WORKING WITH DOCKER

INSTALL DOCKER

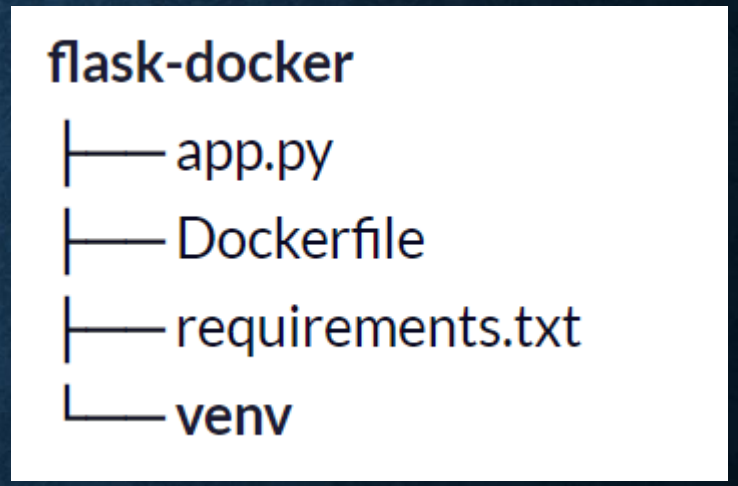
- Docker is available for Windows, Mac and Linux
- For the Ubuntu Version please follow the instructions at

  https://docs.docker.com/engine/install/ubuntu/
- Then, verify your installation by running the hello world image

```
sudo docker run hello-world
```

SET UP YOUR PROJECT

- Create a directory for your project
- Create a venv in that directory
- Activate the venv
- Install flask on the venv
- Create your app.py, requirements.txt, and Dockerfile
- Your Directory structure should resemble the one on the right



GET YOUR DEPENDENCIES AND RESOURCE FILES

- Consider the file “app.py” to be the main file of your flask application
- Organize the rest of your resources, like the templates folder, as required for the application.
- For this step, ignore the existence of Docker

SET UP THE DOCKERFILE

- The Dockerfile is a configuration file that is used by the Docer Builder to build the container.
- The first line tells the Docker Builder they Docker syntax standard used for this particular Dockerifle

```
# syntax=docker/dockerfile:1
```

- The next line is a base image. We can create our mage layers from scratch, but it is very useful to inherit from exiting images. We can start from the OS layer if required, but since we're going to launch our container in a VM, we can just use its native OS. So, we start with Python.

```
FROM python:3.8-slim-buster
```

SETTING UP THE DOCKERFILE

- Specify the working directory for the application inside the container. This helps to keep things organized.

```
WORKDIR /name
```

- The next 2 lines tell the Docker builder to first copy the requirements.txt file into the container image and then use pip to install the libraries in the requirements file if they aren't already installed.

```
COPY requirements.txt requirements.txt
RUN pip3 install -r requirements.txt
```

SETTING UP THE DOCKERFILE

- Next, we copy over all of the remaining files into the image

```
COPY . .
```

- Now, we have all of the working directory files in the image. Finally, we tell Docker how to start up. Since we're running a Flask app using python, we mention we need python3 and flask. The –m is to run it as a module. We also pass the host port.

```
CMD [ "python3", "-m" , "flask", "run", "--host=0.0.0.0"]
```

BUILDING THE IMAGE

- Now, we're ready to build and launch our container.
- Setup the venv and install everything required. Then, deactivate it.
- Build the container using

```
docker build --tag name .
```

- If this were successful, we should see our image when we run

```
docker images
```

- Now, we can run our container with

```
docker run name
```