Smart Livestock Disease Prediction

AQSA SAJID
01-134221-013
SAMEEN SAEED
01-134221-069

Supervisor: Dr. Hafiz Ishfaq Ahmad

Bachelor of Science in Computer Science

Department of Computer Science
Bahria University, Islamabad

December 2025

Certificate

We accept the work contained in the report titled “Smart Livestock Disease Prediction”, written by Aqsa Sajid and Sameen Saeed as a confirmation to the required standard for the partial fulfillment of the degree of Bachelor of Science in Computer Science.

Approved by . . . :

Supervisor:

_____

Internal Examiner:

_____

External Examiner:

_____

Project Coordinator:

_____

Head of the Department:

_____

Abstract

The productivity of agriculture, food security as well as the rural livelihood is at the risk of ever-increasing outbreaks of livestock diseases across the world, especially in developing countries where veterinary services are not readily accessible. Traditional methods of diagnoses, which are time consuming and prone to error, are still what farmers have to depend on, leading to missed or wrong treatment, causing not just monetary loss but also increased spread of diseases. The challenges are more difficult to overcome due to the lack of timely and accurate diagnostic technologies, particularly in remote and marginalized farming areas.

This work addresses these issues by presenting the architecture and design of a Smart Livestock Disease Prediction System based on Artificial Intelligence and Mobile Technology to assist in the early and automated detection of cattle diseases. The system is based on a MobileNetV2 convolutional neural network which has been fine-tuned on a trade dataset by using transfer learning, trained on a dataset comprised of region specific cattle diseases, allowing real time image based classification on a mobile app. The method proposed takes an agile development approach that includes the steps of gathering and preprocessing the data, training the model and deploying it in a React Native mobile app with Node.js backend and MongoDB database.

Farmers, through the mobile app, can take photos or upload existing rural livestock images, and get instant predictions with confidence scores, descriptions of disease, treatment suggestions, and precautions. Veterinarians enhance system accuracy by confirming the AI generated diagnosis and modifying the disease knowledge base. It also includes an interactive dashboard illustrating livestock health, disease incidence patterns and climatological data to inform early decisions. Connecting farmers directly with expert veterinary advice through an easy to use, AI enabled platform, the system improves disease monitoring, encourages early action, and advocates for sustainable management of livestock. The initiative demonstrates that it is feasible to deploy lightweight deep learning models in resource constrained environments characteristic of the agriculture sector, and advances the emerging field of digital agriculture.

Acknowledgments

All praise is due to Allah, the Lord of the worlds. His continuous infinite blessings, guidance, and mercy helped us in bringing this project, *Smart Livestock Disease Prediction* to a fruitful end. Without His blessings and support, it was impossible to achieve this feat.

We thank our parents very much for their unconditional love, prayers and support. Their continuous support and faith in us empowered us with the will to keep going and conquer every challenge along the way.

First and foremost we would like to express our gratitude to our honorable supervisor Dr. Hafiz Ishfaq Ahmad for his precious guidance, eye opening suggestions and relentless co operation. His mentorship was always educational and inspiring, and his support was pivotal to the success of this project.

Finally, we want to thank all those who helped, in one way or another, in the realization of this work. This project's success is due to the dedication and collective support of all those who accompanied us on this journey.

AQSA SAJID
SAMEEN SAEED
Bahria University
E-8, Islamabad, Pakistan

December 2025

Contents

List of Figures

List of Tables

Acronyms and Abbreviations

DSA	Data Structure and Algorithms
OOP	Object Oriented Programming
PF	Programming Fundamentals
SE	Software Engineering
SQL	Structured Query Language
UNESCO	United Nations Educational, Scientific and Cultural Organization
UNICODE	Unique, Universal, and Uniform Character enCoding
XML	Extensible Markup Language

Chapter 1

Introduction

The health of livestock plays a crucial role in agriculture, directly impacting food production, rural livelihoods, and economic development. However, farmers often face significant challenges in maintaining animal health due to delayed disease detection, limited access to veterinary services in remote areas, and reliance on traditional monitoring methods based primarily on visual observation [1]. These limitations contribute to frequent disease outbreaks, reduced productivity, and substantial financial losses within farming communities.

This thesis introduces the *Smart Livestock Disease Prediction System* that uses AI and deep learning for the detection of most common cattle diseases such as Bovine Respiratory Disease (BRD), Lumpy Skin Disease, Contagious Ecthyma, and Dermatitis from images. In contrast to traditional methods, the system is based on an automated real time detection, allowing the farmers to take early decisions on the health care of their animals. The system has been developed as a mobile app, so farmers and vets can take or upload pictures of animals and get immediate disease prediction. Potentially catering for a very broad audience, including those with virtually no technical expertise (or rural users) the application was developed to be as minimal and practical as possible.Through early diagnosis, the system expects to decrease the spread of the disease, mortality rates and increase efficiency in livestock handling.

1.1 Project Background

In Pakistan, livestock is a crucial asset and income source for rural households and also an important commodity in developing countries. Disease control is hard to achieve even though it is important. Farmers usually rely on visual diagnoses or the advice of a veterinarian after the fact, such consultations leading to delayed diagnoses, the disease spreading rapidly among herds, and major financial losses. Recently, the field of AI/ML

has seen exponential growth in agriculture services such as crop monitoring and yield prediction. However, these techniques have not been as extensively applied to detection of disease in animals. Conventional diagnostic procedures are slow, expensive and often farmers in rural areas have no access to them. This work fills these voids by utilizing AI driven image analysis via the MobileNetV2 deep learning architecture. Known for its high performance and low complexity benefiting mobile and resource limited environments [2], MobileNetV2 allows the system to process the image of a livestock and provide real time predictions of the disease on mobile. The presented technology adds practical, affordable and scalable strength to the diagnostic process enabling smallholder farmers.

1.2 Problem Description

Livestock production contributes significantly to meeting the global demand for adequate and nutritious food, supports rural livelihoods, and drives economic development. The sector also is an important source of agricultural gross domestic product in Pakistan and provides millions of small holders with a source of livelihood. In the face of such preciousness, maintaining the health of animals is no easy feat when the tools for diagnosing diseases are outdated, slow, and inaccessible. Farmers' common observation methods are very subjective. With limited farmer experience and informal advice, diagnoses in remote areas, like those we are discussing, are at risk for inaccurate treatments and similar outcomes, more often than not [1]. Bovine Respiratory Disease (BRD), Lumpy Skin Disease, Ecthyma, Dermatitis are among the diseases which are difficult to detect in early stages as signs like low grade fever, skin lesions and reduced feed intake are missed. Since these diseases can rapidly spread through the flock, they result in decreased milk production, poor growth, and high mortality animals. These events mean that what happens is a reduction in the income of the farmers and the threat to the food security and the public health (Zoonotics etc.) which may be brought by such outbreaks. Another major hurdle is the paucity of trained veterinarians in rural and semi urban localities. Farmers must sometimes journey great distances for diagnoses, and the time and costs involved are considerable. However, these tests tend to be very expensive and farmers cannot easily afford them. Such limitations create a deadly interval between disease outbreak and detection in the time when the affected animal begins destroying other livestock.

Poor sanitation, climate variability, and uncontrolled movement of animals are some of the adverse environmental conditions that are contributing to the problem. The lack of digital records and real time tracking impedes disease surveillance and early warning systems. Conventional methods do not provide scalable, data enabled solutions to manage livestock diseases effectively. In order to overcome these challenges, in this work, we present a Smart Livestock Disease Prediction System based on deep learning for image based identification of cattle diseases using MobileNetV2 architecture. The parallel mobile

app allows farmers to take pictures of their animals or upload existing images to receive instantaneous predictions on the health of the animal along with confidence scores and suggested courses of action. This technology closes the gap in terms of technological availability, providing a fast, cost effective, and simple method for laboratory tests. The integration of AI and mobile technology in the solution strengthens disease surveillance, reduces economic loss, and sustains livestock related activities, especially in resource poor rural regions.

1.3 Objectives

The primary objectives of the Smart Livestock Disease Prediction System are:

- To establish a MobileNetV2-based deep learning model for recognizing typical cattle diseases through images.
- To utilize Artificial Intelligence (AI) methodologies for precise and early detection of diseases, enabling appropriate treatment in time.
- The goal is to develop an intuitive and easy to use mobile application that helps farmers in diagnosing possible livestock diseases.

1.4 Project Scope

The Livestock Disease Prediction System leverages the MobileNetV2 model to analyze images of cattle and predict common illnesses such as Bovine Respiratory Disease (BRD), Dermatitis, Lumpy Skin Disease, and Contagious Ecthyma. Developed as a user-friendly mobile application using React Native, it enables farmers and veterinarians to capture or upload photos for instant, accurate disease assessments. This facilitates early detection and prompt treatment, ultimately saving livestock and enhancing herd health. Designed especially for small and medium-scale farmers in remote regions with limited access to veterinary care, the system combines a Node.js/Express.js backend with MongoDB for secure data storage and management. By integrating AI with mobile technology, the project delivers an affordable, scalable solution that strengthens disease surveillance and modernizes livestock health practices. Its offline functionality ensures reliable use in areas with poor internet connectivity, while continuous model updates improve diagnostic precision over time. Ultimately, this innovation empowers farmers with critical insights, reducing economic losses and promoting sustainable agricultural productivity.

1.5 Summary

This chapter has presented the Smart Livestock Disease Prediction System along with its motivation, background and goals. It has overcome the major obstacles experienced by livestock producers such as late disease detection, poor access to veterinary and reliance on manual observation. The chapter has also described the way the proposed system uses Artificial Intelligence and deep learning (MobileNetV2 model) to detect cattle disease by analyzing images. Finally, the scope of the project has been defined to show that this mobile solution utilizing AI provides aid to rural farmers through early detection to minimize the economic loss and help managing their livestock in general.

Furthermore, this system represents a significant step towards the digital transformation of traditional livestock management. By bridging the gap between advanced artificial intelligence and everyday farming practices, it democratizes access to veterinary-level diagnostic tools. The integration of a responsive React Native frontend with a robust Node.js backend ensures the solution is not only technologically sound but also practical and accessible for its target users. As the system evolves, it holds the potential to incorporate additional features such as outbreak tracking, treatment recommendations, and integration with broader agricultural data platforms. This chapter, therefore, lays the foundation for a proactive, data-driven approach to animal healthcare, promising to enhance resilience, productivity, and sustainability within the livestock sector.

Chapter 2

Literature Review

2.1 Overview

Managing livestock diseases remains a significant challenge in agriculture, particularly in developing countries like Pakistan, where access to veterinary services is limited. Traditional disease detection methods primarily depend on manual inspections by farmers or veterinarians, which are often slow, subjective, and susceptible to human error [1]. This reliance results in delayed diagnoses, increased mortality rates, and financial losses due to the spread of infectious diseases within herds [3].

Recent advancements in Artificial Intelligence (AI) and Computer Vision have opened up new avenues for enhancing agricultural practices. Deep learning models, such as Convolutional Neural Networks (CNNs), MobileNet, and EfficientNet, have demonstrated considerable potential for image based disease detection. These technologies can automatically extract and classify features from images, thereby improving the speed and accuracy of diagnoses [4]. Among these models, MobileNetV2 is particularly recognized for its lightweight architecture, making it well suited for mobile devices an essential feature for farmers in rural areas [2]. The evolution of transfer learning techniques has further enabled the adaptation of pre trained models for specific agricultural applications with limited datasets [5].

2.2 Existing Systems

AI has been widely applied in agriculture for purposes like detection of crop diseases, animal tracking and precision farming, among others. However, AI based systems for animal disease detection are still lacking [6]. Existing solutions are often fragmented, focusing on single diseases or requiring expensive, specialized hardware unsuitable for small-scale farmers. Some research prototypes utilize convolutional neural networks (CNNs) for analyzing livestock imagery but remain confined to academic studies or

controlled environments, lacking integration into a practical, field-deployable application. Furthermore, many available tools are designed for desktop use or rely on consistent high-speed internet, creating a significant accessibility barrier in remote rural areas where such challenges are most acute. This gap highlights the critical need for an integrated, mobile-first AI system specifically tailored for on-farm, real-time livestock disease prediction.

2.2.1 Global Agricultural Applications

AI has been successfully employed in many fields of agriculture, such as in monitoring crop health and precision agriculture. These global programs open the path for exploring how analog technologies can be used for managing the health of livestock [7].

2.2.1.1 Plant Disease Detection Systems

There is worldwide research on development of AI based plant disease detection systems using CNN models for detecting crop diseases from images of leaf. For example, Mohanty et. al. proposed a model that uses a large scale crop leaf image dataset for classifying various plant diseases [8]. These platforms have been shown to deliver excellent results in crop protection and yield prediction [9].

2.2.1.2 Precision Livestock Farming Tools

In addition to plant health, AI and IoT based solutions have also been developed for animal monitoring [10]. These animal monitoring systems leverage wearable sensors and IoT devices to continuously monitor the condition and behavior of animals. For example, the review "Wearable IoT enabled Precision Livestock Farming" reports some sensor based systems that collect physiological and surrounding information. Although these instruments provide useful information, they depend on specialized instruments, uninterrupted power, and reliable internet access, which render them prohibitively expensive and impractical for smallholder farmers in developing countries.

2.2.2 Regional Initiatives (South Asia and Africa)

After studying global systems, we must now look at how the developing areas such as South Asia and Africa have adapted these kinds of technologies to improve animal husbandry [11].

2.2.2.1 Mobile Veterinary Services

Several mobile veterinary services have emerged that are region specific, working towards helping farmers in the rural areas [12]. Farmers are able to connect with a vet and submit

animal symptoms from a distance through platforms such as VetAfrica in Ethiopia [13]. The two services improve farmer-veterinarian communication [14].

2.2.2.2 Pakistan Specific Solutions

In Pakistan, most of the available livestock management tools focus on maintaining records, vaccination alerts, and feeding schedules [15]. Although these products are good for managing, these products don't come with elaborate AI and CV based disease detection [16]. As a result, farmers still rely on manual observation, which results in delays and makes it harder to contain disease outbreaks [17].

2.2.3 Existing Manual Systems

Although there has been progress in technology, in many developing countries, disease detection is still based on manual work [18]. Farmers and veterinarians are prone to misdiagnoses and treatment delays based on visual assessments and personal experience [19]. Many diseases have only minimal early signs, which are easily ignored [20]. In addition, there is a scarcity of right qualified veterinarian in rural/remote areas, most of which are under developed in terms of infrastructure[21]. Consequently, diseases may spread widely pre-detection, causing significant production losses and high mortality [22].

2.2.4 Academic Research

A host of academic studies have been conducted and the potential applications of technology to control livestock diseases and to farm management through agricultural digitisation are well documented.

2.2.4.1 Okello et al. (2019)

Okello et al. investigated the economics of livestock management and animal health surveillance, with an emphasis on statistical surveys as opposed to deep learning or image based methods [1]. Their studies together highlight the economics of impact of animal (but not plant) diseases, but do not provide a real time image analysis method.

2.2.4.2 Recent Surveys of AI in Agriculture

Recent surveys have shown that farmers are increasingly adopting AI technology. For example, they also pointed that application of AI in livestock health management is still limited by the absence of data sets, difficulty in performing field tests, and requirement for mobile models that can be used via mobile phones. A few general reviews have also highlighted distinctive technical issues in agriculture AI implementation [23].

2.3 Gaps in Existing Systems

After reviewing the existing literature and technologies, a number of gaps have been identified in the area of cattle disease detection [24]. Such gaps highlight the need for novel and improved approaches, like the present Smart Livestock Disease Prediction system.

- **Domain Gap:** Most AI applications are focused on crops, while livestock disease detection has received little attention.
- **Accessibility Gap:** IoT based solutions depend on expensive hardware that is not affordable for small farmers.
- **Localization Gap:** Very few systems are developed specifically for the livestock diseases common in Pakistan and South Asia [25].
- **Technical Gap:** Many existing models are too large to run on mobile devices, and lightweight architectures like MobileNetV2 are still underused in agricultural applications.
- **Data Gap:** There is a shortage of labeled livestock disease images, which limits the accuracy and performance of AI models.
- **Validation Gap:** Limited real world testing and validation of proposed systems in actual farming conditions.

2.4 Advancements in Livestock Digitization

Recent developments in technology have made the integration of AI, mobile applications, and data analytics for the management of the livestock health possible. The following advances have been instrumental in the development of present day disease detection systems.

- **AI and Machine Learning Integration:** Deep learning models such as MobileNetV2, EfficientNet, and ResNet are increasingly being used for real time disease detection in agriculture.
- **Mobile Accessibility:** The use of lightweight models on smartphones enables on site disease diagnosis without depending on laboratory testing.
- **Scalability:** Cloud based storage allows continuous improvement and retraining of models as more data becomes available.
- **Decision Support:** AI generated predictions can help farmers by providing preventive suggestions and treatment recommendations after detection.

Table 2.1: Smart Livestock Disease Prediction System and Features Addressing Identified Gaps

Core Feature	Implemented Solution	Gap Resolved
Lightweight AI Model	MobileNetV2 fine tuned on livestock disease images [2]	Enables diagnosis on mobile devices without heavy hardware
Mobile Application Access	React Native mobile app for farmers and veterinarians	Provides convenient and user friendly access to detection tools
Localized Dataset	Training on region specific cattle diseases such as BRD, Lumpy Skin Disease, Dermatitis, and Ecthyma	Addresses the lack of regional adaptation for South Asian livestock
Scalable Architecture	Node.js backend with MongoDB and retraining capability for new data	Ensures continuous improvement and model adaptability
Real time Processing	Optimized inference pipeline for quick diagnosis	Reduces delay compared to manual methods
Multi role System	Separate interfaces for farmers, veterinarians, and administrators	Addresses diverse user needs and expertise levels

2.5 Key Innovations and How This Project Closes Critical Gaps

After the literature review and challenges identification, this part illustrate the technique how the proposed Smart Livestock Disease Prediction System eliminates the weakness of previous methods.

As shown in Table 2.1, the proposed system directly addresses key gaps in livestock health management. It utilizes a lightweight MobileNetV2 model for on device diagnosis, a React Native app for accessibility in rural areas, and a localized disease dataset for regional applicability. A scalable backend facilitates ongoing enhancement, and real time processing diminishes diagnostic delays. Lastly, a multi role interface enables farmers, vets, and administrators to access tailored functionalities.

2.6 Summary

Platforms such as VetAfrica and various plant disease detection systems demonstrate the potential for AI to assist in agriculture, but they fall short in catering to the specific requirements of livestock farmers. Many of these are either prohibitively expensive, hardware dependent, or simply not relevant to local conditions, and so farmers in rural areas are unable to use them.

The Smart Livestock Disease Prediction System tackles these issues by deploying a lightweight MobileNetV2 model inside a mobile app based on localized livestock disease datasets. This novel approach ensures all the farmer in Pakistan and under developed world can rely for early detection of livestock disease and management of animal health.

Utilizing cutting edge AI technology, the system allows farmers to monitor real time data and make informed decisions to minimize the financial losses from livestock diseases. In addition, the application is easy to use, enabling farmers with limited technological knowhow to make the most of the product.

In addition, the method enhanced the farmer-veterinarian relationship with better communication and disease control support. The system represents a major step in bridging the digital divide in agriculture and bringing state of the art AI technology to farming communities that have been historically under resourced. The system's modular design allows for future expansion to additional livestock species, additional diseases, and potentially integration with other agricultural management systems. As agricultural systems worldwide are increasingly threatened by the impact of climate change and the spread of new diseases, such technology driven solutions will be essential in addressing food security and sustainable development.

In addition, the system is designed with a strong emphasis on sustainability and community impact. Its low computational footprint ensures compatibility with affordable smartphones, extending its reach to the most resource-limited farming households. By enabling proactive herd management, the tool not only safeguards animal welfare but also contributes to stabilizing farmer incomes and securing local food supplies. The successful deployment and validation of this system in rural Pakistan can serve as a replicable model for other developing regions facing similar veterinary resource gaps. Ultimately, this project establishes a scalable framework for deploying edge AI in agriculture, demonstrating how tailored technological innovation can directly empower communities, strengthen rural economies, and build resilience against evolving biological and environmental challenges.

Chapter 3

Methodology

3.1 Overview

The design of the Smart Livestock Disease Prediction System was a well defined, organized, and repetitive process to make sure the system was effective, easy to use, and dependable. The whole approach is divided into two main parts; the first part describes the technical mobile workflow on how to build a MobileNetV2 based livestock disease detection model and the second part describes the project management aspect using the Agile process. Each stage of the project, starting with dataset acquisition and ending with mobile application distribution, was carefully designed to ensure smooth processing and the uniformity of the model.

3.2 Proposed Methodology

The Agile methodology was chosen for this project because it values flexibility, collaboration, and continuous process enhancements while working on software development [**?**]. The work was divided into mini sprints, where each sprint has a particular goal, e.g. data preprocessing, model training, developing mobile app.At the end of each sprint the deliverables were reviewed, comments were collected from the supervisor and the team members, and required enhancements were made before moving on to the next stage. This iterative approach helped identify problems early and ensure incremental progress during the project. Figure 3.1 depicts the methodology in the project.

3.3 Data Acquisition

The dataset used for this project included images of healthy and sick cattle with different diseases like Lumpy Skin Disease, Dermatitis, Bovine Respiratory Disease (BRD), and Contagious Ecthyma. These images were taken from public online repositories and

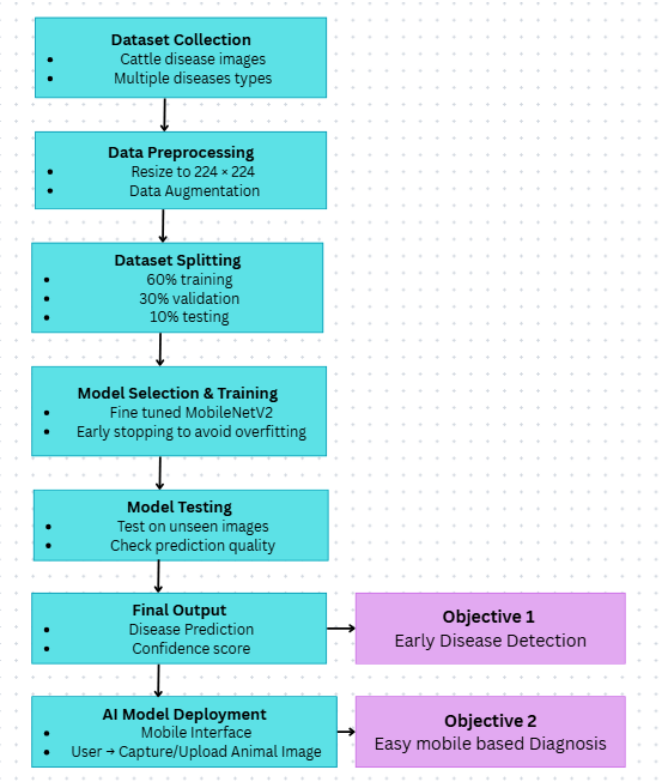


Figure 3.1: Proposed Methodology

authentic web sources [1]. At the beginning, it is noticed that the dataset is unbalanced (many images of healthy cattle or of Lumpy Skin Disease). Therefore, the model was biased towards those classes. To solve this problem, more samples of the minority diseases were collected from the trusted sources which formed a balanced dataset. This change allowed the model to be more evenly distributed among the disease types, improving the overall accuracy of the model.

3.4 Data Preprocessing

Before training the model, the captured images were preprocessed so that data was clean and uniform. All images were rescaled to 224x224 size to fulfill input size requirement of MobileNetV2 model. Various data augmentation methods, such as rotation, flipping, zooming, and changing brightness, were used to increase variance and prevent overfitting. Following the dataset balancing, the new images were also preprocessed, as described, prior to retraining. This strategy guaranteed that the model had the opportunity to learn well and fairly from all the disease classes. Additionally, pixel values were normalized to a range of [0,1] to accelerate convergence during training. The dataset was then split into training, validation, and test sets to rigorously evaluate the model's performance and generalization capability.

3.5 Model Development

The disease identification model was built based on MobileNetV2 architecture, which is known to be lightweight and most suitable architecture for mobile application. MobileNetV2 uses inverted residuals and linear bottlenecks [2], which provide the capacity to capture important features with low computations. The dataset was split into three parts: 60% for training, 30% for validation, and 10% for testing. The Adam optimization algorithm was used for updating the weights and Categorical Cross Entropy was used as the loss function for multi class classification. Accuracy, precision, recall, and F1 score were among the measures used to evaluate performance. The trained model results in better and fair predictions across all disease types with the balanced and augmented dataset.

3.6 System Design and Implementation

Smart Livestock The Disease Prediction System is a mobile application that provides compatibility with Android and iOS platforms developed using React Native. For real time detection of livestock diseases, the application embeds a trained MobileNetV2 model.
The system backend is developed using Node.js and Express.js that facilitates the mobile app to communicate with the server smoothly. MongoDB is used as database to store user details and disease associated the records for efficient data management.
The system's user interface is straightforward and user friendly, allowing even farmers with very little technical knowhow to easily use the app.

3.7 Mobile Application Interface and Functionality

The app is developed for easy use by both farmers and veterinarians. It provides a number of important features such as disease identification, user administration and livestock health monitoring.The following is an overview of the key elements of the app:

3.7.1 User Authentication

The whole system start with a login and signup screen. New users can sign up with their username, password, and email and by choosing a role: *Farmer* or *Veterinarian*. When a field is left blank, the app notifies the user to fill in the blank. It also validates that the username is not already taken. When the registration process is done, users can log out or make use of the system.
Users are required to submit their user name and password for the login process which is authenticated by the app. Incorrect information triggers an alert. Users who opt to stay logged in will not have to enter their credentials again in the next sessions.

3.7.2 Dashboard

After users log in, they are taken to the dashboard where they have a full view of the health of their livestock. The dashboard consists of:

- **Header:** Shows the profile image and account options. Users can edit their personal details like name and email, though the role field cannot be changed.
- **Veterinarian Verification:** Veterinarians can upload their certification documents for admin verification.
- **Animal Health Overview:** A pie chart shows the percentage of healthy and diseased animals.
- **Disease Frequency:** A bar graph shows how many cases were detected for each disease.
- **Weather Information:** Displays the current weather details such as temperature, humidity, and wind speed, as environmental conditions can affect livestock health.
- **Calendar:** A built in calendar shows dates and helps users track detections and related activities.

3.7.3 Image Capture and Diagnosis

In this section you can select images of livestock from your gallery or take the camera to snap new ones. If the users decide to cancel the upload, they go back to the previous page. When an image is chosen, the model runs on the image and predicts if the animal is healthy or not.The prediction result is displayed clearly on the screen, indicating the identified disease or a healthy status along with the model's confidence level. Users are then provided with actionable recommendations and the option to save the diagnosis report for future reference or veterinary consultation.
If the animal is deemed healthy, an animation will display a "healthy" status. If a disease is detected, the app will redirect the user to a detailed page that includes:

- Disease name
- Confidence score
- Short description
- Recommended medicines
- Preventive measures

3.7.4 History Management

The app keeps track of all the past disease detections. Each record contains the name of the disease, a related image, the date of detection and a confidence score. Users can see details or delete any record if needed. This allows to follow the animal's historical health and monitor chronic diseases.

3.7.5 Admin Panel

The admin section allows you to have full control over the system. Admins have the ability to view, edit and delete user accounts and also to validate veterinarians' credentials. They also have the ability to modify disease information, including the description, treatment recommendations, and prevention options.

3.7.6 AI Disease Prediction Module

The app feature is predictor module which uses trained MobileNetV2 model. The model processes the image and predicts the disease with a confidence percentage when an image is submitted. This allows farmers and vets to quickly spot problems and take preventive or remedial action.

3.8 Model Testing and Validation

The model and the app were tested for reliability. The MobileNetV2 model was tested with other models such as EfficientNet and VGG16 to test the precision and the performance. The prediction results were analyzed using a confusion matrix. Further, observations were made on the system's understandability and practicality through user evaluation. The findings showed that the app provides fast and accurate predictions, which help the farmers to identify the diseases at an early stage and take necessary measures.

3.9 Summary

In this chapter, the whole process of Smart Livestock Disease Predction System design is described. Development became agile and iterative within the Agile methodology. The dataset was carefully collected, balanced, and preprocessed to ensure accurate disease classification. MobileNetV2 was chosen because of its lightweight and efficient performance on mobile devices for real time image based predictions.
The system was developed as a React Native application for mobile with a Node JS backend and MongoDB database. Among other things, user authentication, a dashboard, disease diagnosis, history tracking and administrative privileges were added which made

the app more usable and friendly to users. Testing confirmed the accuracy and reliability of the system, validating its usability and efficiency in real life for detection and control of livestock diseases.

Apart from these capabilities, the system allows farmers and veterinarians to communicate easily. It allows for informed, rapid decisions by informing users about the health of the livestock in real time, with updates and alerts. The use of machine learning algorithms also improves the accuracy of disease predictions and allows for continuous learning based on new data inputs. This flexibility will allow the system to stay current and effective in addressing future livestock health issues, and ultimately result in better animal welfare and food production.

Moving forward, the system is poised to serve as a foundational platform for broader agricultural innovation. Future development may integrate features such as geospatial disease mapping, integration with local veterinary networks for telemedicine, and compatibility with IoT sensors for comprehensive herd health monitoring. By continuing to prioritize accessibility and user-centric design, the project embodies a sustainable, scalable solution that empowers rural communities. This chapter thus concludes by affirming that the Smart Livestock Disease Prediction System not only meets its immediate technical and practical objectives but also sets a transformative precedent for the application of AI in global agricultural resilience and development.

Chapter 4

Requirement Specifications

4.1 Overview

This chapter describes fully the requirements of Smart Livestock Disease Prediction System. It defines the software, hardware, and other requirements, including functional and non functional user requirements, along with a detailed use case description. The requirements are structured to provide an unambiguous development guidance for the system, verifying that they are consistent with the stakeholder expectations and are technically feasible.

4.2 Software and Hardware Requirements

4.2.1 Software Requirements

The system is implemented with React Native for the mobile front end and Node.js / Express.js for the backend. MongoDB is used for the database and Python along with Tensor Flow and Keras is used to train the MobileNetV2 model for predicting the disease. Mainly Visual Studio Code and Jupyter Notebook are used in this process. Furthermore, libraries like Recharts are used within the mobile application for graphing and visualization of data.

4.2.2 Hardware Requirements

Using a smartphone with at least 4GB RAM and a working camera is recommended for best experience of the mobile app. A computer with a GPU is required to train and test the model to perform deep learning computations. Despite that a stable internet connection is recommended for uploading images and receiving predictions, some minimal functionalities are available offline. The server backend requires a system with sufficient processing power and memory to handle multiple concurrent prediction requests.

4.3 Requirements Specifications

The Smart Livestock Disease Prediction System is a realistic digital instrument that allows farmers to identify animal disease early and rapidly. It has to be simple enough for people with basic smartphone skills to use it properly, yet have advanced functionalities for vets to share their expertise. Among the system specifications are the following: the system should achieve early detection of disease by analysing images, provide a comprehensible result together with treatment recommendations, and monitor the health of the animal simply over a period of time. In rural areas, the system should be able to run with minimum internet connection so that the farmers can obtain bacteriological information of their animals at any time and any place.

4.3.1 Functional Requirements

The functional requirements specify what the system should do and its necessary features to meet user needs. These are grouped by different user types and by system functionality, covering all the key functions.

- **FR-01: User Registration and Login**
  - **Input**: For registration: username, email, password, role (Farmer/Veterinarian). For login: username/email and password
  - **Process**: Registration: Check all fields, validate email, check unique username, encrypt password, save to database. Login: Verify credentials, create session
  - **Output**: Registration: Account created or error message. Login: Access to dashboard or login error
  - **Description**: Allows users to create accounts and securely access the system. Vets upload certificates during registration
- **FR-02: Profile Management and User Logout**
  - **Input**: Profile update: name, email, profile picture. Logout: user action or timeout
  - **Process**: Profile: Validate new data, update database. Logout: End session, clear data
  - **Output**: Profile: Updated information shown. Logout: Returned to login screen
  - **Description**: Users edit personal information and securely end sessions to prevent unauthorized access
- **FR-03: Capture and Image Upload**
  - **Input**: Camera access or gallery selection, livestock image

  - **Process**: Open camera or gallery, capture/select image, validate size/format, show preview
  - **Output**: Image preview with options to diagnose, retake, or cancel
  - **Description**: Users take photos of livestock or upload images from gallery for disease analysis

- **FR-04: AI-Powered Disease Diagnosis and Result Display**
  - **Input**: Livestock image for analysis
  - **Process**: Upload image, process through MobileNetV2 model, analyze for disease
  - **Output**: Disease name with confidence score, description, medicines, prevention methods
  - **Description**: AI analyzes images to detect diseases and provides detailed treatment information

- **FR-05: Prediction History Management and Analytics**
  - **Input**: User request to view history, record selection
  - **Process**: Retrieve past predictions, organize by date, display with details
  - **Output**: List of past diagnoses with date, disease name, confidence, image thumbnail
  - **Description**: Users review past predictions, view details, and delete records if needed

- **FR-06: Dashboard Visualization**
  - **Input**: User login or dashboard refresh
  - **Process**: Calculate health statistics, create charts, fetch weather data
  - **Output**: Pie charts, bar graphs, weather info, calendar view
  - **Description**: Provides overview of livestock health with visual charts and weather information

- **FR-07: Veterinarian Verification**
  - **Input**: Certificate upload by veterinarian
  - **Process**: Accept certificate file, store securely, flag for admin review
  - **Output**: Verification status (Pending/Approved/Rejected) with re-upload option
  - **Description**: Vets upload certificates for admin verification to access editing features

- **FR-08: Admin Requirements**
  - **Input**: Admin actions for management
  - **Process**: View all users, edit information, verify vets, manage disease database
  - **Output**: System changes confirmed, updated lists, status changes
  - **Description**: Admins manage entire system including users, verifications, and disease info
- **FR-09: Realtime Notification and Alerts**
  - **Input**: System events like predictions, verification updates
  - **Process**: Detect important events, generate notifications, send to users
  - **Output**: Push notifications or in-app alerts for important updates
  - **Description**: Keeps users informed about important system events and updates
- **FR-10: Error Handling and Data Validation**
  - **Input**: All user inputs throughout application
  - **Process**: Check data formats, validate inputs, catch system errors
  - **Output**: Clear error messages for users, system logs for developers
  - **Description**: Ensures data quality and provides helpful error messages

4.3.2 Non Functional Requirements

The non functional requirements are the performance requirements that the system must satisfy, and they concern with how well the system performs rather than what services it provides. These are the requirements which ensures that the system is quick, secure, easy to use and scalable in a sustainable manner to efficiently serve the needs of the farmers and the veterinarians.

- **NFR-01: Performance Requirements**
  The responsiveness of the system shall be considered when designing. Disease identification reports shall be obtainable 2-5 sec after a picture is taken. User dashboard should be quick to load after login and all functions of the application should be smooth without freezing or lagging too much. The time for uploading and processing photos should be as short as possible even with a normal web connection in a rural area, and all UI elements must also be quick when responding to user input. The system must maintain this performance standard while supporting a growing user base and increasing volumes of image data without significant degradation.

- **NFR-02: Scalability Requirements**
  The system design shall support multi access, performance will not degrade under high number of concurrent users. The system should be able to handle peak load of several farmers using the application at the same time and have the possibility to grow for more users. The backend database should be capable of efficiently handling a growing number of animal photos and associated records, while the server infrastructure should scale seamlessly to accommodate increasing load.

- **NFR-03: Security Requirements**
  The system shall include strong security mechanisms to protect sensitive information. Passwords: Passwords for all users must be stored using strong industry standard encryption techniques. Sessions and Credentials Users' login sessions and credentials must be protected, and you must protect all data transmitted. Photos of animals, farm records, and documentation for vet cert must stay private and confidential. Sensitive information will be tightly controlled so that only those who have been authorized will be able to see or manipulate the information to which they have been granted access.

- **NFR-04: Usability Requirements**
  The UI will be simple and intuitive and targeted towards farmers. It should be simple enough for people with minimal education to use and not too technical for them to understand. All buttons, menus and icons are well defined, consistent in location and intuitive. The instructions and system messages should be written in plain and unambiguous language. The flow and navigation within the different functionalities like photo uploading, history monitoring, veterinarian chat, etc. should be natural, intuitive and not to require too much training to learn.

- **NFR-05: Maintainability Requirements**
  The system is to be constructed with the view of simplifying maintenance and future upgrades. The code shall be structured, modular and well documented to allow for efficient debugging and rapid response to problems. The AI model's disease database needs to be extendible which would allow for the addition of new diseases or updating symptoms by the admins/developers with minimal downtime. The administrative back end should provide simple, intuitive interfaces for editing disease data, managing users, and configuring system settings.

- **NFR-06: Accessibility Requirements**
  The app should be available to users in a range of rural settings. It should function in an offline or online mode with synchronized when connected mode, and minimize the data lost in unstable connection, minimize the network traffic. The interface must be readable in all lighting conditions, including direct sun light. The application needs

to work with the standard mobile phone type and IOS/Andriod operating systems which are accessible for the target group. It must be designed for a heterogeneous user group with different educational levels, so that has everyone can use at least the basic functions of the application.

4.4 Use Case Analysis

4.4.1 Complete System Use Case

The Smart Livestock Disease Prediction System serves three primary users, each of which is differentices with unique permissions and features. Farmers can sign up, upload pictures of their livestock to check for disease, view results with treatment suggestions, and monitor health history on dashboards. Veterinarians are empowered to perform further actions like uploading certificates to be verified and granted access to disease information management features. Administration: An administrator has the authority over the entire system, adds and deletes users, vets the credentials of a veterinarian, and updates the disease knowledge base. The entire system design combines mobile device features with cloud based AI computation and database management as shown in Figure 4.1.

The system's multi-tiered access structure ensures data integrity and operational security by strictly segregating functionalities according to user roles. This role-based design prevents unauthorized data manipulation while enabling efficient collaboration between farmers seeking diagnoses and veterinarians providing expert oversight. The architecture facilitates a seamless flow from data capture on mobile devices to cloud-based processing and back, creating a closed-loop ecosystem for livestock health management.

Furthermore, this user-centric design prioritizes accessibility and practicality for its target rural audience. The farmer's interface emphasizes simplicity and clarity, with large buttons and step-by-step guidance for image capture. In contrast, the veterinarian and admin portals offer more detailed data views and management tools, reflecting their advisory and supervisory responsibilities. By tailoring the experience to each user type, the system maximizes utility without overwhelming any single user group, thereby encouraging adoption and sustained use across the entire agricultural support network.

This hierarchical model also establishes a clear chain of responsibility and trust within the platform. Farmers can confidently rely on the system's AI-driven predictions, knowing they are supported by a verified network of veterinary professionals. Veterinarians, in turn, are equipped with a tool that extends their reach and impact, allowing them to manage cases and share knowledge digitally. Ultimately, the administrator acts as the system steward, ensuring its accuracy, security, and relevance by maintaining an up-to-date medical database and user registry.

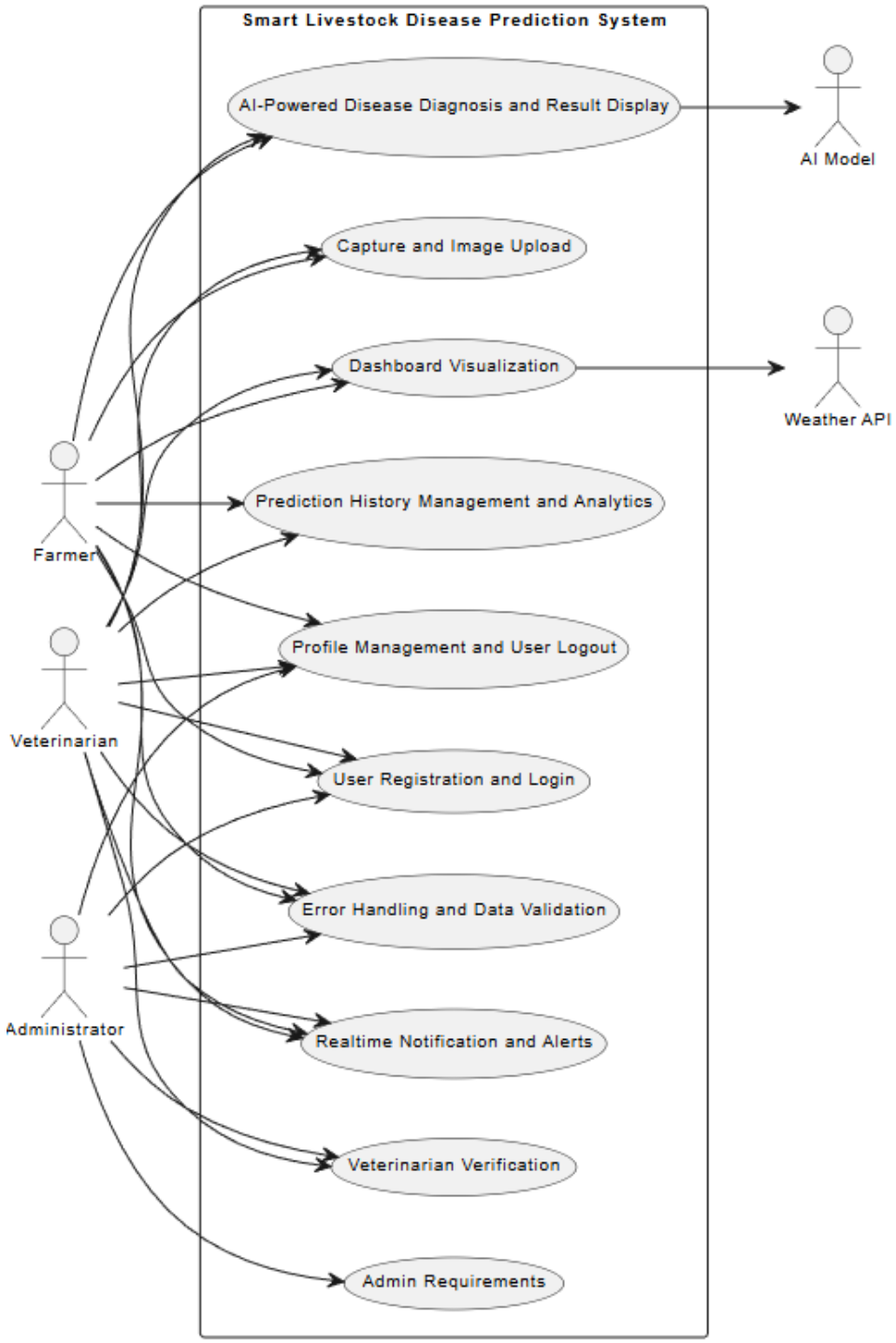


Figure 4.1: Complete System Use Case Diagram of Smart Livestock Disease Prediction System

System workflow begins at user registration and authentication and ends with the main service of disease detection using image processing. The results are saved for historical tracking and visualized by analytic dashboards. Third party integrations include

weather data APIs, and device hardware for camera functionality etc. Each of these use cases correspond to a particular user goal within this eco-system, from simple account management to complex disease information curation.

4.4.2 Use Case Modeling

The Use Case Modeling section provides detailed Use Case illustrations of the system among different types of users. Each use case corresponds to a particular task or goal the user can perform in the system. The following top ten use cases cover all aspects, from simple account creation to complex disease knowledge management. For all use cases, a narrative and a UML use case diagram are provided.

The Use Cases are structured to illustrate the full system workflow, from management of the user account to the core disease detection functions and finally to system administration and support. This addresses all users' requirements and gives at the same time a solid basis for system implementation and testing. They help in visualizing the relations between actors and use cases, the specification tables identify exact details of every interaction scenario. Fig. 4.1 displays the full use case diagram, and the separate use cases are showed in a logical order. This helps the reader to understand the balance between the big picture of the system and the detailed constructs of the interactions for each user. The use cases are traced to individual functional requirements, ensuring traceability as well as coverage of the entire system.

4.4.2.1 UC01-User Registration and Login

This case covers the system to begin with access. Farmers and vets are able to register for new accounts with minimal information (username, email, password, and role). A veterinarian must also upload professional certificates. Use your credentials to sign in if you are an existing user. The system checks all the inputs, Make sure there are no duplicate usernames, encrypt the password and create secure sessions. As a System Root Use Case this use case see Figure 4.2 functions as a critical security gatekeeper for all other use cases. The two main actors Farmer and Veterinarian appear on the diagram performing the daily activities exposed by the system, which are broken down in the core included use cases of *Register New Account* and *Login to System*. This graphical notation identifies the user and enforces the role based security which is needed to identify the features for every user, which is described in detail step by step in Table 4.1. For example, the selected role on register side immediately dictates on whether to show the certificate upload page, and the level of access for the user on such features as disease information management. Also, the login system authenticates a user and initiates persistent sessions which track the user state in different areas of the application to provide continuity of the user experience as the user explores the functionalities of the livestock disease prediction system.

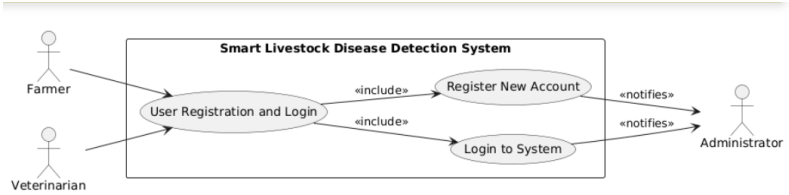


Figure 4.2: User Registration and Login Use Case Diagram

Table 4.1: User Registration and Login Use Case Specification

Use Case ID	UC01
Use Case Name	User Registration and Login
Primary Actor	Farmer, Veterinarian
Secondary Actor	System
Preconditions	User has app installed; Internet connection available
Postconditions	Account created; User logged in to dashboard
Trigger	User selects "Sign Up" or enters login credentials
Main Flow	**Registration:** 1. User selects "Sign Up" 2. System displays registration form 3. User enters username, email, password, role 4. If role=Veterinarian, upload certificate 5. User submits form 6. System validates inputs 7. System creates account 8. Success message displayed **Login:** 9. User enters username/email and password 10. System verifies credentials 11. System creates session 12. User redirected to dashboard
Alternative Flows	**A1: Duplicate username** 7a. Username already exists 7b. Display: "Username already taken" 7c. Return to step 3
Exception Flows	**E1: Network error** 7a/11a. Connection lost 7b/11b. Display: "Network error. Please try again" 7c/11c. Return to appropriate step

4.4.2.2 UC02-Profile Management and User Logout

This use of the service allows users to update their personal details and log out of their sessions with confidence. Users can customize their name, email address, and profile image to keep their account information up to date. When you access the profile options, you

will find a simple and convenient interface that allows you to quickly make any changes you want. After making the modifications, users are required to confirm their updates, upon which the system verifies the new data for its validity and security compliance. The logout option is intended to provide a more secure session by allowing a manual logout after a time span. This protects the user account from being accessed by someone else, for example if the user forgets to logout after using a shared computer. When you securely leave your account, you will also be shown a notification confirming that you have successfully ended your session, so you can be sure that your account is safe. Overall, this use case allows users to not only update their personal details, but also highlights security in account management like nowhere else.

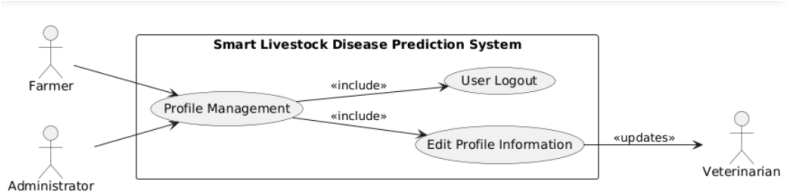


Figure 4.3: Profile Management and User Logout Use Case Diagram

The modular nature of this function is visually illustrated in Figure 4.3. The figure also demonstrates that Farmers and Veterinarians are Actors in the *Profile Management and User Logout* Use Case, which uses two core included sub use cases *Edit Profile Information* and *Logout from System*. This separation makes it clear that these are different, but equally important tasks under like the one user target. The details of these processes, are specified in Table 4.2 describes important system activities related to data validation and session management. The alternative flow for an automatic session timeout introduces a critical layer of passive protection that guards users' accounts against being left open to unauthorized access during idle periods. This safeguard is particularly relevant for a mobile application likely to be used on shared or personal devices in unpredictable rural environments. The timeout mechanism, configurable by role, automatically terminates sessions after a period of inactivity, requiring re-authentication to resume. This not only protects sensitive livestock health data and farmer information but also aligns with security best practices for applications handling personal and agricultural data.

Table 4.2: Profile Management and User Logout Use Case Specification

Use Case ID	UC02
Use Case Name	Profile Management and User Logout
Primary Actor	Farmer, Veterinarian
Secondary Actor	System
Preconditions	User is logged in with valid session
Postconditions	Profile updates saved to database; Session terminated on logout
Trigger	User selects "Edit Profile" or "Logout" option
Main Flow	**Profile Management:** 1. User navigates to profile section 2. System displays current profile information 3. User selects "Edit" 4. System displays editable form with current values 5. User makes changes 6. User saves changes 7. System validates inputs 8. System updates database 9. Success message displayed **Logout:** 10. User selects logout option 11. System clears session token 12. System clears local cache 13. User redirected to login screen
Alternative Flows	**A1: Automatic timeout** 10a. 30 minutes of inactivity detected 10b. System displays timeout warning 10c. If no response, automatic logout with message
Exception Flows	**E1: Email already in use** 7a. Changed email already registered to another user 7b. Display error: "Email already registered to another account" 7c. Return to step 5

4.4.2.3 UC03-Capture and Image Upload

Acquisition of visual information for livestock disease is the aim of this use case. As shown in Figure 4.4, the Farmers and Veterinarians are able to access the service by using two main approaches: to take a new photo with device camera or to upload a photo from the gallery. This action is started at the main screen of the application, and is the basic prior step for diagnosis flow. Once selected, the system switches to the corresponding hardware or software interface, prompts the user to take a clear shot of the animal, confirms with a preview window. The user can then diagnose, retake or cancel. One important feature displayed in this diagram is to validate the quality of images, such as the resolution, lighting, focusing, before the image is sent to AI analysis. This verification is essential to ensure accuracy in later disease predictions.

The layout of this use case (show in Figure 4.4) is directly influenced by the device on which it runs the end user's operational context. It processes rural standard situations like occasional access to network, by means of offline image capture and very lucid feedback. When transposing the physical observation into the digital world, this use case effectively ensures the validated input that is the prerequisite for the AI diagnostic engine of the next stage in the system workflow.

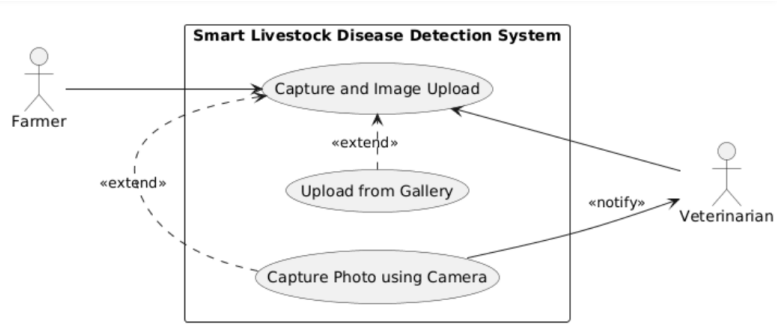


Figure 4.4: Capture and Image Upload Use Case Diagram

Figure 4.4 is a use case diagram that summarizes the most important relationships discussed in this section. The main use case *Capture and Image Upload* is depicted being triggered by both the Farmer and Veterinarian actors. The diagram also makes use of «`extend`» relationship for the two optional and substitute ways for providing input, *Capture Photo using Camera* and *Upload from Gallery*. These extensions cover the major alternative flows (A1, A2) which are the following specification table. The visual modeling language complements the textual specification, by immediately providing a high level view of all the actors that represent the image input use case and the modular system structure for processing image input. This diagram effectively illustrates the system's flexibility in accommodating different user preferences and technical scenarios, such as capturing a fresh image in the field or selecting a previously stored photo from the device's library. The clear separation of the core upload function from its extension points highlights the system's modular design, where the main processing logic remains consistent regardless of the image source. This visual representation ensures stakeholders can quickly grasp the complete user journey for this fundamental feature, from initiation by either user role through to the alternative methods of image acquisition.

Table 4.3: Capture and Image Upload Use Case Specification

Use Case ID	UC03
Use Case Name	Capture and Image Upload
Primary Actor	Farmer, Veterinarian
Secondary Actor	System, Camera Hardware
Preconditions	User is logged in; Camera/gallery permissions granted
Postconditions	Image captured/selected and ready for diagnosis
Trigger	User selects "Capture Image" or "Upload Image"
Main Flow	**Camera Capture:** 1. User selects "Take Photo" 2. System opens camera interface 3. User captures livestock image 4. System shows image preview 5. User confirms image **Gallery Upload:** 6. User selects "Upload from Gallery" 7. System opens gallery 8. User selects livestock image 9. System shows image preview 10. User confirms image
Alternative Flows	**A1: Retake photo** 4a/9a. User selects "Retake" 4b/9b. Return to camera interface **A2: Cancel upload** 4a/9a. User selects "Cancel" 4b/9b. Return to image capture menu
Exception Flows	**E1: Permission denied** 2a/7a. Camera/gallery access denied 2b/7b. Display: "Please enable camera/gallery permissions" 2c/7c. Redirect to settings

4.4.2.4 UC04-AI Powered Disease Diagnosis and Result Display

This is the main scenario where AI scans the pictures of livestock to find diseases. The MobileNetV2 model classifies the image, potential diseases, and the confidence scores. The output was the name of the disease, a description, treatment suggestions and preventive actions.

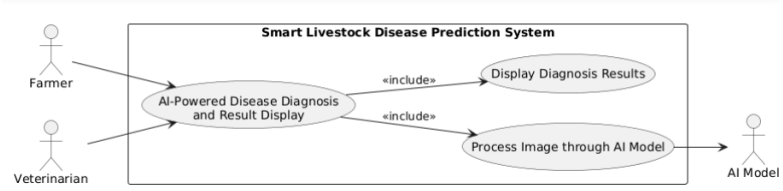


Figure 4.5: AI-Powered Disease Diagnosis and Result Display Use Case Diagram

Table 4.4: AI Powered Disease Diagnosis and Result Display Use Case Specification

Use Case ID	UC04
Use Case Name	AI-Powered Disease Diagnosis and Result Display
Primary Actor	Farmer, Veterinarian
Secondary Actor	AI Model, System
Preconditions	Image captured/uploaded; User selects "Diagnose"
Postconditions	Disease prediction results displayed; Record saved to history
Trigger	User selects "Diagnose" button
Main Flow	1. User selects "Diagnose" on image preview 2. System uploads image to server 3. AI model processes image 4. Model returns prediction with confidence score 5. System retrieves disease information from database 6. System displays results: - Disease name - Confidence percentage - Disease description - Primary medicines - Preventive measures 7. System saves prediction to history database
Alternative Flows	**A1: Healthy animal** 4a. Model returns "Healthy" 4b. System shows: "Animal appears healthy" with happy animation
Exception Flows	**E1: Low confidence** 4a. Confidence score < 70% 4b. System suggests: "Please upload clearer image" **E2: Model error** 3a. AI model fails 3b. Display: "System error. Please try again"

4.4.2.5 UC05-Prediction History Management and Analytics

This is a user story for the Prediction History purpose. Users can browse their predicted history by chronological order, look at details of each prediction, search with a keyword, and delete entries. It serves as a health log for tracking livestock wellness. Each historical entry is displayed with essential information including a thumbnail of the captured image, the disease name, the date of detection, and the confidence score generated by the AI prediction. Users can tap on any record to view full details, including an original image, a detailed description of the disease, offered treatments and preventative measures at the moment of diagnosis.

The History also improves management of the data itself with some filtering tools, letting you filter the records by date range, a specific disease or confidence level. This allow the farmers to track the trend of disease incidents and treatment efficacy. For privacy and

data handling users can selectively remove individual records or erase their entire history. All historical data is synced to a secure cloud database to enable access via multiple devices and ensure data integrity. With this all encompassing monitoring system, users are able to control its livestock's health with informed decisions derived from historical trends.

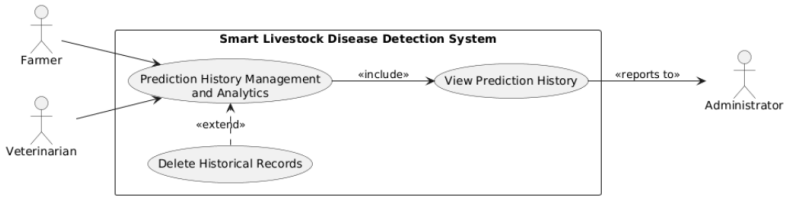


Figure 4.6: Prediction History Management and Analytics Use Case Diagram

The organisation of this functionality is captured visually in Figure 4.6. The diagram also indicates that both Farmers and Veterinarians as primary actors can trigger the main use case, *Prediction History Management and Analytics*. Inside this boundary, two main use cases are included: *View Prediction History* which is a mandatory use case and an included function, and *Delete Historical Records* which is an optional use case and extending function. This graphically represents that to access history is always a part of the use case whether or not the user decides to delete, with that being an alternate activity of the use case. The figure can be seen as a generalization for the remaining sub flows. It provides an exception flow for a mid flow and demonstrates the optional flow two End Users).The diagram aligns well with the following services as they are defined and described in the service table with regard to managing historical health data.

In addition, the diagram depicts the internal data dependency among the use cases. With its primary function, *View Prediction History* to be successful, it needs prior diagnostic records produced by the system's AI engine(UC04). The history tab basically acts as a collection point for and provides the interface to use the outputs generated from past runs of the disease detection process. This linkage highlights how the system is interconnected with each use case depending on the outputs from the previous to deliver a seamless workflow from image capture to diagnosis and finally to long term health tracking and analysis. This graphically represents that to access history is always a part of the use case whether or not the user decides to delete, with that being an alternate activity of the use case. The figure can be seen as a generalization for the remaining sub flows. This linkage highlights how the system is interconnected with each use case depending on the outputs from the previous to deliver a seamless workflow from image capture to diagnosis and finally to long term health tracking and analysis.

Table 4.5: Prediction History Management and Analytics Use Case Specification

Use Case ID	UC05
Use Case Name	Prediction History Management and Analytics
Primary Actor	Farmer, Veterinarian
Secondary Actor	System, Database
Preconditions	User is logged in; Has previous predictions
Postconditions	History viewed; Records possibly deleted
Trigger	User selects "History" from navigation
Main Flow	1. User navigates to History section 2. System retrieves user's prediction records 3. System displays chronological list with: - Image thumbnail - Date detected - Disease name - Confidence score 4. User scrolls through history 5. User selects record for details 6. System shows complete prediction information
Alternative Flows	**A1: Delete record** 6a. User selects "Delete" 6b. System confirms deletion 6c. User confirms 6d. System removes record from database 6e. List updated
Exception Flows	**E1: No history** 2a. User has no previous predictions 2b. Display: "No history yet. Capture your first image!" **E2: Deletion error** 6da. Database deletion fails 6db. Display: "Unable to delete. Please try again"

4.4.2.6 UC06-Dashboard Visualization

This use case provides the user with a single dashboard showing summary information about the health of livestock through visual charts, disease occurrence analysis, weather data, and a calendar interface. The dashboard is refreshed automatically with the most recent data. The dashboard is refreshed automatically with the most recent data. Farmers can quickly assess the overall health status of their herd via a color-coded pie chart, while trend lines and bar graphs help identify seasonal disease patterns. Integrated weather forecasts and a management calendar enable proactive planning for vaccinations and treatments. This consolidated view empowers users to make data-driven decisions efficiently, transforming raw health metrics into actionable insights for improved livestock management.

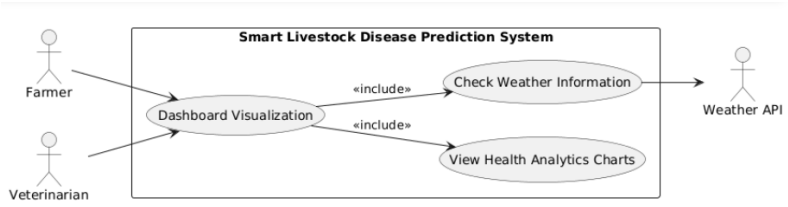


Figure 4.7: Dashboard Visualization Use Case Diagram

Table 4.6: Dashboard Visualization Use Case Specification

Use Case ID	UC06
Use Case Name	Dashboard Visualization
Primary Actor	Farmer, Veterinarian
Secondary Actor	Weather API, System
Preconditions	User is logged in
Postconditions	Dashboard displayed with current analytics
Trigger	User logs in or refreshes dashboard
Main Flow	1. User logs into system 2. System loads dashboard automatically 3. System retrieves user's prediction history 4. System generates: - Pie chart: Healthy vs Diseased animals - Bar graph: Disease frequency 5. System fetches current weather data 6. System displays: - Health overview charts - Current weather (temp, humidity, wind) - Calendar with disease dates highlighted 7. User interacts with dashboard elements
Alternative Flows	**A1: Manual location** 5a. Auto-location fails 5b. User manually selects city 5c. Fetch weather for selected location
Exception Flows	**E1: Weather API error** 5a. Weather service unavailable 5b. Display: "Weather data temporarily unavailable" 5c. Continue with other dashboard elements

4.4.2.7 UC07-Veterinarian Verification

This use case manages the professional identity verification for Veterinarians. The vets send in their certificates and the admins then verify and approve. The verified status is used to enable access to special features, such as the ability to edit disease information.

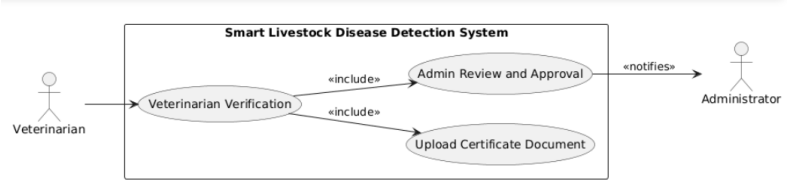


Figure 4.8: Veterinarian Verification Use Case Diagram

Table 4.7: Veterinarian Verification Use Case Specification

Use Case ID	UC07
Use Case Name	Veterinarian Verification
Primary Actor	Veterinarian, Administrator
Secondary Actor	System
Preconditions	Veterinarian has registered account
Postconditions	Verification status updated; Notifications sent
Trigger	Veterinarian uploads certificate or admin reviews requests
Main Flow	**Veterinarian side:** 1. During registration/profile edit, vet uploads certificate 2. System validates file format and size 3. System stores certificate 4. System sets verification status to "Pending" **Admin side:** 5. Admin navigates to verification dashboard 6. System displays pending requests 7. Admin reviews certificate 8. Admin approves or rejects 9. System updates vet's verification status 10. System notifies veterinarian of decision
Alternative Flows	**A1: Re-upload after rejection** 10a. Veterinarian receives rejection with reason 10b. Veterinarian uploads corrected document 10c. Return to step 2
Exception Flows	**E1: Invalid file format** 2a. File not PDF, JPG, or PNG 2b. Display: "Please upload valid certificate format" 2c. Return to step 1

4.4.2.8 UC08-Admin Requirements

This Use Case provides the Administrator with a high level control over the system through a single interface to have an overall view of the platform, depicted by the activity diagram of Figure 4.9 and the description in Table 4.8. The type of operations required are, user account management to create, modify and delete user accounts while the required access control must be achieved. The verification of the veterinarian credentials by means of

professional certificates is also exclusively in the hands of the Administrator, and it is a key point to prevent the platform pieces from falling apart. In addition, this responsibility includes the selection and registration of diseases in the base of knowledge of diseases, so that all diagnostic information related to diseases diagnostic remains up to date, accurate, and evidence based, which are the essentials for the AI core diagnostic function. The management module is like the brain nerve center of the platform, managing all the high level Control functions. It contains essential interfaces for the purpose of system health monitoring, user permission administration and data quality management for the application. Centralized Control is a management option where all decisions are made at the top level within a platform. Role based authentication is used to tightly control access to these functionalities, and the Administrator role is a very special one. This security policy enforces strict separation of tasks between end users (Farmers, Veterinarians) and system administration services. It has been designed to comply with the needs of administrators with dashboards, bulk changes and full audit logs.

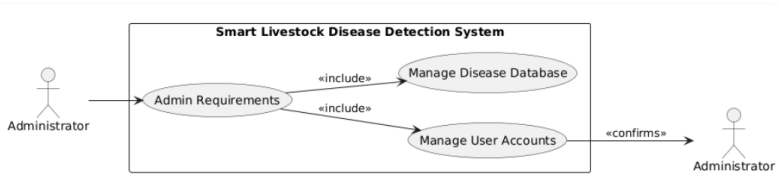


Figure 4.9: Admin Requirements Use Case Diagram

The visual representation in Figure 4.9 emphasizes the essential separation of powers by showing the Administrator as the unique agent for this privilege, which is further divided into managing user accounts and managing the disease database. The boundaries of administrative power are clearly defined in the diagram, along with the high level duties nested within these are that of the system architecture. The detailed workflow specification of Table 4.8 implements the administrative constraints as a series based workflow from authentication to successful completion of a task. It breaks down the duties in Figure 4.9 at a high level into tangible, actionable items that are both safe and efficient to execute. The addition of bulk operations (Alternative Flow A1) caters for practical scale of user administration issues, so that an administrator can deal with an expanding user population. At the same time, the exception processing for disallowed operations (Exception Flow E1) also realizes the security principle of least privilege by introducing validation points even for the Administrator to mitigate possible privilege abuse or mistakes. This carefully crafted process allows for central oversight that goes far beyond simple end user control. It provides proactive system monitoring with built in dashboards, full security reporting on activity logs, and platform enhancement via analytics. With well defined processes for administrative intervention, the system retains its operational integrity while enabling

flexible improvements. This layer of administration is key to ensuring the platform can remain reliable as it scales, keeping the core disease detection functionality supported by strong back end management options.

Table 4.8: Admin Requirements Use Case Specification

Use Case ID	UC08
Use Case Name	Admin Requirements
Primary Actor	Administrator
Secondary Actor	System, Database
Preconditions	Admin is logged in with admin privileges
Postconditions	System changes applied as per admin actions
Trigger	Admin logs into admin panel
Main Flow	1. Admin logs into admin interface 2. System displays admin dashboard 3. Admin selects management option: - Users - Veterinarians - Diseases 4. System displays relevant management interface 5. Admin performs actions: - View all users - Edit user information - Delete users - Verify veterinarians - Update disease information 6. System executes requested operations 7. System confirms successful completion
Alternative Flows	**A1: Bulk operations** 5a. Admin selects multiple users 5b. Admin applies bulk action (delete, change status) 5c. System processes all selected items
Exception Flows	**E1: Unauthorized action** 6a. Admin tries action beyond permissions 6b. Display: "Action not authorized" 6c. Return to step 4

4.4.2.9 UC09-Realtime Notification and Alerts

The following use case describes all the alerts and notifications services of the system and how they can be used to communicate with users as shown in Figure 4.10 and Table 4.9. Users are notified in a timely manner about important events, including finished predictions for diseases, changes in the status of veterinarian verification, and system announcements, by means of push notifications and notifications within the application.

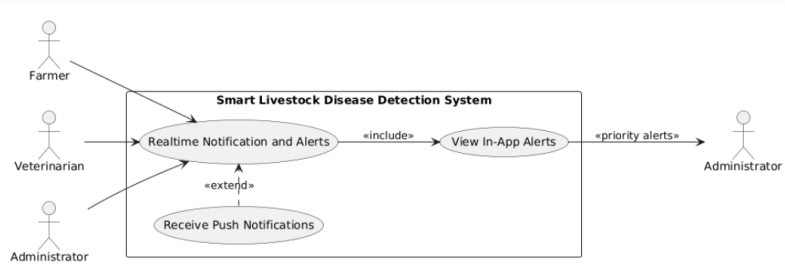


Figure 4.10: Realtime Notification and Alerts Use Case Diagram

Table 4.9: Realtime Notification and Alerts Use Case Specification

Use Case ID	UC09
Use Case Name	Realtime Notification and Alerts
Primary Actor	Farmer, Veterinarian, Administrator
Secondary Actor	System
Preconditions	User has app with notification permissions
Postconditions	Notifications sent and received
Trigger	System event occurs requiring notification
Main Flow	1. System event occurs: - Disease prediction complete - Veterinarian verification update - System announcement 2. System checks which users should receive notification 3. System generates appropriate notification message 4. System sends push notification to user's device 5. User receives notification 6. User views notification details in app
Alternative Flows	**A1: In app alerts only** 4a. User has push notifications disabled 4b. System shows alert only within app 4c. User sees alert next time app is opened
Exception Flows	**E1: Notification service failure** 4a. Notification service unavailable 4b. System retries after delay 4c. Log error for debugging

4.4.2.10 UC10-Error Handling and Data Validation

The whole error handling and data validation of the system is managed by this use case, and it is modeled and tabulated as in Figure 4.11 and Table 4.10, respectively to guarantee that the system working well and provides friendly user interfaces. It makes sure inputs from a user are validated in real time, elegantly handles system errors, and inform users with friendly and helpful message instead of showing raw or encoded technical errors.

This feature is always running in the background, and capturing errors on many levels, from single fields on a form, to deep system processes, to keep your application running and leading your users to successful, completed tasks without confusion or frustration. It makes sure inputs from a user are validated in real time, elegantly handles system errors, and inform users with friendly and helpful message instead of showing raw or encoded technical errors. It makes sure inputs from a user are validated in real time, elegantly handles system errors, and inform users with friendly and helpful message instead of showing raw or encoded technical errors.

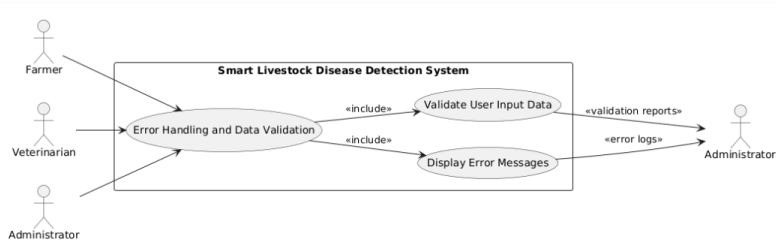


Figure 4.11: Error Handling and Data Validation Use Case Diagram

Figure 4.11 is a use case representation of this concern, indicating that all system actor constructs, Farmer, Veterinarian, and Administrator; have an execution interaction with this core feature. The use case breaks down the main use into two sub processes, which are two core process included in it: *Check User Input Data* and *Show Error Messages*. This architecture reflects that validation and communicating to the user are real time, mandatory aspects for the majority of user interactions, not something users are informed about after they've completed an activity. The conceptual model captured the necessity of treating errors as a core service: errors being abstracted behind simple reports, having them assume dual roles, and so forth, one long standing, cherished feature of the model that all others are tacitly predicated upon. The detailed description in Table 4.11 gives concrete details to those routines. The main flow describes the two treatments for invalid input and unexpected system exception, showing the system philosophy of bright communication with users ("show specific error message") and friendly attitude toward developers ("log technical details for debugging"). By factoring in real time validation as an alternative step the user benefits from real time feedback. The exceptional path for critical system errors establishes a controlled system failure even in such extreme circumstances, to maintain user trust and provide administrators with critical diagnostic information, which can be invaluable in resolving such errors.

The detailed description in Table 4.11 gives concrete details to those routines. The main flow describes the two treatments for invalid input and unexpected system exception, showing the system philosophy of bright communication with users ("show specific error

message") and friendly attitude toward developers ("log technical details for debugging"). By factoring in real time validation as an alternative step the user benefits from real time feedback. The exceptional path for critical system errors establishes a controlled system failure even in such extreme circumstances, to maintain user trust and provide administrators with critical diagnostic information, which can be invaluable in resolving such errors.

Table 4.10: Error Handling and Data Validation Use Case Specification

Use Case ID	UC10
Use Case Name	Error Handling and Data Validation
Primary Actor	Farmer, Veterinarian, Administrator
Secondary Actor	System
Preconditions	User interacts with system or error occurs
Postconditions	Valid data processed or error message shown
Trigger	User submits data or system error occurs
Main Flow	**Data Validation:** 1. User enters data in form field 2. System validates data format and rules 3. If data is valid, system processes it 4. If data is invalid, system shows specific error message 5. User corrects error and resubmits 6. System processes corrected data **Error Handling:** 7. System error occurs 8. System catches error 9. System shows user friendly error message 10. System logs technical details for debugging
Alternative Flows	**A1: Real time validation** 2a. System validates as user types 2b. Shows validation errors immediately
Exception Flows	**E1: Critical system error** 9a. System cannot recover 9b. Display: "System maintenance. Please try again later" 9c. Log full error details

4.5 Summary

This chapter describes the requirements engineering analysis of the Smart Livestock Disease Prediction System and forms the basis for the design and development in the following chapters. It provides a detailed specification of the software and hardware requirements, as well as functional and non functional requirements, use case models to confirm that the system is feasible, and meets the needs of the stakeholders in technical terms. Functional requirements are divided into ten sections related to user authentication, disease diagnosis powered by AI, and system administration; while non functional requirements describe

essential quality attributes such as performance, security, and usability. Use case modeling with diagrams and tabular specifications that are discussed enables specification of exact details of operation for all users including farmers, veterinarians and administrators. This requirements specification acts as a verifiable criteria against which the system will be validated and also enables the doorway for architectural design through which technical solutions that balance functionality, performance, and implementational issues will be plotted.

Chapter 5

Design

In this chapter the general system design of the Smart Livestock Disease Prediction mobile application is described. It covers the limitations, techniques, system design, and data structures of each module. The major modules of the system are shown from the user perspective (high level design), and from the system perspective (low level design) to illustrate how the entire system components work and interact.

5.1 Design Constraints

In this section, the following key considerations and constraints for the system design are presented. These constraints are to make the solution realistic, user friendly, and applicable to real farm environments. The functional scope of the Smart Livestock Disease Prediction mobile application is limited by certain parameters to maintain dependability and usefulness for the farmers and veterinarians. The system requires a good Internet connection to upload images and to perform the AI based disease diagnosis. It is only for early identification and cannot replace the physical visit to the veterinarian.

5.1.1 Exclusions

This section described the functions of the features that were not provided by the system. The app is not a substitute for a conventional veterinarian and can not be used offline. It only covers a specific range of cattle diseases, so rare or complicated cases may not be detected. The application may provide preventive recommendations and drugs of choice, but it cannot be considered to have full medical/ legal responsibilities, e.g. veterinary licensing or certification.

5.1.2 Assumptions

Feasibility was to be ensured by making some assumptions at the time of design and development of the system. It is assumed farmers will have access to smartphones (with cameras) and internet. Vets will have to verify the AI results and advise on medical care. It is further assumed that users adhere to image capturing instructions correctly in order to obtain valid disease prediction.

5.2 Design Methodology

This subsection describes the system design and development methodology adopted during the design and development of the system with feasibility study and resource availability.

5.2.1 Feasibility Study

The feasibility study investigates if the proposed system is realistic, economical and could be operated in the environment of the system. It checks the technical feasibility by validating that the disease detection is consistent for images captured using different illumination and viewing angles. Economic feasibility includes the cost of training the AI model, and hosting and maintaining it. Operational feasibility assesses the potential for farmers and veterinarians to quickly and easily adopt and use the application.

5.2.1.1 Risks Involved

Like any software development project, there were risks involved with this one that could impact the performance or usability. The most important risks are the technical challenges in image processing, maintaining the accuracy of the AI, protecting the user data, and processing multiple image uploads simultaneously by the app. Another issue is varying levels of digital literacy amongst farmers which may impact uptake of the system.

5.2.1.2 Resource Requirements

This section describes the tools, technologies, and hardware that are utilized in system development. Development may be done only with a smartphone, but you will need a powerful PC to train your AI model, and React Native and Expo Go are just two of the many tools you'll be using to write your mobile app. Node.js with REST APIs is used to create the backend, and the database being used is MongoDB. The AI/machine learning code is in TensorFlow/Keras, using the MobileNetV2 model for disease classification. Support for code development and model evaluation is provided by additional tools such as VS Code, GitHub, and Jupyter Notebook.

5.3 High Level Design

The high level design is the system architecture and the main system modules. It also explains the interactions between different modules to make the system fully functional.

5.3.1 Solution Application Areas

Here we prove how our system serves different users. Farmers add or remove cattle images, get wild disease predictions in real time, and receive advice on prevention. Veterinarians may confirm predictions, add disease information, and maintain data concerning livestock health. The aggregated data may be used by researchers and organizations to analyze disease trends.

5.3.2 System Architecture

The architecture describes how the frontend, backend, and database communicate. The application is based on a client server model. The frontend is a React Native mobile application, and the backend is a Node.js REST API handling user profiles, livestock information, and AI predictions. All user, livestock and prediction data is stored in MongoDB. The AI engine classifies diseases and produces confidence levels for predictions using MobileNetV2.

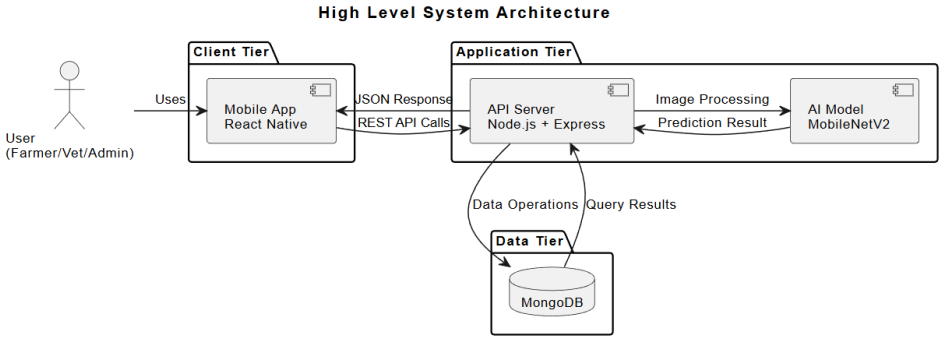


Figure 5.1: High-Level System Architecture Diagram

5.4 Low Level Design

Each component, its architecture and data flow are described in detail, from the high level system architecture down to the component code level, providing the implementation details developers follow when writing the component.

5.4.1 Frontend Component Design

The front end is composed of a series of UI components that are conducive to a fluid user experience. The main screens are the HomeScreen (dashboard displaying animal health summaries combined with recent weather information), UploadImage (to either capture an image or upload one from the device), ResultScreen (to display predictions and recommendations), VetDashboard (allowing veterinarians to confirm diseases), and ProfileScreen (to update user information). React Native components are employed to implement each window and each window has a well defined lifecycle, with each window responsible for its own state including user interactions, as well as querying backend APIs for data. Screen transitions are managed via React Navigation, with a stack navigator managing authentication flows, and a tab navigator managing primary functions including dashboard, image capture, and history. This modular architecture enables maintenance and facilitates the individual testing and enhancement of each user interface component as the system matures.

5.4.2 Backend Modules

The backend implements the core business logic, processing and connection to the database for the frontend. The `UserModule` takes care of user registration, login and profile modification. `LivestockModule` takes care of cattle information and reports. `PredictionModule` performs image analysis and AI inference. `NotificationModule` manages diseases, follow ups and alerts and reminders.

5.4.3 API Endpoints

APIs make the frontend and backend talk to each other. A sample of them include `POST /api/register` to register users, `POST /api/login` for login, `POST /api/livestock/add` to save livestock information, `POST /api/predict` to perform AI based diagnosis, `GET /api/history` to access the history of predictions.

5.5 Database Design

In this section, you will learn about the data organization, storage, and optimization of data processing to facilitate system function with acceptable performance levels under certain security and scalability concerns.

5.5.1 Database Integration

The core data such as user accounts, livestock records, image metadata, notifications are all stored by MongoDB. The users are categorized into farmer, veterinarian and admin,

and each user type is granted different permissions and levels of access. Name, description, medicines, preventions and verification status of each disease record is also listed, representing an extensive knowledge base that can be used to identify and recommend suitable treatment. The schema is modeled with embedded documents and referenced documents to strike a balance between query performance and data normalization, allowing for efficient access to related data and consistency of data when accessed across collections.

5.5.2 Database Optimization

Indexing is used for fast access to the data to make the retrieval faster and partitioning is used for executing tasks on adapting data. Sensitive user information is encrypted and access controlled with role based query restrictions. Querying optimization involves selective field projection in order to minimize the network payload and utilizes aggregation pipelines to perform advanced analytics operations. Routine cleaning of the database, rewriting indexes and archiving data occurs so that the performance is kept up as the number of users and prediction records grow.

5.6 GUI Design

The design and realization of the user interface of the system are described in this section. The overall objective of the design was to have an interface which was visually clean and which could be used by everyone with minimal training, i.e., farmers in rural areas. Intuitive navigation was strongly emphasized as the best way to achieve ease of use, so well known images were used, and a high contrast color pattern and cognitive based information hierarchy were established to decrease the cognitive load of users. The design was influenced by usability heuristics to guarantee (among other things) that crucial operations such as taking an image and viewing the diagnosis results can be done straight away, leading to a device that of the ordinary user as well as for its day to day dependability.

5.6.1 Frontend Development

The app has a mobile first design to provide easy navigation. It has responsive layouts, simplistic icons and an easy to use dashboard which makes farmers and vets apply for services like uploading pics, checking results, requesting profiles etc. Large buttons, high contrast text, and minimal on screen information support its interface for use by potential users in rural areas who may be unfamiliar with smartphones making the disease prediction tool as powerful as it is accessible. Large buttons, high contrast text, and minimal on screen information support its interface for use by potential users in rural areas who may be unfamiliar with smartphones making the disease prediction tool as powerful as it is accessible.

5.6.2 UI Tools

React native and expo go: create responsive and user friendly interfaces for different platforms. These frameworks allow developers to write one codebase which runs on both iOS and Android, this not only saves development time but also provides a consistent experience for the user. Utilizing Expo Go adds additional convenience during initial testing and deployment, facilitating swift iteration of UI designs using feedback from target user groups.

5.7 External Interfaces

This part describes how the system interfaces to external entities including the technical details of the connection and protocols to be used.

5.7.1 Backend REST APIs

The backend delivers JSON APIs for user authentication, livestock management, and image predictions based on AI. JWT tokens are used to authenticate securely and image validation just make sure that only valid files are uploaded. These RESTful endpoints adhere to standard HTTP Method and they respond with standard JSON to make the mobile app server communications reliable under any network environment.

5.7.2 Third party Integrations (Future Scope)

In later versions, the app can integrate with SMS APIs for disease alerts, cloud storage for massive image datasets, and online payment system for veterinary consultations. The integration of these services would extend the system coverage in rural areas with no or limited internet access, and add useful features for farmers and vets at the field, ensuring accessibility and sustainable maintenance of the platform.

5.8 Presentation Tier (Frontend)

This layer controls the visual presentation and the input from the user. Need to support login, sign up, image upload, viewing of AI predictions, dashboard for farmer and veterinarian. Written in React Native, the front end is a responsive, cross platform application tailored for use by people with disparate technical skills such that core functions are always intuitive and accessible. It utilizes a component-based architecture, allowing for reusable UI elements and consistent styling across all screens. The tier communicates with the backend via secure API calls to fetch data and submit user requests, ensuring a dynamic and interactive user experience.

5.9 Application Tier (Backend)

This layer performs the main system logic. Built on Node.js, it takes care of processing AI predictions, handling user roles and specific data about livestock, sending notifications and much more, all with secure data management. This level is responsible for business logic and connects to the AI model and database, and enforces security rules such as input validation and rate limiting to protect and preserve the system.

5.10 Data Tier (Database)

This layer is concerned with trusted data services. MongoDB stores all user profiles, disease reports and prediction histories with indexing and schema optimization for performance and reliability. The NoSQL schema enables flexible modeling of data (e.g., user session to complex prediction meta data) for a broad variety of information, such as user sessions, and allows for efficient queries for dashboard analytics and historical review.

5.11 Sequence Diagrams

This section shows how the Smart Livestock Disease Prediction System works dynamically with the help of sequence diagrams. These diagrams represent visually the sequence of interactions between the system and the actors for the main workflows. They describe processes like user authentication, image based disease prediction and dashboard data fetching and make clear the communication between users, application and the external services at each step. These diagrams serve as a crucial blueprint for developers, illustrating the precise order and nature of messages exchanged between the user interface, the backend server, the AI model, and the database. By mapping out these interactions, the sequence diagrams not only validate the system's design logic but also help in identifying potential bottlenecks, ensuring error handling flows, and clarifying the integration points between different architectural components. This visual documentation is essential for both understanding the system's runtime behavior and for guiding future maintenance and feature enhancements. Furthermore, the sequence diagrams highlight the asynchronous nature of key operations, such as the AI model processing an uploaded image while providing user feedback, and the secure token-based authentication flow that protects sensitive user data. They effectively demonstrate how the system maintains responsiveness and a seamless user experience, even during computationally intensive tasks, by clearly separating user interactions, backend processing, and data persistence layers.

5.11.1 Login / Signup Flow

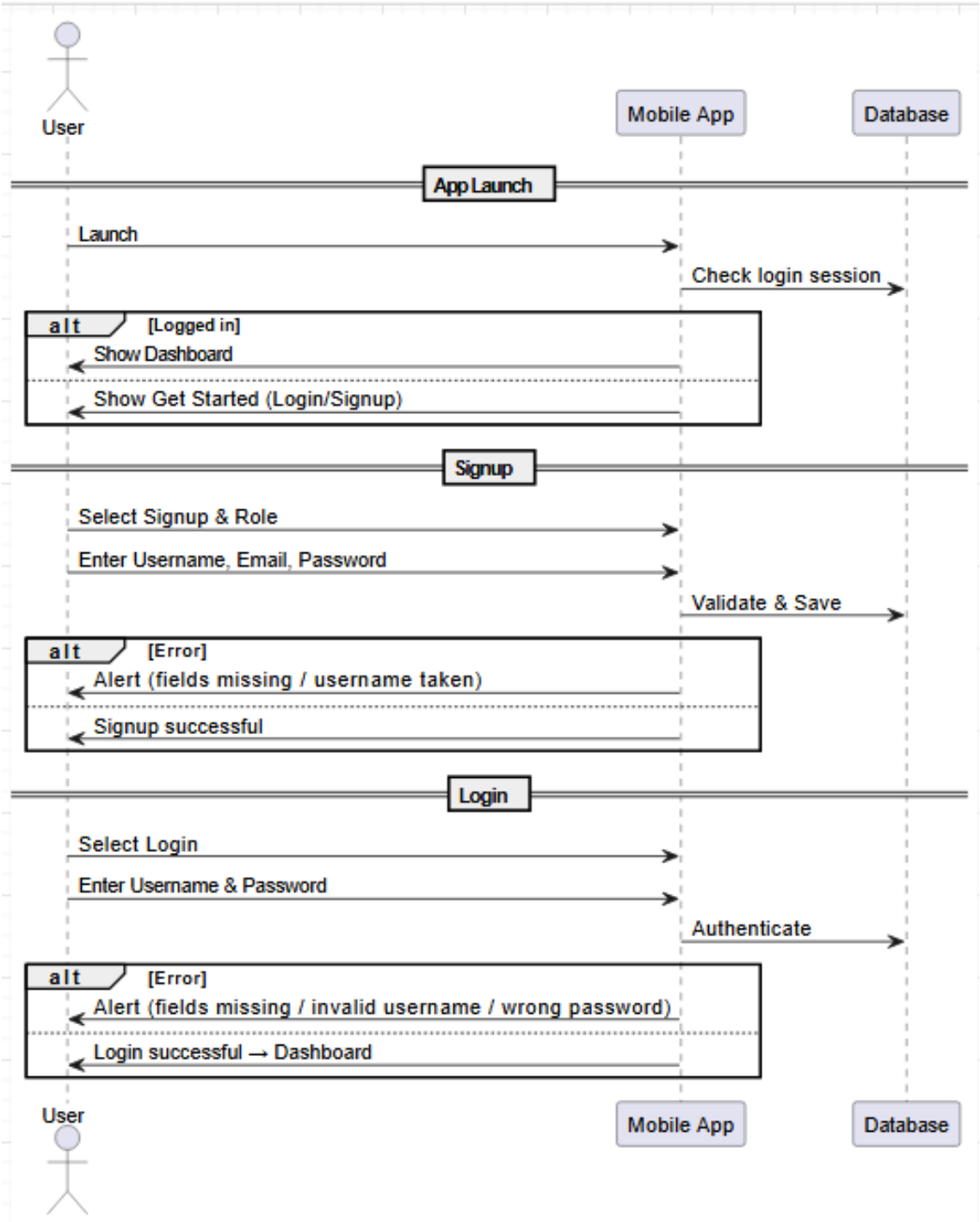


Figure 5.2: Login and Signup Flow Sequence Diagram

This diagram illustrates the user authentication process. It illustrates the way new users register via the sign up form and how existing users get access. The credentials are validated, the user role (farmer or veterinarian) is checked and the user is redirected to the

appropriate dashboard. The credentials are validated, the user role (farmer or veterinarian) is checked and the user is redirected to the appropriate dashboard.

5.11.2 Dashboard and Vet/Farmer Features

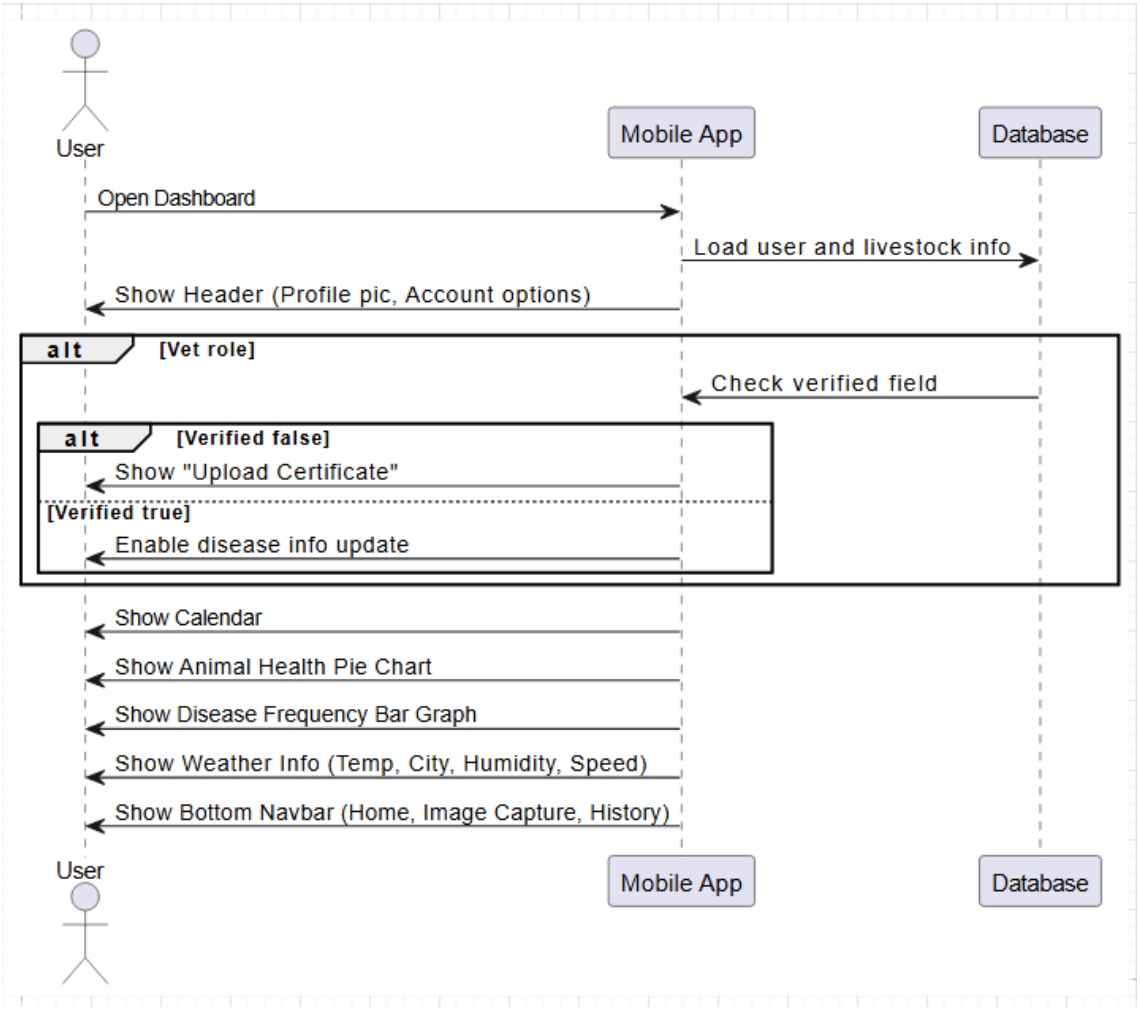


Figure 5.3: Dashboard and Vet/Farmer Features Sequence Diagram

is the main dashboard with the main user actions on it. Farmers can see the statistics of animal health and upload images for analysis, and veterinarians can confirm diseases, attach recommendations and modify case histories. The dashboard provides an at-a-glance overview of herd health metrics, including recent disease trends and the status of pending veterinary consultations. It also serves as a centralized hub for accessing detailed reports, managing farm profiles, and reviewing historical prediction data. The interface is intuitively organized, allowing users of all technical backgrounds to navigate seamlessly between monitoring, diagnosis, and management tasks, thereby streamlining daily operational workflows.

5.11.3 Image Capture and Disease Prediction

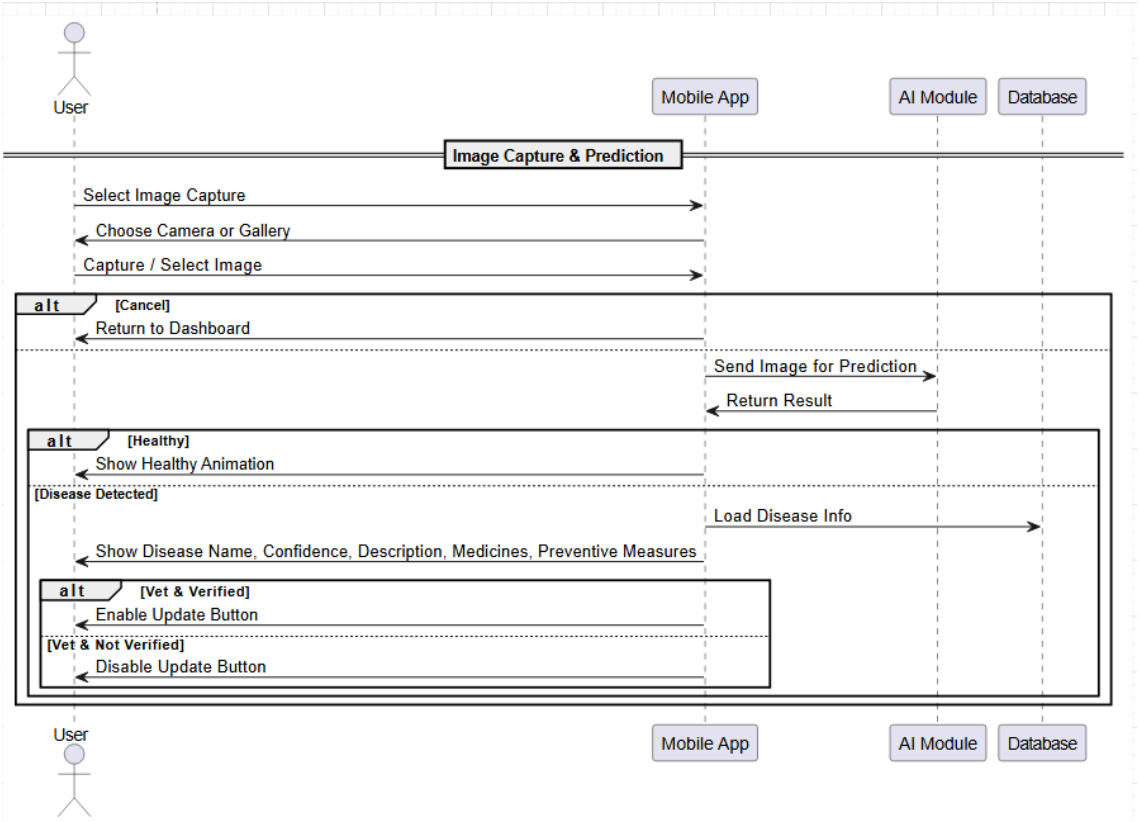


Figure 5.4: Image Capture and Disease Prediction Sequence Diagram

This sequence diagram illustrate the work flow of image based disease prediction in detail. Starting from a user taking a photo of livestock or uploading an existing image using the mobile app. The image is transmitted to the backend server, which processes it and sends it to the AI model for classification. The model detects the disease and returns a confidence score, then the backend queries the database for full treatment details. Then the system collects the complete diagnosis and recommendations and returns it to the user via his/her device. The diagram also shows how the system react to errors (invalid image or undetermined result). The diagram effectively captures the critical handshakes between system components, including the preprocessing steps on the backend before the image is fed to the AI model. It also delineates how the result is packaged with supplementary data—such as prevention methods and medicine lists fetched from the knowledge base—to create a comprehensive report for the end-user. This visual flow confirms the system's ability to integrate machine learning inference with dynamic data retrieval, ensuring that farmers receive actionable, context-rich health insights in near real-time.

5.11.4 History and Admin Module

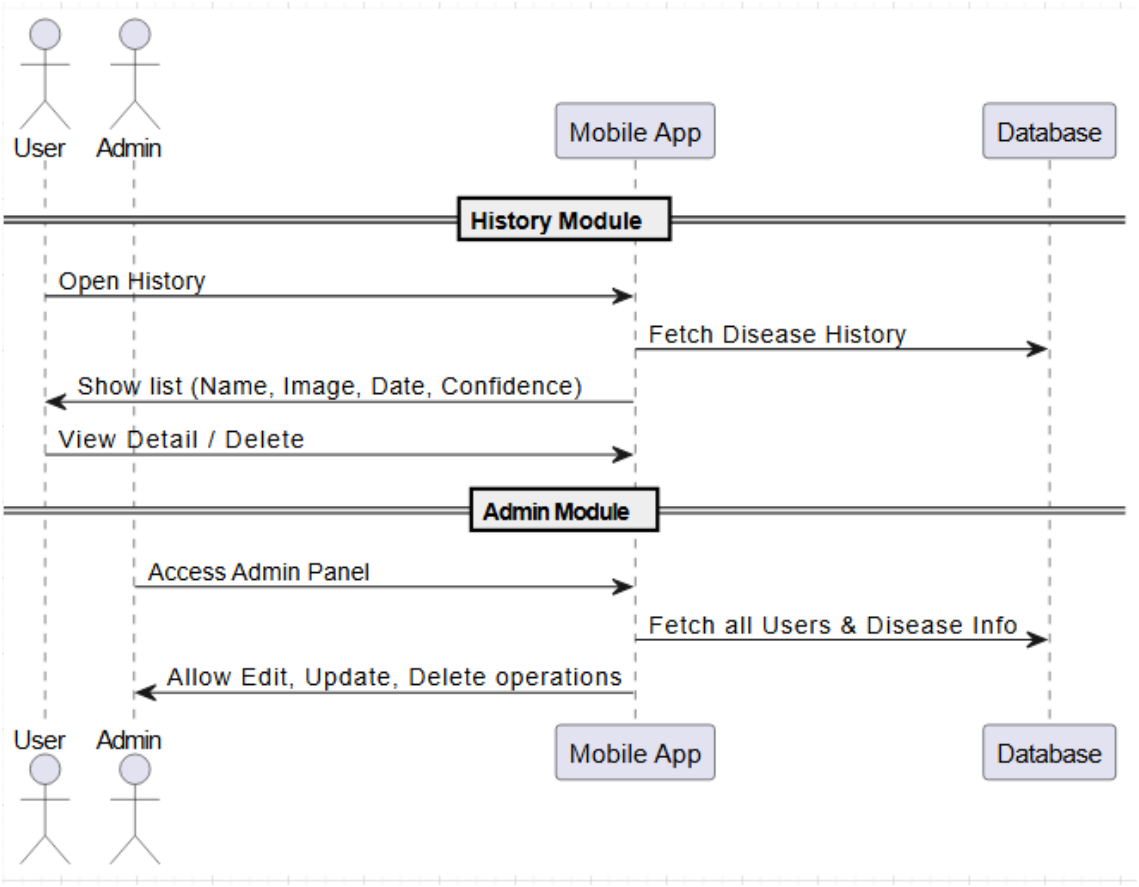


Figure 5.5: History and Admin Module Sequence Diagram

This figure demonstrates two basic operations (historical access of data and system management) from the system side. It explains how a farmer or vet can have access to their previous predictions records, with a query to a secure database to obtain and show the chronological history of diagnoses. At the same time, admin user can work with the administrator workflow, managing the registered users, handling the veterinarians validation requests and updating the disease information database. The flows illustrate how the system implicitly uses role based access control so that users can only access their own data and the admins are able to have the oversight they need to ensure the integrity of the system. The handling of errors like empty history or trying to access without permission is also represented. These parallel workflows highlight the system's scalable architecture, where data access patterns are efficiently separated by user roles to optimize performance and security. The diagram also emphasizes the audit trail created by logging administrative actions, which is essential for maintaining accountability and system transparency.

5.12 ERD Diagram

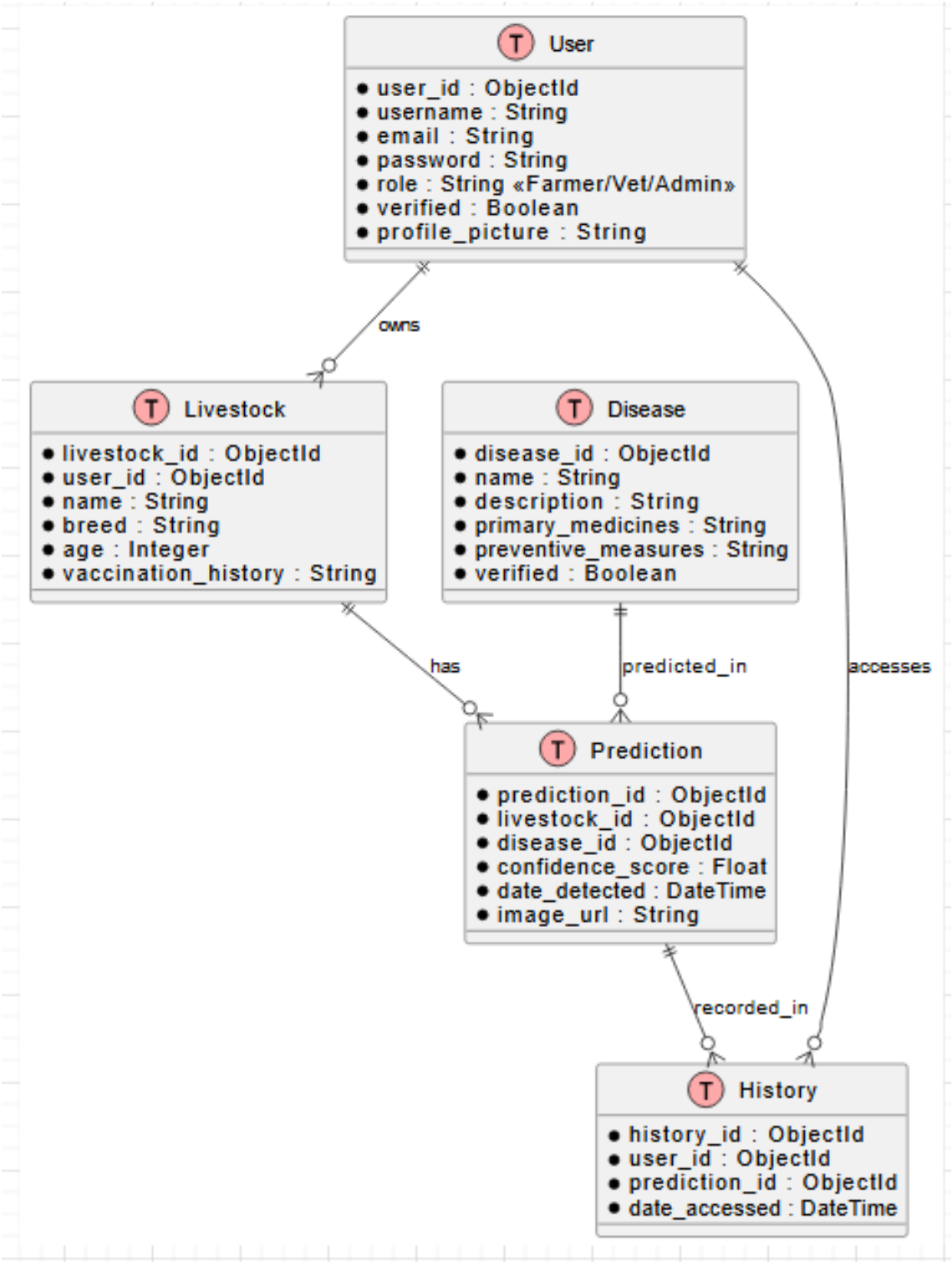


Figure 5.6: Entity Relationship Diagram

Figure above displays the ER diagram; entities such as User, Livestock, Disease, and Prediction are illustrated. It also illustrates how each entity is linked, so it tells you the dependent foreign key information for each module, which can be useful for maintaining data integrity. For instance, the `Prediction` entity serves as a hub, connecting a `User` (the farmer or veterinarian who made the diagnose) to a particular `Livestock` and the detected `Disease`. This makes it possible to link back every prediction. The resulting relationships, e.g., one to many relation from User to Livestock and many to one relation from Prediction to Disease, provides a strong notion of ownership and data integrity. Among the attributes of entities, like `confidence_score` in Prediction or `verified_status` in User, are also represented to indicate which specific datum of the system helps to realize its core functions.

5.13 Activity Diagrams

For the three types of users Farmer, Veterinarian and Administrator, the corresponding activity diagrams are provided to illustrate how they operate with the system. They provide a clear view of the operations the users of the given type can perform. Each diagram captures, for its respective role, the role specific decisions, activities and reactions of the system. For instance, the farmer is a disease detection pipeline, and the veterinarian is a case verification and knowledge base enhancement chart. When considered together, these diagrams showing the two parallel and yet independent processes illustrate how the system architecture instantly allows the role based, task consistent and, most importantly, user friendly experience to cater for the varying user groups like individual/family farmers and agricultural organization managers whose needs and expectation are so vastly different.

5.13.1 Farmer Activity Diagram

The farmer flow shows the farmer activity in our app for logging in, uploading or taking livestock photos, receiving AI predictions, and deciding if they want to consult a veterinarian. It allows to see the entire flow of the farmer in a simple and clear way. The flowchart also illustrates how the farmers are aided, from authentication to disease detection to interpreting the results. It also shows decisions, such as whether to take veterinary consultation following the confidence of AI prediction. In addition, the diagram highlights the user friendly interactions that focus on a seamless user experience, enabling even non technical farmers to easily take actions when needed. In general, it makes sure that the journey of a farmer is streamlined, assisted and productive for early disease detection and preventive management.

Visual flow: the farmer starts by opening the application and logging in, and then has access to a dashboard to see an overview of the health status of his/her livestock. The main

route, in turn, is to the image capture page where the user can either take a new photo with their device's camera, or select an existing image from their gallery to upload for analysis. After the image is processed, the system presents the AI based prediction along with a confidence score and links to immediate recommendations from which the farmer can accept the result, save it for tracking or escalate the case by asking for a consultation with a verified veterinarian through integrated system request.

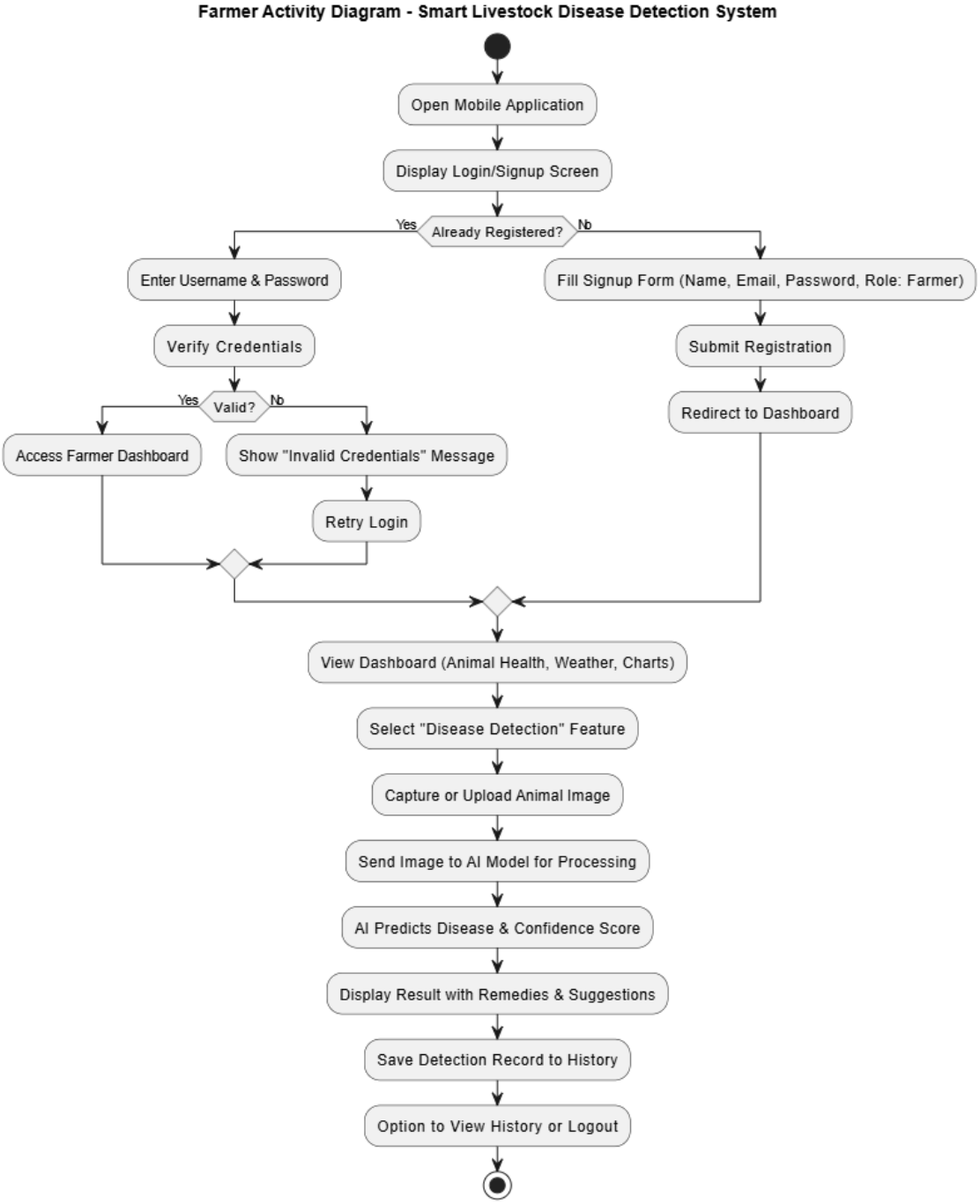


Figure 5.7: Farmer Activity Diagram

5.13.2 Vet Activity Diagram

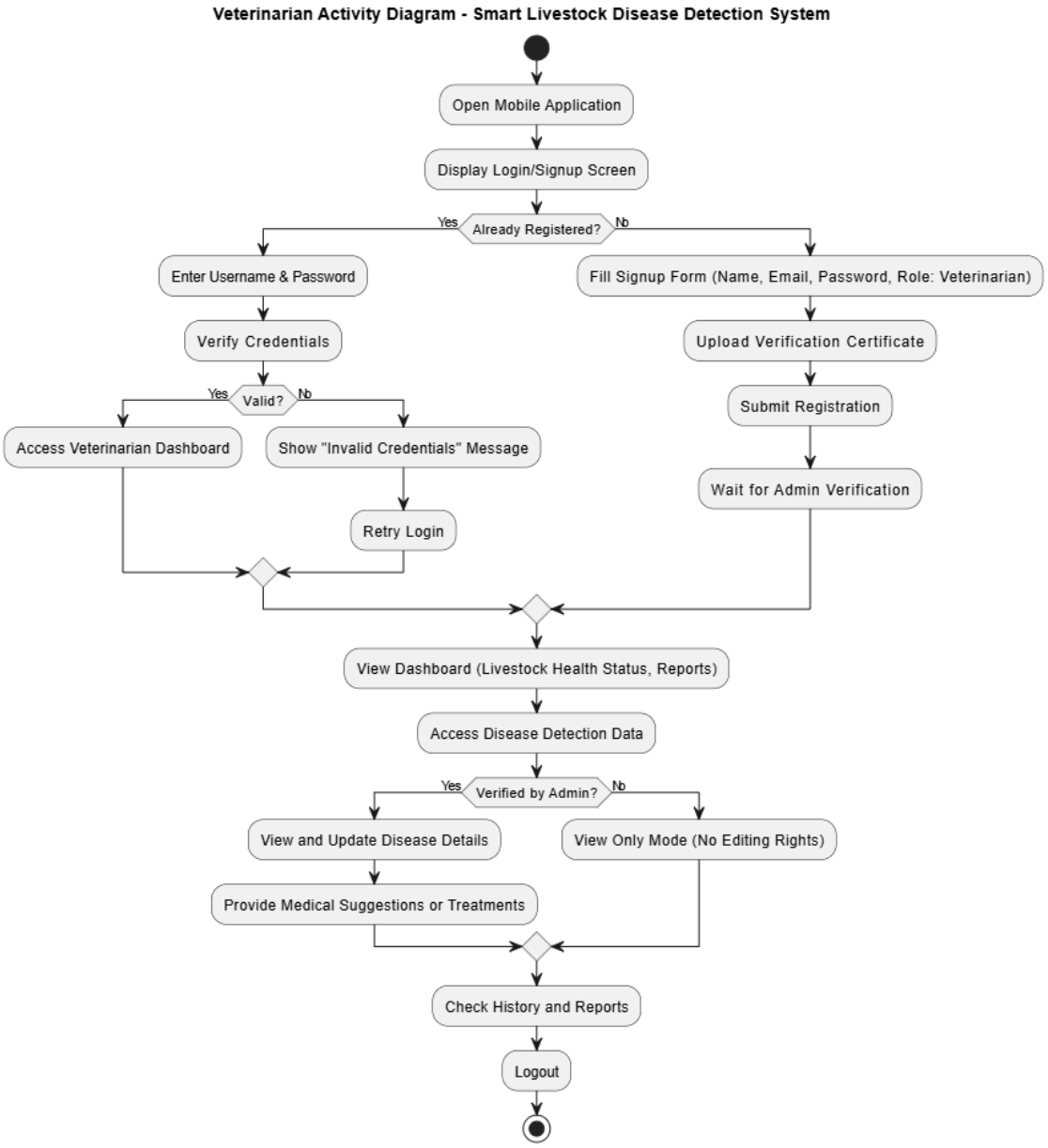


Figure 5.8: Vet Activity Diagram

The veterinarian flow chart describes actions that vets perform once logged in. It involves checking AI outputs, disease records updates and treatment proposals. Professional review is guaranteed with better reliability of predictions based on AI. The diagram also shows how veterinarians access the prediction data, confirm or alter diagnoses, and send updates to farmers via the application. This process ensures that human intelligence is behind the automated predictions, increasing accuracy and confidence in the system. And it shows how vets are part of continuous learning by editing disease data, and recommendations in

the app. The workflow becomes active when the vet logs in with valid credentials and lands on their dashboard looking at cases awaiting their attention as determined by the AI. For each case, the vet reviews images that have been uploaded, contrasts the AI diagnosis with their clinical knowledge, and validates the outcome or submits an alternative assessment that includes detailed notes. Then, the vet may further modify the shared disease database, detailing symptom descriptions or treatment procedures in accordance with the most recent field experiences and medical knowledge. This symbiotic relationship between the AI and the vets, continuously improves the model's accuracy while creating a dynamic authoritative platform for the management of livestock health.

5.13.3 Admin Activity Diagram

Administrator can manage users, system core data, and disease records in the admin activity diagram. It provides the control of the admin on the backend aspects, including monitoring users, updating data sets, and managing the system performance. In addition, the flow chart describes how the admin maintains data integrity, security, and freshness. It illustrates the importance of the admin in managing stability, logs, and troubleshooting. The process of the admin includes managing permissions, installing AI model updates, and tracking the overall app health. This guarantees that the system delivers high quality service that is viable, safe and provides right answers to veterinarians as well as farmers. The flow of activity starts with the authentication of the administrator and then the administrator is granted access to the centralized dashboard which acts as a command center. Now the admin can get down to specific management functions: user management to view all accounts, verify veterinarian credentials, and change user roles or permissions. At the same time, the system data section allows the system administrator to manage updates to the disease knowledge base, introduce new symptoms or treatments, as well as manage the information used by the AI model to diagnose. This organized flow of work depicted in the figure takes into account the sequential and conditional aspects of the duties or obligations of the admins, in that they warrant adequate care of each vital part of the system to enable it to run dependably and securely.

The flow of activity starts with the authentication of the administrator and then the administrator is granted access to the centralized dashboard which acts as a command center. Now the admin can get down to specific management functions: user management to view all accounts, verify veterinarian credentials, and change user roles or permissions. At the same time, the system data section allows the system administrator to manage updates to the disease knowledge base, introduce new symptoms or treatments, as well as manage the information used by the AI model to diagnose.

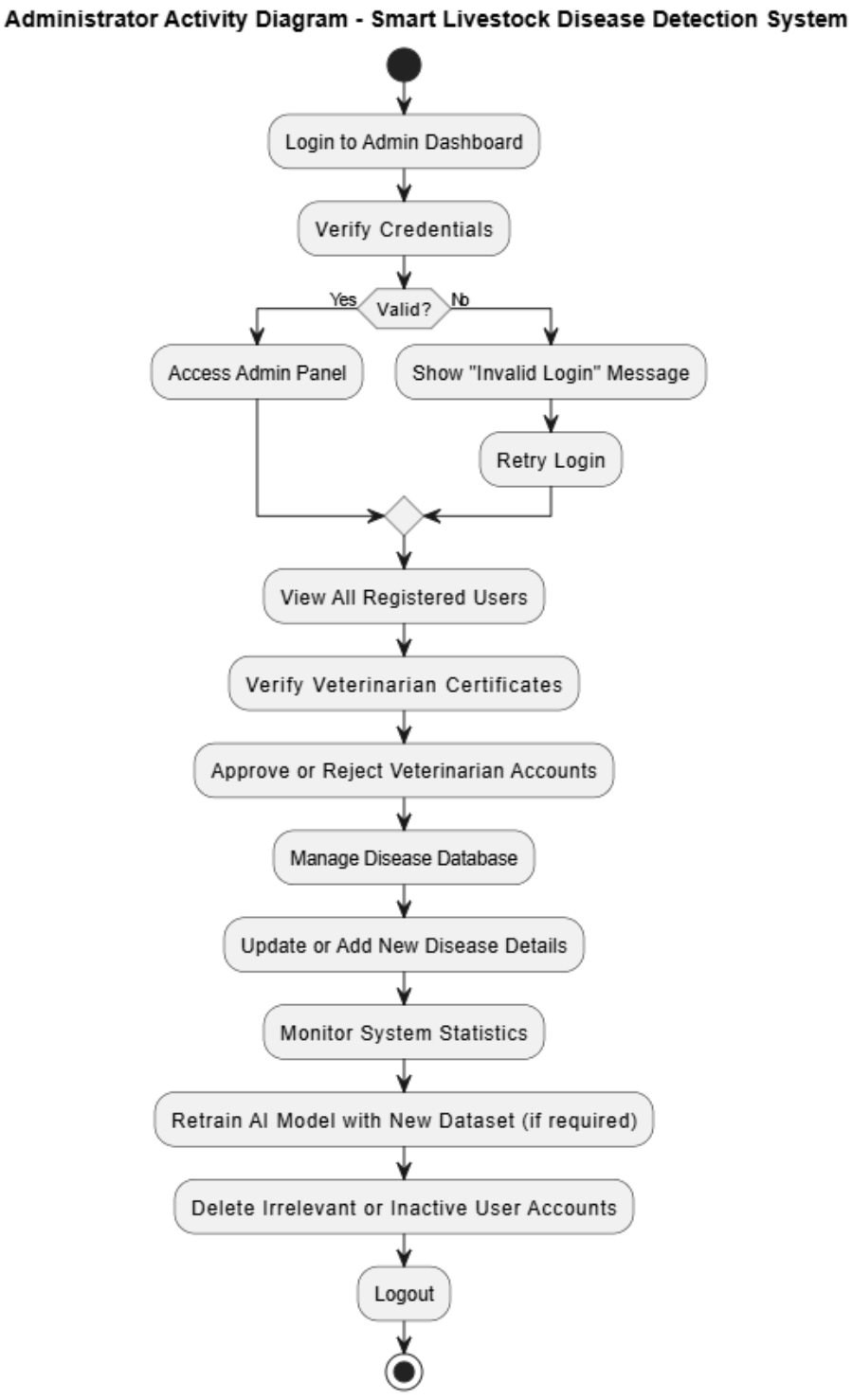


Figure 5.9: Admin Activity Diagram

5.14 Summary

This chapter provides a full exposition on design and architecture of the Smart Livestock Disease Prediction System. It addresses both high level and low level concerns and explains how all the elements (UI, logic, database, and so on.) work together to build a consistent and intelligent mobile application. This chapter discussed design constraints, assumptions and feasibility analyses to keep the design realistic, cost effective, and user friendly in the environment of implementing. The result of the feasibility study found out that the selected technologies such as react native, node.js, mongodb and tensorflow are best fit to bring in the required qualities of efficiency, scalability and high performance within the given constraints. The architectural design is based on separation of concerns and is layered in such a way that the system has well defined layers: presentation, application, and data which makes the overall system more maintainable and scalable. The demonstration of API endpoints and database optimization techniques demonstrate the level of fluid interaction with the system that preserve data integrity. Additionally, a total of three workflow, activity and sequence diagrams were used to show the different types of users; farmers, vets and admins, their access levels and possible interactions. These diagrams focus the logical sequence of the actions, the decisions, and the data exchanged, and helps comprehension of the operational mechanism and the interdependencies of actions among the modules. In general, this chapter paves the way with a robust design for the further implementation and testing phase. It helps to make sure that the Smart Livestock Disease Prediction System are not only effective and robust but also easy to use and well utilized by the targeted end users. Design choices made here are what ultimately lead the project to fruition and make it successful in its primary goal, a reliable, AI backed livestock disease detection solution that enables farmers and veterinarians improve animal health, decrease the spread of disease, and encourage sustainable livestock management.

Chapter 6

System Testing and Evaluation

6.1 Overview

System testing was performed on the Smart Livestock Disease Detection mobile application to check whether it satisfies all its requirement specification. This was to validate that the application gives correct disease predictions, has a good interface for farmers and veterinarians, and manages well the data storage. Qualitative and quantitative evaluations were conducted to evaluate the functionality, usability, performance, and security of the system in the context of practical use.

6.2 Graphical User Interface (GUI) Testing

GUI testing concentrated on the observing of the general appearance and the response time of mobile applications. The login and sign up screens, dashboard design, image upload, prediction screens, and disease history were among the services tested. Everything was tested to make sure buttons and input fields respond as expected and that alerts are shown when the user provides invalid or incomplete information. Visual elements such as the calendar view or pie and bar charts representing disease trends were particularly included. They were tested on different screen sizes and on various devices for alignment and readability to provide any potential user with a uniform and pleasant user experience.

6.3 Usability Testing

Usability tests were real farmers and veterinarians who I observed them using the app to determine if they were able to easily understand how to use it. The test participants tried out functionalities like image upload, disease prediction, fetching older results. They said the app is what pushes them to use online banking. There were a couple of small issues identified during the initial on boarding phase such as confusing instructions for first time

users. They were added with the aid of several short instructions going from message to message. Generally, the response indicated that user enjoyed using the app (even non technical users), and that it was useful in handling their livestock health.

6.4 Software Performance Testing

Performance testing was a test that measured how fast and how efficiently the system responded to user inputs. The app was benchmarked for speed of upload and prediction to produce image based predictions within a few seconds. Load testing was also performed to simulate multiple users working with the app concurrently. The backend successfully processed the load, without a single failure or noticeable delay. The AI model (MobileNetV2) has been compressed for faster inference and additional tricks like caching have been applied to boost speed on recurrent similar requests.

6.5 Compatibility Testing

Compatibility checks were carried out to ensure that the app runs well on various Android devices and versions. Testing included screen resolution changes, permissions for camera and storage access, and app performance on a variety of hardware. The app was consistently good and appeared to layout its interface appropriately on all the devices we tested.

6.6 Exception Handling

This stage was all about testing how the app reacts to unexpected scenarios like slow internet, blank fields, incorrect passwords, or unsupported image types. Each of these resulted in an appropriate (and helpful) alert rather than a crash. When the AI model was unable to classify an image, a simple message was shown to the user. This made the system more robust and friendly in real applications.

6.7 Load Testing

The system was stress tested for performance. We simulated users uploading images and predicting at the same time. The MongoDB REST API was monitored for slow queries. The overall system remained stable and responsive with a moderate number of concurrent requests, demonstrating that it could scale for field deployment. The overall system remained stable and responsive with a moderate number of concurrent requests, demonstrating that it could scale for field deployment.

6.8 Security Testing

Security testing was conducted to verify that user data was private and the system was accessed in a secure manner. Authentication was checked via JSON Web Tokens(JWT), so that only registered users could log in and use their features. All user information, including profiles, livestock photos, and prediction records were stored securely in MongoDB. The API endpoints were also tested to ensure the endpoints were not vulnerable to common attack vectors. These tests showed the app offers a safe environment for users to keep their information.

6.9 Installation Testing

Installation testing was conducted to ensure that the application is installable and executable on real Android devices. All required permissions, including those for the camera and storage, were requested appropriately during testing. After installing, the app started up and communicated with the backend services without any errors, indicating a seamless installation experience for the users.

6.10 Evaluation Metrics and Analysis

The AI model of the system was quantitatively evaluated by standard performance metrics including accuracy, precision, recall, and F1 score. Based on the test data, MobileNetV2 model holds promising results with 84.4% in terms of accuracy for correct recognition of livestock diseases. Survey responses from users of the system also showed a high level of satisfaction with the ease of use and dependability of the system. Farmers and vets liked the quick analysis and simple presentation of the results. In general both technical and user based evaluation demonstrated that the system is capable to fulfill its designed aim and has good performance under the real operating environment.

6.11 Strengths and Limitations

The Smart Livestock Disease Prediction System is the world's first AI based solution that can work offline on a mobile phone to aid in early disease detection. It offers farmers and vets rapid, data driven insights on the health of animals, helping them to make better informed decisions on management and treatment. Nonetheless, there are still some restrictions. The app currently requires an internet connection and is limited to a small number of diseases. The quality of images in captured also affects prediction accuracy. Later on, the system may be enhanced by incorporating offline functionality, increasing the disease database, and optimizing the model for more species of livestock.

6.12 Test Cases

Here are the main test cases of the system. Each test case tests a particular feature and verifies the expected output of the system.

Table 6.1: Test Case 01: User Registration and Login

ID	Test Case 01
Description	Verify that users can register and successfully log in using valid credentials.
Setup	* The app is installed. * No existing account for test user.
Input	Registration: username, email, password, role (Farmer). Login: same email and password
Instructions	1. Complete registration form with valid data. 2. Click "Create Account". 3. Use registered credentials to log in.
Expected Result	Account created successfully, then user logged in and redirected to dashboard.
Actual Result	Registration and login completed without errors.
Verdict	Pass

As listed in column 2 of shown in Table 6.1, this test verifies that the basic mechanism of the system for authentication is working. It is therefore possible for a new user to create a valid account (with credentials validated) and for an existing user to log into the application securely. The test verifies that registration and login process are integrated properly, and it checks for security implementation for user data. Passing this test indicates that the system fulfils the authentication requirements.

Table 6.2: Test Case 02: Profile Management and User Logout

ID	Test Case 02
Description	Ensure users can update profiles and securely log out.
Setup	* User is logged in. * Profile information is loaded.
Input	Updated name, email, and profile picture. Logout command.
Instructions	1. Navigate to profile settings. 2. Edit fields and save. 3. Initiate logout from menu.
Expected Result	Profile updates saved, session terminated, user redirected to login.
Actual Result	Changes persisted; logout successful.
Verdict	Pass

Table 6.2 concerns the management of the user's personal data and the logout from the system. This capability is essential to ensure that data remains accurate and that user accounts are not compromised. The test verifies that profile updates are saved correctly to

the database, and that log out removes session information. This guarantees that all privacy and security standards are met and at the same time gives the user emptying to their own data in the application.

Table 6.3: Test Case 03: Capture and Image Upload

ID	Test Case 03
Description	Verify users can capture or upload livestock images for analysis.
Setup	* User is logged in. * Camera/gallery permissions granted.
Input	Livestock image from camera or gallery.
Instructions	1. Select "Capture Image" or "Upload Image". 2. Take/select a photo and confirm preview.
Expected Result	Image preview shown with options to diagnose, retake, or cancel.
Actual Result	Image preview displayed correctly.
Verdict	Pass

The image capture and upload features, which are essential for the flow of disease detection, are tested according to the test case in the Table 6.3. This test verifies that users can successfully use their cameras or galleries as the source of visual information for analysis. The verification allows to check image file manipulation, permission management and preview features from the user's point of view. A passing run indicates that the system meets the functional requirement that the primary input for AI based disease diagnosis can be reliably obtained from a user.

Table 6.4: Test Case 04: AI Powered Disease Diagnosis

ID	Test Case 04
Description	Validate that AI model processes images and returns detailed results.
Setup	* User is logged in. * Valid livestock image is uploaded.
Input	Uploaded livestock image selected for diagnosis.
Instructions	1. On image preview, click "Diagnose". 2. Wait for AI processing.
Expected Result	Disease name with confidence score, description, medicines, and prevention methods displayed.
Actual Result	Complete diagnosis results shown within expected time.
Verdict	Pass

The basic diagnostic operations of the system are tested in Table 6.4. This test check that the MobileNetV2 model is able to process the livestock images that are you uploaded and produce full diagnostic reports. It tests the correctness of disease classification, confidence scoring, and suggesting treatment. This validation together with preventive actions is also proved to be meaningful showing the usability of the ai based diagnosis module.

Table 6.5: Test Case 05: Prediction History Management

ID	Test Case 05
Description	Ensure users can view and manage past predictions.
Setup	* User is logged in. * At least one prior prediction exists.
Input	Request to view history; selection of a record.
Instructions	1. Navigate to History section. 2. View list and select a record for details.
Expected Result	Chronological list with thumbnails, dates, disease names, and confidence scores displayed.
Actual Result	History loaded correctly; details view accessible.
Verdict	Pass

Table 6.5 outlines the test of historical data management, which is an essential capability for monitoring the health of livestock over time. This test ensures that users are able to view the entire history of their predictions in a neat, chronological manner along with associated metadata. It guarantees that the system keeps a correct record of previous diagnoses and that the presentation allows to drill down when pressed to any item on the list. This feature facilitates long term health tracking, allowing a user to detect trends or recurring problems within their herd, as required.

Table 6.6: Test Case 06: Dashboard Visualization

ID	Test Case 06
Description	Confirm dashboard loads health charts, weather, and calendar.
Setup	* User is logged in. * Prediction history exists for analytics.
Input	User login or dashboard refresh.
Instructions	1. Log in or refresh dashboard. 2. Observe displayed charts and data.
Expected Result	Pie chart (healthy vs diseased), bar graph (disease frequency), weather info, and calendar shown.
Actual Result	All visual elements loaded correctly.
Verdict	Pass

Test FR.06 in Table 6.6 checks the system's ability to visualize data. This test verifies that users have a complete overview of the health of their stock by means of graphical indicators in combination with ambient relevant information. It checks the inclusion of weather data and the calendar to inform decisions. When the run is successful, it indicates that the dashboard is able to provide actionable insights in the form of concise visual analytics, i.e, that it satisfies the visualization needs for managing the health of livestock effectively.

Table 6.7: Test Case 07: Veterinarian Verification

ID	Test Case 07
Description	Validate certificate upload and status update process.
Setup	* Veterinarian account is logged in. * Valid certificate file available.
Input	Certificate file (PDF/JPG/PNG) uploaded by veterinarian.
Instructions	1. During registration or profile edit, upload certificate. 2. Submit for admin review.
Expected Result	Certificate accepted; status set to "Pending"; admin notified.
Actual Result	File uploaded; verification status updated accordingly.
Verdict	Pass

The test for veterinarian credential verification, an essential part to ensure system integrity, is shown in Table 6.7. This test checks that veterinary professionals can upload certification documents for administrative review. It verifies correct file format, document security and status traceability in the verification process. Passing this test guarantees that the diagnostic platform, whose quality standards and reliability are blindly trusted by its users, would not provide access to its advanced features to anyone but truly qualified professionals.

Table 6.8: Test Case 08: Admin Requirements

ID	Test Case -08
Description	Ensure admin can manage users, verify vets, and update disease info.
Setup	* Admin account is logged in. * Pending vet verification exists.
Input	Admin actions: view users, approve vet, edit disease details.
Instructions	1. Access admin panel. 2. Perform user management and disease updates.
Expected Result	Changes applied successfully; system confirms operations.
Actual Result	User list updated; vet status changed; disease DB edited.
Verdict	Pass

The administrative control is tested as shown in Table 6.8. This test exercises the system administrator full range of management of users, verification of veterinarians credentials, maintenance of disease information, etc. It verifies an admin can do his/her job effectively via the admin portal. Passing this test indicates that role based access control and admin workflows are properly implemented and the system secure and contents are valid. This comprehensive verification is crucial for maintaining system integrity and ensuring that sensitive data and core functionalities are only accessible to authorized personnel.

Table 6.9: Test Case 09: Real time Notification and Alerts

ID	Test Case 09
Description	Verify users receive notifications for system events.
Setup	* User is logged in with notifications enabled. * A prediction is completed or vet status updated.
Input	System event: disease prediction completed.
Instructions	1. Complete a disease diagnosis. 2. Check for notification on device/app.
Expected Result	Push notification or in-app alert received about the prediction.
Actual Result	Notification generated and delivered successfully.
Verdict	Pass

The test notification system in Table 6.9 will test live communication with the application. This test ensures that users are alerted in a timely manner for important system events, including completed diagnostics and verification status messages. It also verifies that the app has the necessary integration with the device notification service and fallback mechanism in the form of in app alerts when push notifications are not enabled. Running the test shows that the system is sufficient in keeping user notified for such activities, and the user is more likely to respond fast and stay engaged.

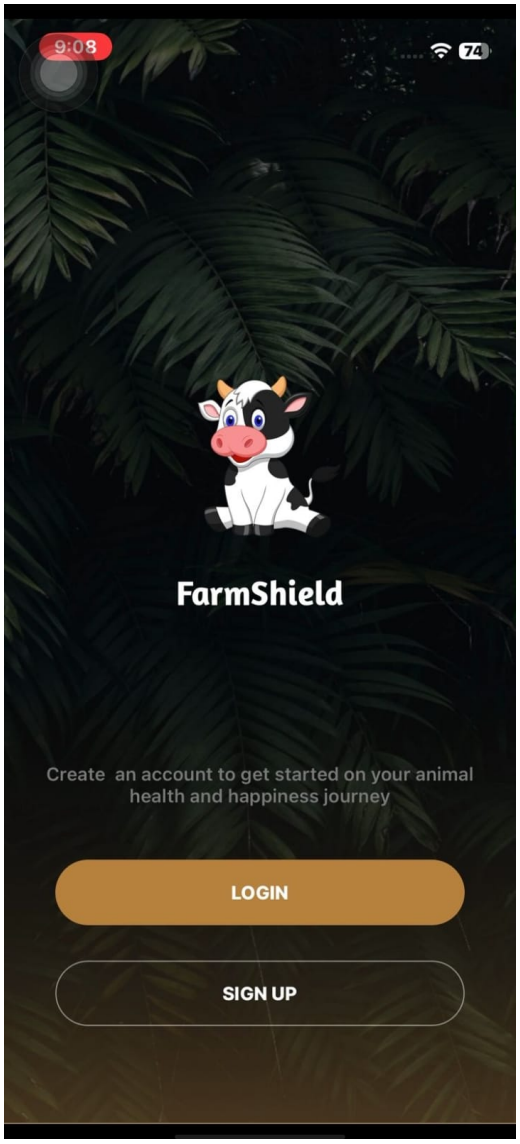
Table 6.10: Test Case 10: Error Handling and Data Validation

ID	Test Case 10
Description	Ensure system validates inputs and handles errors gracefully.
Setup	* App is running. * Test with invalid inputs and simulate errors.
Input	Invalid email format; empty required fields; system error trigger.
Instructions	1. Attempt registration with invalid email. 2. Submit form with empty fields. 3. Simulate a network/server error.
Expected Result	Clear error messages displayed; system logs errors; no crashes.
Actual Result	Validation errors shown; system remained stable; logs created.
Verdict	Pass

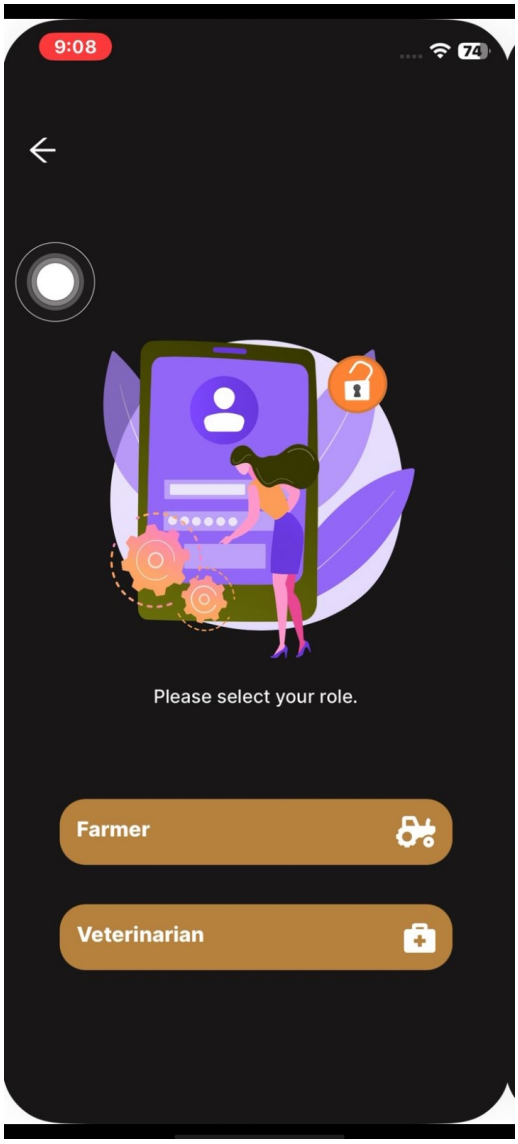
The full error handling test in Table 6.10 is the system stress test: its purpose is to assess how well the system and users can cope with failure. This test ensures that the application detects an invalid input, gives the user meaningful instructions and does not crash or freeze under unexpected scenarios. It bats that developer debugging logs are properly filled and system crashing is prevented. If it completes successfully, it means that validation is fully in place, and gracefully failing at each step of the user journey keeps the overall user experience smooth and pleasant.

6.13 Front end Interface Screenshots

Here are some snapshots of the Smart Livestock Disease Prediction mobile application's key interfaces, through which the layout, navigation flow, and AI based prediction features are demonstrated.



(a) Front Page with Login/Signup Options

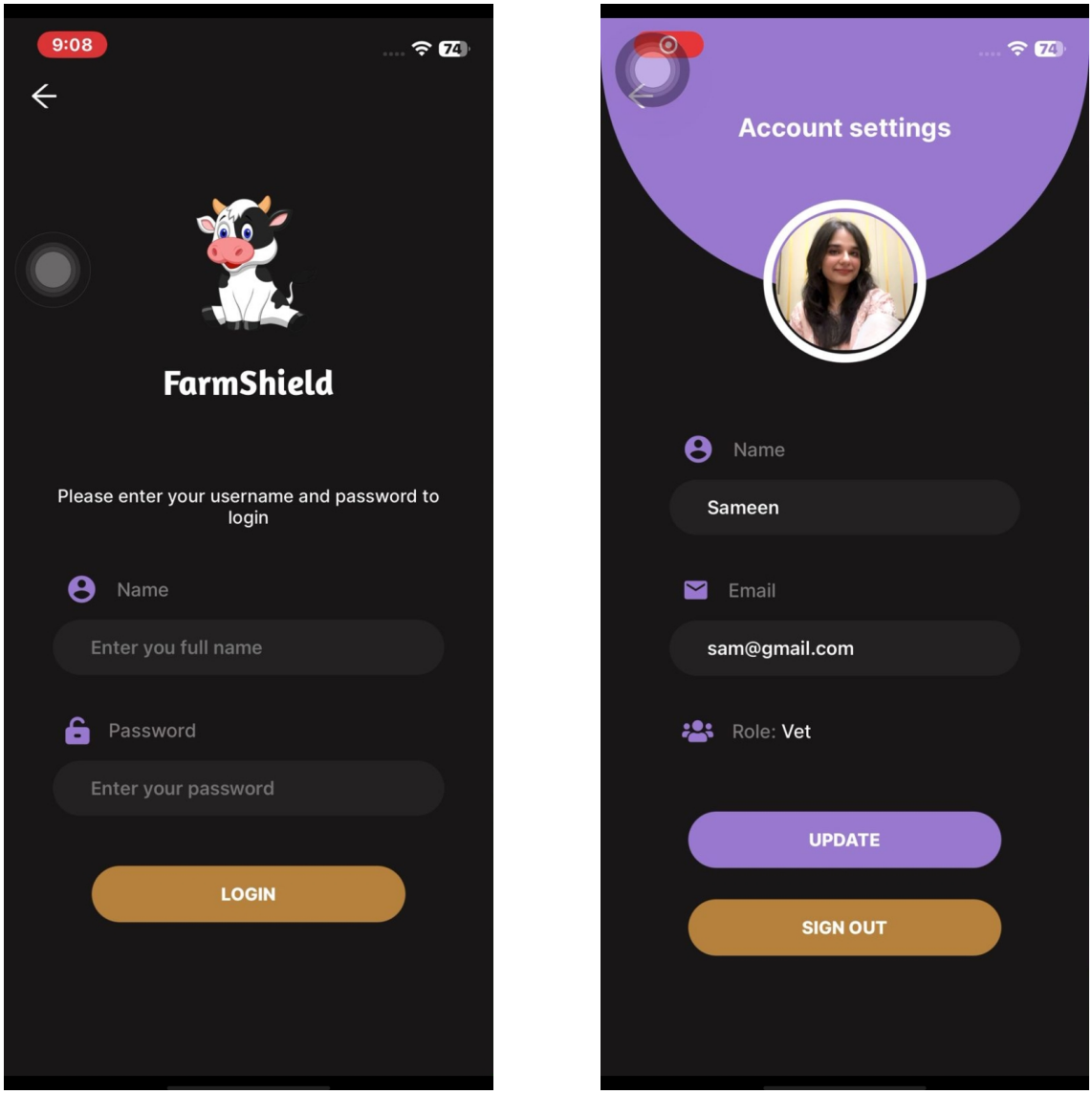


(b) Role Selection Screen (Farmer or Vet)

Figure 6.1: Front Page and Role Selection Screens

The user interface is simple and clean to be friendly with all users from different levels of user experience. User engagement with the system starts from the front page, which offers two main choices: to **LOGIN** or to **SIGN UP**, as shown in Figure 6.1(a). After choosing "SIGN UP", a user must select a role on the role selection screen, Figure 6.1(b),

and choose their role as a **Farmer** or a **Veterinarian**. This initial role assignment is important because it defines what features, privileges, and UI elements the user will have access to through the rest of the application, providing a personalized experience tailored to their particular requirements in the livestock management ecosystem.



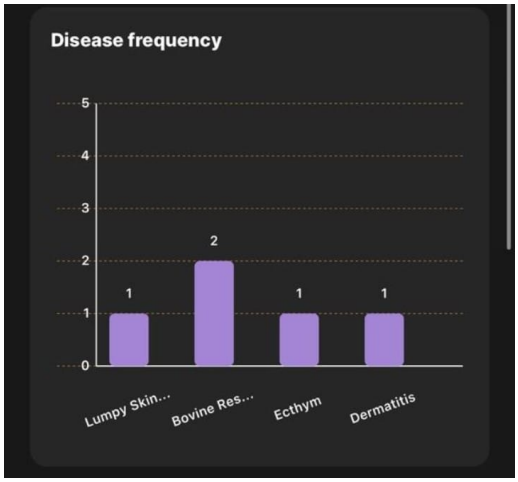
(a) Login Screen (b) Account and Profile Management

Figure 6.2: Login and Account Settings Screens

User Login and User Profile are implemented as separate pages (interfaces) with privacy and security. The login interface is shown in Figure 6.2(a); the user must input a registered user name and password to the system left with a simple and straightforward door to the system core. When authenticated, the user is able to access to own account settings as shown in Figure 6.2(b). Through this screen the user can see aspects of their

profile and manage it, for example by changing their name, contact information, or profile picture. These screen are designed with security and usability in mind to allow the user users to efficiently manage their personal information and credential. The separation of authentication and management interfaces is well defined, to ensure a secure and organized user experience.

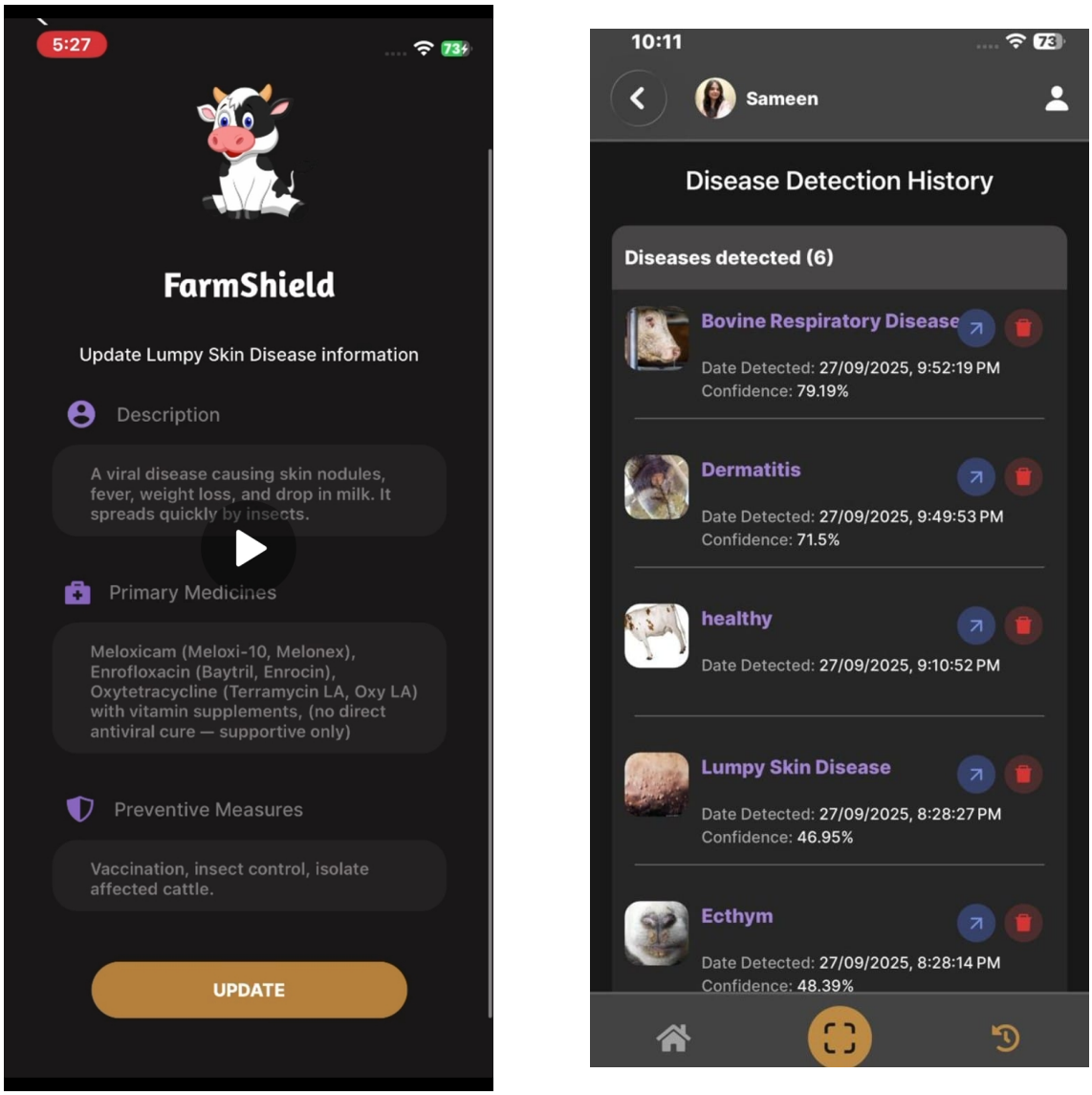
(a) Dashboard Showing Health Overview



(b) Disease Frequency Visualization

Figure 6.3: Dashboard and Disease Frequency Visualizations

After a valid login, the users naturally arrive at the main application dashboard, which serves as the place where they can monitor and manage at a glance. This panoramic view is seen in (a) of Fig. 6.3 is a visual representation of animal health in the stock. It has a strong graphical disposition with a pie chart that allows to obtain a quick snapshot about the health/disease condition of the animals and a bar graph (reported in Figure 6.3 (b)) indicates the frequency of the diseases. These reports help farmers and vets to spot emerging health trends or outbreaks in their herds. The dashboard brings together vital information and presents it in an easy to understand interface so that users are better equipped to make the right decisions based on live data, such is the importance of this for proactive livestock management and prompt response. It has a strong graphical disposition with a pie chart that allows to obtain a quick snapshot about the health/disease condition of the animals



(a) Disease Information Screen

(b) Disease History Records

Figure 6.4: Disease Information and History Screens

The system has specific interfaces for detailed disease knowledge and for reviewing historical diagnostic records. Registered veterinarians are able to view and edit full disease profiles (Figure 6.4(a)) which contains the disease description, the primary medicines and the precautions page by page in a web form. This ensures that the knowledge base of the AI model is both accurate and up to date. In the meantime, users have the option to access all their historical interactions with the system through a chronological log shown in Figure 6.4(b). The History screen along with past predictions with time stamp and confidence score of the AI for every prediction is shown in Figure 6.4. This allows users to monitor the health of their herd over time, and to review the context of previous veterinary decisions.

Chapter 7

Conclusions and Future Work

7.1 Conclusion

Smart Livestock Disease Prediction System is a good example of the combination of AI and mobile cloud computing to provide real time, reliable and accessible disease screening for cattle. Based on a MobileNetV2 convolutional neural network architecture adapted for mobile devices, the system provides the possibility for farmers to take pictures or upload them from their livestock and get instant predictions of diseases with associated confidence scores and suggested preventive actions. The native mobile application (for Android and iOS) with React Native provides simple and clean user interface, designed for users with no-to-limited technical skills, it'll allow a wide spread adoption among farming communities in rural regions. The system offers veterinarians a powerful medium through which to review AI produced diagnoses, verify disease information, and record comprehensive livestock health data in a structured digital format. This allows the data to be available for animal health monitoring and less prone to errors. With the addition of MongoDB as a backend database for scalability purposes, the personal information, image data and prediction history are encrypted and stored securely, and the server architecture of Node.js/Express.js provides a suitable performance and reliability. With role based access control, thereby differentiating among farmers, veterinarians and administrators, the platform tailor its functionalities to the role of each user.

In terms of technology, the work is a good illustration of the use of transfer learning with MobileNetV2 for classifying images in agriculture. The model reached 84.4% accuracy in identifying the most common cattle diseases, such as Bovine Respiratory Disease, Lumpy Skin Disease, Contagious Ecthyma, and Dermatitis. The system's capacity to include confidence values with each prediction improves the interpretability of the AI based decision making process, allowing users to assess the trustability of the suggestions of diagnostics. Also, the addition of extensive disease narratives, several primary treatment options, and prevention protocols makes it a one stop shop education base for farmers who

can be better informed to take decisions on livestock health. In addition to having strong technical merit, the project serves as a breakthrough in making veterinary knowledge available to smallholder farmers in rural areas. The system also tries to tackle major problems in veterinary care availability in developing countries by minimizing the need for physical veterinary visits for early diagnoses. Economic implications are significant, since early detection of disease can prevent a large-scale epidemic and reduce mortality and production loss that farmers' income are greatly dependent on. Also important are the ecological advantages: A healthier herd bodes well for sustainable agricultural practices, and reduces antibiotic overused through treatment targeting. The system also modernizes livestock management through its integrated dashboard, which features health overview analytics, disease frequency tracking, and weather informed recommendations. These capabilities enable proactive health management, allowing farmers to identify patterns and implement preventive strategies. By establishing a digital record keeping system, the platform enhances herd management practices and fosters data driven decision making in livestock operations. The successful implementation of this system validates the potential for AI driven solutions to transform traditional agricultural practices while remaining accessible and practical for end users with varying levels of technological literacy.

7.2 Limitations

The Smart Livestock Disease Prediction System can successfully achieve its main target, though some limitations should be taken into consideration. At present, the AI model has been trained only on a limited set of common cattle diseases, therefore it is possible that newly emerging pathogens, rare diseases, or species specific variations are not included. This restriction limits the system's applicability in regions where the diseases are different or in cases of previously unseen symptoms which are not included in the training samples. In addition, using the two class (informative vs. non informative) classification scheme could add a layer of over simplification to a more complex health state (multiple correlated symptoms) that may suggest a further clinical examination. The performance of the system depends significantly on the quality of the images which might be influenced by some environmental factors. Poor lighting conditions, low resolution of the camera, motion blur, inappropriate angles and background clutter may lead to decreasing the prediction accuracy. It is possible that the model may have difficulty with images in which characteristic symptoms are not prominently displayed or which feature more than one animal at a time. Besides, at the moment the system considers only visual symptoms and does not include any other critical diagnostic signs behavioural changes, temperature measures, laboratory tests results that a veterinarian would normally evaluate to make a thorough diagnosis of a disease. Technical constraints bring additional difficulties. The requirement of stable internet connection raises concerns about availability of the tool

in areas lacking the sufficient network infrastructure, thus preventing farmers who stand to gain the most by this technology from accessing it. Even though the MobileNetV2 architecture was selected for efficiency, the model still requires a kind of computing power that may not be available on older or budget smartphones that many rural residents have. Moreover, it serves as a diagnostic support tool and not as a replacement of veterinary professional care, thus risk of misuse might potentially lead to an over reliance on AI suggestions in absence of medical supervision. The model's performance is also limited by data related constraints, as the quality and variety of its training dataset can affect its accuracy. Certain biases in the training data, such as more examples from certain breeds, ages, or stages of disease, may make the model less accurate on other populations. The system does not have facilities for continuous learning, based on images submitted by users, so that it cannot adjust to emerging diseases or regional variations. Privacy challenges with respect to image data collection and storage are also nontrivial, especially in terms of complying with data protection regulations and establishing user trust when dealing with sensitive data, such as those related to agriculture.

The interface is made to be simple, but it may not sufficiently mitigate literacy issues or serve users with vision disabilities. The system presupposes a baseline of smartphone literacy that is likely not present in all members of the older farming community. And the business model to keep the system running in the long term (with fees for server maintenance, model updates, and technical support), the scalability of the project, and its financial viability for more than the research phase have yet to be determined, which adds to the questions about the project.

7.3 Future Work

The potential avenues of the future work to improve the system robustness, applicability and impacts are obvious. The extension of the disease database is a priority for improvement, with the inclusion of additional livestock species (goats, sheep, poultry), wider disease coverage and region specific pathogens. Multimodal data integration based approach could greatly enhance the accuracy of diagnoses by processing image analyses together with other health status markers, such as body temperature, feeding behavior, movement, breathing sound on audio or respiratory sound analyses. A symptom checklist based on systematic observation might be used to add more context into the image based diagnosis. Technical improvements are needed to enhance accessibility and efficiency. On-device processingEdge processing Some parts of the processing, especially the AI based processing, can be run on the device using TensorFlow Lite or a similar framework, allowing for offline operation and removing the dependency on Internet connectivity, which currently makes it difficult to use in rural areas. Improving PWAs capabilities would enhance accessibility even more for different device types and network environments.

Quantization, pruning and knowledge distillation are examples of model compression techniques that can reduce computational demands without compromising accuracy, thus enabling the system to run on a larger variety of mobile devices. There are a number of novel features by which the user experience may be enriched. Integration with IoT (Internet of Things) devices like wearable sensors for animals could allow for real time health monitoring and alert systems, animal identification via RFIDs, and secure data sharing. Enhancing the system with predictive analytics would allow for prediction of disease outbreaks based on environmental conditions, seasonal trends, and historical data.

Data management and system intelligence offer additional avenues for improvement. Developing a federated learning system would facilitate ongoing model refinement on decentralized data without compromising privacy. Negotiating data sharing agreements with veterinary and animal agriculture research organizations could augment the training data set and help inform surveillance of disease. Tailoring the recommendation for specific farm-related factors including herd size, breed composition, geographic location and management styles could make the system more useful in practical. From an environmental and economic stance, the future work is to find sustainable business models to support the system for the long run or to expand system service. This could take the form of different subscription tiers for use by commercial farms, partnerships with governments for public health initiatives, or incorporation into agricultural insurance products. Collaborations with veterinary pharmaceutical companies might facilitate treatment recommendations that were more targeted, and potentially could generate income streams for the viability of the system. Multilingual support and culturally specific interfaces would enable higher adoption rates in varying agricultural landscapes with different languages and farming methods. Further research shall study the use of more advanced AI techniques such as few shot learning for detection of rare diseases, attention mechanisms for improved symptom localization and ensemble techniques that combine several model architectures. Long term studies assessing the impact of the system on the health of the livestock, the producer's decision making, and economic returns would offer important justification for larger scale implementation. Such comparative studies with conventional diagnostics will help measure how much AI assisted diagnostics in agriculture really adds.

The architecture of the system can be extended to provide full farm management functionalities like vaccination planning, breeding records, feed management and production monitoring. Integration with weather prediction services could allow climate driven health recommendations, and connection to agricultural market platforms may support farmers in making financial decisions informed by herd health status. Creating an API for third party integration would enable the system to serve as one component within a broader agricultural technology suite rather than as a stand alone application.

7.4 Summary

Overall, the Smart Livestock Disease Prediction System provides a smart solution for accurately predicting livestock diseases based on sound AI methodologies, bringing the real feasibility of AI application to the agricultural domain. With an easy to use interface and ability to perform fast, cheap initial disease diagnostics, the service fills a very crucial gap in veterinary services access, while contributing to proactive livestock health management. The combination of deep learning models and mobile technology to successfully provide a scalable solution for a number of agricultural applications and livestock breeds. The impact of the activities going well beyond solving immediate issues to including enabling modernisation of agriculture, raising farmer awareness and helping sustainable livestock management. By converting health records to digital format and delivering data driven insights, the system promotes improved decision making and evidence based farming. The role based design enables functionality separation between the roles involved, and the focus on coin design allows sophisticated technology to be delivered to Citizens with limited levels of technical expertise. In the future, numerous enhancements and extensions can be built off the platform to further expand the impact and address other challenges of livestock health management. The trade offs revealed during development emphasize areas for targeted refinement and the need to consider how best to balance technology advancement with the practical considerations of accessibility, accuracy, and user trust. As agricultural systems worldwide are under increasing strain due to climate change, population growth, and disease emergence, technology-based solutions such as the Smart Livestock Disease Prediction System will become more critical for maintaining food security, farmer livelihoods, and sustainable development. The project proves that intelligent deployment of artificial intelligence in farming can generate tangible value without losing touch with the real world of farmers. By evolving, engaging with the agricultural community, and responding to user feedback, systems like these have the potential to radically change the way we care for livestock and contribute to more resilient agricultural systems worldwide. This work successfully accomplishes our goal to demonstrate that mobile enabled AI applications can meaningfully assist in managing livestock diseases in the developing world. It provide a technical and conceptual foundation that can be modified through iterative research and development. Most importantly, it shows how we can take a technology and use that to tackle urgent real world issues while making it accessible, practical, and useful to the communities it hopes to aid.

References

[1] C. Okello, J. Smith, and A. Muwonge. Artificial intelligence techniques for livestock disease prediction: A review. In *International Conference on Artificial Intelligence in Agriculture*, pages 42–53. Springer, 2019. Cited on pp. 1, 2, 5, 7, and 12.

[2] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. Cited on pp. 2, 5, 9, and 13.

[3] Brian A. Jones, Delia Grace, Richard Kock, Silvia Alonso, and Jonathan Rushton. Economic impact of livestock diseases in developing countries: A systematic review. *Preventive Veterinary Medicine*, 181:105026, 2020. Cited on p. 5.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, volume 25, 2012. Cited on p. 5.

[5] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010. Cited on p. 5.

[6] Konstantinos G. Liakos, Patrizia Busato, Dimitrios Moshou, Simon Pearson, and Dionysis Bochtis. Machine learning in agriculture: A review. *Sensors*, 18(8):2674, 2018. Cited on p. 5.

[7] Sjaak Wolfert, Lan Ge, Cor Verdouw, and Marc-Jeroen Bogaardt. Big data in smart farming—a review. *Agricultural Systems*, 153:69–80, 2017. Cited on p. 6.

[8] Sharada P. Mohanty, David P. Hughes, and Marcel Salathé. Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, 7:1419, 2016. Cited on p. 6.

[9] Konstantinos P. Ferentinos. Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture*, 145:311–318, 2018. Cited on p. 6.

[10] Daniel Berckmans. General introduction to precision livestock farming. *Animal Frontiers*, 7(1):6–11, 2017. Cited on p. 6.

[11] Anjani Kumar, Hiroyuki Takeshima, Ganesh Thapa, Niraj Adhikari, Sunil Saroj, and Manoj Karkee. Agricultural technology adoption in south asia: A systematic review. *Global Food Security*, 29:100537, 2021. Cited on p. 6.

[12] Raj Patel, James Smith, and Taylor Brown. Mobile veterinary services in developing countries: A review. *Tropical Animal Health and Production*, 51(6):1259–1266, 2019. Cited on p. 6.

[13] Tanja Groher, Ignas Heitkönig, Anna-Maija Heikkilä, and Julio Céspedes. Digital tools for animal health service delivery: The case of vetafrica. *Frontiers in Veterinary Science*, 7:596, 2020. Cited on p. 7.

[14] Priya Mehta, Rohit Sharma, and Alok Gupta. Telemedicine applications in livestock health management. *Journal of Veterinary Science*, 22(2):e21, 2021. Cited on p. 7.

[15] Sajjad Ahmed, Muhammad Iqbal, and Faiz-ul Hassan. Livestock management practices in pakistan: Current status and future prospects. *Pakistan Journal of Agricultural Sciences*, 57(4):937–948, 2020. Cited on p. 7.

[16] Muhammad Azam Khan, Iftikhar Ali, and Imran Ashraf. Digital agriculture transformation in pakistan: Challenges and opportunities. *Information Processing in Agriculture*, 8(1):78–90, 2021. Cited on p. 7.

[17] Asif Ali, Dil Bahadur Rahut, and Bhagirath Behera. Technology adoption among rural farmers in pakistan: A case study. *Technology in Society*, 68:101897, 2022. Cited on p. 7.

[18] Balwinder Singh, Rajesh Sharma, and JPS Gill. Traditional methods of animal disease diagnosis: Limitations and alternatives. *Indian Journal of Animal Sciences*, 89(1):1–6, 2019. Cited on p. 7.

[19] Li Chen, Hao Wang, and Ming Zhang. Human error in veterinary diagnosis: A systematic review. *Veterinary Record*, 187(8):e50, 2020. Cited on p. 7.

[20] Xiaolong Wang, Yu Zhang, and Wei Li. Early detection of livestock diseases: Challenges and technological solutions. *Animals*, 11(9):2712, 2021. Cited on p. 7.

[21] James Miller, Sarah Thompson, and Robert Wilson. Access to veterinary services in remote areas: A global review. *Preventive Veterinary Medicine*, 205:105678, 2022. Cited on p. 7.

[22] Rebecca Thompson, Mark Davis, and Paul Johnson. Economic impact of delayed disease diagnosis in livestock. *Journal of Agricultural Economics*, 74(2):389–405, 2023. Cited on p. 7.

[23] Muhammad Sharif, Ali Khan, and Zafar Iqbal. Technical challenges in deploying ai solutions in agriculture: A systematic review. *Computers and Electronics in Agriculture*, 218:108712, 2024. Cited on p. 7.

[24] Ravi Patel, Sanjay Kumar, and Anita Sharma. Research gaps in digital livestock farming: A systematic mapping study. *Livestock Science*, 255:104801, 2022. Cited on p. 8.

[25] Farhan Khan, Saeed Ahmed, and Arif Malik. Need for localized ai solutions in south asian agriculture. *Journal of Rural Studies*, 89:98–107, 2022. Cited on p. 8.