**Week 10 to 12 Graded Assignment**

Quality, Requirements, Architecture, and Microservices.

Note:-this assignment is a continuation of the TA session case study

**PART 1:TA Session Case Study**

W11: TA Session (Week 12)

Quality, Requirements, Architecture, and Microservices.

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

**Context:**

For this case study we are going to work with the following dataset:

<https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)>

Features are computed from a digitized image of a fine needle aspirate (FNA) of a breastmass. They describe characteristics of the cell nuclei present in the image. n the3-dimensional space is that described in: [K. P. Bennett and O. L. Mangasarian: "Robust Linear Programming Discrimination of Two Linearly Inseparable Sets", Optimization Methods and Software 1, 1992, 23-34].

Attribute Information:

●ID number

●Diagnosis (M = malignant, B = benign)

Ten real-valued features are computed for each cell nucleus:

●radius (mean of distances from center to points on the perimeter)

●texture (standard deviation of gray-scale values)

●perimeter

●area.

●smoothness (local variation in radius lengths)

●compactness (perimeter^2 / area - 1.0)

●concavity (severity of concave portions of the contour)

●concave points (number of concave portions of the contour)

●symmetry

●fractal dimension ("coastline approximation" - 1)

**Machine Learning Task:**

To build a machine learning model to predict whether the cancer type is Malignant or Benign.

**Microservice Implementation:**

Our main goal is to design a microservice. Which are standalone, manageable, tested,loosely coupled services that frequently only have one actual use case or function.Microservices are a method for creating a single application as a collection of smallerservices, each of which runs in its own process and communicates with simple tools,frequently an HTTP resource API.

(Fig. Monolithic architecture vs Microservices architecture)

**Python Flask ML Application as a microservice on an Ubuntu virtual machine.**

1.First we must host a Ubuntu virtual machine using the Oracle VM Virtual box.

2.Then we must create the endpoints for subsequently interacting with the client via theHTTP protocol, train and save a machine learning model, and wrap it in a Flask webapplication.

3.Finally we will run and test our application using some example calls.

**Part-1 : Steps and Instructions:**

●Host a Ubuntu Virtual Machine using Oracle VM Virtual Box.

●Set up Visual Studio code on Ubuntu VM.For Setup refer to this

videohttps://www.youtube.com/watch?v=N\_Ve4iAzxq8

●Download the python version 3.10, you may refer to the below commands,

sudo apt update

sudo apt install software-properties-common

sudo add-apt-repository ppa:deadsnakes/ppa

sudo apt install python3.10

Verification of the installation was successful.Python3–version

**Clone the GitHub repository**:

●From Visual Studio code clone the below github

repository.https://github.com/Vikas098766/Microservices.git

**Let's look the project structure below;**

Project Structure is below(Structure of the codes):

MICROSERVICES folder contains multiple folders as below-

1. code\_model\_training

train.py (contains all codes related to training pipeline)

1. data

breastcancer.csv

1. model

model\_ binary.dat.gz (All the model weights are saved here)

4)ms

\_init\_.py

functions.py (Related functions we keep here)

1. tests

example\_calls.test (Any logins etc we will do here)

1. app.py
2. requirements.text (All required packages and corresponding versions are defined here)

**Virtual Environment:**

First we need to create a virtual environment for the cloned github project, to keep track ofevery dependency, it is also useful to use an explicit version of Python.

●To set up a Python 3 virtual environment, navigate to your project folder on yourterminal and type the following command →

python3-m venv venv

This will **create a new virtual environment named venv** using the version ofPython 3 that you have installed on your system.

●Next, you need to **activate the virtual environment** by sourcing the activationscript: command →

source venv/bin/activate

●After executing this command, your prompt will change to indicate that you’renow operating from within the virtual environment.Now with the virtual environment ,we can install the dependencies written inrequirements.txt:Command →

pip install -r requirements.txt

**Model Training and saving the model**

After we have installed all the dependencies we can now run the script in **code\_model\_training/train.py**,this script takes the input data and outputs a trained model and a pipeline for our web service.Command →

python code\_model\_training/train.py

**Web Application**

Finally we can test our web application by running:Command →

flask run -p 5000

**Testing the application and making predictions**:

The below command will run the development server locally and listen to port 5000 where we can test our application! The folder ***/tests*** contain some example calls to test that our application is up and running:

**Example call**:

curl -X GET <http://localhost:5000/info>

**The service should respond**:

{“name”:”Breast Cancer Wisconsin (Diagnostic)”,”version”:”v1.0.0"}

**Example call:**

POST method predict:

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, "concavepoints\_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, "concavepoints\_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>

**The service should respond:**

{“label”:”M”,”prediction”:1,”status”:200}

**PART 2:Graded Assignment**

**Objective:**

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

To create a Docker image containing everything needed to run the application: the application code, libraries, tools, dependencies, and other files and to use the image to run the application in containers.

**Steps to be performed:**

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

2.Set up Visual Studio code on Ubuntu VM. (5 marks)

3.Set up Python. (5 marks)

4.Clone this Github repositoryhttps://github.com/Vikas098766/Microservices.git(1 mark)

5.Create a Virtual Environment. (1 mark)

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

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

8.Test the Flask web application. (5 marks)

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

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

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

**Solution Submission Instructions :**

●Create a git repository name it as {YourName W11\_Graded Assignment },make it public

-Push your solution files to the repository.

-Submit the git URL link in the text entry box.

●Create a document and mention the steps performed, commands for that specific task,and screenshots for each step