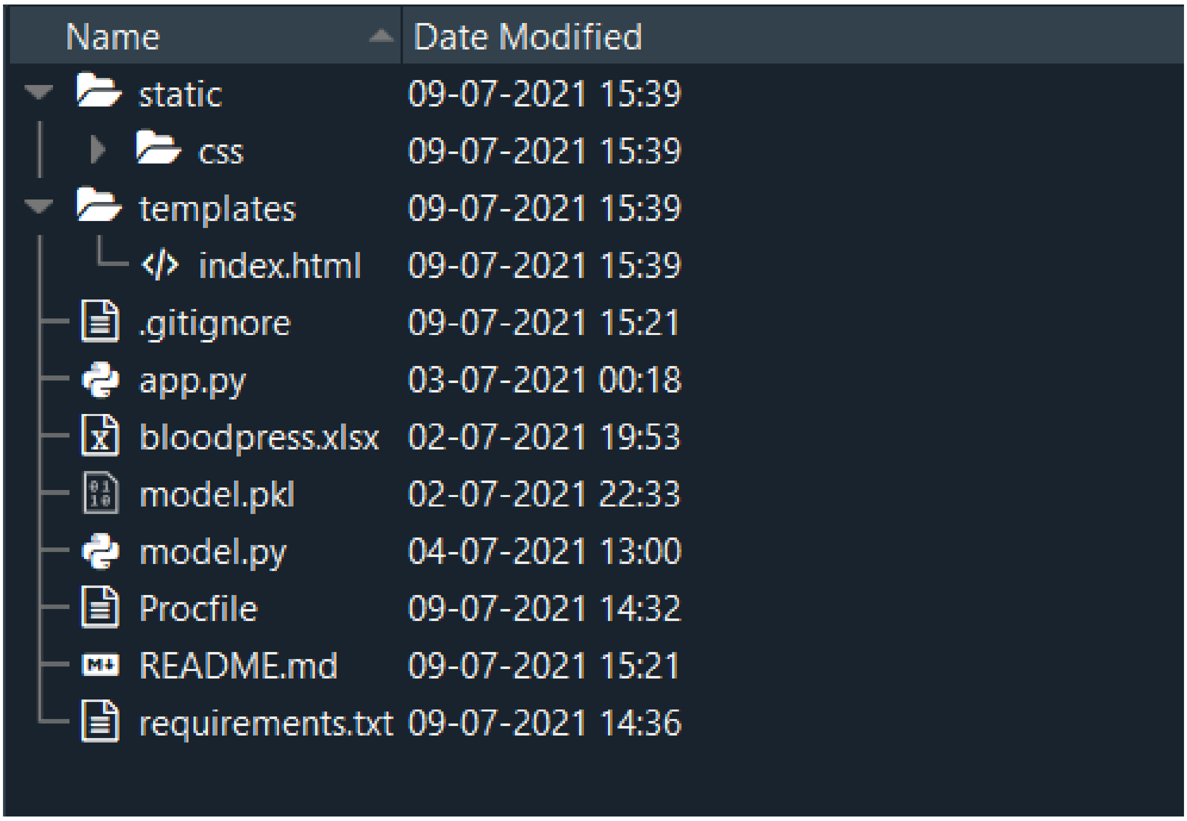
Week 5: Cloud and API Deployment

Name	Runa Veigas
Batch Code	LISUM01
Submission Date	09 July 2021
Submitted To	Data Glacier Team

Directory Structure

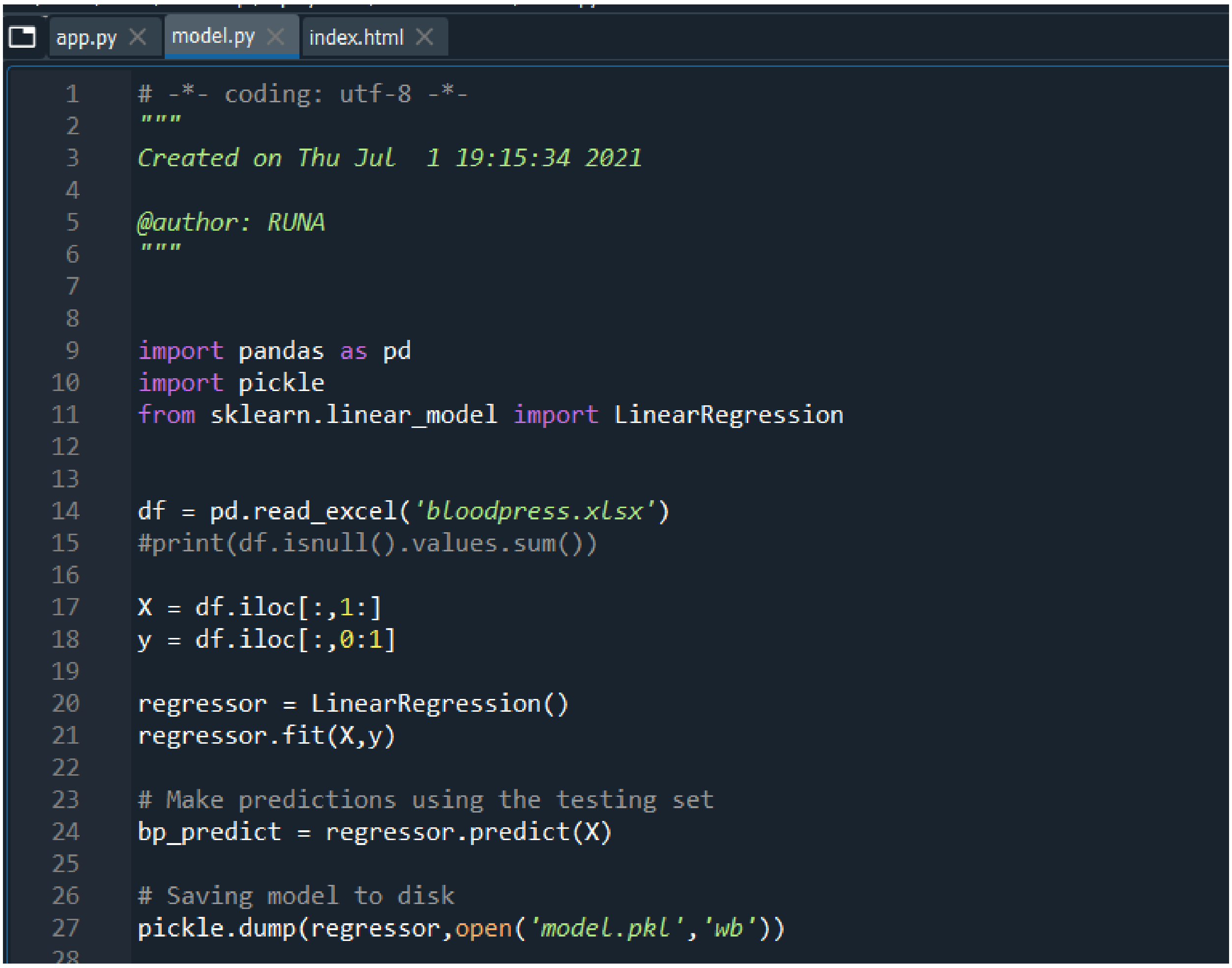


Name	Date Modified
static	09-07-2021 15:39
css	09-07-2021 15:39
templates	09-07-2021 15:39
index.html	09-07-2021 15:39
.gitignore	09-07-2021 15:21
app.py	03-07-2021 00:18
bloodpress.xlsx	02-07-2021 19:53
model.pkl	02-07-2021 22:33
model.py	04-07-2021 13:00
Procfile	09-07-2021 14:32
README.md	09-07-2021 15:21
requirements.txt	09-07-2021 14:36

Steps Involved in the Deployment

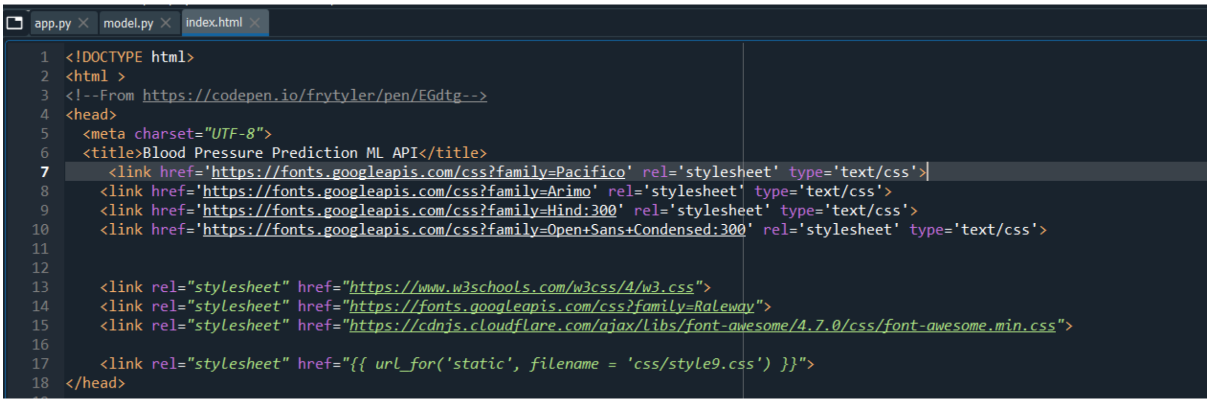
Step 1:

1. Train the Blood Pressure Predictor model
2. Use the pickle operation to serialize the machine learning algorithm (Linear Regression) and dumping the serialized format to the file **model.pkl** in a binary write format.
3. Save the file in **model.py**

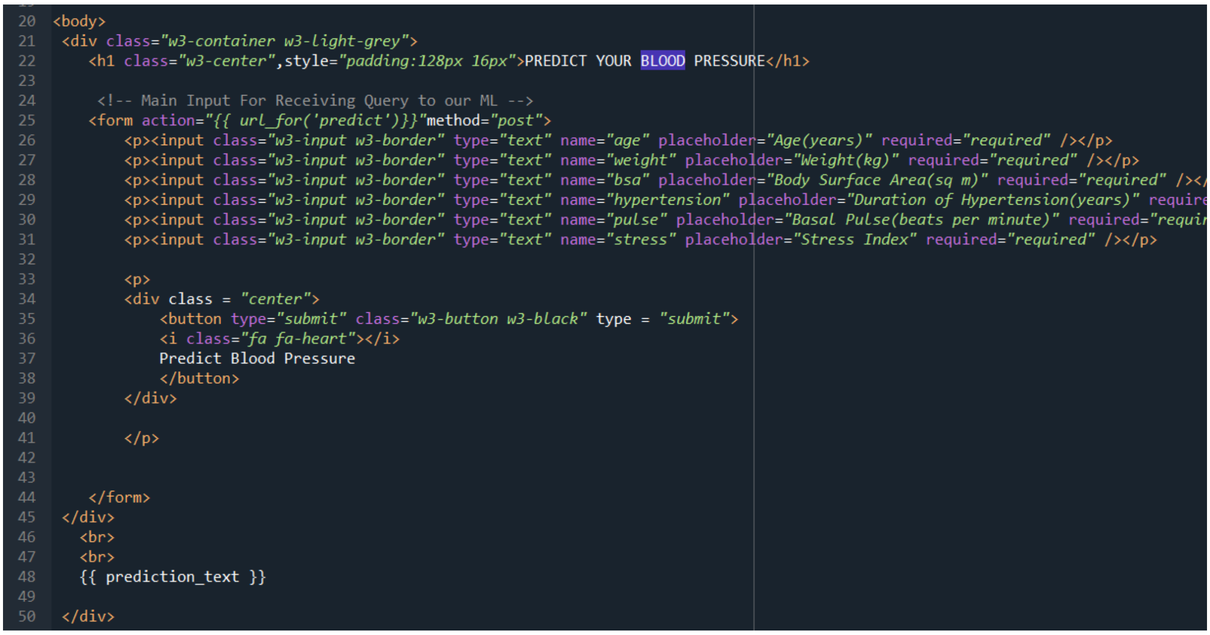


```python
# -*- coding: utf-8 -*-
"""
Created on Thu Jul  1 19:15:34 2021

@author: RUNA
"""



import pandas as pd
import pickle
from sklearn.linear_model import LinearRegression


df = pd.read_excel('bloodpress.xlsx')
#print(df.isnull().values.sum())

X = df.iloc[:,1:]
y = df.iloc[:,0:1]

regressor = LinearRegression()
regressor.fit(X,y)

# Make predictions using the testing set
bp_predict = regressor.predict(X)

# Saving model to disk
pickle.dump(regressor,open('model.pkl','wb'))
```

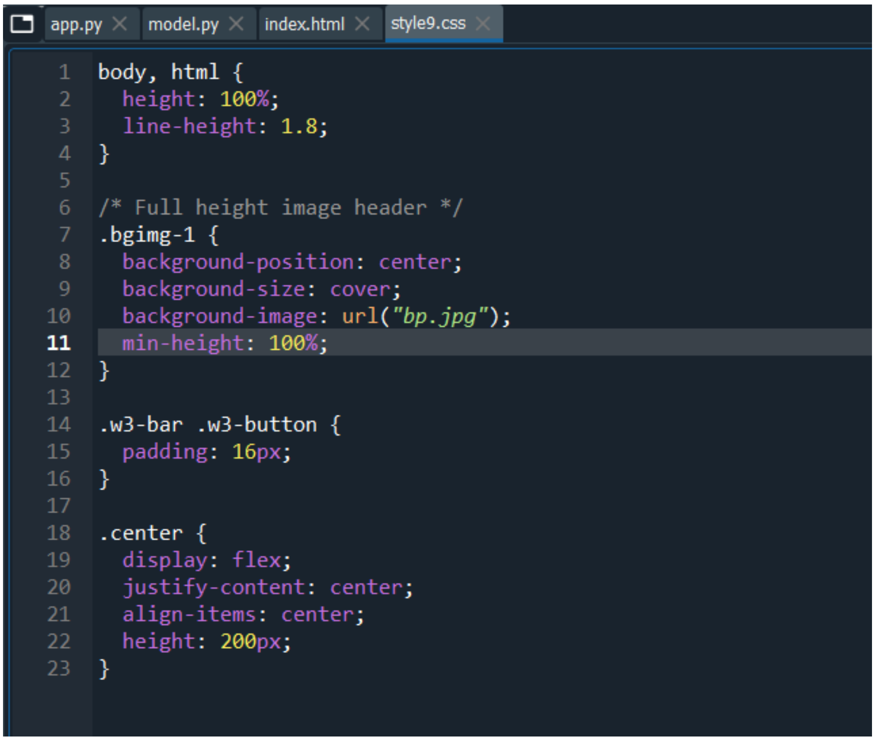
Step 2: Creating the HTML file and saving in **index.html** and **CSS** (for styling) file.



```html
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
  <meta charset="UTF-8">
  <title>Blood Pressure Prediction 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="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Raleway">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">

    <link rel="stylesheet" href="{{ url_for('static', filename = 'css/style9.css') }}">
</head>
```

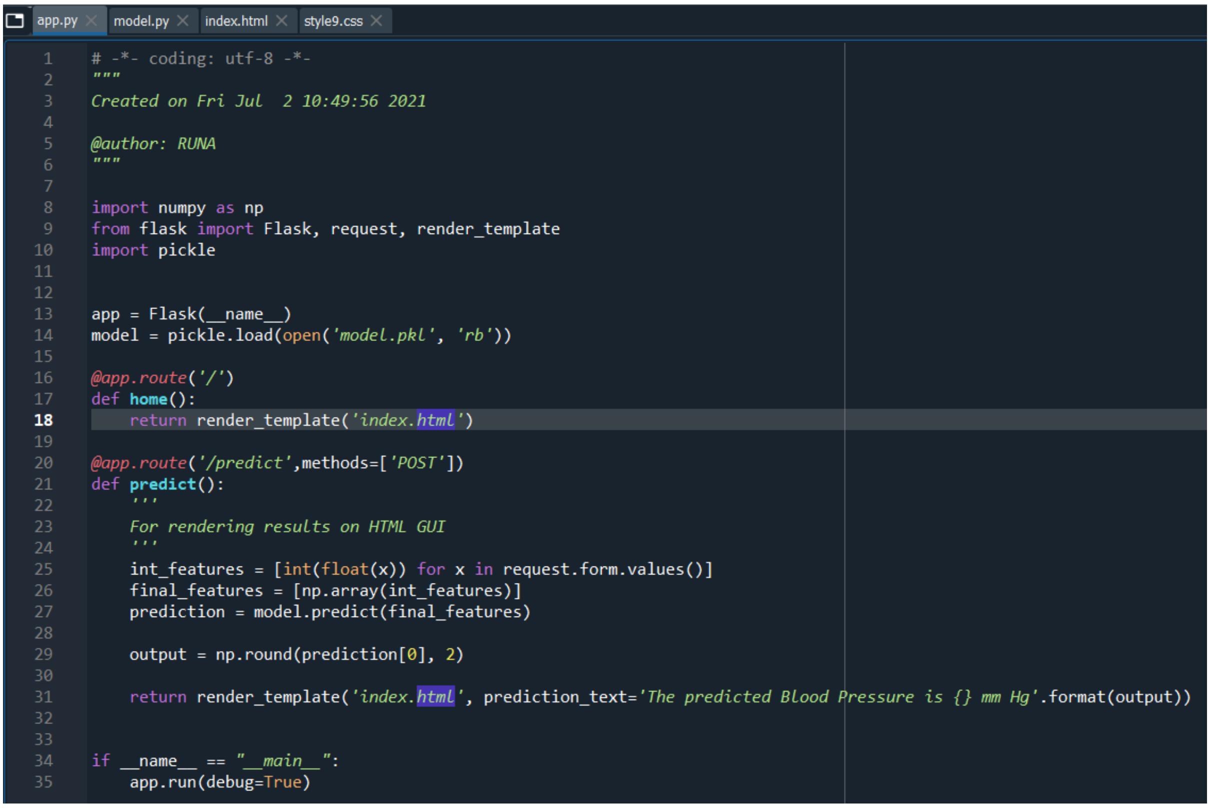


```html
<body>
 <div class="w3-container w3-light-grey">
    <h1 class="w3-center",style="padding:128px 16px">PREDICT YOUR BLOOD PRESSURE</h1>

     <!-- Main Input For Receiving Query to our ML -->
    <form action="{{ url_for('predict')}}"method="post">
        <p><input class="w3-input w3-border" type="text" name="age" placeholder="Age(years)" required="required" /></p>
        <p><input class="w3-input w3-border" type="text" name="weight" placeholder="Weight(kg)" required="required" /></p>
        <p><input class="w3-input w3-border" type="text" name="bsa" placeholder="Body Surface Area(sq m)" required="required" /><
        <p><input class="w3-input w3-border" type="text" name="hypertension" placeholder="Duration of Hypertension(years)" require
        <p><input class="w3-input w3-border" type="text" name="pulse" placeholder="Basal Pulse(beats per minute)" required="requir
        <p><input class="w3-input w3-border" type="text" name="stress" placeholder="Stress Index" required="required" /></p>

        <p>
        <div class = "center">
            <button type="submit" class="w3-button w3-black" type = "submit">
            <i class="fa fa-heart"></i>
            Predict Blood Pressure
            </button>
        </div>

        </p>


    </form>
 </div>
   <br>
   <br>
   {{ prediction_text }}

 </div>
```



```css
body, html {
  height: 100%;
  line-height: 1.8;
}

/* Full height image header */
.bgimg-1 {
  background-position: center;
  background-size: cover;
  background-image: url("bp.jpg");
  min-height: 100%;
}

.w3-bar .w3-button {
  padding: 16px;
}

.center {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 200px;
}
```

Step 3: Creating the web application file using **Flask**.

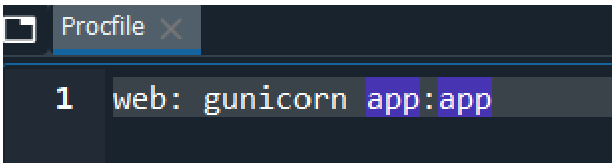


```python
# -*- coding: utf-8 -*-
"""
Created on Fri Jul  2 10:49:56 2021

@author: RUNA
"""

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
    '''
    int_features = [int(float(x)) for x in request.form.values()]
    final_features = [np.array(int_features)]
    prediction = model.predict(final_features)

    output = np.round(prediction[0], 2)

    return render_template('index.html', prediction_text='The predicted Blood Pressure is {} mm Hg'.format(output))


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

1. Creating the object of class **Flask**.
2. Whenever user visits that URL **'/',** index() method would be called automatically, and the index() method returns our main HTML page called **index.html.**
   The **flask.render_template()** looks for the this index.html file in the **templates** folder and dynamically generates/renders a HTML page for the end user
3. **@app.route ('/predict')** maps the **predict()** method with the **/predict** URL , this takes the input given by the user, does all the pre-processing and generates the final feature vector, runs the model on it and gets the final prediction.

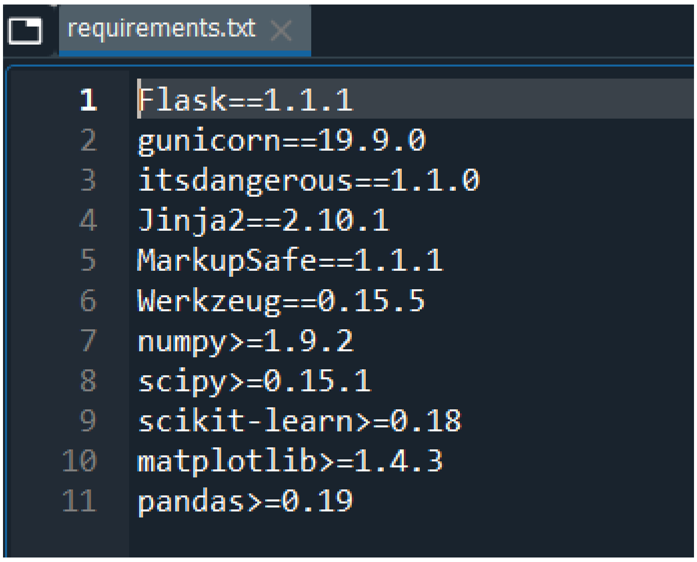
Step 4: Create a requirement.txt and Procfile (without any extensions)

Procfile tells **heroku** about a set of processes/commands that needs to be run beforehand.



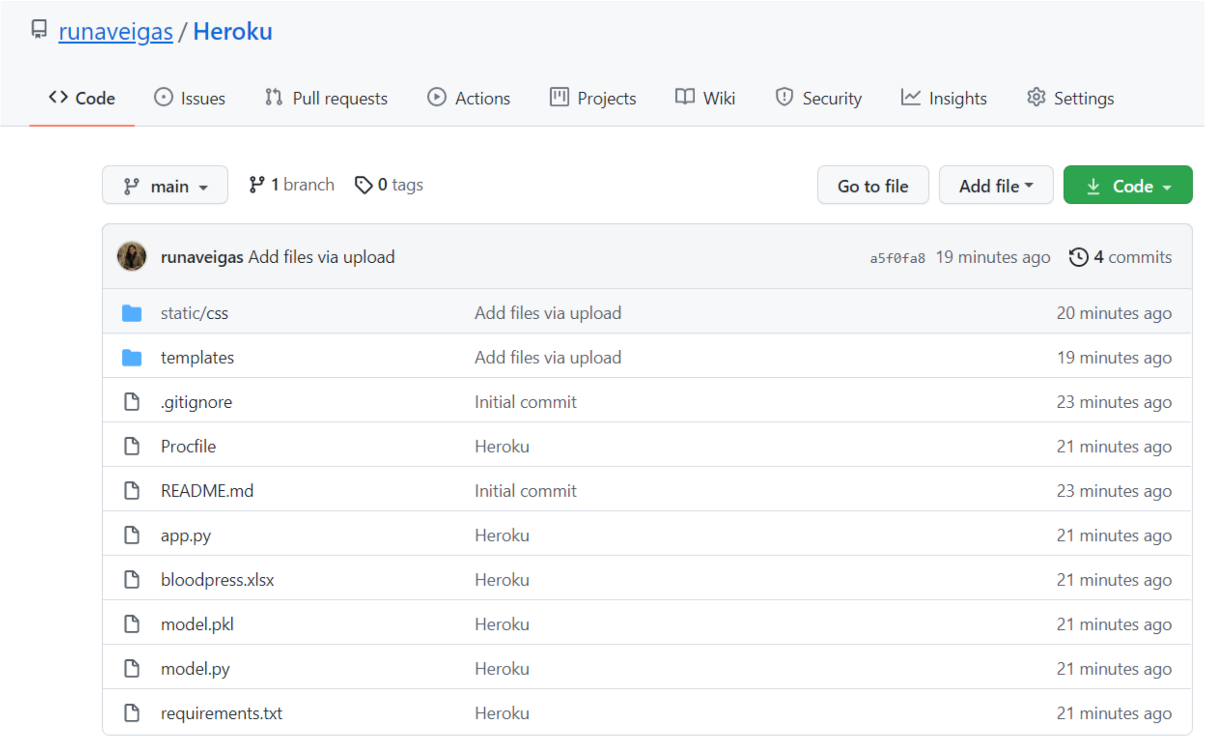
```
web: gunicorn app:app
```

requirements.txt tells **Heroku** the set of applications or libraries needed for the deployment of the application



```
Flask==1.1.1
gunicorn==19.9.0
itsdangerous==1.1.0
Jinja2==2.10.1
MarkupSafe==1.1.1
Werkzeug==0.15.5
numpy>=1.9.2
scipy>=0.15.1
scikit-learn>=0.18
matplotlib>=1.4.3
pandas>=0.19
```

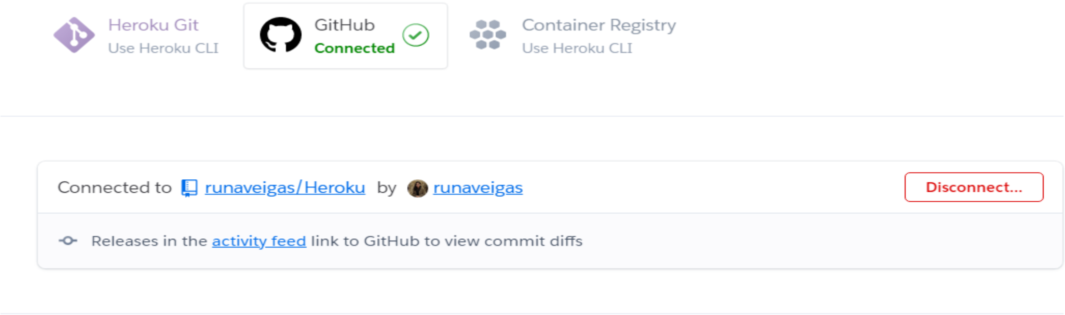
Step 5: Commit the changes to **git-hub**



runaveigas Add files via upload a5f0fa8 19 minutes ago 4 commits

static/css	Add files via upload	20 minutes ago
templates	Add files via upload	19 minutes ago
.gitignore	Initial commit	23 minutes ago
Procfile	Heroku	21 minutes ago
README.md	Initial commit	23 minutes ago
app.py	Heroku	21 minutes ago
bloodpress.xlsx	Heroku	21 minutes ago
model.pkl	Heroku	21 minutes ago
model.py	Heroku	21 minutes ago
requirements.txt	Heroku	21 minutes ago

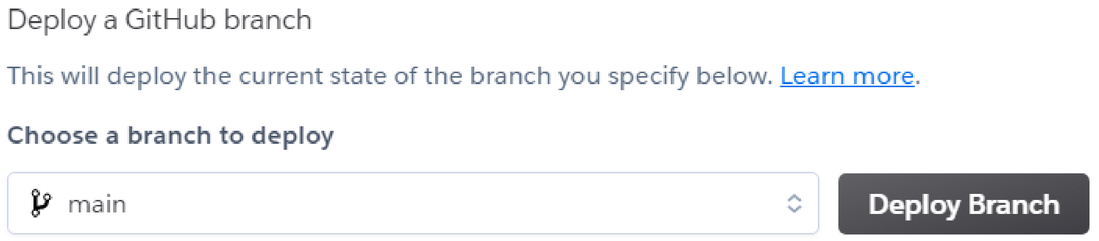
Step 6: Link and connect **Github** to **Heroku**.



Connected to runaveigas/Heroku by runaveigas

Releases in the activity feed link to GitHub to view commit diffs

Step 7: Deploy the application.



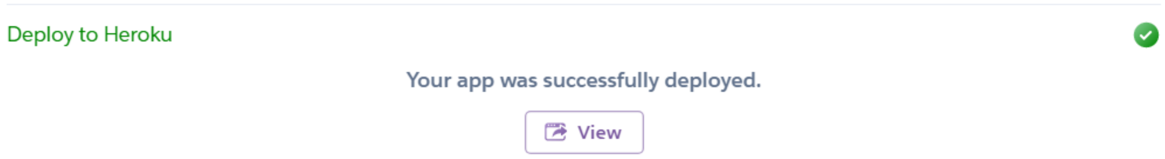
Deploy a GitHub branch

This will deploy the current state of the branch you specify below. Learn more.

Choose a branch to deploy

main

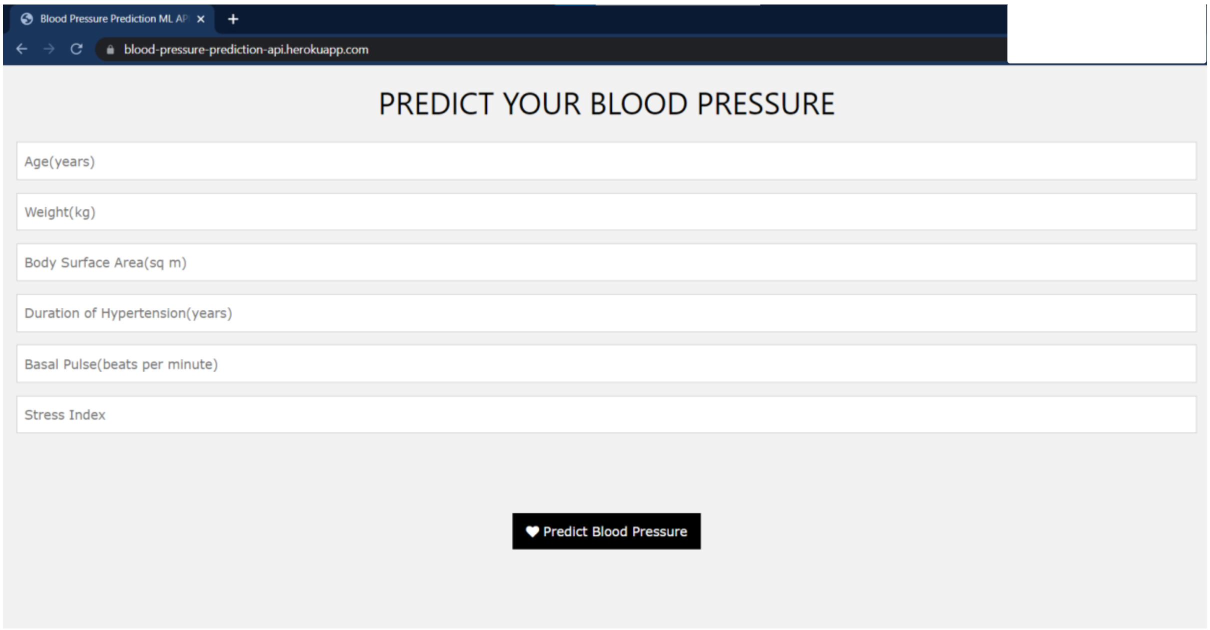
Step 8: After successful deployment we can view the application by clicking on **view**.



Deploy to Heroku

Your app was successfully deployed.

Step 9: Final Result.



Blood Pressure Prediction ML AP
blood-pressure-prediction-api.herokuapp.com

PREDICT YOUR BLOOD PRESSURE

Age(years)

Weight(kg)

Body Surface Area(sq m)

Duration of Hypertension(years)

Basal Pulse(beats per minute)

Stress Index

**************Thank You**************