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”

localhost:8888/edit/bmi-calc.py

jupyter bmi-calc.py Yesterday at 5:05 PM

File Edit View Language

```python
#!/usr/bin/env python
# coding: utf-8

# In[ ]:


#bmi-calc python file
import numpy as np
import pandas as pd
import pickle
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression

bmi_prediction = pd.read_csv("dataset/BMI-Index.csv")


X = bmi_prediction[['Height', 'Weight']]
y = bmi_prediction['Index']

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.4, random_state=3)

lm = LogisticRegression(solver='liblinear')
lm.fit(X_train, y_train)
print(lm.predict(X_test))
pickle.dump(lm, open('saved-model/model-bmi.pkl', 'wb'))
```

2. HTML codes (index.html)

Terminal Help index.html - Visual Studio Code

index.html

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

```html
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
  <meta charset="UTF-8">
  <title>Body Mass Index (BMI) Prediction</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='style.css') }}">
 </head>

<body>
 <div class="login">
  <h1> Body Mass Index (BMI) Prediction </h1>
  <h3> Please Enter Your Height and Weight</h3>

<!-- Main Input For Receiving Query to our ML -->
    <form action="{{ url_for('predict')}}"method="post">
    <input type="text" name="Height" placeholder="height in cm" required="required" />
  <input type="text" name="Weight" placeholder="weight in kg" required="required" />

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

   {{prediction_text}}

 </div>
</body>
</html>
```

3. Style.css

minal Help style.css - Visual Studio Code

index.html # style.css

D: > Work-in-Canada > Data-glacier-virtual-internship > week4 > BMI-calc > static > # style.css > html

```
html { width: 100%; height:100%; overflow:hidden; }

body {
    width: 100%;
    height:100%;
    font-family: 'Open Sans', sans-serif;
    background: #092756;
    color: #fff;
    font-size: 18px;
    text-align:center;
    letter-spacing:1.2px;
    background: -moz-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%),-moz-linear-gradient(to
    background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -webkit-linear-grad
    background: -o-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -o-linear-gradient(top,
    background: -ms-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -ms-linear-gradient(top
    background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), linear-gradient(to
    filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#3E1D6D', endColorstr='#092756',GradientType=1 );

}
.login {
    position: absolute;
    top: 40%;
    left: 50%;
    margin: -150px 0 0 -150px;
    width:400px;
    height:400px;
}
```

Terminal Help style.css - Visual Studio Code

index.html # style.css

D: > Work-in-Canada > Data-glacier-virtual-internship > week4 > BMI-calc > static > # style.css > html

```
}

.login h1 { color: #fff; text-shadow: 0 0 10px rgba(0,0,0,0.3); letter-spacing:1px; text-align:center; }

input {
    width: 100%;
    margin-bottom: 10px;
    background: rgba(0,0,0,0.3);
    border: none;
    outline: none;
    padding: 10px;
    font-size: 13px;
    color: #fff;
    text-shadow: 1px 1px 1px rgba(0,0,0,0.3);
    border: 1px solid rgba(0,0,0,0.3);
    border-radius: 4px;
    box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px rgba(255,255,255,0.2);
    -webkit-transition: box-shadow .5s ease;
    -moz-transition: box-shadow .5s ease;
    -o-transition: box-shadow .5s ease;
    -ms-transition: box-shadow .5s ease;
    transition: box-shadow .5s ease;
}
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

localhost:8888/edit/app.py

jupyter app.py 9 hours ago

File Edit View Language

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

app = Flask(__name__,template_folder='template')
model = pickle.load(open('model-bmi.pkl', 'rb'))
@app.route('/')
def home():
    return render_template('index.html')
@app.route('/predict',methods=["POST"])
def predict():
    '''
    For rendering results on HTML GUI
    '''
    int_features = [int(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    prediction = model.predict(final_features)

    if prediction == 0:
        output = "Extremely Weak. Please go to a doctor to check."
    elif prediction == 1:
        output = "Weak"
    elif prediction == 2:
        output = "Normal"
    elif prediction == 3:
        output = "Overweight"
    elif prediction == 4:
        output = "Obesity. Please go to a doctor to check."
    elif prediction == 5:
        output = "Extreme Obesity. Please go to a doctor to check."
    return render_template("index.html", prediction_text = output)

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

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

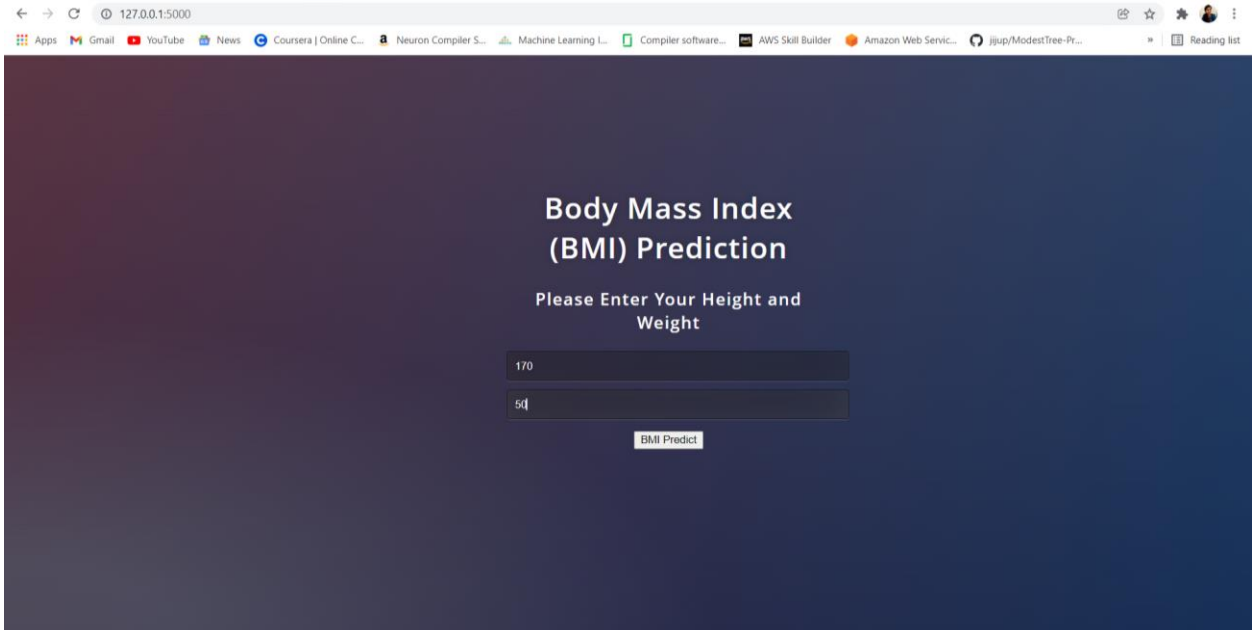
Anaconda Prompt (anaconda3) - python app.py

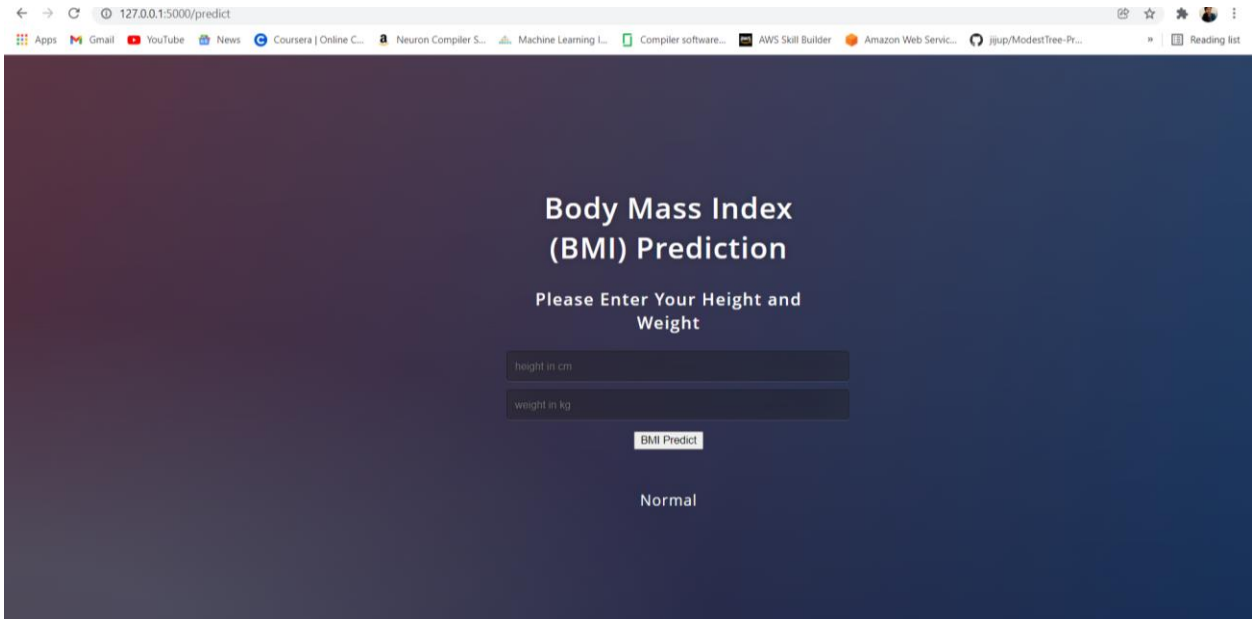
```
(base) C:\Users\prani>d:

(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] "←[37mGET / HTTP/1.1←[0m" 200 -
```

To check the output type http://127.0.0.1.5000/ in the browser:

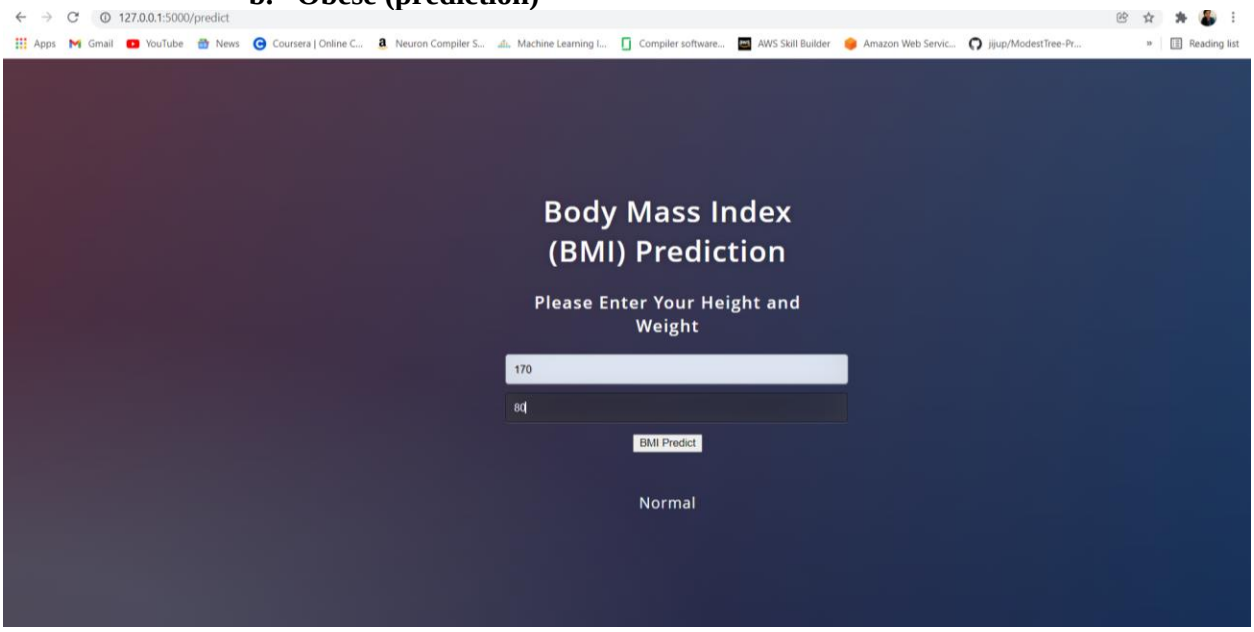
6. **Examples of the model**

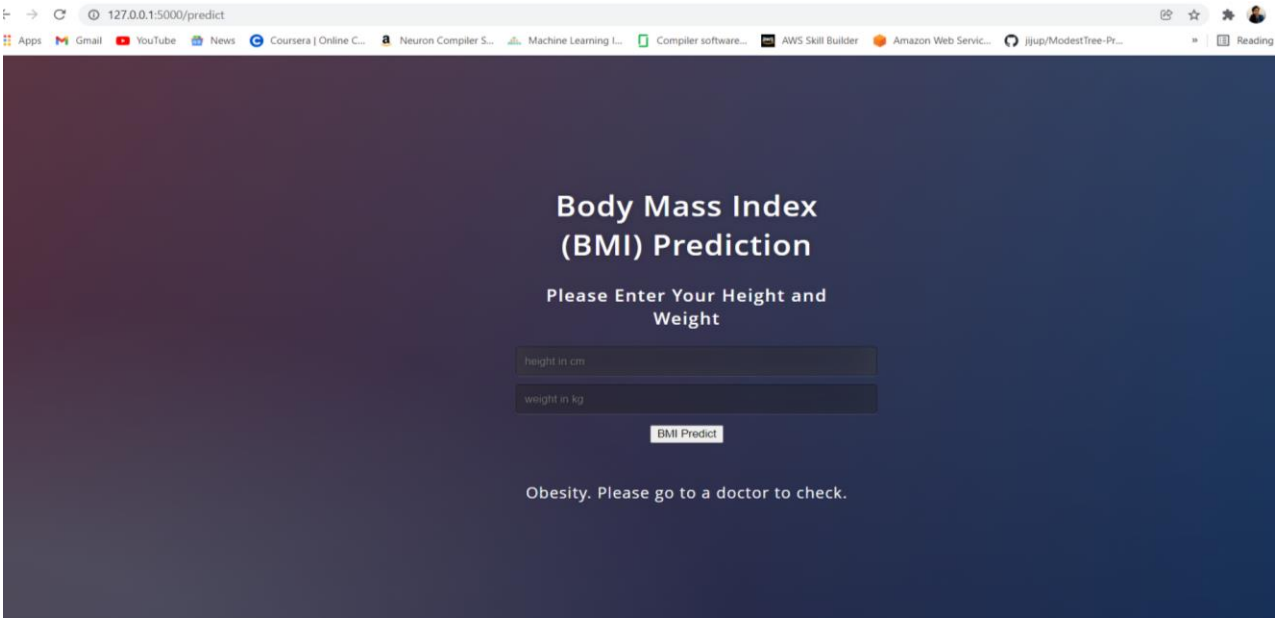
   a. **Normal (Prediction)**



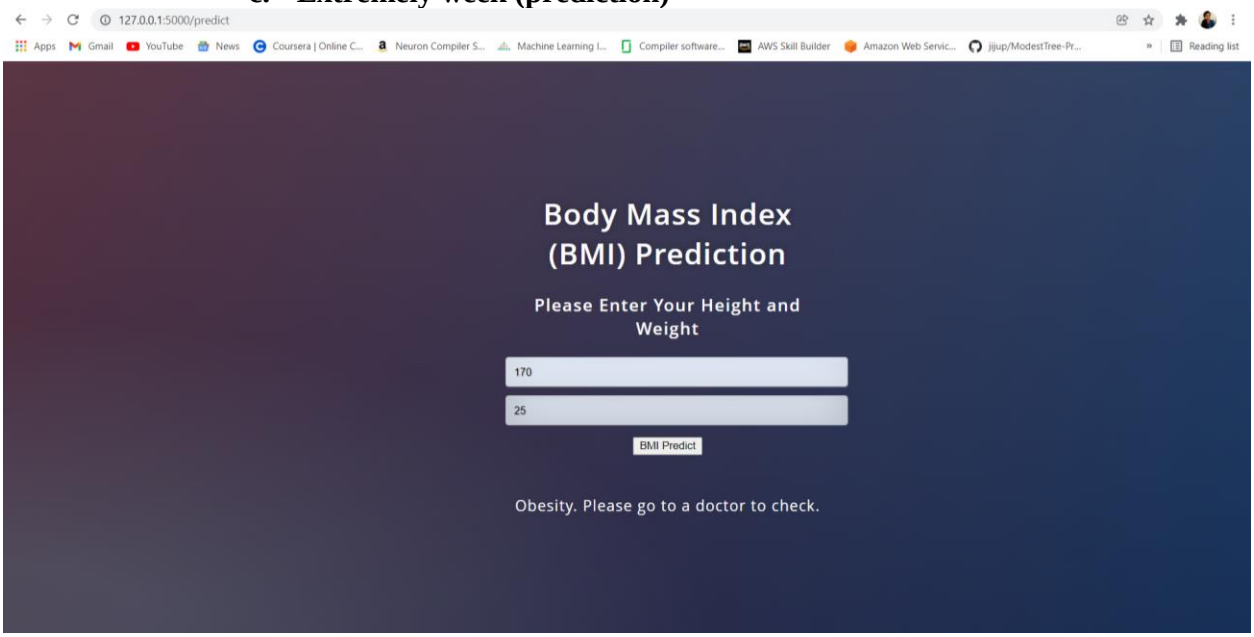


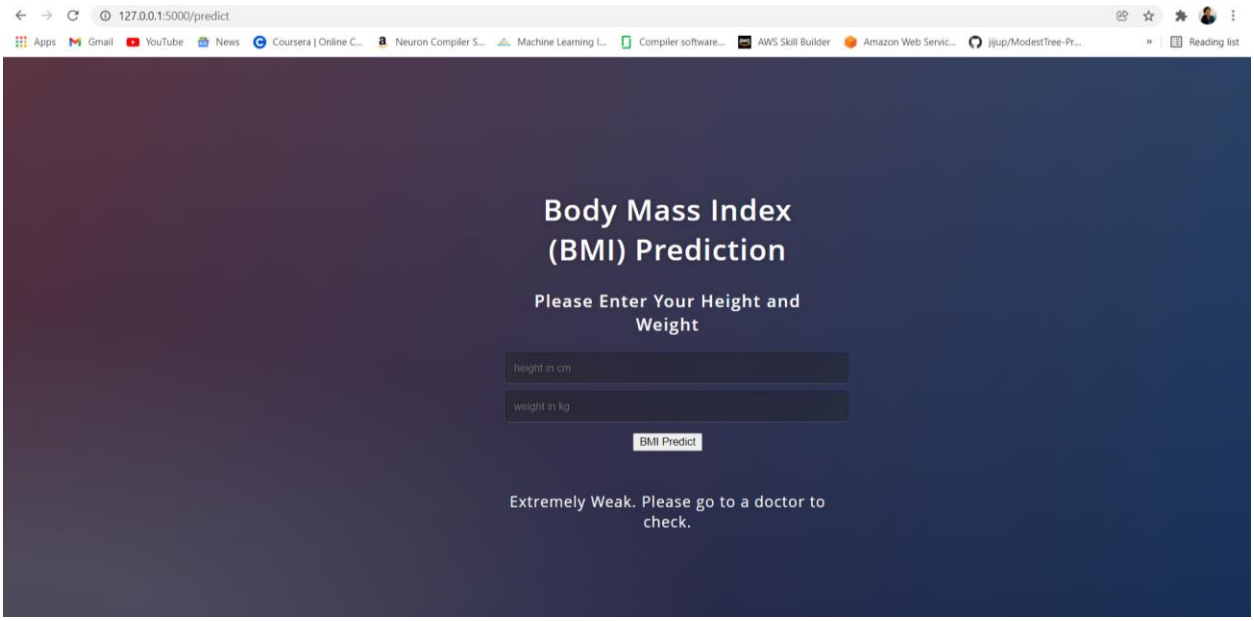
b. Obese (prediction)





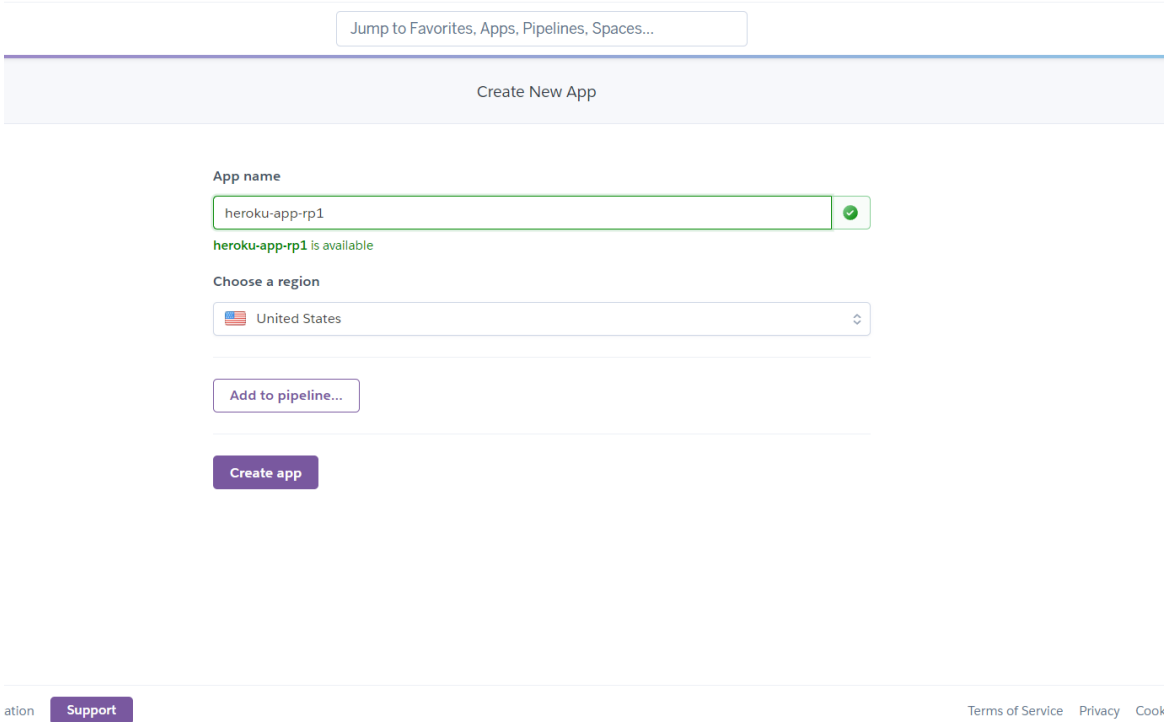
c. Extremely week (prediction)





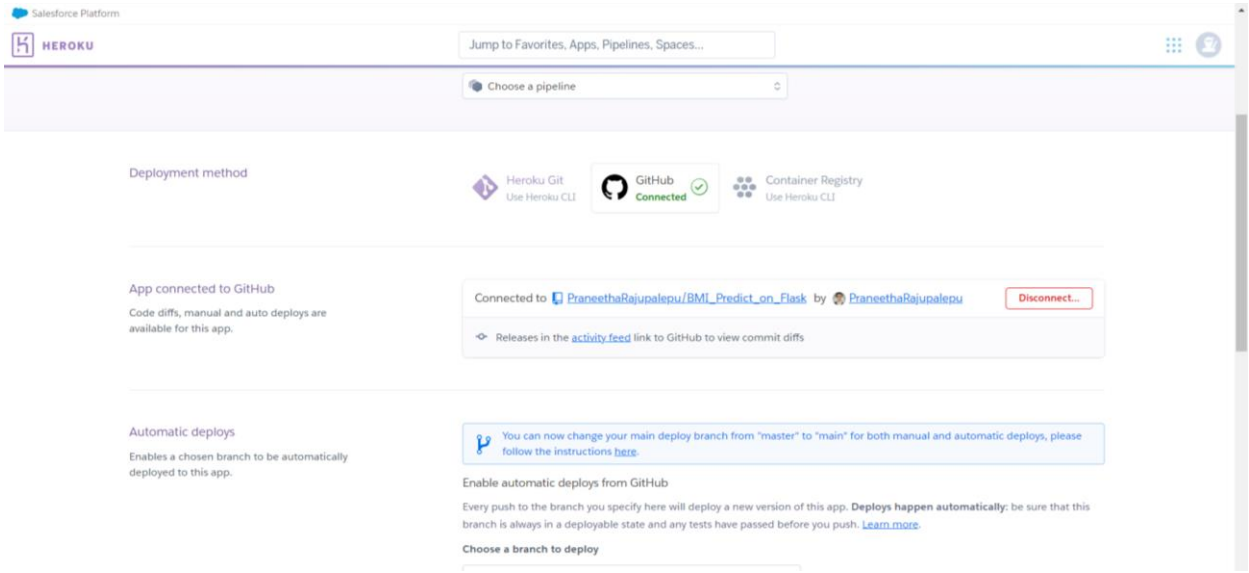
7. Deploying the app in Heroku Cloud API

Step1: Create an app

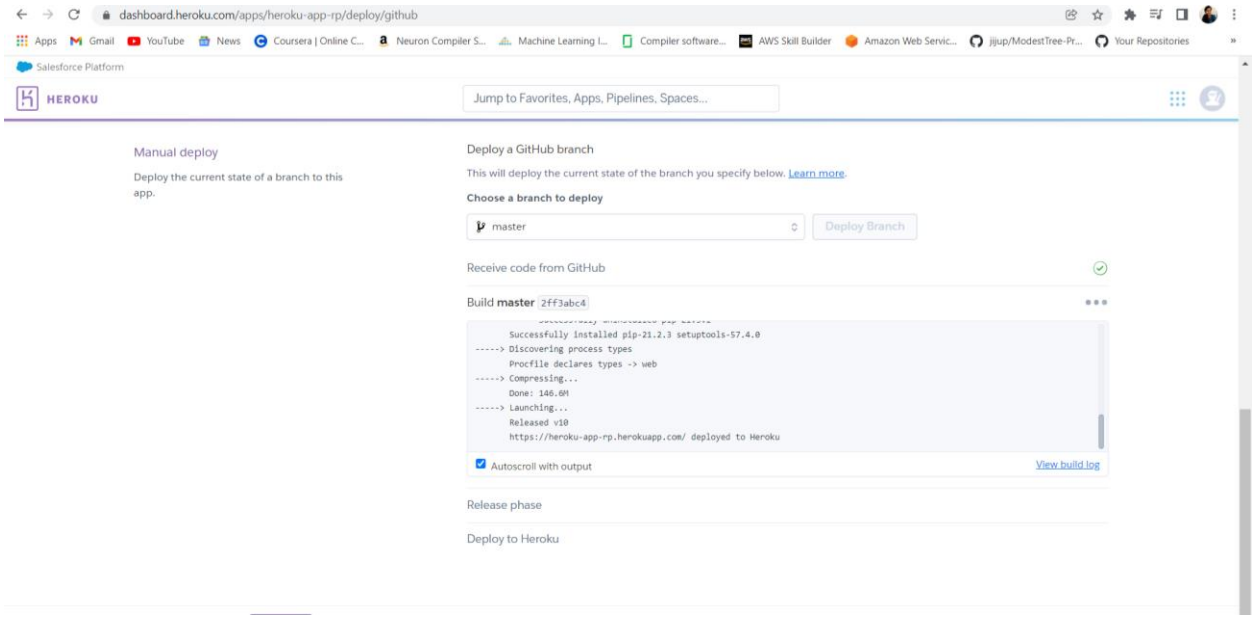


Step2: Connect to Deployment Method

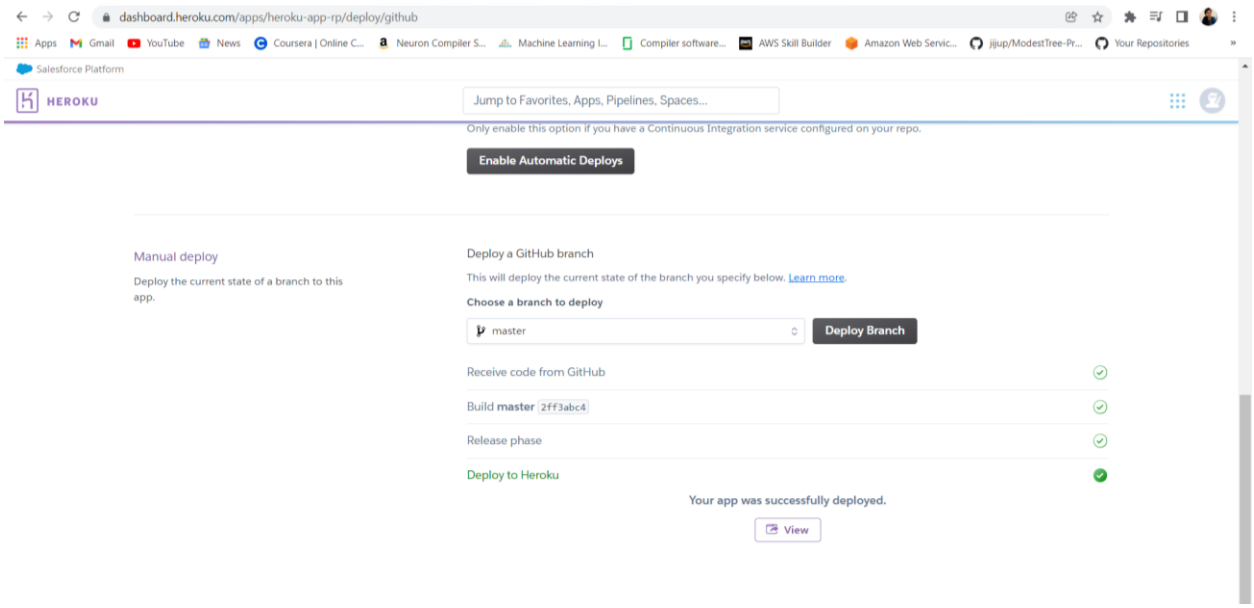
After clicking on create an app, then in deployment method connect to GitHub repository and login to your GitHub account and connect your project repository.



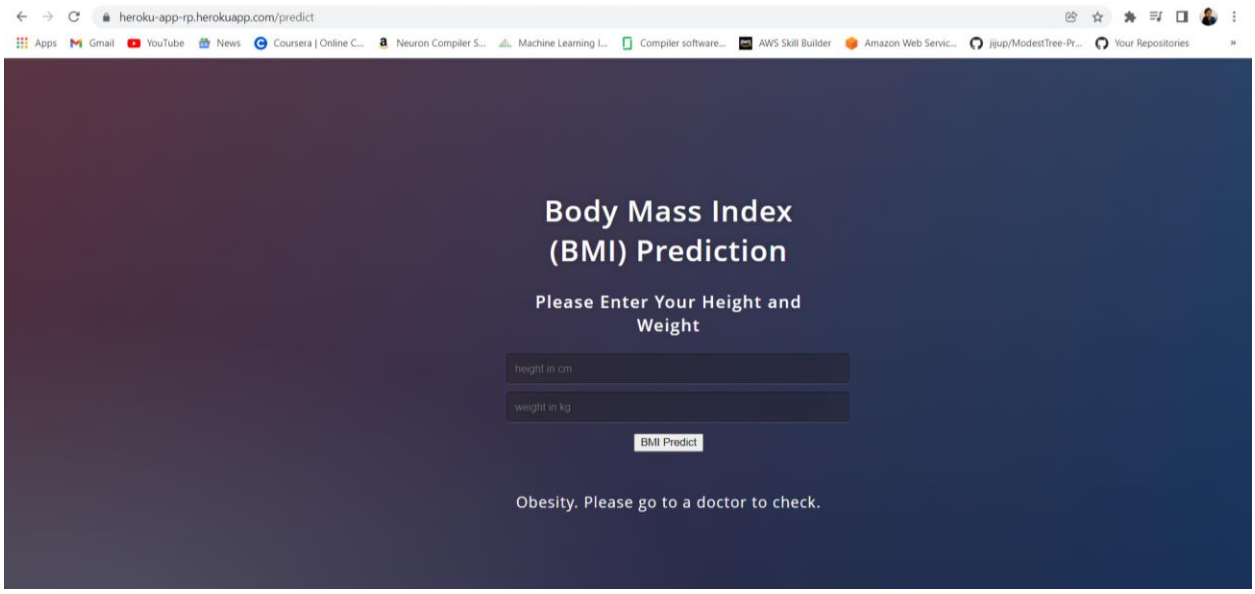
Step3: After connecting to the project repository, select manual deploy in this step is about the required libraries needed to run the web application. After successful installation build master provides a web link that is directed to the web application.



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.



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