**Design**

The design of the Interactive Health Diagnosis System can be divided into several core components, covering both the **frontend** (Graphical User Interface - GUI) and **backend** (data processing and logic). This design ensures the system is modular, easy to maintain, and scalable for future enhancements. Below is a detailed breakdown of the design, including system architecture, data flow, and interaction between components.

**1. System Architecture**

The system follows a **Model-View-Controller** (MVC) design pattern to separate the logic, data, and user interface, ensuring each part can be modified independently without affecting others.

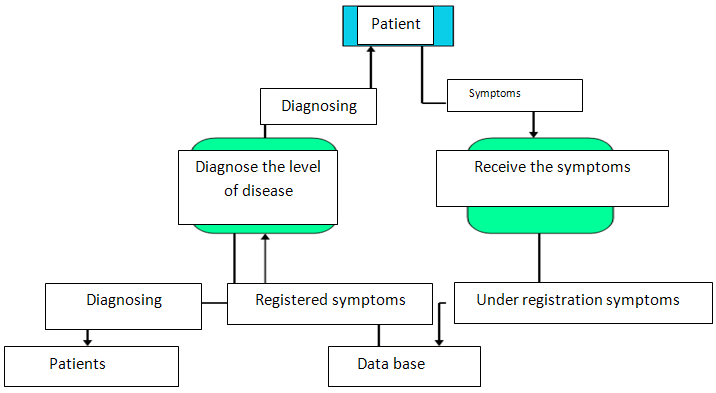
- Model: Responsible for handling data (loading and managing the CSV file).

- View: The graphical user interface (GUI) built using `tkinter` that interacts with the user.

- Controller: The core logic that processes user inputs, matches symptoms with the CSV data, and generates results.

**2. Data Flow Diagram (DFD)**

Here is a simple high-level overview of the data flow:



1. Input Stage: User inputs symptoms via the GUI.

2. Processing Stage:

- Input is passed to the backend.

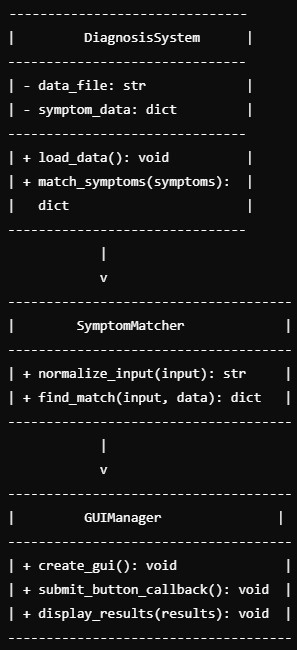
- Input is normalized and validated.

- Symptoms are matched with data in the CSV file.

3. Output Stage: Possible diagnoses, treatments, and causes are displayed on the GUI.

4. Feedback/Error Stage: If no matches are found or input is invalid, the system provides feedback.

**3. Class Diagram**



**4. User Interface (UI) Wireframe**



This design outlines the core components and structure of the Interactive Health Diagnosis System, including a well-organized GUI, data handling mechanisms, symptom matching logic, and user-friendly output presentation. By adhering to this design, the system can be efficiently developed, scaled, and maintained.