Medical Center System

Final Project Report

Sri Lanka Institute of Information Technology

IT2080 Information Technology Project

Group ITP_WE_B1_204

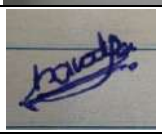
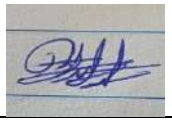
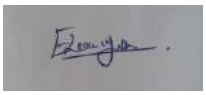
October 2025

Appendix 2 – Declaration

Declaration

This project report is our original work, and the content is not plagiarized from any other resource. References for all the content taken from external resources are correctly cited. To the best of our knowledge, this report does not contain any material published or written by third parties, except as acknowledged in the text.

Authors:

Author SID	Author name	Signature
IT23588714	K.H.S.Dinsara	
IT23543232	Navodya A.K.	
IT23572638	Thathsarani J. R.	
IT23669758	Liyanaarachchi L.A.D. A	
IT23810464	Ekanayaka E.W.I. D	

Date: …17/10/2025…

Abstract

The MediCura Medical Center System is designed to make running a medical facility smoother and more efficient. It brings together all the essential functions in one place, including managing users, keeping track of inventory, handling suppliers, overseeing laboratory operations, and managing billing and invoices.

With **user management**, the system makes it easy to register and organize patients, staff, and administrators, while ensuring secure access and proper permissions. **Inventory management** keeps a real-time check on medical supplies and equipment, helping prevent shortages and delays. Through **supplier management**, the system keeps track of orders, deliveries, and vendor details, simplifying the supply chain process. **Laboratory management** organizes patient tests and results, ensuring timely and accurate reporting. Lastly, **billing and invoice management** automates financial tasks, making payments and record-keeping more accurate and less time-consuming.

Overall, MediCura acts as a reliable partner for medical centers, helping staff work more efficiently, patients receive better care, and the entire facility runs more smoothly.

Acknowledgement

I would like to sincerely thank everyone who supported me during the development of the MediCura Medical Center System. First, I am truly grateful to my lecturers and advisors at SLIIT for their guidance, encouragement, and valuable advice throughout this project. Their feedback helped me stay on track and improve the system at every step.

I also want to thank my family and friends for their patience, understanding, and constant motivation. Their support gave me the energy and focus I needed to complete this project successfully.

Special thanks go to my peers and colleagues who shared their insights, helped me solve technical challenges, and offered ideas to make the system more practical and efficient. Your contributions made a real difference.

Finally, I appreciate all the resources, tools, and references I used along the way-they were essential in turning this project from an idea into a functioning system.

Contents

Declaration 2

Introduction 5

- Background 5
- Problem and motivation 5
- Aim and objectives 7
- Solution overview 8
- Methodology 8

Requirements 9

- Stakeholder Analysis 9
- Requirements Analysis 9
- Requirements Modeling 10

Design and Development 19

- System Architecture 20
- Component Design 22
- Process & Workflow Design 22
- Normalized Schema 29
- ER Diagram 30
- Development Process 30
- High-level system design diagram 31
- Acceptance Criteria 33
- Main Test Cases and Results 34

Evaluation & Conclusion 40

- Evaluation 40
- Conclusion 41

References 42

- Referencing Sources 42
- Referencing tools 42

Individual Contribution 43

Introduction

Background

Healthcare is one of the most vital sectors in society, requiring efficiency, accuracy and effective coordination among staff to ensure high quality patient care. Medical centers, especially midsized private clinics, often provide a range of services including general conclusions, laboratory tests, pharmacy services, and administrative functions. However, many centers in Sri Lanka still rely on paper-based records or outdated standalone software systems. These methods are prone to errors, delays, and inefficiencies that directly affect patient satisfaction and treatment outcomes. The proposed client is a private medical center that provides outpatient consultations, laboratory testing, pharmacy services and basic health screening packages. The center operates with a team of doctors, nurses, lab assistants, pharmacists, and administrative staff, serving an average of 150 to 200 patients daily. The management has identified the need for a centralized, web-based solution to streamline all operations from patient registration to billing to improve service quality, reduce administrative burdens and enhance patient engagement.

Problem and motivation

Problem Statement

Currently, the medical center manages its patient appointments, medical records, billing, and inventory through manual processes or isolated applications that are not interconnected. This results in:

- Scheduling conflicts due to lack of centralized appointment management
- Incomplete medical histories because records are scattered across multiple files or departments Delayed updates on test results and prescriptions
- Inefficient inventory tracking, leading to stock outs or overstocking of
- medicines and medical supplies
- Poor patient communication, as updates are not provided in real time

Motivation

- By introducing a centralized medical center management System (MediCura), the center can Improve patient satisfaction through faster service, real-time updates and online access to records
- Reduce human errors in scheduling, recordkeeping, and billing
- Enable doctors and staff to make better decisions with instant access to patient data
- Optimize inventory management and reduce waste
- Provide management with insights through detailed reports for better resource allocation and strategic planning

Literature review

In developing the MediCura Medical Center Management System, it is important to review existing studies, technologies, and best practices related to healthcare management systems. This section examines relevant research on healthcare information systems, appointment and records management, inventory and billing systems, and user interface considerations.

Healthcare Information Systems (HIS)

Healthcare Information Systems are designed to improve the quality and efficiency of healthcare delivery by centralizing patient and operational data. According to Bassi and Lau (2013), HIS can reduce medical errors, improve patient outcomes, and streamline administrative processes. However, many clinics in Sri Lanka still rely on paper-based or standalone systems, which hinder communication and cause delays in service delivery. The MediCura system aims to address these gaps by offering an integrated, web-based solution

Appointment and Medical Records Management

Efficient appointment scheduling reduces patient wait times and prevents scheduling conflicts. Research by Al-Mutairi et al. (2020) highlights that centralized appointment systems improve patient satisfaction and clinic efficiency. Similarly, Electronic Medical Records (EMR) systems allow healthcare professionals to store and retrieve patient histories, prescriptions, and lab results instantly, leading to better clinical decision-making (Rahman & Abdullah, 2021). Successful implementations, such as Epic and Cerner, show the importance of secure, accessible, and real-time record management.

Inventory and Billing Systems in Healthcare

Effective inventory management ensures the availability of medicines and medical supplies while minimizing wastage. Kumar and Singh (2021) emphasize that automated inventory systems with reorder alerts significantly improve supply chain performance. Integration of billing systems with inventory modules ensures accurate financial tracking and transparency (Patel et al., 2020). The MediCura system will combine inventory management with billing, enabling smooth pharmacy transactions, consultation fee processing, and report generation.

User Interface and User Experience (UI/UX) in Healthcare Applications

The usability of healthcare systems is critical for adoption by medical staff and patients. Nielsen (1993) highlights that simplicity, clear navigation, and consistency are key to effective UI/UX design. In the healthcare sector, where speed and accuracy are vital, responsive design and accessibility features (such as WAI-ARIA standards) ensure that systems are inclusive and easy to use across multiple devices. Systems like My Chart and Health Hub demonstrate how intuitive interfaces can improve engagement and overall satisfaction.

Aim and objectives

Aim

To design and develop a centralized, web-based Medical Center Management System (MediCura) that integrates appointment scheduling, electronic medical records, billing, inventory management, and reporting to improve operational efficiency, reduce errors and enhance patient experience.

Objectives

1. Develop an appointment scheduling module that allows patients and receptionists to manage bookings, cancellations and reminders in real time
2. Implement an electronic medical records system for doctors, nurses and lab assistants to access and update patient histories, prescriptions and test results.
3. Integrate billing and payment processing to handle consultation fees, lab charges and pharmacy transactions securely.
4. Create an inventory management module for tracking medicines, lab kits and medical supplies with low stock alerts
5. Enable role-based access control to ensure secure and authorized access to system features.
6. Design a reporting dashboard for management to monitor patient flow, staff performance and financial summaries.
7. Provide a notification system to keep patients informed of appointments, test results and prescriptions
8. Ensure system scalability and data security through the use of secure web technologies and encrypted data storage

Solution overview

The MediCura Medical Center System is designed to make the day-to-day operations of a medical facility smoother, faster, and more organized. Instead of juggling separate processes manually, this system brings everything together in one place, making it easier for staff to focus on providing quality care.

At its core, the system handles **user management**, allowing staff, administrators, and patients to be registered, organized, and given the right access to the system. **Inventory management** keeps track of medical supplies and equipment in real-time, helping prevent shortages and ensuring the hospital always has what it needs. Through **supplier management**, ordering and tracking deliveries become simple, reducing delays and errors in the supply chain. **Laboratory management** ensures that patient tests and results are handled efficiently, so doctors and patients can get accurate information without unnecessary waiting times. Finally, **billing and invoice management** automates financial tasks, making invoices, payments, and record-keeping faster, more accurately, and less stressful.

Overall, MediCura isn't just a system, it's a supportive tool that helps medical staff work smarter, reduces manual effort, and ensures that patients receive timely and better care.

Methodology

Agile Framework Implementation

We implemented the Scrum methodology for our Medical Center Management System. What this means is we build the system in small sections named sprints, which take 2–3 weeks each. Each sprint produces working features that doctors, laboratory assistants, receptionists, and patients can view and test. We sit down regularly with medical personnel, administrators, and other critical users and get feedback from them. That way, we develop what they truly need, rather than what we think they need.

At the conclusion of every sprint, we discuss what worked and where we can improve.

Agile Benefits:

- Quick testing with real medical center users.
- Flexible changes as healthcare needs evolve.
- Continuous testing throughout development.
- Frequent delivery of working system features.
- Improved team communication.
- Early identification of issues and bugs.

Requirements

Stakeholder Analysis

The system was designed to meet the needs of several key stakeholders who interact with the medical center's daily operations. Each group has its own goals and expectations, which helped shape the system requirements.

Stakeholder	Role / Responsibility	Main Expectations from the System
Administrator	Manages the entire system and users	Needs full control of all modules, reports, and user access levels
Doctor	Review patient details, lab results, and reports	Easy access to patient records and accurate lab reports
Lab Assistant	Manage lab tests, appointments, and results	Simple lab form management, scheduling, and automated report generation
Manager	Handle payments, invoices, and expenses	Smooth billing, online payments, and clear financial summaries
Supplier Manager	Provide medical items and handle orders	Receive and confirm supply requests, create invoices, add/update/delete clients
Inventory Manager	Monitors stock and supply levels	Get alerts for low stock, manage items, and generate reports
Patient / user	Access services, payments, and results	User-friendly interface to register, book tests, and get reports quickly

Requirements Analysis

During this phase, both **functional** and **non-functional** requirements were identified through interviews, workflow observations, and existing system reviews.

Functional Requirements

- User registration, login, and role-based access control.
- Laboratory management (test requests, report uploads, and AI analysis).
- Billing and payment processing (manual and online via PayHere).
- Supplier and inventory management (stock updates, requests, and reports).
- Notification and email alerts for lab confirmations and approvals.
- Report generation and export (PDF and CSV formats).

Non-Functional Requirements

- **Performance:** Pages and dashboards should be loaded within 2 seconds.
- **Security:** Role-based access, encrypted passwords, and secure transactions.
- **Usability:** Simple and responsive interface accessible on desktop and mobile.
- **Reliability:** System should handle multiple users without data loss.
- **Maintainability:** Code should allow easy updates and improvements.
- **Scalability:** Support for adding more users and modules in the future.

Requirements Modeling

The modeling phase visualized how different components of the system interact. Using UML-style diagrams and logical data flow representations, the following models were developed:

Use Case Model

It shows how different users (Admin, Lab Staff, Supplier, etc.) interact with the system.

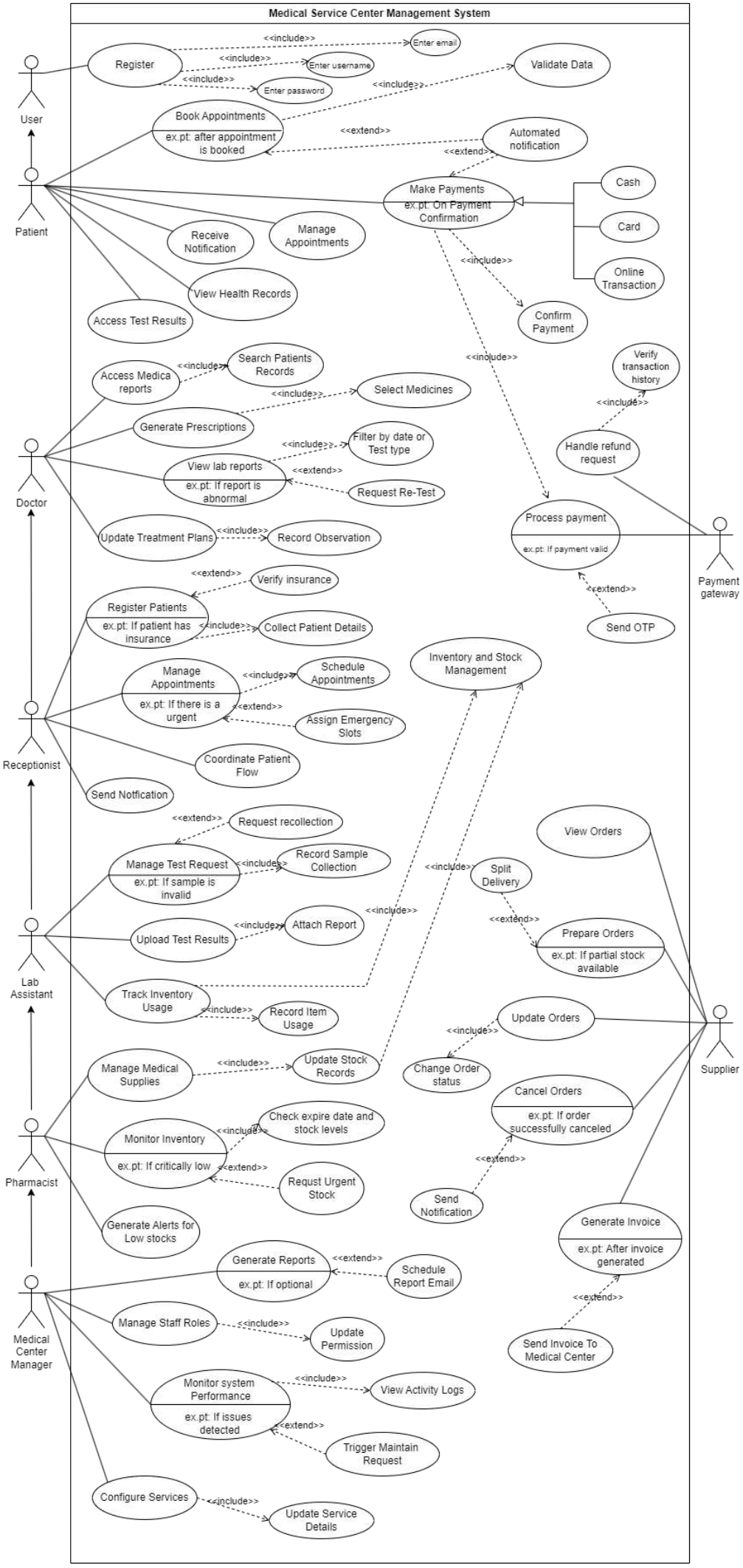
- **Actors:** Admin, Lab Assistant, Supplier, Patient, Inventory Manager.
- **Main Use Cases:**
    - Manage users and roles
    - Handle lab requests and results
    - Process billing and payments
    - Manage inventory and supplier request
    - Generate reports and notifications

Data Flow Model (DFD Level 0–1)

- **Inputs:** User data, lab test data, supplier requests, billing info.
- **Processes:** Data validation, database operations, report generation.
- **Outputs:** Updated dashboards, reports, and confirmation messages.

Entity Relationship (ER) Model

- **Entities:** User, Patient, Lab Report, Invoice, Supplier, Inventory.
- **Relationships:**
    - One **Patient** can have many **Labs Reports**
    - One **Supplier** can handle many **Inventories Requests**
    - One **Admin** manages multiple **Users**


Medical Service Center Management System
User
Register
<<include>>
Enter email
<<include>>
Enter username
<<include>>
Enter password
Validate Data
<<include>>
Book Appointments
ex.pt: after appointment is booked
<<extend>>
Automated notification
<<extend>>
Patient
Make Payments
ex.pt: On Payment Confirmation
Cash
Card
Online Transaction
<<include>>
Confirm Payment
Receive Notification
Manage Appointments
View Health Records
Access Test Results
<<include>>
Verify transaction history
<<include>>
Handle refund request
Doctor
Access Medica reports
<<include>>
Search Patients Records
<<include>>
Select Medicines
Generate Prescriptions
<<include>>
Filter by date or Test type
View lab reports
ex.pt: If report is abnormal
<<extend>>
Request Re-Test
Update Treatment Plans
<<include>>
Record Observation
Process payment
ex.pt: If payment valid
Payment gateway
<<extend>>
Send OTP
<<extend>>
Verify insurance
Register Patients
ex.pt: If patient has insurance
<<include>>
Collect Patient Details
Inventory and Stock Management
Receptionist
Manage Appointments
ex.pt: If there is a urgent
<<include>>
Schedule Appointments
<<extend>>
Assign Emergency Slots
Coordinate Patient Flow
Send Notfication
<<extend>>
Request recollection
View Orders
Manage Test Request
ex.pt: If sample is invalid
<<include>>
Record Sample Collection
<<include>>
Split Delivery
<<extend>>
Prepare Orders
ex.pt: If partial stock available
<<include>>
Attach Report
Upload Test Results
<<include>>
Lab Assistant
Track Inventory Usage
<<include>>
Record Item Usage
<<include>>
Update Orders
<<include>>
Change Order status
Supplier
Manage Medical Supplies
<<include>>
Update Stock Records
Cancel Orders
ex.pt: If order successfully canceled
Check expire date and stock levels
<<include>>
Monitor Inventory
ex.pt: If critically low
<<extend>>
Requst Urgent Stock
<<extend>>
Send Notification
Pharmacist
Generate Alerts for Low stocks
Generate Invoice
ex.pt: After invoice generated
<<extend>>
Generate Reports
ex.pt: If optional
<<extend>>
Schedule Report Email
<<extend>>
Send Invoice To Medical Center
Manage Staff Roles
<<include>>
Update Permission
Medical Center Manager
Monitor system Performance
ex.pt: If issues detected
<<include>>
View Activity Logs
<<extend>>
Trigger Maintain Request
Configure Services
<<include>>
Update Service Details

User registration & login

Use case Name	User registration & Login
Actor	Patient, Doctor, Pharmacist, Lab Assistant, Receptionist, Supplier, Medical Center Manager
Goal	To allow authorized users to register and log in securely to access the medical center system.
Overview	• New users can register by providing the required details. • Existing users can log in using valid credentials. • The system verifies identities, provides role-based access, and ensures data security.
Pre-conditions	• User has a valid email/contact information and role in the medical center. • The system is running and connected to the user database.
Post-conditions	• User is successfully registered or logged in. • User data is securely stored. • Invalid attempts are recorded for security.
Basic path	1) User opens the registration or login page. 2) For registration, user enters the required information and submits. 3) System validates details and stores them in the database. 4) For login, user enters the username/password. 5) System verifies credentials and grants access based on role.
Alternative path	• Invalid Data: Registration fails if mandatory fields are missing or incorrect. • Duplicate Registration: System prevents using an already registered email/contact. • Invalid Login: System denies access on incorrect username/password and prompts retry.
	• Account Lock: After multiple failed login attempts, the account may be locked or flagged.

NFRs & TRs	NFRs (Non-Functional Requirements) • Performance: Registration and login must complete within 2–3 seconds. • Usability: Interface should be simple and user-friendly. • Security: Strong authentication, encrypted passwords, and audit logs. • Reliability: System must be available 99.9% of the time for access. • Scalability: Support many users logging in at the same time without slowdown. TRs (Technical Requirements) • Secure password hashing • Role-based access control for different user types. • Database for storing user credentials securely. • SSL/TLS for secure data transmission. • CAPTCHA or OTP for additional verification if needed.

Check the patient's test request and identify required lab tests

Use case Name	Check the patient's test request and identify required lab tests
Actor	Lab Assistant
Goal	To verify a patient's test request and determine which lab tests need to be performed.
Overview	• The lab assistant checks for requests for tests from doctors. • The system helps decide what laboratory tests are required, eliminating incorrect and duplicate tests, and giving accurate results.
Pre-conditions	• The patient test request is made by an authorized doctor. • The system is functional and connected to the database of patient records.
Post-conditions	• The required lab tests are confirmed and marked for processing. • Any discrepancies (missing or unclear tests) are flagged for review.
Basic path	• Lab assistant logs in to the system. • Opens pending test requests. • Reviews details of the patient request. • Confirms required lab tests from the list. • Marks tests as approved for processing.
Alternative path	• Incomplete Request: If information is missing, the system alerts the lab assistant to contact the doctor. • Duplicate Request: The system flags repeated or overlapping test orders. • Invalid Patient ID: The request is rejected if patient details are incorrect.

NFRs & TRs	NFRs (Non-Functional Requirements) • Performance: Requests and lab test data must load within 2 seconds. • Accuracy: The system must prevent errors in identifying required tests. • Usability: The Interface should clearly display test requests and details. • Security: Patient data must be encrypted and accessible only to authorized staff. • Reliability: The system should be available 24/7 to handle urgent test requests. TRs (Technical Requirements) • Secure access with role-based authentication. • Integration with electronic medical records to retrieve patient data. • Automatic alerts for missing or duplicating information. • Audit logs for all actions taken by lab assistants.

Component operations for medicines

Use case Name	**Component operations for medicines**
Actor	Pharmacist
Goal	To efficiently manage medicine records — adding, updating, deleting, viewing, and tracking medicines in the inventory.
Overview	This function maintains accurate and up-to-date medicine information, ensuring correct stock levels, timely alerts for expiry or low stock, and smooth inventory management.
Pre-conditions	• The pharmacist is authenticated and authorized to access the medicine inventory. • The medicine database is connected and operational.
Post-conditions	• Medicine records are successfully updated (added, modified, deleted, or viewed). • Inventory levels are accurate and reflect real-time status. • Alerts for low stock or expiry are generated automatically if applicable.

Basic path	1. Pharmacist logs in to the system. 2. Opens the medicine inventory module. 3. Selects an action: Add, Update, Delete, or View medicines. 4. System processes the request and updates inventory records. 5. Confirmation is displayed to the pharmacist.
Alternative path	• Invalid Data: If incomplete or incorrect medicine details are entered, the system prompts correction. • Duplicate Records: The System prevents adding a medicine batch already in stock. • Unauthorized Access: If a non-pharmacist attempts to access, the system denies entry. • Database Error: If the system is offline, the operation is queued or rejected with an alert.
NFRs & TRs	NFRs (Non-Functional Requirements) • Performance: All inventory operations must complete within 2–3 seconds. • Usability: The interface should be clear and user-friendly for quick medicine management. • Security: Only authorized staff should modify inventory; data must be encrypted.
	• Reliability: The System should be available 99.9% of the time to avoid medicine stock issues. • Accuracy: Real-time stock levels must always match physical inventory. • Scalability: Capable of handling large medicine databases without slowing down. TRs (Technical Requirements) • Secure database connection with role-based authentication. • Automatic validation to prevent duplicate or incorrect records. • Real-time inventory tracking with automated low-stock and expiry alerts. • Audit logs for all inventory operations (who changed what and when). • Backup and recovery to protect against data loss.

Transaction recording and tracking

Use case Name	Transaction recording and tracking
Actor	Pharmacist (primary) Supplier (secondary, for order/payment updates) Medical Center Manager
Goal	To accurately record all financial and inventory transactions and enable real-time tracking for auditing, reporting, and decision-making.
Overview	• Every transaction, including medicine orders, deliveries, payments, and stock updates, is automatically recorded in the system. • Users can view transaction history, track ongoing processes, and generate reports. • The system provides transparency, traceability, and reliability in the medical center's operations.
Pre-conditions	• Users must be authenticated and authorized. • Inventory and supplier databases must be connected and up to date. • The system must be operational.
Post-conditions	• Transaction is recorded and stored in the system permanently. • Real-time updates reflect in inventory and financial records. • Auditable history is available for review.
Basic path	• User performs a transaction (e.g., placing an order, receiving a delivery, processing payment). • System automatically records transaction details: ID, date/time, items, quantities, amounts, user, and status. • Inventory and financial records are updated in real-time. • Notifications or confirmations are sent to relevant users (pharmacist, manager, supplier). • Transaction is stored for future auditing and reporting.
Alternative path	• Invalid Transaction: If data is missing or incorrect, system prompts user to correct it. • System Failure: If system is offline, transaction is queued and processed once available. • Unauthorized Action: If an unauthorized user attempts a transaction, the system denies it and logs the attempt.
	• Partial Delivery or Payment: System updates transaction with partial status and pending items/amounts.

NFRs & TRs	NFRs (Non-Functional Requirements) • Accuracy: All transactions must be correctly recorded with no duplication or loss. • Security: Role-based access, encrypted data storage, and secure transmission. • Performance: Real-time recording and retrieval of transactions. • Reliability: The System must be highly available (99.9%) to ensure uninterrupted operation. • Auditability: Every action is logged with timestamps and user IDs. TRs (Technical Requirements) • Secure database for storing all transactions. • Integration with inventory and supplier modules. • Notification system for transaction updates. • Logging mechanism for audit trails. • Reporting tools for exporting transaction history. • Backup and recovery to protect against data loss.

Automatic purchase order generation

Use case Name	Automatic purchase order generation
Actor	System (automatic process) Pharmacist / Medical Center Manager (for monitoring/confirmation)
Goal	To automatically generate purchase orders when stock levels fall below minimum thresholds, ensuring timely replenishment.
Overview	• The system monitors inventory continuously. • When stock for a medicine or equipment drops below the predefined threshold, a purchase order is created automatically and sent to the supplier.
Pre-conditions	• Minimum stock levels must be configured. • Supplier information must be available in the system. • Inventory tracking is active and up-to-date.
Post-conditions	• Purchase order is created and sent to the supplier.
	• Notifications sent to relevant staff (pharmacist/manager). • Inventory replenishment process is initiated.

Basic path	1. System monitors stock levels. 2. Stock falls below the reorder point. 3. The system generates a purchase order with the required items and quantities. 4. The order is sent to the supplier. 5. The pharmacist/manager receives notification and can review if needed.
Alternative path	• Supplier details missing → system alerts pharmacist to manually assign supplier. • Auto-ordering disabled → pharmacist is notified to create a manual order. • Multiple low-stock items → system consolidates into a single order per supplier.
NFRs & TRs	NFRs (Non-Functional Requirements) • Performance: Orders generated immediately upon low stock detection. • Accuracy: Correct items, quantities, and supplier information • Reliability: Automatic ordering works consistently. • Security: Only authorized staff can view or modify autoorder settings. • Auditability: All auto-generated orders are logged. TRs (Technical Requirements) • Integration with the inventory module for real-time stock monitoring. • Supplier database with contact and ordering information. • Notification system for alerts to staff. • Logging mechanism for auditing orders. • Option to enable/disable auto-order per item.

Design and Development

The **Medical Center Management System** was designed to provide a complete digital solution for handling patients, lab operations, suppliers, payments, and inventory all within a secure, centralized web platform.
The design focused on clarity, modularity, and smooth interaction between all departments.

System Architecture

The overall system follows a **three-tier architecture**, dividing the project into three layers:

1. **Presentation Layer (Frontend)** – Built using **React.js** for an interactive and user-friendly interface. It allows different users (Admin, Lab Staff, Supplier, etc.) to access role-specific dashboards.
2. **Application Layer (Backend)** – Developed using **Node.js and Express.js**, handling all business logic, form validation, and communication between the frontend and database.
3. **Database Layer** – Managed through **MongoDB**, which stores all user details, patient reports, billing data, and supplier records in secure collections.

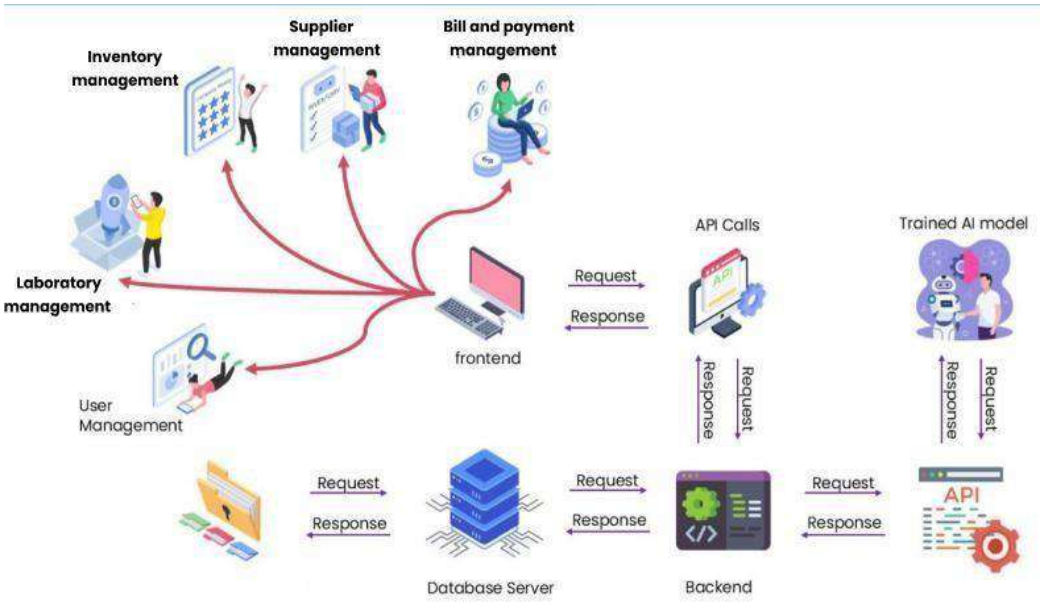


Figure 3.1

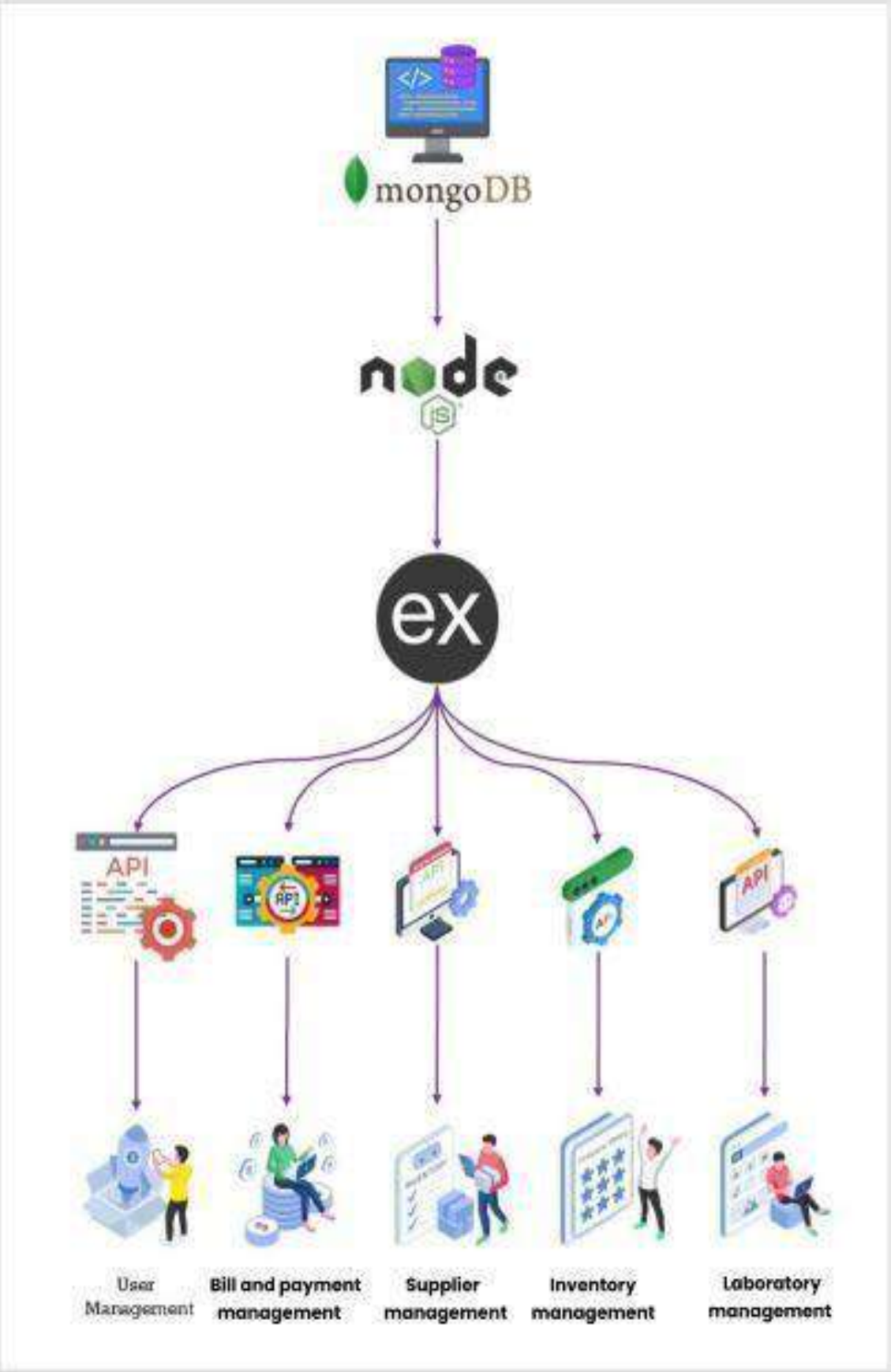

mongoDB
node
ex
API
User Management
Bill and payment management
Supplier management
Inventory management
Laboratory management

Figure 3.2

Component Design

Each major module was built as a separate component to keep the system easy to maintain and expand later.

Component	Purpose
User Management	Handles login, registration, and role-based dashboards.
Laboratory Management	Manages lab requests, report uploads, and AIbased analysis.
Billing & Payment Management	Processes invoices, expenses, and online transactions.
Supplier Management	Tracks supplier requests, invoices, and approvals.
Inventory Management	Monitors stock levels, alerts, and restock requests.
Notification System	Send alerts and confirmation emails automatically.

Table 3.1

Process & Workflow Design

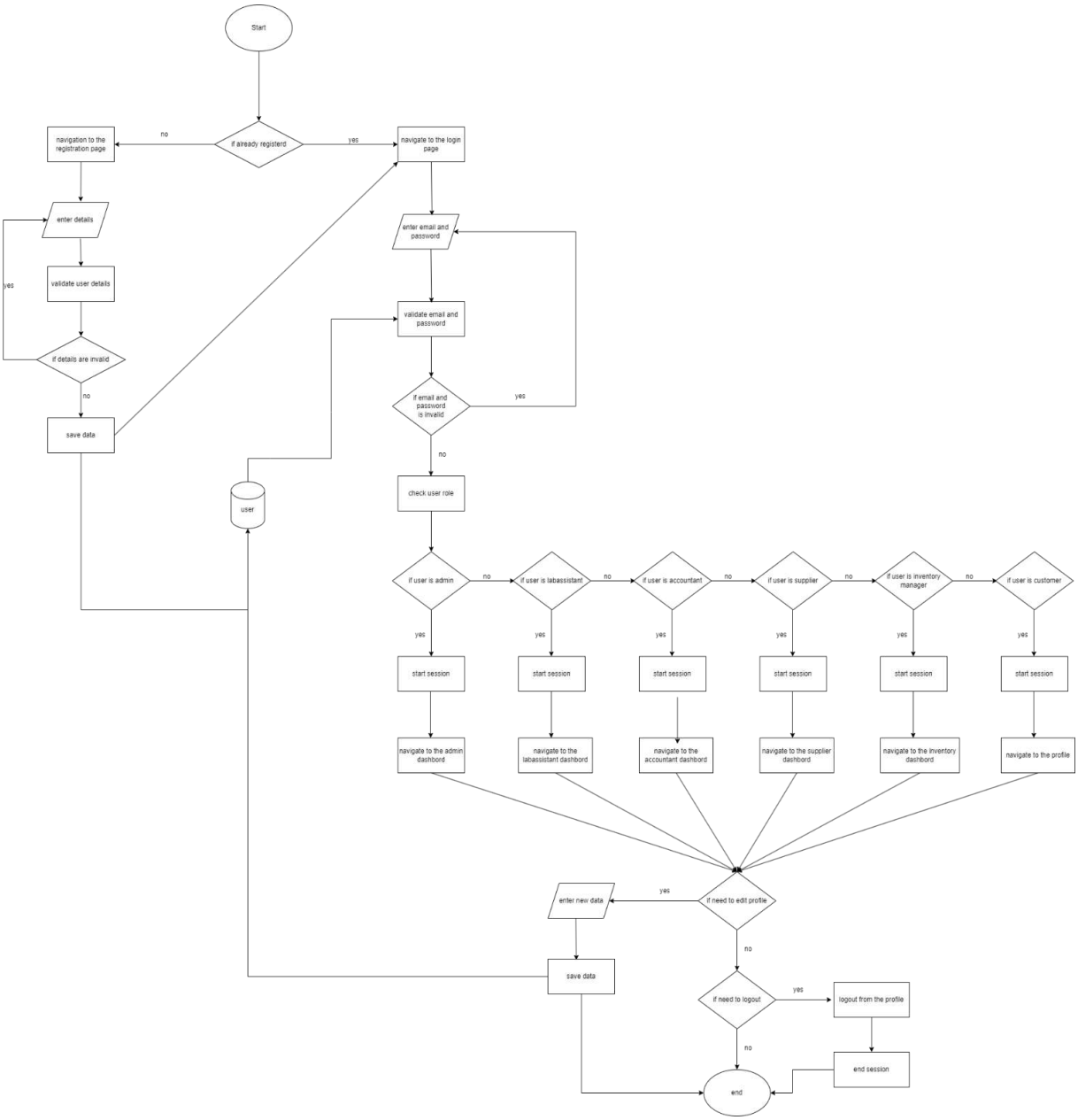
The workflows were designed to ensure smooth coordination between all departments-from patient registration to billing, lab management, supplier coordination, and inventory tracking. Each module follows a logical, automated flow to minimize manual effort and reduce human error.

User Management Workflow

1. **User Registration:**
   New users (patients, staff, or admins) register by submitting required details.
2. **Authentication:**
   Credentials are verified through the Node.js backend using secure password hashing.
3. **Role Assignment:**
   Based on role (Admin, Lab Assistant, Supplier, Inventory Manager, or Manager), the system assigns relevant access rights.
4. **Login and Dashboard Access:**
   Each user logs in and is directed to a customized dashboard.
5. **Profile Management:**
   Users can update or delete their profile details anytime.
6. **Session and Logout:**
   Automatic session timeout and secure logout protect user data.

Result:

Users access only authorized modules and can perform their assigned tasks efficiently.

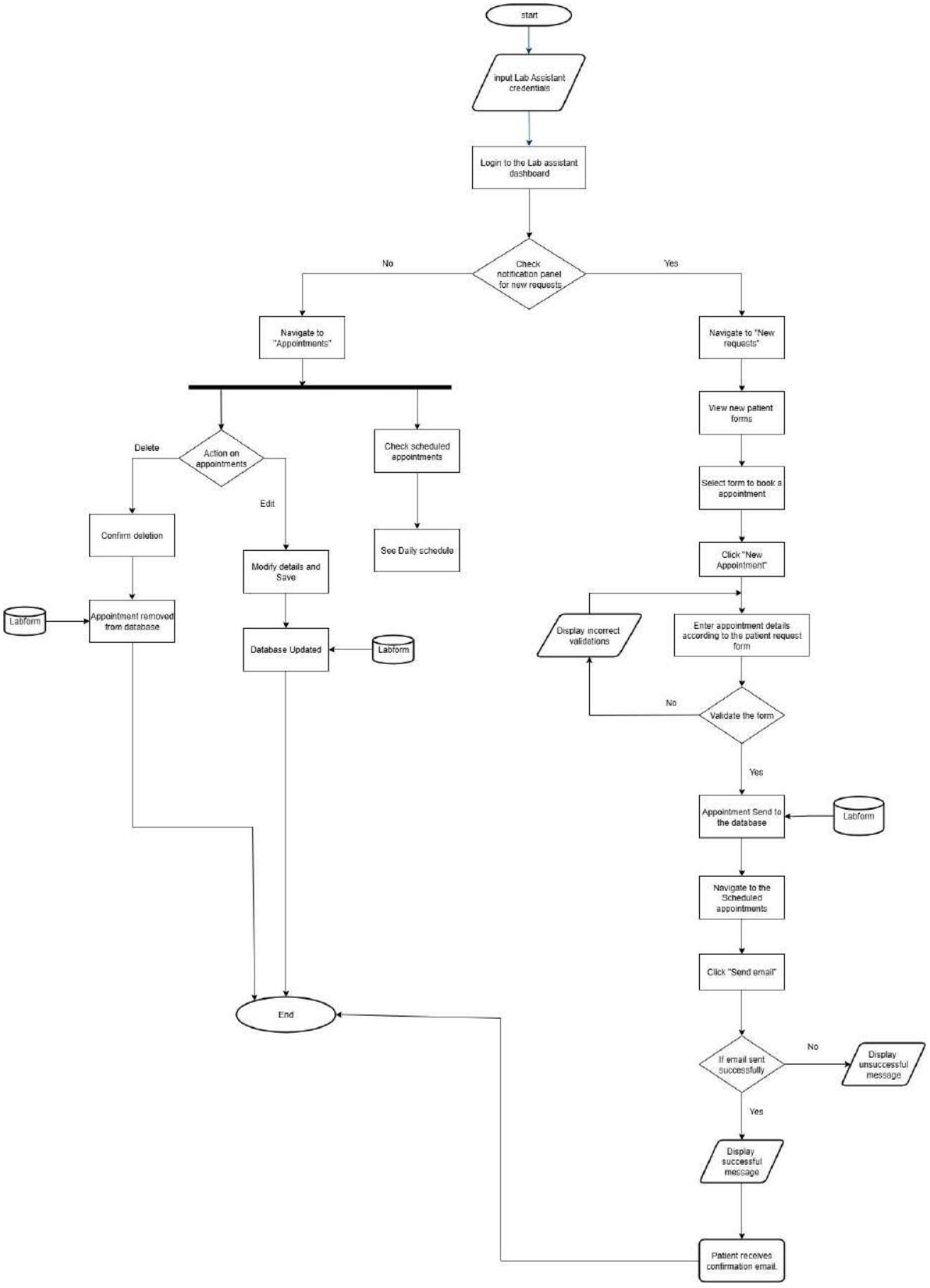

Start
if already registerd
no
yes
navigation to the registration page
navigate to the login page
enter details
validate user details
if details are invalid
yes
no
save data
enter email and password
validate email and password
if email and password is invalid
yes
no
check user role
user
if user is admin
no
if user is labassistant
no
if user is accountant
no
if user is supplier
no
if user is inventory manager
no
if user is customer
yes
start session
navigate to the admin dashbord
navigate to the labassistant dashbord
navigate to the accountant dashbord
navigate to the supplier dashbord
navigate to the inventory dashbord
navigate to the profile
if need to edit profile
yes
enter new data
save data
no
if need to logout
yes
logout from the profile
no
end session
end

Laboratory Management Workflow

1. **Lab Request Submission:**
   Patients fill out a lab request form by entering their personal information, preferred test type, and appointment date and time.
2. **Request Review and Approval:**
   Lab staff review the submitted requests and verify that all details are accurate before approving them for scheduling.
3. **Appointment Scheduling:**
   Once approved, lab appointments are automatically scheduled based on staff and equipment availability.
4. **Lab Report Upload and AI Analysis:**
   After the test is completed, the lab staff uploads the lab report. The integrated AI bot analyzes the report and provides summarized results for easier understanding.
5. **Patient Notification:**
   The system automatically sends an email or notification to the patient containing the test results and any important updates.
6. **Real-Time Dashboard Update:**
   The lab assistant's dashboard updates instantly, showing new requests, appointments, and completed reports.

Result:

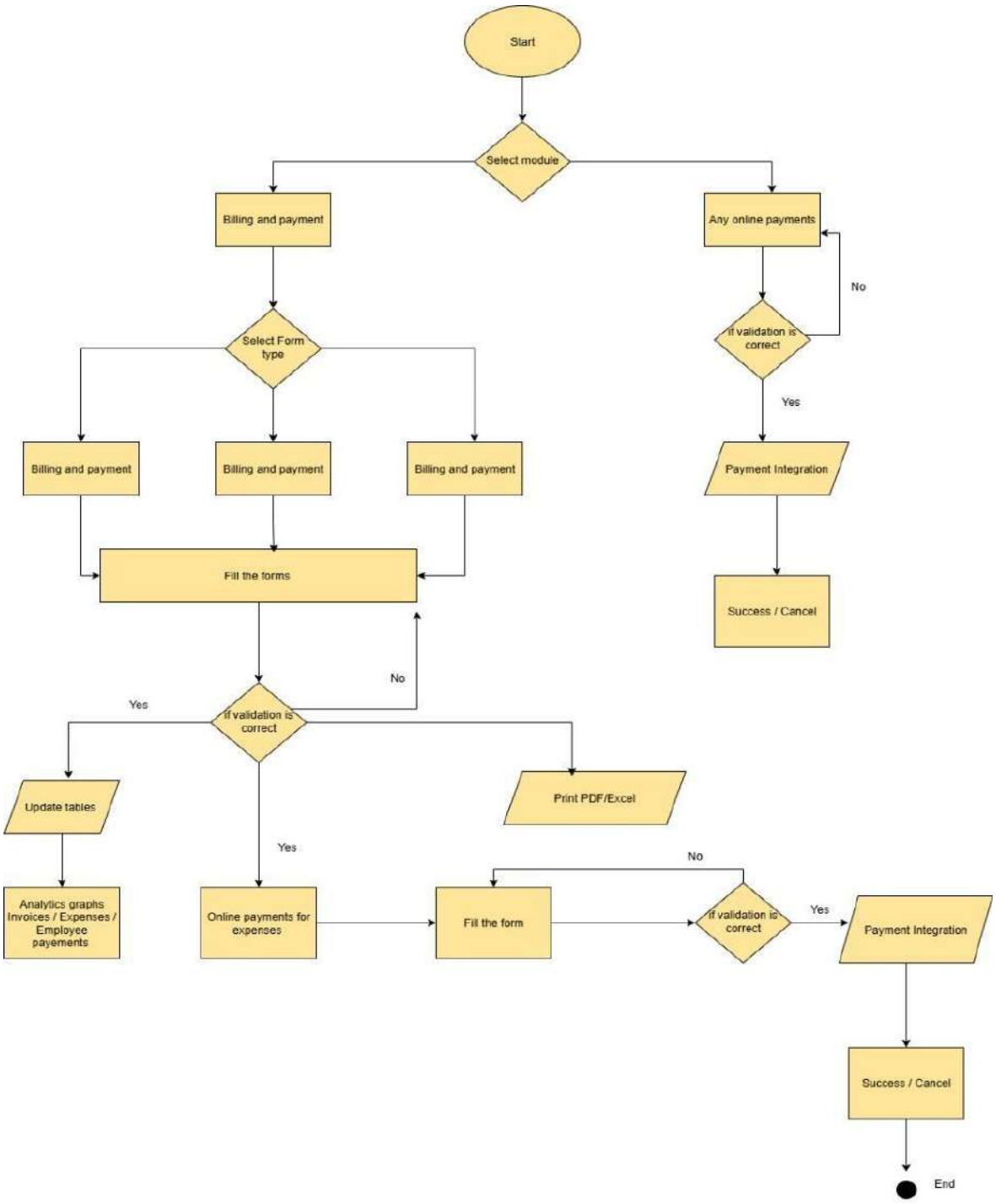
Reduces paperwork, ensures fast processing, and improves accuracy through automated scheduling, AI-driven analysis, and instant notifications.


start
Input Lab Assistant credentials
Login to the Lab assistant dashboard
Check notification panel for new requests
No
Yes
Navigate to "Appointments"
Navigate to "New requests"
Action on appointments
Delete
Edit
Check scheduled appointments
View new patient forms
Select form to book a appointment
Confirm deletion
Modify details and Save
See Daily schedule
Click "New Appointment"
Labform
Appointment removed from database
Database Updated
Labform
Display incorrect validations
Enter appointment details according to the patient request form
No
Validate the form
Yes
Appointment Send to the database
Labform
Navigate to the Scheduled appointments
Click "Send email"
End
If email sent successfully
No
Display unsuccessful message
Yes
Display successful message
Patient receives confirmation email.

Billing & Payment Workflow

1. **Service Completion:**
   Once a lab test or medical service is done, the admin or staff generates an invoice.
2. **Payment Processing:**
   Payment is completed manually or through the **PayHere online gateway**.
3. **Invoice Update:**
   The system marks invoices as *Paid* or *Pending* automatically.
4. **Expense and Salary Recording:**
   Staff payments and expenses are entered into and validated.
5. **Report Generation:**
   Monthly and weekly financial summaries are created for the finance team.
6. **Data Backup:**
   Billing records are stored securely in the database for future audits.

Result:
All financial data stays accurate and transparent, ensuring smooth hospital accounting.



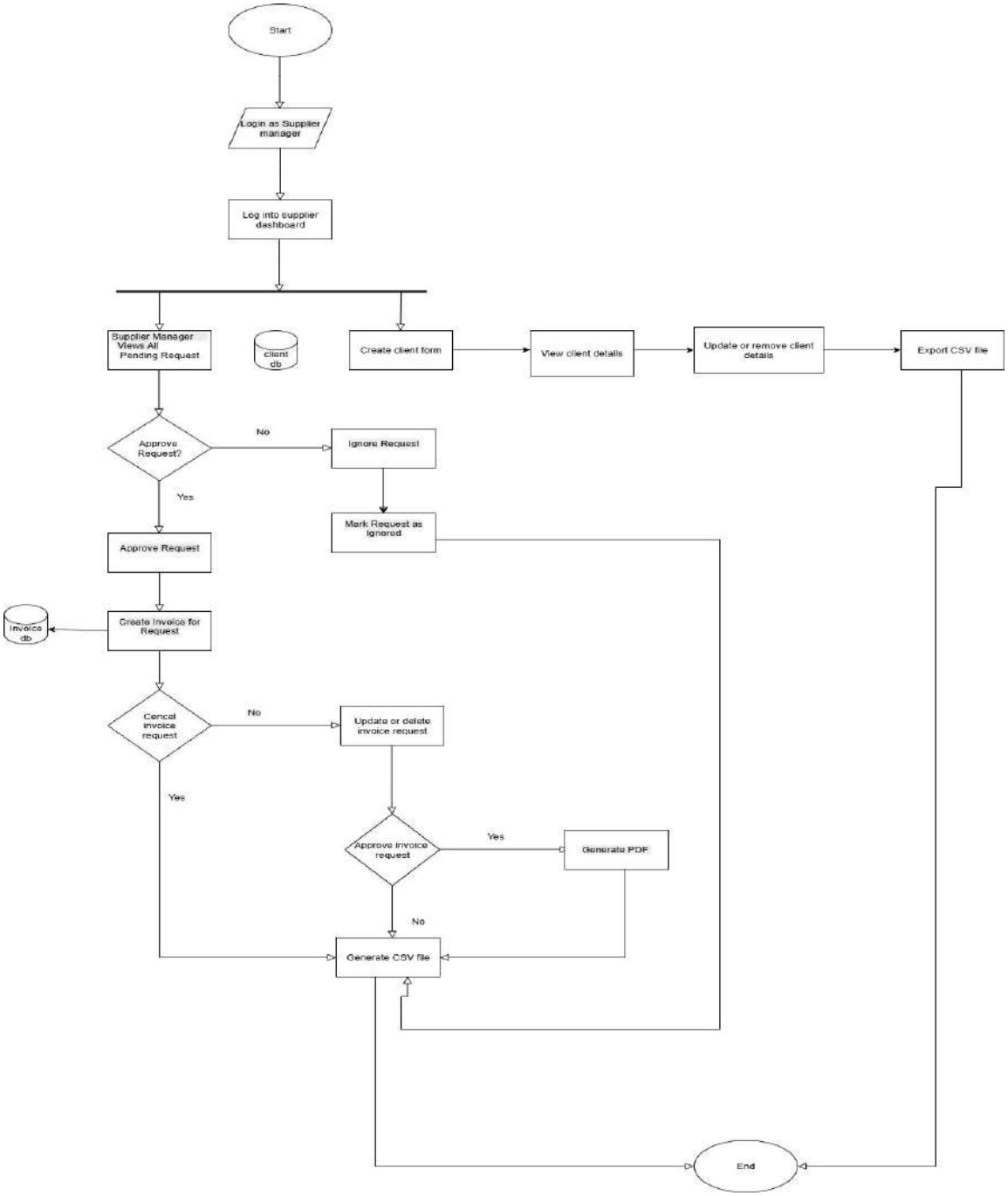
Supplier Management Workflow

1. **Inventory Manager Sends a Request:**
   A low-stock alert triggers an automatic supply request to the supplier.
2. **Supplier Reviews Request:**
   Supplier Manager views pending requests and either approves or ignores them.
3. **Invoice Creation:**
   For approved requests, a supplier invoice is generated. 4.

Supplier Updates or Deletes Invoice (if needed):
   Allowed until the inventory manager confirms receipt.

5. **Invoice Export:**
   Invoices can be downloaded as PDF or CSV for record-keeping.
6. **Supplier Dashboard Update:**
   Dashboard reflects real-time request and invoice statuses.

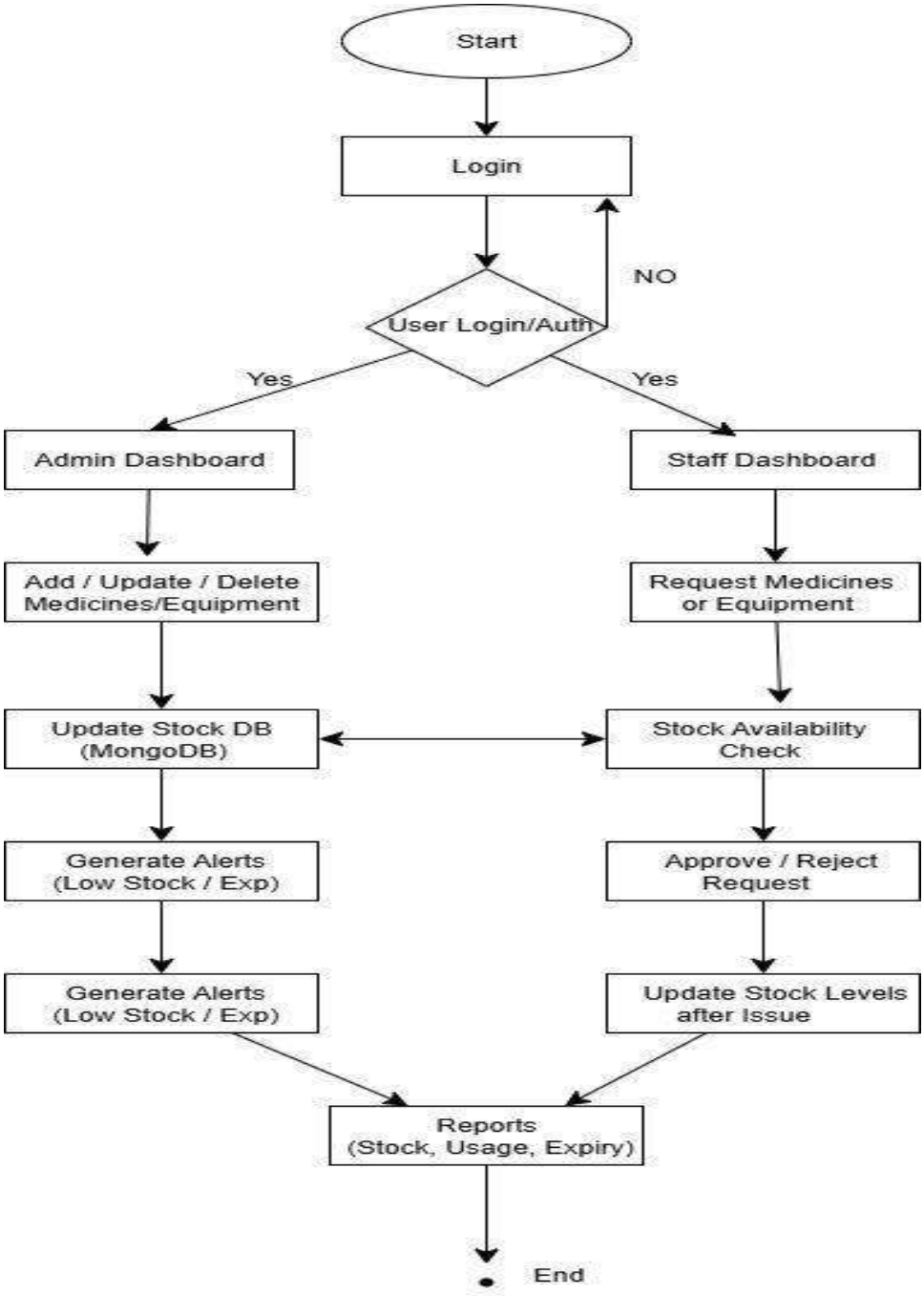
Result:
Creates a clear and accountable link between suppliers and the medical center.



Inventory Management Workflow

1. **Stock Level Monitoring:**
   System continuously monitors available stock in the database.
2. **Low-Stock Alert Triggered:**
   When items fall below threshold (e.g., <10 units), alerts are shown.
3. **Inventory Request Sent:**
   Automatic request goes to Supplier Management module.
4. **Stock Update After Delivery:**
   Once supplier confirms, inventory count updates automatically.
5. **Item Addition or Deletion:**
   New items can be added or outdated ones removed.
6. **Report Generation:**
   Inventory, invoices, and requests exported as PDFs or CSVs.

Result:
Ensures real-time visibility of stock levels and prevents medicine shortages.



Normalized Schema

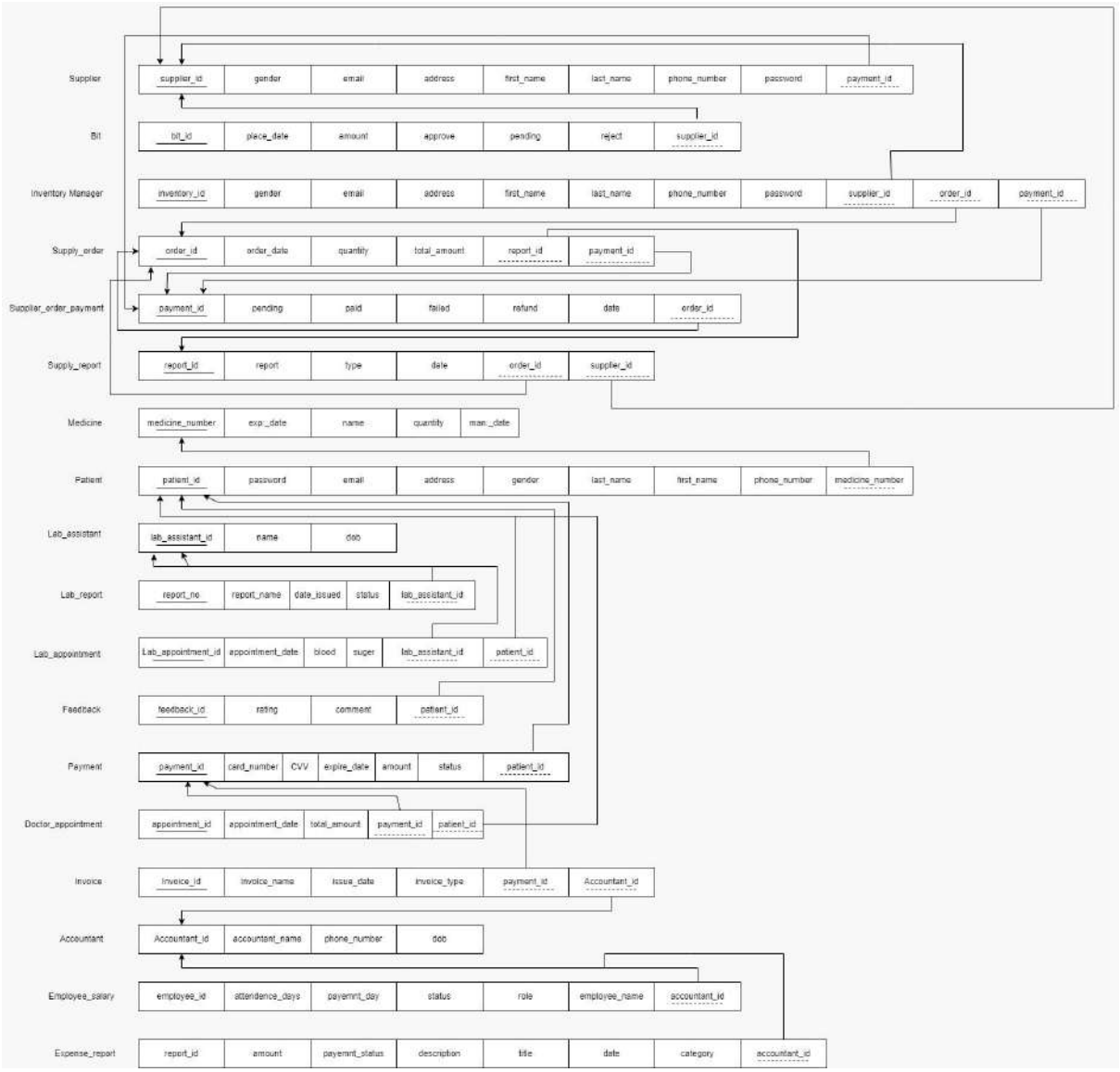


Figure 3.3

ER Diagram

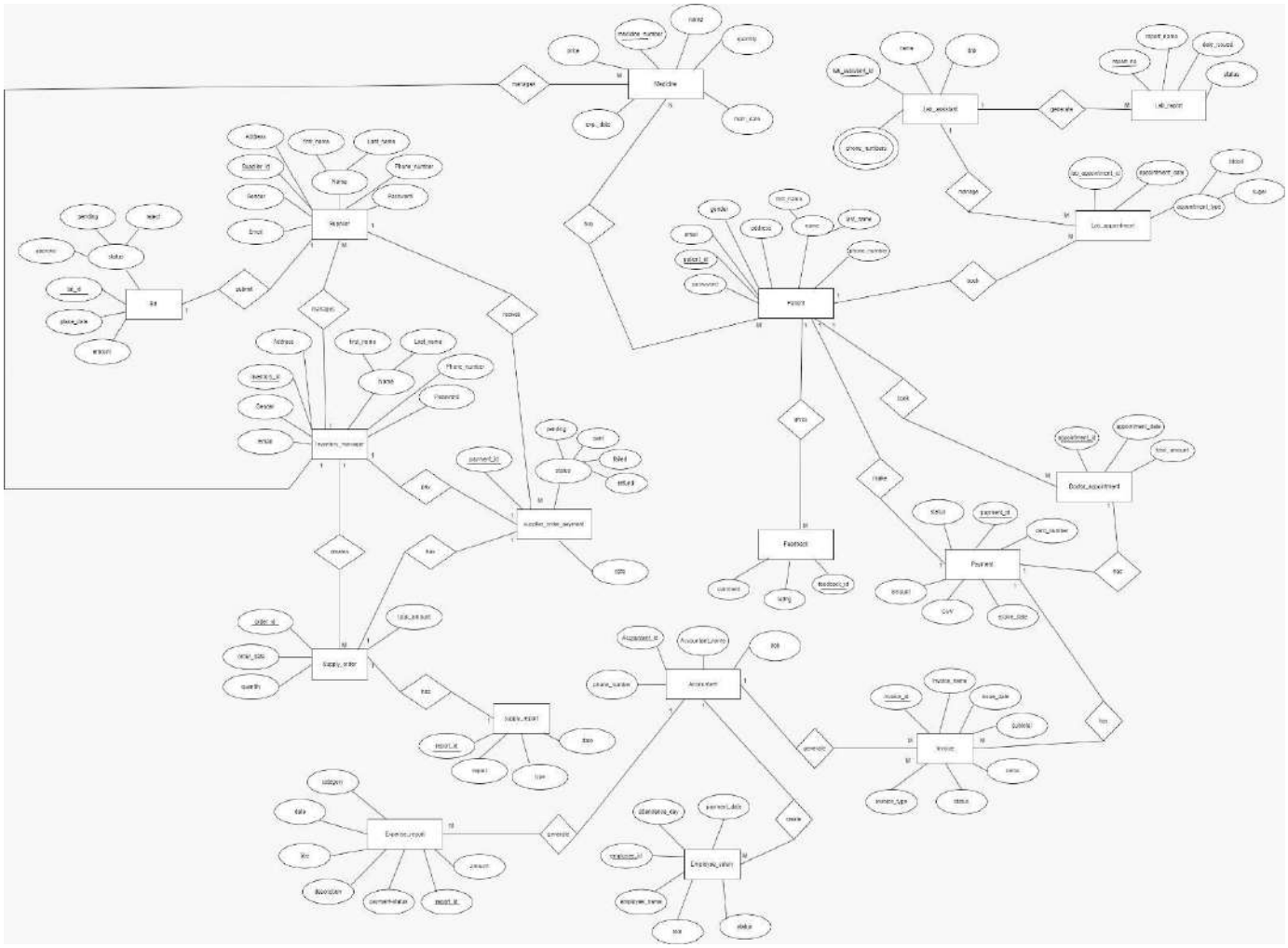


Figure 3.4

Development Process

The development followed an **Agile approach**, with iterative updates and feedback from users at every stage.

Development Stages

1. **Requirement Gathering** – Meetings with hospital staff and admins.
2. **UI/UX Design** – Wireframes and prototypes built in Figma.
3. **Frontend Development** – React.js components created for each module.
4. **Backend Development** – RESTful APIs developed using Express.js.
5. **Database Integration** – MongoDB configured for CRUD operations.
6. **Testing & Debugging** – Unit and integration tests performed.
7. **Deployment** – Application hosted and connected with PayHere payment gateway.

Technologies Used

- **Frontend:** React.js, HTML, CSS, JavaScript
- **Backend:** Node.js, Express.js
- **Database:** MongoDB
- **Version Control:** GitHub
- **API Testing:** Postman
- **Payment Gateway:** PayHere API

High-level system design diagram

MediCura is a hospital management web application that operates under the management of **user management, laboratory management, inventory management, billing & payments, and supplier management**. It is primarily developed using the **MERN web development stack**. This ensures **efficient data processing, real-time updates, and improved accuracy**, enabling hospitals to provide optimal service to patients, staff, and stakeholders.

The technologies used in the system are as follows:

- **Frontend:** React.js
- **Backend:** Express.js + Node.js
- **Database:** MongoDB **System modules**

1. **User Management** – Handles patient/staff registration, authentication, and rolebased access.
2. **Laboratory Management** – Manages test bookings, results, and reports.
3. **Inventory Management** – Tracks medicines, equipment, and stock levels.
4. **Billing & Payments** – Handles invoices, receipts, online payments, and salary management.
5. **Supplier Management** – Manages supplier details, orders, and procurement records.

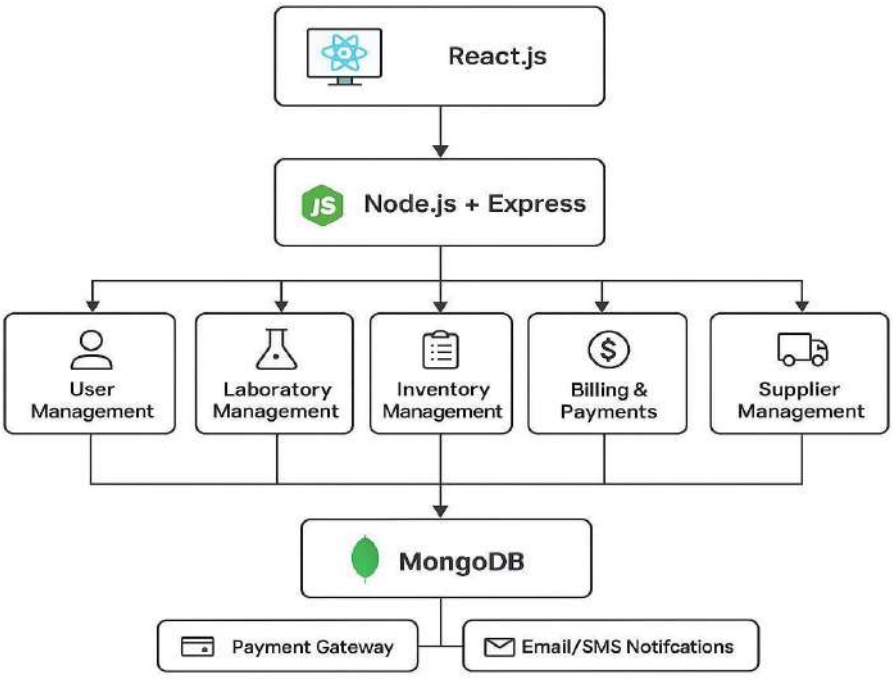


Figure 3.5

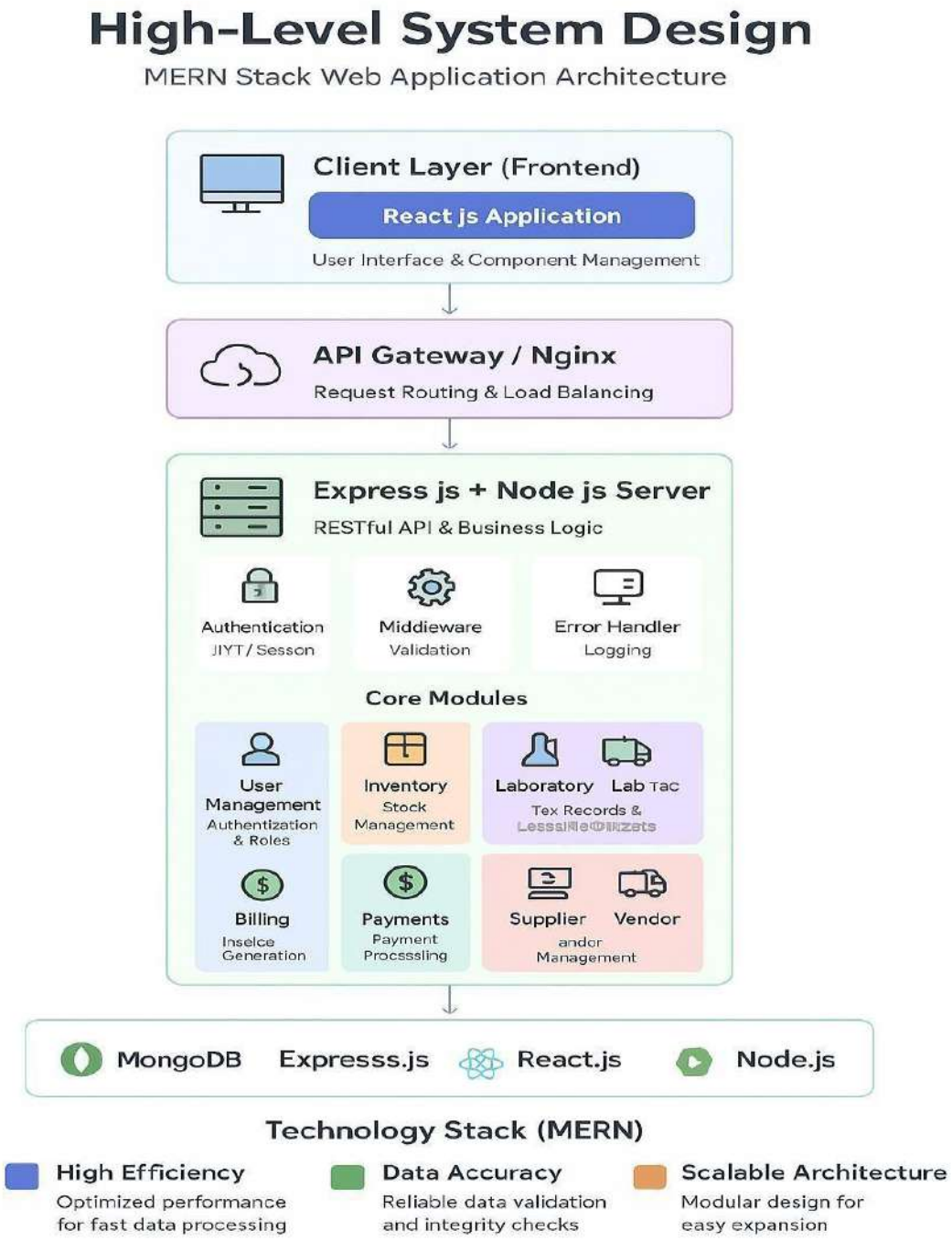

High-Level System Design
MERN Stack Web Application Architecture
Client Layer (Frontend)
React js Application
User Interface & Component Management
API Gateway / Nginx
Request Routing & Load Balancing
Express js + Node js Server
RESTful API & Business Logic
Authentication
JIYT / Sesson
Middieware
Validation
Error Handler
Logging
Core Modules
User Management
Authentization & Roles
Inventory
Stock Management
Laboratory
Lab Tac
Tex Records &
Billing
Inselce Generation
Payments
Payment Procsssling
Supplier
Vendor
andor Management
MongoDB
Expresss.js
React.js
Node.js
Technology Stack (MERN)
High Efficiency
Optimized performance for fast data processing
Data Accuracy
Reliable data validation and integrity checks
Scalable Architecture
Modular design for easy expansion

Figure 3.6

Diagram flow:

- **Client UI (React.js) → Application Server (Node.js + Express) → Database Server (MongoDB)**
- APIs for each module (User, Lab, Inventory, Billing, Supplier) interact with the database.
- External integrations: **Payment Gateway** (PayHere/Stripe) + **Email/SMS notifications**.

Testing

The system was tested to ensure all main modules-**User Management**, **Laboratory Management**, **Billing & Payments**, **Supplier Management**, and **Inventory Management** met the project's requirements.
Each function was verified against its **acceptance criteria**, meaning it works as expected, gives the correct outputs, and interacts properly with other modules.

Acceptance Criteria

1. Users can register, log in, and access features according to their roles.
2. All forms include validation for required fields and show clear error messages.
3. Data entered by users is stored and retrieved correctly from the MongoDB database.
4. Role-based dashboards load correctly and show real-time updates.
5. System sends notifications or emails where applicable (e.g., lab confirmations, supplier approvals).
6. CRUD (Create, Read, Update, Delete) operations work smoothly across all modules.
7. PDF and CSV exports generate correct, downloadable files.
8. Online payment (PayHere) completes transactions successfully.

Main Test Cases and Results

Laboratory Management

Scenario	**Expected Result**	Status
Patient submits lab request form	Form saves successfully in the database	☑ Passed
Required fields left empty	System shows clear error messages	☑ Passed
View pending lab requests	All requests display correctly	☑ Passed
Edit existing lab record	Changes saved successfully	☑ Passed
Delete lab record	Record removed and confirmation shown	☑ Passed
Upload valid lab report	File stored and attached to patient record	☑ Passed
Upload invalid lab report	System shows “Unsupported file format”	☑ Passed
AI Bot analyzes uploaded report	Returns summary of key findings	☑ Passed
Search by patient name or date	Relevant results displayed instantly	☑ Passed
Send email confirmation to patient	Email successfully delivered	☑ Passed
Reschedule appointment	New time saved and notified to user	☑ Passed
Export report as PDF	Download successful and correctly formatted	☑ Passed

Table 4.1

User Management

Scenario	Expected Result	Status
Account created and stored in database	Form saves successfully in the database	☑ Passed
Register with existing email	"Email already exists" message shown	☑ Passed
Login with valid credentials	User redirected to correct dashboard	☑ Passed
Login with wrong password	Error "Invalid username or password" displayed	☑ Passed
Forgot password	Password reset link sent to email	☑ Passed
Update profile details	Profile updated and saved successfully	☑ Passed
Logout from system	Session cleared and login screen displayed	☑ Passed
Attempt to access admin page as user	Access denied message displayed	☑ Passed
System session timeout	User logged out automatically after idle time	☑ Passed
Password validation check	Weak password rejected with warning	☑ Passed
Admin creates new staff user	New user account created successfully	☑ Passed
Delete existing user	Account removed and confirmation shown	☑ Passed

Table 4.2

Billing & Payment Management

Scenario	Expected Result	Status
Create new invoice	Invoice saved and assigned unique ID	☑ Passed
Edit invoice details	Updated data saved correctly	☑ Passed
Record removed from system	Record removed from system	☑ Passed
Add new expense	Expense added and reflected in summary	☑ Passed
Enter invalid expense amount	Error message displayed	☑ Passed
Generate payment report	Accurate report displayed	☑ Passed
Complete PayHere payment	Status updated to “Paid”	☑ Passed
Cancel online payment	Status shown as “Cancelled”	☑ Passed
Generate invoice receipt	PDF downloaded successfully	☑ Passed
Add employee payment record	Saved and visible in payroll list	☑ Passed
Calculate total bill with taxes	Correct total displayed	☑ Passed
Search invoice by patient name	Correct record displayed	☑ Passed

Table 4.3

Supplier Management

Scenario	**Expected Result**	Status
Approved inventory requests	Status updates to “Approved”	☑ Passed
Rejecting pending request	Status updates to “Ignored”	☑ Passed
Create new inventory invoice	Invoice generated successfully	☑ Passed
Edit inventory invoice	Updates saved accurately	☑ Passed
Delete inventory invoice	Record deleted and confirmed	☑ Passed
Add new clients	Client information saved correctly	☑ Passed
Update clients contact info	New details stored successfully	☑ Passed
Attempt to delete approved invoice	System denies with warning	☑ Passed
Export invoices as CSV	File downloaded and formatted	☑ Passed
View all inventory requests	List displayed with correct data	☑ Passed
Filter clients, requests and invoices	Matching results shown	☑ Passed
Generate client report	Summary report created successfully	☑ Passed

Table 4.4

Inventory Management

Scenario	Expected Result	Status
Add new stock item	Item appears in stock list	☑ Passed
Add duplicate item	System displays “Item already exists”	☑ Passed
Update or delete item	Changes reflect immediately	☑ Passed
Enter invalid quantity	Error message prompts correction	☑ Passed
Low-stock alert (<10 items)	Warning message shown	☑ Passed
Create inventory request	Request generated successfully	☑ Passed
Export inventory, invoices, requests report as PDF	File downloads successfully	☑ Passed
Approve or cancel inventory invoice	Status updated	☑ Passed
Search invoices, requests	Matching results shown	☑ Passed
Filter item by category, name, price and stock available	Matching results shown	☑ Passed
Generate monthly stock report	Report generated with correct totals	☑ Passed
Update or Delete inventory request until supplier manager approved or declined	Changes reflect immediately	☑ Passed

Table 4.5

Overall Results Summary

Module	Total Test Cases	Passed	Ongoing / Incomplete
User Management	12	12	0
Laboratory Management	12	12	0
Billing & Payments Management	12	12	0
Supplier Management	12	12	0
Inventory Management	12	12	0

Table 4.6

Total test cases = 12 * 5 = 60

Passed test cases = 12 * 5 = 60

Ongoing test cases = 0

Overall pass rate = (60/60) * 100

= 100%

Evaluation & Conclusion

Evaluation

1. **User Management Workflow**
    - **Functionality Testing:** The system was tested for all user roles Admin, Lab Assistant, Supplier, Inventory Manager, and Finance Staff. Each role successfully registered, logged in, and accessed its specific dashboard without issues.
    - **Security & Authentication:** Secure password hashing and session management ensured only authorized access. Experts confirmed that role-based permissions were implemented effectively.
    - **Usability:** Users found the workflow intuitive. For instance, Finance Staff could directly access financial reports, while Lab Assistants had streamlined access to lab data, reducing unnecessary navigation.
2. **Android App UI & Development**
    - **Layout & Design:** Centered elements, tab highlights, and corrected XML layouts significantly improved visual appeal and usability.
    - **Error Handling:** Resource extraction and XML errors were resolved, reducing app crashes and improving stability across devices.
    - **User Feedback:** Testers appreciated the cleaner interface, responsive layouts, and simplified navigation, especially in the home and profile sections.
3. **React Component (Badge)**
    - **Performance & Flexibility:** The Badge component worked reliably across multiple sizes (sm, md, lg) and variants (primary, secondary, success, etc.).
    - **Maintainability:** Its modular design minimized code duplication and allowed for easy reuse in different parts of the system.
    - **Visual Feedback:** Users highlighted that badges improved information visibility without cluttering the interface.
4. **Presentation & Data Summaries**
    - **Clarity:** Summaries of the Smart Traffic project made complex ideas easy to understand, even for non-technical audiences.
    - **Expert Feedback:** Mentors and peers praised the concise, jargon-free presentation style.
    - **Effectiveness:** Test presentations showed that key points were retained, demonstrating successful communication of the project's purpose and benefits.

Conclusion

The project successfully achieved its intended objectives:

- **System Functionality:** The user management system works efficiently, providing secure and role-specific access.
- **User Experience:** The Android app interface is visually appealing, easy to navigate, and free from major errors.
- **Code Quality:** Modular React components like the Badge enhance maintainability and flexibility.
- **Communication:** Simplified summaries and presentations effectively convey project goals to both technical and non-technical audiences.

Final Insight:

The combination of technical robustness, usability, and clear communication confirms that the project has met its aim. Users can interact with the system intuitively, and the objectives of security, efficiency, and clarity have been fully realized. This demonstrates a successful integration of functionality, design, and effective information delivery.

References

Referencing Sources

- Al-Mutairi, H. A.-M. (2020). *Improving healthcare appointments scheduling system using web-based technologies. International Journal of Advanced Computer Science and Applications, 11(6), pp.657–662. Available at: https://doi.org/10.14569/IJACSA.2020.0110679 [Accessed 7 Aug. 2025].*
- Bassi, J. &. (2013). *Measuring value for money: a scoping review on economic evaluation of health information systems. Journal of the American Medical Informatics Association, 20(4), pp.792–801. Available at: https://doi.org/10.1136/amiajnl-2012-001422 [Accessed 7 Aug. 2025]*.
- Corporation, E. S. (2022). Retrieved from Epic EMR Software Overview. [online] Available at: https://www.epic.com/ [Accessed 7 Aug. 2025].
- HealthHub. (2023). Retrieved from HealthHub - Your Digital Health Companion. [online] Available at: https://www.healthhub.sg/ [Accessed 7 Aug. 2025].
- Kumar, R. &. (2021). Retrieved from Inventory management in healthcare: A casebased approach to reduce drug wastage in hospitals. Journal of Supply Chain Management Systems, 10(1), pp.34–45.
- Nielsen, J. (1993). Retrieved from Usability Engineering. 1st ed. San Diego: Academic Press.
- Patel, K. S. (2020). Retrieved from Integrated hospital management system for pharmacy and billing. International Journal of Engineering and Advanced Technology, 9(5), pp.520–524. Available at: https://doi.org/10.35940/ijeat.E9498.069520 [Accessed 7 Aug. 2025].
- Rahman, A. &. (2021). Retrieved from The impact of electronic medical records on healthcare quality: A literature review. International Journal of Medical Informatics, 150, p.104454. Available at: https://doi.org/10.1016/j.ijmedinf.2021.104454 [Accessed 7 Aug. 2025].

Referencing tools

1. Web Development Tools / IDEs
   For coding websites from scratch:
   - **VS Code** – Free, lightweight, supports HTML, CSS, JavaScript, and frameworks like React, Angular, and Vue.

2. Front-End Frameworks & Libraries
   For faster and responsive website development:
   - **React.js** – For building interactive web applications.
   - **Tailwind CSS** – Prebuilt CSS frameworks for responsive design.

3. Back-End / Database Tools
   For server-side development and database management:
   - **Node.js + Express.js** – For JavaScript-based backend.

4. Hosting & Deployment Tools
   For putting your website online:
   - **GitHub Pages** – Free hosting for static websites.
   - **AWS / Google Cloud / Azure** – For scalable web hosting.

5. Website Design & Optimization Tools
   For designing and improving your website:

- **Figma / Adobe XD** – UI/UX design tools.
- **Canva** – Simple graphics and web visuals.

- **Front-End:** JavaScript + React.js
- **Back-End:** Node.js + Express.js
- **Database:** MongoDB
- **Deployment:** Netlify
- **Design:** Figma + Tailwind CSS
- **Version Control:** Git + GitHub

Individual Contribution

Component	Student Name	Student ID
User Management	K.H.S. Dinsara	IT23588714
Supplier Management	Navodya A.K.	IT23543232
Laboratory Management	Thathsarani J.R.	IT23572638
Inventory Management	Liyanaarachchi L.A.D.A.	IT23669758
Invoice And Billing Management	Ekanayaka E.W.I.D	IT23810464

Table 7.1

For Documentation

Student Name	Contribution
K.H.S. Dinsara	• Contribute to the use case diagram and user case scenario for user management. • Draw component diagram user management function • Create test case for user management • Implement all information About the user management function
Navodya A.K.	• Contribute to the use case diagram and user case scenario for supplier management. • Draw component diagram supplier management function • Create test case for supplier management • Implement all information About the supplier management function

Thathsarani J.R.	• Contribute to the use case diagram and user case scenario for laboratory management. • Draw component diagram laboratory management function • Create test case for laboratory management • Implement all information About the laboratory management function
Liyanaarachchi L.A.D.A.	• Contribute to the use case diagram and user case scenario for inventory management. • Draw component diagram inventory management function • Create test case for inventory management • Implement all information About the inventory management function
Ekanayaka E.W.I.D	• Contribute to the use case diagram and user case scenario for billing & payment management. • Draw component diagram billing & payment management function • Create test case for billing & payment management • Implement all information About the billing & payment management function