INSTITUTO TECNOLÓGICO DE BUENOS AIRES

PROYECTO: RAG FOR JAVA UNIT TESTS

Informe Técnico

Generación Automática de Tests Unitarios en Java

Grupo 4

Autores:
José Burgos (61525)
Felipe Hiba (61219)
Patricio Escudeiro (61156)

Índice

Contexto

Este sistema RAG (Retrieval-Augmented Generation) está diseñado para generar automáticamente métodos de prueba JUnit a partir de clases Java proporcionadas por el usuario. El flujo convierte el código en embeddings, recupera ejemplos relevantes de una base vectorial y redacta el método de test con un modelo de lenguaje. El resultado se entrega como un archivo `.java` listo para su integración.

1. Consideraciones de Responsible AI y Safety

Consideración 1: Privacidad y anonimización. Antes de construir cualquier prompt, se anonimizará el código eliminando rutas de sistema y metadatos sensibles mediante reglas de filtrado y expresiones regulares. Así se evita la exposición de información personal o credencial en las solicitudes al modelo.

Consideración 2: Robustez frente a entradas maliciosas. Se impondrán límites estrictos a la longitud de los prompts y se escaparán caracteres especiales críticos para prevenir ataques de prompt injection. Además, se establecerán restricciones de tiempo de ejecución y tamaño de los mensajes con el LLM, garantizando la estabilidad del servicio ante entradas no deseadas.

Consideración 3: Trazabilidad y auditoría. Cada petición se registrará en formato JSON con un identificador único, versión de modelo, latencia y fragmentos empleados. Estos logs se almacenarán en un repositorio centralizado, facilitando auditorías.

Consideración 4: Gobernanza y mejora continua. Se evaluará la calidad de tests generados, evaluando cobertura y compilación automática.

Consideración 5: Salidas inofensivas y de utilidad. Para asegurar que las pruebas sean harmless y realmente helpful, el sistema incorporará un filtro de contenido que detecte y bloquee cualquier fragmento de código potencialmente dañino. Asimismo, se enriquecerán los tests con comentarios.

2. Estrategias de puesta en producción y precisión de tests

El despliegue se realiza en AWS, aprovechando servicios gestionados para escalabilidad y alta disponibilidad. Primero, las funciones de extracción de datos y de cálculo de embeddings corren en AWS Lambda, consumiendo la base de conocimiento almacenada en Amazon S3. Los embeddings resultantes se indexan en Amazon OpenSearch para búsquedas vectoriales de baja latencia. La generación final de tests se encapsula en otra función Lambda que ensambla el prompt y lo envía a un servidor Ollama desplegado sobre Amazon ECS, devolviendo el método de test al usuario vía API Gateway.

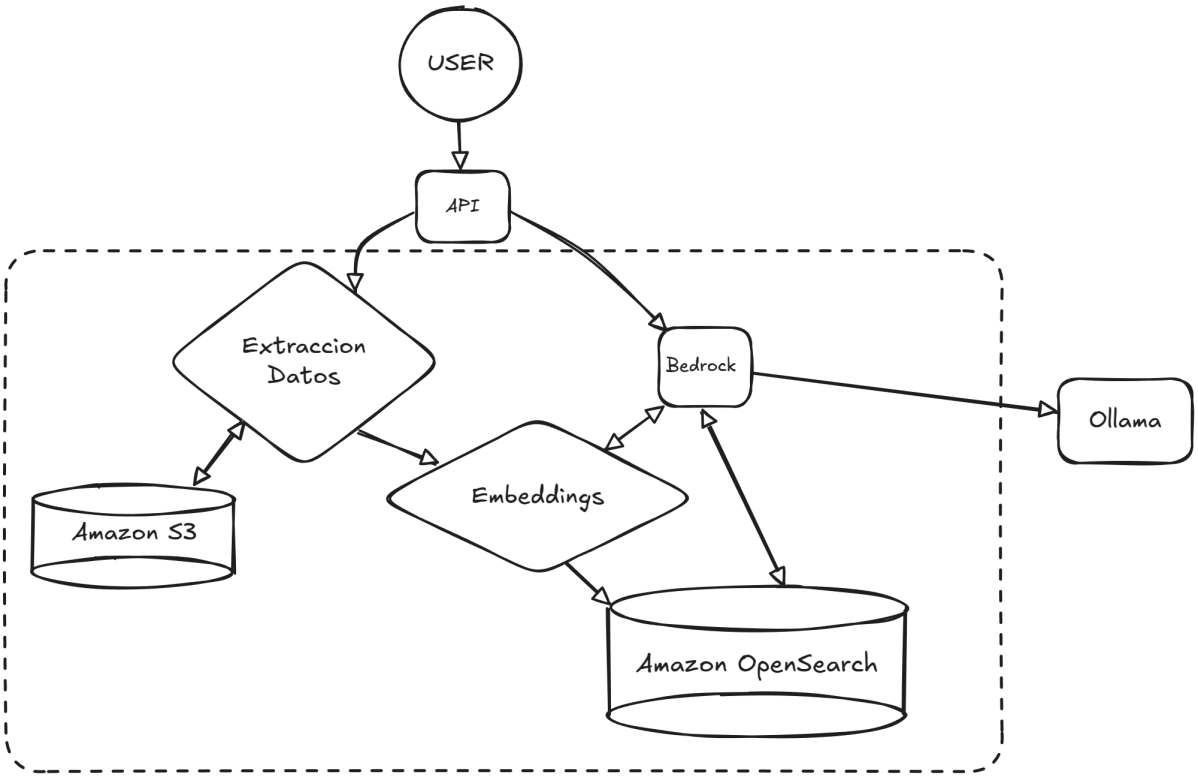

Figura 1: Esquema de despliegue en AWS para el sistema RAG