STUDENT HEALTH CHECKER DOCUMENTATION

DCIT 317 GROUP 8

GROUP 8 UNIVERSITY OF GHANA

Contents

1. OVERVIEW

The **Students Health Checker** is an expert system created to help students assess potential health conditions based on their symptoms and medical history. The system guides users through a series of questions, collects details such as symptom duration and severity, and applies a rule-based inference engine to suggest possible conditions. Additionally, it provides medical advice and home remedies for symptom relief.

This project provides students with hands-on experience in programming and system design while demonstrating how technology can be leveraged to support healthcare.

2. HOW TO USE THE SYSTEM

Getting Started

Setup:

Make sure Python 3 is installed and then install Flask using the command:

pip install flask

Running the Application:

Go to the project folder ***(named health_symptom_checker)*** and run:

python app.py

Then open your web browser and visit ***http://127.0.0.1:5000/.***

Step-by-Step Instructions

1. Select Your Symptoms:

- On the first page, you will see a list of common symptoms such as fever, cough, and headache.
- Check the boxes next to the symptoms you are experiencing, then click the ***Next*** button.

2. Provide More Details:

- The next screen asks for additional information about each symptom you selected.
- You'll need to know how many days each symptom has lasted and choose how severe it is (mild, moderate, or severe).
- Once you've filled in these details, click Submit.

3. Use the “Go Back” Feature:

If needed, users can now return to the symptom selection page before submitting their details by clicking "Go Back".

4. View Your Results:

- The final page shows a summary of the symptoms and details you provided.
- It also displays possible conditions along with a brief explanation of each and advice on when to seek medical help (for example, recommending a hospital visit in urgent cases).
- Home remedies for symptom relief.
- If nothing matches your input, you’ll get a message advising you to see a healthcare professional.

5. Restarting:

- To run another check, simply click the ***Start Over*** button.

3. SYSTEM STRUCTURE

The application is built using Python and the Flask web framework. It is divided into several parts:

User Interface:

There are three main pages:

- **Symptom Selection (index.html):** Where you pick the symptoms you’re experiencing.
- **Details (details.html):** Where you provide extra information about each symptom.
- **Results (result.html):** Where the system shows possible health issues and advice.

These pages are styled using Bootstrap to make them visually appealing and easy to use.

Knowledge Base:

This is stored in a JSON file (knowledge_base.json). It includes:

- A list of common symptoms.
- A set of rules that link certain symptom patterns (including extra details like duration and severity) to specific diagnoses, along with short descriptions and advice.

Inference Engine:

The inference engine is implemented in app.py with two main functions:

- **evaluate_symptoms:** It checks all the rules against your inputs.
- **rule_matches:** It verifies if the user’s data meets the conditions of each rule.

This approach is a forward chaining method, meaning it starts with the input facts and applies rules to generate conclusions.

Application Flow:

The system uses Flask routes to control the flow:

- The index route handles symptom selection.
- The details route collects further information.
- The result route processes the inputs using the inference engine and displays the outcomes.

4. DESIGN CHOICES

Target Audience:

The system is intended for students. The user interface and instructions are designed to be simple and clear, making it easy for students to use and learn from.

Using JSON for the Knowledge Base:

JSON was chosen because it's easy to read and update. It lets you quickly add or change symptoms, conditions, and advice as needed.

Forward Chaining Inference:

We use a straightforward forward chaining technique. This means the system starts with what you tell it (the symptoms and their details) and then checks each rule to see if it fits your situation.

Bootstrap for UI:

The user interface is built with Bootstrap, which makes the pages look modern and responsive. This ensures better overall experience, especially for students.

Modular Design:

The system is split into distinct components (UI, knowledge base, inference engine) which makes it easier to update, test, and expand later.

5. CURRENT LIMITATIONS

Scope of the Knowledge Base:

The current set of symptoms and rules is limited. A real-world application would need a much larger database.

Simplicity of the Inference Engine:

The engine uses basic rule checking. In more complex scenarios, a more advanced system might be necessary.

Reliance on User Input:

The system depends on users accurately reporting their symptoms and details, which may not always happen.

No Real-Time Data Integration:

The system doesn't currently connect to live data sources such as hospital records. Future versions might include dynamic updates.

Not a Substitute for Professional Advice:

While the system can suggest potential conditions, it is not a replacement for professional medical diagnosis.

6. TESTING AND IMPROVEMENTS

Testing with Real Scenarios

Simulation:

Run the system and simulate different cases, like a prolonged fever or severe chest pain, to check that the correct rules trigger and that you receive the appropriate recommendations.

User Feedback:

Ask students and other users to try the system and share their thoughts on the flow of questions, the clarity of the interface, and the usefulness of the advice provided.

Iterative Refinement

Refine Questions:

Use feedback to adjust the phrasing of questions and the details required.

Update the Knowledge Base:

Expand or modify the rules based on test cases to improve the accuracy of diagnoses.

Enhance UI:

Tweak the design further if users suggest improvements for a smoother and more engaging experience.

7. SUMMARY

The Health Symptom Checker is an educational expert system that helps students assess potential health conditions. It uses a rule-based, forward chaining inference engine to process symptoms and extra details provided by the user. The system is divided into an intuitive user interface (with symptom selection, follow-up details, and results pages), a JSON-based knowledge base, and a simple inference engine. While the current version is limited in scope, it serves as a solid foundation for learning about expert systems and can be expanded with more data and improved reasoning techniques based on user feedback.