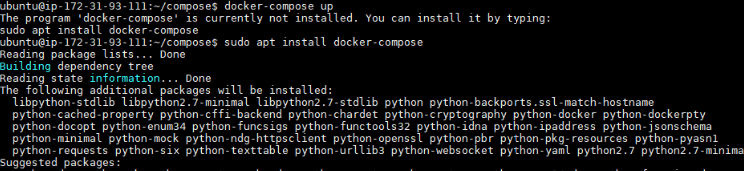
**QUESTION 1.**

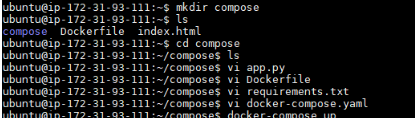
Step 1: Created a Python web application which runs on Docker Compose.It uses the Flask and maintains a hit counter in Redis.

1. Installed docker and docker-compose



1. Started the docker service.
2. Created a directory and inserted four files into it.( app.py, requirements.txt,

Docker-compose.yaml , Dockerfile )



1. The application is written in python as

**Code in app.py:**

import time

import redis

from flask import Flask

app = Flask(\_\_name\_\_)

cache = redis.Redis(host='redis', port=6379)

def get\_hit\_count():

retries = 5

while True:

try:

return cache.incr('hits')

except redis.exceptions.ConnectionError as exc:

if retries == 0:

raise exc

retries -= 1

time.sleep(0.5)

@app.route('/')

def hello():

count = get\_hit\_count()

return 'Hello World! I have been seen {} times.\n'.format(count)

if \_\_name\_\_ == "\_\_main\_\_":

app.run(host="0.0.0.0", debug=True)

1. The image is built from the Dockerfile instructions

**Dockerfile :**

FROM python:3.7-alpine ( building an image with python as baseimage)

ADD . /code ( added current dir. (.) to path /code in the image )

WORKDIR /code ( set the current dir. as /code )

RUN pip install -r requirements.txt ( installled the dependencies)

CMD ["python", "app.py"] ( set default command to run app.py )

1. Defined two services in compose file, **docker-compose.yaml**:

version: '2'

services:

web:

build: . ( used the image build from Dockerfile )

ports:

- "5000:5000" (port forwarding container --> host )

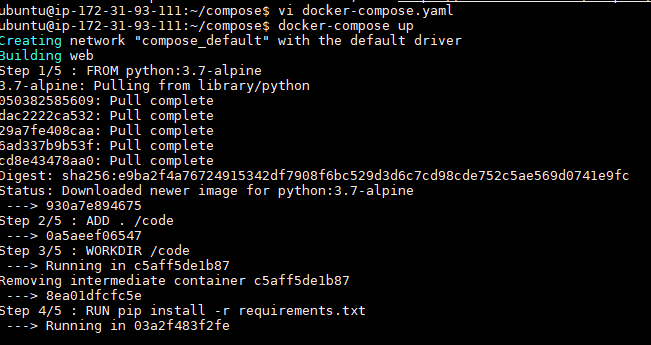
redis:

image: "redis:alpine" ( using public redis image )

1. Where host port (5000) is the default port for the flask web server.
2. Created four files as stated above

1. Run the command by being in the directory created ( /compose )

$ Docker-compose up



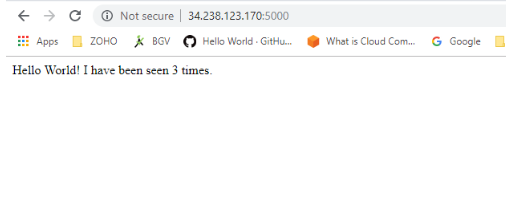
1. Compose pulls a Redis image, builds an image for our code in Dockerfile, and starts the

services (web and redis) we defined in docker-compose.yaml. Here, the code is statically

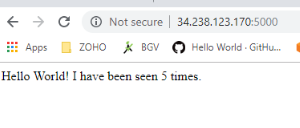
copied into the image at build time.

1. The web app had listened on port 5000 on our Docker daemon host. Entered

**http://my ip:5000/** in a browser to see the application running:



1. When we refreshed the page , the count has incremented accordingly.



1. Stopped the application by running the command $ docker-compose down
2. In this setup, whenever we modify any code in app.py then we need to rebuild the app image.

**QUESTION 2:**

1. Inorder to not to rebuild the image when we changed the code in app.py , we need to mount the volume.
2. mounted a volume via  volume key in the **docker-compose.yaml** as below

version: '2'

services:

web:

build: .

ports:

- "5000:5000"

Volumes :

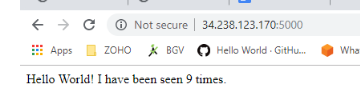
- . : /code ( **mounting the volume** )

redis:

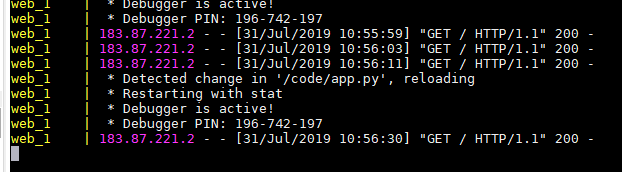
image: "redis:alpine"

1. We added a **bind mount** for the **web** service. The new volumes key mounts the project directory (current directory, '**.**') on the host to /code inside the container,
2. This allows us to modify the code on the fly, without having to rebuild the image.

Intially the message was as shown below



1. Now while the application is running , I modified the code(changed the greeting) in app.py and it got detected by the docker-compose



1. The change we made has reflected in the web application without having to rebuild the image.

