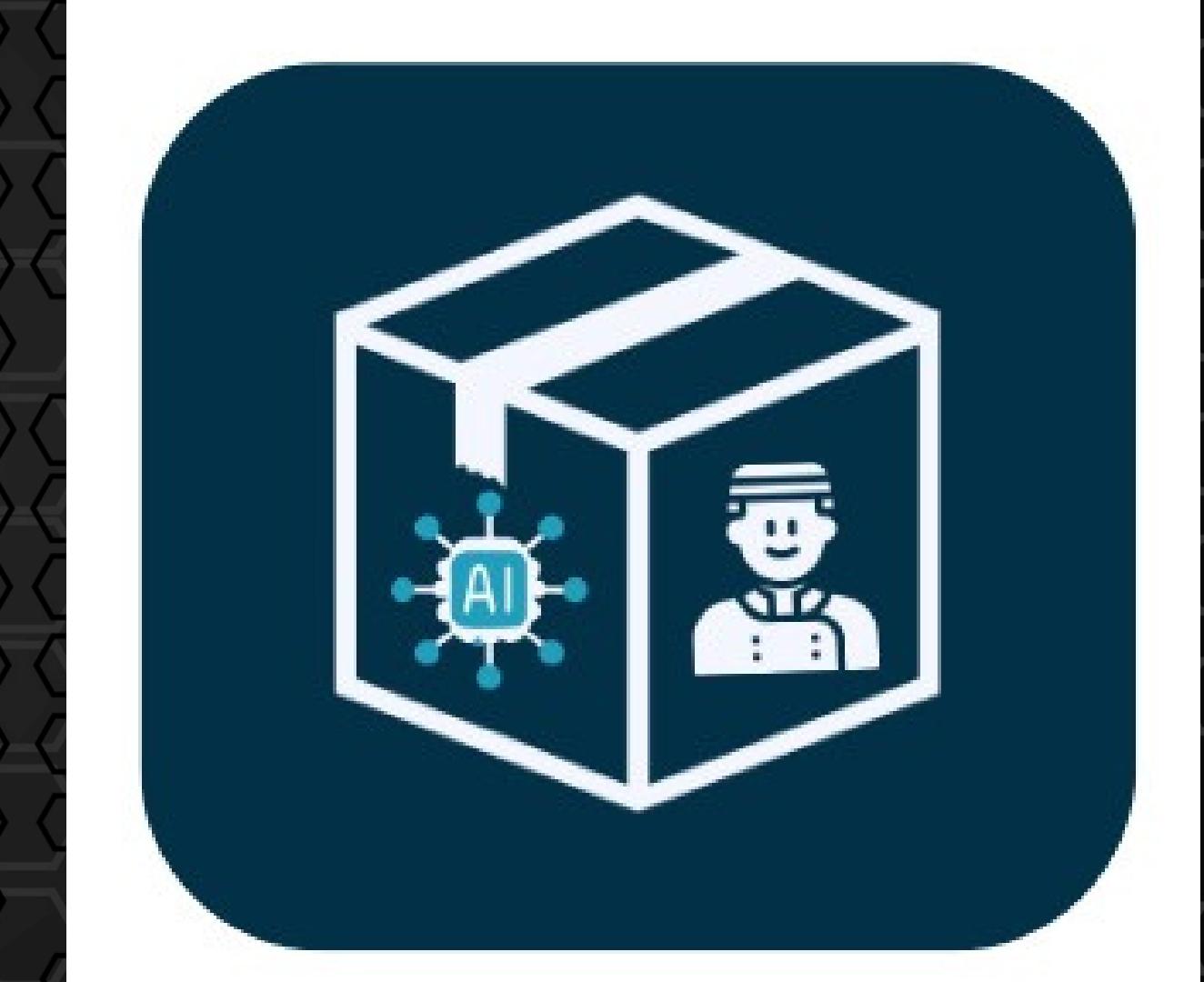
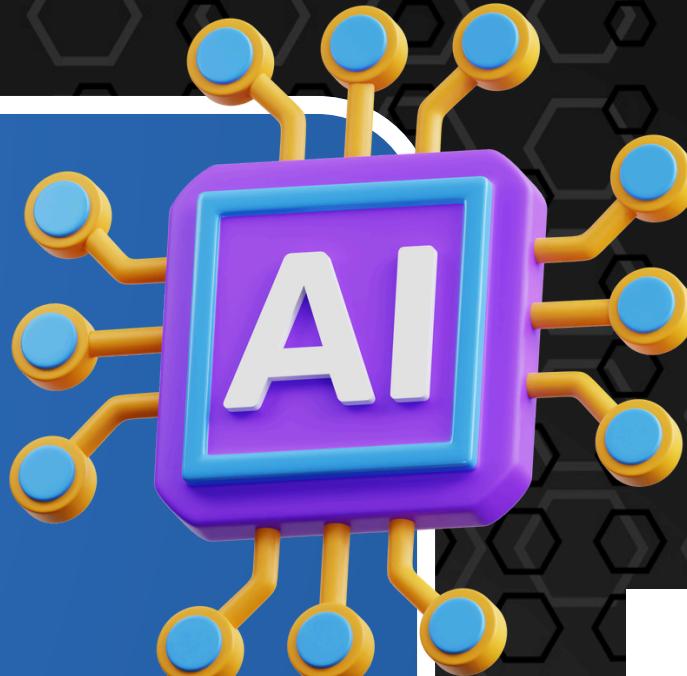




Capstone

# “LlegOKApp” Proyecto

23 noviembre, 2024





# Agenda

- Descripción Proyecto
- Objetivos Específicos
- Alcances y Limitaciones
- Competencias de carrera
- Metodología
- Cronograma
- Arquitectura
- Tecnologías
- Modelo de datos
- Demo
- Resultados
- Obstáculos
- Preguntas

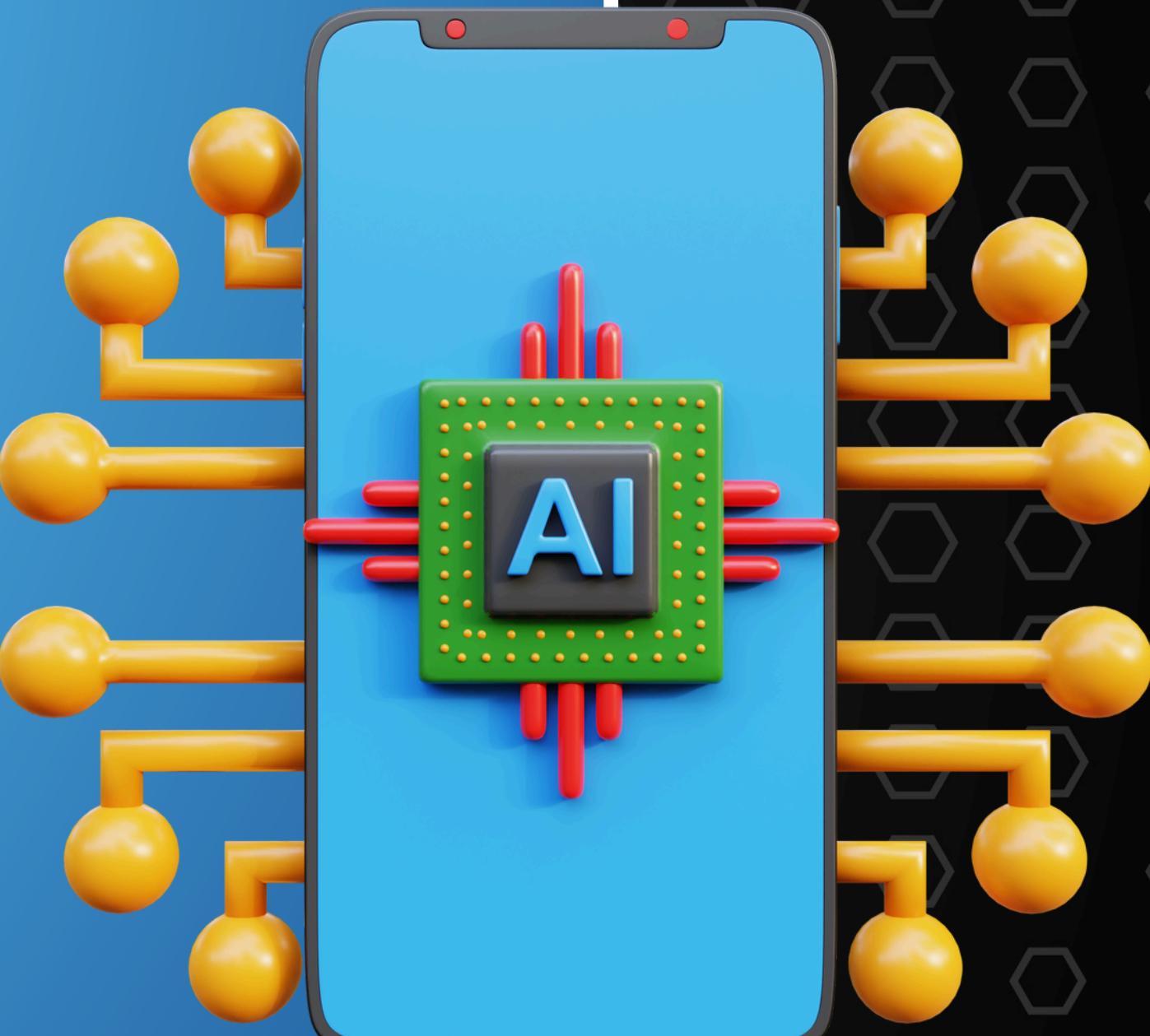


# Descripción Proyecto

El proyecto "jAlnitor" busca desarrollar un sistema automatizado de recepción, gestión de encomiendas y servicios para condominios. Este sistema está diseñado para optimizar la administración de paquetes mediante el uso de tecnologías avanzadas, como inteligencia artificial y microservicios.

## Características Principales

- Reconocimiento de imágenes
- Automatización de notificaciones:
- Gestión de validación



# Objetivos específicos

## Objetivo 01

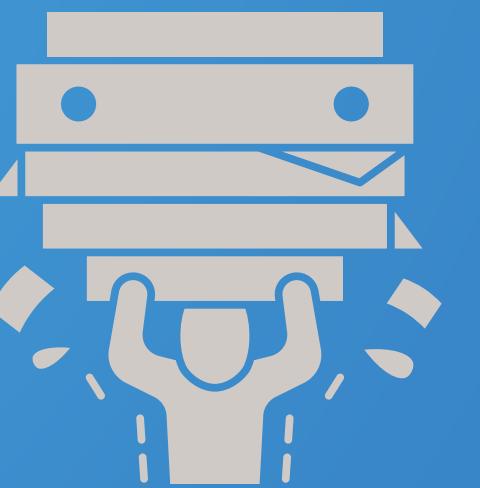
Mejorar la trazabilidad y registro de encomiendas



AI

## Objetivo 02

Bajar la carga operativa de recepción



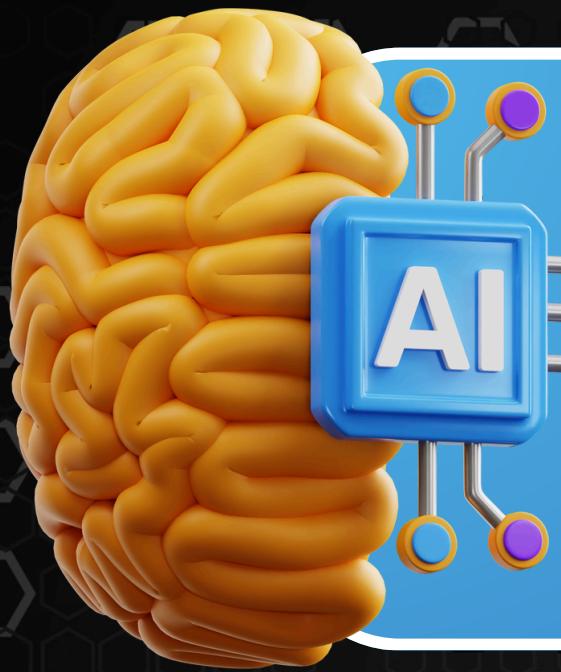
## Objetivo 03

Disminuir errores manuales en la entrega, notificación y recepción de encomienda



AI





# Alcances del Proyecto

## 01 GESTIÓN DE ENCOMIENDAS AUTOMATIZADA

- Fotografía de Paquetes
- Reconocimiento de Texto
- Validación Manual

## 04 VALIDACIÓN Y SEGURIDAD

- Códigos QR Únicos
- Validación de Código

## 02 NOTIFICACIONES AUTOMÁTICAS

- Integración con WhatsApp
- Fotografía del paquete.
- Código QR único para la recogida del paquete.
- Confirmación de Entrega

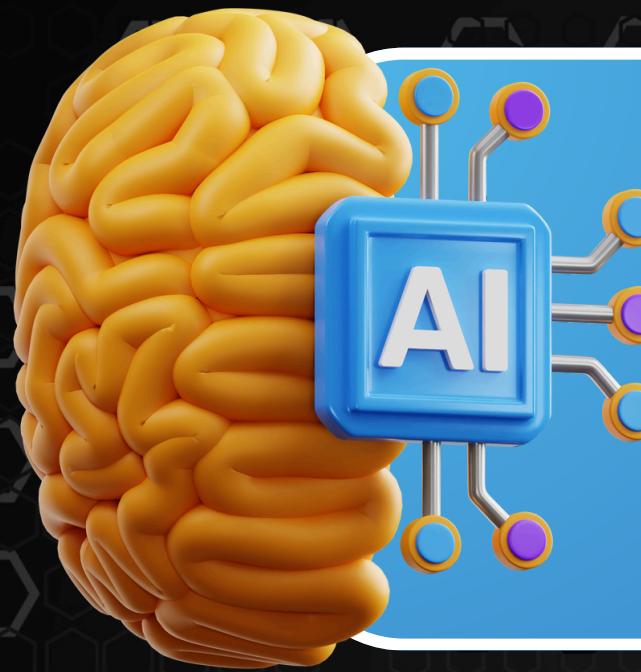
## 05 BASE DE DATOS Y REGISTRO

- Base de Datos MySQL en GCP:
- Almacenamiento de información de residentes, paquetes y transacciones.
- Gestión de Usuarios

## 03 INFRAESTRUCTURA TÉCNICA

- Backend:
  - Implementado con Django y Python.
  - Despliegue en Kubernetes en Google Cloud Platform.
- Frontend: Aplicación móvil exclusivamente para los conserjes.
- CI/CD:
  - Uso de GitHub Actions para la integración y despliegue continuo.





# Limitaciones del proyecto

## 01 RECONOCIMIENTO DE TEXTO

Dependencia de la API de Google Vision

No es un entrenamiento propio

## 02 DEPENDENCIA TECNOLOGICA

Dependencia Tecnológica  
GCP y APIs de Terceros

## 03 EXCLUSIVIDAD DE LA APLICACIÓN MÓVIL

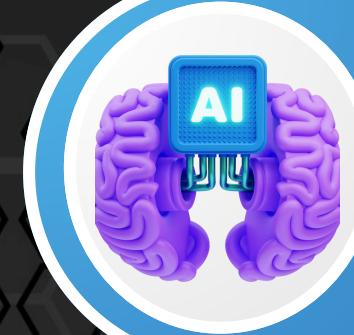
Uso Exclusivo para Conserjes



# Competencias de carrera



**Integración de Servicios Cloud**



**Aplicación de Inteligencia Artificial**



**Desarrollo de soluciones de software**



**Modelado y Gestión de Bases de Datos**



**Gestión de proyectos informáticos**



**Comunicación Efectiva y Trabajo en Equipo**

# Metodología de trabajo



## Metodología de Trabajo

- **Marco Ágil:** Scrum.
- **Duración de Sprints:** 2 semanas por sprint.
- **Total de Sprints:** 5 (10 semanas en total).
- **Definición de Punto:**
  - 1 punto equivale a 1 horas de trabajo.
  - Incluye análisis, desarrollo, pruebas y revisión.



- **Promedio por Sprint:** 56 puntos.
- **Sprints Totales:** 5 sprints.
- **Esfuerzo Total:** 300 horas (10 semanas).
- **Factores Clave:**
  - Resolución de bloqueos.
  - Ciclos de mejora identificados en retrospectivas.
  - Revisión de entregables y validaciones finales.

Proyectos / Captone\_ASR / SCRUM-6 / SCRUM-60

HDU Recibir encomienda por conserje

Proyectos / Captone\_ASR / SCRUM-6 / SCRUM-31

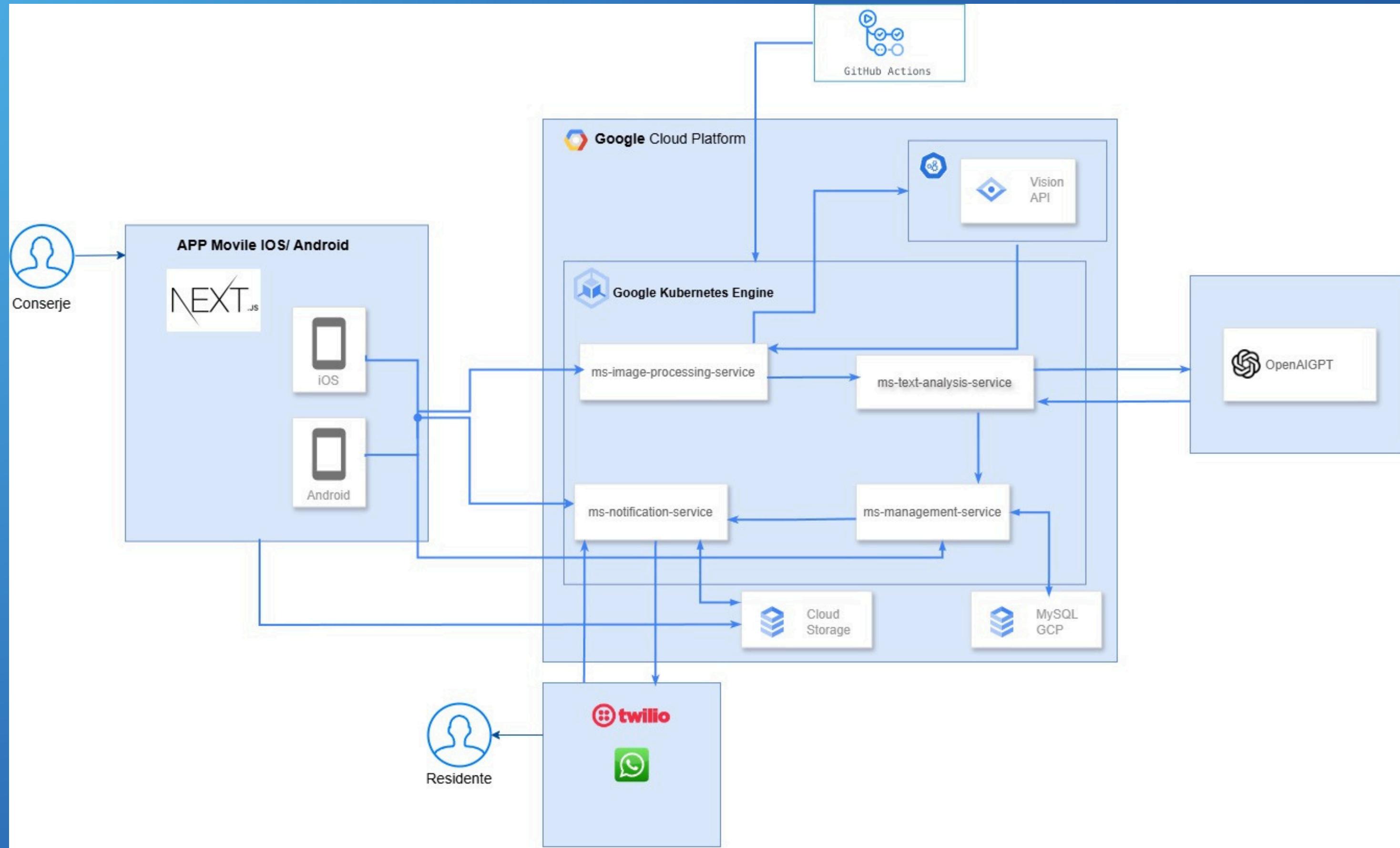
HDU Notificación automática a residente

# Cronograma

Actividad	Fase 1				Fase 2												Fase 3	
	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18
Planificación del Proyecto	X	X	X	X														
Diseño y Configuración Inicial					X	X												
Implementación del Reconocimiento de Imágenes					X	X												
Automatización de Notificaciones							X	X										
Desarrollo del Frontend Funcional									X	X								
Pruebas y Optimización Final											X	X	X	X	X			
Presentación del Proyecto													X		X		X	

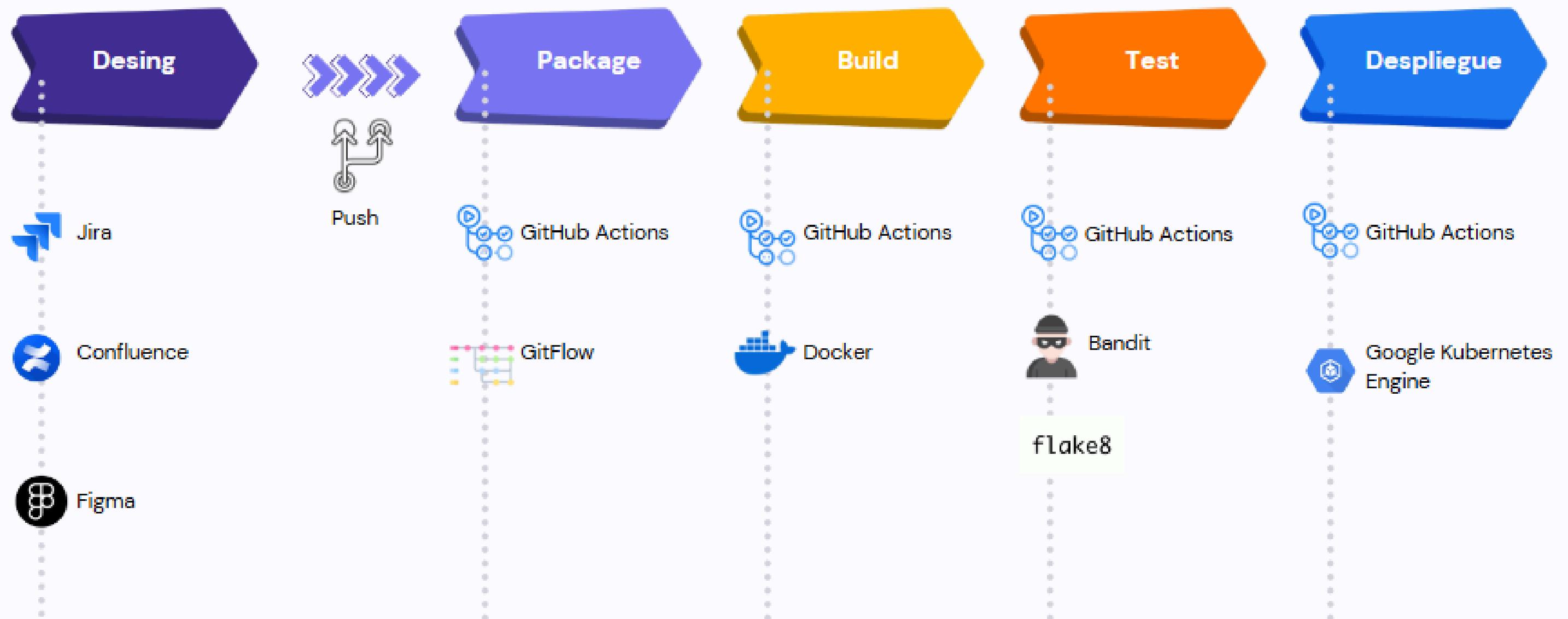
YOU ARE  
HERE

# Arquitectura

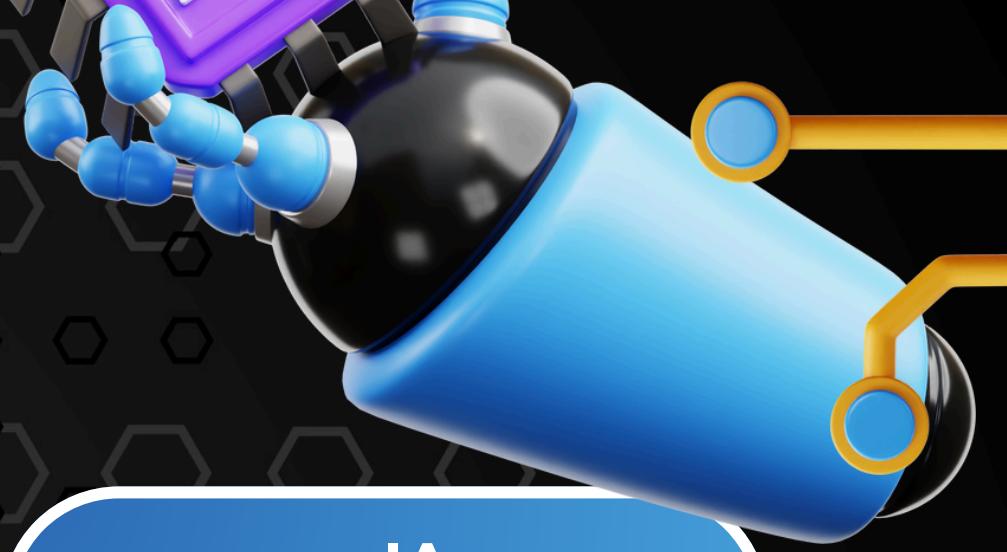
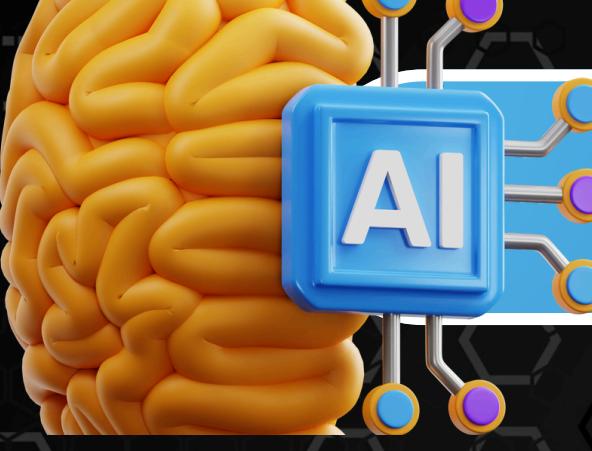


# GitHub Actions

## Proceso CI/CD



# Tecnologías Utilizadas



## Desarrollo Frontend



NextJS

## Desarrollo Backend



Python

Django



RestApi

## GCP



Google  
Kubernetes  
Engine

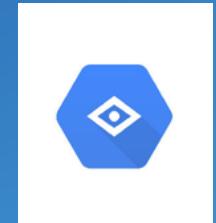


Google  
Cloud  
Storage



MySQL

## IA



Google  
Vision Api

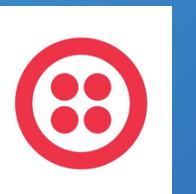


OpenAI

## Notificaciones



WhatsApp



Twillio Api

## Control de Versiones y CI/CD



GitHub Actions



GitHub

## Análisis y Calidad de Código



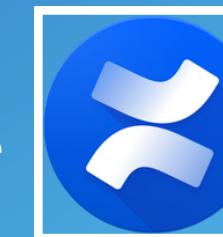
Bandit

flake8

## Metodología Ágil

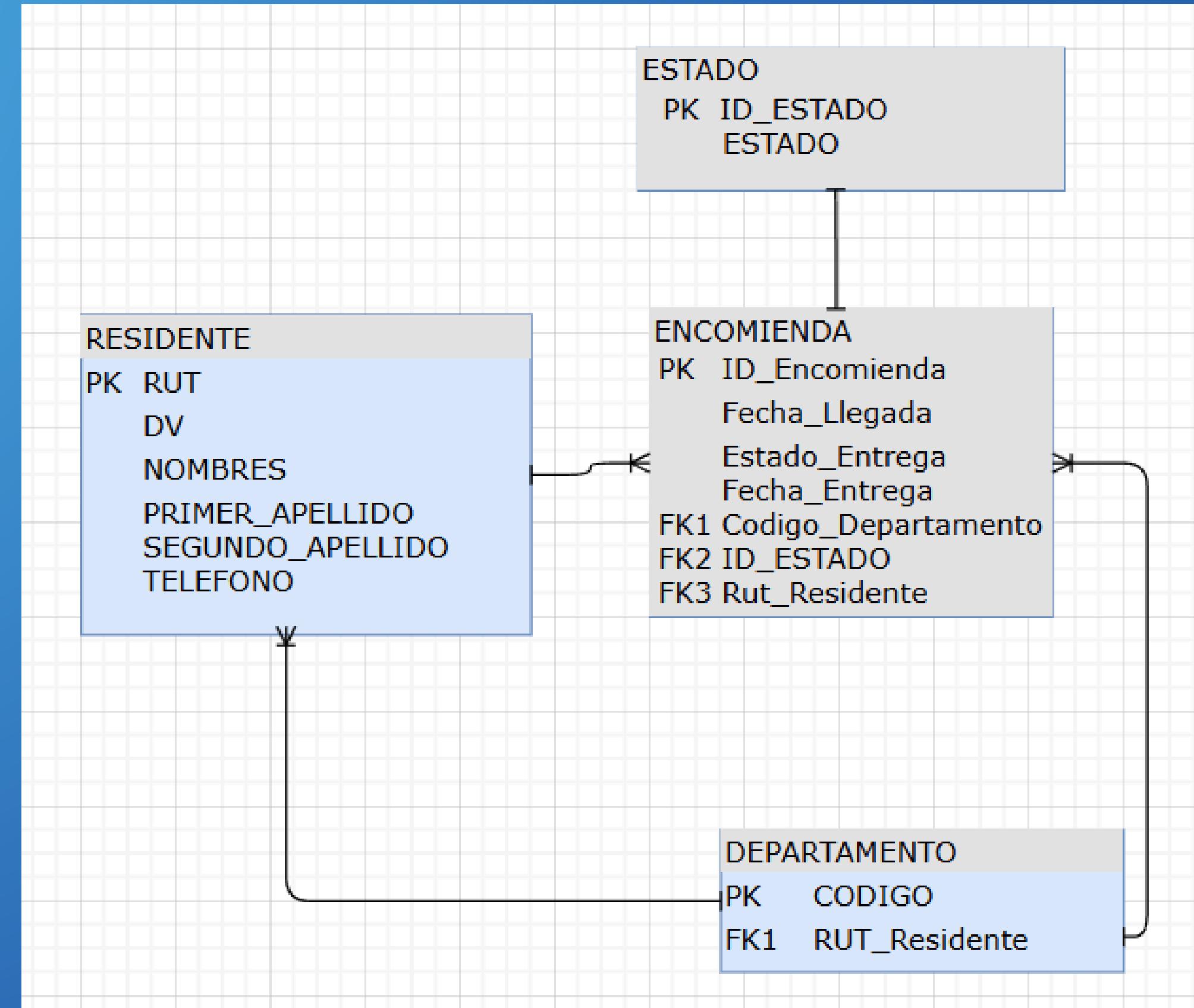


Jira



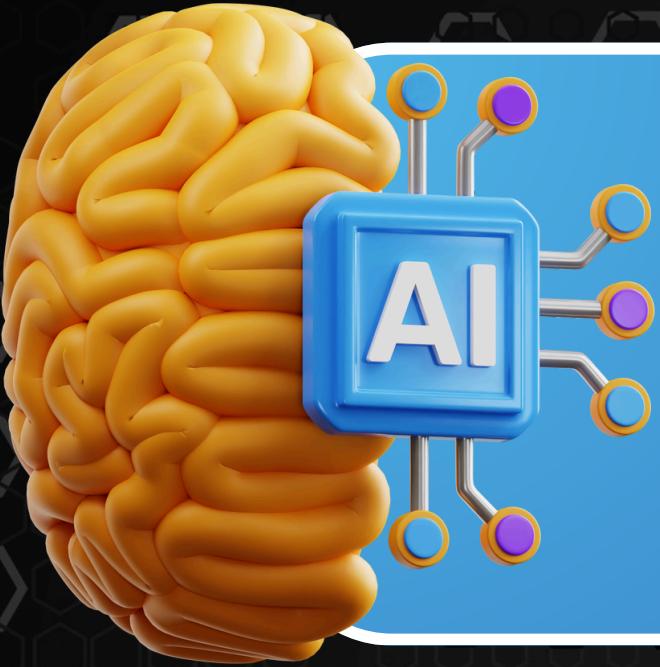
Confluence

# Modelo de Datos



# Demostración





# Resultados Obtenidos

## 01 FUNCIONALIDADES IMPLEMENTADAS

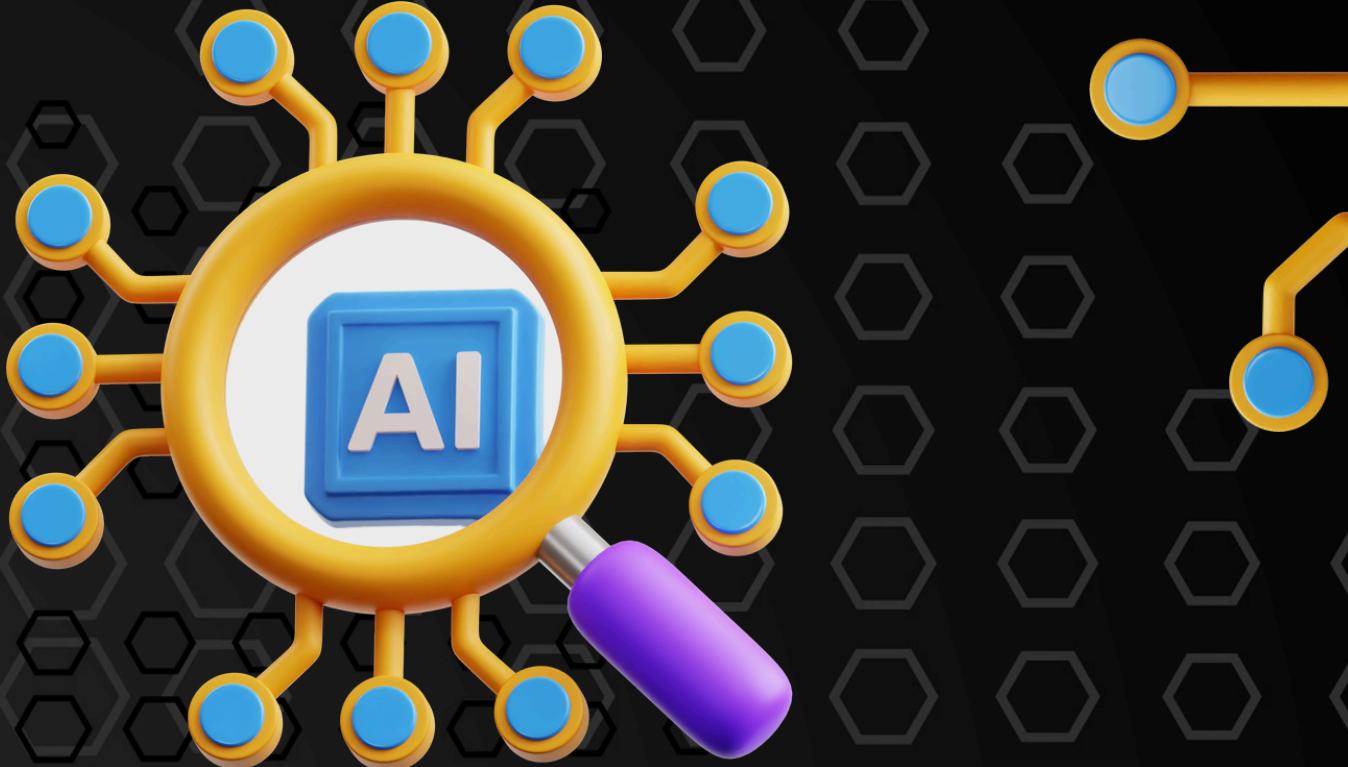
- Reconocimiento de texto mediante imágenes
- Automatización de notificaciones:
- Gestión de estados de encomiendas:
- Aplicación móvil funcional

## 04 APRENDIZAJES DEL EQUIPO

- Gestión de cambios:
- Resolución de bloqueos:
- Optimización de herramientas:
  - Implementar tecnologías como Google Vision API y Twilio

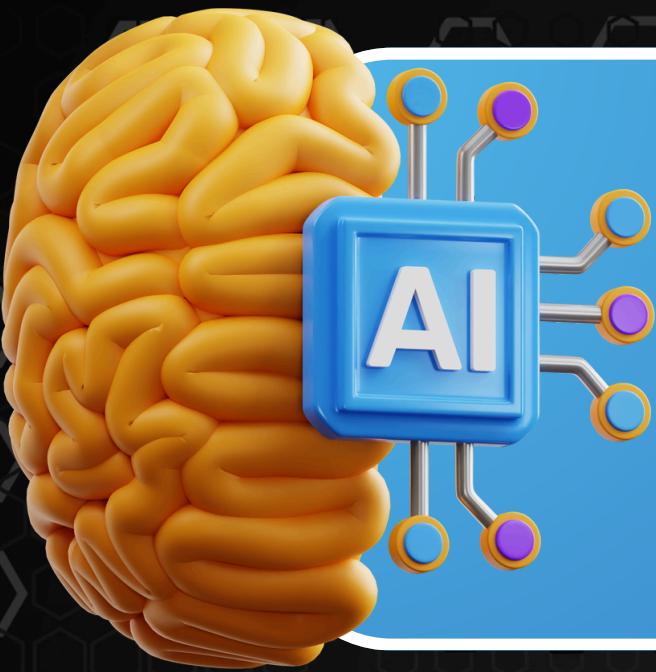
## 02 TECNOLOGÍA INTEGRADA

- Back-end eficiente
- Base de datos robusta
- Pipeline de CI/CD



## 03 IMPACTO EN LA GESTIÓN

- Reducción de tiempo manual
- Mejora en la comunicación hacia residentes
- Trazabilidad completa



# Obstáculos presentados

## 01 DESBALANCE EN EL ENFOQUE DE DESARROLLO

Se dedicó más tiempo y recursos al desarrollo del backend (microservicios, integraciones con APIs externas, configuración de la base de datos) que al frontend.

## 04 DEPENDENCIAS TECNOLÓGICAS

La utilización de OpenAI requirió mayor configuración y depurar código para la correcta extracción del dato requerido para la funcionalidad del sistema.

## 02 CAMBIOS EN LOS REQUERIMIENTOS

Ajustes constantes en las funcionalidades backend, como implementación de CI/CD.

## 03 PROBLEMAS CON LA BASE DE DATOS

Debido a costos monetarios se tuvo que migrar y utilizar otro proyecto en GCP.  
Impacto en tiempo.





# Preguntas de la Comisión

