



FarmwareApp

Final Year Project Report

Submitted by

Maryam Nadeem
BSCS/S21/2437

Muhammad Mujtaba
BSCS/S21/1985

Hamza Bin Asif
BSCS/S21/1961

Supervisor

Faheem Ahmed

In partial fulfilment of the requirements for the degree of
Bachelor of Science in Computer Science
2025

Faculty of Engineering Sciences and Technology

Hamdard University, Main Campus, Karachi, Pakistan

Certificate of Approval



Faculty of Engineering Sciences and Technology

Hamdard Institute of Engineering and Technology
Hamdard University, Karachi, Pakistan

This project “ **Farmware App** ” is presented by Maryam Nadeem, Muhammad Mujtaba, Hamza Bin Asif under the supervision of their project advisor and approved by the project examination committee, and acknowledged by the Hamdard Institute of Engineering and Technology, in the fulfillment of the requirements for the Bachelor degree of Computer Science.

Mr. Faheem Ahmed
(Project Supervisor)

In-charge FYP-Committee

Chairman

(Dean, FEST)

Authors' Declaration

We declare that this project report was carried out in accordance with the rules and regulations of Hamdard University. The work is original except where indicated by special references in the text and no part of the report has been submitted for any other degree. The report has not been presented to any other University for examination.

Dated:

Authors Signatures:

Maryam Nadeem

Muhammad Mujtaba

Hamza Bin Asif

Plagiarism Undertaking

We, Maryam Nadeem, Muhammad Mujtaba and Hamza Bin Asif, solemnly declare that the work presented in the Final Year Project Report titled **Farmware App** has been carried out solely by ourselves with no significant help from any other person except few of those which are duly acknowledged. We confirm that no portion of our report has been plagiarized and any material used in the report from other sources is properly referenced.

Dated:

Authors Signatures:

Maryam Nadeem

Muhammad Mujtaba

Hamza Bin Asif

Acknowledgments

We extend our heartfelt gratitude to Mr. Faheem Ahmed, our supervisor and mentor, for his unwavering support, valuable guidance, and constant encouragement throughout the course of this project. His mentorship exceeded traditional supervision, equipping us with the knowledge and confidence needed to overcome every challenge we faced.

We also sincerely thank the Faculty of Engineering Sciences and Technology at Hamdard University for providing the necessary resources, a nurturing academic environment, and continuous support that allowed us to bring our ideas to life. Lastly, we are deeply thankful to our families and peers for their steadfast encouragement and moral support during this entire journey.

Document Information

Customer	
Project Title	Farmware App
Document	Final Year Project Report
Document Version	2.0
Identifier	Final Report-1
Status	Final
Author(s)	Maryam Nadeem, Muhammad Mujtaba, Hamza Bin Asif
Approver(s)	Faheem Ahmed
Issue Date	

Definition of Terms, Acronyms, and Abbreviations

EID	Electronic Identification
RFID	Radio Frequency Identification
FYP	Final Year Project
FEST	Faculty of Engineering Sciences and Technology
SQL	Structured Query Language
API	Application Programming Interface

Abstract

Livestock health management plays a vital role in the success of farming, yet traditional methods often rely on manual processes that are time-consuming and prone to delays. To address these challenges, we are developing the Farmware App, a mobile application designed to simplify and enhance the way farmers manage their livestock's health.

Farmware offers features such as easy data entry, and organized digital record-keeping, enabling farmers to track and maintain animal health more efficiently. By providing timely insights and supporting early detection of potential issues, the app helps ensure healthier animals, improves productivity, and reduces the stress involved in livestock care. Farmware aims to bring together technology and farming to offer a smart, reliable, and user-friendly solution for modern livestock management.

Keywords:

Livestock health management, Mobile application, Farmware App, Smart farming, Animal health tracking, Digital record-keeping, Personalized health assessment, Agricultural technology, Farming productivity, Livestock care solutions.

Table Of Contents

Section	Page
Certificate of Approval	1
Authors' Declaration	2
Acknowledgments	4
Document Information	5
Abstract	6
List of Tables	10
CHAPTER 1: INTRODUCTION	2
1.1 Motivation	2
1.2 Problem Statement	2
1.3 Goals and Objectives	3
1.4 Project Scope	3
CHAPTER 2: RELEVANT BACKGROUND & DEFINITIONS	4
2.1 Allflex Livestock Intelligence Overview	4
2.2 Current Farming System Challenges	4
2.3 Need for a Digital Solution	5
2.4 Technological Context	5
CHAPTER 3: RELATED WORK & COMPETITIVE ANALYSIS	7
3.1 Related Works	7
3.2 Competitive Analysis	8
CHAPTER 4: METHODOLOGY & TOOLS	9
4.1 Software Engineering Methodology	9
4.2 Project Methodology	9
4.3 Phases of Project	10
4.4 Software/Tools Used	10
4.5 Hardware Used	10
CHAPTER 5: SYSTEM DESIGN & TESTING	11
5.1.1 Proposed System Architecture/Design	11
5.1.2 Functional Specifications	11
5.1.3 Non-Functional Specifications	12
5.1.4 Testing	12
5.1.5 Purpose of Testing	13
Test Beds	13
Test Cases	13
Test Case ID: TC-001	13
Test Case ID: TC-002	14
Test Case ID: TC-003	14
Test Case ID: TC-004	14
Test Case ID: TC-005	14
Test Case ID: TC-006	14

Section	Page
Test Case ID: TC-007	15
Test Case ID: TC-008	15
Test Case ID: TC-009	15
Test Case ID: TC-010	15
Test Case ID: TC-011	15
CHAPTER 6: EXPERIMENTAL EVALUATIONS & RESULTS	17
Components of the Evaluation Testbed	17
Test Scenarios	17
Evaluation Outcomes	18
Results and Discussion	18
Results	18
Discussion	19
CHAPTER 7: CONCLUSION	20
Strengths of the Project	20
Limitations and Future Work	20
Limitations	20
Future Work	20
Reasons for Failure – If Any	21
Conclusion	21
APPENDICES	
A1A. Project Proposal and Vision Document	30
A1B. Copy of Proposal Evaluation Comments by Jury	30
A2. Requirement Specifications	39
A3. Design Specifications	39
A4. Other Technical Detail Documents	53
Test Cases Document	6
UI/UX Detail Document	15
Coding Standards Document	17
Project Policy Document	18
User Manual Document	18
A5. Flyer & Poster Design	7
A6. Copy of Evaluation Comments	8
A7. Meetings' Minutes & Sign-Off Sheet	99
P-test Report Results	101

Section

Page



List of Tables

Table	Page
TABLE 5.1: TEST Scenario 1 (Login & Signup - Farmer Account Creation and Authentication)	95
TC-01	95
TC-02	95
TC-03	95
TC-04	95
TC-05	95
TC-06	95
TC-07	95
TABLE 5.2: TEST Scenario 2 (Adding New Animal)	96
TC-08	96
TC-09	96
TC-10	96
TC-11	96
TC-12	96
TC-13	96
TABLE 5.3: TEST Scenario 3 (Symptoms Selection)	97
TC-14	97
TC-15	97
TC-16	97
TC-17	97
TC-18	97
TC-19	97
TABLE 5.4: TEST Scenario 4 (Disease Prediction Based on Symptoms)	98
TC-01	98
TC-02	98
TC-03	98
TC-04	98
TC-05	98
TABLE 5.5: TEST Scenario 5 (Health Recommendations Display)	99
TC-01	99
TC-02	99
TC-03	99
TABLE 5.6: TEST Scenario 6 (Cow Health Profile)	100
TC-01	100
TC-02	100

Table	Page
TC-03	100
TC-04	100
TC-05	100
TC-07	100
TC-08	100
TABLE 5.7: TEST Scenario 7 (Health Report Generation)	101
TC-01	101
TC-02	101
TC-03	101
TC-04	101
TC-05	101
TC-06	101
TC-07	101
TC-08	101
TABLE 5.8: TEST Scenario 8 (Connectivity Handling)	103
TC-01	103
TC-02	103
TC-03	103
TC-04	103
TC-05	103
TC-06	103
TC-07	103
TC-08	103
TC-09	103
TC-10	103

CHAPTER 1

INTRODUCTION

Taking care of livestock is at the heart of successful farming, but managing their health can be a complex and time-consuming task. Traditional methods often involve a lot of manual work, from keeping track of health records to identifying potential illnesses, which can lead to delays and missed opportunities for timely care. These challenges can make it harder for farmers to ensure the well-being of their animals.

To make this process simpler and more efficient, we are creating a mobile application that's designed to support farmers in managing their livestock's health with ease. This app brings technology and farming together, offering features like easy data entry, and organized record-keeping.

With this application, farmers will have a tool that not only saves time but also provides valuable insights to help them make better decisions. By simplifying health tracking and offering smarter ways to identify potential issues early, this app aims to ensure healthier animals, improve productivity, and make farming a little less stressful. Our goal is to give farmers the support they need to care for their livestock in a way that's simple, effective, and reliable.

1.1 Motivation

Livestock health is a cornerstone of agricultural economies, particularly in regions where dairy and meat production are integral to livelihoods. The lack of efficient disease management in cows often leads to substantial financial losses and impacts food security. These challenges motivated the development of an innovative solution to proactively address cow health issues. Our application aims to empower farmers and veterinary professionals with the tools needed to predict diseases and take timely actions.

1.2 Problem Statement

The absence of a robust, accessible, and predictive solution for livestock health monitoring has left many farmers relying on traditional methods. These methods are often reactive, addressing diseases only after symptoms escalate, which reduces the likelihood of effective treatment and increases operational costs. There is a critical need for a system that predicts diseases early, enabling farmers to act swiftly and minimize risks.

1.3 Goals and Objectives

1. Develop an Intelligent Mobile Application

A mobile-based solution that predicts cow diseases using intelligent algorithms. Farmers toggle observed symptoms, and the app provides accurate predictions and care recommendations.

2. Offer a User-Friendly Experience

Design an intuitive and accessible interface that allows farmers to effortlessly input data, navigate features, and understand results without requiring technical expertise. The app will simplify complex processes and ensure smooth usability for all users.

3. Enable Early Disease Detection and Prevention

Equip farmers with a reliable tool for identifying potential health issues in cows at an early stage. By facilitating timely interventions, the app will help reduce the risk of disease progression and safeguard livestock health.

4. Minimize Financial Losses for Farmers

Support farmers in avoiding unnecessary expenses by providing early warnings and actionable insights. This will not only prevent costly treatments and loss of livestock but also improve overall farm productivity and profitability.

1.4 Project Scope

This project aims to develop a mobile application designed to streamline livestock health management by predicting diseases based on symptoms entered by users. The app will provide timely notifications and alerts to inform farmers about potential health risks, enabling early intervention and prevention. It will also include robust database storage to ensure secure and efficient data management, allowing farmers to maintain detailed health records for their livestock. With a simple and intuitive design tailored for non-technical users, the application ensures ease of use, making advanced health monitoring accessible to all farmers.

CHAPTER 2

RELEVANT BACKGROUND & DEFINITIONS

2.1 Allflex Livestock Intelligence Overview

Allflex Livestock Intelligence is a leading livestock management app that integrates advanced technology to offer comprehensive monitoring and management solutions for farmers. The platform utilizes sophisticated sensors, RFID tags, and EID (Electronic Identification) systems to track individual animals in real time. These tools monitor a variety of health indicators, such as temperature, rumination, and activity levels, providing farmers with precise data about their livestock's condition. By continuously collecting data from wearable sensors and other monitoring devices, Allflex enables farmers to detect health issues early, improve breeding management, and optimize overall herd productivity. The app's seamless integration of real-time data supports proactive decision-making, helping farmers to enhance animal welfare and farm performance. Allflex is renowned for its innovation in the use of technology to streamline livestock management and improve the efficiency of farming operations.

2.2 Current Farming System Challenges

- **High Costs:** The use of RFID tags, EID systems, and wearable sensors can be expensive, making the technology less accessible to small and medium-sized farms. The initial investment required for setting up the system can be a significant financial burden for farmers with limited budgets.
- **Complex Setup and Maintenance:** Setting up and maintaining the advanced sensors and monitoring equipment requires technical expertise. Farmers without a strong background in technology may face challenges in configuring and troubleshooting the system, potentially leading to underutilization of the platform.
- **Data Overload:** While real-time data collection is beneficial, the volume of information generated by sensors can be overwhelming. Farmers may struggle to filter out irrelevant data and focus on the most important health indicators, leading to decision fatigue or missed insights.
- **Dependence on Technology:** The system's reliance on technology means that any technical failure, such as sensor malfunction or connectivity issues, can disrupt the entire monitoring process. This can lead to gaps in data or delays in detecting health issues.

2.3 Need for a Digital Solution

In light of the challenges faced by farmers in managing livestock health, there is a growing need for a digital solution that can streamline and enhance the management process in an affordable way. A comprehensive mobile application can provide the following benefits:

Efficiency: Automating health monitoring, data entry, and disease prediction processes to save time and reduce the likelihood of errors. This enables farmers to quickly track the health of their livestock without extensive manual input, reducing operational costs.

Data-Driven Insights: Providing farmers with valuable insights and analytics to make informed decisions about their livestock's health. By analyzing health data, symptoms, and trends, farmers can proactively address potential health risks, leading to improved herd productivity and reduced financial losses.

Accessibility: Ensuring that farmers, even those with minimal technical expertise, can easily use the app. The app's user-friendly design will make it accessible on mobile devices, giving farmers the ability to monitor their livestock's health on-the-go, whether in the field or at home.

Affordable: The app is designed to offer essential features at a cost-effective price point, making it accessible to farmers of all sizes. By eliminating the need for expensive sensors and advanced tools, it provides an affordable solution to livestock health management without compromising on quality.

2.4 Technological Context

The Farmware mobile application will utilize modern web and mobile technologies to ensure a seamless and efficient user experience. The key technologies include:

Frontend: Ionic framework with Angular, leveraging TypeScript, HTML, and SCSS to create a responsive and dynamic mobile application interface, providing smooth interactions and accessibility for farmers.

Backend: API/ Django/ MySql

Database: MySQL database for structured and efficient data storage, enabling the system to handle large volumes of livestock health data while maintaining fast retrieval and secure management of records.

Authentication: Secure user authentication and authorization mechanisms to protect user data and ensure only authorized access to the system, safeguarding sensitive livestock health information.

CHAPTER 3

RELATED WORK & COMPETITIVE ANALYSIS

3.1 Related Works

Allflex Livestock Intelligence

URL: <https://www.allflex.global/>

Allflex Livestock Intelligence is a comprehensive platform designed to help farmers monitor and manage the health and performance of their livestock. The platform integrates advanced technologies such as RFID tags, EID (Electronic Identification), and wearable sensors to provide real-time data on animals' behavior, health metrics, and location. This data helps farmers make informed decisions regarding their livestock's care and management.

Allflex offers a wide range of tools for tracking the health and productivity of animals, including data on behavior, feeding patterns, reproductive status, and more. The platform's real-time monitoring allows for early detection of health issues, enabling timely intervention. It also provides predictive analytics to identify potential problems before they escalate, thus improving the overall management and profitability of farms.

Targeted primarily at large-scale livestock operations, Allflex is designed to integrate seamlessly with farm management systems, making it easier for farmers to manage large herds. The system is also beneficial for farmers looking to optimize productivity and ensure the well-being of their animals, offering data-driven insights that improve decision-making and operational efficiency. Allflex is a valuable tool for those seeking a comprehensive, technology-driven approach to livestock management.

Cattlytics

<https://www.cattlytics.com/>

Cattlytics is a mobile application designed to streamline the health management and performance tracking of livestock, specifically cattle. By combining data analytics and predictive modeling, Cattlytics helps farmers monitor the health, behavior, and productivity of their herds with ease. The app collects valuable data on individual animals, such as weight, feeding habits, and health metrics, and uses this information to provide insights into potential health risks or productivity issues.

Cattlytics offers farmers an intuitive platform for tracking their livestock's daily activities and overall well-being. The app's data analysis capabilities allow for early detection of health problems, making it easier to implement preventative measures before issues escalate. In addition to health monitoring, Cattlytics also provides valuable insights into breeding, growth patterns, and feeding strategies to optimize herd management and farm productivity.

Primarily targeted at small to medium-sized farms, Cattlytics offers an affordable and accessible solution for livestock management. The user-friendly interface is designed for farmers without

technical expertise, making it easy to input data and interpret the results. By providing actionable insights through predictive analytics, Cattlytics aims to improve livestock health, increase farm profitability, and simplify the overall management process for farmers.

Cow Manager

URL: <https://www.cowmanager.com/>

CowManager is a livestock management platform that uses sensor technology to monitor the health and productivity of dairy cows. It tracks vital indicators like body temperature, activity, and rumination, providing real-time data to detect early signs of health or reproductive issues. The system sends alerts to farmers, enabling quick intervention to improve herd health and milk production. With data analytics and reports, CowManager helps optimize herd management and farm productivity. It is designed for both small and large dairy farms, offering a user-friendly interface for effective, data-driven decision-making.

3.2 Competitive Analysis

Application	Offerings	Features	Target Audience	Usability
Allflex	Livestock health monitoring, performance tracking	RFID tags, EID, wearable sensors, real-time health monitoring, predictive analytics	Large-scale livestock farmers	High-tech, requires integration with existing farm management systems, suitable for large farms
Cattlytics	Livestock health and performance management	Behavior monitoring, health tracking, predictive analytics, individual animal tracking	Small to medium-sized farms	User-friendly interface, affordable solution for smaller farms, easy data input and analysis
CowManager	Dairy cow health and productivity management	Real-time monitoring, activity tracking, body temperature monitoring, alerts for health/reproductive issues	Dairy farmers	Real-time alerts, datadriven decision-making, user-friendly for dairy farm operations
Farmware	Predictive disease detection and health management for cows	Symptom toggling, real-time disease prediction, health record storage, alerts, personalized care recommendations	Small to large-scale farmers, veterinary professionals	Easy-to-use, intuitive interface, affordable, predictive analytics, early disease detection, personalized advice, adaptable to user needs

CHAPTER 4

PROJECT DISCUSSION

1. Software Engineering Methodology

Hybrid Methodology: For the development of the *Farmware App*, a hybrid software engineering methodology was adopted, combining elements of both Agile and Waterfall models to suit the unique demands of the project. The app involves both backend logic for disease prediction and frontend usability tailored for farmers, requiring a flexible yet structured approach.

- **Agile Practices**

Agile principles were applied through iterative development, enabling continuous testing and feedback from farmers. After each phase, the app was evaluated, and insights shaped the next iteration. Fortnightly meetings with the supervisor ensured ongoing review and refinement. This approach was crucial when redesigning the symptom selection feature, replacing the original questionnaire system to enhance user-friendliness and better reflect real-world needs.

- **Waterfall Elements**

Waterfall methods were applied during the **initial planning and design phases**, where the app's architecture, database structure, and disease prediction logic were clearly defined. This structured approach ensured a solid foundation, especially for the symptom-disease mapping system and the storage of historical disease records.

2. Project Methodology

- The development of the **Farmware App** was based on an **iterative and incremental approach**. The team began with a clearly defined planning and system design phase, where core functionalities, such as symptom selection, disease prediction, and health record storage, were structured in detail.
- After this initial setup, the app was developed in **cycles**, with each phase building on the previous one. Features were implemented step-by-step, tested, and improved based on **fortnightly feedback sessions with the supervisor** and hands-on testing by farmers. This helped refine the user interface, improve logic accuracy, and enhance usability, particularly leading to the removal of the questionnaire system in favor of direct symptom selection for easier use.

3. Phases of Project

- **Phase 1: Planning and Requirements**
Defined project goals and gathered input from farmers and supervisor. Decided to replace the questionnaire with direct symptom selection for simplicity.
- **Phase 2: System Design**
Designed the database and system architecture for user, cow, symptom, and disease management.
- **Phase 3: Development**
Implemented user management, symptom selection, disease prediction, and health record saving.
- **Phase 4: Testing**
Conducted unit testing and gathered feedback from farmers and the supervisor to refine usability and fix issues..
- **Phase 5: Deployment and Maintenance**
Deployed the app, monitored performance, and planned regular updates based on user feedback.
- .

4. Software/Tools that Used in Project

- **Frontend:** Ionic Angular
- **Backend:** Python
- **Database:** MySQL
- **Machine Learning:** Scikit-learn, TensorFlow
- **APIs:** RESTful APIs

5. Hardware that Used in Project

1. **Development and Testing:** Standard desktop or laptop computers with at least 8 GB of RAM, 512 GB of hard drive storage, and sufficient processing power to handle development environments and testing suites.
2. **Deployment:** Local servers or cloud-based servers (e.g., AWS EC2 instances) for hosting the web application.
3. **Usage:** The app runs on smartphones and tablets with Android OS, requiring at least 2 GB RAM and Android version 6.0 (Marshmallow) or higher.

Chapter 5

IMPLEMENTATION

5.1.1 Proposed System Architecture/Design

Farmware follows a three-tier architecture:

1. **Presentation Tier:**
Built using Ionic Angular, this tier provides a mobile-friendly interface for farmers to input cow health data and view predictions. It ensures usability for users with minimal technical skills.
2. **Application Tier:**
Developed using Node.js, this tier handles core logic, including symptom processing, disease prediction, and recommendation generation through RESTful APIs.
3. **Data Tier:**
MySQL is used to store user data, symptom records, disease predictions, and health history securely and efficiently.

5.1.2 Functional Specifications

Core Functionalities:

1. **User Management:**
The system allows farmers to create accounts, log in securely, and manage their profiles. Each user has access to their own animals' health data.
2. **Symptom Selection & Health Data Entry:**
Farmers can select symptoms from a predefined list each day to report the condition of their cows. This data is saved under each cow's profile.
3. **Disease Prediction:**
Based on the selected symptoms, the system suggests possible diseases using predefined logic or simple rule-based matching. Predictions are displayed immediately after data submission.
4. **Health Recommendations:**
After predicting the disease, the system shows relevant recommendations and care tips that farmers can follow to manage or prevent worsening conditions.
5. **Health Records & History Tracking:**
Each cow has a dedicated profile storing its daily health data and disease history, allowing farmers to monitor trends over time.
6. **Data Export:**
Farmers can export individual cow health records to maintain offline copies or share with veterinary experts.

5.1.3 Non-Functional Specifications

Performance Requirements

- The system handles daily health data input for multiple cows without performance issues.
- Disease predictions are generated within 5 seconds after submission.
- The platform supports simultaneous usage by up to 1,000 farmers with no noticeable latency.

2. Safety Requirements

- All user and animal data is stored securely with regular automated backups.
- The system complies with relevant data protection laws to maintain user privacy.
- Error handling features prevent data loss and ensure automatic data recovery during system failures.

3. Security Requirements

- Sensitive data is encrypted both at rest and during transmission using industry-standard encryption of sql.

4. Reliability Requirements

- It is scalable and maintains performance even as the number of users or cows increases.

5. Usability Requirements

- The user interface is simple and intuitive for farmers with limited technical knowledge.
- The application runs smoothly on mobile.
- It provides real-time feedback on health entries with clear and understandable recommendations.

7. User Documentation

- The system includes a detailed user manual covering health data entry, prediction interpretation, and following recommendations.
- It also provides FAQs, and step-by-step instructions.

5.1.4 Testing

Testing involves unit, integration, and system-level assessments:

1. **System Testing:**
 - End-to-end validation of functionalities under expected workloads.

2. User Acceptance Testing (UAT):

- Ensuring the application meets user needs through direct feedback from stakeholders.

5.1.5 Purpose of Testing

The purpose of testing is to ensure Farmware App meets functional and non-functional requirements. Testing validates that the system operates as intended, is secure, user-friendly, and performs well under various scenarios.

Specific objectives include:

1. Identifying and fixing bugs.
2. Validating system functionalities against requirements.
3. Ensuring data integrity and security.
4. Testing user interactions for a seamless experience.
5. Ensuring compatibility across devices and platforms.

Test Beds

- **User Access Management**
Tests registration, login, and secure authentication processes for farmers, admins, and veterinary experts.
- **Health Data Entry**
Verifies the accurate input, submission, and storage of daily cow health data.
- **Disease Prediction Engine**
Tests the accuracy, response time (within 5 seconds), and consistency of disease predictions based on submitted data.
- **Health Report Export**
Verifies the export functionality for reports in CSV and PDF formats.
- **System Performance**
Measures application speed, stability, and data flow during high-usage conditions (e.g., up to 1,000 concurrent users).
- **Cross-Platform Compatibility**
Ensures the app functions smoothly on Android devices (\geq Android 6.0, \geq 2 GB RAM) and across various screen sizes.

Test Cases

Test Case ID: TC-001

Title: User Login with Valid Credentials

Objective: Verify that users can log in using valid credentials.

Steps:

1. Open the android application.
2. Enter a valid email and password.
3. Click the "Login" button.

Expected Result: User successfully logs in and is redirected to the home screen.

Test Case ID: TC-002

Title: User Login with Invalid Credentials

Objective: Ensure login fails with incorrect credentials.

Steps:

1. Open the Android application.
2. Enter an invalid email or incorrect password.
3. Click the "Login" button.

Expected Result: User sees an error message and is not allowed to log in.

Test Case ID: TC-003

Title: Disease Prediction Accuracy

Objective: Ensure disease predictions are generated based on entered health data.

Steps:

1. Log in and enter symptom data for a cow.
2. Submit the data.
3. View the disease prediction result.

Expected Result: Disease prediction is displayed within 5 seconds.

Test Case ID: TC-004

Title: View Health History

Objective: Verify that the farmer can view a cow's historical health records.

Steps:

1. Log in.
2. Go to a cow's profile.
3. Click on "Health History."

Expected Result: A timeline of previous health data entries is displayed.

Test Case ID: TC-005

Title: Export Health Report

Objective: Verify farmers can export cow health data in PDF or CSV format.

Steps:

1. Go to a cow's profile.
2. Click "Export."
3. Choose PDF or CSV.

Expected Result: Download of the selected format is initiated.

Test Case ID: TC-006

Title: Secure Data Transmission

Objective: Ensure all data entered is securely transmitted.

Steps:

1. Monitor network requests during login and data entry.

Expected Result: Data that is transmitted is encrypted.

Test Case ID: TC-007

Title: Multi-User Performance Test

Objective: Test the system's performance under concurrent user load.

Steps:

1. Simulate 10 users submitting data at the same time.

Expected Result: No crashes or major slowdowns; responses within 5 seconds.

Test Case ID: TC-008

Title: Add New Cow Entry

Objective: Verify that a user can successfully add a cow.

Steps:

1. Log in to the app.
2. Click the "+" button on the home screen.
3. Fill in the cow form with valid data.
4. Click "Save."

Expected Result: The cow is added and appears on the home screen with its ID.

Test Case ID: TC-009

Title: Generate Report Based on Selected Symptoms

Objective: Confirm that selecting symptoms and submitting generates a report.

Steps:

1. Navigate to the symptoms selection screen.
2. Select multiple symptoms using toggle buttons.
3. Click "Submit" or "Generate Report."

Expected Result: A report is generated and the user is redirected to the cow's medical records page.

Test Case ID: TC-010

Title: Sidebar Navigation

Objective: Verify that sidebar options navigate correctly.

Steps:

1. Open the sidebar menu from the home screen.
2. Click each option (Training, Guide, About Us, Help, etc.) one by one.

Expected Result: Each click opens the corresponding page correctly

Test Case ID: TC-011

Title: Logout Functionality

Objective: Verify that the user can log out successfully.

Steps:

1. Open the sidebar.
2. Click on "Logout."

Expected Result: User is logged out and redirected to the login screen. Click on "Logout."

Expected Result: User is logged out and redirected to the login screen.

Chapter 6

EXPERIMENTAL EVALUATIONS & RESULTS

The evaluation testbed for Farmware serves as a controlled environment to verify the application's functionality, performance, reliability, and compatibility under realistic usage conditions.

6.1 Components of the Evaluation Testbed

The evaluation testbed for CIEET includes the following components:

- **Database Management:**
The app uses a cloud-hosted database (MySQL) to manage structured data such as user accounts, cow profiles, and health records. Testing focused on data accuracy, fast access, and consistency across sessions
- **Mobile Device Compatibility:**
Farmware was tested on Android smartphones and tablets running **Android 6.0 (Marshmallow)** or higher, with at least **2 GB RAM**. This ensured the app's performance and UI remained smooth and responsive on lower-end devices.
- **Web Browsers:** In cases where the backend or admin interface is accessed via browser, compatibility with modern browsers such as **Google Chrome**.

6.2 Test Scenarios

The evaluation testbed for **Farmware** included the following real-world scenarios to ensure robustness and reliability:

- **Simultaneous Access by Multiple Users:**
Multiple farmers accessed the application concurrently to add cow profiles, select symptoms, and generate health reports. This tested the system's performance and responsiveness under load.
- **Database Operations:**
Data operations such as creating, retrieving, updating, and deleting cow records and health reports were validated to ensure consistency, accuracy, and minimal latency.
- **Cow Profile Management:**
Farmers added cow details through the input form. Test cases covered valid/invalid inputs, field validations, and successful data storage with correct display on the home screen.
- **Symptom Selection and Report Generation:**
The symptom toggle interface was tested by selecting various combinations of symptoms to ensure accurate disease prediction and smooth report generation without errors.
- **Medical Records Access:**
After report generation, users navigated to view medical records. The test validated that

data matched selected symptoms and predictions and displayed correctly with historical consistency.

- **Sidebar Navigation and App Pages:**

Testers interacted with the sidebar to navigate to pages like App Training, Guide, Settings, About Us, and Help. This ensured seamless transitions and functionality across all screens.

6.3 Evaluation Outcomes

The evaluation testbed provided the following insights for the **Farmware** application:

- **Database Performance:**

The system efficiently managed concurrent operations such as adding cow profiles, symptom selections, and report generations, with no data loss or noticeable delays.

- **System Robustness:**

Farmware maintained consistent performance and stability under simultaneous usage by multiple users, with no application crashes or critical failures observed.

- **Feature Validation:**

Core functionalities, including cow data entry, symptom selection via toggle buttons, and automatic health report generation, functioned reliably across all tested scenarios.

- **Resilience to Network Disruptions:**

The application handled intermittent network issues gracefully, preserving entered data and maintaining user flow with minimal interruption.

6.4 Results and Discussion

6.4.1 Results

- **User Authentication:**

The system successfully authenticated users during login and registration. Navigation between login and sign-up pages functioned as expected, and valid user sessions were maintained securely.

- **Cow Data Entry and Display:**

Users were able to add cow information through the form interface without errors. Saved entries appeared correctly on the home screen with accurate IDs and associated details.

- **Symptom Selection and Report Generation:**

The toggle-based symptom selection process operated smoothly. The system accurately generated medical reports based on the selected symptoms, concluding the health status without delays.

- **Navigation and UI Flow:**

Navigation across key components—such as Training, Settings, About Us, and Help—was intuitive. Access to cow profile pages and related health data (including picture, ID, gender, and breed) worked reliably.

- **Performance and Scalability:**

The application handled concurrent usage by multiple farmers with no performance degradation. Average symptom processing and report generation time remained under 5 seconds during load testing.

6.4.2 Discussion

- **Strengths:**

The system's minimalistic UI design ensured ease of use, especially for users with limited technical skills. Health data entry and reporting features worked seamlessly, contributing to user satisfaction. The app showed resilience to common usage challenges such as network interruptions.

- **Limitations:**

Occasional UI glitches were observed on low-end Android devices. In rare instances, symptom toggle selections required multiple taps to register.

- **Future Enhancements:**

Planned improvements include offline data caching for remote use, localized language support for rural users, and integration of animal health tips based on historical symptom data.

CHAPTER 7

CONCLUSION

The development and evaluation of the Farmware application have been guided by the objective of simplifying the health monitoring process for livestock, particularly for farmers with limited technical experience. This chapter summarizes the project's strengths, limitations, and future directions based on its design, implementation, and testing outcomes.

7.1 Strengths of the Project

The Farmware App demonstrated several key strengths throughout its development and evaluation phases:

- **Simplified Cow Health Monitoring:** The application enabled farmers to easily input symptoms using toggle buttons, removing the complexity of navigating lengthy questionnaires and making the process more efficient and accessible.
- **Fast and Adaptive Reporting:** Health reports were generated immediately after symptom selection, providing timely feedback and allowing for quicker decision-making regarding animal care.
- **Optimized Mobile Compatibility:** Designed primarily for Android devices, the app offered smooth performance on smartphones and tablets, aligning with the usage patterns of rural farmers.
- **Efficient Data Management:** The structured storage of cow records and medical history ensured easy retrieval and consistent performance, even with multiple entries.
- **User-Centered Design:** The clean interface and simple navigation significantly improved usability for non-technical users, reducing the training time needed to use the system effectively.
- **Stable Performance Under Load:** Testing confirmed that Farmware maintained responsiveness and reliability when used by many users simultaneously, ensuring scalability for larger farming communities.
- **Built-In Educational Resources:** Integrated navigation to training, help, and guide sections within the app empowered users to understand both the app's features and best practices for animal care.

7.2 Limitations and Future Work

While the project achieved its primary objectives, certain limitations were observed:

7.2.1 Limitations:

1. **Dependence on Stable Internet Connectivity:**
Farmware requires continuous internet access for login, cow data entry, and report generation. In rural or remote areas with unreliable connectivity, this may hinder uninterrupted usage.
2. **Limited Offline Capabilities:**
Currently, the system does not support offline symptom recording or report viewing, which restricts access during network outages.
3. **No Real-Time Veterinary Feedback:**
Although the app generates reports based on symptoms, it does not currently integrate live veterinary consultations or real-time expert support.
4. **Basic UI on Low-End Devices:**
On lower-end Android devices, minor layout shifts and slower performance were observed during testing, slightly affecting user experience.

7.2.2 Future Work:

1. **Offline Mode Integration:**
Develop offline functionality for symptom logging and local storage, enabling farmers to use the app in poor connectivity areas, with automatic syncing once online.
2. **Multilingual Support:**
Add regional language options to ensure accessibility for farmers who are not fluent in English, improving usability across diverse regions.
3. **Health Pattern Analytics:**
Implement machine learning-based analytics to identify disease trends, predict future risks, and send early warnings based on historical data.
4. **Expanded Animal Record Types:**
Extend support beyond cows to include goats, buffaloes, and other livestock, allowing broader applicability of the platform.

7.3 Reasons for Failure – If Any

The **Farmware** project did not encounter any major failures during its development and evaluation phases. However, a few **minor challenges** were noted:

- **Symptom Report Processing Delays:**
Initial testing revealed slight delays in generating health reports when multiple symptoms were selected rapidly. This was resolved by optimizing the backend logic for faster processing.
- **UI Performance on Low-End Devices:**
On devices with limited hardware resources, the user interface experienced occasional lags, especially on the symptom selection screen. Refining animations and reducing render complexity improved responsiveness.

7.4 Conclusion

The **Farmware** application marks a significant advancement in simplifying and improving livestock health management for farmers. By eliminating complex workflows and offering a user-friendly interface, the system empowers farmers to easily record animal information and assess health conditions based on symptom selection.

With a solid backend infrastructure and intuitive design tailored for users with limited technical expertise, Farmware offers a **practical, accessible, and scalable solution** for daily livestock care. Its streamlined health reporting, reliable performance, and secure data handling position it as a valuable tool in modern animal husbandry.

Future enhancements—such as offline mode, improved analytics, and broader device compatibility—will continue to strengthen its role in supporting the agricultural community with technology-driven care.

REFERENCES

- **Ionic Framework (Frontend Development)**
Official website of the Ionic framework used to build the frontend.
<https://ionicframework.com/>
- **My Cattle Manager (UI Inspiration)**
Used as a reference for designing user interface elements and layout.
<https://www.bivatec.com/apps/my-cattle-manager>
- **Modular Design Guide (Design Principles)**
Resource explaining modular design concepts for scalable application structure.
<https://denovers.com/blog/what-is-modular-design/>
- **HerdDogg (Livestock Monitoring System)**
Commercial livestock tracking solution used for industry inspiration.
<https://herddogg.com/>
- **CowManager (Animal Monitoring Technology)**
Reference for advanced cow health and behavior monitoring technologies.
<https://www.cowmanager.com/>
- **HerdWatch (Farm Management App)**
Comprehensive farm management tool offering real-world design insights.
<https://herdwatch.com/>
- **Allflex Global (Animal Identification Solutions)**
Provider of livestock identification technologies, offering reference on animal data management.
<https://www.allflex.global/>
- **Ionic + Angular App Tutorial (YouTube)**
Step-by-step tutorial on building apps with Ionic and Angular.
<https://www.youtube.com/watch?v=r2ga-iXS5i4>
- **Database Design Basics (Microsoft)**
A beginner-friendly guide to designing relational databases.
<https://learn.microsoft.com/en-us/office/troubleshoot/access/database-design-basics>

1. APPENDICES

List of Appendices

A0. Copy of Project Registration Form

A1a. Project Proposal and Vision Document

A1b. Copy of Proposal Evaluation Comments by Jury

A2. Requirement Specifications

A3. Design Specifications

A4. Other Technical Details

Test cases

UI/UX Details

Coding Standards

Project Policy

A5. Flyer & Poster Design

A6. Copy of Evaluation Comments

Copy of Evaluation Comments by Supervisor for Project – I Mid Semester Evaluation

Copy of Evaluation Comments by Jury for Project – I End Semester Evaluation

Copy of Evaluation Comments by Supervisor for Project – II Mid Semester Evaluation

Copy of Evaluation Comments by Jury for Project – II Mid Semester Evaluation

Copy of Evaluation Comments by Jury for Project – II End Semester Evaluation

A7. Meetings' Minutes

A8. Research Paper

A10. Any other

A0. COPY OF PROJECT REGISTRATION FORM

A1A. PROJECT PROPOSAL AND VISION DOCUMENT

Any standard template may be used, as per project need approved by Project Coordinator & Supervisor. Following is a suggestive outline. **Also, the same outline should be used for Project Proposal Presentation.**

1. Introduction

In the rapidly evolving landscape of agricultural technology, the need for innovative, farmer-centric tools to manage livestock health and streamline animal care has never been more pressing. Recognizing this critical need, our mobile application — **Farmware** — is designed to revolutionize the way farmers monitor, assess, and manage the well-being of their animals with simplicity and precision.

Farmware offers a comprehensive platform tailored specifically for livestock keepers, enabling them to efficiently record, track, and evaluate the health status of individual cows. With intuitive navigation and a user-friendly interface, the application ensures that even users with limited technical expertise can effortlessly maintain accurate animal records, detect symptoms, and generate health reports — all from the palm of their hand.

Our vision is to empower farmers by providing a centralized solution that eliminates paperwork, supports informed decision-making, and enhances animal welfare. By combining smart symptom tracking, structured health records, and accessible design, Farmware aims to be an essential companion in every farmer's daily routine.

Through this digital innovation, we seek to redefine livestock management in rural and modern farms alike — making animal health tracking not just easier, but smarter, faster, and more accessible.

1. Objective

“To develop an app that monitors cow health, predicts diseases, provides precautions, and tracks recovery. The app will provide early alerts and actionable advice for optimal cattle care”

3. Problem Description

What

Farmware helps farmers manage cow health through daily symptom tracking, early disease prediction, and continuous health monitoring. Focused on Pakistani cow breeds and regional diseases, the app allows farmers to input cow and farm details, toggle observed symptoms daily and receive actionable insights through intelligent prediction algorithms. Key features include health record storage, symptom history, and treatment follow-up tracking.

Why

Cattle health is critical to farmers, especially in Pakistan, where agriculture plays a key economic role. Early detection and timely care reduce mortality, prevent outbreaks, and improve farm productivity. Traditional health tracking is often manual and delayed. Farmware automates this process, delivering timely alerts and personalized recommendations to support animal welfare and sustainable farming.

How

Farmers select visible symptoms daily using a simple toggle interface. This data feeds into a robust backend system that stores health records and uses predictive algorithms to detect possible diseases and suggest precautions. The app is designed for ease of use across varying technical skill levels and ensures reliability through fault-tolerant features, enabling uninterrupted monitoring and care.

4. Methodology

A hybrid software engineering approach combining Agile and Waterfall models was used to develop the Farmware App, balancing flexibility with structure.

- **Agile Practices:**
Iterative development enabled frequent testing and feedback from farmers. Regular fortnightly reviews helped refine features, such as replacing the initial questionnaire with a simpler symptom selection interface for better usability.
- **Waterfall Elements:**
The early stages followed a structured Waterfall approach to define the app's architecture, database schema, and disease prediction logic, ensuring a stable foundation for development.

5. Project Scope

Farmware will focus on monitoring cow health, predicting potential diseases, and providing follow-up monitoring based on user inputs and historical data. The application will emphasize a user-friendly interface for easy data entry and navigation. It will offer disease predictions, safety precautions, and follow-up assessments to help preserve the cow, monitor recovery, and prevent disease spread until an expert arrives. The project will not replace professional veterinary diagnosis and will not integrate with external farm management systems or advanced sensor data. It assumes that farmers have access to basic digital devices and internet connectivity and that basic training will be provided for effective use. The application will maintain a static database of diseases, symptoms, with periodic updates, but will not include real-time syncing with external medical databases. This scope ensures Farmware provides a comprehensive solution for cow health management, disease prevention, and treatment follow-up.

6. Feasibility Study

Risks Involved:

Technical risks may arise in integrating data collection and management features effectively. Mitigation will involve thorough testing and validation during the development phases. Resource risks include dependence on stable internet and server uptime, which may affect system reliability. To address this, we will set up backup servers and ensure critical parts have redundancies. Another risk is user adoption, as ensuring user acceptance among farmers may be challenging. This will be managed through user feedback and iterative improvements based on testing and engagement. Additionally, the accuracy of disease predictions and recommendations will be safeguarded by regularly updating the database with new information on symptoms and diseases to reflect emerging patterns.

Resource Requirements:

Computing resources will include robust servers capable of handling data processing and storage for cow health records, potentially utilizing cloud services. Software development tools will encompass Node.js and Python for backend development, React Native for frontend development, and PostgreSQL for database management. Budgetary considerations will involve allocating funds for software licenses, ongoing maintenance, and ensuring the application remains updated with relevant data. These resources are critical for sustaining the project and ensuring its reliability and effectiveness

Human Resources

1. Development Team: Skilled developers responsible for implementing cow data management, symptom-based diagnosis, and UI design tailored for farmers.
2. Quality Assurance Team: A dedicated QA team to rigorously test functionality, usability, and reliability across devices and network conditions.
3. Field Collaboration: Engaging with veterinarians and farmers for real-world feedback on symptom selection and health record accuracy.

Time Resources

1. Project Timeline: A structured timeline helped track progress and maintain focus on key milestones such as onboarding flow, health reporting, and symptom-based diagnosis.
2. Timely Feedback: Regular input from stakeholders and field testers enabled iterative improvements and faster issue resolution.

7. Solution Application Areas:

Farmware provides real value to the agricultural industry, particularly in livestock management, by offering tools for daily health tracking, disease prediction, and actionable recommendations. Targeted at dairy and beef farmers in Pakistan, it helps in early disease detection, enhances productivity, and reduces costs by enabling timely interventions. By supporting proactive health management and preventing disease outbreaks, Farmware contributes to improved animal welfare, cost savings, and sustainable farming practices.

8. Tools/Technology

Hardware:

Cloud Infrastructure

- Hosting Provider: Amazon Web Services (AWS)
 - Cost: PKR 28,000 – PKR 56,000 per

month Development and Testing Devices

- Computers/Laptops
 - Cost: PKR 224,000 – PKR 420,000 per device
- Mobile Devices
 - Cost: PKR 40,000 – PKR 280,000 per device

Software:

Operating System

- Windows
 - Cost: PKR 39,200 per

license Development Tools & IDEs

- IDE: Visual Studio Code (Free)
- Database Management System (DBMS): MySQL

(Free) Programming Languages

- Backend: Node.js (JavaScript/TypeScript), Python (Free)
- Frontend: React Native (Free)
- Database Queries: SQL

(Free) Frameworks and Libraries

- Scikit-learn – For machine learning and statistical modeling (Free)
- TensorFlow – For machine learning (Free)
- Bootstrap – For responsive design and UI components

(Free) Security Tools

- Bcrypt – For password encryption (Free)
- TLS (Transport Layer Security) – For secure data transmission
 - Cost: PKR 1,435 – PKR 14,350

8. Responsibilities of the Team Members:

Task/Activity	Maryam	Mujtaba	Hamza
UI Design	A, R	R	I
Node JS	I	A, R	I
Python	I	I	A, R
SQL	I	A, R	I
ML Algorithms	I	I	A, R
Data Collection	R	A, R	I
Data Analysis	I	I	A, R
Quality Assurance	A, R	I	R
Task Planning & Tracking	A, R	R	I
Budgeting and Finance	I	A, R	I
Security	I	A, R	I

Milestones

Phase 1: Research and Planning

Duration: 2 weeks Activities:

- Conduct literature review on common cow diseases.
- Analyze existing disease prediction models.
- Define project requirements and constraints.

Phase 2: Data Collection and Preprocessing

Duration: 3 weeks Activities:

- Gather relevant datasets (symptoms, environmental data, etc.).

- Clean and preprocess the data to ensure quality and consistency. - Identify key features for disease prediction.

Phase 3: Model Development

Duration: 4 weeks Activities:

- Select appropriate machine learning algorithms (e.g., Decision Trees, Random Forest, Neural Networks).
- Train models on the preprocessed dataset.
- Evaluate model performance using metrics like accuracy, precision, recall, and F1 score.

Phase 4: Application Development

Duration: 6 weeks Activities:

- Design the user interface (UI) for the application.
- Develop backend functionalities for data input, processing, and prediction.
- Implement data visualization tools for presenting results. - Test the application for usability and functionality.

Phase 5: Testing and Deployment

Duration: 3 weeks Activities:

- Perform rigorous testing to identify and fix bugs.
- Validate predictions against real-world cases.
- Deploy the application on a suitable platform (e.g., mobile or web). - Collect user feedback for final refinements.

Phase 6: Documentation and Presentation

Duration: 2 weeks Activities:

- Prepare detailed project documentation, including user guides and technical reports.
- Develop a presentation to showcase the application's features and results. - Submit the final project report.

11. References

- [1] Andrew Uden "Herddogg". Available at: [<https://herddogg.com/>]
- [2] Cow Manager Available at: [<https://www.cowmanager.com/>]
- [3] HerdWatch. Available at:[<https://herdwatch.ie/>

A1B. COPY OF PROPOSAL EVALUATION COMMENTS BY JURY

Hamdard University
Faculty of Engineering Sciences and Technology
Department of Computing

FYP -PE-2024

FINAL YEAR PROJECT - PROPOSAL EVALUATION

Project Title: Farmware App

Project ID: _____ Project Track: Product

Project Domain: ML/DS Evaluation Date: 31/7/24

Supervisor Name: Farooq Iqbal Co-Supervisor Name: _____

Project Member(s):

No.	Name	CMS ID
1.	Maryam Nadeem	2437-2021
2.	Muhammad Mujtaba	1985-2021
3.	Hamza Bin Asif	1961-2021
4.		

Evaluators only:

Evaluation Parameters	Please select the appropriate option E: Excellent G: Good S: Just Satisfactory N: Not Satisfactory			
	Evaluator #1	Evaluator #2	Evaluator #3	Evaluator #4
Subject Knowledge	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input checked="" type="checkbox"/> E <input type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N
Problem Statement	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N
Organization & Content of Presentation	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N
Project Scope Defined	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N
Methodology	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N
Language & Grammar	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N
Attire, Delivery and Presentation Skills	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N
Work Division	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N	<input type="checkbox"/> E <input checked="" type="checkbox"/> G <input type="checkbox"/> S <input type="checkbox"/> N
Name & Sign of Evaluator:	AB. Razaque A. Bano	Osama Ahmad Ibrahim Razaq Khan	Saleem Ali	Ali

Suggestions of evaluators:

As per recommendation by Jouny discuss with supervisor.
Accepted.
Suggestions: ① Dataset construction is going to take much time.
② Project scope needs to be widened to linked diseases's treatment prescription and follow-up monitoring of animals' health.

For FYP Committee only:

Result Summary

On basis of evaluations, recommended action decided in FYP committee meeting:

☐ Approved ☒ Approved (with Revision) ☐ Re-Evaluate

Date: 31/07/24 Name and Sign of Convener FYP Committee: Sulaman Ahmad Naz

A2. REQUIREMENT SPECIFICATIONS

Table of Contents of SRS

Section Title	Page
Introduction	35
Purpose of Document	35
Overall System Description	36
Project Background	36
Problem Statement	36
Project Scope	36
Not In Scope	36
Project Objectives	37
Stakeholders & Affected Groups	37
Operating Environment	37
System Constraints	37
Assumptions & Dependencies	37
External Interface Requirements	39
Hardware Interfaces	39
Software Interfaces	39
System Functions / Functional Requirements	40
System Functions	40
Use Cases	41
List of Actors	41
List of Use Cases	41
Non-Functional Requirements	44
Performance Requirements	44
Safety Requirements	44
Security Requirements	44
Reliability Requirements	44
Usability Requirements	44
Supportability Requirements	44
User Documentation	45
References	46

1. Introduction

Farmware is a mobile application developed to enhance cattle health management for farmers in Pakistan. By leveraging machine learning and adaptive data collection, the app allows farmers to input daily health data for their cows, receive early disease predictions, and access personalized recovery recommendations. Designed for Android devices, Farmware provides a user-friendly interface in both English and Urdu, ensuring accessibility for users with limited technical expertise. The system aims to reduce livestock mortality and improve overall farm productivity by offering timely insights and actionable advice, bridging the gap between traditional farming practices and modern digital health monitoring.

1.1 Purpose of Document

This document provides a detailed description of the functional and non-functional requirements for the Farmware App. It serves as a guide for the development and testing of the system.

1.2 Intended Audience

- Development Team
- Project Supervisors
- Testers
- Farmers (End-Users)

1.3 Abbreviations

SRS: Software Requirements Specification
AI: Artificial Intelligence
ML: Machine Learning
NLP: Natural Language Processing
DBMS: Database Management System

2. Overall System Description

2.1 Project Background

The **Farmware** project was initiated to address the challenges faced by Pakistani farmers in managing cattle health effectively. In many rural areas, the lack of timely disease detection and absence of proper monitoring tools contribute to high mortality rates and reduced livestock productivity. Farmware aims to fill this gap by providing a digital solution that leverages machine learning to analyze daily health inputs and predict potential diseases. The app not only assists in early diagnosis but also offers tailored health recommendations, enabling farmers to take preventive measures and track recovery. By digitizing and automating the health monitoring process, Farmware aspires to improve animal welfare and empower farmers with actionable insights.

2.2 Problem Statement

Farmers lack efficient tools to monitor cow health, leading to delays in disease detection and increased mortality rates. Farmware automates health monitoring, providing farmers with actionable insights to improve cattle care.

2.3 Project Scope

The scope of **Farmware** includes developing a mobile application that enables farmers to monitor the daily health of their cows, predict potential diseases using machine learning algorithms, provide personalized health recommendations based on symptoms, and track recovery progress over time. The app will support data input through symptom toggles, maintain historical health records for individual cows, and offer report export features. It will function on Android devices with internet access, using cloud storage and a secure backend system. However, the project does **not** cover integration with external farm management systems, real-time syncing with global medical databases, or replacement of professional veterinary diagnoses.

2.4 Not in Scope

- **Integration with external farm management systems:**

Farmware will function as a standalone application and does not include the integration or interoperability with other farm management or livestock tracking software. This means that data from external systems cannot be imported or synchronized automatically with Farmware.

- **Real-time syncing with external medical databases:**

The app does not support real-time communication or data exchange with global or external veterinary or medical databases. Disease predictions and health recommendations are based solely on the data entered by the farmer and the app's internal machine learning models, without real-time updates from external medical sources.

- **Replacing professional veterinary diagnosis:**

Farmware is designed to assist farmers by providing early warnings and health advice based on symptom input and machine learning predictions. However, it is not a substitute for professional veterinary consultation, diagnosis, or treatment. Users are encouraged to seek expert veterinary care for accurate diagnosis and treatment decisions.

2.5 Project Objectives

- **Symptom Input via Toggle Buttons**

Farmers can easily select observed symptoms for each cow daily using simple toggle buttons, making data entry quick and user-friendly.

- **Disease Prediction Using Machine Learning**

The app analyzes the input symptoms with machine learning models to predict potential diseases early, helping farmers take timely action.

- **Personalized Health Recommendations**

Based on the predicted disease, the app provides tailored advice and recovery suggestions to support the cow's health improvement.

- **Historical Health Records**

Farmware keeps a record of each cow's health data over time, enabling farmers to track progress and monitor recovery trends effectively.

2.6 Stakeholders & Affected Groups

- Farmers
- Agricultural organizations
- Veterinary professionals

2.7 Operating Environment

Farmware will operate on Android devices with internet access.

2.8 System Constraints

- Internet connectivity is required for accessing the database.
- The system relies on periodic updates to maintain accuracy in disease prediction

2.9 Assumptions & Dependencies

3. Farmers will have access to basic digital devices.
4. Basic training will be provided to ensure proper usage
5. Android mobile devices for hardware interfaces.
6. Cloud servers (AWS) for data storage.

7. Database: MySQL for managing health records.
8. Backend: Node.js, Python.
9. Frontend: React Native.
10. Internet communication for data synchronization with the cloud.

11. External Interface Requirements

Hardware Interfaces

- Android Devices:
 - Logical Structure: The application will be installed on smartphones and tablets running Android OS.
 - Physical Addresses: Compatible with devices that have a minimum of 2 GB RAM and Android version 6.0 (Marshmallow) or higher.
 - Expected Behavior: The application should utilize device sensors (e.g., GPS for location tracking) and display health data in a user-friendly manner.
- Cloud Servers:
 - Logical Structure: AWS (Amazon Web Services) for data storage and processing.
 - Physical Addresses: The application will interact with cloud endpoints for database access.
 - Expected Behavior: The servers must provide quick response times for data retrieval and storage, ensuring that health data is available to users in real-time.

Software Interfaces

- Database Management System:
 - Name of Application: MySQL
 - External Owner of Application: Open-source community
 - Interface Details: The application will connect to MySQL for managing health records using SQL queries for data retrieval and storage.
- Backend Framework:
 - Name of Application: Node.js
 - External Owner of Application: Open-source community
 - Interface Details: The application will use RESTful APIs to communicate between the frontend and the backend services.
- Machine Learning Libraries:
 - Name of Application: Scikit-learn and TensorFlow
 - External Owner of Application: Open-source community
 - Interface Details: These libraries will be used for disease prediction and analysis, interacting through Python scripts.

Communication Interfaces

- Internet Connectivity:
 - The application will require a stable internet connection for real-time data exchange between the mobile app and the cloud servers. This will ensure timely access to health records and disease predictions.
- Local Area Networks (LAN):
 - The application may also support communication with local servers for organizations that wish to manage their own database, allowing for data sharing within a specific agricultural community.
- APIs:
 - The application will use RESTful APIs for communication between the frontend (React Native) and backend (Node.js) services to ensure smooth data flow and user experience.

12. System Functions / Functional Requirements

12.1 System Functions

1. Health Monitoring

- 1.1 **Daily Symptom Input:** Farmers can input daily health symptoms of cows using toggle buttons. Data entry and saving must be completed within 5 seconds of submission.
- 1.2 **Health History Tracking:** The system displays historical health records and trends for each cow. Historical data must be accessible within 3 seconds upon request.

2. Disease Prediction & Recommendations

- 2.1 **Disease Prediction:** The system uses machine learning to predict diseases based on collected symptom data. Prediction accuracy must be at least 80%.
- 2.2 **Tailored Recommendations:** After disease prediction, the app provides personalized health advice. Recommendations must be easy to understand and clearly presented.
- 2.3 **Adaptive Inputs:** (Note: Not currently used in the app per your previous notes.) The system can be designed to generate adaptive health input flows based on previous entries.

3. User Management

- 3.1 **Secure Login:** Farmers must log in using secure authentication. User credentials must be protected via strong password policies and encrypted transmission (TLS).

4. Reporting & Export

- 4.1 **Data Export:** Farmers can export cow health records in CSV or PDF format upon request.

5. System Attributes

- 5.1 **Performance:** The system must support a response time of less than 5 seconds for data input and 3 seconds for historical data retrieval.
- 5.2 **Concurrent Usage:** The system must support at least 10 users online simultaneously.
- 5.3 **Data Availability:** Health records and reports must be available for access 24/7.
- 5.4 **User Interface:** The app must feature a simple, intuitive UI designed for users with limited technical experience.
- 5.5 **Language Support:** English and Urdu are supported. Additional regional languages may be considered for future phases.
- 5.6 **Database Scope:** The app will use a locally relevant static disease-symptom database, updated periodically. Real-time global integration is out of scope.

5.7 **User Training:** Basic documentation and support resources are provided. In-person training is not included in the current phase.

5.8 **System Expansion:** The architecture should support future additions like other livestock, crop modules, or new animal breeds.

1.1 Use Cases

1.1.1 List of Actors

- **Farmer:** The primary user of the application who inputs health data for cows, views health reports, and receives disease predictions and recommendations.
- **Veterinary Expert:** Provides expert reviews and insights into the cow health data, assisting in validating disease predictions and updating health recommendations.
- **System (Farmware Application):** Responsible for processing input data, analyzing health trends, generating predictions, and storing data.
- **Admin:** Manages user accounts, application settings, and oversees data accuracy in the system.

1.1.2 List of Use Cases

1. Record Health Data

Actors: Farmer, System

Description: Farmers input daily health data for each cow, selecting symptoms and noting behavioral changes.

Preconditions: Farmer must be logged in and have at least one cow registered.

Postconditions: Health data is saved in the system and associated with the respective cow's profile.

2. Generate Disease Predictions

Actors: System

Description: The system analyzes submitted health data using machine learning algorithms to predict potential diseases.

Preconditions: Relevant symptom data must be available for the cow.

Postconditions: Disease prediction results are stored and linked to the cow's health record.

3. Provide Health Recommendations

Actors: System, Farmer

Description: Based on disease predictions, the system offers personalized health advice and

preventive actions.

Preconditions: Disease predictions must be available.

Postconditions: Recommendations are displayed to the farmer for follow-up action.

4. Display Health History

Actors: Farmer

Description: Farmers can view historical health data and trends for each cow.

Preconditions: Health data must have been recorded previously.

Postconditions: Historical records and charts are shown for selected cows.

5. Review and Update Health Data

Actors: Veterinary Expert

Description: Veterinary experts review submitted data and may update or add professional recommendations.

Preconditions: Health data must be available in the system.

Postconditions: Expert insights and updated recommendations are saved.

6. Manage User Accounts

Actors: Admin

Description: Admins create, modify, or delete farmer accounts and manage access permissions.

Preconditions: Admin must be authenticated.

Postconditions: User account changes are applied and updated in the system.

7. Data Backup and Recovery

Actors: System

Description: The system performs automatic data backups and allows recovery in case of system failure.

Preconditions: System backup service must be active.

Postconditions: Data is securely stored and recoverable after failure.

8. Export Health Reports

Actors: Farmer

Description: Farmers can export individual cow health records in CSV or PDF format.

Preconditions: Health data must exist for the cow.

Postconditions: A downloadable file is generated and shared with the farmer.

9. Receive Notifications and Alerts

Actors: System, Farmer

Description: The system sends reminders and alerts related to health updates, risk warnings, and data entry prompts.

Preconditions: Notification settings must be enabled.

Postconditions: Farmer receives timely notifications for relevant events.

10. System Maintenance and Updates

Actors: Admin

Description: Admins maintain the system and push updates to improve performance and fix issues.

Preconditions: Admin must have system access rights.

Postconditions: System is updated and running smoothly with improved performance.

11. Access Health Insights

Actors: Farmer, Veterinary Expert

Description: Farmers and experts can view aggregated insights and trends across different cows, including disease risk levels.

Preconditions: The system must have sufficient data to generate insights.

Postconditions: Visual insights and risk summaries are displayed for review.

1. Non - Functional Requirements

1.1 Performance Requirements

- The system should handle the **daily input of health data for multiple cows** without performance degradation.
- **Disease predictions** must be generated within **5 seconds** of health data submission.
- The system must support **simultaneous usage by up to 1000 farmers** without noticeable latency.

1.2. Safety Requirements

- The system must ensure that **user data, including health records and cow information, is securely stored** with regular automated backups.
- It must comply with **relevant data protection laws** to safeguard user privacy and security.
- The system should include **error-handling mechanisms** to prevent data loss, such as **automated data recovery** during system failure.

1.3. Security Requirements

- **User authentication** must be implemented using **secure passwords**.
- All sensitive data (e.g., personal details, health data) must be **encrypted both at rest and in transit** using **industry-standard encryption protocols** (e.g., AES-256).
- Access to the app should be restricted through **secure login only**, with no role-based permissions structure.

1.4. Reliability Requirements

- The system should maintain an **uptime of 99.9%**, excluding planned maintenance.
- It must be capable of **automatic failure recovery**, ensuring no data loss during downtime.
- The system must be **scalable** to accommodate an increasing number of users and monitored cows **without degrading performance**.

1.5. Usability Requirements

- The **user interface** should be **simple, intuitive, and accessible**, especially for farmers with limited technical skills.
- The system must be available on **Android devices**, with a possibility for desktop access in the future.
- Real-time feedback should be provided on health data entries, with **clear alerts and actionable health recommendations**.

1.6. Supportability Requirements

- The system must include **troubleshooting tools**, such as **error logs** and **diagnostic utilities** for developers and admins.
- It should provide **regular software updates** to address bugs, improve security, and enhance performance.

- **Basic support** should be available via **documentation or help resources**, with live support as a future enhancement.

1.7. User Documentation

- The system must include **clear, farmer-friendly documentation**, such as:
 - A **step-by-step guide** for data entry and understanding predictions.
 - **FAQs** and solutions for common issues.
 - **Educational content** on cow healthcare and disease prevention, accessible directly within the app.

1. References

1. SRS for Livestock Managment System 1.0

<https://www.studocu.com/row/document/riphah-international-university/computersciences/srs-first-final-draft-srs-for-livestock-managment-system-10/42526023>

A3. DESIGN SPECIFICATIONS

Any standard template may be used, as per project need approved by Project Coordinator & Supervisor. Following is a suggestive outline.

Definition of Terms, Acronyms, and Abbreviations

Term	Description
SDS	Software Design Specifications
UI	User Interface
DB	Database
SRS	Software Requirement Specifications
API	Application Programming Interface
PK	Primary Key
FK	Foreign Key

Table of Contents

Section Title	Page
Table of Contents	
1. Introduction	49
1.1 Purpose of Document	49
1.2 Intended Audience	49
1.3 Project Overview	49
1.4 Scope	49
2. Design Considerations	50
3. System Architecture	51
Overview of Architecture	51
3.1 System Level Architecture	51
3.1.1 System Decomposition into Elements	51
3.1.2 The Relationship between the Elements	51
3.1.2.1 Interfaces to External Systems	52
3.1.3 Major Physical Design Issues	53
3.2 Software Architecture	53
3.2.1 User Interface Layer	53
3.2.2 Middle Tier	54
3.2.3 Data Access Layer	54
4. Design Strategy	54
4.1 User Interface Paradigms	54
4.2 Data Management	54
4.3 Concurrency & Synchronization	54
4.4 Future System Extension or Enhancement	55
5. Detailed System Design	56
5.1 Design Class Diagram	56
Database Design	57
ER Diagram	57
Application Design	58
Sequence Diagram	58
◦ Sequence Diagram 1	58
State Diagram	59
◦ State Diagram 1	59
DFD Level 1	60
GUI Design	61

Splash Screen	61
Login	62
Signup	63
Home	64
Side Navigation	65
Medical Records	66
Farmer Profile	67
Cow Profile	68
Cow Form	69
Symptom Selection	70
Vaccinations	71
Settings	72
References	73

1 Introduction

1.1 Purpose of Document

The purpose of this Software Design Specification (SDS) document is to define the architecture, components, data design, and interaction mechanisms of the Farmware application. This document provides a comprehensive blueprint for developers, testers, and other stakeholders to ensure the system is built efficiently and aligns with the functional and non-functional requirements outlined in the Software Requirements Specification (SRS). Additionally, it serves as a guide for the development team to implement and maintain the system effectively.

1.2 Intended Audience

- *Development Team*
- *Project Supervisors*
- *Testers*
- *Farmers (End-Users)* .

1.3 Project Overview

Farmware is a mobile application designed to streamline cattle health management for farmers, with a specific focus on Pakistani cow breeds and regional diseases. The application enables users to monitor the daily health of cows, predict potential diseases using machine learning algorithms, and receive tailored recommendations for treatment and prevention. By leveraging data collection and advanced analytics, Farmware helps improve livestock health, productivity, and overall farm management.

1.4 Scope

Farmware will:

- *Track cow health daily.*
- *Predict diseases using ML algorithms.*
- *Offer tailored advice based on symptoms.*
- *Provide recovery monitoring.*

Not in Scope

- *Integration with external farm management systems.*
- *Real-time syncing with external medical databases.*
- *Replacing professional veterinary diagnosis.*

2 Design Considerations

Issue: As Farmware is aimed at farmers who may not be tech-savvy, the user interface must be intuitive, simple, and easy to navigate.

Consideration: The design should prioritize clarity and ease of use, especially for older generations or those not familiar with complex digital tools. It's essential to minimize user input errors by using clear labels, error messages, and easy-to-understand options.

Action: Focus on a minimalistic and responsive design. The use of large buttons, clear navigation, and a well-organized layout will ensure accessibility.

Issue: The large volume of health data, including cow history and symptom tracking, must be stored efficiently.

Consideration: The backend needs a well-structured database to store health data, cow profiles, and disease predictions. **Scalability** is key to ensuring the system can handle large amounts of data as the number of users grows.

Action: Use a **relational database** like **MySQL** or **PostgreSQL** to store structured data. Consider implementing **data backup** and **restore mechanisms** to ensure data integrity and recovery in case of system failures.

Issue: Disease prediction based on symptoms is a complex task that requires a high degree of accuracy and reliability.

Consideration: Implementing machine learning (ML) algorithms for disease prediction requires accurate training data and fine-tuning to avoid false positives or negatives.

Action: Collaborate with veterinarians and domain experts to collect **highquality training data**. Consider using algorithms like **decision trees** or **neural networks** (e.g., **TensorFlow** or **Scikit-learn**) to analyze historical health data and predict diseases.

Issue: As the app grows, the backend may face performance challenges, especially with large datasets and concurrent users.

Consideration: Ensure that the backend is optimized to handle multiple simultaneous connections and that it can scale with increasing usage.

Action: Implement **caching** for frequently accessed data and use **load balancing** for the server to manage high traffic. Consider optimizing queries and using indexing in the database for faster access to records.

Issue: Farmers may not be familiar with using apps or new technology, making training and support an essential part of the system.

Consideration: Provide clear instructions, **tooltips**, and tutorials within the app. Offering customer support channels (e.g., chat or phone support) is also crucial.

Action: Include a **help section** with frequently asked questions (FAQs), video tutorials, and in-app tips to guide new users.

3 System Architecture

The Farmware system follows a **modular architecture**, where functionality is divided into key components, each responsible for specific tasks. This approach ensures flexibility, scalability, and maintainability.

Overview of Architecture

1. **Presentation Tier**
2. **Application Tier**
3. **Data Tier**

3.1 System Level Architecture

The system-level architecture of the **Farmware** mobile application is designed to deliver a reliable, scalable, and secure platform for farmers to manage and monitor cattle health. The architecture follows a **three-tier design**: the **presentation tier**, the **application tier**, and the **data tier**.

3.1.1 System Decomposition into Elements

- **Presentation Tier:**
This tier consists of the **Android mobile interface** used by farmers. It is designed to be **simple and intuitive**, supporting both **English and Urdu** languages. This tier enables users to input daily health data, view disease predictions, access recommendations, and track historical records.
- **Application Tier:**
The core business logic resides in this tier. It handles **data processing, machine learning-based disease predictions, and generation of health recommendations**. This layer also manages adaptive data collection logic and ensures fast response times for data submission and retrieval.
- **Data Tier:**
This tier is responsible for **storing and managing all application data**. It includes a **cloud-based secure database** that stores user credentials, cow health records, historical logs, and disease prediction results. The system uses **encryption and routine backups** to ensure data integrity and security.

3.1.2 The Relationship between the Elements

Mobile App (Android) interacts with the **Frontend Interface** to allow users (farmers) to input health data and view predictions.

The **Frontend Interface** communicates with the **Backend Server** to send user requests such as health data submission and report viewing.

The **Backend Server** processes these requests, including invoking the **Machine Learning models** to generate disease predictions and recommendations.

The **Backend Server** interacts with the **Cloud Database** through secure **RESTful APIs** to

retrieve or store cow health records and user data.

3.1.2.1 Interfaces to External Systems

- **Email Service:**

Used for sending health report exports, confirmation emails, and support-related notifications to farmers.

- **Cloud Storage Service:**

Integrated to store and retrieve user data, including daily health inputs, reports, and historical records securely.

3.1.3 Major Physical Design Issues

- **Network Dependency:**

As Farmware relies on internet connectivity, the design must ensure efficient performance under low-bandwidth conditions. Techniques such as caching recent data and queuing offline submissions for later syncing may be implemented.

- **Scalability:**

The backend system must scale horizontally to support thousands of users and large volumes of cow health data. Serverless or container-based infrastructure (e.g., AWS Lambda, Docker) can be considered for flexible scaling.

- **Battery and Device Constraints:**

Since the app is used on mobile devices in rural areas, the app is optimized for minimal battery and memory usage, with lightweight UI components and efficient data calls.

3.2 Software Architecture

The Farmware system follows a layered architecture to ensure scalability, maintainability, and separation of concerns. The architecture consists of three main layers: **User Interface Layer**, **Middle Tier (Business Logic)**, and **Data Access Layer**.

3.2.1 User Interface Layer

- The User Interface (UI) Layer provides the front-end experience for farmers. It collects data through symptom selection and displays disease predictions.
- Interaction: This layer communicates with the Business Logic Layer to pass user inputs and display responses.

3.2.2 Middle Tier

- This is the core of the application where the business rules, symptom selection logic, and disease prediction algorithms are executed.
- Interaction: It processes the data received from the UI and sends requests to the Data Access Layer for data storage and retrieval. It also interacts with external services (like ML models) for disease prediction.
- The Data Access Layer manages interactions with the database, ensuring data integrity and handling requests to read and write health records, cow profiles, and predictions.
- Interaction: It receives requests from the Business Logic Layer to fetch, update, or delete data in the database.

4 Design Strategy

The design strategy focuses on creating a robust, scalable, secure, and user-friendly platform through the following key principles:

4.1 User Interface Paradigms

- **Strategy:** A minimalistic and intuitive design is adopted, focusing on ease of navigation. The user interface features clear labels, large toggle buttons, and responsive layouts to support farmers with varying levels of technological literacy. The symptom toggling system simplifies health reporting by allowing users to select observed symptoms directly, reducing complexity and enhancing usability.
- **Reasoning:** An accessible design ensures that the app is usable even by farmers with limited technical expertise, improving adoption rates.

4.2 Data Management (Storage, Distribution, Persistence)

- **Strategy:** Data is stored in a relational database (MySQL), which supports efficient querying and management of structured data like health records and disease predictions. Backup mechanisms ensure data integrity, while future plans include cloud integration for scalability. Data persistence is managed to enable offline data entry, with synchronization when the internet is available.
- **Reasoning:** Relational databases provide robust and scalable solutions for managing large datasets, while cloud integration ensures the system can scale as usage grows.

4.3 Concurrency and Synchronization

- **Strategy:** The backend is designed to handle concurrent requests efficiently by employing asynchronous processing techniques. This ensures that multiple users can interact with the system simultaneously without performance degradation.
- **Reasoning:** This approach ensures the system remains responsive and reliable, even as the number of users grows. By prioritizing data consistency and integrity, it avoids issues arising from simultaneous data updates or network interruptions.

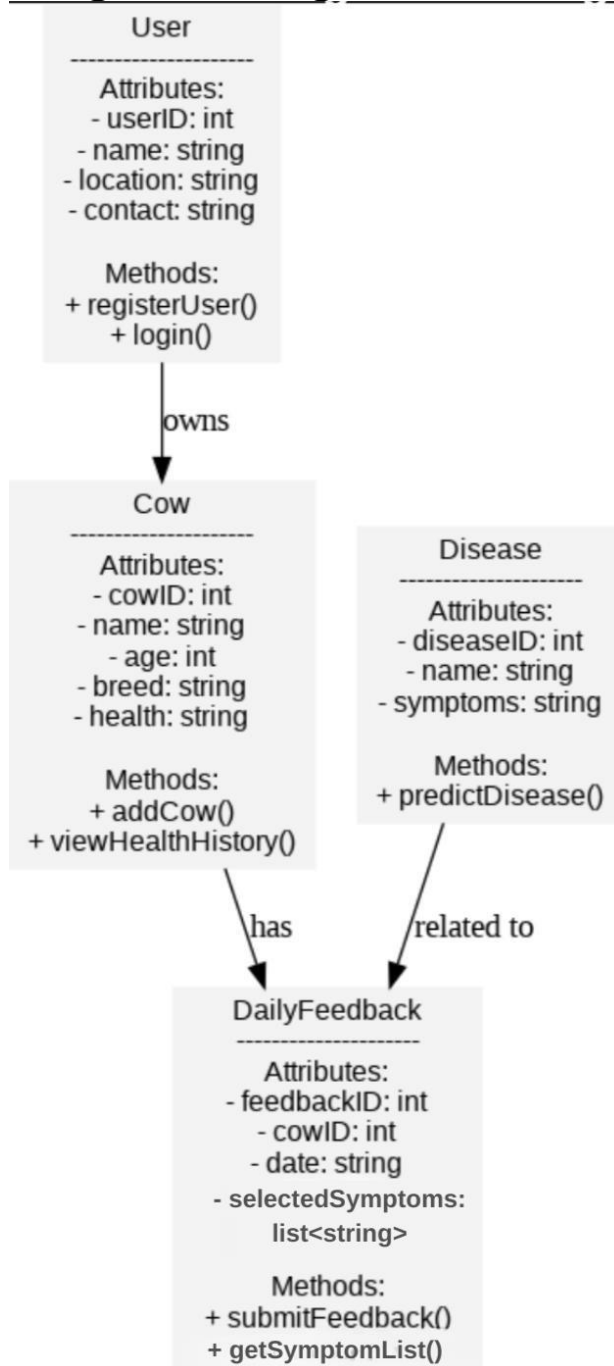
4.4 Future System Extension or Enhancement

- **Strategy:** The system is designed to support future extensions by adopting a flexible and scalable architecture. This ensures that new functionalities can be added seamlessly without affecting the core system.
- **Reasoning:** A flexible design approach enhances maintainability and adaptability, allowing the system to evolve in response to changing needs.

5 Detailed System Design

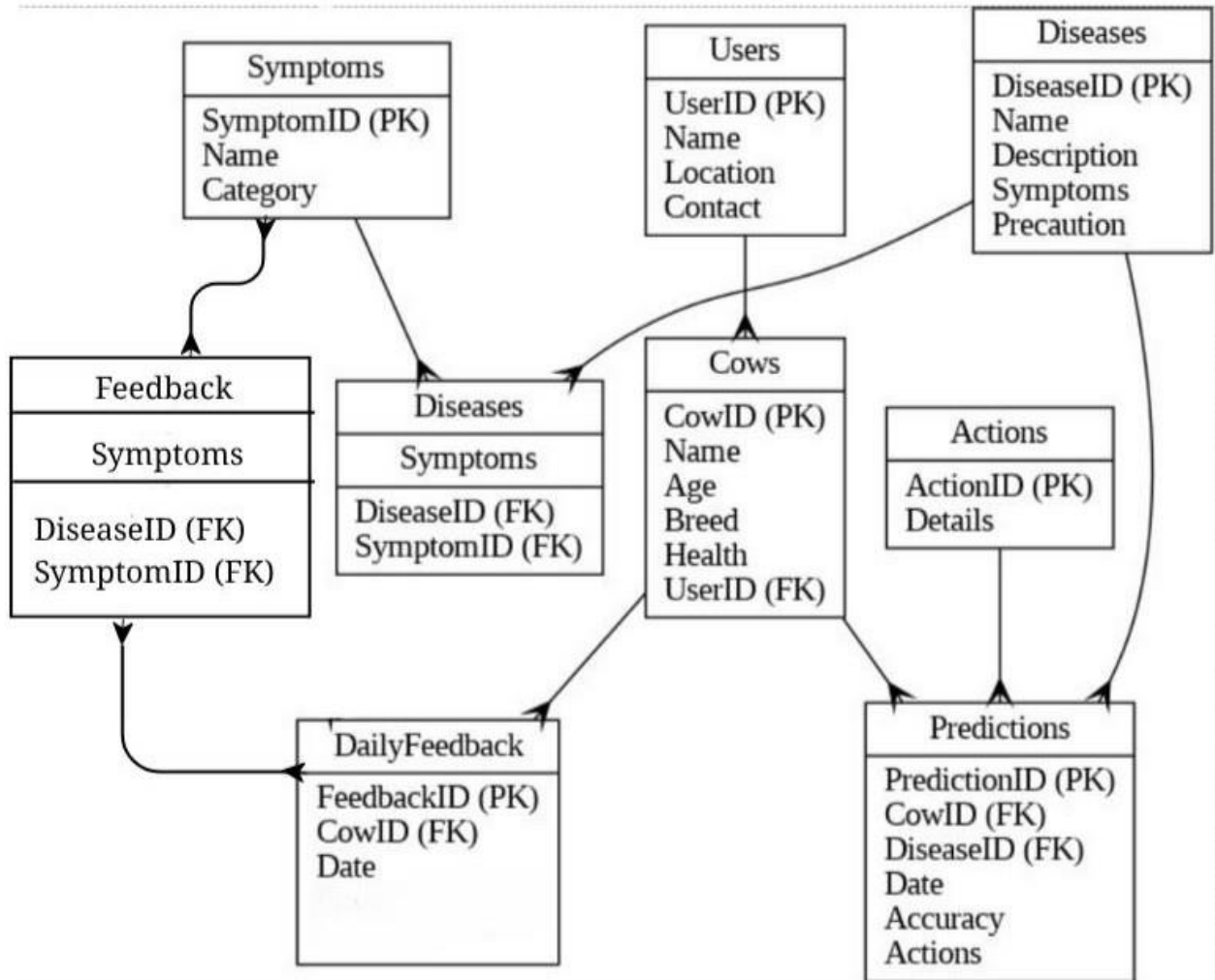
[A detailed design should include the following:

5.1 Design Class Diagram



5.2 Database Design

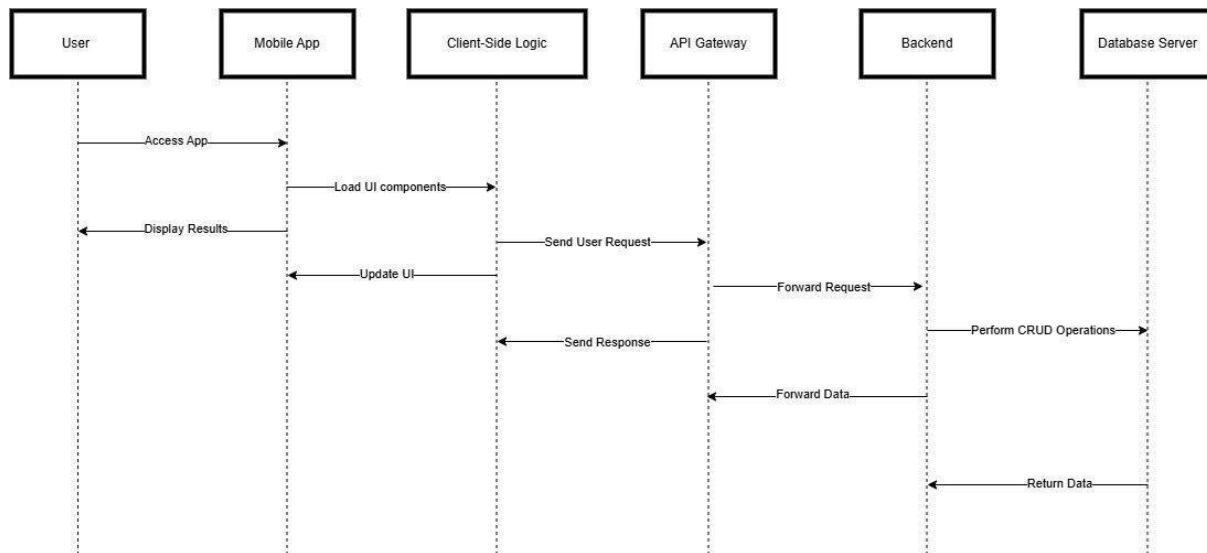
5.3 ER Diagram



5.4 Application Design

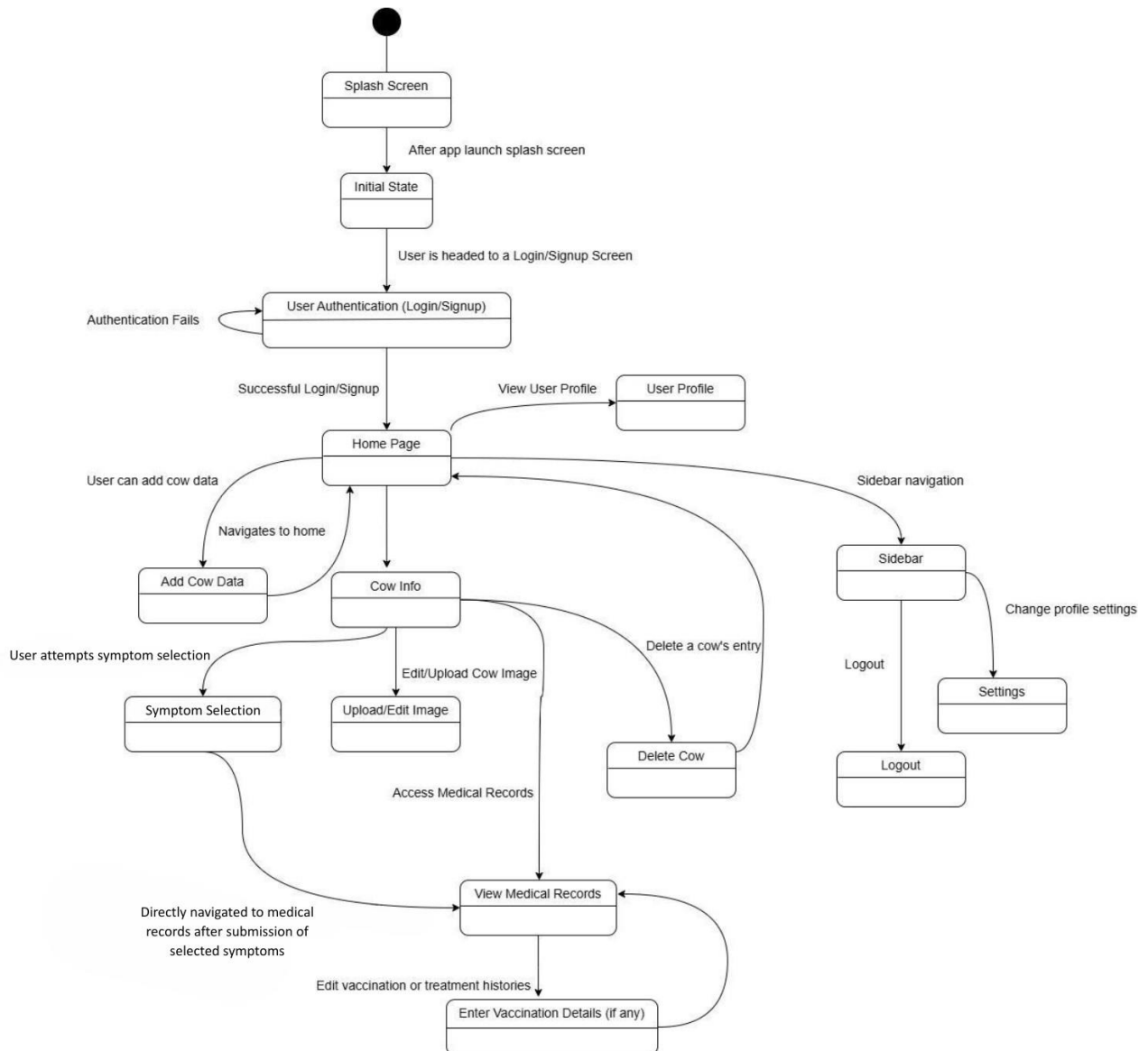
5.4.1 Sequence Diagram

5.4.1.1 <Sequence Diagram 1>

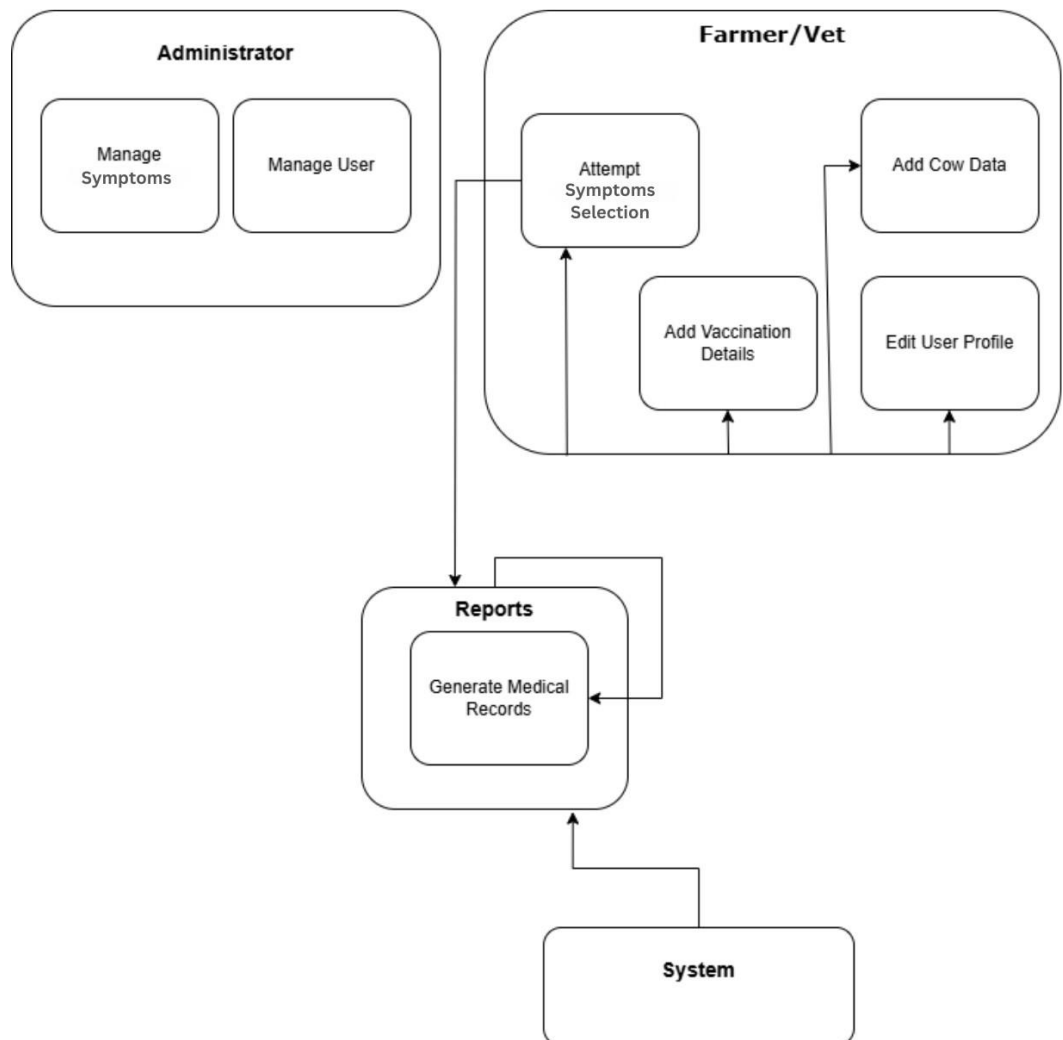


5.4.2 State Diagram

5.4.2.1 <State Diagram 1>

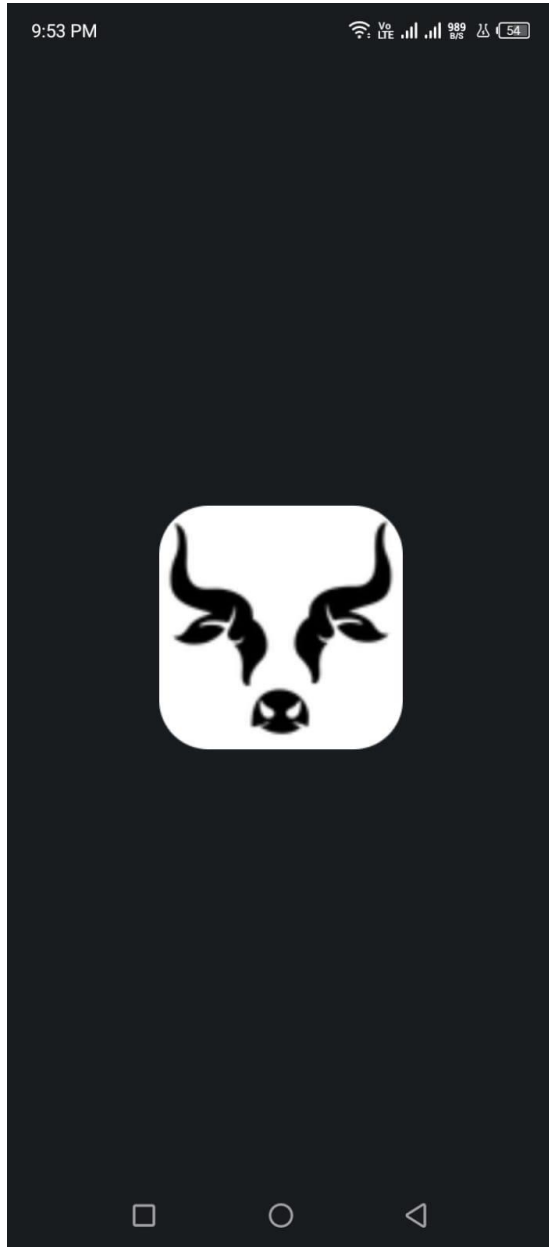


5.4.3 DFD level 1








5.5 GUI Design


5.1 < Splash Screen - Mock Screen 1 >



5.2 Login Page - Mock Screen 2

9:53 PM






Login Your Account

Sign in to continue


Name

Password

Forgot Password?

 LOGIN

Register/SignUp



5.3 Sign Up Page - Mock Screen 3 >

9:53 PM


Vo

LTE

271

3/5

54




Create a New Account

Already Registered? Login here

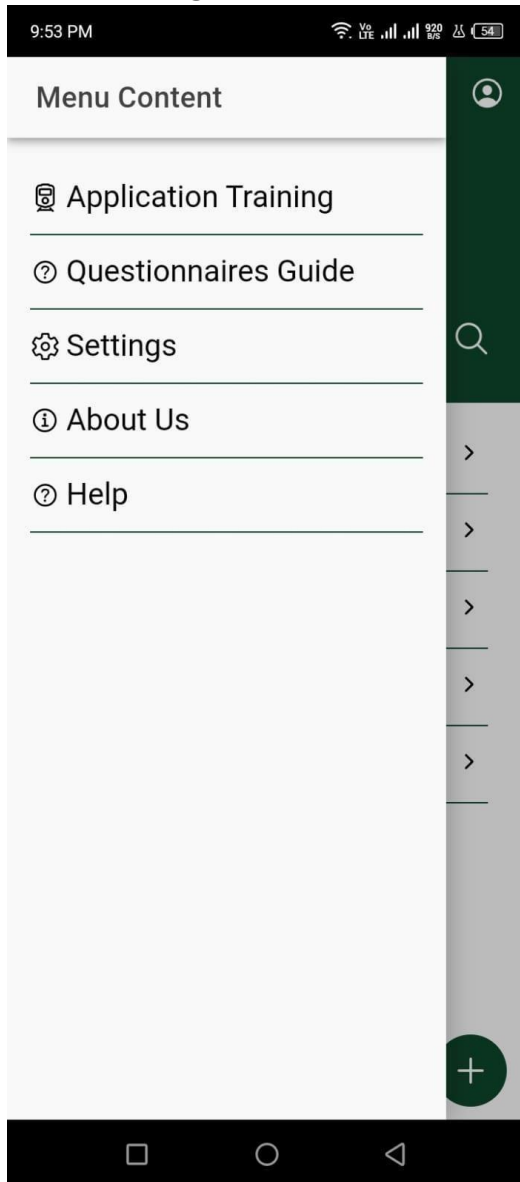
Farmer's Name

Email

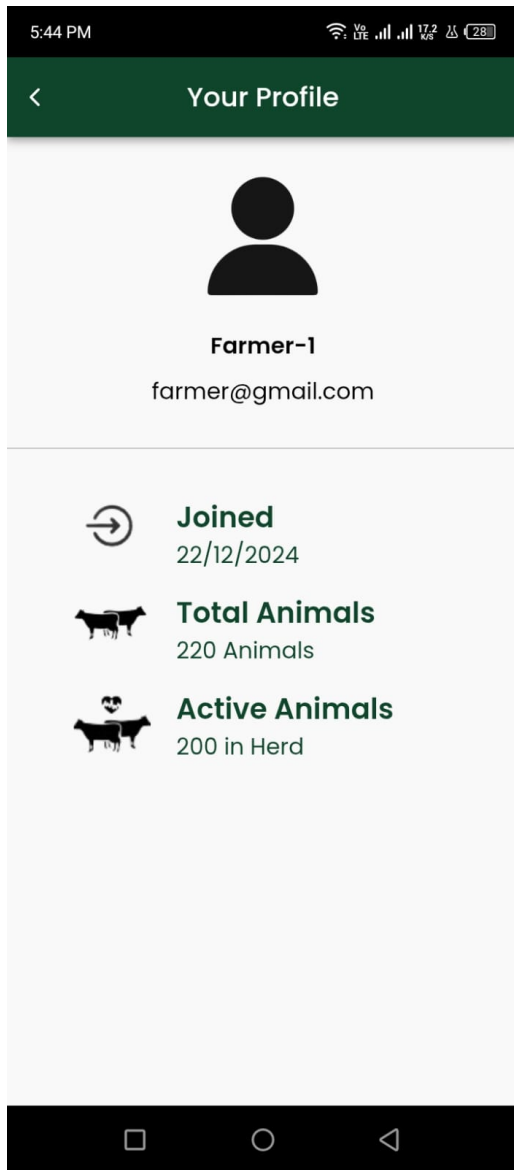
Password

 SIGN UP

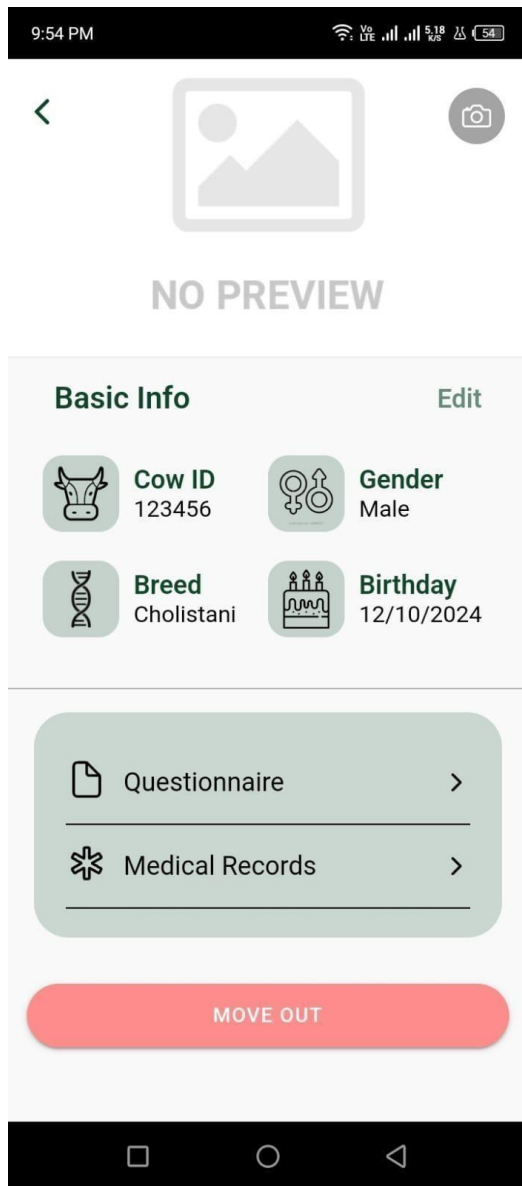
5.5 Side Navigation - Mock Screen 5 >



5.6 Farmer Profile - Mock Screen 6 >



5.7 Cow Profile - Mock Screen 7 >



5.8 Cow Form - Mock Screen 8 >

9:54 PM

< New Cow >

Gender *

Choose a gender ▼

Tag Number *

e.g 1, 2, 3...

Date of Birth *

Image

Add Image

Breed

Enter the breed

Age

Enter the age

Cancel | Done

5.9 <Symptom Selection - Mock Screen 9 >

The mockup shows a mobile app interface for selecting symptoms. At the top, a status bar displays the time as 5:45 PM and various system icons. Below this, a back arrow icon is followed by the title "Choose the Symptoms You See in Your Cow" in bold. A subtitle states, "This will be adapted by the questionnaire". The main content area contains 18 green, rounded rectangular buttons with white text, arranged in a staggered, vertical list. The symptoms listed are: Eating Lesser, Less Water Intake, Seems Tired, Lameness, Weak, Tired, Fever, Blood or lumps in milk, Giving less milk suddenly, Coughing or runny nose, Losing weight quickly, Red or watery eyes, Blood in urine, Yellow nose discharge, Bloody nose discharge, Udder feeling hot, and Swollen Udder. At the bottom of the screen is a black navigation bar with three white icons: a square, a circle, and a triangle.

5:45 PM

< Choose the Symptoms You See in Your Cow

This will be adapted by the questionnaire

Eating Lesser Less Water Intake

Seems Tired Lameness Weak

Tired Fever

Blood or lumps in milk

Giving less milk suddenly

Coughing or runny nose

Losing weight quickly

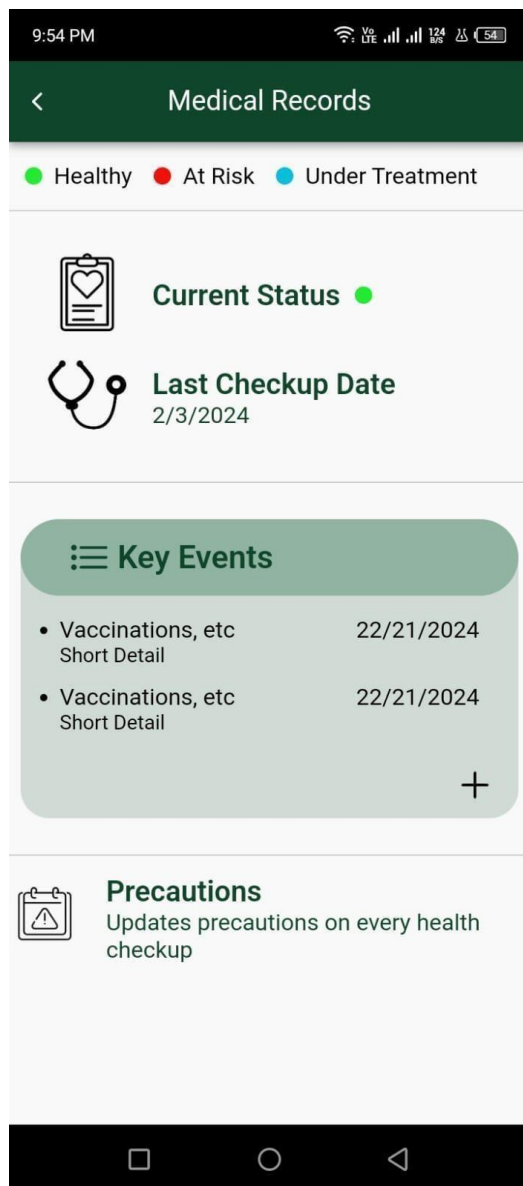
Red or watery eyes Blood in urine

Yellow nose discharge

Bloody nose discharge

Udder feeling hot Swollen Udder

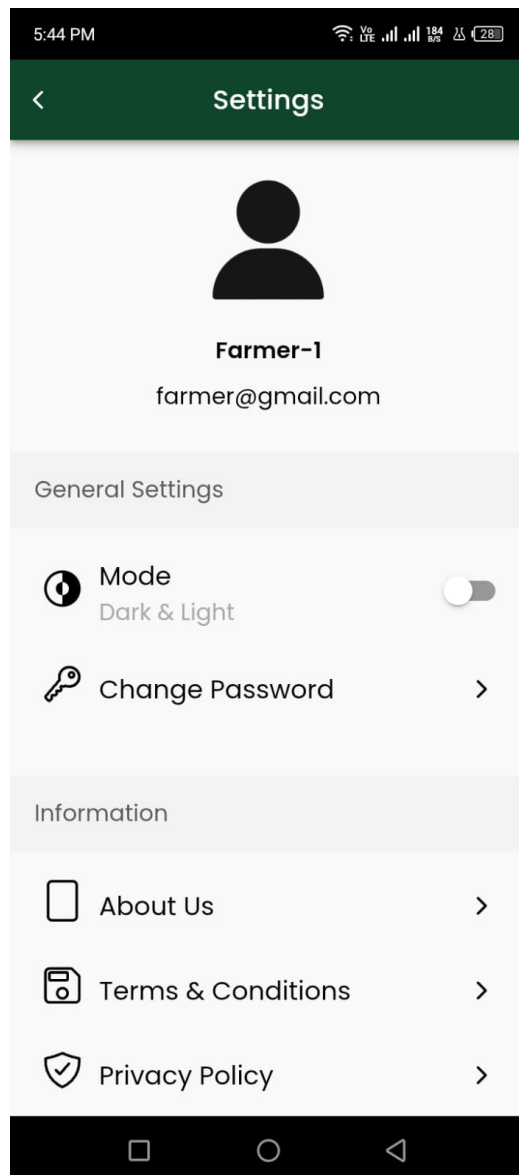
5.10 Medical Records - Mock Screen 10 >



5.11 Vaccination/Surgery Event - Mock Screen 11 >

The mockup shows a mobile application interface for adding an event. At the top, a status bar displays the time as 9:54 PM, along with icons for Wi-Fi, cellular signal, and battery level at 54%. Below the status bar is a header bar with the text 'Add Event(s)' and a close button (X). The main content area is a light gray rectangle. It contains two input fields: the first is labeled 'Enter Event Name' with the placeholder text 'Vaccination, etc'; the second is labeled 'Date' with the placeholder text 'Enter the date when the event occurred'. Below these fields is a green button with the text 'SAVE'. At the bottom of the screen is a black navigation bar with three white icons: a square, a circle, and a triangle.

5.12 Settings - Mock Screen 12 >



6 References

Ionic Official Website: <https://ionicframework.com/>

UI Reference: <https://www.bivatec.com/apps/my-cattle-manager>

Design Guidance: <https://denovers.com/blog/what-is-modular-design/>

A4. OTHER TECHNICAL DETAIL DOCUMENTS

Test Cases Document

Software Test Plan

S. No	Description	Test Engineer	Start Date	End Date
1	Login & Signup (Farmer Account Creation and Authentication)	Muhammad Mujtaba	03-June-2025	04-June-2025
2	Add New Animal Profile (Cow Data Entry: Name, Tag, DOB, Breed)	Hamza Bin Asif	05-June-2025	05-June-2025
3	Daily Symptom Selection (Toggle Buttons for Health Data Collection)	Hamza Bin Asif	06-June-2025	06-June-2025
4	Disease Prediction (Based on Selected Symptoms)	Hamza Bin Asif	07-June-2025	07-June-2025
5	Health Recommendations Display (Tailored Advice Post-Prediction)	Muhammad Mujtaba	08-June-2025	08-June-2025
6	Cow Health Profile (View Health History, Disease Records)	Muhammad Mujtaba	09-June-2025	09-June-2025
7	Report Generation (Daily Health Report Summary)	Muhammad Mujtaba	10-June-2025	10-June-2025
8	Export Health Reports (PDF and CSV Formats)	Muhammad Mujtaba	11-June-2025	11-June-2025
9	Connectivity Handling	Hamza Bin Asif	13-June-2025	13-June-2025
10	Help Section Access (FAQs, Tutorials, Contact)	Maryam Nadeem	14-June-2025	14-June-2025
11	App Logo Navigation (Redirect to Dashboard/Home)	Maryam Nadeem	14-June-2025	14-June-2025

Test Cases (for screen/reports)

Project Name: Farmware

Iteration No: 01

Test Engineer: Muhammad Mujtaba

Date: 03-Jan-2025

Test Scenario: **Login & Signup (Farmer Account Creation and Authentication)**

Test Case Description: To verify that the login and signup functionality works correctly for farmers, including proper input validation, error handling, and successful account creation and authentication.

Test Cases for Login

TC ID	Steps	Input Data	Expected Result	Actual Result	Pass/Fail
TC-01	Leave email and password fields empty and click login	Empty fields	Error: "Fields cannot be empty."	Error: "Fields cannot be empty."	Pass
TC-02	Enter invalid email format and click signup	Email: farmer123.com, Password: abc123	Error: "Please enter a valid email address."	Error: "Please enter a valid email address."	Pass
TC-03	Enter weak password during signup	Email: farmer@test.com, Password: 123	Error: "Password must be at least 6 characters."	Error: "Password must be at least 6 characters."	Pass
TC-04	Enter existing email during signup	Email: existing@test.com, Password: abc123	Error: "Email already registered."	Error: "Email already registered."	Pass
TC-05	Enter valid credentials and click signup	Email: newfarmer@test.com, Password: farm1234	Signup successful. Redirect to dashboard.	Signup successful. Redirect to dashboard.	Pass
TC-06	Enter wrong credentials during login	Email: farmer@test.com, Password: wrongpass	Error: "Incorrect email or password."	Error: "Incorrect email or password."	Pass
TC-07	Enter valid credentials and click login	Email: farmer@test.com, Password: farm1234	Login successful. Redirect to dashboard.	Login successful. Redirect to dashboard.	Pass

Test Cases (for screen/reports)

Project Name: Farmware

Iteration No: 01

Test Engineer: Hamza Bin Asif

Date: 13-01-2025

Test Scenario: **Adding New Animal**

Test Case Description: To verify that a farmer can add a new cow's profile by correctly filling out all required fields and that the system validates and saves the data accurately.

Test Cases for Adding New Animal

	Steps	Input Data	Expected Result	Actual Result	Pass/Fail
TC-08	Leave all fields empty and click submit	Empty fields	"Next" button disabled until all required fields are filled.	"Next" button disabled until all required fields are filled.	Pass
TC-09	Enter only cow id and try to submit	Name: "254", other fields empty	"Next" button disabled until all required fields are filled.	"Next" button disabled until all required fields are filled.	Pass
TC-10	Enter invalid date format in DOB field	DOB: "32-13-2025"	Error: "Invalid date format."	The app uses calendar for selection of date so invalid date format can't be filled in anyway.	Pass
TC-11	Enter past date in correct format	DOB: "01-05-2020"	Date accepted	Date accepted	Pass
TC-12	Enter duplicate tag number	Tag: "COW001" (already exists in DB)	Error: "Tag already exists."	Error: "Tag number already registered."	Pass
TC-13	Enter, unique tag, valid DOB, and select breed	Tag: "COW007", DOB: "10-04-2022", Breed: "Sahiwal"	Cow profile added successfully. Redirect to home	Cow profile added successfully. Redirect to home	Pass

Test Cases (for screen/reports)

Project Name: FarmwareIteration No: 01Test Engineer: Hamza Bin AsifDate: 13-01-2025

Test

Scenario: Symptoms Selection

Test Case Description: To verify that farmers can select symptoms using toggle buttons and submit daily health data for a cow, ensuring that selected symptoms are saved correctly and reflected in the system.

Test Cases for Symptom Selection

TC ID	Steps	Input Data	Expected Result	Actual Result	Pass /Fail
TC-14	Open daily health form without selecting a cow	No cow selected	Error: "Please select a cow before proceeding."	When navigating to disease prediction page through sidebar, it asks to choose cow.	Pass
TC-15	Select a cow but submit form without selecting any symptoms	Cow selected, symptoms not selected	Error: "Select at least one symptom to proceed."	The form will not proceed.	Pass
TC-16	Select multiple symptoms using toggle buttons and submit	Cow: "123", Symptoms: "Cough", "Lethargy"	Data saved successfully. Confirmation message displayed.	Data saved successfully. Confirmation message displayed.	Pass
TC-17	Deselect all symptoms after selecting and try submitting	Cow: "124", Symptoms selected and then all deselected	Error: "Select at least one symptom to proceed."	The form will not proceed.	Pass
TC-18	Select symptoms and disconnect internet before submitting	Symptoms: "Diarrhea", No internet	Error: "No internet connection. Please try again later."	Error: "No internet connection. Please try again later."	Pass
TC-19	Select symptoms and submit, then revisit form to ensure they persisted	Cow: "234", Symptoms: "Cough", "Lethargy"	Previously selected symptoms are still shown for that day	Previously selected symptoms are still shown for that day	Pass

Test Cases (for screen/reports)

Project Name: FarmwareIteration No: 01Test Engineer: Hamza Bin AsifDate: 13-01-2025Test Scenario: **Disease Prediction Based on Selected Symptoms**Test Case Description: To verify that the system correctly analyzes selected symptoms and generates accurate disease predictions within the expected time limit.

Test Cases for Predictions

TC ID	Steps	Input Data	Expected Result	Actual Result	Pass /Fail
TC-01	Attempt disease prediction without selecting any symptoms	Cow selected, symptoms not selected	Error: "No symptoms selected. Please select symptoms to get predictions."	The form won't proceed further	Pass
TC-02	Select common symptoms and request prediction	Cow: "567", Symptoms: "Cough", "Fever"	Prediction generated within 15 seconds showing probable disease	Prediction generated in 12 seconds showing probable disease	Pass
TC-03	Select rare/uncommon symptoms and request prediction	Cow: "123", Symptoms: "Swelling", "Infertility"	System displays lesser-known possible diseases or alerts for rare case	System tries to predict the disease but also gives a warning in the end to consult a doctor	Pass
TC-04	Trigger disease prediction with intermittent internet connectivity	Cow: "123", Symptoms selected, unstable connection	Error: "Connection lost. Please retry." or retry automatically after delay	Displayed "Retrying..." and successfully submitted after reconnection	Pass
TC-05	Select symptoms, request prediction, and check result format	Cow: "123", Symptoms: "Cough", "Lethargy"	Clear, user-friendly result format with icons and summary	Prediction displayed with icons and summary cards for each disease	Pass

Test Cases (for screen/reports)

Project Name: Farmware

Iteration No: _01

Test Engineer: Muhammad Mujtaba

Date: _13-01-2025

Test Scenario: **Health Recommendations Display**

Test Case Description: To verify that the system displays appropriate, clear, and actionable health recommendations after predicting diseases based on selected symptoms.

Test Cases for Health Report

T C ID	Steps	Input Data	Expected Result	Actual Result	Pass/Fail
TC - 01	View recommendations after successful disease prediction	Cow: "234", Predicted Disease: "Mastitis"	Display advice: e.g., isolate cow, monitor milk quality, contact veterinarian	Displayed correct advice for Mastitis	Pass
TC - 02	Trigger prediction and view if advice uses easy-to-understand terms	Cow: "44", Disease: "Lameness"	Display in layman's terms (e.g., "Give soft bedding", "Limit cow's movement")	Clear, farmer-friendly language used	Pass
TC - 03	Check if recommendations are visually accessible (font/icons/colors)	Any cow/disease	Advice should be in readable font, with colored tags/icons if applicable	Recommendations were visually clear and readable	Pass

Test Cases (for screen/reports)

Project Name: Farmware

Iteration No: 01

Test Engineer: Muhammad Mujtaba

Date: 13-01-2025

Test Scenario: **Cow Health Profile**

Test Case Description: To ensure users can successfully view the complete health history and disease records of individual cows, with accurate data presentation and proper date-wise tracking.

Test Cases for Cow Health Profile

TC ID	Steps	Input Data	Expected Result	Actual Result	Pass/Fail
TC-01	Submit daily symptoms for a cow and trigger report generation	Cow: "12" + symptoms	Report generated including cow info, symptoms, prediction, and recommendations	Report generated correctly with all details	Pass
TC-02	Generate report for a cow with no symptoms selected	Cow: "12" (no symptoms)	Message: "No symptoms selected. Report not generated."	The form won't proceed if the symptoms are not selected	Pass
TC-03	View report for a previous date	Date: 08-Jan-2025	Report displays data from the selected date	Correct historical report shown	Pass
TC-04	Attempt to generate report without internet	Offline mode	Error: "Check your connection and try again."	Connection error message shown	Pass
TC-05	Generate report and verify timestamp and cow tag	Cow: "23"	Report includes date of submission and tag number	Timestamp and tag were correct	Pass
TC-06	Generate multiple reports and verify pagination	Multiple cows	Reports paginated or scrollable for better readability	UI handled multiple reports properly	Pass
TC-07	Verify accuracy of symptom list and disease result in the report	Cow: "76" + 3 symptoms	Report matches selected symptoms and predicted disease	Data accuracy confirmed	Pass

Test Cases (for screen/reports)

Project Name: FarmwareIteration No: 01Test Engineer: Muhammad MujtabaDate: 13-01-2025Test Scenario: **Report Generation**

Test Case Description: To verify that the system correctly generates a daily health report summary for each cow after symptom submission and disease prediction, displaying all relevant health metrics clearly.

Test Cases for Health Report Generation

TC ID	Steps	Input Data	Expected Result	Actual Result	Pass/Fail
TC-01	Submit daily symptoms for a cow and trigger report generation	Cow: "Bella" + symptoms	Report generated including cow info, symptoms, prediction, and recommendations	Report generated correctly with all details	Pass
TC-02	Generate report for a cow with no symptoms selected	Cow: "Mira" (no symptoms)	Message: "No symptoms selected. Report not generated."	The form won't proceed if the symptoms are not selected	Pass
TC-03	View report for a previous date	Date: 08-Jan-2025	Report displays data from the selected date	Correct historical report shown	Pass
TC-04	Attempt to generate report without internet	Offline mode	Error: "Check your connection and try again."	Connection error message shown	Pass
TC-05	Generate report and verify timestamp and cow tag	Cow: "Luna"	Report includes date of submission and tag number	Timestamp and tag were correct	Pass
TC-06	Generate multiple reports and verify pagination	Multiple cows	Reports paginated or scrollable for better readability	UI handled multiple reports properly	Pass
TC-07	Verify accuracy of symptom list and disease	Cow: "Daisy" + 3 symptoms	Report matches selected symptoms and predicted disease	Data accuracy confirmed	Pass

	result in the report				
TC-08	Check report formatting (alignment, fonts, sections)	N/A	Neatly formatted sections: cow info, symptoms, prediction, advice	Formatting was clean and sections well defined	Pass

Test Cases (for screen/reports)Project Name: FarmwareIteration No: 01Test Engineer: Hamza Bin AsifDate: 13-01-2025**Test Scenario:** Connectivity Handling**Test Case Description:** To ensure the system can detect and handle internet connectivity issues gracefully during operations such as symptom submission, report generation, or profile viewing—without data loss or app crashes.**Test Cases for Health Report**

T C ID	Steps	Input Data	Expected Result	Actual Result	Pass/Fail
T C-01	Open app with active internet	Internet ON	App loads and remains on home screen	App loads normally	Pass
T C-02	Disconnect internet and try to submit daily symptoms	Cow: "Molly" + 2 symptoms	App returns to home screen, does not allow submission	App returned to home screen	Pass
T C-03	View cow health profile with no internet	Select cow profile	App returns to home screen	App returned to home screen	Pass
T C-04	Generate report while internet is off	Open report section	App returns to home screen	App returned to home screen	Pass
T C-05	Toggle internet off while filling form	Mid-form	App detects disconnect, returns to home screen	App returned to home screen	Pass
T C-06	Open help section without internet	Help tab	App returns to home screen	App returned to home screen	Pass
T C-07	Reconnect to internet after disconnect	Reconnect internet	App requires manual refresh or relaunch to function again	App stays on home until refreshed	Pass
T C-08	Open app without internet	Launch app	App loads to home, remains idle or blocked until internet resumes	App blocks interaction, waits online	Pass
T C-09	Navigate to any feature while offline	Any section	App prevents access and returns to home	App returned to home screen	Pass
T C-10	Ensure app does not crash on disconnect	Disconnect abruptly	App stays stable (no crash), even if returned to home	App stable, no crash	Pass

UI/UX Detail Document

1. Introduction The UI/UX of the **Farmware** application is designed to offer a simple, intuitive, and accessible experience for farmers. The primary goal is to ensure that users with limited technical expertise can easily navigate the app to manage animal health efficiently. Emphasis is placed on **ease of use**, **clarity of information**, and **responsive design** to support daily farm operations across both mobile and desktop platforms.

2. Design Principles

- **Simplicity:** A clean and minimal user interface ensures farmers can perform tasks without confusion or distraction.
- **Consistency:** Consistent use of layout, color schemes, iconography, and typography across all screens improves usability.
- **Responsiveness:** The design is fully responsive, ensuring smooth operation on smartphones, tablets, and desktops used in diverse farming environments.
- **Accessibility:** Designed to be inclusive, adhering to WCAG 2.1 standards to support users with varying abilities.
- **Usability:** Clear, prominent call-to-actions and intuitive navigation make it easy for farmers to add animals, track symptoms, and view health records effortlessly.

3. UI Components

- **Login & Registration:** Simple and secure authentication process for farmers to access their personalized accounts.
- **Dashboard:** Central hub displaying key metrics like number of animals, recent health assessments, and alerts.
- **Animal Profile Management:** Interface for adding, editing, and viewing cow profiles including tag number, breed, and health history.
- **Symptom Selection Module:** Toggle-based interface for daily symptom reporting, optimized for quick input.
- **Disease Prediction & Recommendations:** Immediate feedback showing predicted diseases with tailored health advice.
- **Reports & Export Tools:** View, download, and export daily or historical health reports in PDF/CSV format.

4. User Flow

- **Farmer Flow:** Login/Signup → Dashboard → Add Animal Profile → Daily Symptom Selection → Submit Data → View Disease Prediction & Health Recommendations → Access Health Reports.
- **Admin Flow:** Monitor System Health → Manage User Accounts → Handle Support Requests → Generate System Usage Reports.

5. Tools Used

- **Design Tool:** Canva
- **Front-end Framework:** Ionic with Angular

Coding Standards Document

1. Introduction To ensure high code quality, consistency, scalability, and maintainability for the Farmware application, the following coding standards and best practices are followed throughout the development lifecycle.

2. General Guidelines

- Use clear, descriptive, and meaningful variable and function names.
- Maintain consistent indentation and spacing (2 or 4 spaces as per team preference).
- Avoid hardcoding values; use constants or configuration files.

3. Frontend Coding Standards

- Use **Angular with Ionic** for structured modular development.
- Adhere to **Angular Style Guide** (component-based architecture).

4. Backend Coding Standards

- Follow RESTful API conventions.
- Use MVC (Model-View-Controller) architecture.
- Implement error handling and logging.

5. Database Standards

- Use normalized relational database schema.
- Use indexing for performance optimization.
- Maintain foreign key constraints for data integrity.

6. Security Practices

- Encrypt sensitive data with bcrypt.
- Validate user inputs to prevent SQL injection.
- Use JWT for authentication.

Project Policy Document

1. Introduction Farmware is a mobile/web application designed to assist farmers in managing animal health by logging symptoms, predicting diseases, and providing tailored recommendations.

2. Getting Started

☐

☐

☐ Download the Farmware app.

Log in using your registered farmer credentials or create a new account.

After login, you can begin managing your animals' health records.

3. Features

For Farmers:

- **Add New Animal:** Input cow details like name, tag, DOB, and breed.
- **Daily Health Check:** Select symptoms from an intuitive toggle-based interface.
- **Disease Prediction:** Receive likely disease outcomes based on symptom data.
- **Health Recommendations:** Get advice tailored to the animal's condition.
- **Health Profile:** View complete health history and disease records of each cow.
- **Generate Reports:** Create and export daily reports in PDF or CSV formats.

4. Troubleshooting & Support

- Forgot password? Use the '**Forgot Password**' feature to reset it.
- Facing issues? Visit the **Help Section** for FAQs, tutorials, or contact support.
- Ensure a **stable internet connection** for full app functionality.
- Update the app regularly for improved performance and new features.

User Manual Document

1. Introduction Farmware helps farmers monitor livestock health by simplifying daily health tracking, disease prediction, and report generation..

2. Getting Started

- Open the Farmware app.
- Log in or sign up with your mobile number and create your farmer profile.
- Begin by adding at least one animal to start health monitoring.

3. Features

For Farmers:

- **Animal Profile Creation:** Add cow details such as name, tag number, date of birth, and breed.
- **Symptom Logging:** Select symptoms from toggle buttons each day to assess animal health.
- **Automatic Disease Prediction:** Based on logged symptoms, the system predicts potential diseases.
- **Recommendations:** After prediction, receive customized health tips and actions to take.
- **Health History:** Track and view each animal's past diseases, treatments, and symptoms.
- **Reports:** Generate and export daily summaries in user-friendly formats (PDF, CSV).

4. Troubleshooting & Support

- Use '**Forgot Password**' to recover your account.
- For help, go to the **Help Section** to find FAQs, video tutorials, or send a support message.
- Make sure your **internet connection** is active during use.
- For best experience, keep your app updated to the latest version.

A5. FLYER & POSTER DESIGN



A6. COPY OF EVALUATION COMMENTS

Aijaz Ali	informal behavior while presenting , picking phone call while presenting ,lack of subject knowledge, make sure to avoid these mistakes next time .
Aamir Hussain	Need to work on their documentation and also need in-depth study on the topic
Umer Farooq	Supervisor needs to concentrate on scope
Saeed Ahmed	Focus on documentation, Presentation skills poor,

A7. MEETING MINUTES AND SIGN-OFF SHEET

FYP Fortnightly Sign-Up Sheet

Course: ☐ FYP-1 ☒ FYP-2 Project Code: FYP-013/FL24 Project Name: Paranumare App

Group Members Names & Reg#: Maryam Nadeem (2437-2021) Muhammad Mujtaba (1985-2020) Hamza Bin Asif (1961-2021)

Supervisor Name: Fahem Ahmed Co-Supervisor's Name: _____

Meeting #	Date	Agenda (Brief Statement)	Attended By (Student's Name only)	Supervisor's Sign	Co-supervisor's Sign	FYP Officer's Sign
1	6/3/25	• Data accuracy improvement • Introduction of Project with new supervisor • Discussed the sources of data collection	Maryam Nadeem Muhammad Mujtaba Hamza Bin Asif			
2	20/3/25	• Discussion of model accuracy. • Automate Questionnaire Generation.	Maryam Nadeem Muhammad Mujtaba Hamza Bin Asif			
3	9/4/25	• Discussion of changing the input method and about model accuracy.	Maryam Nadeem Muhammad Mujtaba Hamza Bin Asif			
4	17/4/25	• Discussion Changes in Machine Learning Model.	Maryam Nadeem Hamza Bin Asif Muhammad Mujtaba			
5	28/5/2025	selection of cloud server service	Maryam Nadeem Hamza Bin Asif Muhammad Mujtaba			
6	21/5/2025	Discussion of cloud server service	Maryam Nadeem Hamza Bin Asif Muhammad Mujtaba			
7						
8						
9						

P-Test Report Results

Farmware-Report-1.pdf

ORIGINALITY REPORT

13%	10%	3%	11%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Higher Education Commission Pakistan Student Paper	7%
2	Submitted to University of Wales Institute, Cardiff Student Paper	1%
3	pt.scribd.com Internet Source	<1%
4	Submitted to Harper Adams University College Student Paper	<1%
5	www.coursehero.com Internet Source	<1%
6	123dok.com Internet Source	<1%
7	digitalcollection.utem.edu.my Internet Source	<1%
8	Mohamed Abdel-Basset, Hossam Hawash, Laila Abdel-Fatah. "Artificial Intelligence and	<1%

Internet of Things in Smart Farming", CRC Press, 2024

Publication

9	vdocuments.site Internet Source	<1 %
10	Submitted to IIMT University Student Paper	<1 %
11	Submitted to Arab Open University Student Paper	<1 %
12	Submitted to ESoft Metro Campus, Sri Lanka Student Paper	<1 %
13	public.j.ebilib.com Internet Source	<1 %
14	www.cs.utexas.edu Internet Source	<1 %
15	123dok.net Internet Source	<1 %
16	Submitted to Letterkenny Institute of Technology Student Paper	<1 %
17	Submitted to South Bank University Student Paper	<1 %
18	Submitted to Sydney Institute of Technology and Commerce Student Paper	<1 %

19	Submitted to TMC Education Group Student Paper	<1 %
20	akvateq.com Internet Source	<1 %
21	allmeld.com Internet Source	<1 %
22	Submitted to Harrisburg University of Science and Technology Student Paper	<1 %
23	Submitted to Polytechnics Mauritius Student Paper	<1 %
24	actaenergetica.org Internet Source	<1 %
25	blog.zanobini.com Internet Source	<1 %
26	dora.dmu.ac.uk Internet Source	<1 %
27	fastercapital.com Internet Source	<1 %
28	guyanachronicle.com Internet Source	<1 %
29	ideascale.com Internet Source	<1 %

www.gov.mu

30	Internet Source	<1 %
31	www.minimum.com Internet Source	<1 %
32	link.springer.com Internet Source	<1 %
33	mi-dnu.dp.ua Internet Source	<1 %
34	scholarworks.boisestate.edu Internet Source	<1 %
35	silo.tips Internet Source	<1 %
36	www.e-consolutions.com Internet Source	<1 %
37	www.researchgate.net Internet Source	<1 %
38	Noman Islam, Zubair Ahmed Shaikh, Aqeel-ur Rehman, Muhammad Shahab Siddiqui. "HANDY: a hybrid association rules mining approach for network layer discovery of services for mobile ad hoc network", Wireless Networks, 2013 Publication	<1 %
39	adchitects.co Internet Source	<1 %

40	gsconlinepress.com Internet Source	<1 %
41	listens.online Internet Source	<1 %
42	site-2003-2017.actupparis.org Internet Source	<1 %
43	Submitted to universititeknologimara Student Paper	<1 %
44	catalog.data.metro.tokyo.lg.jp Internet Source	<1 %
45	mmcalumni.ca Internet Source	<1 %
46	www.amazon.com Internet Source	<1 %
47	Aditya Nandan Prasad. "Introduction to Data Governance for Machine Learning Systems", Springer Science and Business Media LLC, 2024 Publication	<1 %
48	Submitted to Institute of Research & Postgraduate Studies, Universiti Kuala Lumpur Student Paper	<1 %