Assignment 1

1. I am already using ubuntu as my base operating system, so There is no need to setup a virtual env
2. Install VScode using SNAP

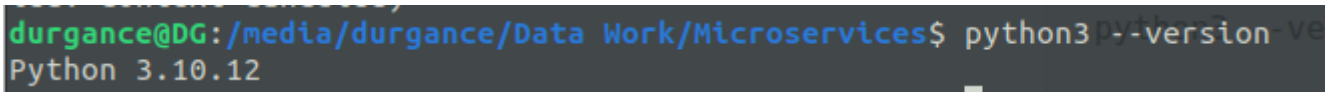
```
sudo snap install code --classic
```

3. Installing Python

```
sudo apt update
sudo apt install python3
```

4. Verifying Installation

```
python3 --version
```

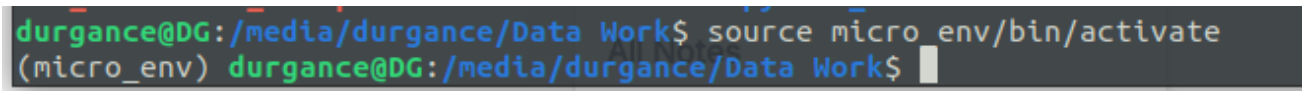


5. Clone the repo

```
git clone git@github.com:Vikas098766/Microservices.git\
```

6. create a virtualenv

```
sudo pip3 install virtualenv
virtualenv micro_env
source micro_env/bin/activate
```



7. Installing dependencies from requirements.txt file

```
pip install -r requirements.txt
```

8. Training and saving the model

```
python code_model_training/train.py
```

```
(micro_env) durgance@DG:/media/durgance/Data Work/Microservices$ python code_mod
el_training/train.py
Accuracy: 0.9736842105263158
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay object at 0x7fa0b
0e646a0>
/media/durgance/Data Work/Microservices/code_model_training/train.py:54: UserWar
ning: Matplotlib is currently using agg, which is a non-GUI backend, so cannot s
how the figure.
  plt.show()
```

9. Running and Testing the Flask App

```
python app.py
```

```
(micro_env) durgance@DG:/media/durgance/Data Work/Microservices$ python app.py
 * Serving Flask app 'ms' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployme
nt.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on all addresses.
   WARNING: This is a development server. Do not use it in a production deployme
nt.
 * Running on http://192.168.1.10:5000/ (Press CTRL+C to quit)
```

Testing the Flask App // Command to test the App

```
curl -X GET http://localhost:5000/info
curl -X GET http://localhost:5000/health
```

```
durgance@DG:/media/durgance/Data Work/Microservices/tests$ curl -X GET http://localhost:5000/info
{"name":"Breast Cancer Wisconsin (Diagnostic)","version":"v1.0.0"}
durgance@DG:/media/durgance/Data Work/Microservices/tests$ curl -X GET http://localhost:5000/health
okdurgance@DG:/media/durgance/Data Work/Microservices/tests$
```

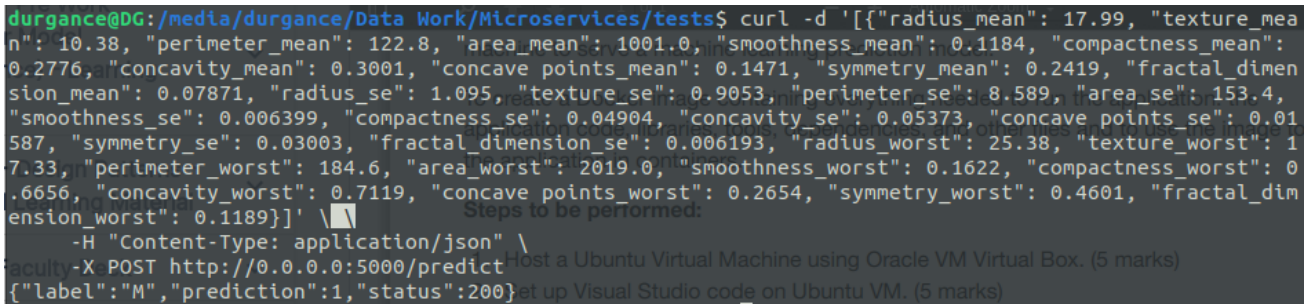
10. Testing the Application to make predictions using example calls

Command :

```
curl -d '[{"radius_mean": 17.99, "texture_mean": 10.38, "perimeter_mean":
122.8, "area_mean": 1001.0, "smoothness_mean": 0.1184, "compactness_mean":
0.2776, "concavity_mean": 0.3001, "concave points_mean": 0.1471,
"symmetry_mean": 0.2419, "fractal_dimension_mean": 0.07871, "radius_se":
1.095, "texture_se": 0.9053, "perimeter_se": 8.589, "area_se": 153.4,
"smoothness_se": 0.006399, "compactness_se": 0.04904, "concavity_se":
0.05373, "concave points_se": 0.01587, "symmetry_se": 0.03003,
"fractal_dimension_se": 0.006193, "radius_worst": 25.38, "texture_worst":
17.33, "perimeter_worst": 184.6, "area_worst": 2019.0, "smoothness_worst":
0.1622, "compactness_worst": 0.6656, "concavity_worst": 0.7119, "concave
points_worst": 0.2654, "symmetry_worst": 0.4601, "fractal_dimension_worst":
0.1189}]' \
     -H "Content-Type: application/json" \
     -X POST http://0.0.0.0:5000/predict
```

Result :

```
{"label":"M","prediction":1,"status":200}
```



11. Create a docker image containing everything needed to run the application

```
FROM ubuntu:latest

WORKDIR /app

# Fix certificate issues
RUN apt-get update

COPY requirements.txt ./requirements.txt

RUN apt-get update && DEBIAN_FRONTEND=noninteractive \
    apt-get install -y python3.10 python3-pip

RUN curl -sSL https://install.python-poetry.org | python3 - --preview
RUN pip3 install --upgrade requests
RUN ln -fs /usr/bin/python3 /usr/bin/python

RUN pip3 install --upgrade pip

RUN pip3 install -r requirements.txt

COPY . .

EXPOSE 8501

RUN python3 ./code_model_training/train.py

ENTRYPOINT ["python3","app.py"]
```

Here is the dockerfile commands, code to create a docker image

```
sudo docker build -t micro_doc .
// Command to run the image
sudo docker run -dp 127.0.0.1:8501:8501 micro_doc
//Command to check docker status
sudo docker ps
// Command to get logs
sudo docker logs {docker_id}
```

By the above commands, the flask container is created and can be run, Check out the link for the running server

12. All the commands can be testing , as per shown here