**Hospital Management System**

***Dissertation Submitted in Partial fulfilment of the Requirement for the Award of the Diploma of***

**Computer Engineering**

**VIth Semester**

**Jan - May, 2023**

**Project Guide: Submitted By:** Nidhi ChawanSuraj kumar Gupta

2013051062

**DSEU Rajokri Campus**

**Department of Training and Technical Education, Delhi 2023**

**DECLARATION**

This project, titled "Hospital Management System" was undertaken to develop a software application that can manage patient records, appointments, and medical information for a hospital or healthcare facility. The scope of the project includes designing a user-friendly interface for doctors, nurses, and administrative staff, implementing secure data storage and retrieval mechanisms, and creating a robust reporting system for analyzing patient data.

The project was completed using Java programming language and various open-source libraries and frameworks. The code was developed using an object-oriented approach, following industry-standard design patterns and best practices.

The goal of this project is to provide a reliable, efficient, and scalable solution for hospitals and healthcare facilities to manage their operations and improve patient outcomes. This project was completed as a part of [name of the program/course/project] and is the original work of Suraj Kumar Gupta. Any references or sources used in this project have been properly cited and acknowledged.

Suraj Kumar Gupta

**CERTIFICATE FROM GUIDE**

It is to certify that the project entitled Hospital Management System, submitted by Mr. Suraj Kumar Gupta to the DSEU Rajokri Campus, Department of Training and Technical Education, Delhi, has been completed under my supervision and the work is carried out and presented in a manner required for its acceptance to Diploma in Computer Engineering VIth semester.

Project Guide

Signature: ………………………..

Name: ………………………..

Date: ………………………..

**CERTIFICATE FROM COMPANY**

**Django Internship Certificate**

****

**Front-End Internship Certificate**

****

**ACKNOWLEDGEMENT**

We would like to express our sincere gratitude to Nidhi Chawla for their invaluable support and guidance during the development of this Hospital Management System. Their expertise and insights were instrumental in shaping the design and functionality of the software, and we could not have completed this project without their help.

We would also like to thank my friend Vikas for their constant encouragement, feedback, and constructive criticism throughout the project. Their mentorship and expertise helped us overcome many challenges and improve the quality of our work.

We are also grateful to the healthcare professionals who generously gave their time and insights to help us understand the needs and challenges of the hospital management industry. Their feedback and suggestions were invaluable in shaping the features and functionality of our software.

Finally, we would like to acknowledge the numerous open-source libraries, frameworks, and tools that we used in the development of this project. Without the contributions of the developers who created and maintained these resources, this project would not have been possible.

Suraj Kumar Gupta

**ABSTRACT**

The Hospital Management System is a software application that was developed to manage patient records, appointments, and medical information for hospitals and healthcare facilities. The system provides a user-friendly interface for doctors, nurses, and administrative staff to access and manage patient data securely and efficiently.

The project was developed using Java programming language and various open-source libraries and frameworks, including JavaFX for the user interface and Hibernate for data storage and retrieval. The software was designed using an object-oriented approach, following industry-standard design patterns and best practices.

The system's features include patient registration, appointment scheduling, medical record management, prescription management, and reporting. The software is highly customizable, with options for configuring user roles and permissions, setting up custom fields, and integrating with other healthcare systems.

The project was completed as part Hospital Management System, and is the original work of Suraj Kumar Gupta. The software was tested thoroughly to ensure its reliability and efficiency, and feedback was collected from healthcare professionals to improve its usability and functionality.

Overall, the Hospital Management System provides a comprehensive solution for hospitals and healthcare facilities to manage their operations and improve patient outcomes. The software is scalable, secure, and customizable, and can be adapted to the unique needs and challenges of any healthcare organization.

Suraj Kumar Gupta

Contents

[CHAPTER 1 8](#_Toc134723316)

[CHAPTER 2 14](#_Toc134723335)

[CHAPTER 3 19](#_Toc134723337)

[CHAPTER 4 21](#_Toc134723339)

[CHAPTER 5 39](#_Toc134723340)

[CHAPTER 6 41](#_Toc134723341)

[CHAPTER 7 43](#_Toc134723342)

[CHAPTER 8 44](#_Toc134723351)

## CHAPTER 1

### Introduction

### Introduction

The Hospital Management System is a software application designed to streamline the management of patient records, appointments, and medical information for hospitals and healthcare facilities. The system provides a user-friendly interface for doctors, nurses, and administrative staff to access and manage patient data securely and efficiently. The Hospital Management System is intended to improve the quality of patient care by providing timely access to accurate medical information and reducing the administrative burden on healthcare professionals.

The goal of this project was to develop a comprehensive solution for hospitals and healthcare facilities to manage their operations and improve patient outcomes. The project was completed using Java programming language and various open-source libraries and frameworks, including JavaFX for the user interface and Hibernate for data storage and retrieval. The software was designed using an object-oriented approach, following industry-standard design patterns and best practices.

The Hospital Management System provides a range of features, including patient registration, appointment scheduling, medical record management, prescription management, and reporting. The software is highly customizable, with options for configuring user roles and permissions, setting up custom fields, and integrating with other healthcare systems.

Overall, the Hospital Management System represents a significant advancement in healthcare technology, providing hospitals and healthcare facilities with a reliable, efficient, and scalable solution for managing their operations and improving patient outcomes.

### Problem Statement

Hospitals and healthcare facilities are faced with numerous challenges in managing their operations and providing quality care to their patients. One of the primary challenges is the need to manage vast amounts of patient data securely and efficiently. With increasing demands on healthcare services, healthcare professionals need a reliable and efficient system to manage patient data and ensure that patients receive the best possible care.

Current methods of managing patient data are often cumbersome, time-consuming, and prone to errors. Paper-based systems are still in use in many hospitals, making it difficult to share patient information among different departments and healthcare providers. Electronic systems are also in use, but many of them are outdated, not user-friendly, and lack essential features required to manage healthcare data efficiently.

The lack of a comprehensive and efficient system for managing patient data creates several problems for healthcare facilities, including increased administrative burden, data security risks, and decreased quality of patient care. Therefore, there is a need for a robust, user-friendly, and efficient Hospital Management System that can address these issues and provide healthcare professionals with timely access to accurate medical information.

The goal of this project is to develop a Hospital Management System that can streamline the management of patient records, appointments, and medical information for hospitals and healthcare facilities. The system aims to improve the quality of patient care by providing timely access to accurate medical information and reducing the administrative burden on healthcare professionals.

### Objective

### The main objective of the Hospital Management System is to develop a comprehensive software application that can improve the management of patient records, appointments, and medical information for hospitals and healthcare facilities. The following are the specific objectives of the project:

### To develop a user-friendly interface that enables healthcare professionals to access and manage patient data securely and efficiently.

### To create a centralized database that can store patient information securely and enable easy retrieval of patient data.

### To develop a system for patient registration and appointment scheduling that can reduce waiting times and improve patient satisfaction.

### To create a system for medical record management that can track patient progress and facilitate the exchange of medical information among healthcare professionals.

### To develop a prescription management system that can ensure accurate medication dosages and reduce medication errors.

### To provide reporting and analysis tools that can help healthcare professionals monitor patient outcomes and identify areas for improvement.

### To develop a highly customizable system that can be tailored to the unique needs and challenges of different healthcare facilities.

### To ensure that the software complies with industry standards for data security and privacy.

### By achieving these objectives, the Hospital Management System aims to improve the quality of patient.

### care by providing healthcare professionals with timely access to accurate medical information and reducing the administrative burden associated with managing patient data. The system also aims to enhance patient satisfaction by providing a seamless and efficient experience throughout their interactions with healthcare facilities.

### **Scope**

The Hospital Management System is a comprehensive software application designed to streamline the management of patient records, appointments, and medical information for hospitals and healthcare facilities. The system provides a range of features and functionalities that can be customized to the unique needs and challenges of different healthcare facilities.

The scope of the project includes the following features:

1. Patient registration and appointment scheduling
2. Medical record management
3. Prescription management
4. Reporting and analysis tools
5. User management and access control
6. Data security and privacy

The system will be developed using Java programming language and various open-source libraries and frameworks, including JavaFX for the user interface and Hibernate for data storage and retrieval. The software will be designed using an object-oriented approach, following industry-standard design patterns and best practices.

The Hospital Management System will be scalable and highly customizable, with options for configuring user roles and permissions, setting up custom fields, and integrating with other healthcare systems. The system will be tested thoroughly to ensure its reliability and efficiency, and feedback will be collected from healthcare professionals to improve its usability and functionality.

The project will focus on developing a system that can meet the needs of hospitals and healthcare facilities of varying sizes and complexity. The system will be designed to be highly flexible, allowing healthcare facilities to tailor the software to their specific requirements.

Overall, the Hospital Management System represents a significant advancement in healthcare technology, providing hospitals and healthcare facilities with a reliable, efficient, and scalable solution for managing their operations and improving patient outcomes.

### Modules

### The entire project consists of 4 modules.

Each module in the Hospital Management System:

1. **Admin Module**: The Admin module is responsible for managing the overall operations of the Hospital Management System. It is designed to provide an administrative interface for system administrators to configure the system, manage user roles and permissions, and perform other administrative tasks. The Admin module includes the following features:

* User Management: The module provides tools for managing user accounts and permissions. Administrators can create new user accounts, assign roles and permissions, and manage user passwords.
* System Configuration: The module enables administrators to configure system settings, including email and security settings.
* Reports and Analytics: The module provides tools for generating reports and analyzing system data, including user activity, patient records, and billing information.
* Backup and Recovery: The module provides tools for backing up and restoring system data to ensure data security and integrity.

1. **Receptionist Management Module**: The Receptionist Management module is responsible for managing patient registration and appointment scheduling. It is designed to provide an interface for receptionists to manage patient records, schedule appointments, and check-in patients. The Receptionist Management module includes the following features:

* Patient Registration: The module provides tools for registering new patients and capturing patient information, including demographics, contact information, and medical history.
* Appointment Scheduling: The module enables receptionists to schedule patient appointments and manage appointment calendars. The module also provides tools for sending appointment reminders and managing waitlists.
* Check-In and Check-Out: The module provides tools for managing patient check-in and check-out procedures, including tracking patient wait times and managing patient queues.

1. **Pharmacy Management Module**: The Pharmacy Management module is designed to help pharmacists manage medication orders and inventory. It is designed to provide an interface for pharmacists to manage medication orders, track medication usage, and monitor inventory levels. The Pharmacy Management module includes the following features:

* Medication Ordering: The module enables pharmacists to manage medication orders, including prescription orders and medication refills.
* Medication Inventory Management: The module provides tools for managing medication inventory levels, including tracking medication usage and generating alerts for low inventory levels.
* Medication Dispensing: The module provides tools for dispensing medication to patients and managing medication administration records.

1. **Patient Module**: The Patient module is responsible for managing patient information and medical records. It is designed to provide an interface for healthcare professionals to manage patient records, schedule appointments, and manage billing and insurance information. The Patient module includes the following features:

* Patient Registration: The module provides tools for registering new patients and capturing patient information, including demographics, contact information, and medical history.
* Medical Record Management: The module enables healthcare professionals to manage patient medical records, including diagnosis, treatment, and medication details. The module also provides tools for tracking patient progress and generating reports.
* Appointment Scheduling: The module enables healthcare professionals to schedule patient appointments and manage appointment calendars. The module also provides tools for sending appointment reminders and managing waitlists.
* Billing and Insurance: The module provides tools for managing patient billing and insurance information, including generating invoices, tracking payments, and managing insurance claims.
* Patient Portal: The module provides patients with access to their medical records, appointment schedules, and billing information. The patient portal enables patients to manage their health information and communicate with healthcare professionals securely.

**Benefits**

Here are some benefits of implementing a hospital management system:

1. Improved patient care: A hospital management system can help healthcare providers to manage patient records, medical histories, and treatment plans more efficiently and accurately, which can lead to better patient care and outcomes.
2. Streamlined operations: With a hospital management system, hospitals can automate many administrative tasks such as appointment scheduling, billing and payments, and inventory management, which can help to streamline operations, reduce costs, and improve efficiency.
3. Enhanced data security: Hospital management systems can provide robust data security features such as user authentication, access controls, and data encryption, which can help to protect sensitive patient data from unauthorized access or theft.
4. Increased collaboration: Hospital management systems can facilitate better communication and collaboration among healthcare providers, enabling them to share patient information, collaborate on treatment plans, and coordinate care more effectively.
5. Improved decision-making: A hospital management system can provide healthcare providers with real-time access to patient data, analytics, and insights, enabling them to make more informed decisions about patient care and treatment.

## CHAPTER 2

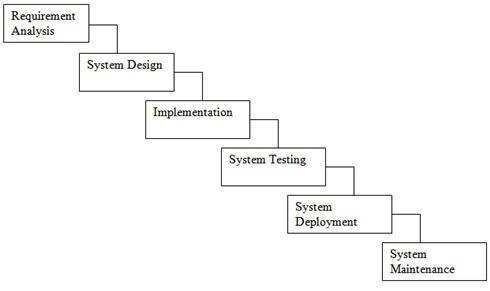
### Project Planning

**Project planning and scheduling**

Project planning is part of project management, which relates to the use of schedules such as Gantt charts to plan and subsequently report progress within the project environment. Initially, the project scope is defined and the appropriate methods for completing the project are determined. Following this step, the durations for the various tasks necessary to complete the work are listed and grouped into a work breakdown structure. The logical dependencies between tasks are defined using an activity network diagram that enables identification of the critical path.

**Methodology**

We have used Iterative and Incremental Development model (IID) for our project development. This development approach is also referred to as Iterative Waterfall Development approach. Iterative and Incremental Development is a software development process developed in response to the more traditional waterfall model. This model is designed to take care of such big project. The large and complicate project chiefly demand better development and testing procedure. The waterfall model is well known for its repeated testing process. Hence, I choose the waterfall model for developing my software.



*Fig. 2.1: Waterfall model*

**Some advantages of waterfall model:**

o Simple and easy to understand and use. o Easy to manage due to the rigidity of the model. o Phases are processed and completed one at a time o Works well for smaller projects where requirements are very well understood.

**Project Management Life Cycle**

The Project Management Life Cycle has four phases. Each project life cycle phase is described along with the tasks need to complete it.

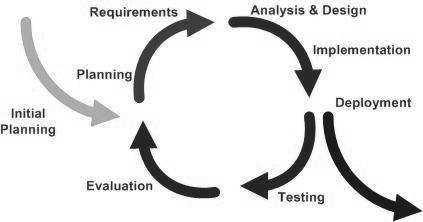
The four phases are.

Initiation

Planning

Execution

Closure.



*Fig. 2.2: Iterative and Incremental Life Cycle*

**Project Plan:**

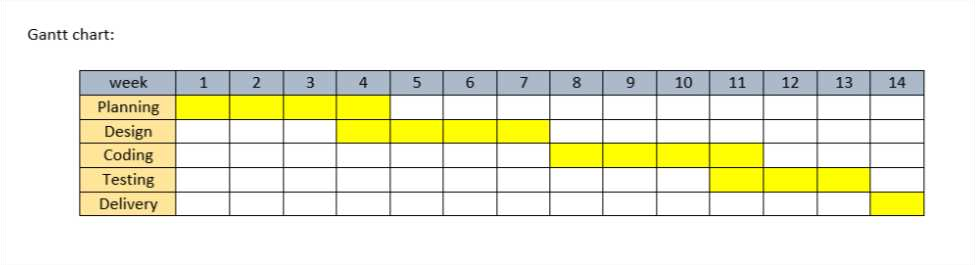
Once we examine that the project is feasible, I undertake project planning. The table below describes how we planned my project.

Table 2.1 Project Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | Task Name | Duration | Start | Finish |
| 2 | Planning | 31 days | 12/01/23 | 12/02/23 |
| 3 | Design | 38 days | 13/02/23 | 13/03/23 |
| 4 | Coding | 48 days | 14/03/23 | 01/05/23 |
| 5 | Delivery | 8 days | 01/05/23 | 04/04/23 |

**Schedule Representation**

Scheduling the project tasks is an important project planning activity. It involves deciding which tasks would be taken up when. To schedule the project activities, a software project manager needs to do the following these rules.



*Fig. 2.3: Gantt chart*

**Risk Management**

Software Risk Management is a proactive approach for minimizing the uncertainty and potential loss associated with a project. Some categories of risk include product size, business impact, customer related, process, technology, development environment, staffing (size and experience), schedule, and cost. Risk Management is a practice with processes, methods, and tools for managing risks in a project.

Risk identification is a systematic attempt to specify threats to the project plan. By identifying known and predictable risks, we can take a first step toward avoiding them when possible and controlling them when necessary. To perform the risk identification, we categorized the risk into different categories as:

Project Risk

Technical Risk

Business Risk

Known Risk

Predictable Risk

Unpredictable

Project risks for a hospital management system:

1. Technical Risk: The software used for the hospital management system may not be compatible with the hospital's existing hardware infrastructure, leading to delays and increased costs.
2. Business Risk: Changes in government regulations related to healthcare may impact the hospital's ability to implement certain features of the hospital management system, leading to delays and increased costs.
3. Known Risk: The hospital management system may require significant customization to meet the hospital's specific needs, which may lead to delays and increased costs.
4. Predictable Risk: The hospital management system may require significant user training and support, which may lead to delays and increased costs.
5. Unpredictable Risk: The hospital management system may experience unexpected downtime due to hardware or software failures, which could impact patient care and lead to increased costs.

**Requirements**

Hardware Requirements:

* Processor: 2 GHz dual-core processor or higher
* RAM: 4 GB or higher
* Storage: At least 500 MB of free hard disk space
* Display: Minimum resolution of 1280x720 pixels

Software Requirements:

* Operating System: Windows 7/8/10 or Linux (Ubuntu, Fedora, or Debian)
* Java Runtime Environment (JRE) 8 or higher
* MySQL Database Server 5.5 or higher
* Apache Tomcat Server 7.0 or higher
* Integrated Development Environment (IDE) like NetBeans, Eclipse, or IntelliJ IDEA for development and debugging.

## CHAPTER 3

Analysis

**Background Study**

System Analysis is a separation of a substance into parts for study and their implementation and detailed examination.

Before designing any system, it is important that the nature of the business and the way it currently operates are clearly understood. The detailed examination provides the specific data required during designing to ensure that all the client's requirements are fulfilled. The investigation or the study conducted during the analysis phase is largely based on the feasibility study. Rather it would not be wrong to say that the analysis and feasibility phases overlap. High-level analysis begins during the feasibility study. Though analysis is represented as one phase of the system development life cycle (SDLC), this is not true. Analysis begins with system initialization and continues until its maintenance. Even after successful implementation of the system, analysis may play its role for periodic maintenance and up gradation of the system. One of the main causes of project failures is inadequate understanding, and one of the main causes of inadequate understanding of the requirements is the poor planning of system analysis.

**Software system attributes**

**Reliability**: This application is a reliable product that produces fast & verified output of all its process.

**Availability**: This application will be available to use and help them to carry their operations conveniently.

**Security**: This application will be designed in a maintainable manner. It will be easy to incorporate new requirements in the individual modules.

**Scope of working:**

The proposed software product is the Hospital Management system (HMS). The system will be used in any hospital, clinic, dispensary, or pathology labs. Clinic, dispensary, or pathology to get the information from the patients and then storing that data for future usages. The current system in use is a paper-based system. It is too slow and cannot provide updated lists of patients within reasonable timeframe. The intention of the system is to reduce over-time pay and increase the number of patients that can be treated statements in these documents are both functional and non-functional.

**Feasibility study**

**Technical Feasibility**: This is concerned with specifying equipment and software that will successfully satisfy the user requirement; the technical needs of the system may vary considerably, but might include:

The facility to produce outputs in each time:

Response time under conditions.

Ability to process a certain volume of transaction at a particular seep.

Facility to communicate data to distant location.

**Operational Feasibility**: It is mainly related to human organization and political aspects. The points to be considered are:

What changes will be brought with the system?

What organizational structures are distributed?

What new skills will be required? Do the existing staff members have these skills? If not, can then the trained due course of time

**Economic Feasibility**: Economic analysis is the most frequently used technique for evaluating the effectiveness of a proposed system. More frequently known as cost/benefit system and compare them with costs. If benefits outweigh costs, a decision is taken to design and implement the system.

**Management Feasibility**: It is a determination of whether a proposed project will be acceptable to management. If does not accept a project of gives a negligible support to it; the analyst will tend to view the project as a no feasible one.

**Social Feasibility**: Social feasibility is a determination of whether the project will be acceptable to the people or not. This determination typically examines the probability of the project accepted by the group directly affected by the proposed system change.

## CHAPTER 4

**System Design**

**About**

1. System Architecture: This section could describe the overall architecture of the hospital management system, including the server-side and client-side components. It could also include information about the technology stack used for the project, such as Java, MySQL, and Swing.
2. Physical Design: This section could provide information on the recommended hardware specifications for the server and client systems, including CPU, memory, and storage requirements. It could also include information on network connectivity, such as the bandwidth required for the system.
3. Object-Oriented Design: This section could include a class diagram and object diagram for the hospital management system, illustrating the relationships between the different objects in the system. It could also provide information on the key classes and methods used in the system, as well as any design patterns used.
4. Database Schema and Relational Design: This section could include a database schema for the hospital management system, showing the tables, columns, and relationships between them. It could also include information on the data types used, as well as any constraints or business rules enforced.
5. User Interface Design: This section could provide information on the user interface for the hospital management system, including screenshots or mockups of the user interface. It could also include details on the functionality of the user interface, such as how users will navigate between different screens and how they will interact with the system.

Overall, the system design should provide a detailed plan for how the hospital management system will be built and implemented, including all the necessary technical specifications and design details. It should also be tailored to the specific needs of the hospital, considering factors such as the size of the hospital, the number of patients, and the types of medical services offered.

**Database Design**

Database design is the process of producing a detailed data model of database. This data model contains all the need logical and physical design choices and physical storage parameters needed to generate a design in a data definition language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity.

The term database design can be used to describe many different parts of the design of an overall database system. Principally, and most correctly, it can be thought of as the logical design of the base data structure used to store the data. In the relational model these are the tables and views. In an object database the entities and relationships map directly to object classes and named relationships. However, the term database design could also be used to apply to the overall process of designing, not just the base data structure, but also the forms and queries used as part of the overall database application within the database management system.

**DATABASE (MYSQL)**

**There are six tables in Hospital management system.**



*Tables in Hospital Management System*

**APPOINTMENT TABLE**

Text

Description automatically generated

*Fig 5.1 Appointment table description*

This table is used to store the data of All appointment. In this table there is Unique number for each new appointment which help to identify the Appointment.

**AUTHENTICATE TABLE**

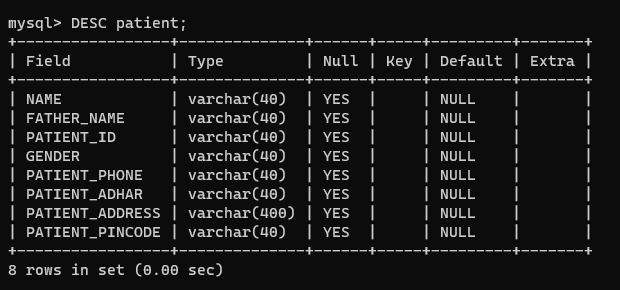
A picture containing text, screenshot, font, line

Description automatically generated

Fig 5.2 Authenticate table description.

This table is used to store the password for the Project it is required for the project to make it secure, and it can be change by Admin this password is required for open the project properly.

**PATIENT TABLE**

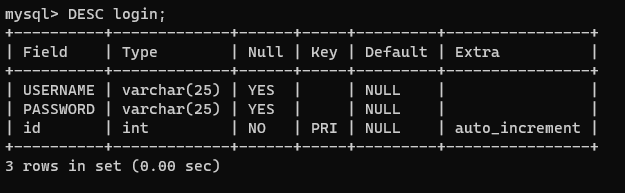


*Fig 5.3 Patient table description*

This table is used to store the data of all patients in this table there is Unique number for each patient which help us to identify the individual patient details.

Also, this table Store the Record the Contact details of the patient and patient can login in feature by using their details.

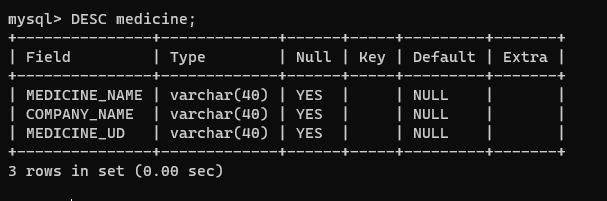
**LOGIN TABLE**



*Fig 5.4 Login table description.*

This table is used to store the data as id and password for the Administer can login by using the details if the details does not exist in the table no one can login in the Administrator option. It can be change by Admin or developer.

**MEDICINE TABLE**



*Fig 5.5 Medicine table description.*

This table is used to store the data of all medicine in this table there is Unique number for each medicine which help us to identify the individual medicine details.

**STAFF TABLE**

Text

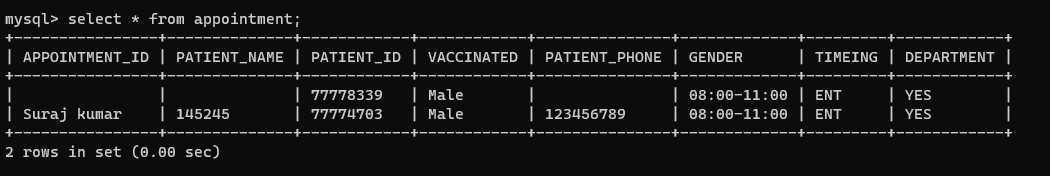
Description automatically generated with medium confidence

*Fig 5.6 Staff table description.*

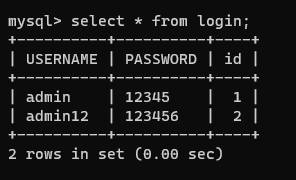
This table is used to store the data of all staff in this table there is Unique number for each staff which help us to identify the individual staff details.

**TABLE WITH VALUES**

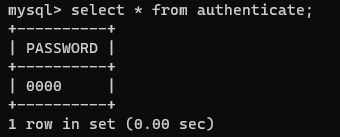
**Appointment table**



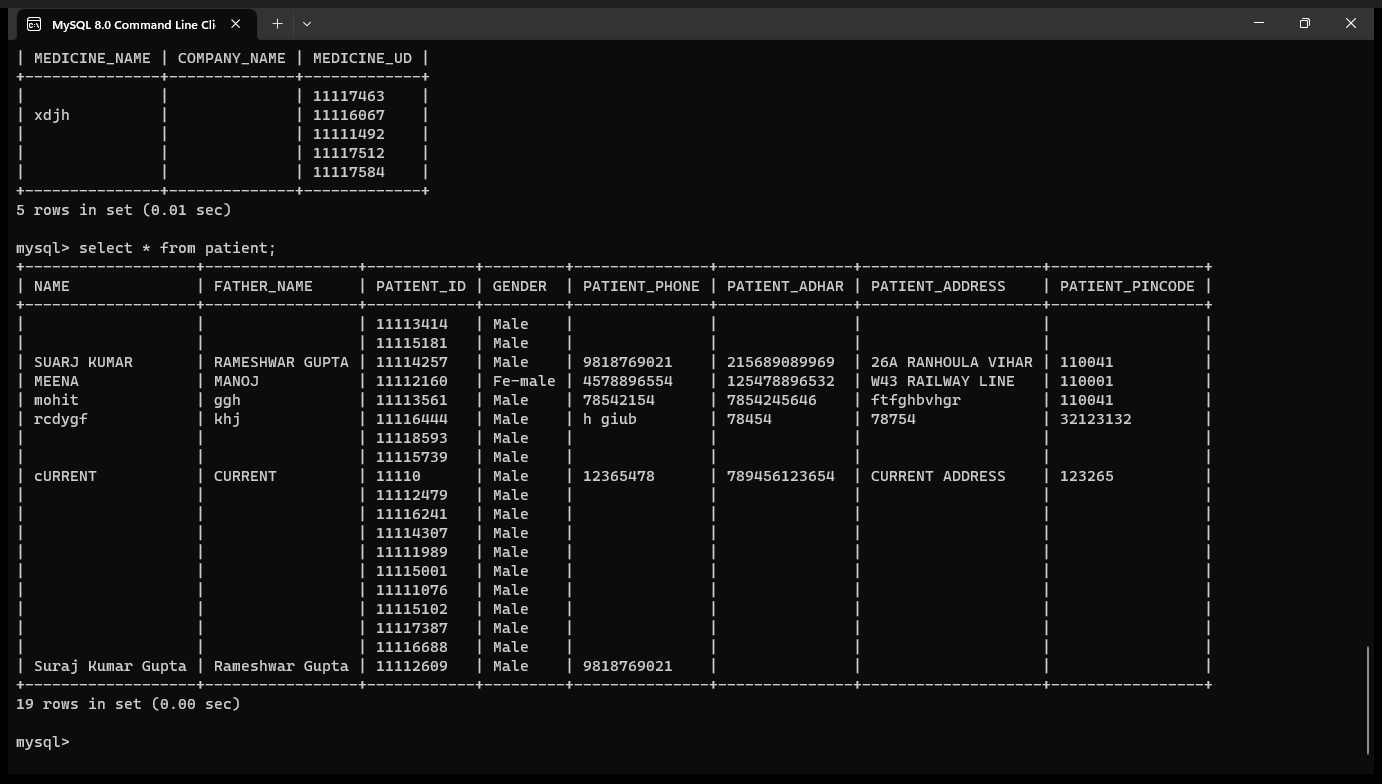
**Login table**



**Authenticate table**



**Patient table**



**Staff table**



**ER DIAGRAM**

An entity-relationship diagram (ERD) is an abstract and conceptual representation of data. Entity relationship modelling is a database modelling method, used to produce a type of conceptual schema or semantic data model of a system, often a relational database, and its requirements in a top-down fashion.

**Fig. 4.1:**

E-R Diagram of Online Appointments

**Hospital**

**Patients**

**Patient**

**information**

**Login**

**Appointment**

DETAILS

**Hospital**

**Doctor**

**record**

**Doctor**

**Patient**

**Hospital**

**Online**

**Appointment**

**Database schema of Hospital Management System**

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data.

A database schema can be divided broadly into two categories −

Physical Database Schema: This schema pertains to the actual storage of data and its form of storage like files, indices, etc. It defines how the data will be stored in a secondary storage.

Logical Database Schema: This schema defines all the logical constraints that need to be applied on the data stored. It defines tables, views, and integrity constraints.

**List of tables:**

LOGIN

STAFF

PATIENT

APPOINMENT

MEDICINE



*Database schema of Hospital management System*

**Data Flow Diagram of Hospital Management System**

The context diagram is the most abstract data flow representation of a system. It represents the entire system as a single bubble and. The various external entities with which the system interacts and the data flows occurring between the system and the external entities are also represented. The name context diagram is well justified because it represents the context in which the system is to exist i.e. the external entities (users) that would interact with the system and specific data items they would be receiving from the system.

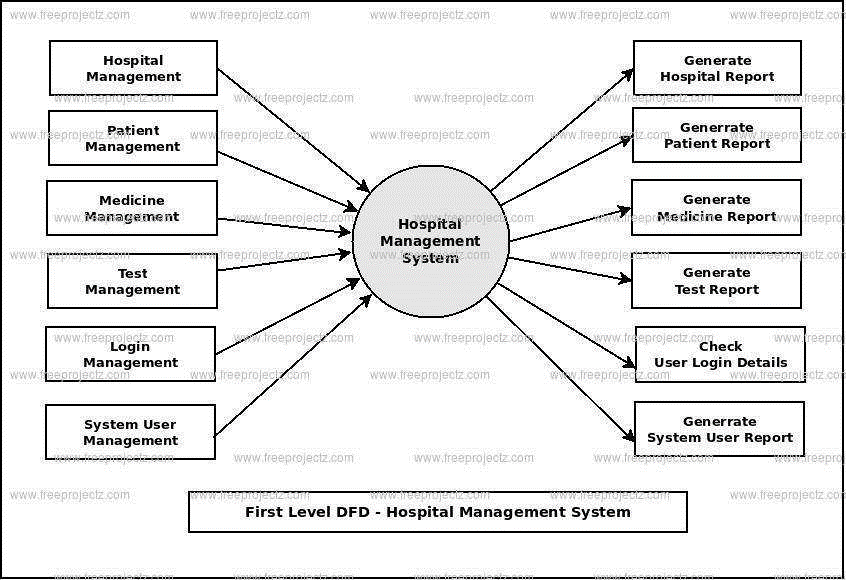
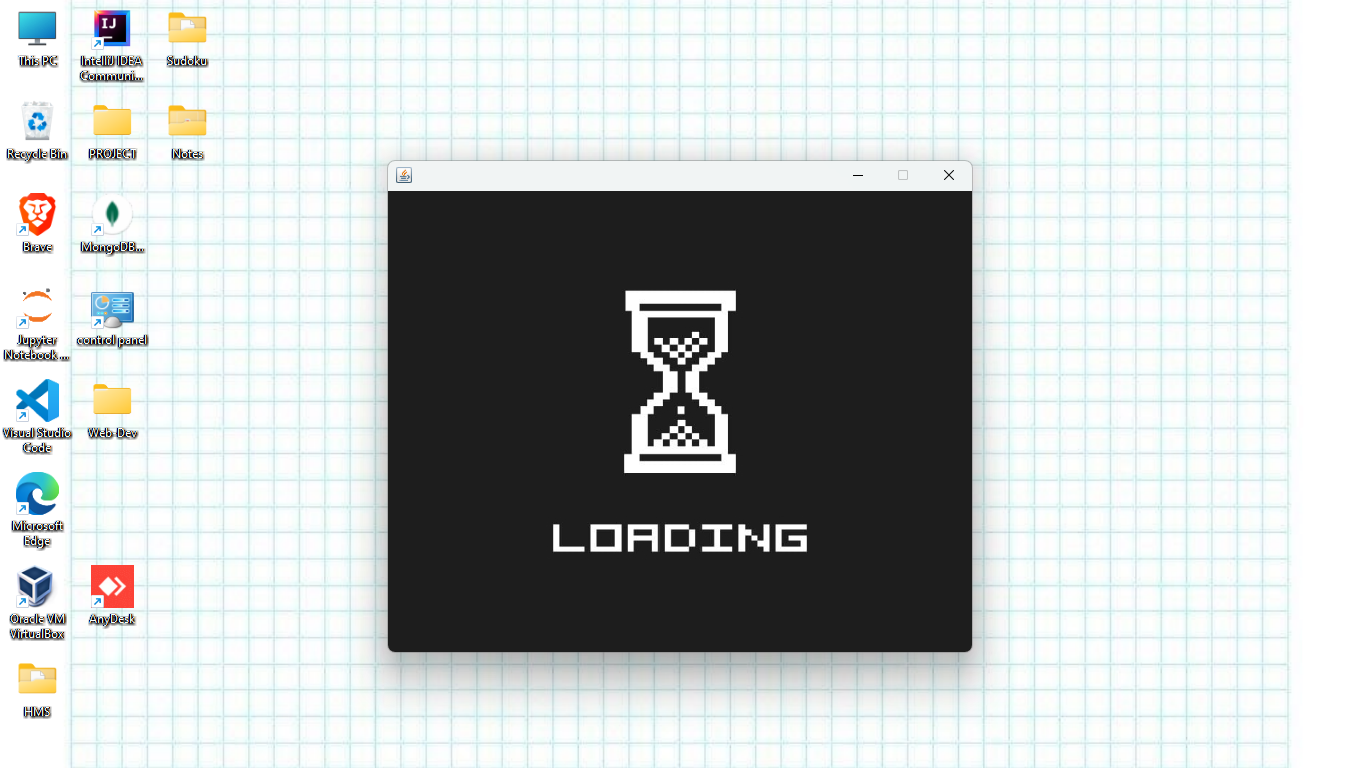


Fig. 4.3: Data flow diagram of Hospital Management Syatem

**User Interface (GUI)**

**Loading Screen**

This is GUI application which start with this screen and after this It ask for password for Verification.



*Loading Screen*

**HOME**

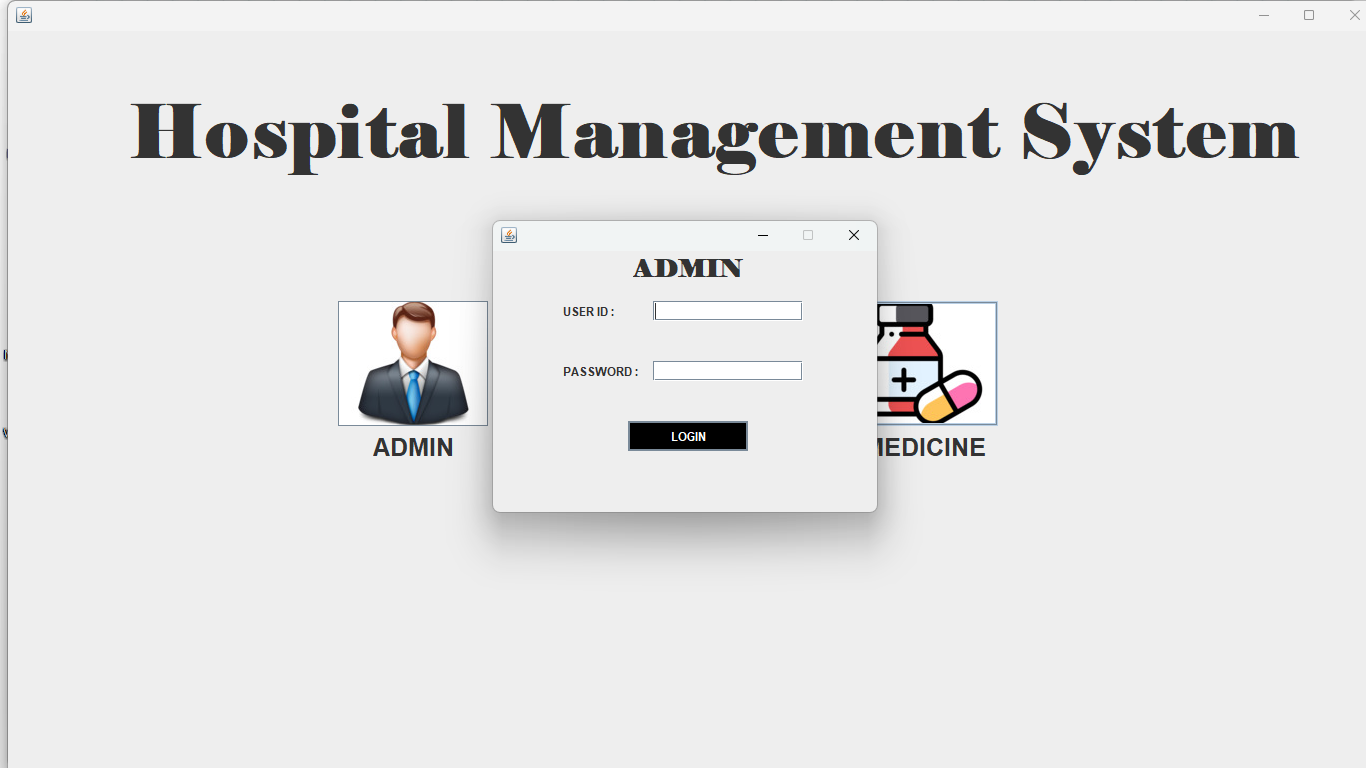
This is Home Screen It appears after the successful verification, and it show our modules (Admin/Reception/Patient/Medicine) which we have already Know.

****

*Home Screen*

**LOGIN**

Each module has their own login option and it ask for id and password if both are correct the Module login and perform their operation.



**DSHARDBOARD**

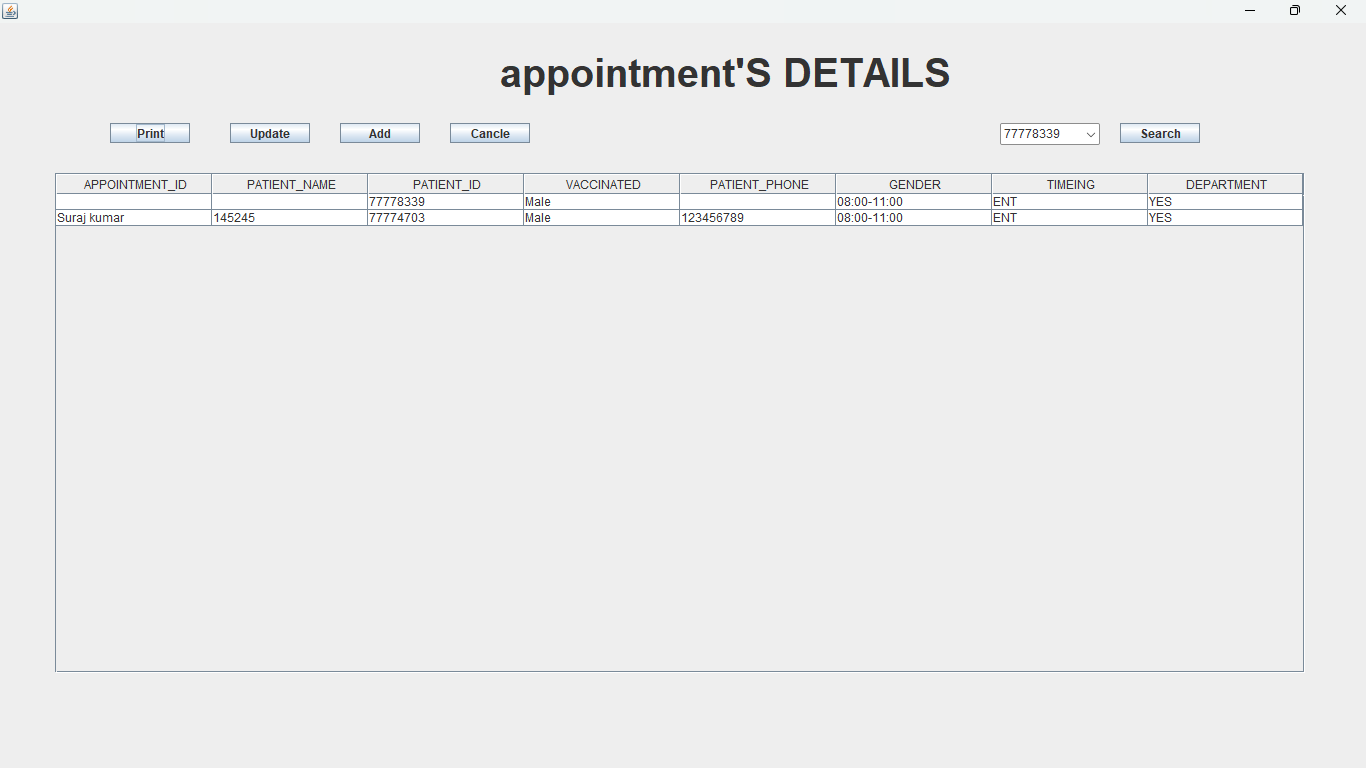
After successful login module use their dashboard and perform their operation which help him to use this project.



Fig. 4.4: Admin details page

**APPOINMENT DETAILS**

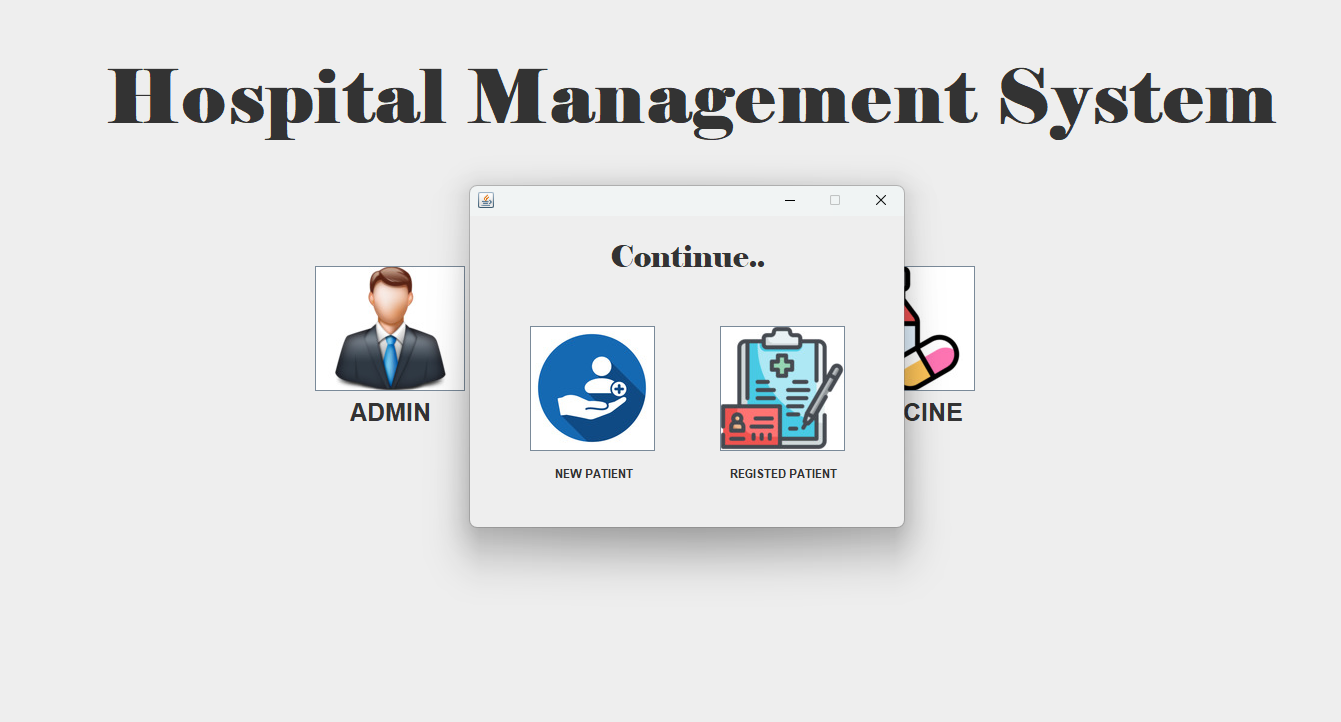
This Screen Shoe the information about the All Appointments to the Admin and Receptions module this is very secret information so Everyone cannot see the information about the appointment.



*Appointment Details*

**PATIENT**

There are two types of Patients in the Module New or registered so There are two options in the same modules and new patient add self and reg patient login there self.



**REGISTER PATIENT**

This is the module for the same in the new patient can register for their self and save their details and login in feature they can perform their option.

Graphical user interface

Description automatically generated

**Source Code**

There are some source code which help us to understand the concept of the project.

**Loading Screen**

|  |
| --- |
| *package HospitalManagementSystem;  import javax.swing.\*; import java.awt.\*;  public class Splash extends JFrame implements Runnable {  Thread t;   public Splash() {   ImageIcon background = new ImageIcon(ClassLoader.getSystemResource("image//loading.jpg"));  Image background2 = background.getImage().getScaledInstance(1400, 700, Image.SCALE\_DEFAULT);  ImageIcon back3 = new ImageIcon(background2);  JLabel backgroundLabel = new JLabel(back3);  add(backgroundLabel);  setResizable(false);  setLocation(380, 160);  setSize(600, 500);  t = new Thread(this);  t.start();  setVisible(true);   }  public void run() {   try {  Thread.sleep(3000);  setVisible(false);  new Authenticate();  } catch (Exception e) {  e.printStackTrace();  }  }  public static void main(String[] args) {  new Splash();  } }* |

**Add Details Screen**

|  |
| --- |
| *package HospitalManagementSystem;  import javax.swing.\*; import java.awt.\*; import java.awt.event.ActionEvent; import java.awt.event.ActionListener; import java.util.Random;  public class AddPatient extends JFrame implements ActionListener {  JTextField patient\_name\_field, patient\_father\_name\_field, patient\_phone\_field, patient\_doc\_field, patient\_address\_field, patient\_pincode\_field;  JButton submit, cancel;  JLabel lidno;  JComboBox cbgander;  Random ran = new Random();  long first4 = Math.abs((ran.nextLong() % 9000L) + 1000L);   AddPatient() {  setLayout(null);    JLabel heading = new JLabel("New Patient's Details");  heading.setFont(new Font("Arial", Font.BOLD, 50));  heading.setBounds(450, 20, 800, 50);  add(heading);  JLabel patient\_name\_label = new JLabel("Patient's name \*");  patient\_name\_label.setFont(new Font("Arial", Font.BOLD, 14));  patient\_name\_label.setBounds(300, 150, 150, 30);  add(patient\_name\_label);  patient\_name\_field = new JTextField();  patient\_name\_field.setBounds(450, 150, 150, 30);  patient\_name\_field.setEnabled(true);  add(patient\_name\_field);  /\*thsi label refer to father details\*/  JLabel patient\_father\_name\_label = new JLabel("father's Name");  patient\_father\_name\_label.setFont(new Font("Arial", Font.BOLD, 14));  patient\_father\_name\_label.setBounds(650, 150, 200, 30);  add(patient\_father\_name\_label);  patient\_father\_name\_field = new JTextField();  patient\_father\_name\_field.setBounds(850, 150, 150, 30);  add(patient\_father\_name\_field);  /\*this label holdes there id and it wiil be genrated by random funcation\*/  JLabel patient\_id\_label = new JLabel("Patient's ID \*");  patient\_id\_label.setFont(new Font("Arial", Font.BOLD, 14));  patient\_id\_label.setBounds(300, 250, 200, 30);  add(patient\_id\_label);  lidno = new JLabel("1111" + first4);  lidno.setBounds(450, 250, 250, 30);  add(lidno);  /\*thisi is label is refer to the patient mobile number details\*/  JLabel patient\_phone\_label = new JLabel("Contact No: \*");  patient\_phone\_label.setFont(new Font("Arial", Font.BOLD, 14));  patient\_phone\_label.setBounds(650, 250, 200, 30);  add(patient\_phone\_label);  patient\_phone\_field = new JTextField();  patient\_phone\_field.setBounds(850, 250, 150, 30);  add(patient\_phone\_field);  /\*gender \*/  JLabel patient\_gender = new JLabel("Gender");  patient\_gender.setFont(new Font("Arial", Font.BOLD, 14));  patient\_gender.setBounds(300, 350, 150, 30);  add(patient\_gender);  String gander[] = {"Male", "Fe-male", "Other"};  cbgander = new JComboBox(gander);  cbgander.setBounds(450, 350, 150, 30);  add(cbgander);  /\*this level of code hends the patients docoument\*/  JLabel patient\_doc\_label = new JLabel("Adhar No:");  patient\_doc\_label.setFont(new Font("Arial", Font.BOLD, 14));  patient\_doc\_label.setBounds(650, 350, 150, 30);  add(patient\_doc\_label);  patient\_doc\_field = new JTextField();  patient\_doc\_field.setBounds(850, 350, 150, 30);  add(patient\_doc\_field);  /\*this code is used to take address of the patient\*/  JLabel patient\_address\_label = new JLabel("Full address");  patient\_address\_label.setFont(new Font("Arial", Font.BOLD, 14));  patient\_address\_label.setBounds(300, 450, 150, 30);  add(patient\_address\_label);  patient\_address\_field = new JTextField();  patient\_address\_field.setBounds(450, 450, 150, 30);  add(patient\_address\_field);  /\*this code is used to have pincode of the adderss\*/  JLabel patient\_pincode\_label = new JLabel("Pin-Code");  patient\_pincode\_label.setFont(new Font("Arial", Font.BOLD, 14));  patient\_pincode\_label.setBounds(650, 450, 150, 30);  add(patient\_pincode\_label);  patient\_pincode\_field = new JTextField();  patient\_pincode\_field.setBounds(850, 450, 150, 30);  add(patient\_pincode\_field);  /\*there are only one OR TWO button to submit the details\*/  submit = new JButton("SUBMIT");  submit.addActionListener(this);  submit.setBounds(500, 600, 120, 30);  add(submit);  cancel = new JButton("CANCEL");  cancel.setBounds(700, 600, 120, 30);  add(cancel);  cancel.addActionListener(this);  JLabel note=new JLabel("Keep your Patient's ID as USER ID and Contact Number as Password");  note.setFont(new Font("Arial", Font.BOLD, 10));  note.setForeground(Color.red);  note.setBounds(500,500,800,80);  add(note);  /\*below this ine the frame command will be run\*/  setSize(1400, 800);  setVisible(true);  }  public static void main(String[] args) {  new AddPatient();  }  @Override  public void actionPerformed(ActionEvent a) {  if (a.getSource() == submit) {  String name = patient\_name\_field.getText();  String f\_name = patient\_father\_name\_field.getText();  String p\_id = lidno.getText();  String gen = (String) cbgander.getSelectedItem();  String p\_con = patient\_phone\_field.getText();  String p\_adhar = patient\_doc\_field.getText();  String p\_add = patient\_address\_field.getText();  String p\_pin = patient\_pincode\_field.getText();  try {  String query = "insert into patient values('" + name + "','" + f\_name + "','" + p\_id + "','" + gen + "','" + p\_con + "','" + p\_adhar + "','" + p\_add + "','" + p\_pin + "')";  Conn con = new Conn();  con.s.executeUpdate(query);  JOptionPane.showMessageDialog(null, "Successfully Added");  setVisible(false);  } catch (Exception ae) {  ae.printStackTrace();  }  setVisible(false);  }  if (a.getSource() == cancel) {  setVisible(false);   }  } }* |

**View Details**

|  |
| --- |
| package HospitalManagementSystem;  import net.proteanit.sql.DbUtils;  import javax.swing.\*; import java.awt.\*; import java.awt.event.ActionEvent; import java.awt.event.ActionListener; import java.sql.ResultSet;  public class PatientDetails extends JFrame implements ActionListener {  Choice cid;  JTable table;  JButton search, print, update, add, cancle;   PatientDetails() {  setLayout(null);  JLabel head = new JLabel("PATIENT'S DETAILS");  head.setFont(new Font("Arial", Font.*BOLD*, 40));  head.setBounds(500, 25, 1800, 50);  add(head);    cid = new Choice();  cid.setBounds(1000, 100, 100, 80);  add(cid);  try {  Conn c = new Conn();  ResultSet rs = c.s.executeQuery("select \* from patient");  while (rs.next()) {  cid.add(rs.getString("PATIENT\_ID"));  }  } catch (Exception e) {  e.printStackTrace();   }  table = new JTable();  try {  Conn c = new Conn();  ResultSet rs = c.s.executeQuery("select \* from patient");  table.setModel(DbUtils.*resultSetToTableModel*(rs));  } catch (Exception e) {  e.printStackTrace();  }  JScrollPane jsp = new JScrollPane(table);  jsp.setBounds(55, 150, 1250, 500);  add(jsp);   search = new JButton("Search");  search.setBounds(1120, 100, 80, 20);  add(search);  search.addActionListener(this);   print = new JButton("Print");  print.setBounds(110, 100, 80, 20);  add(print);  print.addActionListener(this);   update = new JButton("Update");  update.setBounds(230, 100, 80, 20);  add(update);  update.addActionListener(this);   add = new JButton("Add");  add.setBounds(340, 100, 80, 20);  add(add);  add.addActionListener(this);   cancle = new JButton("Cancle");  cancle.setBounds(450, 100, 80, 20);  add(cancle);  cancle.addActionListener(this);   setSize(1400, 800);  setVisible(true);  }   public static void main(String[] args) {  new PatientDetails();  }   @Override  public void actionPerformed(ActionEvent e) {  if (e.getSource() == search) {  String query = "select \* from patient where PATIENT\_ID = '" + cid.getSelectedItem() + "'";  try {  Conn c = new Conn();  ResultSet rs = c.s.executeQuery(query);  table.setModel(DbUtils.*resultSetToTableModel*(rs));  } catch (Exception eo) {  eo.printStackTrace();  }  }  if (e.getSource() == print) {  try {  table.print();  } catch (Exception ae) {  ae.printStackTrace();  }  }  if (e.getSource() == add) {  setVisible(false);  new AddPatient();   }  if (e.getSource() == update) {  //  } else {  setVisible(false);  }   } } |

**Connection**

|  |
| --- |
| package HospitalManagementSystem;  import net.proteanit.sql.DbUtils;  import javax.swing.\*; import java.awt.\*; import java.awt.event.ActionEvent; import java.awt.event.ActionListener; import java.sql.ResultSet;  public class PatientDetails extends JFrame implements ActionListener {  Choice cid;  JTable table;  JButton search, print, update, add, cancle;   PatientDetails() {  setLayout(null);  JLabel head = new JLabel("PATIENT'S DETAILS");  head.setFont(new Font("Arial", Font.*BOLD*, 40));  head.setBounds(500, 25, 1800, 50);  add(head);    cid = new Choice();  cid.setBounds(1000, 100, 100, 80);  add(cid);  try {  Conn c = new Conn();  ResultSet rs = c.s.executeQuery("select \* from patient");  while (rs.next()) {  cid.add(rs.getString("PATIENT\_ID"));  }  } catch (Exception e) {  e.printStackTrace();   }  table = new JTable();  try {  Conn c = new Conn();  ResultSet rs = c.s.executeQuery("select \* from patient");  table.setModel(DbUtils.*resultSetToTableModel*(rs));  } catch (Exception e) {  e.printStackTrace();  }  JScrollPane jsp = new JScrollPane(table);  jsp.setBounds(55, 150, 1250, 500);  add(jsp);   search = new JButton("Search");  search.setBounds(1120, 100, 80, 20);  add(search);  search.addActionListener(this);   print = new JButton("Print");  print.setBounds(110, 100, 80, 20);  add(print);  print.addActionListener(this);   update = new JButton("Update");  update.setBounds(230, 100, 80, 20);  add(update);  update.addActionListener(this);   add = new JButton("Add");  add.setBounds(340, 100, 80, 20);  add(add);  add.addActionListener(this);   cancle = new JButton("Cancle");  cancle.setBounds(450, 100, 80, 20);  add(cancle);  cancle.addActionListener(this);   setSize(1400, 800);  setVisible(true);  }   public static void main(String[] args) {  new PatientDetails();  }   @Override  public void actionPerformed(ActionEvent e) {  if (e.getSource() == search) {  String query = "select \* from patient where PATIENT\_ID = '" + cid.getSelectedItem() + "'";  try {  Conn c = new Conn();  ResultSet rs = c.s.executeQuery(query);  table.setModel(DbUtils.*resultSetToTableModel*(rs));  } catch (Exception eo) {  eo.printStackTrace();  }  }  if (e.getSource() == print) {  try {  table.print();  } catch (Exception ae) {  ae.printStackTrace();  }  }  if (e.getSource() == add) {  setVisible(false);  new AddPatient();   }  if (e.getSource() == update) {  //  } else {  setVisible(false);  }   } } |

**Login**

|  |
| --- |
| package HospitalManagementSystem;  import javax.swing.\*; import java.awt.\*; import java.awt.event.ActionEvent; import java.awt.event.ActionListener; import java.sql.ResultSet;  public class AdminLogin extends JFrame implements ActionListener {   JButton login\_button;  JTextField username\_field;  JPasswordField password\_field;   public AdminLogin() {  setLayout(null);   JLabel login\_heading = new JLabel("ADMIN");  login\_heading.setFont(new Font("Elephant", Font.*BOLD*, 25));  login\_heading.setBounds(140, 0, 200, 35);   add(login\_heading);    /\*this is username and its text filed \*/  JLabel username\_text = new JLabel("USER ID :");  username\_text.setBounds(70, 50, 120, 20);  add(username\_text);  username\_field = new JTextField();  username\_field.setBounds(160, 50, 150, 20);  add(username\_field);    /\*this is password and its text filed\*/  JLabel password\_text = new JLabel("PASSWORD :");  add(password\_text);  password\_text.setBounds(70, 110, 120, 20);  password\_field = new JPasswordField();  password\_field.setBounds(160, 110, 150, 20);  add(password\_field);  /\* there are only one single butane to authenticate \*/  login\_button = new JButton("LOGIN");  login\_button.setBounds(135, 170, 120, 30);  login\_button.setBackground(Color.*BLACK*);  login\_button.setForeground(Color.*WHITE*);  add(login\_button);  login\_button.addActionListener(this);   getRootPane().setDefaultButton(login\_button);  setResizable(false);  setLocation(485, 220);  setSize(400, 300);  setVisible(true);  }   public static void main(String[] args) {  new AdminLogin();  }  @Override  public void actionPerformed(ActionEvent e) {  if (e.getSource() == login\_button) {  String username = username\_field.getText();  String password = password\_field.getText();  String query = "select \* from login where USERNAME='" + username + "'and PASSWORD='" + password + "'";  try {  Conn c = new Conn();  ResultSet rs = c.s.executeQuery(query);  if (rs.next()) {  setVisible(false);  new Admin();  } else {  JOptionPane.*showMessageDialog*(null, "Invalid User ID or Password");  setVisible(false);  }  } catch (Exception ae) {  ae.printStackTrace();   }    }  } } |

## CHAPTER 5

**System Implementation**

**Implementation**

Implementation is the process of having system personal check out and provides new equipment’s into use, train the user to install a new application and construct any files of data needed to use it. There are three types of implementations. Implementation of computer system to replace a manual system. To problem encountered are covering files, training user, creating accurate files, and verifying print outs for integrity. Implementation of a new computer system to replace an existing one. This is usually difficult conversion. If not properly planned, there can be many problems. So large computer system many take as long as a year to convert. Implementation of a modified application to replace the existing one using the same computer. This type of conversing is relatively easy to handle, usually there are no major change in the file. Our project is yet to be implemented.

**Implementation Environment**

The implementation view of software requirement presents the real-world manifestation of processing functions and information structures. This computerized system is specified in a manner that dictates accommodation of certain implementation details.

The implementation environment of the developed system facilitates multiple users to use this system simultaneously. The user interfaces are designed keeping in mind that the users of this system are familiar to using GUI-based systems. Thus, we restricted ourselves to developing a GUI-based system so that it becomes easier for the end user to get acquainted to the developed system.

**Functional Requirement**

This system interface is divided into two sections.

Administrator interface.

Users interface.

**Administrator Interface**

Administrator can Update any details.

Administrator can setup user, Pharmacy, and other account.

Administrator can change Password.

**User Interface**

User can’t do any operation without account.

Log in and log out system.

User can update/edit their own account.

## CHAPTER 6

**Testing**

**Testing**

In the context of software development, testing refers to the process of evaluating the behaviour or performance of a software system to ensure that it meets the specified requirements and quality standards. Testing is an essential part of the software development life cycle (SDLC) and typically involves a series of activities designed to identify defects or errors in the software, measure its quality attributes, and validate its functionality and performance.

**Integration Testing**

Integration testing done before, during and after integration of a new module into the main software package. This involves testing of each individual code module. One piece of software can contain several modules which are often created by several different programmers. It is crucial to test each modules effect on the entire program model. After integration testing the project works successfully.

**Unit Testing**

Unit testing performed on each module or block of code during development. Unit testing is normally done by the programmer who writes the code.

**System Testing**

System testing done by a professional testing agent on the completed software product before it is introduced to the market.

**Acceptance Testing**

Acceptance testing is a beta testing of the product done by the actual end user.

**Recovery Testing**

Recovery testing is done to demonstrate a software salutation is reliable, trustworthy and can successfully recoup form possible crashes.

**Functional Testing**

Functional Testing also known as functional completeness testing. Functional Testing involves trying to think of any possible missing functions. Testers might make a list of additional functionalities that a product could improve it during functional testing.

**Hardware/Software Testing**

IBM refers to Hardware/Software testing as “HW/SW Testing”. This is when the tester focuses his/her attention on the interactions between the hardware and software during system testing.

**Security Testing**

Security Testing is a variant of Software Testing which ensures, that system and applications in an organization, are free from any loopholes that may cause a big loss. Security testing of any system is about finding all possible loopholes and weaknesses of the system which might result into a loss of information at the hands of the employees or outsiders of the Organization.

**Advantages**

The software helps to handle the entire administration of hospitals and healthcare facilities. Typically, such a software includes various modules that help doctors manage their assignments and schedules, carry out patient registration, maintain store inventory records, keep track of medicine, administration, maintain blood bank (with available blood type) details, individual record of patients with their test reports, nursing and housekeeping service details, financial information, including final billing & payments, insurance details and much more. After the customized software is implemented and integrated into the system, patient care and hospital administration becomes an easy job.

## CHAPTER 7

**Conclusion**

## Conclusion

## The Hospital Management System project is an essential tool for managing hospital operations and improving patient care. The system is designed to provide an efficient way of managing patient records, appointments, and pharmacy inventory. The project planning and analysis played a vital role in ensuring that the system met the specified requirements.

## The system's architecture was designed using object-oriented design principles, with a focus on modularity and reusability. The system consists of four modules: Admin, Admin Activity, Receptionist Management, and Pharmacy Management. Each module performs specific functions, and they work together seamlessly to provide a comprehensive hospital management solution.

## The physical design of the system was done using class diagrams and object diagrams, which helped to visualize the system's components and their interactions. The database schema and relational design were carefully designed to ensure data integrity, security, and ease of use.

## The user interface design was another critical aspect of the system. The interface was designed to be user-friendly and intuitive, even for non-technical staff. The interface provides easy access to all the system's functionalities, and the staff can quickly learn how to use the system with minimal training.

## The system's performance was satisfactory, with fast response times and accurate results. The system's scalability may be an issue for larger hospitals, but the system's modular design makes it easy to add new features and functionality as needed.

## There were a few limitations to the system. The system requires a stable internet connection to function, which may not be available in some areas. The system's learning curve may also be a limitation for staff members who are not familiar with the system. Additionally, the system may require frequent updates to keep up with changing hospital regulations and requirements.

## In conclusion, the Hospital Management System project provides a practical solution for managing hospital operations and improving patient care. The project's planning and analysis were crucial in ensuring that the system met the specified requirements. The system's architecture, physical design, database schema, and user interface design were carefully crafted to provide a comprehensive and user-friendly hospital management solution. Despite a few limitations, the system has the potential to benefit hospitals and medical facilities worldwide with further development and refinement.

## CHAPTER 8

**References**

**References**

[1] Kaur, A., & Singh, J. (2020). Hospital Management System using Java and MySQL. International Journal of Recent Technology and Engineering, 8(5), 4890-4893.

[2] Joshi, P., & Patil, D. (2018). Hospital Management System using PHP and MySQL. International Journal of Computer Science and Mobile Computing, 7(5), 120-126.

[3] Ramezani, M., Saberi, M., & Soleymani, S. (2015). A Hospital Information System Architecture Based on the SOA Paradigm. Procedia Computer Science, 73, 242-248.

[4] Zadeh, A. S., & Javanmard, M. (2017). A Comparative Study of Hospital Information Systems (HIS) and Electronic Medical Records (EMR). Journal of Health Management and Informatics, 4(2), 63-68.

[5] El-Masri, S., & Tarhini, A. (2015). Factors Affecting the Adoption of Hospital Information Systems in Lebanon. Journal of Medical Systems, 39(6), 1-13.

[6] Connolly, T. M., & Begg, C. E. (2014). Database Systems: A Practical Approach to Design, Implementation, and Management (6th ed.). Pearson.

[7] Pressman, R. S. (2014). Software Engineering: A Practitioner's Approach (8th ed.). McGraw-Hill Education.

[8] Sommerville, I. (2016). Software Engineering (10th ed.). Pearson.

[9] Sharda, R., Delen, D., & Turban, E. (2015). Business Intelligence and Analytics: Systems for Decision Support (10th ed.). Pearson.

[10] O'Brien, J. A., & Marakas, G. M. (2018). Management Information Systems (11th ed.). McGraw-Hill Education.