Design and Implementation of a Hospital Management System

by

Ruhul Amin Lalan
ID: CSE2102023015

Sumiya Akter
ID: CSE2102023031

Md. Antar
ID: CSE2102023041

Nazia Tabassum
ID: CSE2102023113

Supervised by
Arifur Rahaman

Submitted in partial fulfillment of the requirements for the degree of
Bachelor of Science in Computer Science and Engineering



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SONARGAON UNIVERSITY (SU)
January 2025

Design and Implementation of a Hospital Management System

by

Ruhul Amin Lalan
ID: CSE2102023015

Sumiya Akter
ID: CSE2102023031

Md. Antar
ID: CSE2102023041

Nazia Tabassum
ID: CSE2102023113

Supervised by
Arifur Rahaman

Submitted in partial fulfillment of the requirements for the degree of
Bachelor of Science in Computer Science and Engineering



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SONARGAON UNIVERSITY (SU)
January 2025

APPROVAL

The project titled "**Design and Implementation of a Hospital Management System**" submitted by Ruhul Amin Lalan (CSE2102023015), Sumiya Akter (CSE2102023031), Md Antar (CSE2102023041), Nazia Tabassum (CSE2102023113), to the Department of Computer Science and Engineering, Sonargaon University (SU), has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering and approved as to its style and contents.

Board of Examiners

Arifur Rahaman
Assistant Professor & Coordinator
Department of Computer Science and Engineering
Sonargaon University (SU)

Supervisor

(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

Examiner 1

(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

Examiner 2

(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

Examiner 3

DECLARATION

We, hereby, declare that the work presented in this report is the outcome of the investigation performed by us under the supervision of **Arifur Rahaman**, Assistant Professor and Coordinator, Department of Computer Science and Engineering, Sonargaon University, Dhaka, Bangladesh. We reaffirm that no part of this project has been or is being submitted elsewhere for the award of any degree or diploma.

Countersigned

(Arifur Rahaman)
Assistant Professor and Coordinator
Supervisor

Signature

Ruhul Amin Lalan
ID: CSE2102023015

Sumiya Akter
ID: CSE2102023031

Md. Antar
ID: CSE2102023041

Nazia Tabassum
ID: CSE2102023113

ABSTRACT

Management is recognized as a necessary step in the delivery of successful systems and software projects, discipline is also required by standards, regulations, and quality improvement initiatives. Creating and managing requirements is a challenge of IT, systems and product development projects or indeed for any activity where you have to manage a contractual relationship. Organization need to effectively define and manage requirements to ensure they are meeting needs of the customer, while proving compliance and staying on the schedule and within budge. The impact of a poorly expressed requirement can bring a business out of compliance or even cause injury or death. Requirements definition and management is an activity that can deliver a high, fast return on investment. The Design and Implementation of a Hospital Management System undertaken as a project is based on relevant technologies. The main aim of this project is to develop software for bank management system. This project is to develop software for bank management system. This project has been developed to carry out the processes easily and quickly, which is not possible with the manuals systems, which are overcome by this software. This project is developed using VB language and. Hence it provides the complete solution for the current management system.

ACKNOWLEDGMENT

At the very beginning, we would like to express my deepest gratitude to the Almighty Allah for giving us the ability and the strength to finish the task successfully within the schedule time.

We are auspicious that we had the kind association as well as supervision of **Arifur Rahaman**, Assistant Professor and Coordinator, Department of Computer Science and Engineering, Sonargaon University whose hearted and valuable support with best concern and direction acted as necessary recourse to carry out our project.

We are also thankful to all our teachers during our whole education, for exposing us to the beauty of learning.

Finally, our deepest gratitude and love to my parents for their support, encouragement, and endless love.

LIST OF ABBREVIATIONS

CSS	Cascading Style Sheet
DDL	Data Definition Language
DEY	Don’t Repeat Yourself
DFD	Data Flow Diagram
EHR	Electronic Health Records
ERD	Entity Relationship Diagram
HMS	Hospital Management System
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IOT	Internet of Things
JS	JavaScript
JSON	JavaScript Object Notation
MVT	Model View Template s
ORM	Object Relational Mapping
PC	Personal Computer
RAM	Random Access Memory
RDBMS	Relational Database Management System
ROM	Read Only Memory
SDLC	Software Development Life Cycle
SQL	Structured Query Language
UAT	User Acceptance Testing
UI	User Interface
UML	Unified Modeling Language
URL	Uniform Resource Locator
USDP	Unified Software Development Process
UX	User Experience
VS	Visual Basic

TABLE OF CONTENTS

Title			Page No.
DECLARATION …….……..			i
ABSTRACT ……...…….			ii
ACKNOWLEDGEMENT ……..……			iii
LIST OF ABBREVIATION …….……			iv
CHAPTER 1			1 – 3
INTRODUCTION TO HOSPITAL MANAGEMENT SYSTEM			
1.1	Introduction ……..…….....		1
1.2	Project Objective……...…….....		1
1.3	Project Planning…….....		1-2
1.4	Problem Statement ……...…….....		2
1.5	Feasibility Study …….....		2
1.6	Requirement Specification …….....		3
	1.6.1	Hardware Requirements……...	3
	1.6.2	Software Requirements…….	3
CHAPTER 2			4-7
BACKGROUND OF HOSPITAL MANAGEMENT SYSTEM			
2.1	Background…….…….....		4
2.2	Introduction to Web Development.…….....		4
2.3	Technologies Used……		4-5
	2.3.1	Hypertext Markup Language…….	4
	2.3.2	Cascading Style Sheets……..	4
	2.3.3	Python……	4
	2.3.4	Django……...	5
	2.3.5	MySQL Database……..	5
	2.3.6	JavaScript …….	5
	2.3.7	JavaScript Object Notation……	5
2.4	Role of Each Technology in the Project……...		5
2.5	Advantages of Using This Stack……...		6
2.6	Why These Technologies Were Chosen……...		6
2.7	Related Work……		6-7

CHAPTER 3 8-11
IMPORTANCE OF HOSPITAL MANAGEMENT SYSTEM
3.1 Importance of Hospital Management System…….. 8
3.2 Hospital Management System Website…… 8
3.2.1 Key Features of the Home Page…… 8
3.2.2 Billing Department…… 8-9
3.2.3 Find a Doctor……..... 9
3.2.4 Overview of the "Find a Doctor" Feature…….. 9
3.3 Find a Service ……..... 9-11
3.3.1 Overview of the "Find a Service" Feature.…… 9-10
3.3.2 Find a Visited and Patient…….. 10
3.3.3 Overview of the "Find a Visited and Patient" Feature …….. 10
3.3.4 Overview of the "Find a Media" Feature……... 10-11

CHAPTER 4 12-20
SYSTEM ANALYSIS OF THE PROJECT
4.1 Software Development Life Cycle…… 12
4.2 Implementing the SDLC ……..... 12-14
4.2.1 Planning and Requirement Gathering……... 12-13
4.2.2 System Design…… 13
4.2.3 Implementation/Development…… 13-14
4.2.4 Testing…… 14
4.2.5 Deployment…… 14
4.2.6 Maintenance and Updates…….. 14
4.3 Need for Visual Programming……... 15
4.4 Advantages of Visual Programming…….. 15
4.5 Introduction to Back End Tool…….. 15-17
4.6 Research Methodology…….. 17
4.7 Data Collection Methods……... 17
4.8 Questionnaire…… 17
4.9 Data Analysis Method……... 17
4.10 Software/Product Development Methods…….. 17-20

CHAPTER 5 21-24
GRAPHICAL REPRESENT OF PROJECT
5.1 Data Flow Diagram…….…….... 21
5.2 Waterfall Model……..... 21
5.3 Data flow of Hospital Management …….. 22
5.4 Use case Diagram…….. 23

5.5 E-R Diagram…….. 24

CHAPTER 6 25-30
TESTING AND DEBUGGING
6.1 Introduction……..…..... 25
6.2 Testing and Debugging…….……..... 25-28
6.3 Performance Testing Report…….. 28-30

CHAPTER 7 31-45
PROJECT OVERVIEW
7.1 Introduction…… 31
7.2 System Design and Architecture…… 31
7.3 Technology Stack…….. 31-32
7.4 Implementation Strategy…… 32
7.5 Overview…… 33-45

CHAPTER 8 46-48
CONCLUSION
8.1 Conclusion …..... 46
8.2 Future Scope of the Project…… 46-47
8.3 Limitations of Hospital System……. 47

REFERENCES ……..……...…... 48

LIST OF TABLES

Figure No.	Title	Page No.
Table 1.1	Project Plan	1
Table 1.2	Hardware requirements	3
Table 1.3	Software Requirements	3
Table 4.1	Functional requirements for patient	19
Table 4.2	Functional requirements for admin	19
Table 4.3	Non-functional requirements	20

LIST OF FIGURES

Figure No.	Title	Page No.
Fig 1.1	Project Planning	2
Fig 4.1	Software Development Life cycle	12
Fig 4.2	Unified software development process	18
Fig 5.1	Waterfall Model	21
Fig 5.2	Data flow diagram	22
Fig 5.3	Use case diagram	23
Fig 5.4	ER- Diagram	24
Fig 6.1	Testing and Report Information	28
Fig 6.2	List of API	29
Fig 6.3	Error	29
Fig 6.4	Response Time Percentiles Over Time	30
Fig 6.5	Active Threads Over Time	30
Fig 7.1	Home Page	33
Fig 7.2	Department	34
Fig 7.3	Find Doctor	35
Fig 7.4	Search Doctor	35
Fig 7.5	Doctor Profile	36
Fig 7.6	Doctor Appointment	36
Fig 7.7	Services	37
Fig 7.8	Patient Feedback	38
Fig 7.9	Equipment’s	39
Fig 7.10	Room Rate	39
Fig 7.11	Room Booking	40
Fig 7.12	Contacts	40
Fig 7.13	Media Gallery	41
Fig 7.14	Information	41
Fig 7.15	About Us	42
Fig 7.16	Login Page	43
Fig 7.17	Admin Panel	43
Fig 7.18	User of Admin	44
Fig 7.19	Doctor Add and Changes	44
Fig 7.20	Appointment List of Patient	45
Fig 7.21	Room Booking List of Patient	45

CHAPTER 1

INTRODUCTION TO HOSPITAL MANAGEMENT SYSTEM

1.1 Introduction

Introduction to the Lans Unity Exchange Hospital Management System Website Project: This project focuses on the design and development of a Hospital Management System (HMS) website aimed at improving hospital operations, patient experience, and administrative efficiency. The web-based platform provides a comprehensive solution for managing various aspects of a hospital, including patient registration, appointment scheduling, medical records, billing, and doctor-patient communication.

1.2 Project Objective

Healthcare systems worldwide face increasing pressure to improve efficiency and provide better services to patients. Hospitals and healthcare facilities must ensure that resources are utilized effectively, while also ensuring that patients receive timely care and attention. A comprehensive, easy-to-navigate HMS can drastically reduce administrative burdens, streamline patient care processes, and help healthcare providers deliver a higher standard of service.

The objectives of Hospital Management System (HMS) are:

a) To provide an online appointment.
b) To provide systematic way of storing data.
c) To retrieve patient information.

1.3 Project planning

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

No	Task Name	Duration	Start	End
1	Planning and Analysis	3 Month	06-01-2024	06-03-2024
2	Design & Implementation	3 Month	07-03-2024	16-06-2024
3	Development	3 Month	17-06-2024	09-09-2024
4	Testing and Maintenance	3 Month	10-09-2024	30-12-2024

Table 1.1 Project Plan

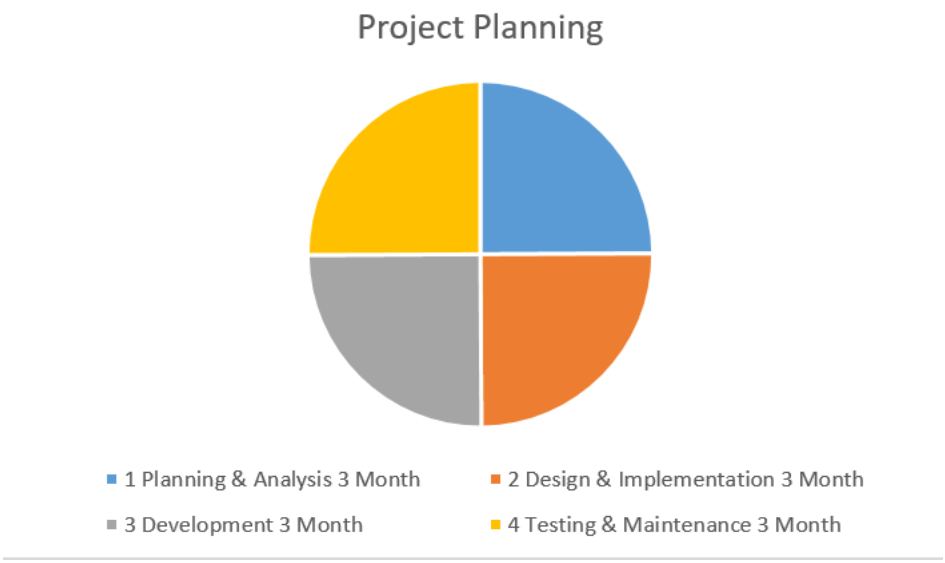


Fig 1.1 Project Planning

1.4 Problem Statement

The current manual system need to call or fill the form to do appointment with doctor. The current manual system takes more time to meet with doctor because there is no schedule. If we do it by online, we need less time to manual system. Online appointment scheduling systems are not only time-saving agents but income-generating assistants too, that provide operational convenience and promote relations built with patients.

1.5 Feasibility Study for a Hospital Management System (HMS)

A Feasibility Study is a critical part of any project to evaluate its practicality, potential costs, and the likelihood of success. When developing a Hospital Management System (HMS), especially for Defense-related use, it's important to assess whether the proposed solution can meet both the technical and business requirements. Here's a comprehensive feasibility study, including technical, operational, legal, and financial aspects.

- **Project Overview**

A Hospital Management System (HMS) is a software application designed to manage hospital operations like patient records, appointment scheduling, billing, inventory, and management. In the case of defense applications additional functionality for, privacy, and specific health needs may be required.

- **Technical Feasibility**

Technical feasibility involves evaluating if the proposed system can be built using and relevant technologies. The technical components of the system must meet the desired performance, security, and functionality requirements.

- **Operational Feasibility**

Operational feasibility assesses if the proposed system aligns with the hospital's operational requirements and can be integrated into the daily workflows of healthcare providers, especially in a defense context.

1.6 Requirement Specification

To successfully implement a Hospital Management System (HMS), it's essential to define both the hardware and software requirements. These requirements ensure the system is scalable, reliable, secure, and can handle the operational needs of a hospital, from managing patient records to inventory control and billing. Below are the detailed hardware and software requirements for a Hospital Management System:

1.6.1 Hardware requirements

To develop this system, hardware requirement is important things to be required. Table 1.2 shows the hardware requirements to develop Hospital Management System (HMS).

Hardware	Description
Laptop with 8GB of RAM	Workstation for the project
Hard disk space (227GB)	To save data
Others	Keyboard and Microsoft Mouse or compatible pointing device and Android mobile etc.
Printer	A printer to test the report generating function.

Table 1.2 Hardware requirements

1.6.2 Software requirements

After hardware requirement, software requirement is the important things to be install.
Table 1.3 shows the software requirement to develop Hospital Management System (HMS) and Android studio.

Software	Description	Cost (RM)
Python	Web framework (open-source)	Free
Django	Web framework(Admin Interface)	Free
Java	Android version	Free
Microsoft Office 2007	Documentation Tools.	Free
Notepad ++	Website coding editing tool	Free
SQLite3	Database used for this system.	Free
Microsoft Visio	To draw Gantt chart	Free
Web hosting	Platform to get hosting	Free
Google Chrome, Internet Explorer	Web Browser.	Free

Table 1.3 Software Requirements

CHAPTER 2

BACKGROUND OF HOSPITAL MANAGEMENT SYSTEM

2.1 Background

When writing the background section for a project involving CSS, Python, HTML, and Django, it's important to explain the significance of each technology, how they work together, and why you chose them for the project. The background should establish the context for the project, introducing the web development technologies you're using and how they contribute to the overall system.

Here's a structured format for the background section of our project.

2.2 Introduction to Web Development

- In this section, briefly explain what web development is and the need for creating dynamic, interactive web applications in modern technology. Highlight how web applications have become essential for businesses, education, healthcare, etc.
- Mention that the project utilizes a combination of front-end (CSS, HTML) and back-end (Python, Django) technologies to build a full-stack web application.

2.3 Technologies Used

In our hospital Management system project, we used HTML, CSS, python and also Django. In our project we used HTML to create the main structure of hospital management system and then style by the CSS for making the hospital management system visually and user friendly we use python for back end logic of hospital management system and at last we used the Django for framework for building secure and scalable web application and also used MySQL database in our hospital management system.

2.3.1 Hypertext Markup Language (HTML)

HTML is the standard language used for structuring web pages. It defines the elements and structure of a webpage, such as headings, paragraphs, forms, links, and images. In this project, HTML serves as the backbone for creating the web pages that display hospital management features like patient records, appointment scheduling, etc. [1].

2.3.2 Cascading Style Sheets (CSS)

CSS is used to style and layout web pages, such as colors, fonts, spacing, and responsiveness. It helps in creating visually appealing and user-friendly interfaces. In this project, CSS is used to design the front-end of the application, making it more intuitive and easier to navigate, enhancing the overall user experience [2].

2.3.3 Python

Python is a high-level programming language known for its simplicity and versatility. It is commonly used for server-side development, data processing, and backend logic. In this project, Python is used as the primary backend language to write the logic of the hospital management system, such as handling user authentication, managing patient data, and processing forms [3].

2.3.4 Django

Django is a powerful web framework written in Python that follows the Model-View-Controller (MVC) architecture. It provides tools for building web applications quickly and securely. In this project, Django is used as the framework to build the web application, handling the routing, database interactions, and business logic for the hospital management system. Django's built-in features like user authentication, form handling, and admin interface simplify the development process [4].

2.3.5 MySQL Database

MySQL is an open source relational database management system based on SQL. It was designed and optimized for web applications and can run on any platform. As new and different requirements emerged with the internet, MySQL became the platform of choice for web developers and web-based applications. Because it's designed to process millions of queries and thousands of transactions, MySQL is a popular choice for ecommerce businesses that need to manage multiple money transfers. On-demand flexibility is the primary feature of MySQL [5].

2.3.6 JavaScript (JS)

JavaScript is a high-level, interpreted programming language that is primarily used to create interactive and dynamic web content. It is one of the core technologies of the World Wide Web, alongside HTML and CSS. JavaScript allows developers to add interactivity to websites, such as responding to user actions, manipulating the Document Object Model, and making asynchronous requests to servers [6].

2.3.7 JavaScript Object Notation (JSON)

JavaScript Object Notation (JSON) is a lightweight data-interchange format that is easy for humans to read and write and easy for machines to parse and generate. JSON is widely used to transmit data between a server and a web application as text [7].

2.4 Role of Each Technology in the Project

- **HTML**
  Defines the structure of the hospital management web pages, including forms for adding patients, scheduling appointments, viewing records, etc.
- **CSS**
  Provides the design for the hospital management interface, ensuring the system is not only functional but also visually appealing and accessible across different devices (desktop, mobile).
- **Python**
  Implements the backend logic, handling tasks like retrieving patient data, updating records, processing forms, and integrating with the database to store and retrieve information.
- **Django**
  Acts as the framework that ties everything together. Django takes care of the routing, views, and templates, allowing Python code to interact seamlessly with HTML pages. It also helps manage the database and ensures security (e.g., through authentication mechanisms)

2.5 Advantages of Using This Stack

- **Django and Python**
  Django allows for rapid development and follows best practices, making it easy to build secure, scalable web applications. Python's clean syntax and large library support make it ideal for handling complex backend operations with minimal code.
- **HTML and CSS**
  These technologies are the foundation of web development. HTML is universally supported, and CSS offers extensive customization for responsive design. Together, they ensure that the hospital management system is accessible and looks good on all devices.
- **Integration**
  Django provides built-in features that integrate easily with Python to handle backend logic, and it allows for easy rendering of dynamic HTML templates. This integration allows the application to seamlessly manage data between the database and the frontend.
- **Security**
  Django comes with many built-in security features like SQL injection protection, cross-site scripting (XSS) prevention, and secure user authentication, which are critical for maintaining sensitive patient data in a hospital management system.

2.6 Why These Technologies Were Chosen

- **Simplicity and Efficiency**
  Python is chosen for its simplicity and readability, which speeds up the development process. Django, built on Python, follows the DRY (Don't Repeat Yourself) principle, making the code base clean and maintainable.
- **Scalability**
  Django is known for its ability to scale applications easily. As the hospital system grows, Django's modularity allows new features to be added without affecting the existing codebase.
- **Web Compatibility**
  HTML and CSS are essential for ensuring that the web application is cross-platform compatible, working well on desktop browsers and mobile devices, which is crucial for real-world usage.

2.7 Related Work

- In this section, mention any existing hospital management systems, frameworks, or similar projects that helped you decide to use Django and Python. Discuss how these existing systems influenced your approach and why you chose to improve upon or replicate certain aspects.

Example Background Section for a Hospital Management System Project

With the increasing demand for efficient management systems in the healthcare industry, a Hospital Management System (HMS) has become a necessity. The primary goal of an HMS is to streamline hospital operations, improve patient care, and reduce administrative workload. This project aims to develop a web-based hospital management system that integrates multiple functions such as patient management, appointment scheduling, doctor-patient

communication, and billing using modern web technologies like HTML**,** CSS**,** Python**,** and Django**.**

- **HTML** is used for structuring the web pages. It defines the layout and content, allowing easy interaction with users through forms, tables, and buttons.
- **CSS** is used to style the web pages, providing a clean, professional look and making the system responsive to different screen sizes.
- **Python**, a versatile and easy-to-learn programming language, is employed for writing the back-end logic of the system. It processes user requests, communicates with the database, and handles tasks like adding or updating patient information, scheduling appointments, and processing bills.
- **Django**, a powerful Python web framework, enables rapid development by providing a robust structure for handling web requests, managing databases, and ensuring security. It also includes built-in features like user authentication and an admin interface to help manage the hospital system efficiently.

The combination of these technologies ensures that the system is easy to develop, scalable, secure, and user-friendly. By utilizing the Django framework, the project benefits from rapid development tools and pre-built solutions for common tasks like form validation and database migrations.

CHAPTER 3

IMPORTANCE OF HOSPITAL MANAGEMENT SYSTEM

3.1 Importance of Hospital Management System

In the modern healthcare environment, efficiency, accuracy, and accessibility are paramount. The rise in patient numbers, the complexity of healthcare services, and the necessity for real-time data access have made traditional, paper-based systems increasingly inadequate. A Hospital Management System (HMS) addresses these challenges by integrating various hospital functions into a cohesive, digital platform that enhances operational efficiency and patient care.

3.2 Hospital Management System Website

The Hospital Management System (HMS) website is designed to streamline the management of hospital operations through an online interface. The website aims to provide users with a simple and intuitive platform to access various hospital services, such as patient management, appointment scheduling, medical record tracking, and billing. The home page serves as the first point of contact for both hospital staff and patients, offering easy navigation to key features.

3.2.1 Key Features of the Home Page

1. **Header Section:**
    - **Logo and Hospital Name:** The hospital logo and name are prominently displayed at the top of the page for easy brand recognition.
    - **Navigation Menu:** Clear and concise navigation links (e.g., Home, Services, Appointments, About Us, Contact) are placed at the top for easy access to the various sections of the website.
    - **Search Bar:** A search function is integrated into the header to allow users to quickly find specific information, such as doctors, treatments, or patient records.
2. **Main Banner:**
    - The home page features a large, visually appealing banner with a welcoming message such as "Welcome to [Hospital Name], Your Health, Our Priority."
3. **Introduction Section:**
    - A brief description of the hospital and its services. This section provides users with an overview of the hospital's mission, values, and the types of services offered (e.g., emergency care, surgeries, outpatient services).

3.2.2 Billing Department

The Billing Department is responsible for generating invoices, processing payments, and handling insurance claims. The HMS simplifies billing by automating invoice creation, integrating insurance data, and providing patients with transparent billing details.

Key Features:

- **Automated Billing**: The system automatically generates patient bills based on the services provided, reducing manual errors.
- **Insurance Integration**: The system facilitates direct communication with insurance providers, making the claims process faster and more efficient.
- **Online Payment**: Patients can pay their bills through the website, reducing the need for in-person transactions.

On the Home Page, patients can access their billing information, view a breakdown of services, and pay bills online.

3.2.3 Find a Doctor – Hospital Management System (HMS)

The Find a Doctor feature is a fundamental aspect of the Hospital Management System (HMS), designed to improve the patient experience by enabling them to quickly search for doctors based on specific criteria, such as specialty, availability, and location. This feature simplifies the process of finding the right healthcare provider and scheduling an appointment, ultimately leading to better access to care and improved patient satisfaction. This section discusses how the Find a Doctor feature works within the context of the HMS, its significance, and the technology that powers it.

3.2.4 Overview of the "Find a Doctor" Feature

The Find a Doctor feature allows patients to search for healthcare providers within the hospital based on various criteria, including:

- **Specialization**: Patients can search for doctors by their medical specialties, such as cardiology, pediatrics, dermatology, etc.
- **Doctor's Name**: Patients may choose to search for a specific doctor by name.
- **Location**: The system allows patients to search for doctors based on their location within the hospital or branch.
- **Availability**: The system provides real-time availability of doctors, allowing patients to book appointments directly.
- **Consultation Type**: Patients can choose to find doctors for either in-person visits or virtual consultations (telemedicine).

3.3 Find a Service – Hospital Management System (HMS)

The Find a Service feature is a critical component of the Hospital Management System (HMS) that enhances the hospital's ability to connect patients with the wide range of healthcare services it provides. With this feature, patients can easily search for and book services such as consultations, diagnostic tests, surgeries, and wellness programs. By making it easier to locate and schedule necessary services, this functionality plays a key role in improving both patient satisfaction and operational efficiency within the hospital.

3.3.1 Overview of the "Find a Service" Feature

The Find a Service feature allows patients and hospital users to search for a variety of services offered by the hospital. These services include medical consultations, diagnostic tests, emergency services, surgical operations, and preventive care programs. The system enables patients to easily locate specific services based on:

- **Service Type**: Find services such as consultations, surgeries, diagnostics, or therapies.
- **Department**: Locate services provided by different departments (e.g., cardiology, radiology, orthopedics, etc.).
- **Specialization**: Services related to specific medical fields (e.g., maternity services, cancer care, pediatric care).
- **Availability**: Patients can search for services available at specific times or days, allowing for real-time scheduling.

3.3.2 Find a Visited and Patient

The Find a Visited & Patient feature is an essential tool in the Hospital Management System (HMS) that helps hospital staff efficiently track and manage the details of patients who have visited or are currently receiving care at the hospital. This feature allows authorized personnel to search for patient records based on various criteria and provides quick access to critical patient information, such as past visits, medical history, ongoing treatments, and discharge status. It is a vital component of hospital operations, ensuring streamlined patient care and data management.

3.3.3 Overview of the "Find a Visited and Patient" Feature

The Find a Visited & Patient feature allows hospital staff (such as doctors, nurses, and administrative personnel) to access detailed records of patients who have visited or are currently visiting the hospital. Key functionalities include:

- **Search Patient by Visit History**: The system enables staff to search for patients based on the history of their visits, including consultations, treatments, and hospitalizations.
- **Track Current Patients**: Hospital staff can check the status of patients currently admitted to the hospital, their assigned doctors, ongoing treatments, and more.
- **Filter by Patient Demographics**: The system allows filtering by patient information such as age, gender, department, or specific diagnosis.
- **Medical Records and History**: Staff can access the full medical records of a patient, including previous diagnoses, treatments, medications, and laboratory results.
- **Visit Details**: View information on each visit, such as the date of visit, reasons for the visit, doctors seen, procedures or tests performed, and outcomes.

This feature improves the hospital's ability to coordinate care, track treatment progress, and provide personalized services to patients based on their past and present medical needs.

3.3.4 Overview of the "Find a Media" Feature

The Find a Media feature serves as a digital library or repository within the Hospital Management System (HMS). It allows authorized users to search, view, and manage media content. These media resources can include:

- **Hospital Promotional Videos and Images**: For marketing, branding, and outreach purposes.
- **Educational Videos and Images**: Related to health awareness campaigns, medical procedures, preventive care, and wellness programs.

- **Event Media**: Content related to hospital events such as health camps, conferences, charity drives, and more.
- **Patient Education**: Informational videos and images related to patient care, post-surgery instructions, and wellness programs.

This feature supports both the hospital's internal needs (for staff training, events, etc.) and external needs (for educating patients, attracting new visitors, and promoting hospital services).

CHAPTER 4

SYSTEM ANALYSIS OF THE PROJECT

4.1 Software Development Life Cycle

A system development life cycle is a logical process by which system analysts, software engineers, programmers, and end users build information systems and computer applications to solve business problems and needs.

The system development life cycle can be divided into seven phases as shown in figure

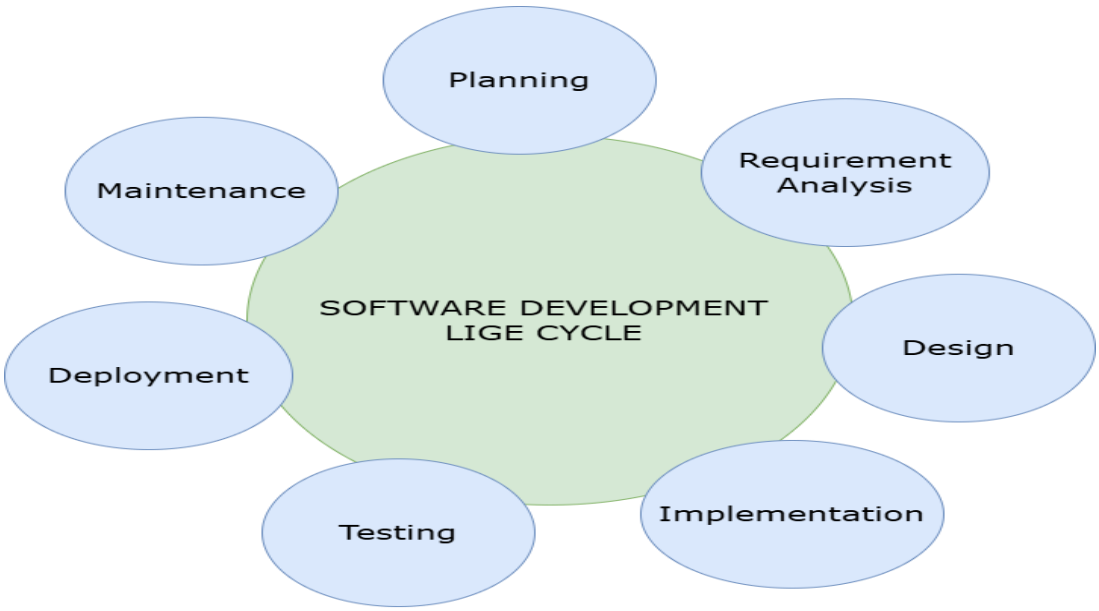


Figure 4.1: Software Development Life Cycle.

4.2 Implementing the SDLC for a Hospital Management System (HMS)

Involves following the seven stages of the software development life cycle to ensure that the software meets the needs of the hospital's administration, staff, and patients while maintaining efficiency, security, and compliance with regulations. Below is a detailed breakdown of how the SDLC can be applied to an HMS project:

4.2.1 Planning and Requirement Gathering

- **Goal:** Understand and document the hospital's needs and define the system's scope.
- **Implementation Steps:**
    1. **Stakeholder Meetings:** Conduct meetings with hospital administrators, doctors, nurses, IT staff, and other stakeholders to gather requirements.
    2. **Identify Functional Requirements:**

- Patient Registration and Admission
- Doctor Scheduling and Appointment Management
- Billing and Payments
- Prescription and Medical Records Management
- Inventory and Pharmacy Management
- Patient Discharge and Reports Generation

3. **Non-Functional Requirements:**
   - Security: Compliance with healthcare regulations (e.g., HIPAA, GDPR)
   - Scalability to handle a large number of patient records
   - Performance to ensure real-time processing and data retrieval
4. **Document Requirements:** Create a Software Requirements Specification (SRS) document outlining both functional and non-functional requirements.
5. **Feasibility Study:** Assess the project's feasibility from a technical, financial, and operational standpoint.
6. **Risk Management:** Identify potential risks such as privacy concerns, data security, and integration with existing hospital systems (e.g., Electronic Health Records - EHR).

4.2.2 System Design

- **Goal:** Create a blueprint of the system, outlining its architecture, database design, and user interfaces.
- **Implementation Steps:**
  1. **High-Level System Architecture Design:**
     - Design the architecture to be modular and scalable.
     - Decide whether the system will be a cloud-based solution, on-premises, or a hybrid.
  2. **Database Design:**
     - Design the database schema to store patient data, medical records, appointments, and billing information.
     - Use Entity Relationship Diagrams **(**ERD**)** to map out data relationships (e.g., patient, doctor, appointments, prescriptions).
  3. **User Interface Design:**
     - Create wireframes or mockups for the system's user interfaces, such as patient registration forms, doctor dashboards, and admin panels.
     - Focus on usability and accessibility for hospital staff and patients.
  4. **Technology Stack Selection:**
     - Frontend: React, Angular, or Vue.js for web interfaces.
     - Backend: Node.js, Python (Django/Flask), or Java (Spring Boot).
     - Database: MySQL, PostgreSQL, or MongoDB.
     - Consider integration with existing Electronic Health Record (EHR) systems and other software in the hospital.

4.2.3 Implementation/Development

- **Goal:** Write the code based on the design specifications.
- **Implementation Steps:**

1. **Frontend Development:** Build user interfaces for various roles (e.g., patient, doctor, nurse, admin) ensuring intuitive and responsive design.
2. **Backend Development:** Implement business logic for managing patient records, appointments, billing, and prescription generation.
3. **API Development:** Build RESTful APIs for inter-system communication, such as integration with hospital billing systems or external pharmacy services.
4. **Database Integration:** Implement the database schema for patient records, doctor schedules, inventory management, etc.
5. **Version Control:** Use tools like Git to manage source code and GitHub/GitLab/Bitbucket for collaboration.

4.2.4 Testing

- **Goal:** Ensure the system works as expected, is secure, and meets quality standards.
- **Implementation Steps:**
  1. **Unit Testing:** Write unit tests for individual functions and modules (e.g., for billing calculations, appointment scheduling).
  2. **Integration Testing:** Ensure that the different modules of the system (patient management, appointment system, pharmacy) work together smoothly.
  3. **System Testing:** Test the system as a whole to ensure all requirements are met, including performance and security.
  4. **User Acceptance Testing (UAT):** Have hospital staff test the system to ensure it meets their expectations. This includes validating workflows such as patient registration and billing.
  5. **Security Testing:** Conduct penetration testing to identify vulnerabilities (e.g., unauthorized access to patient data).
  6. **Compliance Testing:** Ensure the system complies with legal and regulatory requirements (HIPAA, GDPR, etc.).

4.2.5 Deployment

- **Goal:** Deploy the system in a live hospital environment.
- **Implementation Steps:**
  1. **Staging Environment:** Deploy the system to a staging environment that mirrors the production setup to test the system under realistic conditions.
  2. **Deployment Strategy:** Deploy the system incrementally or all at once, depending on hospital readiness. This can include: Data Migration**,** Training.
  3. **Production Deployment:** Deploy the system to the hospital's live environment with monitoring to ensure everything runs smoothly.
  4. **Backup Plans:** Implement rollback strategies in case of issues, such as restoring from backups if the deployment fails.

4.2.6 Maintenance and Updates

- **Goal:** Maintain the system by fixing bugs, adding new features, and ensuring its continued efficiency.
- **Implementation Steps:**
  1. **Software Updates:** Regularly update the system with new features (e.g., telemedicine integration, advanced reporting features).

2. **Security Patches:** Apply security patches and updates to protect patient data from evolving threats.
3. **Monitor System Health:** Monitor the system for issues such as slow load times or database errors and fix them promptly.

4.3 Needs for Visual Programming

- There are several programming tools that allow us to build such visually appealing and intuitive interface. These tools allow us to design interface that employ user friendly features such as menus, buttons, windows etc[8].
- However, the disadvantage of such tools is that the interface is designed using code. The programmer has to code the user interface features specifying the size, position etc. this makes designing the user interface a major task in itself.

4.4 Advantage of Visual Programming

- Visual development of graphical user interface which are easy to use and easy to learn.
- A programmer need not write code to display the required component.

- **For Example: -**

The visual programming environment displays a list of available components. The programmer picks up the required component from this list to display it.

- The component can be moved, resized and even deleted, if so required.
- There is no restriction on the number of controls that can be placed on a form.
- The interface components provided by the visual programming environment have some code built into them.

A button' knows' when it has been clicked upon. In the case of conventional programming tools, the programmer has to write code to determine the component that has been clicked and then execute the appropriate code.

4.5 Introduction to Back End Tool

Introduction to SQL: **-**

SQL is a standard computer language for accessing and manipulating databases.

- SQL stands for **S**tructured **Q**uery **L**anguage.
- SQL allows you to access a database.
- SQL is an ANSI standard computer language.
- SQL can execute queries against a database.
- SQL can retrieve data from a database.
- SQL can insert new records in a database.
- SQL can delete records from a database.
- SQL can update records in a database.
- SQL is easy to learn.

- SQL is an ANSI (American National Standards Institute) standard computer language for accessing and manipulating database systems. SQL statements are used to retrieve and

update data in a database. SQL works with database programs like MS Access, DB2, Informix, MS SQL Server, Oracle, Sybase, etc.

- Unfortunately, there are many different versions of the SQL language, but to be in compliance with the ANSI standard; they must support the same major keywords in a similar manner (such as SELECT, UPDATE, DELETE, INSERT, WHERE, and others).
- SQL Database Tables: - A database most often contains one or more tables. Each table is identified by a name (e.g. "Customers" or "Orders"). Tables contain records (rows) with data.

Below is an example of a table called "Persons": -

Last Name	First Name	Address	City
Akter	Sumiya	Demra	Dhaka
Lalan	Ruhul Amin	Brahmmon Baria	Chattogram
Tabassum	Nazia	Rampura	Dhaka

- The table above contains three records (one for each person) and four columns (Last Name, First Name, Address, and City).
- SQL Queries: -

With SQL, we can query a database and have a result set returned.

- ✓ **A query like this: -**

```
SELECT Last Name FROM Persons
```

- ✓ **Gives a result set like this: -**

3. Last Name
4. Hansen
5. Habib

SQL Data Manipulation Language (DML)

- SQL (Structured Query Language) is syntax for executing queries. But the SQL language also includes syntax to update, insert, and delete records.
- These query and update commands together form the Data Manipulation Language (DML) **Part of SQL**: -

- **SELECT** - extracts data from a database table
- **UPDATE** - updates data in a database table
- **DELETE** - deletes data from a database table
- **INSERT INTO** - inserts new data into a database table

SQL Data Definition Language (DDL)

- The Data Definition Language (DDL) part of SQL permits database tables to be created or deleted. We can also define indexes (keys), specify links between tables, and impose constraints between database tables.

4.6 Research Methodology

The chapter firstly describe about the questionnaire as data collection method which include the sample for mix methods that used during the data collection methods. After successful data collection, collected data will be analyzed and interpreted.

Then, the chapter will continue on the software/product development methods to describe the selected approach used in the project. The detail activities that may have to be done in every stage shall be describes [9].

4.7 Data collection methods

Data collection method is important in developing any project. The data collection method that are used in gathering info to develop the system is questionnaire.

4.8 Questionnaire

Questionnaire has been is made to gather information of this system. The questionnaire is made by my own design that related to the project. (ok state) This questionnaire consists two structure of questions which is question A is for general personal data and section B is for current flow of business processes.

4.9 Data analysis method

All the data or requirement that has been collect will be analyze one by one to find the proper solution to the problem and design the solution they need. In this project the data analysis method that being used for questionnaire data is statistical analysis, the questionnaire are being collected and the statistical of answer are choose, that mean the developer will make a decision depend on the answer most selected for example most of respondent (18 over 25) strongly agree to use the online ordering. The developer will find solution to fulfill the requirement ask by customer.

4.10 Software/product development methods

To develop this system, I had identified a suitable methodology to follow during the implementation of Hospital Management System (HMS). The chosen methodology is Unified Software Development Process or known as the USDP. USDP is an architecture-centric, use-

case driven, iterative and incremental development process that leverages unified modelling language and is compliant with the system process engineering Meta model.

USDP is divided into four phases' inception, elaboration, construction and transition.

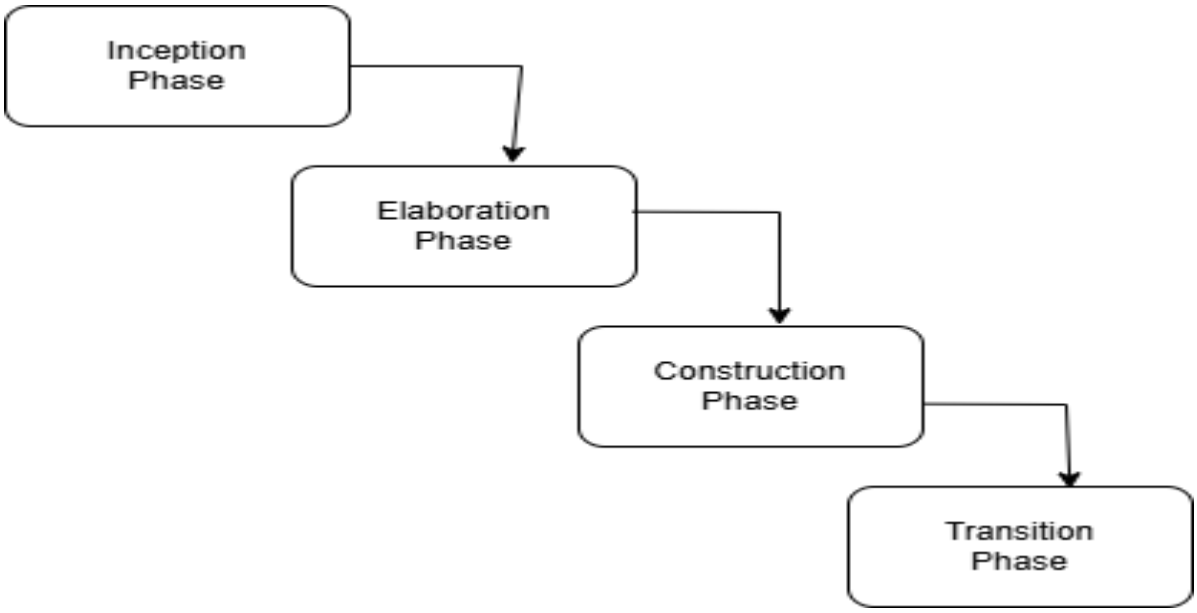


Fig 4.2: Unified Software Development Process

I. Inception Phase

The project planning starts in this phase. Inception is the smallest phase in the project. First, the information is gathered and the end users about the manual system. Before starting to build the system, problem must be analyses in current system then try to solve problem that have in current system. At the same time, element that is in the database should be known.

II. Elaboration Phase

During this phase, the manual system is investigated and improvement opportunities are identified based on the information gathered in the Initiation phase. The user requirements are gathered, analyzed and reviewed. The final deliverable from this phase is plan for the Construction phase.

III. Construction Phase

Construction is the largest phase in this project. The output produced in the Elaboration phase is built on the foundation laid in Elaboration. Common UML (Unified Modeling Language) diagrams used during this phase include Activity and Sequence Diagram.

IV. Transition Phase

The final project phase is Transition. In this phase the system is deployed to the target users. Feedback received from an initial release (or initial releases) may result in further refinements to be incorporated over the course of several Transition phase iterations.

The next chapter will discuss the analysis phase which will cover the user requirements and system requirements of the project. The system requirements will consist functional requirements and non-functional requirements in order to capture the system work's flow.

- **Functional requirements**

List of Requirement	PATIENT
Requirement 1: View patient info module	The system should allow patient to view details which consists of patient name, patient email, patient type, address, and contact number.
Requirement 2: View Appointment module	The system should allow patient to view appointment which consists of doctor name, time of appointment with doctor.
Requirement 5: Make payment module	The system should allow patient to make confirmation payment. Patient have to make payment either by online or cash. Next, the system should allow patient to click complete order and invoice will be produced.

Table 4.1 Functional requirements for patient

List of Requirement	ADMIN
Requirement1: Login	The system should allow admin to access their accounts through the insert correct admin ID and password.
Requirement2: Manage patient module	The system should allow admin to check list of patient that consists of patient name, patient type, address. The system should allow admin to search, update and delete patient information.
Requirement3: Register patient module	The system should allow admin to register new patient by inserting personal information of patient.
Requirement4: Generate bills module	The system should allow admin to view summary of bills. Admin can generate bills of patient.

Table 4.2 Functional requirements for admin

List of Requirement	Description
Requirement 1: Consistency	The Hospital Management System (HMS) provide consistency user interface design to the end-user. The designs of the screen are standardize and consistent that make the end-user feel comfortable to use it
Requirement 2: Availability	The system provides the end-user to login the system 24 hours a day.
Requirement 3: Security	The Hospital Management System (HMS) provide the password security access control to avoid unauthorized user to login to the system.
Requirement 4: Reliability	The HMS provide the effective method to maintain the back-end of the system. All of the subject list are manage by this system effectively.
Requirement 5: Usability	Learning time and experience needed such as hours or number of lessons to learn the system

Table 4.3 Non-functional requirements

CHAPTER 5

GRAPHICAL REPRESENT OF PROJECT

5.1 Data Flow Diagram

The data flow diagram is also known as “bubble chart” has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design so it is the starting point of specification down to the lowest level of detail. A DFDs consists of a series if bubbles joined by lines. The bubbles represent data transformation and the lines represent the data flow in the system.

- **DFD Symbols**
  - ✓ A system defined a source or destination of data.
  - ✓ An arrow identifies data flow, data in motion.
  - ✓ A circle represents the process that transforms incoming data flow to outgoing data flow.
  - ✓ An open rectangular is data store-data at rest or a temporary repository of data.

5.2 Waterfall Model

The Waterfall Model is a linear and sequential software development methodology. It is one of the earliest models in software engineering and is structured in distinct phases. Each phase depends on the deliverables of the previous one and must be completed before moving to the next. We are following the waterfall model in our hospital management system project for make real life project.

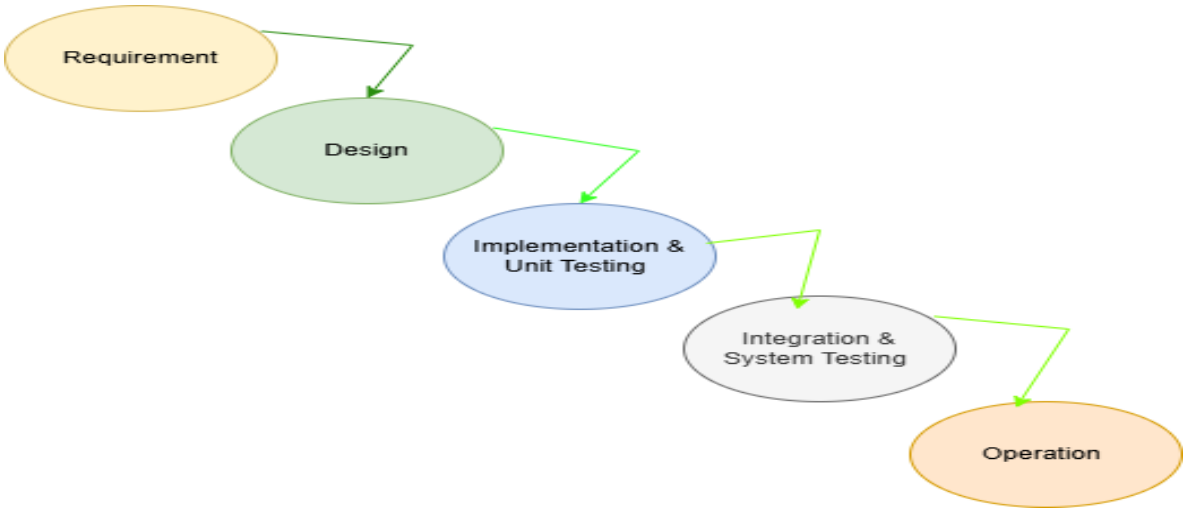


Fig-5.1: Waterfall model

5.3 Data flow of Hospital Management

Data flow refers to the movement and transfer of data between various components, processes, or systems. It describes how data is input, processed, stored, and output within a system. The concept is often visualized using diagrams or models to represent the flow of information between different entities in a system, such as users, databases, or software modules.

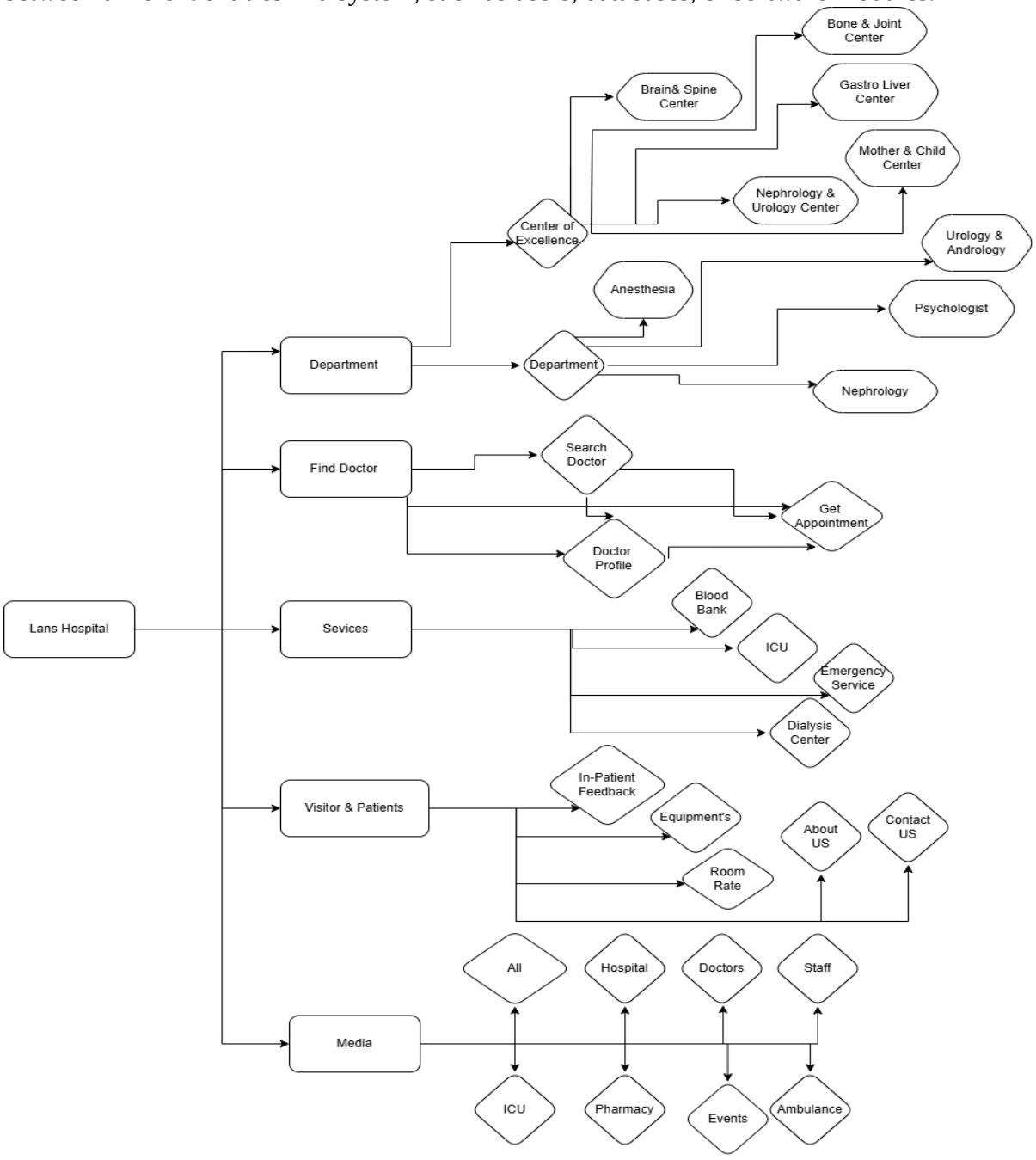


Fig 5.2: Data Flow Diagram

5.4 Use Case Diagram

A Use Case Diagram is a type of behavioral diagram in the Unified Modeling Language (UML) that visually represents the interactions between a system and its external entities, such as users (actors) or other systems. It is a high-level view of a system that shows the relationships between the actors and the use cases (functionalities or goals the system provides).

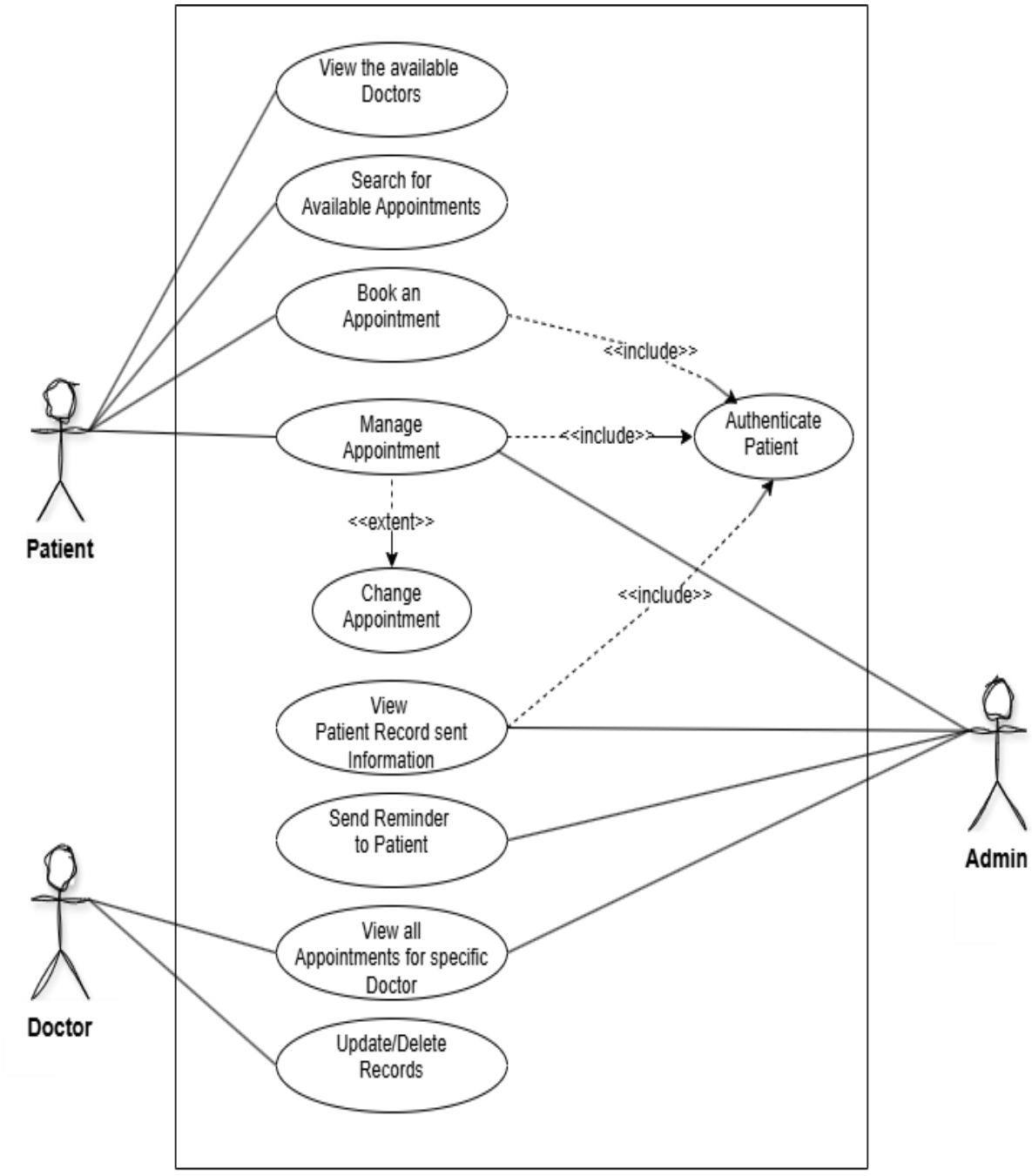


Fig 5.3: Use Case Diagram

5.5 Entity Relationship Diagram (ERD)

Entity relationship diagram is a diagram to show the different entities, its attributes and the relation between them. The entities are the tables in the database. Here we have 3 entities [10]:

1. Service,
2. Patient,
3. Doctor

In an Entity Relationship Diagram, we show entities in rectangular boxes, their attributes in circles and the connection between entity and attribute with a straight line between the two. The relationships between the two entities are shown in diamond boxes, and the connections between the entities and relationships are shown with the help of arrows. The above mentioned Entity Relationship Diagram (ERD) shows the relationship between different entities of Life Line Hospital Database. It also shows the different attributes of the entities of the Life Line Hospital Database.

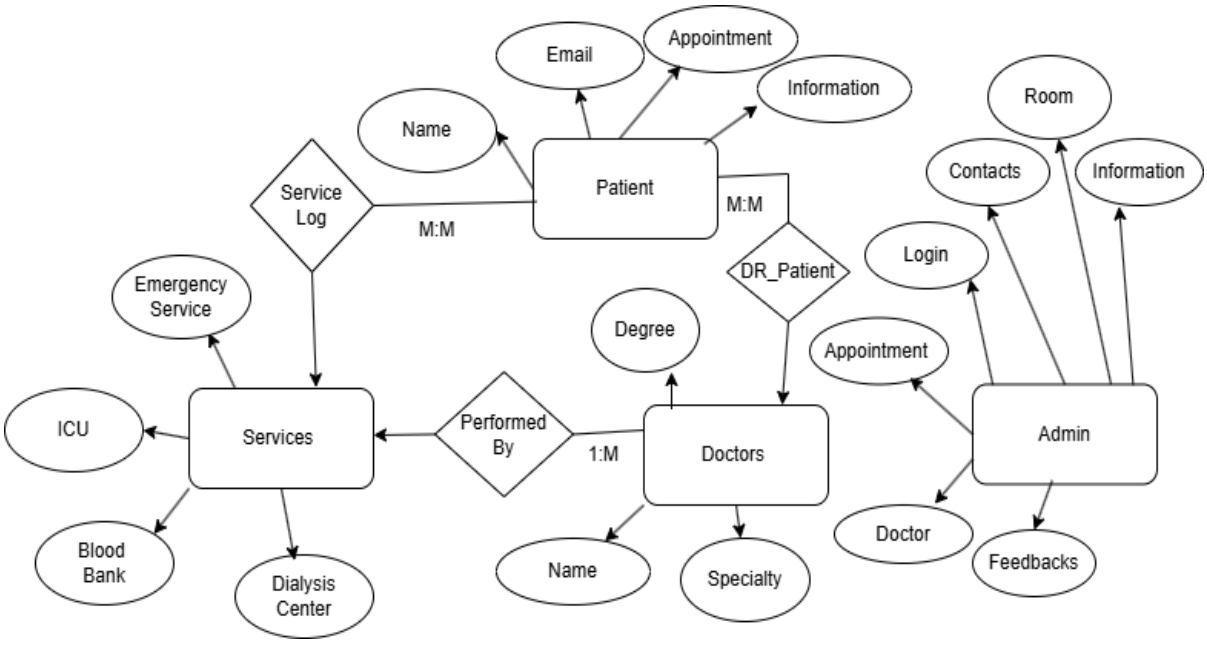


Fig 5.4: ER-Diagram

CHAPTER 6

TESTING AND DEBUGGING

6.1 Introduction

The implementation phase of software development is concerned with translating design specification into source code. The preliminary goal of implementation is to write source code and internal documentation so that conformance of the code to its specifications can be easily verified, and so that debugging, testing and modifications are eased.

The implementation team should be provided with a well-defined set of software requirement, an architectural design specification, and a detailed design description. Each team member must understand the objectives of implementation.

6.2 Testing and Debugging

When working on a Hospital Management System (HMS) with Django, Python, HTML, and CSS, testing is essential to ensure the functionality, security, and performance of the system. Below is a breakdown of key testing terms that are relevant to these technologies, along with how they apply specifically to the development of an HMS project.

1. Unit Testing

- **Definition**: Testing individual components (e.g., functions, methods, or classes) of your application in isolation to ensure they work as expected.
- **Django/Python Context**: In Django, unit testing can be done with the built-in unit test framework or third-party tools like pertest. You'll write tests for models, views, forms, and utility functions.

Example:

python
Copy code

```
from django.test import TestCase
from hospital.models import Patient

class PatientTestCase(TestCase):
   def test_full_name(self):
      patient = Patient (first_name="Ruhul Amin", last_name="Lalan")
      self.assertEqual(patient.get_full_name(), "Ruhul Amin Lalan")
```

2. Integration Testing

- **Definition**: Testing how multiple components work together. This is often done to ensure that the interactions between different modules or services are functioning properly.
- **Django/Python Context**: You might test how models, views, and templates interact with the database to ensure the application behaves as expected end-to-end.

Example:

python
Copy code
```
class AppointmentBookingTest(TestCase):
    def test_patient_appointment_creation(self):
        patient = Patient.objects.create(first_name="Ruhul Amin", last_name="Lalan")
        response = self.client.post('/appointments/', {'patient': patient.id, 'date': '2024-01-15'})
        self.assertEqual(response.status_code, 200)
        self.assertContains(response, "Appointment confirmed")
```

3. Functional Testing

- **Definition**: Testing to ensure that the software's features function according to the specified requirements, simulating end-user behavior.
- **Django/Python Context**: Functional tests will check whether the HMS's features, such as patient registration, appointment scheduling, and billing, perform as expected from the user perspective.

4. Regression Testing

- **Definition**: Ensuring that new code changes do not negatively affect existing functionality.
- **Django/Python Context**: Every time you add new features or fix bugs, you run regression tests to make sure previous features like patient registration or staff management still work correctly.

5. Security Testing Definition:

- Testing for vulnerabilities and ensuring that the system is secure from common threats like SQL injection, cross-site scripting (XSS), and unauthorized access.
- **Django/Python Context**: Django provides built-in protections, but you still need to ensure your authentication system, access control, and data handling are secure.

Example:

python
Copy code
```
def test_user_authentication(self):
    response = self.client.post('/login/', {'username': 'lanshospital', 'password':
'incorrectpassword'})
    self.assertEqual(response.status_code, 403)  # Unauthorized access
```

6. Performance Testing

- **Definition**: Ensuring that the system can handle expected load and scale appropriately.
- **Django/Python Context**: You might need to test how your Django application behaves with many users accessing patient records or scheduling appointments at once.

Tools: Locust and JMeter can be used for load testing.

7. User Interface (UI) Testing

- **Definition**: Verifying that the user interface is intuitive, responsive, and free of visual defects.
- **HTML/CSS Context**: UI testing checks if your HTML elements (forms, tables, buttons) are displayed correctly across screen sizes and devices. It ensures proper layout, typography, and style consistency.

Example: Ensuring the patient registration form is displayed correctly and is accessible.

Tools: Use Selenium or Playwright for automated UI testing.

8. API Testing

- **Definition**: Verifying the correct operation of your application's API endpoints (such as a REST API).
- **Django Context**: If your hospital management system has an API (using Django Rest Framework), you'll want to test endpoints like /api/patients/, /api/appointments/, and /api/users/.

Example:

python
Copy code

```
from rest_framework.test import APIClient
from rest_framework import status

class PatientApiTest(TestCase):
    def setUp(self):
        self.client = APIClient()
        self.patient = Patient.objects.create(first_name="Ruhul Amin", last_name="Lalan")

    def test_get_patient_list(self):
        response = self.client.get('/api/patients/')
        self.assertEqual(response.status_code, status.HTTP_200_OK)
        self.assertContains(response, "Ruhul Amin Lalan")
```

9. Usability Testing

- **Definition**: Testing the application to ensure it is user-friendly and meets the needs of the hospital staff and patients.
- **HTML/CSS Context**: Focuses on whether the web application is intuitive, easy to navigate, and has a well-organized layout.

Example: Testing if the patient portal allows easy access to medical records, appointment booking, etc.

10. Load Testing

- **Definition**: Testing how well the system performs under heavy usage or traffic.
- **Django/Python Context**: For example, testing how the system behaves when multiple doctors or patients are logged in and performing operations like viewing medical histories or making appointments.

Tools: Locust or Apache JMeter can simulate heavy loads on the application to test scalability.

6.3 Performance Testing Report

The dashboard generator is a modular extension of JMeter. Its default behavior is to read and process samples from CSV files to generate HTML files containing graph views. It can generate the report at end of a load test or on demand.

This report provides the following metrics

A request summary graph showing the Success and failed requests (Transaction Controller Sample Results are not taken into account) percentage:

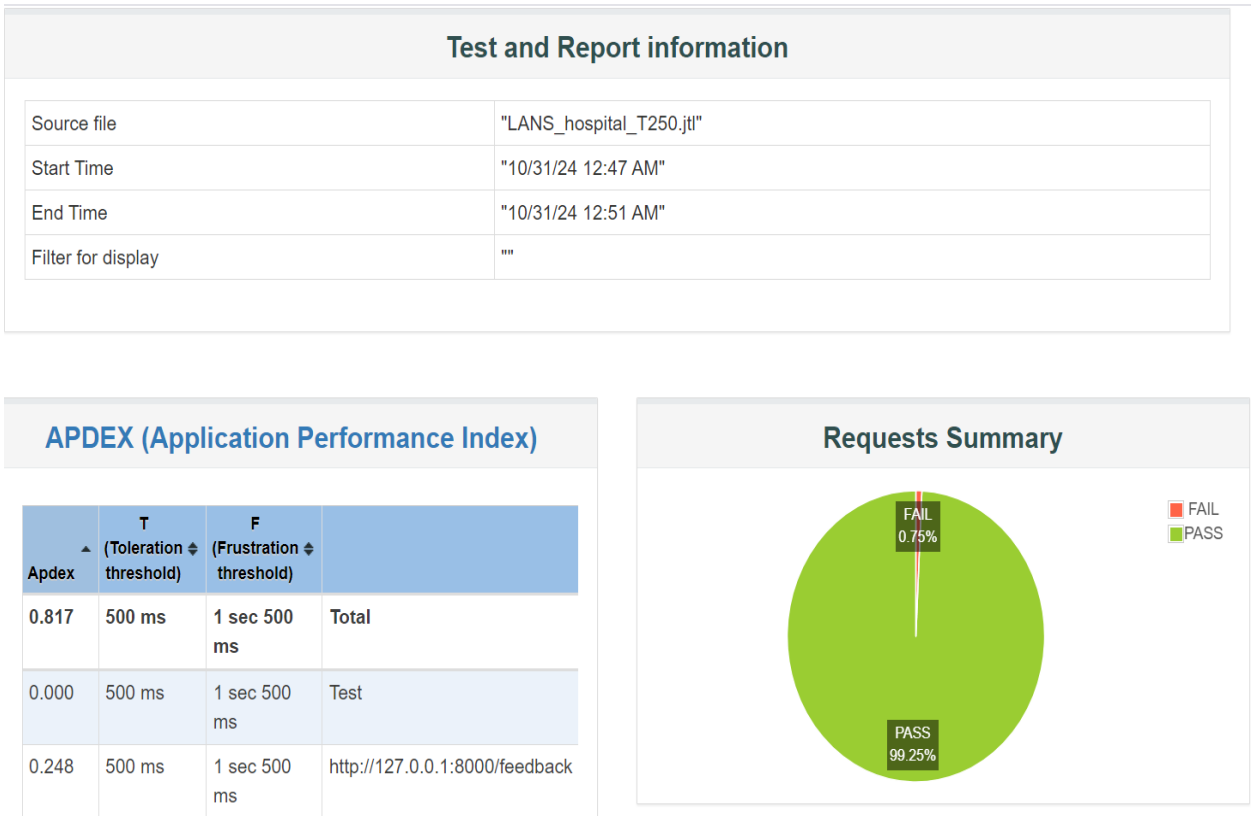
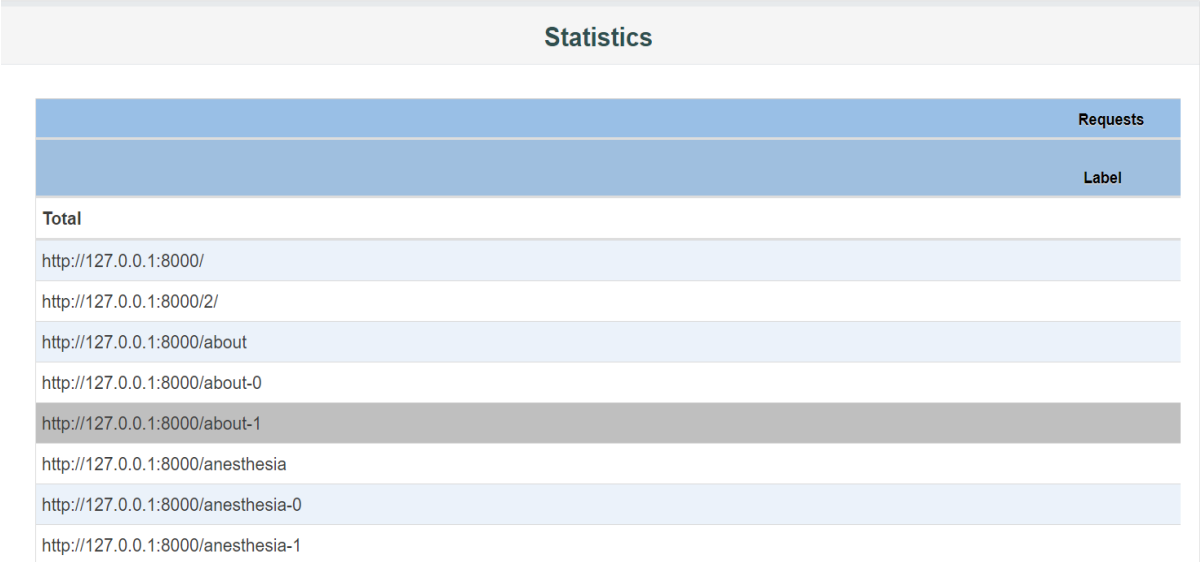


Fig 6.1: Testing and Report Information

❖ **Statistics:** List of API

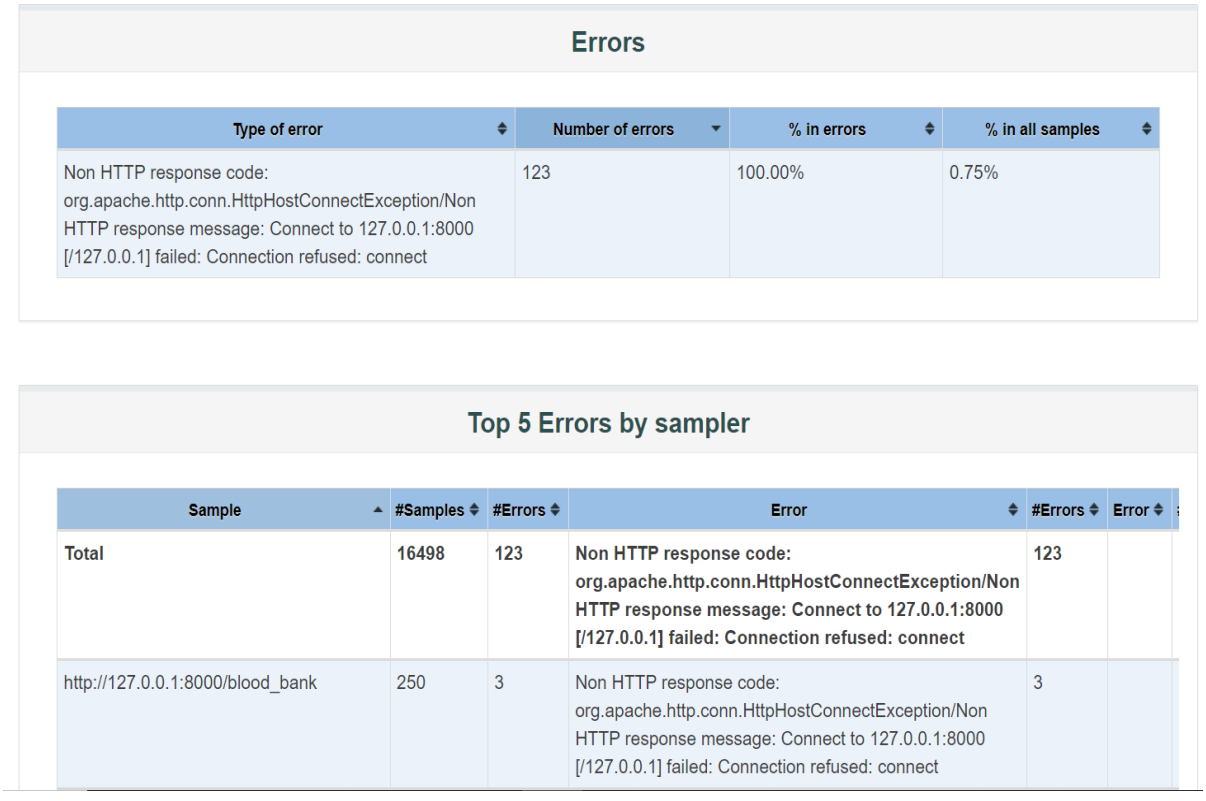


Statistics

Label	Requests
Total	
http://127.0.0.1:8000/	
http://127.0.0.1:8000/2/	
http://127.0.0.1:8000/about	
http://127.0.0.1:8000/about-0	
http://127.0.0.1:8000/about-1	
http://127.0.0.1:8000/anesthesia	
http://127.0.0.1:8000/anesthesia-0	
http://127.0.0.1:8000/anesthesia-1	

Fig 6.2: List of API

❖ **Error:** An error table providing a summary of all errors and their proportion in the total requests



Errors

Type of error	Number of errors	% in errors	% in all samples
Non HTTP response code: org.apache.http.conn.HttpHostConnectException/Non HTTP response message: Connect to 127.0.0.1:8000 [/127.0.0.1] failed: Connection refused: connect	123	100.00%	0.75%

Top 5 Errors by sampler

Sample	#Samples	#Errors	Error	#Errors	Error
Total	**16498**	**123**	**Non HTTP response code: org.apache.http.conn.HttpHostConnectException/Non HTTP response message: Connect to 127.0.0.1:8000 [/127.0.0.1] failed: Connection refused: connect**	**123**	
http://127.0.0.1:8000/blood_bank	250	3	Non HTTP response code: org.apache.http.conn.HttpHostConnectException/Non HTTP response message: Connect to 127.0.0.1:8000 [/127.0.0.1] failed: Connection refused: connect	3	

Fig 6.3: Error

❖ **Response Times Percentiles Over Time (successful responses only)**

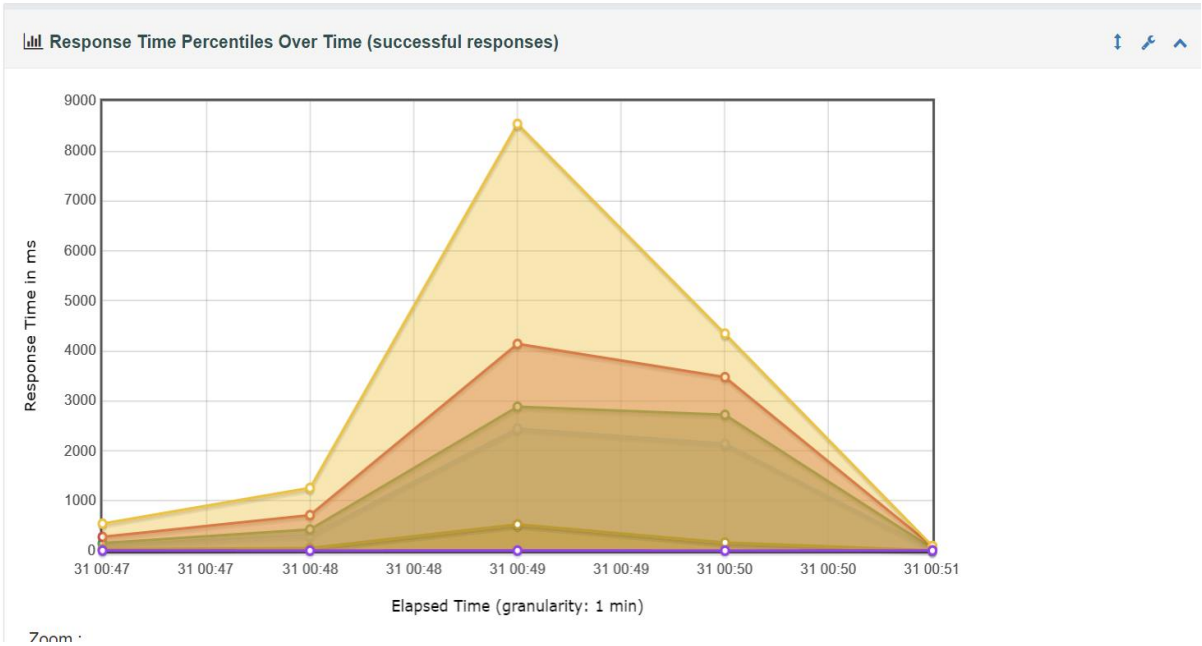


Fig 6.4: Response Times Percentiles Over Time

❖ **Active Threads Over Time**

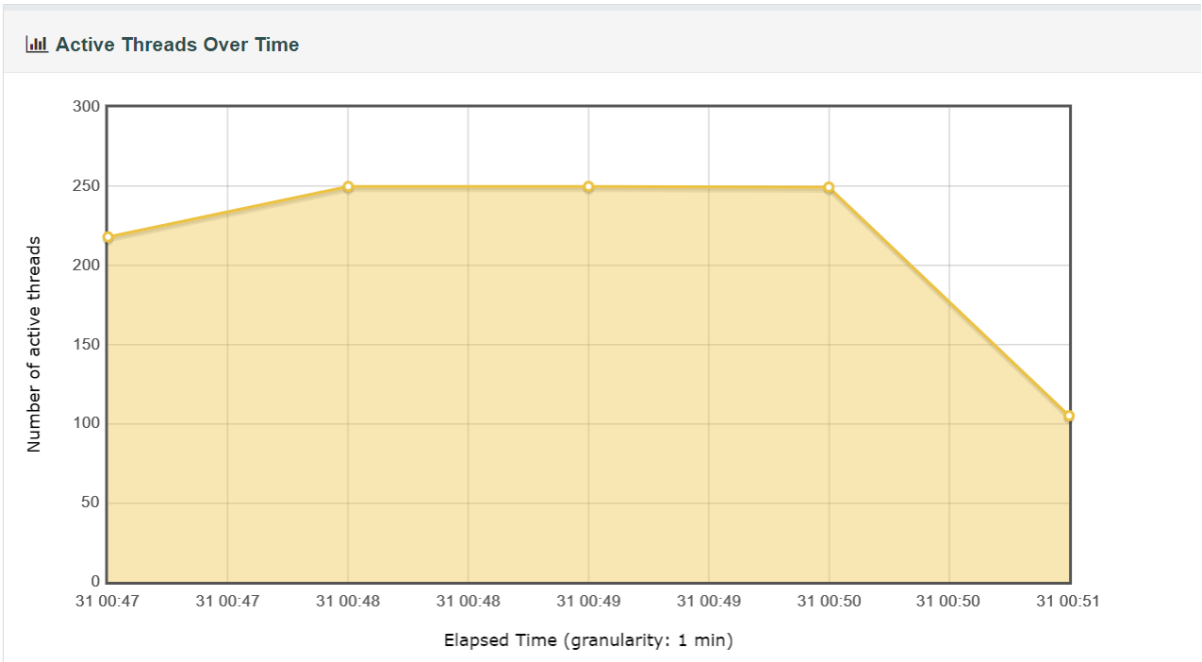


Fig 6.5: Active Threads Over Time

Result:

In our Project we seeing that per ten second 200-250 user can use this web side. Our planning is in future per ten second 500-1000 user will can use this web side.

CHAPTER 7

PROJECT OVERVIEW

7.1 Introduction

In the fast-paced world of healthcare, managing hospital operations efficiently is crucial for ensuring quality patient care. This project focuses on developing a Hospital Management System (HMS) using a combination of HTML**,** CSS**,** MySQL**,** Python, and Django. The system aims to streamline hospital operations, improve patient and staff interaction, and offer administrative tools for better management.

7.2 System Design and Architecture

- **Front-End**:
  - The front-end will use HTML and CSS to create an interactive and visually appealing interface.
  - The CSS framework (e.g., Bootstrap) will be used to ensure the website is responsive, providing a seamless user experience across desktops, tablets, and mobile devices.
  - The front-end will include various pages such as patient registration**,** appointment booking**,** staff management**,** and patient billing**.**
- **Back-End**:
  - The back-end of the system will be built using Django, a Python-based web framework known for its simplicity, scalability, and security.
  - Django will handle all business logic, including data retrieval, processing, and storage.
  - The back-end will include models for managing hospital entities such as patients**,** staff**,** appointments**,** medical history**,** billing, and inventory.
- **Database**:
  - MySQL will be used as the relational database to store all relevant data related to the hospital, such as patient information**,** doctor details**,** appointments**,** medication, and billing records**.**
  - The database will be designed to maintain data integrity, provide efficient data queries, and support scalability as the hospital grows.
- **Security**:
  - The system will incorporate user authentication and role-based authorization to ensure that sensitive data is accessible only to authorized personnel.
  - Django's built-in features such as Django Rest Framework (for APIs) and CSRF protection will be used to secure user interactions.

7.3 Technology Stack

- **HTML & CSS**: For the structure and styling of the web pages, creating an interactive user interface.
- **Bootstrap** (optional): To ensure responsiveness and a modern look for the hospital management system.
- **JavaScript** (optional): To add interactivity to the pages and enhance the user experience (e.g., form validation, dynamic content).

- **MySQL**: A relational database management system (RDBMS) to store hospital records and data related to patients, doctors, appointments, and billing.
- **Python**: The programming language used to develop the back-end logic of the system, working with Django.
- **Django**: A high-level Python web framework for rapid development, ensuring clean, scalable, and secure code.
- **Django Rest Framework** (optional): For creating APIs to interact with front-end interfaces or mobile applications if necessary.

7.4 Implementation Strategy

- **Phase 1: Requirement Gathering and Analysis** – Collect hospital requirements, determine key features, and define user roles and permissions.
- **Phase 2: System Design** – Create wireframes and mockups for the front-end, design the database schema for MySQL, and define the back-end models using Django.
- **Phase 3: Front-End Development** – Develop the website's structure using HTML and CSS, and integrate the design with Bootstrap for responsiveness.
- **Phase 4: Back-End Development** – Implement Django views, models, and templates to handle business logic and interact with the MySQL database.
- **Phase 5: Integration and Testing** – Integrate the front-end with the back-end and perform unit testing, integration testing, and user acceptance testing to ensure that all features work as intended.
- **Phase 6: Deployment and Documentation** – Deploy the system to a web server and provide detailed documentation for users and administrators.

7.5 Overview

1. Home Page

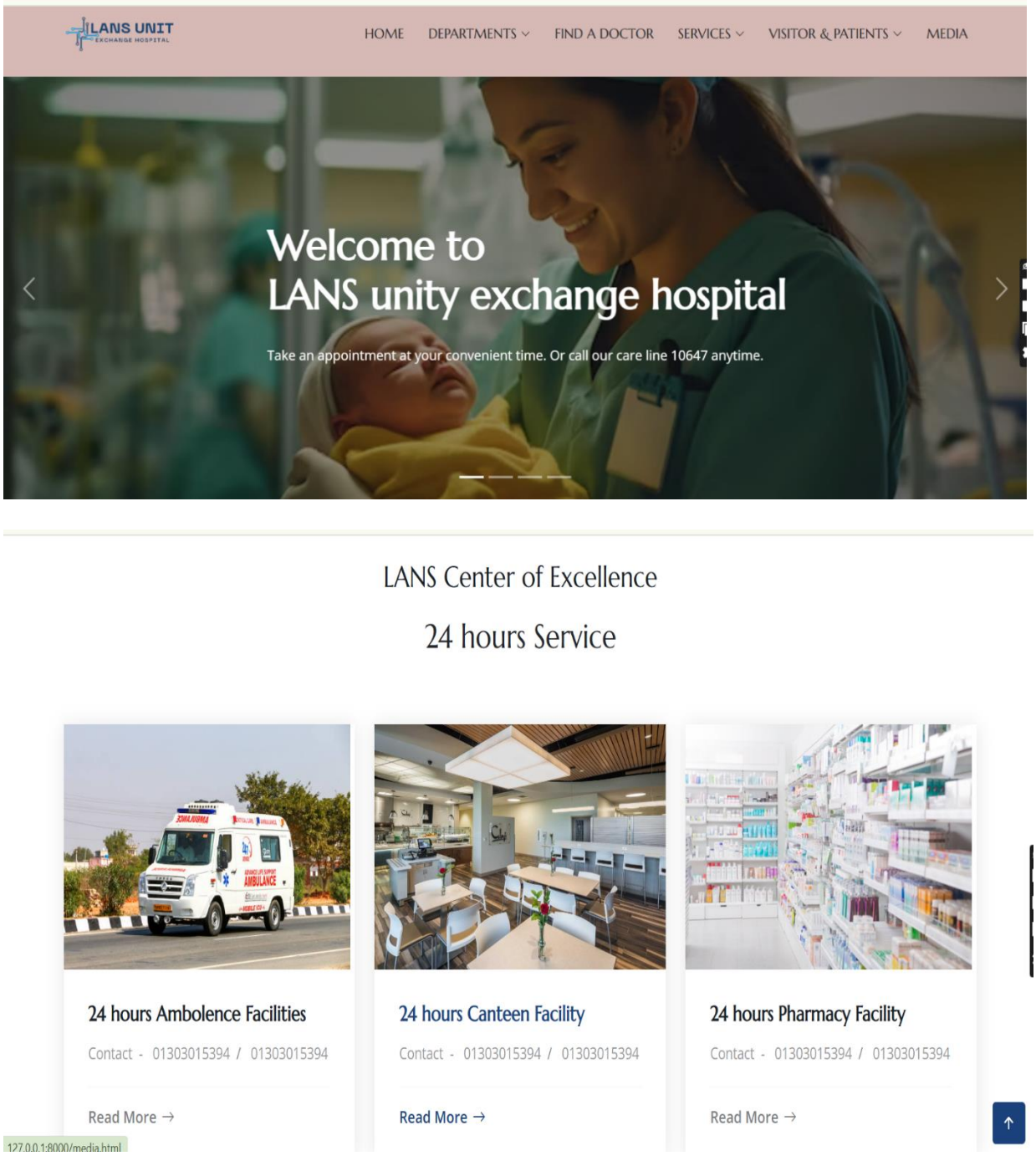
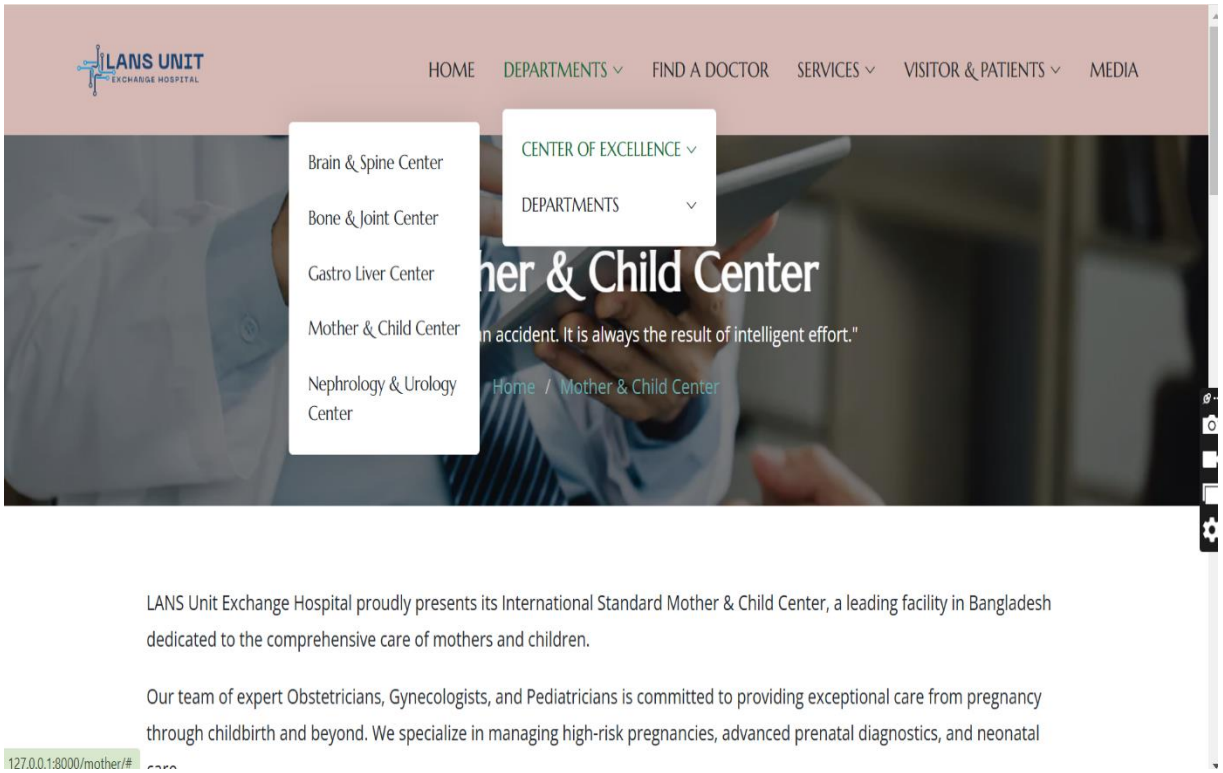


Fig 7.1: Home Page

2. Departments



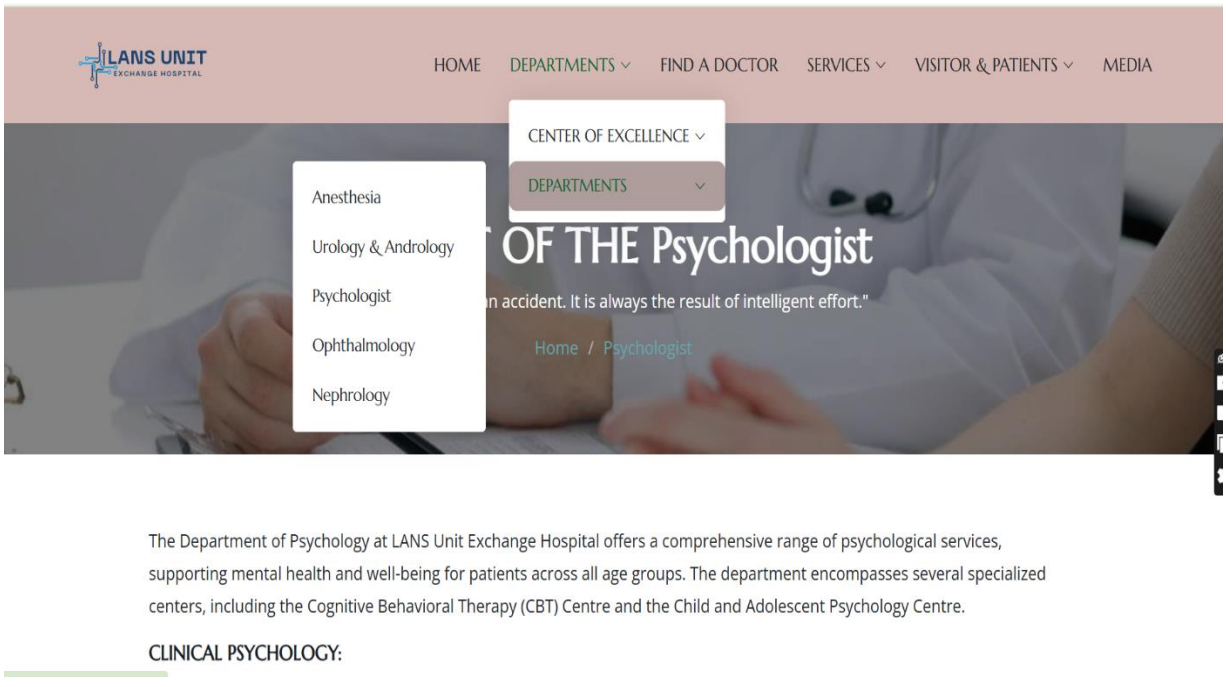


Fig 7.2: Departments

3. Find Doctor

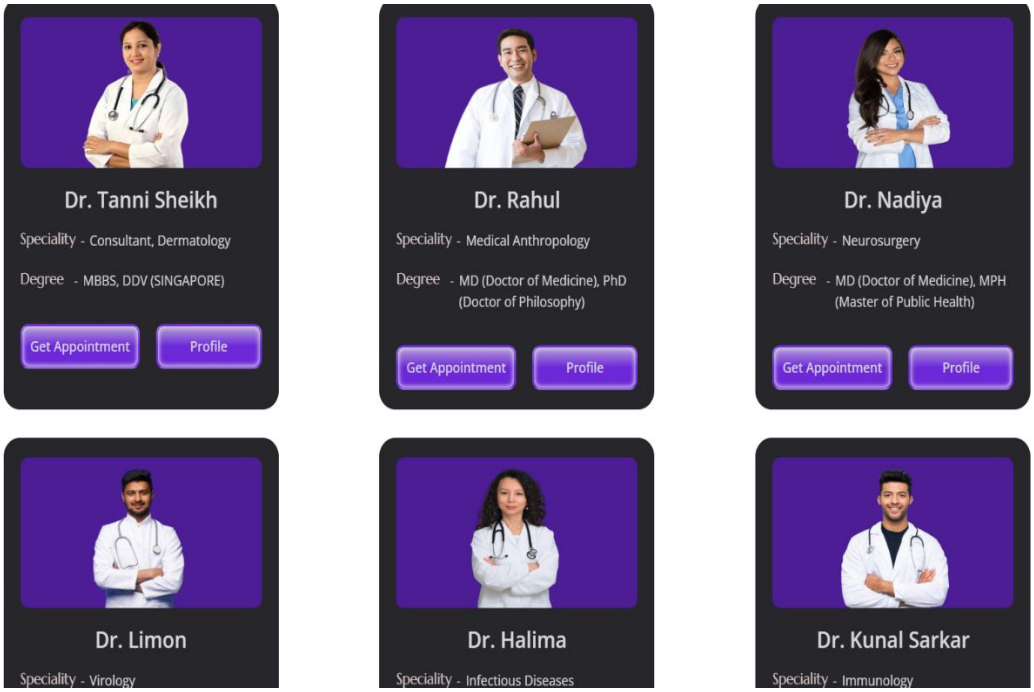


Fig 7.3: Find Doctor

4. Search Doctor

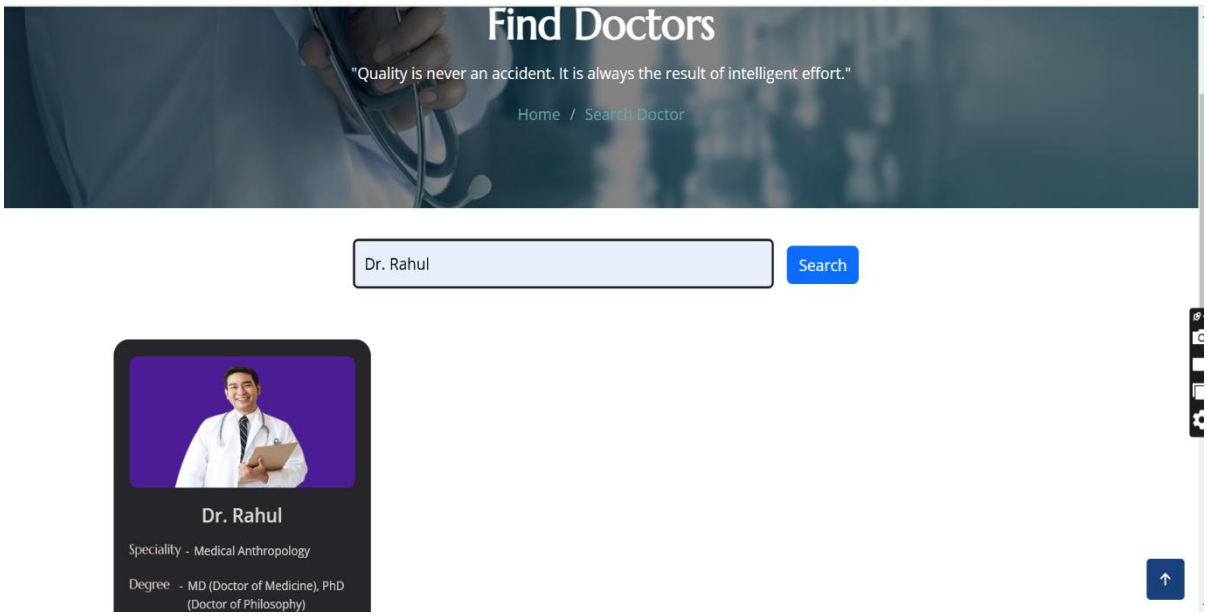


Fig 7.4: Search Doctor

5. Doctor Profile

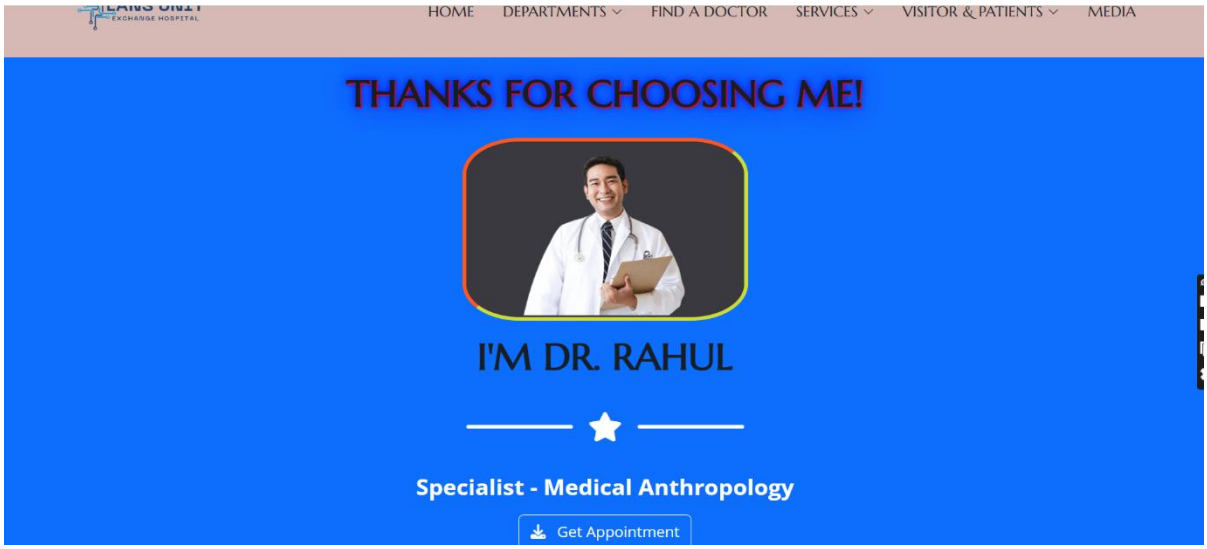


Fig 7.5: Doctor Profile

6. Doctor Appointment

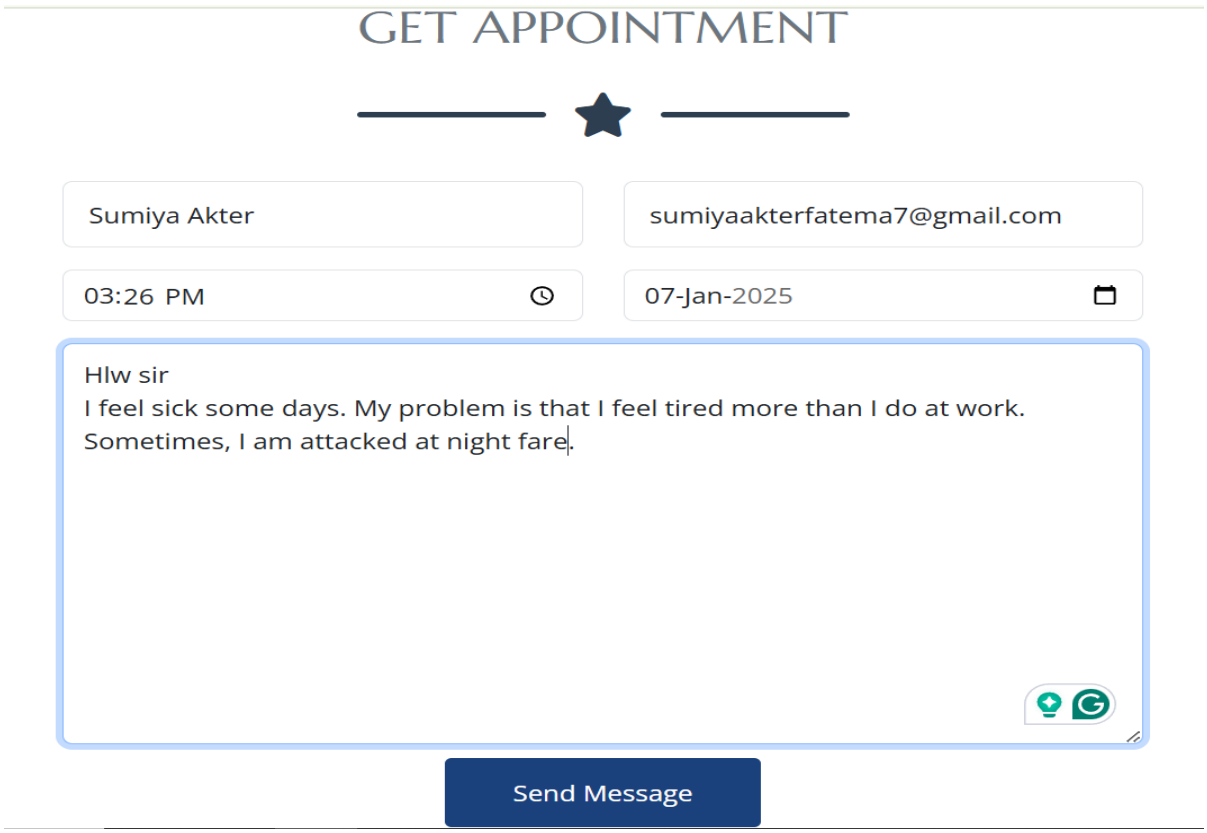


Fig 7.6: Doctor Appointment

7. Services

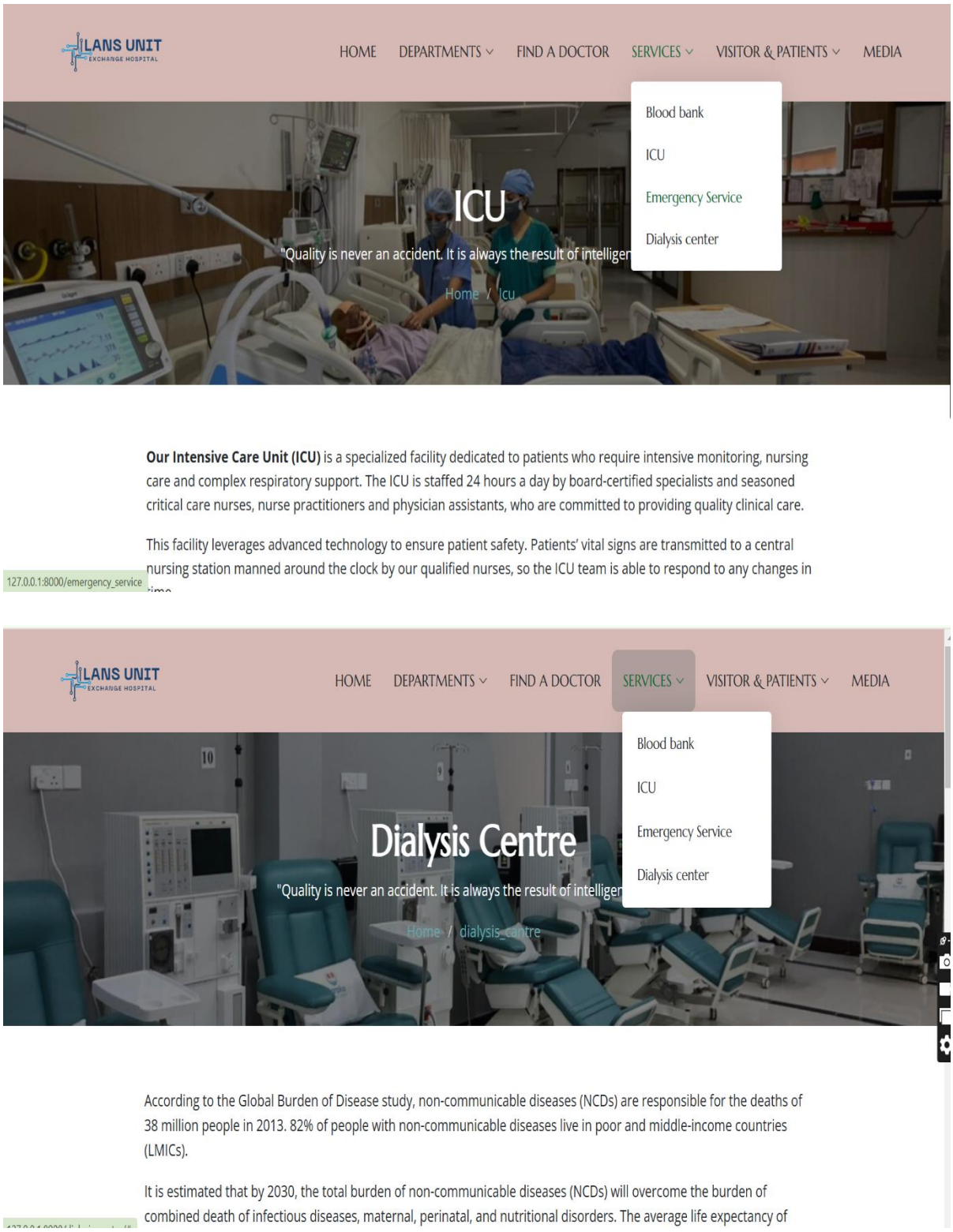


Fig 7.7: Services

8. Patient Feedback

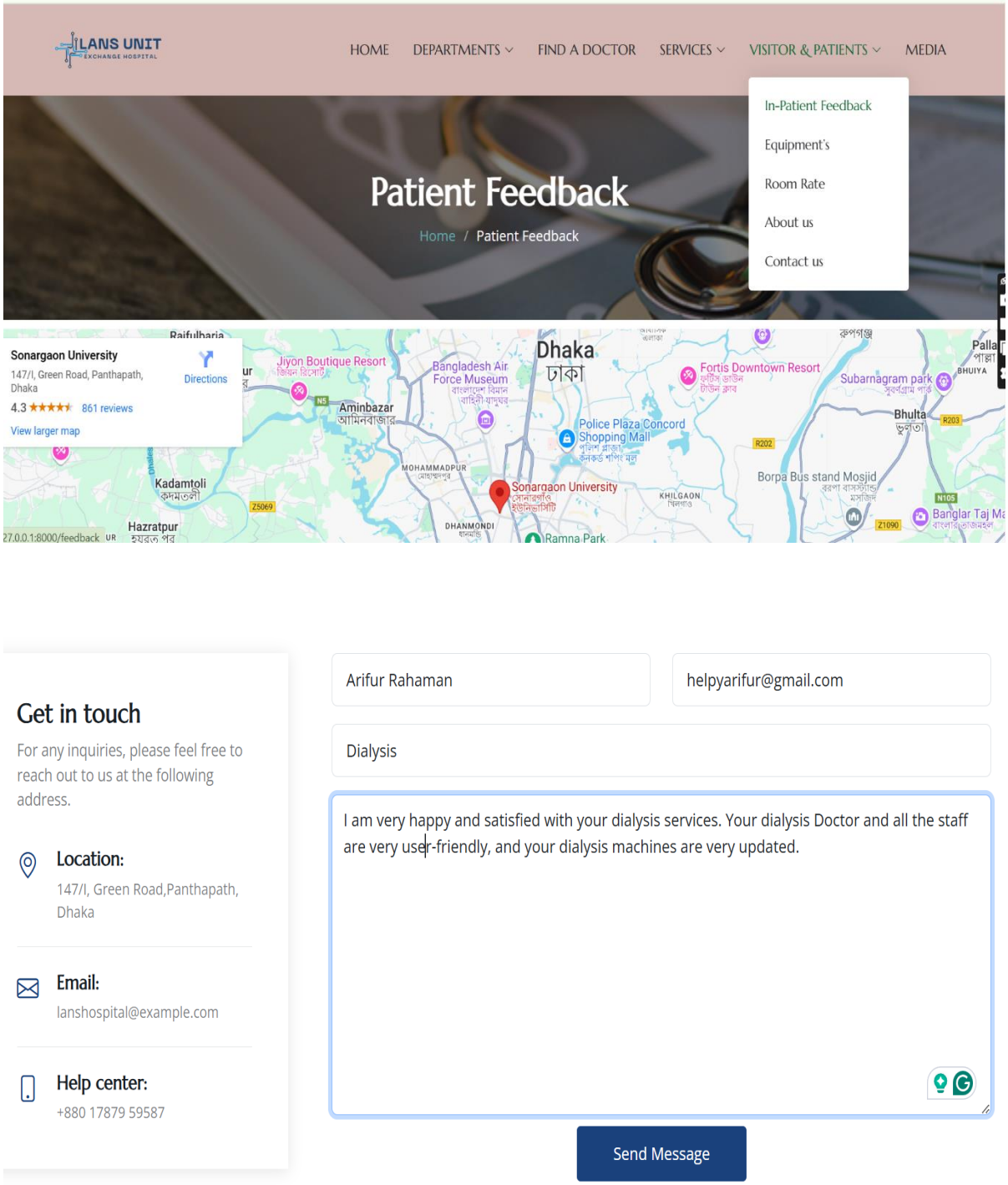


Fig 7.8: Patient Feedback

9. Equipment's

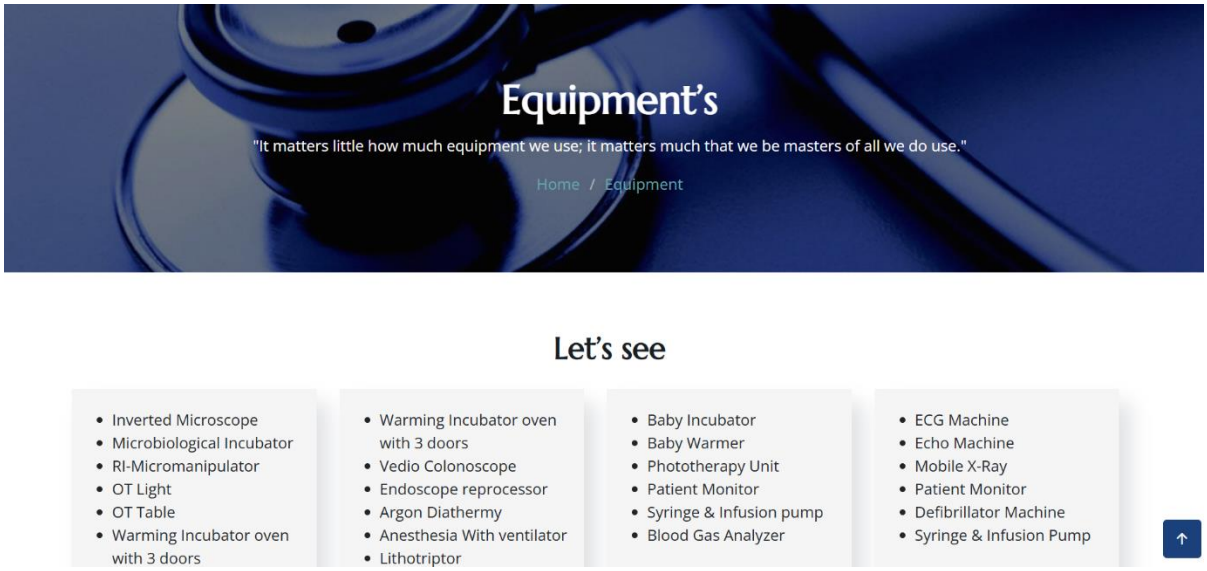
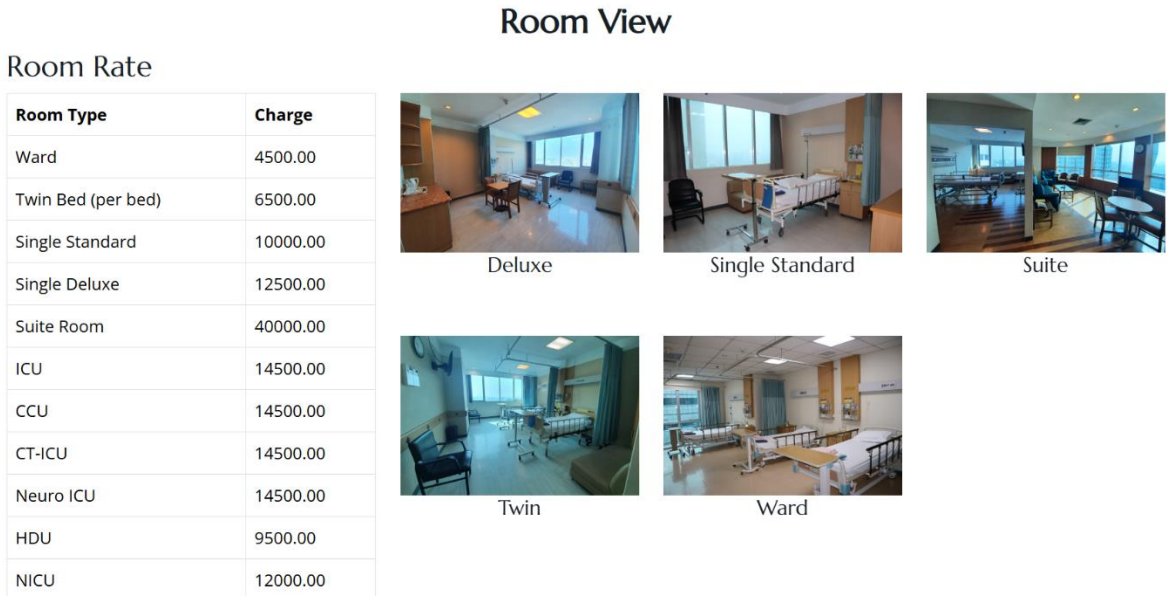


Fig 7.9: Equipment's

10. Room Rate



Room Type	Charge
Ward	4500.00
Twin Bed (per bed)	6500.00
Single Standard	10000.00
Single Deluxe	12500.00
Suite Room	40000.00
ICU	14500.00
CCU	14500.00
CT-ICU	14500.00
Neuro ICU	14500.00
HDU	9500.00
NICU	12000.00

Fig 7.10: Room Rate

11. Room Booking

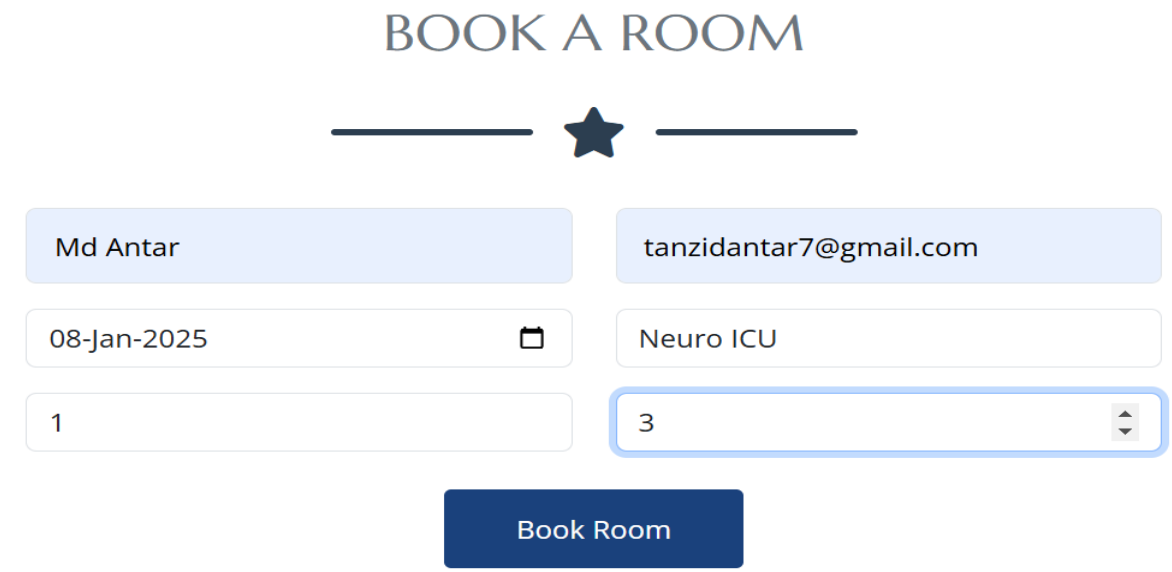


Fig 7.11: Room Booking

12. Contacts

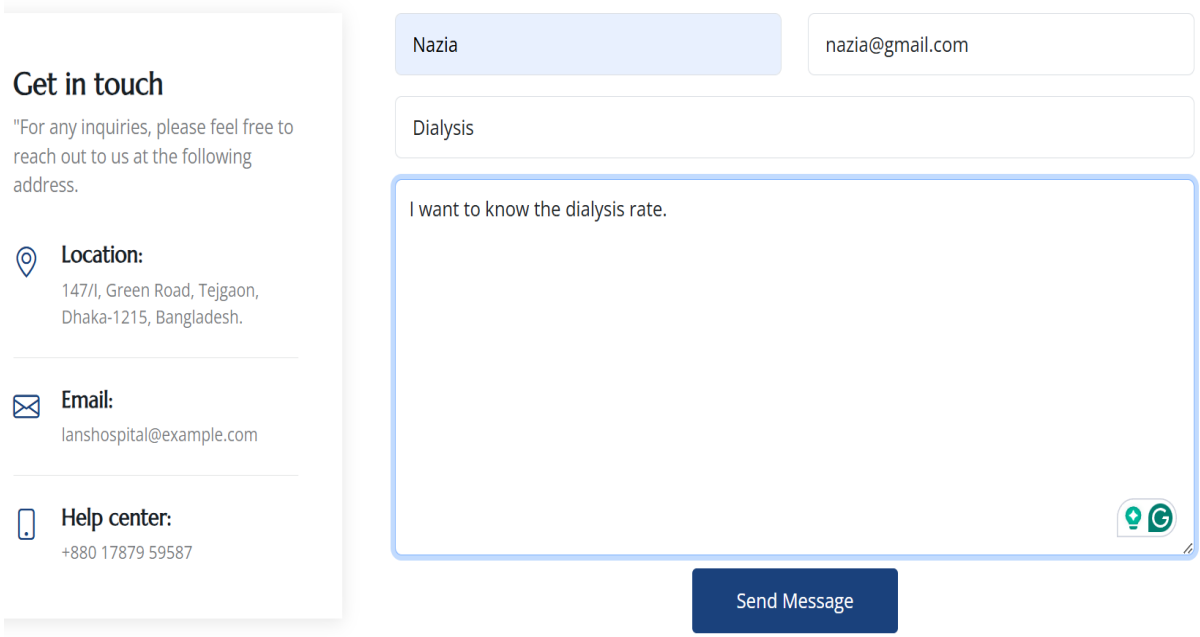


Fig 7.12: Contacts

13. Media Gallery

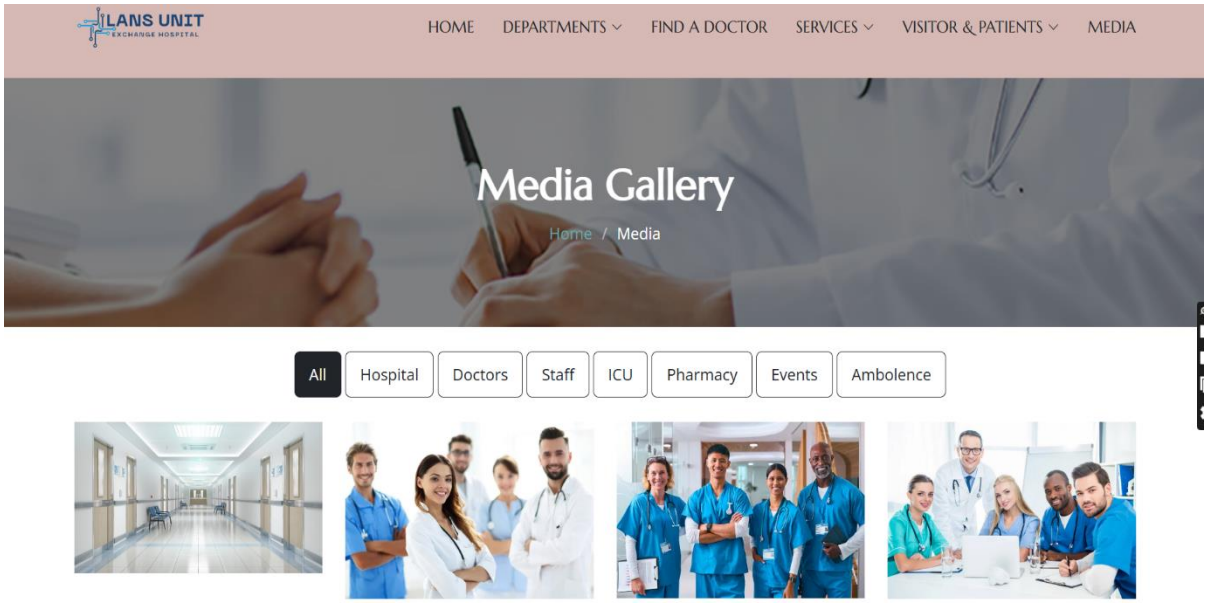


Fig 7.13: Media Gallery

14. Information

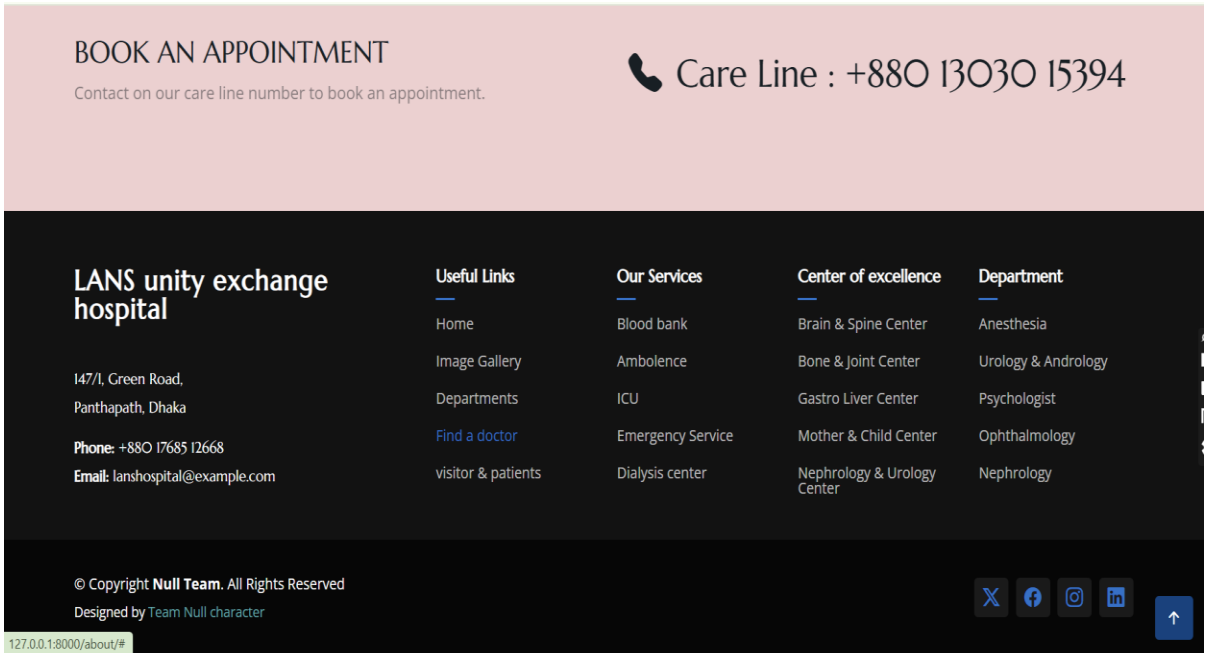


Fig 7.14: Information

15. About Us

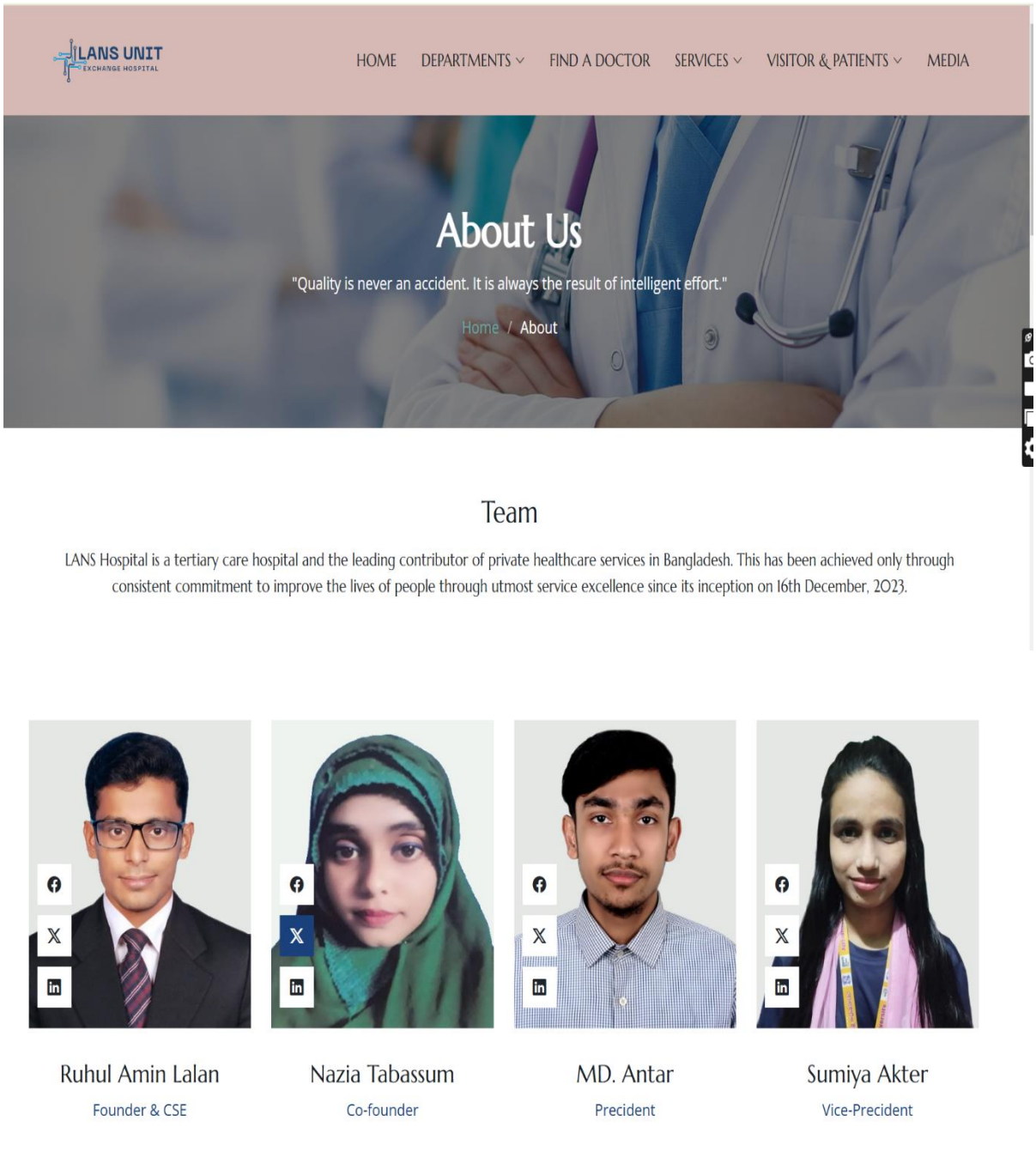


Fig 7.15: About Us

16. Admin Panel of Login Page

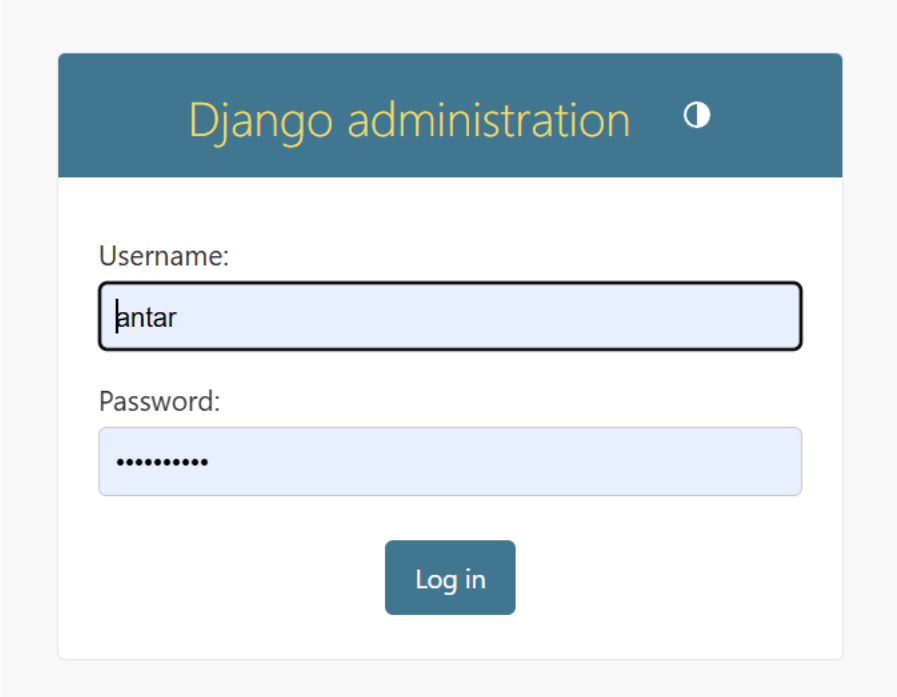


Fig 7.16: Login page

17. Admin Panel

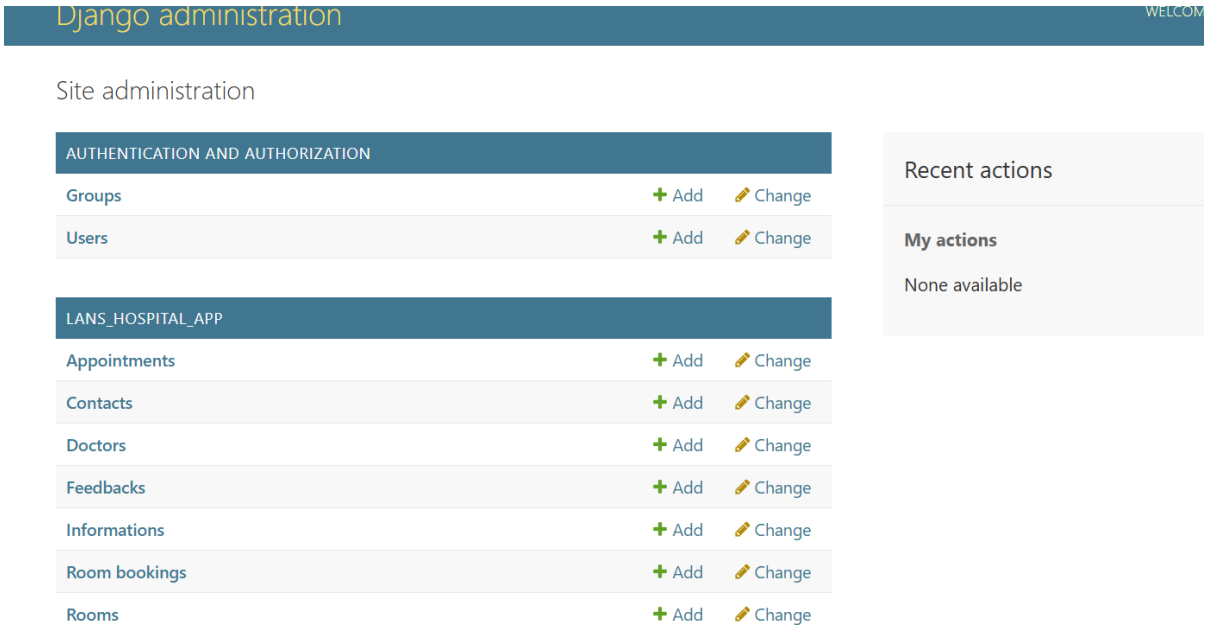


Fig 7.17: Admin Panel

18. User of Admin

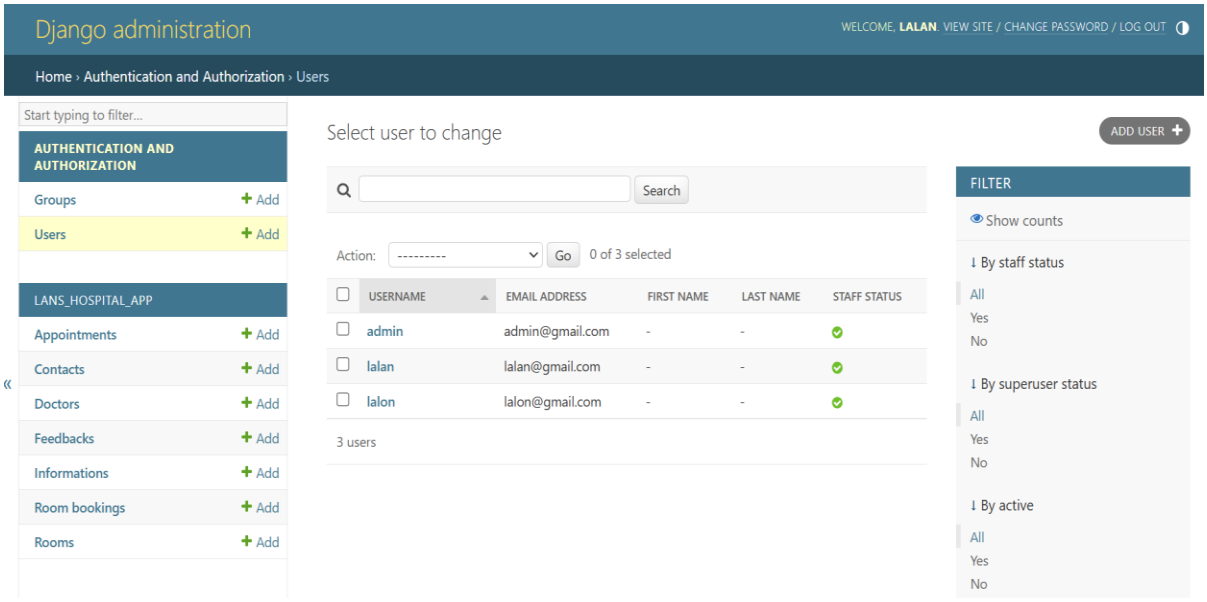


Fig 7.18: User of Admin

19. Doctor Add and Changes

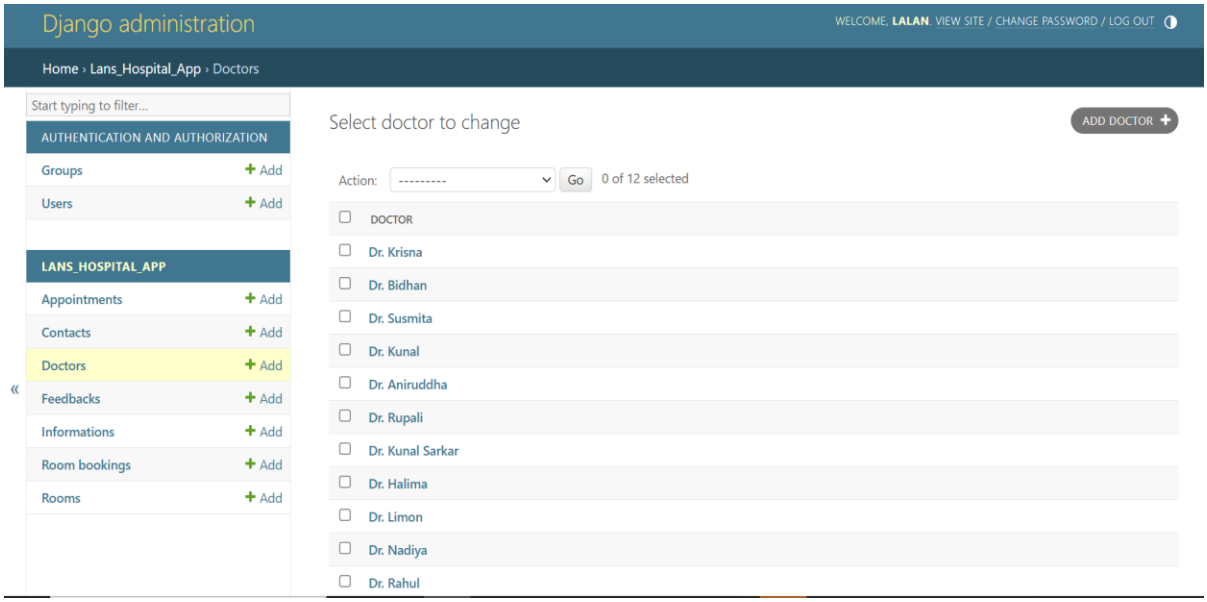


Fig 7.19: Doctor Add and Changes

20. Appointment of Patient

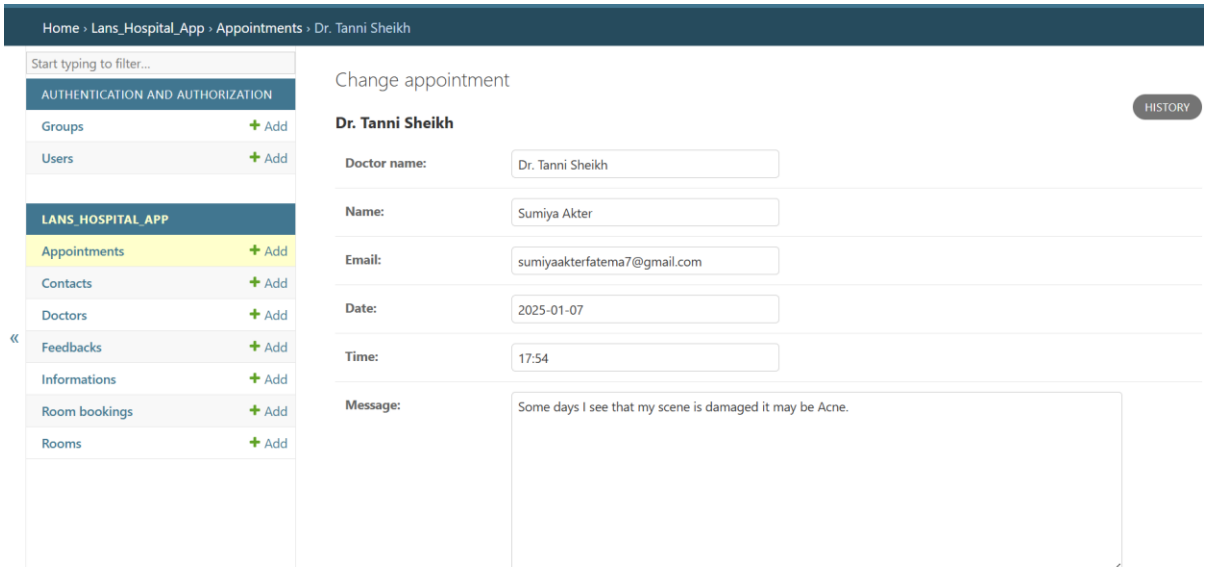


Fig: 7.20: Appointment of Patient

21. Room Booking List of Admin Panel

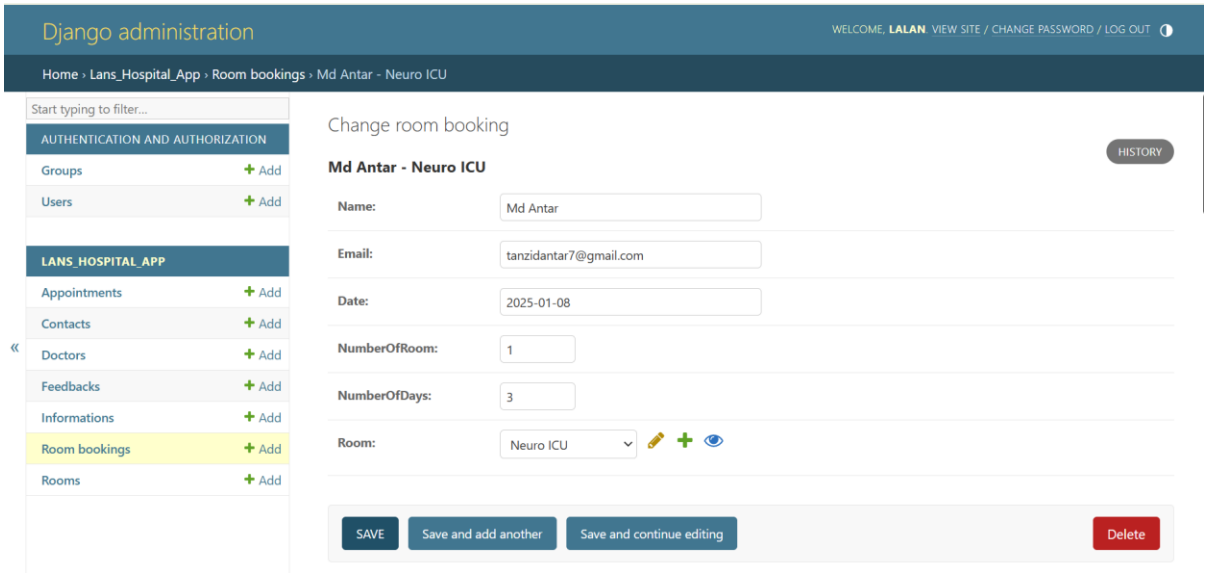


Fig 7.21: Room Booking List of Admin Panel

CHAPTER 8

CONCLUSION

8.1 Conclusion

The Hospital Management System (HMS) developed is a comprehensive solution aimed at automating and streamlining various hospital functions such as patient registration, appointment scheduling, doctor-patient interaction, and billing. The system's implementation has successfully demonstrated the potential of modern web technologies in creating efficient, secure, and user-friendly applications tailored for the healthcare sector. In this project, we developed a web-based hospital management application that can efficiently manage patient data, track appointments, and handle billing. Using HTML and CSS, the system provides an intuitive and responsive user interface, ensuring a seamless experience across devices. The back-end logic was handled by Python using the Django framework, allowing for rapid development and ensuring secure data processing. The system also incorporates role-based authentication for different users, such as administrators, doctors, and patients, providing customized access to various features based on user roles.

8.2 Future Scope of the Project

1. Integration with Emerging Technologies

Artificial Intelligence (AI): AI can be used to improve diagnostic accuracy, predict patient outcomes, personalize treatment plans, and assist in administrative tasks like scheduling and billing. Machine Learning (ML): ML algorithms can help in predictive analytics, such as identifying patients at risk of certain conditions or readmissions, thus enhancing preventive care. Natural Language Processing (NLP): NLP can assist in converting clinical notes and medical records into structured data for better analysis.

2. Cloud Computing and Data Storage

Cloud-based HMS platforms will enable easier data access, seamless updates, and the ability to scale services more effectively. This can help hospitals reduce costs and improve accessibility for both patients and healthcare professionals.

3. Block chain for Data Security and Privacy

Block chain technology can enhance data security and patient privacy. Its decentralized ledger can make medical records tamper-proof, ensuring that patient data is secure and transparent across all touchpoints.

4. Robotic Process Automation (RPA)

RPA can automate administrative tasks such as billing, claims processing, and patient data entry, thus reducing human error and improving efficiency in hospital operations.

5. Patient-Centered Care & Personalized Health Records

Personalized care, based on a patient’s medical history, preferences, and specific needs, will become more prominent in future HMS solutions.

6. Smart Hospital Infrastructure
Integration of IOT (Internet of Things) will create smarter hospital environments, where equipment, rooms, and resources are connected for better patient management, energy optimization, and operational efficiency.

7. Artificial Intelligence for Diagnostics and Clinical Decision Support
Integration of AI for automated diagnostics, such as imaging analysis (e.g., radiology), can assist clinicians in making quicker and more accurate decisions. Decision support systems will also improve by offering evidence-based recommendations for patient care.

8. Global Health Initiatives
As the world becomes more interconnected, HMS platforms may evolve to support global health initiatives, providing data-sharing capabilities for international health organizations and governments.

8.3 Limitation OF Project

While a Hospital Management System (HMS) offers numerous benefits in terms of improving efficiency, managing patient information, and streamlining hospital operations, there are several limitations that could affect its implementation, usage, and maintenance. In our project has same limitations there are:

- Online Payment Issue
- Blood connection by Online System
- Online Treatment System.
- Artificial Intelligence System

Achievements
The project includes the following key features:

- **Patient Management**: Enables the hospital staff to add, update, and manage patient information.
- **Appointment Scheduling**: Allows patients to book appointments with doctors and view available time slots.
- **Billing System**: Automates billing for treatments and services provided to patients.
- **User Authentication**: Ensures secure login for different users (admin, doctor, patient) with proper access control. The project successfully integrates front-end (HTML, CSS) and back-end (Python, Django) technologies to provide a seamless user experience.

- Django's robust Model-View-Template (MVT) architecture allowed for the easy management of the application's data, business logic, and presentation layer.
- The application was designed to be scalable, ensuring that it can be expanded to include additional features in the future, such as medical record management or integration with external healthcare systems.
- The system was tested for functionality, security, and user experience, and all major functionalities, such as adding patients, scheduling appointments, and generating bills, were successfully implemented.

REFERENCES

[1] Raggett, Dave, et al. *Raggett on HTML 4*. Addison-Wesley Longman Publishing Co., Inc., 1998.

[2] Lewis, Joe, and Meitar Moscovitz. *Advanced CSS*. Après, 2009.

[3] Lutz, M., 2001. *Programming python*. " O'Reilly Media, Inc.".

[4] Burch, C., 2010. Django, a web framework using python: Tutorial presentation. *Journal of Computing Sciences in Colleges*, *25*(5), pp.154-155.

[5] Letkowski, J., 2015. Doing database design with MySQL. *Journal of Technology Research*, *6*, p.1.

[6] Crockford, D., 2008. *JavaScript: The Good Parts: The Good Parts*. " O'Reilly Media, Inc.".

[7] Lv, T., Yan, P. and He, W., 2018, August. Survey on JSON data modelling. In *Journal of Physics: Conference Series* (Vol. 1069, No. 1, p. 012101). IOP Publishing.

[8] Boshernitsan, M. and Downes, M.S., 2004. *Visual programming languages: A survey*. Los Angeles, CA, USA: Computer Science Division, University of California.

[9] Bhattacharyya, D.K., 2006. *Research methodology*. Excel Books India.

[10] Ravi, S.Y., Jindal, R. and Anand, S., 2020. A Study on Comparison of UML and ER Diagram. *Int. Res. J. Eng. Technol*, *7*(5), pp.7575-7580.