SMARTINTERNZ EXTERNSHIP

APPLIED DATA SCIENCE

PROJECT REPORT

TITLE: DIABETES PREDICTION

TEAM ID: 248

TEAM MEMBERS:

1)MOHANA R -20MIS0078(VIT VELLORE)

2)SWETHA V-20MIS0405(VIT VELLORE)

3)MONIGA G-20MIS0417(VIT VELLORE)

4)VAISHNAVI S-20MIS0433(VIT VELLORE)

1.INTRODUCTION

1.1 Overview:

The project aims to develop a diabetes prediction system that utilizes advanced machine learning techniques to forecast the likelihood of an individual developing diabetes in the future. By analyzing various factors and patterns from historical data, this system can provide valuable insights and predictions regarding the risk of diabetes onset.

1.2 Purpose:

The purpose of this project is to offer a proactive approach to diabetes management by leveraging predictive modeling. By accurately predicting the likelihood of diabetes, individuals and healthcare professionals can take timely preventive measures, make informed decisions, and develop personalized interventions to mitigate the risk factors associated with the condition.

2. LITERATURE SURVEY

2.1 Existing Problem:

In the field of diabetes prediction, there have been several existing approaches and methods employed to address the problem. Traditional methods often rely on statistical analysis and risk scoring systems based on clinical parameters such as age, body mass index (BMI) and blood glucose levels. However, these approaches have limitations in terms of accuracy and may not capture the complex interplay of various factors contributing to diabetes. Machine learning techniques have gained popularity in recent years for diabetes prediction. These methods utilize large datasets and algorithms to identify patterns and relationships that can aid in predicting the onset of diabetes. Commonly used algorithms include logistic regression, decision trees, support vector machines (SVM), and artificial neural networks.

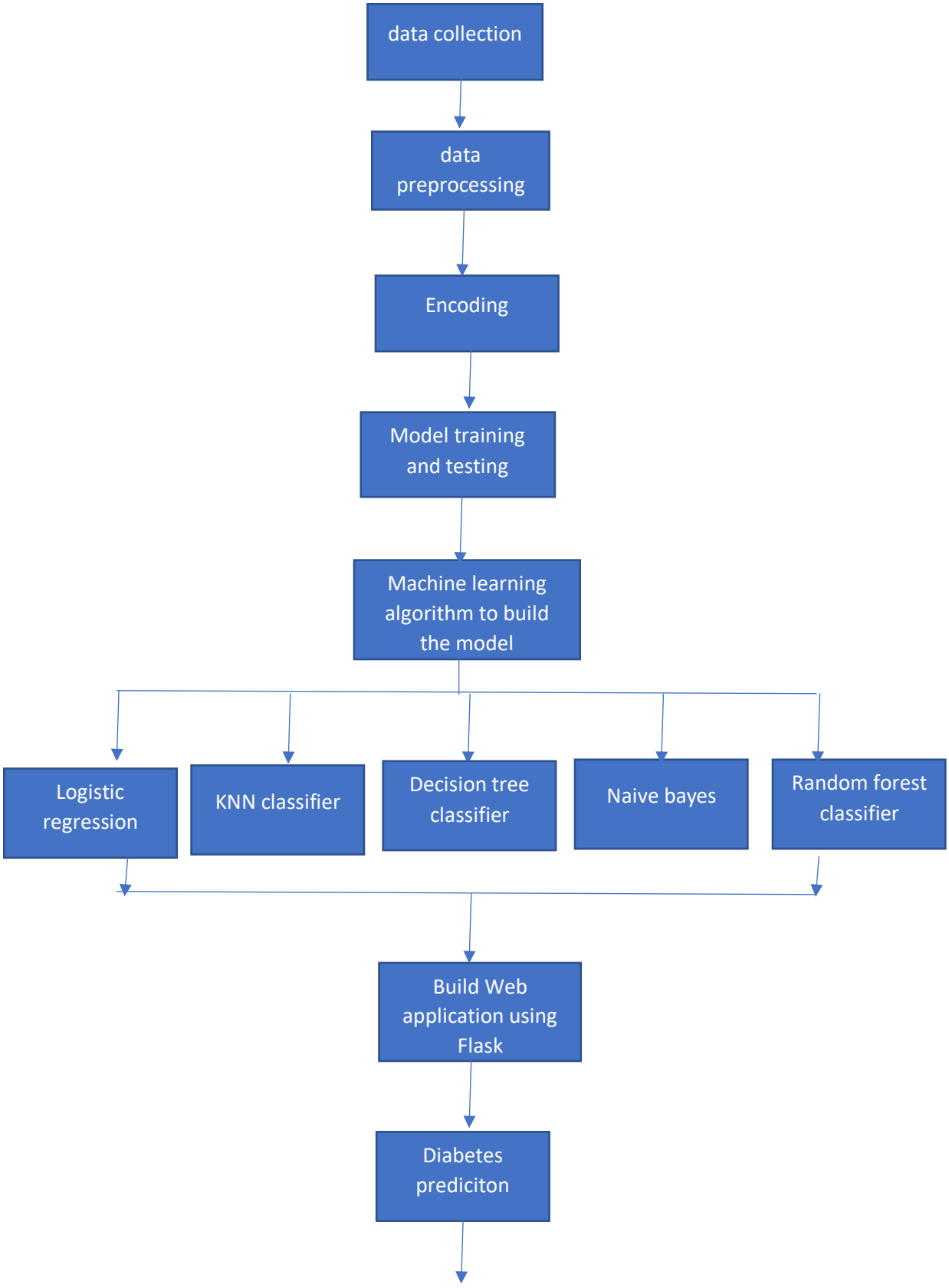
2.2 Proposed Solution:

The proposed solution in this project is to develop a diabetes prediction system that combines and extends the capabilities of existing machine learning algorithms. The system will leverage a diverse range of features, including demographic data, lifestyle factors, medical history, genetic markers, and biomarkers, to enhance the accuracy of predictions. These methods utilize large datasets and algorithms to identify patterns and we used algorithms include logistic regression, decision trees, KNN, naive bayes and random forest classifier.

3 THEORITICAL ANALYSIS

3.1. BLOCK DIAGRAM

Diagrammatic overview of the project



3.2 Hardware/Software Designing:

hardware and software requirements

Hardware Requirements:

Computer or server with sufficient processing power and memory to handle data preprocessing, training, and evaluation of machine learning models. Storage capacity to store the dataset and trained models. Optionally, if deploying the system in a healthcare setting, the hardware infrastructure should support network connectivity and data security measures.

Software Requirements:

Programming languages such as Python for implementing the project.Machine learning and data processing libraries/frameworks like numpy,pandas,matplot lib.Development environment or IDE (Integrated Development Environment) for coding and testing, such as Jupyter Notebook.Web development frameworks or technologies if deploying the system as a web-based application.

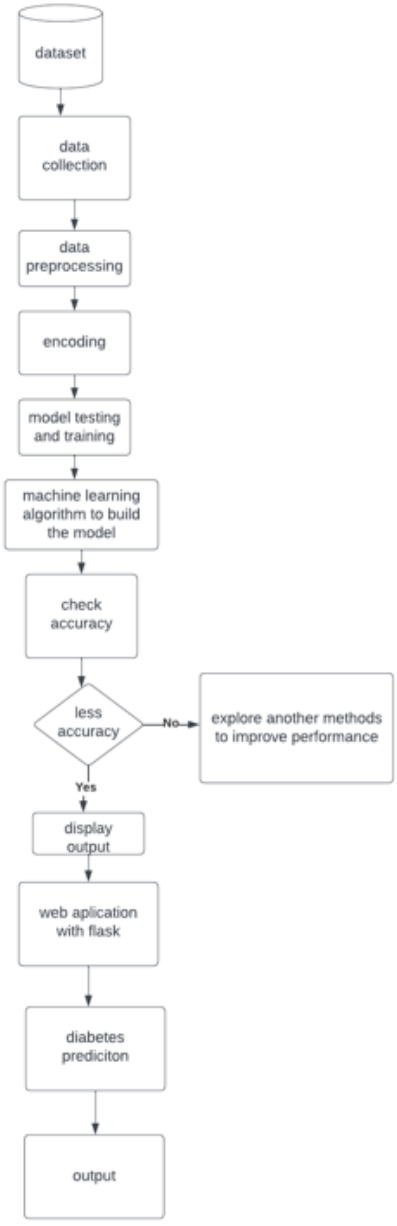
4.EXPERIMENTAL INVESTIGATIONS

Analysis or the investigation made while working on the solution

1. **Data Collection:** The first step is to determine the necessary data required for diabetes prediction.
2. **Data Preprocessing:** Once the data is collected, it's important to preprocess and clean it to ensure accuracy and consistency
3. **Model Selection:** Research and analyze different machine learning algorithms or predictive models that are commonly used for diabetes prediction.
4. **Model Training and Evaluation:** Based on the selected model, split the dataset into training and testing sets. Utilize suitable evaluation metrics like accuracy, precision, recall, F1-score to assess the performance of the trained model.
5. **Integration with Web Application:** Once the model is trained and evaluated, it needs to be integrated into the web application for the diabetes prediction form.
6. **Deployment**: Finally, investigate the deployment options for the web application.

5.FLOW CHART

Diagram showing the control flow of the solution



6.RESULT

Final findings (Output) of the project with screenshots

	ACCURACY	Ranking
Logistic regression	0.812	4
KNN	0.820	3
Naive Bayes	0.648	5
Decision tree	0.838	1
Random forest	0.835	2

```
In [53]: # Train the DecisionTreeClassifier on the reduced feature space
         dt_classifier = DecisionTreeClassifier()
         dt_classifier.fit(x_train_pca, y_train)

Out[53]: DecisionTreeClassifier
         DecisionTreeClassifier()

In [54]: # Make predictions on the testing set
         y_pred = dt_classifier.predict(x_test_pca)

In [55]: y_pred =  dt_classifier.predict(x_test_pca)

In [56]: from sklearn.metrics import accuracy_score,confusion_matrix,classification_report

In [57]: # Evaluate the performance of the model
         accuracy = accuracy_score(y_test, y_pred)
         print("Accuracy:", accuracy)

         Accuracy: 0.8300680731072303

In [58]: from sklearn.metrics import r2_score
         r2=r2_score(y_test,y_pred)
         r2

Out[58]: 0.20953634644183017
```

```
In [58]: from sklearn.metrics import r2_score
         r2=r2_score(y_test,y_pred)
         r2

Out[58]: 0.20953634644183017

In [59]: confusion_matrix(y_test,pred)

Out[59]: array([[90409,  3108,    45],
                [10370,  4396,   280],
                [ 4188,  2810,     5]], dtype=int64)

In [60]: print(classification_report(y_test,y_pred))

                   precision    recall  f1-score   support

                0       0.90      0.95      0.93     93562
                1       0.45      0.48      0.46     15046
                2       0.02      0.00      0.01      7003

         accuracy                           0.83    115611
        macro avg       0.46      0.48      0.47    115611
     weighted avg       0.79      0.83      0.81    115611
```

In [97]:
```
#Initializing the Naive Bayes model
from sklearn.naive_bayes import GaussianNB
nb=GaussianNB()
```

In [98]:
```
#Initializing the Naive Bayes model
from sklearn.naive_bayes import GaussianNB
nb=GaussianNB()
```

In [99]:
```
nb.fit(x_train,y_train)
```

Out[99]:
```
GaussianNB()
```

In [100]:
```
pred1=nb.predict(x_test)
```

In [101]:
```
from sklearn import metrics
```

In [102]:
```
metrics.confusion_matrix(y_test,pred1)
```

Out[102]:
```
array([[65923, 12090, 14537],
       [ 2931,  1489,  4538],
       [ 4208,  2328,  7567]], dtype=int64)
```

In [103]:
```
accuracy_score(y_test,pred1)
```

Out[103]:
```
0.6485455536237901
```

In [45]:
```
knn=KNeighborsClassifier()
knn.fit(x_train,y_train)
```

Out[45]:
```
KNeighborsClassifier()
```

In [46]:
```
pred=knn.predict(x_test)
```

In [47]:
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
```

In [48]:
```
accuracy_score(y_test,pred)
```

Out[48]:
```
0.8200776742697494
```

In [49]:
```
from sklearn.metrics import r2_score
r2=r2_score(y_test,pred)
r2
```

Out[49]:
```
0.06313347269772895
```

In [50]:
```
confusion_matrix(y_test,pred)
```

Out[50]:
```
array([[90409,  3108,    45],
       [10370,  4396,   280],
       [ 4188,  2810,     5]], dtype=int64)
```

In [35]:
```
# Build an model (Logistic Regression)
log_reg = LogisticRegression(random_state=0)
```

In [36]:
```
log_reg.fit(x_train,y_train);
```

```
C:\Users\gokul\anaconda3\Anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:458: ConvergenceWarning: lbfgs failed
to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

In [37]:
```
pred=log_reg.predict(x_test)
```

In [38]:
```
pred1=log_reg.predict(x_train)
```

In [39]:
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
```

In [40]:
```
accuracy_score(y_train,pred1)
```

Out[40]:
```
0.8120819737004288
```

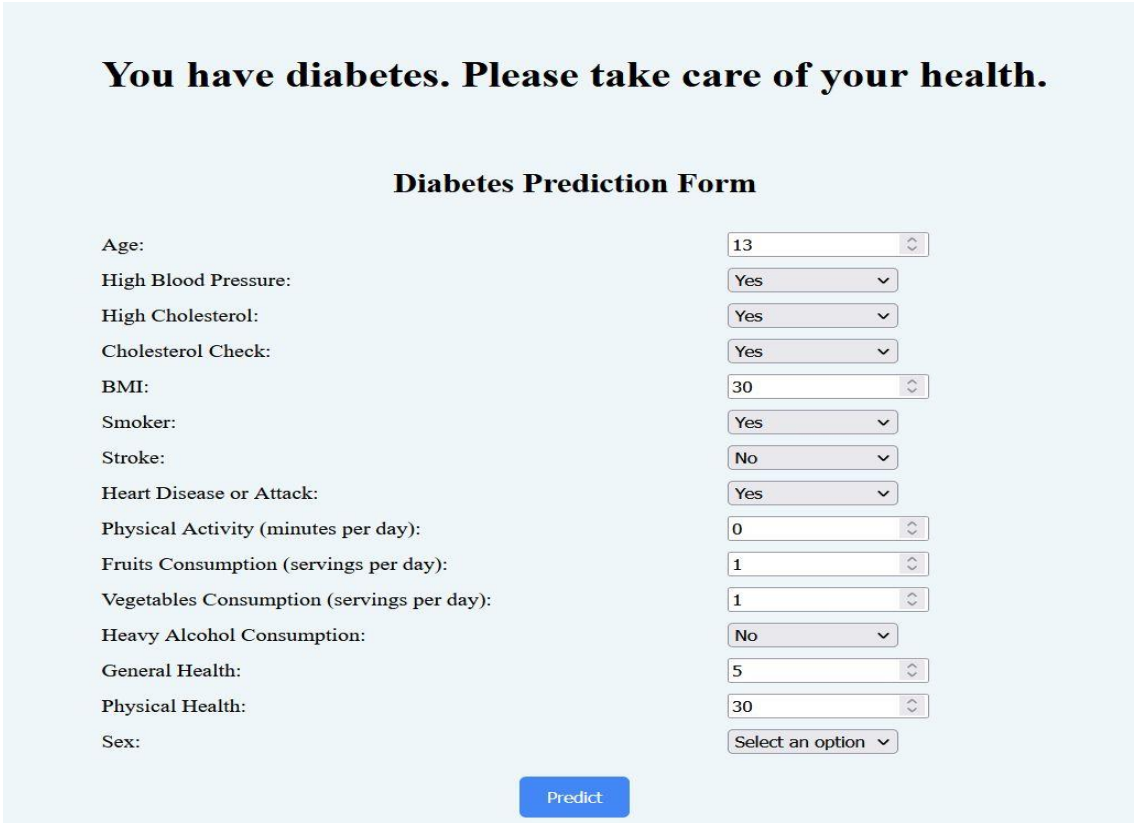
In [41]:
```
accuracy_score(y_test,pred)
```

Out[41]:
```
0.8137547465206598
```

Test the model

```
In [89]: n_components = 15  # Set the desired number of principal components
         pca = PCA(n_components=n_components)
         x_train_pca = pca.fit_transform(x_train)
         x_test_pca = pca.transform(x_test)

In [90]: dt_classifier.fit(x_train_pca, y_train)

Out[90]: DecisionTreeClassifier
         DecisionTreeClassifier()

In [83]: dt_classifier.predict([[1 ,1 ,1 ,13,1 ,1 ,1 ,1 ,1 ,1 ,1 ,2 ,1 ,1 ,9]])
Out[83]: array([1], dtype=int64)

In [85]: dt_classifier.predict([[1 ,1 ,1 ,30,1 ,0 ,1 ,0 ,1 ,1 ,0 ,5 ,30 ,0 ,9]])
Out[85]: array([0], dtype=int64)

In [86]: dt_classifier .predict([[1 ,1 ,1 ,12 ,0 ,0 ,0 ,1 ,1 ,1 ,0 ,2 ,0 ,0 ,11]])
Out[86]: array([0], dtype=int64)

In [87]: dt_classifier .predict([[1 ,1 ,1 ,28 ,1 ,0 ,0 ,0 ,0 ,1 ,0 ,5 ,15 ,0,9]])
Out[87]: array([0], dtype=int64)

In [88]: import pickle
         pickle.dump(dt_classifier,open("Diabetes.pkl","wb"))
```

OUTPUT SCREEN



7.ADVANTAGES AND DISADVANTAGES

List of advantages and disadvantages of the proposed solution

ADVANTAGES

A diabetes prediction solution can help identify individuals who are at risk of developing diabetes at an early stage. With the use of machine learning and predictive algorithms, a diabetes prediction solution can provide personalized recommendations based on an individual's risk factors. Early detection and prevention of diabetes can lead to significant cost savings in healthcare. By implementing preventive measures, the need for expensive treatments and hospitalizations can be reduced.

Disadvantages:

Predictive algorithms may not always be 100% accurate, leading to false positive or false negative results. A diabetes prediction solution typically requires the collection and analysis of personal health data. Not everyone may have access to the technology or resources required to utilize a diabetes prediction solution effectively. Depending solely on a diabetes prediction solution may lead to overreliance on technology and neglect of other important factors in managing diabetes, such as regular medical check-ups and professional guidance.

8.APPLICATIONS

The areas where this solution can be applied

Healthcare and Clinical Settings: Diabetes prediction models can be used in healthcare institutions and clinics to identify individuals who are at high risk of developing diabetes. Public

Health Initiatives: Diabetes prediction can be utilized in public health programs and initiatives aimed at raising awareness, promoting healthy lifestyle choices, and implementing targeted interventions for high-risk populations.

Personalized Medicine and Treatment: Predictive models can assist healthcare professionals in tailoring treatment plans and interventions based on an individual's risk of diabetes

Remote Monitoring and Telemedicine: Diabetes prediction models can be integrated into remote monitoring systems or telemedicine platforms.

9.CONCLUSION

After performing data preprocessing, model evaluation, and training on the diabetes prediction dataset, several machine learning algorithms were employed, including K-Nearest Neighbors (KNN), Decision Tree, Naive Bayes, and Logistic Regression. The dataset was split into independent features and dependent target variables, and unnecessary columns were dropped to focus on relevant attributes. The models were trained using the training data and evaluated using appropriate evaluation metrics such as accuracy, precision, recall, and F1-score. The performances of the models were analyzed, and the best-performing algorithm(s) were selected based on the evaluation results. Additionally, HTML and CSS were utilized to create a interface for executing the models. The implementation was carried out in the Spyder environment, providing a platform for executing and interacting with the developed models. Based on the findings, it can be concluded that the developed models showed promising results in predicting diabetes. The accuracy and performance metrics achieved by the models indicate their

effectiveness in identifying individuals at risk of diabetes. This suggests that the implemented machine learning algorithms can serve as valuable tools for diabetes prediction. Overall, this work demonstrates the application of various machine learning algorithms for diabetes prediction.

10. FUTURE SCOPE

Enhancements that can be made in the future

Improved Accuracy: Continued research and development can focus on enhancing the accuracy of predictive algorithms used in diabetes prediction solutions.

Long-Term Monitoring and Predictive Analytics: Expanding the scope of diabetes prediction beyond short-term risk assessment to long-term monitoring can help identify trends, patterns, and changes in an individual's health over time.

Incorporation of Genetic Information: Genetic factors play a significant role in the development of diabetes. Future enhancements can involve integrating genetic data into diabetes prediction models to provide a more comprehensive risk assessment.

Integration with Electronic Health Records (EHR): Seamless integration of diabetes prediction solutions with electronic health record systems can enhance the overall healthcare ecosystem.

11. BIBILOGRAPHY

References of previous works or websites visited/books referred for analysis about the project, solution previous findings etc.

1) https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset
2) https://towardsdatascience.com/build-deploy-diabetes-prediction-app-using-flask-ml-and-heroku-2de07cbd902d
3) https://github.com/topics/diabetes-prediction

APPENDIX

A. Source Code
code for the solution built.

app6.py

```
from flask import Flask, render_template, request

import pickle


app = Flask(__name__)


# Load the trained model from a pickle file

with open('Diabetes.pkl', 'rb') as f:

    model = pickle.load(f)
```

```
@app.route('/')
def index():
    return render_template('index.html')


@app.route('/predict', methods=['POST'])
def predict():
    # Retrieve the form data
    age = int(request.form['age'])
    high_bp = int(request.form['high-bp'])
    high_chol = int(request.form['high-chol'])
    chol_check = int(request.form['chol-check'])
    bmi = float(request.form['bmi'])
    smoker = int(request.form['smoker'])
    stroke = int(request.form['stroke'])
    heart_disease = int(request.form['heart-disease'])
    phys_activity = int(request.form['phys-activity'])
    fruits = int(request.form['fruits'])
    veggies = int(request.form['veggies'])
    alcohol = int(request.form['alcohol'])
    gen_health = int(request.form['gen-health'])
    phys_health = int(request.form['phys-health'])
    sex = request.form['sex']


    # Perform the prediction
    t = [[age, high_bp, high_chol, chol_check, bmi, smoker, stroke, heart_disease,
phys_activity, fruits, veggies, alcohol, gen_health, phys_health, sex]]
    prediction = model.predict(t)
```

```
    # Process the prediction result
    if prediction[0] == 1:
        result_text = "You have diabetes. Please take care of your health."
    else:
        result_text = "You don't have diabetes. Stay healthy!"

    return render_template('index.html', prediction=result_text)

if __name__ == '__main__' :
    app.run(debug= False)
```

index.html:

```
<!DOCTYPE html>
<html>
<head>
  <title>Diabetes Prediction</title>
  <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='styles.css') }}">

</head>
<body>
  <div class="container">
    <form action="/predict" method="POST">
      <table>
      <tr>
        <td colspan="2" style="text-align: center;"><h1><p style="text-align: center; font-weight: bold;">{{ prediction }}</p><h1></td>
      </tr>
        <tr>
```

```
                <td colspan="2" style="text-align: center;"><h2>Diabetes Prediction
Form</h2></td>
              </tr>
              <tr>
                <td><label for="age">Age:</label></td>
                <td><input type="number" id="age" name="age" required></td>
              </tr>
               <tr>
              <td><label for="high-bp">High Blood Pressure:</label></td>
              <td>
                <select id="high-bp" name="high-bp" required>
                  <option value="">Select an option</option>
                  <option value="0">No</option>
                  <option value="1">Yes</option>
                </select>
              </td>
            </tr>
            <tr>
              <td><label for="high-chol">High Cholesterol:</label></td>
              <td>
                <select id="high-chol" name="high-chol" required>
                  <option value="">Select an option</option>
                  <option value="0">No</option>
                  <option value="1">Yes</option>
                </select>
              </td>
            </tr>
            <tr>
              <td><label for="chol-check">Cholesterol Check:</label></td>
```

```
  <td>
    <select id="chol-check" name="chol-check" required>
      <option value="">Select an option</option>
      <option value="0">No</option>
      <option value="1">Yes</option>
    </select>
  </td>
</tr>
<tr>
  <td><label for="bmi">BMI:</label></td>
  <td><input type="number" id="bmi" name="bmi" required></td>
</tr>
<tr>
  <td><label for="smoker">Smoker:</label></td>
  <td>
    <select id="smoker" name="smoker" required>
      <option value="">Select an option</option>
      <option value="0">No</option>
      <option value="1">Yes</option>
    </select>
  </td>
</tr>
<tr>
  <td><label for="stroke">Stroke:</label></td>
  <td>
    <select id="stroke" name="stroke" required>
      <option value="">Select an option</option>
      <option value="0">No</option>
```

```
            <option value="1">Yes</option>
          </select>
        </td>
      </tr>
      <tr>
        <td><label for="heart-disease">Heart Disease or Attack:</label></td>
        <td>
          <select id="heart-disease" name="heart-disease" required>
            <option value="">Select an option</option>
            <option value="0">No</option>
            <option value="1">Yes</option>
          </select>
        </td>
      </tr>
      <tr>
        <td><label for="phys-activity">Physical Activity (minutes per day):</label></td>
        <td><input type="number" id="phys-activity" name="phys-activity" required></td>
      </tr>
      <tr>
        <td><label for="fruits">Fruits Consumption (servings per day):</label></td>
        <td><input type="number" id="fruits" name="fruits" required></td>
      </tr>
      <tr>
        <td><label for="veggies">Vegetables Consumption (servings per
day):</label></td>
        <td><input type="number" id="veggies" name="veggies" required></td>
      </tr>
      <tr>
        <td><label for="alcohol">Heavy Alcohol Consumption:</label></td>
```

```
  <td>
    <select id="alcohol" name="alcohol" required>
      <option value="">Select an option</option>
      <option value="0">No</option>
      <option value="1">Yes</option>
    </select>
  </td>
</tr>
<tr>
  <td><label for="gen-health">General Health:</label></td>
  <td><input type="number" id="gen-health" name="gen-health" required></td>
</tr>
<tr>
  <td><label for="phys-health">Physical Health:</label></td>
  <td><input type="number" id="phys-health" name="phys-health" required></td>
</tr>
<tr>
  <td><label for="sex">Sex:</label></td>
  <td>
    <select id="sex" name="sex" required>
      <option value="">Select an option</option>
      <option value="0">Male</option>
      <option value="1">Female</option>
    </select>
  </td>
</tr>
<tr>
  <td colspan="2" style="text-align: center;">
```

```
        </td>
        <tr>
          <td colspan="2" style="text-align: center;">
            <button type="submit" class="submit-btn">Predict</button>
          </td>
        </tr>
      </table>
    </form>
  </div>

</body>
</html>
```

Styles.css:

```
/* styles.css */

.container {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
  background-color: #eef6f9;
}

.container h1 {
  margin-bottom: 0;
}

.form-container {
```

```
  background-color: #ffffff;
  padding: 30px;
  border-radius: 10px;
  box-shadow: 0 2px 6px rgba(0, 0, 0, 0.1);
}

.form-title {
  text-align: center;
  margin-bottom: 20px;
  color: #333333;
}

table {
  margin: 0 auto;
  border-collapse: collapse;
}

td {
  padding: 5px;
}

.centered {
  text-align: center;
  padding-top: 20px;
}

.submit-btn {
  background-color: #4287f5;
```

```
  color: #ffffff;
  padding: 10px 20px;
  border-radius: 5px;
  border: none;
  cursor: pointer;
}

.submit-btn:hover {
  background-color: #3467c9;
}
```