



UNIVERSIDAD TÉCNICA
FEDERICO SANTA MARÍA

DEPARTAMENTO DE ELECTROTECNIA E
INFORMÁTICA INGENIERÍA EN INFORMÁTICA

Diseño de infraestructura en la nube para sistema de inventario.

Renato Plaza Diaz

renato.plaza@usm.cl

Gonzalo Mendoza Cárdenas
Profesor Guía

Resumen: Este proyecto se enfoca en el diseño e implementación de una infraestructura en la nube para soportar un sistema de gestión de inventario en la bodega central de la Municipalidad de Viña del Mar, optimizando la distribución de insumos hacia los comedores solidarios. El principal problema identificado es la falta de una infraestructura tecnológica adecuada que permita un control eficiente y en tiempo real sobre las existencias y fechas de caducidad de los productos. La solución propuesta consiste en crear una infraestructura escalable y segura basada en servicios de computación en la nube, que integre un backend y frontend trabajando de manera conjunta. El diseño incluye la configuración de redes privadas virtuales, la implementación de máquinas virtuales para alojar los sistemas, y la integración de bases de datos y almacenamiento en la nube. Se realizaron pruebas prácticas en un entorno controlado para validar la infraestructura, demostrando su capacidad para soportar las operaciones del sistema de inventario, mejorar la visibilidad y control del stock, y asegurar la eficiencia en la gestión de entradas y salidas de productos. Esto ha tenido un impacto positivo en la administración interna de la bodega y los comedores solidarios.

Conceptos Clave: Infraestructura, cloud, cloud-computing.

1. Introducción

1.1 Contexto del problema

El problema central que enfrentan los comedores solidarios de la municipalidad de Viña del Mar radica en la falta de una infraestructura tecnológica adecuada para gestionar los inventarios de productos y alimentos de manera eficiente. Actualmente, no existe una plataforma centralizada que permita controlar en tiempo real el stock de los insumos disponibles en la bodega central y su distribución hacia los comedores. Esta carencia de infraestructura aparte de generar ineficiencias operativas también complica la planificación y el manejo diario de los recursos, afectando la capacidad de los comedores para organizarse de forma óptima y prevenir la pérdida de productos.

Uno de los desafíos más significativos es la dificultad para garantizar que los productos ingresados en la bodega coincidan con los efectivamente entregados a cada comedor. Sin una infraestructura sólida y un sistema digital integrado, es casi imposible obtener datos precisos sobre las cantidades de productos que se despachan y reciben, lo que genera discrepancias en los inventarios y problemas de transparencia en la cadena de distribución. Esto repercute negativamente en la confianza del proceso y puede llevar a la descoordinación entre los encargados de la bodega y los



comedores solidarios.

Asimismo, la falta de digitalización y de una infraestructura tecnológica robusta aumenta el riesgo de errores humanos y fallas técnicas, lo que a su vez dificulta el acceso a información clave en momentos críticos. La ausencia de sistemas automatizados pone en peligro la integridad de los datos y la capacidad de la municipalidad para mantener un control eficaz sobre los productos, lo cual es esencial para asegurar la continuidad de los servicios que brindan los comedores.

1.2 Justificación del proyecto

La justificación del proyecto "Stocknow" radica en la necesidad de contar con una infraestructura moderna, escalable y segura que permita gestionar eficientemente los inventarios de los comedores solidarios de Viña del Mar. Al optar por una solución basada en la nube, se garantiza una serie de ventajas clave que hacen que este enfoque sea el más adecuado para los objetivos del sistema.

Implementar una infraestructura en la nube ofrece escalabilidad inmediata, lo que permite ajustar los recursos según las necesidades del proyecto. Dado que el número de comedores o el volumen de alimentos gestionados puede variar con el tiempo, la nube proporciona la flexibilidad necesaria para expandir o reducir el sistema sin interrupciones. Esto asegura que el sistema pueda crecer conforme a las demandas operativas, evitando limitaciones en capacidad o rendimiento.

Además, la seguridad es un pilar fundamental de la infraestructura en la nube. A través de servicios como la gestión de identidades y accesos (IAM) y sistemas de protección avanzados, se garantiza que los datos sensibles del inventario estén resguardados frente a accesos no autorizados y potenciales amenazas. La nube ofrece un entorno más seguro, con actualizaciones automáticas de seguridad y estándares que cumplen con normativas internacionales.

Otro factor crucial es la alta disponibilidad que ofrece la nube, donde los servicios están diseñados para estar activos en todo momento, minimizando el riesgo de interrupciones. Esto es particularmente importante para el sistema de inventarios, que debe estar operativo en tiempo real para asegurar que la información del stock esté actualizada y disponible en cualquier momento. Los sistemas de monitoreo y alertas permiten detectar problemas de manera anticipada, lo que facilita una gestión proactiva de la infraestructura.

El almacenamiento en la nube también ofrece ventajas significativas. Permite la gestión eficiente de archivos, como imágenes y documentos relacionados con el inventario, sin preocuparse por la capacidad física de almacenamiento. Asimismo, asegura que los datos estén siempre accesibles desde cualquier dispositivo con conexión a internet, lo que facilita la operación tanto desde los comedores como desde la administración central.

Otro tema para destacar es la rentabilidad a largo plazo, al evitar grandes inversiones iniciales en hardware y mantenimiento. Con la facturación basada en el uso, solo se paga por los recursos que realmente se utilizan, optimizando los costos del proyecto y permitiendo una mejor asignación de recursos municipales.

1.3 Objetivos

Objetivo general.

Este proyecto tiene como objetivo resolver los problemas asociados con la gestión de inventarios en las bodegas destinadas a los comedores solidarios de la municipalidad de Viña del Mar, garantizando una organización adecuada de los productos, disminuyendo pérdidas y mejorando la distribución hacia los beneficiarios mediante la sincronización de información en tiempo real. Esto se logrará diseñando la infraestructura en la nube para el sistema de inventario "Stocknow".



Objetivos Específicos.

Los objetivos específicos del proyecto "Stocknow" se centran en la configuración y puesta en marcha de una infraestructura en la nube que permita la correcta operación del sistema de inventario para los comedores solidarios de Viña del Mar. Cada uno de estos objetivos está orientado a garantizar que el sistema funcione de manera segura, eficiente y escalable, proporcionando a la municipalidad las herramientas necesarias para gestionar de forma óptima los recursos de los comedores.

Los objetivos específicos son los siguientes:

- **Configurar un entorno seguro y escalable para la operación del sistema de inventario:** Establecer una infraestructura que permita el aislamiento y control de los recursos, garantizando un entorno protegido para el manejo de la información.
- **Implementar la capacidad de procesamiento y presentación de datos para los componentes del sistema:** Desarrollar una solución que permita la ejecución de los procesos del sistema y la interacción con los usuarios, asegurando un rendimiento adecuado para soportar las demandas operativas.
- **Establecer un sistema de gestión de identidades y permisos:** Diseñar un mecanismo que permita asignar y controlar los niveles de acceso a los recursos, en función de los roles y responsabilidades de los usuarios.
- **Supervisar y mantener la disponibilidad de los datos del sistema:** Implementar procesos que garanticen la accesibilidad, integridad y continuidad de los datos relacionados con el inventario.
- **Configurar un espacio para el almacenamiento de información y archivos relacionados con el inventario:** Diseñar una solución que permita la organización y acceso eficiente a los recursos digitales requeridos por el sistema.

1.4 Solución propuesta.

La solución propuesta para el sistema de inventario "Stocknow" se centra en el diseño e implementación de una infraestructura en la nube que permita abordar de manera efectiva los problemas actuales de gestión y control de inventarios en los comedores solidarios de Viña del Mar. Más allá de resolver las dificultades operativas, "Stocknow" se plantea como una herramienta integral que facilitará una planificación más precisa y una mayor transparencia en todo el proceso, mejorando la eficiencia de los comedores y, en última instancia, contribuyendo al bienestar de las personas que dependen de estos servicios.

La propuesta central es la creación de una infraestructura en la nube que sea segura, escalable y eficiente, lo que permitirá gestionar el sistema de inventarios de manera óptima a largo plazo. Este diseño en la nube incluirá componentes clave:

- **Proyecto en Plataforma Cloud:** El sistema estará alojado en una plataforma cloud, lo que garantiza la flexibilidad y escalabilidad necesarias para ajustarse a las demandas de los comedores solidarios.
- **Configuración de máquinas virtuales (VM).**
El proyecto "Stocknow" incluye dos aplicaciones desarrolladas internamente: un sistema Backend y un sistema Frontend que abarca tanto una plataforma web como una aplicación móvil. Como parte de la solución, se configurarán máquinas virtuales (VM) para alojar estos sistemas y servicios, permitiendo su correcto funcionamiento y disponibilidad para los usuarios.

1.5 Metodología.

Optar por la metodología ágil Scrum tiene un impacto moderado en el desarrollo del sistema backend, influyendo principalmente en la priorización de funcionalidades clave para obtener un Producto Mínimo Viable (MVP) en cada iteración. Scrum es un marco de trabajo que permite gestionar proyectos de manera ágil, dividiendo el desarrollo en sprints, donde se entregan resultados funcionales al finalizar cada ciclo. En este caso, se busca ofrecer a los usuarios un sistema operativo con una interfaz visualmente completa, aunque algunas secciones aún no estén habilitadas, permitiéndoles visualizar el alcance y potencial del producto final.

En cada sprint, se establecen objetivos específicos:

Sprint 1: Se desarrolla la sincronización del inventario entre las plataformas web y móviles, garantizando la coherencia de los datos. Aunque solo la sección de inventario está operativa, se despliega toda la interfaz con las demás secciones deshabilitadas, brindando a los usuarios una visión clara de la estructura del sistema.

Sprint 2: Se implementa el sistema de autenticación y gestión de roles, junto con la funcionalidad para planificar la distribución de productos hacia los comedores solidarios. Durante esta fase, se refina la experiencia del usuario, manteniendo la consistencia y accesibilidad de la plataforma. En esta fase ya se puede acceder a la API mediante peticiones HTTP gracias a la infraestructura montada en GCP.

Sprint 3: Se integra el sistema de reportes y alertas para productos cercanos a su fecha de caducidad, mejorando el control sobre los insumos perecibles. En tanto a la infraestructura se incorpora el balanceador de carga de GCP.

Es importante señalar que, en cada sprint, además de alcanzar los objetivos específicos, se debe presentar una interfaz completa y funcional. Esto asegura que el Producto Mínimo ofrezca una experiencia de usuario sólida.

2. Marco Teórico

Este capítulo aborda los fundamentos del Cloud Computing, destacando cómo esta tecnología ha transformado la gestión de recursos tecnológicos mediante el acceso a servicios escalables y flexibles a través de internet. Se describen los principales modelos de servicio en la nube (IaaS, PaaS y SaaS), resaltando la adopción del modelo SaaS en el proyecto por su capacidad de ofrecer software bajo demanda sin necesidad de infraestructura local. Además, se presentan las principales ventajas del Cloud Computing, como escalabilidad, reducción de costos, alta disponibilidad y seguridad, subrayando su impacto positivo en soluciones accesibles y eficientes, como sistemas de inventario.

2.1 Fundamentos del Cloud Computing.

El Cloud Computing, o computación en la nube, ha transformado la manera en que las empresas y organizaciones gestionan sus recursos tecnológicos, permitiendo acceso a servicios informáticos a través de internet sin necesidad de contar con infraestructura física propia. La nube ofrece una plataforma escalable, flexible y eficiente que permite ejecutar aplicaciones, almacenar datos, y procesar información desde cualquier lugar y en cualquier momento.

Cloud Computing se refiere a la entrega de servicios informáticos a través de la nube, que incluye servidores, almacenamiento, bases de datos, redes, software, análisis y más. Según el Instituto Nacional de Estándares y Tecnología (NIST), el Cloud Computing es un modelo que permite el



acceso ubicuo y bajo demanda a un conjunto compartido de recursos computacionales configurables, como redes, servidores, almacenamiento, aplicaciones y servicios. [1]

El mercado global de Cloud Computing ha experimentado un crecimiento significativo en los últimos años. En el tercer trimestre de 2024, Amazon Web Services lideró con una cuota de mercado del 31%, seguido por Microsoft Azure con el 29% y Google Cloud con el 8%. [2]

El concepto de computación en la nube se originó en la década de 1960, cuando John McCarthy propuso que la informática podría ser ofrecida como un servicio público. [3]

Sin embargo, su implementación moderna se consolidó a mediados de los 2000 con la introducción de servicios como AWS en 2006. Inicialmente, la nube se utilizaba principalmente para almacenamiento remoto, pero ha evolucionado para incluir servicios avanzados como inteligencia artificial, aprendizaje automático y análisis de datos.

2.2 Modelo de servicio

Cloud Computing se ofrece en tres modelos principales de servicio, cada uno diseñado para diferentes niveles de control y responsabilidad sobre los recursos:

- **Infraestructura como Servicio (IaaS):** Entrega aplicaciones completas a través de la nube sin necesidad de instalación local. Ej: Amazon web services, Google Cloud Platform.
- **Plataforma como Servicio (PaaS):** Ofrece plataformas para desarrollar y ejecutar aplicaciones. Ej: Google App Engine, Heroku
- **Software como Servicio (SaaS):** Proporciona recursos informáticos básicos como servidores y almacenamiento. Ej: Uber, Airbnb. Nuestro proyecto se basa en este modelo de servicio.

2.3 Ventajas del Cloud Computing.

El Cloud Computing ofrece múltiples ventajas que lo convierten en una opción atractiva para organizaciones de todos los tamaños. Las principales ventajas incluyen:

- **Escalabilidad:** Permite aumentar o reducir los recursos en función de las necesidades del negocio, lo que asegura que las organizaciones solo paguen por lo que realmente utilizan.
- **Flexibilidad:** Ofrece la posibilidad de acceder a servicios desde cualquier lugar y dispositivo con conexión a internet, mejorando la colaboración y productividad.
- **Reducción de costos:** Elimina la necesidad de invertir en infraestructura propia, ya que los servicios se pagan como un gasto operativo basado en el uso.
- **Alta disponibilidad:** Los proveedores garantizan tiempo de actividad elevado mediante mecanismos de redundancia y recuperación ante desastres.
- **Seguridad:** Los datos son protegidos mediante cifrado, controles de acceso y auditorías constantes, cumpliendo con estándares internacionales de seguridad.

El modelo SaaS ha sido adoptado por una gran variedad de sectores debido a su flexibilidad, escalabilidad y capacidad de integración. Este enfoque permite a las organizaciones utilizar software bajo demanda, eliminando la necesidad de instalar, mantener y actualizar aplicaciones en servidores locales. En lugar de ello, los usuarios acceden a las aplicaciones a través de un navegador web o una interfaz remota. Este modelo es ideal para soluciones que requieren accesibilidad desde diferentes ubicaciones, como sistemas de inventario.

Empresas de todos los tamaños, desde startups hasta corporaciones multinacionales, han adoptado SaaS para gestionar sus operaciones de manera más eficiente. [5]

2.4 Desventajas del Cloud Computing

Aunque el Cloud Computing ofrece numerosos beneficios, también presenta algunos desafíos que deben ser considerados al adoptar esta tecnología:

- **Dependencia del proveedor:** Al usar servicios en la nube, las empresas dependen de un proveedor específico para la continuidad y calidad de los servicios. Cambiar de proveedor puede ser complejo y costoso.
- **Riesgos de seguridad:** Los datos almacenados en la nube están expuestos a riesgos como ciberataques o accesos no autorizados.
- **Costos ocultos:** Los servicios premium pueden incrementar significativamente los gastos.
- **Requerimientos de conectividad:** El acceso a la nube depende de una conexión a internet estable. En lugares con conectividad limitada, esto puede convertirse en un obstáculo.
- **Problemas de cumplimiento:** Las regulaciones locales o internacionales sobre el almacenamiento y manejo de datos pueden complicar el uso de ciertos servicios en la nube.

3. Estado del arte.

En la actualidad, las infraestructuras de software basadas en la nube se han vuelto fundamentales para garantizar la escalabilidad, disponibilidad y eficiencia de los sistemas SaaS. A diferencia de una infraestructura *on-premise*, que requiere una inversión significativa en hardware, mantenimiento, espacio físico y personal especializado, las plataformas en la nube como Google Cloud, AWS y Azure ofrecen una solución más flexible y rentable. [6] Estas plataformas permiten diseñar infraestructuras que se ajustan dinámicamente a las necesidades de cada sistema, asegurando su capacidad para crecer y adaptarse según la demanda. Además, ofrecen servicios avanzados como redes privadas virtuales (VPC), bases de datos gestionadas y herramientas de monitoreo que detectan y resuelven problemas antes de que impacten a los usuarios, todo sin las limitaciones y costos operativos de otra solución.

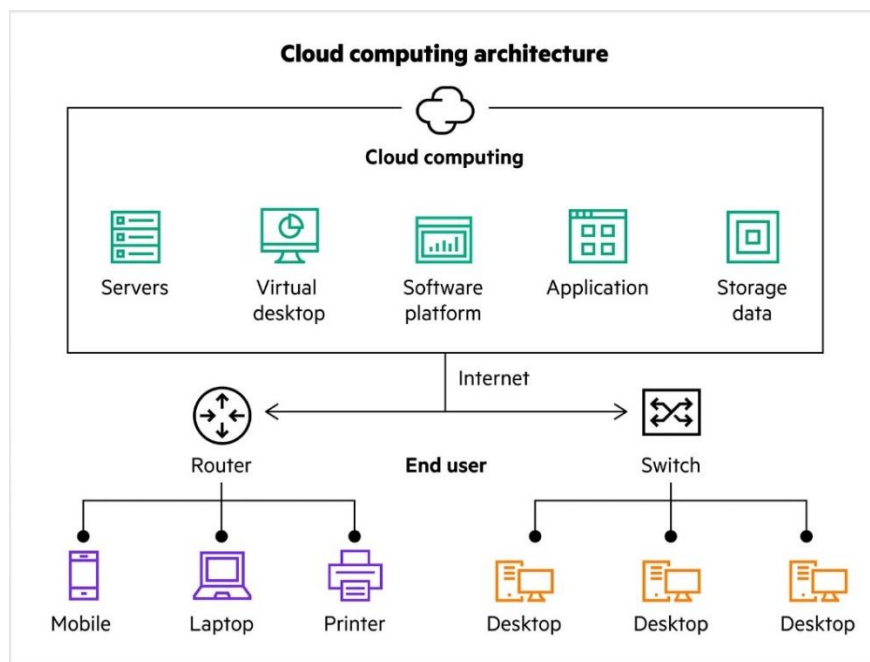


Figura 1 Diagrama Cloud Computing (Elaboración Hewlett Packard Enterprise)



“De acuerdo con un estudio realizado por la consultora Statista, durante 2023, el mercado de servicios *Cloud Computing* pública (ejemplo: los ofrecidos por proveedores como AWS, Google o Microsoft Azure), crecerán un 21% respecto del 2022. Se estima, incluso, que para el próximo año, más de un 50% de las nuevas implementaciones de infraestructura tecnológica en empresas, estará en la «nube pública» y no en *data centers* enfocados a la «nube privada».” [6]

Este crecimiento refleja la creciente adopción de soluciones en la nube por parte de empresas de todos los tamaños, impulsada por la necesidad de mayor flexibilidad, escalabilidad y eficiencia operativa.

Ventajas de las Infraestructuras en la Nube

- **Escalabilidad y Flexibilidad:** Las plataformas en la nube permiten a las organizaciones escalar recursos según la demanda, adaptándose rápidamente a cambios en las cargas de trabajo sin la necesidad de inversiones en infraestructura física.
- **Reducción de Costos:** Al adoptar soluciones en la nube, las empresas pueden reducir gastos operativos y de capital, pagando solo por los recursos que utilizan y eliminando costos asociados con el mantenimiento de hardware y actualizaciones.
- **Alta Disponibilidad:** Los proveedores de servicios en la nube ofrecen acuerdos de nivel de servicio (SLA) que garantizan altos niveles de disponibilidad y redundancia, asegurando el funcionamiento continuo de las aplicaciones críticas.

Las plataformas en la nube ofrecen servicios avanzados diseñados para optimizar la eficiencia y la seguridad de las operaciones. Entre estos servicios destacan las redes privadas virtuales (VPC), que permiten crear entornos de red aislados y seguros dentro de la nube, otorgando un mayor control sobre las aplicaciones. Además, las bases de datos gestionadas, como Amazon RDS y Azure SQL Database, simplifican la configuración, operación y escalado, liberando a las empresas de tareas administrativas complejas. Asimismo, herramientas como Google Cloud Monitoring y AWS CloudWatch posibilitan la supervisión continua del rendimiento de las aplicaciones, detectando problemas de manera proactiva y asegurando una respuesta rápida para minimizar el impacto en los usuarios finales.

La **seguridad** es otro pilar fundamental, con tecnologías como **IAM** para gestionar el acceso y garantizar que la información crítica del sistema esté protegida. En este contexto, los sistemas informáticos se benefician enormemente de estas soluciones en la nube, asegurando una infraestructura sólida, escalable y segura que puede soportar las necesidades requeridas.[7]

4. Comparación de Infraestructuras

En este capítulo se comparará la infraestructura actual basada en Google Cloud Platform con una alternativa en AWS, analizando sus diferencias, ventajas y desventajas en diversos aspectos. También incluirá una sección que explore posibles escenarios de cambio o expansión de la infraestructura, como la migración a otro proveedor o la adición de nuevos componentes y servicios.

4.1 Características generales.

En tanto a las características generales de estos proveedores de servicios en la nube, ambos son uno de los principales en el ámbito, pero cada uno tiene un enfoque único y características distintivas: Google Cloud Platform se destaca por ser pionero en tecnologías de contenedores como Kubernetes, lo que lo hace ideal para proyectos que buscan innovación tecnológica.

También cuenta con una red global de alto rendimiento que forma parte de la misma infraestructura que servicios como Gmail, Youtube, Google search, etc.

Amazon Web Services fue de los primeros en popularizar los servicios en la nube, y ofrece la gama más amplia de servicios, con más de 200 opciones de estos, que abarcan desde almacenamiento hasta inteligencia artificial.

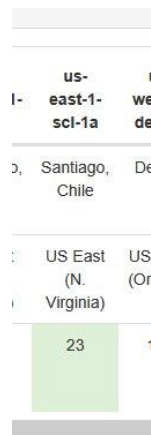
Cabe destacar que AWS tiene una comunidad más amplia y activa lo que facilita encontrar recursos, soporta y herramientas de terceros compatible con el sistema a desarrollar. [8]

4.2 Comparación Técnica.

- GCP destaca por su red definida por software (SDN) y la capacidad de conectar recursos globalmente a través de su infraestructura privada. Esto resulta en menores latencias y mejor rendimiento en aplicaciones distribuidas.[9]
- AWS, aunque ofrece una red global robusta, la latencia puede ser mayor en algunos escenarios debido a su dependencia de redes públicas para ciertas conexiones. No obstante, AWS compensa con un mayor número de regiones y zonas de disponibilidad.[10]

Durante las pruebas realizadas, se midió la latencia promedio para cada plataforma, ambas apuntando a un servidor ubicado en Santiago de Chile. Los resultados obtenidos fueron muy similares.

- GCP: Media de 14 ms de latencia. (Obtenido mediante ping al servidor)
- AWS: Media de 23 ms de latencia, obtenida mediante una herramienta de terceros (aws-latency-test.com).



Captura prueba latencia AWS. (aws-latency-test.com).)

A pesar de que GCP, al estar construido sobre la misma infraestructura de red que utilizan servicios como Google Search y YouTube. Esto indica que ambos proveedores ofrecen una red de alta calidad en el mercado chileno, con tiempos de respuesta suficientemente bajos para soportar aplicaciones críticas que dependen de conexiones rápidas.

También se aplicó una comparación técnica a nuestra misma infraestructura, pero considerando el uso del servicio de base de datos SQL de Google.

A pesar de que GCP ofrece un servicio administrado de bases de datos SQL que proporciona características avanzadas como gestión automática de copias de seguridad, alta disponibilidad y escalabilidad, decidimos no implementarlo en el sistema "Stocknow" por dos razones principales: **costo y rendimiento**.

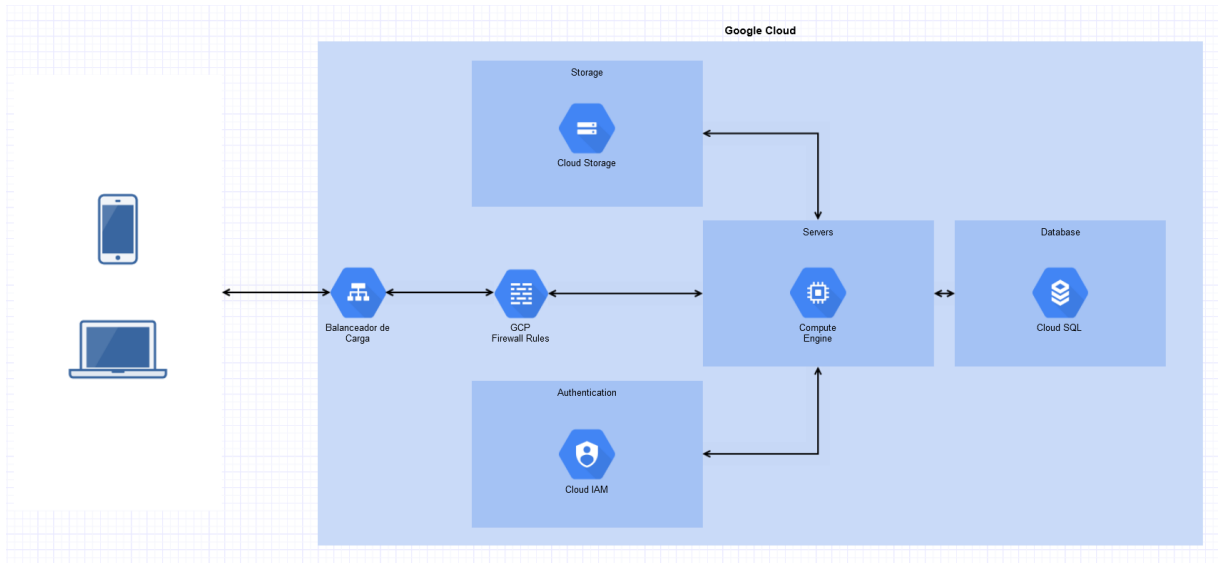


Figura 2 Diagrama GCP alternativo. (Elaboración Propia)

4.2.1. Costos elevados.

El servicio de Google Cloud SQL tiene un costo significativamente mayor en comparación con la implementación de una base de datos directamente en la máquina virtual del backend. Según nuestras estimaciones, el uso de Cloud SQL implicaría un gasto aproximado de **3 dólares diarios** solo por el servicio base, lo que representa casi **tres veces el costo actual** de mantener la base de datos en la VM. Este sobre costo no resulta justificado en el contexto de nuestro proyecto, especialmente considerando que la infraestructura actual cumple con los requisitos operativos de manera eficiente y a un costo considerablemente menor.

Cost Estimate Summary

As of 16 ene 2025 • 12:44 a.m.

Prices in USD

STORAGE	\$85.00
Cloud Storage	\$85.00
Service type Cloud Storage	
Data Transfer within Google Cloud 500 GiB	\$70.00
Total amount of storage 500 GiB	\$15.00
Location type Region	N/A
Location Santiago (southamerica-west1)	N/A
Storage class Standard Storage	N/A
Source region Latin America	N/A
Destination region Latin America	N/A

Total estimated cost

\$85.00 / mo

The estimated fees provided by Google Cloud Pricing Calculator are for discussion purposes only and are not binding on either you or Google. Your actual fees may be higher or lower than the estimate.

Estimacion de costos Google Cloud Storage. (Google Cloud Pricing)



4.2.2. Problemas de latencia y rendimiento.

Al estar en un servicio independiente de la máquina virtual del backend, la comunicación con una base de datos en Cloud SQL podría introducir problemas de latencia que afectarían el rendimiento del sistema. Durante nuestras pruebas, simulamos una configuración similar a Cloud SQL utilizando una base de datos externa a la VM del backend, y los resultados mostraron que las solicitudes eran **aproximadamente 1 segundo más lentas** en comparación con la configuración actual, donde la base de datos está localizada directamente en la misma máquina que el backend.

Cabe destacar que unos de los objetivos del proyecto Stocknow es mantener una comunicación en tiempo real de los dispositivos con el servidor en todo momento, por lo que demoras de ese tipo afectan significativamente al sistema.

4.3 Costos.

Ambos proveedores emplean modelos de facturación flexibles, pero con enfoques diferentes que pueden influir significativamente en la elección según las necesidades del proyecto.

- **Máquinas Virtuales:** Las instancias de Compute Engine en GCP tienden a ser ligeramente más económicas que las instancias EC2 de AWS para configuraciones similares en términos de CPU y memoria. Sin embargo, AWS ofrece una mayor variedad de tipos de instancias, lo que puede ser una ventaja para aplicaciones con requisitos muy específicos.
- **Almacenamiento:** Los costos de almacenamiento en GCP y AWS son comparables, pero GCP incluye características como redundancia regional sin costos adicionales, mientras que en AWS se cobran como opciones separadas.
- **Tráfico de Red:** Ambos proveedores aplican cargos por tráfico de salida, pero AWS tiende a ser más costoso en este aspecto, especialmente en transferencias interregionales.

Se realizó una estimación de costos en AWS Pricing Calculator para estimar cuánto costaría implementar la infraestructura actualmente montada en GCP.

En la siguiente imagen se evidencia la estimación de costos realizada para la misma infraestructura (mismas máquinas), pero, ya que en Santiago, Chile, no existen las mismas máquinas se realizó la estimación con los servidores de Sao Paulo, Brasil.



Contact your AWS representative: [Contact Sales](#)

Export Date: 11/26/2024

Language: English

[Estimate url](#)

Estimate summary

Upfront cost	Monthly cost	Total 12 months cost
0.00 USD	73.35 USD	\$80.20 USD Includes upfront cost

Detailed Estimate

Name	Group	Region	Upfront cost	Monthly cost
Amazon EC2	-	South America (Sao Paulo)	0.00 USD	44.09 USD
Status	-			
Description:	Instancias EC2			
Config summary	Tenancy (Shared Instances), Operating system (Linux), Workload (Consistent, Number of instances: 2), Advance EC2 instance (t3a.small), Pricing strategy (On-Demand Utilization: 100 %Utilized/Month), Enable monitoring (disabled), DT Inbound: Not selected (0 TB per month), DT Outbound: Not selected (0 TB per month), DT Intra-Region: (0 TB per month)			
Name	Group	Region	Upfront cost	Monthly cost
Amazon Simple Storage Service (S3)	-	South America (Sao Paulo)	0.00 USD	1.15 USD
Status	-			
Description:	almacenamiento s3			
Config summary	S3 Standard storage (15 GB per month), PUT, COPY, POST, LIST requests to S3 Standard (100), GET, SELECT, and all other requests from S3 Standard (1000), Data returned by S3 Select (100 GB per month), Data scanned by S3 Select (100 GB per month)			
Name	Group	Region	Upfront cost	Monthly cost
Elastic Load Balancing	-	South America (Sao Paulo)	0.00 USD	28.11 USD
Status	-			
Description:	Balanceador de carga			
Config summary	Number of Availability Zones that Gateway Load Balancer is deployed to (1), Number of Gateway Load Balancer Endpoints (1), Total processed bytes (0.5 GB per hour)			

Acknowledgement

AWS Pricing Calculator provides only an estimate of your AWS fees and doesn't include any taxes that might apply. Your actual fees depend on a variety of factors, including your actual usage of AWS services. [Learn more](#)

Figura 3 Estimación de costos 1 (AWS Pricing calculator)

AWS nos presenta un costo estimado mensual de 73.35 **USD**.

Realizando la misma estimación de costos en los servidores de Santiago, Chile.



Contact your AWS representative: [Contact Sales](#)

Export Date: 11/27/2024

Language: English

[Estimate url](#)

Estimate summary				
Upfront cost 0.00 USD		Monthly cost 74.30 USD	Total 12 months cost \$91.60 USD Includes upfront cost	

Detailed Estimate

Name	Group	Region	Upfront cost	Monthly cost
Amazon EC2	-	Chile (Santiago)	0.00 USD	51.54 USD
Status	-			
Description:	Instancias EC2 Chile			
Config summary	Tenancy (Shared Instances), Operating system (Linux), Workload (Consistent, Number of instances: 2), Advance EC2 instance (13 medium), Pricing strategy (3yr No Upfront), DT Inbound: Not selected (0 TB per month), DT Outbound: Not selected (0 TB per month), DT Intra-Region: (0 TB per month)			

Name	Group	Region	Upfront cost	Monthly cost
Amazon Simple Storage Service (S3)	-	US East (N. Virginia)	0.00 USD	0.85 USD
Status	-			
Description:	Almacenamiento s3			
Config summary	S3 Standard storage (25 GB per month), PUT, COPY, POST, LIST requests to S3 Standard (100), GET, SELECT, and all other requests from S3 Standard (1000), Data returned by S3 Select (100 GB per month), Data scanned by S3 Select (100 GB per month)			

Name	Group	Region	Upfront cost	Monthly cost
Elastic Load Balancing	-	US East (N. Virginia)	0.00 USD	21.91 USD
Status	-			
Description:	Balanceador Carga			
Config summary	Number of Availability Zones that Gateway Load Balancer is deployed to (1), Number of Gateway Load Balancer Endpoints (1), Total processed bytes (1 GB per hour)			

Acknowledgement

AWS Pricing Calculator provides only an estimate of your AWS fees and doesn't include any taxes that might apply. Your actual fees depend on a variety of factors, including your actual usage of AWS services. [Learn more](#)

Figura 4 Estimacion de costos 2 (AWS pricing calculator)

Con la configuración más parecida a GCP, excepto por las configuraciones de hardware de las máquinas virtuales nos da un valor mensual de 74.3 USD.

Diferencias:

- EC2 (más caro)
- Load balancing (más barato)



Figura 5 Consumo de la inf. actual 7 días (GCP billing)

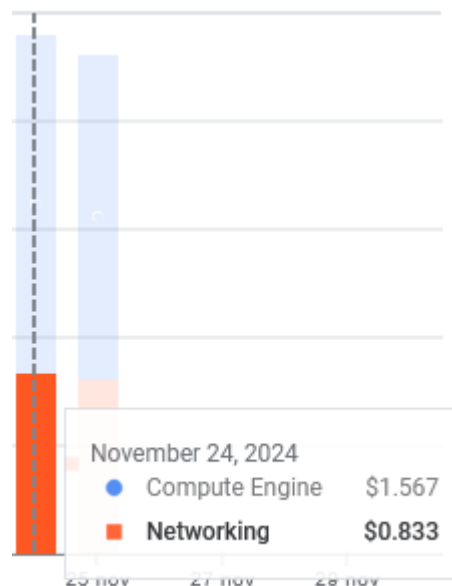


Figura 6 Detalle consumo inf. Actual (GCP billing)

Se evidencia que, con la infraestructura ya desplegada en Google Cloud Platform (GCP), el costo diario asciende a 2.4 USD. Este valor refleja un aproximado a lo que fue específicamente el gasto generado durante el día de la Feria de Software 2024, (15 de noviembre), en el cual se presentó el prototipo del sistema Stocknow y se realizaron un gran número de transacciones, representando un escenario de alto consumo para la aplicación. Se destaca la eficiencia de GCP en términos de manejo de recursos durante picos de demanda, también demuestra que incluso en días de alta actividad, el costo diario se mantiene significativamente económico, reforzando su viabilidad como solución rentable para el proyecto.



En contraste, con las pruebas realizadas directamente en GCP, el costo mensual proyectado es de **71.4USD**, lo que representa un ahorro significativo en el contexto de proyectos con presupuestos ajustados. Aunque la diferencia puede parecer pequeña, este ahorro mensual se acumula con el tiempo y demuestra que GCP es competitivo en términos de rendimiento y flexibilidad, siendo así mismo la opción elegida para la infraestructura actual de Stocknow.

5. Diseño de la infraestructura.

5.1. Introducción.

En este capítulo se detalla la infraestructura en la nube propuesta para el sistema 'Stocknow', incluyendo los recursos y servicios necesarios para garantizar su operatividad. Se presentan diagramas que representan la arquitectura de la nube, flujos de datos y despliegue, además de justificar las decisiones de diseño adoptadas

5.2. Justificación de las Decisiones de Diseño.

La selección de esta infraestructura y de Google Cloud Platform como proveedor de servicios en la nube se basó, en primer lugar, en mi experiencia previa y nivel de familiaridad con esta tecnología. Con más de medio año trabajando con GCP en mi entorno laboral, tengo un conocimiento sólido de sus herramientas, servicios y mejores prácticas, lo que garantiza un desarrollo ágil y eficiente para este proyecto. Al elegir GCP, puedo aprovechar al máximo este conocimiento para diseñar y configurar la infraestructura de manera rápida y eficaz, minimizando tiempos de implementación y reduciendo el riesgo de errores técnicos.

La elección de esta infraestructura y GCP como proveedor de servicios en la nube se basa en un análisis de las necesidades del sistema "Stocknow" y las ventajas específicas que ofrece Google Cloud, las cuales se detallan en el capítulo 4.5. Este análisis permite justificar la selección al destacar características clave como escalabilidad, seguridad y herramientas avanzadas que se alinean con los objetivos del sistema.

Para la justificación de esta infraestructura se evaluaron criterios clave como escalabilidad, flexibilidad, alta disponibilidad, seguridad, y optimización de costos.

La infraestructura diseñada permite tanto escalabilidad vertical como horizontal, adaptándose a las necesidades operativas y de negocio a medida que estas evolucionan. Por ejemplo, las máquinas virtuales en **Compute Engine** pueden escalar verticalmente al incrementar su capacidad de procesamiento o memoria, y horizontalmente al replicarse para soportar un mayor número de usuarios simultáneos. Esto garantiza un rendimiento consistente y una experiencia de usuario ininterrumpida, incluso durante picos de demanda.

Además, la arquitectura modular, con servicios separados (frontend, backend, base de datos y almacenamiento), facilita la expansión del sistema sin afectar otros componentes. Este diseño asegura que el sistema puede evolucionar sin comprometer su estabilidad.

La implementación de un balanceador de carga asegura que el tráfico entrante se distribuya uniformemente entre las máquinas virtuales. Esto mejora la eficiencia del sistema y garantiza que, en caso de fallas en un componente, el tráfico sea redirigido automáticamente hacia los servicios operativos.

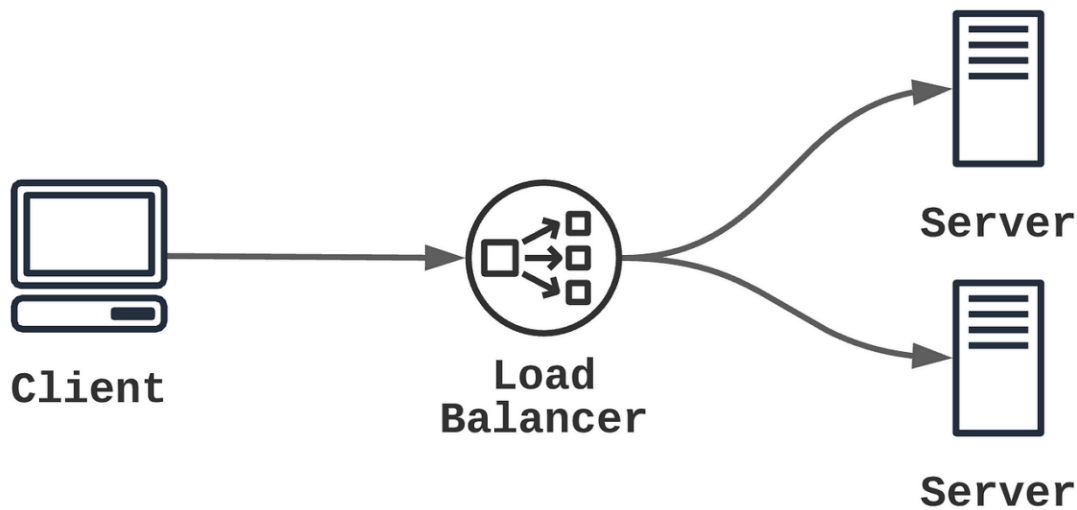


Diagrama Balanceador de carga (Elaboración Diego Esteban C ,médium.com)

Esta capacidad de recuperación rápida minimiza el impacto en los usuarios finales y asegura la continuidad de las operaciones.

Adicionalmente, la redundancia en los servicios críticos, como la base de datos y el almacenamiento, asegura que los datos y recursos estén disponibles en todo momento, incluso frente a interrupciones inesperadas.

En términos de seguridad, el uso de **GCP Firewall Rules** y **Cloud IAM** proporciona un nivel de protección robusto. Las reglas de firewall controlan el acceso a la infraestructura, permitiendo únicamente el tráfico autorizado, mientras que Cloud IAM gestiona los permisos y roles de los usuarios y aplicaciones, asegurando un acceso granular y restringido a los recursos. Este enfoque garantiza que solo usuarios y servicios autenticados puedan interactuar con el sistema, protegiendo tanto los datos sensibles como las operaciones críticas del sistema.

Google Cloud Platform presenta una estructura de facturación basada en el uso, lo que permite optimizar los costos operativos al pagar únicamente por los recursos consumidos. Esto elimina la necesidad de grandes inversiones iniciales en hardware y reduce el riesgo financiero asociado con la subutilización de recursos. Además, los descuentos automáticos por uso prolongado (sustained use discounts) hacen que GCP sea especialmente rentable para sistemas con carga de trabajo constante, como "Stocknow". En el capítulo 5 se realiza una comparación de infraestructura con otro servicio de Cloud Computing.

5.3. Descripción general de la infraestructura.

El sistema "Stocknow" está completamente desplegado en Google Cloud Platform y se basa en una arquitectura optimizada para escalabilidad, seguridad y alta disponibilidad. Se escogió esta plataforma, principalmente por la experiencia que se tenía previamente trabajando con este servicio, lo que disminuyó tiempos de aprendizaje de otras tecnologías como AWS y redujo el tiempo de errores técnicos.

A continuación, se presenta una visión general de los principales componentes y cómo interactúan entre sí para garantizar un rendimiento eficiente:

- **Instancia de Compute Engine:**
 - Comprende dos máquinas virtuales que tienen las siguientes características:
 - 2 núcleos compartidos, 1 núcleo de CPU compartido.
 - 2 GB de RAM.
 - 15 GB de almacenamiento en disco persistente / 10GB para frontend
 - Ubuntu 24.04 amd64.
 - Máquina **Frontend web**, encargado de gestionar las interacciones con los usuarios y servir la interfaz gráfica.
 - Máquina **Backend web** y la **Base de datos**, que procesan las solicitudes, gestionan la lógica del sistema y almacenan la información del inventario.
- **Balanceador de Carga:**
 - Reparte el tráfico entre las máquinas virtuales, asegurando una distribución uniforme y reduciendo la carga en cada servidor. Esto garantiza la disponibilidad continua del sistema incluso bajo alta demanda.
- **GCP Firewall Rules:**
 - Controla y restringe el tráfico hacia los recursos, asegurando que solo las solicitudes autorizadas puedan acceder al sistema, protegiendo los datos y servicios.
- **Cloud Storage:**
 - Utilizado para almacenar archivos relacionados con el inventario, como imágenes de productos y documentos. Este servicio asegura la durabilidad y accesibilidad de los datos desde cualquier ubicación.
- **Cloud IAM (Identity and Access Management):**
 - Proporciona una gestión centralizada de identidades y permisos, permitiendo asignar roles y restricciones de acceso según las necesidades de cada usuario o componente del sistema.

5.4. Diagramas de diseño para la solución actual.

5.4.1. Diagrama GCP

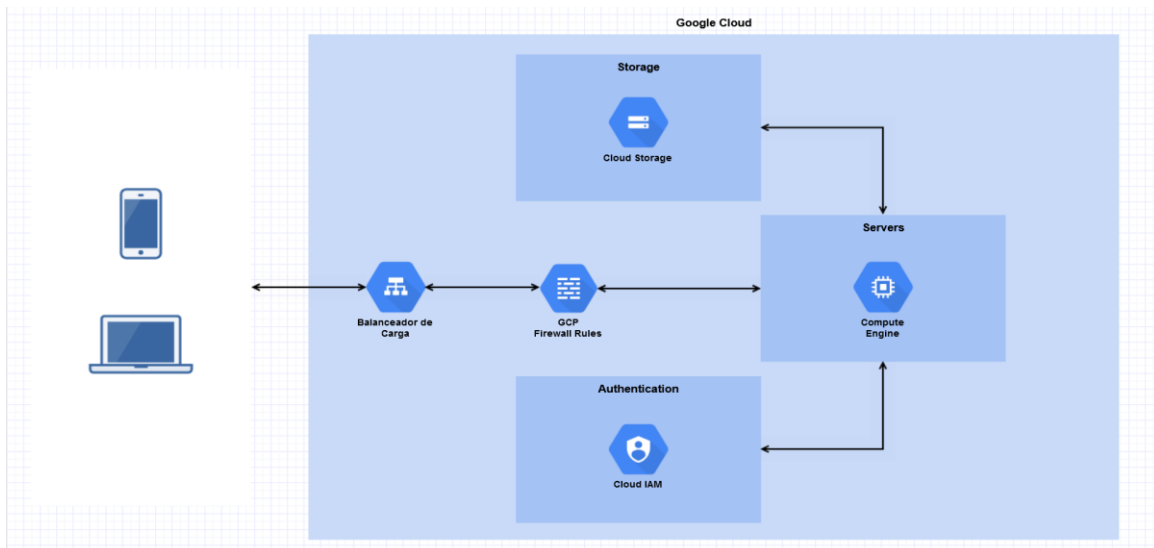


Figura 7 Diagrama GCP (Elaboración Propia)

El diagrama de la arquitectura muestra un balanceador de carga que distribuye el tráfico de manera eficiente entre dos máquinas virtuales, asegurando alta disponibilidad del sistema. Las reglas de firewall de GCP controlan y restringen el acceso a los recursos, limitando las conexiones desde internet. Cloud IAM gestiona los permisos y accesos entre los diferentes componentes, utilizando cuentas de servicio para garantizar la seguridad. Cloud Storage se utiliza para almacenar archivos relacionados con el inventario, mientras que las instancias de Compute Engine incluyen dos máquinas virtuales: una dedicada al frontend y otra que combina el backend con la base de datos.

5.4.2. Diagrama de subredes

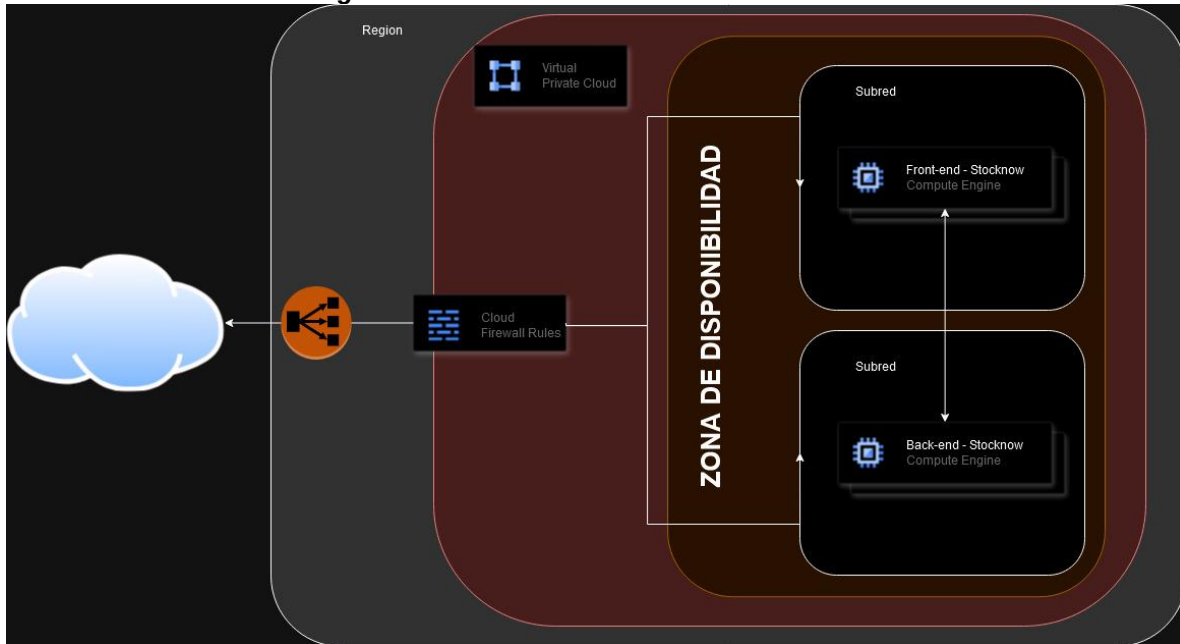


Figura 8 Diagrama de subredes (Elaboración Propia)

Ese diagrama representa una arquitectura de red basada en la nube utilizando subredes dentro de una Virtual Private Cloud (VPC). Es importante ya que presenta una segmentación de recursos. Indica región y disponibilidad, comunicación interna y demuestra que posee un acceso controlado.

5.5. Ventajas del servicio elegido.

GCP sobresale frente a otros proveedores, como AWS, gracias a su infraestructura de red global avanzada, que utiliza la misma red que respalda servicios como Google Search y YouTube. Esto garantiza una latencia baja y conexiones excepcionalmente rápidas en cualquier parte del mundo. Además, GCP ha liderado la innovación en tecnologías como Kubernetes y TensorFlow. Kubernetes es un sistema de orquestación de contenedores de código abierto que permite desplegar, escalar y gestionar aplicaciones en entornos distribuidos, ofreciendo una gran flexibilidad y eficiencia. Por su parte, TensorFlow es una biblioteca de aprendizaje automático de código abierto diseñada para desarrollar modelos de inteligencia artificial, optimizando proyectos que requieren procesamiento avanzado de datos. GCP también destaca por su modelo de facturación flexible, que incluye descuentos automáticos por uso prolongado, lo que lo convierte en una opción ideal para proyectos con presupuestos limitados y cargas de trabajo variables, sin necesidad de compromisos iniciales. [8]

Características	GCP	AWS
Red Global	Alta velocidad y baja latencia	Amplia cobertura, pero con mayor latencia en ciertos casos.
Costo-Eficiencia	Descuentos automáticos sin compromiso.	Requiere reservas para descuentos
Seguridad	Cifrado predeterminado en reposo y tránsito.	Seguridad robusta, pero más compleja de configurar.
Simplicidad de gestión.	Interfaz amigable y moderna.	Interfaz orientada a usuarios avanzados.
Innovación	Kubernetes, Machine Learning líderes.	Amplia oferta, pero menor optimizado en ciertos servicios, como por ejemplo contenedores.

Tabla 1 Comparación AWS – GCP (Elaboración Propia)

5.6. Conexión con los requisitos de Stocknow y con la lógica del negocio.

En este apartado se detalla cómo la infraestructura de GCP satisface los requisitos específicos del sistema "Stocknow" y se alinea con la lógica del negocio. Cada característica de la solución propuesta ha sido diseñada para abordar las necesidades particulares de los comedores solidarios y garantizar un funcionamiento eficiente, seguro y escalable. A continuación, se presentan los principales puntos de conexión entre la infraestructura y los objetivos del sistema:

GCP ofrece la capacidad de ajustar los recursos en tiempo real, permitiendo que el sistema maneje eficientemente las variaciones en el número de usuarios y solicitudes. Esta flexibilidad es especialmente útil para gestionar las fluctuaciones en el inventario y la demanda de los comedores.

La infraestructura global de GCP asegura que el sistema pueda ser accedido de manera confiable tanto por los operadores de la bodega central como por los encargados de los comedores solidarios. Esto permite que cualquier usuario con un dispositivo conectado a internet participe en la gestión del inventario y las operaciones.

Dado que el sistema gestiona información crítica sobre inventarios y cadenas de distribución, la seguridad es un aspecto prioritario. GCP proporciona herramientas avanzadas para proteger datos sensibles, combinando facilidad de implementación con altos estándares de protección.

La propuesta de infraestructura optimiza los costos operativos al emplear recursos basados en demanda, eliminando gastos innecesarios. Esto permite a la municipalidad administrar eficientemente su presupuesto mientras garantiza un servicio de alta calidad.



5.7. Desafíos técnicos y soluciones.

Latencia entre servicios

Uno de los principales desafíos en sistemas distribuidos es la latencia en la comunicación entre componentes, especialmente cuando el frontend, backend y la base de datos están alojados en diferentes máquinas virtuales dentro de la infraestructura. La latencia puede afectar significativamente la experiencia del usuario final al generar retrasos en el procesamiento de solicitudes.

Solución:

- **Proximidad geográfica:** Se seleccionaron zonas y regiones específicas de Google Cloud para alojar todos los recursos del sistema dentro de la misma región. Esto reduce la latencia en la comunicación entre máquinas virtuales, bases de datos y almacenamiento.
- **Optimización de redes internas:** El uso de redes privadas virtuales (VPC) con subredes optimizadas permite una comunicación rápida y directa entre los servicios, evitando el paso por redes públicas.

Configuración de recursos: Charset de PostgreSQL

Otro desafío encontrado durante la implementación fue la incompatibilidad entre el charset configurado en PostgreSQL y el utilizado por la ORM empleada en NestJS. Este problema se manifestó principalmente en escenarios donde los datos string contenían caracteres especiales, como acentos o la letra "ñ", generando errores de codificación y afectando la integridad de los datos.

Solución:

Para resolver este problema, se realizaron las siguientes acciones:

- **Configuración del Charset en PostgreSQL:** Se descargó e instaló un charset compatible con el idioma español, como UTF-8, y se configuró el servidor de base de datos para utilizarlo de manera predeterminada.
- **Ajustes en la Máquina Virtual:** Se actualizó la configuración regional de la máquina virtual donde se aloja PostgreSQL, estableciendo parámetros de idioma y codificación acordes al español, garantizando la consistencia en el manejo de caracteres especiales.

5.8. Costos asociados al tráfico de red

En infraestructuras en la nube, los costos asociados al tráfico de salida (egress) entre regiones, o incluso hacia internet, pueden ser significativos, especialmente en sistemas que dependen de grandes transferencias de datos, como imágenes y documentos en "Stocknow".

Solución:

- **Minimización del tráfico entre regiones:** Todos los componentes se alojaron en la misma región de GCP para evitar costos adicionales por tráfico interregional.
- **Uso de Cloud Storage para distribución directa:** Los archivos se almacenan en Cloud Storage, desde donde los usuarios finales pueden acceder directamente. Esto reduce la necesidad de transferir archivos repetidamente desde el backend hacia los usuarios.

Otras estrategias.

- **Uso de balanceadores de carga:** Para manejar posibles picos de tráfico y evitar cuellos de botella, asegurando que el sistema pueda escalar sin interrupciones. Un ejemplo de balanceador que ofrece GCP es el "HTTP(S) Load Balancer" que está diseñado para aplicaciones web que requieren balanceo de tráfico HTTP o HTTPS. Es global y soporta configuración de URL maps para enrutar tráfico a diferentes backends según el patrón de URL.



- **Optimización de la base de datos:** La base de datos fue configurada de forma personalizada para evitar errores de codificación.

6 Implementación.

En este capítulo se detallará como se configuró el proyecto, habilitando servicios, creando componentes, etc.

6.1. Creación del Proyecto.

El primer paso para implementar la infraestructura en la nube fue crear un nuevo proyecto desde la consola de GCP. Este proyecto, iniciado desde cero, requirió la activación de las APIs esenciales para los recursos que formarían parte de la infraestructura. Entre las principales, se activó la Compute Engine API, indispensable para la creación y gestión de las máquinas virtuales (VM), consideradas el núcleo de la infraestructura, ya que alojan el software desarrollado.

Para las Máquinas Virtuales se configuraron del tipo e2-small la cual cuenta con:

- 2 núcleos compartidos, 1 núcleo de CPU compartido.
- 2 GB de RAM.
- 15 GB de almacenamiento en disco persistente / 10GB en caso de la máquina frontend.
- Ubuntu 24.04 amd64.

Posteriormente se habilitó Cloud Storage para gestionar los archivos relacionados con el sistema, creando un bucket específico para almacenar documentos e imágenes necesarias para el inventario, garantizando accesibilidad desde cualquier ubicación.

También se configuró un **balanceador de carga** para distribuir las solicitudes entrantes de manera uniforme entre las máquinas virtuales del frontend y backend. Este componente actúa como puente entre las solicitudes externas y las subredes correspondientes, asignando una IP pública al balanceador que facilita el acceso al sistema desde internet.

Finalmente, para asegurar la conectividad y la seguridad, se configuraron reglas de firewall que permitieron únicamente el tráfico necesario para el correcto funcionamiento del sistema. Ejemplos de estas reglas incluyen:

- **Puerto 80:** Para tráfico HTTP, habilitando la comunicación con el frontend.
- **Puerto 5432:** Para la conexión con PostgreSQL en el backend.

6.2 Configuración de Servicios de Infraestructura

Sistema Operativo: Ubuntu

Las máquinas virtuales configuradas para el sistema "Stocknow" utilizan **Ubuntu** como sistema operativo. La elección de Ubuntu sobre otras distribuciones como CentOS o ArchLinux se debe a mi familiaridad con este sistema operativo, adquirido a través de años de experiencia en el despliegue de aplicaciones. Es importante señalar que esta alternativa es válida dado el **deadline** del proyecto, el cual al ser cercano no se dedicó tanto tiempo a realizar un análisis exhaustivo de sistemas operativos. Esta decisión permitió reducir los tiempos de configuración y minimizar el riesgo de errores, garantizando una implementación más eficiente.

Configuración de la Máquina del Frontend: frontend-web

La primera máquina configurada fue la destinada al frontend, llamada frontend-web. En esta máquina, el desafío principal fue desplegar la página web desarrollada en Angular.

Pasos de Configuración:

- Instalación de Node.js:
Dado que Angular es un framework que opera sobre Node.js, se instaló la versión v18.19, que es la misma con la que se desarrolló el frontend.
- Instalación y Configuración de Nginx:
Se instaló Nginx como servidor web para servir la aplicación Angular.
La aplicación se ubicó en el directorio /var/www/frontend-web/, donde Nginx se configuró para cargarla automáticamente.

```
##
# server configuration
##
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    root /var/www/frontend_web_muni/browser;

    index index.html index.htm index.nginx-debian.html;

    server_name _;

    location / {
        try_files $uri $uri/ /index.html?$query_string;
    }
}
```

Figura 9 Configuración Nginx servidor Frontend (Terminal GCP)

Muestra el archivo de configuración, en el cual se especifica la raíz del contenido servido, las reglas de acceso y el uso del puerto 80.

- Configuración de Nginx:
El archivo de configuración relevante se encuentra en /etc/nginx/sites-available/default. Este archivo define las reglas para servir el contenido del frontend y asegura que el tráfico HTTP por el puerto 80 (habilitado previamente en las reglas de firewall) sea correctamente gestionado.

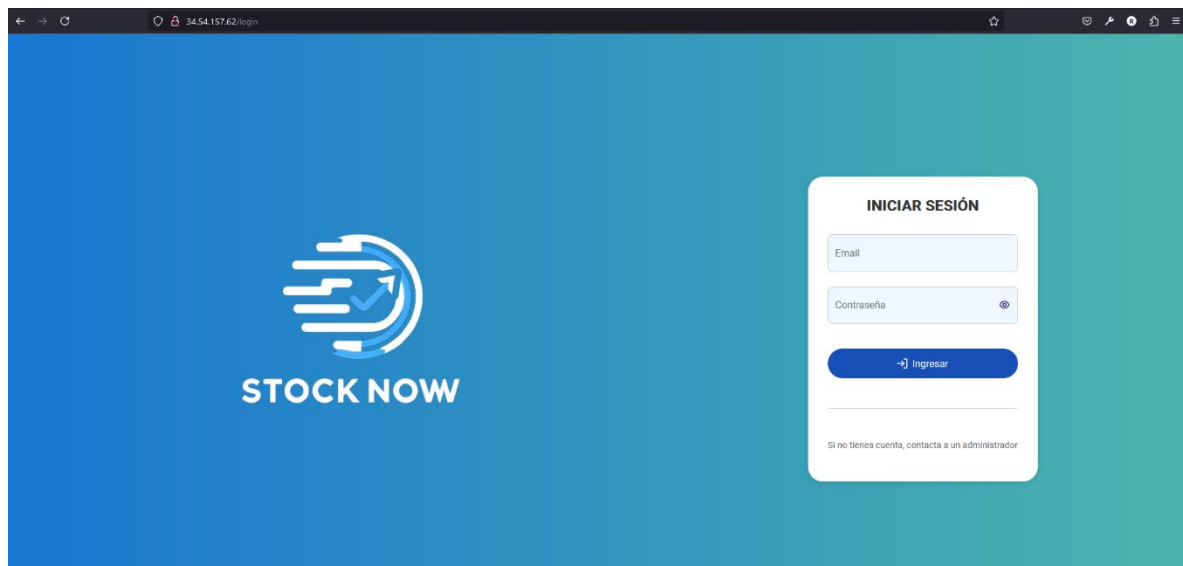


Figura 10 Vista web accediendo a través de HTTP (Captura sitio web Stocknow)

Se muestra la página del frontend accesible desde la IP pública asignada, verificando que la configuración permite el acceso desde internet sin inconvenientes.

Configuración de la Máquina Backend: cl-stocknow

La configuración de la máquina backend implicó varios pasos para garantizar que el servidor API desarrollado en NestJS funcione correctamente, además de integrar la base de datos PostgreSQL. A continuación, se describen los pasos realizados.

Instalación dependencias clave.

- **Node.js:** Instalamos la versión 22.7.0, necesaria para garantizar la compatibilidad con el backend desarrollado en NestJS. Cabe destacar que esta versión no es al azar, sino que es la que se usó para desarrollar la aplicación.
- **Nginx:** Se configuró como servidor proxy reverso para garantizar la compatibilidad con el backend desarrollado en NestJS. Esta configuración permite:
 - Mejorar el rendimiento mediante la optimización del manejo de solicitudes
 - Incrementar la seguridad establecer una regla de redirección de puertos, redirigiendo todo el tráfico entrante desde el puerto 80 al puerto 3000.
 - Aumenta la escalabilidad al permitir una mejor gestión del tráfico.
- **PM2:** Este módulo se instaló para gestionar el proceso del backend como un servicio administrable. Configurar la api para que sea manejada por PM2 ofrece ventajas como:
 - Monitorización en tiempo real.
 - Gestión sencilla de reinicios.
 - Registro de logs directamente en el sistema.

Para la configuración de Nginx, se modificó el archivo `/etc/nginx/sites-available/default`. Haciendo que tome la ruta de la api, la cual es `/var/www/backend_muni`. También se aplicó una configuración para hacer que el tráfico entrante lo resuelva mediante proxy reverso, haciendo que lo que llega desde el puerto 80 se redirija al puerto 3000, donde se ejecuta la API de NestJS.

```
server {
    listen 80;
    server_name stocknow.cl;
    root /var/www/backend_muni;
    location / {
        proxy_pass http://localhost:3000; # puerto 3000 de la api
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
        client_max_body_size 10M;
    }
}
```

Figura 11 Configuración Nginx servidor Backend (Terminal de GCP)

Adicionalmente se instaló **PostgreSQL**. Se instaló la versión 17 de este motor de Base de datos. Por defecto Postgre usa el puerto 5432, puerto que fue configurado anteriormente en la configuración del proyecto GCP para permitir el tráfico a través de este. También se configuró el charset predeterminado al `es-ES.utf-8` para evitar problemas con caracteres como la ñ, tildes y/u otros caracteres específicos del español.

Se crearon las credenciales necesarias (usuario, contraseña, y demás parámetros). Se configuraron en el archivo de entorno (`.env`), ubicado en la raíz de la API.

Finalmente, para el despliegue con Integración Continua, elegí simplificar y automatizar el proceso, implementando un script bash de CI/CD con las siguientes funcionalidades.

- Detiene el proceso existente en PM2.
- Actualiza el código del repositorio remoto desde la rama especificada (por ejemplo, sprint2).
- Reconstruye el proyecto utilizando **yarn** para instalar dependencias y **npm run build** para generar el build del backend.
- Reinicia la aplicación con PM2 y guarda la configuración.

```
#!/bin/sh
#####
#ESTE SCRIPT PULLEA DESDE LA BRANCH INDICADA #
# PARA SUBIR A /var/www/backend-muni #
#####
branch="sprint2"
ruta="/var/www/backend_muni"
cd $ruta || { echo "Error: No se pudo acceder a $ruta"; exit 1; }
pm2 stop 0 || { echo "Error: No se pudo detener el proceso PM2"; exit 1; }
sudo git checkout $branch || { echo "Error: No se pudo hacer checkout a la rama $branch"; exit 1; }
sudo git pull --force || { echo "Error: No se pudo hacer pull del repositorio"; exit 1; }
#npm install no funciona cloudinary :(
echo 'Ejecutando comando "yarn install"'
yarn install
npm run build || { echo "Error: Falló la construcción del proyecto"; exit 1; }
pm2 start dist/main.js --name backend-muni --time || { echo "Error: No se pudo iniciar el proceso PM2"; exit 1; }
pm2 save || { echo "Error: No se pudo guardar la configuración de PM2"; exit 1; }
sudo systemctl restart nginx || { echo "Error: No se pudo reiniciar Nginx"; exit 1; }
echo 'Ha sido actualizado el build CORRECTAMENTE';
```

Figura 12 Configuración de script de "integración continua". (Terminal de GCP)

Una vez ejecutado el script, asegura que el backend se actualice automáticamente con los últimos cambios del repositorio y quedando guardado como un proceso PM2. Podemos visualizar el proceso si ejecutamos el comando "pm2 list".

```
root@cl-stocknow:~# pm2 list
```

id	name	namespace	version	mode	pid	uptime	o	status	cpu	mem	user	watching
0	backend-muni	default	0.0.1	fork	134356	6h	0	online	0%	152.0mb	root	disabled

```
root@cl-stocknow:~#
```

Figura 13 Detalle de aplicación desplegada en PM2 (Terminal de GCP)

A continuación, se presenta un diagrama el cual detalla los pasos a seguir para el proceso de integración continua de la aplicación en la maquina backend.

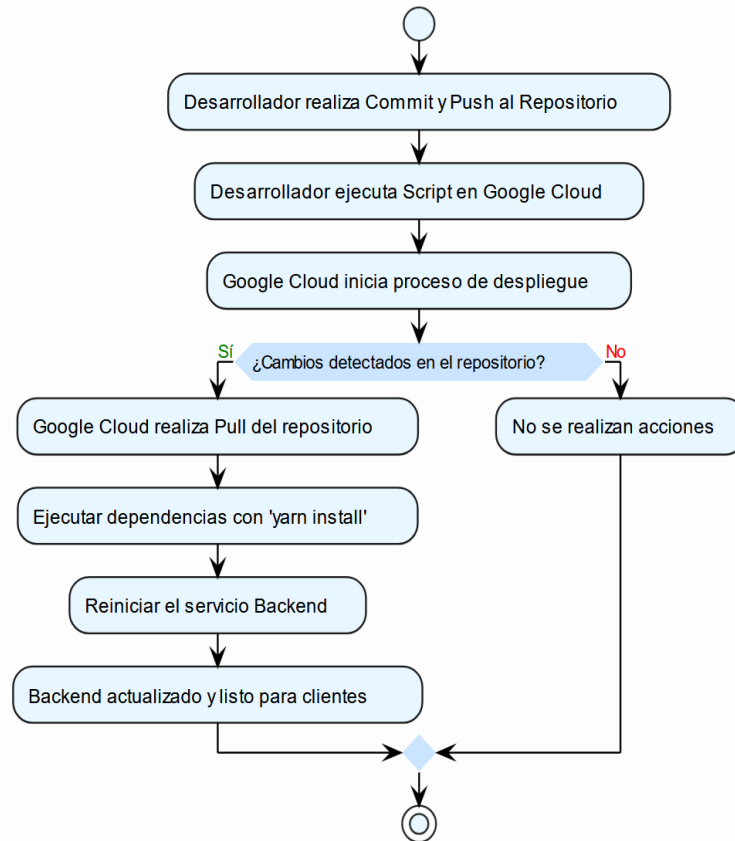


Figura 13 Diagrama de flujo CI/CD (elaboración Cristóbal Herrera)

6.3. Despliegue y Pruebas

Con la infraestructura previamente montada y configurada, se pudo comprobar su correcto funcionamiento al ser accesible desde diversos métodos:

- **Aplicación móvil:** La aplicación móvil se comunica con el backend mediante solicitudes HTTP, y las respuestas son recibidas sin problemas, validando la conectividad entre los componentes.
- **Aplicación web:** La aplicación web interactúa con el backend a través del frontend desplegado en la máquina correspondiente. La comunicación es fluida y consistente.
- **Pruebas directas:** Se realizaron consultas HTTP directamente al backend utilizando herramientas como **Postman** y navegadores web, confirmando que el sistema responde correctamente a las solicitudes.

6.4. Pruebas de carga y rendimiento.

Durante las etapas finales del desarrollo, junto a un miembro del equipo, encargado del backend, realizamos pruebas para evaluar el rendimiento de la infraestructura. Estas pruebas confirmaron que el sistema ofrece un rendimiento aceptable, adecuado para las necesidades del negocio. Optar por



mantener la base de datos integrada en la misma máquina virtual que el backend resultó ser una estrategia efectiva para reducir latencia y optimizar la comunicación entre ambos componentes, en comparación con alternativas que podrían haber introducido demoras adicionales.

6.5. Seguridad.

La infraestructura implementada es segura gracias a **Reglas de Firewall de GCP**, en estas se liberaron los puertos estrictamente necesarios para la operación de la aplicación, como el puerto 80 para tráfico HTTP, puerto 443, para tráfico HTTPS, y el puerto 5432 para PostgreSQL. Otros puertos como el 3000, por ejemplo, permanecen inaccesibles desde internet (ya que están todos los otros puertos cerrados por defecto dada la configuración de Google Firewall).

También el acceso al proyecto y a las APIs del mismo, está protegido por **Cloud IAM** para que solo los usuarios autorizados puedan gestionar o interactuar con los recursos, por ejemplo, con los archivos del Bucket.

6.6. Desafíos técnicos durante la implementación.

Configuración de red: Balanceador de carga

Uno de los principales desafíos fue la implementación del balanceador de carga, ya que anteriormente no habíamos trabajado con este componente en proyectos previos. Tuvimos que familiarizarnos con su configuración en GCP, aprender a crear reglas de enrutamiento y vincular correctamente las máquinas virtuales. Este proceso implicó varias pruebas y ajustes, pero finalmente logramos distribuir el tráfico eficientemente entre las instancias.

Configuración de recursos: Charset de PostgreSQL

El problema más significativo surgió con la configuración del charset en PostgreSQL. Durante las pruebas, algunas consultas fallaban debido a incompatibilidades en los caracteres, como la ñ y vocales acentuadas. Este problema estaba relacionado con el uso de una ORM en el backend, que reducía el control directo sobre el charset en las consultas. Para esto investigué y apliqué una solución consistente en:

- Cambiar el charset predeterminado de PostgreSQL a es-ES.UTF-8.
- Configurar la máquina virtual para que también utilice el mismo charset en su entorno.

7. Conclusiones.

El objetivo principal de diseñar e implementar una infraestructura en la nube para el proyecto Stocknow fue alcanzado con éxito, como lo demuestra el hecho de que el sistema está completamente operativo y accesible a través de internet. Esta infraestructura responde a las necesidades identificadas inicialmente, ofreciendo escalabilidad, rendimiento óptimo, seguridad robusta y costos controlados. Además, está preparada para expandirse fácilmente a más usuarios en el futuro, lo que la hace altamente flexible y viable para escenarios más amplios. La elección de Google Cloud Platform permitió aprovechar herramientas avanzadas como las reglas de firewall y la gestión de usuarios mediante IAM, además proporcionó un control externo y una administración simplificada, garantizando un manejo centralizado y seguro del sistema.

En cuanto al impacto del proyecto en el stakeholder original, la Municipalidad de Viña del Mar, Stocknow ha demostrado mejorar significativamente la operación de sus comedores solidarios. Esta mejora fue validada por los propios usuarios, quienes confirmaron que la plataforma es cómoda, intuitiva y útil para su gestión diaria, asegurando que la adoptarían como herramienta en el futuro.



Más allá de su funcionalidad técnica, el proyecto aporta un importante valor social, ya que optimiza la distribución de recursos y reduce el desperdicio de alimentos, beneficiando directamente a las personas más necesitadas. Esta doble contribución, técnica y social, refuerza el propósito del sistema y lo posiciona como una solución integral a un problema crítico.

Durante la implementación, se superaron desafíos técnicos relevantes, como la configuración del balanceador de carga y la solución de problemas relacionados con el charset en PostgreSQL, lo que permitió adquirir conocimientos valiosos para proyectos futuros. Además, comparar diferentes servicios de Cloud Computing, como AWS y GCP, amplió nuestra experiencia en áreas como la gestión de instancias y configuraciones de seguridad, habilidades altamente demandadas en la actualidad. Si bien el sistema actual cumple con los requisitos del proyecto, existen oportunidades de mejora significativas, como la integración de servicios adicionales como BigQuery para el análisis avanzado de datos y la generación de reportes más precisos. Estas ideas surgen del feedback recibido durante la Feria de Software, donde el prototipo despertó interés en profesionales del área, quienes sugirieron mejoras valiosas que podrían implementarse en el futuro.



Glosario.

1. **Cloud Computing:** Tecnología que permite el acceso remoto a servicios informáticos como almacenamiento, procesamiento y software a través de internet.
2. **GCP (Google Cloud Platform):** Plataforma de servicios en la nube de Google que proporciona herramientas como Compute Engine, Cloud Storage y Cloud IAM.
3. **AWS (Amazon Web Services):** Plataforma de servicios en la nube de Amazon que incluye servicios.
4. **VM (Virtual Machine):** Máquina virtual que emula un entorno informático completo, permitiendo ejecutar sistemas operativos y aplicaciones.
5. **IAM (Identity and Access Management):** Servicio que gestiona identidades y permisos para garantizar un acceso seguro a recursos.
6. **VPC (Virtual Private Cloud):** Red privada virtual que permite a las organizaciones aislar sus recursos en la nube.
7. **Kubernetes:** Sistema de orquestación de contenedores que permite desplegar, gestionar y escalar aplicaciones en entornos distribuidos.
8. **TensorFlow:** Biblioteca de aprendizaje automático de código abierto diseñada para desarrollar modelos de inteligencia artificial.
9. **Balanceador de carga (Load Balancer):** Herramienta que distribuye el tráfico entre servidores para optimizar el rendimiento y la disponibilidad.
10. **Compute Engine:** Servicio de Google Cloud que permite crear y gestionar máquinas virtuales para ejecutar cargas de trabajo en la nube.



Referencias.

- [1] ¿Qué es la informática en la cloud? Ejemplos, ventajas y más (Jackins) splashtop. Recuperado de <https://www.splashtop.com/es/blog/what-is-cloud-computing>
- [2] Statista. "Worldwide Market Share of Leading Cloud Infrastructure Service Providers." Recuperado de <https://www.statista.com/chart/18819/worldwide-market-share-of-leading-cloud-infrastructure-service-providers/>
- [3] CESUMA. "Historia y Propiedades del Cloud Computing." Recuperado de <https://www.cesuma.mx/blog/cloud-computing-historia-y-propiedades.html>
- [4] Las principales tendencias del Cloud Computing para 2023 (s.f.) Pontificia Universidad Católica de Chile. Recuperado de: <https://educacionprofesional.ing.uc.cl/las-principales-tendencias-del-cloud-computing-para-2023>
- [5] ¿Qué es el software como servicio (SaaS)? (s.f) Amazon web services. Recuperado de <https://aws.amazon.com/es/what-is/saas>
- [6] *¿Qué es Cloud Computing? Conceptos básicos y beneficios.* AWS. Recuperado de <https://aws.amazon.com/es/what-is-cloud-computing>
- [7] Google Cloud vs AWS en 2024 (Comparación de los gigantes) (Jones). Recuperado de: <https://kinsta.com/es/blog/google-cloud-vs-aws/>
- [8] CrowdStrike. "What Is Cloud Security? Best Practices and Strategies." Recuperado de <https://www.crowdstrike.com/en-us/cybersecurity-101/cloud-security/>
- [9] Google Cloud Infrastructure. Google Cloud. Recuperado de <https://cloud.google.com/infrastructure>
- [10] Network Performance in AWS: 5 Factors to Consider. Network Computing. Recuperado de <https://www.networkcomputing.com/network-infrastructure/network-performance-in-aws-5-factors>
- [11] Computación en la nube con AWS (s.F) Amazon web services. Recuperado de <https://aws.amazon.com/es/what-is-aws/>

Anexos.

- Link para repositorio Github del proyecto.
<https://github.com/Inventario-Municipalidad-VinaDelMar>
- Proyecto Stocknow, nota FESW 2024
<https://usm.cl/noticias/estudiantes-desarrollan-una-innovadora-app-para-la-gestion-de-inventarios-en-tiempo-real/>
- Video "preview" de la aplicación en ambiente "productivo".
<https://youtu.be/ctnrdgZT3pU>