La prueba será similar a estos últimos ejercicios:

instalación en una máquina virtual de un servicio apache+php

$ vagrant init bento/ubuntu-20.04

Ir a vagrantfile y modificar:

agrant.configure("2") do |config|

config.vm.box = "bento/ubuntu-20.04"

config.vm.network "forwarded\_port", guest: 80, host: 8080, host\_ip: "127.0.0.1"

end

vagrant up

vagrant ssh

sudo apt update

mariadb

Instalación del servidor de base de datos

sudo apt install mariadb-server

Apache y PHP

Instalación de apache2 y del módulo que permite que apache2 interprete PHP (apache2 hará el papel de servidor web y de servidor de aplicaciones).

sudo apt install apache2 libapache2-mod-php php php-mysql

### Configuración de PHP

Archivos de configuración de PHP (según versión. En este caso 7.4):

* /etc/php/7.4/cli: Configuración de php para php7.4-cli (ejecución de php desde la línea de comandos)
* /etc/php/7.4/apache2: Configuración de php para apache2 usado como módulo
* /etc/php/7.4/apache2/php.ini: Configuración de php
* /etc/php/7.4/fpm: Configuración de php para php-fpm
* /etc/php/7.4/mods-available: Módulos disponibles de php

### Comprobación

Creamos un fichero info.php en el documentRoot (/var/www/html ?) con el siguiente contenido:

**<?**php phpinfo**();** *?>*

apache como proxy +gunicorn (o los dos). Bien documentado todo:

Crear la maquina virtual

$ vagrant init bento/ubuntu-20.04

Ir a vagrantfile y modificar:

agrant.configure("2") do |config|

config.vm.box = "bento/ubuntu-20.04"

config.vm.network "forwarded\_port", guest: 80, host: 8080

config.vm.network "forwarded\_port", guest: 8080, host: 8080

end

$ vagrant up

$ vagrant status

$ vagrant ssh

$ cd /vagrant/

Para instalar desde git:

$ git clone Y el enlace https

$ cd y el nombre del git

Como se ejecuta una plicacion con python

$ Python3 -m venv env

$ source env/bin/actívate

Instalar la aplicacion

$cat requiremnts.txt

$ pio install flask lxml

Instalar servidor web gunicorn

source env/bin/activate

pip install gunicorn

**(**env**)** /home/vagrant/flask\_temperaturas$ gunicorn -w **2** -b :8080 wsgi

Como se utiliza:

Lanzar la aplicaciion

$ flask run

$ gunicorn app(nombre de la aplicación que aparecerá como app: nombre)

Control c para parar

$ gunicorn app:app -b :8080

(Con menos -b decimos el puerto )

$cat run.sh (Ejecuta la aplicación si estamos dentro de la carpeta)

Cambiar el entorno virtual al home de vagrant

Pwd

$ ls /vagrant

Programar gunicorn como un servicio

$ cd /etc/systemd/system/

$ sudo nano gunicorn-temperaturas.service

Dentro metos:

[Unit]

Description=gunicorn-temperaturas

After=network.target

[Install]

WantedBy=multi-user.target

[Service]

User=www-data

Group=www-data

Restart=always

ExecStart=/home/debian/venv/flask/bin/gunicorn -w 2 -b :8080 wsgi

ExecReload=/bin/kill -s HUP $MAINPID

ExecStop=/bin/kill -s TERM $MAINPID

WorkingDirectory=/home/debian/flask\_temperaturas

Environment=PYTHONPATH='/home/debian/flask\_temperaturas:/home/debian/venv/flask/lib/python3.9/site-packages'

PrivateTmp=true

Activamos la unidad de systemd, y la inciamos:

systemctl enable gunicorn-temperaturas.service

systemctl start gunicorn-temperaturas.service

Si cambias el contenido de la unidad tendrás que hacer la recarga:

systemctl daemon-reload

Activamos el módulo proxy\_http y en la configuración de algún virtualhost:

DocumentRoot /home/debian/flask\_temperaturas

ProxyPass / http://127.0.0.1:8080/

ProxyPassReverse / http://127.0.0.1:8080/

<Directory /home/debian/flask\_temperaturas>

Require all granted

</Directory>

En la configuración de un virtualhost:

location / {

proxy\_pass http://localhost:8080;

include proxy\_params;

}

Configuarar apache como un proxy inverso:

Para habilitar proxy de apache

$ sudo a2enmod proxy proxy\_http

$sudo nano /etc/apache/sites-available/temperaturas.conf

$sudo a2ensite temperaturas (Habilitar el sitio)

$sudo service apache reload (Reinicar apache)

$cd /vagrant/flask\_temperaturas/

$ls (ver los archivos)

$cat run.sh

Arranca un archivo que tiene dentro: gunicorn app:app -b 8080

$ sh run.sh

$vagrant halt (parar maquina virtual)

$nano vagrantfile (modificar el archivo vagrantfile)

Lo comentamos con #

#config.vm.network "forwarded\_port", guest: 8080, host: 8080

Cmd: vagrant up

Vagrant ssh

$ source env/bin/actívate

$cd /vagrant/flask\_temperaturas/

$ sh run.sh

Control + c parar el servidor

$ cd /etc/apache2/sites-available/

$sudo nano temperaturas.conf

Encima de donde tenemos configurado el proxy decirle que los datos estáticos no los sirva gunicorn que lo sirva aoache, atraves de:

Proxy/static/!

Debajo del proxypass

Alias/static/ /vagrant/flask\_temperaturas/static/

Damos permiso

<Directory /vagrant/flask\_temperaturas/static>

Require all granted

</Directory>

Y reinicar apache

$ sudo systemctl restart apache2

Comandos Linux: c

-Crear directorio : **mkdir**

**-Crar un fichero o un archivo:** $ sudo nano (y el nombre)

**-**