Name: Jackson Taylor

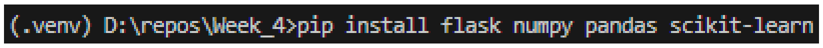
Batch code: LISUM33

Submission date: 05/27/2024

Submitted to: Data Glacier

The same virtual environment is used for model.py and app.py.

These are the installed packages.

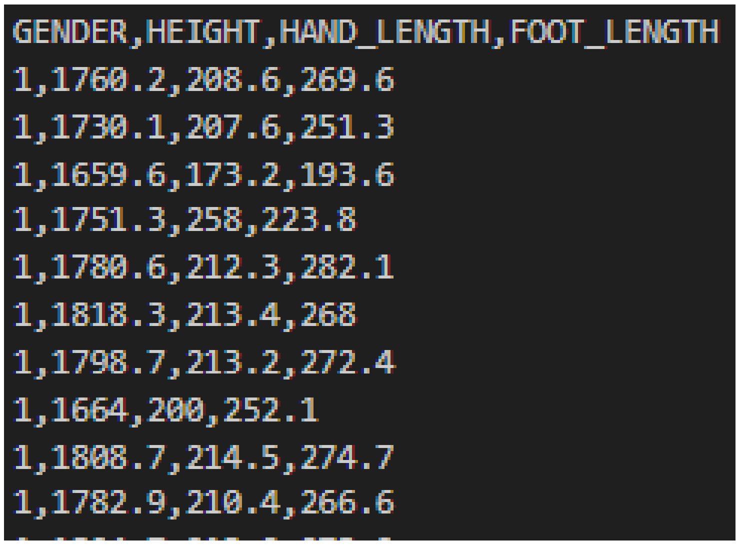


```
(.venv) D:\repos\Week_4>pip install flask numpy pandas scikit-learn
```

This is part of the dataset I used to train the model.

The model predicts gender based on height, hand length, and foot length in millimeters.

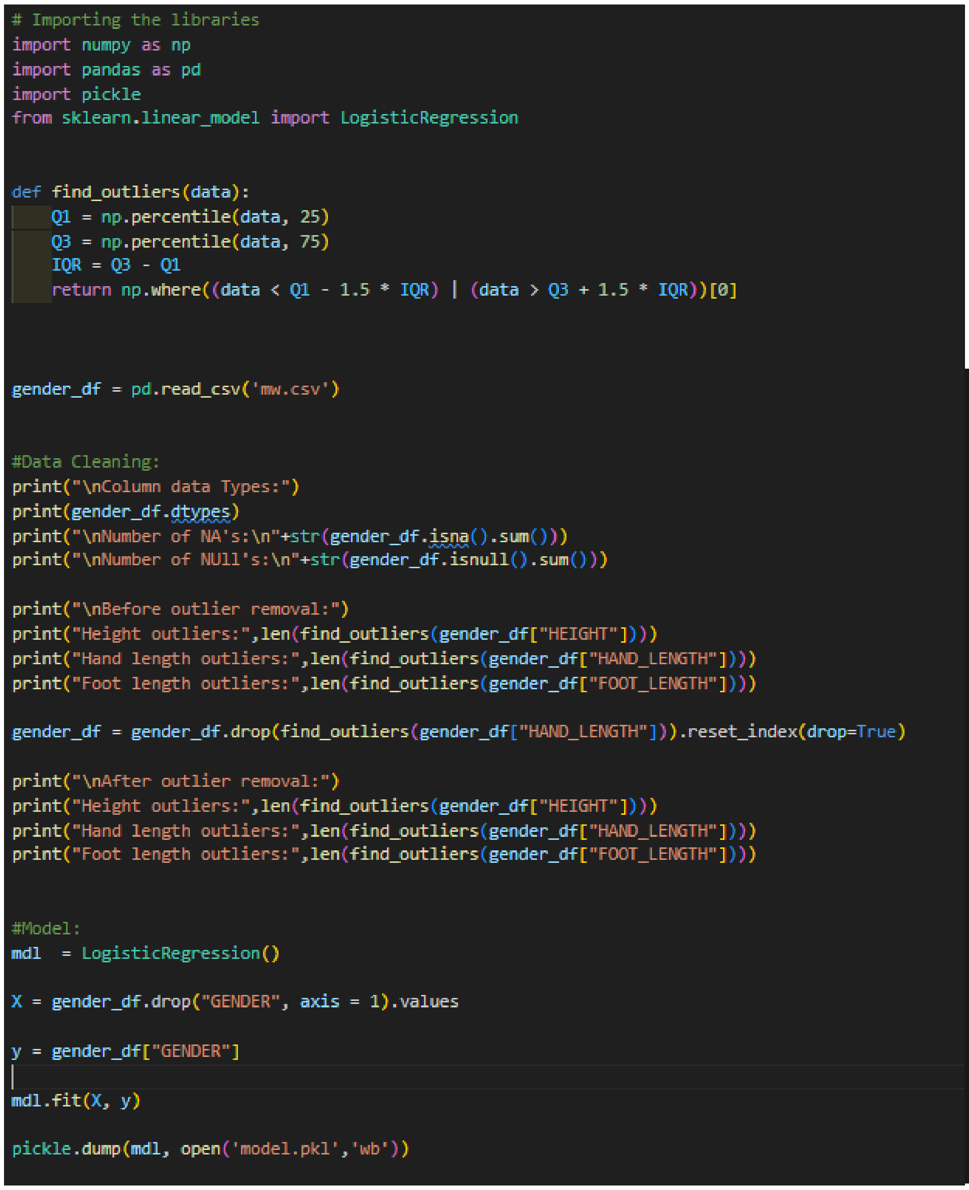
Gender column: 1 represents male and 2 represents female.



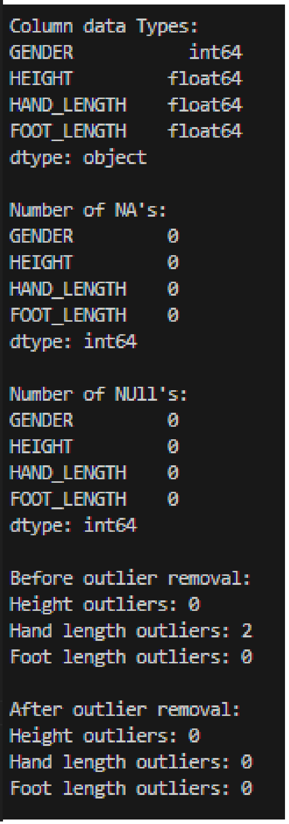
```
GENDER,HEIGHT,HAND_LENGTH,FOOT_LENGTH
1,1760.2,208.6,269.6
1,1730.1,207.6,251.3
1,1659.6,173.2,193.6
1,1751.3,258,223.8
1,1780.6,212.3,282.1
1,1818.3,213.4,268
1,1798.7,213.2,272.4
1,1664,200,252.1
1,1808.7,214.5,274.7
1,1782.9,210.4,266.6
```

This is the model.py code that trains and saves the gender predicting model.

There is also an output for the data cleaning.

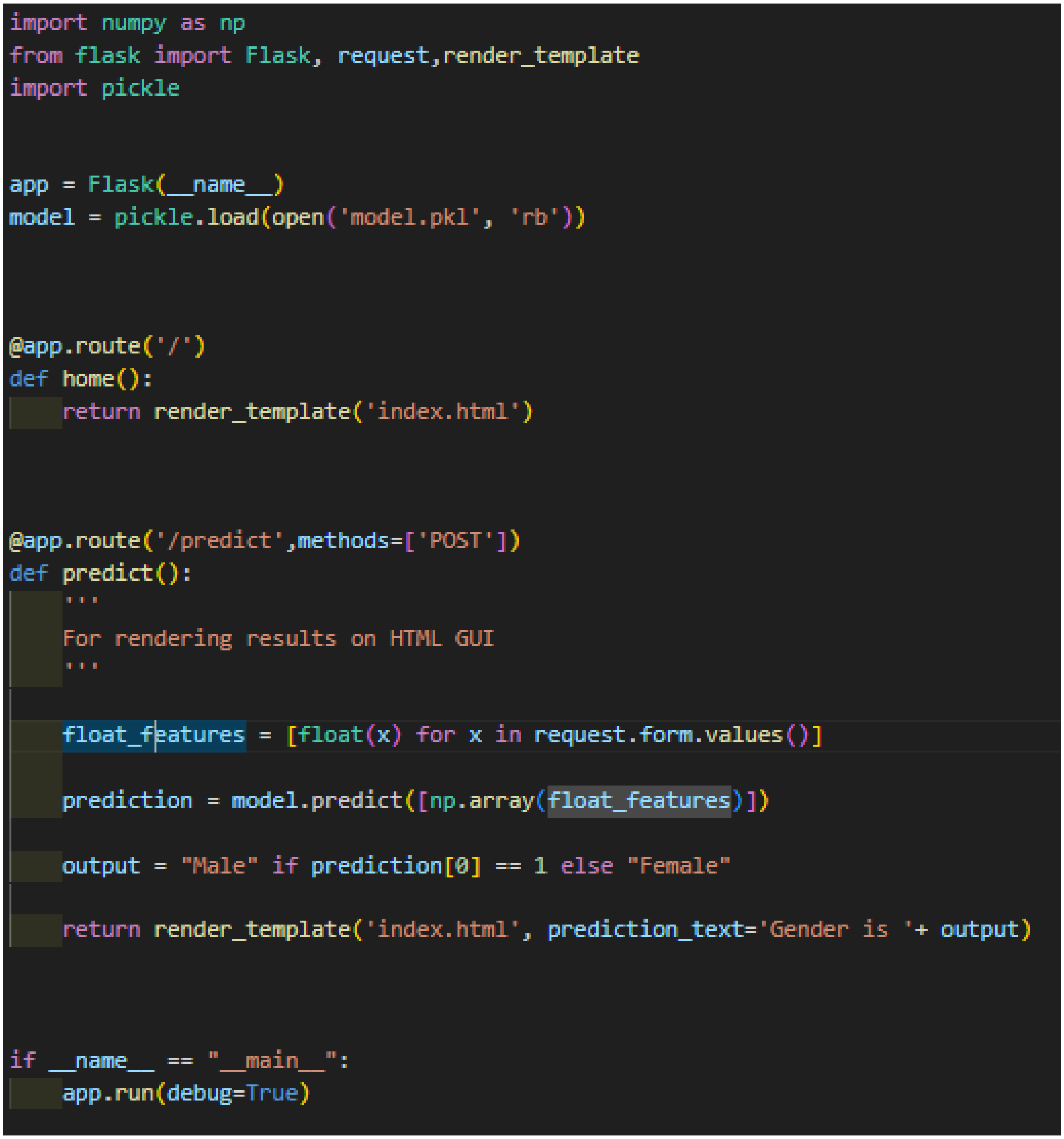


```python
# Importing the libraries
import numpy as np
import pandas as pd
import pickle
from sklearn.linear_model import LogisticRegression


def find_outliers(data):
    Q1 = np.percentile(data, 25)
    Q3 = np.percentile(data, 75)
    IQR = Q3 - Q1
    return np.where((data < Q1 - 1.5 * IQR) | (data > Q3 + 1.5 * IQR))[0]



gender_df = pd.read_csv('mw.csv')


#Data Cleaning:
print("\nColumn data Types:")
print(gender_df.dtypes)
print("\nNumber of NA's:\n"+str(gender_df.isna().sum()))
print("\nNumber of NUll's:\n"+str(gender_df.isnull().sum()))

print("\nBefore outlier removal:")
print("Height outliers:",len(find_outliers(gender_df["HEIGHT"])))
print("Hand length outliers:",len(find_outliers(gender_df["HAND_LENGTH"])))
print("Foot length outliers:",len(find_outliers(gender_df["FOOT_LENGTH"])))

gender_df = gender_df.drop(find_outliers(gender_df["HAND_LENGTH"])).reset_index(drop=True)

print("\nAfter outlier removal:")
print("Height outliers:",len(find_outliers(gender_df["HEIGHT"])))
print("Hand length outliers:",len(find_outliers(gender_df["HAND_LENGTH"])))
print("Foot length outliers:",len(find_outliers(gender_df["FOOT_LENGTH"])))


#Model:
mdl  = LogisticRegression()

X = gender_df.drop("GENDER", axis = 1).values

y = gender_df["GENDER"]

mdl.fit(X, y)

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



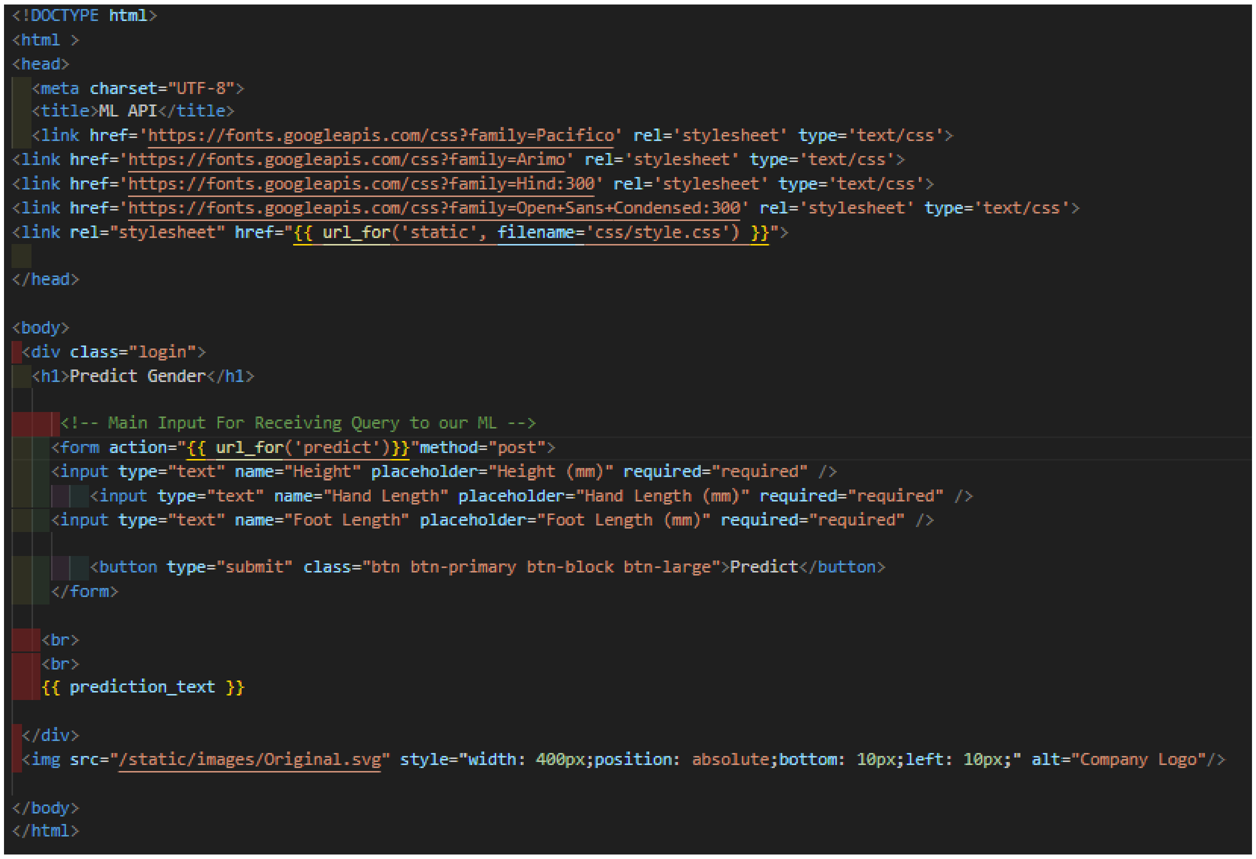
```
Column data Types:
GENDER           int64
HEIGHT         float64
HAND_LENGTH    float64
FOOT_LENGTH    float64
dtype: object

Number of NA's:
GENDER         0
HEIGHT         0
HAND_LENGTH    0
FOOT_LENGTH    0
dtype: int64

Number of NUll's:
GENDER         0
HEIGHT         0
HAND_LENGTH    0
FOOT_LENGTH    0
dtype: int64

Before outlier removal:
Height outliers: 0
Hand length outliers: 2
Foot length outliers: 0

After outlier removal:
Height outliers: 0
Hand length outliers: 0
Foot length outliers: 0
```

This is the Flask code (app.py). It receives a request upon the form submission on the home page. The three form inputs are used by the trained model to make a gender prediction. Then it renders the original home page with a text that includes the output gender prediction.



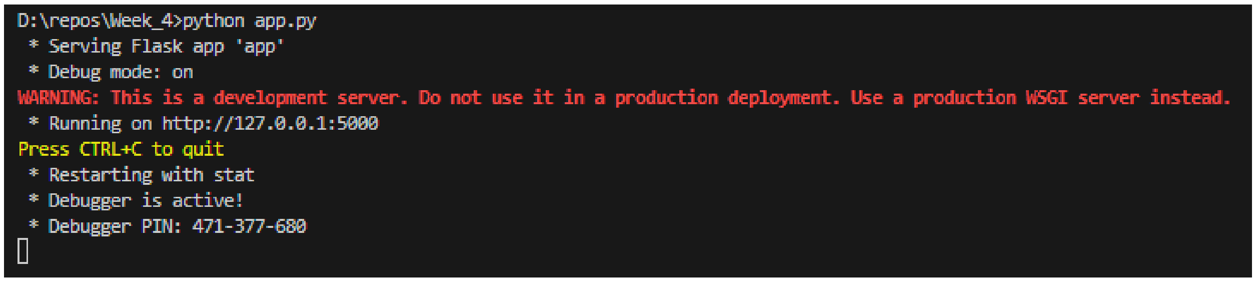
```python
import numpy as np
from flask import Flask, request,render_template
import pickle


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



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



@app.route('/predict',methods=['POST'])
def predict():
    '''
    For rendering results on HTML GUI
    '''

    float_features = [float(x) for x in request.form.values()]

    prediction = model.predict([np.array(float_features)])

    output = "Male" if prediction[0] == 1 else "Female"

    return render_template('index.html', prediction_text='Gender is '+ output)



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

This is the index.html code with the form modified so that the inputs correspond to the feature inputs of my model.



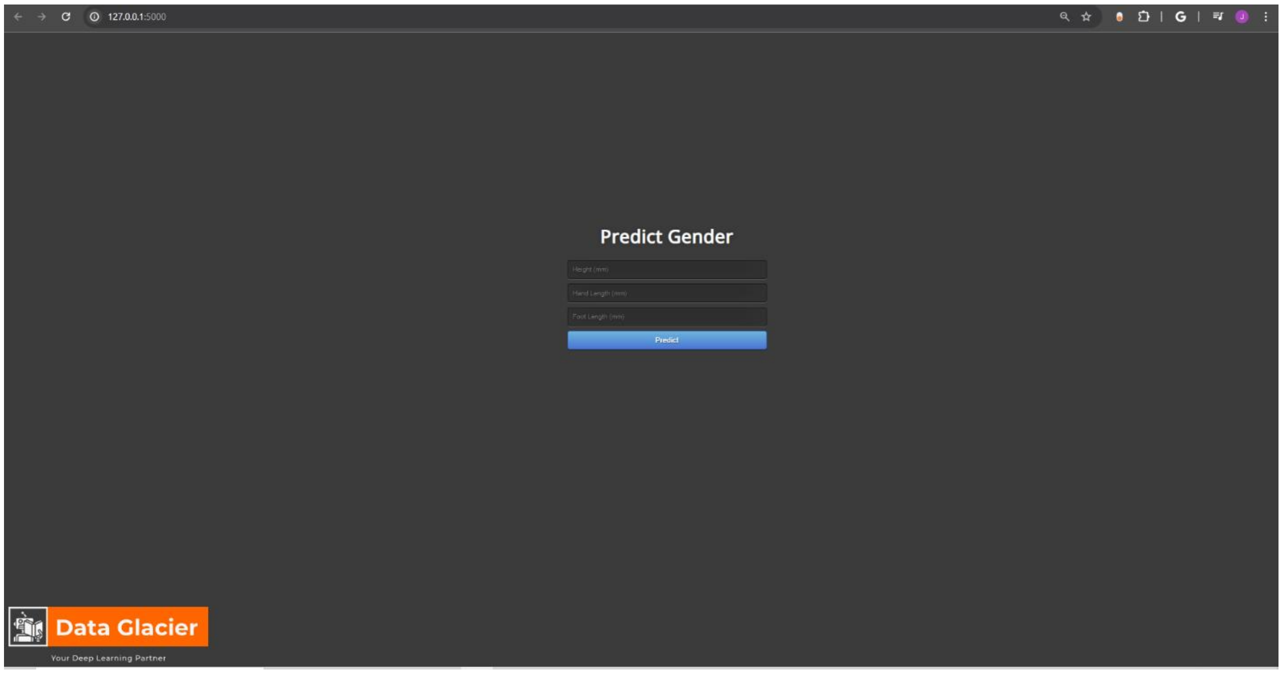
```html
<!DOCTYPE html>
<html >
<head>
  <meta charset="UTF-8">
  <title>ML API</title>
  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet' type='text/css'>
<link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">

</head>

<body>
 <div class="login">
  <h1>Predict Gender</h1>

     <!-- Main Input For Receiving Query to our ML -->
    <form action="{{ url_for('predict')}}"method="post">
    <input type="text" name="Height" placeholder="Height (mm)" required="required" />
        <input type="text" name="Hand Length" placeholder="Hand Length (mm)" required="required" />
    <input type="text" name="Foot Length" placeholder="Foot Length (mm)" required="required" />

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

   <br>
   <br>
   {{ prediction_text }}

 </div>
 <img src="/static/images/Original.svg" style="width: 400px;position: absolute;bottom: 10px;left: 10px;" alt="Company Logo"/>

</body>
</html>
```

Console output when running app.py:

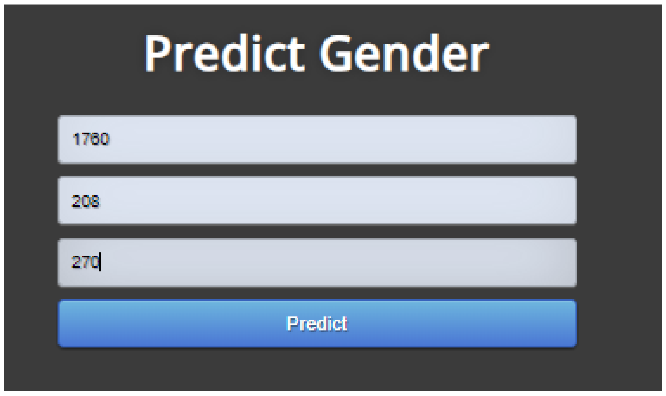


```
D:\repos\Week_4>python app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 471-377-680
```

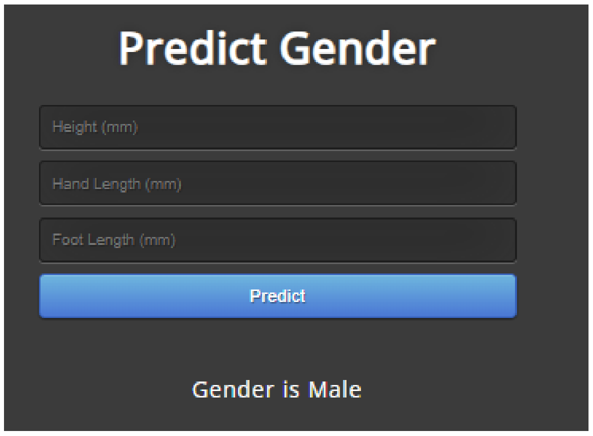
Full home page on the local host:



Form Completed:



Output after the form was submitted (pressing predict button) with the inputs above:



Gender is Male