VEL TECH MULTI TECH
Dr.RANGARAJAN Dr.SAKUNTHALA ENGINEERING COLLEGE
(Approved by AICTE New Delhi, Affiliated to Anna University, Chennai & ISO 9001:2008 Certified Institution & Accredited by NBA New Delhi)

Department of Computer Science and Engineering

191CS87A – Project Work Phase II

Smart Diagnosis and Medicine Recommendation System

Team Members:

AJAY ABISHEK T.K - 113121UG03007
ARULJOTHI.S - 113121UG03012
MUTHURAJ.C - 113121UG03071
SIVAPRAKASH M - 113121UG3101

Batch No:11
Date: 21/03/2025

Guided by,

Mr. N.Insozhan, M.E.,

Assistant Professor,

Dept of CSE.

OUTLINE OF PRESENTATION

- PROBLEM STATEMENT
- OBJECTIVE
- ABSTRACT
- EXISTING SYSTEM
- PROPOSED SYSTEM
- ARCHITECTURE DIAGRAM
- SYSTEM REQUIREMENTS
- IMPLEMENTATION MODELS
- CONCLUSION
- FUTURE ENCHANCEMENT
- REFERENCE

PROBLEM STATEMENT

- Healthcare professionals struggle with accurate and efficient diagnoses due to complex symptoms and vast medical data.
- Traditional methods lead to delays, misdiagnoses, and inconsistent treatment plans, impacting patient outcomes.
- Patients lack personalized, real-time health information for effective condition management.

OBJECTIVE:

- Develop a software solution to accurately predict symptoms, diagnose diseases, and recommend personalized corrective actions (precautions, medications, workouts, and diets) for optimized healthcare outcomes.

- Create an automated system leveraging machine learning and comprehensive datasets for efficient, accurate disease diagnosis and enhanced decision-making.

ABSTRACT

- This Smart Diagnosis System predicts symptoms and recommends corrective actions such as precautions, diets, and exercises.

- It aims to revolutionize healthcare with proactive, tailored support for effective diagnosis and care.

EXISTING SYSTEM

- **Manual Diagnosis Process:** Current systems rely on healthcare professionals to manually assess symptoms and patient histories, leading to delays in diagnosis and treatment.

- **Lack of Personalization:** Traditional methods don’t offer personalized recommendations, limiting tailored treatment plans for individual patients.

- **Data Overload:** Healthcare professionals often struggle to manage large volumes of patient data, making it difficult to find relevant information for decision-making quickly.

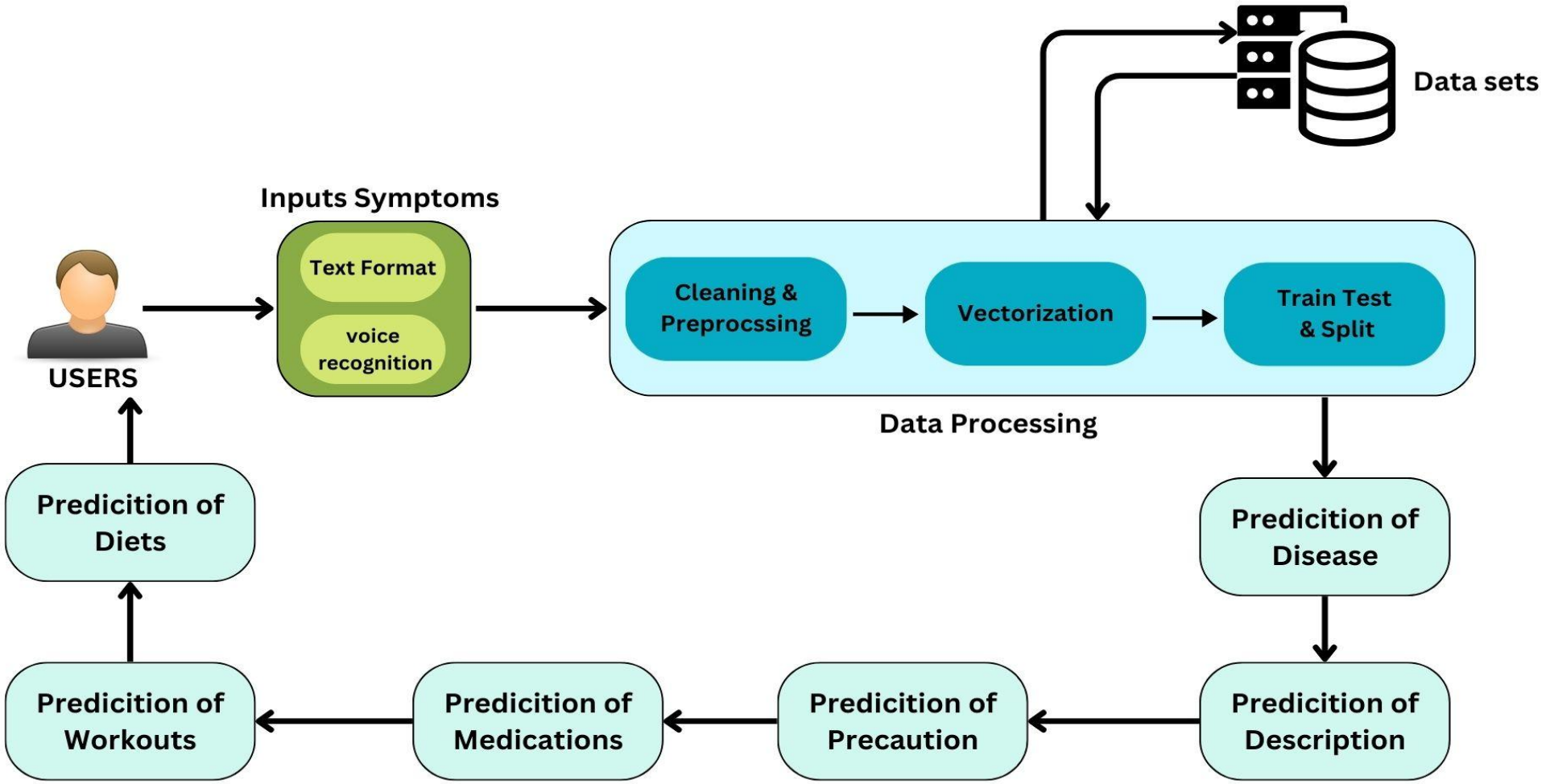
PROPOSED SYSTEM

Voice Assistance:

A voice-assisted medical recommendation system combines speech recognition, Natural Language Processing (NLP), and medical recommendation algorithms to assist patients and healthcare providers.

It enables users to interact with the system using voice commands to receive medical guidance, symptom analysis.

ARCHITECTURE DIAGRAM



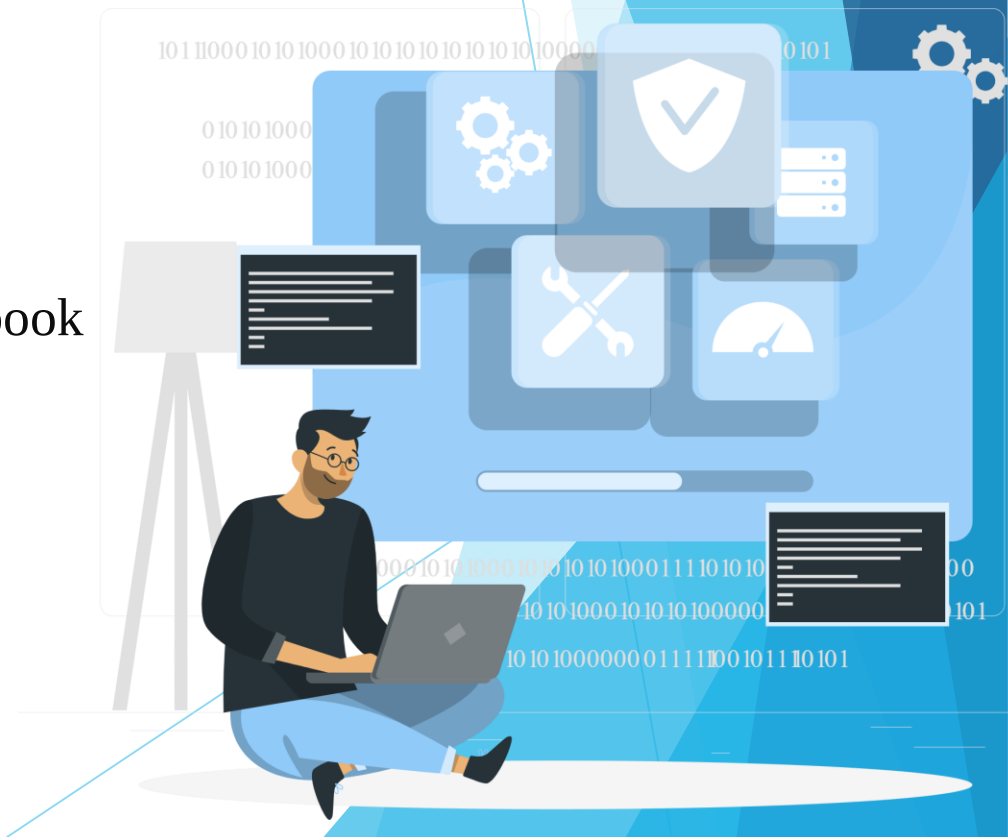
SYSTEM REQUIREMENTS

HARDWARE REQUIREMENTS:

- Intel Core i5 (11th Gen) processor
- 8GB RAM for efficient computing and SSD for faster data access and reliable operations
- NVIDIA RTX 3060 for high-performance AI processing

SOFTWARE REQUIREMENTS:

- **Development Tools:** PyCharm/Visual Studio, Jupyter Notebook for development and testing.
- **Libraries and Frameworks:** TensorFlow, NumPy, Pandas, Flask, Scikit-learn..



Project Phase 1:

1. **Requirement Gathering and Analysis -** Identified key features for the Smart diagnosis system. Collected and analyzed datasets for model training (e.g., patient symptoms, and historical diagnoses).
2. **Literature Review -** Reviewed existing machine learning solutions in healthcare. Explored techniques such as CNNs, Random Forest, and sentiment analysis for system design.
3. **System Design and Architecture -** Drafted initial system architecture for Smart diagnosis and medicine recommendation. Outlined integration of machine learning models with a user-friendly interface.
4. **Dataset Preparation -** Cleaned and preprocessed collected data for training and validation. Split datasets into training, testing, and validation subsets.
5. **Prototype Development -** Implemented a basic prototype using TensorFlow and Scikit-learn. Conducted initial testing on small datasets for model accuracy and performance.

Project Phase 2:

1. Core Model Development

Train Advanced Models

Develop and train Machine Learning models like CNNs for image-related medical diagnosis or Random Forest for tabular data.

Fine-tune hyperparameters for accuracy and efficiency.

Sentiment Analysis Integration: Implement sentiment analysis to interpret patient feedback or self-reported symptoms.

2. System Integration

Backend Development: Develop APIs using Flask or FastAPI to integrate Machine Learning models with the application. Implement endpoints for real-time symptom analysis and medicine recommendations.

Frontend Interface Design: Design a user-friendly interface for patients and healthcare providers to interact with the system.

3. **Data Model Testing**

Real-World Testing: Gather feedback by testing your system with sample users or simulated scenarios.

4. Symptom-to-Specialist Mapping

Create a Mapping Database: Build a dataset that links common symptoms to medical specialties (e.g., fever → general physician, chest pain → cardiologist).

Use publicly available healthcare datasets or medical references to ensure accuracy

AI-Based Symptom Analysis: Enhance your existing symptom analysis model to suggest the most relevant specialist based on user input.

5. Documentation and Reporting: Document the workflow for model training, testing, and deployment.- Include test cases, user guides, and system architecture details.

CODE STRUCTURE:

```
my_flask_app/
│
├── app.py                  # The main application file
├── datasets/               # Directory for CSV files
│   ├── symptoms_df.csv     # Symptoms dataset
│   ├── precautions_df.csv  # Precautions dataset
│   ├── workout_df.csv      # Workout dataset
│   ├── description.csv     # Disease descriptions
│   ├── medications.csv     # Medications dataset
│   └── diets.csv           # Diet recommendations dataset
├── models/                 # Directory for the machine learning model
│   └── svc.pkl             # Pre-trained model
├── templates/              # HTML files for rendering pages
│   ├── index.html          # Main page
│   ├── about.html          # About page
│   ├── contact.html        # Contact page
│   ├── developer.html      # Developer page
│   └── blog.html           # Blog page
└── static/                 # Static files like CSS, JS, and images
    └── (your static files)
```

Main.py

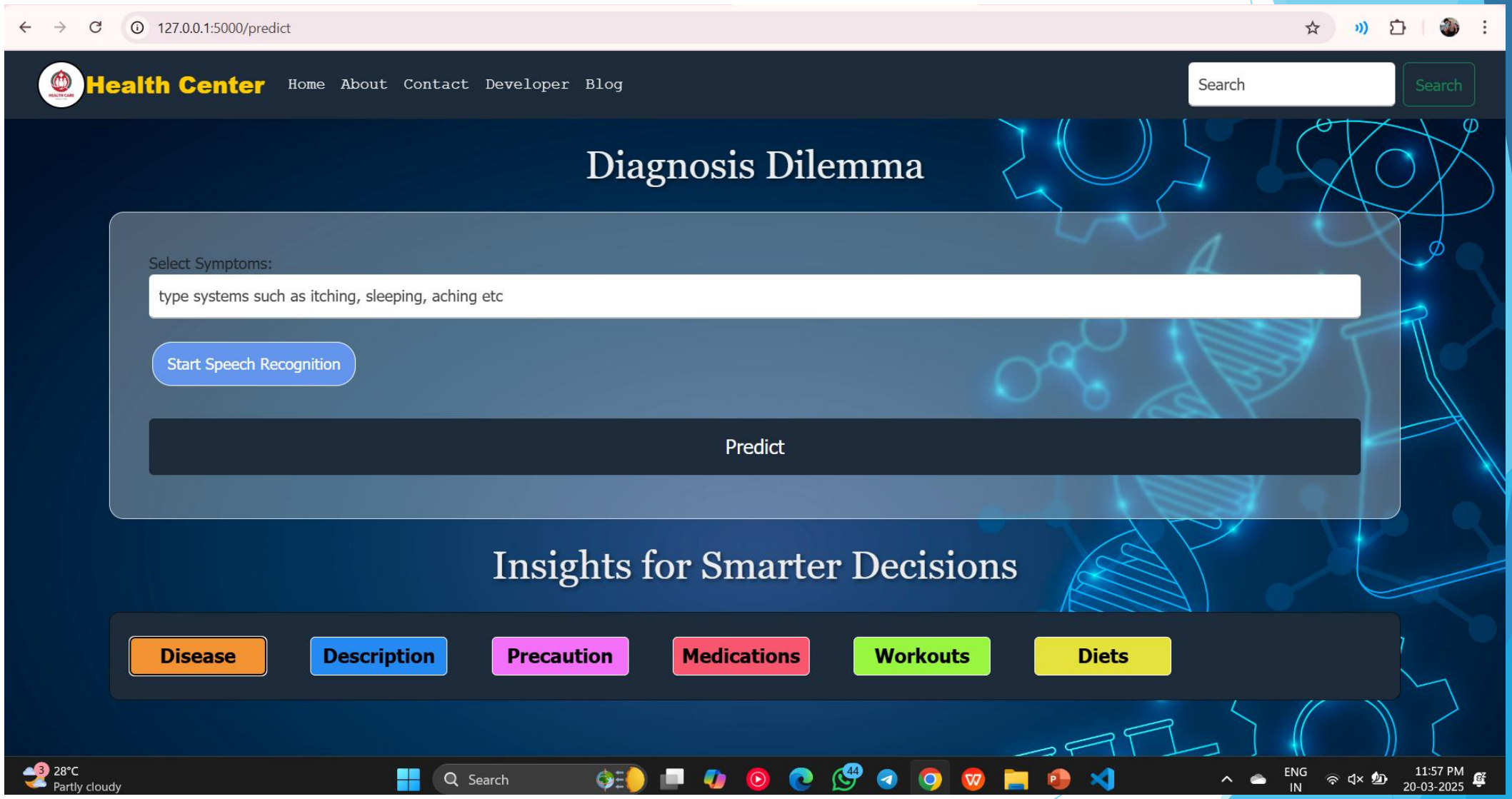
C:\Users\hp\Documents\Project1\main.py

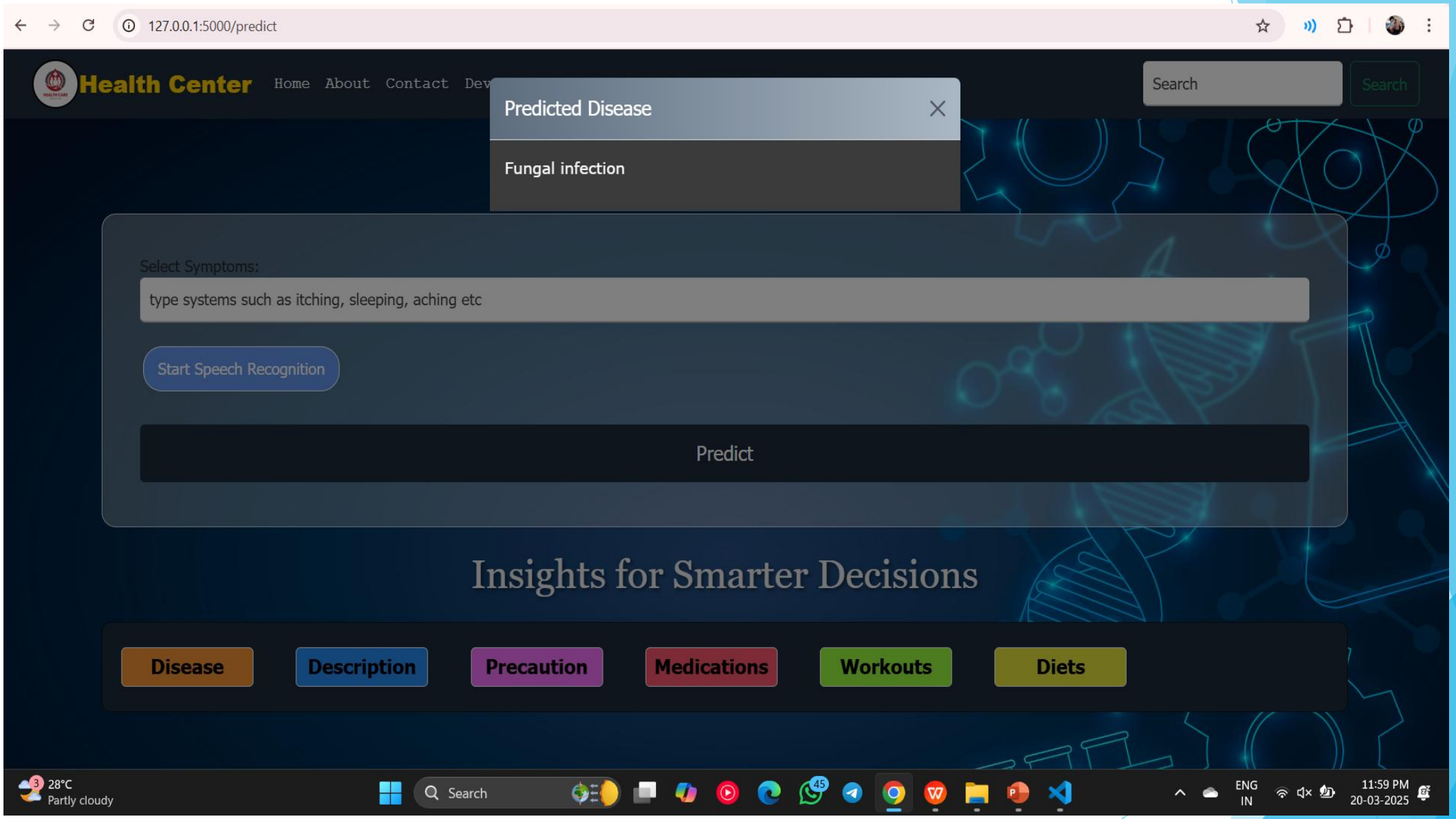
```python
import numpy as np
import pandas as pd
import pickle
import difflib  # For symptom matching suggestions

# Flask app
app = Flask(__name__, template_folder='templates', static_folder='static')

# Load datasets
sym_des = pd.read_csv("datasets/symtoms_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'))

# Symptoms and diseases dictionaries
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 disease', 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',
```

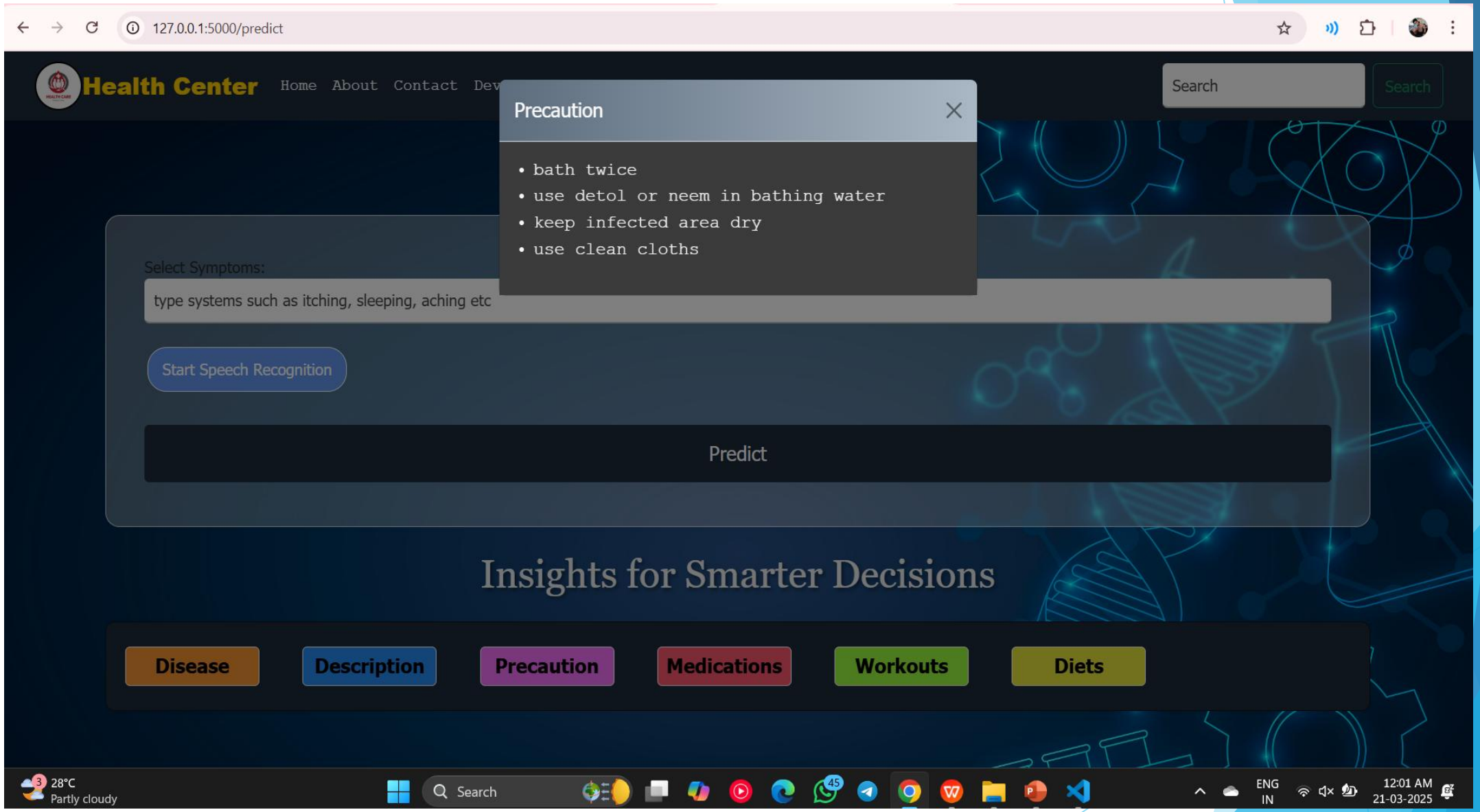
```python
diseases_list = {
    15: 'Fungal infection', 4: 'Allergy', 16: 'GERD', 9: 'Chronic cholestasis', 14: 'Drug Reaction',
    33: 'Peptic ulcer disease', 1: 'AIDS', 12: 'Diabetes', 17: 'Gastroenteritis', 6: 'Bronchial Asthma',
    23: 'Hypertension', 30: 'Migraine', 7: 'Cervical spondylosis', 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 hemorrhoids (piles)', 18: 'Heart attack',
    39: 'Varicose veins', 26: 'Hypothyroidism', 24: 'Hyperthyroidism', 25: 'Hypoglycemia', 31: 'Osteoarthritis',
    5: 'Arthritis', 0: '(Vertigo) Paroxysmal Positional Vertigo', 2: 'Acne', 38: 'Urinary tract infection',
    35: 'Psoriasis', 27: 'Impetigo'
}


# Helper functions
def helper(disease):
    """Fetch disease details like description, precautions, medications, diet, and workouts."""
    desc = description.loc[description['Disease'] == disease, 'Description'].values[0]
    pre = precautions.loc[precautions['Disease'] == disease, ['Precaution_1', 'Precaution_2', 'Precaution_3', 'Precaution_4']].values[0]
    med = medications.loc[medications['Disease'] == disease, 'Medication'].tolist()
    die = diets.loc[diets['Disease'] == disease, 'Diet'].tolist()
    wrkout = workout.loc[workout['disease'] == disease, 'workout'].tolist()
    return desc, pre.tolist(), med, die, wrkout

def get_predicted_value(patient_symptoms):
    """Predict the disease based on symptoms."""
    input_vector = np.zeros(len(symptoms_dict))
    for symptom in patient_symptoms:
        input_vector[symptoms_dict[symptom]] = 1
    prediction = svc.predict([input_vector])[0]
    return diseases_list[prediction]

def suggest_symptom(input_symptom):
    """Suggest closest matching symptom for user input."""
    matches = difflib.get_close_matches(input_symptom, symptoms_dict.keys(), n=1, cutoff=0.7)
    return matches[0] if matches else None

# Routes
@app.route("/")
def index():
    return render_template("index.html")

@app.route('/predict', methods=['GET', 'POST'])
def predict():
    if request.method == 'POST':
        symptoms = request.form.get('symptoms')
        if not symptoms or symptoms.lower() == "symptoms":
            message = "Please enter valid symptoms."
            return render_template('index.html', message=message)

        user_symptoms = [sym.strip().lower().replace(" ", "_") for sym in symptoms.split(',')]
        corrected_symptoms = []

        for symptom in user_symptoms:
            if symptom in symptoms_dict:
                corrected_symptoms.append(symptom)
            else:
                suggestion = suggest_symptom(symptom)
                if suggestion:
```

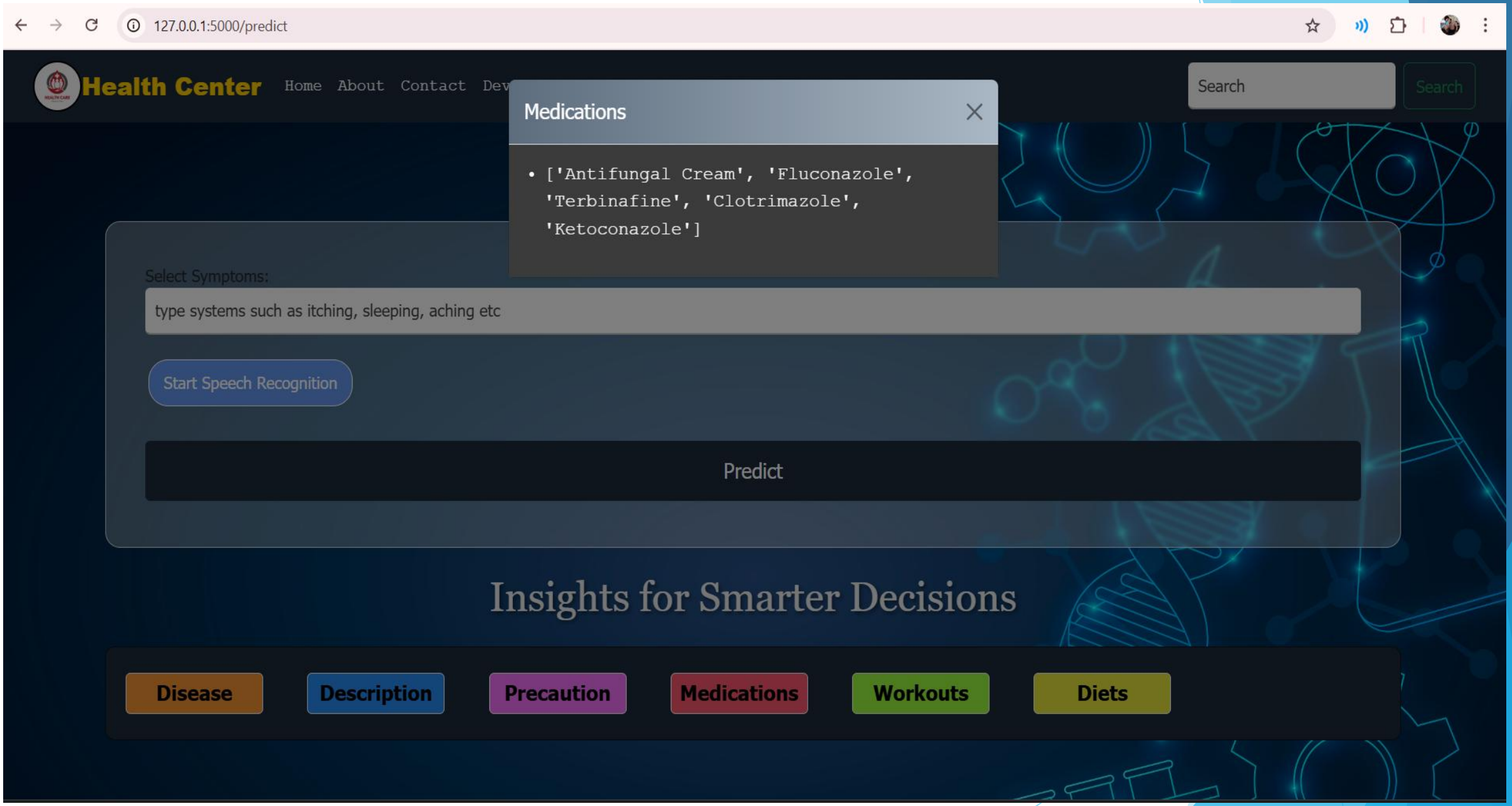
```python
@app.route('/predict', methods=['GET', 'POST'])
def predict():
    if request.method == 'POST':
        symptoms = request.form.get('symptoms')
        if not symptoms or symptoms.lower() == "symptoms":
            message = "Please enter valid symptoms."
            return render_template('index.html', message=message)

        user_symptoms = [sym.strip().lower().replace(" ", "_") for sym in symptoms.split(',')]
        corrected_symptoms = []

        for symptom in user_symptoms:
            if symptom in symptoms_dict:
                corrected_symptoms.append(symptom)
            else:
                suggestion = suggest_symptom(symptom)
                if suggestion:
                    corrected_symptoms.append(suggestion)
                else:
                    message = f"Symptom '{symptom}' not recognized. Please check your input."
                    return render_template('index.html', message=message)

        predicted_disease = get_predicted_value(corrected_symptoms)
        dis_des, precautions, medications, rec_diet, workout = helper(predicted_disease)

        return render_template(
            'index.html',
            predicted_disease=predicted_disease,
            dis_des=dis_des,
            my_precautions=precautions,
            medications=medications,
            my_diet=rec_diet,
            workout=workout
        )

    return render_template('index.html')

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

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

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

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

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

Output:


127.0.0.1:5000/predict
Health Center
Home
About
Contact
Developer
Blog
Search
Search
Diagnosis Dilemma
Select Symptoms:
type systems such as itching, sleeping, aching etc
Start Speech Recognition
Predict
Insights for Smarter Decisions
Disease
Description
Precaution
Medications
Workouts
Diets
28°C
Partly cloudy
Search
ENG
IN
11:57 PM
20-03-2025


127.0.0.1:5000/predict
Health Center
Home
About
Contact
Dev
Search
Search
Predicted Disease
Fungal infection
Select Symptoms:
type systems such as itching, sleeping, aching etc
Start Speech Recognition
Predict
Insights for Smarter Decisions
Disease
Description
Precaution
Medications
Workouts
Diets
28°C
Partly cloudy
Search
ENG
IN
11:59 PM
20-03-2025


127.0.0.1:5000/predict
Health Center
Home
About
Contact
Dev
Search
Search
Precaution
• bath twice
• use detol or neem in bathing water
• keep infected area dry
• use clean cloths
Select Symptoms:
type systems such as itching, sleeping, aching etc
Start Speech Recognition
Predict
Insights for Smarter Decisions
Disease
Description
Precaution
Medications
Workouts
Diets
28°C
Partly cloudy
Search
ENG
IN
12:01 AM
21-03-2025


127.0.0.1:5000/predict
Health Center
Home
About
Contact
Dev
Search
Search
Medications
['Antifungal Cream', 'Fluconazole', 'Terbinafine', 'Clotrimazole', 'Ketoconazole']
Select Symptoms:
type systems such as itching, sleeping, aching etc
Start Speech Recognition
Predict
Insights for Smarter Decisions
Disease
Description
Precaution
Medications
Workouts
Diets

CONCLUSION

A Smart Diagnosis and Medicine Recommendation System enhances healthcare by providing accurate diagnoses and treatment Suggestions. It leverages AI, machine learning, and data analytics to minimize errors and improve efficiency. Continuous refinement, expert validation, and regulatory compliance are essential for reliability.

FUTURE ENCHANCEMENT

To improve a voice assistant in your native language, focus on enhancing language understanding, speech recognition accuracy, and natural language processing (NLP), ensuring the assistant can accurately understand and respond in your language

REFERENCE

1. John, D., & Smith, A. "AI-Based Medication Identification for Accessible Healthcare Solutions." International Conference on Health Informatics, 2021, pp. 112–118.

2. Huang et al. "Deep Learning for Early Cancer Detection." IEEE Transactions on Biomedical Engineering, 2023.

3. Gupta, S., & Sharma, S. "AI-Based Prediction of Drug Interactions." Journal of Healthcare Systems, 2023.

4. Sangamithra, B. "Personalized Ranking Mechanism Using Yandex Dataset on Machine Learning Approaches." Springer, 2022.

5. Ting, H. W., et al. "A Drug Identification Model Developed Using Deep Learning Technologies." BMC Health Serv Res, 2020.

6. Zhou, L., et al. "Real-Time Medication Detection Using Mobile-Based AI Frameworks." IEEE Transactions, 2021.

7. Robinson, M., & Taylor, J. "Natural Language Processing in Clinical Settings." Advances in AI Research, 2020.

8. Patel, A., et al. "Integrating Wearable Data for Enhanced Diagnosis." IEEE IoT Journal, 2022.

9. Kumar, R., & Singh, V. "Machine Learning Models for Chronic Disease Detection." Computational Biology Journal, 2021.

10. Lee, J., & Park, H. "Leveraging AI for Preventive Healthcare." International Conference on Smart Health, 2023.

THANK YOU!