Sri Lanka Institute of Information Technology

Project Proposal Report
Information Technology Project (IT2080)
2023
ITP_2023_Y2_S2_WE10

Innobot Health Care Revenue Cycle Management System

Submitted by:

	Name with Initials	Registration Number	Contact Number	Email
1.	S.A.N. Bamunusinghe	IT22515612	0772187484	it22515612@my.sliit.lk
2.	DME Wimalagunasekara	IT22917270	0768952222	it22917270@my.sliit.lk
3.	S A T Nayanapriya	IT22892058	0725203742	it22892058@my.sliit.lk
4.	Obeyesekere A D	IT22332080	0774801500	it22332080@my.sliit.lk
5.	Chamikara M G S	IT22905840	0712905241	it22905840@my.sliit.lk
6.	M F M Farsith	IT22354556	0770494812	it22354556@my.sliit.lk
7.	D S I Gamage	IT22907516	0779705099	it22907516@my.sliit.lk
8.	W A Malsha Haren	IT22307576	0788760703	it22307576@my.sliit.lk

Contents

Background

Company Background

Innobot health is an US based company with subsidiaries in Colombia, Sri Lanka, and India with a collective staff about 40 people. For our university project we chose Innobot as our client.

The company intends on building a Revenue Cycle Management based Practice Management System to automate and ease the tasks of hundreds of thousands of billers in USA.

Innobot Health is a trailblazer in healthcare technology, led by experts with a deep understanding of the industry. Their mission is to simplify healthcare processes, and our university project with them focuses on developing a cutting-edge Practice Management System. Innobot Health's intelligent automation solutions, utilizing AI and machine learning, optimize revenue cycle management. The company's collaborative approach tailors' solutions to specific stakeholder challenges. Our joint effort aims to contribute to advancing healthcare through automation, aligning with Innobot Health's commitment to excellence.

Problems and Motivations

Problems

During the requirements gathering phase of the project we discovered that, the existing healthcare information technology infrastructure faces notable challenges, marked by inefficiencies and limitations in managing various aspects of healthcare practices. The lack of a comprehensive system hinders healthcare providers in efficiently managing patient information, scheduling appointments, and handling billing processes, and effectively interacting with insurance systems. Additionally, patients often experience challenges accessing and managing their health records seamlessly. The current scenario lacks a unified solution that promotes interoperability with external healthcare systems and third-party applications. In recognizing these challenges, the development of the 'Innobot' healthcare system is imperative to address these issues and usher in a more streamlined, patient-centric, and interoperable healthcare information technology solution.

Current Process and Identified Problems:

Fragmented Practice Management:

Issue:

Healthcare providers currently rely on fragmented systems for practice management, leading to inefficiencies in patient record management, appointment scheduling, and billing processes. This fragmentation complicates day-to-day operations and may result in errors and delays.

Limited Patient Access and Communication:

Issue:

Patients face challenges in accessing their personal health records seamlessly and communicating securely with healthcare providers. The current communication channels may not be user-friendly, limiting patient engagement and satisfaction.

Non-Interoperable Systems:

Issue:

The lack of interoperability with external healthcare systems and third-party applications restricts the seamless exchange of information. This limitation hampers collaborative efforts, data sharing, and the overall efficiency of healthcare operations.

Inefficient Insurance Management:

Issue:

Healthcare providers encounter challenges in efficiently managing insurance-related processes. The current system may lack robust tools for verifying insurance details, processing claims, and ensuring timely reimbursement, leading to financial and administrative burdens.

Motivation

By addressing the identified problems and implementing the 'Innobot' healthcare system, Clients stand to gain several significant benefits,

Streamlined Practice Management:

Healthcare providers will experience streamlined practice management with integrated tools for patient record management, appointment scheduling, and billing processes. This will lead to increased operational efficiency, reduced errors, and smoother day-to-day operations.

Enhanced Patient Engagement:

The portal's patient-centric features, including online access to personal health records and secure communication channels, will empower healthcare providers to enhance patient engagement. This increased interaction and accessibility contribute to improved patient satisfaction and adherence to treatment plans.

Efficient Insurance Management:

Healthcare providers will benefit from robust tools within the 'Innobot' system for insurance management. This includes streamlined processes for verifying insurance details, processing claims, and ensuring timely reimbursement. Improved efficiency in insurance management reduces financial and administrative burdens, allowing healthcare providers to focus more on patient care.

Increased Accuracy and Compliance:

The automated features of the 'Innobot' system contribute to increased accuracy in patient data management, billing processes, and insurance-related tasks. This not only minimizes errors but also ensures compliance with healthcare regulations and standards, mitigating potential risks and liabilities.

Time and Cost Savings:

The efficiency gains from the 'Innobot' system translate into time and cost savings for healthcare providers. Automation of repetitive tasks, streamlined workflows, and reduced manual interventions contribute to optimized resource utilization, allowing healthcare professionals to allocate more time to patient care.

Enhanced Data Security:

The 'Innobot' system prioritizes data security, providing healthcare providers with a secure platform for managing patient information. Improved data security measures safeguard against unauthorized access, ensuring compliance with data protection regulations and instilling trust among patients.

Aims & Objectives

Aims

This project aims to develop a comprehensive system administration software tailored to the needs of Innobot Health, with a primary focus on enhancing the management processes within hospital operations. This software will streamline and optimize system administration tasks, bolster security measures, and elevate overall efficiency in managing the company's IT infrastructure, particularly within the context of hospital management workflows and procedures.

Objective

Process Optimization:

Identify and streamline management processes within hospital operations, such as patient admissions, discharge procedures, inventory management, and staff scheduling, through the implementation of efficient system administration software.

Integration with Hospital Management Systems:

Develop seamless integration capabilities with existing hospital management systems to ensure smooth data flow and interoperability, enabling comprehensive management oversight and analysis.

Patient Data Security Enhancement:

Enhance security measures within the software to safeguard sensitive patient data, ensure compliance with healthcare regulations such as HIPAA, and bolster trust in the management of medical records.

Operational Efficiency Improvement:

Implement features and functionalities aimed at improving operational efficiency within hospitals, such as automated appointment scheduling, real-time bed management, and optimized resource allocation based on demand forecasting.

Decision Support System Implementation:

Integrate decision support functionalities into the software to provide actionable insights and recommendations to hospital administrators, enabling informed decision-making and strategic planning.

Customization for Healthcare Environment:

Tailor the system administration software to meet the unique requirements and workflows of healthcare environments, including specialized modules for clinical workflows, diagnostic imaging, and electronic health record management.

Performance Monitoring and Reporting:

Develop robust monitoring and reporting capabilities to track key performance indicators within hospital operations, allowing for data-driven analysis, performance optimization, and compliance reporting.

Stakeholder Engagement and Feedback Incorporation:

Engage stakeholders, including hospital administrators, healthcare providers, and IT staff, throughout the development process to gather feedback and incorporate user-driven improvements, ensuring that the software meets the evolving needs and expectations of the healthcare industry.

System Overview

Introduction to System

The Practice Management System being specified in this Proposal serves as a pivotal component within Innobot' s suite of healthcare information technology (HIT) solutions. It is a new, self-contained product aimed at enhancing the efficiency and effectiveness of healthcare providers' administrative tasks, patient management, and overall practice management.

Product Features

The Practice Management portal for Innobot encompasses a range of robust features tailored to optimize healthcare information technology solutions. Key functionalities include comprehensive practice management tools for healthcare providers, facilitating seamless patient record management, appointment scheduling, and billing processes. Additionally, the portal offers patient-centric features, empowering individuals with online access to personal health records and secure communication channels. Interoperability is a cornerstone, allowing integration with external healthcare systems and third-party applications.

Operating Environment

The Innobot Practice Management System will operate in a secure and scalable environment tailored to meet the specific needs of healthcare information technology. The software is designed to be platform-independent, supporting a range of hardware configurations. The recommended operating systems include Windows Server 2016 and above.

The web portal seamlessly coexists with common web browsers, including Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge, ensuring accessibility for users across different platforms. It is optimized for compatibility with the latest versions of these browsers to provide an optimal user experience.

To ensure optimal performance, the web portal is recommended to be hosted on servers with sufficient processing power, memory, and storage capacity, and it should be deployed within a network infrastructure that prioritizes data integrity, confidentiality, and availability. Regular updates and patches for the underlying operating system and software components are essential to maintain a secure operating environment for the web portal.

The operating environment emphasizes security and compliance, aligning with industry standards such as HIPAA for the protection of sensitive patient information.

Scope of Work

The project includes the development and deployment of a Healthcare Practice Management System, covering all specified subsystems. It excludes major hardware upgrades or network infrastructure changes outside the immediate system requirements.

Project Objectives:

- Implement a fully functional Healthcare Practice Management System.
- Improve appointment scheduling processes for both staff and patients.
- Enhance inventory management to optimize stock levels and reduce shortages.
- Streamline staff management for efficient onboarding, allocation, & communication.
- Facilitate seamless claim management and integration with external billing services.
- Provide a user-friendly patient management system for accurate record-keeping.
- Implement a notification system for real-time communication and alerts.
- Enable insurance management with features for creating, updating, and reporting.

Key Features and Components:

- Centralized appointment scheduling
- Staff onboarding and performance tracking
- Inventory tracking with barcode integration
- Claim processing and external system integration
- Patient registration and record management
- Real-time notifications and alerts
- Insurance record creation and management

System Diagram

We divided our system into 8 main subsystems. These are Appointment Management System, Staff Management System, Claim Management System, Inventory Management System, Procedure/ DX management System, Notification Management System, Patient Management System, Insurance Management System.

Front end is implemented using React.JS. From the front end, an API request (rest API) will be sent to the backend which is made of Spring Boot. The API validators in the backend will receive the API request and once it's accepted from there, it will go to the authorization middleware, and later it will be sent to the endpoint's service. The service has the function for that endpoint. Finally, to access the DB, a database API request will be sent to the DB (MongoDB)

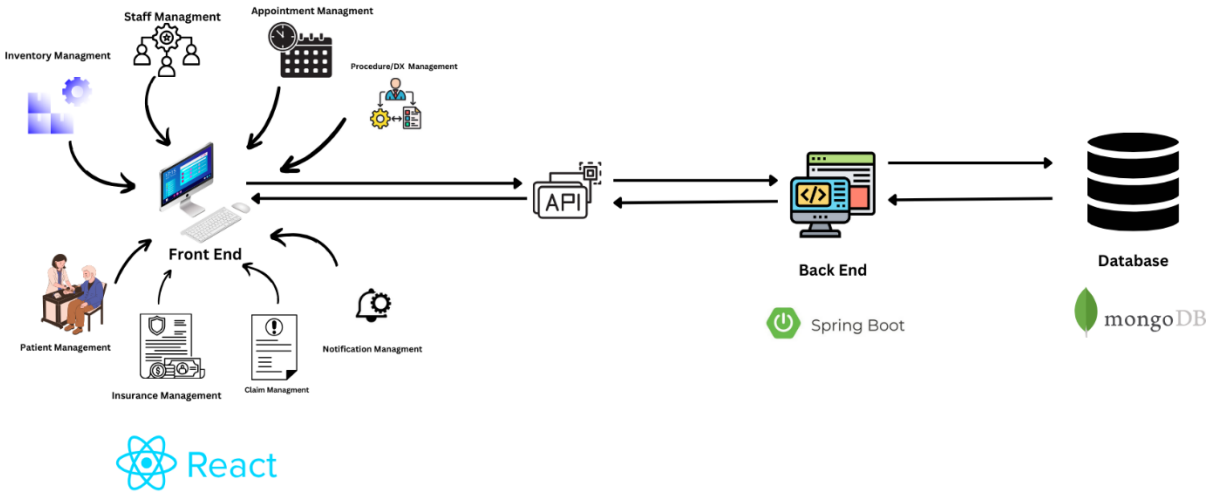


Figure 1 - System Diagram

Functional requirements

1) Inventory Management

This module allows healthcare organizations to efficiently manage their supplies. Users can add, edit, and delete supplier information, including contract details, while monitoring supplier performance through key metrics. Inventory functions include adding, editing, and deleting items, with robust search and filtering options. Seamless integration with barcode scanning technology enhances data accuracy and streamlines tracking. The system supports multi-location inventory management, enabling accurate tracking, movement, and replenishment across various warehouses or facilities.

- ✓ Manage supplier information by adding, editing, and deleting details, upload and oversee contracts, and track supplier performance through key metrics.
- ✓ Perform inventory management functions by adding, editing, and deleting items, and implement search and filtering options for inventory data based on various criteria.
- ✓ The system should integrate seamlessly with barcode scanning technology to enhance accuracy in data entry, streamline the tracking process, and facilitate efficient inventory management.
- ✓ The system should support the management of inventory across multiple locations, allowing for accurate tracking of stock levels, movement, and replenishment in various warehouses or facilities.

2) Notification Management

Users can customize notification preferences based on their roles, selecting from options like email, SMS, or in-system notifications. The system enables the definition of triggers for key events within the Revenue Cycle Management (RCM) system, ensuring timely notifications for registration, appointment scheduling, claim submission, payment posting, and denial

notifications. Notifications are delivered in real-time or near real-time, with a comprehensive audit trail to track all communication within the system.

- ✓ Allow users to set their notification preferences based on their role and responsibilities within the healthcare organization.
- ✓ Options should include email notifications, SMS alerts, in-system notifications.
- ✓ Define triggers for various events within the RCM system, such as registration, appointment scheduling, claim submission, payment posting, or denial notifications.
- ✓ Ensure that notifications are delivered in real-time or near real-time to users to keep them informed of important events and updates.
- ✓ Maintain an audit trail of all notifications sent and received within the system.

3) Staff Management

This subsystem facilitates the onboarding of new practices with dedicated staff management systems. It includes functionalities for allocating coordinators and staff during initial setup, granting specific access and privileges, allowing staff to control visibility of their profiles, sending messages to all logged users by selected users, viewing staff activity logs, monitoring Key Performance Indicators (KPIs), and managing staff data (creating, deleting, and updating user profiles and privileges).

- ✓ Enrolling new practices with dedicated staff management systems.
- ✓ Allocating coordinators and staff during the initial setup.
- ✓ Granting specific access and privileges to staff members.
- ✓ Staff ability to see/hide their staff base.
- ✓ Sending messages to all logged users by selected users.
- ✓ Viewing staff activity logs.
- ✓ Viewing KPIs (Key performance Indicators)
- ✓ Managing staff data (Creating Users, Deleting Users, Updating Users, Updating User privileges, Read Users)

4) Appointment Scheduling Management

This comprehensive module manages patient appointments seamlessly. It stores and manages patient information, allowing patients to submit appointment requests. The system verifies patient eligibility based on insurance information and checks healthcare provider availability, displaying available slots, or suggesting alternatives. It supports appointment cancellations, sends timely reminders to patients and providers, and integrates with Electronic Health Records (EHR) for optimized coordination.

- ✓ The system should be able to store and manage patient information, including personal details, contact information, and relevant medical history.
- ✓ The system should allow patients to submit appointment requests.
- ✓ The system must verify the eligibility of patients for appointments based on insurance information.
- ✓ The system should check the availability of the requested healthcare provider and display available appointment slots to the patient.

- ✓ If the requested time is unavailable, the system should suggest alternative appointment slots.
- ✓ The system should support the cancellation of appointments by patients and update the records accordingly.
- ✓ The system must send timely reminders to both patients and healthcare providers leading up to the scheduled appointment.

5) Insurance Management

This module enables users to create, delete, or archive insurance records. It allows updates and modifications to existing insurance records and empowers administrators to define and manage fee schedules for different CPT codes. Users with appropriate permissions can update fee schedules to reflect changes. The system generates insurance records reports and provides analytics features for identifying trends and patterns.

- ✓ The system should allow users to create new insurance records.
- ✓ Users should be able to delete or archive insurance records when necessary.
- ✓ Users should be able to update and modify details of existing insurance records.
- ✓ The system should enable administrators to define and manage fee schedules for different CPT codes.
- ✓ Users with the appropriate permissions should be able to update fee schedules to reflect changes in the allowed amounts.
- ✓ The system should generate insurance records reports and provide analytics features for users to identify trends and patterns.

6) Claim Management

Role-based access control ensures users have access only to relevant functionalities and data. The system tracks the status of submitted claims, integrates with external systems for seamless data exchange, and allows customizable workflows for claim approval. Comprehensive reports on claim status, processing times, and financial performance aid in monitoring and decision-making. Implement role-based access control to ensure that the users only have access to functionalities and data relevant to their roles.

- ✓ Ability to track status of submitted claims, including approvals, rejections, or pending requests for additional information.
- ✓ Functionality to integrate with external systems, such as insurance databases and billing services, to facilitate seamless data exchange and interoperability.
- ✓ Establish customizable workflows for claim approval, allowing organizations to define approval processes based on their specific policies and procedures.
- ✓ Ability to generate comprehensive reports to claim status, processing times and financial performance.

7) Procedure and Diagnosis Management

For Procedure Codes, the system allows adding, updating, and deleting codes with associated details. Users can search, display a list, and associate procedure codes with diagnosis codes. Similar functionalities apply to Diagnosis Codes. This ensures accurate documentation and facilitates efficient billing processes.

Procedure Codes:

- ✓ Add new procedure codes with descriptions, including mandatory fields like code, name, and description.
- ✓ Update existing procedure codes with new descriptions or other relevant information.
- ✓ Delete procedure codes.
- ✓ Search for procedure codes by code, name, keyword, or description.
- ✓ Display a list of all existing procedure codes with their details.
- ✓ Associate procedure codes with diagnosis codes when applicable.

Diagnosis Codes:

- ✓ Add new diagnosis codes with descriptions, including mandatory fields like code, name, and description.
- ✓ Update existing diagnosis codes with new descriptions or other relevant information.
- ✓ Delete diagnosis codes.
- ✓ Search for diagnosis codes by code, name, keyword, or description.
- ✓ Display a list of all existing diagnosis codes with their details.
- ✓ Associate diagnosis codes with procedure codes when applicable.

8) Patient Management

This module involves the registration of new patients, including details verification. Users can search and access patient information, view and update patient records, manage patient records (including handling duplicates), and efficiently store and organize patient details. The system ensures the centralized and secure management of patient records.

- ✓ Register new patients by entering patient details.
- ✓ Verify insurance details.
- ✓ Search and access patient information.
- ✓ View, update patient records as needed.
- ✓ Manage patient records (Delete duplicated records).
- ✓ Store and manage patient records (Details).

Non-functional requirements

- ✓ **Performance:**
  - Response Time: The system should respond to user actions within acceptable time limits, ensuring efficient interaction.

- Scalability: The system should be scalable to accommodate an increasing number of users, patients, and data without significant degradation in performance.
- Throughput: The system should handle a specified number of transactions or operations per unit of time.

✓ **Reliability:**

- Availability: The system should be available for use during specified operational hours, with minimal downtime for maintenance or unexpected issues.
- Fault Tolerance: The system should be resilient to hardware or software failures, ensuring continuous operation and data integrity.
- Backup and Recovery: Regular backups of data should be performed, and a robust recovery mechanism should be in place to restore the system in case of data loss or system failure.

✓ **Security:**

- Data Encryption: All sensitive data, including patient records and financial information, should be encrypted to prevent unauthorized access.
- Access Control: The system should implement role-based access control to restrict access to functionalities and patient data based on user roles.
- Audit Trails: Detailed logs should be maintained to track user activities within the system for security and compliance purposes.

✓ **Scalability:**

- Horizontal Scalability: The system should support the addition of new servers or nodes to the infrastructure to handle increased load.
- Vertical Scalability: The system should efficiently utilize increased resources on a single server to accommodate growing requirements.

✓ **Usability:**

- User Interface Design: The user interface should be intuitive, with a user-friendly design that minimizes the learning curve for new users.
- Accessibility: The system should adhere to accessibility standards, ensuring that users with disabilities can effectively use the software.

✓ **Compatibility:**

- Browser Compatibility: The system should be compatible with a range of browsers to ensure a consistent user experience across different platforms.
- Integration Compatibility: The system should seamlessly integrate with other healthcare systems, such as electronic health records or laboratory information systems.

✓ **Maintainability:**

- Modularity: The system's architecture should be modular, allowing for easy updates, additions, or replacements of components without affecting the entire system.
- Documentation: Comprehensive documentation should be provided for system architecture, code, and configurations to facilitate maintenance and updates.

✓ **Compliance:**

- Regulatory Compliance: The system should comply with relevant healthcare regulations and standards, such as HIPAA, to ensure the privacy and security of patient information.

Technical requirements

Hardware:

The system should have hardware components that meet the required specifications, such as processing power, memory, and storage capacity.

Operating System:

The system should be designed to work with a specific operating system, such as Windows, MacOS, or Linux.

Database:

The system should have a database to store and manage data efficiently.

User Interface:

The system should have an easy-to-use interface that allows users to interact with the system and perform necessary tasks.

Data Security:

The system should have appropriate security measures in place to protect data from unauthorized access, such as encryption and user authentication.

Networking:

The system should be able to connect to a network to enable data transfer and communication with other devices.

Performance:

The system should be designed to perform efficiently and effectively to meet the needs of the users.

Compatibility:

The system should be compatible with other hardware and software components that are required for its operation.

Scalability:

The system should be scalable and able to handle increasing amounts of data and users as the system grows.

Maintenance:

The system should be easy to maintain, update, and troubleshoot to ensure its continued operation and optimal performance.

Literature Review

For the literary review we did research some similar Revenue management-based hospital management services like, DOC 990, E-Channeling and composed with our web application which is intergraded based on the Revenue Cycle Management System (RCM), this review will analyze existing literature on:

- ✓ Key functionalities of online PMS in healthcare, including patient management, staff management, inventory management, pharmacy management, procedure and diagnosis systems, notification management, and insurance management.
- ✓ Benefits of online PMS for RCM, such as improved efficiency, reduced errors, faster claim processing, and increased revenue collection.
- ✓ Challenges associated with implementing online PMS, including data security concerns, integration with existing systems, and user adoption.
- ✓ Best practices for successful implementation and utilization of online PMS for improved RCM.

Key Functionalities:

- **Patient Management:** Online PMS can streamline patient registration, appointment scheduling, electronic health record (EHR) integration, and communication with patients through appointment reminders, billing notifications, and online portals.

- **Staff Management:** PMS can automate scheduling, payroll, and performance tracking for staff, improving efficiency and communication within the practice.

- **Inventory Management**: Real-time inventory tracking helps optimize stock levels, reduce waste, and ensure timely ordering of medical supplies.

- **Pharmacy Management:** Online PMS can integrate with pharmacy systems to automate prescription management, medication renewals, and insurance verification.

- **Procedure and Diagnosis System:** PMS can facilitate accurate coding and billing for procedures and diagnoses, reducing claim denials and improving revenue collection.

- **Notification Management:** Automated appointment reminders, lab results notifications, and billing updates improve patient communication and satisfaction.

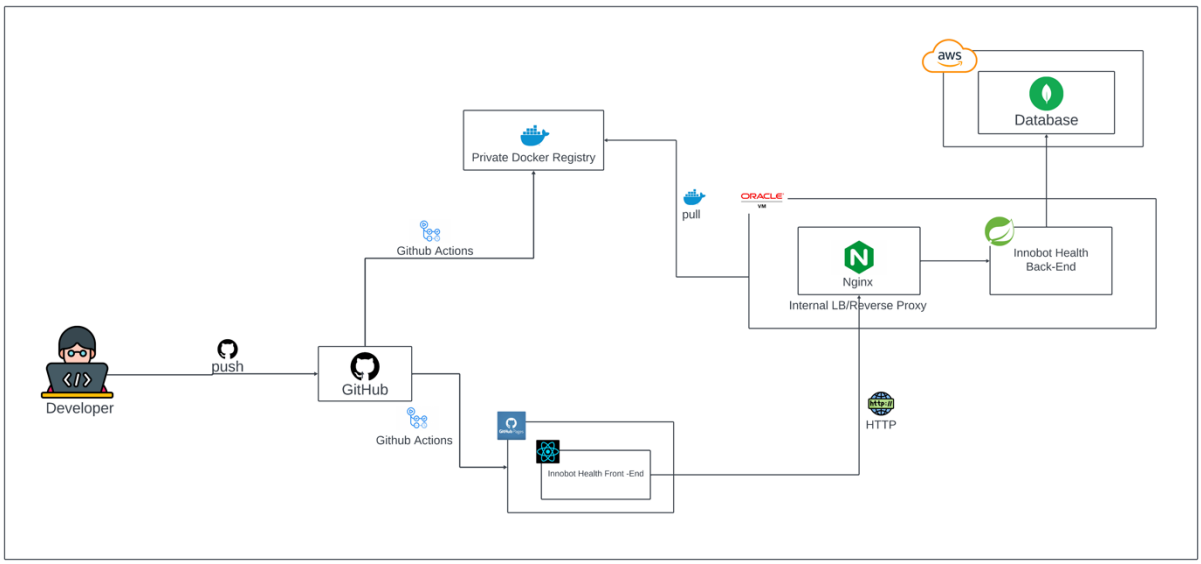
- **Insurance Management**: Online PMS can verify insurance eligibility, submit claims electronically, and track payment status, streamlining the insurance billing process.

Functions	DrChrono	eClinicalWorks	athenaOne	NextGen Office	PatientNow
Inventory Management	✓	✓	✓	✗	✗
Staff Management	✗	✓	✗	✗	✓
Appointment Management	✓	✓	✓	✓	✓
Patient Management	✓	✓	✗	✓	✓
Notification Management	✓	✗	✗	✓	✓
Claim Management	✓	✓	✓	✓	✗
Insurance Management	✓	✗	✓	✓	✗
Procedure and Diagnosis Management	✓	✗	✓	✓	✓

Methodology

Tools and Technologies

Overview of the System Architecture



Tools and Services	Technologies
Git/GitHub (version control)	Spring boot (open-source, micro-service-based Java web framework)
Monday.com (work and project management)	React.js (open-source java script library)
Figma, Balsamiq (UI and wireframes)	Docker (containerizing technology)
IntelliJ Idea (integrated development environment)	MongoDB (An open-source document database that provides high performance and scalability)
Nginx (load balancer/ reverse proxy)	
Oracle Could (cloud computing platform)	
AWS (cloud computing platform)	
Postman (API platform)	
Terminus (SSH platform)	
MongoDB Compass (GUI for MongoDB)	

Spring Boot

Spring Boot is a powerful framework for building Java-based applications with ease. It provides a simplified configuration and setup process, allowing developers to quickly create production-ready applications. With micro service support for auto-configuration, embedded servers, and dependency management, Spring Boot simplifies the development process, enabling developers to focus more on writing business logic rather than boilerplate code. Its convention-over-configuration approach and extensive ecosystem of plugins and starters make it a popular choice for building microservices, web applications, and APIs.

Advantages:

a. Simplified Configuration
b. Spring Boot provides sensible defaults and auto-configuration, reducing the need for manual setup and configuration.
c. Rapid Development
d. Its streamlined development process allows developers to quickly prototype and build applications, saving time and effort.
e. Embedded Servers
f. Spring Boot includes embedded servers like Tomcat, Jetty, or Undertow, simplifying deployment and eliminating the need for external server setup.
g. Dependency Management
h. It manages dependencies effectively, reducing version conflicts and ensuring compatibility between libraries.

Disadvantages:

a. Opinionated
b. While its conventions can speed up development, they may not always align with specific project requirements, leading to limitations or workarounds.
c. Learning Curve
d. Beginners may find Spring Boot's extensive features and configurations overwhelming, requiring time to grasp its concepts fully.
e. Performance Overhead
f. While Spring Boot offers convenience, its auto-configuration and additional layers may introduce some performance overhead compared to lightweight frameworks.

Why did we choose spring boot?

a. Security
b. Spring Boot provides robust security features, essential for handling sensitive health data and complying with HIPAA regulations.
c. Scalability
d. Its support for microservices architecture enables scalability, allowing the system to handle increasing loads and data volumes effectively.
e. Integration
f. With its extensive ecosystem and support for RESTful APIs, Spring Boot facilitates seamless integration with other healthcare systems and third-party services.
g. Reliability
h. Spring Boot's mature ecosystem, community support, and proven track record make it a reliable choice for building mission-critical applications like RCM systems.
i. Productivity
j. By reducing boilerplate code and simplifying configuration, Spring Boot enables developers to focus more on implementing business logic and healthcare-specific functionalities.

React.js

React.js is a JavaScript library for building user interfaces, particularly for single-page applications. Developed by Facebook, it focuses on creating reusable UI components that efficiently update and render when the data changes. Redact's virtual DOM allows for fast and efficient updates to the user interface by minimizing direct manipulation of the actual DOM. With its declarative and component-based architecture, React.js simplifies the process of building interactive and dynamic web applications, making it a popular choice for front-end development.

Advantages:

a. Reusable Components
b. Enhance code reusability and maintainability.
c. Declarative Syntax
d. Simplifies UI development by expressing how the UI should look at any given time.
e. Virtual DOM
f. Boosts performance by reducing actual DOM manipulation and enhancing rendering speed.
g. Component-Based Architecture
h. Facilitates modular development and easier code organization.
i. Strong Ecosystem
j. Access to a vast ecosystem of libraries, tools, and community support.

Disadvantages:

a. Learning Curve
b. Beginners may find its concepts, such as JSX and component lifecycle, challenging initially.
c. Tooling Complexity
d. Setting up a React project with additional tools like Babel and Webpack can be complex.
e. SEO Challenges
f. Single-page applications built with React may face SEO challenges due to initial server-side rendering requirements.

Why did we choose React.js?

a. Enhanced User Experience
b. Redact's efficient rendering and dynamic UI capabilities improve user interaction, vital for a smooth RCM system.
c. Scalability
d. Component-based architecture allows for easy scaling and maintenance of the application as the RCM system grows.
e. Community Support
f. A large and active community provides resources, best practices, and support, ensuring the success of the project.
g. Integration
h. React seamlessly integrates with other JavaScript libraries and frameworks, facilitating integration with existing healthcare systems and APIs.
i. Performance
j. Virtual DOM and efficient rendering contribute to improved performance, crucial for handling large datasets and complex interactions in healthcare applications.

Docker

Docker is a containerization platform that simplifies the process of building, deploying, and managing applications in isolated environments called containers. With Docker, applications and their dependencies can be packaged into a single unit, ensuring consistency across different environments.

Advantages:

a. Portability: Docker containers can run on any platform that supports Docker, providing consistent behavior across development, testing, and production environments.
b. Isolation: Containers isolate applications and their dependencies, preventing conflicts and ensuring reliability.
c. Efficiency: Docker's lightweight containers consume minimal resources and start quickly, improving efficiency and scalability.
d. Consistency: Docker eliminates the "it works on my machine" problem by ensuring that applications run consistently in any environment.
e. Scalability: Docker's container-based architecture allows applications to scale up or down easily, adapting to changing demands.
f. DevOps Integration: Docker integrates seamlessly with DevOps practices, enabling continuous integration, delivery, and deployment pipelines.

Disadvantages:

a. Complexity: Docker introduces additional complexity, especially for teams new to containerization, requiring time to learn and adapt to new concepts and tools.
b. Orchestration Overhead: Managing containerized applications at scale may require additional tools and overhead for orchestration, such as Kubernetes or Docker Swarm.
c. Security Concerns: Improperly configured Docker containers can pose security risks, requiring careful attention to best practices for securing containerized environments.

Why we choose docker?

a. Consistency
b. Docker ensures consistent deployment and runtime environments, critical for maintaining compliance and reliability in healthcare systems.
c. Isolation
d. Containerization isolates applications and dependencies, reducing the risk of conflicts and ensuring data privacy and security.
e. Scalability
f. Docker's container-based architecture allows the RCM system to scale efficiently to handle increasing workloads and data volumes.
g. Flexibility
h. Docker's portability enables easy deployment across different environments, including on-premises servers and cloud platforms, providing flexibility in infrastructure choices.

i. DevOps Alignment

j. Docker aligns well with DevOps practices, facilitating automation, continuous integration, and deployment workflows, improving agility and collaboration in RCM system development and operations.

MongoDB

MongoDB is a NoSQL database management system that offers flexibility, scalability, and performance for modern application development. It uses a document-oriented data model, storing data in flexible JSON-like documents, making it suitable for a wide range of use cases, including real-time analytics, content management, and mobile applications.

Advantages:

a. Flexible Data Model
b. MongoDB's document-oriented approach allows for dynamic and schema-less data structures, accommodating evolving application requirements.
c. Scalability
d. MongoDB's distributed architecture supports horizontal scaling across multiple servers, enabling seamless scaling as data volume and user load grow.
e. Performance
f. With features like indexing, sharding, and in-memory storage engine, MongoDB delivers high-performance queries and data retrieval operations.
g. Replication and High Availability
h. MongoDB provides automatic replication and failover capabilities, ensuring data availability and reliability in case of server failures.
i. Rich Query Language
j. MongoDB Query Language (MQL) supports a wide range of query operations, including CRUD operations, aggregation, text search, and geospatial queries.
k. Community Support
l. MongoDB has a vibrant community of users, contributors, and experts, providing resources, documentation, and support for developers.

Disadvantages:

a. Data Consistency
b. MongoDB sacrifices some aspects of data consistency in favor of scalability and performance, requiring careful consideration of application requirements and data integrity.
c. Learning Curve
d. Developers familiar with relational databases may face a learning curve when adapting to MongoDB's document-oriented data model and query language.

e. Operational Complexity: Managing a distributed MongoDB cluster with multiple nodes requires expertise in deployment, configuration, monitoring, and maintenance, adding complexity to operations.

Why we choose MongoDB?

a. Flexible Data Model
b. MongoDB's schema-less design accommodates the complex and evolving data structures often encountered in healthcare systems, such as patient records and medical histories.
c. Scalability
d. MongoDB's horizontal scaling capabilities allow the RCM system to handle growing data volumes and user loads without sacrificing performance.
e. Performance
f. MongoDB's indexing, sharding, and in-memory storage engine ensure fast query response times, critical for real-time data processing and analytics in healthcare.
g. High Availability
h. MongoDB's replication and failover features ensure data availability and reliability, minimizing downtime and ensuring continuous operation of the RCM system.
i. Community Support
j. MongoDB's active community provides resources, best practices, and support, aiding developers in building and maintaining the RCM system effectively.

Requirements Engineering Methods

1. Use Case Modeling
2. Describe system interactions and behaviors from the perspective of users.
3. User Stories
4. Define system features and functionalities from the end user's viewpoint.
5. Requirement Workshops
6. Gather stakeholders to elicit and prioritize system requirements collaboratively.

Design Methods

1. Domain-Driven Design (DDD)
2. Focus on modeling the healthcare domain and its complexities to design a relevant system architecture.
3. Component-Based Design

4. Break down the system into modular components for easier development and maintenance.
5. Responsive Web Design
6. Ensure the RCM system's user interface is accessible and usable across different devices and screen sizes.

Testing Methods

1. Unit Testing
2. Test individual components and functions to ensure they work as expected.
3. Integration Testing
4. Verify the interaction and integration between different modules and components.
5. End-to-End Testing
6. Test the entire system workflow to ensure all components work together seamlessly.

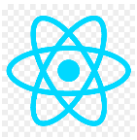
Integration Methods

1. RESTful APIs
2. Enable seamless integration with external systems and services.
3. Message Queues (RabbitMQ – Not included in the first stage)
4. Facilitate asynchronous communication and integration between different parts of the system.

Tools and Technologies

We use these technologies for developing our web application. We use a combination of 4 technologies: MongoDB, Spring boot, React JS.

MongoDB is a document database used to build highly available and scalable internet applications. With its flexible schema approach, it's popular with development teams using agile methodologies [13].



React is a declarative, efficient, and flexible JavaScript library for building user interfaces [15].

Spring Boot is a powerful framework for building Java-based applications with minimal setup and configuration. It provides a convention-over-configuration

Apart from MERN Stack we use the following tools and technologies in our project.

Visual Studio Code is a streamlined code editor with support for development operations like debugging, task running, and version control.

GitHub is a code hosting platform for version control and collaboration. It lets us to work together on projects from anywhere.

Project plan (Gantt chart)

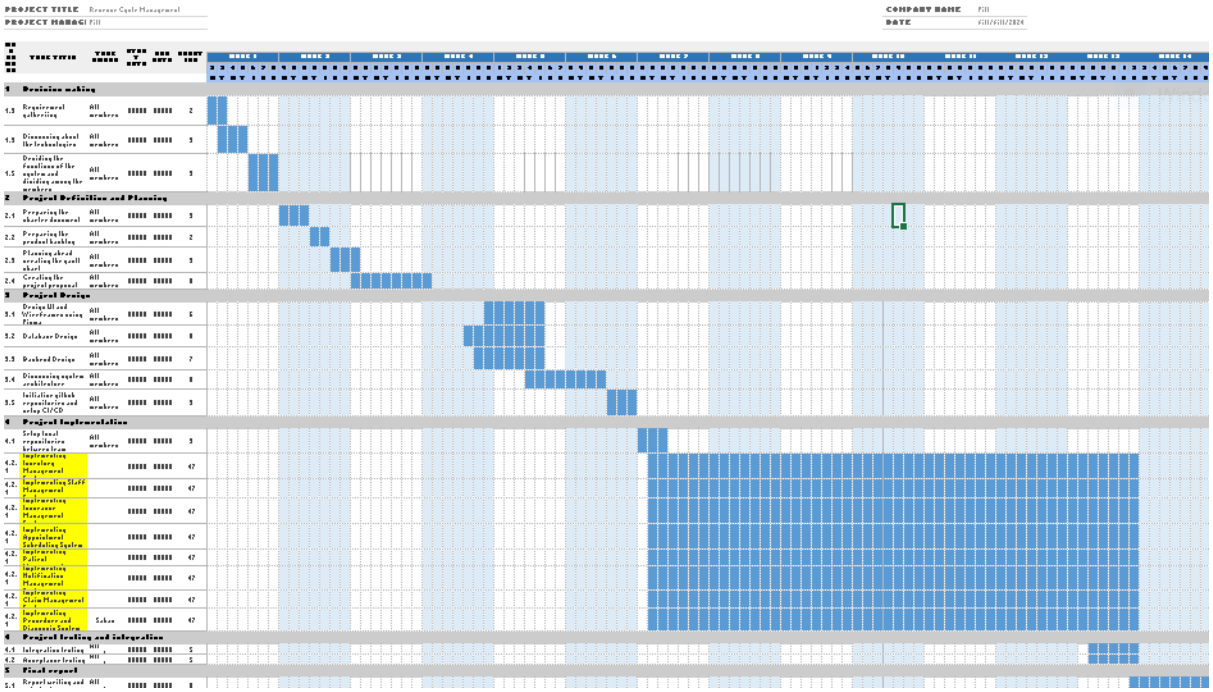


Figure 2 – Gantt Chart

We started gathering details even before the semester started and it took 2 weeks from the semester starting date to complete our requirement analysis and documentation phase. We were able to gather all the necessary information to be able to complete our requirements without any problems.

The second week was spent planning the project and within the third week we started to work on the UI design of the page. We have achieved a great result so far and are looking forward to the next step in the project.

During the fourth week we have decided to work on the database design phase for about three weeks.

Since we've been working on the database design phase, we've decided to add more features to it. We've made progress on the first two levels, but we need to add more features to the second layer. This will make the database more complicated, but it's important work that we need to continue.

Meanwhile, coding within the fifth week begins, and will continue for the next six weeks. This will allow you to complete the project by the end of the seventh week.

We are excited to begin our coding within the ninth week and developing with the hope of covering all aspects of the project. We are looking forward to learning more about coding, as well as the various development tools that we will be using.

Starting on the twelfth week after the testing phase, the web application will be launched. The application will be available during the eleven-week period following the phase.

Work breakdown structure (work distribution)

	Student ID and Name with initials	Tasks
	S.A.N. Bamunusinghe	**Inventory Management System** o Monitor and update real-time information on the quantity and status of medical supplies, equipment, and pharmaceuticals within the hospital. Entirely managed by Innobot. o Maintain a database of suppliers, manage contracts, and track supplier performance to ensure a reliable source of quality medical products. o Implement barcode technology to efficiently scan and label items, aiding in accurate data entry, reducing errors, and streamlining the tracking process. o Can Add, Delete, update, and read a Inventory data as well.
	S A T Nayanapriya	**Staff Management System** o For every practice enrolled it will get a separate staff management system. Staff management system is handled by the practice itself. In the initial setup practice will allocate the coordinators and the staff members. Coordinators will have the privileges to enroll new staff and new coordinators as well. Coordinators/Staff could also be removed. o Staff could be granted specific access and specific privileges to perform specific functions in the system. (Creating Insurances, creating claims, creating patients) o The privileges could allow a staff member to see/hide their Staff base. o sending messages to all logged users. Sending messages to notify all staff members is also only allowed for selected users. o Can see each staff members activity within the system. o Can add, delete, update, and read a staff data as well.
	Obeyesekere A D	**Appointment management System** ▪ Monitor and manage the stock levels of pharmaceuticals and medical supplies, ensuring accurate records of medications available in the pharmacy. Including details such as drug name, dosage, manufacturer, expiration date, and any special ▪ Manage and request drug and supplies via mail to the registered drug stores.

		○ Handle prescription details, track dispensing activities, and manage medication refills, ensuring adherence to regulatory requirements and patient safety. ○ Generate reports on medication usage, expiration dates, regulatory compliance, and other key metrics to support decision-making, regulatory adherence, and audit requirements
	W A Malsha Haren	**Patient Management System** ○ The system focuses on adding the patients to the system. Patients could be added by the staff members and updated and deleted as necessary. ○ Patients will have their Claims and Insurance managed by their relevant systems. Adding a patient is done by filling a form and patients cannot login to the system or perform anything in the system. ○ Patients will only be there as a record and not a user. ○ Patients has their Name, DOB, Address and Accnum, MRN num, Subscriber IDs etc..
	DME Wimalagunasekara	**Notification Management System** ○ User should be able to send notifications for practices, staff members, coordinators, patients, and all other affiliated parties in the system. Notifications will be directed via three channels Gmail, SMS and system notifications based on the notification preference of the users updated in user registration. ○ Users should be able to view, acknowledge and reply to the notifications received by clicking on the notification icon in the header. ○ The System focuses on adding notifications, sending notifications when a staff member allocated/mentioned on a place. ○ Notifications will be also sent out to relevant personnel when sensitive data is accessed via System notifications, Emails and SMS. ○ Notification management is accessed by user base and staff base alike. ○ Sent notifications can be accessed to view
	M F M Farsith	**Claim Management System** ○ The system focuses on Adding, Deleting, Updating and Reading claims in the system. ○ A claim will be sent out the insurances to get the payments for the practices. ○ Claim will contain the procedures, Diagnosis and the patient details attached. ○ Claim status is also can be updated as pending, denied, etc. with a note attached. ○ When Claims get payments, it should be posted. Those payments will be mentioned with where the payment came from. (Patient payment, Insurance payment). ○ Claim balances can be written off as well. Payments should be posted to CPT level of a claim. Completed Our Web Application part in

		Background section and Aims and Objective section of proposal report.
	Chamikara M G S	**Insurance Management System** ○ The system focuses on managing all the Insurances. ○ Insurances will have respective details and respective claims attached. ○ Insurances can be created, Deleted, read, and updated. ○ Insurances will have their fee schedules which shows the allowed amounts for a relevant CPT code. ○ Fee schedules can be updated. ○ Insurances will be linked with their respective portals for ease access for the staff.
	D S I Gamage	**Procedure and Diagnosis System** ○ The system will contain the diagnosis codes (the reason for the procedure) and the procedure code. ○ Procedure codes can be added/updated/deleted and read to get a more descriptive on what the procedure code provides. ○ When adding a procedure code, the description of the code should also be mentioned. ○ When adding a diagnosis code, the description of the code should also be mentioned. ○ New Procedures and diagnosis are added by only privileged users.

6. Evaluation criteria

6.1 Compliance and Security Audits:

Conduct regular audits to ensure the system complies with healthcare regulations and industry standards.
Evaluate the effectiveness of security measures to protect sensitive patient data and maintain data integrity.

6.2 Key Performance Indicators (KPIs):

Efficiency Metrics: Measure the system's impact on reducing the time spent on administrative tasks.
Accuracy Metrics: Assess the system's accuracy in automating processes compared to manual execution.
Productivity Metrics: Evaluate the increase in productivity and the ability of staff to focus on high-value activities.

6.3 User Satisfaction Surveys:

Conduct regular surveys among end-users, including healthcare administrators and staff, to gather feedback on the system's usability, effectiveness, and overall satisfaction.
Evaluate user feedback on the system's ability to streamline workflows, reduce workload, and enhance their experience.

Appendix

Figure 1 - System Diagram

We divided our system into 8 main subsystems. These are Appointment Management System, Staff Management Sytem, Claim Management System,

Inventory Management System, Procedure/ DX management System, Notification Management System, Patient Management System, Insurance Management System.

Figure 2 - Overview of the System Architecture

Our system architecture is modular and efficient, comprising components like Appointment, Staff, Claim, Inventory, Procedure/DX, Notification, Patient, and Insurance Management. This design ensures seamless interactions for optimal system performance.

Figure 3 - Gantt Chart

The Gantt chart visually represents our project timeline, outlining tasks and their respective timelines. It provides a clear roadmap for the project, showing task dependencies and progress, aiding in efficient project management and delivery.

B. Literary Terms

Revenue Cycle Management (RCM): The financial process that manages the administrative and clinical functions associated with claims processing, payment, and revenue generation.

Practice Management System (PMS): A software solution designed to streamline and optimize the day-to-day operations of healthcare practices, including scheduling, billing, and reporting.

Procedure Codes (CPT): Standardized codes used to describe specific medical procedures and services provided by healthcare professionals.

References

Backend:

- ✓ (Spring Boot Reference Documentation for) https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/

- ✓ (Maven Repository)
  https://mvnrepository.com/repos/central
- ✓ Describing the UI using React- https://react.dev/learn/describing-the-ui
- ✓ Database References –
  https://www.mongodb.com/docs/atlas/
  https://www.mongodb.com/docs/tools-and-connectors/
- ✓ Agile Methodologies Tools-
  https://www.atlassian.com/agile/tutorials
- ✓ Axios–
  https://www.geeksforgeeks.org/axios-in-react-a-guide-for-beginners/
  https://www.digitalocean.com/community/tutorials/react-axios-react
- ✓ UI/UX designing - https://www.mygreatlearning.com/academy/learn-for-free/courses/ui-ux
- ✓ Frontend development –
  https://tailwindcss.com/
  https://react-icons.github.io/react-icons/
  https://getbootstrap.com/
  https://getbootstrap.com/