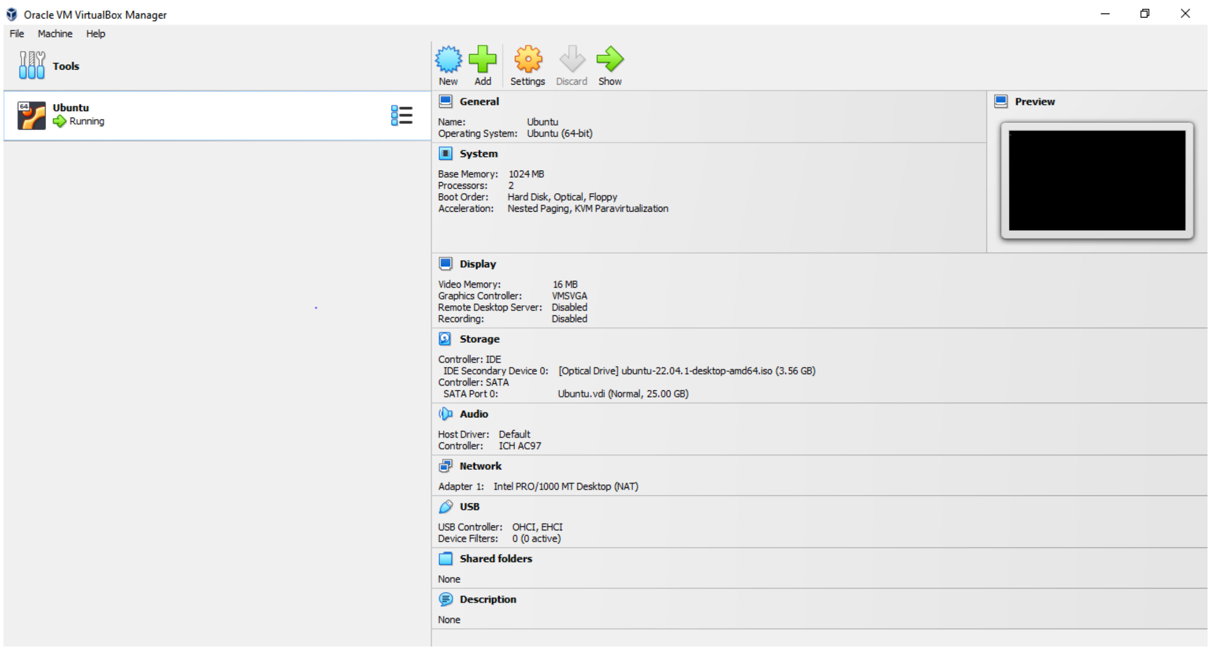
Week 10 - Week 12: Graded Assignment

Objective: Implementing a microservice using the Python Flask framework on an Ubuntu virtual machine to serve a machine learning prediction model.

STEP 1: Host an Ubuntu Virtual Machine using Oracle VM Virtual Box.



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

System
Base Memory: 1024 MB
Processors: 2
Boot Order: Hard Disk, Optical, Floppy
Acceleration: Nested Paging, KVM Paravirtualization

Display
Video Memory: 16 MB
Graphics Controller: VMSVGA
Remote Desktop Server: Disabled
Recording: Disabled

Storage
Controller: IDE
IDE Secondary Device 0: [Optical Drive] ubuntu-22.04.1-desktop-amd64.iso (3.56 GB)
Controller: SATA
SATA Port 0: Ubuntu.vdi (Normal, 25.00 GB)

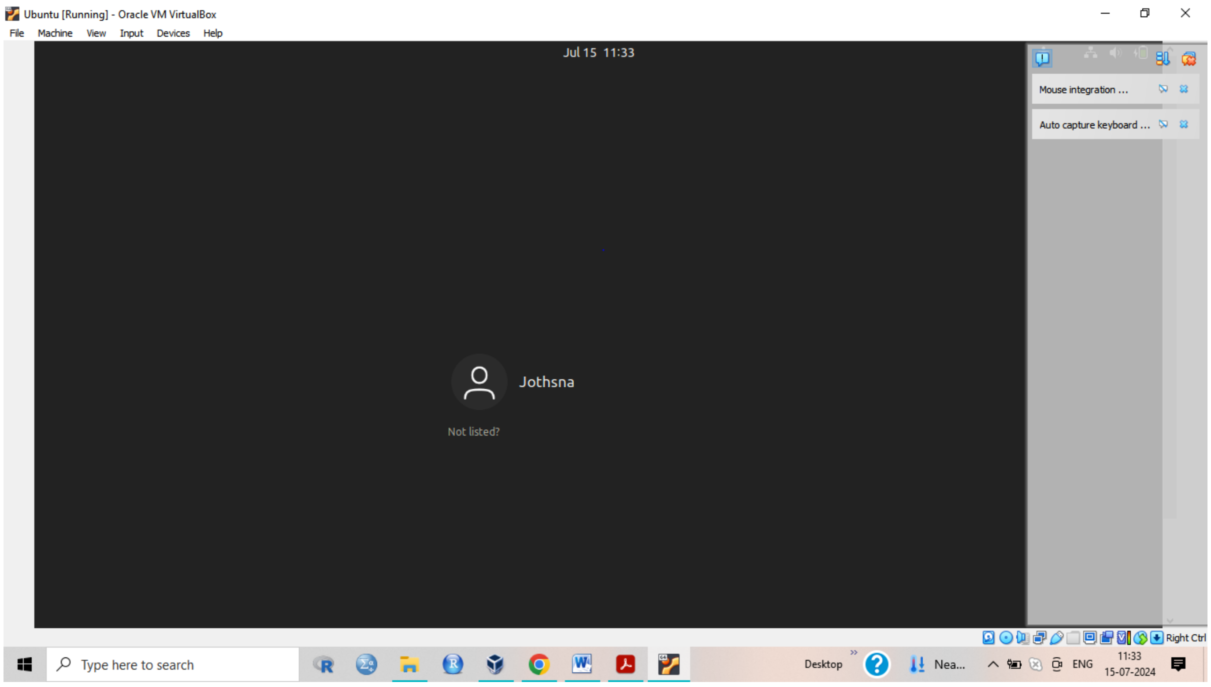
Audio
Host Driver: Default
Controller: ICH AC97

Network
Adapter 1: Intel PRO/1000 MT Desktop (NAT)

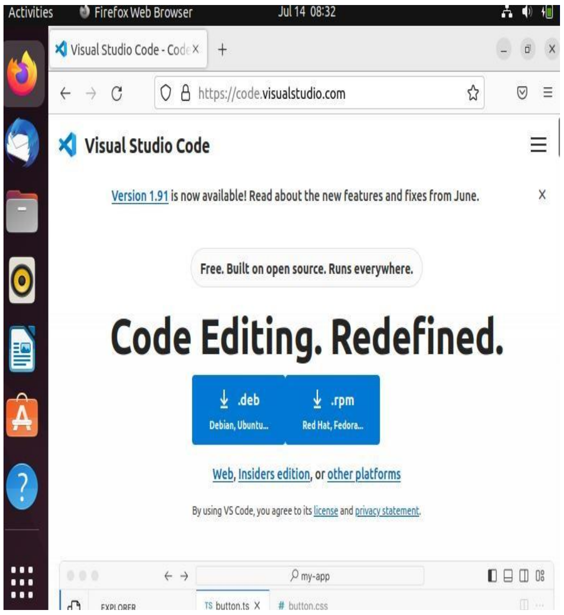
USB
USB Controller: OHCI, EHCI
Device Filters: 0 (0 active)

Shared folders
None

Description
None



STEP 2: Set up Visual Studio code on Ubuntu VM.



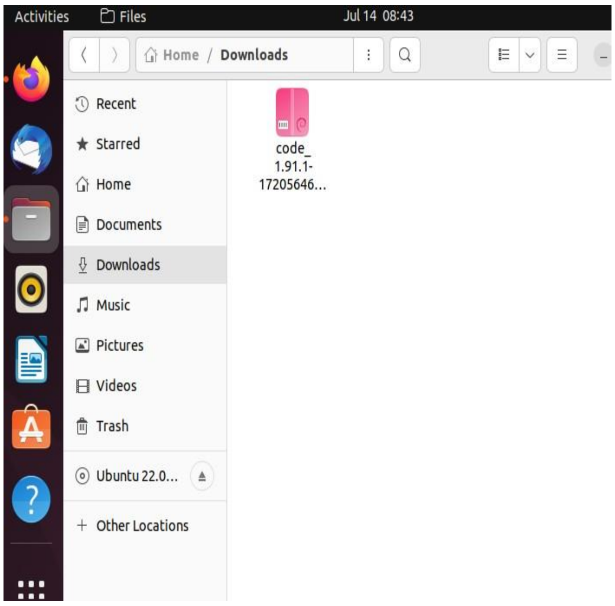
Version 1.91 is now available! Read about the new features and fixes from June.

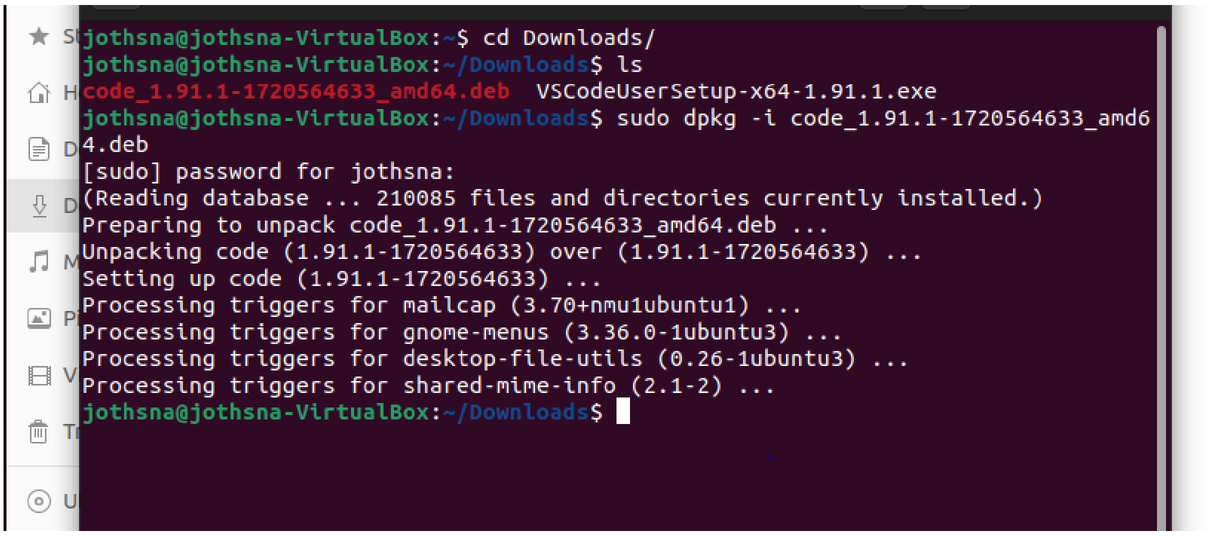
Free. Built on open source. Runs everywhere.

Code Editing. Redefined.

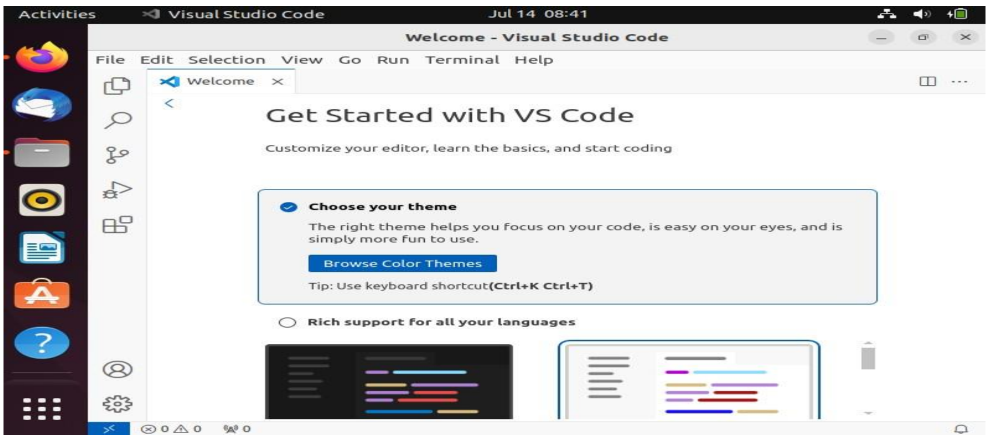
Web, Insiders edition, or other platforms

By using VS Code, you agree to its license and privacy statement.





```
jothsna@jothsna-VirtualBox:~$ cd Downloads/
jothsna@jothsna-VirtualBox:~/Downloads$ ls
code_1.91.1-1720564633_amd64.deb  VSCodeUserSetup-x64-1.91.1.exe
jothsna@jothsna-VirtualBox:~/Downloads$ sudo dpkg -i code_1.91.1-1720564633_amd6
4.deb
[sudo] password for jothsna:
(Reading database ... 210085 files and directories currently installed.)
Preparing to unpack code_1.91.1-1720564633_amd64.deb ...
Unpacking code (1.91.1-1720564633) over (1.91.1-1720564633) ...
Setting up code (1.91.1-1720564633) ...
Processing triggers for mailcap (3.70+nmu1ubuntu1) ...
Processing triggers for gnome-menus (3.36.0-1ubuntu3) ...
Processing triggers for desktop-file-utils (0.26-1ubuntu3) ...
Processing triggers for shared-mime-info (2.1-2) ...
jothsna@jothsna-VirtualBox:~/Downloads$ 
```



Get Started with VS Code

Customize your editor, learn the basics, and start coding

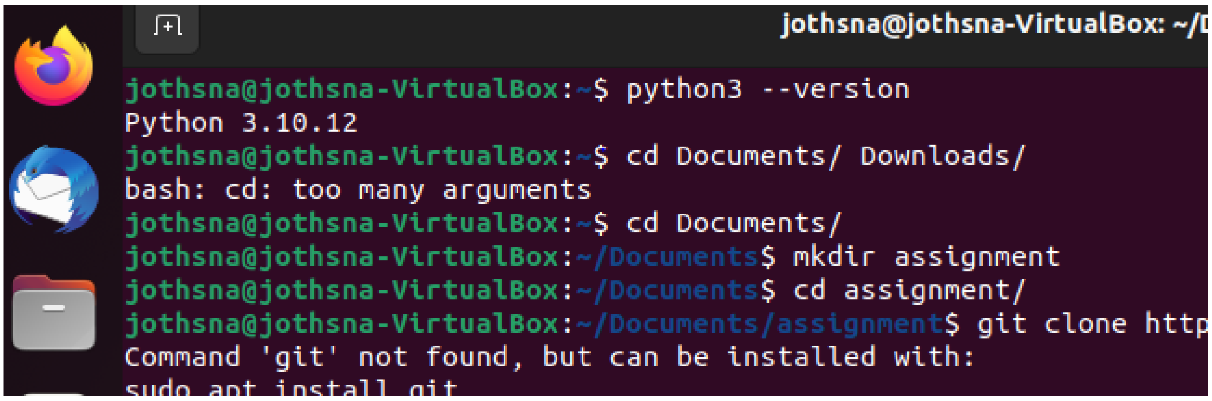
Choose your theme

The right theme helps you focus on your code, is easy on your eyes, and is simply more fun to use.

Tip: Use keyboard shortcut(Ctrl+K Ctrl+T)

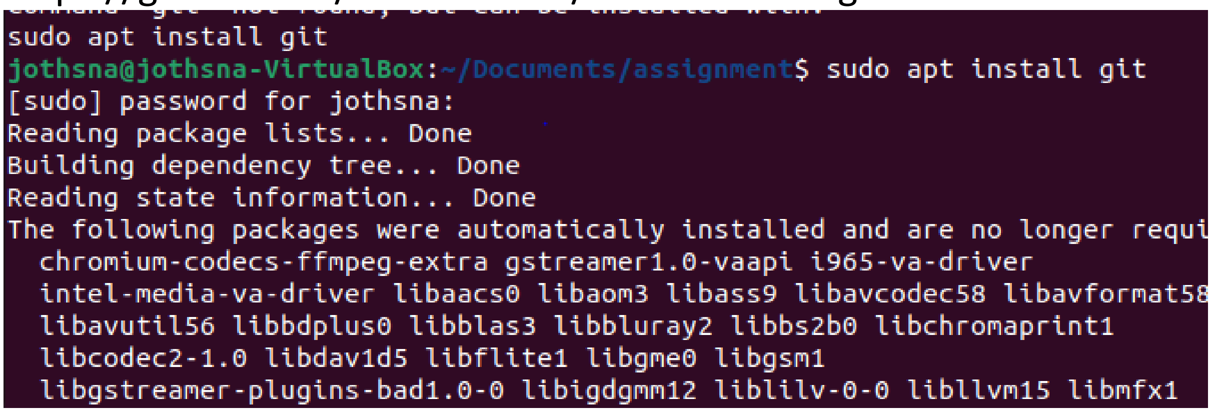
Rich support for all your languages

STEP 3: Set up Python.



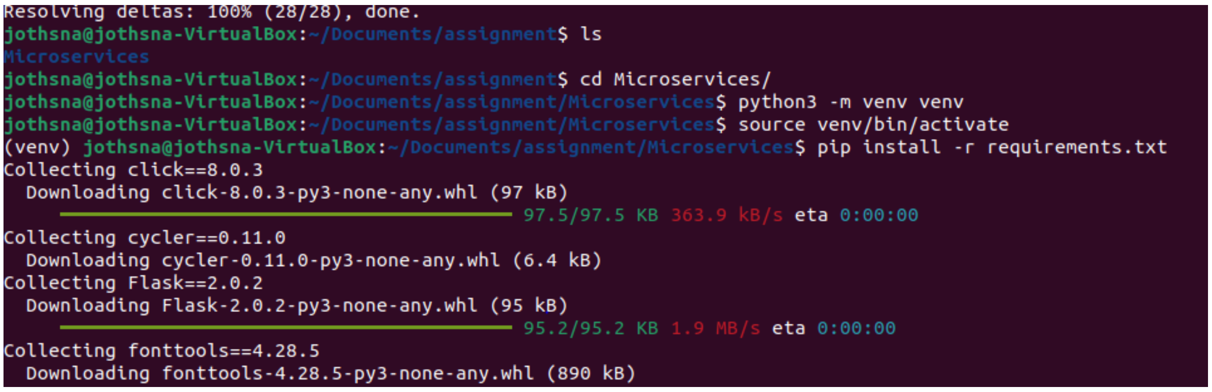
```
jothsna@jothsna-VirtualBox:~$ python3 --version
Python 3.10.12
jothsna@jothsna-VirtualBox:~$ cd Documents/ Downloads/
bash: cd: too many arguments
jothsna@jothsna-VirtualBox:~$ cd Documents/
jothsna@jothsna-VirtualBox:~/Documents$ mkdir assignment
jothsna@jothsna-VirtualBox:~/Documents$ cd assignment/
jothsna@jothsna-VirtualBox:~/Documents/assignment$ git clone http
Command 'git' not found, but can be installed with:
sudo apt install git
```

STEP 4: Clone this Github repository - https://github.com/Vikas098766/Microservices.git

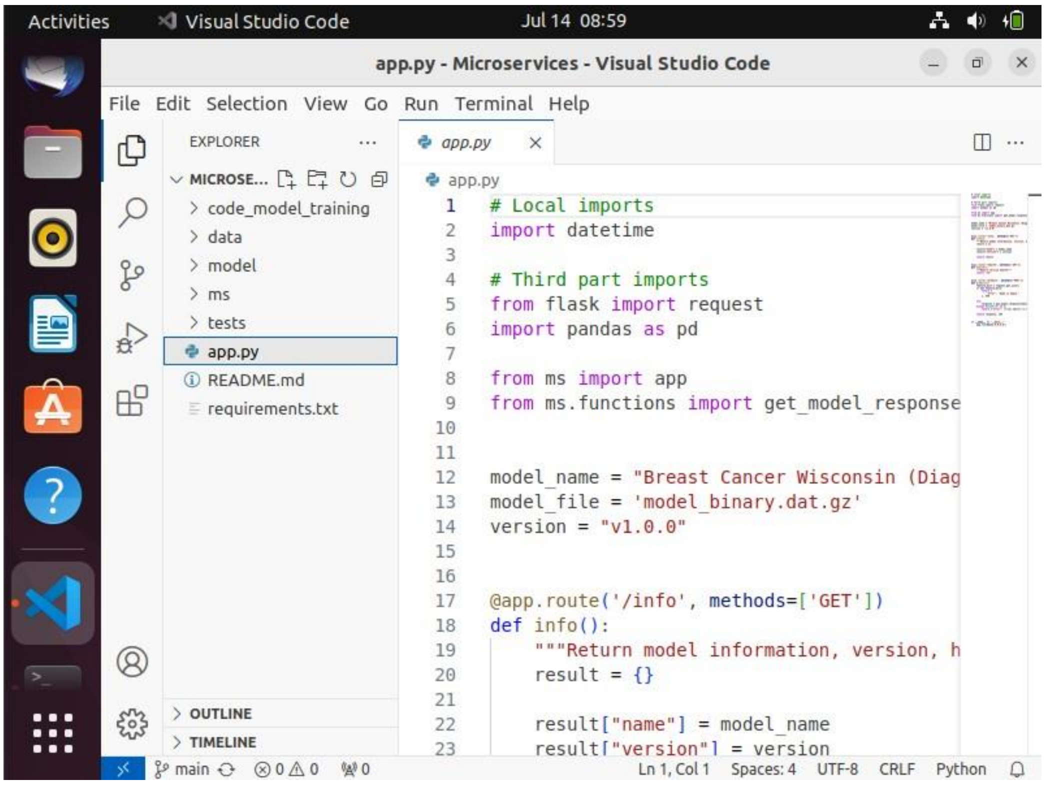


```
sudo apt install git
jothsna@jothsna-VirtualBox:~/Documents/assignment$ sudo apt install git
[sudo] password for jothsna:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer requi
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi i965-va-driver
  intel-media-va-driver libaacs0 libaom3 libass9 libavcodec58 libavformat58
  libavutil56 libbdplus0 libblas3 libbluray2 libbs2b0 libchromaprint1
  libcodec2-1.0 libdav1d5 libflite1 libgme0 libgsm1
  libgstreamer-plugins-bad1.0-0 libigdgmm12 liblilv-0-0 libllvm15 libmfx1
```

STEP 5: Create a Virtual Environment.



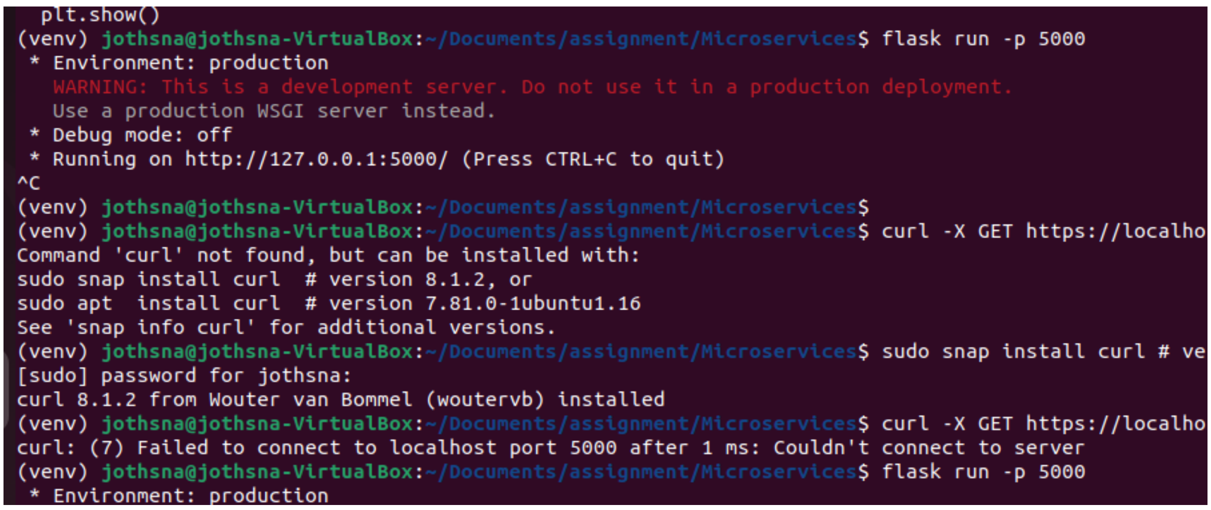
```
Resolving deltas: 100% (28/28), done.
jothsna@jothsna-VirtualBox:~/Documents/assignment$ ls
Microservices
jothsna@jothsna-VirtualBox:~/Documents/assignment$ cd Microservices/
jothsna@jothsna-VirtualBox:~/Documents/assignment/Microservices$ python3 -m venv venv
jothsna@jothsna-VirtualBox:~/Documents/assignment/Microservices$ source venv/bin/activate
(venv) jothsna@jothsna-VirtualBox:~/Documents/assignment/Microservices$ pip install -r requirements.txt
Collecting click==8.0.3
  Downloading click-8.0.3-py3-none-any.whl (97 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 97.5/97.5 KB 363.9 kB/s eta 0:00:00
Collecting cycler==0.11.0
  Downloading cycler-0.11.0-py3-none-any.whl (6.4 kB)
Collecting Flask==2.0.2
  Downloading Flask-2.0.2-py3-none-any.whl (95 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 95.2/95.2 KB 1.9 MB/s eta 0:00:00
Collecting fonttools==4.28.5
  Downloading fonttools-4.28.5-py3-none-any.whl (890 kB)
```



```python
# Local imports
import datetime

# Third part imports
from flask import request
import pandas as pd

from ms import app
from ms.functions import get_model_response


model_name = "Breast Cancer Wisconsin (Diag
model_file = 'model_binary.dat.gz'
version = "v1.0.0"


@app.route('/info', methods=['GET'])
def info():
    """Return model information, version, h
    result = {}

    result["name"] = model_name
    result["version"] = version
```

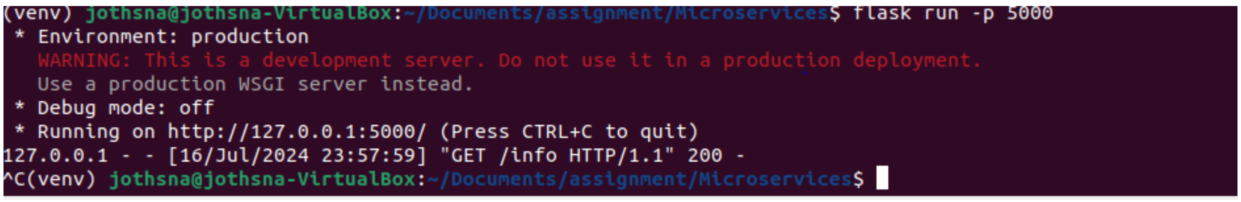
STEP 6: Install the dependencies from requirements.txt file.

STEP 7: Train and save the model



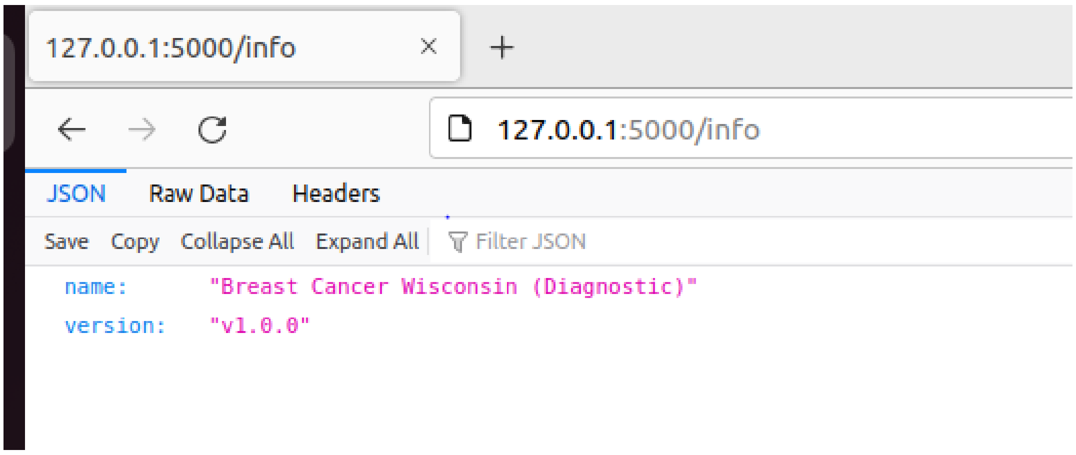
```
  plt.show()
(venv) jothsna@jothsna-VirtualBox:~/Documents/assignment/Microservices$ flask run -p 5000
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
^C
(venv) jothsna@jothsna-VirtualBox:~/Documents/assignment/Microservices$
(venv) jothsna@jothsna-VirtualBox:~/Documents/assignment/Microservices$ curl -X GET https://localho
Command 'curl' not found, but can be installed with:
sudo snap install curl  # version 8.1.2, or
sudo apt  install curl  # version 7.81.0-1ubuntu1.16
See 'snap info curl' for additional versions.
(venv) jothsna@jothsna-VirtualBox:~/Documents/assignment/Microservices$ sudo snap install curl # ve
[sudo] password for jothsna:
curl 8.1.2 from Wouter van Bommel (woutervb) installed
(venv) jothsna@jothsna-VirtualBox:~/Documents/assignment/Microservices$ curl -X GET https://localho
curl: (7) Failed to connect to localhost port 5000 after 1 ms: Couldn't connect to server
(venv) jothsna@jothsna-VirtualBox:~/Documents/assignment/Microservices$ flask run -p 5000
 * Environment: production
```

STEP 8: Test the Flask web application.

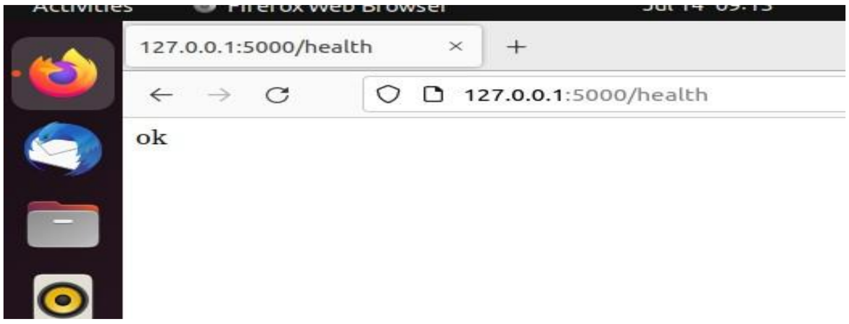


```
(venv) jothsna@jothsna-VirtualBox:~/Documents/assignment/Microservices$ flask run -p 5000
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [16/Jul/2024 23:57:59] "GET /info HTTP/1.1" 200 -
^C(venv) jothsna@jothsna-VirtualBox:~/Documents/assignment/Microservices$
```

STEP 9: Test the application and make predictions using the example calls available in the folder /tests.

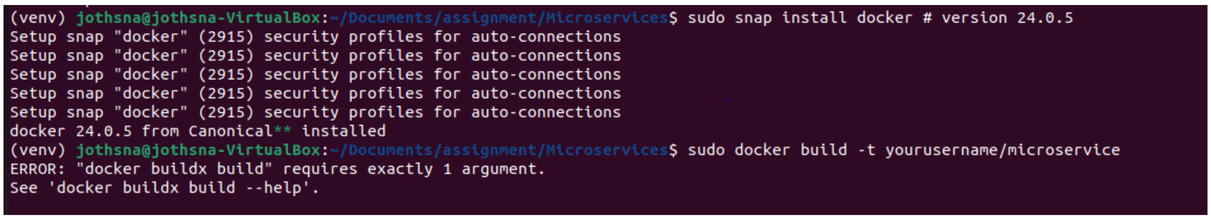


```
name:      "Breast Cancer Wisconsin (Diagnostic)"
version:   "v1.0.0"
```

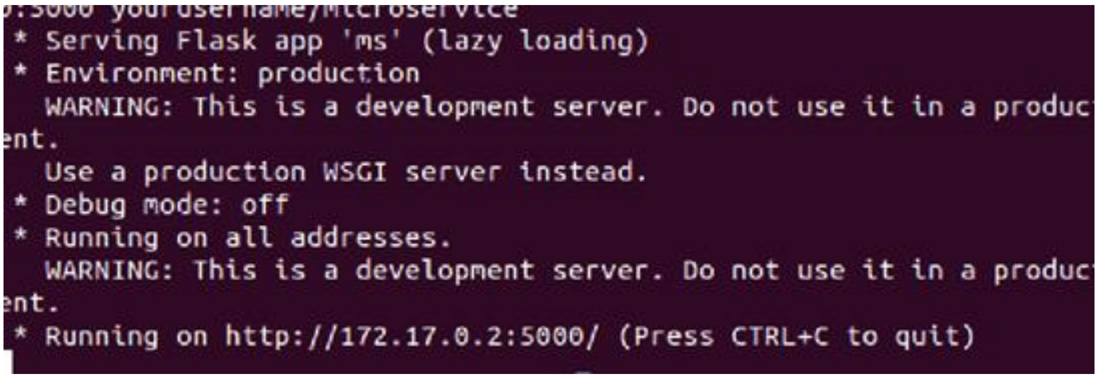


```
ok
```

STEP 10: Create a docker image containing everything needed to run the application.

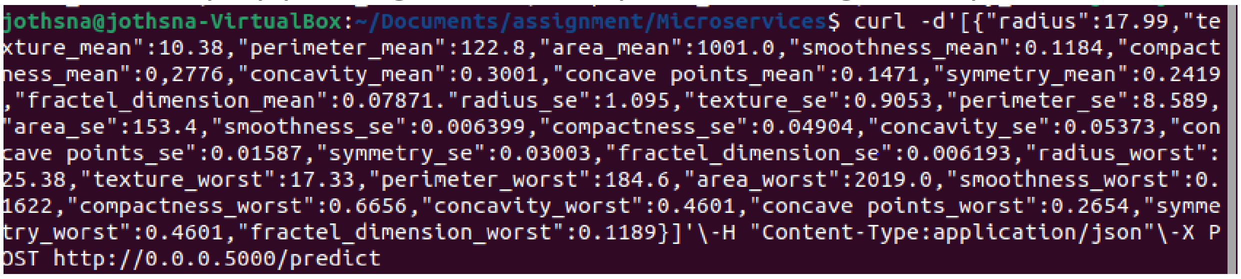


```
(venv) jothsna@jothsna-VirtualBox:~/Documents/assignment/Microservices$ sudo snap install docker # version 24.0.5
Setup snap "docker" (2915) security profiles for auto-connections
Setup snap "docker" (2915) security profiles for auto-connections
Setup snap "docker" (2915) security profiles for auto-connections
Setup snap "docker" (2915) security profiles for auto-connections
Setup snap "docker" (2915) security profiles for auto-connections
docker 24.0.5 from Canonical** installed
(venv) jothsna@jothsna-VirtualBox:~/Documents/assignment/Microservices$ sudo docker build -t yourusername/microservice
ERROR: "docker buildx build" requires exactly 1 argument.
See 'docker buildx build --help'.
```



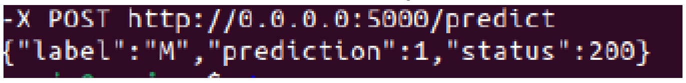
```
):5000 yourusername/microservice
 * Serving Flask app 'ms' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a produc
ent.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on all addresses.
   WARNING: This is a development server. Do not use it in a produc
ent.
 * Running on http://172.17.0.2:5000/ (Press CTRL+C to quit)
```

STEP 11: Run the containerized application as a prediction service and test it locally by passing some example calls and get the prediction.



```
jothsna@jothsna-VirtualBox:~/Documents/assignment/Microservices$ curl -d'[{"radius":17.99,"te
xture_mean":10.38,"perimeter_mean":122.8,"area_mean":1001.0,"smoothness_mean":0.1184,"compact
ness_mean":0,2776,"concavity_mean":0.3001,"concave points_mean":0.1471,"symmetry_mean":0.2419
,"fractel_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,"con
cave points_se":0.01587,"symmetry_se":0.03003,"fractel_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":0.4601,"concave points_worst":0.2654,"symme
try_worst":0.4601,"fractel_dimension_worst":0.1189}]'\-H "Content-Type:application/json"\-X P
OST http://0.0.0.5000/predict
```

And the service will respond as:



```
-X POST http://0.0.0.0:5000/predict
{"label":"M","prediction":1,"status":200}
```