**Project Documentation**

****

**Submitted by:**

MUHAMMAD HAMZA (2024-CS-678)

SHAHZAIB TARIQ (2024-CS-656)

AMMAR JAMIL (2024-CS-683)

**Submitted To:**

MIS. RIDA SAEED

Department of Computer Science

**University of Engineering and Technology,**

**New Campus, Lahore**

**Table of Contents**

[**1** **Introduction** 4](#_Toc196828284)

[**2** **Backend Architecture** 4](#_Toc196828285)

[**2.1** **Overview** 4](#_Toc196828286)

[**2.2** **Technologies Used** 4](#_Toc196828287)

[**3** **Database Schema** 4](#_Toc196828288)

[**3.1** **Entities and Attributes** 4](#_Toc196828289)

[**1** **Department** 4](#_Toc196828290)

[**2** **Doctor** 4](#_Toc196828291)

[**3** **Patient** 5](#_Toc196828292)

[**4** **Staff** 5](#_Toc196828293)

[**5** **Appointment** 5](#_Toc196828294)

[**6** **Medical Records** 6](#_Toc196828295)

[**7** **Pharmacy Inventory** 6](#_Toc196828296)

[**8** **Prescription** 6](#_Toc196828297)

[**9** **Prescription Detail** 7](#_Toc196828298)

[**10** **Test Examination** 7](#_Toc196828299)

[**11** **Room** 7](#_Toc196828300)

[**12** **Payment** 8](#_Toc196828301)

[**3.2** **Relationships** 8](#_Toc196828302)

[**3.3** **Tables** 8](#_Toc196828303)

[**4** **Functionality and Features** 9](#_Toc196828304)

[**4.1** **Department Management** 9](#_Toc196828305)

[**4.2** **Doctor Management** 9](#_Toc196828306)

[**4.3** **Patient Management** 9](#_Toc196828307)

[**4.4** **Appointment Scheduling** 10](#_Toc196828308)

[**4.5** **Medical Record Management** 10](#_Toc196828309)

[**4.6** **Inventory Management** 10](#_Toc196828310)

[**4.7** **Prescription Management** 10](#_Toc196828311)

[**4.8** **Payment Management** 10](#_Toc196828312)

[**5** **Views and Triggers** 10](#_Toc196828313)

[**5.1** **Views** 10](#_Toc196828314)

[**5.2** **Triggers** 11](#_Toc196828315)

[**6** **Normalization** 11](#_Toc196828316)

[**6.1** **First Normal Form (1NF)** 11](#_Toc196828317)

[**6.2** **Second Normal Form (2NF)** 12](#_Toc196828318)

[**6.3** **Third Normal Form (3NF)** 13](#_Toc196828319)

[**7** **ERD Diagram** 15](#_Toc196828320)

[**8** **Enhanced ERD** 16](#_Toc196828321)

[**9** **Deployment** 17](#_Toc196828322)

[**10** **Conclusion** 17](#_Toc196828323)

**Table of Figures**

[Figure 1-ERD 12](#_Toc196635773)

[Figure 2-EERD 13](#_Toc196635774)

# **Introduction**

This documentation provides an overview of the backend architecture and database design for the Hospital Management System. It details the functionalities implemented, database schema, and key features of the backend system.

# **Backend Architecture**

## **Overview**

The backend of the Hospital Management System is built using SQL Server, with a well-structured relational database named **Hospital**. It includes normalized tables for key entities such as **Departments, Doctors, Patients, Staff, Appointments, Medical Records, and Payments**. Relationships are established using **foreign keys** to maintain data integrity and enable efficient querying. The schema supports core hospital operations like patient registration, doctor assignments, prescription management, room allocation, and billing.

## **Technologies Used**

* **Database:** SQL Server

# **Database Schema**

## **Entities and Attributes**

### **Department**

* + Department ID (Primary Key)
  + Department Name
  + Location
  + Contact Number

### **Doctor**

* Doctor ID (Primary Key)
* Name
* Specialization
* Contact Number
* Email
* Department ID (Foreign Key)
* License Number
* Years Of Experience

### **Patient**

* Patient ID (Primary Key)
* Name
* Age
* Gender
* Address
* Blood Type
* Registration Date
* Contact Number

### **Staff**

* Staff ID (Primary Key)
* Name
* Role
* Contact Number
* Email
* Department ID (Foreign Key)
* Shift Timings
* Salary

### **Appointment**

* Appointment ID (Primary Key)
* Patient ID (Foreign Key)
* Doctor ID (Foreign Key)
* Appointment Date Time
* Status
* Reason For Visit

### **Medical Records**

* Record ID (Primary Key)
* Patient ID (Foreign Key)
* Doctor ID (Foreign Key)
* Visit Date
* Diagnosis
* Treatment Notes

### **Pharmacy Inventory**

* Drug ID (Primary Key)
* Name
* Manufacturer
* Category
* Expiry Date
* Stock Quantity
* Price Per Unit

### **Prescription**

* Prescription ID (Primary Key)
* Patient ID (Foreign Key)
* Doctor ID (Foreign Key)
* Issue Date
* Expiry Date
* Status
* Instructions

### **Prescription Detail**

* Prescription Detail ID (Primary Key)
* Prescription ID (Foreign Key)
* Drug ID (Foreign Key)
* Dosage
* Frequency
* Duration
* Special Instructions

### **Test Examination**

* Test ID (Primary Key)
* Patient ID (Foreign Key)
* Doctor ID (Foreign Key)
* Test Type
* Test Date
* Results
* Status
* Technician ID (Foreign Key)
* Cost

### **Room**

* Room ID (Primary Key)
* Room Number
* Room Type
* Status
* Current Patient ID (Foreign Key)
* Department ID (Foreign Key)
* Rate Per Day

### **Payment**

* Payment ID (Primary Key)
* Patient ID (Foreign Key)
* Amount
* Payment Date
* Payment Method
* Payment Status
* Invoice Number
* Billing Staff ID (Foreign Key)

## **Relationships**

* **One-to-Many**: Departments to Doctors, Patients to Appointments, Doctors to Appointments, etc.
* **Many-to-One**: Appointments to Patients, Medical Records to Patients, etc.
* **Many-to-Many**: Not applicable in this schema.

## **Tables**

* **Departments**: Stores department information such as department name, location, contact number, and head doctor.
* **Doctors**: Contains doctor details including name, specialization, contact number, email, department ID, license number, and years of experience.
* **Patients**: Records patient information such as name, age, gender, address, blood type, contact number, and registration date.
* **Staff**: Manages hospital staff details including name, role, contact number, email, department ID, shift timings, and salary.
* **Appointments**: Tracks patient appointments including patient ID, doctor ID, appointment date and time, and status.
* **Medical Records**: Stores patient medical history including diagnosis, treatment, record date, and notes.
* **Billing**: Records billing information such as patient ID, total amount, payment status, and billing date.
* **Prescriptions**: Contains prescriptions issued by doctors including patient ID, doctor ID, prescription date, and notes.
* **Medications**: Lists available medications including medication name, manufacturer, and price.
* **Prescription Details**: Links prescriptions to medications, specifying dosage and duration.

# **Functionality and Features**

## **Department Management**

Create, read, update, and delete departments. This module helps organize different medical specialties within the hospital. It ensures each department is properly categorized and accessible for assignments.

## **Doctor Management**

Manage doctor profiles, including specialization and availability. Details like qualifications, contact information, and department assignment are maintained. Doctors can update their schedules and leave details through this module.

## **Patient Management**

Register new patients and manage patient records. Includes patient demographics, medical history, and contact information. Helps in tracking long-term patient interactions and visits.

## **Appointment Scheduling**

Schedule and manage appointments between patients and doctors. Prevents double-booking and helps doctors plan their day efficiently. Patients can be notified about upcoming or rescheduled appointments.

## **Medical Record Management**

Maintain a record of patient visits and treatments. Stores diagnosis, treatment plans, test results, and progress notes. Provides quick access to patient history during future visits.

## **Inventory Management**

Track pharmacy inventory, including stock levels and expiry dates. Alerts for low stock and soon-to-expire medicines ensure smooth operations. Helps maintain availability of critical supplies and medications.

## **Prescription Management**

Manage prescriptions and their details. Doctors can digitally prescribe medication linked to patient records. Improves clarity and reduces the chance of medication errors.

## **Payment Management**

Handle payments and billing for services rendered. Generates invoices for consultations, tests, and treatments. Supports different payment modes like cash, card, and insurance.

# **Views and Triggers**

Views and Triggers are used in backend of this project and details are as follows:

## **Views**

* Contains basic information about doctors such as names, specialties, and assigned departments.
* Displays a list of all upcoming appointments scheduled between patients and doctors.
* Provides contact details of patients including phone numbers, emails, and addresses.
* Allows authorized users to update and manage patient contact information easily.
* One Patients Visit in Doctor Of How Many Times
* Lists all appointments scheduled specifically for the current day with timing.
* Summarizes prescription details issued to patients, including medicines and dosages

## **Triggers**

* Updates room status to 'Occupied' when a patient is assigned to a room.
* Automatically sets room status to 'Available' when a patient is discharged.
* Marks an appointment as 'Completed' after a related medical record is added.
* Reduces the stock quantity of a drug in inventory after it's prescribed.
* Prints a message when a doctor’s salary is updated.
* Saves a deleted patient’s data into an audit table for record-keeping.

# **Normalization**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **DoctorI D** | **Name** | **Specializations** | **Contact Numbers** | **Department Name** | **License Numbers** | **Salary** |
| 1 | Dr. Ahsan Khan | Cardiology, Internal Med. | 03001234567, 0423456789 | Cardiology | LIC123, LIC456 | |  | | --- | | 200000 |  |  | | --- | |  | |
| 2 | Dr. Fatima Raza | Dermatology | |  | | --- | |  |  |  | | --- | | 0300765432 | | Dermatology | LIC789 | 180000 |

**UNNormalized Table :**

## **First Normal Form (1NF)**

* All tables have **atomic values**; no multi-valued attributes exist.
* Each record is **uniquely identified** by a **primary key**.
* Example:

In the Doctor Table, Suppose originally, you had a Doctors table like this (not normalized:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Doctor ID** | **Name** | **Specializations** | **Contact Numbers** | **Department Name** | **License Numbers** | **Salary** |
| 1 | Dr. Ahsan Khan | Cardiology, | 0300-1234567 | Cardiology | LIC123 | |  | | --- | | 200000 |  |  | | --- | |  | |
| 1 | Dr. Ahsan Khan | Internal Med. | 042-3456789 | Cardiology | LIC456 | 20000 |
| 2 | Dr. Fatima Raza | Dermatology | |  | | --- | |  |  |  | | --- | | 0300-76543 | | Dermatology | LIC789 | 180000 |

**Doctor\_1NF :**

## **Second Normal Form (2NF)**

* The database is fully functionally dependent on the whole primary key.
* No partial dependencies exist.
* Composite primary keys (if any) are properly broken down into separate tables.
* Example:

Doctor table will be 1st normalized and remove of a double value in a cell

**Doctor\_2NF :**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Doctor ID** | **Name** | **Department Name** | **License Numbers** | **Years of Experiences** | **Salary** |
| 1 | Dr. Ahsan Khan | Cardiology | LIC123 | 10 | |  | | --- | | 200000 |  |  | | --- | |  | |
| 2 | Dr. Fatima Raza | Dermatology | LIC789 | 7 | 180000 |

**Doctor Specializations Table;**

|  |  |  |
| --- | --- | --- |
| **Specialization ID** | **Doctor ID** | **Specialization** |
| 1 | 1 | Cardiology |
| 2 | 1 | Internal Medicine |
| 3 | 2 | Dermatology |

**Doctor Contact Numbers Table :**

|  |  |  |
| --- | --- | --- |
| **Contact ID** | **Doctor ID** | **Contact Number** |
| 1 | 1 | 03001234567 |
| 2 | 1 | 03001234567 |
| 3 | 2 | 03007654321 |

## **Third Normal Form (3NF)**

* No transitive dependencies exist.
* Non-key attributes depend only on the primary key.
* Repeated information like doctor’s specialization or patient contact is stored only once in their respective tables.
* Example:

In this table Doctor id foreign key and department id primary and table will be a

3rd Normalized form.

**Department Table:**

|  |  |
| --- | --- |
| **Department ID** | **Department Name** |
| 1 | Cardiology |
| 2 | Dermatology |

**Doctor\_3NF :**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Doctor ID** | **Name** | **Department ID** | **License Numbers** | **Years of Experiences** | **Salary** |
| 1 | Dr. Ahsan Khan | 1 | LIC123 | 10 | |  | | --- | | 200000 |  |  | | --- | |  | |
| 2 | Dr. Fatima Raza | 2 | LIC789 | 7 | 180000 |

# **ERD Diagram**

A diagram of a company

AI-generated content may be incorrect.

Figure 1-ERD

# **Enhanced ERD**

A diagram of a company

AI-generated content may be incorrect.

Figure -EERD

# **Deployment**

The deployment of the Hospital Management System database begins with setting up the environment by installing Microsoft SQL Server and SQL Server Management Studio (SSMS). A new database named Hospital is created, and the provided SQL script is executed to generate all necessary tables, relationships, and constraints. After execution, the database structure is reviewed to ensure that all primary keys and foreign keys are correctly implemented. Initial data is inserted into essential tables such as Departments, Doctors, and Patients to prepare the system for functional testing. The database can be hosted locally for development purposes or deployed on cloud platforms like Microsoft Azure or Amazon RDS for broader accessibility and scalability. Basic security measures, such as authentication, role-based access control, and encryption of sensitive information, are configured to protect the data. Finally, a regular backup schedule is set up to ensure data integrity and disaster recovery. This deployment approach ensures that the database is reliable, scalable, and secure, ready to support the hospital’s operational needs effectively.

# **Conclusion**

The Hospital Management System database has been carefully designed and normalized to ensure efficient data storage, consistency, and easy maintenance. By organizing the data into well-structured tables and applying the principles of first, second, and third normal forms, redundancy has been minimized and data integrity is maintained. The deployment process ensures the database is ready for real-world use, with necessary security measures and scalability options in place. This project not only meets the immediate operational needs of a hospital but also lays a strong foundation for future expansions, such as integrating pharmacy management, insurance handling, or advanced reporting systems. With a reliable, secure, and optimized database structure, the hospital can streamline its processes, improve patient care, and support decision-making effectively.

Future enhancements will focus on automating database management, improving performance through indexing, and implementing advanced analytics for better healthcare outcomes.