Data Glacier

Data Science Intern at Data Glacier

Week 5: Cloud and API Deployment

Name: Taiwo Akingbesote

Batch Code: LISUM22

Date: 4 July 2023

Submitted to: Data Glacier

Contents:

1. Introduction
2. Data Information
3. Build Machine Learning Model
4. Turning Model into Flask Framework
5. Running the package
6. Deploying model on Heroku

1. Introduction

In this project, we are going to deploy a machine learning model (SVM) using the Flask Framework. As a demonstration, our model help to recommend a minimum value for a soccer player based on their age and potential. After careful examination, this model is more accurate for players aged 18-28.

We will be building a machine learning model to determine recommended player value, then create an API for the model, using Flask, the Python micro-framework for building web applications. This API requests information (age + potential) from users and gives out the recommended value.

2. Data Information

This data, obtained from Kaggle.com contains the details of 52 professional soccer player and their attributes.

	ID	Name	Age	Photo	Nationality	Flag	Overall	Poten
0	209658	L. Goretzka	27	https://cdn.sofifa.net/players/209/658/23_60.png	Germany	https://cdn.sofifa.net/flags/de.png	87	
1	212198	Bruno Fernandes	27	https://cdn.sofifa.net/players/212/198/23_60.png	Portugal	https://cdn.sofifa.net/flags/pt.png	86	
2	224334	M. Acuña	30	https://cdn.sofifa.net/players/224/334/23_60.png	Argentina	https://cdn.sofifa.net/flags/ar.png	85	
3	192985	K. De Bruyne	31	https://cdn.sofifa.net/players/192/985/23_60.png	Belgium	https://cdn.sofifa.net/flags/be.png	91	
4	224232	N. Barella	25	https://cdn.sofifa.net/players/224/232/23_60.png	Italy	https://cdn.sofifa.net/flags/it.png	86	
5	212622	J. Kimmich	27	https://cdn.sofifa.net/players/212/622/23_60.png	Germany	https://cdn.sofifa.net/flags/de.png	89	
6	197445	D. Alaba	30	https://cdn.sofifa.net/players/197/445/23_60.png	Austria	https://cdn.sofifa.net/flags/at.png	86	
7	187961	22 Paulinho	32	https://cdn.sofifa.net/players/187/961/22_60.png	Brazil	https://cdn.sofifa.net/flags/br.png	83	
8	208333	E. Can	28	https://cdn.sofifa.net/players/208/333/23_60.png	Germany	https://cdn.sofifa.net/flags/de.png	82	
9	210514	João Cancelo	28	https://cdn.sofifa.net/players/210/514/23_60.png	Portugal	https://cdn.sofifa.net/flags/pt.png	88	
10	228251	L. Pellegrini	26	https://cdn.sofifa.net/players/228/251/23_60.png	Italy	https://cdn.sofifa.net/flags/it.png	84	
11	177003	L. Modrić	36	https://cdn.sofifa.net/players/177/003/23_60.png	Croatia	https://cdn.sofifa.net/flags/hr.png	88	
12	223848	S. Milinković-Savić	27	https://cdn.sofifa.net/players/223/848/23_60.png	Serbia	https://cdn.sofifa.net/flags/rs.png	86	
13	225193	Merino	26	https://cdn.sofifa.net/players/225/193/23_60.png	Spain	https://cdn.sofifa.net/flags/es.png	83	
14	226161	Marcos Llorente	27	https://cdn.sofifa.net/players/226/161/23_60.png	Spain	https://cdn.sofifa.net/flags/es.png	84	
15	212616	R. De Paul	28	https://cdn.sofifa.net/players/212/616/23_60.png	Argentina	https://cdn.sofifa.net/flags/ar.png	84	
16	181458	I. Perišić	33	https://cdn.sofifa.net/players/181/458/23_60.png	Croatia	https://cdn.sofifa.net/flags/hr.png	84	
17	228702	F. de Jong	25	https://cdn.sofifa.net/players/228/702/23_60.png	Netherlands	https://cdn.sofifa.net/flags/nl.png	87	
18	194765	A. Griezmann	31	https://cdn.sofifa.net/players/194/765/23_60.png	France	https://cdn.sofifa.net/flags/fr.png	83	
19	193082	J. Cuadrado	34	https://cdn.sofifa.net/players/193/082/23_60.png	Colombia	https://cdn.sofifa.net/flags/co.png	83	
20	208574	F. Kostić	29	https://cdn.sofifa.net/players/208/574/23_60.png	Serbia	https://cdn.sofifa.net/flags/rs.png	85	
21	231281	T. Alexander-Arnold	23	https://cdn.sofifa.net/players/231/281/23_60.png	England	https://cdn.sofifa.net/flags/gb-eng.png	87	
22	176580	L. Suárez	35	https://cdn.sofifa.net/players/176/580/23_60.png	Uruguay	https://cdn.sofifa.net/flags/uy.png	84	
23	235212	A. Hakimi	23	https://cdn.sofifa.net/players/235/212/23_60.png	Morocco	https://cdn.sofifa.net/flags/ma.png	84	
24	216352	M. Brozović	29	https://cdn.sofifa.net/players/216/352/23_60.png	Croatia	https://cdn.sofifa.net/flags/hr.png	86	
25	209331	M. Salah	30	https://cdn.sofifa.net/players/209/331/23_60.png	Egypt	Numbers fa.net/flags/eg.png	90	

Potential	Club	Club Logo	Value(£)	Wage(£)	Special	Preferred Foot	International Reputation	We
88	FC Bayern München	https://cdn.sofifa.net/teams/21/30.png	91000000.0	115000.0	2312	Right	4.0	
87	Manchester United	https://cdn.sofifa.net/teams/11/30.png	78500000.0	190000.0	2305	Right	3.0	
85	Sevilla FC	https://cdn.sofifa.net/teams/481/30.png	46500000.0	46000.0	2303	Left	2.0	
91	Manchester City	https://cdn.sofifa.net/teams/10/30.png	107500000.0	350000.0	2303	Right	4.0	
89	Inter	https://cdn.sofifa.net/teams/44/30.png	89500000.0	110000.0	2296	Right	3.0	
90	FC Bayern München	https://cdn.sofifa.net/teams/21/30.png	105500000.0	130000.0	2283	Right	4.0	
86	Real Madrid CF	https://cdn.sofifa.net/teams/243/30.png	55500000.0	220000.0	2277	Left	4.0	
83	Al Ahli	https://cdn.sofifa.net/teams/112387/30.png	28500000.0	61000.0	2273	Right	3.0	
82	Borussia Dortmund	https://cdn.sofifa.net/teams/22/30.png	30500000.0	63000.0	2271	Right	3.0	
88	Manchester City	https://cdn.sofifa.net/teams/10/30.png	82500000.0	250000.0	2262	Right	3.0	
87	Roma	https://cdn.sofifa.net/teams/52/30.png	55000000.0	95000.0	2258	Right	2.0	
88	Real Madrid CF	https://cdn.sofifa.net/teams/243/30.png	29000000.0	230000.0	2257	Right	4.0	
87	Lazio	https://cdn.sofifa.net/teams/46/30.png	77500000.0	90000.0	2250	Right	3.0	
86	Real Sociedad	https://cdn.sofifa.net/teams/457/30.png	47000000.0	49000.0	2247	Left	1.0	
85	Atlético de Madrid	https://cdn.sofifa.net/teams/240/30.png	48000000.0	81000.0	2244	Right	3.0	
84	Atlético de Madrid	https://cdn.sofifa.net/teams/240/30.png	42000000.0	85000.0	2240	Right	1.0	
84	Tottenham Hotspur	https://cdn.sofifa.net/teams/18/30.png	26500000.0	130000.0	2238	Right	3.0	
92	FC Barcelona	https://cdn.sofifa.net/teams/241/30.png	116500000.0	230000.0	2238	Right	3.0	
83	Atlético de Madrid	https://cdn.sofifa.net/teams/240/30.png	30500000.0	185000.0	2235	Left	4.0	
83	Juventus	https://cdn.sofifa.net/teams/45/30.png	13500000.0	120000.0	2234	Right	3.0	
85	Eintracht Frankfurt	https://cdn.sofifa.net/teams/1824/30.png	53000000.0	56000.0	2234	Left	3.0	
90	Liverpool	https://cdn.sofifa.net/teams/9/30.png	100500000.0	150000.0	2234	Right	3.0	
84	Club Nacional de Football	https://cdn.sofifa.net/teams/111325/30.png	18000000.0	1000.0	2231	Right	4.0	
87	Paris Saint-Germain	https://cdn.sofifa.net/teams/73/30.png	53500000.0	90000.0	2229	Right	2.0	
86	Inter	https://cdn.sofifa.net/teams/44/30.png	58000000.0	110000.0	2227	Right	3.0	
90	Liverpool	https://cdn.sofifa.net/teams/9/30.png	115500000.0	270000.0	2226	Left	4.0	

Figure 2.1: Dataset Information

3. Building a Model

3.1.1 Importing Required Libraries and Dataset

In this part, we import libraires and dataset which contains information of the players

```
In [39]: import numpy as np
         import pandas as pd
         import pickle
         from scipy.optimize import curve_fit
         import matplotlib.pyplot as plt
         from sklearn.linear_model import LinearRegression

In [ ]: # Load the FIFA23 dataset
        df = pd.read_csv('FIFA23.csv')
```

Figure 3.1.1

3.1.2 Data Manipulating

To ensure more accuracy for our model, we filter out players ranging from age 17 (the least age in the data) – 28. We also filtered out players with values < 0. These measures ensure a more accurate recommendation from the model.

```
In [130]: # Calculate the sum of Age and Potential to create a new feature
filtered_df['Age_potential'] = filtered_df['Age'] + filtered_df['Potential']

# Prepare the data for exponential curve fit
X = filtered_df['Age_potential'].values.reshape(-1,1)    # x, the addition of both age and potential
y = filtered_df['Value(£)'].values                       # y, the result, the calue of the corresponding x
```

Figure 3.1.2

3.1.3 Building the Model

After data manipulation, we implemented linear regression on the new dataset from Scikit-learn. After this step, we generated the predicted values

```
In [ ]: # Perform linear regression on the adjusted data
regressor = LinearRegression()
regressor.fit(X, y)

# Generate the predicted values
y_pred = regressor.predict(X)
```

Figure 3.1.3

3.1.4 Visualizing the Model

After successfully building the model, we formulated a graph to better understand this data and check for any visual trends.

```
In [132]: # Plotting the original data and the adjusted linear regression line
plt.scatter(X, y, label='Adjusted Data')
plt.plot(X, y_pred, color='r', label='Linear Regression')

# Adding labels and legend to the plot
plt.xlabel('Age')
plt.ylabel('Value (£)')
plt.legend()

# Display the plot
plt.show()
```

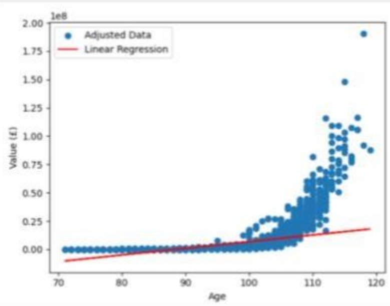


Figure3.1.4

Figure 3.1.1

3.1.5 Saving the Model

After that we save our model using pickle

Figure 3.1.5

4. Turning Model into Flask Framework

To properly deploy this model, we've created a framework on pycharm which request users age + potential and recommends a value. The files are as follows;

4.1 App.py

The app.py file contains the main code that will be executed by the Python interpreter to run the Flask web application, it included the ML code for classifying SD.

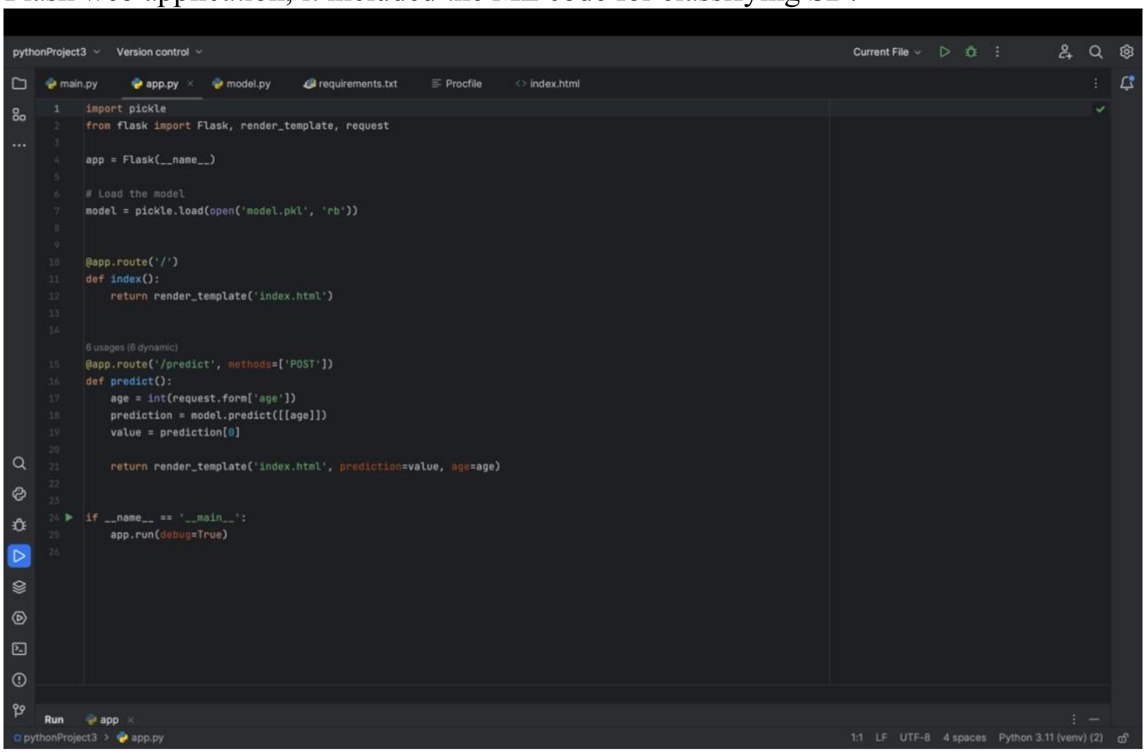


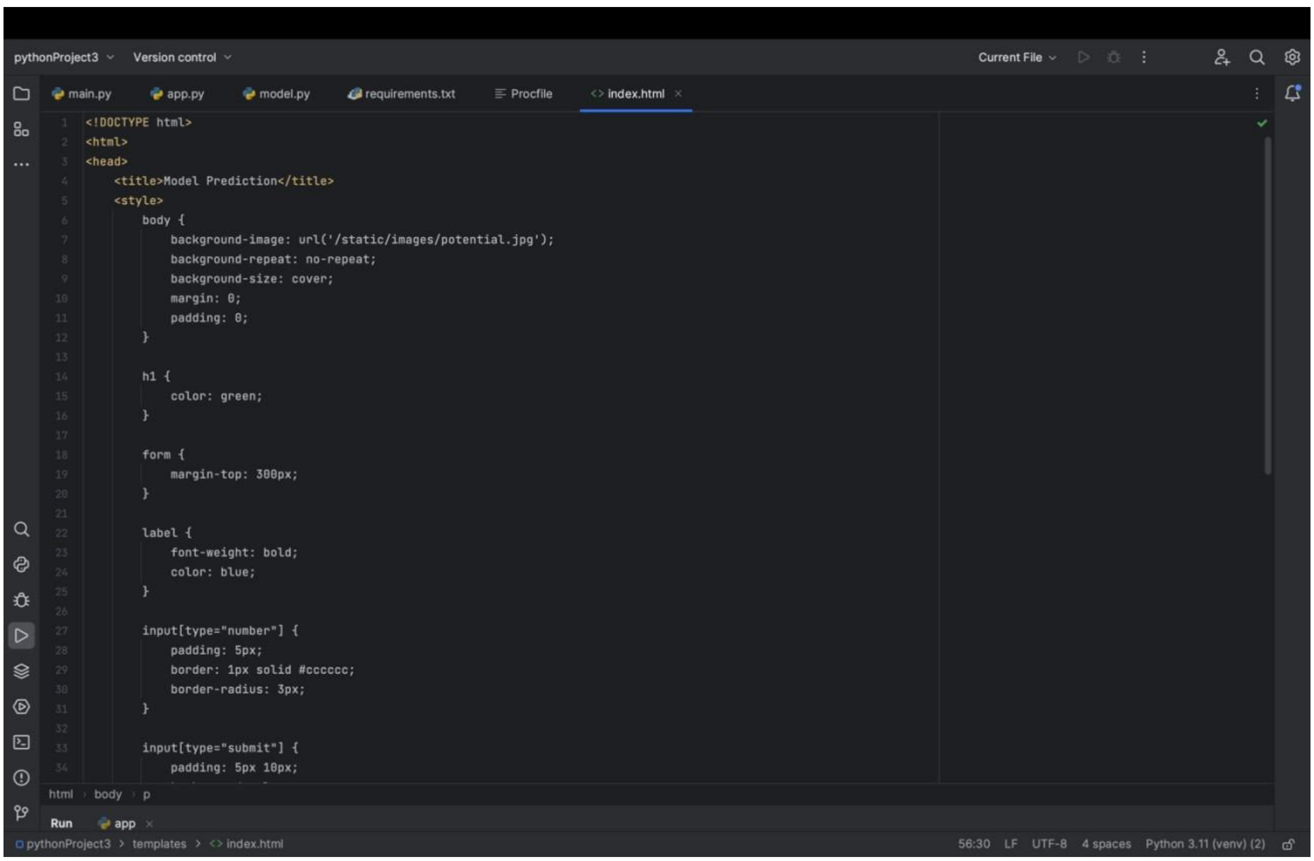
Figure 4.1: App.py

- We ran our application as a single module; thus we initialized a new Flask instance with the argument *__name__* to let Flask know that it can find the index.html template folder (*templates*) in the same directory where it is located.
- Next, we used the route decorator (*@app.route('/')*) to specify the URL that should trigger the execution of the home function.

- Our *home* function simply rendered the *index.html* HTML file, which is in the *templates* folder.
- Inside the *predict* function, we request users age + potential the deploy the model to execute the collected data.
- Lastly, we used the *run* function to only run the application on the server when this script is directly executed by the Python interpreter, which we ensured using the *if* statement with __name__ == '__main__'.

4.2 Index.html

The following are the contents of the index.html file that will render a text form where a user can enter a message.



```
<!DOCTYPE html>
<html>
<head>
    <title>Model Prediction</title>
    <style>
        body {
            background-image: url('/static/images/potential.jpg');
            background-repeat: no-repeat;
            background-size: cover;
            margin: 0;
            padding: 0;
        }

        h1 {
            color: green;
        }

        form {
            margin-top: 300px;
        }

        label {
            font-weight: bold;
            color: blue;
        }

        input[type="number"] {
            padding: 5px;
            border: 1px solid #cccccc;
            border-radius: 3px;
        }

        input[type="submit"] {
            padding: 5px 10px;
```

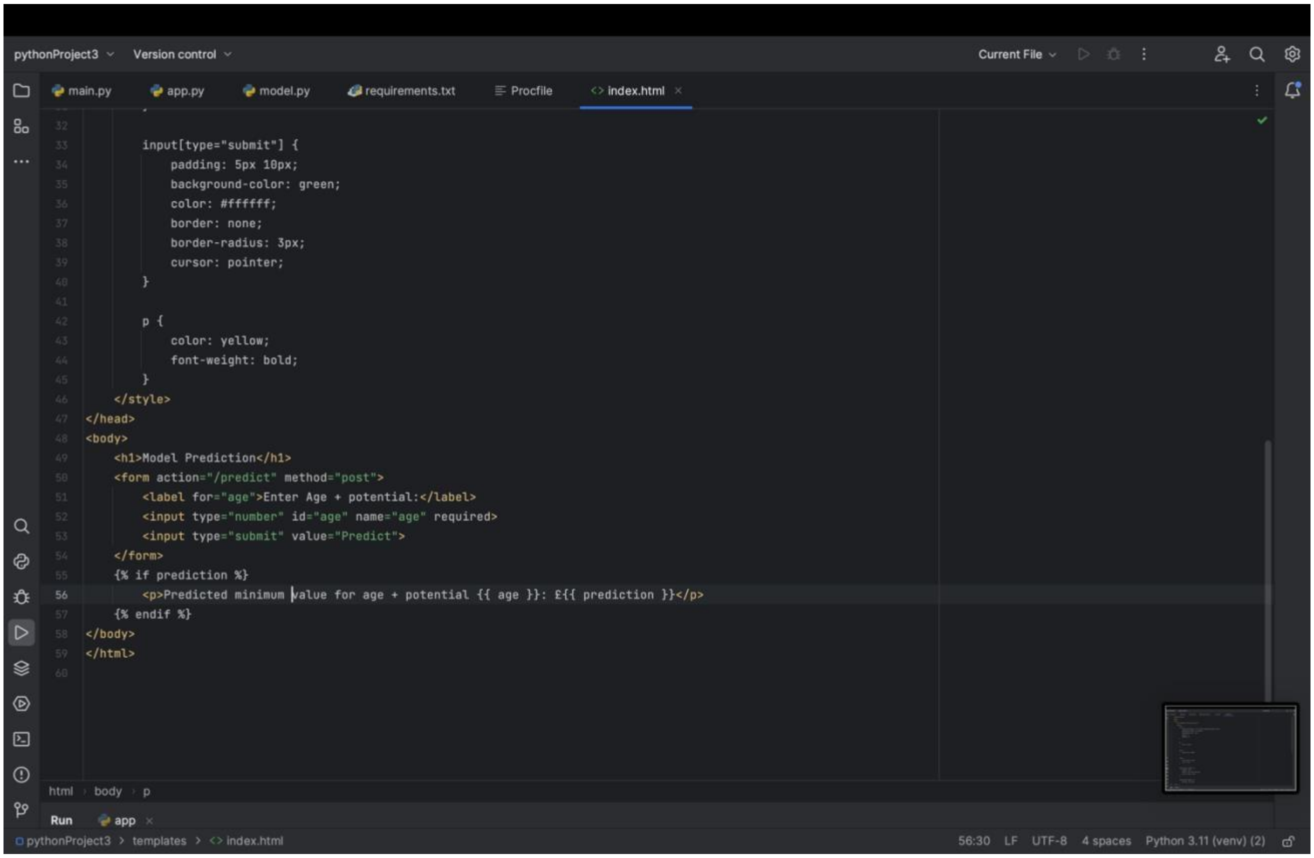


Figure 3.2: Home.html

- Background Image, pontential.jpg located in static/images contains an image that serves as the background instead of a plain color.

5 Running Procedure

Once we have done all the above, we can start running the API by either double clicking run.

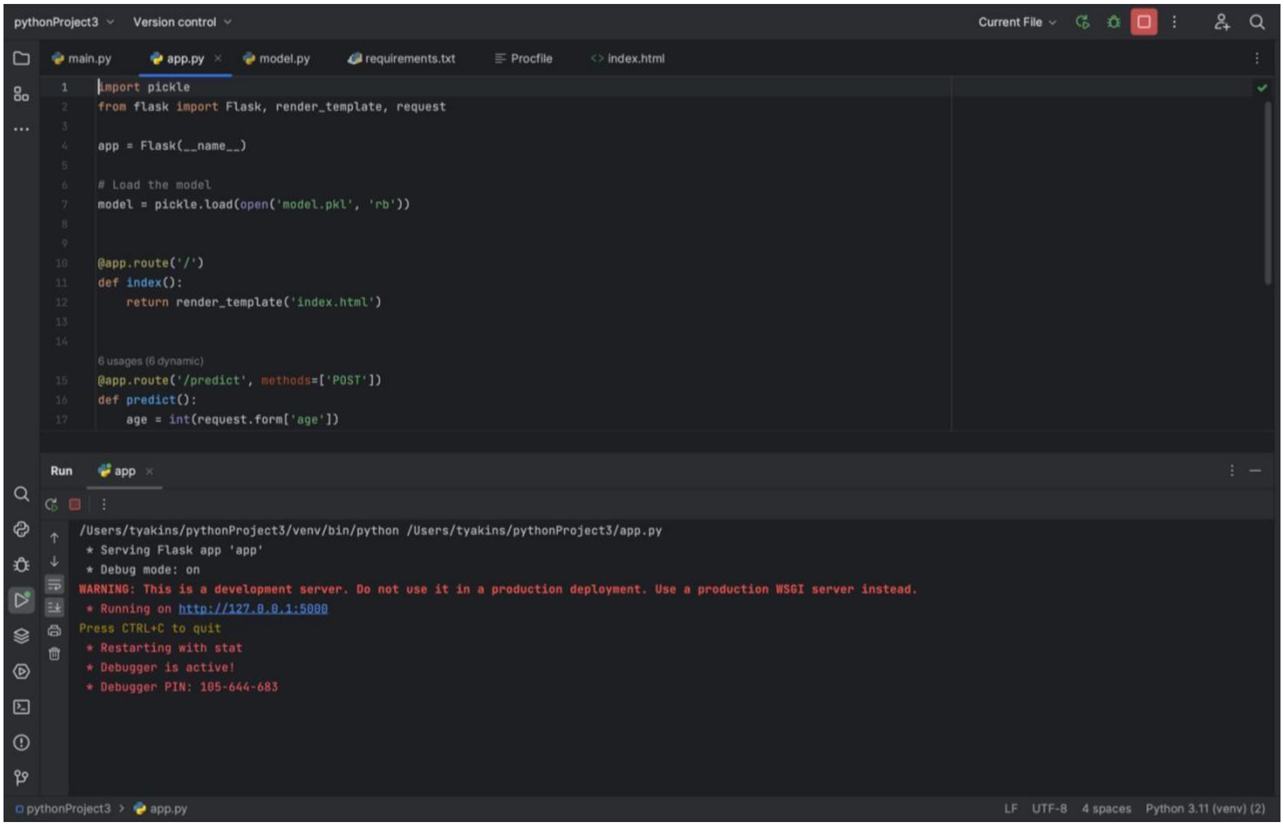


Figure 5.1 Command Execution

Next, we head over to the link provided.

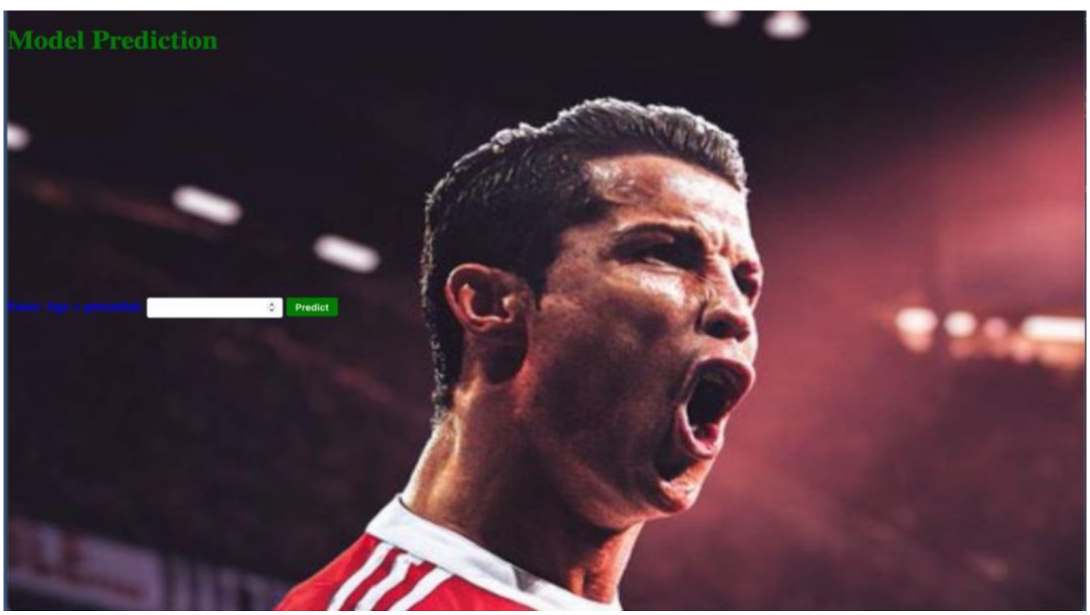


Figure 5.2: Homepage

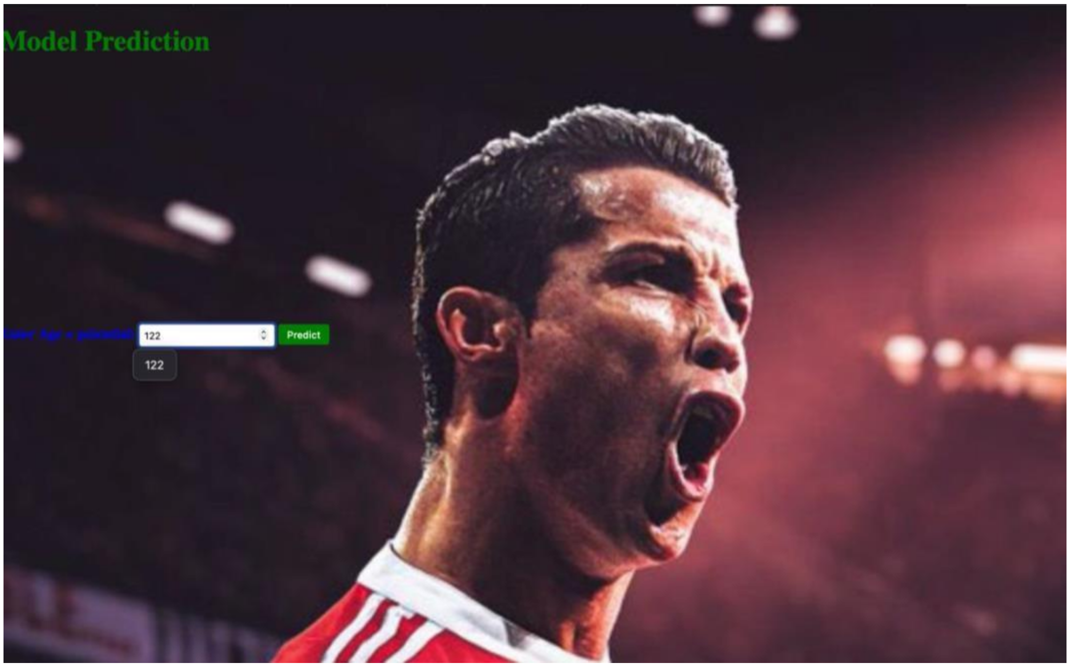


Figure 5.3: Input in The Comments Form

After entering the input click the predict button now, we can the result of our input.

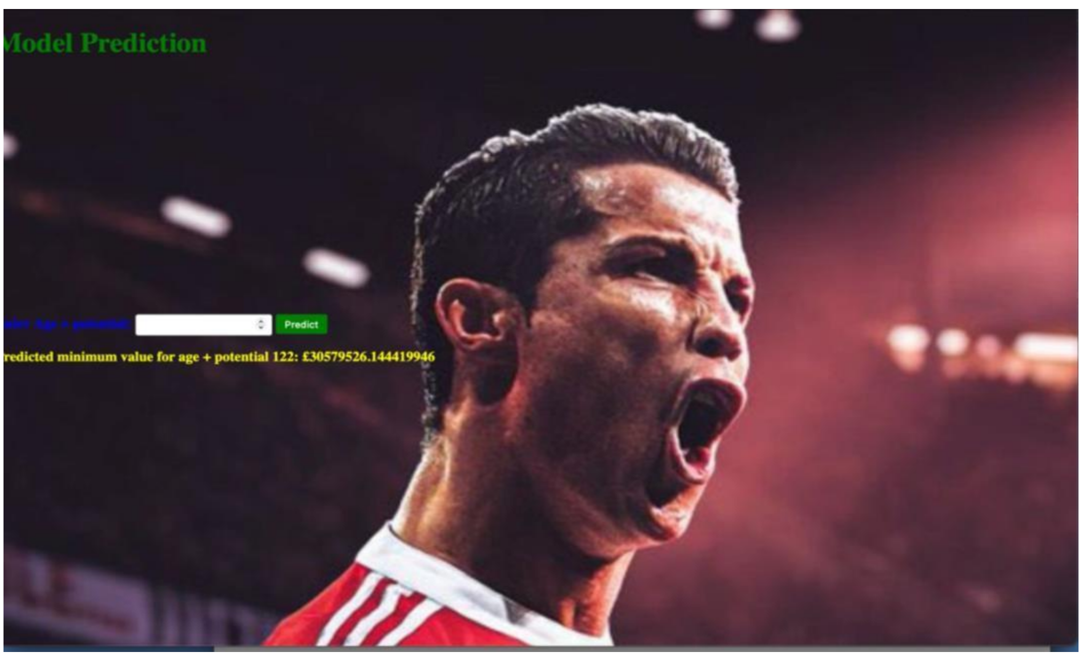


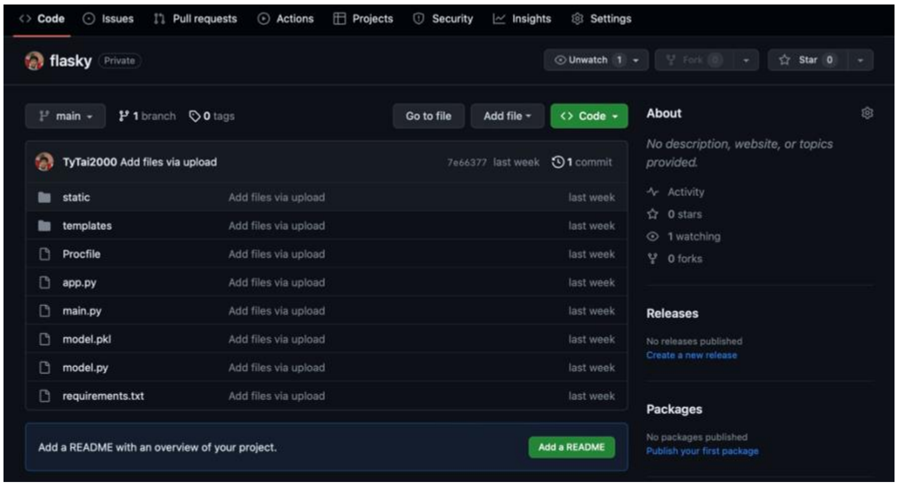
Figure 3.7: Result of Input

6. Deploying model using Heroku

Now that our model has been successfully deployed on flask, we are going to deploy it on a Cloud API called Heroku. We are going to upload the required file into a new GitHub repository and connect Heroku to that repo. The steps are below.

6.1 Uploading file on GitHub

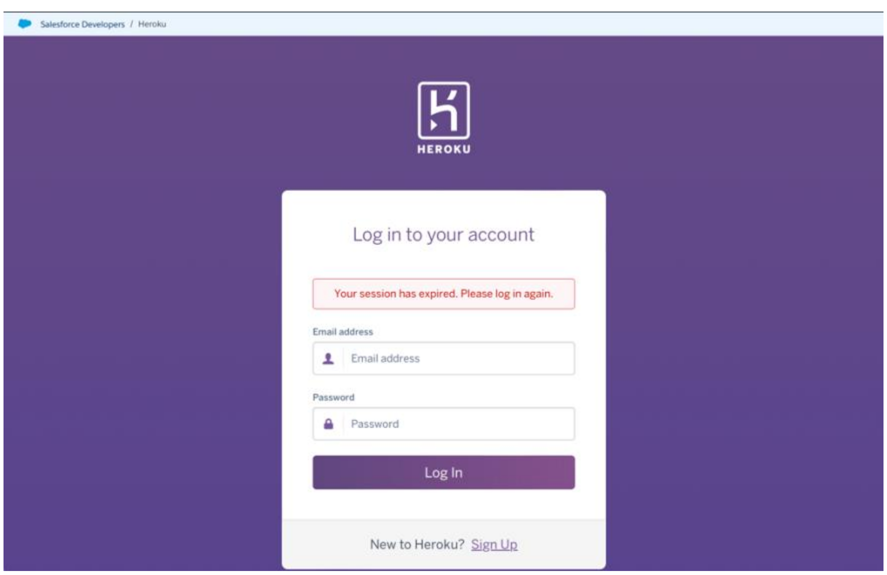
First, we create a GitHub repository and upload the necessary files.



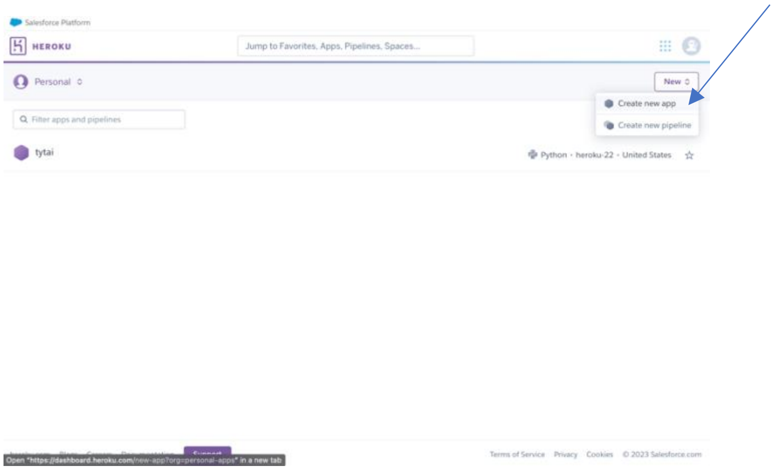
6.2 Setting up Heroku.

Then we create/login on Heroku.com.

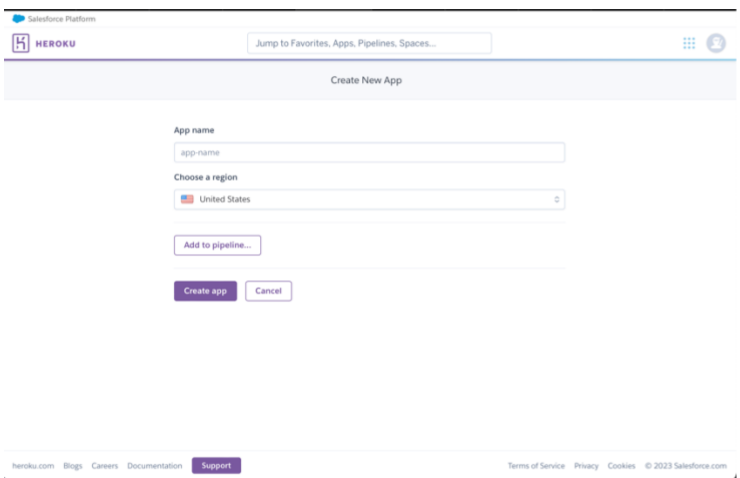
-



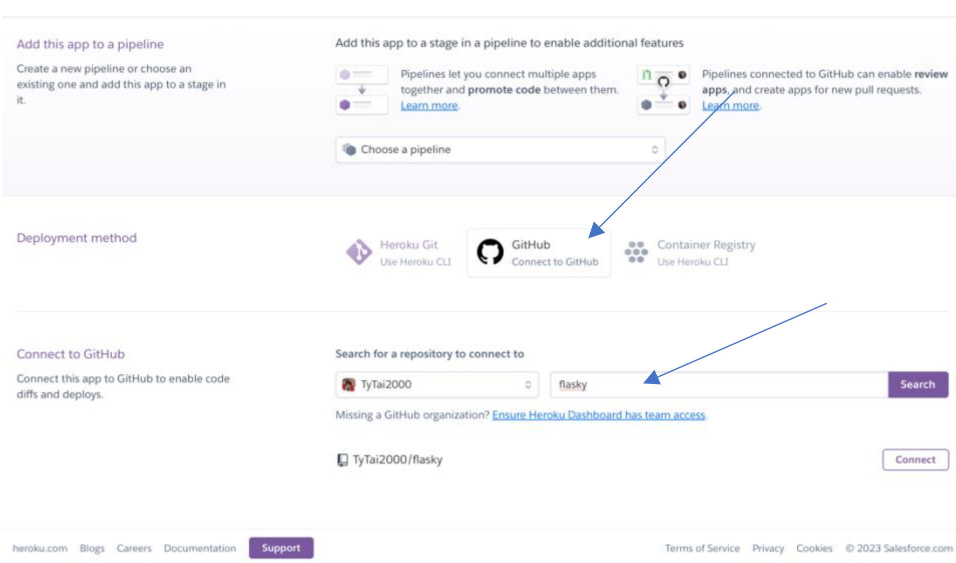
- Then we select create a new app



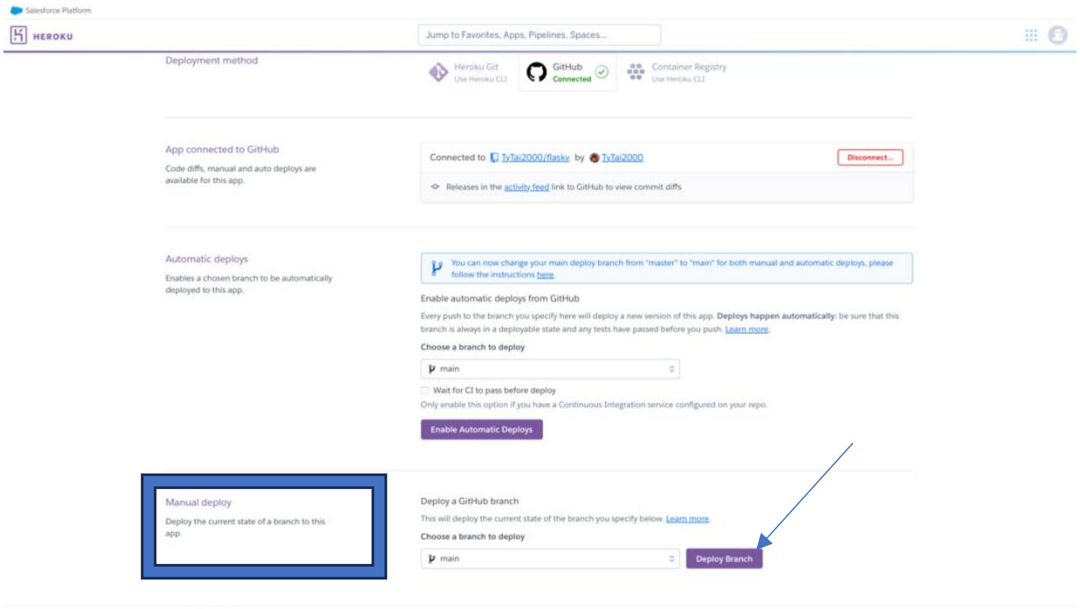
- Add a desired app name



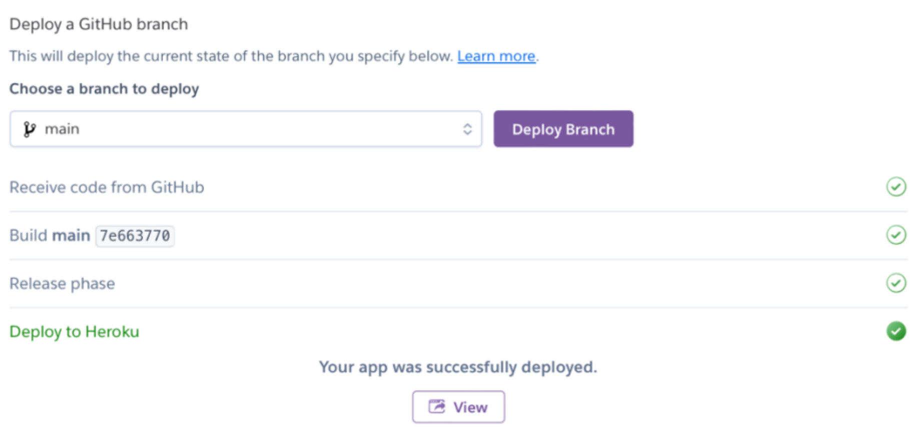
- The we connect GitHub and input the name of the repository from step 1.



- One check to ensure Heroku is ready to be deployed, then we use manual deploy and deploy the branch



- Voila !



- Web app was successfully deployed and can be accessed at https://tytai-c3bff7d0b412.herokuapp.com