Health Care Prediction System

Submitted for the partial fulfilment of the requirement for the Degree in

BACHELOR OF COMPUTER APPLICATIONS

SUBMITTED BY: ASHISH KOTHARI, ANUJ NEGI, ADITYA SINGH RAWAT

University Roll No:

Ashish Kothari:	2221283
Anuj Negi:	2221230
Aditya Singh Rawat:	2221087

Roll No: 17, 10, 4

Course: BCA

Section: D1

Batch-2022-2025

Under the Supervision of:

Mrs Deepti Negi, Assistant Professor

GRAPHIC ERA HILL UNIVERSITY, DEHRADUN

CERTIFICATE

This is to certify that the project titled **"Health Prediction System"** submitted by **Ashish Kothari, Anuj Negi, Aditya Singh Rawat**, to Graphic Era Hill University for the award of the degree of **Bachelor of Computer Application**, is a bona fide record of the research work done by her under our supervision. The contents of this project in full or in parts have not been submitted to any other Institute or University for the award of any degree or diploma.

Mrs. Deepti Negi,

Project Guid

Assistant Professor

GEHU, Dehradun

Place: Dehradun

Date: 05/06/2025

ACKNOWLEDGEMENT

We extend our sincere gratitude to our project guide, Mrs. Deepti Negi, for their invaluable guidance, support, and encouragement throughout this project. Their expertise and insights were instrumental in the successful completion of the "Health Care Prediction System".

We would also like to thank Graphic Era Hill University for providing us with the necessary resources and an conducive environment to pursue this project.

Finally, we express our heartfelt thanks to our families and friends for their constant support and understanding during the course of this project.

Name of student 1 Roll no: 17

Name of student 2 Roll no: 10

Name of student 3 Roll no: 04

ABSTRACT

This project report details the development of the "Health Care Prediction System," a web-based application designed to predict potential diseases based on user-provided symptoms using machine learning models. The system facilitates symptom input via text or speech recognition , and utilizes trained AI models to offer health insights including predicted disease, description, precautions, recommended medications, workouts, and diets. This platform serves as a preliminary diagnostic tool and health information system, enabling users to take initial steps before consulting a medical professional. The application is built using Flask for the backend and integrates various classification models like Support Vector Classifier, Random Forest Classifier, and Gradient Boosting for prediction. The report outlines the system architecture, workflow, dataset details, key modules, and future enhancements. This innovative approach provides quick health insights and promotes health awareness, with potential for expansion into a comprehensive digital healthcare assistant.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

INTRODUCTION

The Health Care Prediction System is a web-based application designed to predict possible diseases based on user symptoms using machine learning models. This system allows users to input symptoms manually or through speech recognition, and then leverages trained AI models to provide health insights, including the predicted disease, its description, precautions, recommended medications, workouts, and diets. This platform serves as an initial diagnostic tool and health information system, helping users take preliminary actions before consulting a medical professional.

PROBLEM DEFINITION

The existing healthcare landscape often presents challenges in timely and accessible preliminary diagnostics, leading to delays in seeking medical attention. Individuals frequently lack immediate tools to understand potential ailments based on early symptoms, which can cause anxiety or the neglect of early intervention. This absence of an initial, user-friendly diagnostic platform creates a gap in proactive health management.

OBJECTIVE

The primary objective of the Health Care Prediction System is to develop an intelligent web application that accurately predicts diseases based on user-reported symptoms. The system aims to provide a seamless user experience through a clean interface and the integration of speech recognition for symptom input. Furthermore, it seeks to offer immediate, personalized health recommendations, including precautions, medications, workouts, and diets, to empower users with preliminary health insights before professional consultation.

FEATURE HIGHLIGHT

The Health Care Prediction System stands out with its ability to predict diseases based on multiple symptoms , enhanced by an intuitive voice recognition input. It boasts an AI-driven backend processing capability, utilizing trained machine learning models for accurate predictions. The system also provides personalized health recommendations as part of its output, making it a comprehensive initial health assessment tool

NEED

There is a significant need for an accessible, intelligent system that can provide immediate, preliminary health insights based on user symptoms. Such a system can bridge the gap between initial symptom awareness and professional medical consultation, enabling users to take proactive measures, alleviate concerns, and potentially reduce the burden on healthcare systems by guiding appropriate next steps.

BENEFITS

The Health Care Prediction System offers several key benefits, including empowering users with quick insights into their health status and promoting health awareness. It acts as an initial diagnostic tool, helping users take preliminary actions before consulting a medical professional. The system's accessibility and personalized recommendations contribute to improved self-management of health and can facilitate more informed discussions with healthcare providers.

CHAPTER 2

REQUIREMENT ANALYSIS

Hardware Requirements

The provided project synopsis does not explicitly detail hardware requirements. However, for a web-based application like the Health Care Prediction System, typical hardware considerations would include:

- **Processor:** A multi-core processor is recommended for efficient execution of machine learning models and serving web requests.
- **RAM:** Sufficient RAM is necessary to handle data loading, model inference, and concurrent user sessions.
- **Storage:** Adequate storage is required for the operating system, application files, datasets (e.g., CSV-based symptom-disease dataset), and any model checkpoints.
- **Network Interface:** A network interface card (NIC) for internet connectivity to host and access the web application.

Software Requirements

The Health Care Prediction System requires the following software components:

- **Operating System:** Compatible operating system (e.g., Windows, macOS, Linux) to run the development environment and host the application.
- **Web Browser:** A modern web browser (e.g., Chrome, Firefox, Edge) for accessing the user interface.
- **Python Environment:** Python 3.x installed for the backend development.
- **Pip (Python Package Installer):** For managing Python dependencies.
- **Integrated Development Environment (IDE):** (Optional but recommended) An IDE like VS Code or PyCharm for development.

Frontend:

- HTML5
- CSS3
- JavaScript
- Bootstrap

Backend:

- Python
- Flask Web Framework

Machine Learning:

- Pickle (for model serialization/deserialization)
- Pandas (for data handling)
- NumPy (for data handling)
- Trained Classification Models:
    - Support Vector Classifier
    - Random Forest Classifier
    - Gradient Boosting

CHAPTER 3

SOFTWARE / PROJECT DESIGN

Data Collection and Preprocessing

A comprehensive dataset containing diseases and their associated symptoms, along with precautions, descriptions, medications, workouts, and diet suggestions, is a foundational element. This raw data undergoes rigorous preprocessing using libraries like Pandas and NumPy to ensure quality, consistency, and suitability for machine learning model training.

Model Training

Several classification machine learning models, including Support Vector Classifier, Random Forest Classifier, and Gradient Boosting, are trained on the preprocessed symptom-disease dataset. The choice of multiple models allows for comparative analysis and the selection of the most accurate predictor. The trained models are then serialized using Pickle for later use in the application.

Backend Development (Flask)

Flask, a Python web framework, is used to build the backend of the application. This involves creating API endpoints to receive symptom input from the frontend, process it, pass it to the trained ML model for prediction, and retrieve relevant health information.

Frontend Development (HTML5, CSS3, JavaScript, Bootstrap)

The user interface is developed using standard web technologies to ensure a responsive and intuitive experience. HTML5 provides the structure, CSS3 handles styling, JavaScript manages interactive elements, and Bootstrap ensures a consistent and mobile-friendly design. Speech recognition integration is handled via the speech_recognition Python library.

Integration and Deployment

The frontend and backend components are integrated to create a cohesive web application. The trained machine learning model is loaded into the Flask application, enabling real-time predictions. The system can then be deployed to a web server, making it accessible to users.

CHAPTER 4

RESULT / TESTING OF PROJECT / SOFTWARE

RESULT

Figure 4.1

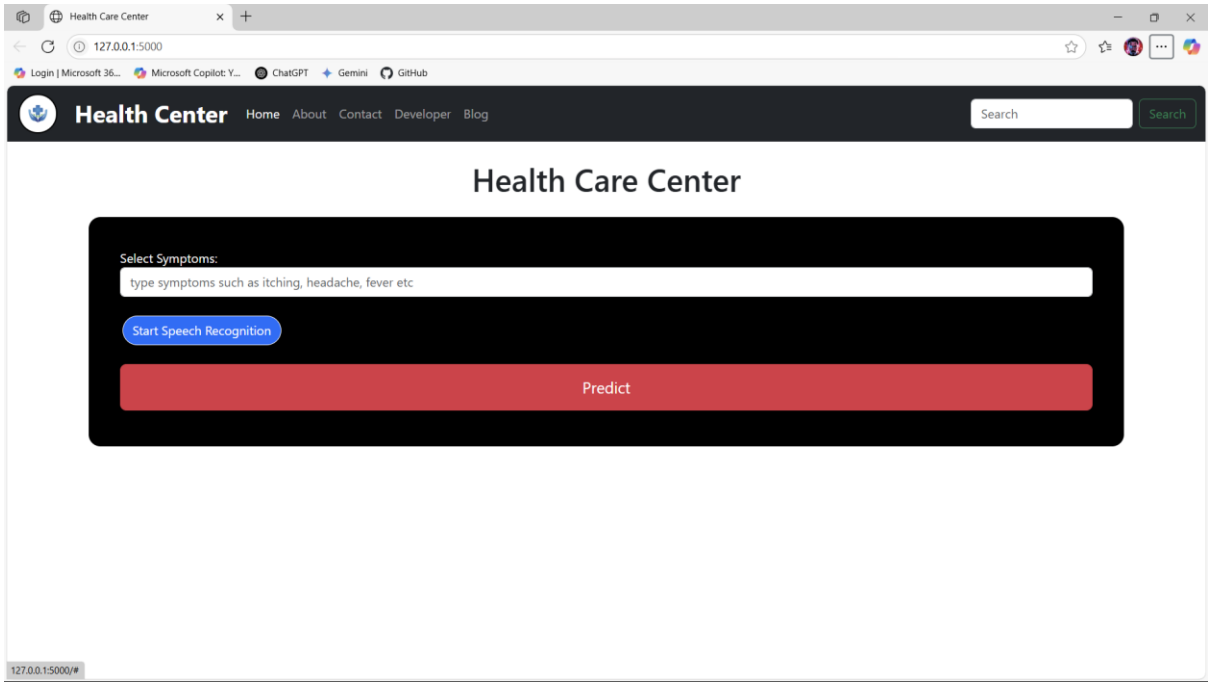


Figure 4.2

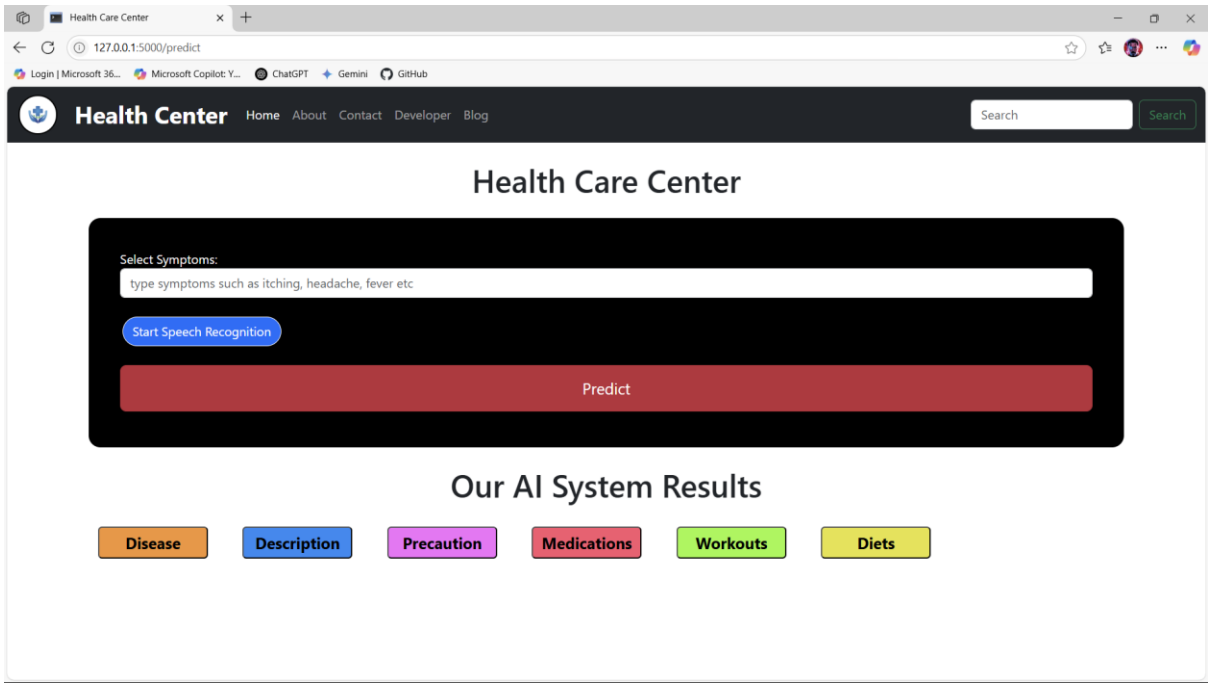
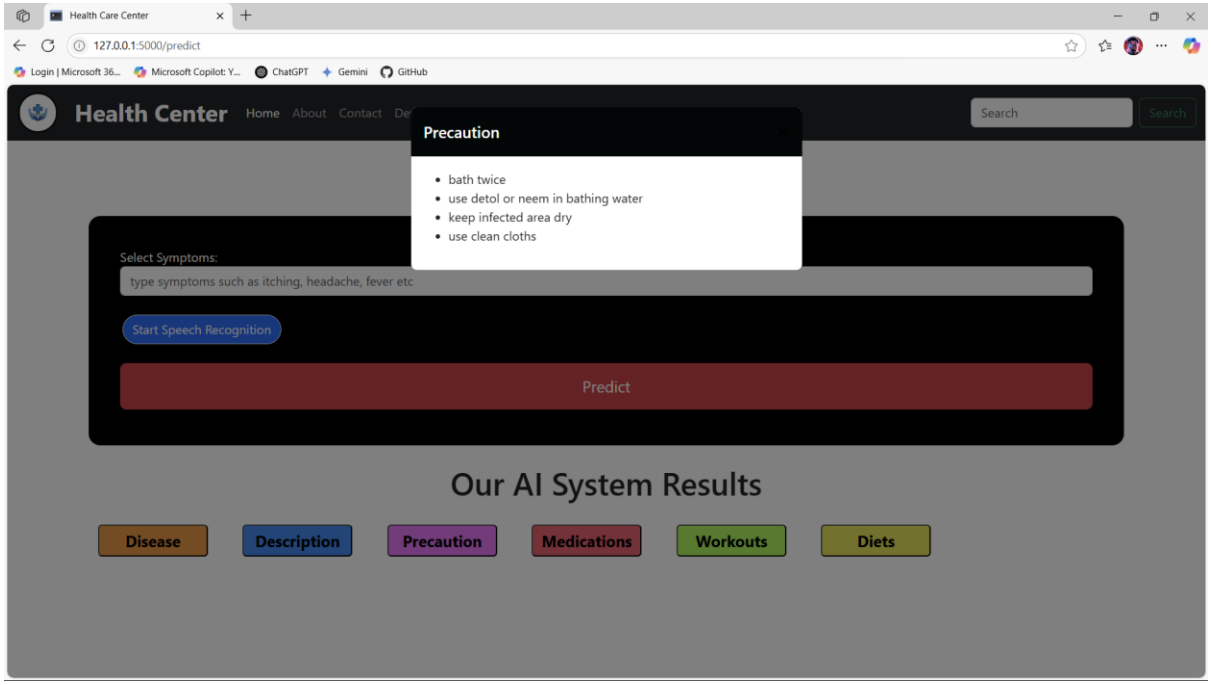


Figure 4.3


Health Care Center
127.0.0.1:5000/predict
Login | Microsoft 36...
Microsoft Copilot: Y...
ChatGPT
Gemini
GitHub
Health Center
Home
About
Contact
Search
Search
Precaution
bath twice
use detol or neem in bathing water
keep infected area dry
use clean cloths
Select Symptoms:
type symptoms such as itching, headache, fever etc
Start Speech Recognition
Predict
Our AI System Results
Disease
Description
Precaution
Medications
Workouts
Diets

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

Conclusion

This project provides an innovative approach to preliminary healthcare diagnostics using AI and machine learning. It empowers users with quick insights into their health status and promotes health awareness. The Health Care Prediction System serves as a valuable initial diagnostic tool, enabling users to take proactive steps before consulting a medical professional. The application's web-based nature and inclusion of features like speech recognition enhance its accessibility and user-friendliness. By leveraging trained machine learning models and comprehensive datasets, the system successfully predicts diseases and offers relevant health recommendations, contributing to better self-management of health. The project is scalable and lays a strong foundation for further development into a complete digital healthcare assistant.

Future Scope

The Health Care Prediction System holds significant potential for future enhancements and expansion. Key areas for future development include:

- **Online Consultation Integration:** Connecting the system with doctors for online consultations would enable users to seamlessly transition from preliminary diagnosis to professional medical advice, creating a more holistic healthcare experience.
- **Real-time Chatbot:** Implementing a real-time chatbot for health queries could provide immediate answers to common health questions, offering interactive support and information to users.
- **Integration with Wearable Devices:** Future iterations could incorporate data from wearable health devices (e.g., smartwatches, fitness trackers) to provide more personalized and real-time health insights based on continuous physiological data.
- **Personalized Health Plans:** Expanding the recommendation engine to generate more detailed and customized health plans, including nutrition, exercise routines, and mental wellness strategies, based on individual user profiles and predicted health risks.
- **Multi-language Support:** To increase accessibility and reach a wider audience, implementing support for multiple languages would be beneficial.

- **Advanced AI Models:** Continuously exploring and integrating more advanced machine learning and deep learning models to improve prediction accuracy and handle more complex symptom patterns.
- **User Feedback and Learning:** Incorporating a feedback mechanism where users can provide input on the accuracy of predictions could be used to continuously retrain and improve the underlying machine learning models.
- **Integration with Electronic Health Records (EHR):** While complex due to privacy concerns, future integration with EHR systems could provide a more comprehensive view of a patient's health history, leading to more accurate predictions and personalized care.

APPENDIX (Code)

Index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <title>Health Care Center</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-4bw+/aepP/YC94hEpVNVgiZdgIC5+VKNBQNGCHeKRQN+PtmoHDEXuppvnDJzQIu9" crossorigin="anonymous" />
    <style>
      .logo {
        width: 50px;
        height: 50px;
        color: black;
        margin-top: 0;
        margin-left: 2px;
      }
      .myimg {
        width: 50px;
        height: 50px;
        border: 2px solid black;
        border-radius: 25px;
      }
    </style>
  </head>
```

```
<body>
    <!-- Navbar -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <div class="container-fluid">
            <!-- Logo in the top-left corner -->
            <div class="logo">
                <img class="myimg" src="{{ url_for('static', filename='logo.png') }}" alt="" />
            </div>
            <a class="navbar-brand" href="#">     <b style="font-size: 30px;">
Health Center </b> </a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-
expanded="false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarSupportedContent">
                <ul class="navbar-nav me-auto mb-2 mb-lg-0">
                    <li class="nav-item">
                        <a class="nav-link active" aria-current="page" href="#">Home</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="/about">About</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="/contact">Contact</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="/developer">Developer</a>
                    </li>
                    <li class="nav-item">
```

```
                    <a class="nav-link" href="/blog">Blog</a>
                </li>
            </ul>
            <form class="d-flex" role="search">
                <input class="form-control me-2" type="search" placeholder="Search" aria-label="Search" />
                <button class="btn btn-outline-success" type="submit">Search</button>
            </form>
        </div>
    </div>
</nav>
<!-- main form of page -->
<h1 class="mt-4 my-4 text-center text-green">Health Care Center</h1>
<div class="container my-4 mt-4" style="background: black; color: white; border-radius: 15px; padding: 40px;">
    <form action="/predict" method="post">
        <div class="form-group">
            <label for="symptoms">Select Symptoms:</label>
            <input type="text" class="form-control" , id="symptoms" name="symptoms" placeholder="type symptoms such as itching, headache, fever etc" />
        </div>
        <br />
        <button type="button" id="startSpeechRecognition" class="btn btn-primary" style="margin-left: 3px; border: 1px solid white; border-radius: 20px;">
            Start Speech Recognition
        </button>
        <br />
        <!-- Display the transcribed text here -->
        <div name="mysysms" id="transcription"></div>
        {% if message %}
```

```
            <p>{{ message }}</p>
            {% endif %}
            <br />
            <button type="submit" class="btn btn-danger btn-lg" style="width: 100%; padding: 14px; margin-bottom: 5px;">
                Predict
            </button>
        </form>
    </div>
    {% if predicted_disease %}
    <!-- Results -->
    <h1 class="text-center my-4 mt-4">Our AI System Results</h1>
    <div class="container">
        <div class="result-container">
            <!-- Buttons to toggle display -->
            <button
                class="toggle-button"
                data-bs-toggle="modal"
                data-bs-target="#diseaseModal"
                style="padding: 4px; margin: 5px 40px 5px 0; font-size: 20px; font-weight: bold; width: 140px; border-radius: 5px; background: #f39334; color: black;"
            >
                Disease
            </button>
            <button
                class="toggle-button"
                data-bs-toggle="modal"
                data-bs-target="#descriptionModal"
                style="padding: 4px; margin: 5px 40px 5px 0; font-size: 20px; font-weight: bold; width: 140px; border-radius: 5px; background: #268af3; color: black;"
```

```
>
    Description
</button>
<button
    class="toggle-button"
    data-bs-toggle="modal"
    data-bs-target="#precautionModal"
    style="padding: 4px; margin: 5px 40px 5px 0; font-size: 20px; font-weight: bold;
width: 140px; border-radius: 5px; background: #f371f9; color: black;"
>
    Precaution
</button>
<button
    class="toggle-button"
    data-bs-toggle="modal"
    data-bs-target="#medicationsModal"
    style="padding: 4px; margin: 5px 40px 5px 0; font-size: 20px; font-weight: bold;
width: 140px; border-radius: 5px; background: #f8576f; color: black;"
>
    Medications
</button>
<button
    class="toggle-button"
    data-bs-toggle="modal"
    data-bs-target="#workoutsModal"
    style="padding: 4px; margin: 5px 40px 5px 0; font-size: 20px; font-weight: bold;
width: 140px; border-radius: 5px; background: #99f741; color: black;"
>
    Workouts
</button>
```

```
            <button
                class="toggle-button"
                data-bs-toggle="modal"
                data-bs-target="#dietsModal"
                style="padding: 4px; margin: 5px 40px 5px 0; font-size: 20px; font-weight: bold; width: 140px; border-radius: 5px; background: #e5e23d; color: black;">
                Diets
            </button>
        </div>
    </div>
    {% endif %}
    <!-- Disease Modal -->
    <div class="modal fade" id="diseaseModal" tabindex="-1" aria-labelledby="diseaseModalLabel" aria-hidden="true">
        <div class="modal-dialog">
            <div class="modal-content">
                <div class="modal-header" style="background-color: #020606; color: white;">
                    <!-- Set header background color inline -->
                    <h5 class="modal-title" id="diseaseModalLabel">Predicted Disease</h5>
                    <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
                </div>
                <div class="modal-body" style="background-color: #modal-body-color;">
                    <!-- Set modal body background color inline -->
                    <p>{{ predicted_disease }}</p>
                </div>
            </div>
        </div>
    </div>
    <!-- Description Modal -->
```

```
    <div class="modal fade" id="descriptionModal" tabindex="-1" aria-labelledby="descriptionModalLabel" aria-hidden="true">
        <div class="modal-dialog">
            <div class="modal-content">
                <div class="modal-header" style="background-color: #020606; color: white;">
                    <h5 class="modal-title" id="descriptionModalLabel">Description</h5>
                    <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
                </div>
                <div class="modal-body">
                    <p>{{ dis_des }}</p>
                </div>
            </div>
        </div>
    </div>
    <!-- Precaution Modal -->
    <div class="modal fade" id="precautionModal" tabindex="-1" aria-labelledby="precautionModalLabel" aria-hidden="true">
        <div class="modal-dialog">
            <div class="modal-content">
                <div class="modal-header" style="background-color: #020606; color: white;">
                    <h5 class="modal-title" id="precautionModalLabel">Precaution</h5>
                    <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
                </div>
                <div class="modal-body">
                    <ul>
                        {% for i in my_precautions %}
                        <li>{{ i }}</li>
                        {% endfor %}
```

```
                </ul>
            </div>
        </div>
    </div>
    <!-- Medications Modal -->
    <div class="modal fade" id="medicationsModal" tabindex="-1" aria-labelledby="medicationsModalLabel" aria-hidden="true">
        <div class="modal-dialog">
            <div class="modal-content">
                <div class="modal-header" style="background-color: #020606; color: white;">
                    <h5 class="modal-title" id="medicationsModalLabel">Medications</h5>
                    <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
                </div>
                <div class="modal-body">
                    <ul>
                        {% for i in medications %}
                        <li>{{ i }}</li>
                        {% endfor %}
                    </ul>
                </div>
            </div>
        </div>
    </div>
    <!-- Workouts Modal -->
    <div class="modal fade" id="workoutsModal" tabindex="-1" aria-labelledby="workoutsModalLabel" aria-hidden="true">
        <div class="modal-dialog">
            <div class="modal-content">
```

```
                <div class="modal-header" style="background-color: #020606; color: white;">
                    <h5 class="modal-title" id="workoutsModalLabel">Workouts</h5>
                    <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
                </div>
                <div class="modal-body">
                    <ul>
                        {% for i in workout %}
                        <li>{{ i }}</li>
                        {% endfor %}
                    </ul>
                </div>
            </div>
        </div>
    </div>
    <!-- Diets Modal -->
    <div class="modal fade" id="dietsModal" tabindex="-1" aria-labelledby="dietsModalLabel" aria-hidden="true">
        <div class="modal-dialog">
            <div class="modal-content">
                <div class="modal-header" style="background-color: #020606; color: white;">
                    <h5 class="modal-title" id="dietsModalLabel">Diets</h5>
                    <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
                </div>
                <div class="modal-body">
                    <ul>
                        {% for i in my_diet %}
                        <li>{{ i }}</li>
                        {% endfor %}
```

```
                </ul>
            </div>
        </div>
    </div>
    <script>
        const startSpeechRecognitionButton =
document.getElementById("startSpeechRecognition");
        const transcriptionDiv = document.getElementById("transcription");
        startSpeechRecognitionButton.addEventListener("click", startSpeechRecognition);
        function startSpeechRecognition() {
            const recognition = new webkitSpeechRecognition(); // Use
webkitSpeechRecognition for compatibility
            recognition.lang = "en-US"; // Set the language for recognition
            recognition.onresult = function (event) {
                const result = event.results[0][0].transcript;
                transcriptionDiv.textContent = result;
            };
            recognition.onend = function () {
                console.log("Speech recognition ended.");
            };
            recognition.start();
        }
    </script>
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
HwwvtgBNo3bZJJLYd8oVXjrBZt8cqVSpeBNS5n7C8IVInixGAoxmnlMuBnhbgrkm"
crossorigin="anonymous"></script>
  </body>
</html>
```

main.py

```
from flask import Flask, request, render_template, jsonify  # Import jsonify
import numpy as np
import pandas as pd
import pickle

# flask app
app = Flask(__name__)


# load database dataset===================================
sym_des = pd.read_csv("datasets/symptoms_df.csv")
precautions = pd.read_csv("datasets/precautions_df.csv")
workout = pd.read_csv("datasets/workout_df.csv")
description = pd.read_csv("datasets/description.csv")
medications = pd.read_csv('datasets/medications.csv')
diets = pd.read_csv("datasets/diets.csv")


# load model=========================================
svc = pickle.load(open('models/svc.pkl','rb'))


#=====================================================================
# custom and helping functions
#==========================helper functions================
def helper(dis):
    desc = description[description['Disease'] == dis]['Description']
    desc = " ".join([w for w in desc])

    pre = precautions[precautions['Disease'] == dis][['Precaution_1', 'Precaution_2',
'Precaution_3', 'Precaution_4']]
    pre = [col for col in pre.values]

    med = medications[medications['Disease'] == dis]['Medication']
    med = [med for med in med.values]

    die = diets[diets['Disease'] == dis]['Diet']
    die = [die for die in die.values]

    wrkout = workout[workout['disease'] == dis] ['workout']

    return desc,pre,med,die,wrkout
```

symptoms_dict = {'itching': 0, 'skin_rash': 1, 'nodal_skin_eruptions': 2, 'continuous_sneezing': 3, 'shivering': 4, 'chills': 5, 'joint_pain': 6, 'stomach_pain': 7, 'acidity': 8, 'ulcers_on_tongue': 9, 'muscle_wasting': 10, 'vomiting': 11, 'burning_micturition': 12, 'spotting_ urination': 13, 'fatigue': 14, 'weight_gain': 15, 'anxiety': 16, 'cold_hands_and_feets': 17, 'mood_swings': 18, 'weight_loss': 19, 'restlessness': 20, 'lethargy': 21, 'patches_in_throat': 22, 'irregular_sugar_level': 23, 'cough': 24, 'high_fever': 25, 'sunken_eyes': 26, 'breathlessness': 27, 'sweating': 28, 'dehydration': 29, 'indigestion': 30, 'headache': 31, 'yellowish_skin': 32, 'dark_urine': 33, 'nausea': 34, 'loss_of_appetite': 35, 'pain_behind_the_eyes': 36, 'back_pain': 37, 'constipation': 38, 'abdominal_pain': 39, 'diarrhoea': 40, 'mild_fever': 41, 'yellow_urine': 42, 'yellowing_of_eyes': 43, 'acute_liver_failure': 44, 'fluid_overload': 45, 'swelling_of_stomach': 46, 'swelled_lymph_nodes': 47, 'malaise': 48, 'blurred_and_distorted_vision': 49, 'phlegm': 50, 'throat_irritation': 51, 'redness_of_eyes': 52, 'sinus_pressure': 53, 'runny_nose': 54, 'congestion': 55, 'chest_pain': 56, 'weakness_in_limbs': 57, 'fast_heart_rate': 58, 'pain_during_bowel_movements': 59, 'pain_in_anal_region': 60, 'bloody_stool': 61, 'irritation_in_anus': 62, 'neck_pain': 63, 'dizziness': 64, 'cramps': 65, 'bruising': 66, 'obesity': 67, 'swollen_legs': 68, 'swollen_blood_vessels': 69, 'puffy_face_and_eyes': 70, 'enlarged_thyroid': 71, 'brittle_nails': 72, 'swollen_extremeties': 73, 'excessive_hunger': 74, 'extra_marital_contacts': 75, 'drying_and_tingling_lips': 76, 'slurred_speech': 77, 'knee_pain': 78, 'hip_joint_pain': 79, 'muscle_weakness': 80, 'stiff_neck': 81, 'swelling_joints': 82, 'movement_stiffness': 83, 'spinning_movements': 84, 'loss_of_balance': 85, 'unsteadiness': 86, 'weakness_of_one_body_side': 87, 'loss_of_smell': 88, 'bladder_discomfort': 89, 'foul_smell_of urine': 90, 'continuous_feel_of_urine': 91, 'passage_of_gases': 92, 'internal_itching': 93, 'toxic_look_(typhos)': 94, 'depression': 95, 'irritability': 96, 'muscle_pain': 97, 'altered_sensorium': 98, 'red_spots_over_body': 99, 'belly_pain': 100, 'abnormal_menstruation': 101, 'dischromic _patches': 102, 'watering_from_eyes': 103, 'increased_appetite': 104, 'polyuria': 105, 'family_history': 106, 'mucoid_sputum': 107, 'rusty_sputum': 108, 'lack_of_concentration': 109, 'visual_disturbances': 110, 'receiving_blood_transfusion': 111, 'receiving_unsterile_injections': 112, 'coma': 113, 'stomach_bleeding': 114, 'distention_of_abdomen': 115, 'history_of_alcohol_consumption': 116, 'fluid_overload.1': 117, 'blood_in_sputum': 118, 'prominent_veins_on_calf': 119, 'palpitations': 120, 'painful_walking': 121, 'pus_filled_pimples': 122, 'blackheads': 123, 'scurring': 124, 'skin_peeling': 125, 'silver_like_dusting': 126, 'small_dents_in_nails': 127, 'inflammatory_nails': 128, 'blister': 129, 'red_sore_around_nose': 130, 'yellow_crust_ooze': 131}

diseases_list = {15: 'Fungal infection', 4: 'Allergy', 16: 'GERD', 9: 'Chronic cholestasis', 14: 'Drug Reaction', 33: 'Peptic ulcer diseae', 1: 'AIDS', 12: 'Diabetes ', 17: 'Gastroenteritis', 6: 'Bronchial Asthma', 23: 'Hypertension ', 30: 'Migraine', 7: 'Cervical spondylosis', 32: 'Paralysis (brain hemorrhage)', 28: 'Jaundice', 29: 'Malaria', 8: 'Chicken pox', 11: 'Dengue', 37: 'Typhoid', 40: 'hepatitis A', 19: 'Hepatitis B', 20: 'Hepatitis C', 21: 'Hepatitis D', 22: 'Hepatitis E', 3: 'Alcoholic hepatitis', 36: 'Tuberculosis', 10: 'Common Cold', 34: 'Pneumonia', 13: 'Dimorphic hemmorhoids(piles)', 18: 'Heart attack', 39: 'Varicose veins', 26:

```
'Hypothyroidism', 24: 'Hyperthyroidism', 25: 'Hypoglycemia', 31: 'Osteoarthristis', 5:
'Arthritis', 0: '(vertigo) Paroymsal  Positional Vertigo', 2: 'Acne', 38: 'Urinary tract infection',
35: 'Psoriasis', 27: 'Impetigo'}

# Model Prediction function
def get_predicted_value(patient_symptoms):
    input_vector = np.zeros(len(symptoms_dict))
    for item in patient_symptoms:
        input_vector[symptoms_dict[item]] = 1
    return diseases_list[svc.predict([input_vector])[0]]

# creating routes========================================
@app.route("/")
def index():
    return render_template("index.html")

# Define a route for the home page
@app.route('/predict', methods=['GET', 'POST'])
def home():
    if request.method == 'POST':
        symptoms = request.form.get('symptoms')
        # mysysms = request.form.get('mysysms')
        # print(mysysms)
        print(symptoms)
        if symptoms =="Symptoms":
            message = "Please either write symptoms or you have written misspelled symptoms"
            return render_template('index.html', message=message)
        else:

            # Split the user's input into a list of symptoms (assuming they are comma-separated)
            user_symptoms = [s.strip() for s in symptoms.split(',')]
            # Remove any extra characters, if any
            user_symptoms = [symptom.strip("[]' ") for symptom in user_symptoms]
            predicted_disease = get_predicted_value(user_symptoms)
            dis_des, precautions, medications, rec_diet, workout = helper(predicted_disease)

            my_precautions = []
            for i in precautions[0]:
                my_precautions.append(i)

            return render_template('index.html', predicted_disease=predicted_disease,
dis_des=dis_des,
                                   my_precautions=my_precautions, medications=medications,
```

```
my_diet=rec_diet,
                            workout=workout)

    return render_template('index.html')

# about view function and path
@app.route('/about')
def about():
    return render_template("about.html")
# contact view funtion and path
@app.route('/contact')
def contact():
    return render_template("contact.html")

# developer view funtion and path
@app.route('/developer')
def developer():
    return render_template("developer.html")

# about view funtion and path
@app.route('/blog')
def blog():
    return render_template("blog.html")


if __name__ == '__main__':

    app.run(debug=True)
```

References (IEEE format)

1. Scikit-learn Developers. Scikit-learn: Machine Learning in Python. URL: https://scikit-learn.org/

2. Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830.

3. Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5-32. DOI: https://doi.org/10.1023/A:1010933404324

4. Friedman, J. H. (2001). Greedy Function Approximation: A Gradient Boosting Machine. Annals of Statistics, 29(5), 1189-1232.

5. Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3), 273-297. DOI: https://doi.org/10.1007/BF00994018

6. Python Software Foundation. Python Language Reference, version 3.x. URL: https://www.python.org/

7. Kaggle Datasets. Disease Symptom Prediction Dataset. URL: https://www.kaggle.com/ [8] World Health Organization (WHO). URL: https://www.who.int/health-topics (Used for disease descriptions or precautionary info)

9 Python Speech Recognition Library Documentation. URL: https://pypi.org/project/SpeechRecognition/

10. Flask Documentation. Flask Web Development Framework. URL: https://flask.palletsprojects.com/

11. Ji, XunSheng, et al. "Application of the digital signal procession in the MEMS gyroscope de-drift." Nano/Micro Engineered and Molecular Systems, 2006. NEMS'06. 1st IEEE International Conference on . IEEE, 2006.

12. Nussbaumer, Henri J., and Martin Vetterli. "Computationally efficient QMF filter banks." Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'84. . Vol. 9. IEEE, 1984.