

Project Report

Name: 500_Person_Gender_Height_Weight_Index

Report date: 21th Jan 2022

Internship Batch: LISUM05

Version: 1.0

Project by: Praneetha Rajupalepu

Data intake reviewer: Data Glacier

Data storage location: <https://www.kaggle.com/yersever/500-person-gender-height-weight-bodymassindex>

Project Location: https://github.com/PraneethaRajupalepu/BMI_Predict_on_Flask.git

Context

Body mass index is a value derived from the mass and height of a person. The BMI is defined as the body mass divided by the square of the body height and is expressed in units of kg/m², resulting from mass in kilograms and height in meters.

Content

The dataset contains information about gender, height, weight, and BMI index of individuals

Gender : Male / Female

Height: Number (cm)

Weight: Number (Kg)

Index

0 - Extremely Weak

1 - Weak

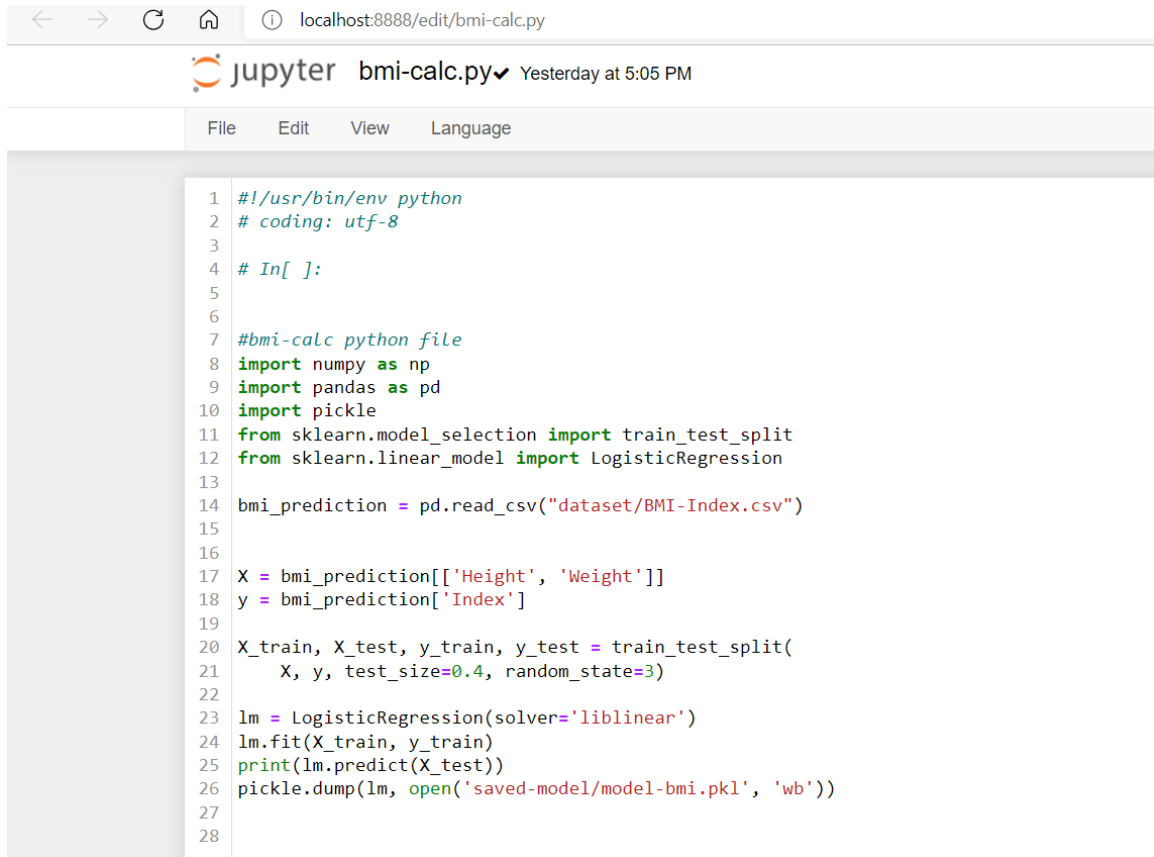
2 - Normal

3 - Overweight

4 - Obesity

5 - Extreme Obesity

1. Modeling the dataset “BMI-Index.csv”



The screenshot shows a Jupyter Notebook interface in a web browser. The address bar indicates the URL is localhost:8888/edit/bmi-calc.py. The Jupyter logo and the filename 'bmi-calc.py' are visible, along with a timestamp 'Yesterday at 5:05 PM'. Below the header is a menu bar with 'File', 'Edit', 'View', and 'Language'. The main area contains a code editor with a Python script. The script starts with a shebang and encoding declaration, followed by imports for numpy, pandas, pickle, and sklearn. It then reads a CSV file, splits the data into training and testing sets, fits a LogisticRegression model, and saves it as a pickle file.

```
1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # In[ ]:
5
6
7 #bmi-calc python file
8 import numpy as np
9 import pandas as pd
10 import pickle
11 from sklearn.model_selection import train_test_split
12 from sklearn.linear_model import LogisticRegression
13
14 bmi_prediction = pd.read_csv("dataset/BMI-Index.csv")
15
16
17 X = bmi_prediction[['Height', 'Weight']]
18 y = bmi_prediction['Index']
19
20 X_train, X_test, y_train, y_test = train_test_split(
21     X, y, test_size=0.4, random_state=3)
22
23 lm = LogisticRegression(solver='liblinear')
24 lm.fit(X_train, y_train)
25 print(lm.predict(X_test))
26 pickle.dump(lm, open('saved-model/model-bmi.pkl', 'wb'))
27
28
```

2. HTML codes (index.html)

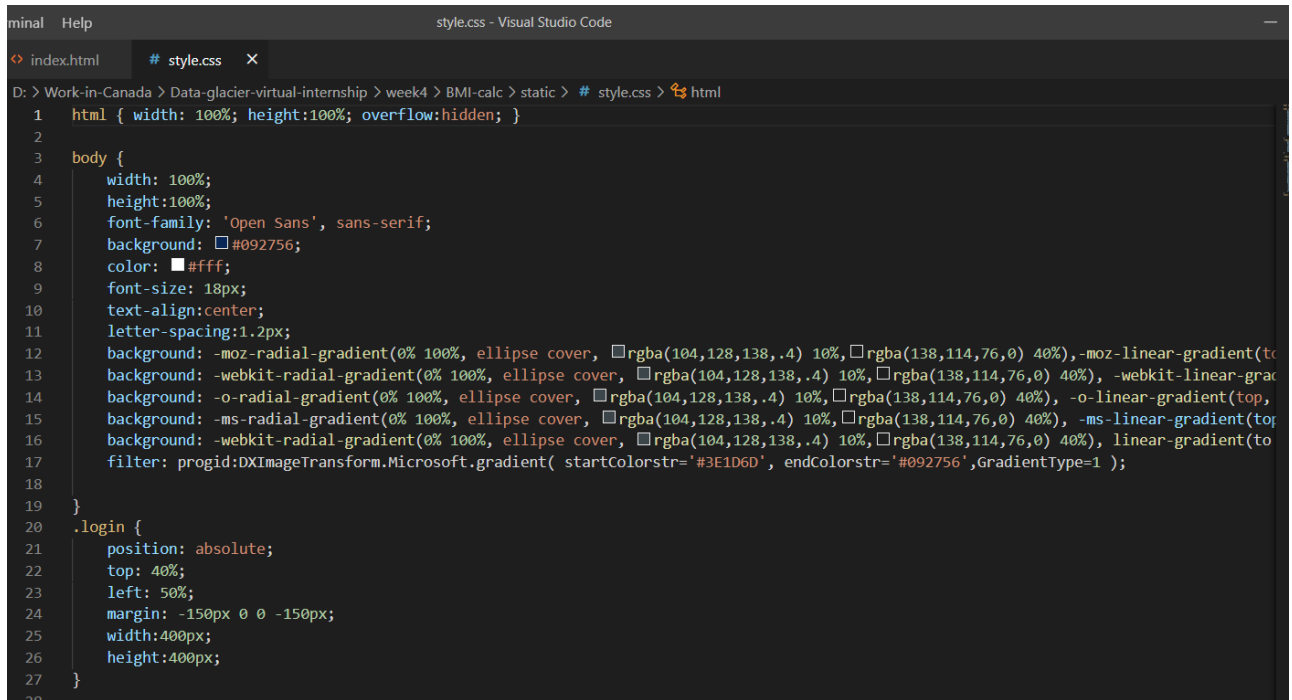
```
Terminal  Help  index.html - Visual Studio Code

index.html X

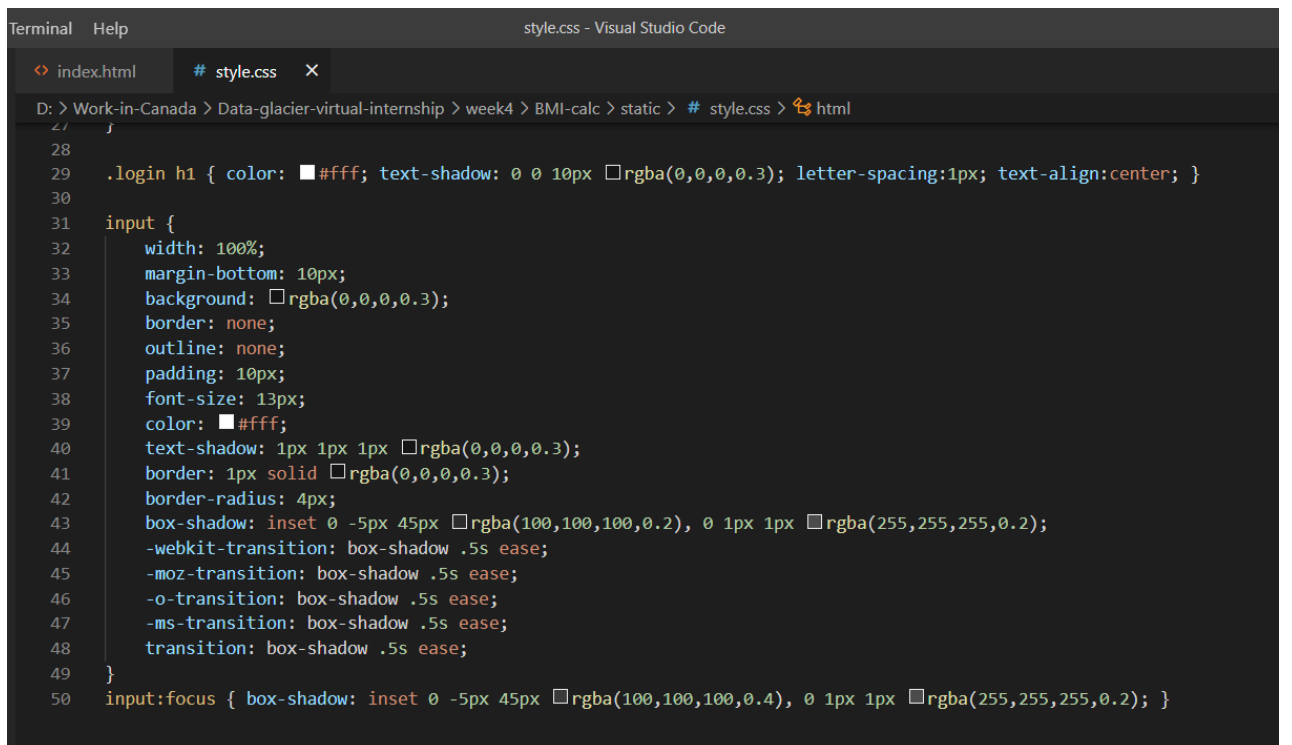
D: > Work-in-Canada > Data-glacier-virtual-internship > week4 > BMI-calc > template > index.html > ...

1  <!DOCTYPE html>
2  <html >
3  <!--From https://codepen.io/frytyler/pen/EGdtg-->
4  <head>
5      <meta charset="UTF-8">
6      <title>Body Mass Index (BMI) Prediction</title>
7      <link href="https://fonts.googleapis.com/css?family=Pacifico" rel="stylesheet" type="text/css">
8      <link href="https://fonts.googleapis.com/css?family=Arimo" rel="stylesheet" type="text/css">
9      <link href="https://fonts.googleapis.com/css?family=Hind:300" rel="stylesheet" type="text/css">
10     <link href="https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300" rel="stylesheet" type="text/css">
11     <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
12 </head>
13
14 <body>
15     <div class="login">
16         <h1> Body Mass Index (BMI) Prediction </h1>
17         <h3> Please Enter Your Height and Weight</h3>
18
19         <!-- Main Input For Receiving Query to our ML -->
20         <form action="{{ url_for('predict') }}" method="post">
21             <input type="text" name="Height" placeholder="height in cm" required="required" />
22             <input type="text" name="Weight" placeholder="weight in kg" required="required" />
23
24             <button type="submit" class="btn btn-primary btn-block btn-large">BMI Predict</button>
25         </form>
26     <br>
27     <br>
28
29     {{prediction_text}}
30
31 </div>
32 </body>
33 </html>
```

3. Style.css

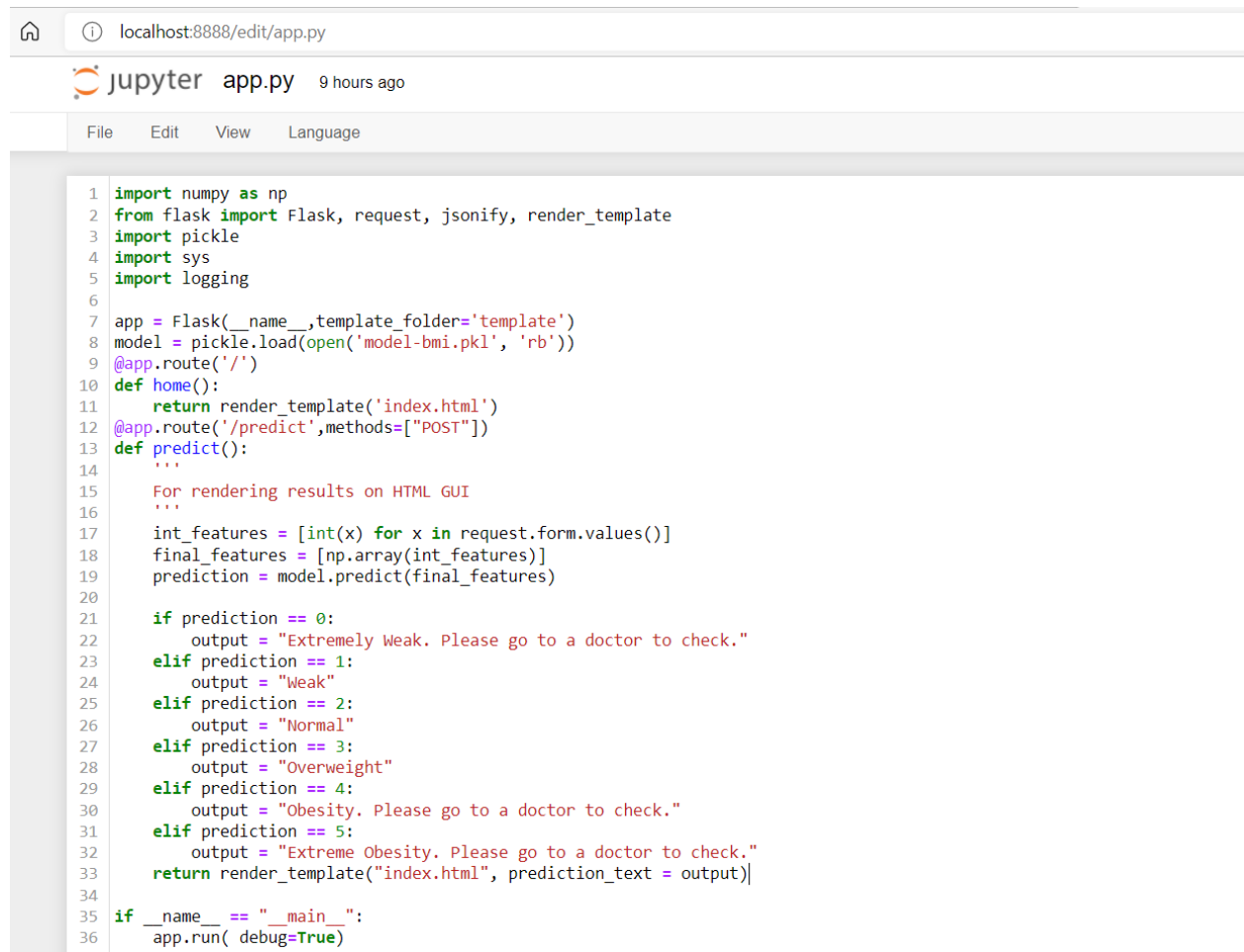


```
1  html { width: 100%; height:100%; overflow:hidden; }
2
3  body {
4      width: 100%;
5      height:100%;
6      font-family: 'Open Sans', sans-serif;
7      background: #092756;
8      color: #ffff;
9      font-size: 18px;
10     text-align:center;
11     letter-spacing:1.2px;
12     background: -moz-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%,-moz-linear-gradient(to
13     background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -webkit-linear-grac
14     background: -o-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -o-linear-gradient(top,
15     background: -ms-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -ms-linear-gradient(to
16     background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), linear-gradient(to
17     filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#3E1D6D', endColorstr='#092756',GradientType=1 );
18 }
19
20 .login {
21     position: absolute;
22     top: 40%;
23     left: 50%;
24     margin: -150px 0 0 -150px;
25     width:400px;
26     height:400px;
27 }
```



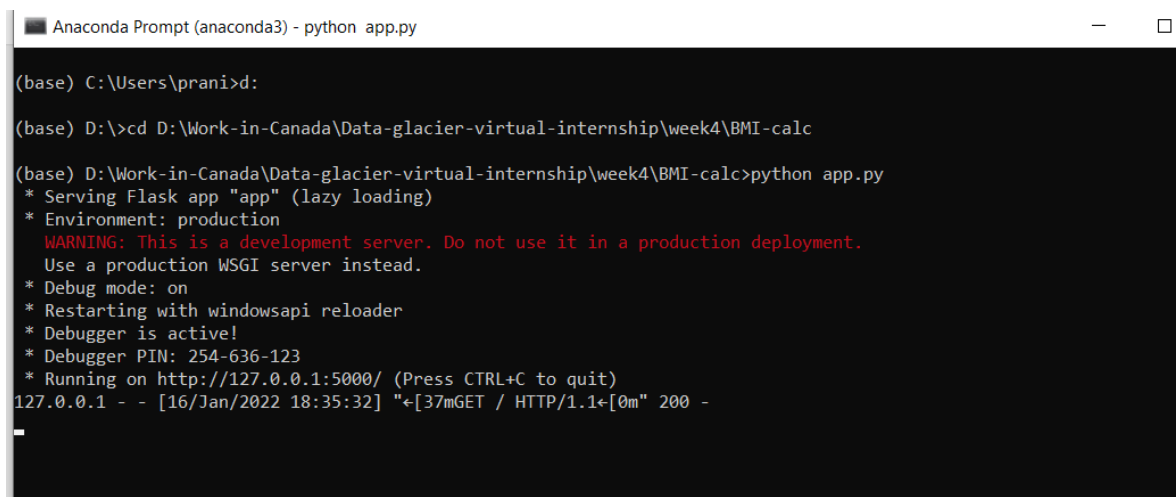
```
28
29 .login h1 { color: #ffff; text-shadow: 0 0 10px rgba(0,0,0,0.3); letter-spacing:1px; text-align:center; }
30
31 input {
32     width: 100%;
33     margin-bottom: 10px;
34     background: rgba(0,0,0,0.3);
35     border: none;
36     outline: none;
37     padding: 10px;
38     font-size: 13px;
39     color: #ffff;
40     text-shadow: 1px 1px 1px rgba(0,0,0,0.3);
41     border: 1px solid rgba(0,0,0,0.3);
42     border-radius: 4px;
43     box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px rgba(255,255,255,0.2);
44     -webkit-transition: box-shadow .5s ease;
45     -moz-transition: box-shadow .5s ease;
46     -o-transition: box-shadow .5s ease;
47     -ms-transition: box-shadow .5s ease;
48     transition: box-shadow .5s ease;
49 }
50 input:focus { box-shadow: inset 0 -5px 45px rgba(100,100,100,0.4), 0 1px 1px rgba(255,255,255,0.2); }
```

4. App.py



```
1 import numpy as np
2 from flask import Flask, request, jsonify, render_template
3 import pickle
4 import sys
5 import logging
6
7 app = Flask(__name__, template_folder='template')
8 model = pickle.load(open('model-bmi.pkl', 'rb'))
9 @app.route('/')
10 def home():
11     return render_template('index.html')
12 @app.route('/predict', methods=['POST'])
13 def predict():
14     """
15     For rendering results on HTML GUI
16     """
17     int_features = [int(x) for x in request.form.values()]
18     final_features = [np.array(int_features)]
19     prediction = model.predict(final_features)
20
21     if prediction == 0:
22         output = "Extremely Weak. Please go to a doctor to check."
23     elif prediction == 1:
24         output = "Weak"
25     elif prediction == 2:
26         output = "Normal"
27     elif prediction == 3:
28         output = "Overweight"
29     elif prediction == 4:
30         output = "Obesity. Please go to a doctor to check."
31     elif prediction == 5:
32         output = "Extreme Obesity. Please go to a doctor to check."
33     return render_template("index.html", prediction_text = output)
34
35 if __name__ == "__main__":
36     app.run(debug=True)
```

5. Converting notebook to .py file and running python code



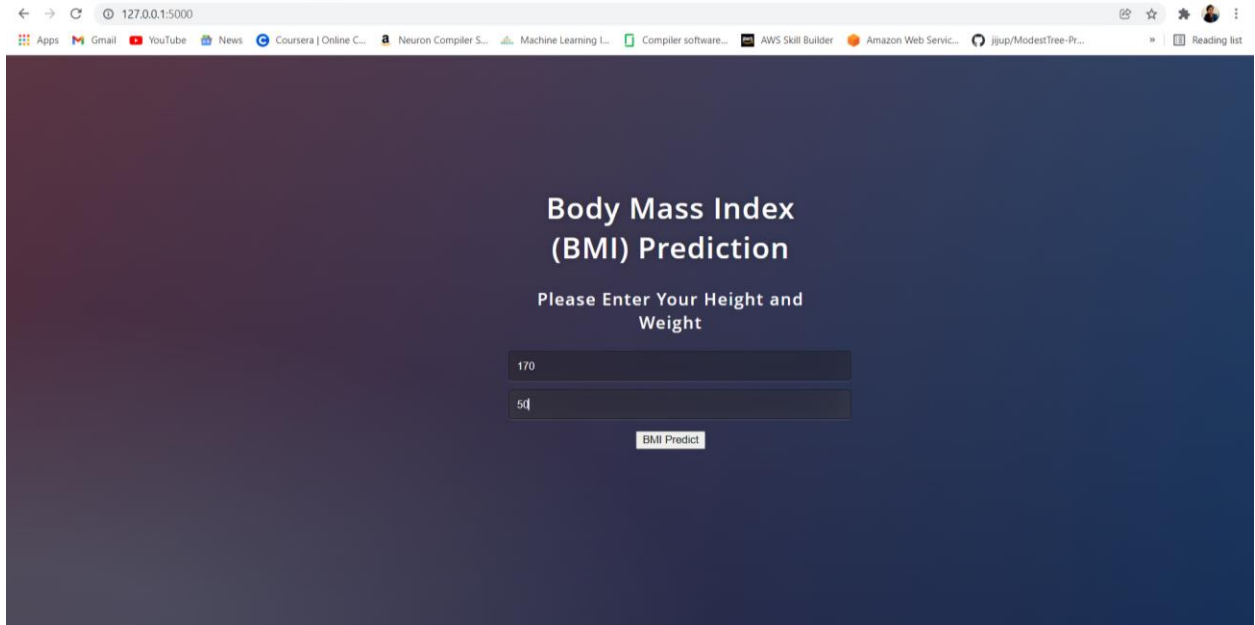
```
Anaconda Prompt (anaconda3) - python app.py

(base) C:\Users\prani>d:
(base) D:\>cd D:\Work-in-Canada\Data-glacier-virtual-internship\week4\BMI-calc
(base) D:\Work-in-Canada\Data-glacier-virtual-internship\week4\BMI-calc>python app.py
* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with windowsapi reloader
* Debugger is active!
* Debugger PIN: 254-636-123
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [16/Jan/2022 18:35:32] "GET / HTTP/1.1" 200 -
```

To check the output type <http://127.0.0.1:5000/> in the browser:

6. Examples of the model

a. Normal (Prediction)



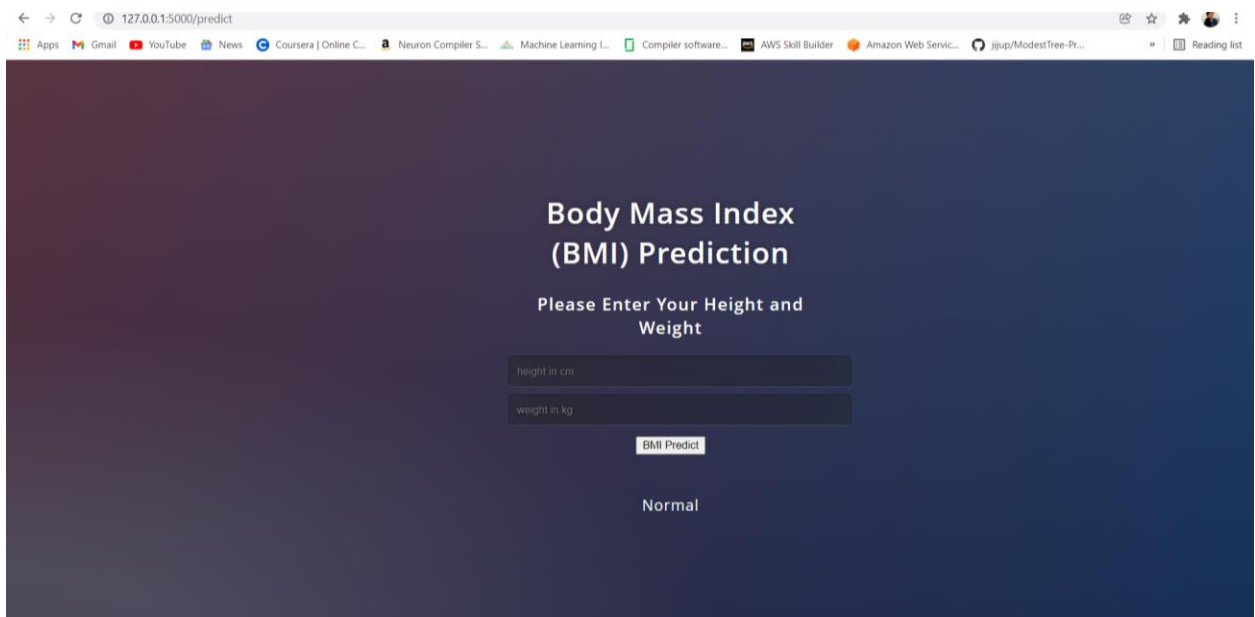
Body Mass Index (BMI) Prediction

Please Enter Your Height and Weight

170

50

BMI Predict



Body Mass Index (BMI) Prediction

Please Enter Your Height and Weight

height in cm

weight in kg

BMI Predict

Normal

b. Obese (prediction)

A screenshot of a web browser displaying a BMI prediction application. The browser's address bar shows the URL `127.0.0.1:5000/predict`. The application has a dark blue background with white text. The title is "Body Mass Index (BMI) Prediction". Below the title, it says "Please Enter Your Height and Weight". There are two input fields: the first contains the number "170" and the second contains "80". Below these fields is a button labeled "BMI Predict". The result displayed at the bottom is "Normal".

A screenshot of the same BMI prediction web application, but with different input values. The title and instructions remain the same. The first input field, labeled "height in cm", contains the text "height in cm". The second input field, labeled "weight in kg", contains the text "weight in kg". The "BMI Predict" button is still present. The result displayed at the bottom is "Obesity. Please go to a doctor to check.".

c. Extremely week (prediction)

← → ↻ 127.0.0.1:5000/predict

Apps Gmail YouTube News Coursera | Online C... Neuron Compiler S... Machine Learning L... Compiler software... AWS Skill Builder Amazon Web Servic... jupyter/ModestTree-Pr... Reading list

Body Mass Index (BMI) Prediction

Please Enter Your Height and Weight

170

25

BMI Predict

Obesity. Please go to a doctor to check.

← → ↻ 127.0.0.1:5000/predict

Apps Gmail YouTube News Coursera | Online C... Neuron Compiler S... Machine Learning L... Compiler software... AWS Skill Builder Amazon Web Servic... jupyter/ModestTree-Pr... Reading list

Body Mass Index (BMI) Prediction

Please Enter Your Height and Weight

height in cm

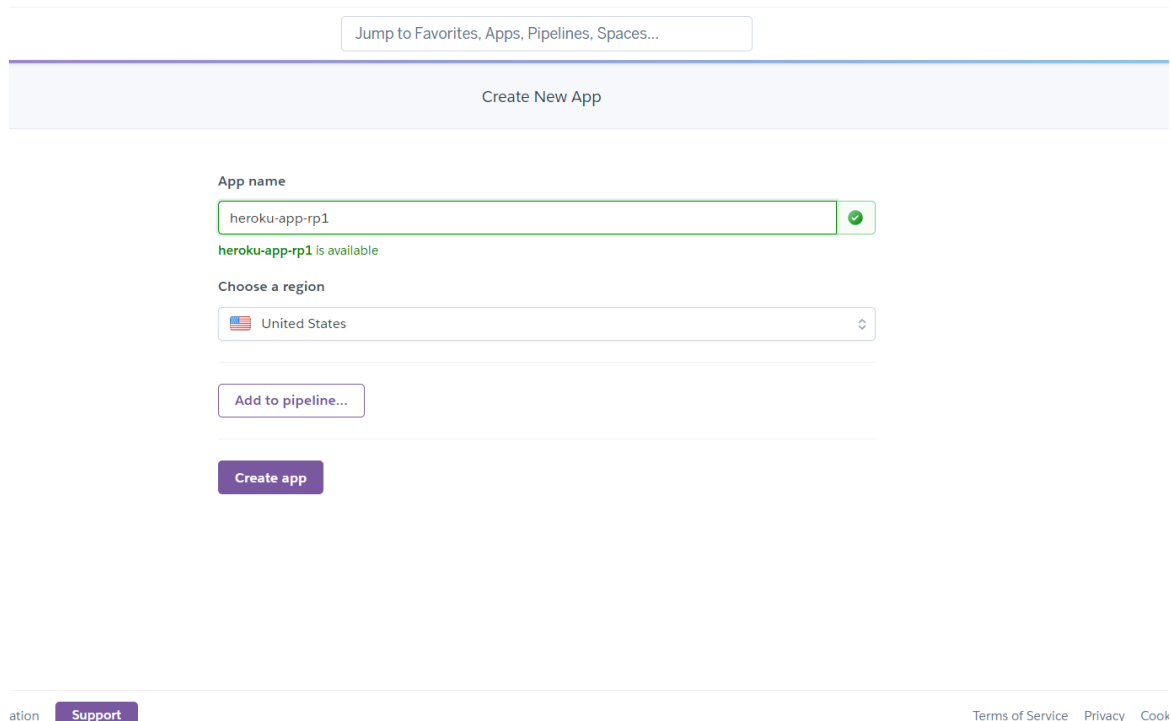
weight in kg

BMI Predict

Extremely Weak. Please go to a doctor to check.

7. Deploying the app in Heroku Cloud API

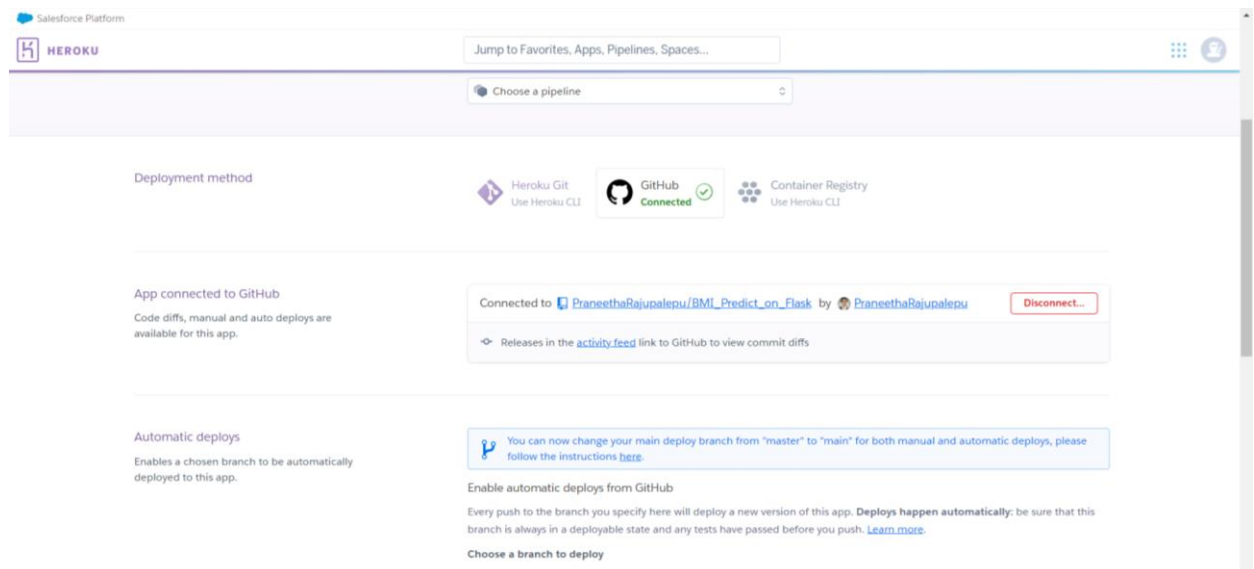
Step1: Create an app



The screenshot shows the 'Create New App' form on the Heroku dashboard. At the top, there is a search bar with the text 'Jump to Favorites, Apps, Pipelines, Spaces...'. Below this is a large button labeled 'Create New App'. The form itself has a section for 'App name' with a text input field containing 'heroku-app-rp1' and a green checkmark icon to its right. Below the input field, it says 'heroku-app-rp1 is available'. The next section is 'Choose a region' with a dropdown menu showing 'United States' and a small flag icon. Below this is a button labeled 'Add to pipeline...'. At the bottom of the form is a large purple button labeled 'Create app'. The footer of the page includes a 'Support' button and links for 'Terms of Service', 'Privacy', and 'Cook'.

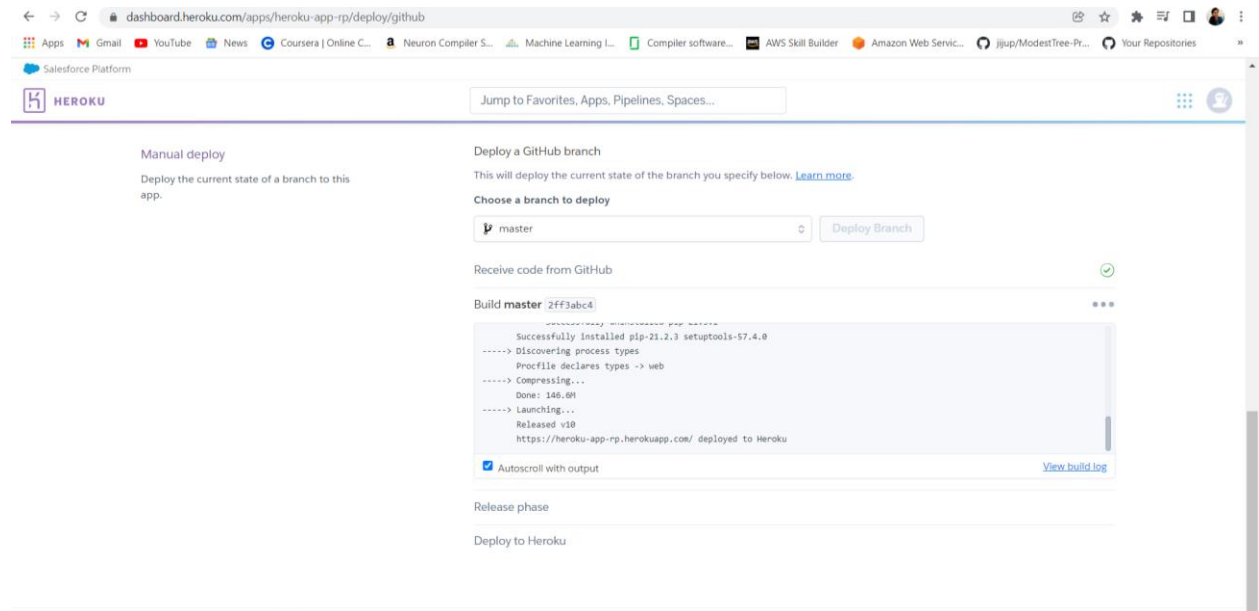
Step2: Connect to Deployment Method

After clicking on create an app, then in deployment method connect to GitHub repository and login to your GitHub account and connect your project repository.

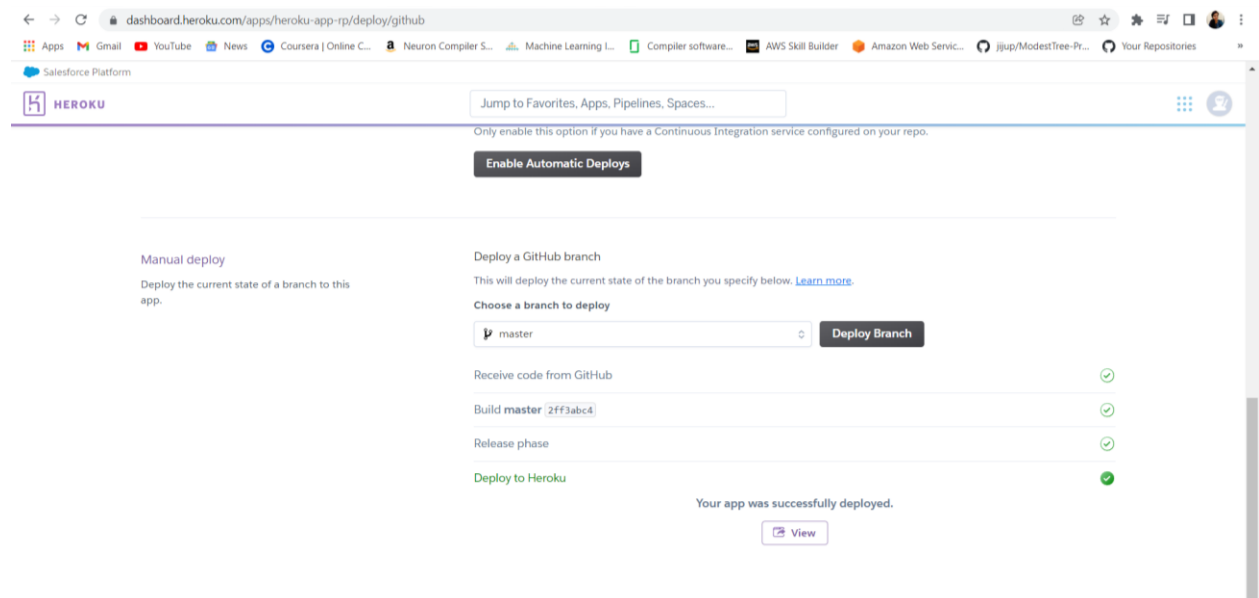


The screenshot shows the Heroku app configuration page for the app 'heroku-app-rp1'. The page has a header with the Heroku logo and a search bar. Below the header is a section for 'Deployment method' with three options: 'Heroku Git' (Use Heroku CLI), 'GitHub' (Connected), and 'Container Registry' (Use Heroku CLI). The 'GitHub' option is selected and marked as 'Connected'. Below this is a section for 'App connected to GitHub' with the text 'Code diffs, manual and auto deploys are available for this app.' and a button labeled 'Disconnect...'. Below this is a section for 'Automatic deploys' with the text 'Enables a chosen branch to be automatically deployed to this app.' and a button labeled 'Enable automatic deploys from GitHub'. The 'Enable automatic deploys from GitHub' section contains instructions on how to change the main deploy branch from 'master' to 'main' and a link to 'Learn more'.

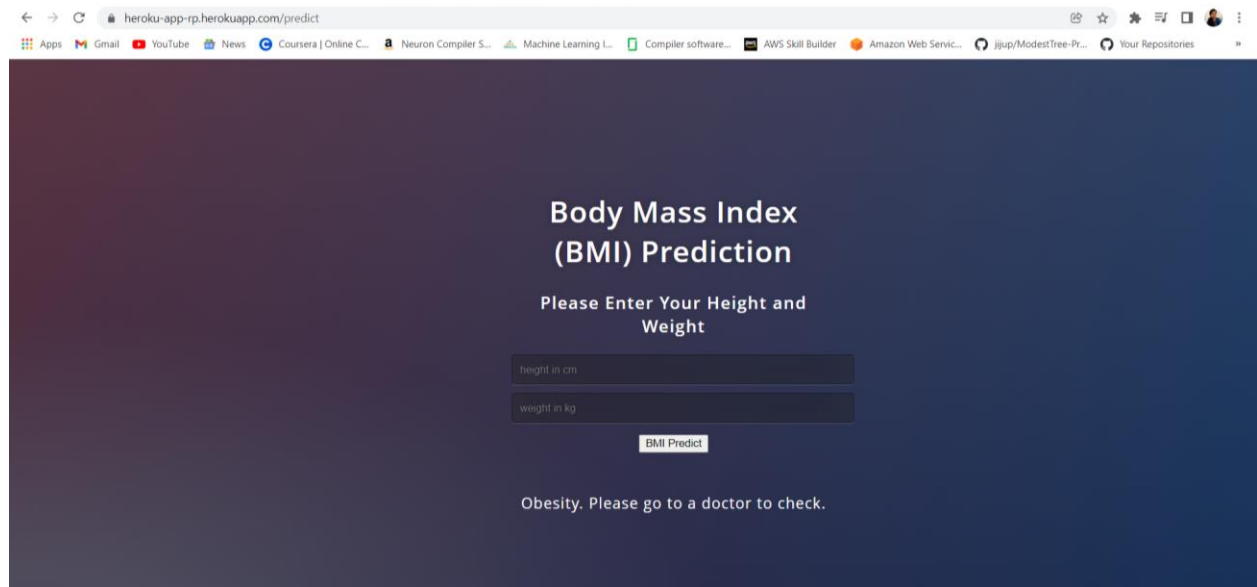
Step3: After connecting to the project repository, select manual deploy in this step is about the required libraries needed to run the web application. After successful installation build master provides a web link that is directed to the web application.



Step4: After successful installation build master provides a web link that is directed to the web application.



Step5: If we click on the view button then it will redirect to your web application URL.



The screenshot shows a web browser window with the address bar displaying `heroku-app-rp.herokuapp.com/predict`. The browser's tab bar includes several open tabs: Apps, Gmail, YouTube, News, Coursera | Online C..., Neuron Compiler S..., Machine Learning L..., Compiler software..., AWS Skill Builder, Amazon Web Servic..., jup/ModestTree-Pr..., and Your Repositories. The main content area of the browser displays a web application titled "Body Mass Index (BMI) Prediction". Below the title, it says "Please Enter Your Height and Weight". There are two input fields: "height in cm" and "weight in kg". A "BMI Predict" button is located below the input fields. At the bottom of the page, there is a message: "Obesity. Please go to a doctor to check."

Note: We can access this URL from anywhere and anytime.