AI Powered Mobile Application For Detecting and Diagnosing Crop Pests, Diseases and Nutrient Deficiencies

By

Wycliff Kimani Karanja

A Research Project Submitted To Zetech University for the Partial Fulfillment of The Award of Degree in Information Technology

March, 2025

DECLARATION

Student

I, Wycliff Kimani Karanja, declare that this project proposal is my original work and has not been presented for a diploma or degree in any other institution.

Signature: [illegible] **_ date: ____________**

Wycliff Kimani

Bachelor of Science in Information Technology

Supervisor

This research proposal has been submitted for examination with my approval as university supervisor.

Signature: ________________ Date: ____________

Miriam Kaara

School of ICT, Media and Engineering

DEDICATION

I dedicated this project to my beloved family for their unwavering support, encouragement, and belief in my abilities, even when I doubted myself, throughout my academic journey. Their sacrifices, prayers, support, and love were a constant source of strength and motivation. I also dedicated this work to the entire Zetech University fraternity whose continuous support, mentorship, and commitment to academic excellence played a vital role in shaping my skills and knowledge in the field of Information Technology and into this project.

ACKNOWLEDGEMENT

I first acknowledge God Almighty for His grace and strength throughout this project. My deepest gratitude goes to my father and my uncles Douglas Mwangi and Jacob Maina, and their families, for their unwavering support and the resources that made this project possible. Special thanks to my friend Festus for his technical contributions in ensuring system security and refining the user interface. I am grateful to AgriTech Analytics and PlantVillage for providing the essential dataset that enabled the training of the AI model central to this research. Finally, I thank Zetech University for providing the foundational knowledge, skills, and learning environment that equipped me to undertake this agricultural technology solution.

May God bless you all.

ABSTRACT

Agriculture contributes 33% of Kenya's GDP and employes over 75% of the rural population. Yet 20–40% of global food production is lost annually to crop pests and diseases, costing about $270 billion, a critical issue given rural Kenya's reliance on traditional pest diagnostic methods. This study aimed to improve smallholder farmers' decision making practices by developing AgroAI, a mobile app that uses machine learning and image recognition technology to diagnose crop pests and diseases via image recognition.

A convolutional neural network was trained on labeled image datasets from AgriTech Analytics and Kaggle PlantVillage, complemented by locally collected images. The AI model was implemented in PyTorch, TensorFlow and this was done using Google Colab, while the mobile interface used Kotlin with Jetpack Compose and Firebase. The back end communication was done and made possible by FastAPI.

The app was evaluated with 20 smallholder farmers in Kinangop Constituency, Nyandarua County. Testing showed high diagnostic accuracy, contributing to reduced input costs, enhanced crop health, improved crop yields and lower environmental impact. These findings supported recommending AI based diagnostic tools for Kenya's agriculture sector. Future work should focus on developing locally trained models tailored to regional crop varieties.

Contents

List of figures

List of Tables

Acronyms

AI - Artificial intelligence

ML - Machine learning

CNN - Convolutional Neural Networks

CV - Computer vision

API - Application Programming Interface

UI - User Interface

UX - User Experience

GPS - Global Positioning System

ICY - Information and Communication Technology

JSON - JavaScript Object Notation

XML - eXtensible Markup Language

HTTP - HyperText Transfer Protocol

IDE - Integrated Development Environment

SQL - Structured Querry Language

SDK - Software Development Kit

APK - Android Package Kit

IoT - Internet of Things

RAM - Random Access Memory

DB - Database

Definitions of Key terms

Smart AgroAI: This is an AI-powered mobile application developed to detect and diagnose crop pests, diseases, and nutrient deficiencies using machine learning and image recognition, to provide precise recommendations that are based on the data to improve yields, reduce crop production costs and reduce land degradation.

Image Recognition: This is a technology used to identify and reognize images. In thi case, this technology will be used

Firebase: A cloud-based platform providing real-time database capabilities, used in AgroAI to ensure that the user interface is communicating properly with the back end and the trained AI model.

Precision Farming: An agricultural approach using technology to optimize inputs usage and application which AgroAI supports by providing and offering data driven recommendations to the farnmer thereby reducing wastage and cost of production

User Acceptance Testing (UAT): This is an activity conducted by end-users to validate that the application meets their expectations and that it solves their problems

Unit Testing: This is a method or procedure where every part or component of the mobile application is tested individually to ensure that they perform as they are intended.

CHAPTER 1

INTRODUCTION

1.1 Introduction

Agriculture faces significant challenges from crop pests and diseases. These challenges threaten global food security and farmer livelihoods. These problems affect farmers around the world, they are particularly devastating for smallholder farmers in developing countries like Kenya, where traditional diagnostic methods often lead to crop losses and economic hardship. This reality, combined with personal experience growing up in a farming family in Nyandarua County, motivated the development of AgroAI - an AI-powered diagnostic tool designed to bridge the technology gap facing Kenyan farmers.

Global Agricultural Context

Agriculture serves as a cornerstone for global food security, employment, and economic stability, yet crop pests and diseases threaten productivity worldwide. The Food and Agricultural Organization estimates that 20 tp 40% of global food production is lost to pests and diseases annually, costing the global economy approximately $290 billion. According to Facini and Facini (2024), with the global population expected to reach 9.2 billion by 2040, these agricultural threats pose significant risks to food security in the long run, particularly in developing countries like Kenya where farmers lack access to professional agronomy services and advanced diagnostic technologies.

The Kenyan Agricultural Context

In Kenya, agriculture contributes 30% of GDP and employs 70% of the rural population, yet smallholder farmers lose up to 48% of their crop yields to pests and diseases. Many farmers in regions like Nyandarua County rely on traditional identification methods, leading to misdiagnosis, inappropriate use of fertilizers and pesticides, resource wastage, soil degradation, and environmental pollution. The lack of access to professional agronomy services forces farmers to depend on trial-and-error methods, resulting in financial strain and threatening both individual livelihoods and national food security.

The Technology Gap

The core problem lies in the gap between traditional farming methods and modern agricultural technologies. While developed countries increasingly utilize artificial intelligence and machine learning for crop management, these tools remain largely inaccessible to local farming communities in developing countries due to affordability issues and lack of localization. Addressing this gap is crucial for improving food security, maintaining soil health, protecting the environment, and ensuring economic stability for farming communities.

Personal Motivation and Research Rationale

The motivation for this study stems from personal experience growing up in Nyandarua County, where potato farming sustained a family of seven - good harvests meant meeting all needs, while poor yields resulted in financial hardship and children missing school. This struggle, shared by many Kenyan families, highlighted the urgent need for accessible, data-driven agricultural solutions and led to the development of AgroAI as an AI-powered mobile application designed to diagnose crop problems and provide tailored recommendations.

The AgroAI Solution

AgroAI represents a technological solution bridging the gap between traditional farming practices and modern agricultural diagnostics through machine learning and image recognition technology that detects and diagnoses crop pests and diseases in real time ad using localized collected datasets. The application provides farmers with immediate, location specific recommendations, democratizing access to agricultural expertise and enabling smallholder farmers to reduce production costs, increase yields, improve soil health, and achieve both food and financial security.

1.2. Statement of the Problem

Agriculture is the backbone of Kenya's economy, contributing 30% of the country's GDP and supporting millions of smallholder farmers and their families (KAAA, n.d). However, this vital sector faces a critical challenge: crop pests and diseases that destroy 40% of Kenya's crop yields annually (State of Food and Agriculture, 2001) and cause $290 billion in global economic losses (Gula, 2023).

The root of this problem is the lack of access to modern agricultural expertise or digital farming solutions by small holder farmers. Professional agronomy services are either unavailable in rural areas or too expensive for small scale farmers. This forces farmers to retain traditional methods of farming which are guesswork and trial and error methods when their crops fall sick. Without proper diagnosis, farmers cannot identify what is actually attacking their crops. They apply the wrong treatments, use incorrect fertilizers, or watch helplessly as their harvests fail. This leads to financial hardship for farming families, food insecurity, and significant economic losses for the country. The government must then import food and divert development funds to address these shortfalls. The tragedy is that many of these crop losses are preventable. Farmers simply need access to accurate, timely, and affordable crop diagnostics and get better localized recommendations.

This research proposed AgroAI, an AI powered mobile application that puts expert crop diagnosis directly into farmers' hands. By using artificial intelligence, machine learning and image recognition technology, AgroAI analyzes crop images and provide instant, accurate diagnoses along with localized treatment recommendations. This technology aimed to break the cycle of crop losses, improve food security, and empower farmers with the digital tools they need to succeed.

1.3. Objectives

1.3.1. General Objective

To develop an AI powered mobile application for detecting and diagnosing crop pests, diseases, and nutrient deficiencies, with the aim of providing data driven, localized recommendations to farmers.

1.3.2. Specific Objectives

1. To investigate the effectiveness of artificial intelligence and image recognition technologies in diagnosing crop pests, diseases and nutrient deficiencies.
2. To develop and train a machine learning model that can classify and diagnose crop health issues from agricultural images and offer the best recommendations based on the area of diagnosis.
3. To develop a digital, AI-powered android mobile application that provides smallholder farmers with accessible, real time diagnostics and localized data driven recommendations.
4. To evaluate the AgroAI's performance and assess how AI can be leveraged in agriculture to promote precision farming while reducing costs and losses, and improving crop yields

1.4. Research Questions

1. How effective are artificial intelligence and image recognition technologies in accurately diagnosing crop pests, diseases, and nutrient deficiencies compared to traditional diagnostic methods?
2. What machine learning approaches and training methodologies are most suitable for developing a crop diagnostic model that can classify agricultural problems from images and generate location-specific recommendations?
3. How can an AI-powered mobile application be designed to provide accessible, real-time crop diagnostics that address the affordability and accessibility challenges faced by smallholder farmers?
4. To what extent does the AgroAI system demonstrate accuracy, usability, and user acceptance among smallholder farmers in Nyandarua County, and how effectively can AI-powered diagnostics promote precision farming while reducing costs associated with crop misdiagnosis?

1.5. Justification

This research addressed critical challenges facing Kenyan smallholder farmers, who form the backbone of the country's agricultural sector. The development of an AI-powered crop pest and disease detection mobile application is justified by pressing issues impacting agricultural productivity, environmental sustainability, and food security.

Economic Impact

Plant pests and diseases reduce global crop yields by 20-40% annually, creating an economic burden of $290 billion worldwide (FAO, 2021; Gula, 2023). In Kenya specifically, total losses due to pests and diseases in maize and beans reach 57% and 42% respectively (Grisley, 2010). The Food and Agriculture Organization indicates that effective application of fertilizers and farm chemicals could increase farm produce by up to 78% (FAO, n.d.), highlighting the immense potential for improvement through data driven and technology interventions. AgroAI provide farmers with accurate, timely diagnostics that enable precise application of agricultural inputs, reducing farming costs, increasing yields, and enhancing food security and financial stability.

Environmental Benefits

Current agricultural practices contribute significantly to environmental degradation. Every 10kg of excess fertilizer emits 2.5kg of greenhouse gases, contributing approximately 9.8 tons of emissions annually (Environmental Implications of Excess Fertilizer and Manure on Water Quality, 2023). Excess fertilizers damage soil structure, reduce long-term fertility, and contaminate water sources through runoff containing high levels of nitrogen and phosphorus (AZoLifeSciences, 2022). By enabling precise application of agricultural inputs, AgroAI helps reduce unnecessary chemical use, mitigate land degradation, and decrease environmental pollution.

Social Impact

The research addressed the needs of 86% of Kenyan smallholder farmers who currently lack access to agronomy services and knowledge about their soil health and crop health. By democratizing agricultural expertise through mobile technology, the project addresses data driven farming decisions in a country where malnutrition affects one in five people, with expectant mothers and children being most vulnerable (Okube et al., 2022).

Technological Innovation

This research contributed to agricultural technology by applying artificial intelligence, computer vision, and machine learning to create practical solutions for smallholder farmers. Unlike existing agricultural technologies requiring extensive technical knowledge, AgroAI is designed for accessibility on basic smartphones, bridging the gap between technology and farming in agricultural innovation. The application provide accurate, timely detection and diagnosis of crop pests, diseases, and nutrient deficiencies, offering a scalable model that can be adapted to other regions facing similar challenges while ensuring economic stability for farming communities and for the national economy.

1.6 Significance of the Study

This study held significant implications for various stakeholders and contributed to broader development goals.

Smallholder Farmers
The system empowered farmers with decision making capabilities, reducing reliance on traditional farming methods, guesswork and trial and error methods and improving crop management efficiency. AgroAI has the potential to transform subsistence farming into more productive and sustainable farming practices.

Agricultural Sector Development
The research supported Kenya's agricultural adoption of technology agenda by demonstrating how artificial intelligence could be effectively integrated into traditional farming systems to increase the yields of farming. The findings provide insights and knowledge of digital agriculture adoption strategies and offer a scalable framework for similar technologies across Kenyan farming communities and the globe at large.

Environmental Conservation
By promoting precision agriculture, the project advanced sustainable farming practices that minimized chemical inputs and reduced environmental degradation. It

contributed to global climate action and supported the achievement of several Sustainable Development Goals, including SDG 2 which is Zero Hunger, SDG 13 Climate Action, and SDG 15 Life on Land by reducing land degradation and environmental pollution.

Academic Knowledge

The study advanced the field of agricultural technology by providing empirical evidence of AI application effectiveness in smallholder farming contexts. Its methodology and outcomes enriches and contributes to the growing body of research on digital agriculture in developing countries and lays a foundation and hold a part for future innovation.

1.7. Research Methodology

This research adopted the mixed methods approach, where it combined both qualitative and quantitative data collection and analysis techniques, to evaluate the performance, usability and impact of AgroAI application. This method is to ensure that the system results are captured for analysis and that the users' experiences are recorded and documented for review and system evaluation. For the system development methodology, this project will adopt the Agile Software Development Lifecycle. The reason SDLC is well suited for this project it is a flexible, iterative approach to software development and its emphasis on collaboration, adaptability and customer feedback. (Inquire)

1.8. Scope

This research focused on the development and evaluation of AgroAI, a mobile based android application designed to detect and diagnose crop pests, diseases and nutrient deficiencies through machine learning and image recognition technology. The project covered the design, development and training of a machine learning model using agricultural images, the integration of the trained model into an android mobile application, and the deployment of the mobile application for use and for testing among selected smallholder farmers in Kinangop constituency, Nyandarua county.

The scope of this project included system development, deployment, user testing and system evaluation to determine if the system met the desired objective. The study specifically determined and evaluated the application’s performance in terms of efficiency, accuracy, ease of usability, user acceptance and its impacts in the real world of farming and decision making. It also included the development of a user friendly mobile interface that allow users to upload images from their gallery or take new images of their crops with their mobile phones and receive real time diagnosis of the crop issue and thereafter receive a recommendation tailored for that diagnosis.

However, this study did not cover commercializing, long-term field test, or the integration with other platforms such as e-commerce, remote sensing, or community interaction section. Additionally, while the model was trained on multiple crops grown in Kenya, the testing was done on a select number of crops commonly grown in Kinangop constituency, Nyandarua County due to time and resources constraint.

Geographically, the study will be based in Kinangop Constituency in Nyandarua County, which will act as a representative of other farming regions. The decision to choose Kinangop was based on the over reliance on small-scale farming by the residents of that area. The results, feedback and recommendations may inform of future more research, expansion into other agricultural regions and integration of other services like weather forecasting.

1.9 Limitations of the Study

Although the AgroAI system was successfully developed and deployed, the study encountered several limitations. One major challenge was the limited diversity in the training dataset. While images were sourced from AgriTech Analytics, Kaggle, and field environments, certain crops and diseases were under presented and lacked enough images for training, which may have reduced diagnostic model accuracy.

Another significant limitation was the hardware constraints faced by some smallholder farmers. Many of the target users owned basic mobile phones that lacked the processing power or camera quality required for optimal system performance. In addition, limited or unstable internet connectivity in rural areas affected features that relied on cloud support, such as model updates and remote recommendations.

The research was also affected by a constrained timeline, which limited the duration and depth of field testing. This made it difficult to gather long-term feedback and monitor seasonal crop changes. Furthermore, language and literacy barriers posed a challenge, as some farmers struggled to interpret the application's feedback, especially when it was presented in English without local language support.

Constrained funding also posed a significant challenge during the development, deployment, and testing phases of the AgroAI system. The limited financial resources affected various aspects of the study, including data collection, system testing, and farmer outreach. This restricted the geographic scope of the project to a single location in Kinangop Constituency in Nyandarua County, despite the system's intended application in broader agricultural contexts across Kenya. Systems used for the model development training and development was also constrained to free or cheap features.

As a result, the diversity of crops tested, the number of participating farmers, and the duration of field trials were all constrained. The project team was unable to conduct extended, many seasonal trials or include other farming farming regions with varying climates and crop profiles. This limitation may have impacted the scope of the findings and delayed potential system enhancements based on varied user feedback. Nonetheless, Kinangop served as a valuable pilot environment due to its strong reliance on smallholder farming, providing relevant insights into the system's practical utility.

Despite these limitations, the project provided key insights into the practical deployment of AI-powered tools in agriculture, and highlighted areas for future improvement in similar interventions.

CHAPTER 2
LITERATURE REVIEW

2.1 Introduction

This chapter presents a literature review of AI applications in agriculture, digital diagnostic tools, image recognition technology, and machine learning implementation for farming decisions. The review examined existing research, theories, and frameworks to identify knowledge gaps and help in the development of AgroAI. The literature review analyzed current AI and machine learning technologies in agriculture, focusing on their effectiveness in addressing farming challenges. The study examined implementation challenges and user adoption patterns of existing agricultural AI systems, while also evaluating technological limitations and accuracy issues that have hindered widespread deployment of such solutions. The review further explored successful case studies and deployment strategies from similar projects worldwide, assessing the factors that influence farmer acceptance of digital diagnostic tools in rural settings. Through this comprehensive analysis, the chapter establishes the foundation for developing localized, AI powered crop diagnostic tool specifically suited for developing countries like Kenya.

2.2. Theoretical Review

This section examines the theoretical foundations that underpin the development of AI powered digital diagnostic tools for agriculture. This study was guided by two theoretical models which are the Technology Acceptance Model, TAM by Davis (1989) and the Unified Theory of Acceptance and Use of Technology, UTAUT by Venkatesh et al. (2003). These models helped explain how farmers perceive, adopt, and interact with digital agricultural technologies. They form the basis for understanding user behavior in relation to AgroAI, particularly in terms of perceived usefulness, ease of use, social influence, and effort expectancy. The study was grounded in established frameworks that provided understanding of how AI systems can deliver real time data driven diagnostic assistance and support farmers in decision making processes in farming contexts. The theoretical review explored how these technologies perform in real life and examined patterns, rate and level of farmer adoption and acceptance. These theoretical foundations provided guidance on both system design principles and user behavior patterns related to technology adoption in agricultural settings. The frameworks also informed the localization strategies needed to ensure that the AI model provide relevant, data driven localized recommendations suited to specific regional contexts. Understanding these theoretical reviews ensured that AgroAI's development aligns with real world farming challenges faced by farmers and addresses actual user needs rather than theory assumptions and understanding.

The integration of TAM and UTAUT into this study ensured that AgroAI’s design considered both the behavior and technology factors that affect user adoption and system success among smallholder farmers

2.2.1. Artificial Intelligence and Machine Learning in the Field of Agriculture

The use of artificial intelligence in farming has been on the rise in the recent years, with machine learning being effective and being used in checking and determining farming problems and helping in making farming decisions. Deep learning machine learning models, like Convolutional Neural Networks, have shown very high accuracy and effectiveness in detecting and diagnosing plant issues. According to Mohanty, Hughes, & Salathe (2016), their trained CNN model have achieved an accuracy of 99.35% on a held out test. The model was trained using the PlantVillage dataset. This shows the potential and effectiveness of training AI models to enable smart phones, which have been adopted in many parts of the continent, to join in this fight against crop pests and diseases, by being used as a digital farming tool by small holder farmers who cannot access or afford the cost of hiring government and private agronomists and their services. Research by Saleem et al.(2019) created a deep learning model for detecting tomato plant diseases that reached 95.65% accuracy. In the same similar studies, Too et al(2019) and Ranjaran et al.(2018) have shown that convolutional neural networks based models can successfully tell the difference between healthy and diseased crops, and also classify the pest, disease or nutrient defficiency.

How well the machine learning trained model will work, how accurate will it be and how effective it will be to farmers depends on the good quality and diverse training data that you have.Hughes and Salathe (2015), created PlantVillage, a dataset that is easily accessed through Kaggle, one of the biggest publicly available collections of plant images, which has become a standard for crop disease detection research. Kamilaris & Prenafeta-Boldu, 2018; Barbedo, (2018) represents several studies that points out the problem of the dataset being biased and indicating the need for more localized data for training purposes. Research done in African setting, for example research by Ramcharan et al. (2017) on cassava disease detection points out the need to train models and to use datasets localized for the intended regions to get the best results in real world use.

Also, the possibility of integrating trained artificial Intelligence models into smart phones present a scalable and cheaper solution for disease diagnosis, especially to growing countries where smallholder farmers have limited or no access to agronomy services. Research by Mohanty et al. (2016) demonstrated that while training and testing of ML models require high performance computing, the final model and prediction process can run in under a second in a basic normal smartphone central processing unit. This makes use of smartphones for disease detection not only possible but easier and cheaper for smallholder farmers.

2.2.2. Mobile Technology Adoption in Agriculture

The framework for understanding how farmers accept new technology is well established through several various models, with the Technology Acceptance Model, being particularly useful for mobile farming apps. Davis (1989) suggested that how useful farmers think a technology is and how easy it is to use are the main factors that determine whether they will adopt it. In farming contexts, research by Munyua et al. (2009) and Lwoga et al. (2011) has shown that these factors strongly influence farming communities willingness to use and adapt to mobile based solutions for their farming practices and to solve their crop problems.

Studies focusing on mobile farming apps in developing countries show that smartphone use has reached levels where agricultural innovation is possible. Research by Aker (2011) and Qiang et al. (2012) shows that mobile technologies can effectively help solve information problems in rural farming communities. In Kenya specifically, studies by Mwombe et al. (2014) and Kikulwe et al. (2014) have shown that farmers are increasingly open to mobile based farming services, especially when these services address specific, practical farming problems that affect them and their produce. The success of platforms like iCow in Kenya, as documented by Baumuller (2018), shows the potential for mobile apps to deliver farming advice to smallholder farmers and reduce the gap between technology and farming.

2.2.3 User Acceptance and System Usability

How well farmers accept farming technologies depends on multiple factors beyond just usefulness and ease of use. Research by Venkatesh et al. (2003) expanded the TAM through the Unified Theory of Acceptance and Use of Technology (UTAUT), which includes social influence, available support, and how much effort is needed to use the technology. In farming specifically, studies by Michels et al. (2019) and Kernecker et al. (2020) have used UTAUT to understand precision agriculture adoption, finding that thought benefits, ease of use, ease of understanding, and social influence strongly affect adoption decisions by the target people or market.

System usability in farming contexts needs special consideration for the target users. Research by Sein and Harindranath (2004) emphasizes how important it is to design technologies that fit with existing farming practices and knowledge systems. Studies focusing on smallholder farmers, such as studies by Ospina and Heeks (2010) and Gakuru et al. (2009), show amd encourage the need for easy to use and understanding interfaces that work for people with different levels of digital skills and education. They encourage interaction and involvement of intended target at every stage of development. The work of Mittal and Mehar (2016) specifically addresses design considerations for mobile apps targeting Indian farmers, providing insights relevant to similar contexts in Kenya. These concepts If used properly and adopted in making similar mobile applications in Kenya, it could work to help fight the problem of crop pests, diseases and nutrient deficiencies.

2.2.4 Diagnostic Accuracy and System Performance

The performance of AI powered crop diagnostic systems is usually measured through accuracy, precision, recall, and yield accrued metrics. Research by Barbedo (2018) provides a detailed review of performance measures in plant disease detection, noting that real world performance often differs from laboratory conditions. Studies by Arsenovic et al. (2019) and Atila et al. (2021) have shown that combining methods and transfer learning can improve diagnostic accuracy, especially when dealing with limited training data.

The challenge of keeping diagnostic accuracy consistent across different environmental conditions and crop varieties has been addressed in several studies. Research by Barbedo (2019) highlights how important dataset diversity is and the need for continuous model updating to maintain performance. Studies by Kamilaris and Prenafeta-Boldú (2018) and Liakos et al. (2018) emphasize that successful use of AI systems in agriculture requires careful consideration of local conditions, crop varieties, the culture and social economy of that region, and disease patterns specific to the target region.

2.2.5 Impact on Farming Decision Making and Productivity

The potential impact of AI and computer vision powered diagnostic tools on farming productivity has been demonstrated in several empirical studies. Research by Basso and Antle (2020) shows that precision agriculture technologies, including diagnostic tools, can increase yields while reducing input costs spent in production. Studies by Finger et al. (2019) and Klerkx et al. (2019) have examined how digital technologies influence farming decision making processes, finding that timely, accurate information significantly improves resource allocation and crop management decisions leading to increased crop yields. This is because it gives recommendations in time for treatment or prevention of further loss or damage.

In the context of smallholder farming, research by Aker (2011) and Jensen (2007) demonstrates that access to timely information can reduce market inefficiencies and improve farmer incomes. This is because the losses of yields to farmers are reduced extremely, increasing their yields and their financial security, and meeting the market demand for food supply. Studies specific to crop health management, such as studies by Oerke (2006) and Chakraborty and Newton (2011), show that early detection and accurate diagnosis of crop problems can significantly reduce yield losses and input costs. Land degradation and reduced environmental pollution are also some of the problems solved by early, timely and data driven detection and diagnosis. The work of Jain et al. (2018) in Indian agricultural contexts provides evidence that mobile based diagnostic tools like web applications and mobile applications can improve both the timeliness and accuracy of crop health interventions, the crop yields and the crop yield loss caused by various crop pests and diseases.

2.3. Conceptual Framework

Based on the theoretical foundations discussed above, this study developed a comprehensive conceptual framework that illustrates both the theoretical relationships affecting AgroAI adoption and the technical system architecture that delivers diagnostic recommendations and other services to farmers. The framework integrates technology acceptance theories, TAM and UTAUT, with artificial intelligence and machine learning effectiveness principles to provide a complete understanding of how AgroAI functions in the smallholder farming context. The conceptual framework (Figure 2.1) demonstrates the pathway from farmer characteristics and system features to technology acceptance, actual system usage, and resulting agricultural improvements. It also highlights moderating factors such as social influence, access to technical support, and regional conditions, which may affect adoption and performance.

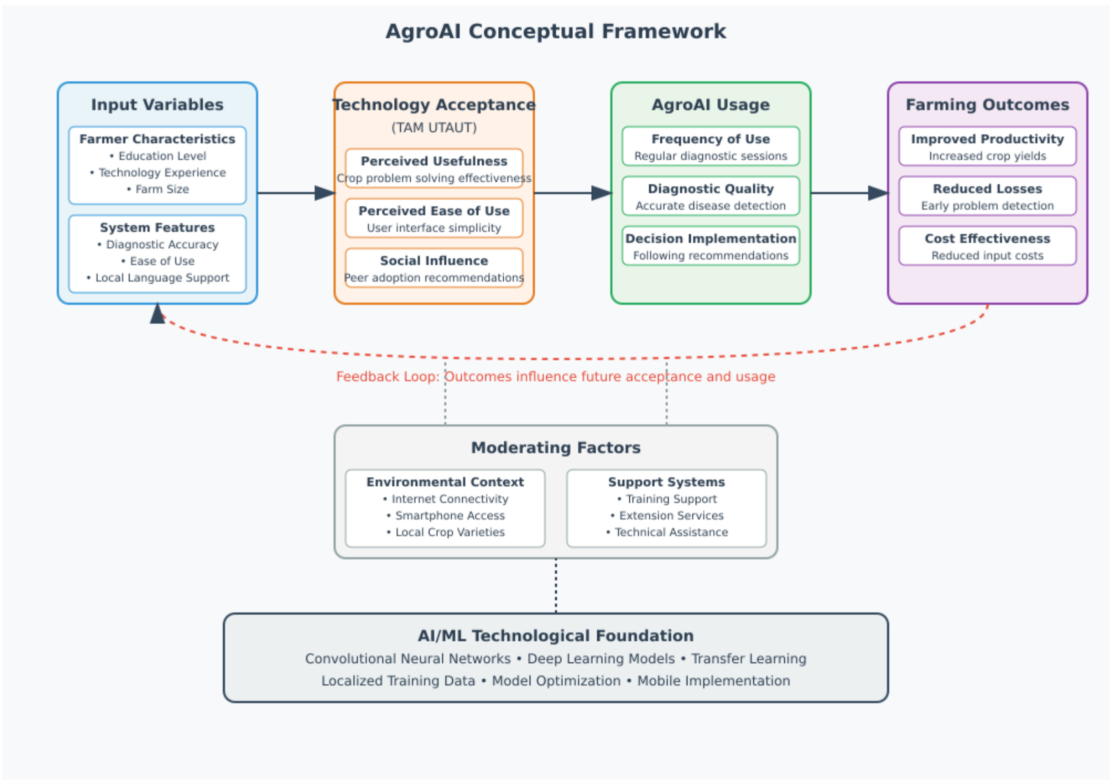


Fig 2.1: AgroAI Conceptual Framework

2.3.1 Theoretical Conceptual Framework

The theoretical framework as shown in Figure 2.1 shows how farmer characteristics and system features influence technology acceptance through perceived usefulness, ease of use, and social influence. This acceptance process determines usage patterns, which ultimately lead to improved farming outcomes. The framework incorporates moderating factors such as environmental context and support systems that can either facilitate or hinder system effectiveness. The framework identifies three key relationships grounded in technology acceptance theory which are:

Input Variables to Technology Acceptance

Farmer characteristics such as education level, technology experience, farm size, among others and system features like accuracy of the AgroAI model, ease of use, local language support, among others, directly influence how farmers perceive and accept the AgroAI system. This relationship is supported by TAM and UTAUT theories, which emphasize that perceived usefulness and ease of use are primary determinants of technology adoption.

Technology Acceptance to System Usage

Once farmers accept the technology, their usage patterns are characterized by frequency of use, diagnostic quality expectations, and willingness to implement system recommendations. This relationship demonstrates how theoretical acceptance translates into practical engagement with the AI diagnostic system.

System Usage to Farming Outcomes

Effective system usage leads to improved productivity, reduced crop losses, and cost effectiveness. These outcomes create a feedback loop that reinforces continued technology acceptance and usage, supporting the sustainability of AgroAI implementation.

This framework guides the AgroAI project in aligning its technical development with user needs and behaviors. By emphasizing the relationship between user-centered design and successful implementation, the framework ensures that AgroAI addresses both the technical and human challenges involved in deploying AI-powered agricultural tools in rural contexts like Kenya.

2.4 Critique of Existing Literature

Several AI-powered diagnostic tools, like Plantix and PlantVillage Nuru, have made significant progress in helping farmers manage crop health. However, these tools also have key limitations, particularly in the Kenyan context. Plantix, for instance, supports over 500 plant diseases worldwide, but it relies heavily on internet connectivity, which limits its use in rural areas without internet access. It often pushes agrochemical solutions and has limited support for local crops and languages. Usability issues, such as crashes and complicated interfaces, further decrease its effectiveness for smallholder farmers with basic digital skills.

PlantVillage Nuru was developed to identify diseases in cassava and maize, including CMD, CBSD, and fall armyworm. It works well within its narrow focus and offers some offline features and voice support, making it more accessible than some other options. However, it is limited in the variety of crops it covers, does not adapt well to different regions, and has been adopted slowly because of usability and awareness issues.

In contrast, AgroAI focuses on localization and simplicity. It is built around filtered, relevant data and aims to provide crop-specific diagnostics for smallholder farmers. Although it is still in the early stages and lacks features like multilingual or offline support, it represents an effort to fill the gaps left by existing tools, particularly regarding local relevance, integrating farmer feedback, and preparing for future offline use.

These critiques show the need for ongoing innovation that balances technical capabilities with local importance, usability, and accessibility for under resourced farming communities.

2.4.1 Limited Focus on Developing Countries

Most of the research done on the area of using and adopting artificial intelligence, machine learning, image recognition technology and other developing countries, it comes from developed countries with different farming conditions and different crop types altogether. In cases where the crop types are the same, you find that the diseases are not entirely the same that affect farmers in developing countries, be they mutated or that the proposed solutions are not available to farmers in developing countries. Studes ny Chen et al. (2021), and Rodriguez and Silva (2020) mainly looked and concentrated on large commercial farms with high technology equipment. Such studies do not consider the challenges being faced by smallholder farmers in Kenya who have limited resources, machinery and technology, and varies in crop types grown. This gap shows that most of these AI solutions and technologies may not work well in Kenyan farming conditions.

2.4.2 Lack of User-Friendly Solutions and UI for less educated farmers.

Many mobile applications and web applications that have been developed already for crop diagnosis are too complex for less educated farmers to comprehend and use.

While Okonkwo et al. (2020) developed a mobile diagnostics tool, their study and feedback reports from farmers and the feedback section in Google Play Store suggested that farmers with limited education levels had a hard time using this tool and often went back to their traditional, loss-accruing methods of farming. Research also showed that most agricultural apps use technical languages that farmers don’t understand (Martinez & Brown, 2019). This creates a barrier for majority of farmers who don’t have advanced knowledge f how to use these tools, who are the majority in growing countries like Kenya.

2.4.3 Incomplete Diagnostics Coverage

Current research and developed tools show or focus on either pest detection or disease detection and neither combines crop pest detection, diseases detection, nutrient deficiencies and offer tailored and localized recommendations in one diagnostics tool. Kumar and Patel (2021) worked on pest detection, while Johnson et al. (2020) focused on plant diseases. This means that farmers need to acquire multiple tools to be well equipped to handle their crops problems. This is not efficient or practical to a farmer who is resource limited. No single tool exists that addresses all the three major crop problems at the same time.

2.4.4 Limited Real World Testing and Validation

Many studies test their AI models in controlled laboratory conditions rather than real farm environments. While Li et al. (2021) achieved 95% accuracy in lab tests, field testing showed much lower accuracy rates. This gap between laboratory results and real-world performance is a major weakness in current research. You might find that images used for training require to go through cleaning processes like background removal and real world testing is advised for maximum efficiency and accuracy testing. Most studies also test with small sample sizes and do not validate their results across different geographical locations. When these tools are adopted and used in other unlocalized regions, the results are not accurate and may result to more crop yields losses.

2.4.5 Inadequate Focus on User Acceptance and Training

The literature review shows limited research on how to train farmers to use AI diagnostic tools effectively. Though Davis et al. (2019) mentioned the importance of user training, most studies do not provide detailed frameworks for farmer education and support. This oversight means that even good diagnostic tools may fail because farmers do not know how to use them properly.

2.4.6 AgroAI's Contribution in Addressing Identified Gaps

The proposed AgroAI mobile application is specifically designed to address the limitations identified in previous literature. First, it focused on Kenyan farming contexts by training its AI model using locally sourced crop images mostly from AgriTech Analytics, a Kenyan company with offices in Spur Mall and Ngara, Nairobi County and PlantVillage dataset from Kaggle, and addressing region specific pests, diseases, and nutrient deficiencies. The app has a user friendly interface with simple language and identifiable icons and UI making it accessible and easy to use by farmers with varying levels of education and digital literacy and understanding.

Unlike most existing tools that are limited to either pest or disease detection, AgroAI provides a comprehensive diagnostic system that combines pest, disease, and nutrient deficiency identification in one platform. It integrates traditional farming knowledge with modern AI technology to enhance local relevance and acceptance. Localized recommendations are then offered to the farmer based on the crop problem diagnosed. Field testing was done in real farm environments across Kinangop Constituency and Ruiru, Kiambu County to validate performance of the AgroAI system in real world conditions. All recommendations are presented in simple, easy language to help farmers make timely, informed decisions.

These design considerations make AgroAI not only technically innovative but also practically useful as it addresses real needs of smallholder farmers in Kenya and contributing meaningful advancements to agricultural AI research.

2.5. Summary

The literature reviewed in this chapter reveals significant developments in the application of artificial intelligence, machine learning, mobile technologies and image recognition technologies in agriculture. Numerous studies have shown the effectiveness of machine learning techniques, particularly convolutional neural networks, in identifying crop pests and diseases from images with high levels of accuracy. Additionally, theoretical models such as the Technology Acceptance Model and Unified Theory of Acceptance and Use of Technology have provided useful frameworks for understanding farmer adoption of digital farming tools.

Despite these advances, the review also shows considerable limitations and gaps in existing systems. Many of the reviewed solutions are developed in high-resource settings and are not adapted to the real world challenges faced by smallholder farmers in developing countries. These include limited digital literacy, inconsistent internet access, and the lack of localized content. Furthermore, most systems focus narrowly on either pest or disease detection and often neglect nutrient deficiency identification, usability, or field testing.

This review shows the need for AI driven agricultural systems that are not only technically robust but also context-aware, user-friendly, and inclusive. These results provide the foundation for the development of AgroAI, a system designed to address the identified gaps by offering a comprehensive, accessible, and locally adapted mobile solution for diagnosing crop pests, diseases, and nutrient deficiencies among Kenyan smallholder farmers.

Although AgroAI system did not solve all the identified gaps, most of the gaps were addressed and solved.

2.6. Research Gaps

Based on the literature reviewed, the following key research gaps have been identified and form the basis for this study:

1. **Limited focus on smallholder farmers in developing countries**
This is where you find that most AI based crop diagnostic solutions and tools are designed for high tech environments ad countries and are not adapted to the realities of small-scale farmers in Kenya.
2. **Lack of multiple purpose diagnostic tools**
This is where you find that existing applications and systems typically address either pests or diseases, with very few offering integrated diagnosis of pests, diseases, and nutrient deficiencies.
3. **Low consideration for user literacy and local language support**
Many systems are not user friendly for farmers who are less educated, and few offer local language interfaces.
4. **Weak and inadequate real world testing and validation of the tools developed like AI models**
A significant portion of prior research has focused on lab-based model testing, with limited deployment in actual farm environments. When these tools are deployed to the real world, their accuracy reduce and their intended effectiveness is affected.
5. **Inadequate farmer training and on boarding frameworks**
Few studies provide structured methods for helping farmers understand and use AI powered tools effectively. Farmers are not able to use these tools since they haven't been trained or taught how to.
6. **Poor alignment with local farming practices and connectivity challenges**
Many tools assume constant internet access and Western farming norms, which are not realistic in rural Kenyan contexts. Internet connection is not guaranteed here.

CHAPTER 3
SYSTEM METHODOLOGY

3.1 Introduction

This chapter outlines the methodology that was used to design, develop, and evaluate AgroAI which is a mobile application that uses and leverages artificial intelligence, machine learning and image recognition technology to detect and diagnose crop pests, diseases, and nutrient deficiencies. The methodology covers both the technical development process and the research approach used to test the system with smallholder farmers in Kinangop Constituency, Nyandarua County. This chapter presents the research design, data collection methods, sampling techniques, system development methodology, and validation strategies used to ensure the application was accurate, context-aware, and user-friendly for its intended audience. This chapter also aims to present the process in a clear and structured way so that the research can be understood, evaluated, and used as a basis for future studies seeking to improve digital tools for smallholder farmers.

3.2. Research Design and Approach

This study adopted a mixed methods approach, combining both qualitative and quantitative techniques to evaluate the AgroAI system's usability, diagnostic accuracy, and real-world effectiveness in smallholder farming contexts. The research approach integrated both descriptive and exploratory components. The descriptive elements focused on capturing measurable aspects such as system accuracy and user interaction patterns, while the exploratory elements aimed to understand farmer perceptions, barriers to technology adoption, and suggestions for improvement.

In terms of system development, the AgroAI mobile application was built using the Agile Software Development Life cycle. This approach was selected to allow for iterative development and continuous improvement based on user feedback from previous testing application. Each development sprint involved testing and evaluating a specific set of features, which were refined according to feedback received during field trials. This made it possible to adapt the system, ensuring that both technical and usability issues were addressed in real time.

By combining a mixed methods research strategy with agile system development, the study ensured that the resulting solution was both technically robust and practically aligned with the needs of the target users.

3.3. Population Sampling and Participants Selection

3.2.1 Target Population

The target population for this study consisted of smallholder farmers in Kinangop Constituency, Nyandarua County, Kenya. Kinangop was selected due to it being a smallholder agricultural region where farming is the main economic activity for 90 percent of the residents. The area is characterized by diverse crop production, including potatoes, vegetables, maize, and legumes such as beans and peas. This agricultural diversity provided a suitable context to evaluate the AgroAI model and mobile application across various crops and farming practices.

Kinangop also consists of challenges common to Kenyan smallholder farmers, such as limited access to expert agronomic services and digital diagnostic tools, difficulties in identifying pests and diseases accurately, and poor nutrient management. These challenges, combined with the reliance on agriculture for income and obtaining basic needs, show the need for effective diagnostic tools. The researcher’s personal familiarity with the region and having grown there facilitated trust and open communication with participants during data collection.

3.2.2 Sampling Strategy and Rationale

This study employed purposive sampling, a non probability technique that enables selection of participants with specific characteristics relevant to the research objectives. This approach ensured the selection of smallholder farmers who were most likely to provide meaningful feedback on AgroAI's usability and effectiveness. Purposive sampling is appropriate given the study's focus on determining and evaluating technology adoption and user experience within a targeted demographic.

3.2.3 Inclusion and Exclusion Criteria

Participants were included if they were active smallholder farmers farming on a land less that 5 acres, had access to an Android smartphone, and were aged between 18 and 65 years. Willingness to engage with new digital tools was also required. Residency within Kinangop Constituency was mandatory to ensure uniform conditions. Verbal informed consent was obtained before participation.

Farmers cultivating more than 5 acres or already using commercial agricultural technology were excluded to focus on smallholders who didn't have access to digital services. Additionally, farmers above 65 years old were excluded due to lower smart phones access and digital skills in this age group. Communication difficulties excluded participants who could not converse in Kikuyu, Kiswahili, or English. Large scale commercial farmers were also excluded to maintain study focus and since they had access or could afford agronomy services.

3.2.4 Sample Size and Composition

Twenty smallholder farmers participated in this study, a sample size consistent with qualitative research guidelines indicating data saturation occurs within 12 to 20 interviews (Guest et al., 2006; Creswell, 2013). The sample included 12 females, who made up 60 percent of the participants and 8 male, aged 22 to 64 years. Participants were drawn from three villages to represent diverse farming practices. Farming experience ranged from 3 years, and educational levels varied from primary to tertiary education. This diversity enhanced the robustness of the findings within the smallholder context.

3.2.5 Participant Recruitment

Recruitment involved identifying and approaching farmers. Those potential participants were approached face to face, with the study's aims, procedures, and rights clearly explained in Kikuyu to ensure understanding. Emphasis was placed on voluntary participation and the option to withdraw at any time.

3.2.6 Ethical Considerations

Verbal informed consent and signed consents on questionnaires and feedback forms were obtained in the presence of witnesses, respecting local cultural norms that favor both verbal and written agreements. The consent process included clear explanations of the study purpose, activities, confidentiality, risks, benefits, and participant rights, delivered in the preferred language. This ensured informed and voluntary participation.

3.2.7 Sampling Limitations

The need for smartphone ownership for the study and research left out older farmers who use basic feature phones which did not have access to features like phone camera and Android 7+. This reduced insights from this test and study group and this may have also left out those who really need the agronomy tools. While purposive sampling limits statistical conclusions, it fits the study's focus on quality.

3.3 Data Collection Methods

This study employed a combination of qualitative and quantitative data collection methods to evaluate AgroAI's usability, effectiveness, and the real world impacts and effects. The multiple data sources helped triangulate findings for better reliability, accuracy and validity.

Interviews

Interviews were the main part of the qualitative data collection process. These interviews were formal, informal and semi-structured, allowing farmers to share their experiences while guiding the discussion toward areas like crop challenges, technology use, challenges they face in their farming journeys, and their feedback after using the app. Most interviews occurred face to face during scheduled farm visits, while others happened through phone calls and WhatsApp voice notes based on availability. Each interview started with a brief explanation of the AgroAI project and its goals. Farmers were then invited to discuss their daily challenges with crop health, how they identify problems, and what tools they use for assistance.

These conversations revealed gaps in knowledge related to diagnosing pests and diseases, as well as the level of digital exposure and interest in mobile solutions. For example, some farmers felt frustrated relying on neighbors or agrovets for advice, while others liked the idea of a tool that provides instant feedback from a photo.The thing about agrovets was that their advise was financially based. Users could only get advise if they paid and how much they would affors. In some areas, we learnt that there were times where the agrovets advised farmers to buy inputs, not because they would benefit the farmer, but just because it was almost expity time for the input. Notes were taken during the interviews, and common suggestions and complaints were recorded immediately afterward. Interview sessions also helped identify language preferences, with several farmers suggesting Kiswahili or local dialects for better understanding. These insights shaped the app interface and feedback texts for future models and upgrades.

Questionnaires

Questionnaires were created to collect standardized responses from a larger group of farmers. The questionnaires contained questions to answer before using the AgroAI application, when using and after using it. Each questionnaire was completed after the farmer used the AgroAI app in real life situations, such as photographing a sick crop and reading the diagnosis provided by the system. The questionnaire included both multiple choice questions and questions that assessed things like ease of use, satisfaction with the results, and confidence in the app's recommendations. It also had short open ended sections for farmers to share any problems they encountered or ideas for improvement in the future.

The questionnaire was distributed in person during farm visits, often with help to ensure clarity. In situations where literacy was limited, the researcher read questions aloud and recorded the answers based on what the respondent said. This ensured that every farmer, regardless of education level, could provide feedback. The simplicity of the questions and their translation into Kiswahili helped gather honest and useful insights. The responses to the questionnaires identified common patterns in user experience and guided updates to the mobile applicaion navigation, image capture process, and content delivery.

Observations by the Researcher

Observations were also crucial to this study, especially in understanding the natural behaviors and hidden challenges that arose when farmers interacted with the AgroAI system. Unlike interviews and surveys that depend on spoken or written feedback, observations allowed the researcher to see users in action. From opening the app to taking pictures and interpreting results. This method took place during on farm visits, often while farmers diagnosed real problems with their crops.

What stood out during these sessions were the small details that often get overlooked. Some farmers found it difficult to take clear pictures due to bright sunlight or poor camera angles. Others struggled to read instructions quickly, particularly if they were written in formal English. Observing these challenges helped the researcher improve app features like text size, icon placement, and the prompts guiding users through the image capture process. Some farmers also attempted to use the app while multitasking, such as holding tools or using one hand, which led to the decision to reduce the number of clicks needed for a diagnosis.

Feedback Forms

Along with structured methods, informal feedback sessions played a key supporting role in the entire process. These were casual conversations held after a farmer used the app and mostly sometimes while walking through the farm or sharing tea or just sharing stories about our childhood. They offered honest, unfiltered opinions, especially on points farmers didn’t consider “important enough” to mention in interviews or forms. Feedback from these informal sessions often included valuable design suggestions, like “make it show pictures of the disease too” or “can it read out the answer?” Such comments were recorded in a notebook and helped prioritize and design and develop future updates.

Datasets Collection

To develop and train the AgroAI diagnostic models, existing datasets were utilized. These included publicly available crop image data from Kaggle called PlantVillage, as well as image datasets sourced from AgriTech Analytics, a Kenyan agricultural technology firm with offices in Spur Mall, Ruiru and Ngara, Nairobi. These datasets provided the necessary diverse and localized images of pests, diseases, and nutrient deficiencies critical for effective model training, accuracy and system performance.

Overall, these methods ensured that the AgroAI app was shaped not only by technical design principles but also by the real life experiences of the people it aims to help. Farmers were not just research subjects but they were active contributors in the every phase of developing our AgroAI mobile application, whose insights directly influenced how the system developed and how it turned out to be. The multiple layers of engagement created a strong foundation of understanding the needs and the needed output which is essential for building a practical, usable, and relevant solution.

3.4 Data Analysis Methods and Processes.

After collecting data from interviews, questionnaires, and field observations, a thorough analysis was conducted to understand the findings and draw conclusions that would guide the development and improvement of the AgroAI system. This analysis aimed to extract meaning, identify patterns, and interpret the views and behaviors of smallholder farmers using the AgroAI application so as to know how and where to improve and change based on the feedbacks from farmers. Since this study used a mixed methods approach, both qualitative and quantitative data analysis techniques were applied to ensure a comprehensive interpretation.

3.4.1 Qualitative Data Analysis

For the qualitative data, mainly from interviews and observation notes, a thematic analysis approach was used. This method included several steps. First, all interview forms and observation diaries were transcribed and reviewed multiple times to get familiar with the content. During this phase, initial thoughts were recorded. Next, the data was coded by highlighting important phrases, sentences, or quotes that represented key issues, concerns, or suggestions from the farmers. These codes were then organized into broader themes that appeared across multiple responses. For example, frequent mentions of difficulties in navigating the app were clustered under a theme called "usability challenges," while responses indicating trust in AI generated recommendations created a theme like "trust in diagnosis."

Thematic analysis helped reveal patterns in the farmers' responses that would not have been clear through numbers alone. For example, while some farmers provided positive answers on the questionnaires, interviews showed that a few struggled with specific features but were reluctant to express their dissatisfaction. These insights became visible only through deeper narrative analysis. Themes like the need for local language support, simpler instructions, and offline access were consistently mentioned by many participants, highlighting areas that needed attention in the system's design and development.

3.4.2 Quantitative Data Analysis

The quantitative data, mainly collected from structured questionnaires, was examined using basic statistical techniques. All questionnaire responses were recorded where the data was cleaned and checked for accuracy. Rating responses were assigned numerical values for example, 1 for "Very Unsatisfied" to 5 for "Very Satisfied") to enable mathematical analysis. Using Excel, averages, frequencies, and percentages were calculated for each question. This helped the researcher see, for instance, how many farmers rated the application as easy to use or how many felt confident in the application's ability to accurately diagnose crop issues accurately. The statistical summaries allowed for notification and addressing of areas needing improvement. These visualizations were especially useful for identifying patterns and sharing findings during review sessions.

3.4.3 Data Validation

To ensure the reliability of the findings, comparison was used to compare insights from various data sources. For instance, if a farmer stated in a questionnaire that the app was "easy to use" but struggled during field observations, the researcher viewed this as a possible misunderstanding or social desirability bias, where participants offer favourable answers instead of their true experiences. In such cases, more importance was placed on what the farmer did by looking at that researcher’s observational data rather than what they said or what was in the questionnaire or feedback forms. This approach helped ensure that conclusions were based on a thorough and honest understanding of how the application performed in real farming situations.

3.4.4 Comparative Analysis

Additionally, relevant variables were examined to find the relationship between user characteristics such as age and education level and their experience with the app. For example, comparing the satisfaction levels of younger versus older farmers helped determine whether digital literacy affected perceived ease of use. Although the sample size was relatively small and advanced statistical methods like regression were not used, these comparisons added depth and understanding to the findings.

3.4.5 Integrated Analysis Framework

The mix of thematic and descriptive analysis provided a rich, detailed, and balanced understanding of the AgroAI system's usability, effectiveness, accuracy, ease of use, and acceptance among smallholder farmers. This analysis process not only confirmed the system's strengths but also highlighted gaps and limitations that were critical for refining both the AI model and the mobile application interface before full deployment.

3.5 Methodology for System Design Process

3.5.1 Design Methodology Framework

The system design methodology employed a user centered design approach combined with iterative development principles to ensure the final system met the specific needs of target smallholder farmers. This methodology prioritized simplicity, accessibility, and practical functionality over complex technical features that may not align with the target users' capabilities and needs.

The design process followed a structured framework that began with user needs analysis, progressed through conceptual design, and ended in detailed system architecture planning. This methodology ensured that all the decisions were driven by user requirements and their needs rather than technology preferences, maintaining focus on solving real world agricultural challenges facing small scale farmers.

3.5.2 User Centered Design Approach

The methodology adopted a participatory design approach where target users were actively involved throughout the design process. This approach recognized that smallholder farmers possess valuable domain knowledge about their agricultural practices and challenges, which needed to be included and put into the system design to ensure practical relevance and usability. Stakeholder engagement techniques were employed to gather design requirements through multiple touchpoints including field observations, informal interviews, and prototype testing sessions. The methodology emphasized understanding user needs, technical constraints, and factors such as education levels, device capabilities, and connectivity limitations that influenced design decisions.

3.5.3 Iterative Prototyping Methodology

The design methodology employed rapid prototyping techniques to validate design concepts early and frequently throughout the development process. This iterative approach allowed for quick testing of design assumptions and enabled continuous refinement based on user feedback before committing to full development. Prototyping followed a progressive approach, starting with low quality prototype to establish basic user flows and interaction patterns. The interactive prototypes were developed using Jetpack Compose and Kotlin, leveraging Firebase for backend integration. This approach enabled realistic simulation of key system functions, allowing users to experience the proposed interface and provide meaningful feedback on usability and functionality.

3.5.4 System Architecture Design Methodology

The architectural design methodology employed established software architecture patterns to create a foundation that could accommodate future enhancements and changing requirements. The methodology prioritized designing for low resource environments by incorporating minimal data usage capabilities, and compatibility with basic mobile devices. Architecture decisions were evaluated against criteria including performance on low-end hardware, network efficiency, and operational simplicity to ensure the system remained accessible to the target user base.

3.5.5 Design Validation Methodology

The design validation methodology employed multiple validation techniques to ensure the proposed system design met user needs and technical requirements. This included heuristic evaluation methods where fellow developers and agronomists assessed interface usability, technical feasibility analysis to confirm implementation viability, and user feedback collection to validate design decisions against real world usage scenarios. Validation criteria were established based on measurable usability features, performance scores, and user satisfaction feedback. The methodology incorporated both formal evaluation techniques and informal feedback mechanisms to capture comprehensive insights about the design's effectiveness and to note the various areas for improvement.

3.6 Ethical Considerations

This section addresses the ethical principles and guidelines that were followed throughout the research and development process to ensure the study was conducted responsibly and with integrity.

3.6.1 Informed Consent

All participants involved in data collection activities, including questionnaires, interviews, surveys, and system testing, were provided with clear information about the purpose and scope of the research, how their data would be collected, used, and stored, and their right to withdraw from participation at any time. The voluntary nature of their participation was emphasized, and verbal and sometimes written consent was obtained from all participants before any data collection commenced. This approach ensured that participants were fully aware of their involvement and could make informed decisions about their participation.

3.6.2 Data Privacy and Confidentiality

Strict measures were implemented to protect participant privacy throughout the research process. Personal identifying information was was only handled by the researcher and no unauthorized personnel or party got access to personal information . All data collected was stored securely and accessed only by authorized personnel, with confidentiality agreements being adhered to throughout the research process. These measures ensured that participant identities remained protected and that sensitive information could not be traced back to individual contributors.

3.6.3 Data Protection Compliance

The research adhered to relevant data protection regulations, including compliance with local data protection laws and institutional policies. Electronic data was stored securely with appropriate access controls, and proper disposal procedures were followed for sensitive documents after the research period. Regular backup procedures were implemented to prevent data loss while maintaining security protocols. This comprehensive approach to data protection ensured that all collected information was handled in accordance with established legal and institutional requirements.

3.6.4 Participant Welfare

Priority was given to ensuring participant well being throughout the research process. Interview and survey questions were carefully designed to avoid causing distress, and participants were informed they could skip questions they were uncomfortable answering. No deceptive practices were employed in data collection, and participants were treated with respect and dignity throughout the process. This approach created a safe and comfortable environment for all participants while maintaining the integrity of the research.

3.6.5 Institutional Approval

Research approval was sought from the supervisor, and all research activities were conducted within the guidelines set by Zetech University. Regular consultation with the supervisor ensured that ethical standards were maintained throughout the study. This oversight provided an additional layer of protection for both participants and the research process itself.

3.6.6 Intellectual Property and Attribution

All sources used in the research were properly cited and acknowledged, with permission being obtained for use of copyrighted materials where necessary. Original contributions were clearly distinguished from existing work, and plagiarism was strictly avoided through proper referencing practices. This ensured that the research maintained academic integrity while properly recognizing the contributions of other scholars and practitioners in the field.

3.7 Research Limitations

This section outlines the key limitations that were encountered during the course of the research and development of the AgroAI mobile application to the final step. These constraints may have influenced the study's scope, the implementation process, the depth of data analysis, the results, the accuracy, and maybe the findings in some way.

3.7.1 Funding and Resource Limitations

Limited financial resources restricted access to advanced data collection tools, commercial AI services, and high end computing infrastructure. The development and deployment of the AgroAI prototype were executed with minimal personal funding, which affected the ability to scale testing, acquire better devices for compatibility testing, and pay for extended field engagement or participant incentives and security testing of the application.

3.7.2 Dataset Size and Diversity

Although real world image data was sourced from both Kaggle and AgriTech Analytics, the available datasets were not large or diverse enough to represent all possible pest, disease, and nutrient deficiency scenarios affecting Kenyan crops. This may have constrained the AI model’s ability to generalize across different geographical zones and crop varieties. Also, the relatively small number of participants in the usability testing phase limited the depth of statistical analysis.

3.7.3 Limited Availability of Localized Datasets

Much of the publicly available data used to train and test the AI models originated from international sources like PlantVillage. These datasets may not accurately reflect the unique environmental conditions, crop diseases, or pest strains found in the Kenyan context. This limitation could potentially reduce the precision and accuracy of the model when applied in real world local farming settings.

3.7.4 Geographic Scope

The study was confined to smallholder farmers in Kinangop Constituency, Nyandarua County. While this provided a focused and manageable testing environment, it limits the scope of findings to other regions in Kenya with different farming practices, languages, farming problems, and environmental conditions.

3.7.5 Technology Access and Compatibility

Participation in the system testing required access to an Android smartphone, thereby excluding farmers using basic mobile phones, commonly referred to as "kabambe". This limitation may have biased the sample toward younger, more tech-savvy farmers, and restricted the inclusivity of the study. Additionally, the application could not be tested on iOS or older Android versions due to technical constraints.

3.7.6 Language and Digital Literacy Barriers

Despite efforts to make the AgroAI system as much user friendly as possible, education levels and digital ability varied across participants. Some users experienced difficulties understanding interface instructions or fully engaging with features, especially those with minimal education or first time smartphone users. This may have impacted the quality of feedback and real world engagement with the system.

3.7.7 Researcher Role and Subjectivity

The researcher also served as the system developer, data collector, and evaluator, which introduces a possibility of bias. Although care was taken to maintain objectivity, having the same individual design, build, and assess the solution could have affected how usability issues or participant responses were interpreted.

3.7.8 Lack of Long Term Evaluation

Due to time and resource constraints, the study was not able to track and test long term adoption, behavior change, or seasonal usage of the application. Consequently, conclusions about the sustained impact of the AgroAI system remain limited.

CHAPTER 4
SYSTEM ANALYSIS AND DESIGN

4.1 Introduction

This chapter presents a detailed analysis and design of AgroAI, a mobile based Android application tailored to assist smallholder farmers and farming communities in Kenya in diagnosing crop health issues through image recognition technologies and machine learning and delivering localized treatment recommendations. It begins by introducing the system development methodology that guided the research and the project. It provides the structure for analysis, modelling and implementation. A feasibility study is then discussed to assess how viable the system is in practical real world agricultural contexts. The chapter further explores the system's requirements elicitation and analysis, based on insights gathered from farmers and early prototype testing. Modelling tools like data flow diagrams, entity relationship diagrams and use case diagrams are used to visualize systems processes and user interactions. It concludes with the physical design of the system and it shows how the various named system components, the mobile application, the API and the AI model, integrate and work together to deliver reliable crop diagnosis and and tailored recommendations. AN early app prototype called CropScan was developed and tested to validate core functionalities and guide the final system design and development.

4.2 System Development Methodology

The development of AgroAI system followed the Agile software development lifecycle methodology. This methodology was selected due to its flexibility, adaptability and the ability to accept and accommodate continuous feedback from farmers who were involved in the every step of the design and development. These factors were essential given the real world challenges faced by farmers and the limited resources available during the research and development of this project. This iterative nature of Agile SDLC allowed the system to evolve over time, guided by the users or farmers feedback, practical testing of CropScan, and more importantly continuous improvement, ensuring the final product met the solution required met the real needs of the Kenyan smallholder farmers.

Agile software development lifecycle divided the design and development process into manageable parts, each focusing on the delivery of specific improvements or creations or outputs. The initial phase involved gathering user needs and requirements from farmers in Kinangop Constituency, Nyandarua County through interviews, questionnaires and observation. The insights gained from this part were used to figure out and inform the early design and planning of core functionalities of the system. The development was then carried out starting starting with the creation and development pf a working prototype called CropScan which was developed using Kotlin and Jetpack Compose. CropScan provided a simple environment for testing core functions such as logins and registration, image capture and uploading crops images sections, navigating through the application, ease of use, and the visualization of results and recommendations.The results and insights gained from CropScan helped shape and improve the AgroAI development.Subsequent developments and iterations introduced the integration of a machine learning AI model the recommendation files and engine and support for additional features to the AgroAI application. Each section or feature was tested for practical evaluation in the real world. Each iteration followed a cycle of planning, development, testing and feedback collection. The AI model however did include the feedback process.

Agile methodology enabled rapid and constant immediate problem solving and constant alignment with the expectations of the users. This approach was useful and important for identifying challenges on usability, network constraints, device compatibility and language constraints and challenges.It also ensured that the final version of AgroAI was tailored not only to meet the technical requirements but also the real world realities and the needs of the smallholder farmers in the Kenyan context. Agile SDLC delivered a solution that was practically relevant by placing smallholder farmers and their needs at the center of the development process.

4.3 Feasibility Study

A feasibility study was conducted to determine the practicability, how viable and the sustainability of developing and implementing AgroAI, an AI powered mobile application with the purpose of helping Kenyan smallholder farmers in diagnosing crop diseases and pests. The feasibility analysis focused in four areas namely technical, economical, operational and legal feasibility.

4.3.1. Technical Feasibility

The AgroAI system was technically feasible based on the availability of essential development tools, infrastructure and the developer expertise and knowledge. The project used readily available and accessible technologies such as Kotlin and JetPack Compose for the android application front end development, PyTorch for training the model, Firebase for storage together with FastAPI for back end services like communication. Most farmers possessed android smart phones capable of running the light weight application. The AI model was also optimized to run on device, minimizing the need for high computational power or consistent internet access which was a challenge for more than half of small scale farmers.Also, the testing of our prototype, CropScan, validated the technical feasibility in the real world conditions and environment.

4.3.2. Economic Feasibility

The project was developed under tight financial constraints using personal resources. Despite this, it remained viable due to the free open sourced tools like PyTorch, Google Colab and others, the freely available and accessible datasets and cheap cloud services. If scaled, the application could be monetized through partnerships with agricultural Non government organizations, governments programs, or premium advisory services so as to ensure long term sustainability.

4.3.3. Operational Feasibility

Operational feasibility was confirmed through feedback from smallholder farmers in Kinangop, Nyandarua County who participated in testing the prototype. The farmers expressed a desire and interest in using the a tool that could identify crop issues using their smart phones. The applications simple UI and lack of complexity were all tailored to align with the technological literacy levels and connectivity conditions of rural areas and users.

4.3.4. Legal and Ethical Feasibility

The development and deployment of AgroAI and CropScan adhered to relevant data protection standards set by the Kenya's Data Protection Act (2019) which mandates informed consent, lawful data processing and user protection. All data collected from from users was made anonymous and used only for research, development and training purposes. Participant consent was required and obtained before any interaction. The application did not break any rules on intellectual property rights, as datasets like PlantVillage and those from AgriTech Analytics were used within the bounds of their of their terms of use.

4.4 Requirements Elicitation

Requirements elicitation was a very important phase in the development of AgroAI, guiding the identification of user needs, system functionalities and technical constraints. The goal of this was to gather relevant information that would shape the design and features of a mobile based diagnostic tool tailored fir small farmers in Kenya.

4.4.1. Data Collections Tools Used

To gather accurate requirements, multiple data collection tools were used:

Questionnaires

Before development, questionnaires were administered to twenty small holder farmers in Kinangop Constituency, Nyandarua County. The questionnaire focused on common farming problems, familiarity with digital farming tools and solutions, their expectations of digital tools for farming, and languages that should be used.

Informal Interviews

One on one interviews were conducted by the researcher with the selected farmers and a local agricultural officer. These conversations explored the attitudes toward mobile apps, expectations for usability, willingness to try digital tools and any specific frustrations with the existing digital tools and the gaps they leave. These interviews revealed the important needs if the farmers such as the need for simplicity in a mobile application, trust in technology, and the importance of features like local languages support.

Observation

Field visits conducted by the researcher allowed observation of how farmers traditionally managed and went about crop diseases and pests. This included noting of how they they visually inspect crops, how long it takes and how available are the agronomy services, and the decisions made by the farmers based on these observations.

Feedback Forms and Researchers Notes

After the testing and use of the prototype application called CropScan, participants were given feedback forms to document their experiences. These forms included questions on ease of use, the accuracy of the diagnosis, layout preferences, complexity of the app and any changes and comments they would like to add or make. The researcher also maintained detailed field notes taken through the whole process, capturing various aspects in different sessions, common errors and points maybe missed in questionnaires and interviews, and questions posed by the farmers.

4.4.2. Relevance of the Data Collected

The data collected during this phase directly supported the research objectives and the design of AgroAI system. It helped confirm that the farmers needed a mobile diagnostic tool that could be accessible at any time, easy to use, navigate and understand, work offline, use visual elements for navigation, provide feedback in local languages, and deliver localized recommendations based on the crop and the crop issue. Farmers also emphasized on the need and importance to take a photo of a crop issue or upload one and within a short time receive a diagnosis and the recommendation that is accurate.

4.4.3. Participant Involved

A total of twenty smallholder farmers were selected. They came from various villages within Kinangop to ensure diversity in age, education levels, gender crop types and digital exposure. Each participant either owned an android smart phone and was actively engaged in crop farming. This selection process was designed to ensure that the voices of those most likely to use AgroAI were accurately captured and their need understood perfectly and that they were represented in the system design.

4.4.4. Outcomes of the Elicitation Process

The elicitation process led to the identification of key user driven requirements that shaped the final system. First it revealed the need for a digital agronomy tool that could run on low end smart phones accessible by small holder farmers. Second, it emphasized on the importance of having an interface that uses large, clear icons and simple instructions that can be understood even by the end users with limited education and digital knowledge. Third, it supported the development of a multiple purpose AI model capable of detecting both diseases and pests and nutrient deficiencies in one tool. Also, it validated the use if localized recommendations based on one crop type, region and the diagnosis which made the system more relevant to the users.

See Appendix E, F, and G for a sample Questionnaire, User feedback and Observers notes used in this requirements and data gathering phase.

4.5 Data and System Analysis

This section presents the analysis of the data collected from the farmers during the requirements elicitation phase. Both qualitative and quantitative techniques were used to interpret responses from questionnaires and the other gathering techniques. The analysis focused in identifying the core user needs challenges faced during diagnosis, and the expectations the farmers had toward a digital AI powered mobile solution.

4.5.1. Data Analysis Process

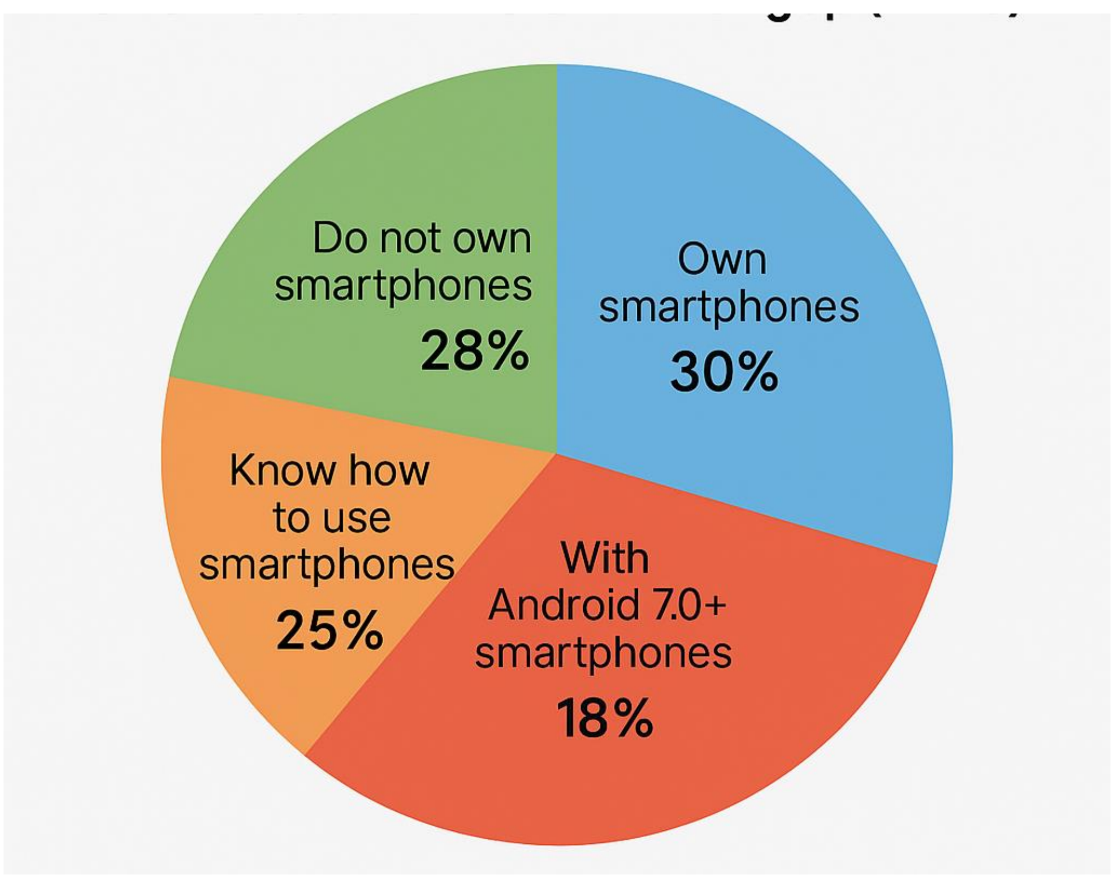
The data collected from twenty small holder farmers in Kinangop Constituency was compiled and analyzed using Excel. Responses were from structured questionnaires were categorized so as to enable calculations of percentages and identification of trends and problems. Open ended feedback was coded into thematic categories, for example, “Technology comfort level” and “Expectations of mobile digital diagnostic tools”. These themes helped determine the users behaviors and system expectations that AgroAI needed to address. Visual representations were generated to represent the information and insights derived.

Smart Phone accessibility

Table 4.1. Smart Phone accessibility among farmers in Kinangop(n = 75)

Parameter	Number of farmers	
Total number of farmers surveyed	40	
Farmers who owned smart phones	28	
Farmers who know how to use smart phones	25	
Farmers with Android 7+ smartphones	22	

Figure 4.1 Smart Phone accessibility among Farmers in Kinangop



4.5.2. Sample Data Analysis Findings

The following insights were extracted from the analysis:

- More than 70% of the respondents owned smart phones, and 87 percent of this owned Android 7+ powered phones, while about 23% of the respondents used non-smart phones.
- Only 15% of the farmers had ever used any farming mobile application and even less interacted with AI powered farming tools.
- 60% preferred Kiswahili while 35% preferred English. A few suggested use of Kikuyu which was translated to use of local languages for basic instructions and responses.
- More than 85% of the respondents emphasized on the need for the offline capabilities due to inconsistent and unreliable internet connection in the rural regions.
- Farmers expressed the desire to have one tool have the capabilities of to detect the crop pests, the diseases and the nutrients deficiencies rather that have multiple applications.
- Most of the farmers requested the simplicity and non complexity of the application and for it to have only relevant items and pages.

These findings directly shaped the design of AgroAI, confirming the need for a single simple, lightweight, non complex android application with offline capabilities.

Figure 4.2: Sample Data Findings For Data Analyzed

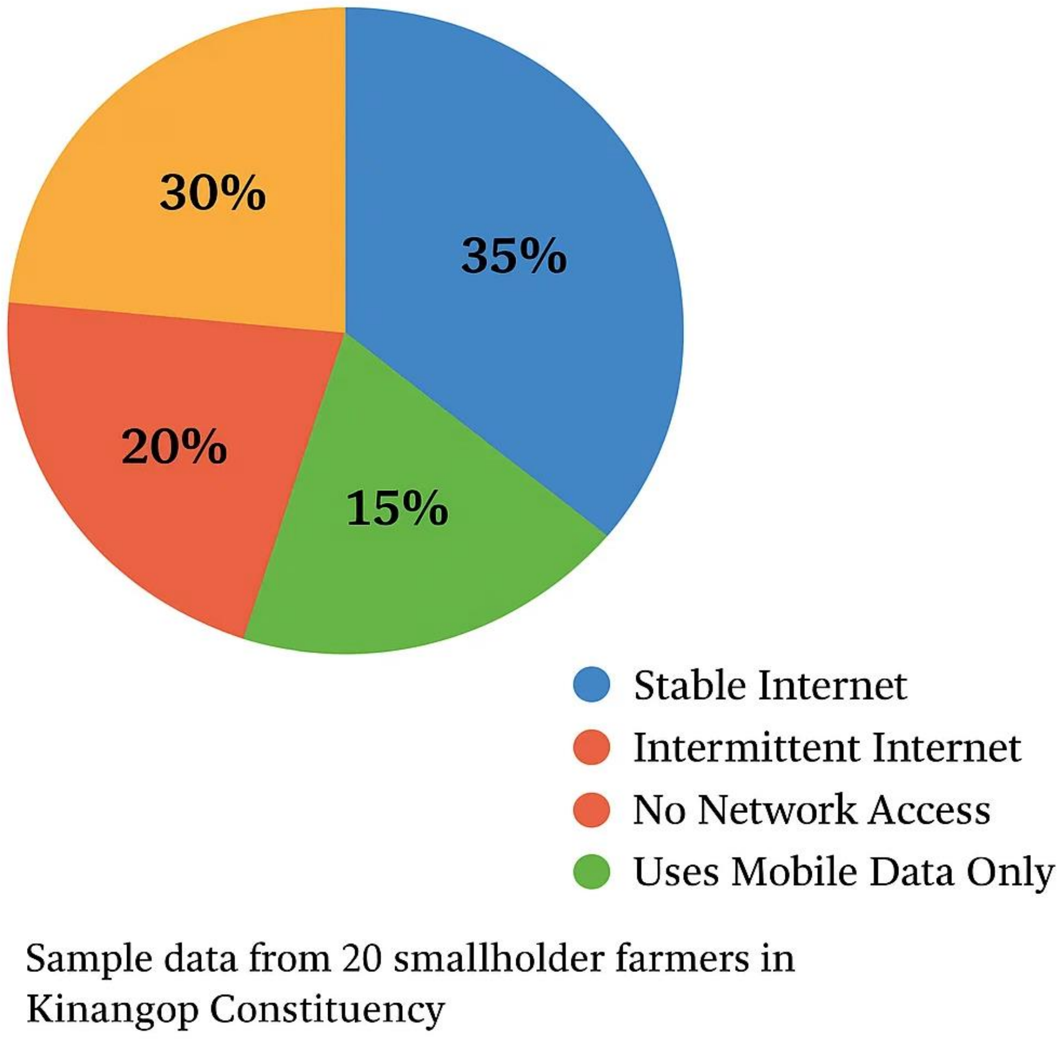


Figure 4.3. Language Preferrence of Farmers

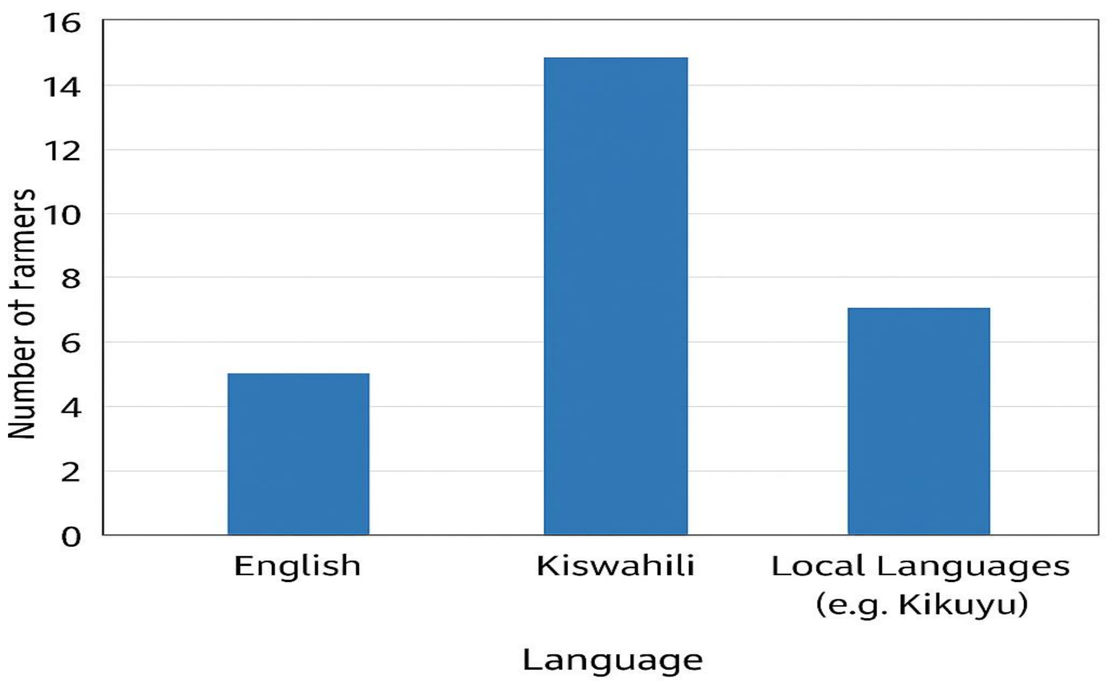
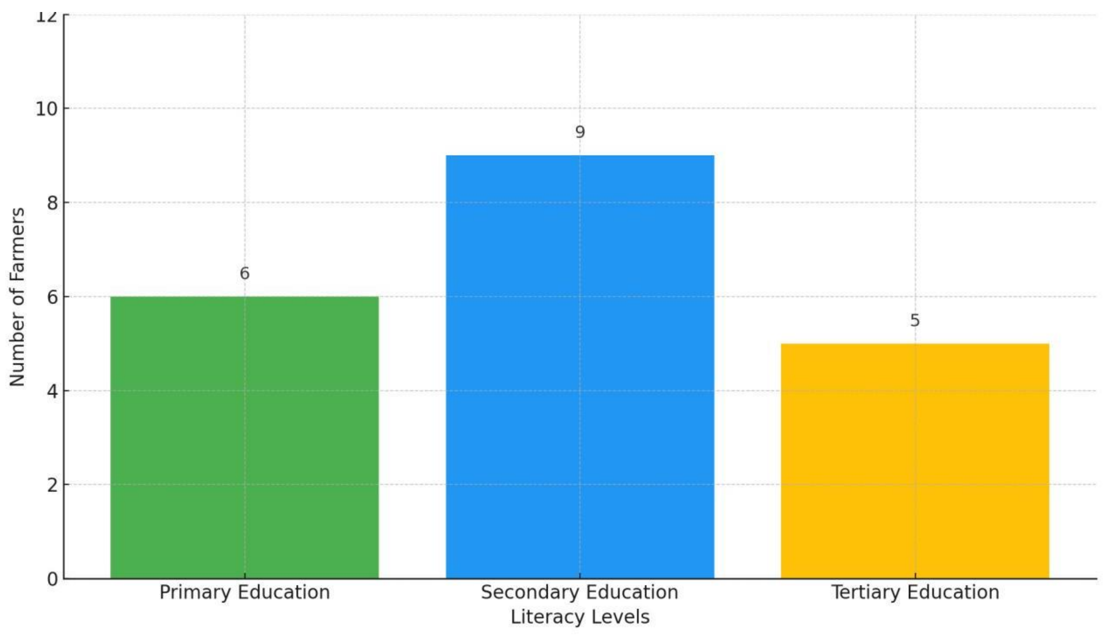


Figure 4.4: Literacy Levels and Ease of AgroAI App Usage



4.5.3. Integration of Findings into System Design

The insights gathered from the data analysis process were important in refining the system design and ensuring that AgroAI addressed the real actual needs of the smallholders in Kenya. First, it became clear that the application needed to be light weight and optimized for performance on low power smart phones that were owned by the farmers. The user interface design was influenced by the feedback received from the farmers regarding ease of navigation, leading to a simplified, icon based layout with minimal pages to reduce confusion and increase accessibility especially for users with limited literacy. Offline capability was also regarded given the poor internet connectivity reported across the region and other rural areas where this system would be most used. Also, farmers expressed the desire for all in one platform or tool that performs the task of diagnosing crop pests, diseases and nutrient deficiencies. The system was also designed to provide actionable and localized recommendations based on the diagnosis, the crop type and the region where the farmer operates. These user informed decisions made sure that AgroAI remained grounded in practicality and real world usability throughout the design and development.

4.6 System Specification

This section outlines the specific requirements of the AgroAI mobile application based on the information and findings from the requirements elicitation and the data analysis phases. These specifications define what the system must do, functional specifications, and how it should perform, non-functional requirements. They serve as the foundation for the system's architecture, design and implementation, ensuring that the AgroAI solution meets the needs of smallholder farmers in Kenya effectively.

4.6.1. Functional Requirements

The functional requirements describe the core actions and operations pf the AgroAI system is expected to support. The AgroAI system was built to fulfill the following:

i. The system allows farmers to register an account and log in securely using simple credentials
ii. The system allows farmers to take a photo of their crops via phone camera or upload an image for analysis and diagnosis.
iii. The system processes the uploaded images using a trained an AI trained model to identify the crop health issues, either a pest, disease or nutrient deficiency.
iv. After diagnosis, the system shall display the diagnosis or the results to the user stating the identified the issue.

v. The system displays a localized recommendations based on the diagnosis, crop type, and the regional data to assist farmers in responding appropriately.
vi. The system enables and allows for submission of feedback from the app users for continuous improvement of the application and the whole system.

4.6.2. Non-functional Requirements

These requirements define how well the system performs and describe qualities essential for user satisfaction, accessibility and long term sustainability.

Usability

The application has a simple user interface with large icons, clear buttons, and a few necessary pages.

Reliability

AgroAI model gives a 95% accurate diagnosis for pests, diseases and nutrient deficiencies.

Performance

The image processing, diagnosis and recommendation takes between 4 and 10 seconds on Android 7+ phones.

Scalability

The back end handles 35 crops, 770 pests, diseases and nutrient deficiencies and can handle a large number of tests.

Availability

The app have offline working capabilities for image capture and submission and the application is easily accessible and available for all farmers.

Security

User data, like logins and registration information is safely stored in highly encrypted databases and user data is deleted at the users request.

Compatibility

The app is compatible runs on phones with as low power as Android 7, and fits different screen sizes.

4.8. Design

This section presents the design that guided the development of the AgroAI system.The system design shows the transformation of farmers requirements and needs into a structured system architecture that showed how different functions and components function. The phase was divided into three phases, the traditional understanding of traditional farming and diagnostic process, the logical and the physical design so as that the system may align with the user needs.

4.8.1. Traditional Pests and Diseases Diagnosis Process

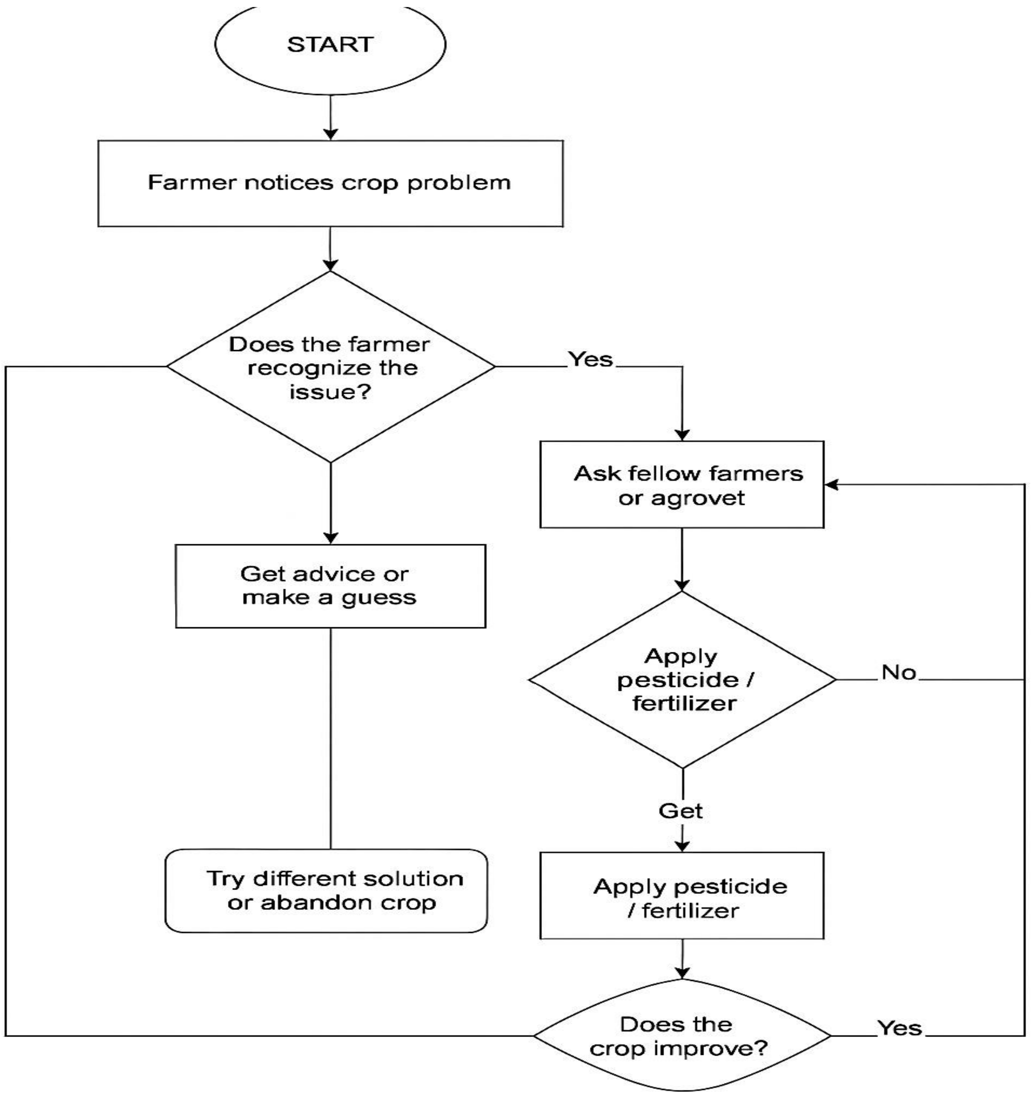

START
Farmer notices crop problem
Does the farmer recognize the issue?
Yes
Ask fellow farmers or agrovet
Get advice or make a guess
Apply pesticide / fertilizer
No
Get
Try different solution or abandon crop
Apply pesticide / fertilizer
Does the crop improve?
Yes

Figure 4.5. Traditional Diagnosis Process

Before the implementation of AgroAI, smallholder farmers relied on traditional methods to identify and address their crop issues such as crop pests, diseases and nutrient deficiencies. These methods were mostly based on guesswork, traditional beliefs, community knowledge and informal consultations with other farmers.Most of these times, this resulted in low crop yields, waste in funds and resources, crop losses and increased food insecurity. These are the processes and steps that took place when using traditional farming methods:

1. **Noticing the problem**

The process begins when the farmers notices changes in his crops, such as discoloration of leaves, yellowing of leaves, or even green worms on his crop leaves and stem. The farmer then inspects the crop inspects the crop manually using visual observation.

2. **Diagnosis or Consulting with neighbouring farmers**

The farmer proceed to make a diagnosis based on guesswork or personal experience, and sometimes seeks advice from his or her fellow farmers.

3. **Applying a chosen treatment**

Based on gathered opinions from friends or his own diagnosis, the farmers purchases an input and applies to the crop. The solution is chosen without proper diagnosis, making it a gamble. Wrong use of fertilizers and other inputs is cery common in this phase. In case the diagnosis was wrong, the farmer loses the resources spent on acquiring the inputs, the inputs finds their ways into the soil leading to soil degradation and later environmental pollution. Since the issue to the crops is not solved, the crops are therefore lost leading to low crop yields and financial insecurity for the farmer.

4. **Waits to observe results**

After applying the inputs, the farmer waits for results monitoring the crop to see any changes. If the symptoms begin to fade or disappear, the treatment is considered to be working. In most cases, no changes are seen due to misdiagnosis and application of the wrong inputs

5. **If the Problem Persists**

If this problem persists, the farmer either applies more inputs or chooses a new input to apply. All this time, the crop continues to worsen and the chances of curing the plant lower.

6. Yield Impact

The delay in identifying and treating the crop lead to damage of the crop. The result of this is reduced harvest, lower income, land degradation, environmental pollution, mental stress, and total crop failure.

7. **End**

The process ends when the farming season ends either with partial recovery or total loss. Lessons maybe learned for future farming seasons but the cycle is very much likely to repeat. The lack of reliable and timely and accurate diagnostics tools remains a very major problem for the smallhoder farmers in developing countries.

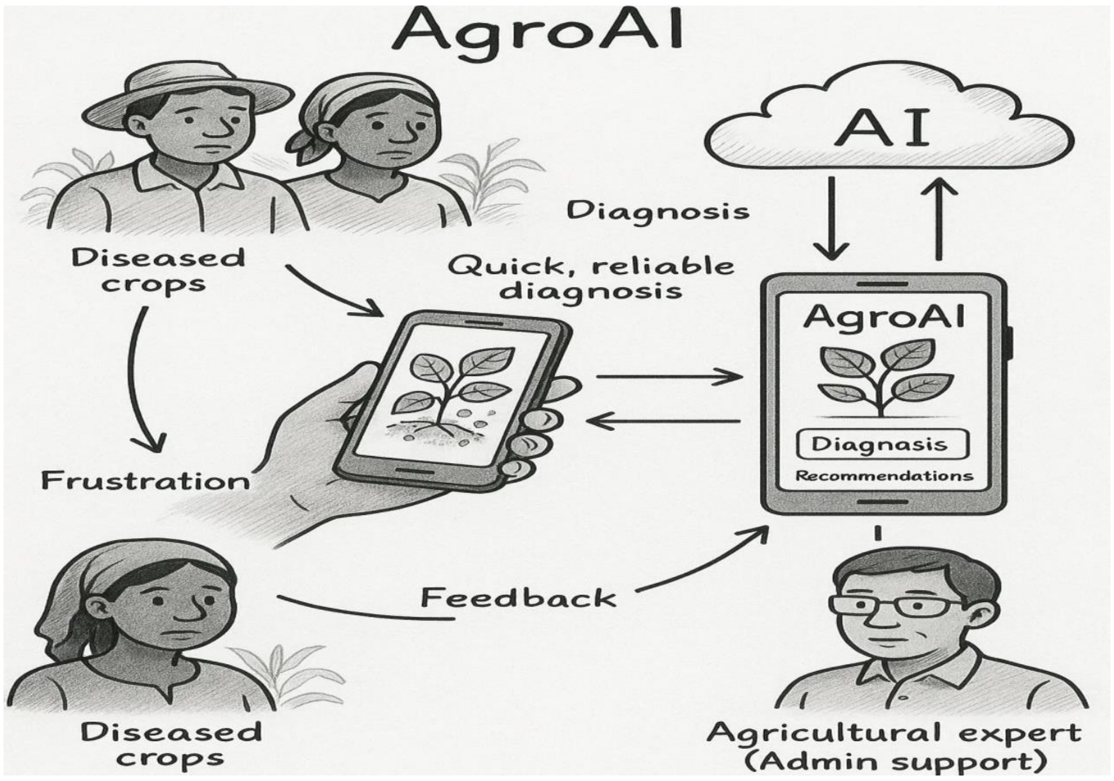
4.8.2. Logical Design of AgroAI

The logical design of the AgroAI system provided a representation of the AgroAI system's components and their interactions without involving the technical details of implementation like the real coding and implementation and the testing.This design phase mostly focused on the conceptual framework of the system, and highlighted all the necessary and required paths and procedures to follow so that the system can meet the desired and intended functionality.

Rich Picture Diagram

A rich picture diagram was generated using Sora to show all the stakeholders, the challenges, the communication paths and the expectations of the system. The primary stakeholders included the smallholder farmers who were the target of the research, the researcher and Zetech University.

Figure 4.5. Rich Picture Diagram for AgroAI system



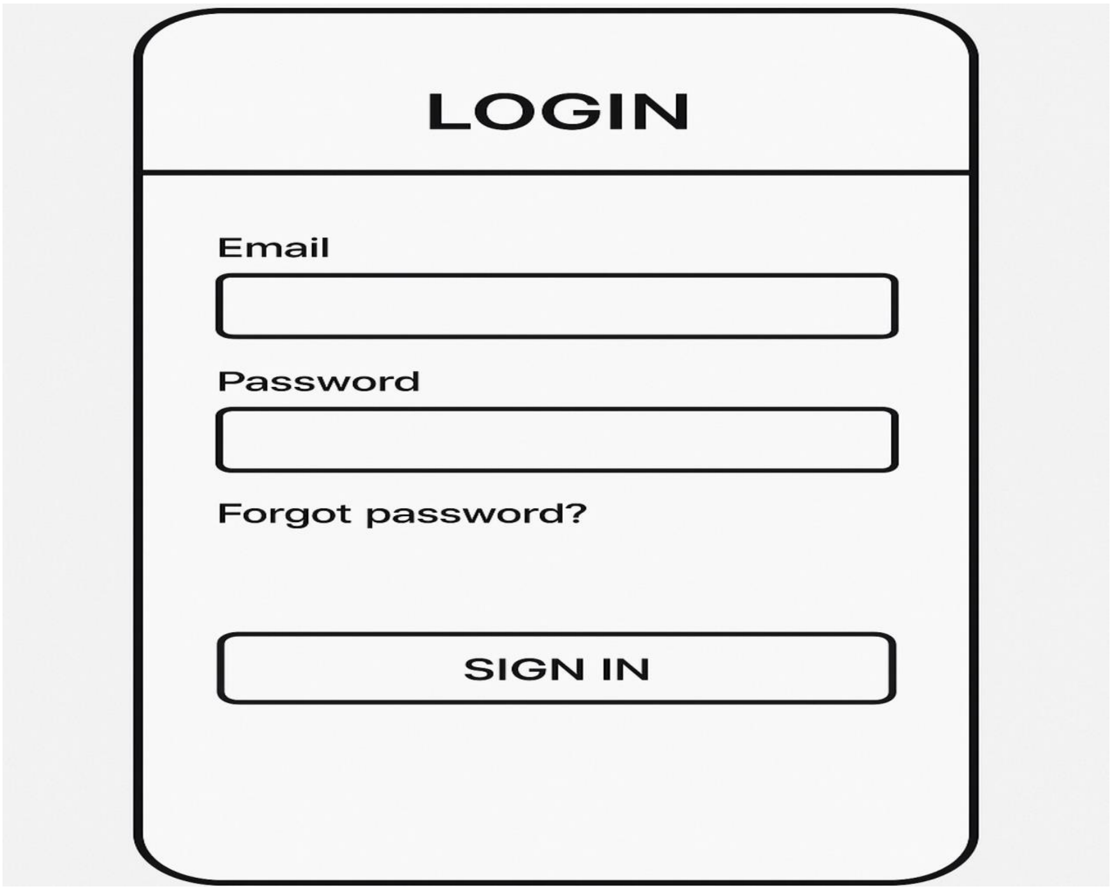
The diagram depicted typical scenarios, the main one being where the farmers notices their diseased crops, then they use the AgroAI application in their phones or uploads an already photographed photo, then receiving real time diagnostic feedback and then receiving a recommendation on what to do.

The rich picture helped identify users concerns such as farmer frustrations for their crop losses and the demand for a timely and accurate diagnostic tool that addressed their issues.

User Interface Wireframes

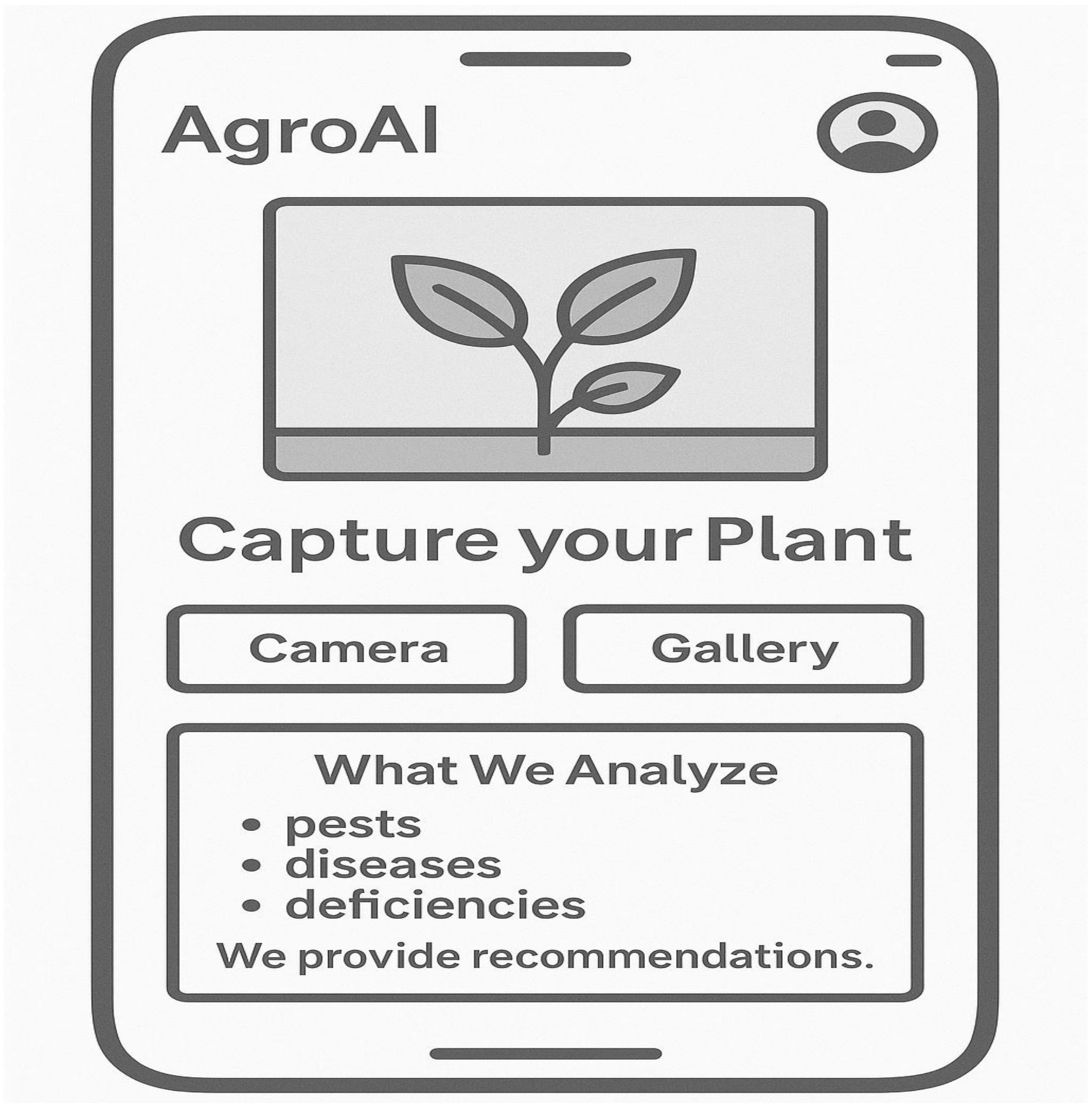
Wire frames were drawn on paper and later digitally generated and created using Sora to outline the structure and functionality of the screens of the AgroAI apps that would be important. These wireframes and visuals were used during the design and development phase to ensure that the system being created aligned with the interests and needs of the small scale farmers.

Figure 4.6. Wireframe of the AgroAI Login Screen



The login page featured standard authentication fields for email and password, a ‘Forgot Password’ link to reset the password in case of loss, and a ‘SIGN IN’ button to login into the system.

Figure 4.7. Wireframe of the AgroAI Home Screen



The Home Screen displayed the landing page of the AgroAI mobile application, clear buttons with clear instructions of Camera or Gallery for photographing a crop or uploading the images of already photographed ones. The home screen also displays the icon and button for visiting the users account and also a box detailing of the services and the issues addressed by the application.

Figure 4.8. Wireframe of AgroAI Analyze Screen

This screen displays the chosen image from the Home Screen, the profile button and icon, the buttons with clear instructions to analyze Plant health or choose another image if the user is not content with the one they chose. The section at the bottom showed the crop problems addressed by the application.

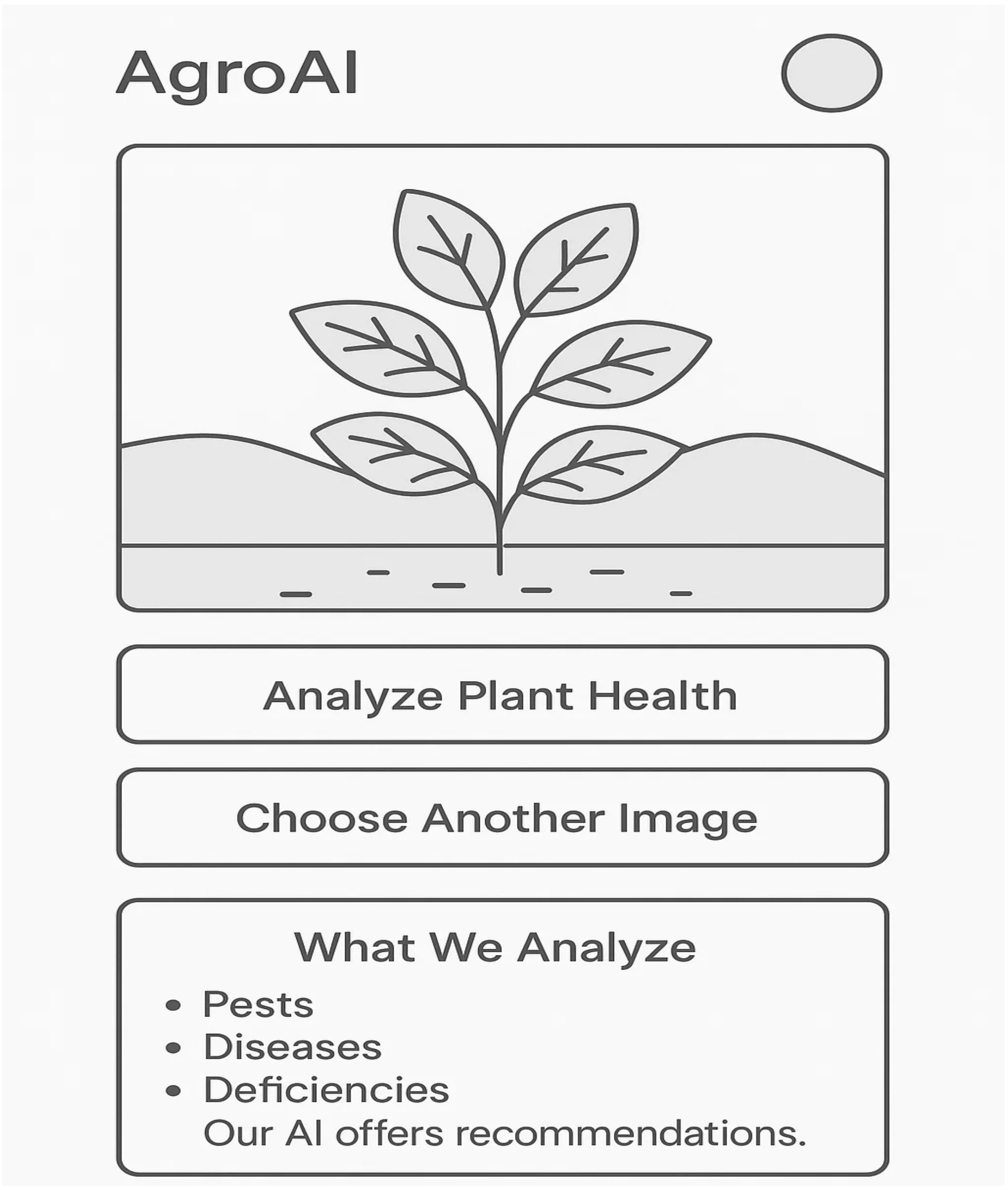
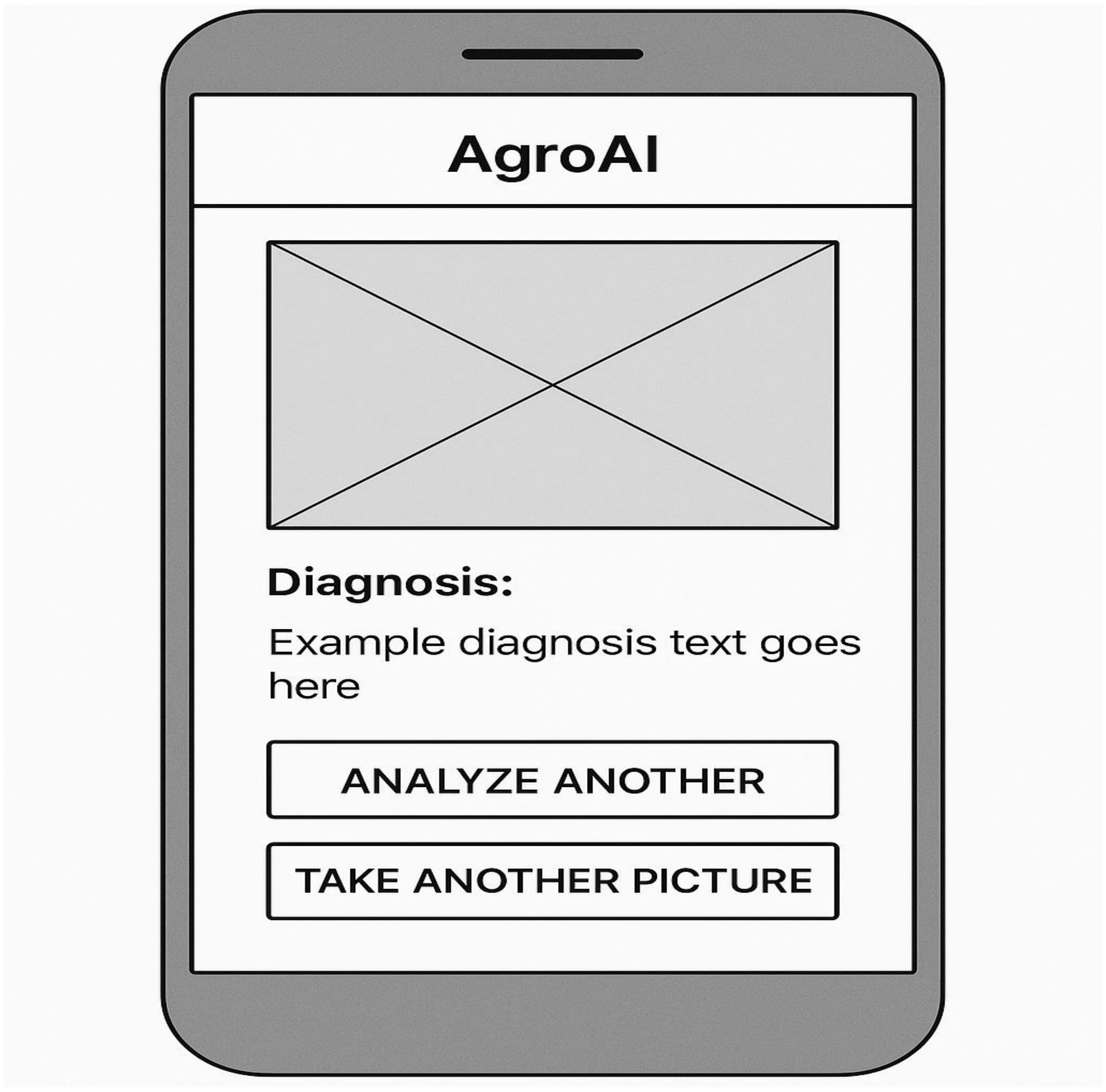


Figure 4.9. Wireframe of AgroAI Results Page

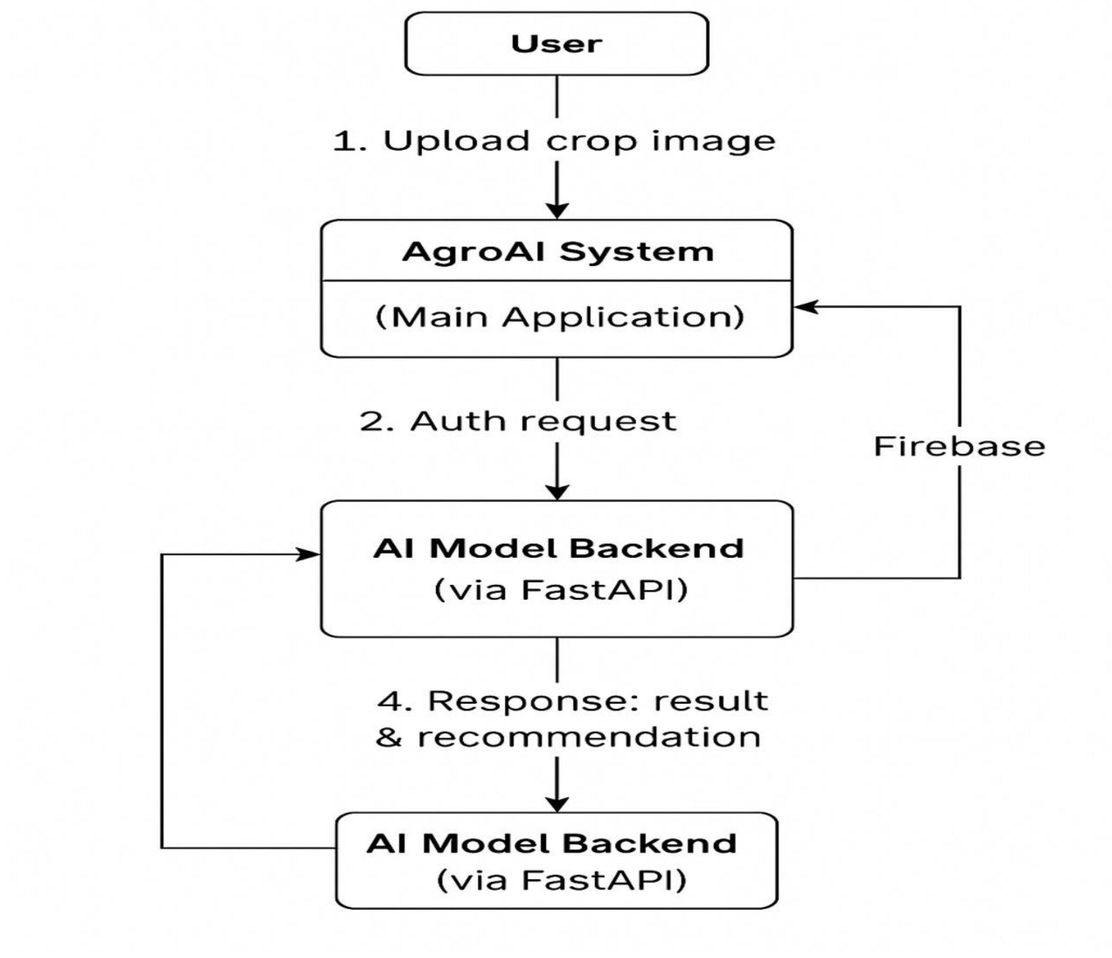
The result page displayed the uploaded image, the diagnosis results and a treatment recommendations. The page also included a button, 'Analyze Another Plant' that took the user back to the uploading screen to allow the farmers to upload and scan more plants.



Level 0 Data Flow Diagram

The level 0 DFD provides a high level overview of the AgroAI system. It presents the entire system as a single process showing how the user, Firebase and the AI model interact with the system. The diagram outlines how the crop images are submitted by the user, processed through the system and later through the AI model and the results are returned without detailing the internal operations.

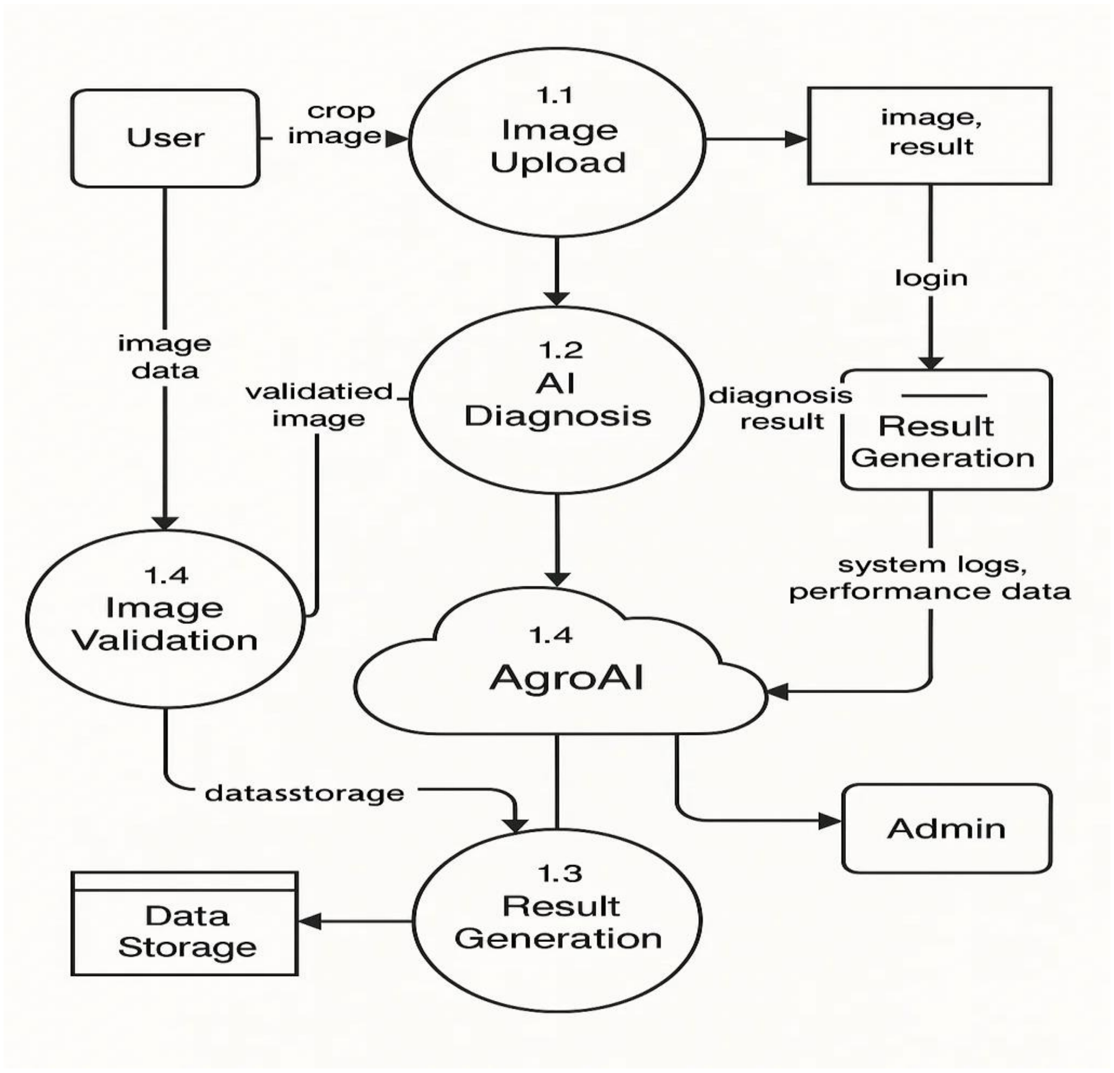
Figure 4.10. Level 0 Data Flow Diagram of the AgroAI diagnosis Process



Level 1 Data Flow Diagram

A level 1 DFD was developed to present and help understand the internal data processing of the in greater details. It consisted of the of primary functions of the AgroAI functions in processes like image capture and uploading, data validation, AI-based diagnosis, results generation, feedback delivery and communication between the various components of the system. The DFD diagram showed how an image uploaded was processed by the system, how the results were generated, how communications was handled and how data moved between the components.

Figure 4.11. Level 1 Data Flow Diagram of the AgroAI diagnosis Process



4.8.3. Physical Design

The physical design of AgroAI translated the logical design into real components and made them functional. This involved creating user interfaces, defining the database structures and putting in place system interactions structures.

4.8.3.1. User Interface Design

The user interface was developed using Jetpack Compose for Android. The layout and structure emphasized simplicity, easy use and navigation, and accessibility for farmers.Minimum screens were used to maintain simplicity.

The login screen included fields for the farmers to input their credentials, the emails or usernames and the password and buttons to register or reset forgotten credentials. The registration screen allowed users who didn’t have an account to enter their names, email, and password after which they were redirected to the home screen. The home screen presented two main options, the capture a crop image using phone camera or select one from gallery. Once an image was captured or uploaded, the farmer could review it and then choose the ‘Analyze Plant Health’ button which sent the image to the backend for processing and the AI model for diagnosis. The result screen displayed the crop name, diagnosed disease if any, the confidence level or the accuracy level, and the recommendation based on the diagnosis. Users could then choose to analyze another crop or return to Home page. The UI used contrasting colors and bigger font sizes to ensure visibility among the farmers as the application was designed to be used outdoors. Icons were also highly used to simplify the user interaction and mostly for users with low literacy levels.

4.8.3.2. Database Design

The physical database design of AgroAI focused on managing user credentials and related authentication data. AgroAI utilized Firebase for managing user authentication and cloud based data storage. Firebase authentication handled processes like user login, sign up and session management. This eliminated the need for manually setting up security features as Firebase provided secure, built in methods for managing user data and their accounts and data.

The database design focused on user accounts and information handling. During registration, user details such as user name, email address and password were captured through the registration mobile interface and stored securely using Firebase Authentication. The farmers passwords were were encrypted and securely automatically by Firebase, ensuring strong data protection and ensuring that no sensitive data was exposed to the application layer or to unauthorized personnel.

No crop images or analysis results were saved to allow for application simplicity. The images were processed in real time by the back end and the results were displayed directly to the users without being saved.The integration of Firebase allowed for successful and easy communication between the various components of the system.

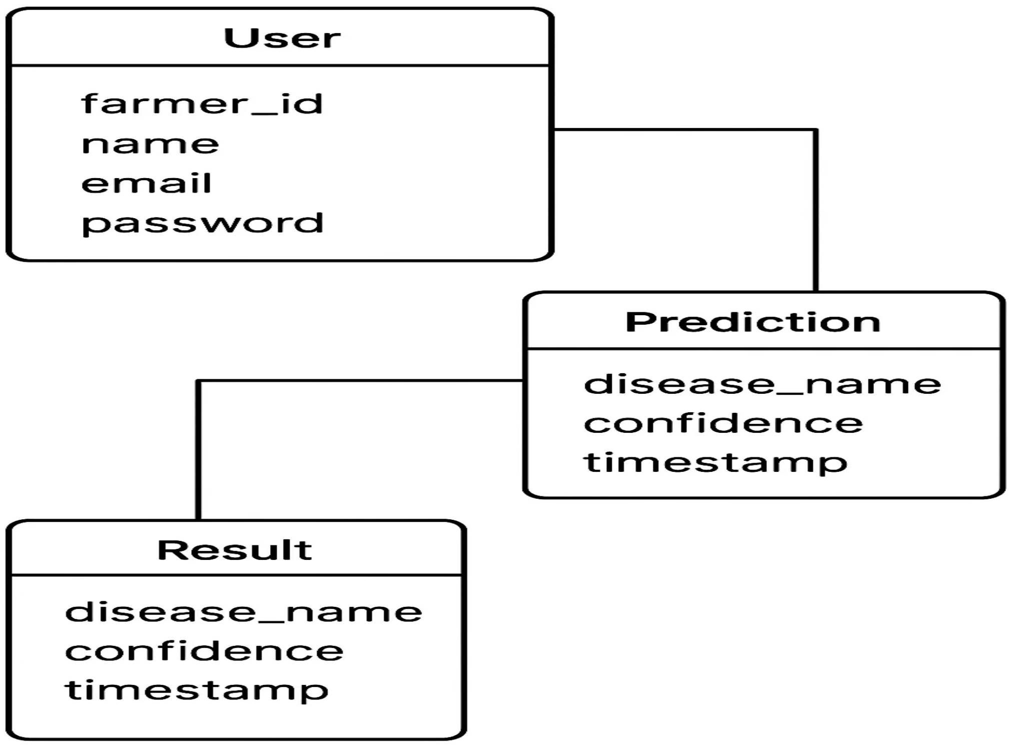
4.8.3.3. API and Backend Design

Most back end logic was implemented directly within the AgroAI android application using Jetpack Compose and Kotlin. This included screen navigation, image selection and user authentication through Firebase. FastAPI was used to enable communication between the back end and the trained AI model. The AI model was designed and developed separately and then wrapped using fastAPI. This API receives crop images from the app after processing through a POST request, processes the image and returns a JSON response with the crop type, pest or disease name, the accuracy or confidence level and fetches the best recommendation from the recommendations file. Security was handled using HTTPS and only authenticated users could access the back end services of the application. This approach kept the app lightweight and allowed the model to be updated independently and that that process did not affect the working of the application.

4.8.3.4 Entity Relationship Diagram

The Entity Relationship Diagram for AgroAI shows the structure of key data entities within the system. The main entity is the User, which stores information such as farmers ID, their name, email addresses, and their password. This is securely managed through Firebase Authentication. Other entities like Prediction or Result may be included to represent the outcomes of crop image analysis, storing details such as disease name, confidence score, and time stamp. The design follows normalization rules and supports data separation to improve maintainability and scaling in future updates.

Figure 4.12. AgroAI Entity Relationship Diagram of the AgroAI system



4.9. System Architecture

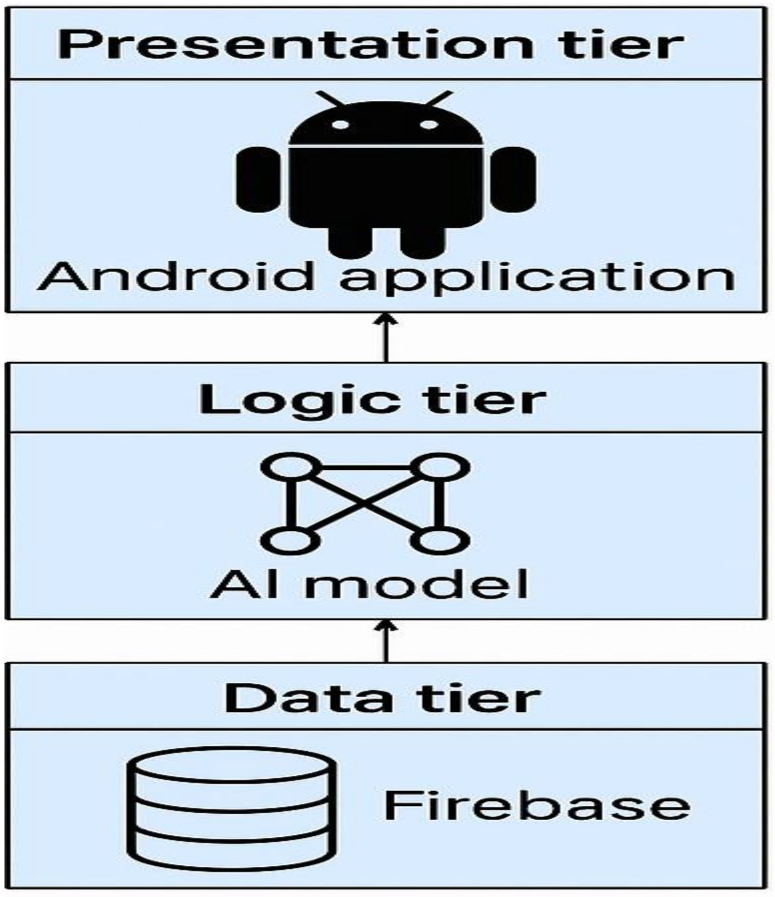
The AgroAI system was designed using a 3-tier architecture that follows the principles of both the N-tier and the client-server architecture. This system divided the system into various layers, the presentation layer, the logic layer and the data layer which was the back end.

The Client tier, the client, consisted of the android application developed using Jetpack Compose. This layer handled all the user interaction like the registration, the login and the image uploading and capture and displaying the results and the recommendations. The middle tier served as the business logic layer which was made possible through an API that acted as an interface between the client, and the AI model. It handled image processing, requests, and packaging of diagnosis results and recommendations into a well structured results to display to the user. The data tier was powered by Firebase. Firebase authentication managed secure user data such as emails and the passwords wile other features like the Firebase Cloud Storage could be implemented in future images and responses and history storage.

The client server model allowed separation of responsibilities of the components. The mobile application acted as the client, the API service and the other Firebase components acted as server side services. This structure ensured that the system was scalable in case of future research, secure and easy to maintain. Maintenance like updates could be done to one layer without affecting the working of the others.

The system design followed an object oriented approach, consistent with the methodology used in the analysis and design stages. This approach enabled simplified future feature additions such as keeping farmers analyzing history and offline capabilities. In addition, process oriented and data oriented techniques were applied, as reflected in the use of Data Flow Diagrams. This also gave a foundation for future enhancements.

Figure 4.13. AgroAI System Architecture



4.10. Design Techniques Alignment

The design of the AgroAI system used a both the object oriented and data oriented techniques to match the analysis technique.Data oriented techniques were used to show how information moves within the system. It explained the internal processes of the AgroAI system. The entity relationship diagram showed the data entities such as the users. Object oriented techniques helped define how the various components of the AgroAI system interacted. This was done through diagrams that captured the system behavior,structure and the systems activities.

By using both the object oriented and the data oriented techniques, the system ,maintained a strong link between analysis and design. This made the development more accurate and easier to maintain and also easier expansion in the future.

4.11. Normalized Database Assurance

The database structure of the AgroAI system was designed to follow normalization rules to avoid cases like data duplication and also to improve consistency. Each table focused on a single type of data such as a table focusing on user passwords or another focusing on email addresses. Although Firebase was used in the mobile app development rather than the traditional relational databases, the designed of the database still followed the normal logic way. For example, user details were kept separate from prediction results making it easier to manage the system and scaling in future. This approach ensured that theere was clean data handling and that the system was prepared for future research and features like offline capabilities and history and result storage.

4.12. Summary of the Design Vulnerable

The AgroAI design produced all the key components required for development. The systems boundaries were shown using a context diagram while a level 1 data flow diagram explained the main processes of the AgroAI system.Mock ups of the main user interfaces were also created including the registration page, the login page, Home page and the results page. All the designs matched the requirements gathered during analysis. The system was organized into components that were clearly defined including the mobile app, the back end API, and the Firebase services following a three-tier structure.

4.13 Hardware and Software Requirements

You can find detailed hardware and software specifications in Appendices C and D.

CHAPTER 5

SYSTEM CODE GENERATION AND TESTING, CONCLUSIONS AND RECOMMENDATIONS

5.1. Introduction

This chapter explains how the AgroAI system was implemented and tested to ensure it works correctly. It covers the development environment and processes, the tools and technologies used, the setup process, and the testing steps. It also discusses the challenges faced during implementation and how they were resolved. The purpose of this chapter is to show that the system was successfully built, meets its functional requirements, and works reliably in real-world conditions.

5.2. System Code Generation

This phase of AgroAI implementation started with the the development of the mobile application . The process was then followed by the integration of backend logic also coded using Jetpack compose and the trained AI model. The code generation process followed the specifications put in place and defined during the design phase and utilized the tools and platforms previously identified, such as Kotlin with Jetpack Compose, Firebase, FastAPI, PyTorch, Android studio and TensorFlow. Each part of the system, from the user interface to backend logic and the AI model, was implemented and tested separately, then combined to create a complete functional product. The implementation process used an iterative approach to ensure that each feature met user needs, worked reliably, and fit well with the others.

5.2.1 User Interface Development

The various components of the AgroAI user interface was built using Jetpack Compose, Android’s modern UI toolkit for creating native interfaces in a straightforward way. The design aimed to be simple, intuitive, and easy for farmers to use, as many users might not have much tech experience. The user interface components included the registration screen, login screen, home screen, image analysis screen, and results screen. All these were developed as composable functions in Jetpack Compose. Each screen has awesome readable font sizes, high contrast colors to ensure that the farmers are able to use them outdoors as intended, and very easy to understand icons. This ensured understanding and simplicity when being used outdoors and made it easier for users with low literacy levels. The NavController from Jetpack Compose's navigation library enabled the navigation between screens.

5.2.1.1 Home Screen

The home screen is the main screen and home for user activity after a farmer logs in. It offered various options but the main options being the capture a crop image using the phone camera, or to upload a crop image from the gallery. The home screen also included a logout icon and clear text headers to assist users, and an icon button that takes the farmers to their account. The layout used Column and Box composables along with Modifier.padding, fillMaxSize, Spacer function and align among other functions and composables to organize the interface components effectively and to ensure seamless and easy navigation and ease if use,. It focused on simplicity and clear action prompts.

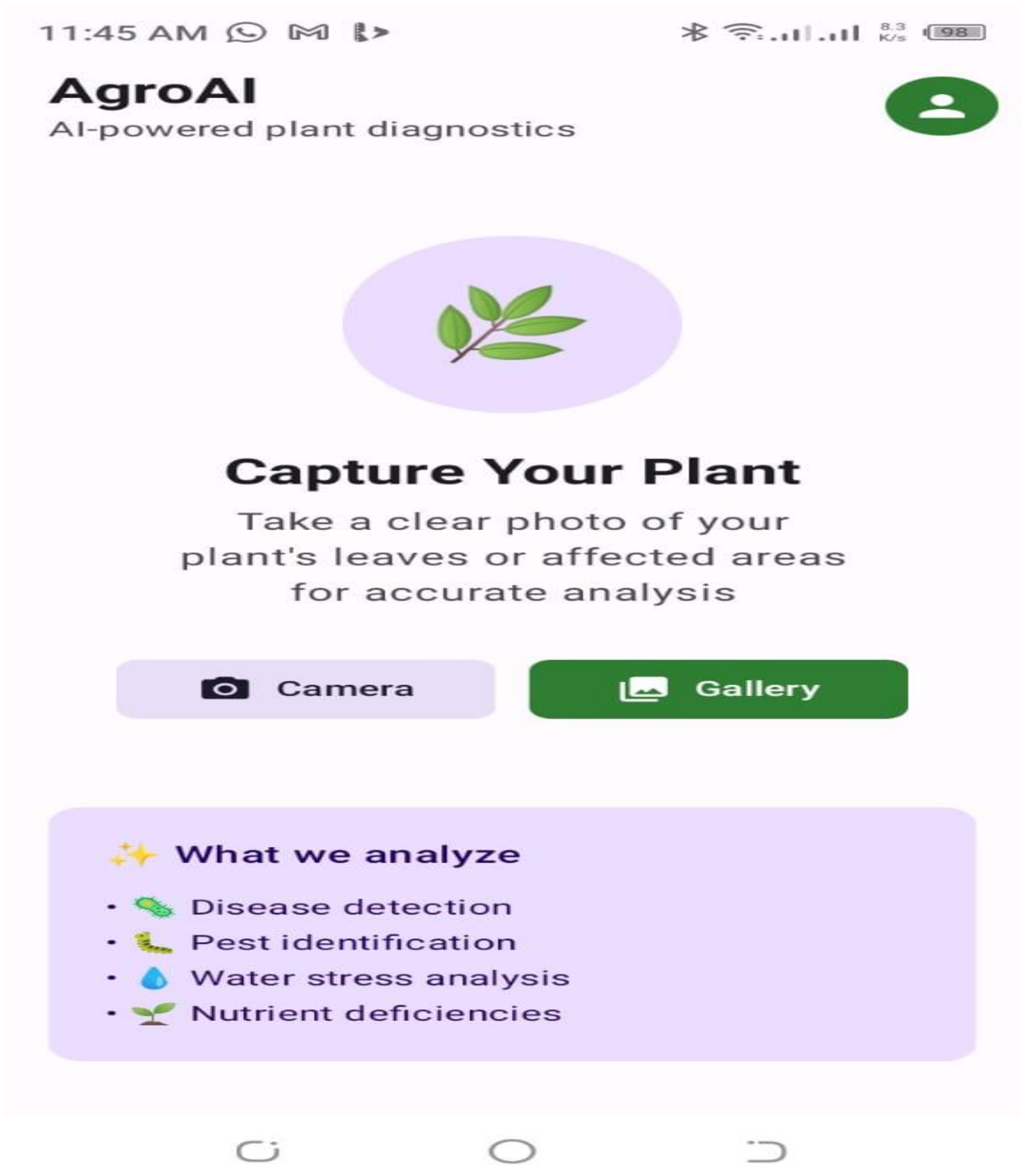
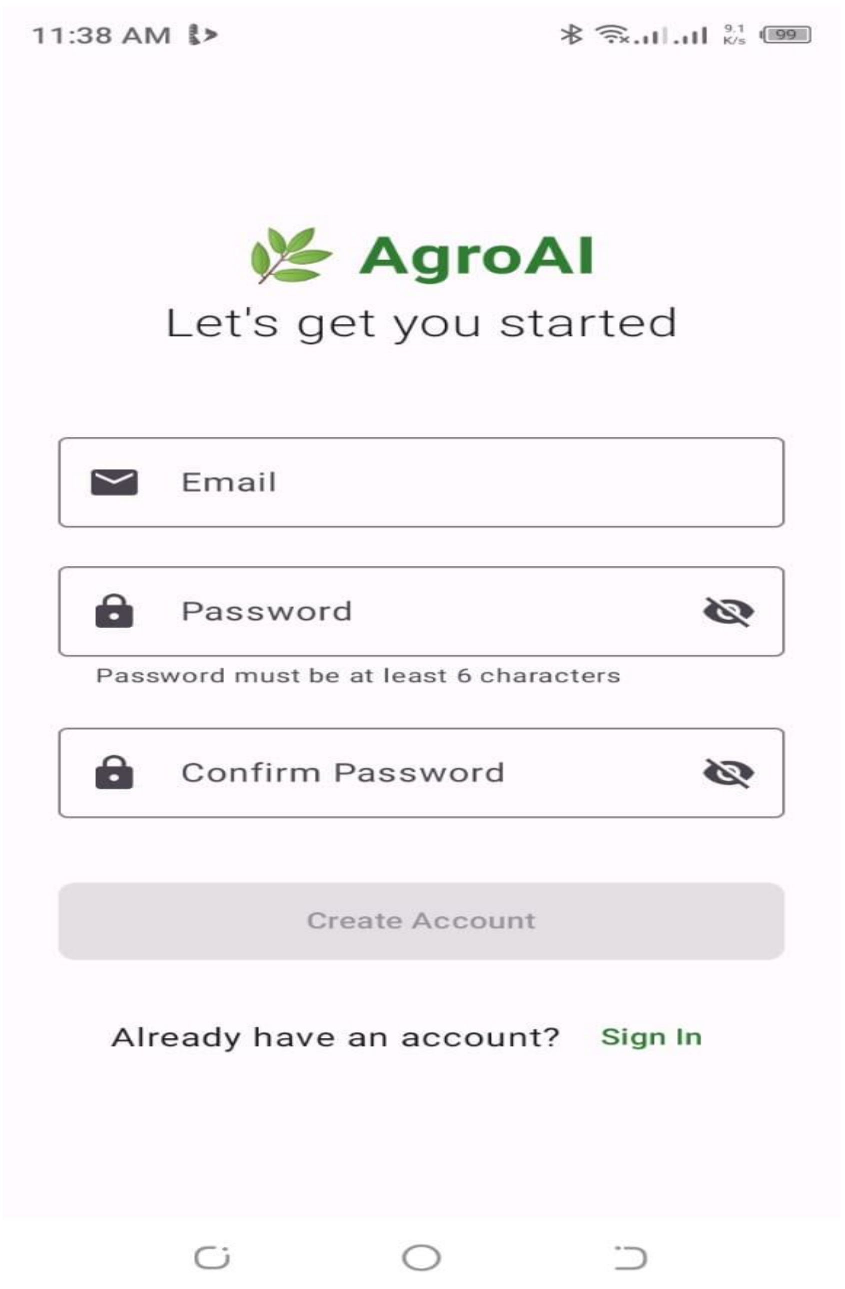

11:45 AM
AgroAI
AI-powered plant diagnostics
Capture Your Plant
Take a clear photo of your plant's leaves or affected areas for accurate analysis
Camera
Gallery
✨ What we analyze
• 🦠 Disease detection
• 🐛 Pest identification
• 💧 Water stress analysis
• 🌱 Nutrient deficiencies

Figure 5.1. AgroAI Home Page

5.2.1.2 Registration Screen

The registration screen was designed to enable new users to the AgroAI application to create an account before using the application. The screen was developed and implemented using Jetpack Compose and it consisted of input fields for email address, password and a confirmation of the password. The page also consisted of a Register button that upon clicking,triggered the Firebase authentication to create a new user account for the new farmer using the entered data. Basic validation was implemented to ensure that the data entered was complete and that one email address did not register for multiple accounts unless authorized by the application admin.In the event of successful user registration, the farmer was automatically redirected to the home screen.The design of this page ensured simplicity, ease of understanding, and large input fields with icons for ease of use during outdoors use. A navigation link was included for users to go back to the login screen if they already had an existing account.

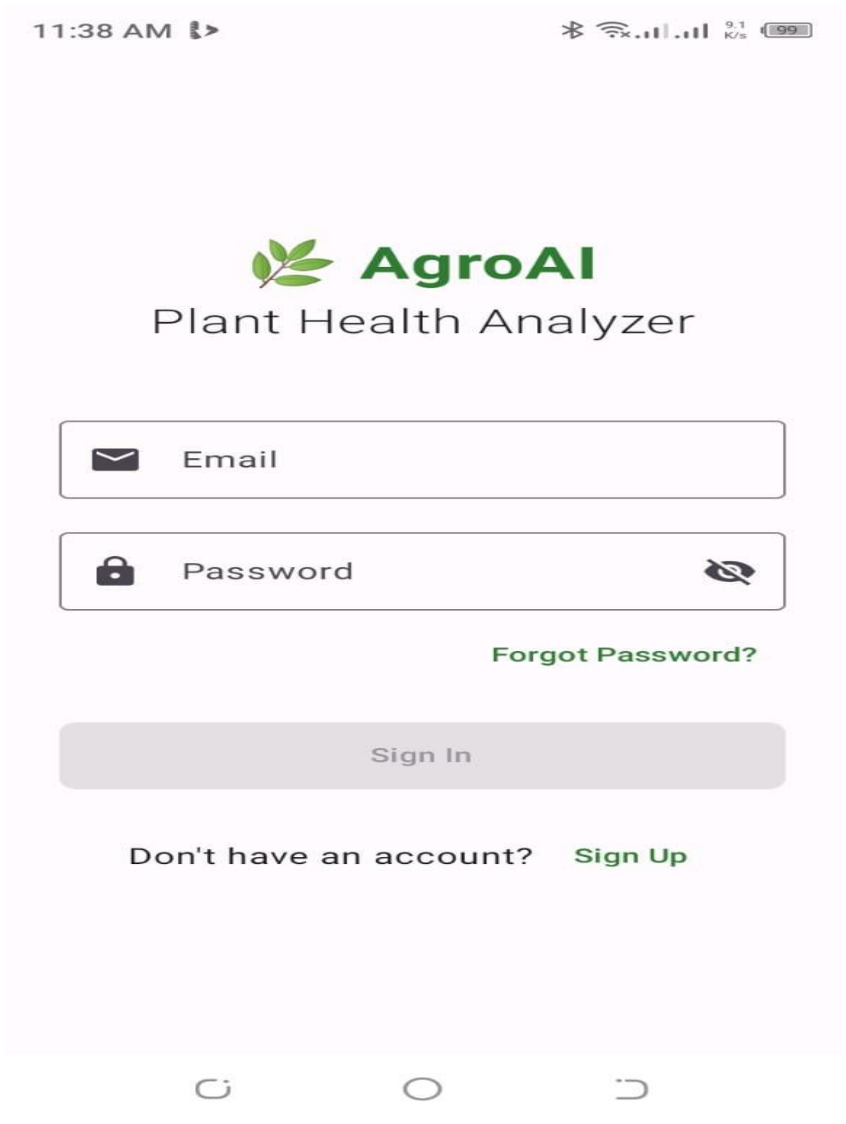
Figure 5.2. AgroAI Registration Screen



5.2.1.3 Login Screen

The login screen is responsible for authenticating existing users and those allowed to use the application. The login screen was developed using Jetpack Compose with fields for email and password and a login button. There is also a link for resetting a password in case forgotten and another ‘Sign Up’ link that redirects the farmers to register an account in case they haven’t. Once authenticated, the farmer is redirected to the Home Screen where crop analyzing and diagnosis can begin. The UI followed the set criteria with bold text, contrasting colors and direct and understandable instructions ensuring that the users can navigate and use with utmost ease.

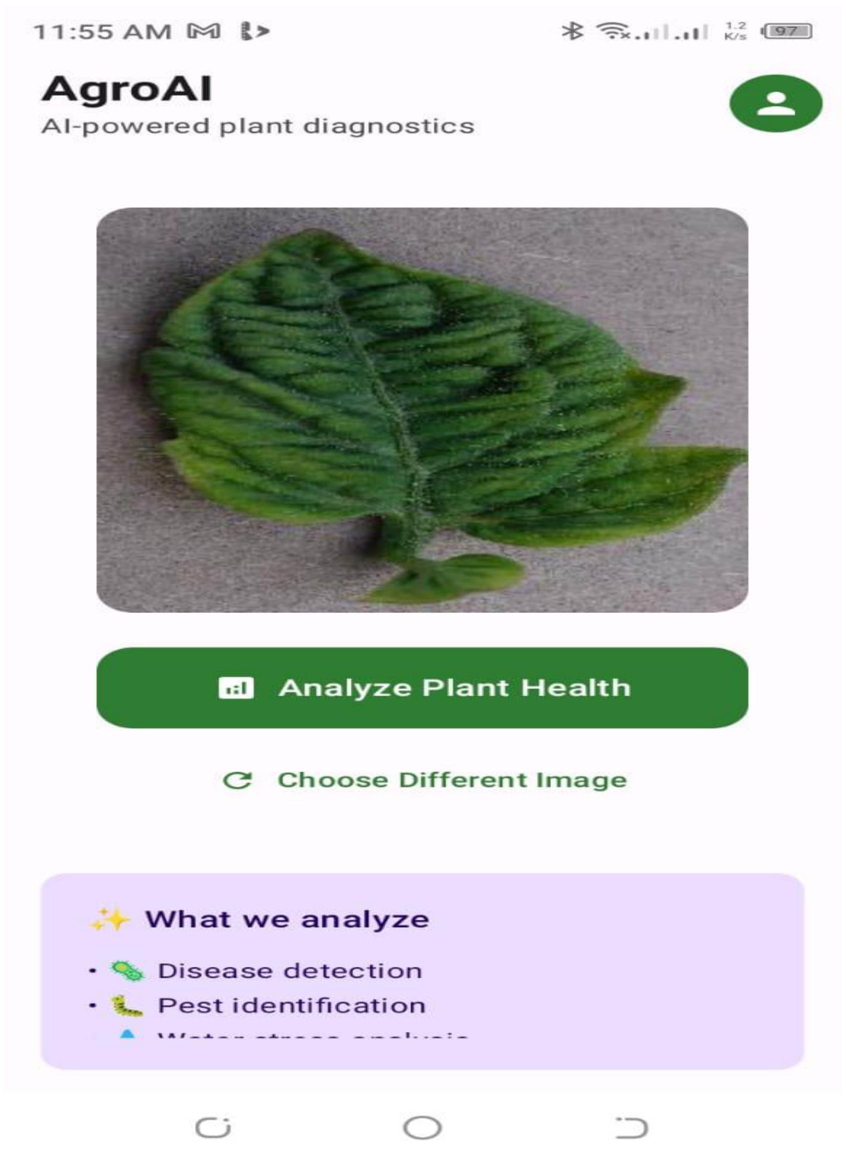
Figure 5.5. AgroAI Login Screen



5.2.1.4. Image Analysis Screen

After login, the users are taken to the Home Screen where they choose to either capture a crop image or upload from their phone gallery.Upon choosing, they are redirected to the Image Analysis Screen where the chosen image is displayed for review. The user also gets two choices, where they can decide to continue to analyze the selected image or choose another image.If they choose to continue and clicks the ‘Analyze Plant Health’ button, the image is submitted to the back end and to the model for processing and diagnosis.The layout used in this page was clean using icons and buttons to guide the user towards the next step.

Figure 5.6. AgroAI Analysis Screen

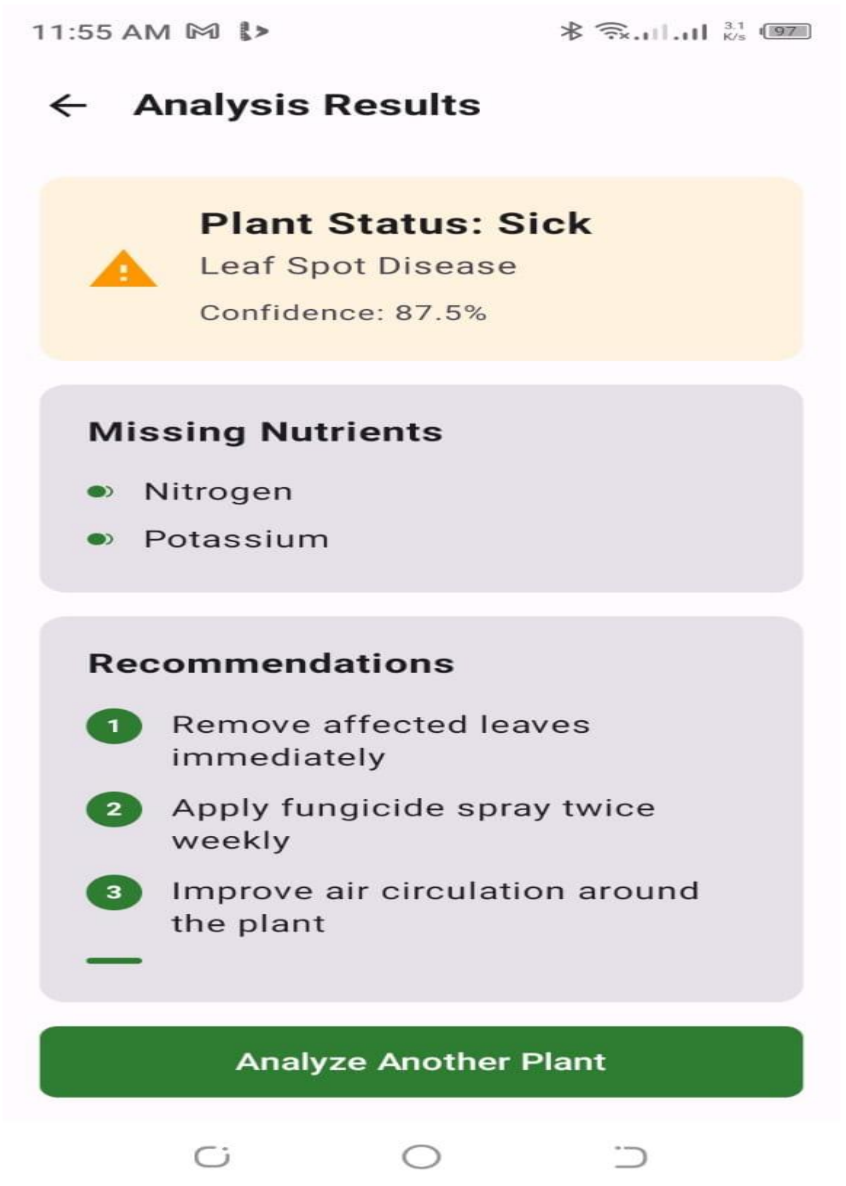


5.2.1.5. Results Screen

Once the processing and the diagnosis is complete, the user is redirected to the

Results Screen. This screen displays the Crop name, the diagnosed Crop Pest or disease if there was any, the level of confidence or accuracy score as a percentage and a recommendation based on the pest or disease diagnosed. The screen also has two buttons, one that redirects the user to Home Screen and the other that offers the choice to analyze another crop.The use of large texts and lively colors were used to ensure ease of use for the farmers and also for outdoor use.

Figure 5.7. AgroAI Results Screen



11:55 AM

Analysis Results

Plant Status: Sick
Leaf Spot Disease
Confidence: 87.5%

Missing Nutrients
- Nitrogen
- Potassium

Recommendations
1. Remove affected leaves immediately
2. Apply fungicide spray twice weekly
3. Improve air circulation around the plant

5.2.2. Firebase Authentication

Firebase Authentication was integrated into the AgroAI application to securely manage registration of the users, their login, and password recovery in case forgotten.

It offered a way to handle user accounts. Firebase was chosen for its seamless compatibility with Android, ease of use, and strong support for email and password sign in. After connecting the app to Firebase with the Google services.json configuration file, the system was set up to allow farmers to create accounts and sign in through simple interfaces built with Jetpack Compose. Once users registered successfully, they were automatically signed in and taken to the Home Screen. Firebase handled login by checking credentials. A password recovery option let users receive reset instructions via email, which improved usability. The logout feature cleared the session and directed users back to the login page.

Firebase streamlined AgroAI's authentication system by providing a secure backend service, simplifying farmers accounts management and ensuring reliable account handling throughout the application.

5.2.3. AI Model Training and Deployment

The AI model in AgroAI was created to classify plant diseases based on leaf images. Training used a curated dataset from the PlantVillage dataset and Agritech Analytics Limited, focusing on crops relevant to Kenya, such as maize, potato, tomato, and bell pepper. The images underwent preprocessing, which included resizing the images, normalization, and augmentation to improve generalization. Training took place using PyTorch and Tensorflow in Google Colab, which provided GPU support, higher RAM and access to cloud resources. This setup ensured efficient training without overloading local hardware. The model architecture relied on a Convolutional Neural Network (CNN), which is ideal for image classification tasks. After multiple training sessions and tries and adjusting various parameters and figures, the model reached a validation accuracy of about 98.54%. This result confirmed its strong ability to detect diseases accurately. To reduce its size and optimize loading on mobile devices, the trained model was saved in .pt, .pth, and a copy in .tflite format for deployment.

After training and validation, the model was integrated into the backend using a FastAPI server. This setup allowed the mobile app to send plant images to the server and receive predictions in real time. The backend API was deployed in a secure server environment and underwent extensive testing to ensure it handled requests quickly and reliably. This integration enabled the AgroAI system to offer smart, AI-powered diagnostics to farmers on their mobile phones, even in low-resource settings.

Table 5.1. Model Performance Summary

Metric	Value
Accuracy	98.54%
Precision	87.52%
Recall	87.47%
F1-Score	87.25%
Total Image Samples Used	26,980 images

5.3.4. Back End API Integration and Implementation

The backend API was built with FastAPI to let the mobile app send images and get diagnosis from the trained AI model and also match the recommendations. When a user uploads or takes a picture of a plant through the app, the backend gets the image, resizes it to 256x256 pixels, normalizes it, and processes it through the loaded tflite model. The API sends back the predicted class and confidence score in JSON format. This setup allows for real time diagnosis without needing the model to run directly on the device.

5.3. Testing

This section describes the approach and strategy used for testing the AgroAI system. It aimed to ensure that the developed application met its functional and non-functional requirements. This section is about the main methodology and types of testing considered during the system design phase. The main goal was to create a strong testing framework that would confirm the system's accuracy, simplicity in usability and reliability for smallholder farmers who were the intended users.

5.3.1. Testing Methodology

The testing of the AgroAI system was mainly planned and involved manual testing. This choice was based on the project's size, scope, and the application's focus on user interaction and real world environmental factors. Manual testing was considered sufficient and appropriate for verifying the user experience, visual accuracy, and determining the overall system behavior for the target users, the Kenyan smallholder farmers. This approach allowed for direct human interaction with the system, helping to identify user experience issues and real world challenges that automated tools might overlook.

To ensure thorough validation of the AgroAI system, various testing types were planned and included in the development lifecycle. Unit testing aimed to confirm the correctness of individual components or modules of the application in isolation. This included testing specific functions such as user authentication which were the registration, login, password reset functions, image capture and upload methods, and the internal logic of data processing before integration. Code segments and functions were tested regularly so as to ensure that they worked as intended based on their design specifications.

Integration Testing

After unit testing, integration testing was planned to verify that different services of the AgroAI system interacted correctly when combined or integrated. This involved testing the smooth flow of data and control between the mobile application's front end, the FastAPI backend, the AI model and Firebase services. Key integration points, such as sending an image for diagnosis and receiving a recommendation, would be tested through manual execution of scenarios involving multiple interconnected components to find interface defects and ensure smooth communication.

System Testing

System testing was designed to evaluate the complete, integrated AgroAI system against its outlined functional and non functional requirements. This thorough testing would adopt real world use scenarios, assessing the application's stability, performance like accuracy and the response time, reliability, and ease of usability as a whole. This phase included end to end manual testing covering all user flows and system functions, including cases like error handling.

User Acceptance Testing

UAT was an essential planned phase aimed at confirming that the AgroAI application met the real needs and expectations of smallholder farmers and addressed their actual problems. This involved providing the prototype, CropScan, to end users who were smallholder farmers in Kinangop Constituency in Nyandarua County to use and interact with the prototype and get the clear understanding of the system, and determine whether the system would solve their problems in their natural farming environment. Their feedback on ease of use, accuracy of diagnosis, practicality of recommendations, and overall satisfaction was collected. UAT methodology included direct observation, structured questionnaires as detailed in Appendix E, and informal interviews to gather qualitative and quantitative feedback from farmers. This iterative feedback loop was crucial for refining the application to promote high user adoption and practical utility.

5.3.3. Test Environment and Resources

The testing environment was carefully planned to reflect the real world conditions of the target users, who were small holder farmers in Kenya. Hardware for testing would mainly consist of Android smartphones, Android 7+ with a minimum of 2GB RAM, to match devices commonly available to smallholder farmers. On the software side, the testing environment included the deployed Android application, the FastAPI backend server hosting the AI model, and Firebase services for authentication and data management. For data, a dataset of labeled crop images, including those from AgriTech Analytics and Kaggle PlantVillage dataset, was used to train and validate the AI model's diagnostic accuracy. Locally collected images would also be included to ensure relevance to Kenyan farming contexts. This planned testing strategy aimed to identify and resolve issues systematically throughout the development lifecycle. It sought to ensure that the final AgroAI AI model and the application were strong, accurate, and genuinely helpful to they intended users.

Performance Testing

The performance testing of the AI model was a key step in developing the AgroAI system. It directly impacted the accuracy and reliability of diagnosing crop diseases and pests. A classification report was used to evaluate the model. This report showed its precision, recall, and F1-score across different crop conditions. These metrics are vital for understanding how well the model can tell apart healthy plants from those affected by specific diseases or pests.

The classification report revealed that the model had an overall accuracy of about 87.47%. This percentage shows how well the model can identify crop health issues based on input images. The precision and recall values for individual classes varied, highlighting the model's strengths and weaknesses in diagnosing specific conditions. For example, the model performed well for diseases like "Pepper bell Bacterial spot" and "Tomato Tomato Yellow Leaf Curl Virus," showing high precision and recall. However, some diseases, such as "Tomato Early blight," had lower performance metrics, indicating areas that need improvement.

5.4 Conclusions

The AgroAI system effectively dived into the main challenges faced by smallholder farmers in Kinangop Constituency, Nyandarua county. It handed a simple, easy to use mobile operation for diagnosing crop pests and conditions. By using a locally trained AI model, FastAPI back end and a very friendly user interface, the system allowed for identifying conditions in real time with their smartphone cameras and with ease, simplicity and at no expenditure. Testing and feedback showed that the system met its functional conditions by directly detecting and diagnosing diseases and pests from images and giving helpful, localized recommendations. The research also met important non functional prospects like usability, performance, and trust, and ease of use in areas where farmers have low literacy and education levels. Although there is room for enhancement in areas like automated feedback circles and offline capability, the system has shown great potential in helping smallholder farmer by perfecting crop health, and reducing losses from undiagnosed infections and wrong diagnosis.

5.5. Limitations

The AgroAI system met its main intended goals, but the development process faced some problems along the way that affected the overall implementation. First, limited financial resources restricted the purchase of cloud infrastructure and professional testing tools. This led to reliance on manual testing and local environments. Second, while the AI model was trained on a carefully selected and diverse PlantVillage and AgriTech Analytics Limited datasets, high quality local image data for some rare diseases was hard to find. This may have impacted the model's ability to generalize in Kenyan real world scenarios and practices. Another issue was related to hardware. Testing and performance optimization were mainly done on mid range Android smartphones, which may not accurately represent performance on low end devices that some farmers use. Finally, although user feedback was gathered during informal testing, the project did not conduct a formal large scale pilot study due to time and logistical challenges. A large scale pilot would have allowed for study and research into the projects long term effects. Also, the issue of bulk users testing was not implemented where dozens of people use the system at the same time to test how much can it take.

5.6. Overview of the Chapter

This chapter described how the AgroAI system was built and tested grounded on the specifications and design from earlier chapters. It outlined the development process, including the use of tools like Kotlin and Jetpack Compose for the Android front end development, FastAPI for the backend, Firebase for storage operations, and TensorFlow and PyTorch for training the AgroAI model. The system components were implemented and integrated successfully, creating a functional crop disease diagnostic application for smallholder farmers in Kenya.

Testing followed to confirm the system's functionality, accuracy, and usability. Different types of testing were performed, including unit testing, performance testing of the AI model, and User Acceptance Testing. The results showed that the system met its objectives and worked reliably in real-world scenarios.

Manual testing was mainly used throughout the project because of the scope and budget. Although there are advanced testing tools like Appium and Selenium available, the AgroAI system was effectively tested using manual methods. This approach allowed for real user interaction and accurate validation of functionality.

The successful completion of this chapter shows that the AgroAI system was implemented as planned and tested thoroughly. It ensures that it provides value to its users smallholder farmers by offering reliable crop health diagnosis and practical recommendations.

5.7 Recommendations

Based on the successful use and evaluation of the AgroAI system, several recommendations were made for its future development and practical use. First, future versions of the system should include offline functionality and capabilities so as to be able to handle and tackle network issues in rural farming areas of Kenya. This would provide continuous access to diagnostic services, even in places with unreliable internet. Second, adding real time weather data and geo location features could further tailor the recommendations for farmers, making them more relevant and effective. Additionally, it would be good to expand the crop and disease database to include a broader range of locally grown crops beyond what is currently available.Moving into industries such as horticulture and farming of flowers would also be a huge step towards solving the pests and diseases issue in Kenya and the globe at large. This would increase the tool's usefulness across different agricultural regions. To encourage use, organized training and awareness programs should be set up to help farmers understand the system's features and benefits. Finally, working with agricultural institutions and county governments could help grow the platform, build trust, and improve the long-term viability of AgroAI as an essential tool in precision agriculture for smallholder farmers.

REFERENCES

Aker, J. C. (2011). Dial “A” for agriculture: a review of information and communication technologies for agricultural extension in developing countries. *Agricultural Economics, 42*(6), 631–647. https://doi.org/10.1111/j.1574-0862.2011.00545.x

AZoLifeSciences. (2022, August 15). *Excess fertilizer use and soil health*. https://www.azolifesciences.com/article/Excess-Fertilizer-Use-and-Soil-Health.aspx

Barbedo, J. G. (2018). Factors influencing the use of deep learning for plant disease recognition. *Biosystems Engineering*, *172*, 84–91. https://doi.org/10.1016/j.biosystemseng.2018.05.013

CABI. (2022, February 22). *Addressing scale insect threats in Kenya - CABI.org*. CABI.org. https://www.cabi.org/projects/addressing-scale-insect-threats-in-kenya/

Coelho, F. C., Coelho, E. M., & Egerer, M. (2017a). Local food: benefits and failings due to modern agriculture. *Scientia Agricola*, *75*(1), 84–94. https://doi.org/10.1590/1678-992x-2015-0439

Coelho, F. C., Coelho, E. M., & Egerer, M. (2017b). Local food: benefits and failings due to modern agriculture. *Scientia Agricola*, *75*(1), 84–94. https://doi.org/10.1590/1678-992x-2015-0439

Environmental implications of excess fertilizer and manure on water quality. (2023, February 21). NDSU Agriculture. https://www.ndsu.edu/agriculture/extension/publications/environmental-implications-excess-fertilizer-and-manure-water-quality

Facini, A., & Facini, A. (2024, June 13). Pest and pathogen threats to food security - The Council on Strategic Risks. *The Council on Strategic Risks - Anticipating, Analyzing, and Addressing Systemic Risks*. https://councilonstrategicrisks.org/2024/06/13/pest-and-pathogen-threats-to-food-security/

Ferentinos, K. P. (2018a). Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture*, *145*, 311–318. https://doi.org/10.1016/j.compag.2018.01.009

Ferentinos, K. P. (2018b). Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture*, *145*, 311–318. https://doi.org/10.1016/j.compag.2018.01.009

Okube, O. T., Wanjiru, M., & Andemariam, W. (2022). Magnitude and Determinants of Undernutrition among Pregnant Women Attending a Public Hospital in Kenya. *Open Journal of Obstetrics and Gynecology*, *12*(06), 541–561. https://doi.org/10.4236/ojog.2022.126048

Ramcharan, A., Baranowski, K., McCloskey, P., Ahmed, B., Legg, J., & Hughes, D. P. (2017). Deep Learning for Image-Based Cassava Disease Detection. *Frontiers in Plant Science*, *8*. https://doi.org/10.3389/fpls.2017.01852

THE STATE OF FOOD AND AGRICULTURE 2001. (n.d.). https://www.fao.org/4/x9800e/x9800e14.htm

Venkatesh, N., Morris, N., Davis, N., & Davis, N. (2003). User acceptance of information Technology: toward a unified view. *MIS Quarterly*, *27*(3), 425. https://doi.org/10.2307/30036540

APPENDICES

Appendix A: Proposed Budget Table

Table 1.1: Proposed Budget for the Development of AgroAI, Author, 2025

Item	Description	Estimate Cost(kshs)
Internet/ Data bundles	Back end testing, syncing, testing, online documentation, communication, model upload, app development.	2,400
Hosting (Firebase, API)	Hosting for back end, AI model API, and database usage	1,700
Transport for field testing	Two trips to Engineer and Matundura, Kinangop constituency using public transport.	1,400
Printing and resources	Final report printing, binding and cover, printing questionnaires and feedback forms	1,400
Laptop	For AI model development and training, mobile application development, etc	0
Mobile phone	Testing mobile application	0
AI model training resources	Local training - Google colab GPU pro	500
Power/Back up access	Solar or fuel cost for consistent development and training	400

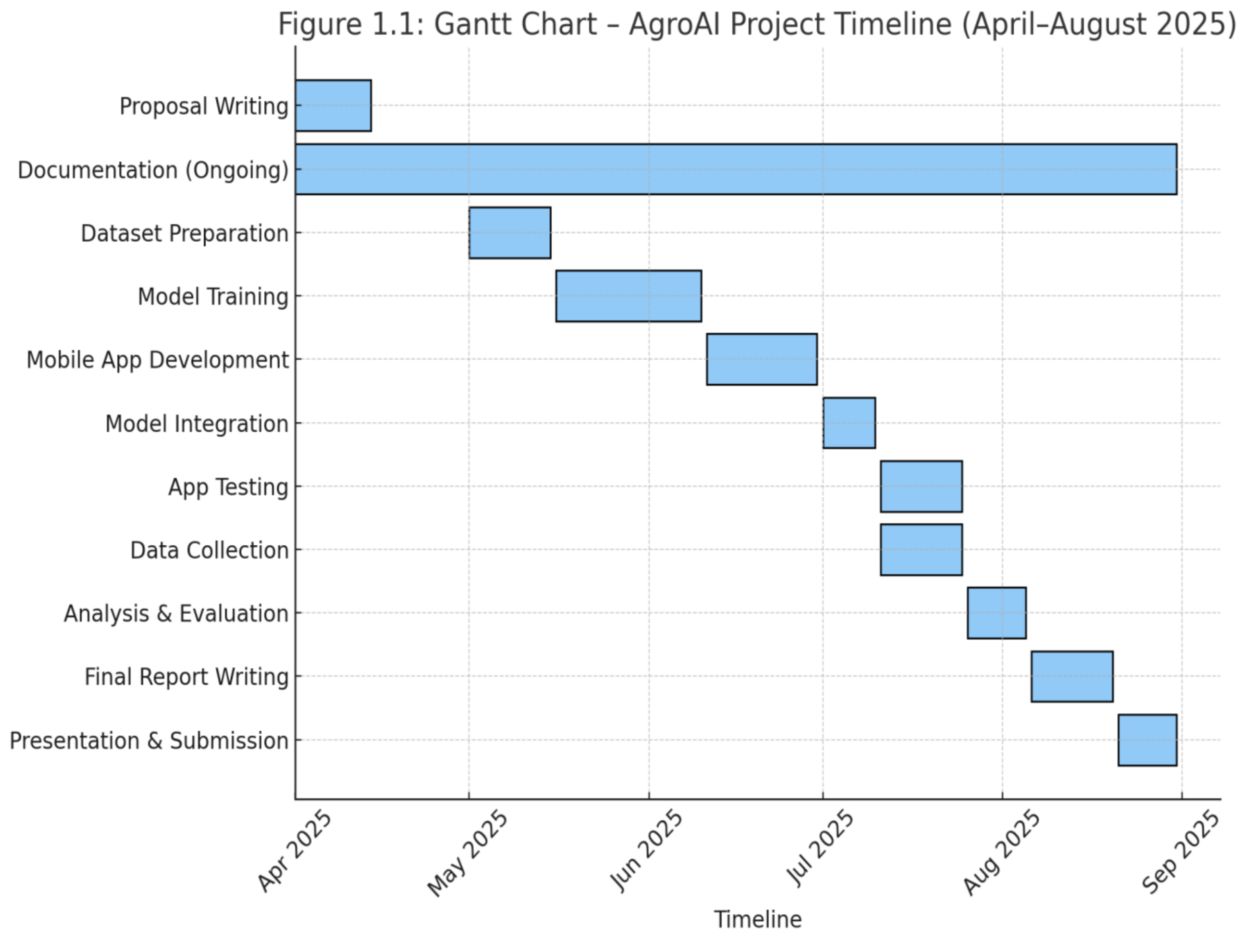
Developer tools	Paid plugin, API access, etc	600
Miscallenous	Farmer appreciation, USB drives, snacks,Airtime, unforeseen minor expenses	1,600
Total cost		**10,000**

1.10.2 Justification of the Budget

The total budget for this project is 10,000 Kenyan shillings, which supports the activities taking place in this project, both research and development activities. Internet bundles will be used to login into the internet to download datasets, integration of the AI trained model, testing the mobile application, and communication between the parties involved. The transport costs will cover the travel to Kinangop, twice, for field testing and data and feedback collection. Hosting will cover the costs of APIs hosting the application for access by smallholder farmers, data synchronization and AI model deployment. Printing and stationery will support the production of research instruments such as questionnaires, binders, printing of final project and binding process. It will also cover printing of printing forms, and purchases of other stationery like pens and sticky notes. There will be no laptop and mobile phone since they are already available. AI model training costs will account for power, limited cloud computing and Google Colab GPU pro access. Lastly, the miscellaneous cost will provide for unexpected small costs, such as airtime, and tokens for the participant

Appendix B: Project TimeLine Gantt Chart

Fig 1.1: Gantt Chart - AgroAI Project Timelime, Author, 2025



Appendix C: Hardware Requirements Table

Table 1.2: Hardware Requirements

Hardware	**Description/Specification**
Laptop	8GB RAM, Intel core i5, 256GB SSD, For model training, coding, and development
Smart phone (Testing)	Android 10+, 2GB RAM(minimum) - For app installation and user testing
Internet Access	Stable 4G or WI-FI - For syncing data, cloud access, API testing
Back up power	Temporary solar - For continued work in case f outages

Appendix D: Software Requirements Table

Table 1.3: Software Requirements, Author, 2025

Software	**Purpose**
Python 3.10 +	For AI model development and training
Tensorflow and Keras	Deep learning libraries used for model training
Android studio	Mobile application development using Kotlin and Jetpack Compose
Firebase	Back end services, real-time database and authentication
Fast API	Building API for model communication and back end communication.
Google colab	GPU support during training
Git/ GitHub	Version control and code backup
WPS office / Google Docs/ Word	Proposal writing, project documentation and report writing
Powerpoint	Presentation preparation

Appendix E: Sample Data Collection Questionnaire

Research Title: Development of AgroAI: An AI-Powered Mobile Application for Crop Disease and Pest Detection

Location: Kinangop Constituency, Nyandarua County
Date: _______________
Questionnaire ID: _______________

INFORMED CONSENT

This questionnaire is part of a research study to develop and test an AI-powered mobile application for crop disease detection. Your participation is voluntary, and all information provided will be kept confidential and used solely for academic purposes. The questionnaire will take approximately 8-15 minutes to complete.

Do you consent to participate in this study? ☐ Yes ☐ No

Participant Signature: _____________________ **Date:** ___________

SECTION 1: DEMOGRAPHIC INFORMATION

1.1 Age: _______ years

1.2 Gender:
☐ Male
☐ Female
☐ Prefer not to say

1.3 Highest level of education completed:
☐ No formal education
☐ Primary school

☐ Secondary school

☐ Certificate/Diploma

☐ University degree

☐ Other: ____________________

1.4 Years of farming experience: _______ years

1.5 Farm size: _______ acres

1.6 Main crops grown (select all that apply):

☐ Maize

☐ Beans

☐ Potatoes

☐ Cabbages

☐ Tomatoes

☐ Carrots

☐ Onions

☐ Other: ____________________

1.7 Primary source of farming income:

☐ Crop sales

☐ Livestock

☐ Mixed farming

☐ Other: ____________________

SECTION 2: CURRENT FARMING PRACTICES

2.1 How do you currently identify crop diseases or pests?

☐ Personal experience

☐ Ask other farmers

☐ Visit agricultural extension officer

☐ Internet/online resources

☐ Agricultural books/manuals

☐ Other: ____________________

2.2 How often do you experience crop diseases or pest problems?

☐ Very frequently (monthly)

☐ Frequently (2-3 times per season)

☐ Occasionally (once per season)

☐ Rarely (once per year)

☐ Never

2.3 When you have crop problems, how long does it typically take to get help?

☐ Same day

☐ 1-3 days

☐ 1 week

☐ More than 1 week

☐ Never get help

2.4 What is your biggest challenge in crop management?

☐ Identifying diseases/pests

☐ Getting treatment advice

☐ Cost of treatments

☐ Access to agricultural experts

☐ Weather conditions

☐ Other: ____________________

SECTION 3: TECHNOLOGY USAGE

3.1 Do you own a smartphone?

☐ Yes

☐ No (If No, skip to Section 4)

3.2 How long have you owned a smartphone?

☐ Less than 1 year

☐ 1-2 years

☐ 3-5 years

☐ More than 5 years

3.3 How often do you use mobile applications?

☐ Daily

☐ Several times per week

☐ Weekly

☐ Monthly

☐ Rarely

☐ Never

3.4 Have you ever used any farming-related mobile applications?

☐ Yes

☐ No

If Yes, please specify: ___________________________________

3.5 What is your main barrier to using mobile applications? (Select all that apply)

☐ Lack of technical knowledge

☐ Language barriers

☐ Internet connectivity issues

☐ Cost of data

☐ Applications are too complicated

☐ No barriers

☐ Other: ___________________

APPENDIX F: AGROAI USER FEEDBACK FORM

Research Title: Development of AgroAI: An AI-Powered Mobile Application for Crop Disease and Pest Detection

Form Type: After Use Feedback Collection
Date: _______________
Session ID: _______________
Location: Kinangop Constituency, Nyandarua County

USER INFORMATION

Participant ID: _______________
Name (Optional): _____________________________
Contact (Optional): ___________________________
Date of App Testing: _______________

SECTION 1: SESSION DETAILS

1.1 Crop type tested: ___________________________

1.2 Issue identified by farmer: ___________________________

1.3 Time taken to complete diagnosis:
☐ Less than 1 minute
☐ 1-2 minutes
☐ 3-5 minutes
☐ More than 5 minutes

1.4 Number of photos taken: ___________

1.5 App diagnosis result: ___________________________

1.6 Treatment recommendation provided:

☐ Yes

☐ No

☐ Partially

SECTION 2: IMMEDIATE EXPERIENCE FEEDBACK

2.1 How was your first impression of the app?

☐ Excellent - Very impressed

☐ Good - Positive impression

☐ Neutral - No strong feeling

☐ Poor - Disappointed

☐ Very Poor - Frustrated

2.2 Was the app interface intuitive?

☐ Very intuitive - Found everything easily

☐ Intuitive - Minor confusion

☐ Somewhat intuitive - Needed some help

☐ Not intuitive - Needed significant help

☐ Very confusing - Could not navigate

2.3 How clear were the photo-taking instructions?

☐ Very clear - Understood immediately

☐ Clear - Understood with minimal effort

☐ Somewhat clear - Needed clarification

☐ Unclear - Required assistance

☐ Very unclear - Could not follow

2.4 Rate the app's response time:

☐ Very fast (instant)

☐ Fast (within 5 seconds)

☐ Acceptable (5-15 seconds)

☐ Slow (15-30 seconds)

☐ Very slow (over 30 seconds)

SECTION 3: ACCURACY AND USEFULNESS

3.1 How accurate was the diagnosis compared to your expectation?

☐ 100% accurate - Exactly what I expected

☐ Very accurate - Close to my expectation

☐ Moderately accurate - Somewhat helpful

☐ Inaccurate - Not what I expected

☐ Completely wrong - Totally different

3.2 Were the treatment recommendations practical?

☐ Very practical - Easy to implement

☐ Practical - Can be implemented

☐ Somewhat practical - Some challenges

☐ Impractical - Difficult to implement

☐ Not applicable - No recommendations given

3.3 Did the app provide information you didn't know before?

☐ Yes, completely new information

☐ Yes, some new information

☐ Yes, but mostly familiar

☐ No, nothing new

☐ Information was incorrect

3.4 How helpful were the preventive measures suggested?

☐ Very helpful - Will definitely use

☐ Helpful - Will likely use

☐ Somewhat helpful - Might use

☐ Not helpful - Won't use

☐ No preventive measures provided

SECTION 4: USABILITY ASSESSMENT

4.1 Rate the following aspects (1 = Very Poor, 5 = Excellent):

Aspect	**1**	**2**	**3**	**4**	**5**
Ease of taking photos	☐	☐	☐	☐	☐
Image quality guidance	☐	☐	☐	☐	☐
Loading/processing speed	☐	☐	☐	☐	☐
Result presentation	☐	☐	☐	☐	☐
Recommendation clarity	☐	☐	☐	☐	☐
Overall user experience	☐	☐	☐	☐	☐

4.2 Did you encounter any technical problems?

☐ No problems

☐ Minor issues (didn't affect usage)

☐ Moderate issues (slowed down usage)

☐ Major issues (significantly affected usage)

☐ Severe issues (couldn't complete task)

If yes, please describe: ________________________________

4.3 Was the app language appropriate for you?

☐ Perfect - Understood everything

☐ Good - Understood most things

☐ Acceptable - Understood key points

☐ Poor - Struggled to understand

☐ Very poor - Could not understand

SECTION 5: COMPARISON AND VALUE

5.1 Compared to your usual method of identifying crop problems, this app is:

☐ Much better

☐ Better

☐ About the same

☐ Worse

☐ Much worse

5.2 Would you trust the app's diagnosis enough to act on it?

☐ Yes, completely trust it

☐ Yes, but would seek confirmation

☐ Maybe, depends on the situation

☐ No, would need other sources

☐ No, would not trust it

5.3 If this app were available for free, how often would you use it?

☐ Daily

☐ Weekly

☐ Monthly

☐ Only when problems occur

☐ Rarely or never

5.4 What would you be willing to pay for this app per month?

☐ Nothing - should be free

☐ Ksh 50-100

☐ Ksh 101-200

☐ Ksh 201-500

☐ More than Ksh 500

SECTION 6: IMPROVEMENT SUGGESTIONS

6.1 What frustrated you most about using the app?

6.2 What impressed you most about the app?

6.3 What features would you like to see added?

☐ Voice instructions

☐ Local language support

☐ Offline capability

☐ Weather information

☐ Market price information

☐ Farming calendar

☐ Expert consultation

☐ Other: ___________________________

6.4 How could the app better serve farmers like you?

SECTION 7: BEHAVIORAL INTENTION

7.1 How likely are you to recommend this app to other farmers?

☐ Very likely (9-10 out of 10)

☐ Likely (7-8 out of 10)

☐ Neutral (5-6 out of 10)

☐ Unlikely (3-4 out of 10)

☐ Very unlikely (1-2 out of 10)

7.2 Would you like to participate in future testing of app updates?

☐ Yes, very interested

☐ Yes, somewhat interested

☐ Maybe

☐ Probably not

☐ No, not interested

7.3 Can we contact you for follow-up questions about your experience?

☐ Yes

☐ No

If yes, preferred contact method:

☐ Phone call

☐ SMS

☐ WhatsApp

☐ In-person visit

SECTION 8: FINAL THOUGHTS

8.1 Overall rating of AgroAI application:

☐ ⭐⭐⭐⭐⭐ Excellent (5/5)

☐ ⭐⭐⭐⭐ Very Good (4/5)

☐ ⭐⭐⭐ Good (3/5)

☐ ⭐⭐ Fair (2/5)

☐ ⭐ Poor (1/5)

8.2 In one sentence, how would you describe this app to another farmer?

8.3 Any additional comments or suggestions?

Appendix G: OBSERVER NOTES (For Research Team)

Observer: ____________________________

Session Duration: _______ minutes

Notable Observations:

Technical Issues Observed:

User Behavior Notes:

Thank you for your valuable feedback!

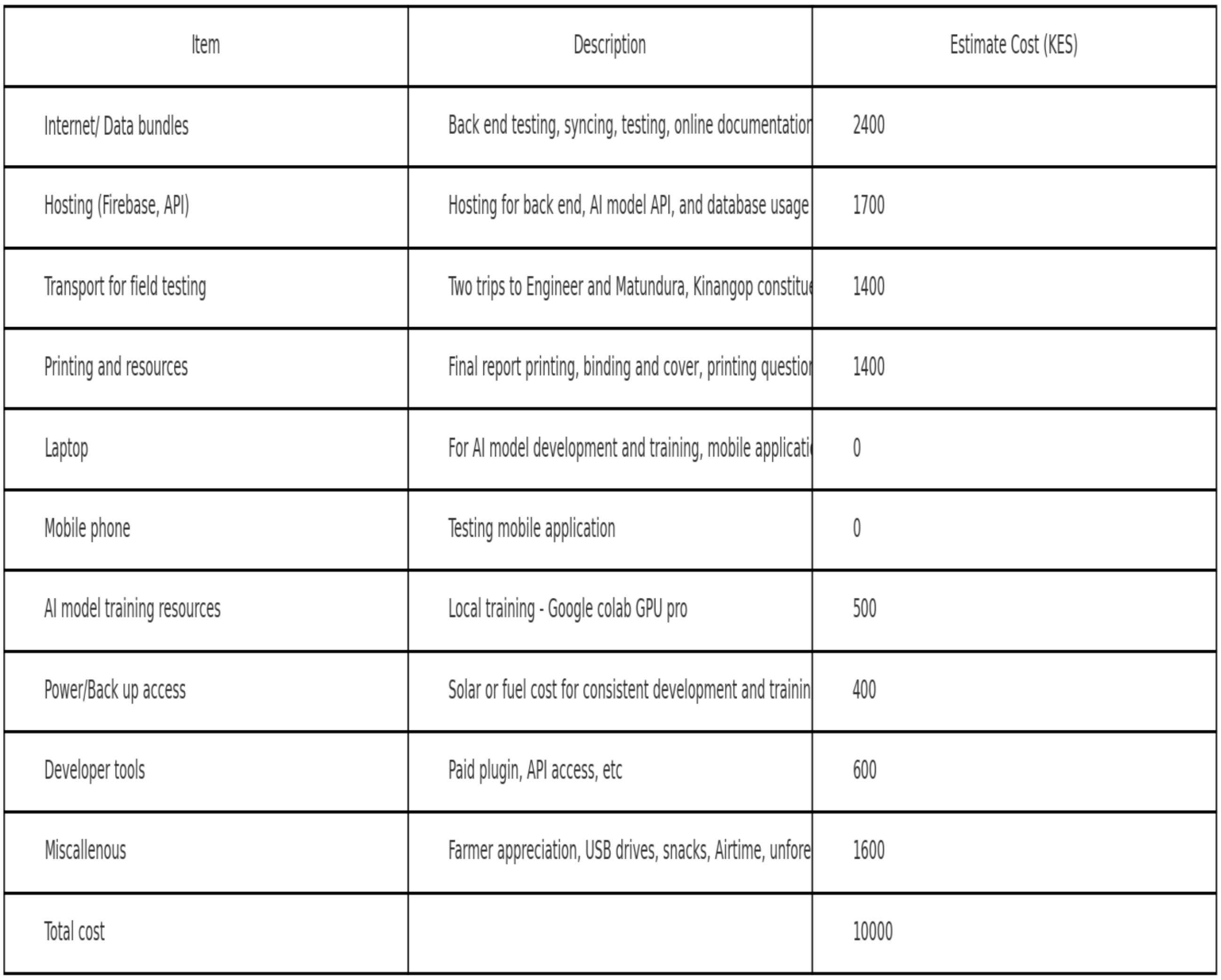
Your input will help us improve AgroAI to better serve farming communities in different regions of Kenya. All information provided will be kept confidential and used solely for research and development purposes.

Form Completed: _______

Time: _______

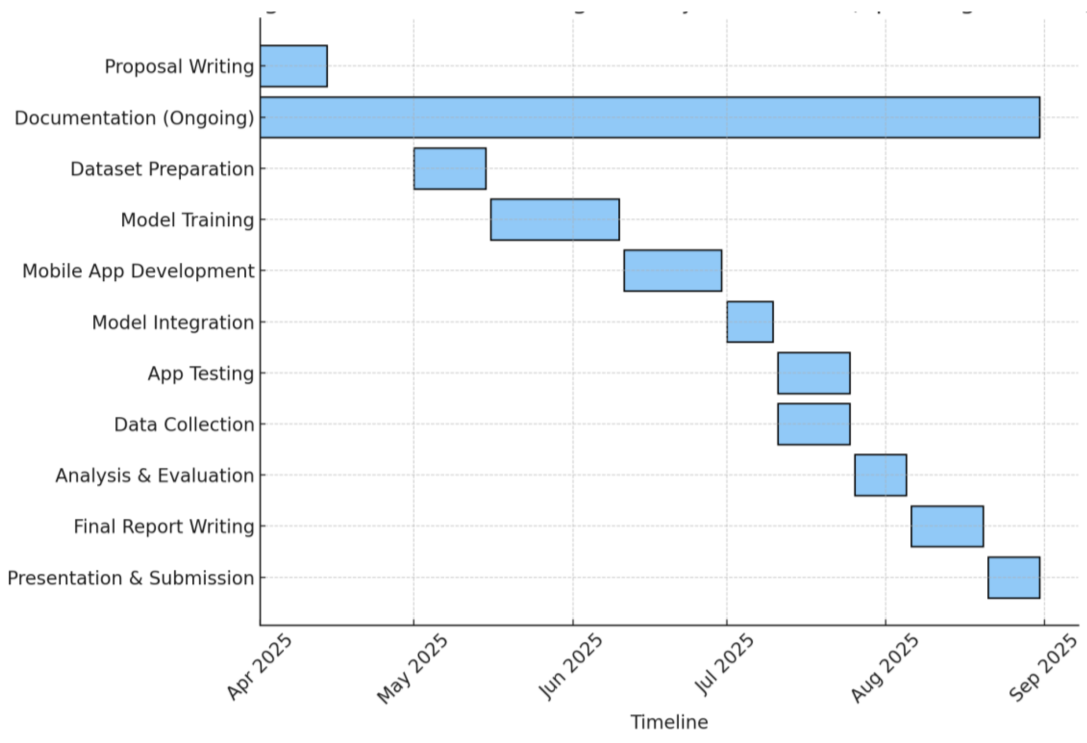
Signature:

Appendix H: AgroAI Project Budget



Item	Description	Estimate Cost (KES)
Internet/ Data bundles	Back end testing, syncing, testing, online documentation	2400
Hosting (Firebase, API)	Hosting for back end, AI model API, and database usage	1700
Transport for field testing	Two trips to Engineer and Matundura, Kinangop constitu	1400
Printing and resources	Final report printing, binding and cover, printing questior	1400
Laptop	For AI model development and training, mobile applicati	0
Mobile phone	Testing mobile application	0
AI model training resources	Local training - Google colab GPU pro	500
Power/Back up access	Solar or fuel cost for consistent development and trainin	400
Developer tools	Paid plugin, API access, etc	600
Miscallenous	Farmer appreciation, USB drives, snacks, Airtime, unfore	1600
Total cost		10000

Appendix I: AgroAI Work Plan



Appendix J: Code Snippets

Register Screen Code Snippet

```
@OptIn(ExperimentalMaterial3Api::class)

@Composable

fun DashboardScreen(

    onNavigateToProfile: () -> Unit,
    onNavigateToResults: () -> Unit
) {
    var           selectedImageUri           by           remember
{ mutableStateOf<Uri?>(null) }
    var isAnalyzing by remember { mutableStateOf(false) }
    val context = LocalContext.current

    val tempImageFile = remember {
        File(context.cacheDir, "temp_plant_image.jpg")
    }

    val tempImageUri = remember {
        FileProvider.getUriForFile(
            context,
            "${context.packageName}.fileprovider",
            tempImageFile
        )
    }

    // Gallery launcher
    val galleryLauncher = rememberLauncherForActivityResult(
        contract = ActivityResultContracts.GetContent()
    ) { uri ->
        selectedImageUri = uri
    }

    // Camera launcher
    val cameraLauncher = rememberLauncherForActivityResult(
        contract = ActivityResultContracts.TakePicture()
    ) { success ->
        if (success) {
            selectedImageUri = tempImageUri
        }
    }

    // Camera permission launcher
    val cameraPermissionLauncher =
rememberLauncherForActivityResult(
        contract = ActivityResultContracts.RequestPermission()
    ) { isGranted ->
        if (isGranted) {
            cameraLauncher.launch(tempImageUri)
        }
    }
```

Registration Screen Code Snippet

```
@Composable
fun RegisterScreen(
    onNavigateToLogin: () -> Unit,
    onRegisterSuccess: () -> Unit,
    authViewModel: AuthViewModel
) {
Column(
    horizontalAlignment = Alignment.CenterHorizontally,
    verticalArrangement = Arrangement.Center,
    modifier = Modifier.fillMaxSize()
) {
    Text(
        text = "□ AgroAI",
        fontSize = 32.sp,
        fontWeight = FontWeight.Bold,
        color = MaterialTheme.colorScheme.primary,
        modifier = Modifier.padding(bottom = 8.dp)
    )
OutlinedTextField(
    value = email,
    onValueChange = { email = it },
    label = { Text("Email") },
    leadingIcon = { Icon(Icons.Default.Email, contentDescription =
null) },
    keyboardOptions = KeyboardOptions(
        keyboardType = KeyboardType.Email,
        imeAction = ImeAction.Next
    ),
    modifier = Modifier
        .fillMaxWidth()
        .padding(bottom = 16.dp),
    singleLine = true
)
Button(
    onClick = {
        if (canRegister) {
            authViewModel.signUp(email, password, onRegisterSuccess)
        }
    },
    enabled = !isLoading && canRegister,
    shape = ShapeDefaults.Small,
    modifier = Modifier
        .fillMaxWidth()
        .height(48.dp)
) {
    if (isLoading) {
        CircularProgressIndicator(
            modifier = Modifier.size(20.dp),
            color = MaterialTheme.colorScheme.onPrimary
        )
    } else {
        Text("Create Account")
    }
}

Row(
    modifier = Modifier
        .fillMaxWidth()
        .padding(top = 24.dp),
    horizontalArrangement = Arrangement.Center,
    verticalAlignment = Alignment.CenterVertically
) {
    Text("Already have an account? ")
    TextButton(onClick = onNavigateToLogin) {
        Text("Sign In")
```

Login Screen Code Snippet

```
@Composable
fun LoginScreen(
    onNavigateToRegister: () -> Unit,
    onNavigateToForgotPassword: () -> Unit,
    onLoginSuccess: () -> Unit,
    authViewModel: AuthViewModel
) {
OutlinedTextField(
    value = email,
    onValueChange = { email = it },
    label = { Text("Email") },
    leadingIcon = { Icon(Icons.Default.Email, contentDescription =
null) },
    keyboardOptions = KeyboardOptions(
        keyboardType = KeyboardType.Email,
        imeAction = ImeAction.Next
    ),
    modifier = Modifier
        .fillMaxWidth()
        .padding(bottom = 16.dp),
    singleLine = true
)
```

```
OutlinedTextField(
    value = password,
    onValueChange = { password = it },
    label = { Text("Password") },
    leadingIcon = { Icon(Icons.Default.Lock, contentDescription =
null) },
    trailingIcon = {
        IconButton(onClick = { passwordVisible = !passwordVisible }) {
            Icon(
                painterResource(if (passwordVisible)
R.drawable.visibility else R.drawable.visibility_off),
                contentDescription = if (passwordVisible) "Hide
password" else "Show password"
            )
        }
    },
    visualTransformation = if (passwordVisible)
VisualTransformation.None else PasswordVisualTransformation(),
    keyboardOptions = KeyboardOptions(
        keyboardType = KeyboardType.Password,
        imeAction = ImeAction.Done
    ),
    modifier = Modifier
        .fillMaxWidth()
        .padding(bottom = 8.dp),
    singleLine = true
)
TextButton(
    onClick = onNavigateToForgotPassword,
    modifier = Modifier
        .fillMaxWidth()
        .padding(bottom = 24.dp)
) {
    Text(
        text = "Forgot Password?",
        textAlign = TextAlign.End,
        modifier = Modifier.fillMaxWidth()
    )
}
Button() {
    if (isLoading) {
        CircularProgressIndicator(
            modifier = Modifier.size(20.dp),
            color = MaterialTheme.colorScheme.onPrimary
        )
    } else {
        Text("Sign In")
    }
}

Row(
    modifier = Modifier
        .fillMaxWidth()
        .padding(top = 24.dp),
    horizontalArrangement = Arrangement.Center,
    verticalAlignment = Alignment.CenterVertically
) {
    Text("Don't have an account? ")
    TextButton(onClick = onNavigateToRegister) {
        Text("Sign Up")
    }
}
```

Results Screen Code Snippet

```
@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun ResultsScreen(
    onNavigateBack: () -> Unit
) {
Column(
    modifier = Modifier
        .fillMaxSize()
        .padding(paddingValues)
        .padding(horizontal = 16.dp)
) {
    Spacer(modifier = Modifier.height(16.dp))

    // Health Status Card
    Card(
        modifier = Modifier
            .fillMaxWidth()
            .padding(bottom = 16.dp),
        colors = CardDefaults.cardColors(
            containerColor = if (mockResult.status == "Healthy")
                Color(0xFFE8F5E8) else Color(0xFFFFF3E0)
        )
    ) {
        Row(
            modifier = Modifier
                .fillMaxWidth()
                .padding(20.dp),
            verticalAlignment = Alignment.CenterVertically
        ) {
            Icon(
                if (mockResult.status == "Healthy")
```

```
Icons.Default.CheckCircle else Icons.Default.Warning,
                contentDescription = null,
                modifier = Modifier
                    .size(48.dp)
                    .padding(end = 16.dp),
                tint = if (mockResult.status == "Healthy")
Color(0xFF4CAF50) else Color(0xFFFF9800)
            )

            Column {
                Text(
                    text = "Plant Status: ${mockResult.status}",
                    fontSize = 20.sp,
                    fontWeight = FontWeight.Bold
                )

                if (mockResult.disease.isNotEmpty()) {
                    Text(
                        text = mockResult.disease,
                        fontSize = 16.sp,
                        color =
MaterialTheme.colorScheme.onSurfaceVariant,
                        modifier = Modifier.padding(top = 4.dp)
                    )
                }

                Text(
                    text = "Confidence: ${mockResult.confidence}%",
                    fontSize = 14.sp,
                    color =
MaterialTheme.colorScheme.onSurfaceVariant,
                    modifier = Modifier.padding(top = 8.dp)
                )
            }
        }
    }

    // Missing Nutrients Card
    if (mockResult.missingNutrients.isNotEmpty()) {
        Card(
            modifier = Modifier
                .fillMaxWidth()
                .padding(bottom = 16.dp)
        ) {
            Column(
                modifier = Modifier.padding(20.dp)
            ) {
                Text(
                    text = "Missing Nutrients",
                    fontSize = 18.sp,
                    fontWeight = FontWeight.Bold,
                    modifier = Modifier.padding(bottom = 12.dp)
                )

                mockResult.missingNutrients.forEach { nutrient ->
                    Row(
                        modifier = Modifier.padding(vertical = 4.dp),
                        verticalAlignment = Alignment.CenterVertically
                    ) {
                        Icon(
                            painterResource(R.drawable.fiber),
                            contentDescription = null,
                            modifier = Modifier
                                .size(24.dp)
                            .padding(end = 12.dp),
                        tint = MaterialTheme.colorScheme.primary
                    )
                    Text(
                        text = nutrient,
                        fontSize = 16.sp
                    )
                }
            }
        }
    }
    }
```

Firebase Authentication Code Snippet

```
class AuthViewModel : ViewModel() {
    private val auth = FirebaseAuth.getInstance()

    private val _isAuthenticated =
MutableStateFlow(auth.currentUser != null)
    val isAuthenticated: StateFlow<Boolean> =
_isAuthenticated.asStateFlow()

    private val _currentUser = MutableStateFlow(auth.currentUser)
    val currentUser: StateFlow<FirebaseUser?> =
_currentUser.asStateFlow()

    private val _isLoading = MutableStateFlow(false)
    val isLoading: StateFlow<Boolean> = _isLoading.asStateFlow()
    private val _errorMessage = MutableStateFlow<String?>(null)
    val errorMessage: StateFlow<String?> =
_errorMessage.asStateFlow()

    init {
        auth.addAuthStateListener { firebaseAuth ->
            _isAuthenticated.value = firebaseAuth.currentUser != null
            _currentUser.value = firebaseAuth.currentUser
        }
    }

    fun signIn(email: String, password: String, onSuccess: () -> Unit)
{
        viewModelScope.launch {
            try {
                _isLoading.value = true
                _errorMessage.value = null
                auth.signInWithEmailAndPassword(email,
password).await()
                onSuccess()
            } catch (e: Exception) {
                _errorMessage.value = e.message
            } finally {
                _isLoading.value = false
            }
        }
    }

    fun signUp(email: String, password: String, onSuccess: () -> Unit)
{
        viewModelScope.launch {
            try {
                _isLoading.value = true
                _errorMessage.value = null
                auth.createUserWithEmailAndPassword(email,
password).await()
                onSuccess()
            } catch (e: Exception) {
                _errorMessage.value = e.message
            } finally {
                _isLoading.value = false
            }
        }
    }

    fun resetPassword(email: String, onSuccess: () -> Unit) {
        viewModelScope.launch {
            try {
                _isLoading.value = true
                _errorMessage.value = null
                auth.sendPasswordResetEmail(email).await()
                onSuccess()
            } catch (e: Exception) {
                _errorMessage.value = e.message
            } finally {
                _isLoading.value = false
            }
        }
    }

    fun signOut() {
        auth.signOut()
    }
```

```
        fun clearError() {
            _errorMessage.value = null
        }
    }
```

AgroAI Model Training and Testing

```
from google.colab import files
files.upload()

import os
import shutil

os.makedirs("/root/.kaggle", exist_ok=True)
shutil.move("kaggle.json", "/root/.kaggle/kaggle.json")
os.chmod("/root/.kaggle/kaggle.json", 600)

!pip install kaggle
!kaggle datasets download -d vipoooool/new-plant-diseases-dataset

import zipfile

with zipfile.ZipFile("new-plant-diseases-dataset.zip", 'r') as zip_ref:
    zip_ref.extractall("vipool_dataset")

import os

data_dir = "/content/vipool_dataset/new plant diseases
        dataset(augmented)/New Plant Diseases
        Dataset(Augmented)/train"
classes = sorted(os.listdir(data_dir))
print(f"Total classes: {len(classes)}")
print("Class labels:")
for c in classes:
    print("-", c)

with open("class_labels.txt", "w") as f:
    for c in classes:
        f.write(c + "\n")

import warnings
warnings.filterwarnings("ignore")
import tensorflow as tf
import matplotlib.pyplot as plt
tf.compat.v1.set_random_seed(0)
from tensorflow import keras
import numpy as np
np.random.seed(0)
import itertools
from tensorflow.keras.utils import image_dataset_from_directory
from tensorflow.keras.layers import Rescaling
from sklearn.metrics import precision_score, accuracy_score, recall_score,
        confusion_matrix, ConfusionMatrixDisplay

# Print all disease (class) names from the dataset
train_gen                                                              =
        image_dataset_from_directory(directory="/content/vipool_dataset/n
        ew  plant  diseases  dataset(augmented)/New  Plant  Diseases
        Dataset(Augmented)/train",
                                   image_size=(256, 256))
test_gen                                                               =
        image_dataset_from_directory(directory="/content/vipool_dataset/n
        ew  plant  diseases  dataset(augmented)/New  Plant  Diseases
        Dataset(Augmented)/valid",
                                        image_size=(256, 256))
print("Disease Classes in the Dataset:")
for idx, class_name in enumerate(train_gen.class_names):
    print(f"{idx}: {class_name}")

#train_gen    =    image_dataset_from_directory(directory="../input/new-
      plant-diseases-dataset/train",image_size=(256, 256))
#test_gen     =    image_dataset_from_directory(directory="../input/new-
      plant-diseases-dataset/valid",image_size=(256, 256))

train_gen                                                              =
      image_dataset_from_directory(directory="/content/vipool_dataset/n
      ew  plant  diseases  dataset(augmented)/New  Plant  Diseases
      Dataset(Augmented)/train",
                                        image_size=(256, 256))
test_gen                                                               =
      image_dataset_from_directory(directory="/content/vipool_dataset/n
      ew  plant  diseases  dataset(augmented)/New  Plant  Diseases
      Dataset(Augmented)/valid",
                                        image_size=(256, 256))

rescale = Rescaling(scale=1.0/255)
train_gen = train_gen.map(lambda image,label:(rescale(image),label))
test_gen  = test_gen.map(lambda image,label:(rescale(image),label))
# Print all disease (class) names from the dataset

train_gen_raw = image_dataset_from_directory(
    directory="/content/vipool_dataset/new        plant        diseases
      dataset(augmented)/New              Plant              Diseases
      Dataset(Augmented)/train",
    image_size=(256, 256)
)
```

```
print(train_gen_raw.class_names)

class_names = train_gen_raw.class_names

with open("class_names.txt", "w") as f:
    for class_name in class_names:
        f.write(f"{class_name}\n")

import json

with open("class_names.json", "w") as f:
    json.dump(class_names, f)

model = keras.Sequential()

model.add(keras.layers.Conv2D(32,(3,3),activation="relu",padding="same"
        ,input_shape=(256,256,3)))
model.add(keras.layers.Conv2D(32,(3,3),activation="relu",padding="same"
        ))
model.add(keras.layers.MaxPooling2D(3,3))

model.add(keras.layers.Conv2D(64,(3,3),activation="relu",padding="same"
        ))
model.add(keras.layers.Conv2D(64,(3,3),activation="relu",padding="same"
        ))
model.add(keras.layers.MaxPooling2D(3,3))

model.add(keras.layers.Conv2D(128,(3,3),activation="relu",padding="sam
        e"))
model.add(keras.layers.Conv2D(128,(3,3),activation="relu",padding="sam
        e"))
```

```
model.add(keras.layers.MaxPooling2D(3,3))

model.add(keras.layers.Conv2D(256,(3,3),activation="relu",padding="sam
        e"))
model.add(keras.layers.Conv2D(256,(3,3),activation="relu",padding="sam
        e"))

model.add(keras.layers.Conv2D(512,(5,5),activation="relu",padding="sam
        e"))
model.add(keras.layers.Conv2D(512,(5,5),activation="relu",padding="sam
        e"))

model.add(keras.layers.Flatten())

model.add(keras.layers.Dense(1568,activation="relu"))
model.add(keras.layers.Dropout(0.5))

model.add(keras.layers.Dense(38,activation="softmax"))

opt = keras.optimizers.Adam(learning_rate=0.0001)
model.compile(optimizer=opt,loss="sparse_categorical_crossentropy",metric
        s=['accuracy'])
model.summary()

ep = 10
history = model.fit(
    train_gen,
    validation_data=test_gen,
    epochs=ep
)

!rm -rf /content/drive
```

```
from google.colab import drive
drive.mount('/content/drive')

model.save('/content/drive/MyDrive/PlantVillageModel/agroai_final_mode
        l.h5')
model.save('/content/drive/MyDrive/PlantVillageModel/vipool_dataset.ker
        as')

from google.colab import files
files.download('vipool_dataset.h5')
files.download('vipool_dataset.keras')

# Then save
model.save('/content/drive/MyDrive/vipoool_dataset3.h5')

model.save("vipool_dataset2.h5")

# Then save
model.save('/content/drive/MyDrive/vipoool_dataset.keras')

model.save("vipool_dataset.keras")

# Then save
model.save('/content/drive/MyDrive/PlantVillageModel/vipool_dataset.h5')

# Save the model
model.save("plant_disease_model.h5")
model.save("plant_disease_model.keras")

import os
import json
```

```
from google.colab import files
import shutil

# Define the existing Drive folder path
drive_folder = '/content/drive/MyDrive/PlantVillageModel'

# List of model files to upload
files_to_upload = [
    'plant_disease_model.h5',
    'plant_disease_model.keras',
    'class_labels.txt',
    'class_names.txt',
    'class_names.json',
    'agroai_final_model.h5',
    'vipool_dataset.keras',
    'vipool_dataset.h5'
]
# Upload each file
for filename in files_to_upload:
    src_path = os.path.join('/content', filename)
    dst_path = os.path.join(drive_folder, filename)
    if os.path.exists(src_path):
        shutil.copy(src_path, dst_path)
        print(f"uploaded: {filename}")
    else:
        print(f"nnot found: {filename}")

plt.figure(figsize=(10, 5))

# Plot for Loss
plt.subplot(1, 2, 1)
plt.title("Train and Validation Loss")
plt.xlabel("Epoch")
plt.ylabel("Loss")
plt.plot(history.history['loss'],    label="Train    Loss",    color='dodgerblue')
       # blue
plt.plot(history.history['val_loss'],  label="Validation  Loss",  color='orange')
       # orange
plt.xlim(0, 10)
plt.ylim(0.0, 1.0)
plt.legend()

# plt.figure(figsize = (20,5))
# plt.subplot(1,2,1)
# plt.title("Train and Validation Loss")
# plt.xlabel("Epoch")
# plt.ylabel("Loss")
# plt.plot(history.history['loss'],label="Train Loss")
# plt.plot(history.history['val_loss'], label="Validation Loss")
# plt.xlim(0, 10)
# plt.ylim(0.0,1.0)
# plt.legend()

# plt.subplot(1,2,2)
# plt.title("Train and Validation Accuracy")
# plt.xlabel("Epoch")
# plt.ylabel("Accuracy")
# plt.plot(history.history['accuracy'], label="Train Accuracy")
# plt.plot(history.history['val_accuracy'], label="Validation Accuracy")
# plt.xlim(0, 9.25)
# plt.ylim(0.75,1.0)
# plt.legend()
# plt.tight_layout()
plt.figure(figsize=(10, 5))
```

```
# Plot for Loss
plt.subplot(1, 2, 1)
plt.title("Train and Validation Loss")
plt.xlabel("Epoch")
plt.ylabel("Loss")
plt.plot(history.history['loss'],    label="Train    Loss",    color='dodgerblue')
        # blue
plt.plot(history.history['val_loss'],  label="Validation  Loss",  color='orange')
        # orange
plt.xlim(0, 10)
plt.ylim(0.0, 1.0)
plt.legend()
plt.figure(figsize=(10, 5))
# Plot for Accuracy
plt.subplot(1, 2, 2)
plt.title("Train and Validation Accuracy")
plt.xlabel("Epoch")
plt.ylabel("Accuracy")
plt.plot(history.history['accuracy'],  label="Train  Accuracy",  color='green')
        # green
plt.plot(history.history['val_accuracy'],      label="Validation      Accuracy",
        color='red')     # red
plt.xlim(0, 9.25)
plt.ylim(0.75, 1.0)
plt.legend()

plt.tight_layout()
plt.show()

# labels = []
# predictions = []
# for x,y in test_gen:
#     labels.append(list(y.numpy()))
#     predictions.append(tf.argmax(model.predict(x),1).numpy())

import numpy as np

# Convert predictions and labels to flat lists
predictions = np.array(predictions).tolist()
labels = np.array([np.argmax(batch) for batch in labels]).tolist()

print("Train   Accuracy     :  {:.2f}  %".format(history.history['accuracy'][-
       1]*100))
print("Test Accuracy    : {:.2f} %".format(accuracy_score(labels, predictions)
       * 100))
print("Precision Score : {:.2f} %".format(precision_score(labels, predictions,
       average='micro') * 100))
print("Recall  Score        :  {:.2f}  %".format(recall_score(labels,  predictions,
       average='micro') * 100))

# plt.figure(figsize= (20,5))
# cm = confusion_matrix(labels, predictions)
# disp = ConfusionMatrixDisplay(confusion_matrix=cm,
#                               display_labels=list(range(1,39)))
# fig, ax = plt.subplots(figsize=(15,15))
# disp.plot(ax=ax,colorbar= False,cmap = 'YlGnBu')
# plt.title("Confusion Matrix")
# plt.xlabel('Predicted Labels')
# plt.ylabel('True Labels')
# plt.show()
plt.figure(figsize=(20, 5))
cm = confusion_matrix(labels, predictions)
```

```
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
    display_labels=list(range(1, 39)))
fig, ax = plt.subplots(figsize=(15, 15))
disp.plot(ax=ax, colorbar=False, cmap='Blues')  # changed colormap here
plt.title("Confusion Matrix")
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.show()

# Disease-to-Pesticide mapping (simplified example, can be expanded or
    customized)
disease_to_pesticide = {
  "Apple___Apple_scab": "Captan, Mancozeb",
  "Apple___Black_rot": "Ziram, Captan",
  "Apple___Cedar_apple_rust": "Myclobutanil",
  "Apple___healthy": "No pesticide needed",
  "Blueberry___healthy": "No pesticide needed",
  "Cherry_(including_sour)___Powdery_mildew": "Sulfur, Myclobutanil",
  "Cherry_(including_sour)___healthy": "No pesticide needed",
  "Corn_(maize)___Cercospora_leaf_spot Gray_leaf_spot": "Strobilurins",
  "Corn_(maize)___Common_rust_": "Propiconazole",
  "Corn_(maize)___Northern_Leaf_Blight": "Azoxystrobin",
  "Corn_(maize)___healthy": "No pesticide needed",
  "Grape___Black_rot": "Mancozeb, Captan",
  "Grape___Esca_(Black_Measles)": "No effective chemical control",
  "Grape___Leaf_blight_(Isariopsis_Leaf_Spot)": "Mancozeb",
  "Grape___healthy": "No pesticide needed",
  "Orange___Haunglongbing_(Citrus_greening)": "Vector control:
    Imidacloprid",
  "Peach___Bacterial_spot": "Copper-based sprays",
  "Peach___healthy": "No pesticide needed",
  "Pepper,_bell___Bacterial_spot": "Copper-based sprays",
  "Pepper,_bell__healthy": "No pesticide needed",
  "Potato__Early_blight": "Chlorothalonil, Mancozeb",
  "Potato__Late_blight": "Metalaxyl, Mancozeb",
  "Potato__healthy": "No pesticide needed",
  "Raspberry__healthy": "No pesticide needed",
  "Soybean__healthy": "No pesticide needed",
  "Squash__Powdery_mildew": "Sulfur, Neem oil",
  "Strawberry__Leaf_scorch": "Captan",
  "Strawberry__healthy": "No pesticide needed",
  "Tomato__Bacterial_spot": "Copper sprays",
  "Tomato__Early_blight": "Chlorothalonil, Mancozeb",
  "Tomato__Late_blight": "Metalaxyl",
  "Tomato__Leaf_Mold": "Copper-based fungicides",
  "Tomato__Septoria_leaf_spot": "Chlorothalonil",
  "Tomato__Spider_mites Two-spotted_spider_mite": "Insecticidal soap,
    Neem oil",
  "Tomato__Target_Spot": "Chlorothalonil",
  "Tomato__Tomato_Yellow_Leaf_Curl_Virus":        "Vector        control:
    Imidacloprid",
  "Tomato__Tomato_mosaic_virus": "Sanitation, resistant varieties",
  "Tomato__healthy": "No pesticide needed"
}

train_gen = image_dataset_from_directory(
  directory="/content/vipool_dataset/new          plant          diseases
    dataset(augmented)/New                  Plant                Diseases
    Dataset(Augmented)/train",
  image_size=(256, 256),
  shuffle=True
)

# Save class names in a separate variable
```

```
class_names = train_gen.class_names

# from PIL import Image

# def predict_disease_and_recommend(image_path):
#     # Load and preprocess image
#         img = tf.keras.preprocessing.image.load_img(image_path,
      target_size=(256, 256))
#     img_array = tf.keras.preprocessing.image.img_to_array(img) / 255.0
#     img_array = tf.expand_dims(img_array, 0)  # Add batch dimension

#     # Predict
#     predictions = model.predict(img_array)
#     predicted_class = train_gen.class_names[np.argmax(predictions)]
#         pesticide = disease_to_pesticide.get(predicted_class, "No data
      available")
#     plt.figure(figsize=(6, 6))
#     plt.imshow(img)
#     plt.axis('off')
#     plt.title(f"Disease: {predicted_class}\nPesticide: {pesticide}",
#               fontsize=12, color='darkgreen', loc='center')
#     plt.tight_layout()
#     plt.show()

#     print(f"\n Predicted Disease: {predicted_class}")
#     print(f" Recommended Pesticide: {pesticide}")
import matplotlib.pyplot as plt
from PIL import Image
import os

def predict_disease_and_recommend(image_path, save_output=True):
    # Load and preprocess image
    img            =            tf.keras.preprocessing.image.load_img(image_path,
       target_size=(256, 256))
    img_array = tf.keras.preprocessing.image.img_to_array(img) / 255.0
    img_array = tf.expand_dims(img_array, 0)  # Add batch dimension

    # Predict
    predictions = model.predict(img_array)
    predicted_class = class_names[np.argmax(predictions)]
    pesticide = disease_to_pesticide.get(predicted_class, "No data available")

    # Display and save image with overlayed text
    plt.figure(figsize=(6, 6))
    plt.imshow(img)
    plt.axis('off')
    plt.title(f"Disease: {predicted_class}\nPesticide: {pesticide}",
              fontsize=12, color='darkgreen', loc='center')
    plt.tight_layout()

    if save_output:
        output_path = "prediction_output.png"
        plt.savefig(output_path)
        print(f"\nImage saved to: {os.path.abspath(output_path)}")

    plt.show()

# Just call this function with the path to your image
predict_disease_and_recommend("/content/testImage1.jpeg")

# Just call this function with the path to your image
predict_disease_and_recommend("/content/testImagetlateblight.jpeg")
```

Clarification Report

```
{
    "Pepper_bell__Bacterial_spot": {
        "precision": 0.9427083333333334,
        "recall": 0.905,
        "f1-score": 0.923469387755102,
        "support": 200.0
    },
    "Pepper_bell__healthy": {
        "precision": 0.904320987654321,
        "recall": 0.9898648648648649,
        "f1-score": 0.9451612903225807,
        "support": 296.0
    },
    "Potato___Early_blight": {
        "precision": 0.9441624365482234,
        "recall": 0.93,
        "f1-score": 0.9370277078085643,
        "support": 200.0
    },
    "Potato___Late_blight": {
        "precision": 0.9289340101522843,
        "recall": 0.915,
        "f1-score": 0.9219143576826196,
        "support": 200.0
    },
    "Potato___healthy": {
        "precision": 0.9375,
        "recall": 0.967741935483871,
        "f1-score": 0.9523809523809523,
        "support": 31.0
    },
    "Tomato_Bacterial_spot": {
        "precision": 0.8763796909492274,
        "recall": 0.931924882629108,
        "f1-score": 0.9032992036405005,
        "support": 426.0
    },
    "Tomato_Early_blight": {
        "precision": 0.8095238095238095,
        "recall": 0.68,
        "f1-score": 0.7391304347826086,
        "support": 200.0
    },
    "Tomato_Late_blight": {
        "precision": 0.8414322250639387,
        "recall": 0.8612565445026178,
        "f1-score": 0.851228978007762,
        "support": 382.0
    },
    "Tomato_Leaf_Mold": {
        "precision": 0.813953488372093,
        "recall": 0.7329842931937173,
        "f1-score": 0.7713498622589532,
        "support": 191.0
    },
    "Tomato_Septoria_leaf_spot": {
        "precision": 0.8233695652173914,
        "recall": 0.8535211267605634,
        "f1-score": 0.8381742738589212,
        "support": 355.0
    },
    "Tomato_Spider_mites_Two_spotted_spider_mite": {
        "precision": 0.8964285714285715,
        "recall": 0.7470238095238095,
        "f1-score": 0.814935064935065065,
        "support": 336.0
    },
    "Tomato__Target_Spot": {
        "precision": 0.8125,
        "recall": 0.693950177935943,
        "f1-score": 0.7485604606525912,
        "support": 281.0
    },
    "Tomato__Tomato_YellowLeaf__Curl_Virus": {
        "precision": 0.932824427480916,
        "recall": 0.9517133956386293,
        "f1-score": 0.9421742482652274,
        "support": 642.0
    },
    "Tomato__Tomato_mosaic_virus": {
        "precision": 0.9411764705882353,
        "recall": 0.8533333333333334,
        "f1-score": 0.8951048951048951,
        "support": 75.0
    },
    "Tomato_healthy": {
        "precision": 0.7984886649874056,
        "recall": 0.9937304075235109,
        "f1-score": 0.8854748603351955,
        "support": 319.0
    },
    "accuracy": 0.8746976294146106,
    "macro avg": {
        "precision": 0.8802468454199833,
        "recall": 0.8671363180926647,
        "f1-score": 0.8712923985194359,
        "support": 4134.0
    },
    "weighted avg": {
        "precision": 0.8751875429064093,
        "recall": 0.8746976294146106,
        "f1-score": 0.8725431631722057,
        "support": 4134.0
    }
}
```