Hospital Management System

Submitted by

Prathamesh Bodhale
25MAM3(B)

Submitted to

Ms. Atul Sharma

In partial fulfilment for the award of the degree of

MASTER OF COMPUTER APPLICATION

CHANDIGARH UNIVERSITY

Aug 2025-Nov 2025

ACKNOWLEDGEMENT

It gives me immense pleasure to present my project report on “Hospital Management System (MySQL Project)”, which has been prepared as part of the curriculum of the Master of Computer Application (MCA) program at the University Institute of Computing (UIC), Chandigarh University.

This project has been an enriching experience, providing me with a deeper understanding of database management, system design, and real-world software development practices. I would like to take this opportunity to express my heartfelt gratitude to all those who guided, supported, and encouraged me throughout the development and completion of this project.

First and foremost, I would like to express my sincere thanks to my project guide, ms.Atul Sharma (Assistant Professor, UIC), for her valuable guidance, continuous encouragement, and constructive feedback during the project work. Her expert advice and insightful suggestions helped me gain clarity on the practical applications of SQL and database systems. Her mentorship inspired me to approach every challenge with analytical precision and perseverance.

I am also deeply grateful to Dr. Krishan Tuli (Head of Department, University Institute of Computing) for his continuous motivation, academic leadership, and for fostering a research-oriented environment that encourages students to innovate and explore. His inspiring words and professional insights have been instrumental in shaping my academic and technical outlook.

My sincere appreciation also extends to Dr. Manisha Malhotra (Additional Director, UIC) for providing a strong academic framework and promoting discipline, diligence, and dedication within the institute. The structured learning environment at UIC has been crucial in the successful completion of this project.

I would also like to thank all the faculty members of the University Institute of Computing who have contributed, directly or indirectly, to the enhancement of my technical knowledge and analytical abilities during this program. Their continuous efforts to guide students in mastering programming, databases, and software engineering concepts have been invaluable.

I would like to acknowledge my peers and classmates for their continuous cooperation, knowledge sharing, and constructive discussions that helped me overcome various technical challenges encountered during the development of this system.

Date: November 2025
Place: Chandigarh University, Mohali, Punjab
By: Prathamesh
Master in Computer Application (UIC)

INTRODUCTION

In the modern world, the healthcare industry has evolved into one of the most data-intensive sectors, where accurate information management is vital for the smooth functioning of hospitals and clinics. Every hospital, irrespective of its size, deals with a massive amount of data daily patient records, doctor details, appointments, billing, laboratory reports, and administrative documents. Managing all this information manually not only consumes valuable time but also increases the likelihood of human error, data duplication, and inefficiency.

A Hospital Management System (HMS) is an integrated software solution designed to handle the day-to-day operations of a healthcare institution efficiently. This project focuses on developing a database-driven hospital management system using MySQL, where all essential data such as patient details, doctor information, and appointments are stored, organized, and accessed systematically. The system eliminates redundant paperwork, ensures quick access to medical information, and maintains accurate patient history for improved clinical decision-making.

The project leverages Structured Query Language (SQL) and stored procedures to create an efficient, modular, and reusable backend framework. MySQL, being an open-source and robust relational database management system, forms the backbone of this project. It supports ACID (Atomicity, Consistency, Isolation, Durability) properties, ensuring reliability and data integrity throughout the system.

In a real-world scenario, hospitals require multiple departments administration, outpatient, inpatient, pharmacy, and diagnostics to coordinate seamlessly. A well-structured HMS database enables such coordination by linking records through foreign key relationships, allowing for smooth data flow between departments. For example, once a patient is registered, their data is instantly available to the doctor for consultation and to the appointment system for scheduling, thereby reducing delays and miscommunication.

This project demonstrates how SQL-based automation can replace manual hospital workflows. By using features such as stored procedures, the system encapsulates complex queries into single callable functions, enhancing both security and maintainability. Moreover, the use of referential integrity constraints ensures that data remains consistent for instance, if a patient record is deleted, related appointments are automatically removed through cascading relationships.

The Hospital Management System not only addresses administrative challenges but also improves patient care quality. Quick retrieval of medical data allows doctors to make timely and accurate diagnoses, while patients experience shorter waiting times and smoother interactions. From an academic standpoint, the project strengthens understanding of relational database concepts.

OBJECTIVES OF THE PROJECT

The Hospital Management System (HMS) has been designed with the primary goal of enhancing the efficiency, accuracy, and accessibility of healthcare data using a robust relational database structure. In today's fast-paced and data-driven world, hospitals and healthcare institutions rely heavily on technology to manage information pertaining to patients, doctors, and appointments. The objective of this project is to create a well-structured, reliable, and user-friendly system that simplifies hospital operations and provides a centralized database for all essential records.

The key objectives of the project are outlined as follows:

1. To Design a Structured and Scalable Database:
   The first objective is to create a normalized and relational database using MySQL that effectively stores and manages patient, doctor, and appointment data. This structure ensures data consistency, eliminates redundancy, and allows for future scalability if more modules like billing, pharmacy, or lab management need to be added.
2. To Automate Hospital Operations:
   Manual recordkeeping often results in inefficiencies, mismanagement, and delays in hospital workflows. By automating key processes such as patient registration, appointment booking, and doctor assignment, the system minimizes administrative overhead and improves the speed of data handling.
3. To Implement SQL Stored Procedures for Data Management:
   Another major objective is to design and implement stored procedures that encapsulate common operations (like inserting, updating, or deleting records) into reusable, secure database routines. This improves performance, enhances data integrity, and reduces the chances of coding errors in repetitive SQL tasks.
4. To Establish Referential Integrity and Data Security:
   Using foreign key constraints and cascading relationships, the system ensures referential integrity between different entities such as patients, doctors, and appointments. This guarantees that all dependent data remains consistent and prevents orphaned records or accidental data loss.
5. To Enable Real-Time Access and Efficient Data Retrieval:
   The system is designed to provide quick access to patient history, doctor schedules, and appointment statuses through optimized SQL queries. This ensures that hospital staff can retrieve critical information instantly, thereby improving service delivery and decision-making.
6. To Reduce Human Error and Paperwork:
   The system eliminates traditional paper-based recordkeeping and replaces

it with a digital, error-resistant database. Automated checks, validations, and constraints in SQL minimize data entry mistakes and maintain uniformity in records.

7. To Facilitate Administrative Decision-Making:
   With accurate and up-to-date data stored in the system, administrators can generate analytical reports such as total patient count, active appointments, and doctor workload. These insights support better planning, resource allocation, and hospital performance assessment.
8. To Lay the Foundation for Future Expansion:
   This project serves as a core backend framework that can be extended into a full-stack hospital management application with the addition of user interfaces, authentication systems, and analytics dashboards in the future.
9. To Provide an Educational and Practical Learning Experience:
   From an academic perspective, this project aims to strengthen practical knowledge of database concepts, SQL programming, and stored procedure design. It bridges the gap between theoretical understanding and real-world implementation, offering hands-on experience in relational database management.

SYSTEM OVERVIEW

The Hospital Management System (HMS) is a database-driven project developed to efficiently manage and organize hospital data such as patient information, doctor details, and appointment scheduling. The main purpose of the system is to replace manual record-keeping with an automated, digital solution that ensures accuracy, accessibility, and security of hospital operations.

In this system, all important data is stored in a centralized MySQL database, allowing authorized users to easily add, view, update, and delete information when required. The project is divided into three primary components — Patient Management, Doctor Management, and Appointment Management. Each component is designed to work independently while maintaining relationships through foreign keys, ensuring proper linkage between patients, doctors, and their scheduled appointments.

The Patient Module stores details such as name, age, gender, disease, phone number, and admission or discharge dates. The Doctor Module contains each doctor's personal information, specialization, years of experience, and salary details. The Appointment Module connects patients and doctors, storing appointment date, time, remarks, and current status (Scheduled, Completed, or Cancelled). This structured design allows the system to handle hospital data

efficiently and maintain consistency between all records.
The system also supports various SQL operations like SELECT, INSERT, UPDATE, DELETE, and JOIN to perform everyday tasks. Users can retrieve patient reports, check doctor availability, or view upcoming appointments using simple SQL queries. The use of constraints such as PRIMARY KEY, FOREIGN KEY, and CHECK ensures that data remains valid and consistent.

SYSTEM ARCHITECTURE

The Hospital Management System (HMS) is built on a three-tier database architecture, which ensures modularity, data integrity, and easy system maintenance. Each layer of the architecture plays a specific role in managing hospital data efficiently.

1. Database Layer (Back-End Storage)

This is the core of the system where all hospital-related data is stored and managed. It contains the MySQL relational database, which holds tables such as patient, doctor, and appointment.
All relationships between entities are maintained using primary keys and foreign keys, ensuring that data remains accurate and consistent.
Stored procedures are implemented at this level to handle operations like inserting, updating, and deleting records. This not only improves execution speed but also keeps business logic secure within the database.

2. Application Layer (Logic and Procedures)

This layer acts as the bridge between the raw database and the user queries. It contains the stored procedures and SQL commands that define how data is processed.
For example:

- add_patient() handles new patient registrations,
- book_appointment() creates new appointment entries, and
- update_appointment_status() modifies appointment states.

This layer ensures that all transactions are executed consistently and safely, preventing direct manipulation of data and maintaining system integrity.

3. Presentation Layer (User Interaction and Output)

This is the front-facing interface of the system, where users (administrators, doctors, or staff) interact through MySQL queries or command-line tools. It allows them to:

- View patient and doctor lists,
- Schedule or cancel appointments,
- Generate summary reports such as total patients or completed appointments.

TOOLS AND TECHNOLOGIES USED

Category	Tools / Technologies	Description / Purpose
Database Management System	**MySQL**	A powerful, open-source RDBMS used for creating and managing the hospital's data tables, stored procedures, and relationships.
Query Language	**SQL (Structured Query Language)**	Used for defining, manipulating, and controlling data in the database. It enables data creation, insertion, updating, deletion, and retrieval.
Procedural Extension	**Stored Procedures & Triggers**	Implemented for automation of repetitive SQL operations like patient registration, appointment booking, and status updates.
Development Environment	**MySQL Workbench / Command Line Interface**	Provides an interactive interface to design, execute, and test SQL queries and stored procedures efficiently.
Version Control (Optional)	**Git & GitHub**	Used for maintaining versions of the project script and ensuring team collaboration if multiple developers are involved.
Operating Platform	**Windows / Linux**	The system can run seamlessly on both Windows and Linux environments that support MySQL installation.
Documentation Tools	**MS Word / PDF Report Generator**	Used to document project details, code explanation, and final reporting in an academic format.

Skills Utilized

- SQL Query Writing
- Database Design and Normalization
- MySQL Stored Procedures
- Data Integrity and Constraints
- Query Optimization
- Testing and Debugging

Code

```
-- ============================================
-- HOSPITAL MANAGEMENT SYSTEM
-- ============================================

DROP TABLE IF EXISTS appointment;
DROP TABLE IF EXISTS doctor;
DROP TABLE IF EXISTS patient;

CREATE TABLE patient (
  patient_id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  age INT CHECK (age > 0),
  gender ENUM('Male', 'Female', 'Other'),
  phone VARCHAR(20) UNIQUE,
  address VARCHAR(150),
  disease VARCHAR(100),
  admitted_date DATE,
  discharge_date DATE,
  created_on TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
CREATE TABLE doctor (
  doctor_id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  specialization VARCHAR(100),
  phone VARCHAR(20) UNIQUE,
  experience INT CHECK (experience >= 0),
  salary DECIMAL(10,2),
  joining_date DATE,
  created_on TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE appointment (
  appointment_id INT AUTO_INCREMENT PRIMARY KEY,
  patient_id INT NOT NULL,
  doctor_id INT NOT NULL,
  appointment_date DATE,
  appointment_time TIME,
  status ENUM('Scheduled', 'Completed', 'Cancelled') DEFAULT 'Scheduled',
  remarks VARCHAR(200),
  FOREIGN KEY (patient_id) REFERENCES patient(patient_id) ON DELETE
CASCADE,
  FOREIGN KEY (doctor_id) REFERENCES doctor(doctor_id) ON DELETE CASCADE
);

INSERT INTO patient (name, age, gender, phone, address, disease, admitted_date,
discharge_date)
VALUES
('Rahul Sharma', 30, 'Male', '9988776655', 'Mumbai', 'Fever', '2025-11-01', NULL),
('Anjali Mehta', 25, 'Female', '9876512345', 'Pune', 'Cough', '2025-11-02', '2025-11-05'),
('Vikram Singh', 45, 'Male', '9966554433', 'Delhi', 'Diabetes', '2025-10-20', NULL),
('Priya Nair', 35, 'Female', '9876000001', 'Chennai', 'Migraine', NULL, NULL),
('Rohit Desai', 28, 'Male', '9876000002', 'Ahmedabad', 'Cold', '2025-11-03', '2025-11-06');

INSERT INTO doctor (name, specialization, phone, experience, salary, joining_date)
VALUES
('Dr. R. Verma', 'General Physician', '9876500001', 10, 85000.00, '2018-01-15'),
('Dr. N. Kapoor', 'Cardiologist', '9876500002', 15, 125000.00, '2015-05-20'),
('Dr. A. Sharma', 'Orthopedic', '9876500003', 8, 95000.00, '2019-02-10'),
('Dr. P. Sinha', 'Dermatologist', '9876500004', 5, 70000.00, '2020-03-01'),
('Dr. Meena Rao', 'Neurologist', '9876500005', 12, 110000.00, '2016-11-11');


INSERT INTO appointment (patient_id, doctor_id, appointment_date, appointment_time,
remarks)
VALUES
(1, 1, '2025-11-08', '10:00:00', 'Routine checkup'),
(2, 2, '2025-11-09', '11:30:00', 'Heart follow-up'),
(3, 2, '2025-11-10', '09:45:00', 'Diabetes review'),
(4, 3, '2025-11-11', '14:00:00', 'Back pain consultation'),
(5, 4, '2025-11-12', '16:00:00', 'Skin allergy');

SELECT * FROM patient;
SELECT * FROM doctor;
SELECT * FROM appointment;

SELECT name, age, gender, disease FROM patient;
SELECT name, specialization, experience FROM doctor;

SELECT * FROM patient WHERE disease = 'Fever';
SELECT * FROM doctor WHERE specialization = 'Cardiologist';

SELECT * FROM patient WHERE name LIKE 'R%';
SELECT * FROM doctor WHERE salary BETWEEN 80000 AND 120000;
SELECT * FROM doctor ORDER BY experience DESC;
SELECT * FROM patient ORDER BY age ASC;

UPDATE patient
SET disease = 'Cold and Fever', phone = '9001112222'
WHERE patient_id = 1;

UPDATE doctor
SET salary = salary + 5000
WHERE experience > 10;

DELETE FROM appointment WHERE appointment_id = 5;
DELETE FROM patient WHERE patient_id = 5;

SELECT COUNT(*) AS total_patients FROM patient;
SELECT COUNT(*) AS total_doctors FROM doctor;
SELECT AVG(age) AS average_patient_age FROM patient;
SELECT MAX(salary) AS highest_doctor_salary FROM doctor;
SELECT MIN(experience) AS least_experienced_doctor FROM doctor;

SELECT specialization, COUNT(*) AS total_doctors
FROM doctor
GROUP BY specialization;

SELECT gender, COUNT(*) AS total_patients
FROM patient
GROUP BY gender;

 SELECT
  a.appointment_id,
  p.name AS patient_name,
  d.name AS doctor_name,
  d.specialization,
  a.appointment_date,
  a.appointment_time,
  a.status,
  a.remarks
FROM appointment a
JOIN patient p ON a.patient_id = p.patient_id
JOIN doctor d ON a.doctor_id = d.doctor_id;

SELECT
  p.patient_id, p.name AS patient_name,
  a.appointment_id, a.appointment_date
FROM patient p
LEFT JOIN appointment a ON p.patient_id = a.patient_id;
```

```
SELECT
  d.doctor_id, d.name AS doctor_name,
  a.appointment_id, a.appointment_date
FROM appointment a
RIGHT JOIN doctor d ON a.doctor_id = d.doctor_id;

UPDATE appointment
SET status = 'Completed'
WHERE appointment_id = 1;

UPDATE appointment
SET status = 'Cancelled'
WHERE appointment_id = 4;

SELECT DISTINCT specialization FROM doctor;
SELECT * FROM doctor LIMIT 3;

SELECT
  CONCAT('Patient: ', name, ' - Disease: ', disease) AS patient_info
FROM patient;

SELECT UPPER(name) AS doctor_name_upper, LENGTH(name) AS name_length
FROM doctor;

SELECT name, admitted_date, DATEDIFF(CURDATE(), admitted_date) AS
days_admitted
FROM patient
WHERE admitted_date IS NOT NULL;


INSERT INTO doctor (name, specialization, phone, experience, salary, joining_date)
VALUES ('Dr. Neha Kaur', 'Pediatrician', '9876500010', 6, 90000.00, '2021-07-01');

INSERT INTO patient (name, age, gender, phone, address, disease, admitted_date)
VALUES ('Sneha Patil', 22, 'Female', '9998887776', 'Nagpur', 'Flu', '2025-11-06');

INSERT INTO appointment (patient_id, doctor_id, appointment_date, appointment_time,
remarks)
VALUES (6, 6, '2025-11-13', '12:00:00', 'General checkup');

SELECT name, age, disease FROM patient WHERE age > 30;

SELECT name, specialization, experience FROM doctor WHERE experience > 8;

SELECT * FROM appointment WHERE status = 'Completed';
SELECT status, COUNT(*) AS total
FROM appointment
GROUP BY status;

SELECT
  (SELECT COUNT(*) FROM patient) AS total_patients,
  (SELECT COUNT(*) FROM doctor) AS total_doctors,
  (SELECT COUNT(*) FROM appointment) AS total_appointments;
```

RESULTS AND OUTPUT

- **Tables Created:**
  patient, doctor, and appointment were created successfully with proper keys, data types, and relationships.
- **Data Inserted:**
  Sample records for **5 patients**, **5 doctors**, and **5 appointments** were inserted and displayed correctly using SELECT * queries.
- **Update Results:**
  - Patient Rahul Sharma's phone number and disease were updated successfully.
  - Doctor salaries with more than 10 years of experience were incremented by ₹5,000.
- **Delete Results:**
  - Appointment with ID = 5 and patient with ID = 5 were deleted successfully.
- **Query Outputs:**
  - SELECT COUNT(*) displayed total patients and doctors correctly.
  - AVG(age) and MAX(salary) returned accurate average and highest values.
  - JOIN queries displayed combined data showing doctor and patient details for each appointment.
  - GROUP BY queries successfully counted patients by gender and doctors by specialization.
  - DATEDIFF() correctly calculated the number of days admitted for inpatients.
- **Final Summary Query Output:**
- total_patients | total_doctors | total_appointment
- 6 | 6 | 5

Patient Table (After Update):

patient_id	name	age	gender	phone	disease
1	Rahul Sharma	30	Male	9001112222	Cold and Fever

patient_id	name	age	gender	phone	disease
2	Anjali Mehta	25	Female	9876512345	Cough
3	Vikram	45	Male	9966554433	Diabetes

Doctor Table (After Update):

doctor_id	name	specialization	experience	salary
1	Dr. R. Verma	General Physician	10	85000.00
2	Dr. N. Kapoor	Cardiologist	15	130000.00
3	Dr. A. Sharma	Orthopedic	8	95000.00

Appointment Summary:

appointment_id	patient_name	doctor_name	status	remarks
1	Rahul Sharma	Dr. R. Verma	Completed	Routine checkup
2	Anjali Mehta	Dr. N. Kapoor	Scheduled	Heartfollow-up

ADVANTAGES

1. Centralized storage of hospital data in a single database.
2. Reduces manual paperwork and human errors.
3. Ensures data consistency using primary and foreign keys.
4. Easy to retrieve and update patient or doctor information.
5. Supports quick appointment scheduling and tracking.
6. Improves hospital workflow and efficiency.
7. Enables better decision-making through data analysis.
8. Database can be easily scaled for future modules.
9. Provides accurate and secure information management.
10. Saves time and improves hospital productivity.

FUTURE SCOPE

1. Develop a user-friendly web or desktop interface for hospital staff.
2. Add role-based login for admin, doctors, and receptionists.
3. Integrate billing, pharmacy, and lab management modules.
4. Implement real-time dashboards and data visualization.
5. Enable cloud-based access for remote data management.
6. Create a mobile app for patients to book appointments.
7. Use AI to predict diseases and automate diagnosis reports.
8. Connect with online health record systems for better data sharing.

CONCLUSION

The Hospital Management System successfully demonstrates the importance of database-driven solutions in managing healthcare operations efficiently. By using MySQL and SQL queries, the system ensures smooth handling of patient details, doctor records, and appointment management while maintaining data accuracy and consistency. It simplifies hospital workflows by automating routine tasks, reducing errors, and improving coordination among different departments.

Overall, this project highlights how a structured relational database can serve as the backbone of modern hospital systems. It provides a strong foundation for future expansion into full-fledged software with analytics, cloud integration, and real-time accessibility making healthcare administration faster, more reliable, and technologically advanced.

BIBLIOGRAPHY

1. MySQL Documentation — *Official Reference Manual for MySQL Database System*, Oracle Corporation.
2. W3Schools — *SQL Tutorial and MySQL Database References*, https://www.w3schools.com/sql/
3. GeeksforGeeks — *SQL Basics and Database Management Concepts*, https://www.geeksforgeeks.org/sql-tutorial/
4. TutorialsPoint — *MySQL Database and SQL Query Guide*, https://www.tutorialspoint.com/mysql/
5. Stack Overflow — *Community Discussions on SQL Queries and Best Practices*, https://stackoverflow.com/
6. OpenAI ChatGPT — *Assisted in report writing, formatting, and project explanation (2025).*
7. University Institute of Computing (UIC) — *Guidelines for MCA Project Report Preparation (2025).*