

Task

Hosting Python Flask App using Docker

Name – Shashank Sharma

1. Create EC2 instance with ubuntu image.
2. In instance install docker. (before installing docker you should be in root user using command – sudo -i)

Link - <https://docs.docker.com/engine/install/ubuntu/>

3. Make git clone of repo .

git clone <https://github.com/shashanksharma1309/python-flask-docker-image.git>

(for files you can clone my repo using

Link - <https://github.com/shashanksharma1309/python-flask-docker-image.git>)

```
root@ip-172-31-23-222:~# git clone https://github.com/shashanksharma1309/python-flask-docker-image.git
Cloning into 'python-flask-docker-image'...
remote: Enumerating objects: 23, done.
remote: Counting objects: 100% (23/23), done.
remote: Compressing objects: 100% (21/21), done.
remote: Total 23 (delta 2), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (23/23), 1.31 MiB | 5.74 MiB/s, done.
Resolving deltas: 100% (2/2), done.
```

4. After completing the git clone go upto the your repo.

In my case

ls

cd python-flask-docker-image

ls

vim Docker file (create a docker file)

```
root@ip-172-31-23-222:~# ls
python-flask-docker-image snap
root@ip-172-31-23-222:~# cd python-flask-docker-image/
root@ip-172-31-23-222:~/python-flask-docker-image# ls
Procfile README.md app.py nltk.txt readme_images requirements.txt runtime.txt static summarizer.py templates
root@ip-172-31-23-222:~/python-flask-docker-image# vim Dockerfile
```

5. After entering into the Dockerfile add docker image script.

vim Dockerfile

```
FROM python:3.8

LABEL Folder="FlaskApp"

LABEL Author="shashank"

COPY . .

RUN pip install -r requirements.txt

EXPOSE 5000

CMD ["flask", "run", "--host", "0.0.0.0"]

~
~
~
```

6. After writing script save & exit from it.

7. Use command to build docker image.

docker build .

```
root@ip-172-31-19-91:~/python-flask-dockerfile# docker build .
[+] Building 30.2s (8/8) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 204B
=> [internal] load metadata for docker.io/library/python:3.8
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load build context
=> => transferring context: 30.53kB
=> [1/3] FROM docker.io/library/python:3.8sha256:23e62414c3310930888bb1690b7f723f52f7ab3a26ff9671e9747f60d169ee96
=> => resolve docker.io/library/python:3.8sha256:23e62414c3310930888bb1690b7f723f52f7ab3a26ff9671e9747f60d169ee96
=> => sha256:22c5bcf0d5e7e36298568c4c3c596dd5993e68f6478cefb13c9f65a9cb1b835 7.38kB / 7.38kB
=> => sha256:3cb5f9c23302e175d87a827f0alc376bd59b1f6949bd3bc24ab8da0d669cdfa0 24.05MB / 24.05MB
=> => sha256:5f899db30843f8330d5a40dlac26bb00e93a9f21bfff253f31c20562fa264767 64.14MB / 64.14MB
=> => sha256:23e62414c3310930888bb1690b7f723f52f7ab3a26ff9671e9747f60d169ee96 1.86kB / 1.86kB
=> => sha256:71215d55680cf0ab2dccc0e1dd65ed76414e3fb0c294249b5b9319a8fa7c399e4 49.55MB / 49.55MB
=> => sha256:d6b065a56b298584b27245bflch1fe6a4714184d349f1fb0c910a0c99d3f153ea 2.01kB / 2.01kB
=> => sha256:567db30df8d441ffe43e050ede26996c87e3b3c99f79d4fba0bfdb7ffa0213 211.14MB / 211.14MB
=> => sha256:d68cd2123173935e339e3feb56990a0aefdf7364ad43ca2b9750699e60fbf74c6 6.39MB / 6.39MB
=> => sha256:cd85456c3b32156db361af5b11d0830f05edc527e63ebf224f5c9ffde08ab08 17.28MB / 17.28MB
=> => extracting sha256:71215d55680cf0ab2dccc0e1dd65ed76414e3fb0c294249b5b9319a8fa7c399e4
=> => sha256:e0a9b8fc4891b1c7703200916d1c542972066f67e3854812d01ef405a6520596 245B / 245B
=> => sha256:3f14a07c5a5fe7f08cf239a72d7446451ald7e1e071dc6214bf5366f1c0a9159 2.85MB / 2.85MB
=> => extracting sha256:3cb5f9c23302e175d87a827f0alc376bd59b1f6949bd3bc24ab8da0d669cdfa0
=> => extracting sha256:5f899db30843f8330d5a40dlac26bb00e93a9f21bfff253f31c20562fa264767
=> => extracting sha256:567db30df8d441ffe43e050ede26996c87e3b3c99f79d4fba0bfdb7ffa0213
=> => extracting sha256:d68cd2123173935e339e3feb56990a0aefdf7364ad43ca2b9750699e60fbf74c6
=> => extracting sha256:cd85456c3b32156db361af5b11d0830f05edc527e63ebf224f5c9ffde08ab08
=> => extracting sha256:e0a9b8fc4891b1c7703200916d1c542972066f67e3854812d01ef405a6520596
=> => extracting sha256:3f14a07c5a5fe7f08cf239a72d7446451ald7e1e071dc6214bf5366f1c0a9159
=> [2/3] COPY . .
=> [3/3] RUN pip install -r requirements.txt
=> exporting to image
=> exporting layers
=> writing image sha256:a5f37e3ff37011d5fa022ef2f9f0b21116affad4ba4ba02e66fbd14395d9f9
```

8. After completion of image building use command to see docker image.

docker images

```
root@ip-172-31-23-222:~/python-flask-docker-image# docker images
REPOSITORY          TAG          IMAGE ID          CREATED          SIZE
<none>              <none>      49f032826faa     About a minute ago  1.68GB
shashank9532dockerhub/docker flask        a89aa627562e     45 minutes ago    1.68GB
root@ip-172-31-23-222:~/python-flask-docker-image#
```

9. Add port 5000 to the image.

docker run -d -p 5000:5000 <image-id>

```
root@ip-172-31-23-222:~/python-flask-docker-image# docker run -d -p 5000:5000 49f
da33af0df10810123638ef0bf2d7200c11a809793386bdea71eac9f08ff9e39b
root@ip-172-31-23-222:~/python-flask-docker-image#
```

10. Hit the instance IP to check application.

<http://<instance-ip>:5000/web/>

In my case

<http://13.57.237.154:5000/web/>

