Pontificia Universidad Católica del Perú

Facultad de Ciencias e Ingeniería

DISEÑO E IMPLEMENTACIÓN DE UNA PLATAFORMA WEB-MÓVIL PARA GESTIÓN LOGÍSTICA EN PYMES CON MEDICIÓN AUTOMÁTICA DE DIMENSIONES DE PAQUETES MEDIANTE INTELIGENCIA ARTIFICIAL

Tesis para obtener el título profesional de Ingeniero de las Telecomunicaciones

Autor:

Juan Alfonso Chapoñan Espinoza

Asesor:

Oscar Antonio Díaz Barriga

Lima, agosto, 2025

INFORME DE SIMILITUD

Yo, **OSCAR ANTONIO DÍAZ BARRIGA**, docente de la Facultad de **CIENCIAS E INGENIERÍA** de la Pontificia Universidad Católica del Perú, asesor de la tesis titulada **PLATAFORMA WEB Y MÓVIL CON PROCESAMIENTO DE IMÁGENES EN LA NUBE E INTELIGENCIA ARTIFICIAL PARA LA IDENTIFICACIÓN DE PAGOS, ESTIMACIÓN DE DIMENSIONES Y OPTIMIZACIÓN LOGÍSTICA EN SERVICIOS DE DELIVERY URBANO EN LIMA**, del autor **JUAN ALFONSO CHAPOÑAN ESPINOZA**, dejo constancia de lo siguiente:

- El mencionado documento tiene un índice de puntuación de similitud de [##] %. Así lo consigna el reporte de similitud emitido por el software Turnitin el [**DD/MM/AAAA**].
- He revisado con detalle dicho reporte y la Tesis, y no se advierte indicios de plagio.
- Las citas a otros autores y sus respectivas referencias cumplen con las pautas académicas.

Lugar y fecha: San Miguel, [DD] de [MES] de [AAAA].

Apellidos y nombres del asesor: **DÍAZ BARRIGA, OSCAR ANTONIO**	
DNI: **71477000**	Firma
ORCID: 0000-0002-9930-5984	

RESUMEN

[Por completar]

DEDICATORIA

A mis padres.

AGRADECIMIENTOS

[Por completar]

ÍNDICE

ÍNDICE DE TABLAS

ÍNICE DE FIGURAS

[Por completar]

INTRODUCCIÓN

[Por completar]

Capítulo 1. Presentación del problema de ingeniería

La transformación digital en el sector logístico demanda soluciones innovadoras que aprovechen tecnologías emergentes para resolver desafíos operacionales críticos. Esta investigación aborda la problemática de la inexactitud en la determinación de dimensiones de paquetes en la industria de *delivery*, factor que genera ineficiencias significativas en la planificación de rutas y utilización de capacidad vehicular. Mediante la convergencia de aplicaciones IoT, procesamiento de imágenes con inteligencia artificial y arquitecturas distribuidas en la nube, se propone una solución integral que mejora la precisión operacional y democratiza el acceso a tecnologías sofisticadas para empresas de diferentes escalas.

1.1 Identificación temática y motivación personal

1.1.1 Área de especialización en telecomunicaciones

Aplicaciones IoT (*Internet of Things*)

Las aplicaciones IoT constituyen un ecosistema tecnológico integral que integra inteligencia artificial, redes de comunicación y automatización para crear una infraestructura de conectividad ubicua entre objetos físicos y sistemas digitales [1]. Se define como la implementación de una arquitectura de cinco capas interconectadas:

Tabla 1.1: Arquitectura IoT [2].

Capa	**Descripción**
Percepción	Sensores y dispositivos de captura
Red	Protocolos de comunicación y transmisión
Middleware	Procesamiento y almacenamiento de datos
Aplicación	Servicios e interfaces de usuario
Negocio	Análisis y toma de decisiones

Servicios de Telecomunicaciones para Logística

Los servicios de telecomunicaciones para logística se definen como el conjunto de tecnologías y protocolos de comunicación que operan principalmente en la capa de red del ecosistema IoT, actuando como puente crítico entre la percepción de datos y su procesamiento funcional. Estos servicios garantizan la transmisión eficiente y segura de información tanto estática como móvil durante todas las fases del proceso logístico [2].

Convergencia Tecnológica

La tesis desarrollada en esta investigación representa la convergencia de múltiples disciplinas dentro de la ingeniería de telecomunicaciones:

- **Arquitecturas Distribuidas en la Nube**: Para el procesamiento remoto y almacenamiento escalable
- **Inteligencia Artificial**: Específicamente visión por computadora para el procesamiento automatizado de imágenes

Esta integración tecnológica permite abordar desafíos reales del sector logístico mediante soluciones que aprovechan las capacidades de procesamiento remoto, almacenamiento distribuido y comunicaciones continuas, características fundamentales de los sistemas modernos de telecomunicaciones en el contexto de la transformación digital.

1.1.2 Relación con los estudios realizados

La presente investigación se fundamenta en una progresión curricular especializada que abarca desde los fundamentos del desarrollo web hasta la implementación de soluciones IoT avanzadas, estableciendo una relación directa y sistemática con tres cursos clave que proporcionan las competencias técnicas necesarias.

TEL131 Ingeniería Web para Telecomunicaciones

Este curso proporciona la base tecnológica para desarrollar la capa de aplicación e interfaces de gestión en soluciones de logística inteligente. Se abordan fundamentos de programación, desarrollo web con conexión a bases de datos, y modelado relacional con SQL, aplicables en *dashboards* para monitoreo en tiempo real, interfaces de control y manejo de datos IoT.

TEL137 Gestión de Servicios de TICs

Este curso se enfoca en la gestión de servicios dentro del ecosistema IoT, brindando competencias para desarrollar infraestructuras seguras, escalables y robustas. A través de *frameworks* modernos y servicios web, se construyen sistemas capaces de optimizar rutas y asignar recursos inteligentemente.

1TEL05 Servicios y Aplicaciones para IoT

Este curso se centra en la construcción de la capa de aplicación IoT y su integración con la nube, facilitando el desarrollo de soluciones logísticas orientadas al usuario final. Se abordan competencias en aplicaciones móviles conectadas a servicios SaaS, útiles para rastreo de productos y alertas inteligentes.

1.1.3 Motivación personal y experiencia profesional

La motivación personal para abordar esta problemática combina una vocación por la automatización de procesos con un interés social en mejorar la eficiencia empresarial. La experiencia profesional durante la carrera, desde almacenero hasta asistente logístico, brindó una comprensión directa de los desafíos operacionales, destacando la importancia del cumplimiento de tiempos en cada etapa del proceso logístico.

1.2 Descripción y características del problema

1.2.1 Contexto del problema en la industria de *delivery*

La industria de servicios de *delivery* y logística de última milla ha experimentado un crecimiento exponencial en los últimos años, impulsada por el auge del comercio electrónico y los cambios en los hábitos de consumo de la población [3].

Tabla 1.2: Proyecciones del Mercado de delivery en Perú.

Año	**Crecimiento (%)**	**Ingresos (millones USD)**
2024	—	2,500
2029	11.03	2,951

Uno de los principales desafíos que enfrentan las empresas de *delivery* es obtener información precisa sobre las dimensiones y características de los paquetes que deben recoger y entregar.

1.2.2 Desafío técnico desde la perspectiva de telecomunicaciones

Desde la perspectiva de la ingeniería de telecomunicaciones, el problema central radica en la necesidad de desarrollar un sistema distribuido que permita el procesamiento automatizado de imágenes capturadas por dispositivos móviles para la determinación precisa de dimensiones de paquetes.

1.2.3 Características específicas del problema

El problema presenta características específicas que lo hacen particularmente complejo desde el punto de vista técnico y operacional:

- Gran variabilidad en dimensiones, formas y características físicas de los paquetes
- Limitación de capacidad de los motorizados
- Necesidad de confiabilidad en los procesos de verificación
- Requisito de escalabilidad del sistema

1.3 Importancia del problema y su solución

1.3.1 Perspectiva técnica

Resolver este problema es clave por el uso de tecnologías emergentes que transforman telecomunicaciones y computación distribuida. El procesamiento en la nube ha madurado para realizar análisis complejos de imágenes sin necesidad de ejecución local en dispositivos [4], [5].

1.3.2 Perspectiva económica y financiera

La automatización en medición de paquetes mejora rentabilidad y competitividad logística, generando ahorros significativos y optimizando recursos de transporte [6].

Figura 1.1: Gasto en tecnologías de la información - América Latina.

1.3.3 Perspectiva social y cultural

Socialmente, la solución mejora la calidad de vida de trabajadores logísticos y usuarios finales. La automatización de tareas repetitivas permite enfocar recursos humanos en actividades de mayor valor, mejorando satisfacción laboral y desarrollo profesional.

1.3.4 Perspectiva ambiental y de sostenibilidad

Abordar este problema es clave para reducir emisiones de gases de efecto invernadero y usar recursos energéticos eficientemente. La optimización de rutas, basada en medidas precisas de paquetes, permite planificar trayectos más cortos, disminuyendo consumo de combustible y emisiones de CO_2.

1.3.5 Perspectiva legal y reglamentaria

En Perú, la medición automatizada de paquetes con IoT debe cumplir la Ley N.º 29733, que exige consentimiento previo, finalidad clara, calidad y seguridad en el tratamiento de datos [7].

1.3.6 Perspectiva ética

La automatización plantea dilemas éticos sobre responsabilidad social, equidad digital, sostenibilidad ambiental, privacidad de datos y rendición de cuentas. Esta visión ética integral asegura que la tecnología respete valores humanos y apoye un desarrollo sostenible.

1.4 Impactos previstos y beneficiarios

1.4.1 Impactos operacionales

La implementación tendrá impactos operacionales significativos, mejorando la eficiencia mediante tecnologías IoT, IA y visión artificial [8], [9].

Reducción de tiempos de procesamiento

La medición automatizada elimina procesos manuales, reduciendo el tiempo requerido para registrar, verificar y procesar solicitudes [4].

Optimización de rutas de entrega

Con datos precisos sobre dimensiones, los sistemas pueden planificar rutas más eficientes considerando capacidad de carga, tiempo y distancia.

1.4.2 Impactos tecnológicos

La implementación tendrá un impacto tecnológico significativo en telecomunicaciones e IoT, estableciendo nuevos paradigmas en procesamiento de imágenes para logística.

1.4.3 Impactos económicos

Los impactos económicos se reflejan desde la productividad individual hasta la competitividad sectorial, aprovechando procesamiento de imágenes con IA en la nube para optimizar entregas urbanas.

1.4.4 Beneficiarios directos

Empresas de *delivery* y logística

Son los principales beneficiarios, mejorando eficiencia y rentabilidad gracias al procesamiento de imágenes con IA en la nube.

Motorizados y personal operativo

Mejoran productividad y condiciones laborales mediante rutas optimizadas que permiten más entregas en menos tiempo.

Clientes emisores de paquetes

Experimentan mayor fiabilidad y transparencia con rutas inteligentes que optimizan entregas y recojo.

Destinatarios de entregas

Reciben un servicio más rápido y predecible, cumpliendo estándares de *quick commerce* en menos de 90 minutos.

1.4.5 Beneficiarios indirectos

Sector de Telecomunicaciones

Este sector verá un aumento en la demanda de servicios especializados y mejoras en infraestructura.

Industria de Desarrollo de Software

Se generarán nuevas oportunidades y mayor demanda de talento especializado en aplicaciones móviles, NoSQL y microservicios.

Medio Ambiente y Sociedad

La optimización de rutas reducirá emisiones de CO_2 y consumo de combustible, contribuyendo a un entorno urbano más saludable.

Ecosistema de Innovación Tecnológica

La solución fortalecerá la innovación al demostrar cómo tecnologías emergentes resuelven problemas reales.

Capítulo 2. Estado del arte o de la cuestión, alternativas de solución al problema o desafío a resolver

2.1 Antecedentes de solución semejantes o similares al desafío de ingeniería

2.1.1 Sistemas de análisis visual con inteligencia artificial multimodal

Caso 1: GPT-4 Vision para análisis de objetos (OpenAI, Estados Unidos, 2023)

GPT-4 Vision, lanzado en septiembre de 2023, representa el primer modelo de lenguaje de gran escala con capacidades multimodal nativas de OpenAI. El sistema demuestra capacidades avanzadas de interpretación visual con un 95 % de precisión en reconocimiento de objetos comunes y 78.5 % de efectividad en análisis de gráficos complejos según evaluaciones técnicas independientes [10].

En términos de estimación dimensional, GPT-4V utiliza razonamiento contextual para comparar tamaños relativos entre objetos, empleando elementos conocidos como referencias de escala. Las evaluaciones técnicas reportan una precisión de ±15–25 % de error en estimaciones dimensionales cuando se proporcionan referencias visuales adecuadas, mejorando a ±10–20 % bajo condiciones de iluminación controlada.

Figura 2.1: Consulta a GPT-4 para análisis de múltiples imágenes.

La arquitectura se basa en un *transformer* multimodal que integra un codificador visual especializado con el modelo de lenguaje GPT-4, estimado en 1.76 trillones de parámetros. El sistema procesa imágenes de hasta 2048×2048 píxeles en formatos JPEG, PNG, GIF y WebP.

Las evaluaciones técnicas del sistema revelan un rendimiento variable según el contexto de aplicación:

- MMMU *Benchmark*: 56.8 % en tareas multimodal complejas
- *MathVista*: 49.9 % en razonamiento visual-matemático
- AI2D: 78.2 % en interpretación de diagramas técnicos
- ChartQA: 78.5 % en análisis de gráficos y visualizaciones

Las limitaciones incluyen:

- Sin calibración externa: Incremento del error a ±30–50 %
- Sensibilidad ambiental: Degradación con variaciones en iluminación
- Limitaciones de escala: Rendimiento reducido en objetos menores a 5cm
- Objetos problemáticos: Dificultades con elementos transparentes

Caso 2: Claude 3 Vision

Claude 3, lanzado en marzo de 2024 en sus variantes Haiku, Sonnet y Opus, establece un nuevo estándar en interpretación multimodal con un enfoque específico en razonamiento avanzado sobre contenido visual complejo. El sistema demuestra capacidades superiores a GPT-4V en interpretación de gráficos y documentos técnicos, alcanzando 86.8 % en el *benchmark* MMLU y 60.1 % en problemas matemáticos complejos.

Figura 2.2: Identificación de objetos visualmente usando modelos de Claude 3.

La arquitectura del modelo incorpora una ventana de contexto extendida de 200,000 tokens que incluye contenido visual, permitiendo el análisis simultáneo de múltiples imágenes y documentos dentro de una sola conversación.

Figura 2.3: Solicitud de reconocimiento de una imagen y reorganización en formato JSON.

Para estimación dimensional, Claude 3 utiliza razonamiento contextual sofisticado que combina reconocimiento de objetos conocidos con análisis proporcional de elementos en la imagen. El sistema puede interpretar planos técnicos con dimensiones especificadas y extrapolar esta información para estimar las dimensiones de objetos fotografiados, logrando precisiones de ±20–30 % en estimaciones sin referencias calibradas externas.

Las implementaciones documentadas incluyen:

- Análisis de especificaciones técnicas
- Interpretación de diagramas de embalaje
- Auditoría visual de inventarios
- Análisis de documentos comerciales

Ventajas competitivas: ventana de contexto extendida (200K vs 128K tokens), mejor comprensión de documentos complejos, razonamiento espacial más avanzado, y menor tendencia a alucinaciones en datos técnicos críticos.

Caso 3: Gemini 2.0 Flash

Gemini 2.0 Flash, lanzado en diciembre de 2024, representa la segunda generación de modelos multimodal de Google con optimizaciones específicas para velocidad de procesamiento y análisis visual en tiempo real [11]. El modelo incorpora una arquitectura *transformer* multimodal de segunda generación optimizada para respuestas rápidas, logrando velocidades hasta 2× superiores a Gemini 1.0.

Figura 2.4: Se solicita a Gemini reconocer las imágenes y encontrar una relación entre ellas.

El sistema soporta modalidades múltiples incluyendo texto, imagen, audio y video, con capacidad de procesamiento de imágenes de hasta 30MB en formatos JPEG, PNG, GIF, WebP, PDF, SVG y HEIC. La ventana de contexto se expande masivamente a 2 millones de tokens.

Gemini 2.0 Flash integra Google Lens como módulo nativo, eliminando la arquitectura de integración externa de generaciones anteriores. El sistema demuestra precisión mejorada en estimaciones dimensionales, alcanzando márgenes de ±8–15 % con objetos de

referencia en productos estándar, ±5–12 % en condiciones de laboratorio controlado, y ±12–25 % en uso típico de usuarios reales [12].

Implementaciones específicas en logística incluyen:

- Análisis de inventario en tiempo real para Google Shopping
- Medición automatizada de productos para Google Merchant Center
- Optimización de embalaje en centros de distribución
- Análisis de fotografías de productos para Google Lens Shopping

Caso 4: Vision Transformer (ViT) para reconocimiento y dimensionado de productos

Vision Transformer (ViT), introducido por Dosovitskiy et al. en Google Research (2020) y publicado en ICLR 2021, revolucionó el campo de *computer vision* al demostrar que arquitecturas transformer puras pueden superar las redes neuronales convolucionales tradicionales en tareas de reconocimiento de imágenes. El modelo alcanza 94.2 % de precisión en *ImageNet classification* cuando se *pre-entrena* en *datasets* suficientemente grandes [13].

La arquitectura divide imágenes en *patches* de 16×16 píxeles que se procesan como secuencias, similar al procesamiento de tokens en modelos de lenguaje. Esta metodología permite *transfer learning* eficiente para nuevas categorías de productos.

Las implementaciones académicas posteriores han demostrado aplicabilidad directa en *retail analytics*. DINOv2 (Meta AI Research, 2023) introduce *self-supervised learning* que elimina la dependencia de datasets etiquetados masivos, logrando representaciones visuales robustas especialmente efectivas para reconocimiento de productos [14].

Para estimación dimensional, las investigaciones académicas reportan precisiones de ±12–18 % en objetos con referencias visuales, utilizando visual *reasoning* sobre *features transformer* que capturan relaciones espaciales complejas.

2.1.2 Aplicaciones de LLMs multimodal en logística y comercio electrónico

Florence (Microsoft Research, 2021) establece un framework multimodal que combina *vision transformers* con capacidades de lenguaje natural, demostrando aplicabilidad directa para tareas que requieren comprensión semántica de productos y estimación de propiedades físicas.

Las evaluaciones académicas independientes confirman escalabilidad para procesamiento de millones de productos, con implementaciones que mantienen eficiencia computacional mediante técnicas de *patch embedding* optimizadas.

2.2 Características de soluciones semejantes o similares

2.2.1 Arquitecturas de procesamiento predominantes

Las soluciones analizadas convergen en arquitecturas de procesamiento que combinan múltiples enfoques tecnológicos. Los sistemas comerciales líderes utilizan *transformers* multimodal que integran vision encoders especializados (CLIP, ALIGN, ViT) con *large language models* para interpretación semántica avanzada.

Las implementaciones actuales emplean *attention mechanisms cross-modal* que permiten correlación directa entre características visuales y comprensión textual, facilitando estimación dimensional mediante razonamiento contextual [13].

2.2.2 Capacidades de interpretación dimensional

Las capacidades de interpretación dimensional se fundamentan en:

1. **Reconocimiento automático de referencias de escala**: Los modelos identifican objetos conocidos (monedas, tarjetas, manos humanas) para establecer proporciones relativas.
2. **Análisis de perspectiva y profundidad visual**: Utiliza *depth estimation* implícito derivado de características de textura y sombras.
3. **Razonamiento contextual sofisticado**: Combina reconocimiento de objetos conocidos con análisis proporcional de elementos en la imagen [14].

2.2.3 Precisión y confiabilidad según implementación

La precisión varía significativamente según la implementación y condiciones operacionales:

Tabla 2.1: Comparativa de precisión por tipo de sistema.

Tipo de Sistema	**Condiciones Óptimas**	**Condiciones Reales**
LLMs multimodal	±10–25 %	±15–30 %
Vision Transformers	±12–18 %	±15–25 %
Sistemas con hardware	±2–5mm	±5–10mm

La degradación de precisión con iluminación deficiente, ángulos subóptimos y *backgrounds* complejos representa un desafío común.

2.3 Compendio de tecnologías, herramientas, métodos, modelos utilizados con éxito

2.3.1 Modelos de lenguaje multimodal predominantes

Modelos de gran escala comerciales:

- GPT-4 Vision (OpenAI) - 1.76 trillones de parámetros
- Claude Sonnet 4 (Anthropic) - Optimizado para razonamiento visual
- Gemini 2.0 Flash (Google) - Optimizaciones de velocidad
- LLaVA - Alternativas *open-source*

Arquitecturas especializadas predominantes:

- CLIP (*Contrastive Language-Image Pre-training*)
- BLIP-2 (*Bootstrapped vision-language pre-training*)
- InstructBLIP - Seguimiento de instrucciones específicas
- MiniGPT-4 - Eficiencia en recursos limitados

2.3.2 Frameworks de visión por computadora académicos

Vision Transformers (ViT) representan el estado del arte en análisis visual académico:

- ViT-Base: 86 millones de parámetros
- ViT-Large: 307 millones de parámetros

- ViT-Huge: 632 millones de parámetros

Modelos híbridos incluyen DeiT (*Data-efficient image Transformers*), Swin Transformer optimizado para imágenes de alta resolución, y EfficientViT diseñado para implementaciones móviles.

2.3.3 Tecnologías de infraestructura en la nube

Plataformas de IA como servicio:

- OpenAI API con GPT-4 Vision endpoints
- Anthropic Claude API para análisis multimodal
- Google Vertex AI con modelos Gemini integrados
- Azure Cognitive Services
- AWS Bedrock

Servicios de computación distribuida:

- AWS Lambda - Procesamiento *serverless*
- Google Cloud Run - Containerización
- Azure Container Instances - Escalabilidad elástica
- Kubernetes - Orquestación de microservicios

2.3.4 Herramientas de desarrollo e integración

Bibliotecas de integración:

- LangChain - Framework para aplicaciones LLM
- LlamaIndex - *Retrieval-Augmented Generation*
- Haystack - Pipelines *end-to-end*
- Transformers (HuggingFace) - Acceso a modelos pre-entrenados

SDKs para desarrollo móvil:

- React Native con plugins para APIs de IA

- Flutter con packages para computer vision
- Swift/Kotlin con SDKs nativos
- Progressive Web Apps para acceso universal

2.3.5 Metodologías de prompt engineering y optimización

Técnicas avanzadas de diseño de prompts:

- *Few-shot learning* para medición dimensional
- *Chain-of-thought prompting* para razonamiento paso a paso
- *Structured output formatting* para consistencia
- *Multi-turn conversation* para refinación iterativa

Estrategias de optimización:

- *Fine-tuning* con datos del dominio logístico
- *Retrieval-Augmented Generation* (RAG)
- *In-context learning* con ejemplos calibrados
- LoRA y Adapters para personalización eficiente

2.4 Conjunto de características y especificaciones para la solución óptima

2.4.1 Arquitectura tecnológica óptima

Basándose en el análisis de soluciones existentes, la arquitectura óptima debe combinar:

- Modelos de lenguaje multimodal de última generación
- Infraestructura *cloud* escalable
- Procesamiento distribuido eficiente
- Interfaces móviles intuitivas

2.4.2 Precisión y confiabilidad requeridas

Para aplicaciones logísticas comerciales, el sistema debe alcanzar:

- Precisión de ±15–20 % en condiciones reales de uso
- Mejora a ±10–15 % con referencias de escala adecuadas
- Tiempos de respuesta inferiores a 5 segundos
- Disponibilidad superior al 99.5 %

Esta precisión es suficiente para categorización de tarifas de envío, optimización básica de carga vehicular, y estimaciones de costos logísticos sin requerir inversión en hardware especializado.

2.4.3 Limitaciones y alcances reconocidos

El sistema está diseñado para:

- Estimaciones dimensionales aproximadas para categorización logística
- Paquetes regulares de e-commerce (5cm a 2 metros)
- Condiciones de iluminación mínima adecuada
- Mercado peruano inicialmente, con escalabilidad regional

Limitaciones reconocidas:

- No apto para aplicaciones que requieran precisión milimétrica
- Degradación con objetos transparentes o altamente reflectivos
- Requiere iluminación mínima adecuada
- Formas extremadamente irregulares presentan mayor error

Capítulo 3. Diseño de la solución al desafío de ingeniería

El capítulo describe el diseño de la solución técnica al problema de la estimación manual e inexacta de dimensiones de paquetes en pymes de delivery en Lima. Basado en la metodología Design Thinking, el proceso asegura que cada decisión técnica responda a las necesidades reales de los usuarios y al contexto local.

Se siguen cuatro fases del método aplicadas al diseño: Empatizar, Definir, Idear y Prototipar, dejando la fase de Evaluación para el siguiente capítulo donde se analizará la viabilidad técnica, económica y social de la solución diseñada.

Cada etapa genera entregables que reflejan las competencias del Ingeniero de Telecomunicaciones de la PUCP, vinculando el diseño con el perfil profesional. El capítulo cierra con una reflexión crítica sobre fortalezas, limitaciones y mejoras futuras, sentando la base para la implementación del sistema.

3.1 Introducción metodológica – Enfoque Design Thinking

3.1.1 Justificación del enfoque centrado en el usuario

La aplicación de Design Thinking en este proyecto de ingeniería de telecomunicaciones permite equilibrar la complejidad técnica con las necesidades reales de las pymes de delivery en Lima. Este enfoque integra el rigor de la ingeniería —infraestructura cloud,

latencia, disponibilidad, escalabilidad y optimización en dispositivos móviles— con la comprensión de las condiciones operativas reales, evitando soluciones avanzadas pero inviables o desconectadas del contexto.

Su carácter iterativo facilita ajustes basados en retroalimentación, garantizando una solución técnicamente robusta, económicamente viable y adoptable, alineada con las competencias profesionales del ingeniero de telecomunicaciones y los Objetivos de Desarrollo Sostenible.

3.1.2 Fases de Design Thinking aplicadas en este capítulo

Este capítulo desarrolla cuatro de las cinco fases del Design Thinking, enfocándose en el diseño de la solución:

- **Empatizar**: Se busca comprender las necesidades reales de los usuarios, identificando los problemas operativos que enfrentan las empresas de delivery al estimar manualmente las dimensiones de paquetes.
- **Definir**: Se establecen los requerimientos funcionales y no funcionales que debe cumplir el sistema de gestión logística a partir de los hallazgos obtenidos.
- **Idear**: Se evalúan múltiples alternativas tecnológicas mediante criterios de escalabilidad, costo y viabilidad para seleccionar la solución óptima.
- **Prototipar**: Se materializa el diseño técnico detallado, incluyendo la arquitectura de sistema, modelo de datos, módulo de IA y especificaciones de red.

3.2 Fase 1: Empatizar – Análisis del contexto y usuarios

3.2.1 Contexto del problema y metodología de investigación

Contexto geográfico e institucional

El proyecto se desarrolla en el contexto de las pequeñas y medianas empresas de delivery que operan en Lima, Perú, una ciudad de alta densidad urbana. El mercado logístico local se caracteriza por la predominancia de procesos manuales, la alta informalidad, el uso intensivo de motorizados en motocicletas para entregas de último kilómetro y una creciente demanda de servicios de delivery impulsada por el comercio electrónico.

Estas empresas operan sin sistemas tecnológicos integrados, dependiendo de la comunicación por WhatsApp, los registros en hojas de cálculo y la coordinación telefónica, lo que genera ineficiencias, falta de trazabilidad y dificultades para escalar sus operaciones.

Técnicas de investigación aplicadas

Para comprender el contexto y las necesidades de los usuarios, se aplicaron dos técnicas principales:

1. **Análisis del flujo logístico**: Se estudió el proceso desde el registro de pedidos hasta la entrega final en pequeñas empresas de delivery, identificando ineficiencias y oportunidades de automatización.
2. **Análisis comparativo de soluciones**: Se evaluaron las herramientas tecnológicas existentes, tanto informales como comerciales, analizando sus funciones, limitaciones y costos. Este análisis evidenció vacíos en las herramientas actuales y oportunidades para integrar tecnologías emergentes como la inteligencia artificial multimodal.

Actores clave del sistema

El análisis identificó cinco actores clave en el sistema:

- **Clientes emisores**: Solicitan envíos y requieren transparencia en costos y trazabilidad de sus pedidos.
- **Administradores**: Gestionan la operación logística y necesitan herramientas para supervisión centralizada, asignación eficiente de recursos y toma de decisiones basada en datos.
- **Motorizados**: Ejecutan las entregas en campo y demandan información clara sobre asignaciones, rutas optimizadas y herramientas para registrar evidencias.
- **Destinatarios**: Reciben los paquetes y esperan puntualidad e información sobre el estado de su entrega.
- **Empresas de delivery**: Buscan crecer sosteniblemente con soluciones tecnológicas de bajo costo.

La tabla del Anexo A detalla el rol específico de cada actor, sus necesidades principales y los problemas operativos actuales que enfrentan, información fundamental para la definición de requerimientos del sistema. Esta caracterización exhaustiva de actores permite asegurar que el diseño técnico considere las perspectivas de todos los usuarios que interactuarán con el sistema, desde la interfaz móvil optimizada para operación en campo por motorizados hasta el dashboard web con análisis en tiempo real para administradores.

3.2.2 Síntesis de hallazgos

La fase de empatía reveló hallazgos críticos organizados en tres categorías que fundamentan el diseño técnico posterior: necesidades identificadas, restricciones del contexto y oportunidades de mejora mediante tecnología. Estos hallazgos, desarrollados en el Anexo B, representan la transformación de observaciones cualitativas en insumos estructurados que guiarán las decisiones de arquitectura, selección de tecnologías y priorización de funcionalidades en las fases subsecuentes del proceso de diseño.

Estos hallazgos revelan que el desafío técnico trasciende la simple automatización de mediciones, representando una oportunidad para transformar integralmente la gestión logística de pequeñas empresas mediante tecnologías de telecomunicaciones modernas que hasta ahora estaban reservadas para empresas con grandes presupuestos tecnológicos.

3.3 Fase 2: Definir – Reformulación del problema técnico

3.3.1 Declaración del problema de ingeniería

Con base en los hallazgos de la fase de empatía, se formula el siguiente problema técnico:

Las pequeñas y medianas empresas de delivery en Lima enfrentan ineficiencias operativas derivadas de la estimación manual e inexacta de dimensiones de paquetes, la falta de trazabilidad en tiempo real de las entregas, y la gestión desarticulada mediante herramientas informales (WhatsApp, hojas de cálculo). Esta problemática genera costos ocultos por viajes innecesarios, insatisfacción de clientes por reprogramaciones, y limitaciones para escalar operaciones.

Desde la perspectiva de ingeniería de telecomunicaciones, se requiere diseñar una arquitectura distribuida que integre procesamiento de imágenes con inteligencia artificial en la nube y aplicaciones móviles, todo bajo un modelo económicamente viable para empresas emergentes y cumpliendo con normativas de protección de datos personales.

3.3.2 Requerimientos funcionales del sistema

Gestión de usuarios y roles

- **RF1.1**: El sistema debe permitir el registro y autenticación de usuarios.
- **RF1.2**: El sistema debe permitir que el Administrador gestione pedidos, usuarios y visualice motorizados en tiempo real.

- **RF1.3**: El sistema debe permitir que el Cliente cree pedidos, registre datos y suba fotos.
- **RF1.4**: El sistema debe permitir que el Motorizado reciba y actualice el estado de pedidos asignados.

Gestión de pedidos

- **RF2.1**: El cliente debe poder crear pedidos en el sistema, incluyendo datos básicos: dirección de recojo, dirección de entrega, cliente, foto del pedido, número celular y distrito de entrega.
- **RF2.2**: Al crear un pedido, el cliente debe subir fotos del paquete.
- **RF2.3**: El sistema debe procesar las fotos usando IA multimodal para detectar objetos de referencia (tarjeta/moneda), activar Cloud Functions que ejecuten el modelo de IA para calcular las dimensiones del paquete basadas en la referencia detectada.
- **RF2.4**: El sistema debe guardar las dimensiones junto con la información del pedido y permitir la edición manual de las dimensiones estimadas en caso de error en el cálculo automático.
- **RF2.5**: El administrador debe poder asignar pedidos a un motorizado.

Gestión de motorizados

- **RF3.1**: El motorizado debe visualizar en la aplicación móvil los pedidos asignados.
- **RF3.2**: El motorizado debe poder capturar foto en el recojo del pedido (evidencia).
- **RF3.3**: El motorizado debe poder capturar foto en la entrega (evidencia).
- **RF3.4**: El sistema debe cambiar automáticamente el estado del pedido al momento de capturar fotos de evidencia: de "pendiente.ª .en tránsito.ªl capturar foto de recojo, y de .en tránsito.ª .entregado.ªl capturar foto de entrega.
- **RF3.5**: El motorizado puede visualizar los pedidos en un mapa.

Dashboard Web (administrador)

- **RF4.1**: El administrador debe visualizar en un dashboard la lista de pedidos, estados, dimensiones estimadas, fotos de evidencia y motorizado asignado.
- **RF4.2**: El dashboard debe permitir filtrar pedidos por estado, cliente o motorizado.

- **RF4.3**: El administrador debe poder visualizar la ubicación en tiempo real de los motorizados en un mapa dentro del dashboard web.

Notificaciones y sincronización

- **RF5.1**: El motorizado debe recibir notificaciones push automáticas cuando se le asigne un nuevo pedido.
- **RF5.2**: El cliente debe recibir notificaciones push sobre cambios de estado del pedido.

Seguridad y consistencia

- **RF6.1**: El sistema debe asegurar el acceso por roles (cada usuario solo ve lo que le corresponde).
- **RF6.2**: El sistema debe cifrar las comunicaciones mediante TLS y proteger el acceso a datos mediante reglas de seguridad de Firebase.

3.3.3 Requisitos no funcionales

Rendimiento

- **RNF1.1**: El sistema debe procesar la estimación de dimensiones de un paquete mediante IA en menos de 10 segundos después de subir la foto.
- **RNF1.2**: El dashboard web debe cargar la información principal en menos de 5 segundos.

Disponibilidad

- **RNF2.1**: El sistema debe mantener una disponibilidad mínima del 99.5 % aprovechando la infraestructura cloud de Firebase.
- **RNF2.2**: Debe existir backup automático de datos en Firebase para evitar pérdida de información.

Seguridad

- **RNF3.1**: Toda comunicación entre cliente y servidor debe usar cifrado SSL/TLS (implementado por defecto en Firebase).

- **RNF3.2**: Las contraseñas deben almacenarse de forma segura mediante Firebase Auth.
- **RNF3.3**: El acceso a datos debe estar protegido mediante reglas de seguridad basadas en roles.

Compatibilidad

- **RNF4.1**: La aplicación móvil debe ser compatible con dispositivos Android 12.0 (API level 31) o superior, considerando el uso de terminales de gama media a baja.
- **RNF4.2**: El dashboard web debe funcionar correctamente en navegadores modernos (Chrome, Firefox, Edge).

Usabilidad

- **RNF5.1**: La interfaz debe ser intuitiva y permitir completar las tareas principales sin necesidad de capacitación extensa.
- **RNF5.2**: Los mensajes de error deben ser claros y comprensibles.

Fiabilidad del Modelo de IA

- **RNF6.1**: El modelo de IA debe tener un error promedio menor al 20 % en la estimación de dimensiones cuando se proporciona un objeto de referencia reconocible.
- **RNF6.2**: El sistema debe permitir corrección manual de las dimensiones estimadas.

Cumplimiento Legal

- **RNF7.1**: El sistema debe cumplir con la Ley N.º 29733 - Ley de Protección de Datos Personales del Perú.
- **RNF7.2**: Debe implementarse un consentimiento informado al registrar usuarios, indicando:
  - Qué datos personales se recopilan (nombre, correo, teléfono, dirección, fotografías)
  - Para qué se utilizarán (gestión de pedidos, seguimiento de entregas)
  - Quién tendrá acceso (administradores, motorizados asignados)
  - Derechos del usuario (acceso, rectificación, cancelación, oposición - ARCO)

- **RNF7.3**: Las fotografías de paquetes y evidencias deben almacenarse de forma segura y solo ser accesibles por usuarios autorizados.
- **RNF7.4**: Los datos de ubicación de motorizados solo deben ser visibles durante entregas activas.
- **RNF7.5**: Debe existir una Política de Privacidad clara y accesible en la aplicación.

Mantenibilidad

- **RNF8.1**: El código debe estar documentado para facilitar su comprensión y futuras modificaciones.
- **RNF8.2**: El sistema debe generar logs básicos de errores críticos.

Eficiencia de recursos

- **RNF9.1**: Las imágenes capturadas deben comprimirse automáticamente a un tamaño máximo de 500KB para minimizar el uso de datos móviles.
- **RNF9.2**: El procesamiento intensivo (análisis de IA) debe ejecutarse en el backend (Cloud Functions) para no sobrecargar dispositivos de recursos limitados.
- **RNF9.3**: La aplicación debe utilizar thumbnails (máximo 30KB) en listas y cargar imágenes completas solo cuando el usuario lo solicite explícitamente.

3.3.4 Matriz de trazabilidad: necesidades vs. requerimientos

La tabla en el anexo C establece la trazabilidad entre las necesidades identificadas en la fase de empatía y los requerimientos funcionales y no funcionales definidos, asegurando que cada necesidad del usuario tenga una respuesta técnica específica en el diseño.

3.3.5 Restricciones de diseño

Las restricciones identificadas en la fase de empatía condicionan las decisiones técnicas del diseño de la solución y establecen los límites operativos, normativos y éticos dentro de los cuales debe funcionar el sistema.

Restricciones económicas

Las microempresas de delivery requieren tecnologías de bajo costo que respalden su operación sin comprometer su sostenibilidad financiera. Dado su limitado presupuesto

y la necesidad de costos operativos escalables, no pueden asumir gastos fijos elevados por infraestructura o licencias. Por ello, esta tesis emplea las capas gratuitas de servicios cloud (Firebase Free Tier) para desarrollar, prototipar y validar la solución sin incurrir en costos significativos durante la fase académica.

Restricciones técnicas

Los motorizados operan con conectividad móvil variable (3G/4G intermitente) en los distintos distritos de Lima, dependiendo del operador contratado y la cobertura disponible en cada zona. Esta condición externa no está bajo control del presente desarrollo, ya que responde a la infraestructura de telecomunicaciones existente. Dado el uso predominante de dispositivos Android de gama media a baja, el sistema debe contemplar capacidades de procesamiento en el borde (edge), permitiendo la compresión y reducción de imágenes antes de su envío a la nube, con el fin de optimizar el uso de datos y garantizar la viabilidad operativa.

Restricciones operativas

Los motorizados requieren flujos de interacción simples y ágiles durante las operaciones de recojo y entrega, realizadas en exteriores con condiciones variables. La aplicación debe funcionar eficientemente en dispositivos móviles estándar, sin depender de capacitación extensa, considerando la alta rotación de personal en el sector logístico. Esta restricción se aborda desde el diseño funcional del sistema, sin profundizar en aspectos de interfaz gráfica.

Restricciones normativas

El sistema gestiona información sensible como teléfonos, direcciones, razón social, imágenes y evidencias de entrega, por lo que debe cumplir estrictamente con la Ley N.º 29733 de Protección de Datos Personales. Esto incluye consentimiento informado al registro, definición clara de uso y acceso a datos, y respeto a los derechos ARCO (Acceso, Rectificación, Cancelación, Oposición).

Las imágenes procesadas por IA se almacenan en Firebase Storage con reglas de acceso por rol, y los datos de ubicación de motorizados solo se visualizan durante entregas activas, sin persistencia innecesaria.

Restricciones éticas

Las imágenes procesadas por el módulo de inteligencia artificial se utilizarán exclusivamente para estimar dimensiones de paquetes, garantizando un uso ético y limitado de esta información sensible. No se emplearán para entrenar modelos externos, compartir con terceros ni otros fines distintos a los declarados. El sistema debe informar claramente a los usuarios sobre el análisis automatizado de sus fotografías y aplicar políticas de minimización de datos para proteger la privacidad de personas que pudieran aparecer incidentalmente.

Restricciones de alcance y tiempo

El proyecto se desarrollará en un plazo estimado de cinco meses, centrado en construir un prototipo funcional que permita validar la propuesta técnica. Se priorizan funcionalidades esenciales como gestión de usuarios y roles, seguimiento de pedidos, medición automática de dimensiones con IA, asignación a motorizados, registro de evidencias y trazabilidad básica en tiempo real.

No se incluyen optimizaciones avanzadas ni características empresariales como reportes analíticos, integraciones con ERP, facturación o analítica predictiva, las cuales se consideran fuera del alcance comprometido y posibles extensiones futuras. El enfoque se limita a una experiencia operativa suficiente para evaluar la viabilidad técnica del sistema.

3.3.6 Arquitectura conceptual del sistema

Basándose en los requerimientos funcionales y no funcionales definidos, el sistema requiere una arquitectura distribuida compuesta por:

- **Capa de presentación**: Interfaces móviles y web
- **Capa de lógica de negocio**: Procesamiento serverless, reglas de negocio
- **Capa de datos**: Base de datos, almacenamiento de archivos, autenticación
- **Servicios complementarios**: Geolocalización, notificaciones, procesamiento IA

Esta arquitectura conceptual será materializada en la Fase 3 mediante la evaluación de alternativas tecnológicas específicas.

3.4 Fase 3: Idear – Generación y selección de alternativas

3.4.1 Introducción a la generación de alternativas

Con los requerimientos definidos, la fase de ideación evalúa alternativas tecnológicas que respondan a las necesidades identificadas. Se comparan sistemáticamente opciones de arquitectura, telecomunicaciones, plataformas cloud y enfoques de implementación mediante criterios técnicos, económicos y operativos. Este análisis multicriterio permite justificar la solución seleccionada con rigor ingenieril, evitando decisiones basadas en preferencias subjetivas o familiaridad tecnológica.

3.4.2 Criterios de evaluación

Para la evaluación de alternativas se establecen seis criterios principales que reflejan los requerimientos y restricciones identificados en la fase anterior:

Tabla 3.1: Criterios de evaluación de alternativas tecnológicas.

Criterio	**Descripción**	**Peso**	**Justificación**
Escalabilidad	Capacidad del sistema para crecer automáticamente con la demanda sin requerir reconfiguración manual significativa	20 %	Fundamental para pequeñas empresas que planean crecer progresivamente sin inversiones disruptivas
Costo	Inversión inicial, costos operativos mensuales y modelo de pago (fijo vs. variable)	25 %	Criterio crítico dado el presupuesto limitado de las empresas objetivo y el uso de capas gratuitas para el desarrollo de tesis
Facilidad de integración	Complejidad de integrar los diferentes componentes del sistema y tiempo de desarrollo requerido	15 %	Importante considerando el plazo de 5 meses para el desarrollo de la tesis
Mantenimiento	Esfuerzo requerido para mantener la infraestructura, actualizar componentes y gestionar la seguridad	15 %	Relevante para empresas pequeñas con recursos técnicos limitados
Rendimiento	Latencia, throughput y tiempo de respuesta del sistema en condiciones reales de uso	15 %	Necesario para garantizar experiencia de usuario aceptable y sincronización en tiempo real
Ecosistema y soporte	Disponibilidad de documentación, comunidad de desarrolladores, SDKs y herramientas de desarrollo	10 %	Facilita la resolución de problemas y reduce riesgos durante la implementación

3.4.3 Alternativas consideradas

Se identifican tres alternativas principales de arquitectura que cumplen con los requerimientos funcionales establecidos.

Alternativa 1: Ecosistema Firebase/Google Cloud

Descripción técnica: Arquitectura completamente basada en servicios de Google Cloud Platform, utilizando Firebase como Backend-as-a-Service (BaaS). Los componentes principales incluyen:

- Firebase Firestore (base de datos NoSQL en tiempo real)
- Firebase Authentication (gestión de usuarios y roles)
- Firebase Storage (almacenamiento de imágenes)
- Firebase Cloud Functions (lógica serverless)
- Firebase Cloud Messaging (notificaciones push)
- Firebase Hosting (dashboard web)
- Google Maps API (geolocalización)
- Integración con modelos de IA multimodal (Gemini, Claude, GPT-4 Vision) mediante Cloud Functions

Arquitectura de telecomunicaciones:

- Sincronización en tiempo real mediante WebSockets gestionados por Firestore
- Protocolo HTTPS/TLS para todas las comunicaciones
- CDN global de Google para distribución de contenido estático
- Latencia típica de 50-200ms según ubicación geográfica
- Escalamiento automático horizontal sin configuración

Ventajas:

- Integración nativa entre todos los componentes del ecosistema
- Capa gratuita generosa para desarrollo y validación de tesis
- Sincronización en tiempo real nativa (offline-first)
- Escalabilidad automática sin gestión de servidores
- SDKs optimizados para Android y Web

- Reglas de seguridad declarativas integradas con autenticación
- Despliegue simplificado con Firebase CLI

Desventajas:

- Vendor lock-in con Google Cloud Platform
- Menor flexibilidad en configuraciones avanzadas
- Costos pueden incrementarse significativamente al superar la capa gratuita
- Modelo de datos NoSQL puede requerir desnormalización

Estimación de costos:

- Desarrollo (capa gratuita): $0/mes
- Operación inicial (<1000 usuarios activos/mes): $0-25/mes
- Operación media (1000-5000 usuarios): $50-150/mes

Alternativa 2: Amazon Web Services (AWS)

Descripción técnica: Arquitectura basada en servicios de AWS utilizando:

- Amazon RDS o DynamoDB (base de datos)
- AWS Cognito (autenticación)
- Amazon S3 (almacenamiento)
- AWS Lambda (funciones serverless)
- Amazon SNS (notificaciones)
- Amazon CloudFront (CDN)
- Amazon API Gateway (exposición de APIs)
- EC2 o Elastic Beanstalk (dashboard web)
- Integración con servicios de IA mediante Amazon Rekognition o APIs externas

Arquitectura de telecomunicaciones:

- API REST mediante API Gateway con throttling configurable

- Sincronización mediante polling o WebSockets (AWS AppSync)
- Protocolo HTTPS con certificados SSL de AWS Certificate Manager
- Latencia variable según configuración de regiones
- Escalamiento mediante Auto Scaling Groups

Ventajas:

- Ecosistema maduro con amplia documentación
- Flexibilidad máxima en configuraciones avanzadas
- Servicios especializados para diversos casos de uso
- Posibilidad de usar bases de datos relacionales (RDS) o NoSQL (DynamoDB)
- Control granular sobre infraestructura de red
- Amplio soporte empresarial

Desventajas:

- Mayor complejidad de configuración e integración
- Curva de aprendizaje pronunciada
- Capa gratuita limitada a 12 meses
- Requiere más tiempo de desarrollo por configuraciones manuales
- Sincronización en tiempo real no nativa (requiere configuración adicional)
- Costos más difíciles de predecir

Estimación de costos:

- Desarrollo (primeros 12 meses con capa gratuita): $10-30/mes
- Operación inicial: $50-100/mes
- Operación media: $150-300/mes

Alternativa 3: Solución híbrida con backend propio

Descripción técnica: Backend personalizado desarrollado con Node.js/Express o Python/Django desplegado en servicios de hosting como Heroku, DigitalOcean o Railway. Componentes:

- Base de datos PostgreSQL o MongoDB gestionada
- Autenticación con JWT
- Almacenamiento en AWS S3 o Cloudinary
- Integración directa con APIs de IA multimodal
- Notificaciones mediante servicios de terceros (OneSignal, Firebase Cloud Messaging)
- Frontend web con React deployado en Vercel o Netlify

Arquitectura de telecomunicaciones:

- API REST con autenticación mediante tokens JWT
- Sincronización mediante polling o WebSockets con Socket.io
- Protocolo HTTPS configurado manualmente
- Latencia dependiente del proveedor de hosting
- Escalamiento mediante configuración manual de instancias

Ventajas:

- Control total sobre la lógica de negocio
- No hay vendor lock-in (portabilidad entre proveedores)
- Posibilidad de usar tecnologías específicas según preferencia
- Costos predecibles con planes de hosting fijos
- Integración flexible con múltiples servicios de terceros

Desventajas:

- Mayor esfuerzo de desarrollo e integración
- Requiere gestionar múltiples servicios de diferentes proveedores

- Sincronización en tiempo real requiere implementación manual compleja
- Mayor responsabilidad en seguridad y mantenimiento
- Escalabilidad manual (no automática)
- Tiempo de desarrollo significativamente mayor (2-3x)
- Costos de mantenimiento y actualización recaen en el desarrollador

Estimación de costos:

- Desarrollo: $15-40/mes (hosting + base de datos + almacenamiento)
- Operación inicial: $40-80/mes
- Operación media: $100-200/mes + tiempo de mantenimiento

3.4.4 Matriz de evaluación multicriterio

La siguiente tabla evalúa cada alternativa según los criterios establecidos, utilizando una escala de 1 a 5 donde 5 representa el mejor desempeño en ese criterio. La justificación detallada de cada puntuación asignada se encuentra en el Anexo G.

Tabla 3.2: Matriz de evaluación multicriterio de alternativas.

Criterio	**Peso**	**Alt 1: Firebase**		**Alt 2: AWS**		**Alt 3: Propio**	
		Punt.	**Pond.**	**Punt.**	**Pond.**	**Punt.**	**Pond.**
Escalabilidad	20 %	5	1.00	4	0.80	2	0.40
Costo	25 %	5	1.25	3	0.75	3	0.75
Facilidad integración	15 %	5	0.75	3	0.45	2	0.30
Mantenimiento	15 %	5	0.75	3	0.45	2	0.30
Rendimiento	15 %	4	0.60	4	0.60	3	0.45
Ecosistema y soporte	10 %	5	0.50	5	0.50	3	0.30
TOTAL	**100 %**		**4.85**		**3.55**		**2.50**

3.4.5 Selección de la solución óptima

Con base en la evaluación multicriterio, se selecciona la alternativa del ecosistema Firebase/Google Cloud como la arquitectura óptima para el desarrollo de este proyecto, obteniendo una puntuación ponderada de 4.85/5.00, significativamente superior a las alternativas evaluadas.

Justificación técnica de la selección

Firebase ofrece una infraestructura distribuida en la nube que satisface los requerimientos críticos del sistema desde una perspectiva de ingeniería de telecomunicaciones. Su arquitectura serverless permite escalabilidad automática sin necesidad de provisión previa, con sincronización en tiempo real mediante WebSockets (latencias de 50-200ms), alta disponibilidad y replicación multi-región. La seguridad está integrada mediante OAuth 2.0, reglas de acceso por rol y cifrado TLS, cumpliendo con los requisitos RNF1.2, RNF2.1 y RNF3.x.

Además, los SDKs nativos para Android están optimizados para dispositivos de gama media-baja, con soporte offline-first, lo que permite una operación eficiente en condiciones de conectividad variable. Esta solución se alinea con las restricciones operativas identificadas: bajo costo (Spark Plan gratuito), bajo mantenimiento (infraestructura gestionada), cumplimiento normativo (Ley N.º 29733) y plazo limitado de desarrollo (5 meses), gracias a la integración nativa entre componentes.

Se reconoce el riesgo de vendor lock-in con Google Cloud Platform, pero se considera aceptable en el contexto de un prototipo académico. Este riesgo se mitiga mediante el uso de patrones de diseño como Repository Pattern, que facilitan la portabilidad futura del código de negocio.

3.4.6 Selección de componentes específicos dentro del ecosistema Firebase

Una vez seleccionada la arquitectura general, se detallan los componentes específicos que se utilizarán:

Tabla 3.3: Componentes seleccionados del ecosistema Firebase.

Componente del sistema	**Tecnología seleccionada**	**Justificación**
Base de datos en tiempo real	Firebase Firestore	NoSQL en tiempo real con sincronización automática, escalabilidad horizontal, modelo flexible para datos logísticos variables
Servicio de Autenticación	Firebase Authentication	Gestión de sesiones segura, soporte de roles mediante custom claims, integración nativa con Firestore Security Rules
Servicio de almacenamiento	Firebase Storage	Almacenamiento escalable de imágenes con CDN global, reglas de seguridad integradas, URLs firmadas para acceso controlado
Lógica de negocio serverless	Firebase Cloud Functions	Ejecución de código bajo demanda, triggers automáticos ante eventos en Firestore/Storage, integración con APIs de IA externas
Aplicación móvil	Android Nativo (Kotlin)	Rendimiento óptimo, acceso completo a APIs del dispositivo (cámara, GPS), SDKs de Firebase nativos, amplia documentación
App Web	React.js + Firebase Hosting	Framework moderno con alto rendimiento, despliegue global automático con CDN, certificado SSL incluido
Notificaciones push	Firebase Cloud Messaging	Integración nativa con Android, envío masivo eficiente, segmentación por roles, delivery garantizado
Geolocalización	Google Maps API	Precisión superior en ubicaciones de Lima, integración directa con ecosistema Google, APIs de rutas y geocodificación
Procesamiento de IA	GPT-4 / Gemini / Claude	Modelos multimodales capaces de analizar imágenes y estimar dimensiones mediante razonamien-

3.4.7 Conclusión de la fase de ideación

La selección del ecosistema Firebase/Google Cloud se fundamenta en un análisis multicriterio riguroso que pondera aspectos técnicos, económicos y operativos relevantes para el contexto del proyecto. Esta decisión equilibra las necesidades de pequeñas empresas de delivery con las restricciones de un proyecto de tesis, proporcionando una base tecnológica robusta, escalable y económicamente viable para el desarrollo del prototipo y su eventual adopción comercial.

3.5 Fase 4: Prototipar – Diseño técnico detallado

3.5.1 Introducción al diseño de la solución

La fase de prototipado materializa la alternativa seleccionada en un diseño técnico completo que especifica la arquitectura del sistema, los componentes de telecomunicaciones, el modelo de datos, las interfaces de usuario y los protocolos de comunicación. Este diseño debe ser lo suficientemente detallado para permitir su implementación y validación posterior, asegurando que todos los requerimientos funcionales y no funcionales establecidos en la fase de definición sean satisfechos mediante decisiones técnicas justificadas.

El prototipo conceptual incluye diagramas de arquitectura, especificaciones de red, diseño de base de datos, flujos de información y mockups de interfaces, proporcionando una visión integral de la solución propuesta desde la perspectiva de ingeniería de telecomunicaciones.

3.5.2 Arquitectura general del sistema

La arquitectura del sistema se basa en un modelo cliente-servidor distribuido en la nube con tres capas principales que interactúan mediante protocolos estándar de telecomunicaciones.

Diagrama de arquitectura de alto nivel

Capa de presentación (Frontend):

- Aplicación móvil Android nativa para clientes y motorizados
- Dashboard web React.js para administradores
- Interfaces responsive con Material Design

Capa de lógica de negocio (Backend):

- Firebase Cloud Functions (serverless)
- Procesamiento de imágenes con IA
- Reglas de negocio y validaciones
- Integración con servicios externos (Google Maps, APIs de IA)

Capa de datos (Data Layer):

- Firebase Firestore (base de datos NoSQL)
- Firebase Storage (almacenamiento de imágenes)
- Firebase Authentication (gestión de usuarios)

Flujo de datos y comunicación

El sistema opera mediante comunicación asíncrona basada en eventos:

1. **Autenticación**: Firebase Authentication maneja login/logout con tokens JWT
2. **Sincronización en tiempo real**: Firestore utiliza WebSockets para actualizaciones bidireccionales
3. **Almacenamiento de archivos**: Firebase Storage recibe imágenes con URLs firmadas
4. **Procesamiento serverless**: Cloud Functions se activan mediante triggers automáticos
5. **Notificaciones**: Firebase Cloud Messaging envía push notifications a dispositivos registrados

Protocolos de comunicación utilizados:

Tabla 3.4: Protocolos de comunicación del sistema.

Protocolo	**Uso**	**Justificación técnica**
HTTPS (TLS 1.3)	Todas las comunicaciones cliente-servidor	Cifrado extremo a extremo, autenticación de servidor
WebSockets sobre TLS	Sincronización en tiempo real de Firestore	Comunicación bidireccional de baja latencia
OAuth 2.0	Autenticación y autorización	Estándar de la industria, secure token management
REST API	Integración con servicios externos	Interoperabilidad con Google Maps y APIs de IA

3.5.3 Flujo crítico del sistema: Creación de pedido con procesamiento de IA

El siguiente diagrama describe el flujo completo desde que un cliente captura la foto de un paquete hasta que recibe las dimensiones estimadas:

1. Cliente captura foto del paquete con objeto de referencia
2. App móvil comprime imagen localmente (máx. 500KB)
3. Imagen se sube a Firebase Storage
4. Cloud Function se activa automáticamente por trigger
5. Function descarga imagen y envía a API de IA multimodal
6. Modelo IA procesa imagen y estima dimensiones
7. Dimensiones se guardan en Firestore
8. Cliente recibe actualización en tiempo real vía WebSocket
9. Sistema permite edición manual si confianza es baja

3.5.4 Módulo de IA para estimación de dimensiones

Enfoque de prompt engineering

El módulo de IA utiliza modelos de lenguaje multimodal (Gemini 2.0 Flash, Claude Sonnet 4) mediante técnicas de prompt engineering estructurado para estimar dimensiones de paquetes a partir de fotografías.

Estructura del prompt:

- **Rol del sistema**: "Eres un asistente especializado en logística que estima dimensiones de paquetes..."
- **Contexto**: "El motorizado está en el domicilio del remitente fotografiando un paquete para delivery en Lima, Perú..."
- **Tarea específica**: "Analiza la imagen y estima las dimensiones (alto, ancho, largo) en centímetros..."
- **Formato de salida**: JSON estructurado

```
{
    "alto_cm": número,
    "ancho_cm": número,
    "largo_cm": número,
    "confianza": "alta|media|baja",
    "observaciones": "texto"
}
```

- **Restricciones**: "Si no puedes estimar con confianza razonable, indica confianza 'baja' y explica el motivo..."

Validación y edición manual

El sistema permite corrección manual de las dimensiones estimadas:

- **Confianza alta (>80 %)**: Dimensiones se aceptan automáticamente pero son editables
- **Confianza media (50-80 %)**: Sistema sugiere verificación manual
- **Confianza baja (<50 %)**: Requiere medición manual obligatoria

3.5.5 Interfaces funcionales del sistema

El sistema requiere tres interfaces funcionales que permitan demostrar el flujo de comunicación y procesamiento de datos diseñado. El enfoque está en la funcionalidad técnica, no en el diseño visual, ya que el objetivo es validar la arquitectura de telecomunicaciones propuesta.

Aplicación móvil (Cliente y Motorizado)

Funcionalidades técnicas implementadas:

- Autenticación mediante Firebase Authentication (OAuth 2.0)
- Captura de imagen y compresión local en el dispositivo
- Upload de archivos a Firebase Storage mediante HTTPS
- Recepción de datos en tiempo real vía WebSocket (Firestore)
- Envío y recepción de geolocalización (Google Maps API)
- Recepción de notificaciones push (Firebase Cloud Messaging)

Propósito: Demostrar la comunicación entre dispositivo móvil y servicios cloud, validando latencias, consumo de datos y sincronización en tiempo real.

Dashboard web (Administrador)

Funcionalidades técnicas implementadas:

- Visualización de datos en tiempo real desde Firestore (WebSocket)
- Renderización de mapa con ubicaciones geográficas (Google Maps API)
- Consulta y filtrado de información desde base de datos NoSQL
- Visualización de imágenes almacenadas en Firebase Storage
- Gestión de usuarios mediante Firebase Authentication

Propósito: Demostrar sincronización bidireccional, actualización en tiempo real de múltiples clientes conectados simultáneamente y consultas eficientes a base de datos distribuida.

3.5.6 Especificaciones de seguridad y cumplimiento normativo

El diseño incorpora mecanismos de seguridad que cumplen con la Ley N.º 29733 de Protección de Datos Personales de Perú.

Seguridad en capa de transporte

Tabla 3.5: Especificaciones de seguridad en transporte.

Mecanismo	Especificación	Justificación
Cifrado	TLS 1.3	Versión más reciente, elimina vulnerabilidades de TLS 1.2
Certificados	Automáticos (Let's Encrypt)	Renovación automática sin intervención manual
Perfect Forward Secrecy	Habilitado	Protege comunicaciones pasadas si clave se compromete

Especificaciones técnicas consolidadas

- **OAuth 2.0**: Autenticación delegada con tokens JWT
- **Roles y permisos**: Administrador, Cliente, Motorizado
- **Security Rules de Firestore**: Control granular de acceso a nivel de documento
- **Sesiones**: Expiración automática tras 24 horas de inactividad

Protección de datos personales

Según Ley N.º 29733:

Tabla 3.6: Cumplimiento de Ley N.º 29733.

Requisito legal	Implementación técnica
Consentimiento informado	Aceptación explícita en registro
Derechos ARCO (Acceso, Rectificación, Cancelación, Oposición)	Opciones en configuración de cuenta
Seguridad técnica	Cifrado TLS + reglas de acceso
Plazo de conservación	Eliminación automática tras 90 días de inactividad

3.5.7 Conclusión de la fase de prototipado

El diseño técnico propuesto combina robustez tecnológica, viabilidad económica y eficiencia operativa mediante una arquitectura serverless basada en Firebase, que ofrece escalabilidad automática, costos proporcionales al uso, sincronización en tiempo real, seguridad normativa y desarrollo ágil. Los componentes detallados se documentan en el Anexo D, permitiendo que este capítulo se centre en la justificación de alto nivel.

El diseño técnico propuesto establece una base sólida que será evaluada en el Capítulo 4 mediante análisis de viabilidad técnica, económica y de sostenibilidad, validando su capacidad para cumplir los requerimientos establecidos en el contexto de las pymes de delivery en Lima.

Conclusiones

[Por completar]

REFERENCIAS BIBLIOGRAFICAS

[1] «IoT-based TPL whole supply chain logistics information system model,» *Proceedings - International Conference on Machine Learning and Cybernetics*, vol. 4, págs. 1758-1762, jul. de 2013, ISSN: 21601348. DOI: 10.1109/ICMLC.2013.6890882. dirección: https://ieeexplore.ieee.org/document/6890882.

[2] «Applications of Computer Internet of Things Technologies in Logistics,» *2nd IEEE International Conference on Advanced Technologies in Intelligent Control, Environment, Computing and Communication Engineering, ICATIECE 2022*, 2022. DOI: 10.1109/ICATIECE56365.2022.10047270. dirección: https://ieeexplore.ieee.org/document/10047270.

[3] R. TLW, *Desafíos y soluciones en la distribución de última milla: ¿Cómo adaptarse al futuro?* Feb. de 2025. dirección: https://thelogisticsworld.com/logistica-y-distribucion/desafios-y-soluciones-en-la-distribucion-de-ultima-milla-en-entornos-urbanos/.

[4] «Design and actualization of IoT-based intelligent logistics system,» *IEEE International Conference on Industrial Engineering and Engineering Management*, págs. 2245-2248, 2012, ISSN: 2157362X. DOI: 10.1109/IEEM.2012.6838146.

[5] «Research on Cloud Logistics-based One-stop Service Platform for Logistics Center,» *Proceedings of the 2012 IEEE 16th International Conference on Computer Supported Cooperative Work in Design, CSCWD 2012*, págs. 558-563, 2012. DOI: 10.1109/CSCWD.2012.6221873.

[6] J. Krysińska, *On-premise vs. cloud: definition & differences | NordLayer*, ago. de 2024. dirección: https://nordlayer.com/blog/on-premise-vs-cloud-differences/.

[7] E. Perú, «REGLAMENTO DE LA LEY Nº 29733 LEY DE PROTECCIÓN DE DATOS PERSONALES,» 2973. dirección: https://cdn.www.gob.pe/uploads/document/file/272360/Ley%20N%C2%BA%2029733.pdf.pdf?v=1618338779.

[8] R. TLW, *Supervisión de seguridad en tiempo real en la cadena de suministro: Innovación con IoT*, nov. de 2024. dirección: https://thelogisticsworld.com/tecnologia/el-uso-de-iot-para-la-supervision-de-seguridad-en-tiempo-real-en-la-cadena-de-suministro/.

[9] «Green and Network-Aware Smart IoT Logistics Applications,» *2023 International Conference on Information Technology: Cybersecurity Challenges for Sustainable Cities, ICIT 2023 - Proceeding*, págs. 441-446, 2023. DOI: 10.1109/ICIT58056.2023.10225953.

[10] «MM-Vet: Evaluating Large Multimodal Models for Integrated Capabilities,» 2024. dirección: https://huggingface..

[11] «Gemini: A Family of Highly Capable Multimodal Models,» 2025.

[12] G. Team, «Gemini 2.5: Pushing the Frontier with Advanced Reasoning, Multimodality, Long Context, and Next Generation Agentic Capabilities.,» 2025.

[13] «AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE,» dirección: https://github.com/.

[14] «DINOv2: Learning Robust Visual Features without Supervision,» dirección: https://openreview.net/forum?id=a68SUt6zFt.

Apéndice A. Caracterización detallada de actores del sistema

Actor: Clientes (Emisores)

Rol en el sistema	Usuarios que solicitan el servicio de envío de paquetes
Necesidades principales	Necesitan enviar paquetes de manera rápida y confiable sin invertir tiempo en mediciones manuales. Requieren conocer el estado de sus pedidos en tiempo real y tener transparencia en los costos del servicio.
Puntos de dolor actuales	Actualmente deben medir manualmente los paquetes o proporcionar estimaciones que pueden no corresponder con la realidad. Experimentan falta de trazabilidad durante la entrega y sufren reprogramaciones cuando los paquetes están mal estimados, generando incertidumbre sobre las condiciones del servicio.

Actor: Administradores

Rol en el sistema	Gestores de la operación logística de la empresa
Necesidades principales	Buscan maximizar la eficiencia operativa atendiendo el mayor número de pedidos posible. Necesitan asignar pedidos adecuadamente a motorizados, supervisar entregas en tiempo real y poder rechazar pedidos que excedan la capacidad operativa del servicio.
Puntos de dolor actuales	Enfrentan una gestión manual de pedidos mediante WhatsApp y hojas de Excel que no escala. Carecen de visibilidad en tiempo real de la operación y tienen dificultad para validar las dimensiones reportadas por los clientes, lo que genera pérdida de eficiencia por aceptar paquetes sobredimensionados.

Actor: Motorizados (Repartidores)

Rol en el sistema	Operadores de campo que recogen y entregan paquetes
Necesidades principales	Requieren recibir información clara sobre los pedidos asignados para optimizar el espacio en su mochila o cajas de reparto. Necesitan conocer rutas y ubicaciones con precisión, registrar evidencias de recojo y entrega fácilmente, y evitar viajes innecesarios por paquetes que no pueden transportar.
Puntos de dolor actuales	Experimentan sorpresas frecuentes con paquetes más grandes de lo reportado, lo que genera pérdida de tiempo y combustible en recojos no viables. Carecen de herramientas adecuadas para comunicar el estado del pedido en tiempo real y enfrentan dificultad para gestionar múltiples entregas simultáneas de manera eficiente.

Actor: Destinatarios (Receptores)

Rol en el sistema	Usuarios finales que reciben los paquetes
Necesidades principales	Esperan recibir sus paquetes en el tiempo estimado y conocer cuándo llegará el motorizado. Necesitan tener confianza en el servicio y sentirse informados durante el proceso de entrega.
Puntos de dolor actuales	Sufren reprogramaciones frecuentes debido a problemas logísticos en la cadena de entrega. Experimentan falta de información sobre el estado real de su pedido y quedan insatisfechos cuando el servicio se retrasa sin explicación clara.

Apéndice B. Síntesis de hallazgos de la fase de empatía

B.1 Necesidades identificadas

- **Automatización de procesos manuales**: La medición de dimensiones de paquetes consume tiempo y genera estimaciones inexactas, mientras que la gestión de pedidos mediante WhatsApp y hojas de cálculo resulta ineficiente para escalar operaciones.
- **Visibilidad y trazabilidad en tiempo real**: Clientes requieren conocer el estado de sus envíos, administradores necesitan supervisar la operación completa, y motorizados demandan información clara sobre asignaciones y rutas.
- **Optimización de recursos limitados**: Las empresas necesitan soluciones de bajo costo con modelos de pago progresivos, y los motorizados requieren maximizar el aprovechamiento del espacio de transporte para cumplir múltiples entregas eficientemente.

B.2 Restricciones del contexto

- **Restricción económica**: Presupuestos limitados para inversión tecnológica inicial y necesidad de evitar costos fijos elevados que comprometan la viabilidad del negocio.
- **Restricción técnica**: El uso predominante de dispositivos Android de gama media a baja condiciona las decisiones de diseño hacia arquitecturas que minimicen el procesamiento local. Se implementan estrategias básicas de eficiencia (compresión de imágenes, procesamiento en cloud, uso de SDKs nativos).
- **Restricción operativa**: Conectividad variable en diferentes zonas de Lima que requiere estrategias de sincronización y manejo de estados offline para garantizar continuidad del servicio.
- **Restricción normativa**: Cumplimiento obligatorio de la Ley N.º 29733 de Protección de Datos Personales en el manejo de ubicaciones, fotografías de evidencia y datos personales de usuarios.

B.3 Oportunidades de mejora mediante tecnología

- **Inteligencia artificial para visión por computadora**: Automatización de estimación de dimensiones mediante fotografías con objetos de referencia, eliminando tiempo de medición manual y reduciendo errores humanos.
- **Infraestructura cloud escalable (Firebase)**: Sincronización en tiempo real entre dispositivos móviles y dashboard web sin requerir servidores físicos costosos, proporcionando trazabilidad completa de operaciones.
- **Notificaciones push (Firebase Cloud Messaging)**: Comunicación instantánea de cambios de estado a todos los actores, mejorando coordinación operativa y experiencia de usuario.
- **Geolocalización (Google Maps API)**: Optimización de rutas de entrega y seguimiento en tiempo real de motorizados, aumentando eficiencia del servicio y reduciendo tiempos de entrega.
- **Modelo de costos por uso**: Servicios cloud con pago progresional que se alinea con el crecimiento de empresas pequeñas, eliminando barreras de entrada por inversiones iniciales prohibitivas.

Apéndice C. Matriz de trazabilidad completa

Tabla C.1: Matriz de trazabilidad: Necesidades vs. Requerimientos.

Necesidad identificada	**Req. funcionales**	**Req. no funcionales**	**Justificación técnica**
Automatización de medición de dimensiones	RF2.3, RF2.4	RNF1.1, RNF6.2, RNF7.1, RNF7.2	El módulo de IA con Cloud Functions procesa imágenes automáticamente, reduciendo tiempo de medición manual y errores humanos, con compresión optimizada para minimizar consumo de datos.
Trazabilidad en tiempo real de pedidos	RF2.1, RF2.5, RF3.4, RF4.1, RF5.1, RF5.2	RNF1.2, RNF2.1	La sincronización en tiempo real de Firestore combinada con notificaciones push proporciona visibilidad completa del estado de pedidos a todos los actores.
Gestión centralizada para administradores	RF4.1, RF4.2, RF4.3	RNF1.2, RNF4.2, RNF5.1	El dashboard web con React permite supervisión completa, filtrado eficiente y visualización geográfica de operaciones desde cualquier navegador.
Herramientas operativas para motorizados	RF3.1, RF3.2, RF3.3, RF3.5	RNF4.1, RNF5.1, RNF6.1	La aplicación Android nativa optimizada proporciona acceso eficiente a funciones del dispositivo (cámara, GPS) con bajo consumo de batería.
Solución de bajo costo escalable	RF1.1, RF6.1, RF6.2	RNF2.1, RNF3.1, RNF9.1, RNF9.2	La arquitectura serverless de Firebase elimina costos de servidores físicos y escala automáticamente con modelo de pago por uso.

Continuación de la tabla

Necesidad identificada	**Req. funcionales**	**Req. no funcionales**	**Justificación técnica**
Seguridad de información sensible	RF6.1, RF6.2	RNF3.1, RNF3.2, RNF3.3, RNF7.1-RNF7.5	Firebase Auth, reglas de seguridad de Firestore, cifrado SSL/TLS y cumplimiento de Ley N.º 29733 protegen datos personales y operativos.
Optimización de espacio de transporte	RF2.3, RF2.4, RF3.5	RNF6.1	Las dimensiones precisas permiten a motorizados planificar mejor la capacidad de carga y a administradores asignar pedidos compatibles.
Compatibilidad con dispositivos limitados	RF3.1, RF3.2, RF3.3	RNF4.1, RNF9.1, RNF9.2	El diseño considera dispositivos Android de gama media-baja, optimizando procesamiento, memoria y consumo de batería.

Apéndice D. Modelo de datos detallado

D.1 Colección: USUARIOS

Ruta: usuarios/{uid}

```
{
  uid: string,                // ID unico de Firebase Auth
  email: string,              // Correo electronico
  phone: string,              // Telefono de contacto
  nombre: string,             // Nombre del usuario
  apellido: string,           // Apellido del usuario
  rol: string,                // 'administrador' | 'cliente' | '
   motorizado'

  // Datos especificos para clientes empresariales (opcional)
  datosEmpresariales: {
    nombreEmpresa: string,
    ruc: string,
    razonSocial: string
  } | null,

  // Datos especificos para motorizados (opcional)
  datosMotorizado: {
    estadoServicio: string, // 'disponible' | 'ocupado' | '
   desconectado'
    vehiculo: {
      placa: string,
      tipo: string,           // 'motocicleta' | 'bicicleta' | '
   auto'
      modelo: string,
      anio: number
    },
    estadisticas: {
      pedidosCompletados: number,
      calificacionPromedio: number,
      tiempoPromedioEntrega: number  // en minutos
    }
  } | null,
```

```
31
32    // Ultima ubicacion conocida (para motorizados)
33    ultimaUbicacionConocida: {
34      latitud: number,
35      longitud: number,
36      timestamp: timestamp,
37      distrito: string
38    } | null,
39
40    // Metadatos
41    creadoEn: timestamp,
42    actualizadoEn: timestamp,
43    activo: boolean,
44    ultimaConexion: timestamp
45  }
```

Listing D.1: Estructura de documento Usuario

D.2 Colección: PEDIDOS

Ruta: pedidos/{pedidoId}

```
{
  id: string,  // ID unico del pedido

  // Estructura de asignacion (recojo y entrega pueden ser
   diferentes)
  asignacion: {
    recojo: {
      estado: string,       // 'pendiente' | 'asignado' | '
   completado'
      rutaId: string | null,
      rutaNombre: string | null,
      motorizadoUid: string | null,
      motorizadoNombre: string | null,
      asignadaEn: timestamp | null,
      razonPendiente: string | null
    },
    entrega: {
      // Estructura similar a recojo
    }
  },
```

```
19
20    // Control de visibilidad por rol
21    visibilidad: {
22      motorizadoRecojo: boolean,
23      motorizadoEntrega: boolean,
24      administrador: boolean
25    },
26
27    // Ciclo operativo del pedido
28    cicloOperativo: {
29      diaCreacion: string,    // YYYY-MM-DD
30      diaAsignado: string,
31      cerradoPorAdmin: boolean,
32      fechaCierreAdmin: timestamp | null,
33      horaLimiteRecojo: timestamp,
34      permiteEntregaAntes13h: boolean
35    },
36
37    // Indices para consultas eficientes
38    indices: {
39      requiereAsignacionManual: boolean,
40      motorizadoRecojoUid: string | null,
41      motorizadoEntregaUid: string | null,
42      rutaRecojoId: string | null,
43      rutaEntregaId: string | null,
44      distritoRecojo: string,
45      distritoEntrega: string
46    }
47  }
```

Listing D.2: Estructura de documento Pedido (parte 1)

```
1   {
2     // Informacion del proveedor (quien envia)
3     proveedor: {
4       uid: string,
5       nombre: string,
6       correo: string,
7       telefono: string,
8       direccion: {
9         link: string,          // URL de Google Maps
10        distrito: string,
```

```
      coordenadas: {
        latitud: number,
        longitud: number
      }
    }
  },

  // Informacion del destinatario (quien recibe)
  destinatario: {
    nombre: string,
    telefono: string,
    direccion: {
      link: string,
      distrito: string,
      coordenadas: {
        latitud: number,
        longitud: number
      }
    }
  },

  // Informacion del paquete
  paquete: {
    detalle: string,        // Descripcion del contenido
    observaciones: string,
    dimensiones: {
      alto: number,         // en centimetros
      ancho: number,
      largo: number,
      volumen: number,      // calculado automaticamente
      peso: number | null,  // opcional
      estimadoPorIA: boolean,
      editadoManualmente: boolean,
      excedeMaximo: boolean,
      confianzaIA: number | null  // 0-1, nivel de confianza
 del modelo
    },
    fotos: {
      recojo: {
        url: string,
        thumbnail: string,
```

```
        timestamp: timestamp,
        procesadaPorIA: boolean
      },
      entrega: {
        url: string,
        thumbnail: string,
        timestamp: timestamp
      },
      comprobantePago: {
        url: string,
        thumbnail: string,
        timestamp: timestamp
      }
    }
  }
}
```

Listing D.3: Estructura de documento Pedido (parte 2)

```
{
  // Informacion de pago
  pago: {
    seCobra: boolean,
    metodoPago: string,     // 'efectivo' | 'transferencia' | 'yape' | 'plin'
    monto: number,
    comision: number,
    montoTotal: number,
    estadoPago: string,     // 'pendiente' | 'pagado' | 'por_cobrar'
    billeteraUsada: {
      id: string,
      propietarioTipo: string,  // 'proveedor' | 'empresa'
      metodo: string,
      titular: string
    }
  },

  // Informacion del motorizado asignado (desnormalizado)
  motorizado: {
    uid: string,
    nombre: string,
```

```
    asignadoEn: timestamp,
    aceptadoEn: timestamp | null
  } | null,

  // Fechas del ciclo de vida del pedido
  fechas: {
    creacion: timestamp,
    entregaProgramada: timestamp,
    recojo: timestamp | null,
    entrega: timestamp | null,
    anulacion: timestamp | null
  },

  // Control de version para actualizaciones concurrentes
  actualizadoEn: timestamp,
  version: number
}
```

Listing D.4: Estructura de documento Pedido (parte 3)

D.3 Colección: BILLETERAS

Ruta: billeteras/{billeteraId}

```
{
  id: string,
  propietario: {
    tipo: string,           // 'proveedor' | 'empresa'
    uid: string | null,     // Si es proveedor, referencia a
   usuario
    nombreEmpresa: string
  },
  alias: string,            // Nombre descriptivo
  metodo: string,           // 'yape' | 'plin' | '
   transferencia_bancaria'
  cuenta: {
    titular: string,
    telefono: string,       // Para Yape/Plin
    numeroDocumento: string | null,  // DNI/RUC
    tipoDocumento: string | null     // 'DNI' | 'RUC'
  },
  qr: {
```

```
    urlImagen: string,       // URL de Firebase Storage
    urlImagenThumbnail: string | null,
    hash: string | null      // Para validacion
  },
  config: {
    activo: boolean,
    porDefecto: boolean,
    soloEfectivo: boolean,
    limiteMaximo: number | null,
    comisionPorcentaje: number,
    prioridad: number        // Para ordenar opciones
  },
  verificado: boolean,
  verificadoPor: {
    uid: string,
    nombre: string,
    fecha: timestamp
  },
  creadoEn: timestamp,
  actualizadoEn: timestamp,
  ultimoUso: timestamp | null,
  totalTransacciones: number
}
```

Listing D.5: Estructura de documento Billetera

D.3.1 Índices compuestos necesarios

- `proveedor.uid + fechas.creacion` (pedidos por cliente)
- `motorizado.uid + fechas.creacion` (pedidos por motorizado)
- `asignacion.recojo.estado + cicloOperativo.diaCreacion` (pedidos pendientes del día)
- `indices.distritoEntrega + fechas.entregaProgramada` (optimización de rutas)

Apéndice E. Configuración de seguridad y autenticación

E.1 Estructura de custom claims

```
{
  rol: 'administrador' | 'cliente' | 'motorizado',
  permisos: ['crear_pedido', 'asignar_motorizado', '
   ver_dashboard'],
  empresaId: string | null  // Para clientes empresariales
}
```

Listing E.1: Custom claims en Firebase Authentication

E.2 Reglas de seguridad de Firestore

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    // Funcion helper para verificar autenticacion
    function isAuthenticated() {
      return request.auth != null;
    }

    // Funcion helper para verificar rol
    function hasRole(role) {
      return isAuthenticated() &&
             request.auth.token.rol == role;
    }

    // Usuarios: pueden leer/actualizar su propio perfil
    match /usuarios/{userId} {
      allow read: if isAuthenticated() &&
                    (request.auth.uid == userId ||
                     hasRole('administrador'));
      allow write: if isAuthenticated() &&
                     request.auth.uid == userId;
      allow create: if hasRole('administrador');
    }

    // Pedidos: acceso segun rol
    match /pedidos/{pedidoId} {
```

```
      allow read: if isAuthenticated() && (
        hasRole('administrador') ||
        (hasRole('cliente') &&
         resource.data.proveedor.uid == request.auth.uid) ||
        (hasRole('motorizado') &&
         resource.data.motorizado.uid == request.auth.uid)
      );

      allow create: if isAuthenticated() &&
                       (hasRole('administrador') ||
                        hasRole('cliente'));

      allow update: if isAuthenticated() && (
        hasRole('administrador') ||
        (hasRole('motorizado') &&
         resource.data.motorizado.uid == request.auth.uid)
      );

      allow delete: if hasRole('administrador');
    }

    // Billeteras: solo administradores y propietarios
    match /billeteras/{billeteraId} {
      allow read: if isAuthenticated();
      allow write: if hasRole('administrador') ||
                      resource.data.propietario.uid == request.auth.
    uid;
    }
  }
}
```

Listing E.2: Firestore Security Rules

E.3 Reglas de seguridad de Storage

```
rules_version = '2';
service firebase.storage {
  match /b/{bucket}/o {
    // Helper functions
    function isAuthenticated() {
      return request.auth != null;
    }

    function hasRole(role) {
      return isAuthenticated() &&
             request.auth.token.rol == role;
```

```
    }

    // Imagenes de pedidos
    match /pedidos/{pedidoId}/{tipo}/{filename} {
      allow read: if isAuthenticated();
      allow write: if isAuthenticated() && (
        hasRole('administrador') ||
        hasRole('cliente') ||
        hasRole('motorizado')
      );
    }

    // Fotos de perfil
    match /usuarios/{userId}/{filename} {
      allow read: if isAuthenticated();
      allow write: if isAuthenticated() &&
                      request.auth.uid == userId;
    }

    // QR de billeteras
    match /billeteras/{billeteraId}/{filename} {
      allow read: if isAuthenticated();
      allow write: if hasRole('administrador');
    }
  }
}
```

Listing E.3: Firebase Storage Security Rules

Apéndice F. Justificación de puntuaciones en matriz multicriterio

F.1 Escalabilidad

- **Firebase (5)**: Escalamiento automático sin configuración, maneja desde 0 hasta millones de usuarios transparentemente.
- **AWS (4)**: Excelente escalabilidad pero requiere configuración de Auto Scaling y gestión de capacidad.
- **Backend propio (2)**: Requiere planificación manual y redimensionamiento de servidores.

F.2 Costo

- **Firebase (5)**: Capa gratuita generosa, ideal para desarrollo de tesis y etapas iniciales de empresas pequeñas.
- **AWS (3)**: Capa gratuita limitada a 12 meses, costos variables pero competitivos.
- **Backend propio (3)**: Costos fijos predecibles pero presentes desde el inicio.

F.3 Facilidad de integración

- **Firebase (5)**: Componentes nativamente integrados, SDKs optimizados, despliegue con un comando.
- **AWS (3)**: Requiere integrar múltiples servicios manualmente, configuración compleja.
- **Backend propio (2)**: Integración completamente manual de cada componente.

F.4 Mantenimiento

- **Firebase (5)**: Gestión de infraestructura completamente delegada a Google.
- **AWS (3)**: Gestión parcial, requiere actualización de componentes y monitoreo.

- **Backend propio (2)**: Responsabilidad total del mantenimiento, actualizaciones de seguridad, backups.

F.5 Rendimiento

- **Firebase (4)**: Excelente para operaciones en tiempo real, latencia baja en sincronización.
- **AWS (4)**: Rendimiento configurable según recursos asignados, muy bueno con optimización.
- **Backend propio (3)**: Depende del proveedor de hosting y configuración manual.

F.6 Ecosistema y soporte

- **Firebase (5)**: Documentación excelente, amplia comunidad, tutoriales especializados.
- **AWS (5)**: Documentación exhaustiva, soporte empresarial, ecosistema maduro.
- **Backend propio (3)**: Depende de múltiples fuentes, documentación fragmentada.