Medicine Recommendation System Using Machine Learning and Flask

Semester Project Report

Session 2023-2027

BS in Data Science



Department of Software Engineering

Faculty of Computer Science & Information Technology

The Superior College, Lahore

FALL 2023

**Submitted by:**

**Maryam Sharif**

**Su92-bsdsm-f23-038**

**BSDS-3A**

**Submitted to:**

**Prof, Rasikh Ali**

**Abstract:**

This project develops a Medicine Recommendation System to provide personalized medical advice based on user symptoms. The system uses a Support Vector Classifier (SVC) trained on symptom-disease data and is deployed as a web application using Flask. The interface enables users to input symptoms and receive recommendations on diseases, medicines, precautions, diets, and workouts.

**1. Introduction**

Healthcare systems increasingly require innovative solutions to deliver efficient and accurate recommendations. This Medicine Recommendation System bridges the gap between patients and healthcare providers by suggesting medicines and additional medical advice based on user symptoms. With machine learning at its core, the system offers reliable disease predictions and recommendations.

**Objective:**

To create a Flask-based web application that predicts diseases from symptoms and provides appropriate medical recommendations.

**Applications:**

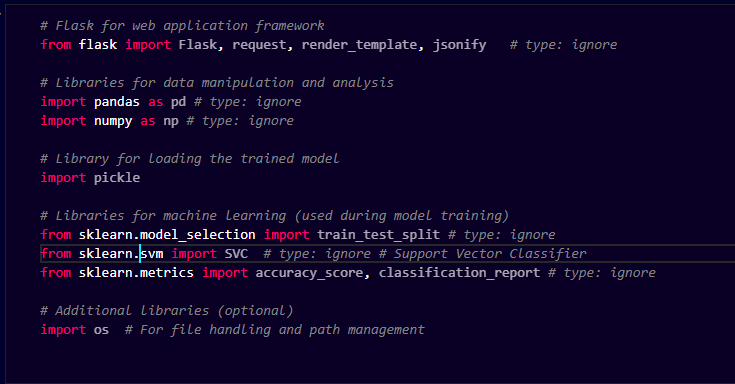
* Healthcare applications
* Telemedicine platforms
* Clinics for assisting practitioners

**2. Methodology**

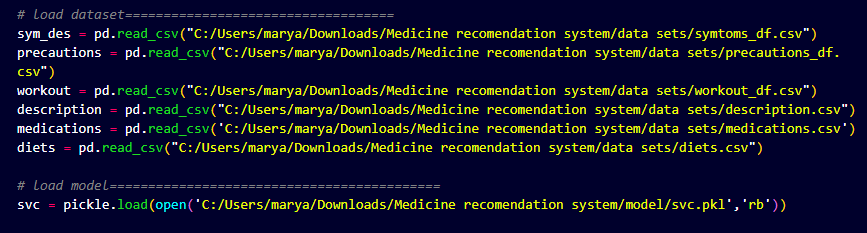
**2.1 Data Sources:**

* Datasets include symptoms\_df, precautions\_df, workout\_df, description.csv, medications.csv, and diets.csv.
* Each dataset provides specific information: symptoms, disease descriptions, medications, and associated recommendations.

**Important Libraries**

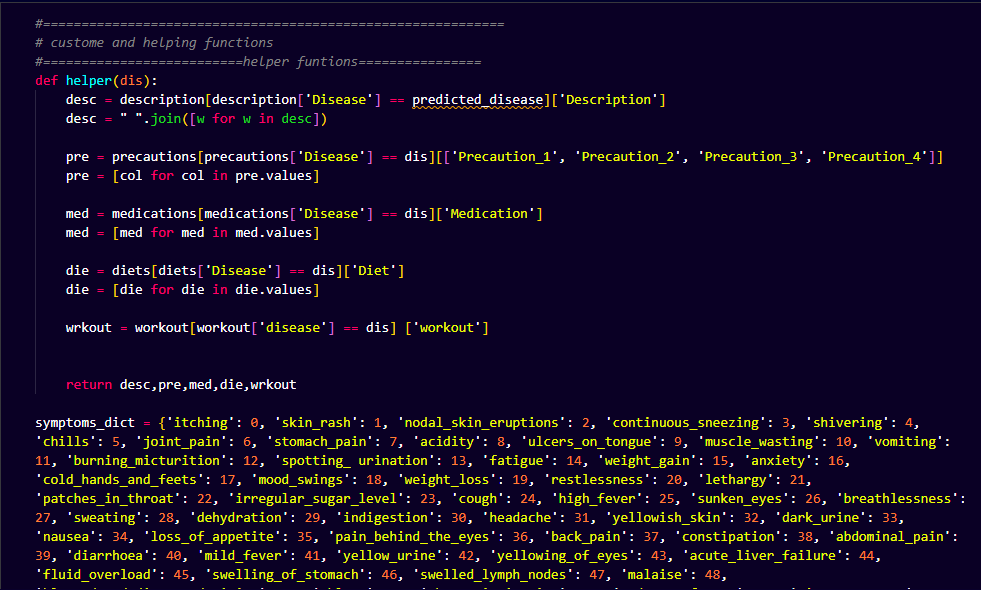
****

**Load Data Sets**

****

**2.2 Data Preprocessing:**

* Created a dictionary of symptoms mapped to numerical indices.
* Designed a helper function for extracting disease-specific details like precautions, medications, diets, and workouts.



**2.3 Machine Learning Model:**

* **Algorithm Used:** Support Vector Classifier (SVC)
* **Input:** One-hot encoded symptom vector.
* **Output:** Predicted disease mapped from the training dataset.
* **Training Method:** The model was trained using labeled datasets containing symptoms and associated diseases.

**2.4 Flask Web Application Integration:**

* **Frontend:** HTML templates for user interaction.
* **Backend:** Flask routes for predictions, user queries, and additional pages like "About," "Contact," and "Developer."
* **API Integration:** A POST method fetches user symptoms and predicts diseases via the trained SVC model.

**3. Implementation**

**3.1 Technology Stack:**

* **Languages:** Python (backend), HTML/CSS (frontend)
* **Framework:** Flask
* **Libraries:** Pandas, NumPy, Scikit-learn, Pickle
* **Tools:** Jupyter Notebook for training, Flask for deployment

**3.2 Flask Route Structure:**

1. /: Home page
2. /predict: Accepts symptom input and returns predictions and recommendations.
3. /about, /contact, /developer, /blog: Static content pages.

**Error Handling: Invalid Symptoms Input**

When a user enters a symptom that is not present in the symptoms CSV file, the application identifies it as invalid and provides meaningful feedback to guide the user. Here's how the error handling works:

**Symptom Validation:**Each symptom entered by the user is compared against a predefined dictionary or list (symptoms\_dict) derived from the symptoms CSV file.

If a symptom exists in the CSV file, it is marked as valid.

If a symptom is not found, it is flagged as invalid.

**User Feedback:**  
If any invalid symptoms are detected:

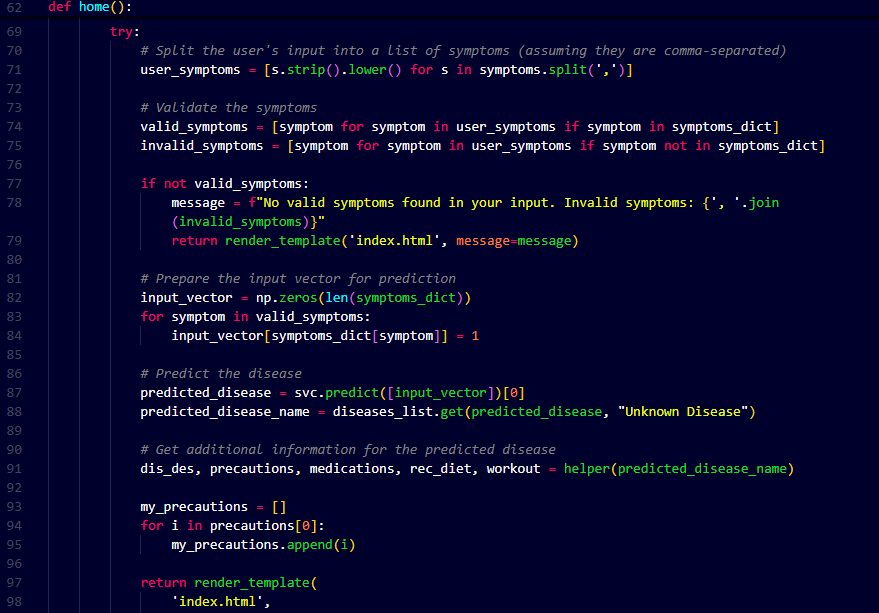
The application returns a clear message to the user, specifying which symptoms are not recognized.

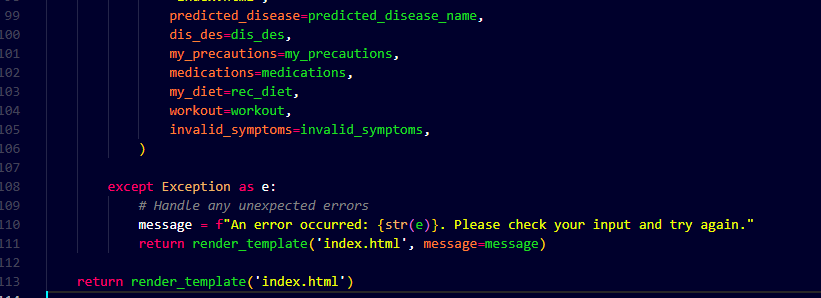
This helps the user correct their input by ensuring it matches the available symptoms listed in the system.

**Graceful Handling:**  
Even if all entered symptoms are invalid:

The application does not crash.

Instead, it displays a helpful message, encouraging the user to provide valid symptoms.

**Prevention of Empty Input:**The system also checks for empty or default placeholder inputs, prompting the user to enter symptoms if no input is provided.

****

**4. System Architecture**

1. **Frontend:**
   * A responsive HTML form accepts symptoms as input.
   * Results are displayed with detailed descriptions, medicines, diets, and workout recommendations.
2. **Backend:**
   * Routes handle form submissions and integrate predictions from the SVC model.
3. **Model:**
   * A pre-trained SVC model stored in Pickle format (svc.pkl) predicts diseases based on symptoms.
4. **Database:**
   * CSV files for symptoms, medications, precautions, diets, and disease descriptions.

**5. Results**

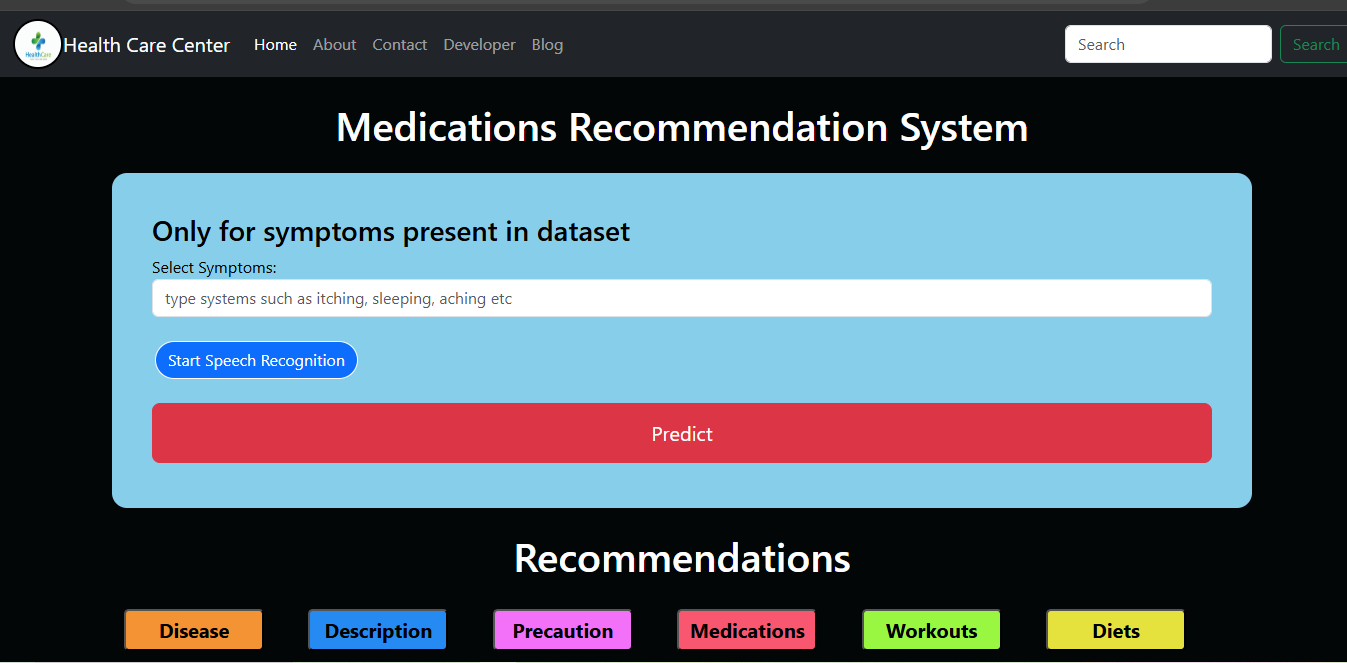
**5.1 Model Performance:**

* The SVC model achieved high prediction accuracy for the given dataset.
* Features like multi-symptom input, detailed recommendations, and precautions enhance usability.

**5.2 User Interface:**

* Intuitive design with options to input symptoms, read disease details, and contact developers.
* Outputs include the predicted disease, its description, medicines, diets, precautions, and workout tips.

**Home Page**

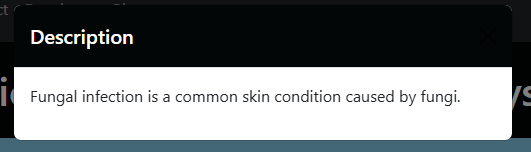
****

**After giving input**

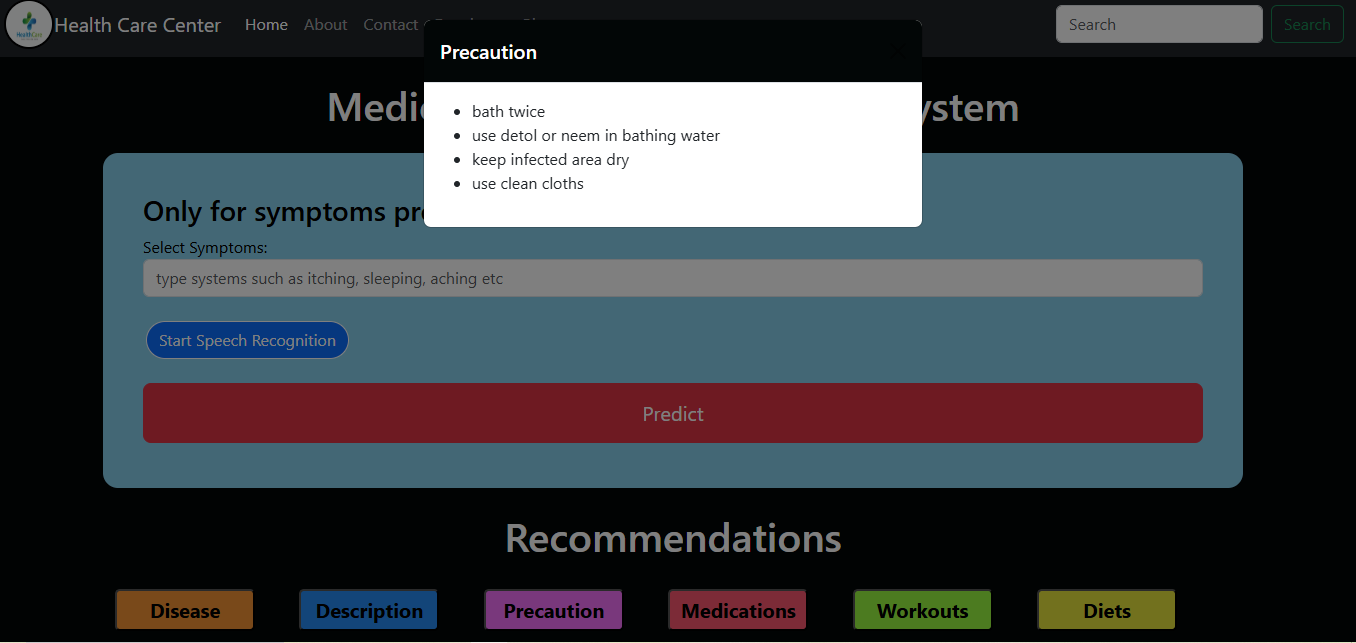
**Predicted disease**



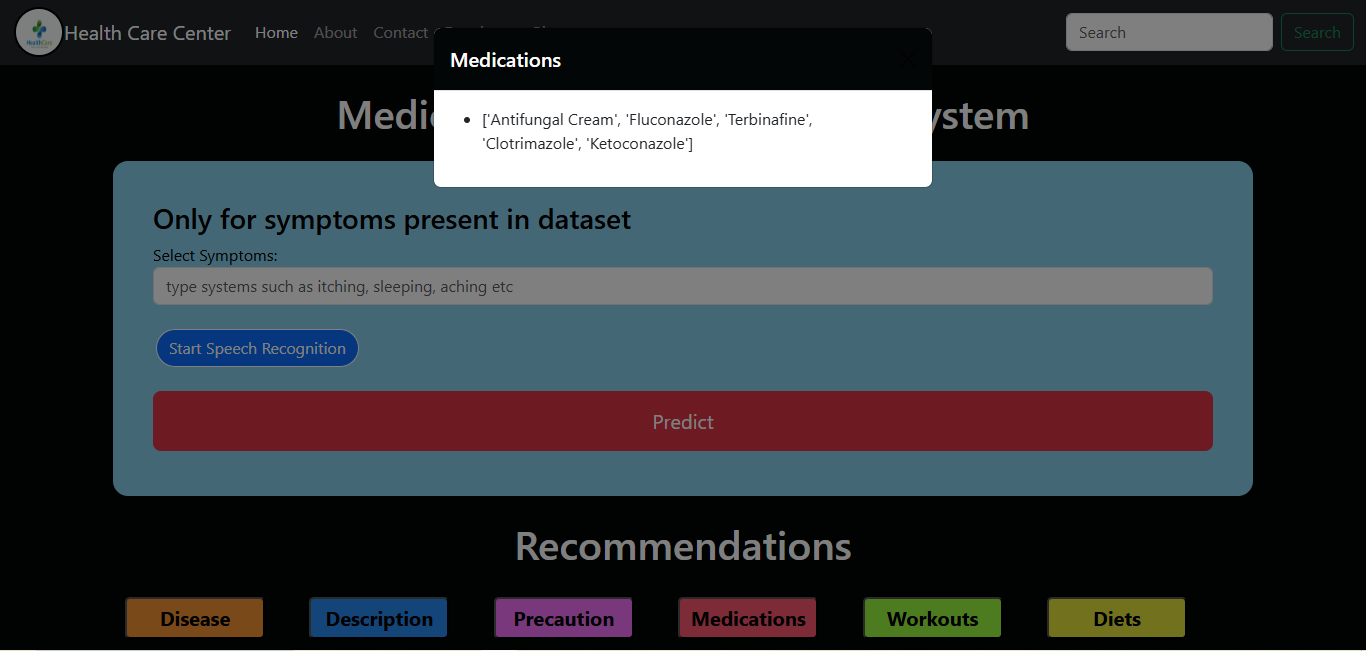
**Description**

****

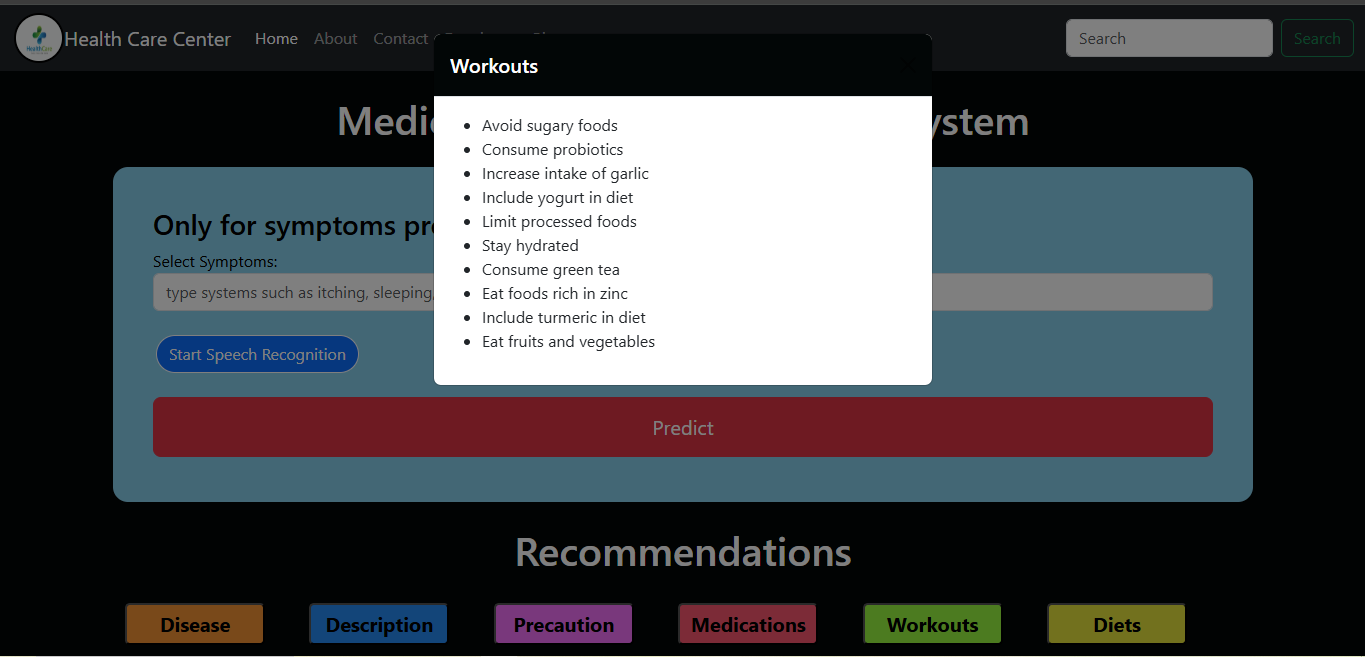
**Precautions**

****

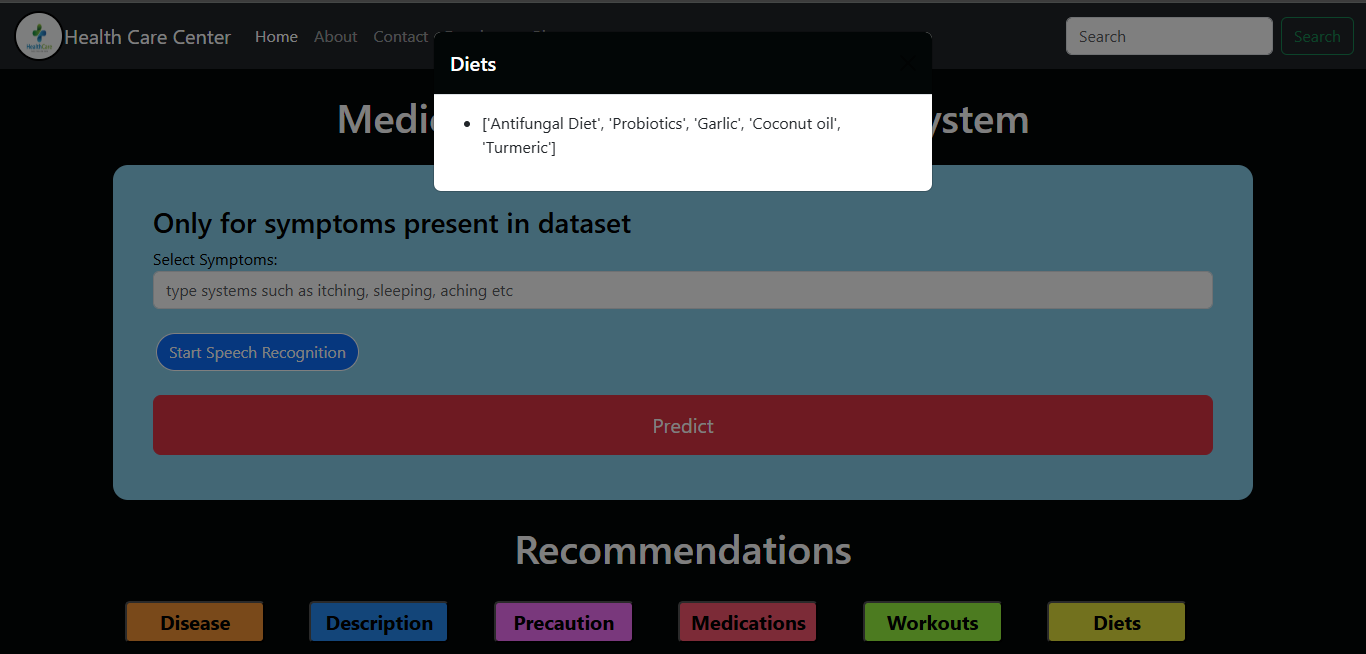
**Medication**



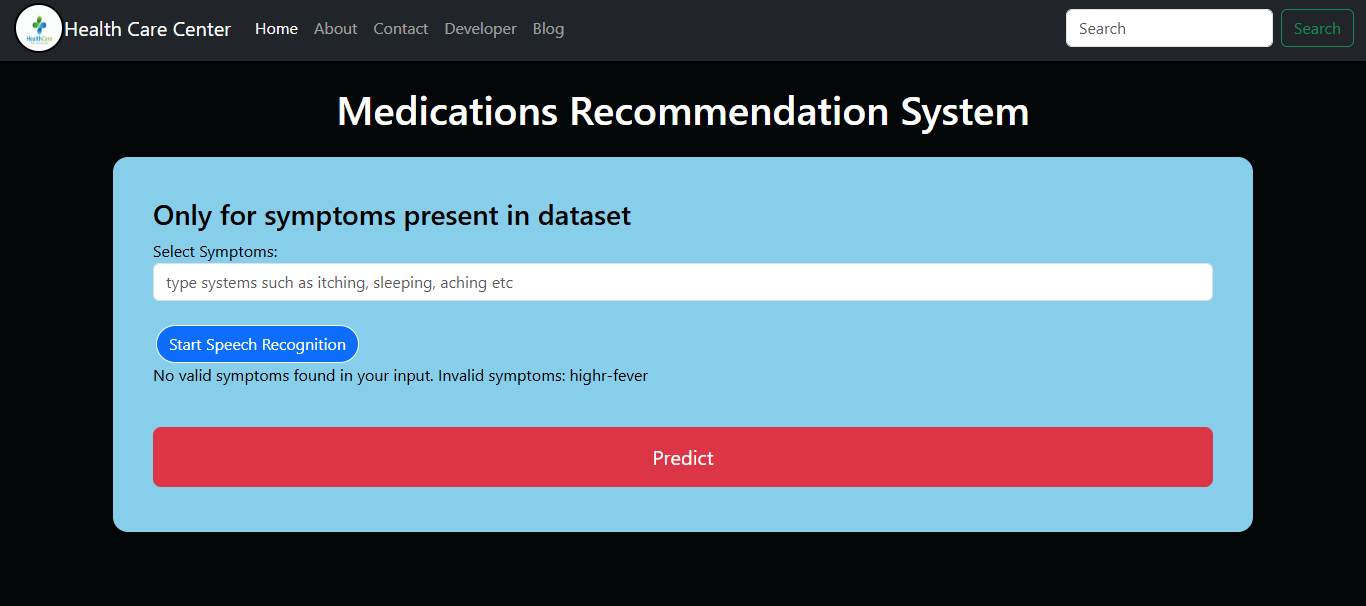
**Workouts**

****

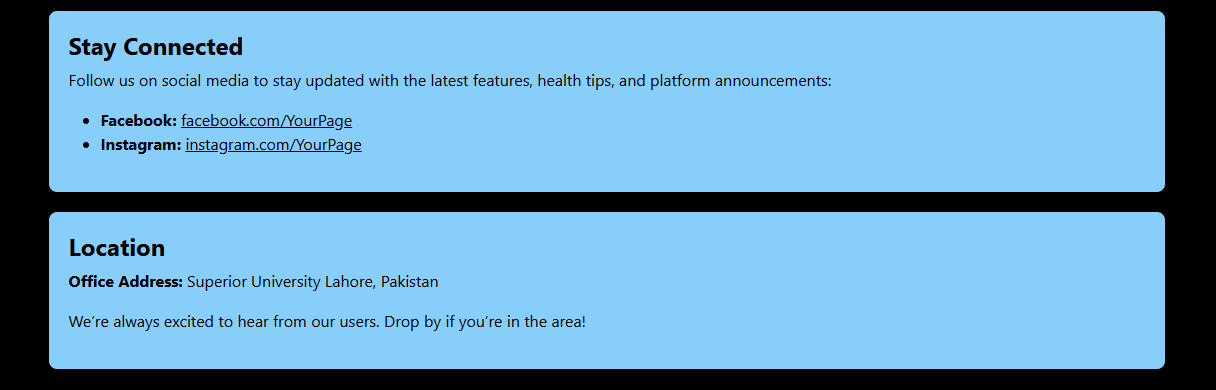
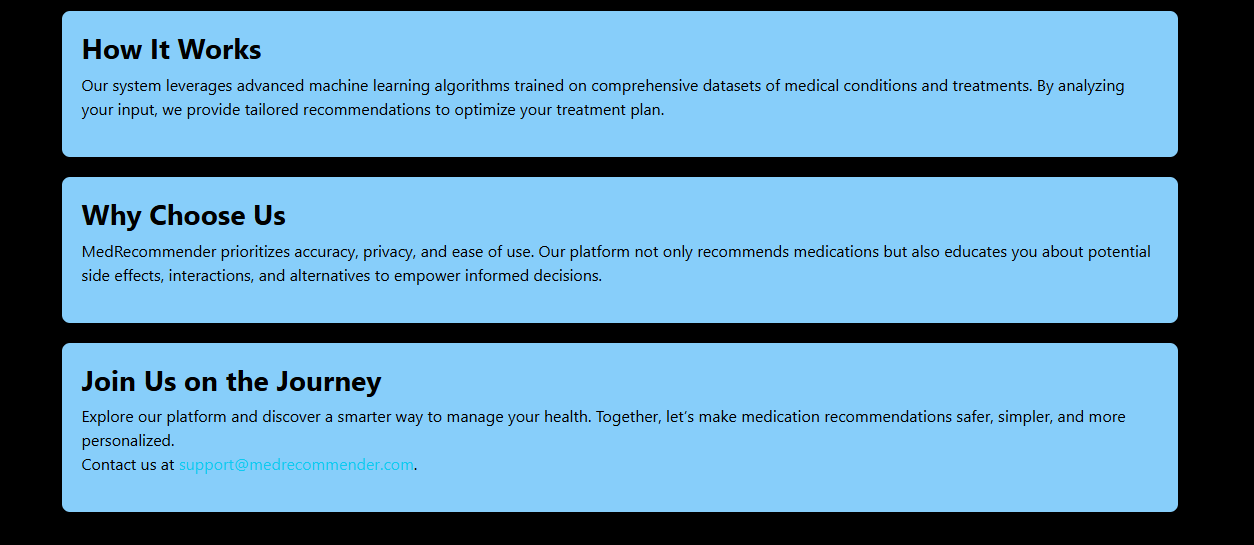
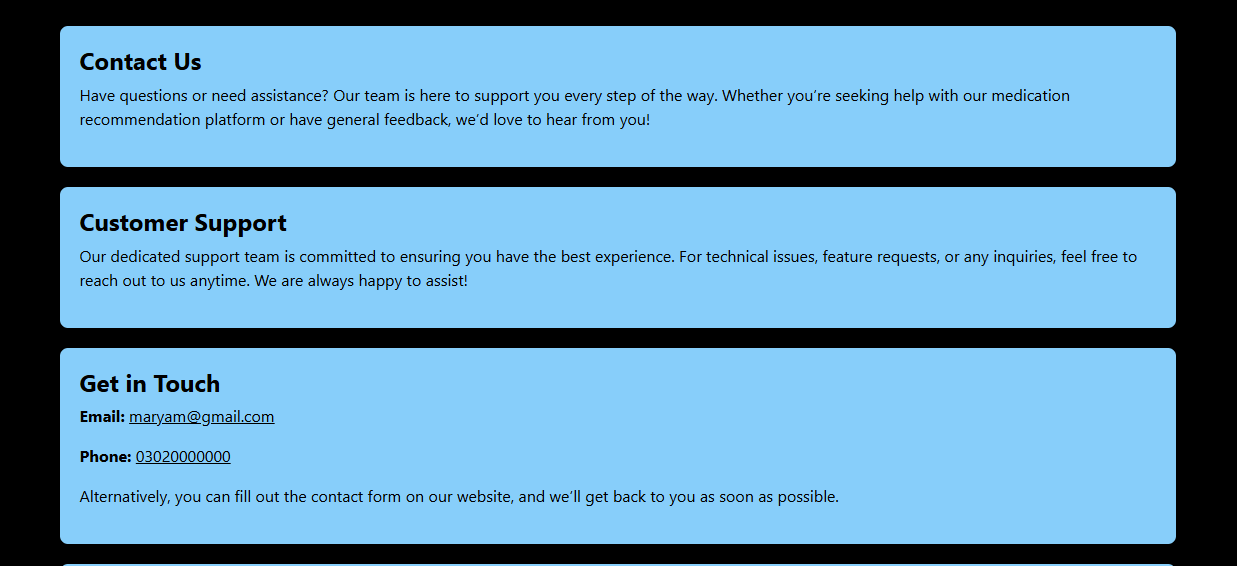
**Diets**

****

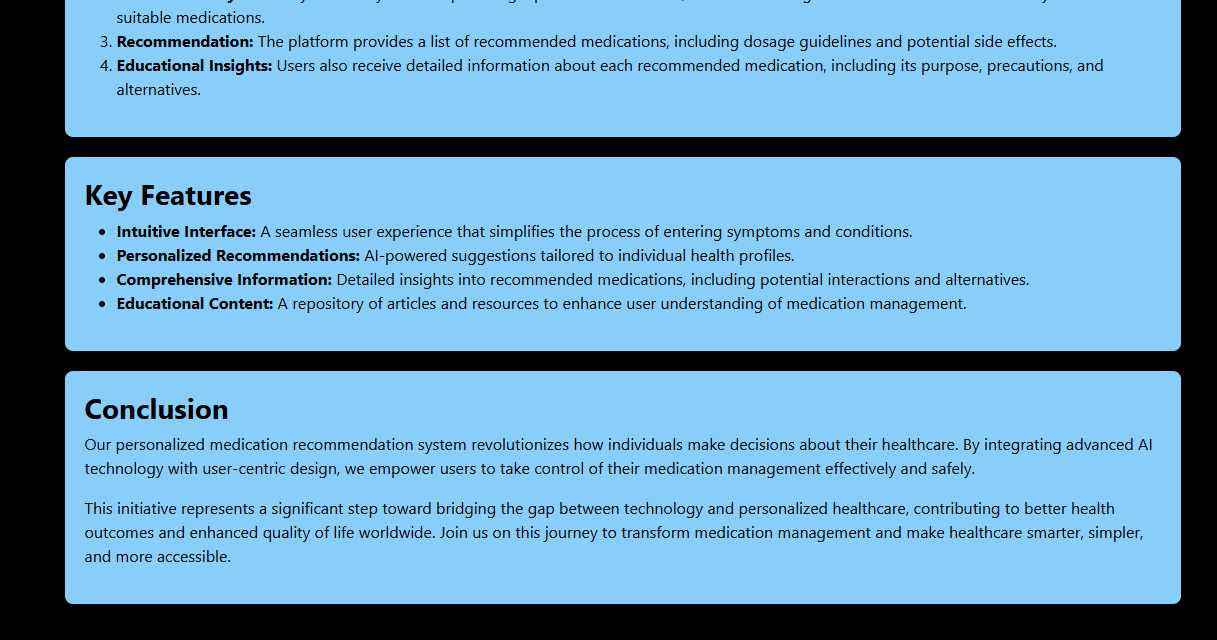
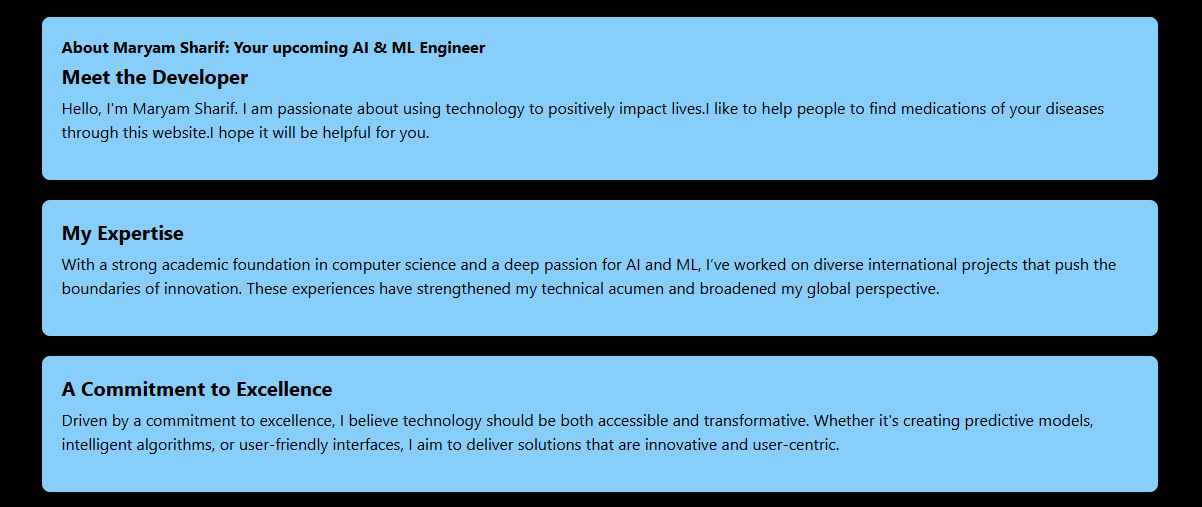
**Error Handling: Invalid Symptoms Input**

****

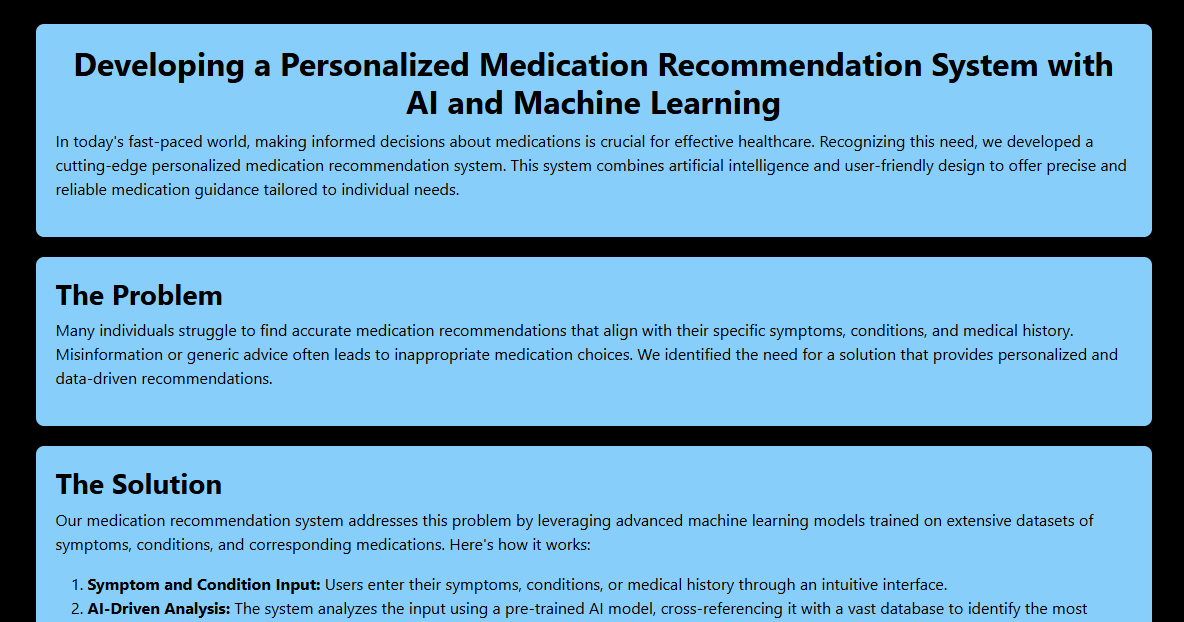
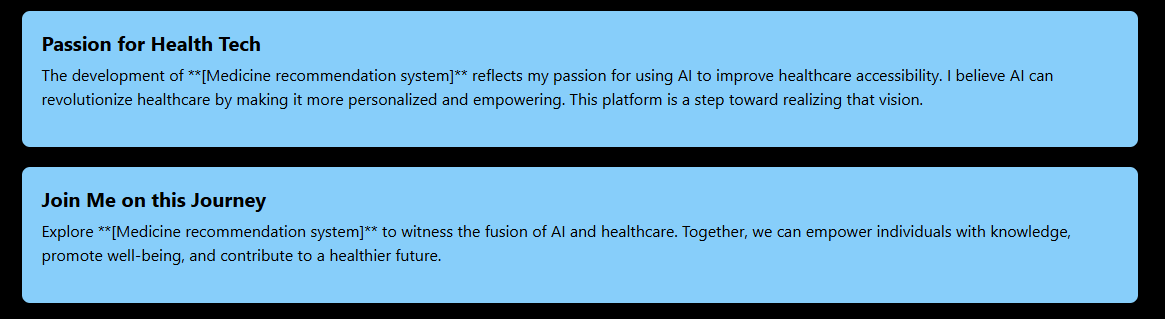
**About Us page**

**Contact Page**

**Developer**

****

**Blog**

****

**6. Challenges and Solutions**

**6.1 Challenges:**

* Managing ambiguous or misspelled user inputs.
* Ensuring comprehensive disease mapping for multi-symptom cases.

**6.2 Solutions:**

* Added input validation to handle incomplete or erroneous symptoms.
* Enhanced the symptom-disease mapping with additional rules.

**7. Future Enhancements**

* Extend the dataset to include more diseases and treatments.
* Incorporate **NLP** for natural language symptom input.
* Provide multilingual support for non-English speakers.

**8. Conclusion**

* The Medicine Recommendation System combines machine learning's predictive capability with a user-friendly web interface. Its integration of medical data into a simple Flask application demonstrates how AI can support healthcare decisions effectively.