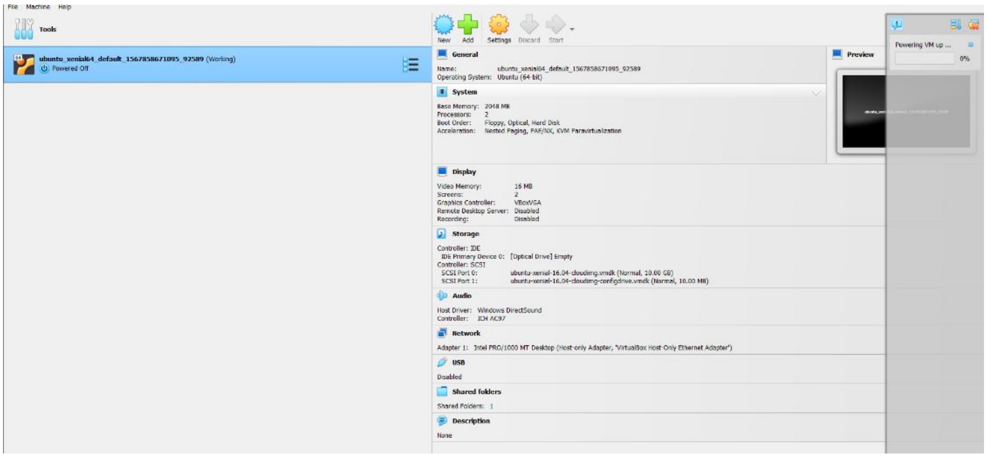
Steps Performed:-

Host a Ubuntu Virtual Machine using Oracle VM Virtual Box. (5 marks)--



ubuntu_xenial64_default_1567858671095_92589 (Working)
Powered Off

General
Name: ubuntu_xenial64_default_1567858671095_92589
Operating System: Ubuntu (64-bit)

System
Base Memory: 2048 MB
Processors: 2
Boot Order: Floppy, Optical, Hard Disk
Acceleration: Nested Paging, PAE/NX, KVM Paravirtualization

Display
Video Memory: 16 MB
Screens: 2
Graphics Controller: VBoxVGA
Remote Desktop Server: Disabled
Recording: Disabled

Storage
Controller: IDE
IDE Primary Device 0: [Optical Drive] Empty
Controller: SCSI
SCSI Port 0: ubuntu-xenial-16.04-cloudimg.vmdk (Normal, 10.00 GB)
SCSI Port 1: ubuntu-xenial-16.04-cloudimg-configdrive.vmdk (Normal, 10.00 MB)

Audio
Host Driver: Windows DirectSound
Controller: ICH AC97

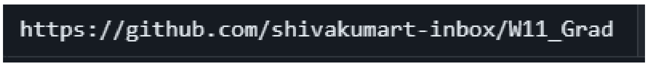
Network
Adapter 1: Intel PRO/1000 MT Desktop (Host-only Adapter, 'VirtualBox Host-Only Ethernet Adapter')

USB
Disabled

Shared folders
Shared Folders: 1

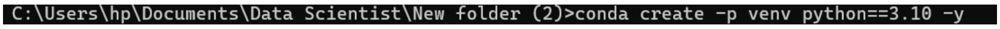
Description
None

Cloned the GitHub repository using the link and using command git clone—



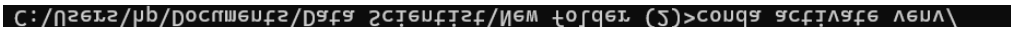
```
https://github.com/shivakumart-inbox/W11_Grad
```

Set up Python and created the virtual environment. (5 marks) ---



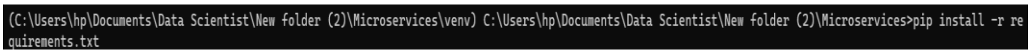
```
C:\Users\hp\Documents\Data Scientist\New folder (2)>conda create -p venv python==3.10 -y
```

Activate the environment--



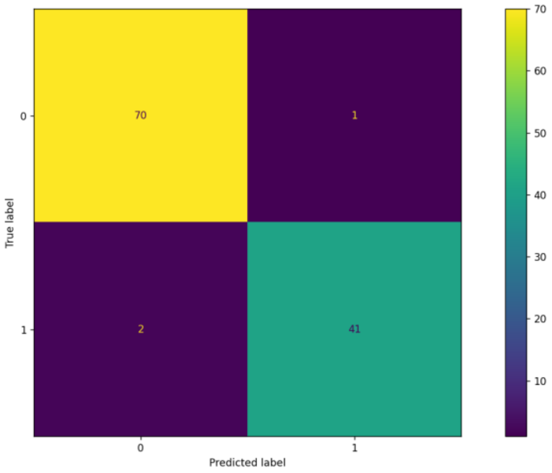
```
C:/Users/hp/Documents/Data Scientist/New folder (2)>conda activate venv\
```

Install the dependencies from requirements.txt file. (1 mark) ---

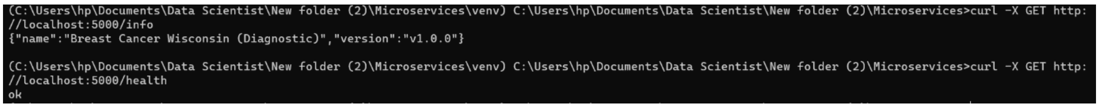


```
(C:\Users\hp\Documents\Data Scientist\New folder (2)\Microservices\venv) C:\Users\hp\Documents\Data Scientist\New folder (2)\Microservices>pip install -r re
quirements.txt
```

Train and save the model. (2 marks) ---



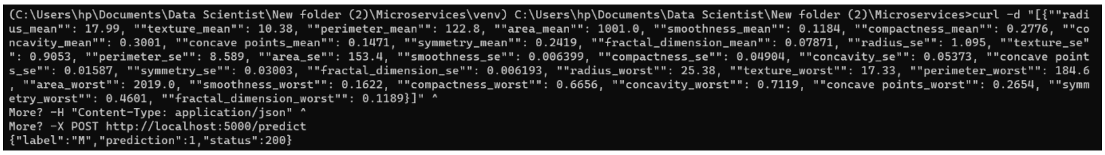
Test the Flask web application using python app.py . (5 marks)



```
(C:\Users\hp\Documents\Data Scientist\New folder (2)\Microservices\venv) C:\Users\hp\Documents\Data Scientist\New folder (2)\Microservices>curl -X GET http:
//localhost:5000/info
{"name":"Breast Cancer Wisconsin (Diagnostic)","version":"v1.0.0"}

(C:\Users\hp\Documents\Data Scientist\New folder (2)\Microservices\venv) C:\Users\hp\Documents\Data Scientist\New folder (2)\Microservices>curl -X GET http:
//localhost:5000/health
ok
```

Test the application and make predictions using the example calls available in the folder /tests. (5 marks) ---

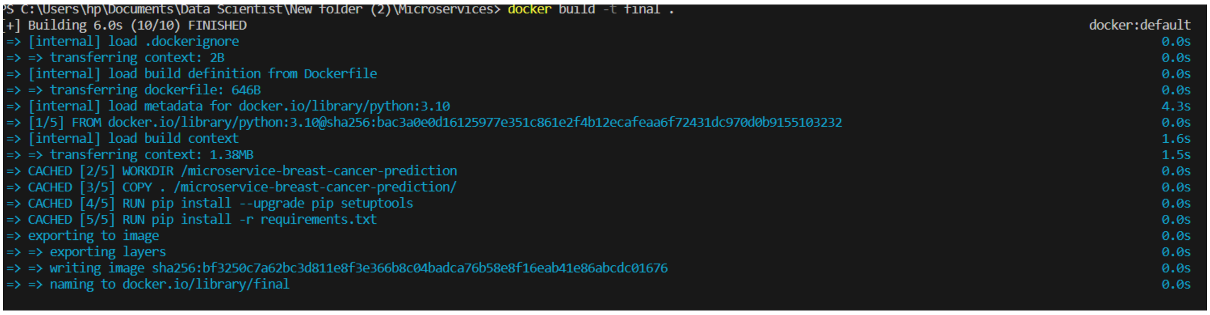


```
(C:\Users\hp\Documents\Data Scientist\New folder (2)\Microservices\venv) C:\Users\hp\Documents\Data Scientist\New folder (2)\Microservices>curl -d "[{""radi
us_mean"": 17.99, ""texture_mean"": 10.38, ""perimeter_mean"": 122.8, ""area_mean"": 1001.0, ""smoothness_mean"": 0.1184, ""compactness_mean"": 0.2776, ""co
ncavity_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 point
s_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, ""symm
etry_worst"": 0.4601, ""fractal_dimension_worst"": 0.1189}]" ^
More? -H "Content-Type: application/json" ^
More? -X POST http://localhost:5000/predict
{"label":"M","prediction":1,"status":200}
```

Create a docker image containing everything needed to run the application.(10 marks)---

Using--

Docker build -t final .



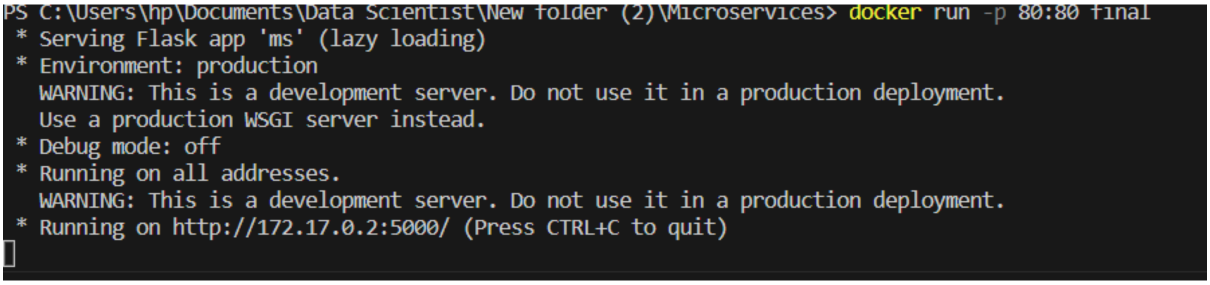
```
PS C:\Users\hp\Documents\Data Scientist\New folder (2)\Microservices> docker build -t final .
[+] Building 6.0s (10/10) FINISHED                                                                                        docker:default
=> [internal] load .dockerignore                                                                                                    0.0s
=> => transferring context: 2B                                                                                                      0.0s
=> [internal] load build definition from Dockerfile                                                                                 0.0s
=> => transferring dockerfile: 646B                                                                                                 0.0s
=> [internal] load metadata for docker.io/library/python:3.10                                                                       4.3s
=> [1/5] FROM docker.io/library/python:3.10@sha256:bac3a0e0d16125977e351c861e2f4b12ecafeaa6f72431dc970d0b9155103232                 0.0s
=> [internal] load build context                                                                                                    1.6s
=> => transferring context: 1.38MB                                                                                                  1.5s
=> CACHED [2/5] WORKDIR /microservice-breast-cancer-prediction                                                                      0.0s
=> CACHED [3/5] COPY . /microservice-breast-cancer-prediction/                                                                      0.0s
=> CACHED [4/5] RUN pip install --upgrade pip setuptools                                                                            0.0s
=> CACHED [5/5] RUN pip install -r requirements.txt                                                                                 0.0s
=> exporting to image                                                                                                               0.0s
=> => exporting layers                                                                                                              0.0s
=> => writing image sha256:bf3250c7a62bc3d811e8f3e366b8c04badca76b58e8f16eab41e86abcdc01676                                         0.0s
=> => naming to docker.io/library/final                                                                                             0.0s
```

Run the containerized application as a prediction service and test it locally by passing some example calls and get the prediction. (10 marks) ---

Now the image built successfully run the container—

```
docker run -d -p 5000:5000 image
```

Test the app--



```
PS C:\Users\hp\Documents\Data Scientist\New folder (2)\Microservices> docker run -p 80:80 final
 * Serving Flask app 'ms' (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: off
 * Running on all addresses.
   WARNING: This is a development server. Do not use it in a production deployment.
 * Running on http://172.17.0.2:5000/ (Press CTRL+C to quit)
```

****