Diabetes\_Prediction

***AI&ML Project Report***

***Submitted in partial fulfillment of the***

***Requirements for the award of the Degree of***

**BACHELOR OF ENGINEERING IN**

**INFORMATION TECHNOLOGY**

*BY*

**<Sharath Chandra B> <1602-22-737-175>**

**<Shiva Pallavi R> <1602-22-737-165>**

Under the guidance of Mrs. Satyadevi



Department of InformationTechnology

Vasavi College of Engineering(Autonomous)

(Affiliated to Osmania University)

Ibrahimbagh, Hyderabad-31 2024-25

# DECLARATION BY THE CANDIDATE

We, **Sharath Chandra B** and **Shiva Pallavi R**, bearing hall ticket numbers, **1602-22-737-175** and **1602-22-737-165**, hereby declare that the project report entitled **“Diabetes\_Prediction”** Department of Information Technology, Vasavi College of Engineering, Hyderabad, is submitted in partial fulfillment of the requirement for the award of the degree of Bachelor of Engineering in Information Technology

This is a record of bonafide work carried out by me and the results embodied in this project report have not been submitted to any other university or institute for the award of any other degree or diploma.

**<Sharath Chandra B> <1602-22-737-175>**

**<Shiva Pallavi R> <1602-22-737-165>**

**(Faculty In-Charge) (Head,Dept Of IT)**

### **AIM AND PRIORITY OF THE PROJECT**

The aim of the project is to develop a user-friendly, reliable, and accurate system for predicting diabetes based on clinical parameters using machine learning. The priority is to aid early detection and classification of diabetes (Diabetic/Non-Diabetic), allowing individuals and healthcare providers to take timely preventive or curative measures.

### INTRODUCTION

Diabetes is a chronic disease affecting millions globally, characterized by high blood sugar levels. Early detection is crucial to managing the disease and avoiding complications. This project leverages logistic regression and a web-based application to predict diabetes based on patient data. The system combines clinical parameters with machine learning to provide accurate results through a simple and intuitive interface.

### OBJECTIVES

1. **Develop a Predictive Model:** Train a logistic regression model to classify individuals as diabetic or non-diabetic.
2. **Enhance Accuracy:** Use data preprocessing techniques like standardization and label encoding to improve prediction accuracy.
3. **User-Friendly Interface:** Create a web-based application for healthcare providers and individuals to input clinical data and get instant results.
4. **Real-Time Predictions:** Ensure the system processes input data quickly for real-time diabetes prediction.
5. **Scalability:** Design the system for potential future integration with larger datasets and other advanced machine learning algorithms.

### ARCHITECTURE AND TECHNOLOGY USED

#### Architecture:

1. **Data Preprocessing:**
   * Cleaned and encoded the dataset to handle categorical variables.
   * Standardized features for better performance.
2. **Model Training:**
   * Logistic regression algorithm used for classification.
   * Trained using historical data with an 80-20 train-test split.
3. **Model Deployment:**
   * Trained model and scaler saved as .pkl files using joblib.
   * Flask web framework used to serve predictions through a web interface.
4. **Web Interface:**
   * HTML template renders the prediction form and displays results.

### TECHNOLOGIES USED

1. **Python Libraries:**
   * **Pandas:** For data manipulation and cleaning.
   * **Scikit-Learn:** For preprocessing, training, and evaluating the machine learning model.
   * **Joblib:** For saving and loading the trained model and scaler.
2. **Web Framework:**
   * **Flask:** For creating a lightweight web server and routing functionalities.
3. **Frontend:**
   * HTML, CSS: For designing a responsive and user-friendly interface.
4. **Deployment Tools:**
   * Flask’s development server for local hosting.

### ADDITIONAL DEPENDENCIES

1. **Flask-Related:**
   * Flask framework installed via pip install flask.
2. **Machine Learning Dependencies:**
   * Scikit-Learn: Ensure compatibility with Python version.
   * Joblib: For model serialization and loading.
3. **Browser Compatibility:**
   * Ensure compatibility with modern web browsers for the interface.
4. **Dataset Requirements:**
   * Pima Indians Diabetes Dataset or similar datasets cleaned and formatted with required features like age, urea, HbA1c, cholesterol, and more.
5. **Testing Tools:**
   * Postman: For testing Flask endpoints during development.

**Source Code:**

**diabetes\_model.py**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler, LabelEncoder

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score

import joblib

# Load dataset

file\_path = 'D:/projects/Diabetes\_Prediction/Dataset of Diabetes .csv'

diabetes\_data = pd.read\_csv(file\_path)4

# Drop irrelevant columns

diabetes\_data\_cleaned = diabetes\_data.drop(columns=["ID", "No\_Pation"])

# Encode categorical variables

label\_encoder = LabelEncoder()

diabetes\_data\_cleaned["Gender"] = label\_encoder.fit\_transform(diabetes\_data\_cleaned["Gender"])

diabetes\_data\_cleaned["CLASS"] = label\_encoder.fit\_transform(diabetes\_data\_cleaned["CLASS"])

# Separate features and target

X = diabetes\_data\_cleaned.drop(columns=["CLASS"])

y = diabetes\_data\_cleaned["CLASS"]

# Normalize the features

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)

# Train a logistic regression model

logistic\_model = LogisticRegression(random\_state=42, max\_iter=1000)

logistic\_model.fit(X\_train, y\_train)

# Evaluate the model

y\_pred = logistic\_model.predict(X\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

print(f"Model Accuracy: {accuracy \* 100:.2f}%")

# Save the trained model and scaler

joblib.dump(logistic\_model, "diabetes\_detection\_model.pkl")

joblib.dump(scaler, "scaler.pkl")

**app.py**

from flask import Flask, request, jsonify, render\_template

import joblib

import numpy as np

# Load the trained model and scaler

model = joblib.load("diabetes\_detection\_model.pkl")

scaler = joblib.load("scaler.pkl")

# Initialize Flask app

app = Flask(\_\_name\_\_)

@app.route("/")

def home():

return render\_template("index.html")

@app.route("/predict", methods=["POST"])

def predict():

# Get form data

data = [float(request.form[field]) for field in [

"Gender", "AGE", "Urea", "Cr", "HbA1c", "Chol", "TG", "HDL", "LDL", "VLDL", "BMI"

]]

# Preprocess input

data\_scaled = scaler.transform([data])

# Predict

prediction = model.predict(data\_scaled)[0]

result = "Diabetic" if prediction else "Non-Diabetic"

return render\_template("index.html", prediction=result)

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=True)

**templates/index.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Diabetes Detection</title>

<style>

body {

font-family: Arial, sans-serif;

margin: 20px;

padding: 20px;

background-color: #f4f4f4;

}

.container {

max-width: 500px;

margin: auto;

background: white;

padding: 20px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

input, button {

width: 90%;

margin: 10px 0;

padding: 10px;

font-size: 16px;

}

button {

background-color: #007BFF;

color: white;

border: None;

border-radius: 5px;

cursor: pointer;

}

button:hover {

background-color: #0056b3;

}

.result {

font-size: 20px;

font-weight: bold;

color: green;

text-align: center;

}

</style>

</head>

<body>

<div class="container">

<h1>Diabetes Detection</h1>

<form action="/predict" method="post">

<input type="text" name="Gender" placeholder="Gender (0 = Female, 1 = Male)" required>

<input type="number" name="AGE" placeholder="Age" required>

<input type="number" name="Urea" placeholder="Urea" step="any" required>

<input type="number" name="Cr" placeholder="Creatinine" step="any" required>

<input type="number" name="HbA1c" placeholder="HbA1c" step="any" required>

<input type="number" name="Chol" placeholder="Cholesterol" step="any" required>

<input type="number" name="TG" placeholder="Triglycerides" step="any" required>

<input type="number" name="HDL" placeholder="HDL" step="any" required>

<input type="number" name="LDL" placeholder="LDL" step="any" required>

<input type="number" name="VLDL" placeholder="VLDL" step="any" required>

<input type="number" name="BMI" placeholder="BMI" required>

<button type="submit">Detect</button>

</form>

{% if prediction %}

<div class="result">{{ prediction }}</div>

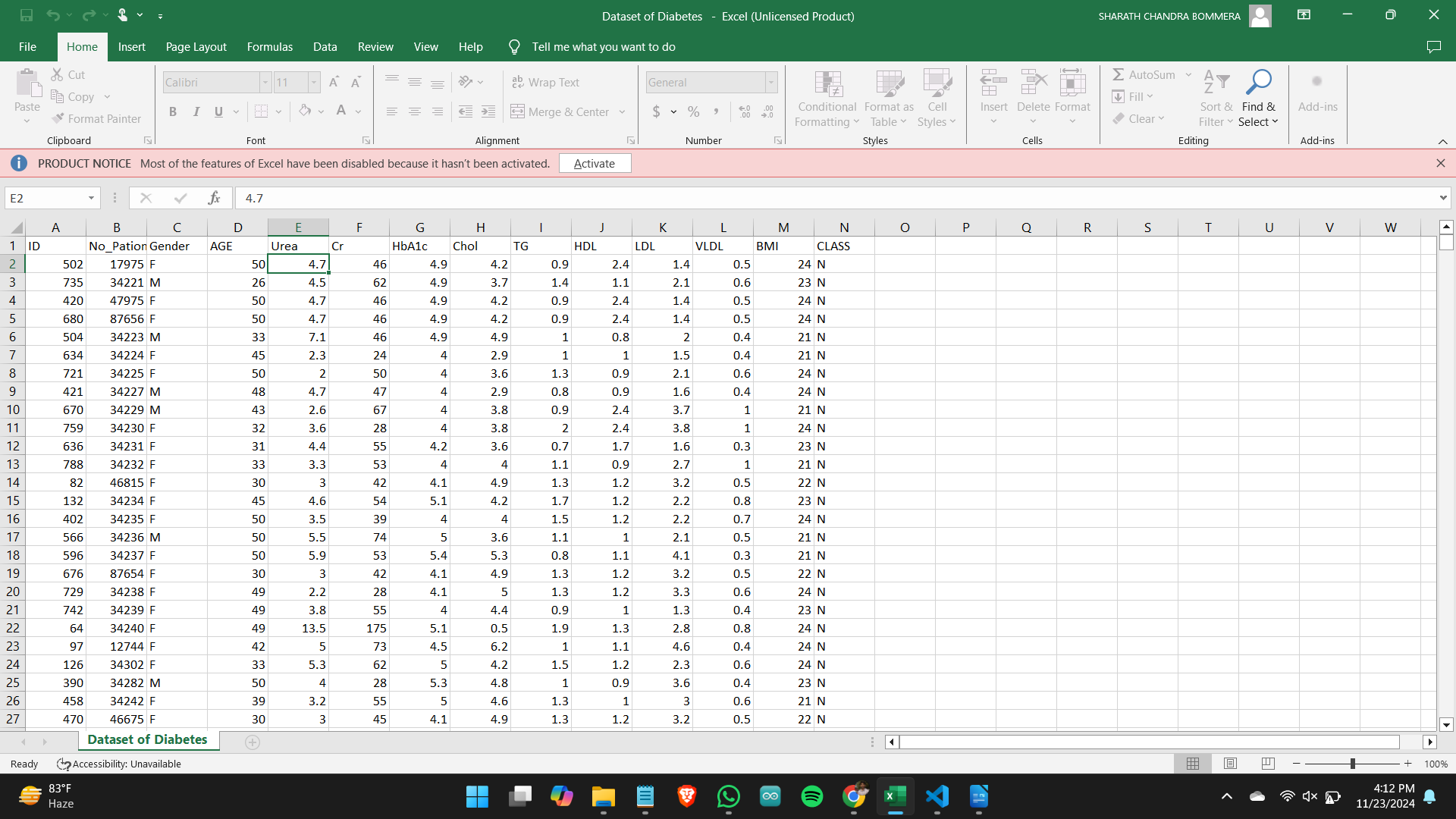
{% endif %}

</div>

</body>

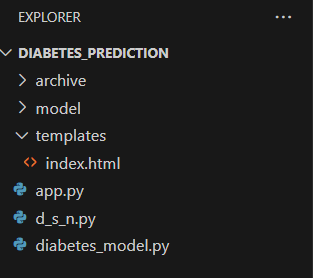
</html>

**Dataset:**

****

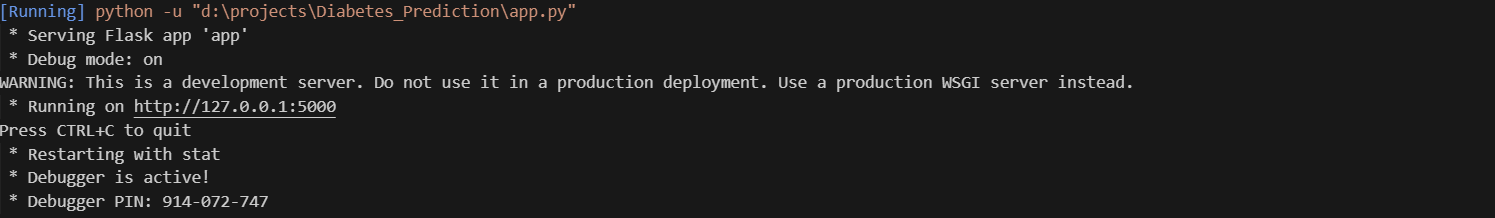
**OUTPUT:**

**Project Folder :**

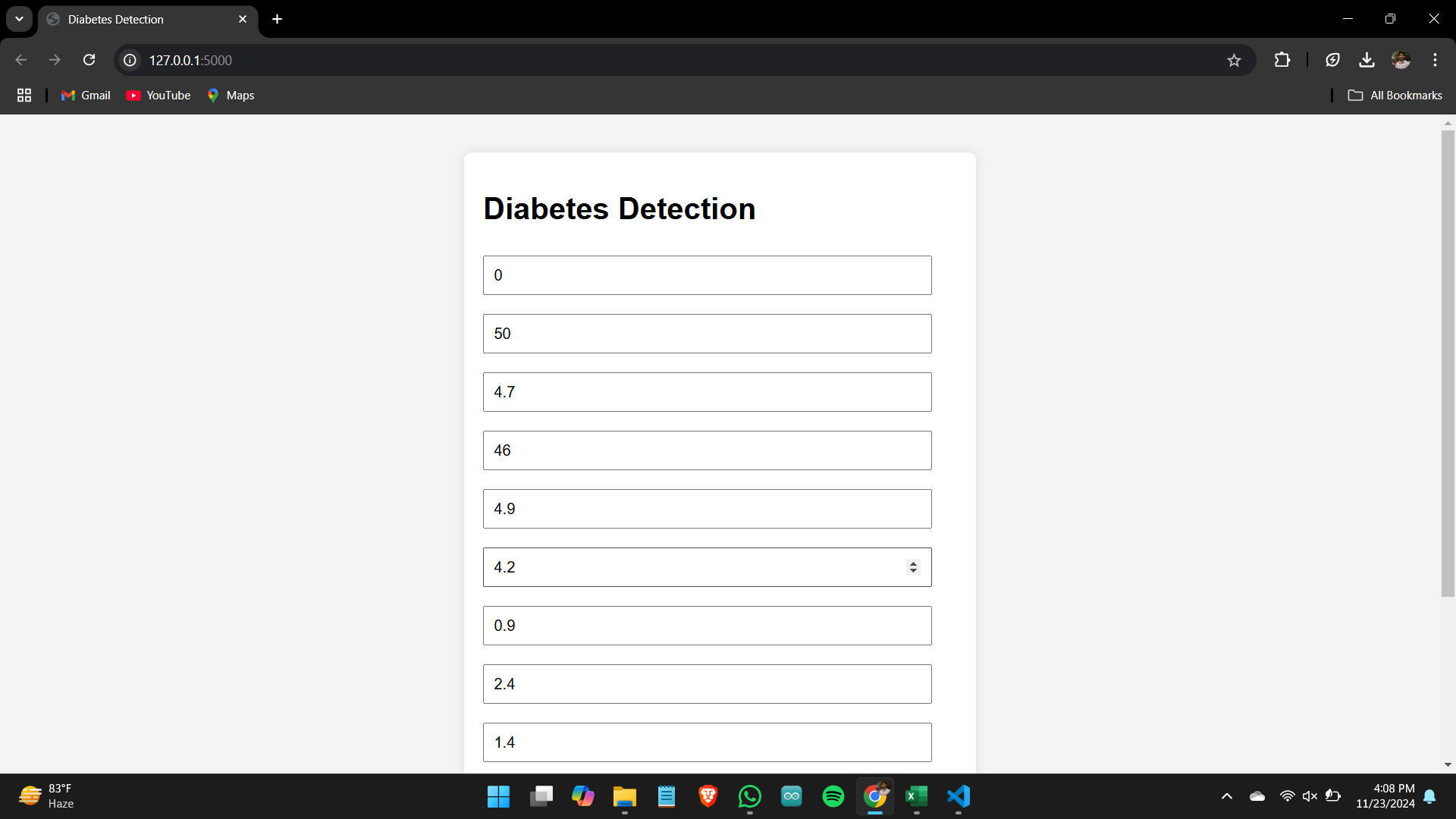
****

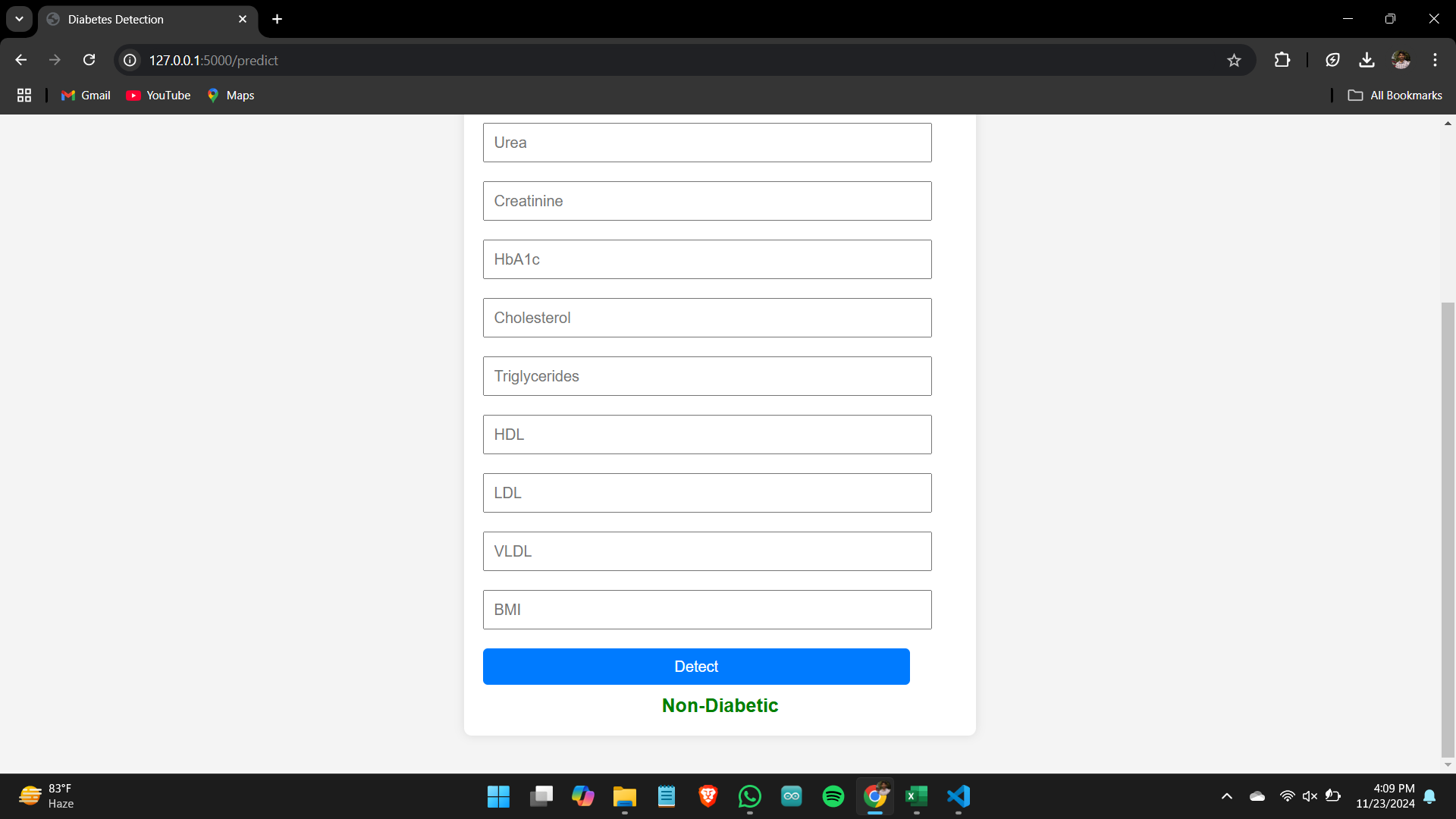
**Model Score:**

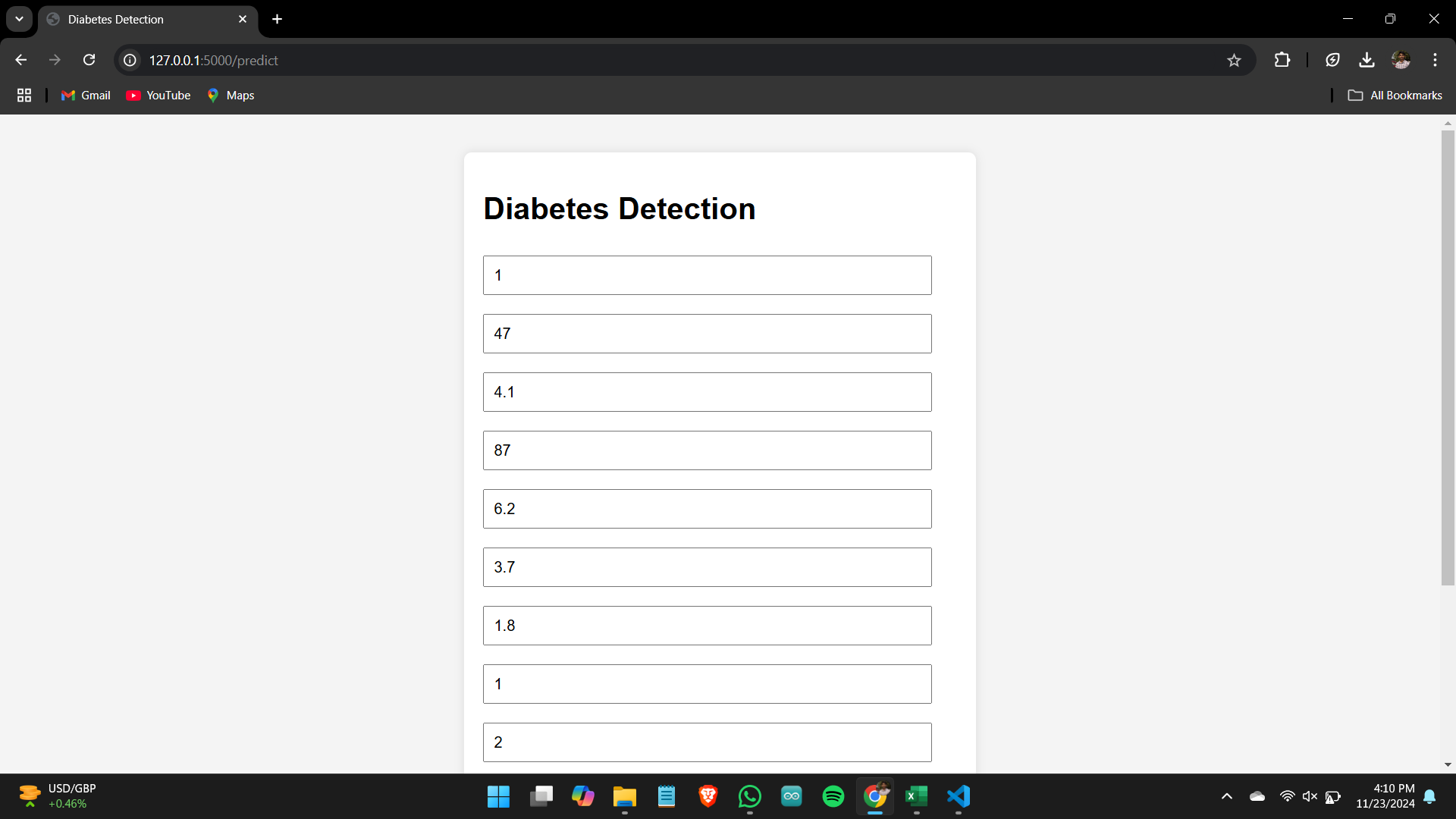
**93.5%**

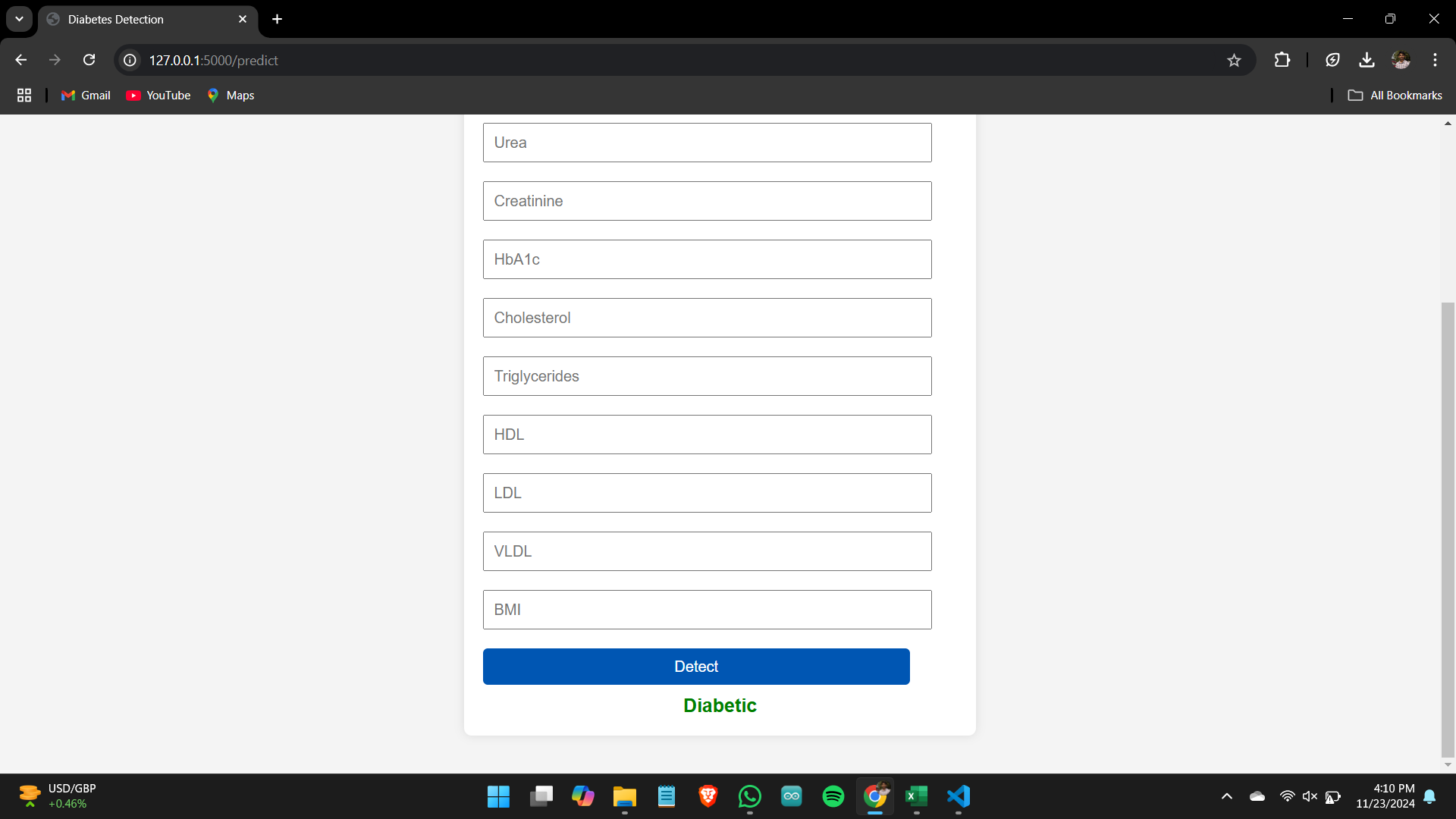
**Running the Flask:**

**Web Page:**

****

****

****

****