

A
Minor Project-II Report
On

“Queuing models in OPDs/ availability of beds/ admission of patients. A hospital based solution is ideal which can be integrated with city wide module.”

Submitted in partial fulfillment of
The requirements for the 6th Semester Sessional Examination of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

By

ABHISEK PANDA – 22UG010159
DEBABRATA MISHRA - 22UG010273
DIPTESH NARENDRA - 22UG010357

Under the Supervision of
Mrs. Gitanjali Mishra

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



GANDHI INSTITUTE OF ENGINEERING AND TECHNOLOGY
UNIVERSITY, ODISHA, GUNUPUR

2024 – 25



**GANDHI INSTITUTE OF ENGINEERING AND
TECHNOLOGY
UNIVERSITY, ODISHA, GUNUPUR**

Dist. – Rayagada-765022, Visit us:- www.giet.edu

Department of Computer Science & Engineering

CERTIFICATE

*This is to certify that the project work entitled “**Queuing models in OPDs/ availability of beds/ admission of patients. A hospital based solution is ideal which can be integrated with city wide module**” is done by Abhisek Panda (22UG010159), Debabrata Mishra (22UG010273), Diptesh Narendra (22UG010357) in partial fulfillment of the requirements for the 6th Semester Sessional Examination of Bachelor of Technology in **Computer Science and Engineering** during the academic year 2024-25. This work is submitted to the department as a part of evaluation of 6th Semester Minor Project-II.*

Project Supervisor

Class Teacher

Project Coordinator, 3rd Year

HoD, CSE

ACKNOWLEDGEMENT

"We would like to express our deepest gratitude to our esteemed project supervisor, **Mrs. Gitanjali Mishra**, Department of Computer Science and Engineering, for providing us with the opportunity to undertake this project. Her invaluable guidance, expertise, and unwavering support have been instrumental in the successful completion of this project report.

We also extend our sincere thanks to our class teacher, **Mrs. Gitanjali Mishra**, for their constant encouragement and support throughout the project's execution.

Additionally, we would like to thank **Mr. Bhavani Sankar Panda**, Project Coordinator, **Dr. Premanshu Sekhar Rath**, Head of the Department of Computer Science and Engineering, and **Prof. (Dr.) Kakita Murali Gopal**, Deputy Dean, Computational Science, SOET, for their consistent support, guidance, and assistance.

Last but not least, we would like to thank our friends, family members, and everyone else who has provided us with unconditional support and encouragement during the project's execution. Your contributions have been invaluable to us.

ABHISEK PANDA – 22UG010159

DEBABRATA MISHRA - 22UG010273

DIPTESH NARENDRA - 22UG010357

Abstract

Our Project Queuing models in OPDs/ availability of beds/ admission of patients. A hospital based solution is ideal which can be integrated with city wide module aim in Super Admin Dashboard for a comprehensive web-based system designed to optimize outpatient department (OPD) queuing, bed availability, and patient admissions while integrating a city-wide healthcare module for real-time data sharing. The system enables efficient management of patient records, bed tracking, queue management, and hospital inventory through a scalable database architecture built with MongoDB Atlas. By ensuring real-time updates and a seamless data flow between hospitals, the platform enhances coordination and efficiency across healthcare institutions.

The project began with defining objectives, reviewing the software requirements specification (SRS) document, finalizing the technology stack, and setting milestones for development. The database schema was designed to handle patient data, bed availability, queue management, and hospital inventory, ensuring structured and efficient data handling. UI wireframes were carefully designed for patients, doctors, and administrators, with a strong emphasis on user experience and accessibility. A secure authentication system using JSON Web Tokens (JWT) and role-based access control was integrated to ensure data security and controlled access to functionalities based on user roles.

A priority-based queue management system was developed, allowing dynamic patient sorting based on severity and appointment status to minimize waiting times. The system also incorporates a patient feedback mechanism to enhance hospital services and an appointment management module that enables centralized tracking and scheduling of patient visits. Additional features include real-time bed availability tracking, patient admission integration, and automated alerts for low availability. The city-wide module facilitates seamless data sharing and referrals between hospitals, ensuring that patients receive timely care.

Notification systems for appointments, bed availability updates, and inventory tracking are integrated, with alerts delivered via email to ensure proactive management of resources. An analytics dashboard provides insights into hospital operations, helping administrators optimize efficiency and decision-making. After extensive testing and debugging, the system was prepared for final review and deployment. Future enhancements include further optimization of queuing models, improved security measures, scalability improvements, and automation features to drive continuous advancement in hospital management and healthcare delivery. The platform is designed to be adaptable, ensuring it can accommodate future technological advancements and evolving healthcare needs. Machine learning models may be incorporated to predict patient inflow trends, enabling hospitals to allocate resources more effectively. Integration with telemedicine services is also a possibility, allowing remote consultations and better accessibility for patients. By leveraging modern web technologies and scalable infrastructure, the system aims to revolutionize hospital administration and create a more efficient, responsive, and data-driven healthcare environment.

TABLE OF CONTENTS

CHAPTER 1	1 - 13
1.1 Introduction	2
• Purpose	4
• Scope	6
• Features	8
• Works Done in the Related Area	10
CHAPTER 2. SYSTEM ANALYSIS	14 - 19
2.1 User Requirements (SRS)	14
2.2 Hardware Requirements	17
2.3 Software Requirements	18
CHAPTER 3	20 - 22
3.1 Language Used	20
3.2 Library Used	21
CHAPTER 4. SYSTEM DESIGN & SPECIFICATION	23 - 42
High Level Design (HLD)	
4.1 Project Model	23
4.2 Structure Chart	26
4.3 DFD	27 - 28
4.4 E-R Diagram	29
4.5 UML (Use Case, Class, Activity Diagram)	30
4.6 Interaction/Collaboration Diagram	33
Low Level Design (LLD)	
4.1.1 Pseudo code	34
4.2.1 FLOW CHART	38
4.3.1 SCREEN SHOTS	39 - 43
CHAPTER 5. CODING	44 - 49
CHATER 6. TESTING	50 - 52
CHATER 7. CONCLUSION & LIMITATION	53
CHATER 8. REFERENCE /Bibliography	54

CHAPTER - 1

1.1 Introduction

Effective healthcare management is crucial for ensuring timely and efficient patient care. Hospitals and medical institutions often face challenges in managing outpatient department (OPD) queues, bed availability, and patient admissions due to the increasing demand for healthcare services. A well-structured hospital management system can significantly enhance the efficiency of medical facilities by streamlining processes, reducing patient wait times, and optimizing resource allocation. This project focuses on developing a comprehensive hospital management system that integrates various hospital operations into a single, user-friendly dashboard.

The system is designed to address critical aspects of hospital administration, including real-time queue management, bed availability tracking, and seamless patient admission processes. By leveraging a centralized database and real-time data updates, the platform ensures that healthcare providers can efficiently allocate resources while minimizing delays. One of the key features of the system is its priority-based queue management, which dynamically sorts patients based on severity and appointment status, ensuring that critical cases receive prompt attention. Additionally, the integration of automated notifications for appointments, bed status, and inventory updates helps hospitals maintain smooth operations and proactive resource management.

A well-organized database architecture is essential for handling vast amounts of patient records, appointment details, and hospital inventory data. The system utilizes MongoDB Atlas for scalable and secure data storage, enabling hospitals to manage patient information efficiently. The database schema has been structured to support real-time updates, ensuring that hospital staff always have access to the latest information regarding bed availability, admissions, and medical supplies. This approach minimizes discrepancies in data handling and enhances decision-making at all levels of hospital administration.

User experience plays a crucial role in the effectiveness of a hospital management system. To ensure accessibility and ease of use, the user interface has been designed with a strong focus on clarity and responsiveness. The platform provides dedicated dashboards for different users, including patients, doctors, and administrators. Each user role is assigned specific permissions to ensure controlled access to sensitive information while allowing seamless interaction with the system. The interface is developed using modern web technologies, ensuring a smooth and efficient user experience across various devices.

Security is a key concern in healthcare management systems, as they handle sensitive patient data and hospital records. The system incorporates a secure authentication mechanism using JSON Web Tokens (JWT) and role-based access control to restrict functionalities based on user roles. By implementing robust security measures, the platform ensures data confidentiality and prevents unauthorized access. Encryption and secure communication protocols further enhance data protection, reducing the risk of breaches or unauthorized modifications to patient records.

Another essential component of the system is the real-time integration of hospital resources with city-wide healthcare modules. The system is designed to facilitate seamless data sharing between hospitals, allowing for better coordination and patient referrals. Hospitals can update bed availability in real-time, ensuring that emergency cases are efficiently directed to facilities with available capacity. This interconnected approach enhances overall hospital efficiency and ensures that patients receive timely medical attention.

The hospital management system also features a comprehensive analytics dashboard that provides valuable insights into hospital operations. Administrators can monitor key performance metrics such as patient inflow, bed occupancy rates, and appointment trends. This data-driven approach enables hospital management to optimize staffing, predict resource shortages, and improve overall operational efficiency. With the ability to generate reports and track historical data, hospitals can make informed decisions that contribute to better patient care and streamlined workflows.

In addition to operational efficiency, the system includes a patient feedback mechanism that allows hospitals to assess and improve the quality of their services. Patients can submit feedback on their experiences, helping hospital administrators identify areas for improvement. This feature promotes transparency and encourages continuous enhancements in healthcare service delivery. The integration of a structured feedback system ensures that patient concerns are addressed promptly, leading to higher satisfaction levels and improved hospital performance.

While the current implementation focuses on optimizing OPD queuing, bed management, and hospital inventory, the system is designed to support future enhancements. Potential advancements include the incorporation of machine learning algorithms to predict patient inflow patterns and further optimize resource allocation. Additionally, integration with telemedicine services could enable remote consultations, expanding healthcare access for patients who may face geographical or logistical barriers.

The project follows a structured development approach, beginning with objective definition and system requirements analysis. The technology stack, including MongoDB, Bootstrap, and JWT-based authentication, was carefully selected to ensure scalability and security. The development process involved designing a robust database schema, creating user-friendly UI wireframes, and implementing core functionalities such as queue management, appointment scheduling, and real-time updates. Extensive testing and debugging were conducted to ensure the reliability and accuracy of the system before deployment.

The hospital management system aims to modernize healthcare administration by integrating real-time data sharing, optimized queuing models, and advanced security features. By providing an intuitive and efficient platform for hospital staff, doctors, and patients, the system enhances coordination, reduces wait times, and improves overall hospital efficiency.

1.2.1 Purpose

Our primary purpose of this hospital management system is to create a structured and efficient digital platform that streamlines hospital operations while ensuring optimal resource utilization. The system is designed to address critical challenges in hospital administration, such as patient queue management, bed availability tracking, appointment scheduling, and inventory control. By integrating these functionalities into a unified platform, the system enhances hospital workflows, reduces administrative burdens, and improves overall patient care.

One of the main objectives of the system is to eliminate inefficiencies in patient queue management. Traditional queuing methods often result in long wait times, mismanagement of appointments, and difficulties in prioritizing critical cases. This system introduces a dynamic, priority-based queue management model that ensures patients with severe conditions are given immediate attention while also maintaining an organized appointment schedule. By implementing automated queue updates and real-time notifications, hospitals can provide better service while reducing patient frustration and operational delays.

Another key purpose of the system is to enhance the transparency and accessibility of hospital resources. Many hospitals face challenges in managing bed availability, leading to overcrowding or inefficient allocation of hospital beds. This system provides real-time tracking of bed occupancy, ensuring that hospitals can efficiently manage patient admissions. By offering accurate and up-to-date information about available beds, hospital staff can make quicker decisions regarding patient placements, emergency cases, and referrals to other hospitals if necessary.

Apart from patient and bed management, the system plays a crucial role in optimizing hospital inventory. Medical supplies, medications, and essential resources must be carefully tracked to prevent shortages or overstocking. The system ensures that hospitals receive alerts when supplies are running low, allowing administrators to place timely orders. This level of automation reduces the chances of medical stockouts, ensuring that hospitals are always equipped to handle patient needs. The inventory module also tracks usage patterns, helping administrators plan purchases more efficiently and reduce waste.

A significant purpose of the system is to provide a centralized data management approach that simplifies hospital operations. Many hospitals still rely on manual record-keeping, which is prone to errors, misplacement of documents, and delays in retrieving patient information. By digitizing hospital records, the system ensures that all patient information, medical histories, and administrative details are stored securely and can be accessed instantly by authorized personnel. This digital transformation not only reduces paperwork but also improves coordination between different hospital departments.

The system is also designed to enhance communication between patients, doctors, and hospital staff. Miscommunication in hospitals can lead to delays in treatment, appointment mix-ups, and inefficient resource allocation. The system integrates notification mechanisms for appointment reminders, bed availability updates,

and urgent medical alerts. By keeping all stakeholders informed in real time, the system helps reduce misunderstandings and improves hospital efficiency.

A well-organized hospital management system must also ensure compliance with security and data privacy regulations. Protecting patient data is a key concern, as hospitals handle sensitive medical records that must not be exposed to unauthorized access. The system incorporates a robust security framework, including authentication mechanisms and access control policies. By restricting data access based on user roles, the system ensures that only authorized individuals can access specific patient and hospital data, thus maintaining confidentiality and preventing data breaches.

Another fundamental purpose of the system is to provide hospital administrators with analytical insights into hospital operations. Data-driven decision-making is essential for hospitals to improve efficiency and optimize resources. The system offers real-time reports and statistics on key hospital metrics, such as patient inflow, appointment trends, resource utilization, and staff performance. This data allows hospital management to identify bottlenecks, allocate resources more effectively, and implement process improvements that enhance hospital functionality.

The system is also designed to be scalable and adaptable to future advancements. As hospital requirements evolve, new features can be integrated to further improve efficiency. Future enhancements may include artificial intelligence-driven patient flow predictions, automation of administrative tasks, and integration with emerging healthcare technologies. The modular nature of the system ensures that additional functionalities can be incorporated without disrupting existing operations, making it a long-term solution for hospital management.

Additionally, the system facilitates seamless communication between different stakeholders in the hospital environment. Doctors, administrators, nurses, and patients all require timely updates and notifications to ensure smooth operations. The system integrates automated notifications for appointment reminders, test results, bed availability updates, and urgent alerts. This ensures that hospital staff remain informed about critical updates while patients receive timely reminders, reducing missed appointments and improving coordination across departments.

Ultimately, the purpose of this project is to develop a robust and efficient hospital management system that enhances hospital operations, improves patient care, and optimizes the use of hospital resources. By digitizing processes, automating routine tasks, and ensuring real-time communication between hospital staff and patients, the system aims to revolutionize healthcare administration. The long-term vision is to create a sustainable and scalable solution that can adapt to the ever-changing demands of healthcare institutions while continuously improving efficiency and service quality.

1.2.2 SCOPE

The hospital management system is designed to improve healthcare operations by automating administrative tasks, optimizing resource allocation, and enhancing communication between hospital staff, doctors, and patients. The system aims to streamline patient admissions, appointment scheduling, bed availability tracking, and inventory management while ensuring security, scalability, and accessibility. The scope of this system extends to various aspects of hospital administration, focusing on efficiency, accuracy, and real-time data integration.

One of the primary areas within the system's scope is outpatient department (OPD) management. The system facilitates patient registration, appointment booking, and queue management through a structured and automated process. By introducing a dynamic, priority-based queuing system, the platform ensures that critical cases are attended to first, while routine appointments follow a well-organized schedule. This approach reduces waiting times, improves patient satisfaction, and enhances overall operational efficiency. The system also enables doctors and hospital staff to access patient records instantly, making consultations more effective and informed.

The system also covers bed availability tracking and patient admission management. Hospitals often struggle with maintaining accurate and up-to-date records of available beds, leading to overcrowding and inefficient patient placement. This system provides real-time updates on bed occupancy, ensuring that hospital administrators and staff can make quick and informed decisions regarding patient admissions. In emergency situations, this feature is particularly useful in directing patients to hospitals with available capacity, reducing delays in critical care.

Another key component within the system's scope is hospital inventory management. Maintaining an adequate supply of medical equipment, medications, and other essential resources is vital for hospital operations. The system automates inventory tracking, providing alerts when stock levels are low and ensuring that supplies are replenished in a timely manner. By preventing shortages and overstocking, the system helps hospitals operate more efficiently and ensures that necessary medical supplies are always available.

Additionally, the system facilitates seamless communication between different stakeholders in the hospital environment. Doctors, administrators, nurses, and patients all require timely updates and notifications to ensure smooth operations. The system integrates automated notifications for appointment reminders, test results, bed availability updates, and urgent alerts. This ensures that hospital staff remain informed about critical updates while patients receive timely reminders, reducing missed appointments and improving coordination across departments.

Security and access control are also included within the system's scope. Since hospitals handle highly sensitive patient data, it is crucial to protect this information from unauthorized access. The system implements secure authentication mechanisms and role-based access control, ensuring that only authorized personnel can access

specific data and functionalities. By maintaining strict security protocols, the system ensures patient confidentiality and compliance with healthcare data regulations.

Beyond the internal hospital operations, the system also has the potential to integrate with external healthcare networks. Future enhancements could include data sharing between hospitals to facilitate patient referrals, emergency response coordination, and remote consultations. This would enable hospitals to operate in a more interconnected healthcare ecosystem, improving patient care and resource distribution across multiple medical institutions.

Another aspect within the scope of this project is the analytics and reporting module. The system generates real-time insights into hospital operations, including patient inflow, appointment trends, resource utilization, and staff efficiency. These analytics enable hospital administrators to make data-driven decisions, identify inefficiencies, and optimize workflows for improved hospital management. Reports generated by the system can also assist in strategic planning and regulatory compliance.

The system is designed to be scalable, allowing hospitals to expand functionalities as needed. Future developments may include the incorporation of machine learning models to predict patient inflow, optimize resource allocation, and automate administrative tasks. Additionally, integrating telemedicine features could enable remote consultations, making healthcare services more accessible to patients who cannot visit hospitals physically.

In conclusion, the scope of this hospital management system extends across multiple areas of healthcare administration, from patient admission and queue management to inventory tracking, communication, security, and data analytics. By automating processes, improving efficiency, and ensuring real-time data access, the system enhances hospital functionality and provides a strong foundation for future technological advancements in healthcare management.

The system also plays a significant role in enhancing patient engagement and experience. By providing patients with access to their medical records, appointment history, and prescription details through a user-friendly interface, the system empowers them to take control of their healthcare journey. Patients can easily book appointments, receive updates on their consultations, and provide feedback on the services received. This feature not only increases patient satisfaction but also helps hospitals gather valuable insights into areas that may require improvement.

Furthermore, the system can be adapted for use in different types of healthcare institutions, including multi-specialty hospitals, smaller clinics, and emergency care centers. Its modular architecture allows customization based on the specific requirements of different healthcare providers. Whether a hospital requires enhanced queue management, advanced inventory tracking, or seamless patient referral mechanisms, the system can be tailored to fit their needs. This flexibility ensures that the platform remains relevant across various healthcare environments.

1.2.3 FEATURES

The hospital management system is designed to enhance hospital operations by integrating advanced features that streamline patient management, resource tracking, and administrative functionalities. The system provides an efficient, user-friendly interface with secure authentication, data encryption, and role-based access control. With real-time analytics, notification alerts, and a scalable architecture, the system ensures a seamless experience for patients, doctors, hospital administrators, and super admins.

One of the core features of the system is the **user authentication and role-based access** module. Every user, including patients, doctors, admins, and super admins, must log in securely using an authentication system that verifies credentials before granting access. Role-based access ensures that each user type has the appropriate permissions to access specific data and functionalities. This enhances security by preventing unauthorized access to sensitive medical records and administrative controls.

The **dashboard system** is another major feature that provides users with an interactive interface displaying relevant information. Patients can track their appointments and receive notifications, while doctors can manage their schedules and view patient medical histories. Admin dashboards provide insights into hospital operations, including bed availability, patient statistics, and inventory levels. Super admins have access to high-level analytics, allowing them to oversee hospital-wide performance, manage user roles, and maintain system functionality.

The **appointment scheduling** module enables patients to book appointments with doctors efficiently. The system displays available slots in the outpatient department (OPD), allowing patients to select a suitable time. Appointment management ensures proper scheduling to minimize waiting times and prevent overcrowding. Automated reminders and notifications are sent to both patients and doctors to reduce missed appointments.

Another essential feature is **bed availability tracking**, which allows hospital staff to monitor real-time occupancy status. This feature prevents overcrowding and ensures that patients are directed to hospitals with available beds. The system provides instant updates on occupied, reserved, and available beds, allowing administrators to make informed decisions regarding patient admissions.

The **inventory management system** ensures that medical supplies and hospital equipment are tracked efficiently. Administrators can monitor stock levels, receive alerts for low inventory, and schedule timely reorders to prevent shortages. The system also tracks expiration dates for medicines, ensuring that outdated supplies are not used in patient treatment.

The **patient record management system** securely stores patient data, including medical history, prescriptions, test results, and previous appointments. Doctors can easily access patient records, making consultations more effective and informed. The system ensures that all medical data is encrypted, maintaining confidentiality and data integrity.

A **priority-based queue management system** is integrated into the platform to optimize patient flow within the hospital. The system dynamically sorts patients based on the severity of their condition and appointment

status. Emergency cases receive priority attention, while regular check-ups follow a structured scheduling approach. This system significantly reduces patient waiting times and ensures that critical cases receive immediate medical care.

The **reporting and analytics module** provides hospital administrators and super admins with valuable insights into hospital performance. Graphs and charts display patient inflow trends, resource utilization, appointment statistics, and inventory levels. These visual reports help in data-driven decision-making, allowing hospitals to optimize operations and improve efficiency. The super admin and admin dashboards include graphical representations of hospital data using **Doughnut Charts, Bar Graphs, and Pie Charts**, ensuring that key statistics are easily accessible.

The system includes **telemedicine functionality**, enabling patients to consult doctors remotely via video calls. This feature is particularly useful for follow-up appointments and non-emergency cases, reducing the need for in-person hospital visits. The platform ensures secure video consultations with encrypted communication channels, maintaining patient confidentiality.

A **feedback and grievance system** is integrated to collect patient reviews about hospital services. Patients can submit feedback on their experiences, helping hospitals improve service quality. The system also allows administrators to address grievances efficiently, ensuring better patient satisfaction.

The **notification and alert system** keeps users informed about appointment reminders, bed availability, prescription updates, and emergency alerts. Automated messages are sent via email or in-app notifications, ensuring that important updates reach the intended recipients promptly.

Another feature is the **staff and workforce management module**, which allows administrators to manage doctor schedules, assign shifts, and track staff availability. This system ensures that hospital personnel are distributed efficiently, preventing overburdening of medical staff and maintaining optimal hospital operations.

Additionally, the **hospital services module** provides an overview of available healthcare services. Patients can explore the range of medical specialties, diagnostic services, and emergency care options offered by the hospital. This helps in better decision-making and improves patient awareness of hospital facilities.

The system's **scalability and adaptability** ensure that new features can be integrated as required. Future enhancements may include AI-driven patient diagnosis assistance, IoT-based real-time patient monitoring, and blockchain-based medical record security. These upgrades will further improve hospital efficiency and enhance the patient experience.

1.2.4 Works Done in the Related Area

The hospital management system has been designed and developed with several critical components to ensure efficiency, accessibility, and seamless management of hospital resources. The following sections outline the work completed in various areas, highlighting the implementation of essential functionalities.

Landing Page

- A structured homepage providing an overview of the hospital's mission, services, and essential information.
- Integrated quick access links for login, registration, and hospital services.
- Designed with an intuitive user interface to improve navigation and user experience.

User Registration

- Secure registration module allowing users to create accounts with validated input fields.
- Data protection measures ensure encrypted storage of user credentials.
- Implemented error-handling mechanisms for incorrect or missing details.

User Login

- A dedicated secure login system for patients, doctors, admins, and super admins.
- Password recovery and reset functionality to enhance accessibility.
- Authentication system with role-based access restrictions.

User Dashboard

- Interactive dashboard allowing users to manage profiles and track appointments.
- Notification panel providing real-time updates regarding healthcare services.
- Integrated access to hospital resources and healthcare-related documentation.

Super Admin Dashboard

- A centralized panel for super admins to oversee hospital operations.
- Features include user management, hospital analytics, and system functionality monitoring.
- Graphical insights displaying patient inflow, doctor availability, and inventory status.

Super Admin Login

- A dedicated authentication page for super admins with secured access.
- Integrated session management to maintain data integrity.
- Role-based controls limiting access to administrative functionalities.

Admin Dashboard

- Comprehensive dashboard for managing hospital operations, patient statistics, and inventory.
- Features include real-time monitoring of bed availability and staff schedules.
- Role assignment and access control mechanisms for efficient administration.

Admin Login

- Secure login page for hospital administrators with controlled access.
- Features include verification systems and multi-factor authentication.
- Data encryption ensuring login details remain confidential.

Add Patient

- A structured form-based interface for inputting patient details.
- Includes fields for medical history, personal information, and contact details.
- Secure data storage ensuring confidentiality of patient records.

Add Doctor

- Dedicated page for adding doctor profiles, including specialization and availability.
- Integrated search and filter options to locate doctors efficiently.
- Profile management features allowing updates and modifications.

Inventory Management

- **Add Product** – Module for adding new medicines and medical supplies.
- **Manage Product** – Feature to update, track, and remove inventory items.
- **Manage Sell** – Section to record medicine sales and monitor stock levels.
- **Manage Seller** – Vendor management system for tracking suppliers and orders.
- **Manage Purchase** – Order management tool for tracking and restocking medical supplies.

Doctor Dashboard

- Personalized portal for doctors with an intuitive interface for managing patients.
- Access to patient records, appointment schedules, and medical history.
- Communication system allowing doctors to provide virtual consultations.

Feedback System

- A structured platform for patients to submit service reviews and feedback.
- Data-driven approach to improve hospital services based on patient insights.

- Grievance resolution module to handle patient complaints efficiently.

Bed Availability

- Real-time tracking system displaying the current status of hospital beds.
- Interactive dashboard allowing efficient patient admission and discharge management.
- Automated updates to prevent bed shortages and ensure proper utilization.

OPD Availability

- Scheduling module allowing patients to check OPD service availability.
- Appointment booking and confirmation system integrated into the platform.
- Automated reminders sent to patients and doctors for scheduled consultations.

Appointment Scheduling

- Patients can book, modify, and cancel appointments through an easy-to-use interface.
- Doctors receive real-time updates on their schedules.
- Automated notifications for upcoming appointments.

Patient Records

- Secure storage of medical history, prescriptions, and diagnostic reports.
- Role-based access ensuring only authorized personnel can view patient data.
- Advanced search and filtering options to locate patient information quickly.

Reports & Analytics

- Graph-based reports providing insights into hospital performance.
- Metrics on patient admissions, doctor availability, and hospital capacity.
- Custom report generation for administrative decision-making.

Telemedicine Portal

- Virtual consultation module for remote doctor-patient interactions.
- Encrypted communication ensuring secure video calls.
- Online prescription generation and digital health documentation.

Supplies & Equipment Inventory

- Module for tracking medical supplies, ensuring efficient stock management.
- Automatic alerts for low inventory and near-expiry products.
- Vendor and supplier management system integrated with purchasing records.

Notifications & Alerts

- System-wide notifications for upcoming appointments, hospital updates, and emergency alerts.
- Automated reminders for patients regarding their consultation schedules.
- Alert system for doctors regarding critical patient cases.

System Administration

- Dedicated settings module for managing hospital operations.
- User role assignment and permission control.
- Customization options for hospital services and department management.

CHAPTER – 2

SYSTEM ANALYSIS

2.1 User Requirements (SRS)

The development of this hospital management system requires a well-structured Software Requirements Specification (SRS) document that outlines the functional and non-functional requirements. These requirements ensure that the system meets the expectations of various stakeholders, including patients, doctors, administrators, and super admins. The system has been designed to streamline hospital operations, enhance patient care, and improve efficiency in healthcare service delivery.

The **user requirements** encompass the needs of different users interacting with the system. The primary users include **patients, doctors, hospital administrators, and super admins**, each with specific roles and permissions. Patients require an intuitive and accessible interface to book appointments, manage their health records, and receive real-time updates on hospital services. Doctors need a structured dashboard to manage appointments, review patient history, and coordinate with hospital staff. Administrators handle overall hospital operations, such as bed management, inventory tracking, and appointment scheduling, while super admins oversee the entire system and ensure smooth functionality.

A **secure login and authentication system** is an essential requirement to maintain privacy and data protection. The system must provide separate login portals for different user roles and restrict access based on assigned privileges. Patients should only have access to their medical records, whereas doctors and administrators must have additional permissions to view and update patient data. The authentication mechanism must be robust, incorporating password encryption and session management to prevent unauthorized access.

Another significant requirement is **appointment management**, which enables patients to schedule and modify their appointments easily. The system should allow doctors to view and manage their schedules, ensuring optimal utilization of available consultation slots. Automated notifications and reminders should be integrated to inform both patients and doctors about upcoming appointments, reducing missed consultations.

A **real-time bed availability system** is crucial for effective hospital management. Patients and administrators should be able to check the status of available beds to streamline the admission process. This feature should also notify staff when bed occupancy reaches critical levels, allowing for better resource allocation. The bed management system should be integrated with patient admission and discharge processes to update availability status dynamically.

The **inventory management module** is another key component, ensuring that hospital supplies, medicines, and medical equipment are tracked efficiently. The system should support functionalities like adding, updating, and removing stock items, as well as monitoring expiration dates and setting alerts for low inventory levels. A well-organized inventory system minimizes the risk of shortages and ensures that necessary medical supplies are always available.

A **role-based access control mechanism** is necessary to maintain data security and operational integrity. Different levels of access must be granted to users based on their roles within the hospital. For instance, a patient should only have access to their health records, while doctors should have permission to update medical histories and prescribe treatments. Administrators and super admins should have full control over system configurations, user management, and hospital operations.

A **feedback and grievance system** is essential for improving hospital services. Patients should have the option to provide feedback on their experiences, rate the quality of services, and submit complaints if necessary. Hospital administrators must be able to review and address these concerns promptly to enhance patient satisfaction and maintain service quality.

For enhanced **data security**, the system must incorporate strong encryption methods for storing and transmitting sensitive medical data. Secure authentication methods, including multi-factor authentication and token-based authorization, should be implemented to prevent unauthorized access. Compliance with healthcare data regulations should be a priority to ensure patient confidentiality.

The system should also include a **real-time notification system** to keep users informed about important updates. Patients should receive alerts regarding appointment confirmations, prescription refills, and health check-up reminders. Doctors and hospital staff should be notified about scheduled appointments, critical patient conditions, and inventory restocking requirements.

A **scalable architecture** is a fundamental requirement to accommodate hospital growth and expansion. The system should be able to handle increasing numbers of users, patient records, and operational tasks without performance degradation. The backend should be designed to support a high volume of concurrent requests, ensuring seamless functionality even during peak usage.

To improve hospital operations, an **analytics and reporting dashboard** must be integrated. This dashboard should generate reports on hospital performance, patient inflow, doctor availability, and resource utilization. The insights provided by these reports can help administrators make data-driven decisions, optimize hospital workflows, and improve overall efficiency.

The system must support **telemedicine functionalities** to facilitate virtual consultations between doctors and patients. This feature is particularly beneficial for remote healthcare services, enabling patients to receive medical advice without visiting the hospital physically. Secure video conferencing, online prescription generation, and electronic health records should be incorporated into the telemedicine module.

A **comprehensive user interface** is essential for ensuring ease of use and accessibility. The system should be designed with a responsive and mobile-friendly layout, allowing users to access hospital services from different devices, including smartphones, tablets, and desktops. The interface should be intuitive, ensuring that users can navigate the system effortlessly.

Finally, the system must support **multi-channel communication**, enabling patients and hospital staff to communicate through email, SMS, or in-app messaging. This functionality ensures that important

notifications, appointment reminders, and emergency alerts reach users in a timely manner. The integration of a chatbot or automated assistance feature can further enhance user engagement and provide instant support for common queries.

By implementing these **user requirements**, the hospital management system ensures efficient patient care, optimized hospital operations, and seamless interaction between stakeholders. The focus on security, scalability, and user-friendliness enhances the overall functionality of the system, making it a valuable tool for healthcare management.

In addition to the core functionalities outlined above, the hospital management system is designed to foster seamless collaboration between all stakeholders involved in healthcare delivery. By incorporating real-time data tracking and automated notifications, the system enhances communication and ensures that critical information is always up to date. The integration of **telemedicine** functionalities allows for more flexible and accessible healthcare services, especially for patients in remote areas or those unable to physically visit the hospital. With the inclusion of **feedback and grievance systems**, the hospital can continuously monitor patient satisfaction and address concerns, leading to improved service quality. The system's **scalable architecture** ensures that it can grow with the hospital's evolving needs, accommodating an increasing number of users, records, and operational tasks without compromising on performance. Furthermore, the system's robust **role-based access control** ensures that sensitive patient data is protected while granting authorized users the necessary access to perform their duties. The **analytics and reporting** dashboard provides administrators with actionable insights, empowering them to make informed decisions that optimize hospital workflows and resource utilization. By prioritizing **security**, **user experience**, and **efficiency**, this hospital management system lays the foundation for improved healthcare delivery, ultimately benefiting patients, medical professionals, and administrators alike. Moreover, the system's integration with **inventory management** plays a crucial role in ensuring that the hospital's medical supplies and resources are always available and adequately stocked. By enabling real-time tracking and automated alerts for low inventory, the hospital can proactively manage stock levels and avoid shortages of essential medicines and equipment. This not only improves operational efficiency but also contributes to better patient care by ensuring that the necessary resources are readily available. Additionally, the **appointment management** feature simplifies the scheduling process for patients and doctors alike, reducing administrative overhead and enhancing the patient experience. Automated reminders and notifications for both patients and doctors help minimize missed appointments and ensure that consultations happen as planned. The system also supports **multi-channel communication**, allowing hospital staff to easily reach patients via **SMS**, **email**, or **in-app messaging**, ensuring timely delivery of critical updates.

2.2 HARDWARE REQUIREMENT

To ensure the efficient operation of the hospital management system, the following hardware components are required:

1. Server Requirements

- Processor: Quad-core Intel i5
- RAM: Minimum 8 GB (Recommended: 16 GB for high-volume transactions).
- Storage: Minimum 1 TB SSD for fast data access and retrieval.
- Database Hosting: MongoDB Atlas or an equivalent cloud-based/dedicated database server.
- Network Connectivity: High-speed internet with at least 10 Mbps bandwidth.

2. Client Devices (Doctors, Staff, and Admins)

- Desktops:
 - Processor: Quad-core Intel i5.
 - RAM: Minimum 8 GB.
 - Storage: 256 GB SSD or higher.
 - Screen Resolution: 1920x1080 pixels for a better UI experience.
- Mobile & Tablets Support:
 - The system should be accessible on modern Android and iOS devices.

3. Networking Infrastructure

- Internet Speed: Minimum 10 Mbps (Higher speed for cloud-based hosting).
- Wi-Fi/LAN Setup: Secure and stable network connectivity.
- Firewalls & VPN: Protection against unauthorized access.

4. Data Backup & Recovery

- Backup Server: Minimum 4 TB HDD or cloud-based backup solution.
- Automated Backup Schedules: To prevent data loss in case of system failures.
- UPS (Uninterruptible Power Supply): Ensuring system uptime during power outages.

By implementing these **hardware components**, the hospital management system can operate seamlessly, ensuring **scalability, security, and high performance**.

2.3 Software Requirements

Admin Dashboard

- Modern Sidebar Navigation: Includes sections like Add Hospital, Check Hospital Status, Order Medicine, and Feedback.
- Dynamic Metrics: Displays key metrics including Total Hospitals, Doctors, Patients, Inventory Distributors, Nurses, Admin Staff, and Bed Availability.
- Visual Data Representations:
 - Doughnut Charts to show filled and available resources.
 - Bar Graphs for hospital statistics.
 - Pie Chart for patient distribution.

Doctor Dashboard

- Appointment Management:
 - A card-based layout for appointments (instead of traditional tables).
 - Visual representation of appointments, follow-ups, and patients checked through charts (Bar, Pie, and Line Graphs).
- Quick Communication: Features for instant call and video consultations to facilitate easy doctor-patient communication.
- Patient Data Reports: Option to generate PDF reports for patient data.

Hospital Management

- Resource Management: A streamlined approach to manage resources such as hospital beds, equipment, and staff.
- Interactive Data: Real-time data updates for metrics like bed occupancy and resource allocation.

Inventory Management

- Medicine Ordering: A dedicated section for ordering medicines and monitoring inventory levels.
- Distributor Integration: Allows checking the availability and status of various inventory distributors.

User Interface Design

- Responsive and User-Friendly: The interface adapts to various devices with a modern, interactive design using Bootstrap and Tailwind CSS for styling.
- Data Fetching & Dynamic Updates: The system uses the Fetch API to dynamically load and display real-time data.

Security & Authentication

- Role-Based Access Control: Different access levels for Super Admin, Doctors, and Hospital Admin to ensure data security and proper role handling.
- Session Management: Secure user login and session management to maintain user privacy and safety.

Backend & Database

- Data Storage: Utilizes MongoDB for scalable data storage and Node.js with Express.js for backend handling.
- Efficient Data Handling: Fast and reliable fetching of data using modern API design principles, ensuring quick responses and reduced latency.

Real-Time Data Handling

- Real-Time Monitoring: Hospital stats, doctor appointments, and resource availability are monitored and updated in real-time.

Feedback and Communication

- Feedback Collection: Allows collection of feedback for hospitals and services, helping improve the quality of healthcare provided.
- Instant Communication: Quick access to communication features, especially for urgent requests, through chat or call.

The system is designed with the intention of providing a comprehensive and efficient solution for managing healthcare resources, ensuring that both the administrative staff and medical professionals have easy access to essential data. The **Admin Dashboard** facilitates a centralized view of hospital operations, from resource management to real-time monitoring of key metrics such as available beds and patient distribution. The **Doctor Dashboard** offers a modern, user-friendly interface that enables doctors to efficiently manage appointments, monitor follow-ups, and access patient data. **Real-time updates** ensure that users always have the most current information available. **Security** is a top priority, with **role-based access control** and secure session management ensuring that sensitive data is accessible only to authorized personnel.

CHAPTER - 3

3.1 LANGUAGE USED

Frontend Development

- HTML, CSS, and JavaScript: All pages on the frontend are built using these technologies. The layout is responsive, interactive, and modern, leveraging tools like Bootstrap and Tailwind CSS for styling.
- Data Visualization:
 - Doughnut Charts for resource availability and filling status.
 - Bar Graphs for hospital metrics.
 - Pie Chart for patient distribution.
- TypeScript for Inventory Management: The inventory section is built using TypeScript, providing type safety and more efficient data management for handling dynamic resources.

Backend Development

- Python Flask: The backend is implemented with Python Flask, allowing for efficient route handling, API creation, and serving dynamic content.
- MongoDB Atlas: Both the primary data and inventory data are stored in MongoDB Atlas, ensuring secure, scalable, and cloud-hosted data management. This allows for real-time data fetching and updating. Data is dynamically fetched using Fetch in the frontend, interacting seamlessly with the Python Flask backend.

Inventory Management System

- TypeScript: Handles inventory logic both on the frontend and backend. This ensures smooth operations, such as inventory updates, checking stock levels, and managing medicine orders.
- MongoDB Atlas: The inventory data is stored and managed in the same MongoDB Atlas database as other project data, ensuring consistency and ease of access across different system components.

The project integrates frontend and backend systems smoothly using **HTML, CSS, JavaScript, TypeScript, Python Flask**, and **MongoDB Atlas**, ensuring a modern, secure, and scalable healthcare management solution.

3.2 Library Used

Flask:

- The core framework for building the backend, handling routes, sessions, and rendering templates.
- Flask-Bcrypt: Used for securely hashing passwords and managing authentication.
- Flask-Session: To manage user sessions and ensure secure login/logout functionality.

MongoDB and Pymongo:

- The project leverages MongoDB Atlas for scalable cloud data storage.
- Pymongo is used for interacting with the MongoDB collections (such as appointment_collection, patients_collection, feedback_collection, etc.) efficiently.

Report Lab:

- For generating PDF reports, used to create professional and customizable reports (such as appointment details or patient data) through the SimpleDocTemplate class.
- QR Code Generation: Using the qrcode library, the project generates QR codes for appointment details or patient identification, enhancing security and ease of access.
- Table Handling: TableStyle is used for creating structured tables in PDF reports, useful for displaying patient and hospital data.

Smtplib:

- Used for sending email notifications to users (such as appointment reminders or feedback confirmations).
- This feature can be extended to notify doctors and patients of important events (though it is currently implemented for feedback purposes).

Flask-Login (if used in future):

- Could be integrated for more advanced user authentication features (though the existing login mechanisms may already provide sufficient functionality).

Datetime:

- The **datetime** module is crucial for handling appointment scheduling, calculating time differences, and managing date-based operations, such as appointment reminders or report generation deadlines.

Flask-Mail:

- In the future, **Flask-Mail** could be used to send bulk or personalized emails (reminders, confirmations, feedback requests) to users within the system.

Flask-Blueprints:

- The application uses Blueprints to organize the code in a modular way, ensuring that each section (e.g., `doctors_dashboard`, `admin_dashboard`) is easy to maintain and scalable.

Authentication:

- **login_required**: Custom decorators to restrict access to authenticated users for specific routes, ensuring that only authorized personnel (admin, doctors) can access sensitive information like patient data, feedback, or inventory management.

Data Handling:

- Custom **db** module imports from `modules.db` like `doctors_collection`, `patients_collection`, and `feedback_collection` help in managing and querying MongoDB data effectively.
- Specific collections such as `hospital_data_collection` and `stock_collection` provide structured access to hospital-related and inventory data, streamlining data management for both frontend and backend.

Error Handling:

- The project is built with error-handling mechanisms using **Flash messages** to notify users of form submission errors, invalid data, or system failures.

Utility Libraries:

- **functools**: Used for creating decorators like **login_required**, streamlining access control across routes.
- **jsonify**: Used to create JSON responses for AJAX requests, enabling smooth communication between the frontend and backend.

The libraries chosen for both the backend and frontend of this project provide a robust, secure, and scalable foundation for the healthcare management system. **Flask** serves as the backbone of the application, enabling efficient route handling, user authentication, and seamless integration with the frontend. The use of **MongoDB Atlas** ensures that data storage is highly scalable, secure, and easily accessible, accommodating the growing data requirements of the project.

CHAPTER - 4

SYSTEM DESIGN & SPECIFICATION

High Level Design (HLD)

4.1 Project Model

Our hospital management system is designed to streamline hospital operations, improve patient care, and enhance overall efficiency. The system offers functionalities for multiple user roles, including **Patients**, **Doctors**, **Administrators**, and **Super Admins**, providing tailored access and operations based on their responsibilities. The system's key features focus on **appointment management**, **resource allocation**, **inventory tracking**, **feedback collection**, **telemedicine** capabilities, and **real-time data updates**.

1. Technologies Used:

- Frontend:
 - HTML, CSS, JavaScript for page design and interactivity.
 - Bootstrap and CSS for responsive, modern UI.
 - TypeScript for inventory management and ensuring data type safety.
- Backend:
 - Python Flask for server-side operations and routing.
 - MongoDB Atlas for scalable, cloud-based data storage.
 - Flask-Bcrypt for secure password hashing.
 - Flask-Session for secure user session management.
 - ReportLab for generating PDF reports.
 - Smtplib for sending email notifications.
- Libraries/Packages:
 - qrcode for generating QR codes.
 - functools for creating decorators, like login_required.
 - datetime for managing appointment timings and notifications.

2. User Roles and Permissions:

Patient:

- View and Book Appointments: Patients can browse available doctors and book appointments.
- Access Medical Records: Patients can view their health history, upcoming appointments, and reports.

- **Receive Notifications:** Get alerts about upcoming appointments and other health-related reminders.
- **Provide Feedback:** After treatment, patients can give feedback on their experience.

Doctor:

- **Manage Appointments:** Doctors can view, update, and manage their appointments and follow-ups.
- **View Patient Data:** Access detailed patient histories, including medical records and previous treatments.
- **Quick Communication:** Ability to initiate video calls and chats for consultations.
- **Generate Reports:** Doctors can generate and download patient data in PDF format.

Admin:

- **Hospital Management:** Admins can manage hospital beds, staff, and appointments.
- **Inventory Management:** Admins track medicine stocks and order supplies.
- **Real-Time Bed Availability:** View and manage the real-time status of hospital beds.
- **Feedback and Grievance Management:** Admins can view patient feedback and resolve any grievances.

Super Admin:

- **Full Access Control:** Super admins can configure the system, manage users, and oversee the overall operation.
- **Monitor System Health:** Super admins can monitor the status of hospital resources, staff, and operations.
- **Generate Reports:** Super admins can access and generate high-level system-wide reports, including performance analytics.

3. Core Functionalities:

Appointment Management:

- **Patient Appointments:** Patients can book, view, and modify appointments.
- **Doctor's Appointment Dashboard:** Doctors can manage their schedules and check patient details.
- **Notifications:** Automated reminders for appointments are sent to both doctors and patients.

Resource Management:

- **Hospital Resources:** Track and manage resources like beds, staff, and medical equipment.
- **Real-Time Updates:** Admins and patients can view bed availability in real-time.
- **Alerts:** Notifications are triggered when bed availability reaches critical levels, or resources are running low.

Inventory Management:

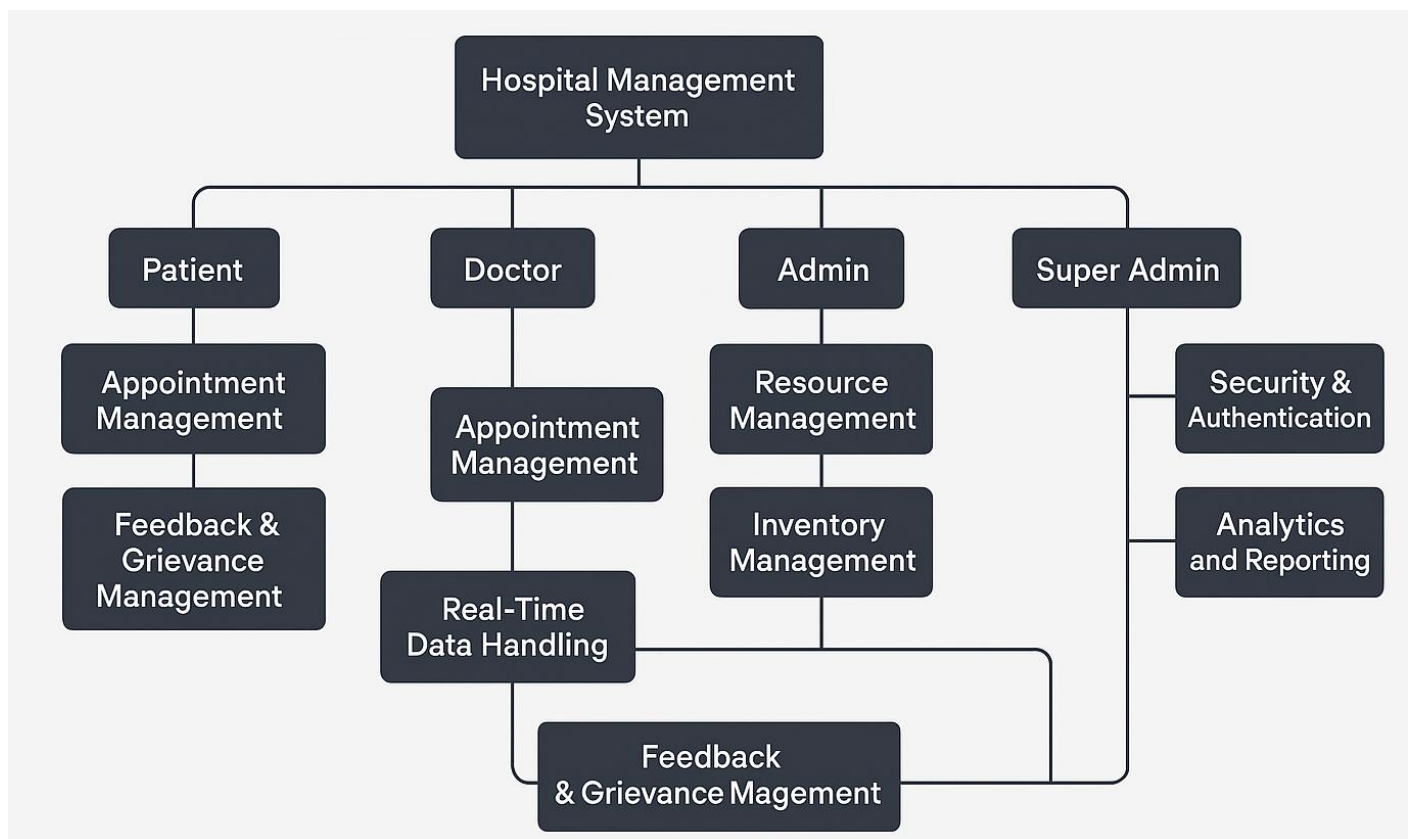
- **Track Inventory:** Manage stock levels for medicines and medical supplies.
- **Stock Alerts:** Receive notifications when inventory levels fall below a predefined threshold.
- **Distributor Integration:** Check availability and statuses from various inventory distributors.

Feedback & Grievance Management:

- **Patient Feedback:** Patients can submit feedback or complaints regarding their experiences.
- **Admin Review:** Admins can review feedback and resolve grievances to improve service quality.
- **Quality Monitoring:** Continuous tracking of feedback to identify areas for improvement.

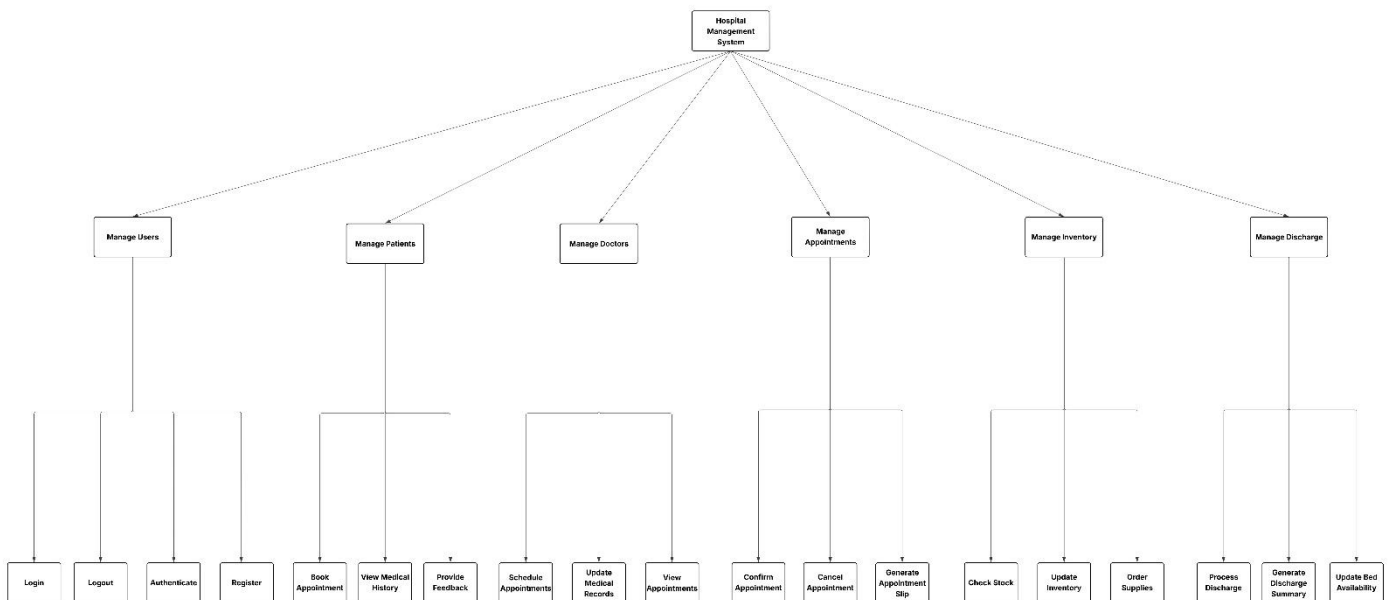
4. Analytics and Reporting:

- **Hospital Performance Metrics:** Generate reports on hospital performance, including bed occupancy, doctor availability, and patient inflow.
- **Patient Data Reports:** Generate detailed PDFs with patient treatment history, prescriptions, and appointments.
- **Inventory Reports:** Track inventory usage and manage medicine stock levels through automated reports.



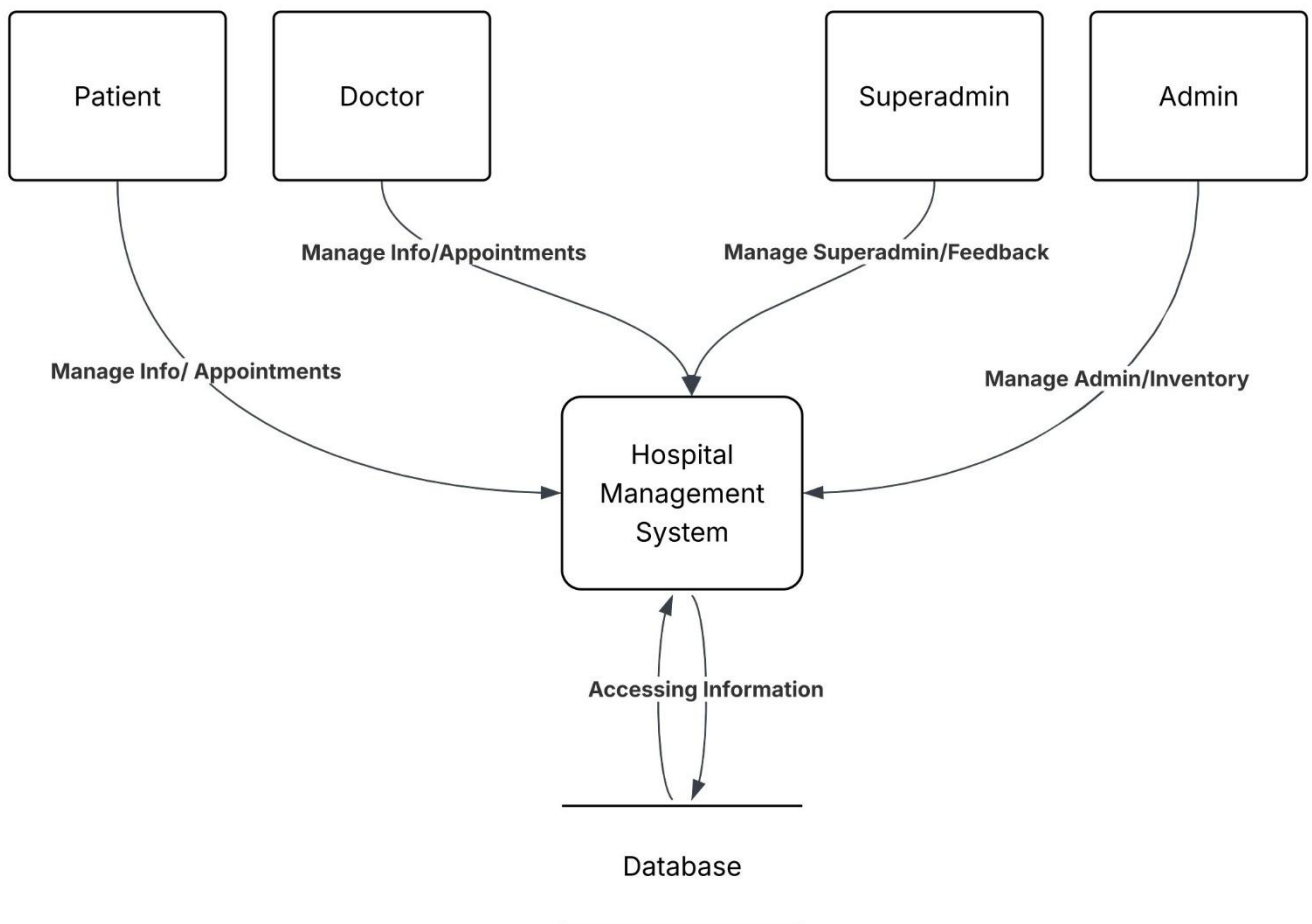
4.2 Structure Chart

Our project aims to develop a comprehensive and interactive healthcare management system for hospital administration and doctor's operations. It features a modern, user-friendly dashboard with key metrics like total hospitals, doctors, patients, and available resources, visually represented through dynamic charts like doughnut, bar, and pie charts. The system offers an intuitive sidebar for easy navigation, including options for adding hospitals, checking hospital status, managing medicine inventory, and handling feedback. Appointment management is enhanced with a card-based layout, replacing traditional tables for a more engaging user experience. Additionally, the doctor's dashboard includes metrics such as total appointments and patient trends, with integrated communication features for quick calls and video consultations. The backend, built with Flask and MongoDB Atlas, efficiently handles data management, user authentication, and report generation, ensuring seamless operation and data security across all modules, making it an effective solution for healthcare management.



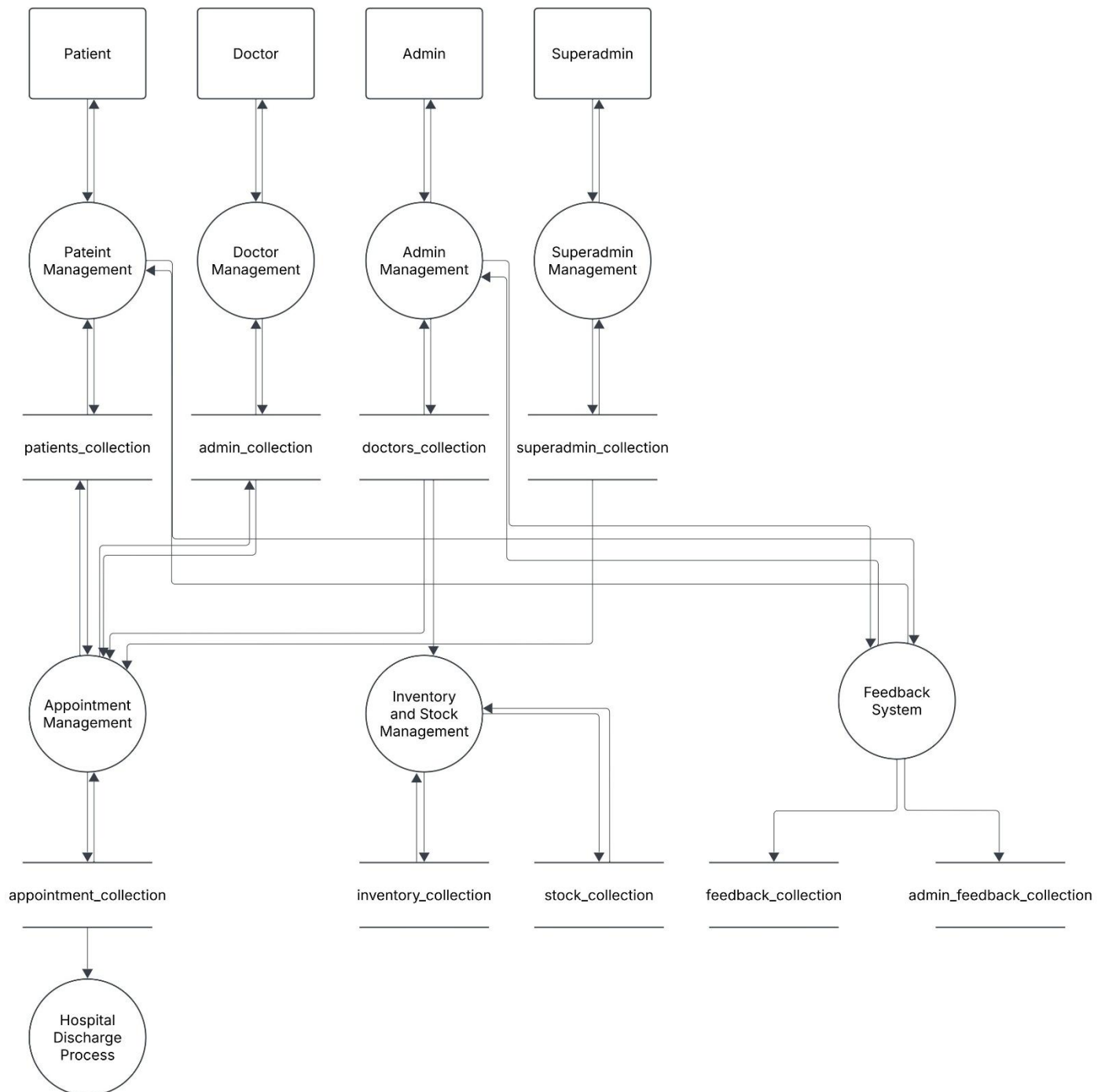
4.3 DFD Level 0

In DFD Level 0, the Hospital Management System is represented as a single, unified process that interacts with various external entities including Patients, Doctors, Admins, and Superadmins. These users provide and receive data such as appointments, personal details, inventory requests, and feedback. All data flows into a central process where it is managed and routed accordingly. A shared database (MongoDB) handles the secure storage and retrieval of information, enabling smooth communication between users and the system. This level provides a high-level overview of the entire system's data interaction without diving into internal processes.

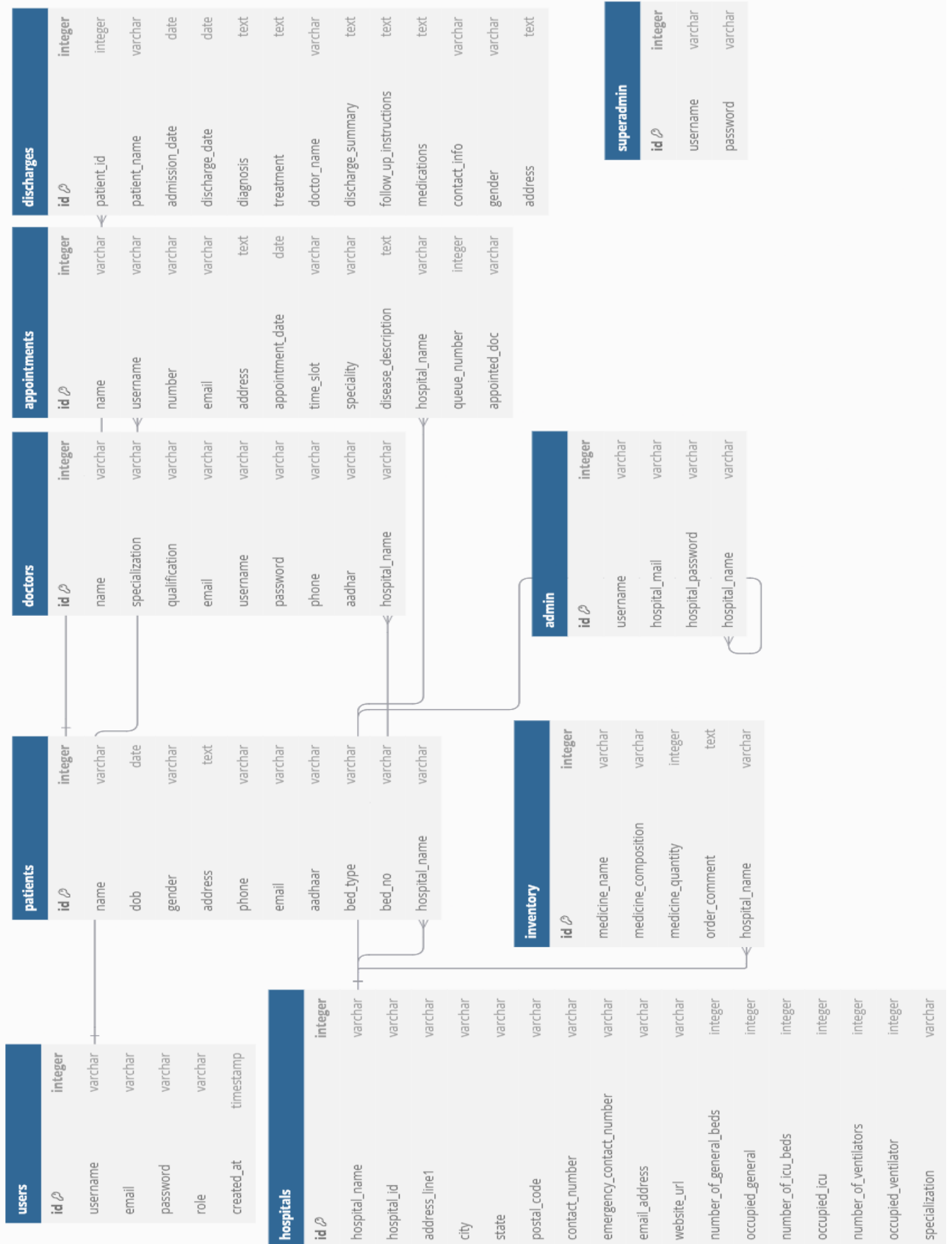


4.3 DFD Level – 1

In DFD Level 1, the system processes inputs like hospital data, appointments, and patient feedback. Admins and doctors interact with the system to manage resources, track hospital statuses, and view key metrics. Data is sent to the backend for validation and storage in MongoDB, ensuring secure and efficient data flow.



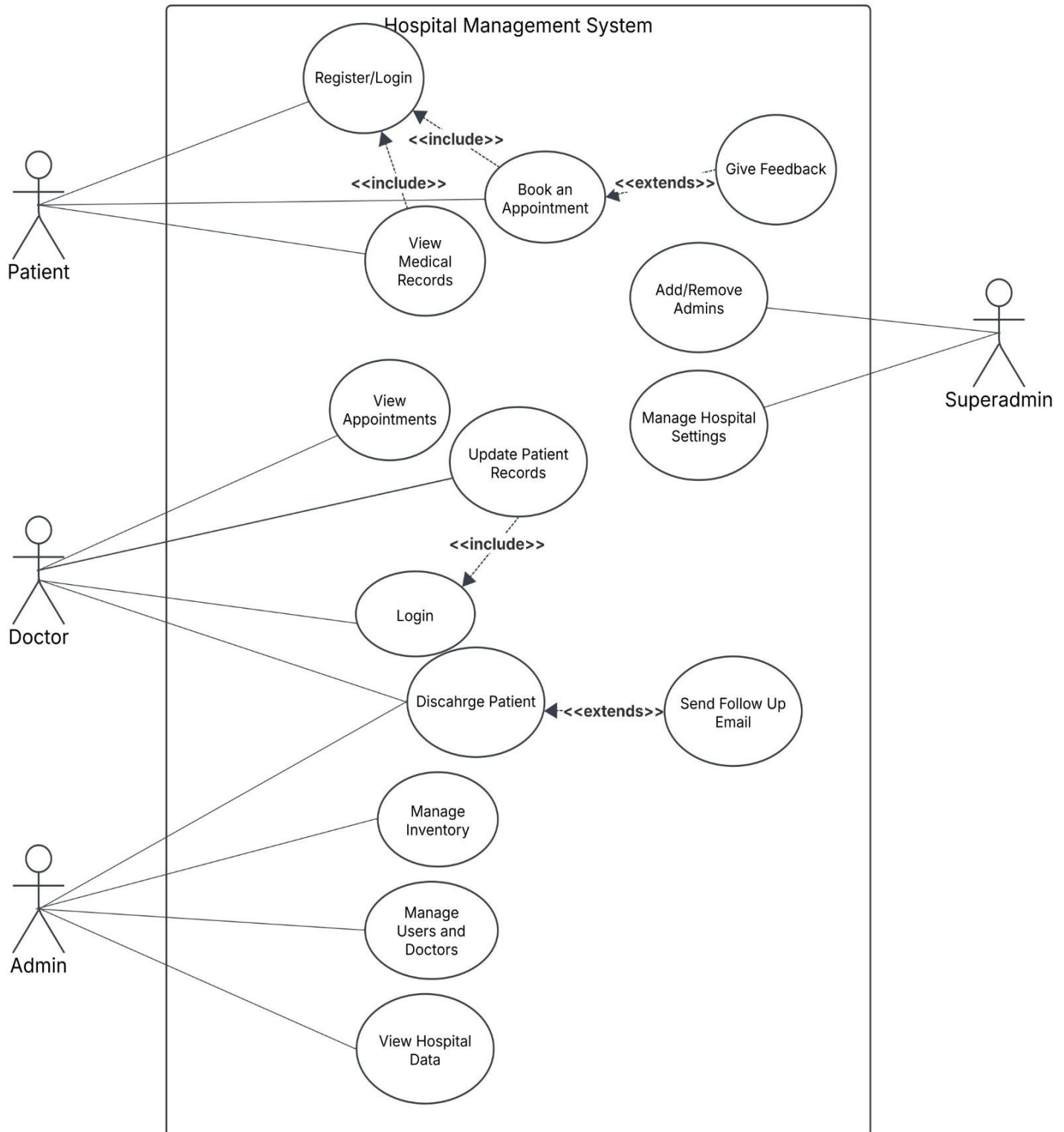
4.3 E-R Diagram



4.4 UML

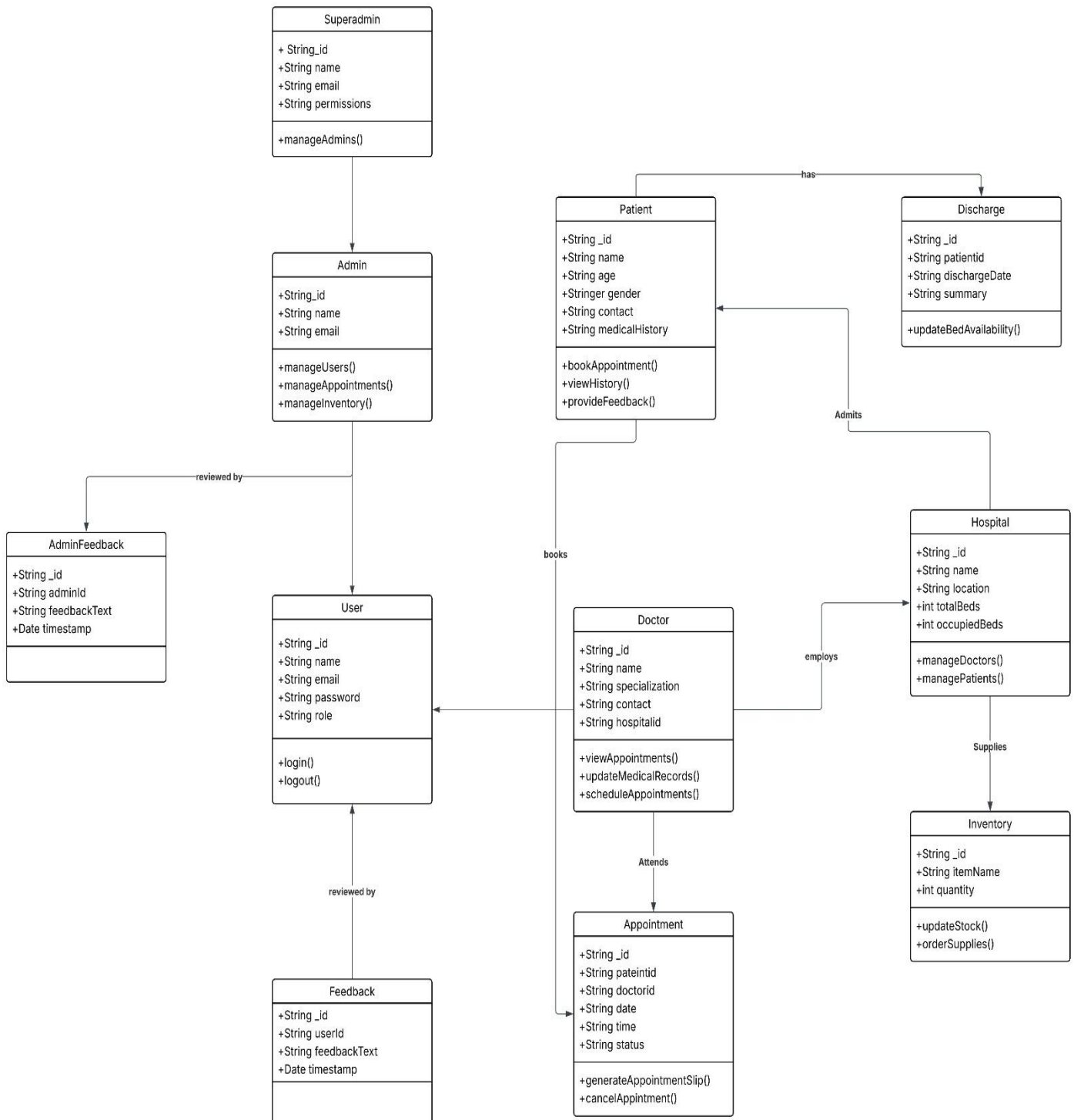
Use Case Diagram

The Use Case Diagram of our project highlights key user interactions with the system. Admins can manage hospital data, view metrics, and generate reports. Doctors can manage appointments, consult patients via calls/video, and access patient data. Patients can provide feedback, view appointment statuses, and communicate with healthcare professionals.



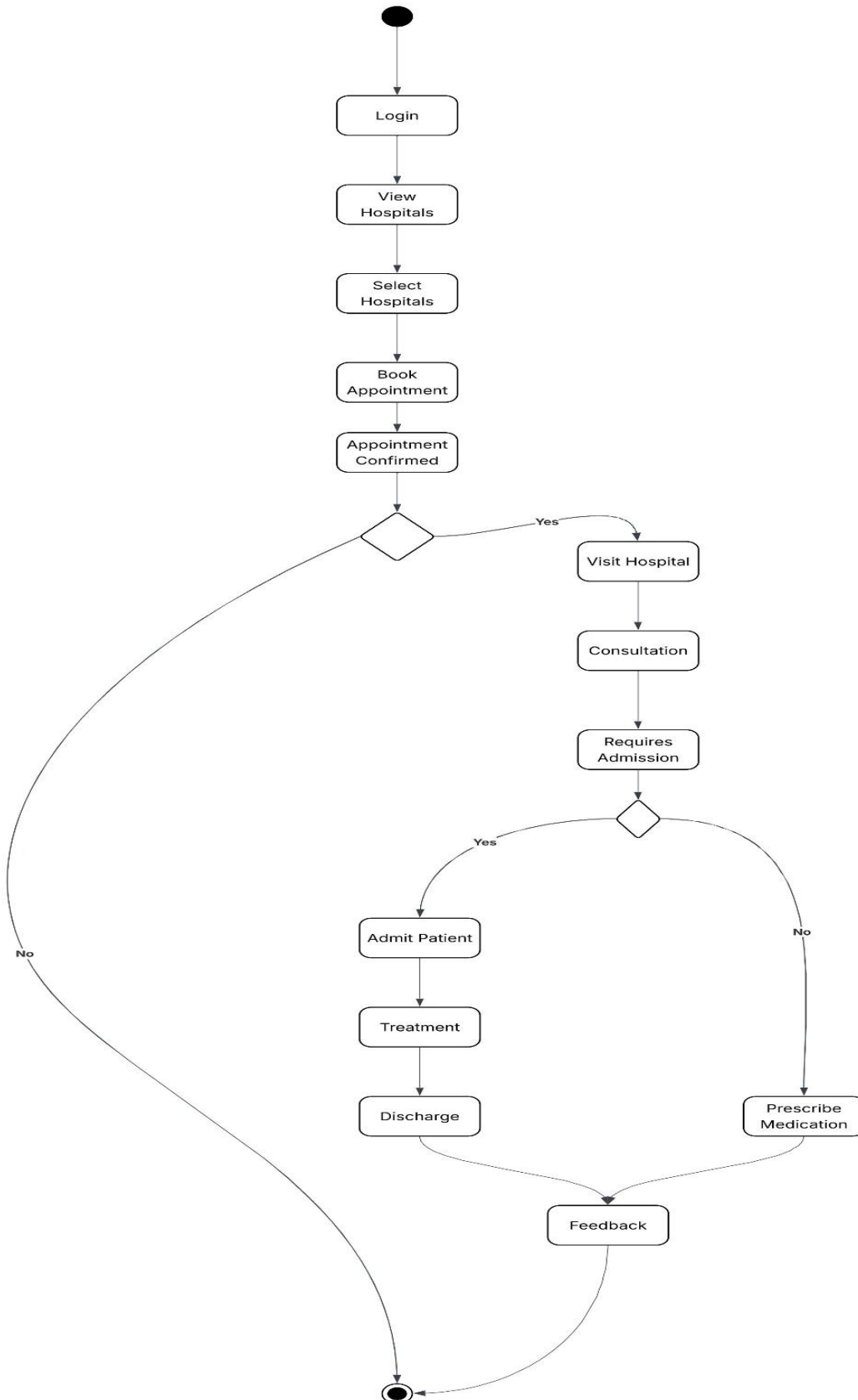
Class Diagram

The Class Diagram for the project defines key entities such as Hospital, Doctor, Patient, Appointment, and Inventory. Each class contains attributes like hospital details, doctor schedules, patient information, and appointment status. Methods include functionalities for managing appointments, generating reports, updating inventory, and handling user feedback, ensuring efficient data management.



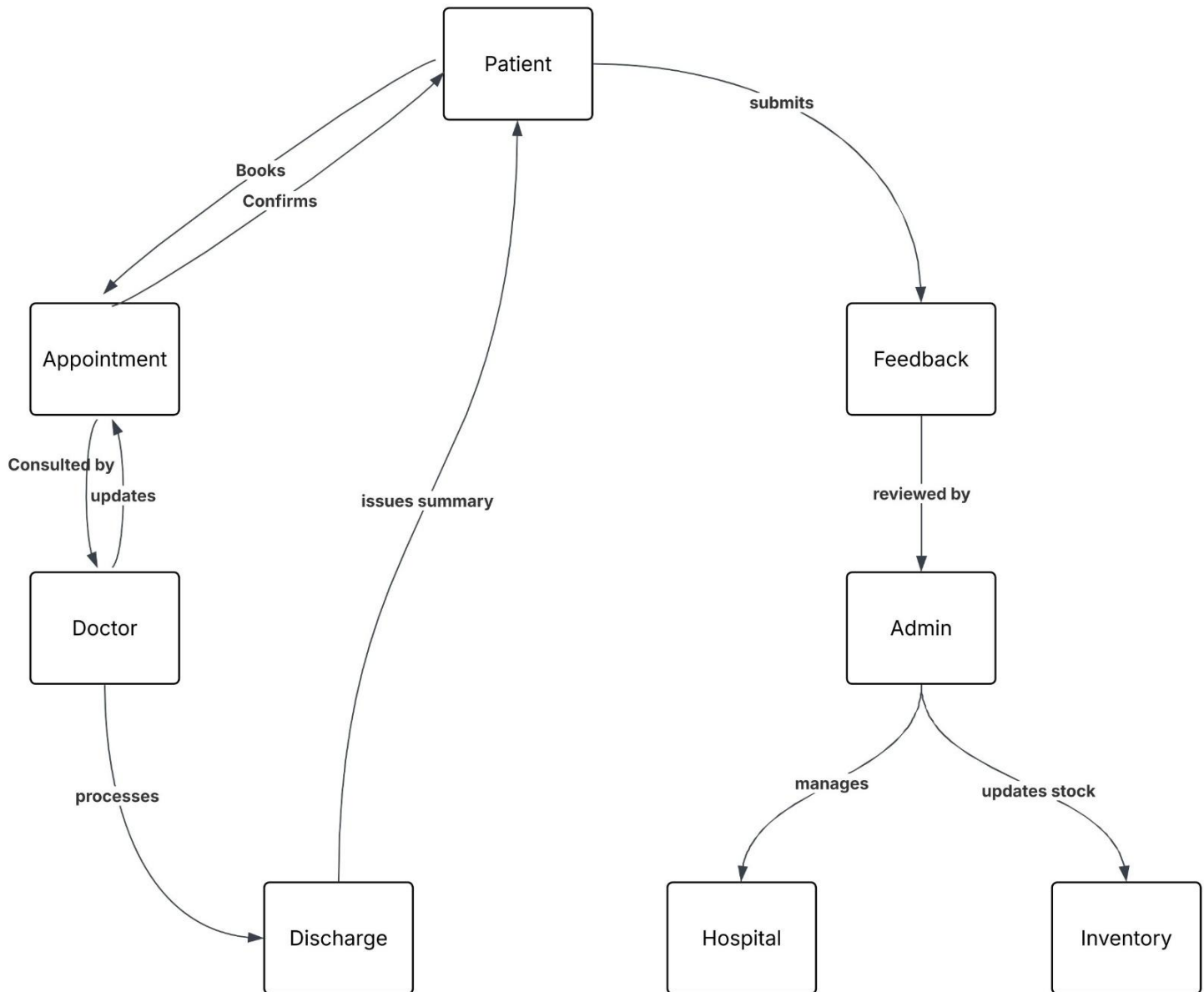
Activity Diagram

The Activity Diagram outlines the workflow for managing appointments and hospital data. Users interact with the system to add hospitals, check statuses, or manage inventory. Doctors schedule and track appointments, while patient data is processed. Feedback is collected, and reports are generated, ensuring smooth data flow and interaction across modules.



Interaction/Collaboration Diagram

The system ensures smooth interaction between Patients, Doctors, Admins, and Super Admins through secure login and role-based access. Data flows between modules like appointment scheduling, patient records, and inventory management in real-time. Each user role collaborates via dashboards tailored to their specific functionalities, ensuring coordinated healthcare delivery.



4.1.1 Pseudo Code

START

// Initialization

Initialize Database Connection (MongoDB Atlas)

Initialize Flask Server

Initialize Frontend Components (HTML, CSS, JavaScript)

Load Dependencies (Bootstrap, Chart.js, FontAwesome)

// User Authentication

FUNCTION userLogin(email, password)

 VALIDATE email and password

 IF valid THEN

 GENERATE JWT Token

 REDIRECT to appropriate dashboard based on role

 ELSE

 DISPLAY "Invalid Credentials"

 ENDIF

END FUNCTION

FUNCTION userRegistration(userData)

 VALIDATE userData

 IF valid THEN

 STORE userData in Database

 DISPLAY "Registration Successful"

 ELSE

 DISPLAY "Error in Registration"

 ENDIF

END FUNCTION

// Super Admin Functions

FUNCTION superAdminDashboard()

 LOAD totalHospitals, totalDoctors, totalPatients, inventoryStatus

 DISPLAY Bar Graphs, Pie Charts, and Doughnut Charts

 ALLOW user role management

```

    ALLOW data export (PDF Reports)
END FUNCTION

// Admin Functions
FUNCTION adminDashboard()
    LOAD hospital statistics
    DISPLAY available beds, appointment list, doctor schedules
    MANAGE inventory (Add, Edit, Delete Medicines)
    HANDLE feedback system
END FUNCTION

FUNCTION addDoctor(doctorDetails)
    VALIDATE doctorDetails
    IF valid THEN
        STORE doctorDetails in Database
        DISPLAY "Doctor Added Successfully"
    ELSE
        DISPLAY "Error in Adding Doctor"
    ENDIF
END FUNCTION

FUNCTION addPatient(patientDetails)
    VALIDATE patientDetails
    IF valid THEN
        STORE patientDetails in Database
        DISPLAY "Patient Registered Successfully"
    ELSE
        DISPLAY "Error in Adding Patient"
    ENDIF
END FUNCTION

// Doctor Functions
FUNCTION doctorDashboard()
    LOAD assigned appointments
    VIEW patient records

```

```
    UPDATE patient status (Consulted, Follow-up Needed)
    GENERATE patient reports (PDF Download)
END FUNCTION
```

```
FUNCTION manageAppointments()
    DISPLAY upcoming appointments
    ALLOW rescheduling or cancellations
    UPDATE appointment status
END FUNCTION
```

```
// Queue & Bed Management
```

```
FUNCTION priorityBasedQueue()
    SORT patients based on severity and appointment type
    DISPLAY optimized queue order
    NOTIFY doctors about critical cases
END FUNCTION
```

```
FUNCTION bedAvailability()
    LOAD total beds, occupied beds, available beds
    DISPLAY status in real-time
    NOTIFY admin if bed availability is low
END FUNCTION
```

```
// Inventory Management
```

```
FUNCTION manageInventory()
    DISPLAY available medicines and supplies
    ALLOW adding, updating, deleting inventory items
    NOTIFY when stock is low
END FUNCTION
```

```
// Feedback System
```

```
FUNCTION patientFeedback(feedbackData)
    VALIDATE feedbackData
    STORE feedback in database
    DISPLAY "Feedback Submitted Successfully"
```


END FUNCTION

// Notifications

```
FUNCTION sendNotifications(type, message)
    CHECK recipient type (Doctor, Admin, Patient)
    SEND notification via Email and Dashboard
END FUNCTION
```

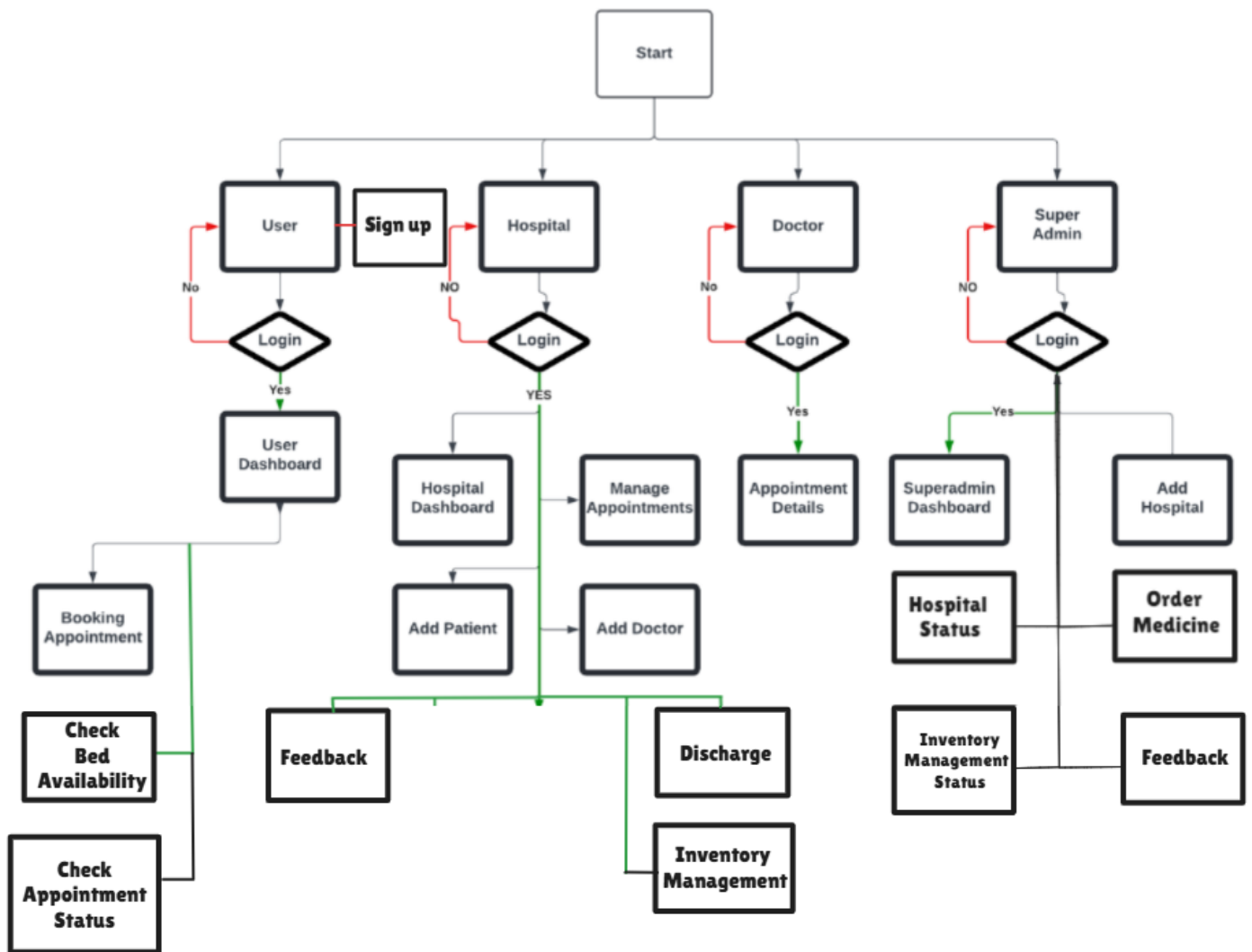
// Security & Authentication

```
FUNCTION validateJWT(token)
    IF token is valid THEN
        ALLOW access
    ELSE
        REDIRECT to login page
    ENDIF
END FUNCTION
```

// System Exit

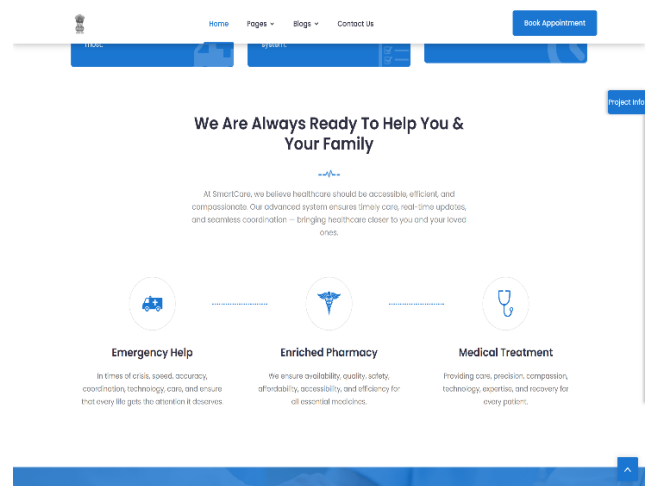
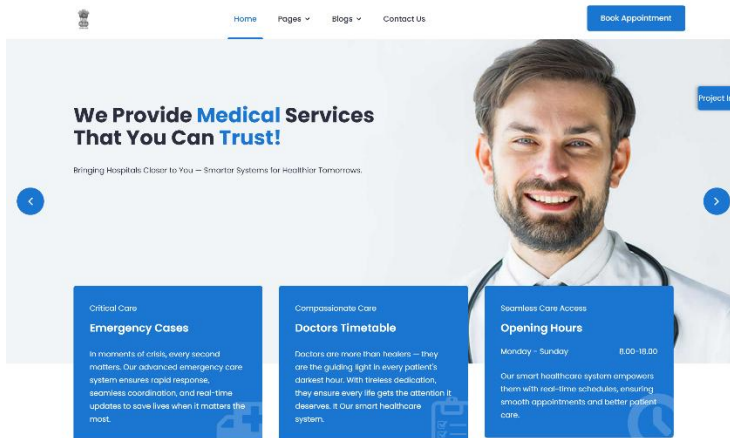
```
FUNCTION logout()
    INVALIDATE session token
    REDIRECT to login page
END FUNCTION
STOP
```

4.1.2 FLOW CHART

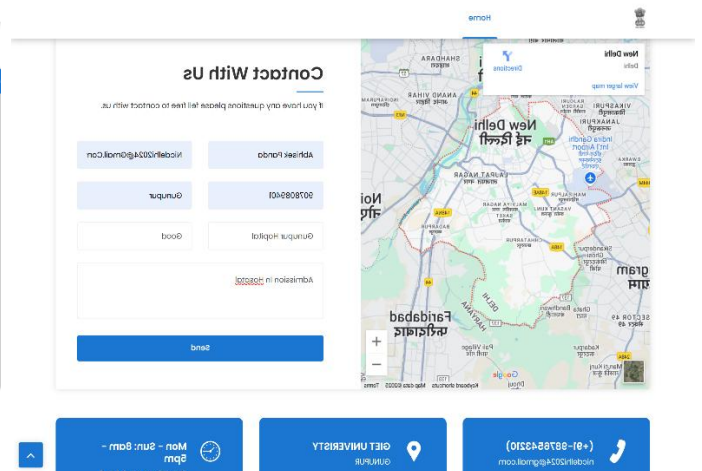
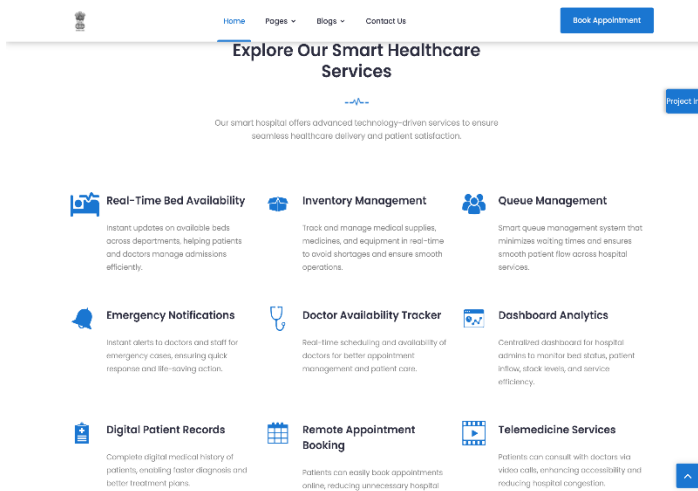


4.2.3 SCREEN SHOTS

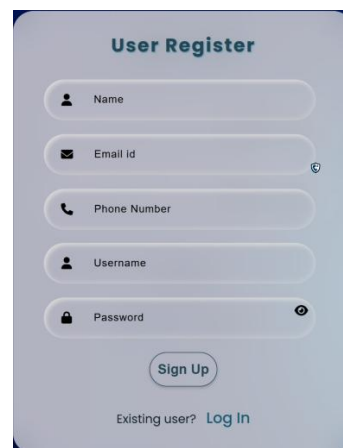
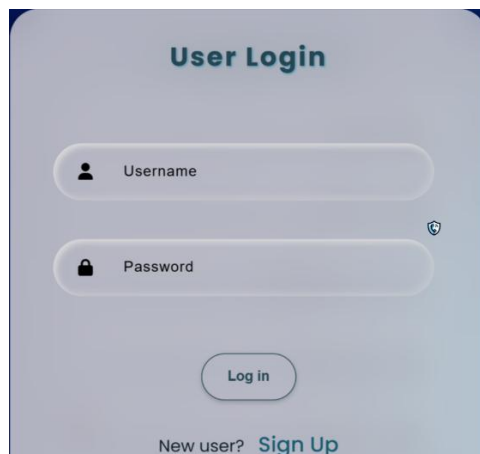
website View



Website Landing Page



Website Landing Page



User Login

User Sign Up

Welcome, **Abhisek Panda**

Appointment Details

Name: Abhisek Panda
Email: abhisek2004panda@gmail.com
Number: 9876578910
Appointment Date: 2025-03-22
Hospital Name: Gunupur Municipal
Time-slot: 08:00 - 09:00
Department: Neurology
Doctor Name: Sobhagya
Description: Back Pain
Queue Number: 1

Pending

Download PDF

Appointment Details

Name: Abhisek
Email: nicdelhi2024@gmail.com
Number: 9876578910
Appointment Date: 2025-03-21
Hospital Name: Gunupur Municipal
Time-slot: 08:00 - 09:00
Department: General Medicine
Doctor Name: Ayush ku
Description: Cold
Queue Number: 2

Pending

Download PDF

User DashBoard

Book Appointment

Full Name

Mobile Number

Email Address

Address

Appointment Date

Select Time Slot:

Select Hospital

User Booking App

Appointment Details

Name: Abhisek Panda
Email: abhisek2004panda@gmail.com
Number: 9876578910
Appointment Date: 2025-03-22
Hospital Name: Gunupur Municipal
Time-slot: 08:00 - 09:00
Department: Neurology
Doctor Name: Sobhagya
Description: Back Pain
Queue Number: 1

Bed Availability Dashboard

KIMS

Click to check

General Beds
200/300

Ventilator Beds
55/80

ICU Beds
100/150

User App Pdf

Bed Status

Doctor Login

Username

Password

Log in

Doctor Login

Welcome to the Doctor Dashboard

Doctor Details

Name: Sobhagya
Email: sobhagya@gmail.com
Phone Number: 7894561203
Aadhar: 74852109333
Hospital Name: Gunupur Municipal

Total Appointments
2

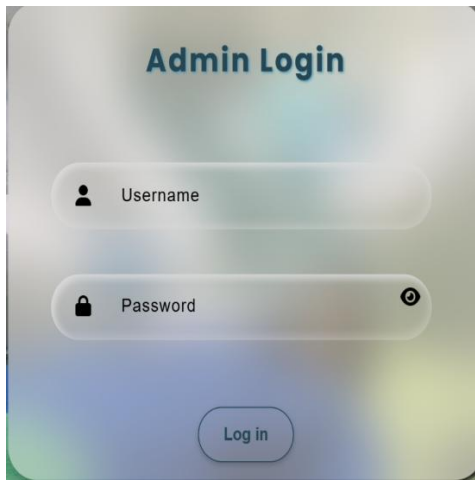
Patients Checked

Upcoming Appointments

Abhisek Panda
Email: abhisek2004panda@gmail.com
Phone: 9876578910
Date: 2025-03-22
Time: 08:00 - 09:00
Department: Neurology
Queue Number: 1

Call Video Consult Download PDF

Doctor Dashboard



Admin Login

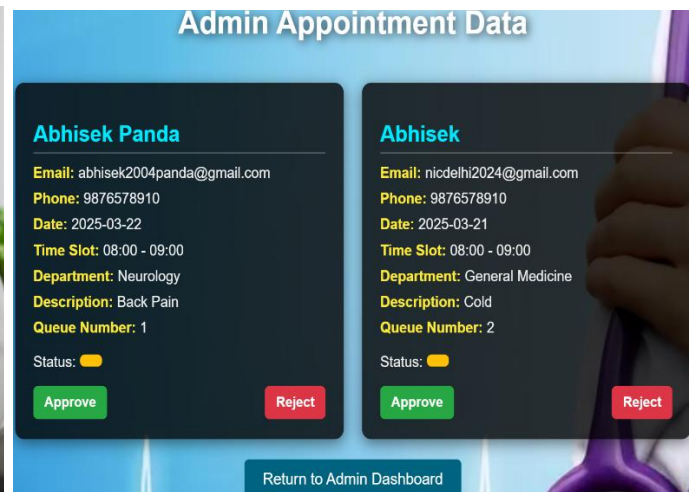
Admin Login



Admin Dashboard



Admin Dashboard



Admin Appointment Data

Abhisek Panda

Email: abhisek2004panda@gmail.com

Phone: 9876578910

Date: 2025-03-22

Time Slot: 08:00 - 09:00

Department: Neurology

Description: Back Pain

Queue Number: 1

Status: Pending

Abhisek

Email: nicdelhi2024@gmail.com

Phone: 9876578910

Date: 2025-03-21

Time Slot: 08:00 - 09:00

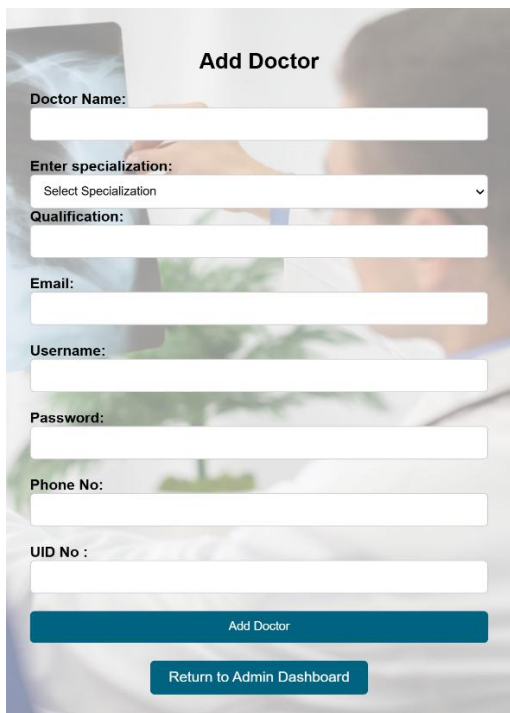
Department: General Medicine

Description: Cold

Queue Number: 2

Status: Pending

Admin Appointment Data



Add Doctor

Doctor Name:

Enter specialization:

Qualification:

Email:

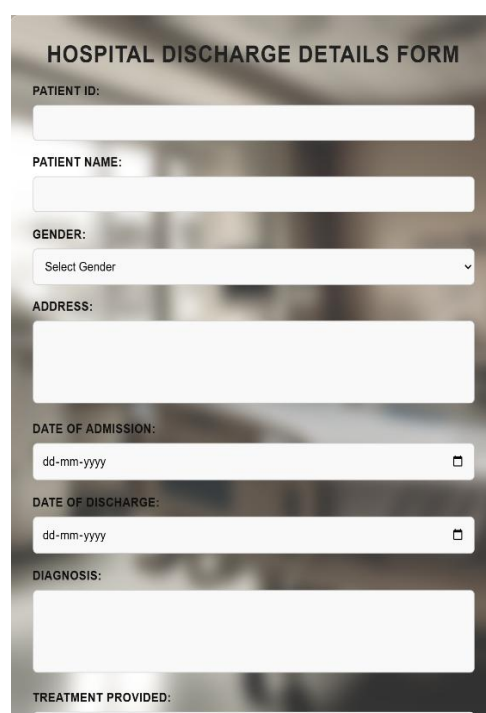
Username:

Password:

Phone No:

UID No :

Admin Add Doctor



HOSPITAL DISCHARGE DETAILS FORM

PATIENT ID:

PATIENT NAME:

GENDER:

ADDRESS:

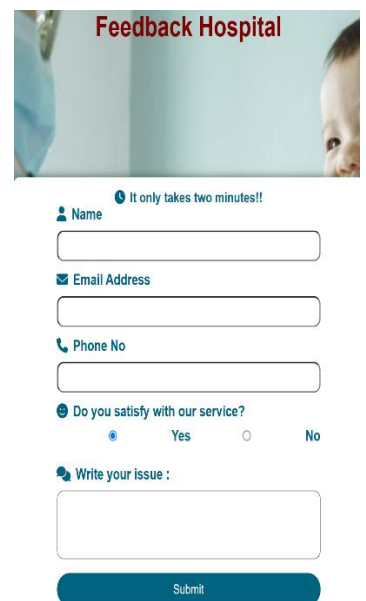
DATE OF ADMISSION:

DATE OF DISCHARGE:

DIAGNOSIS:

TREATMENT PROVIDED:

Hospital Patient Discharge



Feedback Hospital

It only takes two minutes!!

Name

Email Address

Phone No

Do you satisfy with our service?

☒ Yes
 ☐ No

Write your issue :

Admin Feedback

Gunupur Muncipal

Patient Discharge Summary

Patient Discharge Summary	
Patient Name	Abhisek Panda
Admission Date	2025-03-01
Discharge Date	2025-04-01
Diagnosis	Fever
Treatment	Yes
Doctor Name	Sourav
Discharge Summary	All Ok And Good
Follow-Up Instructions	Take healthy food
Medications	No
Contact Info	9874102550
Gender	male
Address	Giet
Bed Type	icu
_id	67e2c8364272ecb9c1f72d21

Health & Wellness Tips for Recovery:

Drink Plenty of Water – Stay hydrated to help your body recover faster.
 Take a Bath Daily – Maintain good hygiene to prevent infections.
 Eat an Apple Every Day – ‘An apple a day keeps the doctor away!’
 Consume a Balanced Diet – Include proteins, vitamins, and minerals.
 Avoid Junk Food – Say NO to excessive sugar, salt, and oily foods.
 Take Your Medicines on Time – Follow the prescribed dosage carefully.
 Get Enough Rest & Sleep – Allow your body to heal and regain energy.
 Avoid Smoking & Alcohol – These slow down recovery and harm your health.
 Do Light Exercise – Gentle movements help in faster recovery.
 Keep Your Surroundings Clean – Prevent infections and maintain hygiene.
 Regularly Change Wound Dressings – If applicable, as per doctor's advice.
 Follow Your Doctor's Instructions – Always stick to medical advice.
 Keep Emergency Contacts Handy – Save the hospital and doctor's numbers.
 Wash Hands Frequently – Avoid germs and stay safe.
 Monitor Your Symptoms – Report any unusual pain, fever, or discomfort.
 Attend All Follow-Up Appointments – Ensure complete recovery.
 Stay Positive & Stress-Free – Mental health is just as important.

Gunupur Muncipal

ID: HMSDL001

Hospital Name:

Gunupur Muncipal

Address:

Gunupur, Gunupur, Odisha - 765022

Contact Number:

9852678540

Emergency Contact:

9852678540

Email:

nicdelhi2024@gmail.com

Website:

Number of Beds:

100

General Beds Occupied:

3

Super Admin Login

Username

Password

Log in

Patient Discharge Pdf

Hospital Data

Super Admin Login



Super Admin Dashboard

Add New Hospital

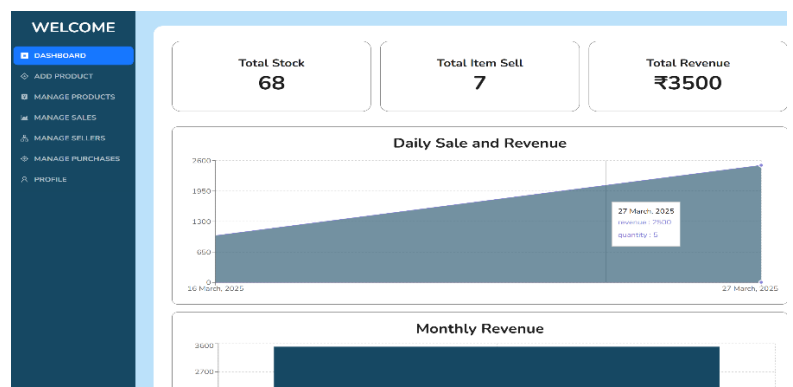
Hospital Name:

Hospital Email ID:

Password:

Add Hospital Return to Dashboard

Super Admin Add Hospital



Inventory management Dashboard

WELCOME

DASHBOARD

ADD PRODUCT

MANAGE PRODUCTS

MANAGE SALES

MANAGE SELLERS

MANAGE PURCHASES

PROFILE

ADD NEW PRODUCT

Name

Adrenaline

Price

500

Stock

100

Seller

Jp Traders

Category

Antiviral

Brand

Cipro

Description

Medicine

Size

Large

ADD PRODUCT

CREATE NEW SELLER

CREATE SELLER

CREATE NEW CATEGORY

Category Name

CREATE CATEGORY

CREATE NEW BRAND

Brand Name

CREATE BRAND

Add Product , Create Seller , New Category , New Brand

WELCOME

DASHBOARD

ADD PRODUCT

MANAGE PRODUCTS

MANAGE SALES

MANAGE SELLERS

Price Range

Search by product name

Search by Product Name

Filter by Category

Filter by Category

Filter by Brand

Filter by Brand

Product Name	Category	price	stock	Purchase From	Action
Adrenaline	Antiviral	500	100	Jp Traders	Sell Add Stock
Nitroglycerin	Antibiotics	500	68	Chaka Dolla	Sell Add Stock

Manage Product

DASHBOARD

ADD PRODUCT

MANAGE PRODUCTS

MANAGE SALES

MANAGE SELLERS

Search Sold Products...

Product Name	Product Price	Buyer Name	Quantity	Total Price	Selling Date	Action
Nitroglycerin	500	Diptesh	50	25000	27 Mar, 2025	
Nitroglycerin	500	Deba	5	2500	27 Mar, 2025	
Nitroglycerin	500	Abhi	2	1000	16 Mar, 2025	

Manage Sales

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 118KB TOTAL DOCUMENTS: 5

Find Indexes Schema Anti-Patterns Aggregations

Generate queries from natural language in Compass

Filter Type a query: { field: 'value' }

```

{
  "product": {
    "createdAt": "2025-03-26T17:33:24.498+00:00",
    "updatedAt": "2025-03-26T17:33:24.498+00:00",
    "_id": "67d57113601b89b9c268ab71"
  },
  "seller": {
    "sellerName": "Chaka Dolla",
    "productName": "Nitroglycerin",
    "quantity": 100,
    "unitPrice": 500,
    "totalPrice": 50000,
    "paid": 0,
    "createdAt": "2025-03-26T17:33:46.897+00:00",
    "updatedAt": "2025-03-26T17:33:46.897+00:00",
    "_id": "67d56ef4601b89b9c268ab57"
  },
  "user": {
    "user": "Ashwini Hospital",
    "hospital_id": "HMS_BBSR_002",
    "address_line1": "Bhubaneswar",
    "city": "Khordha",
    "state": "Odisha",
    "postal_code": "765022",
    "contact_number": "9854878540",
    "emergency_contact_number": "8520147960",
    "email_address": "ashwini@gmail.com",
    "website_url": "",
    "number_of_general_beds": 300,
    "occupied_general": 202,
    "number_of_icu_beds": 165,
    "occupied_icu": 100,
    "number_of Ventilators": 100,
    "occupied Ventilator": 50,
    "emergency_department": "Yes",
    "specialization": Array (6),
    "hospital_operating_hours": "24",
    "visiting_hours": "12",
    "pharmacy_on_site": "Yes",
    "total_number_of_nurses": 50,
    "administrative_staff_count": 40
  }
}
```

```

_id: ObjectId('67e43a7a88ad2d210881541e')
hospital_name: "Ashwini Hospital"
hospital_id: "HMS_BBSR_002"
address_line1: "Bhubaneswar"
city: "Khordha"
state: "Odisha"
postal_code: "765022"
contact_number: "9854878540"
emergency_contact_number: "8520147960"
email_address: "ashwini@gmail.com"
website_url: ""
number_of_general_beds: 300
occupied_general: 202
number_of_icu_beds: 165
occupied_icu: 100
number_of Ventilators: 100
occupied Ventilator: 50
emergency_department: "Yes"
specialization: Array (6)
hospital_operating_hours: "24"
visiting_hours: "12"
pharmacy_on_site: "Yes"
total_number_of_nurses: 50
administrative_staff_count: 40
```

```

_id: ObjectId('67e4359c53d53fd4bf3e47cc')
name: "Abhisek Panda"
username: "abhisek72"
number: "9078089401"
email: "abhisek2004panda@gmail.com"
address: "Giet"
appointment_date: "2025-03-27"
time_slot: "08:00 - 09:00"
speciality: "General Medicine"
disease_description: "Cold , Fever"
hospital_name: "Gunupur Muncipal"
queue_number: 1
appointed_doc: "Ayush ku"
```

Mongo Atlas

CHAPTER - 5

PROJECT CODING

Backend Code

```
from flask_bcrypt import Bcrypt

from flask import Flask, render_template, request, redirect

from modules.log_out import logout_bp

from modules.admin import admin_blueprint

from modules.superadmin import superadmin_blueprint

from modules.doctor import doctor_blueprint

from modules.db import contact_collection, feedback_collection

from modules.user import user_blueprint

app = Flask(__name__)

app.register_blueprint(logout_bp)

app.register_blueprint(admin_blueprint)

app.register_blueprint(doctor_blueprint)

app.register_blueprint(superadmin_blueprint)

app.register_blueprint(user_blueprint)

bcrypt = Bcrypt()

app.secret_key = "anything_secret_is_good"

@app.route('/', methods=['GET'])

def home():

    return render_template('index.html')
```



```
@app.route('/contact', methods=['GET', 'POST'])

def landing():

    if request.method == 'POST':

        name = request.form['name']

        email = request.form['email']

        number = request.form['phone']

        city = request.form['city']

        nearest_hospital = request.form['nearest_hospital']

        subject = request.form['subject']

        message = request.form['message']

        contact_data = {

            'name': name,

            'email': email,

            'number': number,

            'city': city,

            'nearest_hospital': nearest_hospital,

            'message': message,

            'subject': subject

        }

        contact_collection.insert_one(contact_data)

    return render_template('contact.html')
```

```
@app.route('/all_doctor')

def all_doc():

    return render_template('doctor.html')

@app.route('/confirmation', methods=['POST', 'GET'])

def confirmation():

    return render_template('conformation.html')

@app.route('/stock_detail')

def detail():

    return render_template('inv_stock_product.html')

@app.route('/home_feedback', methods=['POST'])

def feedback():

    name = request.form['name']

    email = request.form['email']

    city = request.form['city']

    nearest_hospital = request.form['nearest_hospital']

    message = request.form['message']

    data = {

        'name': name,

        'email': email,

        'city': city,

        'nearest_hospital': nearest_hospital,
```

```

        'message': message
    }

    feedback_collection.insert_one(data)

    return redirect('/')

@app.route('/blog', methods=['GET', 'POST'])

def blog_single():

    return render_template('blog-single.html')

# a customized error handler

@app.errorhandler(404)

def page_not_found(error):

    return render_template('404.html'), 404

if __name__ == '__main__':

    app.run(port=8000, debug=True)

```

Frontend code

```

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>User Dashboard</title>

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css">

    <script src="https://cdnjs.cloudflare.com/ajax/libs/jspdf/2.4.0/jspdf.umd.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/html2canvas/1.4.1/html2canvas.min.js"></script>

```

<style>

```
* { margin: 0; padding: 0; box-sizing: border-box; font-family: Arial, sans-serif; }
```

```
body { display: flex; }
```

```
.sidebar { width: 250px; background: #343a40; color: white; height: 100vh; padding-top: 20px; transition: width 0.3s; }
```

```
.sidebar.collapsed { width: 80px; }
```

```
.sidebar .logo { text-align: center; margin-bottom: 20px; font-size: 1.2em; }
```

```
.sidebar ul { list-style: none; padding: 0; }
```

```
.sidebar ul li { padding: 15px 20px; cursor: pointer; transition: background 0.3s; }
```

```
.sidebar ul li:hover { background: #495057; }
```

```
.toggle-btn { position: absolute; top: 15px; left: 260px; font-size: 20px; cursor: pointer; }
```

```
.main-content { flex: 1; padding: 20px; transition: margin-left 0.3s; }
```

```
.sidebar.collapsed ~ .main-content { margin-left: 80px; }
```

```
.appointments { display: flex; flex-wrap: wrap; gap: 20px; }
```

```
.card { width: 300px; background: white; padding: 15px; border-radius: 8px; box-shadow: 0 0 10px rgba(0, 0, 0, 0.1); }
```

```
.card h3 { margin-bottom: 10px; }
```

```
.download-btn { margin-top: 10px; padding: 8px 12px; background: #007bff; color: white; border: none; cursor: pointer; border-radius: 5px; }
```

</style>

</head>

<body>

<div class="sidebar">

<div class="logo">Dashboard</div>

Home

Appointments

```

    <li>Settings</li>

</ul>

</div>

<div class="toggle-btn" onclick="toggleSidebar()">≡</div>

<div class="main-content">

    <h2>My Appointments</h2>

    <div class="appointments">

        <div class="card" id="card-1">

            <h3>Dr. John Doe</h3>

            <p>Date: 2025-04-10</p>

            <p>Time: 10:30 AM</p>

            <p>Location: City Hospital</p>

            <button class="download-btn" onclick="downloadPDF(1)">Download PDF</button>

        </div>

    </div>

</div>

</div>

<script>

function toggleSidebar() {

    const sidebar = document.querySelector(".sidebar");

    sidebar.classList.toggle("collapsed");

    localStorage.setItem("sidebarState", sidebar.classList.contains("collapsed") ? "collapsed" : "expanded");

}

document.addEventListener("DOMContentLoaded", () => {

    if (localStorage.getItem("sidebarState") === "collapsed") } </body></html>

```

CHATER - 6

TESTING

Testing is a critical phase in the development our project **Hospital Management System**, ensuring that each module functions correctly, meets security standards, and performs efficiently under various conditions. Multiple testing methodologies, including **Unit Testing, Integration Testing, Security Testing, Performance Testing, and User Acceptance Testing (UAT)**, were performed to validate the system's functionality.

1. Unit Testing

Unit testing was performed to verify the accuracy and functionality of **individual components** of the system before integrating them. Each module was tested independently to identify and fix early-stage errors.

Testing Performed:

- **Authentication & Authorization:** Ensured that only registered users could log in and role-based permissions worked correctly.
- **Patient Registration:** Verified form validation for correct data entry, error handling, and successful database storage.
- **Appointment Scheduling:** Checked that patients could book, reschedule, and cancel appointments without issues.
- **Inventory Management:** Tested the addition, modification, and deletion of inventory items (medicines, supplies).

Results: Bugs related to incorrect form validation and session expiration were fixed before module integration.

2. Integration Testing

Integration testing was performed to ensure that different modules worked together seamlessly. The **interactions between the Super Admin, Admin, Doctor, and Patient dashboards** were tested.

Testing Performed:

- **Database Connectivity:** Verified that all changes made by users were correctly updated in MongoDB Atlas.
- **API Functionality:** Ensured smooth data exchange between the frontend (HTML, CSS, JavaScript) and backend (Flask).
- **Real-Time Bed Availability:** Checked that bed status updates reflected correctly across all dashboards.

- **Notification & Alerts:** Ensured users received appointment confirmations, inventory alerts, and system notifications.

Results: Identified and resolved synchronization issues between **bed availability, appointment updates, and notification triggers**.

3. Security Testing

Since the system handles sensitive patient data, security testing was performed to **prevent data breaches, unauthorized access, and cyber threats**.

Testing Performed:

- **Role-Based Access Control:** Checked that patients, doctors, admins, and super admins could only access permitted sections.
- **Data Encryption:** Confirmed that sensitive data (patient records, medical history) was encrypted before storage.

Results: Strengthened security by **fixing API vulnerabilities, implementing stricter access policies, and enhancing data encryption**.

4. Performance Testing

Performance testing was conducted to check system stability under **high user load and simultaneous requests**.

Testing Performed:

- **Load Testing:** Simulated multiple concurrent users booking appointments and accessing dashboards.
- **Database Query Optimization:** Ensured MongoDB queries executed efficiently under high traffic.
- **Dashboard Loading Speed:** Tested how fast graphs, statistics, and hospital metrics were displayed.
- **API Response Time:** Measured how quickly data was fetched and displayed on the frontend.

Results: Improved response times by **optimizing database queries and reducing unnecessary API calls**

5. User Acceptance Testing (UAT)

User acceptance testing was conducted to ensure that the system met user expectations in **usability, accessibility, and functionality**.

Testing Performed:

- **Admin & Super Admin Dashboards:** Verified that they could **add users, manage hospital data, and view statistics**.

- **Doctor Portal:** Checked if doctors could **access patient records, update medical history, and schedule follow-ups.**
- **Patient Portal:** Ensured **patients could book appointments, view hospital services, and receive notifications.**
- **Inventory Management:** Tested **medicine stock updates, order placement, and low-stock alerts.**
- **Mobile Responsiveness:** Ensured the system worked on **various screen sizes (desktop, tablet, mobile).**

Results: UI/UX refinements were made based on feedback, improving **navigation, accessibility, and responsiveness.**

6. Regression Testing

Regression testing was conducted after each **bug fix or system update** to ensure no new issues were introduced.

Testing Performed:

- **Re-tested all authentication flows** after improving login security.
- **Verified dashboard functionalities** after modifying graph display mechanisms.
- **Checked all appointment booking features** after updating the priority-based queue system.

Results: Continuous testing ensured that **new updates did not break existing functionalities.**

7. Bug Fixing & System Improvements

After completing the testing phase, several **bugs were fixed, and system enhancements were made.**

Key Fixes & Enhancements:

- **Optimized inventory tracking** to reduce API load time.
- **Fixed broken UI components** for smoother user experience.
- **Enhanced error messages** to guide users effectively.

Final Outcome: The system is now **secure, efficient, and user-friendly**, with further enhancements planned for **automation, AI-based queue management, and predictive analytics.**

CHAPTER - 7

CONCLUSION & LIMITATION

The Hospital Management System was developed to optimize and streamline various hospital operations, including patient registration, appointment scheduling, bed availability tracking, inventory management, and real-time analytics. By integrating a secure and scalable architecture, the system ensures efficient management of healthcare resources, reducing manual work and improving coordination among doctors, administrators, and patients.

Throughout the development process, several key functionalities were successfully implemented. The system was incorporated to provide secure authentication for different user roles, ensuring that only authorized personnel could access sensitive medical records and system functionalities. Additionally, the priority-based queue management system allows hospitals to dynamically sort patient appointments based on severity, reducing waiting times and improving emergency handling.

For hospital inventory management, a dedicated module was implemented, enabling administrators to monitor stock levels, receive automated alerts for low supplies, and efficiently manage medicine distribution. Super admin dashboards were designed to provide a centralized view of all hospital activities, including real-time graphs and reports for decision-making. The doctor dashboard allows medical professionals to access patient records, manage appointments, and track follow-ups seamlessly.

To enhance patient experience, the system includes an interactive feedback mechanism, allowing hospitals to assess patient satisfaction and make improvements where necessary. Real-time notifications and alerts were integrated for appointment reminders, emergency updates, and inventory restocking. The bed availability tracking module was also designed to assist hospital staff in effectively managing admissions and ensuring that patients receive timely medical attention.

Our Hospital Management System was designed and developed to optimize hospital operations, improve patient care, and streamline administrative tasks through a centralized, digital platform. The system successfully integrates appointment scheduling, bed availability tracking, patient records management, inventory control, and real-time analytics, ensuring seamless coordination among patients, doctors, administrators, and super admins.

During the development process, various key modules were implemented, including secure authentication (JWT-based login), role-based access control, real-time notifications, automated appointment management, and priority-based queue handling. The system also enhances hospital resource management by efficiently tracking inventory and sending alerts for low stock, ensuring continuous availability of essential medical supplies.

CHAPTER - 8

REFERENCE

Bibliography

The development of this project **Hospital Management System** was supported by various online resources, technologies, and documentation. The following references were utilized throughout the project lifecycle:

Official Documentation & Development Resources

- MDN Web Docs – JavaScript, HTML, and CSS best practices.
 - <https://developer.mozilla.org/>
- Bootstrap Documentation – UI components and responsive design.
 - <https://getbootstrap.com/docs/>
- Flask Official Documentation – Backend development with Python Flask.
 - <https://flask.palletsprojects.com/>
- MongoDB Atlas Documentation – NoSQL database structuring and indexing.
 - <https://www.mongodb.com/docs/>

Research Papers & Technical Articles

- Queue Management in Healthcare Systems – Study on OPD optimization.
 - <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4147743/>

UI/UX Design & Wireframing

- Material Design Guidelines – Modern UI trends and best practices.
 - <https://m3.material.io/>
- FontAwesome – Icons for better visual design.
 - <https://fontawesome.com/>

Learning Platforms

- Stack Overflow – Debugging and problem-solving community.
 - <https://stackoverflow.com/>
- YouTube Coding Tutorials – Learning advanced web and backend development.
 - <https://www.youtube.com/>
- W3Schools – Web development learning platform.
 - <https://www.w3schools.com/>