Name	Anju Paul
Batch Code	LISUM10: 30
Submission Date	03-07-2022

Deployment on Flask

1) Select the Dataset:

The IRIS dataset from Kaggle

url: https://gist.github.com/netj/8836201

2) Save the Model:

Read the data, identify the features, divide it to test and rain, fit and transform the model, and save the model.

```
model.py  ✕      app.py          <> index.html

model.py > ...
  1    import pandas as pd
  2    import numpy as np
  3    from sklearn.linear_model import LogisticRegression
  4
  5    df = pd.read_csv('./iris.csv')
  6
  7    print(df.head())
  8
  9    from  sklearn.preprocessing  import LabelEncoder
 10    le = LabelEncoder()
 11
 12    df['variety'] = le.fit_transform(df['variety'])
 13    print(df.head())
 14
 15    # Import the feature values:
 16    x = df.iloc[:, 0:-1].values
 17    y = df.iloc[:, -1].values
 18
 19
 20    # Split the dataset into train and test
 21    from sklearn.model_selection import train_test_split
 22    X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=50)
 23
 24    # Feature scaling
 25    from sklearn.preprocessing import StandardScaler
 26    sc = StandardScaler()
 27    X_train = sc.fit_transform(X_train)
 28    X_test= sc.transform(X_test)
 29
 30
 31    print("x_train : ")
 32    print(X_train)
 33
 34    print("y_train : ")
 35    print(X_test)
 36
```

```
 37    log_reg = LogisticRegression(max_iter = 1000)
 38    log_reg.fit(X_train, y_train)
 39
 40    prediction = log_reg.predict(X_test)
 41    print(prediction)
 42
 43    import pickle
 44    pickle.dump(log_reg, open('model.pickle', 'wb'))
 45
 46    |
```

   3) Deploy the model in Flask

Create an html web page to accept the inputs to identify the type of flower.

```html
<!DOCTYPE html>
<html>

<head>
    <meta charset="UTF-8"
    <title>ML API</title>
</head>

<body>
    <div class="login">
        <h1>Predict Flower Type</h1>

        <form action = "{{ url_for('predict')}}"method="post">
            <input type="text" name="sepal_length" placeholder="Sepal Length" required="required" />
            <input type="text" name="sepal_width" placeholder="Sepal Width" required="required" />
            <input type="text" name="petal_length" placeholder="Petal Length" required="required" />
            <input type="text" name="petal_width" placeholder="Petal Width" required="required" />

            <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
        </form>

        <br>
        <br>
        {{ prediction_text }}

    </div>
</body>
</html>
```

Create app.py to render the above webpage html to accept the inputs and predict the ouput using the saved model.

```
model.py        app.py  X      <> index.html
app.py > ...
  1   import numpy as np
  2   from flask import Flask, request, render_template
  3   import pickle
  4
  5   app = Flask(__name__)
  6   model = pickle.load(open('./model.pickle','rb'))
  7
  8
  9   @app.route('/')
 10   def home():
 11       return render_template('index.html')
 12
 13   @app.route('/predict', methods=['POST'])
 14   def predict():
 15
 16       features = [float(x) for x in request.form.values()]
 17       final_features = [np.array(features)]
 18       prediction = model.predict(final_features)
 19
 20       output = prediction[0]
 21       if output == 1 :
 22           o1 = 'Setosa'
 23       elif output == 2:
 24           o1 = 'Versicolor'
 25       else:
 26           o1 = 'Virginica'
 27
 28
 29       return render_template('index.html', prediction_text='The category of the flower is  {}'.format(o1))
 30
 31   if __name__ == "__main__":
 32       app.run(port=5000, debug=True)
```

Run the app.py script

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   JUPYTER

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS D:\Data Glacier Virtual Internship\Flask Deployment>  & 'C:\Users\Anju Paul\AppData\Local\Programs\Python\Python310\python.exe' 'c:\Users\Anju Paul\.v
scode\extensions\ms-python.python-2022.8.1\pythonFiles\lib\python\debugpy\launcher' '53770' '--' 'd:\Data Glacier Virtual Internship\Flask Deployment\app
.py'
 * Serving Flask app 'app' (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: on
 * Restarting with watchdog (windowsapi)
 * Debugger is active!
 * Debugger PIN: 349-272-658
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```
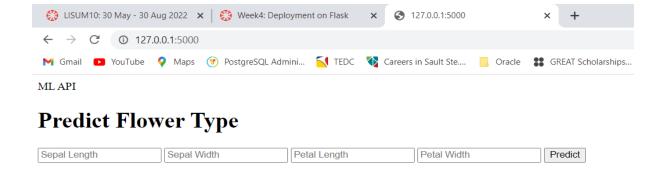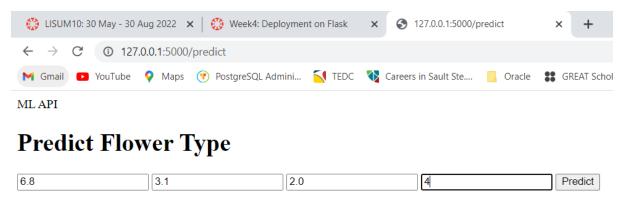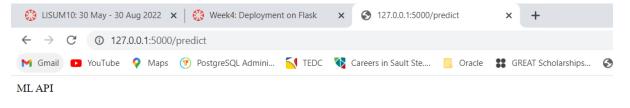
Open the url in the web browser.

ML API

Predict Flower Type

| Sepal Length | Sepal Width | Petal Length | Petal Width | Predict |

Fill in the values and click predict.

ML API

Predict Flower Type

| 6.8 | 3.1 | 2.0 | 4 | Predict |

The output will be as shown in the below figure.

ML API

Predict Flower Type
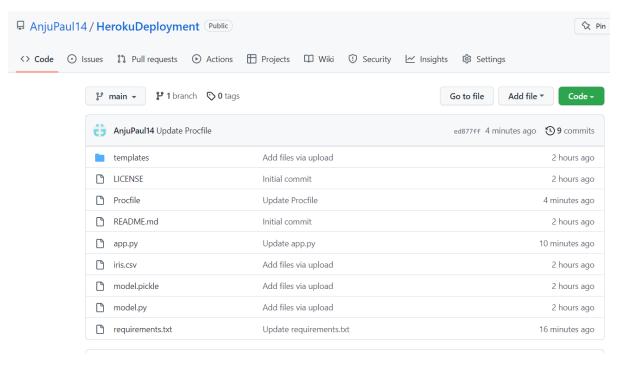
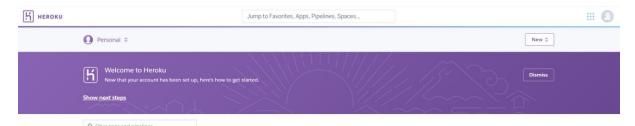| Sepal Length | Sepal Width | Petal Length | Petal Width | Predict |

The category of the flower is Versicolor

Deployment in Heroku:

1) Upload the scripts to the Github library
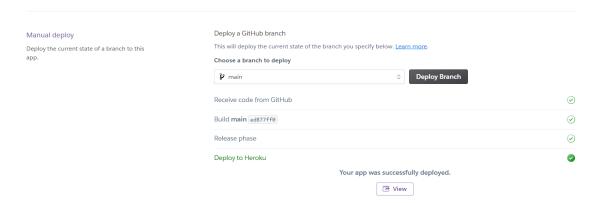

2) Create Heroku Account

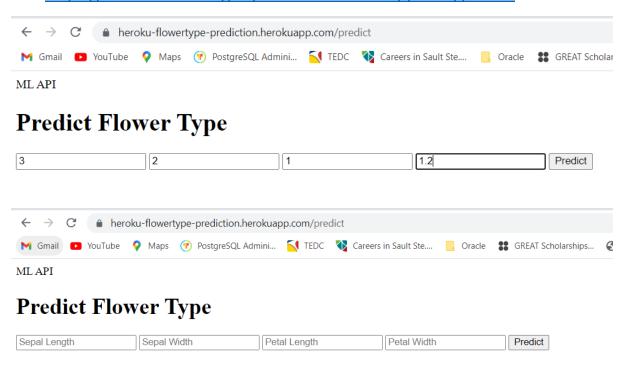3) Link the Github account to Heroku and connect the repository

4) Deploy the Github branch

Deploy a GitHub branch

This will deploy the current state of the branch you specify below. Learn more.

Choose a branch to deploy

main

Deploy Branch

Receive code from GitHub ⊘

Build **main** `ed877ff0` ⊘

Release phase ⊘

Deploy to Heroku ✅

Your app was successfully deployed.

⧉ View

5) Test the app

url : https://heroku-flowertype-prediction.herokuapp.com/predict

heroku-flowertype-prediction.herokuapp.com/predict

M Gmail   ▶ YouTube   ♦ Maps   ⊕ PostgreSQL Admini...   ◀ TEDC   ◆ Careers in Sault Ste....   Oracle   ▦ GREAT Scholar

ML API

Predict Flower Type

| 3 | 2 | 1 | 1.2 | Predict |

heroku-flowertype-prediction.herokuapp.com/predict

M Gmail   ▶ YouTube   ♦ Maps   ⊕ PostgreSQL Admini...   ◀ TEDC   ◆ Careers in Sault Ste....   Oracle   ▦ GREAT Scholarships...

ML API

Predict Flower Type

| Sepal Length | Sepal Width | Petal Length | Petal Width | Predict |

The category of the flower is Versicolor