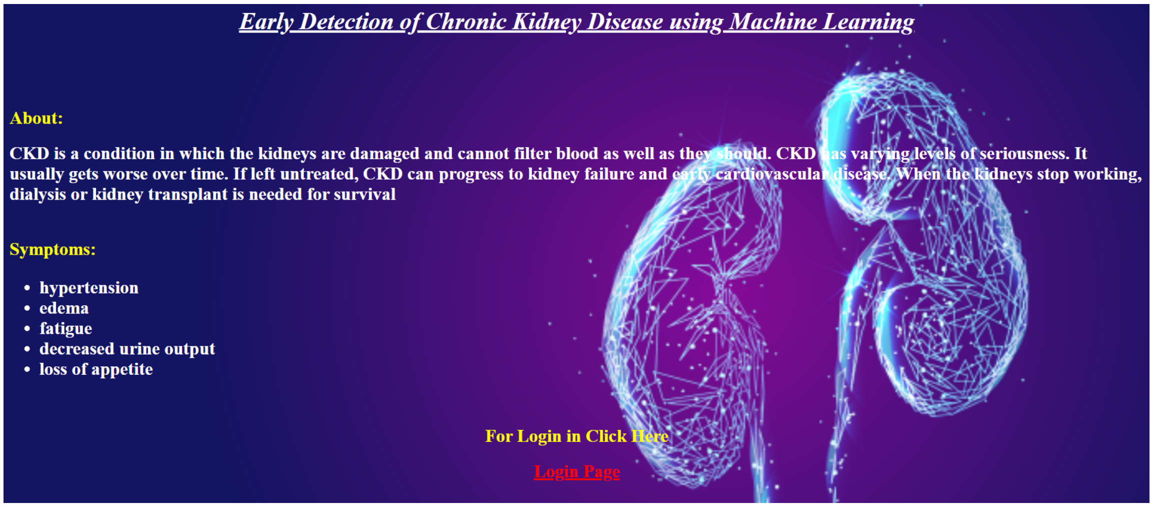
Early Detection of Chronic Kidney Disease Using Machine Learning

Team ID	**PNT2022TMID03830**
Team Leader	Robsi Rani
Team Members	Swathi K Subashini V Mythilee KL

Sprint 4:

LOCAL DEPLOYMENT RESULTS:



Early Detection of Chronic Kidney Disease using Machine Learning

About:

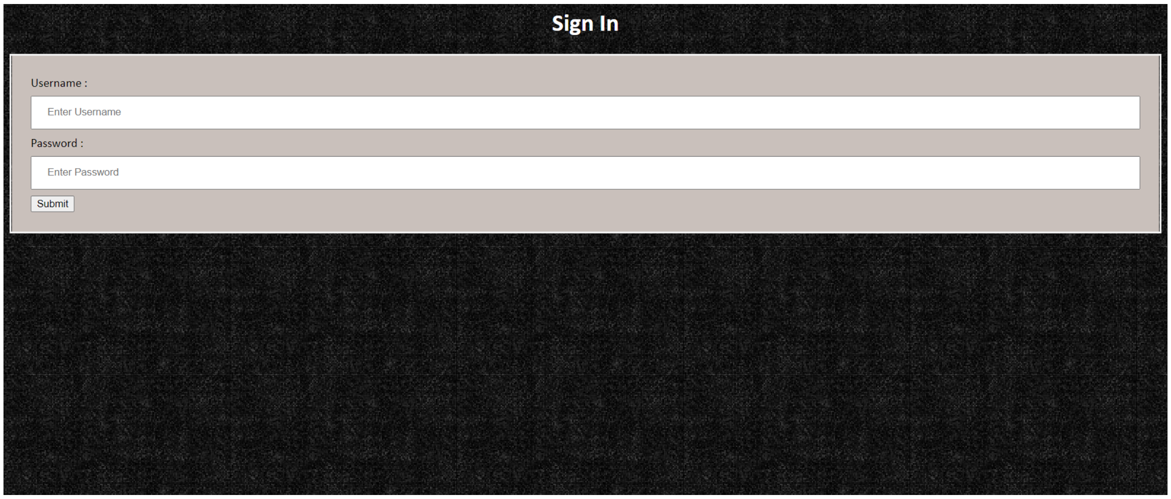
CKD is a condition in which the kidneys are damaged and cannot filter blood as well as they should. CKD has varying levels of seriousness. It usually gets worse over time. If left untreated, CKD can progress to kidney failure and early cardiovascular disease. When the kidneys stop working, dialysis or kidney transplant is needed for survival

Symptoms:

- **hypertension**
- **edema**
- **fatigue**
- **decreased urine output**
- **loss of appetite**

For Login in Click Here

Login Page



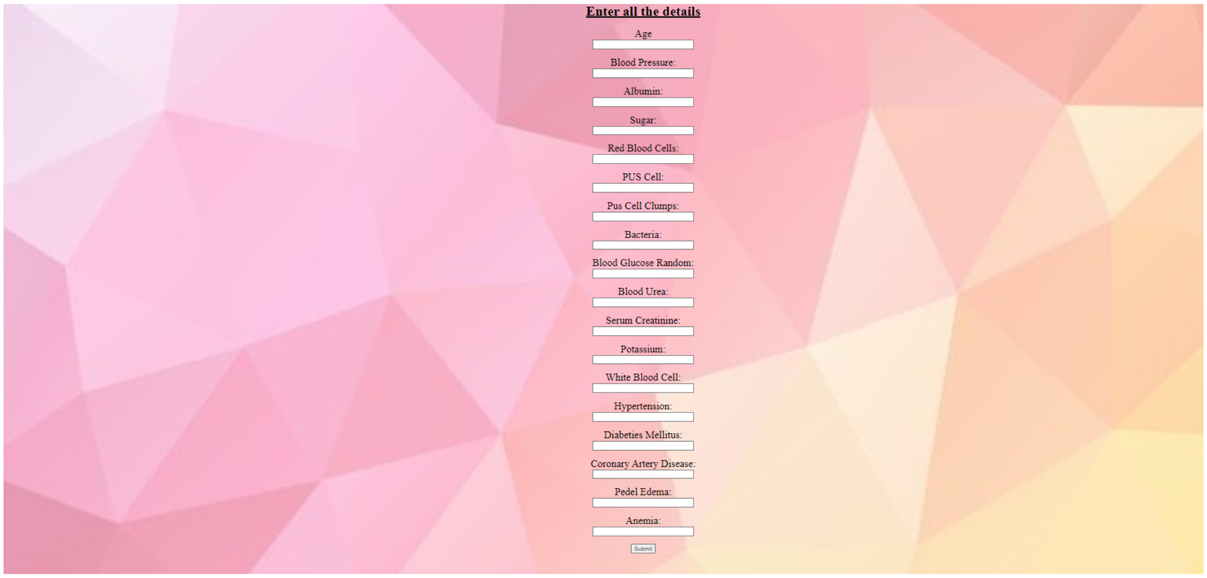
Sign In

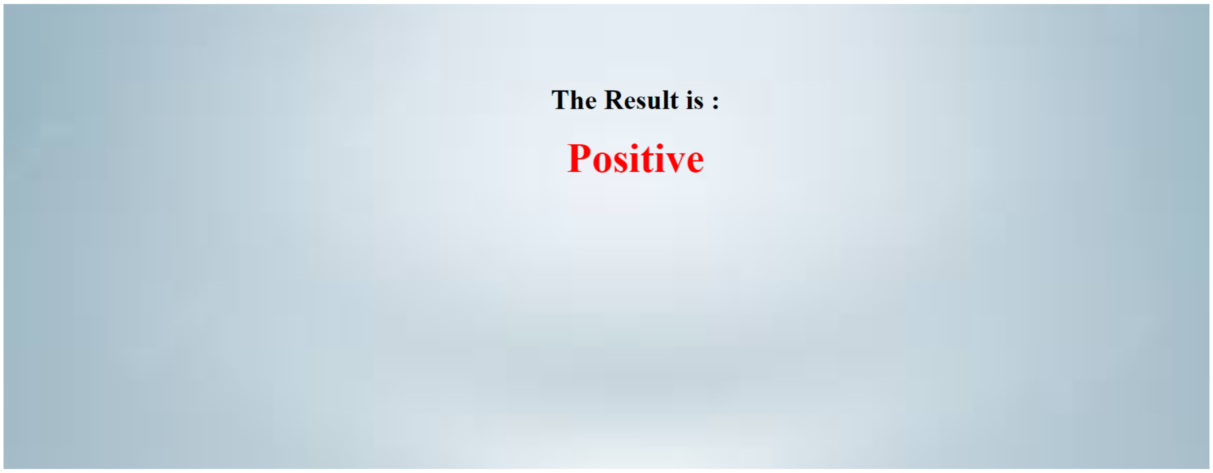
Username :

Enter Username

Password :

Enter Password

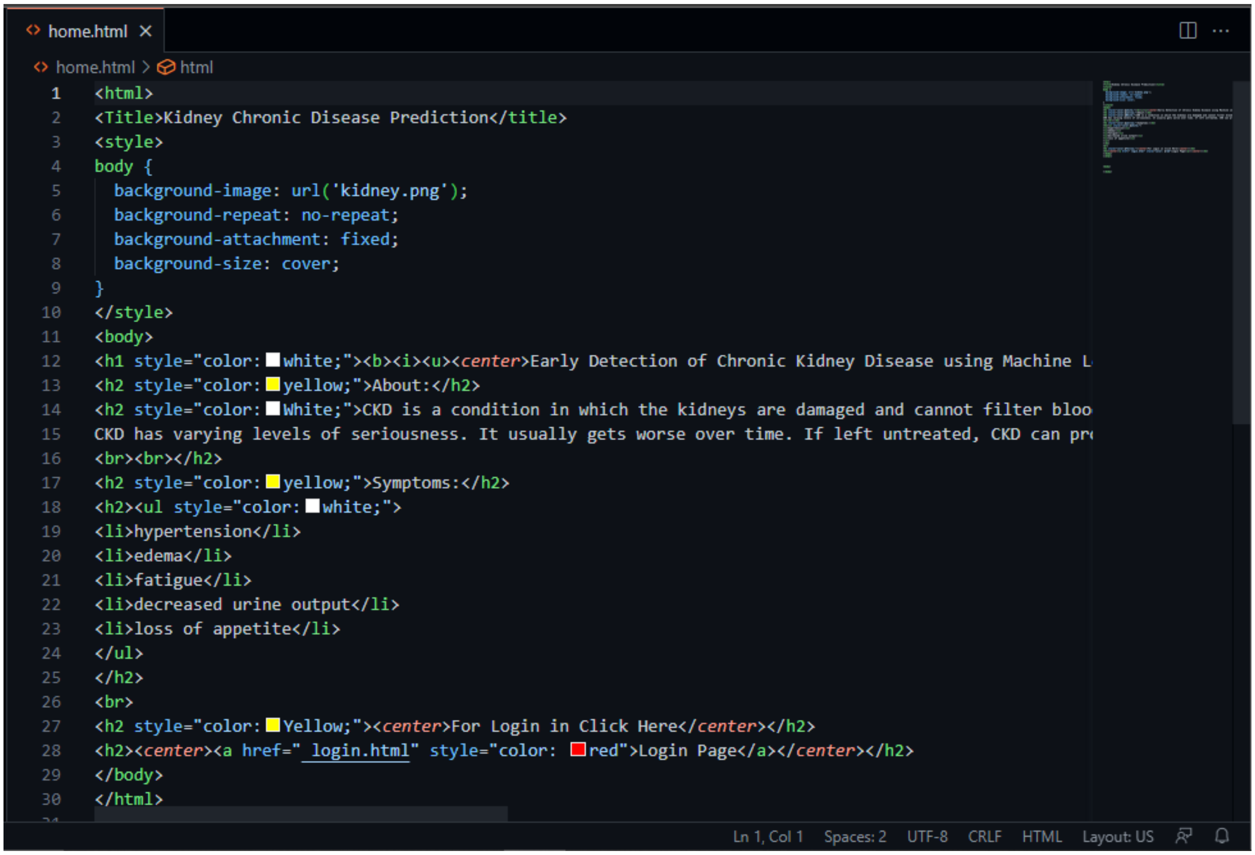




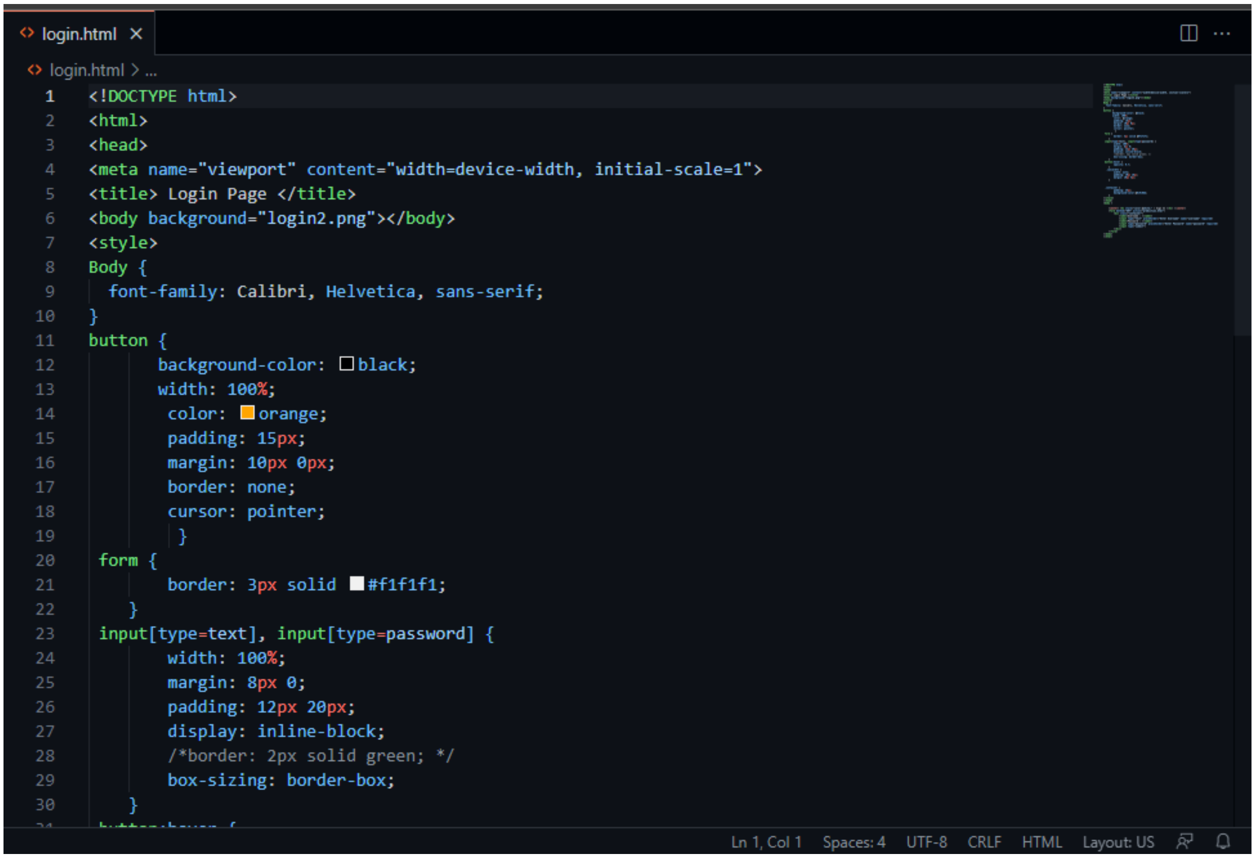
The Result is :

Positive

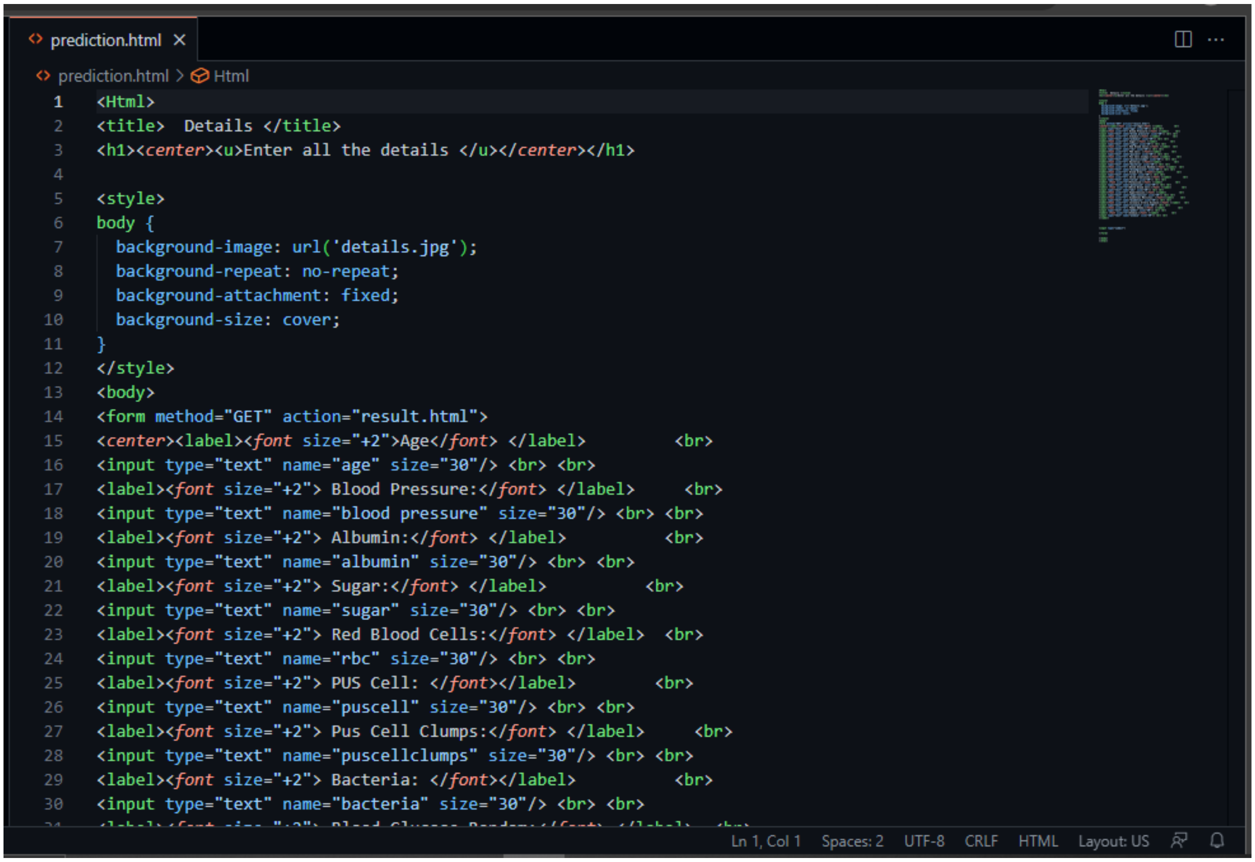
Creation of HTML Files:



```html
<html>
<Title>Kidney Chronic Disease Prediction</title>
<style>
body {
  background-image: url('kidney.png');
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-size: cover;
}
</style>
<body>
<h1 style="color:white;"><b><i><u><center>Early Detection of Chronic Kidney Disease using Machine L
<h2 style="color:yellow;">About:</h2>
<h2 style="color:White;">CKD is a condition in which the kidneys are damaged and cannot filter bloo
CKD has varying levels of seriousness. It usually gets worse over time. If left untreated, CKD can pr
<br><br></h2>
<h2 style="color:yellow;">Symptoms:</h2>
<h2><ul style="color:white;">
<li>hypertension</li>
<li>edema</li>
<li>fatigue</li>
<li>decreased urine output</li>
<li>loss of appetite</li>
</ul>
</h2>
<br>
<h2 style="color:Yellow;"><center>For Login in Click Here</center></h2>
<h2><center><a href=" login.html" style="color: red">Login Page</a></center></h2>
</body>
</html>
```

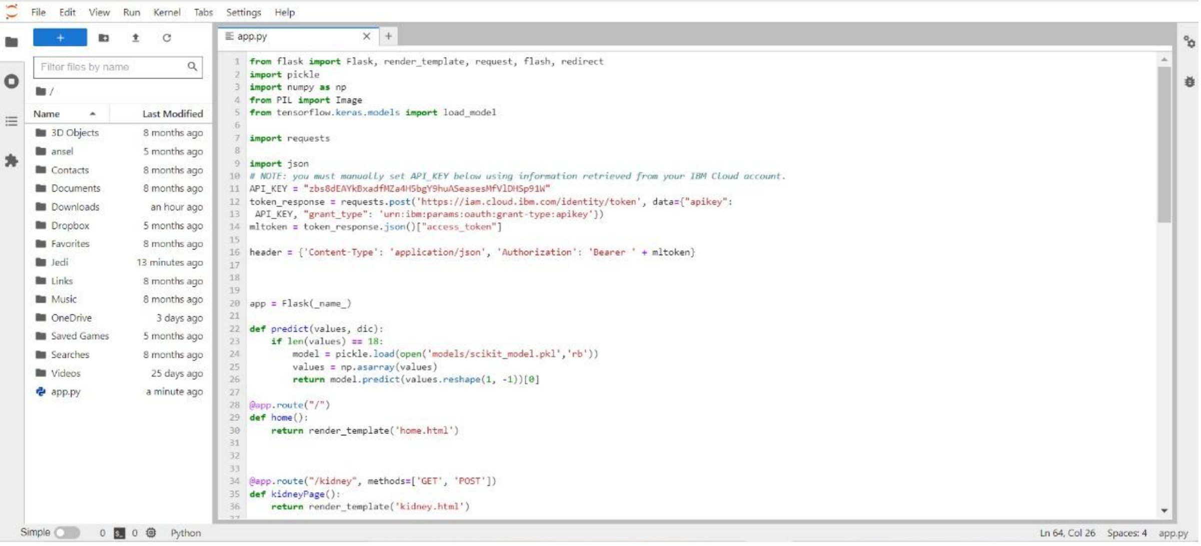


```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<title> Login Page </title>
<body background="login2.png"></body>
<style>
Body {
  font-family: Calibri, Helvetica, sans-serif;
}
button {
       background-color: black;
       width: 100%;
        color: orange;
        padding: 15px;
        margin: 10px 0px;
        border: none;
        cursor: pointer;
         }
 form {
        border: 3px solid #f1f1f1;
    }
 input[type=text], input[type=password] {
        width: 100%;
        margin: 8px 0;
        padding: 12px 20px;
        display: inline-block;
        /*border: 2px solid green; */
        box-sizing: border-box;
    }
 button:hover {
```



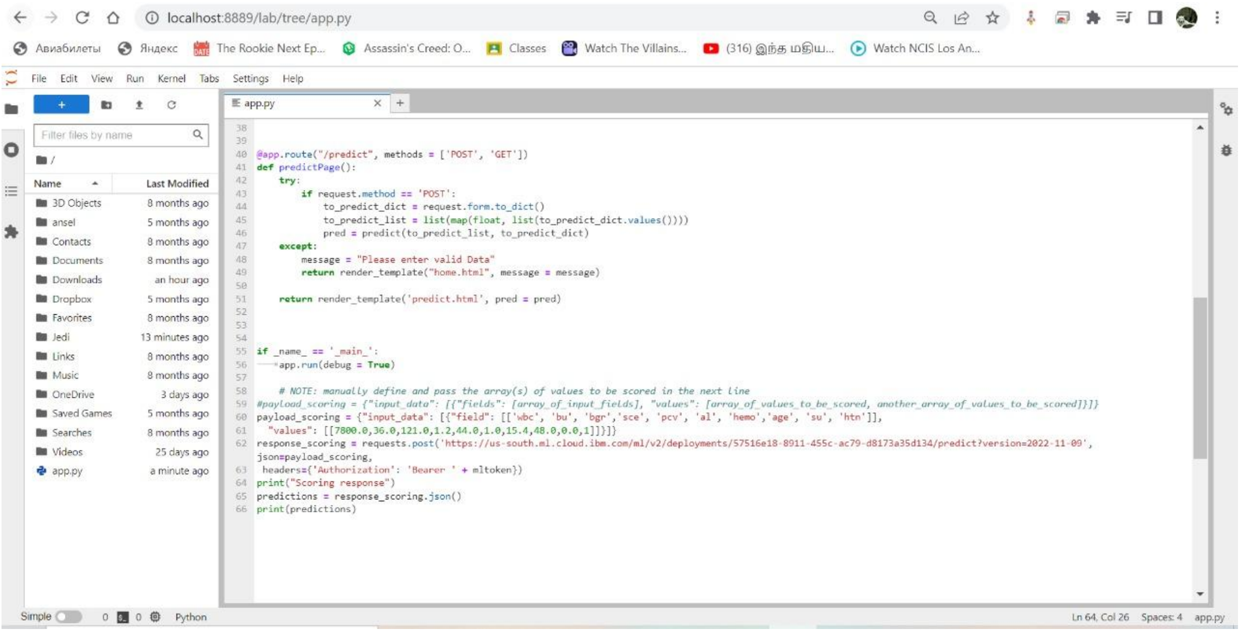
```html
<Html>
<title>  Details </title>
<h1><center><u>Enter all the details </u></center></h1>

<style>
body {
  background-image: url('details.jpg');
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-size: cover;
}
</style>
<body>
<form method="GET" action="result.html">
<center><label><font size="+2">Age</font> </label>         <br>
<input type="text" name="age" size="30"/> <br> <br>
<label><font size="+2"> Blood Pressure:</font> </label>     <br>
<input type="text" name="blood pressure" size="30"/> <br> <br>
<label><font size="+2"> Albumin:</font> </label>          <br>
<input type="text" name="albumin" size="30"/> <br> <br>
<label><font size="+2"> Sugar:</font> </label>          <br>
<input type="text" name="sugar" size="30"/> <br> <br>
<label><font size="+2"> Red Blood Cells:</font> </label>  <br>
<input type="text" name="rbc" size="30"/> <br> <br>
<label><font size="+2"> PUS Cell: </font></label>        <br>
<input type="text" name="puscell" size="30"/> <br> <br>
<label><font size="+2"> Pus Cell Clumps:</font> </label>     <br>
<input type="text" name="puscellclumps" size="30"/> <br> <br>
<label><font size="+2"> Bacteria: </font></label>          <br>
<input type="text" name="bacteria" size="30"/> <br> <br>
<label><font size="+2"> Blood Glucose Random:</font> </label>   <br>
```

Integrate Flask With Scoring End Point:



Name	Last Modified
3D Objects	8 months ago
ansel	5 months ago
Contacts	8 months ago
Documents	8 months ago
Downloads	an hour ago
Dropbox	5 months ago
Favorites	8 months ago
Jedi	13 minutes ago
Links	8 months ago
Music	8 months ago
OneDrive	3 days ago
Saved Games	5 months ago
Searches	8 months ago
Videos	25 days ago
app.py	a minute ago

```
from flask import Flask, render_template, request, flash, redirect
import pickle
import numpy as np
from PIL import Image
from tensorflow.keras.models import load_model

import requests

import json
# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
API_KEY = "zbs8dEAYkBxadfMZa4H5bgY9huASeasesMfVlDHSp91W"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}



app = Flask(_name_)

def predict(values, dic):
    if len(values) == 18:
        model = pickle.load(open('models/scikit_model.pkl','rb'))
        values = np.asarray(values)
        return model.predict(values.reshape(1, -1))[0]

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



@app.route("/kidney", methods=['GET', 'POST'])
def kidneyPage():
    return render_template('kidney.html')
```



Name	Last Modified
3D Objects	8 months ago
ansel	5 months ago
Contacts	8 months ago
Documents	8 months ago
Downloads	an hour ago
Dropbox	5 months ago
Favorites	8 months ago
Jedi	13 minutes ago
Links	8 months ago
Music	8 months ago
OneDrive	3 days ago
Saved Games	5 months ago
Searches	8 months ago
Videos	25 days ago
app.py	a minute ago

```


@app.route("/predict", methods = ['POST', 'GET'])
def predictPage():
    try:
        if request.method == 'POST':
            to_predict_dict = request.form.to_dict()
            to_predict_list = list(map(float, list(to_predict_dict.values())))
            pred = predict(to_predict_list, to_predict_dict)
    except:
        message = "Please enter valid Data"
        return render_template("home.html", message = message)

    return render_template('predict.html', pred = pred)




if _name_ == '_main_':
    app.run(debug = True)

    # NOTE: manually define and pass the array(s) of values to be scored in the next line
#payload_scoring = {"input_data": [{"fields": [array_of_input_fields], "values": [array_of_values_to_be_scored, another_array_of_values_to_be_scored]}]}
payload_scoring = {"input_data": [{"field": [['wbc', 'bu', 'bgr','sce', 'pcv', 'al', 'hemo','age', 'su', 'htn']],
  "values": [[7800.0,36.0,121.0,1.2,44.0,1.0,15.4,48.0,0.0,1]]}]}
response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v2/deployments/57516e18-8911-455c-ac79-d8173a35d134/predict?version=2022-11-09',
json=payload_scoring,
 headers={'Authorization': 'Bearer ' + mltoken})
print("Scoring response")
predictions = response_scoring.json()
print(predictions)
```