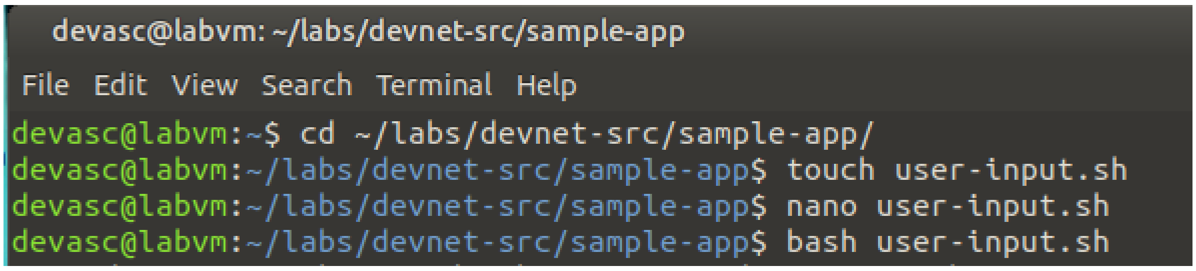
Práctica de laboratorio 6a: Crear una aplicación web de muestra en un contenedor Docker

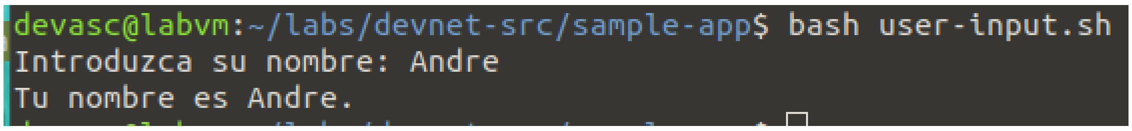
Scripts de bash

Creando script bash, editándolo y ejecutándolo.


devasc@labvm: ~/labs/devnet-src/sample-app
File Edit View Search Terminal Help

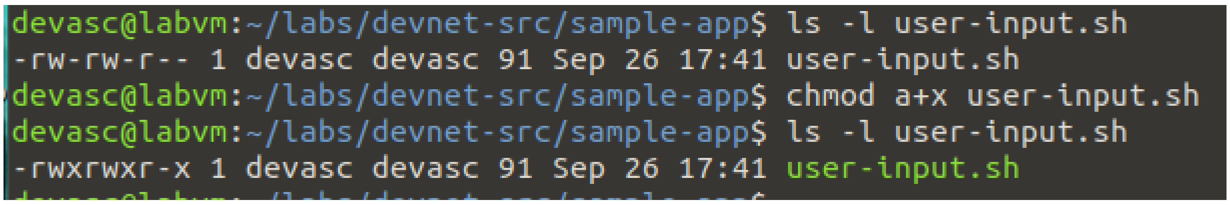
```
devasc@labvm:~$ cd ~/labs/devnet-src/sample-app/
devasc@labvm:~/labs/devnet-src/sample-app$ touch user-input.sh
devasc@labvm:~/labs/devnet-src/sample-app$ nano user-input.sh
devasc@labvm:~/labs/devnet-src/sample-app$ bash user-input.sh
```

Ejecución:



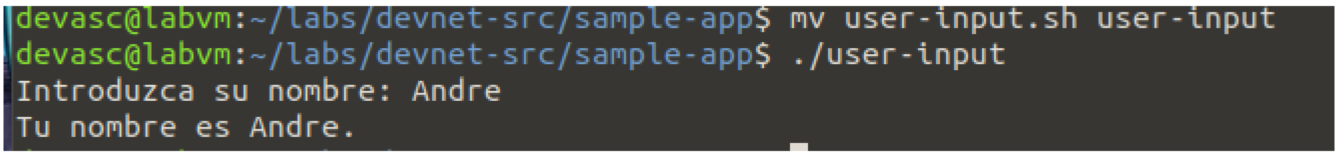
```
devasc@labvm:~/labs/devnet-src/sample-app$ bash user-input.sh
Introduzca su nombre: Andre
Tu nombre es Andre.
```

Cambiando los permisos del archivo con la flag a+x para que sea ejecutable desde todos los usuarios:



```
devasc@labvm:~/labs/devnet-src/sample-app$ ls -l user-input.sh
-rw-rw-r-- 1 devasc devasc 91 Sep 26 17:41 user-input.sh
devasc@labvm:~/labs/devnet-src/sample-app$ chmod a+x user-input.sh
devasc@labvm:~/labs/devnet-src/sample-app$ ls -l user-input.sh
-rwxrwxr-x 1 devasc devasc 91 Sep 26 17:41 user-input.sh
```

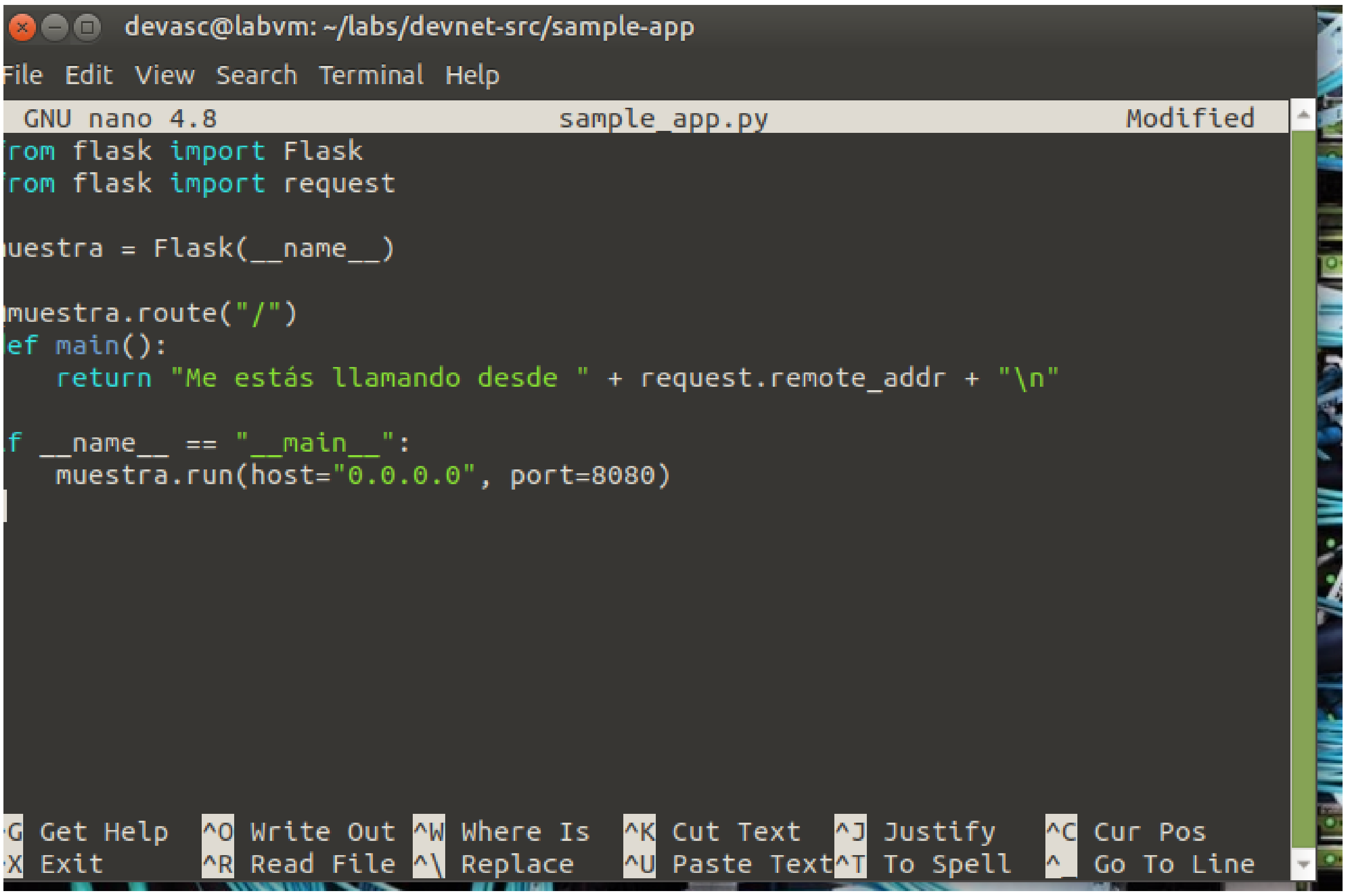
Cambiando nombre del archivo para eliminar la extensión .sh. Ahora se puede ejecutar directamente desde la terminal con ./user-input:



```
devasc@labvm:~/labs/devnet-src/sample-app$ mv user-input.sh user-input
devasc@labvm:~/labs/devnet-src/sample-app$ ./user-input
Introduzca su nombre: Andre
Tu nombre es Andre.
```

Crear App web de muestra

Editando archivo con vim y creando una app web de muestra en python con Flask:

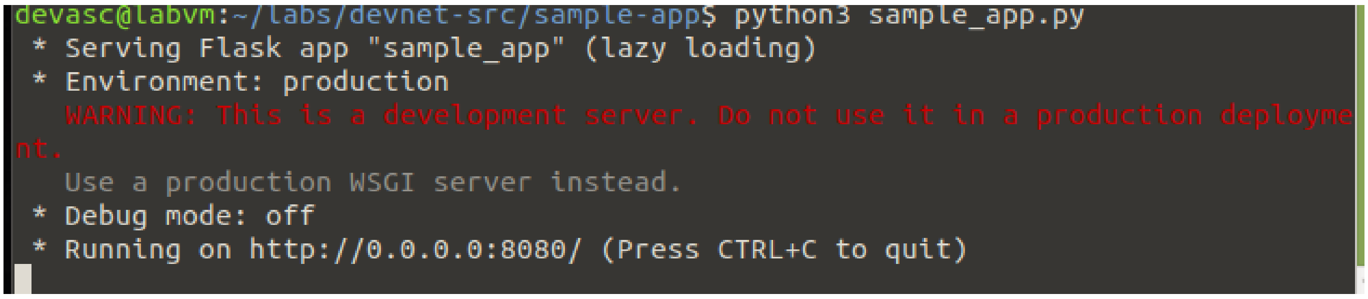


```
devasc@labvm: ~/labs/devnet-src/sample-app
File  Edit  View  Search  Terminal  Help
  GNU nano 4.8                  sample_app.py                   Modified
rom flask import Flask
rom flask import request

uestra = Flask(__name__)

muestra.route("/")
ef main():
   return "Me estás llamando desde " + request.remote_addr + "\n"

f __name__ == "__main__":
   muestra.run(host="0.0.0.0", port=8080)


G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
X Exit       ^R Read File  ^\ Replace    ^U Paste Text^T To Spell    ^  Go To Line
```

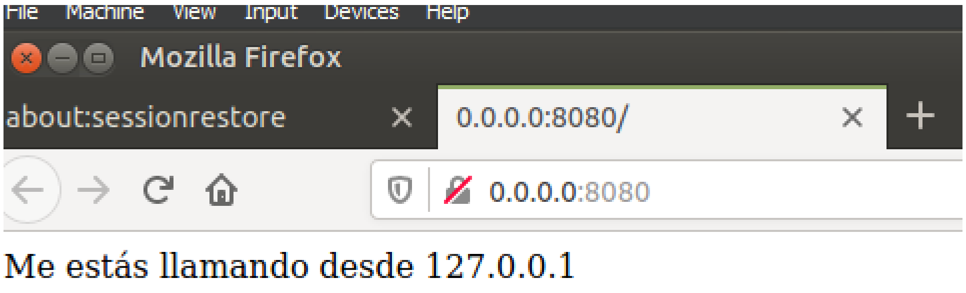
Código:

```
from flask import Flask
from flask import request

muestra = Flask(__name__)

@muestra.route("/")
def main():
    return "Me estás llamando desde " + request.remote_addr + "\n"

if __name__ == "__main__":
    muestra.run(host="0.0.0.0", port=8080)
```

Ejecutando el archivo .py:



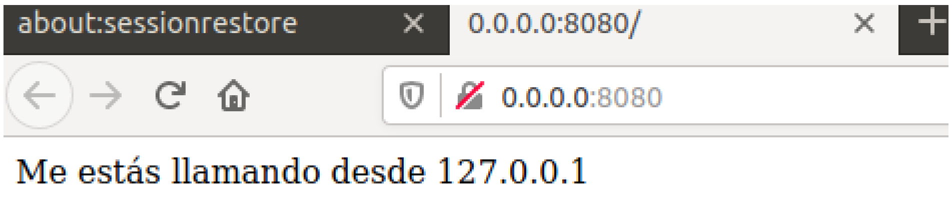
```
devasc@labvm:~/labs/devnet-src/sample-app$ python3 sample_app.py
 * Serving Flask app "sample_app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployme
nt.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)
```



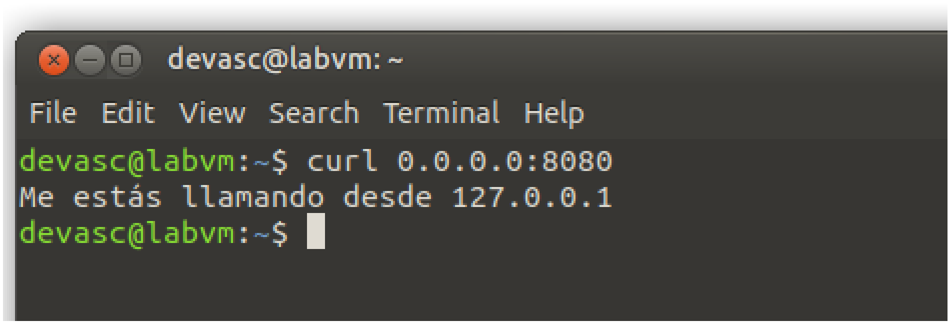
Me estás llamando desde 127.0.0.1

Interfaz web:

Asegurándonos de que la app funcione correctamente con curl:

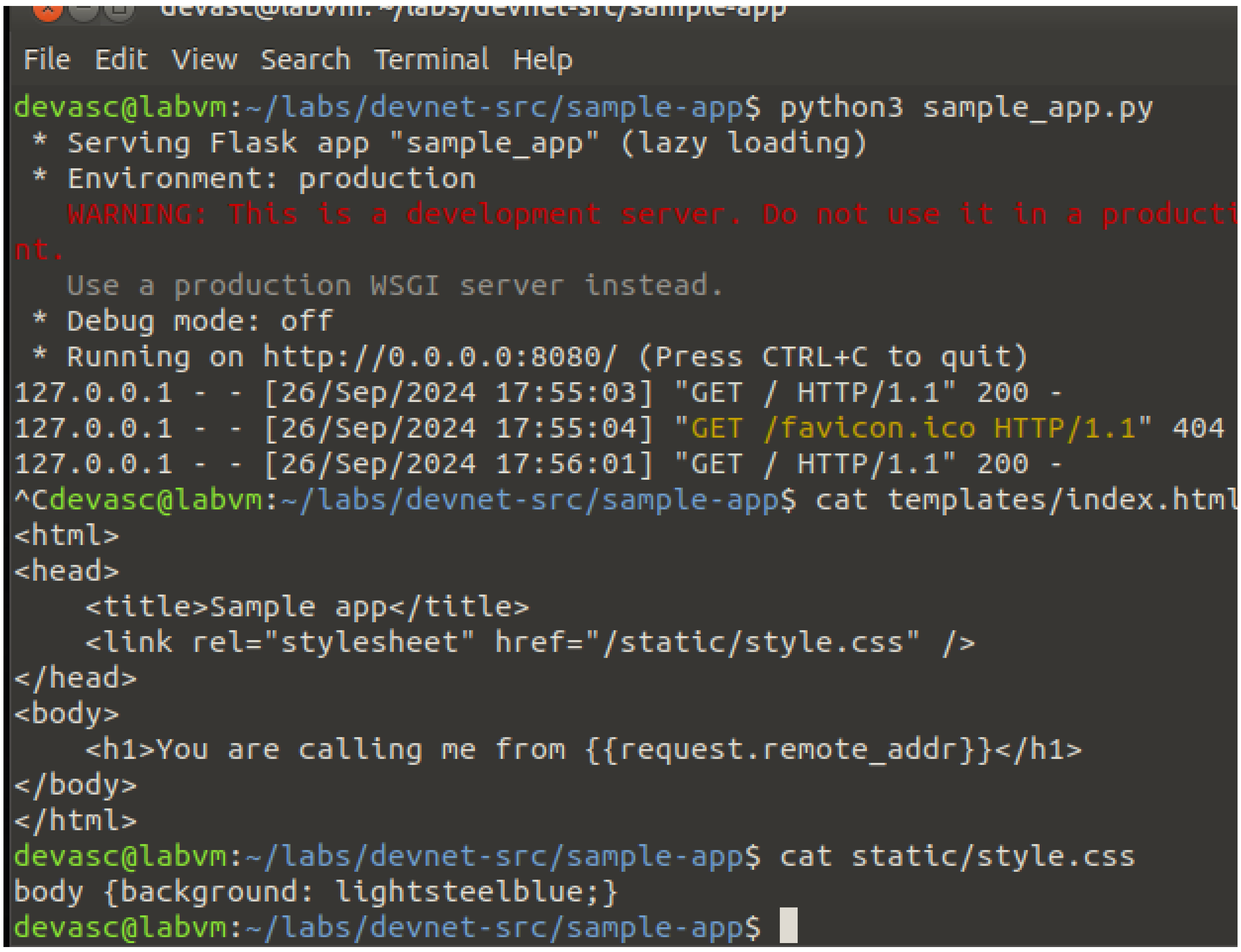


Me estás llamando desde 127.0.0.1



```
devasc@labvm:~$ curl 0.0.0.0:8080
Me estás llamando desde 127.0.0.1
devasc@labvm:~$
```

Viendo index.html y style.css



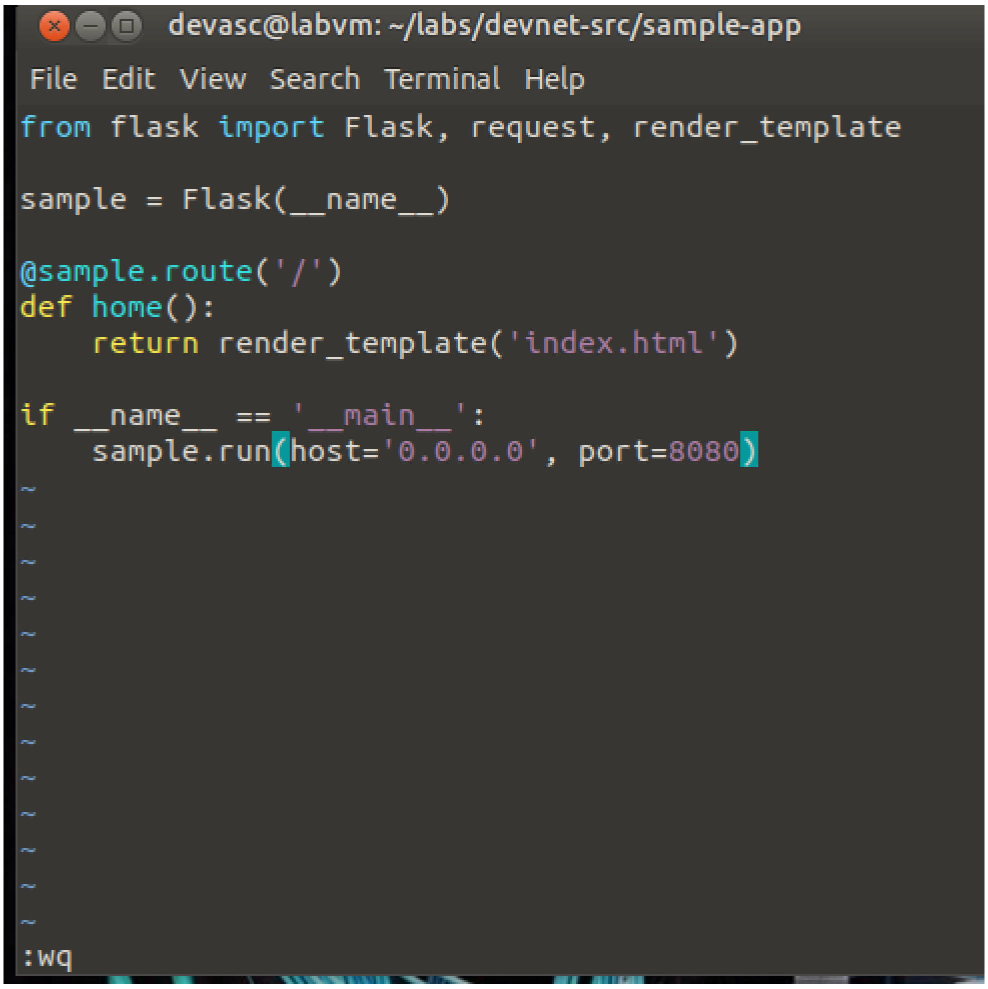
```
File  Edit  View  Search  Terminal  Help
devasc@labvm:~/labs/devnet-src/sample-app$ python3 sample_app.py
 * Serving Flask app "sample_app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a producti
nt.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)
127.0.0.1 - - [26/Sep/2024 17:55:03] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [26/Sep/2024 17:55:04] "GET /favicon.ico HTTP/1.1" 404
127.0.0.1 - - [26/Sep/2024 17:56:01] "GET / HTTP/1.1" 200 -
^Cdevasc@labvm:~/labs/devnet-src/sample-app$ cat templates/index.html
<html>
<head>
    <title>Sample app</title>
    <link rel="stylesheet" href="/static/style.css" />
</head>
<body>
    <h1>You are calling me from {{request.remote_addr}}</h1>
</body>
</html>
devasc@labvm:~/labs/devnet-src/sample-app$ cat static/style.css
body {background: lightsteelblue;}
devasc@labvm:~/labs/devnet-src/sample-app$
```

Renderizando index.html y style.css con flask

Código:

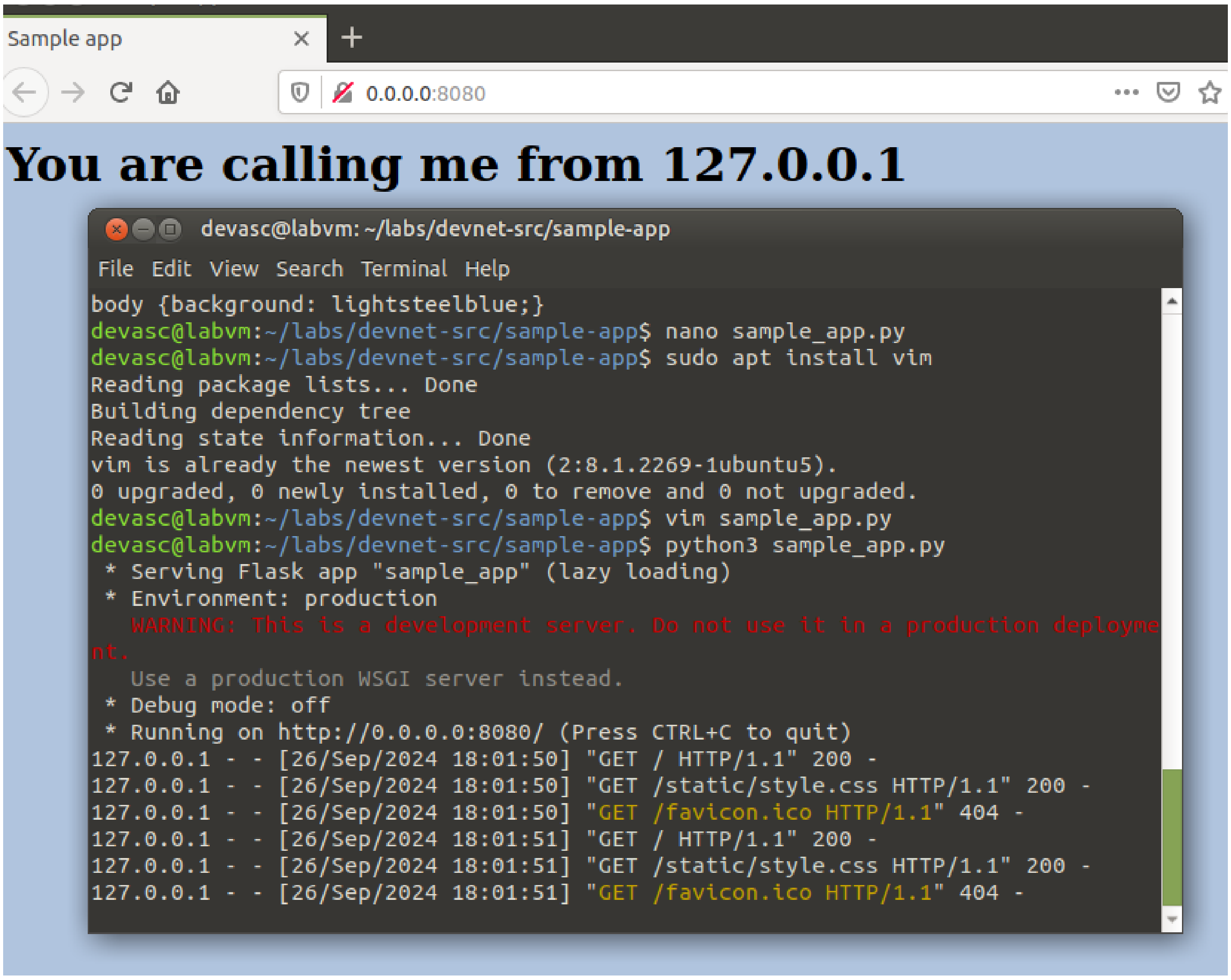
```
from flask import Flask, request, render_template

sample = Flask(__name__)

@sample.route('/')
def home():
    return render_template('index.html')

if __name__ == '__main__':
    sample.run(host='0.0.0.0', port=8080)
```

Me cansé de usar nano así que empecé a usar vim.


devasc@labvm: ~/labs/devnet-src/sample-app
File Edit View Search Terminal Help

```
from flask import Flask, request, render_template

sample = Flask(__name__)

@sample.route('/')
def home():
    return render_template('index.html')

if __name__ == '__main__':
    sample.run(host='0.0.0.0', port=8080)
~
~
~
~
~
~
~
~
~
~
~
~
~
:wq
```

Ejecutando el archivo:



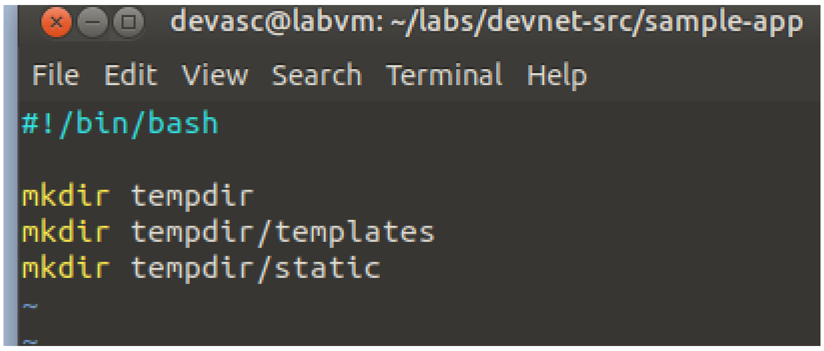
You are calling me from 127.0.0.1

devasc@labvm: ~/labs/devnet-src/sample-app
File Edit View Search Terminal Help

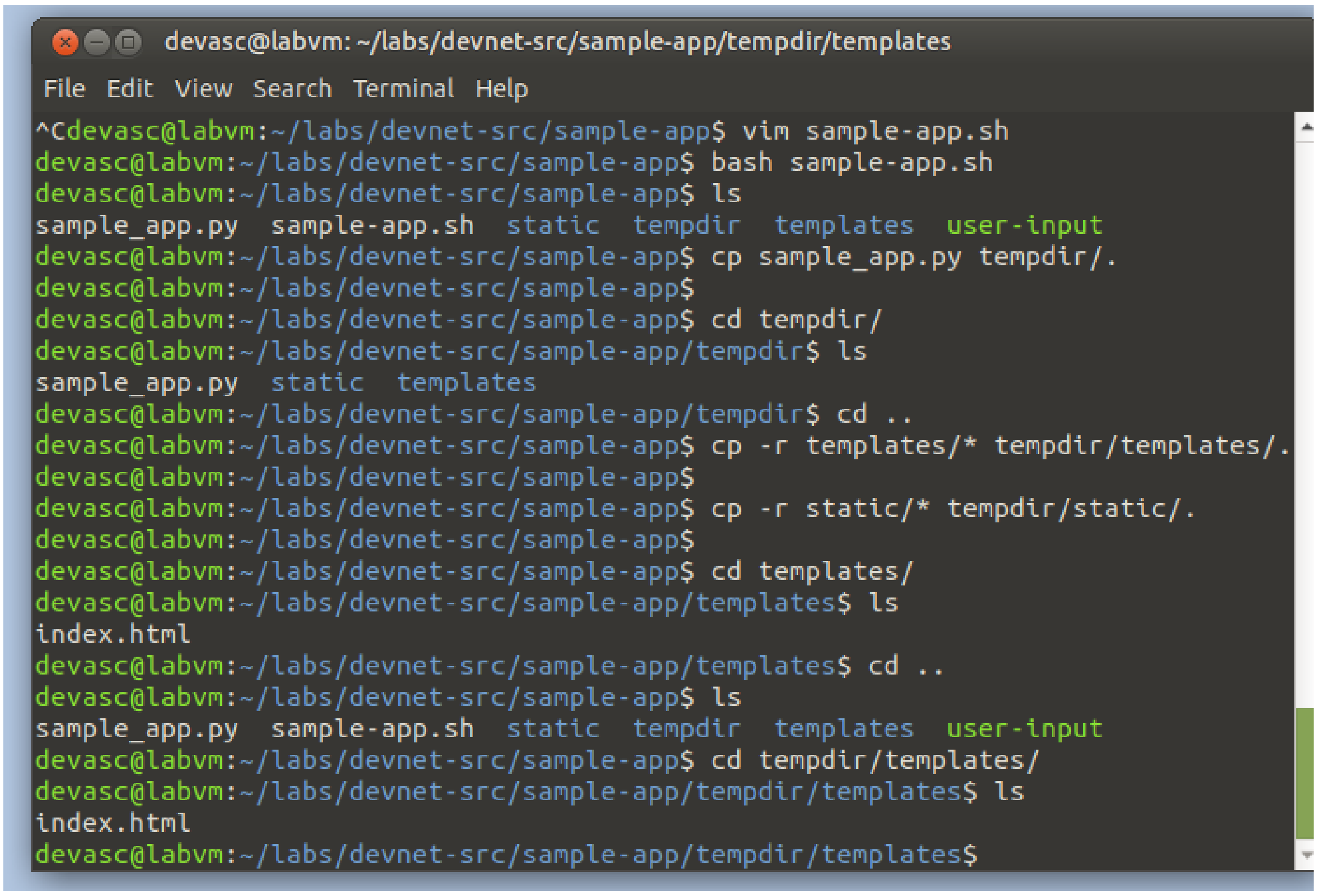
```
body {background: lightsteelblue;}
devasc@labvm:~/labs/devnet-src/sample-app$ nano sample_app.py
devasc@labvm:~/labs/devnet-src/sample-app$ sudo apt install vim
Reading package lists... Done
Building dependency tree
Reading state information... Done
vim is already the newest version (2:8.1.2269-1ubuntu5).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
devasc@labvm:~/labs/devnet-src/sample-app$ vim sample_app.py
devasc@labvm:~/labs/devnet-src/sample-app$ python3 sample_app.py
 * Serving Flask app "sample_app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployme
nt.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)
127.0.0.1 - - [26/Sep/2024 18:01:50] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [26/Sep/2024 18:01:50] "GET /static/style.css HTTP/1.1" 200 -
127.0.0.1 - - [26/Sep/2024 18:01:50] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [26/Sep/2024 18:01:51] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [26/Sep/2024 18:01:51] "GET /static/style.css HTTP/1.1" 200 -
127.0.0.1 - - [26/Sep/2024 18:01:51] "GET /favicon.ico HTTP/1.1" 404 -
```

Crear un script de Bash para compilar y ejecutar un contenedor Docker

Creando bash para crear las carpetas temporales y copiar los archivos necesarios


devasc@labvm: ~/labs/devnet-src/sample-app
File Edit View Search Terminal Help

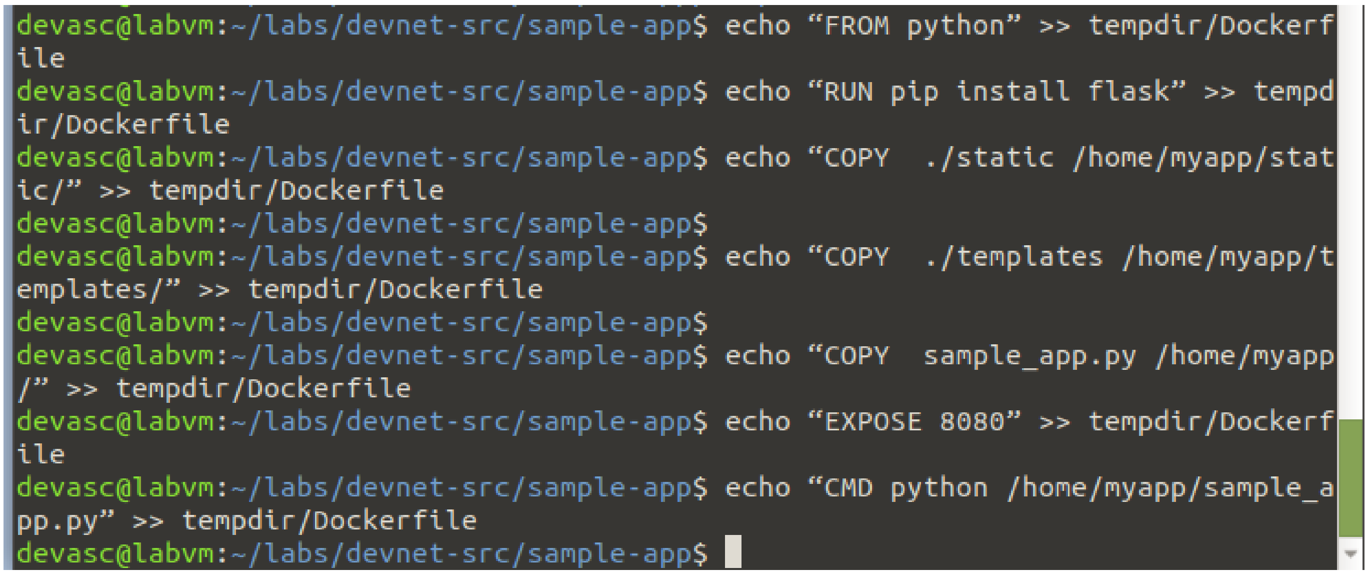
```
#!/bin/bash

mkdir tempdir
mkdir tempdir/templates
mkdir tempdir/static
~
~
```


devasc@labvm: ~/labs/devnet-src/sample-app/tempdir/templates
File Edit View Search Terminal Help

```
^Cdevasc@labvm:~/labs/devnet-src/sample-app$ vim sample-app.sh
devasc@labvm:~/labs/devnet-src/sample-app$ bash sample-app.sh
devasc@labvm:~/labs/devnet-src/sample-app$ ls
sample_app.py  sample-app.sh  static  tempdir  templates  user-input
devasc@labvm:~/labs/devnet-src/sample-app$ cp sample_app.py tempdir/.
devasc@labvm:~/labs/devnet-src/sample-app$
devasc@labvm:~/labs/devnet-src/sample-app$ cd tempdir/
devasc@labvm:~/labs/devnet-src/sample-app/tempdir$ ls
sample_app.py  static  templates
devasc@labvm:~/labs/devnet-src/sample-app/tempdir$ cd ..
devasc@labvm:~/labs/devnet-src/sample-app$ cp -r templates/* tempdir/templates/.
devasc@labvm:~/labs/devnet-src/sample-app$
devasc@labvm:~/labs/devnet-src/sample-app$ cp -r static/* tempdir/static/.
devasc@labvm:~/labs/devnet-src/sample-app$
devasc@labvm:~/labs/devnet-src/sample-app$ cd templates/
devasc@labvm:~/labs/devnet-src/sample-app/templates$ ls
index.html
devasc@labvm:~/labs/devnet-src/sample-app/templates$ cd ..
devasc@labvm:~/labs/devnet-src/sample-app$ ls
sample_app.py  sample-app.sh  static  tempdir  templates  user-input
devasc@labvm:~/labs/devnet-src/sample-app$ cd tempdir/templates/
devasc@labvm:~/labs/devnet-src/sample-app/tempdir/templates$ ls
index.html
devasc@labvm:~/labs/devnet-src/sample-app/tempdir/templates$
```

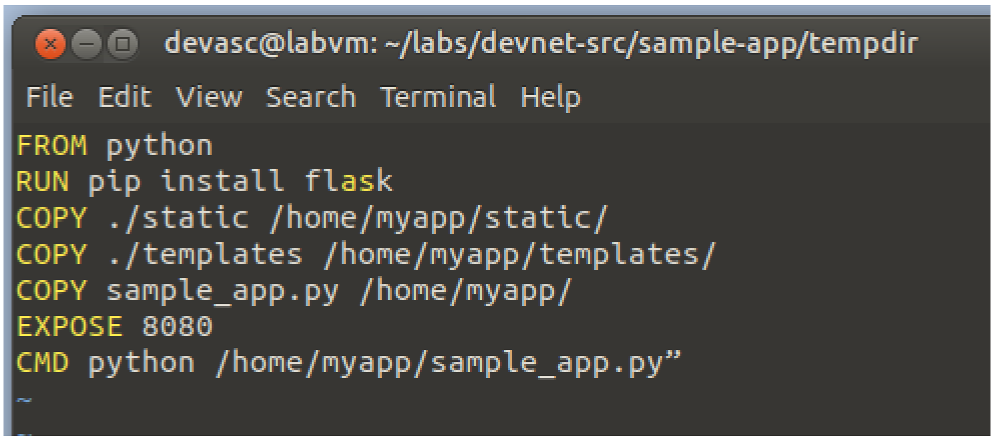
Creando Dockerfile

Creándolo con echo desde la terminal:



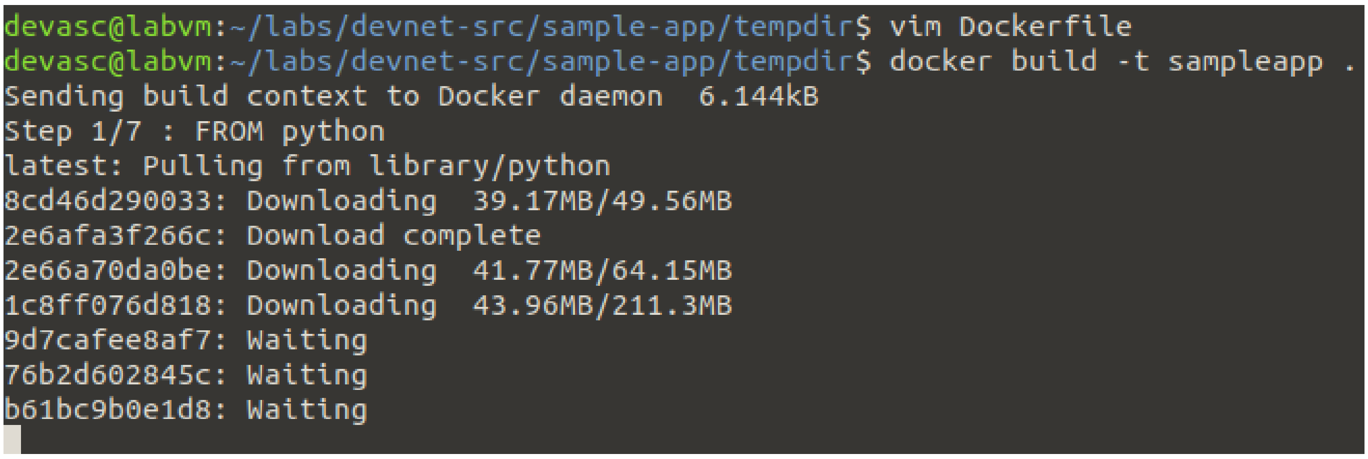
```
devasc@labvm:~/labs/devnet-src/sample-app$ echo “FROM python” >> tempdir/Dockerf
ile
devasc@labvm:~/labs/devnet-src/sample-app$ echo “RUN pip install flask” >> tempd
ir/Dockerfile
devasc@labvm:~/labs/devnet-src/sample-app$ echo “COPY  ./static /home/myapp/stat
ic/” >> tempdir/Dockerfile
devasc@labvm:~/labs/devnet-src/sample-app$
devasc@labvm:~/labs/devnet-src/sample-app$ echo “COPY  ./templates /home/myapp/t
emplates/” >> tempdir/Dockerfile
devasc@labvm:~/labs/devnet-src/sample-app$
devasc@labvm:~/labs/devnet-src/sample-app$ echo “COPY  sample_app.py /home/myapp
/” >> tempdir/Dockerfile
devasc@labvm:~/labs/devnet-src/sample-app$ echo “EXPOSE 8080” >> tempdir/Dockerf
ile
devasc@labvm:~/labs/devnet-src/sample-app$ echo “CMD python /home/myapp/sample_a
pp.py” >> tempdir/Dockerfile
devasc@labvm:~/labs/devnet-src/sample-app$
```

Así se ve el Dockerfile:


devasc@labvm: ~/labs/devnet-src/sample-app/tempdir
File Edit View Search Terminal Help

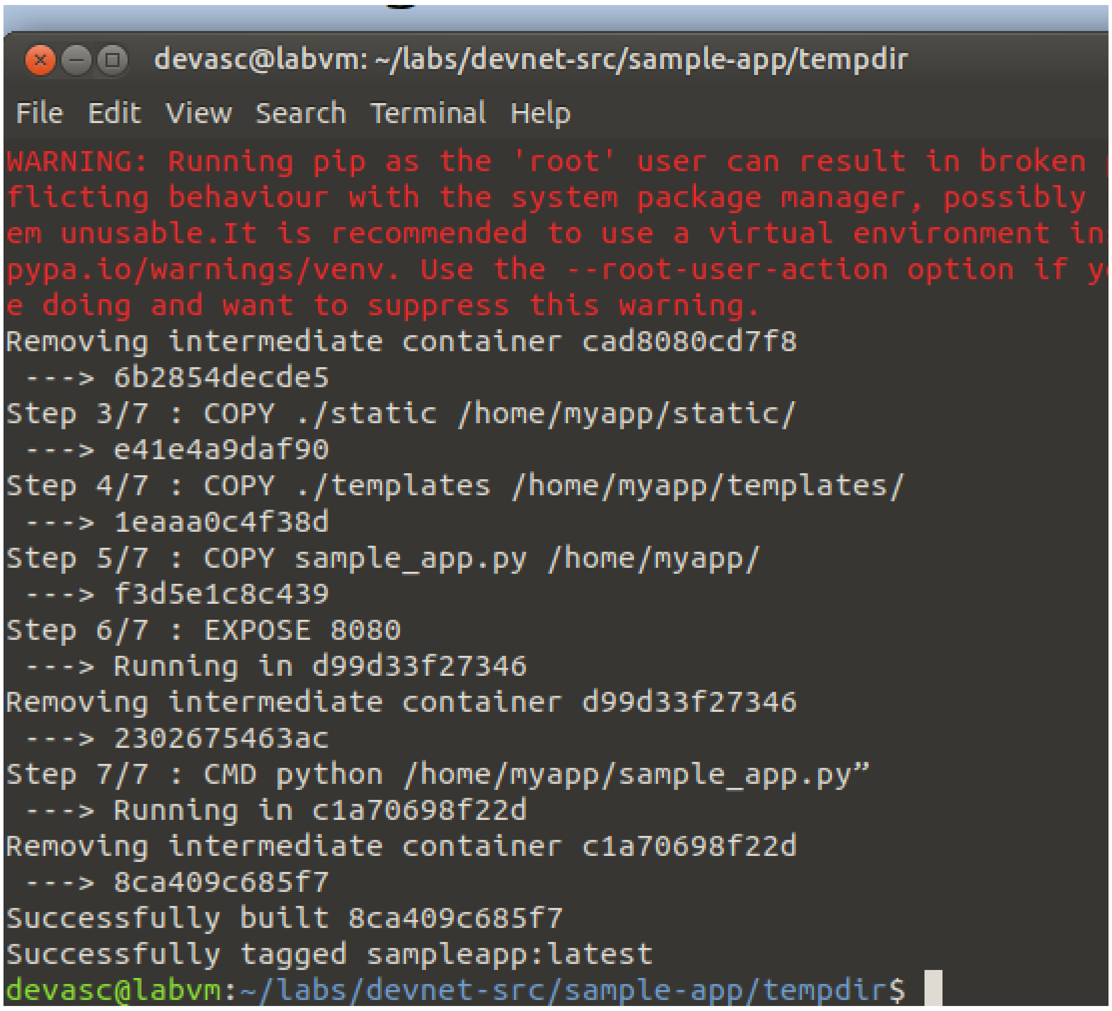
```
FROM python
RUN pip install flask
COPY ./static /home/myapp/static/
COPY ./templates /home/myapp/templates/
COPY sample_app.py /home/myapp/
EXPOSE 8080
CMD python /home/myapp/sample_app.py”
~
```

Construyendo el contenedor...



```
devasc@labvm:~/labs/devnet-src/sample-app/tempdir$ vim Dockerfile
devasc@labvm:~/labs/devnet-src/sample-app/tempdir$ docker build -t sampleapp .
Sending build context to Docker daemon  6.144kB
Step 1/7 : FROM python
latest: Pulling from library/python
8cd46d290033: Downloading  39.17MB/49.56MB
2e6afa3f266c: Download complete
2e66a70da0be: Downloading  41.77MB/64.15MB
1c8ff076d818: Downloading  43.96MB/211.3MB
9d7cafee8af7: Waiting
76b2d602845c: Waiting
b61bc9b0e1d8: Waiting
```

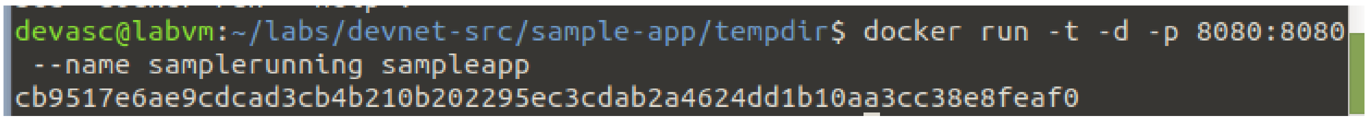
Al terminar:


devasc@labvm: ~/labs/devnet-src/sample-app/tempdir
File Edit View Search Terminal Help

```
WARNING: Running pip as the 'root' user can result in broken
flicting behaviour with the system package manager, possibly
em unusable.It is recommended to use a virtual environment in
pypa.io/warnings/venv. Use the --root-user-action option if y
e doing and want to suppress this warning.
Removing intermediate container cad8080cd7f8
 ---> 6b2854decde5
Step 3/7 : COPY ./static /home/myapp/static/
 ---> e41e4a9daf90
Step 4/7 : COPY ./templates /home/myapp/templates/
 ---> 1eaaa0c4f38d
Step 5/7 : COPY sample_app.py /home/myapp/
 ---> f3d5e1c8c439
Step 6/7 : EXPOSE 8080
 ---> Running in d99d33f27346
Removing intermediate container d99d33f27346
 ---> 2302675463ac
Step 7/7 : CMD python /home/myapp/sample_app.py”
 ---> Running in c1a70698f22d
Removing intermediate container c1a70698f22d
 ---> 8ca409c685f7
Successfully built 8ca409c685f7
Successfully tagged sampleapp:latest
devasc@labvm:~/labs/devnet-src/sample-app/tempdir$
```

Ahora, ejecutando el contenedor con:

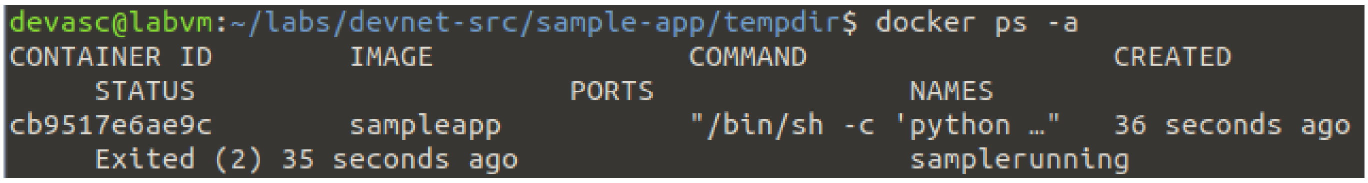
```
docker run -t -d -p 8080:8080 --name samplerunning sampleapp
```



```
devasc@labvm:~/labs/devnet-src/sample-app/tempdir$ docker run -t -d -p 8080:8080
 --name samplerunning sampleapp
cb9517e6ae9cdcad3cb4b210b202295ec3cdab2a4624dd1b10aa3cc38e8feaf0
```

Como ven nos devuelve la id.

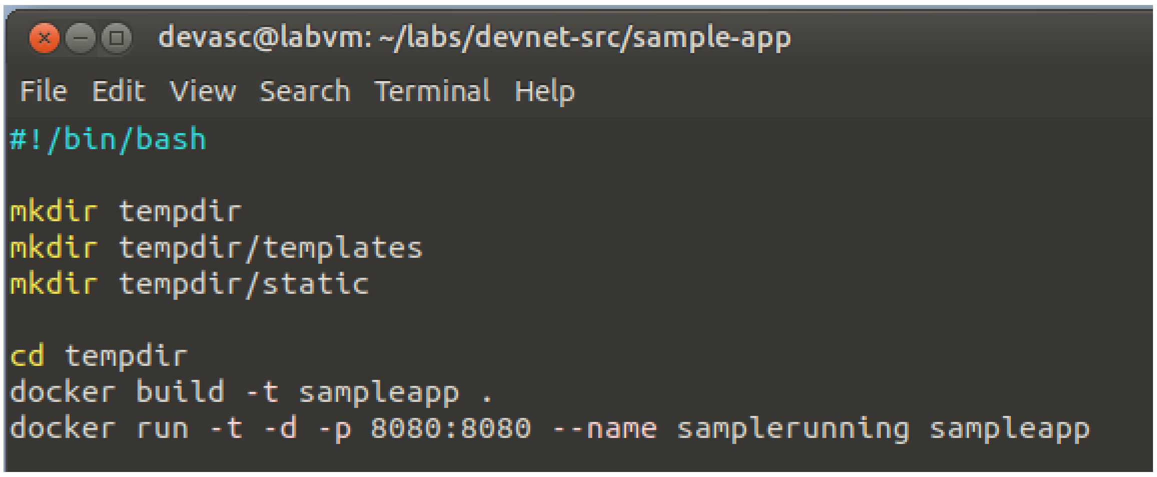
Viendo el estado del contenedor con `docker ps -a`:



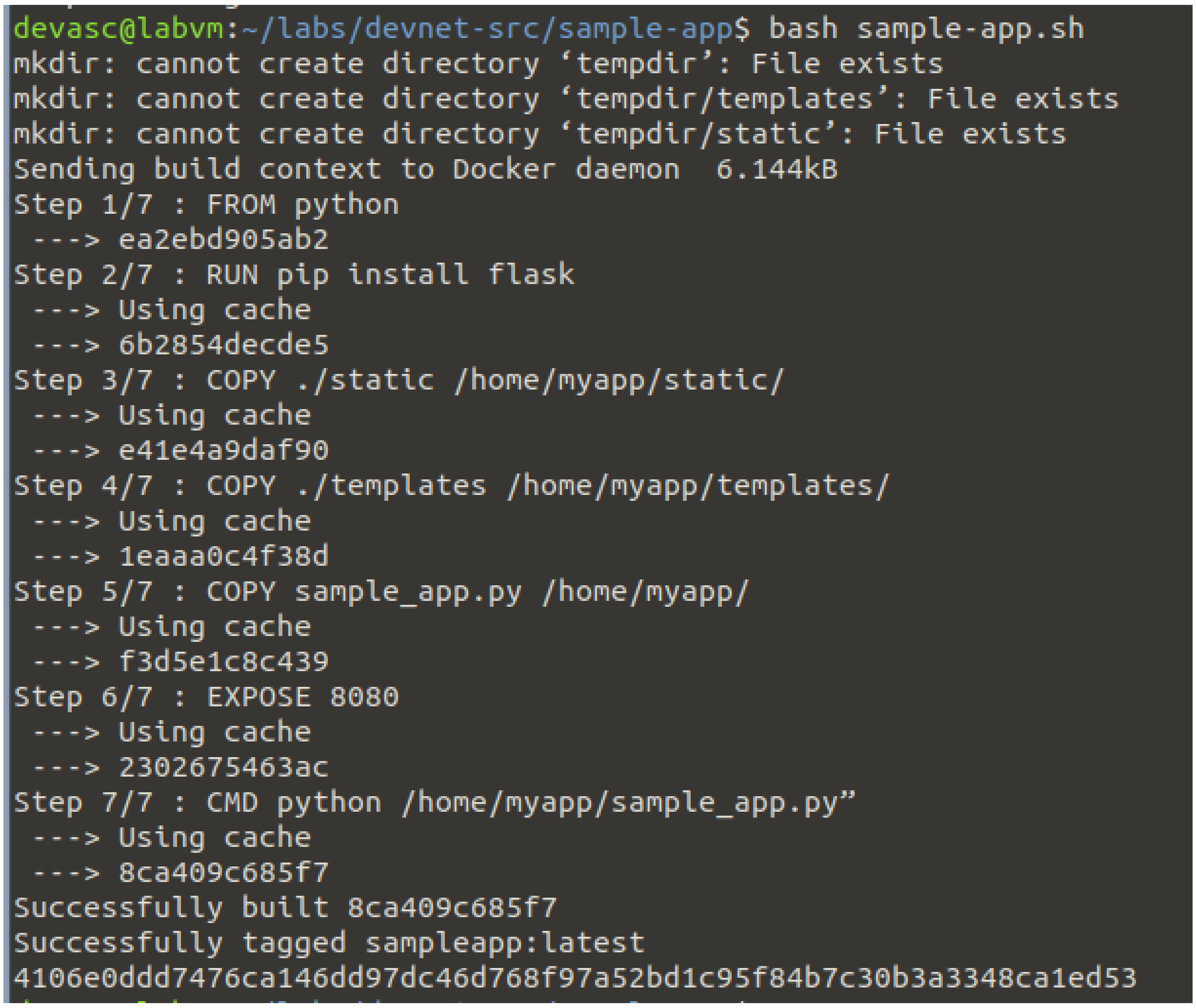
```
devasc@labvm:~/labs/devnet-src/sample-app/tempdir$ docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED
     STATUS                     PORTS               NAMES
cb9517e6ae9c        sampleapp           "/bin/sh -c 'python …"   36 seconds ago
     Exited (2) 35 seconds ago                      samplerunning
```

Exited (2) 35 seconds ago indica que el contenedor ha terminado su ejecución. El código de salida (2) generalmente indica un error.

Ejecutando script de bash

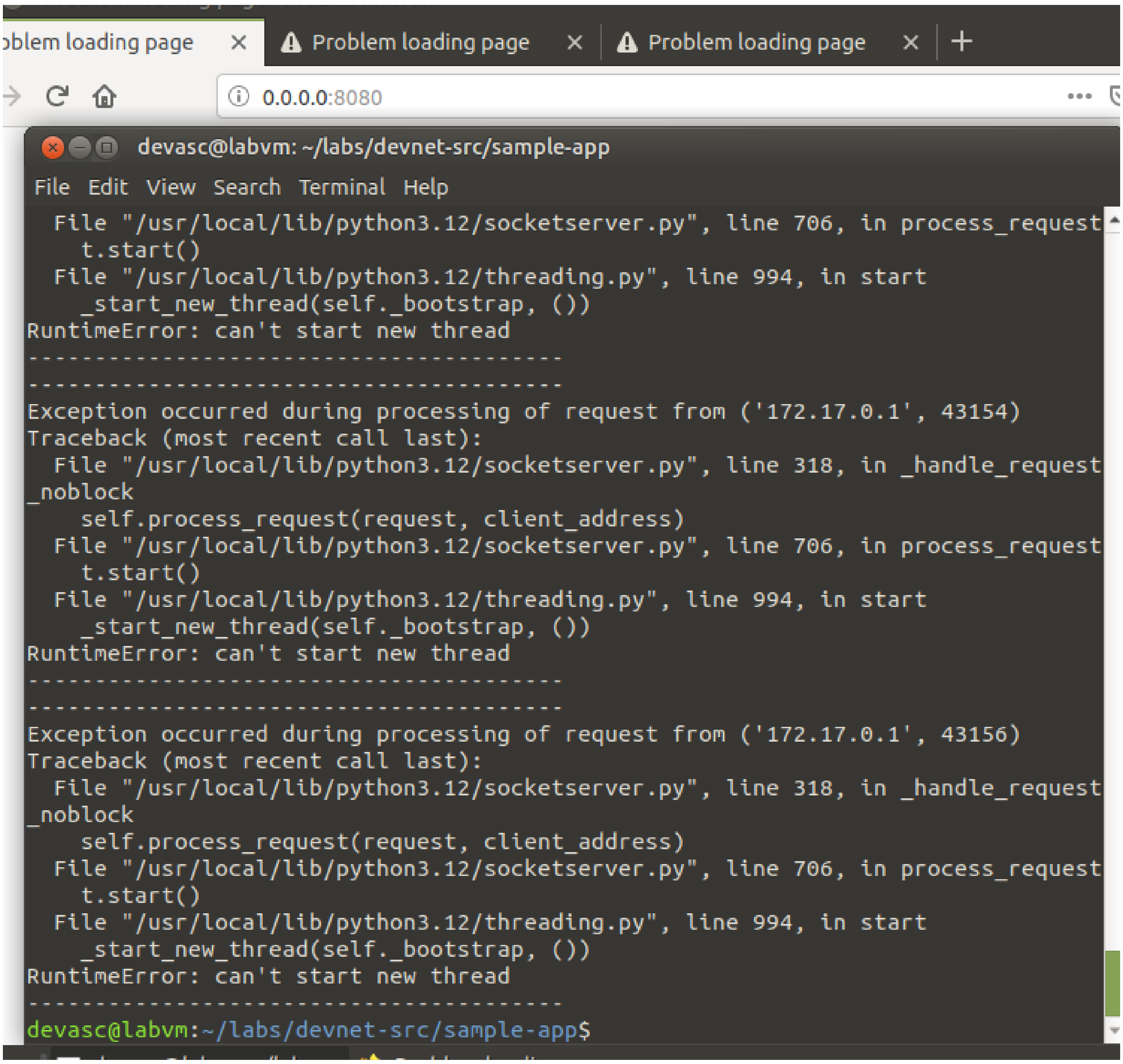

devasc@labvm: ~/labs/devnet-src/sample-app
File Edit View Search Terminal Help

```
#!/bin/bash

mkdir tempdir
mkdir tempdir/templates
mkdir tempdir/static

cd tempdir
docker build -t sampleapp .
docker run -t -d -p 8080:8080 --name samplerunning sampleapp
```



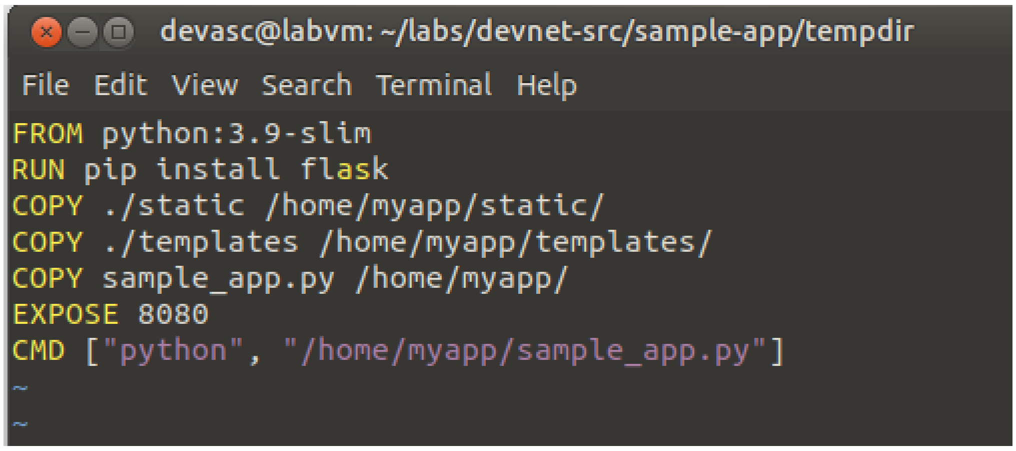
```
devasc@labvm:~/labs/devnet-src/sample-app$ bash sample-app.sh
mkdir: cannot create directory ‘tempdir’: File exists
mkdir: cannot create directory ‘tempdir/templates’: File exists
mkdir: cannot create directory ‘tempdir/static’: File exists
Sending build context to Docker daemon  6.144kB
Step 1/7 : FROM python
 ---> ea2ebd905ab2
Step 2/7 : RUN pip install flask
 ---> Using cache
 ---> 6b2854decde5
Step 3/7 : COPY ./static /home/myapp/static/
 ---> Using cache
 ---> e41e4a9daf90
Step 4/7 : COPY ./templates /home/myapp/templates/
 ---> Using cache
 ---> 1eaaa0c4f38d
Step 5/7 : COPY sample_app.py /home/myapp/
 ---> Using cache
 ---> f3d5e1c8c439
Step 6/7 : EXPOSE 8080
 ---> Using cache
 ---> 2302675463ac
Step 7/7 : CMD python /home/myapp/sample_app.py”
 ---> Using cache
 ---> 8ca409c685f7
Successfully built 8ca409c685f7
Successfully tagged sampleapp:latest
4106e0ddd7476ca146dd97dc46d768f97a52bd1c95f84b7c30b3a3348ca1ed53
```

Después de ejecutar el script, no cargaba la app. Vi el log del contenedor con `docker logs samplerunning`.


oblem loading page
Problem loading page
Problem loading page
0.0.0.0:8080
devasc@labvm: ~/labs/devnet-src/sample-app
File Edit View Search Terminal Help

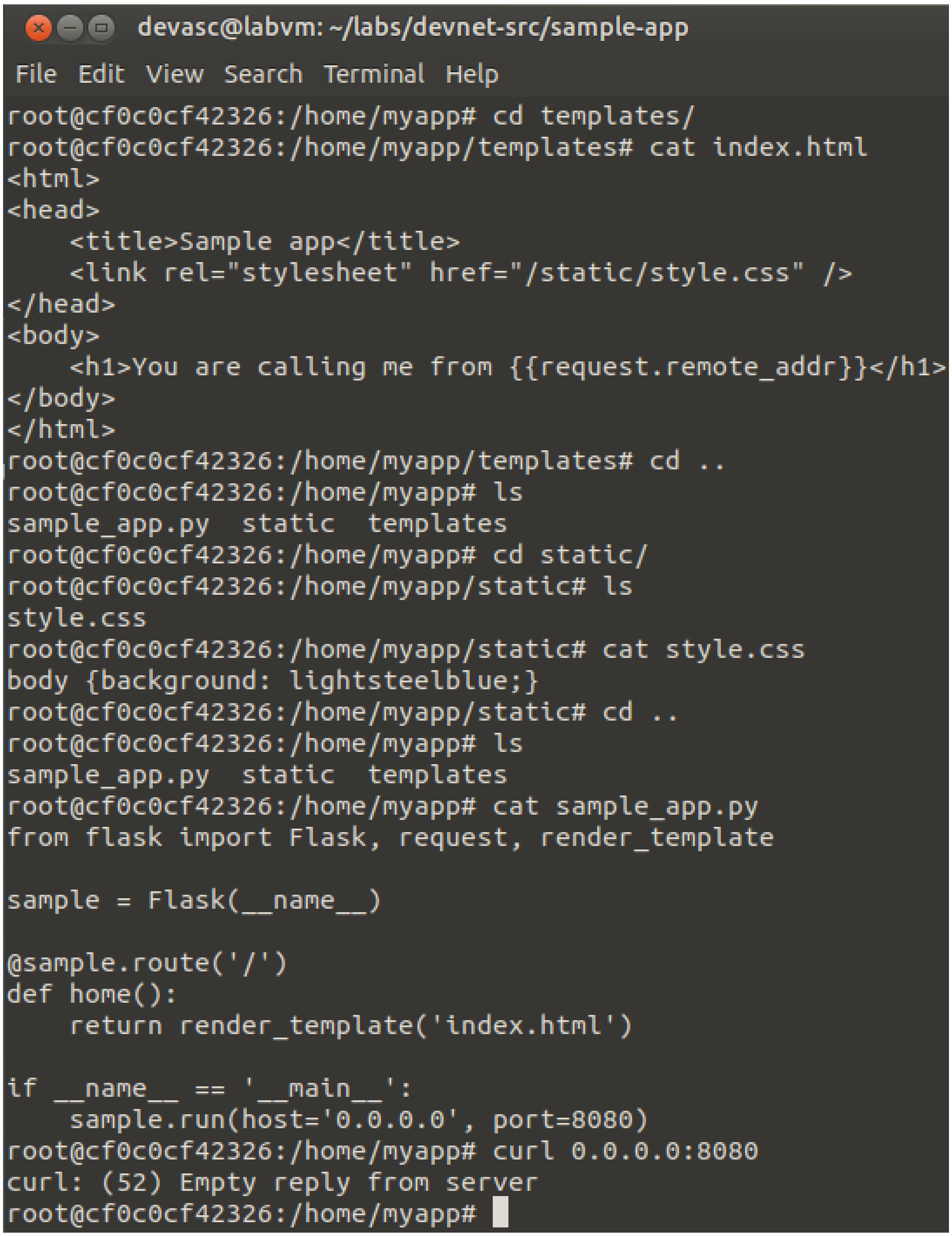
```
  File "/usr/local/lib/python3.12/socketserver.py", line 706, in process_request
    t.start()
  File "/usr/local/lib/python3.12/threading.py", line 994, in start
    _start_new_thread(self._bootstrap, ())
RuntimeError: can't start new thread
----------------------------------------
----------------------------------------
Exception occurred during processing of request from ('172.17.0.1', 43154)
Traceback (most recent call last):
  File "/usr/local/lib/python3.12/socketserver.py", line 318, in _handle_request
_noblock
    self.process_request(request, client_address)
  File "/usr/local/lib/python3.12/socketserver.py", line 706, in process_request
    t.start()
  File "/usr/local/lib/python3.12/threading.py", line 994, in start
    _start_new_thread(self._bootstrap, ())
RuntimeError: can't start new thread
----------------------------------------
----------------------------------------
Exception occurred during processing of request from ('172.17.0.1', 43156)
Traceback (most recent call last):
  File "/usr/local/lib/python3.12/socketserver.py", line 318, in _handle_request
_noblock
    self.process_request(request, client_address)
  File "/usr/local/lib/python3.12/socketserver.py", line 706, in process_request
    t.start()
  File "/usr/local/lib/python3.12/threading.py", line 994, in start
    _start_new_thread(self._bootstrap, ())
RuntimeError: can't start new thread
----------------------------------------
devasc@labvm:~/labs/devnet-src/sample-app$
```

Me di cuenta que la imagen Python no se había especificado en el Dockerfile, así que agregué la imagen Python:3.9-slim como base de la imagen. Además cambié la última línea del Dockerfile para que sea `CMD ["python", "/home/myapp/sample-app.py"]`.


devasc@labvm: ~/labs/devnet-src/sample-app/tempdir
File Edit View Search Terminal Help

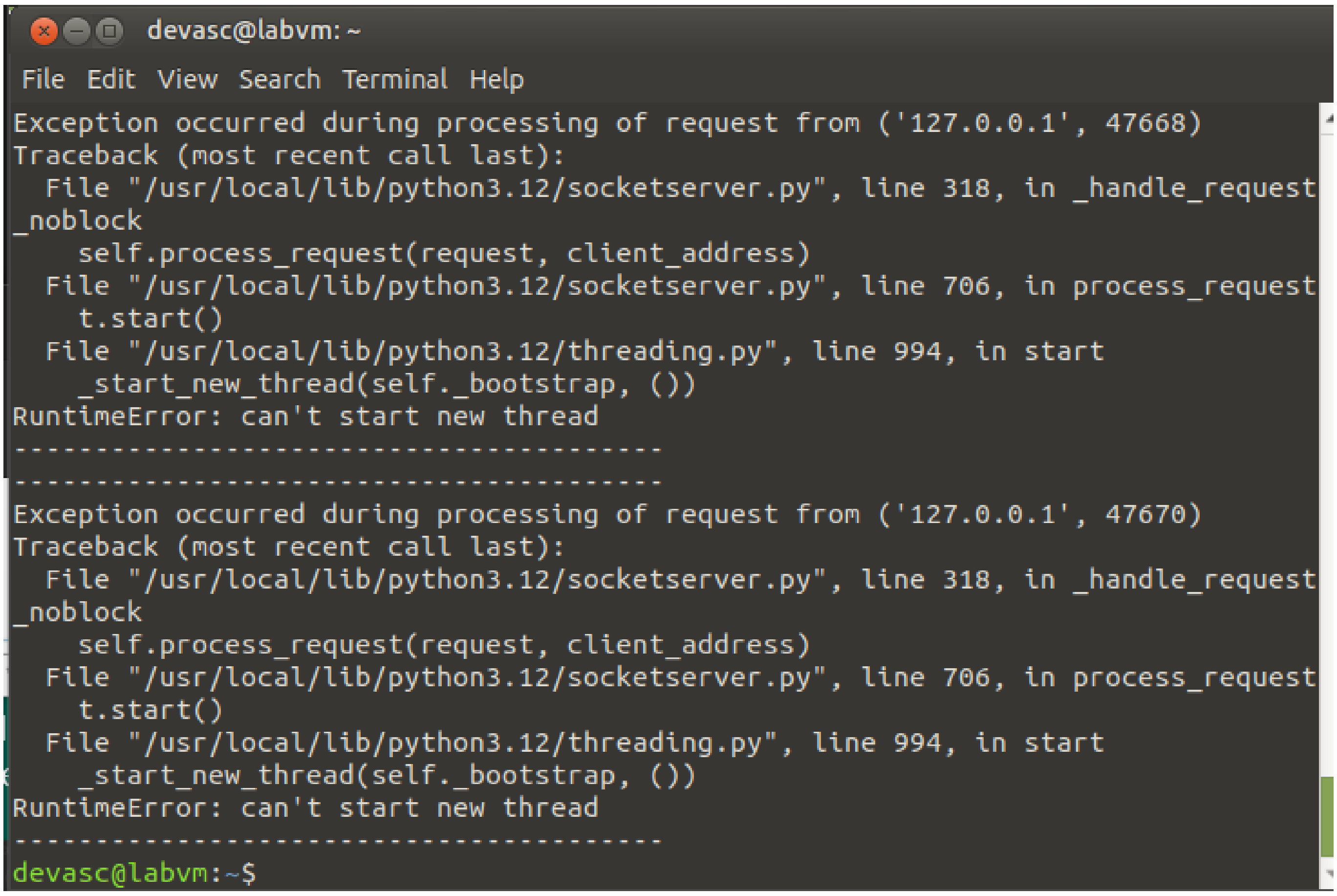
```
FROM python:3.9-slim
RUN pip install flask
COPY ./static /home/myapp/static/
COPY ./templates /home/myapp/templates/
COPY sample_app.py /home/myapp/
EXPOSE 8080
CMD ["python", "/home/myapp/sample_app.py"]
~
~
```

Luego de instanciar el contenedor con el nuevo Dockerfile, la app se cargó correctamente, pero todavía no se podía acceder a la interfaz web. Entré al contenedor con `docker exec -it samplerunning /bin/bash` y con curl me di cuenta que no retornaba nada a pesar de que la app se estaba ejecutando.


devasc@labvm: ~/labs/devnet-src/sample-app
File Edit View Search Terminal Help

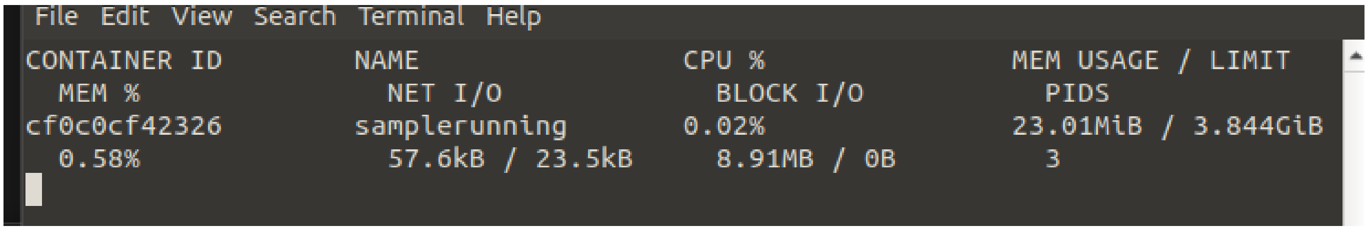
```
root@cf0c0cf42326:/home/myapp# cd templates/
root@cf0c0cf42326:/home/myapp/templates# cat index.html
<html>
<head>
    <title>Sample app</title>
    <link rel="stylesheet" href="/static/style.css" />
</head>
<body>
    <h1>You are calling me from {{request.remote_addr}}</h1>
</body>
</html>
root@cf0c0cf42326:/home/myapp/templates# cd ..
root@cf0c0cf42326:/home/myapp# ls
sample_app.py  static  templates
root@cf0c0cf42326:/home/myapp# cd static/
root@cf0c0cf42326:/home/myapp/static# ls
style.css
root@cf0c0cf42326:/home/myapp/static# cat style.css
body {background: lightsteelblue;}
root@cf0c0cf42326:/home/myapp/static# cd ..
root@cf0c0cf42326:/home/myapp# ls
sample_app.py  static  templates
root@cf0c0cf42326:/home/myapp# cat sample_app.py
from flask import Flask, request, render_template

sample = Flask(__name__)

@sample.route('/')
def home():
    return render_template('index.html')

if __name__ == '__main__':
    sample.run(host='0.0.0.0', port=8080)
root@cf0c0cf42326:/home/myapp# curl 0.0.0.0:8080
curl: (52) Empty reply from server
root@cf0c0cf42326:/home/myapp# 
```

al ver el `docker logs samplerunning` pude ver que había un problema al iniciar los hilos.



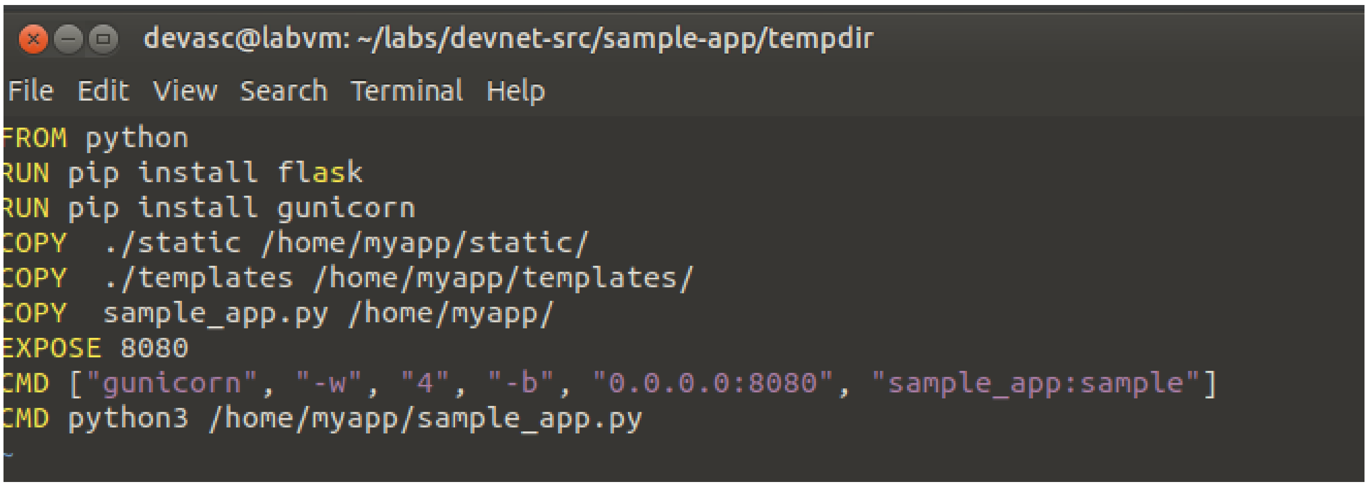
```
Exception occurred during processing of request from ('127.0.0.1', 47668)
Traceback (most recent call last):
  File "/usr/local/lib/python3.12/socketserver.py", line 318, in _handle_request
_noblock
    self.process_request(request, client_address)
  File "/usr/local/lib/python3.12/socketserver.py", line 706, in process_request
    t.start()
  File "/usr/local/lib/python3.12/threading.py", line 994, in start
    _start_new_thread(self._bootstrap, ())
RuntimeError: can't start new thread
----------------------------------------
----------------------------------------
Exception occurred during processing of request from ('127.0.0.1', 47670)
Traceback (most recent call last):
  File "/usr/local/lib/python3.12/socketserver.py", line 318, in _handle_request
_noblock
    self.process_request(request, client_address)
  File "/usr/local/lib/python3.12/socketserver.py", line 706, in process_request
    t.start()
  File "/usr/local/lib/python3.12/threading.py", line 994, in start
    _start_new_thread(self._bootstrap, ())
RuntimeError: can't start new thread
----------------------------------------
devasc@labvm:~$
```

al ver `docker logs samplerunning` pude ver que el problema no era de recursos de CPU, pues su uso es mínimo.



```
CONTAINER ID        NAME                CPU %               MEM USAGE / LIMIT
  MEM %               NET I/O             BLOCK I/O           PIDS
cf0c0cf42326        samplerunning       0.02%               23.01MiB / 3.844GiB
  0.58%               57.6kB / 23.5kB     8.91MB / 0B         3
```

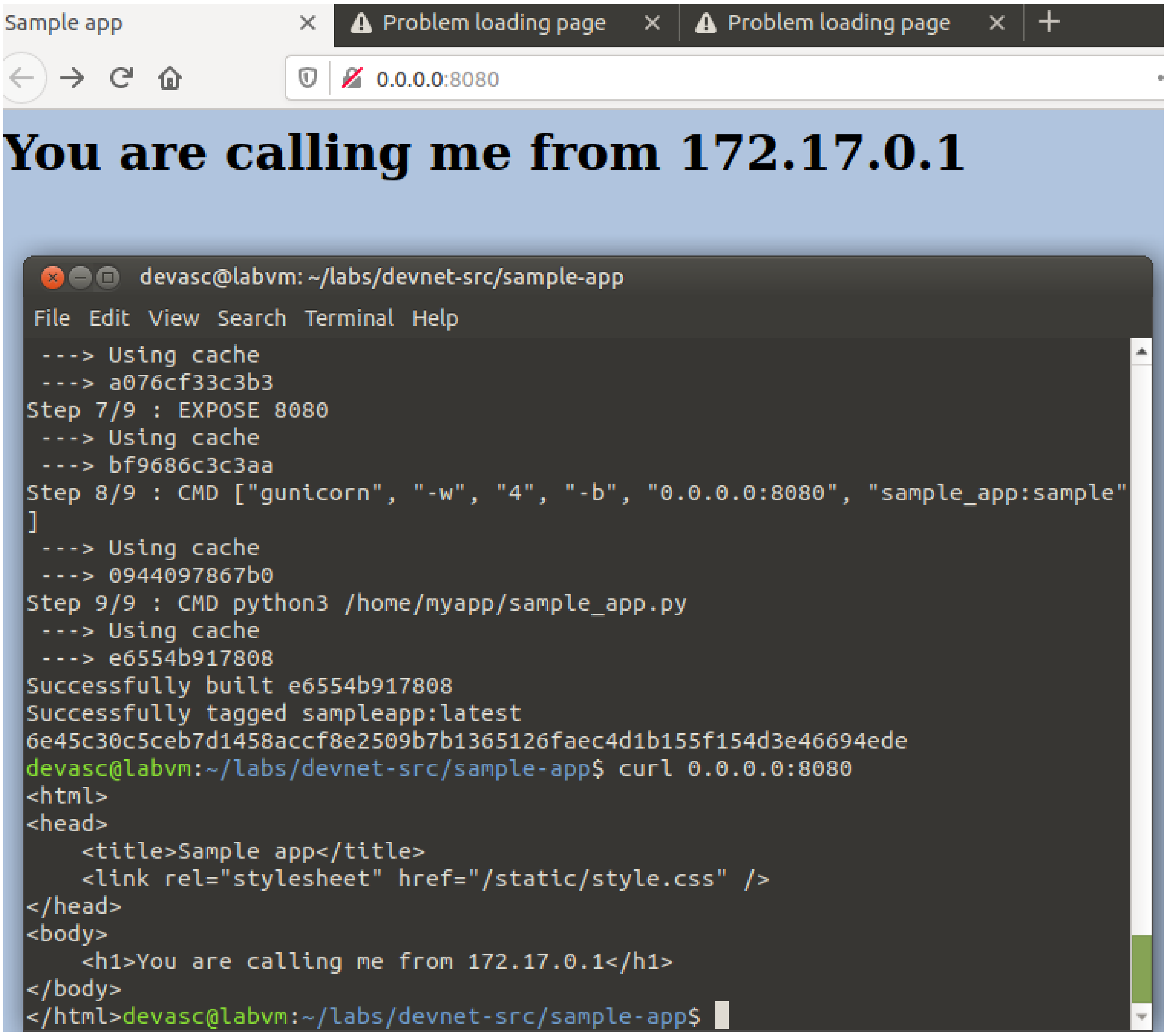
Investigando di con que Flask tenía problemas con el threading. Por ello usé gunicorn para ejecutar la app. Para ello edité el archivo Dockerfile para que instale gunicorn y agregué una línea para ejecutar la app con gunicorn.


devasc@labvm: ~/labs/devnet-src/sample-app/tempdir
File Edit View Search Terminal Help

```
FROM python
RUN pip install flask
RUN pip install gunicorn
COPY  ./static /home/myapp/static/
COPY  ./templates /home/myapp/templates/
COPY  sample_app.py /home/myapp/
EXPOSE 8080
CMD ["gunicorn", "-w", "4", "-b", "0.0.0.0:8080", "sample_app:sample"]
CMD python3 /home/myapp/sample_app.py
~
```

Luego eliminé todas las instancias de los contenedores con `docker ps -a` y `docker rm` y volví a ejecutar el script.

Ahora sí al fin pude acceder a la app.

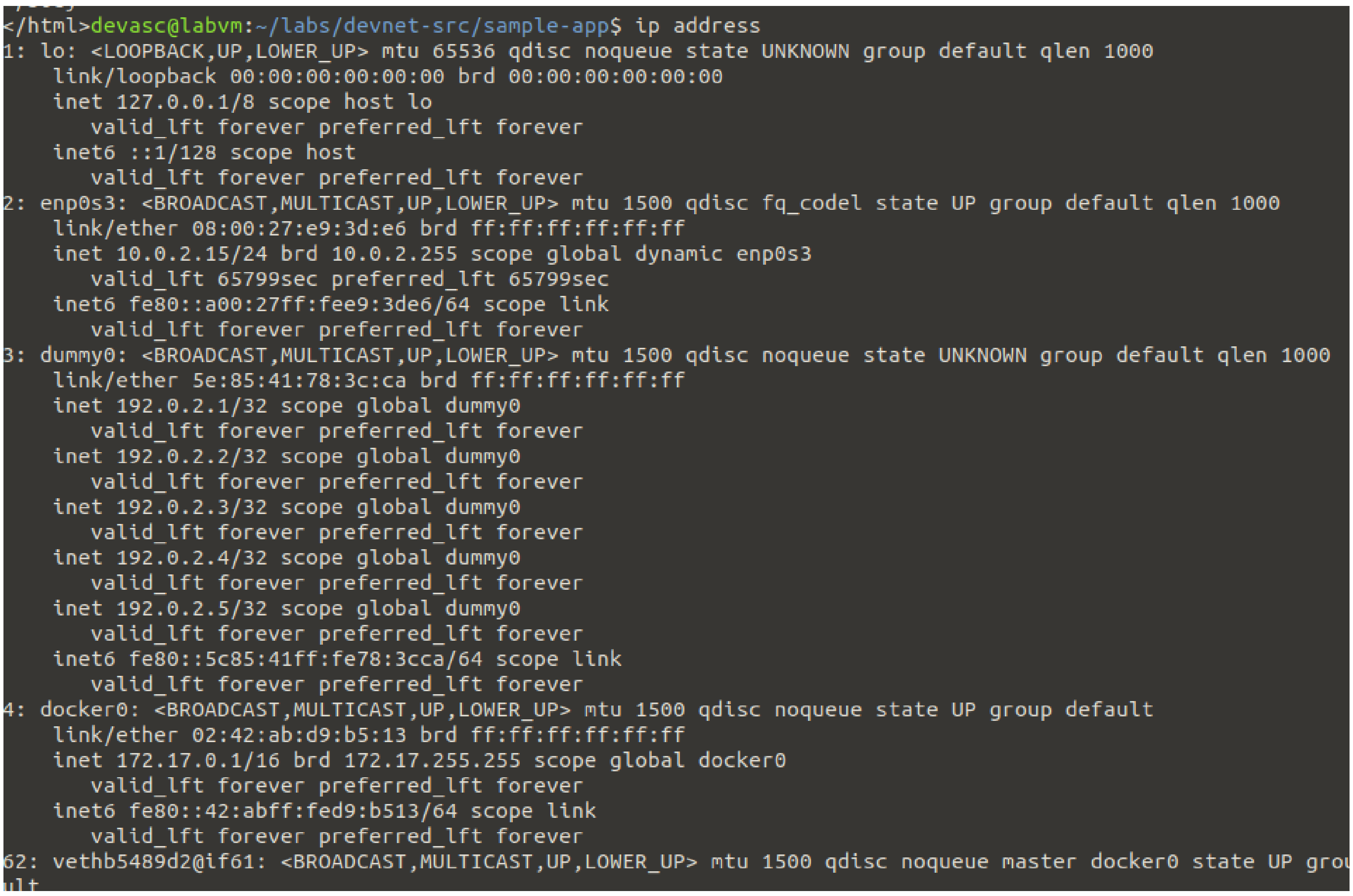

Sample app
Problem loading page
Problem loading page
0.0.0.0:8080
devasc@labvm: ~/labs/devnet-src/sample-app
File Edit View Search Terminal Help

You are calling me from 172.17.0.1

```
 ---> Using cache
 ---> a076cf33c3b3
Step 7/9 : EXPOSE 8080
 ---> Using cache
 ---> bf9686c3c3aa
Step 8/9 : CMD ["gunicorn", "-w", "4", "-b", "0.0.0.0:8080", "sample_app:sample"
]
 ---> Using cache
 ---> 0944097867b0
Step 9/9 : CMD python3 /home/myapp/sample_app.py
 ---> Using cache
 ---> e6554b917808
Successfully built e6554b917808
Successfully tagged sampleapp:latest
6e45c30c5ceb7d1458accf8e2509b7b1365126faec4d1b155f154d3e46694ede
devasc@labvm:~/labs/devnet-src/sample-app$ curl 0.0.0.0:8080
<html>
<head>
    <title>Sample app</title>
    <link rel="stylesheet" href="/static/style.css" />
</head>
<body>
    <h1>You are calling me from 172.17.0.1</h1>
</body>
</html>devasc@labvm:~/labs/devnet-src/sample-app$ 
```

Ahora sí seguimos con el laboratorio 6b.

Usando ip address:



```
</html>devasc@labvm:~/labs/devnet-src/sample-app$ ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:e9:3d:e6 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
       valid_lft 65799sec preferred_lft 65799sec
    inet6 fe80::a00:27ff:fee9:3de6/64 scope link
       valid_lft forever preferred_lft forever
3: dummy0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group default qlen 1000
    link/ether 5e:85:41:78:3c:ca brd ff:ff:ff:ff:ff:ff
    inet 192.0.2.1/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet 192.0.2.2/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet 192.0.2.3/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet 192.0.2.4/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet 192.0.2.5/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet6 fe80::5c85:41ff:fe78:3cca/64 scope link
       valid_lft forever preferred_lft forever
4: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:ab:d9:b5:13 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
    inet6 fe80::42:abff:fed9:b513/64 scope link
       valid_lft forever preferred_lft forever
62: vethb5489d2@if61: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state UP grou
ult
```

Efectivamente, la IP es 172.17.0.0/16 para la red de los contenedores de docker.

Luego de hacer esto me di cuenta que hice todo lo que pedía el resto del laboratorio 6b; lo cual fue necesario para debugear el problema de la app. Aprendí mucho sobre como se comportan las redes de docker y como se puede acceder a ellas.