

## NGINX & TRAEFIK

MACHINE 1: Over a container , setup a server serving an api /howareyou

STEPS:

1. Install Docker and python3 on VM1

```
1 sudo apt update
2 sudo apt upgrade
3 sudo apt install docker.io
4 sudo snap install docker
5 docker --version
6 sudo apt install software-properties-common
7 sudo apt install python3
8 python --version
9 python3 --version
```

2. Create a directory and create a python file in it.

```
10 mkdir flaskapp
11 cd flaskapp
12 touch app.py
13 sudo nano app.py
14 touch requirements.txt
15 sudo nano requirements.txt
16 sudo nano app.py
17 touch Dockerfile
18 sudo nano Dockerfile
```

3. Write python code with the help of flask serving an api /howareyou

```
GNU nano 4.8 app.py Modified
from flask import Flask
app = Flask(__name__)

@app.route('/')
def welcome():
    return 'Flask app is running on docker\n'

@app.route('/howareyou')
def howareyou():
    return 'Hi! How are you?\n'

if __name__ == '__main__':
    app.run(host = "0.0.0.0" , debug = True)
```

4. Add requirements file.

```
vm1@vm1-VirtualBox: ~  
vm1@vm1-VirtualBox: ~/flaskapp  
GNU nano 4.8 requirements.txt  
Flask==2.0.3
```

5. Create Dockerfile.

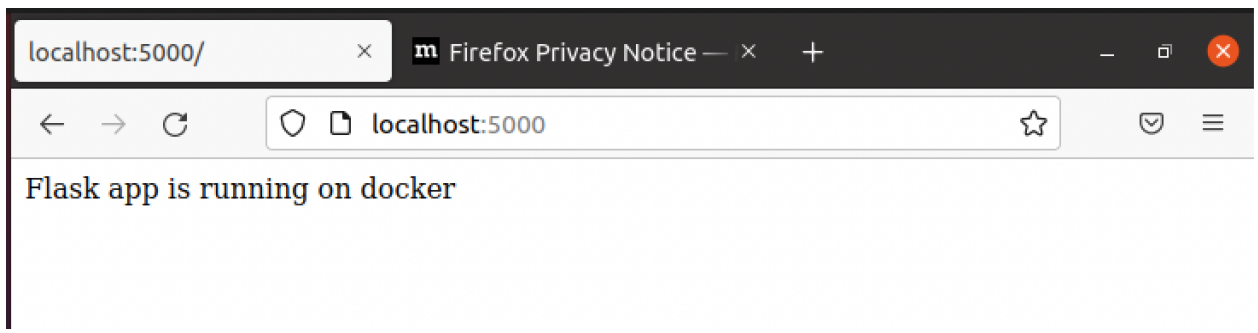
```
vm1@vm1-VirtualBox: ~  
vm1@vm1-VirtualBox: ~/flaskapp  
GNU nano 4.8 Dockerfile  
FROM python:alpine3.7  
COPY . /app  
WORKDIR /app  
RUN pip install -r requirements.txt  
EXPOSE 500  
ENTRYPOINT [ "python" ]  
CMD [ "app.py" ]
```

6. Build your dockerfile in the path where all your files are present.

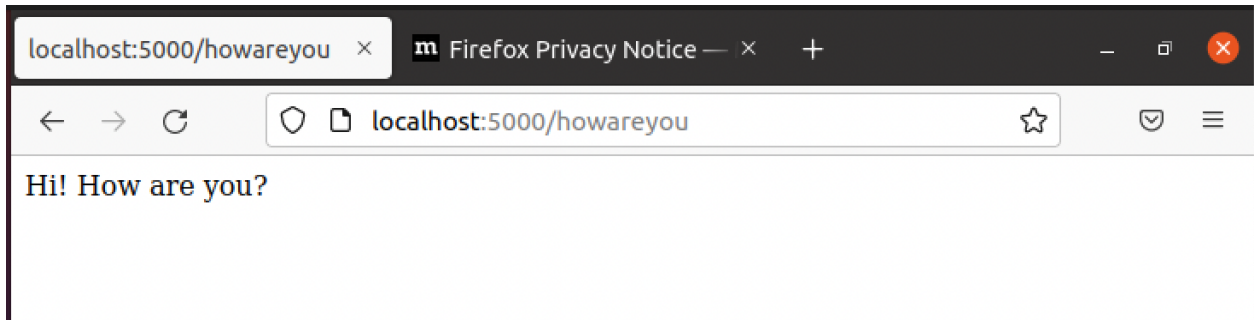
```
vm1@vm1-VirtualBox:~/flaskapp$ sudo docker build -t flask .  
Sending build context to Docker daemon 4.096kB  
Step 1/7 : FROM python:alpine3.7  
--> 00be2573e9f7  
Step 2/7 : COPY . /app  
--> 88a59c4738b7  
Step 3/7 : WORKDIR /app  
--> Running in 10d3b5454207  
Removing intermediate container 10d3b5454207  
--> b3646ccbaa94  
Step 4/7 : RUN pip install -r requirements.txt  
--> Running in 42efd931a560  
Collecting Flask==2.0.3 (from -r requirements.txt (line 1))  
  Downloading https://files.pythonhosted.org/packages/cd/77/59df23681f4fd19b7cb  
bb5e92484d46ad587554f5d490f33ef907e456132/Flask-2.0.3-py3-none-any.whl (95kB)  
Collecting Jinja2>=3.0 (from Flask==2.0.3->-r requirements.txt (line 1))  
  Downloading https://files.pythonhosted.org/packages/20/9a/e5d9ec41927401e41ae  
a8af6d16e78b5e612bca4699d417f646a9610a076/Jinja2-3.0.3-py3-none-any.whl (133kB)  
Collecting itsdangerous>=2.0 (from Flask==2.0.3->-r requirements.txt (line 1))  
  Downloading https://files.pythonhosted.org/packages/76/9b/88ac47681ba6af8ee99  
4c9e83ecdffc0048df59f8f6df5c2f766998fe87e7/itsdangerous-2.1.1-py3-none-any.whl  
Collecting click>=7.1.2 (from Flask==2.0.3->-r requirements.txt (line 1))  
  Downloading https://files.pythonhosted.org/packages/4a/a8/0b2ced25639fb20cc1c  
9784de90a8c25f9504a7f18cd8b5397bd61696d7d/click-8.0.4-py3-none-any.whl (97kB)  
Collecting Werkzeug>=2.0 (from Flask==2.0.3->-r requirements.txt (line 1))  
  Downloading https://files.pythonhosted.org/packages/f4/f3/22afbdb20cc4654b10c  
98043414a14057cd27fdb9d4ae61cea596000ba2/Werkzeug-2.0.3-py3-none-any.whl (289k  
B)
```

7. Use the docker run command to run your docker image and start the flask on your localhost.

```
vm1@vm1-VirtualBox:~/flaskapp$ sudo docker run -p 5000:5000 --network=host flask
k
WARNING: Published ports are discarded when using host network mode
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Running on all addresses.
  WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://10.0.2.15:5000/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 434-736-125
```



The screenshot shows a web browser window with the address bar set to 'localhost:5000/'. The page content displays the text 'Flask app is running on docker'.



The screenshot shows a web browser window with the address bar set to 'localhost:5000/howareyou'. The page content displays the text 'Hi! How are you?'.

MACHINE 2: Setup a server using nginx that serves as front (reverse proxy) to container running as front (reverse proxy) to continue running in machine 1 proxy server should return request for /howareyou when required.

STEPS:



1. Install nginx on VM2

```
vm2@vm2-VirtualBox:~$ sudo apt install nginx
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libfwupdplugin1
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  libnginx-mod-http-image-filter libnginx-mod-http-xslt-filter
  libnginx-mod-mail libnginx-mod-stream nginx-common nginx-core
Suggested packages:
  fcgiwrap nginx-doc
The following NEW packages will be installed:
  libnginx-mod-http-image-filter libnginx-mod-http-xslt-filter
  libnginx-mod-mail libnginx-mod-stream nginx nginx-common nginx-core
0 upgraded, 7 newly installed, 0 to remove and 0 not upgraded.
Need to get 603 kB of archives.
After this operation, 2,134 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 nginx-common
  all 1.18.0-0ubuntu1.2 [37.5 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 libnginx-mod
  -http-image-filter amd64 1.18.0-0ubuntu1.2 [14.4 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 libnginx-mod
  -http-xslt-filter amd64 1.18.0-0ubuntu1.2 [12.7 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 libnginx-mod
  -mail amd64 1.18.0-0ubuntu1.2 [42.5 kB]
Get:5 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 libnginx-mod
  -stream amd64 1.18.0-0ubuntu1.2 [67.3 kB]
^C
vm2@vm2-VirtualBox:~$ sudo nano /etc/nginx/sites-available/default
vm2@vm2-VirtualBox:~$ sudo systemctl restart nginx
vm2@vm2-VirtualBox:~$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset:
   Active: active (running) since Wed 2022-03-16 09:59:50 IST; 2s ago
     Docs: man:nginx(8)
   Process: 1833 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_proce
   Process: 1834 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (
 Main PID: 1835 (nginx)
    Tasks: 2 (limit: 1087)

Welcome to Ubuntu
system.slice/nginx.service
├─1835 nginx: master process /usr/sbin/nginx -g daemon on; master
└─1836 nginx: worker process

Mar 16 09:59:50 vm2-VirtualBox systemd[1]: Starting A high performance web ser
Mar 16 09:59:50 vm2-VirtualBox systemd[1]: Started A high performance web serv
lines 1-15/15 (END)
```

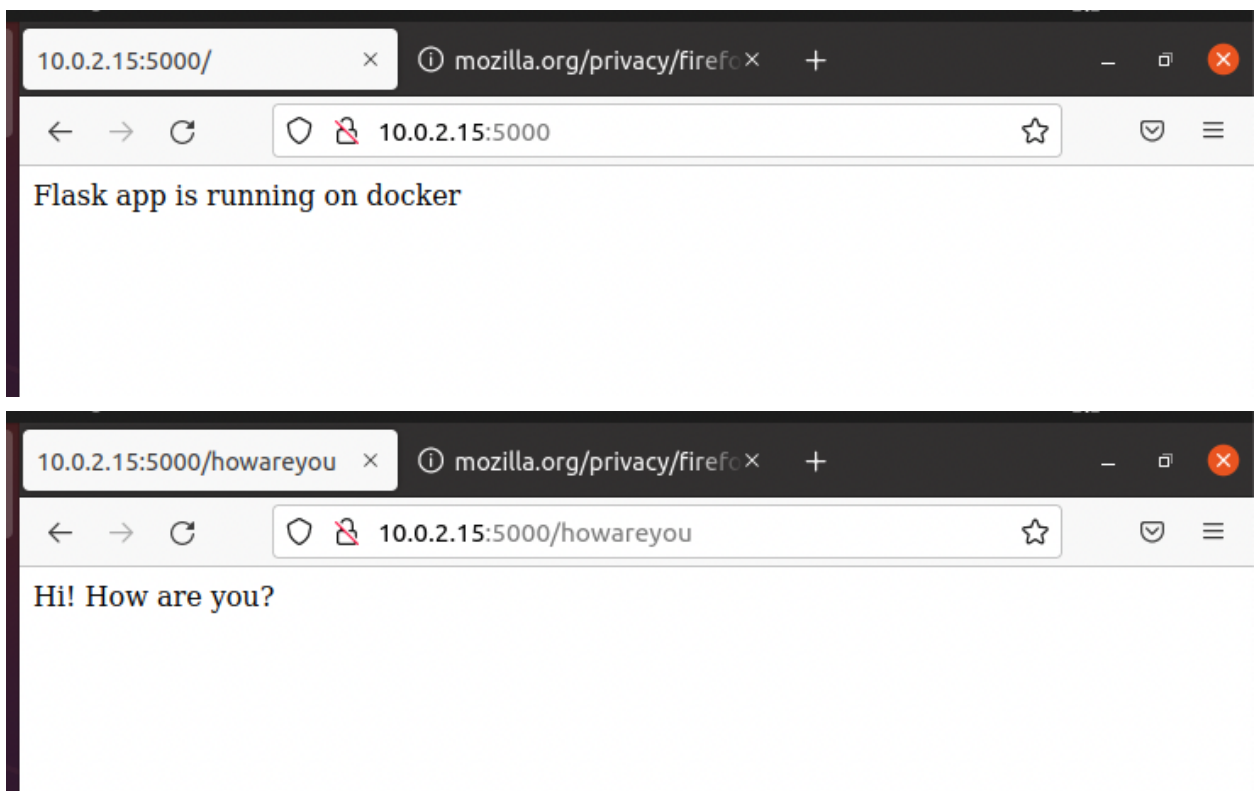
2. Do the configuration in the /etc/nginx/sites-available/default.

```
server_name 10.0.2.15;

location / {
    # First attempt to serve request as file, then
    # as directory, then fall back to displaying a 404.
    proxy_pass http://10.0.2.15:5000;
}

location /howareyou {
    proxy_pass http://10.0.2.15:5000/howareyou;
}
```

3. Restart the nginx and check the addresses on the browser.



4. Check the logs generated on VM1 by VM2 when we run it on the flask app.

```
vm1@vm1-VirtualBox:~$ sudo docker run -p 5000:5000 --network=host flask
[sudo] password for vm1:
WARNING: Published ports are discarded when using host network mode
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Running on all addresses.
  WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://10.0.2.15:5000/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 128-491-939
10.0.2.29 - - [16/Mar/2022 04:36:52] "GET / HTTP/1.0" 200 -
10.0.2.29 - - [16/Mar/2022 05:28:34] "GET / HTTP/1.1" 200 -
10.0.2.29 - - [16/Mar/2022 05:28:34] "GET /favicon.ico HTTP/1.1" 404 -
10.0.2.29 - - [16/Mar/2022 05:28:55] "GET /howareyou HTTP/1.1" 200 -
```

Now the nginx is serving as reverse proxy to container running in VM1