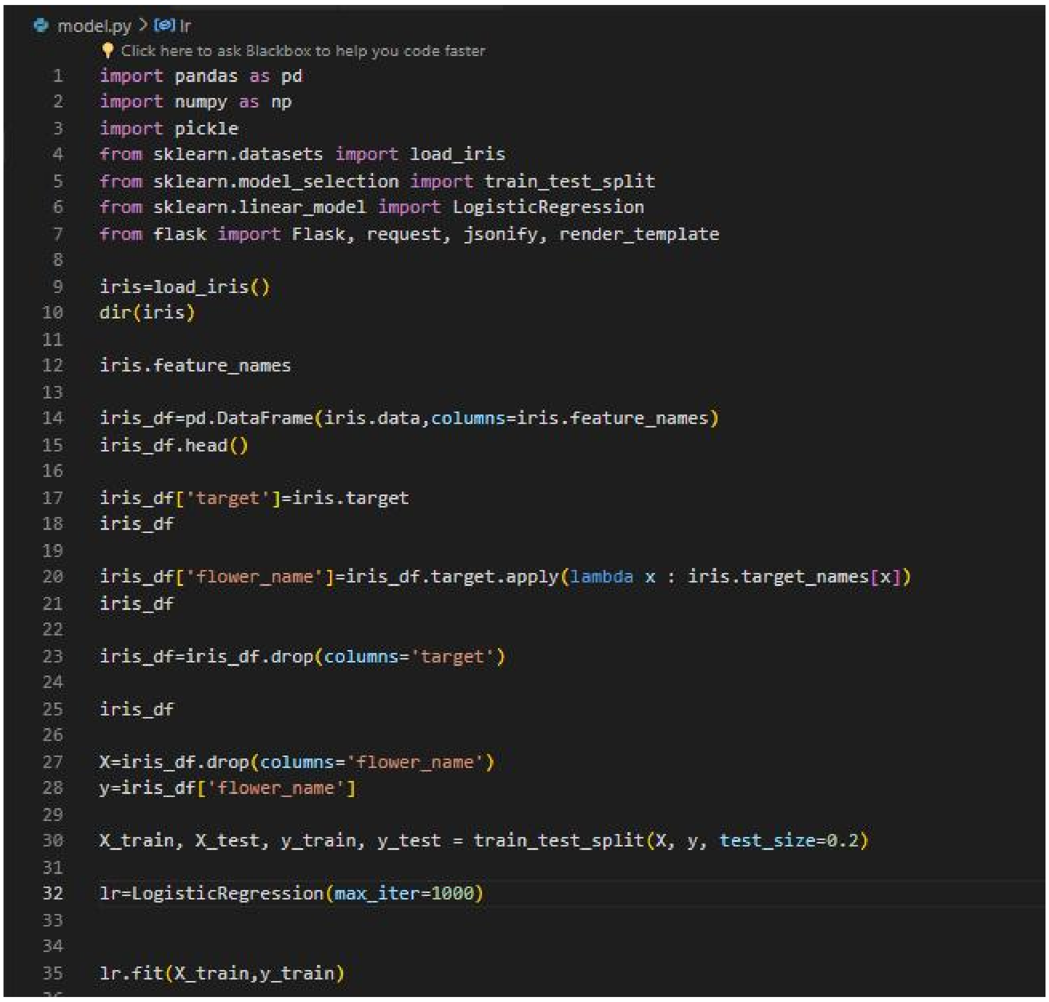
Model Deployment on Flask by Bisma Azeem

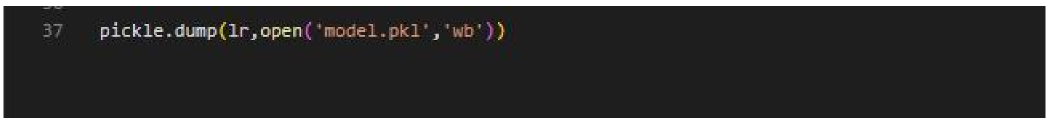
Batch code: LISUM32

Step 1: Data pre processing and Data Modeling (model.py)



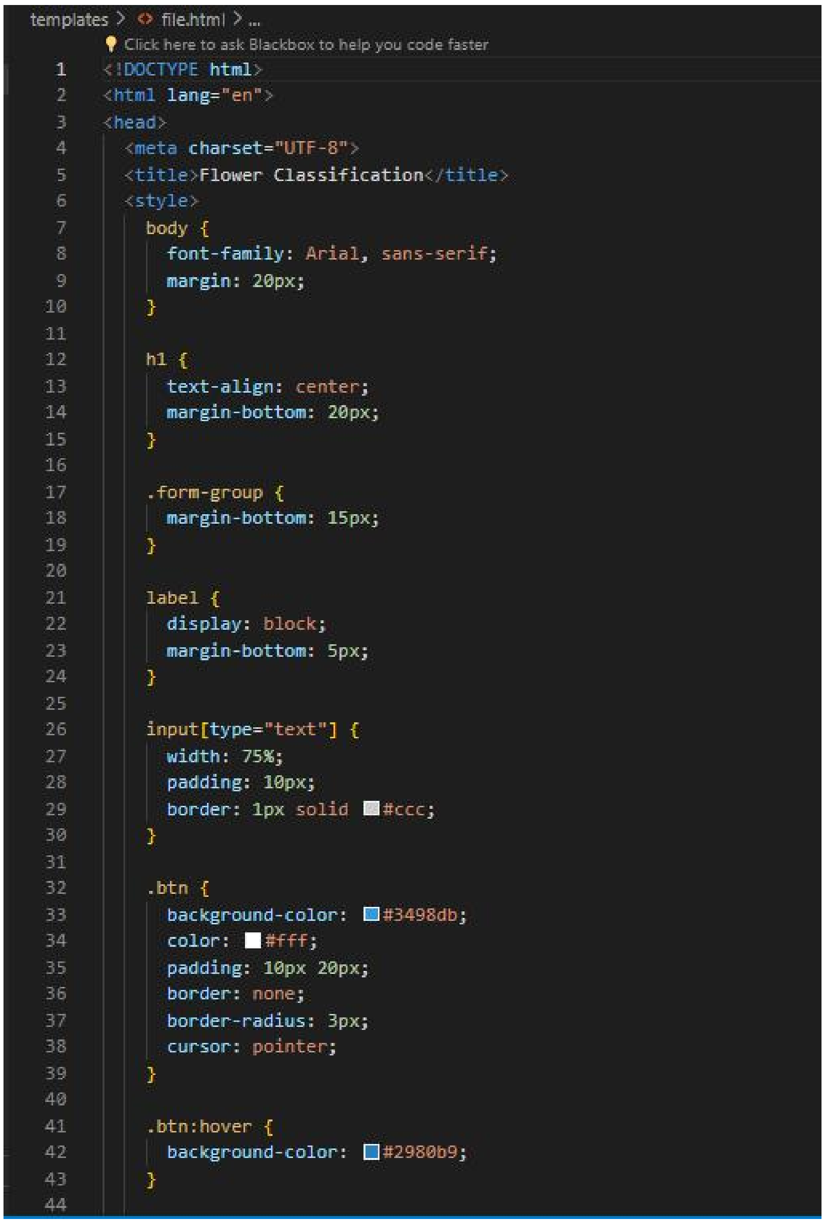
```python
import pandas as pd
import numpy as np
import pickle
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from flask import Flask, request, jsonify, render_template

iris=load_iris()
dir(iris)

iris.feature_names

iris_df=pd.DataFrame(iris.data,columns=iris.feature_names)
iris_df.head()

iris_df['target']=iris.target
iris_df

iris_df['flower_name']=iris_df.target.apply(lambda x : iris.target_names[x])
iris_df

iris_df=iris_df.drop(columns='target')

iris_df

X=iris_df.drop(columns='flower_name')
y=iris_df['flower_name']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

lr=LogisticRegression(max_iter=1000)


lr.fit(X_train,y_train)
```

Step 2: Pickling the model (model.pkl)



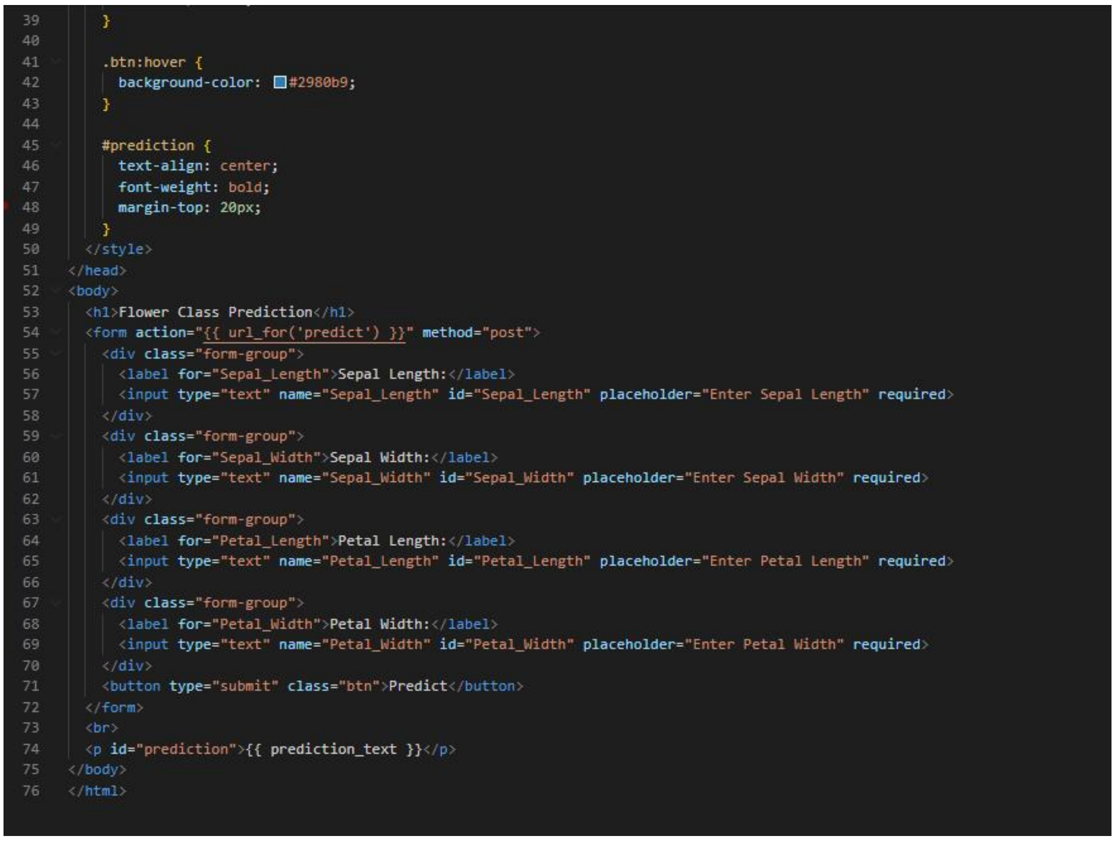
```python
pickle.dump(lr,open('model.pkl','wb'))
```

Step 3: Website design file(file.html)



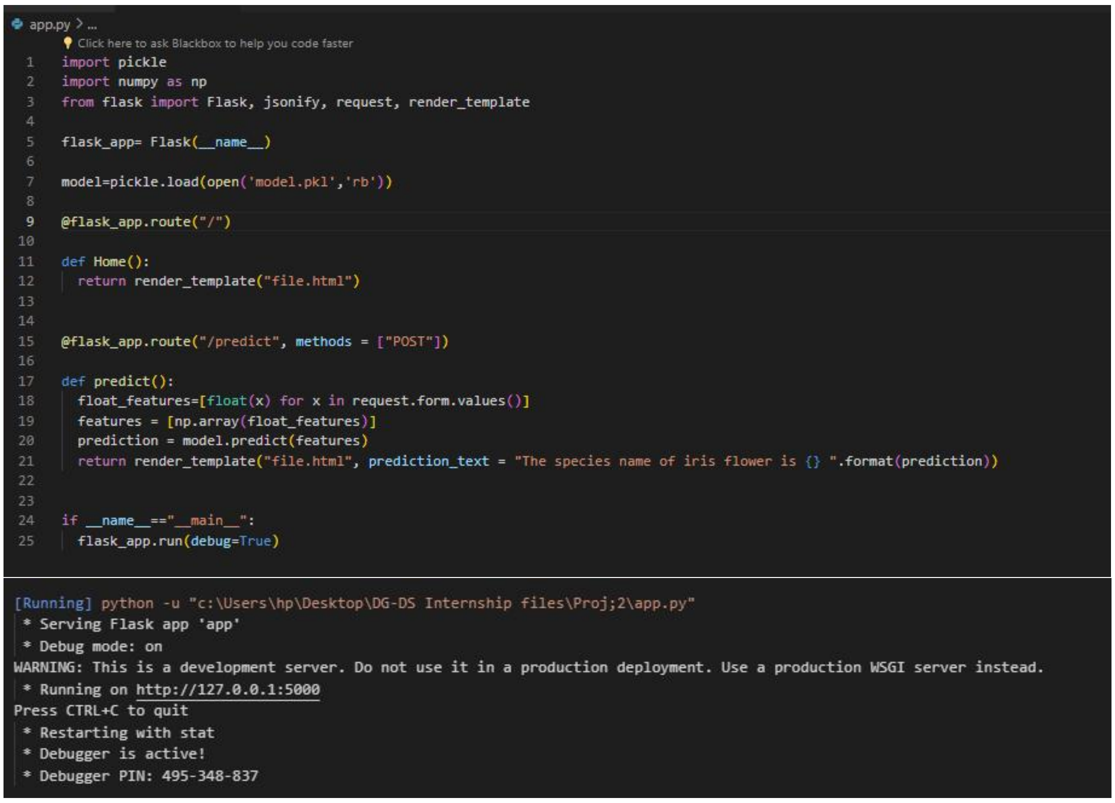
templates > file.html > ...
Click here to ask Blackbox to help you code faster

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Flower Classification</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      margin: 20px;
    }

    h1 {
      text-align: center;
      margin-bottom: 20px;
    }

    .form-group {
      margin-bottom: 15px;
    }

    label {
      display: block;
      margin-bottom: 5px;
    }

    input[type="text"] {
      width: 75%;
      padding: 10px;
      border: 1px solid #ccc;
    }

    .btn {
      background-color: #3498db;
      color: #fff;
      padding: 10px 20px;
      border: none;
      border-radius: 3px;
      cursor: pointer;
    }

    .btn:hover {
      background-color: #2980b9;
    }

```



```
    }

    .btn:hover {
      background-color: #2980b9;
    }

    #prediction {
      text-align: center;
      font-weight: bold;
      margin-top: 20px;
    }
  </style>
</head>
<body>
  <h1>Flower Class Prediction</h1>
  <form action="{{ url_for('predict') }}" method="post">
    <div class="form-group">
      <label for="Sepal_Length">Sepal Length:</label>
      <input type="text" name="Sepal_Length" id="Sepal_Length" placeholder="Enter Sepal Length" required>
    </div>
    <div class="form-group">
      <label for="Sepal_Width">Sepal Width:</label>
      <input type="text" name="Sepal_Width" id="Sepal_Width" placeholder="Enter Sepal Width" required>
    </div>
    <div class="form-group">
      <label for="Petal_Length">Petal Length:</label>
      <input type="text" name="Petal_Length" id="Petal_Length" placeholder="Enter Petal Length" required>
    </div>
    <div class="form-group">
      <label for="Petal_Width">Petal Width:</label>
      <input type="text" name="Petal_Width" id="Petal_Width" placeholder="Enter Petal Width" required>
    </div>
    <button type="submit" class="btn">Predict</button>
  </form>
  <br>
  <p id="prediction">{{ prediction_text }}</p>
</body>
</html>
```

Step 4: Model Deployment on Flask(app.py)



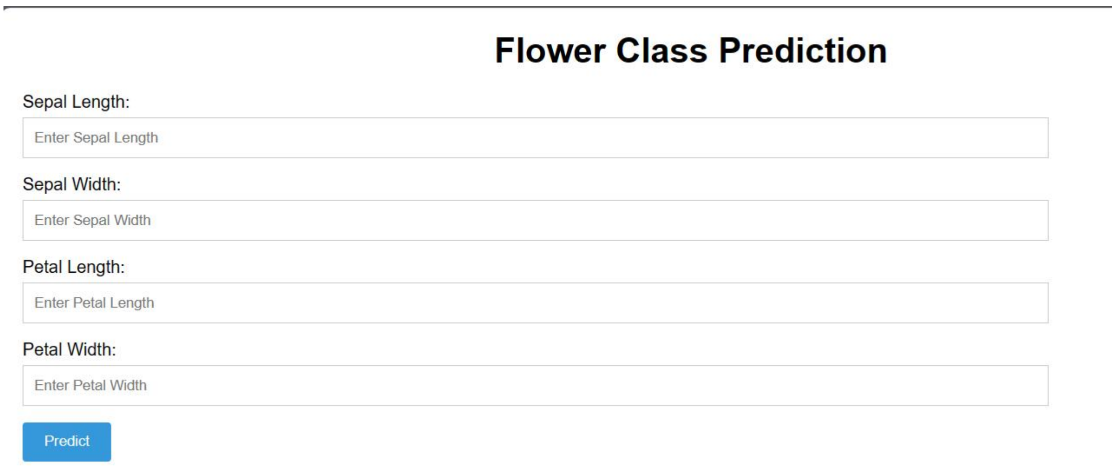
app.py > ...

Click here to ask Blackbox to help you code faster

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

flask_app= Flask(__name__)

model=pickle.load(open('model.pkl','rb'))

@flask_app.route("/")

def Home():
    return render_template("file.html")


@flask_app.route("/predict", methods = ["POST"])

def predict():
    float_features=[float(x) for x in request.form.values()]
    features = [np.array(float_features)]
    prediction = model.predict(features)
    return render_template("file.html", prediction_text = "The species name of iris flower is {} ".format(prediction))


if __name__=="__main__":
    flask_app.run(debug=True)
```

```
[Running] python -u "c:\Users\hp\Desktop\DG-DS Internship files\Proj;2\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: 495-348-837
```

Step 5: Website Interface of my App:



Flower Class Prediction

Sepal Length:

Enter Sepal Length

Sepal Width:

Enter Sepal Width

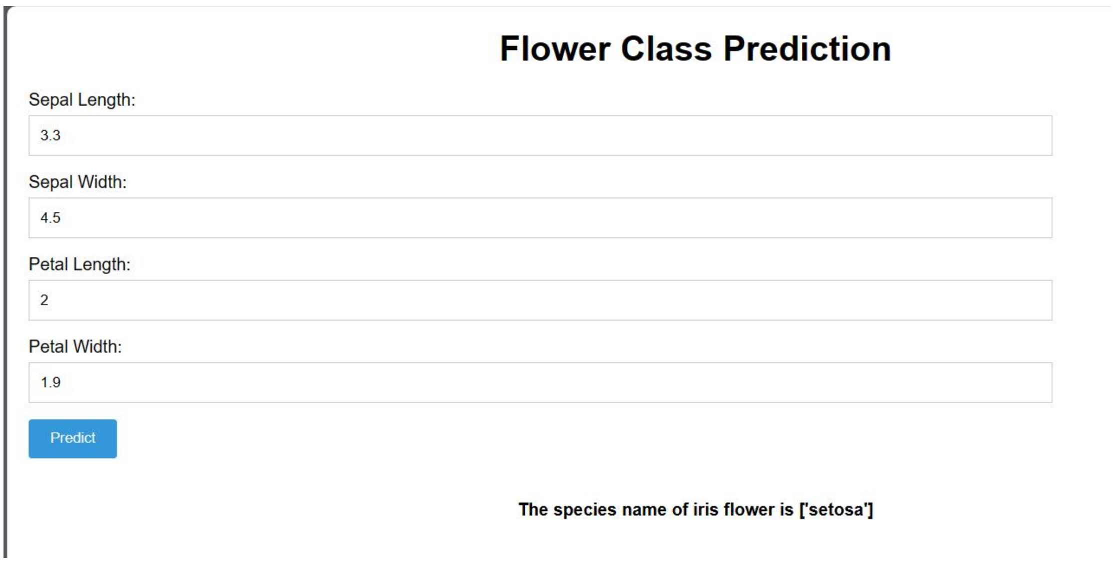
Petal Length:

Enter Petal Length

Petal Width:

Enter Petal Width

Predict



Flower Class Prediction

Sepal Length:

3.3

Sepal Width:

4.5

Petal Length:

2

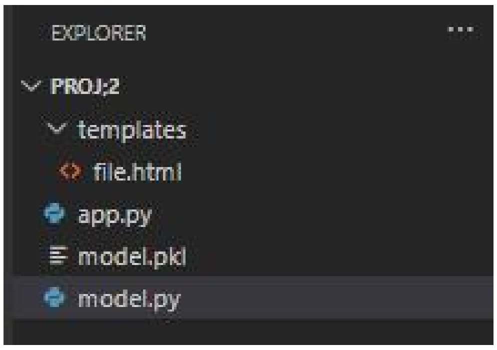
Petal Width:

1.9

Predict

The species name of iris flower is ['setosa']

All files overview:



```
EXPLORER
PROJ:2
  templates
    file.html
  app.py
  model.pkl
  model.py
```

##THE END######################################