Proyecto 2

Documentación análisis y diseño del sistema

Abraham Miguel Lora Vargas

amlorav@eafit.edu.co

Mario Alejandro Muñetón Durango

mamunetond@eafit.edu.co

Grupo 001

Universidad EAFIT

Departamento de informática y sistemas

ST0263-001 Tópicos especiales en telemática

Medellín

2021

Tabla de contenidos:

1. **Asignación de roles y responsabilidades de cada integrante del equipo en el desarrollo del proyecto:**

1.1. Roles:

a) **Documentación de la Comunidad de Aprendizaje:** Mario Alejandro Muñetón Durango.

b) **Documentación del análisis y diseño del sistema:** Mario Alejandro Muñetón Durango.

c) **Documentación técnica del proceso de instalación:** Abraham Miguel Lora Vargas.

d) **GitHub del proyecto 2:** Abraham Miguel Lora Vargas.

1.2. Responsabilidades:

a) **Consultar acerca de los proyectos de la materia proyecto integrador 1 y 2 con los profesores:** Mario Alejandro Muñetón Durango

b) **Aplicación monolítica desplegada en AWS:** Abraham Miguel Vargas Lora y Mario Alejandro Muñetón Durango.

c) **Crear una cuenta de GCP para el proyecto 2:** Mario Alejandro Muñetón Durango.

d) **Comprar dominio grupal para el proyecto 2 en NameSilo:** Abraham Miguel Vargas Lora.

e) **Crear repositorio en GitHub para el proyecto 2:** Abraham Miguel Vargas Lora.

f) **Diseño de la solución:** Mario Alejandro Muñetón Durango.

g) **Implementación de la solución planteada:** Abraham Miguel Vargas Lora.

h) **Monitoreo para el portal web desplegado:** Mario Alejandro Muñetón Durango.

i) **Presupuesto y estimación de costos:** Abraham Miguel Vargas Lora.

2. **GitHub del proyecto:**

Enlace del proyecto 2: https://github.com/ToxicSSJ/blogpi

3. **Especificación de requisitos no funcionales:**

a) **Disponibilidad:** Nuestra CMS cuenta con una alta disponibilidad en la capa de servicios, es decir, es capaz de garantizar la continuidad de los servicios, incluso en situaciones donde se presenten fallas, tiene una arquitectura de diseño que replica el almacenamiento de Host y Networking, eliminando puntos únicos de fallos y cuellos de botella. A su vez tiene alta disponibilidad en la capa de persistencia de datos, ya que realiza una replicación multirregional, esto consiste en hacer una copia del Google Cloud Storage en una región diferente, de esta manera los usuarios de diferentes países puedan acceder a los archivos que hay dentro del portal web. Por último, se maneja una alta disponibilidad en la capa de bases de datos, al igual que en el manejo de archivos, se hace una replicación multirregional de la Database en otra región, de esta manera, sí alguien desea subir archivos al portal web, otra persona que se encuentre ubicado en una región diferente pueda visualizar los cambios que hizo la otra persona.

b) **Rendimiento:** El rendimiento es la medición objetiva y la experiencia percibida por el usuario del tiempo de carga y de ejecución. El rendimiento web es el tiempo que tarda un sitio en cargarse, en ser interactivo y receptivo, y en el grado de fluidez del contenido durante las interacciones de usuario, el rendimiento web incluye mediciones objetivas como el tiempo de carga, cuadros por segundo, y subjetivas de cuanto tiempo tardó en cargarse el contenido. El diseño de la solución plantea minimizar los tiempos de carga y respuesta entre 1 y 2 segundos como máximo, y agregar funciones adicionales para ocultar la latencia.

c) **Seguridad:** La seguridad web implica invertir en acciones y soluciones centradas en garantizar la protección de datos. Para lograr este objetivo, es necesario mantener WordPress actualizado, usar contraseñas seguras, configurar perfiles de usuarios, hacer backup de seguridad y autenticación en dos pasos. La seguridad web consiste en cada acción o herramienta adoptada para evitar que la información sea expuesta o propensas a recibir ataques. Nuestra CMS cuenta con CloudFlare para ataques DDoS y certificado SSL.

4. Atributos de calidad:

a) **Idoneidad:** Capacidad del producto software para proporcionar un conjunto apropiado de funciones para tareas y objetivos de usuario especificado. En nuestro caso consultar los proyectos de las materias de Proyecto Integrador 1 y 2 para el programa de ingeniería de sistemas de la Universidad EAFIT

b) **Tolerancia a fallos:** Capacidad del software para mantener un nivel especificado de prestaciones en cada fallo de software o de infringir sus interfaces especificados.

c) **Inteligibilidad:** Capacidad del producto software que permite al usuario entender si el software es adecuado y como puede ser usado para unas tareas o condiciones de uso en particular. Los estudiantes comprenden como usar la CMS, para consultar el proyecto que deseen acceder.

d) **Utilización de recursos:** Capacidad del producto software para usar las cantidades y tipos de recursos adecuados cuando el software lleva a cabo su función bajo condiciones determinadas

e) **Adaptabilidad:** Capacidad del producto software para ser adaptado a diferentes entornos especificados, sin aplicar acciones o mecanismos distintos de aquellos proporcionados para este propósito por el propio software considerado.

5. Diseño para la escalabilidad (disponibilidad, rendimiento, seguridad).

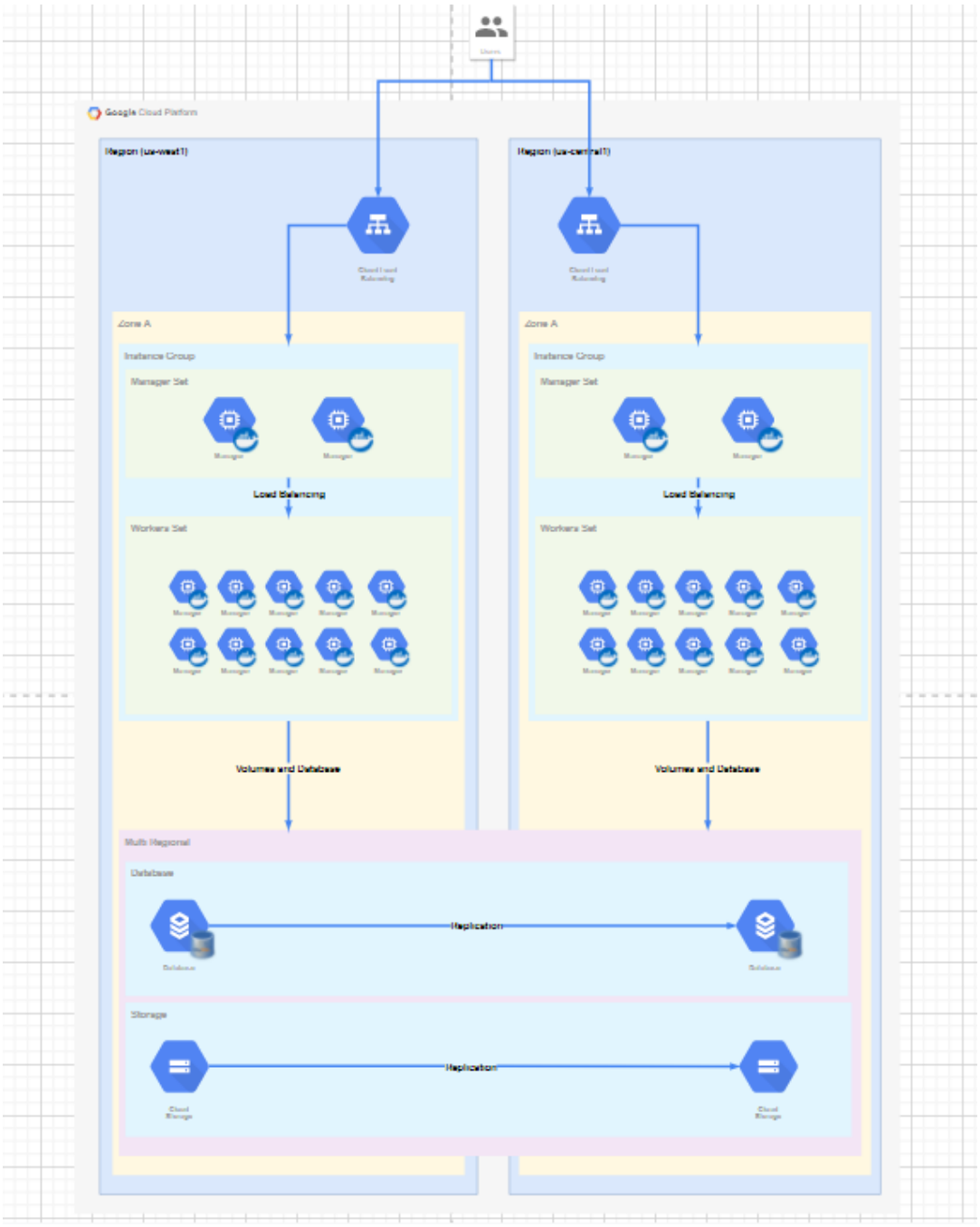

Figura 1. Diseño de la solución planteada, elaborado en diagrams.net usando una plantilla preestablecida de Google Cloud Platform (GCP).

El diseño de la solución fue planteado para un sistema de 20.000 usuarios, con un nivel de concurrencia del 10%, almacenamiento total de 500 GB, y permite el tráfico por un mismo país (80%) y fuera del país (20%), cada usuario estará conectado por una conexión de banda ancha de mínimo 20Mbps.

El usuario se conecta con cualquier dispositivo, celular, computador, tableta, desde cualquier parte del mundo donde se encuentre, como es un sistema para 20.000 usuarios y una concurrencia al 10%, vamos a tener 2000 peticiones al mismo tiempo. Se hace necesario un balanceador de carga que reciba las peticiones y las distribuya, como la VPC se maneja entre dos regiones, se tiene un balanceador de carga para la región (us-west1) y otro para la región(us-central1), luego se define una zona A que contiene el grupo de instancias, estas instancias están Docker izadas, haciendo uso de Docker Swarm, dentro del grupo de instancias, se encuentra el conjunto de administradores, que consiste en dos administradores que se encargan de manejar las peticiones recibidas, una vez que se reciban, pasan al balanceador de carga interno que distribuye las peticiones al conjunto de trabajadores, se tiene un total de 10 trabajadores, ya que mediante la ley de Little se determinó:

$L = \lambda * W$

$10 = 200 * W$

$W = 10/200$

$W = 0{,}05\ s$

Si tengo 10 trabajadores y cada uno recibe 200 peticiones por segundo, obtengo un tiempo de 0,05 segundos.

Después tenemos la base de datos que es MySQL, es una base de datos relacional, la cual se replica a la otra región, permitiendo la alta disponibilidad en la capa de base de datos.

Similar a lo que sucede en la capa de bases de datos, sucede en la capa de persistencia de datos, que replica el Cloud Storage a la otra región, en el Cloud Storage se manejan los archivos y se garantiza la disponibilidad dentro de esta capa.

El diseño de la solución está basado en patrones de escalabilidad y buenas prácticas, como patrón de computación distribuida, el cual distribuye las solicitudes entre el conjunto de trabajadores, ayudando a proporcionar una escalabilidad de carga optimizada, el patrón de computación paralela que ayuda a distribuir la carga de trabajo a través de la ejecución en paralelo, distribución de carga de trabajo que facilita la escalabilidad de carga del sistema en general, uso de patrón de escalabilidad de la base de datos, con el almacenamiento en caché y la replicación, al igual que los patrones, es importante hacer uso de buenas prácticas como una solicitud independiente de la otra, reducir la adherencia de la sesión para ayudar a los balanceadores de carga, separar la aplicación por servicios ubicados en diferentes servidores (escalabilidad funcional)

Referencias bibliográficas:

1. ¿Qué es alta disponibilidad? Retomado de: **https://www.blockbit.com/es/blog/que-es-alta-disponibilidad/**

2. **¿**Qué es HA – High Availability? Retomado de. **https://www.hostingred.com/servidores/alta-disponibilidad-informacion/**

3. Rendimiento Web. Retomado de: **https://developer.mozilla.org/es/docs/Learn/Performance**

4. Seguridad web: revisa los factores claves para tener un sitio web seguro y sólido. Retomado de: **https://rockcontent.com/es/blog/seguridad-web/**

5. Aseguramiento de la calidad en el diseño del software, Jaime Eduardo Marulanda López, Universidad EAFIT, Medellín, 2014. Retomado de: **https://core.ac.uk/download/pdf/47246132.pdf**