Docker-Compose

Created by

CHANDRADIP PATIL
Intern at

Unnati Development And Training Centre Pvt Ltd.

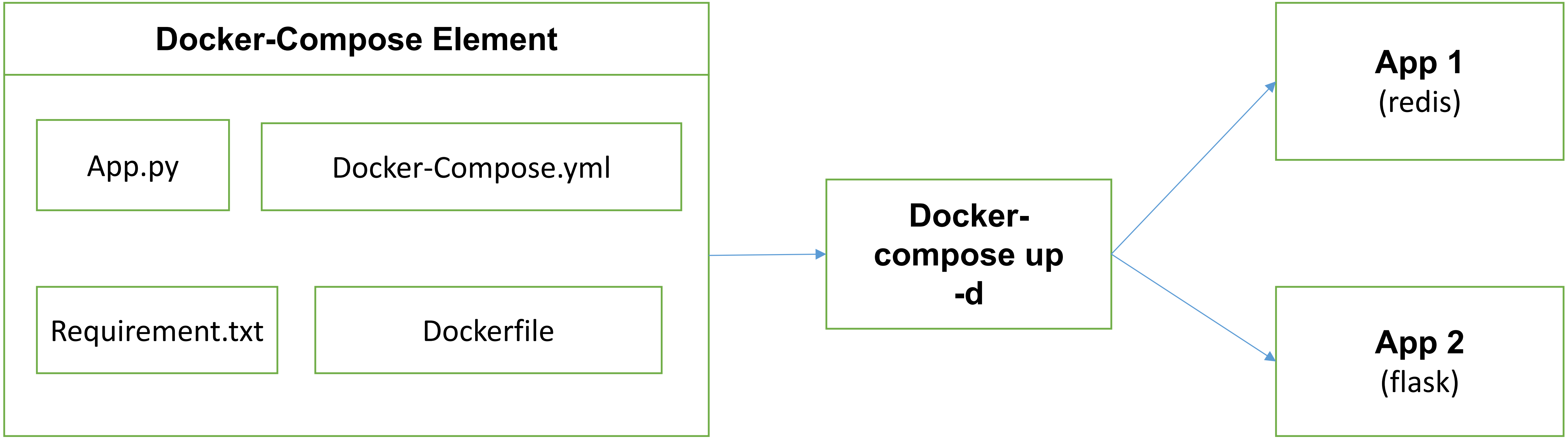
What is “Docker-Compose”

- Docker Compose is **a tool that was developed to help define and share multi-container applications**.

- With Compose, we can **create a YAML** file to define the services and with a single command, can spin everything up or tear it all down. ...

- Someone would only need to clone your repo and start the compose app.

Architecture Diagram



Docker-Compose Element

App.py

Docker-Compose.yml

Requirement.txt

Dockerfile

Docker-compose up -d

App 1 (redis)

App 2 (flask)

Use cases Of “Docker-Compose”

- Simplifying Configuration.
- Developer’s Productivity.
- Multi-Tendency.
- Rapid Deployment.
- Server Consolidation.
- Code-Pipeline Management.
- App Isolation.
- Debugging Capabilities.

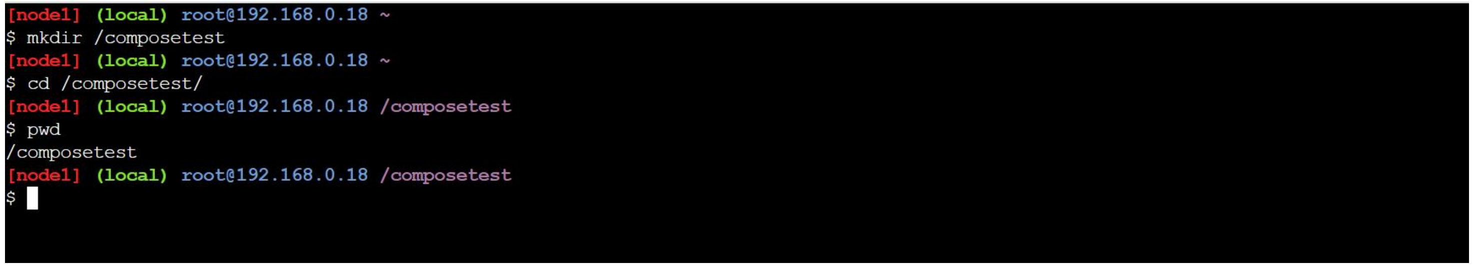
Steps to deploy…

- Docker Compose requires three steps:
- 1. **Define the application environment** with a Dockerfile.
- 2. **Define the application services** in docker-compose.yml.
- 3. Run docker-compose **to start and run** applications.

Let’s do some practical stuff now…

Create working directory

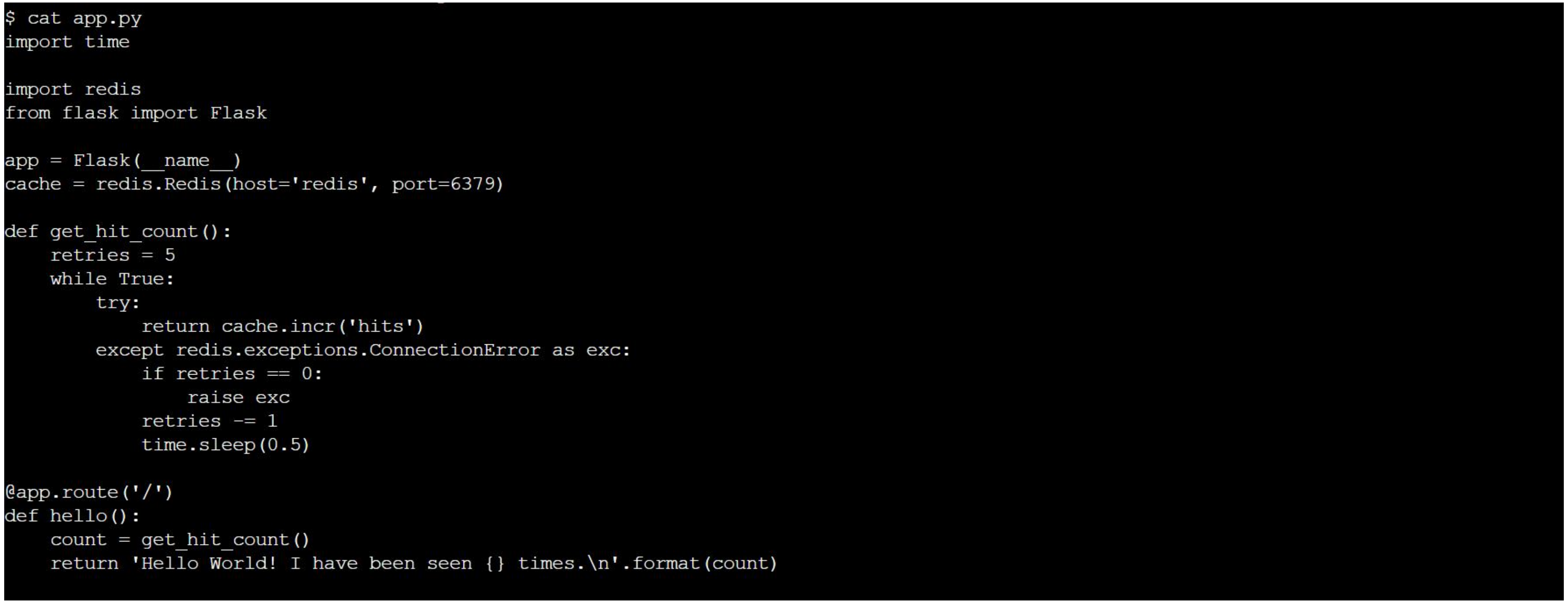
- Command:- mkdir /composetest



```
[node1] (local) root@192.168.0.18 ~
$ mkdir /composetest
[node1] (local) root@192.168.0.18 ~
$ cd /composetest/
[node1] (local) root@192.168.0.18 /composetest
$ pwd
/composetest
[node1] (local) root@192.168.0.18 /composetest
$
```

Create files

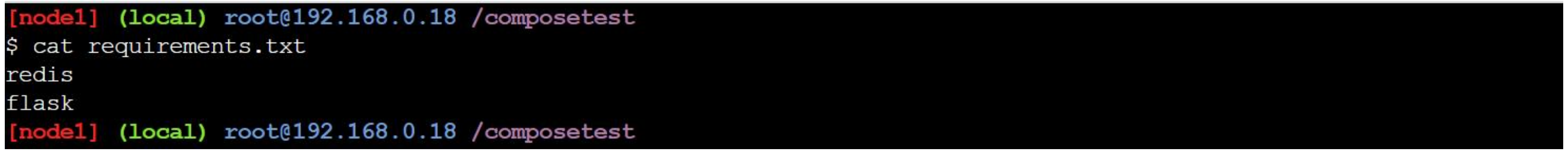
- cat app.py



```
$ cat 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)
```

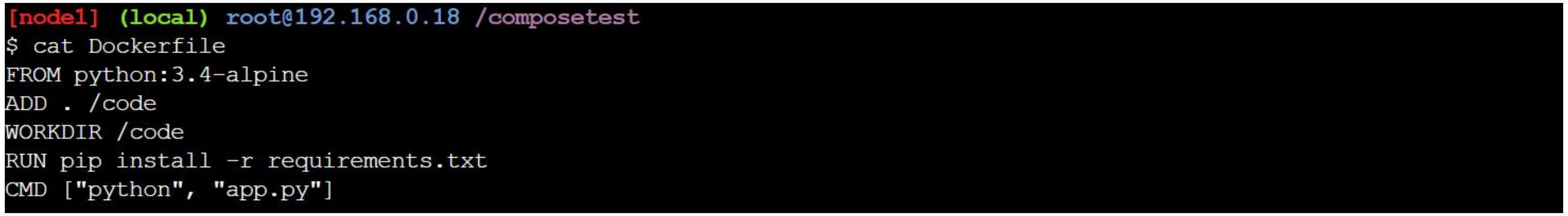
Crate dockerfile, yaml file and all stuffs.

- cat requirement.txt



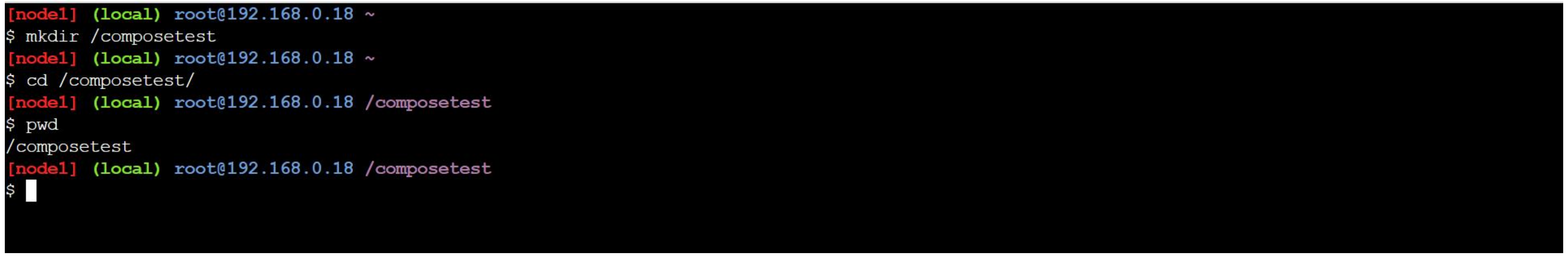
```
[node1] (local) root@192.168.0.18 /composetest
$ cat requirements.txt
redis
flask
[node1] (local) root@192.168.0.18 /composetest
```

- cat Dockerfile



```
[node1] (local) root@192.168.0.18 /composetest
$ cat Dockerfile
FROM python:3.4-alpine
ADD . /code
WORKDIR /code
RUN pip install -r requirements.txt
CMD ["python", "app.py"]
```

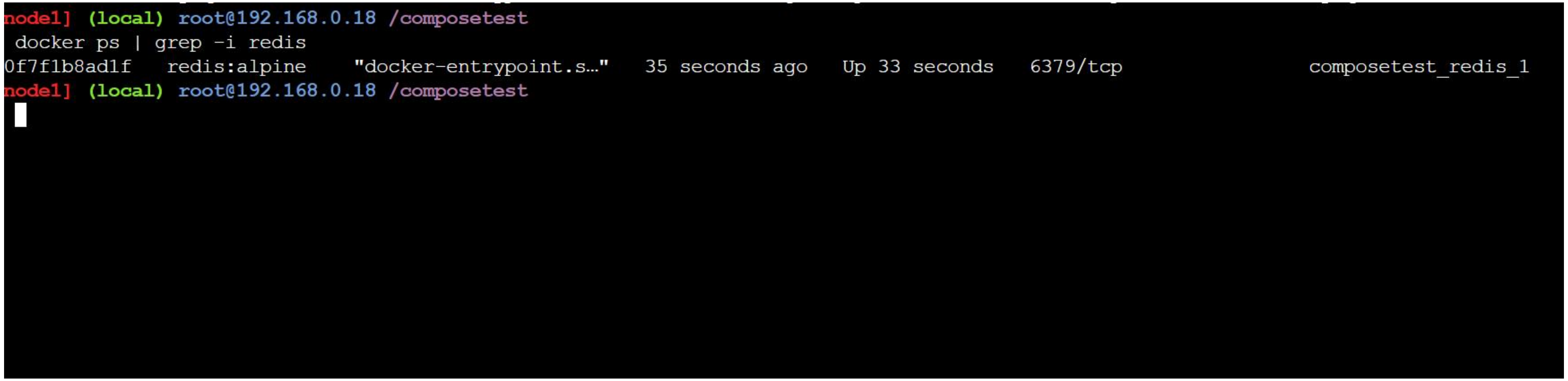
- cat docker-compose.yml



```
[node1] (local) root@192.168.0.18 ~
$ mkdir /composetest
[node1] (local) root@192.168.0.18 ~
$ cd /composetest/
[node1] (local) root@192.168.0.18 /composetest
$ pwd
/composetest
[node1] (local) root@192.168.0.18 /composetest
$
```

Compose container applications.

- Command:- docker-compose up –d



```
node1] (local) root@192.168.0.18 /composetest
 docker ps | grep -i redis
0f7f1b8ad1f   redis:alpine    "docker-entrypoint.s…"   35 seconds ago   Up 33 seconds   6379/tcp                composetest_redis_1
node1] (local) root@192.168.0.18 /composetest
```