

Virtualización en la Nube
Universidad Politécnica Salesiana
Computación - Periodo 67

Integrantes: Karen Ortiz - Andrés Encalada

Objetivo de la práctica:

Se debe diseñar, desplegar y documentar una arquitectura distribuida de tres capas en una plataforma en la nube (AWS, Google Cloud o Azure), incorporando servicios de almacenamiento compartido y balanceo de carga.

La solución simulará el backend de un portal académico donde se gestionan recursos multimedia, generados o cargados por los usuarios del sistema.

Plataforma: Google Cloud

Esta práctica fue realizada haciendo uso de la plataforma Google Cloud, la cual proporciona las herramientas necesarias para cumplir con los objetivos propuestos en el enunciado. Entre los servicios destacados que fueron de utilidad para la práctica tenemos creación de máquinas virtuales, balanceadores de carga y autoscaling, creación de redes y reglas de Firewall, almacenamiento compartido y bases de datos.

Desarrollo de la actividad

Parte 1: Preparación y aprovisionamiento del entorno

1. Infraestructura base en la nube

• Instancias Virtuales

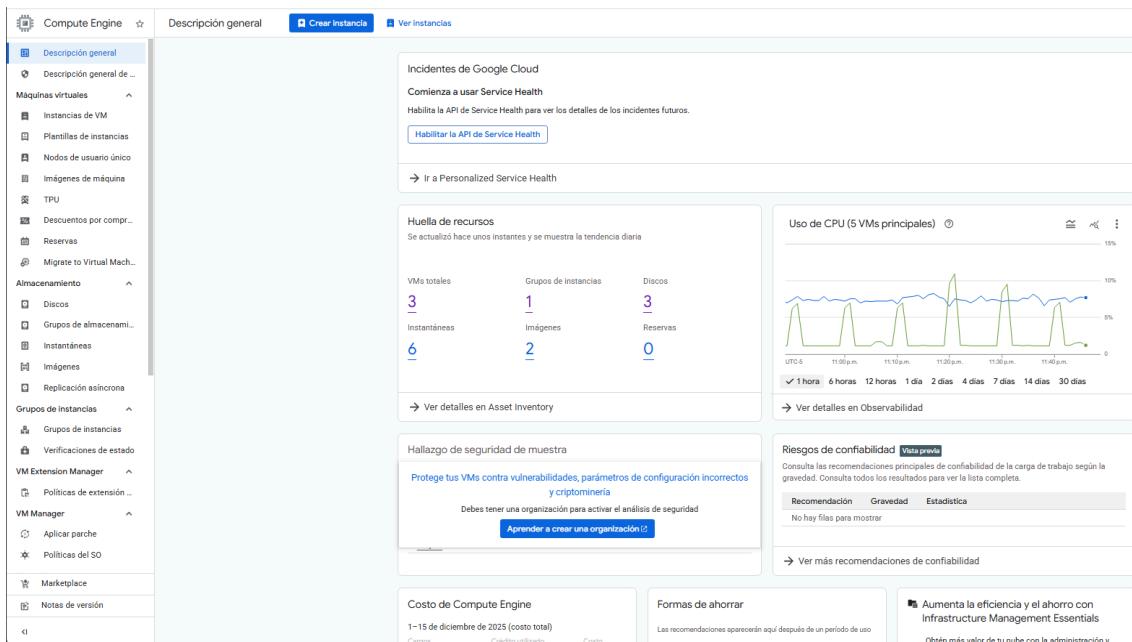
Todas las instancias fueron creadas con la misma configuración, incluida las máquinas generadas por el balanceador de carga. Por lo que todas usan un sistema operativo Ubuntu 22.04 LTS x86-64 y están alojadas en el servidor ‘us-central1-c’. A continuación se presentan las instancias creadas inicialmente.

- ui-server:** Instancia que posee el servidor de la interfaz gráfica, pertenece a la red ‘subred-publica’.
- api-server:** Instancia que posee la API, la cual conecta al frontend con la base de datos y el almacenamiento compartido, pertenece a la red ‘subred-privada’. Después de generar una plantilla está instancia es detenida y apagada.

• Creación de componentes

Para crear una instancia debemos dirigirnos a la sección Compute Engine, Instancias de VM y posteriormente, Crear una instancia. La configuración de cada instancia dependerá directamente de las necesidades del proyecto. Además, en la sección Compute Engine, encontramos otras funciones, como grupos de instancias, plantillas de instancias e imágenes de instancias, funciones esenciales para el funcionamiento del autoescalamiento y balanceo de carga.

Fig. 1. Panel principal de la sección Compute Engine



2. Configuración de seguridad y redes

Se creó una red llamada ‘red-práctica’, la cual contiene 2 subredes, ‘subred-publica’ y ‘subred-privada’

- Acceso HTTP público al servidor UI.

La subred pública permite el acceso HTTP a los usuarios desde cualquier dirección IP, garantizando un ingreso público al servidor U. Para esto se habilitó el puerto 80 con el protocolo TCP.

- Acceso restringido API → DB mediante puertos específicos.

Para la comunicación interna se definió una regla de firewall que permite la comunicación desde los destinos 10.0.0.0/8 (Máquinas virtuales) y 10.129.0.0/23 (Proxy), además, se definieron puertos específicos para lograr una comunicación entre los servicios. TCP: 22,3000,5432,111,2049,80, UDP: 2049,111 y el protocolo ICMP para habilitar el servicio ping. A continuación, se presenta un desglose de los puertos y su función.

Tabla 1. Explicación de puertos habilitados

Protocolo	Puerto	Función
TCP	22	Usado para ingresar mediante SSH
	3000	Es usado por el servidor de backend para escuchar las peticiones.
	5432	Se usa para que la API pueda comunicarse con la base de datos (PostgreSQL)
	80	Usado para la comunicación con el servidor web.
TCP/UDP	2049	Usado para almacenar los archivos en Filestore
	111	Utilizado por los servicios para poder obtener la ruta del almacenamiento compartido
ICMP	-	Protocolo utilizado para habilitar el comando ping.

- Acceso del API Instance al servicio de almacenamiento compartido.

Este acceso se da gracias al uso del puerto 111 y 2049, explicados en la Tabla 1.

- SSH solo para direcciones IP autorizadas.

Para esto se estableció un rango de direcciones 35.235.240.0/20, usando el protocolo TCP en el puerto 22, con esto logramos que solo los administradores autenticados en la consola de Google puedan establecer una conexión.

- Etiquetado, agrupamiento y subredes según la plataforma escogida.

Se utilizaron las etiquetas ‘allow-health-check’ y ‘allow-ssh’ para autorizar el tráfico entre las instancias. De esta forma se logra un agrupamiento en las instancias, a las que se les aplicaron leyes de firewall donde se establecen las subredes, direcciones IP y puertos con los que se pueden comunicar.

- **Creación de componentes**

Para crear las redes y subredes, debemos dirigirnos a la sección Redes de VPC, donde podremos crear nuestra red principal, sus subredes, y las políticas y reglas de Firewall.

Fig. 2. Panel principal de la sección Redes de VPC

The screenshot shows the 'Redes de VPC' (VPC Networks) page. On the left, there's a sidebar with navigation links: 'Redes de VPC', 'Direcciones IP', 'Rangos internos', 'Usa tu propia IP', 'Firewall', 'Rutas', 'Intercambio de tráfico entre...', 'VPC compartida', 'Acceso a VPC sin servido...', 'Duplicación de paquetes', and 'Registros de flujo de VPC'. The main content area has tabs: 'Redes de VPC' (selected), 'Crear red de VPC', and 'Actualizar'. Below these are sections for 'Redes del proyecto actual' and 'Subredes del proyecto actual'. A prominent callout box highlights Network Connectivity Center (NCC) with text: '¿Buscas conectividad de red entre redes de VPC a gran escala? Usa NCC para la conectividad de red entre redes de VPC.' It also mentions that NCC offers connectivity between VPC networks up to 10 times faster than traditional interconnection methods. Another section discusses traffic exchange rules and their impact on network performance. At the bottom, there are buttons for 'Ir a NCC', 'Más información', and 'Recordármelo más tarde'. A note at the bottom states: 'El puerto SMTP 25 no está autorizado en este proyecto. [Más información](#)'.

3. Almacenamiento compartido

Se configuró el servicio Filestore integrado en la plataforma Google Cloud, posteriormente se creó un directorio de montaje local en '/mnt/storage', el cual sirve como interfaz de acceso al disco remoto. Finalmente, se editó el archivo '/etc/fstab' para garantizar que cualquier nueva instancia creada por el Autoescalador monte el disco automáticamente al arrancar.

- **Creación de componentes**

Para crear una instancia de FileStore, debemos buscar la sección del mismo nombre. Al crear una instancia podremos configurar diferentes parámetros, como la capacidad, la región y la red a la que pertenece, entre otros.

Fig 3. Panel principal de Filestore

The screenshot shows the 'Filestore / Instancias' (Instances) page. On the left, there's a sidebar with navigation links: 'Instancias', 'Copias de seguridad', and 'Replicaciones'. The main content area has tabs: 'Instancias' (selected), 'Crear instancia', and 'Más información'. A descriptive text block explains what a Filestore instance is: 'Una instancia es un sistema de almacenamiento conectado a la red completamente administrado que se puede usar junto con las instancias de Google Compute Engine y Kubernetes Engine. [Más información](#)'. Below this is a filter input field labeled 'Filtro' with the placeholder 'Escribir el nombre o valor de la propiedad'. A table lists existing instances:

ID de instancia	Nombre del archivo compartido	Fecha y hora de creación	Nivel de servicio	Ubicación	Protocolo	Dirección IP	Capacidad	Etiquetas
default	archivos	12 dic 2025, 6:59 p.m.	BASIC_HDD	us-central1-a	NFSv3	10.207.0.18	1 TiB	

Parte 2: Implementación de la plataforma

1. Capa de Interfaz

Esta capa actúa como el punto de entrada para los usuarios finales, proporciona una interfaz gráfica para interactuar con los servicios del sistema. Se implementó siguiendo el modelo SPA (Single Page Application) con React.js.

La interfaz actúa como consumidor de la API RESTful, por medio del protocolo HTTP asíncrono y se realiza el intercambio de información en formato JSON. Se divide en dos vistas lógicas:

- **Módulo de autenticación:** permite al usuario iniciar sesión y registrar usuarios (el rol de admin tiene visualización completa a todos los archivos subidos por los usuarios). Además de poder visualizar errores (credenciales inválidas y usuario creado).

Fig 4. Módulo de autenticación

Iniciar Sesión

🔒

Entrar

¿No tienes cuenta?

[Crear Usuario Nuevo](#)

Crear Cuenta

📝

Tipo de Usuario:

Registrarse

¿Ya tienes una cuenta?

[Ir a Iniciar Sesión](#)

- **Panel Principal:** Visualización en cuadrícula de los archivos subidos con previsualización y opción de descarga.

Fig 5. Dashboard con los archivos subidos por usuarios

Hola, Estudiante 2

[Salir](#)

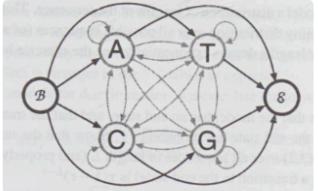
Mis Archivos

Subir Nuevo

ExamenIA.html
Tipo: text/html | Usuario ID: 3

[⬇ Descargar Archivo](#)

Captura de pantalla 2025-11-11 014737.png
Tipo: image/png | Usuario ID: 3



Captura de pantalla 2025-09-07 231838.png
Tipo: image/png | Usuario ID: 3

Electrónico	Resumen de la transacción o descripción	Provee	Para la Recolección o Publicidad	Aplicación
E0001				
E0002				
E0003				
E0004				
E0005				
E0006				
E0007				
E0008				
E0009				
E0010				
E0011				
E0012				
E0013				
E0014				
E0015				
E0016				
E0017				
E0018				
E0019				
E0020				
E0021				
E0022				
E0023				
E0024				
E0025				
E0026				
E0027				
E0028				
E0029				
E0030				
E0031				
E0032				
E0033				
E0034				
E0035				
E0036				
E0037				
E0038				
E0039				
E0040				
E0041				
E0042				
E0043				
E0044				
E0045				
E0046				
E0047				
E0048				
E0049				
E0050				
E0051				
E0052				
E0053				
E0054				
E0055				
E0056				
E0057				
E0058				
E0059				
E0060				
E0061				
E0062				
E0063				
E0064				
E0065				
E0066				
E0067				
E0068				
E0069				
E0070				
E0071				
E0072				
E0073				
E0074				
E0075				
E0076				
E0077				
E0078				
E0079				
E0080				
E0081				
E0082				
E0083				
E0084				
E0085				
E0086				
E0087				
E0088				
E0089				
E0090				
E0091				
E0092				
E0093				
E0094				
E0095				
E0096				
E0097				
E0098				
E0099				
E0100				
E0101				
E0102				
E0103				
E0104				
E0105				
E0106				
E0107				
E0108				
E0109				
E0110				
E0111				
E0112				
E0113				
E0114				
E0115				
E0116				
E0117				
E0118				
E0119				
E0120				
E0121				
E0122				
E0123				
E0124				
E0125				
E0126				
E0127				
E0128				
E0129				
E0130				
E0131				
E0132				
E0133				
E0134				
E0135				
E0136				
E0137				
E0138				
E0139				
E0140				
E0141				
E0142				
E0143				
E0144				
E0145				
E0146				
E0147				
E0148				
E0149				
E0150				
E0151				
E0152				
E0153				
E0154				
E0155				
E0156				
E0157				
E0158				
E0159				
E0160				
E0161				
E0162				
E0163				
E0164				
E0165				
E0166				
E0167				
E0168				
E0169				
E0170				
E0171				
E0172				
E0173				
E0174				
E0175				
E0176				
E0177				
E0178				
E0179				
E0180				
E0181				
E0182				
E0183				
E0184				
E0185				
E0186				
E0187				
E0188				
E0189				
E0190				
E0191				
E0192				
E0193				
E0194				
E0195				
E0196				
E0197				
E0198				
E0199				
E0200				
E0201				
E0202				
E0203				
E0204				
E0205				
E0206				
E0207				
E0208				
E0209				
E0210				
E0211				
E0212				
E0213				
E0214				
E0215				
E0216				
E0217				
E0218				
E0219				
E0220				
E0221				
E0222				
E0223				
E0224				
E0225				
E0226				
E0227				
E0228				
E0229				
E0230				
E0231				
E0232				
E0233				
E0234				
E0235				
E0236				
E0237				
E0238				
E0239				
E0240				
E0241				
E0242				
E0243				
E0244				
E0245				
E0246				
E0247				
E0248				
E0249				
E0250				
E0251				
E0252				
E0253				
E0254				
E0255				
E0256				
E0257				
E0258				
E0259				
E0260				
E0261				
E0262				
E0263				
E0264				
E0265				
E0266				
E0267				
E0268				
E0269				
E0270				
E0271				
E0272				
E0273				
E0274				
E0275				
E0276				
E0277				
E0278				
E0279				
E0280				
E0281				
E0282				
E0283				
E0284				
E0285				
E0286				
E0287				
E0288				
E0289				
E0290				
E0291				
E0292				
E0293				
E0294				
E0295				
E0296				
E0297				
E0298				
E0299				
E0300				
E0301				
E0302				
E0303				
E0304				
E0305				
E0306				
E0307				
E0308				
E0309				
E0310				
E0311				
E0312				
E0313				
E0314				
E0315				
E0316				
E0317				
E0318				
E0319				
E0320				
E0321				
E0322				
E0323				
E0324				
E0325				
E0326				
E0327				
E0328				
E0329				
E0330				
E0331				
E0332				
E0333				
E0334				
E0335				
E0336				
E0337				
E0338				
E0339				
E0340				
E0341				
E0342				
E0343				
E0344				
E0345				
E0346				
E0347				
E0348				
E0349				
E0350				
E0351				
E				

- Subida de archivos: Formulario para seleccionar la categoría y el archivo a subir.

Fig 6. Sección subir archivos

Hola, Estudiante 2 user

Mis Archivos Subir Nuevo

Subir Archivo

Selecciona Categoría

Seleccionar archivo Ningún archivo seleccionado

Subir a la Nube

2. Capa de Servicios

Se desarrolló un conjunto de endpoints REST, los cuales sirven para registrar usuarios, subir, listar y obtener metadatos de los archivos.

Para comprobar el funcionamiento de los endpoints se realizaron pruebas en Postman. A continuación se presentan los resultados de cada endpoint.

- Conexión API a base de datos

Fig 7. Operación Post registrando un usuario

POST http://35.223.73.122/api/register Send

Body Cookies

raw JSON Schema Beautify

```

1 {
2   "username": "EstudianteNube",
3   "email": "test@universidad.edu.ec",
4   "password": "securePass123",
5   "role": "admin"
6 }
```

Body 200 OK 423 ms 355 B Elapsed Size Raw Copy

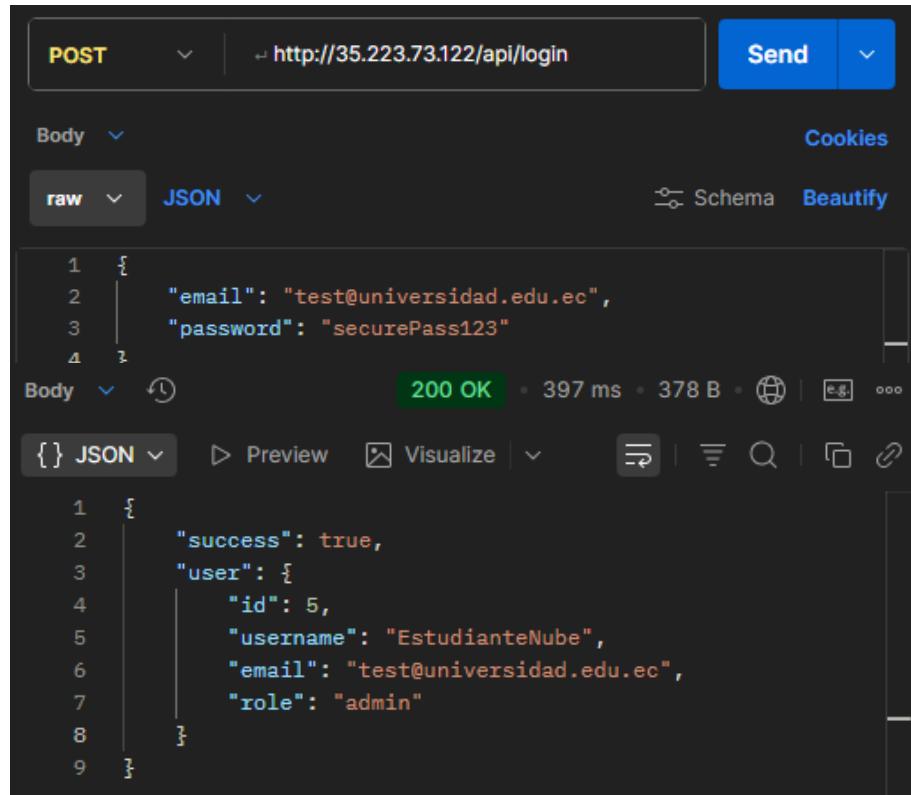
{ } JSON Preview Visualize Copy Find Clear

```

1 {
2   "message": "Usuario creado",
3   "user": {
4     "id": 5,
5     "username": "EstudianteNube",
6     "role": "admin"
7   }
8 }
```

- Prueba de Login

Fig 8. Operación Post para iniciar sesión



```

POST http://35.223.73.122/api/login
Send

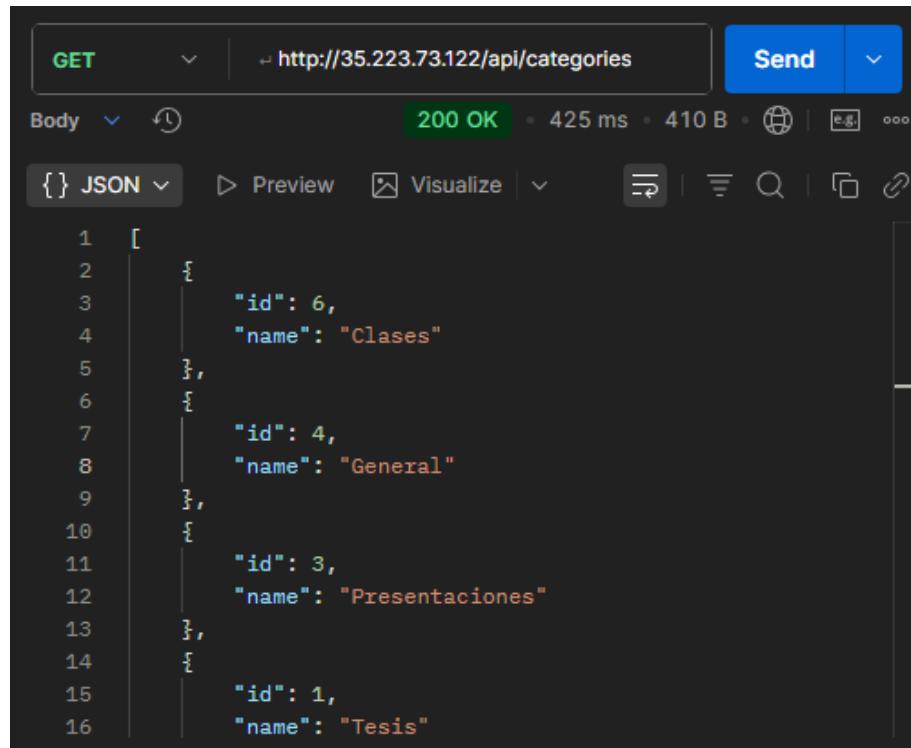
Body Cookies
raw JSON Schema Beautify

1 {
2   "email": "test@universidad.edu.ec",
3   "password": "securePass123"
4 }

Body 200 OK 397 ms 378 B
[] JSON Preview Visualize
1 {
2   "success": true,
3   "user": {
4     "id": 5,
5     "username": "EstudianteNube",
6     "email": "test@universidad.edu.ec",
7     "role": "admin"
8   }
9 }
  
```

- Obtención de datos

Fig 9. Operación Get para obtener categorías



```

GET http://35.223.73.122/api/categories
Send

Body 200 OK 425 ms 410 B
[] JSON Preview Visualize
1 [
2   {
3     "id": 6,
4     "name": "Clases"
5   },
6   {
7     "id": 4,
8     "name": "General"
9   },
10  {
11    "id": 3,
12    "name": "Presentaciones"
13  },
14  {
15    "id": 1,
16    "name": "Tesis"
  
```

- Subida de archivos

Fig 10. Operación Post para subir archivos

The screenshot shows a POST request to `http://35.223.73.122/api/upload`. The request body is set to `form-data` and contains three fields: `file` (selected), `user_id` (Text, value 5), and `category_id` (Text, value 3). The response status is `200 OK` with a response time of 1.36 seconds and a response size of 362 B. The JSON response is:

```

1 { "message": "Archivo subido correctamente", "filename": "1765851268373-ExamenIA.html", "db_id": 15 }

```

- Visualización de archivos

Fig 11. Operación Get para obtener metadatos

The screenshot shows a GET request to `http://35.223.73.122/api/resources`. The response status is `200 OK` with a response time of 0.00 seconds and a response size of 1.36 B. The JSON response is:

```

1 [
2   {
3     "id": 1,
4     "user_id": 1,
5     "category_id": 1,
6     "title": "tesis.txt",
7     "file_url": "/mnt/storage/1765599311923-tesis.txt",
8     "content_type": "text/plain",
9     "created_at": "2025-12-13T04:15:11.972Z"
10   },
11   {
12     "id": 2,
13     "user_id": 1,
14     "category_id": 2,
15     "title": "Video de Prueba",

```

3. Capa de Datos

Se describe la elección, el rol y la ubicación estratégica de la capa de datos en nuestra arquitectura. Se optó por utilizar PostgreSQL como motor de base de datos relacional para la gestión de usuarios, metadatos de los recursos y de logs generados por el sistema al momento de que un usuario sube un archivo.

La base de datos se implementó como una instancia externa dedicada, separada de las instancias de la API. Con esto logramos una separación de responsabilidades, ya que no comparte recursos con las aplicaciones frontend y backend. La instancia se encuentra en la subred privada y el acceso está controlado por las reglas de Grupo de Seguridad, permitiendo conexiones sólo desde las instancias API por el puerto 5432.

Fig 12. Configuración de la Instancia

✓ virtualizacion-db

PostgreSQL 17

Resumen	Redes	Seguridad	Pruebas de conectividad
Redes			
Nombre de la conexión			
	virtualizacion-481016:us-central1:virtualizacion-db		🔗
Conectividad de IP privada	Habilitado		
Redes asociadas	projects/virtualizacion-481016/global/networks/red-practica		🔗
Red	red-practica		
Método de conexión del servicio	Acceso privado a servicios		
Rango de IP asignada	Rango de IP asignado automáticamente		
Dirección IP interna	10.7.112.3		
Conectividad de IP pública	Inhabilitado		

Fig 13. Configuración de la Base de Datos y Usuario

✓ virtualizacion-db

PostgreSQL 17

+ Crear base de datos

Nombre ↑	Intercalación	Grupo de caracteres
db-virtualizacion	en_US.UTF8	UTF8

✓ virtualizacion-db

cutable SQL 17

Las cuentas de usuario permiten que los usuarios y las aplicaciones accedan a la base de datos.

+ Agregar cuenta de usuario

Usuarios agregados

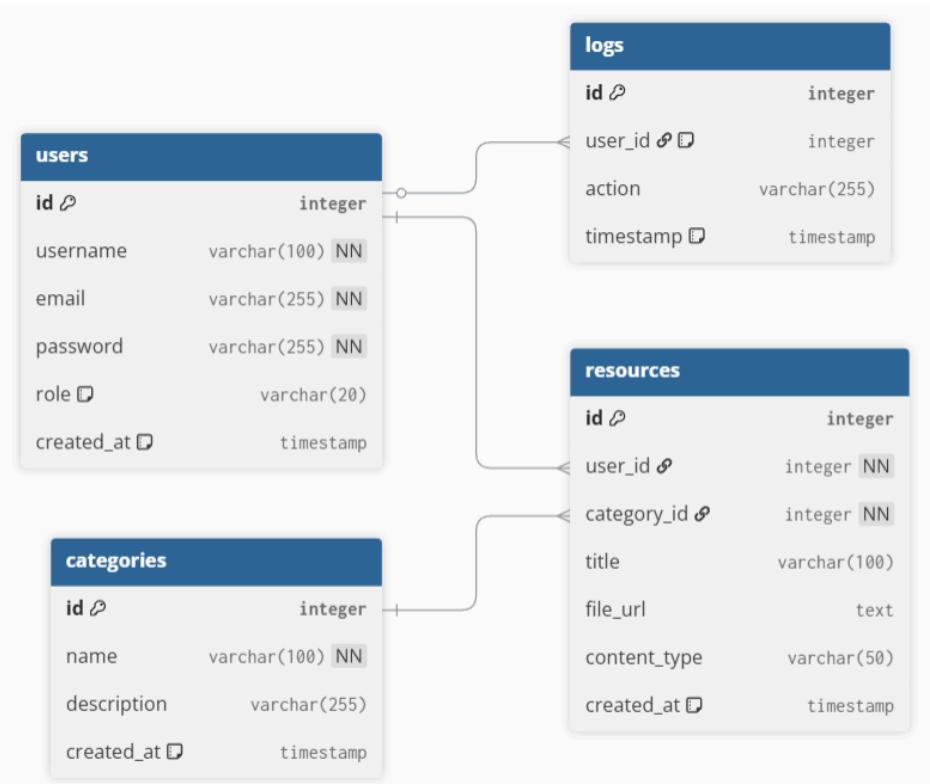
Miembros de grupos de IAM autorizados

Esas son cuentas a las que les otorgaste acceso a la instancia usada.

	Nombre de usuario ↑	Autenticación
app_user	Integrado	

La base de datos cuenta con tres tablas para guardar usuarios, los recursos multimedia y la categoría de dichos de recursos. La tabla de recursos no almacena los archivos binarios, almacena la URL de ubicación del archivo dentro del Storage Service (Filestore).

Fig 14. Modelo Relacional



Las instancias de API se conectan a la base por medio de la red interna (Subred Privada) utilizando el puerto 5432 para realizar las operaciones de lectura y escritura.

Fig 15. Conexión API a la base de datos

```
// 1. CONEXIÓN A POSTGRESQL
const pool = new Pool({
  host: '10.7.112.3',
  user: 'app_user',
  password: '#Practica123',
  database: 'db-virtualizacion',
  port: 5432,
  ssl: { rejectUnauthorized: false }
});
```

4. Integración con almacenamiento compartido

Para lograr una integración con el almacenamiento compartido se usó la librería multer para utilizar una ruta absoluta ‘mnt/storage’ como destino de escritura, esta no es una ruta local, sino un punto de montaje que apunta la instancia de Filestore. A continuación se presentan imágenes que describen la configuración principal del almacenamiento compartido.

Fig 16. Configuración de multer

```
const storagePath = '/mnt/storage';
const storage = multer.diskStorage({
  destination: (req, file, cb) => cb(null, storagePath),
  filename: (req, file, cb) => cb(null, Date.now() + '-' + file.originalname)
});
const upload = multer({ storage: storage });
```

Fig 17. Configuración de montaje persistente

```
LABEL=cloudimg-rootfs   /          ext4    discard,commit=30,errors=remount-ro  0 1
LABEL=BOOT     /boot      ext4    defaults          0 2
LABEL=UEFI    /boot/efi   vfat    umask=0077        0 1
10.207.0.18:/archivos /mnt/storage nfs defaults 0 0
```

Fig 18. Vinculación de la carpeta al servicio Filestore

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/root	8.7G	3.9G	4.8G	45%	/
tmpfs	2.0G	0	2.0G	0%	/dev/shm
tmpfs	783M	1.1M	782M	1%	/run
tmpfs	5.0M	0	5.0M	0%	/run/lock
efivars	56K	24K	27K	48%	/sys/firmware/efi/efivars
/dev/sda16	881M	65M	755M	8%	/boot
/dev/sda15	105M	6.2M	99M	6%	/boot/efi
10.207.0.18:/archivos	1007G	2.0M	956G	1%	/mnt/storage
tmpfs	392M	12K	392M	1%	/run/user/1002
tmpfs	392M	12K	392M	1%	/run/user/1001

Finalmente, comprobaremos que el archivo subido en la sección anterior (Fig. 4) se encuentra en la carpeta ‘/mnt/storage’, para esto hacemos uso del comando ‘ls -l /mnt/storage’, donde podemos observar que el archivo ‘ExamenIA.html’ se encuentra dentro de la carpeta.

Fig 19. Salida recortada del comando ‘ls’

```
-rw-r--r-- 1 encaladaandres15 encaladaandres15 406776 Dec 16 02:14 1765851268373-ExamenIA.html
```

Debido a que el balanceador de carga usa una plantilla de instancia para crear nuevas máquinas, todas tendrán la misma configuración. Con estas pruebas, hemos comprobado la integración con el servicio de almacenamiento compartido y la consistencia entre los nodos.

5. Pruebas de conectividad integral

Para concluir la parte 2, realizaremos 3 pruebas finales que comprobarán la funcionalidad y la integración de los servicios.

- La UI consume la API correctamente

Al pulsar el botón de iniciar sesión, podemos visualizar en la consola como la petición tiene un código 200, lo que significa que la interfaz se ha comunicado exitosamente con el servidor y ha recibido una respuesta.

Fig 20. Código 200 en la consola del navegador

▼ General	
Request URL	http://35.223.73.122/api/login
Request Method	POST
Status Code	● 200 OK
Remote Address	35.223.73.122:80

- La API accede a la base de datos sin problemas

El endpoint para obtener los datos de los archivos funciona correctamente, devolviendo un JSON con esta información.

Fig 21. Información obtenida por la API

```
← ⌂ ▲ No seguro 35.223.73.122/api/resources
Impresión con sangría ✓

[{"id": 1, "user_id": 1, "category_id": 1, "title": "tesis.txt", "file_url": "/mnt/storage/1765599311923-tesis.txt", "content_type": "text/plain", "created_at": "2025-12-13T04:15:11.972Z"}]
```

- El backend puede guardar y recuperar archivos desde el almacenamiento compartido

Al consultar la carpeta donde fue montado el servicio Filestore, observamos todos los archivos que han sido guardados en el servicio, los archivos ‘thesis.txt’ fueron archivos que se colocaron directamente para realizar pruebas, por lo que no presentan usuario.

Fig 22. Salida del comando ls

```
encaladaandres15@grupo-backend-g2nw:~$ ls -l /mnt/storage
total 1844
-rwxrwxrwx 1 root          root          20 Dec 13 01:46 1765590384596-test.txt
-rwxrwxrwx 1 root          root          28 Dec 13 04:15 1765599311923-tesis.txt
-rwxrwxrwx 1 encaladaandres15 encaladaandres15 406776 Dec 14 20:12 1765743137442-ExamenIA.html
-rwxrwxrwx 1 encaladaandres15 encaladaandres15 225091 Dec 14 20:28 1765744128231-Investigacion virtualizacion.pdf
-rwxrwxrwx 1 encaladaandres15 encaladaandres15 32794 Dec 14 20:31 1765744283885-Investigacion virtualizacion.docx
-rwxrwxrwx 1 encaladaandres15 encaladaandres15 225091 Dec 14 20:33 1765744385408-Investigacion virtualizacion.pdf
-rwxrwxrwx 1 encaladaandres15 encaladaandres15 164244 Dec 14 20:37 1765744670850-Captura de pantalla 2025-11-11 014737.png
-rwxrwxrwx 1 encaladaandres15 encaladaandres15 17746 Dec 14 20:43 1765745000987-Captura de pantalla 2025-09-07 231838.png
-rw-r--r-- 1 encaladaandres15 encaladaandres15 408234 Dec 15 00:54 1765760074484-Prototipol.html
-rw-r--r-- 1 encaladaandres15 encaladaandres15 152848 Dec 15 02:34 1765766090415-VirtualizaciÃ³n_Conceptos, implementaciÃ³n y aplicaciones.pdf
drwxrwxrwx 2 root          root          16384 Dec 13 00:02 lost+found
-rwxrwxrwx 1 root          root          211706 Dec 13 04:41 video.mp4
```

Parte 3: Escalabilidad y balanceo

1. Balanceador de carga

En la siguiente Figura se muestra el balanceador de carga, junto con su IP y su puerto, además, podemos observar el grupo de instancias, el cual contiene una instancia (debido al bajo uso).

Se implementó un mecanismo de Health Check (Sondeo de estado) configurado en el protocolo TCP puerto 3000. Este servicio verifica periódicamente la disponibilidad de la API en cada instancia. Si una instancia falla o deja de responder, el balanceador la marca como 'No saludable' y deja de enviarle tráfico automáticamente hasta que se recupere.

Fig 23. Detalles del balanceador

lb-practica

Balanceador de cargas de aplicaciones externo regional

Región
us-central1

Frontend

Protocolo ↑	IP-Puerto	Nivel de red	Certificado	Política de SSL	Tiempo de espera keepalive
HTTP	35.223.73.122:80	Premium	-		610 segundos

Reglas de host y ruta

Hosts ↑	Rutas	Backend
Todos los que no coincidan (predeterminado)	Todos los que no coincidan (predeterminado)	servicio-backend-api

Backend

Servicios de backend

1. servicio-backend-api

Protocolo de extremo	HTTP
Puerto con nombre	http
Tiempo de espera	30 segundos
Política de selección de direcciones IP <small>(?)</small>	Solo IPv4
Verificación de estado	verificador-api
Registros	Inhabilitada
Política de seguridad del backend	Ninguna

[▼ Mostrar opciones avanzadas](#)

Backends

Nombre ↑	Tipo	Tipo de pila de IP	Alcance	En buen estado	Ajuste de escala automático
grupo-backend	Grupo de instancias	IPv4	us-central1-a	✓ 1 de 1	Activado: Objetivo Uso de CPU 60%

2. Autoscaling

Se configuró una política de autoescalado basada en métricas de rendimiento del sistema. Se definió un umbral de CPU del 60%: si el promedio de uso de CPU del grupo supera este valor, el orquestador aprovisiona nuevas instancias automáticamente. Se establecieron límites estrictos (Mín: 1, Máx: 3) para controlar el presupuesto y garantizar la disponibilidad mínima.

Fig 24. Detalles de autoescalado

Ajuste de escala automático

Modo de escalado automático	Activo
Mínimo # instancias	1
Máximo # instancias	3
Periodo de inicialización	60 segundos
Ajuste de escala automática indicador	
Uso de CPU	60%
Ajuste de escala automática predictivo	Desactivado
Controles de reducción de escala	Inactivo
Escalamiento de programas	Administrar programas

• Creación de componentes

Para crear un balanceador de carga (donde se configura el autoescalado) debemos dirigirnos a la sección ‘Servicios de red’, donde encontraremos la pestaña balanceador de cargas, al crear un nuevo balanceador podremos definir distintos parámetros, como son el tipo (aplicaciones o red), orientación (público o uso interno), implementación (global o regional), se requerirán diferentes parámetros en función de la configuración elegida. A continuación, podremos definir varias configuraciones más, como frontends, backends y la red a la que pertenece el balanceador.

Fig 25. Panel principal de la sección Servicios de red

Fig 26. Panel de configuración del balanceador de carga

3. Pruebas de rendimiento

Finalmente, se realizaron pruebas para comprobar el funcionamiento del balanceador de carga. Primero, se observaron los datos del sistema cuando estaba en un estado tranquilo, los cuales pueden ser visualizados en el panel del grupo de instancias, estos datos pueden ser observados en la Fig. 15.

Fig 27. Panel del grupo de instancias



Posteriormente, se utilizó una herramienta llamada ‘stress-ng’ para saturar intencionalmente los núcleos de la instancia, logrando superar el umbral del 60%. A lo que el balanceador respondió creando más instancias, este cambio en el uso del CPU y el número de instancias puede ser visualizado en las siguientes figuras.

Fig 28. Gráfica de uso del CPU

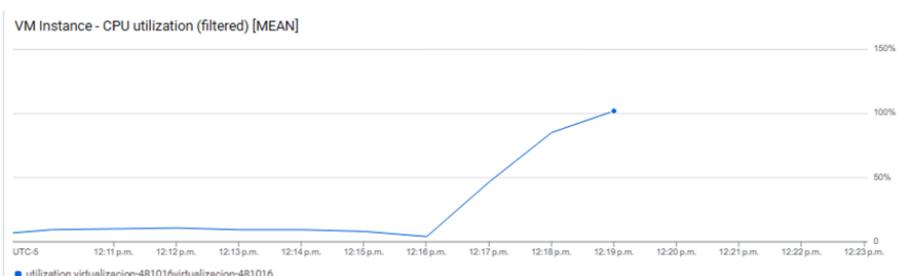
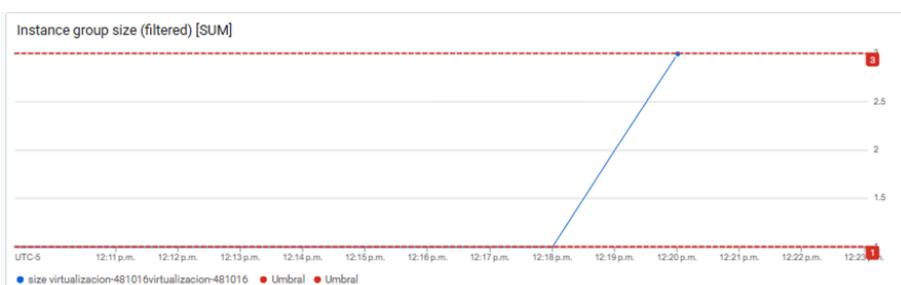


Fig 29. Gráfica del número de instancias



Nota. Debido a la simplicidad del proyecto se estableció un límite de solo 3 instancias.

Fig 30. Panel de instancias del grupo de instancias

Instancias de VM							
		Suspender	Detener	Iniciar/Rearanudar	Quitar del grupo	Borrar	
Filtro Escribir el nombre o valor de la propiedad							
<input type="checkbox"/> Estado	Nombre ↑	Fecha/hora de creación	Plantilla	Configuración por instancia	IP Interna	IP externa	
<input checked="" type="checkbox"/>	grupo-backend-4lcl	dic 13, 2025, 12:10:18 a.m. UTC-05:00	plantilla-api (Regional)		10.0.2.3 (nic0)	34.71.235.92	
<input checked="" type="checkbox"/>	grupo-backend-7sfm	dic 13, 2025, 12:17:28 p.m. UTC-05:00	plantilla-api (Regional)		10.0.2.4 (nic0)	34.57.131.253	
<input checked="" type="checkbox"/>	grupo-backend-bzg5	dic 13, 2025, 12:17:46 p.m. UTC-05:00	plantilla-api (Regional)		10.0.2.5 (nic0)	34.133.237.179	

Una vez el estresor es detenido, el uso del CPU baja y el balanceador comienza a eliminar instancias. Además, se observó un cambio en las gráficas del panel del balanceador, donde se observa un pico en la gráfica de uso del CPU.

Fig 31. Detalles de instancias activas

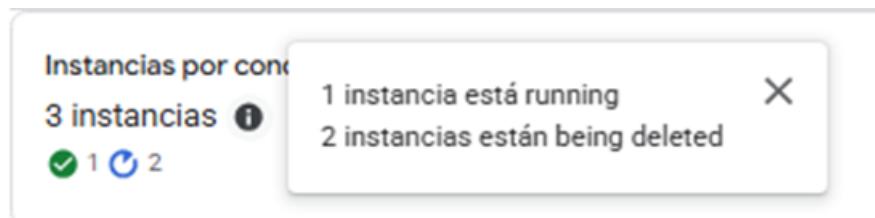


Fig 32. Gráficas de estadísticas



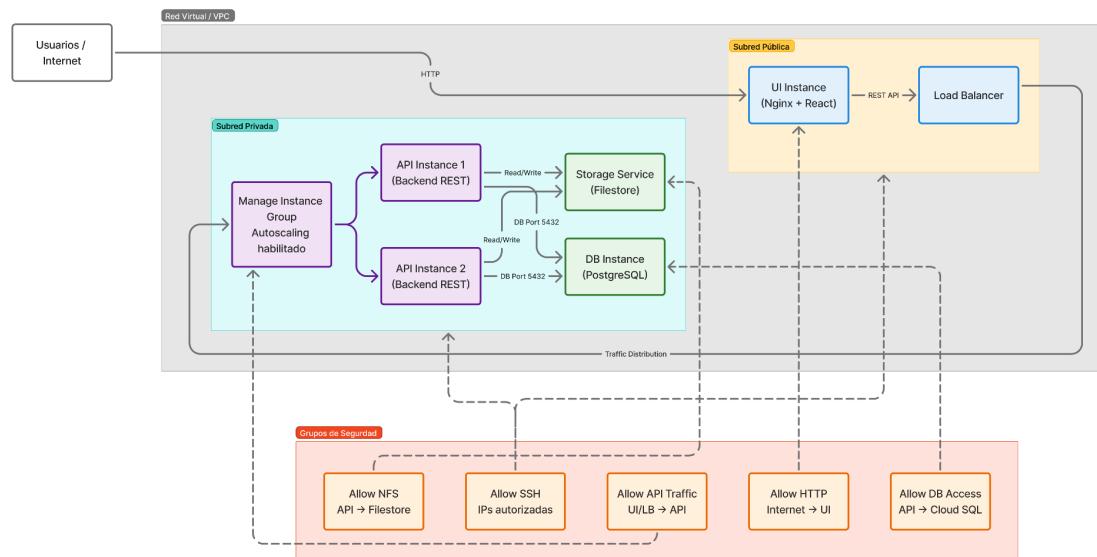
Finalmente, mediante PowerShell, se realizó una medida del tiempo de respuesta del sistema, donde se obtuvo un tiempo de 0.287604 segundos.

Parte 4: Documentación

1. Diagrama Arquitectónico

En el siguiente diagrama se presenta la arquitectura del proyecto, resaltando elementos de redes, instancias, el balanceador de carga y el servicio de almacenamiento compartido.

Fig 33. Diagrama arquitectónico



2. Evidencias

Se creó un repositorio en github, donde se presentan los archivos principales del proyecto. Se separó el repositorio en carpetas para cada elemento (Backend y Frontend). Enlace al repositorio: <https://github.com/AndresEncalada/Virtualizaci-n-en-la-Nube>

3. Resultados obtenidos

Los resultados obtenidos demostraron que el Grupo de Instancias responde eficazmente ante picos de demanda, activando el autoescalado horizontal para duplicar la capacidad de cómputo al superar el umbral de CPU del 60%, mientras que el Balanceador de Carga mantuvo la continuidad del servicio distribuyendo el tráfico sin interrupciones. Asimismo, se confirmó la consistencia integral de los datos, donde la sincronización entre Cloud SQL y el sistema de archivos compartido Filestore (NFS) aseguró que la información transaccional y los archivos multimedia fueran accesibles y persistentes desde cualquier instancia del clúster