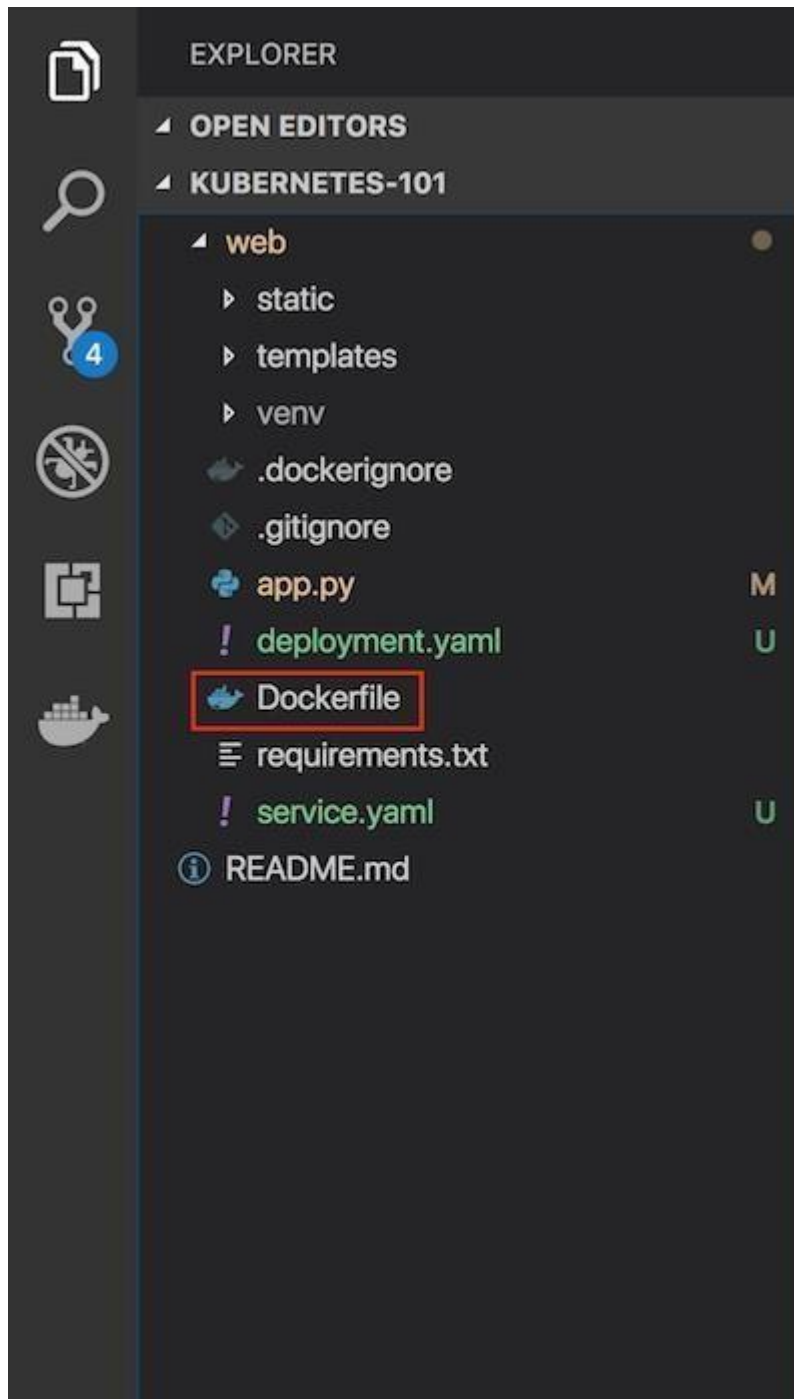


Containerize Flask Application

Date	12 November 2022
Team ID	PNT2022TMID43323
Project Name	Inventory Management System for Retailers

In project directory, create a file named "Dockerfile."



A "Dockerfile" is used to indicate to Docker a base image, the Docker settings you need, and a list of commands you would like to have executed to prepare and start your new container.

In the file, paste this code:

```
# init a base image (Alpine is small Linux distro)
FROM python:3.6
# update pip to minimize dependency errors
RUN pip install --upgrade pip
# define the present working directory
WORKDIR /docker-flask-test
# copy the contents into the working dir
ADD . /docker-flask-test
# run pip to install the dependencies of the flask app
RUN pip install -r requirements.txt
# define the command to start the container
CMD ["python", "app.py"]
```

Build an image from the Dockerfile

Open the terminal and type this command to build an image from your Dockerfile: `docker build -t <image_name>:<tag>` .(note the period to indicate we're in our apps top level directory). For example: `docker build -t app:latest` .

```
PS D:\7th sem\Nalaiya Thiran\IMS> & "d:/7th sem\Nalaiya Thiran\IMS\env\Scripts\Activate.ps1"
(env) PS D:\7th sem\Nalaiya Thiran\IMS> deactivate
PS D:\7th sem\Nalaiya Thiran\IMS> docker build -t test .
[+] Building 191.5s (11/11) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 32B
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load metadata for docker.io/library/python:3.6
=> [auth] library/python:pull token for registry-1.docker.io
=> [1/5] FROM docker.io/library/python:3.6@sha256:f8652afaf88c25f0d22354d547d892591067aa4026a7fa9a6819df9f300af6fc
=> [internal] load build context
=> => transferring context: 124.72MB
=> CACHED [2/5] RUN pip install --upgrade pip
=> CACHED [3/5] WORKDIR /docker-flask-test
=> [4/5] ADD . /docker-flask-test
=> [5/5] RUN pip install -r requirements.txt
=> exporting image
=> exporting layers
=> => writing image sha256:1a6ba5ed14e9b63372ab1e57745faa6118aa8a9eb9c6e26d9704f2e4742531f4
=> => naming to docker.io/library/test
0.3s
0.1s
0.1s
0.1s
0.1s
3.9s
0.0s
0.0s
18.2s
18.1s
0.0s
0.0s
3.0s
157.3s
7.9s
7.6s
0.0s
0.0s
```

```
Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
PS D:\7th sem\Nalaiya Thiran\IMS>
```

Run your container locally and test

After you build your image successfully, type: `docker run -d -p 5000:5000 app`.

This command will create a container that contains all the application code and dependencies from the image and runs it locally.

```
PS D:\7th sem\Nalaiya Thiran\IMS> docker run -p 127.0.0.1:9696:8202 test
<ibm_db.IBM_DBConnection object at 0x7fea3a33a170>
connection successful...
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Running on all addresses.
  WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://172.17.0.2:8202/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 954-053-065
172.17.0.1 - - [19/Nov/2022 19:08:59] "GET / HTTP/1.1" 200 -
172.17.0.1 - - [19/Nov/2022 19:08:59] "GET /static/css/signup.css HTTP/1.1" 200 -
172.17.0.1 - - [19/Nov/2022 19:08:59] "GET /static/assets/signup1.png HTTP/1.1" 200 -
172.17.0.1 - - [19/Nov/2022 19:08:59] "GET /favicon.ico HTTP/1.1" 404 -
```

