Deployment of Data Model on Flask

Submitted By:Mohammad Tohin Bapari
Batch No: LISUM34
Submitted To: Data Glacier
Submission Date: 26 June 2024

Content

1. Introduction
2. Pre-processing
3. Script
4. Templates
5. Conclusion

1. Introduction

Objective: To deploy a machine learning model using Flask, create a web application for predicting income based on user inputs, and document the deployment process.

1.1Task Overview:

- **Data Selection:** Utilize a simplified dataset for predicting income levels based on demographic and work-related attributes.
- **Model Training:** Train a Decision Tree Classifier on the dataset to predict whether an individual earns more than or less than $50,000 annually.
- **Flask Web Application:** Develop a web application using Flask framework to interactively predict income based on user-inputted features.
- **Documentation:** Create a comprehensive PDF document capturing each step of the deployment process, including model training, web app development, and deployment on a local server.
- **Submission:** Upload the PDF document to GitHub for sharing and evaluation.

1.2 Tools and Technologies Used:

- Python (scikit-learn, pandas, Flask)
- HTML/CSS for front-end design
- GitHub for version control and documentation storage

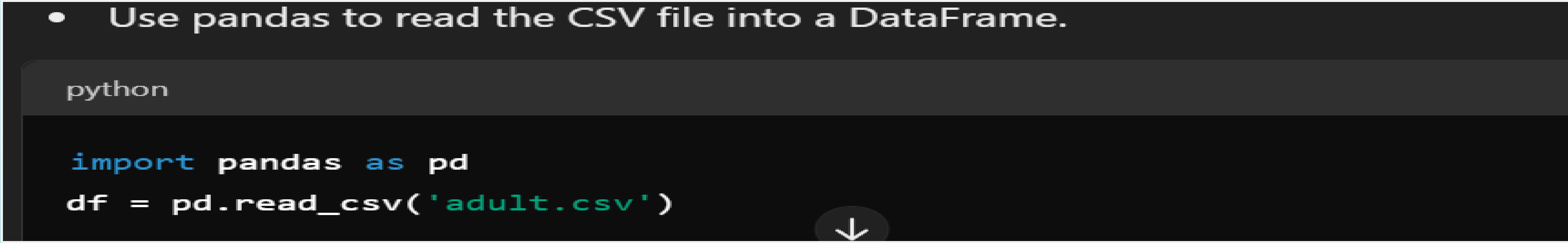
1.3 Benefits:

- Enables interactive prediction of income levels through a user-friendly web interface.
- Demonstrates proficiency in machine learning model deployment and documentation.
- Enhances understanding of integrating machine learning models into real-world applications.

2. Pre-processing

- Objective: Prepare the dataset for training a machine learning model to predict income levels.

- **2.1 Loading the Dataset:**

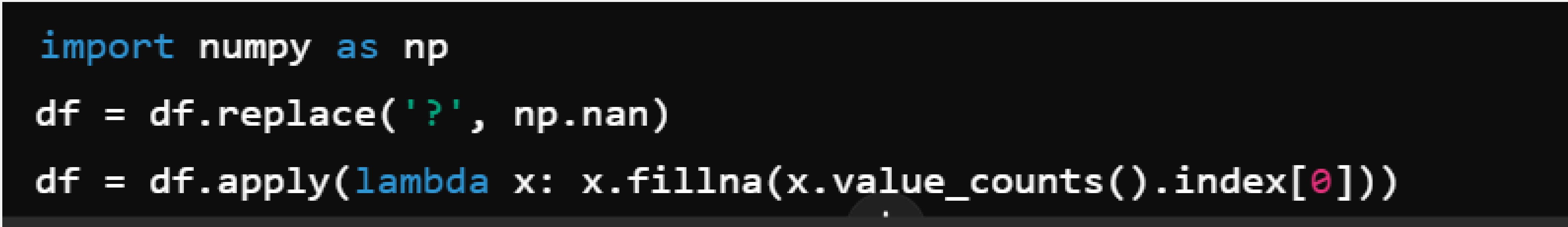


- Use pandas to read the CSV file into a DataFrame.

python

```python
import pandas as pd
df = pd.read_csv('adult.csv')
```

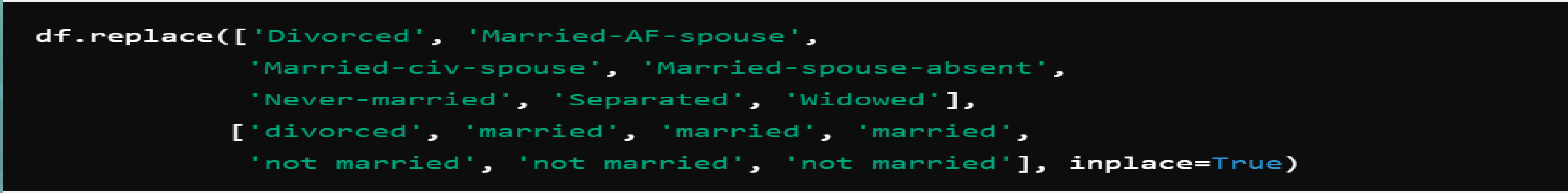
- **2.2 Handling Missing Values**
  - Replace '?' with NaN.
  - Fill missing values with the most frequent value in each column.



```python
import numpy as np
df = df.replace('?', np.nan)
df = df.apply(lambda x: x.fillna(x.value_counts().index[0]))
```

2.3 Discretizing Categorical Data

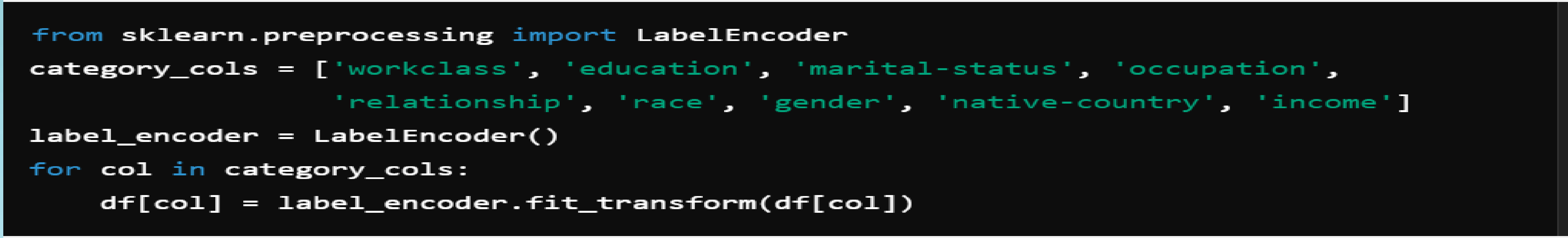
- Standardize marital status categories.



```
df.replace(['Divorced', 'Married-AF-spouse',
            'Married-civ-spouse', 'Married-spouse-absent',
            'Never-married', 'Separated', 'Widowed'],
           ['divorced', 'married', 'married', 'married',
            'not married', 'not married', 'not married'], inplace=True)
```

2.4 Encoding Categorical Variables

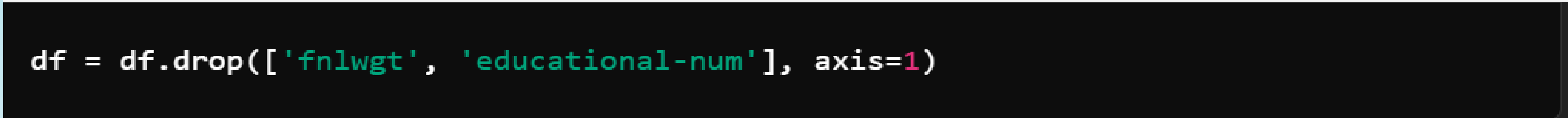
- Use Label Encoder from sklearn to transform categorical variables into numerical values.



```
from sklearn.preprocessing import LabelEncoder
category_cols = ['workclass', 'education', 'marital-status', 'occupation',
                 'relationship', 'race', 'gender', 'native-country', 'income']
label_encoder = LabelEncoder()
for col in category_cols:
    df[col] = label_encoder.fit_transform(df[col])
```

2.5 Feature Selection

- Drop redundant columns not contributing to the prediction task.

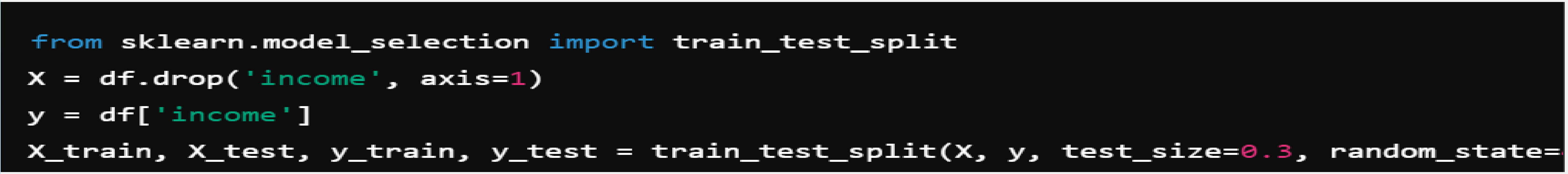


```
df = df.drop(['fnlwgt', 'educational-num'], axis=1)
```

2.6 Training and Testing Split

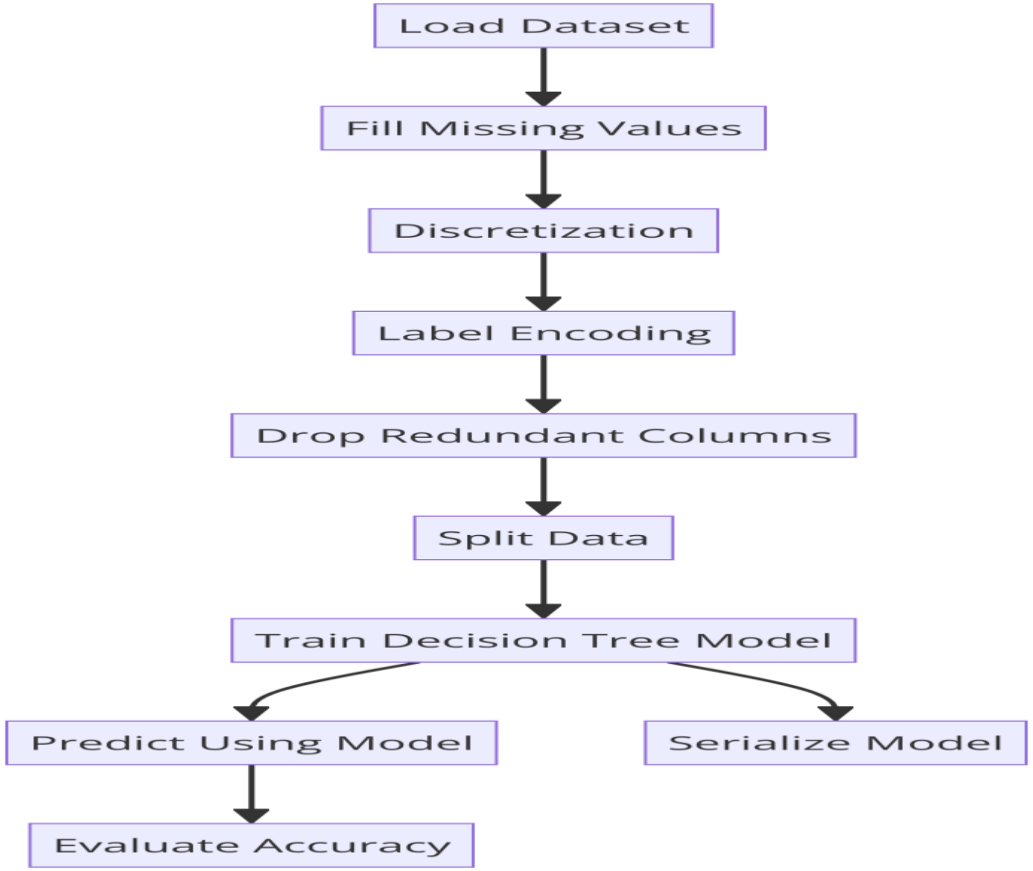
- Split the dataset into training and testing sets

.



```
from sklearn.model_selection import train_test_split
X = df.drop('income', axis=1)
y = df['income']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=
```

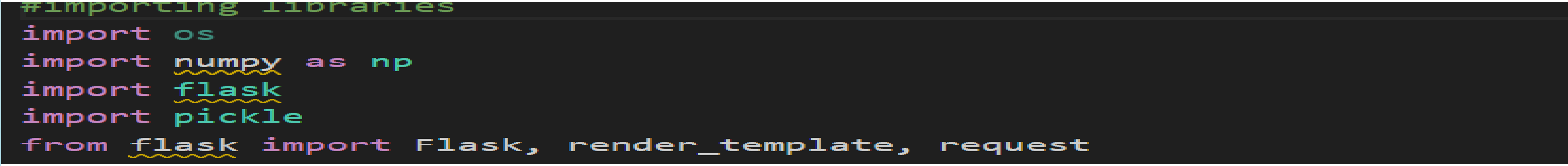
2.7 Flowchart



3. Script

Objective: Deploy a machine learning model using Flask for predicting income based on user inputs.

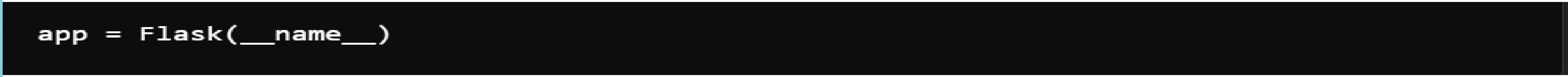
3.1 Importing Libraries



```
#importing libraries
import os
import numpy as np
import flask
import pickle
from flask import Flask, render_template, request
```

3.2 Creating Flask App

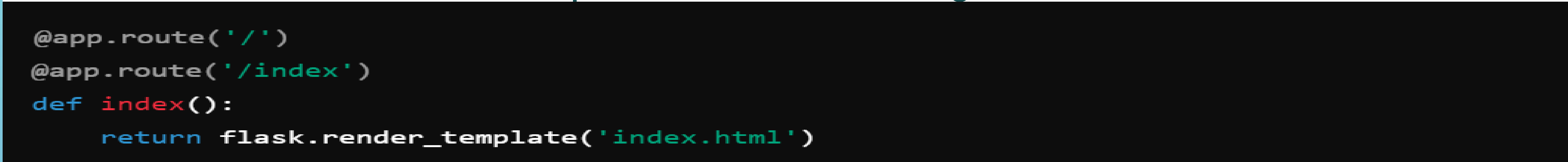
- Initialize a Flask application.
- Define routes for handling different URLs.



```
app = Flask(__name__)
```

3.3 Route for Index

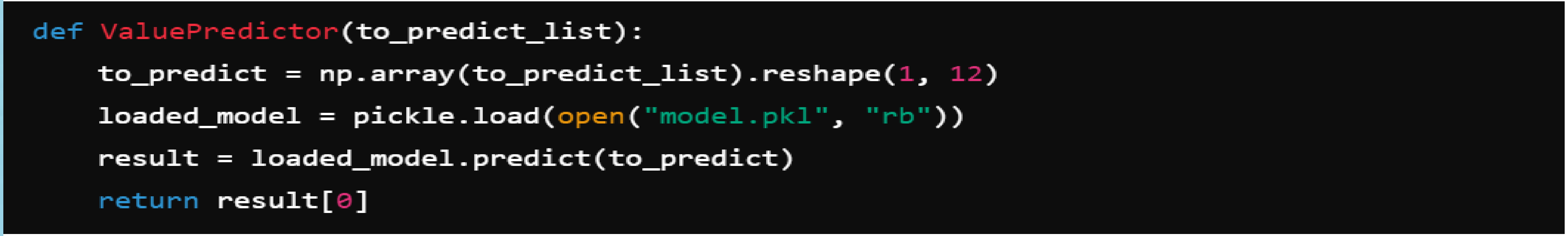
- Renders index.html template when accessing / or /index.



```
@app.route('/')
@app.route('/index')
def index():
    return flask.render_template('index.html')
```

3.4 Prediction Function:

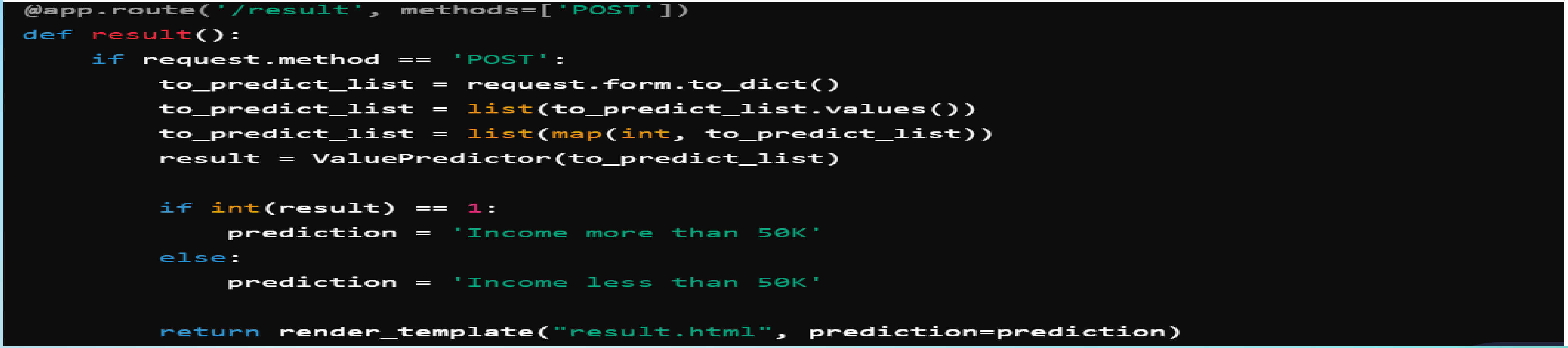
- Loads the trained model (model.pkl).
- Uses model to predict income level based on input features.



```
def ValuePredictor(to_predict_list):
    to_predict = np.array(to_predict_list).reshape(1, 12)
    loaded_model = pickle.load(open("model.pkl", "rb"))
    result = loaded_model.predict(to_predict)
    return result[0]
```

3.5 Handling POST Request

- Receives data from a form submission via POST method.
- Calls Value Predictor function to get prediction based on user input.
- Returns prediction result to result.html.



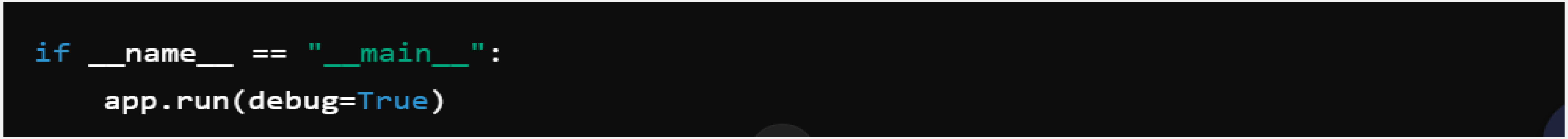
```
@app.route('/result', methods=['POST'])
def result():
    if request.method == 'POST':
        to_predict_list = request.form.to_dict()
        to_predict_list = list(to_predict_list.values())
        to_predict_list = list(map(int, to_predict_list))
        result = ValuePredictor(to_predict_list)

        if int(result) == 1:
            prediction = 'Income more than 50K'
        else:
            prediction = 'Income less than 50K'

        return render_template("result.html", prediction=prediction)
```

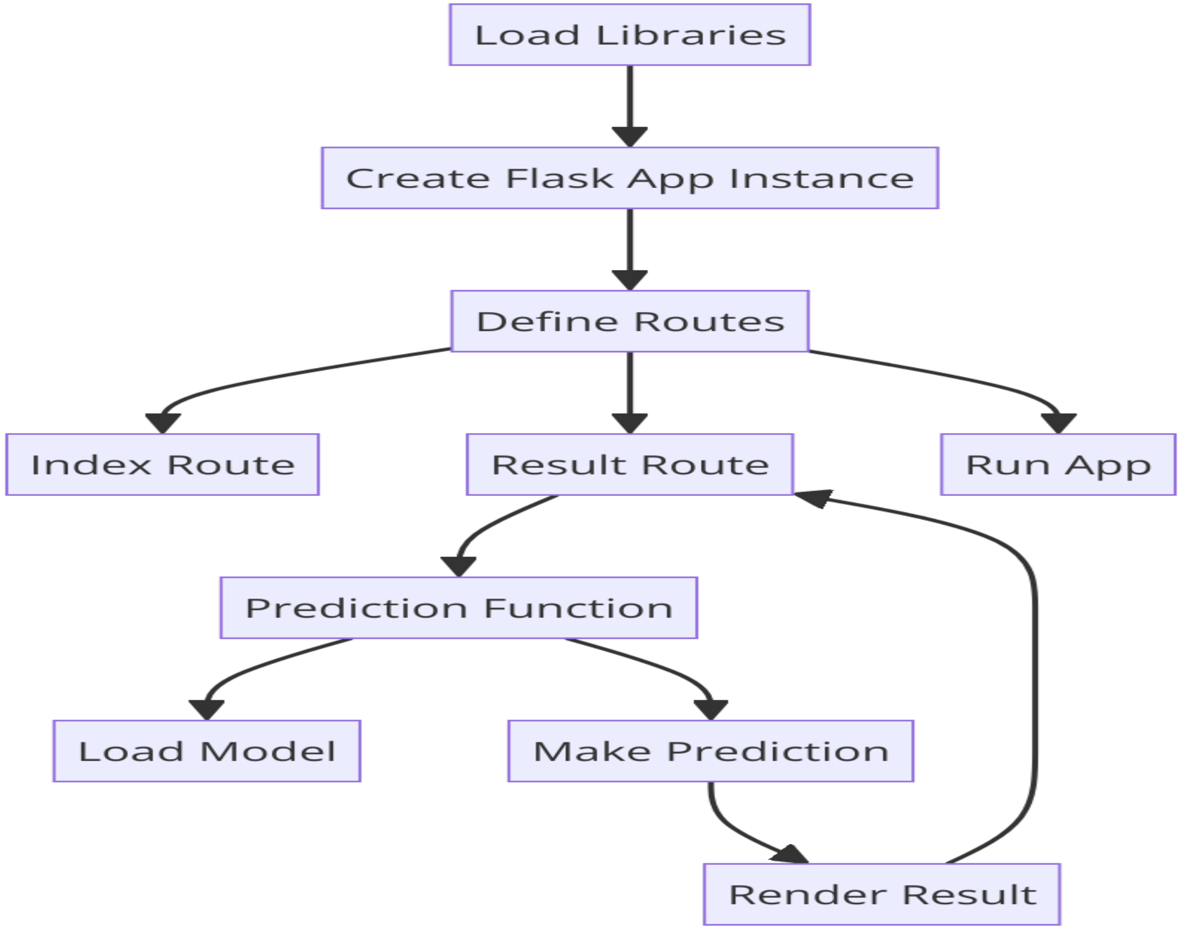
3.6 Running the Flask App

- Starts the Flask application in debug mode for development.



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

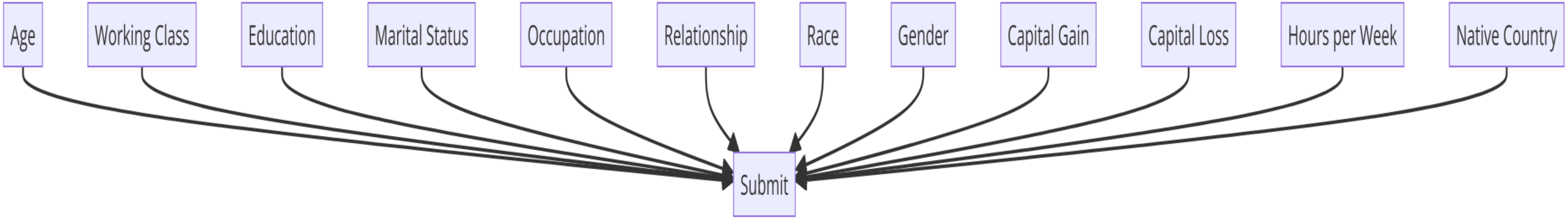
3.7. Flowchart



4. Templates

Objective: Gather user input for predicting income based on various demographic and work-related features.

4.1 index.html



Age

Working Class

Education

Marital Status

Occupation

Relationship

Race

Gender

Capital Gain

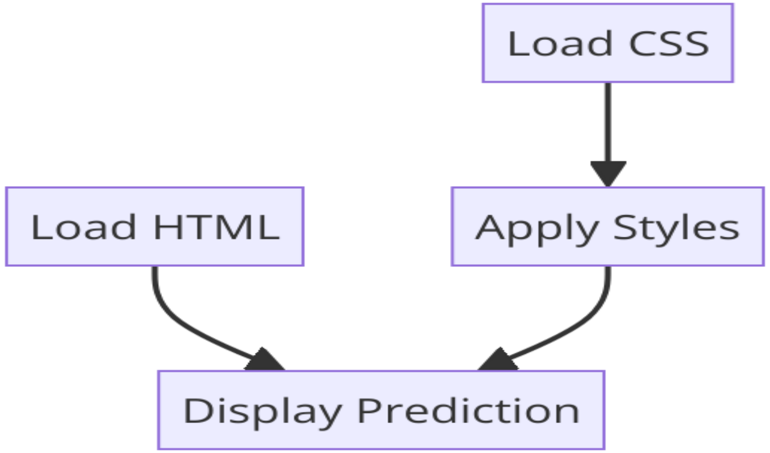
Capital Loss

Hours per Week

Native Country

Submit

4.2 result.html



Load CSS

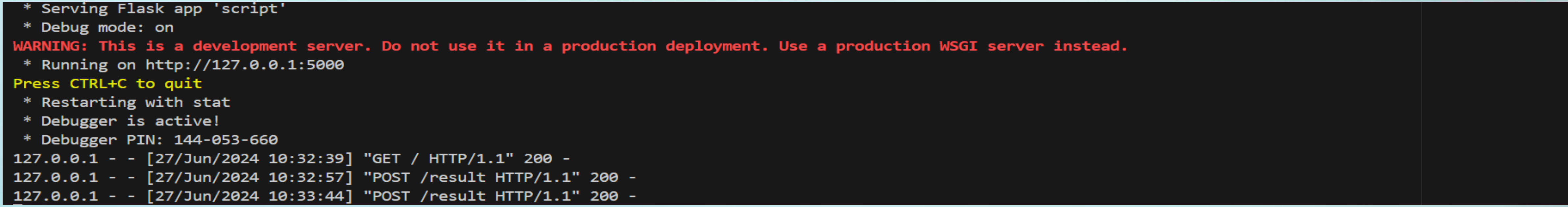
Load HTML

Apply Styles

Display Prediction

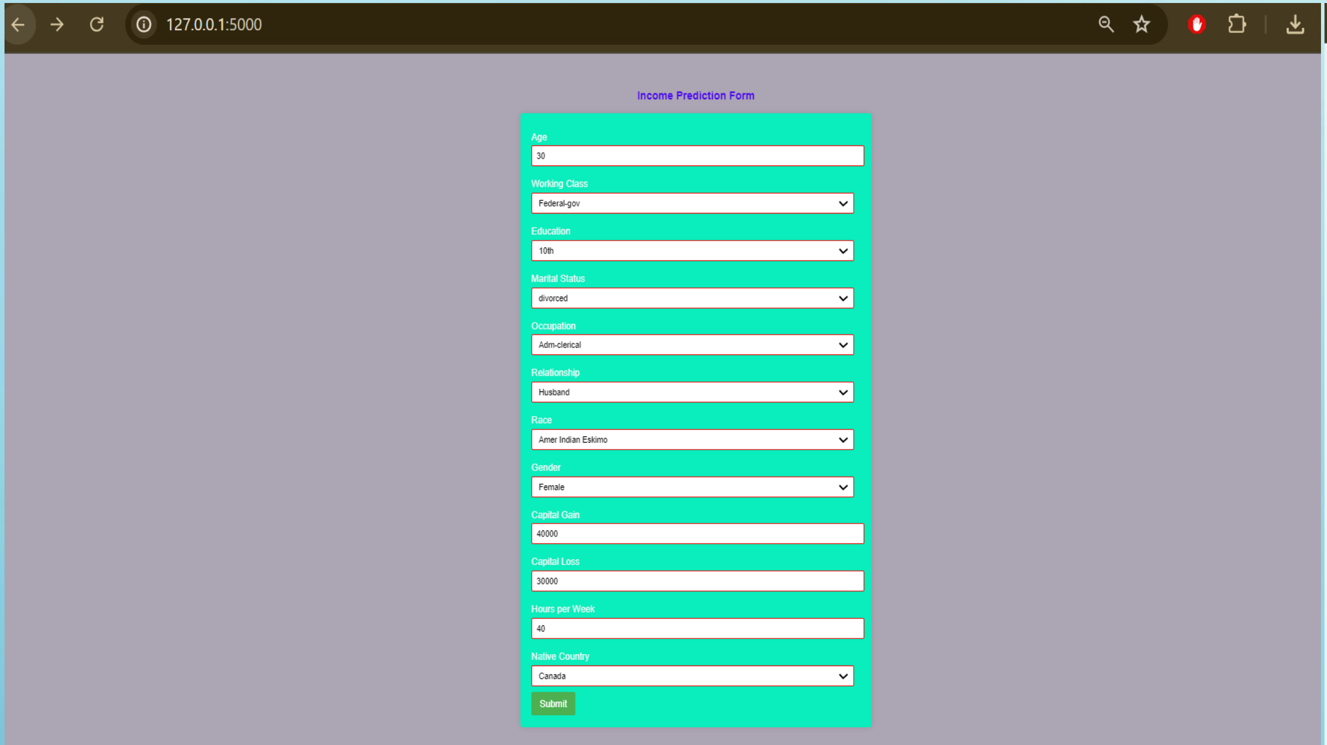
5 Conclusion

5.1 Running Flask app



```
 * Serving Flask app 'script'
 * 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: 144-053-660
127.0.0.1 - - [27/Jun/2024 10:32:39] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [27/Jun/2024 10:32:57] "POST /result HTTP/1.1" 200 -
127.0.0.1 - - [27/Jun/2024 10:33:44] "POST /result HTTP/1.1" 200 -
```

5.2 Visualization



Income Prediction Form

Age: 30

Working Class: Federal-gov

Education: 10th

Marital Status: divorced

Occupation: Adm-clerical

Relationship: Husband

Race: Amer Indian Eskimo

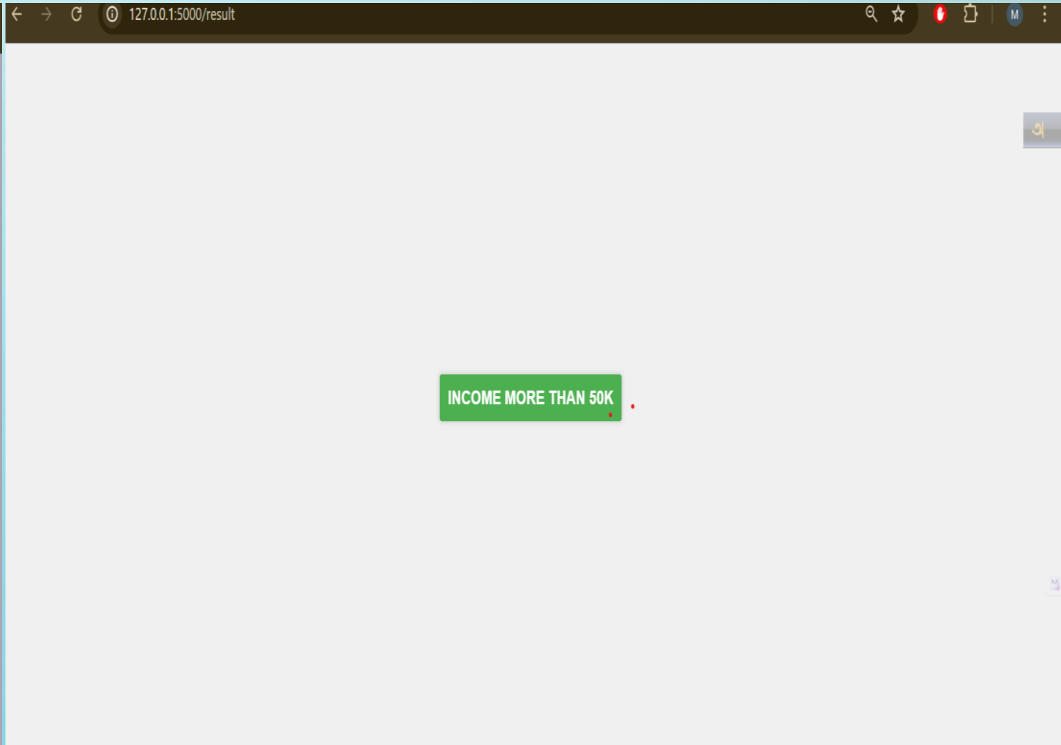
Gender: Female

Capital Gain: 40000

Capital Loss: 30000

Hours per Week: 40

Native Country: Canada



INCOME MORE THAN 50K