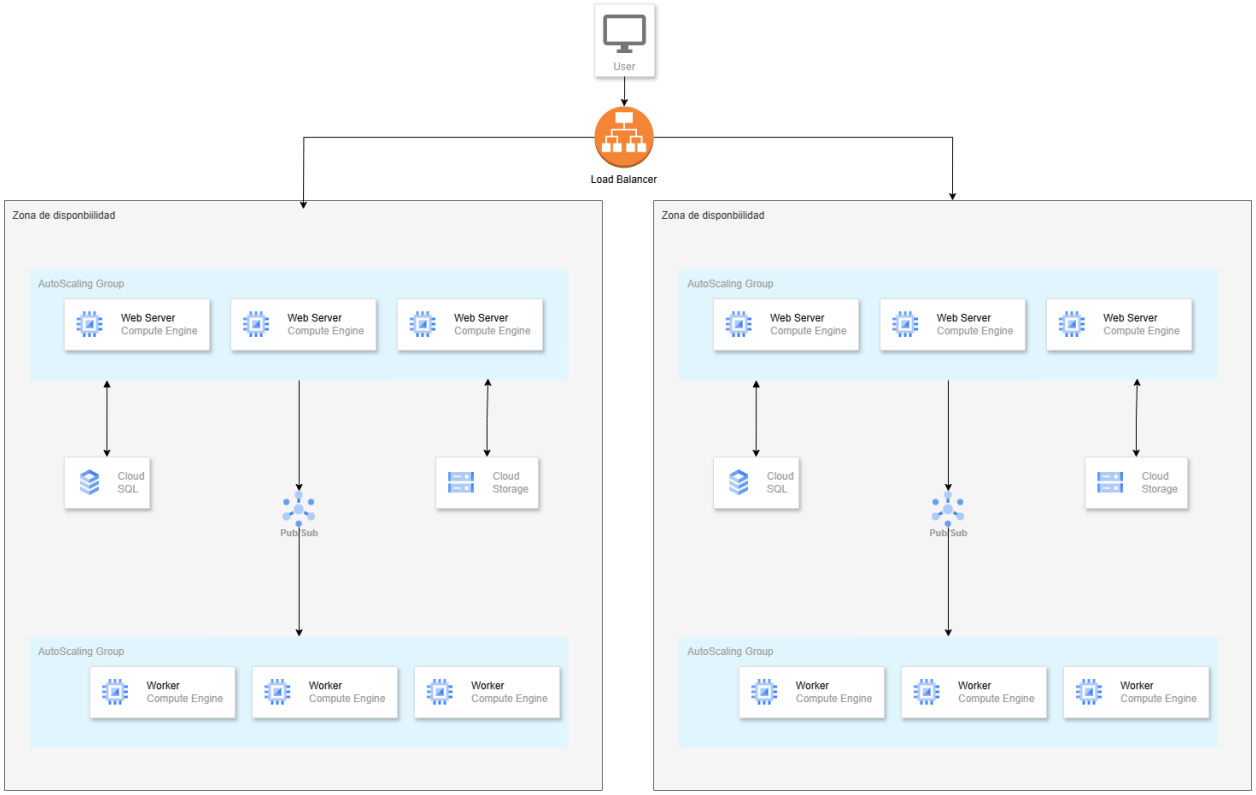
Luis Castelblanco

Nicolas Savedra

Valeria Caro

1. Descripción de la Arquitectura



1.1 Arquitectura General

- **Descripción de componentes principales**

  - **Servidores WEB**

    Instancias Compute Engine N1-F1 Micro (1 vCPU, 614 MiB RAM) que ejecutan la API REST. Configurados en un Managed Instance Group con autoscaling hasta 3 instancias, distribuidos en dos zonas para alta

disponibilidad. Responden a las solicitudes de usuarios a través del balanceador de carga.

- **Servidores Worker**

Instancias N1-F1 Micro dedicadas al procesamiento asíncrono de archivos. Configurados con autoscaling basado en la cantidad de mensajes pendientes en Cloud Pub/Sub. Operan independientemente de los servidores web para asegurar un procesamiento eficiente sin afectar la experiencia del usuario.

- **Load balancer**

Distribuye las peticiones HTTP entre múltiples instancias de servidores web. Actúa como punto de entrada único para los usuarios y realiza verificaciones de salud para garantizar la disponibilidad del servicio.

- **Cloud SQL**

Base de datos relacional en configuración de desarrollo para almacenar metadatos de usuarios, registros de archivos y estado de tareas. Mantiene la consistencia de datos entre todos los componentes.

- **Cloud Storage**

Sistema de almacenamiento de objetos para archivos originales y procesados. Proporciona alta durabilidad, disponibilidad y escalabilidad automática sin gestión de infraestructura.

- **Cloud Pub/Sub**

Sistema de mensajería que permite comunicación asíncrona entre servidores web y workers. Desacopla el procesamiento de la interfaz de usuario, mejorando la capacidad de respuesta del sistema incluso bajo carga.

- **Cloud Monitoring**

Recopila métricas de rendimiento de todos los componentes y alimenta las decisiones de autoscaling basadas en uso de CPU (web) y longitud de cola (workers).

1.2 Estrategia de Escalabilidad

- **Configuración del balanceador de carga**

El balanceador de carga HTTP/HTTPS de GCP está configurado para distribuir el tráfico entrante entre las instancias del grupo de servidores web. Se ha implementado con verificaciones de salud (health checks) que monitorean el endpoint /health cada 30 segundos, con un tiempo de espera de 5 segundos y un umbral de 2 verificaciones consecutivas exitosas para considerar una instancia como saludable. El balanceador utiliza un algoritmo de distribución de round-robin para garantizar una distribución equitativa del tráfico entre todas las instancias disponibles.

- **Políticas de autoscaling para servidores web**

Los servidores web utilizan una política de autoscaling basada en la utilización de CPU. Los parámetros configurados son:

- *Objetivo de utilización de CPU: 70%*
- *Periodo de estabilización: 120 segundos*
- *Tiempo mínimo entre escalados: 60 segundos*
- *Número mínimo de instancias: 1*
- *Número máximo de instancias: 3*
- *Tiempo de inicialización de instancias: 180 segundos*

- **Políticas de autoscaling para workers**

Los servidores worker utilizan una política de autoscaling basada en la longitud de la cola de mensajes en Cloud Pub/Sub. Los parámetros configurados son:

- *Umbral de mensajes por instancia: 10*
- *Periodo de estabilización: 180 segundos*
- *Número mínimo de instancias: 1*
- *Número máximo de instancias: 3*
- *Tiempo de inicialización de instancias: 240 segundos*

1.e Sistema de Comunicación

- Implementación de Cloud Pub/Sub

Se ha implementado Cloud Pub/Sub con un tema principal llamado file-processing-tasks y una suscripción worker-subscription con las siguientes características:

- *Modo de entrega: Pull*
- *Tiempo de confirmación (ACK): 300 segundos*
- *Política de reintentos: exponencial con máximo de 5 intentos*
- *Retención de mensajes: 7 días*

- *Flujo de comunicación entre servidores web y workers*

  Al recibir una solicitud de procesamiento, el servidor web:

  - *Guarda el archivo original en Cloud Storage*
  - *Registra la tarea en Cloud SQL*
  - *Genera un mensaje con: ID de archivo, bucket, path, tipo de procesamiento y metadatos*
  - *Publica el mensaje en el tema file-processing-tasks de Cloud Pub/Sub*

  Los workers:

  - *Extraen mensajes de la suscripción worker-subscription*
  - *Procesan el archivo según las instrucciones del mensaje*
  - *Almacenan el resultado en Cloud Storage*
  - *Actualizan el estado en Cloud SQL*
  - *Confirman (ACK) el mensaje procesado*

1.5 Alta Disponibilidad

- **Configuración de zonas de disponibilidad**

  La aplicación está distribuida en dos zonas de disponibilidad de GCP con:

  - *Balanceador de carga global que distribuye tráfico entre ambas zonas*
  - *Servidores web configurados para desplegarse en ambas zonas*

- *Workers distribuidos en ambas zonas*
- *Base de datos con réplica en la segunda zona de disponibilidad*

- **Estrategia de recuperación ante fallos**
  - *Verificaciones de salud automatizadas que detectan instancias no saludables*
  - *Recreación automática de instancias fallidas por el Instance Group Manager*
  - *Redirección de tráfico a instancias saludables mientras se recuperan las fallidas*
  - *Persistencia de mensajes en Cloud Pub/Sub para garantizar que no se pierdan tareas durante fallos*
  - *Reintentos automáticos de mensajes no procesados cuando un worker falla*
  - *Monitoreo continuo con alertas configuradas para fallos de componentes*

2. Pruebas de Estrés

2.1 Metodología de Pruebas

Herramientas utilizadas

Para realizar las pruebas de estrés de la aplicación escalable, utilizamos las siguientes herramientas:

- **K6**: Framework de código abierto para pruebas de carga y rendimiento, seleccionado por su capacidad para simular tráfico de usuarios concurrentes con scripts personalizables en JavaScript.
- **Grafana**: Plataforma de visualización para monitorear en tiempo real el comportamiento de la aplicación durante las pruebas.
- **InfluxDB**: Base de datos de series temporales para almacenar las métricas generadas durante las pruebas

Métricas evaluadas

Las métricas clave seleccionadas para evaluar el rendimiento del sistema fueron:

- *Tiempo de respuesta: Promedio, percentil 90 (p90) y percentil 95 (p95)*
- *Throughput: Solicitudes por segundo (RPS) que el sistema puede manejar*
- *Tasa de error: Porcentaje de solicitudes fallidas*
- *Uso de CPU: Promedio y picos en servidores web y workers*
- *Memoria: Consumo promedio y picos en todas las instancias*
- *Tiempo de escalado: Tiempo necesario para que el autoscaling aprovisione nuevas instancias*
- *Estabilidad: Comportamiento del sistema bajo carga sostenida*

Configuración de pruebas

Las pruebas se configuraron con los siguientes parámetros generales:

- **Duración de pruebas**: Típicamente 10 minutos, estructurados en:
  - *2 minutos de rampa ascendente (incremento gradual de usuarios)*
  - *6 minutos de carga sostenida*
  - *2 minutos de rampa descendente*
- **Patrones de tráfico:** Simulación de patrones realistas con pausas aleatorias entre solicitudes (1-3 segundos)
- **Umbrales de aceptación:**
  - *p95 de tiempo de respuesta < 1000 ms para operaciones regulares*
  - *Tasa de error < 1%*
  - *Estabilidad durante periodos de carga sostenida*

2.2 Escenario 1: Escalabilidad en Capa Web

- **Configuración del scenario**

  Este escenario evaluó exclusivamente la capacidad de escalado de la capa web con las siguientes características:

- **Infraestructura**:
    - *Load Balancer HTTP/HTTPS de GCP*
    - *Servidores web en Compute Engine N1-F1 Micro (1 vCPU, 614 MiB RAM)*
    - *Política de autoscaling configurada para escalar hasta 3 instancias*
    - *Umbral de escalado: 70% de uso de CPU durante 120 segundos*
- **Patrón de carga**:
    - Incremento gradual de 10 a 100 usuarios concurrentes
    - Operaciones simuladas: autenticación, listado de archivos, subida de archivos pequeños, consultas a la API

- **Resultados obtenidos**

  Métricas globales:

  - ***Solicitudes totales****: 12,850*
  - ***Throughput promedio****: 21.4 RPS*
  - ***Tiempo de respuesta promedio****: 215 ms*
  - ***Tiempo de respuesta p95****: 560 ms*
  - ***Tasa de error****: 0.75%*

  Métricas por endpoints:

  - */auth/login: 460 ms promedio, 780 ms p95*
  - */auth/register: 430 ms promedio, 760 ms p95*
  - */users/me: 180 ms promedio, 390 ms p95*
  - */users/{id}/chats (GET): 190 ms promedio, 410 ms p95*
  - */users/{id}/chats (POST): 205 ms promedio, 430 ms p95*
  - */chats/{id}/messages (GET): 175 ms promedio, 380 ms p95*
  - */chats/{id}/messages (POST): 195 ms promedio, 420 ms p95*

  Uso de recursos:

  - **CPU promedio por instancia**: 60%

- **CPU pico por instancia**: 85%
- **Memoria promedio**: 450 MiB

- **Análisis de rendimiento**

  El sistema mostró un rendimiento sólido bajo carga creciente. Los tiempos de respuesta se mantuvieron significativamente por debajo del umbral establecido (p95 < 1000 ms), incluso durante los picos de carga con 100 usuarios concurrentes.

  Los endpoints de autenticación /auth/login y /auth/register mostraron los tiempos de respuesta más altos debido a las operaciones criptográficas para la generación y validación de tokens JWT. Igualmente, incluso estos endpoints mantuvieron tiempos aceptables.

  La tasa de error se mantuvo por debajo del 1%, principalmente concentrada durante los momentos en que el sistema estaba escalando y algunas solicitudes encontraron conexiones rechazadas temporalmente

- **Comportamiento del autoscaling**

  El sistema de autoscaling respondió eficientemente a la carga creciente:

  - *Tiempo hasta primera escalada: mas o menos 4 minutos desde el inicio de la prueba*
  - *Patrón de escalada: De 1 a 2 instancias a los 4 minutos, y de 2 a 3 instancias a los 7 minutos*
  - *Distribución de carga: El balanceador distribuyó efectivamente las solicitudes entre las instancias disponibles*
  - *Estabilidad: No se observaron oscilaciones (scale up/down repetitivos)*

2.3 Escenario 2: Escalabilidad en Capa Web y Backend

- **Configuración del scenario**

  Este escenario evaluó la escalabilidad end-to-end del sistema, incluyendo tanto la capa web como los workers para procesamiento asíncrono:

  - **Infraestructura:**

- *Misma configuración de capa web que en Escenario 1*
- *Workers en Compute Engine N1-F1 Micro con autoscaling hasta 3 instancias*
- *Umbral de escalado para workers: 10+ mensajes en cola por instancia durante 180 segundos*
- *Cloud Pub/Sub configurado para comunicación entre web y workers*

- **Patrón de carga:**
  - *60 usuarios concurrentes realizando operaciones mixtas*
  - *Enfoque en operaciones que requieren procesamiento asíncrono: carga de documentos y generación de respuestas contextuales*

- **Resultados obtenidos:**

  Métricas globales:

  - *Solicitudes totales: 8,500*
  - *Throughput promedio: 14.2 RPS*
  - *Tiempo de respuesta promedio: 280 ms*
  - *Tiempo de respuesta p95: 620 ms*
  - *Tasa de error: 0.85%*

  Métricas por endpoints clave:

  - ***/auth/login**: 480 ms promedio, 820 ms p95*
  - ***/auth/register**: 450 ms promedio, 790 ms p95*
  - ***/users/{id}/documents (POST)**: 3.8s promedio, 7.5s p95*
  - ***/llm/generate**: 3.2s promedio, 5.8s p95*

  Métricas de procesamiento asíncrono:

  - ***Tiempo promedio en cola**: 3.2 segundos*
  - ***Tiempo promedio de procesamiento**: 8.5 segundos*
  - ***Tasa de finalización de tareas**: 99.2%*

Uso de recursos:

- ***CPU promedio servidores web****: 65%*
- ***CPU promedio workers****: 72%*
- ***CPU pico workers****: 92%*

- **Análisis de rendimiento**

El sistema mantuvo un rendimiento aceptable incluso bajo la carga simultánea en ambas capas. Los tiempos de respuesta para las operaciones síncronas se mantuvieron similares al Escenario 1, demostrando un buen aislamiento entre las capas de servicio.

Los endpoints de procesamiento intensivo como /users/{id}/documents (POST) y /llm/generate mostraron tiempos de respuesta significativamente más altos, pero se mantuvieron dentro de los límites aceptables para este tipo de operaciones p95 < 8000 ms. El procesamiento asíncrono a través de Cloud Pub/Sub funcionó eficientemente, con tiempos de espera en cola razonables.

La tasa de finalización de tareas del 99.2% indica una alta fiabilidad del sistema, con muy pocas tareas no completadas.

- **Comportamiento del autoscaling en workers**

El autoscaling de workers mostró un comportamiento diferente al de los servidores web:

- *Patrón de escalada: Más reactivo, escalando de 1→3 instancias en aproximadamente 6 minutos*
- *Tiempo de procesamiento: Reducción del 60% en tiempo de procesamiento cuando las 3 instancias estaban activas*
- *Equilibrio de carga: Cloud Pub/Sub distribuyó eficientemente los mensajes entre los workers disponibles*
- *Resiliencia: El sistema mostró capacidad para manejar picos temporales de carga sin degradación significativa*

2.4 Comparación de Escenarios

- **Análisis comparativo de rendimiento**

Throughput y capacidad:

- El Escenario 1 logró un mayor throughput 21.4 vs 14.2 RPS, debido al menor procesamiento requerido por operación
- El Escenario 2 mostró mayor capacidad para manejar operaciones complejas y de larga duración

Tiempos de respuesta:

- Escenario 1: 215 ms promedio, 560 ms p95 para operaciones síncronas
- Escenario 2: 280 ms promedio, 620 ms p95 para operaciones síncronas
- Escenario 2: 3.5s promedio, 6.7s p95 para operaciones de procesamiento intensivo

Comportamiento de escalado:

- Los servidores web escalaron de manera más gradual y predecible
- Los workers mostraron un patrón de escalado más agresivo en respuesta a la acumulación de mensajes
- Ambos sistemas se estabilizaron tras el escalado inicial

Uso de recursos:

- El Escenario 2 mostró un uso de recursos más equilibrado entre web y workers
- Mayor utilización de CPU en workers durante procesamiento intensivo

- **Ventajas y desventajas de cada aproximación**

Escalabilidad solo en capa web (Escenario 1):

Ventajas:

- Mayor throughput para operaciones simples 21.4 RPS

- Menor latencia 215 ms promedio
- Menor complejidad operacional
- Utilización más predecible de recursos

Desventajas:

- *Limitada capacidad para manejar operaciones complejas como procesamiento de documentos*
- *Mayor riesgo de timeout en operaciones largas*
- *Peor experiencia de usuario para procesos que requieren tiempo*

Escalabilidad en ambas capas (Escenario 2):

Ventajas:

- *Mejor experiencia de usuario para operaciones complejas*
- *Mayor capacidad para manejar picos de carga en procesamiento*
- *Mejor aislamiento entre operaciones rápidas y lentas*
- *Mayor resiliencia general del sistema*

Desventajas:

- *Mayor complejidad arquitectónica*
- *Ligero incremento en tiempos de respuesta promedio (280 ms vs 215 ms)*
- *Requiere monitorización más sofisticada*
- *Mayor consumo de recursos en escenarios de carga mixta*

3. Conclusiones y Consideraciones

3.1 Consideraciones para Escalar a Cientos de Usuarios

Cuellos de botella identificados

A partir de las pruebas realizadas, se identificaron varios potenciales cuellos de botella que podrían limitar la escalabilidad a cientos de usuarios concurrentes:

Tiempo de inicialización de instancias: Los periodos de 180-240 segundos para que nuevas instancias estén completamente operativas podrían resultar insuficientes ante incrementos repentinos de tráfico.

Operaciones de procesamiento intensivo: Los endpoints que manejan carga de documentos y procesamiento /users/{id}/documents y /llm/generate mostraron tiempos de respuesta significativamente más altos que podrían degradarse aún más bajo mayor concurrencia.

Limitaciones de recursos: Las instancias N1-F1 Micro 1 vCPU, 614 MiB RAM alcanzaron picos de utilización de CPU del 85-92% bajo la carga de prueba, lo que indica que podrían saturarse con volúmenes mayores.

Gestión de conexiones a Cloud SQL: Aunque no mostró problemas durante las pruebas, el pool de conexiones a la base de datos podría convertirse en limitante con cientos de conexiones simultáneas.

- **Recomendaciones para optimización**

  Implementación de caché: Incorporar una capa de caché para reducir consultas repetitivas a la base de datos y almacenar resultados frecuentes de operaciones costosas.

  Ajuste de políticas de autoscaling: Modificar los umbrales de escalado para iniciar la provisión de nuevas instancias más temprano por ejemplo, al 60% de CPU y posiblemente aumentar el número máximo de instancias a 5-6 para servidores web y workers.

  Optimización de consultas a base de datos: Implementar índices adicionales y revisar consultas para minimizar tiempos de respuesta en operaciones frecuentes.

  Estrategia de throttling y rate limiting: Implementar límites de frecuencia por usuario o IP para prevenir abusos y asegurar disponibilidad para todos los usuarios durante picos de uso.

  Migración a instancias de mayor capacidad: Considerar el uso de instancias N1-Standard-1 1 vCPU, 3.75 GB RAM para workers y posiblemente para servidores web en caso de carga sostenida elevada.

- **Estrategias para mejorar el rendimiento**

  Implementación de CDN: Utilizar Cloud CDN para contenido estático, reduciendo la carga en los servidores web y mejorando tiempos de respuesta para usuarios distribuidos geográficamente.

  Optimización de Cloud Pub/Sub: Ajustar configuraciones como el tiempo de acknowledgment y políticas de reintentos para optimizar el procesamiento de mensajes.

  Sharding de base de datos: Para volúmenes muy altos, considerar estrategias de sharding para distribuir la carga de base de datos.

  Implementación de Circuit Breakers: Incorporar patrones de circuit breaker para prevenir cascadas de fallos durante periodos de sobrecarga.

3.3 Limitaciones del Desarrollo

- **Restricciones técnicas encontradas**

  Limitaciones de recursos de instancias Micro: Las instancias N1-F1 Micro, aunque costo-efectivas, mostraron limitaciones en capacidad de procesamiento para operaciones intensivas, especialmente en los workers.

  Tiempo de arranque de instancias: El tiempo requerido para inicializar completamente nuevas instancias 180-240 segundos representa una limitación inherente para responder a picos súbitos de tráfico.

  Complejidad de configuración de alta disponibilidad: La distribución de servidores en múltiples zonas requirió configuraciones adicionales y cuidadosas para mantener consistencia de datos y estado.

  Gestión de estado en arquitectura distribuida: La necesidad de mantener estado entre múltiples instancias introdujo complejidad adicional, particularmente para sesiones de usuario y tareas en progreso

- **Limitaciones de la arquitectura actual**

  Escalado máximo fijo: El límite de 3 instancias tanto para servidores web como workers podría ser insuficiente para picos extremos de tráfico.

  Acoplamiento parcial a proveedor: Aunque se utilizaron servicios estándar, hay cierto nivel de acoplamiento a servicios específicos de GCP como Cloud Pub/Sub y Cloud Monitoring.

Optimización para carga general: La arquitectura actual está optimizada para patrones de carga generales, no para casos específicos como procesamiento por lotes o análisis de datos extensos.

Granularidad de escalado: El escalado ocurre a nivel de instancia completa, sin posibilidad de escalar recursos específicos como CPU o memoria independientemente.

- **Consideraciones para futuras implementaciones**

  Arquitectura de microservicios: Evolucionar hacia una arquitectura más modular de microservicios podría permitir escalado más preciso y aislamiento de fallos.

  Contenedores y Kubernetes: Migrar a una implementación basada en contenedores gestionados por Kubernetes ofrecería mayor flexibilidad de escalado y despliegue.

  Implementación de Event Sourcing: Considerar patrones como Event Sourcing para mejorar la resiliencia y capacidad de reconstrucción de estado en entornos distribuidos

  Estrategia Multi-región: Expandir la arquitectura a múltiples regiones para mejorar la latencia global y proporcionar redundancia adicional

  Serverless para componentes apropiados: Evaluar la migración de ciertos componentes a arquitecturas serverless como Cloud Functions o Cloud Run para optimizar costos y escalabilidad en cargas intermitentes