2: Create a volume, mount the volume and configure it in docker-compose.yaml file such that the code can be edited inside the container without having to rebuild the docker image. Use the same Web application created in Task 1 to test.

Steps:-

1. Create a directory having the following files - app.py , Dockerfile , requirements.txt , docker-compose.yml
2. Create a requirements.txt with the following content :-
   flask
   redis
3. Create a Dockerfile having the following content
   FROM python:2.7
   ADD . /code
   WORKDIR /code
   RUN pip install -r requirements.txt
   CMD python app.py
4. Create docker-compose.yml file with the following content:-

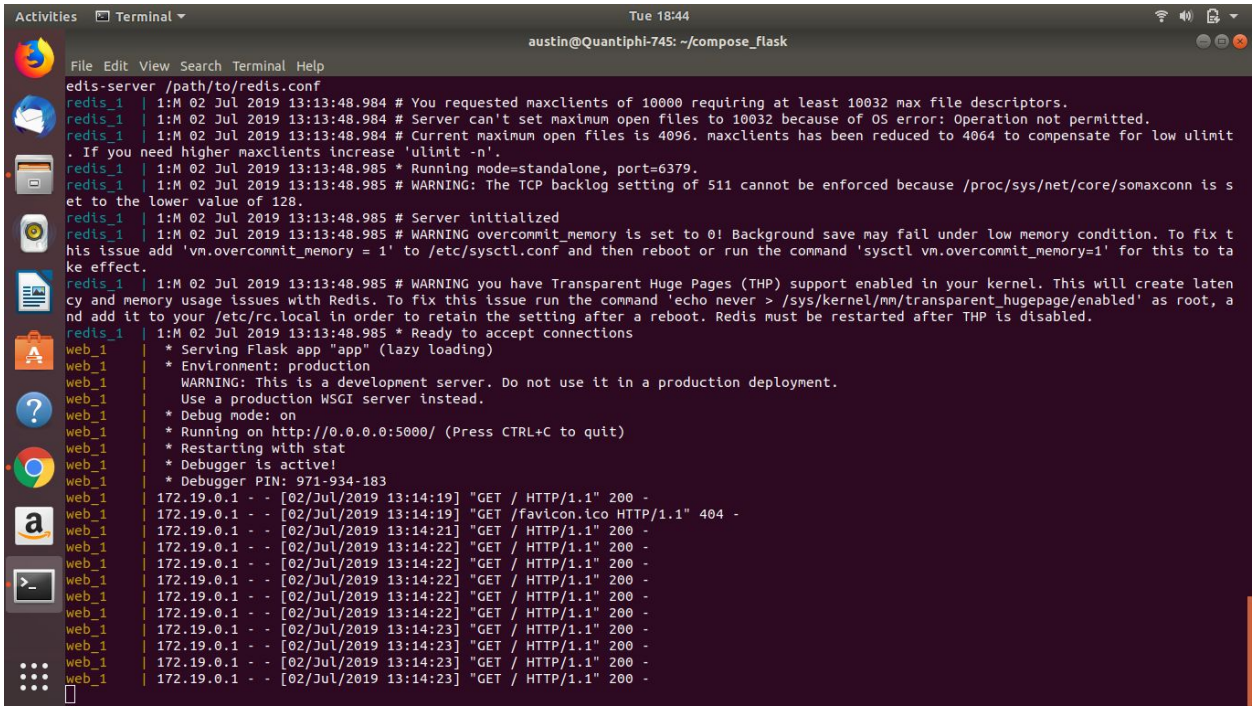
```
version: '2'
 services:
    web:
       build: .
       ports:
          - "5000:5000"
       volumes:
          - .:/code
       depends_on:
          - redis
    redis:
       image: redis
```

5. Create app.py with the following content :-

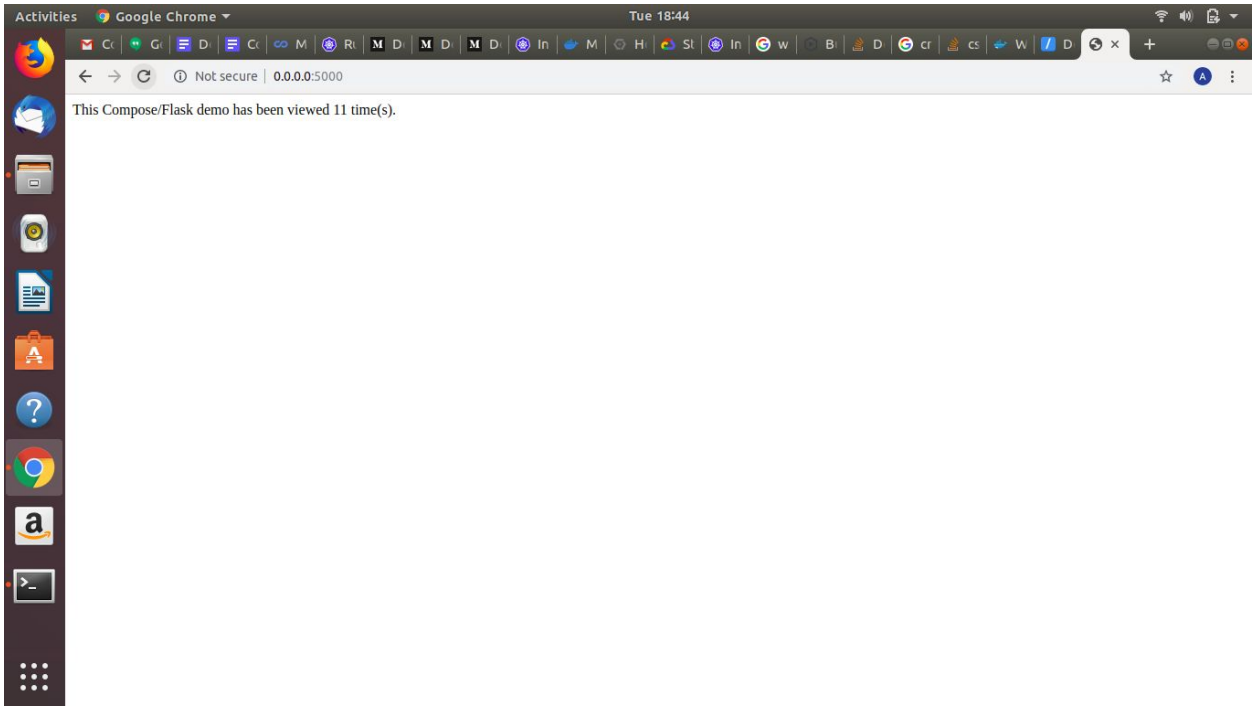
```
# compose_flask/app.py
from flask import Flask
from redis import Redis

app = Flask(__name__)
redis = Redis(host='redis', port=6379)

@app.route('/')
def hello():
   redis.incr('hits')
   return 'Hey Docker! The file has been viewed %s time(s).' % redis.get('hits')
```

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

6. Now build your docker with the following command
   sudo docker build -t compose_flask .
7. Now run
   sudo docker-compose up



Each refresh is counted as a hit



The new **volumes** key allows us to modify the code on the fly, without having to rebuild the image.