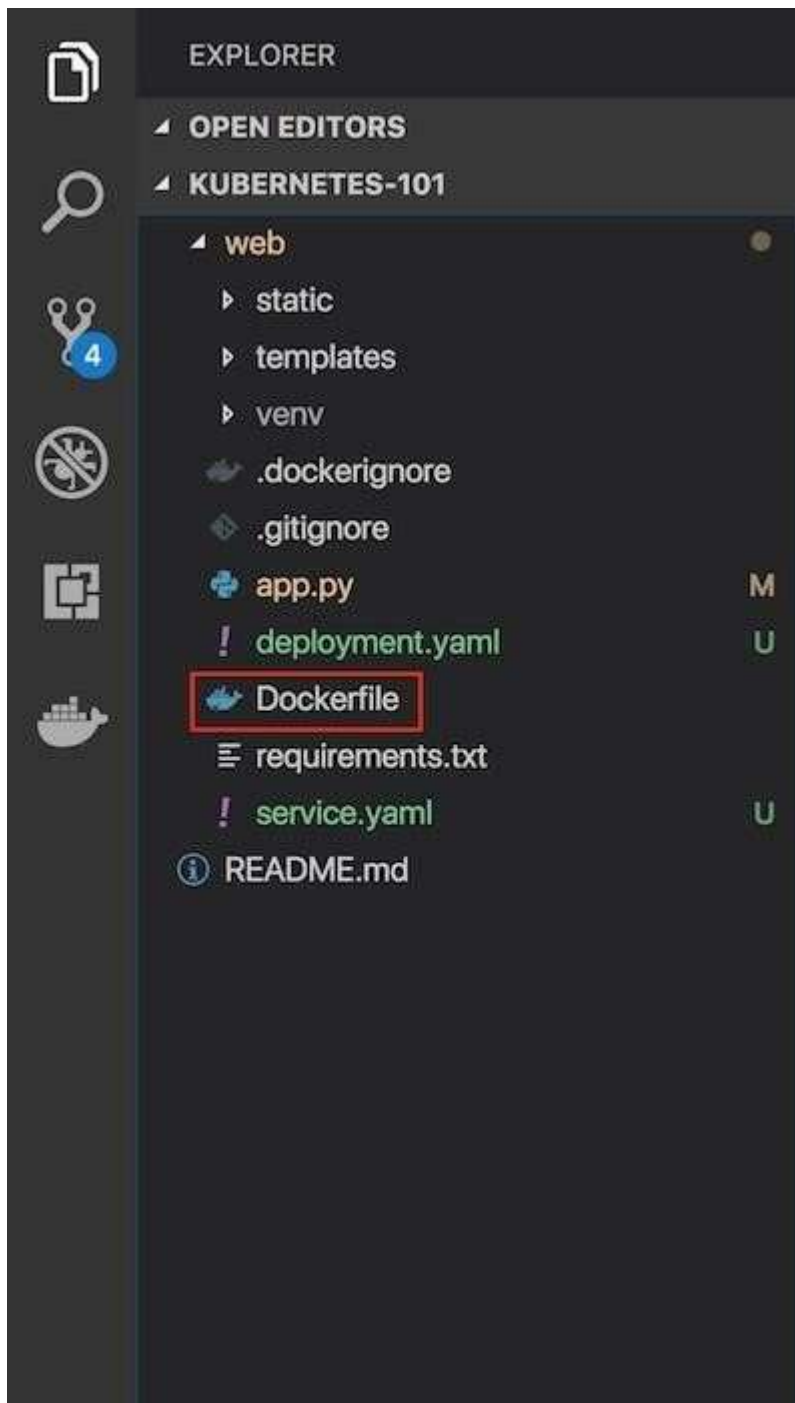


## CONTAINERIZE THE APP

<b>Team ID</b>	PNT2022TMID25039
<b>Project Name</b>	Skill / Job Recommender Application
<b>Date</b>	09 October 2022

Containerize your Flask application

- In your project directory, create a file named "Dockerfile." *Suggestion: Name your file exactly "Dockerfile," nothing else.*



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:

```
FROM python:2.7
LABEL maintainer="Greeshma,sgreeshma10@ibm.com"
```
- RUN apt-get update
- RUN mkdir /app WORKDIR /app COPY . /app
- RUN pip install -r requirements.txt

- EXPOSE 5000
- ENTRYPOINT [ "python" ]
- CMD [ "app.py" ]

Show more

## Explanation and breakdown of the above Dockerfile code

1.

The `FROM python:2.7`

first

part of the code above is:

2.

Show more

Because this Flask application uses Python 2.7, we want an environment that supports it and already has it installed. Fortunately, DockerHub has an official image that's installed on top of Ubuntu. In one line, we will have a base Ubuntu image with Python 2.7, virtualenv, and pip. There are tons of images on DockerHub, but if you would like to start off with a fresh Ubuntu image and build on top of it, you could do that.

3. Let's look at the next part of the code:
4. `LABEL maintainer="Greeshma sgreeshma10@ibm.com"`
5. `RUN apt-get update`

Show more

6. Note the maintainer and update the Ubuntu package index. The command is `RUN`, which is a function that runs the command after it.

7. `RUN mkdir /app`

8. `WORKDIR /app`

9. `COPY . /app`

Show more

10. Now it's time to add the Flask application to the image. For simplicity, copy the application under the `/app` directory on our Docker Image.

`WORKDIR` is essentially a **cd** in bash, and `COPY` copies a certain directory to the provided directory in an image. `ADD` is another command that does the same thing as `COPY`, but it also allows you to add a repository from a URL. Thus, if you want to clone your git repository instead of copying it from your local repository (for staging and production purposes), you can use that. `COPY`, however, should be used most of the time unless you have a URL.

11. Now that we have our repository copied to the image, we will install all of our dependencies,

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

which is defined in the `requirements.txt` part of the code.

12.

Show more

13.

We want

to expose the port(5000) the Flask application runs on, so we use `EXPOSE`.

14.

Show more

15. `ENTRYPOINT` specifies the entrypoint of your application.

16. `ENTRYPOINT [ "python" ]`

17. `CMD [ "app.py" ]`

Show more

## 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 .`

```
sun@sun-vm:~/sun@allhome$ docker build -t app:latest .
Sending build context to Docker daemon 348.2kB
Step 1/8 : FROM python:2.7
--> 6c7639d7cfe
Step 2/8 : LABEL maintainer="Guraj Nalhetra, sun@allhome.com"
--> Using cache
--> 48059d1291e
Step 3/8 : RUN apt-get update
--> Using cache
--> 6262d134d4e
Step 4/8 : COPY . /app
--> f809237800f
Step 5/8 : WORKDIR /app
Running intermediate container f80d893d2fe
--> 8b0c6f70c2e
Step 6/8 : RUN pip install -r requirements.txt
--> Running in 815304b0b07
Collecting click==6.7 (from -r requirements.txt (line 13))
  Downloading https://files.pythonhosted.org/packages/34/c1/80b6f5073d8d99c396c362c2f966f18209f6d75d0ff1a6fd700775a77/click-6.7-py2.py3-none-any.whl (71kB)
Collecting Flask==1.0.2 (from -r requirements.txt (line 22))
  Downloading https://files.pythonhosted.org/packages/7f/a0/4857674a8536d342b14d0c949638663680af82ac997302c8bdc4b/Flask-1.0.2-py2.py3-none-any.whl (55kB)
Collecting itsdangerous==0.24 (from -r requirements.txt (line 32))
  Downloading https://files.pythonhosted.org/packages/d0/b4/60862b045c08f608b807131c0f729c272bcefa1d62211051946294/itsdangerous-0.24.tar.gz (46kB)
Collecting Jinja2==2.10 (from -r requirements.txt (line 40))
  Downloading https://files.pythonhosted.org/packages/7f/76/a084e0cf05f7279f01a7e08d806763a4d77a78a1673209322b731751eja2-2.10-py2.py3-none-any.whl (128kB)
Collecting MarkupSafe==1.0 (from -r requirements.txt (line 52))
  Downloading https://files.pythonhosted.org/packages/0c/22/41a1a33648f6b68822e03701e77e4825e9c99d0e4f1d73b/MarkupSafe-1.0.tar.gz
Collecting Werkzeug==0.14.1 (from -r requirements.txt (line 62))
  Downloading https://files.pythonhosted.org/packages/20/c4/22a3e6473e5275ac3c47647bd118f3ef006829ef8d0ee04fe33243/Werkzeug-0.14.1-py2.py3-none-any.whl (329kB)
Building wheels for collected packages: itsdangerous, MarkupSafe
  Running setup.py bdist_wheel for itsdangerous: started
    Stored in directory: /root/.cache/pip/wheels/0c/4e/9d/5599611c155476d0529b94b08c726f31791837ac462f1fd
  Running setup.py bdist_wheel for MarkupSafe: started
    Stored in directory: /root/.cache/pip/wheels/23/56/20/a0e48dcd127f91c5e632140b18938f064670708f63e4e46
Successfully built itsdangerous MarkupSafe
Installing collected packages: click, itsdangerous, MarkupSafe, Jinja2, Werkzeug, Flask
Successfully installed Flask-1.0.2 Jinja2-2.10 MarkupSafe-1.0 Werkzeug-0.14.1 click-6.7 itsdangerous-0.24
Running intermediate container 815304b0b07
--> 0d0336d7bc
Step 7/8 : ENTRYPOINT [ "python" ]
--> Running in b0c3c8813d1
Running intermediate container b0c3c8813d1
--> 720efc38d1c
Step 8/8 : CMD [ "app.py" ]
--> Running in a744030d8f
Running intermediate container a744030d8f
--> d80bd8763d5
Successfully built app:latest
Successfully tagged app:latest
sun@sun-vm:~/sun@allhome$
```

## 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.

