Kibur College

Department of Computer Science

Project On: **PATIENT RECORD MANAGEMENT FOR DR.SEIDNUR DENTAL CLINIC**

Final project document

Submitted to department of Computer Science in partial fulfilment of the requirements for the degree of Bachelor Science in Computer Science

By:

Group members

1. Eneb Hussen (URC 104/14)
2. Leul Tesfu (URC 141/14)
3. Muhaba Ahmedteyib (URC 110/14)
4. Sami Fahmi (URC 115/14)

Advisor name: Girmay .G

July, 2025

Addis Ababa, Ethiopia

Project On: PATIENT RECORD MANAGEMENT FOR DR.SEIDNUR DENTAL CLINIC

Final project document

Submitted to department of Computer Science and in partial fulfilment of the

Requirements for the degree of Bachelor Science in Computer Science

By:

1. Eneb Hussen (URC 104/14)
2. Leul Tesfu (URC 141/14)
3. Muhaba Ahmedteyib (URC 110/14)
4. Sami Fahmi (URC 115/14)

Advisor name_____ Signature_____

Examiner name_____ Signature_____

Abstract

In Ethiopia, healthcare facilities particularly at the community level often face significant challenges in managing patient records, scheduling appointments, and maintaining efficient clinical workflows. These issues are largely due to the continued reliance on manual processes and fragmented information systems, leading to data loss, reduced service quality, and limited access to reliable medical information. To address these gaps, we designed and implemented a Patient Record Management System (PRMS) tailored for community clinics. The primary objective is to develop a secure, user-friendly digital platform to streamline patient registration, appointment scheduling, treatment documentation, and medical reporting. Using an Agile development methodology, the system is built incrementally with ongoing feedback from healthcare professionals to ensure practicality and relevance.

The PRMS supports multiple user roles including administrators, doctors, and receptionists through role-based access control. Core features include digital patient record storage, appointment and treatment tracking, reporting tools, and audit logs for accountability. Technically, the system is implemented as a web-based application using React.js, Node.js, Express.js, and MongoDB, with secure access provided via JWT authentication. Initial deployment has demonstrated improvements in clinic workflow efficiency, reduced paperwork, and enhanced data integrity. By centralizing patient data and automating routine processes, the PRMS contributes to better healthcare delivery and more accurate medical documentation. The platform is designed with scalability in mind and can be extended with additional modules or integrated with broader health information systems.

Keywords: Patient Record Management System, Digital Health Records, Appointment Scheduling System Role-Based Permissions, Administrative Dashboard, User Authentication

Acknowledgment

First, we are thankful to God for giving us the strength, stamina, and health that allowed us to start and finish this project. Without His help, none of this would have been possible. We sincerely thank our advisor Mr.Girmay for helping us with this task from beginning to end. With your help, we were able to stay on course, correct our mistakes, and gain a better understanding of the technical aspects. Every suggestion you gave us made our project better. We also thank Dr. Seidnur Dental Clinic for allowing us to use their workspace as the model for our system. Their support, openness, and helpful guidance allowed us to better understand how a dental clinic actually operates. This helped us develop a useful system. We appreciate the help of our instructors and department staff, who gave us the materials and background knowledge we needed to build our project. Because of the lectures, homework, and lab work we did during our study, we were prepared for this phase. We also want to express our gratitude to our friends and classmates who were always available to us, whether it was to offer advice, answer questions, or simply offer support when things got tough. Finally, we would especially like to thank our families. We appreciate your love, tolerance, and encouragement, especially when we were worn out or angry. Even on long nights and difficult days, your faith in us kept us going.

List of abbreviation

API	Application Programming Interface
CI/CD	Continuous Integration / Continuous Deployment
CORS	Cross-Origin Resource Sharing
CRUD	Create, Read, Update, Delete
CSV	Comma-Separated Values
DBMS	Database Management System
ERD	Entity Relationship Diagram
HTTPS	Hypertext Transfer Protocol Secure
IDE	Integrated Development Environment
JWT	JSON Web Token
MS Word	Microsoft Word
NPM	Node Package Manager
PRMS	Patient Record Management System
QA	Quality Assurance
REST API	Representational State Transfer API
SMS	Short Message Service
UI	User Interface
UAT	User Acceptance Testing
UX	User Experience
VS Code	Visual Studio Code

List of Tables

List of Figures

Table of Content

Chapter One Introduction

1.1 Background of the Organization

Dr. Seidnur Dental Clinic is a privately operated healthcare center offering general dental care to individuals in the nearby community. It provides various services such as dental examinations, cleanings, tooth removals, fillings, and other essential oral care procedures. The clinic has experienced a rise in patient numbers over the years, making it more challenging to handle patient information with the existing manual system.

Currently, the clinic depends on a paper system for maintaining patient records, appointment logs, and treatment histories. Upon a patient's arrival, receptionists jot down the patient's information using notebooks or paper forms. Doctors frequently write treatment notes by hand and store them in folders sorted by patient name or appointment date. Appointments are arranged using paper planners or basic daily sheets.

This conventional system might appear straightforward, yet it leads to numerous issues as the clinic expands. Crucial patient documents may become misplaced, harmed, or even vanish over time. Employees frequently invest several minutes sifting through stacks of documents to locate the correct patient file. Appointments may be overlooked or double-scheduled due to the absence of automated notifications or measures to prevent overlaps. Physicians do not always have prompt access to the complete treatment history of the patient they are caring for. This may impact the standard of care and result in delays during peak times.

The absence of a digital system complicates the clinic's ability to monitor daily patient counts, identify the most frequently administered treatments, or track the number of missed appointments. Reports need to be created by hand, which requires time and might include mistakes. Furthermore, the clinic lacks a contingency plan for its paper documents. In the event the clinic experiences fire, flooding, or other hazards, the complete archive of patient records might be destroyed.

With the increasing demand for dental services, it is evident that a quicker, more dependable, and secure system is necessary. A digital solution would simplify the storage and retrieval of patient records for staff, cut down on paperwork, and enable doctors to provide enhanced care.

The objective of this project is to substitute the existing manual procedure with a Patient Record Management System (PRMS) an online platform enabling the clinic to enroll patients, set

appointments, document treatments, and create reports, all from a safe and accessible digital system.

1.2 Statement of the Problem and Justification

1.2.1 Statement of the Problem

Dr. Seidnur Dental Clinic, patient information is still managed by hand with paper forms, folders, and tangible logbooks. Although this method has benefited the clinic for a period, it now imposes significant constraints as the patient count rises. The existing system lacks efficiency, is susceptible to mistakes, and cannot scale. It hampers operations, causes avoidable delays, and jeopardizes patient data.

A significant issue is the loss of data and physical harm. Paper documents can readily be lost, damaged, or ruined by moisture, flames, or careless handling. If a file is lost, the complete treatment history of the patient could be missing, potentially impacting current or future dental treatment.

Another problem is the sluggish and challenging process of obtaining records. When a patient arrives at the clinic, employees need to sift through physical folders to locate their record. This consumes time for both employees and patients, becoming increasingly challenging during peak times or emergencies.

Scheduling appointments is also performed manually with the use of notebooks or wall calendars. This approach frequently results in double bookings or forgotten appointments due to the absence of an automated system for verifying availability or sending reminders. Consequently, the clinic might waste time, encounter misunderstandings, and diminish patient happiness.

The absence of a centralized system results in patient data being difficult for doctors and other staff to access unless they are nearby the file cabinet. This hampers communication and decision-making, particularly when treatment history is required promptly.

A proper reporting system is also absent. Creating monthly reports, summaries of patient visits, or counts of appointments has to be done manually, which is labour-intensive and could result in errors. Without precise data, the clinic is unable to efficiently monitor performance or strategize enhancements.

Finally, the existing system lacks any security features. Anyone who has access to the physical storage can see, alter, or delete patient records without permission. There is no record available to monitor who accessed or changed the information.

1.2.2 Justification

To tackle these issues, it is essential to substitute the paper-based system with a secure, efficient, and reliable Patient Record Management System (PRMS). A digital approach will:

- Facilitate rapid and straightforward patient enrollment.
- Allow physicians and personnel to access and view patient records instantly.
- Avoid schedule clashes with intelligent planning.
- Safely keep treatment records and enable updates with access restrictions.
- Minimize documentation and physical storage area.
- Create reports in only a few clicks, enhancing precision and reducing time.

This system will enhance the clinic's operations, minimize the chance of mistakes, and elevate the overall patient experience. It will additionally ready the clinic for upcoming expansion and facilitate the adoption of additional digital health tools later on.

1.3 Objective of the Project

General Objective

The general objective is to construct a digital patient record management system that will enhance Dr. Seidnur Dental Clinic's patient data management's effectiveness, precision, and accessibility.

Specific Objectives

• To develop functionalities for registering new patients, modifying existing records, and removing outdated or incorrect entries.

• To Design and integrate modules for scheduling appointments, as well as updating or cancelling them as needed.

• To Implement secure user authentication and enforce role-based access control to ensure appropriate system access.

• To enable healthcare providers to create, update, and retrieve patient treatment records efficiently.

• To generate automated reports and visualize key patient and operational data to support decision-making.

• To incorporate comprehensive audit logging mechanisms to track user activities and record modifications for accountability and transparency.

1.4 Methodologies

In this project, we implemented the Agile methodology to create the Patient Record Management System for Dr Seidnur Dental Clinic. The project was segmented into brief sprints, with each concentrating on particular modules like user authentication, patient administration, and appointment scheduling. At the beginning of every sprint, we establish specific objectives and rank tasks according to the use case diagram.

We conducted frequent meetings to assess progress and collect input from prospective users, such as clinic personnel. Following each sprint, we evaluated the new features, resolved bugs, and implemented enhancements before proceeding to the subsequent module. We utilized GitHub for managing versions and collaboration, guaranteeing that every code update was monitored and evaluated.

Frontend components were developed with React.js and designed using CSS Tailwind. Backend APIs were created using Node.js and Express.js, linked to a MongoDB database via Mongoose. We employed JWT authentication to safeguard user access and adhered to best practices for data validation and error management. Postman was utilized to verify API endpoints during the development process. This iterative method enabled us to swiftly provide functional features, respond to feedback, and guarantee the system fulfilled user requirements.

Team structure and duties

Numbers	Members	Major roles and responsibilities
1	Eneb Hussen	Requirement analysis Project documentation Backend API support
2	Leul Tesfu	Backend development (Node.js/Express, API design) Database integration (MongoDB) Security & testing
3	Muhaba Ahmedteyib	System testing and quality assurance

		Database design Project documentation
4	Sami Fahmi	Frontend development (React components & UI design) User interface prototyping Integration with backend

Table 1 Team structure and duties

1.4 Tools

To make sure the Patient Record Management System (PRMS) for Dr. Seidnur Dental Clinic is reliable, scalable, and easy to use, a variety of contemporary software tools and technologies were used in its creation.

Numbers	Tools	Category
1	React.js	Front-end framework
2	Node.js and Express.js	Back-end framework
3	MongoDB	Database
4	Git,GitHub	Version Control
5	Visual Studio Code	Code Editor/Ide
6	Npm	Package Management
7	JWT	Authentication
8	Postman	API Testing Tool
9	Google Docs / MS Word	Documentation
10	Local Server / Heroku / Render	Deployment

Table 2 Tools used

1.5 Scope

The Patient Record Management System (PRMS) is designed to digitalize and streamline the management of patient records and related clinic operations at Dr. Seidnur Dental Clinic. The system focuses on key functionalities including patient registration, appointment scheduling, medical history tracking, and treatment documentation. Users with different roles such as administrators, receptionists, and dentists can access tailored interfaces and features suitable for their responsibilities.

The receptionist can register new patients, schedule and reschedule appointments, and update patient profiles. Dentists can access patient histories, record diagnoses and treatments, and attach relevant documents. Administrators can manage users, roles, audit logs, and system settings.

Additionally, the system includes a billing and payment tracking module that allows users to record treatment charges, track payment status (such as paid or pending), view payment history, and generate billing records. This ensures better financial tracking and transparency for both the clinic staff and the patients.

The system is built to be accessible through web browsers, ensuring ease of use, real-time access to data, and enhanced efficiency in clinic operations. However, it does not include modules such as pharmacy inventory management or diagnostic imaging integration, as these are considered outside the current scope.

1.6 Significance of the Study

The Patient Record Management System (PRMS) proposed for Dr. Seidnur Dental Clinic provides an efficient and user-friendly digital solution to handle patient data. By automating the process of patient registration, appointment scheduling, and treatment record management, the system improves the speed and accuracy of administrative operations within the clinic.

It reduces the burden of paper-based systems, minimizes errors caused by manual data entry, and ensures easy access to patient history and reports. Through secure access control, the system also ensures data privacy and allows only authorized personnel to access sensitive information.

Moreover, the system facilitates better communication between staff and patients by keeping appointment schedules clear and organized. It also supports data-driven decision-making through its report generation feature.

The billing integration enhances transparency and accountability, reduces errors in manual invoice handling, and helps clinic staff quickly verify payment status before or after treatments. This system not only boosts operational efficiency but also significantly improves the quality of service offered to patients.

1.7 Feasibility of the Study

To determine the feasibility and practicality of creating and deploying the Patient Record Management System (PRMS) for DR. SEIDNUR DENTAL CLINIC, a feasibility study was carried out. To make sure the project would succeed, the study looked at its technical, financial, operational, and schedule feasibility.

Technical Feasibility

The initiative employs popular and well-supported technologies like React.js for the frontend, Node.js and Express.js for the backend, along with MongoDB for the database. All necessary development tools (Visual Studio Code, Git, GitHub, npm) are available for free and work with standard hardware. The team has the required technical expertise in full-stack web development, and the clinic's current computer setup is adequate for operating the system. Consequently, the project is realistically achievable.

Economic Feasibility

The PRMS was created with open-source technologies, removing licensing expenses. The primary costs involve internet access, minor hardware improvements (if necessary), and possible hosting charges for deployment (which can be reduced by utilizing free or inexpensive platforms such as Heroku or Render). The system aims to cut administrative expenses, lessen paperwork, and enhance resource use, leading to long-term financial gains for the clinic. Consequently, the project is financially viable.

Feasibility of Operations

The system’s intuitive interface guarantees that clinic personnel can swiftly adjust to digital processes with little training required. The PRMS tackles the clinic’s key issues, like missing records and scheduling conflicts, while enhancing the general efficiency of everyday

operations. Participants (dentist, receptionist, administrator) have shown enthusiasm for the new system and are ready to engage in training and implementation. Consequently, the project is practically viable.

Feasibility of Scheduling

The project was organized and carried out according to the academic schedule, featuring specific milestones for gathering requirements, designing the system, implementing, testing, and deploying. The incremental development method facilitated prompt release of essential features and consistent feedback from users. All key functionalities were finished within the designated timeframe, rendering the project timeline achievable.

1.8 Risk assessment

The implementation of the Patient Record Management System (PRMS) at Dr Seidnur Dental Clinic brings various technical and operational risks that need to be effectively controlled. A major risk involves the security of data and the privacy of patients. Given that the system holds confidential personal and medical data, there exists a potential for unauthorized entry or data breaches, particularly if user verification or access restrictions are weakened. To address this, robust encryption, reliable authentication methods (like JWT), and frequent security assessments are crucial. Staff must also be educated on optimal methods for password management and data handling to reduce the chances of unintentional data exposure.

A further major risk pertains to system dependability and data accuracy. Any technical issue, including server outages, database damage, or software errors, might interrupt clinic functions, cause the loss of vital patient data, or create erroneous records. These risks can be mitigated by performing regular data backups, utilizing strong error-handling and validation processes, and carrying out comprehensive testing prior to deployment. Regular upkeep and swift problem-solving of technical difficulties are essential for maintaining consistent system availability and reliability.

Ultimately, there are dangers linked to user acceptance and workflow incorporation. Clinic personnel might resist moving from manual methods to a digital system, or they might commit mistakes because of their lack of familiarity with the new platform. This may impact the effectiveness and precision of clinic procedures. To mitigate this risk, thorough user training,

clear documentation, and prompt technical support ought to be offered. Collecting user feedback and continually enhancing the system based on actual usage will also contribute to ensuring seamless acceptance and lasting success of the PRMS at Dr. Seidnur Dental Clinic.

1.9 Work breakdown

Chapter One provides an introduction to the project by presenting the background, outlining the problem statement, and explaining the motivation for developing the Patient Record Management System for Dr. Seidnur Dental Clinic. It further defines the general and specific objectives, highlights the significance of the study, and specifies the scope and limitations of the project. The chapter concludes with a brief overview of the structure of the remaining chapters.

Chapter Two offers a comprehensive review of existing literature and related work concerning patient record management systems. It contrasts traditional manual methods with modern digital approaches, evaluates the advantages and limitations of existing systems, and identifies gaps in current solutions that the proposed system aims to fill.

Chapter Three presents the system analysis and design phase of the project. It discusses the methods used to gather requirements and outlines both functional and non-functional system requirements. The chapter also includes the use case diagram, system architecture, database schema, and the design of the user interfaces, providing a clear blueprint for system development.

Chapter Four describes the implementation and testing of the system. It details the development methodology and technology stack employed, explains the module-by-module implementation process, and outlines the testing procedures used to validate system functionality and reliability. The chapter concludes with a presentation and discussion of the testing results.

Chapter Five summarizes the project's outcomes, highlighting key findings and assessing the performance of the developed system. It reflects on the benefits and limitations encountered during the project and offers recommendations for future enhancements or related areas of research.

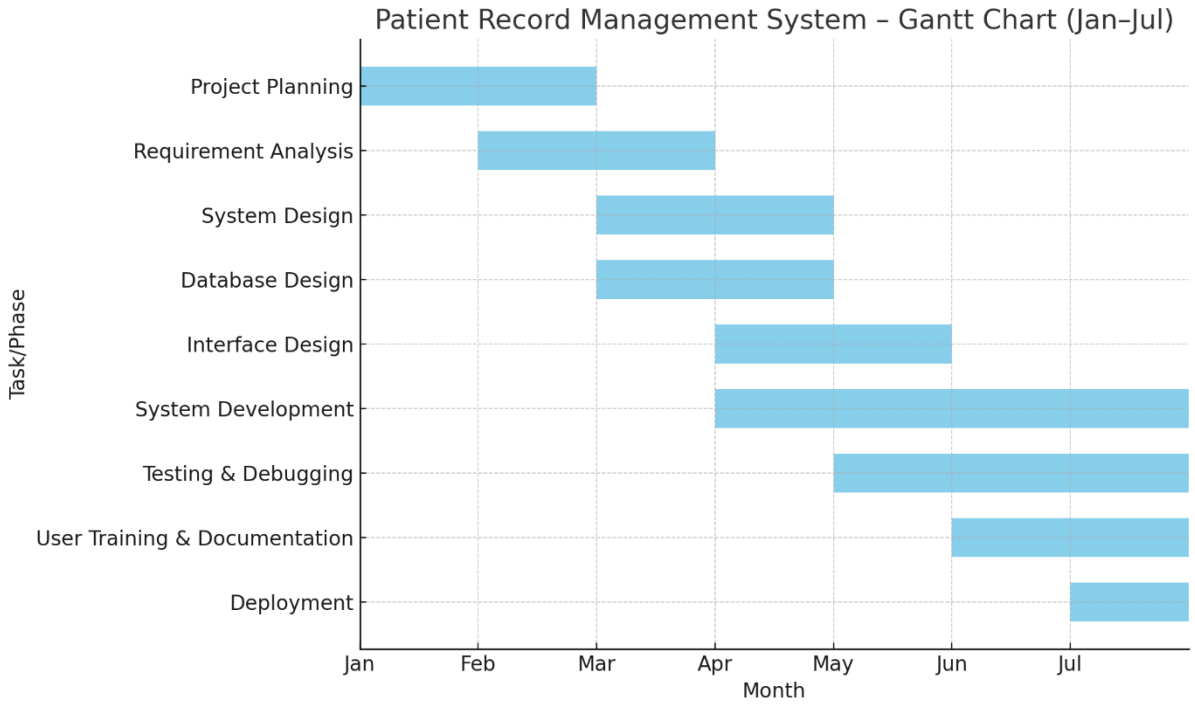


Figure 1 Grant chart diagram for PRMS project phases and timeline

Chapter Two Requirement Analysis and Specification

2.1Current system

The current patient record management system at Dr Seidnur Dental Clinic is mostly analog and reliant on paper. Patient data, encompassing personal information, health histories, and treatment documentation, is documented and kept in physical files. These documents are stored in cabinets and accessed by personnel when required. Appointment scheduling is often handled manually, frequently through notebooks or simple logs, necessitating staff to verify availability and update records manually.

This hands-on method presents various operational difficulties. Accessing patient records can take a considerable amount of time, particularly when handling numerous files or when documents are lost or incorrectly filed. Modifying patient records or treatment histories involves staff finding the appropriate file, making manual edits, and verifying that the information is clear and correct. This procedure is susceptible to mistakes, including incomplete submissions, unclear writing, and unintentional loss or destruction of documents.

Manual work is also involved in administrative tasks, like creating reports or monitoring clinic activities. Employees are required to examine various files and logs to gather information, making the process laborious and heightening the likelihood of errors. Safety is another issue, as physical documents are susceptible to unauthorized access, theft, or loss from fire or other calamities. There is a restricted capacity to track who accessed or changed records, complicating the enforcement of accountability.

In general, the existing system restricts the clinic’s effectiveness, data precision, and capacity to deliver prompt and high-quality patient care. With the clinic's expansion and rising patient numbers, these challenges become increasingly evident, underscoring the necessity for a digital Patient Record Management System to optimize operations, improve data security, and facilitate enhanced clinical and administrative decision-making.

2.2. Business rules

The following business rules guide the development and functionality of the Patient Record Management System (PRMS) for Dr. Seidnur Dental Clinic. These rules are based on existing clinic policies, operational workflows, and required compliance standards.

1. Patient Registration Rules

A patient must be registered before any appointment or treatment is recorded. All patient details, including full name, gender, date of birth, contact information, and address, are required for registration. Duplicate email addresses or phone numbers are not allowed in the system.

2. Appointment Scheduling Rules

Appointments can only be created for registered patients. Only available time slots can be assigned for a new appointment. Appointments must include assigned staff/doctor, reason for visit, and scheduled date.

3. Treatment History Management Rules

Treatment records must be linked to both the patient and the doctor who provided the service. Each treatment must include diagnosis, procedures, medication (if applicable), and date.

4. User Access Rules

Role-based access control is enforced. Admins can manage users, view reports, and oversee billing. Staff can register patients and manage appointments. Doctors can view patient records and enter treatment details. General users (patients) can only view their own profiles, appointment schedules, and payment status. Unauthorized users are restricted from accessing or modifying sensitive data.

5. Billing & Payment Rules

A billing record must be created for each treatment or service rendered. Billing entries must include the amount charged, issued by (staff or admin), status (paid or pending), and a timestamp. Only authorized staff or admins are allowed to mark a bill as paid. Payment status cannot be changed after being marked as “paid” unless verified by admin. Users (patients) can view but not modify their own billing history.

6. Data Integrity Rules

Required fields must be validated before submission. Duplicate records are not allowed for patients, appointments, or billing. Deleting a patient will also remove their associated appointments, history, and billing records to maintain database consistency.

7. Audit and Logging Rules

All actions related to patient registration, treatment updates, and billing changes must be logged. Logs should include user identity, date, and type of action for accountability and traceability.

2.3 Proposed system

To overcome the inefficiencies of manual record-keeping at Dr. Seidnur Dental Clinic, a web-based Patient Record Management System (PRMS) has been proposed. This system aims to centralize all patient data, streamline appointment scheduling, and improve the overall management of clinic operations through a secure, role-based digital platform.

The PRMS will store detailed patient records, including personal information, treatment history, and appointment logs, in a centralized database. Access to the system is role-specific, meaning users can only view or interact with the features assigned to their role. The main user roles and their capabilities are as follows:

- **Administrator**: Full access to all features, including user management, record deletion, and data exports.
- **Doctor**: Can view assigned patient records, add or update treatment details, and access their appointment schedule.
- **Receptionist**: Manages appointment scheduling, patient registrations, and basic contact updates.
- **Patient**: Can log in to view their own profile and appointments, and update personal information.

The appointment scheduling follows basic CRUD operations with role-based access control where only Admin and Staff can create appointments, while the system automatically creates patient records if they don't exist during appointment booking.

Security is a key component of the system. All users must log in with credentials, and sensitive actions like deleting records are restricted to administrators. Every user action is tracked through an audit log, and data can be exported in CSV format for backup and reporting purposes.

Though the system is currently deployed locally within the clinic, it is designed for future scalability. Planned improvements include cloud deployment, remote access, and mobile application support to enhance accessibility.

The interface is user-friendly and designed for ease of use, even for users with minimal technical skills. Training and user support materials will be provided to ensure a smooth transition from the manual system to the digital platform.

2.3.1 Overview

The suggested Patient Record Management System (PRMS) for Dr. Seidnur Dental Clinic provides a significant technological advancement over the manual, paper-based processes currently in use. The goal of this web-based system is to centralize and automate the management of patient records, appointment scheduling, and treatment documentation in order to address the limitations, risks, and inefficiencies of the current system.

Serving as a secure, centralized digital repository for all patient data will be the main duty of the PRMS. Personal information, dental and medical records, and a detailed record of all clinic visits and treatments will all be included in each patient's unique digital profile. The system will enable authorized users, like dentists and receptionists, to view, edit, and manage these records in real time, ensuring that the data is always current, accurate, and readily available when needed.

The appointment management module will provide a calendar-based, easy-to-use interface for making, rescheduling, and canceling patient appointments. This innovation will improve clinic workflow overall by eliminating scheduling conflicts and reducing the likelihood of double booking. Automated reminders can be sent to patients, increasing patient involvement and reducing missed visits.

The streamlined treatment documentation process will allow dentists to directly enter diagnoses, treatments, prescriptions, and follow-up instructions into the system. By ensuring

that every aspect of patient treatment is thoroughly documented and easily accessible for future use, this supports continuity of care and informed clinical decision-making.

Security and privacy are the core elements of the proposed system. The PRMS's robust authentication and role-based access controls will ensure that only authorized personnel can view or change sensitive patient data. All system operations will be documented for auditing purposes, and data will be protected through encryption and regular backups.

Effective deployment is made possible by the system's accessible and user-friendly design, which also lowers the learning curve for clinic staff. By automating and digitizing crucial processes, the PRMS will improve the accuracy and accessibility of patient data. Additionally, it will free up important staff time, allowing the clinic to focus more on offering excellent dental care.

2.3.2 Functional Requirements

The Patient Record Management System (PRMS) is expected to meet the following functional requirements:

- The system shall allow users to register new patients and store their basic information.
- The system shall allow searching, viewing, and updating patient records.
- The system shall allow appointment scheduling, rescheduling, and cancellation.
- The system shall allow dentists to record treatment details and view patient treatment history.
- The system shall provide role-based access for administrators, receptionists, and dentists.
- The system shall allow secure login and logout functionality.
- The system shall provide an interface for generating reports.
- The system shall track user actions for audit purposes.
- The system shall allow adding, viewing, and updating payment status for patient treatments.
- The system shall display service details, total amount charged, payment status (paid or pending), and timestamps for each transaction.
- The system shall allow filtering of payment records by status (e.g., paid or pending) to assist in financial review and tracking.

2.3.3 Non-functional Requirements

- The Patient Record Management System must fulfill several non-functional requirements to ensure that it performs effectively, securely, and reliably in a real-world dental clinic environment. These include:
- **Performance:** The system should respond to user interactions within two seconds under normal operating conditions. Billing records and patient data should load efficiently without delays, even as the number of records increases.
- **Security:** Sensitive data such as patient information and payment records must be protected using authentication and authorization mechanisms. Only authorized users (e.g., staff, admin) can access or modify billing data. Secure protocols must be used to prevent unauthorized access and protect data during transmission.
- **Usability:** The interface must be simple and intuitive, allowing clinic staff to navigate modules, including billing, without needing advanced technical skills. Users should be able to view, enter, and update payment information easily.
- **Reliability:** The system must operate consistently without crashing or losing data. All transactions, including billing operations, must be processed reliably and stored securely.
- **Availability:** The system should be accessible during clinic working hours with minimal downtime. Payment features must be available at all times during operation to avoid service delays.
- **Scalability:** The system should be capable of handling increasing numbers of patients, appointments, and payment records without compromising performance.
- **Maintainability:** The system should be easy to update and modify. For example, adding new payment categories or updating billing statuses should not require major code changes.
- **Auditability:** All billing operations should be logged with timestamps and issuer information to allow for traceability and accountability.

2.3.4 System model

2.3.4.1 Usecase

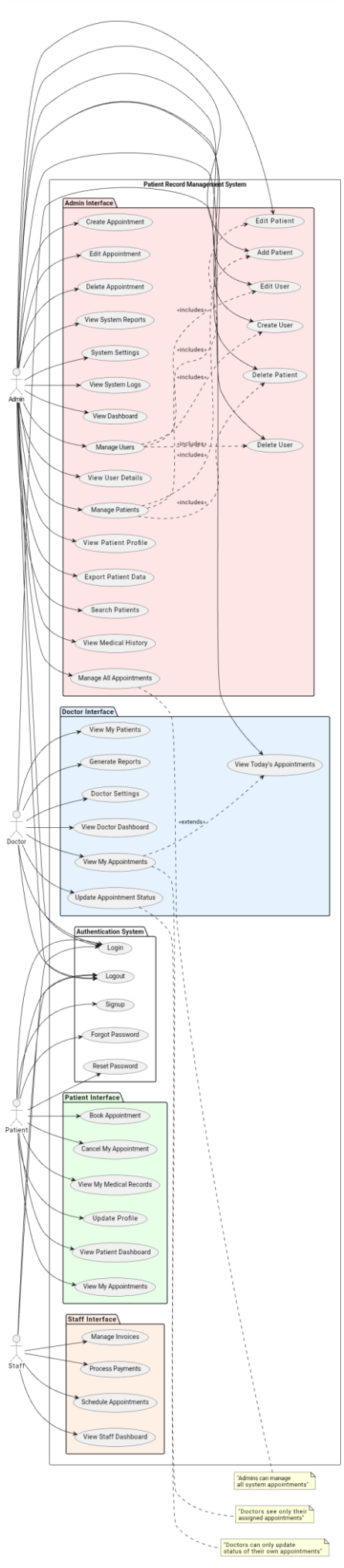


Figure 2 Use case model

Use case Description

The Patient Record Management System is a role-based web application designed to manage patient information, appointments, and billing efficiently. It supports user authentication through email and password, redirecting users to different dashboards based on their roles—admin, staff, doctor, or patient. New users can register, confirm their password, and are automatically logged in upon successful signup. Administrators have full control over the system, including managing users, patients, and appointments, with features like search, data export, and role/status assignment. Doctors can only view and update their own assigned appointments, with restricted access that limits visibility to just appointment status updates. Staff members handle patient registration, schedule appointments, and process billing, while patients can only view their profiles, appointments, and payment history. The system enforces strict access control through protected routes and role-based permissions, ensuring that doctors have the most limited access and administrators maintain complete oversight—creating a secure and structured environment for clinic operations.

2.3.4.2 Sequence diagram

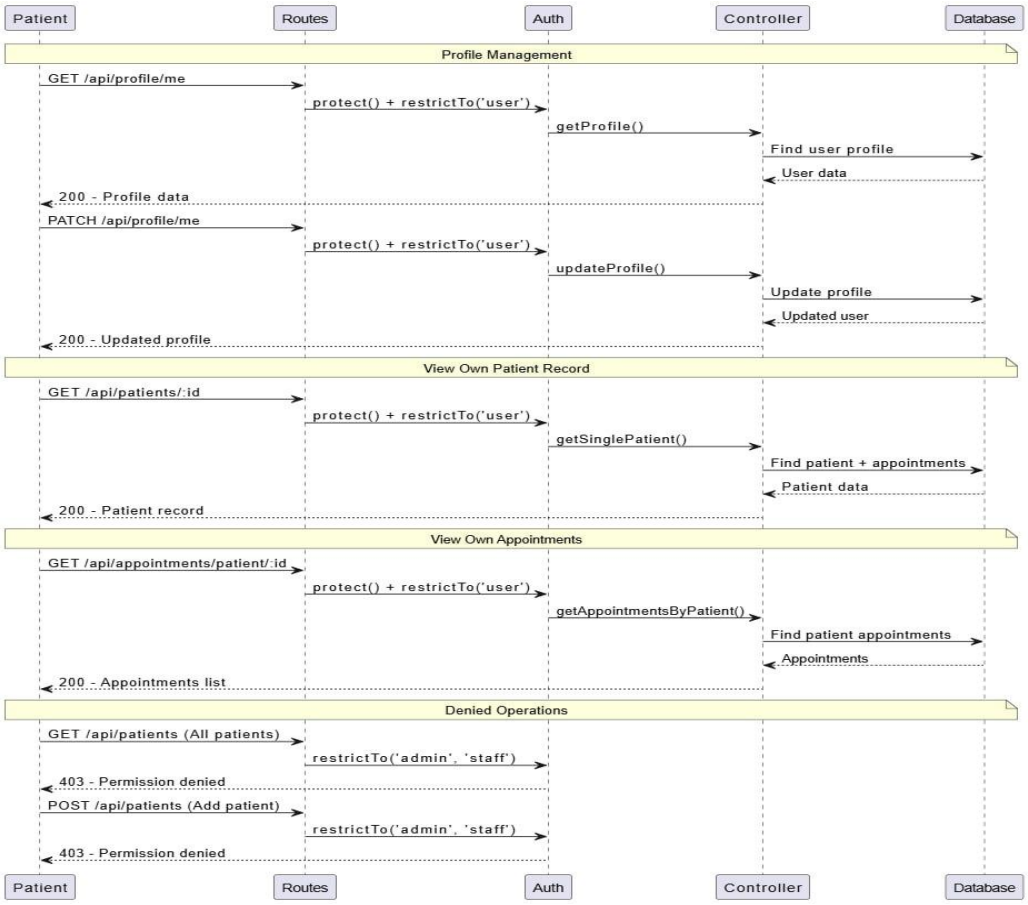


Figure 3 Patient Sequence Diagram

Patient Sequence Diagram Description

The patient/user role in the PRMS system represents the most restricted access level, designed specifically for patients to manage their own healthcare information while maintaining strict privacy boundaries. Users with this role can perform self-service operations including viewing and updating their personal profile information, accessing their own patient records with populated appointment data, and viewing their appointment history. All operations are protected by JWT authentication middleware and role-based access control that validates the

user's permissions before allowing access to resources. However, patients are explicitly denied access to administrative functions such as viewing all patients, creating new patient records, managing other users' appointments, or accessing system-wide reporting features, ensuring that each patient can only access their own medical information in compliance with healthcare privacy regulations.

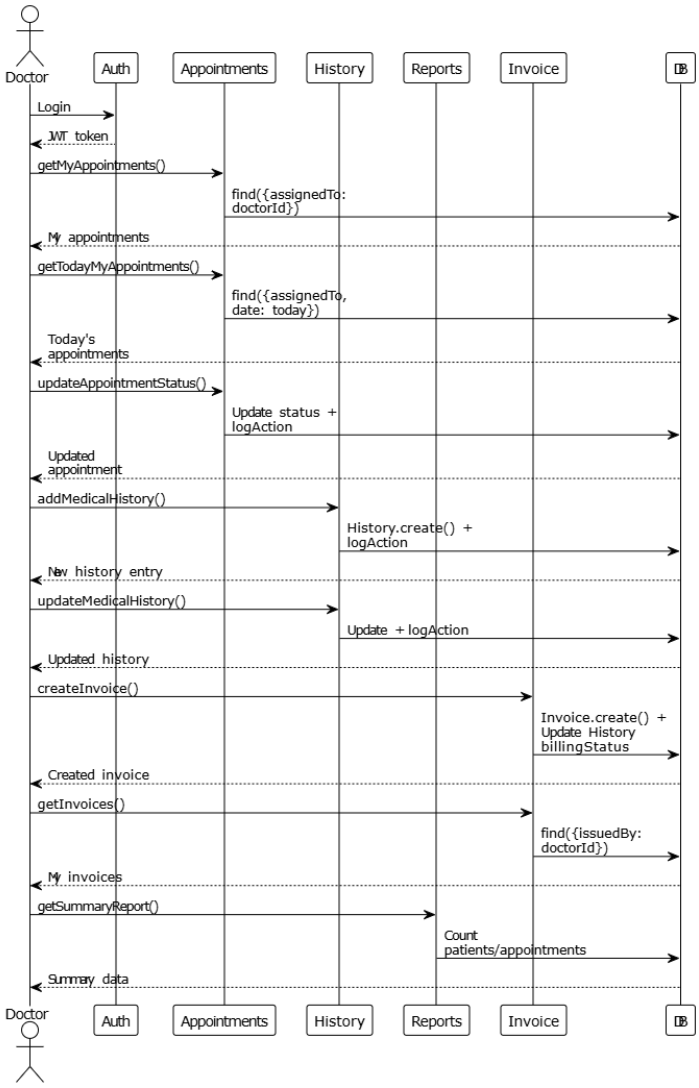


Figure 4 Doctor Sequence diagram

Doctor Sequence Diagram Description

The doctor workflow sequence diagram illustrates the core operations available to doctors in the PRMS system through eight main interactions. The workflow begins with authentication

where the doctor logs in and receives a JWT token. For appointment management, doctors can view all their assigned appointments, view today's appointments with a maximum of 10 results, and update appointment status with proper authorization checks. Medical history operations include adding new patient medical records and updating existing medical history entries. Invoice management capabilities allow doctors to create patient invoices and view their issued invoices. Finally, doctors can access system summary statistics through reporting functionality.

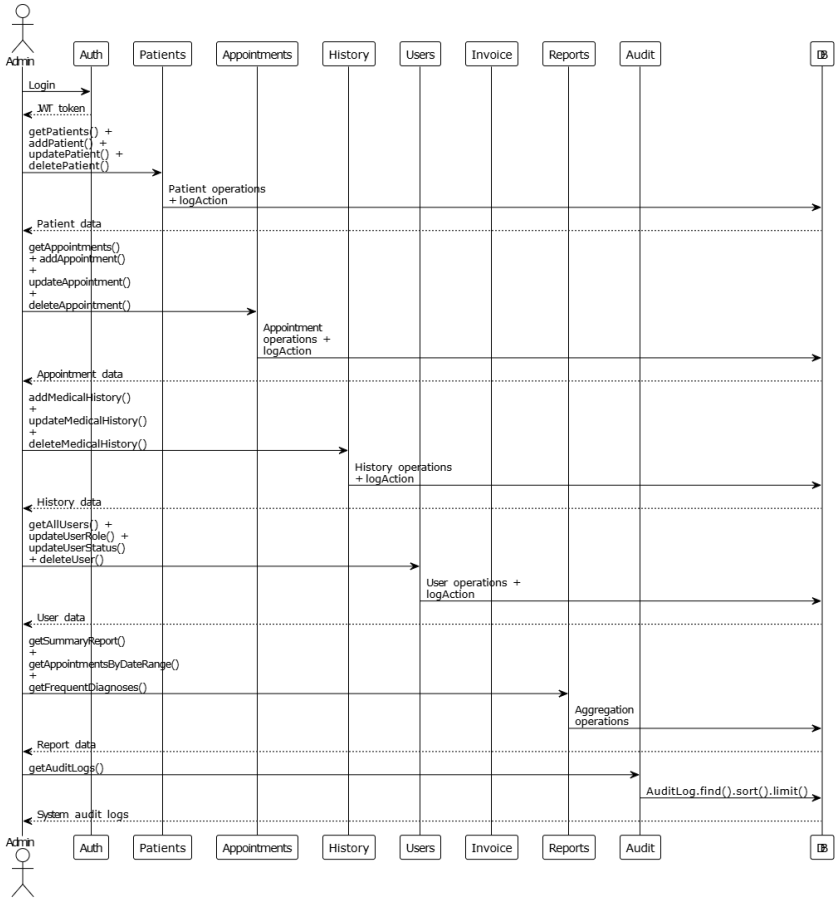


Figure 5 Admin Sequence Diagram

Admin Sequence Diagram Description

The admin workflow sequence diagram illustrates the comprehensive administrative operations available to administrators in the PRMS system through eight main interactions. The workflow begins with authentication where the admin logs in and receives a JWT token. For patient management, administrators have full CRUD capabilities including viewing, adding, updating, and deleting patients with automatic user account creation. Appointment management provides complete control over scheduling operations including viewing,

creating, modifying, and removing appointments with comprehensive audit logging. Medical history operations allow administrators to add, update, and delete patient medical records, providing the highest level of access to clinical data. User management capabilities are exclusive to administrators, including viewing all users, updating user roles, managing user status, and deleting user accounts. Reporting functionality provides access to system analytics including summary reports, date-range appointment reports, and frequent diagnosis analysis. Finally, audit log access is restricted exclusively to administrators, allowing them to monitor all system activities for compliance and security purposes.

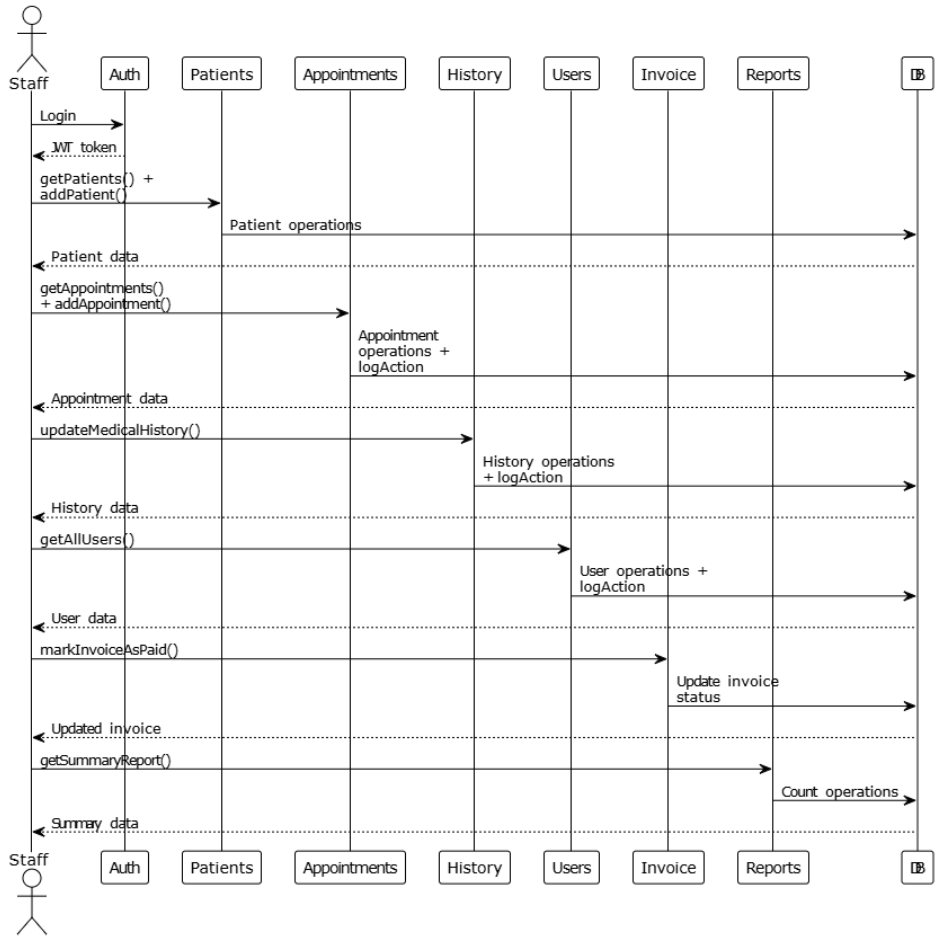


Figure 6 Staff Sequence Diagram

.

Staff Sequence Diagram Description

The staff workflow sequence diagram illustrates the comprehensive operations available to staff members in the PRMS system through seven main interactions. The workflow begins with

authentication where staff logs in and receives a JWT token. For patient management, staff can view all patients and add new patients with automatic user account creation. Appointment management includes viewing all appointments and creating new appointments with audit logging. Medical history operations focus on updating existing medical records, though staff actually have broader permissions including add and delete capabilities as shown in the role matrix. User management allows staff to view all users in the system. Invoice management enables staff to mark invoices as paid, a capability unique to staff members. Finally, staff can access system summary reports for operational insights. All operations include automatic audit logging and role-based authorization, representing a streamlined workflow that covers the essential administrative and clinical operations within the PRMS system.

2.3.4.3 State chart diagram

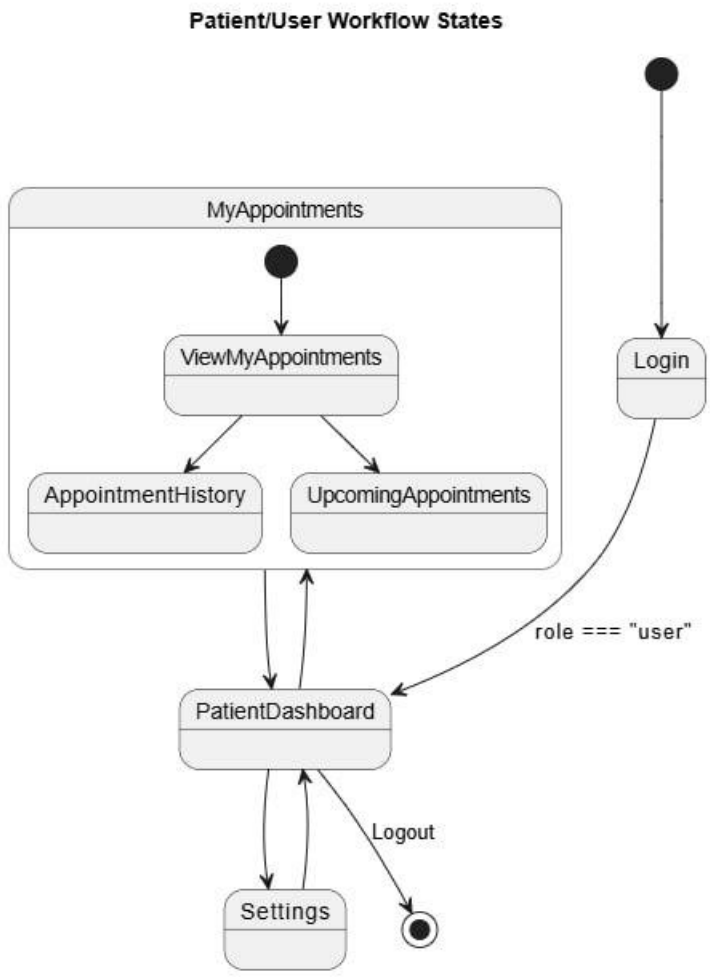


Figure 7 Patient state chart

The Patient/User Role State Chart represents the simplest self-service workflow designed for end-users, where patients can only view their personal appointment history and upcoming appointments, manage basic profile settings and account preferences through the patient routes ( ), with minimal functionality that excludes any administrative capabilities and focuses entirely on personal data access, ensuring patients have appropriate visibility into their own healthcare information while maintaining system security.

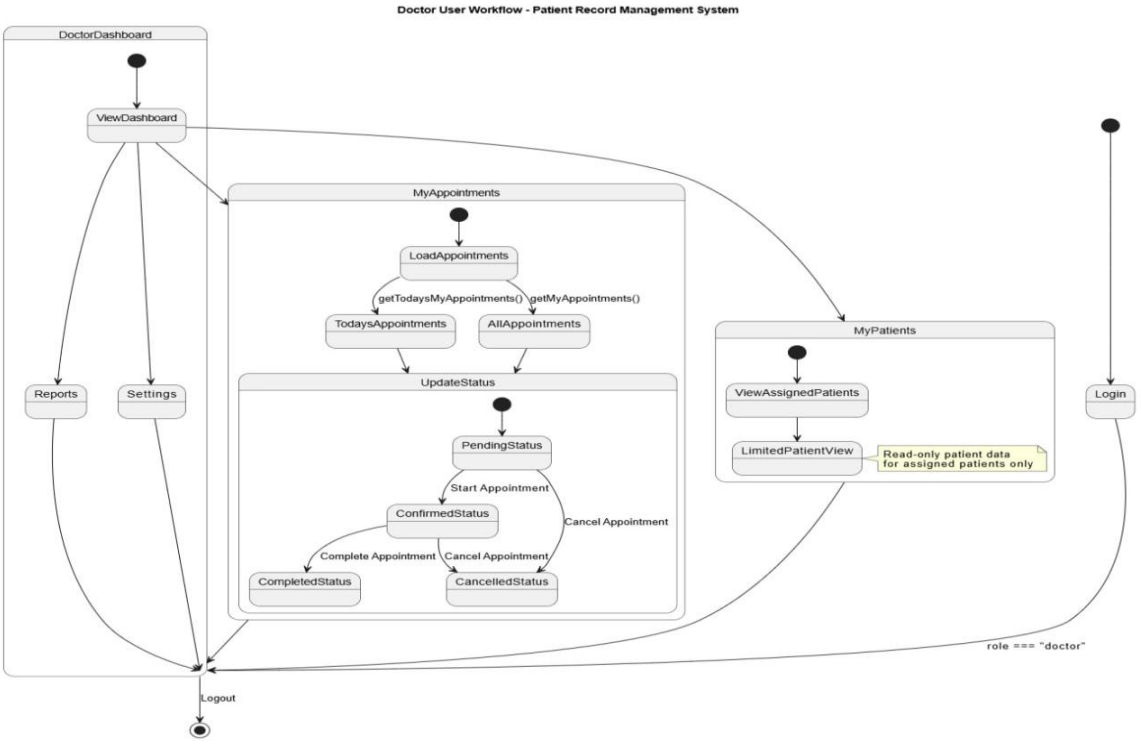


Figure 8 Doctor State Chart

The Doctor Role State Chart represents the most specialized clinical workflow, focusing on appointment management where doctors can only view their own appointments through dedicated API endpoints, manage appointment status through a three-state workflow (pending → confirmed → completed) with cancel options, have read-only access to patient information for assigned patients only, and are restricted from creating appointments, managing users, or accessing detailed appointment views, ensuring doctors focus purely on clinical care rather than administrative functions.

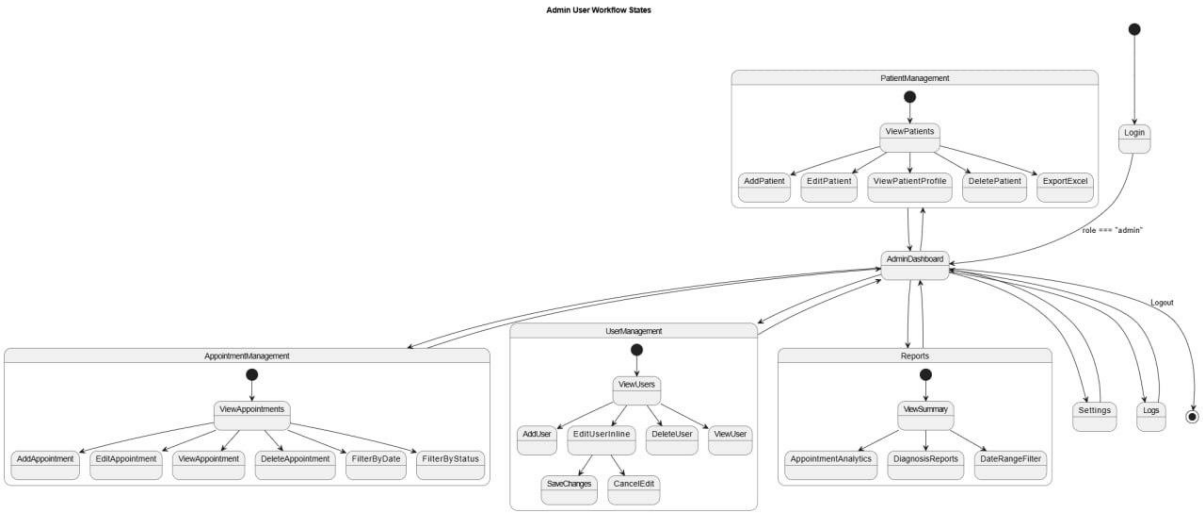


Figure 9 Admin State Chart

Admin State Chart Description

Admin Role State Chart

The Admin Role State Chart represents the most comprehensive workflow in the PRMS system, showing how administrators have full system access including complete patient management (CRUD operations, export functionality), appointment management with full lifecycle control, unique user management capabilities with inline editing of roles and status ( ), comprehensive reports and analytics access, and system administration features like settings and logs, all accessible through the protected admin routes ( ) that ensure only users with admin role can access these privileged functions.

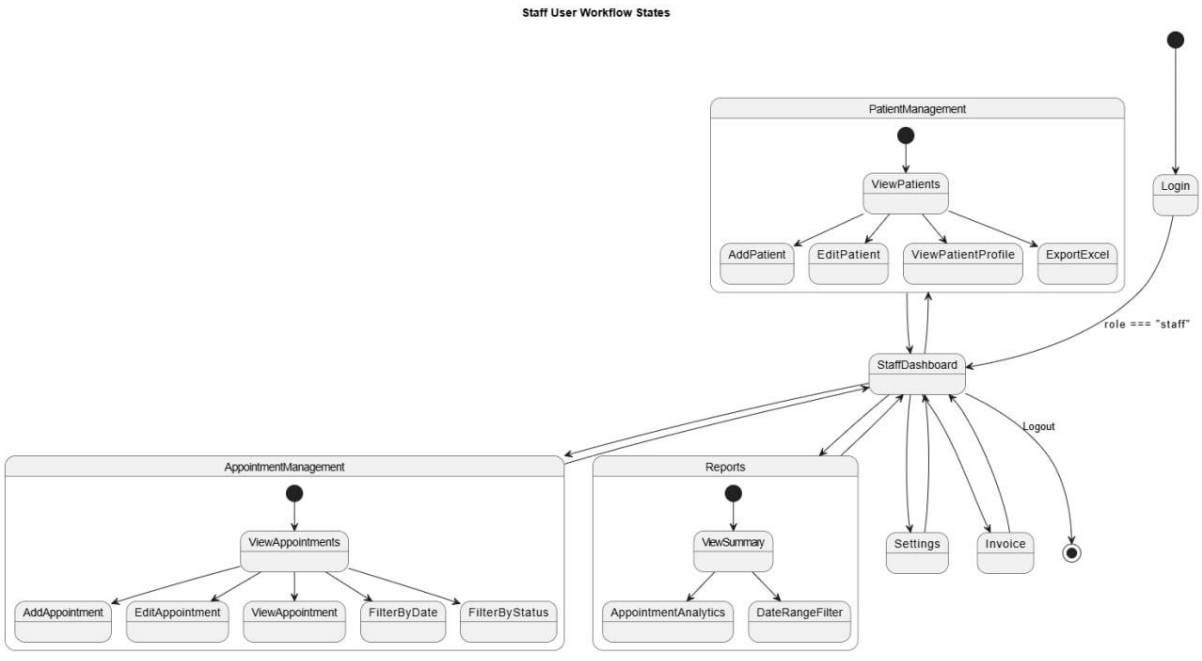


Figure 10 Staff State Chart

The Staff Role State Chart demonstrates a workflow similar to admin but with restricted permissions, where staff members can perform the same patient operations as admins (add, edit, view, export) and have full appointment scheduling and management capabilities through the staff routes ( ), can access operational reports for day-to-day healthcare management, but critically cannot create or modify user accounts, making user management an admin-only privilege that maintains proper access control hierarchy in the healthcare system.

2.3.4.4 Activity Diagram

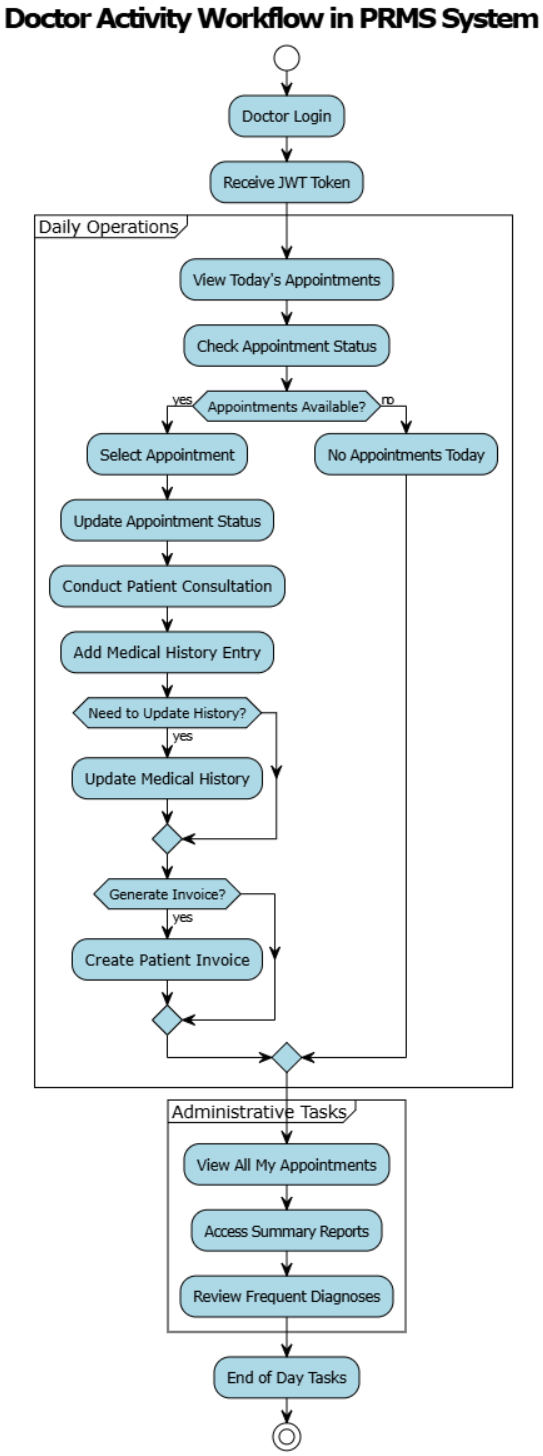


Figure 11 Doctor Activity Diagram

Doctor Activity Diagram Description

The doctor activity diagram illustrates the daily workflow operations available to doctors in the PRMS (Patient Records Management System) through a structured decision-based process. The workflow begins with doctor authentication where they log in and receive a JWT token for system access.

The core workflow is organized into two main partitions: Daily Operations and Administrative Tasks. In the Daily Operations section, doctors start by viewing their today's appointments and checking appointment status. If appointments are available, the workflow branches into patient consultation activities including selecting appointments, updating appointment status, conducting patient consultations, and adding medical history entries .

The workflow includes conditional decision points for updating medical history and generating patient invoices based on clinical needs. If no appointments are available, the system displays "No Appointments Today."

The Administrative Tasks partition covers viewing all assigned appointments, accessing summary reports, and reviewing frequent diagnoses analytics. The workflow concludes with end-of-day tasks before termination.

This activity diagram represents a comprehensive clinical workflow optimized for doctor role permissions within the PRMS system, emphasizing patient care activities while maintaining proper authorization controls and audit logging throughout all operations.

Figure 12 Patient Activity Diagram–

Patient Activity Diagram Description

The Patient Activity Diagram illustrates the complete patient journey through the Patient Record Management System (PRMS) as a continuous workflow organized into three swimlanes representing Patient, System, and Doctor interactions. The diagram begins with patient login and dashboard access, then branches into two main paths: appointment scheduling or viewing existing records. When a patient requests an appointment, the system creates it with "pending" status and notifies the assigned doctor, who can then accept (changing status to "confirmed"), reject (changing to "cancelled"), or later complete the appointment (changing to "completed"). The diagram incorporates the role-based access control where patients have limited permissions for viewing their own data while doctors can update appointment statuses through dedicated UI controls.The workflow concludes with patients being able to view their appointment history, medical records, and update personal information, reflecting the comprehensive patient management capabilities supported by the underlying API operations.

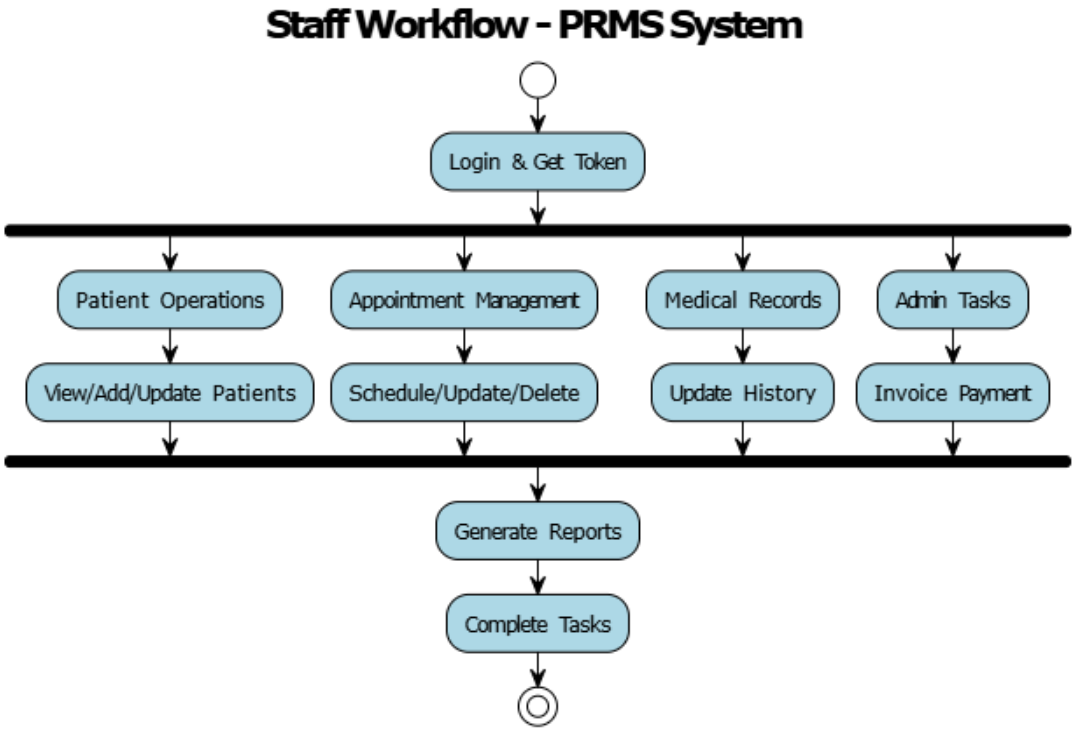


Figure 13 Staff Activity Diagram

Staff Activity Diagram Description

The staff activity diagram shows a parallel workflow optimized for A4 printing where staff authenticate and then perform four main operational branches simultaneously. Staff handles patient operations including viewing, adding, and updating patients with automatic user account creation. Appointment management covers scheduling, updating, and deleting appointments with full administrative control. Medical records operations allow staff to update history entries, though they actually have broader add and delete permissions. Administrative tasks include invoice payment processing, which is unique to staff members, and user status management.

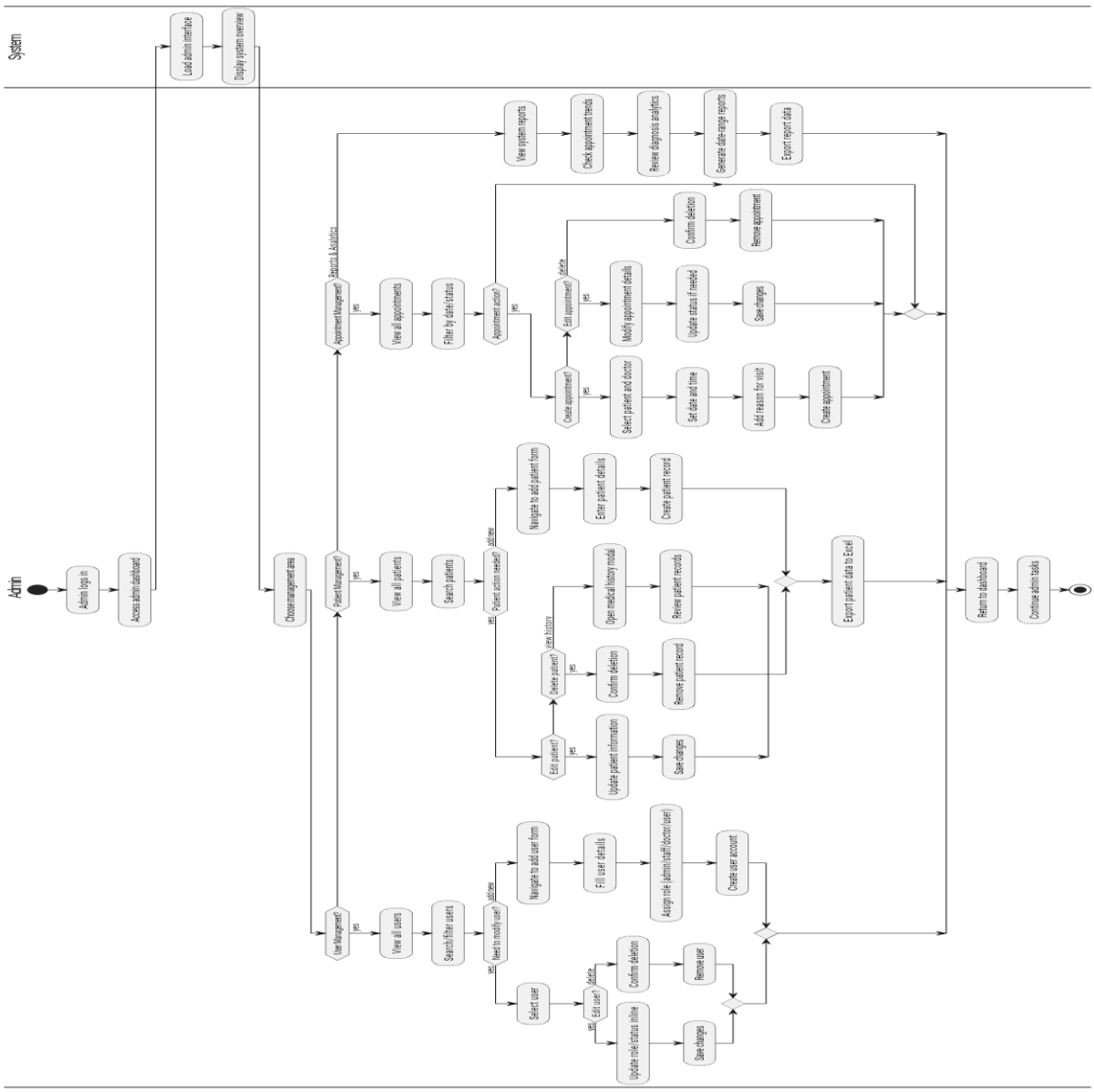


Figure 14 Admin Activity Diagram

Admin Activity Diagram Description

Admin Activity Diagram illustrates the comprehensive administrative workflow within the PRMS, showing how administrators navigate through the system's core management areas with full access privileges. The diagram begins with admin login and dashboard access, then branches into four primary management areas: user management where admins can search, filter, and perform inline editing of user roles and statuses, patient management with full CRUD operations including adding, editing, deleting patient records and exporting data to spreadsheet format, appointment management with complete scheduling control including creation, modification, and deletion of appointments, and reports & analytics for viewing system-wide data and trends. The workflow incorporates the system's role-based access control where admins have unrestricted permissions across all modules, unlike doctors who have limited access to specific functionalities. The diagram demonstrates the admin's ability to manage the entire healthcare ecosystem, from user account administration and patient record management to appointment scheduling and system reporting, with features like inline editing for user management, downloadable data export capabilities for patient information, and comprehensive appointment lifecycle control.

2.3.4.5 Class diagram

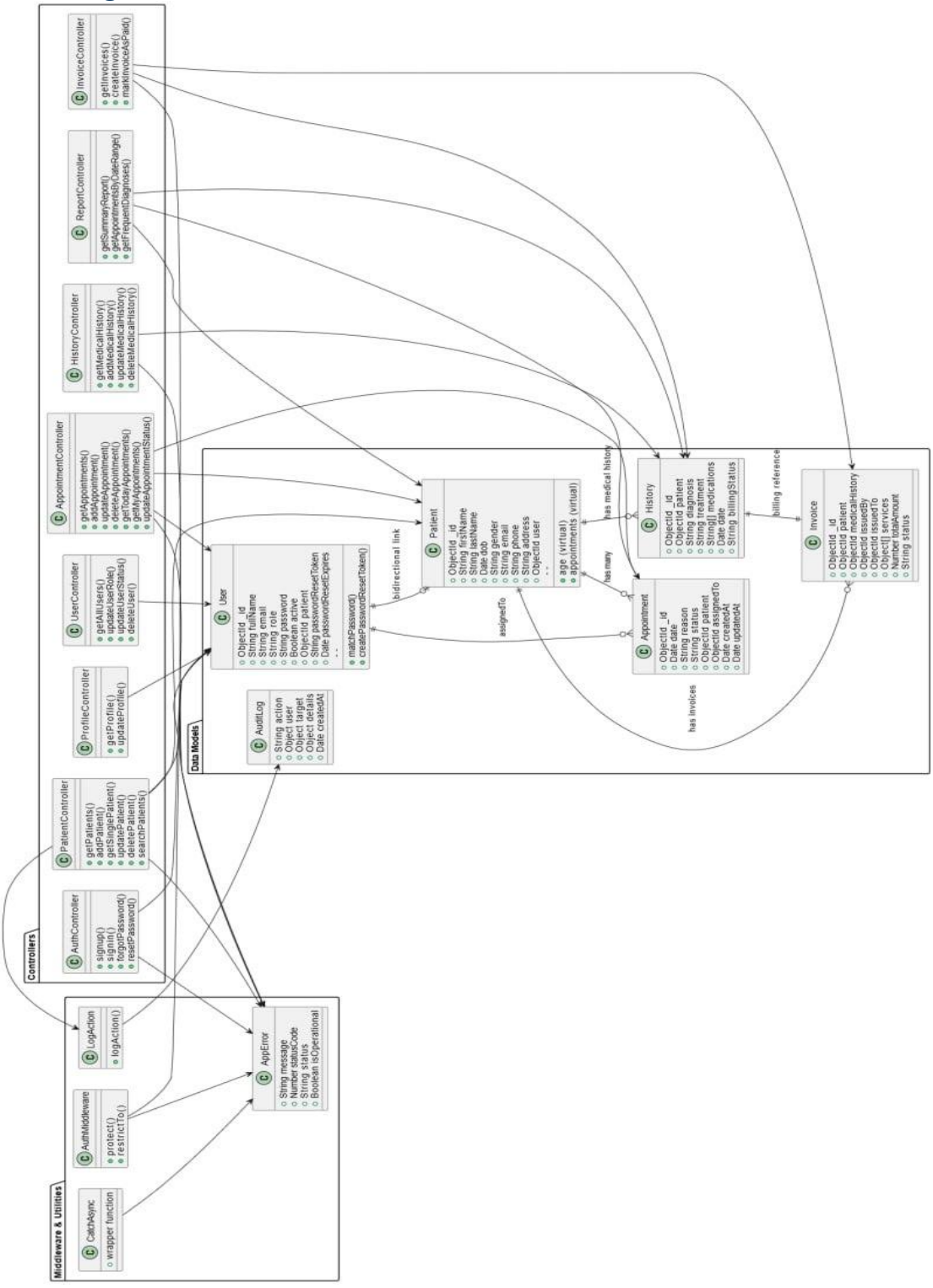


Figure 15 Class Diagram

Class Diagram Description

This class diagram represents the core architecture of the PRMS backend, showing the relationships between data models, business logic controllers, and supporting utilities. The system maintains strict separation of concerns with role-based access control enforced at the middleware level, comprehensive audit logging, and bidirectional relationships between users and patients for healthcare data management.

Chapter Three System Design

3.1 Introduction

System design is a vital stage in the software development life-cycle, acting as the link between requirement analysis and the actual execution of the Patient Record Management System (PRMS) for DR. Seidnur Dental Clinic This chapter transitions from grasping what the system needs to achieve, as outlined in the earlier chapter, to exploring how those requirements will be implemented through specific technical solutions.

The main objective of system design is to convert both functional and non-functional requirements into a detailed plan that directs developers, testers, and stakeholders during the development of the PRMS. This entails making important choices regarding the system's structure, data handling approaches, user interface design, and the relationships among different system elements. The design process guarantees that the end product will be reliable, adaptable, safe, and easy to use, fulfilling the operational requirements of the clinic as well as the staff's expectations.

This chapter starts with an overview of the selected system architecture, explaining the reasoning for choosing technologies like React for the front-end, Node.js/Express for the back-end, and MongoDB for data storage. It subsequently shows the database architecture, featuring entity-relationship diagrams and data models that outline how patient, appointment, and treatment data will be structured and retrieved. The user interface design segment includes mock-ups and details of important screens, highlighting usability and accessibility for various user roles, including admin, doctor, and receptionist.

Additionally, the chapter provides comprehensive specifications for system modules and their interactions, accompanied by diagrams including use case diagrams, sequence diagrams, class diagrams, and activity diagrams. These visual representations illustrate the movement of data and control inside the system, aiding in discovering possible bottlenecks or security issues early on in the development phase.

This chapter establishes the framework for methodical implementation, comprehensive testing, and seamless PRMS deployment by offering an organized and comprehensive design. The design choices outlined here are meant to guarantee that the system not only performs as intended but also grows and changes with the clinic's operations over time.

3.2 Design goal

The Patient Record Management System (PRMS) is designed to enhance clinic efficiency by automating the collection, storage, and access of patient data. The system is built to streamline workflows, allowing users to register patients, arrange appointments, and modify treatment records with ease. Through the automation of essential processes, the PRMS minimizes administrative burdens, removes duplicate paperwork, and speeds up access to precise patient information, thus facilitating prompt and informed decision-making in the clinic.

To improve security and accountability, the system applies rigorous role-based access controls and oversees all vital activities. It limits sensitive tasks to approved users, records all create, read, update, and delete (CRUD) activities, and offers administrators detailed audit logs. These measures safeguard patient confidentiality, guarantee adherence to healthcare standards, and promote a culture of accountability among personnel by rendering all actions traceable and subject to review.

The PRMS is designed to enhance user experience and flexibility. It provides a responsive, web-oriented interface that users can reach from any contemporary device, guaranteeing ease and adaptability. The design emphasizes user-friendly navigation, straightforward feedback, and strong validation to avoid mistakes and assist users efficiently. By enhancing scalability and maintainability, the system enables the clinic to handle future expansion, incorporate new features, and adapt to changing healthcare requirements without sacrificing performance or user experience.

3.3 Design trade-off

In developing the Patient Record Management System (PRMS) for DR. Seidnur dental clinic, various significant trade-offs were evaluated to balance usability, performance, security, scalability, and maintainability. These compromises represent deliberate choices made during the system design stage to optimally satisfy the clinic's requirements within technical, financial, and operational limitations.

Simplicity vs. Feature Richness

- **Trade-off:** The system emphasizes an easy and user-friendly interface to support users with different levels of technical expertise. This implies that certain advanced functionalities (like automated appointment notifications or analytics dashboards) were

postponed or simplified to prevent overwhelming users and to facilitate training and adoption.

- **Rationale:** Concentrating on essential features (patient registration, appointment scheduling, treatment documentation) guarantees system accessibility and minimizes the likelihood of user mistakes or push-back.

Security vs. Usability

- **Trade-off:** Robust authentication and role-based access controls are employed to safeguard sensitive patient information. Nonetheless, these security protocols can create additional hurdles for users (e.g., repeated logins, restricted access based on roles), potentially diminishing convenience somewhat.
- **Rationale:** Due to the clinic's duty to safeguard patient confidentiality, security takes precedence, even if it necessitates users adjusting to more stringent access measures.

Efficiency vs. Scalability

- **Trade-off:** The system is built to operate effectively for the clinic's current scale, emphasizing fast data access and interactive interfaces. Although MongoDB and RESTful APIs facilitate future growth, certain enhancements for extremely large datasets (like advanced indexing or distributed databases) have not been applied at this point.
- **Rationale:** This method guarantees quick performance currently, while permitting future enhancements as the clinic expands, without making the initial setup overly complex.

Personalization vs. Durability

- **Trade-off:** The system employs standardized forms and processes for managing patients and appointments instead of allowing highly customizable templates for every user or department.
- **Rationale:** Standardization streamlines upkeep, shortens development duration, and guarantees uniformity throughout the clinic, despite potentially restricting certain individual preferences.

Development Speed vs. Extensibility

- **Trade-off:** The system was built using commonly used technologies (React, Node.js/Express, MongoDB) and modular coding methodologies. Certain features were purposefully postponed for later phases to guarantee prompt delivery and comprehensive testing of essential modules.
- **Rationale:** Focusing on a reliable, operational release enables the clinic to take advantage of digital record management sooner, with a defined trajectory for subsequent improvements.

Cloud vs. Local Deployment

- **Trade-off:** The initial implementation prefers a local or on-site solution, which streamlines data management and lowers ongoing expenses. Yet, this implies that remote access and cloud backup capabilities are restricted in the initial version.
- **Rationale:** On-premises deployment coincides with the clinic's existing systems and privacy issues, with the possibility of transitioning to the cloud as requirements change.

3.4 Subsystem Decomposition

The Patient Record Management System (PRMS) is divided into several interrelated subsystems to ensure modularity, ease of maintenance, and scalability. The User Management Subsystem is responsible for handling user registration, authentication, role-based access control, and profile management, ensuring that only authorized individuals can access system features based on their assigned roles such as admin, dentist, or receptionist. The Patient Management Subsystem focuses on registering new patients, storing their personal information and contact details, and supporting data updates when needed. The Appointment Scheduling Subsystem allows for efficient booking, rescheduling, and cancellation of appointments while maintaining a calendar view for both doctors and receptionists. The Medical History & Treatment Subsystem enables dentists to record diagnoses, treatment procedures, medications, and notes, providing a complete view of a patient's clinical journey. The Reporting and Audit Subsystem generates summaries and reports related to appointments, treatment records, and user activities, while also maintaining audit trails for system operations to support accountability and decision-making., The system includes a Billing Module, which manages payment records, displays treatment-related charges, tracks payment status (such as paid or pending), and stores timestamps along with the identity of the user who processed each

transaction. This module enhances financial transparency and makes it easier for the clinic staff to verify and manage billing activities efficiently. Together, these subsystems ensure that PRMS delivers a comprehensive and reliable digital solution for managing the dental clinic's daily operations.

3.5 Design Phase model

The requirements acquired during analysis are converted into a comprehensive technical blueprint that directs implementation during the Design Phase. Its main goal is to develop models that specify the behaviour, structure, and data management of the system. The design process for Dr. Seidnur Dental Clinic's Patient Record Management System (PRMS) attempts to guarantee that the system is scalable, secure, and maintainable. The system's primary design models class models, persistent models, user interfaces, deployment architecture, and network design are described in this chapter.

3.5.1 Class Modeling

Class modeling is a crucial task in object-oriented analysis and design. It requires recognizing the primary entities (classes) within the system, along with their attributes, behaviors (methods), and the connections among them. In the Patient Record Management System (PRMS), class modeling converts the functional requirements illustrated in the use case diagram into an organized, object-oriented framework that directs both database creation and application development.

Main Classes and Their Responsibilities

User

- **Attributes:** userID, username, password, role (Admin, Doctor, Receptionist), contactInformation, status
- **Methods:** signIn(), signOut(), changeProfile(), recoverPassword()
- **Responsibility:** Depicts system users, each assigned a distinct role and access level.

Patient

- **Attributes:** patientID, fullName, dateOfBirth, gender, contactInfo, address, registrationDate, medicalHistory
- **Methods:** register(), updateInfo(), viewRecord(), deleteRecord()

- **Responsibility:** Maintains and organizes patient demographic details and medical records.

Appointment

- **Attributes:** appointmentID, patientID, doctorID, appointmentDate, appointmentTime, status, notes
- **Methods:** schedule(), update(), cancel(), view()
- **Responsibility:** Oversees the organization and monitoring of patient visits with physicians

TreatmentRecord

- **Attributes:** treatmentID, patientID, doctorID, date, diagnosis, treatmentDetails, prescription
- **Methods:** addTreatment(), updateTreatment(), viewHistory()
- **Responsibility:** Records the specifics of every patient's diagnosis and therapy.

Report

- **Attributes:** reportID, category, dateCreated, createdBy, material
- **Methods**: create(), display(), output()
- **Responsibility**: Utilized for creating summaries and statistical reports for management.

AuditLog

- **Attributes:** logID, userID, action, timestamp, details
- **Methods:** recordAction(), viewLog()
- **Responsibility:** Monitors system operations for accountability and safety.

Relationships Among Classes

- The User class engages with all other classes based on designated roles, ensuring access control and security.
- Patient objects are linked to various Appointment and TreatmentRecord entries, illustrating continuous clinical care.
- Every Appointment connects a particular patient to a physician.

- Doctors create TreatmentRecord entries for patients, documenting the specifics of the medical care given.
- Entries in the AuditLog are created for actions taken by users, aiding in traceability and compliance.
- Report objects gather and compile information from patients, appointments, and treatments for management use.

Significance of Class Modeling

Class modeling offers a distinct and systematic framework for the PRMS, guaranteeing that:

- All key data entities and their relationships are clearly outlined.
- The system is composed of modules, allowing for simpler upkeep and future improvements.
- Data integrity, security, and workflow automation are maintained across the system.
- The design accommodates both existing clinic functions and expected future needs.

3.5.2 Persistent Model

Additionally, to support the newly added financial features, the system includes a Billing collection. This table is designed to store payment-related information for treatments. The billing_id serves as the primary identifier for each billing record. The patient_id field is a foreign key linking each payment to its corresponding patient. The amount field records the total charge for the treatment or service. The status field captures whether the payment is marked as paid or pending. The issued_by field logs the user (e.g., receptionist or admin) responsible for entering the billing data. Finally, created_at and updated_at fields automatically track the time the billing record was created and last modified.

The enduring model specifies the method of storing, handling, and preserving core information over time in the Patient Record Management System (PRMS). This model guarantees that all essential data including patient records, appointments, treatments, user accounts, and system logs are securely stored and can be accessed or modified as necessary to facilitate the clinic's functions.

Core Persistent Entities

Considering the extensive functionalities illustrated in the use case diagram, the key entities that need persistent storage are as follows:

User: Holds all data concerning system users, such as their roles (admin, doctor, receptionist), login credentials, and profile information. This allows for safe sign-in, access based on roles, and management of users.

Patient: Includes demographic and contact information for each patient, along with registration details and medical history. This entity is crucial to the system, facilitating registration, modifications, and thorough record maintenance.

Appointment: Documents all scheduling details, connecting patients with physicians for particular dates and times. Appointment status (scheduled, completed, cancelled) is kept to monitor clinic operations.

TreatmentRecord: Keeps a comprehensive medical record for every patient, encompassing diagnoses, treatments administered, prescriptions, and physicians' notes.

Report: Stores produced reports for management and clinical evaluation, encompassing overviews of appointments, treatments, and patient data.

Auditlog: Records important activities carried out in the system, including the creation, updating, deleting of records, and user logins. This promotes responsibility and adherence to regulations.

Relationships and Data Integrity

• Every Patient may have several Appointments and TreatmentRecords.

• Appointments are linked to both Patients and Doctors (users who have a doctor role).

• TreatmentRecords are connected to both Patients and Doctors.

• AuditLogs document activities performed by any User on any of the lasting entities.

Distinct identifiers (like patientID, userID, appointmentID) are utilized to preserve referential integrity and facilitate effective data access.

Data Storage approach

The PRMS utilizes a strong database management system (like MongoDB or a relational database) to execute the persistent model. The structure of the database is created to:

• Avoid data redundancy and maintain uniformity.

• Facilitate effective searches for querying, modifying, and generating reports.

• Implement access restrictions and safeguard confidential data.

• Facilitate dependable backup and restoration procedures.

Significance of the Persistent Model

• **Reliability:** Guarantees that crucial clinic information is maintained and can be retrieved, even during system malfunctions.

• **Safety:** Safeguards confidential medical and personal data via regulated access and secure storage solutions.

• **Auditability:** Keeps a comprehensive record of user activities and system modifications to ensure transparency and adherence to regulations.

• **Scalability:** Accommodates the increase of patient, appointment, and treatment information as the clinic grows.

3.5.2.1 Mapping Class Diagram to Relation

Mapping the class diagram to a relational model is an essential part of system design, as it bridges the gap between object-oriented analysis and relational database implementation. This process involves converting each class and its relationships into corresponding tables, columns, and constraints within a relational database management system.

Core Entity Mapping

Our PRMS system maps four main classes to MongoDB collections:

User Class → users Collection

The User class maps to a users collection with fields:

- **_id** (ObjectId) - Primary key
- **fullName**, **email**, **role**, **password** - Core attributes
- **active**, **passwordResetToken**, **passwordResetExpires** - State management
- **createdAt**, **updatedAt** - Timestamps (auto-generated)

Patient Class → patients Collection

The Patient class maps to a patients collection with:

- **_id** (ObjectId) - Primary key
- **firstName**, **lastName**, **dob**, **gender** - Demographics
- **email**, **phone** - Contact info (both unique indexes)
- **address** - Optional field
- Virtual fields for **age** and appointments relationship

Appointment Class → appointments Collection

The Appointment class creates an appointments collection with:

- **_id** (ObjectId) - Primary key
- **date**, **reason**, **status** - Core appointment data
- **patient** (ObjectId) - Foreign key to patients collection
- **assignedTo** (ObjectId) - Foreign key to users collection
- **createdAt**, **updatedAt** - Auto-timestamps

AuditLog Class → auditlogs Collection

The AuditLog class maps to an auditlogs collection with:

- _id (ObjectId) - Primary key
- action - Action description
- user - Embedded user info (id, role)
- target - Dynamic reference using refPath
- details - Flexible object for additional data
- createdAt – Timestamp

Relationship Mapping

The class diagram relationships are implemented through MongoDB references:

One-to-Many Relationships:

User → Appointments: **assignedTo** field in appointments references user **_id**

Patient → Appointments: **patient** field in appointments references patient **_id**

Patient → History: Medical history records reference patient **_id**

Virtual Relationships:

The Patient model includes a virtual **appointments** field that populates related appointments, enabling easy navigation from patient to their appointments without storing redundant data.

Index Strategy

The system implements strategic indexing:

- Unique indexes on email fields in both User and Patient collections
- Unique index on phone field in Patient collection
- These support the relationship integrity and query performance

This mapping approach allows the PRMS system to maintain object-oriented design principles while leveraging MongoDB's document-based storage efficiently.

3.5.2.2 Normalization

Database Normalization Analysis for PRMS Backend

The PRMS system uses MongoDB with Mongoose schemas, but we can still apply relational normalization principles to analyze the data structure design.

1. Users Collection (userModel)

Current Schema Structure:

_id (ObjectId - Primary Key)

fullName, email, role, password, active, passwordResetToken, passwordResetExpires

Normalization Analysis:

1NF: All attributes are atomic (single values)

2NF: No partial dependencies - all non-key attributes fully depend on _id

3NF: No transitive dependencies - all attributes directly relate to the user entity

2. Patients Collection (patientsModel)

Current Schema Structure:

_id (ObjectId - Primary Key)

firstName, lastName, dob, gender, email, phone, address

Virtual fields: age, appointments

Normalization Analysis:

1NF: All attributes are atomic

2NF: All non-key attributes fully functionally dependent on patient_id

3NF: No transitive dependencies

The virtual age field is calculated from dob, maintaining normalization while providing computed convenience.

3. Appointments Collection (appointmentsModel)

Current Schema Structure:

_id (ObjectId - Primary Key)

date, reason, patient (ObjectId ref), assignedTo (ObjectId ref), status

Normalization Analysis:

1NF: All fields are atomic

2NF: All non-key attributes fully functionally dependent on appointment_id

3NF: No transitive dependencies - foreign keys properly reference other entities

The patient and assignedTo fields use proper foreign key references, maintaining referential integrity.

4. Medical History Collection (historyModel)

Current Schema Structure:

_id (ObjectId - Primary Key)

patient (ObjectId ref), diagnosis, treatment, medications (Array), date

Normalization Analysis:

1NF: medications is an array - technically violates 1NF in relational terms

2NF: All non-key attributes fully functionally dependent on treatment_id

3NF: No transitive dependencies

Note: The medications array is acceptable in MongoDB's document model but would require normalization in a relational database.

5. Audit Logs Collection (auditLogModel)

Current Schema Structure:

_id (ObjectId - Primary Key)

action, user (embedded object), target (embedded object with dynamic ref), details, createdAt

Normalization Analysis:

1NF: Embedded objects (user, target) violate strict 1NF

2NF: All attributes relate to the specific audit log entry

3NF: No transitive dependencies within the document context

The embedded structure is optimized for MongoDB's document model and audit log query patterns.

Properly Normalized Aspects:

1. Primary Keys: All collections use MongoDB's _id as primary key
2. Foreign Key References: Proper ObjectId references between collections

3. Atomic Values: Most fields contain single, indivisible values
4. No Redundancy: User information isn't duplicated across patient records

MongoDB-Specific Design Decisions:

1. Arrays: medications array in history model trades strict 1NF for query efficiency
2. Embedded Documents: Audit logs embed user info for performance
3. Virtual Fields: Calculated fields like age maintain normalization while providing convenience

3.5.3 User interface design

The user interface design of the Patient Record Management System (PRMS) for Dr Seidnur Dental Clinic prioritizes straightforwardness, clarity, and ease of use for all user roles such as administrators, doctors, and receptionists. The application is developed as a web-based platform with a responsive design, allowing users to easily access it from desktops, laptops, or tablets via any contemporary browser. The interface showcases a tidy dashboard that displays essential information and fast access to main functions like patient registration, appointment booking, and treatment record handling. Navigation is intuitive due to uniform menus and clearly marked buttons, which lowers the learning curve for newcomers and decreases the chances of mistakes. Every user role receives a customized view, showing only the modules and actions pertinent to their duties, thereby improving usability and security. Forms incorporate validation checks and informative tooltips to assist users in entering data, while tables and search functionalities facilitate effective retrieval and organization of records. Moreover, the colour palette and design are selected to enhance readability and accessibility, guaranteeing that the system is approachable for users with different degrees of technical skill. In general, the PRMS interface is designed to enhance workflow, increase precision, and assist the clinic's objective of providing top-notch patient care via effective digital practices.

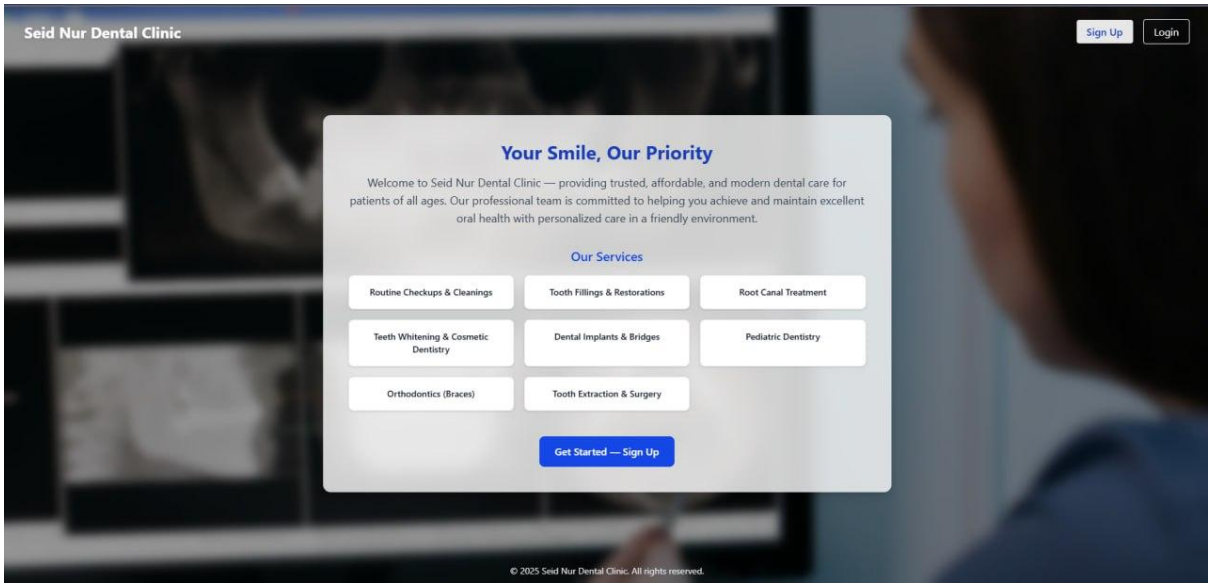


Figure 16 Landing page

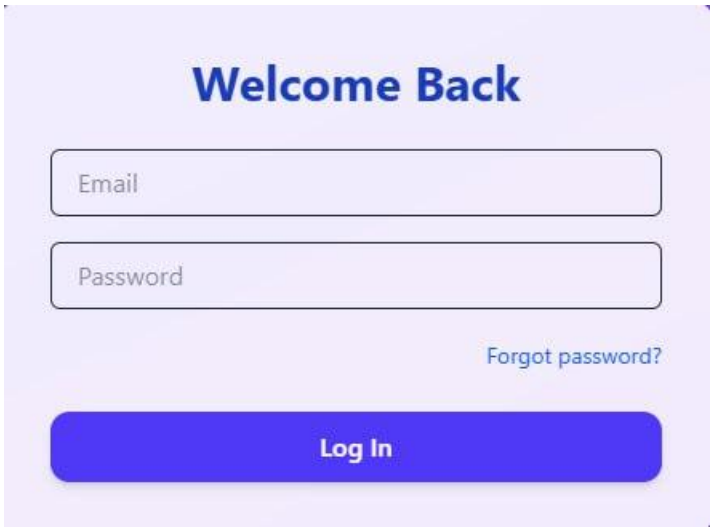


Figure 17 Login page

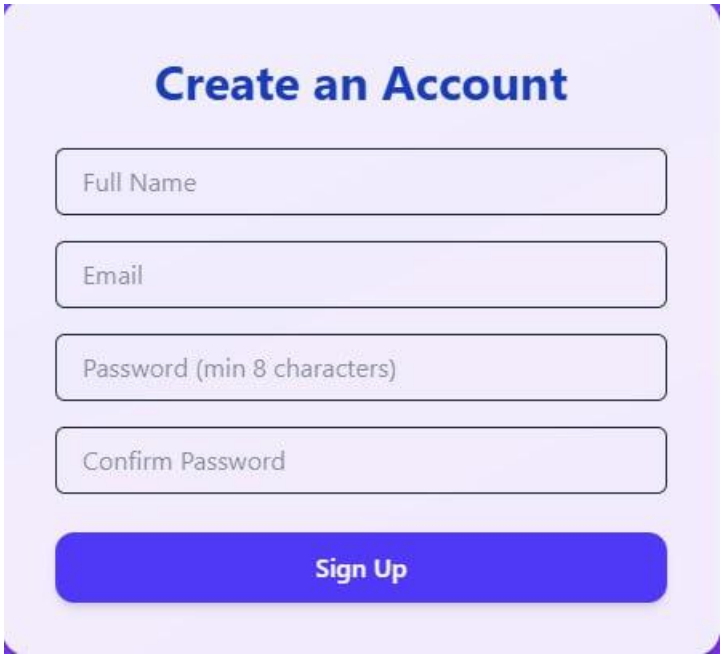


Figure 18 Register page

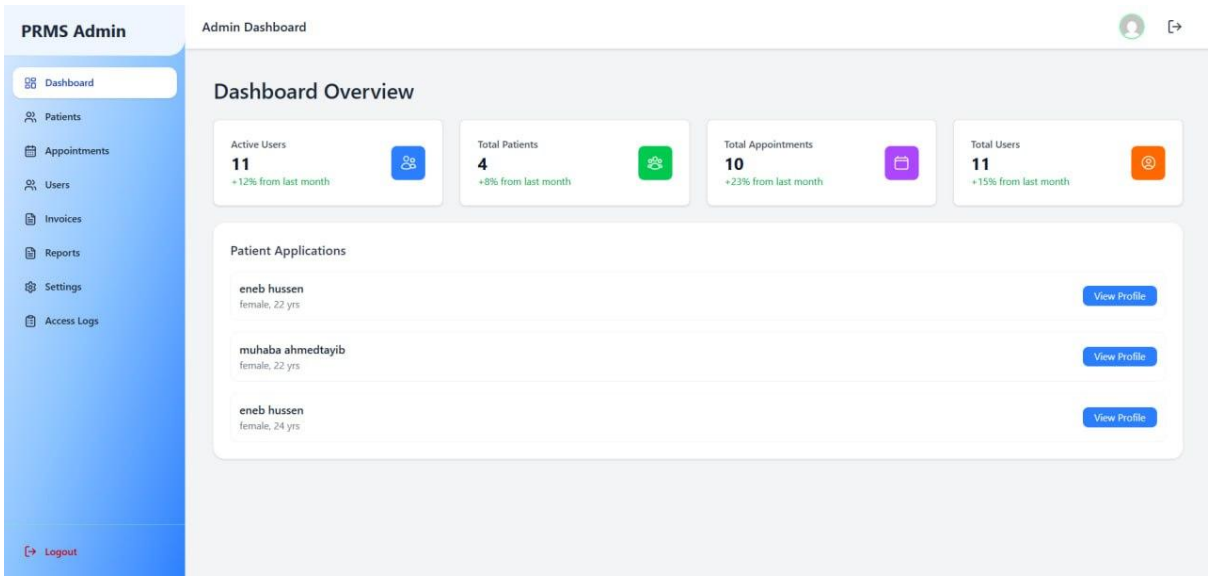


Figure 19 Dashboard overview

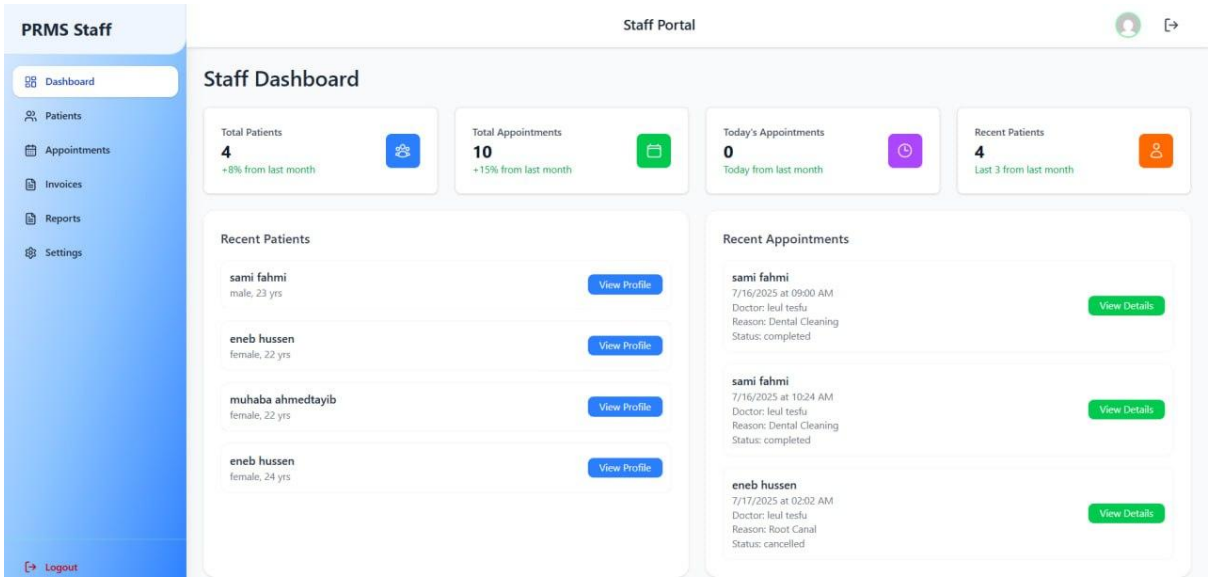


Figure 20 Staff dashboard

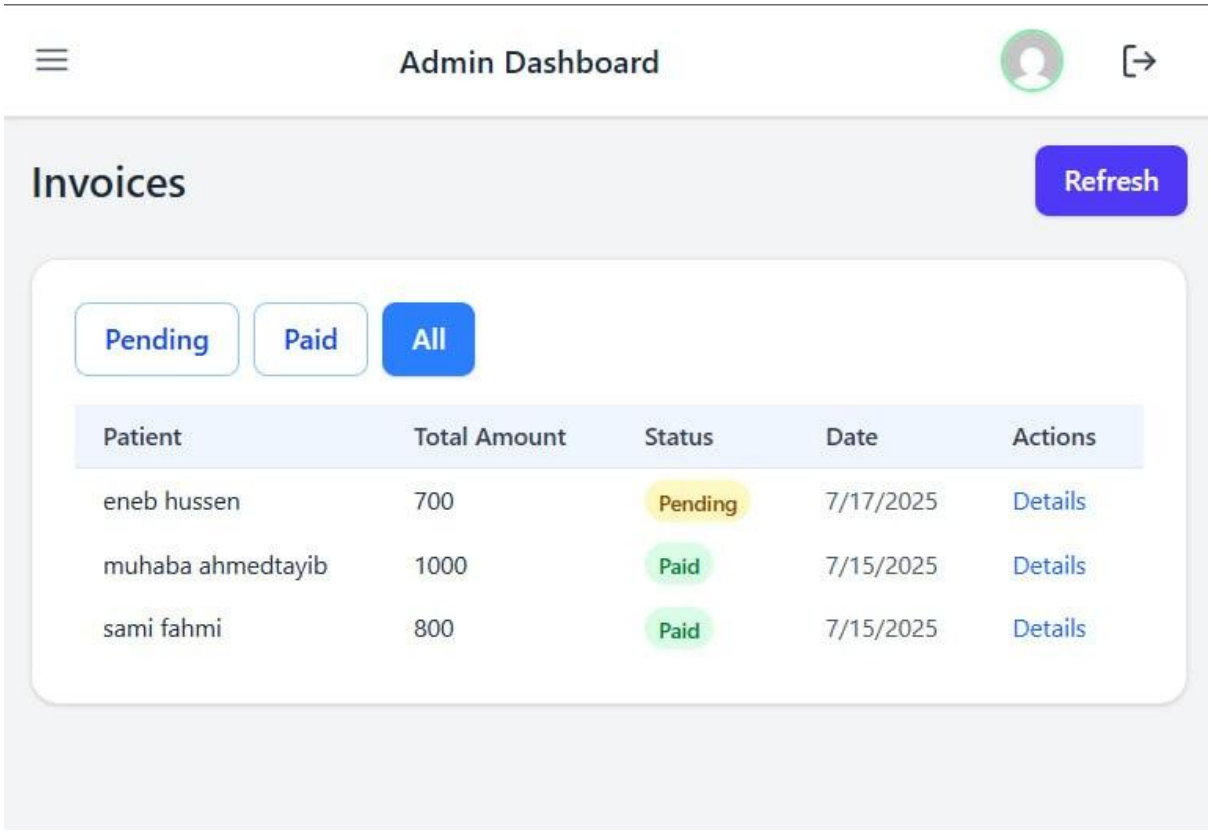


Figure 21 invoice

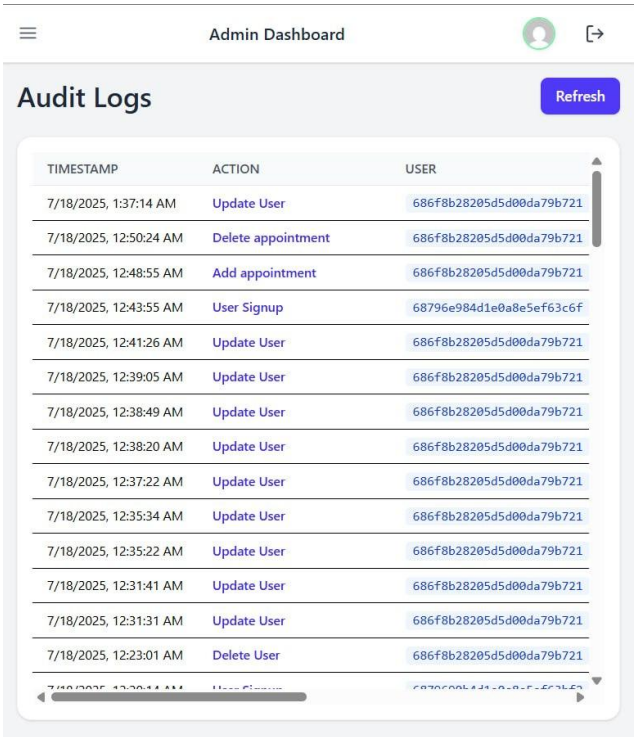


Figure 22 Audit log

3.5.4 Deployment Diagram

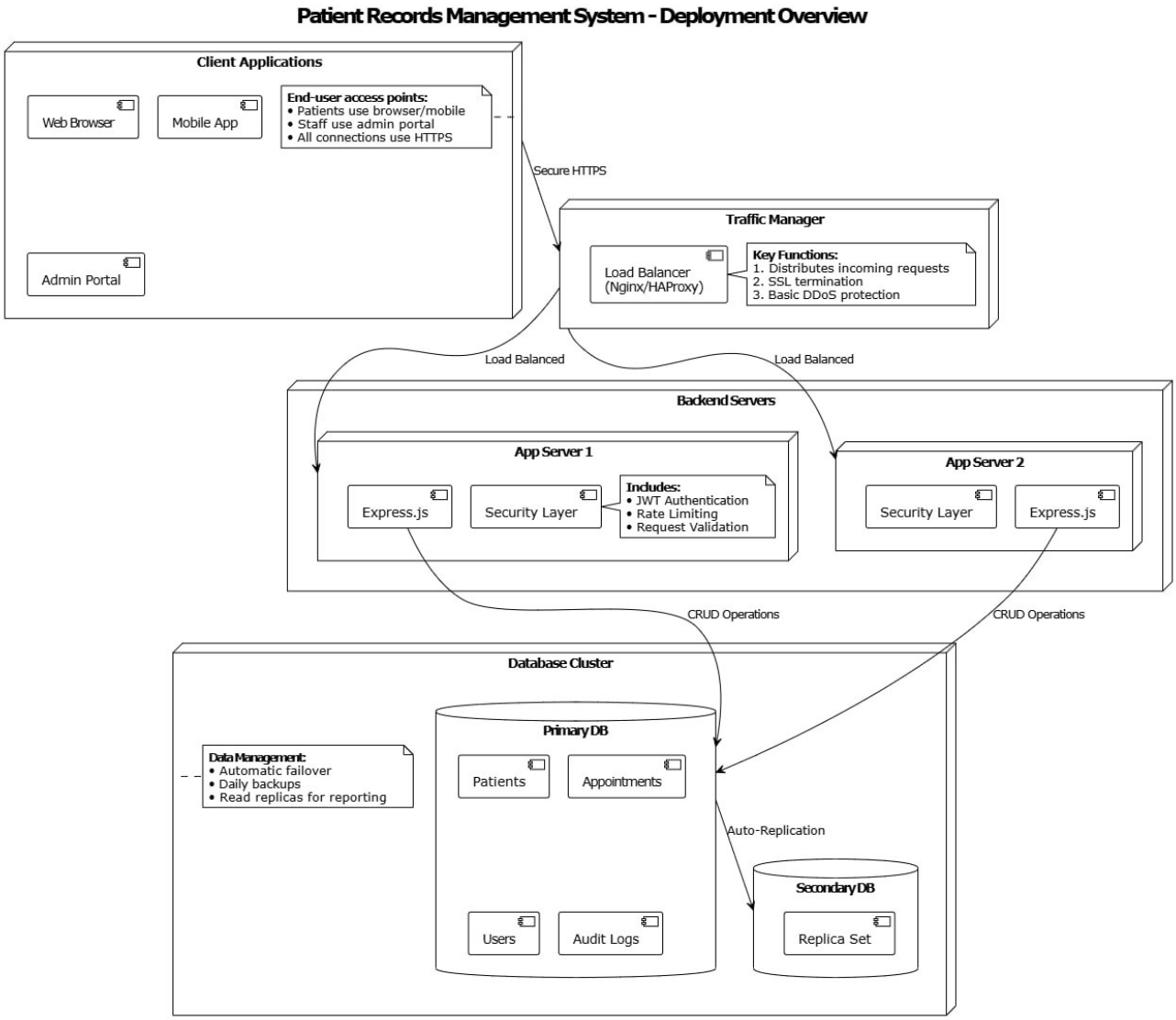


Figure 23 Deployment Diagram

The Deployment Diagram for the Patient Record Management System (PRMS) at Dr. Seidnur Dental Clinic illustrates how the system components are distributed and interact within a real-world environment. It helps to visualize how users access the system, where data is processed, and where it is stored.

Key Elements in the Deployment Diagram:

- **Client Application Layer:** Includes web browsers and mobile apps that the clinic's staff or doctors use to access the PRMS system via the internet.
- **Traffic Manager / Load Balancer:** Ensures that incoming requests are efficiently distributed across multiple backend servers, improving performance and reliability.
- **Backend Application Layer:** Contains the core logic of the system, implemented using Node.js and Express. It includes modules such as authentication, API routing, and business logic.
- **Security Layer:** Handles login verification and access control, ensuring that each user only accesses the data permitted by their role.
- **Database Cluster:** Stores all the system's persistent data such as users, patients, appointments, treatments, audit logs, etc. MongoDB is used to handle this data securely and efficiently.

This deployment model supports scalability, modularity, and remote accessibility, allowing the system to be used in a real clinic environment while also being easy to maintain and expand in the future.

3.5.5 Network design

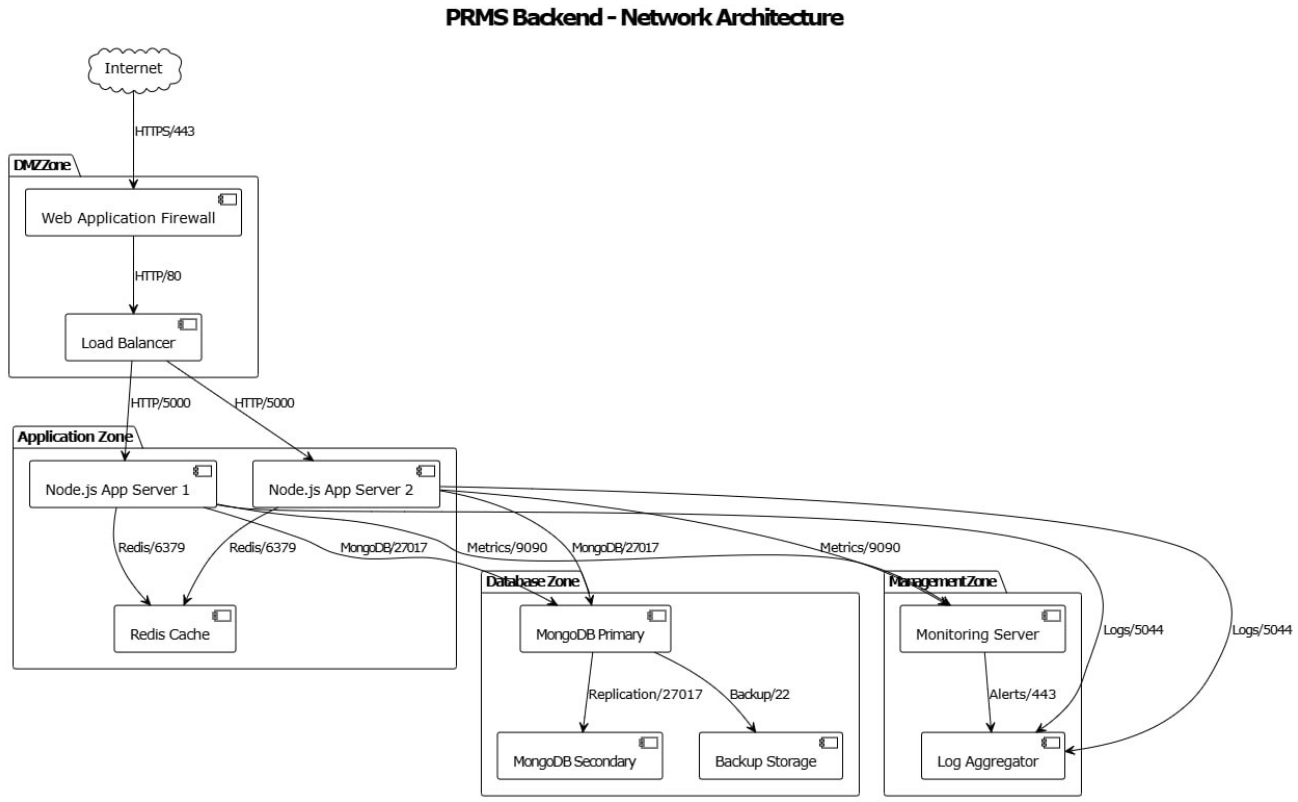


Figure 214 Network design

The Network Design diagram provides a more technical, backend-focused view of how the PRMS system is structured across servers and how data flows between components.

Key Elements in the Network Design:

- **DNS Server and Load Balancer:** Users first connect to the system via a domain, which routes through a load balancer to the available backend servers.
- **Node.js Application Server:** This is where most of the system's logic runs. It handles client requests, processes them through authentication and validation layers, and communicates with the database.
- **Caching Layer (Optional):** A Redis cache may be used to store frequently accessed data and reduce database load.
- **Database Layer:** The MongoDB database stores all clinical and administrative records. It may use replica sets for redundancy and high availability.

- **Logging and Monitoring Tools:** Log aggregators and monitoring services are included to track usage, detect errors, and monitor system health.
- **Backup and Recovery Systems:** Ensure that clinic data is never lost by providing automatic backups and the ability to restore from them if needed.

This network architecture enhances security, performance, and fault tolerance. By separating each responsibility across layers, it ensures that even if one part fails, the rest of the system remains functional.

Chapter 4 Implementation

4.1 Introduction

This section details the execution phase of the Patient Record Management System (PRMS) developed for Dr. Seidnur Dental Clinic. After evaluating the specifications and completing the system design, the next step was to build the system by coding, creating the database, and integrating all operational components. The implementation transforms the theoretical design into a working, usable software application.

The PRMS is a web-based tool that allows clinic staff to safely manage patient files, treatment records, and appointment schedules. It was created using modern web technologies and follows a modular, component-based architecture. The frontend was developed using React.js, while the backend API was built with Node.js, utilizing the Express framework. The data is stored in a MongoDB NoSQL database, and the entire system is connected through RESTful APIs.

This phase was carried out in gradual steps, following Agile principles. Development tasks were divided into shorter sprints. Features were introduced one by one, starting with fundamental functions like user authentication, patient registration, and appointment scheduling, and then advancing to medical history records and reporting modules.

Before the ultimate launch, the system experienced thorough testing at every stage. This facilitated the prompt identification and rectification of mistakes, guaranteeing that all functions performed as expected. User feedback gathered during testing sessions helped improve the system's usability and features.

Essential Implementation Responsibilities:

- **Setting Up the Development Environment:** Tools like Visual Studio Code, Git, and MongoDB Compass enabled efficient development and testing.
- **Database Schema Layout:** Key entities such as Users, Patients, Appointments, and Medical History were structured using Mongoose schemas and stored as documents within MongoDB.
- **API Development:** The backend API pathways were constructed using Express, incorporating endpoints for user authentication, administration, patient records, appointment scheduling, and medical history details.

- **Frontend Development:** React components were created for each essential feature, including login forms, registration forms, dashboards, and appointment displays. The design of the interface was created with Tailwind CSS.
- **Security and Access Management:** API routes were secured using JWT (JSON Web Tokens) and role-based access was implemented for users like Admin, Staff, and Doctor.
- **Rate Limiting and Protection:** Established thorough rate limiting with distinct thresholds for authentication endpoints (10 requests/hour) and general API endpoints (100 requests/hour) to avoid misuse and maintain system reliability.
- **Audit Logging System:** Thorough audit trail execution records every system activity for compliance and oversight functions.
- **Error Management and Verification:** Unified error management utilizing a custom AppError class along with thorough input validation for every endpoint.
- **Integration and Deployment:** The backend was deployed on Render, a cloud hosting provider, with correct environment setup and database connection management. The deployment environment ensured internet access through suitable CORS setup and trusted proxy configurations.
- **API Documentation:** An interactive Swagger UI was incorporated to offer detailed specifications and testing functionalities for API endpoints.

The deployment effectively provides a strong healthcare management solution featuring enterprise-grade security, detailed audit trails, and a scalable infrastructure ideal for production use.

Technologies Used

Component	**Technology**
Frontend	React.js, CSS Tailwind

Backend	Node.js, Express.js
Database	MongoDB with Mongoose
API Security	JWT (JSON Web Token) authentication
Hosting	Render (backend deployment), GitHub (code)
Development Tools	Visual Studio Code, Postman, Git

Table 3 Technology used

4.2 Sample code

```
const AdminLayout = () => {
            <Sidebar role="admin" />
          </div>
          {useSelector((state) => state.admin.sidebarOpen) && (
            <div
              onClick={() => dispatch(setSidebarOpen(false))}
              className="fixed inset-0 bg-black/30 z-30 md:hidden"
            />
          )}
          <div className="flex-1 flex flex-col overflow-hidden">
            <Navbar />
            <main className="flex-1 overflow-x-hidden overflow-y-auto bg-gray-100">
              <Outlet />
            </main>/.flex-1.overflow-x-hidden.overflow-y-auto.bg-gray-100
          </div>/.flex-1.flex.flex-col.overflow-hidden
        </div>/.flex.w-full.min-h-screen.bg-gray-100
        {showLogoutConfirm && (
          <div className="fixed inset-0 z-50 flex items-center justify-center bg-black bg-opacity-30">
            <div className="bg-white p-6 rounded shadow">
              <p className="mb-4">Are you sure you want to logout?</p>
              <div className="flex gap-4">
                <button
                  className="px-4 py-2 bg-red-600 text-white rounded"
                  onClick={handleLogout}
                >
                  Yes, Logout
                </button>/.px-4.py-2.bg-red-600.text-white.rounded
                <button
                  className="px-4 py-2 bg-gray-200 rounded"
                  onClick={() => setShowLogoutConfirm(false)}
                >
                  Cancel
                </button>/.px-4.py-2.bg-gray-200.rounded
              </div>/.flex.gap-4
            </div>/.bg-white.p-6.rounded.shadow
          </div>/.fixed.inset-0.z-50.flex.items-center.justify-center.bg-black.bg-opacity-30
        )}
```

Figure 25 Admin Dashboard Code

Figure 26 Admin Dashboard OutputFigure 25 Admin Dashboard Code

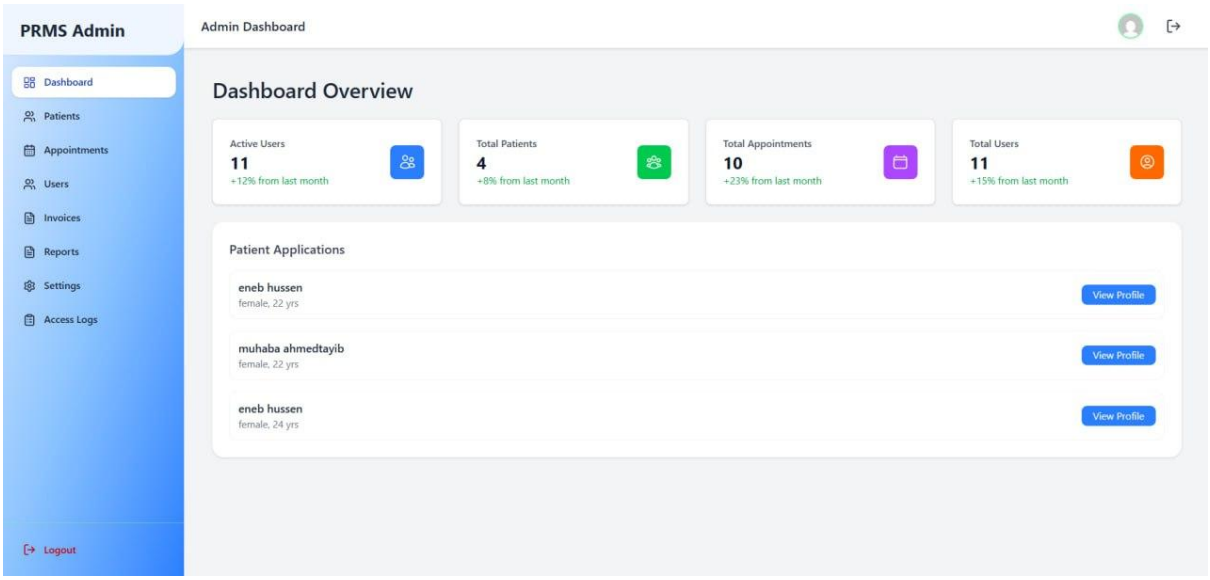


Figure 26 Admin Dashboard Output

Figure 27 Medical History Code

Figure 26 Admin Dashboard Output

ing

age

change indicators and color-coded icons. It includes a “Patient Applications” section listing recent patients with quick access to detailed profiles. The dashboard integrates with Redux state management, includes loading states with spinner components, handles error states with toast notifications, and uses a clean card-based layout with responsive design for efficient system monitoring and patient data access.error states with toast notifications, and uses a clean card-based layout with responsive design for efficient system monitoring and patient data access.

```jsx
const MedicalHistoryModal = ({ patient, patientId, onClose }) => {
        {!loading && !showForm && (
          <>
            <button
              onClick={handleAddNew}
              className="mb-4 bg-blue-500 hover:bg-blue-600 text-white px-4 py-2 rounded-lg"
            >
              Add New Entry
            </button>/.mb-4.bg-blue-500.hover:bg-blue-600.text-white.px-4.py-2.rounded-lg
            {list.length === 0 ? (
              <div className="text-gray-400 text-center">No medical history found.</div>
            ) : (
              <div className="space-y-4">
                {list.map((entry) => (
                  <div
                    key={entry._id}
                    className="flex flex-col sm:flex-row sm:items-center sm:justify-between gap-2 border border-gray-100 rounded-xl p-4
                    bg-gray-50"
                  >
                    <div>
                      <div className="font-semibold text-lg text-gray-800">{entry.diagnosis}</div>
                      <div className="text-sm text-gray-600">{entry.treatment}</div>
                      <div className="text-xs text-gray-500 mt-1">
                        Medications: {entry.medications?.join(", ") || "N/A"}
                      </div>/.text-xs.text-gray-500.mt-1
                      <div className="text-xs text-gray-500">
                        Date: {entry.date ? new Date(entry.date).toLocaleDateString() : "N/A"}
                      </div>/.text-xs.text-gray-500
                      <div className="text-xs text-gray-500">Reason: {entry.reason || "-"}</div>
                    </div>
                    <div className="flex flex-col xs:flex-row gap-2 mt-2 sm:mt-0 w-full sm:w-auto">
                      <button
                        onClick={() => handleEdit(entry)}
                        className="px-3 py-1 bg-yellow-400 hover:bg-yellow-500 text-gray-800 rounded-lg text-sm w-full xs:w-auto"
                      >
                        Edit
                      </button>/.px-3.py-1.bg-yellow-400.hover:bg-yellow-500.text-gray-800.rounded-lg.text-sm.w-full.xs:w-auto
```

Figure 27 Medical History Code

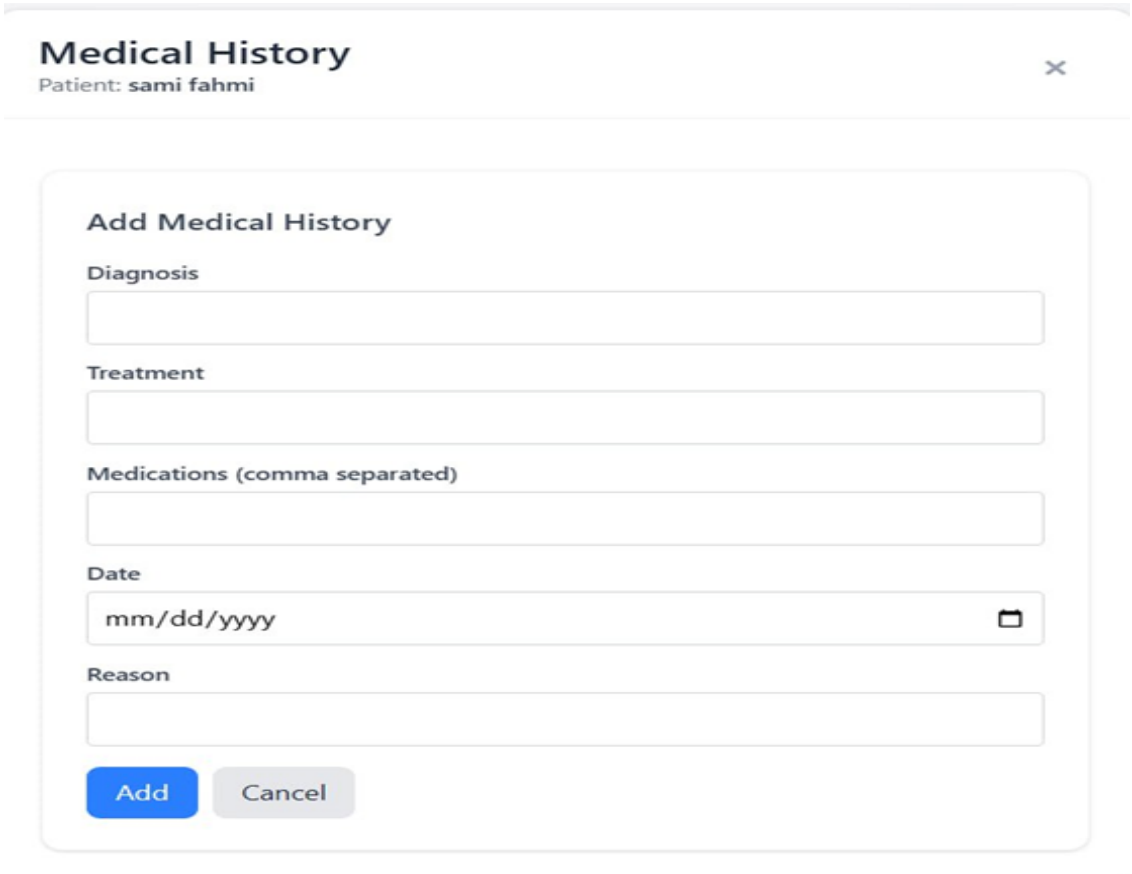


Figure 28 Medical History Output

Figure 30 Access Log OutputFigure 28 Medical History Output

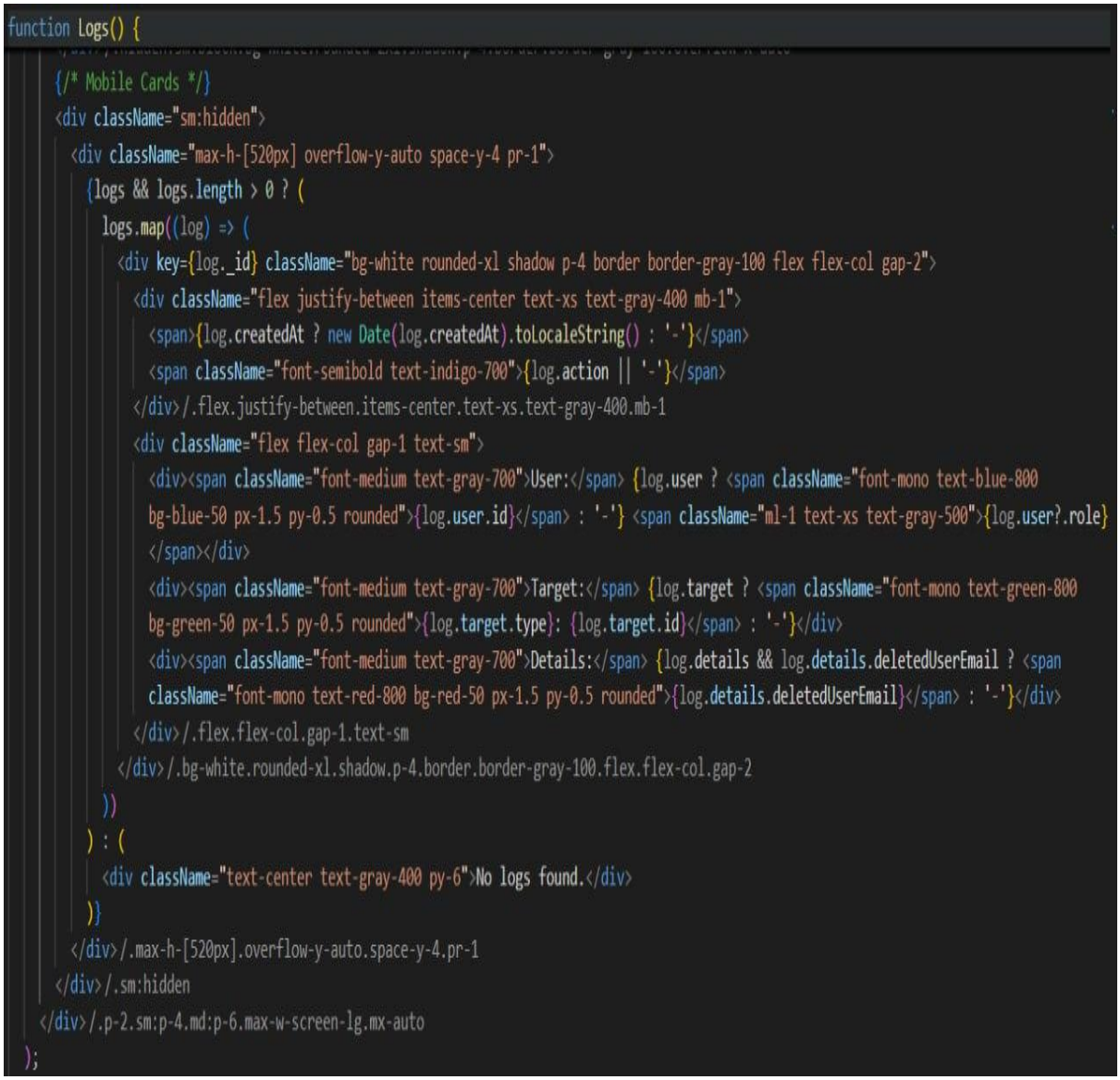
Figure 31 Invoice List CodeFigure 26 Access Log Code

Figure 30 Invoice List CodeFigure 27 Access Log Code

Figure 30 Invoice List CodeFigure 28 Access Log Code

Figure 30 Invoice List CodeFigure 29 Access Log Code

management, provides real-time feedback through toast notifications, and includes additional functionality like generating invoices for medical entries. The interface uses color-coded actions (blue for primary, yellow for edit, red for delete, green for invoices) and responsive design for optimal user experience.



```
function Logs() {
    {/* Mobile Cards */}
    <div className="sm:hidden">
      <div className="max-h-[520px] overflow-y-auto space-y-4 pr-1">
        {logs && logs.length > 0 ? (
          logs.map((log) => (
            <div key={log._id} className="bg-white rounded-xl shadow p-4 border border-gray-100 flex flex-col gap-2">
              <div className="flex justify-between items-center text-xs text-gray-400 mb-1">
                <span>{log.createdAt ? new Date(log.createdAt).toLocaleString() : '-'}</span>
                <span className="font-semibold text-indigo-700">{log.action || '-'}</span>
              </div>{/* .flex.justify-between.items-center.text-xs.text-gray-400.mb-1
              <div className="flex flex-col gap-1 text-sm">
                <div><span className="font-medium text-gray-700">User:</span> {log.user ? <span className="font-mono text-blue-800
                bg-blue-50 px-1.5 py-0.5 rounded">{log.user.id}</span> : '-'} <span className="ml-1 text-xs text-gray-500">{log.user?.role}
                </span></div>
                <div><span className="font-medium text-gray-700">Target:</span> {log.target ? <span className="font-mono text-green-800
                bg-green-50 px-1.5 py-0.5 rounded">{log.target.type}: {log.target.id}</span> : '-'}</div>
                <div><span className="font-medium text-gray-700">Details:</span> {log.details && log.details.deletedUserEmail ? <span
                className="font-mono text-red-800 bg-red-50 px-1.5 py-0.5 rounded">{log.details.deletedUserEmail}</span> : '-'}</div>
              </div>{/* .flex.flex-col.gap-1.text-sm
            </div>{/* .bg-white.rounded-xl.shadow.p-4.border.border-gray-100.flex.flex-col.gap-2
          ))
        ) : (
          <div className="text-center text-gray-400 py-6">No logs found.</div>
        )}
      </div>{/* .max-h-[520px].overflow-y-auto.space-y-4.pr-1
    </div>{/* .sm:hidden
  </div>{/* .p-2.sm:p-4.md:p-6.max-w-screen-lg.mx-auto
);
```

Figure 29 Access log code

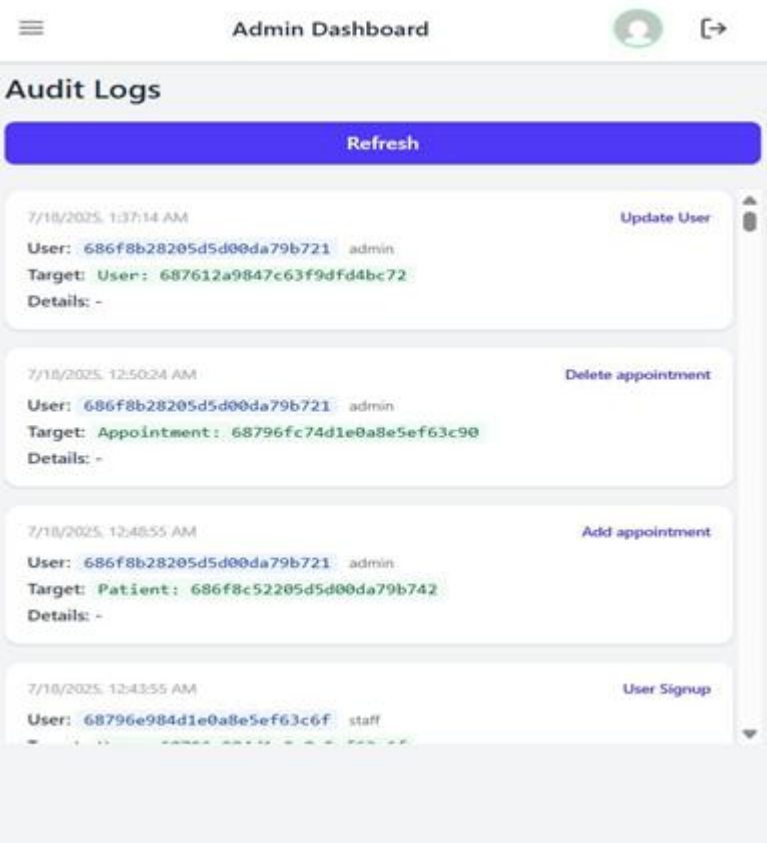


Figure 30 Access Log Output

Figure 30 Access Log Code

Figure 30 Invoice List CodeFigure 31 Access Log Code

Figure 30 Invoice List CodeFigure 32 Access Log Code

Figure 30 Invoice List CodeFigure 33 Access Log CodeFigure 30 Access Log Output

and user activity records in a
esign with a desktop table view
ns, and a mobile card layout for
for real-time log updates, loading
iendly messages, and scrollable
mobile) for better visibility. The
r users, green for targets, red for
hrough logSlice for efficient data
retrieval and display.

```
const InvoiceList = () => {
        <div className="overflow-x-auto">
          <table className="min-w-full divide-y divide-gray-200 text-sm md:text-base">
            <thead>
              <tr className="bg-blue-50">
                <th className="px-4 py-2 text-left font-semibold text-gray-700">
                  Patient
                </th>/.px-4.py-2.text-left.font-semibold.text-gray-700
                <th className="px-4 py-2 text-left font-semibold text-gray-700">
                  Total Amount
                </th>/.px-4.py-2.text-left.font-semibold.text-gray-700
                <th className="px-4 py-2 text-left font-semibold text-gray-700">
                  Status
                </th>/.px-4.py-2.text-left.font-semibold.text-gray-700
                <th className="px-4 py-2 text-left font-semibold text-gray-700">
                  Date
                </th>/.px-4.py-2.text-left.font-semibold.text-gray-700
                <th className="px-4 py-2 text-left font-semibold text-gray-700">
                  Actions
                </th>/.px-4.py-2.text-left.font-semibold.text-gray-700
              </tr>/.bg-blue-50
            </thead>
            <tbody>
```

Figure 31 Invoice List Code

—

Figure 32 Invoice List OutputFigure 31 Invoice List Code

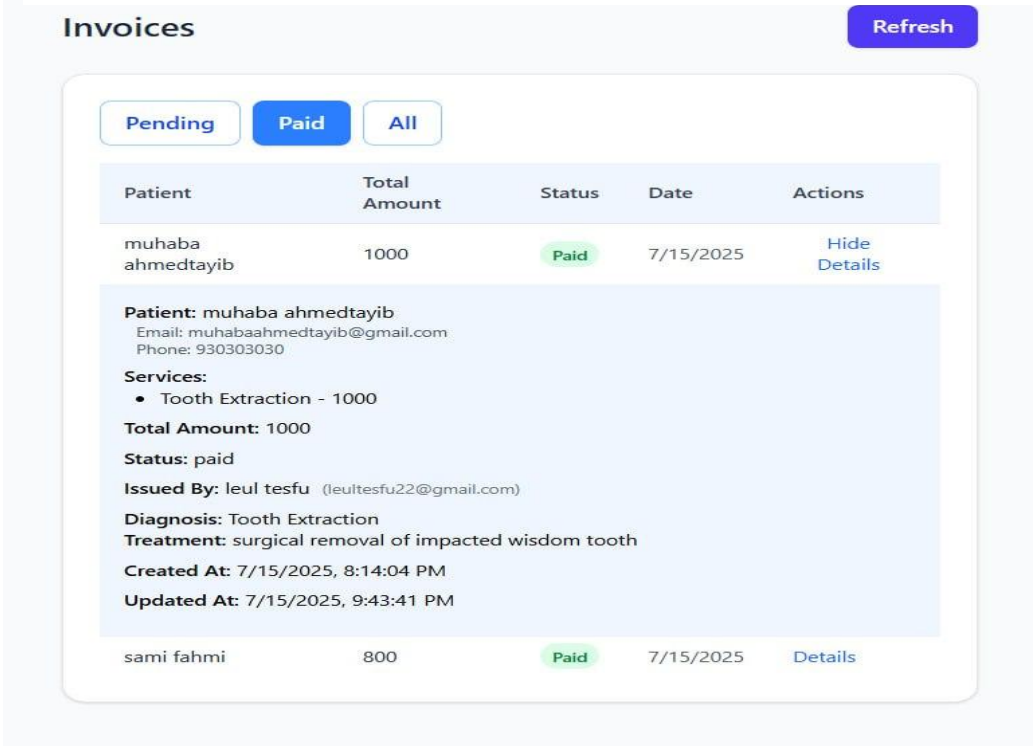


Figure 32 Invoice List Output

The Invoice List component manages billing records and payment tracking for the PRMS system. It features a comprehensive table interface displaying patient information, total amounts, payment status, and creation dates with expandable rows for detailed invoice information. The component includes filtering capabilities (Pending, Paid, All) with color-coded status badges (green for paid, yellow for pending), a refresh button for real-time updates, and detailed view functionality showing patient contact details, service breakdowns, medical history, and issuer information. It integrates with Redux state management through invoiceSlice, provides payment status updates with loading states and toast notifications, and includes responsive design with hover effects and transition animations for optimal user experience.

Chapter 5 Conclusion and Recommendations

5.1 conclusion

The development of the Patient Record Management System (PRMS) for Dr Seidnur Dental Clinic represents a major improvement in the clinic's capacity to handle patient data and optimize everyday functions. The detailed use case diagram demonstrates the system's extensive capabilities, facilitating various tasks including patient registration, appointment scheduling, treatment record handling, user management, and audit logging. Through the use of a web-based, responsive system, the clinic now enjoys better accessibility, improved data protection, and streamlined workflows that minimize manual mistakes and administrative burdens.

The PRMS implements rigorous role-based access control, guaranteeing that sensitive information is available solely to authorized individuals and that all essential actions are monitored for accountability. Automated systems, including conflict checks for appointments and real-time data verification, improve the trustworthiness and precision of the clinic's operations. The audit log module specifically offers a clear documentation of all notable system actions, facilitating internal assessments and adherence to legal obligations.

In general, the system tackles the drawbacks of the earlier manual, paper-driven method by centralizing patient information, streamlining report creation, and allowing quick access to data. This digital transformation not only addresses the clinic's present requirements but also establishes a solid groundwork for future development and connectivity with wider healthcare technologies. The PRMS ultimately enables Dr Seidnur Dental Clinic to provide better patient care, enhance operational efficiency, and sustain a secure and accountable digital setting.

5.2 Recommendations

Although the PRMS fulfill the fundamental requirements and provides key functionalities, the subsequent suggestions are put forth for potential improvements:

- **Enhance Feature Range:** Add extra components like billing, stock control, and connections with lab systems to better optimize clinic functions.
- **Strengthen Security:** Adopt multi-factor authentication and conduct frequent security audits to better safeguard sensitive patient information.

- **Enhance Reporting:** Create sophisticated analytics and tailored reporting tools to facilitate data-informed decision-making.
- **User Training:** Deliver detailed training resources and user assistance to ensure seamless integration and efficient utilization of the system.
- **Scalability:** Optimize performance and enable distributed data management to ready the system for use in larger or multi-location healthcare facilities.
- **Mobile Accessibility:** Create a mobile-optimized site or a specific mobile application to enhance accessibility for healthcare professionals and employees while on the move.

5.3 Future work

The existing Patient Record Management System (PRMS) at Dr Seidnur Dental Clinic fulfills key functions like patient registration, appointment scheduling, treatment record management, user administration, and audit logging; however, there are still numerous potential avenues for future improvement.

A significant focus for upcoming efforts is the combination of billing and insurance modules. Incorporating these features would allow the clinic to handle patient payments, create invoices, and manage insurance claims directly in the system, enhancing administrative workflows. Moreover, linking the PRMS to outside healthcare systems or national health databases might enhance care continuity and enable data sharing between clinics and hospitals, while adhering to privacy and regulatory standards.

An additional beneficial expansion would be creating a specific mobile app for both employees and patients. A mobile application could give doctors and receptionists access to schedules and patient data while enabling patients to schedule appointments, get reminders, and check their treatment history from afar. Improving the analytics and reporting features of the system through dashboards and visualizations would enable clinic management to make better, data-informed decisions.

Ultimately, continuous enhancements in security, including the adoption of two-factor authentication and frequent penetration testing, will guarantee that the system stays resilient against new threats. Regular collection of user feedback and usability studies will inform iterative improvements, making sure the PRMS keeps evolving to meet the clinic's requirements and technological progress in healthcare.

5.4 References

Sommerville, I. (2016). Software Engineering (10th ed.). Pearson Education.

Dennis, A., Wixom, B. H., & Tegarden, D. (2015). Systems Analysis and Design (6th ed.). Wiley.

MongoDB, Inc. (2024). MongoDB Documentation. Retrieved from https://www.mongodb.com/docs/

React.js. (2024). React – A JavaScript Library for Building User Interfaces. Retrieved from https://react.dev/

Node.js Foundation. (2024). Node.js Documentation. Retrieved from https://nodejs.org/en/docs/

Express.js. (2024). Express – Node.js web application framework. Retrieved from https://expressjs.com/

Tailwind Labs. (2024). Tailwind CSS Documentation. Retrieved from https://tailwindcss.com/docs

JWT.io. (2024). JSON Web Tokens Introduction. Retrieved from https://jwt.io/introduction

Render. (2024). Render Documentation. Retrieved from https://render.com/docs

GitHub. (2024). GitHub Docs. Retrieved from https://docs.github.com/

Appendix

A. System Screenshots

1. **Landing Page**

   (Figure 16 in main document)

   - Displays clinic statistics (patients served, appointments, staff).
   - Provides quick access to login and registration.
2. **Login Page**

   (Figure 17 in main document)

   - Secure JWT-based authentication.
   - Role-based redirection for admin, doctor, staff, and patient users.
3. **Registration Page**

   (Figure 18 in main document)

   - Validates input fields such as email and password strength.
   - Ensures all required fields are completed.
4. **Admin Dashboard**

   (Figure 19 in main document)

   - Displays an overview of patients, appointments, and system analytics.
5. **Staff Dashboard**

   (Figure 20 in main document)

   - Designed for appointment management and accessing patient records.

Appendix B: Sample Code

1. **User Login Function (Frontend)**

   (Figure 23–24 in main document)
2. **Patient Registration (Backend)**

   (Figure 25–26 in main document)
3. **API Endpoints**

Endpoint	Method	Description
/api/v1/auth/login	POST	User authentication

Endpoint	Method	Description
/api/v1/patients	GET	Fetch all patients
/api/v1/appointments	POST	Schedule new appointment

Appendix C: Database Schema

(Refer to Section 3.5.2 for ERD and normalization)

Collections in MongoDB:

- **Users**
  Fields: _id, fullName, email, role, password (hashed), active
- **Patients**
  Fields: _id, firstName, lastName, dob, phone, email
- **Appointments**
  Fields: _id, patientId, doctorId, date, status

Appendix D: Deployment Details

1. **Hosting Platforms**
   - **Frontend:** Static hosting via Netlify or Vercel
   - **Backend:** Node.js app deployed on Render or Heroku
   - **Database:** MongoDB Atlas (cloud-based) or local MongoDB instance
2. **Security Measures**
   - JWT-based user authentication
   - API rate limiting (100 requests per hour)
   - Audit logs to track all CRUD operations

Appendix E: Testing Scenarios

Test Case	Expected Result
User login with valid credentials	Redirects to the appropriate user dashboard
Invalid login attempt	Displays error message
Admin deletes patient	Patient record is deleted and logged

Appendix F: User Manual

A brief guide for clinic staff

1. **How to Register a Patient**
    - Navigate to **Patients > Add New**
    - Fill in patient details (name, date of birth, contact info)
    - Click **Save** to store the record
2. **How to Schedule Appointments**
    - Use the calendar interface on the dashboard
    - Ensure no overlapping appointments (system prevents double-booking)

Appendix G: Project Timeline (Gantt Chart)

(Refer to Figure 1 in main document)

Phase	Duration
Requirement Analysis & Design	January – March
Development	April – May
Testing & Deployment	June