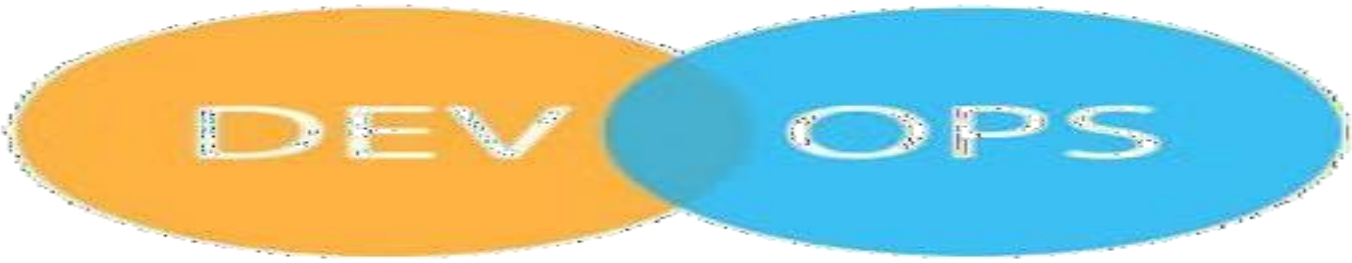

DEV
OPS

Docker Tasks:

Application-1:

Docker php with Apache

a) File: index.php

```
<?php
  echo "Welcome to Alnafi</br>"; echo
  "Running php with Apache on Docker";
?>
```

b) File:Dockerfile

```
FROM php:7.4-apache

COPY . /var/www/html
```

Build Docker Image

```
docker build -t img-php-apache-example .
```

Run the Docker Image

```
docker run -it -d -p 8080:80 img-php-apache-example
```

Application-2:

RUN php-CLI on Docker

```
cd ~/tutorial/

nano cli.php
```

```
<?php
  echo "Welcome to alnafi \n";
  echo "Running php CLI script with docker \n";
?>
```

$ nano Dockerfile

```
FROM php:7.4-cli

COPY . /usr/src/myapp

WORKDIR /usr/src/myapp

CMD[ "php", "./cli.php" ]
```

```
docker build -t  img-php-cli-example .

docker run -it  --rm img-php-cli-example
```

Application-3:
Run Python Script on Docker

$ nano script.py

```
print("Welcome to Alnafi");

print("This is Python running in Docker");
```

$ nano Dockerfile

```
FROM python:3

WORKDIR /usr/src/app


## Un-comment below lines to install dependencies

#COPY requirements.txt ./

#RUN pip install --no-cache-dir -r requirements.txt


COPY . .

CMD [ "python", "./script.py" ]
```

Application-4:

```
version: "3.9"

services:
  db:
    image: mysql:5.7
    volumes:
      - db_data:/var/lib/mysql
    restart: always
    environment:
```

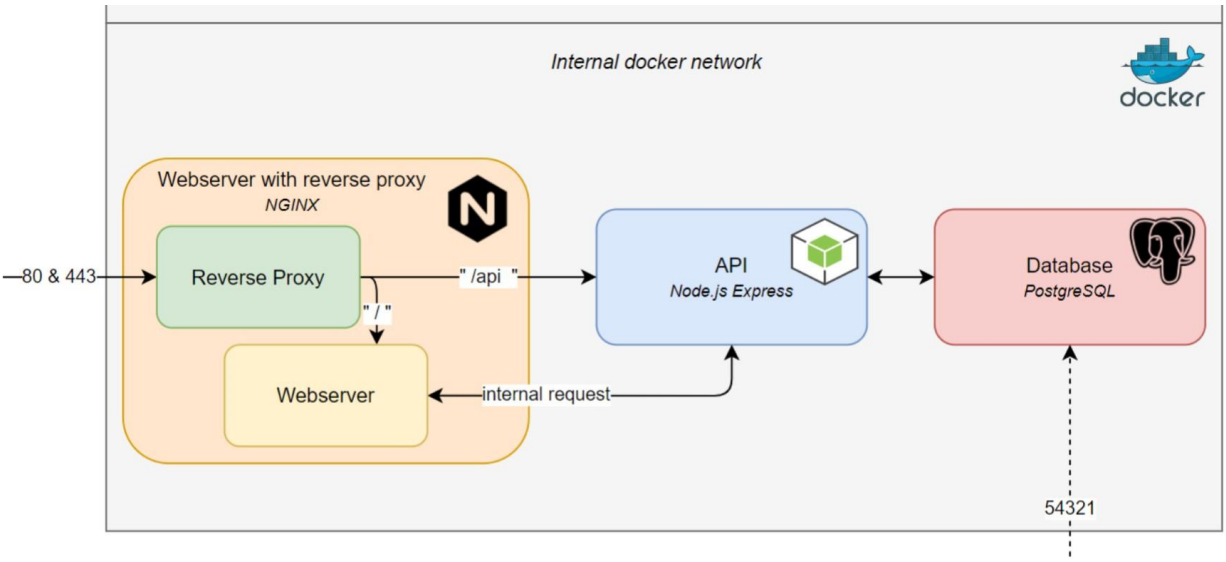
```
      MYSQL_ROOT_PASSWORD: somewordpress
      MYSQL_DATABASE: wordpress
      MYSQL_USER: wordpress
      MYSQL_PASSWORD: wordpress

  wordpress:
    depends_on:
      - db
    image: wordpress:latest
    volumes:
    - wordpress_data:/var/www/html
    ports:
      - "8000:80"
    restart: always
    environment:
      WORDPRESS_DB_HOST: db:3306
      WORDPRESS_DB_USER: wordpress
      WORDPRESS_DB_PASSWORD: wordpress
      WORDPRESS_DB_NAME: wordpress
volumes:
  db_data: {}
  wordpress_data: {}
```

Application-5:

beersnob_compose-main.zip

Extract the code and try to understand the same.



- Go to localhost:54322/api/test to test the API.
- Go to localhost or localhost:80 to test the webserver
- Use a database management system (like PgAdmin) to connect to our database on localhost:54321 with the credentials from the docker-compose.yml (line 12–14)