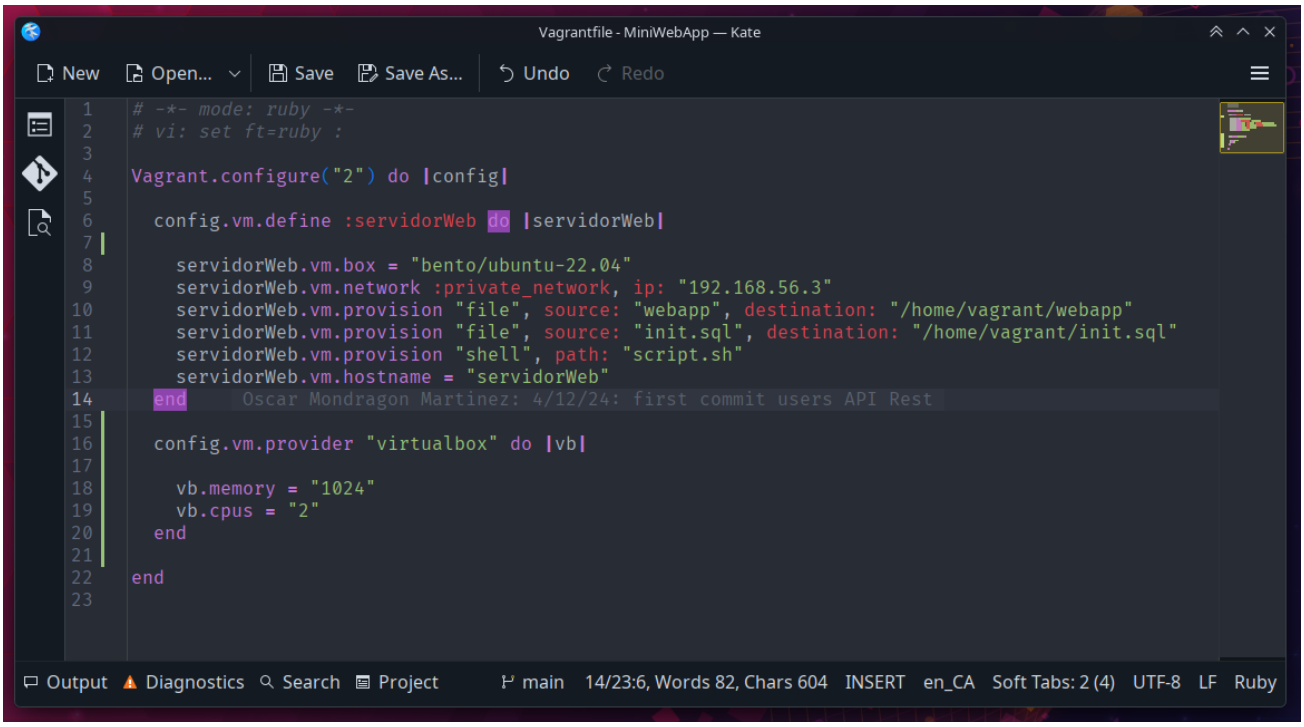
uao	**UNIVERSIDAD AUTÓNOMA DE OCCIDENTE**				
	FACULTAD DE INGENIERÍA		**NOMBRE DE LA ASIGNATURA**	***Servicios Telemáticos***	
	CÓDIGO:	2195889	**NOMBRE:**	*Isabella Castañeda & Camilo Franco Moya*	***Valoración***
EXAMEN FINAL				**FECHA SUSTENTACIÓN: martes, noviembre 18 de 2025**	

Caso de Estudio: Despliegue Seguro, Monitoreo y Visualización de una Aplicación Web en la Nube
La empresa ficticia CloudNova desea migrar su aplicación web de desarrollo a un entorno de producción seguro y monitoreado. Su objetivo es garantizar disponibilidad, seguridad y visibilidad del rendimiento mediante herramientas de código abierto y servicios en la nube. Usted ha sido contratado como ingeniero DevOps para liderar este proceso.

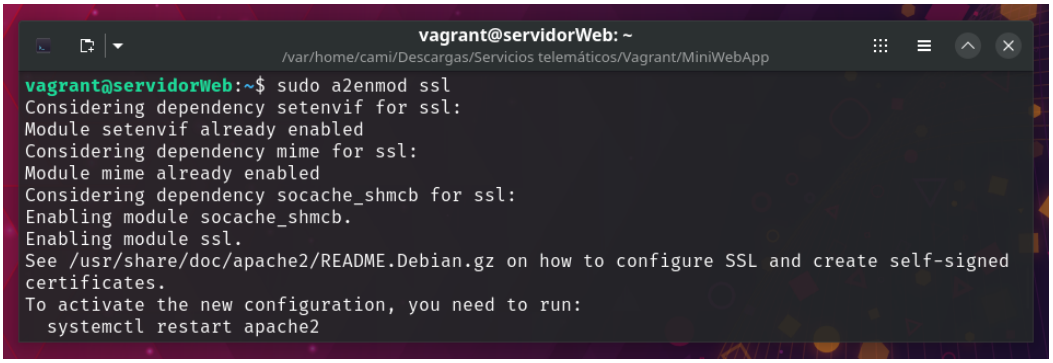
1. **Empaquetado y despliegue local con Docker (1.5 puntos)**
   - Clone el repositorio MiniWebApp en GitHub proporcionado en clase.
   - Configure un servidor web (Apache o Nginx) para servir la aplicación mediante HTTPS, usando un certificado SSL válido.
   - Asegure la redirección automática de HTTP a HTTPS.
   - Cree un Dockerfile y un docker-compose.yml para orquestar la aplicación.
   - Verifique el funcionamiento accediendo desde un navegador local.

   Recursos sugeridos (Sin garantía ni soporte)
   Asegurar Apache con SSL en Docker

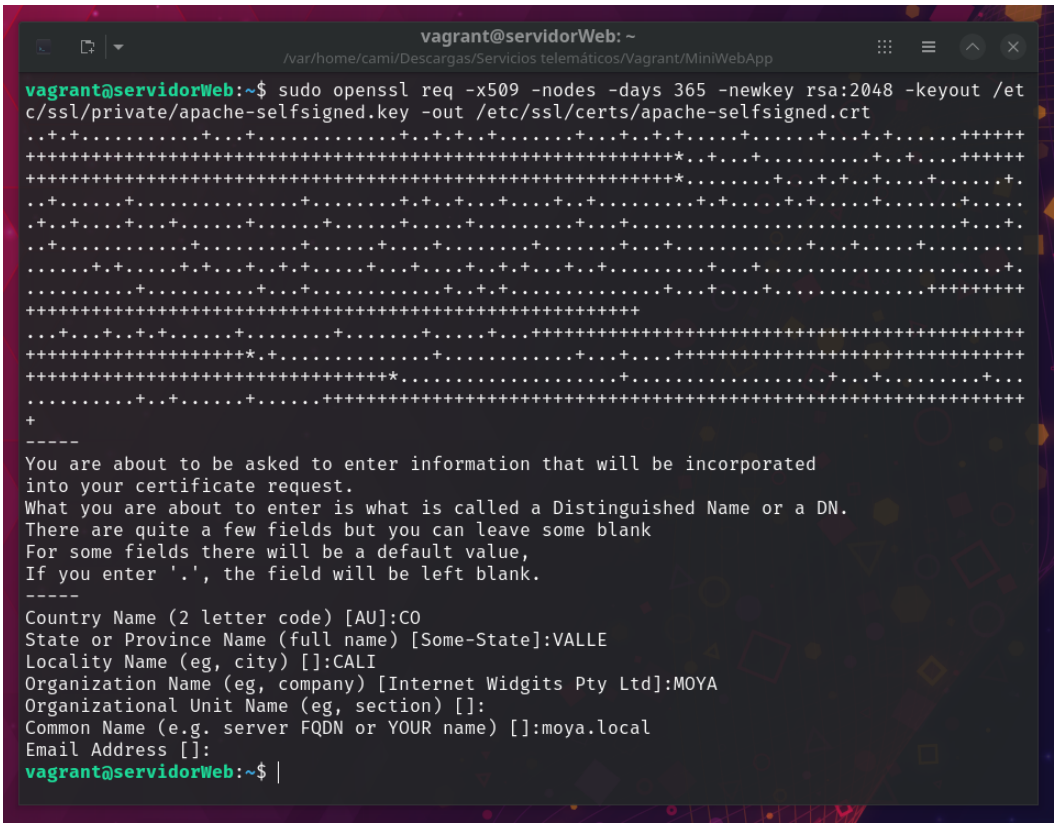



```
$ sudo apt update
$ sudo apt install apache2 openssl ca-certificates curl libapache2-mod-wsgi-py3

$ sudo install -m 0755 -d /etc/apt/keyrings
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
$ sudo chmod a+r /etc/apt/keyrings/docker.asc
```

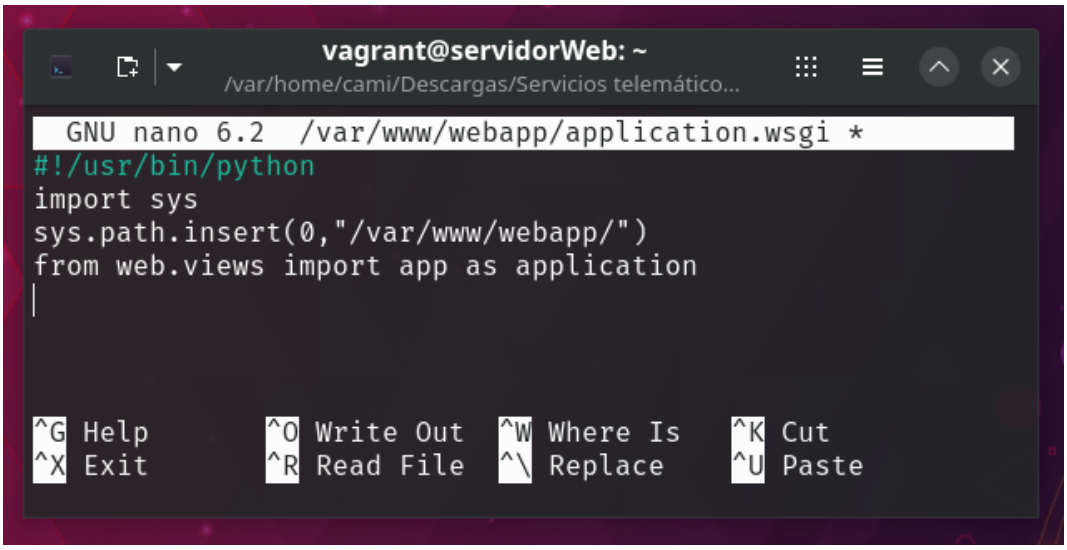
```
$ echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
$ sudo apt update
$ sudo apt install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

$ sudo usermod -aG docker $USER
$ sudo cp -r ./webapp /var/www/

$ sudo a2enmod ssl rewrite
```



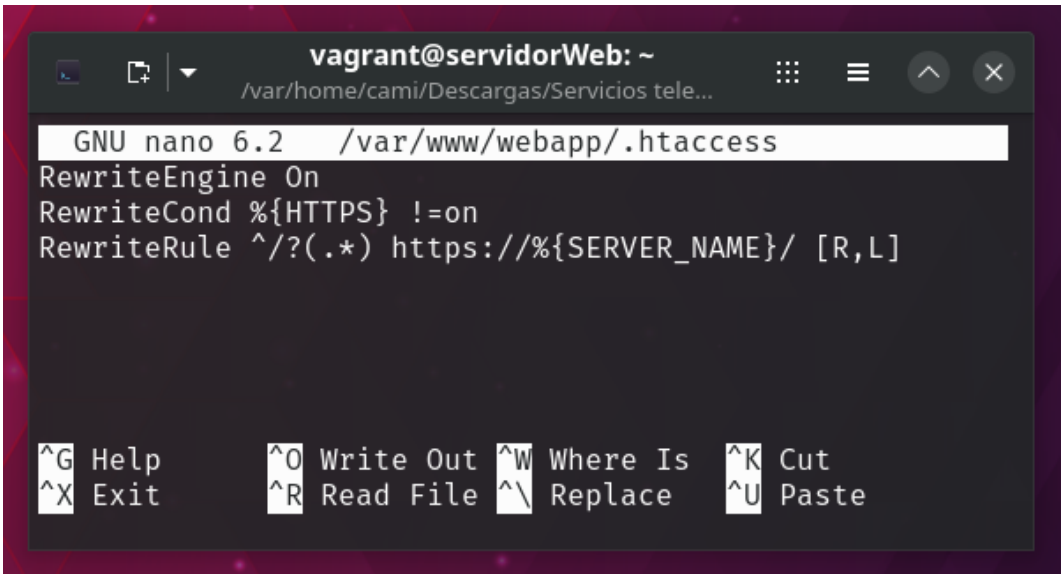

```
$ sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/apache-selfsigned.key -out /etc/ssl/certs/apache-selfsigned.crt
```



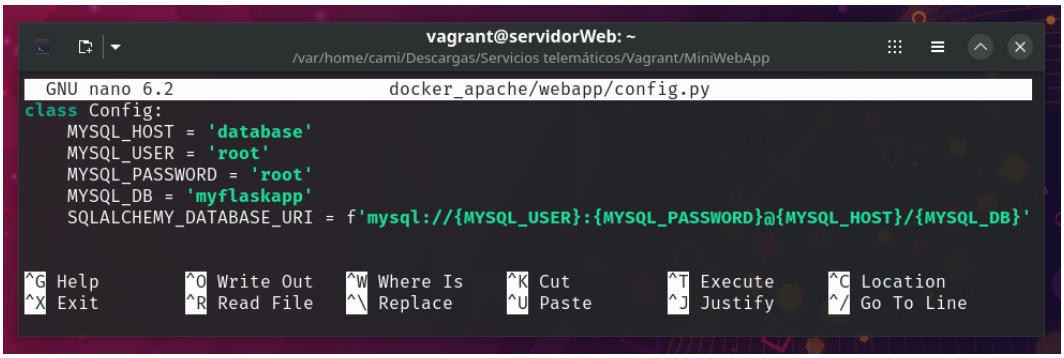
/var/www/webapp/application.wsgi



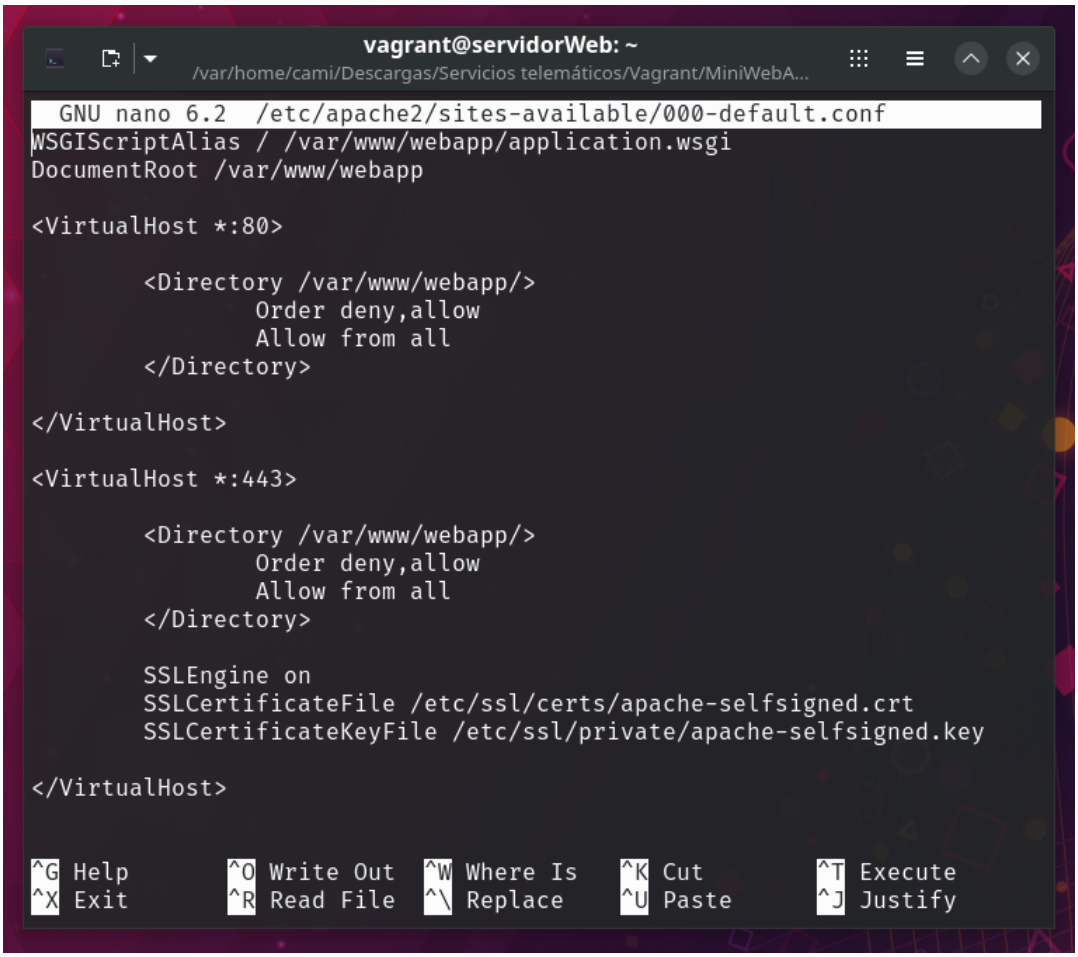
/var/www/webapp/.htaccess



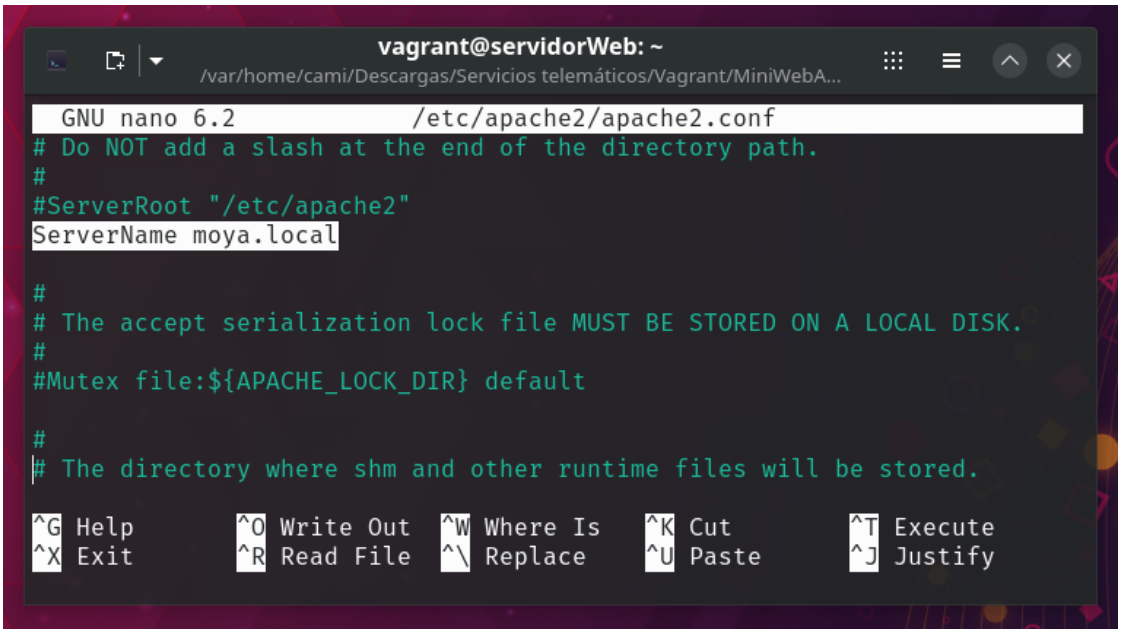
/var/www/webapp/config.py

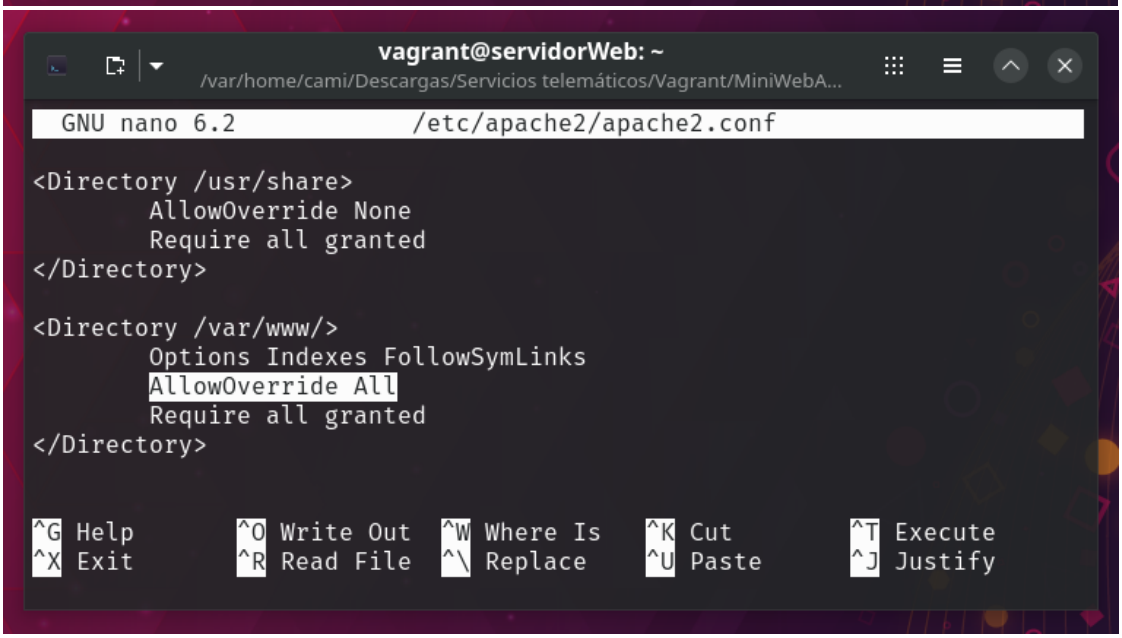


/etc/apache2/sites-available/000-default.conf



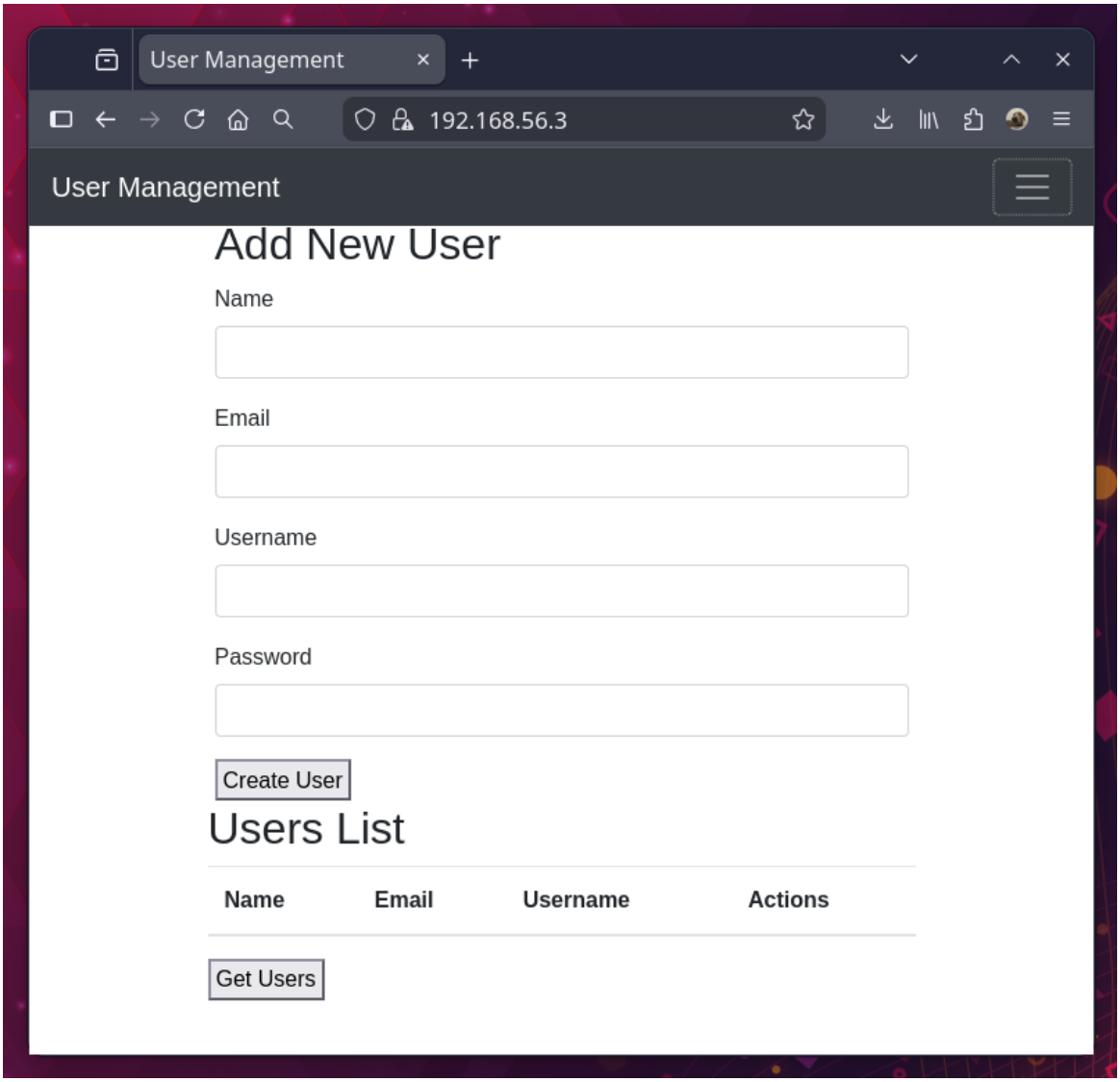
/etc/apache2/apache2.conf






```
$ sudo apache2ctl configtest
```

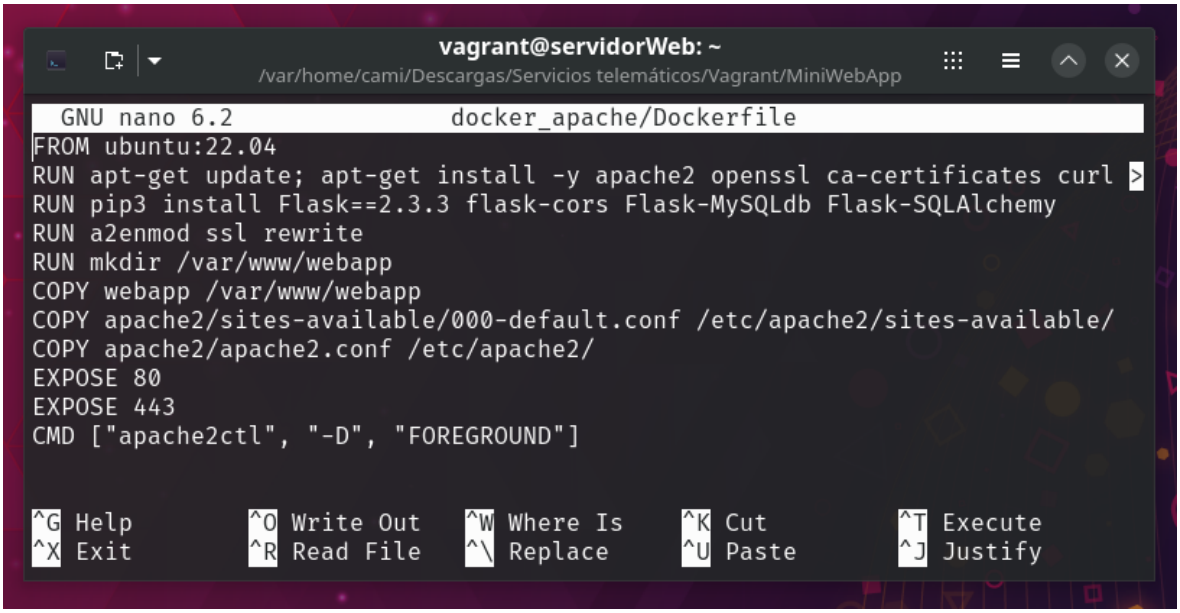
```
$ sudo systemctl restart apache2
```

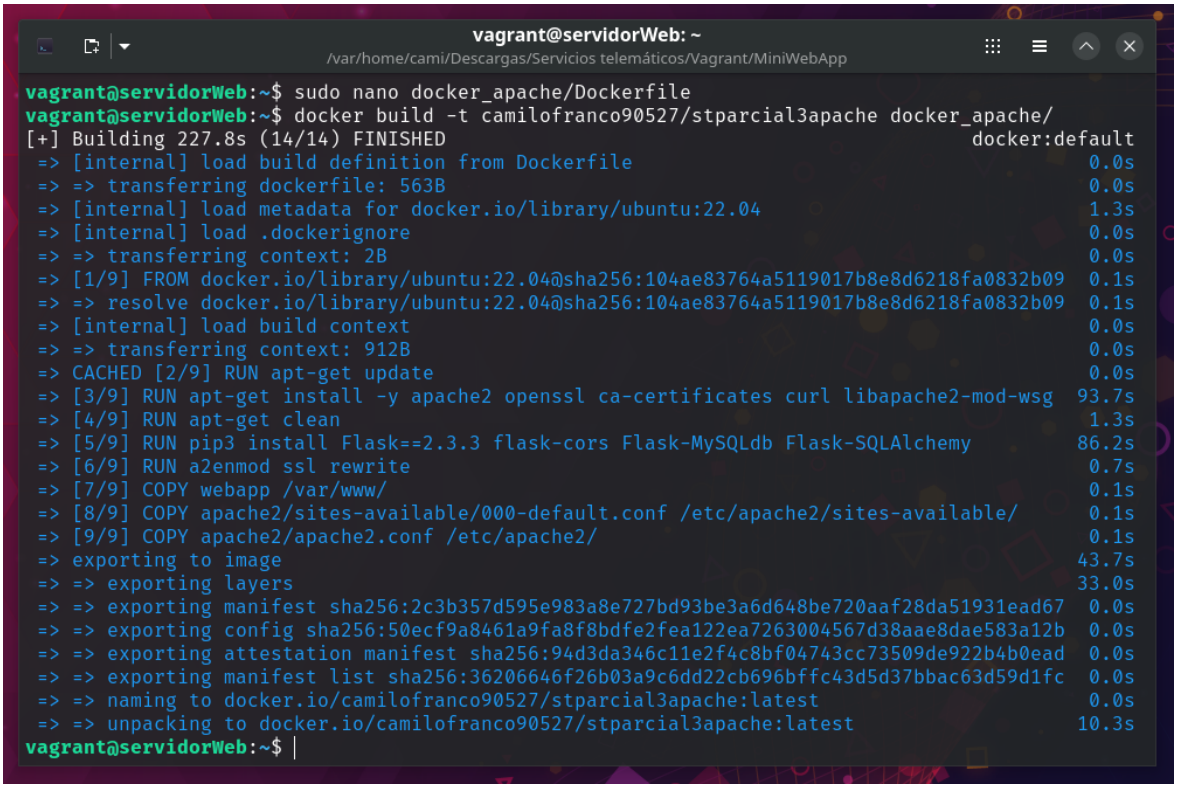



```
$ mkdir docker_apache

$ mkdir -p docker_compose/db

$ mkdir -p docker_compose/ssl_private

$ mkdir -p docker_compose/ssl_certs


$ sudo cp -r /var/www/webapp docker_apache/

$ sudo cp -r /etc/apache2/sites-available/000-default.conf docker_apache/

$ sudo cp -r /etc/apache2/apache2.conf docker_apache/
```

```
$ sudo cp init.sql docker_compose/db/

$ sudo cp /etc/ssl/private/apache-selfsigned.key docker_compose/ssl_private/

$ sudo cp /etc/ssl/certs/apache-selfsigned.crt docker_compose/ssl_certs/
```

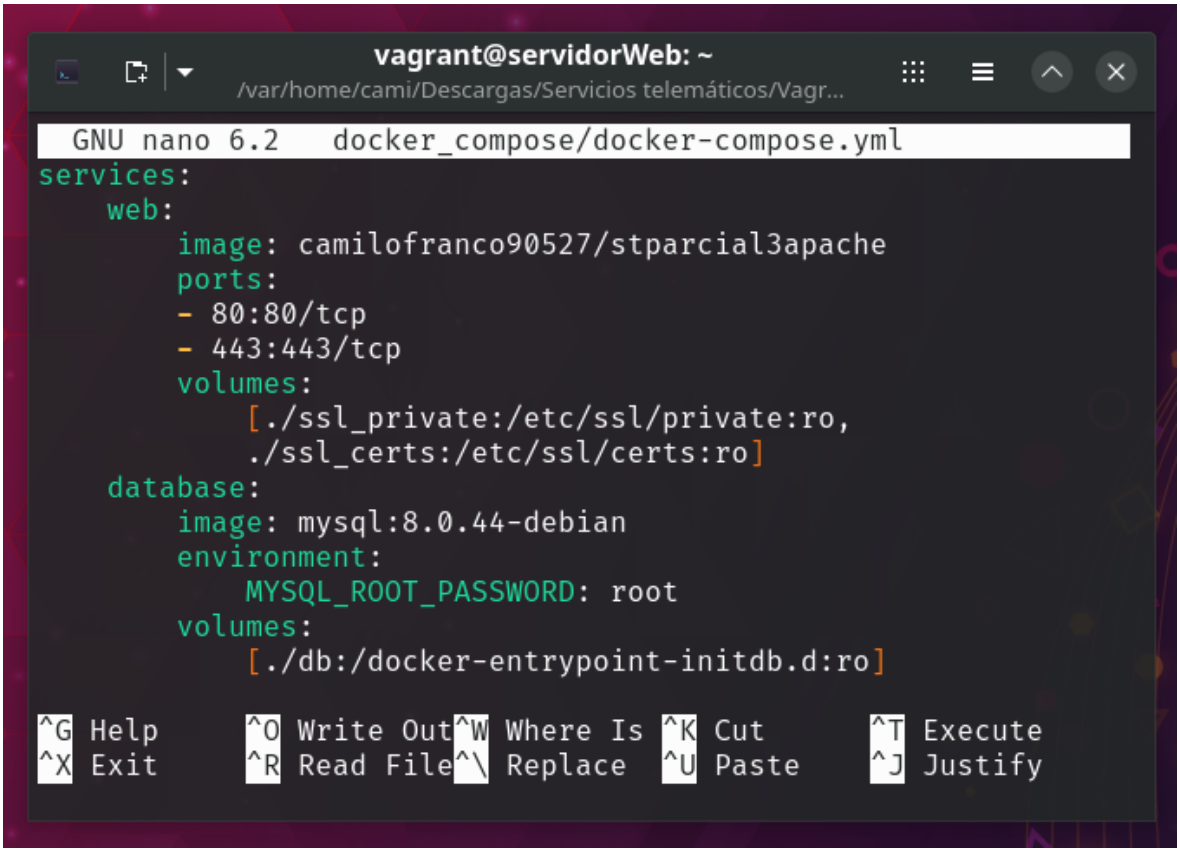
~/docker_apache/Dockerfile

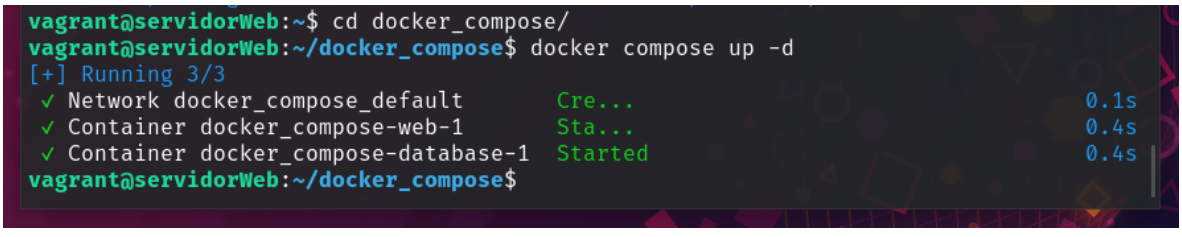




```
$ docker build -t camilofranco90527/stparcial3apache docker_apache/
```

~/docker_compose/docker-compose.yml

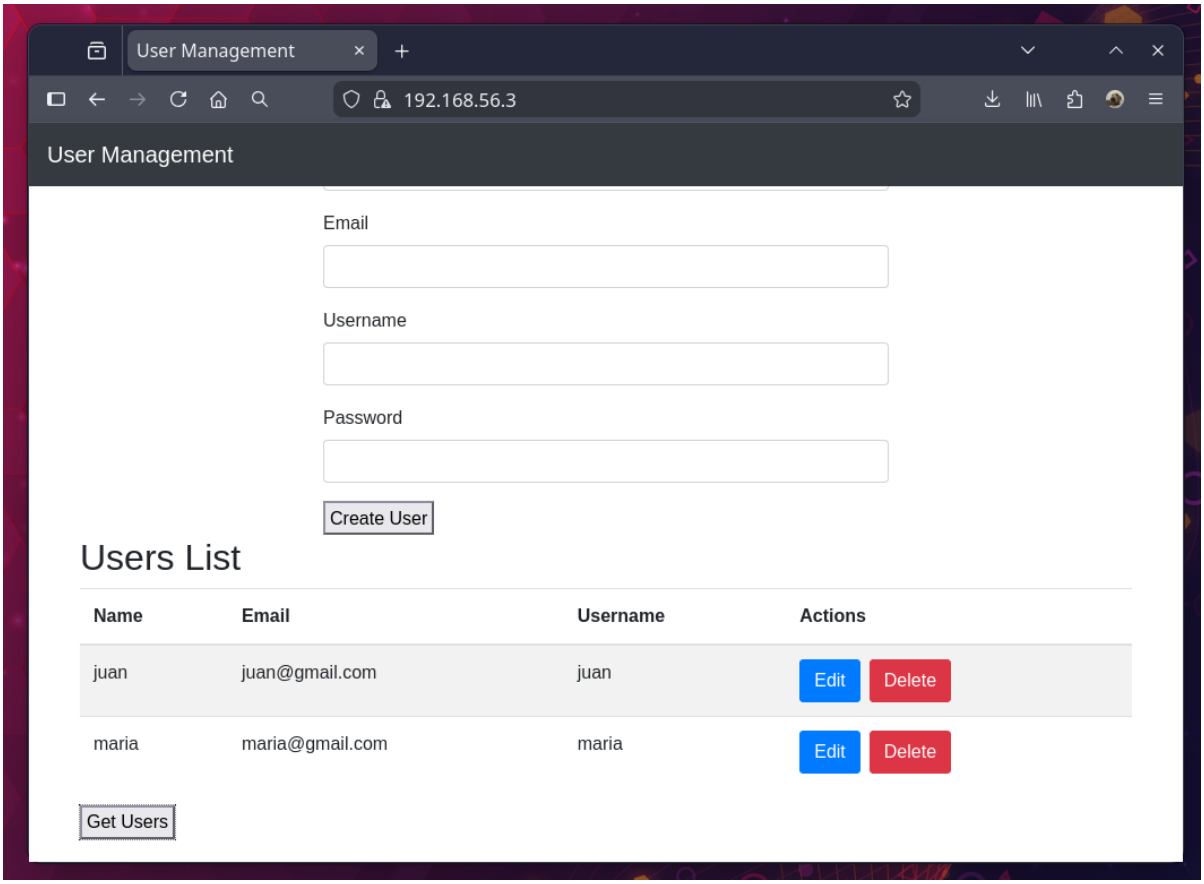





```
$ sudo systemctl stop apache2
```

```
$ sudo systemctl stop mysql
```

```
$ docker compose up -d
```

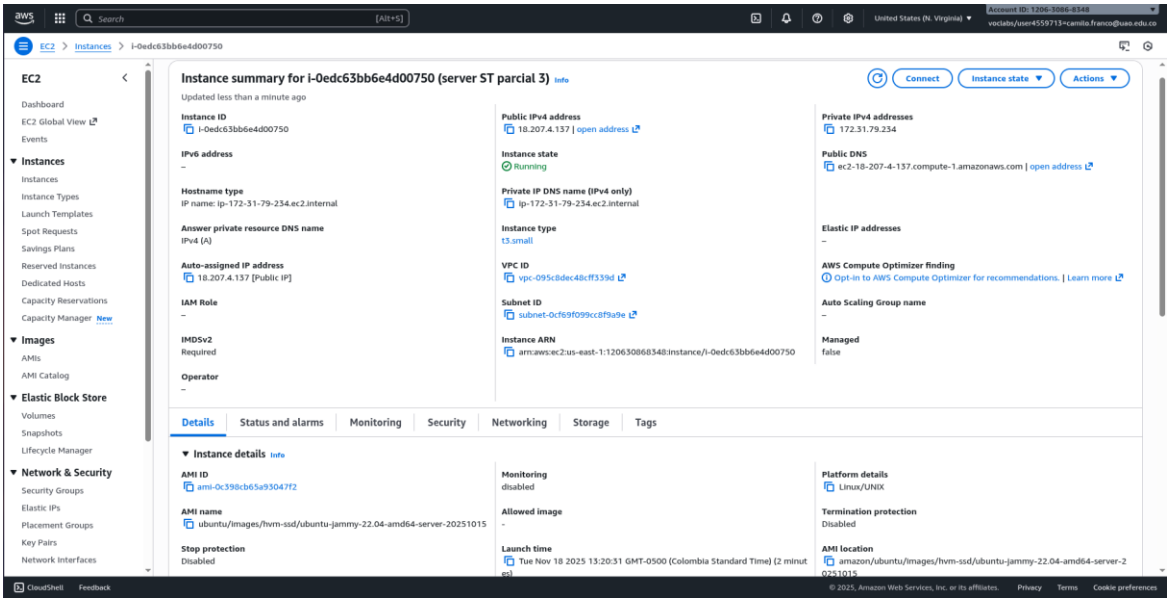


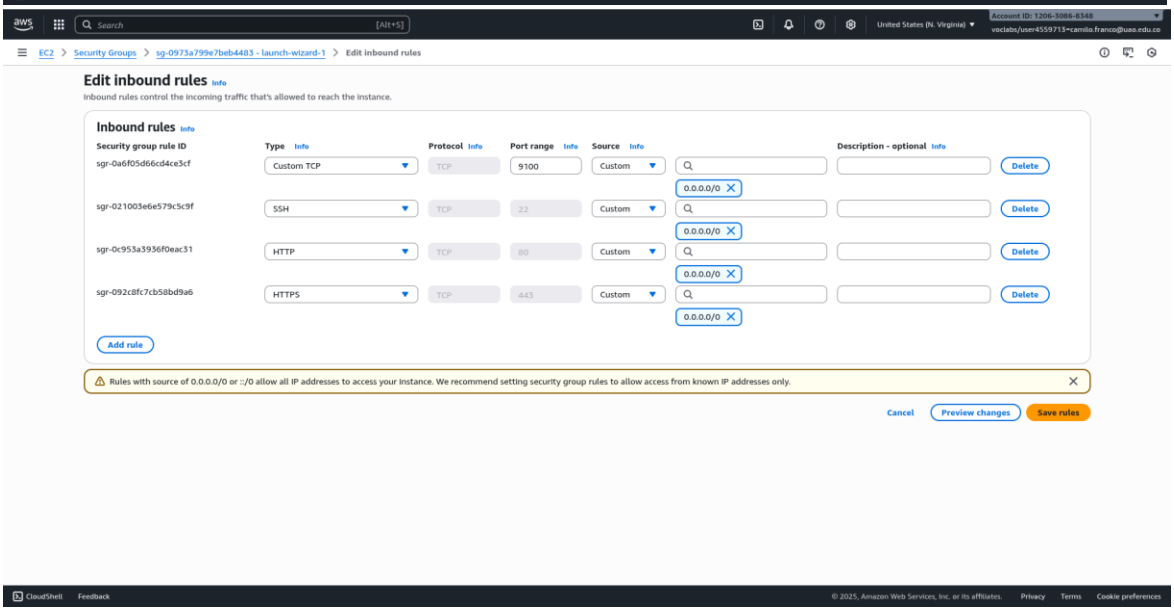

```
Login Succeeded
vagrant@servidorWeb:~/docker_compose$ docker tag camilofranco90527/stparcial3apache c
amilofranco90527/stparcial3apache:v1
vagrant@servidorWeb:~/docker_compose$ docker push camilofranco90527/stparcial3apache:
v1
The push refers to repository [docker.io/camilofranco90527/stparcial3apache]
7e49dc6156b0: Pushed
ba187f87d45b: Pushed
4ca3b651e9d8: Pushed
2225fb99aca9: Pushed
dd21f26033f4: Pushed
08df169ddd2b: Pushed
b1164e41f01c: Pushed
8e7de48aa835: Pushed
678445a777bf: Pushed
v1: digest: sha256:33fc030c1ed77473fc58db418fc4666ead475a86d677e7862cb37c9694ff37e7 s
ize: 856
vagrant@servidorWeb:~/docker_compose$
```

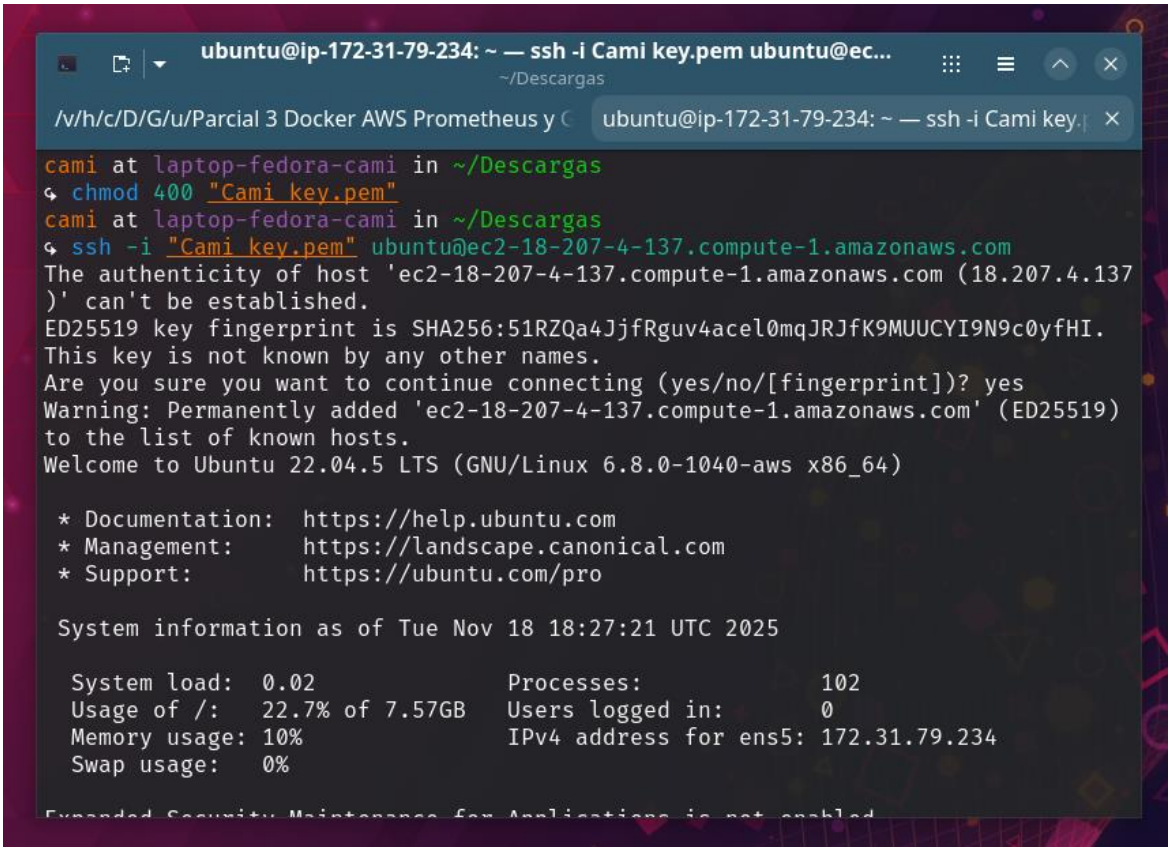
```
$ docker login

$ docker tag camilofranco90527/stparcial3apache camilofranco90527/stparcial3apache:v1

$ docker push camilofranco90527/stparcial3apache:v1
```

2. Despliegue en la nube con AWS EC2 (1.0 punto)

- Inicie una instancia EC2 en AWS y configure las reglas de seguridad para permitir tráfico HTTP/HTTPS.
- Instale Docker y ejecute el contenedor de la aplicación usando Docker Compose.
- Verifique el acceso remoto a la aplicación a través de la IP pública.

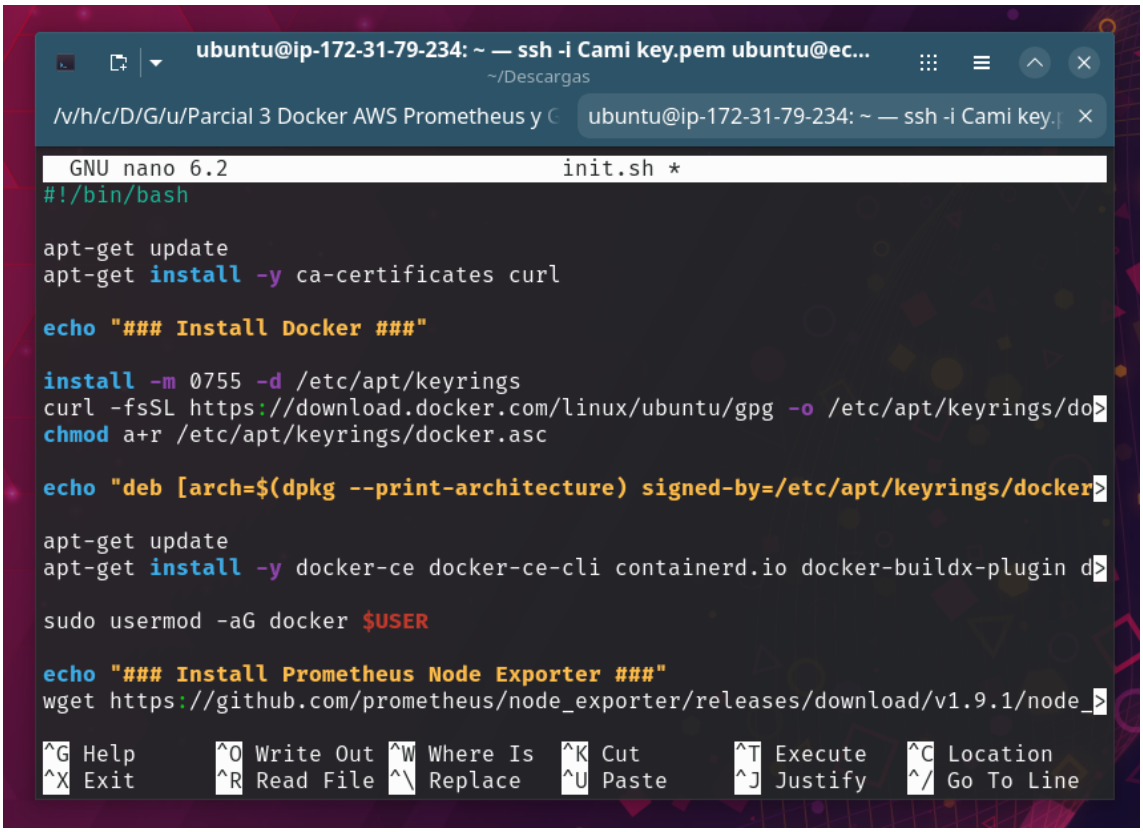


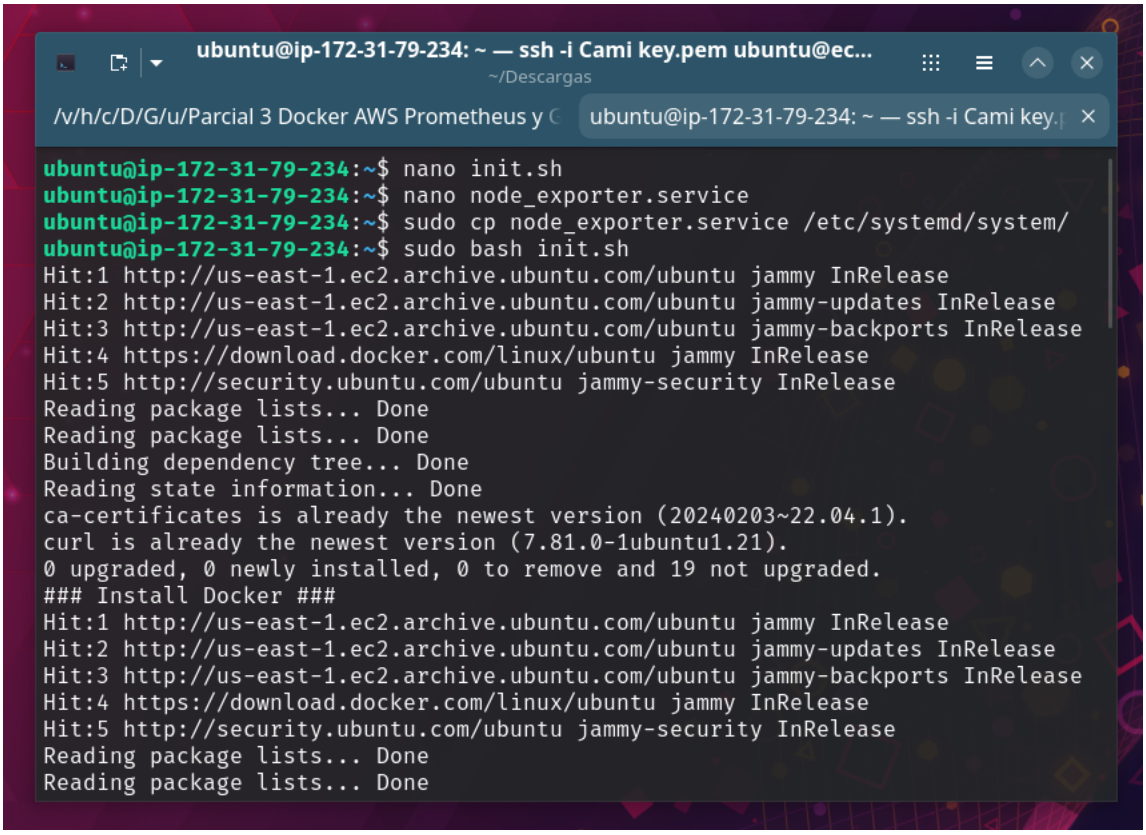




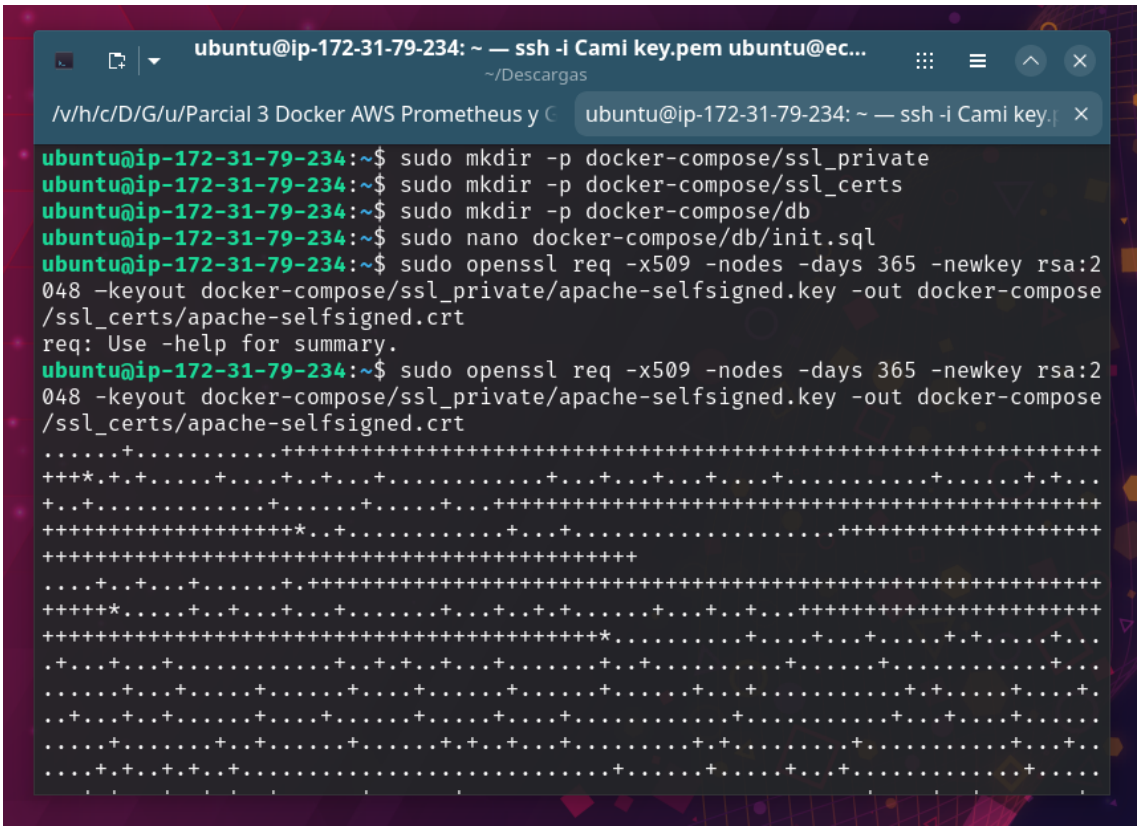

```
$ chmod 400 "Cami key.pem"
$ ssh -i "Cami key.pem" ubuntu@ec2-18-207-4-137.compute-1.amazonaws.com
```

Se ejecuta el mismo script que ScriptServidor2.sh para aprovisionar parcialmente la máquina.

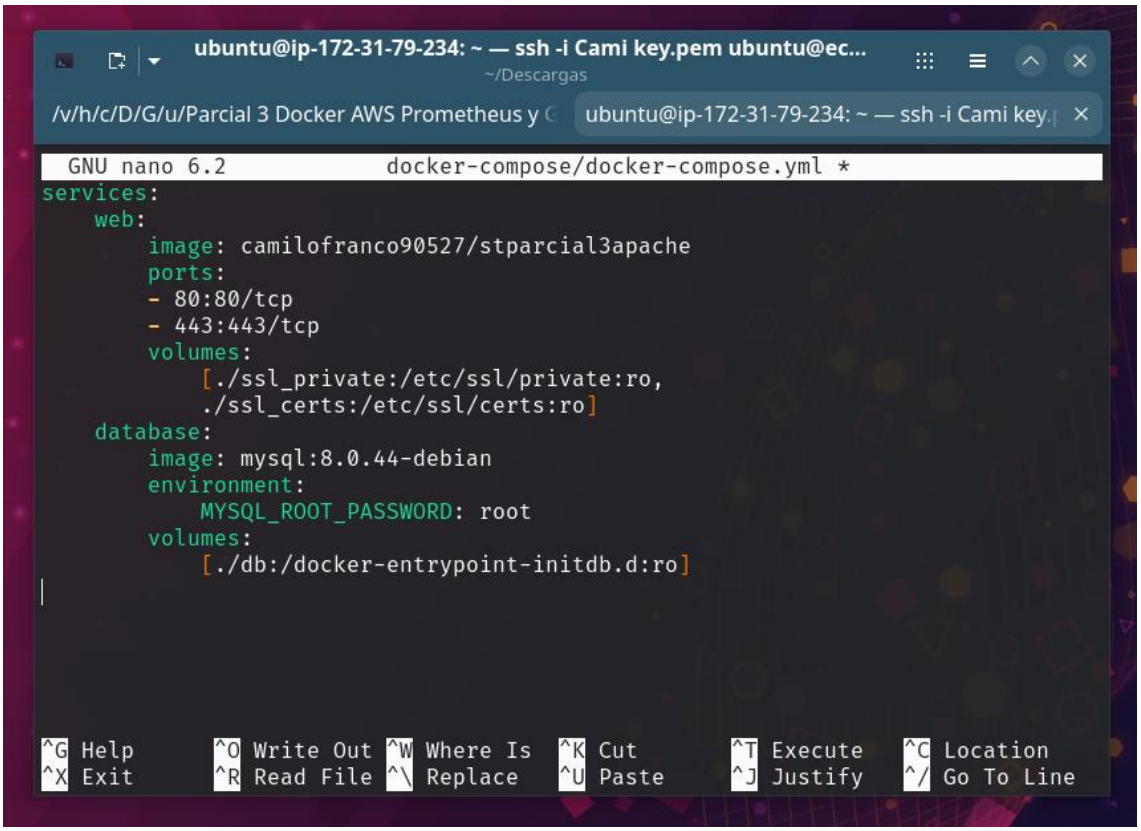


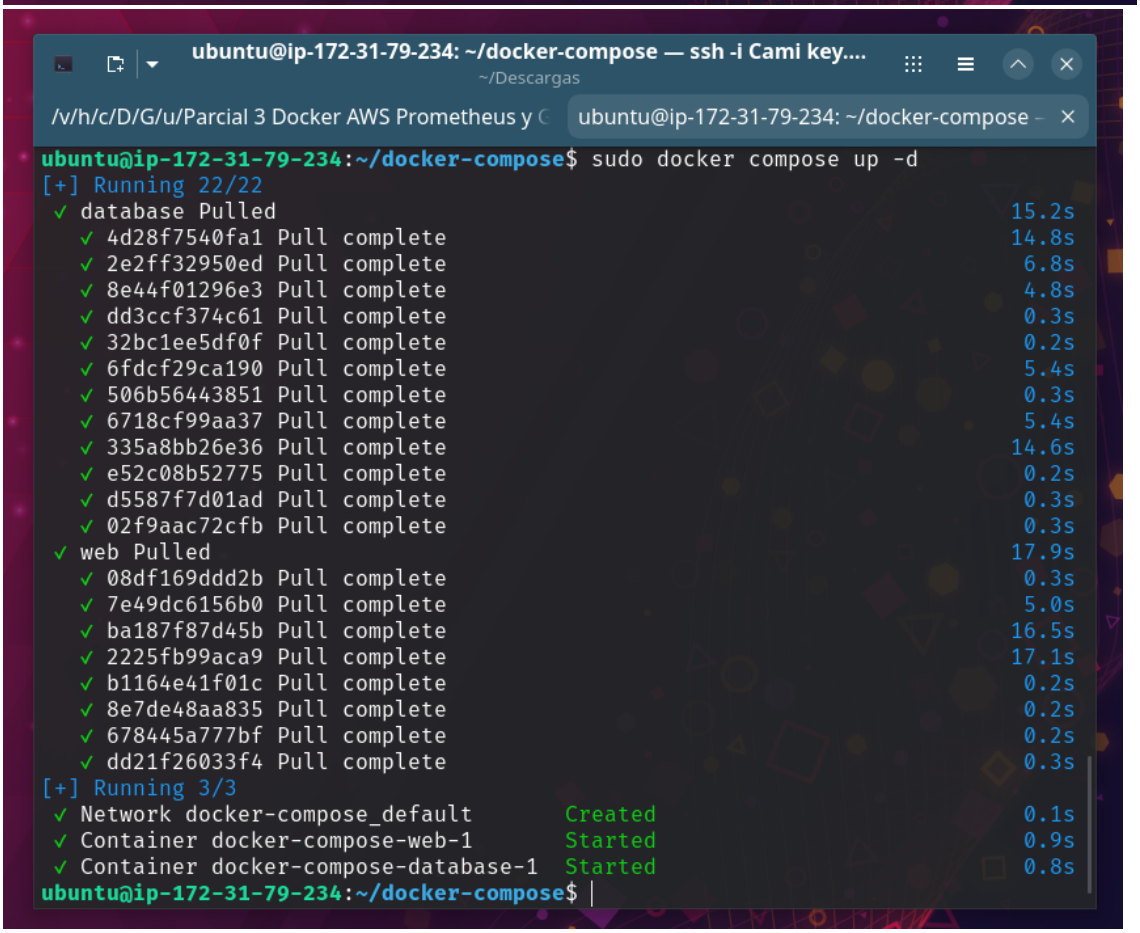


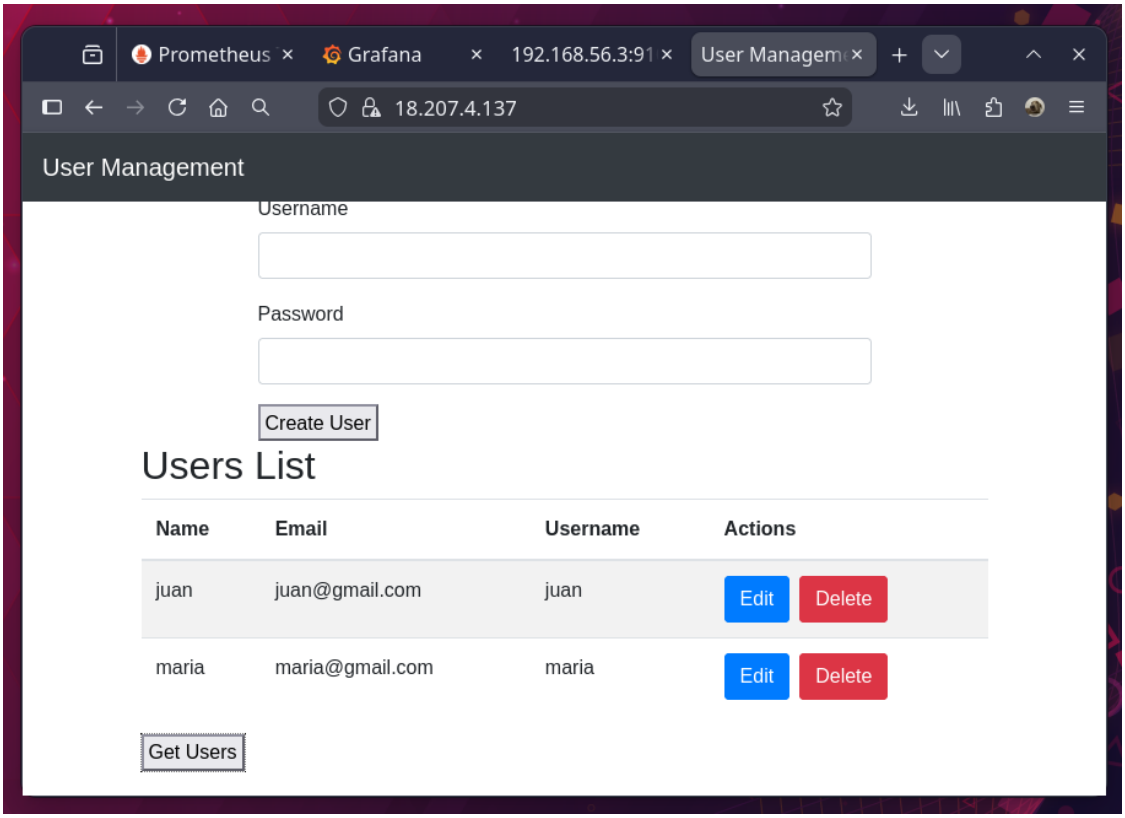

```
$ sudo cp node_exporter.service /etc/systemd/system/
$ sudo bash init.sh
```



~/docker-compose/docker-compose.yml






Prometheus
Grafana
192.168.56.3:91
User Managem
18.207.4.137
User Management
Username
Password
Create User
Users List
Name
Email
Username
Actions
juan
juan@gmail.com
juan
Edit
Delete
maria
maria@gmail.com
maria
Edit
Delete
Get Users

3. **Monitoreo con Prometheus y Node Exporter (1.5 puntos)**

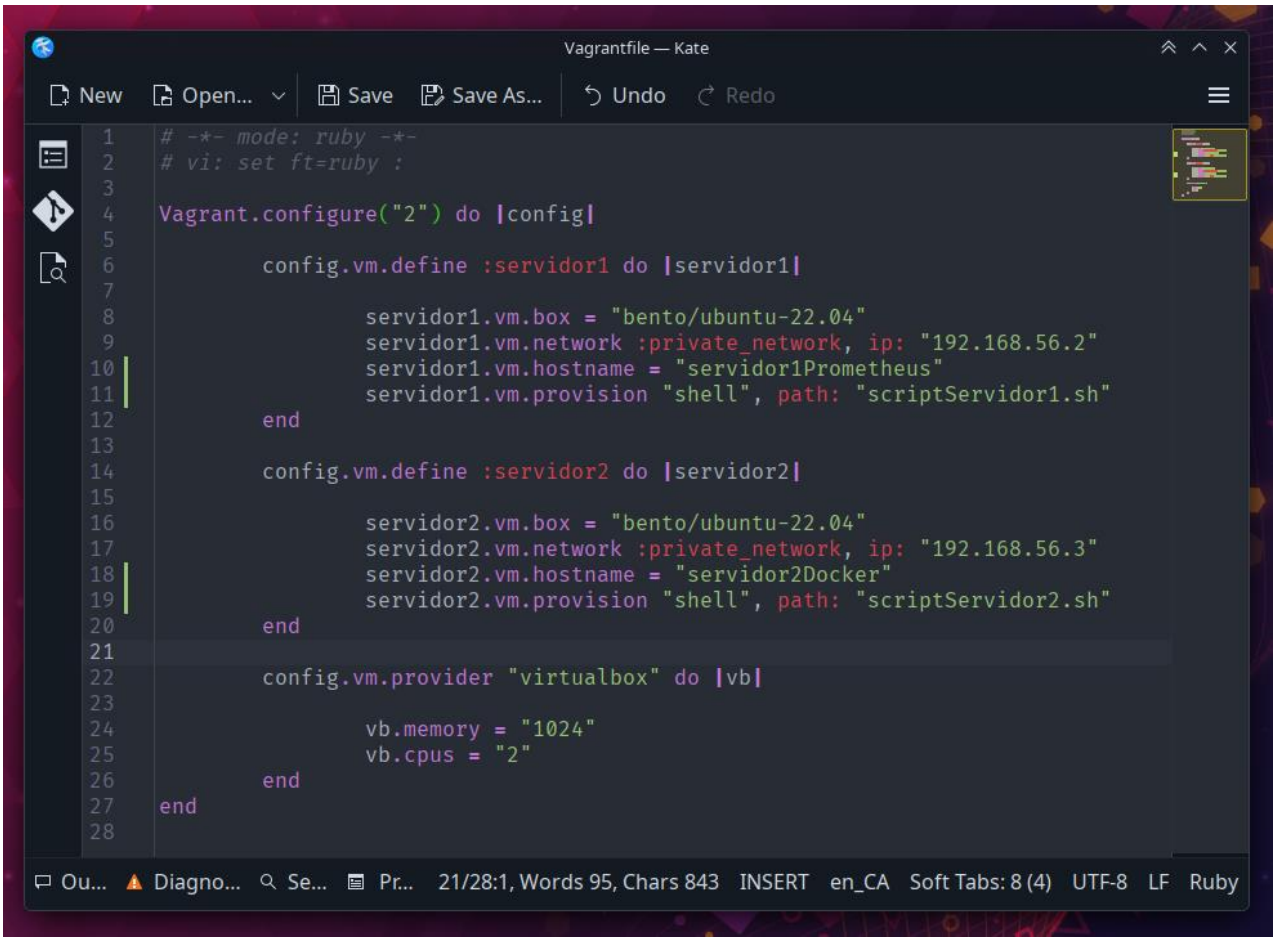
- Instale y configure Prometheus
- Configure prometheus.yml para recolectar métricas locales y de Node Exporter.
- Instale Node Exporter para recopilar métricas del sistema (CPU, memoria, disco).
- Documente tres métricas específicas y explique su utilidad en el monitoreo de un sistema Linux.
- Configure alertas básicas en Prometheus (por ejemplo, CPU > 80%).

Recursos sugeridos (Sin garantía ni soporte)

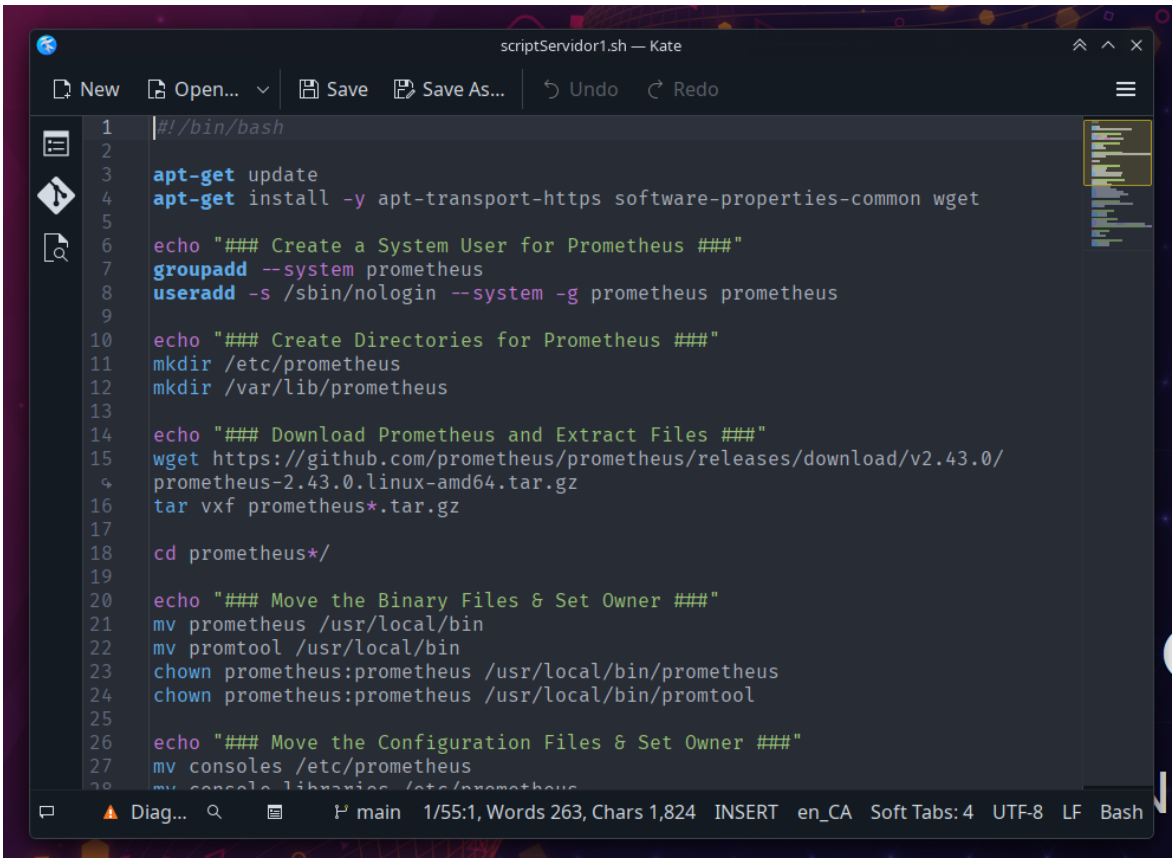
Guía para instalar Prometheus en Ubuntu

Guía para instalar y ejecutar Node Exporter

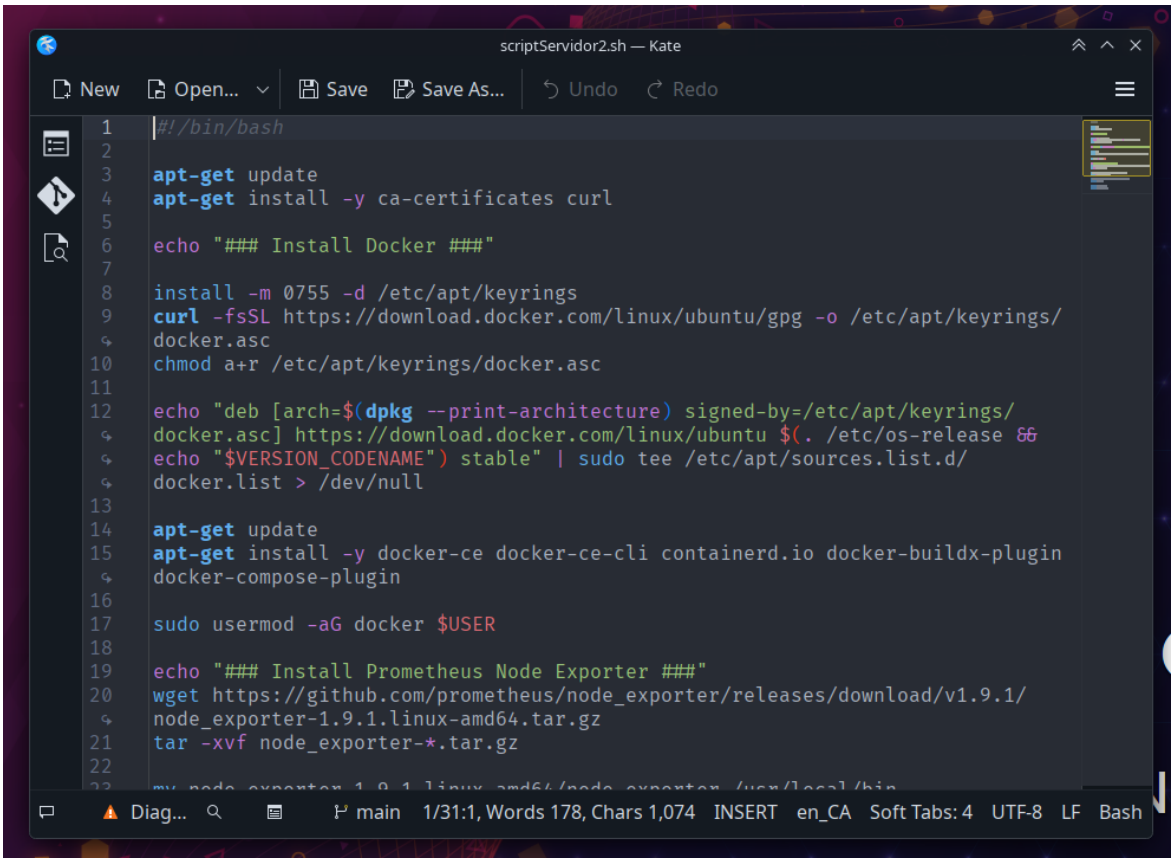
`Vagrantfile`



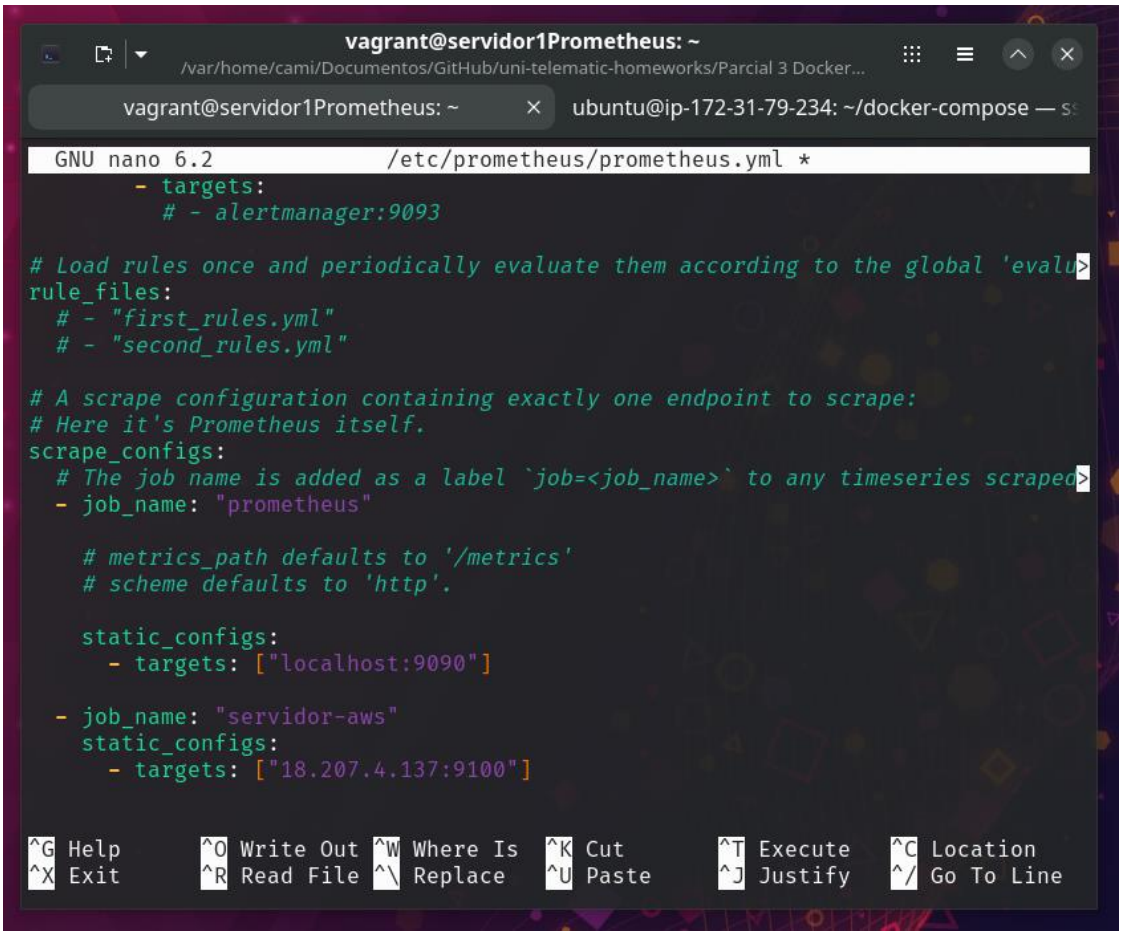
ScriptServidor1.sh



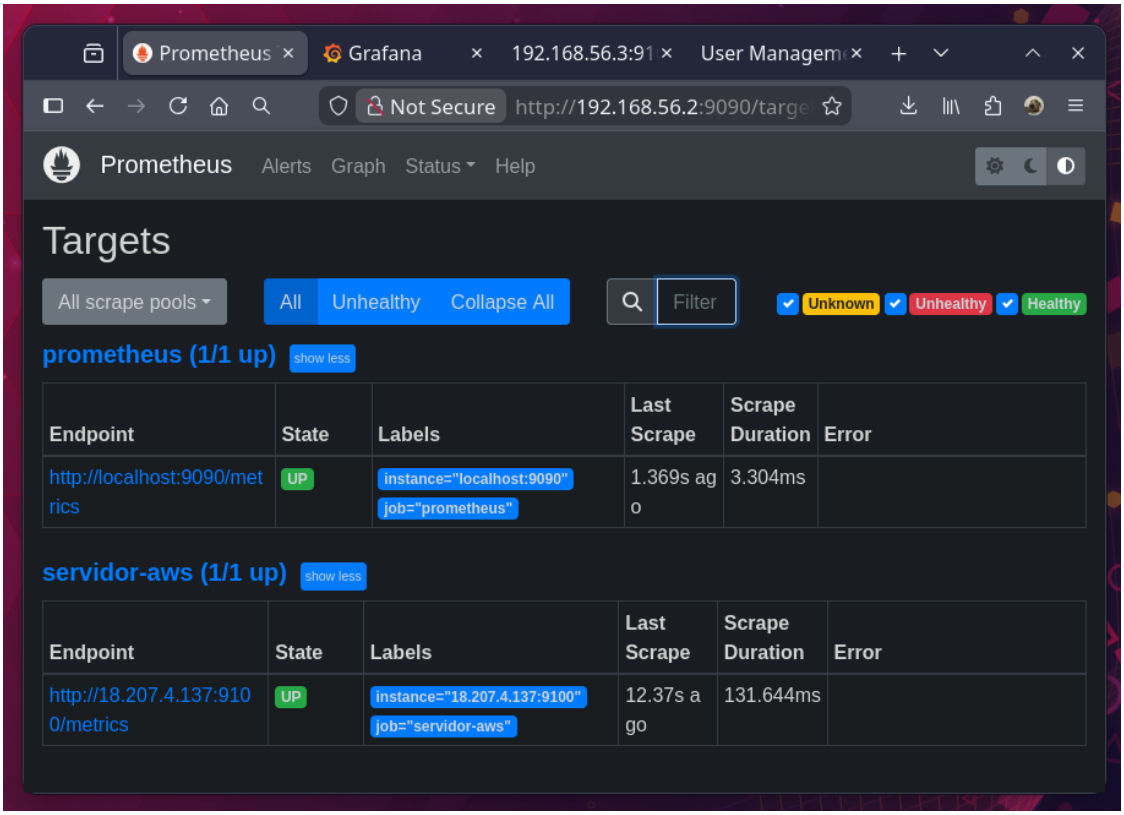
ScriptServidor2.sh



/etc/prometheus/prometheus.yml



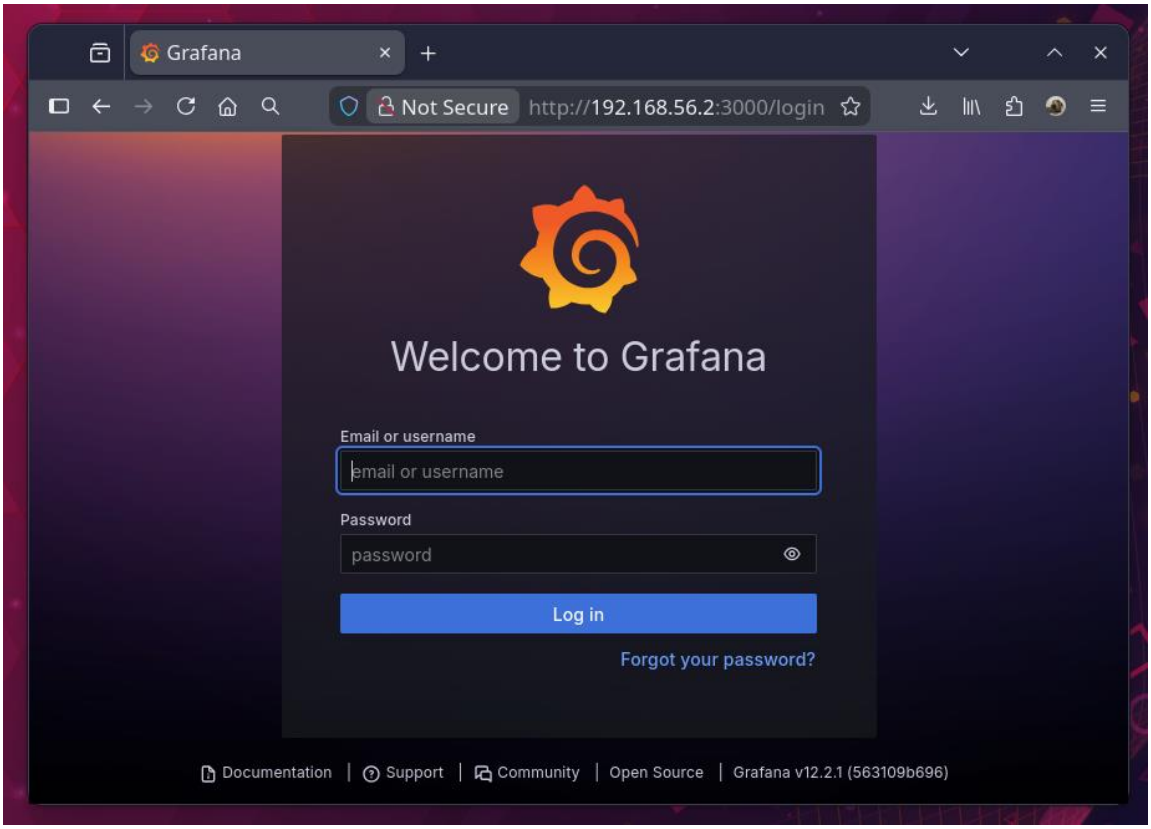

```
$ sudo systemctl restart prometheus
```

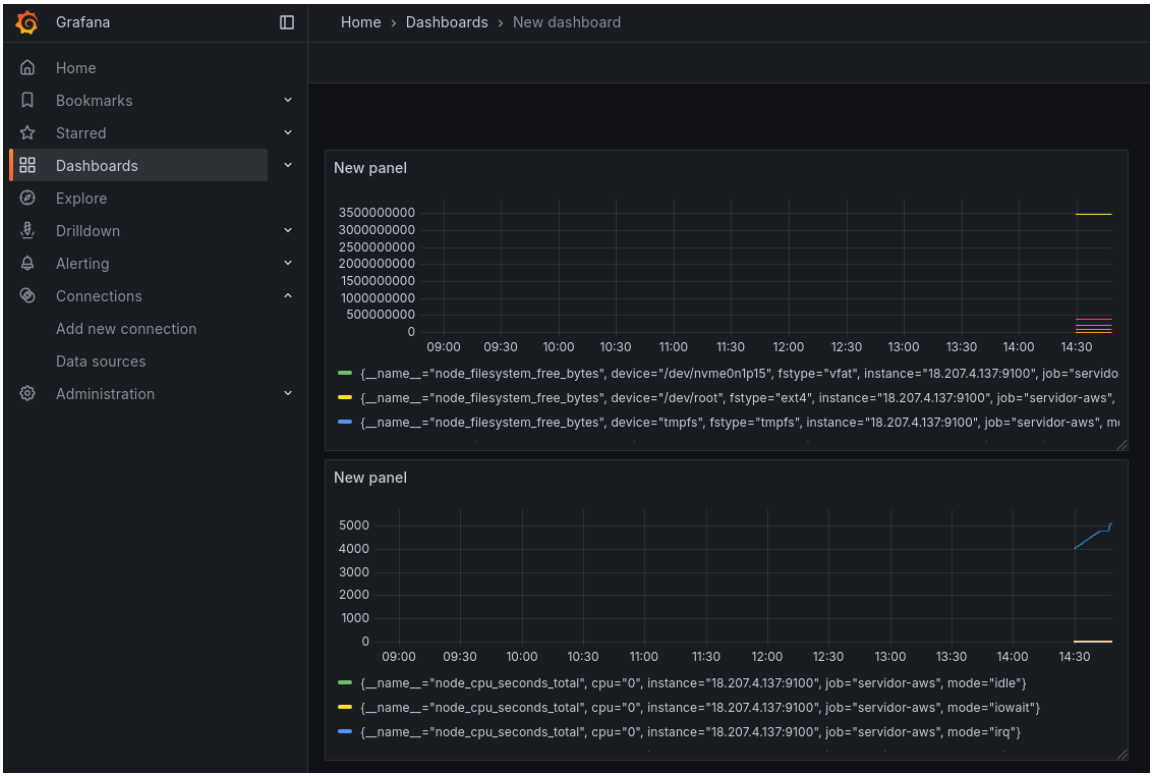

Prometheus
Grafana
192.168.56.3:91
User Managem
Not Secure
http://192.168.56.2:9090/targe
Prometheus
Alerts
Graph
Status
Help
Targets
All scrape pools
All
Unhealthy
Collapse All
Filter
Unknown
Unhealthy
Healthy
prometheus (1/1 up)
show less
Endpoint
State
Labels
Last Scrape
Scrape Duration
Error
http://localhost:9090/metrics
UP
instance="localhost:9090"
job="prometheus"
1.369s ago
3.304ms
servidor-aws (1/1 up)
show less
Endpoint
State
Labels
Last Scrape
Scrape Duration
Error
http://18.207.4.137:9100/metrics
UP
instance="18.207.4.137:9100"
job="servidor-aws"
12.37s ago
131.644ms

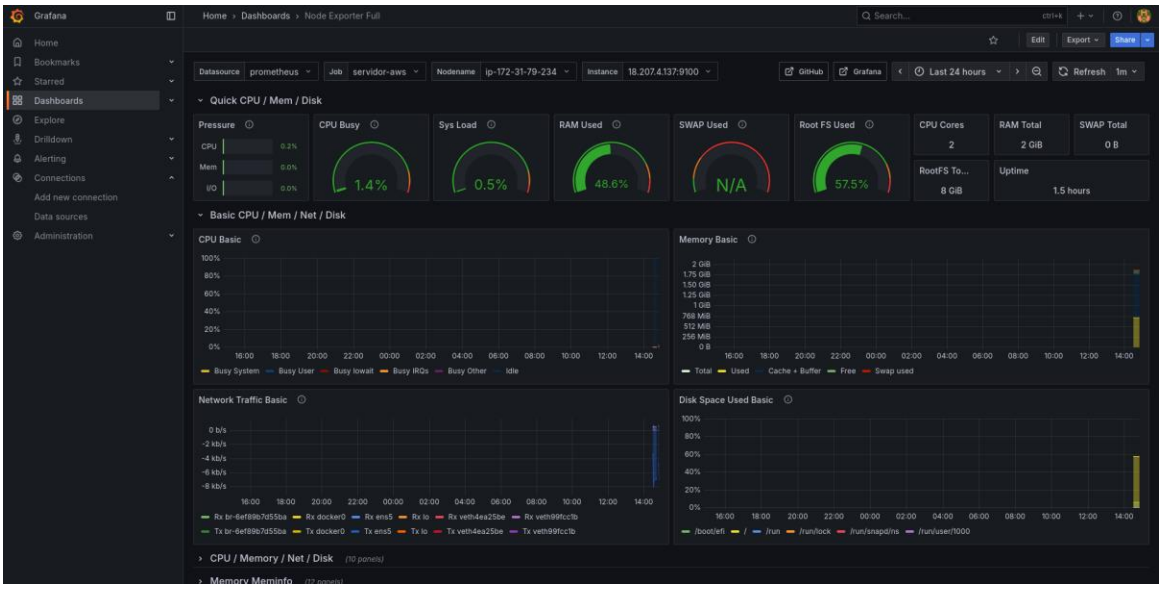
4. **Visualización con Grafana (1.0 puntos)**

- Instale Grafana y conéctelo a Prometheus como fuente de datos.
- Cree un dashboard con al menos dos paneles: uno de uso de CPU/memoria y un gauge de espacio en disco.
- Importe un panel preconfigurado desde la biblioteca oficial de Grafana y verifique su funcionamiento.

Recurso sugerido ***Guía para integrar Grafana con Prometheus***




Grafana
Home › Dashboards › New dashboard
Home
Bookmarks
Starred
Dashboards
Explore
Drilldown
Alerting
Connections
Add new connection
Data sources
Administration
New panel
3500000000
3000000000
2500000000
2000000000
1500000000
1000000000
500000000
0
09:00 09:30 10:00 10:30 11:00 11:30 12:00 12:30 13:00 13:30 14:00 14:30
{__name__="node_filesystem_free_bytes", device="/dev/nvme0n1p15", fstype="vfat", instance="18.207.4.137:9100", job="servido
{__name__="node_filesystem_free_bytes", device="/dev/root", fstype="ext4", instance="18.207.4.137:9100", job="servidor-aws",
{__name__="node_filesystem_free_bytes", device="tmpfs", fstype="tmpfs", instance="18.207.4.137:9100", job="servidor-aws", m
New panel
5000
4000
3000
2000
1000
0
09:00 09:30 10:00 10:30 11:00 11:30 12:00 12:30 13:00 13:30 14:00 14:30
{__name__="node_cpu_seconds_total", cpu="0", instance="18.207.4.137:9100", job="servidor-aws", mode="idle"}
{__name__="node_cpu_seconds_total", cpu="0", instance="18.207.4.137:9100", job="servidor-aws", mode="iowait"}
{__name__="node_cpu_seconds_total", cpu="0", instance="18.207.4.137:9100", job="servidor-aws", mode="irq"}


Grafana
Home › Dashboards › Node Exporter Full
Edit
Export
Share
Datasource prometheus
Job servidor-aws
Nodename ip-172-31-79-234
Instance 18.207.4.137:9100
GitHub
Grafana
Last 24 hours
Refresh 1m
Quick CPU / Mem / Disk
Pressure
CPU Busy
1.4%
Sys Load
0.5%
RAM Used
48.6%
SWAP Used
N/A
Root FS Used
57.5%
CPU Cores
2
RAM Total
2 GiB
SWAP Total
0 B
RootFS To...
8 GiB
Uptime
1.5 hours
Basic CPU / Mem / Net / Disk
CPU Basic
Memory Basic
Network Traffic Basic
Disk Space Used Basic
CPU / Memory / Net / Disk
Memory Meminfo

Entrega de Resultados

Cree un repositorio público en GitHub con todos los archivos de configuración, scripts, dashboards y un README.md explicativo. Incluya evidencias del despliegue (capturas o video corto). Comparta el enlace del repositorio en el sitio del curso para su evaluación.

Link del repositorio de Github

Conclusión Técnica

Responda brevemente en su README.md:

• ¿Qué aprendió al integrar Docker, AWS y Prometheus?

Aprendí a orquestar contenedores de forma correcta, y reforcé mis conocimientos de cursos anteriores sobre el despliegue de servidores de producción y el uso de herramientas de monitoreo.

• ¿Qué fue lo más desafiante y cómo lo resolvería en un entorno real?

Lo más desafiante fue hacer que los contenedores se integraran de forma correcta, ya que surgían problemas que no se podían entender a simple vista. La forma de resolverlo sería practicar más con ese tipo de despliegues de múltiples contenedores para aprender más sobre ello.

• ¿Qué beneficio aporta la observabilidad en el ciclo DevOps?

La observabilidad permite detectar y resolver problemas de manera proactiva, mejorando la confiabilidad y el rendimiento de las aplicaciones. Facilita la identificación de cuellos de botella, optimiza recursos y mejora la experiencia del usuario final.