

Entrega 1 – Análisis de Capacidad

Aplicaciones Web Escalables en un Entorno Tradicional

Grupo 20

Juan P Hernández

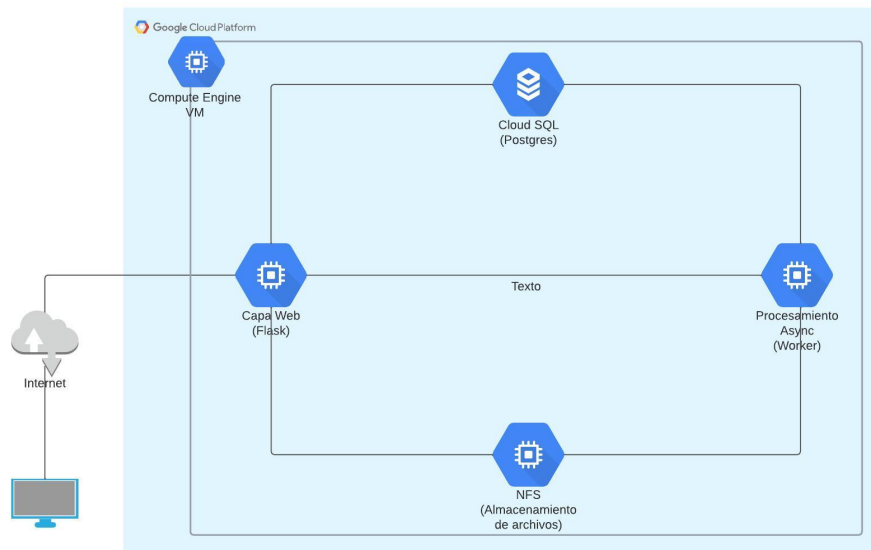
Juan D Díaz

Juan F Castaño

Kevin H Castrillón

1. Arquitectura de la aplicación.

1.1. Diagrama Arquitectura



1.2. Infraestructura

Para la infraestructura se utilizó las siguientes soluciones tecnológicas:

Máquina virtual: se configuró una instancia de tipo N1 con los requerimientos de cómputo y almacenamiento definidos, es decir, 1 GB de RAM, 1vCPU y 20GB de RAM. El sistema operativo instalado en la misma es Ubuntu 20.04. Se le agregó una etiqueta de red para que las reglas necesarias de firewall para acceder desde internet fueran posibles.

Instancia SQL: se configuró una instancia de SQL en la cual corrimos PostgreSQL 14.

Red VPC: se aprovechó la red VPC configurada por defecto en un proyecto de Google Cloud. Se configuraron reglas de firewall respectivas para poder acceder a la máquina virtual por TCP en el puerto 5000. Se aprovechó el puerto 3389 que ya está configurado por defecto para acceder por TCP.

Para el ambiente de software se usaron las siguientes tecnologías:

A continuación, se listan las tres principales herramientas tecnológicas usadas en la máquina virtual para el desarrollo de la entrega 1:

Flask: framework web minimalista para Python. Es una herramienta popular utilizada para construir aplicaciones web pequeñas y medianas que son rápidas y eficientes.

Celery: herramienta de Python que permite la programación de tareas en segundo plano y la ejecución de trabajos de forma asíncrona y distribuida. Se utiliza ampliamente en aplicaciones web para procesar tareas que pueden ser realizadas fuera del ciclo de solicitud-respuesta.

Redis: sistema de almacenamiento en memoria de código abierto, rápido y eficiente que se utiliza como base de datos, caché y agente de mensajes. Fue diseñado para manejar grandes cantidades de datos y proporcionar un alto rendimiento y disponibilidad.

2. Análisis de capacidad:

2.1 Entorno de pruebas.

Máquina virtual: se configuró una instancia de tipo N1 con los requerimientos de cómputo y almacenamiento definidos, es decir, 1 GB de RAM, 1vCPU y 20GB de RAM. El sistema operativo instalado en la misma es Ubuntu 20.04. Se le agregó una etiqueta de red para que las reglas necesarias de firewall para acceder desde internet fueran posibles.

Instancia SQL: se configuró una instancia de SQL en la cual corrimos PostgreSQL 14.

Red VPC: se aprovechó la red VPC configurada por defecto en un proyecto de Google Cloud. Se configuraron reglas de firewall respectivas para poder acceder a la máquina virtual por TCP en el puerto 5000. Se aprovechó el puerto 3389 que ya está configurado por defecto para acceder por TCP.

Para el ambiente de software se usaron las siguientes tecnologías:

A continuación se listan las tres principales herramientas tecnológicas usadas en la máquina virtual para el desarrollo de la entrega 1:

Flask: framework web minimalista para Python. Es una herramienta popular utilizada para construir aplicaciones web pequeñas y medianas que son rápidas y eficientes.

Celery: herramienta de Python que permite la programación de tareas en segundo plano y la ejecución de trabajos de forma asíncrona y distribuida. Se utiliza ampliamente en aplicaciones web para procesar tareas que pueden ser realizadas fuera del ciclo de solicitud-respuesta.

Redis: sistema de almacenamiento en memoria de código abierto, rápido y eficiente que se utiliza como base de datos, caché y agente de mensajes. Fue diseñado para manejar grandes cantidades de datos y proporcionar un alto rendimiento y disponibilidad.

Paquetes de Python

flask==2.2.3

flask-SQLAlchemy==3.0.3

flask-RESTful==0.3.9

flask-marshmallow==0.15.0

marshmallow_sqlalchemy==0.29.0

marshmallow==3.19.0

flask-jwt-extended==4.4.4
flask-cors==3.0.10
werkzeug==2.2.3
celery==5.2.7
redis==4.5.4
gunicorn==20.1.0
psycpg2-binary==2.9.2
pydump==0.25.1
pylzma==0.5.0

2.2 Criterios de aceptación.

Rta/ Los criterios de aceptación para las pruebas de estrés incluyen los siguientes objetivos y limitaciones:

Tiempo de respuesta: se espera que la aplicación responda a las solicitudes de los usuarios en un tiempo razonable, sin generar tiempos de espera excesivos. El objetivo es que el tiempo de respuesta promedio sea inferior a 3 segundos, con un límite máximo de 5 segundos.

Rendimiento: se espera que la aplicación pueda manejar una carga de trabajo adecuada, sin experimentar una disminución significativa en el rendimiento. El objetivo es que la aplicación pueda manejar al menos 100 usuarios concurrentes sin afectar negativamente el rendimiento.

Utilización de recursos: se espera que la aplicación utilice los recursos del sistema de manera eficiente, sin generar cuellos de botella en el procesamiento o en la utilización de memoria. El objetivo es que la utilización de recursos se mantenga por debajo del 80% en todo momento.

Además de estos objetivos y limitaciones, los criterios de éxito del proyecto también incluyen la identificación de los cuellos de botella en la aplicación y su infraestructura, y la determinación de las configuraciones óptimas para maximizar el rendimiento.

2.3 Escenarios de prueba

Rta/ Escenarios de prueba:

1. Conversión de (1, 10, 100) archivos pequeños a ZIP.
2. Conversión de (1, 10, 100) archivos grandes a ZIP.
3. Conversión de (1, 10, 100) archivos pequeños a 7Z.
4. Conversión de (1, 10, 100) archivos grandes a 7Z.
5. Conversión de (1, 10, 100) archivos pequeños a TAR.GZ.
6. Conversión de (1, 10, 100) archivos grandes a TAR.GZ.
7. Conversión de (1, 10, 100) archivos pequeños a TAR.BZ2.
8. Conversión de (1, 10, 100) archivos grandes a TAR.BZ2.

Escenarios clave:

- Comprobar que la aplicación puede manejar diferentes tipos de archivos y tamaños.

- Verificar que la conversión de los archivos se realiza correctamente a los diferentes formatos.
- Validar que la aplicación no pierda información o datos durante el proceso de compresión.
- Asegurar que la aplicación mantiene la estructura de carpetas durante la compresión.

Variabilidad entre servicios representativos:

Se puede simular la variabilidad en los servicios representativos de la aplicación mediante la generación de diferentes tipos de archivos de diferentes tamaños y formatos.

Datos de prueba:

- Archivos pequeños y grandes de diferentes formatos.
- Varios archivos para comprimir.
- Si el archivo comprimido se puede abrir correctamente.
- Si la estructura de carpetas se mantiene durante la compresión.

2.4 Parámetros de configuración.

Los parámetros para el get se configuran inicialmente en 500 thread y van decreciendo hasta el punto de éxito de éxito y fallo del escenario, además se autoriza a través de jwt.

Los parámetros para el Post se configuran inicialmente en 100 thread y van decreciendo hasta el punto de éxito de éxito y fallo del escenario, además se autoriza a través de jwt.

2.5 Escenario 1.

Se prueba obtener un archivo procesado, la restricción es la siguiente (En las pruebas de estrés el tiempo de respuesta promedio de la aplicación debe ser de máximo 1.500 ms, si este tiempo no se cumple, se concluye que el sistema NO soporta la cantidad de requests de la prueba. En caso de que durante una prueba se generen más de un 1% de errores en los requests de la prueba, se concluye que la aplicación NO soporta la cantidad de requests de la prueba.), se desea saber cuántas conexiones máximas se pueden tener en simultaneo, tal que se cumplan

Ensayo con 420 hilos: Se valida que con 420 hilos sigue funcionando cumpliendo el estándar.

Pruebas Automatizadas.jmx (J:\Users\Juan\Desktop\Universidad\BSemestre\ProgEnNube\ConvertToolG20\data\pruebas_jmeter\Pruebas Automatizadas.jmx) - Apache JMeter (5.5)

Summary Report

Name: Summary Report

Comments:

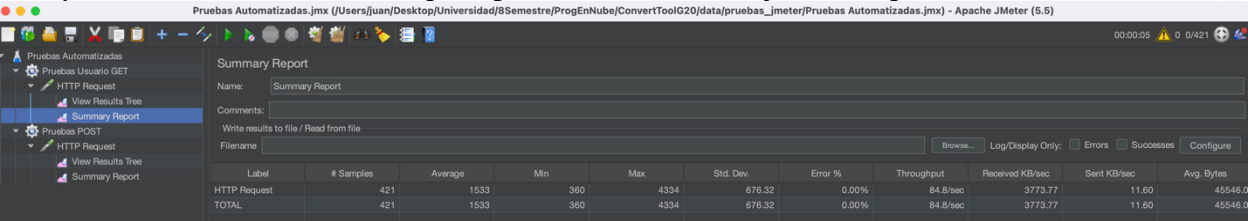
Write results to file / Read from file

Filename: Browse...

Log/Display Only: ☐ Errors ☐ Successes ☐ Configure

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
HTTP Request	420	1475	377	4299	647.37	0.00%	86.7/sec	3858.11	11.86	43546.0
TOTAL	420	1475	377	4299	647.37	0.00%	86.7/sec	3858.11	11.86	43546.0

Ensayo con 421 hilos: Se valida que a partir de 421 hilos deja de cumplir el estándar.



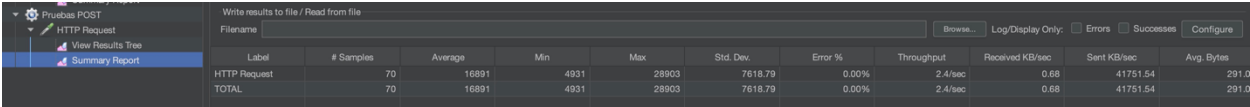
The screenshot shows the Apache JMeter Summary Report for a test named 'Pruebas Automatizadas.jmx'. The report is for a 'Summary Report' with 421 samples. The table below shows the performance metrics for the 'HTTP Request' and 'TOTAL'.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
HTTP Request	421	1533	360	4334	676.32	0.00%	84.8/sec	3773.77	11.60	45546.0
TOTAL	421	1533	360	4334	676.32	0.00%	84.8/sec	3773.77	11.60	45546.0

2.6 Escenario 2.

Se prueba enviar y procesar un archivo, se busca saber cuántos archivos máximo se pueden procesar en un tiempo menos a 600 segundos y con 0% de error que nos indica que se procese correctamente

Ensayo con 70 hilos: Se valida que con 70 hilos sigue funcionando cumpliendo el estándar.



The screenshot shows the Apache JMeter Summary Report for a test named 'Pruebas POST'. The report is for a 'Summary Report' with 70 samples. The table below shows the performance metrics for the 'HTTP Request' and 'TOTAL'.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
HTTP Request	70	16891	4931	28903	7618.79	0.00%	2.4/sec	0.68	41751.54	291.0
TOTAL	70	16891	4931	28903	7618.79	0.00%	2.4/sec	0.68	41751.54	291.0

Ensayo con 85 hilos: Se valida que entre el rango de 75-85 se empiezan a generar errores y no cumple el estándar.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
HTTP Request	85	25029	8223	41634	8536.09	32.94%	2.0/sec	2.31	23690.26	
TOTAL	85	25029	8223	41634	8536.09	32.94%	2.0/sec	2.31	23690.26	