Cloud Computing

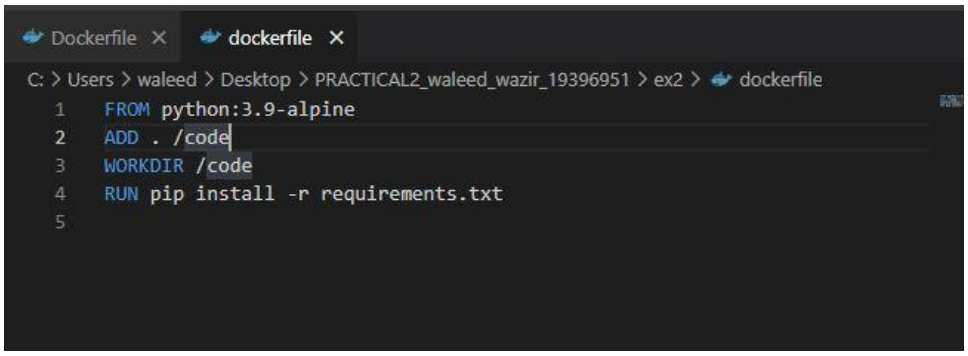
Practical 2

Using docker

Student No: 19396951

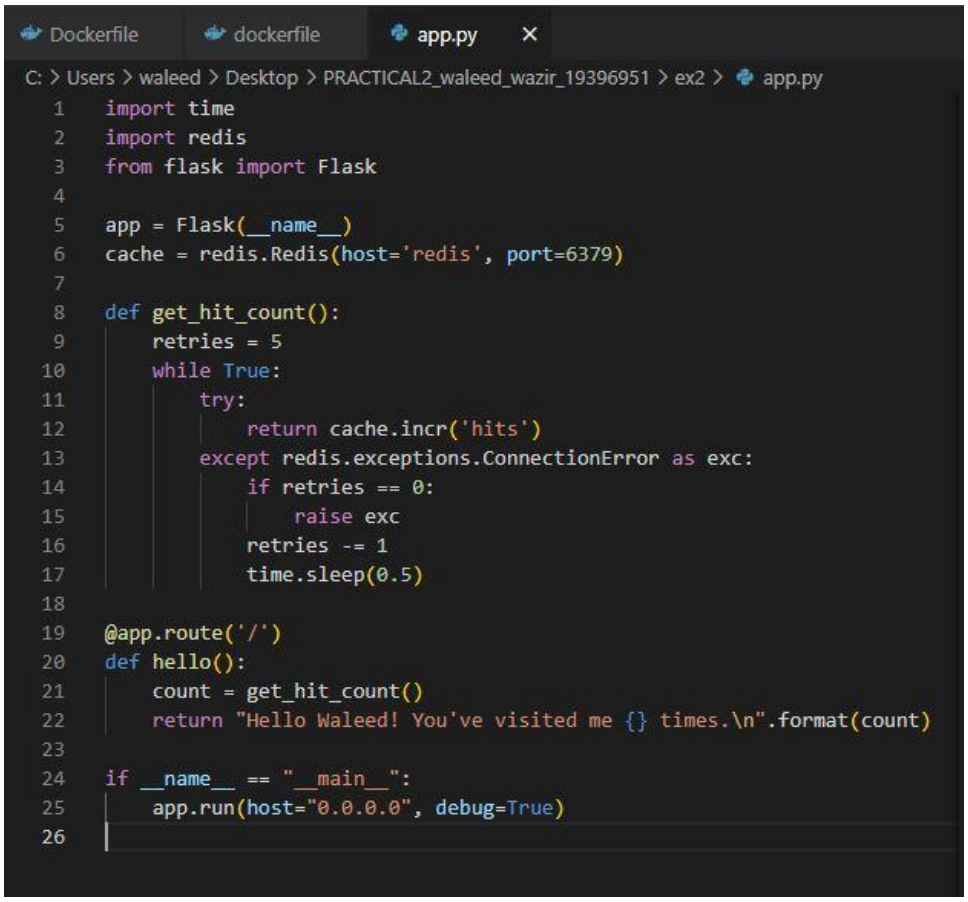
Exercise Two - docker-compose

Task 1: For this task I created a new file called dockerfile and put it into a folder called "ex2", in the dockerfile I put the code that was given to us in the practical pdf.


Dockerfile
dockerfile
C: > Users > waleed > Desktop > PRACTICAL2_waleed_wazir_19396951 > ex2 > dockerfile

```
FROM python:3.9-alpine
ADD . /code
WORKDIR /code
RUN pip install -r requirements.txt
```

I then created a file called app.py, in the image below we can see the code. I got this code from the website https://docs.docker.com/compose/gettingstarted/ and added the last two lines below which runs the python code app.py, In this code, redis is the hostname of the redis container on the application's network. We use the default port for Redis, 6379. Take note of the get_hit_count function's phrasing. If the redis service is unavailable, we can retry our request several times using this simple retry loop. This is helpful when the app first launches and is online, but it also strengthens its resilience in case the Redis service needs to be restarted at any point while the app is running. This aids in handling brief connection drops between nodes in a cluster.

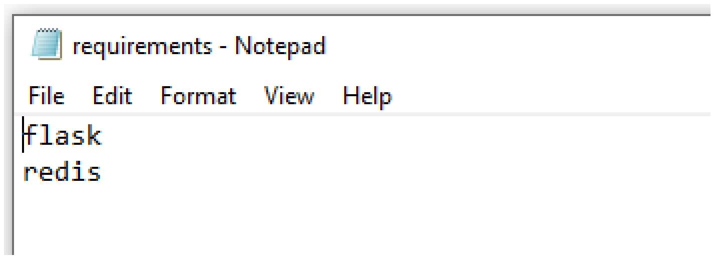


Dockerfile   dockerfile   app.py ×

C: > Users > waleed > Desktop > PRACTICAL2_waleed_wazir_19396951 > ex2 > app.py

```python
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 Waleed! You've visited me {} times.\n".format(count)

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

```

I then created another file called requirements.txt and put the file in a folder called "ex2". As we can see in the image above where we created a dockerfile in exercise 2 that docker file will use the requirements file and install the python dependencies.

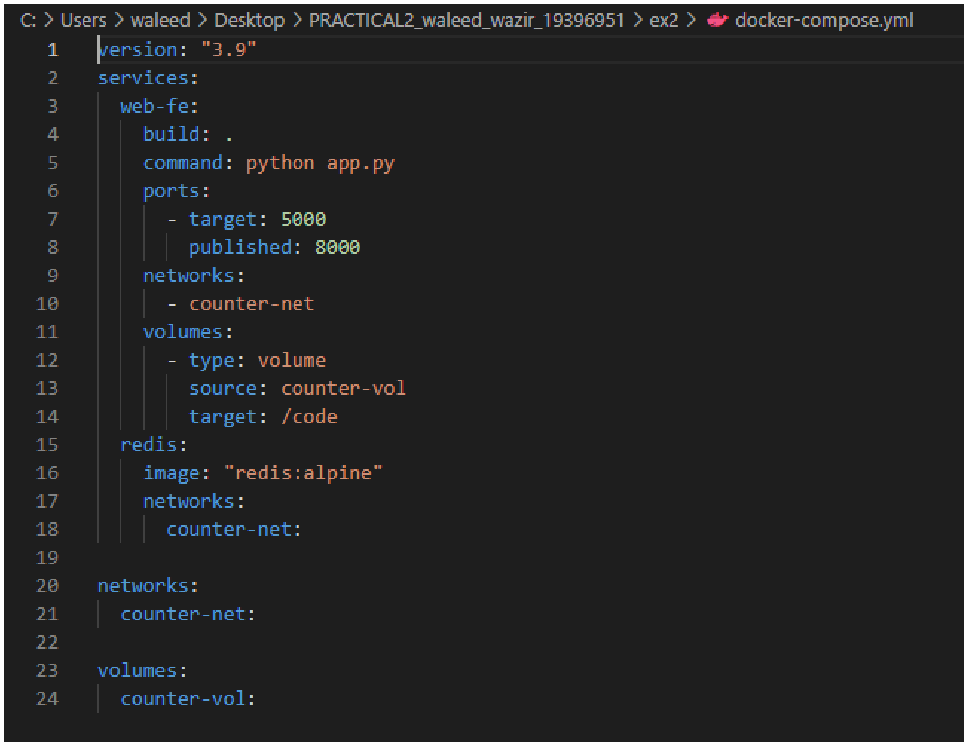


requirements - Notepad

File Edit Format View Help

```
flask
redis
```

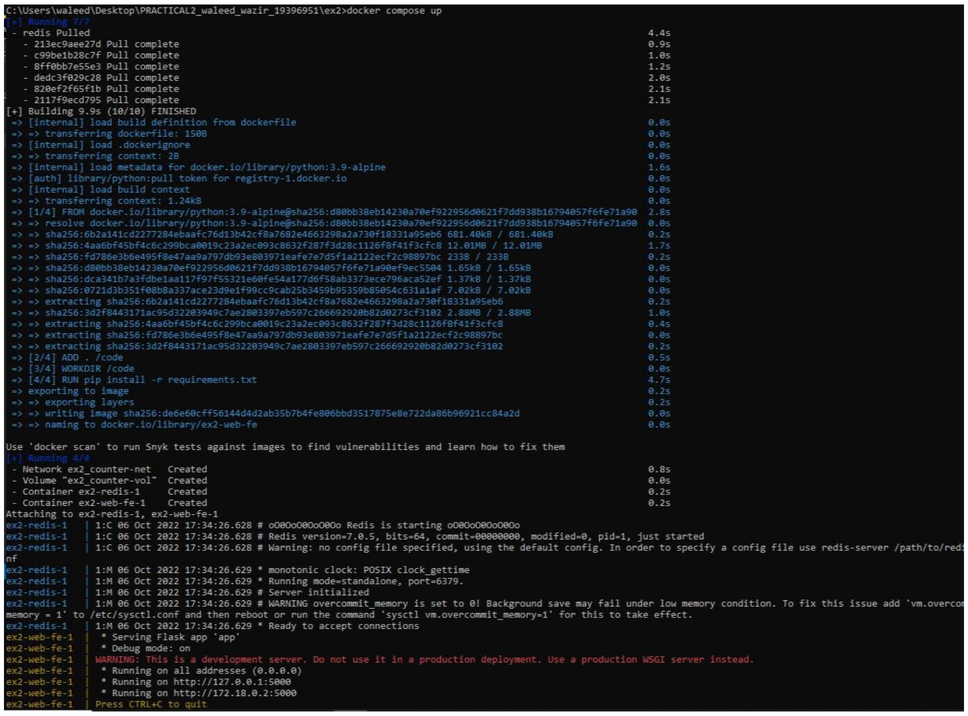
I then created a another file called docker-compose.yml and put the file in the folder "ex2", this is the source code below. Here we use the example from the website https://docs.docker.com/compose/gettingstarted/ and the add on to it. This compose file defines two services web and redis. The web service uses an image that's built from the Dockerfile in the current directory. It then binds the container and the host machine to the exposed port, 8000. This service uses the default port for the Flask web server, 5000.The redis service uses a public Redis image pulled from the Docker Hub registry. We the add on the network which allows us to comunicate between containers. and we also added volume which allows us to store the code of app.py on the web service, So volumes are used to store persistent data generated and used from containers.



C: > Users > waleed > Desktop > PRACTICAL2_waleed_wazir_19396951 > ex2 > docker-compose.yml

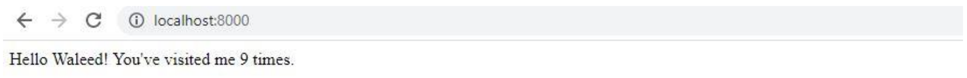
```yaml
version: "3.9"
services:
  web-fe:
    build: .
    command: python app.py
    ports:
      - target: 5000
        published: 8000
    networks:
      - counter-net
    volumes:
      - type: volume
        source: counter-vol
        target: /code
  redis:
    image: "redis:alpine"
    networks:
      counter-net:

networks:
  counter-net:

volumes:
  counter-vol:
```

After we have created this docker compose file we then run the command - docker compose up in cmd. This command will build and run our app.py with compose.



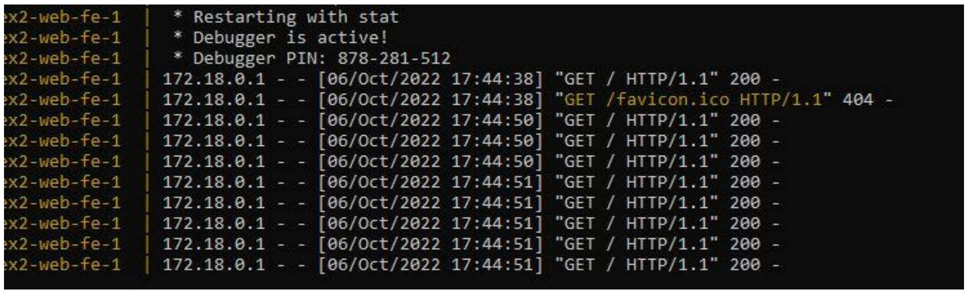
```
C:\Users\waleed\Desktop\PRACTICAL2_waleed_wazir_19396951\ex2>docker compose up
[+] Running 7/7
 - redis Pulled                                                                                        4.4s
   - 213ec9aee27d Pull complete                                                                        0.9s
   - c99be1b28c7f Pull complete                                                                        1.0s
   - 8ff0bb7e55e3 Pull complete                                                                        1.2s
   - dedc3f029c28 Pull complete                                                                        2.0s
   - 820ef2f65f1b Pull complete                                                                        2.1s
   - 2117f9ecd795 Pull complete                                                                        2.1s
[+] Building 9.9s (10/10) FINISHED
 => [internal] load build definition from dockerfile                                                   0.0s
 => => transferring dockerfile: 150B                                                                   0.0s
 => [internal] load .dockerignore                                                                      0.0s
 => => transferring context: 2B                                                                        0.0s
 => [internal] load metadata for docker.io/library/python:3.9-alpine                                   1.6s
 => [auth] library/python:pull token for registry-1.docker.io                                          0.0s
 => [internal] load build context                                                                      0.0s
 => => transferring context: 1.24kB                                                                    0.0s
 => [1/4] FROM docker.io/library/python:3.9-alpine@sha256:d80bb38eb14230a70ef922956d0621f7dd938b16794057f6fe71a90  2.8s
 => => resolve docker.io/library/python:3.9-alpine@sha256:d80bb38eb14230a70ef922956d0621f7dd938b16794057f6fe71a90  0.0s
 => => sha256:6b2a141cd2277284ebaafc76d13b42cf8a7682e4663298a2a730f18331a95eb6 681.40kB / 681.40kB     0.2s
 => => sha256:4aa6bf45bf4c6c299bca0019c23a2ec093c8632f287f3d28c1126f8f41f3cfc8 12.01MB / 12.01MB       1.7s
 => => sha256:fd786e3b6e495f8e47aa9a797db93e803971eafe7e7d5f1a2122ecf2c98897bc 233B / 233B             0.2s
 => => sha256:d80bb38eb14230a70ef922956d0621f7dd938b16794057f6fe71a90ef9ec5504 1.65kB / 1.65kB         0.0s
 => => sha256:dca341b7a3fdbe1aa117f97f55321e60fe54a177d6f58ab3373ece796aca52ef 1.37kB / 1.37kB         0.0s
 => => sha256:0721d3b351f08b8a337ace23d9e1f99cc9cab25b3459b95359b85054c631a1af 7.02kB / 7.02kB         0.0s
 => => extracting sha256:6b2a141cd2277284ebaafc76d13b42cf8a7682e4663298a2a730f18331a95eb6              0.2s
 => => sha256:3d2f8443171ac95d32203949c7ae2803397eb597c266692920b82d0273cf3102 2.88MB / 2.88MB         1.0s
 => => extracting sha256:4aa6bf45bf4c6c299bca0019c23a2ec093c8632f287f3d28c1126f8f41f3cfc8              0.4s
 => => extracting sha256:fd786e3b6e495f8e47aa9a797db93e803971eafe7e7d5f1a2122ecf2c98897bc              0.0s
 => => extracting sha256:3d2f8443171ac95d32203949c7ae2803397eb597c266692920b82d0273cf3102              0.2s
 => [2/4] ADD . /code                                                                                  0.5s
 => [3/4] WORKDIR /code                                                                                0.0s
 => [4/4] RUN pip install -r requirements.txt                                                          4.7s
 => exporting to image                                                                                 0.2s
 => => exporting layers                                                                                0.2s
 => => writing image sha256:de6e60cff56144d4d2ab35b7b4fe806bbd3517875e8e722da86b96921cc84a2d           0.0s
 => => naming to docker.io/library/ex2-web-fe                                                          0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
[+] Running 4/4
 - Network ex2_counter-net   Created                                                                   0.8s
 - Volume "ex2_counter-vol"  Created                                                                   0.0s
 - Container ex2-redis-1     Created                                                                   0.2s
 - Container ex2-web-fe-1    Created                                                                   0.2s
Attaching to ex2-redis-1, ex2-web-fe-1
ex2-redis-1   | 1:C 06 Oct 2022 17:34:26.628 # oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo
ex2-redis-1   | 1:C 06 Oct 2022 17:34:26.628 # Redis version=7.0.5, bits=64, commit=00000000, modified=0, pid=1, just started
ex2-redis-1   | 1:C 06 Oct 2022 17:34:26.628 # Warning: no config file specified, using the default config. In order to specify a config file use redis-server /path/to/redi
nf
ex2-redis-1   | 1:M 06 Oct 2022 17:34:26.629 * monotonic clock: POSIX clock_gettime
ex2-redis-1   | 1:M 06 Oct 2022 17:34:26.629 * Running mode=standalone, port=6379.
ex2-redis-1   | 1:M 06 Oct 2022 17:34:26.629 # Server initialized
ex2-redis-1   | 1:M 06 Oct 2022 17:34:26.629 # WARNING overcommit_memory is set to 0! Background save may fail under low memory condition. To fix this issue add 'vm.overcom
memory = 1' to /etc/sysctl.conf and then reboot or run the command 'sysctl vm.overcommit_memory=1' for this to take effect.
ex2-redis-1   | 1:M 06 Oct 2022 17:34:26.629 * Ready to accept connections
ex2-web-fe-1  |  * Serving Flask app 'app'
ex2-web-fe-1  |  * Debug mode: on
ex2-web-fe-1  | WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
ex2-web-fe-1  |  * Running on all addresses (0.0.0.0)
ex2-web-fe-1  |  * Running on http://127.0.0.1:5000
ex2-web-fe-1  |  * Running on http://172.18.0.2:5000
ex2-web-fe-1  | Press CTRL+C to quit
```

Compose pulls a Redis image, builds an image for your code, and starts the services you defined. In this case, the code is statically copied into the image at build time. We Enter http://localhost:8000/ in a browser to see the application running.



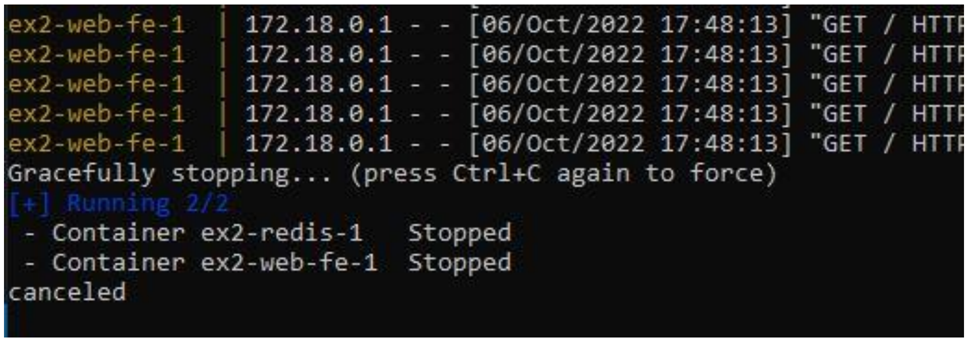
```
localhost:8000
Hello Waleed! You've visited me 9 times.
```

Every time I refreshed the page the number would be incremented and so far I have visited the web site 9 times, as we can see from the image below, the terminal records the amount of times I had revisited the website.



```
x2-web-fe-1  |  * Restarting with stat
x2-web-fe-1  |  * Debugger is active!
x2-web-fe-1  |  * Debugger PIN: 878-281-512
x2-web-fe-1  | 172.18.0.1 - - [06/Oct/2022 17:44:38] "GET / HTTP/1.1" 200 -
x2-web-fe-1  | 172.18.0.1 - - [06/Oct/2022 17:44:38] "GET /favicon.ico HTTP/1.1" 404 -
x2-web-fe-1  | 172.18.0.1 - - [06/Oct/2022 17:44:50] "GET / HTTP/1.1" 200 -
x2-web-fe-1  | 172.18.0.1 - - [06/Oct/2022 17:44:50] "GET / HTTP/1.1" 200 -
x2-web-fe-1  | 172.18.0.1 - - [06/Oct/2022 17:44:50] "GET / HTTP/1.1" 200 -
x2-web-fe-1  | 172.18.0.1 - - [06/Oct/2022 17:44:51] "GET / HTTP/1.1" 200 -
x2-web-fe-1  | 172.18.0.1 - - [06/Oct/2022 17:44:51] "GET / HTTP/1.1" 200 -
x2-web-fe-1  | 172.18.0.1 - - [06/Oct/2022 17:44:51] "GET / HTTP/1.1" 200 -
x2-web-fe-1  | 172.18.0.1 - - [06/Oct/2022 17:44:51] "GET / HTTP/1.1" 200 -
x2-web-fe-1  | 172.18.0.1 - - [06/Oct/2022 17:44:51] "GET / HTTP/1.1" 200 -
```

I then stopped the website using Ctrl+C.



```
ex2-web-fe-1  | 172.18.0.1 - - [06/Oct/2022 17:48:13] "GET / HTTP
ex2-web-fe-1  | 172.18.0.1 - - [06/Oct/2022 17:48:13] "GET / HTTP
ex2-web-fe-1  | 172.18.0.1 - - [06/Oct/2022 17:48:13] "GET / HTTP
ex2-web-fe-1  | 172.18.0.1 - - [06/Oct/2022 17:48:13] "GET / HTTP
ex2-web-fe-1  | 172.18.0.1 - - [06/Oct/2022 17:48:13] "GET / HTTP
Gracefully stopping... (press Ctrl+C again to force)
[+] Running 2/2
 - Container ex2-redis-1   Stopped
 - Container ex2-web-fe-1  Stopped
canceled
```