Diagrama de alto nivel sobre componentes y arquitectura

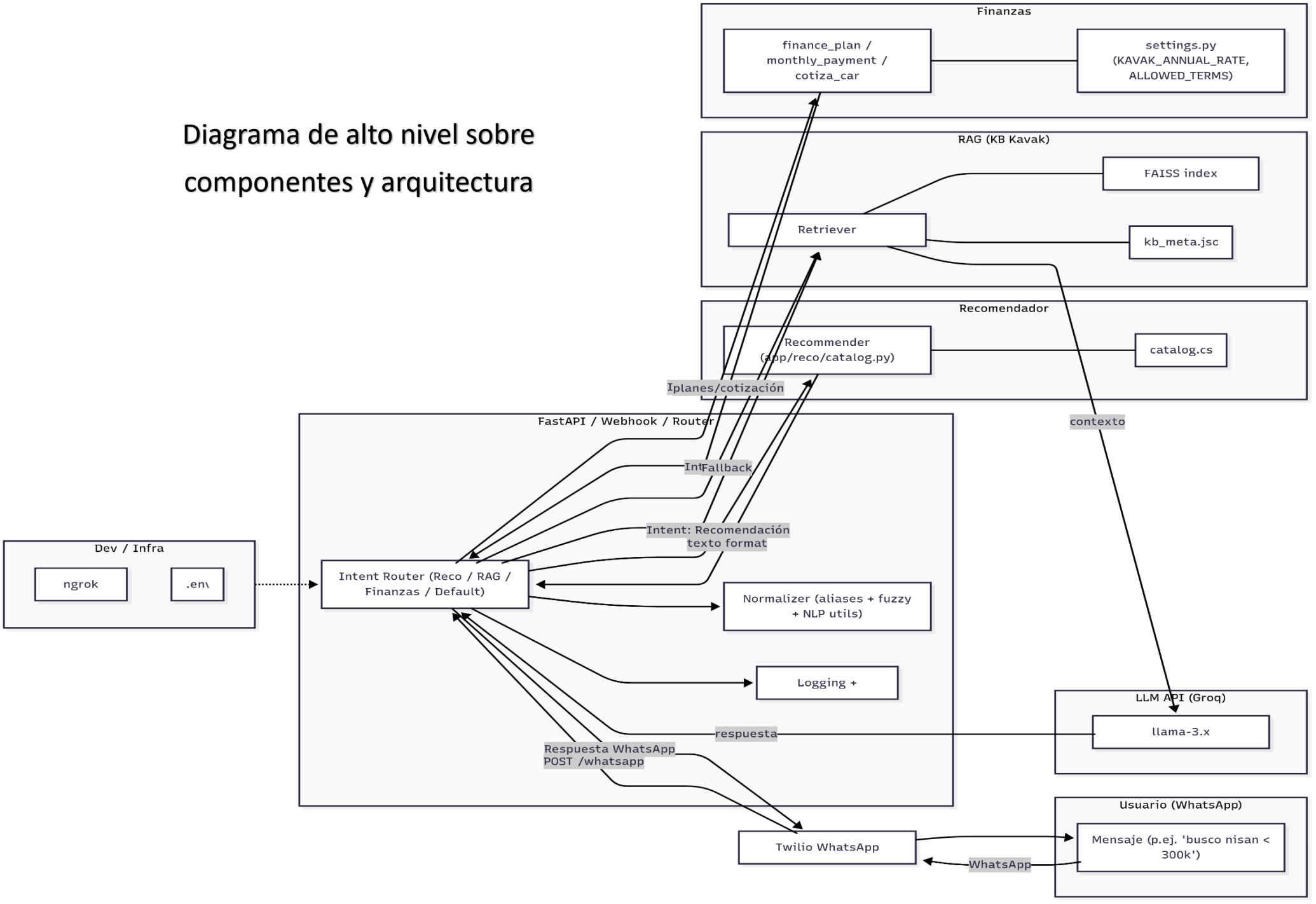


Diagrama sobre prompts, arquitectura de agentes y tools

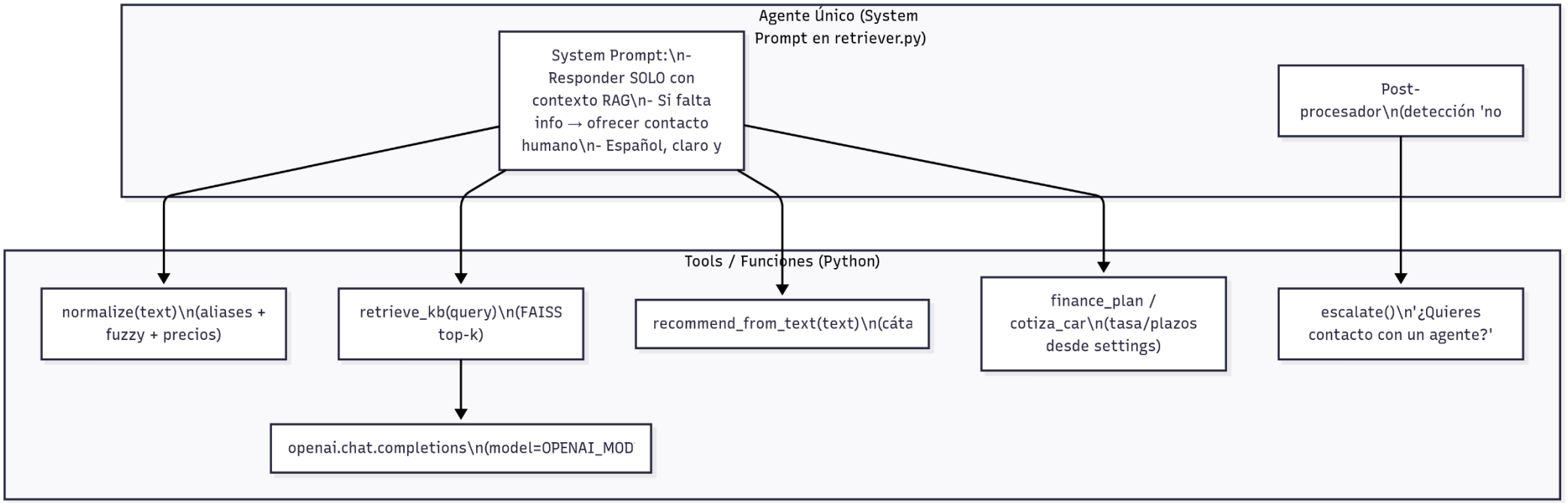
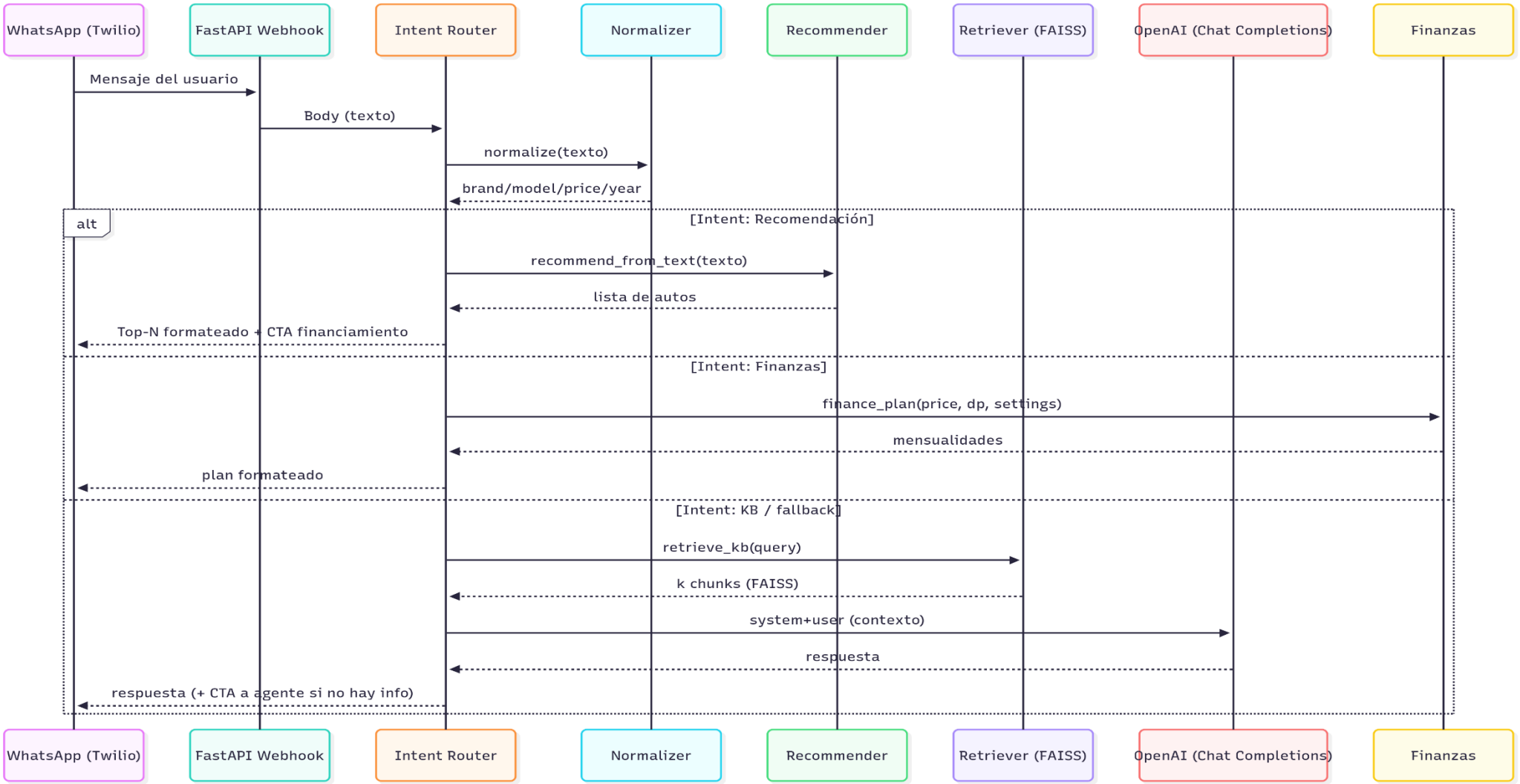


Diagrama sobre prompts, arquitectura de agentes y tools – Secuencia de iteración



Resumen ejecutivo

Este proyecto desarrolla un asistente conversacional de Kavak, integrado a WhatsApp mediante Twilio, diseñado para mejorar la experiencia de clientes interesados en autos seminuevos. El bot combina un motor de **RAG** (búsqueda y generación aumentada con conocimiento) con un **recomendador de catálogo** y un módulo de **finanzas**, lo que le permite responder consultas sobre la propuesta de valor de la empresa, entender preguntas con errores de redacción, recomendar vehículos de acuerdo con preferencias del usuario y ofrecer planes de financiamiento claros. Se han implementado mecanismos para reducir alucinaciones y asegurar que, en caso de falta de información, el bot derive la conversación a un agente humano. Con estas capacidades, el asistente busca reducir tiempos de respuesta, mejorar la calidad de la atención y aumentar la conversión de prospectos. El siguiente paso es llevarlo a producción siguiendo un roadmap que contempla despliegue en contenedores, integración de métricas de desempeño y pruebas de regresión automatizadas para garantizar su confiabilidad.

📌 Propuesta de Roadmap y Backlog – Asistente Kavak

1. Roadmap hacia Producción

Fase 1 – Consolidación técnica

- Refinar alias y normalización de texto (NLP robusto).
- Definir dataset de **preguntas frecuentes** y asegurar cobertura en la KB (FAISS).
- Congelar versión inicial del catálogo y pipelines de ingestión.

Fase 2 – Infraestructura y despliegue

- Contenerizar el bot con **Docker**.
- Orquestación en **Kubernetes / ECS / AKS** (según plataforma).
- Uso de **CI/CD (Jenkins/GitHub Actions)** para despliegue automatizado.
- Exponer API con **FastAPI** detrás de un **API Gateway / Ingress**.
- Configurar **Twilio WhatsApp sandbox → producción**.

Fase 3 – Observabilidad y escalabilidad

- Integrar **monitoring**: logs estructurados, métricas (Prometheus/Grafana), trazas (OpenTelemetry).
- Dashboards de **uso y errores**: consultas contestadas, fallback a agente humano, tiempos de respuesta.
- Configurar **autoscaling** según tráfico (HPA en K8s o auto scaling groups).

Fase 4 – Seguridad y cumplimiento

- Manejo seguro de claves (Vault/Secret Manager).
- Cifrado de datos en tránsito y en reposo.
- Políticas de retención de datos de usuarios.

---

2. Evaluación del desempeño del agente

Métricas técnicas

- **Tiempo medio de respuesta (latencia)** < 2s.
- **Disponibilidad** > 99.9%.

- **Uso de fallback** (% de respuestas “contactar agente” vs respuestas útiles).

Métricas de calidad de respuesta

- **Exactitud factual (Ground Truth QA)**: dataset de ~200 preguntas frecuentes con respuestas “golden” → comparar outputs del bot.
- **Tasa de alucinación**: # de respuestas incorrectas o inventadas ÷ total.
- **Cobertura de intenciones**: % de queries mapeadas correctamente a Recomendación / KB / Finanzas.

Métricas de negocio

- % de conversaciones resueltas sin escalar a un agente humano.
- Conversión a **cotización completada**.
- NPS / satisfacción del usuario (encuesta post-chat).

---

3. Estrategia de pruebas y regresión

Pruebas unitarias y de integración

- **Pytest** para cada módulo:
  - Normalizer (alias + fuzzy).
  - Recommender (catálogo).
  - Retriever (RAG).
  - Finanzas (cálculo de mensualidades).
- **Tests de API (FastAPI)** con TestClient para simular eventos de Twilio.

Conjunto de pruebas de regresión

- Dataset de preguntas frecuentes (ej. “propuesta de valor”, “busco nisan barato”, “cotízame un Sentra 2018 con 50k enganche”).
- Cada release debe pasar este **suite de regresión automatizada**.

Pruebas de desempeño y carga

- **Locust / k6** para simular miles de mensajes concurrentes.
- Verificar escalado automático y que no degrade latencia.

Canary / A/B Testing

- Desplegar la nueva versión a un % pequeño de usuarios.
- Comparar métricas de calidad (respuesta útil, fallback) vs versión anterior.
- Si pasa el umbral → rollout al 100%.

---

4. Backlog Propuesto

- Automatizar ingestión y re-entrenamiento del índice FAISS.
- Ampliar alias de marcas/modelos con logs de usuarios reales.
- Integrar feedback loop: marcar respuestas útiles vs no útiles.
- Añadir dashboard en Grafana con métricas de latencia, fallbacks y satisfacción.
- Construir regresión automatizada con dataset de QA golden.
- Implementar pruebas de carga con k6.
- Desplegar pipeline de CI/CD con validación previa de tests.