Plant Disease Detection Using Mobile Application

Final Year Project Report

Submitted by

Syed Hasnain Imran
(BSAI/ IT-21-337)
1653-2021
Khadija Rashid Mughal
(BSSE/ IT-21-233)
1814-2021
Zainab Jamil
(BSAI/ IT-21-347)
2179-2021

Supervisor

Sir Waqas Pasha

In partial fulfillment of the requirements for the degree of
Bachelor of Science in Artificial Intelligence/Software Engineering

2025

Faculty of Engineering Sciences and Technology

Hamdard Institute of Engineering and Technology

Hamdard University, Main Campus, Karachi, Pakistan

Certificate of Approval

Faculty of Engineering Sciences and Technology

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

This project "**Plant Disease Detection Using Mobile Application**" is presented by **"Syed Hasnain Imran", "Khadija Rashid Mughal", and "Zainab Jamil"** under the supervision of their project advisor and approved by the project examination committee, and acknowledged by the Hamdard Institute of Engineering and Technology, in the fulfillment of the requirements for the Bachelor degree of Artificial Intelligence/ Software Engineering.

Mr. Waqas Pasha
(Project Supervisor)

In-charge FYP- Committee

Dr. Umer Frooq
(Chairman Departing Of Computing)

Eng. Prof. Dr.Aamer Saleem
(Dean, Fest)

Authors' Declaration

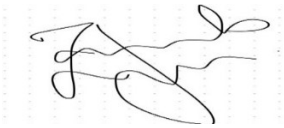
We declare that this project report was carried out in accordance with the rules and regulations of Hamdard University. The work is original except where indicated by special references in the text and no part of the report has been submitted for any other degree. The report has not been presented to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

Dated:

Authors Signatures:

Syed Hasnain Imran

Khadija Rashid Mughal



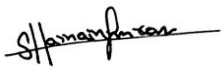
Zainab Jamil

Plagiarism Undertaking

We, **Syed Hasnain Imran**, **Khadija Rashid Mughal**, and **Zainab Jamil**, solemnly declare that the work presented in the Final Year Project Report titled **Plant Disease Detection Using Mobile Application** has been carried out solely by ourselves with no significant help from any other person except few of those which are duly acknowledged. We confirm that no portion of our report has been plagiarized and any material used in the report from other sources is properly referenced.

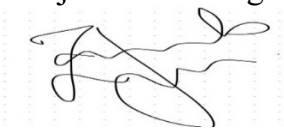
Dated:

Authors Signatures:



Syed Hasnain Imran

Khadija Rashid Mughal



Zainab Jamil

Acknowledgments

Team members would like to acknowledge **Hamdard Institute of Engineering and Technology** for support of this Project, a highly appreciated achievement for us in the undergraduate level. It is obliged to our Supervisor **Mr. Waqas Pasha** who supported as our major advisor. It would like to express our gratitude for their keen guidance, sincere help and friendly manner which motivated us to do well in the project and makes it a reality. Many people, especially our classmates and team members themselves, have made valuable comment suggestions on this proposal which gave us inspiration to improve our project.

In conclusion, we are humbled and honored to have that privilege of collaborating with such a humble and talented group of individuals. The success of plant disease detection using mobile application stands as testament to our collective capabilities and the remarkable achievements we can accomplish together.

Thank you all for your contributions, commitment, and unwavering support.

Document Information

Table 1: Document Information

Customer	Students/Educational Institutes
Project Title	Plant Disease Detection Using Mobile Application
Document	Final Year Project Report
Document Version	1.0
Identifier	FYP-028/FL24 Final Report
Status	Final Report
Author(s)	Syed Hasnain Imran Khadija Rashid Mughal Zainab Jamil
Approver(s)	Mr. Waqas Pasha
Issue Date	3-july-2025

Definition of Terms, Acronyms, and Abbreviations

TERM	ABBREVIATIONS
FYP	Final Year Project
CNN	Convolutional Neural Network
AI	Artificial Intelligence
ANN	Artificial Neural Networks
DL	Deep Learning
ML	Machine Learning
KNN	K Nearest Neighbors
CV	Computer Vision
SVM	Support Vector Machine
RF	Random Forest
V & V	Verification and Validation
MobileNetV2	Mobile Net Version 2
MGP	Mukul Goyal Products
UI	User Interface

Abstract

A mobile-based system utilizing deep learning algorithms for detecting plant diseases in nurseries. The system employs a convolutional neural network (CNN), trained on large dataset of plant leaf images, to accurately identify a range of diseases, including those affecting specific plant species and crops. MobileNetV2 and other state-of-the art techniques for efficient feature extraction and lightweight model architecture are integrated into the system to ensure real-time processing in mobile devices. The app's intuitive design allows farmers, nursery staff, and non- experts to detect plant disease early, enabling timely interventions and reducing crop losses. This solution is particularly focused on nurseries, where early disease detection maintaining plant health during critical growth stages. The system also compares various techniques and approaches used to classify and identify plant disease in order to guarantee consistent performance across various environmental conditions. The application which is meant for regions with limited resources, encourage sustainable farming methods by helping specific measures and lowering dependency on inappropriate chemical treatments. By managing problems with early detection and disease control, this will realty improve nursing and agriculture efficiency.

Keywords:
Plant disease detection, nursery-based system, mobile application, deep learning, convolutional neural network, MobileNetV2, real-time processing, early-stage detection

Table of Contents

Table of Contents

List of Figures

List of Tables

CHAPTER 1
INTRODUCTION

The detection of diseases is crucial for the preservation of nursery plants and elevating productivity in agriculture. Older techniques of disease diagnosis usually rely on manual processes which are slow, labor-intensive, and full of mistakes which makes them useless in the rapidly changing world of nurseries. In an attempt, to bridge this gap a mobile application that uses artificial intelligence image processing technologies has been developed enabling the identification of plant disease using pictures of leaves.

Given the limited availability of sophisticated tools and the specialized technology, this solution focuses specially on nursery workers. This system's integration of Convolutional Neural Networks (CNNs) ensures precision in classifying diseases, even in real –world scenarios. The system is made, more strong through techniques such as data augmentation and transfer learning to improve model performance and adaptability.

Mobile devices with restricted processing power can run experienced deep learning model like MobileNetV2. Therefore, executing disease detection on smartphones is quick and efficient, even in the absence of advanced equipment. Ultimately, the aim is to provide users with reliable effective, yet practical and moveable, device to monitor plant health at the nursery level.

The app supports nursery workers to respond instantly to plant disease detection through real-time analysis and easy-to-use. Further usability arises from the addition of diagnosed diseases, treatment suggestion, PDF health reports and feedback reports. This not only aid in plant maintenance but also help in smooth nursery management systems, making them more efficient and responsive.

Generally, this project takes the AI technology and puts it to agriculture and nursery need. It allows the nursery staff to detect plant disease early enough to carry out proper treatment, minimize loses, and manage the nursery endurable and efficiently.

1.1 Motivation

The question of why develop a software solution that already exists and is deployed in the industry is valid and can be addressed by highlighting the unique aspects and advantages of this project. While various plant disease detection systems are available in the market, many of them may have limitations such as high cost, complex interfaces, or limited accessibility. This project aims to address these gaps and provide a more practical, user-friendly, and accessible solution for plant disease detection.

The primary motivation behind this project is to develop a mobile-based plant disease detection system that is cost-effective, easy to use, and accessible to a wider range of users. By leveraging the power of deep learning, particularly Convolutional Neural Networks (CNNs) such as MobileNetV2, this system aims to provide accurate disease detection while overcoming challenges faced by traditional systems.

1.2 Problem Statement

Identifying plant diseases is vital for crop health, but traditional methods are time-consuming and error-prone. Resource limitations and the rise of new diseases highlight the urgent need for faster and more accurate detection. Resource limitations and the rise of new diseases highlight the urgent need for faster, more accurate detection. This emphasizes the potential of deep learning techniques, especially CNNs, in automating disease identification. Mobile apps powered by CNNs can enhance sustainable farming, minimize crop losses, and improve disease management reduction.

The increasing threat of new plant diseases is impacting the agricultural economy. Automated solutions, such as CNN-based mobile applications, provide rapid diagnostic tools for nursery workers, helping to reduce losses and strengthen the agriculture and nursing sectors.

1.3 Goals and Objectives

This project utilizes Convolutional Neural Networks (CNNs) along with optimized models like MobileNetV2 to identify and classify plant diseases from images captured on mobile devices. The compact design of MobileNetV2 ensures precise and efficient processing on mobile platforms without compromising performance. By employing this deep learning model to analyze leaf images, the app delivers quick and accurate disease detection, serving as an individual tool for nursery workers to monitor their plants.

This mobile application will feature a user-friendly interface designed for nursery staff and owners. Key functionality will include real-time disease detection, a secure logout option, and the ability to upload images for analysis. After detecting a disease, the application will show the symptoms related with the identified disease and provide helpful recommendations to manage it efficiently. The application will also offer a download disease report feature, allows users to generate and store reports for reference and record keeping.

1.4 Project Scope

Our objective is to develop a deep learning-powered mobile plant disease detection system to help nursery workers identify diseases from leaf images. The system plant health management offers an intuitive interface for instant diagnosis.

- To accurately detect plant diseases using a CNN-based deep learning model.
- To enable early detection of diseases, reducing their spread and impact on nursery plants.
- To design a simple user-friendly mobile application that can be easily used by nursery personnel with minimal technical expertise.
- To optimize the system for quick, highly accurate real-time disease detection.
- To support sustainable nursery management by reducing labor costs and enabling timely plant health interventions.

CHAPTER 2
RELEVANT BACKGROUND & DEFINITIONS

2.1 Introduction

Plant diseases significantly impact crop production and global food availability. Detecting these diseases at an early stage helps avoid severe damage and financial losses for farmers. With rising food demand and limited agricultural land and resources, the farming industry is adopting automated approaches for disease identification and management. With the help ML and DL approaches improves the performance and speed of plant disease detection and classification, emphasizing the effectiveness of convolutional neural network (CNN). These technologies streamline the diagnosis process and reduce reliance on agriculture specialist. [1].

Moreover, artificial intelligence enables the development of intuitive tools that require minimal technical knowledge from users. For instance the integration of image-based disease recognition into mobile apps for disease detection of plants. Such systems can not only detect disease but also provide treatment. As a result, it can be used to help people for tracking their house plants and also enables the farmers to keep a track of the harvest. [2].

2.2 Historical Background

In the initial stages of agricultural technology, researchers working on plant disease detection primarily relied on traditional image processing methods. The systems that were developed evaluated the plant's appearance, based on color segmentation, texture analysis, and feature extraction. These classical image analysis techniques were limited in terms of generalization across species growing conditions, as earlier work discussing the symptoms we detect the tree leaves and fruits disease using image processing techniques, using images to extract the information from the images. Image processing basically includes the some steps: 1.Importing the image through image acquisition tools Analyzing and manipulating the image 3. The output result can be altered image or that is based on image analysis. [3].

The shift from traditional methods to intelligent systems marked a significant milestone in plant health monitoring. This was driven from advancements in deep learning architectures, which enables the system to automatically learn relevant features without relying on manual input. Furthermore, the introduction of

lightweight models for mobile device like user interface is developed a mobile app, allowing farmers to capture a photo of the infected plant leaves. It then displays the disease category along with the confidence percentage, demonstrated the deep learning could be efficiently deployed on smartphones to support real-time disease diagnosis. [4].

2.3 Current Situation

Modern plant disease detection system heavily depend on deep learning models that are trained on large-scale image dataset. A widely used example is the PlantVillage dataset, which contains labeled leaf images of crop like tomato, potato, and pepper. This dataset has been utilized by numerous researchers to develop scalable detection frameworks that are adaptable to multiple plant or crop varieties.

In practical applications, these deep learning models are integrated into mobile apps to deliver quick results. Lightweight CNN architectures like MobileNet enables users to capture and examine leaf images swiftly and accurately, functioning efficiently on devices with limited computing resources in constrained environment. In certain implementations, Convolutional Neural Network is used to with a goal to detect the diseases in the crops. The model is basically tested on some types of plant species with some types of plant diseases. The overall system results show that the Mobile Net model works better as compared to the other models and provide better accuracy in detecting the diseases. [5].

2.4 Key terms and definitions

In the context of plant disease diagnosis, the machine learning (ML) is a data-centric method that enables system to learn patterns directly from sample data. These samples, typically annotated images, are used to train models to distinguish between healthy and diseased plants. In this regard machine learning and image classifier are used to identify healthy plant from infected ones, making ML a core component of many modern agriculture tools.

Deep learning (DL) is a branch of machine learning that utilizes multiple layers of neural networks to extract, high level feature from data. DL methods such as Convolutional Neural Networks (CNNs) are widely recognized for their effectiveness in handling visual recognition tasks. One study highlighted that it offers a comprehensive explanation of DL models used to visualize various plant diseases, using visualization techniques to detect and classify the symptoms of plant diseases. [6]. Moreover, it frequently mentioned that CNNs are commonly adopted

due their inherent adaptability particularly in scenarios requiring analysis of spatial patters and textures.

Before applying deep learning models, various image processing techniques are typically implemented to enhance image quality. These steps help improve contrast, minimize visual noise, and isolate infected regions. It is emphasized the importance of image processing techniques to accurately identify and classify disease from leaf images as a vital stage in the workflow. MobileNetV2 is one model of interest when it concerns mobile deployment, the convolutional neural network which has been optimized for mobile and embedded system. Mobile Net model works better and provide better accuracy in detecting the diseases.

2.5 CNN and MobileNetV2 Model

Convolutional Neural Networks (CNNs) are among the most widely applied models for identifying plant disease. The way CNNs are arranged in layers mimics the way human eye recognizes shapes, and patterns with in the image. Each layer extracts increasingly complex information to classify the plant disease, using a Convolution Neural Network (CNN), which is made up of several layers that are used to forecast diseases. We can accurately diagnose and distinguish between different plant diseases using image processing, and has proven effective for automated diagnosis across various crop types. [7].

To increase the use of these models in the community, MobileNetV2 was introduced. It is a lightweight convolutional neural network designed to reduce computational requirements while maintaining strong classification accuracy. The overall system results show that the Mobile Net model works better as compared to the other models and provide better accuracy in detecting the diseases, making it well-suited for mobile and low-resource setting where fast results are essential.

Mobile-Based Detection highlighted the importance of mobile applications in managing agricultural disease. They introduced a mobile device-based system that utilizes deep learning algorithms to detect plant diseases in real-time. By swiftly identifying diseases, this system allows farmers to take immediate action, reducing crop losses. The central focus of their research was the need for efficient models that perform well on mobile devices with limited resources, such as low memory and processing capacity. This technology is now more accessible to farmers, even in areas with limited infrastructure.

2.6 Importance of the Topic

The ability to detect plant disease before they severely impact plant and crop productivity is crucial. Timely identification helps prevent yield reduction and limit the excessive use of pesticide. Studies indicate that accurate recognition and classification of plant diseases is highly important and this can be achieved through image processing, highlighting the effectiveness of automated techniques over manual inspection for reliable disease detection.

Moreover, these AI-powered systems no only detect plant diseases but also offer treatment suggestions and enable continuous monitoring, contributing to long-term sustainability. This functionality is especially valuable in developing regions. For that such technology are use at improving crop yield, and also monitor plant health. On a large scale, the intelligent use of this technology contributes meaningfully to sustainable growth in plants and agriculture and in other similar farming-driven economies also in nurseries.

CHAPTER 3
LITERATURE REVIEW & RELATED WORK

3.1 Literature Review

Numerous systems have been designed employing Convolutional Neural Network (CNNs) to identify plant diseases through analysis of leaf, and fruit based images. The overall system results show that the Mobile Net model works better as compared to the other models and provide better accuracy in detecting the diseases. [5]. Mobile application enables users to capture images of infected plant disease and instantly displays the identified disease category along with confidence score indicating the likelihood of infection. The user interface is developed as an Android mobile app, allowing farmers to capture a photo of the infected plant leaves. It then displays the disease category along with the confidence percentage. [4].

To enhance early detection, various methods have developed useable solutions that combine Convolutional Neural Network (CNNs) with machine learning workflows to extract features and classify multiple plant disease, Once model proposed a computationally efficient and cost-effective system for 20 different diseases of 5 common plants using computer vision and machine learning techniques, making them effective for real-world use in environments with limited computational power proposed detect 20 different diseases of 5 common plants with accuracy. [8]. These systems have also been evaluated for their capacity to directly support farmers by merging functionality with actionable insights in the output, including treatment suggestions such as suggest appropriate insecticides to farmers for effective disease management and contribute to the sustainable development of India's agricultural sector. [9].

CNNs are commonly trained using standard dataset such as PlantVillage which contain annotated images of various crop diseases. Using images that were obtained from (Plant Village dataset) website, include tomatoes, pepper, and potatoes. [10]. since the objective of researchers is to construct models based, on standardized methodologies, consistency is crucial, which can help develop actionable guidelines related to training CNNs. Then, using these principles as a guide, we developed our own set of models to recognize diseases in strawberry images, comparing the accuracy of various architectures. [11]. Furthermore, it enables to trust the reproducibility of results and more efficiently evaluated model performance across different architectures and datasets.

In addition to classification, some systems are being enhanced with features that improve usability and overall performance. This present work can contribute to agricultural domain and can be used to help people for tracking their house plants and also enables the farmers to

keep a track of the harvest. This work can be expanded into further to develop an app through which one would also know the remedy to a plant disease. [2]. Others focus on usability, achieves an overall classification accuracy for disease categories in crop species, *and* provides a user interface as an mobile app , allowing farmers to capture a photo of the infected plant leaves. [4].

Despite technological advancements, several limitations remain in existing systems. A major limitation is the lack of capability to track disease progression or severity over time. The severity of plant diseases changes with the passage of time, therefore, DL models should be improved/modified to enable them to detect and classify diseases during their complete cycle of occurrence. [6]. Furthermore, the deploying trained models across various crops types continues to be a challenge. The performance of state-of-the-art techniques are analyzed to identify those that seem to work well across several crops or crop categories. Due to resource limitations, mobile-based applications especially those relying on video data may face restrictions in rural or low-resource environments. [12]. lastly, the absence of consistent evaluation criteria makes it challenging to accurately assess how well model perform. This lack standardization also complicates the process of comparing different implementations, which can hinder progress and development of more effective and reliable system.

3.2 Related Work

3.2.1 PlantNet Plant Identification:

PlantNet is a mobile application that assists users in recognizing various plants. The app uses image recognizing technology to identify submitted and suggest possible plant names. The app encourages user to provide multiple image as possible to improve the chances of accurate identification. PlantNet mainly serves the purpose of plant recognition.

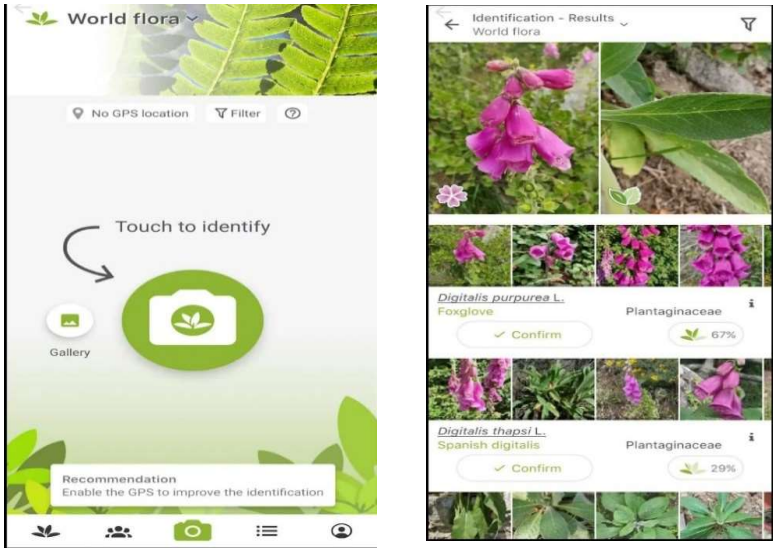


Figure 3-1 PlantNet Identification Interface

3.2.2 Plant Identifier & Care – Greg:

Plant Identifier & Care – Greg is an application that allow users to recognize plants by capturing an image of leaf or other plant parts. The app utilizes the AI to determine the plant species and offer a personalized care schedule. This application is created to simplify plant care, especially for new and inexperienced one. It is an easy-to-use application perfect for taking care of plants at home.

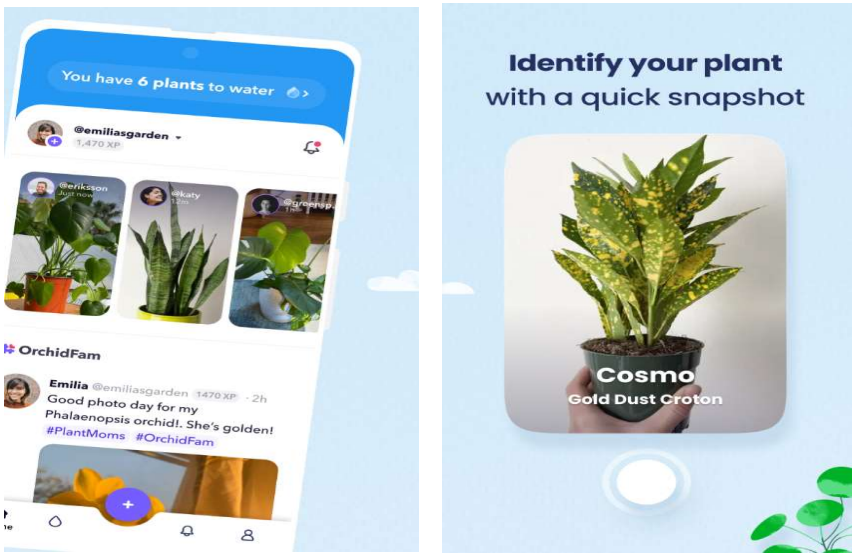


Figure 3-2 Plant Identifier & care- Greg Interface

3.2.3 Plant Disease Detector:

Plant Disease Detector is a smartphone application created by MGP(The Mukul Goyal Products) that assists users in recognizing diseases that affect plants by analyzing images of their leaves. The app features an easy-to-use interface where individuals can either capture a photo or upload one, allowing the system to predict the type of disease affecting the plant.

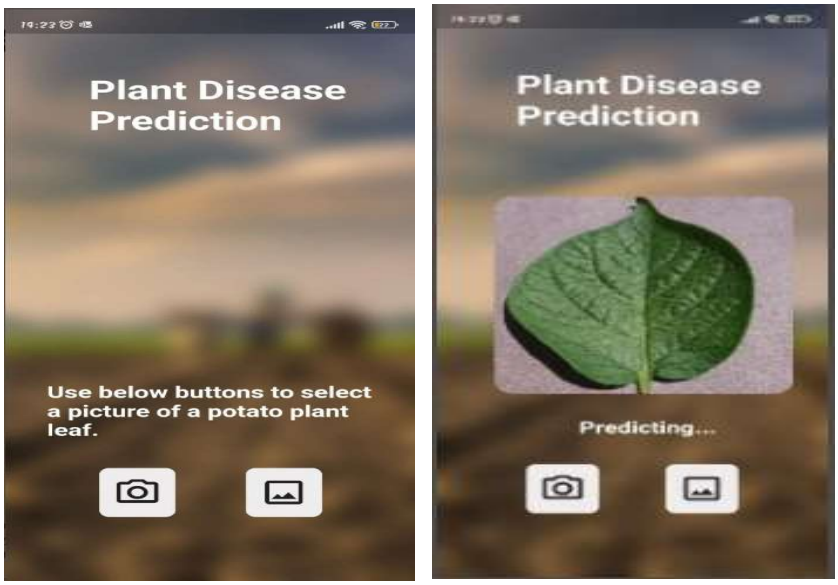


Figure 3-3 Plant Disease Detector Interface

3.2.4 LeafCare – Plant Care:

Leaf Care – Plant Health Care is a mobile app designed to support the recognition of various plant spices through AI-driven image analysis. Users can either capture or upload a photo of a leaf or plant part to receive immediate identification, including plant common names. The app allow users to store their identified plants in a personal collection. Although LeafCare provides reliable identification and informative plant data, but it's still lack in additional features.

Figure 3-4 LeafCare-Plant Health Care Interface

3.3 Gap Analysis

Our application shares some common features with other plant-related apps, such as disease detection. However it provides a unique range of tools specially developed for nursery workers. Alongside utilizing deep learning and image analysis to recognize plant diseases, the app also displays the visible symptoms of the plant and recommends appropriate treatment methods. It generates a comprehensive PDF health report available for download. We include a simple feedback area to secure input from users for future improvement opportunities. The app is designed for user simplicity and mobile user-friendliness. It is also appropriate for a non-technical user and provide a focused plant health evaluation.

No.	Related Applications	Disease Detection	Symptoms	Recommendations	Generate PDF	Feedback
1	PlantNet Plant Identification	✗	✗	✗	✗	✗
2	Plant Identifier & Care	✗	✗	✓	✗	✗
3	Plant Disease Detector	✓	✗	✗	✗	✗
4	LeafCare - Plant Health	✗	✗	✗	✗	✗
5	Plant Care	✓	✓	✓	✓	✓

Table 3-1 Comparison Table

CHAPTER 4
PROJECT DISCUSSION

4.1 Software Engineering Methodology

The primary purpose of a methodology is to establish a structured framework that outlines the necessary steps, activities, and tasks for achieving a specific objective. It serves as a valuable tool for both teams and individuals by providing clear guidance, defining roles and responsibilities, and fostering a systematic approach. Methodologies can be broadly applicable to different projects or tailored to suit specific industries or domains. Well-defined methodologies bring several advantages. They encourage the adoption of standardized practices, enabling teams to work cohesively and efficiently, even with multiple team members. By providing a structured framework, methodologies ensure that essential steps are not overlooked or skipped, thus minimizing the risks of errors or omissions. Moreover, methodologies enhance project planning and control by offering a roadmap that outlines project milestones, deliverables, and timelines. Furthermore, methodologies often incorporate best practices and lessons learned, facilitating continuous improvement and process optimization. In summary, methodologies provide a structured path and systematic approach to accomplishing project objectives. They help teams work together effectively, reduce risks, and deliver high-quality outcomes. By following a methodology, organizations can establish consistency, enhance collaboration, and increase the likelihood of project success. The simplest definition of methodology is the study of research techniques. But the term can also refer to the methods themselves, or to the philosophical examination of the underlying assumptions. A technique is deliberate approach used to accomplish a certain goal. In the development of this project developers follow the V-model approach to the end of the project.

4.2 Project Methodology

The project followed a phased approach where each step was carefully planned, developed and tested. Initial search helped identify the real world need for plant disease detection. The system was designed using modular architecture to support features like disease detection, symptoms explain and recommendation generate. Image processing and machine learning technique were used to detect disease, while a mobile friendly interface was designed for ease of use. Testing was integrated for each phase following the v-model structure. V-Model demonstrates the relationships between each phase of the development life cycle and its associated phase of testing.

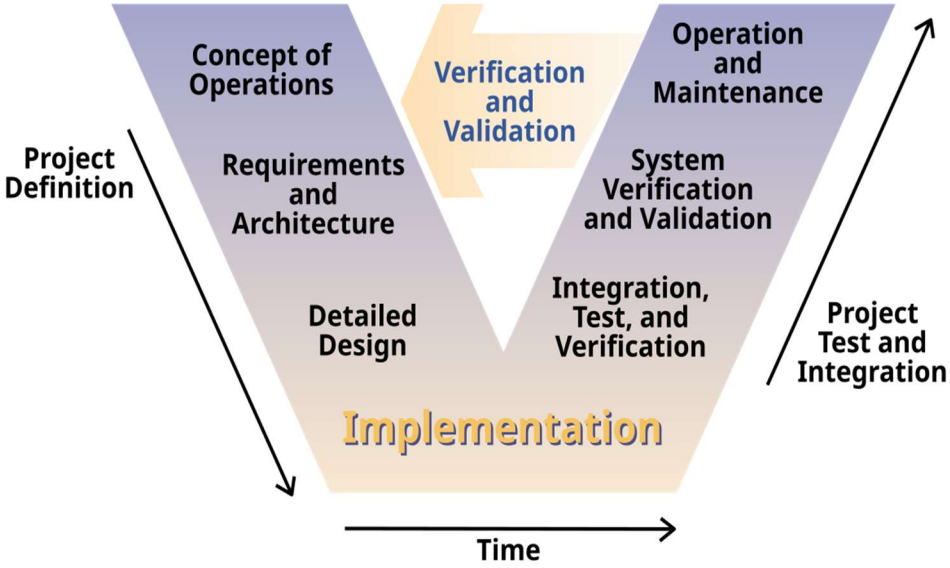


Figure 4-5 V-Model Methdology

Figure 4-5 tells us the fundamental principle of the V-Model underscores the concurrent advancement of development and testing, emphasizing that testing is not just a final stage but an integral element integrated throughout the entire development process. This methodology improves the early identification of problems and leads to cost savings. It is especially suitable for critical applications, fostering assurance in the software's excellence and its alignment with user requirements.

4.3 Phases of the Project

The v model (verification and validation model) was implemented in the development of this project plant disease detection to ensure a structured and test driven approach. This methodology support early detection of errors by aligning every development phase with cross ponding testing phase. The v-model is ideal for this academic approach project, where requirements and deliverables were well-defined from the beginning.

In the V Model, the left side of the "V" represents the requirements gathering and analysis phase. In the V-Model, the right side of the "V" represents the testing and validation phases. These phases correspond directly to the development phases on the left side. As each development activity is completed, a matching testing activity is performed to verify and validate that the system is functioning as expected.

1. System Design: Moving down to the left side of the 'V', this phase involved converting the gather requirements into a complete system design for mobile application. This included designing the system interface, defining the architecture for the disease detection module. The design phase ensures that the system is well-planned and aligns with the identified requirements.
2. Component implementation and unit testing: At the bottom of the 'V', this phase focused on implementing individual system component image upload feature, machine based learning based disease detection, symptoms display screen and recommendation module. Unit testing was conducted at this stage to ensure each module performed its specific function accurately and independently, such as checking whether an image could be uploaded and whether the detection result was displayed correctly.
3. Integration and system testing: After individual modules were implemented and tested they were integrated to form a complete mobile application. This phase involved the testing the interaction between components. Integration testing verified that all parts worked together seamlessly, while system testing ensured the entire application met performance expectations, provided accurate results, and maintained stability across devices.
4. Acceptance Testing and Validation: Ascending the right side of the 'V', this phase validated whether the application meet the actual user expectations. User acceptance testing (UAT) is performed to ensure that the system aligns with the business needs, is user-friendly, and satisfies the expected functionalities. This

phase involves real-world scenarios and user feedback to validate the system's effectiveness and usability.

5. Deployment and Maintenance: At the top of the 'V', the application was prepared for the deployment. This include installing the app on mobile devices. The maintenance system will focus on fixing bugs, ensuring security and updating the model for improved accuracy.

4.4 Software/Tools that Used in Project

1. Flutter: Flutter is an open source UI toolkit developed by google for building natively compiled applications for mobile, web, and desktop from a single codebase.
2. Firebase: Firebase is a backend as a service (Baas) platform by google that provides real-time database services, authentication and cloud service. It allows to store a list of objects. Google Firebase is Google-backed application development software which allows developers to develop applications for Android, iOS, and Web apps.
3. Python: Python is weirdly used programming in machine learning. Tensorflow and Keras are libraries that simplify the creation of deep learning model. They were used to train and implement the plant disease detection model based on image inputs.
4. Google Colab: Google colab and Jupiter notebook are online and local environments used for coding in python especially for data science and ML. These platform were used to experiment, train, and test the machine learning model.
5. Android Studio: Android Studio are integrated development environment (IDEs) used for writing debugging application code.
6. GitHub: GitHub is a version control and collaboration platform that allows teams to manage code changes efficiently. It was used to store the project code securely, track revisions, and collaborate during development stages.
7. Microsoft Word: Microsoft word is used for documentation purpose such as preparing the final thesis report, recording test cases planning project timelines and managing the project written content.
8. Chrome: Google officially released Chrome, a free Internet application, on December 11, 2008. Its components include automated website interpretation

and spell checking, selective browsing, and synchronization with Google administrations and records.

Figure 4-6 Software Tools

4.5 Hardware that Used in the Project

1. Laptops: For Training machine learning models and app development.
2. Smartphone: For testing and running the mobile application.
3. Internet connection: For downloading dataset and tools. A 3G or 4G mobile internet device.

Chapter 5
IMPLEMENTATION

5.1 Proposed System Architecture/Design

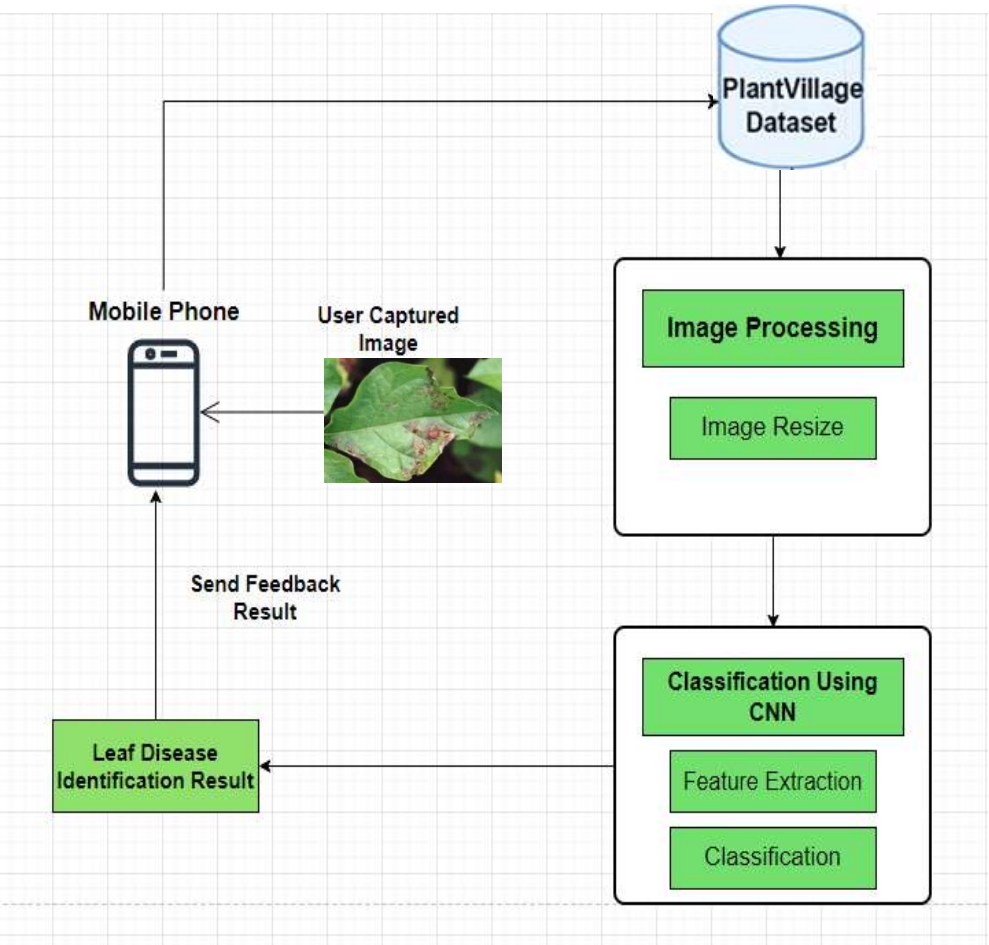


Figure 5-7 System architecture

Figure 5-7 the system architecture of the Plant Disease Detection Mobile Application is designed with a modular structure to ensure smooth functionality. The user interface begins with a homepage, offering options to log in or sign up. After authentication, users access the main page, where they can either capture plant images using the mobile camera or upload images from the gallery. These images are analyzed by a disease detection engine, which utilizes preloaded datasets and algorithms to identify plant diseases, such as bacterial spots. The analysis results, including the plant's health status and detected disease, are displayed to the user. The architecture also incorporates a database layer to manage user details, uploaded images, and detection results. Furthermore, the app includes features for secure authentication and logout, ensuring reliability, security, and effective disease detection for nursery workers.

5.2 Flowchart Diagram

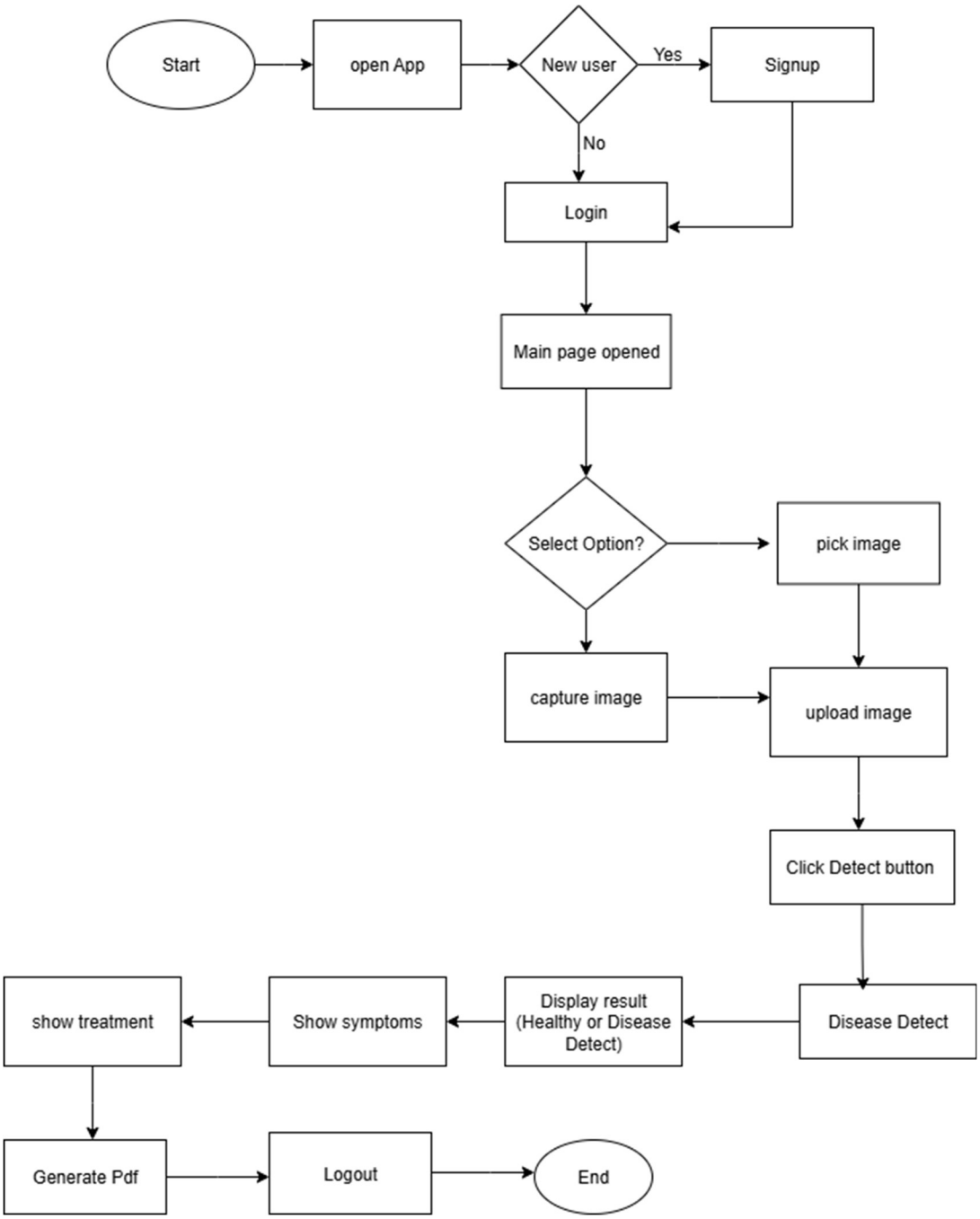


Figure 5- 8 Flowchart

Figure 5-8 the flowchart is created to provide a visual representation of the complete workflow of the Plant Disease Detection Mobile Application. It breakdowns down the users interaction with the system from starting to end in a sequence. By mapping this process, the flowchart helps stakeholders including developers, testers, and supervisors understand the apps functionality at a look. It make sure clarity in design, supports well planning, and acts as a standard guideline for development and documentation. It also helps identify pronouncement and user actions, making it easier to spot.

5.3 Sequence Diagram

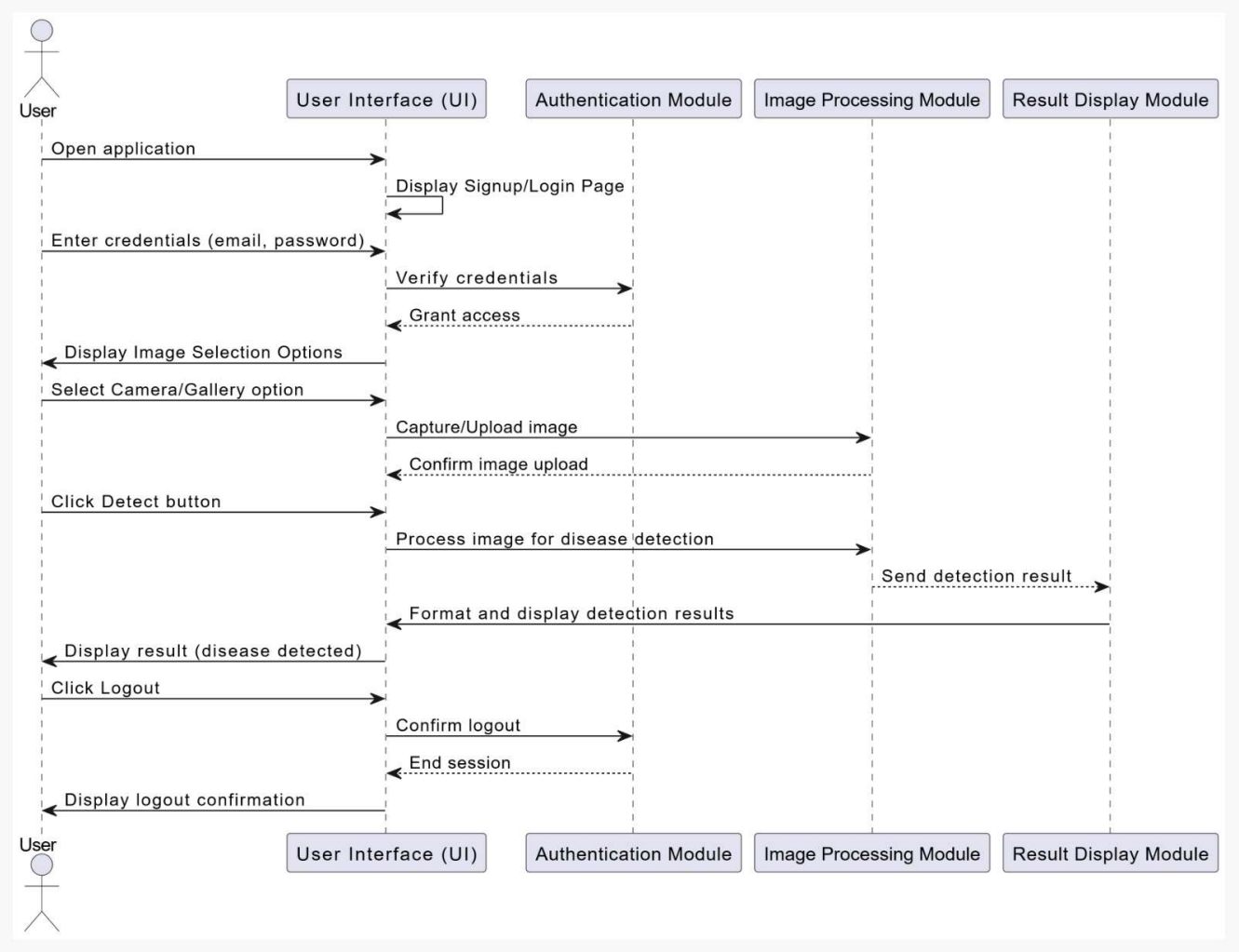


Figure 5-9 Sequence Diagram

Figure 5-9 the sequence diagram is used to show the flow of communication between different system components overtime. It displays how each module (User Interface, Authentication, Image Processing, and Result Display) interrelates as the user performs actions like logging in, uploading an image, and receiving results.

5.4 Use Case Diagram

Figure 5-10 Use Case Diagram

Figure 5-10 The Use Case Diagram is intended to visually characterize the interactions between the system and external actors (User and Server). It provides a high-level overview of the system's functionalities and how the Nursery Worker (User) and the Server act together with different use cases. This diagram captures the fundamental functionalities such as login, capturing and uploading images, preprocessing, disease detection, and viewing results.

5.5 State Diagram

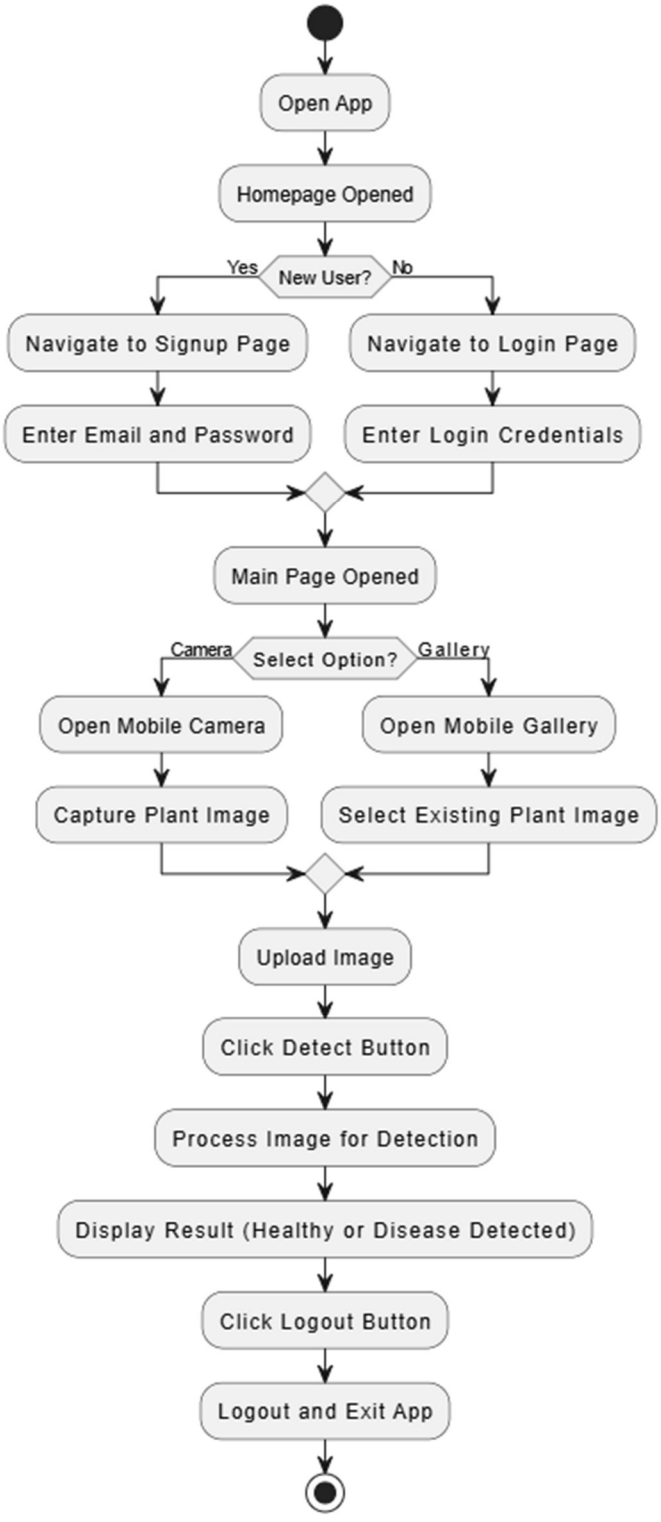


Figure 5-11 State Diagram

Figure 5-11 the state diagram provides a detailed flow of actions taken by user within the app. It visualize the step-by-step operations from the moment the app is opened until the user logs out and exits. The diagram includes decision nodes to represent alternate flows based on user input.

5.6 Functional Specifications

1 The system provide a secure user authentication system using firebase authentication. New users can create an account by registering it with a valid email and password, which is surely stored in firebase. Existing users can log in to access the application with those without the account are redirected to signup page. This make sure that only authorized users can access the application features, improving the data security.

2 To recruit the disease detection process, the system delivers users with two useful options for submitting plant leaf images. Users can either pick up a new image straight using the device integral camera or upload a present image from the device gallery. The image capture option is useful for analysis when users want to observe a plant on the spot, while the upload option allows users to select formerly saved images for testing. This flexibility ensures that users can use the application in many positions, whether they are really present in the nursery or look over stored images later. Both options are integrated into the application, making the process rapid, effective, and user-friendly.

3 Once the user upload or capture the image of leave, the application process the input by sending it to preprocessed trained input model integrated with in model. The model determine the leaf image and analyze whether the plant is healthful state or diseased. If an infection is detected it returns the predicated disease class along with the confidence level in percentage, demonstrating how confident the model is about its prediction. This helps users assess the consistency of the results.

4 Once the disease is identified the system displays a list of common symptoms associated with the detected disease. This help users understand the current condition of plant and recognize the visible sign of infection such as spot. By clearly giving these symptoms, the app assists as an educational tool for nursery workers, letting them to learn how different diseases apparent.

5 Based on the disease identified, the app offers recommendations and preventive measures. By acting on these recommendations users can reduce plant loss, maintain healthy stock and enhance overall management.

6 Users can give options to generate PDF report, which contains the uploaded image, associated symptoms and the suggested treatment. This report can be used for record keeping, future reference, or to share the diagnosis with experts or supervisors. It provide a structured summary of the disease detection process in readable format. This feature is especially useful for nurseries managing a large number of plants, as it helps in maintain

and organize documentation of plant health overtime.

5.7 Non-Functional Specifications

Non-functional requirement plays a crucial role ensuring that the plant disease detection operates efficiently and reliability. While functional requirement focus on core features like image upload, disease predication and report generation, non-functional requirement defines the quality attributes of the system such as performance, usability, security and reliability. These requirements are vital for providing a smooth user experience, guarding sensitive data, and make sure the app remains robust and scalable as user needs advance. Meeting NFRs helps build user trust, supports long-term maintenance, and underwrites to the overall success and sustainability of the application.

5.7.1 Performance

Performance is generally perceived as a time expectation. This is one of the most important considerations especially when the project is in the architecture phase. This include identifying the disease class, displaying the confidence level and loading related symptoms and recommendations. Accomplishing this level of performance requires the use of enhanced machine learning models, efficient image pre-processing, and smooth integration between the frontend and backend systems. The app should also remain responsive under varying network conditions and hardware configurations.

5.7.2 Safety

The application must handle user data in a manner that handle prevent loss or unauthorized access. This include implementing encryption, access controls, and regular backups. Ensuring data safety is essential for building user trust and adhering to data protection laws. The system must also contain proper error handling mechanisms to prevent failures or data loss during processes. In the case of system failure or bugs, the application should not cooperate any sensitive user data. Make sure of data safety not only defends users but also builds confidence and defiance with privacy regulations.

5.7.3 Security

The system must apply strong access control and session management to guarantee that only authorized users can log in and access the applications features. User verification is handled through Firebase, which manage secure login, logout, and session control, preventing unauthorized access to user data and functionality. Moreover, all user data must be stored in a secure make sure that sensitive info such as account credentials and diagnostic records are protected against data violations or damaging. Firebase database

and storage facilities support secure data management. This make sure that data transferred over the network, including images and diagnosis results, cannot be interrupted or changed, maintaining the privacy and reliability of user information.

5.7.4 Reliability

The system must ensure reliability, making the application consistently available for users. Achieving high reliability requires regular maintenance, backup systems, and robust testing to avoid unexpected outages. This is critical for users who rely on the app for timely results. It is necessary to guarantee and alert about the system commerce and processing as simple as keep a system log will increase the time and effort to get it done from the very start. Data should be conveyed in a trustworthy way and using trustful conventions.

5.7.5 Usability

The app interface must be simple and user-friendly, catering to non-technical users like nursery workers. Easy design to ensure approachability on various devices. A well-designed interface ensures the app is accessible and encourages consistent use. Since the app targets non-technical users such as nursery staff and plant caretakers, usability is serious. The interface must be instinctive, and responsive, ensuring easy navigation through features such as login, image upload, detection results, and PDF generate. The app must support multiple device sizes and alignments using easy design, ensuring convenience across a wide range. Clear icons, simple language, slight input fields, and visual clues will help decrease user misperception and improve the user experience, encouraging recurrent and confident use of the app.

5.7.6 Maintainability

The system must be designed for easy maintenance to ensure long-term functionality and adaptability. This includes writing clean, modular, and well-documented code to facilitate updates or bug fixes. Moreover, a robust error logging mechanism should be in place to quickly classify and resolve problems, decreasing interruption and maintenance efforts. Maintenance might show scalability to raise and enhance the system features and functionalities. Developers should follow standard coding practices and version control using GitHub for following changes. Any faults or bugs or failures must be charted through a central error logging system to rapidly classify the root cause. This allows developers to do well-organized debugging and system updates. Maintainability make sure that the system can adapt to upcoming requirements, such as adding new plant

diseases or improving the accuracy of the ML model, without troublemaking the existing functionality.

5.7.7 User Documentation

All projects require a least of documentation at changed levels. In many circumstances the operators might even need preparation on it, so keeping good documentation observes and values will do this task spread along the project development; but as well this must be create since the project development to include this task in the list. A detailed user manual should be created to describe the technical and functional details of the project. This manual must outline the system architecture, core features, development process, and the technologies used. It should also include system requirements, and maintenance guidelines to provide a clear understanding of the project.

5.7.8 Compatibility

This app is designed to ensure compatibility across wide range of devices, providing a consistent and smooth user experience regardless of screen size or hardware configuration. Built using flutter, the app support various Android versions and is optimized for different device resolutions and orientations. It is compatible for both older and newer smartphone using commonly in nurseries and ensuring accessibly for all users. The system also ensures compatibility with essential component such as camera, file storage and internet availability, which are critical for image upload and disease detection functionality.

5.8 Testing

Testing plays a multifaceted role in product development, extending beyond the identification of errors to refine and align products with specific needs and requirements.

1. Verification involves a comprehensive assessment of various components of the system to ensure they align with specified requirements and design documentation. It checks whether the system is being developed correctly, with the tools and methodologies. This include reviewing code quality, interface layout and backend integration, the proper functionally of the ML model used for disease detection. Verification confirms the app features such as login/signup, prediction handling and report generation are implemented according to the original technical specifications.
2. Error Detection Testing focuses on finding defects, bugs, or irregularities within the system through deliberate and complete analysis. This type of testing involves

simulating real-world practice, testing edge cases, and applying input that could possibly cause the application to fail. It aims to discover unseen problems such as broken functionality, slow responses, incorrect predictions, or crashes. By keenly looking for out bugs before the system is deployed, this testing phase helps avoid faulty performance from affecting end users. It acts as a defense that guarantees the application maintains functional integrity, accuracy, and robustness across various use cases and device conditions.

3. Validation assess whether completed system meets the actual user needs, expectations and real world application scenario. It ensures that the application just not work technically but also provide the relevant and effective solutions for the intended audience. Validation answers the question are we building the write product, this phase include user acceptance testing where real user interact with the application to confirms it performs correctly, easy to use and provide accurate disease detection, useful recommendations and accessible report.

5.9 Purpose of Testing

The purpose of testing in this project is to ensure that the plant disease detection mobile application is functionally accurate user friendly and reliable before it is delivered to the end user. Testing helps to detect defects, irregularities, or unconventionalities from requirements at many stages of development. It authorizes that the system performs as predictable under different situations, make sure that all features such as image upload, disease prediction, symptom display, recommendations, and report generation work correctly and effectively.

5.10 Test Cases

Test cases constitute a pivotal element within the realm of software testing and quality assurance. Their primary purpose is to systematically assess the operational, performance, and reliability aspects of software applications. These test cases encompass a collection of directives, conditions, and anticipated results, enabling testers to pinpoint flaws or incongruities in the software. They span diverse scenarios, encompassing typical operations, boundary conditions, and error-handling procedures. Their overarching goal is to affirm that the software behaves in accordance with its intended functionality and aligns with user expectations. Ultimately, well-defined test cases stand as a cornerstone for delivering software of superior quality the following test cases are on both correct and wrong data to check if the data is accepted rightfully and validations are doing well.

5.10.1 Test Case 1

Title: Verify user authentication by the system

Precondition	**User Authentication**
Actions	Verify successful login with valid credentials.
Expected Result	User is logged in successfully with access to the mobile application.
Tested by	Khadija Rashid
Results	Verification successful

Table 5-2 Test Case1

Table 5-1: The system's test case is designed to validate user authentication, ensuring that the system accurately recognizes and authorizes individuals based on their provided credentials, such as email and passwords. This process guarantees secure access for authorized users.

5.10.2 Test Case 2

Title: Verify failed user authentication

Precondition	**Failed User Authentication**
Actions	Validate rejection of incorrect or invalid credentials.
Expected Result	System displays appropriate error message and prevents login.
Tested by	Khadija Rashid
Result	Verification Successful

Table 5-3 Test Case2

Table 5-2: The system's test case is intended to confirm the occurrence of unsuccessful user authentication. It entails testing the system's ability to accurately detect and block unauthorized access attempts.

5.10.3 Test Case 3

Title: Verify user signup functionality.

Precondition	**Signup Functionality**
Actions	Navigate to the signup page, enter valid email and password.
Expected Result	A new user account is successfully created and the user is directed to the login screen.
Tested by	Khadija Rashid
Result	Verification Successful

Table 5-4 Test Case3

Table 5-4 this test case verifies the system's ability to register new users by storing their credentials securely using firebase authentication. It ensures that only valid email formats and strong passwords are accepted, maintaining both accessibility and security.

5.10.4 Test Case 4

Title: Verify invalid login

Precondition	**Invalid Login**
Actions	Invalid/incorrect credentials.
Expected Result	User login fails with appropriate error.
Tested by	Khadija Rashid
Result	Verification Successful

Table 5-5 Test Case4

Table 5-5: The test case to validate an invalid login focuses on determining how the system reacts when users enter incorrect login details. It encompasses evaluating the system's ability to accurately detect invalid inputs, deliver relevant error messages.

5.10.5 Test Case 5

Title: Test Image upload

Precondition	Test Image Upload
Actions	Upload a Leaf image from gallery
Expected Result	App display whether the plant is healthy or not.
Tested by	Khadija Rashid
Result	Verification Successful

Table 5-6 Test Case5

Table 5-6: This test validates the image processing and disease prediction flow via the machine learning model.

5.10.6 Test Case 6

Title: Test Image capture via camera

Precondition	Test Image Capture
Actions	Take a picture of plant leaf.
Expected Result	System successfully captures the image and returns accurate disease result.
Tested by	Khadija Rashid
Result	Verification Successful

Table 5-7 Test Case6

Table 5-7: This ensures the mobile camera is properly integrated and compatible with the image detection pipeline.

5.10.7 Test Case 7

Title: Verify disease detection

Precondition	**Verify Disease Detect**
Actions	Tap the disease detect button.
Expected Result	The system process the image and display whether the plant is healthy or not.
Tested by	Khadija Rashid
Result	Verification Successful

Table 5-8 Test Case7

Table 5-8 this test case ensures that machine learning model integrated correctly into the application analyze the input image, perform disease classification, and returns accurate result with confidence level.

5.10.8 Test Case 8

Title: Show disease Symptoms

Precondition	**Disease Symptoms**
Actions	Tap the show symptoms button.
Expected Result	A list of symptoms related to the detect disease is displayed correctly.
Tested by	Khadija Rashid
Result	Verification Successful

Table 5-9 Test Case8

Table 5-9 this test case confirms that the system map display the symptoms after detection.

5.10.9 Test Case 9

Title: show treatment

Precondition	Show Treatment
Actions	Tap the show treatment button.
Expected Result	A display the preventive and treatment suggestion for the detected disease.
Tested by	Khadija Rashid
Result	Verification Successful

Table 9-10 Test Case 9

Table 5-10 this ensures the recommendations are relevant and align with the disease

5.10.10 Test Case 10

Title: Generate PDF report

Precondition	Generate report
Actions	Tap the generate PDF button.
Expected Result	A display the downloadable report that contains the image, confidence score, symptoms and recommendations.
Tested by	Khadija Rashid
Result	Verification Successful

Table 5-11 Test Case 10

Table 5-11 this test confirms the PDF report working correctly

Chapter 6
EXPERIMENTAL EVALUATIONS & RESULTS

6.1 Evaluation

The evaluation of plant disease detection focus on assessing the performance and accuracy of the system in real-world environment. The app was tested for its disease detection capabilities, user interaction flow, and responsiveness. Evaluation also include system testing based on predefined test cases. The goal was to ensure that the application meet functional and non-functional requirement such as a detection time, usability and reliability. The machine learning was analyzed using pre-collected dataset. The evaluation helped validate that the app is not only technically accurate but also practical and supportive for users, farmers and nursery staff managing in plant health.

6.2 Testbed

The system was tested on various mobile devices ensuring compatibility with device running. This hardware configuration allowed smooth app performance, quick image processing, and reliable user interaction. The backend of the application uses firebase which handle user authentication and stores data in cloud database. This ensure login/signup process and efficient storage of user profiles and detection result. The machine learning model is deployed using tensorflow lite, allowing lightweight on-device disease detection. The model was trained and evaluated using the PlantVillage dataset, a widely used high quality image dataset that contains labeled plant leaf images across various disease classes and healthy examples. Users tested the system by either capturing real-time images using the mobile camera or uploading from their gallery, giving flexibility for image input in different use cases. The application was tested under Wi-Fi and 4G internet connections to ensure fast communication between the app and backend services and to support real-time disease analysis and result generation.

6.3 Result and Discussion

The result of the evaluation demonstrates that the plant disease detection using mobile application performs effectively and efficiently under real-world conditions. The accuracy rate of the machine learning model confirms the robustness of the CNN trained using plantVillage dataset. This indicates that the model is capable of correctly identifying plant disease. Some photos take few seconds some take more seconds it's all depend on internet and model accuracy.

User feedback highlights the app intuitive interface, making it accessible even for non-technical users. The clear display of disease name, confidence score, symptoms and recommended treatment helps user not only detect problems but also understand and act upon them.

Moreover, the successful implementation of the PDF report generation feature adds practical value, allowing users to document disease occurrences and take prevention action based on recorded data. The apps integration with firebase ensured secure authentication and reliable cloud-based data handling without system crashes or data.

CHAPTER 7
CONCLUSION AND DISCUSSION

7.1 Strength of this Project

This project has a number of strengths theoretically and practically in agriculture, and particularly with nursery workers.

7.1.1 Easy-to-Use Mobile Interface

The application has a clean interface and an easy-to-use interface for anyone and non-technical users and nursery staff similar to recognize plant diseases with little need for guidance. This reduces the need for formal training and promotes general use, even in rural communities and those with non-literacy issues.

7.1.2 Deep Learning Accuracy

The system implements deep learning techniques such as Convolutional Neural Network (CNNs) for the identification and classification of various plant disease from leaf images. Timely detection gives the user a chance to move early and reduce the risk of the disease development.

7.1.3 Instant Detection Capability

By analyzing real-time images, the app enables on the spot disease identification, eliminating the need hard inspection. It can quickly identify strange symptoms, allowing for timely and effective disease control in the nursery.

7.1.4 Optimized for Low-End Devices

The detection system does not require expensive equipment because it is designed to function well on smartphones with low processing power. This makes the app very accessible in developing countries where experienced mobile devices are less common.

7.1.5 Treatment Suggestions

In additional to identifying plant diseases, the app offer personalized treatment recommendations to help nursery workers manage a range of plant diseases. It serves as a virtual assistant, helping users in managing the illness.

7.1.6 Download Plant Health Report

For every diagnosis, a digital PDF report is produced, enabling users to keep well-organized plant health records and monitor plant health history. Additionally, it adds a professional touch to nursery operations.

7.1.7 User Feedback Systems

The built-in feedback procedure helps with user feedback and suggestions, as well as feedback from use's experience with app. This will support ongoing app improvements and enable the system to adjust and improve in response to user needs.

7.1.8 Focus on Nursery level Applications

This integrated system was created especially for nurseries to take into account their unique needs, in contrast to general plant health apps, and it has more features that are relevant to nurseries than other apps. This contextualization can increase the system's relevance and link it to beneficial real-world effects.

7.2 Limitations and Future Work

Despite the project's encouraging results, there are still a number of limitations that present opportunities for improvement and development.

7.2.1 Limited Variety of Datasets

The PlantVillage dataset, which consists of clean, standardized photo taken under controlled circumstances, is the main source of training data for the model. The performance of the model may be unfriendly affected by the different lighting, background mess up, leaf direction, and are partial visibility that are frequent found in real-world nursery setting. When used in controlled settings, the system is less strong due to a lack of noisy, real-world data.

7.2.2 Requirement for Fixed Image Dimensions

Image with a certain fixed dimension can be accepted by the system. The model will be unable to process the image and may return an error rather than identify the disease if it is taken outside of this predetermined dimension, either too large or too small. This restricts user flexibility and requires that image be appropriately resized prior to use.

7.2.3 Dependency on the Internet for Certain Features

Some useful feature such as active treatment recommendation, feedback submission, or app updates, still requires an internet connection. This could have a negative effect on the user experience by restricting access to crucial app features in remote nursery locations or rural farms with not enough connectivity.

7.2.4 Identification Based on a Single Leaf

The app can only process one leaf image at a time at the moment, which may not accurately represent the health of the plant. Analyzing only one part of a disease could

result in a partial or inaccurate diagnosis because some diseases affect multiple parts (such as the stem, fruit, or entire plant).

7.2.5 Insufficient Multi-Language Help

The application is currently only accessible in one language, which is English. This limits the app's usability and accessibility for a large user base by creating a barrier for nursery workers on areas where English is not widely spoken or understood.

7.2.6 Limited Input Dimensionality

For disease detection, the current system solely uses image-based data of individual leaves. This restricts the system's comprehension of the entire context of the plant, including the soil, weather, and general plant structure. As a result, the model might overlook complicated system that call for more thorough examination than just leaf signals.

7.3 Future Work

As technology advances and user needs increase, there is always scope to improve the capabilities of the Plant Disease Detection Mobile Application. While the current system delivers the accurate and timely disease detection for nursery workers, future improvements can make the application more versatile. The following points outline key areas for potential development and refinement.

7.3.1 Increase and Vary Dataset

Gathering and training the model on a more varied dataset that includes actual and ore variety of photos will be significant improvement. To help the model perform accurately generalize outside of the testing facility, this includes images taken with different lighting conditions, backgrounds, plant angles and leaf types.

7.3.2 Boost the Dimensionality of the Features

To improve decision making for future iterations of the tool, we may include inputs of various kinds, such as data describing the growth stage of the plant, the temperature, soil health, or humidity. If we can include multimodal input, the model will be able to provide better predictive value by using more than just visual input.

7.3.3 Include Multiple Languages Support

To reach a large audience, especially in multi-language area, the app should have interfaces and support for local languages. Adding options like Punjabi, Urdu, or other regional languages can improve usability and boost acceptance.

7.3.4 Enhance Multi-Part plant Identification

The capability to scan multiple plant parts (such as stem, fruit, or flower) or analyze multiple leaves in a single session are examples of advanced updates. This would enable more through and accurate disease assessments and provide a more comprehensive picture of the plant's condition.

7.3.5 Boost Performance Offline

If all of the app's features, such as treatment recommendations and feedback, are made to function fully offline, it will be more beneficial n remote locations with poor internet access. Data can be locally stored and then synchronized when the internet is accessible.

7.3.6 Access to Nursery Management Systems

The next step might be to link the app with nursery inventory systems. The ability of nursery employees to automate record-keeping, link health records to particular plants or groups, and track disease trends would improve operational planning and efficiency.

7.3.7 Simple Image Preprocessing

In order to address the issue of fixed input size, future updates may automatically resize or modify uploaded images internally, removing the need for the user to complete this by hand. This will make system ore flexible and easy to use.

7.4 Reasons for Failure – If Any

For a variety of reasons, the system may occasionally make mistakenly identify or fail to protect plant diseases:

- The input images have fixed size in order to be used by the application. The system might generate an error or fail to rad the image if the uploaded image is large or smaller than the one of these sizes.
- The model may not accurately identify patterns linked to the disease and might produce inaccurate or no results at all if the captured image is fuzzy, too dark , too light, or of poor quality
- The background image used to train model was clear and consistent. On the other hand , real-world nursery cases may include shadows, dirt, a human hand, or tools in the background which confuses the system and lowers accuracy
- Only illnesses that the present in the training dataset can be detected by the mode. This means that the model may incorrectly classify or provide no

result if the user upload the picture of leaf isn't present is the dataset.

- In, certain cases the illness in the early stages and present with few symptoms. If the symptoms are wild, the model might just search the dataset for diseases with similar symptoms, neglect any symptoms, or this the plant is healthy.
- Disease that impact other parts, such as stems, flowers, and roots, may go unnoticed or unidentified because the app only analyze one leaf, and it would not offer a complete diagnosis.
- Non-visual information that have an important impact on disease development, such as soil or air moisture levels, soil health, and earth temperature, cannot be taken into account by the system the model is unable to offer an accurate assessment if these conditions are not taken into consideration.

REFERENCES

[1] W. B. Demilie, J. Greening, K. Scab, and L. Melanose, “Plant disease detection and classification techniques: a comparative study of the performances,” *Journal of Big Data*, vol. 11, no. 5, 2024, doi:10.1186/s40537-023-00863-9.

[2] G. Shrestha and M. Das, “Plant disease detection using CNN,” *2020 IEEE Applied Signal Processing Conference (ASPCON)*, 2020, doi:10.1109/ASPCON49795.2020.9276800.

[3] N. I. Nivetha and M. Padmaa, “Tree leaves and fruits disease detection using convolutional neural networks,” *International Journal of Scientific Research and Engineering Development*, vol. 2, no. 3, pp. 1–6, 2018.

[4] A. A. Abdelmoamen and G. H. Reddy, “A Mobile-Based System for Detecting Plant Leaf Diseases Using Deep Learning,” *AgriEngineering*, vol. 3, no. 3, pp. 478–493, 2021, doi:10.3390/agriengineering3030032.

[5] N. Bhise, S. Kathet, S. Jaiswar, and A. Adgaonkar, “Plant disease detection using machine learning,” *International Research Journal of Engineering and Technology (IRJET)*, vol. 7, no. 7, pp. 2924–2929, Jul. 2020.

[6] M. H. Saleem, J. Potgieter, and K. M. Arif, “Plant disease detection and classification by deep learning,” *Plants*, vol. 8, no. 11, Art. 468, 2019, doi:10.3390/plants8110468.

[7] T. Pandit, N. Thombare, R. Kazi, and M. Pangaonkar, “Smart plant disease detection,” *International Journal of Research Publication and Reviews*, vol. X, no. Y, pp. A–B, 2025.

[8] P. Kulkarni, A. Karwande, T. Kolhe, S. Kamble, A. Joshi, and M. Wyawahare, “Plant disease detection using image processing and machine learning,” *arXiv preprint arXiv:2106.10698*, 2021, doi:10.48550/arXiv.2106.10698.

[9] P. S. Ghodekar, V. Yermune, A. Sable, R. Mandhare, and M. Bhosale, “Plant leaf disease detection using CNN,” *International Research Journal of Modernization in Engineering Technology and Science*, vol. 5, no. 3, Apr. 2023.

[10] M. A. Jasim and J. M. Al-Tuwaijari, “Plant leaf diseases detection and classification using image processing and deep learning techniques,” in *2020 International Conference on Computer Science and Software Engineering (CSASE)*, Apr. 2020, pp. 259–265, doi:10.1109/CSASE48920.2020.9142097.

[11] K. T. Label, K. T. Label, F. Kurfess, M. Pantoja, and S. Ding, “Plant Disease Detection Through Convolutional Neural Networks: A Survey of Existing Literature, Best Practices, and Implementation,” *M.S. thesis*, California Polytechnic State University, San Luis Obispo, Dec. 2021, doi:10.15368/theses.2021.162.

[12] S. Kaur, S. Pandey, and S. Goel, “Plants disease identification and classification through leaf images: A survey,” *Archives of Computational Methods in Engineering*, vol. 26, no. 2, pp. 507–530, Apr. 2019, doi:10.1007/s11831-018-9271-7.

[13] N. Shelar, S. Shinde, S. Sawant, S. Dhumal, and K. Fakir, “Plant Disease Detection Using CNN,” *ITM Web of Conferences*, vol. 44, 03049, 2022, doi:10.1051/itmconf/20224403049.

[14] F. Badar and A. Naaz, “Fruit leaves disease detection: a review,” *International Journal of Agriculture and Environmental Research*, vol. 8, no. 2, pp. 318–325, Apr. 2022, doi:10.22004/ag.econ.333839.

[15] S. Uma, “Deep learning based plant disease prediction using convolutional neural network,” *Journal/Conference Name*, vol. X, no. Y, pp. A–B, 2022.

[16] S. Malathy et al., “Disease Detection in Fruits using Image Processing,” in *2021 6th International Conference on Inventive Computation Technologies (ICICT)*, Jan. 2021, doi:10.1109/ICICT50816.2021.9358541.

[17] V. Sathya, H. Rafidha, and G. S. Rani, “Fruit and leaves disease prediction using deep learning algorithm,” *International Research Journal of Multidisciplinary Technovation*, vol. 1, no. 5, pp. 8–16, Sep. 2019, doi:10.34256/irjmt1952.

[18] A. Ashwika, V. Shetty, A. Ashika, A. Amshitha, D. Deeksha, and L. Gulappagol, “Leaf disease detection using image processing,” *Conference/Journal Name*, Year.

APPENDICE

APPENDIX A. Project Proposal

1. Introduction

Detection of plant diseases is essential for food security and agricultural output. Plant diseases significantly impact global agriculture by reducing yields and affecting trade. Ensuring nurseries produce healthy plants is crucial for sustainable farming. However, current disease diagnostic methods, often costly and labor intensive, are not ideal for quick on-site use in nurseries. There's a pressing need for new, fast, and cost-effective diagnostic technologies that can identify plant diseases directly where plants are grown. This project aims to explore technologies to improve disease detection in nurseries, enhance plant health management, and support sustainable agriculture worldwide.

2. Objectives

The objective of plant disease detection using leaf images is to create a model that can identify diseases in plants by analyzing images of their leaves. This model should work well despite external factors like the environment, noise, and background. By using Convolutional Neural Networks (CNNs), which are very effective for this task, the system can accurately detect various plant diseases. This approach not only improves performance but also makes the project easy to maintain and manage.

3. Problem Description

The project aims to develop a mobile application tailored for nurseries that facilitates the early detection of plant diseases. This application will utilize advanced image processing techniques, particularly Convolutional Neural Networks (CNNs), to analyze leaf images captured using mobile devices. By employing machine learning algorithms, the app will accurately identify various diseases affecting plants based on visual symptoms exhibited on their leaves. The ultimate goal is to provide nursery workers with a reliable tool that automates and enhances the process of disease detection, thereby ensuring prompt intervention and minimizing the spread of infections. Early detection of plant diseases is critical in nurseries to maintain plant health and prevent significant economic losses. Current methods, predominantly reliant on manual inspection by trained experts, are time-consuming, costly, and subject to human error. These limitations often result in delayed diagnosis and ineffective disease management strategies. By introducing a mobile application equipped with AI-driven disease detection capabilities, nurseries can streamline

their operations, reduce dependency on specialized expertise, and promptly address emerging health issues in their plant populations. This technological advancement not only improves efficiency but also enhances overall nursery management practices, leading to healthier plants and sustainable agricultural practices.

4. Methodology

The project followed a phased approach where each step was carefully planned, developed and tested. Initial search helped identify the real world need for plant disease detection. The system was designed using modular architecture to support features like disease detection, symptoms explain and recommendation generate. Image processing and machine learning technique were used to detect disease, while a mobile friendly interface was designed for ease of use. Testing was integrated for each phase following the v-model structure. V-Model demonstrates the relationships between each phase of the development life cycle and its associated phase of testing.

5. Project Scope

The scope of this project is to create a mobile application that is easy to use for users, incorporating many cutting-edge technologies to improve agricultural sustainability and production. The following are the main elements of the project scope:

1. User-Friendly Interface: Develop a user-friendly web application for automated plant disease detection. Implement machine learning models for disease classification using image processing techniques.

2. Image-Based Disease Detection: Develop a feature that uses a Convolutional Neural Network (CNN) based on MobileNet V2 architecture to identify plant diseases from images captured or uploaded by the users. Upon detection, users will receive detailed information about the disease, including its symptoms and possible causes.

3. Instant Alerts: Immediate alerts will notify users as soon as a disease is detected in any uploaded image. These notifications will include recommendations for immediate actions to prevent the spread of the disease, helping nursery workers respond quickly and effectively.

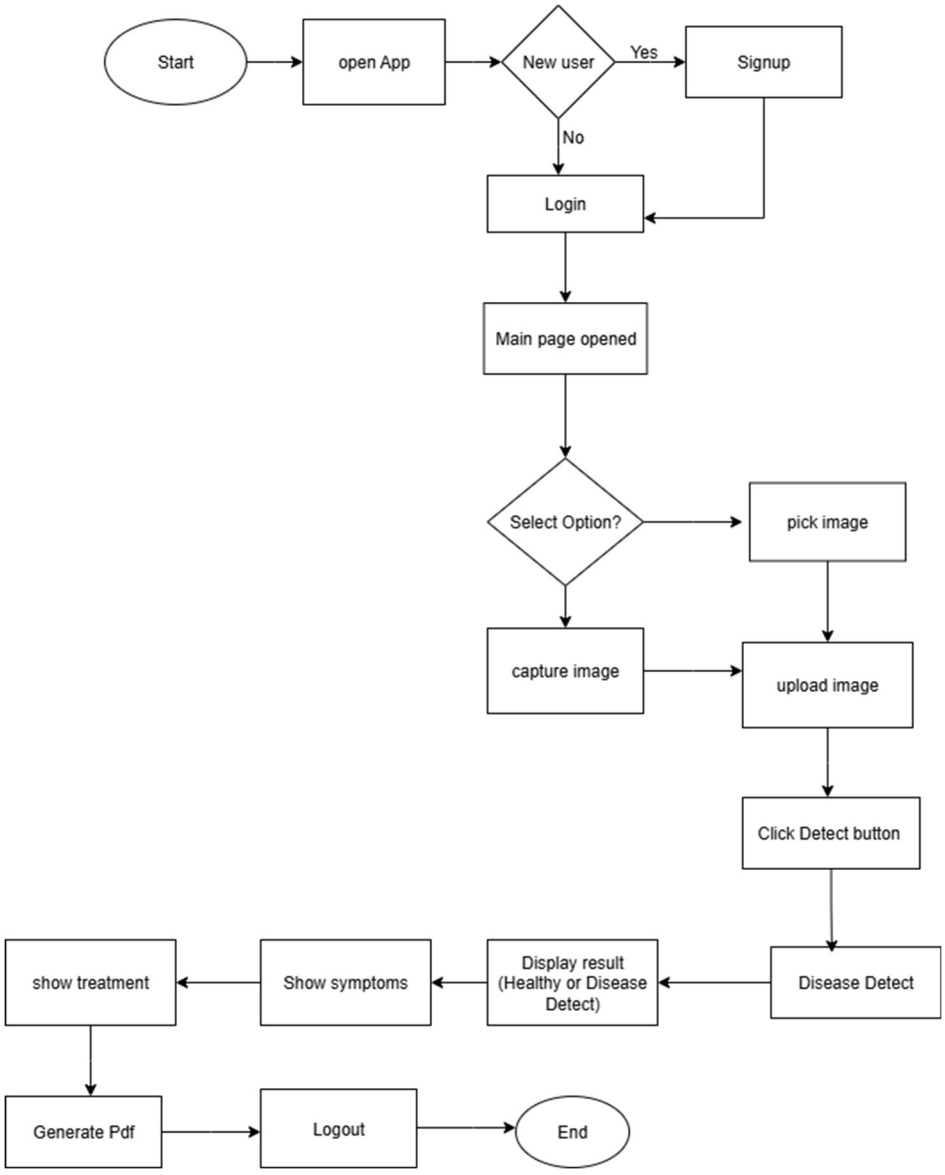
4. Treatment Recommendations: For each detected disease, the app will offer suggested treatments, tailored to the specific conditions of the nursery. Users will receive step-by-step guides on how to apply these treatments, ensuring that they can take the necessary steps to manage and eradicate the disease.

5. Data and Reports: The app will maintain a history of all disease detections and treatments, allowing users to generate reports on plant health and disease trends. This historical data can be invaluable for identifying recurring issues and improving overall plant care strategies.

6. Community Forum: A built-in community forum will enable users to share their experiences, tips, and advice with one another. This platform will foster a supportive network of nursery workers who can collaborate and learn from each other's experiences.

7. Updates and Improvements: Update the app with new features, disease databases, and improved algorithms. Collect user feedback to continuously enhance the app's functionality and user experience.

6. Flowchart



7. Feasibility Study

The feasible study for the plant disease detection using mobile application assesses the possibility of developing a novel approach to enhancing agricultural practices using technology. This software addresses important farming concerns like fast disease identification, effective fertilizer management, and access to local weather and resources. It does this by combining advanced image processing, machine learning, and real-time data analysis. In order to increase agricultural productivity and sustainability, the app automates disease identification and offers real-time data.

i. **Risk involved:** The plant disease detection using mobile application faces several risks that could affect how well it is implemented. One of the main hazards is the technical difficulty in integrating third-party APIs. Project schedule delays could result from integration problems with these APIs, which are essential for delivering location-based services and real-time meteorological data. The precision and dependability of the machine learning model employed to detect plant diseases present another important danger. Furthermore, there are risks associated with user adoption and engagement because low initial uptake or unfavorable feedback may prevent application usage and acceptance. Project deadlines, customer happiness, and legal compliance may all be impacted by these risks.

ii. **Resource involved:** To mitigate risk and ensure successful delivery of the Plant Disease Detection using Mobile Application. These resources include dependable cloud services for backend hosting and machine learning model training, as well as high performance development computers. It is essential to have access to extensive and varied datasets in order to train and improve the illness detection algorithms. Effective development and deployment require a talented team with experience in machine learning, project management, development, UI/UX design, and QA testing. Collaboration tools are necessary for team members to handle tasks and communicate with each other effectively. Budgetary allocations for user outreach and marketing are also essential to encourage adoption and guarantee the application successfully serves the needs of the agricultural community.

8. Solution Application Areas

The Mobile Application targets small to medium-scale farmers, offering real-time data and community support to enhance crop management, optimize resources, and boost productivity. It uses computer vision and machine learning to detect plant diseases early, reducing crop losses. The fertilizer calculator provides tailored recommendations, minimizing waste and costs. Real-time weather data helps farmers make informed decisions about planting, and harvesting, mitigating weather-related risks. The app also offers information on nearby plant shops, making resources easily accessible. Additionally, a community forum enables farmers to share knowledge and best practices. Overall, the application promotes sustainable farming and improves farm productivity through technology and collaboration.

9. Tools/Technology

- Software Requirements:
  - i. VS Code
  - ii. Jupiter Notebook
  - iii. Anaconda
  - iv. Android Studio
  - v. Emulator
  - vi. Firebase
- Hardware Requirements:
  - vii. Laptop
  - viii. USB
  - ix. Mobile Phone

APPENDIX B. Software Requirement Specification (SRS)

1. Introduction

Detection of plant diseases is crucial for maintaining healthy plants and improving nursery productivity. This project aims to develop a mobile application specifically designed for nursery workers to efficiently detect plant diseases. By utilizing advancements in deep learning and image processing, the application will analyze captured or uploaded images of plant leaves to identify potential diseases. The application will offer a user-friendly interface, making it accessible even for individuals with minimal technical expertise. By focusing on reliable disease detection, this tool aims to streamline plant health assessment in nurseries and reduce manual effort. In recent years, deep learning methods have greatly improved plant disease detection capabilities. One study highlights the effectiveness of convolutional neural networks (CNNs) in identifying plant diseases through image analysis. The study demonstrates how CNN models can accurately classify diseases by analyzing leaf images, facilitating early detection and timely intervention in agricultural practices. This approach surpasses traditional manual methods, as it can quickly process large volumes of plant images with enhanced accuracy and efficiency. By efficiently analyzing numerous images in a short period, CNN-based systems reduce the need for expert involvement and manual inspections, positioning them as a vital tool for modern agricultural practices.

1.1 Purpose of Document

The purpose of this Software Requirement Specifications (SRS) document is to provide detailed insights into the design considerations, architecture, and functionality of the Plant Disease Detection Using Mobile Application (Nursery Base). This document serves as a blueprint for the development team, outlining the technical requirements and specifications necessary for the successful implementation of the application. The application leverages deep learning and computer vision techniques to accurately identify plant diseases and aims to assist users, such as nursery owners and farmers, in managing plant health effectively.

1.2 Motivation

The question of why develop a software solution that already exists and is deployed in the industry is valid and can be addressed by highlighting the unique aspects and advantages of this project. While various plant disease detection systems are available in the market, many of them may have limitations such as high cost, complex interfaces, or limited accessibility. This project aims to address these gaps and provide a more practical, user-friendly, and accessible solution for plant disease detection. The primary motivation behind this project is to develop a mobile-based plant disease detection system that is cost-effective, easy to use, and accessible to a wider range of users. By leveraging the power of deep learning, particularly Convolutional Neural Networks (CNNs) such as MobileNetV2, this system aims to provide accurate disease detection while overcoming challenges faced by traditional systems.

1.3 Intended Audience

The intended audience for this project includes nursery workers, who will directly use the plant disease detection application for improved plant health management. It also target developers, academic supervisors involved in guiding and building the system and also researchers studying plant disease who can use the application. Additionally, this document serves as a guide for project team members, including designers, developers, and researchers, to understand the system's objectives, requirements, and deliverables.

2. Overall System Description

2.1 Project Background

Plant diseases significantly affect the productivity of nurseries, leading to economic losses and the wastage of valuable resources. Traditional disease detection methods such as manual observation often demands expert knowledge that nursery workers may lack. However, these methods can be time-consuming, labor-intensive, and may not be readily accessible to all nursery workers, particularly those in remote locations. Artificial Neural Networks (ANNs) are effective tools for managing complicated data, drawing inspiration from the analytical and processing powers of the human brain. Because they can recognize spatial patterns and hierarchies in visual data, Convolutional Neural Networks

(CNNs) are the best of these for image-related applications. For this project, the MobileNetV2 architecture a lightweight and mobile-optimized version of CNN is employed to ensure accurate and efficient plant disease detection. Recent advancements in computer vision and deep learning, specifically the emergence of Convolutional Neural Networks (CNNs), have demonstrated remarkable potential in image based disease detection. This project leverages these challenges by developing a mobile application that uses AI-powered image processing techniques to detect plant diseases.

2.2 Problem Statement

The detection of plant diseases plays a vital role in modern agriculture, yet traditional approaches are often slow and rely heavily on expert knowledge. Recent progress in deep learning, particularly with Convolutional Neural Networks (CNNs), has proven to significantly enhance the precision and speed of plant disease detection. A comparative evaluation of various detection methods indicates that while traditional techniques face challenges with scalability and real-time processing, deep learning models present a promising solution to these issues. This project aims to address these challenges by developing a mobile application that utilizes deep learning techniques, specifically CNNs to automate plant disease detection. By providing nursery workers with a rapid and accurate diagnostic tool, this application will enhance disease management practices, minimize crop losses, and contribute to a more sustainable and resilient agricultural system.

2.3 Project Scope

- Focuses on detecting and classifying plant diseases from images captured through mobile devices.
- Utilizes Convolutional Neural Networks (CNNs), specifically optimized models like MobileNetV2.
- The project will integrate lightweight models like MobileNetV2 for efficient and accurate processing on mobile devices.
- The app will feature an easy-to-use interface for nursery owners, Includes features like image upload, real-time detection, and logout functionality.

2.4 Not In Scope

- Doesn't include external farming tools or equipment like drones and sensors.
- No advanced editing features for uploaded images like manual adjustments or filters.

2.5 Project Objectives

The primary objectives of plant disease detection is to identify the disease accurately and quickly by utilizing deep learning Convolutional Neural Networks (CNNs) model. This is important to facilitate early disease detection to minimize crop losses and promote sustainable agriculture practices.

2.6 Stakeholders & Affected Groups

- Nursery workers
- Home Gardeners
- Software Development Team
- Supervisor

2.7 Operating Environment

The system will run on both Android and iOS devices, leveraging Flutter's cross-platform capabilities, using optimized frameworks for efficient deep learning on mobile devices. It will be designed for smartphones with sufficient processing power for real-time detection, offering offline functionality, with an internet connection required for cloud updates.

2.8 System Constraints

- Mobile devices must have a functioning camera.
- Detection accuracy relies heavily on image quality, including lighting and focus.
- An internet connection is required for updates, though offline functionality is supported for image uploads.

2.9 Assumptions & Dependencies

System relies on the availability and utilization of the PlantVillage dataset as a fundamental resource for the development and evaluation of the deep learning

model. This open-access repository of plant health images offers a vast and diverse range of categorized images essential for effectively training the CNN model. The system assumes that users will utilize mobile devices to capture and upload images of plant leaves, with the real-time detection and disease classification process designed to operate smoothly on these mobile platforms.

3. External Interface Requirement

3.1 Hardware Interfaces

- Mobile phone with a functionally camera for capturing images.
- No other external hardware interface needed.

3.2 Software Interfaces

- **OpenCV:** OpenCV is utilized for various image processing tasks such as resizing, cropping, and improving the quality of plant leaf images before they are fed into the deep learning model for classification.
- **TensorFlow:** TensorFlow is employed to create and implement deep learning models, with TensorFlow Lite being used to deploy the trained model on mobile devices for real-time plant disease detection. It ensures efficient model inference on mobile platforms with constrained resources.
- **PlantVillage Dataset:** The PlantVillage dataset offers images of plant diseases, which are used for training, validating, and testing the deep learning model. It contains labeled images of plant leaves, categorized into different disease types and healthy conditions.
- **Firebase:** Firebase is utilized for storing images, user information, and disease diagnosis outcomes. It also offers authentication services for user management and ensures smooth synchronization between the app and the cloud backend.
- **External Libraries (NumPy and Pandas):** Provide essential functionality for data preprocessing and analysis.

3.3 Communications Interfaces

- The internet connection plays an important role which is updating the data in real time.

4. System Functions / Functional Requirements

4.1 System Functions

- Signup Form: For new users to create account
- Login Form: For existing users to login to the system
- Select Camera: Users can capture images using the camera.
- Select Gallery: Users can upload images of plants for disease detection.
- Detect Disease: AI processes the image and identifies the disease.
- Logout Form: To Logout from app.

4.2 Function Categories

Ref #	Functions	Category	Attribute	Details & Boundary Constraints
R1.1	**Upload Image**	Evident	System Response time	Uploads plant images within 5 seconds on a stable internet connection.
R1.2	**Capture Image**	Evident	System Response time	Camera opens instantly to capture clear plant images.
R1.3	**Detect Disease**	Evident	Accuracy	Identifies diseases using the CNN model (MobileNetV2).
R1.4	**Logout**	Evident	Data Security	Ensure secure user logout
R1.5	**Download Disease Report**	Frill	Future Feature	Enables users to generate and download detailed disease reports (planned for FYP-2).

4.3 Use Cases

List of Actors

- User: The primary user who interacts with the mobile app to upload images and receive disease detection results.
- System/Server: Handles backend processing like disease detection and fetching disease details.

List of Use Cases

- **Upload Image:** The user uploads a plant image for analysis.
- **Capture Image:** The user captures an image using the device camera.
- **Preprocess image:** The system preprocesses the uploaded image to ensure it is ready for disease detection.

- Train model: The CNN model is trained using labeled plant disease images to enable accurate disease classification.
- Fetch Disease Details: The system retrieves and displays detailed information about the detected disease
- **Detect Disease:** The system processes the image and detects potential plant diseases.
- **View Disease Details:** Displays detected disease information.
- **Logout:** Allows the user to securely log out of the application.

4.4 Use Case Diagram

4.5 Description of Use Cases

Use Case Name	**Detect Disease**
Actors	Nursery Worker, system/server
Purpose	To identify plant diseases based on uploaded or captured images.
Preconditions	The user must be logged into the application and have an image ready for upload or capture.
Postconditions	The system displays the disease details.
Normal Flow	1. User logs into the app.2. Uploads or captures an image.3. Clicks the "Detect" button.4. System analyzes the image and displays results.

5. Non-functional Requirement

5.1 Performance Requirements

The application should analyze uploaded images and generate results within 5 seconds to provide a smooth user experience. This fast processing time is crucial for nursery workers who may require immediate information to act promptly. Achieving this will require optimized algorithms and efficient use of hardware or cloud resources.

5.2 Safety Requirements

User data must be securely stored and managed to prevent loss, unauthorized access, or corruption. This includes implementing encryption, access controls, and regular backups. Ensuring data safety is essential for building user trust and adhering to data protection laws.

5.3 Security Requirements

If your system involves any data collection or online components, security features such as data encryption, user authentication, and secure data storage need to be mentioned. This might also involve protecting sensitive user data if applicable. Login credentials provided by users should be encrypted to safeguard against cyberattacks and unauthorized access. Proper security measures are necessary to protect sensitive user data and minimize system vulnerabilities.

s

5.4 Reliability Requirements

The system must ensure reliability, making the application consistently available for users. Achieving high reliability requires regular maintenance, backup systems, and robust testing to avoid unexpected outages. This is critical for users who rely on the app for timely results.

5.5 Usability Requirements

The app's interface must be simple and user-friendly, catering to non-technical users like nursery workers. Responsive design to ensure accessibility on various devices. A well-designed interface ensures the app is accessible and encourages consistent use.

5.6 Maintainability Requirements

The system must be designed for easy maintenance to ensure long-term functionality and adaptability. This includes writing clean, modular, and well-documented code to facilitate updates or bug fixes. Additionally, a robust error-logging mechanism should be in place to quickly identify and resolve issues, reducing downtime and maintenance efforts.

5.7 User Documentation

A detailed user manual should be created to describe the technical and functional details of the project. This manual must outline the system's architecture, core features, development process, and the technologies used. It should also include system requirements, and maintenance guidelines to provide a clear understanding of the project.

APPENDIX C. DESIGN SPECIFICATIONS

1. Introduction

Detection of plant diseases is crucial for maintaining healthy plants and improving nursery productivity. This project aims to develop a mobile application specifically designed for nursery workers to efficiently detect plant diseases. By utilizing advancements in deep learning and image processing, the application will analyze captured or uploaded images of plant leaves to identify potential diseases. The application will offer a user-friendly interface, making it accessible even for individuals with minimal technical expertise. By focusing on reliable disease detection, this tool aims to streamline plant health assessment in nurseries and reduce manual effort.

In recent years, deep learning methods have greatly improved plant disease detection capabilities. One study highlights the effectiveness of convolutional neural networks (CNNs) in identifying plant diseases through image analysis. The study demonstrates how CNN models can accurately classify diseases by analyzing leaf images, facilitating early detection and timely intervention in agricultural practices. This approach surpasses traditional manual methods, as it can quickly process large volumes of plant images with enhanced accuracy and efficiency. By efficiently analyzing numerous images in a short period, CNN-based systems reduce the need for expert involvement and manual inspections, positioning them as a vital tool for modern agricultural practices.

2. Purpose of Document

This document presents the design framework for the plant disease detection mobile app. It is designed for developers, testers, supervisors, and future contributors. Adopting an Object-Oriented Design (OOD) approach, the document offers instructions for constructing, managing, and enhancing the app. It outlines the system architecture, database schema, and user interaction flow, promoting clear communication among teams. Furthermore, it acts as a reference for quality assurance and future revisions, ensuring alignment and supporting smooth development.

3. Intended Audience

The intended audience for this project includes nursery workers, who will directly use the plant disease detection application for improved plant health management. It also target developers, academic supervisors involved in guiding and building the system and also researchers studying plant disease who can use the application. Additionally, this document serves as a guide for project team members, including designers, developers, and researchers, to understand the

system's objectives, requirements, and deliverables.

4. Document Convention

This document is prepared in Times New Roman 12pt font for the text and 14pt bold for section headers and 16 for main heading.

5. Project Overview

The Mobile Application for Plant Disease Detection is developed to help nursery workers and gardeners identify plant diseases through image analysis. It employs deep learning, specifically Convolutional Neural Networks (CNNs), to analyze plant images and detect potential diseases. By integrating TensorFlow Lite for on-device processing, the app ensures efficient performance without requiring continuous internet access.

The main goal of the project is to offer an intuitive tool that allows users to upload plant images, receive diagnostic results, and generate downloadable reports. The system utilizes MobileNetV2, a lightweight CNN model optimized for mobile devices, for disease classification. This approach ensures quick and accurate plant disease detection on smartphones, enabling users to take prompt actions to safeguard their plants.

6. Scope

- Focuses on detecting and classifying plant diseases from images captured through mobile devices.
- Utilizes Convolutional Neural Networks (CNNs), specifically optimized models like MobileNetV2.
- The project will integrate lightweight models like MobileNetV2 for efficient and accurate processing on mobile devices.
- The app will feature an easy-to-use interface for nursery owners, Includes features like image upload, real-time detection, and logout functionality.

7. Not In Scope

- Doesn't include external farming tools or equipment like drones and sensors.
- No advanced editing features for uploaded images like manual adjustments or filters.

8. Design Considerations

The Design Considerations section discusses the key decisions shaping the system's structure and function. It ensures the architecture aligns with project objectives, prioritizes usability, and remains adaptable to future enhancements. This section also highlights how the design addresses constraints, scalability, error handling, and the efficient integration of external systems like TensorFlow.

9. Assumptions and Dependencies

System relies on the availability and utilization of the PlantVillage dataset as a fundamental resource for the development and evaluation of the deep learning model. This open-access repository of plant health images offers a vast and diverse range of categorized images essential for effectively training the CNN model. The system assumes that users will utilize mobile devices to capture and upload images of plant leaves, with the real-time detection and disease classification process designed to operate smoothly on these mobile platforms.

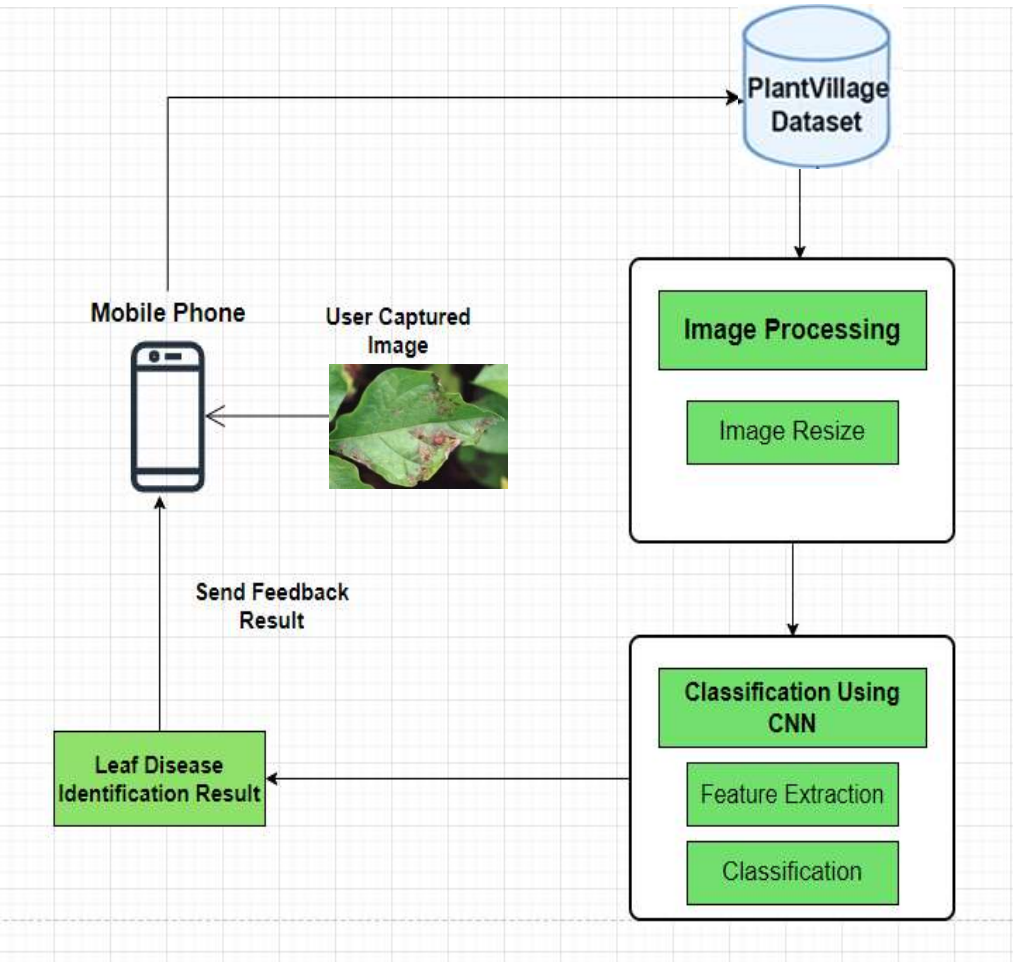
10. Risks and Volatile Areas

- Data Quality: The effectiveness of the disease detection model is closely tied to the quality and variety of the data used for training. Inaccurate or limited data could lead to incorrect diagnoses, reducing the app's overall reliability.
- Algorithm Performance: The performance of the deep learning model could be affected by slow processing speeds or its inability to accurately recognize new diseases or plant species, which may impact the app's efficiency and accuracy.
- User Adoption and Satisfaction: For the app to succeed, it is crucial that users find it easy to use and reliable. If the app is complicated, prone to errors, or fails to deliver consistent results, it could lead to reduced user engagement and abandonment.
- Regulatory Compliance: The app must meet all necessary data protection and privacy standard. Failing to ensure proper data security could lead to legal complications and erode user trust.
- Resource Constraints: Mobile devices have limited processing power and storage capacity, which may affect the app's performance. These limitations could cause slow processing or even crashes, particularly when handling large images or datasets.
- Integration Challenges: Integrating the deep learning model with the mobile platform and external services like databases could face technical obstacles. Compatibility issues or

integration failures might disrupt the app's functionality and reliability.

11.System Architecture

The system architecture of the Plant Disease Detection Mobile Application is designed with a modular structure to ensure smooth functionality. The user interface begins with a homepage, offering options to log in or sign up. After authentication, users access the main page, where they can either capture plant images using the mobile camera or upload images from the gallery. These images are analyzed by a disease detection engine, which utilizes preloaded datasets and algorithms to identify plant diseases, such as bacterial spots. The analysis results, including the plant's health status and detected disease, are displayed to the user. The architecture also incorporates a database layer to manage user details, uploaded images, and detection results. Furthermore, the app includes features for secure authentication and logout, ensuring reliability, security, and effective disease detection for nursery workers.

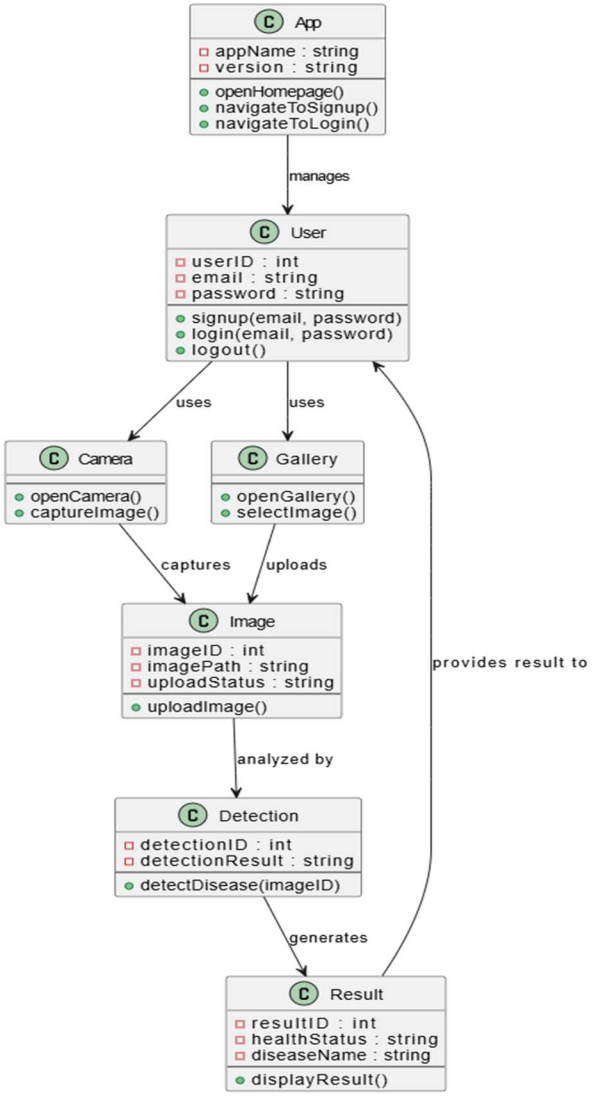


12.Design Strategy

1. Future System Extension or Enhancement: To accommodate future system extension or enhancement, the system architecture is designed with modularity and scalability in mind. The use of a layered architecture allows for the independent development and modification of individual components. Each component has well-defined interfaces, facilitating the addition of new features without impacting the entire system. This design decision enables the system to adapt and grow as new requirements emerge.
2. System Reuse: The system architecture aims to maximize reusability of components and modules. By separating concerns into distinct layers, components can be reused across different projects or systems.
3. User Interface Paradigms: The user interface is designed to be intuitive and user-friendly. The system employs standard user interface paradigms and design principles to ensure familiarity and ease of use for the users. The user interface design follows established best practices to enhance usability and improve the overall user experience.
4. Concurrency and Synchronization: The system design considers concurrency and synchronization to handle potential concurrent access to shared resources. For example, if multiple users are accessing the application simultaneously, the system ensures that there are no conflicts in data access.

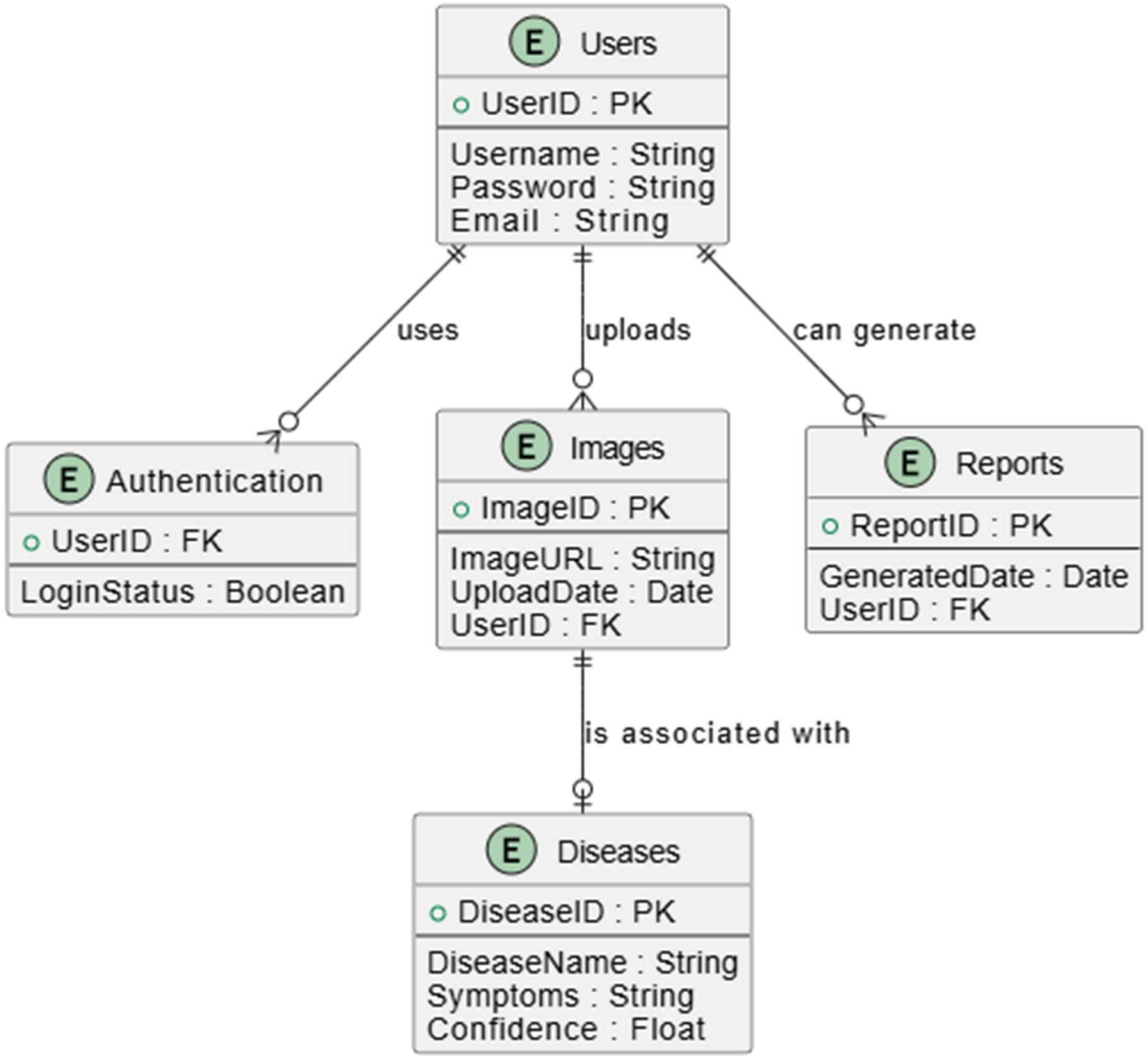
13.Detailed System Design

5.8 Design Class Diagram



5.9 Database Design

ER Diagram



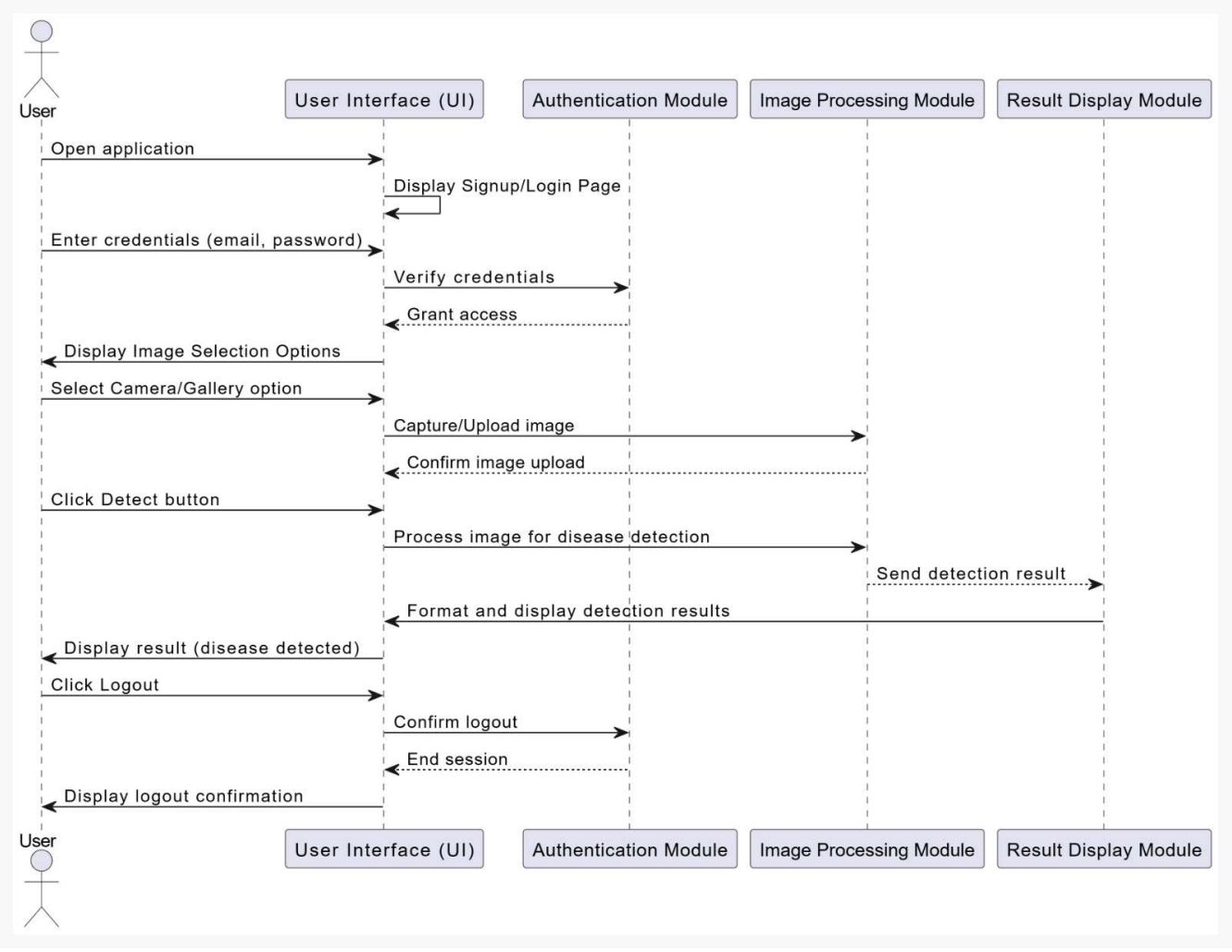
Data Dictionary

Data Name	Type	Description	Possible Values/Format
User Email	String	Email address entered during Signup or Login for user authentication.	Example: user@example.com
User Password	String	Password entered during Signup or Login for user authentication.	Minimum 8 characters, including letters, numbers, and symbols.
Camera Option	Boolean	Represents the user selecting the camera to capture an image.	true (selected), false (not selected).
Gallery Option	Boolean	Represents the user selecting the gallery to upload an image.	true (selected), false (not selected).
Plant Image	File (Image)	The image file selected or captured by the user for disease detection.	JPEG, PNG formats supported.
Upload Status	String	Status of the image upload process.	Pending, Success, Failed.
Detection	String	Status of the disease detection process.	Pending, In Progress, Completed.

Data Name	Type	Description	Possible Values/Format
Status			
Detection Result	String	Result of disease detection indicating whether the plant is healthy or a disease is detected.	Healthy, Bacterial Spot, etc.
Logout Action	Boolean	Represents whether the user logs out of the system after viewing the result.	true (user logged out), false (still logged in).

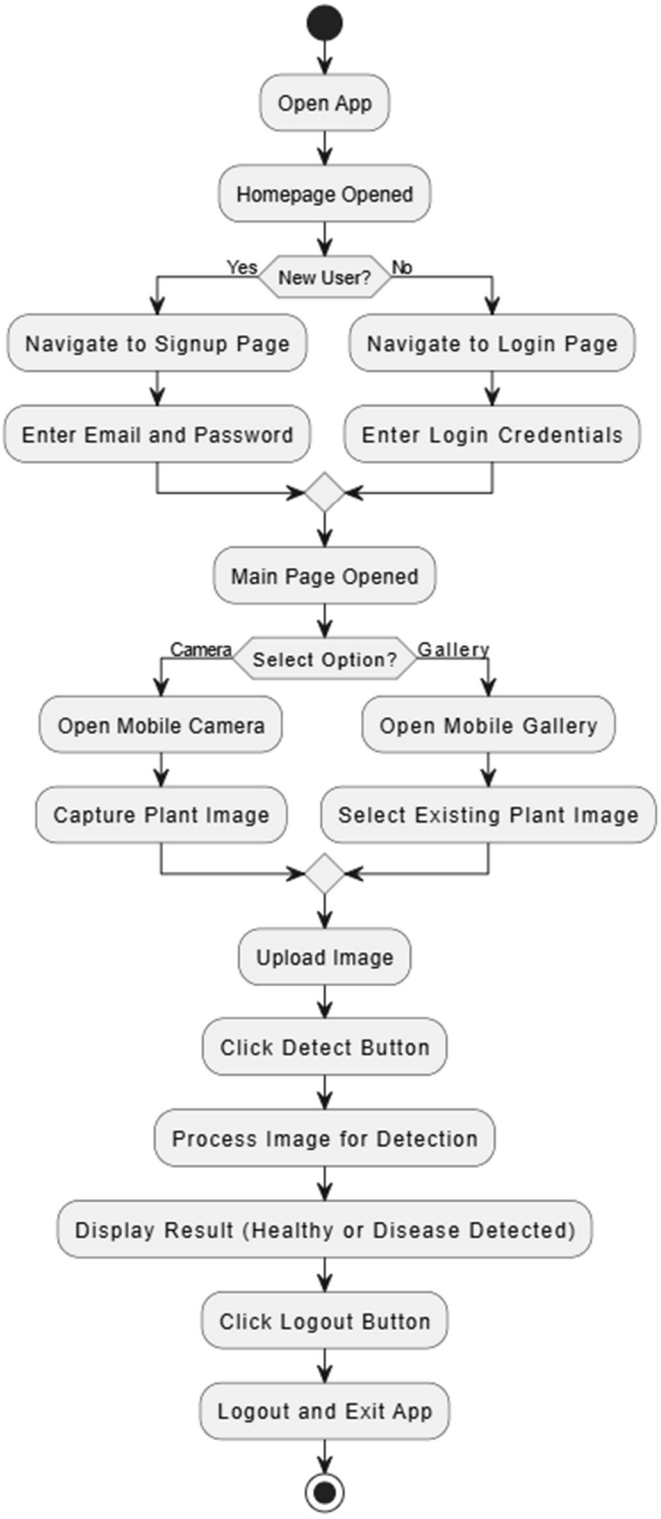
14. Application Design

5.10 Sequence Diagram



5.11 UseCase Diagram

5.12 State Diagram



APPENDIX D. OTHER TECHNICAL DETAIL DOCUMENTS

1. Test Cases Document

Test Case 1

Title: Verify user authentication by the system

Precondition	**User Authentication**
Actions	Verify successful login with valid credentials.
Expected Result	User is logged in successfully with access to the mobile application.
Tested by	Khadija Rashid
Results	Verification successful

Test Case 2

Title: Verify failed user authentication

Precondition	**Failed User Authentication**
Actions	Validate rejection of incorrect or invalid credentials.
Expected Result	System displays appropriate error message and prevents login.
Tested by	Khadija Rashid
Result	Verification Successful

Test Case 3

Title: Verify user signup functionality.

Precondition	**Signup Functionality**
Actions	Navigate to the signup page, enter valid email and password.
Expected Result	A new user account is successfully created and the user is directed to the login screen.
Tested by	Khadija Rashid
Result	Verification Successful

Test Case 4

Title: Verify invalid login

Precondition	**Invalid Login**
Actions	Invalid/incorrect credentials.
Expected Result	User login fails with appropriate error.
Tested by	Khadija Rashid
Result	Verification Successful

Test Case 5

Title: Test Image upload

Precondition	**Test Image Upload**
Actions	Upload a Leaf image from gallery
Expected Result	App display whether the plant is healthy or not.
Tested by	Khadija Rashid
Result	Verification Successful

Test Case 6

Title: Test Image capture via camera

Precondition	**Test Image Capture**
Actions	Take a picture of plant leaf.
Expected Result	System successfully captures the image and returns accurate disease result.
Tested by	Khadija Rashid
Result	Verification Successful

Test Case 7

Title: Verify disease detection

Precondition	**Verify Disease Detect**
Actions	Tap the disease detect button.
Expected Result	The system process the image and display whether the plant is healthy or not.
Tested by	Khadija Rashid
Result	Verification Successful

Test Case 8

Title: Show disease Symptoms

Precondition	**Disease Symptoms**
Actions	Tap the show symptoms button.
Expected Result	A list of symptoms related to the detect disease is displayed correctly.
Tested by	Khadija Rashid
Result	Verification Successful

Test Case 9

Title: show treatment

Precondition	**Show Treatment**
Actions	Tap the show treatment button.
Expected Result	A display the preventive and treatment suggestion for the detected disease.
Tested by	Khadija Rashid
Result	Verification Successful

Test Case 10

Title: Generate PDF report

Precondition	**Generate report**
Actions	Tap the generate PDF button.
Expected Result	A display the downloadable report that contains the image, confidence score, symptoms and recommendations.
Tested by	Khadija Rashid
Result	Verification Successful

2. UI/UX Detail

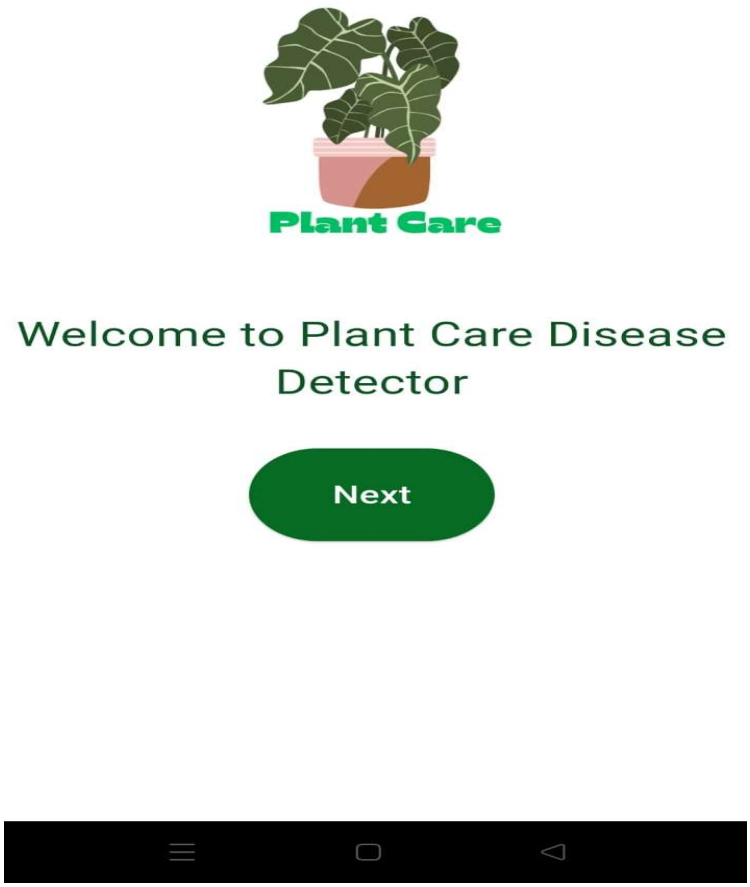


Figure 12-1 Application UI-1

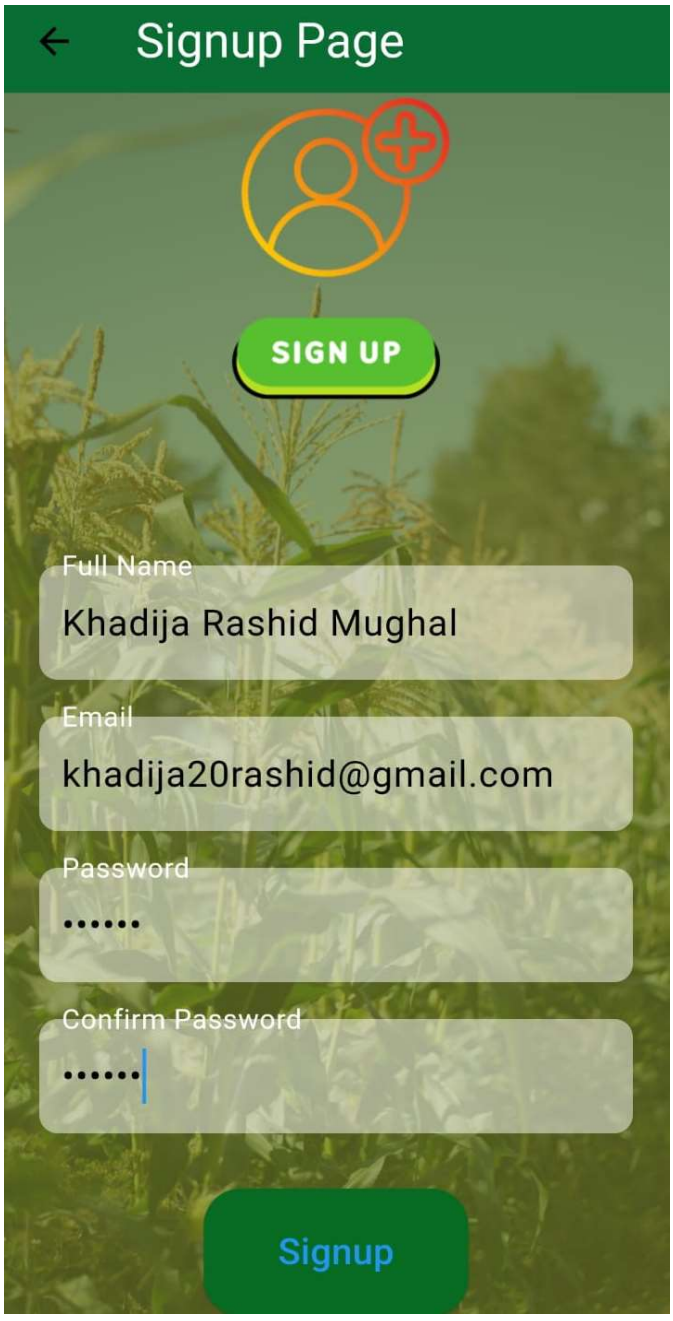


Figure 13-2 Application UI-2

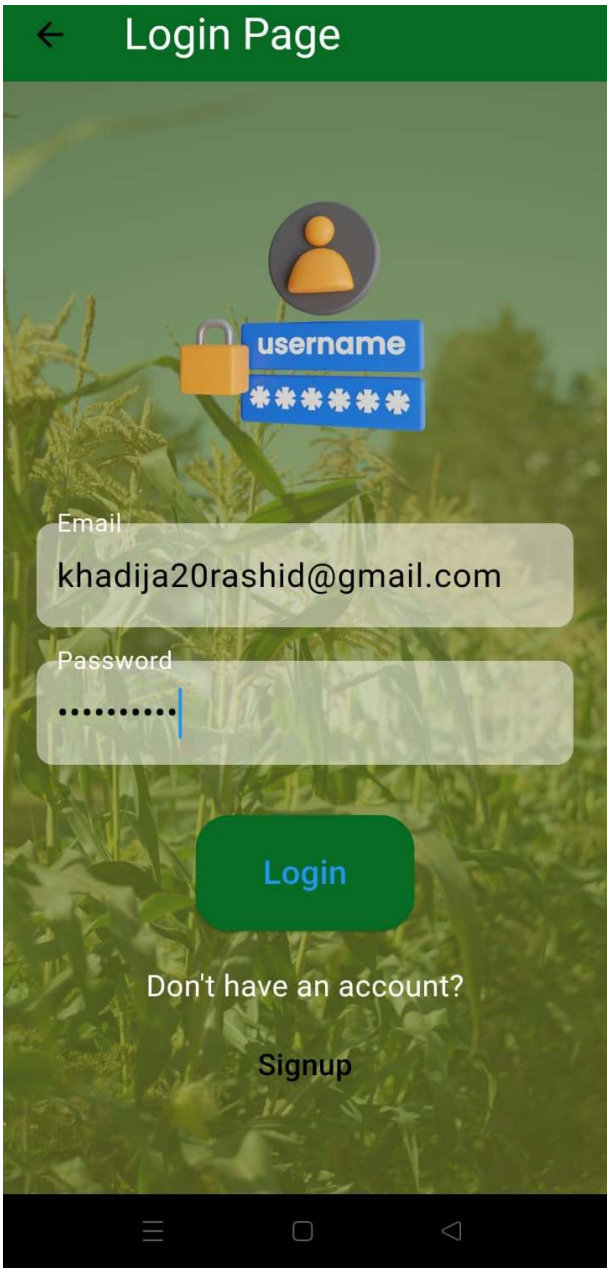


Figure 14-3 Application UI-3

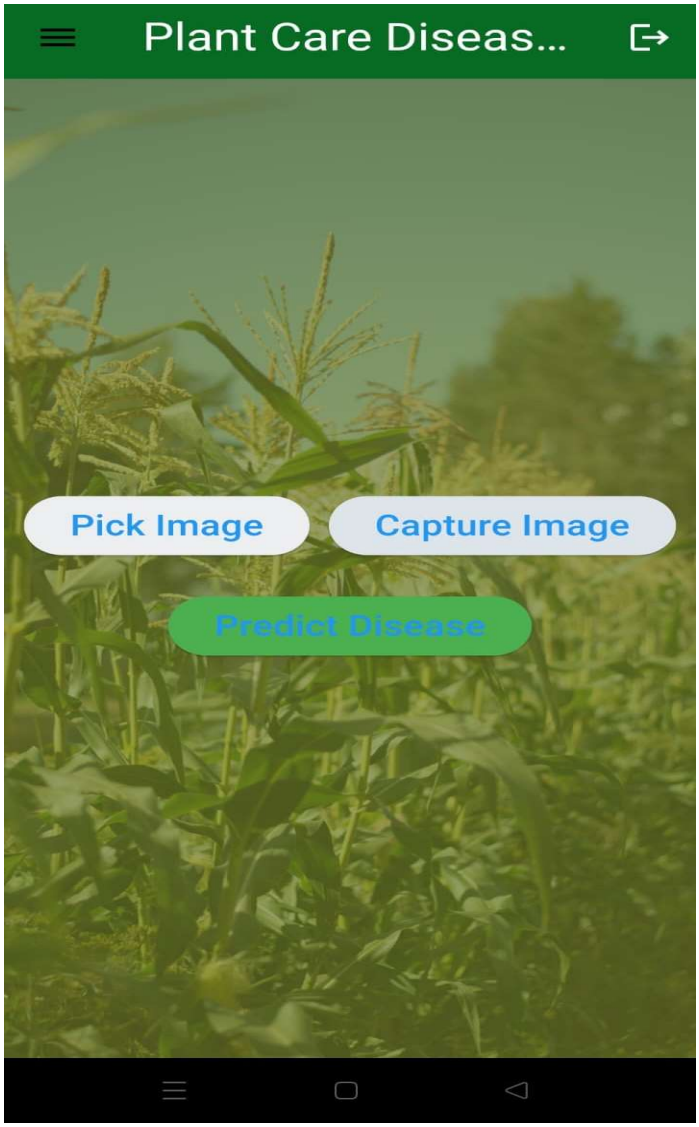


Figure 15-4 Application UI-4

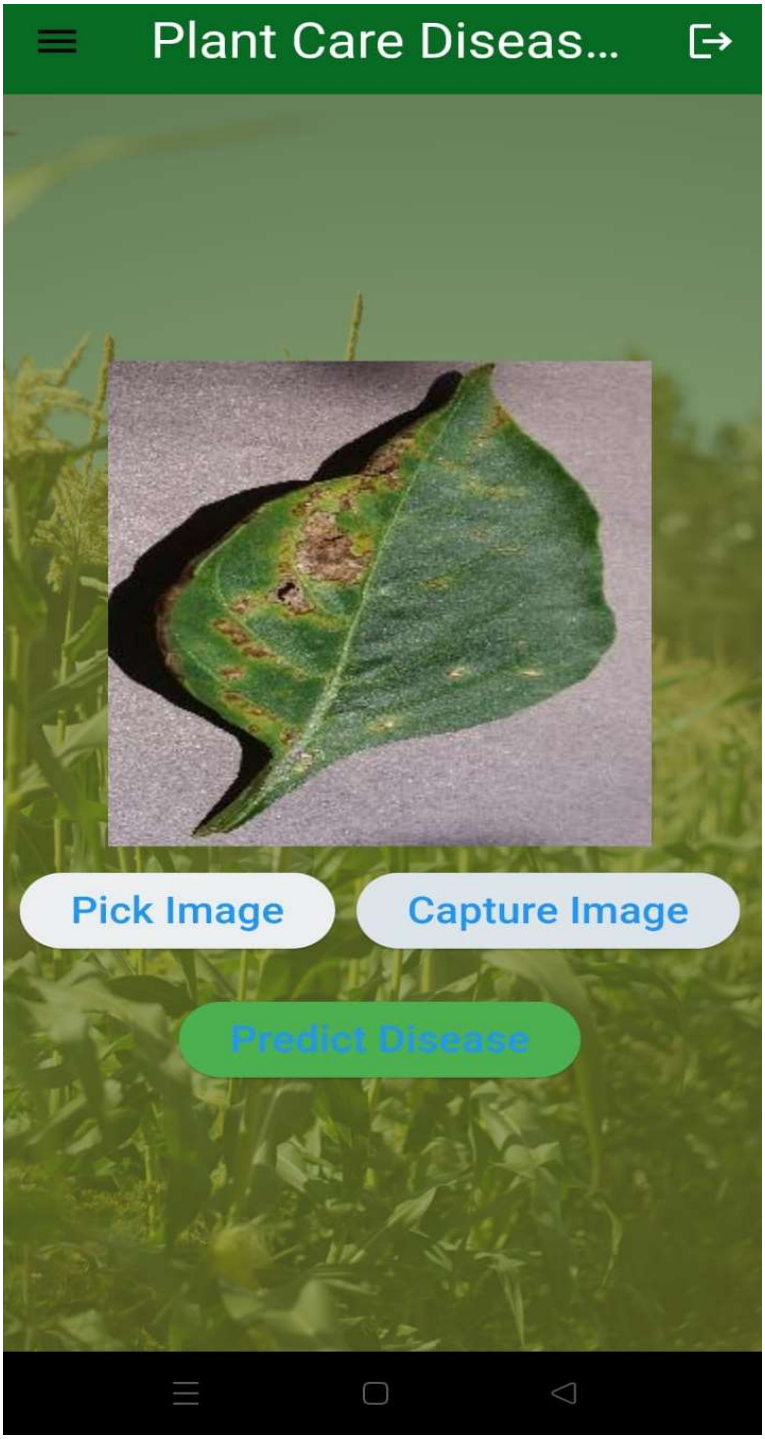


Figure 16 Application UI-5

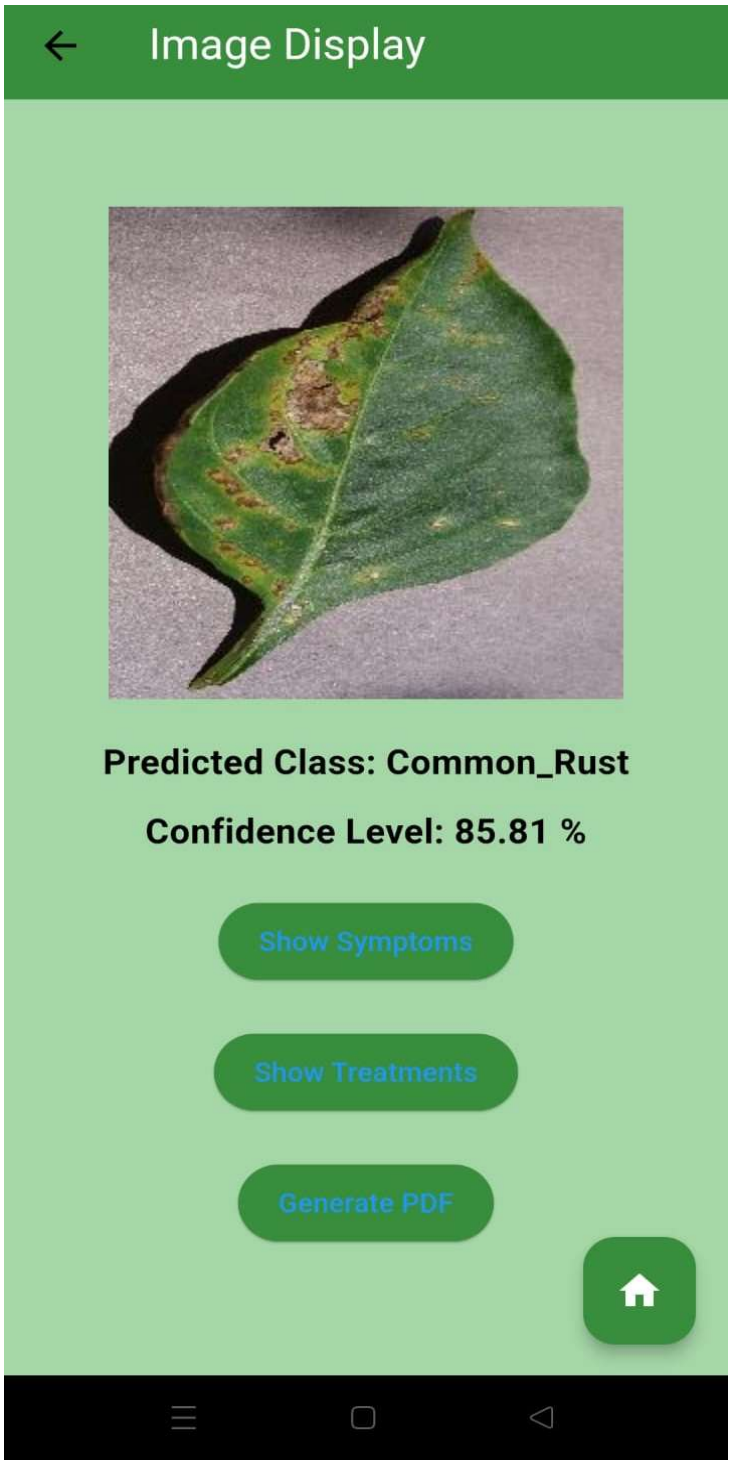


Figure 17 Application UI-6

Figure 18 Application UI-7

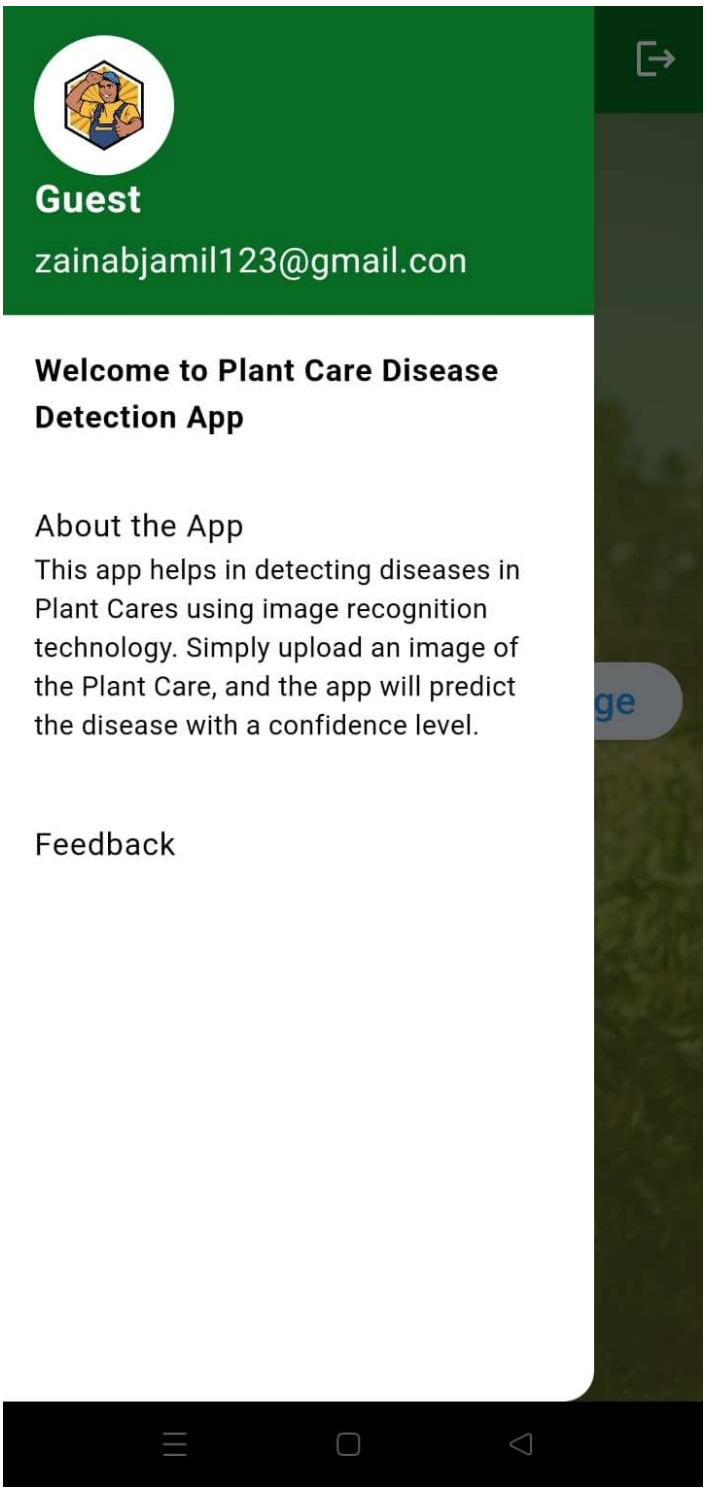


Figure 19 Application UI-8

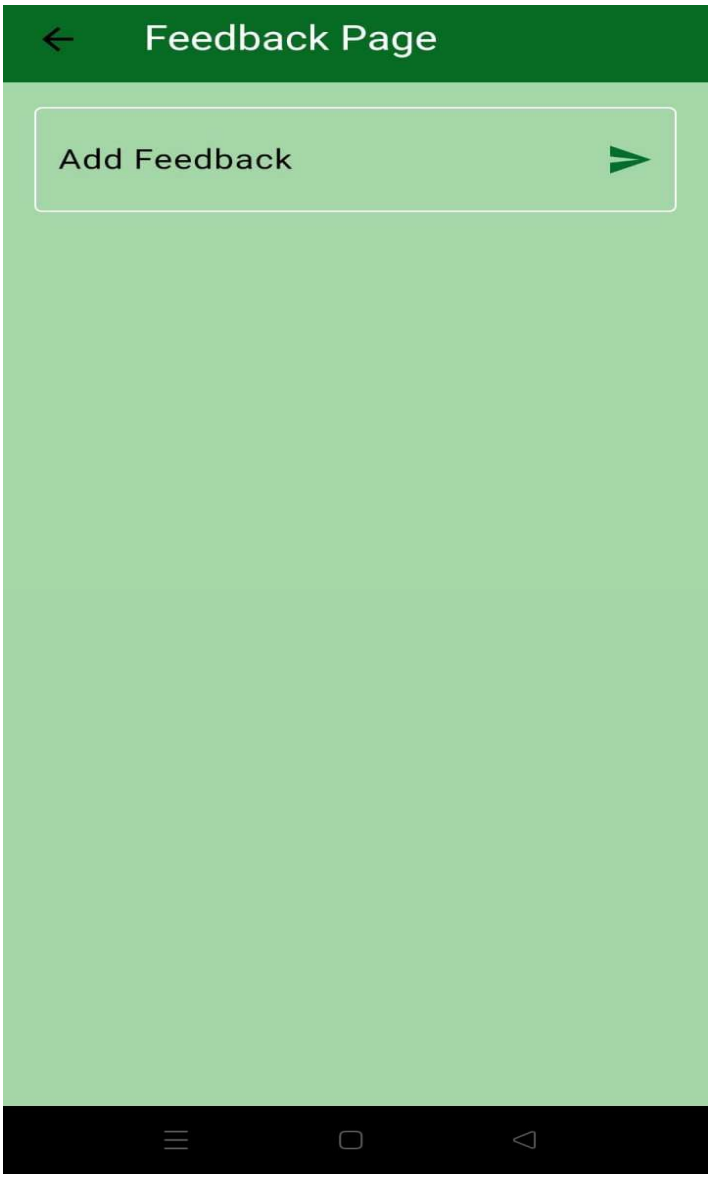


Figure 20 Application UI-9

3. Coding Standards

To maintain readability, consistency and long-term maintainability of the Plant Disease Detection, standard coding practices were strictly followed throughout the development process. These standards ensured that the source code was clean, well-organized and easily understandable by any developer or contributor.

- **Meaningful name:** Meaningful name should be used to clearly define the purpose.
- **Modularity:** Code is organized into reusable function instead of inline blocks.
- **Commenting:** Inline comments were added to explain complex logic, and function headers included docstrings to describe inputs, outputs, and purposes. This made the code base easier to understand and maintain.
- **Consistency:** consistent formatting, spacing and indentation used throughout.
- **Version Control:** GitHub was used for version control to manage the code changes and maintain the history of development progress.

APPENDIX E. FLYER & POSTER DESIGN

Figure 21 Flyer and Poster Design

APPENDIX F. COPY OF EVALUATION COMMENTS BY JURY FOR PROJECT – I

Satisfactory Work!
Students have demonstrated commendable progress in research and documentation. However, there is a need to develop a functional machine learning model to achieve a presentable and impactful outcome in FYP-II.
Satisfactory
1. Project Title should explicitly mention "Fruit Plant" instead of "Plant" only. 2. Literature Review needs to be extensive and include literature on fruit plants' diseases' detection techniques too. 3. Only Prototype exists. Work on GUI has not started yet.

APPENDIX G User Manual Document

1. User Manual Document

This user guide is for nursery workers and other general users to use this mobile application for plant disease detection. The application detects plant diseases via the leaves of the plant using image processing and deep learning. The application provides some information about the symptoms, treatment suggestions and a health report that can be downloaded.

2. Starting the Application

Once the app is installed, end user can:

- Find the app icon that is labeled "Plant Care Disease Detection" on their phone.
- Tap the app icon to open.
- The splash screen will display the app logo, then the home screen.

3. Using the Application – Step – by – Step Instruction

Step 1: Sign Up/Login

Once the app is open, two options displayed: Login and Signup

- New users will tap "Sign Up" to create an account by entering their name, email and password.
- Existing users will tap "Login" the app with their registered email and password.
- After logging in, the user is directed into the home screen.

Step 2: Upload or Capture an Image of the Leaf

Once you have logged in, the app will show the image input screen. You will see two buttons.

- Capture Image: Open your phone's camera to take an image of the affected plant leaf.
- Upload Image: Choose a leaf image that is already saved in your devices gallery.

Make sure the leaf image is:

- Well-lit and focused.
- Clearly shows symptoms.
- Not too distance and blurry.

Step 3: Disease Prediction

- After choosing the image, press the “predict disease” button.
- The app analyze the image with a trained algorithms.

After few seconds, the screen will show:

- The name of the predicted plant disease.
- A confidence score.
- A message if the image is bad or no disease is found.

Step 4: View Disease Symptoms

Once the disease is forecasted, click the button to check the symptoms.

This could be:

- Leaf spots and discoloration.
- Yellowing of leaves.
- Bacterial or fungal symptoms.

This helps users verify whether the forecast accurately reflects the state of their plant.

Step 5: Recommendation for Treatment

The app shows general treatment recommendations for managing the illness beneath the symptoms.

These suggestions might include:

- Adjustments for sunlight or watering.
- Suggest of pesticides or fungicides.

The instructions are straight forward and easy enough for nursery staff to understand.

Step 7: Create and Save a PDF Report

- A button labeled “Create PDF” appears on the screen.
- When you tap o it, the app creates a PDF health report with the following information:
- The leaf image that was uploaded and capture.
- The expected name of the illness.
- Symptoms that are visible.

- Suggestions for treatment.
- The user's device stores the PDF, which can be shared and or printed as needed.

4. Troubleshooting Tips

Here are some typical problems users might sun into when utilizing the app, along with easy fixes:

Unable to Upload an Image

If you can't upload or take a picture:

- Check tour phone settings to see if the app has permission to use the camera.

No Response After upload

After choosing or taking a picture, if the app doesn't react:

- Check your internet connection. In order to process the image and make the disease prediction, the app needs an active internet connection.
- Try uploading the picture again.

Login Issues

If you are experiencing difficulties logging in

- Verify you password and email address again.