Desarrollo de Soluciones Cloud

Proyecto – Entrega No. 4

Por:

Luis Castelblanco Quintero
Nicolas Saavedra Gonzales
Valeria Caro Ramirez

Contenido

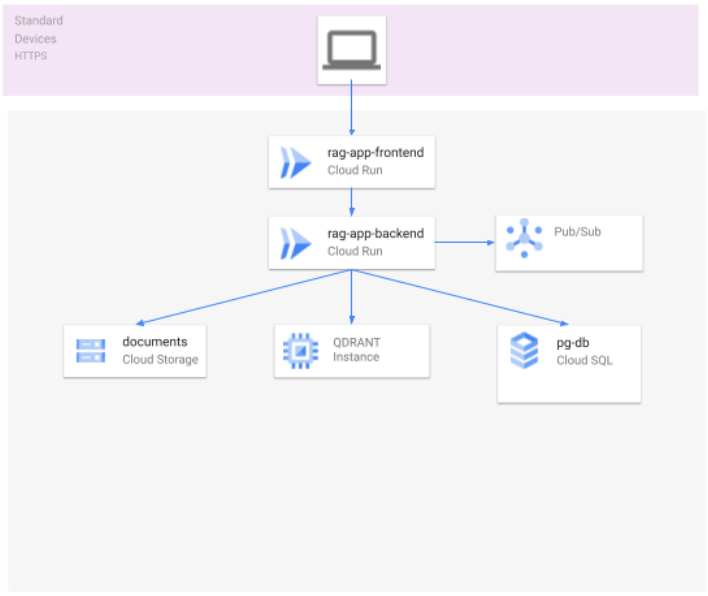
Introducción

Como parte del desarrollo del **Proyecto No. 4**, y sobre la base del despliegue básico en la nube ya realizado, se ha planteado una fase final centrada en la adopción de tecnologías **serverless** en Google Cloud con el objetivo de simplificar la operación de la aplicación. Esta etapa busca reemplazar, a discreción del equipo, uno o más componentes actuales utilizando servicios gestionados como **Cloud Pub/Sub**, **Cloud Run** y **Cloud Functions**.

La iniciativa no tiene como objetivo evaluar la capacidad de escalabilidad automática de estos servicios, sino demostrar su **integración funcional** dentro de la arquitectura existente. Para validar la efectividad de la solución propuesta, se han llevado a cabo pruebas de funcionamiento que permiten evidenciar el comportamiento del sistema.

Esta transición a una arquitectura serverless responde a la necesidad de reducir la complejidad operativa, facilitar el mantenimiento y aprovechar las ventajas del modelo basado en eventos y ejecución bajo demanda que ofrece Google Cloud.

Presentación diagrama de arquitectura



Componentes de la arquitectura

Cloud Run

rag-app-frontend

Este componente representa la capa de presentación del sistema. Está desplegado en **Cloud Run**, lo que permite una ejecución bajo demanda, con escalabilidad automática y sin necesidad de gestionar servidores. Su función principal es interactuar con el usuario y canalizar las solicitudes hacia el backend.

rag-app-backend

Corresponde a la lógica central de la aplicación. También se encuentra desplegado en **Cloud Run**. Este servicio se encarga de:

- Procesar las solicitudes del frontend.
- Interactuar con bases de datos, sistemas de almacenamiento y servicios de mensajería.
- Coordinar la comunicación entre los distintos componentes mediante interfaces REST o eventos.

Cloud Pub/Sub

Servicio de mensajería asincrónica utilizado para **desacoplar procesos** dentro de la arquitectura. El backend puede publicar eventos relevantes en un tema de Pub/Sub, los cuales pueden ser consumidos posteriormente por otros servicios, como funciones Cloud Functions, sin bloquear el procesamiento principal.

Cloud Storage

Este bucket de **Cloud Storage** actúa como repositorio para documentos y archivos no estructurados. Permite almacenar datos persistentes de forma escalable y segura, que luego pueden ser procesados o consultados por el backend o por servicios adicionales que consuman los eventos de Pub/Sub.

Compute Engine – VM Instance

Qdrant es una base de datos especializada en **almacenamiento y búsqueda vectorial**, ideal para soluciones de búsqueda semántica o Recuperación Aumentada por Generación (RAG). El backend utiliza esta instancia para almacenar y consultar representaciones vectoriales (embeddings) de los documentos, facilitando búsquedas más precisas y contextuales.

Cloud SQL

Servicio de base de datos relacional gestionado. utilizado para almacenar información estructurada como usuarios, metadatos de documentos, registros de actividad, entre otros. Proporciona integridad, persistencia y capacidad de consulta mediante SQL estándar.

Justificación de los componentes reemplazados

Web Server Tradicional ➜ Cloud Run (rag-app-frontend)

- **Antes:**
  El frontend de la aplicación estaba desplegado en máquinas virtuales gestionadas manualmente (Compute Engine). Esto implicaba mantener la infraestructura, aplicar actualizaciones del sistema operativo y configurar el escalado de manera explícita.
- **Ahora:**
  El componente fue migrado a **Cloud Run** bajo el nombre rag-app-frontend. Cloud Run permite ejecutar contenedores en un entorno totalmente gestionado, con escalado automático según el tráfico HTTP recibido. También incluye soporte nativo para HTTPS y autenticación, eliminando tareas administrativas.
- **Ventajas:**
  - Eliminación de la gestión de infraestructura.
  - Escalado automático sin configuración adicional.
  - Tiempo de despliegue reducido.

Worker Backend (Compute Engine) ➜ Cloud Run (rag-app-backend)

- **Antes:**
  La lógica de negocio y procesamiento central residía en instancias dedicadas o contenedores orquestados que requerían configuración manual del entorno, puertos, variables de entorno y conexión con otros servicios.
- **Ahora:**
  Este componente se implementa como rag-app-backend en **Cloud Run**, simplificando el despliegue y facilitando la integración con Cloud SQL, Qdrant, Cloud Storage y Pub/Sub. El backend sigue respondiendo a peticiones HTTP, pero lo hace en un entorno sin servidor.
- **Ventajas:**
  - Fácil integración con otros servicios de Google Cloud (IAM, Pub/Sub, SQL).
  - Simplificación de la lógica de despliegue y CI/CD.

Procesamiento Asíncrono Manual ➜ Cloud Pub/Sub + Cloud Functions

- **Antes:**
  El backend procesaba tareas pesadas o asincrónicas de forma directa, como análisis de documentos, generación de embeddings o notificaciones. Esto bloqueaba recursos del backend y podía afectar la experiencia del usuario.
- **Ahora:**
  Se introdujo **Cloud Pub/Sub** como capa de mensajería asincrónica, y algunas tareas específicas fueron delegadas a **Cloud Functions** que se activan automáticamente al recibir mensajes. Esto permite escalar tareas pesadas de forma independiente, sin afectar el rendimiento del backend.
- **Ejemplo de uso:**
  - Cuando se carga un documento, el backend publica un mensaje en un tópico de Pub/Sub.
  - Una función en Cloud Functions se activa, procesa el documento, genera embeddings y actualiza Qdrant.
- **Ventajas:**
  - Aislamiento de responsabilidades (Single Responsibility).
  - Procesamiento distribuido y desacoplado.

Componentes Conservados en la Arquitectura Serverless

Aunque la migración a una arquitectura serverless implicó el reemplazo de varios elementos tradicionales, algunos componentes clave fueron **conservados** debido a su funcionalidad crítica, su compatibilidad con servicios gestionados y su estabilidad en el ecosistema de Google Cloud. A continuación, se describen estos componentes:

1. Cloud Storage – documents

- **Función:**
  Cloud Storage sigue siendo el repositorio principal de documentos no estructurados, como archivos PDF, imágenes y textos enriquecidos, que son utilizados en procesos de recuperación aumentada por generación (RAG).

2. Cloud SQL – pg-db

- **Función:**
  Cloud SQL aloja la base de datos relacional utilizada por la aplicación. Aquí se

almacenan datos estructurados como usuarios, metadatos de documentos, registros de actividad, configuraciones y relaciones del sistema.

3. Qdrant Instance – VM Instance

- **Función:**
  Qdrant actúa como base de datos vectorial, almacenando representaciones semánticas (embeddings) de documentos para permitir búsquedas contextuales avanzadas dentro del flujo RAG.

Pasos de integración

La nueva arquitectura del **Proyecto N4** se basa en servicios serverless de Google Cloud como **Cloud Run**, **Cloud Pub/Sub** y **Cloud Functions**, permitiendo una integración ágil, escalable y de bajo mantenimiento. Se reemplazaron los componentes tradicionales de frontend y backend por versiones totalmente gestionadas, mientras que se conservaron servicios clave como **Cloud Storage**, **Cloud SQL** y **Qdrant** por su relevancia funcional. Esta integración orientada a eventos y desacoplada facilita el procesamiento eficiente de documentos, mejora la recuperación semántica y reduce significativamente la carga operativa del sistema.

Variables de entorno

En cuanto a la gestión de configuraciones sensibles y parámetros del entorno, se utilizó la funcionalidad nativa de **Google Cloud Run** para definir **variables de entorno** y acceder de forma segura a **secretos** mediante **Secret Manager**. Esta estrategia permite mantener la separación entre código y configuración, facilita el manejo de entornos (desarrollo, pruebas, producción) y garantiza un alto nivel de seguridad para credenciales, tokens y claves de acceso.

Además, para la conexión con la base de datos relacional, se empleó el conector nativo de **Cloud SQL para Cloud Run**, lo que permite establecer conexiones seguras, gestionadas y eficientes sin necesidad de exponer direcciones IP públicas o manejar túneles manuales.

Costos aproximados

Componente	Costo Estimado Mensual (USD)
PostgreSQL (Cloud SQL)	114.46
Cloud Storage	99.9
Backend (Cloud Run)	5.19
Frontend (Cloud Run)	4.15
Cloud Pub/Sub	0.8
Worker (VM)	13.23
Qdrant (VM)	25.46
Total	263.19

Como parte del análisis financiero del despliegue de la arquitectura serverless para el **Proyecto No. 4**, se ha consolidado una estimación mensual de costos considerando tanto los servicios gestionados de Google Cloud como las máquinas virtuales utilizadas para funciones específicas. La estructura de costos refleja un uso eficiente de recursos, combinando servicios como **Cloud Run, Cloud SQL, Cloud Pub/Sub y Cloud Storage**, junto con el soporte de **VMs dedicadas** para el procesamiento especializado (Worker) y la base de datos vectorial Qdrant.

El **costo total mensual estimado** asciende a **$263.19 USD**, distribuidos entre procesamiento, almacenamiento, mensajería y servicios de bases de datos, lo cual proporciona una base sólida y escalable para la operación continua del sistema.

Pruebas de Funcionamiento

Para evidenciar el correcto funcionamiento de la aplicación, se presentarán diversas **capturas de pantalla** que ilustran el comportamiento esperado de cada componente en distintos escenarios de uso. Estas imágenes demostrarán la operatividad de la interfaz, la interacción entre servicios y la correcta ejecución de los flujos definidos en la arquitectura serverless del Proyecto No 4.

Despliegue de la aplicación

En las siguientes imágenes se evidencia el correcto funcionamiento de las aplicaciones desplegadas en **Google Cloud Run**. Además, se presentan métricas extraídas desde la consola de Google Cloud que confirman la ejecución estable de los servicios, permitiendo

verificar el procesamiento de solicitudes, el uso de recursos y el comportamiento general de la aplicación en un entorno productivo.

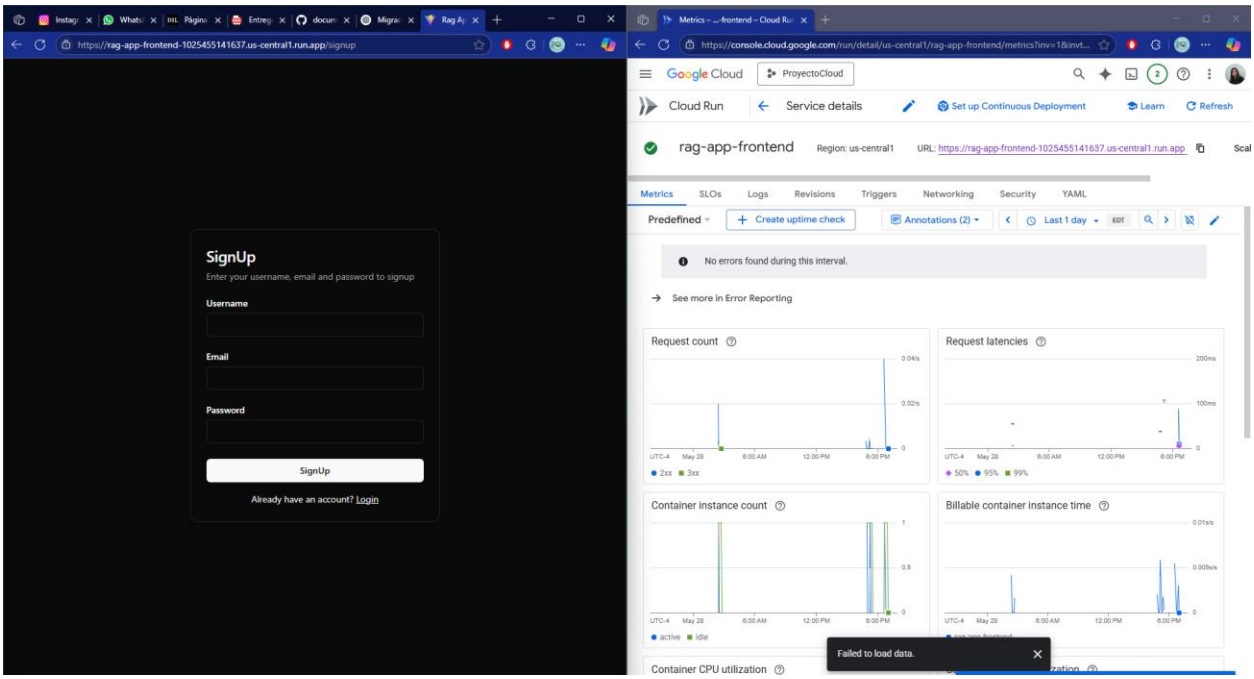


Ilustración 1. Despliegue del frontend

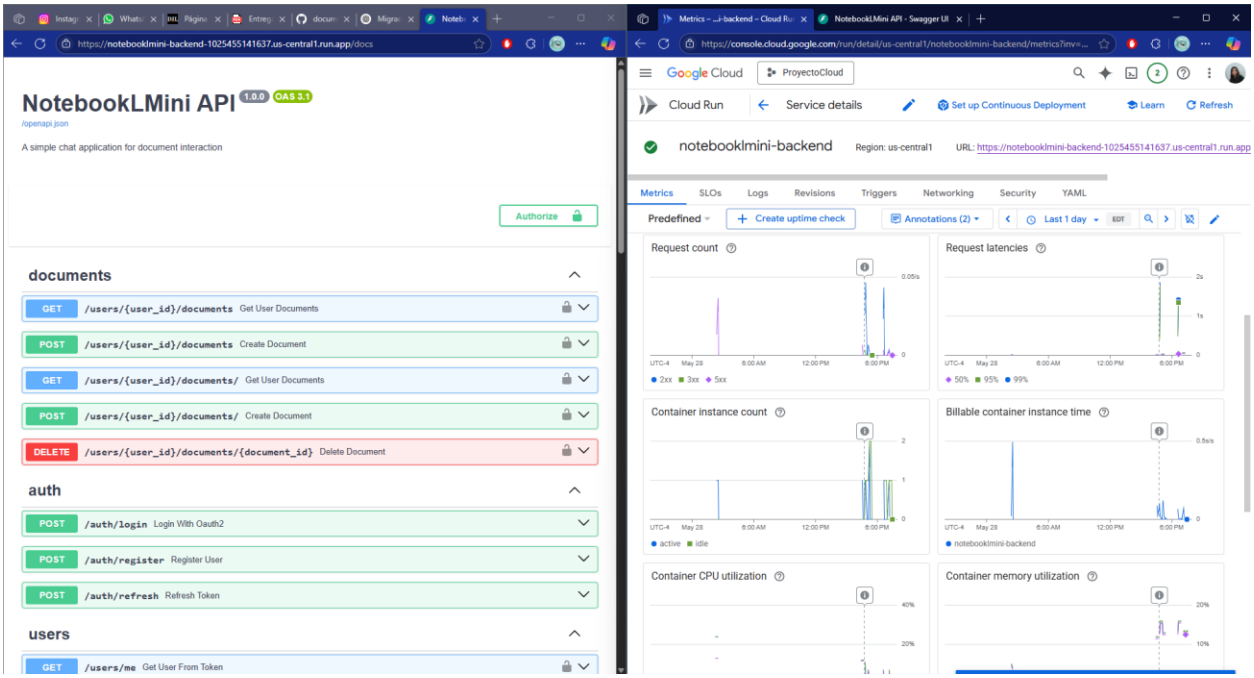


Ilustración 2. Despliegue del backend

Funcionamiento básico de la aplicación

En las imágenes a continuación se observa cómo el **frontend** de la aplicación envía peticiones al **backend alojado en Google Cloud Run** inmediatamente después de que el usuario inicia sesión. Estas solicitudes permiten recuperar la información del usuario autenticado y obtener las respuestas del LLM.

Además, si el usuario tiene documentos previamente cargados, estos se incluyen en la respuesta y se presentan en la interfaz. Se puede verificar que la **petición se realiza correctamente al servidor de Google Cloud Run**, ya que la respuesta se obtiene de manera exitosa y sin errores, lo que confirma la correcta integración entre los componentes de la arquitectura.

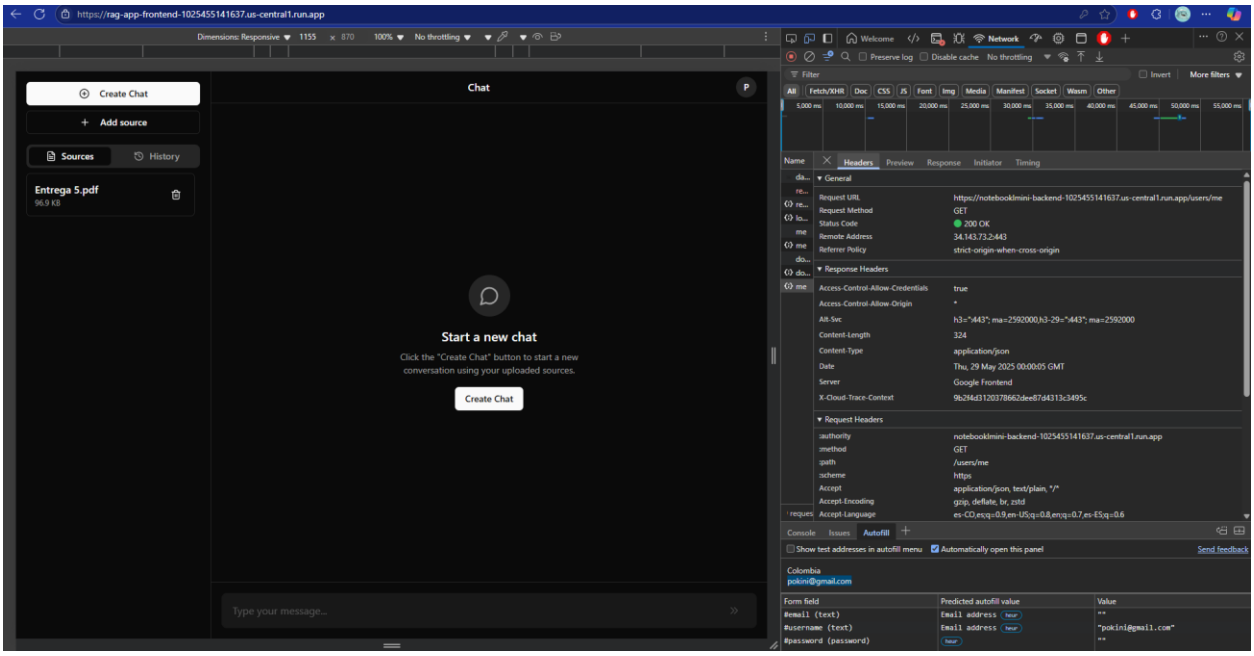


Ilustración 3. Información del Usuario (Petición)

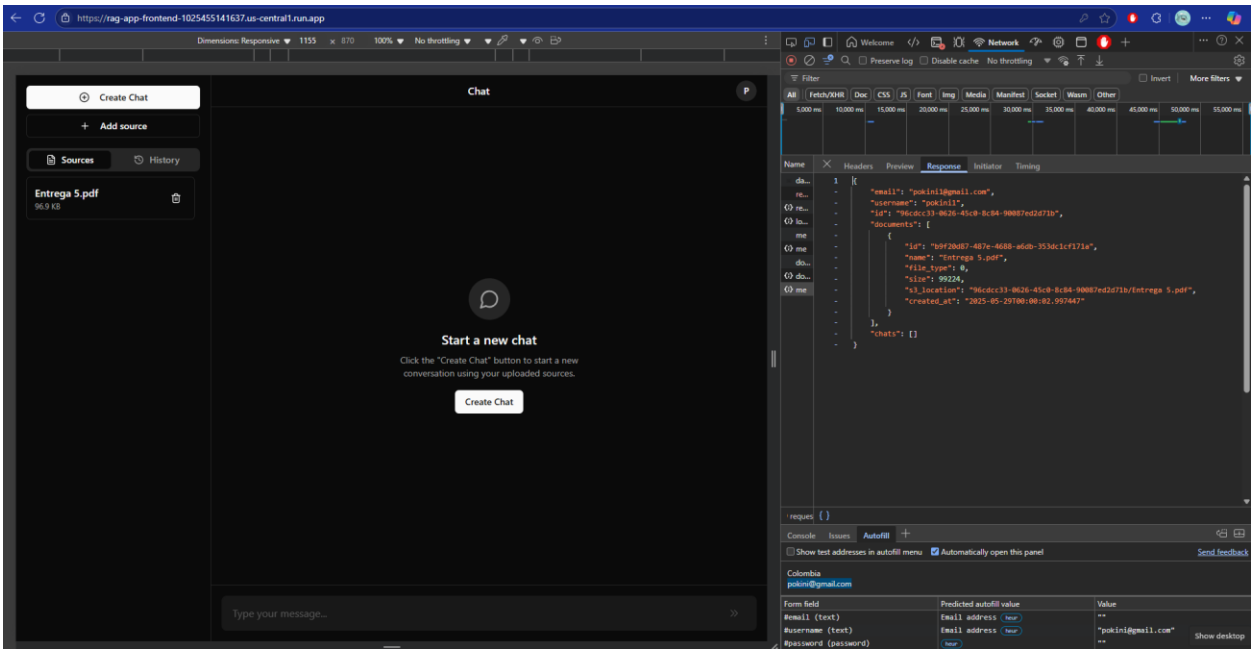


Ilustración 4. Información del Usuario (Respuesta)

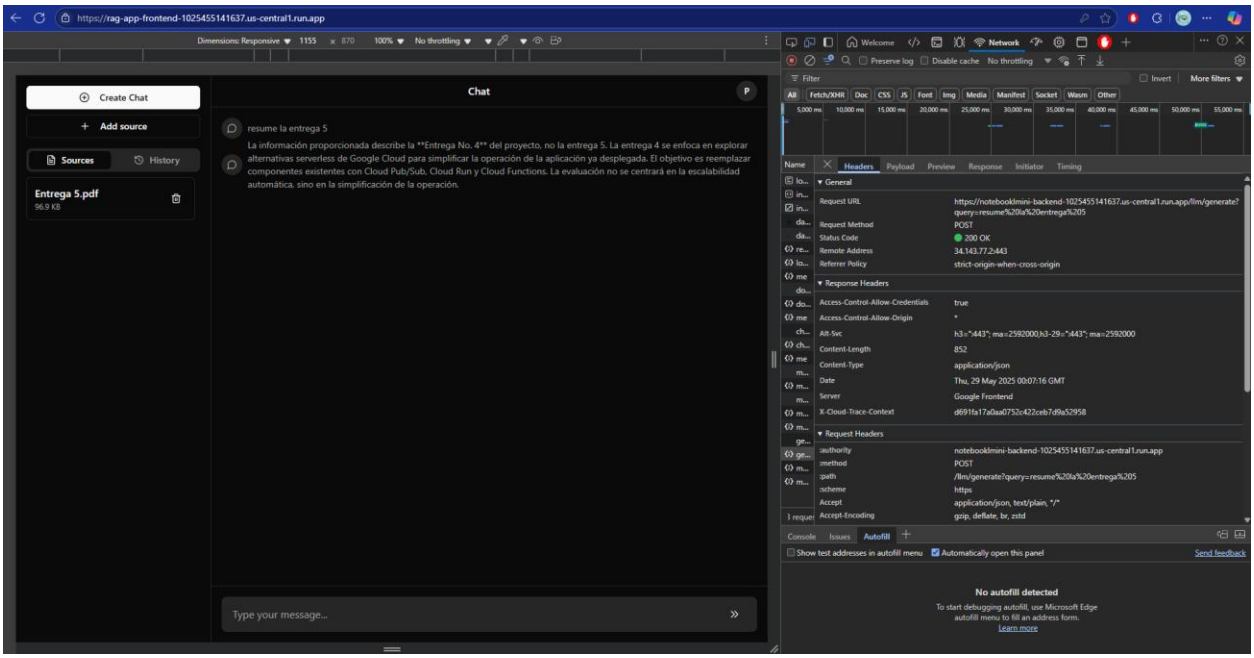


Ilustración 5. Petición al LLM para hacer un resumen de un documento (Petición)

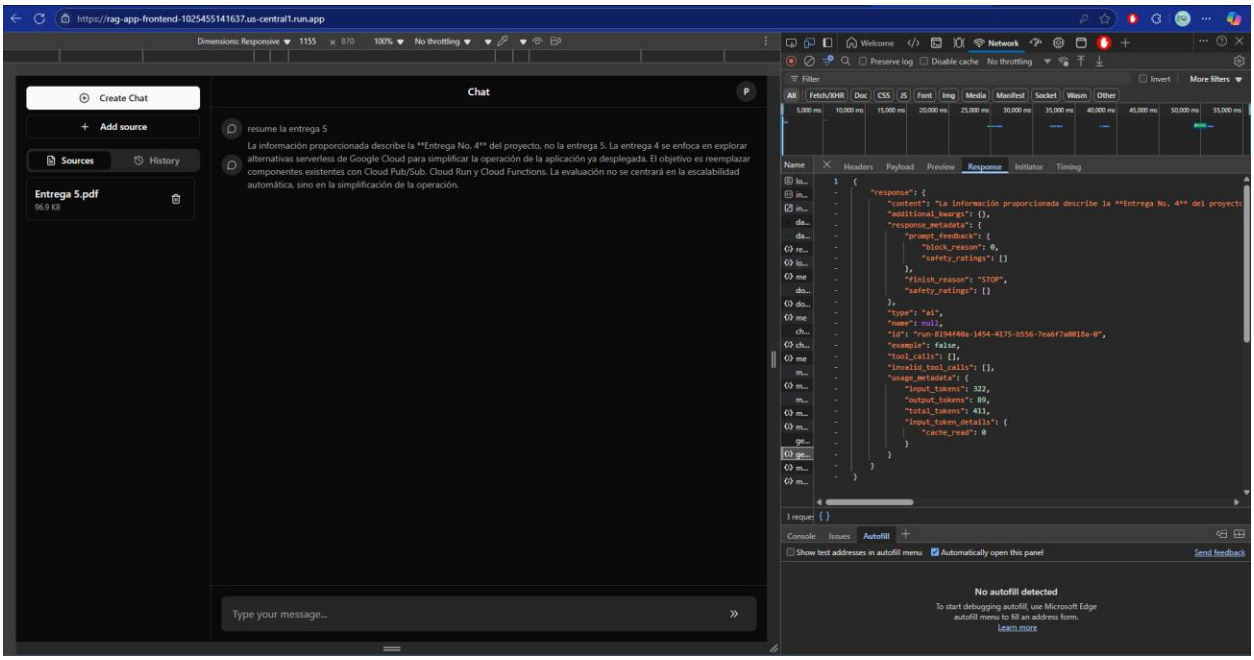


Ilustración 6. Respuesta del endpoint de LLM del backend

Limitaciones

A pesar de los avances logrados con la implementación de una arquitectura serverless moderna en Google Cloud, el Proyecto N4 presenta ciertas limitaciones inherentes a su diseño actual, a los servicios utilizados y al alcance del entorno de pruebas. A continuación, se detallan las principales restricciones identificadas:

1. Dependencia de Componentes Externos (Qdrant)

El uso de **Qdrant** como base de datos vectorial, aunque esencial para la funcionalidad de recuperación aumentada (RAG), implica una dependencia de un componente **no nativo de Google Cloud**, el cual debe ser gestionado manualmente en una VM. Esto conlleva:

- Mayor carga operativa en actualizaciones y monitoreo.
- Posible falta de integración directa con otros servicios serverless.
- Costos de mantenimiento y escalado no tan eficientes como los servicios nativos.

2. Persistencia en Máquinas Virtuales (Workers y Qdrant)

Si bien la mayoría de la arquitectura se apoya en servicios serverless, todavía se requieren **VMs para funciones específicas**:

- Procesamiento intensivo o dependiente de bibliotecas que no están soportadas en Cloud Functions.
- Servicios como Qdrant, que aún no están disponibles como oferta gestionada.

Esto contradice parcialmente el objetivo de operación completamente sin servidor y obliga a asumir tareas de infraestructura.

3. Costos Acumulativos

Aunque los servicios serverless facturan por uso, el conjunto de servicios (especialmente Cloud SQL y Cloud Storage) puede **acumular costos relevantes** a medida que crece la carga o el volumen de datos, afectando la viabilidad en escenarios de gran escala si no se optimizan recursos constantemente.