**DCLD Navigator: Blood Inspection to Reduce Complications in DCLD**

**A MINI PROJECT REPORT**

***Submitted to***

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

***In partial fulfillment of the award of the degree of***

**BACHELOR OF ENGINEERING**

***By***

**Name and Register Number: K. Bharath Teja - 192111113**

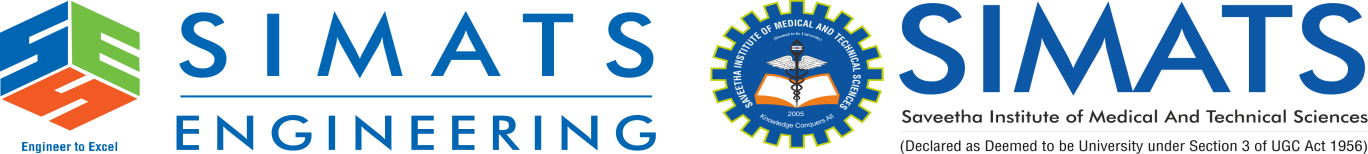


**SIMATS ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES,**

**CHENNAI – 602 105**

**MONTH AND YEAR OF EXAM**



# **SIMATS ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

### **CHENNAI – 602105**

### **BONAFIDE CERTIFICATE**

Certified that this product development report **“**PROJECT TITLE**”** is the Bonafide work of **“K. Bharath Teja** - 192111113”who carried out the product development work under my supervision.

**NAME OF THE HOD** **NAME OF THE SUPERVISOR**

# **HEAD OF THE DEPARTMENT**

Professor

Department of Information Technology

SIMATS Engineering

Saveetha Institute of Medical and Technical Sciences

Chennai – 602105

# **SUPERVISOR**

Professor

Department of Information Technology

SIMATS Engineering

Saveetha Institute of Medical and

Technical Sciences

Chennai - 602105

## **INTERNAL EXAMINER EXTERNAL EXAMINER**

### **INDEX**

| **S.NO** | **TOPICS** | **PAGE NO** |
| --- | --- | --- |
| **1** | EXECUTIVE SUMMARY | **1** |
| **2** | INTRODUCTION | **1** |
| **3** | GPCU | **2** |
| **4** | STAKEHOLDERS | **5** |
| **5** | ROADMAP FOR THE PROJECT | **6** |
| **6** | DESIGN AND ENGINEERING STANDARDS | **9** |
| **7** | PROTOTYPE DESCRIPTION: PROTOTYPE DESIGN (FIGMA OR SIMILAR TOOL) | **10** |
| **8** | SOFTWARE AND HARDWARE SPECIFICATIONS | **17** |
| **9** | DEVELOPMENT | **17** |
| **10** | TESTING AND VALIDATION | **20** |
| **11** | IMPLEMENTATION AND DEPLOYMENT | **20** |
| **12** | DEVELOPMENT CHALLENGES | **21** |
| **13** | QUESTIONS ASKED BY REVIEW PANEL | **21** |
| **14** | CONCLUSION AND FUTURE WORK | **22** |
| **15** | REFERENCES | **23** |
| **16** | APPENDIX | **24** |

**DCLD Navigator: Blood Inspection to Reduce Complications in DCLD**

**Executive Summary:**Decompensated Liver Disease (DCLD) is a severe and complex condition requiring constant monitoring, personalized care, and proactive intervention to manage symptoms and prevent complications. This mobile application directly addresses these challenges, offering an innovative platform that bridges the gap between patients and healthcare providers.

DCLD patients face critical needs for daily monitoring, medication adherence, and early detection of warning signs to prevent rapid disease progression. Despite the severity of this condition, current healthcare systems lack a streamlined, real-time tool to support both patients and their doctors in collaborative disease management. This app fills that gap, leveraging digital tools to enhance patient engagement, reduce readmissions, and improve outcomes.

This app stands out through its dual-portal design for doctors and patients, providing tailored features that foster interactive and effective healthcare management. Its real-time alerts, personalized care plans, and intuitive data visualization set it apart from generic health-tracking applications.

**Introduction:**

**2.1 Problem Statement:**Decompensated Liver Disease (DCLD) requires continuous monitoring, early detection of complications, and strict adherence to treatment, yet current healthcare systems lack tools for real-time, patient-specific management. This gap leads to delayed interventions, poor symptom tracking, and high rates of preventable hospital readmissions. The growing prevalence of DCLD underscores the urgent need for a solution that integrates personalized care, real-time alerts, and seamless doctor-patient communication.

**2.2 Purpose:**The primary objective of this application development project is to empower patients with Decompensated Liver Disease (DCLD) and their healthcare providers with tools for real-time monitoring, personalized care, and proactive intervention. By integrating patient and doctor portals, the app fosters seamless communication and enhances collaboration in managing this critical condition. It aims to improve drug adherence, detect complications early, and reduce hospital readmissions. The app also supports patients in maintaining quality of life through tailored rehabilitation plans. Ultimately, it seeks to bridge the gap in DCLD care and optimize clinical outcomes.

**2.3 Scope:**This project involves developing a mobile application tailored for Decompensated Liver Disease (DCLD) management, featuring separate patient and doctor portals. The app will include tools for symptom tracking, medication adherence, and personalized care plans, along with real-time monitoring and alerts. Initial development will focus solely on DCLD, with limited third-party device integration and reliance on internet connectivity for updates. Future expansions may address broader liver conditions. The goal is to deliver an effective and user-friendly solution for collaborative care.

**GPCU**

**3.1 Gap Analysis**

**Market Gap or Lacune:**Existing healthcare apps lack specialized features tailored to the needs of Decompensated Liver Disease (DCLD) patients, such as real-time monitoring, personalized care plans, and seamless doctor-patient collaboration. Studies show high readmission rates (over 40% within 30 days for cirrhosis patients) due to delayed symptom detection and poor drug adherence. User feedback highlights frustration with generic health trackers that fail to address DCLD-specific challenges like dietary management or fluid intake monitoring. This app bridges the gap by providing a focused, disease-specific solution to improve outcomes and quality of life.

**3.2 Application Description:**

**Application Overview:**The application is designed to address the critical needs of patients with Decompensated Liver Disease (DCLD) by providing real-time monitoring, personalized care plans, and seamless communication between patients and healthcare providers. It aims to bridge the gap in existing healthcare tools by offering a dedicated platform for symptom tracking, medication adherence, and early detection of complications, which are often overlooked by generic health apps.

Key Features and Functions:

Doctor Portal: Provides a comprehensive dashboard for managing patient records, monitoring vitals, reviewing lab results, and receiving real-time alerts for abnormal symptoms.

Patient Portal: Allows patients to track daily symptoms, record vitals, manage medication schedules, and receive personalized care instructions, including dietary guidelines and rehabilitation plans.

Real-Time Alerts and Data Visualization: Sends instant notifications for abnormal symptoms and health data changes, along with visual representations (graphs) of patient progress.

Interactive Questionnaires: Patients can complete regular symptom questionnaires to ensure timely intervention based on current health status.

Core Technologies:

* Frontend: Developed using React Native, ensuring cross-platform compatibility for both Android and iOS devices, providing a seamless user experience across platforms.
* Backend: Utilizes XAMPP for local server management, with MySQL as the database for storing patient data, and PHP for building secure APIs that handle communication between the front end and the database.

**3.3 Comparison of Alternative Applications:**

| **Application Name** | **Key Features** | **Strengths** | **Weaknesses** |
| --- | --- | --- | --- |
| MyCirrhosis | Symptom tracking, Medication reminders, Patient education | Strong brand, Comprehensive health data | Expensive, Limited customization |
| Liver Care | Daily symptom logs, Doctor-patient chat, Medication adherence | Affordable, User-friendly interface | Lack of real-time monitoring, Basic features |
| LiverTox | Fluid intake tracking, Weight monitoring, Real-time alerts | Advanced monitoring, Real-time alerts | High cost, Limited disease-specific focus |

**Analysis:**

* MyCirrhosis offers a robust set of features, including symptom tracking and medication reminders, but its high cost may be a barrier for some users, and it doesn't offer much customization.
* Liver Care is more affordable and easier to use, but it lacks the real-time monitoring and specific tracking tools needed for DCLD management, such as fluid intake and early symptom alerts.
* LiverTox excels with advanced monitoring features and real-time alerts but is priced higher and doesn't focus specifically on the unique needs of DCLD patients, limiting its appeal for targeted care.

**3.4 Uniqueness of Your Application:**

Unlike competing applications, our app specifically targets the needs of Decompensated Liver Disease (DCLD) patients with real-time symptom monitoring, personalized care plans, and seamless doctor-patient communication. It addresses the gap in existing solutions by offering advanced features like fluid intake tracking, weight monitoring, and real-time alerts for complications, which many apps lack. Our app also provides detailed, customizable health dashboards for both patients and doctors, improving patient engagement and adherence to treatment. In contrast to expensive, generic apps, our solution is affordable, tailored, and focuses solely on DCLD care. These innovations ensure more effective management, reducing hospital readmissions and improving patient outcomes.

1. **Stakeholders**

* **End Users (Primary Users):**
* **DCLD Patients:** They directly use the app for daily symptom tracking, medication adherence, and accessing personalized care plans. Their engagement is critical for successful disease management and improved quality of life.
* **Healthcare Providers (Doctors, Nurses):** They use the doctor portal to monitor patient health, provide medical advice, and update care plans. Timely intervention based on real-time data is essential for better patient outcomes.
* **Secondary Users:**
* **Family Members or Caregivers:** They may indirectly interact with the app by helping patients track their health data or provide additional support for adherence to treatment and monitoring.
* **Business Owners (App Developer):**
* **App Developer:** Responsible for building, maintaining, and updating the app, ensuring that it functions smoothly and meets both patient and healthcare provider needs.
* **Regulators:**
* **Healthcare Regulators:** Ensure that the app complies with healthcare standards, such as data privacy regulations (e.g., HIPAA in the US, GDPR in Europe) and other medical device regulations to protect patient data and ensure its clinical efficacy.
* **Regulatory Authorities:** Oversee the app's certification and ensure it meets medical and technological standards, making it safe and reliable for healthcare use.

1. **Roadmap for the Project**

* **Phase 1: Project Initiation**
  + Objective:
    - Lay the foundation for the DCLD Navigator app by identifying user needs, requirements, and the scope of work.
  + Tasks:
    - Market Research:
      * Analyze challenges in managing Decompensated Liver Disease (DCLD) such as symptom tracking, medication adherence, and readmission rates.
      * Identify gaps in existing solutions for monitoring and rehabilitation of DCLD patients.
    - Requirement Gathering:
      * Engage with healthcare professionals and patients to define core app features.
      * Outline requirements for real-time data tracking, care plans, vitals monitoring, and alert systems.
    - Deliverables:
      * Market Research Report
      * Functional and Non-Functional Requirements Document
      * Compliance Framework Document
* **Phase 2: Design and Planning**
  + UI/UX Design
    - Wireframes and Prototypes:
      * Use Figma to design wireframes for the app's core features:
    - Onboarding Screens: Step-by-step guidance for first-time users.
    - Doctor Dashboard: Manage patient details, vitals, assessments, and discharge summaries.
    - Patient Dashboard: Record daily vitals, symptoms, and medication adherence.
    - Real-Time Graphs: Visual representation of health progress and alerts.
    - Care Plans and Guidelines: Personalized recommendations and dietary guidelines.
    - Notifications: Alerts for symptom changes, medication schedules, and appointments.
  + User Flows:
    - Map user journeys for critical interactions:
    - Recording daily vitals and symptoms.
    - Accessing health graphs, care plans, and discharge summaries.
    - Adding or updating patient notes and alerts (Doctor Portal).
  + Accessibility Features:
    - Clear fonts, high-contrast colors, and intuitive navigation.
    - Support for multilingual features for better usability across regions.
* **Phase 3: Development**
  + Objective:
    - Develop the DCLD Navigator app with quality, security, and performance in mind.
  + Tasks:
    - Backend Development:
    - Implement the database for patient data, vitals, and care plans.
    - Develop APIs for real-time alerts, data visualization, and medication tracking.
    - Frontend Development:
      * Build intuitive interfaces for both Doctor and Patient Portals using React Native.
      * Integrate interactive features like health graphs, symptom tracking, and questionnaires.
    - Real-Time Alert System:
      * Implement logic for sending timely alerts based on symptom deterioration or abnormal vitals.
    - Security Implementation:
      * Secure data storage and communication with encryption protocols.
      * Adhere to ISO/IEC 27001 for data protection.
    - Testing and Quality Assurance:
      * Conduct functional, usability, and performance testing.
      * Ensure reliability through ISO 9001-aligned quality checks.
    - Deliverables:
      * Fully Functional App (Alpha Version)
      * Testing and Quality Assurance Reports
      * Security Audit Report
    - Milestone:
      * Alpha version release for internal testing.
* **Phase 4: Pilot Testing and Feedback**
  + Objective:
    - Test the app in real-world scenarios to gather feedback and improve its functionality.
  + Tasks:
    - Pilot Launch:
      * Deploy the app among healthcare providers and DCLD patients.
    - User Feedback Collection:
      * Gather insights on usability, alert accuracy, and overall effectiveness.
    - Performance Monitoring:
      * Track user engagement, data input rates, and responsiveness of real-time alerts.
    - Improvements:
      * Refine features and optimize performance based on user feedback.
  + Deliverables:
    - User Feedback Report
    - Pilot Performance Metrics
    - Improved Beta Version of the App
  + Milestone:
    - Successful pilot testing and app refinement.
* **Phase 5: Deployment and Launch**
  + Objective:

Officially launch the DCLD Navigator app for public use.

* + Tasks:
    - App Store Deployment: Publish the app on Google Play Store and Apple App Store.
    - Marketing and Awareness Campaign:
      * Promote the app among hospitals, healthcare providers, and patient communities.
      * Focus on outreach to regions with high DCLD prevalence.
    - Training for Healthcare Providers:
      * Conduct webinars and workshops on app usage for doctors and caregivers.
    - Post-Launch Support:
      * Provide technical support and monitor app performance and user feedback.
  + Deliverables:
    - Live App on App Stores
    - Marketing and Awareness Campaign Report
    - Training Materials for Healthcare Providers
  + Milestone:
    - Successful public launch of DCLD Navigator.

1. **Design and Engineering Standards:**In the development of this application, several design and engineering standards were applied to ensure quality, security, and effective software development processes. These standards align with industry best practices and ensure the app is reliable, secure, and user-friendly.

ISO Standards:

ISO 9001: Quality Management Systems: Ensured that the app development process adhered to established quality management practices, focusing on continuous improvement and delivering high-quality outcomes.

ISO/IEC Standards for Software Development:

* ISO/IEC 27001: Information Security Management Systems: This standard guides the implementation of secure data handling, ensuring the app complies with best practices for protecting sensitive patient information and medical data.
* ISO/IEC 12207: Software Lifecycle Processes: Followed this standard to establish a structured approach to the software development lifecycle, including requirements gathering, design, development, testing, and maintenance.

IEEE Software Engineering Standards:

* IEEE 829: Standard for Software Test Documentation: Used this standard for developing comprehensive test plans and documentation, ensuring the app met functional requirements and user expectations during testing.
* IEEE 1012: Standard for Software Verification and Validation: Applied this standard during the testing phase to validate that the app’s features met defined specifications and requirements, ensuring high-quality performance.

**7. Prototype Description: Prototype Design (Figma or Similar Tool)**

* The initial phase involved creating low-fidelity wireframes for each screen, focusing on the layout and essential features like symptom tracking, medication adherence, and navigation flow.
* Interactive prototypes were built in Figma to simulate user interactions, linking screens and demonstrating key actions such as data entry, alerts, and doctor-patient communication.
* User testing was conducted with a small group of target users, including patients and healthcare providers, to gather feedback on usability, design, and overall navigation.
* Feedback from the user testing phase was analysed and used to refine the design, ensuring the app met the needs of users and improved any areas that caused confusion.
* Figma's collaborative features were used to make quick design adjustments and ensure the app's flow aligned with the project's goals, allowing for smooth iterations and progress.

**7.1 Non-Functional Components and Functional Components (Figma Prototype):  
Functional Components:**

* **Questionnaire Module**
  + Personalized, dynamic questionnaires tailored to patient responses for symptom monitoring.
  + Automated scoring and result summaries to assist doctors in prioritizing interventions.
* **Daily Assessment**
  + Modules for recording vitals, fluid/salt intake, and symptoms with real-time tracking.
  + Push notifications to remind patients to complete daily health assessments.
* **User Profile and Data Management**
  + Detailed, editable profiles with patient history, medical records, and treatment plans.
  + Role-based access control to ensure secure data sharing between patients and doctors.
* **Danger Symptom Notifications**
  + Real-time alerts triggered by critical symptoms to prompt immediate action.
  + Notification logs for patients and doctors to review critical events and responses.
* **Data Privacy and Security**
  + End-to-end encryption and multi-factor authentication to protect user data.
  + Compliance with HIPAA/GDPR standards and anonymization of sensitive information.

**Non-Functional Components:**

1. **Colour Scheme:** A consistent colour palette was used to establish a professional, calming theme suitable for healthcare, with distinct colours for actionable elements (e.g., buttons, alerts).
2. **Typography:** Appropriate fonts were selected for headings and text to maintain clarity and legibility, with clear hierarchy and easy reading on various screen sizes.
3. **Icons:** Visual elements like Search, Notification, and Settings icons were included to enhance the interface's functionality and ease of use.
4. **Design Consistency:** Uniform spacing, alignment, and margins were used across all screens to ensure a clean, organized look and avoid clutter.
5. **Accessibility:** Considerations like high contrast text and backgrounds were implemented to improve readability, especially for users with visual impairments.
6. **Feedback Indicators:** Loading spinners and success messages were incorporated to inform users about the system’s status during actions like saving data or updating health information.
7. **Annotations:** Notes were added in Figma to clarify how interactive elements like buttons and input field’s function, serving as a guide for future development or testing phases.

**7.2 Sample Screenshots of the Prototype:**

* **Fig 1: Doctor’s Dashboard:** Navigate to various pages through dashboardA screenshot of a phone

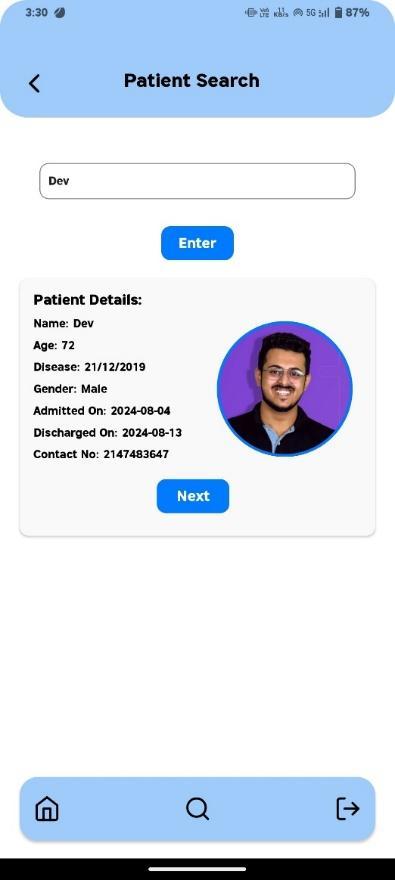
  Description automatically generated
* **Fig 2: Doctor Notifications:** Get warnings if a patient has danger symptom

A screenshot of a phone

Description automatically generated

* **Fig 3: Add Patient Details** : Create patients and their portal credentials A screenshot of a phone

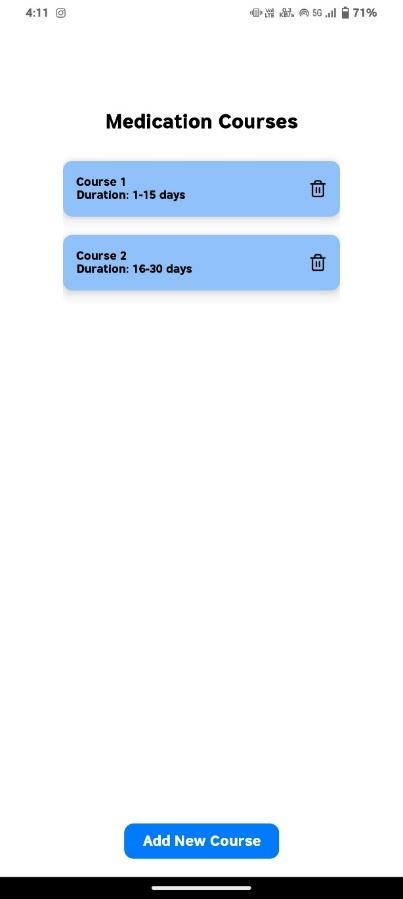
  Description automatically generated

* **Fig 4: Search**: Search for patient records
* **Fig 5&6: Discharge Summaries:** Add discharge summary with all the values from lab reports.

 A screenshot of a test

Description automatically generated

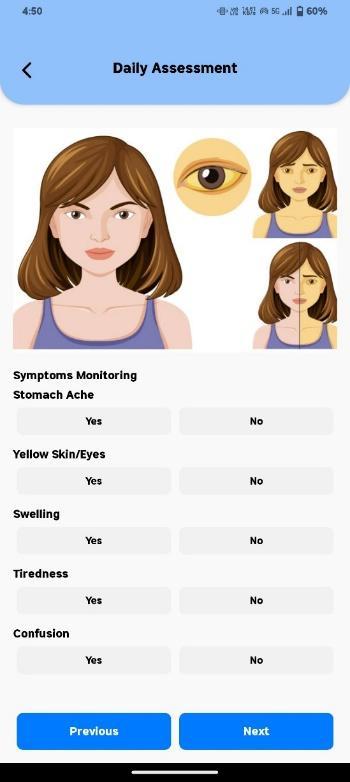
* **Fig 7: Medication Courses:** Add medicines for consecutive courses for every 15 days.



* **Fig 8: Graph Reports:** Track the patients’ progress through graphs with various lab values.



* **Fig 9&10: Questionnaires & Daily Assessments:** Track your daily activities and habits through questions and assessments.



1. **Software and Hardware Specifications:**

* Figma: Used for designing the app's user interface and creating interactive prototypes.
* React Native: The framework used for developing the app, ensuring cross-platform compatibility for Android and iOS.
* XAMPP (Apache, MySQL, PHP): Used for setting up the local development environment and backend server, with MySQL for database management and PHP for creating the API.
* Visual Studio Code: The primary code editor for writing and editing the app’s code in React Native.
* Postman: Used for testing and debugging the API calls between the app and the backend.

**9. Development:  
 Project Setup**

* **Development Platform Selection:**

The app was developed using React Native to ensure cross-platform compatibility for both Android and iOS. This choice allowed for efficient development and deployment on both platforms using a single codebase.

* **Development Environment Configuration:**

The development environment was set up on a Windows PC. Key tools included Visual Studio Code for code editing, XAMPP for the backend, and Android Studio for Android emulator testing. Necessary libraries and dependencies were installed, including React Native, React Navigation, and Axios for API requests.

* **Version Control Repository:**

A GitHub repository was created for version control, ensuring that the code was tracked and changes were documented throughout the development process. Git was used for collaborative management of code (even though developed as a solo project) and easy rollback of changes.

**Backend Development**

**Database Setup:**

The MySQL database was set up using XAMPP as the local development environment. Key data models were defined for user information, symptom tracking, medication logs, and doctor-patient communications.

**API Development:**

PHP was used to create RESTful APIs for data management and application logic. These APIs handled user authentication, patient data management, and communication between the frontend and backend.

**Frontend Development**

UI Design:

The user interface was designed in Figma and later implemented using React Native, ensuring that all design elements were faithfully replicated. Key UI components included login forms, health tracking dashboards, and doctor-patient communication features.

Core Feature Development:

Core features such as navigation, user authentication forms, and interactive elements like buttons and forms were built using React Navigation and React Native Elements. The app allows users to easily track symptoms, medications, and communicate with their healthcare providers.

Responsive Design:

The app was made responsive to fit various screen sizes (mobile and tablet). Using flexbox layouts and media queries, the app adapts to different devices, ensuring an optimized user experience.

**Integration**

The frontend was connected with the backend APIs to ensure seamless communication for user data, symptom tracking, and medication logs. Axios was used to make API requests for user login, data retrieval, and submission.

**9.1 Version History with Screen Images:**

* **Fig 11 & 12: Doctor Dashboard:** Added data download button.

A screenshot of a medical application

Description automatically generated A screenshot of a phone

Description automatically generated

**Version 1.0:** Built core features like Login, Signup, and Home Page with basic navigation; milestone achieved: functional demo.

**Version 2.0:** Enhanced with search filters, dynamic pages, and API integration; milestone achieved: feature-rich app completion.

**Version 3.0:** Optimized performance, added animations, error messages, and accessibility; milestone achieved: release-ready application.Added the download option to download data from the backend in a CSV file.

**10. Testing and Validation:**

Functional testing included unit tests for components like login forms and symptom trackers, and API testing using Postman to ensure proper data exchange. Integration testing verified smooth communication between frontend and backend. Usability testing focused on UI elements, ensuring the design was intuitive and easy to navigate. Beta testing was conducted with real users, gathering feedback on usability and overall user experience. Based on this feedback, adjustments were made to improve functionality and user satisfaction.

**11. Implementation and Deployment**

#### **Implementation**

#### Installed React Native and Visual Studio Code for frontend development, configured XAMPP for the backend, and set up version control using GitHub for code management.

#### Developed the frontend using React Native, integrated backend APIs using PHP and MySQL, and tested core features like login, navigation, and data management for functionality.

#### Configured the MySQL database using XAMPP, ensuring secure data storage and management for user details, symptom tracking, and medication logs.

#### Conducted thorough testing on each module, fixed bugs, and ensured smooth operation across various devices and platforms.

#### Performed User Acceptance Testing (UAT) with real users to ensure the app met the requirements and made final adjustments based on feedback.

#### **Deployment**

* Packaged the app into an APK for Android and set up backend hosting on a cloud server to ensure smooth data access.
* Published the app to Google Play Store, ensuring that backend services were live and accessible for real-time data handling.
* Monitored app performance in the live environment, fixing any emerging issues related to speed or usability.
* Regularly released app updates, adding new features and fixing bugs based on user feedback to maintain a high-quality user experience.

**12. Development Challenges:**

* One major challenge was integrating the frontend with the backend APIs, as there were issues with data formatting and API responses, requiring debugging and refinement.
* Managing state across various screens and components, especially for dynamic data like symptom tracking, was complex and required the implementation of Redux for better state management.
* Ensuring cross-platform compatibility between Android and iOS caused discrepancies in UI alignment and behaviour, requiring adjustments to the layout and style definitions for responsiveness.
* The app initially faced performance issues with slow loading times and laggy UI transitions, which were improved by optimizing code, minimizing re-renders, and implementing lazy loading.
* Securing user authentication and encrypted data storage was challenging, requiring the use of JWT for token-based authentication and ensuring the safe handling of sensitive health information.

**13. Questions Asked by Review Panel:**

* How does the app ensure data privacy and security for sensitive health information?

**Response**: The app uses JWT for token-based authentication and encrypts sensitive data both in transit and at rest, ensuring compliance with privacy regulations.

* How does the app handle real-time updates for patient data and alerts?

**Response**: Real-time updates are managed using API calls, and notifications are delivered via Firebase for immediate alerts on symptoms or medication schedules.

* What steps were taken to ensure the app's accessibility across different device types and screen sizes?

**Response**: The app is designed with a responsive layout, using flexbox to adjust to various screen sizes and testing on multiple devices to ensure uniformity.

* What measures have been taken to ensure the app's scalability as the user base grows?

**Response**: The backend is built using PHP and MySQL, both of which are scalable technologies. The architecture is designed to handle growing data volume and increased API traffic efficiently.

**13.1 Modifications Carried Out:**

* Implemented additional encryption measures for sensitive data storage and strengthened authentication processes to address concerns about privacy and compliance.
* Optimized the alert system by refining Xampp configurations to ensure timely delivery of notifications for critical updates.
* Made adjustments to the user interface for better accessibility and consistency across devices, addressing feedback about alignment and navigation issues.
* Updated backend architecture and database indexing to handle larger datasets and increased API traffic, ensuring smooth performance as the user base grows.

**14. Conclusion and Future Work:**

The project successfully addressed critical healthcare challenges for patients with Decompensated Liver Disease (DCLD) by offering real-time monitoring, personalized care plans, and effective doctor-patient collaboration. Key achievements include a responsive, user-friendly app developed in React Native, secure backend integration with PHP and MySQL, and successful deployment on the Google Play Store. The app has demonstrated potential to improve patient outcomes, reduce hospital readmissions, and enhance quality of life through guided care.

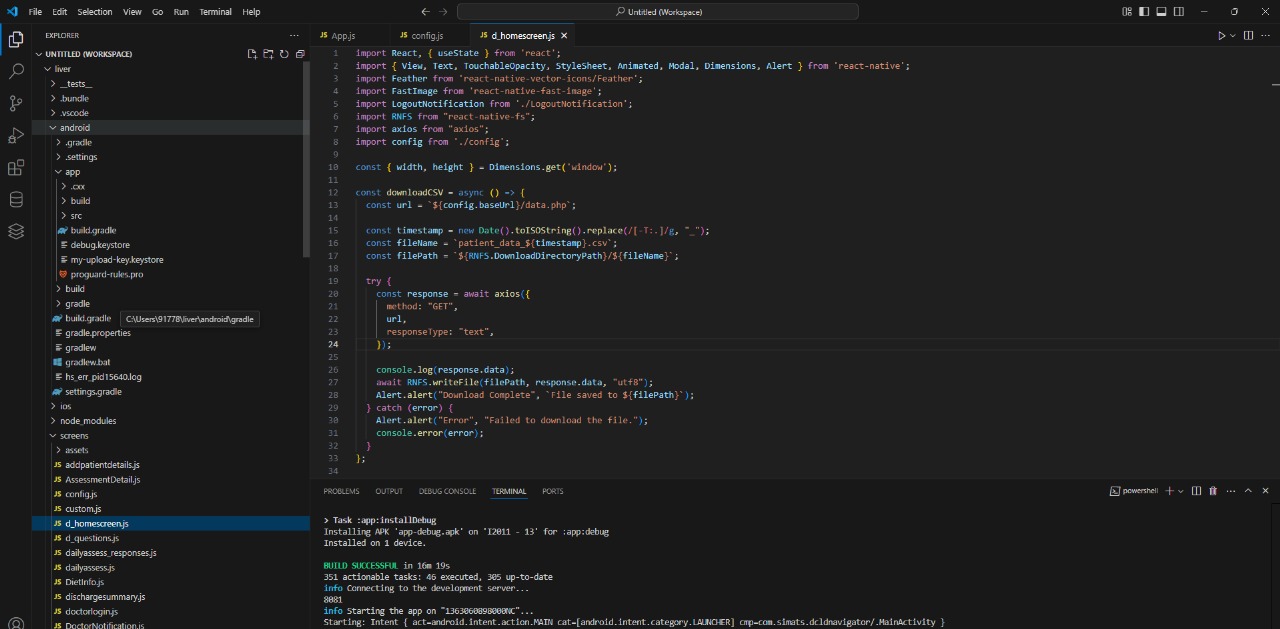
Future enhancements could include integrating AI-based predictive analytics for early warning of complications, expanding multilingual support for diverse user bases, and incorporating wearable device integration for continuous vitals tracking. Regular updates and user feedback will drive further optimization and innovation.

**15. References:**

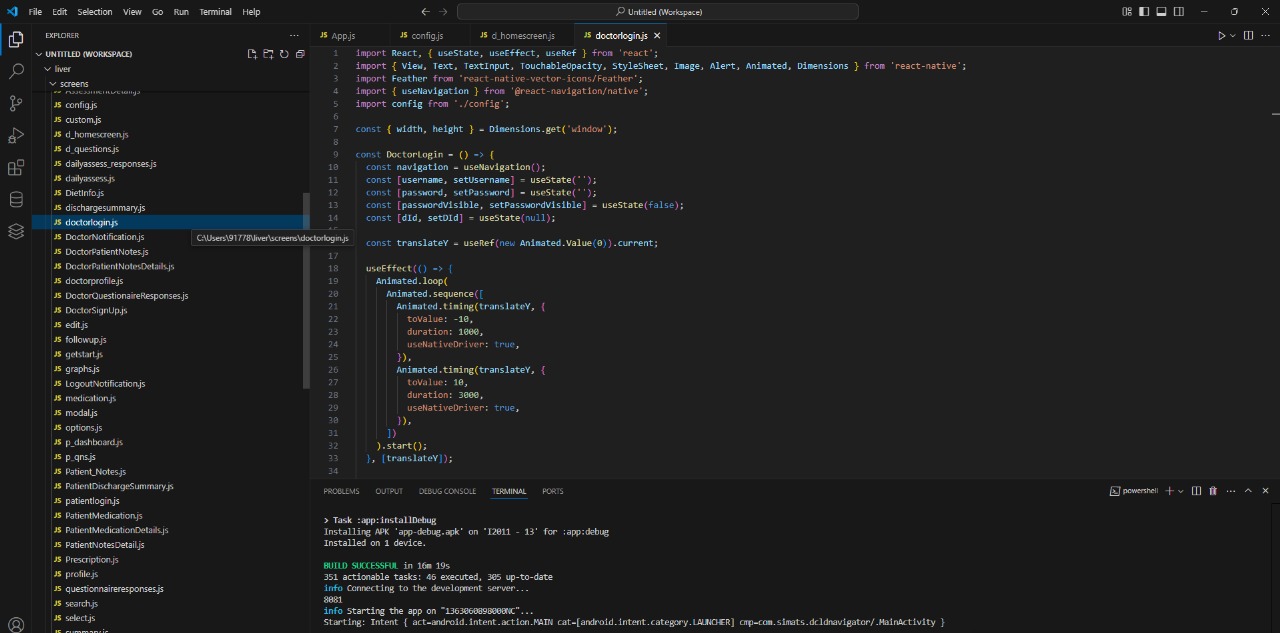
1. **React Native Documentation:** Official documentation for developing cross-platform mobile applications. Available at: https://reactnative.dev/docs
2. **XAMPP Documentation:** Resources for setting up PHP, MySQL, and Apache servers. Available at: https://www.apachefriends.org/docs/
3. **Postman API Testing Tool:** Used for testing and validating API calls. Available at:<https://www.postman.com>
4. **Figma Design Tool:** For prototyping and creating interactive wireframes. Available at:<https://www.figma.com>
5. **ISO/IEC Standards:** References for software development and data security standards, including **ISO/IEC 27001** and **ISO/IEC 12207**.
6. **AsyncStorage Library:** For implementing offline functionality in React Native. Available at: https://react-native-async-storage.github.io/async-storage/
7. **GitHub Repository:** Version control system used for managing code. Available at:<https://github.com>
8. **Various Online Tutorials and Forums:** Stack Overflow, Medium, and Dev.to for troubleshooting coding challenges and optimizing app performance.
9. **Academic Papers and Clinical Studies:** References for understanding DCLD and its care requirements, accessed from PubMed and other reliable healthcare journals.

**16. Appendix:**

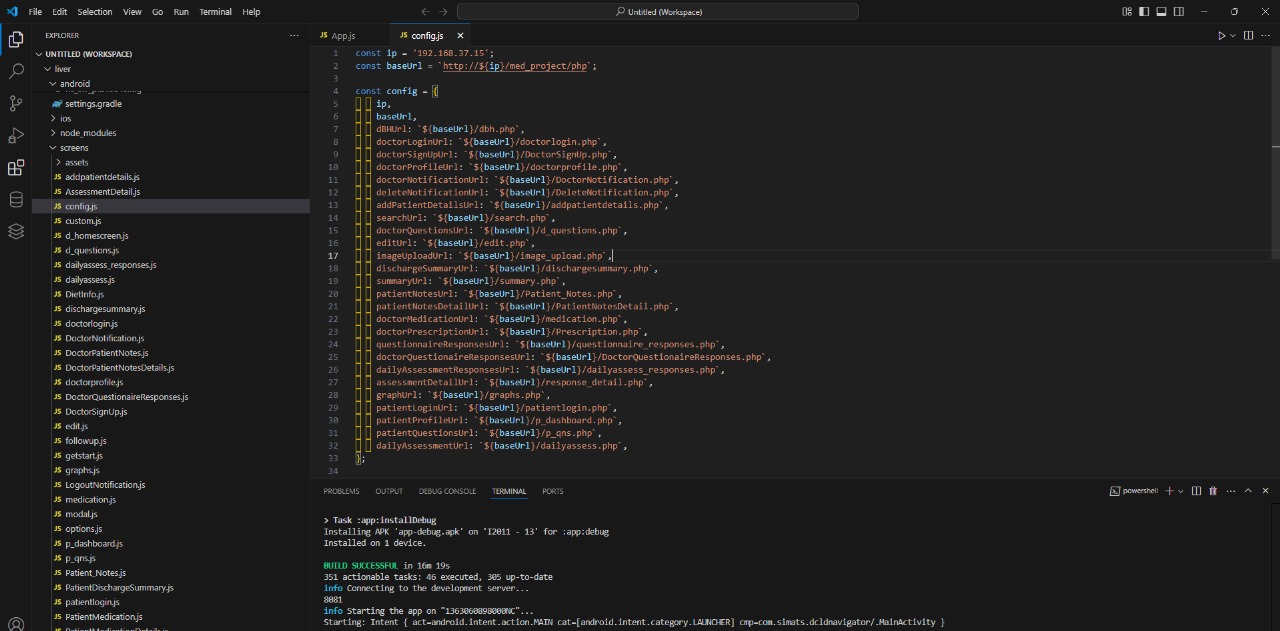
**Fig 13:** Coding Snippet of Doctor Dashboard.



**Fig 14:** Coding snippet of doctor login.



**Fig 15:** Coding snippet of config (common IP address declaration for all APIs)



**Fig 16:** Coding Snippet of daily assessment screen

