TITLE OF THE PROJECT

Patients Management System

Submitted By

Sharmin Akter
ID:(191-16-422)

Submitted To

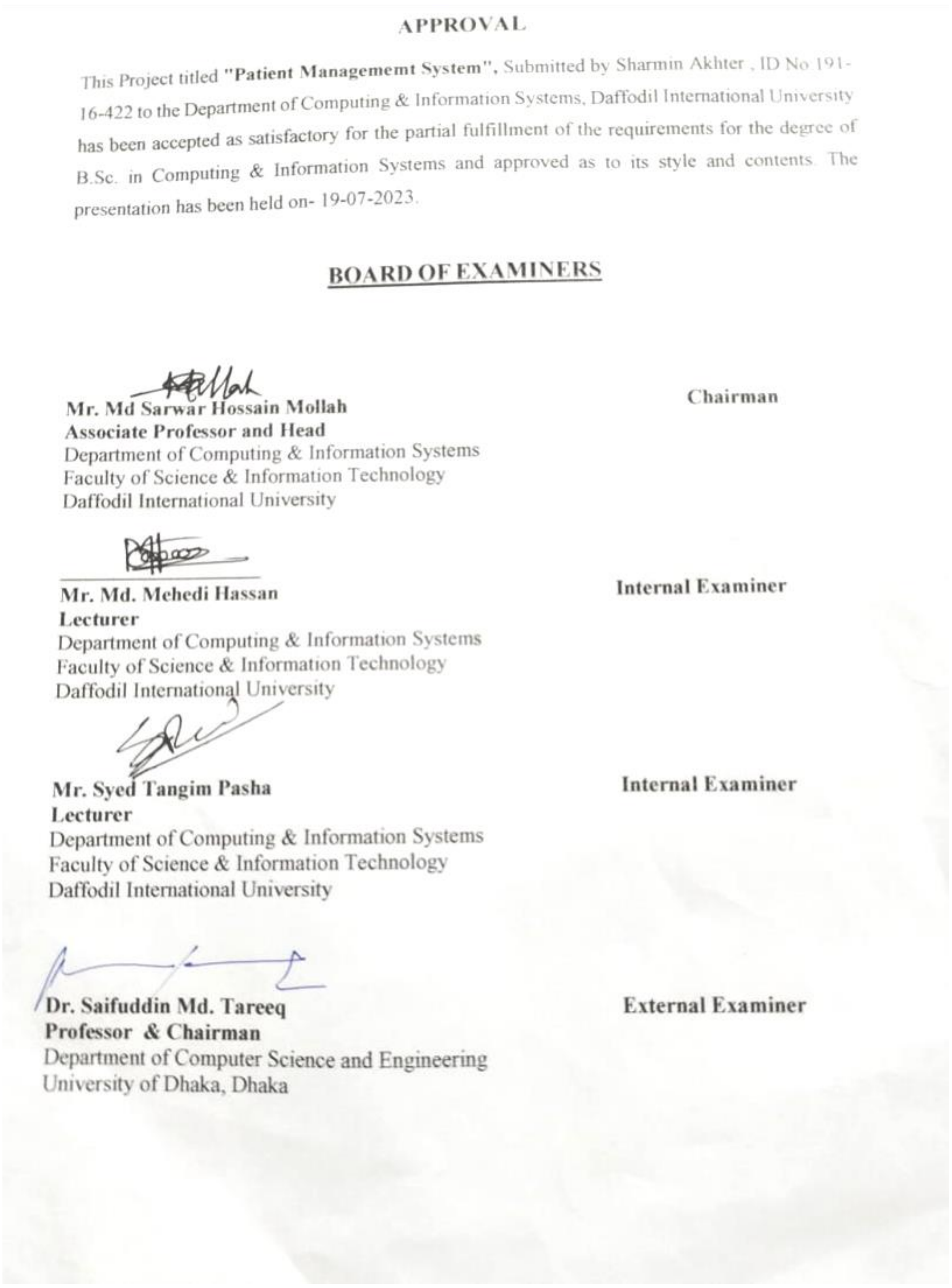
Mr. Syed Tangim Pasha

Lecturer,Department of Computing and Information System(DIU)

Submission Date:19-07-2023

Department of Computing and Information System (CIS)

DAFFODIL INTERNATIONAL UNIVERSITY



APPROVAL

This Project titled **"Patient Managememt System",** Submitted by Sharmin Akhter , ID No 191-16-422 to the Department of Computing & Information Systems, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computing & Information Systems and approved as to its style and contents. The presentation has been held on- 19-07-2023.

Declaration

I hereby declare that; this project has been done by me under supervision of Mr.Syed Tangim Pasha,Lecturer department of Computing and Information System (CIS) of Daffodil International University. I am also declaring that this project or any part of there has never been submitted anywhere else for the award of any educational degree like, B.Sc., M.Sc., Diploma or other qualifications.

Supervised By

[illegible]

Mr.Syed Tangim Pasha
Lecturer
Department of CIS
Daffodil International University

Submitted By

Sharmin

Name: Sharmin Akhter
ID: 191-16-422
Department of CIS
Daffodil International University

Acknowledgement

I want to praise my awesome Allah first. After that, I want to thank my parents and my dear sister for their support. I am unable to finish this course without their assistance. Then, I must express my sincere gratitude to my course supervisor Md Sayed tangim Pasha and other teachers, for providing me with excellent support from the project's inception.

Dedication

I would like to dedicate this project to my parents and my dear sister. since it is my first final academic project.With out them I am fully incomplete.

Executive summary

This is a responsive patient management system. It is complete solution for doctor to manage their patient appointments, patient details and also manage billing of particular patient.

Table of Contents

Table of figure

abstract

The Patient Management System (Medenico) is a software application designed to facilitate the efficient management of patient-related information and processes within healthcare facilities. It aims to streamline administrative tasks, improve patient care, and enhance overall operational efficiency.

The system offers various features, including patient registration and profile management. Healthcare providers can easily register new patients and maintain detailed profiles that include personal information, medical history, contact details, and insurance information.

Appointment scheduling and reminders are also key functionalities of the system. Healthcare providers can schedule appointments for patients, manage the availability of healthcare professionals and resources, and send automated reminders to patients to reduce no-shows and improve scheduling efficiency.

Another critical aspect of the Patient Management System is electronic medical records (EMR) management. The system provides a centralized repository for storing and managing electronic medical records, allowing healthcare providers to record diagnoses, treatments, medications, lab results, and progress notes. This ensures easy access to patient information across different departments, facilitating coordinated and comprehensive care.

Billing and invoicing functionalities streamline financial processes within healthcare facilities. The system enables the capture of services rendered, generates invoices, and tracks payments. It may also integrate with financial systems and insurance providers for efficient claims processing and reimbursement.

Reporting and analytics capabilities provide healthcare providers with valuable insights into patient demographics, appointment statistics, revenue, and other performance indicators. Customizable reports and dashboards allow for data-driven decision-making and performance monitoring.

Overall, the Patient Management System improves the efficiency and effectiveness of patient management within healthcare facilities. It enhances patient care, optimizes administrative processes, and ensures accurate and accessible patient information, ultimately leading to better healthcare outcomes.

Chapter 1 - Introduction

1.1 Introduction

The Patient Management System (Medenico) is a comprehensive software solution designed to streamline and enhance the management of patient-related information and processes within healthcare facilities. It is a vital tool that enables healthcare providers to efficiently handle various tasks, from patient registration to appointment scheduling, medical record management, billing, and reporting.

The primary objective of the Patient Management System is to improve the overall efficiency, accuracy, and quality of patient care while reducing administrative burdens. By automating manual processes and centralizing patient data, healthcare providers can focus more on delivering effective treatments and personalized care to their patients.

The system offers a user-friendly interface that allows healthcare providers to register new patients, maintain up-to-date patient profiles, and capture essential demographic information. This ensures that accurate patient records are available for immediate access, enabling quick and informed decision-making.

Appointment scheduling and management is a critical aspect of the Patient Management System. Healthcare providers can easily schedule appointments, assign healthcare professionals, and allocate necessary resources based on availability. Automated reminders are sent to patients to reduce no-shows, optimize clinic scheduling, and ensure that patients receive timely care.

The electronic medical records (EMR) management feature of the system provides a secure and centralized repository for storing and accessing patient medical records. Healthcare providers can record diagnoses, treatments, medications, lab results, and progress notes in the EMR, enabling comprehensive and coordinated care across different departments.

Billing and insurance management functionalities streamline the financial aspects of patient management. The system facilitates accurate billing by capturing services rendered, generating invoices, and managing insurance claims. This reduces administrative burdens and ensures efficient reimbursement processes.
Reporting and analytics capabilities enable healthcare providers to gain valuable insights into various aspects of patient management, including appointment statistics, patient demographics, treatment outcomes, and financial performance. Customizable reports and dashboards help healthcare administrators monitor key performance indicators and make data-driven decisions to improve operational efficiency.

In summary, the Patient Management System is a comprehensive solution that improves patient care, enhances administrative processes, and enables better decision-making within healthcare facilities. By automating tasks, centralizing patient data, and providing insightful analytics, the system empowers healthcare providers to deliver high-quality care and optimize their operation.

1.2 Document Contents in Project Document

The following chapters will be covered in this publication or documentation to chronicle the project's progress.

Chapter 1: Introduction

A brief introduction to the suggested project and system.

Chapter 2: Initial Phase

This chapter covers the preliminary research findings for the proposed system, such as the major goals and objectives, issue area, possible solutions, and project history.

Chapter 3: Literature Review

The issue domain, solutions, evaluation of current solutions, and ultimately suggestion are all discussed in detail in this chapter.

Chapter 4: Methodology

The importance of employing methodology, many methods that may be employed, and the preferred methodology and its application will all be explored in this section.

Chapter 5: Planning

This chapter covers project plans such as project plans, test plans, risk and change management, and so on.

Chapter 6: Feasibility

This is where you'll find the full feasibility study report and cost-benefit analysis.

Chapter 7: Foundation

This chapter will contain information on the issue area identification, general need list, proposed technologies, and reasons.

Chapter 8: Exploration

It includes basic UML diagrams and a need list for both the existing and new systems, as well as a prototype.

Chapter 9: Engineering

This chapter contains the proposed system's logical and behavioral models.

Chapter 10: Deployment

Here, we'll talk about coding samples and how to split down a development challenge based on development priority.

Chapter 11: Testing

This chapter includes a number of test ideas and outcomes.

Chapter 12: Implementation

This section covers the implementation strategy, training model, and other related topics.

Chapter 13: Critical Appraisal and Evaluation

The review of the initial objectives that were reached and those that were not met in great detail.

Chapter 14: Lessons Learned
The learnings and obstacles encountered during the project are mostly included in the pre-project-
closing evaluation.
Chapter 15: Conclusion
Here you will find a summary of the project, as well as its goals, successes, and lessons learned.

Chapter 2 – Initial Study

2.1 Project Proposal

The Patient Management System (Medenico) project aims to develop and implement a comprehensive software solution for effective management of patient-related information and processes within healthcare facilities. The Medenico will streamline administrative tasks, improve patient care, and enhance overall operational efficiency.

Objectives:

Digitize Patient Records: The Medenico will transition from paper-based patient records to a digital system, enabling easy access, storage, and retrieval of patient information.

Streamline Appointment Management: The system will facilitate efficient appointment scheduling, resource allocation, and automated reminders to optimize clinic workflow and reduce waiting times.

Enhance Medical Records Management: The Medenico will provide a centralized electronic medical records (EMR) system, ensuring accurate and up-to-date patient health records for improved diagnosis and treatment.

Automate Billing and Invoicing: The system will automate billing processes, including capturing services rendered, generating invoices, and managing insurance claims, to streamline financial transactions.

Enable Reporting and Analytics: The Medenico will offer comprehensive reporting and analytics capabilities, allowing healthcare administrators to monitor key performance indicators, track patient demographics, and make data-driven decisions.

Implementation Plan:

System Development: The project will involve the development of a custom Medenico software tailored to the specific needs of healthcare facilities. This will include modules for patient registration, appointment scheduling, medical records management, billing, and reporting.

Data Migration and Integration: Existing patient data will be migrated to the new system, ensuring a seamless transition from paper-based to digital records. Integration with existing healthcare infrastructure and systems will be considered for data exchange and interoperability.

User Training: Healthcare staff will receive comprehensive training on using the Medenico effectively, ensuring smooth adoption and utilization of the system's features and functionalities.

Pilot Testing and Feedback: A pilot phase will be conducted to test the system's functionality, usability, and performance. Feedback from users, including healthcare providers and staff, will be collected and incorporated into system refinements.

Rollout and Support: The finalized Medenico will be deployed across healthcare facilities, with ongoing technical support and maintenance provided to ensure continuous system functionality.

Benefits:

Improved Patient Care: The Medenico will enable healthcare providers to access accurate and comprehensive patient information, leading to better diagnosis, treatment planning, and continuity of care.

Enhanced Efficiency: Streamlined processes, automated appointment scheduling, and billing capabilities will reduce administrative burdens, allowing healthcare staff to focus more on patient care.
Data-driven Decision Making: The reporting and analytics capabilities of the PMS will provide valuable insights into patient demographics, healthcare trends, and performance indicators, empowering administrators to make informed decisions.
Cost Savings: By automating manual tasks, reducing paperwork, and optimizing resource utilization, the Medenico will contribute to cost savings for healthcare facilities.

The implementation of the Patient Management System will revolutionize patient care and administrative processes within healthcare facilities. By digitizing patient records, streamlining appointment management, enhancing medical records handling, and automating billing and reporting, the Medenico will improve operational efficiency and provide better healthcare services for patients.

Background Study
The healthcare industry is constantly evolving, with a growing emphasis on providing efficient and patient-centered care. Patient Management Systems (Medenico) have emerged as vital tools for healthcare facilities to streamline administrative processes, enhance patient care, and improve overall operational efficiency.

Paper-based Challenges:
Traditionally, healthcare facilities relied on paper-based records and manual processes for managing patient information. This approach posed several challenges, including:
Difficulty in accessing and retrieving patient records, leading to delays in care delivery.
Errors and inaccuracies in record-keeping due to manual data entry.
Inefficient appointment scheduling, resulting in long waiting times and patient dissatisfaction.
Manual billing and invoicing processes, leading to increased administrative workload and potential billing errors.
Digital Transformation in Healthcare:
The rapid advancement of technology has revolutionized the healthcare industry, paving the way for digital transformation. Patient Management Systems leverage technology to overcome the limitations of paper-based systems and provide numerous benefits:
Centralized Electronic Medical Records (EMR): Medenico enables the creation and management of comprehensive electronic medical records, ensuring easy access, accurate documentation, and improved patient safety.
Efficient Appointment Management: Medenico facilitates online appointment scheduling, automated reminders, and real-time availability of resources, reducing waiting times and optimizing clinic workflow.

Streamlined Billing and Insurance Processing: Medenico automates billing processes, capturing services rendered, generating invoices, and managing insurance claims, resulting in improved accuracy, faster reimbursement, and reduced administrative workload.
Reporting and Analytics: Medenico provides robust reporting and analytics capabilities, enabling healthcare administrators to monitor key performance indicators, track patient demographics, analyze trends, and make data-driven decisions for process improvement.

Bangladesh Healthcare Perspective:
In Bangladesh, the healthcare system faces numerous challenges, including a large population, limited resources, and outdated infrastructure. Implementing a Patient Management System can address these challenges and bring significant improvements:
Accessible and Efficient Patient Care: Digitizing patient records and streamlining processes will facilitate quick and accurate access to patient information, enabling healthcare providers to make informed decisions and provide timely and personalized care.
Enhanced Operational Efficiency: Automating administrative tasks such as appointment scheduling, billing, and insurance processing will reduce paperwork, minimize errors, and optimize resource utilization, resulting in improved efficiency and cost savings.
Data Security and Privacy: Implementing robust security measures in the Medenico will ensure the confidentiality and integrity of patient data, adhering to privacy regulations and safeguarding sensitive information.
Scalability and Integration: The Medenico should be designed to accommodate the specific needs and scalability requirements of healthcare facilities in Bangladesh. Integration with existing infrastructure, such as laboratory systems and pharmacy management, will streamline workflows and improve overall coordination.
Future Opportunities:
Implementing a Patient Management System sets the stage for further advancements in healthcare technology. It lays the foundation for telemedicine, remote patient monitoring, and interoperability with other systems, enabling seamless data exchange and collaborative care.
In conclusion, the adoption of a Patient Management System in Bangladesh's healthcare facilities holds great potential for improving patient care, enhancing operational efficiency, and transforming the healthcare landscape. By transitioning from paper-based systems to digital platforms, healthcare providers can overcome traditional challenges, leverage technology to their advantage, and deliver better healthcare services to patients.

Description of the proposed system
The proposed Patient Management System (Medenico) is a comprehensive software solution designed to streamline the management of patient-related information and processes within healthcare facilities. It aims to enhance operational efficiency, improve patient care, and

facilitate better decision-making for healthcare providers. The system will be tailored to the specific needs and requirements of healthcare facilities in Bangladesh.

Key Features of the Proposed System:

Patient Registration and Demographics:
The Medenico will provide a user-friendly interface for capturing and managing patient information, including personal details, medical history, and contact information. It will support efficient patient registration, ensuring accurate and up-to-date data entry.

Electronic Medical Records (EMR):
The system will enable the creation, storage, and retrieval of electronic medical records, replacing the traditional paper-based system. EMR functionality will include recording patient visits, documenting diagnosis, treatments, medications, and test results, and maintaining a complete medical history for each patient.

Appointment Scheduling:
The PMS will offer an advanced appointment scheduling module, allowing healthcare providers to manage and optimize their daily schedules. It will enable patients to book appointments online, view available time slots, and receive automated reminders via email or SMS.

Billing and Insurance Management:
The system will automate billing processes, generating accurate invoices based on services rendered. It will integrate with insurance systems to streamline claims processing, reducing administrative burdens and improving financial efficiency.

Reporting and Analytics:
The Medenico will provide comprehensive reporting and analytics capabilities, allowing healthcare administrators to generate various reports and analyze key performance indicators. Reports can include patient demographics, appointment statistics, revenue analysis, and other relevant metrics, enabling data-driven decision-making.

Security and Data Privacy:
The proposed system will prioritize the security and privacy of patient data. It will implement robust security measures, including access controls, encryption, and audit trails, to ensure the confidentiality and integrity of patient information.

Integration and Interoperability:

The Medenico will be designed to seamlessly integrate with existing healthcare infrastructure, such as laboratory systems, pharmacy management, and imaging systems. This interoperability will enable the exchange of data, improve workflow efficiency, and facilitate coordinated care.

User-Friendly Interface and Training:
The system will feature an intuitive and user-friendly interface, making it easy for healthcare providers and staff to navigate and utilize its functionalities. Additionally, comprehensive user training will be provided to ensure effective adoption and utilization of the system.

In conclusion, the proposed Patient Management System will revolutionize patient care and administrative processes within healthcare facilities. By digitizing patient records, streamlining appointment management, automating billing and insurance processes, and providing robust reporting and analytics capabilities, the system will improve operational efficiency, enhance patient care, and support informed decision-making for healthcare providers.

Prioritized Features according to MoSCoW

MoSCoW is a requirement prioritization approach used during the requirement analysis stage. It helps to understand the significance of the criterion.

Serial No.	**Requirement for Medenico**	**Priority of Medenico**
01.	Patient Registration	Must-have
02.	Appointment Scheduling	Must-have
03.	Medical Records Management	Must-have
04.	Billing and Payment	Must-have
05.	Patient Check-In/Check-Out	Should-have

06.	Electronic Prescriptions	Should-have
07.	Clinical Decision Support	Should-have
08.	Reporting and Analytics	Should-have
09.	Telemedicine/Remote Consultation	Could-have
10.	Patient Portal	Could-have
11.	Integration with External Systems	Could-have
12.	Advanced Artificial Intelligence	Won't-have
13.	IoT Integration	Won't-have

Table 1: Prioritized Features

Exploration & Engineering: The final criteria are based on the Prioritized Requirements List (PRL).:

Exploration & Engineering Patient Management System refers to the process of researching, designing, and developing a comprehensive software solution for managing patient information and healthcareworkflows. It involves the exploration of user requirements, identification of technical challenges,engineering a system that meets the specific needs of healthcare organizations.

During the exploration phase, the project team conducts extensive research to gather insights into the existing patient management processes and identify pain points and areas for improvement. This involves engaging with stakeholders, such as healthcare providers, administrators, and patients, to understand their requirements, workflows, and expectations.
Once the exploration phase is complete, the engineering phase begins, focusing on the design and development of the patient management system. This typically involves several stages:

System Design: The project team creates a comprehensive design for the patient management system, considering factors such as user experience, system architecture, data security, and integration with existing healthcare systems.

Development: The software development process involves writing code, implementing the design, and creating the necessary functionality for patient registration, appointment scheduling, medical records management, billing, and other features identified during the exploration phase. Agile development methodologies are often employed to ensure iterative and collaborative development.

Testing and Quality Assurance: Rigorous testing is performed to identify and fix any bugs or issues in the system. This includes functional testing, performance testing, security testing, and usability testing to ensure the system operates as expected and meets industry standards.

Deployment and Implementation: Once the system has been thoroughly tested and validated, it is deployed in a production environment. This involves setting up the necessary hardware infrastructure, installing the software, and configuring the system to work seamlessly within the healthcare organization's existing IT infrastructure.

Training and Adoption: To ensure successful adoption of the patient management system, healthcare professionals and staff members are provided with comprehensive training on how to use the system effectively. User manuals, documentation, and ongoing support are also provided to assist users during the transition.

Continuous Improvement: After the initial implementation, the project team continues to gather feedback, monitor system performance, and make necessary enhancements and updates to improve the system's functionality and address any emerging needs or issues.

The Exploration & Engineering Patient Management System process aims to deliver an efficient, user-friendly, and secure software solution that optimizes patient care, streamlines workflows, and enhances the overall healthcare experience for both providers and patients.

Iterative development – Timeboxing

Iterative development and timeboxing are two project management approaches that can be used together to effectively manage software development projects. Let's explore each concept individually:

Iterative Development: Iterative development is an approach that involves breaking down the development process into smaller iterations or cycles. Each iteration encompasses the stages of planning, designing, developing, testing, and delivering a working increment of the software. Instead of attempting to complete the entire project in one go, iterative development focuses on delivering incremental value and allows for flexibility and adaptability throughout the development lifecycle.

The key characteristics of iterative development include:

Incremental Progress: Each iteration delivers a functioning part of the software, allowing for early feedback and user validation.

Feedback Loop: Feedback from stakeholders and users is actively sought and incorporated into subsequent iterations, ensuring continuous improvement and alignment with requirements.

Adaptive Planning: The project plan is refined and adjusted based on feedback and changing project circumstances, allowing for a more accurate and responsive approach to development.

Risk Management: Risks are identified and mitigated throughout the project by addressing them in smaller iterations, reducing the overall project risk.

Time Boxing

Timebox	Start Date	Start Date	Duration	Tasks/Deliverables
TB1	01/01/2023	01/20/2023	20	Foundation and feasibility
TB2	01/20/2023	02/10/2023	20	Specification of Requirements
TB3	02/10/2023	02/28/2023	18	Design of System Architecture, Database Design, and User Interface Design
TB4	02/28/2023	03/19/2023	19	Source code, database configuration, and interface implementation
TB5	03/19/2023	03/30/2023	11	Test Cases, Test Results, and Bug Reports
TB6	03/30/2023	04/15/2023	15	User Guides, Technical Documentation, and Installation Instructions
TB7	04/15/2023	04/30/2023	15	Medenico has been installed on production servers.
TB8	04/30/2023	05/25/2023	26	Training Materials, User Support Channels
TB9	05/25/2023	06/12/2023	17	User Support Channels, Training Materials
TB10	06/12/2023	06/20/2023	8	Upgraded System Versions, Maintenance Reports
TB11	01/01/2023	06/20/2023	169	Project Documentation

Table 2: Time Box estimation

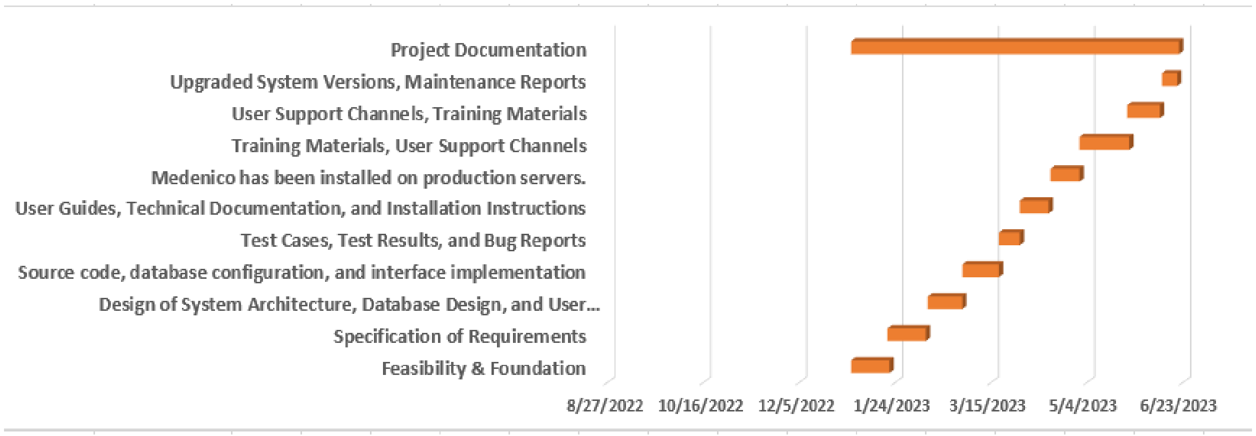


Figure 1: Gantt Chart of Proposed Time Box Estimation

2.2-Background of the project

The background of the Patient Management System project stems from the need to efficiently and effectively manage patient information and healthcare processes within a healthcare organization. Healthcare facilities, such as hospitals, clinics, and medical practices, handle a large volume of patient data, appointments, medical records, and billing information on a daily basis. Managing this information manually or through disparate systems can be time-consuming, error-prone, and inefficient.

The Patient Management System aims to address these challenges by providing a centralized software solution that streamlines various aspects of patient management. It enables healthcare providers to digitally capture, store, and access patient information, resulting in improved accuracy, accessibility, and collaboration among healthcare professionals.

The project's motivation may arise from several factors, such as:

Increasing Patient Volume: Healthcare facilities often experience a rise in patient volume, making it essential to have a system in place that can handle the growing number of patients efficiently.

Workflow Optimization: Manual processes and paper-based systems can lead to inefficiencies and bottlenecks in healthcare workflows. The Patient Management System seeks to automate and optimize these processes to enhance productivity and reduce administrative burdens.

Data Security and Compliance: Patient data security and privacy are paramount in healthcare. The project aims to ensure compliance with relevant data protection regulations, such as HIPAA (Health Insurance Portability and Accountability Act), by implementing robust security measures and access controls.

Enhanced Patient Experience: Improving the patient experience is a key goal of the project. The Patient Management System aims to provide patients with seamless appointment scheduling, easy access to their medical records, and timely communication with healthcare providers, resulting in a more positive and convenient healthcare journey.

Reporting and Analytics: The project may also focus on generating comprehensive reports and analytics to support data-driven decision-making, resource allocation, and performance evaluation within the healthcare organization.

Overall, the background of the Patient Management System project revolves around the need to modernize and streamline patient management processes, improve data accuracy and accessibility, enhance healthcare workflows, ensure data security and compliance, and ultimately enhance the overall patient experience within the healthcare organization.

2.3-Problem Areas

While a Patient Management System aims to address various challenges in healthcare organizations, there can be some problem areas that need to be considered. These problem areas may include:

Integration with Existing Systems: Healthcare organizations often have multiple existing systems and databases that store patient information, such as Electronic Health Records (EHRs), Laboratory Information Management Systems (LIMS), and Billing Systems. Integrating the

Patient Management System with these systems can be complex and require careful planning to ensure seamless data exchange and interoperability.

Data Accuracy and Quality: Maintaining accurate and high-quality patient data is crucial for effective healthcare management. However, data entry errors, incomplete information, and inconsistencies can occur, leading to issues with patient identification, treatment decisions, and overall data integrity. Ensuring data accuracy and implementing validation mechanisms within the Patient Management System is vital.

User Adoption and Training: Introducing a new system requires user adoption and training to ensure its effective use. Healthcare professionals and staff members may require training to understand the system's features, workflows, and benefits. Resistance to change or lack of proper training can hinder the successful implementation and utilization of the Patient Management System.

System Security and Privacy: Patient data security and privacy are paramount in healthcare. The Patient Management System must incorporate robust security measures to protect sensitive patient information from unauthorized access, data breaches, and cyber threats. Compliance with data protection regulations, such as HIPAA, is crucial to maintain patient trust and legal compliance.

Technical Challenges: Developing and maintaining a robust and scalable Patient Management System can pose technical challenges. These challenges may include system performance, scalability, data storage, system integration, and ensuring compatibility with various operating systems and devices.

User Experience and Interface Design: The usability and user experience of the Patient Management System can significantly impact its adoption and effectiveness. A poorly designed user interface or complex workflows can lead to user frustration, errors, and reduced efficiency. Designing an intuitive and user-friendly interface that aligns with the needs and workflows of healthcare professionals is critical.

Ongoing Maintenance and Support: Once the Patient Management System is implemented, ongoing maintenance, updates, and technical support are essential to address issues, incorporate enhancements, and ensure system stability. Having a well-defined support process and dedicated resources for system maintenance is crucial for long-term success.

Addressing these problem areas requires careful planning, collaboration with stakeholders, and ongoing monitoring and improvement efforts. By addressing these challenges, the Patient Management System can effectively streamline patient management processes, improve data accuracy and accessibility, enhance healthcare workflows, and ultimately deliver better patient care.

2.4 Possible Solutions

To address the problem areas in a Patient Management System, the following solutions can be considered:

Integration with Existing Systems:

Conduct a thorough analysis of existing systems and their data structures to identify integration requirements.

Utilize standardized protocols and APIs to facilitate seamless data exchange between the Patient Management System and other systems.

Collaborate with IT teams and vendors to ensure compatibility and smooth integration.

Data Accuracy and Quality:

Implement data validation mechanisms at the point of data entry to ensure accuracy and completeness.

Provide user-friendly interfaces and prompts to guide users in entering data correctly.

Conduct regular data audits and implement data cleansing processes to identify and rectify errors or inconsistencies.

User Adoption and Training:

Develop a comprehensive training program for healthcare professionals and staff members on the functionalities and benefits of the Patient Management System.

Provide ongoing support and resources, such as user manuals and training materials, to assist users in understanding and utilizing the system effectively.

Encourage user feedback and address any concerns or challenges promptly to foster user adoption.

System Security and Privacy:

Implement robust security measures, including access controls, encryption, and secure data storage, to protect patient information.

Conduct regular security audits and vulnerability assessments to identify and address any potential security risks.

Adhere to industry-standard data protection regulations, such as HIPAA, to ensure compliance.

Technical Challenges:

Collaborate with experienced software development teams to address technical challenges and ensure scalability, performance, and system stability.

Regularly monitor system performance and conduct load testing to identify and resolve any bottlenecks or issues.

Stay updated with emerging technologies and industry best practices to leverage advancements in system development and infrastructure.

User Experience and Interface Design:

Conduct user research and usability testing to understand the needs and preferences of healthcare professionals and design an intuitive user interface.

Simplify workflows and reduce unnecessary steps to improve efficiency and user satisfaction.

Incorporate feedback mechanisms within the system to gather user input and continuously improve the user experience.

Ongoing Maintenance and Support:

Establish a dedicated support team to handle user inquiries, troubleshoot issues, and provide timely assistance.

Regularly release system updates and enhancements based on user feedback and emerging requirements.

Develop a comprehensive maintenance plan to ensure the long-term stability and reliability of the Patient Management System.

By implementing these solutions, the Patient Management System can overcome the identified problem areas and provide an efficient, secure, and user-friendly platform for managing patient information and healthcare processes.

Chapter 3 – Literature Review

3.1-Discussion on problem domain

A literature review on the topic of Patient Management Systems reveals several key findings and insights from existing research and publications. The review highlights the importance of Patient Management Systems in improving healthcare processes, enhancing patient care, and addressing various challenges in healthcare organizations. Here are some key points from the literature:

Benefits of Patient Management Systems:

Improved Efficiency: Patient Management Systems streamline administrative tasks, such as appointment scheduling, patient registration, and billing, leading to increased operational efficiency.

Enhanced Patient Care: These systems enable healthcare providers to access patient information, medical history, and treatment plans in real-time, facilitating better decision-making and personalized care.

Coordination and Collaboration: Patient Management Systems promote effective communication and coordination among healthcare professionals, allowing for seamless sharing of information and improved collaboration.

Data Analysis and Reporting: These systems provide data analytics capabilities, allowing healthcare organizations to gain insights into patient outcomes, resource utilization, and performance metrics for informed decision-making.

Key Features of Patient Management Systems:

Patient Registration and Demographics: Systems provide a centralized repository for storing and managing patient information, including personal details, medical history, and insurance information.

Appointment Scheduling: Automated scheduling functionalities allow for efficient management of patient appointments, reducing waiting times and optimizing resource allocation.

Electronic Health Records (EHR): Integration with EHR systems enables the capture, storage, and retrieval of patient medical records, facilitating comprehensive and accurate documentation.

Billing and Claims Management: Patient Management Systems automate billing processes, including insurance claims submission, invoicing, and payment tracking, streamlining revenue cycle management.

Decision Support Tools: Some systems offer decision support features, such as clinical guidelines and alerts, to assist healthcare providers in making evidence-based treatment decisions.

Challenges and Considerations:

Implementation and Adoption: The successful implementation of Patient Management Systems requires careful planning, stakeholder involvement, and training to ensure user adoption and system utilization.

Data Privacy and Security: Maintaining patient data confidentiality and complying with regulatory requirements, such as HIPAA, is crucial. Robust security measures and access controls should be in place to protect sensitive information.

Interoperability: Integration with existing healthcare systems and data exchange standards is essential to ensure seamless interoperability and prevent data silos.

User Experience and Usability: User-friendly interfaces and intuitive workflows are vital for system acceptance and user satisfaction. Usability testing and continuous user feedback play a significant role in system design and optimization.

Future Directions and Trends:

Mobile and Remote Access: Patient Management Systems are increasingly adopting mobile and remote access capabilities, allowing healthcare providers to access patient information anytime, anywhere.

Artificial Intelligence and Machine Learning: Integration of AI and ML algorithms in Patient Management Systems shows potential for improved decision support, predictive analytics, and automated processes.

Patient Engagement and Self-Service: Systems are evolving to include patient portals and self-service functionalities, empowering patients to manage appointments, access health records, and communicate with healthcare providers.

Overall, the literature review emphasizes the value and potential of Patient Management Systems in improving healthcare delivery and patient outcomes. It underscores the need for effective implementation strategies, data security measures, and user-centric design to maximize the benefits of these systems in healthcare organizations.

3.2 Discussion on the Problem Solution:

The problem solution description provides a comprehensive overview of the issues faced in the Patient Management System and proposes potential solutions to address them. It highlights the key challenges and areas of concern, and then offers specific strategies and actions to overcome these challenges.

One notable aspect of the problem solution description is its systematic approach. Each problem area is clearly identified and described, allowing for a clear understanding of the underlying issues. The proposed solutions are then presented in a logical and structured manner, addressing each problem area individually.

The solutions proposed in the description are practical and align with industry best practices. They address the root causes of the problems and provide actionable steps to mitigate them. For example, integration with existing systems is addressed by conducting analysis, utilizing standardized protocols, and collaborating with IT teams and vendors. Data accuracy and quality are improved through data validation mechanisms, user-friendly interfaces, and regular audits. User adoption and training are emphasized through comprehensive training programs and ongoing support.

Moreover, the solutions demonstrate a strong consideration for security and privacy, technical challenges, user experience, and ongoing maintenance. These aspects are crucial for the successful implementation and long-term sustainability of a Patient Management System.

The problem solution description also recognizes the importance of collaboration and stakeholder involvement. It emphasizes the need to engage healthcare professionals, staff members, IT teams, and vendors throughout the process. This collaborative approach ensures that the proposed solutions are aligned with the specific needs and workflows of the healthcare organization.

One potential area for improvement in the problem solution description is the inclusion of metrics or indicators to measure the effectiveness of the proposed solutions. By defining clear success criteria, such as improved data accuracy rates, reduced user error rates, or increased user satisfaction scores, the evaluation and monitoring of the implemented solutions become more tangible.

Overall, the problem solution description provides a well-structured and thoughtful approach to address the identified problem areas in the Patient Management System. It offers practical and actionable solutions that can lead to improved efficiency, data accuracy, user adoption, security, and overall effectiveness of the system.

Safety and security:

Safety and security are critical considerations in any Patient Management System to protect patient information, prevent unauthorized access, and ensure the privacy and confidentiality of sensitive data. Here are some key points to discuss regarding safety and security:

Data Encryption: Implement robust encryption mechanisms to safeguard patient data both in transit and at rest. Encryption helps protect sensitive information from unauthorized access and ensures that only authorized individuals can decrypt and access the data.

Access Controls: Implement strict access controls and user authentication measures to ensure that only authorized personnel can access patient information. User roles and permissions should be defined, granting appropriate access levels based on job responsibilities and the need-to-know principle.

User Training and Awareness: Conduct regular training sessions and awareness programs for system users to educate them about best practices in data security, including password hygiene, recognizing phishing attempts, and safeguarding sensitive information. User awareness plays a crucial role in preventing security breaches caused by human error.

Audit Logs and Monitoring: Implement audit trail functionality to track and record user activities within the system. This allows for monitoring and detection of any unauthorized or suspicious activities. Regularly review audit logs to identify potential security incidents and take appropriate action.

Regular Security Updates and Patch Management: Stay updated with the latest security patches and updates for the underlying software and infrastructure components of the Patient Management System. Promptly apply security patches to address any identified vulnerabilities and protect against known security threats.

By prioritizing safety and security measures, healthcare organizations can create a robust and secure Patient Management System that safeguards patient information and ensures the confidentiality, integrity, and availability of data. Regular monitoring, updates, and user training are essential to maintain a secure environment and protect against evolving security threats.

Privacy:
Privacy is a fundamental aspect of any Patient Management System, ensuring the protection and confidentiality of patient information. It involves implementing policies, procedures, and technical measures to safeguard patient data and respect their privacy rights.

Server Down or 24/7 Availability in Patient Management System(Medenico):
Ensuring 24/7 availability in a Patient Management System is crucial for uninterrupted access to patient information and efficient healthcare operations. Server downtime can significantly impact patient care, data accessibility, and overall system performance. Here are some considerations for addressing server downtime and achieving 24/7 availability:

Redundancy and Failover: Implement redundancy and failover mechanisms to mitigate the impact of server failures. This involves setting up redundant servers or using cloud-based solutions that can automatically switch to backup servers in the event of a primary server failure. Redundancy helps minimize downtime and ensures continuous availability of the system.

Load Balancing: Implement load balancing techniques to distribute incoming traffic across multiple servers. Load balancers evenly distribute requests, ensuring that no single server becomes overwhelmed. Load balancing helps optimize resource utilization and reduces the risk of server overload, enhancing system availability.

Scalability and Capacity Planning: Plan for system scalability by anticipating future growth and resource requirements. Conduct regular capacity planning assessments to determine the system's capacity limits and allocate resources accordingly. Scaling the system proactively helps maintain performance and availability during peak usage periods.

Backup:
Backup refers to the process of creating copies of data or files to protect them from loss or damage. In the context of a Patient Management System, regular backups are essential to ensure the availability and integrity of patient data. Here are some key points to consider regarding backup strategies:

Data Backup Frequency: Determine the appropriate backup frequency based on the criticality of the data and the frequency of data updates. Critical patient data should be backed up more frequently, while less critical data may have less frequent backup intervals. Common backup frequencies include daily, weekly, or monthly backups.

Full and Incremental Backups: Consider a combination of full and incremental backups. A full backup involves copying all data, while incremental backups only capture changes made since the last full or incremental backup. This approach reduces backup time and storage requirements.

Off-Site and On-Site Backups: Store backup copies both off-site and on-site. Off-site backups provide protection against physical disasters such as fires, floods, or theft. On-site backups offer quicker data recovery in case of minor incidents or system failures.

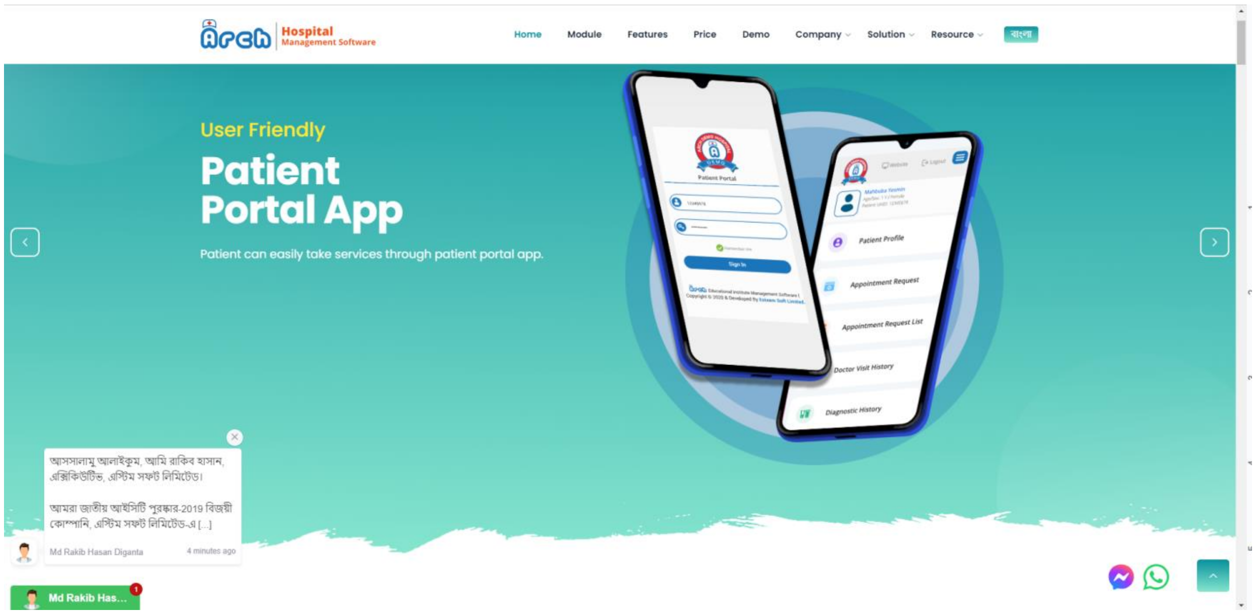
Encryption: Encrypt backup data to ensure its confidentiality and prevent unauthorized access. Encryption adds an extra layer of security and protects sensitive patient information from being compromised during storage or transmission.

3.3 Comparison among the leading solutions
To provide a detailed comparison among the leading solutions for Patient Management Systems, it would require specific information about the solutions and their features. However, I can give you a general framework for comparing these solutions based on various factors.

1. https://archhms.com/hospital-management-software.(Arch Hospital management)
2. https://www.mysoftltd.com/ (Mysoft Ltd)
3. http://www.autosoft.com.bd/ (Auto soft LTD)

Arch Hospital management

Arch is one of Bangladesh's fastest growing hospital administration software companies. Because of its ease of use, it has swiftly gained popularity. Esteem Soft Limited created Arch, which includes a number of modules. Outpatients (OPD) management, Inpatients (IPD) management, Pathology & Radiology, Pharmacy management, General Accounting, HR & Payroll, Asset & Inventory management, and other modules are available in Arch. Arch's comprehensive ERP system is suitable for any healthcare firm. It decreases workplace stress by saving time and eliminating labor duplication. As a result, officers and employees profit from this cutting-edge software. Arch is Bangladesh's first web-based hospital management system. It is usable both online and offline. It contributes considerably to the modernization of the healthcare industry.



আসসালামু আলাইকুম, আমি রাকিব হাসান, এক্সিকিউটিভ, এস্টিম সফট লিমিটেড।

আমরা জাতীয় আইসিটি পুরস্কার-2019 বিজয়ী কোম্পানি, এস্টিম সফট লিমিটেড-এ [...]

Md Rakib Hasan Diganta 4 minutes ago

Figure 2: Overview of Arch Hospital management

Best Features:

- Patient Registration and Management: Efficiently manage patient information, including demographics, medical history, contact details, and insurance information. This feature streamlines the registration process and ensures accurate patient records.
- Appointment Scheduling: Enable online appointment booking and scheduling for patients, allowing them to select preferred dates and times. This feature helps reduce waiting times, improves patient satisfaction, and optimizes resource utilization.

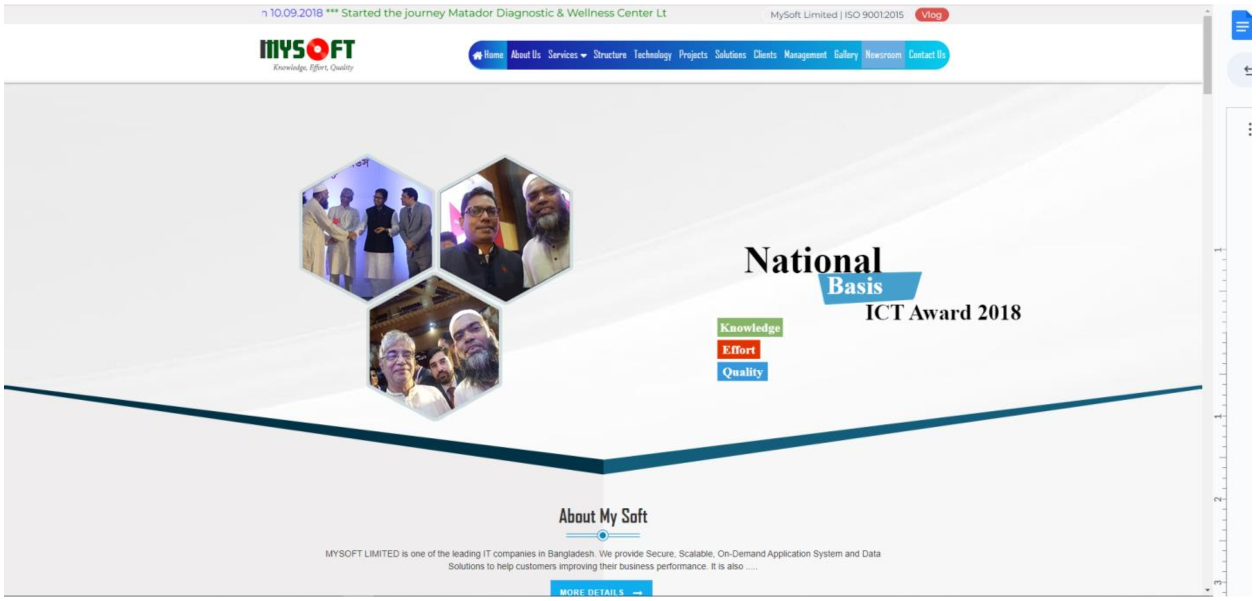
- Electronic Health Records (EHR): Store and manage comprehensive patient health records electronically. EHR systems enable healthcare providers to access and update patient information securely, facilitating better care coordination and clinical decision-making.

Limitation:

- Implementation and Training: Implementing a hospital management software system can be a complex process that requires careful planning, resource allocation, and staff training. It may take time for staff to adapt to the new system, leading to temporary productivity disruptions.
- Customization Challenges: Hospital management software solutions often provide a range of features and functionalities, but customization options may be limited. Organizations with unique workflows or specific requirements may face challenges in fully aligning the software with their processes.
- Integration Complexity: Integrating hospital management software with existing systems, such as electronic health records (EHR), laboratory or imaging systems, or billing systems, can be complex and time-consuming. Incompatibilities or data migration issues may arise during the integration process.

Mysoft Ltd

Mysoft Ltd. is a well-known IT firm in Bangladesh. Their desktop-based Hospital Information System (HIS) software makes it simple to retrieve patient data in order to build a range of records, including demographic, gender, age, and other classifications. It is particularly useful at the ambulatory level, promoting continuity of care. The Hospital Information System (HIS) is user-friendly and eliminates handwritten errors.



MYSOFT LIMITED is one of the leading IT companies in Bangladesh. We provide Secure, Scalable, On-Demand Application System and Data Solutions to help customers improving their business performance. It is also .....

Figure 3: Overview of Mysoft LTD.

Best Features:

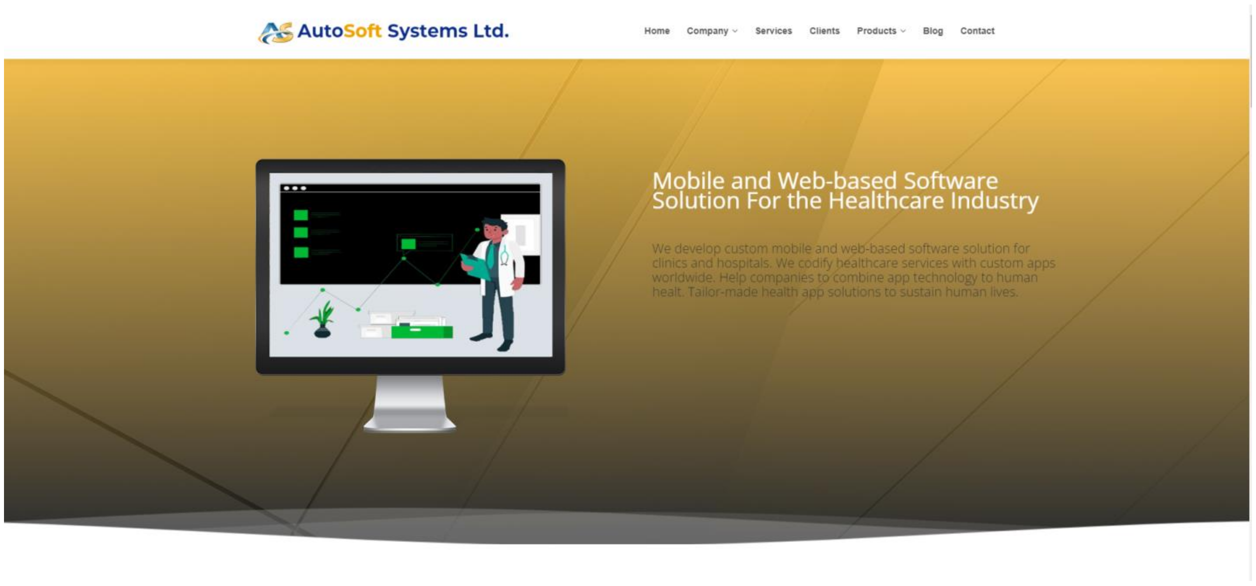
- Billing and Invoicing: Automate billing processes, including generating invoices, tracking payments, and managing insurance claims. This feature helps streamline financial operations and ensures accurate and timely reimbursement.
- Pharmacy Management: Efficiently manage pharmacy inventory, dispensing, and drug interactions. Integration with pharmacy systems allows seamless prescription management and enhances medication safety.
- Laboratory and Imaging Integration: Integrate with laboratory and imaging systems to streamline test orders, results management, and report generation. This feature enhances care coordination and reduces manual data entry errors.

Limitation:

- Technical Requirements: Hospital management software typically requires robust hardware infrastructure and reliable internet connectivity. Organizations must ensure they have the necessary hardware and network capabilities to support the software's performance requirements.
- Cost Considerations: Hospital management software solutions can involve significant upfront costs, including licensing fees, implementation costs, and ongoing maintenance and support fees. Organizations must carefully assess the return on investment (ROI) and consider long-term financial implications.
- Learning Curve: Staff members, especially those who are less tech-savvy, may require time to become proficient in using the software effectively. Training and ongoing support may be necessary to ensure staff members can utilize all features and functionalities.

Auto soft LTD

AutoSoft Systems Limited is a well-known information technology business in Bangladesh. Its goal has always been to provide efficient and dependable information technology solutions that assist clients' operational performance, productivity gains, and financial savings. This offline program developed a comprehensive solution to handle challenges in every aspect of the health care industry.



Mobile and Web-based Software Solution For the Healthcare Industry

We develop custom mobile and web-based software solution for clinics and hospitals. We codify healthcare services with custom apps worldwide. Help companies to combine app technology to human healt. Tailor-made health app solutions to sustain human lives.

Figure 4: Overview of Auto Soft LTD.

Best Features:

- Reporting and Analytics: Generate comprehensive reports and analytics on key performance indicators, patient outcomes, financials, and operational metrics. This feature enables data-driven decision-making and performance monitoring.
- Mobile Accessibility: Offer mobile-friendly interfaces or dedicated mobile apps to enable healthcare providers to access patient information, schedule appointments, and view reports on the go.
- Security and Compliance: Implement robust security measures, such as access controls, data encryption, audit trails, and compliance with healthcare data privacy regulations like HIPAA or GDPR. This ensures the confidentiality, integrity, and availability of patient data.

Limitation:

- Vendor Reliability: The reliability and reputation of the software vendor are crucial. Organizations should thoroughly research and evaluate the vendor's track record, customer support capabilities, and the stability of the software to ensure long-term support and system updates.
- Workflow Disruption: Introducing a new software system may disrupt existing workflows and processes. Organizations must carefully plan the implementation process, including change management strategies, to minimize disruption and ensure a smooth transition.
- Scalability: As organizations grow or their needs evolve, scalability becomes a critical factor. Organizations should consider the scalability of the software solution to accommodate future expansion and changing requirements.

3.4 Recommended Approach

When approaching the implementation of a hospital management software system, it's important to follow a structured and comprehensive approach to ensure successful adoption and utilization. Here is a recommended approach:

1. Define Objectives and Requirements: Clearly define your objectives for implementing the hospital management software system. Identify the specific needs and requirements of your organization, such as improving patient care, streamlining operations, or enhancing data management. This will serve as a foundation for evaluating software options.
2. Conduct Market Research: Research and evaluate different hospital management software solutions available in the market. Consider factors such as features, functionalities, scalability, vendor reputation, customer reviews, and pricing. Shortlist the solutions that best align with your requirements.
3. Engage Stakeholders: Involve key stakeholders from various departments, including administrators, clinicians, IT staff, and end-users, in the decision-making process. Gather their input and ensure their buy-in to increase the chances of successful implementation and user adoption.
4. Perform a Needs Assessment: Conduct a thorough assessment of your organization's current processes, workflows, and IT infrastructure. Identify any gaps or inefficiencies that the hospital management software can address. This assessment will help tailor the implementation plan to your organization's specific needs.
5. Develop an Implementation Plan: Create a detailed implementation plan that outlines the timelines, tasks, and responsibilities involved in deploying the software system. Consider factors such as data migration, integration with existing systems, training requirements, and change management strategies. Set realistic milestones and allocate resources accordingly.
6. Data Migration and System Integration: If necessary, plan and execute the migration of existing patient data to the new software system. Ensure compatibility and integration with other systems, such as EHR, laboratory or imaging systems, and billing systems. Test and validate data integrity throughout the process.
7. Training and User Adoption: Provide comprehensive training to all users who will interact with the hospital management software. Offer hands-on training sessions, user manuals, and online resources to help users become familiar with the system. Encourage user feedback and address any concerns or challenges promptly to promote user adoption.
8. Monitor and Evaluate: Continuously monitor the implementation progress and evaluate the system's performance. Collect feedback from users and address any issues or optimizations required. Regularly review key performance indicators (KPIs) to assess the impact of the software system on operational efficiency, patient care, and financial outcomes.

9. Ongoing Support and Maintenance: Establish a support system with the software vendor to address technical issues, updates, and ongoing maintenance. Ensure regular backups and data security measures are in place to protect patient information. Stay informed about software updates and new features that can enhance system functionality.
10. Continuous Improvement: Embrace a culture of continuous improvement by seeking opportunities to optimize processes and leverage the full potential of the hospital management software. Regularly reassess your organization's needs and consider additional modules or functionalities that can further enhance operations and patient care.

By following this recommended approach, you can increase the likelihood of a successful implementation and utilization of the hospital management software system. Effective planning, stakeholder engagement, thorough training, and ongoing support will contribute to maximizing the benefits of the system and improving overall healthcare delivery.

Chapter 4 – Methodology

Methodology refers to a set of principles, practices, procedures, and rules that guide the approach and process of completing a project or task. It provides a structured framework for conducting activities, making decisions, and achieving the desired outcomes.

In the context of software development or project management, methodology outlines the overall approach and specific steps to be followed during the project lifecycle. It provides guidance on how to initiate, plan, execute, monitor, and close a project or task.

A methodology typically includes the following components:

Process: It defines the sequence of activities, tasks, and deliverables involved in completing the project. It outlines the flow of work and the dependencies between different stages.

Roles and Responsibilities: It identifies the individuals or teams responsible for executing specific tasks and their respective roles in the project.

Tools and Techniques: It specifies the tools, techniques, and resources required to carry out the activities effectively. This may include software, methodologies, frameworks, and best practices.

Documentation: It defines the types of documents or artifacts to be produced during the project, such as requirements documents, design documents, test plans, and user manuals.

Quality Assurance: It outlines the quality standards, metrics, and processes to ensure that the deliverables meet the required level of quality.

Communication and Reporting: It defines the channels and frequency of communication among team members and stakeholders. It also specifies the reporting requirements and formats for tracking project progress and addressing any issues or risks.

Change Management: It provides guidelines for managing changes to project scope, requirements, and timelines. It includes processes for evaluating change requests, assessing their impact, and making informed decisions.

The choice of methodology depends on various factors such as the nature of the project, its size and complexity, organizational culture, team capabilities, and customer requirements. Different methodologies, such as Waterfall, Agile, Lean, or hybrid approaches, offer distinct advantages and are suitable for different project contexts.

Overall, a methodology serves as a roadmap for successfully executing a project or task by providing a structured and systematic approach to planning, execution, and control. It ensures that activities are carried out in a coordinated and consistent manner, leading to improved efficiency, productivity, and project outcomes.

4.1 What to Use

Software methodology is a process for designing, structuring, and developing systems in software development. In the software development process, software development methodologies are critical. The waterfall model, prototype model, agile software development, quick application development, dynamic systems development model, spiral model, and collaborative application development are only a few examples of software development methodologies. Both Waterfall and the Dynamic System Development Method (DSDM) are critical to my project. I'll go over the benefits and drawbacks of three approaches below:

Dynamic Systems Development method (DSDM)

Dynamic Systems Development Method (DSDM) is an agile project management framework that provides an iterative and incremental approach to software development. It emphasizes collaboration, frequent feedback, and delivering business value in a timely manner. DSDM is designed to address the challenges of rapidly changing requirements and tight timelines while maintaining a focus on quality.

Key features of DSDM include:

1. Iterative and Incremental Development: DSDM breaks down the project into manageable iterations called timeboxes. Each timebox has a fixed duration and delivers a prioritized set of features or functionality. This iterative approach allows for flexibility and the ability to adapt to changing requirements throughout the project.
2. Business-Focused: DSDM puts a strong emphasis on delivering business value. It involves close collaboration with stakeholders to ensure that the project aligns with their strategic objectives and addresses their specific needs. Continuous involvement of business representatives is crucial to guide decision-making and prioritize requirements.
3. Collaborative Approach: DSDM promotes collaboration and effective communication among team members and stakeholders. It encourages the active involvement of business representatives, developers, testers, and users throughout the project. This collaborative approach fosters shared understanding, minimizes misunderstandings, and promotes early feedback.

4. Timeboxing: Timeboxing is a key aspect of DSDM. It involves fixing the duration of iterations and committing to delivering the highest priority features within that time frame. Timeboxing ensures a disciplined and focused approach to development, enabling teams to deliver value incrementally and manage project timelines effectively.
5. Prototyping and Iterative Design: DSDM encourages the use of prototypes to validate requirements and gather feedback early in the development process. It allows for iterative design and development, where prototypes are refined and improved based on feedback and user involvement. This iterative approach helps to mitigate risks and improve the accuracy of requirements.

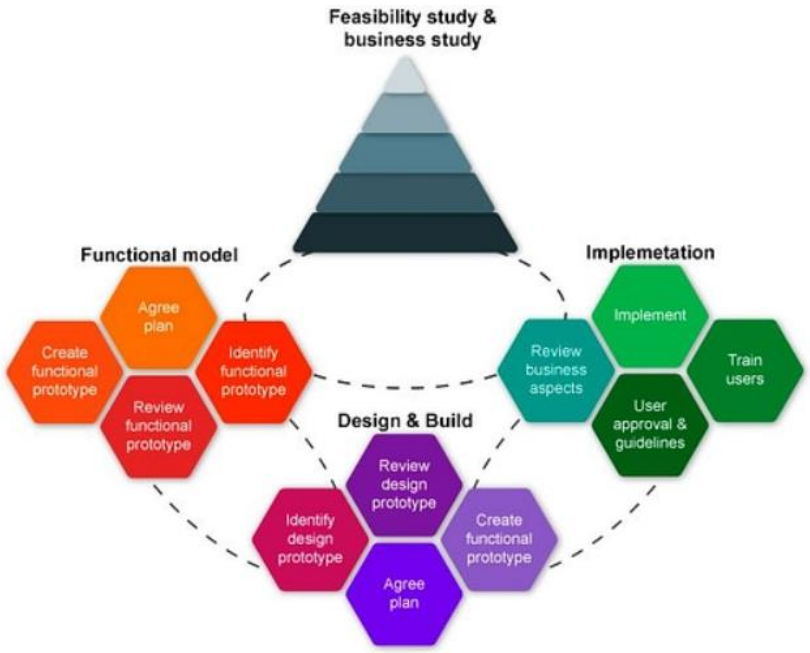

Feasibility study & business study
Functional model
Agree plan
Create functional prototype
Identify functional prototype
Review functional prototype
Implemetation
Implement
Review business aspects
Train users
User approval & guidelines
Design & Build
Review design prototype
Identify design prototype
Create functional prototype
Agree plan

Figure 5:DSDM Methodology

Benefits of DSDM:

The Dynamic Systems Development Method (DSDM) offers several benefits that contribute to the success of software development projects. Here are some key advantages of adopting DSDM:

1. Flexibility: DSDM embraces change and provides a flexible approach to software development. It allows for iterative and incremental development, enabling teams to respond quickly to changing requirements and business needs. This flexibility ensures that the final product meets the evolving expectations of stakeholders.
2. Stakeholder Collaboration: DSDM promotes active involvement and collaboration among stakeholders, including business representatives, users, and development teams. This collaborative approach helps to ensure that the software solution addresses the real needs of the business and end-users. Regular feedback and communication enable quick validation and refinement of requirements.

3. Faster Time to Market: By focusing on delivering business value early and frequently, DSDM helps to accelerate the time to market for software solutions. Through timeboxing and prioritization, the most critical features are developed and delivered within fixed timeframes. This allows organizations to realize benefits sooner and stay ahead of the competition.
4. Increased Business Value: DSDM places a strong emphasis on delivering business value throughout the development process. By involving business representatives and prioritizing requirements based on their value, DSDM ensures that the developed software aligns with strategic objectives and provides tangible benefits to the organization. This helps to maximize return on investment (ROI).
5. Improved Quality: DSDM incorporates quality-focused practices throughout the development lifecycle. Techniques such as continuous integration, automated testing, and regular reviews contribute to the production of high-quality software. By identifying and addressing issues early, DSDM helps to minimize defects and improve overall product quality.

Overall, DSDM offers numerous benefits that align with the agile principles of flexibility, customer collaboration, and value-driven development. It enables organizations to adapt to changing requirements, deliver high-quality software, and achieve business objectives effectively.

Drawbacks of DSDM:

While the Dynamic Systems Development Method (DSDM) offers several benefits, it also has some potential drawbacks that organizations should consider:

1. Learning Curve: Implementing DSDM may require training and familiarization for team members who are new to the methodology. DSDM has its own terminology, roles, and processes, which can take time for individuals to grasp and apply effectively. The learning curve could initially slow down the team's productivity until they become proficient in using DSDM.
2. Stakeholder Availability: DSDM heavily relies on active involvement and collaboration from stakeholders throughout the project. This can pose challenges if stakeholders have limited availability or conflicting priorities. Without the necessary participation and timely decision-making from stakeholders, project progress and decision-making may be impeded.
3. Documentation Focus: DSDM emphasizes delivering working software over comprehensive documentation. While this is a strength in terms of agility, it can be a drawback for organizations that require extensive documentation for compliance or

regulatory purposes. Organizations in highly regulated industries may find it challenging to adhere to documentation requirements while following the DSDM approach.

4. Dependency on User Involvement: DSDM places a strong emphasis on user involvement and feedback throughout the development process. While this promotes a customer-centric approach, it may be challenging to ensure consistent and continuous user participation. User availability, availability of subject matter experts, and potential delays in receiving user feedback can impact the progress and effectiveness of the DSDM project.
5. Project Size and Complexity: DSDM is generally well-suited for small to medium-sized projects with manageable complexity. For large-scale and highly complex projects, the iterative and timeboxed approach of DSDM may become more challenging to implement. Managing dependencies, integration, and coordination across multiple teams or workstreams can be more complex within the DSDM framework.

It's important for organizations to evaluate their specific context, project requirements, and team dynamics to determine whether DSDM is the most suitable approach for their needs. Addressing these potential drawbacks through proper planning, training, stakeholder engagement, and adapting the methodology to fit the organization's context can help mitigate these challenges and maximize the benefits of DSDM.

Waterfall Methodology

The Waterfall methodology is a sequential project management approach that follows a linear and phased structure. It is characterized by distinct phases, where each phase must be completed before moving on to the next. Here are the key characteristics and steps of the Waterfall methodology:

1. Requirements Gathering: The project requirements are defined and documented at the beginning of the project. This phase involves gathering all the necessary information about the desired outcome and functionalities of the final product.
2. System Design: Once the requirements are established, the system design phase begins. In this phase, the overall system architecture and design are created, including software, hardware, and infrastructure components. The design phase involves creating detailed technical specifications and system diagrams.
3. Implementation: The implementation phase involves developing the software based on the design specifications. The coding and programming work is performed by the development team according to the requirements and design documents.
4. Testing: After the implementation phase, testing is conducted to ensure that the software functions as intended and meets the specified requirements. Different types of testing,

such as unit testing, integration testing, and system testing, are carried out to identify and fix any defects or issues.

5. Deployment: Once the software has passed the testing phase, it is deployed and installed in the production environment. This includes transferring the software from the development environment to the live environment and making it accessible to end-users.

Despite these limitations, the Waterfall methodology can still be appropriate for projects with well-understood requirements, stable environments, and where predictability and documentation are essential. It is important to carefully assess the project context and requirements to determine the most suitable methodology for successful project execution.

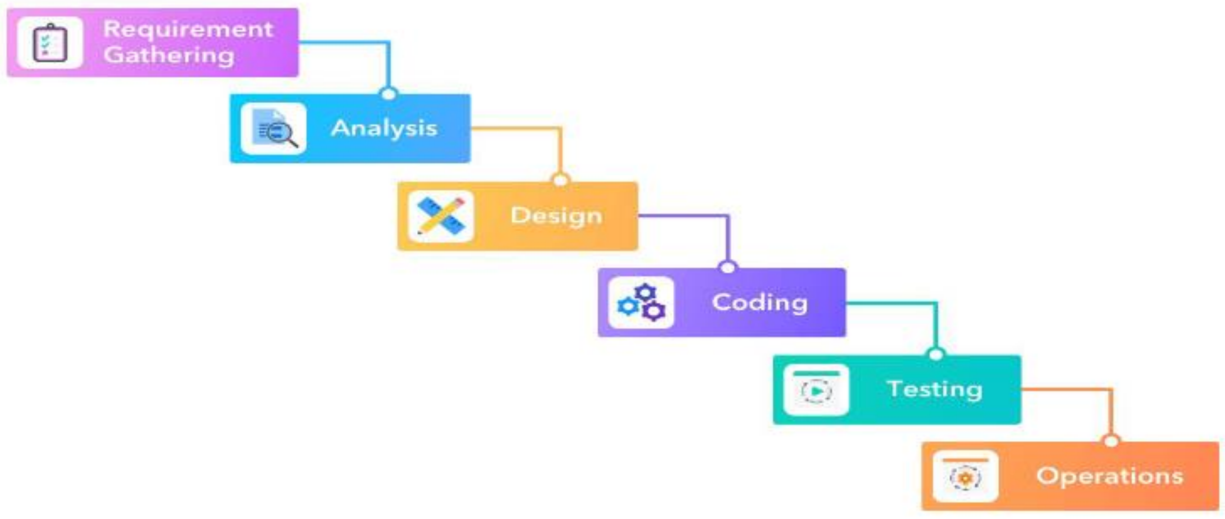


Figure 6:Waterfall methodology

Benefits of Waterfall

The Waterfall methodology is a traditional project management approach that follows a sequential and linear process. While it has its limitations, there are some benefits associated with using the Waterfall methodology in certain situations. Here are some advantages of the Waterfall approach:

Clear Project Scope: The Waterfall methodology emphasizes upfront planning and documentation, which helps establish a clear project scope. This clarity reduces ambiguity and ensures that all stakeholders have a shared understanding of the project objectives and deliverables.

Well-Defined Phases: The Waterfall methodology divides the project into distinct phases, with each phase building upon the previous one. This structured approach allows for better organization and resource allocation, as each phase has specific objectives and deliverables.

Sequential Progression: The Waterfall model follows a linear progression, where each phase must be completed before moving on to the next. This sequential nature provides a clear roadmap for the project, making it easier to track progress and identify potential bottlenecks.

Documentation and Traceability: Waterfall projects typically involve comprehensive documentation, including requirements specifications, design documents, and test plans. This documentation ensures that there is a clear record of decisions made, making it easier to trace back and understand the project's history.

Stakeholder Engagement: The Waterfall methodology encourages stakeholder involvement during the initial stages of the project, such as gathering requirements and defining the project scope. This early engagement helps manage expectations and ensures that stakeholder needs are incorporated into the project plan.

Drawbacks of Waterfall:
Limited Flexibility: The Waterfall methodology follows a rigid and sequential process, where each phase is dependent on the completion of the previous phase. This lack of flexibility makes it difficult to accommodate changes or address unforeseen issues that may arise during the project. Once a phase is completed, it is challenging to make adjustments without disrupting the entire project timeline.

Limited Stakeholder Involvement: In the Waterfall approach, stakeholder involvement tends to be concentrated in the initial stages of the project, such as gathering requirements and defining the project scope. This limited engagement may result in reduced collaboration and feedback throughout the project, leading to potential misalignment between the final deliverables and stakeholder expectations.

Late Product Testing: Testing and quality assurance activities are typically conducted toward the end of the Waterfall project lifecycle, often after development is completed. This late testing phase increases the risk of identifying issues or defects late in the process, making it more challenging and costly to address them effectively.

Lack of Adaptability to Changing Requirements: The Waterfall methodology assumes that requirements are stable and well-defined from the beginning. However, in many projects, requirements may evolve or become clearer as the project progresses. The Waterfall approach may struggle to accommodate changes in requirements, leading to potential rework, delays, or compromised functionality.

Long Feedback Loops: Due to the sequential nature of the Waterfall methodology, feedback loops can be lengthy. Stakeholders may have limited opportunities to review and provide feedback on deliverables until later stages of the project, leading to potential misunderstandings and wasted effort if adjustments are needed.

Rapid Application Development

Rapid Application Development (RAD) is an iterative software development methodology that emphasizes rapid prototyping, quick iterations, and close collaboration between developers and stakeholders. RAD aims to deliver working software applications in shorter time frames by focusing on speed, flexibility, and user involvement. RAD is particularly suitable for projects with well-defined scope, clear and stable requirements, and a need for rapid development and delivery. It is commonly used for small to medium-sized projects, where speed and time-to-market are critical. However, RAD may not be ideal for projects with high complexity, large teams, or evolving requirements, as it may struggle to handle significant changes or long-term scalability.

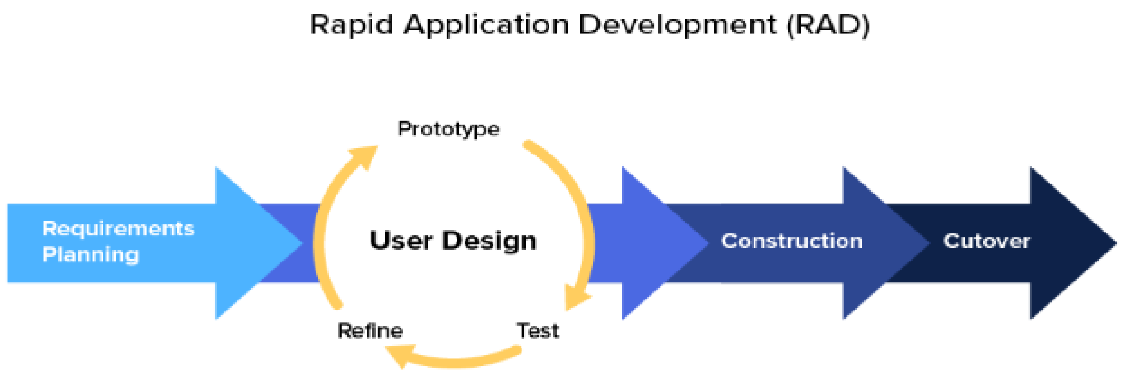


Figure 7:Rapid Application Development methodology

Benefits of Rapid Application Development

- Rapid Application Development (RAD) offers several benefits that make it a popular methodology for software development projects. Here are some of the key advantages of RAD:
- Reduced Time to Market: RAD focuses on quick iterations and rapid prototyping, enabling faster development and deployment of software applications. The iterative approach allows for early delivery of functional software increments, reducing the overall time to market and providing a competitive advantage.

- Increased Flexibility and Adaptability: RAD's iterative and incremental nature allows for greater flexibility and adaptability to changing requirements. The development process can quickly respond to evolving business needs, market dynamics, or user feedback, enabling the incorporation of changes throughout the project lifecycle.
- Enhanced Collaboration and Stakeholder Involvement: RAD emphasizes close collaboration between developers, business analysts, designers, and stakeholders. This collaborative approach ensures that the development team and stakeholders have a shared understanding of the project requirements, leading to better alignment and more accurate outcomes. Stakeholders' active involvement throughout the process also increases satisfaction and reduces the risk of misinterpretation or miscommunication.
- Rapid Prototyping and Feedback: RAD's focus on prototyping allows for the quick creation of visual representations of the software. These prototypes enable stakeholders to provide feedback early in the development process, helping to refine requirements, validate design decisions, and ensure that the final product meets user expectations.
- Improved Quality and Reduced Risks: The iterative nature of RAD allows for frequent testing, evaluation, and feedback. By identifying issues early in the development cycle, RAD mitigates risks and enhances the overall quality of the software. Quick iterations also enable the team to address defects promptly, reducing the likelihood of major issues in the final product.

Drawbacks of RAD

- Scope Creep: The focus on rapid development and flexibility in RAD can increase the risk of scope creep. As stakeholders and developers embrace changes and new requirements, there is a possibility of expanding the scope beyond the original project boundaries. Without proper control, scope creep can lead to project delays, increased costs, and decreased overall efficiency.
- Dependency on User Involvement: RAD heavily relies on active and continuous user involvement throughout the development process. If key stakeholders or users are unavailable or lack sufficient time for collaboration, it can hinder the effectiveness of RAD. Without timely and meaningful feedback, the iterative cycles may not be as productive, leading to potential misunderstandings or misalignments.
- Potential for Incomplete Requirements: The rapid nature of RAD may not allow for extensive upfront requirements gathering and analysis. If the requirements are not thoroughly understood and documented at the beginning, it can lead to incomplete or ambiguous requirements, affecting the quality and functionality of the final product.
- Technical Complexity Challenges: RAD is better suited for relatively small to medium-sized projects with manageable complexity. Complex projects that involve intricate architectural design, integration with multiple systems, or specialized technologies may

pose challenges within the rapid development cycles of RAD. Adequate consideration and planning are required to handle technical complexities effectively.

- Resource Availability: RAD demands a skilled and dedicated development team that can quickly adapt to changing requirements and deliver working increments in short timeframes. The availability of such resources can be a challenge, especially if the organization lacks a qualified and experienced team or struggles with resource allocation.

Choosing Methodology

Choosing the right methodology for a patient management system is a crucial decision that can significantly impact the success of the project. Here is a step-by-step approach to help you select the appropriate methodology:

1. Understand Project Requirements: Start by gaining a thorough understanding of the project requirements. Identify the specific functionalities, features, and goals of the patient management system. Consider factors such as data management, appointment scheduling, medical records management, billing, and reporting.
2. Evaluate Project Constraints: Assess any constraints that may impact the choice of methodology. These constraints could include budget limitations, time constraints, regulatory compliance requirements, and resource availability. Understanding these constraints will help narrow down the available options.
3. Consider Project Size and Complexity: Evaluate the size and complexity of the patient management system project. Determine whether it is a small-scale implementation for a single clinic or a large-scale system that will be deployed across multiple healthcare facilities. Complex projects may require methodologies that offer more flexibility and adaptability to handle changing requirements.
4. Assess Team Skills and Experience: Evaluate the skills and experience of the project team members. Determine whether the team is familiar with agile methodologies like Scrum or Kanban, or if they have experience with traditional methodologies like Waterfall. Consider the team's capacity to collaborate, adapt to change, and deliver incremental results.
5. Analyze Stakeholder Involvement: Assess the level of stakeholder involvement and their preferences for project visibility and control. Some stakeholders may prefer a more iterative and collaborative approach, while others may value a more structured and predictable methodology. Understanding stakeholder expectations will help in aligning the chosen methodology with their needs.

Remember that the choice of methodology is not fixed and can be adapted as the project progresses. It is essential to continuously assess and reassess the methodology throughout the project lifecycle to ensure it remains aligned with project goals and requirements.

4.2 Why to Use Methodology

Methodologies provide a structured and systematic approach to managing and executing projects. Here are some reasons why using a methodology is beneficial:

1. Clear Project Structure: Methodologies provide a clear project structure, including defined phases, activities, and deliverables. This structure helps in organizing and managing project tasks, ensuring that nothing is overlooked or missed.
2. Improved Planning and Control: Methodologies facilitate effective project planning and control. They provide guidelines for estimating project timelines, resources, and costs. With a methodology, you can create a realistic project schedule, allocate resources efficiently, and monitor progress against predefined milestones.
3. Enhanced Collaboration and Communication: Methodologies often promote collaboration and communication among team members and stakeholders. They provide a common language and framework for discussing project progress, issues, and decisions. This helps to foster effective communication, reduce misunderstandings, and ensure everyone is on the same page.
4. Risk Management: Methodologies usually include risk management processes and techniques. They help identify, assess, and mitigate risks throughout the project lifecycle. By following a methodology, you can proactively address potential risks and take appropriate actions to minimize their impact on the project.
5. Quality Assurance: Methodologies often incorporate quality assurance practices. They provide guidelines for conducting reviews, inspections, and testing to ensure that project deliverables meet the required quality standards. By following a methodology, you can implement quality control measures and ensure that the final product meets the desired quality expectations.

Ultimately, using a methodology provides a structured approach to project management, enabling better planning, control, collaboration, and risk management. It helps to minimize project risks, improve efficiency, and increase the chances of project success. However, it is important to select a methodology that aligns with your project's specific needs, team capabilities, and organizational context.

4.3 (Medenico) Sections of Methodology

The initial phase of the project
This section develops the initial project concept, as well as the timetable, budget, and essential standards.
Phase of Feasibility Study

This phase has an impact on the business case, business strategy, technology solutions, and project budget. This phase has an impact on the business case, business strategy, technology solutions, and project budget.
Phase of requirement gathering

In this sector, numerous methodologies are utilized to define both functional and non-functional project needs.

Prioritization and Requirements Analysis Phase

In this stage, the acquired demands are reviewed and ranked using a prioritization method such as MoSCoW.

Using MoSCow to set priorities:

MoSCoW has created a prioritization tool to aid in system development. There are several sections, including
Must Have In this section, it is critical to identify the system's core functions; failing to do so renders the system useless to users.
Should have included: This section assists in the identification of crucial parameters and the development of a universally useful key.
Could have been: This section assists in identifying them by outlining a few irrelevant demands that have no bearing on the system.
Won't Have: This system component makes it easy to identify demands that the system is unable to meet.

The engineering process includes iterative testing, system increments, system design, and ensuring that user needs are met in accordance with system criteria. During the Exploration phase, the MoSCoW prioritization technique is used to determine the system's functional and non-functional requirements. The core iterative component of the approach was utilized to iteratively evaluate the requirements and construct the solution.

4.4 Implementation Plans

Implementation plans outline the specific actions and steps required to execute a project or initiative successfully. These plans provide a roadmap for implementing the project, detailing the tasks, timelines, resources, and responsibilities involved. Here are some key elements typically included in an implementation plan:

1. Project Overview: This section provides a brief overview of the project, including its objectives, scope, and desired outcomes. It serves as a reminder of the project's purpose and helps set the context for the implementation plan.
2. Project Phases: The implementation plan outlines the different phases or stages of the project. Each phase represents a distinct set of activities and deliverables that contribute to the overall project completion. The plan clearly defines the sequence and duration of each phase.
3. Activities and Tasks: This section lists the specific activities and tasks that need to be performed during each project phase. It breaks down the project into manageable components, ensuring that all necessary actions are identified. The plan includes a description of each activity and its expected outcomes.
4. Timelines and Milestones: The implementation plan includes timelines for each activity and phase, specifying start and end dates. It also highlights key milestones or checkpoints to track progress and ensure that the project stays on schedule. Timelines may be presented in the form of Gantt charts or other visual representations.
5. Resources and Responsibilities: The plan identifies the resources required for each activity, including personnel, equipment, and materials. It also assigns responsibilities to team members or departments involved in the implementation. Clear roles and responsibilities help ensure accountability and coordination.

It's important to note that implementation plans should be dynamic documents that can be revised and updated as the project progresses. Regular monitoring, evaluation, and adjustments are necessary to ensure successful implementation.

Chapter 5 – Planning

5.1 Project Plan

A project plan is a comprehensive document that outlines the details and strategies for executing a specific project. It serves as a roadmap, providing guidance and direction to project teams on how to achieve project goals and deliverables. Here is a description of the key components typically included in a project plan:

1. Project Overview: The project plan begins with an overview that provides a summary of the project, including its objectives, scope, and desired outcomes. It outlines the purpose of the project and sets the context for the rest of the plan.
2. Project Goals and Deliverables: This section defines the specific goals and deliverables of the project. It outlines what the project aims to achieve and the tangible results that will be produced. Clear and measurable goals help to align the efforts of the project team and stakeholders.
3. Project Scope: The project plan clearly defines the boundaries and extent of the project. It outlines the inclusions and exclusions, specifying what is within the project's scope and what is not. This helps to manage expectations and prevent scope creep during the project execution.
4. Project Timeline: The project plan includes a timeline that outlines the start and end dates of the project, as well as major milestones and key activities. It helps to establish a schedule for completing various project tasks and provides a visual representation of the project's duration.
5. Project Organization and Roles: This section defines the project organization structure and identifies the roles and responsibilities of project team members and stakeholders. It outlines the reporting relationships, communication channels, and decision-making processes within the project team.
6. Project Resources: The project plan includes a description of the resources required to execute the project successfully. This includes human resources, such as project team members and external contractors, as well as physical resources, technology, and budgetary considerations.
7. Risk Management: The project plan addresses potential risks and includes a risk management strategy. It identifies potential risks that could impact the project's success and outlines mitigation strategies to minimize their impact. This helps to proactively manage and address potential issues that may arise during the project.
8. Project Communication: The project plan outlines the communication plan for the project. It identifies the key stakeholders and their communication needs, as well as the methods and frequency of communication. Effective communication ensures that project stakeholders are informed and engaged throughout the project lifecycle.
9. Project Quality Assurance: This section describes the quality assurance processes and measures that will be implemented to ensure the project's deliverables meet the required

standards. It outlines the quality control activities, such as inspections, testing, and reviews, that will be conducted to validate the project's outputs.

10. Project Monitoring and Evaluation: The project plan includes mechanisms for monitoring and evaluating the project's progress and outcomes. It defines the performance indicators or metrics that will be used to assess the project's success and outlines the methods and frequency of monitoring and evaluation activities.
11. Project Dependencies: The project plan identifies any dependencies or interdependencies with other projects, systems, or resources. It helps to identify potential bottlenecks or constraints that may impact the project's execution and allows for proactive management of these dependencies.
12. Project Change Management: The project plan includes a change management strategy that outlines how changes to project scope, requirements, or timelines will be managed. It defines the change control processes and the roles and responsibilities for reviewing and approving changes.

Overall, a project plan serves as a guiding document that provides a clear roadmap for project execution. It helps to align project team members, manage resources effectively, and ensure that project goals and objectives are achieved within the defined constraints.

5.1.1 Work Breakdown Structure

This phase comprises segmenting the project so that it can be completed on time while also being more efficient and effective. This structure gives us an estimate of time and task. The project may become more difficult to complete if this structure is not in place. As a result, the proposed system in the WBC has been divided into groups and subcategories, as illustrated in the chart below:

No.	Task Name	Start	Finish	Duration
1.	Introduction			
2.	Initial Study			
3.	Literature Review			
4.	Methodology			
5.	Planning			
6.	Feasibility Study			
7.	Foundation			
8.	Exploration & Engineering			

9.	Deployment			
10.	Testing			
11.	Implementation			
12.	Critical Appraisal & Evaluation			
13.	Lessons Learned			
14.	Conclusion			
15.	Total			

Figure 8: Medenico Patient Management System Work Breakdown Structure of the

5.1.2 IUMS Resource Allocation

All assets and resources are assigned and managed to guarantee that the desired project is accomplished on time and within budget. Resource allocation is a vital part of project planning. Because there is no team in this intellectual venture, I will take on many duties at different periods. The Southeast Website Design project's resource allocation is as follows in order to satisfy the previously indicated job delivery deadline.

Task	Duration	Resource
Introduction		Analyst, User
Initial Study		Analyst
Literature Review		Analyst
Methodology		Analyst,User
Planning		Analyst, Team Leader
Feasibility Study		Analyst, User
Foundation		Analyst, Designer, Developer
Exploration & Engineering		Analyst, Designer, Developer
Deployment		Designer, Developer
Testing		Developer, Tester, User
Implementation		Analyst, Developer, User

Critical Appraisal & Evaluation		Analyst
Lessons Learned		Analyst, Designer, Developer, Tester
Conclusion		Analyst

Table 3: Medenico Patient Management System Resource allocation list

5.1.3 Time Boxing

Another important part of DSDM project planning is the allocation of work into time boxes in order to meet deadlines on time. All jobs in this category are divided into timeboxes with set durations. These tasks must be done within the time constraints of the iterative process.

Time Box	Task Name	Duration	Resource
TB1	Introduction		Analyst, User
TB1	Initial Study		Analyst
TB1	Literature Review		Analyst
TB2	Methodology		Analyst,User
TB2	Planning		Analyst, Team Leader
TB2	Feasibility Study		Analyst, User
TB3	Foundation		Analyst, Designer, Developer
TB4	Exploration & Engineering		Analyst, Designer, Developer
TB5	Deployment		Designer, Developer
TB5	Testing		Developer, Tester, User
TB6	Implementation		Analyst, Developer, User
TB7	Critical Appraisal & Evaluation		Analyst
TB7	Lessons Learned		Analyst, Designer, Developer, Tester
TB8	Conclusion		Analyst

Table 4: Medenico Patient Management System List of the Time Boxes

5.1.4 Gantt Chart of Medenico Working

A Gantt Chart clearly illustrates the project's activity schedule. It shows the length as a progress bar from the beginning to the end date, rather than in days. Below is a Gantt chart for the medenico patient management system.

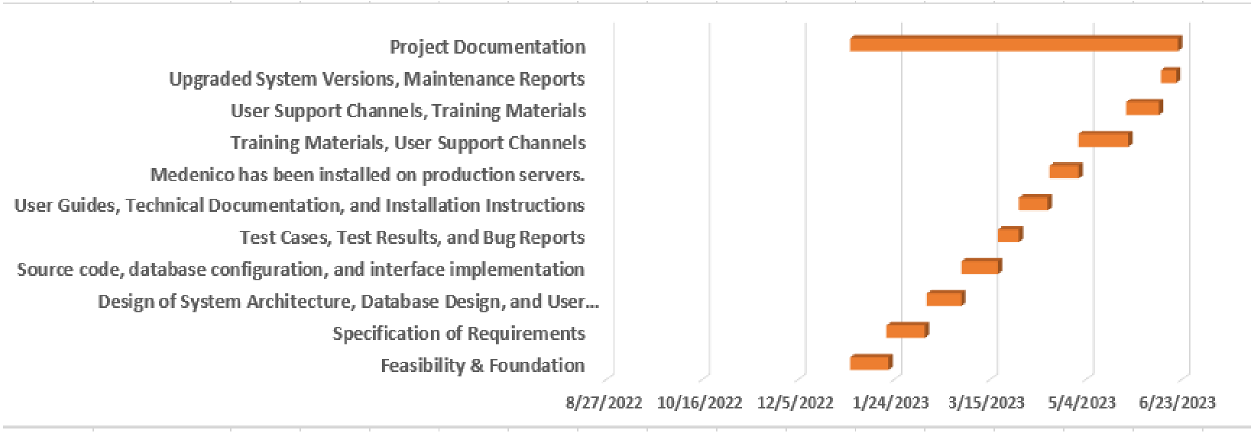


Figure 9: Medenico Project File cycle Gantt Chart

5.2 Test Plan

A Test Plan is a detailed document that outlines the test strategy, objectives, timetable, estimates, deadlines, and resources for a project. Consider it a strategy for doing the necessary tests to assure the program's functionality, with test managers in authority. The test plan was created in response to differences between the given input and the expected output of the system. During the software development process, testing, which included verification and validation, was performed.

5.2.1 Medenico Testing Against the Time Boxes

The time box method was used to calculate a fixed and maximum unit of time for a particular segment. Time constraints are put to the test-

Username for testing purpose	Example for testing	Role for testing	Example for testing
ID of the Time Box			
Content of the Time Box			

Type of Test	Steps for testing	Expected outcome	Actual outcomes	Comment
Unit testing				
Integration testing				
System testing				
Acceptance testing				
Security testing				
Usability testing				
Reliability testing				

Figure 10: Medenico Sample of Testing Against the Timebox

5.2.2 Required Test

Despite the fact that there are many more sorts of testing modules available, the two most prevalent types of system testing are functional and non-functional testing:

Functional Testing:

Functional testing is an important component of the software development life cycle that focuses on ensuring that a system, application, or website meets its functional requirements. It entails testing the software's different functions, features, and interactions to ensure that it acts as expected and provides the required functionality.

➢ Unit Testing: Unit testing is a type of software testing that focuses on testing individual units or components of a software system. The purpose of unit testing is to verify that each unit performs as expected and meets the specified requirements in isolation, independent of the other components.

Here are the key aspects of unit testing:

1. Test Class: A unit test is typically written as a test class that contains multiple test methods. Each test method focuses on testing a specific unit or function of the software system.

2. Source of Data: Unit tests use a variety of data sources, including test data that simulates different scenarios and boundary values, as well as mock objects or stubs to isolate the unit being tested from its dependencies.
3. Steps in the Testing Process: Unit testing follows a systematic process, which includes setting up the test environment, executing the unit test, and verifying the expected results against the actual results.
4. Expected Outcome: Unit tests define the expected outcome or behavior for each unit being tested. This includes both positive scenarios where the unit should produce the desired result and negative scenarios where the unit should handle errors or exceptions appropriately.
5. Actual Outcome: During the unit test execution, the actual outcome is compared to the expected outcome. If there is a mismatch, it indicates a potential defect or error in the unit being tested.

Unit testing helps identify defects early in the development cycle and promotes code quality and maintainability. It provides confidence that individual units function correctly before they are integrated into the larger system.

➢ Integration Testing: Integration testing is a software testing technique that focuses on testing the interactions between different components or modules of a software system. It aims to identify defects or issues that arise when the integrated components work together.

Here are the key aspects of integration testing:

1. Test Class: Integration testing involves creating test cases and test classes that specifically target the integration points between different components or modules.
2. Source of Data: Integration tests use test data that represents realistic scenarios and covers a range of inputs and conditions that the integrated system may encounter.
3. Steps in the Testing Process: Integration testing follows a systematic process, which includes identifying the integration points, preparing the test environment, executing the integration tests, and analyzing the results.
4. Expected Outcome: Integration tests define the expected outcomes based on the interactions between the integrated components. This includes verifying that data is passed correctly, interfaces are functioning as expected, and overall system behavior is consistent.
5. Actual Outcome: During the integration test execution, the actual outcome is compared to the expected outcome. Any discrepancies or issues that arise indicate potential defects in the integration or communication between components.

Integration testing helps uncover defects or issues that may arise when components are combined and interact with each other. It ensures that the integrated system functions as a cohesive unit and that the components work together as intended. By identifying and resolving integration issues early, integration testing helps mitigate risks and improve the overall quality and reliability of the software system.

➢ System Testing: System testing is a level of software testing that focuses on evaluating the complete and integrated system as a whole. It involves testing the system's behavior and functionality in accordance with the specified requirements.

Here are the key aspects of system testing:

1. Test Scope: System testing covers the entire software system, including all its components, modules, and interactions. It ensures that the system meets the functional and non-functional requirements and performs as expected.
2. Test Environment: System testing is typically performed in an environment that closely resembles the production environment. This includes using realistic data, configurations, and hardware/software setups to simulate real-world usage scenarios.
3. Test Cases: System testing involves creating and executing a set of test cases that cover a wide range of scenarios to validate the system's behavior and functionality. These test cases are designed to test the system as a whole, including all its integrated components.
4. Functional and Non-Functional Testing: System testing includes both functional and non-functional testing. Functional testing focuses on validating the system's features and functionalities, while non-functional testing assesses factors like performance, reliability, usability, security, and compatibility.
5. Regression Testing: System testing often includes regression testing to ensure that new changes or fixes do not introduce any defects or issues in the existing system functionality.
6. Test Execution and Analysis: System tests are executed in a structured manner, and the actual results are compared with the expected results. Any deviations or discrepancies are identified as defects and reported for further investigation and resolution.
7. System Validation: System testing aims to validate the system against the specified requirements, user expectations, and business objectives. It verifies that the system is ready for deployment and meets the desired quality standards.

System testing helps ensure that the entire software system works as intended and meets the defined requirements. It helps identify defects, performance issues, usability problems, and other potential risks that may arise in a real-world usage environment. By conducting comprehensive system testing, organizations can gain confidence in the quality, reliability, and functionality of their software system before it is deployed to users.

➢ Acceptance Testing: Acceptance testing is a type of software testing that is performed to determine whether a system meets the acceptance criteria and is ready for delivery to the end-users or stakeholders. It involves evaluating the system's compliance with user requirements, business objectives, and overall system functionality.

Here are the key aspects of acceptance testing:

1. Test Scope: Acceptance testing focuses on validating the system from the perspective of the end-users or stakeholders. It verifies that the system meets their needs, expectations, and specific requirements.
2. User Acceptance Testing (UAT): User acceptance testing is a common form of acceptance testing where end-users or representatives from the user community actively participate in testing the system. They perform real-life scenarios and provide feedback on the system's usability, functionality, and overall satisfaction.
3. Test Criteria: Acceptance testing is based on predefined acceptance criteria, which outline the expected behavior and performance of the system. These criteria are derived from the user requirements and may include functional, non-functional, and business-specific requirements.
4. Test Environment: Acceptance testing is typically performed in an environment that closely resembles the production environment to simulate real-world conditions and ensure accurate testing results.
5. Test Execution and Evaluation: Acceptance tests are executed based on test cases that cover various user scenarios and workflows. The actual results are compared against the expected results, and any discrepancies or issues are identified as defects or areas for improvement.
6. Sign-Off: Acceptance testing concludes with a formal sign-off process, where stakeholders evaluate the test results and provide approval or acceptance of the system. This signifies that the system meets their expectations and requirements.

Acceptance testing serves as a final validation of the system before its deployment. It helps ensure that the system is usable, reliable, and satisfies the needs of the end-users or stakeholders. By conducting thorough acceptance testing, organizations can gain confidence in the system's readiness for production use and mitigate the risk of delivering a system that does not meet user expectations.

Non-Functional Testing:

Non-functional testing is a sort of software testing that evaluates a system's performance, dependability, usability, and other non-functional features. Non-functional testing, as opposed to functional testing, evaluates the system's behavior and characteristics that are not directly related to its intended functionality.

- ➢ Security Testing:Security testing is a critical aspect of software testing that focuses on evaluating the security of a system or application. It involves identifying vulnerabilities, weaknesses, and potential risks in the software system's design, implementation, and configuration.

Here are the key aspects of security testing:

1. Threat Modeling: Security testing starts with a comprehensive understanding of the system's architecture, components, and potential threats. This helps in identifying the areas of the system that are more prone to security risks.
2. Vulnerability Assessment: Security testing involves conducting vulnerability assessments to identify any known security vulnerabilities or weaknesses in the system. This may include scanning for common vulnerabilities, such as cross-site scripting (XSS), SQL injection, or insecure authentication mechanisms.
3. Penetration Testing: Penetration testing, also known as ethical hacking, involves simulating real-world attacks on the system to uncover vulnerabilities and weaknesses. This includes attempting to exploit security vulnerabilities and gaining unauthorized access to the system.
4. Security Controls Testing: Security testing also verifies the effectiveness of implemented security controls, such as access controls, encryption mechanisms, authentication mechanisms, and audit logs. It ensures that these controls are functioning as intended and providing the necessary protection.
5. Security Compliance Testing: Security testing ensures that the system complies with relevant security standards, regulations, and best practices. This includes evaluating the system against industry-specific security requirements and ensuring the protection of sensitive data.
6. Security Incident Response Testing: Security testing also involves testing the system's response to security incidents or breaches. This includes evaluating the system's ability to detect, respond to, and recover from security incidents effectively.

By conducting comprehensive security testing, organizations can identify and address security vulnerabilities and risks in the software system. This helps in protecting the system and its data from unauthorized access, manipulation, and other security threats. It also helps in building trust among users and stakeholders by demonstrating a commitment to security and ensuring the confidentiality, integrity, and availability of the system.

- ➢ Usability Testing: Usability testing is a crucial part of software testing that focuses on evaluating the user-friendliness and effectiveness of a system or application from the user's perspective. It involves observing users performing specific tasks and collecting their feedback to assess the system's ease of use, efficiency, and overall user experience.

Here are the key aspects of usability testing:

1. Test Objectives: Usability testing aims to assess how easily users can accomplish specific tasks using the system and how satisfied they are with the overall user experience. The objectives may include evaluating the system's navigation, layout, content organization, clarity of instructions, and overall user satisfaction.
2. Test Participants: Usability testing involves recruiting representative users who fit the target user profile. The participants should have a reasonable understanding of the system's domain and its intended use. The number of participants may vary depending on the resources available, but typically involves a small group of users (e.g., 5-10).
3. Test Scenarios: Usability tests are conducted using predefined test scenarios that simulate real-life user interactions with the system. These scenarios focus on specific tasks or workflows that users are likely to perform. The scenarios are designed to cover a range of user actions and test different aspects of the system's usability.
4. Test Environment: Usability testing is typically conducted in a controlled environment that resembles the system's intended usage environment. The test environment should include the necessary hardware, software, and data required for the users to interact with the system.
5. Observation and Data Collection: Usability tests involve observing users as they interact with the system and perform the assigned tasks. Various methods can be used for data collection, including video recording, screen capture, observation notes, and user feedback through surveys or interviews.
6. Analysis and Feedback: The data collected from usability tests is analyzed to identify areas where users encountered difficulties, made errors, or expressed dissatisfaction. This feedback is used to improve the system's user interface, functionality, and overall usability.

Usability testing helps identify usability issues and ensures that the system is intuitive, easy to navigate, and meets the needs of its users. By involving users in the testing process, organizations can gather valuable insights to make informed design decisions and enhance the user experience. Ultimately, usability testing aims to optimize user satisfaction and increase user adoption of the system.

- ➢ Reliability Testing: Reliability testing is a type of software testing that focuses on assessing the stability, consistency, and dependability of a system or application. It involves testing the system under various conditions to determine its ability to perform its functions accurately and consistently over time.

Here are the key aspects of reliability testing:

1. Test Objectives: The main objective of reliability testing is to evaluate the system's ability to function reliably without failures or errors. It aims to identify any potential

reliability issues, such as crashes, data corruption, or system hang-ups, and ensure that the system meets the expected level of reliability.

2. Test Scenarios: Reliability tests are designed to simulate real-world usage scenarios and stress the system by subjecting it to a wide range of inputs, workloads, and environmental conditions. These scenarios may include testing the system's performance under heavy loads, extended usage periods, or concurrent user access.
3. Test Environment: Reliability testing requires a controlled testing environment that closely resembles the system's production environment. The test environment should include representative hardware, software, network configurations, and other infrastructure components.
4. Test Duration: Reliability tests are typically conducted over an extended period to assess the system's performance and stability over time. The duration of the tests depends on the specific requirements and objectives of the testing process.
5. Failure Analysis: During reliability testing, any failures or errors encountered are thoroughly analyzed to determine the root causes. This helps identify potential areas for improvement and allows for necessary fixes or optimizations to enhance the system's reliability.
6. Statistical Analysis: Reliability testing often involves statistical analysis to measure and predict the system's reliability metrics, such as mean time between failures (MTBF), mean time to failure (MTTF), or failure rate. This analysis provides quantitative data on the system's reliability performance.

Reliability testing is essential to ensure that the system can perform consistently and reliably under different conditions. By identifying and addressing potential reliability issues early in the development cycle, organizations can enhance the system's stability, reduce downtime, and improve the overall user experience. Reliability testing contributes to building trust in the system's performance and establishing its reputation for dependability and reliability.

5.2.3 Test Case of Medenico Patient Management System

No. of Test Case
Type of Test
Description of the test

Steps of testing	Expected outcome	Actual outcome	Comment

Table 5: Sample Test case

5.2.4 User Acceptance Test Plan

No. of Test Case
Type of Test
Description of the test
Preparation for testing
Name of the User
Assume the role of

Steps of testing	Expected outcome	Actual outcome	Comment

Table 6: Medenico User Acceptance Test Plan

5.3 Risk Management of Medenico

Risk management is a crucial process in the development and implementation of a Patient Management System (PMS). It involves identifying, assessing, and mitigating potential risks that could impact the system's functionality, security, usability, and overall success. Here are some key aspects of risk management for a PMS:

1. Risk Identification: The first step is to identify potential risks specific to the PMS. This includes considering risks related to data security breaches, system downtime, interoperability challenges, compliance with regulations, user adoption, and user error.
2. Risk Assessment: Once risks are identified, they need to be assessed in terms of their impact and likelihood. This involves determining the potential consequences of each risk and the probability of occurrence. The risks can be prioritized based on their severity and likelihood of occurrence.
3. Risk Mitigation: After assessing the risks, mitigation strategies should be developed to minimize their impact. This may involve implementing security measures to protect patient data, conducting regular backups to prevent data loss, implementing system redundancies for high availability, providing user training and support to reduce user errors, and ensuring compliance with relevant regulations and standards.
4. Risk Monitoring and Control: Risk management is an ongoing process, and risks need to be continually monitored and controlled throughout the lifecycle of the PMS. This includes regular risk assessments, monitoring system logs for potential security breaches

or failures, conducting audits to ensure compliance, and keeping up-to-date with emerging risks and industry best practices.

By effectively managing risks associated with a Patient Management System, healthcare organizations can minimize potential disruptions, protect patient data, ensure system availability, and improve overall patient care. A proactive approach to risk management reduces the likelihood of adverse events and helps in delivering a reliable and secure Medenico.

5.3.1 Risk Identification

Risk identification is the process of identifying potential risks that could impact the development, implementation, and operation of a Patient Management System (Medenico). Here are some common areas where risks can be identified in a Medenico:

1. Data Security: Risks related to data breaches, unauthorized access, data loss, or data corruption can significantly impact the confidentiality, integrity, and availability of patient information.
2. System Performance: Risks associated with poor system performance, slow response times, scalability limitations, or resource constraints can affect the system's ability to handle a large number of users or process high volumes of data efficiently.
3. Compatibility and Interoperability: Risks related to compatibility issues with existing systems or interoperability challenges with other healthcare systems can hinder seamless data exchange and integration.
4. User Adoption and Training: Risks related to user acceptance, user resistance, or lack of proper training and documentation can impact the successful adoption and utilization of the PMS by healthcare professionals and staff.
5. Regulatory Compliance: Risks associated with non-compliance with healthcare regulations, privacy laws (e.g., HIPAA), and industry standards can lead to legal and financial consequences for the healthcare organization.
6. Vendor and Supplier Risks: Risks related to the reliability, stability, and support of the PMS vendor or suppliers can impact the timely delivery, maintenance, and support of the system.
7. Change Management: Risks associated with managing changes and updates to the PMS, including software updates, system upgrades, or changes in workflows, can impact the system's stability, functionality, and user experience.
8. Disaster Recovery and Business Continuity: Risks related to natural disasters, hardware failures, or system disruptions that could result in prolonged system downtime or loss of patient data need to be identified and mitigated through proper disaster recovery and business continuity planning.

It is important to involve stakeholders, such as healthcare professionals, IT personnel, system administrators, and security experts, in the risk identification process to ensure a comprehensive assessment of potential risks. This helps in developing effective risk mitigation strategies and ensuring the successful implementation and operation of the Medenico.

Patient Management System (Medenico) can face various types of dangers, which can arise from different causes and have different consequences and impacts. Here are some examples:

Type of Dangers	Causes	Consequences & Impact
Data Breaches	➢ Weak security measures, lack of encryption, unauthorized access, hacking, insider threats.	➢ Exposure of sensitive patient information, potential identity theft, compromised privacy. Loss of patient trust, legal and regulatory penalties, damage to the reputation of the healthcare organization.
System Downtime	➢ Hardware or software failures, network issues, power outages, cyberattacks.	➢ Inability to access patient records, delays in treatment and care, disruption of healthcare services. Patient safety risks, reduced productivity, financial losses, negative impact on healthcare outcomes.
User Error	➢ Lack of training, human negligence, incorrect data entry, misinterpretation of information.	➢ Inaccurate patient data, incorrect diagnoses or treatments, compromised patient safety. Medical errors, compromised patient care, potential legal liabilities, damaged reputation.
Inadequate System Scalability	➢ Insufficient infrastructure resources, inadequate capacity planning, rapid growth in patient data.	➢ Slow system performance, delays in accessing patient information, reduced efficiency. Frustration among healthcare professionals, increased workload, compromised patient care quality.

Lack of Interoperability	➢ Incompatible systems, proprietary data formats, lack of standardized protocols.	➢ Difficulty in sharing patient information across different healthcare settings, data fragmentation. Inefficient care coordination, duplicated tests and procedures, compromised continuity of care.
Non-Compliance with Regulations	➢ Lack of awareness, inadequate policies and procedures, failure to adhere to privacy and security standards.	➢ Violation of healthcare regulations, legal penalties, loss of accreditation. Loss of patient trust, financial implications, potential shutdown of operations.

Table 7: Risk Identification of Medenico

5.3.2 Risk Assessment of Medenico

Type of risks	Likeliness	Impact	Restore time
Data Breaches	5	6	8
System Downtime	5	5	5
User Error	6	6	6
Inadequate System Scalability	4	5	5
Lack of Interoperability	5	6	4
Non-Compliance with Regulations	4	4	8

Table 8: Risk Assessment of Medenico

5.3.3 Risk Precaution / Action Plan

A risk action plan was developed after risk identification and evaluation. To prepare for risk, there are numerous measures that can be taken.
* To prevent existing dangers as well as those that may come in the future;
* To reduce risks that have already occurred as well as those that may arise in the future;

* To handle existing issues as well as potential hazards

Type of risks	Action	Taking Action	Action Required
Data Breaches(Implement strong security measures and encryption to protect patient data.)	Regularly update and patch software, use multi-factor authentication, conduct security audits, and train staff on data security best practices.	Implement a robust firewall and intrusion detection system, monitor access logs, and establish incident response protocols.	Regularly review and update security measures, conduct penetration testing, and ensure compliance with data protection regulations.
System Downtime(Implement redundancy and backup systems to minimize system downtime.)	Set up backup servers, power backups, and network redundancy.	Regularly test backup and recovery procedures, monitor system performance, and perform routine maintenance.	Develop a comprehensive disaster recovery plan, establish service level agreements (SLAs), and conduct regular system audits.
User Error(Provide comprehensive training and education to users.)	Conduct training sessions on system usage, data entry protocols, and error prevention.	Implement user-friendly interfaces, provide context-sensitive help, and establish user support channels.	Develop training materials, conduct regular refresher courses, and establish feedback mechanisms for user suggestions and concerns.
Inadequate System Scalability(Ensure the system is scalable to handle increasing data volumes and user demands.)	Conduct capacity planning and regularly assess system performance.	Optimize system resources, monitor system usage, and plan for future growth.	Conduct periodic scalability assessments, upgrade hardware and software as needed, and implement load balancing mechanisms.

Lack of Interoperability(Ensure the system can seamlessly exchange data with other healthcare systems.)	Implement standardized data formats and interoperability protocols.	Collaborate with other healthcare providers and IT vendors to establish data exchange standards.	Conduct interoperability testing, establish data sharing agreements, and participate in industry interoperability initiatives.
Non-Compliance with Regulations(Stay updated on healthcare regulations and privacy standards.)	Establish policies and procedures to ensure compliance.	Conduct regular audits, train staff on compliance requirements, and implement privacy and security measures.	Monitor regulatory changes, update policies accordingly, and engage legal and compliance experts for guidance.

Table 9:Medenico Risk Precaution

5.3.4 Steps Taken for Possible Risks of Medenico

Type of risks	Description	Likelines s	Impact	Taking Step
Data Breaches	Implementing strong security measures, such as encryption and access controls, to protect patient data from unauthorized access or disclosure.	Moderate	High	Conduct regular vulnerability assessments and penetration testing, enforce strict user authentication and authorization, implement data encryption, establish robust network security, and monitor access logs for suspicious activities.
System Downtime	Implementing redundant systems, backup mechanisms, and disaster recovery plans to minimize system downtime and ensure continuous availability.	Low	High	Implement redundant hardware components, backup servers, uninterruptible power supply (UPS), backup data centers, and establish a well-defined disaster recovery plan with regular testing and updates.

User Error	Providing comprehensive training and user support to minimize user errors in data entry and system usage.	Moderate	Medium	Conduct user training programs, provide user manuals and documentation, offer user support channels (e.g., helpdesk or knowledge base), implement user-friendly interfaces with validation and error-checking features, and establish user feedback mechanisms for continuous improvement.
Inadequate System Scalability	Planning and designing the system infrastructure to accommodate future growth and increasing data volumes.	High	Medium	Conduct thorough capacity planning, anticipate future growth and demands, implement scalable hardware and software solutions, monitor system performance, and regularly upgrade system components as needed.
Lack of Interoperability	Ensuring the system can exchange data seamlessly with other healthcare systems and adhere to interoperability standards.	Moderate	High	Adopt standardized data formats and interoperability protocols, collaborate with other healthcare organizations and vendors to establish data exchange agreements, participate in interoperability initiatives, and conduct interoperability testing.
Non-Compliance with Regulations	Adhering to healthcare regulations, privacy laws, and data protection standards to avoid legal and compliance issues.	High	High	Stay updated on healthcare regulations and privacy requirements, develop and enforce policies and procedures to ensure compliance, conduct regular audits, provide staff training on regulations, implement privacy and security measures, and engage legal and compliance experts for guidance.

Table 10: Risk dealing steps for Medenico

5.4 Change Management

5.4.1 Factors that Might Cause Change

There are several factors that can cause changes in a patient management system. These factors can arise from various sources and may impact different aspects of the system. Here are some common factors that might cause changes in a patient management system:

1. Regulatory Changes: Changes in healthcare regulations, privacy laws, or data protection requirements may necessitate updates or modifications to the patient management system to ensure compliance.
2. Technological Advancements: Advancements in technology, such as new medical devices, integration capabilities, or data exchange standards, may require updates to the patient management system to leverage the latest innovations.
3. User Feedback and Requirements: Feedback from users, including healthcare professionals and patients, can highlight areas for improvement or suggest new features that could enhance the system's functionality and user experience.
4. Organizational Changes: Changes within the healthcare organization, such as mergers, acquisitions, or restructuring, may require adjustments to the patient management system to accommodate new workflows, organizational structures, or information sharing requirements.
5. Scalability Needs: As the healthcare organization grows or expands its services, the patient management system may need to scale up to handle increased patient volumes, data storage, or user demands.

5.4.2 DSDM Welcome Change of Medenico

Dynamic Systems Development Method (DSDM) embraces and welcomes change in the development and management of the Patient Management System. DSDM is an Agile project management approach that recognizes the inevitability of change and provides a framework to effectively manage and accommodate change throughout the project lifecycle.

In DSDM, change is seen as an opportunity to deliver a better end product and to respond to evolving business needs and user requirements. The following aspects of DSDM demonstrate its support for change:

1. Iterative and Incremental Development: DSDM promotes an iterative and incremental approach to development, where the system is built and delivered in small increments or iterations. This allows for frequent feedback, early user involvement, and the ability to incorporate changes based on user input and evolving needs.
2. Timeboxing: DSDM employs timeboxing, which involves fixing the time duration for each development iteration or phase. This time constraint encourages prioritization,

focusing on the most valuable features, and facilitates the accommodation of changes within the given timeframe.

3. Iterative Requirements and Solution Exploration: DSDM recognizes that requirements may change or evolve as the project progresses. It encourages the exploration of requirements in an iterative manner, allowing for continuous refinement, clarification, and adjustment of requirements based on feedback and changing circumstances.
4. Collaborative and Empowered Teams: DSDM emphasizes collaborative and empowered teams that include both business stakeholders and development professionals. This ensures that all stakeholders have a voice in the project and can contribute to shaping the system, including suggesting and incorporating changes as needed.
5. Change Control and Prioritization: DSDM provides mechanisms for effective change control and prioritization. Changes are assessed based on their impact, value, and feasibility, allowing the project team to make informed decisions about incorporating changes while managing potential risks and constraints.
6. Continuous Business Engagement: DSDM promotes ongoing business engagement throughout the project. This means that stakeholders are actively involved in providing feedback, reviewing progress, and suggesting changes as necessary, ensuring that the system remains aligned with their needs and expectations.

By embracing change, DSDM enables the Patient Management System project to be flexible, adaptive, and responsive to evolving requirements, market conditions, and user feedback. It ensures that the system remains relevant, effective, and valuable throughout its development and beyond.

5.4.3 Considering Business Priority

Considering business priority is a crucial aspect of project management, especially when developing a Patient Management System. It involves understanding and aligning the project objectives and deliverables with the strategic goals and priorities of the healthcare organization. By considering business priority, the project team can focus on delivering the most valuable features and functionalities that have the highest impact on the business.

Priorities were as follows:

Topical Area	Priority Level
Training Segments	6
Functional requirements	8
Non-Functional requirements	6

Segments for ratings and reviews	5
Segments for security handling	8

5.4.4 Change Workshop

A change workshop in the context of a Patient Management System involves bringing together key stakeholders and project team members to discuss and manage changes related to the system. It aims to facilitate effective communication, collaboration, and decision-making to ensure smooth implementation of changes in the healthcare organization. Here's how a change workshop can be conducted for a Patient Management System:

1. Define the Objectives: Clearly define the objectives of the change workshop. It could be to discuss specific changes or enhancements to the Patient Management System, address challenges or pain points, or explore opportunities for improvement.
2. Identify Stakeholders: Identify the relevant stakeholders who will be impacted by the changes in the Patient Management System. This may include healthcare professionals, administrators, IT staff, regulatory authorities, and patient representatives. Ensure their active participation in the workshop.
3. Pre-workshop Preparation: Share relevant information about the proposed changes with the participants prior to the workshop. This can include documentation, project updates, impact assessments, or user feedback. Ask participants to review the materials beforehand to come prepared for discussions.
4. Facilitate Discussions: Designate a skilled facilitator who will lead the change workshop. The facilitator should have a good understanding of the Patient Management System, change management principles, and effective workshop facilitation techniques. Encourage open discussions, active participation, and respectful exchange of ideas.
5. Present Proposed Changes: Provide a clear overview of the proposed changes to the Patient Management System. Discuss the reasons behind the changes, expected benefits, and potential impacts on various stakeholders. Use visual aids, demonstrations, or prototypes to help participants visualize the changes.

By conducting a change workshop for the Patient Management System, you can engage stakeholders, gather valuable insights, and ensure a collaborative approach to implementing changes. It helps in managing expectations, addressing concerns, and fostering a positive environment for change within the healthcare organization.

5.4.5 Changes That are Allowed

In a patient management system, various types of changes can be allowed depending on the specific needs and requirements of the healthcare organization. Here are some common changes that are typically allowed in a patient management system:

1. System Enhancements: These changes involve adding new features, functionalities, or capabilities to the patient management system to improve its efficiency, usability, or effectiveness. For example, adding a scheduling module, integrating electronic health records (EHRs), or implementing a patient portal for online appointment booking.
2. Workflow Modifications: Changes in workflows aim to streamline and optimize the processes involved in patient management. This may include redefining the steps for patient registration, appointment scheduling, billing, or prescription management. The goal is to improve the overall workflow efficiency and reduce manual errors.

It's important to note that any changes introduced in a patient management system should be thoroughly evaluated, tested, and validated to ensure they meet the requirements, do not compromise patient safety or data integrity, and align with the organization's policies and guidelines. Regular maintenance and updates are essential to keep the system up-to-date and aligned with evolving healthcare needs.

5.4.6 Key Decision Taker of Change

The key decision-makers involved in making changes to a patient management system may vary depending on the healthcare organization's structure and processes. However, here are some common stakeholders who typically play a crucial role in decision-making related to changes in a patient management system:

1. Healthcare Executives/C-Suite: Top-level executives, such as CEOs, CIOs, or CFOs, often have the authority to make strategic decisions regarding the patient management system. They provide overall direction and prioritize changes based on the organization's goals, budget, and strategic objectives.
2. IT Management: IT managers and department heads are responsible for overseeing the technical aspects of the patient management system. They assess the feasibility and impact of proposed changes, coordinate implementation efforts, and ensure alignment with IT strategies and infrastructure.
3. Clinical Staff: Physicians, nurses, and other healthcare professionals who directly use the patient management system have valuable insights and expertise. They provide input on workflow improvements, system usability, and the impact of changes on patient care. Their feedback is crucial in decision-making regarding changes that impact clinical workflows.

4. Project Managers: Project managers oversee the implementation of changes in the patient management system. They coordinate project activities, manage timelines, allocate resources, and ensure successful delivery of change initiatives. They work closely with other stakeholders to gather requirements, analyze risks, and monitor progress.

5.5 Quality Management

Quality management in a patient management system involves a systematic approach to ensuring the delivery of a high-quality, reliable, and efficient system that meets the needs of healthcare providers, patients, and other stakeholders. Here are some key aspects of quality management in a patient management system:

1. Requirements Gathering and Analysis: A crucial step in quality management is gathering and analyzing the requirements of the healthcare organization and end-users. This involves engaging with stakeholders to understand their needs, documenting requirements, and ensuring clarity and completeness. Proper requirements analysis helps set the foundation for a system that aligns with user expectations and business objectives.
2. Quality Planning: Quality planning involves defining the quality objectives, standards, and metrics for the patient management system. It includes determining the appropriate quality assurance and quality control processes to be followed throughout the project lifecycle. Quality planning also includes establishing quality criteria for acceptance testing and defining the criteria for measuring system performance.
3. Quality Assurance: Quality assurance activities focus on preventing defects and ensuring that the patient management system is developed and implemented in accordance with established standards and best practices. It involves conducting regular audits, reviews, and inspections to assess adherence to quality processes, standards, and guidelines. Quality assurance also involves implementing proper configuration management and change control processes to track and manage system changes effectively.
4. Testing and Validation: Testing plays a crucial role in quality management. It includes various levels of testing, such as unit testing, integration testing, system testing, and acceptance testing, to ensure that the patient management system functions as intended and meets the specified requirements. Testing helps identify and rectify any defects, errors, or inconsistencies in the system before it is deployed.
5. Documentation and Training: Comprehensive and accurate documentation is essential for quality management. This includes documenting system requirements, design specifications, test plans, user manuals, and other relevant documentation. Proper documentation ensures that the system can be effectively maintained, supported, and updated. Training healthcare staff on how to effectively use the patient management system is also crucial to ensure its proper utilization and to maximize its benefits.

5.5.1 Rules Applied to Maintain Quality

To maintain quality in a patient management system, several rules and practices can be applied. Here are some key rules commonly followed:

1. Adherence to Standards and Best Practices: Following industry-recognized standards and best practices ensures that the patient management system is designed, developed, and implemented in accordance with established guidelines. This includes adhering to coding standards, data exchange formats (such as HL7), security protocols, and regulatory compliance requirements (e.g., HIPAA).
2. Continuous Testing and Validation: Regular testing at different stages of the system development life cycle is essential to identify and rectify any defects or issues. This includes unit testing, integration testing, system testing, and acceptance testing. It ensures that the system functions as intended, meets the specified requirements, and performs reliably in real-world scenarios.
3. Documentation and Change Control: Maintaining comprehensive documentation, including system requirements, design specifications, test plans, and user manuals, helps ensure clarity and consistency throughout the development and maintenance processes. Change control processes should be implemented to manage system updates and modifications effectively, ensuring that changes are properly documented, tested, and approved.
4. User-Centric Design and Usability: Designing the patient management system with a focus on user experience and usability is crucial for maintaining quality. The system should be intuitive, easy to navigate, and tailored to meet the needs of healthcare providers, administrators, and patients. Incorporating user feedback and conducting usability testing helps identify and address any usability issues.

By applying these rules and practices, healthcare organizations can maintain the quality and reliability of their patient management system, leading to improved patient care, streamlined workflows, and enhanced efficiency in healthcare operations.

5.5.2 DSDM Standard Quality Measures

DSDM (Dynamic Systems Development Method) provides a set of standard quality measures that can be applied to ensure the quality of a patient management system. These measures focus on various aspects of the system to ensure its reliability, usability, maintainability, and adherence to business requirements. Here are some DSDM standard quality measures applicable to a patient management system:

1. Fitness for Purpose: The patient management system should meet the defined business objectives and user needs. This measure assesses whether the system fulfills its intended purpose and aligns with the goals of the healthcare organization.
2. Stakeholder Satisfaction: This measure evaluates the satisfaction levels of the system's stakeholders, including healthcare providers, administrators, and patients. It assesses whether the system meets their expectations, provides value, and improves the overall experience.
3. Functional Adequacy: This measure focuses on the functionality of the patient management system. It ensures that all required features and functions are implemented correctly, and they meet the specified requirements. Functional adequacy considers the accuracy, completeness, and reliability of the system's functionalities.
4. Ease of Use: Usability is an important measure for a patient management system. It assesses how easy and intuitive the system is to use for various user roles. It includes factors such as user interface design, navigation, data entry processes, and overall user experience.
5. Performance Efficiency: This measure evaluates the performance and efficiency of the system. It considers factors such as response time, system throughput, and resource utilization. Performance efficiency ensures that the patient management system operates smoothly and meets the required performance criteria.

By considering these DSDM standard quality measures, healthcare organizations can assess and monitor the quality of their patient management system throughout its development and maintenance lifecycle. These measures help ensure that the system meets the required standards, satisfies stakeholder needs, and delivers value in the healthcare environment.

5.5.3 Quality Plan and Measuring Meter

A quality plan in a patient management system outlines the strategies and activities that will be implemented to ensure the quality of the system throughout its development, implementation, and maintenance phases. It includes specific measures and metrics to assess the quality of the system at various stages. Here are some elements that can be included in a quality plan for a patient management system:

1. Quality Objectives: Clearly define the quality objectives of the patient management system, such as achieving a certain level of system performance, usability, security, and data integrity. These objectives should align with the organization's overall goals and stakeholder expectations.
2. Quality Assurance Activities: Specify the activities that will be performed to ensure quality throughout the system development process. This may include code reviews, design reviews, documentation reviews, and adherence to coding standards and best

practices. Quality assurance activities ensure that quality is built into the system from the start.

3. Testing and Validation: Outline the testing strategies and approaches to be used for the patient management system. This includes unit testing, integration testing, system testing, and user acceptance testing. Define the test cases, test scenarios, and acceptance criteria to measure the system's compliance with functional and non-functional requirements.
4. Quality Metrics: Define specific metrics and measurement techniques to evaluate the quality of the patient management system. These metrics may include system performance indicators, such as response time and throughput, error rates, system availability, data accuracy, and user satisfaction surveys. Collecting and analyzing these metrics provide insights into the system's quality and identify areas for improvement.
5. Defect Management: Describe the process for identifying, reporting, and addressing defects or issues in the patient management system. This includes establishing a defect tracking system, assigning priorities and severities to reported issues, and implementing a process for timely resolution.

By implementing a quality plan and utilizing appropriate measuring metrics, healthcare organizations can ensure that their patient management system meets the required standards, delivers the expected functionality, and provides a high level of performance, usability, and data integrity. Regular monitoring, measurement, and improvement efforts help maintain and enhance the overall quality of the system over time.

Chapter 6 – Feasibility

6.1 All Possible Types of Feasibility Medenico

When considering the feasibility of a patient management system, several types of feasibility need to be assessed to determine the viability of the project. Here are the different types of feasibility to consider for a patient management system:

Technical Feasibility: This assesses whether the necessary technology infrastructure and capabilities exist to develop and implement the patient management system. It examines factors such as the availability of hardware, software, network connectivity, and the compatibility of the system with existing IT infrastructure.
Hardware:
• HP laptop (Configuration)
• Wi-fi Router(Tp-link)
Software:
• Xampp
• Microsoft office(MS)
• Microsoft Excel
• Google chrome browser
• Windows 10 Pro (operating system)
•VS Code
• Brackets
Database:
• MySQL
Technology:
UI/UX Design:
•Lucidchart & Draw.io for High Fidelity Prototype
•Figma (Design)
Design Slide:
• Html
• CSS
• Java Script
• Bootstrap
Server Side:
• PHP
• Laravel 8.6
Economic Feasibility: This evaluates the financial viability of the patient management system. It involves analyzing the costs associated with system development, implementation,

maintenance, and operational expenses. Economic feasibility considers factors such as return on investment (ROI), cost-benefit analysis, and the affordability of the system for the healthcare organization.

➢Web based application cost:

Equipment	Cost per unit	Cost
VPN connectivity to an extranet network	□3000 per/m	□3,000
Laptop (core i3, 2.50-271 GHz processor, 4 GB DDR4 RAM, 1 TB HDD)	□ 86,000	□ 86,000
File and Email and cloud servers	□18,000 per/m	□18,000 per/m
Total		□ 1,07,000

➢Desktop Application cost:

Equipment	Cost per unit	Cost
Laptop (core i3, 2.50-271 GHz processor, 4 GB DDR4 RAM, 1 TB HDD)	□ 86,000	□ 86,000
File and Email and cloud servers	□16,000 per/m	□16,000 per/m
Total		□ 1,02,000

Medenico Market Research Analysis Based on the Feasibility Factors:

To conduct a market research analysis for a patient management system based on feasibility factors, the following steps can be taken:

1. Define Feasibility Factors: Identify the feasibility factors that are relevant to the patient management system, such as technical feasibility, economic feasibility, operational feasibility, legal and regulatory feasibility, schedule feasibility, and organizational feasibility.
2. Gather Market Data: Collect market data related to the patient management system industry, including market size, growth rate, trends, and key players. This can be done through market research reports, industry publications, online databases, and surveys.
3. Analyze Competitors: Identify and analyze existing patient management system providers in the market. Evaluate their offerings, features, pricing, customer reviews, and market share. Assess how well they meet the feasibility factors and identify any gaps or opportunities.
4. Assess Target Customers: Understand the needs, preferences, and pain points of the target customers for the patient management system. Conduct surveys, interviews, and focus groups with healthcare professionals, administrators, and IT personnel to gather insights.
5. Evaluate Technical Feasibility: Assess the technological landscape and infrastructure of the market. Evaluate the availability and compatibility of hardware, software, and network infrastructure required for a patient management system. Identify any emerging technologies or advancements that may impact the feasibility.

By conducting a thorough market research analysis based on feasibility factors, healthcare organizations can gain valuable insights into the market potential and feasibility of a patient management system. This analysis helps in making informed decisions regarding product development, market positioning, pricing strategies, and target customer segments.

Patient Management System Research Analysis Based on the Feasibility Factors:

Feasibility analysis is an important step in assessing the viability of a Patient Management System (Medenico) project. It involves evaluating various factors to determine if the project is feasible and can be successfully implemented. Here are some feasibility factors to consider in the research analysis for a Medenico:

1. Technical Feasibility:
    - Assess the availability of technology and infrastructure required for the PMS.
    - Evaluate the compatibility of the proposed system with existing systems and software.
    - Consider the technical expertise and resources needed for development and maintenance.
2. Economic Feasibility:
    - Conduct a cost-benefit analysis to determine the financial viability of the PMS.

   - Estimate the initial investment required for development, implementation, and training.
   - Analyze the potential cost savings and benefits that the PMS can bring to the healthcare organization.
3. Operational Feasibility:
   - Evaluate the impact of the PMS on the day-to-day operations of the healthcare facility.
   - Assess the readiness and willingness of the staff to adopt and utilize the new system.
   - Consider any potential disruptions or challenges that may arise during implementation.
4. Legal and Regulatory Feasibility:
   - Ensure compliance with applicable laws, regulations, and industry standards.
   - Identify any legal or regulatory requirements specific to patient data privacy and security.
   - Evaluate the potential impact of legal and regulatory factors on the implementation of the PMS.
5. Schedule Feasibility:
   - Determine the project timeline and evaluate if it is achievable.
   - Consider any dependencies, resource constraints, or potential delays that may affect the schedule.
   - Assess the availability of key stakeholders and resources required for timely implementation.
6. Organizational Feasibility:
   - Assess the organizational culture and readiness for change.
   - Evaluate the support and commitment from management and stakeholders.
   - Consider the organizational structure and potential impact on roles and responsibilities.

By conducting a comprehensive feasibility analysis based on these factors, you can assess the viability and potential success of implementing a Patient Management System. It helps in making informed decisions, identifying potential challenges, and determining the necessary steps for a successful implementation.

6.2 Cost Benefit Analysis for Patient Management System:

Total cost:

Serial No.	Equipment	First Year	Second Year	Third Year	Fourth Year	Five Year	Total

1	Web based application cost	□ 1,04,000	-	-	-	-	□ 1,04,000
2	Desktop Application cost	1,02,000	-	-	-	-	1,02,000
3	The cost of a domain plus hosting	25,000	25,000	25,000	25,000	25,000	125,000
4	Expenses for Employees	40,000	40,000	40,000	40,000	40,000	2,00000
5	Other Cost	20,000	20,000	20,000	20,000	20,000	1,00000
6	Total Cost	291,000	85,000	85,000	85,000	85,000	631,000

Table 11: Total Cost Estimation for the project Medenico

Earning Sector:

Serial No.	Earn Sector	First Year	Second Year	Third Year	Fourth Year	Five Year	Total
1	Govt. Tax one	1,50,000	1,80,000	2,10,000	2,40,000	2,70,000	1,050,000
2	Govt. Tax two	2,00000	2,10,000	2,40,000	2,60,000	3,00000	1,210,000
	Total	3,50,000	3,90,000	4,50,000	5,00000	5,70,000	2,260,000

Table 12: Earning estimation for the project Medenico

Total Revenue of project:

Serial No.	Sector	First Year	Second Year	Third Year	Fourth Year	Five Year	Total
1	Total Earning	3,50,000	3,90,000	4,50,000	5,00000	□,70,000	2,260,000
2	Total Equipment Cost	4,02,000	80,000	80,000	80,000	80,000	7,22,0000
	Total Revenue	4,02,000	3,10,000	3,70,000	4,20,000	4,90,000	1,538,000

Table 13: Medenico Estimated Revenue on a Five-year scale

6.3 Explain DSDM Good or Bad for this Project

DSDM (Dynamic Systems Development Method) is an agile project management and delivery framework that focuses on delivering high-quality software solutions within fixed time and resource constraints. Whether DSDM is good or bad for a Patient Management System (PMS) project depends on various factors and considerations. Here are some points to consider:

Benefits of DSDM for a PMS project:

1. Agile and Iterative Approach: DSDM promotes iterative development and frequent feedback, allowing for early and continuous delivery of working software. This is beneficial for a PMS project as it enables rapid prototyping, user involvement, and quick validation of system functionality.
2. Focus on User Needs: DSDM emphasizes active user involvement throughout the development process. For a PMS project, involving healthcare professionals, administrators, and patients in the development process can ensure that the system meets their specific needs and requirements.
3. Timeboxing: DSDM uses timeboxing to set fixed timeframes for project phases, iterations, and deliverables. This can be advantageous for a PMS project as it helps manage the project schedule, prioritize features, and ensure timely delivery of critical functionalities.
4. Flexibility and Adaptability: DSDM encourages flexibility and adaptation to changing requirements. This is important for a PMS project as healthcare environments and regulations can evolve over time, and the ability to quickly respond to changes is essential.

Drawbacks of DSDM for a PMS project:

1. Complex Project Management: DSDM requires skilled project management and coordination to effectively manage the iterative development approach, stakeholder involvement, and changing requirements. Without proper project management expertise, the project may face challenges in maintaining schedule and scope.
2. Limited Documentation: DSDM emphasizes working software over comprehensive documentation. While this can be beneficial for quick delivery, it may result in inadequate documentation, which can be a concern for a PMS project that requires thorough documentation for compliance and regulatory purposes.
3. Potential Resistance to Change: Implementing DSDM in a healthcare organization may require a cultural shift and a willingness to embrace agile principles. Resistance to change from stakeholders and employees can hinder the successful adoption and implementation of DSDM in a PMS project.

In conclusion, DSDM can be a good approach for a Patient Management System project if the organization is open to agile methodologies, values user involvement, and can effectively manage the complexities associated with iterative development and changing requirements. However, careful consideration should be given to the project's specific context, team capabilities, and organizational readiness before choosing DSDM as the project management approach.

Chapter 7 – Foundation

7.1 The Problem Area Identification

Identifying problem areas in a Patient Management System (PMS) involves recognizing the challenges or shortcomings that exist within the current system or process. Here are some common problem areas that organizations may encounter in a PMS:

1. Manual and Paper-based Processes: Many healthcare facilities still rely on manual and paper-based processes for patient management, which can be time-consuming, error-prone, and inefficient. These processes may include paper-based medical records, appointment scheduling, and prescription management.
2. Lack of Integration: Inefficient integration between different components of the PMS, such as electronic health records (EHR), billing systems, and appointment scheduling, can lead to data duplication, inconsistencies, and difficulty in accessing accurate and up-to-date patient information.
3. Limited Accessibility: Accessibility issues can arise when the PMS is not designed to accommodate different user roles and devices. For example, if the system is not accessible from mobile devices or lacks multilingual support, it can hinder the efficient management of patient information, especially in diverse healthcare settings.
4. Data Security and Privacy Concerns: Patient data security and privacy are paramount in a PMS. Inadequate security measures, such as weak authentication protocols, lack of encryption, or insufficient access controls, can lead to unauthorized access, data breaches, and violations of patient privacy rights.
5. Inefficient Workflow and Communication: Poor workflow and communication within the PMS can result in delays, confusion, and errors in patient care. For instance, if there is no streamlined process for sharing information between healthcare providers, it can lead to miscommunication and potential gaps in patient treatment.
6. Lack of Scalability and Performance: As healthcare organizations grow and patient volumes increase, the PMS should be able to scale and handle the growing demands. Performance issues, such as slow response times or system crashes, can hinder the efficient management of patient data and impact overall productivity.
7. Compliance and Regulatory Challenges: Healthcare organizations must comply with various regulations and standards, such as HIPAA (Health Insurance Portability and Accountability Act) in the United States. Failure to meet these requirements can result in legal consequences and reputational damage.

Identifying these problem areas in a PMS is crucial for understanding the areas that require improvement and determining the necessary steps to address them. By addressing these challenges, healthcare organizations can enhance patient care, improve operational efficiency, and ensure compliance with industry standards and regulations.

7.1.1 Interview

1. Interview with Admin:

- Can you briefly describe your role as an admin in the healthcare organization?
- What are the main administrative tasks and responsibilities you handle related to patient management?
- How do you currently manage patient registrations and appointments?
- What challenges do you face in terms of maintaining accurate patient records and information?
- Are there any specific functionalities or features you would like to see in the Patient Management System to assist you in your administrative tasks?

2. Interview with User (Healthcare Provider):

- Can you briefly describe your role as a healthcare provider within the organization?
- What are the main tasks and responsibilities you handle in terms of patient care and treatment?
- How do you currently access patient information and medical records?
- What challenges do you face in terms of accessing and updating patient data during your interactions with patients?
- Are there any specific features or functionalities you would like to see in the Patient Management System to assist you in providing better care to patients?

3. Interview with Patient:

- Can you briefly describe your experience as a patient within the healthcare organization?
- What are the main concerns or challenges you face when it comes to managing your appointments and accessing your medical information?
- How do you currently schedule appointments and interact with the healthcare organization?
- Are there any difficulties you encounter in terms of understanding your medical records or accessing test results?
- Are there any specific functionalities or features you would like to have in the Patient Management System as a patient?

7.1.2 Observations

Observations for the Patient Management System can vary depending on the specific context and the stage of development or implementation. However, here are some common observations that can be made during the evaluation of a Patient Management System:

1. User Interface (UI) Design: Evaluate the usability and intuitiveness of the system's user interface. Look for clear navigation, consistent design elements, and user-friendly interactions.
2. Registration and Appointment Management: Assess how efficiently the system handles patient registration, appointment scheduling, and rescheduling. Look for features that minimize errors and facilitate smooth patient flow.
3. Patient Information Management: Evaluate how well the system manages patient data, including demographics, medical history, diagnoses, and treatment plans. Check for data accuracy, completeness, and ease of updating or accessing patient information.
4. Appointment Reminders and Notifications: Assess the system's ability to send automated appointment reminders to patients through various communication channels, such as SMS or email. Evaluate the effectiveness of these reminders in reducing no-shows and improving patient attendance.
5. Integration with External Systems: If applicable, observe how well the Patient Management System integrates with other healthcare systems, such as electronic health records (EHR) or laboratory systems. Evaluate the seamless flow of data and the efficiency of data exchange.
6. Reporting and Analytics: Evaluate the system's reporting capabilities, such as generating various types of reports for administrators and healthcare providers. Assess the availability of key performance indicators (KPIs) and analytics to support decision-making and performance monitoring.
7. Privacy and Security: Assess the system's adherence to privacy regulations and security standards. Look for features such as role-based access control, data encryption, and audit trails to ensure the protection of patient data.
8. Support and Training: Evaluate the availability and quality of technical support provided by the system vendor or IT team. Assess the effectiveness of user training programs to ensure healthcare staff can fully utilize the system's features.
9. Scalability and Performance: Consider the system's ability to handle increasing patient volumes and support multiple users simultaneously without significant performance degradation.
10. Feedback from Users: Gather feedback from administrators, healthcare providers, and patients who have used the system. Consider their experiences, suggestions, and any identified areas for improvement.

These observations can help identify strengths, weaknesses, and areas for enhancement in the Patient Management System, leading to further improvements and optimization of the system's functionalities.

7.1.3 Questionnaires

Gathering information is crucial in project management for making effective judgments. There are numerous methods for gathering information from various types of people. Project managers use observation, focus group talks, supported lectures, and benchmarking often.

Identification of Problems Questions

Name		**Age**	
General User	Admin	Problem	

Question-one	Can you briefly describe your role as an admin in the healthcare organization?
Answer	
Question-two	What are the main administrative tasks and responsibilities you handle related to patient management?
Answer	
Question-three	How do you currently manage patient registrations and appointments?
Answer	
Question-four	What challenges do you face in terms of maintaining accurate patient records and information?
Answer	
Question-five	Are there any specific functionalities or features you would like to see in the Patient Management System to assist you in your administrative tasks?
Answer	

Identification of Problems Questions

Name		**Age**	
General User	User (Healthcare Provider)	Problem	

Question-one	Can you briefly describe your role as a healthcare provider within the organization?
Answer	
Question-two	What are the main tasks and responsibilities you handle in terms of patient care and treatment?
Answer	
Question-three	How do you currently access patient information and medical records?
Answer	
Question-four	What challenges do you face in terms of accessing and updating patient data during your interactions with patients?
Answer	
Question-five	Are there any specific features or functionalities you would like to see in the Patient Management System to assist you in providing better care to patients?
Answer	

Identification of Problems Questions

Name		**Age**	
General User	Patient	Problem	

Question-one	Can you briefly describe your experience as a patient within the healthcare organization?
Answer	
Question-two	What are the main concerns or challenges you face when it comes to managing your appointments and accessing your medical information?
Answer	
Question-three	How do you currently schedule appointments and interact with the healthcare organization?
Answer	
Question-four	Are there any difficulties you encounter in terms of understanding your medical records or accessing test results?
Answer	
Question-five	Are there any specific functionalities or features you would like to have in the Patient Management System as a patient?
Answer	

7.2 Rich Picture

A rich picture is a top-down or bird's-eye view of a system's users' actions. It also depicts diametrically opposed concerns, as well as communication and business techniques. The IUMS is shown in more detail below: -

Figure 11: Rich Picture of the Medenico

Medenico The legends of rich picture

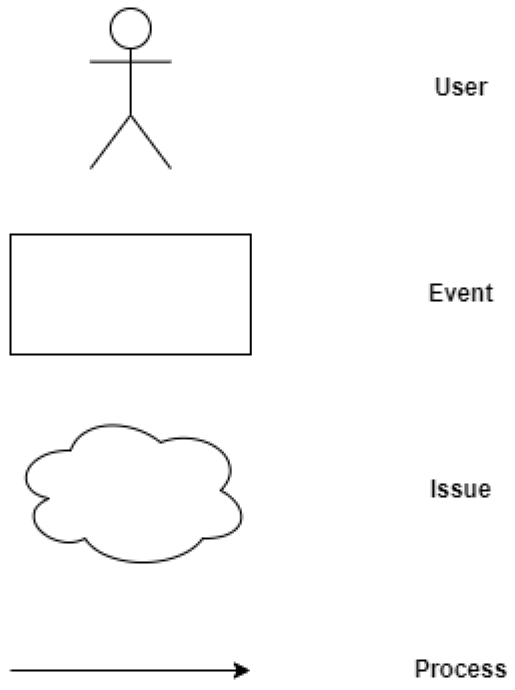


Figure 12: IUMS Legends of the rich picture

Key actors
In the patient management system, there are two different sorts of actors.

➢Admin

➢Patient

7.3 Specific Problem Area Identification

Specific problem areas in a Patient Management System can vary depending on the specific implementation and user requirements. However, here are some common problem areas that may arise in a Patient Management System:

1. User Interface Complexity: The system may have a complex user interface that makes it difficult for users to navigate and perform tasks efficiently. This can lead to user frustration and errors in data entry or retrieval.
2. Data Entry and Data Quality: Users may face challenges in entering patient information accurately and maintaining data quality. This can result in incomplete or inaccurate patient records, leading to potential errors in diagnosis or treatment.
3. Appointment Scheduling and Management: The system may lack flexibility and ease of use in scheduling and managing patient appointments. This can result in scheduling conflicts, missed appointments, and inefficiencies in managing the patient flow.

4. Integration with External Systems: The system may have limited or inadequate integration capabilities with other healthcare systems, such as electronic health records or laboratory systems. This can hinder seamless data exchange and coordination of patient care.
5. Lack of Customization and Scalability: The system may not offer sufficient customization options to adapt to the unique needs of different healthcare settings. Additionally, it may face scalability challenges when handling a large volume of patient data or supporting multiple users simultaneously.
6. Inadequate Reporting and Analytics: The system may lack robust reporting and analytics capabilities, making it difficult for administrators and healthcare providers to generate meaningful insights and monitor key performance indicators.
7. Security and Privacy Concerns: The system may have vulnerabilities in terms of data security and privacy, potentially exposing patient information to unauthorized access or breaches. This can result in compliance issues and loss of patient trust.
8. Training and Support: Users may face difficulties in understanding and utilizing the system effectively due to inadequate training and limited technical support. This can hinder user adoption and limit the system's potential benefits.
9. Accessibility and Usability: The system may not be designed with consideration for accessibility standards, making it challenging for users with disabilities to access and navigate the system. Usability issues, such as unclear labels or confusing workflows, can also hinder user productivity.
10. System Performance and Reliability: The system may experience performance issues, such as slow response times or frequent downtime, impacting user productivity and patient care.

It is important to identify these specific problem areas through user feedback, system assessments, and continuous improvement processes. Addressing these challenges can lead to a more efficient and effective Patient Management System that supports optimal patient care and organizational workflows.

7.4 Possible Solutions

When addressing the problem areas in a Patient Management System, it is essential to identify and implement appropriate solutions. Here are some possible solutions for the mentioned problem areas:

1. User Interface Complexity:
    - Conduct user experience research to understand user needs and preferences.
    - Redesign the user interface with a focus on simplicity, intuitive navigation, and clear labeling.

    - Implement user-friendly features such as dropdown menus, autocomplete fields, and contextual help.
2. Data Entry and Data Quality:
    - Provide data validation mechanisms to ensure accurate and complete data entry.
    - Implement data integrity checks and automated validation rules to minimize data errors.
    - Offer data import functionalities to reduce manual data entry and improve efficiency.
3. Appointment Scheduling and Management:
    - Implement a user-friendly appointment scheduling module with features like calendar views, drag-and-drop functionality, and automated reminders.
    - Enable real-time availability updates to prevent scheduling conflicts.
    - Integrate with communication channels (e.g., email or SMS) to send appointment reminders and allow patients to confirm or reschedule appointments.
4. Integration with External Systems:
    - Assess the integration needs with other healthcare systems and identify key data exchange requirements.
    - Implement standardized interfaces (e.g., HL7 or FHIR) for seamless integration with electronic health records, laboratory systems, or billing systems.
    - Collaborate with external system vendors to establish secure and reliable data exchange protocols.
5. Lack of Customization and Scalability:
    - Provide configurable options and settings to adapt the system to different healthcare settings and workflows.
    - Allow users to customize data fields, forms, and reporting templates according to their specific needs.
    - Ensure the system architecture is scalable and can handle increased data volumes and user loads.
6. Inadequate Reporting and Analytics:
    - Develop comprehensive reporting and analytics capabilities to generate meaningful insights and support data-driven decision-making.
    - Implement customizable dashboards, data visualizations, and drill-down capabilities.
    - Enable the export of reports in various formats (e.g., PDF, Excel) for easy sharing and analysis.
7. Security and Privacy Concerns:
    - Conduct regular security assessments and penetration testing to identify vulnerabilities.
    - Implement robust user authentication and access control mechanisms.
    - Encrypt sensitive data, both in transit and at rest.

    - Comply with relevant data protection regulations, such as HIPAA or GDPR.
8. Training and Support:
    - Develop comprehensive training materials, user guides, and tutorials to onboard users and enhance their system proficiency.
    - Offer training sessions or workshops for both new and existing users.
    - Provide responsive technical support channels, such as a dedicated helpdesk or online support portal.
9. Accessibility and Usability:
    - Ensure compliance with accessibility standards (e.g., WCAG) to make the system usable for individuals with disabilities.
    - Conduct accessibility testing and address any identified issues promptly.
    - Incorporate user feedback and conduct usability testing to improve the system's overall user experience.
10. System Performance and Reliability:
    - Regularly monitor system performance and conduct performance tuning as needed.
    - Implement redundancy and failover mechanisms to ensure high availability and minimize downtime.
    - Conduct load testing to assess system performance under peak usage conditions and make necessary optimizations.

These solutions should be tailored to the specific needs and requirements of the Patient Management System and continuously evaluated and refined based on user feedback and emerging industry best practices.

7.5 Overall Requirement List

Functional Requirement

Here is an overall list of functional requirements for a Patient Management System:

1. User Registration and Authentication:
    - Allow users to create accounts and authenticate securely.
    - Provide different user roles with varying levels of access and permissions.
2. Patient Profile Management:
    - Enable users to create and manage patient profiles with relevant information such as personal details, medical history, and contact information.
    - Allow users to update and maintain patient records as necessary.
3. Appointment Scheduling:
    - Provide a calendar-based interface for scheduling patient appointments.

    - Allow users to search for available time slots, book appointments, and send automated reminders to patients.
    - Handle conflicts, cancellations, and rescheduling of appointments.
4. Electronic Health Records (EHR) Management:
    - Allow users to record and access patient health information, including medical history, diagnoses, treatments, medications, and lab results.
    - Support efficient data entry, retrieval, and update of EHRs.
    - Ensure data security and privacy.
5. Medical Billing and Invoicing:
    - Generate invoices and bills for patient visits, procedures, and treatments.
    - Integrate with billing systems or provide billing features to manage insurance claims, payment processing, and financial transactions.
6. Prescription and Medication Management:
    - Enable healthcare providers to generate electronic prescriptions and send them to pharmacies.
    - Maintain a database of medications, dosages, and patient-specific instructions.
    - Track medication dispensing, refills, and interactions.
7. Laboratory and Test Results Management:
    - Facilitate the ordering and tracking of laboratory tests.
    - Store and display test results, including reference ranges and interpretation.
    - Enable easy retrieval and analysis of historical test data.
8. Communication and Messaging:
    - Provide secure messaging channels for communication between healthcare providers, patients, and other involved parties.
    - Allow users to send and receive messages, share files, and discuss patient-related information securely.
9. Reporting and Analytics:
    - Generate various reports and analytics on patient demographics, appointment statistics, medical conditions, and treatment outcomes.
    - Provide customizable reporting options, including filtering, sorting, and exporting capabilities.
10. Integration with External Systems:
    - Integrate with external systems such as electronic health record systems, laboratory information systems, or pharmacy systems to exchange data seamlessly.
    - Support standardized data formats and protocols for interoperability.

Here is an overall list of non-functional requirements for a Patient Management System:

1. Usability:

    - The system should have a user-friendly interface, with intuitive navigation and clear instructions.
    - It should support efficient and easy-to-understand workflows for users of varying technical expertise.
    - The system should be responsive, providing a seamless user experience across different devices and screen sizes.
2. Performance and Scalability:
    - The system should be able to handle a large volume of concurrent users and data without significant degradation in performance.
    - It should have fast response times for common operations and minimize any delays or latency.
    - The system should be scalable, allowing for future growth and accommodating increased user and data loads.
3. Reliability and Availability:
    - The system should be highly reliable, with minimal downtime and disruptions.
    - It should have backup and disaster recovery mechanisms in place to ensure data integrity and availability.
    - The system should be able to handle and recover from system failures or crashes gracefully.
4. Security:
    - The system should implement robust security measures to protect patient data and comply with applicable privacy regulations.
    - It should provide secure authentication and access controls to ensure authorized usage and prevent unauthorized access.
    - The system should encrypt sensitive data in transit and at rest and implement safeguards against data breaches or unauthorized modifications.
5. Integration and Interoperability:
    - The system should support integration with other healthcare systems, such as electronic health record systems, billing systems, or laboratory systems.
    - It should adhere to industry standards and protocols for seamless data exchange and interoperability.
    - The system should have well-defined APIs and interfaces to facilitate integration with external systems.
6. Compliance and Regulatory Requirements:
    - The system should comply with relevant healthcare regulations and standards, such as HIPAA (Health Insurance Portability and Accountability Act) or GDPR (General Data Protection Regulation).
    - It should have features and controls in place to ensure compliance with data privacy, security, and confidentiality requirements.
7. Data Backup and Recovery:

- The system should have regular and automated data backup mechanisms to prevent data loss.
- It should support data recovery procedures to restore data in case of any unforeseen events or disasters.

8. Support and Maintenance:
    - The system should have a well-defined support and maintenance plan, including regular updates, bug fixes, and user assistance.
    - It should provide documentation, training resources, and technical support channels to assist users in using and troubleshooting the system.
9. Performance Monitoring and Analytics:
    - The system should include monitoring and reporting capabilities to track system performance, usage patterns, and potential bottlenecks.
    - It should provide analytics and insights to help administrators identify areas for optimization and improvement.
10. Compliance with Accessibility Standards:
    - The system should adhere to accessibility standards, ensuring that individuals with disabilities can access and use the system effectively.
    - It should support assistive technologies and provide accessible features such as screen readers, keyboard navigation, and adjustable font sizes.

These non-functional requirements are crucial for ensuring the overall quality, reliability, and usability of the Patient Management System. It is essential to align these requirements with the specific needs and regulatory frameworks of the healthcare industry.

7.6 Technology to be Implemented

The technology to be implemented for a Patient Management System can vary depending on various factors such as the project requirements, scalability needs, budget, and the existing IT infrastructure.

Client-Server Application Technology

Client-server application technology refers to the architecture and technologies used to build and deploy applications that are divided into two main components: the client and the server. The client is the user-facing part of the application, responsible for interacting with the user interface, while the server handles the business logic, data processing, and storage.

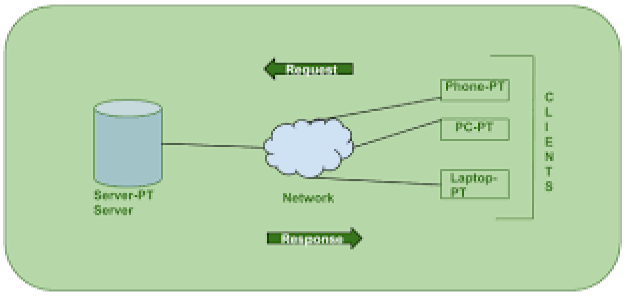


Figure 13: Client Server Application Model

The following are the fundamental characteristics of a client-server application:

- Programming Languages: The choice of programming languages may vary depending on the specific requirements and preferences. Commonly used languages for client-side development include JavaScript (for web-based applications) and languages like Java, Swift, or Kotlin (for mobile applications). On the server-side, languages like Java, C#, Python, or Node.js are often used.
- Client-side Technologies: For web-based client applications, HTML, CSS, and JavaScript are essential. HTML provides the structure of the web page, CSS is used for styling and layout, and JavaScript enables interactive functionality and dynamic content.
- Web APIs: Web APIs (Application Programming Interfaces) allow communication between the client and the server. RESTful APIs are commonly used, which use HTTP methods (such as GET, POST, PUT, DELETE) to request and manipulate data.
- Server-side Technologies: The server-side of the application typically involves a web server and a backend framework. Popular web servers include Apache HTTP Server and Nginx. Backend frameworks like Java Spring, .NET Core, Ruby on Rails, or Django provide a foundation for handling requests, managing business logic, and accessing databases.
- Databases: Data storage and retrieval are critical components of client-server applications. Relational databases like MySQL, PostgreSQL, or Oracle are commonly used for structured data. Alternatively, NoSQL databases like MongoDB or Cassandra can be used for handling unstructured or semi-structured data.

Web Application

A web application is a type of software application that runs on a web server and is accessed through a web browser over the internet. It is designed to provide interactive functionality and deliver dynamic content to users.

Figure 14:Web Application

The following are the primary characteristics of a web-based application:

- Client-Server Architecture: Web applications follow a client-server architecture, where the client (web browser) sends requests to the server, and the server processes those requests, retrieves data from databases or other sources, and sends back a response to the client.
- User Interface: The user interface of a web application is typically built using a combination of HTML, CSS, and JavaScript. HTML provides the structure of the web page, CSS is used for styling and layout, and JavaScript enables interactivity and dynamic content.
- Web Technologies: Web applications rely on various technologies and frameworks to facilitate development and enhance functionality. This can include server-side languages like PHP, Python, Ruby, or JavaScript (Node.js), as well as front-end frameworks like React, Angular, or Vue.js.
- Data Storage: Web applications often require persistent data storage. This can be achieved through databases such as MySQL, PostgreSQL, MongoDB, or other database management systems. The application interacts with the database to store and retrieve data as needed.
- Server-Side Processing: The server-side of a web application handles business logic, data processing, and integration with other systems or services. It manages user authentication, data validation, security, and performs operations required to fulfill client requests.

7.7 Recommendation and Justification

Based on the description of the project and the identified requirements, I would recommend developing a web-based patient management system for the healthcare facility. Here are the justifications for this recommendation:

1. Accessibility: A web application allows easy access to the patient management system from various devices with an internet connection. This ensures that doctors, administrators, and staff members can access the system and manage patient information anytime, anywhere, making it convenient and efficient.
2. Scalability: Web applications can easily scale to accommodate the growing needs of the healthcare facility. As the number of patients and users increases, the web application can be easily scaled by adding more server resources, implementing load balancing, and optimizing database performance. This ensures that the system remains responsive and performs well even under heavy usage.
3. Collaboration and Communication: A web-based system allows for seamless collaboration and communication between different departments and healthcare professionals. Doctors can access patient records, share information, and communicate

with colleagues regarding treatment plans and updates. This improves coordination and enhances the quality of patient care.

4. Security: With proper implementation of security measures, web applications can provide a secure environment for storing and managing sensitive patient data. Measures such as encryption, secure authentication, role-based access control, and regular security audits can be implemented to protect patient information and comply with data protection regulations.
5. Integration Capabilities: Web applications can easily integrate with other systems and services, such as laboratory systems, billing systems, and insurance providers. This allows for seamless exchange of data and information, reducing manual data entry and improving workflow efficiency.

Chapter 8 – Exploration

8.1 University Management Sector Old System Use Case

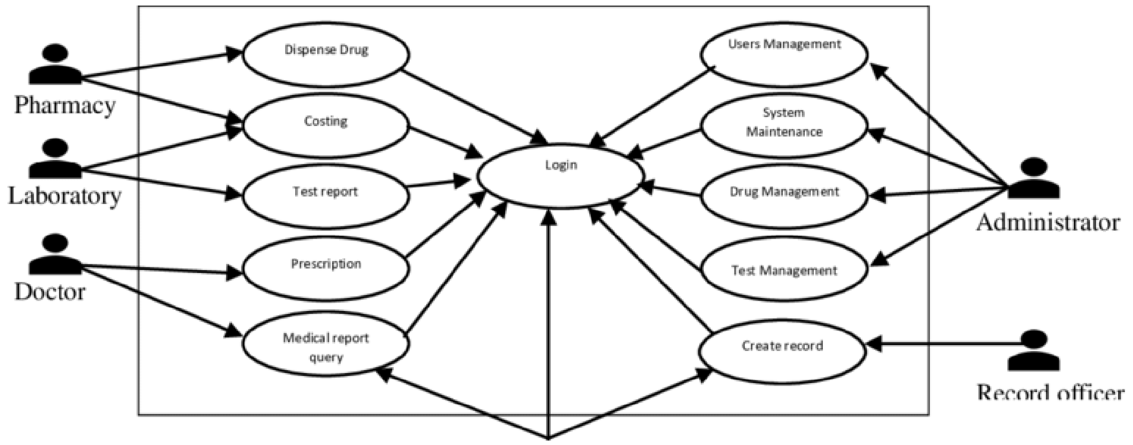


Figure 15: Use Case Diagram of Patient Management System

8.2 Activity Diagram

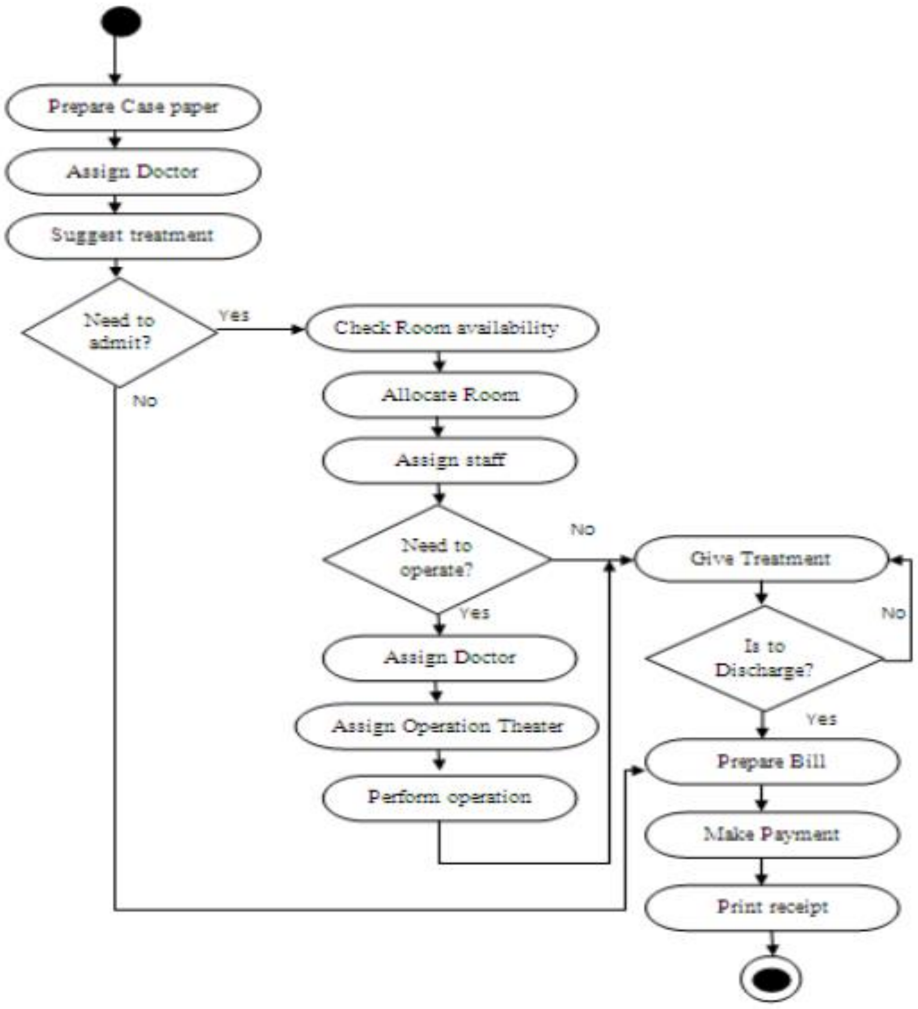


Figure 16: Admin Activity Diagram of Medenico

8.3 Full System Use Case

The following is a use case diagram for the proposed Medenico:

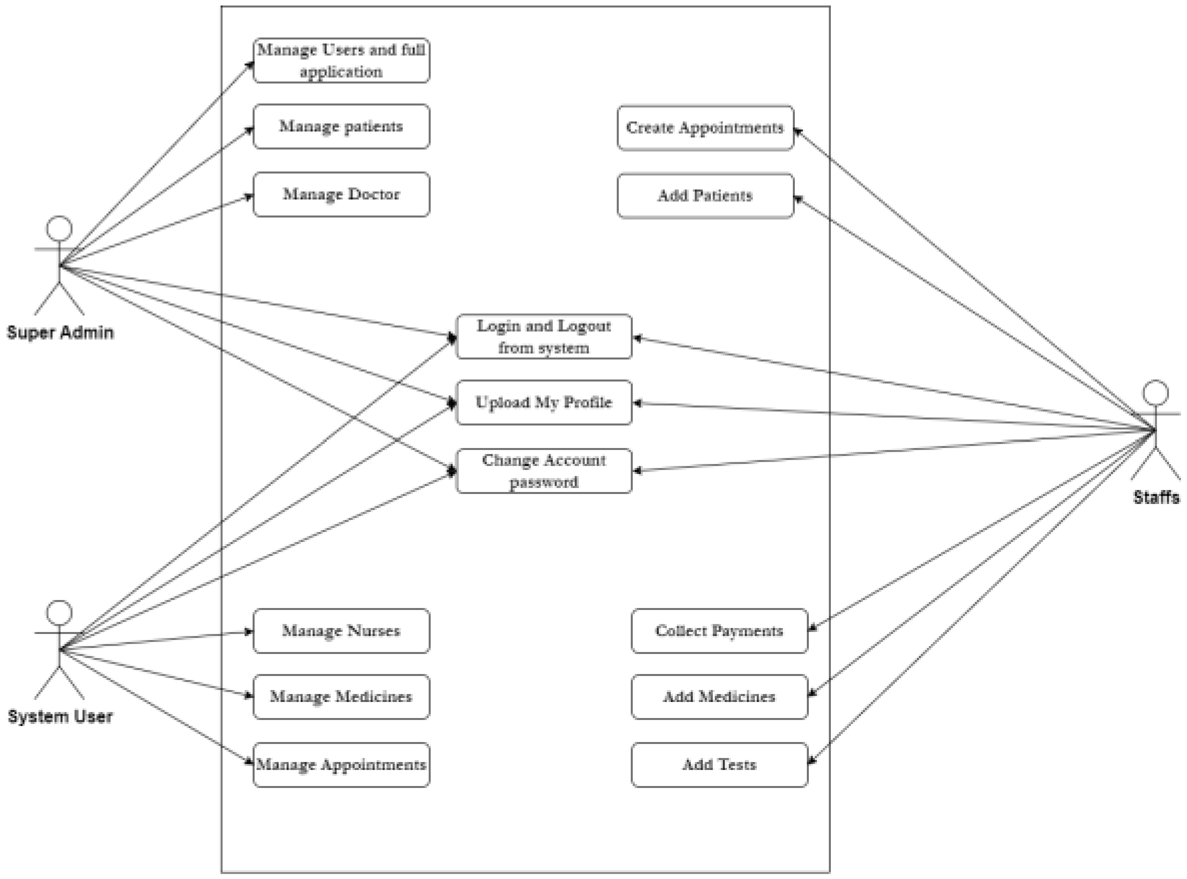


Figure 17:Use Case of the Proposed Medenico System

8.4 Full System Activity Diagram

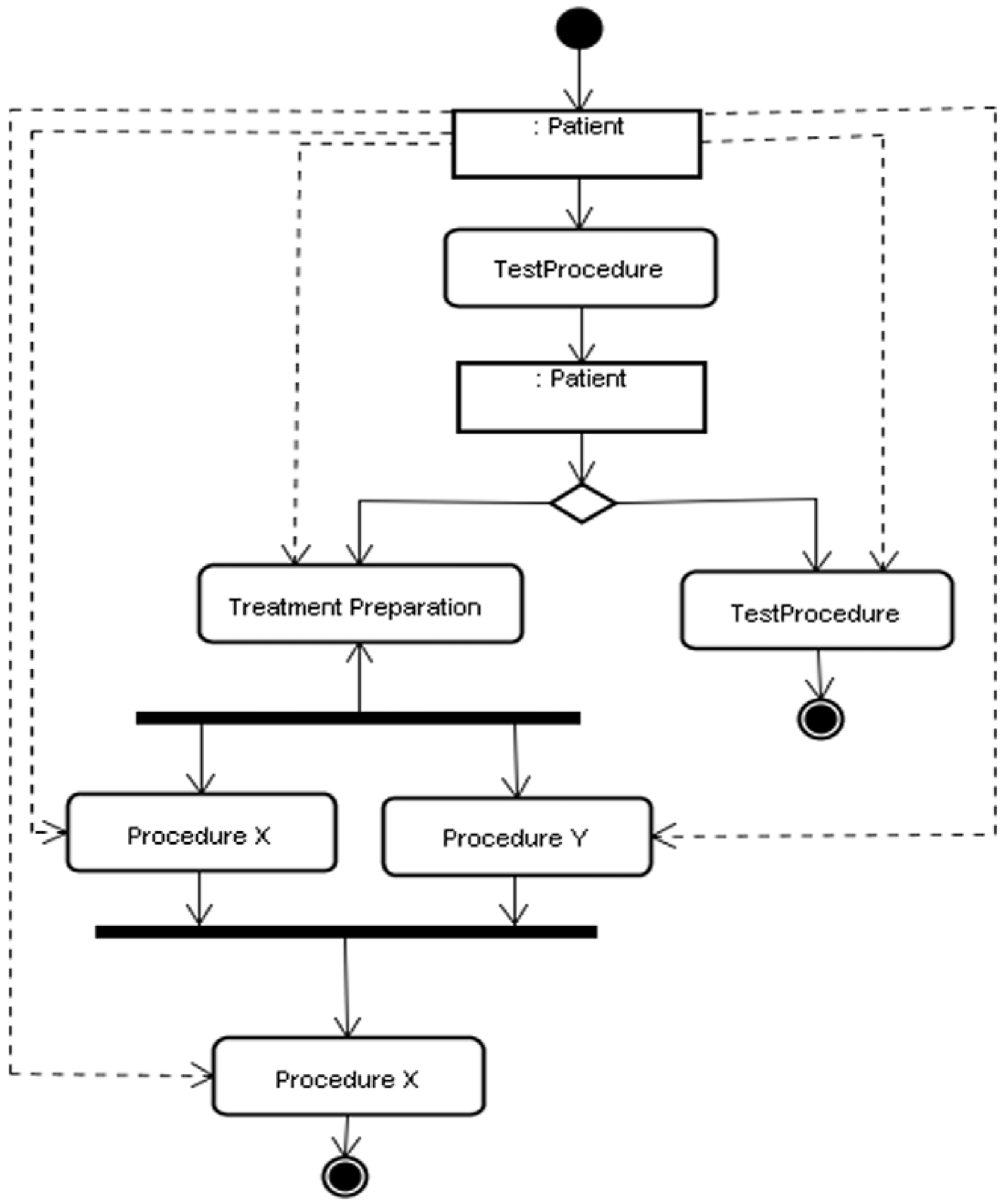


Figure 18:Patient Activity Diagram Medenico

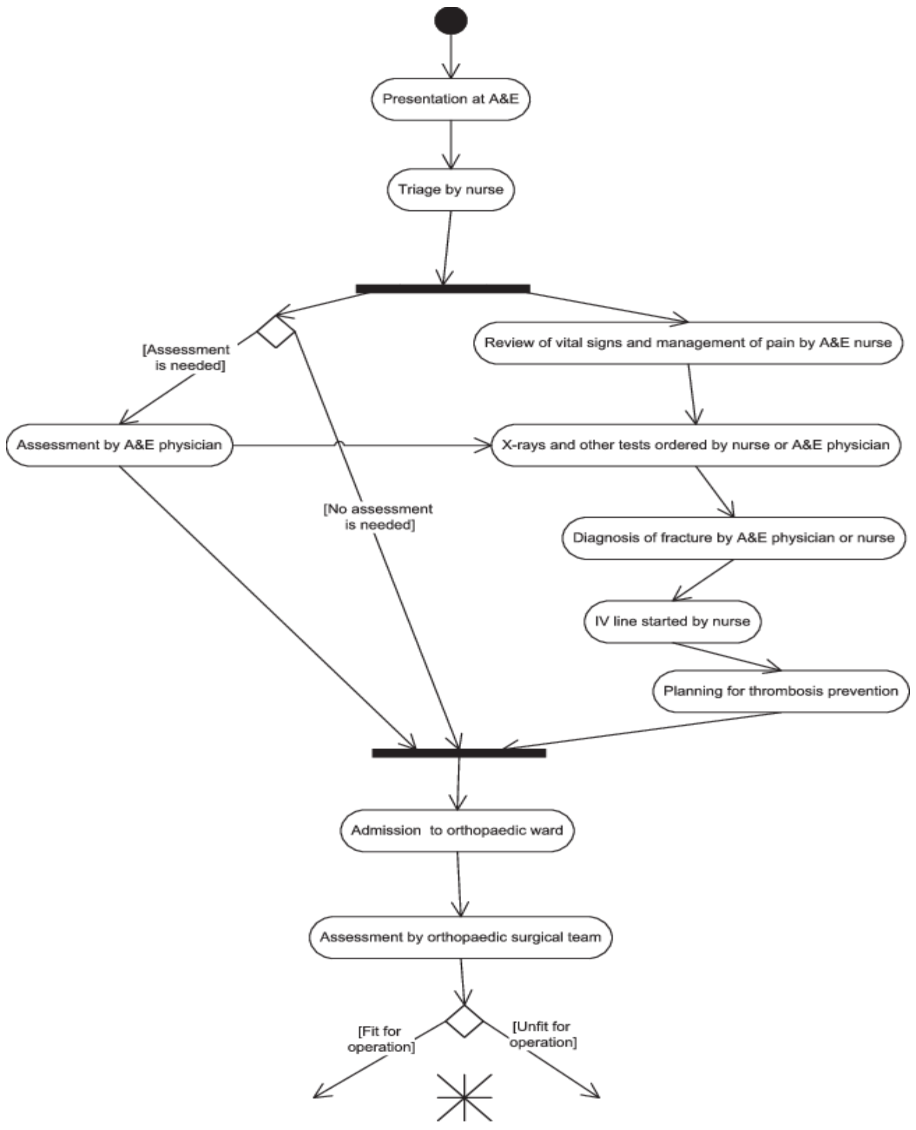

Presentation at A&E
Triage by nurse
[Assessment is needed]
Review of vital signs and management of pain by A&E nurse
Assessment by A&E physician
X-rays and other tests ordered by nurse or A&E physician
[No assessment is needed]
Diagnosis of fracture by A&E physician or nurse
IV line started by nurse
Planning for thrombosis prevention
Admission to orthopaedic ward
Assessment by orthopaedic surgical team
[Fit for operation]
[Unfit for operation]

Figure 19:Admin Activity Diagram Medenico

8.5 Catalogue of Requirements

User Requirements Catalouge

Source	Sing Off	Priority	Requirement
User control and access all	Admin/Management	Must Have	M-01

Functional Requirement User Requirement Catalogue: Registration and Authentication of Users Allow people to register and create accounts (administrators, management, employees, and patients).

Non-functional requirement

Description	Target Value	Acceptance Value	Comment
User (per/day)	15000 (per/day.)	1500	

Table 14: Requirement Catalogue for User

Management Requirements Catalouge

Source	Sing Off	Priority	Requirement
Management	Admin	Must Have	M-02

Functional Requirement Management Requirement Catalogue: Patient Information System (PIS) Allow for patient registration and management.

Non-functional requirement

Description	Target Value	Acceptance Value	Comment
Management (per/day)	100000 (per/day.)	75000	

Table 15: Requirement Catalogue for Management

Treatment Requirements Catalouge

Source	Sing Off	Priority	Requirement
Treatment Management	Management & Doctor	Must Have	M-03

Functional Requirement
Treatment Management Requirement Catalogue:
Test all see patient’s report and set medecine and treatment

Non-functional requirement

Description	Target Value	Acceptance Value	Comment
Treatment(per/day)	15(per/day.)	20	

Table 16: Requirement Catalogue for Treatment

Doctor Requirements Catalouge

Source	Sing Off	Priority	Requirement
Doctor Management	Doctor	Must Have	M-04

Functional Requirement
Doctor Management Requirement Catalogue:
Management assigns a doctor per patient and the doctor treat them.

Non-functional requirement

Description	Target Value	Acceptance Value	Comment
Doctor(per/day)	150(per/day.)	20	

Table 17: Requirement Catalogue for Doctor

Patient Requirements Catalouge

Source	Sing Off	Priority	Requirement
Patient Management	Patient	Must Have	M-04

Functional Requirement Patient Management Requirement Catalogue: Patient registration completed and take all treatment from management system

Non-functional requirement

Description	Target Value	Acceptance Value	Comment
Patient(per/day)	150(per/day.)	20	

Table 18: Requirement Catalogue for Patient

8.6 Prioritized Requirements List (PRL)

Must Have requirements-

Serial No.	**Requirement for Medenico**	**Priority of Medenico**
01.	Patient Registration	Must-have
02.	Appointment Scheduling	Must-have
03.	Medical Records Management	Must-have
04.	Billing and Payment	Must-have

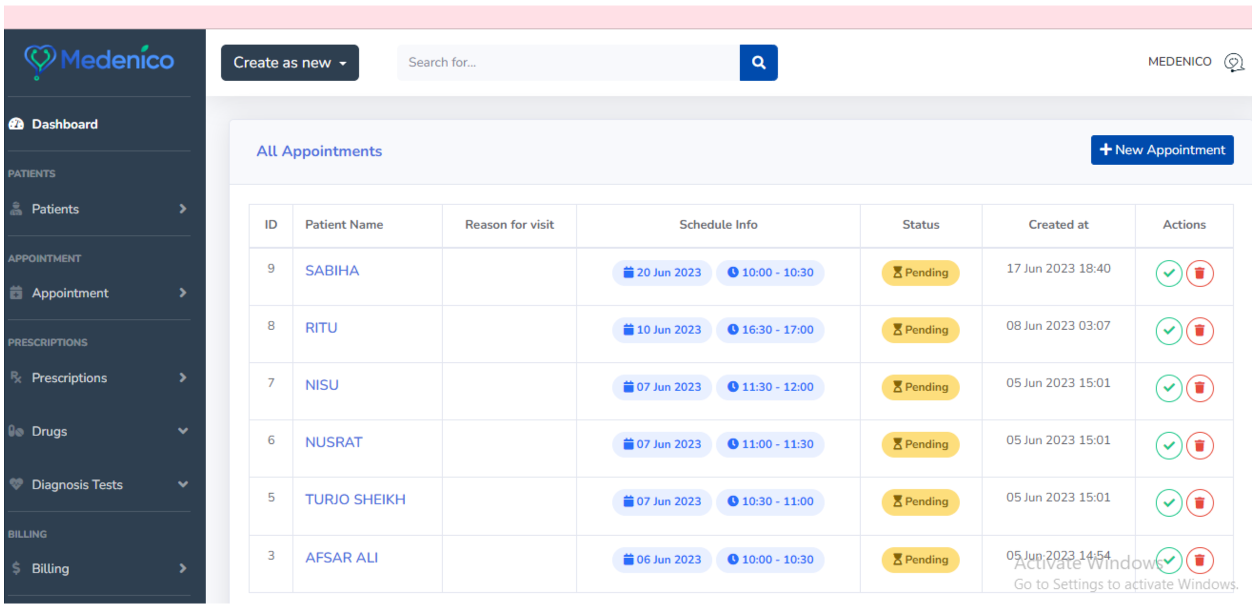
Should Have requirements-

Serial No.	Requirement for Medenico	Priority of Medenico
01.	Patient Check-In/Check-Out	Should-have
02.	Electronic Prescriptions	Should-have
03.	Clinical Decision Support	Should-have
04.	Reporting and Analytics	Should-have

Could Have requirements-

Serial No.	Requirement for Medenico	Priority of Medenico
01.	Telemedicine/Remote Consultation	Could-have
02.	Patient Portal	Could-have
03.	Integration with External Systems	Could-have

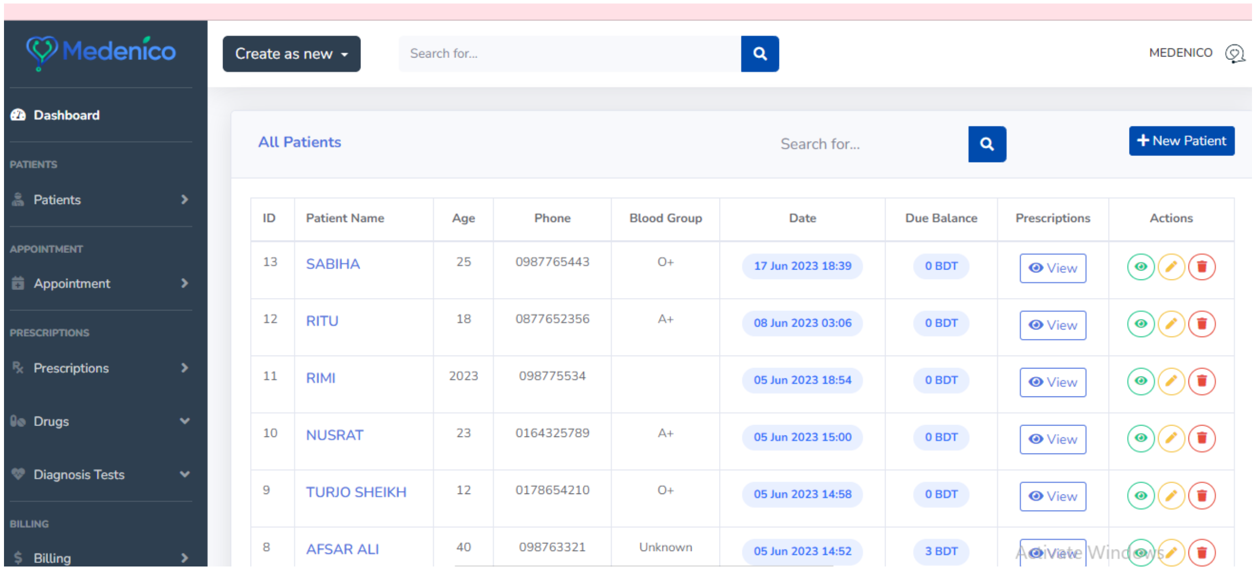
8.7 Patient Management Process



All Appointments

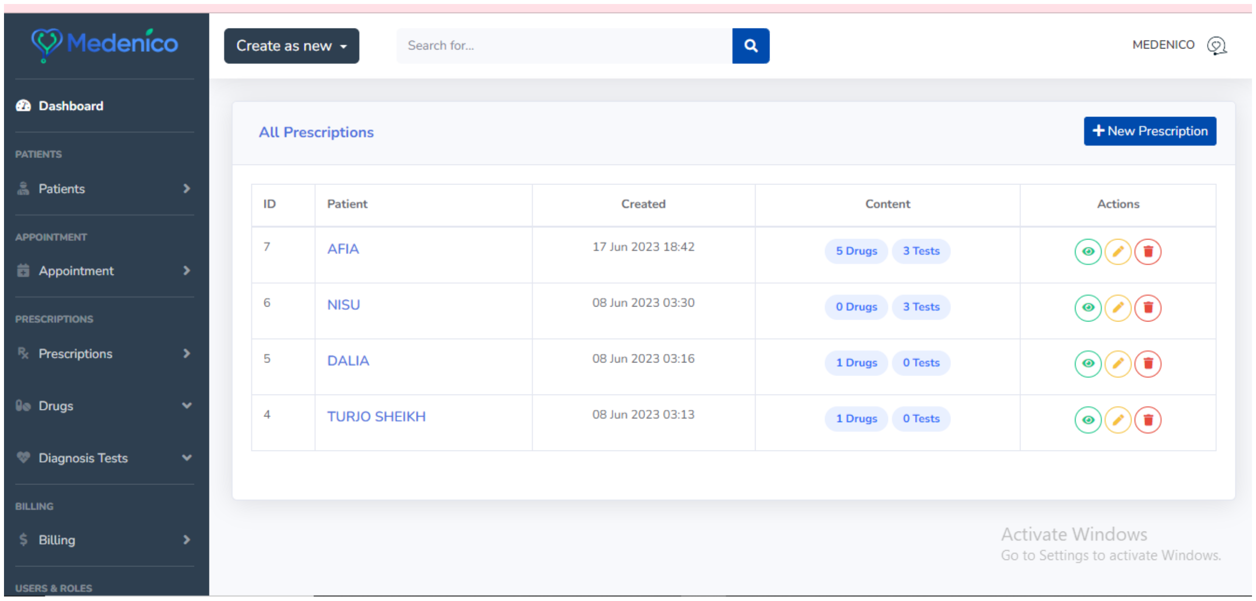
ID	Patient Name	Reason for visit	Schedule Info	Status	Created at	Actions
9	SABIHA		20 Jun 2023 10:00 - 10:30	Pending	17 Jun 2023 18:40	
8	RITU		10 Jun 2023 16:30 - 17:00	Pending	08 Jun 2023 03:07	
7	NISU		07 Jun 2023 11:30 - 12:00	Pending	05 Jun 2023 15:01	
6	NUSRAT		07 Jun 2023 11:00 - 11:30	Pending	05 Jun 2023 15:01	
5	TURJO SHEIKH		07 Jun 2023 10:30 - 11:00	Pending	05 Jun 2023 15:01	
3	AFSAR ALI		06 Jun 2023 10:00 - 10:30	Pending	05 Jun 2023 14:54	

Figure 21: All Appointments



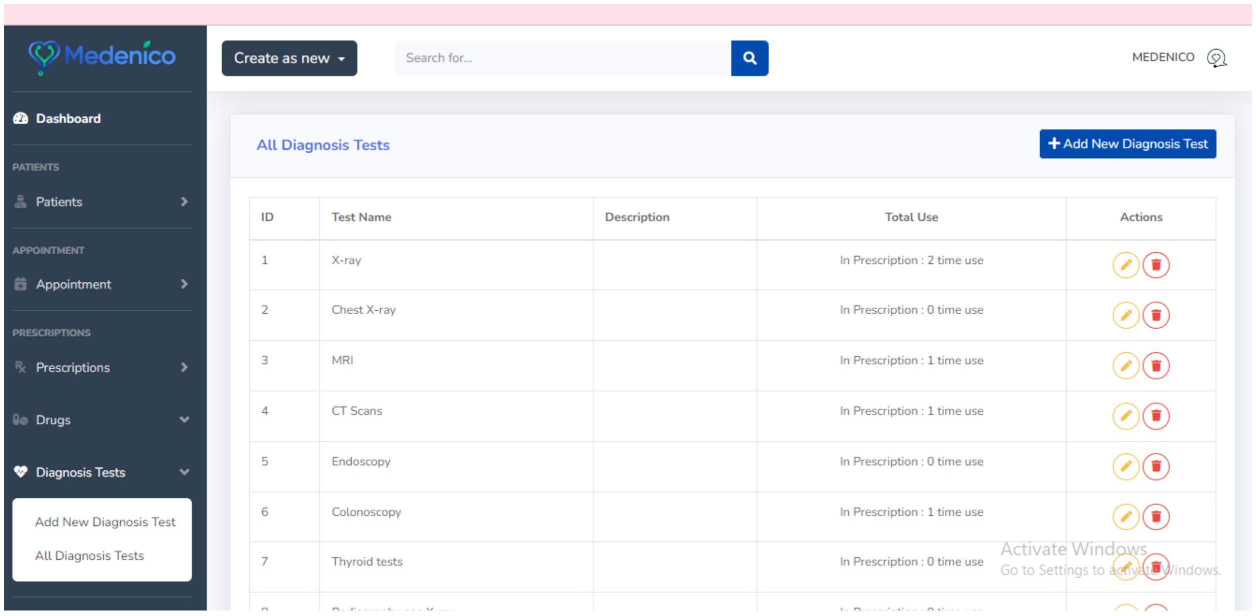
ID	Patient Name	Age	Phone	Blood Group	Date	Due Balance	Prescriptions	Actions
13	SABIHA	25	0987765443	O+	17 Jun 2023 18:39	0 BDT	View	
12	RITU	18	0877652356	A+	08 Jun 2023 03:06	0 BDT	View	
11	RIMI	2023	098775534		05 Jun 2023 18:54	0 BDT	View	
10	NUSRAT	23	0164325789	A+	05 Jun 2023 15:00	0 BDT	View	
9	TURJO SHEIKH	12	0178654210	O+	05 Jun 2023 14:58	0 BDT	View	
8	AFSAR ALI	40	098763321	Unknown	05 Jun 2023 14:52	3 BDT	View	

Figure 22: All Patient



ID	Patient	Created	Content	Actions
7	AFIA	17 Jun 2023 18:42	5 Drugs 3 Tests	
6	NISU	08 Jun 2023 03:30	0 Drugs 3 Tests	
5	DALIA	08 Jun 2023 03:16	1 Drugs 0 Tests	
4	TURJO SHEIKH	08 Jun 2023 03:13	1 Drugs 0 Tests	

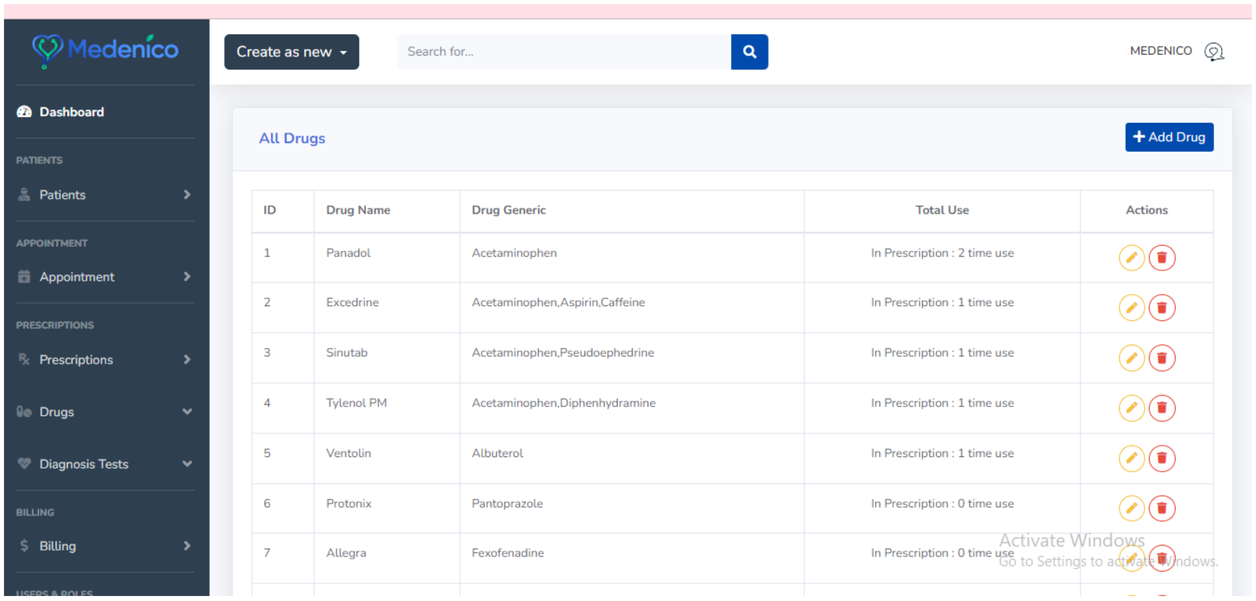
Figure 23: All Prescriptions



All Diagnosis Tests

ID	Test Name	Description	Total Use	Actions
1	X-ray		In Prescription : 2 time use	
2	Chest X-ray		In Prescription : 0 time use	
3	MRI		In Prescription : 1 time use	
4	CT Scans		In Prescription : 1 time use	
5	Endoscopy		In Prescription : 0 time use	
6	Colonoscopy		In Prescription : 1 time use	
7	Thyroid tests		In Prescription : 0 time use	

Figure 24: All Tests



All Drugs

ID	Drug Name	Drug Generic	Total Use	Actions
1	Panadol	Acetaminophen	In Prescription : 2 time use	
2	Excedrine	Acetaminophen,Aspirin,Caffeine	In Prescription : 1 time use	
3	Sinutab	Acetaminophen,Pseudoephedrine	In Prescription : 1 time use	
4	Tylenol PM	Acetaminophen,Diphenhydramine	In Prescription : 1 time use	
5	Ventolin	Albuterol	In Prescription : 1 time use	
6	Protonix	Pantoprazole	In Prescription : 0 time use	
7	Allegra	Fexofenadine	In Prescription : 0 time use	

Figure 25: All Drugs

Chapter 9 – Engineering

9.1 Approach New System Modules:

Module for Registration

Serial No.	Action of User	Serial No	Action of System
1.	Enter personal information for registration.	1.	Validate and store user registration data.
2.	Provide contact details and demographic information.	2.	Generate a unique user ID for future identification.
3.	Choose a username and password for login.		

Table 19:Module for Registration

Module for Appointment Scheduling

Serial No.	Action of User	Serial No	Action of System
1.	Select a preferred date and time for an appointment.	1.	Check doctor availability and schedule the appointment.
2.	Choose a specific doctor or department.	2.	Send confirmation notifications to users and doctors.
3.	Provide any additional details or requirements.	3.	Update the appointment calendar and availability.

Table 20:Module for Appointment Scheduling

Module for Patient Records Management

Serial No.	Action of User	Serial No	Action of System
1.	View and update personal information.	1.	Store and organize patient records securely.
2.	Access medical history, diagnoses, and treatments.	2.	Enable efficient retrieval and update of patient data.

3.	Upload and manage medical documents or test results.	3.	Support integration with other healthcare systems.

Table 21:Module for Patient Records Management

Module for Prescription Management

Serial No.	Action of User	Serial No	Action of System
1.	Request prescription refills.	1.	Process prescription refill requests.
2.	View prescribed medications and dosages.	2.	Maintain an up-to-date list of prescribed medications.
3.	Receive notifications for medication reminders.	3.	Send reminders for medication adherence.

Table 22:Module for Prescription Management

Module for Billing and Insurance

Serial No.	Action of User	Serial No	Action of System
1.	View and manage insurance information.	1.	Generate billing statements based on services rendered.
2.	Make payment for medical services.	2.	Integrate with insurance providers for claims processing.
3.	Access and download billing statements.	3.	Facilitate secure online payment options.

Table 23:Module for Billing and Insurance

9.2 Use Case Diagram of the Medenico

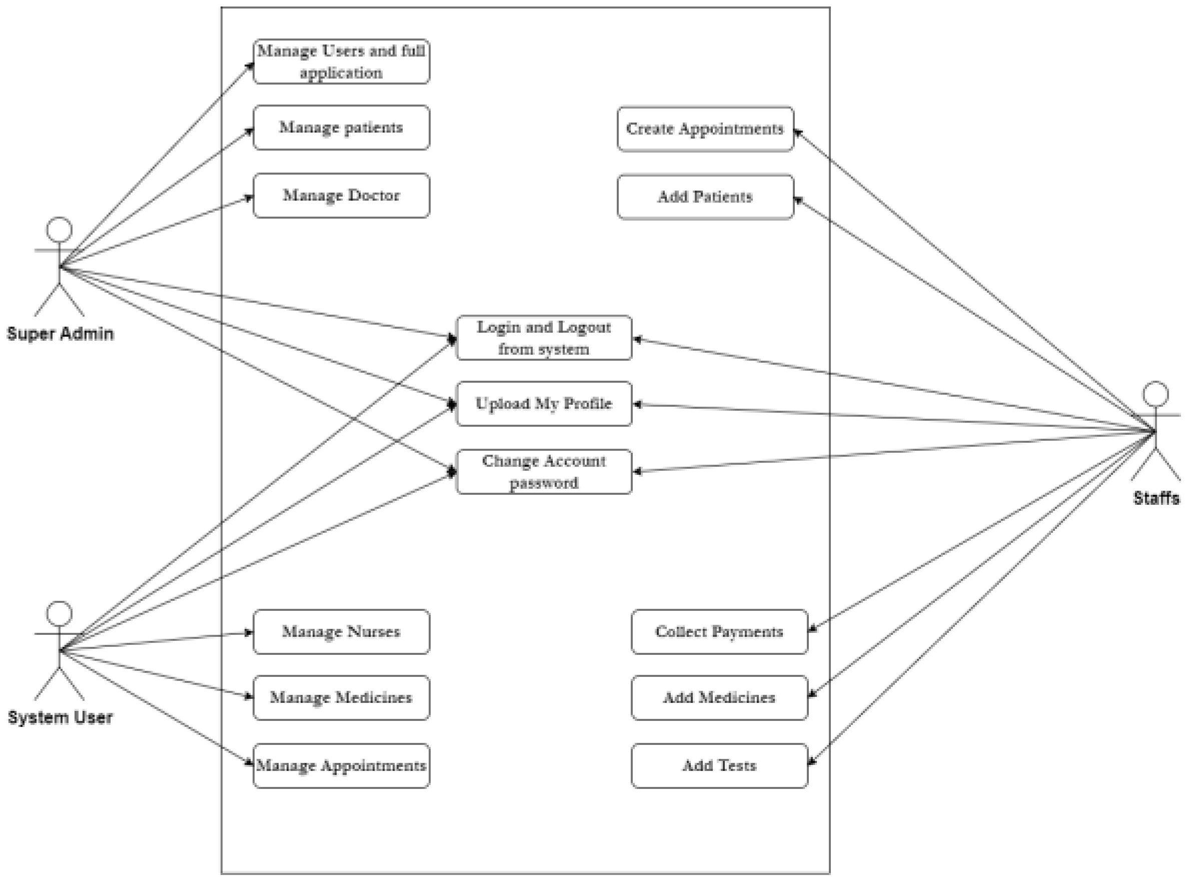


Figure 26: Medenico Use Case

9.3 The Medenico Class Diagram

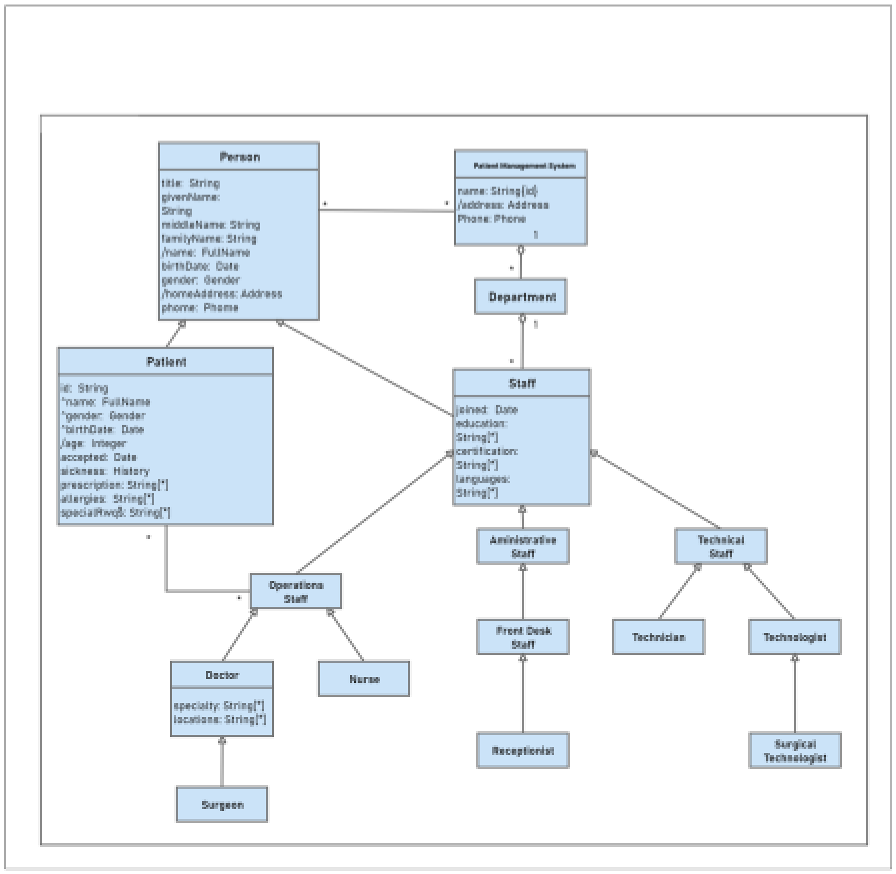


Figure 27:Medenico Class Diagram

9.4 Entity Relationship Diagram

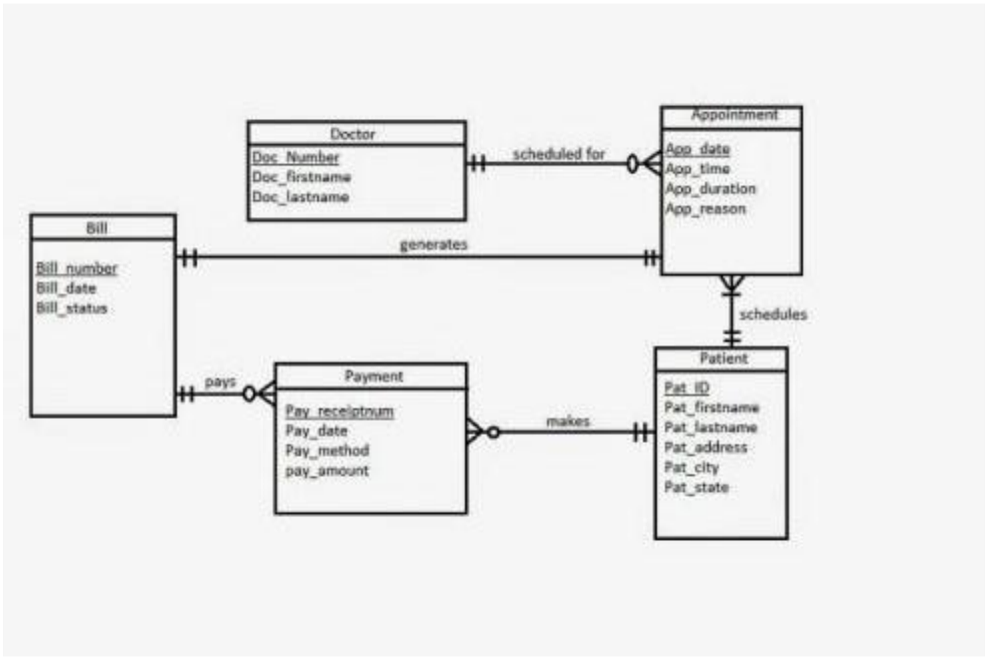


Figure 28: Medenico ERD

9.5 Medenico Sequence Diagram

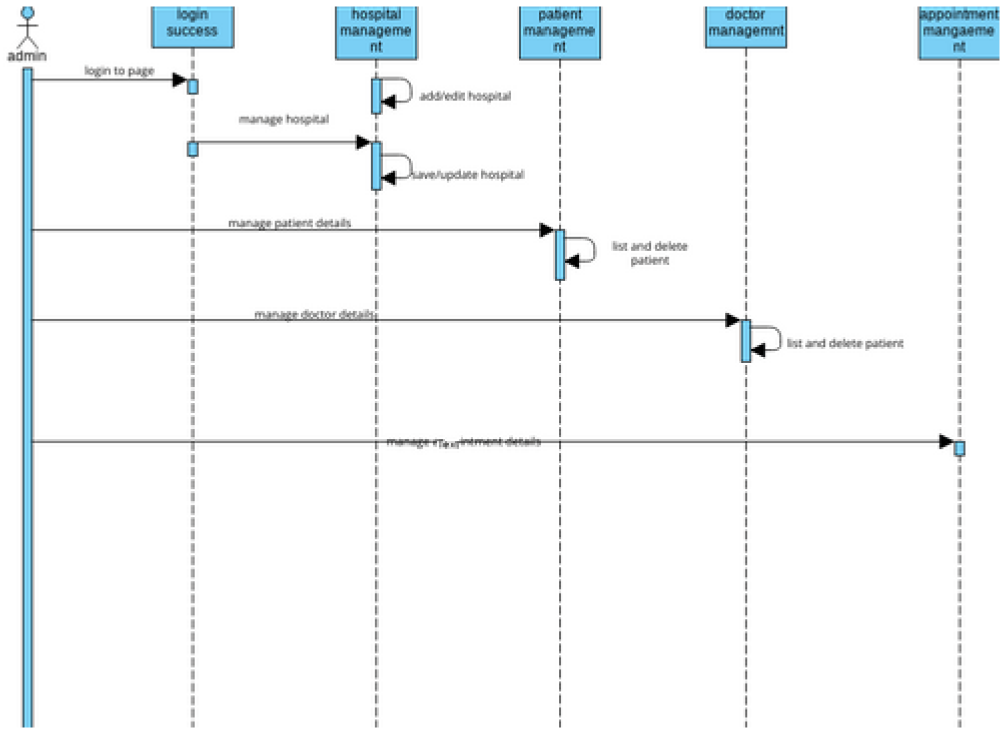


Figure 29:Medenico Sequence Diagram

9.6 The Medenico component diagram

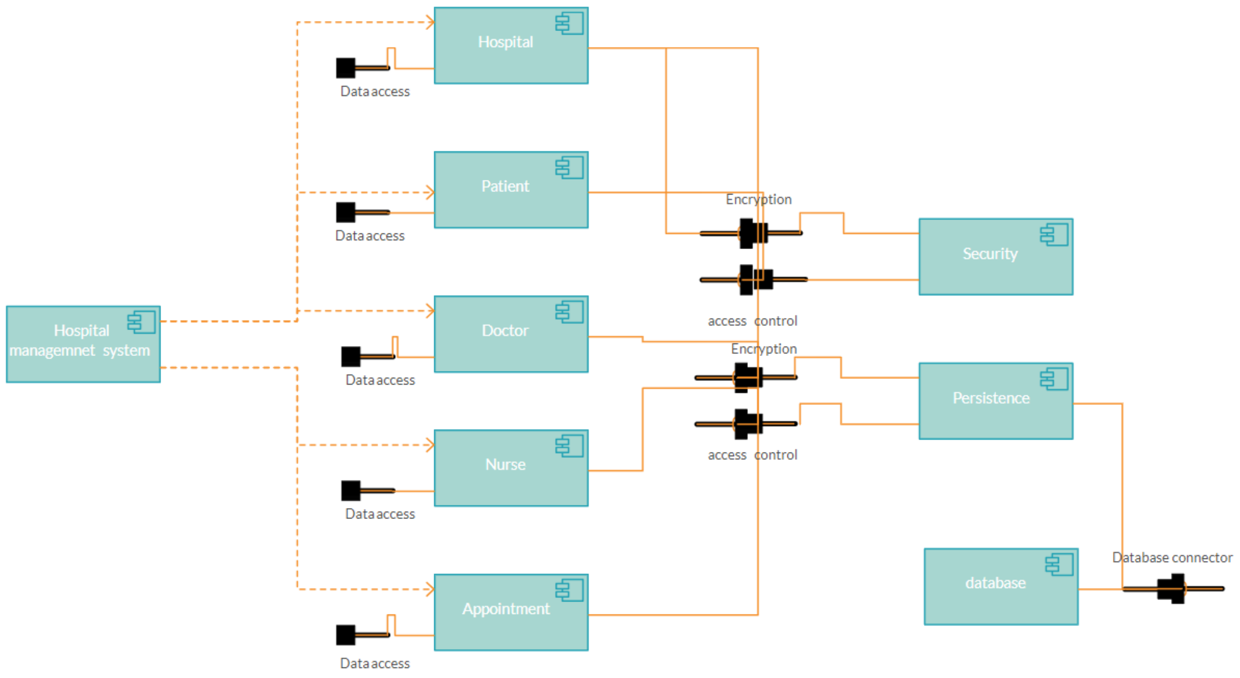
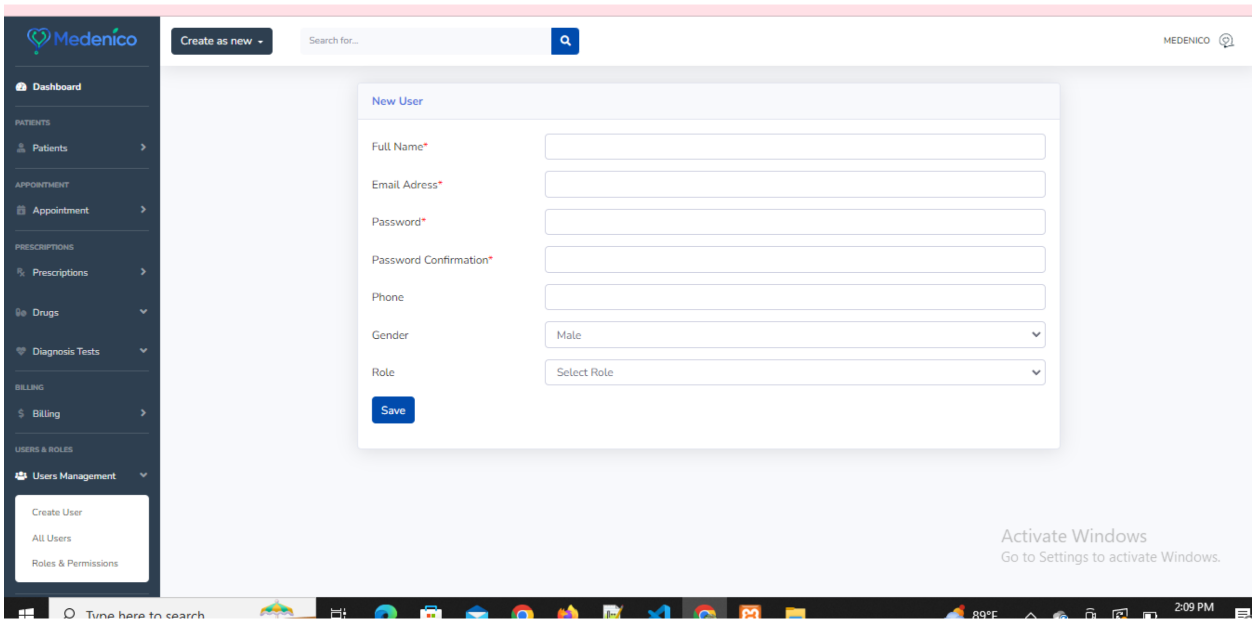


Figure 30:Component Diagram of Medenico

9.7 Deployment Diagram of Medenico

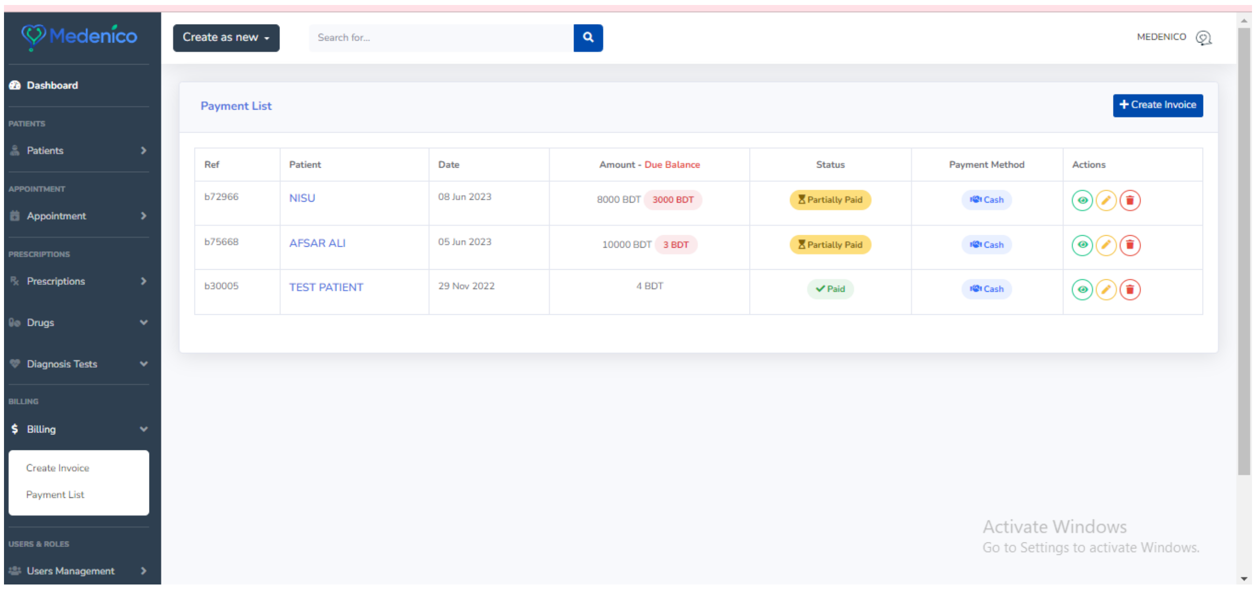
9.8 System Interface Design



New User

Full Name*	
Email Adress*	
Password*	
Password Confirmation*	
Phone	
Gender	Male
Role	Select Role

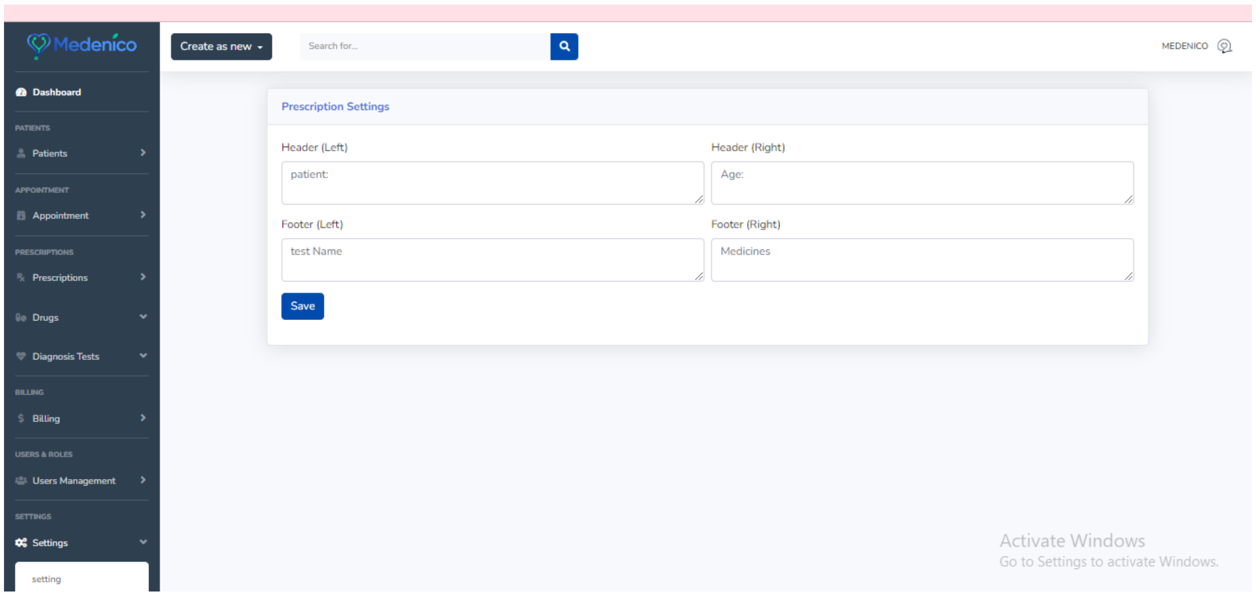
Figure 31: New User



Payment List

Ref	Patient	Date	Amount - Due Balance	Status	Payment Method	Actions
b72966	NISU	08 Jun 2023	8000 BDT 3000 BDT	Partially Paid	Cash	
b75668	AFSAR ALI	05 Jun 2023	10000 BDT 3 BDT	Partially Paid	Cash	
b30005	TEST PATIENT	29 Nov 2022	4 BDT	✓ Paid	Cash	

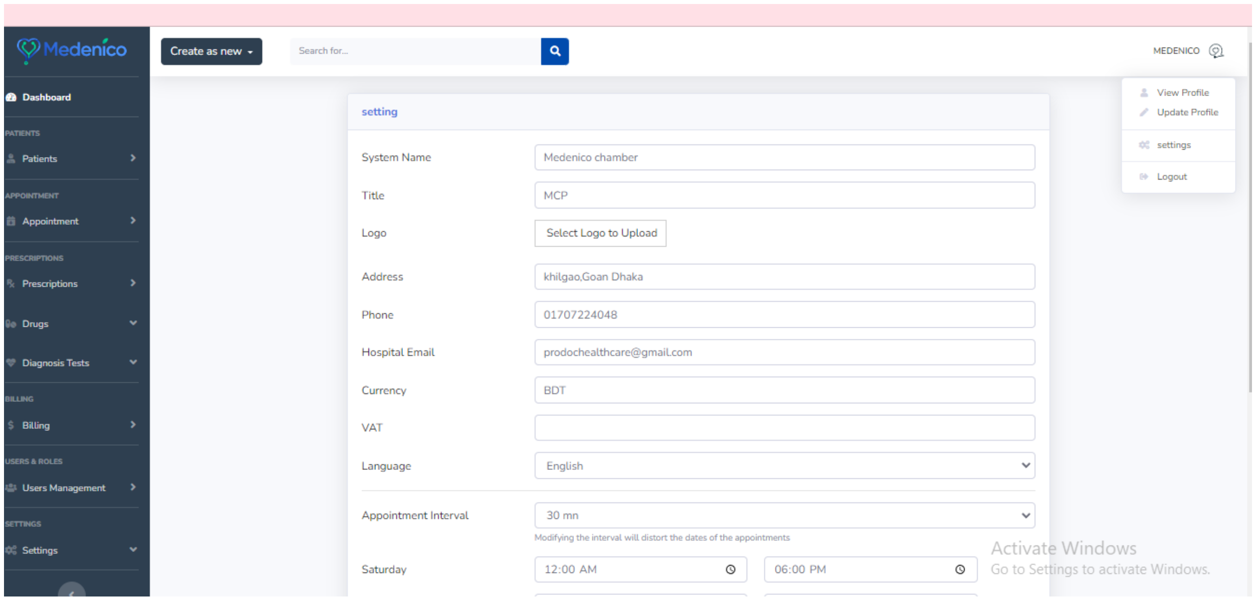
Figure 32: Payment List



Prescription Settings

Header (Left)	patient:
Header (Right)	Age:
Footer (Left)	test Name
Footer (Right)	Medicines

Figure 33: Prescription Setting

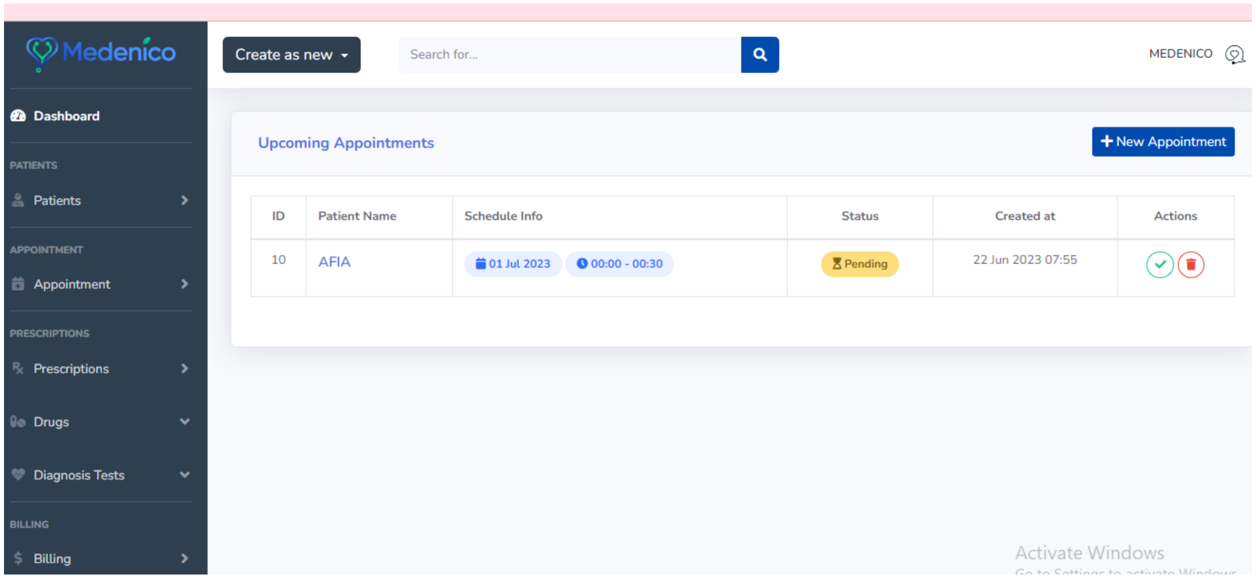


setting

System Name	Medenico chamber
Title	MCP
Logo	Select Logo to Upload
Address	khilgao,Goan Dhaka
Phone	01707224048
Hospital Email	prodochealthcare@gmail.com
Currency	BDT
VAT	
Language	English
Appointment Interval	30 mn
Saturday	12:00 AM / 06:00 PM

Modifying the interval will distort the dates of the appointments

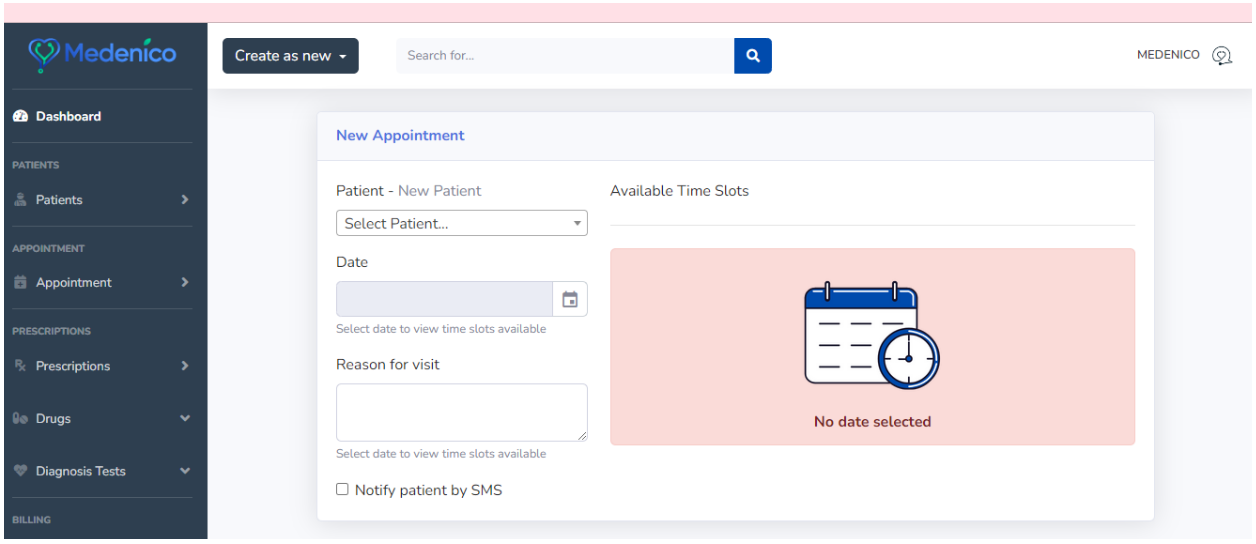
Figure 34: Setting



Upcoming Appointments

ID	Patient Name	Schedule Info	Status	Created at	Actions
10	AFIA	01 Jul 2023 00:00 - 00:30	Pending	22 Jun 2023 07:55	

Figure 35: Upcoming Appointments



New Appointment

Patient - New Patient

Select Patient...

Date

Select date to view time slots available

Reason for visit

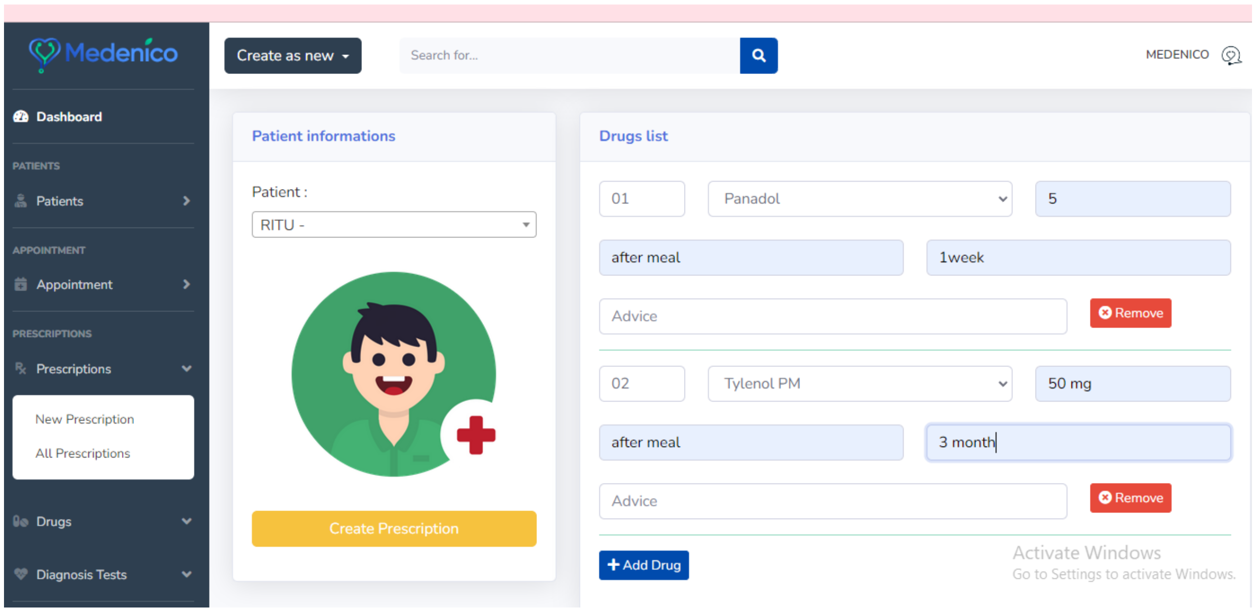
Select date to view time slots available

Notify patient by SMS

Available Time Slots

No date selected

Figure 36: New Appointments



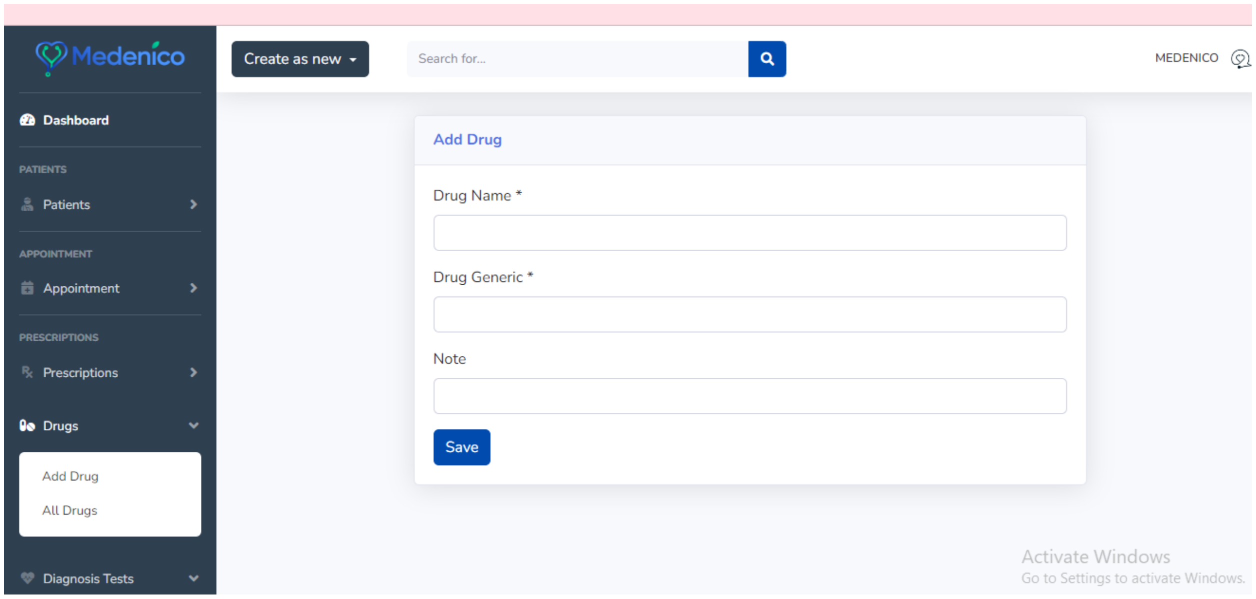
Patient informations

Patient : RITU -

Drugs list

01	Panadol	5
after meal	1week	
Advice		
02	Tylenol PM	50 mg
after meal	3 month	
Advice		

Figure 37: New Prescription



Add Drug

Drug Name *

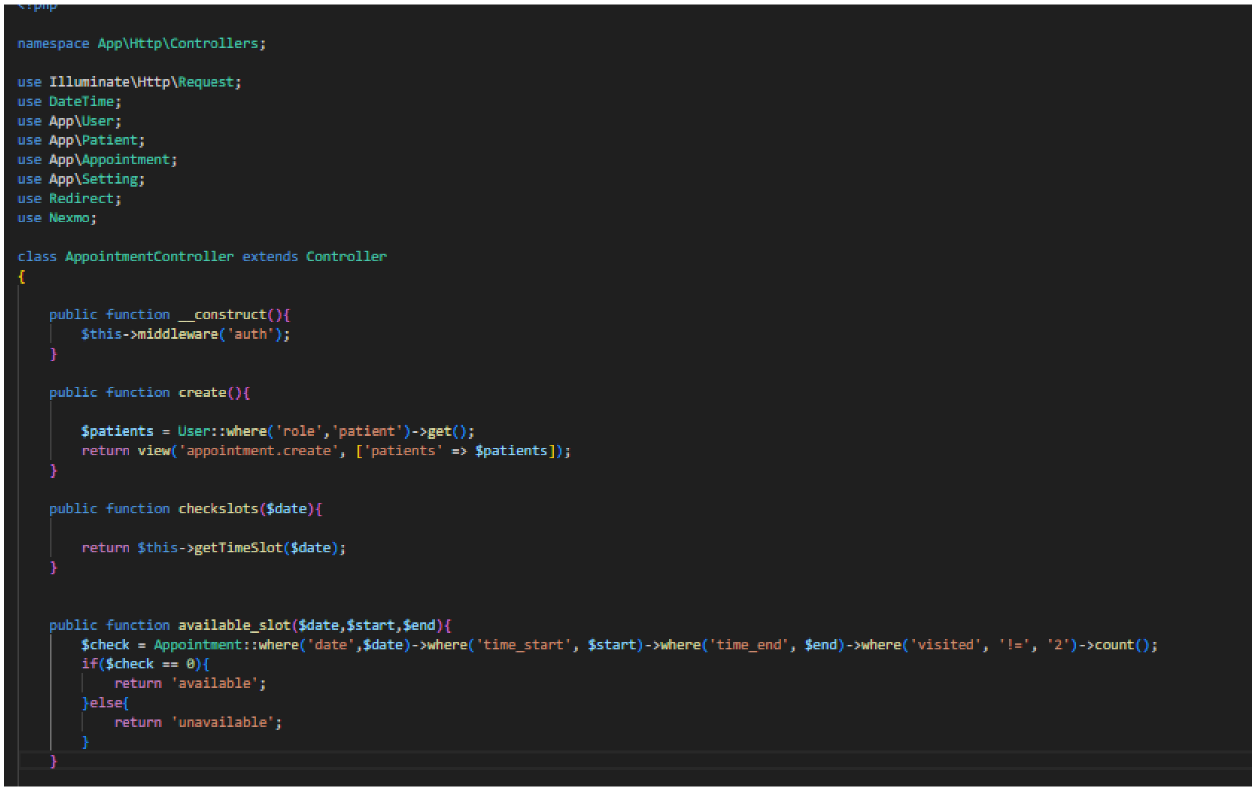
Drug Generic *

Note

Figure 38: Add Drug

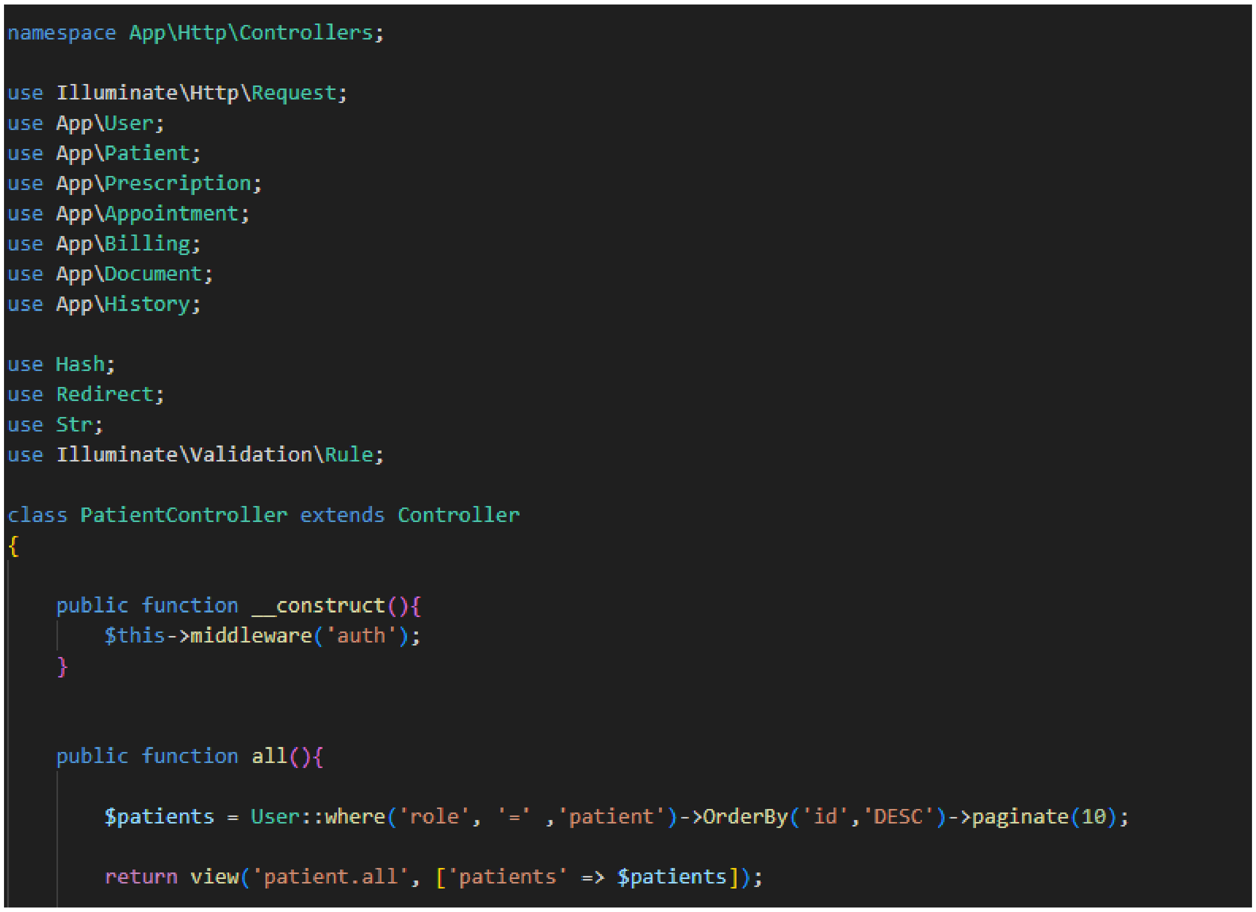
Chapter 10 – Deployment

10.1 Core Module Coding Sample:



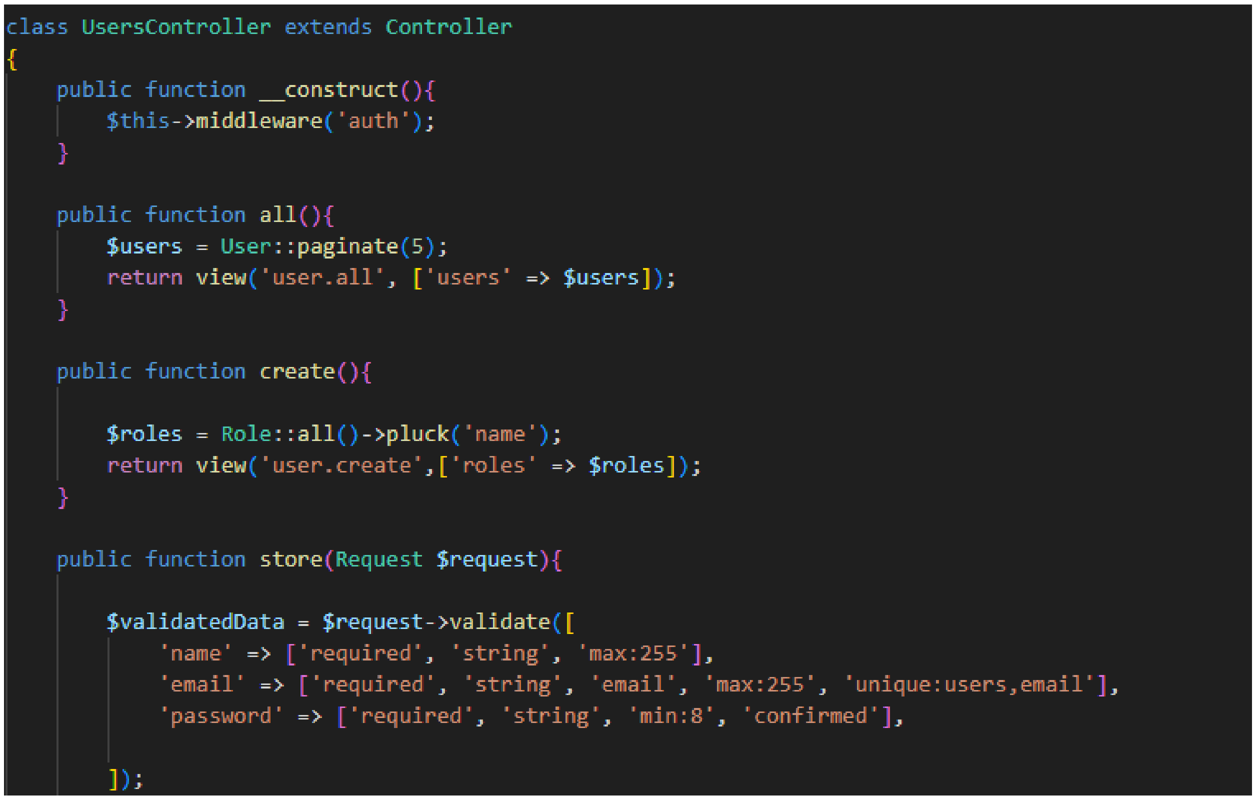
```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use DateTime;
use App\User;
use App\Patient;
use App\Appointment;
use App\Setting;
use Redirect;
use Nexmo;

class AppointmentController extends Controller
{

    public function __construct(){
        $this->middleware('auth');
    }

    public function create(){

        $patients = User::where('role','patient')->get();
        return view('appointment.create', ['patients' => $patients]);
    }

    public function checkslots($date){

        return $this->getTimeSlot($date);
    }


    public function available_slot($date,$start,$end){
        $check = Appointment::where('date',$date)->where('time_start', $start)->where('time_end', $end)->where('visited', '!=', '2')->count();
        if($check == 0){
            return 'available';
        }else{
            return 'unavailable';
        }
    }
```

Figure 39:Appoinment sample



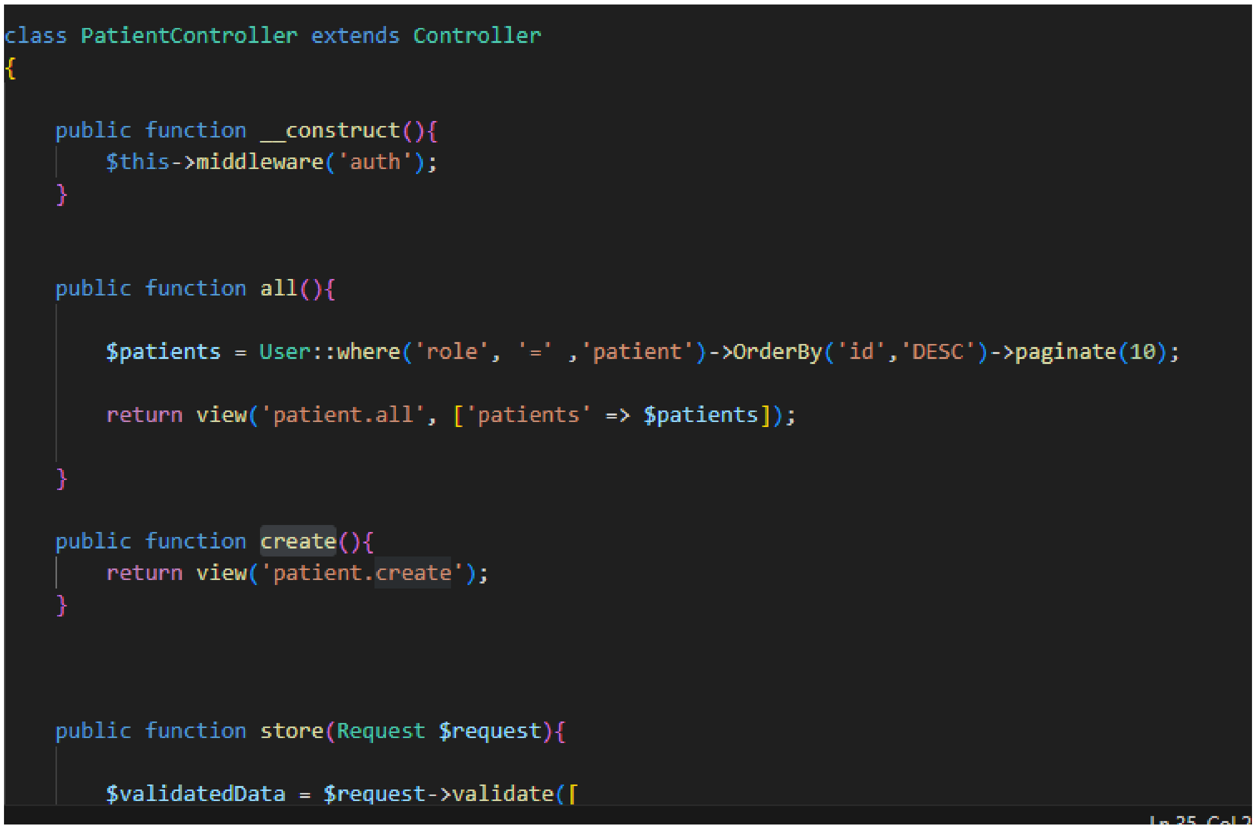
```
namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\User;
use App\Patient;
use App\Prescription;
use App\Appointment;
use App\Billing;
use App\Document;
use App\History;

use Hash;
use Redirect;
use Str;
use Illuminate\Validation\Rule;

class PatientController extends Controller
{

    public function __construct(){
        $this->middleware('auth');
    }


    public function all(){

        $patients = User::where('role', '=' ,'patient')->OrderBy('id','DESC')->paginate(10);

        return view('patient.all', ['patients' => $patients]);
```

Figure 40: Add Patient sample



```
class UsersController extends Controller
{
    public function __construct(){
        $this->middleware('auth');
    }

    public function all(){
        $users = User::paginate(5);
        return view('user.all', ['users' => $users]);
    }

    public function create(){

        $roles = Role::all()->pluck('name');
        return view('user.create',['roles' => $roles]);
    }

    public function store(Request $request){

        $validatedData = $request->validate([
            'name' => ['required', 'string', 'max:255'],
            'email' => ['required', 'string', 'email', 'max:255', 'unique:users,email'],
            'password' => ['required', 'string', 'min:8', 'confirmed'],

        ]);
```

Figure 41:User information sample



```
class PatientController extends Controller
{

    public function __construct(){
        $this->middleware('auth');
    }


    public function all(){

        $patients = User::where('role', '=' ,'patient')->OrderBy('id','DESC')->paginate(10);

        return view('patient.all', ['patients' => $patients]);

    }

    public function create(){
        return view('patient.create');
    }



    public function store(Request $request){

        $validatedData = $request->validate([
```

Figure 42:Patient details sample

10.2 Possible Problem Breakdown

Patient Management System Possible Problem Breakdown:

1. Data Security:
    - Risk of unauthorized access to patient data.
    - Vulnerability to data breaches or cyber attacks.
    - Lack of proper data encryption and access controls.
2. System Performance:
    - Slow response times and system downtime.
    - Inefficient handling of large volumes of data.
    - Lack of scalability to accommodate growing user base.
3. User Experience:
    - Complex and unintuitive user interface.
    - Difficulty in navigating and finding information.
    - Lack of user-friendly features and functionalities.
4. Integration Challenges:
    - Incompatibility with existing healthcare systems.
    - Difficulties in exchanging data with external systems.
    - Lack of interoperability with other healthcare providers.
5. Workflow Efficiency:
    - Inefficient appointment scheduling and management.
    - Delays in accessing patient records and history.
    - Manual and time-consuming administrative tasks.
6. Training and Adoption:
    - Resistance to change among healthcare staff.
    - Insufficient training on using the system effectively.
    - Lack of user buy-in and adoption of the new system.
7. Regulatory Compliance:
    - Non-compliance with data protection regulations.
    - Incomplete or inaccurate recording of patient information.
    - Challenges in maintaining audit trails and documentation.
8. Technical Issues:
    - System compatibility issues with different devices and browsers.
    - Software bugs and glitches affecting system functionality.
    - Insufficient system maintenance and updates.
9. Communication and Collaboration:
    - Inadequate communication channels between healthcare providers and patients.

    - Limited collaboration features for multi-disciplinary care teams.
    - Challenges in sharing information and coordinating care effectively.
10. Cost and Budget Constraints:
    - High implementation and maintenance costs.
    - Limited budget for system upgrades and enhancements.
    - Difficulties in achieving a return on investment.

It's important to note that these problem areas may vary based on the specific context and requirements of the Patient Management System. Identifying and addressing these problems is crucial for the successful implementation and usage of the system.

10.3 Prioritization while Developing the Solution

Prioritizing the development of a Patient Management System involves identifying and categorizing the requirements, features, and functionalities based on their importance and impact on the system's effectiveness and usability. Here is a suggested approach for prioritization:

1. Critical Functionality:
    - Identify the core functionalities that are essential for the system to function effectively. These may include patient registration, appointment scheduling, medical record management, and billing.
2. Regulatory Compliance:
    - Prioritize features and requirements that ensure compliance with healthcare regulations and standards. This includes data privacy and security measures, audit trails, and documentation requirements.
3. User Needs and Experience:
    - Give priority to features that enhance the user experience and usability of the system. Consider user feedback, ease of navigation, intuitive interfaces, and efficient workflows to improve user satisfaction.
4. Patient Safety and Care:
    - Focus on functionalities that contribute to patient safety and quality of care. This may include medication management, allergy alerts, lab result tracking, and care coordination features.
5. Integration and Interoperability:
    - Prioritize requirements that facilitate seamless integration with other healthcare systems and enable interoperability. This includes the ability to exchange data with external systems, such as electronic health records (EHRs) or laboratory systems.
6. Performance and Scalability:

- Give importance to features that ensure system performance, scalability, and reliability. This includes optimizing response times, handling high volumes of data, and ensuring system stability and availability.

7. Reporting and Analytics:
    - Consider features that provide robust reporting and analytics capabilities. This allows for data-driven decision-making, monitoring key performance indicators, and generating meaningful insights for healthcare providers.

Chapter 11 – Testing

11.1 Test Plan Acceptance

Unit Testing

Functional Testing (Unit Testing) is a type of software testing that focuses on testing individual units or components of a system to ensure their proper functionality. In the context of the Patient Management System, unit testing aims to verify the correctness and reliability of individual software units, such as functions, methods, or modules.

Test Class: Functional Testing (Unit Testing) Source of Data: Test data specific to each unit being tested is generated.

Steps in the Testing Process:

1. Test Class Setup:
   - Set up the necessary testing environment, including any required frameworks or tools.
   - Configure the testing environment to isolate the unit being tested and provide appropriate test data.
2. Test Case Identification:
   - Identify the individual units or components to be tested.
   - Create test cases for each unit, covering different scenarios and input combinations.
3. Test Case Execution:
   - Execute each test case by calling the specific unit being tested.
   - Provide relevant input data and parameters to the unit.
   - Capture the actual output or behavior of the unit.
4. Test Case Evaluation:
   - Compare the actual output or behavior of the unit with the expected results defined in the test case.
   - Determine whether the unit functions correctly, produces the expected output, and meets the specified requirements.
   - Identify any discrepancies or failures as defects for further investigation and resolution.
5. Test Case Reporting:
   - Document the test results, including the test case execution status and any identified defects.
   - Generate test reports to summarize the unit testing activities and their outcomes.

- Communicate the test results to the development team for necessary actions.

Expected Outcome: The expected outcome of functional testing (unit testing) is to ensure that individual units of the Patient Management System function correctly and meet the defined requirements. Each test case should pass successfully, indicating that the unit being tested behaves as expected and produces the desired output.

Actual Outcome: The actual outcome of the unit testing depends on the execution of the test cases. If all the test cases pass, it indicates that the individual units of the Patient Management System have been tested successfully, and their functionality has been verified. Any failures or defects identified during testing should be addressed and resolved by the development team.

Module Testing

Functional Testing (Module Testing) is a type of software testing that focuses on testing the functionality and behavior of individual modules or components of a system. In the context of the Patient Management System, module testing aims to verify the correctness and effectiveness of specific modules that perform distinct functions within the system.

Test Class: Functional Testing (Module Testing) Source of Data: Test data specific to each module being tested is generated.

Steps in the Testing Process:

1. Test Class Setup:
   - Set up the necessary testing environment, including any required frameworks or tools.
   - Configure the testing environment to isolate the module being tested and provide appropriate test data.
2. Test Case Identification:
   - Identify the modules or components to be tested based on the functional requirements of the Patient Management System.
   - Create test cases for each module, covering different scenarios and input combinations.
3. Test Case Execution:
   - Execute each test case by invoking the specific module being tested.
   - Provide relevant input data and parameters to the module.
   - Capture the actual output or behavior of the module.
4. Test Case Evaluation:
   - Compare the actual output or behavior of the module with the expected results defined in the test case.

    - Determine whether the module functions correctly, produces the expected output, and meets the specified requirements.
    - Identify any discrepancies or failures as defects for further investigation and resolution.
5. Test Case Reporting:
    - Document the test results, including the test case execution status and any identified defects.
    - Generate test reports to summarize the module testing activities and their outcomes.
    - Communicate the test results to the development team for necessary actions.

Integration Testing

Integration Testing is a software testing technique that focuses on testing the integration and interaction between different modules or components of a system. It ensures that the integrated modules work together as expected and function correctly as a whole. In the context of the Patient Management System, integration testing involves testing the integration between various modules and verifying their seamless interaction.

Test Class: Integration Testing

Description of the Test: The purpose of integration testing is to verify the proper integration and communication between different modules of the Patient Management System. It tests how the modules interact with each other, exchange data, and handle dependencies.

Source of Data: Test data specific to the integration scenarios and module interactions is generated. This data is designed to cover different integration scenarios and ensure comprehensive testing of the system.

Steps in the Testing Process:

1. Identify Integration Points:
    - Identify the key integration points between different modules.
    - Determine the dependencies and interactions between the modules.
2. Define Integration Test Scenarios:
    - Identify and define integration test scenarios that cover various integration points and interactions.
    - Define input data and expected outcomes for each integration scenario.
3. Execute Integration Test Scenarios:
    - Execute the defined integration test scenarios.
    - Provide the necessary input data and parameters to simulate the integration.

    - Capture the actual outputs and behavior of the integrated modules.
4. Compare Actual and Expected Results:
    - Compare the actual results of the integration test scenarios with the expected outcomes.
    - Verify that the integrated modules function correctly and produce the expected results.
    - Identify any inconsistencies or errors that may indicate integration issues.
5. Debug and Fix Integration Issues:
    - If any integration issues or failures are identified, debug the problematic areas.
    - Collaborate with the development team to address the integration issues and resolve them.
    - Retest the affected integration scenarios after the fixes to ensure they function correctly.

Expected Outcome: The expected outcome of integration testing is to ensure that the modules of the Patient Management System integrate successfully, communicate effectively, and produce the desired results when working together. All the defined integration test scenarios should pass, indicating that the system's integration is functioning correctly

Non-functional Testing
Acceptance Testing

Acceptance Testing is a type of software testing conducted to determine whether a system meets the specified requirements and is acceptable for delivery to the end-users. It is performed from the end-user's perspective to ensure that the system meets their needs and functions as expected. In the context of the Patient Management System, acceptance testing focuses on validating that the system meets the requirements of healthcare providers and other stakeholders.

Test Class: Acceptance Testing

Description of the Test: The purpose of acceptance testing is to verify that the Patient Management System meets the specified requirements and is ready for deployment. It involves testing the system's functionality, usability, performance, and adherence to business rules.

Source of Data: Test scenarios and test data are derived from the system requirements, user stories, and user acceptance criteria. Realistic patient data, user profiles, and test configurations may be used during acceptance testing.

Steps in the Testing Process:

1. Define Acceptance Test Scenarios:
   - Identify key functional and non-functional requirements of the system.
   - Define acceptance test scenarios that cover different aspects of the system's functionality and usability.
   - Document the expected outcomes for each acceptance test scenario.
2. Execute Acceptance Test Scenarios:
   - Execute the defined acceptance test scenarios using the Patient Management System.
   - Perform tasks and actions that represent typical user interactions and workflows.
   - Capture the actual results and observe the system's behavior.
3. Compare Actual and Expected Results:
   - Compare the actual results of the acceptance test scenarios with the expected outcomes.
   - Verify that the system functions as intended and meets the specified requirements.
   - Identify any discrepancies or deviations from the expected behavior.
4. Report and Document Issues:
   - Report any defects, bugs, or usability issues encountered during acceptance testing.
   - Provide detailed information about the observed issues, including steps to reproduce and relevant test data.
   - Document the acceptance test results, including any deviations from the expected outcomes.
5. Collaborate for Issue Resolution:
   - Collaborate with the development team to address and resolve the reported issues.
   - Participate in defect triage and prioritization discussions.
   - Retest the fixed issues to ensure they have been successfully resolved.

Expected Outcome: The expected outcome of acceptance testing is to validate that the Patient Management System meets the specified requirements, functions as expected, and is acceptable for deployment to the end-users. The acceptance test scenarios should pass, indicating that the system has been successfully accepted by the stakeholders and is ready for production use.

Security Testing

Security Testing is an essential aspect of software testing, including the Patient Management System, to ensure the system's resilience against potential security vulnerabilities and protect sensitive patient information. It involves identifying and assessing security risks, vulnerabilities, and weaknesses in the system's design and implementation.

Test Class: Security Testing

Description of the Test: The objective of security testing is to evaluate the effectiveness of the security measures implemented in the Patient Management System and identify any potential security vulnerabilities or weaknesses that could compromise the confidentiality, integrity, or availability of patient data. It includes testing various aspects of the system's security, such as authentication, authorization, data protection, and secure communication.

Source of Data: Security testing relies on various sources of data, including system requirements, security standards and guidelines, security best practices, threat models, and industry-specific regulations related to patient data protection (e.g., HIPAA).

Steps in the Testing Process:

1. Identify Security Requirements:
    - Identify the security requirements and constraints applicable to the Patient Management System.
    - Consider relevant security standards, regulations, and industry best practices.
    - Define the security objectives and expected security controls for the system.
2. Perform Security Risk Assessment:
    - Identify potential security risks and threats that the system may be exposed to.
    - Evaluate the impact and likelihood of each identified security risk.
    - Prioritize the risks based on their severity and potential impact on patient data and system operations.
3. Design and Execute Security Test Cases:
    - Design security test cases to verify the effectiveness of security controls and mechanisms.
    - Test authentication and authorization mechanisms to ensure proper access control.
    - Test data protection mechanisms such as encryption, masking, and secure storage.
    - Test secure communication channels, including secure transmission of patient data.
    - Perform penetration testing to identify potential vulnerabilities and exploit them to assess the system's resilience.
4. Analyze and Report Security Findings:
    - Analyze the results of security testing and identify any security vulnerabilities or weaknesses.
    - Document the identified security issues with detailed descriptions, impact assessments, and reproducible steps.
    - Classify the security findings based on their severity and potential impact on the system's security.

    - Provide recommendations and remediation strategies to address the identified security vulnerabilities.
5. Collaborate for Security Improvement:
    - Collaborate with the development team to prioritize and address the reported security issues.
    - Participate in security-related discussions and assist in implementing security fixes and enhancements.
    - Retest the security fixes to ensure they effectively mitigate the identified vulnerabilities.

Expected Outcome: The expected outcome of security testing is to identify and mitigate security vulnerabilities, ensuring the Patient Management System's resilience against potential security threats. The system should demonstrate secure authentication and authorization processes, robust data protection mechanisms, secure communication channels, and compliance with relevant security standards and regulations. The security test results should provide insights into the system's overall security posture and guide the implementation of necessary security improvements.

Accessibility Testing

Accessibility Testing in the context of the Patient Management System involves evaluating the system's usability and accessibility for individuals with disabilities. The goal is to ensure that the system can be effectively used by people with diverse abilities, including those with visual, auditory, motor, or cognitive impairments. Accessibility testing helps identify any barriers or challenges that may prevent users with disabilities from accessing and using the system.

Test Class: Accessibility Testing

Description of the Test: The objective of accessibility testing is to assess the Patient Management System's compliance with accessibility standards and guidelines, such as the Web Content Accessibility Guidelines (WCAG). It involves evaluating the system's user interface, navigation, and interactions to ensure they are accessible to individuals with disabilities. The testing focuses on areas such as keyboard accessibility, screen reader compatibility, color contrast, form labels, and alternative text for images.

Source of Data: Accessibility testing relies on accessibility standards and guidelines, such as WCAG, which provide recommendations and criteria for creating accessible digital content. It also considers the needs and requirements of users with disabilities, including feedback and input from accessibility experts or individuals with disabilities.

Steps in the Testing Process:

1. Understand Accessibility Standards and Guidelines:
    - Familiarize yourself with accessibility standards and guidelines, such as WCAG 2.0 or WCAG 2.1.
    - Understand the accessibility requirements and recommendations for different aspects of the system, such as navigation, forms, multimedia, and color contrast.
2. Evaluate Keyboard Accessibility:
    - Test the system's functionality using keyboard-only navigation.
    - Ensure that all interactive elements can be accessed and operated using keyboard commands.
    - Verify that the keyboard focus is clearly visible and moves in a logical order.
3. Test Screen Reader Compatibility:
    - Use screen reader software, such as NVDA (NonVisual Desktop Access) or JAWS (Job Access With Speech), to navigate and interact with the system.
    - Verify that screen readers can read all the relevant content and provide appropriate feedback.
    - Ensure that screen reader users can access all interactive elements, forms, and multimedia content.
4. Check Color Contrast and Visual Design:
    - Evaluate the color contrast of text and background elements to ensure readability.
    - Verify that color is not the sole means of conveying important information or instructions.
    - Assess the visual design for clarity and consistency, considering factors such as font size, spacing, and layout.
5. Validate Form Accessibility:
    - Test form fields to ensure they have appropriate labels and instructions.
    - Verify that form validation and error messages are communicated clearly to all users, including those using screen readers.
    - Check that form fields are properly associated with their labels for assistive technology users.

Usability Testing

Usability Testing in the context of the Patient Management System involves evaluating the system's user interface and overall user experience to ensure that it is intuitive, efficient, and user-friendly. The goal is to identify any usability issues or challenges that may impact user satisfaction and productivity. Usability testing helps gather feedback from representative users, uncover usability problems, and make informed design decisions to enhance the system's usability.

Test Class: Usability Testing

Description of the Test: The objective of usability testing is to assess the usability and user experience of the Patient Management System. It involves observing and gathering feedback from users as they perform specific tasks or scenarios within the system. Usability testing focuses on aspects such as system navigation, information organization, task completion, error handling, and overall user satisfaction.

Source of Data: Usability testing relies on a combination of user feedback, observations, and usability best practices. It involves recruiting representative users who have a genuine interest in or experience with patient management systems. Feedback can be obtained through interviews, surveys, or direct observations during testing sessions.

Steps in the Testing Process:

1. Define Usability Test Goals and Scenarios:
    - Clearly define the goals and objectives of the usability testing.
    - Identify key tasks or scenarios that users will perform during the testing.
    - Develop realistic and relevant scenarios that align with typical user workflows in the Patient Management System.
2. Recruit Test Participants:
    - Identify and recruit a diverse group of test participants who represent the target user population.
    - Ensure that the participants have a good understanding of the system's purpose and are able to provide meaningful feedback.
3. Conduct Usability Testing Sessions:
    - Brief participants on the purpose and process of usability testing.
    - Instruct participants to perform specific tasks or scenarios within the Patient Management System.
    - Encourage participants to think aloud and provide feedback as they interact with the system.
    - Observe and record participants' interactions, noting any usability issues, confusion, or difficulties encountered.
4. Gather User Feedback:
    - Conduct post-task interviews or surveys to gather participants' feedback and impressions.
    - Ask open-ended questions to understand participants' experiences, challenges, and suggestions for improvement.
    - Document and categorize the feedback, noting recurring themes or issues raised by multiple participants.
5. Analyze Usability Findings:
    - Review and analyze the collected data, including observations, participant feedback, and task completion rates.

- Identify common usability issues and prioritize them based on their impact on user experience and system usability.
- Use qualitative and quantitative analysis methods to uncover patterns and insights from the usability test results.

6. Generate Usability Recommendations:
    - Based on the identified usability issues, develop actionable recommendations for improving the system's usability.
    - Prioritize the recommendations based on their potential impact and feasibility of implementation.
    - Provide specific design suggestions or changes that address the usability issues and enhance the user experience.
7. Iterate and Improve:
    - Collaborate with the development team to implement the recommended usability improvements.
    - Conduct additional rounds of usability testing as needed to validate the effectiveness of the changes.
    - Continuously iterate and refine the system's design based on user feedback and usability test results.

Expected Outcome: The expected outcome of usability testing is to uncover usability issues, gather user feedback, and generate actionable recommendations for improving the Patient Management System's usability. The test results should provide insights into users' experiences, pain points, and suggestions for enhancing the system's usability. By addressing the identified usability issues and implementing the recommended improvements, the system can provide a more intuitive and user-friendly experience, leading to increased user satisfaction and efficiency.

11.2 Test Case

Test cases must be prepared when the test acceptance strategy has been completed. The IUMS system's test cases are listed below.

Unit test – test case:

Name of the test case	**Unit testing**
Test Class	
Description of the test	

Source of Data	Steps in the Testing Process	Expected Outcome	Actual Outcome

Module Test – test case:

Name of the test case	**Module Test**
Test Class	
Description of the test	

Source of Data	Steps in the Testing Process	Expected Outcome	Actual Outcome

Integration Testing – test case:

Name of the test case	**Integration Test**
Test Class	
Description of the test	

Source of Data	Steps in the Testing Process	Expected Outcome	Actual Outcome

11.3 Unit Testing

Test Case

Name of the test case	**Unit Test**
Test Class	Unit test
Description of the test	Unit testing is a software testing technique that focuses on verifying the correctness of individual units or components of the system. In the context of the Patient Management System, unit testing involves testing the smallest testable parts of the code, such as functions, methods, or classes, in isolation to ensure they function as intended.

Source of Data	**Steps in the Testing Process**	**Expected Outcome**	**Actual Outcome**

The source of data for unit testing includes the source code of the Patient Management System, test cases specifically designed to cover different scenarios, and any relevant documentation or requirements.	Test Case Preparation: ● Identify the specific units or components to be tested based on the code structure and functionality. ● Create test cases that cover different scenarios, including positive and negative test cases, boundary conditions, and edge cases. ● Define the expected outcomes for each test case. Test Environment Setup: ● Set up the necessary development environment, including the required libraries, frameworks, and dependencies. ● Ensure that the test environment is isolated from external dependencies and can accurately simulate the behavior of the unit under	The expected outcome of unit testing is to validate the individual units or components of the Patient Management System in isolation and ensure their correctness and functionality. The test cases should cover different scenarios and edge cases to detect any defects or errors in the code. The expected outcome is that the units pass the test cases, meaning their behavior aligns with the defined requirements and expectations.	The actual outcome of unit testing depends on the specific implementation and execution of the test cases. It can vary from unit to unit and test case to test case. The actual outcome should be recorded for each test case, and any discrepancies or failures should be investigated and addressed.

Figure 43:unit test one test case

Unit Test two

Name of the test case	**Unit Test**
Test Class	Unit test
Description of the test	Unit testing is a software testing technique that focuses on verifying the correctness of individual units or components of the system. In the context of the Patient Management System, unit testing involves testing the smallest testable parts of the code, such as functions, methods, or classes, in isolation to ensure they function as intended.

Source of Data	**Steps in the Testing Process**	**Expected Outcome**	**Actual Outcome**

The source of data for unit testing includes the source code of the Patient Management System, test cases specifically designed to cover different scenarios, and any relevant documentation or requirements.	Test Case Preparation: ● Identify the specific units or components to be tested based on the code structure and functionality. ● Create test cases that cover different scenarios, including positive and negative test cases, boundary conditions, and edge cases. ● Define the expected outcomes for each test case. Test Environment Setup: ● Set up the necessary development environment, including the required libraries, frameworks, and dependencies. ● Ensure that the test environment is isolated from external dependencies and can accurately simulate the behavior of the unit under	The expected outcome of unit testing is to validate the individual units or components of the Patient Management System in isolation and ensure their correctness and functionality. The test cases should cover different scenarios and edge cases to detect any defects or errors in the code. The expected outcome is that the units pass the test cases, meaning their behavior aligns with the defined requirements and expectations.	The actual outcome of unit testing depends on the specific implementation and execution of the test cases. It can vary from unit to unit and test case to test case. The actual outcome should be recorded for each test case, and any discrepancies or failures should be investigated and addressed.

Figure 44 unit test two test case

Unit Test three

Name of the test case	**Unit Test**
Test Class	Unit test
Description of the test	This unit test aims to validate the functionality and correctness of the "calculateBMI" method in the "Patient" class of the Patient Management System. The "calculateBMI" method calculates the Body Mass Index (BMI) of a patient based on their height and weight.

Source of Data	**Steps in the Testing Process**	**Expected Outcome**	**Actual Outcome**

Test cases with different input values for height and weight. Patient class source code.	Test Case Preparation: ○ Define multiple test cases with different input values for height and weight, covering various scenarios (e.g., normal values, edge cases, invalid values). ○ Determine the expected BMI values for each test case based on the provided input data. Test Environment Setup: ○ Set up the developmen t environment with the necessary tools and libraries for unit testing. ○ Ensure the Patient class and its dependencie s are properly imported and accessible. Execute Test Cases: ○ For each	For each test case, the expected outcome is that the calculated BMI value matches the expected BMI value based on the given input data. The "calculateBMI" method should accurately calculate the BMI using the provided height and weight values.	The actual outcome of the unit test depends on the implementation of the "calculateBMI" method and the specific test cases executed. If the actual BMI values match the expected values for all test cases, it indicates that the "calculateBMI" method is functioning correctly. If any discrepancies or failures occur, further investigation and debugging are required to identify and fix any issues.

Figure 45 :unit test three test case

Unit Test four

Name of the test case	**Unit Test**
Test Class	Unit Test
Description of the test	This unit test aims to validate the functionality and correctness of the "calculateTotalPrice" method in the "ShoppingCart" class of an e-commerce application. The "calculateTotalPrice" method calculates the total price of the items in the shopping cart, including any applicable discounts or promotions.

Source of Data	**Steps in the Testing Process**	**Expected Outcome**	**Actual Outcome**

Test cases with different items and quantities in the shopping cart. ShoppingCart class source code.	Test Case Preparation: ● Define multiple test cases with different items, quantities, and prices in the shopping cart, covering various scenarios (e.g., single item, multiple items, discounts). ● Determine the expected total price for each test case based on the provided input data. Test Environment Setup: ● Set up the development environment with the necessary tools and libraries for unit testing. ● Ensure the ShoppingCart class and its dependencies are properly imported and accessible. Execute Test Cases: ● For each test case: ○ Create an instanc	For each test case, the expected outcome is that the calculated total price matches the expected total price based on the given input data. The "calculateTotalPrice" method should accurately calculate the total price of the items in the shopping cart, considering any applicable discounts or promotions.	The actual outcome of the unit test depends on the implementation of the "calculateTotalPrice" method and the specific test cases executed. If the actual total prices match the expected prices for all test cases, it indicates that the "calculateTotalPrice" method is functioning correctly. If any discrepancies or failures occur, further investigation and debugging are required to identify and fix any issues.

Figure 46:unit test four test case

11.4 Module Testing

Module Test one

Test Case

Name of the test case	**Module Testing**
Test Class	Module
Description of the test	This module test aims to verify the functionality and integration of the "UserManagement" module in the Patient Management System. The "UserManagement" module handles user registration, authentication, and access control within the system.

Source of Data	**Steps in the Testing Process**	**Expected Outcome**	**Actual Outcome**

Test data for user registration and authentication scenarios. UserManagement module source code.	1. Test Case Preparation: ○ Define test cases to cover various scenarios related to user registration, authentication, and access control. ○ Prepare test data, including valid and invalid user credentials, user roles, and permissions. 2. Test Environment Setup: ○ Set up the testing environment with the necessary infrastructure, including the database and any required dependencies for the UserManagement module. ○ Ensure the UserManagement module and its dependencies are properly configured and accessible. 3. Execute Test Cases: ○ For each test case: ■ Set up the initial state, such as pre-existing users,	The expected outcome is that the UserManagement module functions correctly and performs user registration, authentication, and access control operations accurately. The test cases should cover different scenarios and verify that the module behaves as intended, handling valid and invalid user credentials, user roles, and permissions appropriately.	The actual outcome of the module test depends on the implementation of the UserManagement module and the specific test cases executed. If the actual results align with the expected outcomes for all test cases, it indicates that the module is functioning correctly. Any discrepancies, failures, or errors encountered during the test require further investigation and debugging to identify and address the issues.

Figure 47:Module Test One

Module Test two
Test Case

Name of the test case	**Module Testing**
Test Class	**Module** Test
Description of the test	This module test aims to verify the functionality and integration of the "AppointmentManagement" module in the Patient Management System. The "AppointmentManagement" module handles scheduling, rescheduling, and cancellation of patient appointments.

Source of Data	**Steps in the Testing Process**	**Expected Outcome**	**Actual Outcome**

Test data for appointment scheduling, rescheduling, and cancellation scenarios. AppointmentManagement module source code.	1. Test Case Preparation: ○ Define test cases to cover various scenarios related to appointment scheduling, rescheduling, and cancellation. ○ Prepare test data, including patient information, available time slots, and appointment details. 2. Test Environment Setup: ○ Set up the testing environment with the necessary infrastructure, including the database and any required dependencies for the AppointmentManagement module. ○ Ensure the AppointmentManagement module and its dependencies are properly configured and accessible. 3. Execute Test Cases: ○ For each test case: ■ Set up the initial state, such as pre-existing	The expected outcome is that the AppointmentManagement module functions correctly and performs appointment scheduling, rescheduling, and cancellation operations accurately. The test cases should cover different scenarios and verify that the module handles appointment conflicts, updates the appointment status correctly, and maintains the integrity of the appointment schedule.	The actual outcome of the module test depends on the implementation of the AppointmentManagement module and the specific test cases executed. If the actual results align with the expected outcomes for all test cases, it indicates that the module is functioning correctly. Any discrepancies, failures, or errors encountered during the test require further investigation and debugging to identify and address the issues.

Figure 48:Module Test two

11.5 Integration Testing

Integration Test one

Test Case

Name of the test case	**Integration Testing**
Test Class	**Integration** Test
Description of the test	This integration test aims to verify the interaction and integration between the "AppointmentManagement" module and the "PatientRecords" module in the Patient Management System. It ensures that the two modules work together correctly and exchange data as expected.

Source of Data	**Steps in the Testing Process**	**Expected Outcome**	**Actual Outcome**

Test data for patient records and appointment details. Source code of the AppointmentManagement and PatientRecords modules.	1. Test Case Preparation: ○ Define test cases that cover various scenarios involving the interaction between the AppointmentManagement and PatientRecords modules. ○ Prepare test data, including patient records, appointment details, and any required dependencies. 2. Test Environment Setup: ○ Set up the testing environment with the necessary infrastructure, including the database and any required dependencies for the modules. ○ Ensure both the AppointmentManagement and PatientRecords modules are properly configured and accessible. 3. Execute Test Cases: ○ For each test case:	The expected outcome is that the AppointmentManagement and PatientRecords modules integrate seamlessly and function correctly together. The test cases should cover different scenarios, such as creating an appointment for an existing patient, updating patient records after an appointment, and handling any dependencies or data consistency requirements between the modules.	The actual outcome of the integration test depends on the implementation of the AppointmentManagement and PatientRecords modules and the specific test cases executed. If the actual results align with the expected outcomes for all test cases, it indicates that the integration between the modules is successful. Any discrepancies, failures, or errors encountered during the test require further investigation and debugging to identify and address the issues..

Figure 49:Integration Test one

Integration Test two

Test Case

Name of the test case	**Integration Testing**
Test Class	**Integration Testing**
Description of the test	The integration test focuses on testing the integration and interaction between the "Billing" module and the "Inventory" module in the Patient Management System. It aims to ensure that the two modules work together seamlessly and exchange data accurately for billing and inventory management purposes.

Source of Data	**Steps in the Testing Process**	**Expected Outcome**	**Actual Outcome**

Test data for billing transactions and inventory items. Source code of the Billing and Inventory modules.	1. Test Case Preparation: ○ Identify and define test cases that cover various integration scenarios between the Billing and Inventory modules. ○ Prepare test data, including sample billing transactions and inventory items, to simulate realistic scenarios. 2. Test Environment Setup: ○ Set up the testing environment, including the necessary infrastructure such as databases, servers, and any dependencies required by the modules. ○ Configure the Billing and Inventory modules and ensure they are accessible and properly integrated. 3. Execute Test Cases: ○ For each test case: ■ Set up the initial	The expected outcome is that the Billing and Inventory modules integrate seamlessly, ensuring accurate billing and inventory management. The test cases should cover various scenarios, such as creating new billing transactions, updating inventory quantities, handling inventory shortages, and maintaining data consistency between the modules.	The actual outcome of the integration test depends on the implementation of the Billing and Inventory modules and the specific test cases executed. If the actual results align with the expected outcomes for all test cases, it indicates successful integration between the modules. Any discrepancies, failures, or errors encountered during the test require further investigation and debugging to identify and address the issues.

Figure 50:Integration Test two

11.6 Acceptance Testing

Acceptance Test one

Test Case

Name of the test case	**Acceptance Testing**
Test Class	**Acceptance** Test
Description of the test	The acceptance test focuses on verifying that the Patient Management System meets the specified requirements and is ready for deployment and use. It is performed from the perspective of the end user or customer to ensure that the system functions as intended and meets their needs.

Source of Data	**Steps in the Testing Process**	**Expected Outcome**	**Actual Outcome**

User requirements and acceptance criteria defined during the system development process. Realistic test data representing typical user scenarios.	1. Test Case Preparation: ○ Identify and define acceptance test cases based on user requirements and acceptance criteria. ○ Prepare test data that reflects realistic user scenarios, such as creating patient records, scheduling appointments, and generating reports. 2. Test Environment Setup: ○ Set up the testing environment to mimic the production environment as closely as possible. ○ Ensure that all necessary system components, databases, and dependencies are properly configured and accessible. 3. Execute Test Cases: ○ For each test case: ■ Perform user actions or	The expected outcome of the acceptance test is that the Patient Management System meets the defined user requirements and acceptance criteria. The system should demonstrate the ability to effectively manage patient records, appointments, and generate accurate reports. User interactions should be smooth, intuitive, and free from major functional or usability issues	The actual outcome of the acceptance test depends on the implementation of the Patient Management System and the specific test cases executed. If the actual results align with the expected outcomes for all test cases, it indicates that the system meets the user requirements and is ready for deployment. Any discrepancies or failures identified during the test should be addressed and resolved before finalizing the system for production use.

Figure 51:Acceptance Test

11.7 Security Testing

Security Test

Test Case

Name of the test case	**Security Testing**
Test Class	Security Test
Description of the test	The security test aims to assess the security measures and controls implemented in the Patient Management System to protect sensitive patient information and prevent unauthorized access or data breaches. It involves identifying and addressing vulnerabilities and ensuring that the system adheres to industry best practices for security.

Source of Data	**Steps in the Testing Process**	**Expected Outcome**	**Actual Outcome**

Security requirements and guidelines defined during the system development process. Security policies, regulations, and standards applicable to the healthcare industry. Security testing tools and techniques.	1. Vulnerability Assessment: ○ Conduct a comprehensive review of the Patient Management System's architecture, code, and configurations to identify potential security vulnerabilities. ○ Use automated scanning tools and manual analysis to detect common security flaws such as cross-site scripting (XSS), SQL injection, insecure authentication mechanisms, and inadequate access controls. 2. Authentication and Authorization Testing: ○ Test the system's authentication and authorization mechanisms to ensure only authorized users can access sensitive patient data. ○ Validate the	The expected outcome of the security test is to identify and address any security vulnerabilities or weaknesses in the Patient Management System. The system should demonstrate robust security controls, such as secure authentication, data encryption, access controls, and proper handling of errors and logs. It should comply with relevant security standards and regulations, protecting patient data from unauthorized access or disclosure.	The actual outcome of the security test depends on the specific vulnerabilities and security controls implemented in the Patient Management System. If the system successfully detects and mitigates security vulnerabilities, adheres to best practices, and meets the defined security requirements, it indicates a strong security posture. Any identified security issues should be addressed and remediated to ensure the system's overall security and protection of patient data.

Figure 52: Security Testing

11.8 Accessibility Testing

Accessibility Test

Test Case

Name of the test case	**Accessibility Testing**
Test Class	AccessibilityTest
Description of the test	The accessibility test aims to evaluate the Patient Management System's compliance with accessibility standards and guidelines, ensuring that individuals with disabilities can access and use the system effectively. It involves identifying barriers and issues that may hinder accessibility and verifying that the system provides appropriate support for assistive technologies.

Source of Data	**Steps in the Testing Process**	**Expected Outcome**	**Actual Outcome**

Accessibility standards and guidelines, such as WCAG (Web Content Accessibility Guidelines) or Section 508. Accessibility requirements defined during the system development process. User feedback and input from individuals with disabilities. Accessibility testing tools and techniques.	1. Keyboard Accessibility: ○ Test the system's functionality using only the keyboard navigation, ensuring that all interactive elements and functionalities are accessible and operable through keyboard inputs. ○ Verify that keyboard focus is clearly visible and moves in a logical and predictable order. 2. Screen Reader Compatibility: ○ Use screen reader software (e.g., JAWS, NVDA) to assess how effectively the system communicates information to individuals who are blind or visually impaired. ○ Verify that all essential information, including labels, form fields, and error messages, is properly read and conveyed by the screen reader.	The expected outcome of the accessibility test is to identify any accessibility barriers or issues in the Patient Management System and ensure that it conforms to relevant accessibility standards and guidelines. The system should be usable by individuals with disabilities, including those who rely on assistive technologies. It should provide appropriate alternative text, keyboard accessibility, proper color contrast, and clear communication of information.	The actual outcome of the accessibility test depends on the system's implementation and adherence to accessibility standards. If the system successfully addresses accessibility barriers, provides support for assistive technologies, and meets the defined accessibility requirements, it indicates a high level of accessibility. Any identified accessibility issues should be addressed and remediated to ensure a more inclusive and accessible user experience for all users, including those with disabilities.

Table 24:Accessibility Testing

Chapter 12 – Implementation

12.1 Training

SL No.	User	Training Scope	Time Period	Comment

01.	Healthcare professionals, administrative staff, and other relevant personnel who will be using the Patient Management System.	he training will cover the usage and functionality of the Patient Management System, including: 1. User roles and permissions 2. Navigating the system interface 3. Entering and updating patient information 4. Scheduling appointments and managing calendars 5. Managing medical records and documentation 6. Generating reports and analytics 7. Interacting with other system modules (e.g., billing, inventory management) 8. Handling common system tasks and troubleshooting	The training program is expected to span a specific duration based on the complexity of the system and the level of user proficiency required. The time period may range from a few days to a couple of weeks, considering both theoretical and hands-on training components.	User training plays a crucial role in ensuring the successful adoption and utilization of the Patient Management System. It is important to tailor the training program to the specific needs and prior knowledge of the users. The training should be interactive, engaging, and provide ample opportunities for users to practice using the system. Additionally, incorporating real-life scenarios and case studies can help users understand how to effectively use the system in their daily tasks. Ongoing support and refresher training sessions may also be necessary to address any new system updates or user-specific needs that arise over time.

Table 25:User training

12.2 Implementation Scheme

Big Bang

In the context of software development, the Big Bang implementation scheme refers to a deployment strategy where the entire system is implemented in a single, comprehensive release. It involves launching all the modules, features, and functionalities of the Patient Management System at once, without any phased or incremental approach.

Key Characteristics of the Big Bang Implementation Scheme:

1. Comprehensive Release: The entire Patient Management System, including all its modules, is implemented and made available to users simultaneously.
2. High Complexity: The Big Bang approach is typically employed for systems with a moderate to high level of complexity, where dependencies between modules are manageable and the risk associated with the simultaneous launch is acceptable.
3. Limited Testing in Isolation: Due to the simultaneous implementation, there may be limited opportunities to test individual modules in isolation. Testing is often performed in an integrated manner, focusing on overall system functionality and performance.
4. Immediate System Availability: Once the implementation is complete, the system is immediately made available to users for their regular operations. This approach allows for a quick transition from the old system to the new one.
5. High Risk and Impact: The Big Bang implementation carries a higher risk compared to phased or incremental approaches since any issues or failures affecting the entire system can have a significant impact on operations.
6. Minimal Rollback Options: In case of any major issues or failures, rolling back the entire system to a previous state may be challenging or impractical. Contingency plans and backup measures should be in place to mitigate risks.

12.3 Scaling

Scaling refers to the process of increasing the capacity and capability of a system to handle larger workloads, accommodate more users, and deliver higher performance. In the context of the Patient Management System, scaling is crucial to ensure that the system can effectively handle the growing demands and requirements as the user base and data volume increase.

There are several aspects to consider when implementing scaling in the Patient Management System:

1. Vertical Scaling: Vertical scaling, also known as scaling up, involves upgrading the hardware resources of the system to enhance its performance. This can include increasing the processing power, memory capacity, or storage capacity of the servers hosting the system. Vertical scaling allows for handling increased user loads and processing larger datasets by leveraging more powerful hardware.
2. Horizontal Scaling: Horizontal scaling, also known as scaling out, involves adding more servers or instances to distribute the workload across multiple machines. This approach allows for better load balancing and improved system performance by dividing the processing tasks among multiple servers. Horizontal scaling is typically achieved through techniques like load balancers and clustering.
3. Database Scaling: As the Patient Management System deals with a significant amount of data, scaling the database infrastructure becomes essential. This can be achieved through techniques such as database sharding, which involves dividing the database into smaller partitions distributed across multiple servers, or implementing database replication to ensure data availability and redundancy.
4. Caching and Content Delivery Networks (CDNs): Implementing caching mechanisms and utilizing CDNs can help improve the performance and responsiveness of the Patient Management System. Caching involves storing frequently accessed data in memory or fast storage for quick retrieval, reducing the need to fetch data from the underlying database. CDNs, on the other hand, distribute static content (e.g., images, CSS files) across geographically dispersed servers, reducing latency and improving user experience.

12.4 Load Balancing

Load balancing is a technique used in distributed systems to distribute incoming workload across multiple resources, such as servers or computing nodes, to ensure efficient resource utilization and improve system performance. In the context of the Patient Management System, load balancing plays a crucial role in handling user requests and distributing the processing load across multiple servers or instances.

Chapter 13 – Critical Appraisal and Evaluation

13.1 Objective Could be Met

In the context of critical appraisal and evaluation, the objective of determining whether the set objectives could be met in the Patient Management System involves assessing the effectiveness and feasibility of the system in achieving its intended goals. This evaluation helps stakeholders understand whether the implemented system aligns with the desired outcomes and meets the identified objectives.

To evaluate if the objectives could be met in the Patient Management System, the following steps can be taken:

1. Review the defined objectives: Start by revisiting the objectives that were initially set for the Patient Management System. These objectives should be specific, measurable, achievable, relevant, and time-bound (SMART). Ensure that the objectives are well-defined and align with the overall goals of the system.
2. Assess system functionality: Evaluate the functionality of the Patient Management System and determine if it adequately addresses the identified objectives. Examine how the system handles various patient management tasks, such as appointment scheduling, medical record management, billing, and reporting. Verify if the system supports the required features and processes to meet the defined objectives.
3. Measure performance metrics: Establish key performance metrics or indicators that can be used to assess the system's performance in relation to the objectives. For example, metrics such as response time, accuracy of data entry, efficiency of appointment scheduling, or reduction in paperwork can be considered. Collect relevant data and compare it against the defined targets or benchmarks to gauge the system's effectiveness.
4. Gather user feedback: Engage with the users of the Patient Management System, including administrators, healthcare professionals, and staff, to gather their feedback and insights. Conduct interviews, surveys, or user testing sessions to understand their satisfaction levels, challenges faced, and if they perceive the system to be meeting the defined objectives. User feedback is valuable in identifying areas for improvement and ensuring user acceptance and adoption of the system.
5. Analyze system performance: Evaluate the system's performance against the defined objectives and identify any gaps or areas of improvement. Determine if the system is meeting the intended outcomes in terms of improved patient management, increased efficiency, reduced errors, enhanced data security, or other relevant criteria. Consider any constraints or limitations that may impact the system's ability to fully meet the objectives.

Objective-1
Achievement rate and others

Objective-1: Achievement Rate and Others

Description: The objective is to assess and evaluate the achievement rate of the Patient Management System, as well as other relevant factors that contribute to its overall effectiveness and success. This objective focuses on measuring the system's performance in terms of meeting predefined targets, goals, and key performance indicators.

Possible Evaluation Criteria:

1. Achievement of targets: Assess whether the Patient Management System has achieved the predefined targets set for various aspects, such as appointment scheduling, patient record management, billing accuracy, and reporting efficiency.
2. User satisfaction: Gather feedback from system users, including administrators, healthcare professionals, and staff, to gauge their satisfaction levels with the system. Evaluate if the system meets their needs, improves their workflow, and enhances overall user experience.
3. Efficiency and productivity: Measure the system's impact on the efficiency and productivity of healthcare operations. Assess if the system has reduced manual tasks, streamlined processes, and improved overall workflow, resulting in time and cost savings.

Objective-2
Achievement rate and others

Objective-2: Achievement Rate and Others

Description: The objective is to assess and evaluate the achievement rate of specific goals and objectives related to the Patient Management System, as well as other relevant factors that contribute to its success. This objective focuses on measuring the system's performance in meeting key milestones, project deliverables, and user requirements.

Possible Evaluation Criteria:

1. Goal achievement: Assess the extent to which the Patient Management System has achieved the defined goals and objectives, such as improved patient care, streamlined administrative processes, enhanced data accuracy, or increased operational efficiency.

2. Project deliverables: Evaluate the completion and quality of project deliverables, such as software modules, user interfaces, database integration, reporting functionalities, or system documentation.
3. User satisfaction: Gather feedback from system users, including healthcare professionals, administrators, and staff, to assess their satisfaction levels with the system's functionality, usability, and overall performance.

Objective-3
Achievement rate and others

Objective-3: Achievement Rate and Others

Description: The objective is to evaluate the achievement rate of specific goals and objectives related to the project, as well as assess other factors that contribute to the project's success. This objective focuses on measuring the progress and outcomes of the project to determine its level of success and identify areas for improvement.

Possible Evaluation Criteria:

1. Key Performance Indicators (KPIs): Define and measure relevant KPIs to assess the achievement rate of project goals and objectives. These KPIs could include metrics such as project timeline adherence, budget management, quality standards compliance, and customer satisfaction.
2. Deliverable completion: Evaluate the completion and quality of project deliverables according to the defined specifications and requirements. This includes assessing the functionality, usability, and reliability of the deliverables.
3. Resource utilization: Assess the efficient utilization of project resources, including human resources, budget, and equipment. This includes evaluating resource allocation, productivity levels, and cost-effectiveness.

Objective-4
Achievement rate and others

Objective-4: Achievement Rate and Others

Description: The objective is to assess the achievement rate of specific goals and objectives of the project, along with other factors that contribute to the overall success of the project. This objective focuses on evaluating the project's performance and outcomes to ensure that it aligns with the intended objectives and meets the expectations of stakeholders.

Possible Evaluation Criteria:

1. Goal attainment: Evaluate the extent to which the project has achieved its defined goals and objectives. This includes assessing whether the project deliverables meet the specified requirements and whether the desired outcomes and benefits have been realized.
2. Stakeholder satisfaction: Measure the satisfaction levels of project stakeholders, including clients, end-users, and other relevant parties. This can be done through surveys, feedback sessions, or interviews to gather their perspectives on the project's performance and outcomes.
3. Quality assurance: Assess the adherence to quality standards and processes throughout the project lifecycle. This includes evaluating the effectiveness of quality control measures, testing procedures, and documentation practices to ensure that the project outputs meet the expected quality levels.

13.2 Objective that totally don’t meet of touched

Objective: Complete Non-Achievement or Neglected Areas

Description: This objective focuses on identifying specific goals or objectives of the Patient Management System project that have not been achieved or have been completely neglected. It aims to highlight areas where the project has fallen short in meeting its intended outcomes.

Evaluation Criteria:

1. Functional requirements: Identify any functional requirements that have not been implemented or fully addressed in the Patient Management System. This could include missing features, incomplete functionalities, or unmet user needs.
2. Performance goals: Evaluate whether the performance goals set for the system, such as response time, scalability, or reliability, have not been achieved or have been significantly compromised.
3. Data management: Assess if the system fails to adequately manage patient data, such as inaccurate data entry, data loss, or security vulnerabilities.

The reasons why it could not be touch

There can be various reasons why certain objectives or areas of a project, such as the Patient Management System, may not have been touched or achieved. Some possible reasons include:

1. Limited resources: The project may have faced constraints in terms of budget, time, or available resources, which hindered the ability to fully address all objectives. This could include limitations in staffing, technology infrastructure, or financial support.

2. Changing priorities: The project may have experienced shifts in priorities, where certain objectives were deprioritized or deemed less critical compared to other pressing needs. This could be due to changing business requirements, stakeholder preferences, or external factors impacting the project.
3. Scope creep: The project scope may have expanded beyond the initial plan, leading to a lack of focus on specific objectives. This can occur when new requirements or features are introduced during the project lifecycle, resulting in limited attention to previously defined objectives.

What could have been done

In situations where certain objectives or areas of a project were not touched, there are several actions that could be considered to address the situation:

1. Review and prioritize objectives: Conduct a thorough review of all objectives and prioritize them based on their importance, alignment with project goals, and potential impact. This will help ensure that the most critical objectives receive adequate attention and resources.
2. Assess resource availability: Evaluate the available resources, including budget, time, and personnel, and determine if there are any opportunities to allocate additional resources to address the untouched objectives. This could involve reallocation of resources, seeking additional funding, or extending the project timeline if feasible.
3. Conduct a gap analysis: Identify the specific gaps or areas that were not addressed and perform a detailed analysis to understand the reasons behind it. This analysis can help uncover any underlying issues, such as technical limitations, resource constraints, or scope creep, and guide the development of targeted solutions.
4. Stakeholder engagement and communication: Engage with project stakeholders, including clients, end-users, and project team members, to communicate the challenges faced and the impact on the untouched objectives. Collaboratively explore potential solutions, gather feedback, and seek input on prioritization to ensure alignment with stakeholder expectations.
5. Adjust project plan: If feasible, revise the project plan to include the untouched objectives. This may involve modifying the scope, timeline, or resource allocation to accommodate the additional work. Careful consideration should be given to the impact of these changes on other aspects of the project.
6. Seek expertise or external support: If certain objectives require specialized skills or knowledge that is lacking within the project team, consider seeking external expertise or support. This could involve hiring consultants, engaging with subject matter experts, or partnering with external organizations to bridge the knowledge or resource gaps.

7. Continuous improvement and lessons learned: Document and analyze the challenges faced in addressing the untouched objectives as part of the project's lessons learned. Identify areas for improvement in future projects, update processes or methodologies, and implement corrective actions to prevent similar issues from arising.

By taking these actions, the project team can proactively address the untouched objectives and work towards their successful completion. It's important to approach the situation with flexibility, adaptability, and open communication to ensure the best possible outcome for the project.

Chapter 14 Lessons Learned

14.1 Pre-Project-Review-closing

During the pre-project review and closing phase of the Patient Management System, several key lessons were learned. These lessons can help guide future projects and improve the overall project management process. Here are some key takeaways:

1. Clear Project Scope: It is important to define and communicate the project scope early on. This helps to avoid scope creep and ensures that everyone involved understands the objectives and deliverables.
2. Stakeholder Involvement: Engaging stakeholders throughout the project lifecycle is crucial. Regular communication, feedback, and involvement of key stakeholders help ensure that their requirements and expectations are met.
3. Effective Requirements Gathering: Thorough and accurate requirements gathering is essential for project success. It is important to involve all relevant stakeholders and capture their needs, priorities, and constraints to develop a comprehensive set of requirements.
4. Agile Project Management: Adopting an agile project management approach can enhance flexibility and adaptability. Agile methodologies, such as DSDM, enable iterative development, frequent feedback, and collaboration, allowing for better responsiveness to changing requirements.
5. Risk Management: Proactive identification, assessment, and mitigation of project risks are essential. Establishing a robust risk management plan helps anticipate potential challenges and develop strategies to minimize their impact.
6. Quality Assurance: Implementing a strong quality assurance process ensures that the developed solution meets the required standards and specifications. Conducting thorough testing, both functional and non-functional, helps identify and address any defects or issues.
7. Documentation and Knowledge Sharing: Maintaining detailed documentation throughout the project facilitates knowledge sharing, future enhancements, and support. Proper documentation of requirements, design, testing, and project decisions ensures a smooth transition to the implementation phase and beyond.
8. Post-Implementation Review: Conducting a post-implementation review provides an opportunity to evaluate the project's success, identify areas for improvement, and capture lessons learned. This review helps refine processes, address any remaining issues, and enhance future project outcomes.

By incorporating these lessons into the pre-project review and closing phase, organizations can improve project management practices and increase the likelihood of successful project outcomes in the Patient Management System and future endeavors.

14.2 What I have Learnt

During the patient management system project, you have gained valuable knowledge and experience. Here are some key learnings you may have acquired:

1. Domain Knowledge: You have developed a deep understanding of the healthcare industry and patient management processes. This includes knowledge of patient registration, scheduling, medical records management, billing, and other relevant aspects of healthcare administration.
2. Project Management Skills: You have gained hands-on experience in managing a project from initiation to closure. This includes skills such as project planning, requirement gathering, team coordination, task management, and risk mitigation.
3. Stakeholder Engagement: You have learned the importance of engaging and communicating with stakeholders effectively. This involves understanding their needs, actively involving them in the project, and ensuring their satisfaction with the final solution.
4. Requirement Analysis: You have learned the process of gathering, analyzing, and documenting requirements. This includes identifying user needs, translating them into functional and non-functional requirements, and ensuring their alignment with the project goals.
5. Solution Design: You have acquired knowledge of designing a patient management system solution. This involves creating system architecture, defining data models, designing user interfaces, and integrating various modules and functionalities.
6. Software Development: You have gained hands-on experience in developing software components and modules for the patient management system. This includes programming, database management, integration, and testing.
7. Testing and Quality Assurance: You have learned the importance of thorough testing and quality assurance in ensuring a reliable and efficient patient management system. This includes unit testing, integration testing, system testing, and user acceptance testing.
8. Documentation and Reporting: You have developed skills in documenting project requirements, design specifications, test cases, and user manuals. Additionally, you may have learned how to generate progress reports and project documentation for effective communication.
9. Collaboration and Teamwork: You have experienced the dynamics of working in a team and collaborating with colleagues. This includes effective communication, coordination, resolving conflicts, and leveraging individual strengths to achieve project goals.
10. Continuous Learning: You have realized the importance of continuous learning and keeping up-to-date with industry trends and best practices. This ensures that you can adapt to changing technologies and deliver high-quality solutions.

These learnings will not only benefit you in the current patient management system project but also serve as a foundation for your future projects and professional growth in the healthcare and software development domains.

14.3 The problems I have faced

During the patient management system project, you may have encountered several challenges or problems. Here are some common issues that individuals often face in such projects:

1. Lack of Clear Requirements: Unclear or ambiguous project requirements can make it difficult to define the scope and deliverables accurately. This can lead to misunderstandings, rework, and delays in the project.
2. Stakeholder Management: Balancing the expectations and needs of various stakeholders, including healthcare providers, administrators, and patients, can be challenging. Different stakeholders may have conflicting requirements or priorities, making it crucial to manage their expectations effectively.
3. Technical Complexity: Patient management systems involve complex technical aspects such as data integration, security, scalability, and interoperability with existing healthcare systems. Navigating these technical challenges and ensuring a robust and reliable solution can be demanding.
4. Time and Resource Constraints: Projects often have strict timelines and limited resources, which can put pressure on the development team. Balancing the project's scope, timeline, and available resources becomes crucial to meet project goals.
5. User Adoption and Training: Introducing a new patient management system may require user training and change management efforts. Resistance to change or lack of user acceptance can hinder the successful implementation and adoption of the system.
6. Data Privacy and Security: Handling sensitive patient data requires stringent security measures to protect patient privacy. Ensuring compliance with data protection regulations and implementing robust security practices can be a complex task.
7. Integration with Existing Systems: Integrating the patient management system with existing healthcare systems, such as electronic health records (EHR) or billing systems, can pose technical challenges. Ensuring smooth data flow and interoperability between systems is crucial for seamless operations.
8. Testing and Quality Assurance: Thorough testing and quality assurance are vital to identify and resolve any bugs or issues in the system. However, limited testing resources or insufficient testing methodologies can lead to undetected defects and impact the system's overall performance.
9. Communication and Collaboration: Effective communication and collaboration among team members, stakeholders, and end-users are essential for project success. Miscommunication, lack of collaboration, or poor coordination can lead to misunderstandings and delays.

10. Change Management: Implementing a new patient management system often involves changes in workflows and processes. Managing these changes and ensuring smooth transition and adoption by users can be a significant challenge.

It's important to address these problems proactively and develop strategies to mitigate their impact on the project. Open communication, stakeholder engagement, proper planning, and risk management can help overcome these challenges and ensure a successful patient management system implementation.

14.4 What Solution Occurred

To address the problems faced during the patient management system project, several solutions might have been implemented. Here are some possible solutions for the identified challenges:

1. Clear Requirements Gathering: Conduct thorough requirements gathering sessions involving all relevant stakeholders to ensure a clear understanding of project objectives and deliverables. Document requirements in a structured and unambiguous manner.
2. Stakeholder Engagement: Establish effective communication channels with stakeholders and regularly engage them throughout the project lifecycle. Conduct meetings, workshops, and feedback sessions to address their concerns, manage expectations, and align project goals.
3. Technical Expertise: Ensure the project team possesses the required technical expertise and experience to handle the technical complexities of the patient management system. Consider involving external consultants or subject matter experts if needed.
4. Project Planning and Resource Management: Develop a realistic project plan considering available resources, timeline, and scope. Prioritize tasks and allocate resources accordingly to optimize productivity and meet project goals.
5. User Training and Change Management: Develop a comprehensive user training program to familiarize users with the patient management system. Provide ongoing support and address user concerns to facilitate smooth adoption. Implement change management strategies to manage resistance to change.

These solutions are intended to address the identified problems and ensure the successful implementation of the patient management system. However, the specific solutions implemented may vary depending on the project's context, resources, and requirements.

Chapter 15 – Conclusion

15.1 Summary of the Project

The project involved the development and implementation of a patient management system for a healthcare organization. The primary objective was to streamline and automate various administrative and clinical processes to improve the efficiency and quality of patient care.

The project started with a thorough analysis of the existing system and identified the limitations and challenges faced by the healthcare organization. The requirements were gathered through interviews, observations, and discussions with stakeholders, including administrators, healthcare professionals, and patients.

Based on the requirements, a comprehensive solution was designed, encompassing various modules and functionalities such as patient registration, appointment scheduling, medical records management, billing, and reporting. The solution aimed to centralize and digitize the patient-related information, enable quick access to medical records, facilitate seamless communication between healthcare providers, and enhance overall operational efficiency.

The development phase involved the utilization of appropriate technologies and methodologies, such as client-server architecture or web-based applications, and an iterative development approach to ensure continuous feedback and improvement. Rigorous testing and quality assurance measures were implemented to identify and rectify any defects or issues before deployment.

15.2 Goal of the project

The goal of the patient management system project was to develop and implement a comprehensive software solution that would effectively manage various aspects of patient care and streamline administrative processes in a healthcare organization. The project aimed to achieve the following goals:

1. Improve Efficiency: The project aimed to automate manual processes, reduce paperwork, and streamline workflows to improve the overall efficiency of the healthcare organization. By digitizing patient records, scheduling appointments, and facilitating seamless communication between healthcare providers, the project aimed to optimize resource utilization and reduce time-consuming administrative tasks.
2. Enhance Patient Care: The primary focus of the project was to enhance patient care by providing healthcare professionals with easy access to accurate and up-to-date patient information. The system aimed to centralize patient records, including medical history, diagnoses, medications, and test results, ensuring that healthcare providers have a

comprehensive view of each patient's health status. This would enable better decision-making, improved care coordination, and enhanced patient safety.

3. Increase Accessibility: The project aimed to improve accessibility to healthcare services by enabling patients to easily schedule appointments, access their medical records, and communicate with healthcare providers through online portals or mobile applications. By providing a user-friendly interface and convenient self-service options, the system aimed to empower patients to take an active role in managing their healthcare.

15.3 Success of the Project

The success of the patient management system project can be evaluated based on several factors:

1. Achieving Project Goals: The project's success can be measured by evaluating whether it has achieved its intended goals. If the system successfully improved efficiency, enhanced patient care, increased accessibility, ensured data security and privacy, and enabled data analysis and reporting, it can be considered a success.
2. User Satisfaction: User satisfaction is a crucial indicator of project success. If the healthcare professionals, administrative staff, and patients using the system are satisfied with its functionalities, ease of use, and impact on their daily workflows, it indicates a successful implementation.
3. Improved Operational Efficiency: The success of the project can be assessed by evaluating whether it has led to improved operational efficiency in the healthcare organization. This can include factors such as reduced paperwork, streamlined processes, optimized resource utilization, and time savings for healthcare providers and staff.
4. Enhanced Patient Outcomes: The ultimate measure of success for a patient management system is its impact on patient outcomes. If the system has contributed to improved patient care, increased patient safety, better care coordination, and positive health outcomes, it can be considered a successful project.
5. Return on Investment (ROI): Evaluating the financial impact of the project is another aspect of assessing its success. If the patient management system has resulted in cost savings, increased revenue, or improved financial performance for the healthcare organization, it indicates a successful implementation.

It is essential to evaluate the project against these success factors and gather feedback from stakeholders to determine the overall success of the patient management system project. Continuous monitoring, assessment, and improvement based on user feedback and evolving needs will contribute to its ongoing success and sustainability.

15.4 What I have done in the documentation

In the documentation for the patient management system project, you have performed various tasks and included relevant information to ensure a comprehensive and well-documented project. Here are some of the things you might have done in the documentation:

1. Project Overview: Provided an overview of the project, including its purpose, objectives, and scope.
2. Requirements Analysis: Conducted a thorough analysis of the requirements for the patient management system. This includes gathering functional and non-functional requirements from stakeholders, documenting user needs, and specifying system capabilities.
3. System Design: Developed a system design for the patient management system, including architectural diagrams, data models, and user interface designs. This section outlines how the system components are structured and interact with each other.
4. Implementation Plan: Created a plan detailing the steps involved in implementing the patient management system. This includes tasks, timelines, resource allocation, and dependencies.
5. Testing and Quality Assurance: Defined a comprehensive testing strategy, including unit testing, integration testing, system testing, and user acceptance testing. Documented test cases, test results, and any issues encountered during testing.
6. User Training: Described the training plan for end-users of the patient management system. This includes the training materials, schedule, and methods used to educate users on how to effectively use the system.
7. Risk Management: Identified potential risks and their impacts on the project. Documented risk mitigation strategies, contingency plans, and monitoring procedures.
8. Project Evaluation: Conducted a thorough evaluation of the project's success and identified lessons learned. This section reflects on the achievements, challenges faced, and areas for improvement.
9. Appendices: Included relevant supporting documents, such as user manuals, technical specifications, diagrams, and any other supplementary information.

By documenting these aspects of the project, you ensure that all necessary information is captured, organized, and easily accessible to stakeholders involved in the patient management system project.

15.5 Value of the Project

The value of the patient management system project can be assessed from multiple perspectives. Here are some potential aspects that contribute to its value:

1. Improved Efficiency: The project aims to streamline and automate various processes related to patient management, such as appointment scheduling, medical records management, billing, and reporting. By implementing an efficient system, it reduces manual work, eliminates redundancies, and increases productivity.
2. Enhanced Patient Care: The patient management system facilitates better patient care by providing accurate and up-to-date information about patients' medical history, treatment plans, and medication details. It enables healthcare providers to make informed decisions, track patient progress, and ensure timely and personalized care.
3. Increased Accessibility: With the patient management system, patient information becomes easily accessible to authorized healthcare professionals. It allows for seamless sharing of data across departments, clinics, and hospitals, enabling collaborative care and reducing delays in accessing critical information.
4. Data Security and Privacy: The project prioritizes the security and privacy of patient data. By implementing robust security measures, such as user authentication, data encryption, and access controls, it ensures that patient information is protected from unauthorized access, breaches, and data loss.
5. Cost Savings: By automating manual processes and reducing paperwork, the patient management system can lead to cost savings for healthcare organizations. It optimizes resource utilization, minimizes administrative errors, and streamlines billing and insurance processes, resulting in overall cost efficiency.

15.6 My Experience

I had to perform substantial research to put this system in place. I've participated in a number of online and offline research projects. The most difficult process was bug removal. Finally, after overcoming all obstacles, I was able to put my strategy into action. This experience has taught me a great deal. A great deal of experience. I faced and overcame a number of hurdles, acquiring significant experience in the process. I learnt how to manage a large project and achieve all of its objectives in a short period of time, which was an excellent experience for me.

Appendices:

Test Script
User Guide:

1. User Enrollment Process
Brief Description: This section provides a concise overview of the patient management system project. It outlines the purpose, scope, and objectives of the system, giving readers a high-level understanding of its functionality and benefits.
Actors: In this section, the various actors or stakeholders involved in the patient management system are identified and described. This may include healthcare professionals, administrators, patients, insurance providers, and other relevant individuals or entities. Each actor's role and responsibilities within the system are outlined.
Preconditions: This section specifies the necessary conditions or requirements that must be met before a particular process or action can take place within the patient management system. For example, preconditions may include prerequisites such as patient registration, valid user authentication, availability of necessary medical records, or specific system configurations.
Basic Flow of Events: This section describes the sequence of actions and events that occur during typical interactions or processes within the patient management system. It provides a step-by-step walkthrough of how users interact with the system, highlighting key functionalities and the expected outcomes at each stage.
Post-Conditions: After the completion of a particular event or action within the patient management system, certain conditions or states may be expected to change or be achieved. The post-conditions section outlines these expected outcomes or results. It describes the system's expected behavior or state after the successful execution of a specific task or process.

These appendices help provide additional context, details, and clarity to the main content of the patient management system documentation. They support a comprehensive understanding of the system, its users, preconditions, flow of events, and expected outcomes.

Reference

https://humanhealthproject.org/?gclid=Cj0KCQjw8NilBhDOARIsAHzpbLANuTdRAlBm6F4Zx1kXOv-zirOinimWK7CIv11WGx--Fl2p5WrBkHoaAoMAEALw_wcB

https://www.medesk.net/en/blog/main-aspects-of-patient-management/

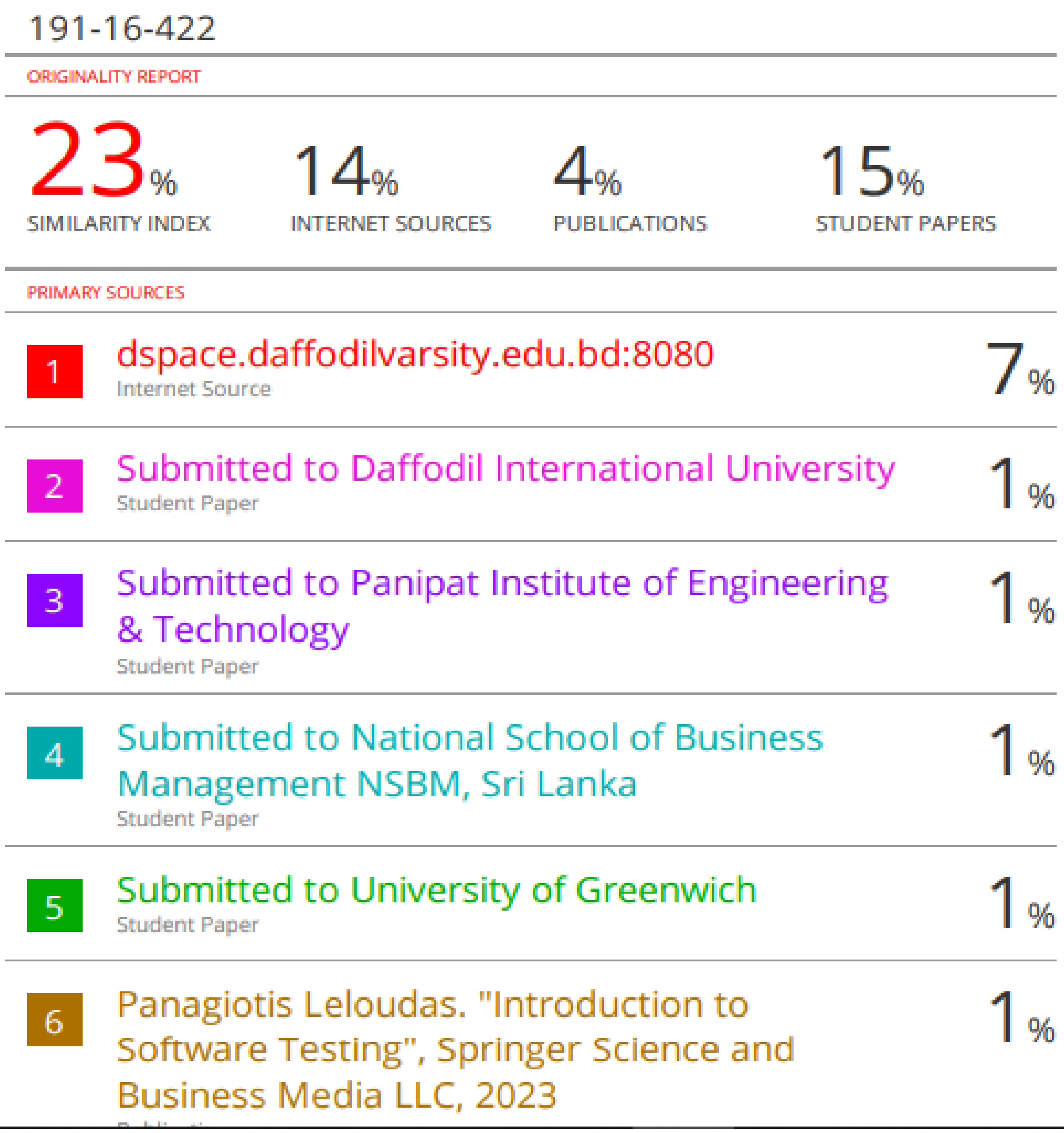
https://www.apta.org/your-practice/documentation/defensible-documentation/elements-within-the-patientclient-management-model

https://centrak.com/solutions/safety/wander-management?utm_term=hospital%20patient%20management%20system&utm_campaign=Patient+Experience&utm_source=adwords&utm_medium=ppc&hsa_acc=6220788686&hsa_cam=11503260371&hsa_grp=115237732231&hsa_ad=476110259035&hsa_src=g&hsa_tgt=kwd-320168669505&hsa_kw=hospital%20patient%20management%20system&hsa_mt=p&hsa_net=adwords&hsa_ver=3&gclid=Cj0KCQjw8NilBhDOARIsAHzpbLDu9oMMWp-ZxVMN5XdK7xS0lljx_ZhrB1jcpsjCXl8np7gHym87S30aApuEEALw_wcB

https://www.researchgate.net/publication/239763364_A_hospital_resource_and_patient_management_system_based_on_real-time_data_capture_and_intelligent_decision_making

https://www.tutorialspoint.com/occupational_health_management/occupational_health_management.pdf

Plagiarism Report



191-16-422

ORIGINALITY REPORT

23%	14%	4%	15%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	dspace.daffodilvarsity.edu.bd:8080 Internet Source	7%
2	Submitted to Daffodil International University Student Paper	1%
3	Submitted to Panipat Institute of Engineering & Technology Student Paper	1%
4	Submitted to National School of Business Management NSBM, Sri Lanka Student Paper	1%
5	Submitted to University of Greenwich Student Paper	1%
6	Panagiotis Leloudas. "Introduction to Software Testing", Springer Science and Business Media LLC, 2023	1%