciden con su ruta.	Publica horarios y rutas usando formularios simples dentro de la app; el sistema optimiza coincidencias automáticamente.	Revisa perfiles verificados de estudiantes interesados y acepta solicitudes con un clic.	Confirmación automática con detalles del viaje, punto de encuentro, y contacto por chat integrado.	Sigue la ruta desde la app, con GPS activo y sistema de alertas para puntualidad.	Recibe calificaciones directamente en la app, acceso a estadísticas y reseñas de sus viajes anteriores.
Thinking	"Esta app me facilita encontrar pasajeros y me siento más seguro."	"Publicar rutas nunca fue tan fácil. La app se encarga del emparejamiento."	"Me siento respaldado. Todos están verificados y hay un sistema claro de reputación."	"No tengo que perseguir a nadie, todo quedó coordinado automáticamente."	"Estoy tranquilo porque la app me guía y los pasajeros están bien informados."	"Puedo seguir mejorando como conductor. El feedback me ayuda mucho."
Feeling	Seguridad, motivación.	Confianza, eficiencia.	Tranquilidad, respaldo.	Alivio, control.	Relajación, comodidad.	Orgullo, crecimiento.

3.2 User Stories

Epic ID	Título	Descripción
EP01	Registro de Usuario	Registro de usuarios en la plataforma
EP02	Recuperación de Cuenta	Recuperación de cuenta de usuario
EP03	Autenticación	Autenticación de cuenta para acceder a la plataforma
EP04	Gestión de Perfil	Gestión de información de cuenta del usuario
EP05	Comunicación	Funciones de comunicación entre usuarios
EP06	Seguridad	Garantiza la seguridad del usuario y su información personal
EP07	Verificación	Verificación de la cuenta mediante servicios externos
EP08	Búsqueda de Viajes	Búsqueda de conductores disponibles en el momento
EP09	Solicitud de Viaje	Solicita un viaje
EP10	Calificación Post-Viaje	Calificación del usuario tras haber culminado un viaje
EP11	Rutas de viaje en el mapa	Ruta más óptima a seguir
EP12	Historial de Viajes	Acceso al historial de viajes realizados en la plataforma
EP13	Planificación de Viajes	Planificación anticipada de viajes
EP14	Publicación de Viajes	Publicación de viajes disponibles en el momento
EP15	Gestión de Solicitudes de Viajes	Gestionar la disponibilidad e información del viaje
EP16	Reputación	Calificación promedio obtenida
EP17	Notificaciones	Alertas para mantenerse informado
EP18	Historial y Finanzas	Historial de viajes y ganancias

Epic / User Story ID	Título	Descripción	Criterios de Aceptación	Relacionado con (Epic ID)
US01	Registro de nueva cuenta	Como usuario nuevo no registrado, deseo poder crear una cuenta en la aplicación, para acceder a las funcionalidades exclusivas de ÑanGo.	<p>Scenario 1: Registro con datos válidos Dado que el usuario no está registrado, Y se encuentra en la pantalla de "registro", Cuando completa los campos requeridos [Nombres, Apellidos, Correo electrónico, Contraseña, Confirmar Contraseña] con datos válidos Y selecciona su perfil de usuario [Conductor o Pasajero], Y acepta los "términos y condiciones", Entonces el sistema crea la cuenta Y envía un link de verificación al correo.</p> <p>Scenario 2: Registro con datos inválidos o incompletos Dado que el usuario está en la pantalla de "registro", Cuando deja campos obligatorios vacíos o introduce un correo inválido, Entonces el sistema muestra un mensaje de error Y no permite completar el registro hasta corregir los datos.</p>	EP01

Epic / User Story ID	Título	Descripción	Criterios de Aceptación	Relacionado con (Epic ID)
US02	Recuperación de contraseña	Como usuario que olvidó su contraseña, quiero recuperar el acceso a mi cuenta, para poder seguir usando la app.	<p>Scenario 1: Solicitud de recuperación de contraseña Dado que el usuario no recuerda su contraseña, Y accede a la opción "¿Olvidaste tu contraseña?", Cuando tu correo esta en el apartado de "iniciar sesion", Y es un electrónico registrado, Entonces el sistema envía un enlace para restablecer la contraseña. Cuando el Usuario abra el link podra ingresar a la recuperacion de cuenta, Y completa los campos requeridos Y le da click a guardar nueva contraseña, Entonces su contraseña es actualizada por el sistema.</p> <p>Scenario 2: Recuperación con correo no registrado Dado que el usuario ha olvidado su contraseña, Y accede a la opción "¿Olvidaste tu contraseña?", Cuando introduce un correo no vinculado a ninguna cuenta, Entonces el sistema muestra un mensaje "Este correo no está registrado."</p>	EP02
US03	Inicio de sesión	Como usuario registrado, quiero iniciar sesión en la app, para acceder a mis funcionalidades personalizadas.	<p>Scenario 1: Inicio de sesión exitoso Dado que el usuario está registrado, Y se encuentra en la pantalla de "Iniciar Sesión", Cuando introduce su "correo" y "contraseña" válidos, Entonces el sistema lo autentica, Y lo redirige a su "pantalla principal".</p> <p>Scenario 2: Inicio de sesión fallido Dado que el usuario está en la pantalla de login, Cuando introduce un correo o contraseña incorrectos, Entonces el sistema muestra un mensaje de error Y no permite iniciar sesión.</p> <p>Scenario 3: Inicio de sesión exitoso y guardado de inicio Dado que el usuario está registrado, Y se encuentra en la pantalla de "Iniciar Sesión [Nueva contraseña, Confirmar Contraseña]", Cuando introduce su correo y contraseña válidos, Y marca la casilla Recuérdame, Entonces el sistema lo autentica, Y genera un token de recordatorio único y seguro, Y almacena el token asociado al usuario en la base de datos, Y lo redirige a su "pantalla principal".</p>	EP03
US04	Cierre de sesión	Como usuario autenticado, quiero cerrar sesión, para proteger el acceso a mi cuenta.	<p>Scenario 1: Logout exitoso Dado que el usuario está autenticado, Cuando selecciona su perfil, Y se despliega el menu cascada, Y selecciona la opción "Salir", Entonces el sistema cierra la sesión actual Y redirige al usuario a la pantalla de inicio.</p> <p>Scenario 2: Logout fallido por conexión Dado que el usuario está autenticado, Cuando selecciona la opción "Cerrar sesión" y no hay conexión de red, Entonces el sistema muestra un mensaje de error Y no cierra la sesión hasta que se recupere la conexión.</p>	EP03
US05	Edición de perfil	Como usuario autenticado, quiero poder editar mi perfil, para mantener mi información actualizada.	<p>Scenario 1: Edición de campos personales Dado que el usuario está autenticado, Y accede a la sección "Mi perfil", Y accede a la opcion "Editar", Cuando modifica los campos de "nombre, email, celular, Plan, o foto de perfil", Y guarda los cambios, Entonces el sistema actualiza su información correctamente.</p> <p>Scenario 2: Edición con datos inválidos Dado que el usuario está autenticado, Y accede a la sección "Mi perfil", Y accede a la opcion "Editar", Cuando introduce caracteres inválidos en el campo "nombre", O intenta subir una imagen no permitida, Entonces el sistema muestra mensajes de validación Y no guarda los cambios hasta que se corrijan.</p>	EP04
US06	Chat de comunicación	Como usuario, quiero comunicarme con otros mediante chat, para coordinar viajes y resolver dudas.	<p>Scenario 1: Envío de mensaje en chat Dado que el usuario está en una conversación activa, Cuando escribe un mensaje y pulsa "enviar", Entonces el mensaje se muestra en el chat Y es recibido por el otro usuario en tiempo real.</p> <p>Scenario 2: Fallo de red al enviar mensaje Dado que el usuario está en una conversación activa, Y hay un problema de conexión a internet, Cuando intenta enviar un mensaje, Entonces el sistema muestra un mensaje "No se pudo enviar el mensaje. Intenta nuevamente."</p>	EP05

Epic / User Story ID	Título	Descripción	Criterios de Aceptación	Relacionado con (Epic ID)
US07	Cambiar contraseña	Como usuario autenticado, quiero cambiar mi contraseña, para reforzar la seguridad de mi cuenta.	<p>Scenario 1: Cambio exitoso de contraseña Dado que el usuario está autenticado, Cuando selecciona su perfil, Y se despliega el menu cascada, Y selecciona la opción "Cambiar contraseña", Y cuando completa los campos de "contraseña" y "nueva contraseña", Y le da click al boton "Guardar nueva contraseña", Entonces el sistema lo lleva a la pagina "Cambiar contraseña", actualiza la "contraseña" Y muestra un mensaje de confirmación.</p> <p>Scenario 2: Contraseña actual incorrecta Dado que el usuario accede a "Configuración > Seguridad", Cuando introduce una contraseña actual incorrecta, Entonces el sistema muestra un mensaje de error Y no permite cambiar la contraseña.</p>	EP06
US08	Verificación con correo institucional	Como estudiante, quiero verificar mi cuenta utilizando mi "DNI" y "Carnet Universitario", para tener acceso a las funcionalidades de la aplicación.	<p>Scenario 1: Validacion con documentos válido Dado que el estudiante está registrado, Y se encuentra en la pantalla de "Verificacion", Cuando da click en las opciones de subir documento en "DNI" y "Carnet Universitario", Y sube sus documentos validos, Entonces el sistema valida su cuenta.</p> <p>Scenario 2: Validacion con documentos inválidos Dado que el estudiante está registrado, Y se encuentra en la pantalla de "Verificacion", Cuando da click en las opciones de subir documento en "DNI" y "Carnet Universitario", Y sube documentos son invalidos, Entonces el sistema valida su cuenta.</p>	EP07
US09	Búsqueda de viajes disponibles	Como estudiante sin vehículo, quiero buscar viajes disponibles, para poder unirme a ellos.	<p>Scenario 1: Búsqueda de viajes según filtro Dado que el estudiante está en la pantalla de cotizaciones, Cuando selecciona filtros como "fecha", "origen" y "destino", Entonces el sistema muestra una lista de viajes disponibles que coinciden con los filtros.</p> <p>Scenario 2: Sin resultados de búsqueda Dado que el estudiante está en la pantalla de cotizaciones, Cuando no hay viajes disponibles para los filtros seleccionados, Entonces el sistema muestra un mensaje "No se encontraron viajes disponibles."</p>	EP08
US10	Solicitud de unirse a un viaje	Como estudiante, quiero solicitar unirme a un viaje disponible, para poder participar en el transporte compartido.	<p>Scenario 1: Solicitud de unirse a un viaje disponible Dado que el estudiante ha encontrado un viaje disponible, Cuando selecciona la opción "Unirse al viaje", Y le aparece la ventana para reservar Y selecciona la opción en el apartado de "pago", Y selecciona la opción "Solicitar", Entonces el sistema envía la solicitud al conductor.</p> <p>Scenario 2: Solicitud de unirse a un viaje lleno Dado que el estudiante ha encontrado un viaje, Cuando intenta unirse a un viaje que ya está lleno, Entonces el sistema muestra el mensaje "Este viaje ya está completo."</p>	EP09
US11	Notificaciones en tiempo real	Como estudiante, quiero recibir notificaciones en tiempo real sobre el estado de mi solicitud de viaje, para estar informado de cualquier cambio.	<p>Scenario 1: Notificación de aceptación de solicitud Dado que el estudiante ha solicitado unirse a un viaje, Cuando el conductor acepta la solicitud, Entonces el sistema envía una notificación al estudiante con la confirmación.</p> <p>Scenario 2: Notificación de rechazo de solicitud Dado que el estudiante ha solicitado unirse a un viaje, Cuando el conductor rechaza la solicitud, Entonces el sistema envía una notificación al estudiante con el mensaje "Solicitud rechazada."</p>	EP05
US12	Sistema de calificación post-viaje	Como estudiante, quiero calificar el viaje después de completarlo, para evaluar el servicio y ayudar a mejorar la calidad de los viajes.	<p>Scenario 1: Calificación exitosa del viaje Dado que el viaje ha finalizado, Cuando el estudiante accede a la sección de "calificación", Y selecciona una "calificación" y "comentarios", Entonces el sistema guarda la calificación y muestra un mensaje de confirmación.</p> <p>Scenario 2: Calificación incompleta Dado que el estudiante ha finalizado el viaje, Cuando intenta calificar sin proporcionar "comentarios" o "calificación", Entonces el sistema muestra un mensaje "Por favor, proporciona una calificación para continuar."</p>	EP10

Epic / User Story ID	Título	Descripción	Criterios de Aceptación	Relacionado con (Epic ID)
US13	Chat interno con Grupo	Como estudiante, quiero poder comunicarme con el conductor a través del chat interno una vez que mi solicitud de viaje haya sido aceptada.	<p>Scenario 1: Enviar mensaje al conductor después de aceptación Dado que el estudiante ha sido aceptado en un viaje, Cuando el estudiante envía un mensaje grupo, Entonces el mensaje aparece en el chat y es recibido por el grupo en tiempo real.</p> <p>Scenario 2: No poder enviar mensaje antes de aceptación Dado que el estudiante no ha sido aceptado aún, Cuando intenta enviar un mensaje, Entonces el sistema muestra el mensaje "Esperando aceptación del conductor para iniciar el chat."</p>	EP05
US14	Ruta de mi viaje	Como estudiante, quiero poder ver la ruta del viaje, para saber dónde se encuentra el conductor.	<p>Scenario 1: Visualización del mapa Dado que el estudiante ha solicitado un viaje, Cuando accede al seguimiento del viaje, Entonces el sistema muestra la ruta en el mapa.</p> <p>Scenario 2: Sin señal Dado que el estudiante está en un viaje activo, Cuando no puede obtener señal, Entonces el sistema muestra el mensaje "No se puede obtener la ruta y ubicación."</p>	EP11
US15	Historial de viajes y gastos	Como estudiante, quiero consultar el historial de mis viajes y los gastos generados, para llevar un control de mis viajes y pagos.	<p>Scenario 1: Ver historial completo de viajes Dado que el estudiante está en la sección de "historial", Cuando selecciona la opción de "ver todos los viajes", Entonces el sistema muestra la lista de viajes realizados, con "fechas" y "detalles".</p> <p>Scenario 2: Ver historial con filtros de búsqueda Dado que el estudiante está en la sección de "historial", Cuando aplica filtros como "fecha" o "tipo de viaje", Entonces el sistema muestra solo los viajes que coinciden con los criterios de búsqueda.</p>	EP12
US16	Planificación de viajes recurrentes	Como estudiante, quiero poder programar viajes recurrentes, para no tener que buscar cada vez que necesite realizar el mismo trayecto.	<p>Scenario 1: Crear un viaje recurrente Dado que el estudiante está en la pantalla de "planificación", Cuando selecciona la opción de "planificar un viaje recurrente", Y establece las fechas y parámetros para el viaje, Entonces el sistema guarda el viaje como recurrente y muestra la próxima fecha.</p> <p>Scenario 2: No crear viaje recurrente sin fecha válida Dado que el estudiante está en la pantalla de "planificación", Cuando no introduce una fecha válida para el viaje recurrente, Entonces el sistema muestra un mensaje de error "Fecha inválida, por favor ingrese una fecha correcta."</p>	EP13
US17	Registro como conductor	Como estudiante con vehículo, quiero registrarme como conductor para poder ofrecer viajes a otros estudiantes.	<p>Scenario 1: Registro de conductor con datos válidos Dado que el estudiante no está registrado como conductor, Y se encuentra en la pantalla de "registro" como conductor, Cuando introduce sus datos personales, datos del vehículo y licencia de conducir válidos, Y acepta los términos y condiciones, Entonces el sistema crea la cuenta de conductor Y envía un correo de verificación al estudiante.</p> <p>Scenario 2: Registro con datos inválidos Dado que el estudiante está en la pantalla de "registro" como conductor, Cuando introduce datos del vehículo o licencia inválidos, Entonces el sistema muestra un mensaje de error Y no permite continuar con el registro.</p>	EP01
US18	Verificación de licencia de conducir y datos del vehículo	Como conductor registrado, quiero que mis datos y licencia sean verificados, para asegurarme de que puedo ofrecer viajes de manera legal y segura.	<p>Scenario 1: Verificación exitosa de licencia y vehículo Dado que el conductor ha registrado su licencia y datos del vehículo, Cuando el sistema verifica que los datos son válidos, Entonces el sistema confirma la verificación y permite publicar viajes.</p> <p>Scenario 2: Verificación fallida de licencia o vehículo Dado que el conductor ha registrado su licencia y datos del vehículo, Cuando el sistema detecta datos inválidos o incorrectos, Entonces el sistema muestra un mensaje de error Y solicita corrección de los datos.</p>	EP07

Epic / User Story ID	Título	Descripción	Criterios de Aceptación	Relacionado con (Epic ID)
US19	Publicación de viajes	Como conductor, quiero poder publicar mis viajes disponibles indicando la ruta, horarios, asientos disponibles y costo por pasajero, para que los estudiantes puedan unirse a ellos.	<p>Scenario 1: Publicación de viaje con datos válidos Dado que el conductor está en la pantalla de "publicación de viaje", Cuando introduce "la ruta", "horario", "asientos disponibles" y "costo por pasajero", Y publica el viaje, Entonces el sistema crea el viaje y lo muestra en la lista de viajes disponibles para los estudiantes.</p> <p>Scenario 2: Publicación de viaje con datos inválidos Dado que el conductor está en la pantalla de "publicación de viaje", Cuando introduce datos incompletos o inválidos, Entonces el sistema muestra un mensaje de error Y no permite publicar el viaje.</p>	EP13
US20	Gestión de solicitudes	Como conductor, quiero poder aceptar o rechazar solicitudes de pasajeros con perfiles verificados, para asegurarme de que los pasajeros sean confiables.	<p>Scenario 1: Aceptar solicitud de pasajero Dado que el conductor ha recibido una solicitud para unirse a su viaje, Cuando el conductor acepta la solicitud, Entonces el sistema confirma la aceptación al pasajero Y lo agrega a la lista de pasajeros del viaje.</p> <p>Scenario 2: Rechazar solicitud de pasajero Dado que el conductor ha recibido una solicitud para unirse a su viaje, Cuando el conductor rechaza la solicitud, Entonces el sistema notifica al pasajero que la solicitud ha sido rechazada.</p> <p>Scenario 3: Solicitud con perfil no verificado Dado que el conductor ha recibido una solicitud de un pasajero con perfil no verificado, Cuando el conductor revisa el perfil, Entonces el sistema muestra el mensaje "Perfil no verificado" Y el conductor no puede aceptar la solicitud.</p>	EP13
US21	Visualización de calificación	Como conductor, quiero ver la calificación promedio y los comentarios de pasajeros previos, para poder evaluar mi desempeño y mejorar el servicio.	<p>Scenario 1: Visualización de calificación promedio Dado que el conductor ha completado varios viajes, Cuando accede a la sección de "reputación", Entonces el sistema muestra la calificación promedio obtenida en todos los viajes.</p> <p>Scenario 2: Visualización de comentarios de pasajeros previos Dado que el conductor ha completado varios viajes, Cuando accede a la sección de "reputación", Entonces el sistema muestra los "comentarios de los pasajeros" sobre los viajes anteriores.</p>	EP16
US22	Chat interno con pasajeros	Como conductor, quiero comunicarme con los pasajeros a través del chat, para coordinar puntos de encuentro y cambios de última hora.	<p>Scenario 1: Enviar mensaje a un pasajero Dado que el conductor tiene un pasajero confirmado, Cuando escribe un mensaje en el chat, Entonces el mensaje es enviado y recibido por el pasajero en tiempo real.</p> <p>Scenario 2: No poder enviar mensaje antes de aceptación Dado que el pasajero no ha sido aceptado en el viaje, Cuando el conductor intenta enviar un mensaje, Entonces el sistema muestra el mensaje "Esperando aceptación del pasajero."</p>	EP05
US23	Historial de viajes y ganancias	Como conductor, quiero ver el historial de viajes realizados y las ganancias generadas, para llevar un registro de mi actividad.	<p>Scenario 1: Ver historial de viajes Dado que el conductor ha completado varios viajes, Cuando el conductor accede a la sección de "historial" de viajes, Entonces el sistema muestra todos los viajes realizados y las ganancias generadas.</p> <p>Scenario 2: Sin historial de viajes Dado que el conductor no ha realizado viajes, Cuando accede a la sección de "historial" de viajes, Entonces el sistema muestra el mensaje "No tienes viajes registrados."</p>	EP05
US24	Notificaciones de demanda	Como conductor, quiero recibir alertas sobre zonas y horarios con alta demanda de transporte, para poder ajustar mis viajes y atender la necesidad de los estudiantes.	<p>Scenario 1: Recibir alerta de alta demanda Dado que el conductor ha indicado disponibilidad en ciertos horarios, Cuando hay alta demanda en esa zona y horario, Entonces el sistema envía una notificación al conductor.</p> <p>Scenario 2: No recibir alerta sin disponibilidad Dado que el conductor no tiene viajes programados, Cuando hay alta demanda en una zona, Entonces el sistema no envía ninguna alerta.</p>	EP17

Epic / User Story ID	Título	Descripción	Criterios de Aceptación	Relacionado con (Epic ID)
US25	Reporte de incidentes	Como estudiante, quiero reportar incidentes relacionados con conductores o pasajeros, para contribuir a la seguridad del sistema.	<p>Scenario 1: Reportar un incidente Dado que el familiar ha tenido una experiencia negativa, Cuando accede a la opción de reporte, Entonces puede seleccionar el motivo del incidente, describirlo y enviarlo al sistema.</p> <p>Scenario 2: Confirmación de envío Dado que el familiar ha completado el reporte, Cuando lo envía, Entonces el sistema confirma que el incidente ha sido registrado correctamente.</p>	EP06

Technical Stories

Endpoint de Registro de Usuario

Campo	Detalle
Épica	Microservicio – Autenticación
ID-TS	TS01
Título TS	Endpoint para registro de usuario
Descripción TS	Como Developer, necesito implementar un endpoint que permita a los estudiantes registrarse con datos institucionales, almacenando la información de forma segura en la base de datos.
Criterios de Aceptación (Gherkin)	Scenario 1: Registro exitoso Dado que se recibe una request POST a /api/auth/register con datos válidos, Cuando el correo es institucional, la contraseña cumple con las políticas de seguridad y el nombre está completo, Entonces el sistema responde con código 201 y confirma el registro.
	Scenario 2: Registro inválido Dado que se recibe una request POST a /api/auth/register , Cuando falta algún campo obligatorio o el correo no es institucional, Entonces el sistema responde con código 400 y un mensaje de error.

Endpoint de Login

Campo	Detalle
Épica	Microservicio – Autenticación
ID-TS	TS02
Título TS	Endpoint para inicio de sesión
Descripción TS	Como Developer, necesito habilitar un endpoint de login que permita a los usuarios autenticarse y recibir un token JWT para acceder a los demás servicios.
Criterios de Aceptación (Gherkin)	Scenario 1: Login válido Dado que se recibe una request POST a /api/auth/login con credenciales correctas, Cuando las credenciales corresponden a un usuario registrado, Entonces el sistema responde con código 200 y un token JWT válido.
	Scenario 2: Login inválido Dado que se recibe una request POST a /api/auth/login , Cuando el correo o la contraseña no coinciden, Entonces el sistema responde con código 401 y un mensaje de error.

Endpoint para publicar viaje

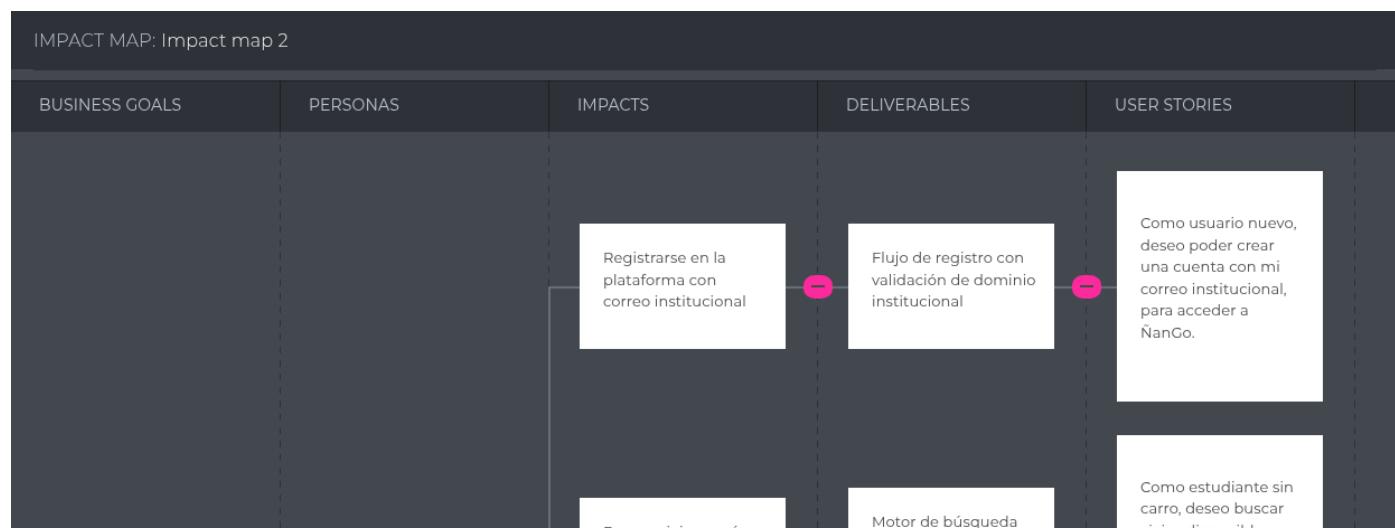
Campo	Detalle
Épica	Microservicio – Gestión de Viajes
ID-TS	TS03
Título TS	Endpoint para publicar un viaje
Descripción TS	Como Developer, necesito crear un endpoint que permita a los conductores publicar un viaje indicando ruta, hora, cupos y costo.

Campo	Detalle
Criterios de Aceptación (Gherkin)	<p>Scenario 1: Publicación exitosa Dado que el conductor está autenticado, Cuando envía una request POST a <code>/api/trips</code> con información válida, Entonces el sistema almacena el viaje y responde con código 201.</p> <p>Scenario 2: Publicación inválida Dado que se recibe una request POST a <code>/api/trips</code>, Cuando falta algún campo obligatorio (ej. ruta, fecha, cupos), Entonces el sistema responde con código 400.</p>
Endpoint para solicitar unirse a un viaje	
Campo	Detalle
Épica	Microservicio – Gestión de Viajes
ID-TS	TS04
Título TS	Endpoint para solicitar unirse a un viaje
Descripción TS	Como Developer, necesito permitir que los pasajeros envíen solicitudes para unirse a un viaje previamente publicado.
Criterios de Aceptación (Gherkin)	<p>Scenario 1: Solicitud enviada con éxito Dado que el pasajero está autenticado, Cuando envía una request POST a <code>/api/trips/{trip_id}/join</code>, Entonces el sistema registra la solicitud y responde con código 200.</p> <p>Scenario 2: Solicitud duplicada Dado que el pasajero ya solicitó unirse a ese viaje, Entonces el sistema responde con código 409 y un mensaje de error.</p>

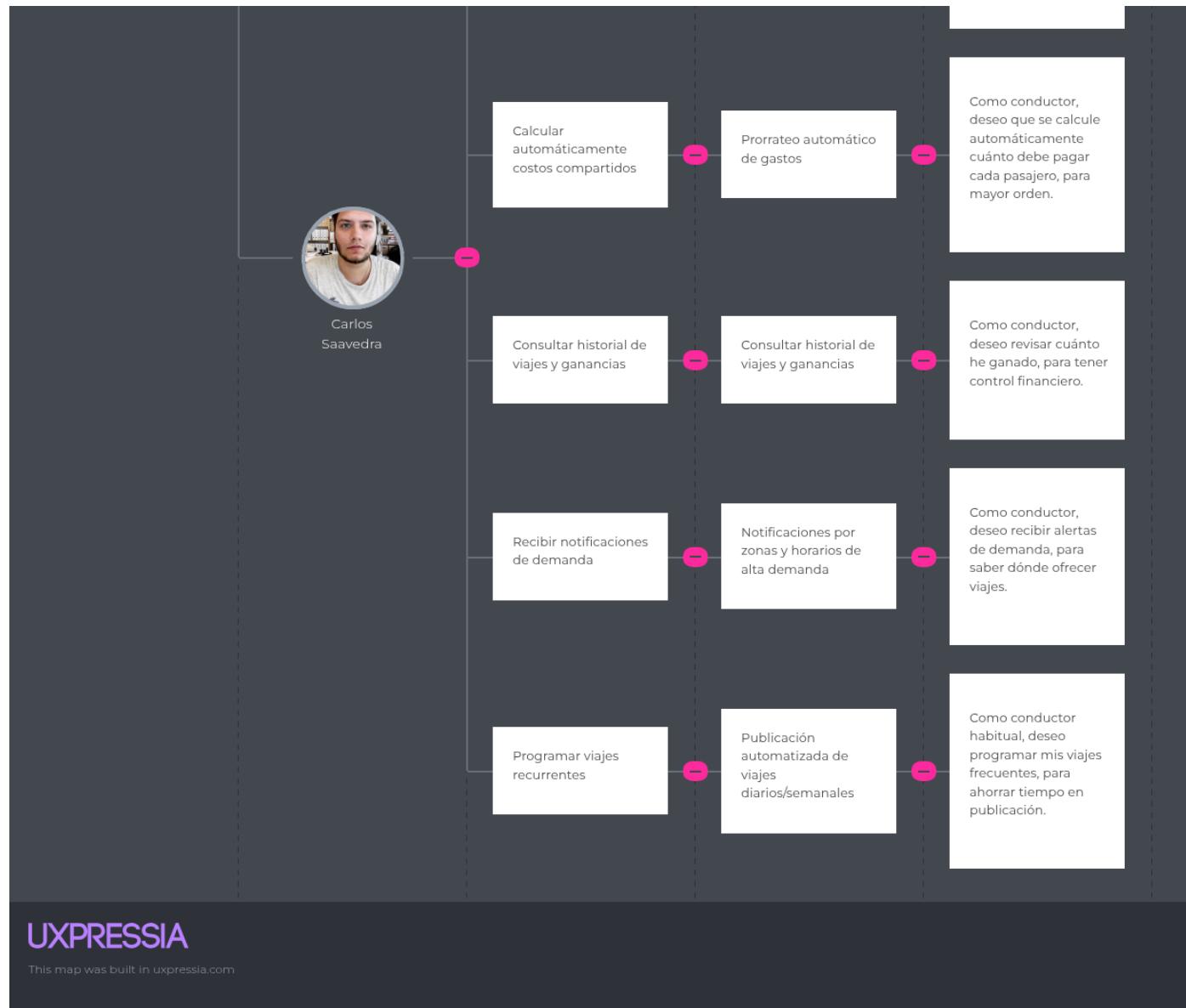
Endpoint de Validación de Identidad Estudiantil

Campo	Detalle
Épica	Microservicio – Validación de Identidad
ID-TS	TS05
Título TS	Validación de ID universitario
Descripción TS	Como Developer, necesito implementar la integración con un servicio externo que valide la autenticidad del carné universitario para garantizar que los usuarios pertenecen a una institución educativa.
Criterios de Aceptación (Gherkin)	<p>Scenario 1: Validación exitosa Dado que se recibe una request POST a <code>/api/identity/validate</code> con la imagen del carné, Cuando el servicio externo confirma que el documento es válido, Entonces el sistema responde con código 200 y <code>{ "valid": true, "institution": "Universidad X" }</code>.</p> <p>Scenario 2: Validación fallida Dado que el documento es ilegible o no válido, Entonces el sistema responde con código 400 y <code>{ "valid": false, "error": "Documento no válido" }</code>.</p>

3.3 Impact Map







3.4. Product Backlog

#Orden	User Story ID	Título	Descripción	Story Points
1	TS01	Endpoint para registro de usuario	POST /api/auth/register para crear cuentas con validación institucional.	3
2	TS02	Endpoint para inicio de sesión	POST /api/auth/login para autenticar y devolver JWT.	2
3	US01	Registro de nueva cuenta	Crear cuenta con datos válidos y envío de verificación.	5
4	US03	Inicio de sesión	Autenticación y redirección a la pantalla principal (recordarme opcional).	3
5	US04	Cierre de sesión	Cerrar sesión y volver a inicio; manejo de error por red.	2
6	US02	Recuperación de contraseña	Enviar enlace y restablecer contraseña; validar correos registrados.	3
7	US07	Cambiar contraseña	Actualizar contraseña autenticada; validar contraseña actual.	3
8	US05	Edición de perfil	Editar nombre, email, celular, plan y foto con validaciones.	5
9	US08	Verificación con DNI y Carnet	Subir y validar documentos para acceso completo.	8
10	TS05	Validación de ID universitario	Integración externa para validar carné; responde {valid, institution}.	8
11	US17	Registro como conductor	Alta de conductor con datos personales, vehículo y licencia.	5
12	US18	Verificación de licencia y vehículo	Validar legalidad para publicar viajes.	8
13	US19	Publicación de viajes	Crear viaje con ruta, horario, cupos y costo.	5
14	TS03	Endpoint publicar viaje	POST /api/trips para crear viajes.	3
15	US09	Búsqueda de viajes disponibles	Filtrar por fecha, origen y destino; mensajes sin resultados.	5

#Orden	User Story ID	Título	Descripción	Story Points
16	US10	Solicitud de unirse a un viaje	Enviar solicitud, reservar y manejar viajes llenos.	3
17	TS04	Endpoint solicitar unirse a un viaje	POST /api/trips/{trip_id}/join; manejar duplicados.	3
18	US11	Notificaciones en tiempo real	Avisos de aceptación/rechazo de solicitudes.	8
19	US13	Chat interno con Grupo (pasajero)	Chat con conductor y grupo tras aceptación; bloqueo previo.	5
20	US22	Chat interno con pasajeros (conductor)	Coordinar puntos y cambios; bloqueo antes de aceptación.	5
21	US14	Ruta de mi viaje	Visualizar ruta y ubicación; manejo sin señal.	5
22	US16	Planificación de viajes recurrentes	Programar recorrenias y validar fechas.	8
23	US15	Historial de viajes y gastos (estudiante)	Ver historial completo y con filtros.	5
24	US23	Historial de viajes y ganancias (conductor)	Listar viajes realizados y ganancias; vacío sin registros.	5
25	US21	Visualización de calificación (conductor)	Ver promedio y comentarios de pasajeros.	3
26	US24	Notificaciones de demanda	Alertas por zonas/horarios con alta demanda según disponibilidad.	8
27	US12	Calificación post-viaje	Guardar calificación y comentarios; validar incompletos.	3
28	US20	Gestión de solicitudes (conductor)	Aceptar/rechazar; bloquear perfiles no verificados.	5
29	US25	Reporte de incidentes	Reportar motivo y descripción; confirmar recepción.	3
30	EP11/Infra	Rutas de viaje en el mapa (infra y SDK)	Preparar SDK/mapa, claves y capas para rutas óptimas.	5

4.1 Design Concepts, ViewPoints & ER Diagrams

4.1.1 Principles Statements

- Desacoplamiento de componentes:** Diseñar el sistema como un conjunto de módulos independientes con responsabilidades claras e interfaces bien definidas. Esto permite escalar funcionalidades de manera eficiente, facilitar el mantenimiento y realizar pruebas más precisas.
- Seguridad por diseño:** Incorporar medidas de seguridad en todas las capas del sistema para garantizar la seguridad de los datos de los estudiantes y conductores, así como la verificación de estos, incluyendo validación estricta en el dominio, autenticación y autorización robusta, cifrado de datos sensibles, y protección contra vulnerabilidades.
- Resiliencia a fallos:** Diseñar el sistema para que pueda recuperarse ante errores y continuar operando con mínima interrupción. Esto implica implementar mecanismos como reinicios automáticos, circuit breakers, timeouts configurables, monitoreo activo y trazabilidad distribuida para detectar y resolver problemas rápidamente.
- Consistencia de interfaz:** Ofrecer una experiencia de usuario coherente, intuitiva y accesible en todas las vistas de la aplicación. Respetando las interfaces y una estética uniforme centradas en la facilidad de uso para los usuarios.
- Escalabilidad internacional:** Preparar el sistema para crecer en funcionalidades y cobertura internacional. Esto incluye soporte multitenant para distintas universidades, configuración dinámica de reglas de negocio, y capacidades de internacionalización (i18n) para adaptarse a distintos contextos culturales y lingüísticos.
- Interacciones asincrónicas sobre dependencias sincrónicas:** La comunicación entre módulos y servicios debe favorecer la asincronía (notificaciones, coordinación de viajes, etc.) para mejorar escalabilidad y resiliencia.
- Capacidad de integración con APIs externas:** Permitir que el sistema evolucione con integraciones externas (pasarelas de pago, mapas de terceros, servicios de autenticación).
- Uso de bibliotecas y frameworks con soporte comercial:** Para garantizar continuidad, seguridad y soporte, optar por librerías consolidadas y con respaldo de comunidad o proveedor.
- Experiencia de usuario centrada en la confianza:** Todas las decisiones tecnológicas deben estar alineadas a la confianza de los usuarios, la transparencia en costos y la facilidad de uso.
- Observabilidad y Trazabilidad:** Facilitar el monitoreo, diagnóstico y mejora continua del sistema mediante métricas clave y registros estructurados, facilitando la toma de decisiones informada y dar un correcto mantenimiento a la aplicación.

4.1.2 Approaches Statements Architectural Styles & Patterns

- Domain Driven Design (DDD):** Modelar el sistema enfocándose en el dominio universitario y de transporte. Esto permite construir una arquitectura centrada en conceptos clave como aggregates, entities, value objects, commands y events, asegurando que las decisiones técnicas reflejen las reglas del negocio y evolucionen junto con él.
- Attribute-Driven Design:** Se empleará para tomar decisiones arquitectónicas basadas en atributos de calidad como rendimiento, seguridad, escalabilidad, mantenibilidad y disponibilidad. Este enfoque asegura que el sistema cumpla con sus requisitos no funcionales desde el diseño, priorizando los escenarios críticos para los usuarios y stakeholders.
- Architectural Styles and Pattern:**

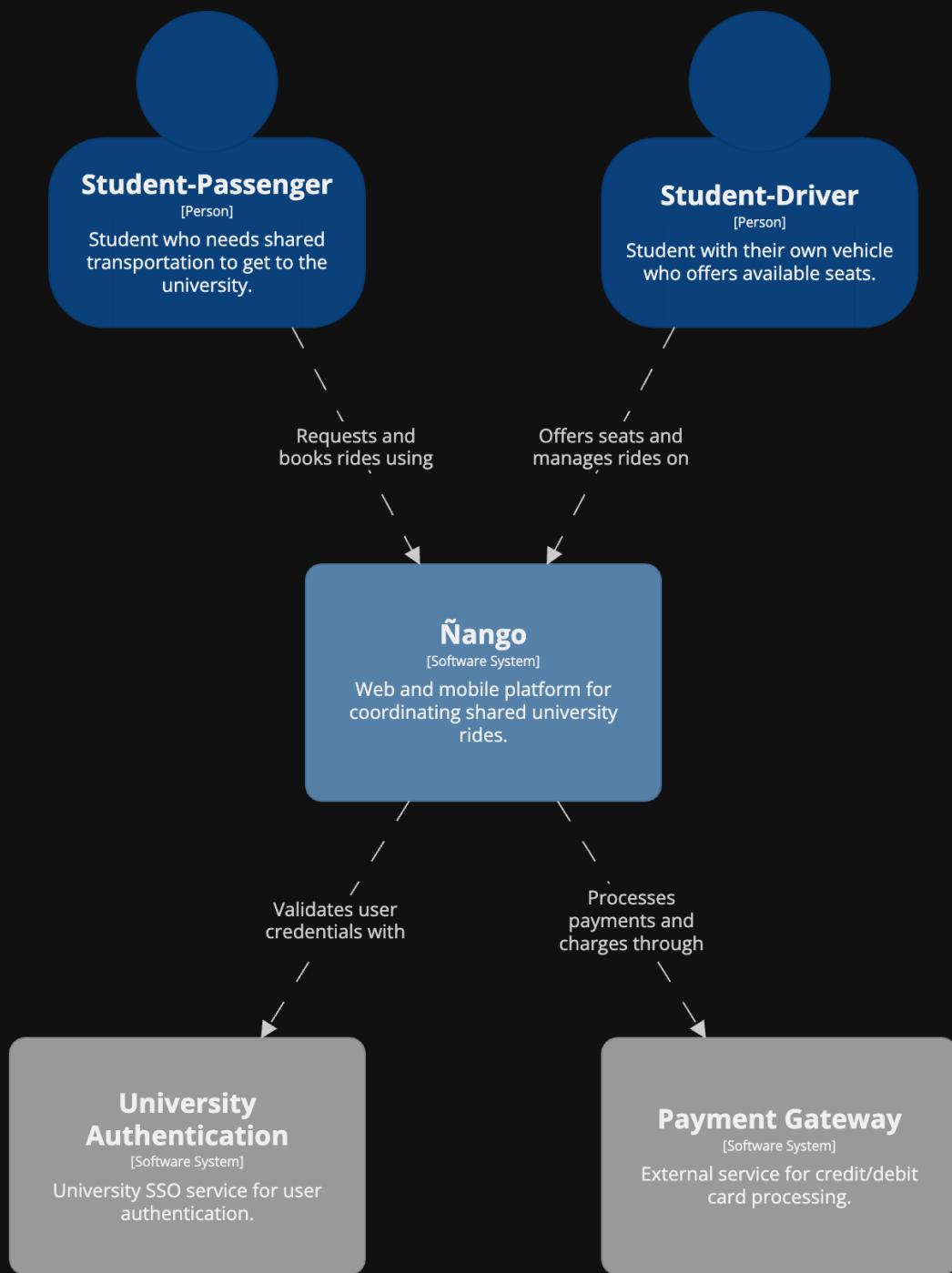
- **Estilos Arquitectónicos:**

- **Cliente-Servidor:** Se Dividirá la aplicación en dos componentes principales: cliente (front-end web) y servidor (back-end). Este estilo se ajusta al esquema operativo de ÑanGo, permitiendo una clara separación de responsabilidades, una comunicación estructurada y una implementación eficiente de la lógica de presentación y negocio.
- **Microservicios:** La aplicación se dividirá en servicios pequeños e independientes, cada uno responsable de una funcionalidad específica (gestión de usuarios, rutas, pagos, notificaciones) y comunicándose por medio de APIs, lo que permitirá escalabilidad y despliegues independientes.

- **Patrones de Diseño:**

- **Modelo-Vista-Controlador (MVC):** Se aplicará principalmente en la capa del cliente, separa la lógica de presentación (Vista), la lógica de interacción (Controlador) y el modelo de datos (Modelo). Esto mejora la organización del código, facilita el mantenimiento y permite una experiencia de usuario más coherente y dinámica.
- **API Gateway:** Actúa como punto único de entrada de las solicitudes del cliente, gestionando el enruteamiento hacia los microservicios correspondientes, además de ofrecer beneficios en seguridad y monitoreo. Simplifica la interacción entre el cliente y el backend distribuido.

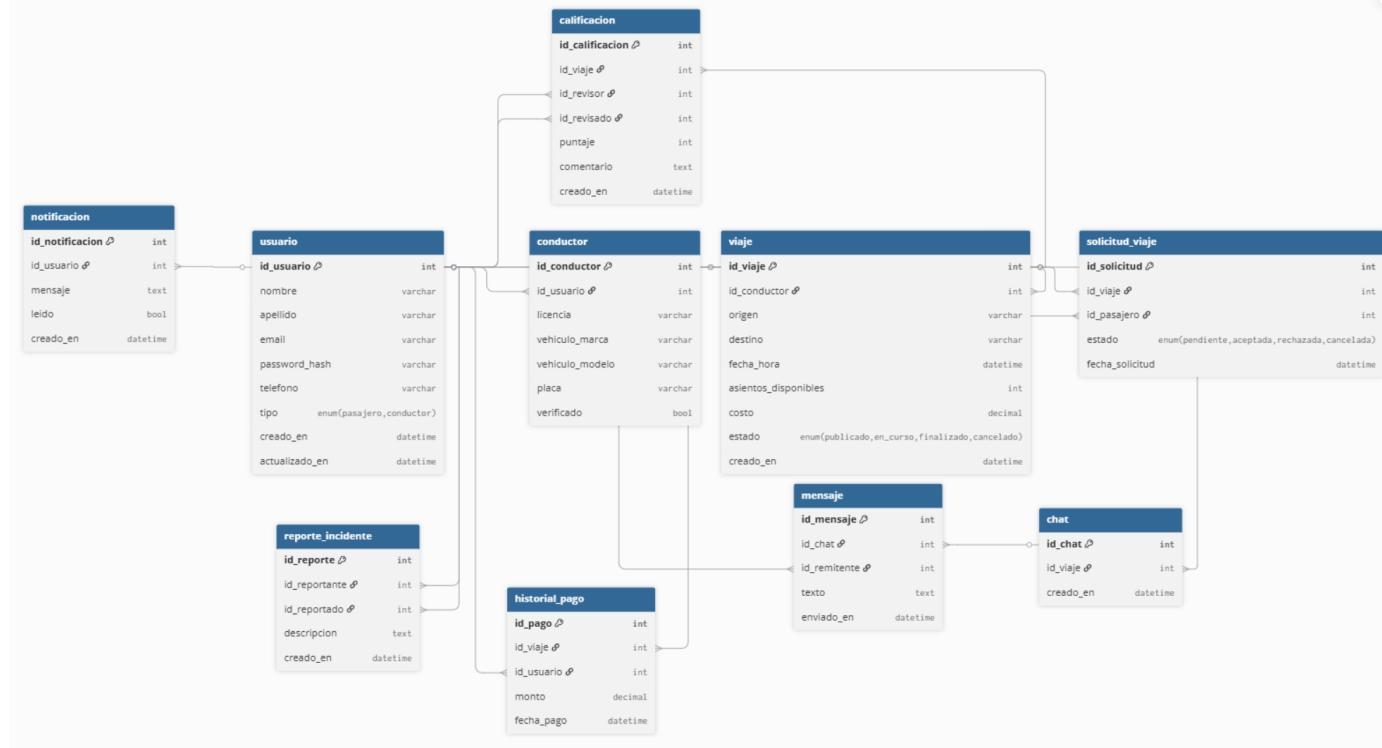
4.1.3 Context Diagram



4.6.1 System Context Diagram - Ñango

viernes, 10 de octubre de 2025, 4:22 hora de verano de Europa central

4.1.5 Relational/Non Relational Database Diagram



4.1.6 Design Patterns

Patrones seleccionados

1. Singleton (Creacional)

- Problema:** Se requiere una sola instancia para manejar la configuración y la conexión a la base de datos.
- Aplicación:** Clase `DatabaseConnection` asegurará una única conexión compartida por todos los módulos.
- Beneficios:** Evita múltiples conexiones, centraliza la configuración.

2. Repository (Estructural)

- Problema:** Separar la lógica de negocio de la persistencia de datos.
- Aplicación:** Repositorios `UsuarioRepository`, `ViajeRepository`, etc., que encapsulan consultas y operaciones de la base de datos.
- Beneficios:** Facilita pruebas unitarias, cambio de motor de BD sin alterar la lógica de dominio.

3. Observer / Publisher–Subscriber (Comportamiento)

- Problema:** Enviar notificaciones en tiempo real cuando se actualiza un viaje o se acepta una solicitud.
- Aplicación:** El módulo de notificaciones se suscribe a eventos del módulo de viajes y emite mensajes a los usuarios.
- Beneficios:** Desacopla el emisor de los receptores, facilita añadir nuevos canales de notificación.

4. Strategy (Comportamiento)

- Problema:** Permitir distintos cálculos de tarifas o métodos de pago.
- Aplicación:** Una interfaz `PaymentStrategy` con implementaciones para tarjeta, efectivo, billetera digital.
- Beneficios:** Permite añadir nuevos métodos sin cambiar la lógica central.

5. Factory Method (Creacional)

- Problema:** Crear objetos de notificación (Push, Email, SMS) sin acoplar el código a clases concretas.
- Aplicación:** `NotificationFactory` devolverá el tipo de notificación adecuado según configuración.
- Beneficios:** Centraliza la creación y facilita la extensión.

4.1.7 Tactics

• Seguridad

- Autenticación basada en **sesiones**, garantizando persistencia segura de las credenciales de usuario.
- Encriptación de contraseñas para proteger datos sensibles.
- Verificación de identidad estudiantil mediante **Student Beans**, con respaldo de validación manual por correo institucional y carné universitario.
- Uso de canales cifrados bajo **HTTPS/TLS** en todas las comunicaciones.

• Disponibilidad

- Balanceo de carga a través de un **API Gateway** para distribuir tráfico.
- Replicación de base de datos en entornos relacionales y no relacionales para tolerancia a fallos.

- Estrategias de reintentos automáticos en colas de mensajes para operaciones críticas.

- **Escalabilidad**

- Arquitectura de **microservicios cloud-native** desplegada en contenedores.
- Escalado automático en entornos de nube según demanda.
- Desacoplamiento de componentes a través de colas de mensajería y almacenamiento en caché distribuido.

- **Usabilidad**

- Notificaciones en tiempo real mediante **Firebase Cloud Messaging (FCM)**.
- Chat interno soportado por **Socket.IO** para comunicación en tiempo real.
- Interfaces simples y accesibles con diseño responsivo.

- **Rendimiento**

- Cacheo de datos de uso frecuente para reducir tiempos de respuesta.
- Indexación de consultas en base de datos.
- Aplicación de **CQRS** en la gestión de viajes para separar operaciones de lectura y escritura.

- **Mantenibilidad**

- Organización del acceso a datos bajo patrones de diseño.
- Documentación de servicios mediante especificación estandarizada de APIs.
- Convenciones de control de versiones y buenas prácticas de colaboración en repositorios de código.

4.2 Architectural Drivers

4.2.1 Design Purpose

El diseño arquitectónico de ÑanGo tiene como propósito asegurar que la plataforma de movilidad estudiantil sea **segura, confiable y escalable**, ofreciendo una experiencia fluida a conductores y pasajeros dentro de la comunidad universitaria. La arquitectura basada en microservicios y desplegada en la nube permite que cada módulo —como autenticación, gestión de viajes, solicitudes, notificaciones y chat— evolucione de manera independiente, facilitando la incorporación de nuevas funcionalidades sin afectar al sistema completo.

El uso de **sesiones para autenticación** refuerza la seguridad y la confianza de los usuarios, mientras que la verificación estudiantil con **Student Beans** garantiza que la comunidad se mantenga cerrada y validada. Además, la integración de **Firebase Cloud Messaging** y **Socket.IO** respalda la comunicación en tiempo real, un aspecto clave para notificaciones instantáneas y coordinación entre estudiantes.

En conjunto, este diseño busca cumplir con los **atributos de calidad** más relevantes para el producto: seguridad, disponibilidad, escalabilidad, usabilidad y mantenibilidad. Al mismo tiempo, responde a los objetivos de negocio de la startup, centrados en ofrecer una solución tecnológica que aumente la seguridad en el transporte universitario y genere confianza en la comunidad académica.

4.2.2 Primary Functionality (Primary User Stories)

Registro y Verificación de Identidad (Núcleo de Seguridad)

User Story ID	Título	Descripción
US01	Registro de nueva cuenta	Como usuario nuevo no registrado, deseo poder crear una cuenta en la aplicación, para acceder a las funcionalidades exclusivas de ÑanGo.
US17	Registro como conductor	Como estudiante con vehículo, quiero registrarme como conductor para poder ofrecer viajes a otros estudiantes.
US08	Verificación con correo institucional	Como estudiante, quiero verificar mi cuenta utilizando mi "DNI" y "Carnet Universitario", para tener acceso a las funcionalidades de la aplicación.

Gestión del Ciclo de Vida de los Viajes (Funcionalidad Central)

User Story ID	Título	Descripción
US19	Publicación de viajes	Como conductor, quiero poder publicar mis viajes disponibles indicando la ruta, horarios, asientos disponibles y costo por pasajero, para que los estudiantes puedan unirse a ellos.
US09	Búsqueda de viajes disponibles	Como estudiante sin vehículo, quiero buscar viajes disponibles, para poder unirme a ellos.
US10	Solicitud de unirse a un viaje	Como estudiante, quiero solicitar unirme a un viaje disponible, para poder participar en el transporte compartido.
US20	Gestión de solicitudes	Como conductor, quiero poder aceptar o rechazar solicitudes de pasajeros con perfiles verificados, para asegurarme de que los pasajeros sean confiables.

Comunicación y Seguimiento en Tiempo Real

User Story ID	Título	Descripción
US06	Chat de comunicación	Como usuario, quiero comunicarme con otros mediante chat, para coordinar viajes y resolver dudas.
US13	Chat interno con Grupo	Como estudiante, quiero poder comunicarme con el conductor a través del chat interno una vez que mi solicitud de viaje haya sido aceptada.
US22	Chat interno con pasajeros	Como conductor, quiero comunicarme con los pasajeros a través del chat, para coordinar puntos de encuentro y cambios de última hora.
US11	Notificaciones en tiempo real	Como estudiante, quiero recibir notificaciones en tiempo real sobre el estado de mi solicitud de viaje, para estar informado de cualquier cambio.
US14	Ruta de mi viaje	Como estudiante, quiero poder ver la ruta del viaje, para saber dónde se encuentra el conductor.

4.2.3 Quality Attribute Scenarios

1. Escenario de Disponibilidad

ENTRADAS	Fuente de Estímulo	Un gran número de usuarios (estudiantes pasajeros y conductores).
	Estímulo	Múltiples intentos simultáneos de publicar, buscar y coordinar viajes.
CONDICIONES	Entorno	El sistema está operando en condiciones normales durante una hora punta (ej. 7-9 AM y 5-7 PM en un día laborable), que es cuando la demanda es más alta.
	Artefacto	Servicios de Gestión de Viajes y Búsqueda.
RESULTADOS ESPERADOS	Respuesta	El sistema procesa todas las solicitudes sin fallos, permitiendo a los usuarios completar sus operaciones de búsqueda y coordinación de manera exitosa.
	Medida de Respuesta	El sistema mantiene una disponibilidad del 99.9% durante las horas punta. La tasa de éxito de las solicitudes críticas (buscar, publicar, solicitar) es superior al 99.5%.

2. Escenario de Seguridad

ENTRADAS	Fuente de Estímulo	Un actor externo malintencionado.
	Estímulo	Un intento de acceso no autorizado para ver el historial de viajes o la información personal de un usuario.
CONDICIONES	Entorno	El sistema está en operación normal.
	Artefacto	La base de datos de usuarios y los endpoints de la API que gestionan la información del perfil y el historial de viajes.
RESULTADOS ESPERADOS	Respuesta	El sistema, a través de sus tácticas de seguridad como la autenticación basada en sesiones y la encriptación, detecta y bloquea el intento de acceso. No se expone ninguna información sensible y el intento es registrado para auditoría.
	Medida de Respuesta	El intento de acceso no autorizado es rechazado en el 100% de los casos. Se genera una alerta de seguridad para el equipo de operaciones en menos de 1 segundo tras el intento fallido.

3. Escenario de Usabilidad

ENTRADAS	Fuente de Estímulo	Un nuevo usuario (estudiante pasajero) que utiliza la aplicación por primera vez.
	Estímulo	El usuario intenta realizar la tarea completa de buscar un viaje disponible para mañana, seleccionar uno y enviar una solicitud para unirse.
CONDICIONES	Entorno	El usuario está utilizando la aplicación web en un dispositivo móvil con una conexión a internet estándar.
	Artefacto	La interfaz de usuario (UI) de la aplicación Nango.
RESULTADOS ESPERADOS	Respuesta	El usuario completa la tarea de buscar y solicitar un viaje de forma exitosa y sin frustración, siguiendo un flujo intuitivo como se espera de una interfaz "amigable y confiable".
	Medida de Respuesta	Al menos el 95% de los nuevos usuarios en un entorno de prueba pueden completar esta tarea en menos de 90 segundos sin necesidad de consultar un manual de ayuda o soporte técnico.

4. Escenario de Rendimiento

ENTRADAS	Fuente de Estímulo	Un usuario (estudiante pasajero).
	Estímulo	El usuario realiza una búsqueda de viajes disponibles aplicando filtros de fecha, origen y destino.
CONDICIONES	Entorno	El sistema se encuentra bajo una carga de 100 usuarios concurrentes realizando operaciones de lectura (búsquedas, visualización de perfiles) y escritura (publicación de viajes, envío de mensajes).
	Artefacto	Servicio de Búsqueda y la base de datos de viajes.

RESULTADOS ESPERADOS	Respuesta	El sistema procesa la consulta, aplica los filtros correspondientes y devuelve una lista de resultados relevantes al usuario.
	Medida de Respuesta	Los resultados de la búsqueda se muestran en la interfaz del usuario en menos de 2 segundos para el 99% de las solicitudes.

4.2.4 Constraints

Los "constraints" se refieren a las limitaciones o restricciones que deben ser consideradas durante la planificación, diseño y ejecución de un proyecto de software y arquitectura de sistemas.

ID	Constraint
CON-001	Arquitectura de Microservicios <ul style="list-style-type: none"> La plataforma debe construirse bajo una arquitectura de microservicios, garantizando escalabilidad y flexibilidad en el despliegue y mantenimiento.
CON-002	Autenticación Universitaria Externa <ul style="list-style-type: none"> El sistema debe integrarse obligatoriamente con Student Beans para validar la pertenencia universitaria de los usuarios antes de permitir el acceso completo a la plataforma.
CON-003	Notificaciones en Tiempo Real <ul style="list-style-type: none"> Las notificaciones deben gestionarse a través de Firebase Cloud Messaging (FCM) para asegurar alertas inmediatas sobre viajes, solicitudes y actualizaciones.
CON-004	Procesamiento de Pagos Seguro <ul style="list-style-type: none"> Todas las transacciones deben realizarse mediante un gateway de pagos certificado (PCI DSS), asegurando la protección de datos financieros de los estudiantes.
CON-005	Autenticación y Autorización Segura <ul style="list-style-type: none"> Se debe implementar un esquema de autenticación basado en estándares modernos como OAuth 2.0 y JWT, protegiendo los accesos y la comunicación con los servicios internos y externos.
CON-006	Disponibilidad del Sistema <ul style="list-style-type: none"> El sistema debe garantizar un uptime mínimo del 99.5%, asegurando accesibilidad incluso en horas pico (inicio y fin de clases).
CON-007	Cifrado de Datos <ul style="list-style-type: none"> Los datos sensibles de los usuarios deben estar cifrados en tránsito (TLS 1.3) y en reposo (AES-256) para garantizar la confidencialidad y seguridad de la información.
CON-008	Tiempo de Respuesta <ul style="list-style-type: none"> Las operaciones críticas (ej. reserva de asiento, aceptación de solicitud) deben ejecutarse en menos de 3 segundos para asegurar una experiencia fluida.
CON-009	Compatibilidad Multiplataforma <ul style="list-style-type: none"> La aplicación debe estar disponible tanto en web como en dispositivos móviles (Android/iOS), asegurando accesibilidad para todos los estudiantes.
CON-010	Escalabilidad <ul style="list-style-type: none"> La plataforma debe poder escalar horizontalmente para soportar al menos 10,000 usuarios concurrentes sin afectar el rendimiento.

4.2.5 Architectural Concerns

ID	Architectural Concerns
ARC-001	Disponibilidad del Sistema <ul style="list-style-type: none"> Descripción: Existe la preocupación de que el sistema no cumpla con el requisito de disponibilidad mínima del 99.5%, especialmente durante horarios pico (entrada y salida de clases), lo que podría afectar la confianza y la experiencia de los usuarios.
ARC-002	Dominio Tecnológico del Equipo de Desarrollo <ul style="list-style-type: none"> Descripción: El equipo podría enfrentar una curva de aprendizaje o falta de experiencia con las tecnologías seleccionadas, lo que puede impactar en los tiempos de entrega y calidad del producto.
ARC-003	Adecuación del Modelo de Dominio <ul style="list-style-type: none"> Descripción: Existe la preocupación de que el modelo de dominio no cubra completamente aspectos clave como la gestión de rutas, validación universitaria o solicitudes de viaje, limitando la escalabilidad futura.
ARC-004	Escalabilidad de la Arquitectura <ul style="list-style-type: none"> Descripción: La arquitectura podría no soportar de manera eficiente un aumento rápido en la cantidad de usuarios concurrentes (ej. más de 10,000), comprometiendo la experiencia de uso.
ARC-005	Interoperabilidad con Servicios Externos <ul style="list-style-type: none"> Descripción: La integración con servicios como Student Beans (autenticación universitaria) y Firebase Cloud Messaging (notificaciones) puede presentar riesgos de dependencia y fallos si no se gestiona correctamente.
ARC-006	Seguridad y Protección de Datos <ul style="list-style-type: none"> Descripción: Es necesario garantizar que la arquitectura cumpla con altos estándares de seguridad, protegiendo datos sensibles de estudiantes mediante cifrado, autenticación robusta y control de accesos.

ID	Architectural Concerns													
ARC-	Procesamiento de Pagos													
007	<ul style="list-style-type: none"> Descripción: La correcta integración con una pasarela de pagos segura es crítica; cualquier error en disponibilidad o validación podría afectar la confianza y generar pérdidas financieras. 													
4.3 ADD Iterarions														
4.3.1 Iteration N: Autenticación, Gestión de Usuario y Ride														
4.3.1.1 Architectural Design Backlog – Iteración: Autenticación y Gestión de Usuario														
ID	Tipo	Driver (Descripción)	Prioridad	Historias / TS relacionadas	Componentes afectados	Criterios de aceptación	Notas / Riesgos							
ADB-01	Funcional	<p>Registro y autenticación de usuarios: flujo de registro con verificación por correo, login con JWT, recuperación y cambio de contraseña.</p>	Alta	US01, US02, US03, TS01, TS02, US07	Auth Service, User Service, Email Service, DB usuario , API Gateway, Security middleware	<ul style="list-style-type: none"> - POST /api/auth/register crea usuario no verificado y retorna 201. - Se envía correo con token de verificación. - POST /api/auth/login devuelve access y refresh token. - Flujo de recuperación de contraseña envía link y permite cambiar la contraseña. 	Uso de hashing seguro (bcrypt/argon2). JWT short-lived + refresh tokens. Riesgo: entrega de correo (proveedor externo).							
ADB-02	Calidad (Seguridad)	Seguridad global: TLS en transporte, hashing de contraseñas, RBAC, validaciones de entrada, rate limiting, protección contra CSRF/OWASP.	Alta	—	Todos los servicios (Auth crítico), API Gateway, DB	<ul style="list-style-type: none"> - Todas las APIs bajo TLS. - Contraseñas nunca en texto plano. - Accesos a endpoints críticos requieren token válido y rol. - Rate limits aplicados. 	Meta: 0 vulnerabilidades críticas en escaneo SAST básico.							
ADB-03	Calidad (Disponibilidad)	Alta disponibilidad y tolerancia a fallos: health checks, retries, circuit breaker en servicios de autenticación.	Media	—	Auth Service, API Gateway, Monitoring/Logging	<ul style="list-style-type: none"> - Health checks configurados. - Retries con backoff y circuit breakers frente a servicios externos. - Monitoreo y alertas activas. 	SLA objetivo 99.9 % para Auth y User Service.							
ADB-04	Calidad (Rendimiento)	Escalabilidad: garantizar respuesta <200 ms en login/registro y capacidad de escalar horizontalmente servicios de autenticación y usuario.	Alta	—	Auth Service, User Service, Cache (Redis), Load Balancer	<ul style="list-style-type: none"> - Login y registro responden <200 ms en condiciones normales. - Arquitectura soporta escalamiento horizontal (stateless). - Cache para consultas frecuentes de usuarios verificados. 	Meta inicial: 1 k usuarios concurrentes; escalar a 10 k con autoscaling.							
ADB-05	Restricción (Persistencia)	Base de datos relacional obligatoria: PostgreSQL para usuarios y tokens; esquema relacional documentado y versionado por migraciones.	Alta	DB Diagram	PostgreSQL, Migrations (Flyway/TypeORM), Backups	<ul style="list-style-type: none"> - Esquema implementado por migraciones. - Backups diarios automáticos. 	Evaluar uso de Redis como complemento para sesiones o blacklisting de tokens.							

ID	Tipo	Driver (Descripción)	Prioridad	TS relacionadas	Historias / Componentes afectados	Criterios de aceptación	Notas / Riesgos
ADB-06	Restricción (Despliegue)	Contenerización y Cloud: empaquetar Auth y User Service en Docker; despliegue en AWS ECS/EKS (o equivalente).	Alta	—	Docker, Orquestador (ECS/EKS), CI/CD	- Imágenes Docker construidas y almacenadas en registry. - Despliegue a entorno staging reproducible.	Definir infraestructura como código (Terraform/CloudFormation).
ADB-07	Riesgo / Dependencia externa	Integraciones externas: proveedor de correo (para verificación y recuperación).	Alta	TS02	Email Provider, Auth Service	- Integración con proveedor de correo con fallback. - Manejo de errores y límites de cuota.	Riesgo: dependencia de SLA de terceros; plan de degradación si falla envío de emails.

4.3.1.2. Establish Iteration Goal by Selecting Drivers

- **Objetivo de la iteración:** habilitar un onboarding seguro y confiable de usuarios (estudiantes conductores/pasajeros) con autenticación basada en **sesiones** y verificación de condición de **estudiante** antes de permitir funcionalidades núcleo.
- **Drivers funcionales (prioritarios):**
 - Registro de cuenta y acceso (sign-up / sign-in / sign-out).
 - Gestión básica de perfil de usuario.
 - Verificación de estatus de estudiante (Student Beans) con fallback manual (correo institucional + carné).
- **Atributos de calidad (QA) que guían el diseño:**
 - **Seguridad:** sesiones seguras (cookies HttpOnly/Secure), políticas de bloqueo ante fuerza bruta.
 - **Disponibilidad:** servicio de autenticación estable en horas pico de ingreso/salida.
 - **Usabilidad:** flujo de registro/inicio sencillo y claro; mensajes de error comprensibles.
 - **Mantenibilidad:** separación de responsabilidades (auth, perfiles, verificación), contratos claros.
- **Restricciones:**
 - Autenticación **por sesiones** (no tokens).
 - Verificación de estudiante con **Student Beans**.
 - Persistencia relacional para usuarios; caché/almacén de sesión centralizado.
- **Criterios de aceptación de la iteración:**
 - Usuario puede registrarse, iniciar/cerrar sesión y actualizar datos básicos.
 - Estado “estudiante verificado” se refleja en el perfil tras validación (o flujo manual).
 - Cookies de sesión con banderas seguras; protección CSRF activa.
 - Logs/auditoría mínimos para accesos y cambios de perfil.

4.3.1.3 Choose One or More Elements of the System to Refine

- **Contenedores / servicios a refinar:**
 - **Auth Service (Sesiones):** creación/validación/invalidación de sesión; políticas de duración y renovación.
 - **User/Profile Service:** alta de usuario, lectura/edición de perfil, estado de verificación.
 - **Verification Adapter (Student Beans):** integración API, manejo de callbacks/estados y fallback manual.
 - **API Gateway / Edge:** rutas públicas/privadas, redirecciones post-login, rate-limiting para intentos de login.
- **Integraciones externas:**
 - **Student Beans** (verificación), **correo institucional** (fallback), **servicio de email** para confirmaciones/recuperación.
- **Políticas transversales a definir:**
 - **Cookies de sesión:** HttpOnly, Secure.
 - **CSRF:** token por sesión/form.
 - **Password policy:** complejidad, hashing, rotación/recuperación.
 - **Account lockout / throttling:** ante intentos fallidos consecutivos.
 - **Auditoría y logging:** altas, accesos, cambios de perfil.
- **Alcance explícitamente fuera de esta iteración (para próximas):**
 - Publicación/búsqueda de viajes, chat y notificaciones; se habilitan solo para cuentas con sesión válida y estado verificado.

4.3.1.4: Choose One or More Design Concepts That Satisfy the Selected Drivers

Driver Seleccionado	Concepto de Diseño	Justificación	Historias Relacionadas
Seguridad	Autenticación multifactor (MFA)	Refuerza la protección de cuentas de estudiantes y conductores, garantizando confianza y privacidad.	US02, US07, US08
Escalabilidad	Microservicios	Divide el sistema en servicios autónomos por funcionalidad. Permite escalar la solución por demanda.	US01, US06, US10, US11, US13, US19, US20, US23
Disponibilidad	API Gateway	Centraliza la gestión de accesos, autenticación y balanceo de carga. Mejora la disponibilidad y la seguridad del sistema.	US01, US03, US07, US10, US20
Seguridad	Validación en el dominio	Aplica reglas de negocio directamente en entidades y objetos de valor.	US01, US17, US18, US19, US10
Usabilidad	Modelo-Vista-Controlador (MVC) en el cliente	Separa presentación, interacción y datos en la interfaz.	US01, US03, US05, US06, US09, US13, US14

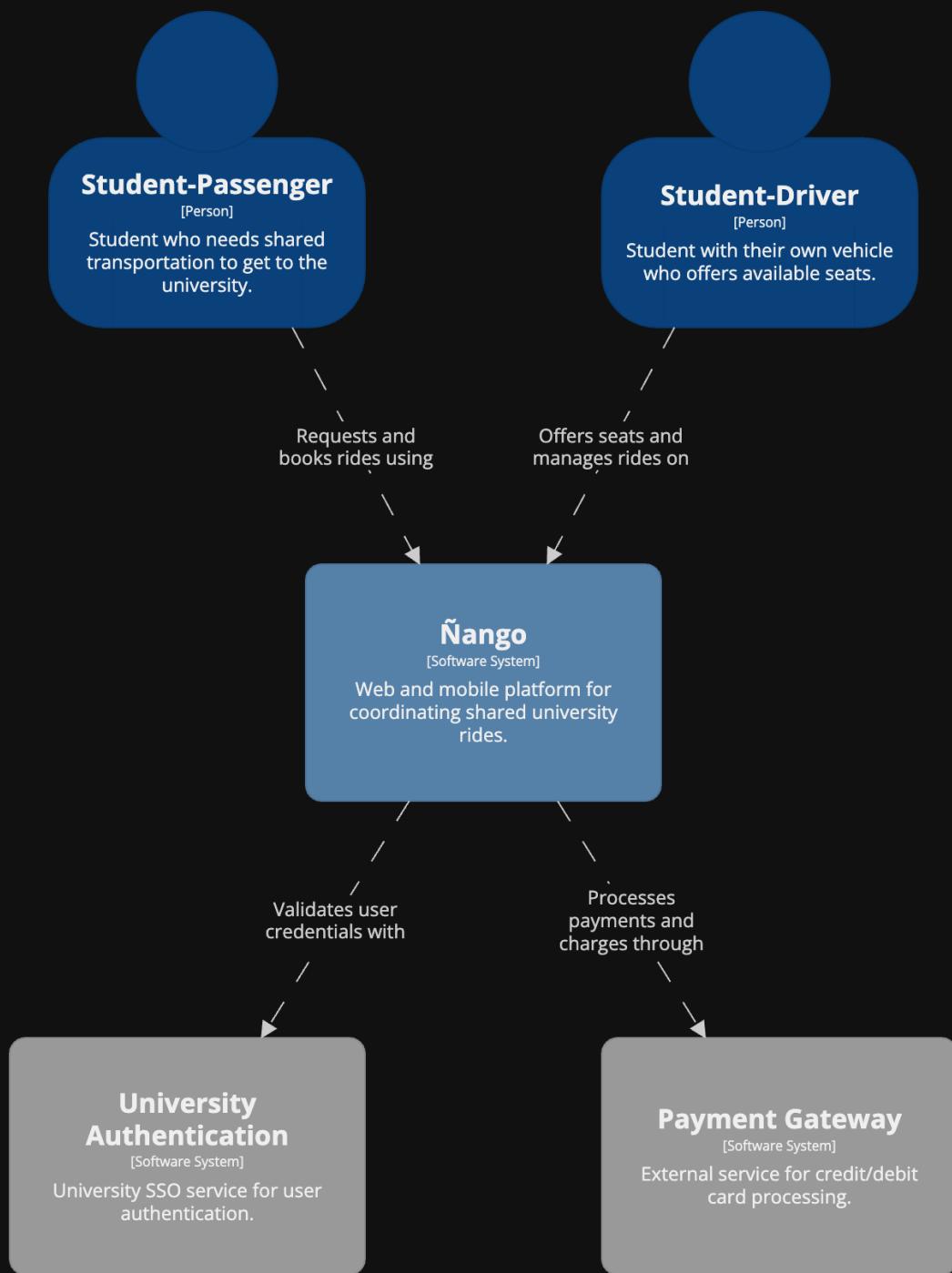
4.3.1.5 Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces

Elemento Arquitectónico	Responsabilidades	Interfaces
API Gateway	<ul style="list-style-type: none"> - Punto único de entrada para clientes - Gestión de autenticación, autorización y balanceo - Enrutamiento hacia microservicios 	<code>POST /auth/login</code> <code>POST /auth/register</code> <code>GET /routes</code> <code>POST /payments</code>
Microservicio de Gestión de Usuarios	<ul style="list-style-type: none"> - Registro, autenticación y manejo de perfiles - Validación de identidades con MFA 	<code>GET /users/{id}</code> <code>PUT /users/{id}</code> <code>POST /users/verify</code>
Front-End MVC	<ul style="list-style-type: none"> - Interfaz para estudiantes y conductores - Visualización de rutas, pagos y perfiles - Comunicación con API Gateway 	Pantallas: Login/Register, Mis Rutas, Mis Viajes, Pagos, Perfil Comunicación vía REST

4.3.1.6 Sketch Views (C4 & UML) and Record Design Decisions

C4 Diagram

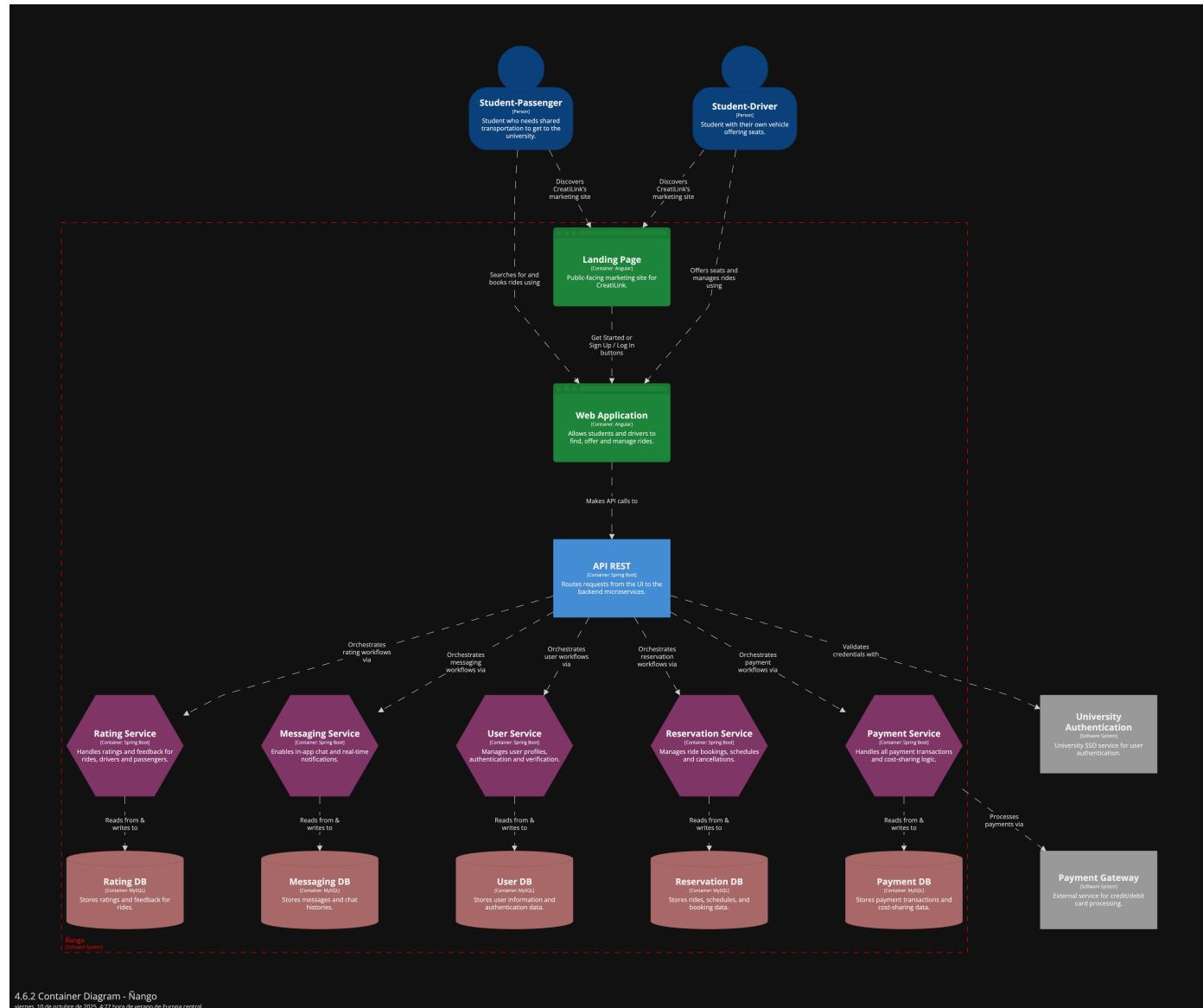
Context Diagram



4.6.1 System Context Diagram - Ñango

viernes, 10 de octubre de 2025, 4:22 hora de verano de Europa central

Container Diagram

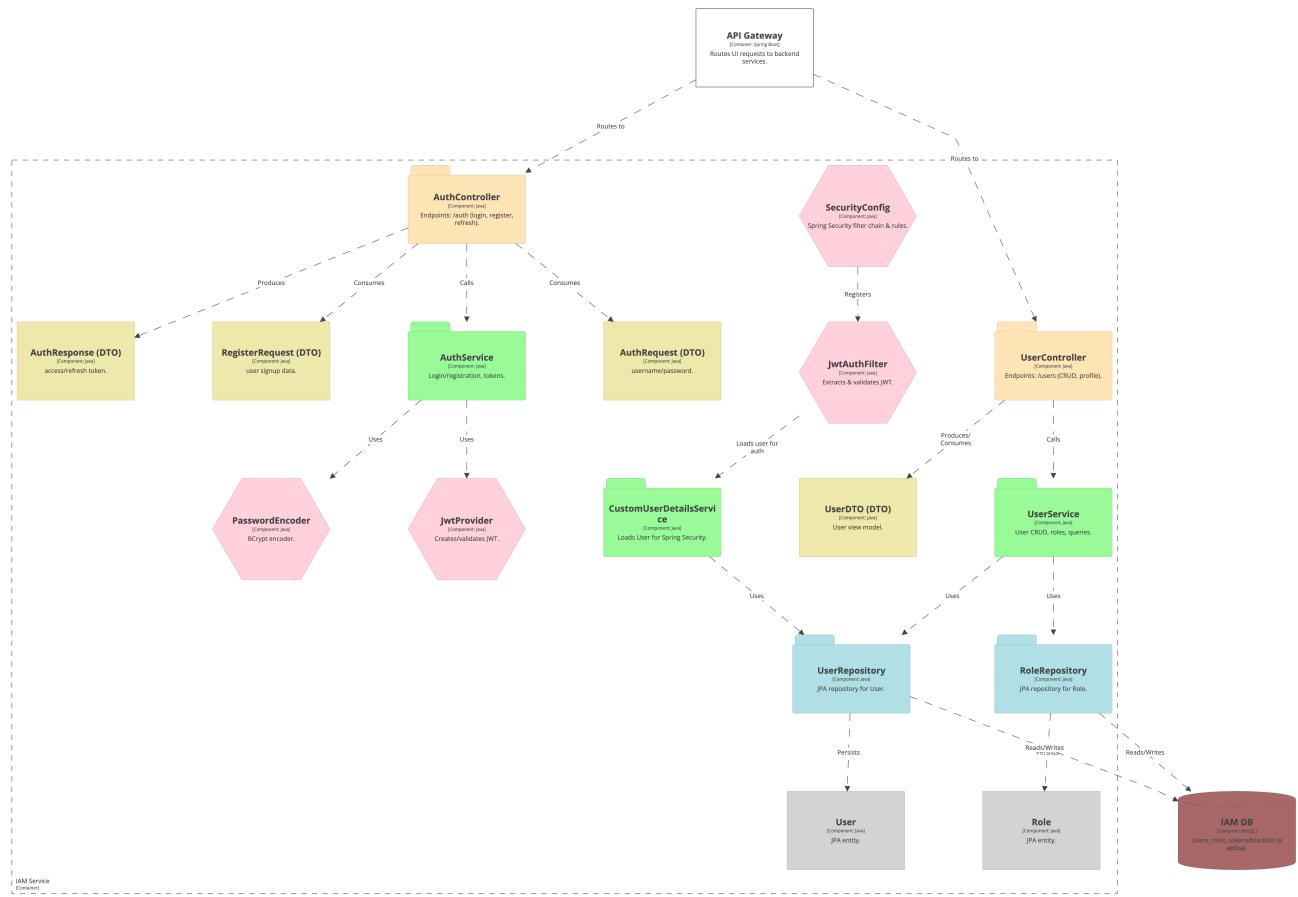


4.6.2 Container Diagram - Nango

viernes, 10 de octubre de 2025, 4:27 hora de verano de Europa central

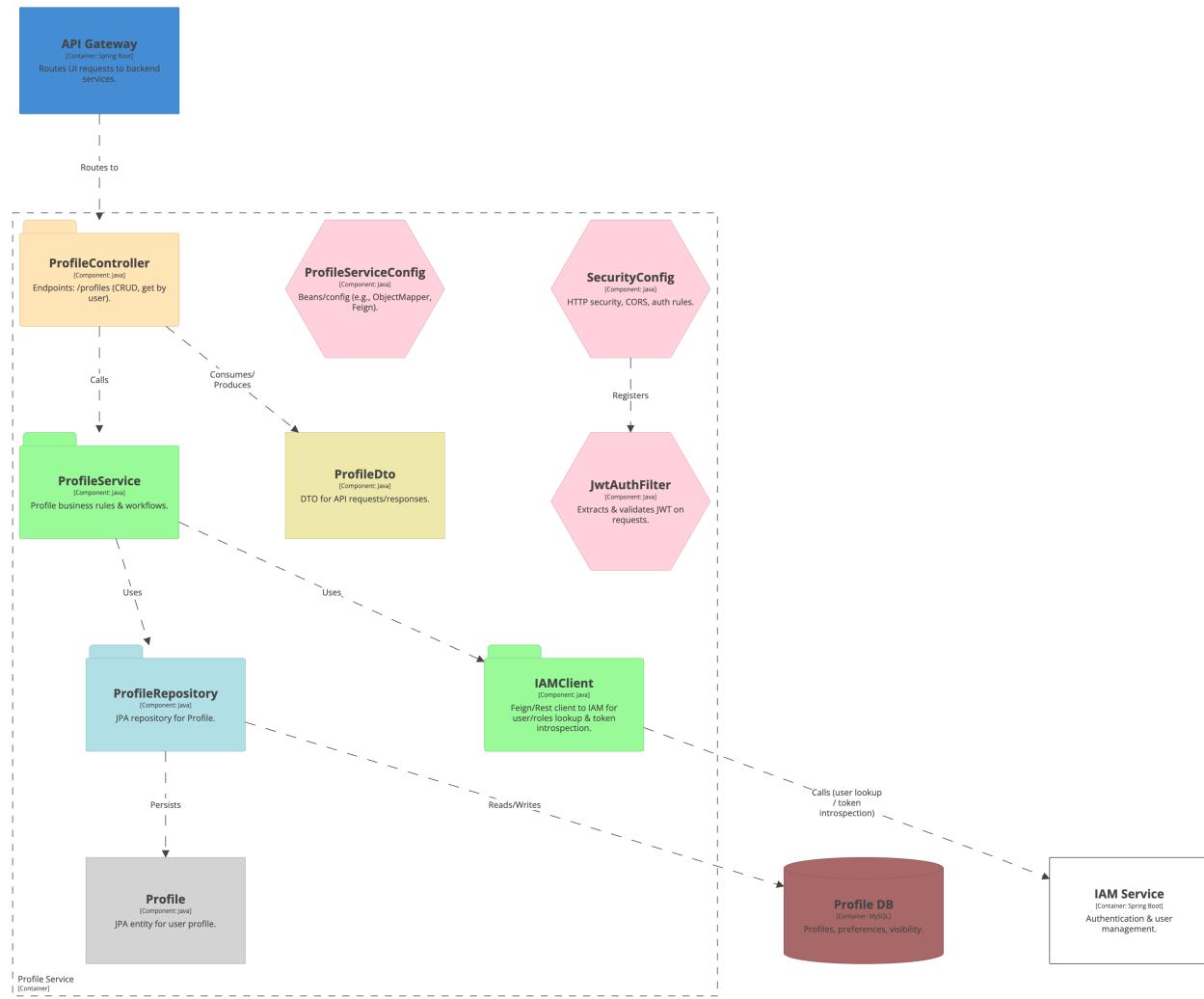
Component Diagram

Iam Service



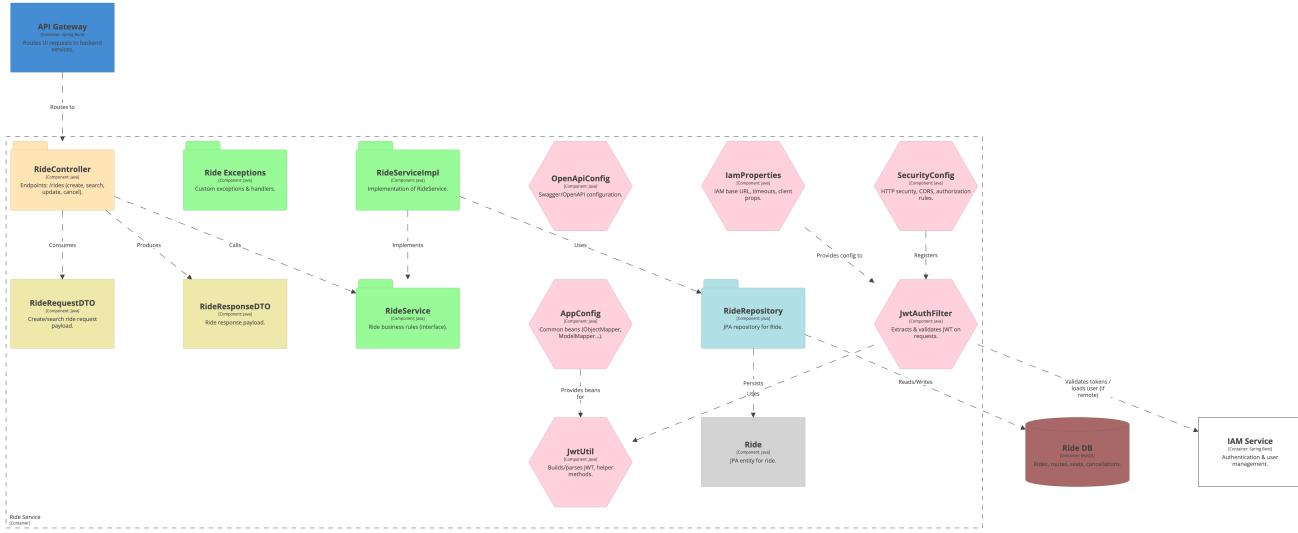
4.6.3 Component Diagram – IAM Service (Nango)
viernes, 10 de octubre de 2025, 4:37 hora de verano de Europa central

Profile Service



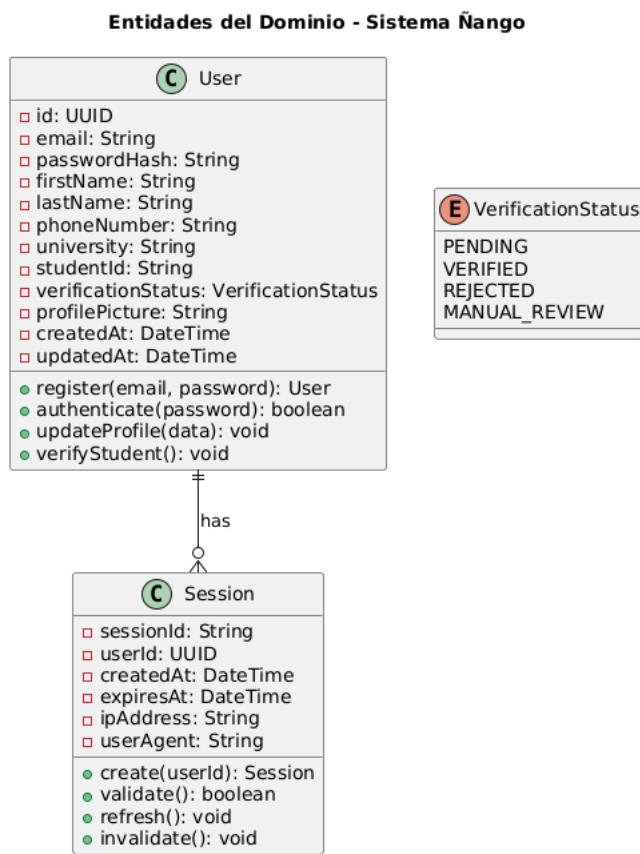
4.6.3 Component Diagram – Profile Service (Nango)
viernes, 10 de octubre de 2025, 4:40 hora de verano de Europa central

Ride Service



4.6.3 Component Diagram – Ride Service (Nango)
viernes, 10 de octubre de 2025, 4:40 hora de verano de Europa central

Uml diagram



4.3.1.7. Analysis of Current Design and Review Iteration Goal (Kanban Board)

En esta iteración se utilizó un tablero Kanban para organizar y dar seguimiento a las tareas asociadas al backlog de diseño de autenticación y gestión de usuarios. El tablero permitió visualizar el estado de cada issue en columnas de To Do, In Progress, In Review y Done, facilitando la coordinación del equipo y la trazabilidad del avance.

A continuación, se muestra una captura del Kanban en GitHub Projects, donde se incluyen issues relacionadas con la creación del flujo de registro e inicio de sesión por sesiones, integración con Student Beans para verificación de estudiantes, configuración de cookies seguras y pruebas básicas de flujo de usuario.

The Kanban board displays the following tasks:

- Todo Column (3 / 5 Estimate: 0):**
 - docs #60: 4.3. ADD Iterations
 - docs #61: 4.3.1. Iteration 1: Autenticación y Gestión de Usuarios
 - docs #69: 4.3.2. Iteration 2: Gestión de Viajes y Solicitudes
 - docs #77: 4.3.3. Iteration 3: Notificaciones y Comunicación
 - docs #70: 4.3.2.1. Architectural Design Backlog 2
 - docs #71: 4.3.1.7. Analysis of Current Design and Review Iteration Goal (Kanban Board)
- In Progress Column (4 / 5 Estimate: 0):**
 - docs #58: 4.2.4. Constraints
 - docs #67: 4.3.1.6. Sketch Views (C4 & UML) and Record Design Decisions
 - docs #46: 4.1. Design Concepts, ViewPoints & ER Diagrams
 - docs #50: 4.1.4. Approach Driven ViewPoints Diagrams
 - docs #54: 4.2. Architectural Drivers
 - docs #59: 4.2.5. Architectural Concerns
- Done Column (15 Estimate: 0):**
 - docs #52: 4.1.6. Design Patterns
 - docs #47: 4.1.1. Principles Statements
 - docs #29: 3.4. Product Backlog
 - docs #65: 4.3.1.4. Choose One or More Design Concepts That Satisfy the Selected Drivers
 - docs #66: 4.3.1.5. Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces
 - docs #64: 4.3.1.6. Sketch Views (C4 & UML) and Record Design Decisions

CAPÍTULO V: Product Implementation, Validation & Deployment

5.1 Testing Suites & General Patterns

El éxito de cualquier aplicación compleja, particularmente aquellas desarrolladas bajo una arquitectura de microservicios, depende en gran medida de la correcta implementación de pruebas y patrones. En este capítulo se detallan los conjuntos de pruebas y patrones implementados en el backend para garantizar la calidad,

seguridad y rendimiento del sistema. El enfoque principal ha sido asegurar que la plataforma cumpla con los altos estándares de escalabilidad, seguridad, disponibilidad y experiencia de usuario definidos.

5.1.1 Backend Application Core Testing Suite

5.1.2 Pattern Based Backend Application(s)

Uso del Patrón Observer en el Microservicio de Seguimiento de Viajes

En Ñango, el patrón **Observer** se aplica en el microservicio de **seguimiento de viajes** (Ride Tracking). Su función es notificar automáticamente a otros servicios y a la interfaz cuando se produce un cambio en el estado de un viaje, por ejemplo, cuando un viaje pasa de "pendiente" a "en curso" o "finalizado".

El sistema publica eventos internos que son escuchados por otros microservicios, como el de **reservas**, que debe actualizar la disponibilidad de asientos, o el de **mensajería**, que envía notificaciones en tiempo real a los usuarios.

Ventajas principales del patrón Observer en Ñango:

- **Descentralización de la lógica:** cada servicio reacciona de forma independiente ante los cambios, sin depender directamente del servicio emisor.
- **Escalabilidad y modularidad:** nuevos servicios pueden suscribirse a los eventos sin modificar el código existente.
- **Experiencia en tiempo real:** los cambios en el estado de los viajes se reflejan instantáneamente en la aplicación web.

Uso del Patrón Singleton en el Microservicio de Autenticación (IAM)

El patrón **Singleton** se utiliza en el microservicio de **autenticación y autorización** (IAM Service), encargado de gestionar los tokens de seguridad y validación de identidad de los usuarios. Este patrón garantiza que exista una única instancia responsable de generar, validar y renovar los **JSON Web Tokens (JWT)**, asegurando así la consistencia y eficiencia del sistema.

Este enfoque centraliza la administración de credenciales, evitando duplicaciones en los demás microservicios y reduciendo el riesgo de errores en la validación. Además, permite controlar el ciclo de vida de los tokens y facilita la auditoría de accesos en la plataforma.

Ventajas principales del patrón Singleton en Ñango:

- **Control centralizado de la seguridad:** un único componente gestiona todos los tokens JWT.
- **Mayor seguridad:** el acceso a los recursos se regula desde un solo punto de control.
- **Mantenimiento simplificado:** la lógica de autenticación no se replica en otros servicios.

Uso del Patrón Factory Method en la Creación de Recursos Dinámicos

En la plataforma Ñango, el patrón **Factory Method** se aplica en los microservicios que gestionan la creación de recursos dinámicos, como **viajes, reservas, usuarios o pagos**. Este patrón permite instanciar diferentes tipos de objetos en función de las solicitudes del usuario sin depender de clases concretas.

Por ejemplo, el sistema puede crear distintos tipos de viajes (inmediatos, programados o recurrentes) a partir de las necesidades del estudiante o conductor. La lógica de creación se encuentra encapsulada, lo que facilita la extensión del sistema sin alterar el código ya existente.

Ventajas principales del patrón Factory Method en Ñango:

- **Flexibilidad en la creación de objetos:** se generan instancias dinámicamente según las reglas de negocio.
- **Facilidad de mantenimiento:** la lógica de creación está centralizada y aislada.
- **Extensibilidad:** permite agregar nuevos tipos de viajes o usuarios sin modificar la estructura actual.

5.1.3 Pattern Based Custom Software Library

En esta etapa del proyecto Ñango aún no se ha implementado una librería de software compartida entre microservicios. Sin embargo, se ha definido el diseño para su futura incorporación, con el objetivo de estandarizar la comunicación, reducir la duplicación de código y favorecer la reutilización de componentes comunes en toda la arquitectura.

5.1.4 Framework Pattern Driven Refactoring Report

Identificación de oportunidades de refactorización

Durante el proceso de mejora continua del sistema Ñango, se identificaron varias áreas con oportunidades de refactorización, especialmente relacionadas con la duplicación de lógica y la gestión del estado de los recursos. Para optimizar el rendimiento y la mantenibilidad, se introdujeron diversos patrones de diseño que mejoraron la modularidad, la escalabilidad y la cohesión del sistema.

Las principales áreas detectadas fueron:

- **Código redundante** en la lógica de cálculo de precios y validaciones.
- **Gestión de estados** de los viajes y reservas con estructuras condicionales complejas.
- **Altos tiempos de respuesta** en momentos de alta concurrencia.

Aplicación de patrones de refactorización

Patrón Strategy

Se utilizó el patrón **Strategy** para aislar la lógica variable, por ejemplo en el cálculo de tarifas dentro del servicio de pagos o en la búsqueda y coincidencia de viajes dentro del servicio de transporte.

Gracias a este patrón, se pueden cambiar o añadir estrategias de negocio sin modificar el flujo principal de la aplicación.

Beneficio: modularización de reglas de negocio, reducción de código duplicado y mayor facilidad para añadir nuevas estrategias de precios o asignación.

Patrón State

Se implementó el patrón **State** en la gestión del ciclo de vida de los viajes y las reservas.

Cada estado (creado, confirmado, en curso, completado o cancelado) tiene su propio comportamiento y transiciones válidas, lo que evita condicionales extensos y mejora la claridad del código.

Beneficio: mayor control sobre las transiciones, reducción de errores y un flujo más predecible en los procesos de reserva y seguimiento de viajes.

Patrón Proxy

El patrón **Proxy** se incorporó en el **API Gateway** y en los microservicios de lectura intensiva, como el de viajes o perfiles, para gestionar el balanceo de carga y la comunicación entre servicios.

Gracias a este patrón, se añadió soporte de caché y tolerancia a fallos, lo que permite servir datos desde cachés locales cuando un servicio no está disponible o se encuentra saturado.

Beneficio: reducción del tiempo de respuesta y mejora en la resiliencia de la plataforma, incluso con más de 10,000 usuarios simultáneos.

Resultados de la refactorización

- **Reducción del tiempo de respuesta:** la combinación del patrón Proxy y el uso de cachés locales disminuyó los tiempos de respuesta de las consultas más utilizadas, cumpliendo con el objetivo de mantener respuestas críticas por debajo de los 3 segundos.
- **Mayor mantenibilidad:** los patrones Strategy y State simplificaron el código, permitiendo modificaciones sin afectar el comportamiento global.
- **Escalabilidad mejorada:** la modularización de los servicios y la introducción de eventos (Observer) permitieron que Ñango manejara grandes volúmenes de solicitudes con un consumo de recursos optimizado.

5.2. Software Configuration Management

5.2.1. Software Development Environment Configuration

1. Project Management:

- **Trello:** Gestión visual de tareas por Sprint, seguimiento de backlog y prioridades. <https://trello.com/home>

2. Requirements Management:

- **Structurizr:** Diagramas de arquitectura de software C4. <https://structurizr.com/>
- **Lucidchart:** Diagrama de clases. <https://www.lucidchart.com/pages/es>

3. Product Design:

- **Figma:** Diseño de interfaces y prototipos interactivos <https://www.figma.com/>

4. Software Development:

- **Visual Studio Code:** Editor de código para el desarrollo del informe. <https://code.visualstudio.com/>
- **IntelliJ IDEA:** Entorno de desarrollo del backend. <https://www.jetbrains.com/es-es/idea/download/?section=windows>
- **Java + Maven:** Lenguaje de programación para el desarrollo del backend con gestión de dependencias <https://maven.apache.org/>
- **Docker:** Contenerización de servicios. <https://www.docker.com/>
- **MySQL:** Base de datos. <https://www.mysql.com/>

5. Software Testing:

- **Spring Boot Test:** Framework de pruebas integrado en Spring Boot para pruebas unitarias, de integración y de contexto. Utilizado en microservicios desarrollados en Java. <https://spring.io/projects/spring-boot>

6. Software Deployment:

- **GitHub Actions:** Automatización de CI/CD: build, test, deploy. <https://github.com/features/actions>
- **Dokploy:** Plataforma de despliegue basada en Docker Compose, con soporte para múltiples servidores y monitoreo en tiempo real. <https://dokploy.com/es>

7. Software Documentation:

- **Swagger:** Documentación de APIs RESTful de microservicios. <https://swagger.io/>

5.2.2. Source Code Management

Para el control de versiones y seguimiento de modificaciones en los productos de software se utilizará GitHub como plataforma principal. Cada microservicio y componente de la solución ÑanGo cuenta con su propio repositorio, incluyendo tanto el código fuente como los archivos de prueba (.feature).

1. **Repositorios:**

- **Repositorio IAM Service:** <https://github.com/ASI0657-2520-faw-6327/iam-service>
- **Repositorio Profile Service:** <https://github.com/ASI0657-2520-faw-6327/profile-service>
- **Repositorio Ride Service:** <https://github.com/ASI0657-2520-faw-6327/ride-service>

2. **GitFlow Workflow:**

Para el desarrollo del Web Services se implementa el modelo GitFlow propuesto por Vincent Driessen, adaptado a un entorno de microservicios y despliegue continuo. Las ramas son:

- **main:** Rama principal para producción
- **develop:** Rama de desarrollo
- **feature:** Rama para desarrollo de nuevas funcionalidades
- **fix:** Rama para corrección de errores
- **release:** Rama para despliegue de versiones estables

3. **Conventional Commits:** Se utilizarán **Conventional Commits** para facilitar el versionado semántico.

- **feat:** Nueva funcionalidad
- **fix:** Corrección de errores
- **test:** Pruebas agregadas o modificadas
- **docs:** Documentación

4. **Source Code Style Guide & Coding Conventions:** Para mantener la coherencia y legibilidad del código, el equipo adopta las siguientes guías y convenciones:

• **Nomenclatura:**

- Todos los nombres de variables, funciones, clases y archivos se escriben en inglés.
- Se aplica camelCase para variables y funciones, PascalCase para clases, y snake_case para nombres de archivos.

5.3 Microservices Implementation

5.2.1 Sprint 1

5.2.1.1 Sprint Backlog 1

User Story ID	User Story Title	Work-Item ID	Work-Item Title	Description	Estimation (Hours)	Assigned To	Status (To-do / In-Process / To-Review / Done)
US01	Registro de nueva cuenta	SB1-01	Crear estructura base del proyecto	Configurar arquitectura DDD y dependencias iniciales (Spring Web, JPA, Security).	6	Anderson Gamarra	Done
US01	Registro de nueva cuenta	SB1-02	Implementar endpoints de registro y autenticación	Desarrollar controladores, servicios y repositorios con JWT y validación de credenciales.	8	Vicente Quijandria	Done
US03	Inicio de sesión	SB1-03	Configurar seguridad con Spring Security	Proteger endpoints y manejar roles de usuario.	6	Anderson Gamarra	Done
US05	Edición de perfil	SB1-04	CRUD de perfiles	Implementar endpoints REST para crear, editar y consultar perfiles.	7	Pedro Nanfuñay	Done
US08	Verificación universitaria	SB1-05	Mock API Student Beans	Simular validación universitaria mediante API externa (mock).	5	Miguel Hallasi	Done
US19	Publicación de viajes	SB1-06	Endpoint publicar viajes	Crear endpoint para registrar viajes (origen, destino, horario, cupos).	7	Samuel Valera	Done
US09	Búsqueda de viajes	SB1-07	Endpoint búsqueda de viajes	Implementar búsqueda filtrada por fecha, origen y destino.	6	Samuel Valera	Done

User Story ID	User Story Title	Work-Item ID	Work-Item Title	Description	Estimation (Hours)	Assigned To	Status (To-do / In-Process / To-Review / Done)
US20	Gestión de solicitudes	SB1-08	Endpoint solicitudes de viaje	Permitir al conductor aceptar o rechazar solicitudes de pasajeros.	8	Samuel Valera	In-Process
—	—	SB1-09	Configurar bases de datos MySQL	Crear esquemas <code>iam_db</code> , <code>profile_db</code> , <code>ride_db</code> y entidades con relaciones.	5	Equipo Backend	Done
—	—	SB1-10	Crear archivo docker-compose.yml	Orquestar microservicios y bases de datos en contenedores.	6	Anderson Gamarra	Done
—	—	SB1-11	Implementar pruebas unitarias con JUnit y Mockito	Validar casos de uso y lógica de negocio.	8	Vicente Quijandria / Pedro Nanfuñay	Done
—	—	SB1-12	Configurar comunicación REST entre servicios	Conectar IAM, Profile y Ride.	7	Anderson Gamarra	In-Process
—	—	SB1-13	Documentar endpoints con Swagger	Generar documentación automática de las APIs.	5	Miguel Hallasi	To-Review
—	—	SB1-14	Crear imágenes Docker individuales	Crear <code>Dockerfile</code> para cada microservicio.	6	Samuel Valera	Done
—	—	SB1-15	Pruebas de integración con Docker Compose	Validar persistencia y comunicación entre servicios en entorno local.	6	Equipo Backend	Done
—	—	SB1-16	Actualizar README técnico	Documentar resultados, endpoints y configuración del sprint.	4	Vicente Quijandria	Done

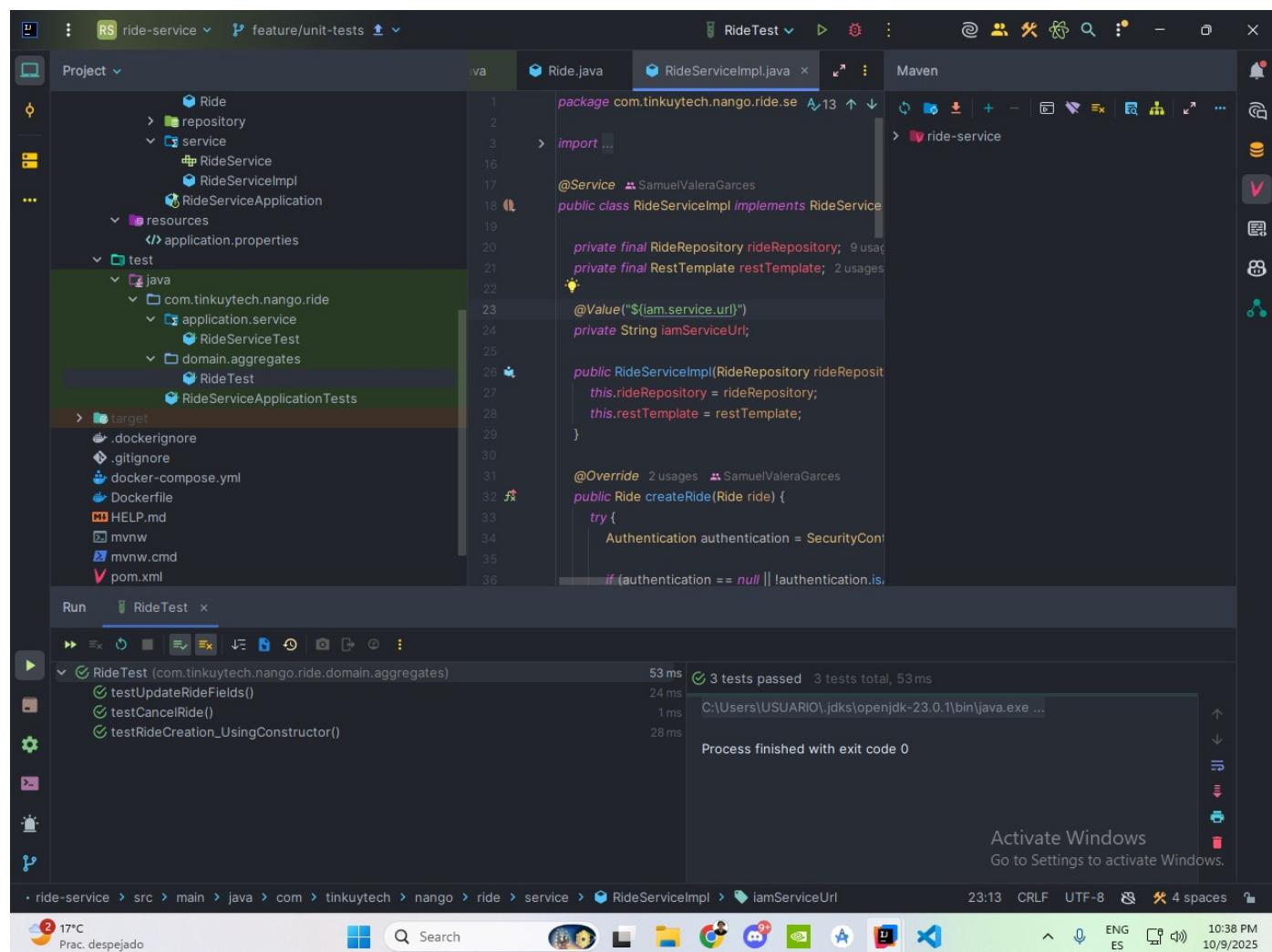
5.2.1.2 Development Evidence for Sprint Review

Repository / Service	Branch	Commit ID	Type / Message	Commit Body (Descripción resumida)	Date (dd/mm/yyyy)
Landing-page	prod	46bbe04	chore: initialize project structure	Creación inicial del repositorio y estructura base del proyecto.	07/10/2025
Landing-page	prod	a4a7b55	fix: Nginx production configuration	Corrección en la configuración de Nginx para entorno productivo.	07/10/2025
Landing-page	prod	4703af8	chore: update production container name	Actualización del nombre del contenedor para despliegue en producción.	07/10/2025
Landing-page	prod	f288caf	ci: add deploy job to pipeline	Incorporación de un job de despliegue en el pipeline CI/CD.	07/10/2025
Landing-page	prod	6b7a542	fix: update Docker image tags	Actualización de etiquetas de imágenes Docker para versión y <code>latest</code> .	07/10/2025
Landing-page	prod	9a079e8	chore: add Docker Compose configuration	Se añadió archivo <code>docker-compose.yml</code> para orquestar servicios.	07/10/2025
Landing-page	prod	e96572f	ci: add verify and build pipeline steps	Configuración de verificación y build automáticos en el pipeline.	07/10/2025
Profile Service	profile	b62a41e	feat(app): setup FastAPI project	Se configuró la aplicación base con FastAPI y dependencias principales.	08/10/2025
Profile Service	profile	ca84b23	feat(domain): add user and profile entities (DDD)	Se implementó la capa de dominio con entidades <code>User</code> y <code>Profile</code> bajo patrón DDD.	08/10/2025
Profile Service	profile	2c9db45	feat(api): add profile CRUD endpoints	Se añadieron endpoints REST para crear, leer, actualizar y eliminar perfiles.	09/10/2025
Profile Service	profile	dc1a8e7	test: add unit tests for profile use cases	Se implementaron pruebas unitarias para los casos de uso del perfil.	09/10/2025
Ride Service	ride	03db29b	feat(api): add POST endpoint for ride creation	Implementación inicial del endpoint POST <code>/rides</code> para crear viajes.	08/10/2025
Ride Service	ride	7bccd94	feat(api): extend CRUD endpoints	Se añadieron endpoints DELETE y GETALL para operaciones completas de viajes.	09/10/2025
Ride Service	ride	5754776	chore: update .gitignore	Se actualizó archivo <code>.gitignore</code> para exclusión de archivos locales.	09/10/2025
IAM Service	main	ae2494f	chore: initialize IAM service	Configuración inicial del servicio de autenticación.	08/10/2025

Repository / Service	Branch	Commit ID	Type / Message	Commit Body (Descripción resumida)	Date (dd/mm/yyyy)
IAM Service	main	75835b5	chore: add FastAPI dependency	Adición de dependencia principal de FastAPI.	08/10/2025
IAM Service	main	0e9a584	chore: setup project structure	Creación de estructura base: app/ , routes/ , core/ , etc.	08/10/2025
IAM Service	main	49ab2f9	chore: update dev start steps	Ajuste en pasos de ejecución para entorno de desarrollo.	08/10/2025
IAM Service	main	73c526b	feat: release v0.1 of IAM service	Versión funcional inicial con endpoints básicos de autenticación.	09/10/2025
IAM Service	main	643a047	ci: add deploy webhook for Dokploy	Configuración de webhook de despliegue automático a Dokploy.	09/10/2025

5.2.1.3 Testing Suite Evidence for Sprint Review

Ride Service



The screenshot shows the IntelliJ IDEA interface with the following details:

- Project Structure:** The left sidebar shows the project structure for `ride-service` under `feature/unit-tests`. It includes packages for `Ride`, `repository`, `service` (containing `RideService`, `RideServiceImpl`, and `RideServiceApplication`), `resources` (with `application.properties`), and `test` (containing `java` (with `com.tinkuytech.nango.ride` package) and `resources`). A specific test class, `RideServiceApplicationTests`, is selected in the tree.
- Code Editor:** The main editor window displays the Java code for `RideServiceImpl.java`. The code implements the `RideService` interface, using `RideRepository` and `RestTemplate`. It includes annotations like `@Value("${iam.service.url}")` for the IAM service URL.
- Run Tab:** The bottom tab bar is set to `RideTest`. The run results show three tests passed: `testUpdateRideFields()`, `testCancelRide()`, and `testRideCreation_UsingConstructor()`. The total execution time was 53 ms.
- Status Bar:** The status bar at the bottom provides system information: `17°C Prac. despejado`, `Search`, `23:13 CRLF UTF-8`, `Activate Windows Go to Settings to activate Windows.`, `ENG ES`, `10:38 PM 10/9/2025`.

Profile Service

The screenshot shows a Java project structure in the left panel. The `ProfileTest` class under `ProfileServiceApplicationTests` is selected. The right panel displays the source code for `ProfileServiceApplicationTests.java` and `ProfileServiceTest.java`. The bottom panel shows the test results in the terminal, indicating 8 tests passed and 1 ignored.

```

1 package com.tinkuytech.nango.profile.domain.aggrega
2
3 import com.tinkuytech.nango.profile.model.Profile;
4 import org.junit.jupiter.api.Test;
5 import org.junit.jupiter.api.DisplayName;
6 import org.junit.jupiter.api.BeforeEach;
7
8 import static org.junit.jupiter.api.Assertions.*;
9
10 @DisplayName("Pruebas del modelo Profile") - flendoh
11 class ProfileTest {
12
13     private Profile profile; 9 usages
14
15     @BeforeEach - flendoh
16     void setUp() {
17         profile = Profile.builder()
18             .id(1L)
19             .fullName("Juan Pérez")
20             .bio("Conductor profesional con 10")
21             .licenseNumber("LIC123456")
22             .rating(4.5f)
23             .userId("user123")
24             .userType("DRIVER")
25             .build();
}

```

Run `java in profile-service`

Test	Time
<default package>	954 ms
Pruebas del modelo Profile	22 ms
Debería crear perfil con patrón builder	20 ms
Debería establecer y obtener propiedades correctamente	1 ms
Debería manejar valores nulos correctamente	1 ms
Debería validar métodos equals y hashCode	1 ms
> ProfileServiceApplicationTests	
Pruebas del Servicio Profile	932 ms
Debería obtener perfil por userId exitosamente	920 ms
Debería obtener todos los perfiles exitosamente	2 ms
Debería lanzar excepción cuando perfil no existe	3 ms
Debería crear perfil exitosamente	7 ms

Tests passed: 8, Ignored: 1 of 9 tests – 954 ms

5.2.1.4 Execution Evidence for Sprint Review

Landing Page

Web App

5.2.1.5 Microservices Documentation Evidence for Sprint Review

Ride Service Documentation

GET /api/rides

Parameters

No parameters

Responses

Curl

```
curl -X 'GET' \
'http://localhost:8083/api/rides' \
-H 'Accept: */*' \
-H 'Authorization: Bearer eyJhbGciOiJIUzI1NiJ9eyJyb2xlcjI6IyJDT05EVUNUT1IXSwic3ViIjoidXNlcnRlc3Q8IiwiaWF0IjoxNzYwMDY2MDkyLCJleHAiOjE3NjAxMDIwOTJ9.eyJ0eXAiOiJKV1QiLCJleHBvcnRpbmcIjoiMjAyMjIwMjIwMjIwIiwidmFsdWVucyI6ImShqqtbmF8'
```

Request URL

```
http://localhost:8083/api/rides
```

Server response

Code	Details
200	Response body <pre>[{ "id": 1, "driverUserId": "anonymousUser", "origin": "Lima", "destination": "Miraflores", "availableSeats": 3, "departureTime": "2025-10-07T15:00:00", "status": "OPEN" }, { "id": 2, "driverUserId": "user12", "origin": "Lima", "destination": "Cusco", "availableSeats": 3, "departureTime": "2025-10-07T09:00:00", "status": "OPEN" }]</pre> <div style="text-align: right;"> Copy Download </div> Response headers <pre>cache-control: no-cache,no-store,max-age=0,must-revalidate connection: keep-alive content-type: application/json date: Fri,18 Oct 2025 03:18:29 GMT expires: 0 keepalive: timeout=60 pragma: no-cache transfer-encoding: chunked x-content-type-options: nosniff x-frame-options: DENY x-xss-protection: 0</pre>

Responses

Code	Description	Links
200	OK	No links

Media type

/

Controls Accept header.

[Example](#) [Value](#) | [Schema](#)

```
[
  {
    "id": 1,
    "driverUserId": "string",
    "origin": "string",
    "destination": "string",
    "availableSeats": 1,
    "departureTime": "2025-10-10T03:18:29.041Z",
    "status": "OPEN"
  }
]
```

Iam Service Documentation

The screenshot shows a REST API documentation interface for a POST request to the endpoint `/api/auth/login`. The request body is required and contains the following JSON:

```
{
  "username": "usertest4",
  "password": "usertest4"
}
```

Below the request body, there are two buttons: **Execute** and **Clear**.

Responses

Curl

```
curl -X 'POST' \
'http://localhost:8081/api/auth/login' \
-H 'Accept: */*' \
-H 'Content-Type: application/json' \
-d '{
  "username": "usertest4",
  "password": "usertest4"
}'
```

Request URL

```
http://localhost:8081/api/auth/login
```

Server response

Code	Details	Links
200	<p>Response body</p> <pre>{ "username": "usertest4", "token": "eyJhbGciOiJIUzI1NiJ9.eyJyb2xlcjI6WyJDT05EVUNUT1IIXSwic3ViIjoidXNlcnRlc3Q0TiwiakF0IjoxNzYwMDY2MDE4LC1leHA1OjE3NjAxMDIwMTI9.BauAHGUlxhyM65XuvD3queRmsucxQ2FAwhbxbr" }</pre> <p>Response headers</p> <pre>cache-control: no-cache,no-store,max-age=0,must-revalidate connection: keep-alive content-type: application/json date: Fri, 10 Oct 2025 03:13:38 GMT expires: 0 keep-alive: timeout=60 pragma: no-cache transfer-encoding: chunked vary: Origin,Access-Control-Request-Method,Access-Control-Request-Headers x-content-type-options: nosniff x-frame-options: DENY x-xss-protection: 0</pre>	

Responses

Code	Description	Links
200	OK	No links

Media type

A dropdown menu showing `*/*`, with a note: Controls Accept header.

Example Value | Schema

```
{}
```

Profile Service Documentation

GET /api/profiles

Parameters

No parameters

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:8082/api/profiles' \
  -H 'accept: */*' \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiJ9eyJyb2xlcyI6IjQD05EVNUUT1IiXSwic3VIIjoidXNlcnRlc3Q0IiwiaWF0IjoxNzYwMDY2MDkyLC1leHAiOjE3NjAxMDIwOTJ9.86ytwPwf3ZUIkXFUzCUzPoMtwp69iPc8ImShqqtbnF8'
```

Request URL

http://localhost:8082/api/profiles

Server response

Code	Details
200	Response body <pre>[{ "id": 1, "fullName": "Juan Perez", "bio": "Me gusta viajar", "licenseNumber": "ABC123", "rating": 0, "userId": "user1", "userType": "PASAJERO" }, { "id": 2, "fullName": "Maria Lopez", "bio": "Disfruto conducir", "licenseNumber": "XYZ789", "rating": 0, "userId": "user2", "userType": "CONDUCTOR" }</pre> <p>Response headers</p> <pre>cache-control: no-cache,no-store,max-age=0,must-revalidate connection: keep-alive content-type: application/json date: Fri,18 Oct 2025 03:16:25 GMT expires: 0 keep-alive: timeout=60 pragma: no-cache transfer-encoding: chunked x-content-type-options: nosniff x-frame-options: DENY x-xss-protection: 0</pre>

Responses

Code	Description	Links
200	OK	No links

Media type

/

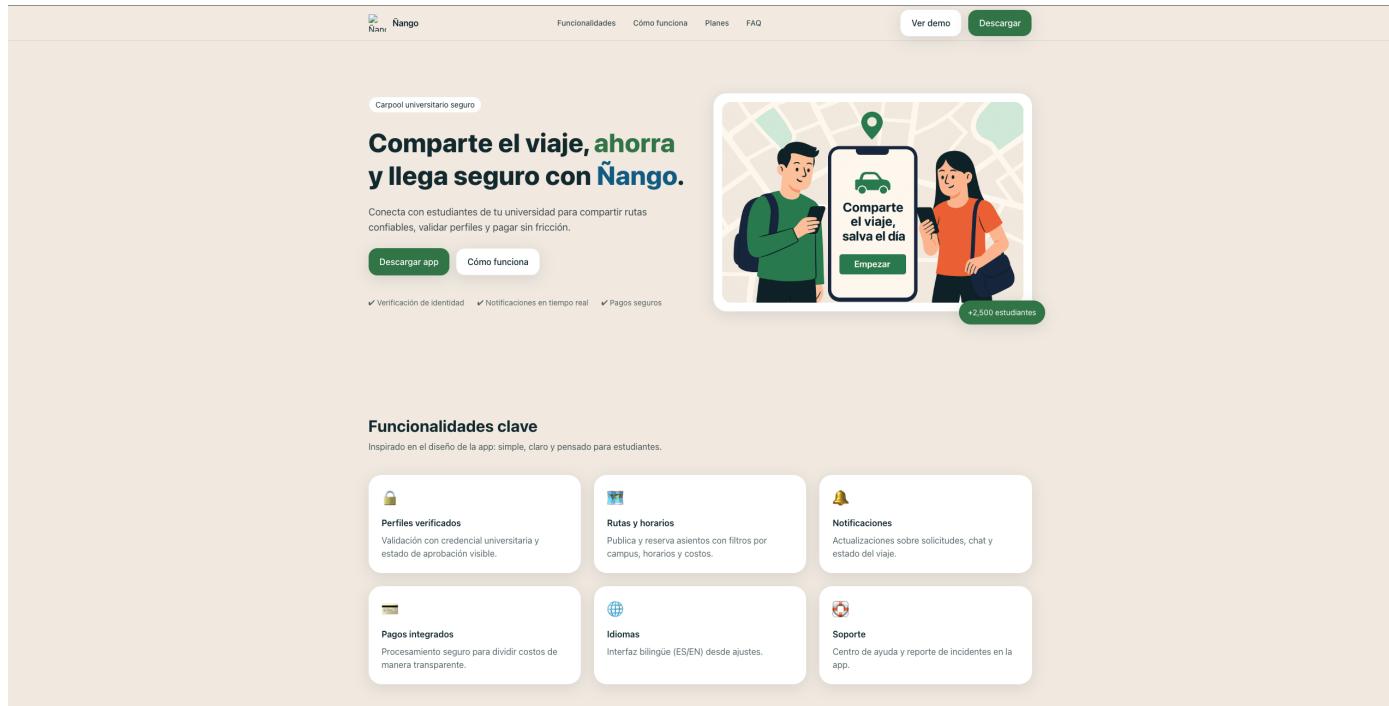
Controls Accept header.

[Example Value](#) | [Schema](#)

```
[ {
    "id": 0,
    "fullName": "string",
    "bio": "string",
    "licenseNumber": "string",
    "rating": 0,
    "userId": "string",
    "userType": "string"
}]
```

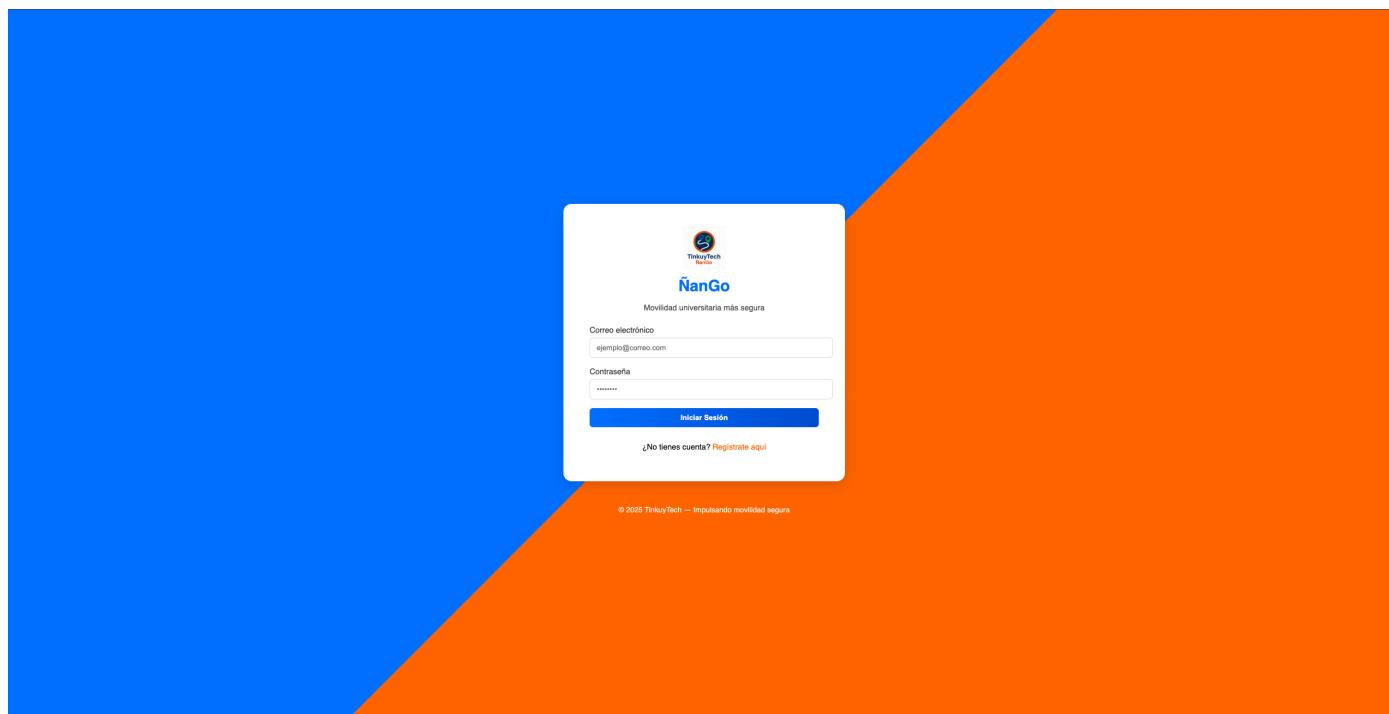
5.2.1.6 Software Deployment Evidence for Sprint Review

Landing Page



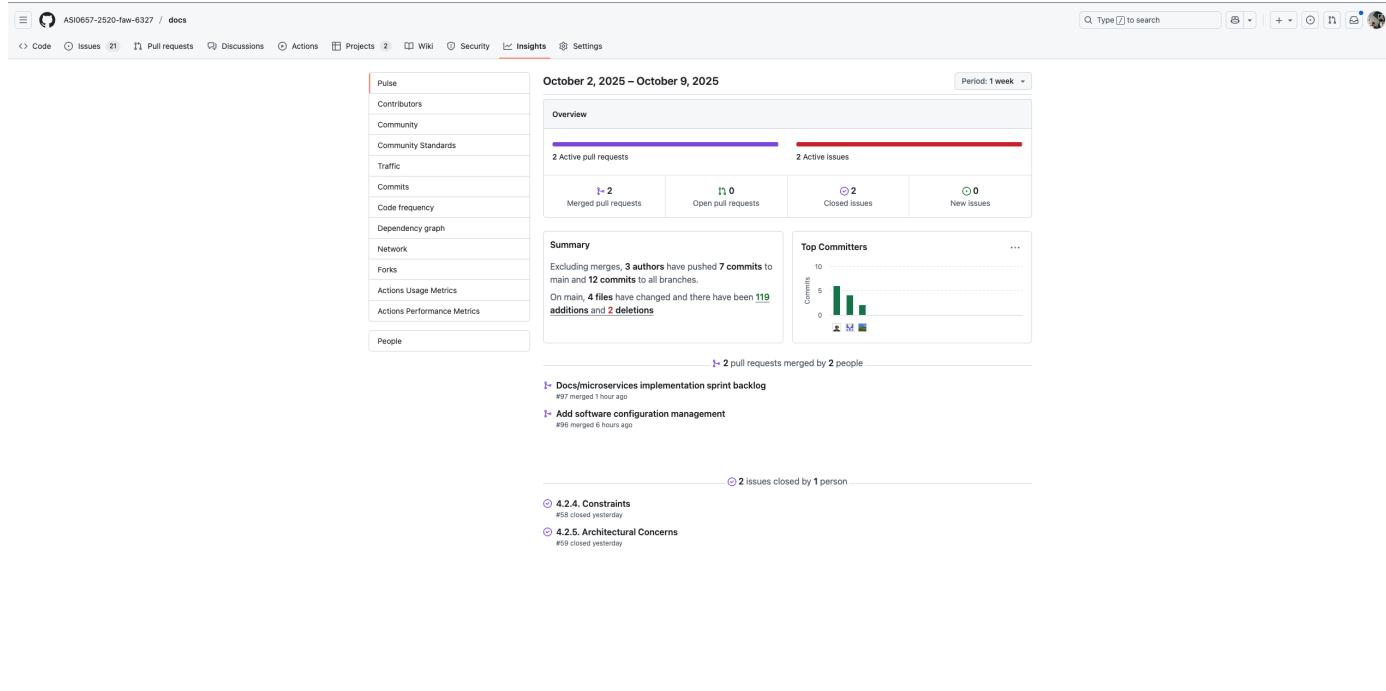
The screenshot shows the homepage of the Ñango app website. At the top, there's a navigation bar with links for 'Funcionalidades', 'Cómo funciona', 'Planes', and 'FAQ'. There are also 'Ver demo' and 'Descargar' buttons. Below the header, a main banner features the text 'Carpool universitario seguro' and 'Comparte el viaje, ahorra y llega seguro con Ñango.' It includes a callout for 'Carpool universitario seguro' and a large graphic showing two students using phones with the app interface. A green button at the bottom left says 'Descargar app' and another one next to it says 'Cómo funciona'. Below the banner, there's a section titled 'Funcionalidades clave' with six cards: 'Perfiles verificados' (verified profiles), 'Rutas y horarios' (routes and schedules), 'Notificaciones' (notifications), 'Pagos integrados' (integrated payments), 'Idiomas' (languages), and 'Soporte' (support). Each card has a small icon and a brief description.

App Web



The screenshot shows the login screen of the Ñango app web. It features a white form on a blue and orange background. The form has fields for 'Correo electrónico' (example@correo.com) and 'Contraseña' (password). Below the fields are 'Iniciar Sesión' (Sign In) and '¿No tienes cuenta? Regístrate aquí' (Don't have an account? Register here). At the bottom, there's a copyright notice: '© 2025 TinkuyTech — Impulsando movilidad segura'.

5.2.1.7 Team Collaboration Insights during Sprint



5.2.1.8 Kanban Board

El Kanban Board del Sprint 1 muestra la organización y seguimiento de las tareas del equipo durante la primera iteración del proyecto. Este tablero permite visualizar el flujo de trabajo dividido en columnas que representan los diferentes estados de las tareas: por hacer, en progreso y completadas.

To Do	In Progress	To Review	Done
+ Añade una tarjeta	SB1-08 – Endpoint solicitudes de viaje [US20] – 8h – Samuel Valera SB1-12 – Configurar comunicación REST entre servicios – 7h – Anderson Gamarra	SB1-13 – Documentar endpoints con Swagger – 5h – Miguel Hallasi + Añade una tarjeta	SB1-10 – Crear archivo docker-compose.yml – 6h – Anderson Gamarra SB1-01 – Crear estructura base del proyecto [US01] – 6h – Anderson Gamarra SB1-02 – Implementar endpoints de registro y autenticación [US01] – 8h – Vicente Quijandria SB1-11 – Implementar pruebas unitarias con JUnit y Mockito – 8h – Vicente Quijandria / Pedro Nanfuhay SB1-03 – Configurar seguridad con Spring Security [US03] – 6h – Anderson Gamarra SB1-04 – CRUD de perfiles [US05] – 7h – Pedro Nanfuhay SB1-09 – Configurar bases de datos MySQL – 5h – Equipo Backend SB1-15 – Pruebas de integración con Docker Compose – 6h – Equipo Backend

Link de Trello: <https://trello.com/invite/b/68d9fec4a886565a738c5578/ATTI5fe0d222c5142993d845a183ced85bc8475F9959/sprint-1>

5.2.2 Sprint 2

5.2.2.1 Sprint Backlog 2

User Story ID	User Story Title	Work-Item ID	Work-Item Title	Description	Estimation (Hours)	Assigned To	Status (To-do / In-Process / To-Review / Done)
---------------	------------------	--------------	-----------------	-------------	--------------------	-------------	--

User Story ID	User Story Title	Work-Item ID	Work-Item Title	Description	Estimation (Hours)	Assigned To	Status (To-do / In-Process / To-Review / Done)
US17	Reporte de problemas o incidentes	SB2-01	Implementar servicio de reportes	Crear endpoints para que los usuarios puedan reportar problemas o incidentes durante un viaje.	8	Vicente Quijandria	Done
---	---	SB2-02	Pruebas unitarias para servicio de reportes	Implementar pruebas unitarias con JUnit y Mockito para el servicio de reportes.	6	Vicente Quijandria	Done
US16	Sistema de mensajería	SB2-03	Implementar servicio de mensajería	Desarrollar un sistema de mensajería para la comunicación entre conductor y pasajeros.	8	Pedro Nanfuñay	Done
---	---	SB2-04	Pruebas unitarias para servicio de mensajería	Implementar pruebas unitarias con JUnit y Mockito para el servicio de mensajería.	6	Pedro Nanfuñay	Done
US01, US03, US05	Gestión de Cuentas y Perfiles	SB2-05	Desarrollar UI para autenticación (iam)	Crear los componentes de frontend para registro, login y gestión de perfil.	10	Miguel Hallasi	Done
US09, US10, US19	Gestión de Viajes	SB2-06	Desarrollar UI para gestión de viajes (rides)	Crear los componentes de frontend para buscar, ver y publicar viajes.	12	Miguel Hallasi	Done
---	---	SB2-07	Pruebas unitarias para frontend	Implementar pruebas unitarias con Vitest para los componentes de iam y Rides.	8	Miguel Hallasi	Done
TS01, TS02	Endpoints de Autenticación	SB2-08	Modificar servicio de autenticación (iam)	Realizar ajustes y mejoras en el microservicio de autenticación.	8	Samuel Valera	In-Process
TS03, TS04	Endpoints de Gestión de Viajes	SB2-09	Modificar servicio de viajes (ride)	Realizar ajustes y mejoras en el microservicio de gestión de viajes.	8	Samuel Valera	In-Process
---	---	SB2-10	Dockerizar los nuevos servicios	Crear Dockerfiles para los servicios de reportes y mensajería.	6	Anderson Gamarra	To-do
---	---	SB2-11	Actualizar configuración de despliegue	Actualizar el <code>docker-compose.yml</code> para incluir los nuevos servicios y configurar el despliegue continuo.	6	Anderson Gamarra	To-do

5.2.2.2 Development Evidence for Sprint Review

Repository / Service	Branch	Commit ID	Type / Message	Commit Body (Descripción resumida)	Date (dd/mm/yyyy)
message-service	main	544f1c7	feat: add message service	PedroJ18 committed yesterday	01/11/2025
report-service	main	c014e39	feat: implemented ReportTest	vquijandria committed 2 days ago	31/10/2025
report-service	main	f0b5935	feat: implemented ReportServiceTest	vquijandria committed 2 days ago	31/10/2025
report-service	main	bbfc126	feat: added mockito dependencies to pom	vquijandria committed 2 days ago	31/10/2025
report-service	main	0a2bc00	.	vquijandria committed 2 days ago	31/10/2025
web-app	main	ba11a68	chore: add dockerfile for deploy	AndersonGamarraJW committed 5 minutes ago	02/11/2025
web-app	main	3b5d168	feat: add iam and rides	mhallasi authored and AndersonGamarraV committed 3 hours ago	02/11/2025
web-app	main	fb1573f	rebuild	mhallasi authored and AndersonGamarraV committed 3 hours ago	02/11/2025
web-app	main	78a855c	feat: add web-app	SamuelValeraGarces authored 3 weeks ago	09/10/2025

5.2.2.3 Testing Suite Evidence for Sprint Review

Message Service

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

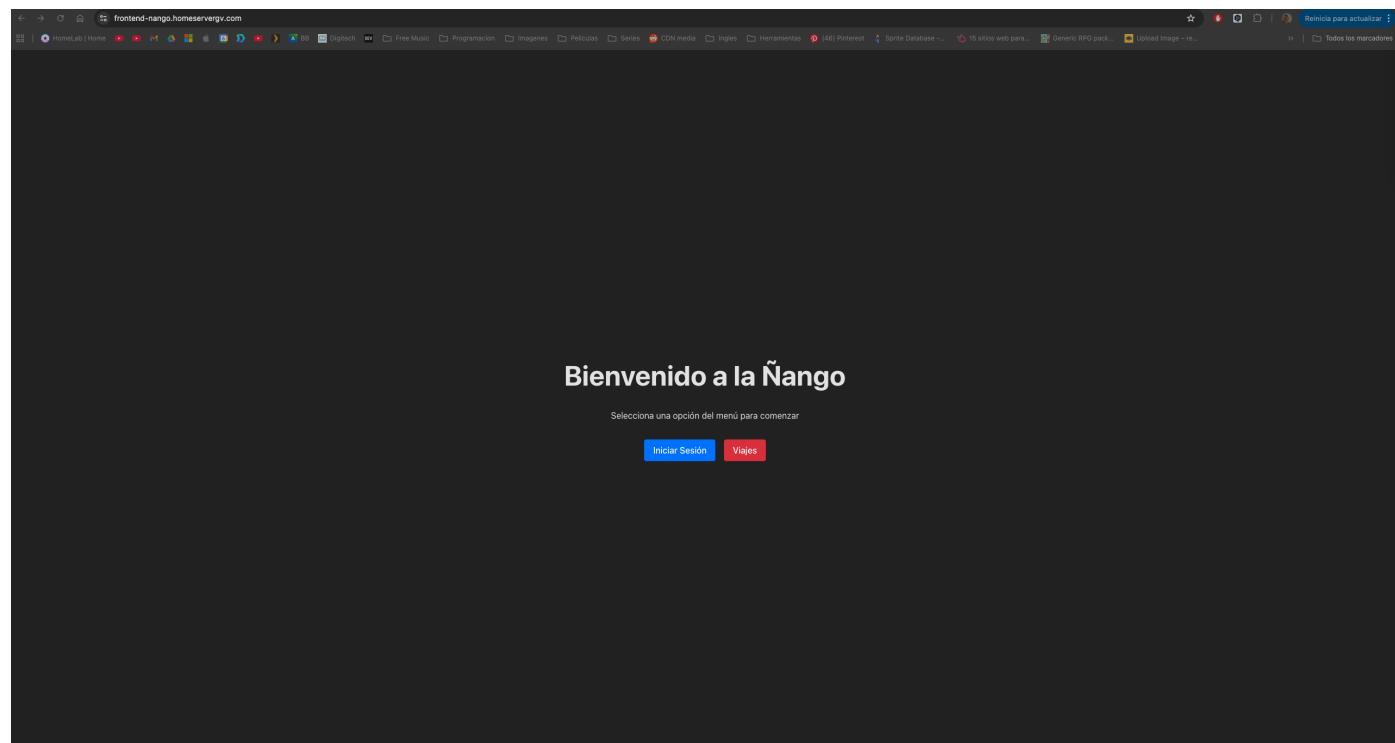
- Project:** message-service > feature/unit-tests
- File:** MessageServiceImplTest.java
- Test Results:**
 - 6 tests passed
 - 6 tests total
 - 4 sec 410 ms
- Test Cases:**
 - DeleteMessageShouldDeleteWhenExists() (4 sec 315 ms)
 - GetMessageByldShouldReturnOptionalMessage() (26 ms)
 - GetAllMessagesShouldReturnList() (12 ms)
 - GetMessagesBySenderIdShouldReturnMessagesFromSender() (22 ms)
 - DeleteMessageShouldThrowExceptionWhenNotExists() (18 ms)
 - SendMessageShouldSaveAndReturnMessage() (17 ms)

Report Service

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

- Run:** ReportServiceApplication > ReportServiceTest
- Test Results:**
 - Tests passed: 4 of 4 tests
 - Process finished with
- Test Cases:**
 - createReport_shouldPersist_andReturnDTO() (1 sec 603 ms)
 - createReport_shouldThrow_whenInvalidInput() (1 sec 589 ms)
 - getReportByld_shouldReturnEmpty_whenNotFound() (4 ms)
 - getReportByld_shouldReturnDTO_whenFound() (7 ms)

5.2.2.4 Execution Evidence for Sprint Review



5.2.2.5 Microservices Documentation Evidence for Sprint Review

Report Service Documentation

POST /api/reports

Parameters

No parameters

Request body required

application/json

```
{  
    "rideId": 4,  
    "reason": "Going above speed limit"  
}
```

Execute **Clear**

Responses

Curl

```
curl -X 'POST' \  
  'http://localhost:8089/api/reports' \  
  -H 'accept: */*' \  
  -H 'Content-Type: application/json' \  
  -d '{  
    "rideId": 4,  
    "reason": "Going above speed limit"  
}'
```

Request URL

<http://localhost:8089/api/reports>

Server response

Code	Details	Links
201	Response body <pre>{ "reportId": 1, "rideId": 4, "reportDate": "2025-11-02T19:46:35.8756245", "reason": "Going above speed limit" }</pre> Download	
	Response headers <pre>cache-control: no-cache,no-store,max-age=0,must-revalidate connection: keep-alive content-type: application/json date: Mon,03 Nov 2025 00:46:36 GMT expires: 0 keep-alive: timeout=60 pragma: no-cache transfer-encoding: chunked x-content-type-options: nosniff x-frame-options: DENY x-xss-protection: 0</pre>	

Responses

Code	Description	Links
201	Created	No links

Media type

/

Controls Accept header.

[Example Value](#) [Schema](#)

GET /api/reports/{id}

Parameters

Name Description

id * required
integer(\$int64)
(path) 1

Cancel

Execute Clear

Responses

Curl

```
curl -X 'GET' \
'http://localhost:8089/api/reports/1' \
-H 'accept: */*' Copy
```

Request URL

```
http://localhost:8089/api/reports/1 Copy
```

Server response

Code Details

200 Response body

```
{
  "reportId": 1,
  "rideId": 4,
  "reportDate": "2025-11-02T19:46:35.875625",
  "reason": "Going above speed limit"
} Copy Download
```

Response headers

```
cache-control: no-cache,no-store,max-age=0,must-revalidate
connection: keep-alive
content-type: application/json
date: Mon,03 Nov 2025 00:47:06 GMT
expires: 0
keep-alive: timeout=60
pragma: no-cache
transfer-encoding: chunked
x-content-type-options: nosniff
x-frame-options: DENY
x-xss-protection: 0
```

Responses

Code	Description	Links
200	OK	No links

Media type

```
*/* ▼
```

Controls Accept header.

Example Value | Schema

```
{
  "reportId": 0,
  "rideId": 0,
  "reportDate": "2025-11-03T00:47:06.730Z",
  "reason": "string"
}
```

message-controller

POST /api/messages/send

Parameters
No parameters

Request body required

```
{
  "senderId": "Julia03",
  "receiverId": "vicente055",
  "content": "Buenas tardes",
  "messageTime": "2025-11-03T00:05:45.737Z"
}
```

Responses

Curl

```
curl -X 'POST' \
'http://localhost:8080/api/messages/send' \
-H 'Content-Type: application/json' \
-d '{
  "senderId": "Julia03",
  "receiverId": "vicente055",
  "content": "Buenas tardes",
  "messageTime": "2025-11-03T00:05:45.737Z"
}'
```

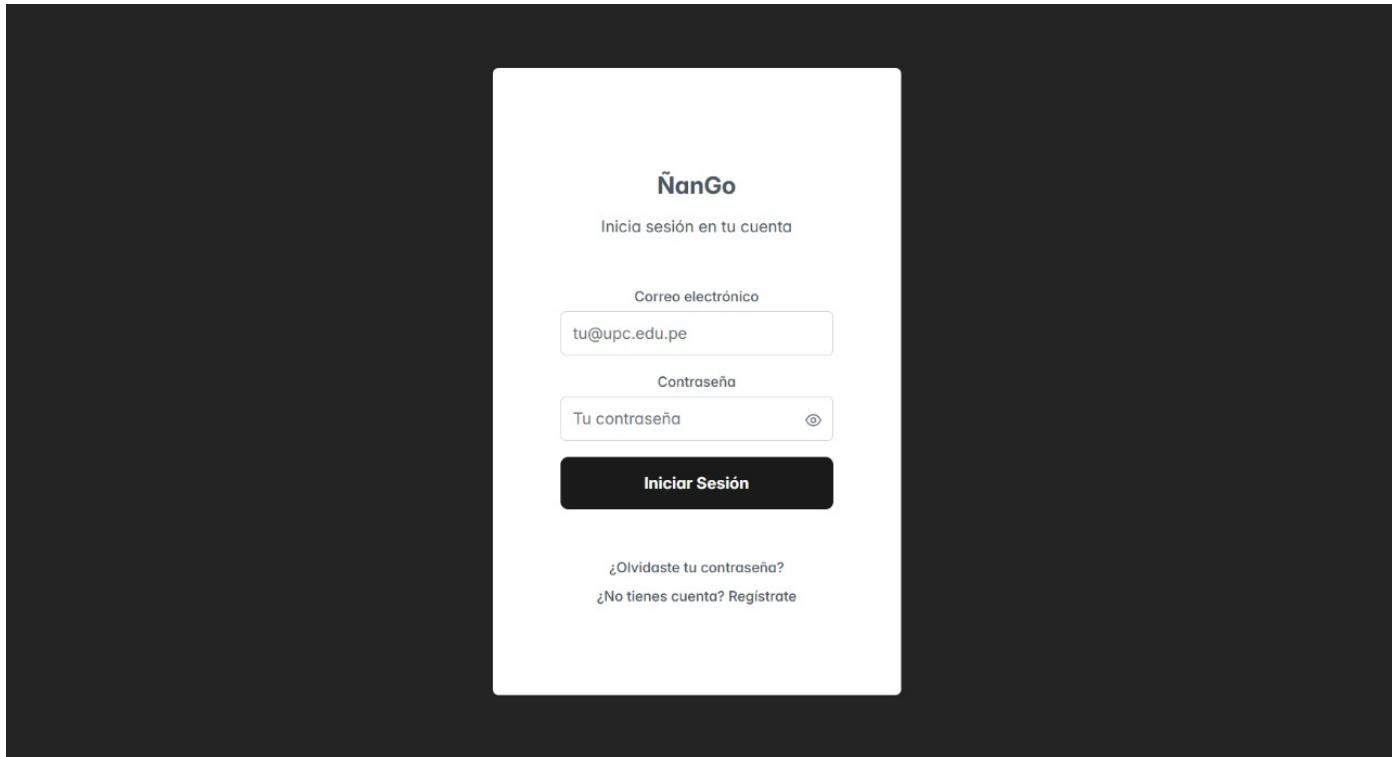
Request URL

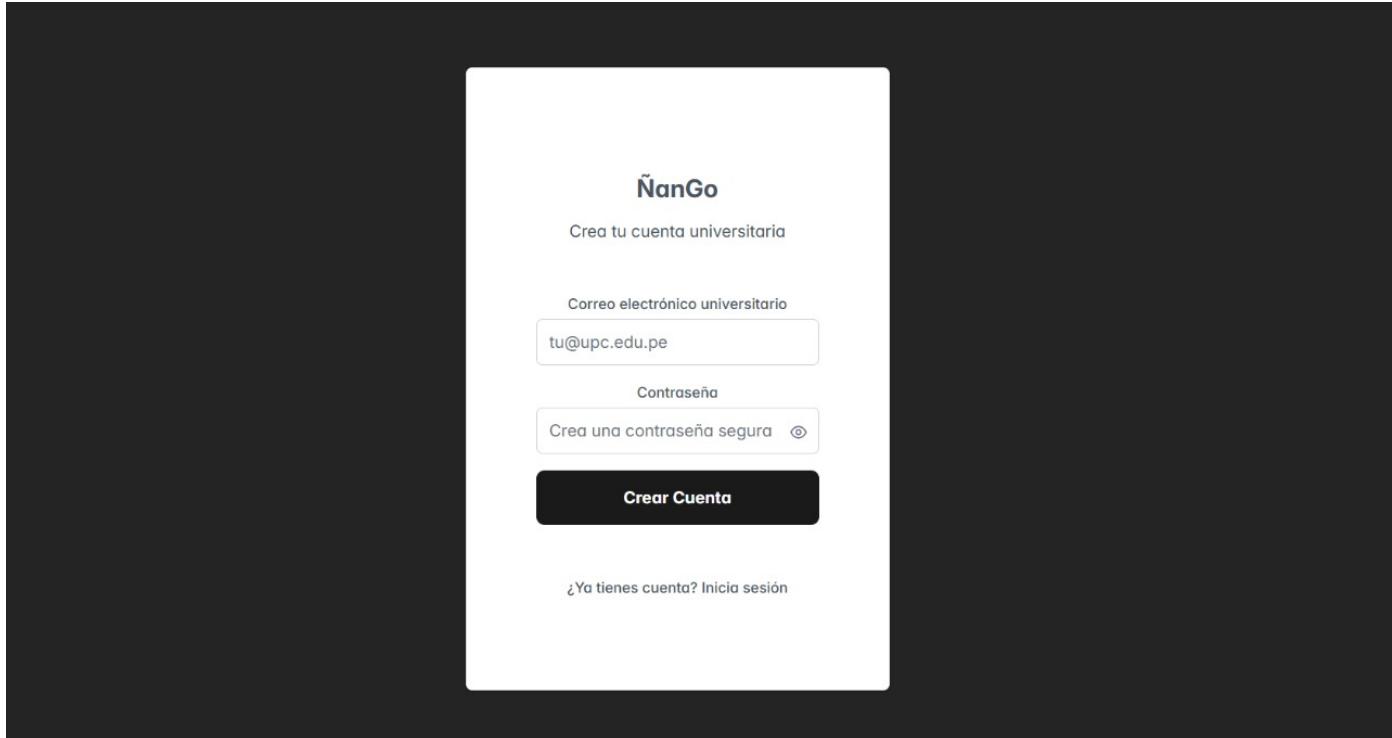
http://localhost:8080/api/messages/send

Server response

Code	Details
200	Response body <pre>{ "id": null, "senderId": "Julia03", "receiverId": "vicente055", "content": "Buenas tardes", "messageTime": "2025-11-03T00:05:45.737Z", "status": "SENT" }</pre> <p>Download</p>

Response headers

5.2.2.6 Software Deployment Evidence for Sprint Review**App Web****Login****Register**



Search Trips

The interface for searching trips is shown. The title 'Viajes Disponibles' is at the top, followed by the subtitle 'Encuentra el viaje perfecto para tu destino'. Below this is a section titled 'Filtros de Búsqueda' with three input fields: 'Origen' (placeholder '¿Desde dónde viajas?'), 'Destino' (placeholder '¿A dónde vas?'), and 'Fecha' (placeholder 'Selecciona fecha'). A search bar with a magnifying glass icon and the word 'Buscar' is at the bottom. Below the search bar, the text '3 viajes encontrados' is displayed, followed by a note 'Ordenados por hora de salida' and a button '+ Solicitar Viaje'.

Found Trips

3 viajes encontrados

Ordenados por hora de salida

Disponible Asientos 3

Universidad UPC - San Isidro
Miraflores - Parque Kennedy

FECHA HORA
lun, 15 ene 08:00

★ ★ ★ ★ ☆ 4.8

Ver Detalles Reservar

Disponible Asientos 2

Universidad UPC - Monerrico
San Borja - Metro

FECHA HORA
lun, 15 ene 18:30

★ ★ ★ ★ ☆ 4.5

Ver Detalles Reservar

Trip Details

Detalles del Viaje
Información completa del viaje

Disponible ID: #1

Origen
Universidad UPC - San Isidro

Destino
Miraflores - Parque Kennedy

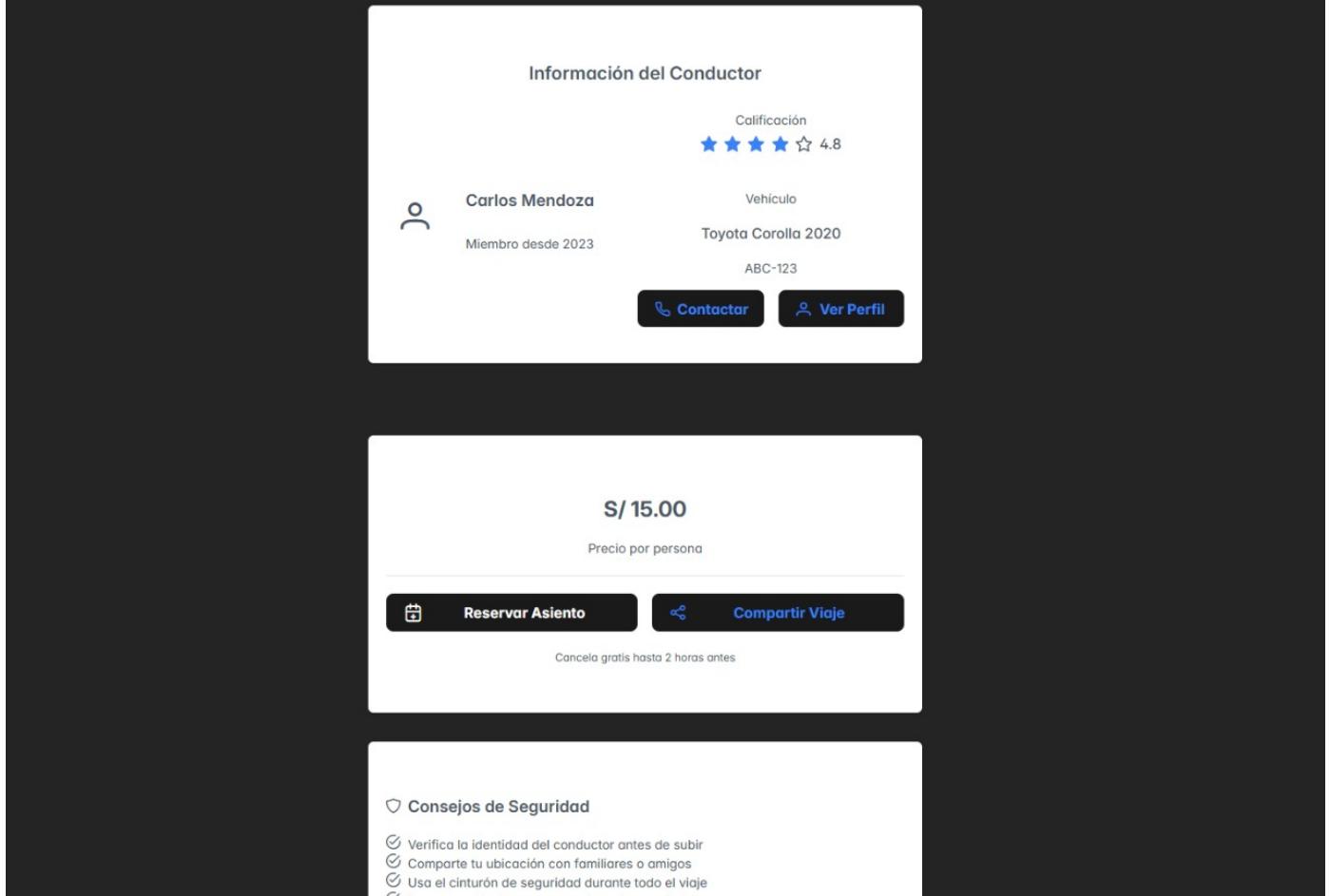
Fecha
lunes, 15 de enero de 2024

Hora de salida
08:00 a. m.

Asientos disponibles
3

Precio estimado
S/ 15.00

Driver Trip



5.2.2.7 Team Collaboration Insights during Sprint

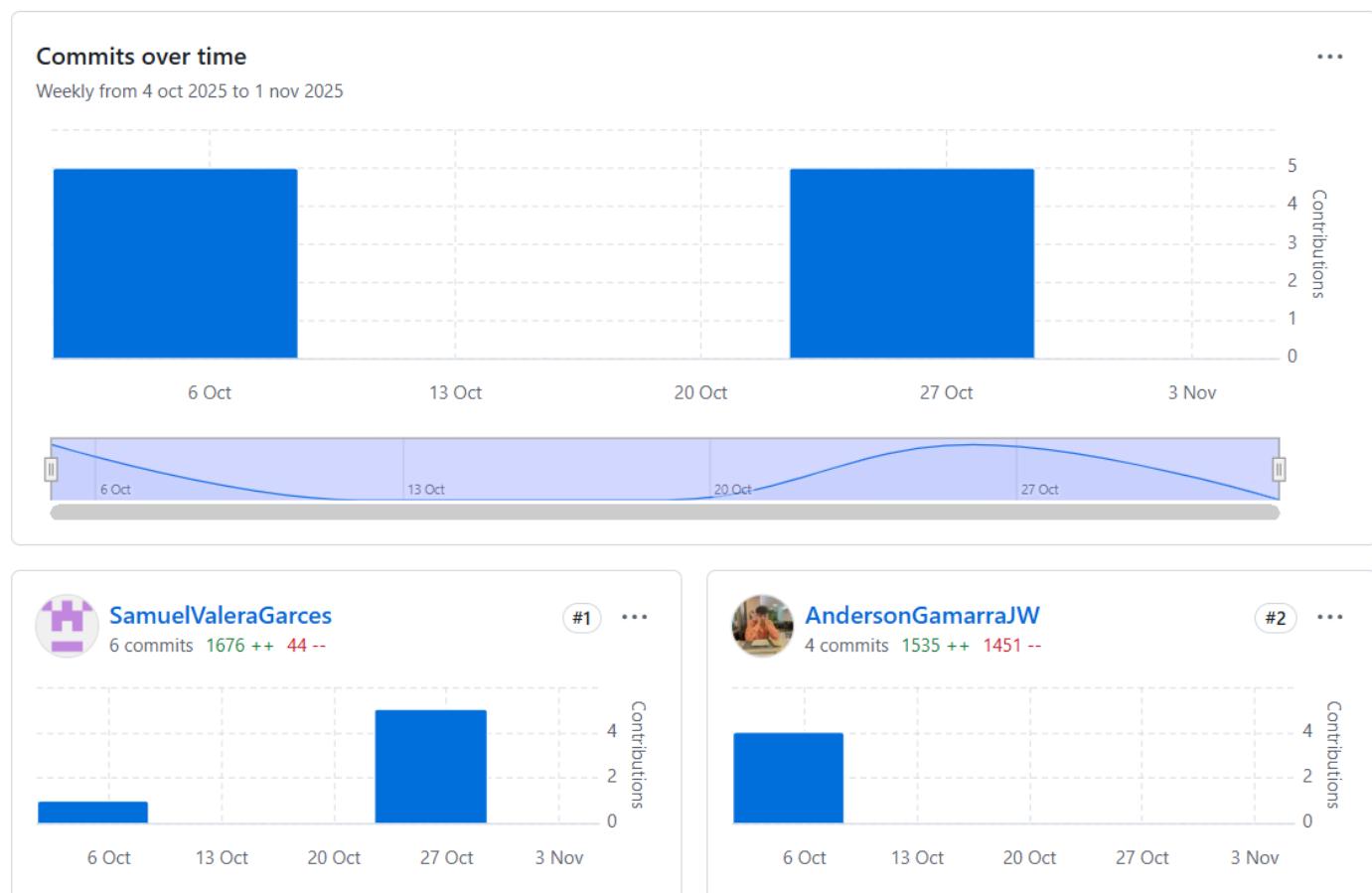
Ride Service

Contributors

Contributions per week to main, excluding merge commits

Period: All ▾

Contributions: Commits ▾



Report Service

Contributors

Contributions per week to main, excluding merge commits

Period: All ▾

Contributions: Commits ▾

Commits over time

Weekly from 25 oct 2025 to 1 nov 2025



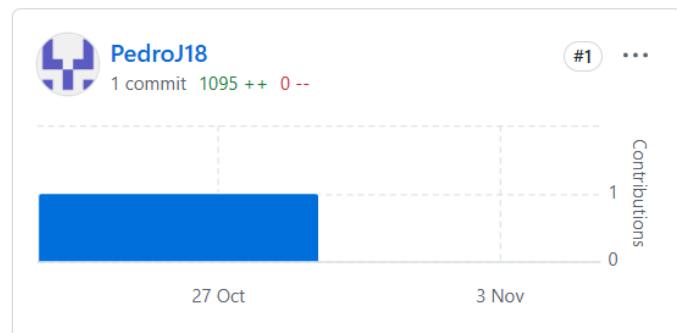
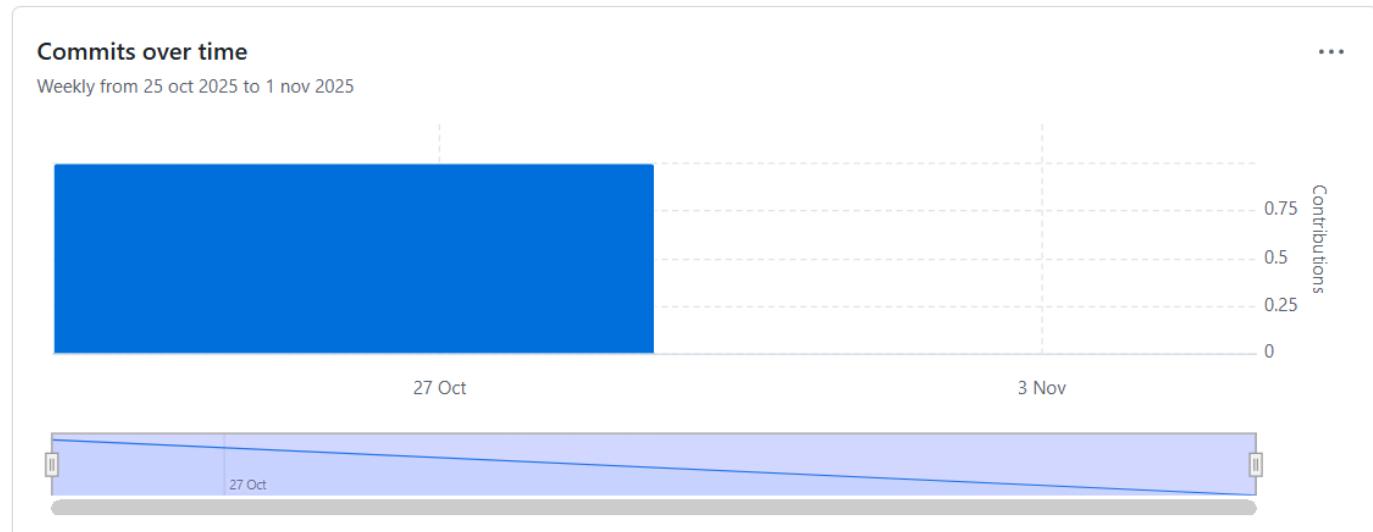
Message Service

Contributors

Contributions per week to main, excluding merge commits

Period: All ▾

Contributions: Commits ▾



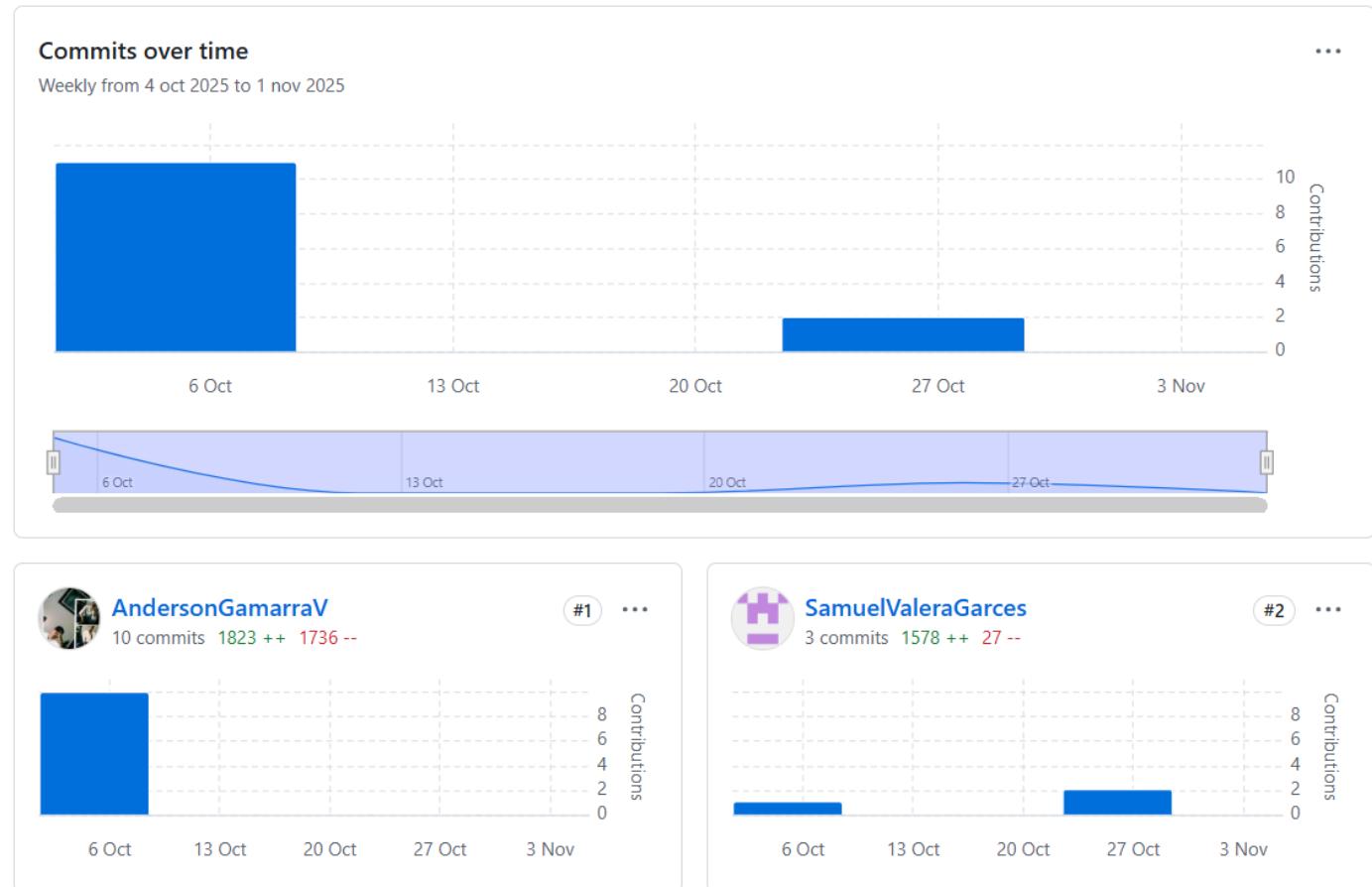
Iam Service

Contributors

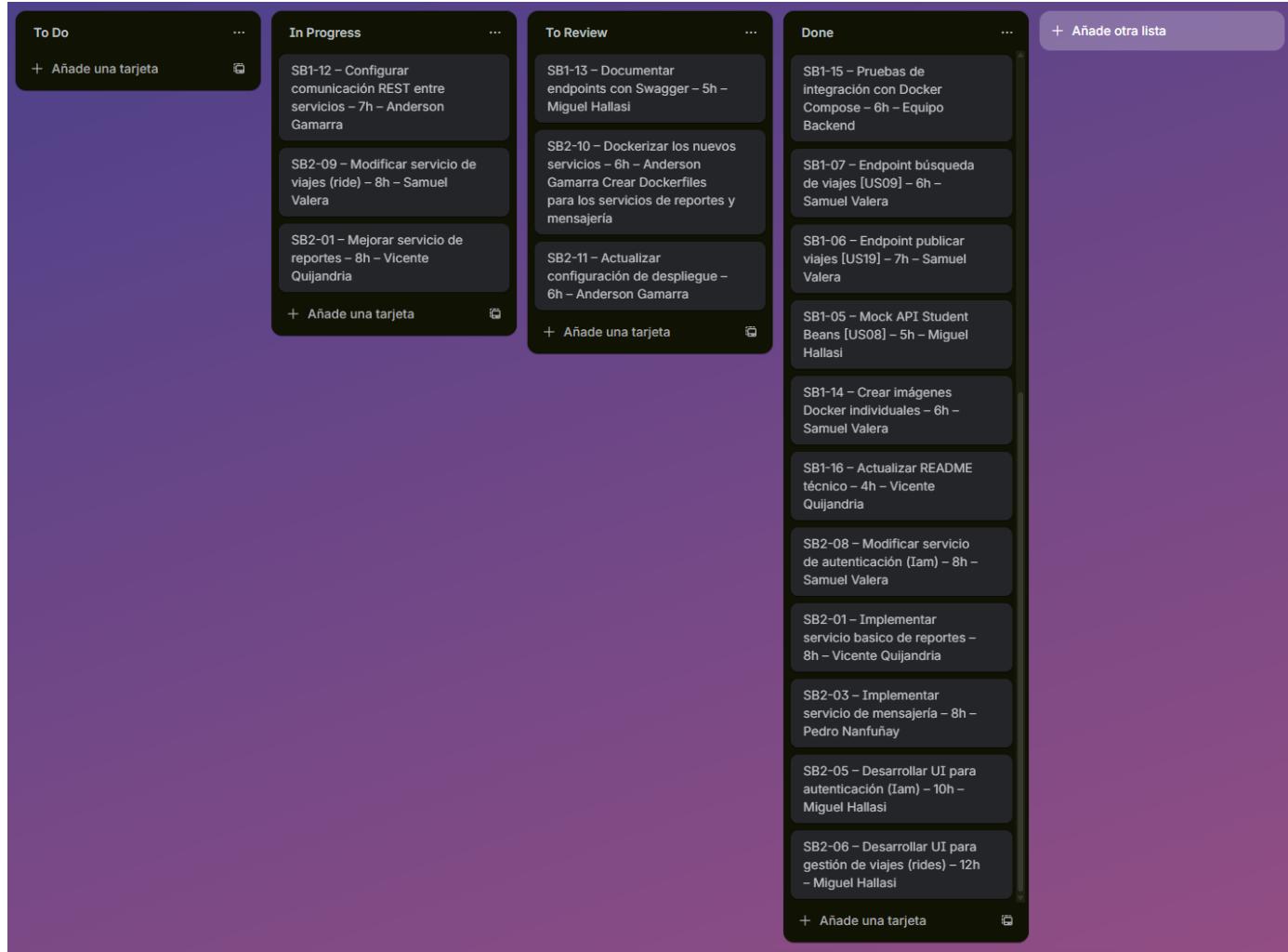
Contributions per week to main, excluding merge commits

Period: All ▾

Contributions: Commits ▾



5.2.2.8 Kanban Board



5.2.3 Sprint 3

5.2.3.1 Sprint Backlog 3

User Story ID	User Story Title	Work-Item ID	Work-Item Title	Description	Estimation (Hours)	Assigned To	Status
TS05	Dockerización de servicios	SB3-01	Dockerizar Ride Service	Crear Dockerfile y configurar entorno para el Ride Service.	6	Samuel Valera	Done
TS05	Dockerización de servicios	SB3-02	Dockerizar Profile Service	Crear Dockerfile y configurar entorno para el Profile Service.	6	Samuel Valera	Done
TS05	Dockerización de servicios	SB3-03	Dockerizar IAM Service	Crear Dockerfile y configurar entorno para el Iam Service.	6	Samuel Valera	Done
TS05	Dockerización de servicios	SB3-04	Dockerizar Message Service	Crear Dockerfile y configurar entorno para el Message Service.	6	Pedro Nanfuñay	Done
TS05	Dockerización de servicios	SB3-05	Dockerizar Report Service	Crear Dockerfile y configurar entorno para el Report Service.	6	Vicente Quijandria	Done
TS06	Actualización del entorno de despliegue	SB3-06	Actualizar docker-compose con todos los servicios	Integrar todos los servicios dockerizados en el <code>docker-compose.yml</code> , definir redes, volúmenes y dependencias.	8	Anderson Gamarra	Done
TS07	Validación de contenedores	SB3-07	Probar ejecución conjunta de servicios	Verificar que todos los servicios funcionen correctamente de forma orquestada con Docker.	6	Miguel Hallasi	Done

5.2.3.2 Development Evidence for Sprint Review

Date (dd/mm/yyyy)	Commit Message	Commit Body (Descripción resumida)	Type	Commit ID	Branch	Repository / Service
14/11/2025	chore: add dockerfile for deploy	AndersonGamarraJW committed 5 minutes ago	chore	ba11a68	main	web-app

Date (dd/mm/yyyy)	Commit Message	Commit Body (Descripción resumida)	Type	Commit ID	Branch	Repository / Service
14/11/2025	feat: add iam and rides	mhallasi authored and AndersonGamarraV committed 3 hours ago	feat	3b5d168	main	web-app
14/11/2025	rebuild	mhallasi authored and AndersonGamarraV committed 3 hours ago	other	fb1573f	main	web-app
12/11/2025	feat: add message service	PedroJ18 committed yesterday	feat	544f1c7	main	message-service
10/11/2025	feat: implemented ReportTest	vquijandria committed recently	feat	c014e39	main	report-service
10/11/2025	feat: implemented ReportServiceTest	vquijandria committed recently	feat	f0b5935	main	report-service
10/11/2025	feat: added mockito dependencies to pom	vquijandria committed recently	feat	bbfc126	main	report-service
10/11/2025	.	vquijandria committed recently	other	0a2bc00	main	report-service
01/11/2025	feat: add web-app	SamuelValeraGarces authored weeks ago	feat	78a855c	main	web-app

5.2.3.3 Testing Suite Evidence for Sprint Review

Message Service

The screenshot shows the IntelliJ IDEA interface with the following details:

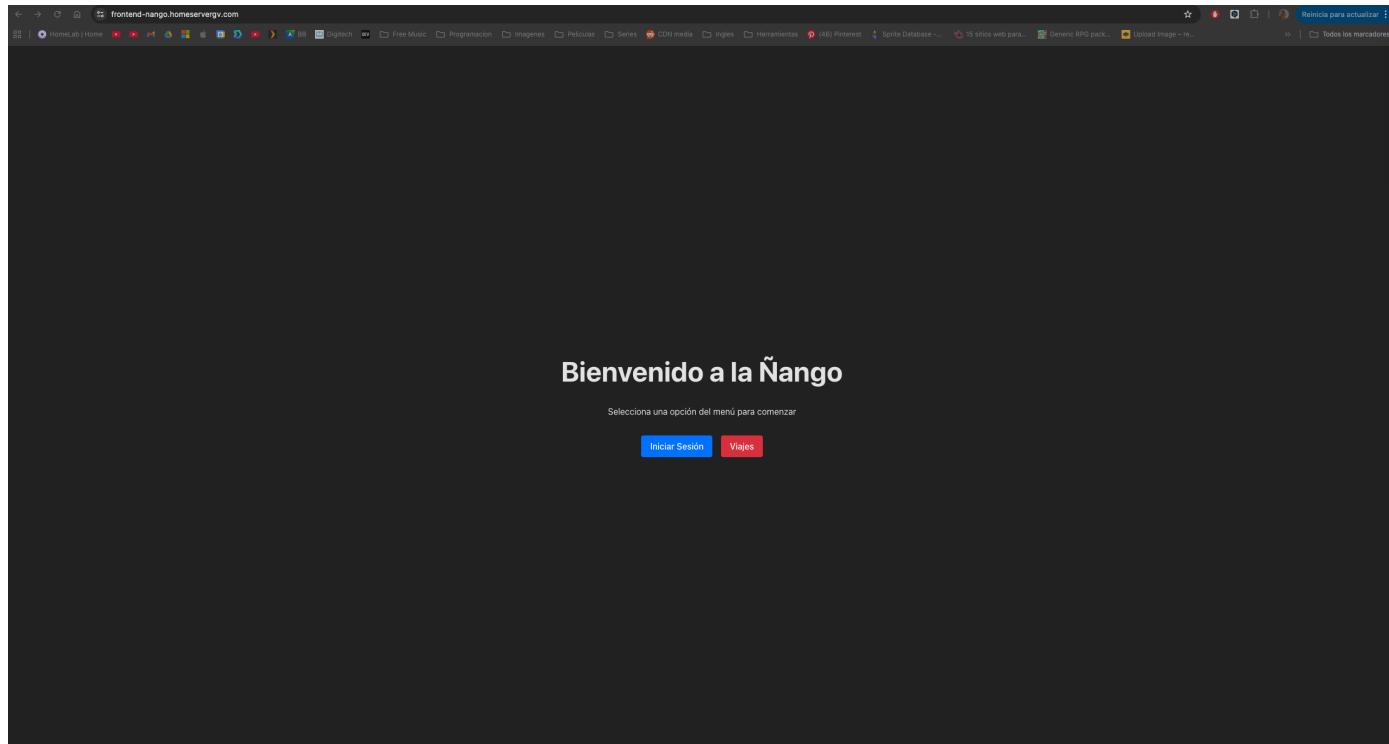
- Project Structure:** The project is named "message-service" and contains packages for model, repository, and service.
- Code Editor:** The file "MessageServiceImplTest.java" is open, showing Java code for testing the MessageService implementation.
- Run Tab:** The "Run" tab is selected, showing a run configuration for "MessageServiceApplication" and the current test "MessageServiceImplTest".
- Tool Window:** The "Tests" tool window is open, displaying the test results:
 - 6 tests passed
 - 6 tests total, 4 sec 410 ms
 - Test cases listed: DeleteMessageShouldDeleteWhenExists(), GetMessageByIdShouldReturnOptionalMessage() (highlighted), GetAllMessagesShouldReturnList(), GetMessagesBySenderIdShouldReturnMessagesFromSender(), DeleteMessageShouldThrowExceptionWhenNotExists(), SendMessageShouldSaveAndReturnMessage()
- Status Bar:** Shows the file path as "message-service > src > test > java > com > tinkuytech > nango > message > service > MessageServiceImplTest" and the status as "89:9 CRLF UTF-8 4 spaces".

Report Service

The screenshot shows the IntelliJ IDEA interface with the following details:

- Project Structure:** The project is named "ReportServiceApplication" and contains a package for ReportServiceTest.
- Code Editor:** The file "ReportServiceTest.java" is open, showing Java code for testing the ReportService implementation.
- Run Tab:** The "Run" tab is selected, showing a run configuration for "ReportServiceApplication" and the current test "ReportServiceTest".
- Tool Window:** The "Tests" tool window is open, displaying the test results:
 - Tests passed: 4 of 4 tests
 - Process finished with exit code 0
 - Warnings listed: WARNING: If a service, WARNING: If a service, WARNING: Dynamic load, OpenJDK 64-Bit Server VM
- Status Bar:** Shows the file path as "ReportServiceApplication > src > test > java > com > tinkuytech > nango > report > application > service > ReportServiceTest" and the status as "89:9 CRLF UTF-8 4 spaces".

5.2.3.4 Execution Evidence for Sprint Review



A screenshot of the Ñango website. At the top, there is a navigation bar with links to "Funcionalidades", "Cómo funciona", "Planes", and "FAQ". There are also "Ver demo" and "Descargar" buttons. The main section features a heading "Comparte el viaje, ahorra y llega seguro con Ñango." with a subtext "Conecta con estudiantes de tu universidad para compartir rutas confiables, validar perfiles y pagar sin fricción." Below this is a button "Descargar app" and a "Cómo funciona" button. A note below says "✓ Verificación de identidad ✓ Notificaciones en tiempo real ✓ Pagos seguros". To the right is an illustration of two people using phones to share a ride. A green button "Empezar" is overlaid on the phone screen. A callout bubble says "+2,500 estudiantes". Below this section is a heading "Funcionalidades clave" with a note "Inspired in the design of the app: simple, clear and thought for students." Six functional boxes are shown: "Perfiles verificados" (lock icon), "Rutas y horarios" (map icon), "Notificaciones" (bell icon), "Pagos integrados" (credit card icon), "Idiomas" (globe icon), and "Soporte" (support ticket icon).

5.2.3.5 Microservices Documentation Evidence for Sprint Review

Profile Service

Profile Service API 1.0 OAS 3.0

/v3/api-docs

Servers

http://profile-service-nango-fas.homeservergy.com - Generated server url ▾

Authorize 

profile-controller

GET	/api/profiles/user/{userId}	🔒 ▾
PUT	/api/profiles/user/{userId}	🔒 ▾
GET	/api/profiles	🔒 ▾
POST	/api/profiles	🔒 ▾

Schemas

ProfileDto >

Iam Service

IAM Service API 1.0 OAS 3.0

/bus/v3/api-docs

Microservice for user authentication and role management in nanGo

Servers

http://iam-service-nango-fas.homeservergy.com - Generated server url ▾

Authorize 

auth-controller

POST	/api/auth/register	▼ 
POST	/api/auth/login	▼ 

user-controller

GET	/api/users	▼ 
GET	/api/users/{username}	▼ 
GET	/api/users/exists/{username}	▼ 

Profile Service

Profile Service API 1.0 OAS 3.0

/v3/api-docs

Servers

http://profile-service-nango-fas.homeservergv.com - Generated server url ▾

Authorize 

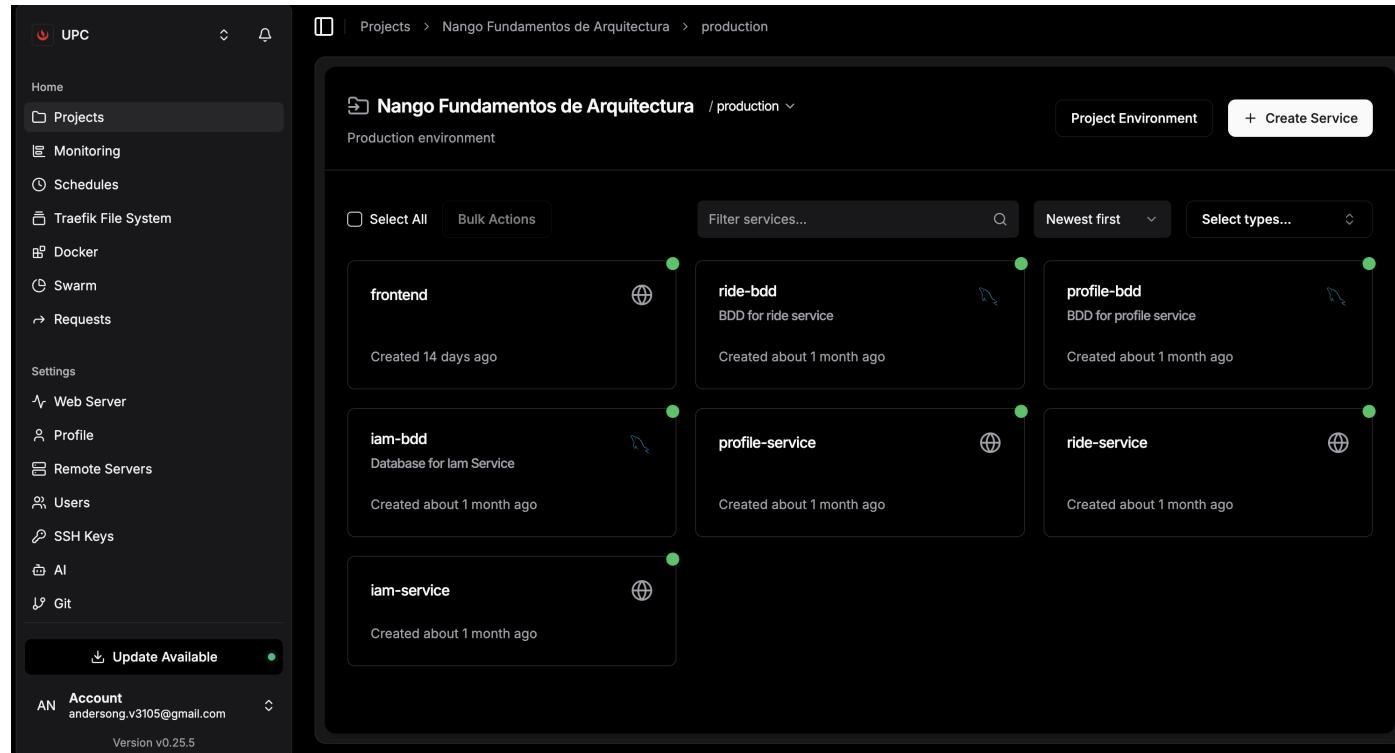
profile-controller

<code>GET</code>	/api/profiles/user/{userId}	
<code>PUT</code>	/api/profiles/user/{userId}	
<code>GET</code>	/api/profiles	
<code>POST</code>	/api/profiles	

Schemas

ProfileDto >

5.2.3.6 Software Deployment Evidence for Sprint Review



The screenshot shows the Nango Fundamentos de Arquitectura dashboard in the production environment. The sidebar on the left includes options like Home, Projects, Monitoring, Schedules, Traefik File System, Docker, Swarm, Requests, Settings, Web Server, Profile, Remote Servers, Users, SSH Keys, AI, and Git. A 'Update Available' button is also present. The main area displays a grid of service cards:

- frontend**: BDD for frontend service, created 14 days ago.
- ride-bdd**: BDD for ride service, created about 1 month ago.
- profile-bdd**: BDD for profile service, created about 1 month ago.
- iam-bdd**: Database for iam Service, created about 1 month ago.
- profile-service**: Database for profile service, created about 1 month ago.
- ride-service**: Database for ride service, created about 1 month ago.
- iam-service**: Database for iam service, created about 1 month ago.

5.2.3.7 Team Collaboration Insights during Sprint

Screenshot of the GitHub Insights page for the repository ASI0657-2520-faw-6327 / report-service. The page shows activity from October 16, 2025, to November 16, 2025.

Overview:

- 0 Active pull requests
- 0 Active issues
- Merged pull requests: 0
- Open pull requests: 0
- Closed issues: 0
- New issues: 0

Summary:

Excluding merges, 1 author has pushed 8 commits to main and 8 commits to all branches. On main, 0 files have changed and there have been 0 additions and 0 deletions.

Top Committers:

User	Commits
[User Icon]	1

Screenshot of the GitHub Insights page for the repository ASI0657-2520-faw-6327 / docs. The page shows activity from October 16, 2025, to November 16, 2025.

Overview:

- 4 Active pull requests
- 0 Active issues
- Merged pull requests: 4
- Open pull requests: 0
- Closed issues: 0
- New issues: 0

Summary:

Excluding merges, 5 authors have pushed 20 commits to main and 29 commits to all branches. On main, 21 files have changed and there have been 148 additions and 9 deletions.

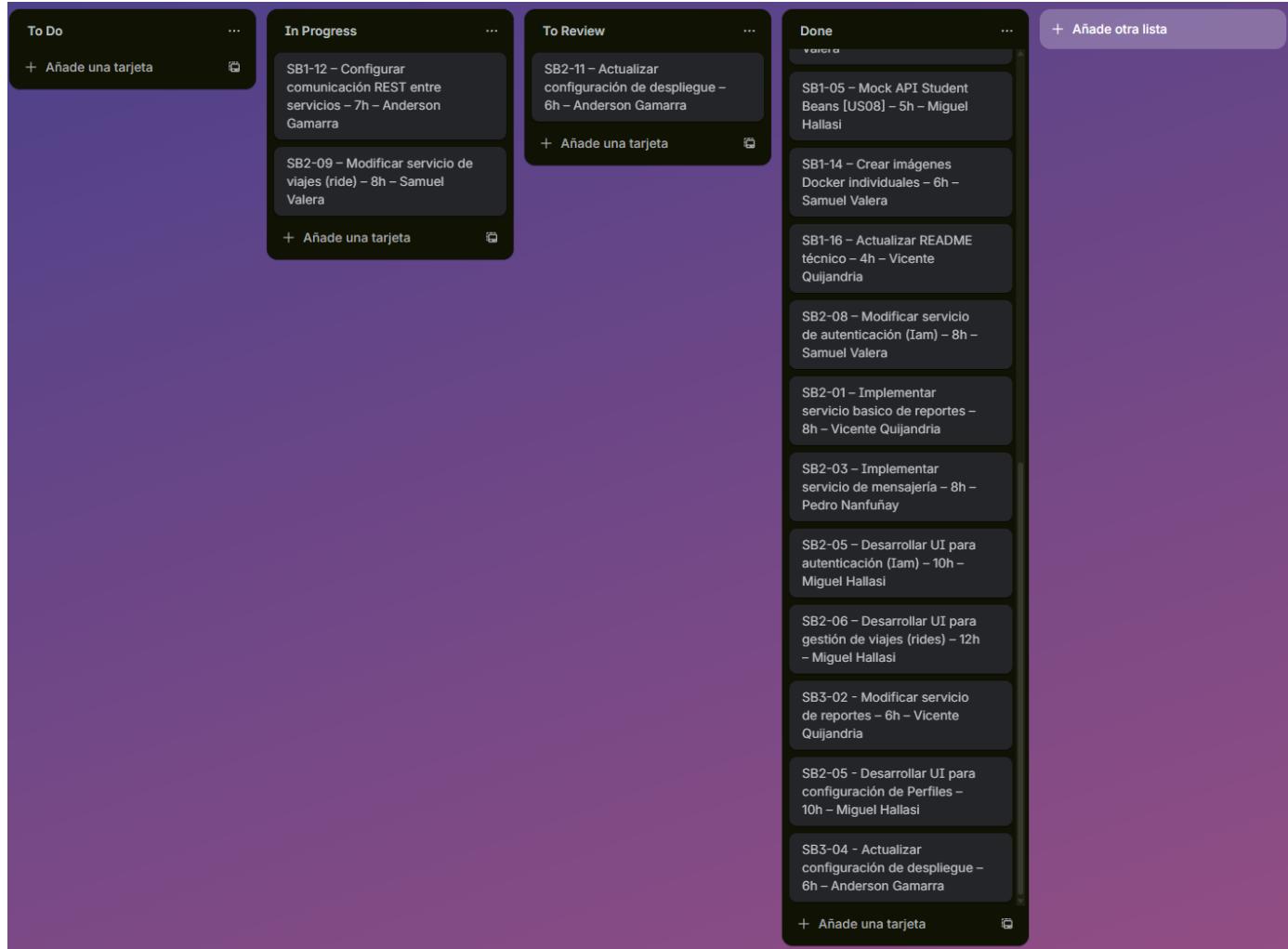
Top Committers:

User	Commits
[User Icon]	8
[User Icon]	5
[User Icon]	5
[User Icon]	3
[User Icon]	2

Kanban Board:

#101 merged 2 hours ago

5.2.3.8 Kanban Board



Link de Trello: <https://trello.com/invite/b/68d9fec4a886565a738c5578/ATTI5fe0d222c5142993d845a183ced85bc8475F9959/sprint-1>

Conclusiones

Conclusiones y Recomendaciones

- Ñango es una solución tecnológica efectiva que ofrece servicio de movilidad compartido entre estudiantes universitarios que garantiza la seguridad de los pasajeros y conductores que buscan un medio de transporte eficiente y rápido. Así mismo, permite a los usuarios conocer a nuevas personas de su centro de estudios.
- La colaboración efectiva entre los miembros del equipo fue clave para avanzar de manera organizada y cubrir todas las etapas del proyecto. Cada integrante asumió con claridad sus responsabilidades a través de issues en la plataforma Github, lo que permitió una documentación organizada.
- La definición de User Stories y del Product Backlog permitió priorizar las funcionalidades esenciales, asegurando que la solución respete y cumpla con lo que requieren los usuarios y el mercado.

Referencias Bibliográficas

Anexos

Links

Url de la organización: <https://github.com/ASI0657-2520-faw-6327>

Url del informe: <https://github.com/ASI0657-2520-faw-6327/docs>

Url del Landing Page: <https://frontend-nango.homeservergy.com/>

Url del Web App: <https://frontend-nango.homeservergy.com/>

Microservicios

Url Ride Service: [link](#)

Url Profile Service: [link](#)

Url Iam Service: [link](#)

Url Message Service: [link](#)

Url Report Service: [link](#)