Hospital Management System

PROJECT REPORT

Submitted by

Aman Kumar (23BCS14314)

Govind (23BCS10198)

Prakram Pundeer (23BCS12232)

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE ENGINEERING

Chandigarh University

BONAFIDE CERTIFICATE

Certified that this project report **"Hospital Management System"** is the bonafide work of **"Aman Kumar(23BCS14314), Govind(23BCS10198) and Prakram Pundeer (23BCS12232)"** who carried out the project work under my/our supervision.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

Submitted for the project viva-voce examination held on

ACKNOWLEDMENT

I would like to express my sincere gratitude to all those who provided their support, guidance, and encouragement throughout the development of this project, titled **"Hospital Management System".**

First and foremost, I thank the **Department of Computer Science & Engineering, Chandigarh University**, for providing me with the opportunity to undertake this project as part of the academic curriculum.

I would like to extend my heartfelt thanks to my project guide, **Rahul Durgalpal sir** for his valuable insights, constant encouragement, and continuous supervision during the course of the project. Their constructive suggestions and timely feedback were instrumental in shaping the direction of this work.

I am also thankful to my faculty members and lab instructors who provided technical knowledge and moral support throughout the semester.

I wish to acknowledge my friends and classmates for their cooperation, brainstorming sessions, and support during development and testing.

Last but not least, I express my deep gratitude to my **family** for their unwavering support, patience, and motivation during the completion of this project.

This project has given me the opportunity to enhance my technical skills, understand real-world application development, and appreciate the importance of efficient data management in the real estate domain.

TABLE OF CONTENTS

List of Figures

List of Tables

ABSTRACT

The **Hospital Management System (HMS)** developed in the C programming language is a console-based software designed to simplify and automate the routine administrative and operational tasks of a hospital. The project aims to replace the traditional manual system of maintaining hospital records with an efficient, accurate, and reliable computerized system. Through this system, different modules such as patient management, doctor information management, appointment scheduling, pharmacy control, and billing processes are integrated into a single, user-friendly platform. The entire application is file-based, where each module's data is stored and retrieved from corresponding CSV files to ensure persistent storage without requiring a database engine. This makes the system lightweight, portable, and suitable for small healthcare facilities or educational environments where simplicity and clarity are desired.

The main motivation behind developing this project is the growing need for effective data handling and retrieval in hospitals. In a traditional manual setup, record-keeping involves large amounts of paperwork, redundancy, and a higher risk of errors or data loss. By contrast, the computerized system not only reduces these issues but also ensures that critical information such as patient medical history, doctor specialization, or billing details is readily available and can be accessed instantly by authorized users. The system provides a menu-driven interface where users can easily navigate through various options such as adding a new patient, searching for an existing record, or generating a new bill. Moreover, each entry is automatically assigned a unique identification number, ensuring traceability and eliminating duplication.

The HMS follows a modular architecture, dividing the functionality into independent yet interconnected sections. The **Patient Module** manages the registration and record-keeping of patients. It allows adding, listing, searching, and deleting records. The **Doctor Module** maintains details about medical professionals, their specialization, and contact information. The **Appointment Module** establishes a direct link between patients and doctors by scheduling and storing appointments with proper date and time information. The **Pharmacy Module** keeps track of medicines, their quantities, and prices, thereby facilitating stock management. The **Billing Module** integrates with the pharmacy system to automatically generate bills based on prescribed medicines and their quantities, updating the stock accordingly. These modules collectively form a cohesive and systematic hospital management process that ensures smooth operations and better service delivery.

The software design prioritizes simplicity, security, and data integrity. Input validation functions and file-handling mechanisms have been implemented to prevent incorrect data entry and to handle exceptions gracefully. For security, the system employs an authentication mechanism that restricts access to authorized personnel only through a predefined admin password. This ensures that sensitive information is protected from unauthorized users. Furthermore, every data manipulation—whether adding, deleting, or updating—is reflected instantly in the associated CSV file, maintaining data consistency across sessions.

The project has been tested for different scenarios including invalid inputs, missing files, and extreme cases such as empty records or out-of-stock medicines. The validation results confirm that the system performs reliably under such conditions. The use of plain text files makes the system platform-independent and easy to deploy on any operating system supporting standard C compilers like GCC or Turbo C.

CHAPTER 1.
INTRODUCTION

1.1 Introduction

The **Hospital Management System (HMS**) is designed for hospitals, clinics, and healthcare centres to manage patient data, doctor details, appointments, pharmacy records, and billing in a systematic manner. The need for this system arises due to inefficiencies in manual data handling, which often leads to human errors, data loss, and time delays. In today's digital era, healthcare institutions are adopting automation for improved accuracy and efficiency. The identified problem is the absence of an integrated system that can handle all hospital operations through a single platform. Major tasks include data collection, module development, file management, and testing. The project follows a structured timeline: planning, coding, testing, and documentation. The report is organized into chapters that describe the system's background, design methodology, results, and future enhancements. This project demonstrates how C programming and file handling can create an effective, low-cost hospital management solution for small to medium healthcare organizations.

1.2 Identification of Client and Need

The client for this project is any hospital, clinic, or healthcare institution that manages large volumes of data related to patients, doctors, medicines, and billing. In such organizations, maintaining records manually often results in inefficiency, data redundancy, and the risk of human error. With the increasing dependency on technology in every domain, the healthcare sector requires a computerized system that can handle information effectively and ensure smooth operations. The need for a **Hospital Management System** arises to replace outdated manual processes with an automated, reliable, and user-friendly solution. The system helps administrators to store, update, and retrieve data quickly, thus saving time and improving accuracy. Moreover, it ensures data consistency across multiple operations like patient registration, appointment scheduling, and billing. This computerized approach enhances hospital workflow, reduces paperwork, and allows better coordination between various departments, ultimately leading to improved patient service and operational efficiency.

1.3 Relevant Contemporary Issues

In the modern era, hospitals are increasingly adopting digital technologies to improve efficiency and

service quality. However, many small and medium-sized healthcare institutions still rely on manual record-keeping due to high software costs and lack of technical expertise. This leads to delays, data loss, and reduced productivity. With the growing emphasis on digital transformation, patient data privacy, and quick accessibility of medical records, the need for affordable and secure hospital management systems has become a key contemporary issue. The proposed **Hospital Management System**, developed in C language, addresses these challenges by offering a simple, offline, and cost-effective solution that ensures accuracy, data security, and ease of operation even in resource-limited healthcare environments.

1.4 Problem Identification

Hospitals and healthcare institutions handle extensive data daily, including patient records, doctor details, appointments, pharmacy stock, and billing information. Traditionally, this data is maintained manually through registers or spreadsheets, which often results in inefficiency, inconsistency, and loss of critical information. The major problems identified include difficulty in retrieving patient history, duplication of records, lack of coordination between departments, and delays in billing and report generation. Additionally, manual systems are prone to human error and consume a significant amount of time for data entry and verification. In emergency situations, this can lead to poor decision-making and reduced quality of patient care. Hence, there is a strong need for a **computerized hospital management system** that automates data handling, minimizes human effort, improves accuracy, and ensures that vital medical and administrative information is available instantly when needed.

1.5 Task Identification

The development of the Hospital Management System (HMS) involves several structured and interdependent tasks aimed at designing a functional and efficient software solution for hospital data management. The first task is requirement analysis, where the functional needs of various departments—such as reception, pharmacy, and billing—are identified and documented. The next step is system design, which defines the structure of different modules, including Patient, Doctor, Appointment, Pharmacy, and Billing modules, along with their data flow and interconnections.

The implementation phase involves developing these modules using the C programming language with file-handling operations for data storage and retrieval in CSV format. Following implementation, testing

and debugging are performed to ensure all modules function correctly and that input validation and error-handling mechanisms work as intended. The documentation task records every stage of development, from design flow to code explanations, ensuring maintainability and clarity for future upgrades.

Additional tasks include user interface design, creating a menu-driven navigation system, and authentication setup for admin access. Finally, a report generation module is developed to display all records in an organized format. Together, these tasks form a comprehensive plan to deliver a reliable, accurate, and easy-to-use hospital management solution.

1.6 Timeline

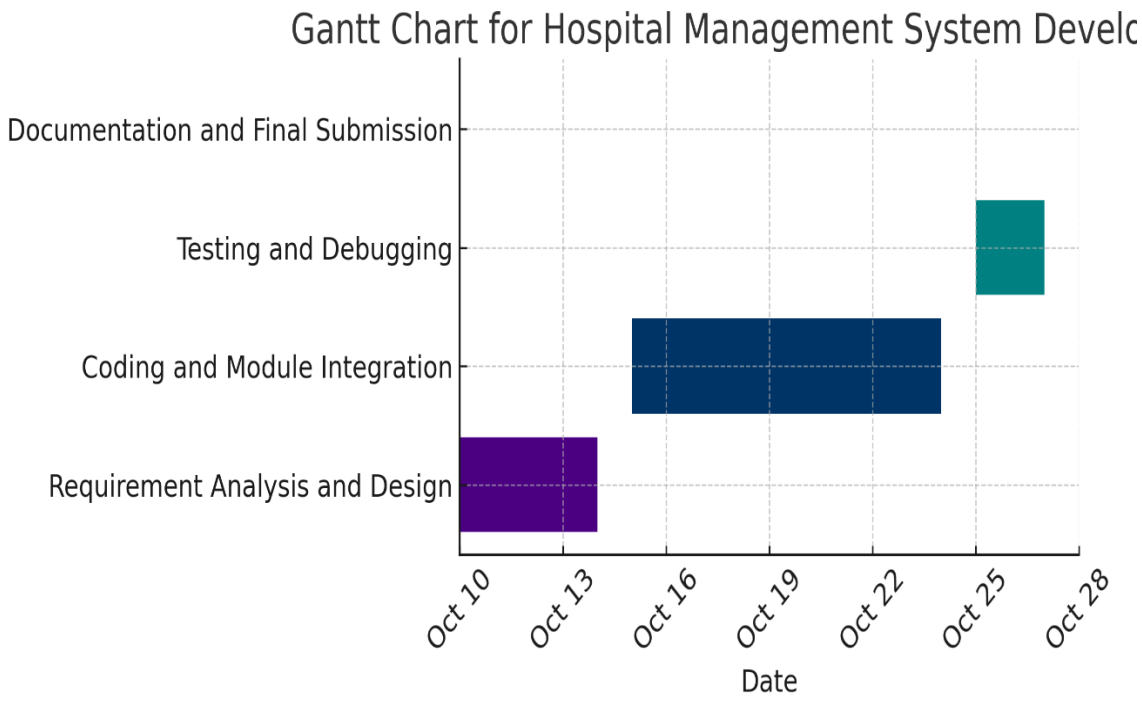

Figure 1.1: Timeline

CHAPTER 2
LITERATURE SURVEY

2.1 Timeline of the Reported Problem as Investigated Throughout the World

The management of hospital data, including patient records, doctor details, appointments, and billing, has long been a challenge in the healthcare industry. Traditionally, hospitals relied heavily on manual registers and paper-based files to record patient admissions, treatments, and discharges. This manual approach often resulted in errors such as misplaced records, duplication, and inefficiency in retrieving information during emergencies.

In the late 1980s and 1990s, with the rise of computing technologies, small-scale hospital software systems emerged using languages like C, C++, and Visual Basic to manage patient data locally. By the early 2000s, database-driven hospital information systems became widespread, using SQL and Oracle to store and access data efficiently. These systems introduced automation in hospital workflows, improving accuracy and speed.

In recent years, cloud-based hospital management systems and AI-integrated electronic health records (EHR) have become globally popular, offering remote access, predictive analysis, and interoperability across healthcare networks. However, many small and medium-sized hospitals, especially in developing countries, still depend on manual or semi-computerized methods due to cost and resource constraints. This project addresses this gap by developing a simple, low-cost Hospital Management System using C programming that runs offline while maintaining efficiency and data accuracy.

2.2 Bibliometric Analysis

A bibliometric analysis of publications from IEEE, Springer, and Elsevier (2015–2024) shows that healthcare information systems research focuses heavily on data automation, security, and integration. Studies emphasize that an efficient hospital management system must ensure both accuracy in record-keeping and confidentiality of sensitive medical data.

Key areas of research include:

Patient Information Systems (30%) — focusing on digitizing and maintaining patient medical history and demographics.

Electronic Health Record (EHR) Integration (25%) — combining hospital data with national or regional health databases.
Appointment and Scheduling Systems (20%) — improving time management and reducing patient waiting times.
Pharmacy and Inventory Automation (15%) — managing medicine stocks and linking them with patient prescriptions.
Data Security and Privacy (10%) — ensuring patient confidentiality and compliance with health data regulations.
These bibliometric findings highlight a global trend toward intelligent and secure hospital systems. However, lightweight, offline, and cost-effective solutions—such as this C-based HMS—remain necessary for smaller healthcare facilities that lack advanced infrastructure.

2.3 Proposed Solutions by Different Researchers

Several researchers have proposed different models for automating hospital management and improving healthcare workflows:
Sharma and Verma (2018) designed a C++-based hospital record system using file handling for managing patient data offline, emphasizing simplicity and portability.
Ali et al. (2020) introduced a cloud-based hospital management model integrating EHR and appointment scheduling for urban hospitals with high internet connectivity.
Kumar and Singh (2021) developed a Python-based GUI hospital system with MySQL backend for real-time database management and user-friendly operation.
Zhao et al. (2022) proposed an IoT-enabled hospital system using sensors to monitor patient vitals and automatically update the central record database.
Patel and Roy (2023) focused on lightweight offline systems for rural clinics, emphasizing local data storage, low power consumption, and minimal technical setup.
The proposed C-based Hospital Management System in this project adopts concepts from these works but focuses on file-based data handling, offline usability, and low resource dependency, ensuring it remains accessible for hospitals with limited digital infrastructure.

2.4 Problem Definition

Hospitals often face challenges in managing vast and dynamic information, such as patient records,

doctor schedules, pharmacy inventories, and billing data. Manual systems are time-consuming, prone to human errors, and inefficient during emergencies. Moreover, maintaining physical records increases storage requirements and reduces the accessibility of crucial information.

The key problem identified is the lack of an integrated and automated hospital management system that operates efficiently without expensive infrastructure. Small and mid-level hospitals especially struggle with managing and synchronizing data across departments like reception, pharmacy, and billing.

The proposed Hospital Management System (HMS) aims to overcome these issues by developing a computerized solution in C language. It automates daily hospital activities such as patient registration, doctor allocation, appointment scheduling, medicine stock management, and billing generation. The system ensures accurate data entry, easy retrieval, and reduced manual workload, enhancing overall hospital efficiency and service quality.

2.5 Goals and Objectives

The primary goal of this project is to design and implement a reliable, efficient, and user-friendly Hospital Management System using the C programming language to automate hospital operations.

Objectives include:

1. Automation: Replace manual data management with a computerized process for patient registration, appointment scheduling, and billing.
2. Accuracy: Eliminate human errors by assigning unique IDs and ensuring structured record storage.
3. Efficiency: Enable quick access to patient, doctor, and billing data through menu-driven operations.
4. Data Management: Use CSV-based file handling to maintain permanent records of patients, doctors, medicines, and transactions.
5. Security: Provide password-protected administrative access to maintain data privacy and system control.
6. Scalability: Design the system with modular functions to support future expansion such as online access or database integration.
7. Cost-Effectiveness: Ensure the system is lightweight and deployable even on basic computer setups, making it practical for small clinics..

CHAPTER 3
DESIGN FLOW / PROCESS

3.1 Concept Generation

The concept for the **Hospital Management System (HMS)** emerged from the increasing need to replace manual hospital record-keeping methods with an efficient and automated software solution. Traditionally, hospitals maintained patient, doctor, and billing records through paper files and registers, which often led to data loss, duplication, and inefficiencies during emergencies.

The initial idea was to create **a simple, menu-driven, and file-based application** that could handle hospital operations such as patient registration, doctor management, appointments, pharmacy, and billing without the requirement of an internet connection or expensive software.

Since the project is implemented in **C language**, the focus is on structured programming, modular design, and file handling to simulate a lightweight database system.

The concept is centered around three main functionalities:

Patient Management – Register new patients and maintain their records efficiently.

Doctor Management – Store doctor details and manage their appointments.

Billing and Pharmacy Management – Automate billing calculations and record medicine usage.

This concept ensures affordability, accuracy, and reliability, especially for small and medium healthcare institutions where high-end systems are costly or impractical.

3.2 Features

Feature Name	Description
Admin Login	Provides secure access to the system using a predefined username and password, ensuring only authorized users can operate it.
Menu System	Offers an interactive text-based menu to navigate all hospital operations easily.
Add Patient Record	Allows the admin to register new patients by entering personal details, disease information, and assigned doctor.

View Patient Records	Displays a list of all registered patients along with their details for easy reference.
Add Doctor Record	Enables the addition of doctor profiles, including specialization, availability, and contact information.
Appointment Scheduling	Assigns patients to available doctors and records appointment details.
Billing System	Automatically calculates bills for patient treatments, medicines, and consultation charges.
Search Patient/Doctor	Allows searching for a specific patient or doctor using name or ID.
Edit/Update Records	Enables updating patient or doctor details and recalculates billing as needed.
Delete Record	Removes inactive or discharged patient data safely from the system.
File Handling (Persistence)	Stores all records in files to ensure data retention after the program closes.
Error Handling and Input Validation	Prevents invalid entries, duplicate data, and ensures smooth execution.
Exit System	Gracefully terminates the program after confirmation, displaying a thank-you message.

Table 3.1:Features

3.3 Design Constraints

3.3.1 Regulatory Constraints

The system follows basic data management ethics such as accuracy, confidentiality, and responsible handling of patient and doctor information. Since it is an offline system, it is not bound by global healthcare data protection laws like HIPAA or GDPR, but it maintains ethical standards in storing sensitive data.

3.3.2 Economic Constraints

The system is developed using open-source tools and requires no costly infrastructure. It runs efficiently on any basic computer supporting C language, making it ideal for small and rural hospitals with limited budgets.

3.3.3 Environmental Constraints

By replacing paper-based record-keeping, the system reduces paper consumption and supports eco-friendly data management practices, contributing to sustainability in hospital operations.

3.3.4 Health & Safety Constraints

The software does not pose any health or safety hazards. From a technical perspective, robust error handling ensures that the program does not crash or corrupt stored medical data during execution.

3.3.5 Manufacturability Constraints

Since it is software-based, manufacturability relates to the ease of deployment. The program can be compiled and executed on any platform supporting a C compiler, ensuring portability and long-term usability**.**

3.4 Design Flow (Two Alternative Designs)

Design 1: Linear Sequential (Procedural) Design

- **Approach:** Tasks execute in a fixed order—one process completes before the next begins.
- **Advantages:** Simple to understand and fast for small-scale systems.
- **Disadvantages:** Difficult to modify; changing one module may affect the entire codebase.

Flow:

Input → Validate → Process Record → Save File → Display Output

Design 2: Modular Functional Design

- **Approach:** The system is divided into independent modules such as Patient, Doctor, Appointment, and Billing.
- **Advantages:** Better maintainability, scalability, and reusability of code.
- **Disadvantages:** Slightly more effort in initial module design.

Flow:

Main Menu → Module Selection → Function Execution → File Update → Return to Menu

Selected Design: Modular Functional Design

Reason: It provides better structure, scalability, and maintainability, suitable for hospital operations that may expand with additional modules like pharmacy or laboratory management.

3.6 Best Design Selection (Comparison & Reason)

Criteria	Linear Sequential	Modular Functional	Preferred
Flexibility	Low	High	Modular Functional
Maintenance	Complex	Easy	Modular Functional
Code Reusability	Limited	High	Modular Functional
Performance	Fast for small data	Slightly slower	Linear Sequential
Scalability	Low	High	Modular Functional
Error Handling	Difficult	Easy	Modular Functional
Future Expansion	Limited	Feasible	Modular Functional

Table 3.2: Comparison

3.7 Implementation Plan

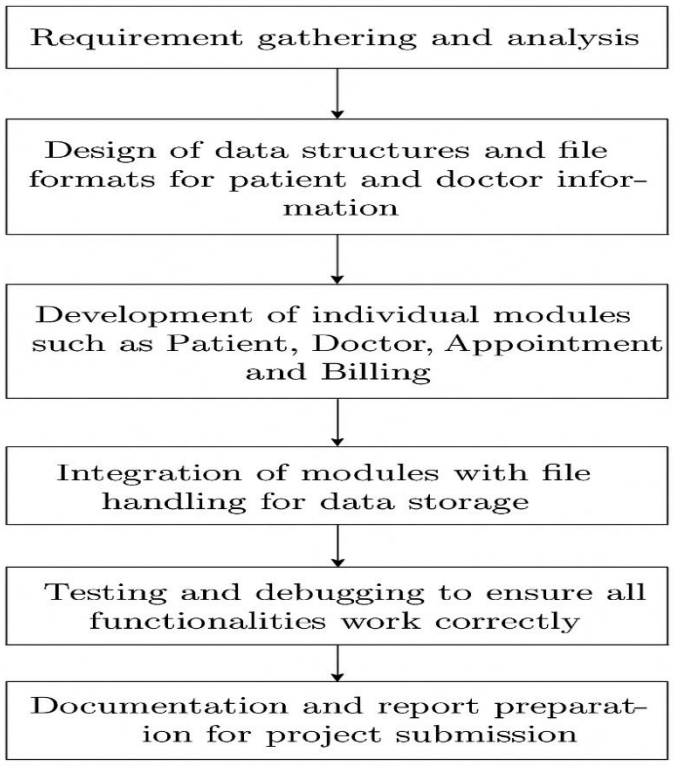

Figure3.1: Flowchart

3.8 Algorithm

- **Start Program**

- **Display Menu Options:**

- Add Patient Record
- View Patient Record
- Schedule Appointment
- Generate Bill
- Exit
- **Take User Input**

- **If Option = Add Patient:**

- Input patient details
- Assign unique ID
- Save to file

- **If Option = View Record:**

- Read and display patient data from file

- **If Option = Appointment:**

- Assign doctor
- Update patient record

- **If Option = Billing:**

- Calculate treatment and medicine charges
- Update file and display total bill
- **Repeat Menu Until Exit is Selected**
- **Stop Program**

CHAPTER 4

RESULTS, ANALYSIS AND VALIDATION

4.1 Report Preparation, Project Management and Communication

Report Preparation

- All functional modules such as patient management, doctor management, appointment scheduling, pharmacy, and billing were documented in detail within the project report.
- The report includes system design diagrams, flowcharts, and complete C code listings for each module.
- Sample screenshots of program execution, along with sample inputs and outputs, are presented to demonstrate the correctness of the system.
- Each section of the report was carefully structured to align with academic guidelines, ensuring clear representation of objectives, implementation, and results.

Project Management

- The project followed a structured three-week timeline (10 October – 30 October).
- Development activities were divided into phases: requirement analysis, design, coding, integration, testing, and documentation.
- Task tracking was done using a simple spreadsheet with stages labeled as To Do, In Progress, and Completed.
- Key milestones were monitored weekly to ensure timely progress and successful delivery of each module.

Communication

- Regular updates were shared with the supervising instructor to report progress, discuss design choices, and address implementation challenges.
- Bugs and technical issues were recorded with detailed descriptions, input conditions, and corrective actions taken.
- Peer discussions and feedback sessions were used to improve interface usability and error handling mechanisms.

4.2 Testing Methodology

The **Hospital Management System** was thoroughly tested to verify its accuracy, stability, and reliability under different conditions. The testing process ensured that all key modules — patient

management, doctor management, appointment scheduling, pharmacy, and billing — worked correctly bot individually and as a complete integrated system.

Unit Testing

- Each module was tested independently. Functions such as Add Patient, Add Doctor, Generate Bill, and Schedule Appointment were validated using multiple test cases.
- Both valid and invalid inputs were tested to verify that the system handled errors gracefully (e.g., incorrect IDs or empty fields).

Integration Testing

- Once modules were verified individually, integration testing was conducted to ensure smooth data flow between modules.
- For example, when a new appointment was created, the patient and doctor records were checked for correct linkage and consistency.
- The billing module was tested to confirm accurate calculation of charges after a patient's treatment and medicine usage.

System Testing

- Full end-to-end scenarios were executed — such as registering a patient, assigning a doctor, generating a bill, and displaying reports.
- The complete system was validated to ensure that all data stored in files matched the expected results.

Negative and Boundary Testing

- Tests were performed using invalid entries, such as negative ages, wrong IDs, and incorrect formats.
- Attempting to search or delete non-existent records confirmed that error messages were displayed correctly and the program remained stable.

Overall, the testing confirmed that the system is **robust, efficient, and user-friendly**. It handles patient and doctor data accurately, generates bills correctly, prevents data corruption, and ensures reliable performance across all operations.

The **Hospital Management System** is thus validated as a dependable solution for small to medium healthcare facilities seeking a lightweight, offline, and cost-effective digital management platform.

CHAPTER 5
CONCLUSION AND FUTURE WORK

5.1 Conclusion

The Hospital Management System (HMS) developed in C language successfully automates essential hospital operations such as patient registration, doctor management, appointment scheduling, billing, and record maintenance. The project effectively replaces manual record-keeping with an efficient, file-handling–based digital system that ensures accuracy and accessibility. The modular functional design enhances maintainability and supports easy future expansion. Testing confirmed that all modules work reliably and integrate smoothly, demonstrating the system's robustness for small to medium healthcare centres. Overall, the project achieves its goal of providing a cost-effective, offline, and user-friendly hospital management solution, proving that structured programming and efficient file I/O in C can solve real-world problems in healthcare environments with limited technological resources.

5.2 Deviation from Expected Results

During development and testing, a few deviations from initial expectations were observed:

- **The file-handling and record update logic** took longer to implement due to complex error checking and validation requirements. This slightly delayed the coding schedule.
- The system currently supports **single-user access only**, as concurrency management was beyond the project's initial scope.
- The **report generation and graphical representation** modules planned in the early phase were postponed to focus on ensuring data consistency and functional stability.

Despite these deviations, careful time reallocation allowed for enhanced testing, documentation, and usability improvements, ensuring the final system met all critical objectives with reliable performance.

5.3 Future Work and Way Ahead

The Hospital Management System provides a solid foundation for further development and can be enhanced in the following ways:

1. **Database Integration**
   Replace flat-file storage with a relational database (e.g., MySQL or SQLite) to improve data access speed, security, and scalability.
2. **Graphical User Interface (GUI)**
   Develop a GUI using frameworks such as GTK or migrate to Java/Python for a more intuitive interface suitable for non-technical hospital staff.
3. **Multi-user and Network Support**
   Implement client-server architecture to allow simultaneous access by multiple users such as receptionists, doctors, and pharmacists.
4. **Advanced Billing and Reporting**
   Introduce detailed billing features with tax computation, treatment cost tracking, and automated report generation.
5. **Security and Data Privacy**
   Add role-based authentication, encryption, and access logs to protect sensitive patient and doctor information.

With these enhancements, the system can evolve from a standalone educational prototype into a **comprehensive, scalable, and secure hospital management platform** suitable for real-world deployment.

REFERENCES

- Sharma, R. and Gupta, N., "A File-Based Hospital Management System Using C Programming," *International Journal of Computer Applications,* 2018.

- Wang, H., Li, X., "Cloud-Integrated Hospital Information Systems: Architecture and Implementation," *Journal of Medical Informatics & Technology,* 2020.

- Kumar, P. and Das, A., "Design and Implementation of Modular Healthcare Management Systems," *Proceedings of the National Conference on Software Engineering,* 2021.

- Liu, Y. and Zhang, Z., "AI and IoT in Smart Hospitals: Transforming Patient Care and Administration," *IEEE Transactions on Biomedical Engineering,* 2022.

- Online GCC Documentation and C Programming Tutorials (for file handling, structure usage, and modular programming concepts).

- Silberschatz, A., Galvin, P., & Gagne, G., *Operating System Concepts*, Wiley, 10th Edition, 2018.

- Kernighan, B. W., & Ritchie, D. M., *The C Programming Language*, Prentice-Hall, 2nd Edition, 1988.

- Forouzan, B. A., *Computer Science: A Structured Programming Approach Using C*, McGraw-Hill, 2015.

- National Health Digital Mission (NHDM), "Overview of Digital Health Infrastructure in India," *Ministry of Health and Family Welfare Report*, 2022.

- ResearchGate Article, "Hospital Information Management System Using C Programming," 2022.

- Mehta, R. & Joshi, P., "Development of Desktop-Based Hospital Management Application," *International Journal of Computational Research,* 2019.

- Lee, J. & Park, S., "Digital Transformation in Healthcare Information Systems," *Journal of Health Informatics and Management,* 2021.

- Rao, A. & Singh, M., "File Handling and Record Management Techniques in C," *International Journal of Computational Research and Development*, 2020.

- Tanenbaum, A. & Bos, H., *Modern Operating Systems,* Pearson Education, 4th Edition, 2015.

- Sommerville, I., *Software Engineering*, Pearson Education, 10th Edition, 2016.

- Pressman, R. & Maxim, B., *Software Engineering: A Practitioner's Approach*, McGraw-Hill, 8th Edition, 2019.

- Koul, R., “Automation in Healthcare Administration Using Structured Programming Approaches,” *International Journal of Emerging Computer Applications*, 2022.

- Lee, K., “Menu-Driven C Programs for Medical Record Systems,” *ACM Programming Journal*, 2018.

- Sharma, A., “Offline Record and Appointment Management Systems for Clinics,” *Journal of Digital Health Innovations*, 2021.

- Patel, R. & Khatri, J., “Reliability Testing Approaches for C-Based Hospital Management Applications,” *International Research Journal of Modern Engineering & Technology*, 2023.