#name: Himani Aryan
#batchcode: LISUM22
#submissionDate: 28-06-23
#task: Flask Deployment

Deployment on Flask

This task involves the following parts:
In this task, machine learning model is deployed, allowing us to interact with the model through a web interface.

1- Building ML model
2- Deployment using Flask
3-

Introduction:

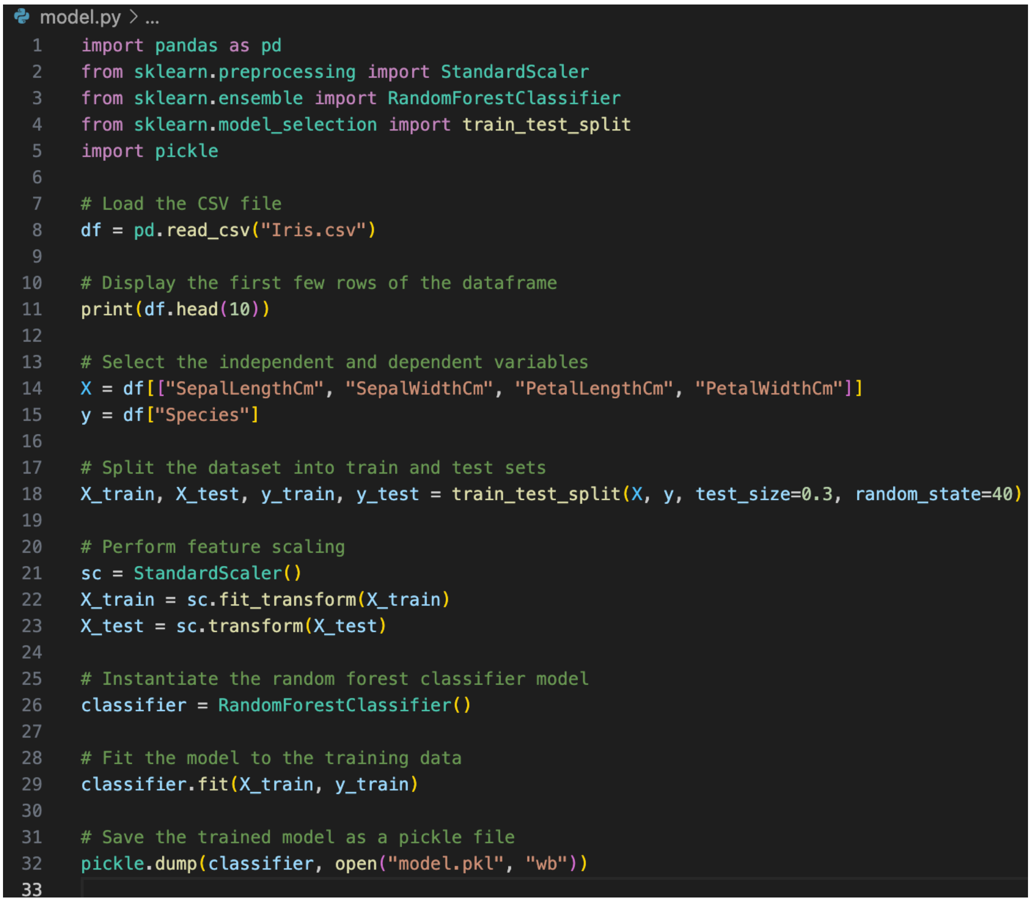
Flask is a web framework for building web applications in Python. It provides a simple and flexible way to create web applications by providing tools and libraries for handling routing, HTTP requests, rendering templates, and managing sessions. Flask is known for its simplicity and ease of use, making it a popular choice for developing web applications and APIs.

Steps followed:

1- Imported iris dataset
2- Created model.py file
3- Created pickle file
4- Created flask application

The screenshots for each step is pasted below:

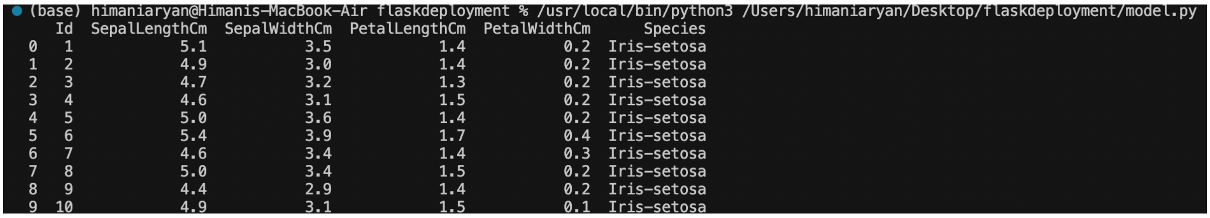
1- Creating model.py



model.py > ...

```python
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
import pickle

# Load the CSV file
df = pd.read_csv("Iris.csv")

# Display the first few rows of the dataframe
print(df.head(10))

# Select the independent and dependent variables
X = df[["SepalLengthCm", "SepalWidthCm", "PetalLengthCm", "PetalWidthCm"]]
y = df["Species"]

# Split the dataset into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=40)

# Perform feature scaling
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Instantiate the random forest classifier model
classifier = RandomForestClassifier()

# Fit the model to the training data
classifier.fit(X_train, y_train)

# Save the trained model as a pickle file
pickle.dump(classifier, open("model.pkl", "wb"))

```

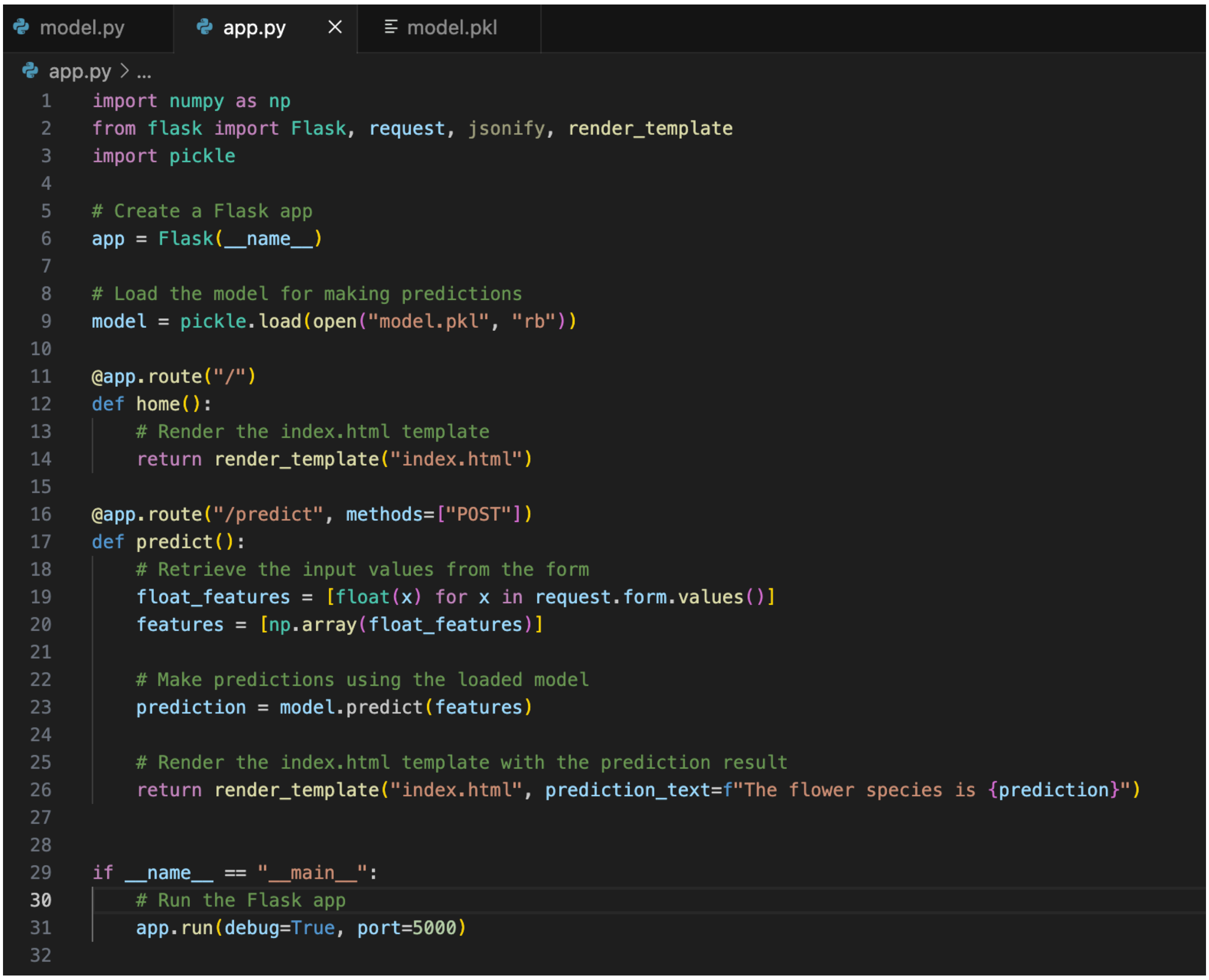
2- result



```
(base) himaniaryan@Himanis-MacBook-Air flaskdeployment % /usr/local/bin/python3 /Users/himaniaryan/Desktop/flaskdeployment/model.py
   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species
0   1            5.1           3.5            1.4           0.2  Iris-setosa
1   2            4.9           3.0            1.4           0.2  Iris-setosa
2   3            4.7           3.2            1.3           0.2  Iris-setosa
3   4            4.6           3.1            1.5           0.2  Iris-setosa
4   5            5.0           3.6            1.4           0.2  Iris-setosa
5   6            5.4           3.9            1.7           0.4  Iris-setosa
6   7            4.6           3.4            1.4           0.3  Iris-setosa
7   8            5.0           3.4            1.5           0.2  Iris-setosa
8   9            4.4           2.9            1.4           0.2  Iris-setosa
9  10            4.9           3.1            1.5           0.1  Iris-setosa
```

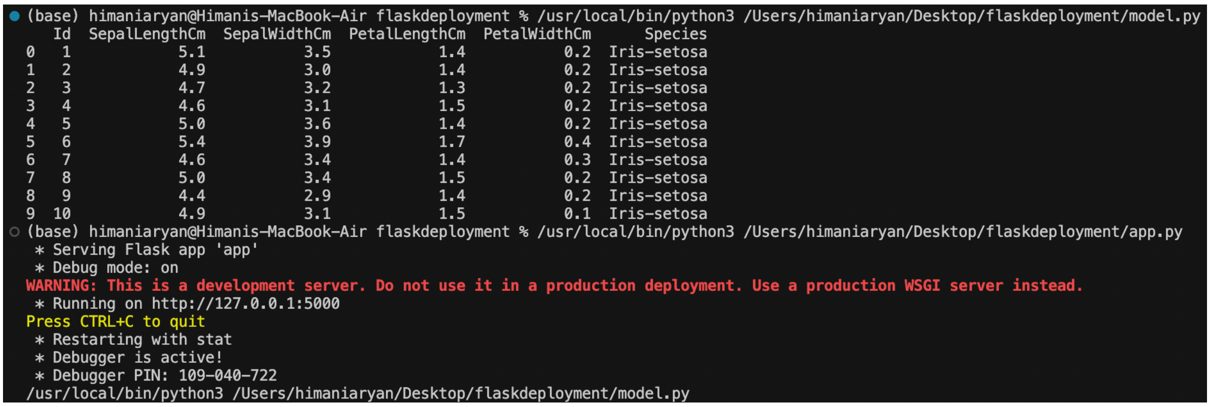
In this task, the Iris dataset is chosen for its simplicity and widespread usage in the machine learning community. The dataset consists of measurements of various attributes of iris flowers, making it a suitable choice for building and deploying a machine learning model. The simplicity of the dataset allows for easy understanding and interpretation of the model's predictions

3- creating app.py



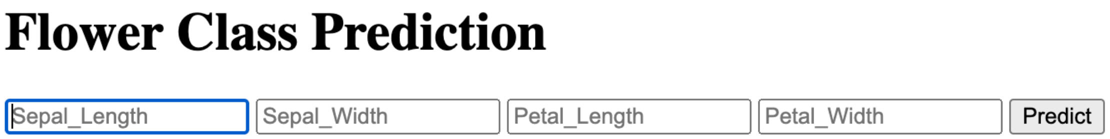
```python
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle

# Create a Flask app
app = Flask(__name__)

# Load the model for making predictions
model = pickle.load(open("model.pkl", "rb"))

@app.route("/")
def home():
    # Render the index.html template
    return render_template("index.html")

@app.route("/predict", methods=["POST"])
def predict():
    # Retrieve the input values from the form
    float_features = [float(x) for x in request.form.values()]
    features = [np.array(float_features)]

    # Make predictions using the loaded model
    prediction = model.predict(features)

    # Render the index.html template with the prediction result
    return render_template("index.html", prediction_text=f"The flower species is {prediction}")


if __name__ == "__main__":
    # Run the Flask app
    app.run(debug=True, port=5000)

```

4- result



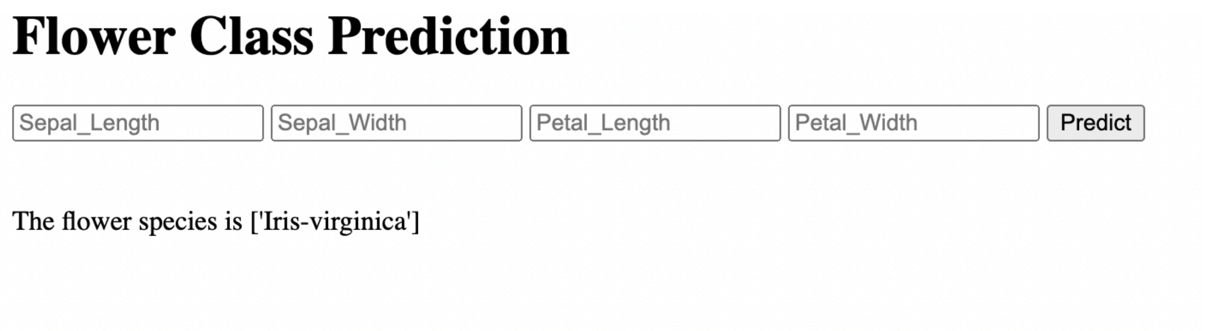
```
● (base) himaniaryan@Himanis-MacBook-Air flaskdeployment % /usr/local/bin/python3 /Users/himaniaryan/Desktop/flaskdeployment/model.py
     Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species
  0   1            5.1           3.5            1.4           0.2  Iris-setosa
  1   2            4.9           3.0            1.4           0.2  Iris-setosa
  2   3            4.7           3.2            1.3           0.2  Iris-setosa
  3   4            4.6           3.1            1.5           0.2  Iris-setosa
  4   5            5.0           3.6            1.4           0.2  Iris-setosa
  5   6            5.4           3.9            1.7           0.4  Iris-setosa
  6   7            4.6           3.4            1.4           0.3  Iris-setosa
  7   8            5.0           3.4            1.5           0.2  Iris-setosa
  8   9            4.4           2.9            1.4           0.2  Iris-setosa
  9  10            4.9           3.1            1.5           0.1  Iris-setosa
○ (base) himaniaryan@Himanis-MacBook-Air flaskdeployment % /usr/local/bin/python3 /Users/himaniaryan/Desktop/flaskdeployment/app.py
   * Serving Flask app 'app'
   * Debug mode: on
  WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
   * Running on http://127.0.0.1:5000
  Press CTRL+C to quit
   * Restarting with stat
   * Debugger is active!
   * Debugger PIN: 109-040-722
  /usr/local/bin/python3 /Users/himaniaryan/Desktop/flaskdeployment/model.py
```

5- testing predict method



Flower Class Prediction

6- prediction



Flower Class Prediction

The flower species is ['Iris-virginica']

By following the steps of model training, feature scaling, and deployment, a functional web application for predicting the species of Iris flowers is created.