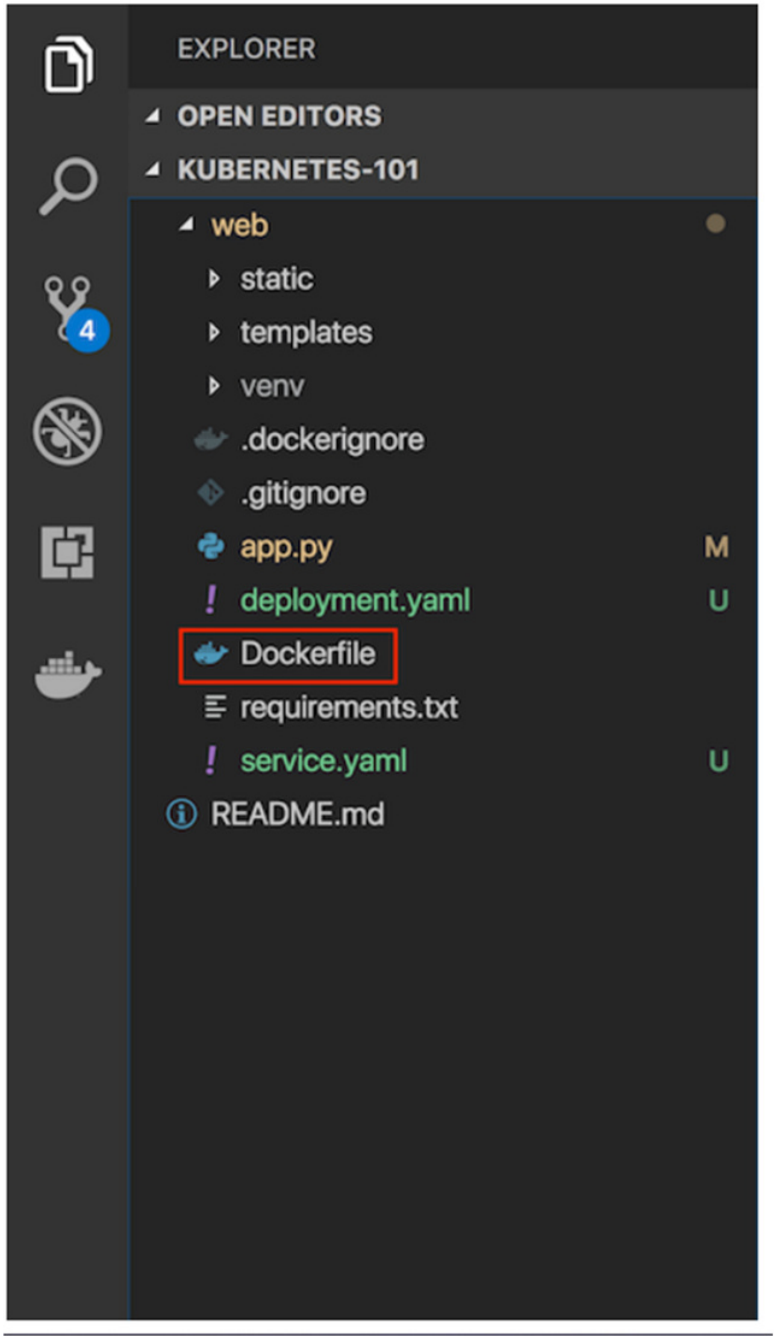
DEPLOYMENT OF APP IN IBM CLOUD

CONTAINERIZE YOUR FLASK APPLICATION



A "Docker file" 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="Kunal Malhotra, kunal.malhotra1@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" ]
```

Explanation and breakdown of the above Docker file code

1.The first part of the code above is:

```
FROM python:2.7
```

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.

2.Let's look at the next part of the code:

```
LABEL maintainer="Kunal Malhotra, kunal.malhotra1@ibm.com"
RUN apt-get update
```

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

```
RUN mkdir /app
 WORKDIR /app
COPY . /app
```

4.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.

5.Now that we have our repository copied to the image, we will install all of our dependencies, which is defined in the requirements.txt part of the code.

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

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

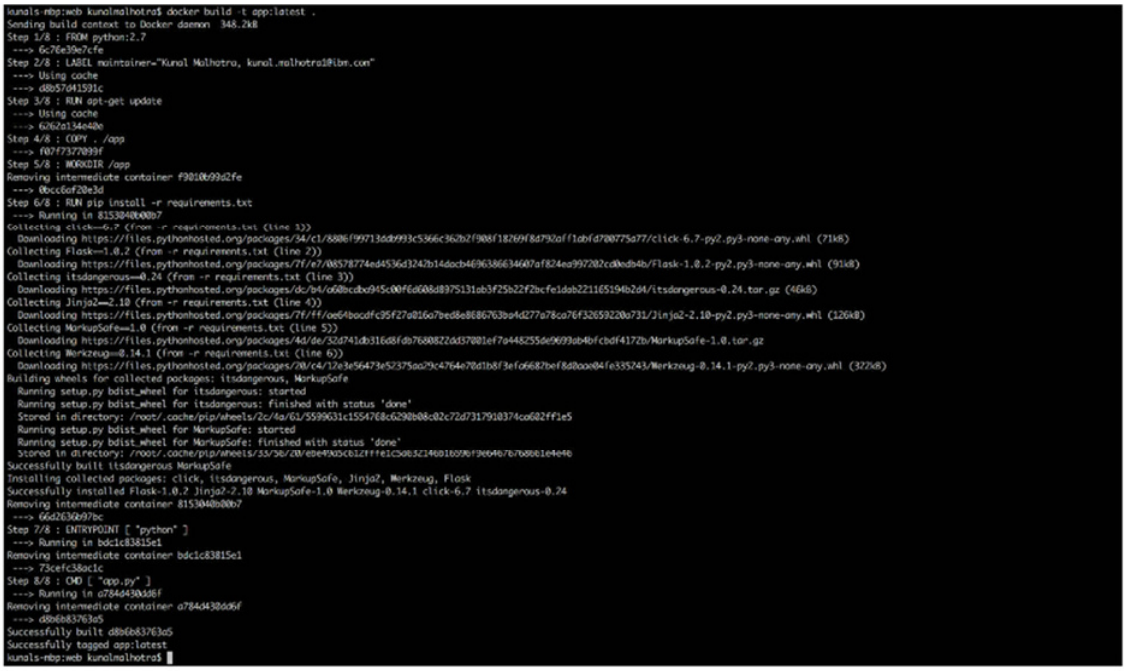
```
EXPOSE 5000
```

7.ENTRYPOINT specifies the entry point of your application.

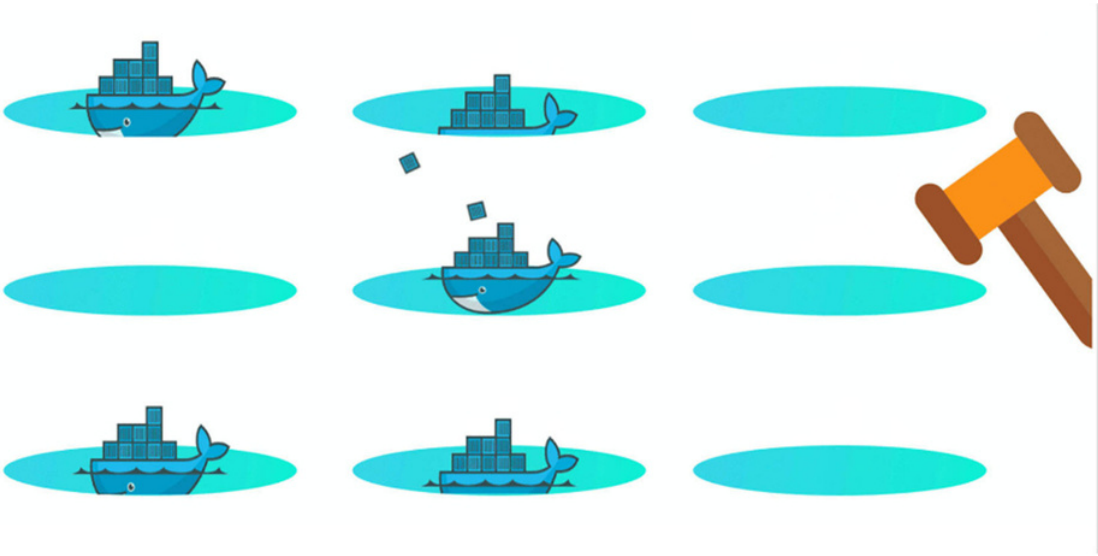
```
ENTRYPOINT [ "python" ]
CMD [ "app.py" ]
```

Build an image from the Docker file

Open the terminal and type this command to build an image from your Docker file: 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 **.**



Run your container locally and test

After you build your image succesfully, 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.