

# PROJECT BASED LEARNING REPORT

On  
HOSPITAL MANAGEMENT SYSTEM

***Submitted by:***

Mustak - 1/24/SET/BCS/080

**Under the Guidance of:**

Ms. Shobha Tyagi

*in partial fulfillment for the award of the degree of*

**BACHELOR OF TECHNOLOGY**  
**IN**  
**Computer Science & Engineering**



**School of Engineering & Technology**

**MANAV RACHNA INTERNATIONAL INSTITUTE OF  
RESEARCH AND STUDIES, Faridabad NAAC  
ACCREDITED 'A++' GRADE  
2025**

## **ABSTRACT**

A Hospital Management System (HMS) is a computerized platform designed to efficiently manage the day-to-day operations of a healthcare facility. This system streamlines administrative, clinical, and financial processes by integrating patient information, appointment scheduling, billing, medical records, and staff management into a unified interface. The primary objective of the HMS is to enhance the accuracy, accessibility, and security of hospital data while reducing manual workload and operational delays. By automating routine tasks, the system improves coordination among departments, minimizes errors, and supports better decision-making for both medical and administrative staff. Overall, the Hospital Management System increases productivity, ensures timely patient care, and contributes to the effective functioning of healthcare services.

# TABLE OF CONTENTS

<b>Table</b>	<b>Content</b>	<b>Page No</b>
1	Abstract	01
2	Table of Contents	02
3	Introduction	03
4	Objectives	04
5	Methodology	04
6	Mathematical & Logical Concepts	05
7	Uml Diagram	05
8	Requirements	08
9	Program Code	08
10	Results	09
11	Limitations & Observations	10
12	Future Enhancements	10
13	Conclusion	10
14	References	11

## INTRODUCTION

The aim of this project is to design and implement a simple and user-friendly Hospital Management System (HMS) using the concepts of Object-Oriented Programming. The main purpose of the system is to make basic hospital record-keeping easier by allowing users to add, view, update, and manage details of patients, doctors, and appointments in an organised manner. This project also aims to show how backend logic written in C++ (using OOP concepts like classes, objects, inheritance, and encapsulation) can work together with a web-based frontend built using HTML, CSS, and JavaScript. The system tries to provide a smooth interaction between data storage and user interface so that the information is displayed clearly to the user.

In simple words, the aim is :

- To build a small management system that works properly and looks clean.
- To apply OOP concepts in a real-world situation.
- To show how C++ logic can be connected with a web page.
- To design a working mini-project that can be used for learning and demonstration.

## OBJECTIVES

- Improve patient care by maintaining accurate and updated medical records.
- Streamline hospital operations such as appointments, admissions, and billing.
- Enhance communication between doctors, nurses, and administrative staff.
- Reduce paperwork through digital record-keeping and automation.
- Increase efficiency & transparency in resource management and reporting.

## METHODOLOGY

The development of the project followed a clear and practical workflow :

### 1. Requirement Analysis

We identified the core features the system must support, such as adding patients, doctors, and appointments, and displaying all records on the interface.

### 2. Object-Oriented Design

We planned the necessary OOP structure and created the main classes: Person, Patient, Doctor, Appointment, and Hospital.

### 3. Backend Implementation (C++)

The data-handling logic was implemented using vectors to store and manage all records efficiently.

### 4. Frontend Development (HTML, CSS)

Basic UI pages were built for the dashboard, patient list, doctor list, and appointment list.

### 5. Integration of Frontend and Backend

JavaScript was used to bridge the frontend with the C++ backend, enabling dynamic display of data on the web interface.

### 6. Testing and Verification

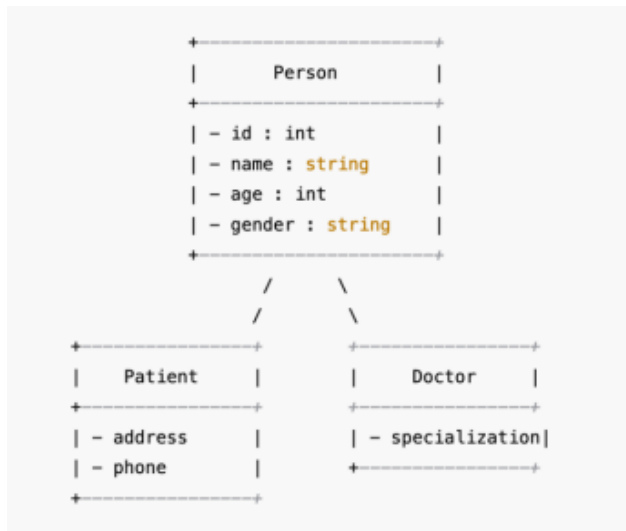
Sample data was inserted, and the application was tested thoroughly to ensure that all functions performed as expected.

## MATHEMATICAL & LOGICAL CONCEPTS

- Use of vectors for storing records
- Linear search for matching names or IDs
- Basic string operations
- Simple condition checks
- Use of counters for generating unique IDs
- Pipe-separated data formats for sending and receiving lists

## UML DIAGRAM

UML diagram for our project



# **REQUIREMENTS**

## **Software Requirements**

- C++ compiler
- Browser with HTML/CSS/JS support
- Text editor (VS Code, Notepad++, etc.)
- System with local storage support

## **Hardware Requirements**

- Any basic computer or laptop
- Minimum 2 GB RAM
- Browser-compatible system

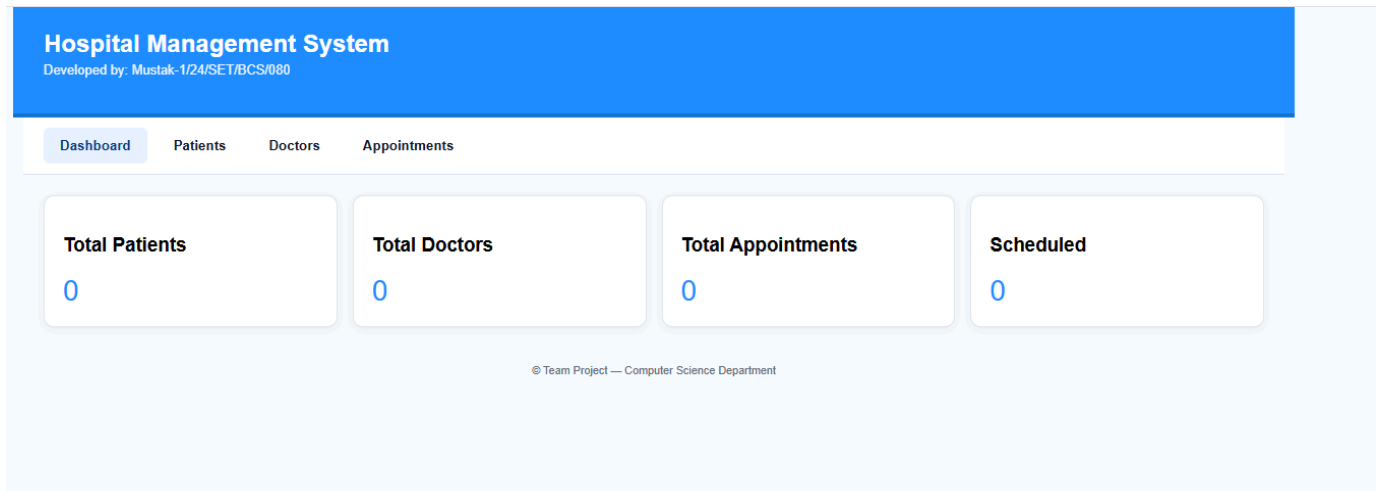
# **PROGRAM CODE**

Instead of adding large code, paste the following screenshots :

1. C++ Backend Logic (Take screenshot of your hms.cpp file showing class structure.)
2. Frontend HTML Page (index.html main structure)
3. JavaScript Functions (script.js: functions for adding patient/doctor)
4. CSS Design (styles.css: basic styling)

# RESULTS

Below are the results screenshot



**ID Name Age Gender Address Phone Actions**

## Register Patient

Name \*  Age \*  Gender  Address

Phone \*



## **LIMITATIONS & OBSERVATIONS**

- Data is stored temporarily in browser storage.
- No database or authentication system.
- IDs change when reloading from local storage.
- Suitable only for small-scale demonstration.

## **FUTURE ENHANCEMENTS**

- Add a proper database for permanent storage.
- Add a user login system.
- Improve UI with more modern components.
- Add editing options for patients and doctors.

## **CONCLUSION**

This project helped us understand how Object-Oriented Programming can be used in real applications. By designing a simple Hospital Management System, we learned how to structure a program using classes and objects, how to connect a backend with a frontend, and how to make a clean and useful interface. The project is simple, easy to use, and demonstrates all the basic concepts of OOP clearly. It also shows how different technologies can work together in a single system. Overall, this project was a complete learning experience and helped us understand both logic and design in a better way.

## REFERENCES

1. C++ Documentation – [cplusplus.com](http://cplusplus.com) Used for understanding basic syntax, string handling, vectors, and object-oriented principles.
2. GeeksforGeeks – Object Oriented Programming Concepts Referred for clarity on concepts like inheritance, encapsulation, and class design.
3. MDN Web Docs (Mozilla Developer Network) Used to understand JavaScript DOM functions, event handling, and local storage usage
4. W3Schools – HTML, CSS & JavaScript Tutorials Helpful for designing and styling the frontend part of the project.
5. Project Source Code (hms.cpp, script.js, index.html, styles.css) The main project files provided for implementation and demonstration.
6. Class Notes & Faculty Instructions Used as reference for formatting, project structure, and expected output of the PBL report.