

## 1. Build a Docker container from a custom Dockerfile

Step-1: Create a Dockerfile using **nano** in the pwd

→ **Dockerfile**

```
FROM python:3.9-slim
WORKDIR /app
COPY req.txt .
RUN pip install --no-cache-dir -r req.txt
COPY . .
ENV FLASK_APP=app.py
ENV FLASK_RUN_HOST=0.0.0.0
ENV FLASK_RUN_PORT=5000
EXPOSE 5000
CMD ["flask","run"]
```

Step-2: Create a python file **app.py**

→ **app.py**

```
from flask import Flask

app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello, Dockerrrr...!"

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000)
```

Step-3: Create a requirements file **req.txt**

flask

```
niranjan@ubuntu: ~/Devops/P1          niranjan@ubuntu: ~/Devops/P2
niranjan@ubuntu:~/Devops/P1$ nano Dockerfile
niranjan@ubuntu:~/Devops/P1$ nano app.py
niranjan@ubuntu:~/Devops/P1$ nano req.txt
niranjan@ubuntu:~/Devops/P1$ sudo docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
secondfile      latest   227209cd2cf2  4 days ago   17.8MB
hello-world     latest   1b44b5a3e06a  2 months ago  10.1kB
niranjan@ubuntu:~/Devops/P1$ sudo docker build -t p1 .
[+] Building 2.7s (10/10) FINISHED
          docker:default
=> [internal] load build definition from Dockerfile
               0.0s
=> => transferring dockerfile: 267B
               0.0s
=> [internal] load metadata for docker.io/library/python:3.9-slim
               2.6s
=> [internal] load .dockerignore
               0.0s
=> => transferring context: 2B
               0.0s
=> [1/5] FROM docker.io/library/python:3.9-slim@sha256:545badebace9a958b
98d3e272f0f0d46c0a1a389ac77e24c33f2e7b548ce1b6b
=> [internal] load build context
               0.0s
=> => transferring context: 518B
               0.0s
=> CACHED [2/5] WORKDIR /app
               0.0s
=> CACHED [3/5] COPY req.txt .
               0.0s
=> CACHED [4/5] RUN pip install --no-cache-dir -r req.txt
               0.0s
=> [5/5] COPY . .
               0.0s
=> exporting to image
               0.0s
=> => exporting layers
               0.0s
=> => writing image sha256:903d731a4933d3daf88b77d157e39384b0504dc6edd3
428417ba382f8d4b002
               0.0s
```

Step-4: Build the docker image using the below command

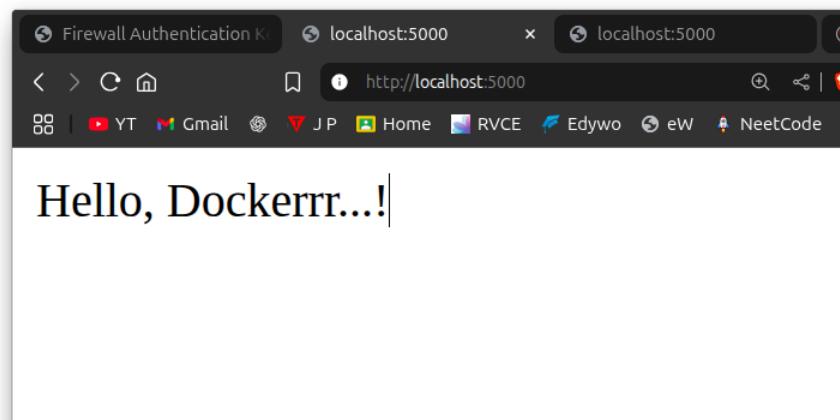
```
docker build -t p1 .
```

Step-5: Run the Docker Run command to create a container specifying the port number and name for the container

```
docker run -d -p 5000:5000 --name flask-app p1
```

```
niranjan@ubuntu:~/Devops/P1$ sudo docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
p1              latest   903d731a4933  7 seconds ago  132MB
secondfile      latest   227209cd2cf2  4 days ago   17.8MB
hello-world     latest   1b44b5a3e06a  2 months ago  10.1kB
niranjan@ubuntu:~/Devops/P1$ sudo docker run -d -p 5000:5000 --name Program-1 p1
bbb828aeb0f23deb66efd2398528c4b132315052d2938f7e7ac756def986701c
niranjan@ubuntu:~/Devops/P1$ sudo docker ps
CONTAINER ID      IMAGE      COMMAND      CREATED      STATUS      PORTS
TS
bbb828aeb0f2      p1        "flask run"  7 seconds ago  Up 6 seconds  0.0
.0.0:5000->5000/tcp, [::]:5000->5000/tcp  Program-1
```

Step-6: Test the container by verifying whether its showing some results on <http://localhost:5000> stating “**hello Dockerrr!**”



You can also check the status of the container by using below command

```
docker ps // displays current running containers
Docker ps -a // displays all present containers
```

Step-7: Stop the container, remove the container using container number

```
sudo docker container ls -a or docker ps -a  
sudo docker container stop <container-id>  
sudo docker container rm <container-id>  
sudo docker image rm <image-id> or docker rmi <image-id>
```

```
niranjan@ubuntu:~/Devops/P1$ sudo docker ps  
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES  
TS  
bbb828aeb0f2 p1 "flask run" 7 seconds ago Up 6 seconds 0.0  
.0.0:5000->5000/tcp, [::]:5000->5000/tcp Program-1  
niranjan@ubuntu:~/Devops/P1$ sudo docker stop bbb  
bbb  
niranjan@ubuntu:~/Devops/P1$ sudo docker rm bbb  
bbb  
niranjan@ubuntu:~/Devops/P1$ sudo docker rmi p1  
Untagged: p1:latest  
Deleted: sha256:903d731a4933d3daf88b77d157e39384b0504cd6edd3428417ba382f  
8d4b002  
niranjan@ubuntu:~/Devops/P1$ sudo docker images  
REPOSITORY TAG IMAGE ID CREATED SIZE  
secondfile latest 227209cd2cf2 4 days ago 17.8MB  
hello-world latest 1b44b5a3e06a 2 months ago 10.1kB  
niranjan@ubuntu:~/Devops/P1$ █
```