

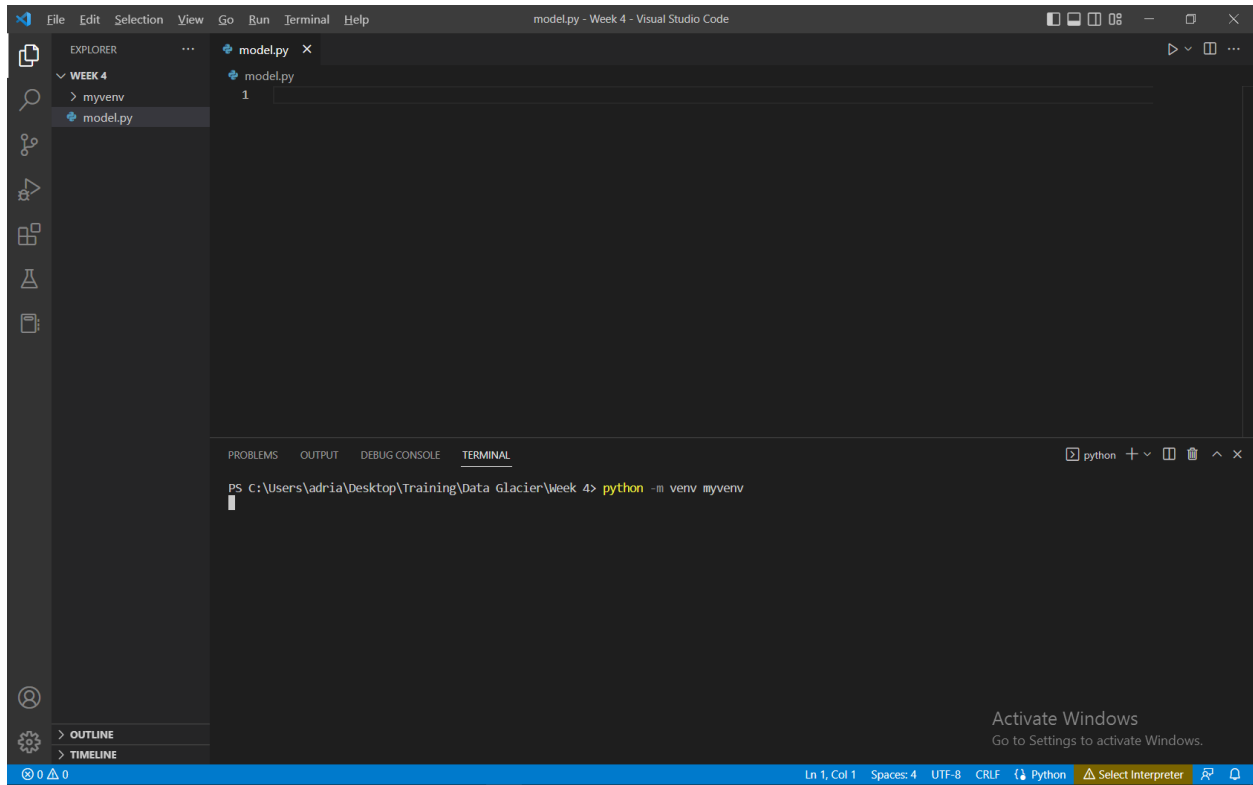
Name: Adrian Baysa

Batch Code: LISUM16

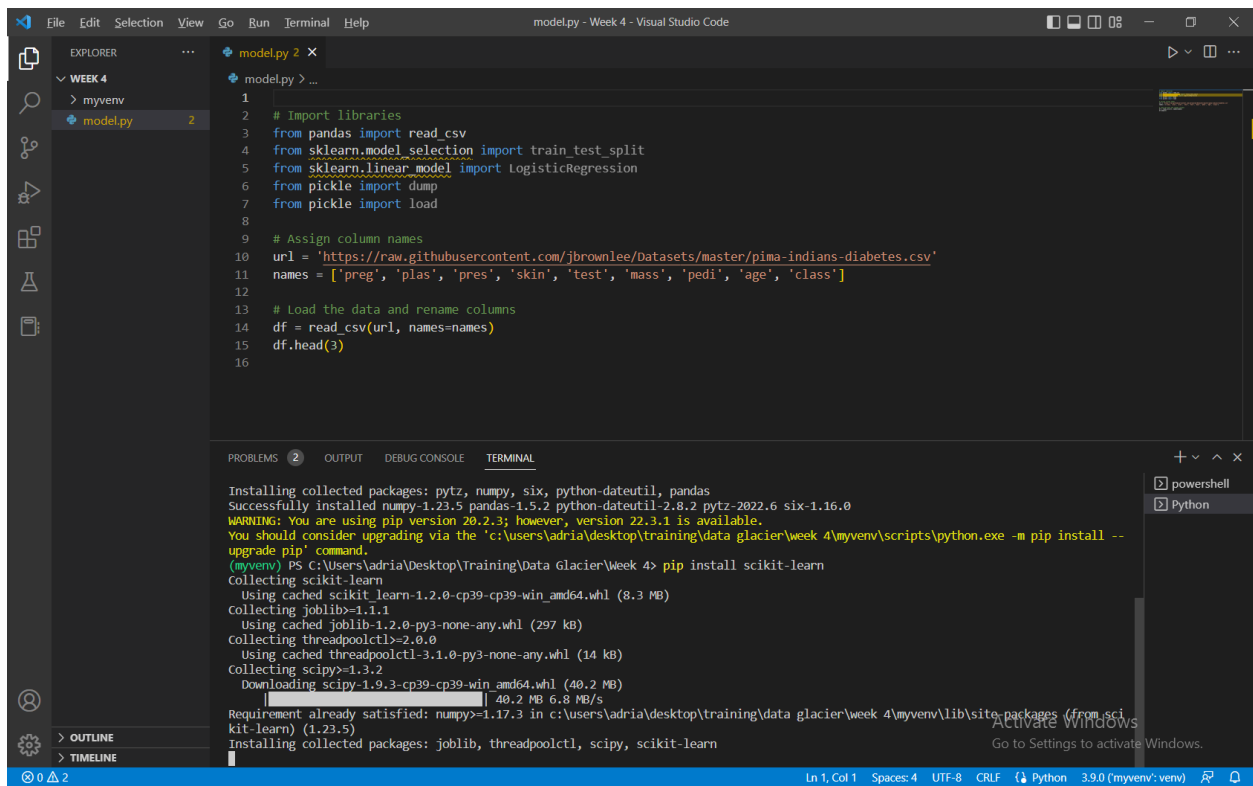
Submission Date: 13 December 2022

Submitted to: n/a

Create environment



Install libraries



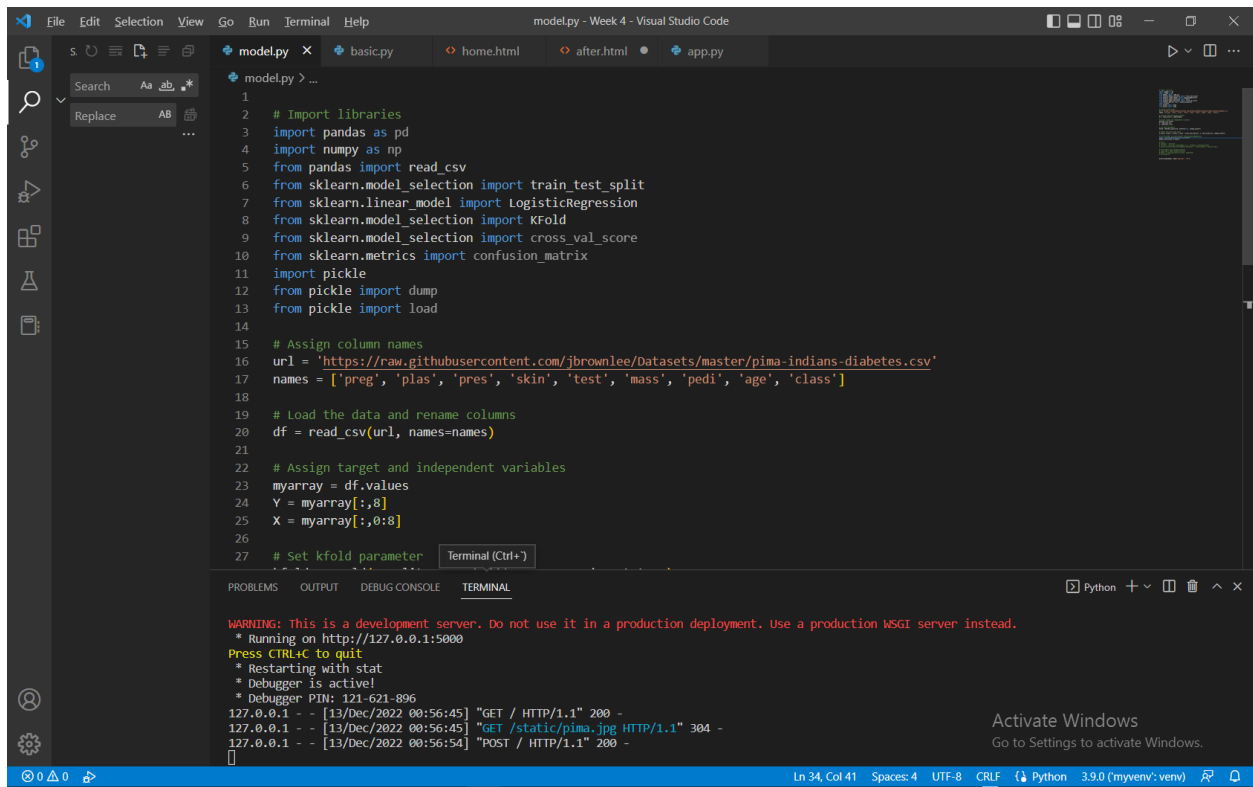
The screenshot shows the Visual Studio Code interface with a Python file named `model.py` open. The Explorer sidebar on the left shows the project structure with `WEEK 4` and `myenv` folders. The `model.py` file is selected, showing the following code:

```
1
2 # Import libraries
3 from pandas import read_csv
4 from sklearn.model_selection import train_test_split
5 from sklearn.linear_model import LogisticRegression
6 from pickle import dump
7 from pickle import load
8
9 # Assign column names
10 url = 'https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.csv'
11 names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
12
13 # Load the data and rename columns
14 df = read_csv(url, names=names)
15 df.head(3)
16
```

The TERMINAL panel at the bottom shows the output of the `pip install` command. It indicates that several packages (numpy, pandas, python-dateutil, pytz, joblib, threadpoolctl, scipy) were successfully installed or updated. The status bar at the bottom shows the current file is `model.py` at line 1, column 1, with 4 spaces, using UTF-8 encoding and CRLF line endings. The Python interpreter is set to `Python 3.9.0 (myenv: venv)`.

Create Model:

- Import libraries
- Download the dataset
- Identify X and y (target variable)
- Set kfold parameter
- Create train and test data
- Fit the model (Logistic Regression)
- Dump the model as a pickle file



The screenshot shows the Visual Studio Code interface. The editor window displays a Python script named `model.py` with the following code:

```
1
2 # Import libraries
3 import pandas as pd
4 import numpy as np
5 from pandas import read_csv
6 from sklearn.model_selection import train_test_split
7 from sklearn.linear_model import LogisticRegression
8 from sklearn.model_selection import KFold
9 from sklearn.model_selection import cross_val_score
10 from sklearn.metrics import confusion_matrix
11 import pickle
12 from pickle import dump
13 from pickle import load
14
15 # Assign column names
16 url = 'https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.csv'
17 names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
18
19 # Load the data and rename columns
20 df = read_csv(url, names=names)
21
22 # Assign target and independent variables
23 myarray = df.values
24 Y = myarray[:,8]
25 X = myarray[:,0:8]
26
27 # Set kfold parameter
```

The terminal window at the bottom shows the output of a web server running on `http://127.0.0.1:5000`. It displays a warning about using a development server, followed by messages indicating the server is running, restarting with `stat`, and the debugger is active. The terminal also shows several HTTP requests and responses:

```
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: 121-621-896
127.0.0.1 - - [13/Dec/2022 00:56:45] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2022 00:56:45] "GET /static/pima.jpg HTTP/1.1" 304 -
127.0.0.1 - - [13/Dec/2022 00:56:54] "POST / HTTP/1.1" 200 -
```

The status bar at the bottom indicates the current file is `Ln 34, Col 41`, with `Spaces: 4`, `UTF-8` encoding, `CRLF` line endings, and the Python environment is `Python 3.9.0 (myenv\venv)`.

```
File Edit Selection View Go Run Terminal Help
model.py - Week 4 - Visual Studio Code
model.py x basic.py home.html after.html app.py
model.py > ...
14
15 # Assign column names
16 url = 'https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.csv'
17 names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
18
19 # Load the data and rename columns
20 df = read_csv(url, names=names)
21
22 # Assign target and independent variables
23 myarray = df.values
24 Y = myarray[:,8]
25 X = myarray[:,0:8]
26
27 # Set kfold parameter
28 kfold = KFold(n_splits=10, shuffle=True, random_state=7)
29
30 # Create test and train split
31 X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.33, random_state=7)
32
33 # Fit the model on train dataset using LogisticRegression
34 model = LogisticRegression(max_iter=1000)
35 model.fit(X_train, Y_train)
36
37 # Score
38 # scoring = 'accuracy'
39 # results = cross_val_score(model, X, Y, cv=kfold, scoring=scoring)
40 # print("Accuracy Mean and Standard Deviation:", results.mean(), results.std())
41
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
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: 121-621-896
127.0.0.1 - - [13/Dec/2022 00:56:45] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2022 00:56:45] "GET /static/pima.jpg HTTP/1.1" 304 -
127.0.0.1 - - [13/Dec/2022 00:56:54] "POST / HTTP/1.1" 200 -
Ln 34, Col 41 Spaces: 4 UTF-8 CRLF Python 3.9.0 (myenv\venv)
```

```
model.py x basic.py home.html after.html app.py
model.py > ...
27 # Set kfold parameter
28 kfold = KFold(n_splits=10, shuffle=True, random_state=7)
29
30 # Create test and train split
31 X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.33, random_state=7)
32
33 # Fit the model on train dataset using LogisticRegression
34 model = LogisticRegression(max_iter=1000)
35 model.fit(X_train, Y_train)
36
37 # Score
38 # scoring = 'accuracy'
39 # results = cross_val_score(model, X, Y, cv=kfold, scoring=scoring)
40 # print("Accuracy Mean and Standard Deviation:", results.mean(), results.std())
41
42 # Test Predict and confusion_matrix
43 # predicted = model.predict(X_test)
44 # matrix = confusion_matrix(Y_test, predicted)
45 # print(matrix)
46
47
48 pickle.dump(model, open('pima.pkl', 'wb'))
Terminal (Ctrl+)
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
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: 121-621-896
127.0.0.1 - - [13/Dec/2022 00:56:45] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2022 00:56:45] "GET /static/pima.jpg HTTP/1.1" 304 -
127.0.0.1 - - [13/Dec/2022 00:56:54] "POST / HTTP/1.1" 200 -
Ln 34, Col 41 Spaces: 4 UTF-8 CRLF Python 3.9.0 (myenv\venv)
```

Create home.html page

The screenshot shows the Visual Studio Code interface with the 'home.html' file open. The file contains HTML code for a Pima Indians Diabetes Prediction form. The form has a title 'Pima Indians Diabetes Prediction' and a POST method. It includes input fields for Preg, Plas, Pres, Skin, Test, Mass, Pedi, and Age, each with a placeholder text. A submit button is labeled 'Predict Cases of Diabetes'. Below the form is an image placeholder for 'pima.jpg'. The terminal at the bottom shows a warning about the development server and logs for a GET request to the static/pima.jpg file.

```
1 <html>
2 <body bgcolor=#B2BEB5>
3
4 <center>
5
6 <h1> Pima Indians Diabetes Prediction </h1><br>
7
8 <form method="POST", action="{{url_for('home')}}">
9 <b> Preg : <input type="text", name='a', placeholder="Enter value for Preg "> <br><br>
10 Plas : <input type="text", name='b', placeholder="Enter value for Plas"> <br><br>
11 Pres : <input type="text", name='c', placeholder="Enter value for Pres"> <br><br>
12 Skin : <input type="text", name='d', placeholder="Enter value for Skin"> <br><br>
13 Test : <input type="text", name='e', placeholder="Enter value for Test"> <br><br>
14 Mass : <input type="text", name='f', placeholder="Enter value for Mass"> <br><br>
15 Pedi : <input type="text", name='g', placeholder="Enter value for Pedi"> <br><br>
16 Age : <input type="text", name='h', placeholder="Enter value for Age"> <br><br><br></b>
17 <input type="submit" , value='Predict Cases of Diabetes' >
18 </form>
19
20 <img src='static/pima.jpg' alt="pima">
21
22 </center>
23
24 </body>
25 </html>
```

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: 121-621-896
127.0.0.1 - - [13/Dec/2022 00:56:45] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2022 00:56:45] "GET /static/pima.jpg HTTP/1.1" 304 -
127.0.0.1 - - [13/Dec/2022 00:56:54] "POST / HTTP/1.1" 200 -

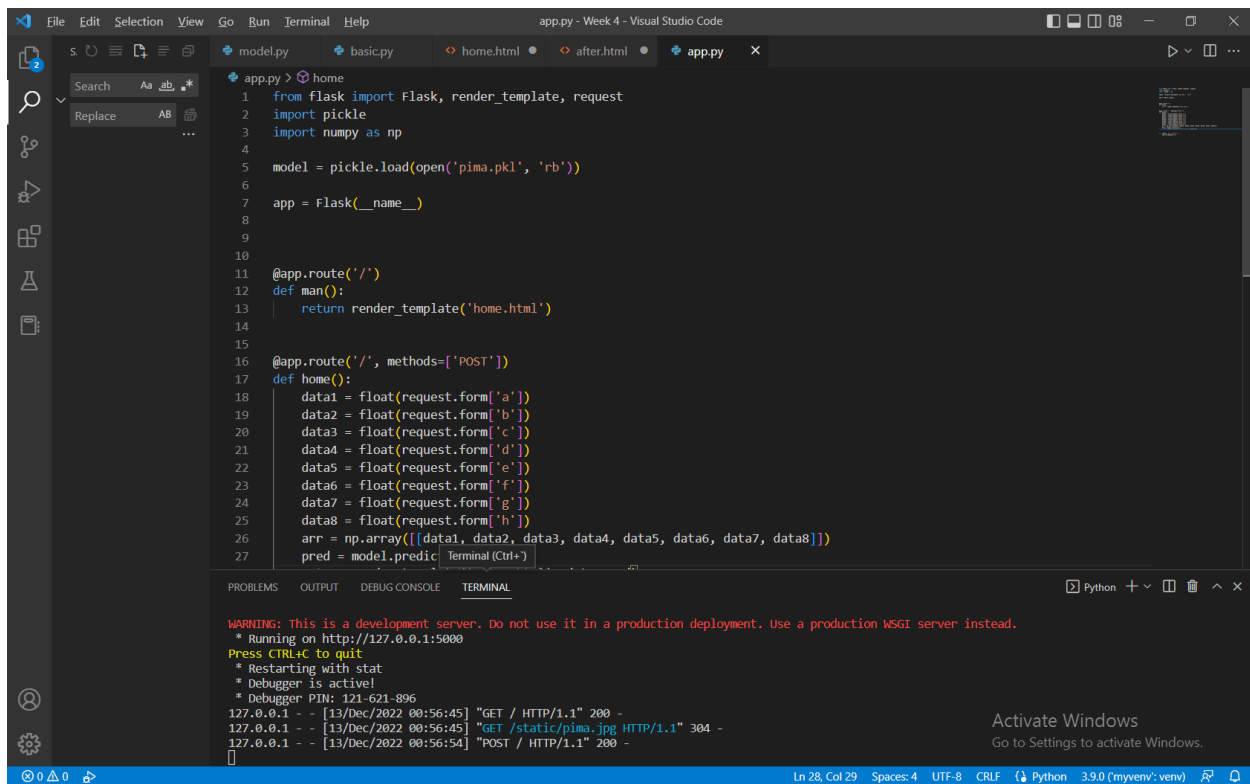
Create after.html page

The screenshot shows the Visual Studio Code interface with the 'after.html' file open. The file contains HTML code for a prediction result page. It has a title 'PREDICTION : ' and a conditional display based on the 'data' variable. If data is 1, it displays 'Tested Positive for Diabetes'. Otherwise, it displays 'Did Not Test Positive for Diabetes'. A link is provided to go back to the home page. The terminal at the bottom shows a warning about the development server and logs for a GET request to the static/pima.jpg file.

```
1 <html>
2
3 <body bgcolor=#B2BEB5>
4
5 <center>
6
7 <h1> PREDICTION : </h1>
8
9 {%if data == 1%}
10 <h1>Tested Positive for Diabetes</h1>
11
12
13 {%else%}
14 <h1>Did Not Test Positive for Diabetes</h1>
15
16
17 {%endif%}
18
19 <br><br>
20 <a href='/'>go back to home page</a>
21
22 </center>
23
24 </body>
25
26
27 </html>
```

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: 121-621-896
127.0.0.1 - - [13/Dec/2022 00:56:45] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2022 00:56:45] "GET /static/pima.jpg HTTP/1.1" 304 -
127.0.0.1 - - [13/Dec/2022 00:56:54] "POST / HTTP/1.1" 200 -

Create app.py

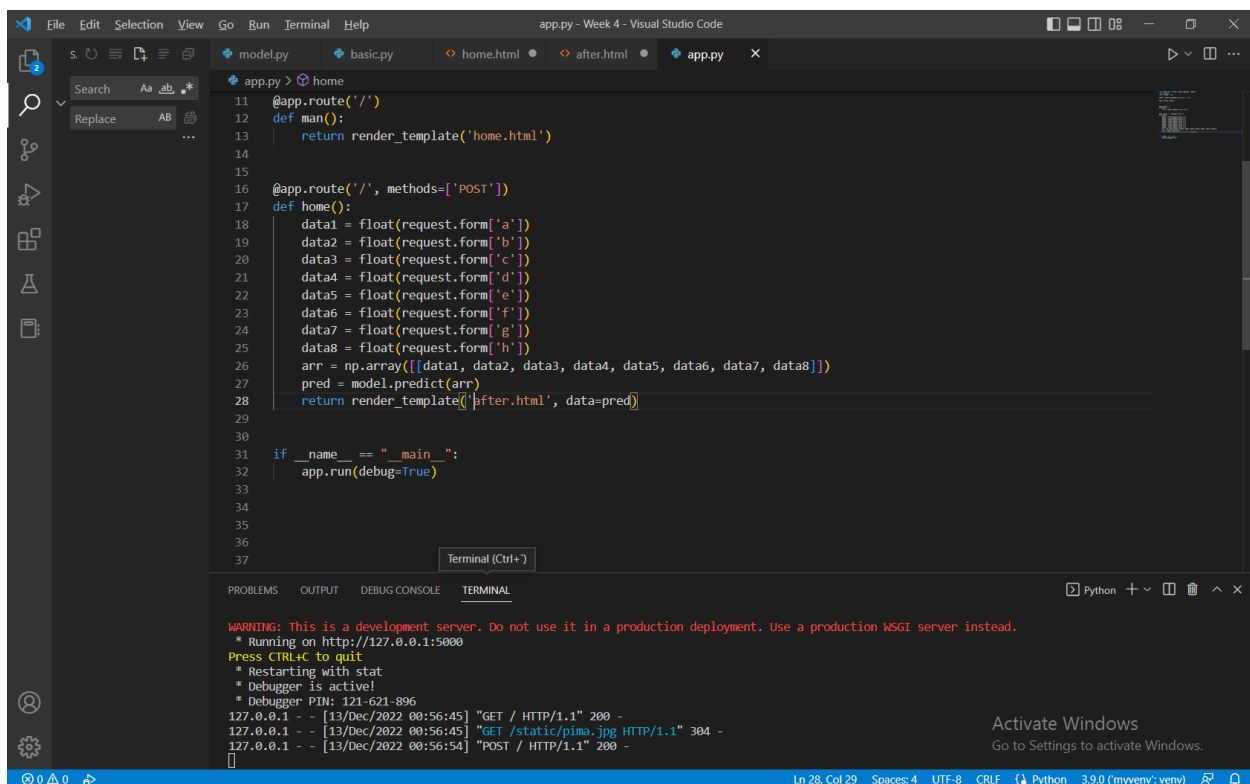


The screenshot shows the Visual Studio Code editor with the file `app.py` open. The code defines a Flask application that loads a pre-trained model and serves a home page. The terminal shows the application running on `http://127.0.0.1:5000` and receiving several requests.

```
1 from flask import Flask, render_template, request
2 import pickle
3 import numpy as np
4
5 model = pickle.load(open('pima.pkl', 'rb'))
6
7 app = Flask(__name__)
8
9
10
11 @app.route('/')
12 def man():
13     return render_template('home.html')
14
15
16 @app.route('/', methods=['POST'])
17 def home():
18     data1 = float(request.form['a'])
19     data2 = float(request.form['b'])
20     data3 = float(request.form['c'])
21     data4 = float(request.form['d'])
22     data5 = float(request.form['e'])
23     data6 = float(request.form['f'])
24     data7 = float(request.form['g'])
25     data8 = float(request.form['h'])
26     arr = np.array([data1, data2, data3, data4, data5, data6, data7, data8])
27     pred = model.predict(arr)
```

Terminal Output:

```
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: 121-621-896
127.0.0.1 - - [13/Dec/2022 00:56:45] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2022 00:56:45] "GET /static/pima.jpg HTTP/1.1" 304 -
127.0.0.1 - - [13/Dec/2022 00:56:54] "POST / HTTP/1.1" 200 -
```



The screenshot shows the Visual Studio Code editor with the file `app.py` open. The code has been updated to include a `home` route that uses the model's prediction to render the `after.html` template. The terminal shows the application running on `http://127.0.0.1:5000` and receiving several requests.

```
11 @app.route('/')
12 def man():
13     return render_template('home.html')
14
15
16 @app.route('/', methods=['POST'])
17 def home():
18     data1 = float(request.form['a'])
19     data2 = float(request.form['b'])
20     data3 = float(request.form['c'])
21     data4 = float(request.form['d'])
22     data5 = float(request.form['e'])
23     data6 = float(request.form['f'])
24     data7 = float(request.form['g'])
25     data8 = float(request.form['h'])
26     arr = np.array([data1, data2, data3, data4, data5, data6, data7, data8])
27     pred = model.predict(arr)
28     return render_template('after.html', data=pred)
29
30
31 if __name__ == "__main__":
32     app.run(debug=True)
33
34
35
36
37
```

Terminal Output:

```
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: 121-621-896
127.0.0.1 - - [13/Dec/2022 00:56:45] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [13/Dec/2022 00:56:45] "GET /static/pima.jpg HTTP/1.1" 304 -
127.0.0.1 - - [13/Dec/2022 00:56:54] "POST / HTTP/1.1" 200 -
```

Test app.py

Search re... x Datasets/... x Home Pa... x Untitled4... x Create ve... x deploy-m... x machine l... x python -... x 127.0.0.1... x

127.0.0.1:5000

Pima Indians Diabetes Prediction

Preg :

Plas :

Pres :


Skin :

Test :

Mass :

Pedi :

Age :



Activate Windows
Go to Settings to activate Windows.

Enter values:

Search re...Datasets/Untitled4:Create ver...deploy-m...machine l...python - l...127.0.0.1:5000

127.0.0.1:5000

Pima Indians Diabetes Prediction

Preg : 6

Plas : 148

Pres : 72

Skin : 35


Test : 0

Mass : 33.6

Pedi : 0.627

Age : 50

Predict Cases of Diabetes



Activate Windows
Go to Settings to activate Windows.

See prediction results:

