# **Hospital Registration System Documentation**

## **Overview**

The Hospital Registration System is a C++ program designed to manage patient information efficiently. It allows users (e.g., hospital staff or administrators) to:  
 - Register new patients.  
 - View all registered patients.  
 - Search for a patient by their unique ID.  
 - Edit a patient's information.  
 - Delete a patient's record.  
  
 The program stores data in a CSV (Comma-Separated Values) file called 'patients.csv', which preserves data even after the program is closed. This makes it easy to access and manage patient data whenever the program is opened.

## **Table of Contents**

1. Features  
 2. System Requirements  
 3. File Structure  
 4. Data Structure  
 5. Functions  
 6. How to Use  
 7. Code Explanation  
 8. Future Enhancements  
 9. Acknowledgments

## **Features**

- Add New Patient: Register a new patient and save their information to a CSV file.  
 - Search Patient by ID: Look up a patient’s details by entering their unique ID.  
 - Display All Patients: View the list of all registered patients and their details.  
 - Edit Patient Information: Modify an existing patient's details.  
 - Delete Patient Record: Remove a patient's record from the system.  
 - Data Persistence: Data is saved to a CSV file, ensuring it remains accessible every time you open the program.

## **System Requirements**

To run this program, you need:  
 - A C++ Compiler: Examples include GCC, Clang, or Visual Studio. Make sure your compiler supports C++11 or later.  
 - File Access Permissions: The program requires permission to read and write files in the directory where it’s located, to access 'patients.csv'.

## **File Structure**

The system uses a file named 'patients.csv' to store patient data in a format that looks like this:  
 ID, Name, Age, Gender, Contact, MedicalHistory  
  
 Each line represents a single patient’s data, with values separated by commas. Example:  
  
 101, John Doe, 30, M, 123-456-7890, No known allergies  
 102, Jane Smith, 27, F, 987-654-3210, Asthma

## **Data Structure**

The program defines a 'struct' for patients, grouping relevant fields together.

struct Patient {  
 int id;  
 string name;  
 int age;  
 string gender;  
 string contact;  
 string medicalHistory;  
 };

Field Descriptions:  
 - id: A unique identifier for each patient.  
 - name: The patient’s full name.  
 - age: The patient’s age.  
 - gender: The patient’s gender (e.g., 'M' or 'F').  
 - contact: The patient’s contact number.  
 - medicalHistory: Brief description of the patient’s medical history.

## **Functions**

Core Functions  
 1. addPatient() - Collects and saves patient details.  
 2. loadPatients() - Loads data from 'patients.csv'.  
 3. searchPatientById() - Searches for a patient by ID.  
 4. displayAllPatients() - Lists all patients.  
 5. editPatientById() - Edits patient details by ID.  
 6. deletePatientById() - Deletes patient details by ID.  
 7. menu() - Displays an interactive menu for program options.  
  
 Utility Function  
 - getIntInput(string prompt): Validates integer inputs.

## **How to Use**

1. Compile the Program:  
 Compile the source code with a C++ compiler, e.g.,  
 g++ -o hospital\_system hospital\_system.cpp  
  
 2. Run the Program:  
 Run the compiled program:  
 ./hospital\_system  
  
 3. Use the Main Menu:  
 Choose an option by entering the corresponding number.  
  
 4. Add a Patient:  
 Input details to register a new patient.  
  
 5. Search for a Patient:  
 Enter the patient ID to retrieve details.  
  
 6. Display All Patients:  
 View all registered patients.  
  
 7. Edit a Patient:  
 Enter the ID of a patient to modify their information.  
  
 8. Delete a Patient:  
 Enter the ID of a patient to remove their record.

## **Code Explanation**

File Handling  
 - Data Addition: addPatient() uses ofstream to append new data to 'patients.csv'.  
 - Data Loading: loadPatients() uses ifstream to read data from 'patients.csv'.  
 - Data Saving: savePatients() saves updated records, used for editing and deleting.  
  
 Input Validation  
 - getIntInput(): Ensures valid integer input to prevent errors.  
  
 Error Handling  
 - Displays an error if 'patients.csv' is inaccessible.  
  
 Edit and Delete Functions  
 - editPatientById(): Loads patients, finds the specified ID, and allows field edits.  
 - deletePatientById(): Loads patients, finds and removes a patient by ID, and saves the updated data.

## **Future Enhancements**

1. Update Patient Information: Allow further customization of fields.  
 2. Advanced Search: Add filtering by multiple fields (e.g., name, contact).  
 3. Appointment Scheduling: Manage appointments with additional date/time fields.  
 4. Data Encryption: Secure 'patients.csv' for privacy compliance.

## **Acknowledgments**

This project was created with the support of the C++ programming community, with various tutorials and resources available online. Special thanks to WVSU professors as well for the guidance that is given to us throughout the journey