TASK 16

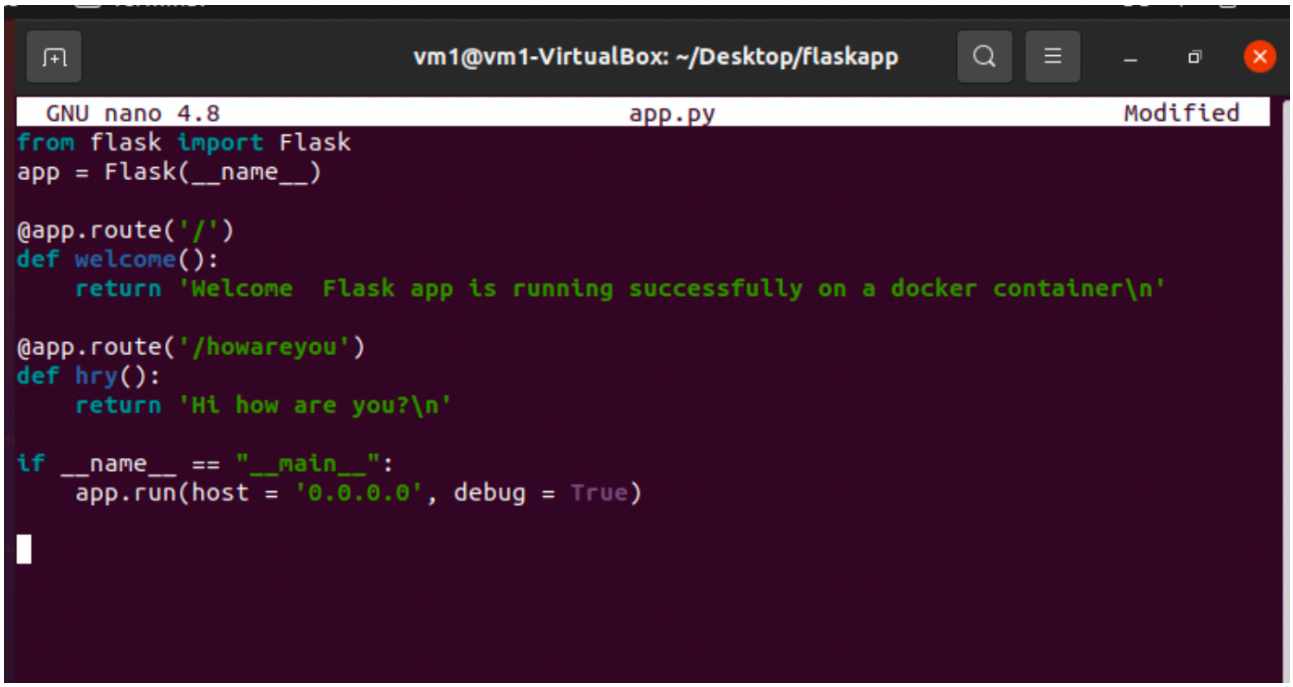
Machine 1: Over a container, setup a server serving an api /howareyou

STEPS:—

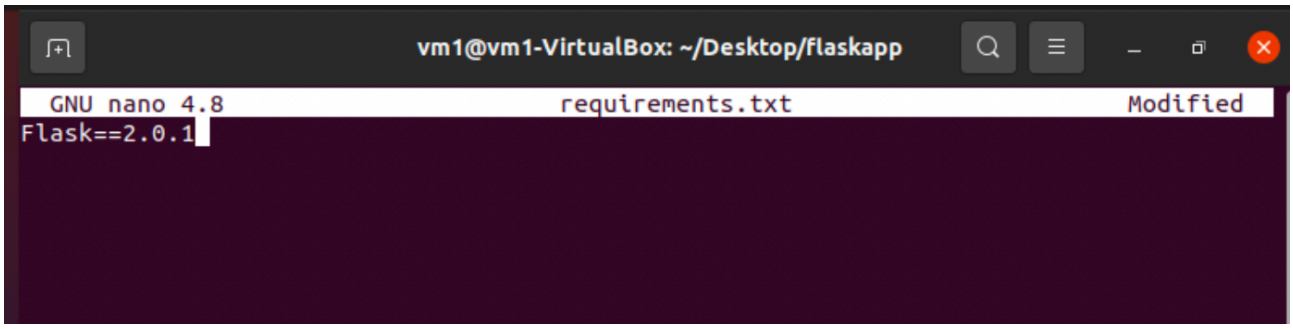
1. Install Docker and python3 on VM1.

2. Create a directory of the name flaskapp and in it create a file of the name app.py

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



4. Create a requirements.txt file and write the below line:



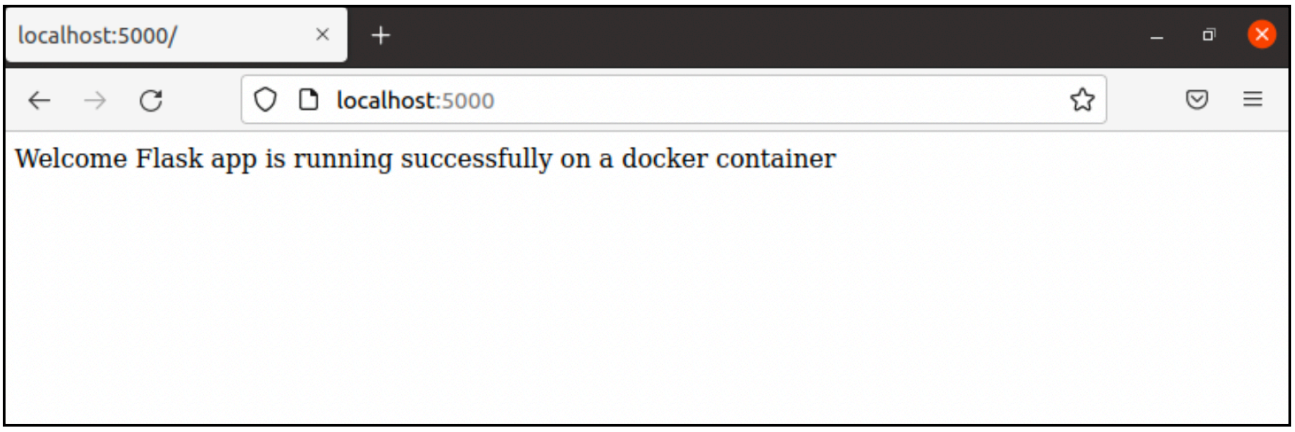
```
vm1@vm1-VirtualBox: ~/Desktop/flaskapp
  GNU nano 4.8                    Dockerfile                    Modified
FROM python:alpine3.7
COPY . /app
WORKDIR /app
RUN pip install -r requirements.txt
EXPOSE 5000
ENTRYPOINT [ "python" ]
CMD [ "app.py" ]
```

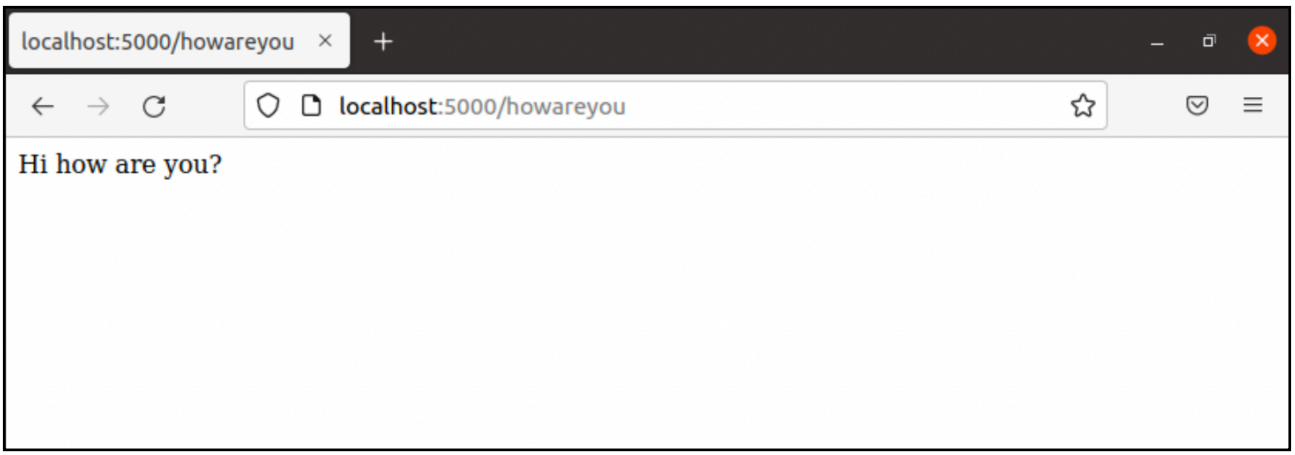
5. Create a Dockerfile as below:
6. Now build your docker file in the path where all your files are present (using the below command).

```
vm1@vm1-VirtualBox: ~/Desktop/flaskapp
vm1@vm1-VirtualBox:~/Desktop/flaskapp$ ls
app.py  Dockerfile  requirements.txt
vm1@vm1-VirtualBox:~/Desktop/flaskapp$ sudo docker build -t flask .
Sending build context to Docker daemon  4.096kB
Step 1/7 : FROM python:alpine3.7
alpine3.7: Pulling from library/python
48ecbb6b270e: Pull complete
692f29ee68fa: Pull complete
6439819450d1: Pull complete
3c7be240f7bf: Pull complete
ca4b349df8ed: Pull complete
Digest: sha256:35f6f83ab08f98c727dbefd53738e3b3174a48b4571ccb1910bae480dcdba847
Status: Downloaded newer image for python:alpine3.7
 ---> 00be2573e9f7
Step 2/7 : COPY . /app
 ---> 15718faede34
Step 3/7 : WORKDIR /app
 ---> Running in 0208a23aa063
Removing intermediate container 0208a23aa063
 ---> d6cb7757aa57
Step 4/7 : RUN pip install -r requirements.txt
 ---> Running in f3cfed3b0b1d
Collecting Flask==2.0.1 (from -r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/54/4f/1b294c1a4ab7b2ad5ca5fc4a9a
65a22ef1ac48be126289d97668852d4ab3/Flask-2.0.1-py3-none-any.whl (94kB)
Collecting Werkzeug>=2.0 (from Flask==2.0.1->-r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/f4/f3/22afbdb20cc4654b10c9804341
4a14057cd27fdba9d4ae61cea596000ba2/Werkzeug-2.0.3-py3-none-any.whl (289kB)
Collecting click>=7.1.2 (from Flask==2.0.1->-r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/4a/a8/0b2ced25639fb20cc1c9784de9
0a8c25f9504a7f18cd8b5397bd61696d7d/click-8.0.4-py3-none-any.whl (97kB)
Collecting Jinja2>=3.0 (from Flask==2.0.1->-r requirements.txt (line 1))
```

7. Use the docker run command to run your docker image and start your flask app on your localhost (using the command below):

```
^Cvm1@vm1-VirtualBox:~/Desktop/flaskapp$ sudo docker run -p 5000:5000 --network=host flask
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: 737-926-148
```



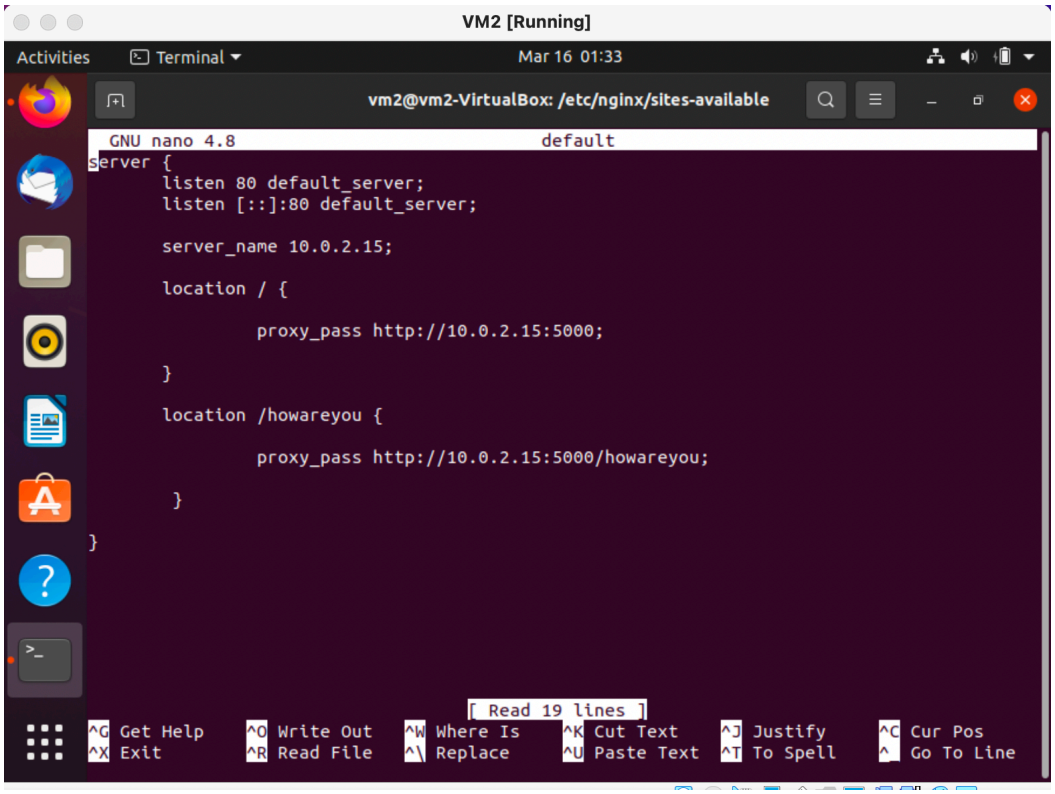


Machine 2: Setup a server using nginx that serves as front (reverse proxy) to container running in machine 1 Proxy server should return request for /howareyou when queried.

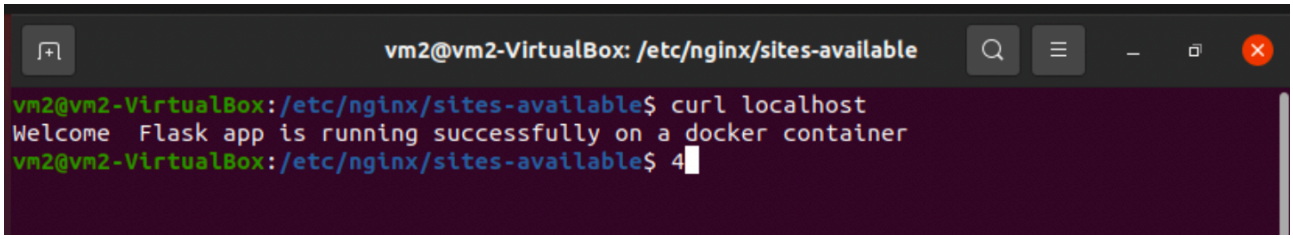
STEPS:-

1. Install nginx on VM2 using the below command:
   sudo apt install nginx

2. Now do the below configuration in **/etc/nginx/sites-available/default** file



3. Now restart nginx and try to curl localhost and localhost/howareyou you will get the desired result on your Vm2



```
vm2@vm2-VirtualBox:/etc/nginx/sites-available$ curl localhost/howareyou
Hi how are you?
```

4. Here on Vm1 we will be able to see logs of vm2 accessing our flask app

```
^Cvm1@vm1-VirtualBox:~/Desktop/flaskapp$ sudo docker run -p 5000:5000 --network=host flask
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: 347-367-044
10.0.2.7 - - [15/Mar/2022 20:05:00] "GET / HTTP/1.0" 200 -
10.0.2.7 - - [15/Mar/2022 20:05:24] "GET /howareyou HTTP/1.0" 200 -
```

Hence we are successfully able setup a server using nginx that serves as front (reverse proxy) to container running in Vm1.