

uaO	UNIVERSIDAD AUTÓNOMA DE OCCIDENTE				Valoración
	FACULTAD DE INGENIERÍA		NOMBRE DE LA ASIGNATURA	Servicios Telemáticos	
	CÓDIGO:	2195889	NOMBRE:	Isabella Castañeda & Camilo Franco Moya	
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](#)

```

Vagrantfile - MiniWebApp — Kate
New Open... Save Save As... Undo Redo
1 # -*- mode: ruby -*-
2 # vi: set ft=ruby :
3
4 Vagrant.configure("2") do |config|
5
6   config.vm.define :servidorWeb do |servidorWeb|
7
8     servidorWeb.vm.box = "bento/ubuntu-22.04"
9     servidorWeb.vm.network :private_network, ip: "192.168.56.3"
10    servidorWeb.vm.provision "file", source: "webapp", destination: "/home/vagrant/webapp"
11    servidorWeb.vm.provision "file", source: "init.sql", destination: "/home/vagrant/init.sql"
12    servidorWeb.vm.provision "shell", path: "script.sh"
13    servidorWeb.vm.hostname = "servidorWeb"
14  end
15  Oscar Mondragon Martinez: 4/12/24: first commit users API Rest
16
17  config.vm.provider "virtualbox" do |vb|
18
19    vb.memory = "1024"
20    vb.cpus = "2"
21  end
22
23 end

```

Output Diagnostics Search Project main 14/23:6, Words 82, Chars 604 INSERT en_CA Soft Tabs: 2 (4) UTF-8 LF Ruby

```

$ 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
```

```
vagrant@servidorWeb:~$ sudo a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create self-signed
certificates.
To activate the new configuration, you need to run:
    systemctl restart apache2
```

```
$ 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
```

vagrant@servidorWeb: ~
/var/home/cami/Descargas/Servicios telemáticos...

```
GNU nano 6.2 /var/www/webapp/application.wsgi *
#!/usr/bin/python
import sys
sys.path.insert(0,"/var/www/webapp/")
from web.views import app as application
|
```

GNU nano 6.2 Help Commands:
^G Help ^O Write Out ^W Where Is ^K Cut
^X Exit ^R Read File ^\ Replace ^U Paste

```
/var/www/webapp/.htaccess
```

vagrant@servidorWeb: ~
/var/home/cami/Descargas/Servicios telemáticos...

```
GNU nano 6.2 /var/www/webapp/.htaccess
RewriteEngine On
RewriteCond %{HTTPS} !=on
RewriteRule ^/?(.*) https:// %{SERVER_NAME}/ [R,L]
```

GNU nano 6.2 Help Commands:
^G Help ^O Write Out ^W Where Is ^K Cut
^X Exit ^R Read File ^\ Replace ^U Paste

```
/var/www/webapp/config.py
```

vagrant@servidorWeb: ~
/var/home/cami/Descargas/Servicios telemáticos/Vagrant/MiniWebApp

```
GNU nano 6.2 docker_apache/webapp/config.py
class Config:
    MYSQL_HOST = 'database'
    MYSQL_USER = 'root'
    MYSQL_PASSWORD = 'root'
    MYSQL_DB = 'myflaskapp'
    SQLALCHEMY_DATABASE_URI = f'mysql:// {MYSQL_USER}:{MYSQL_PASSWORD}@{MYSQL_HOST}/{MYSQL_DB}'
```

GNU nano 6.2 Help Commands:
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line

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

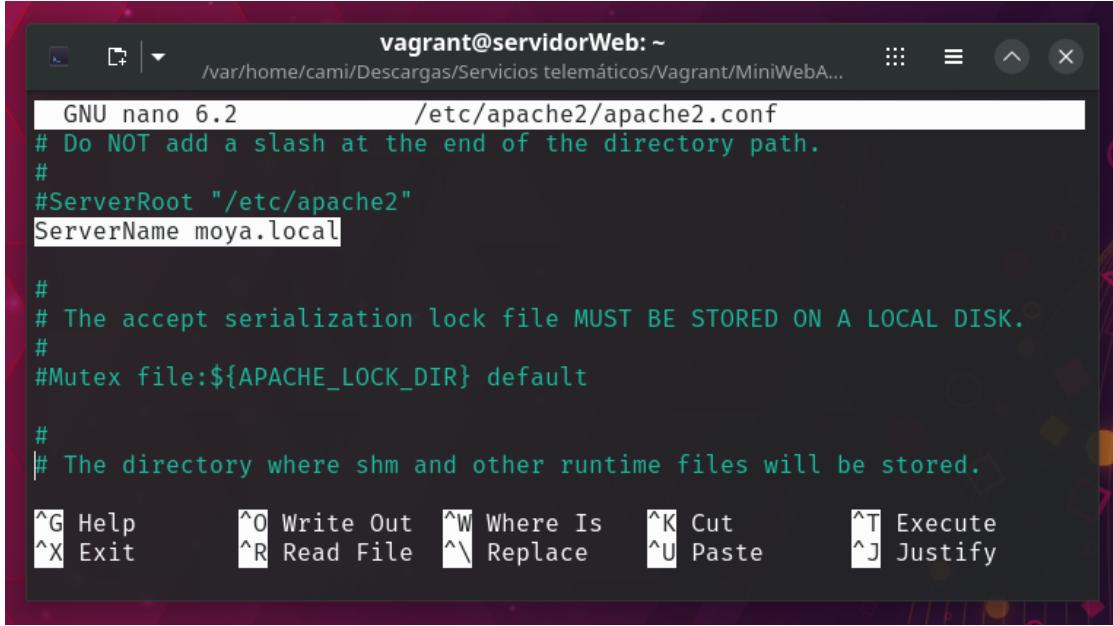
```
vagrant@servidorWeb: ~
/var/home/cami/Descargas/Servicios telemáticos/Vagrant/MiniWebA...
GNU nano 6.2 /etc/apache2/sites-available/000-default.conf
WSGIScriptAlias / /var/www/webapp/application.wsgi
DocumentRoot /var/www/webapp

<VirtualHost *:80>
    <Directory /var/www/webapp/>
        Order deny,allow
        Allow from all
    </Directory>
</VirtualHost>

<VirtualHost *:443>
    <Directory /var/www/webapp/>
        Order deny,allow
        Allow from all
    </Directory>
    SSLEngine on
    SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt
    SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key
</VirtualHost>

^G Help      ^O Write Out   ^W Where Is   ^K Cut       ^T Execute
^X Exit      ^R Read File   ^\ Replace    ^U Paste     ^J Justify
```

```
/etc/apache2/apache2.conf
```

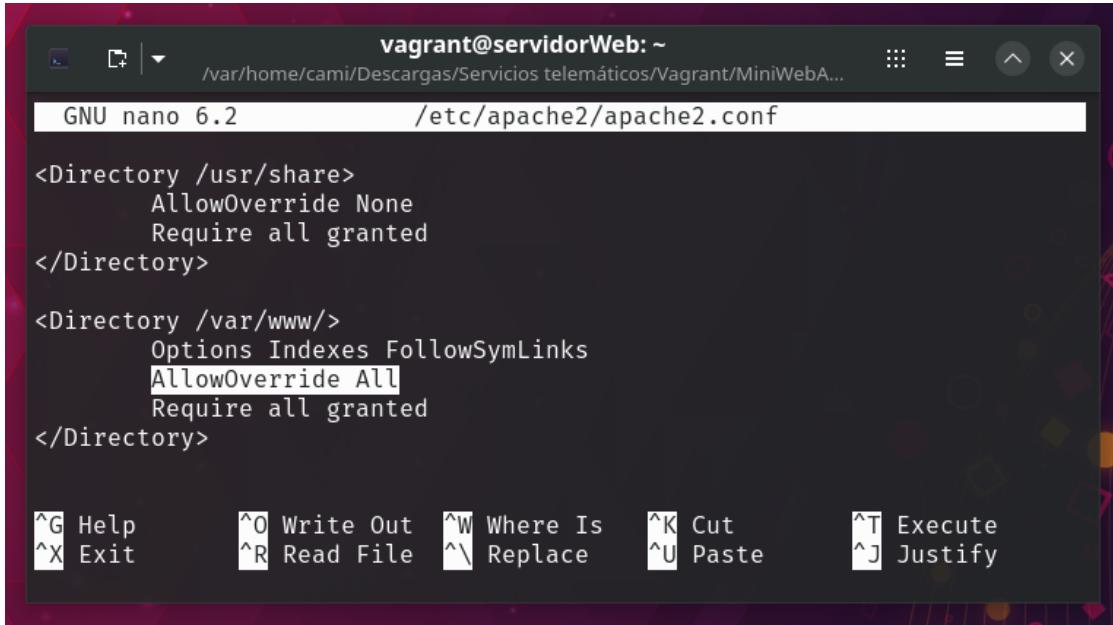


```
vagrant@servidorWeb: ~
/var/home/cami/Descargas/Servicios telemáticos/Vagrant/MiniWebA...
GNU nano 6.2          /etc/apache2/apache2.conf
# Do NOT add a slash at the end of the directory path.
#
#ServerRoot "/etc/apache2"
ServerName moya.local

#
# The accept serialization lock file MUST BE STORED ON A LOCAL DISK.
#
#Mutex file:${APACHE_LOCK_DIR} default

#
# The directory where shm and other runtime files will be stored.

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute
^X Exit      ^R Read File   ^\ Replace    ^U Paste      ^J Justify
```



```
vagrant@servidorWeb: ~
/var/home/cami/Descargas/Servicios telemáticos/Vagrant/MiniWebA...
GNU nano 6.2          /etc/apache2/apache2.conf

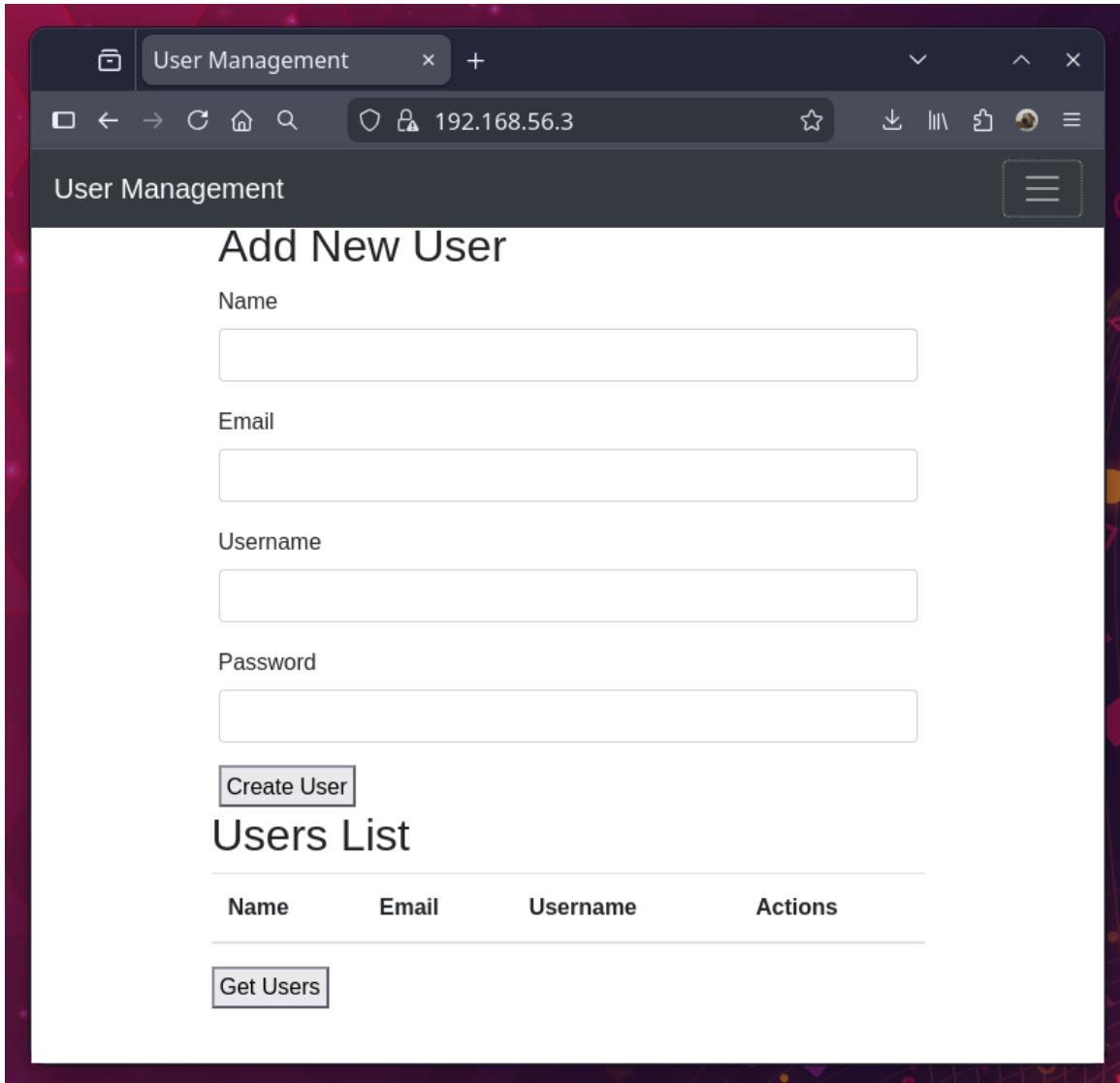
<Directory /usr/share>
    AllowOverride None
    Require all granted
</Directory>

<Directory /var/www/>
    Options Indexes FollowSymLinks
    AllowOverride All
    Require all granted
</Directory>

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute
^X Exit      ^R Read File   ^\ Replace    ^U Paste      ^J Justify
```

```
$ sudo apache2ctl configtest
```

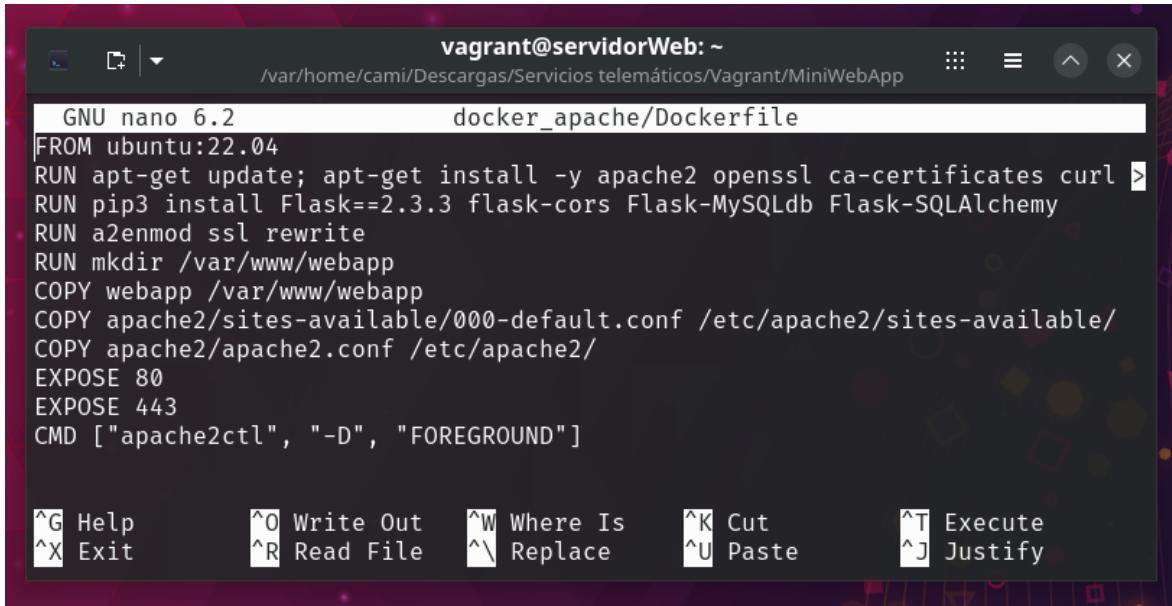
```
$ sudo systemctl restart apache2
```



```
$ 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/
```

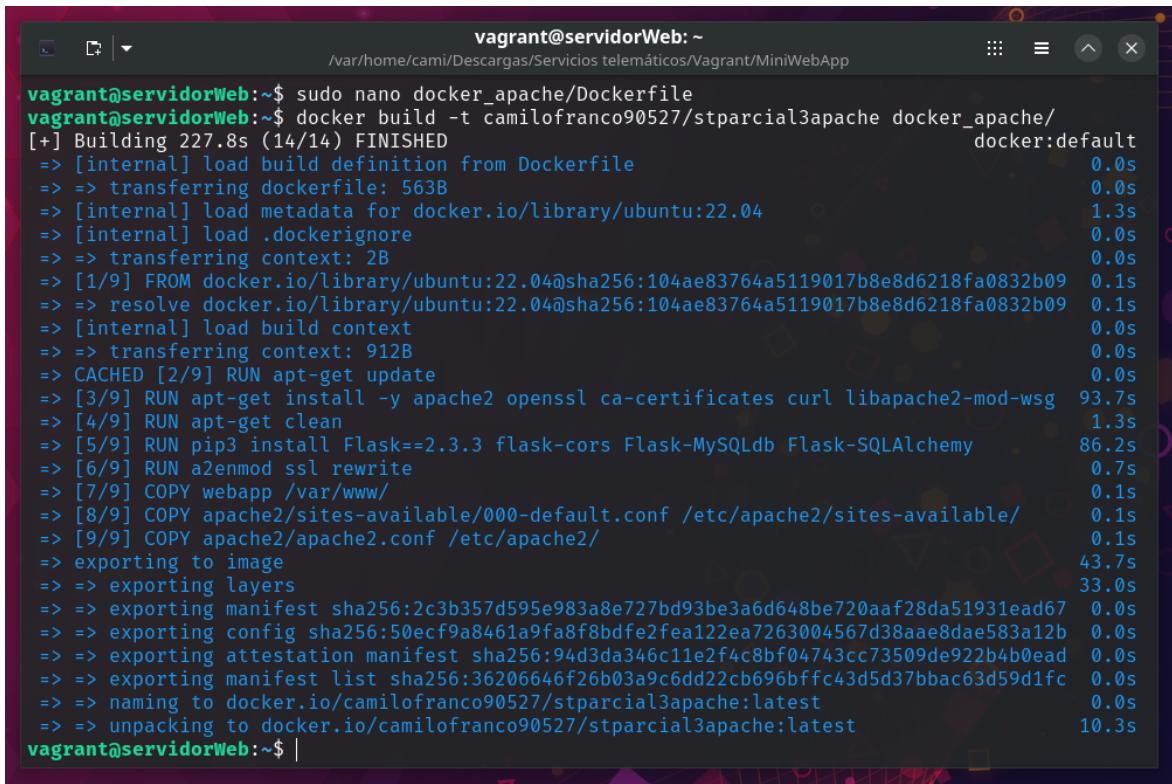
```
$ sudo cp init.sql docker_compose/db/
$ sudo cp /etc/ssl/private/apache-selfsigned.key docker_compose/ssl_private/
$ sudo cp /etc/ssl/certs/apache-selfsigned.crt docker_compose/ssl_certs/
```

~/docker_apache/Dockerfile



```
GNU nano 6.2          docker_apache/Dockerfile
FROM ubuntu:22.04
RUN apt-get update; apt-get install -y apache2 openssl ca-certificates curl >
RUN pip3 install Flask==2.3.3 flask-cors Flask-MySQLdb Flask-SQLAlchemy
RUN a2enmod ssl rewrite
RUN mkdir /var/www/webapp
COPY webapp /var/www/webapp
COPY apache2/sites-available/000-default.conf /etc/apache2/sites-available/
COPY apache2/apache2.conf /etc/apache2/
EXPOSE 80
EXPOSE 443
CMD ["apache2ctl", "-D", "FOREGROUND"]
```

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute
 ^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify



```
vagrant@servidorWeb:~$ sudo nano docker_apache/Dockerfile
vagrant@servidorWeb:~$ docker build -t camilofranco90527/stparcial3apache docker_apache/
[+] Building 227.8s (14/14) FINISHED
   => [internal] load build definition from Dockerfile          0.0s
   => => transferring dockerfile: 563B                         0.0s
   => [internal] load metadata for docker.io/library/ubuntu:22.04 1.3s
   => [internal] load .dockerignore                            0.0s
   => => transferring context: 2B                           0.0s
   => [1/9] FROM docker.io/library/ubuntu:22.04@sha256:104ae83764a5119017b8e8d6218fa0832b09 0.1s
   => => resolve docker.io/library/ubuntu:22.04@sha256:104ae83764a5119017b8e8d6218fa0832b09 0.1s
   => [internal] load build context                          0.0s
   => => transferring context: 912B                         0.0s
   => CACHED [2/9] RUN apt-get update                        0.0s
   => [3/9] RUN apt-get install -y apache2 openssl ca-certificates curl libapache2-mod-wsg 93.7s
   => [4/9] RUN apt-get clean                                1.3s
   => [5/9] RUN pip3 install Flask==2.3.3 flask-cors Flask-MySQLdb Flask-SQLAlchemy      86.2s
   => [6/9] RUN a2enmod ssl rewrite                         0.7s
   => [7/9] COPY webapp /var/www/                           0.1s
   => [8/9] COPY apache2/sites-available/000-default.conf /etc/apache2/sites-available/ 0.1s
   => [9/9] COPY apache2/apache2.conf /etc/apache2/        0.1s
   => exporting to image                                     43.7s
   => => exporting layers                                    33.0s
   => => exporting manifest sha256:2c3b357d595e983a8e727bd93be3a6d648be720aaf28da51931ead67 0.0s
   => => exporting config sha256:50ecf9a8461a9fa8f8bdfe2fea122ea7263004567d38aae8dae583a12b 0.0s
   => => exporting attestation manifest sha256:94d3da346c11e2f4c8bf04743cc73509de922b4b0ead 0.0s
   => => exporting manifest list sha256:36206646f26b03a9c6dd22cb696bf43d5d37bbac63d59d1fc 0.0s
   => => naming to docker.io/camilofranco90527/stparcial3apache:latest                  0.0s
   => => unpacking to docker.io/camilofranco90527/stparcial3apache:latest                10.3s
vagrant@servidorWeb:~$ |
```

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

~/docker_compose/docker-compose.yml

The screenshot shows a terminal window titled "vagrant@servidorWeb:~" with the path "/var/home/cami/Descargas/Servicios telemáticos/Vagr...". The file being edited is "GNU nano 6.2 docker_compose/docker-compose.yml". The content of the file is as follows:

```
services:
  web:
    image: camilofranco90527/stparcial3apache
    ports:
      - 80:80/tcp
      - 443:443/tcp
    volumes:
      - ./ssl_private:/etc/ssl/private:ro,
      - ./ssl_certs:/etc/ssl/certs:ro
  database:
    image: mysql:8.0.44-debian
    environment:
      MYSQL_ROOT_PASSWORD: root
    volumes:
      - ./db:/docker-entrypoint-initdb.d:ro
```

At the bottom of the terminal window, there is a menu bar with various keyboard shortcuts for navigating and editing the file.

```
vagrant@servidorWeb:~$ cd docker_compose/
vagrant@servidorWeb:~/docker_compose$ docker compose up -d
[+] Running 3/3
  ✓ Network docker_compose_default      Create                         0.1s
  ✓ Container docker_compose-web-1      Started                        0.4s
  ✓ Container docker_compose-database-1 Started                        0.4s
vagrant@servidorWeb:~/docker_compose$
```

```
$ sudo systemctl stop apache2
```

```
$ sudo systemctl stop mysql
```

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

User Management

Email

Username

Password

Create User

Users List

Name	Email	Username	Actions
juan	juan@gmail.com	juan	<button>Edit</button> <button>Delete</button>
maria	maria@gmail.com	maria	<button>Edit</button> <button>Delete</button>

Get Users

```
Login Succeeded
vagrant@servidorWeb:~/docker_compose$ docker tag camilofranco90527/stparcial3apache camilofranco90527/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
size: 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.

Instance summary for i-0edc63bb6e4d00750 (server ST parcial 3) [Info](#)

Updated less than a minute ago

Instance ID i-0edc63bb6e4d00750

IPv6 address -

Hostname type IP name: ip-172-31-79-234.ec2.internal

Answer private resource DNS name (IPv4 [A])

Auto-assigned IP address 18.207.4.137 [Public IP]

IAM Role -

IMDSv2 Required

Operator -

Public IPv4 address 18.207.4.137 | [open address](#)

Private IP4 address 172.31.79.234

Instance state Running

Private IP DNS name (IPv4 only) ip-172-31-79-234.ec2.internal

Instance type t3.small

VPC ID vpc-095c8dec48cff339d

Subnet ID subnet-0cf69f099cc8f9a9e

Instance ARN arn:aws:ec2:us-east-1:120630068548:instance/i-0edc63bb6e4d00750

Elastic IP addresses -

AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. | Learn more

Auto Scaling Group name -

Managed false

Details [Status and alarms](#) [Monitoring](#) [Security](#) [Networking](#) [Storage](#) [Tags](#)

Instance details [Info](#)

AMI ID ami-0c598cb65a950472

AMI name Ubuntu/images/hvm-ssd/ubuntu-jammy-22.04-amd64-server-20251015

Stop protection Disabled

Monitoring disabled

Allowed image -

Platform details Linux/UNIX

Termination protection Disabled

AMI location amazon/ubuntu/images/hvm-ssd/ubuntu-jammy-22.04-amd64-server-20251015

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

Edit inbound rules [Info](#)

Inbound rules control the incoming traffic that's allowed to reach the instance.

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-0a6f05d66cd4ce3cf	Custom TCP	TCP	9100	Custom	0.0.0.0/0
sgr-021003e6e579c5c9f	SSH	TCP	22	Custom	0.0.0.0/0
sgr-0c953a956f0eac51	HTTP	TCP	80	Custom	0.0.0.0/0
sgr-092c8fc7cb58bd9a6	HTTPS	TCP	443	Custom	0.0.0.0/0

Add rule

⚠ Rules with source of 0.0.0.0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

[Cancel](#) [Preview changes](#) [Save rules](#)

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

```
ubuntu@ip-172-31-79-234: ~ — ssh -i Cami key.pem ubuntu@ec...      ~/Descargas
/v/h/c/D/G/u/Parcial 3 Docker AWS Prometheus y C  ubuntu@ip-172-31-79-234: ~ — ssh -i Cami key... X

cami at laptop-fedora-cami in ~/Descargas
↳ chmod 400 "Cami key.pem"
cami at laptop-fedora-cami in ~/Descargas
↳ ssh -i "Cami key.pem" ubuntu@ec2-18-207-4-137.compute-1.amazonaws.com
The authenticity of host 'ec2-18-207-4-137.compute-1.amazonaws.com (18.207.4.137)' can't be established.
ED25519 key fingerprint is SHA256:51RZQa4JjfRguv4acel0mqJRJfK9MUUCYI9N9c0yfHI.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-18-207-4-137.compute-1.amazonaws.com' (ED25519)
to the list of known hosts.
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.8.0-1040-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

System information as of Tue Nov 18 18:27:21 UTC 2025

System load: 0.02          Processes: 102
Usage of /: 22.7% of 7.57GB Users logged in: 0
Memory usage: 10%          IPv4 address for ens5: 172.31.79.234
Swap usage: 0%             
```

```
$ 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.
```

```
ubuntu@ip-172-31-79-234: ~ — ssh -i Cami key.pem ubuntu@ec...      ~/Descargas
/v/h/c/D/G/u/Parcial 3 Docker AWS Prometheus y C  ubuntu@ip-172-31-79-234: ~ — ssh -i Cami key... X

GNU nano 6.2          init.sh *
#!/bin/bash

apt-get update
apt-get install -y ca-certificates curl

echo "### Install Docker ###"

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

echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | tee /etc/apt/sources.list.d/docker.list >> /dev/null

apt-get update
apt-get install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

sudo usermod -aG docker $USER

echo "### Install Prometheus Node Exporter ###"
wget https://github.com/prometheus/node_exporter/releases/download/v1.9.1/node_
```

```
ubuntu@ip-172-31-79-234: ~ — ssh -i Cami key.pem ubuntu@ec... ~/Descargas
/v/h/c/D/G/u/Parcial 3 Docker AWS Prometheus y C ubuntu@ip-172-31-79-234: ~ — ssh -i Cami key.| x

ubuntu@ip-172-31-79-234:~$ nano init.sh
ubuntu@ip-172-31-79-234:~$ nano node_exporter.service
ubuntu@ip-172-31-79-234:~$ sudo cp node_exporter.service /etc/systemd/system/
ubuntu@ip-172-31-79-234:~$ sudo bash init.sh
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu jammy InRelease
Hit:5 http://security.ubuntu.com/ubuntu jammy-security InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240203~22.04.1).
curl is already the newest version (7.81.0-1ubuntu1.21).
0 upgraded, 0 newly installed, 0 to remove and 19 not upgraded.
### Install Docker ###
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu jammy InRelease
Hit:5 http://security.ubuntu.com/ubuntu jammy-security InRelease
Reading package lists... Done
Reading package lists... Done
```

```
$ sudo cp node_exporter.service /etc/systemd/system/
```

```
$ sudo bash init.sh
```

```
~/docker-compose/docker-compose.yml
```

```
ubuntu@ip-172-31-79-234: ~ — ssh -i Cami key.pem ubuntu@ec...      :~ /Descargas
/v/h/c/D/G/u/Parcial 3 Docker AWS Prometheus y C  ubuntu@ip-172-31-79-234: ~ — ssh -i Cami key... X

GNU nano 6.2          docker-compose/docker-compose.yml *
services:
  web:
    image: camilofranco90527/stparcial3apache
    ports:
      - 80:80/tcp
      - 443:443/tcp
    volumes:
      [./ssl_private:/etc/ssl/private:ro,
       ./ssl_certs:/etc/ssl/certs:ro]
  database:
    image: mysql:8.0.44-debian
    environment:
      MYSQL_ROOT_PASSWORD: root
    volumes:
      [./db:/docker-entrypoint-initdb.d:ro]

^G Help      ^O Write Out  ^W Where Is  ^K Cut      ^T Execute  ^C Location
^X Exit      ^R Read File  ^\ Replace   ^U Paste    ^J Justify  ^/ Go To Line
```

```
ubuntu@ip-172-31-79-234: ~/docker-compose — ssh -i Cami key.... ~ /Descargas
/v/h/c/D/G/u/Parcial 3 Docker AWS Prometheus y C  ubuntu@ip-172-31-79-234: ~/docker-compose - X

ubuntu@ip-172-31-79-234:~/docker-compose$ sudo docker compose up -d
[+] Running 22/22
  ✓ database Pulled                               15.2s
    ✓ 4d28f7540fa1 Pull complete                 14.8s
    ✓ 2e2ff32950ed Pull complete                  6.8s
    ✓ 8e44f01296e3 Pull complete                  4.8s
    ✓ dd3ccf374c61 Pull complete                  0.3s
    ✓ 32bc1ee5df0f Pull complete                  0.2s
    ✓ 6fdcf29ca190 Pull complete                  5.4s
    ✓ 506b56443851 Pull complete                  0.3s
    ✓ 6718cf99aa37 Pull complete                  5.4s
    ✓ 335a8bb26e36 Pull complete                 14.6s
    ✓ e52c08b52775 Pull complete                  0.2s
    ✓ d5587f7d01ad Pull complete                  0.3s
    ✓ 02f9aac72cfb Pull complete                  0.3s
  ✓ web Pulled                                    17.9s
    ✓ 08df169ddd2b Pull complete                 0.3s
    ✓ 7e49dc6156b0 Pull complete                 5.0s
    ✓ ba187f87d45b Pull complete                 16.5s
    ✓ 2225fb99aca9 Pull complete                 17.1s
    ✓ b1164e41f01c Pull complete                 0.2s
    ✓ 8e7de48aa835 Pull complete                 0.2s
    ✓ 678445a777bf Pull complete                 0.2s
    ✓ dd21f26033f4 Pull complete                 0.3s
[+] Running 3/3
  ✓ Network docker-compose_default             Created      0.1s
  ✓ Container docker-compose-web-1            Started     0.9s
  ✓ Container docker-compose-database-1        Started     0.8s
ubuntu@ip-172-31-79-234:~/docker-compose$ |
```

The screenshot shows a web application interface with two main sections: "User Management" and "Users List".

User Management:

- Form fields for "Username" and "Password".
- A "Create User" button.

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)

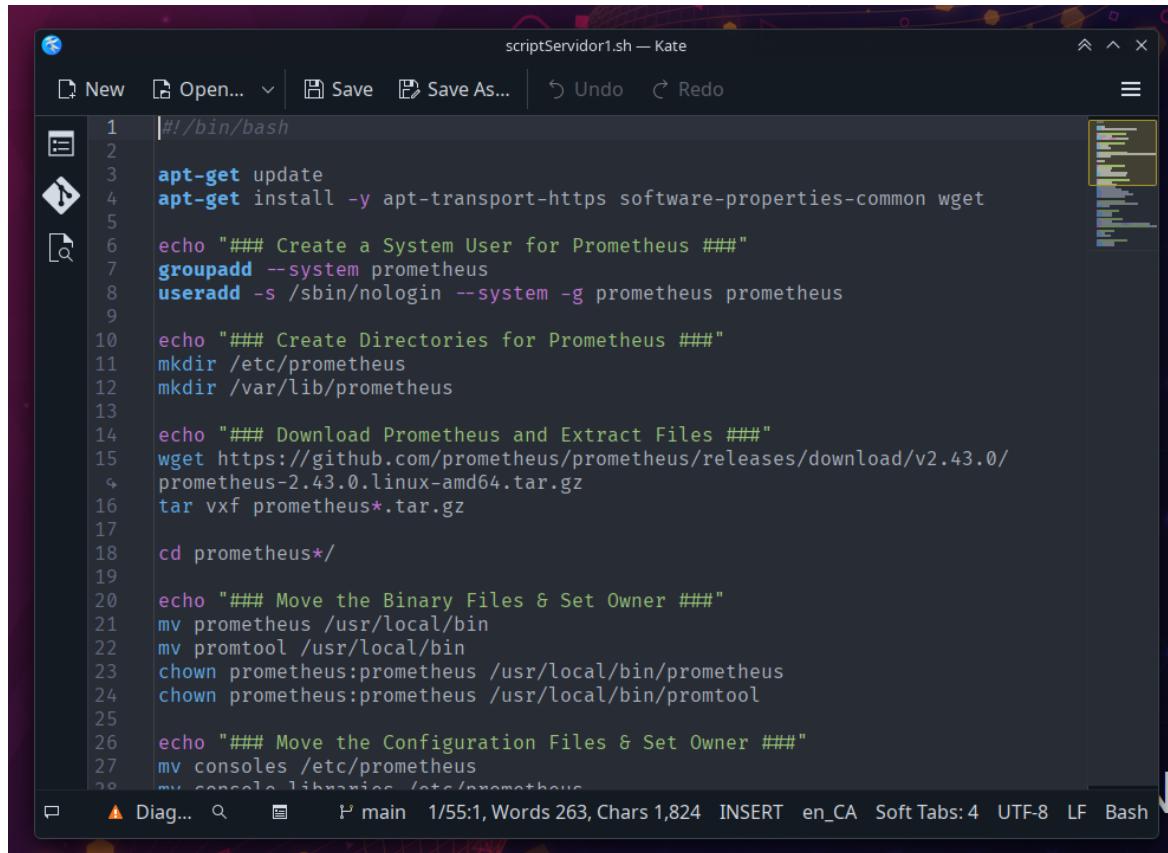
Guía para instalar Prometheus en Ubuntu

Guía para instalar y ejecutar Node Exporter

Vagrantfile

```
Vagrantfile — Kate
File New Open... Save Save As... Undo Redo
1 # -*- mode: ruby -*-
2 # vi: set ft=ruby :
3
4 Vagrant.configure("2") do |config|
5
6     config.vm.define :servidor1 do |servidor1|
7
8         servidor1.vm.box = "bento/ubuntu-22.04"
9         servidor1.vm.network :private_network, ip: "192.168.56.2"
10        servidor1.vm.hostname = "servidor1Prometheus"
11        servidor1.vm.provision "shell", path: "scriptServidor1.sh"
12    end
13
14    config.vm.define :servidor2 do |servidor2|
15
16        servidor2.vm.box = "bento/ubuntu-22.04"
17        servidor2.vm.network :private_network, ip: "192.168.56.3"
18        servidor2.vm.hostname = "servidor2Docker"
19        servidor2.vm.provision "shell", path: "scriptServidor2.sh"
20    end
21
22    config.vm.provider "virtualbox" do |vb|
23
24        vb.memory = "1024"
25        vb.cpus = "2"
26    end
27
28 end
```

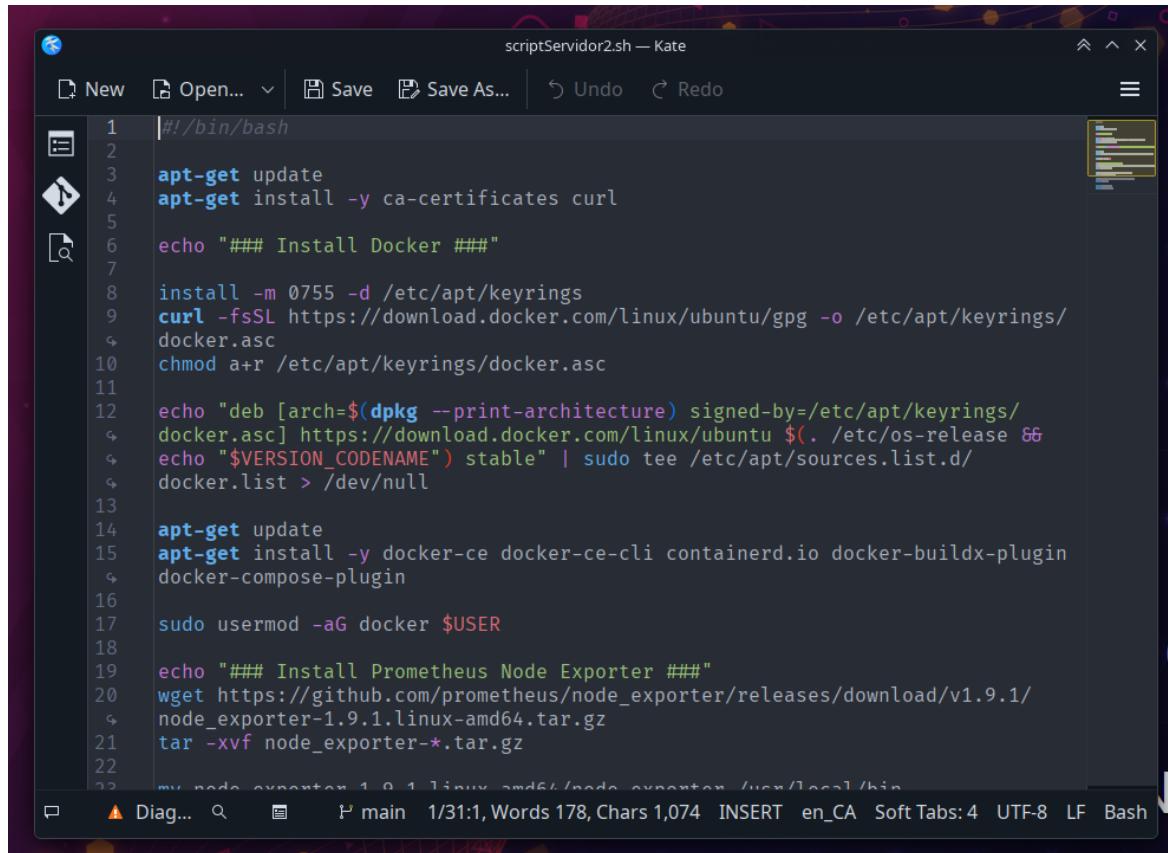
ScriptServidor1.sh



The screenshot shows the Kate code editor with the file "scriptServidor1.sh" open. The code is a Bash script for setting up Prometheus. It includes commands for updating the package list, installing curl, creating a system user, creating directories, downloading and extracting the Prometheus binary, moving files, and setting ownership. The code is color-coded for syntax.

```
#!/bin/bash
apt-get update
apt-get install -y apt-transport-https software-properties-common wget
echo "### Create a System User for Prometheus ###"
groupadd --system prometheus
useradd -s /sbin/nologin --system -g prometheus prometheus
echo "### Create Directories for Prometheus ###"
mkdir /etc/prometheus
mkdir /var/lib/prometheus
echo "### Download Prometheus and Extract Files ###"
wget https://github.com/prometheus/prometheus/releases/download/v2.43.0/
prometheus-2.43.0.linux-amd64.tar.gz
tar xvf prometheus*.tar.gz
cd prometheus*/
echo "### Move the Binary Files & Set Owner ###"
mv prometheus /usr/local/bin
mv promtool /usr/local/bin
chown prometheus:prometheus /usr/local/bin/prometheus
chown prometheus:prometheus /usr/local/bin/promtool
echo "### Move the Configuration Files & Set Owner ###"
mv consoles /etc/prometheus
mv consoles/libraries /etc/prometheus
```

ScriptServidor2.sh



The screenshot shows the Kate code editor with the file "scriptServidor2.sh" open. The code is a Bash script for setting up Docker. It includes commands for updating the package list, installing curl, adding the Docker GPG key, updating the sources.list.d file, installing Docker CE, and adding the current user to the docker group. The code is color-coded for syntax.

```
#!/bin/bash
apt-get update
apt-get install -y ca-certificates curl
echo "### Install Docker ###"
install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/
docker.asc
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 $(lsb_release -cs) \
$VERSION_CODENAME" | sudo tee /etc/apt/sources.list.d/
docker.list > /dev/null
apt-get update
apt-get install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin
docker-compose-plugin
sudo usermod -aG docker $USER
echo "### Install Prometheus Node Exporter ###"
wget https://github.com/prometheus/node_exporter/releases/download/v1.9.1/
node_exporter-1.9.1.linux-amd64.tar.gz
tar -xvf node_exporter-*.tar.gz
mv node_exporter-1.9.1.linux-amd64/node_exporter /usr/local/bin
```

/etc/prometheus/prometheus.yml

The screenshot shows a terminal window titled "vagrant@servidor1Prometheus: ~" with the command "/var/home/cami/Documentos/GitHub/uni-telematic-homeworks/Parcial 3 Docker..." in the title bar. The terminal is running the "nano" editor on the file "/etc/prometheus/prometheus.yml". The file content is a Prometheus configuration file:

```
GNU nano 6.2          /etc/prometheus/prometheus.yml *
- targets:
  # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'evaluator' configuration
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped by this configuration
  - job_name: "prometheus"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["localhost:9090"]

  - job_name: "servidor-aws"
    static_configs:
      - targets: ["18.207.4.137:9100"]
```

At the bottom of the terminal window, there is a menu of keyboard shortcuts:

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line

\$ sudo systemctl restart prometheus

The screenshot shows the Prometheus Targets page. At the top, there are tabs for Prometheus, Grafana, 192.168.56.3:91, and User Management. The URL is http://192.168.56.2:9090/targets. The page has a dark theme with a navigation bar at the top.

Targets

Filter: 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	

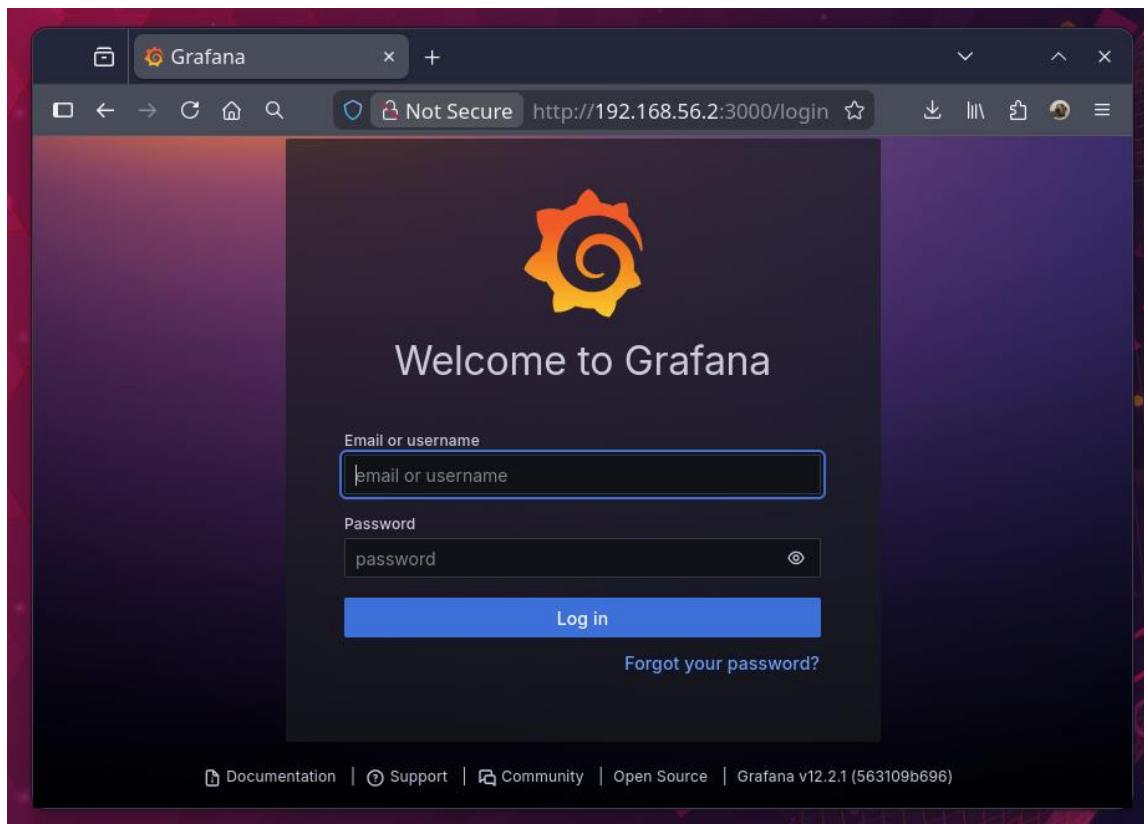
servidor-aws (1/1 up) [show less](#)

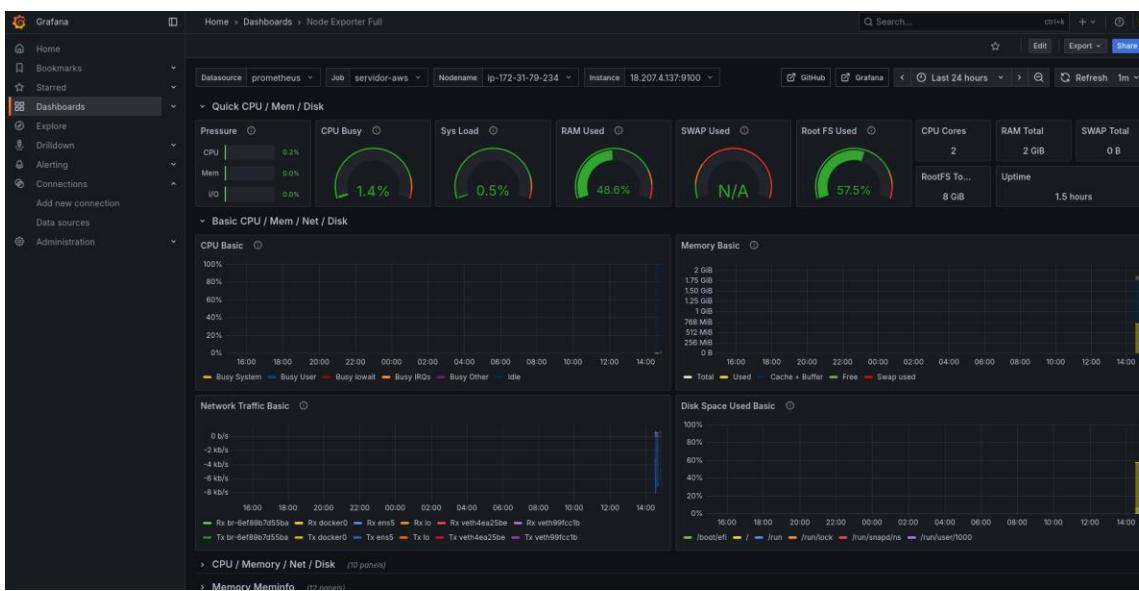
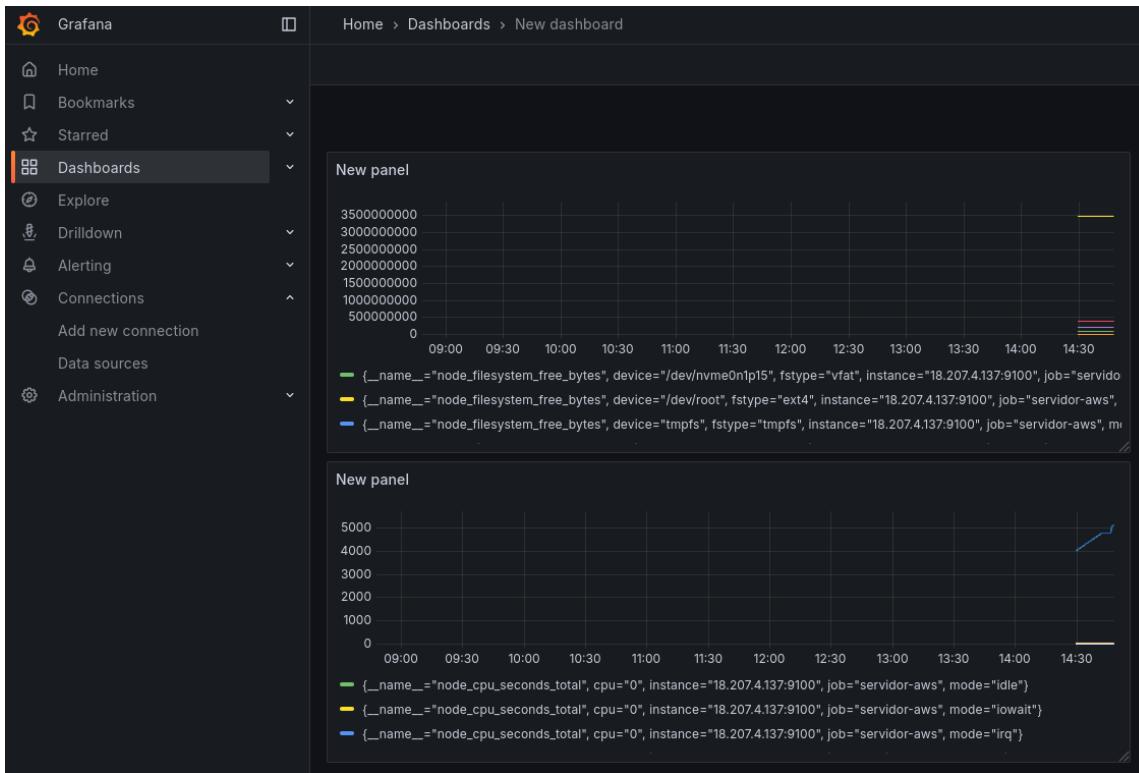
Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://18.207.4.137:9100/metrics	UP	instance="18.207.4.137:9100" job="servidor-aws"	12.37s ago	131.644ms	

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](#)





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.