# Laboratorio 6

## Network Troubleshooting Tools

---

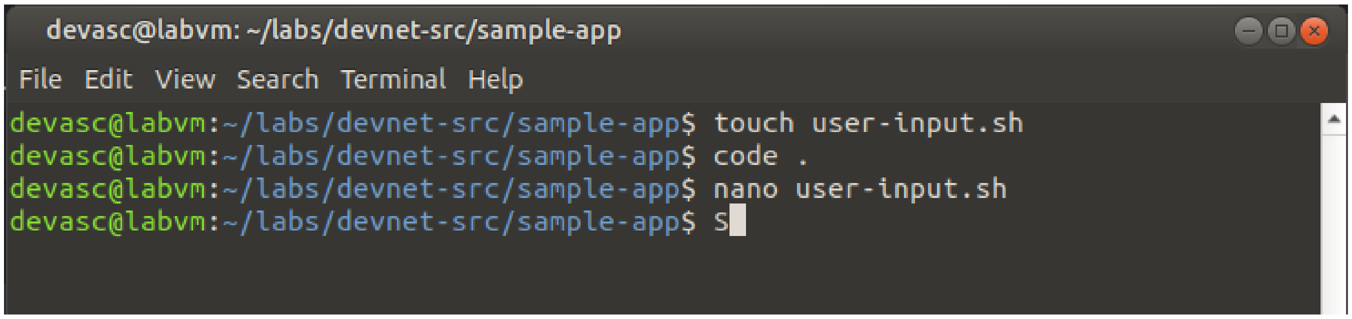
Sergio Sebastian Pezo Jimenez - 20224087G

---

## Parte 1: Inicializamos la VM de DEVASC.

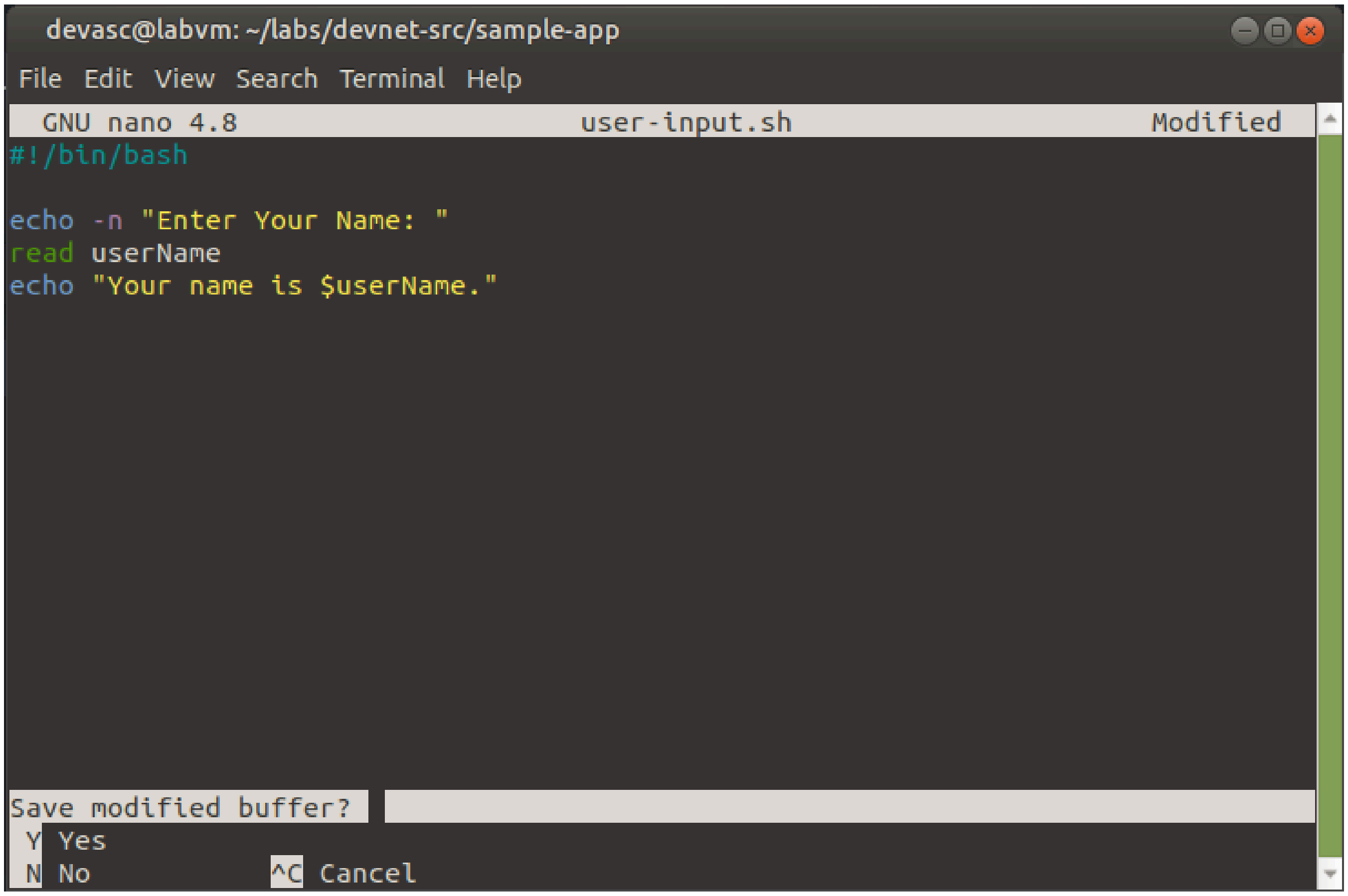
## Parte 2: Creamos un simple script de bash

Creamos un archivo en de bash y lo abrimos mediante CLI con nano


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

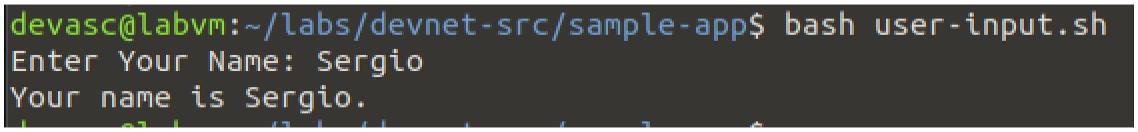
```
devasc@labvm:~/labs/devnet-src/sample-app$ touch user-input.sh
devasc@labvm:~/labs/devnet-src/sample-app$ code .
devasc@labvm:~/labs/devnet-src/sample-app$ nano user-input.sh
devasc@labvm:~/labs/devnet-src/sample-app$ S
```

Agregamos un poco de código y lo guardamos con `Ctrl + X` e `Y`.


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

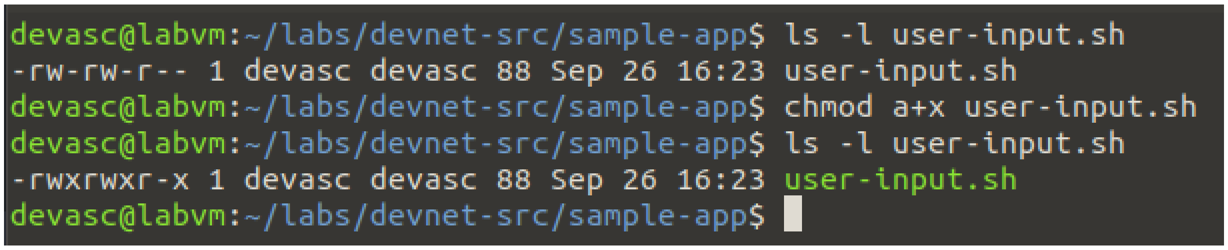
```
  GNU nano 4.8                    user-input.sh                    Modified
#!/bin/bash

echo -n "Enter Your Name: "
read userName
echo "Your name is $userName."












Save modified buffer?
 Y Yes
 N No              ^C Cancel
```

Ejecutamos el archivo:



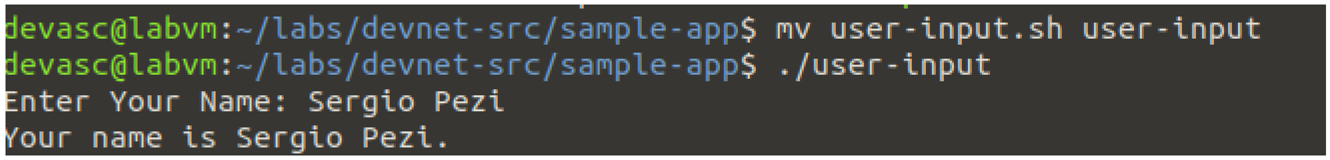
```
devasc@labvm:~/labs/devnet-src/sample-app$ bash user-input.sh
Enter Your Name: Sergio
Your name is Sergio.
```

Cambiamos el modo del script a una achivo ejecutable para todos los usuarios.



```
devasc@labvm:~/labs/devnet-src/sample-app$ ls -l user-input.sh
-rw-rw-r-- 1 devasc devasc 88 Sep 26 16:23 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 88 Sep 26 16:23 user-input.sh
devasc@labvm:~/labs/devnet-src/sample-app$ 
```

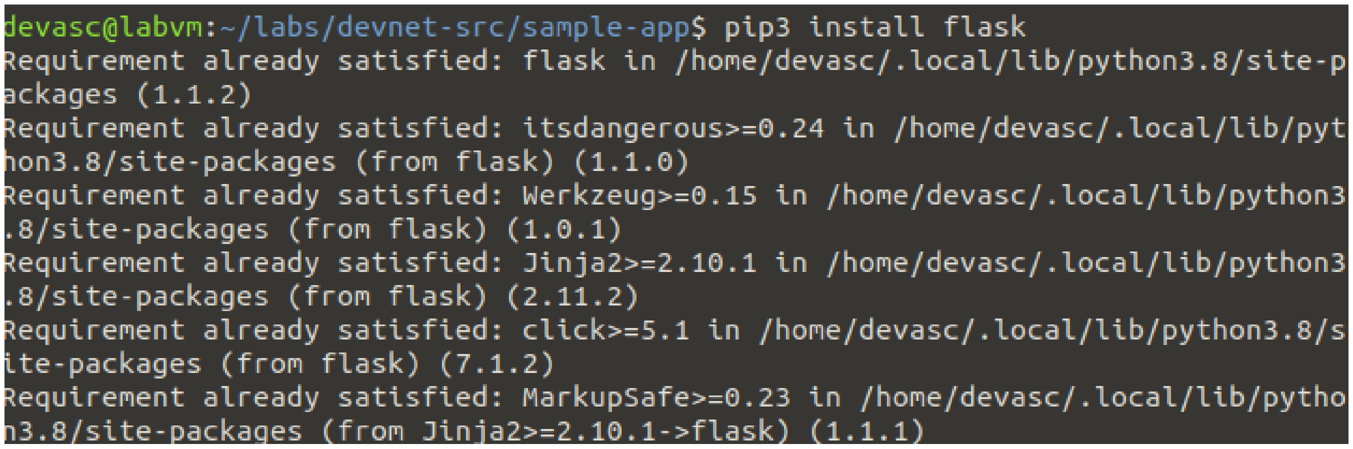
Renombramos el archivo para quitar la extensión de .sh, para ejecutar sin el source:



```
devasc@labvm:~/labs/devnet-src/sample-app$ mv user-input.sh user-input
devasc@labvm:~/labs/devnet-src/sample-app$ ./user-input
Enter Your Name: Sergio Pezi
Your name is Sergio Pezi.
```

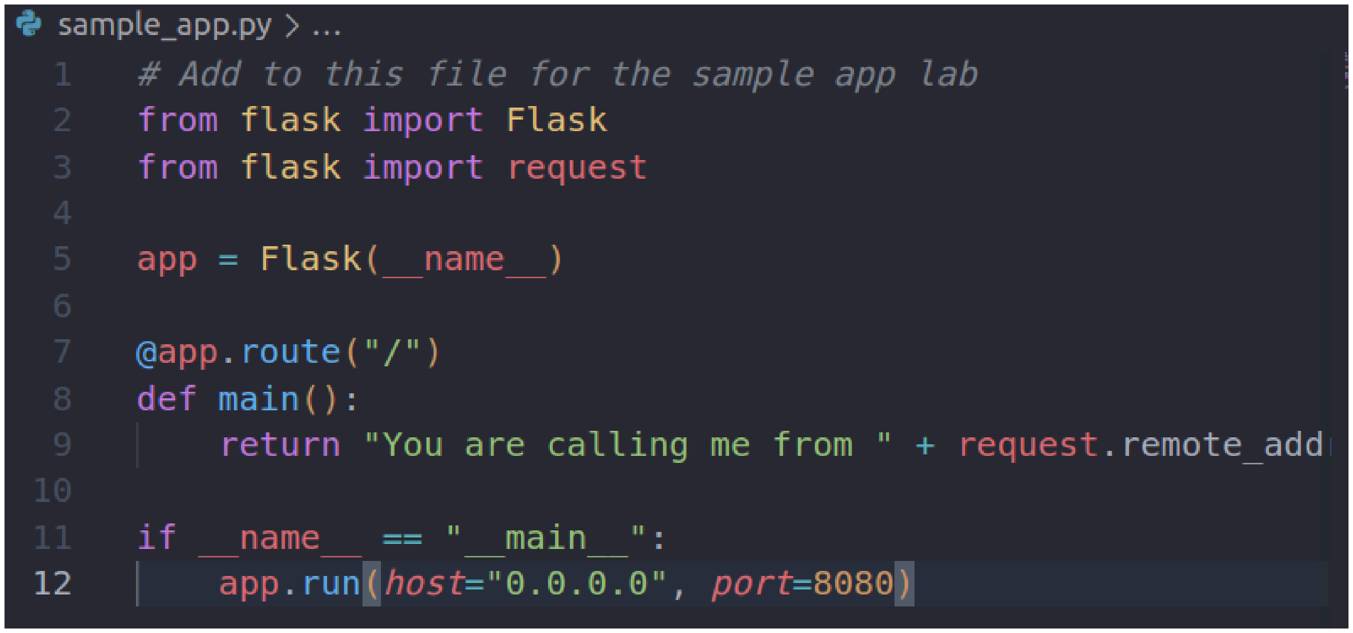
## Parte 3: Creamos una web de ejemplo

Primero instalaremos Flask



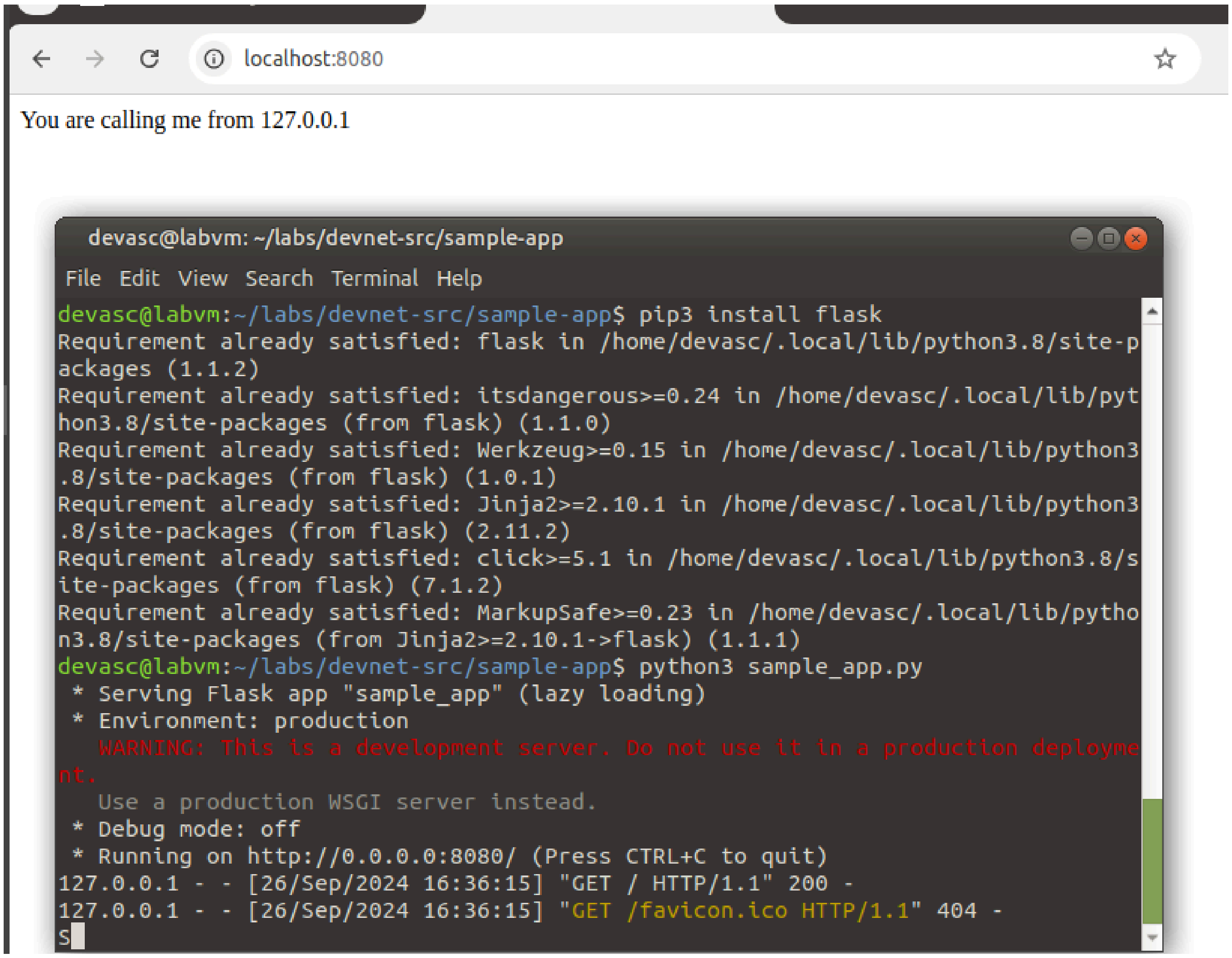
```
devasc@labvm:~/labs/devnet-src/sample-app$ pip3 install flask
Requirement already satisfied: flask in /home/devasc/.local/lib/python3.8/site-p
ackages (1.1.2)
Requirement already satisfied: itsdangerous>=0.24 in /home/devasc/.local/lib/pyt
hon3.8/site-packages (from flask) (1.1.0)
Requirement already satisfied: Werkzeug>=0.15 in /home/devasc/.local/lib/python3
.8/site-packages (from flask) (1.0.1)
Requirement already satisfied: Jinja2>=2.10.1 in /home/devasc/.local/lib/python3
.8/site-packages (from flask) (2.11.2)
Requirement already satisfied: click>=5.1 in /home/devasc/.local/lib/python3.8/s
ite-packages (from flask) (7.1.2)
Requirement already satisfied: MarkupSafe>=0.23 in /home/devasc/.local/lib/pytho
n3.8/site-packages (from Jinja2>=2.10.1->flask) (1.1.1)
```

Creamos una web simple



```python
# Add to this file for the sample app lab
from flask import Flask
from flask import request

app = Flask(__name__)

@app.route("/")
def main():
    return "You are calling me from " + request.remote_add

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

Procedemos a correr la web, y notamos que funciona a la perfección



localhost:8080

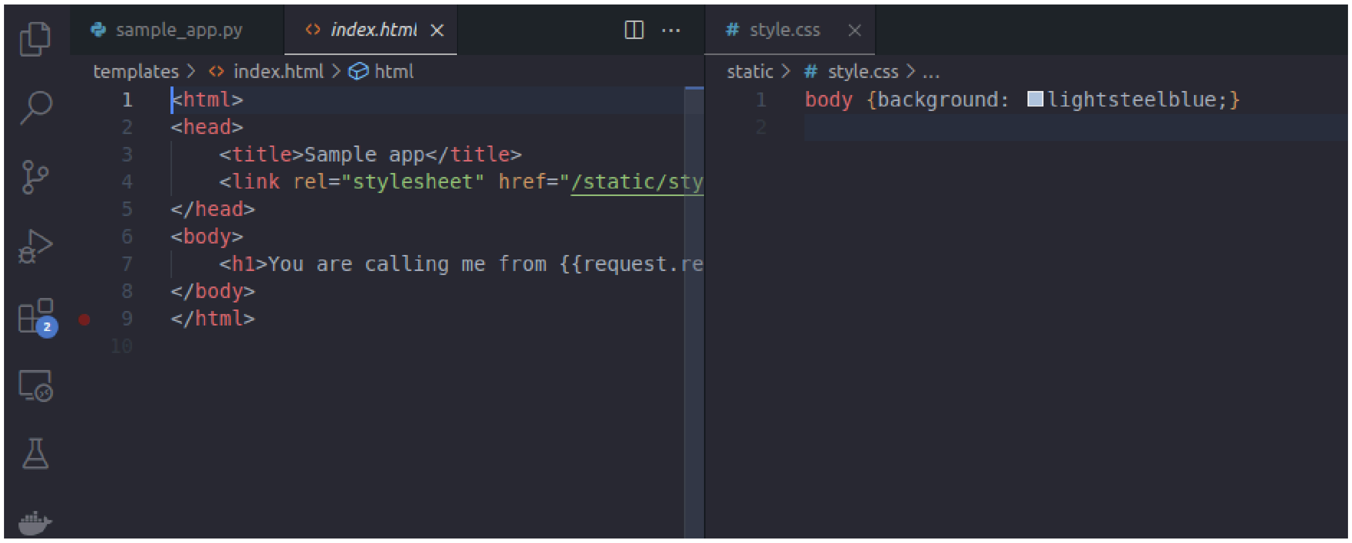
You are calling me from 127.0.0.1

devasc@labvm: ~/labs/devnet-src/sample-app

```
devasc@labvm:~/labs/devnet-src/sample-app$ pip3 install flask
Requirement already satisfied: flask in /home/devasc/.local/lib/python3.8/site-p
ackages (1.1.2)
Requirement already satisfied: itsdangerous>=0.24 in /home/devasc/.local/lib/pyt
hon3.8/site-packages (from flask) (1.1.0)
Requirement already satisfied: Werkzeug>=0.15 in /home/devasc/.local/lib/python3
.8/site-packages (from flask) (1.0.1)
Requirement already satisfied: Jinja2>=2.10.1 in /home/devasc/.local/lib/python3
.8/site-packages (from flask) (2.11.2)
Requirement already satisfied: click>=5.1 in /home/devasc/.local/lib/python3.8/s
ite-packages (from flask) (7.1.2)
Requirement already satisfied: MarkupSafe>=0.23 in /home/devasc/.local/lib/pytho
n3.8/site-packages (from Jinja2>=2.10.1->flask) (1.1.1)
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 16:36:15] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [26/Sep/2024 16:36:15] "GET /favicon.ico HTTP/1.1" 404 -
S
```

## Parte 4: Configuramos la aplicación para usar archivo website

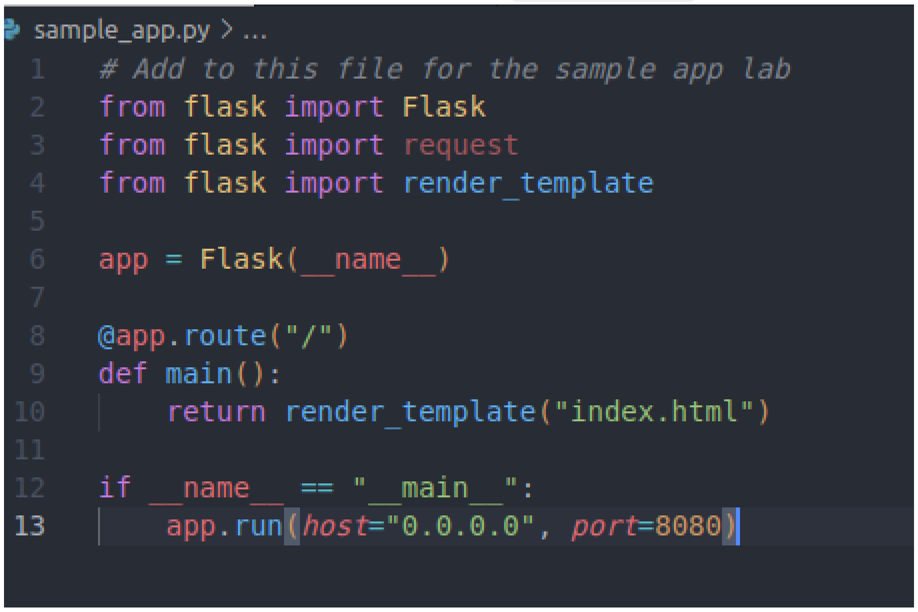
Primero exploramos los arhcivos a utilizar



```
<html>
<head>
    <title>Sample app</title>
    <link rel="stylesheet" href="/static/sty
</head>
<body>
    <h1>You are calling me from {{request.re
</body>
</html>
```

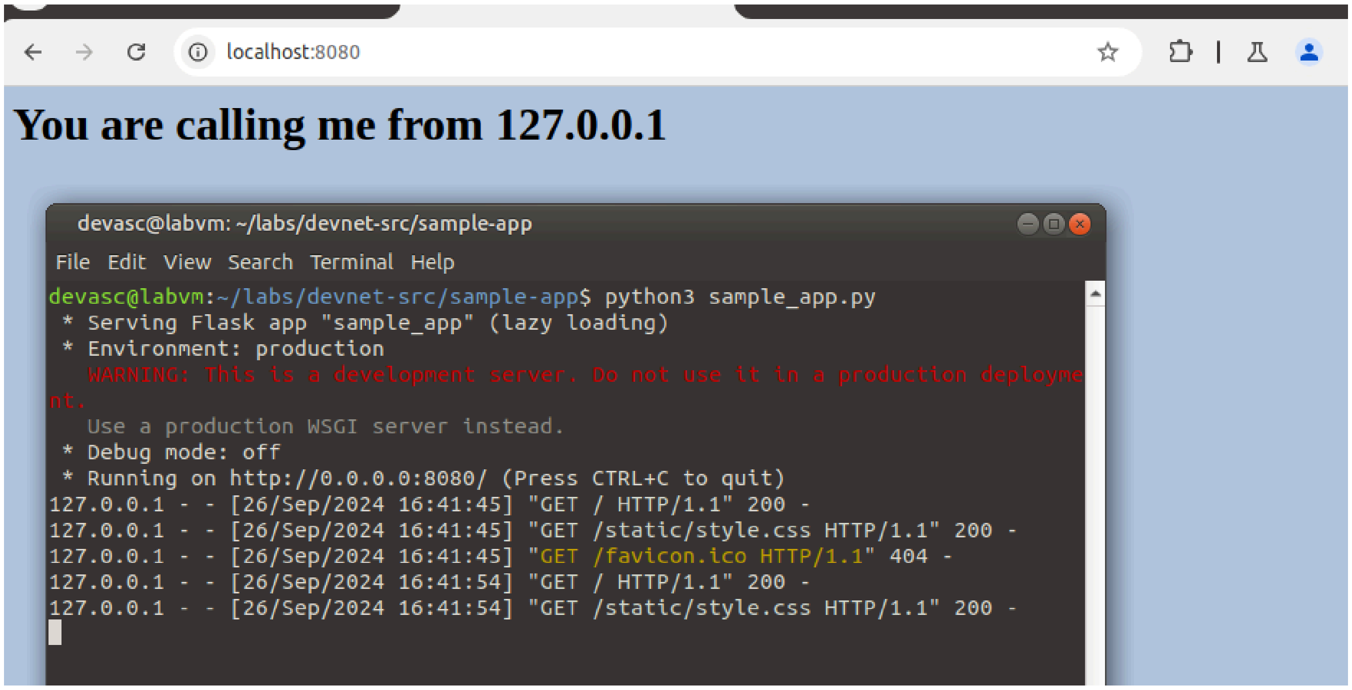
```
body {background: lightsteelblue;}
```

Ahora devolveremos el archivo `index.html`



```python
# Add to this file for the sample app lab
from flask import Flask
from flask import request
from flask import render_template

app = Flask(__name__)

@app.route("/")
def main():
    return render_template("index.html")

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

Y nuestra aplicación se verá así



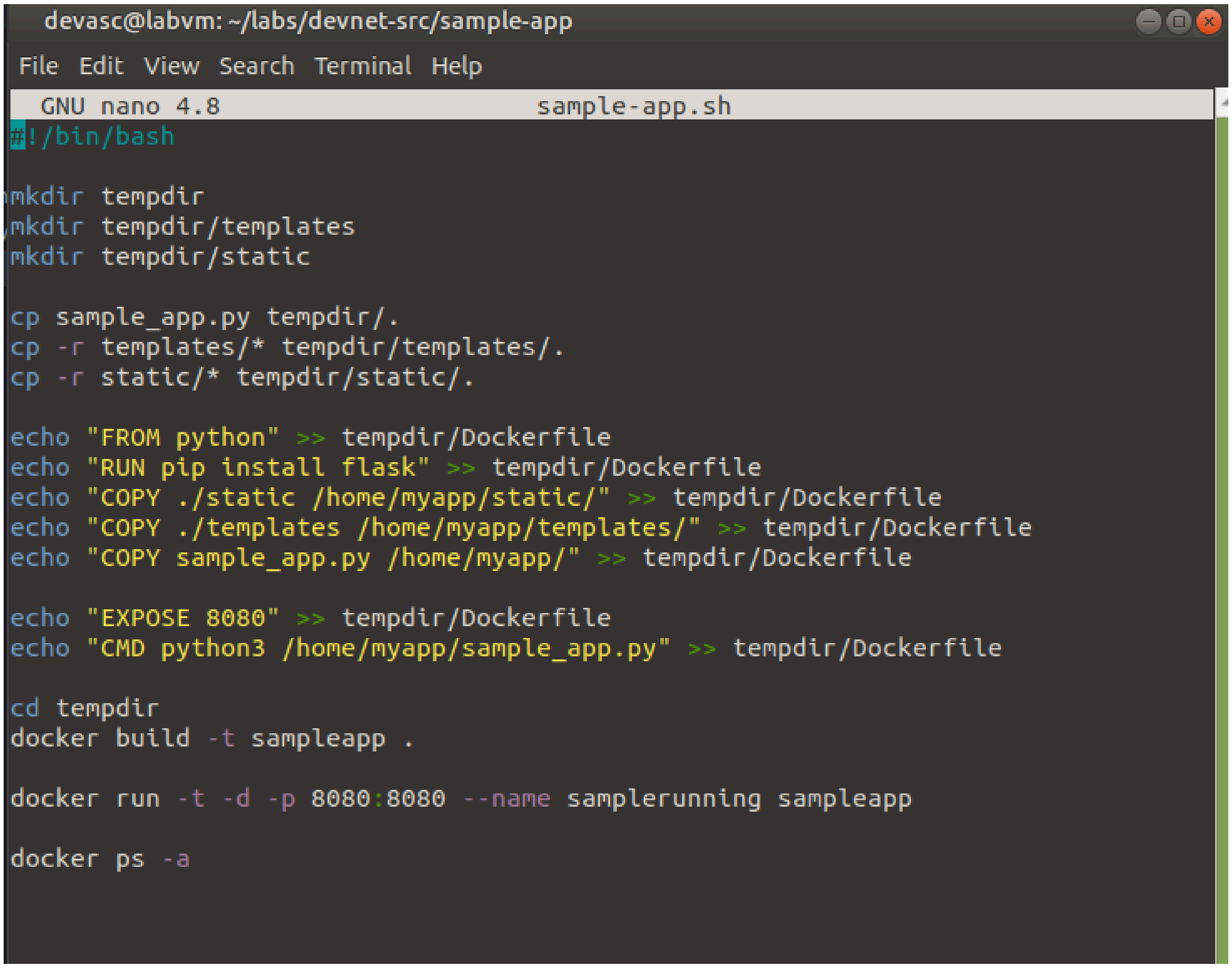
localhost:8080
devasc@labvm: ~/labs/devnet-src/sample-app
File Edit View Search Terminal Help

You are calling me from 127.0.0.1

```
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 16:41:45] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [26/Sep/2024 16:41:45] "GET /static/style.css HTTP/1.1" 200 -
127.0.0.1 - - [26/Sep/2024 16:41:45] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [26/Sep/2024 16:41:54] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [26/Sep/2024 16:41:54] "GET /static/style.css HTTP/1.1" 200 -
```

## Parte 5: Creamos un script de Bash para hacer Build y correr un contenedor de Docker.

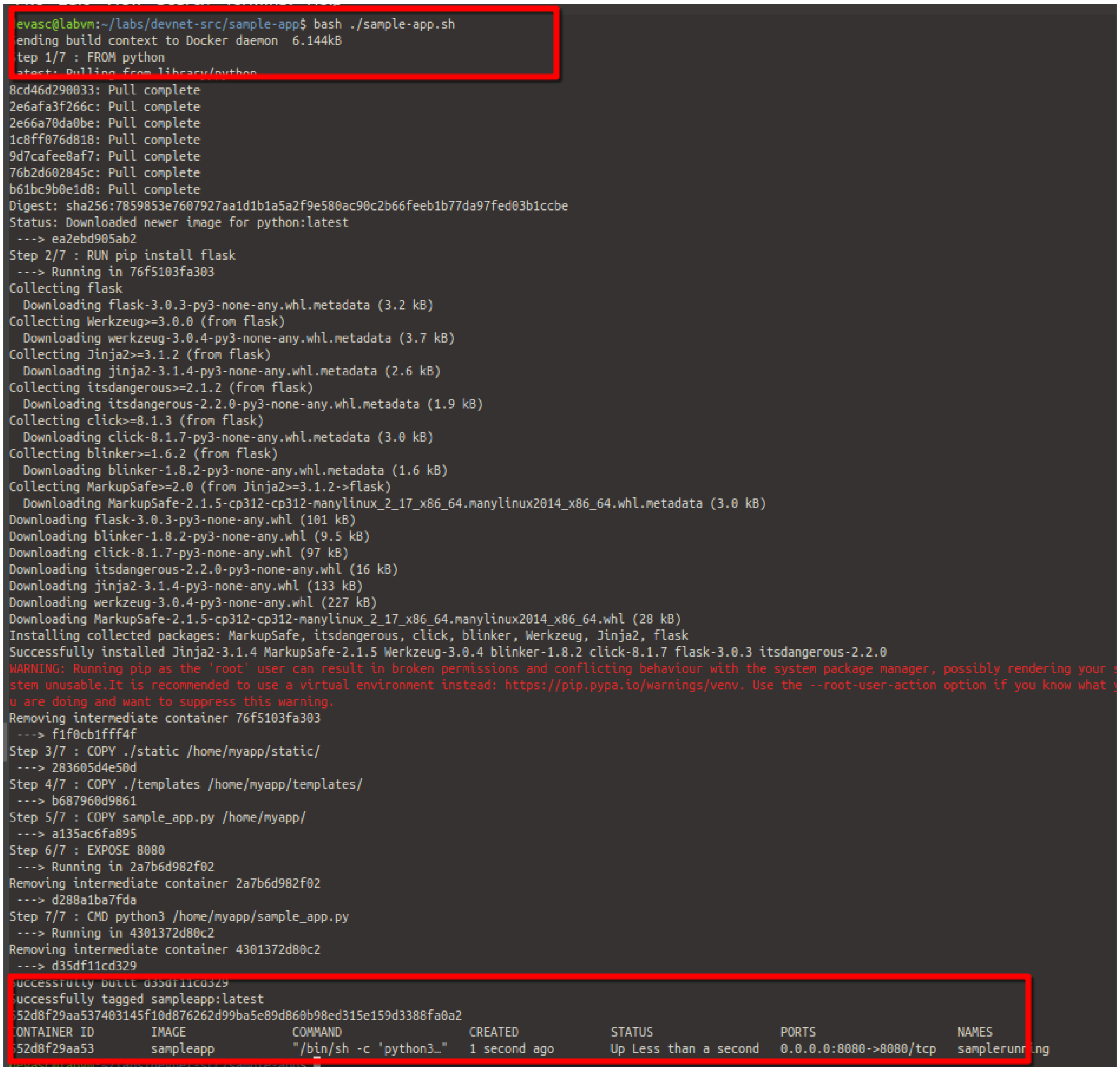
Básicamente, creamos carpetas temporales y dentro un Dockerfile para crear un imagen y nuestro contendedor que serán llamados `sampleapp` y `samplerunning` respectivamente, cuyo puerto en el host y en el contenedor será el 8080, finalmente desplegaremos la vista para ver todos los conetenedores.


devasc@labvm: ~/labs/devnet-src/sample-app
File Edit View Search Terminal Help
GNU nano 4.8 sample-app.sh

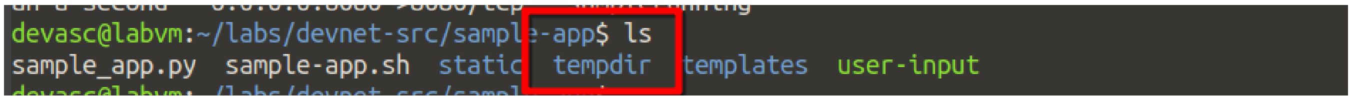
```
#!/bin/bash

mkdir tempdir
mkdir tempdir/templates
mkdir tempdir/static

cp sample_app.py tempdir/.
cp -r templates/* tempdir/templates/.
cp -r static/* tempdir/static/.

echo "FROM python" >> tempdir/Dockerfile
echo "RUN pip install flask" >> tempdir/Dockerfile
echo "COPY ./static /home/myapp/static/" >> tempdir/Dockerfile
echo "COPY ./templates /home/myapp/templates/" >> tempdir/Dockerfile
echo "COPY sample_app.py /home/myapp/" >> tempdir/Dockerfile

echo "EXPOSE 8080" >> tempdir/Dockerfile
echo "CMD python3 /home/myapp/sample_app.py" >> tempdir/Dockerfile

cd tempdir
docker build -t sampleapp .

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

docker ps -a
```

## Parte 6: Buildeamos, corremos y verificamos nuestro contenedor de Docker.

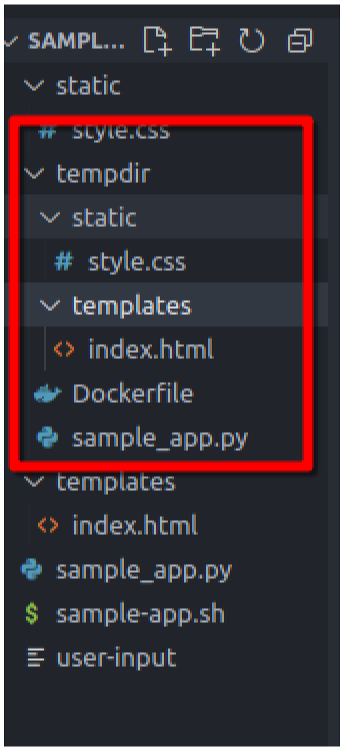
Efectivamente funciona como lo mencionado en la parte 5, hemos construido nuestra imagen



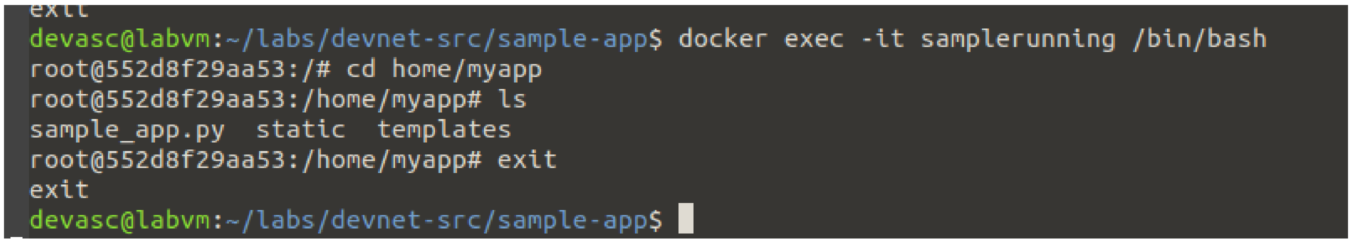
```
evasc@labvm:~/labs/devnet-src/sample-app$ bash ./sample-app.sh
ending build context to Docker daemon  6.144kB
tep 1/7 : FROM python
atest: Pulling from library/python
8cd46d290033: Pull complete
2e6afa3f266c: Pull complete
2e66a70da0be: Pull complete
1c8ff076d818: Pull complete
9d7cafee8af7: Pull complete
76b2d602845c: Pull complete
b61bc9b0e1d8: Pull complete
Digest: sha256:7859853e7607927aa1d1b1a5a2f9e580ac90c2b66feeb1b77da97fed03b1ccbe
Status: Downloaded newer image for python:latest
 ---> ea2ebd905ab2
Step 2/7 : RUN pip install flask
 ---> Running in 76f5103fa303
Collecting flask
  Downloading flask-3.0.3-py3-none-any.whl.metadata (3.2 kB)
Collecting Werkzeug>=3.0.0 (from flask)
  Downloading werkzeug-3.0.4-py3-none-any.whl.metadata (3.7 kB)
Collecting Jinja2>=3.1.2 (from flask)
  Downloading jinja2-3.1.4-py3-none-any.whl.metadata (2.6 kB)
Collecting itsdangerous>=2.1.2 (from flask)
  Downloading itsdangerous-2.2.0-py3-none-any.whl.metadata (1.9 kB)
Collecting click>=8.1.3 (from flask)
  Downloading click-8.1.7-py3-none-any.whl.metadata (3.0 kB)
Collecting blinker>=1.6.2 (from flask)
  Downloading blinker-1.8.2-py3-none-any.whl.metadata (1.6 kB)
Collecting MarkupSafe>=2.0 (from Jinja2>=3.1.2->flask)
  Downloading MarkupSafe-2.1.5-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (3.0 kB)
Downloading flask-3.0.3-py3-none-any.whl (101 kB)
Downloading blinker-1.8.2-py3-none-any.whl (9.5 kB)
Downloading click-8.1.7-py3-none-any.whl (97 kB)
Downloading itsdangerous-2.2.0-py3-none-any.whl (16 kB)
Downloading jinja2-3.1.4-py3-none-any.whl (133 kB)
Downloading werkzeug-3.0.4-py3-none-any.whl (227 kB)
Downloading MarkupSafe-2.1.5-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (28 kB)
Installing collected packages: MarkupSafe, itsdangerous, click, blinker, Werkzeug, Jinja2, flask
Successfully installed Jinja2-3.1.4 MarkupSafe-2.1.5 Werkzeug-3.0.4 blinker-1.8.2 click-8.1.7 flask-3.0.3 itsdangerous-2.2.0
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager, possibly rendering your s
stem unusable.It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv. Use the --root-user-action option if you know what y
u are doing and want to suppress this warning.
Removing intermediate container 76f5103fa303
 ---> f1f0cb1fff4f
Step 3/7 : COPY ./static /home/myapp/static/
 ---> 283605d4e50d
Step 4/7 : COPY ./templates /home/myapp/templates/
 ---> b687960d9861
Step 5/7 : COPY sample_app.py /home/myapp/
 ---> a135ac6fa895
Step 6/7 : EXPOSE 8080
 ---> Running in 2a7b6d982f02
Removing intermediate container 2a7b6d982f02
 ---> d288a1ba7fda
Step 7/7 : CMD python3 /home/myapp/sample_app.py
 ---> Running in 4301372d80c2
Removing intermediate container 4301372d80c2
 ---> d35df11cd329
uccessfully built d35df11cd329
uccessfully tagged sampleapp:latest
52d8f29aa537403145f10d876262d99ba5e89d860b98ed315e159d3388fa0a2
ONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS                  PORTS                    NAMES
52d8f29aa53        sampleapp           "/bin/sh -c 'python3…"   1 second ago        Up Less than a second   0.0.0.0:8080->8080/tcp   samplerunning
```



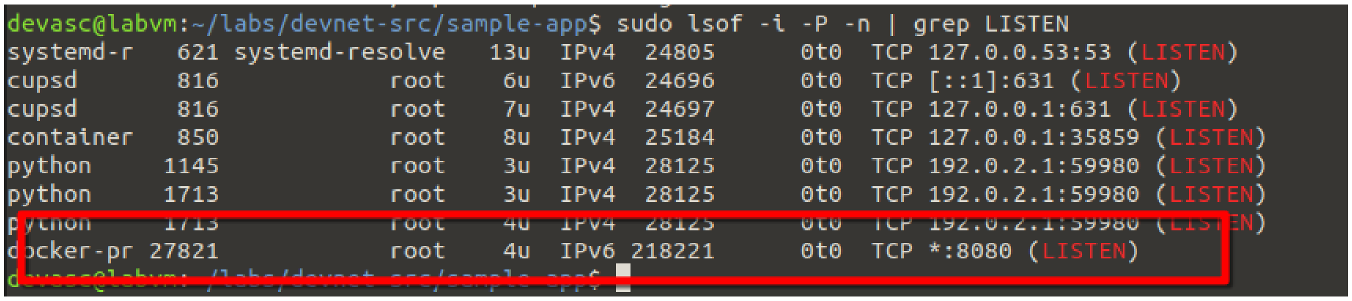
```
devasc@labvm:~/labs/devnet-src/sample-app$ ls
sample_app.py  sample-app.sh  static  tempdir  templates  user-input
```


```
SAMPL...
  static
    style.css
  tempdir
    static
      style.css
    templates
      index.html
    Dockerfile
    sample_app.py
  templates
    index.html
  sample_app.py
  sample-app.sh
  user-input
```

Revisemos nuestro el interior de nuestro contenedor de la siguienta manera:

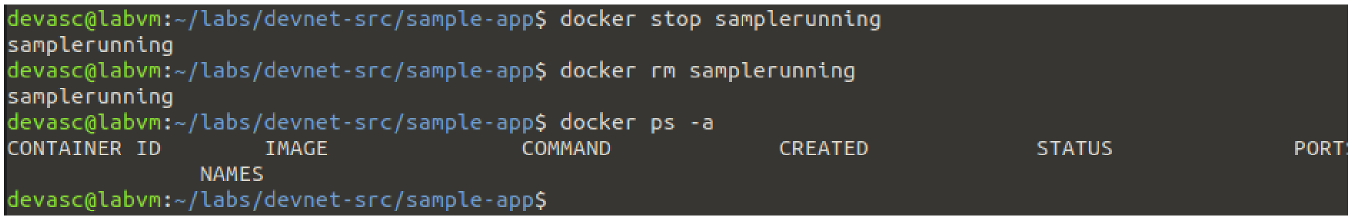

```
exit
devasc@labvm:~/labs/devnet-src/sample-app$ docker exec -it samplerunning /bin/bash
root@552d8f29aa53:/# cd home/myapp
root@552d8f29aa53:/home/myapp# ls
sample_app.py  static  templates
root@552d8f29aa53:/home/myapp# exit
exit
devasc@labvm:~/labs/devnet-src/sample-app$ 
```

Nuestro puerto está siendo usando, por lo que todo perfecto


```
devasc@labvm:~/labs/devnet-src/sample-app$ sudo lsof -i -P -n | grep LISTEN
systemd-r    621 systemd-resolve   13u  IPv4  24805      0t0  TCP 127.0.0.53:53 (LISTEN)
cupsd        816            root    6u  IPv6  24696      0t0  TCP [::1]:631 (LISTEN)
cupsd        816            root    7u  IPv4  24697      0t0  TCP 127.0.0.1:631 (LISTEN)
container    850            root    8u  IPv4  25184      0t0  TCP 127.0.0.1:35859 (LISTEN)
python      1145            root    3u  IPv4  28125      0t0  TCP 192.0.2.1:59980 (LISTEN)
python      1713            root    3u  IPv4  28125      0t0  TCP 192.0.2.1:59980 (LISTEN)
python      1713            root    4u  IPv4  28125      0t0  TCP 192.0.2.1:59980 (LISTEN)
docker-pr  27821            root    4u  IPv6 218221      0t0  TCP *:8080 (LISTEN)
devasc@labvm:~/labs/devnet-src/sample-app$ 
```

## Parte 7: Paramos y borramos nuestro contenedor

Finalmente, paramos el servicio y eliminamos el contenedor:


```
devasc@labvm:~/labs/devnet-src/sample-app$ docker stop samplerunning
samplerunning
devasc@labvm:~/labs/devnet-src/sample-app$ docker rm samplerunning
samplerunning
devasc@labvm:~/labs/devnet-src/sample-app$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORT
              NAMES
devasc@labvm:~/labs/devnet-src/sample-app$
```

Finalizado.