



# Daffodil International University

*Department of Software Engineering*

## Project Proposal

---

### Healthcare Management System

Course Name: Capstone Project

Course Code: SE 133

#### Submitted to:

Khalid Masum

Lecturer

Department of Software Engineering

Daffodil International University

#### Submitted by:

*Students from section 44-J2*

Name	Registration Number
Zunait Hossain Ratul (Leader)	251-35-362*
Arvin yefat orin	251-35-571
Md Ridoy Ali	251-35-582
Musfiqur Rahman Shabuj	251-35-090

*Date of Submission: December 02, 2025*

# **Table of Contents** (clickable)

- I. Introduction
- II. Use Case Diagram
- III. Functional Requirements
- IV. Non-Functional Requirements
- V. Requirements Priority Summary
- VI. Data Design
- VII. System Constraints
- VIII. Challenges
- IX. Development Plan & Timeline
- X. Deliverables
- XI. Future Enhancements
- XII. GitHub Links

# 1. Introduction

## 1.1 Problem Statement

Small clinics and healthcare centers in resource constrained environments face significant operational challenges due to reliance on paper-based record keeping systems. These manual processes lead to:

- **Inefficiency:** Time-consuming patient registration and data retrieval
- **Data Loss Risk:** Physical records are vulnerable to damage, loss, or misplacement
- **Scheduling Conflicts:** Manual appointment booking often results in double-bookings
- **Billing Errors:** Manual calculations increase the risk of financial discrepancies
- **Limited Accessibility:** Patient history is difficult to track across multiple visits

These challenges reduce operational efficiency, increase administrative burden, and compromise the quality of patient care.

## 1.2 System Overview

The Healthcare Management System (HMS) addresses these challenges by providing a CLI-based application built in C to streamline operations in small clinics. The system manages patient records, doctor schedules,

appointments, and billing digitally, replacing paper-based systems.

### **Key Features:**

- Patient and doctor information management
- Appointment scheduling with conflict prevention
- Billing and payment tracking
- Role-based access control
- Data backup and restore

### **1.3 Project Scope**

#### **In Scope:**

- Patient registration and records management
- Doctor profile management
- Appointment scheduling
- Billing and payment tracking
- User authentication with role-based access
- Basic reporting
- Data backup and restore functionality

#### **Out of Scope:**

- Network/LAN capabilities
- Medical imaging and Lab test integration
- Insurance processing and claims
- Pharmacy and Inventory management
- SMS/Email notifications

- Web or mobile interface
- Multi-user concurrent access

## 1.4 User Roles & Permissions

HMS supports three user roles with file-based data storage:

Role	Responsibilities
Administrator	User management, doctor profiles, reports, backup
Doctor	View schedule, patient records, add notes
Receptionist	Patient registration, appointments, billing

## Environment:

- Cross-platform CLI

## System Requirement:

- **Architecture:** x86, x64, arm
- **OS:** Windows, linux, macOs
- **Storage:** 100MB minimum
- **RAM:** 256MB minimum

## 1.5 Stakeholders

The Hospital Management System involves multiple stakeholders, each with distinct roles and interests in the project's success:

### 1.5.1 Primary Stakeholders

#### **Healthcare Clinic Staff:**

- **Interest:** Streamlined daily operations, reduced administrative burden
- **Impact:** Direct users who rely on the system for patient registration, appointment scheduling, and billing
- **Needs:** User-friendly interface, reliable performance, minimal training requirements

#### **Clinic Administrators:**

- **Interest:** Operational efficiency, accurate reporting, financial tracking
- **Impact:** Decision-makers who use system reports for clinic management
- **Needs:** Comprehensive reports, user management capabilities, data backup features

#### **Doctors:**

- **Interest:** Quick access to patient information, efficient schedule management
- **Impact:** Medical professionals who need patient history for consultations
- **Needs:** Easy patient record access, clear daily schedule view, minimal time investment

### **Receptionists:**

- **Interest:** Simplified patient registration and appointment management
- **Impact:** Front-desk staff handling majority of system interactions
- **Needs:** Fast search capabilities, conflict-free appointment scheduling, straightforward billing

### **1.5.2 Secondary Stakeholders**

#### **Patients:**

- **Interest:** Accurate record-keeping, reduced waiting times, proper billing
- **Impact:** Indirect beneficiaries through improved clinic services
- **Needs:** Data privacy, accurate medical history, transparent billing

#### **Project Development Team:**

- **Interest:** Successful project completion, learning experience, portfolio addition
- **Impact:** Students responsible for design, development, and deployment
- **Needs:** Clear requirements, manageable scope, technical support

#### **Academic Supervisors:**

- **Interest:** Educational objectives met, quality capstone project
- **Impact:** Faculty evaluating project success and student learning

- **Needs:** Proper documentation, adherence to best practices, timely completion

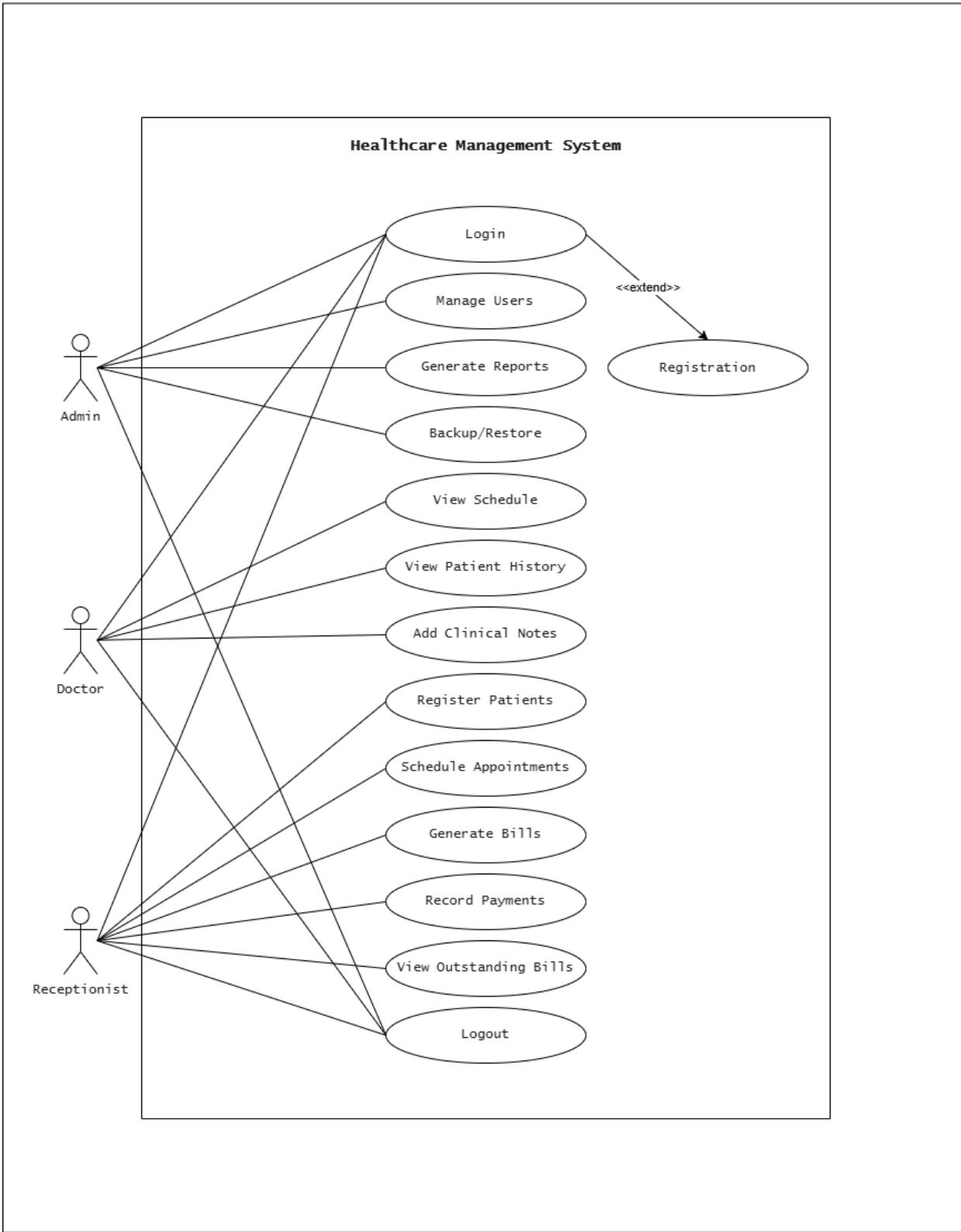
### **Future Developers/Maintainers:**

- **Interest:** Maintainable codebase, comprehensive documentation
- **Impact:** Those who may extend or modify the system
- **Needs:** Clean code, detailed comments, technical documentation

### **IT Support/Technical Staff:**

- **Interest:** Easy system deployment, troubleshooting capabilities, stable performance
- **Impact:** Personnel responsible for installation, configuration, and ongoing technical support
- **Needs:** Clear installation instructions, system requirements documentation, diagnostic tools, minimal dependencies

## 2. Use Case Diagram



## 2.1 System Modules

Healthcare Management System

- Authentication Module
- Patient Management Module
- Doctor Management Module
- Appointment Management Module
- Billing Module

## 3. Functional Requirements

### 3.1 Authentication Module:

#### FR-101: User Login

- As a system user
  - I want to log in securely with username and password
  - So that I can access role-specific functions
- 

#### FR-102: Password Management

- As a logged-in user
  - I want to change my password
  - So that I can maintain account security
-

### **FR-103: System Logout**

- As a logged-in user
  - I want to securely logout from the system
  - So that my session is terminated and data is protected
- 

### **FR-104: View System Info**

- As a user
  - I want to view developer and system details
  - So that I can see project information and credits
- 

## **3.2 Patient Management Module:**

### **FR-201: Register New Patient**

- As a receptionist
  - I want to register patients with demographic and medical information
  - So that their records are available for healthcare services
- 

### **FR-202: Update Patient Information**

- As a receptionist
- I want to modify patient demographic information

- **So that** records remain current and accurate
- 

### **FR-203: Search Patients**

- **As a** receptionist
  - **I want to** search patients by name, ID, or phone number
  - **So that** I can find specific records even if I don't know the exact ID
- 

### **FR-204: View Patient Profile**

- **As a** receptionist or doctor
  - **I want to** view complete patient information and appointment history
  - **So that** I can provide appropriate care and service
- 

### **FR-205: Patient Visit History**

- **As a** doctor or receptionist
  - **I want to** view chronological history of patient visits
  - **So that** I can track treatment progress over time
-

### **3.3 Doctor Management Module:**

#### **FR-301: Manage Doctor Profiles**

- As an administrator
  - I want to add and update doctor information
  - So that doctor schedules and fees are accurate
- 

#### **FR-302: View Doctor List**

- As a receptionist or administrator
  - I want to view all active doctors with specializations
  - So that I can assist patients in doctor selection
- 

### **3.4 Appointment Management Module:**

#### **FR-401: Schedule Appointment**

- As a receptionist
  - I want to book appointments with date and time
  - So that patient visits are organized and double-booking is prevented
-

## **FR-402: View Appointments**

- As a receptionist or doctor
  - I want to view appointments by date or doctor
  - So that I can manage schedules effectively
- 

## **FR-403: Cancel Appointment**

- As a receptionist
  - I want to cancel appointments with reason
  - So that slots become available and records are updated
- 

## **FR-404: Doctor's Daily Schedule**

- As a doctor
  - I want to view my appointments for selected date
  - So that I can prepare for consultations
- 

## **FR-405: Reschedule Appointment**

- As a receptionist
  - I want to change appointment date or time
  - So that I can accommodate patient requests
-

### **3.5 Billing Module:**

#### **FR-501: Generate Bill**

- As a receptionist
  - I want to create bills for consultation fees
  - So that patients are charged accurately
- 

#### **FR-502: Record Payment**

- As a receptionist
  - I want to record payment amount and method
  - So that billing records are complete
- 

#### **FR-503: Track Unpaid Bills**

- As a receptionist
  - I want to view all unpaid and partially paid bills
  - So that I can follow up on pending payments
- 

### **3.6 Reporting Module:**

#### **FR-601: Patient Registration Report**

- As an administrator
- I want to generate patient registration statistics

- So that I can track clinic growth
- 

### **FR-602: Revenue Summary Report**

- As an administrator
  - I want to view total revenue collected
  - So that I can monitor financial performance
- 

### **FR-603: Appointment Statistics**

- As an administrator
  - I want to generate appointment statistics by doctor or date
  - So that I can analyze clinic utilization
- 

### **FR-604: Export Patient List**

- As an administrator
  - I want to export the patient list to a CSV/text file
  - So that I can use the data in other applications
-

## **4. Non-Functional Requirements**

### **NFR-1: Performance**

- System responds quickly to user inputs
- Search operations complete in reasonable time
- No noticeable delays in operations

### **NFR-2: Security**

- Username and password authentication required
- Simple XOR Cipher for passwords
- Role-based access control enforced

### **NFR-3: Usability**

- Clear numbered menu system
- Informative error messages
- Confirmation prompts for critical operations
- Consistent interface layout

### **NFR-4: Reliability**

- Input validation prevents invalid data
- Error handling prevents crashes
- Backup and restore functionality available

### **NFR-5: Maintainability**

- Modular code structure
- Well-commented code
- Functions kept reasonably short
- Clear README with compilation instructions

## 5. Requirements Priority Summary

ID	Description	Priority
FR-101	User Login	Must Have
FR-102	Password Management	Should Have
FR-103	System Logout	Must Have
FR-104	View System Info	Should Have
FR-201	Register New Patient	Must Have
FR-202	Update Patient Information	Must Have
FR-203	Search Patients	Must Have
FR-204	View Patient Profile	Must Have
FR-205	Patient Visit History	Could Have
FR-301	Manage Doctor Profiles	Must Have
FR-302	View Doctor List	Must Have
FR-401	Schedule Appointment	Must Have
FR-402	View Appointments	Must Have
FR-403	Cancel Appointment	Must Have
FR-404	Doctor's Daily Schedule	Must Have
FR-405	Reschedule Appointment	Should Have
FR-501	Generate Bill	Must Have
FR-502	Record Payment	Must Have
FR-503	Track Unpaid Bills	Should Have
FR-601	Patient Registration Report	Could Have
FR-602	Revenue Summary Report	Should Have
FR-603	Appointment Statistics	Could Have
FR-604	Export Patient List	Could Have

# **6. Data Design**

## **6.1 Core Data Entities**

**User:** UserID, Username, Password, Role, Full Name, Contact

**Patient:** PatientID, Full Name, DOB, Gender, Contact, Address, Blood Group, Allergies

**Doctor:** DoctorID, Full Name, Specialization, Contact, Consultation Fee, Available Days

**Appointment:** AppointmentID, PatientID, DoctorID, Date, Time, Status, Reason

**Bill:** BillID, PatientID, AppointmentID, Amount, Payment Status, Payment Method

## **6.2 Validation Rules**

- **Contact Number:** Exactly 11 digits
- **Name:** Alphabets and spaces only, min 3 characters
- **Password:** Minimum 6 characters, case-sensitive
- **Age:** 0-120 years
- **Appointment Date:** Must be today or future date
- **Blood Group:** Restricted to valid set {A+, B+, ...}

## 7. System Constraints

- Single user system (no concurrent access)
- File-based storage only (no database)
- Console based interface (no GUI)
- Not HIPAA/healthcare compliant
- Platform-dependent file paths (Windows/Linux)

## 8. Challenges

### 8.1 Technical Challenges

- **Data Consistency:** Managing file pointers and ensuring data integrity without a relational database.
- **Memory Management:** Efficiently handling patient records within limited RAM using C structures.
- **Input Validation:** Robustly sanitizing all user inputs to prevent system crashes.
- **Cross-Platform Compatibility:** Ensuring the CLI application works correctly on Windows and Linux environments.

### 8.2 Operational Challenges

- **Time Constraints:** Completing full-stack CLI implementation within the 8-week semester timeline.
- **Resource Limitations:** Limited to 4 developers for design, coding, testing, and documentation.

## 9. Development Plan

### 9.1 Timeline (8 Weeks)

Week	Module	Tasks
1	Planning	SRS, design, setup
2	Foundation	Authentication, file I/O, menu
3	Patient Module	CRUD operations
4	Doctor & Appointment	Management, scheduling
5	Billing	Bill generation, payments
6	Reports & Admin	Reports, backup
7	Testing	Unit, integration, system tests
8	Documentation	User manual, presentation

### 9.2 Testing Approach

- Unit Testing:** Individual function testing
- Integration Testing:** Module interaction testing
- System Testing:** End-to-end workflow testing
- User Acceptance:** Validate against requirements

## 10. Deliverables

### Software

- Source code (.c and .h files)
- Compiled executable
- Makefile

### Documentation

- Software Requirements Specification
- README.md with setup instructions

## 11. Future Enhancements

The following features are planned for future versions to expand system capabilities:

- 1. Database Integration:** Migrate from file-based storage to SQL database for better scalability.
- 2. Graphical User Interface (GUI):** Develop a desktop UI using C# or Java for better usability.
- 3. Network Support:** Enable multi-user access over LAN for simultaneous operations.
- 4. SMS Notifications:** Integrate API to send appointment reminders to patients.
- 5. Prescription Printing:** Auto-generate printable prescriptions from doctor notes.

## 12. GitHub Links

### Leader's Repository

**Repository:** [Hospital-Management-System \(Main\)](https://github.com/AmiValoHoteChai/Hospital-Management-System)

**Direct Link:**

<https://github.com/AmiValoHoteChai/Hospital-Management-System>

**Leader:** Zunait Hossain Ratul (AmiValoHoteChai)

### Team Member's Forked Repositories

- **Orin:** <https://github.com/251-35-571-rgb/Hospital-Management-System.git>
- **Ridoy:** <https://github.com/Ridoy1237/Hospital-Management-System.git>
- **Mushfik:** <https://github.com/musigit2004/Hospital-Management-System.git>

**Note:** All team members will fork the leader's repository and contribute via pull requests following Git collaborative workflow. The main repository links is provided as both a hyperlink and direct link.

**EOF (End of File)**