**Module Code & Module Title**

**CS6P05NI Final Year Project**

**Assessment Weightage & Type**

**25% FYP Interim Report**

**Year and Semester**

**2021-22 Autumn**

**Student Name:**

**Group: C4**

**London Met ID: 21039638**

**College ID: NP01CP4A210010**

**Assignment Due Date: 2024-Jan-3rd 1:00 PM**

**Assignment Submission Date: 2024-Jan-3rd 6:00 AM**

**Submitted To: Suyog Man Singh**

**Word Count (Where Required): 3500**

*I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.*

# Acknowledgment

We extend our sincere gratitude to our esteemed supervisors, Suyog Man Singh and Yunisha Bajracharya, whose invaluable guidance and support have been instrumental throughout the development of the "Farm Assist- Plant Disease Classification" project. Suyog Man Singh's external supervision, coupled with his industry insights, has enriched the project with a broader perspective. Yunisha Bajracharya, our internal supervisor, has provided unwavering academic guidance, ensuring the project aligns with the highest standards.

Our gratitude extends to mentors, advisors, and the broader community for their guidance, support, and sharing of knowledge. Special thanks to dataset providers for their valuable contributions, open-source communities for tools and libraries, and educational institutions for providing essential resources.

We would also like to acknowledge the test users and beta testers for their crucial feedback, family and friends for their unwavering support, and the community at large for their interest and encouragement.

This project's collaborative nature has been a source of strength, and we appreciate the collective effort that has gone into its development.

# Abstract

The "Farm Assist- Plant Disease Classification" project aims to revolutionize agriculture by leveraging advanced technology for the early detection and classification of plant diseases. Utilizing Convolutional Neural Networks (CNNs), the system provides a comprehensive solution with a Django-based web application and a Flutter-powered mobile application. Users, including farmers and agricultural enthusiasts, can upload images of plants, and the system swiftly diagnoses diseases, contributing to sustainable agriculture. The project addresses the global challenge of food security by bridging the gap between cutting-edge machine learning techniques and practical agricultural needs. Open-source contributions and a user-friendly interface further enhance the impact of this innovative plant disease classification system.

This project's significance extends beyond its technical prowess. It envisions a future where the fusion of cutting-edge machine learning technologies and agriculture becomes seamless, fostering early disease detection and informed decision-making among farmers. The commitment to open-source contributions ensures that the benefits of this plant disease classification system permeate the broader research and agricultural communities, ultimately contributing to global food security and the advancement of sustainable farming practices.

# Table of Contents

## List of Figures

## List of Tables

# 1 Introduction

## 1.1 Introduction to the Topic

Agriculture stands as the backbone of human civilization, providing sustenance, economic stability, and livelihoods to communities worldwide. However, the agricultural sector faces an ongoing threat from plant diseases, which can result in significant crop losses if not addressed promptly. The "Farm Assist - Plant Disease Classification" project is a proactive response to this pressing challenge.

In this section, we delve into the critical importance of early detection and management of plant diseases in the realm of agriculture. We explore the impact of these diseases on crop yields, food security, and the economic stability of farming communities. The "Farm Assist" project emerges as a technological solution leveraging Convolutional Neural Networks (CNN) to revolutionize the way we detect and classify plant diseases. (Barbedo, 2019)

Through the lens of technology and machine learning, this project seeks to empower farmers and agricultural enthusiasts with a sophisticated tool capable of accurately identifying diseases across a diverse range of plant categories. By providing instant and reliable diagnoses, "Farm Assist" aims to minimize crop losses, support sustainable agriculture practices, and contribute to global food security.

## 1.2 Current Scenario in Agriculture

### 1.2.1 Agriculture in Nepal

The current state of agriculture is marked by a complex interplay of factors that significantly impact global food production, economic sustainability, and the well-being of farming communities. In this section, we provide an overview of the prevailing conditions in agriculture, shedding light on key challenges and opportunities. (Mondal, 2015)

Nepal's agriculture sector, serving as the backbone of its economy, plays a vital role in sustaining livelihoods and contributing to food security. However, the sector grapples with a unique set of challenges within the country's diverse topography and climatic conditions.

**1. Smallholder Farming:** The majority of farmers in Nepal are smallholders with limited access to resources and technology. Subsistence farming is prevalent, making communities vulnerable to fluctuations in crop yields and market dynamics.

**2. Topographical Diversity:** Nepal's landscape varies from the plains of Terai to the high-altitude regions of the Himalayas. This diversity poses challenges in terms of selecting suitable crops and implementing uniform agricultural practices. (Padol, 2016)

**3. Climate Vulnerability:** The impact of climate change is increasingly evident in Nepal, leading to erratic weather patterns, unpredictable rainfall, and the spread of pests and diseases. These factors heighten the vulnerability of crops to various stressors.

**4. Traditional Farming Practices:** While traditional farming methods are deeply ingrained in Nepalese agriculture, there is a need for modernization and the adoption of innovative technologies to improve productivity and resilience.

In the context of Nepal's agricultural scenario, addressing plant diseases becomes crucial for sustaining crop production and ensuring food security. The "Farm Assist - Plant Disease Classification" project, with its focus on leveraging technology for disease detection, aligns with the need for practical solutions tailored to the challenges faced by Nepalese farmers. By providing a tool for early disease identification, the project contributes to the resilience and sustainability of agriculture in Nepal.

| Resilience Factors | Efforts to Enhance Resilience |
|---|---|
| **Adoption of Technology** | Promoting modern agricultural practices, providing access to farming tools. |
| Climate-Resilient Crops | Research and promotion of crops resilient to changing climate conditions. |
| Farmer Education and Training | Training programs to update farmers on best practices and innovative methods. |
| Market Access and Diversification | Facilitating access to markets, encouraging crop diversification. |
| Community-Based Adaptation Strategies | Collaborative efforts within communities to address shared challenges. |

*Table 1: Agriculture Resilience Factors*

*Figure 1: Crop Diseases*

## 1.3 Problem Domain: Plant Disease Detection

Agriculture, being the backbone of global sustenance, faces a critical challenge in the form of plant diseases. Plant diseases can lead to significant losses in crop yield, affecting food security and economic stability. The traditional methods of disease detection are often slow, subjective, and prone to inaccuracies. The pressing need is to develop an advanced and efficient system that can rapidly and accurately identify plant diseases, enabling timely intervention and effective disease management**. (**Reza, 2016)

**Below are the problems identified by the survey with over 203 Peoples. This survey is a huge success for the plant disease classification:**

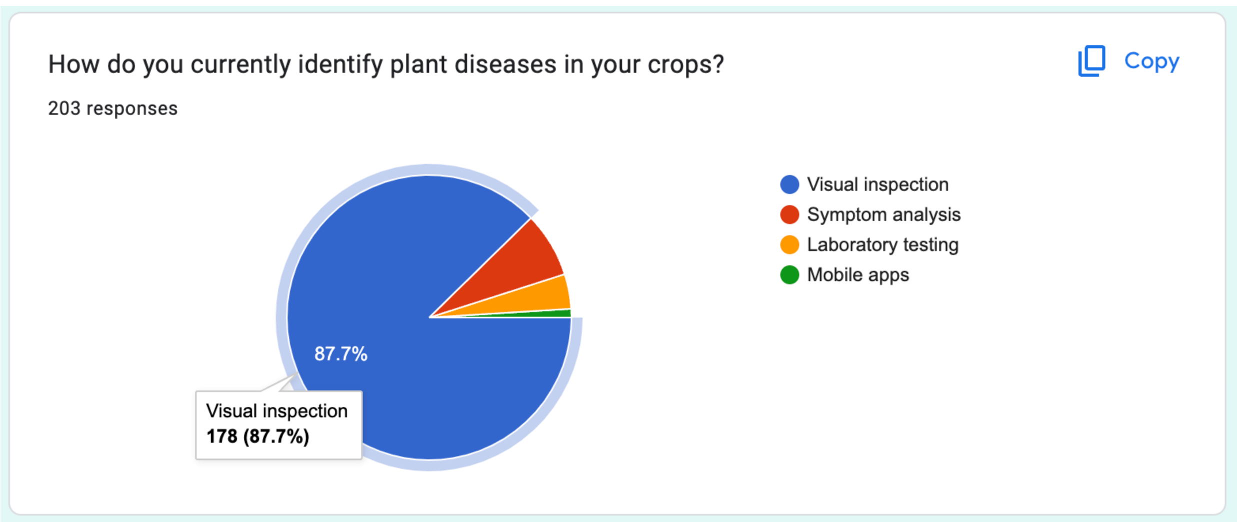



*Figure 2: Problem Domain1-survey*

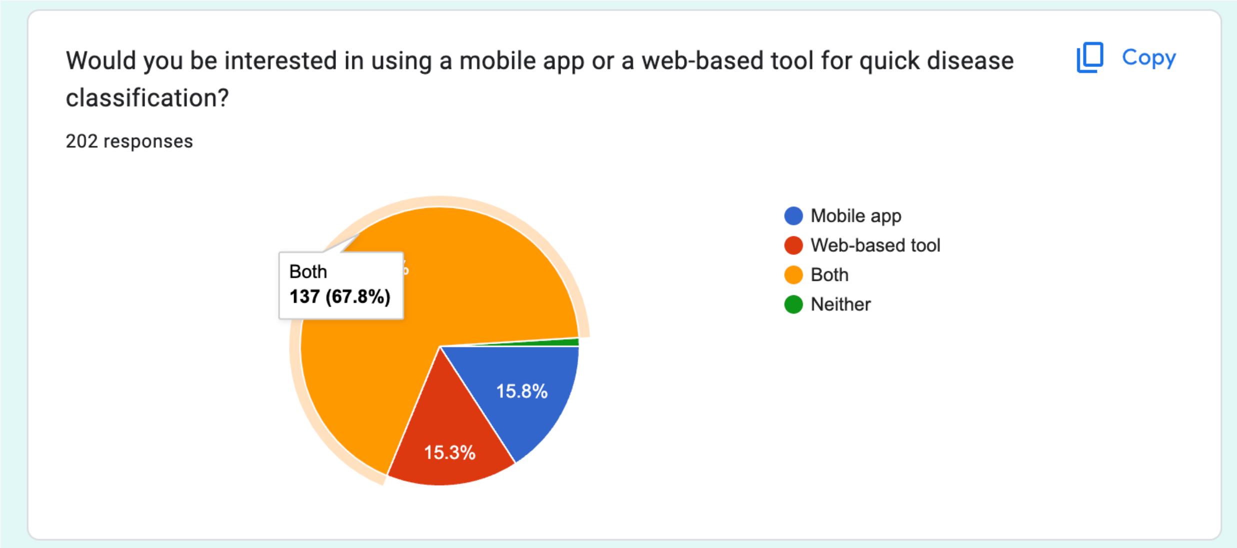



*Figure 3: Problem Domain2-survey*

**Challenges in Plant Disease Detection:**

**Impact on Crop Yield:** Plant diseases can cause substantial losses in crop production, impacting both the livelihoods of farmers and global food security.

**Slow and Inaccurate Detection Methods:** Traditional disease detection methods are often time-consuming and reliant on human expertise, leading to delays in diagnosis.

**Globalization and Spread:** With increased global trade, there's a risk of diseases spreading across regions, emphasizing the need for early detection to prevent outbreaks.

**Diverse Plant Species:** Agriculture involves a wide range of plant species, each susceptible to different diseases, making a comprehensive detection system challenging.

## 1.4 Project Scope: Farm Assist - Plant Disease Classification

The proposed "Farm Assist - Plant Disease Classification" project aims to address the challenges in plant disease detection by developing a robust classification system. This system will leverage advanced technologies, specifically Convolutional Neural Networks (CNNs), to create a model capable of accurately identifying diseases across a diverse set of plant categories. The scope of the project includes the development of both web and mobile applications for user-friendly access.

**Build a CNN-Based Model:** Develop a Convolutional Neural Network model trained on a diverse dataset encompassing various plant categories for effective disease classification.

**Create User-Friendly Applications:** Develop a Django-based web application and a Flutter-powered mobile application for seamless user interaction.

**Integrate Comprehensive Plant Disease Database:** Incorporate a comprehensive plant disease database to support the classification model and provide users with reference information.

**Ensure User Accessibility:** Prioritize user-friendliness in both web and mobile applications to cater to users with varying levels of technological proficiency.

**Contribute to Research Community:** Make the disease classification model and associated tools open-source, contributing to both the research and agricultural communities.

The "Farm Assist - Plant Disease Classification" project seeks to bridge the gap between advanced machine learning techniques and practical agricultural needs. By addressing the challenges in plant disease detection, the project aims to empower farmers and agricultural enthusiasts with a tool that facilitates early and accurate disease diagnosis, ultimately contributing to global food security and sustainable agriculture practices. (Tejoindh, 2016)

## 1.5 End User: Farmers and Agricultural Enthusiasts

The end users of the "Farm Assist - Plant Disease Classification" project are primarily farmers and agricultural enthusiasts. Here's a detailed description of the end users:

**Proposed System**

## 1.6 Aims and Objectives

### 1.6.1 Aims:

Bridge the Gap Between Technology and Agriculture:

The primary aim of the "Farm Assist - Plant Disease Classification" project is to bridge the gap between cutting-edge technology and the agricultural sector. By leveraging advanced machine learning techniques, specifically Convolutional Neural Networks (CNNs), the project seeks to empower farmers with a robust plant disease classification model.

### 1.6.2 Objectives:

1. **Develop a CNN-Based Model:**
2. **Create a Web Application:**
3. **Design a Mobile Application:**
4. **Ensure User Accessibility:**

A detail description can be found in the appendix

**Aims and Objective Description:**

## 1.7 Structure of Report

### 1.7.1 Introduction

The Introduction section provides comprehensive information about the project, encompassing details about the problem domain, end users, project scope, and the aims and objectives guiding the project.

### 1.7.2 Background/Literature Review

This section delves into the background and literature related to the project, covering insights into end users, problem-solving strategies, a review of similar applications, and an exploration of the technology and methodology employed in the project.

#### 1.7.3 Development Progress

The Development to Date chapter offers an overview of the project's progress, detailing the advancements and milestones achieved during its development.

#### 1.7.4 Analysis of Progress

The Analysis of Progress chapter involves a thorough examination of the project's current state, providing insights into its development trajectory. This analysis aligns with the project's Gantt chart, aiding in gauging the actual progress made.

#### 1.7.5 Future Work

The Future Work section outlines the pending tasks yet to be undertaken in the project's future development.

#### 1.7.6 References

The References section compiles all the sources referenced during the project, providing a comprehensive list of the literature and materials consulted.

#### 1.7.7 Appendix

The Appendix section contains concise descriptions of key project elements, including the Software Requirements Specification (SRS) document, high-level use case diagram, wireframes with accompanying screenshots, entity relationship explanations, data dictionaries, and brief overviews of the project as a solution. It also covers the considered and selected methodologies. Additionally, images from the survey and screenshots of Android and web code are included here.

# 2 Background/Literature Review

The selected technology stack for this project encompasses a range of cutting-edge tools and frameworks designed to optimize the development and functionality of the Plant Disease Classification System. Each component has been carefully chosen to contribute to the overall effectiveness and user-friendliness of the application.

## 2.1 Technology Used

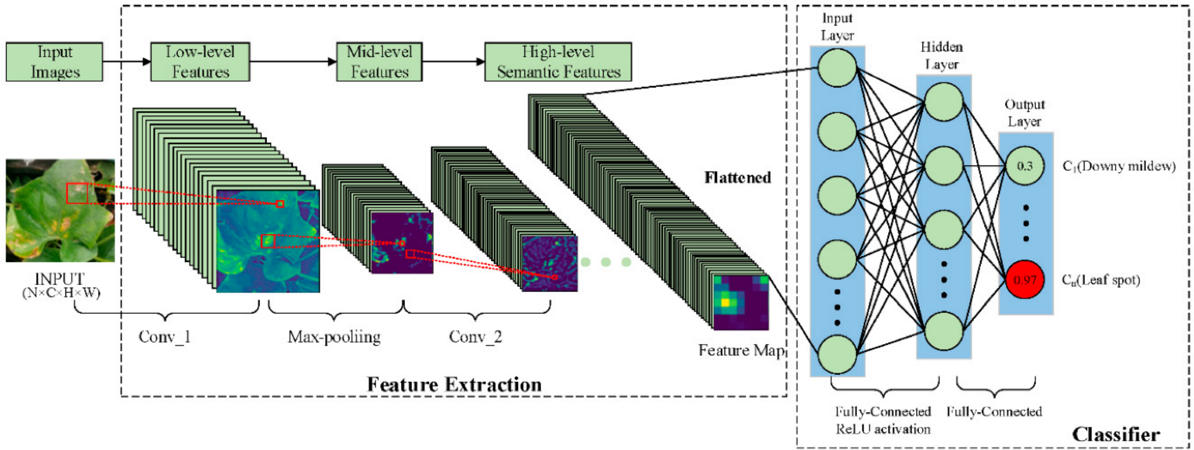
### 1.1.1 AI and Machine Learning

The used algorithm is CNN Resnet 8 which is a powerful ML for the computer Vision in detecting the patterns in the datasets.

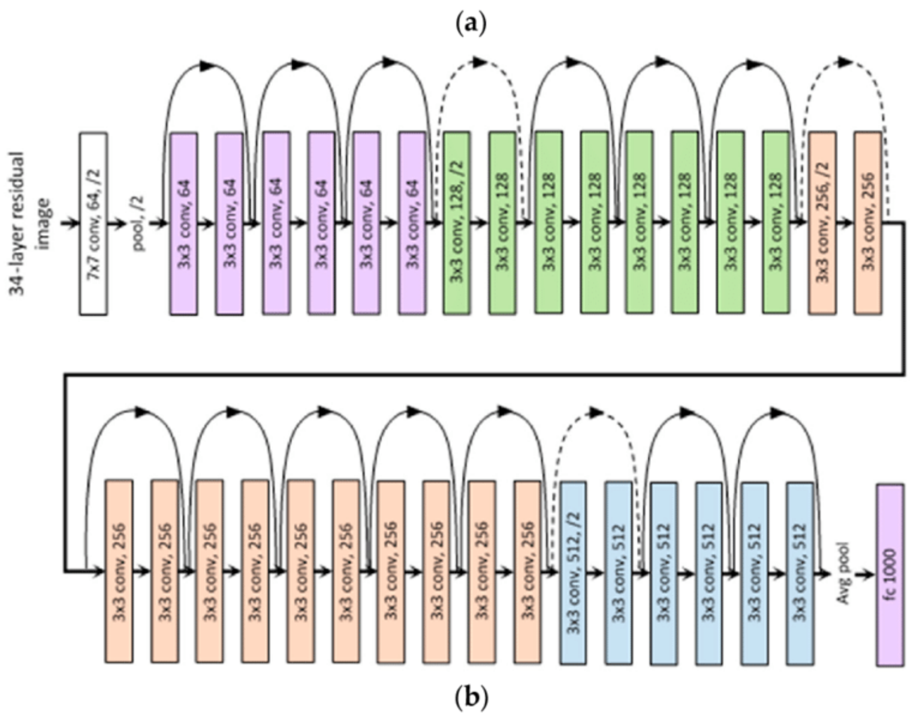
RESNET 8 won an award back in 2018 Google AI challenge.

The chosen model will work flawlessly in our project

1. **CNN**




2. **RESNET 8**




(Arya, 2018)

### 2.1.1 UI Framework

For the UI, React Tailwind is chosen for the web and Material UI is chosen for the Mobile App.

1. **Tailwind for Web**

**2. MaterialUI for Mobile**

### 2.1.2 Programming Language (e.g., Python)

Python, known for its readability and versatility, serves as the primary programming language for this project. Its extensive set of libraries and frameworks, especially in the context of AI and ML, makes it an ideal choice. Python contributes to the efficient implementation of complex algorithms and the overall development process.

For the Web: Python is chosen as a primary language.

For the mobile: Dart is chosen as a Primary Language which is flutters own used language.

**1) Python**

**2) JavaScript**

3) **Dart**

### 2.1.3 Framework (e.g., Django, Flutter)

**1. Django REST Framework**

Django REST Framework (DRF) is a powerful and flexible toolkit for building Web APIs using Django, which is a high-level Python web framework. DRF simplifies the process of creating APIs, making it easier to build robust and scalable web applications.

**2. React**

React is a JavaScript library for building user interfaces, particularly for single-page applications where the user interacts with the page dynamically. Developed by Facebook, React allows developers to build reusable UI components that update efficiently in response to data changes.

**3. Flutter**

Flutter is an open-source UI software development toolkit developed by Google. It is used for building natively compiled applications for mobile, web, and desktop from a single codebase. Flutter uses the Dart programming language.

A detail Description can be found here: **Framework**

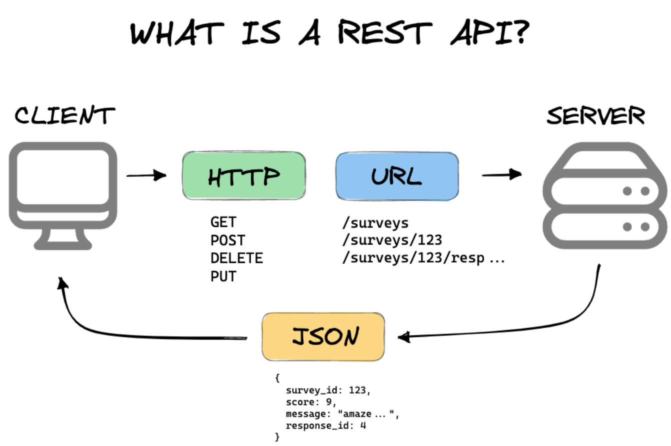
## 4. Vite

Vite is a build tool designed to accelerate and improve the efficiency of current web applications (Vite, 2023).

### 2.1.4 API

**1. RestAPI**

The integration of Application Programming Interfaces (APIs) plays a crucial role in facilitating communication between various components of the system. APIs enhance interoperability, allowing seamless data exchange between the web and mobile applications. This integration contributes to the overall efficiency and responsiveness of the system.




**2. API Testing Postman**

for the API testing we will use postman.

### 2.1.5 Deployment

1. **Amazon AWS**

To deploy our service we will use AWS.

### 2.1.6 Database

**MySQL**

MySQL is an open-source relational database management system. Its name is a combination of "My", the name of co-founder Michael Widenius's daughter My, and "SQL", the acronym for Structured Query Language

## 2.2 Methodology

**Considered Methodology**

a) Waterfall Model

b) Prototype Model

c) Agile Methodology

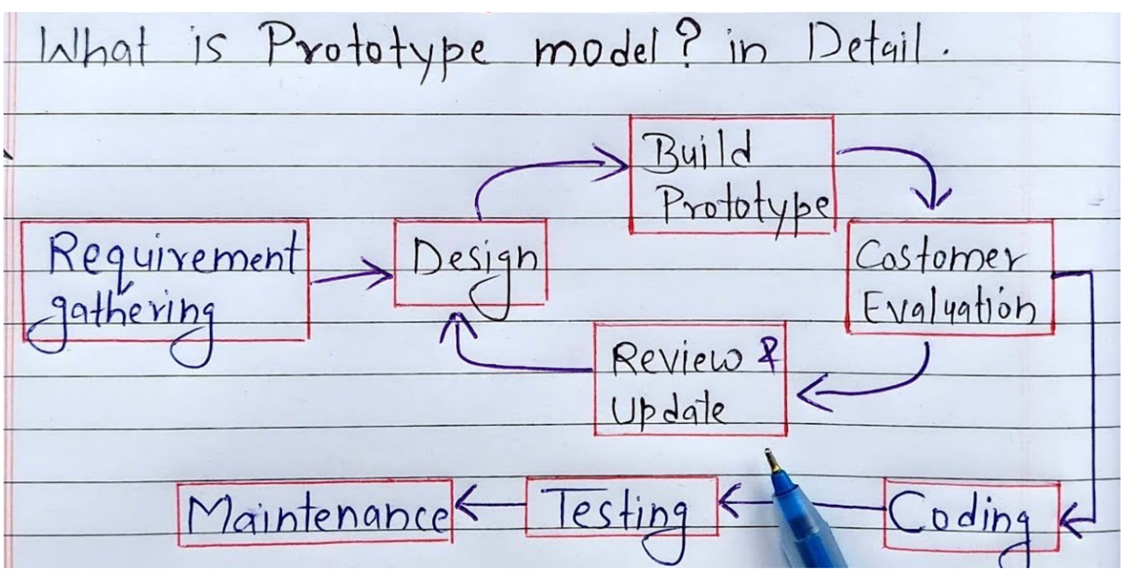
Note: The description of considered methodology are kept in appendix section

i.e. Considered Methodology.

### 2.2.1 Selected Methodology: Prototype Model

**Prototype Model**

The Prototype Model is a development methodology chosen for this project due to its iterative and user-focused approach. The key reasons for selecting the Prototype Model are outlined below:




The detail about Selected Methodology can be found in the Appemdix:

**Selected Methodology: Prototype Model**

### 2.2.2 Review of Similar Projects

There are 6 (Six) Research Papers Reviewed to see how the other model is built and how they perform.

1. **Authors: I Ketut Eddy Purnama**

   Department of Computer Engineering, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

   *Disease Classification based on Dermoscopic Skin Images Using Convolutional Neural Network in Teledermatology System* (no date) *IEEE Conference Publication | IEEE Xplore*. Available at: https://ieeexplore.ieee.org/document/8973303.

   In this research project, a novel system for the classification and detection of skin diseases is proposed, specifically tailored for application in Teledermatology. The primary objective is to leverage advanced technologies, particularly Deep Learning, with a focus on the Convolutional Neural Network (CNN) algorithm, to classify skin diseases based on dermoscopic images.

**More about this Research in Appendix: 1. Authors: I Ketut Eddy Purnama**

2. ***Authors:* Ayushi Verma**

Department of Computer Engineering and Applications, GLA University, Mathura, India

*Plant disease classification using Deep learning framework* (no date) *IEEE Conference Publication | IEEE Xplore*. Available at: https://ieeexplore.ieee.org/document/9844352.

In this report the authors writes about Agriculture stands as the primary livelihood source in India, facing significant challenges attributed to plant diseases that cause substantial devastation. This research addresses the urgent need for early disease detection in the agricultural sector, proposing an automatic system designed to identify and categorize plant diseases at their initial stages.

**More about this Research in Appendix: *2. Authors:* Ayushi Verma**

### 2.2.3 Similar Projects in Plant Disease Classification

**1. Authors:** Mohanty, S. P

Hughes, D. and Salathé, M. (2016) "Using deep learning for Image-Based plant disease detection," *Frontiers in Plant Science*, 7. doi: 10.3389/fpls.2016.01419.

In this research Paper the authors talks about Crop diseases pose a significant threat to global food security, exacerbated by challenges in rapid identification, particularly in regions lacking adequate infrastructure. The intersection of widespread smartphone adoption and advancements in computer vision, propelled by deep learning, opens avenues for transformative smartphone-assisted disease diagnosis. Leveraging a comprehensive dataset of 54,306 images capturing both diseased and healthy plant leaves in controlled conditions, we employ a deep convolutional neural network for the identification of 14 crop species and detection of 26 diseases or their absence. sMehra, 2016)

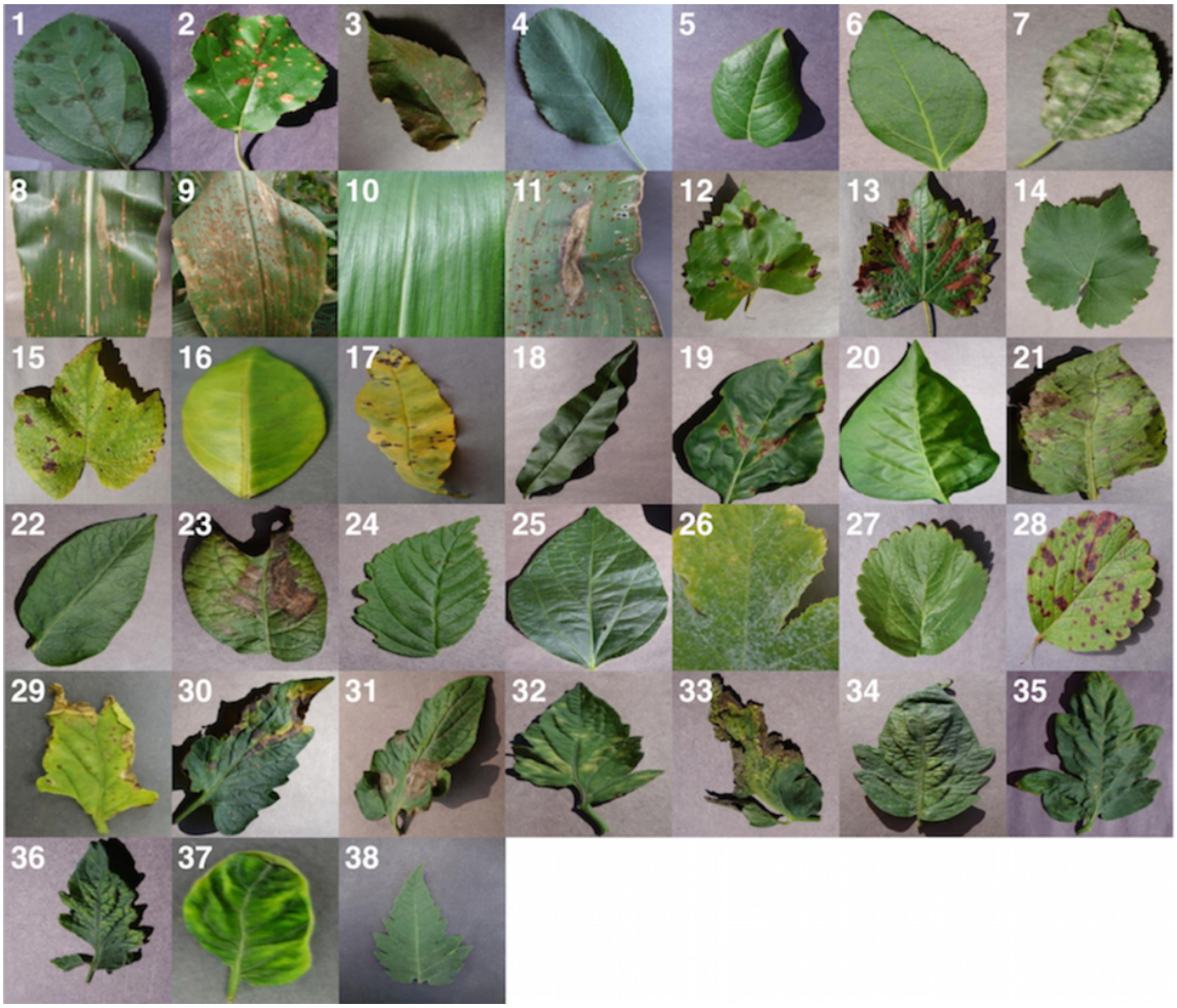



*Table 2: EXAMPLE OF LEAF IMAGES FROM THE PLANTVILLAGE DATASET, REPRESENTING EVERY CROP-DISEASE PAIR USED*

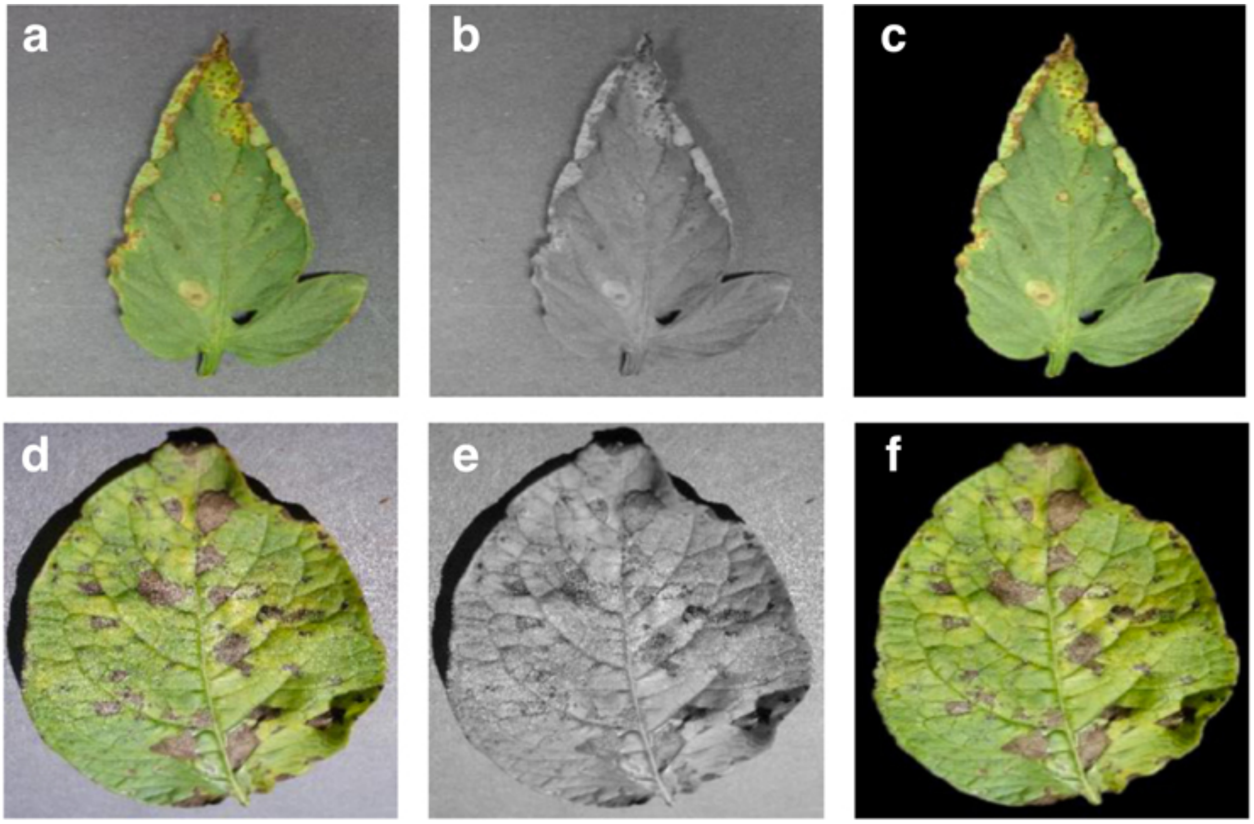


*Table 3: SAMPLE IMAGES FROM THE THREE DIFFERENT VERSIONS OF THE PLANTVILLAGE DATASET USED IN VARIOUS EXPERIMENTAL CONFIGURATIONS*

**1. Choice of deep learning architecture:**

AlexNet,

GoogLeNet.

**2. Choice of training mechanism:**

Transfer Learning,

Training from Scratch.

**3. Choice of dataset type:**

Color,

Gray scale,

Leaf Segmented.

**4. Choice of training-testing set distribution:**

Train: 80%, Test: 20%,

Train: 60%, Test: 40%,

Train: 50%, Test: 50%,

Train: 40%, Test: 60%,

Train: 20%, Test: 80%.

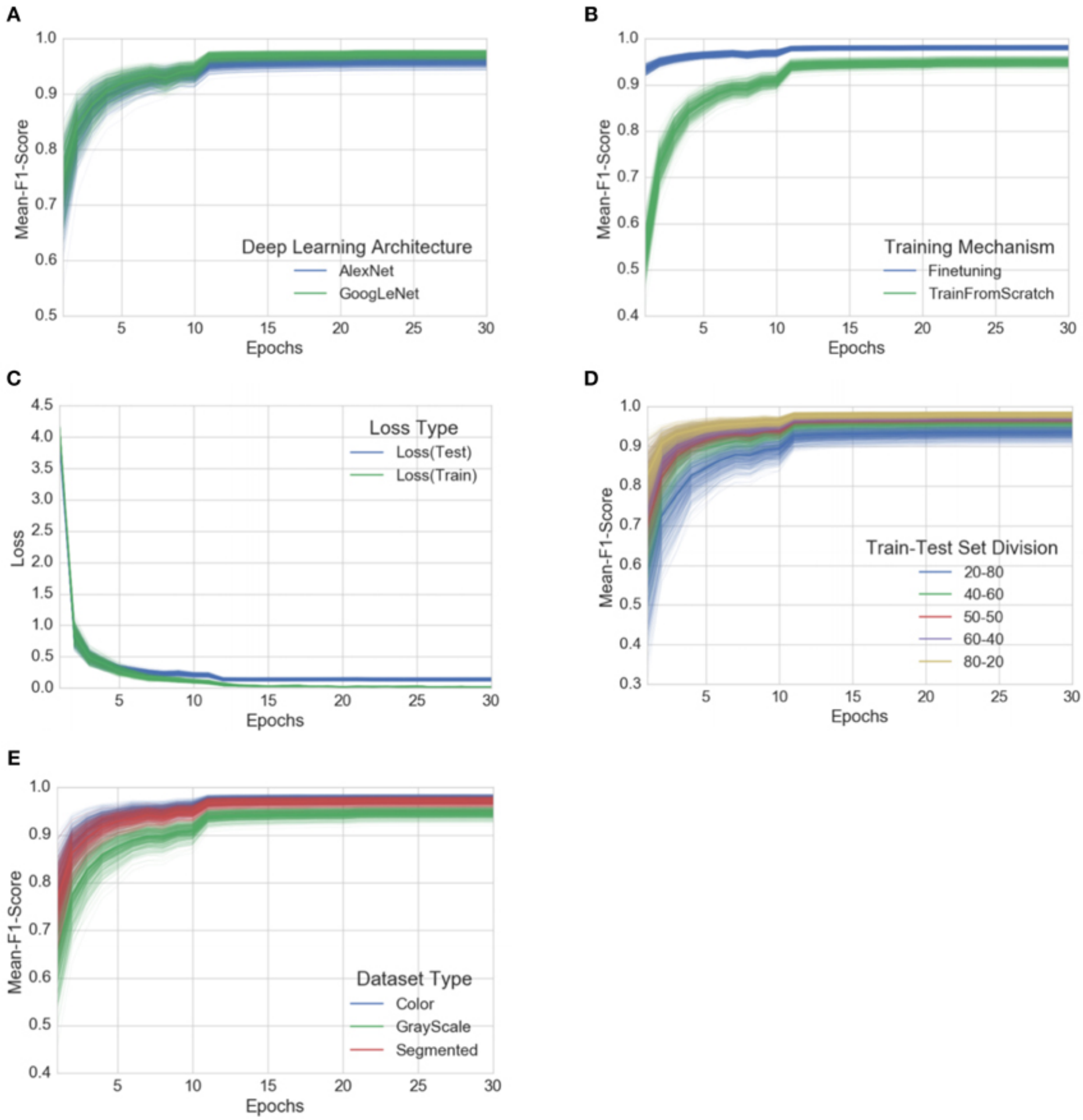



*Table 4: PROGRESSION OF MEAN F1 SCORE AND LOSS THROUGH THE TRAINING PERIOD OF 30 EPOCHS ACROSS ALL EXPERIMENTS, GROUPED BY EXPERIMENTAL CONFIGURATION PARAMETERS*

**2. Authors: Mahlein, A.-K. (2016) "Plant disease detection by imaging sensors – parallels and specific demands for precision agriculture and plant phenotyping,"** ***Plant Disease*****, 100(2), pp. 241–251. doi: 10.1094/pdis-03-15-0340-fe.**

In this report the authors talks about The timely and precise identification of plant diseases is critical for optimizing plant production and minimizing losses in both quality and quantity of crop yields. Optical techniques, encompassing RGB imaging, multi- and hyperspectral sensors, thermography, chlorophyll fluorescence, and more recently, 3D scanning, have demonstrated their efficacy in developing automated, objective, and reproducible detection systems. These technologies contribute to the identification and quantification of plant diseases at early stages of epidemics. (Singh, 2016)

**More about this Research in Appendix:** [2. Authors: Mahlein, A.-K. (2016)]

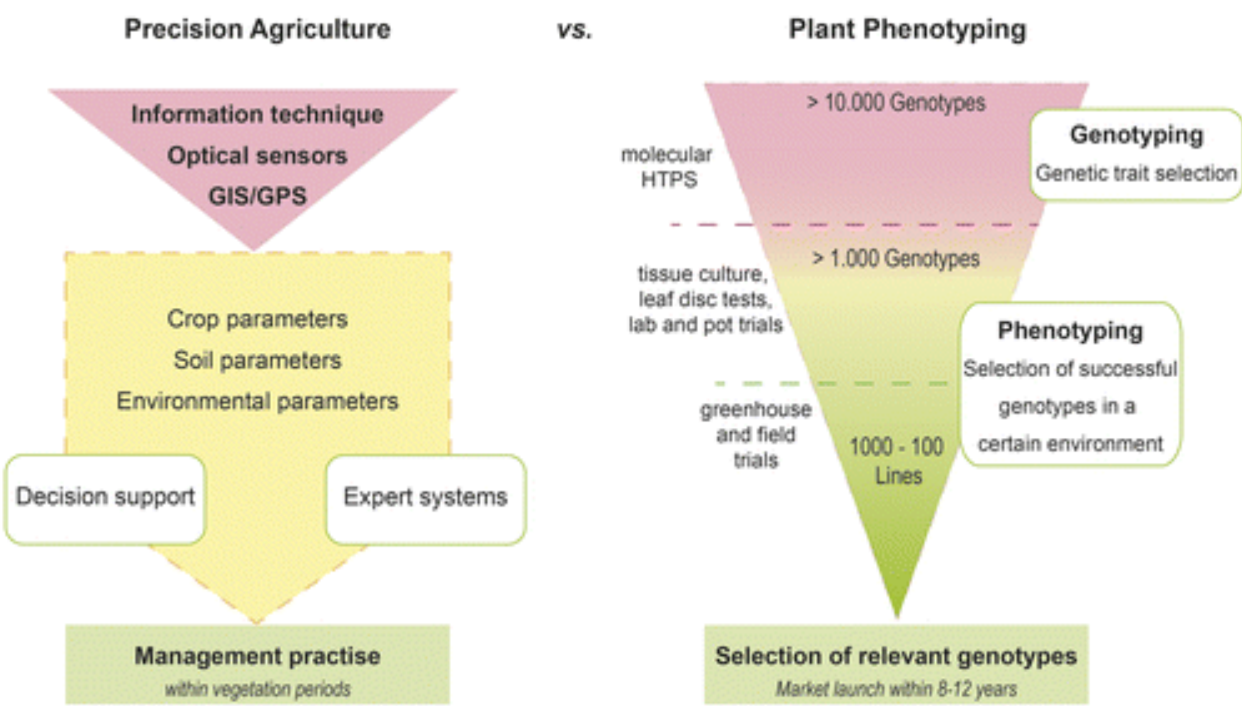



*Figure 4: Schematic diagrams of workflows and parameters in precision agriculture (left) and plant phenotyping (right).*

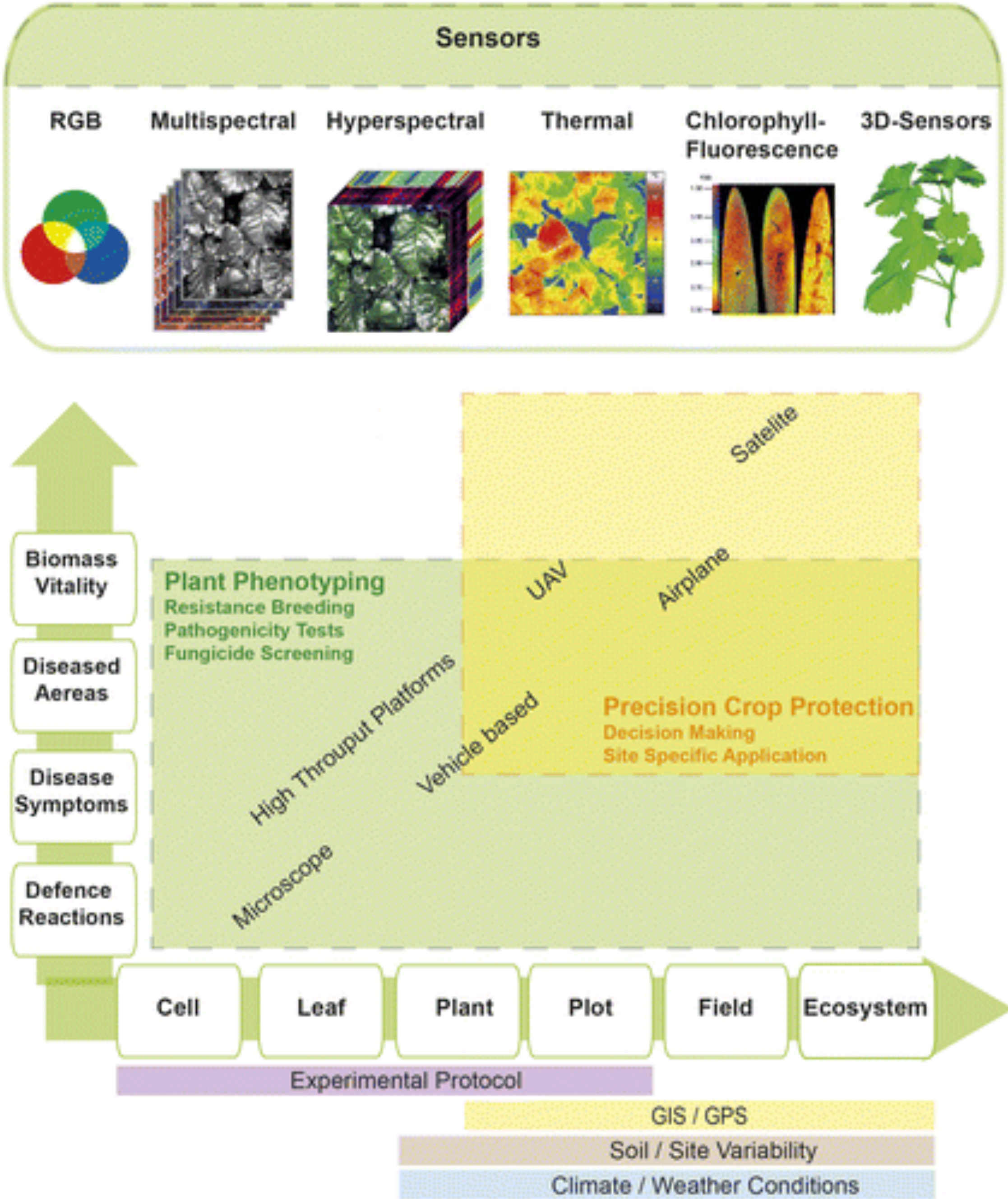



*Figure 5: Overview of current sensor technologies used for the automated detection and identification of host-plant interactions. These sensors can be implemented in precision agriculture applications and plant phenotyping on different scales from single cells*

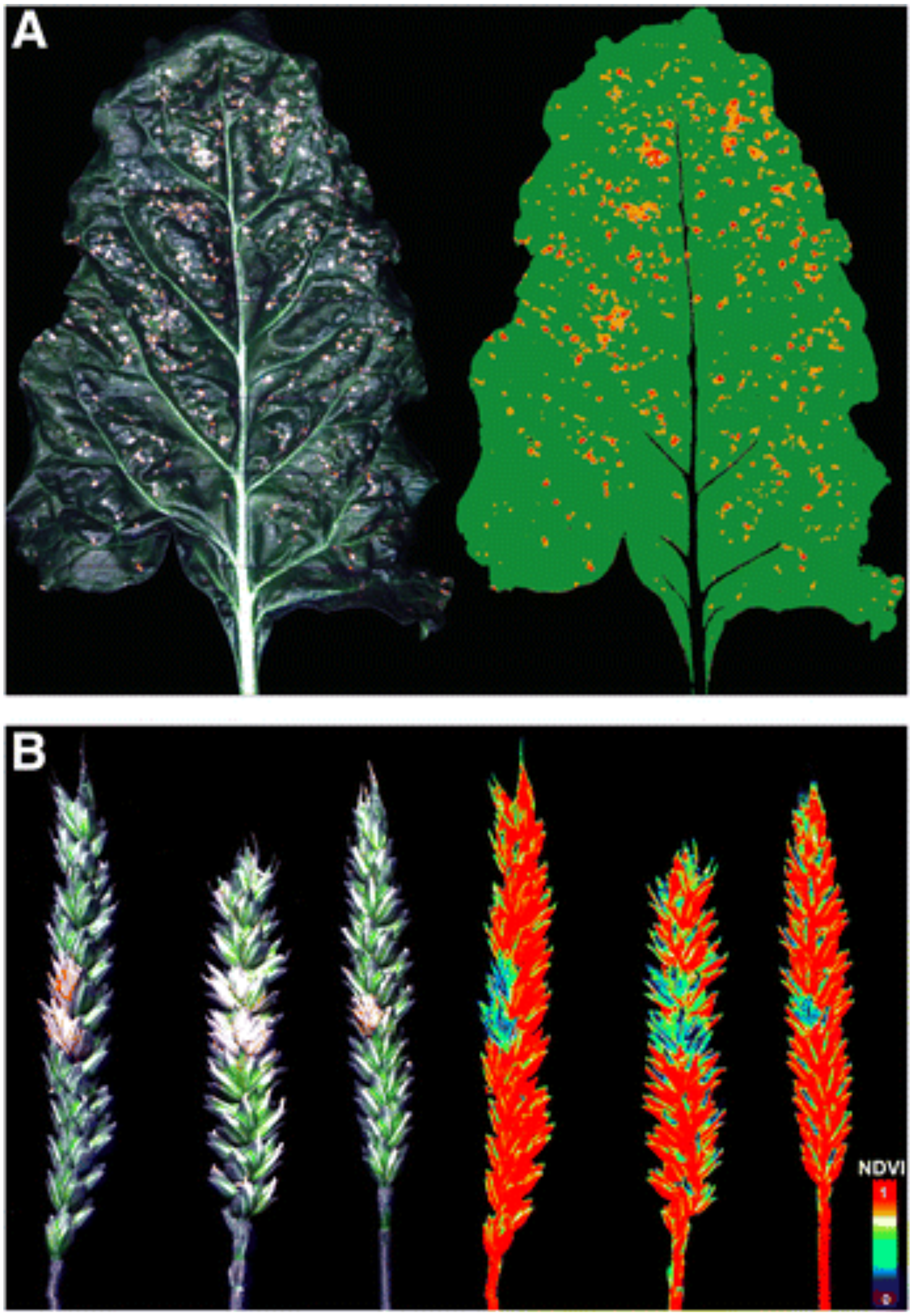



*Figure 6: Disease detection of fungal plant diseases based on hyperspectral images. A, Supervised classification (spectral angle mapper) of Cercospora leaf spot on sugar beet. The green color denotes healthy leaf tissue, the yellow color the border of Cercospora*

**Thermal sensors.**

Infrared thermography (IRT) assesses plant temperature and is correlated with plant water status (**Jones et al. 2002**), the microclimate in crop stands (**Lenthe et al. 2007**), and with changes in transpiration due to early infections by plant pathogens (**Oerke et al. 2006**). Emitted infrared radiation in the thermal infrared range from 8 to 12 µm can be detected by thermographic and infrared cameras and is illustrated in false color images, where each image pixel contains the temperature value of the measured object. In plant science, IRT can be used at different temporal and spatial scales from airborne to small scale applications.

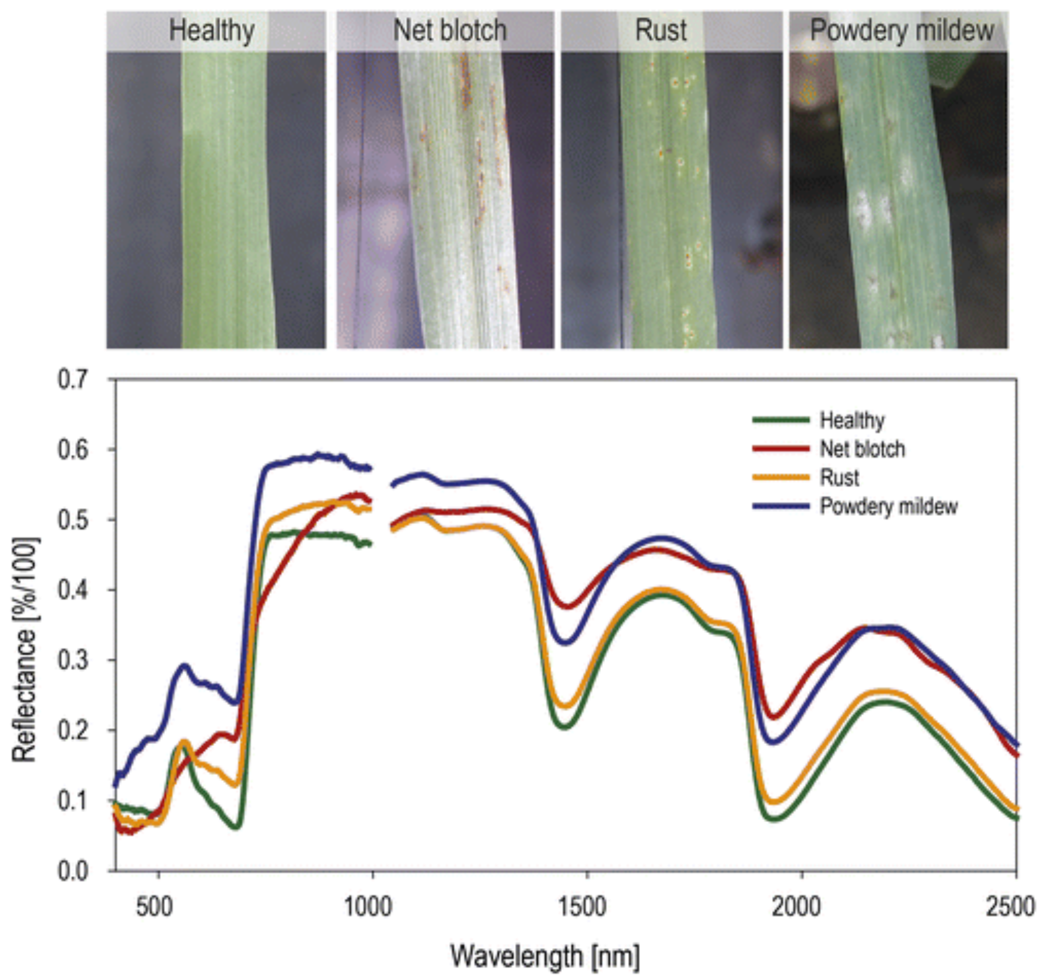



*Figure 7: Characteristic spectral signatures of barley leaves diseased with net blotch, rust, and powdery mildew, respectively.*

## 3. Plant Village

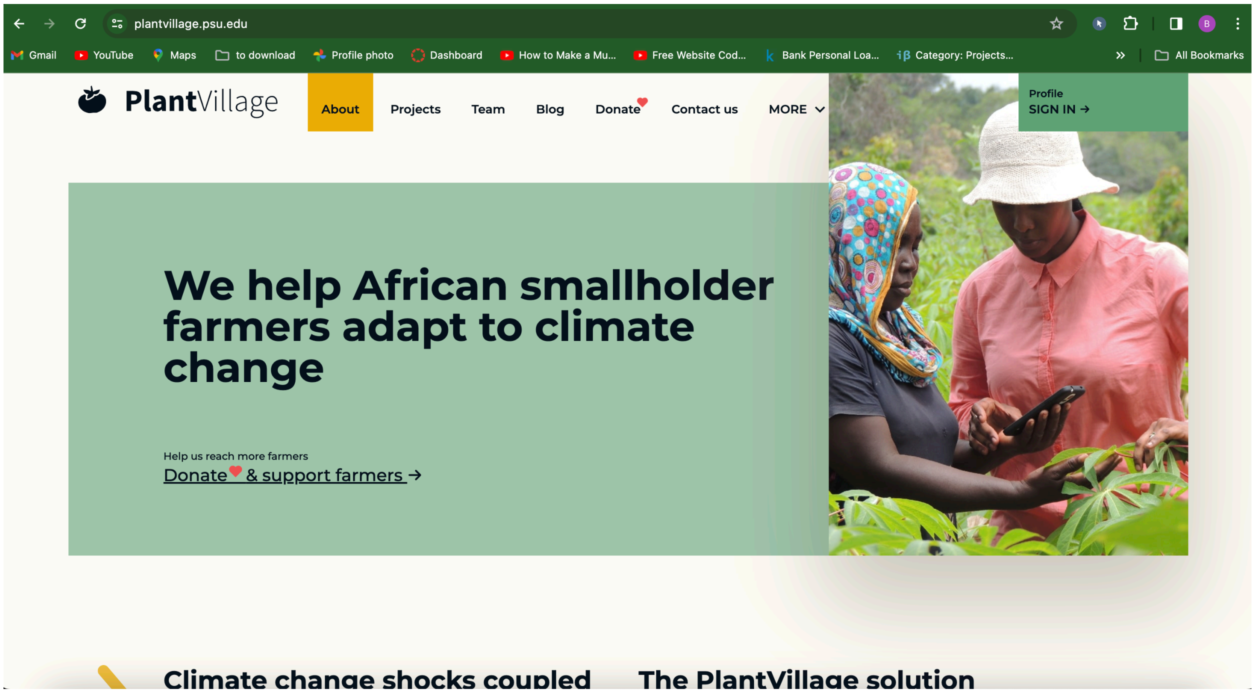



*Figure 8: Plant Village*

*PlantVillage* (no date). Available at: https://plantvillage.psu.edu/.

PlantVillage has developed a triple A model (Algorithmic Agricultural Advice) that works to increase the yield and profitability for millions of farmers. It is our goal to reach hundreds of millions in partnership with an ecosystem of farmer facing organizations and the farmers themselves. Our algorithms come from our integration of AI, satellite technology and our unique field force (the Dream Team). Once a farmer inputs 3 critical details (crop type, location, planting date) the algorithms within the PlantVillage engine can send out advice via smartphone, SMS, TV or real world social networks. (Kaur, 2019).

### 2.2.4 Comparison Between Systems

| | FarmAssist- Plant Disease Classification | Plant disease detection by imaging sensors | Plant disease classification using Deep learning framework | PlantVillage | Disease Classification based on Dermoscopic Skin Images Using Convolutional Neural Network in Teledermatology System |
|---|---|---|---|---|---|
| **Front End** | Yes | NO | NO | Yes | NO |
| **Backend** | Yes | NO | NO | NO | NO |
| **ML Model** | Yes | Yes | Yes | NO | Yes |
| **Language** | Javascript,Python | Python | Python | Javascript | Python |
| **Framework** | React | NO | NO | NO | NO |
| **Model** | CNN,Resnet8 | CNN | CNN | NO | Teledermatology |
| **Backend Framework** | Django | NO | NO | NO | NO |
| **MobileAPP** | Yes, Flutter | NO | NO | NO | NO |
| **Deployment** | YES, AWS | NO | NO | NO | NO |
| **Database** | YES | NO | NO | YES | NO |

*Table 5: Comparison Between Systems*

# 3 Development Progress

## 3.1 Requirement Gathering

The requirements for the Farm Assist - Plant Disease Classification System were systematically gathered through a combination of surveys conducted both online and on-site. The data collection process took place from Dec 1 to Jan 1, 2024, employing various methodologies to ensure comprehensive coverage.

### 4.1.1 Online Survey with the Public

An online survey was conducted using Google Forms to reach a diverse audience, including farmers, agricultural enthusiasts, and potential users of the Plant Disease Classification System. The survey aimed to gather insights into user preferences, expectations, and challenges related to plant disease diagnosis.

**Survey Duration:** Dec 1 to Jan 1, 2024

**Participants:** A total of 203 individuals responded to the online survey.

### 3.1.1 Data Collection for Training

The data is collected from various sources like: Kaggle, PlantVillage.

### 3.1.2 User Needs Assessment

To ensure the Farm Assist - Plant Disease Classification System aligns with the needs and expectations of its users, a comprehensive assessment of user needs was conducted. This assessment encompassed both online and providing valuable insights into the preferences and challenges faced by potential users.

#### 4.1.2.1 Online Survey Insights

The online survey, conducted from December 1 to Jan 3, 2024, targeted a diverse audience, including farmers, agricultural experts, and individuals interested in plant disease diagnosis. The responses from 203 participants shed light on several critical aspects.

## 3.2 Survey

The results of survey conducted are listed in Appendix:

From the survey we get the total of 203 responses.

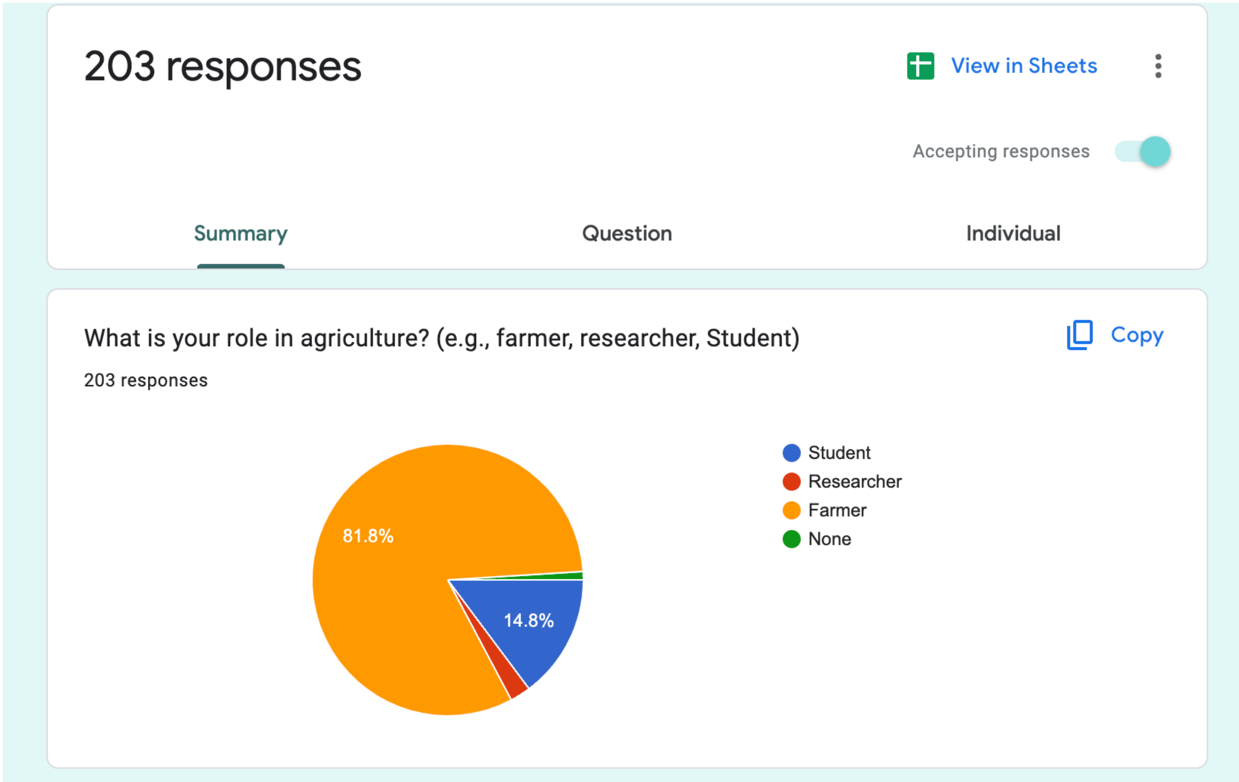



*Figure 9: What is your role in agriculture? (e.g., farmer, researcher, Student)*

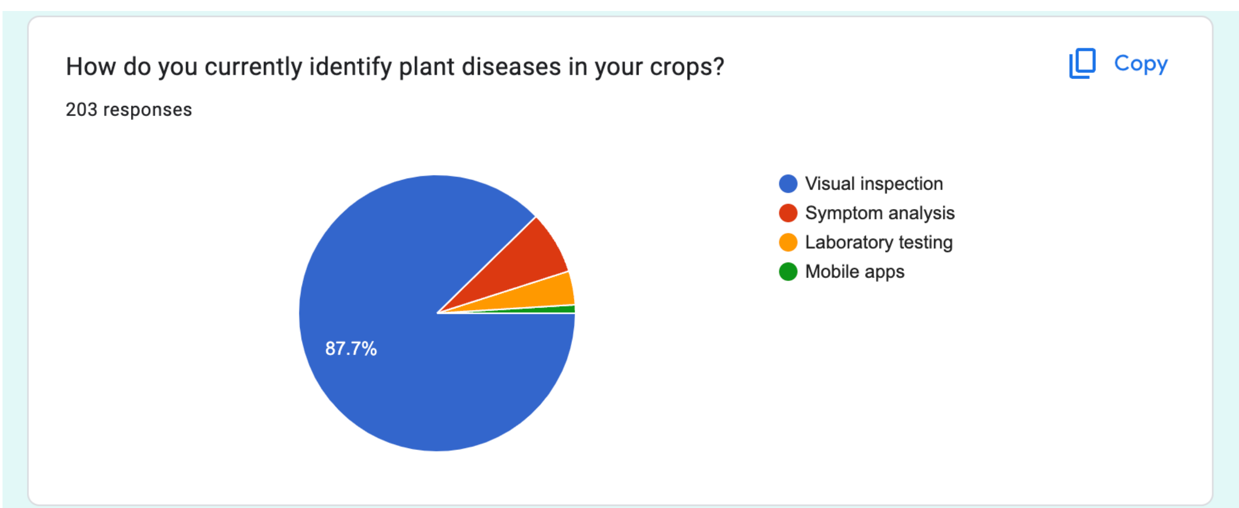



*Figure 10: How do you currently identify plant diseases in your crops?*

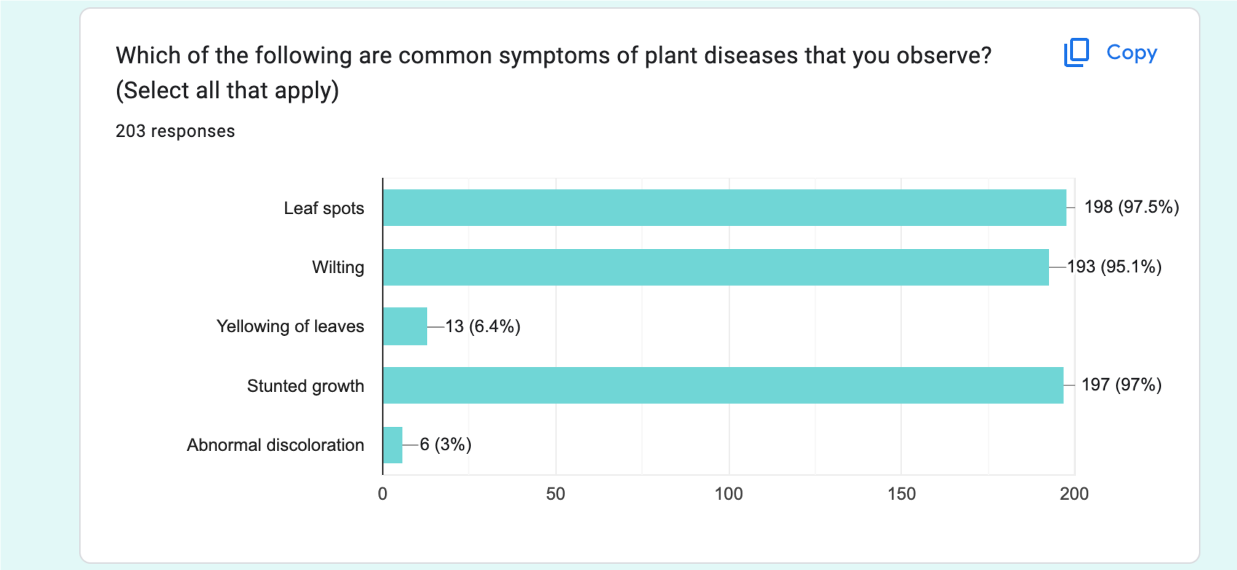



*Figure 11: Which of the following are common symptoms of plant diseases that you observe? (Select all that apply)*

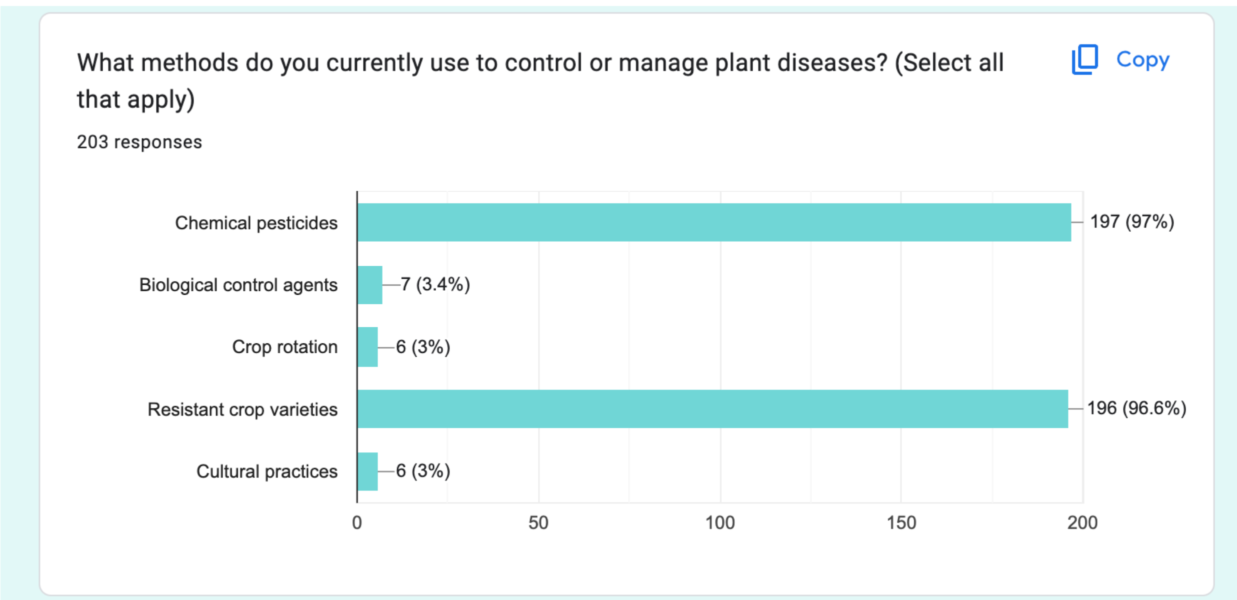



*Figure 12: What methods do you currently use to control or manage plant diseases? (Select all that apply)*

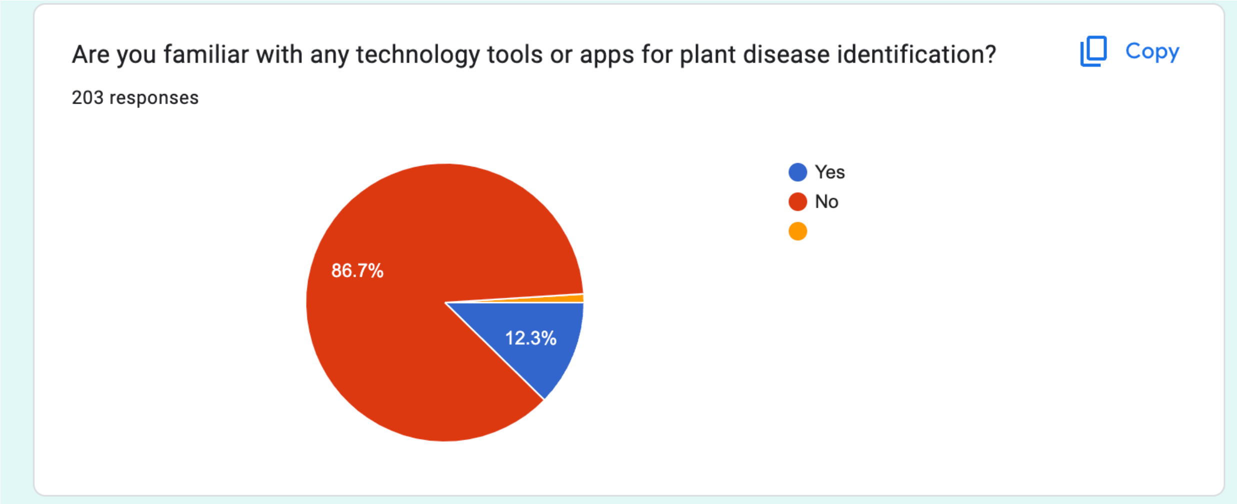


Figure 13: Are you familiar with any technology tools or apps for plant disease identification?

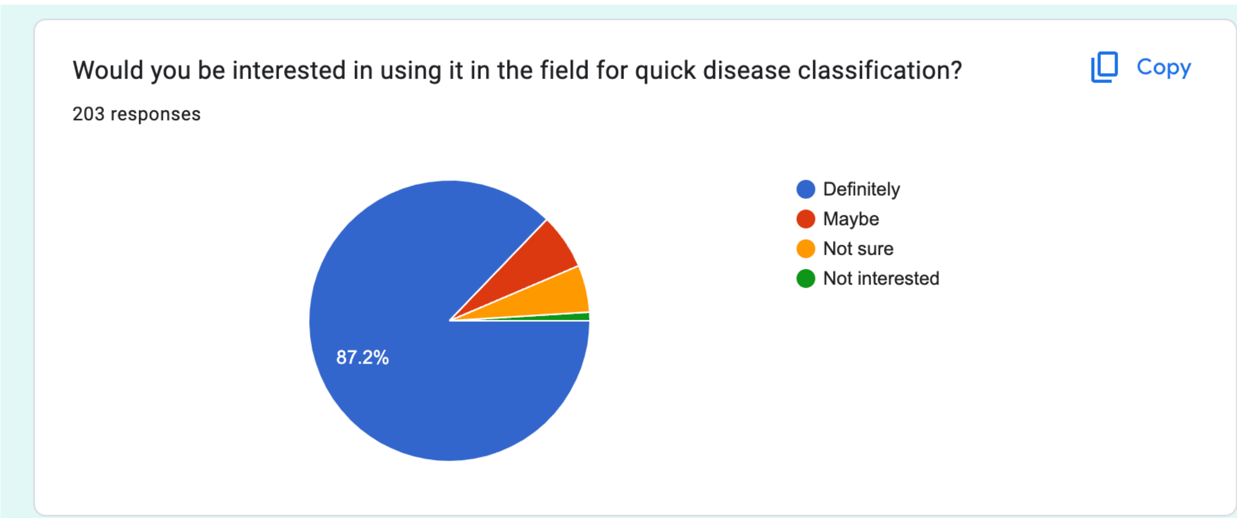


Figure 14: Would you be interested in using it in the field for quick disease classification?

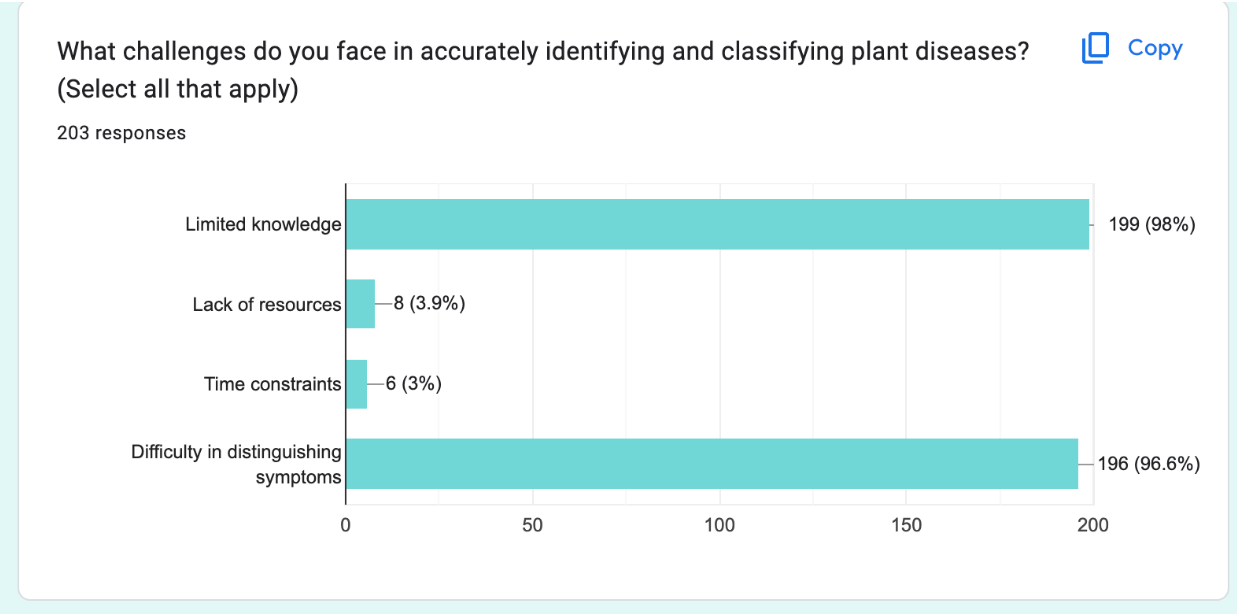



*Figure 15: What challenges do you face in accurately identifying and classifying plant diseases? (Select all that apply)*

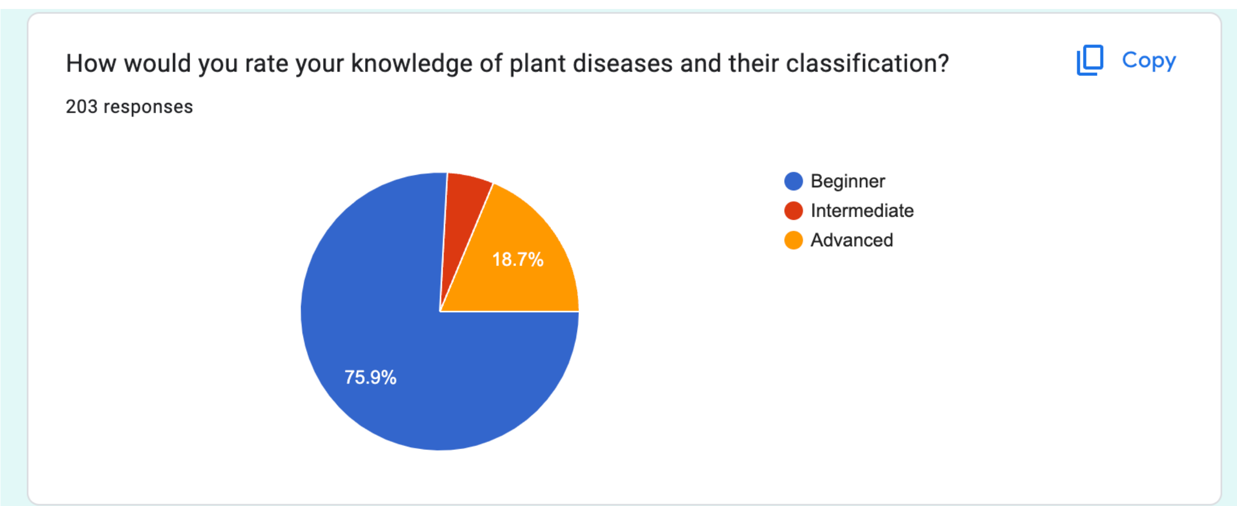



*Figure 16: How would you rate your knowledge of plant diseases and their classification?*

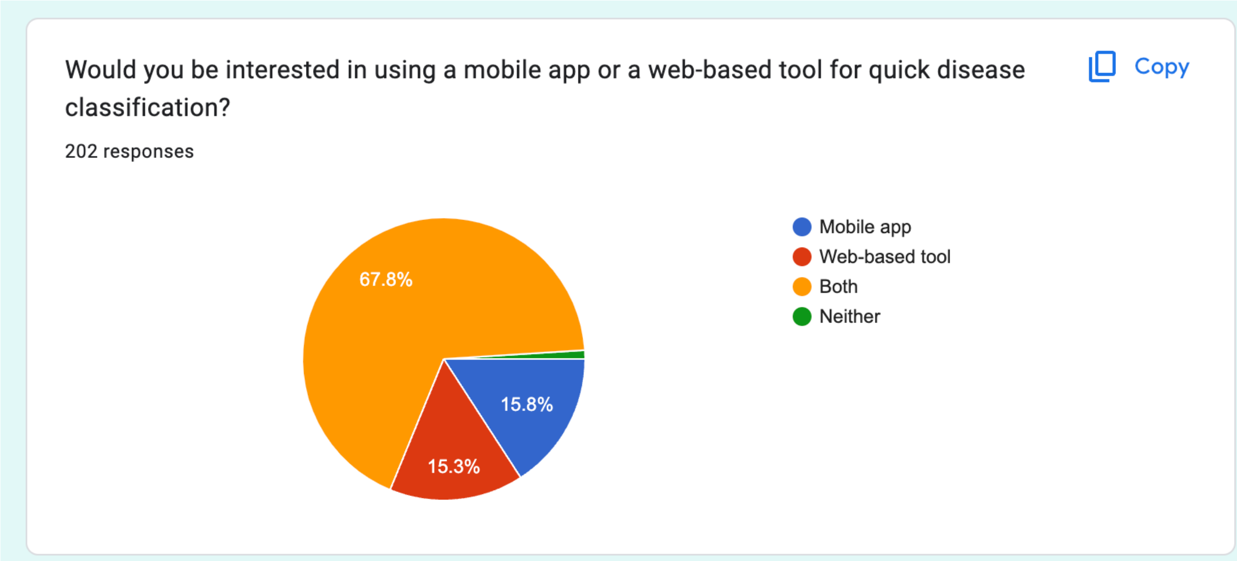



*Figure 17: Would you be interested in using a mobile app or a web-based tool for quick disease classification?*

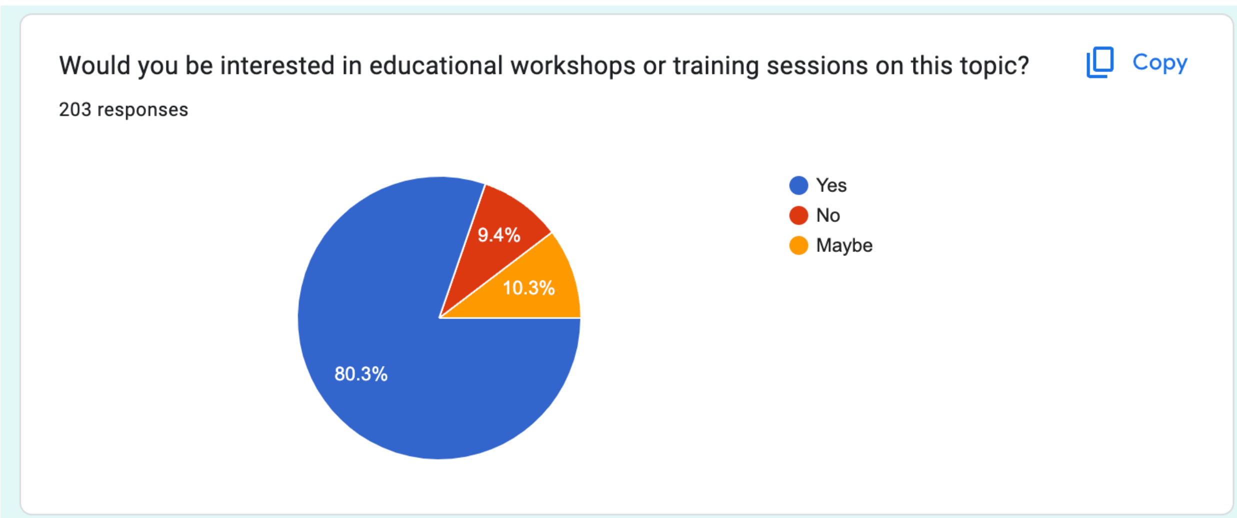



*Figure 18: Would you be interested in educational workshops or training sessions on this topic?*

## 3.3 Use Case

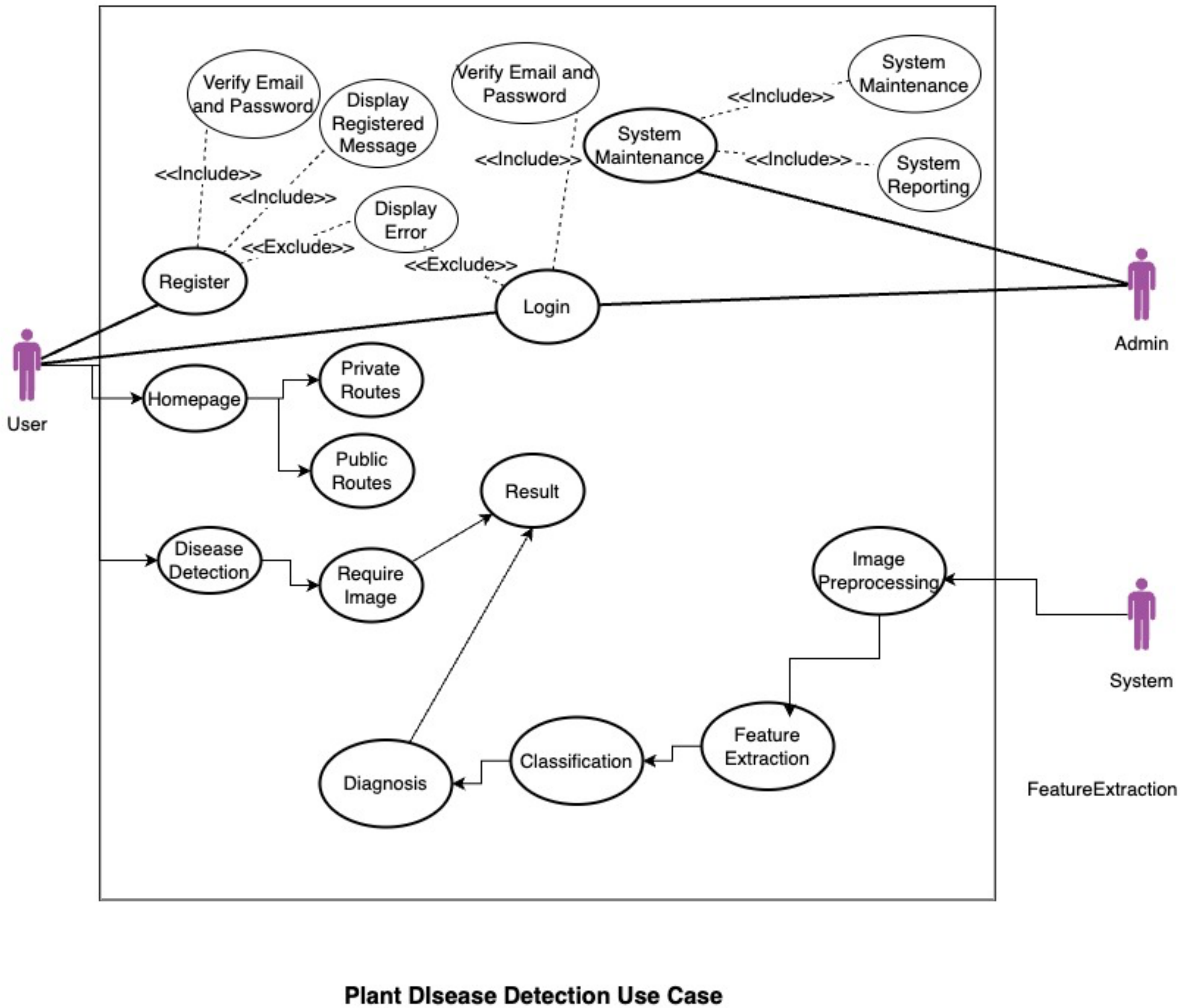



*Figure 19: Overall System Use Case Diagram of FarmAssist- Plant Disease Classification*

## 3.4 System Flow chart

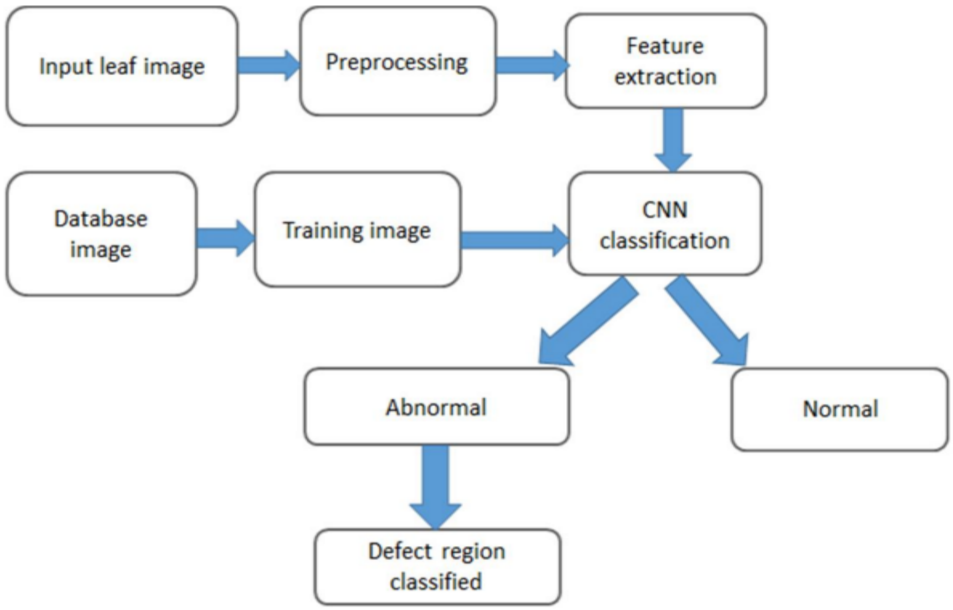



*Figure 20: System Flow Chart*

## 3.5 Activity Diagram

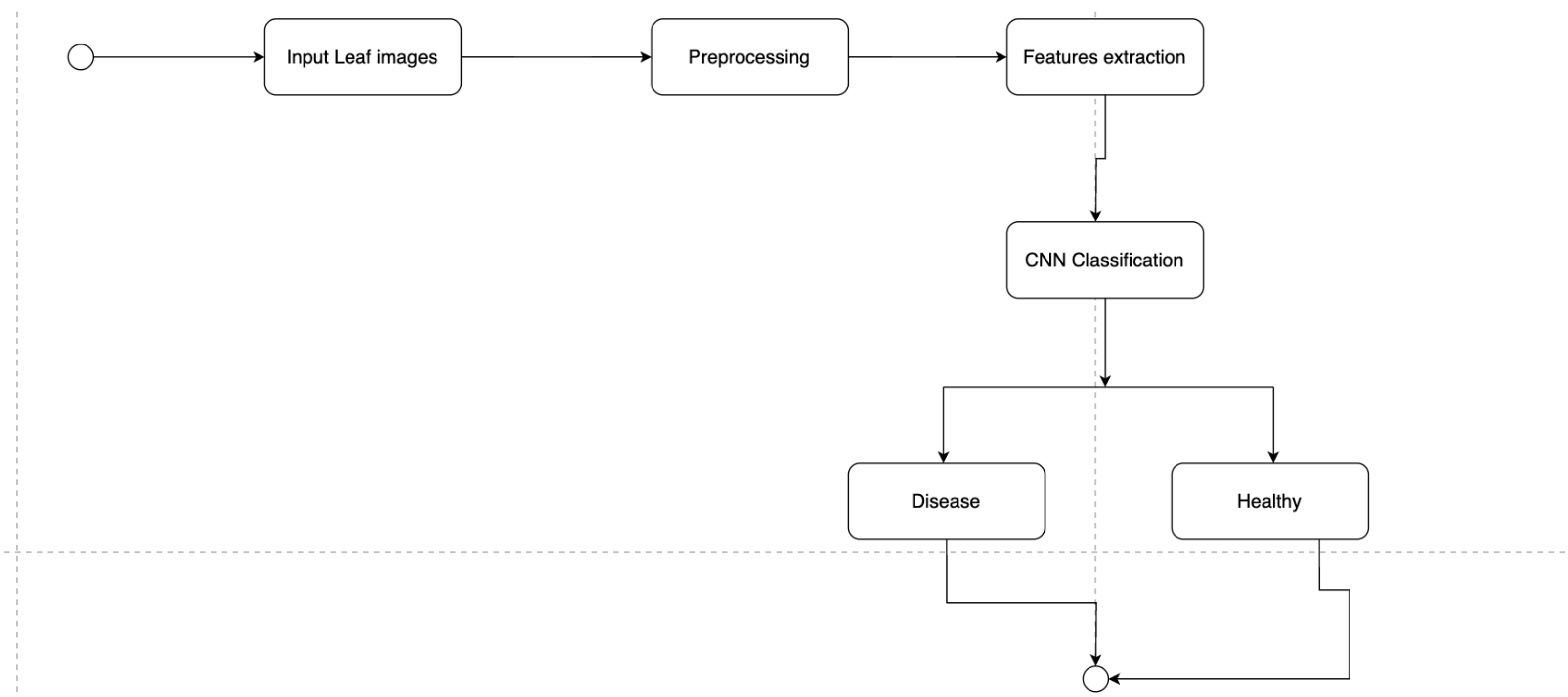



*Figure 21: Activity Diagram*

### 3.6 Designed Logo for Application

s

### 3.7 SRS Documents

The full description of SRS document is kept in Appendix section i.e. SRS Document

## 3.8 Wireframe:

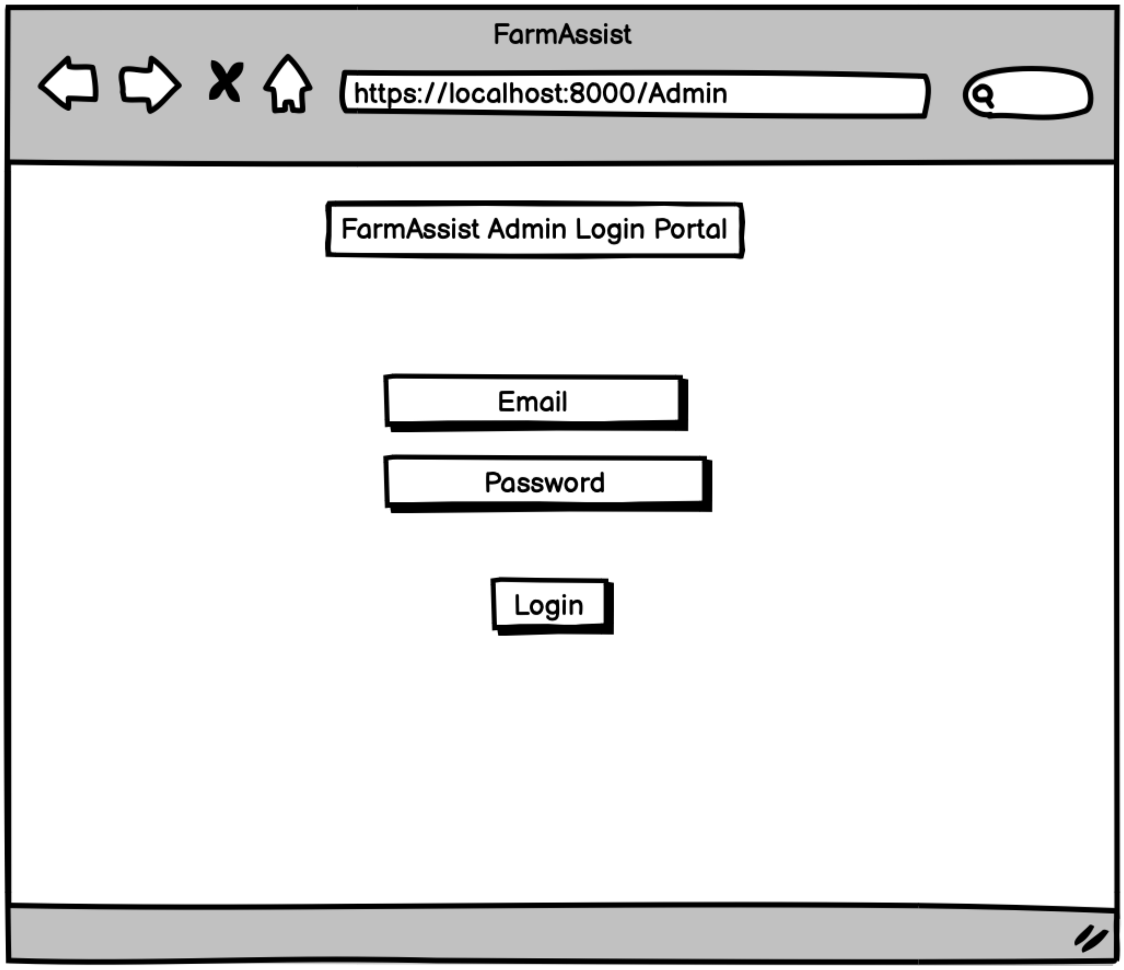



*Figure 22: Wireframe-Admin Login*

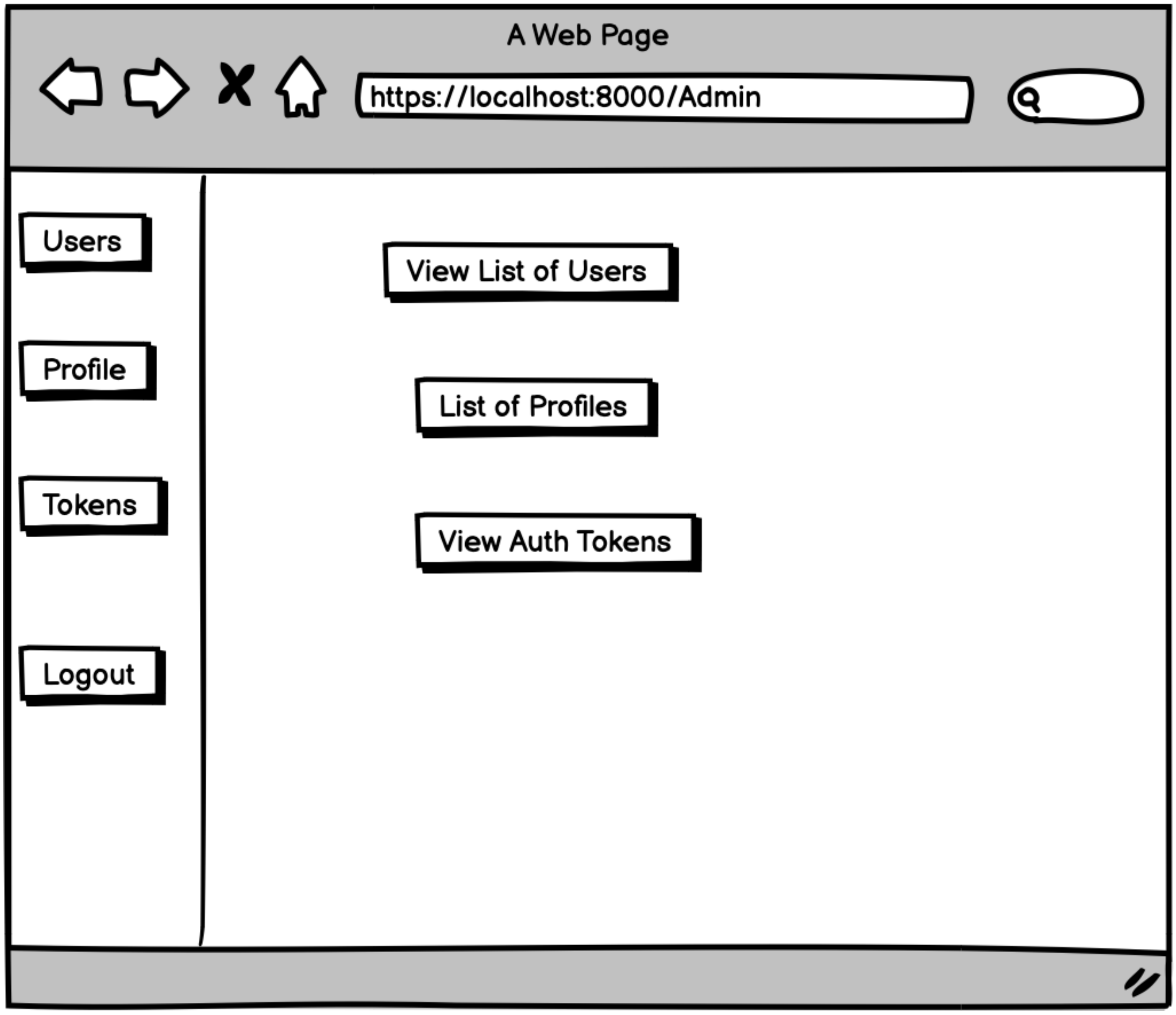



*Figure 23: Wireframe-Admin Dashboard*

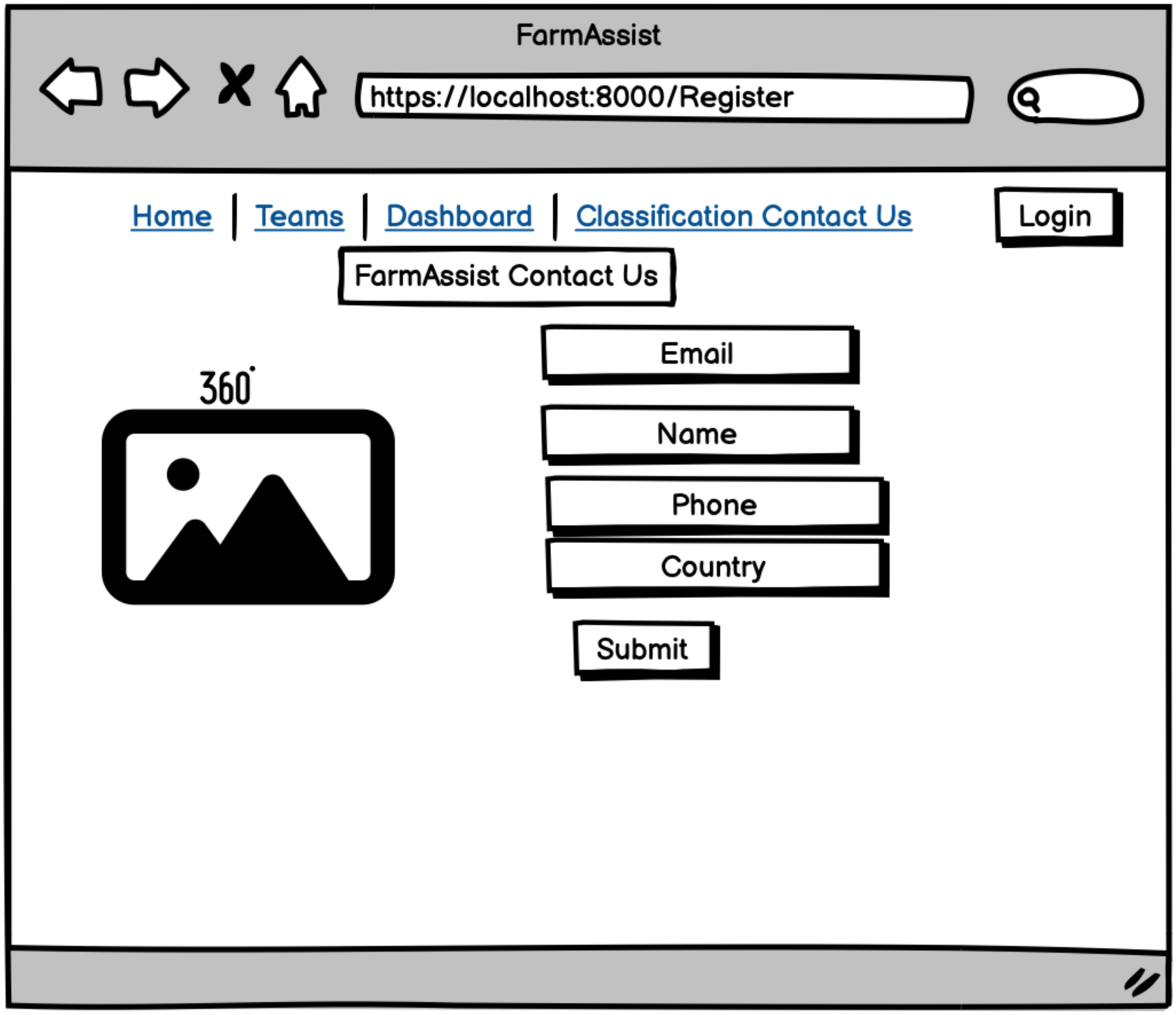



*Figure 24: Wireframe-user Contact Us*

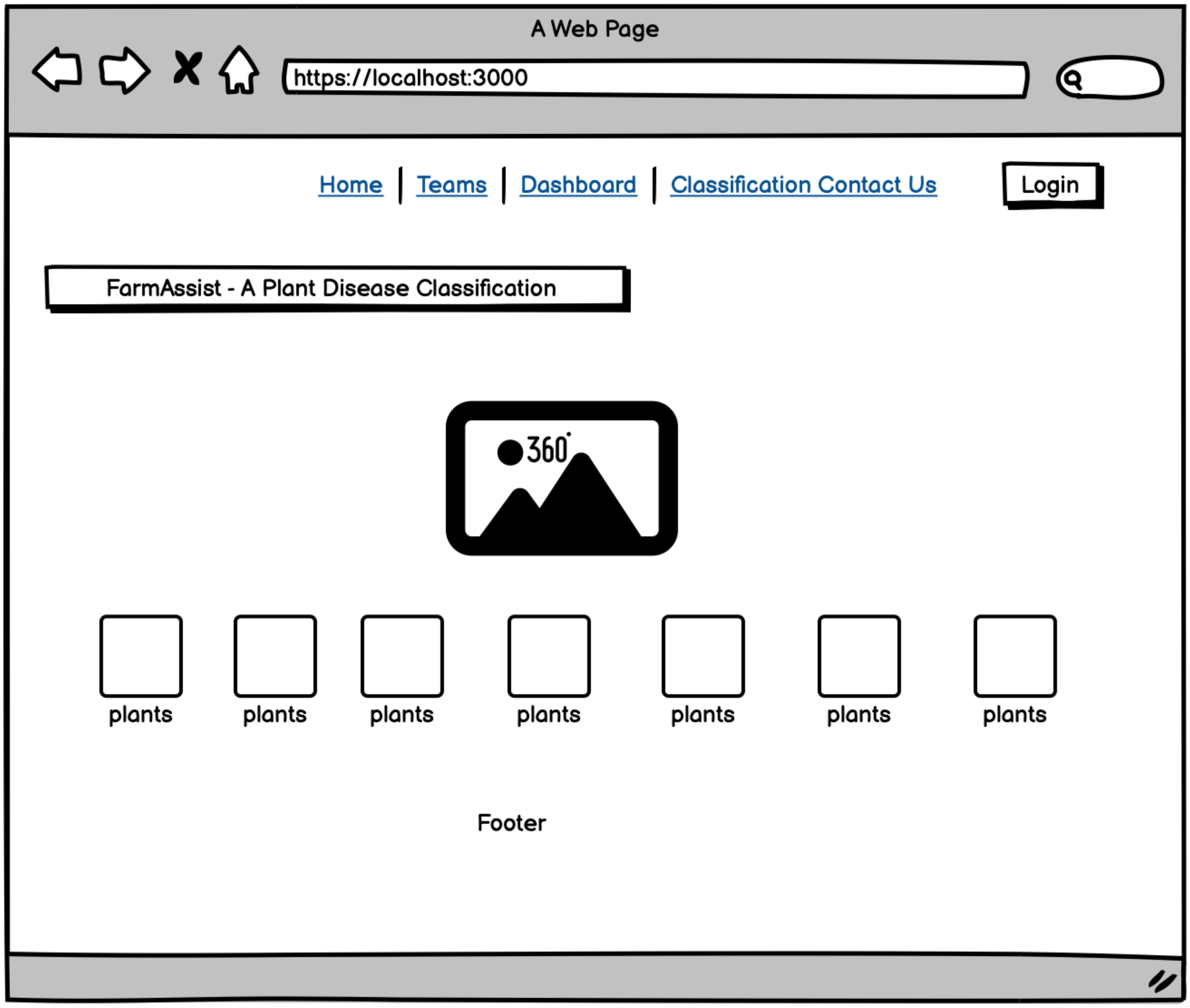



*Figure 25: Wireframe-user Dashboard*

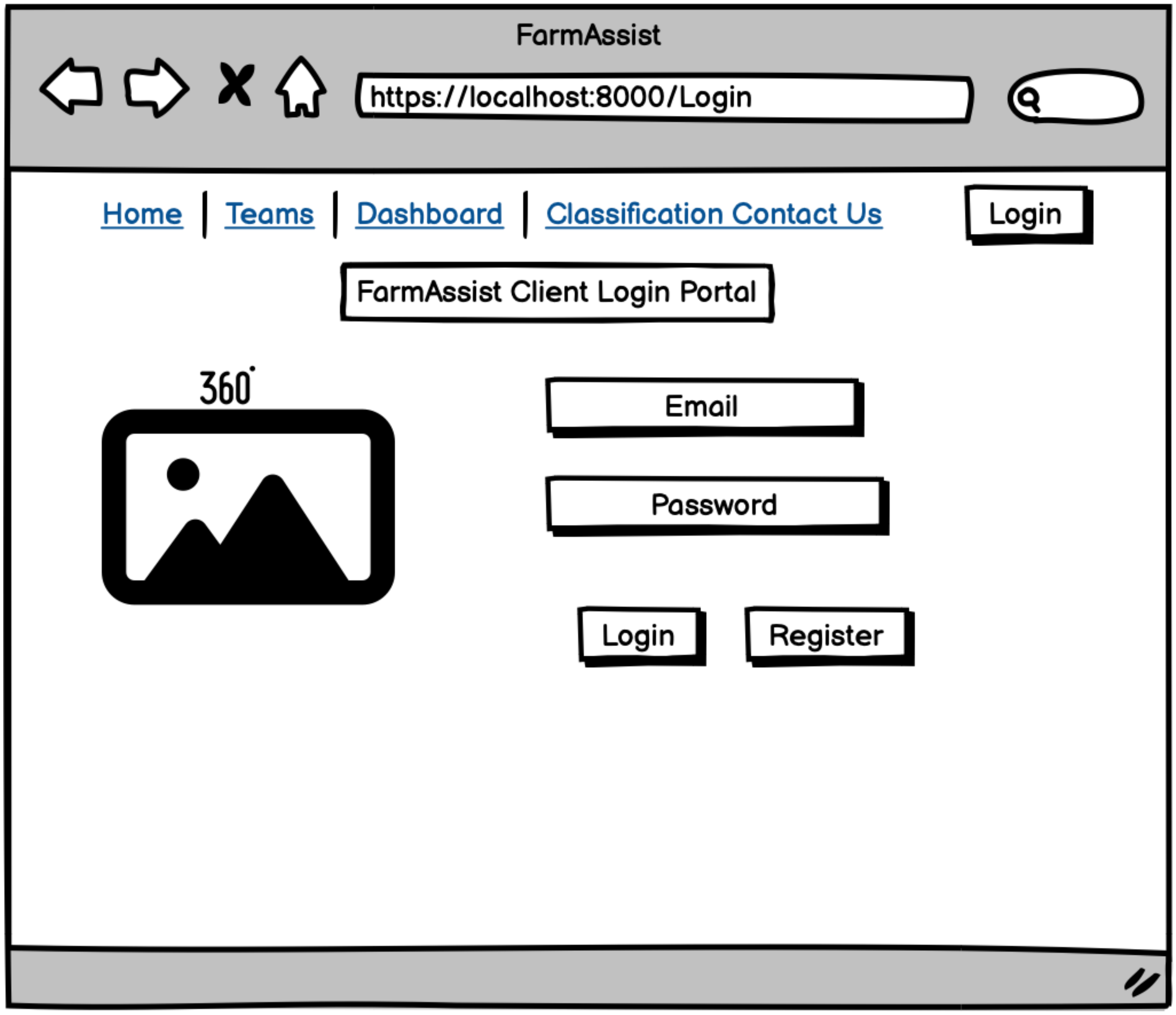



*Figure 26: Wireframe-User-login*

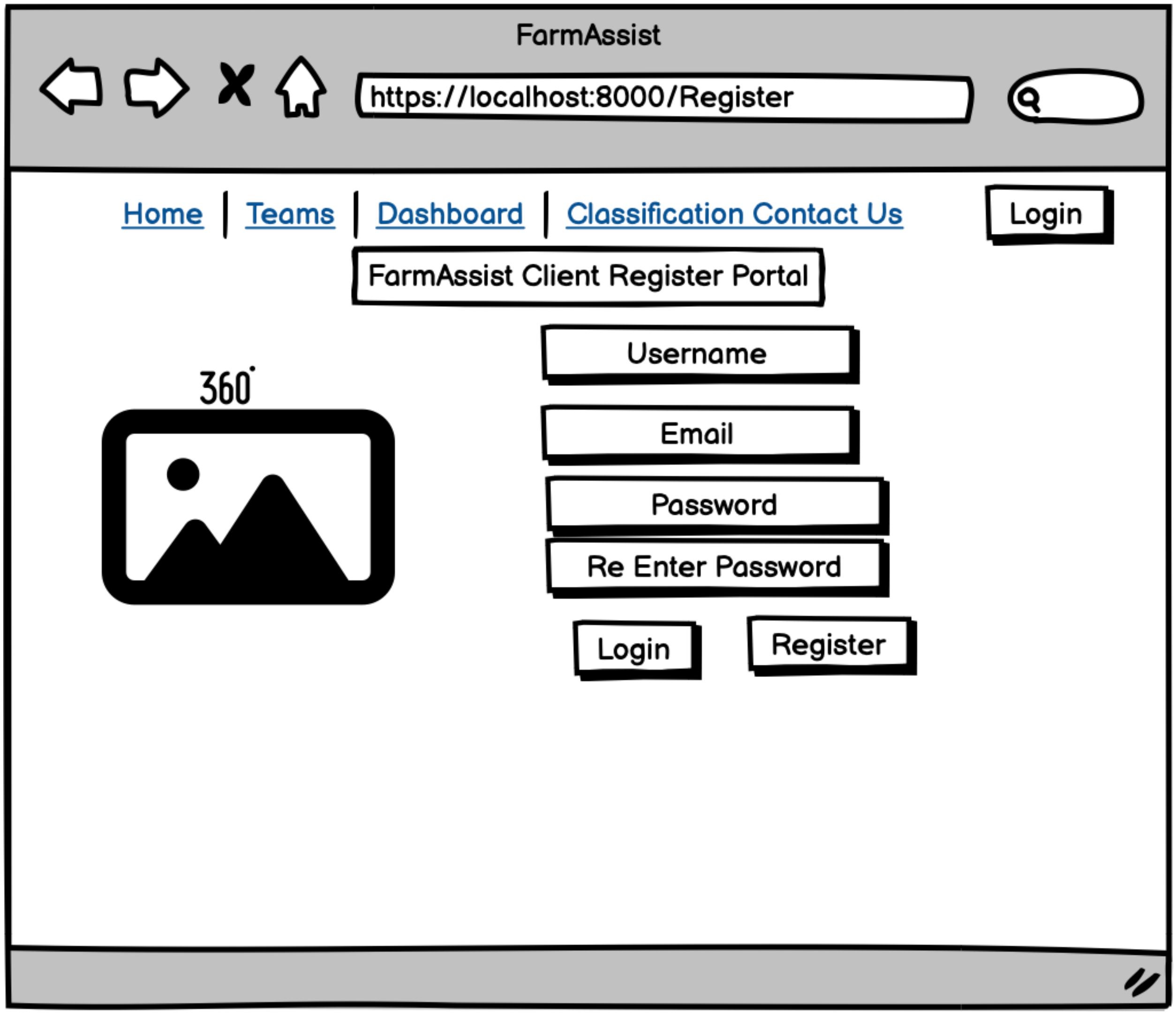



*Figure 27: Wireframe-User-Register*

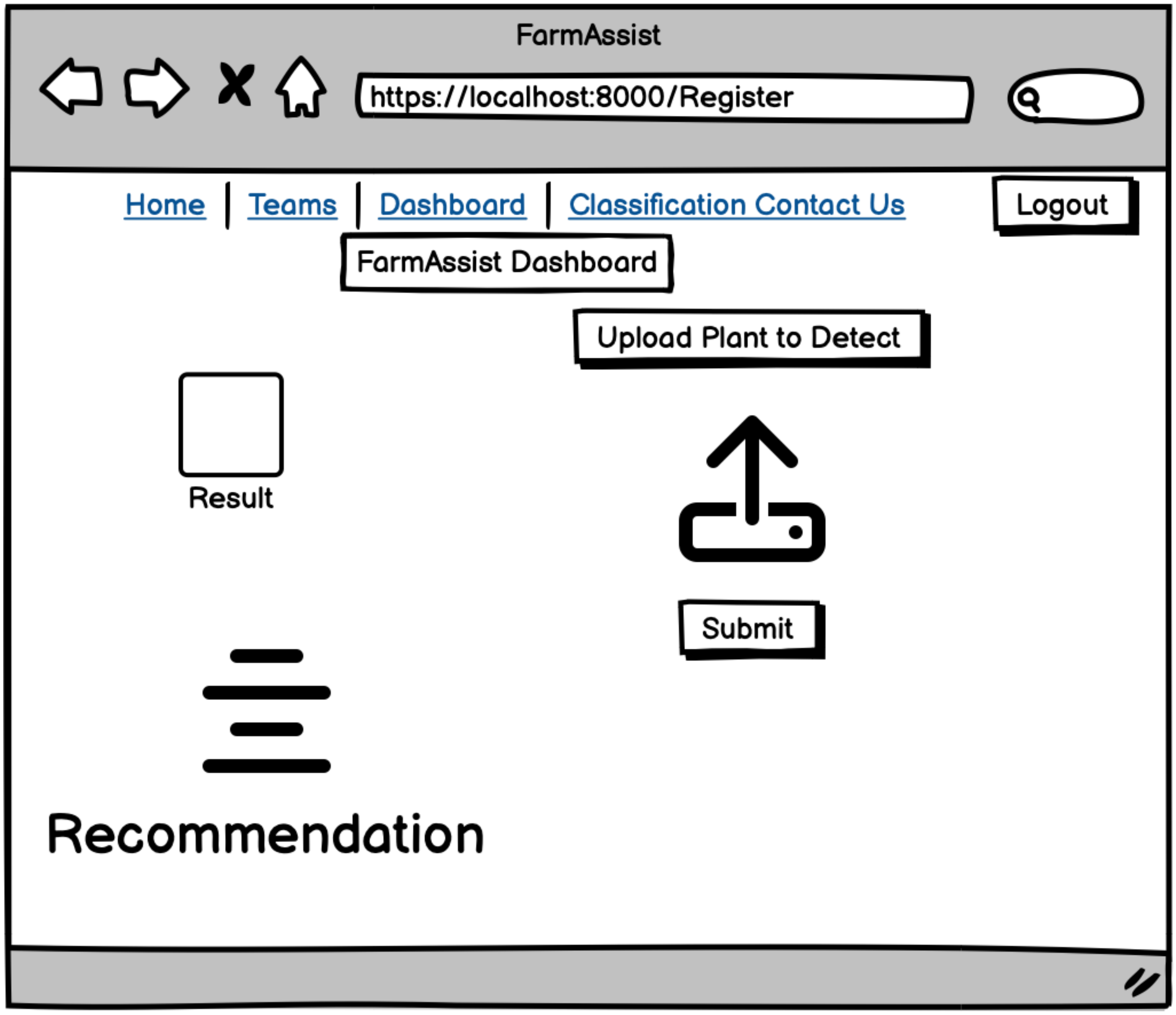



*Figure 28: Wireframe-Detection*

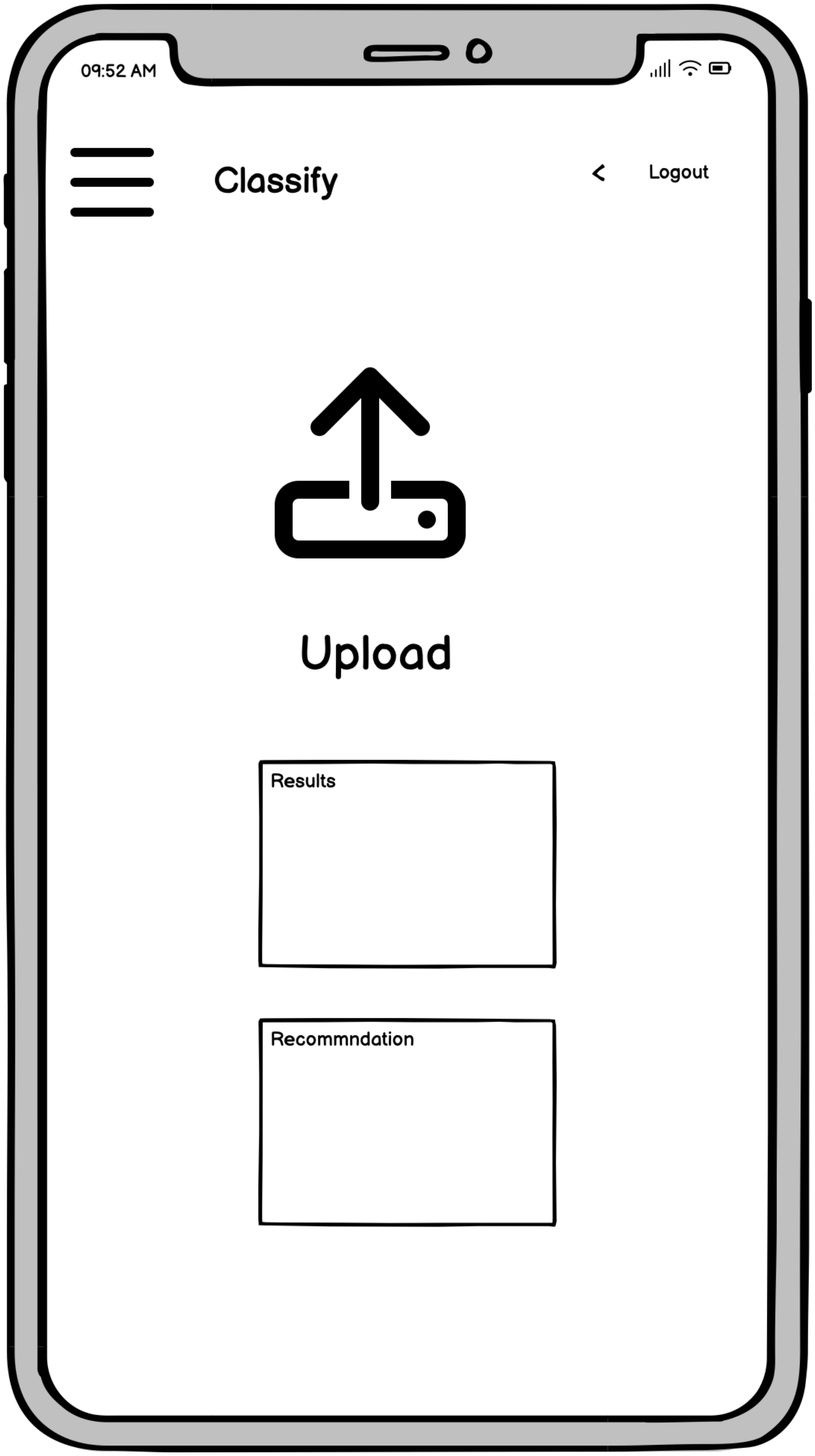



*Figure 29: Wireframe-User-Detection-Mobile*

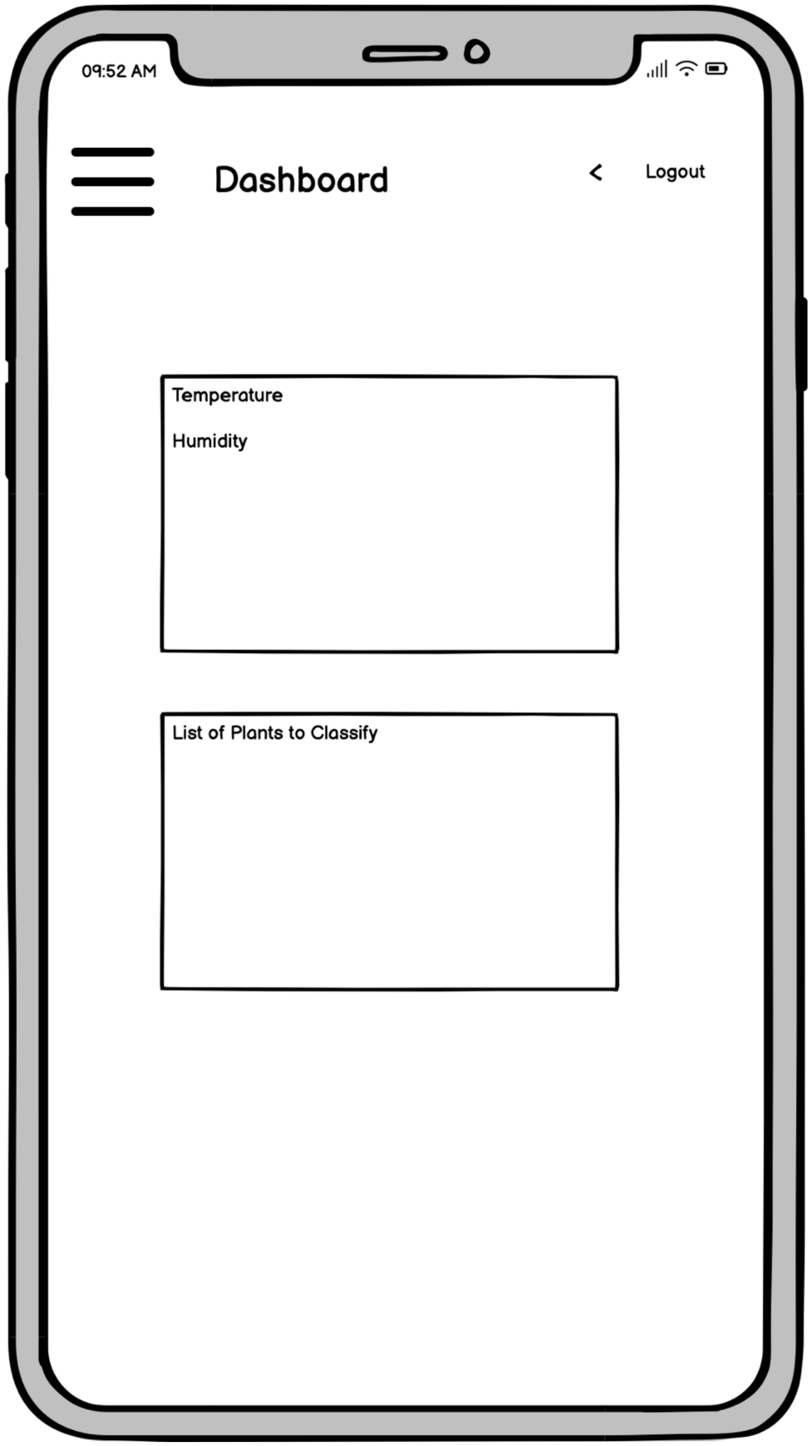



*Figure 30: Wireframe-User-Mobile-Dashboard*

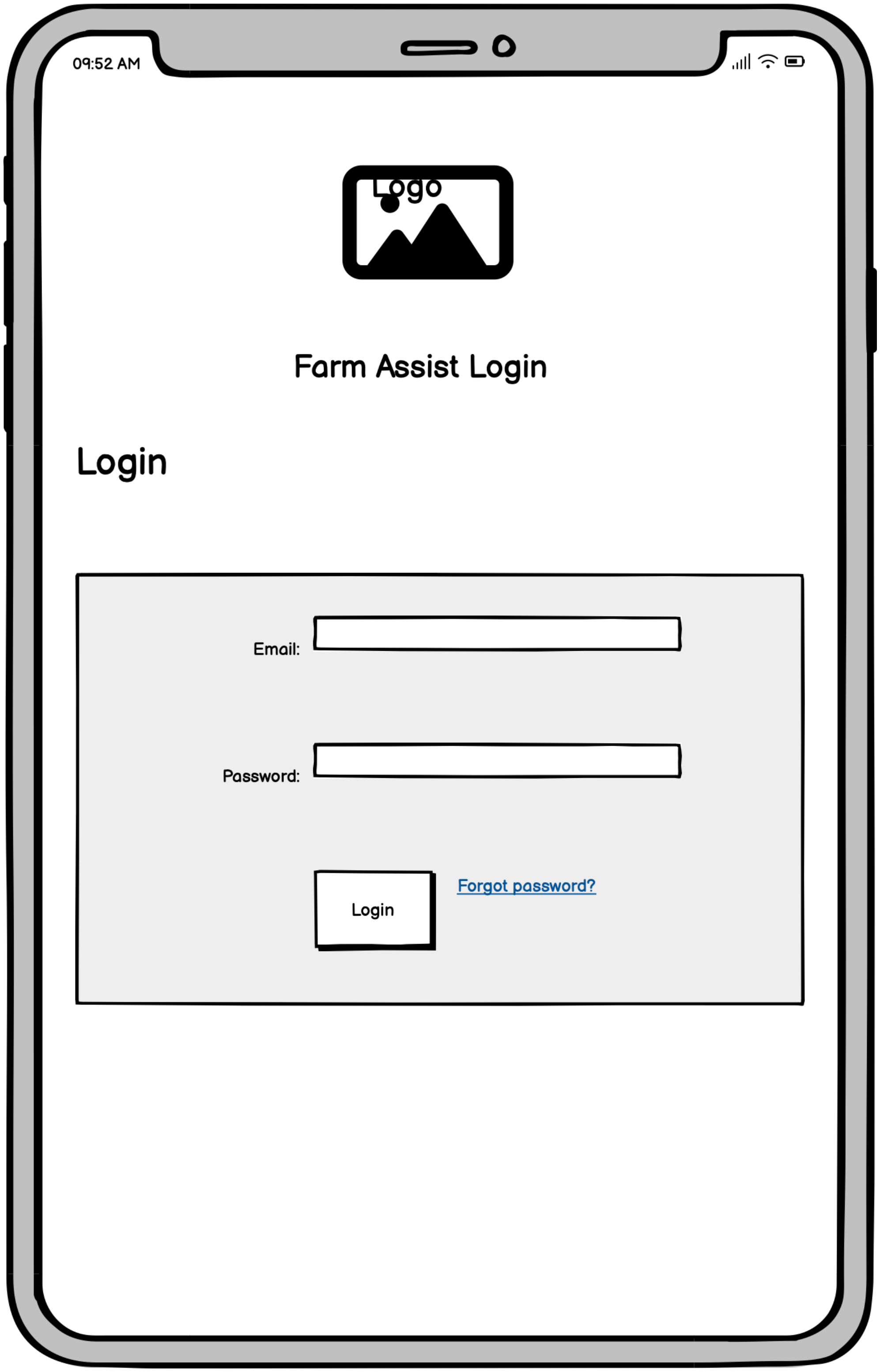



*Figure 31: Wireframe-User-Mobile-Login*


*Figure 32: Wireframe-User-Mobile-Register*

## 3.9 Project as a solution

The "Plant Disease Classification System" offers a modern and comprehensive solution to the pervasive issue of plant diseases in agriculture. Traditional methods of disease detection are often time-consuming and subjective, leading to substantial crop losses. To address this challenge, our project harnesses the capabilities of Convolutional Neural Networks (CNNs), proven effective in image classification tasks (Krizhevsky et al., 2012).

1. **Web Application (Django-based):** Users can conveniently upload images of diseased plants via a web application. The integrated CNN model promptly analyzes these images, providing accurate disease diagnoses. This tool benefits farmers and agricultural experts, enhancing disease identification speed and precision.

2. **Mobile Application (Flutter-based):** For on-the-go access, a mobile application developed with Flutter allows users to capture and upload images directly from their smartphones. This feature is particularly valuable for farmers in the field.

## 3.10 Developed Feature in Web

### 3.10.1 Prototype-1

### 3.10.2 Admin Dashboard

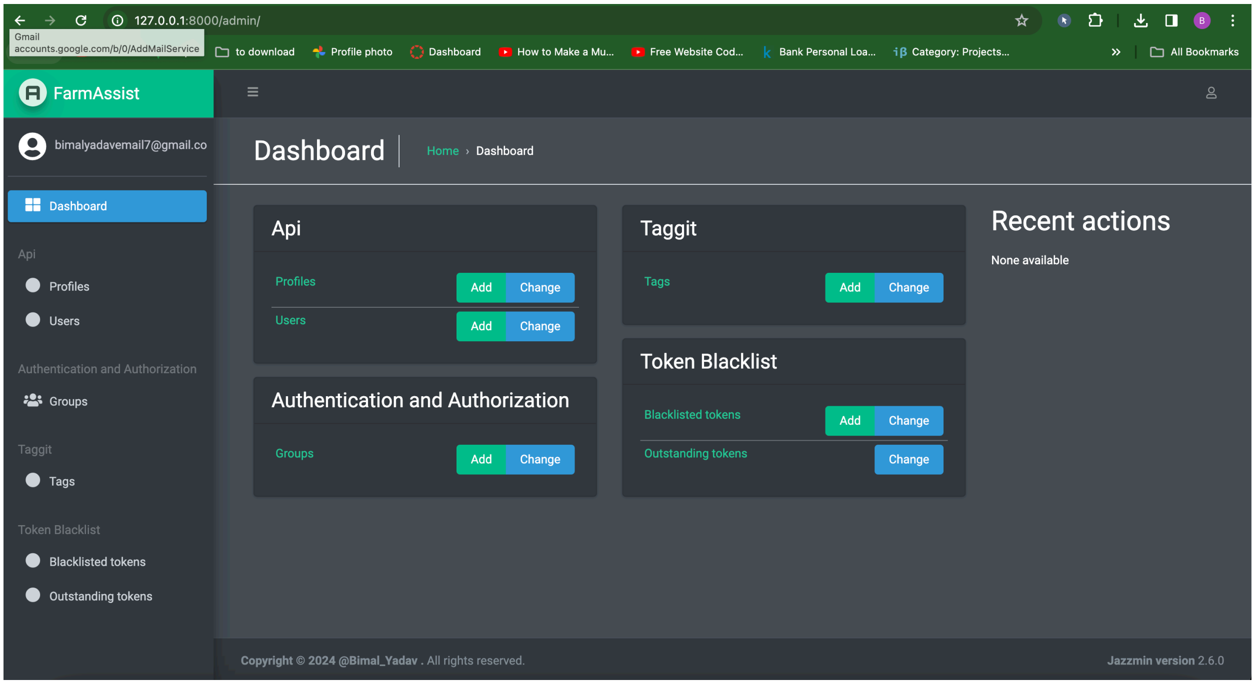



*Figure 33: Admin Dashboard*

### 3.10.3 User Profiles List

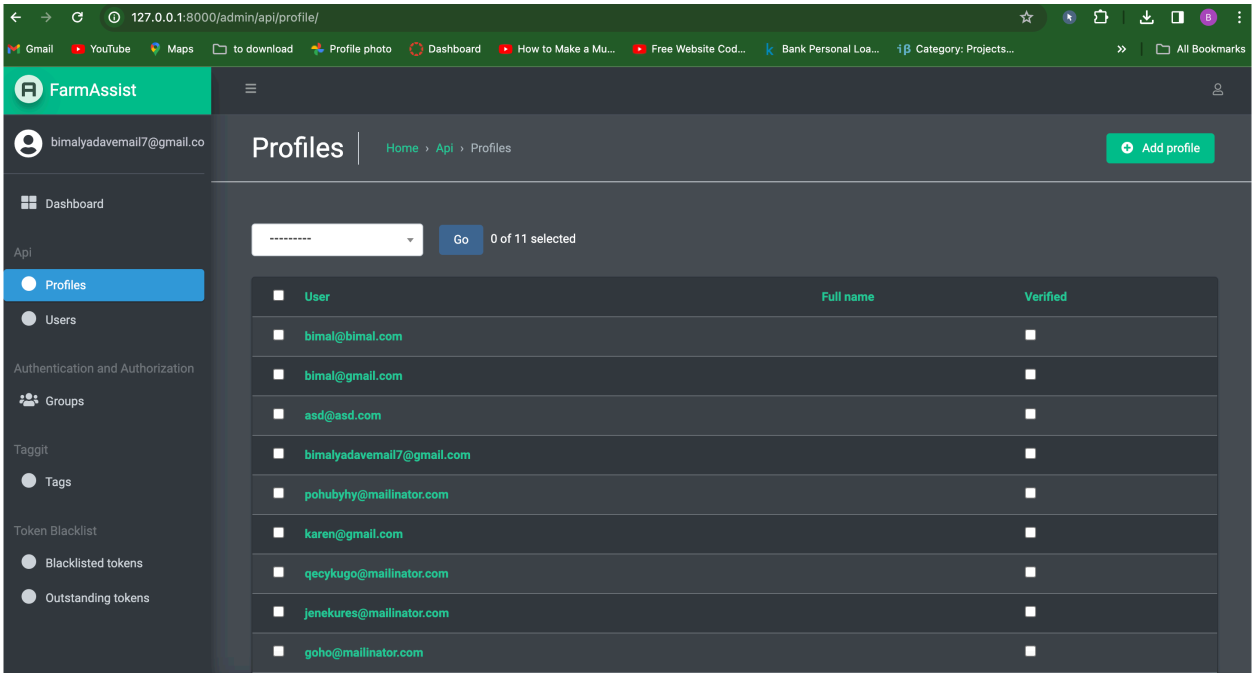



*Figure 34: User Profiles List*

### 3.10.4 Tokens

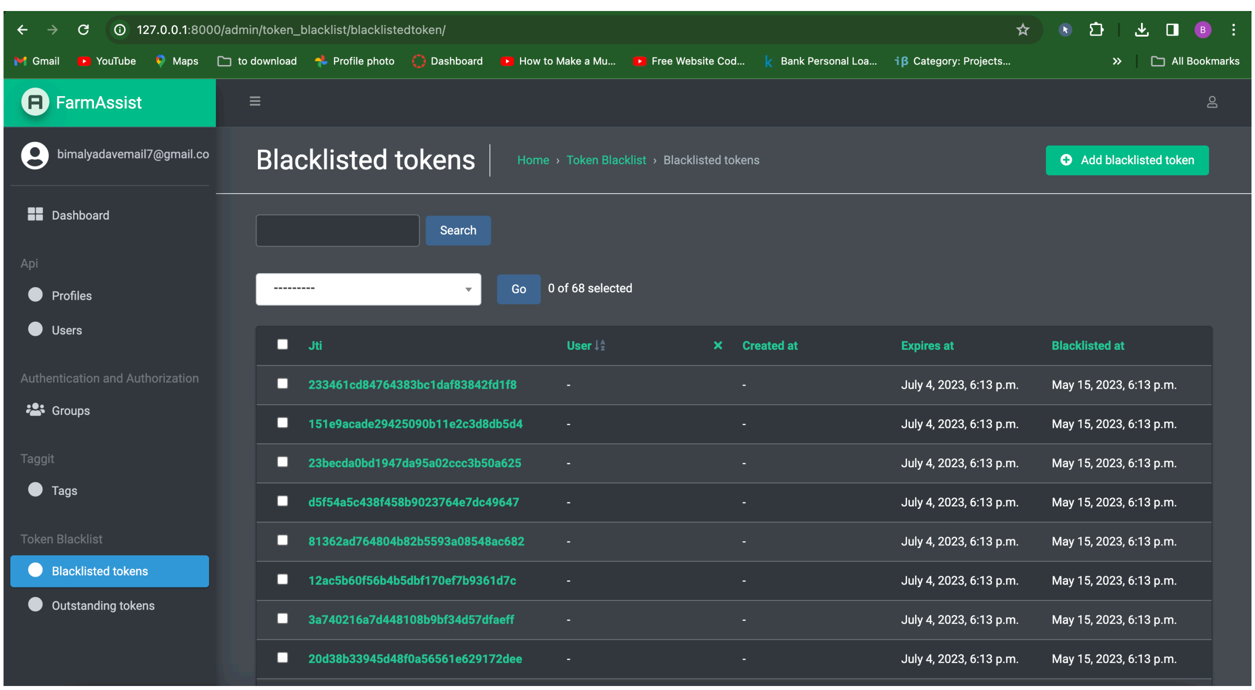



*Figure 35: Tokens*

### 3.10.5 Prototpye-2

### 3.10.6 Dashboard

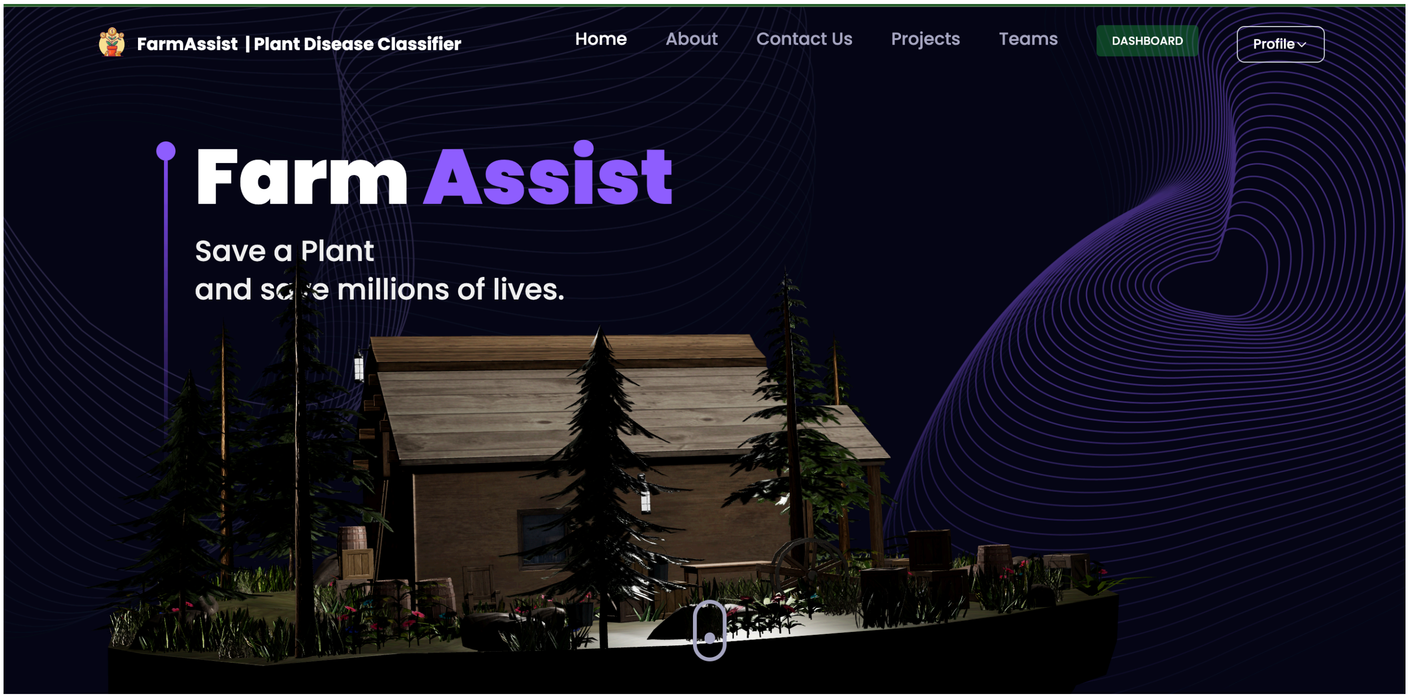



*Figure 36: Web Dashboard*

### 3.10.7 Contact Us Page

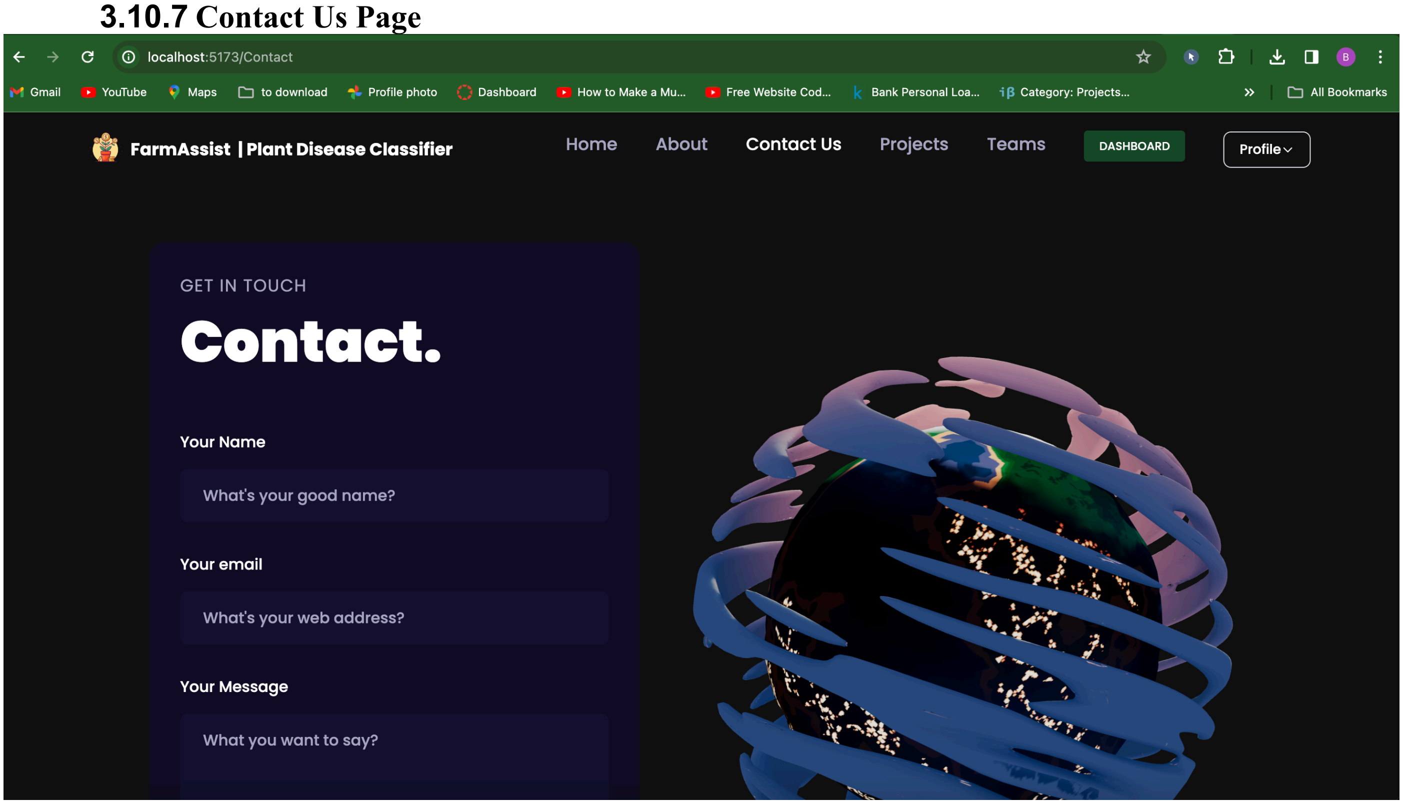



*Figure 37: Contact us Page*

### 3.10.8 Profile Button

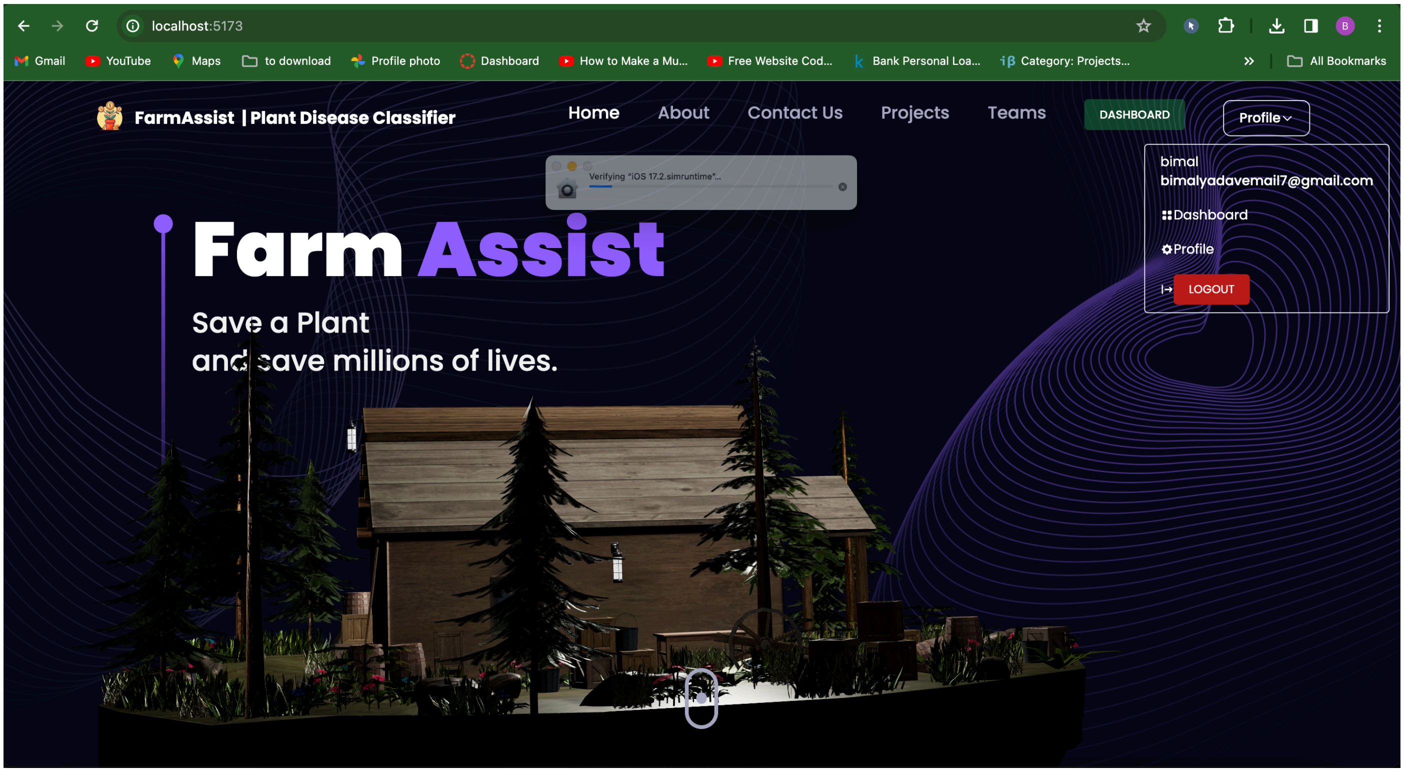



*Figure 38: Profile Button*

### 3.10.9 Login UI

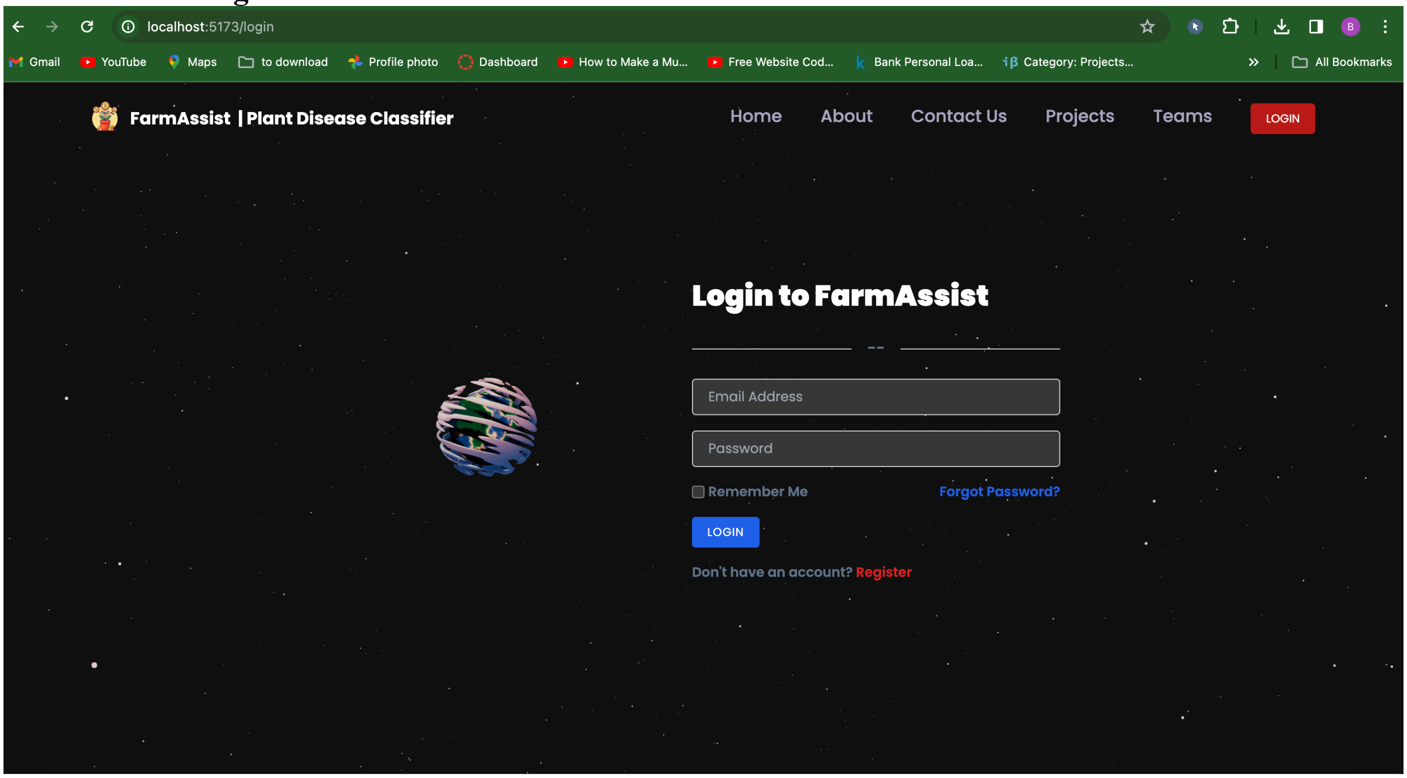



*Figure 39: Login UI*

### 3.10.10 Register UI

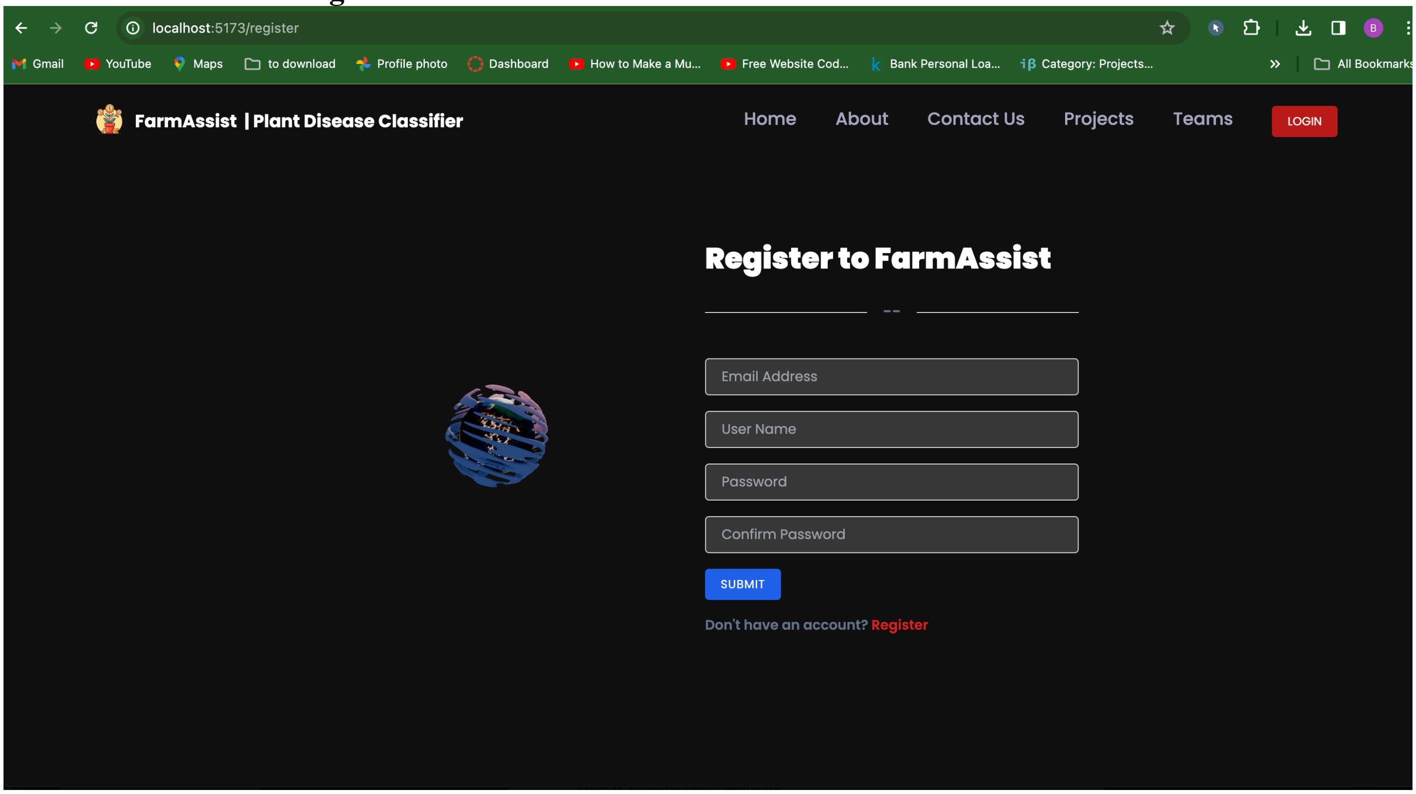



*Figure 40: Register UIs*

# 4 Analysis of Progress

Analysis of progress sections includes the progress of final year project and how it is developing? This section helps to determine the actual progress of project by comparing with Gantt Chart.

## 4.1 Progress Table

| | Tasks | Status | Progress (%) |
|---|---|---|---|
| **1.** | Topic Selection | Completed | **100%** |
| **2.** | Feasibility Study | Completed | **100%** |
| **3.** | Scheduling Resources | Completed | **100%** |
| **4.** | Estimation | Completed | **100%** |
| **5.** | Research on similar projects | Completed | **100%** |
| **6.** | Finalize Proposal | Completed | **100%** |
| **7.** | Conduct Public Survey | Completed | **100%** |
| **9.** | Design Logo for the project | Completed | **100%** |
| **10.** | Develop Overall System Use Case Diagram | Completed | **100%** |
| **12.** | Develop SRS Document | Completed | **100%** |
| **13.** | Develop High Level Use Case Diagram | Completed | **100%** |
| **14.** | Develop Wireframes | Completed | **100%** |
| **15.** | Finalized Initial ERD | Completed | **100%** |
| **16.** | Develop System Architecture Diagram | Completed | **100%** |
| **17.** | Develop Admin Panel in Web Application | Partially Completed | **10%** |
| **18.** | Develop WebUI | Partially Completed | **10%** |

| | | | |
|---|---|---|---|
| **18.a** | Develop AndroidUI | Not Completed | **0%** |
| **19.** | Finalize Interim Report | Completed | **100%** |
| **20.** | Develop System Operation Manual | Not Completed | **0%** |
| **21.** | Prepared Test Cases and Testing | Not Completed | **0%** |
| **22.** | Backend Deployment | Not Completed | **0%** |
| **23.** | Finalize FYP Report | Not Completed | **0%** |
| **24.** | Review and Refinement | Not Completed | **0%** |
| **25.** | **Submit Final Report to RTE** | **Not Completed** | **0%** |

## 4.2 Progress Review

The "Plant Disease Classification System" project has made significant strides since its initiation, demonstrating substantial progress across various facets. This section provides a comprehensive overview of the achievements, challenges encountered, and the roadmap for the upcoming phases.

**More about this in the Appendix: Progress Review**

## 4.3 Progress Timeline

The works of the project has been carried out on time according to the Gantt Chart which was submitted in the proposal.

## 4.4 Action Plan

A new Gantt Chart is deemed necessary because the initial Gantt Chart, which was included in the proposal, contained inaccuracies in the assigned dates. To address this, a revised Gantt Chart will be created to serve as a more accurate and up-to-date project timeline. The tasks delineated in the project plan will be executed in accordance with the timelines outlined in this revised Gantt Chart, ensuring proper synchronization.

The revised Gantt Chart will play a crucial role in guiding the progression of various project activities. It encompasses a range of tasks, including reporting, development, testing, and project refinement. Notably, these tasks will be conducted concurrently, allowing for a streamlined and efficient workflow.

The simultaneous execution of tasks is designed to incorporate feedback received from project supervisors in real-time. This iterative approach aims to address any potential issues promptly and align the project's trajectory with the intended goals. Overall, the revised Gantt Chart serves as a dynamic tool to maintain project momentum and adherence to timelines, contributing to the successful and timely completion of the project.

## 4.5 Work Breakdown Structure

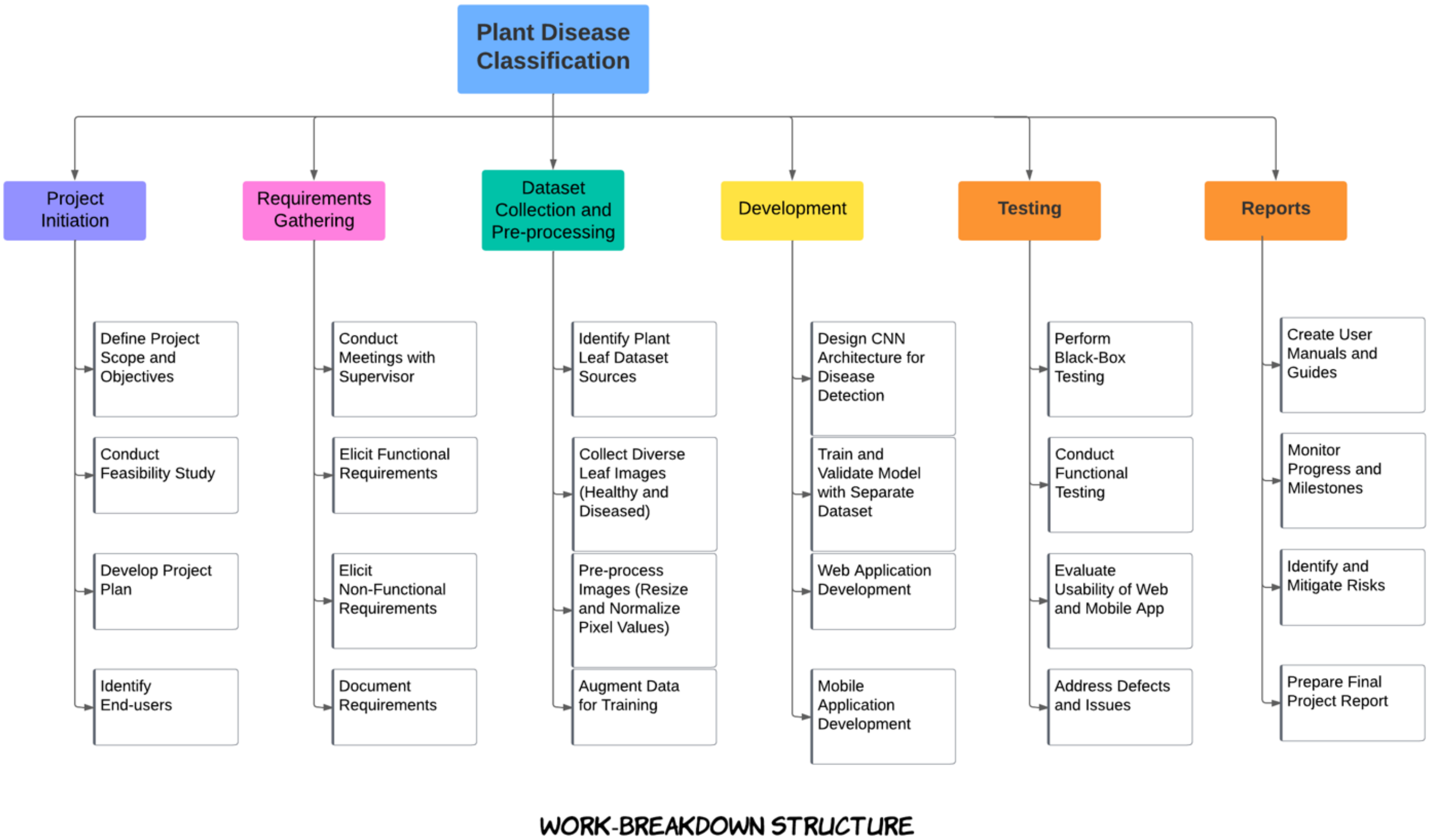



*Figure 41: Work Breakdwown Structure*

## 4.6 Project Gantt chart

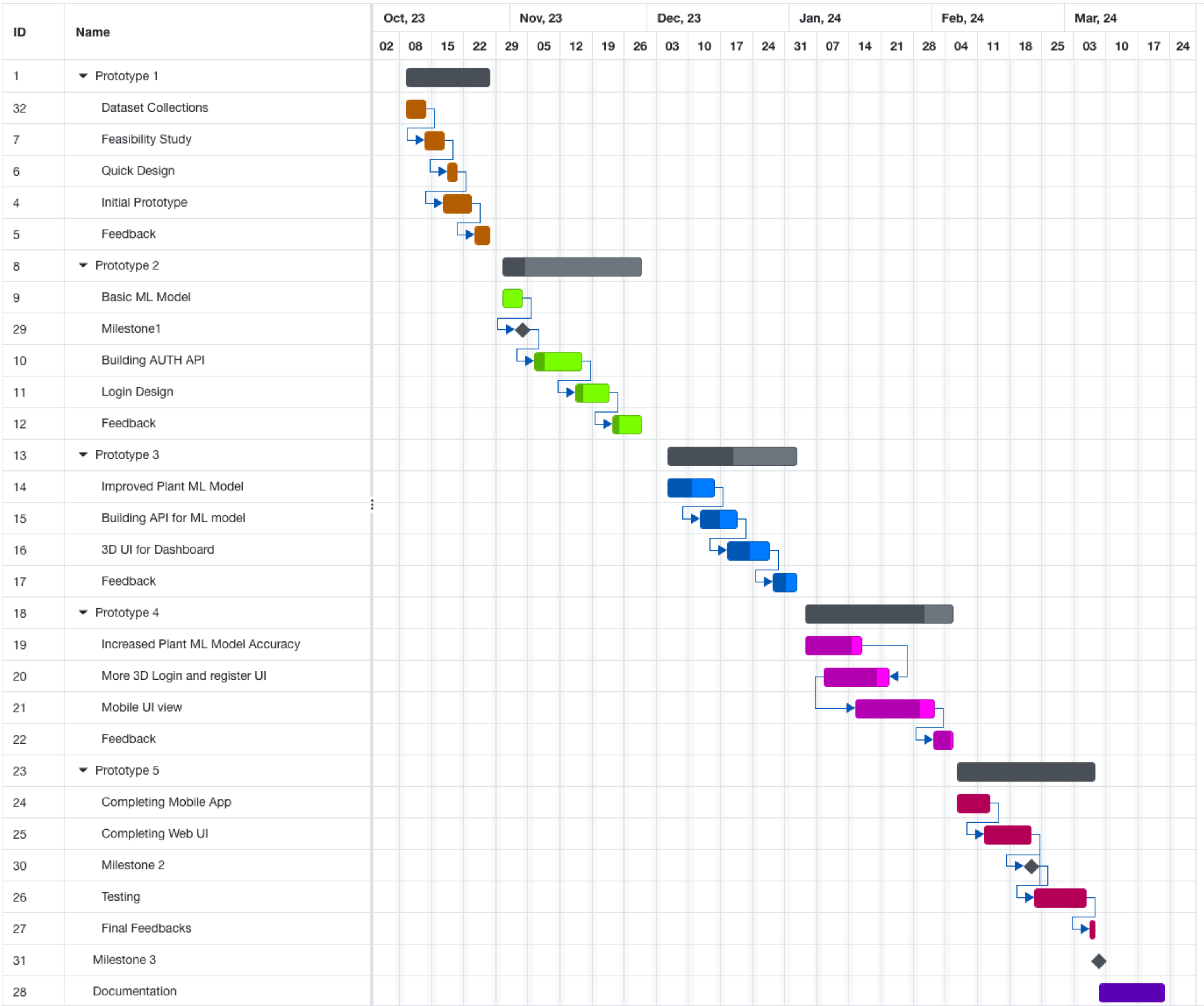
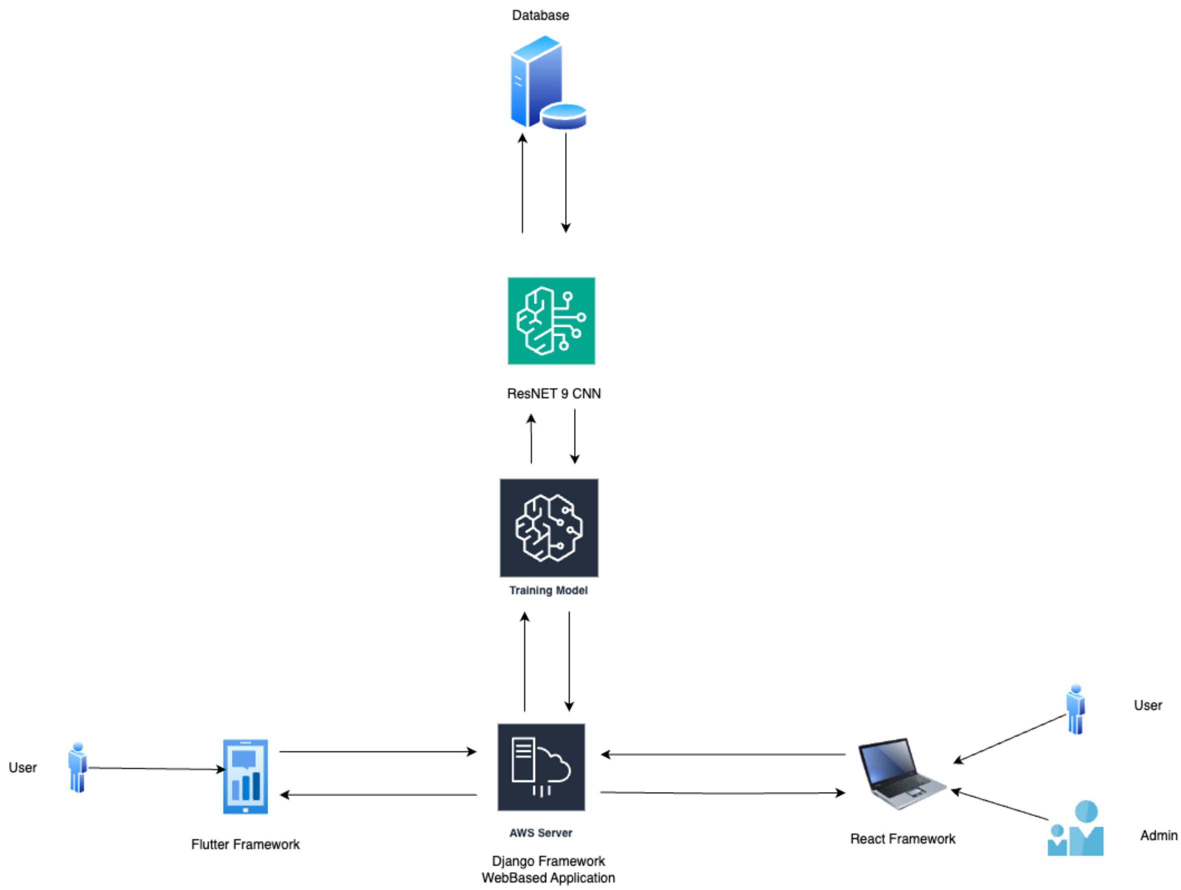



*Figure 42: Project Gantt chart*

## 4.7 System Architecture




**System Architecture**

## 4.8 Diagrams (Expanded, Collaboration , Sequences

### 4.8.1 Actors of FarmAssist

1. **Web User:** Represents users interacting with the system through the web application.
2. **Mobile User:** Represents users interacting with the system through the mobile application

| **Actor** | **Description** |
|---|---|
| Web User | Users interacting via the web application. |
| Mobile User | Users interacting via the mobile application. |

*Table 6: High level use case description of Actors*

### 4.8.2 Register

| **Name** | **Register People** |
|---|---|
| **Actors** | User (People) |
| **Description** | General public will enroll in this application by filling enrollment form. So that enrolled people can use the features of FarmAssist |

*Table 7: High level use case description of Register*

#### 4.8.2.1 Collobration diagram Register

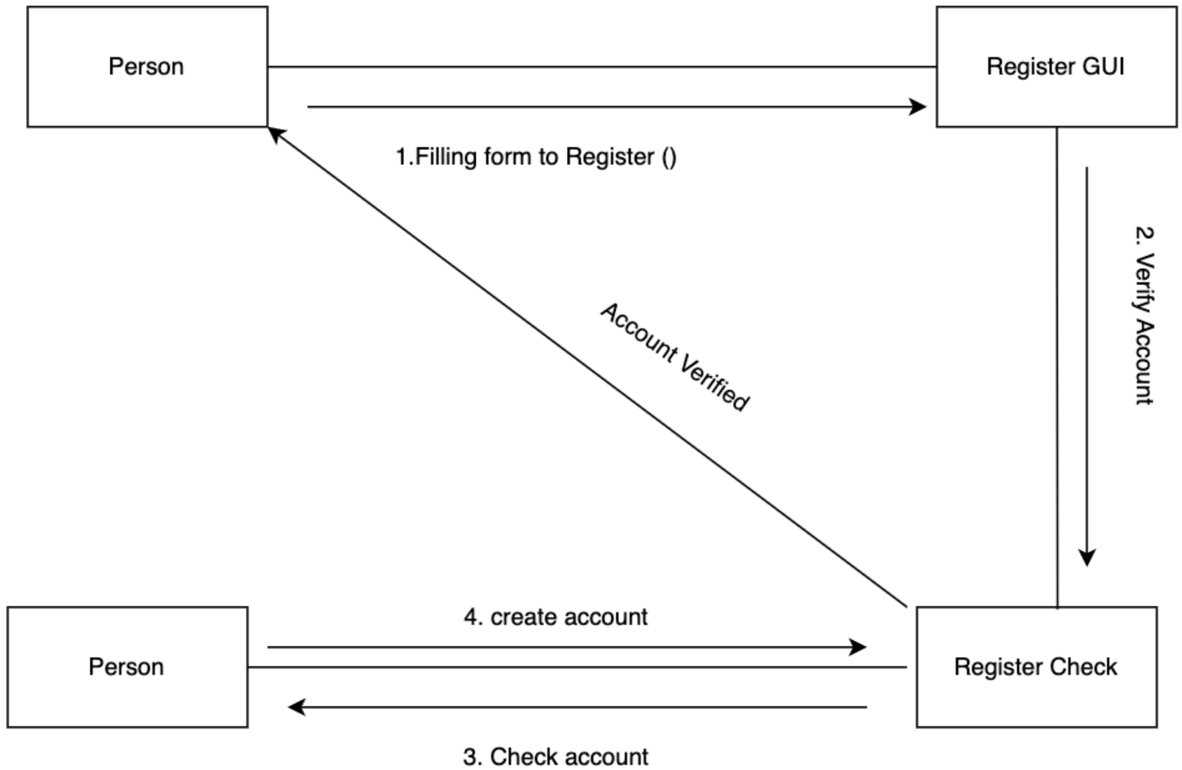



*Figure 43: Collobration diagram Register*

#### 4.8.2.2 Sequence digram of Register

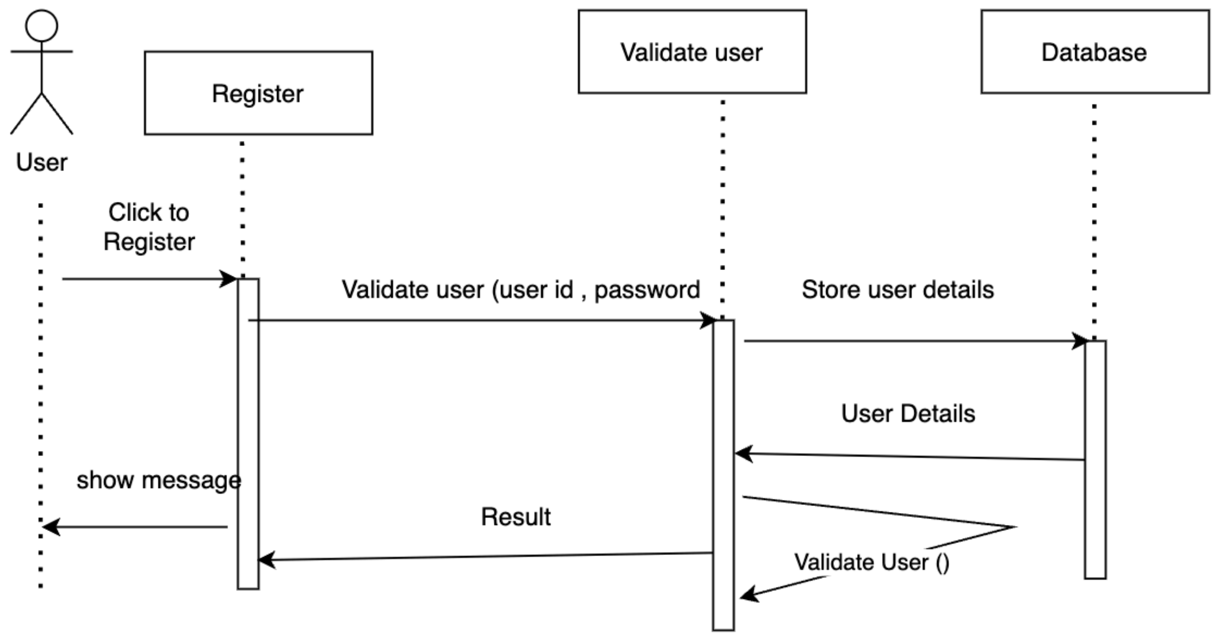



*Figure 44: Sequence digram of Register*

### 4.8.3 Login

**Expanded use case of Login**

| **Name** | **Login** |
|---|---|
| **Actors** | User (People) |
| **Description** | Registered users will log in to the application using their credentials to access personalized features. |

*Table 8: High level use case description of Login*

#### 4.8.3.1 collaboration diagram of Login

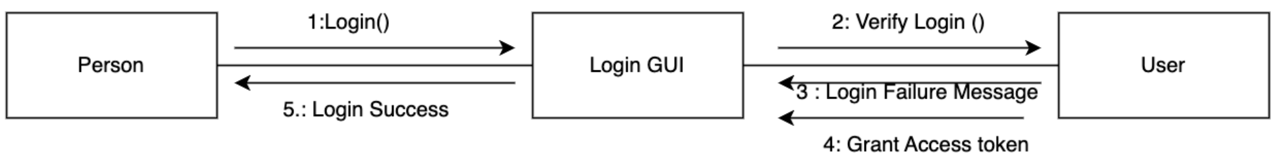



*Figure 45: Collaboration diagram of Login*

#### 4.8.3.1 Sequence digram of Login

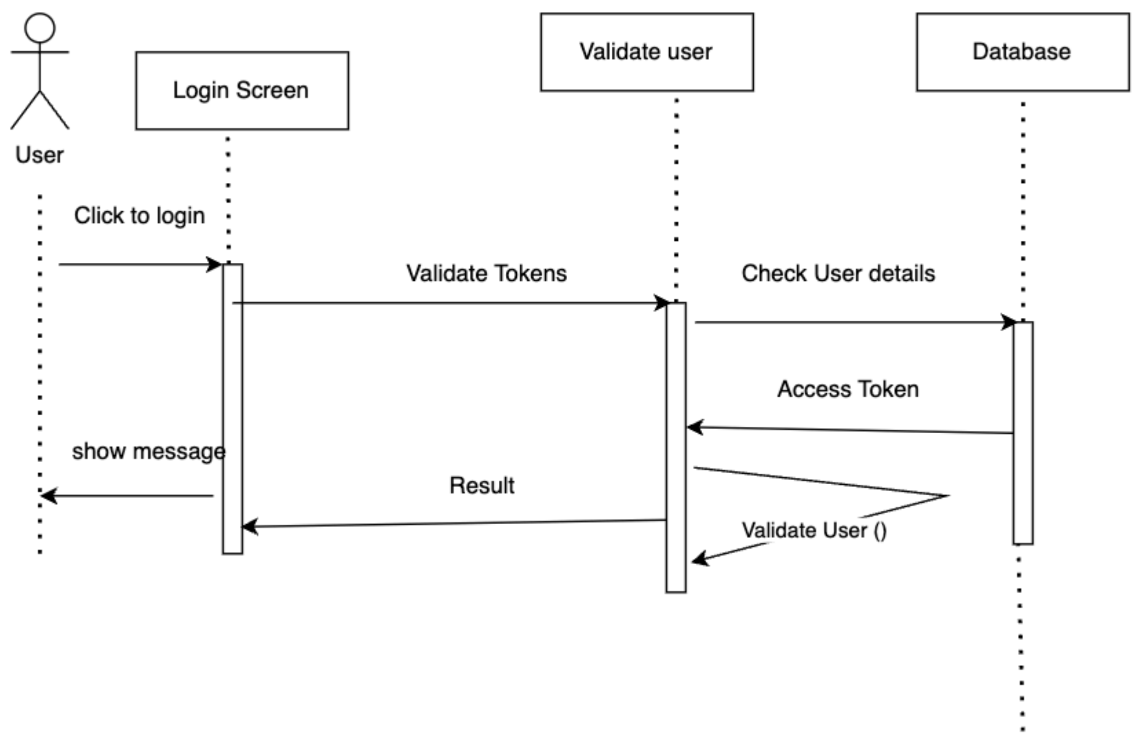



*Figure 46: Sequence diagram of Login*

### 4.8.4 View Profile

| Name | View Profile |
|---|---|
| **Actors** | User (People) |
| **Description** | Enrolled users can view their personalized profiles containing information and settings related to their account. |

*Table 9: High level use case description of View profile*

#### 4.8.4.1 Collobration diagram of View Profile

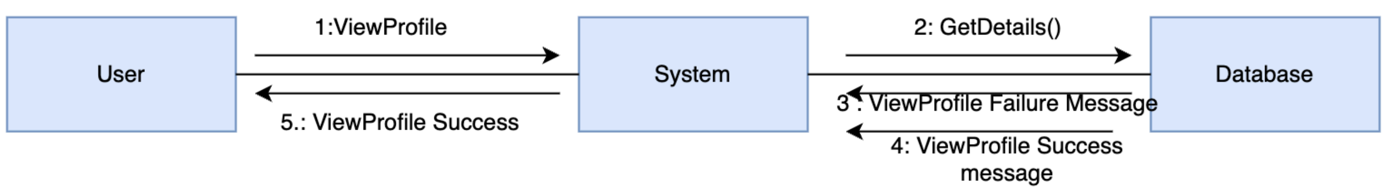



*Figure 47: collaboration diagram of View Profile*

#### 4.8.4.1 Sequence digram of View Profile

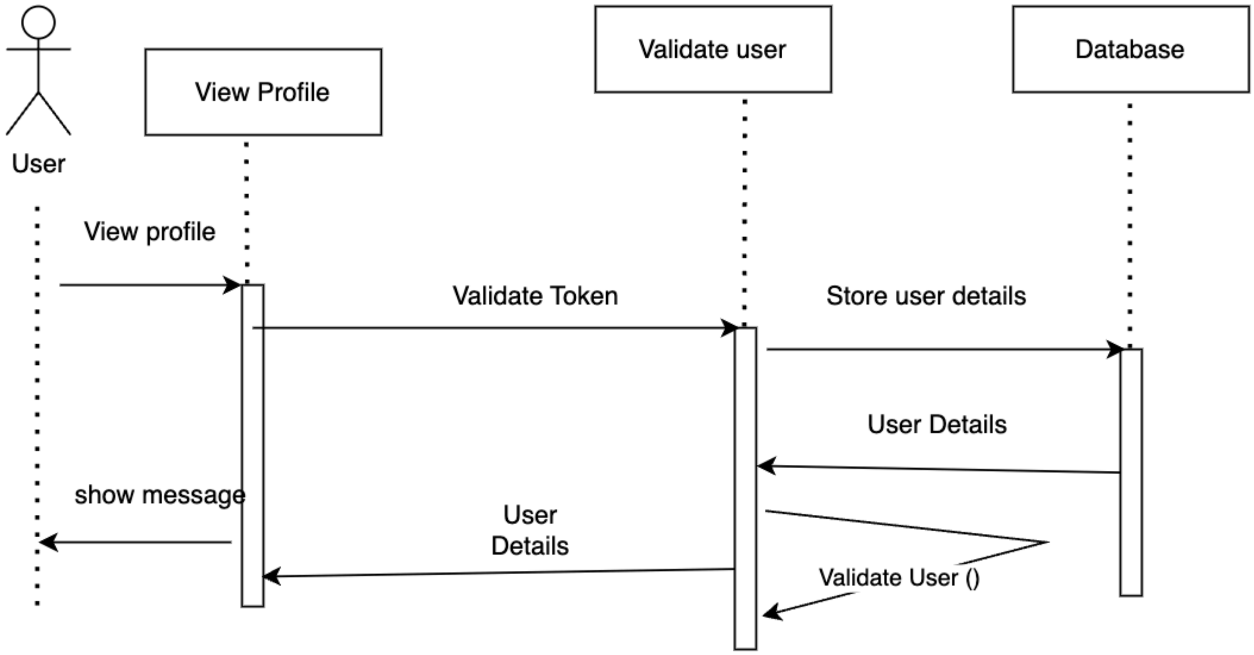



Figure 48: Sequence diagram of View Profile

### 4.8.5 Plant Detection Result

| **Name** | **Plant Detection Result** |
|---|---|
| **Actors** | User (Farmers, Agricultural Enthusiasts) |
| **Description** | Users can view the results of the plant disease classification, including information about detected diseases and relevant details regarding severity or recommended treatments. |

Table 10: High level use case description of Result

#### 4.8.5.1 Sequence digram of Plant Detection Result

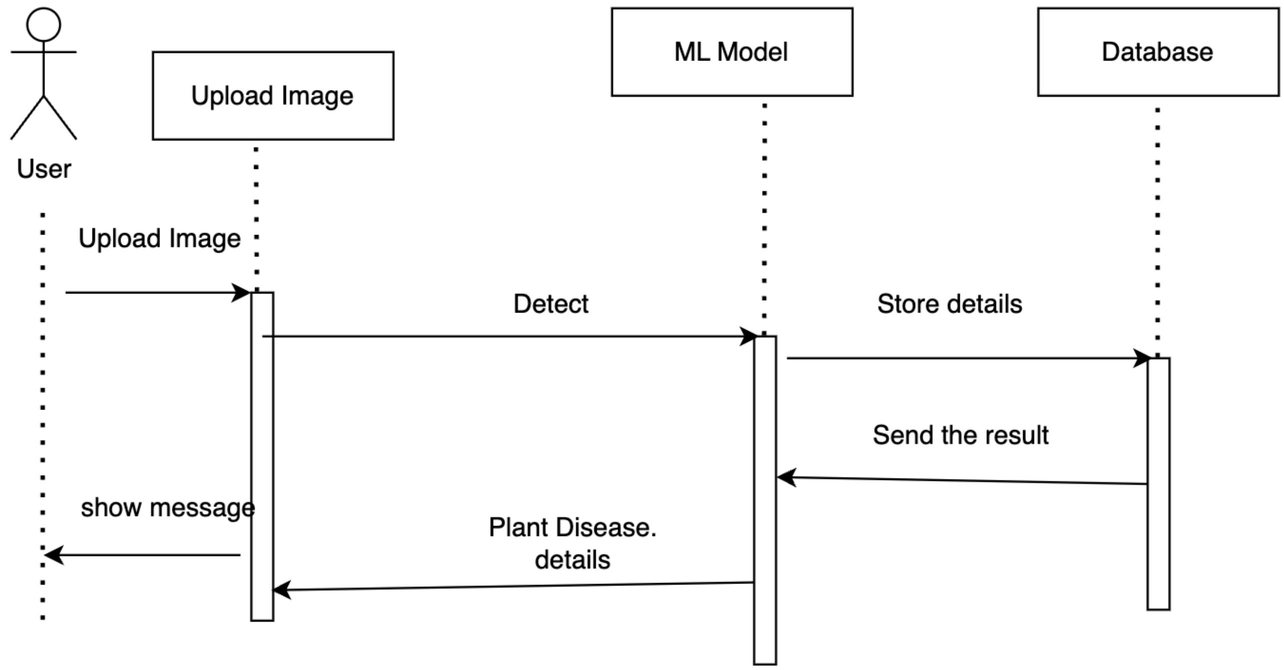



*Figure 49: Sequence diagram of Plant Detection Result*

### 4.8.6 Image Preprocessing

| | |
|---|---|
| **Name** | **Image Preprocessing** |
| **Actors** | System (Automated Process) |
| **Description** | The system processes uploaded plant images for disease detection using preprocessing techniques. This includes removing background noise, normalizing intensity levels, eliminating reflections, and masking irrelevant image sections. |
| **Input** | Plant images uploaded by users. |
| **Output** | Preprocessed images ready for disease detection. |

*Table 11: High level use case description of Image Processing*

### 4.8.6.1 Collobration diagram Image Processing

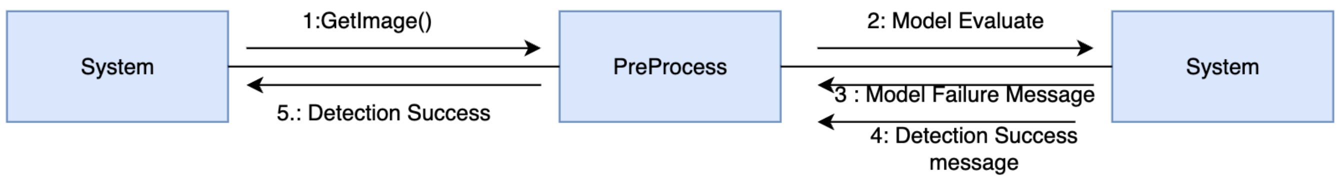



*Figure 50: collaboration diagram Image Processing*

### 4.8.6.1 Sequence digram of Image Processing

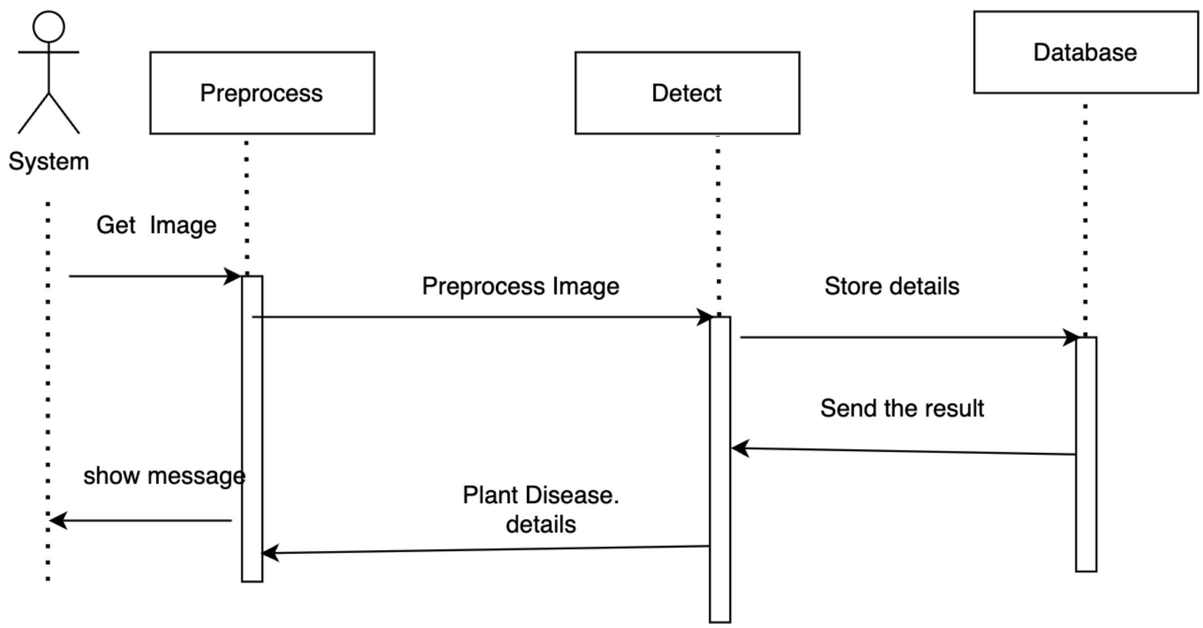



*Figure 51: Sequence diagram of Image Processing*

### 4.8.7 System Maintenance

| **Name** | **System Maintenance** |
|---|---|
| **Actors** | System Admin (Maintenance Personnel) |
| **Description** | Routine maintenance tasks performed on the system to ensure its optimal performance and reliability. This includes updates, backups, and resolving technical issues. |
| **Frequency** | Regularly scheduled maintenance activities. |
| **Tasks** | - Software updates and patches |
| | - Database backups |
| | - Monitoring system health and performance |
| | - Addressing reported issues and bugs |

*Table 12: High level use case description of System Maintenance*

#### 4.8.7.1 Collobration diagram System Maintenance

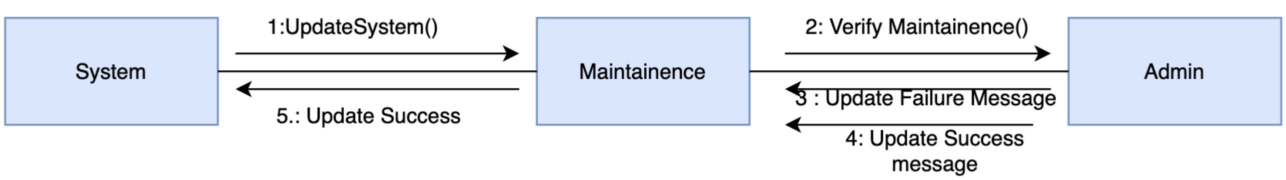



*Figure 52: collaboration diagram System Maintenance*

#### 4.8.7.1 Sequence digram of System Maintenance

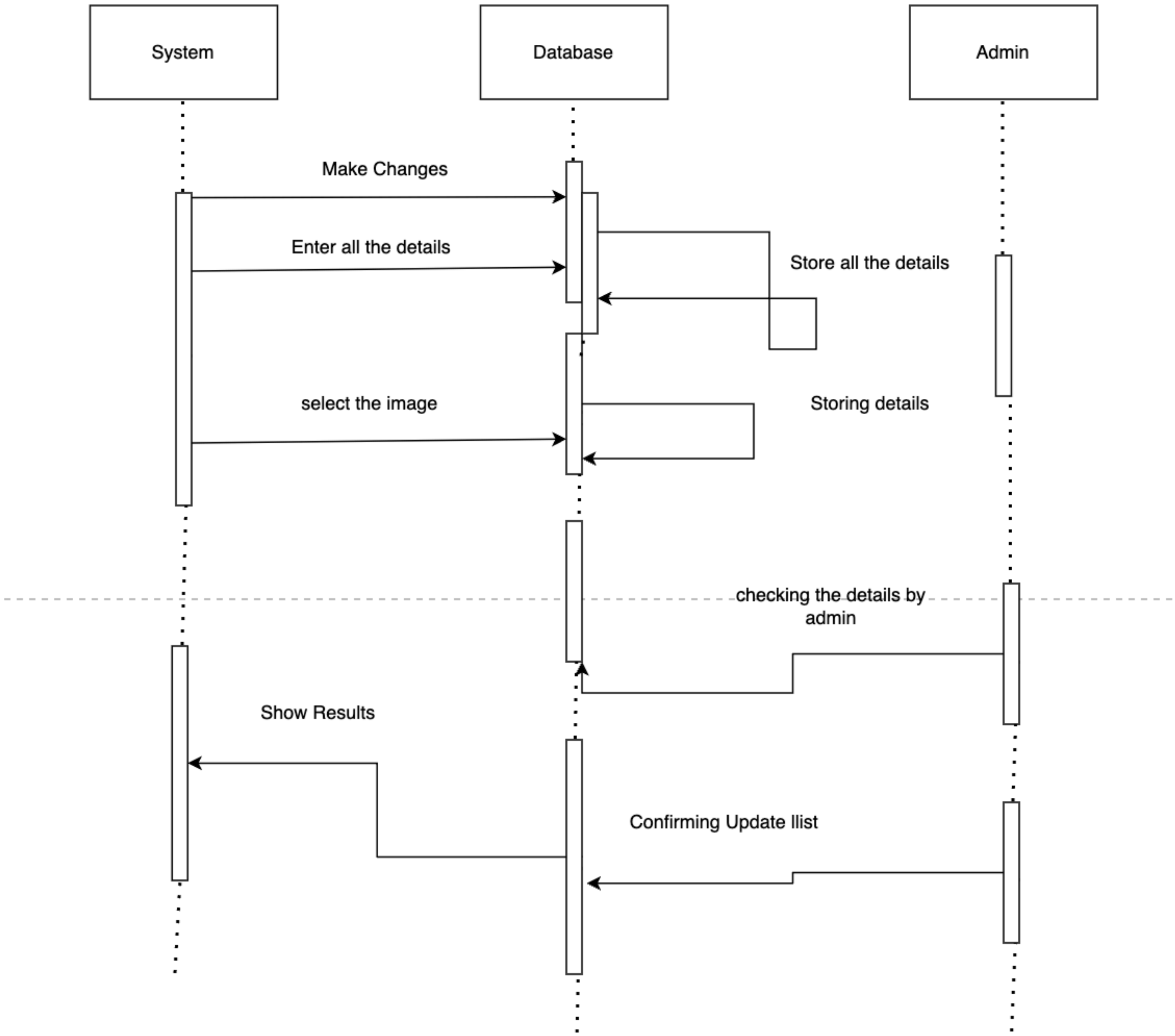



*Figure 53: Sequence diagram of System Maintenance*

### 4.8.8 Admin Login

| Name | Admin Login |
|---|---|
| Actors | Administrator (Admin) |
| Description | The admin accesses the system by providing valid credentials (username and password) to perform administrative tasks and manage system settings. |
| Preconditions | The admin account must be created and active. |
| Postconditions | Successful login redirects the admin to the admin dashboard. |
| Exceptions | - Invalid credentials result in an error message. |
| Main Flow | 1. Admin enters username and password. |
| | 2. System verifies credentials. |
| | 3. If valid, the admin is logged in; otherwise, an error message is displayed. |
| Extensions | - Admin can reset the password if forgotten. |

*Table 13: High level use case description of Admin Login*

#### 4.8.8.1 Collobration diagram of Admin Logins

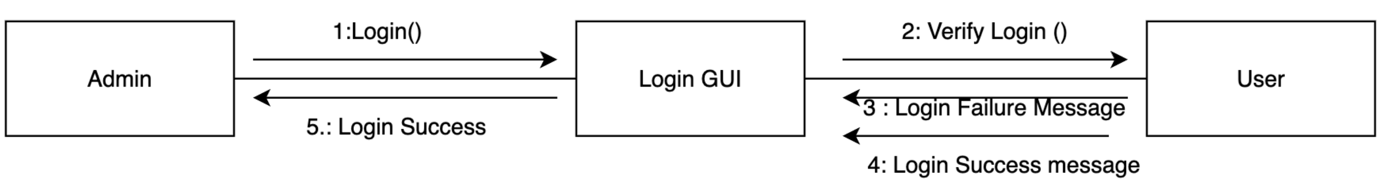



*Figure 54: collaboration diagram of Admin Logins*

#### 4.8.8.1 Sequence digram of Admin Login

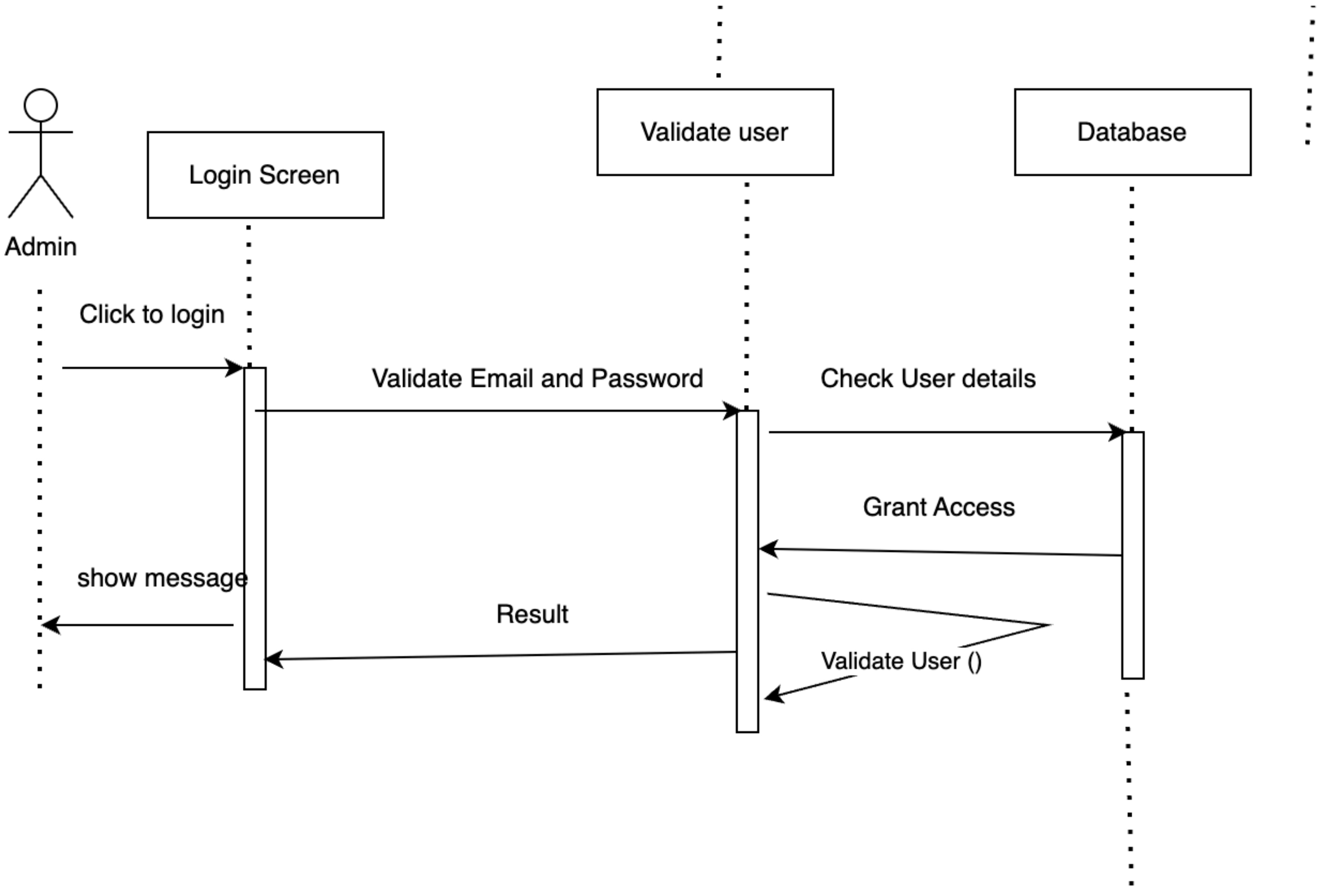
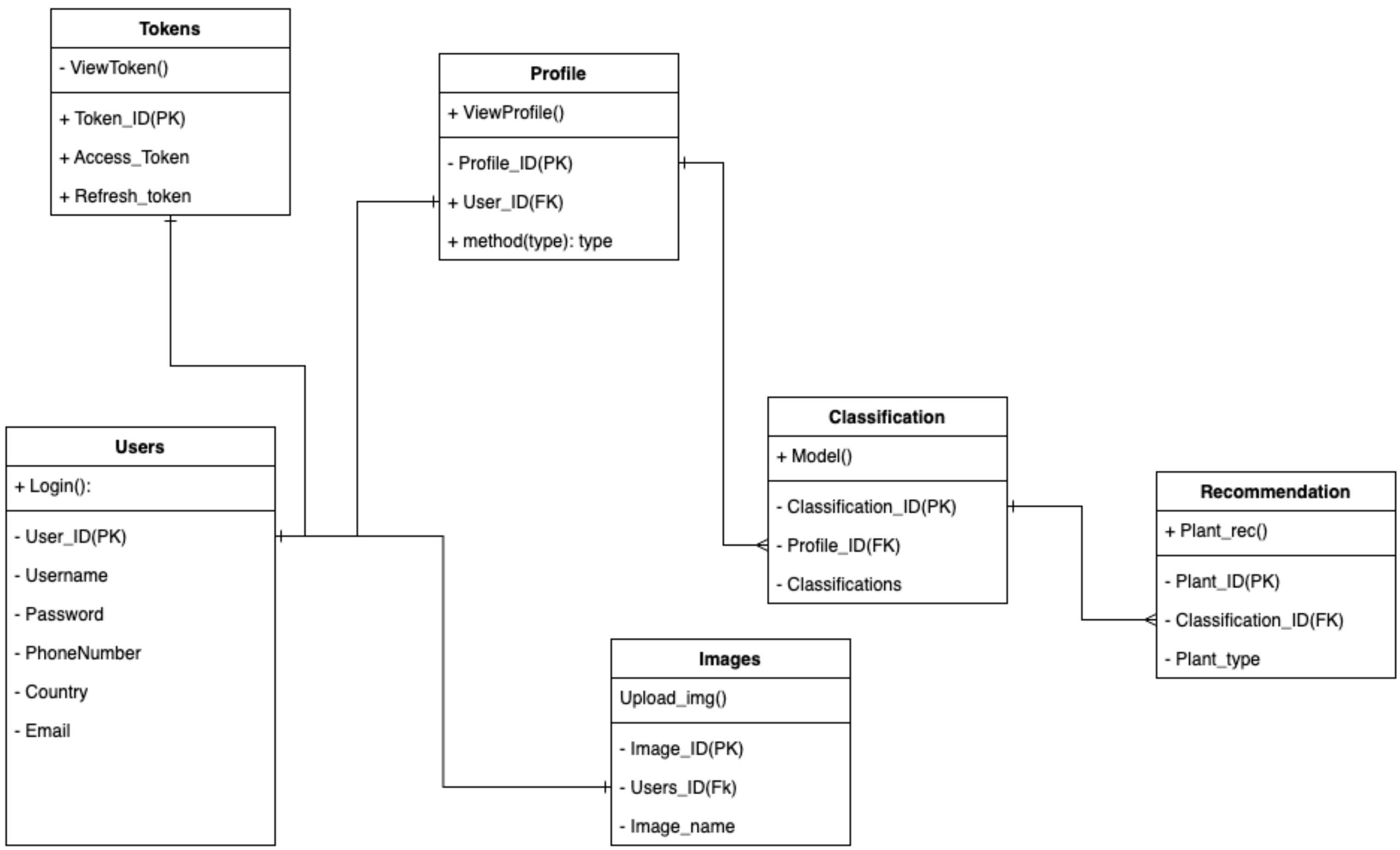



*Figure 55: Sequence diagram of Admin Login*

## 4.9 Class Diagram




**Class Diagram**

### 4.10 Entity Relationship Diagram

More about this in the Appendix: **Entity Relationship Diagram**

### 4.11 Data Dictionary

More about this in the Appendix: **Data Dictionary**

## 5 Future Work

The upcoming tasks in the project are delineated as future work, with their execution planned within the estimated time frame outlined in the Gantt Chart. The outstanding tasks include:

a) **Development of Admin Panel in the Web Application:**

- The admin panel will empower administrators to Users, Tokens, Profiles and database.

b) **Creation of Android Applications for Users:**

- For customers, the Android application will fascinate the disease detection more precisely by taking the images directly from the phone.

c) **Formulation of System Operational Manual:**

- The System Operational Manual will comprehensively guide users on processes such as how to detect and list of plants to be added.

d) **Development of Test Cases and Testing:**

- Rigorous test cases will be devised for systematic testing to identify any potential errors in the developed system. Detected errors or bugs will be addressed by developers to ensure a flaw-free system.

e) **Finalization of Final Year Project (FYP) Report:**

- The finalization of the FYP report will align with the Gantt Chart. Components such as the Use Case diagram, Entity Relationship Diagram (ERD), System Architecture diagram, sequence diagram, and test cases will undergo finalization during this phase.

f) **Review and Refinement of Project Report:**

- Based on feedback from supervisors, necessary adjustments will be made to refine the FYP report and ensure its adherence to well-structured documentation standards.

g) **Submission of Final Year Project Report to RTE:**

- Within the stipulated deadline, the Final Year Project report will be submitted in the RTE classroom. This submission will include the final documentation report in PDF format, along with folders like the Project folder and Development folder. The Project folder will contain evidence of the project, such as screenshots of Stack Overflow, journal articles, etc. The Development folder will encompass the code development for both the Android and web applications.

In Summary Table of Future Works:

| Tasks | Description |
|---|---|
| Improved Model(More Accuracy) | Model is currently lagging in the performance |
| Complete Frontend | One of the main duties is to finish the remaining portion of the frontend before the project is submitted in its final form. |
| Complete Backend | The backend is still not structured in a REST architecture. Will be fixed in next Ptototype |
| Overall Testing | After the project's frontend and backend have been completed and integrated, the system still has to be tested overall. It aids in ensuring system functionality and locating any potential flaws. |
| Final Deployment | The project's ultimate objective is its final deployment. It must be made available for use by everyone on the planet. |

## 5.1 Revised milestone

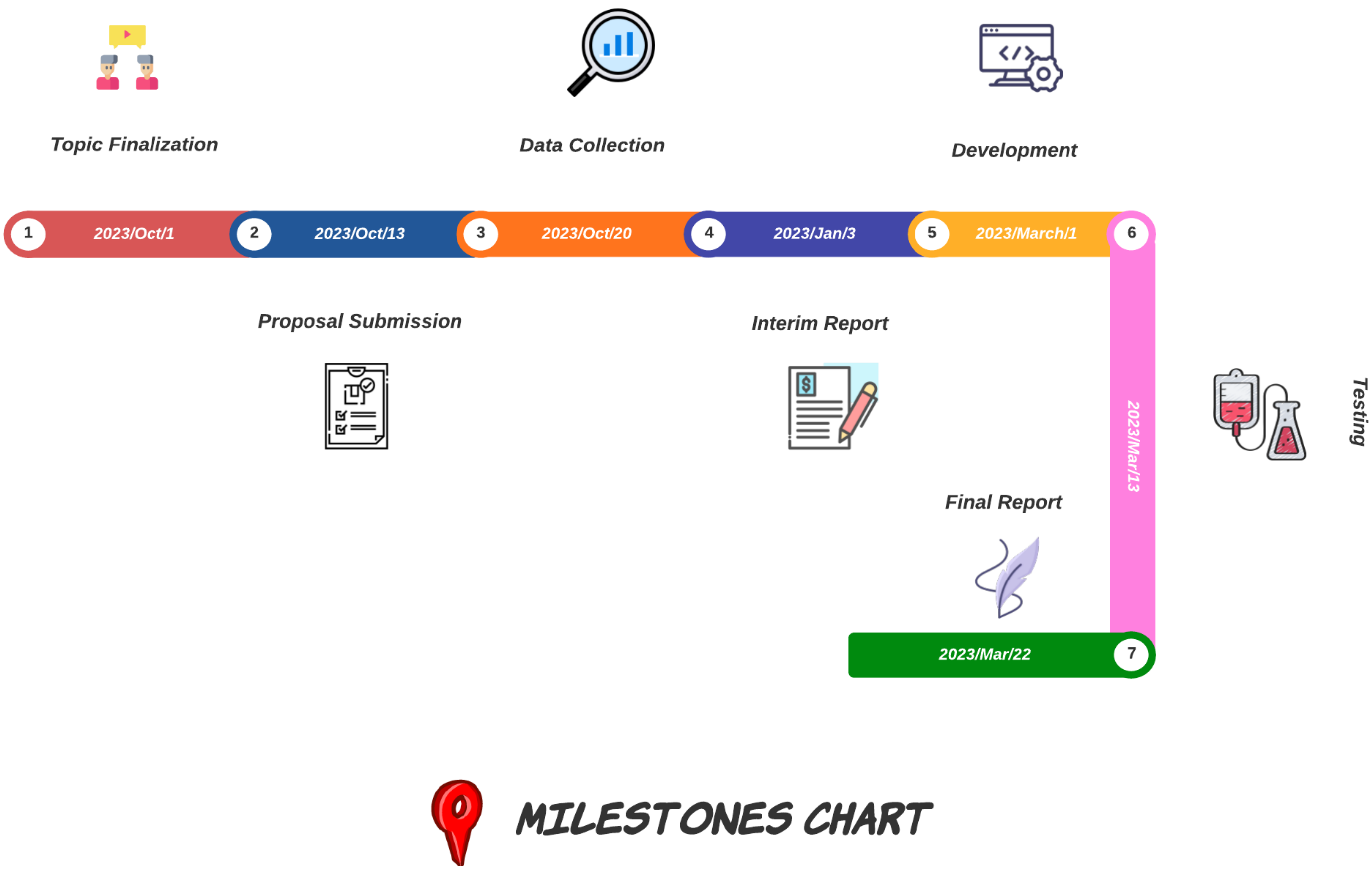



*Figure 56: Revised milestone*

## 5.2 Revised Gantt Chart

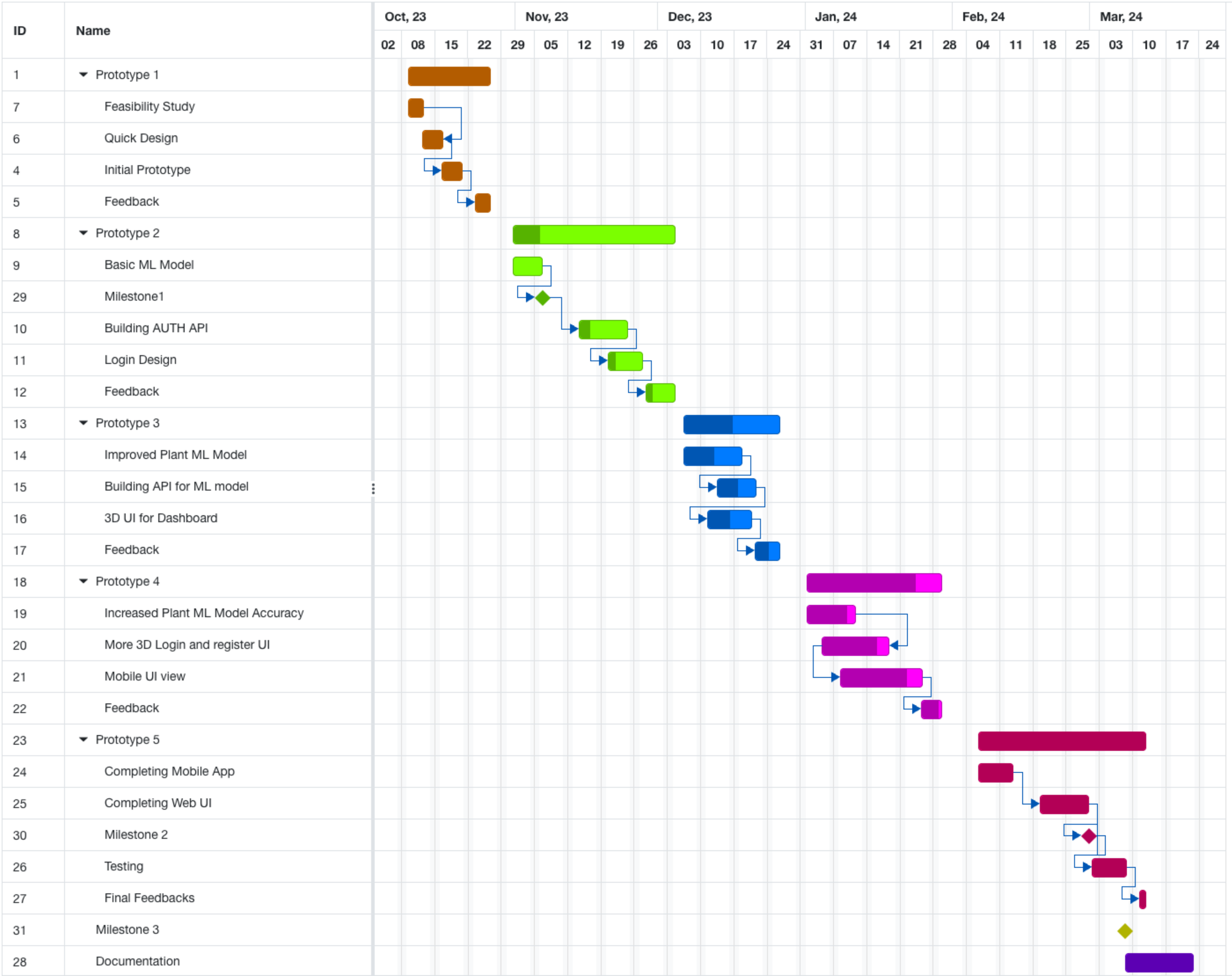



*Figure 57: Revised Gantt Chart*

# 6 References

Barbedo, J.G.A. (2019) 'Plant disease identification from individual lesions and spots using deep learning.' *Biosystems Engineering, 180*, pp. 96-107.

Mondal, D., Chakraborty, A., Kole, D.K. and Majumder, D.D. (2015) 'Detection and classification technique of yellow vein mosaic virus disease in okra leaf images using leaf vein extraction and Naive Bayesian classifier.' In: *2015 International Conference on Soft Computing Techniques and Implementations (ICSCTI)*. IEEE, pp. 166-171.

Padol, P.B. and Yadav, A.A. (2016) 'SVM classifier based grape leaf disease detection.' In: *2016 Conference on advances in signal processing (CASP)*. IEEE, pp. 175-179.

Reza, Z.N., Nuzhat, F., Mahsa, N.A. and Ali, M.H. (2016) 'Detecting jute plant disease using image processing and machine learning.' In: *2016 3rd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT)*. IEEE, pp. 1-6.

Tejoindhi, M.R. and Nanjesh, B.R. (2016) 'Plant Disease Analysis Using Histogram Matching Based On Bhattacharya's Distance Calculation.' In: *International Conference on Electrical, Electronics and Optimization Techniques (ICEEOT)-2016*.

Arya, M.S., Anjali, K. and Unni, D. (2018) 'Detection of unhealthy plant leaves using image processing and genetic algorithm with Arduino.' In: *2018 International Conference on Power, Signals, Control and Computation (EPSCICON)*. IEEE, pp. 1-5.

Mehra, T., Kumar, V. and Gupta, P. (2016) 'Maturity and disease detection in tomato using computer vision.' In: *2016 Fourth International Conference on Parallel, Distributed and Grid Computing (PDGC)*. IEEE, pp. 399-403.

Singh, R., Singh, R. K., & Srivastava, S. (2016) 'SVM based leaf disease detection using feature selection and fusion.' *Procedia Computer Science, 84*, pp. 103-110. doi: 10.1016/j.procs.2016.04.066

Kaur, S., Kaur, A., & Singh, R. (2019) 'SVM-based detection of tomato leaf diseases: A comparative study.' *Computers and Electronics in Agriculture, 156*, pp. 32-41. doi: 10.1016/j.compag.2018.12.016

Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016) 'Using deep learning for image-based plant disease detection.' *Frontiers in Plant Science, 7*, 1419. doi: 10.3389/fpls.2016.01419

Ferentinos, K. P. (2018) 'Deep learning models for plant disease detection and diagnosis.' *Computers and Electronics in Agriculture, 145*, pp. 311-318. doi: 10.1016/j.compag.2017.11.008

Shakya, S., Sharma, S., & Gautam, S. (2019) 'Plant disease detection using digital image processing and deep learning techniques: A review.' *Archives of Computational Methods in Engineering, 26(5)*, pp. 1301-1317. doi: 10.1007/s11831-018-9293-9

Singh, R., Dhillon, M. S., & Singh, J. (2016) 'Sensor-based disease detection in plants: A review.' *Journal of Innovative Optical Health Sciences, 9(4)*, 1630003. doi: 10.1142/S1793545816300036

Shakoor, N., Lee, H., & Mockler, T. C. (2017) 'High-throughput phenotyping of plant height using unmanned aerial vehicle and its application to genomic prediction analysis in barley.' *The Plant Genome, 10(2)*, pp. 1-10. doi: 10.3835/plantgenome2016.09.0093

Kamilaris, A., & Prenafeta-Boldú, F. X. (2018) 'Deep learning in agriculture: A survey.' *Computers and Electronics in Agriculture, 147*, pp. 70-90.

Zhang, Y., Zhang, S., Zhao, F., Jin, X., & Xie, X. (2020) 'Plant disease recognition via a multi-scale attention convolutional neural network.' *Sensors, 20(9)*, 2692.

Singh, A., Ganapathysubramanian, B., & Singh, A. K. (2016) 'Machine learning for high-throughput stress phenotyping in plants.' *Trends in Plant Science, 21(2)*, pp. 110-124.

Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016) 'Using deep learning for image-based plant disease detection.' *Frontiers in Plant Science, 7*, 1419.

Liakos, K. G., Busato, P., Moshou, D., Pearson, S., & Bochtis, D. (2018) 'Machine learning in agriculture: A review.' *Sensors, 18(8)*, 2674.

Das, M., Dey, N., Shrestha, G., & Deepsikha. (2020, October 9) 'Plant Disease Detection Using CNN.' In: *2020 IEEE Applied Signal Processing Conference (ASPCON)*, pp. 109-113.

Militante, V. S., Gerardo, D. B., & Dionisio, V. N. (2019) 'Plant Leaf Detection and Disease Recognition using Deep Learning.' In: *2019 IEEE Eurasia conference on IOT, communication and engineering (ECICE)*, pp. 579-582.

Mishra, S., Sachan, R., & Rajpal, D. (2020) 'Deep convolutional neural network-based detection system for real-time corn plant disease recognition.' *Procedia Computer Science*, pp. 2003-2010.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). "ImageNet Classification with Deep Convolutional Neural Networks." *Advances in Neural Information Processing Systems. Retrieved from arXiv.*

Savary, S., et al. (2019). "The global burden of pathogens and pests on major food crops." *Nature Ecology & Evolution*, 3(3), 430-439.

Food and Agriculture Organization of the United Nations (FAO). (2019). "The State of Food and Agriculture 2019. Moving forward on food loss and waste reduction." Retrieved from FAO Website.

United Nations. (2019). "World Population Prospects 2019." Retrieved from United Nations Website.

Madden, L. V., Hughes, G., & van den Bosch, F. (2007). *The Study of Plant Disease Epidemics. American Phytopathological Society.*

Agrios, G. N. (2005). *Plant Pathology* (5th ed.). Academic Press.

Hsiang, T. (2018). *Deep Learning for Image-Based Plant Disease Detection*. CRC Press.

Singh, A., & Ganapathysubramanian, B. (2017). *"A Robust Deep-Learning-Based Detector for Real-Time Tomato Plant Diseases."* Computers and Electronics in Agriculture, 138, 200-209.

Sladojevic, S., Arsenovic, M., & Anderla, A. (2016). *"Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification."* Computers in Industry, 87, 12-20.

Mahlein, A.-K. (2016). *"Plant Disease Detection by Imaging Sensors – Parallels and Specific Demands for Precision Agriculture and Plant Phenotyping."* Plant Disease, 100(2), 241-251.s

# 7 Appendix

## 7.1 Proposed System

**Farmers:**

Farmers represent a crucial demographic that stands to benefit significantly from the "Farm Assist - Plant Disease Classification" project. Agriculture forms the backbone of their livelihoods, and the project aims to provide them with a powerful tool for managing plant diseases effectively. Key aspects of farmers as end users include:

1. **Disease Detection and Management:** Farmers can use the classification system to identify diseases affecting their crops promptly. Early detection allows for timely intervention, reducing the risk of widespread crop losses.

2. **User-Friendly Interface:** The web and mobile applications are designed with a user-friendly interface, ensuring that farmers with varying levels of technological proficiency can easily navigate and utilize the system.

3. **On-the-Go Access:** The Flutter-powered mobile application facilitates on-the-go access, enabling farmers to capture and upload images directly from their smartphones while in the field.

4. **Crop Management Support:** The comprehensive plant disease database integrated into the system provides valuable information for disease management, offering insights into various diseases and their treatments.

**Agricultural Enthusiasts:**

Beyond traditional farmers, agricultural enthusiasts, which may include researchers, students, or individuals passionate about agriculture, constitute another segment of end users. Their involvement in the project brings additional benefits:

1. **Research and Knowledge Sharing:** The open-source nature of the project encourages collaboration and knowledge sharing within the agricultural community. Researchers and students can explore the classification model, contributing to advancements in plant disease detection.

2. **Educational Resources:** The system provides educational resources, including guides and references, enhancing the knowledge base of agricultural enthusiasts interested in plant pathology and disease management.

3. **Contribution to Sustainable Agriculture:** By empowering farmers with a tool for early disease detection, agricultural enthusiasts indirectly contribute to the promotion of sustainable agriculture practices, aligning with global efforts to ensure food security.

## 7.2 Aims and Objective Description

## 7.3 Aims

**Bridge the Gap Between Technology and Agriculture:** The primary aim of this project is to create a robust Plant Disease Classification model based on Convolutional Neural Networks (CNNs) that can accurately and swiftly identify plant diseases across a wide range of plant categories.

## 7.4 Objective

1. **Develop a CNN-Based Model:** Build and train a CNN-based model using a diverse dataset encompassing 38 plant categories for effective disease classification.
2. **Create a Web Application:** Develop a user-friendly web application using the Django framework, allowing users to upload plant images for instant disease diagnosis.
3. **Design a Mobile Application:** Build a mobile application using Flutter, enabling users to capture and upload plant images via smartphones for on-the-go disease diagnosis.
4. **Integrate Comprehensive Plant Disease Database:** Incorporate a comprehensive plant disease database to support the classification model and provide users with a reference for different plant diseases.
5. **Conduct Rigorous Testing and Validation:** Thoroughly test and validate the CNN model to ensure high accuracy in disease classification.
6. **Ensure User Accessibility:** Prioritize user-friendliness in both the web and mobile applications to cater to users with varying levels of technological proficiency.
7. **Provide Open-Source Resources:** Make the disease classification model and associated tools open-source, contributing to the research and agricultural communities.

## 7.5 Technology Used:

### 7.5.1 Django REST Framework

Django REST Framework (DRF) is a powerful and flexible toolkit for building Web APIs using Django, which is a high-level Python web framework. DRF simplifies the process of creating APIs, making it easier to build robust and scalable web applications.

**Key Features:**

1. **Serialization:** DRF provides a serialization mechanism to convert complex data types (such as Django models) into Python data types that can be easily rendered into JSON or other content types.
2. **Viewsets and Serializers:** DRF introduces the concept of viewsets and serializers. Viewsets define the view behavior, while serializers define how the data should be converted to a JSON format.
3. **Authentication and Permissions:** DRF includes built-in support for various authentication methods, such as Token Authentication and OAuth. It also provides a flexible permissions system to control access to views.
4. **Browsable API:** DRF includes a browsable API, which means that developers can interact with the API directly from the browser. It provides a user-friendly interface for exploring and testing API endpoints.

### 7.5.2 React

React is a JavaScript library for building user interfaces, particularly for single-page applications where the user interacts with the page dynamically. Developed by Facebook, React allows developers to build reusable UI components that update efficiently in response to data changes.

**Key Features:**

1. **Component-Based Architecture:** React follows a component-based architecture, where the UI is broken down into reusable components. Each component manages its own state, making it easier to develop and maintain complex applications.
2. **Virtual DOM:** React uses a virtual DOM to improve performance by updating only the parts of the actual DOM that have changed. This leads to faster rendering and a smoother user experience.

3. **JSX (JavaScript XML):** React uses JSX, a syntax extension for JavaScript that resembles XML/HTML. JSX allows developers to write HTML elements and components in a more concise and readable manner.
4. **Unidirectional Data Flow:** React follows a unidirectional data flow, meaning that data flows in a single direction. This makes it easier to understand how data changes over time and helps in debugging.

### 7.5.3 Flutter

Flutter is an open-source UI software development toolkit developed by Google. It is used for building natively compiled applications for mobile, web, and desktop from a single codebase. Flutter uses the Dart programming language.

**Key Features:**

1. **Hot Reload:** Flutter's hot reload feature allows developers to instantly see the result of the code changes without restarting the entire application. This significantly speeds up the development process.
2. **Widgets:** Everything in Flutter is a widget, including the app itself. Flutter has a rich set of customizable widgets that make it easy to create complex UIs.
3. **Single Codebase:** With Flutter, developers can write code once and deploy it on multiple platforms (iOS, Android, web, and desktop). This helps in maintaining a single codebase for different platforms.
4. **Expressive UI:** Flutter provides a rich set of pre-designed widgets and allows developers to create custom widgets. This results in expressive and flexible user interfaces.

Django REST Framework is used for building robust APIs with Django, React is a JavaScript library for building dynamic user interfaces, and Flutter is a UI toolkit for building natively compiled applications for various platforms. Each technology serves a specific purpose in the development stack, and their combination allows for efficient and scalable application development.

## 7.6 Review of Similar Project appendix

There are 6 (Six) Research Papers Reviewed to see how the other model is built and how they perform.

### 7.6.1 1. Authors: I Ketut Eddy Purnama

Department of Computer Engineering, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

*Disease Classification based on Dermoscopic Skin Images Using Convolutional Neural Network in Teledermatology System* (no date) *IEEE Conference Publication | IEEE Xplore*. Available at: https://ieeexplore.ieee.org/document/8973303.

In this research project, a novel system for the classification and detection of skin diseases is proposed, specifically tailored for application in Teledermatology. The primary objective is to leverage advanced technologies, particularly Deep Learning, with a focus on the Convolutional Neural Network (CNN) algorithm, to classify skin diseases based on dermoscopic images.

**Dataset and Data Source:** The study utilizes dermoscopic image data sourced from the MNIST HAM10000 dataset, comprising a substantial collection of 10,015 images. This dataset is notable for its publication by the International Skin Image Collaboration (ISIC). The images in the dataset are categorized into seven classes of skin diseases, all falling within the spectrum of skin cancer.

**Methodology:** The core methodology involves employing two pre-trained CNN models, namely MobileNet v1 and Inception V3, for the image classification process. These models undergo a learning process using the provided dataset. Subsequently, the acquired knowledge is applied to a web-classifier, facilitating the classification and detection of skin diseases in real-world scenarios.

**Results and Comparative Analysis:** The research report provides a detailed comparison of predictive accuracy between the two utilized CNN models. The web-classifier integrated with the CNN Inception V3 model demonstrates a notable accuracy value of 72%. In contrast, the web-classifier utilizing the MobileNet v1 model exhibits a slightly lower accuracy value of 58%. This comparative analysis sheds light on the performance differences between the two CNN models in the context of skin disease classification.

### 7.6.2 *2 . Authors:* Ayushi Verma

Department of Computer Engineering and Applications, GLA University, Mathura, India

*Plant disease classification using Deep learning framework* (no date) *IEEE Conference Publication | IEEE Xplore*. Available at: https://ieeexplore.ieee.org/document/9844352.

In this report the authors writes about Agriculture stands as the primary livelihood source in India, facing significant challenges attributed to plant diseases that cause substantial devastation. This research addresses the urgent need for early disease detection in the agricultural sector, proposing an automatic system designed to identify and categorize plant diseases at their initial stages.

**Objective:** The core objective of this research is to develop an automatic disease detection system that plays a crucial role in preventing agricultural losses by diagnosing plant diseases and identifying their categories. Early intervention is key to mitigating the impact of diseases on crop health.

**Methodology:** The proposed system leverages a deep learning approach, specifically the Convolutional Neural Network (CNN), for the classification of plant diseases. Diseases are categorized into three main types: fungal, bacterial, and viral, based on symptoms observed on the leaf surface. This categorization ensures a targeted and effective strategy for disease management.

**Dataset and Training:** The dataset utilized in this study comprises an extensive collection of 64,963 samples. The training phase involves 80% of the dataset, with the remaining 20%

reserved for validation. The use of a sizable dataset enables the CNN model to effectively learn and generalize, leading to robust disease classification.

**Results:** The simulation results demonstrate the effectiveness of the Convolutional Neural Network. The trained model achieves an impressive accuracy rate of 99.12%, indicating its capability to accurately recognize and categorize plant diseases based on symptoms.

**Significance and Implications:** This research holds significant implications for Indian agriculture, offering a proactive and technology-driven approach to disease prevention. The high accuracy achieved by the CNN model positions it as a valuable tool for farmers, enabling timely intervention and fostering sustainable agricultural practices.

**Conclusion:** In conclusion, the proposed automatic disease detection system, employing Convolutional Neural Network technology, emerges as a promising solution for the agricultural challenges in India. By providing accurate and rapid disease classification, the system contributes to the overarching goal of preserving crop health and ensuring the sustainability of agriculture practices.

## 7.7 Similar Projects in Plant Disease Classification - Appendix

### 7.7.1 1. Authors: Mohanty, S. P

Hughes, D. and Salathé, M. (2016) "Using deep learning for Image-Based plant disease detection," *Frontiers in Plant Science*, 7. doi: 10.3389/fpls.2016.01419.

In this research Paper the authors talks about Crop diseases pose a significant threat to global food security, exacerbated by challenges in rapid identification, particularly in regions lacking adequate infrastructure. The intersection of widespread smartphone adoption and advancements in computer vision, propelled by deep learning, opens avenues for transformative smartphone-assisted disease diagnosis. Leveraging a comprehensive dataset of 54,306 images capturing both diseased and healthy plant leaves in controlled conditions, we employ a deep convolutional neural network for the identification of 14 crop species and detection of 26 diseases or their absence.

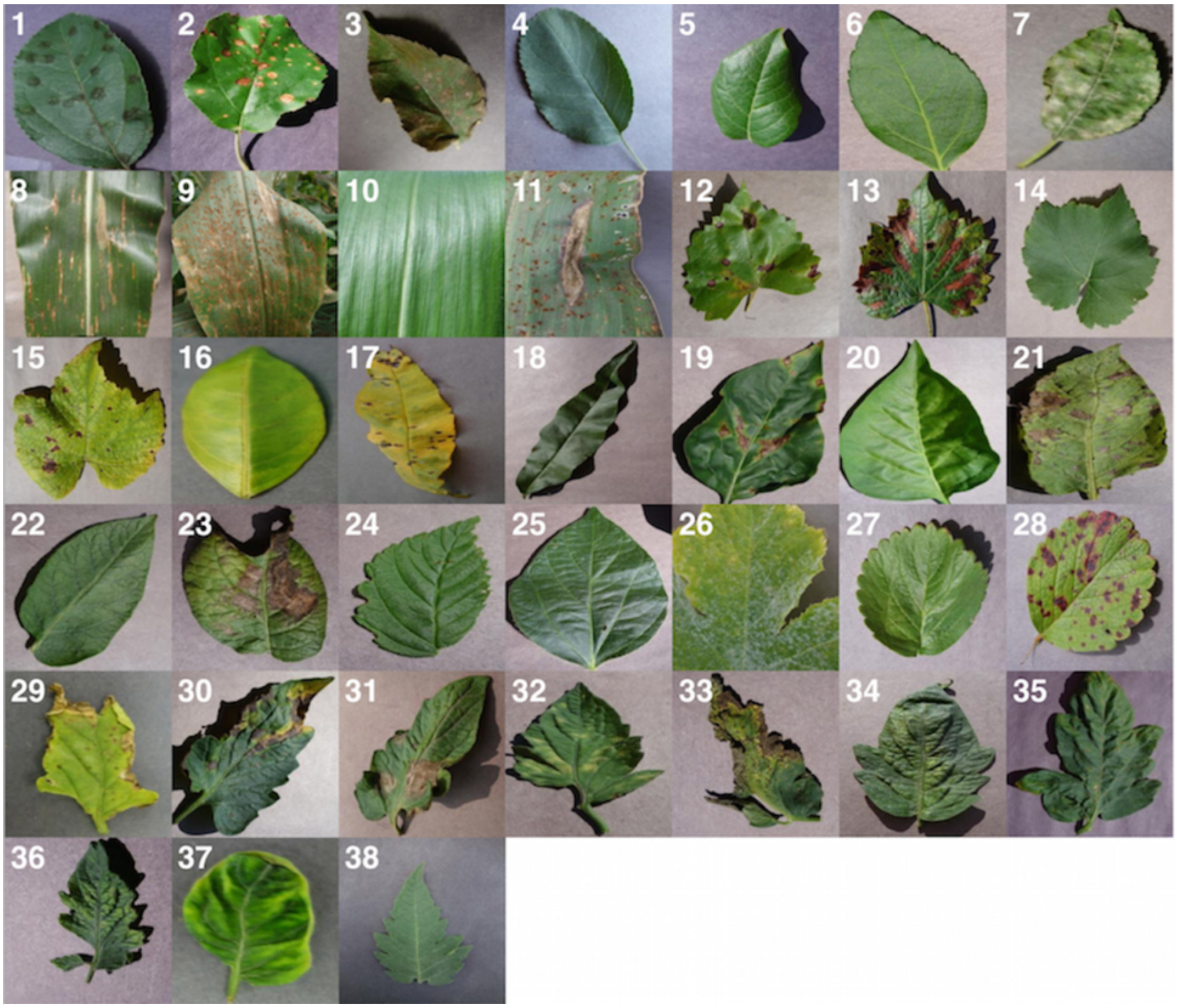



*Table 14: EXAMPLE OF LEAF IMAGES FROM THE PLANTVILLAGE DATASET, REPRESENTING EVERY CROP-DISEASE PAIR USED*

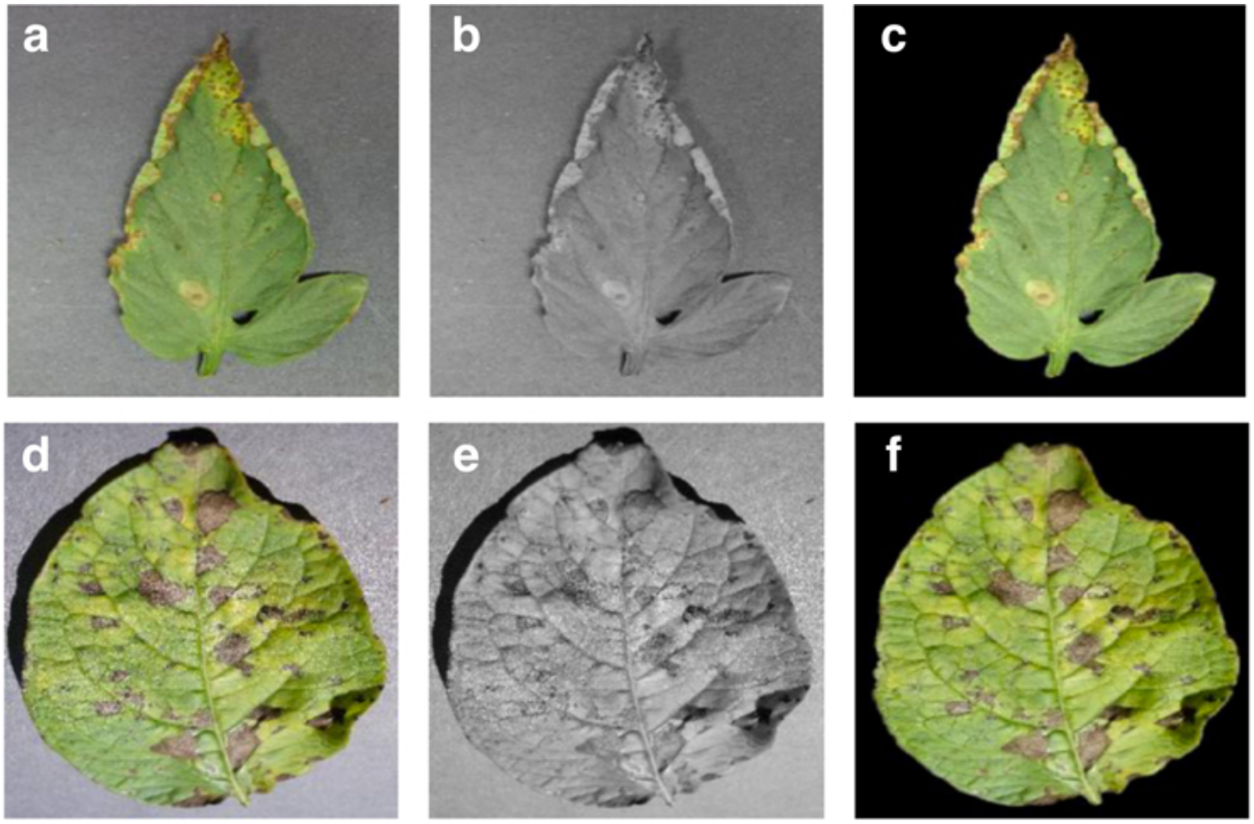



*Table 15: SAMPLE IMAGES FROM THE THREE DIFFERENT VERSIONS OF THE PLANTVILLAGE DATASET USED IN VARIOUS EXPERIMENTAL CONFIGURATIONS*

**1. Choice of deep learning architecture:**

AlexNet,

GoogLeNet.

**2. Choice of training mechanism:**

Transfer Learning,

Training from Scratch.

**3. Choice of dataset type:**

Color,

Gray scale,

Leaf Segmented.

**4. Choice of training-testing set distribution:**

Train: 80%, Test: 20%,

Train: 60%, Test: 40%,

Train: 50%, Test: 50%,

Train: 40%, Test: 60%,

Train: 20%, Test: 80%.

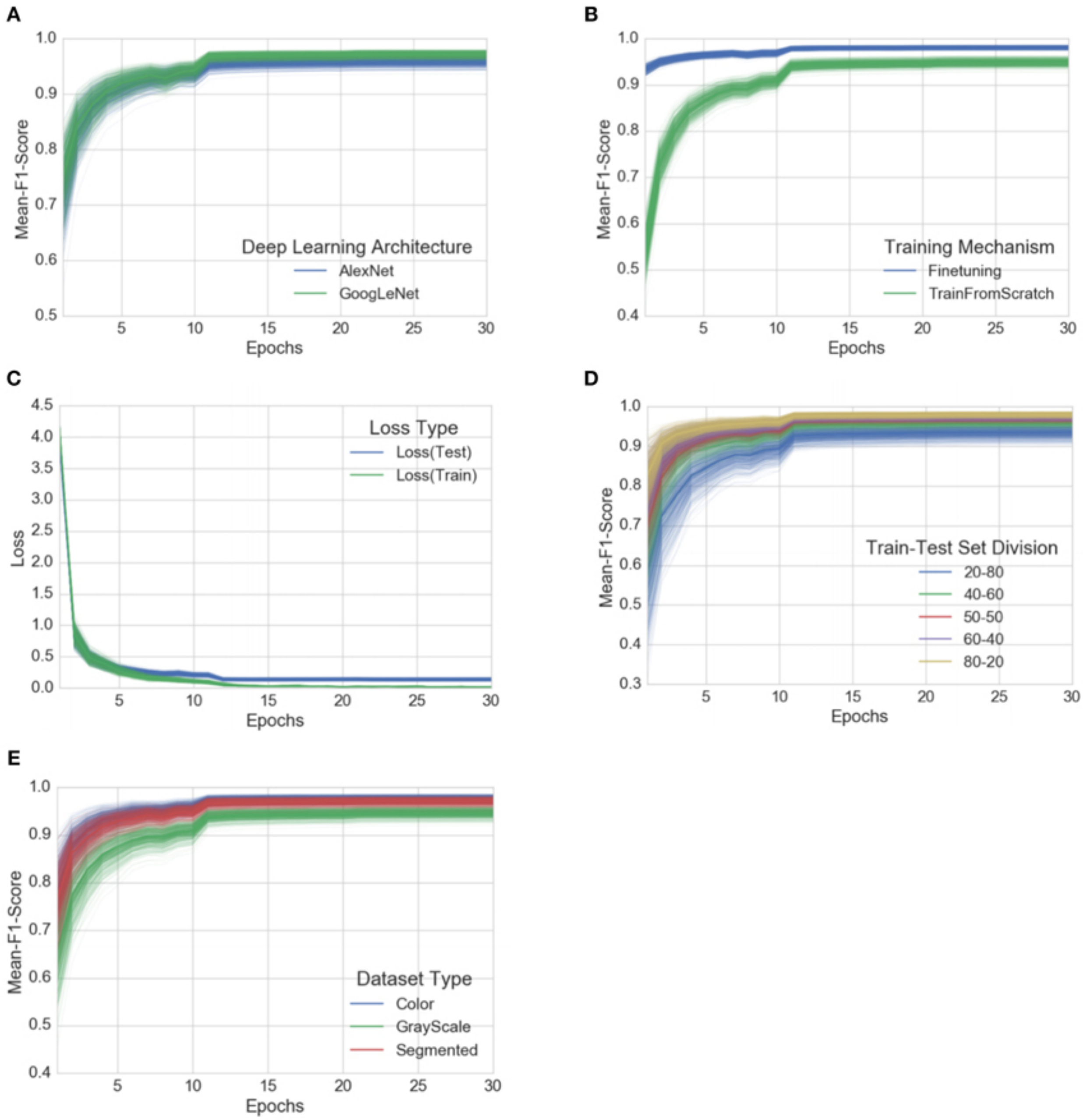



*Table 16: PROGRESSION OF MEAN F1 SCORE AND LOSS THROUGH THE TRAINING PERIOD OF 30 EPOCHS ACROSS ALL EXPERIMENTS, GROUPED BY EXPERIMENTAL CONFIGURATION PARAMETERS*

### 7.7.2 2. Authors: Mahlein, A.-K. (2016) “Plant disease detection by imaging sensors – parallels and specific demands for precision agriculture and plant phenotyping,” *Plant Disease*, 100(2), pp. 241–251. doi: 10.1094/pdis-03-15-0340-fe.

In this report the authors talks about The timely and precise identification of plant diseases is critical for optimizing plant production and minimizing losses in both quality and quantity of crop yields. Optical techniques, encompassing RGB imaging, multi- and hyperspectral sensors, thermography, chlorophyll fluorescence, and more recently, 3D scanning, have demonstrated their efficacy in developing automated, objective, and reproducible detection systems. These technologies contribute to the identification and quantification of plant diseases at early stages of epidemics.

Various platforms, ranging from proximal to remote sensing, offer multiscale monitoring capabilities, addressing single crop organs or entire fields. The integration of 3D scanning as an optical analysis adds valuable insights into crop plant vitality. The success of disease detection relies not only on advanced sensing technologies but also on sophisticated methods of data analysis. These methods extract novel insights from sensor data, particularly in the context of complex plant-pathogen systems.

Nondestructive, sensor-based approaches complement visual and molecular methods, enhancing the overall assessment of plant diseases. The primary domains benefiting from sensor-based analyses include precision agriculture and plant phenotyping. The marriage of cutting-edge optical techniques with innovative data analysis methods presents a promising frontier for advancing our understanding of plant diseases and fostering more effective strategies for their management and mitigation.

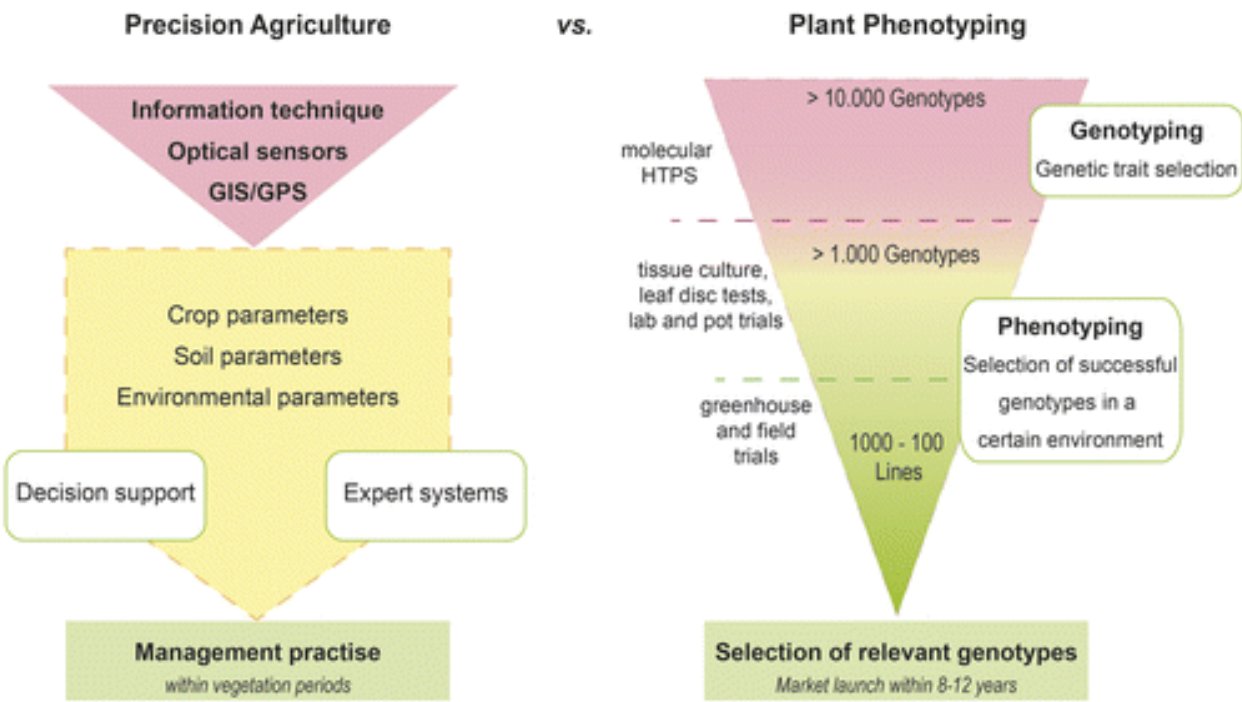



*Figure 58: Schematic diagrams of workflows and parameters in precision agriculture (left) and plant phenotyping (right).*

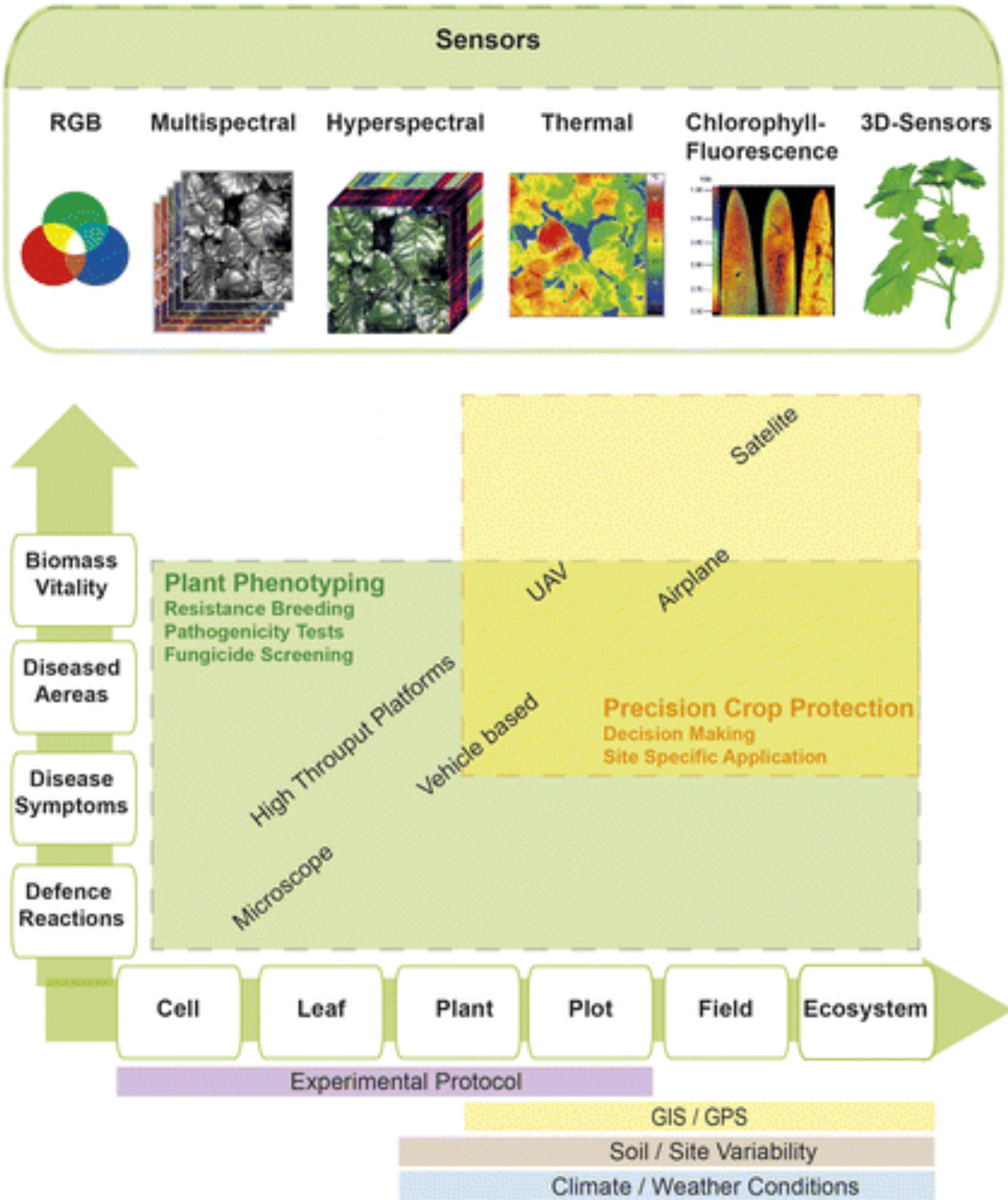



*Figure 59: Overview of current sensor technologies used for the automated detection and identification of host-plant interactions. These sensors can be implemented in precision agriculture applications and plant phenotyping on different scales from single cells*

Spectral sensors are generally categorized based on the spectral resolution (i.e., the number and width of measured wavebands), on their spatial scale, and on the type of detector, (i.e., imaging or nonimaging sensor systems). Multispectral sensors were the first spectral sensors invented. These sensors typically assess the spectral information of objects in several relatively broad wavebands. Multispectral imaging cameras may provide data, for instance, in the R, G, and B wavebands and in an additional near-infrared band. The evolution of modern hyperspectral sensors increased the complexity of the measured data by a spectral range of up to 350 to 2,500 nm and a possible narrow spectral resolution below 1 nm (**Steiner et al. 2008**). Contrary to nonimaging sensors, which average the spectral information over a certain area, hyperspectral imaging sensors provide spectral and spatial information for the imaged object. Hyperspectral data can be observed as huge matrices with spatial x- and y-axes, and the spectral information as reflectance intensity per waveband in the third dimension, z. The spatial resolution strongly depends on the distance between the sensor and the object. Thus, airborne or spaceborne, far range systems have lower spatial resolution than near-range or microscopic systems.

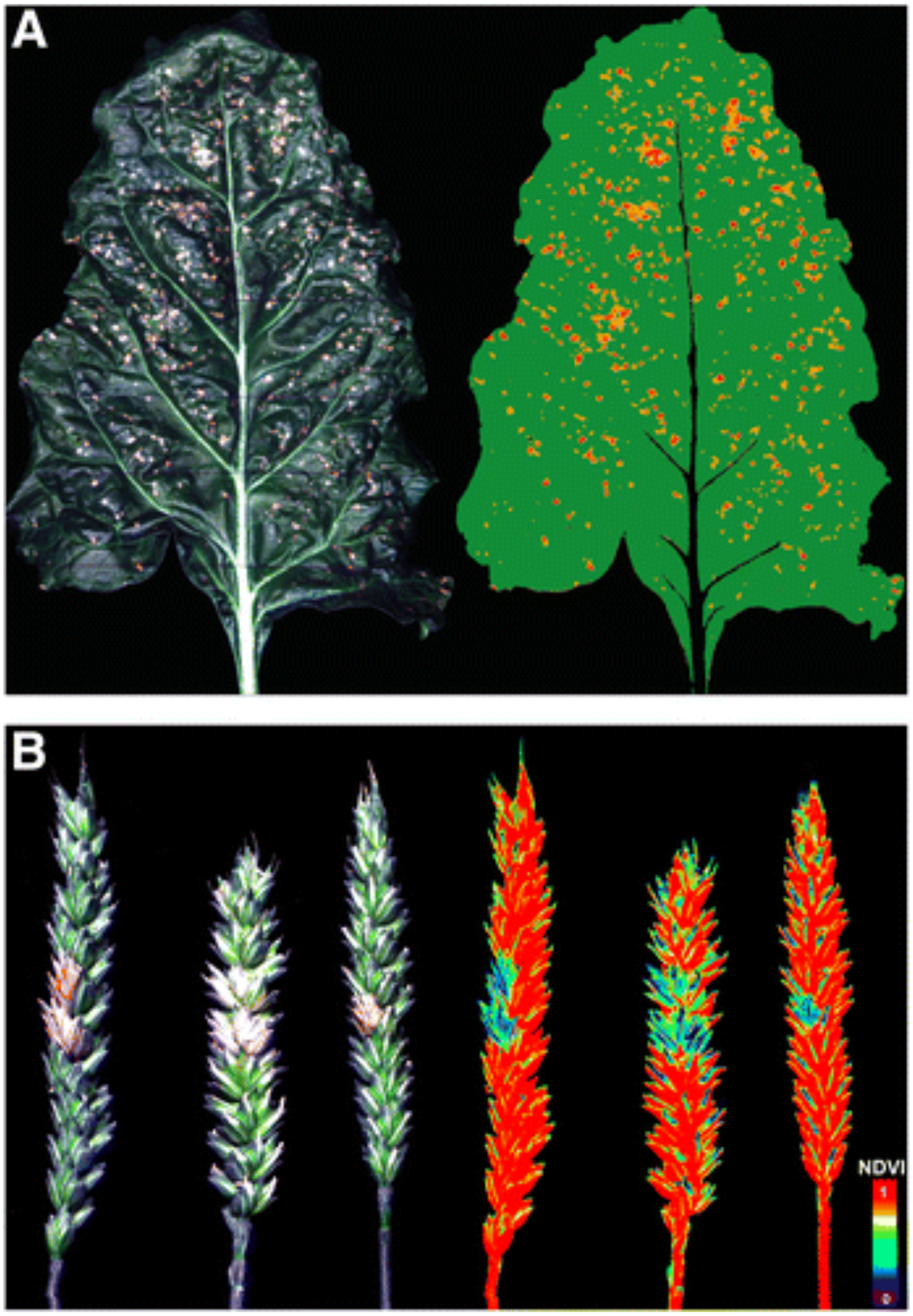



*Figure 60: Disease detection of fungal plant diseases based on hyperspectral images. A, Supervised classification (spectral angle mapper) of Cercospora leaf spot on sugar beet. The green color denotes healthy leaf tissue, the yellow color the border of Cercospora*

**Thermal sensors.**

Infrared thermography (IRT) assesses plant temperature and is correlated with plant water status (**Jones et al. 2002**), the microclimate in crop stands (**Lenthe et al. 2007**), and with changes in transpiration due to early infections by plant pathogens (**Oerke et al. 2006**). Emitted infrared radiation in the thermal infrared range from 8 to 12 µm can be detected by thermographic and infrared cameras and is illustrated in false color images, where each image pixel contains the temperature value of the measured object. In plant science, IRT can be used at different temporal and spatial scales from airborne to small scale applications.

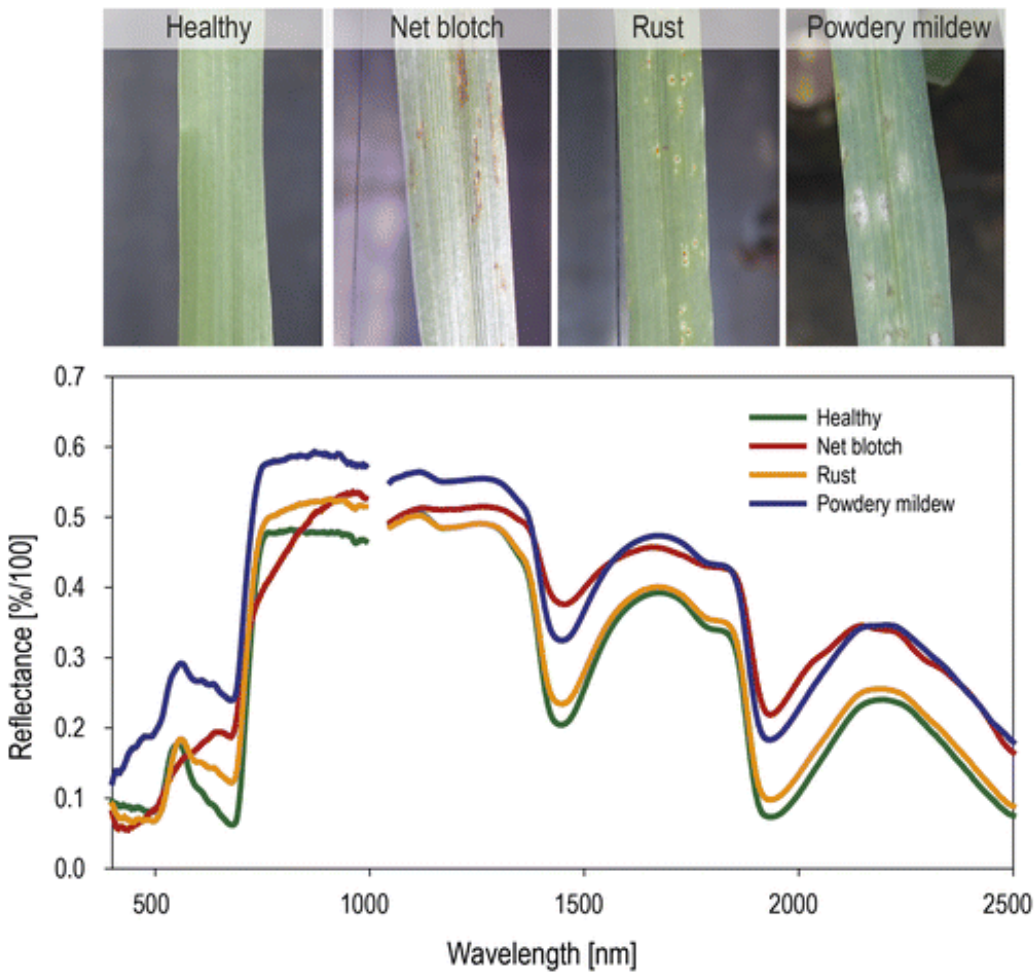



*Figure 61: Characteristic spectral signatures of barley leaves diseased with net blotch, rust, and powdery mildew, respectively.*

### 7.7.3 3. Plant Village

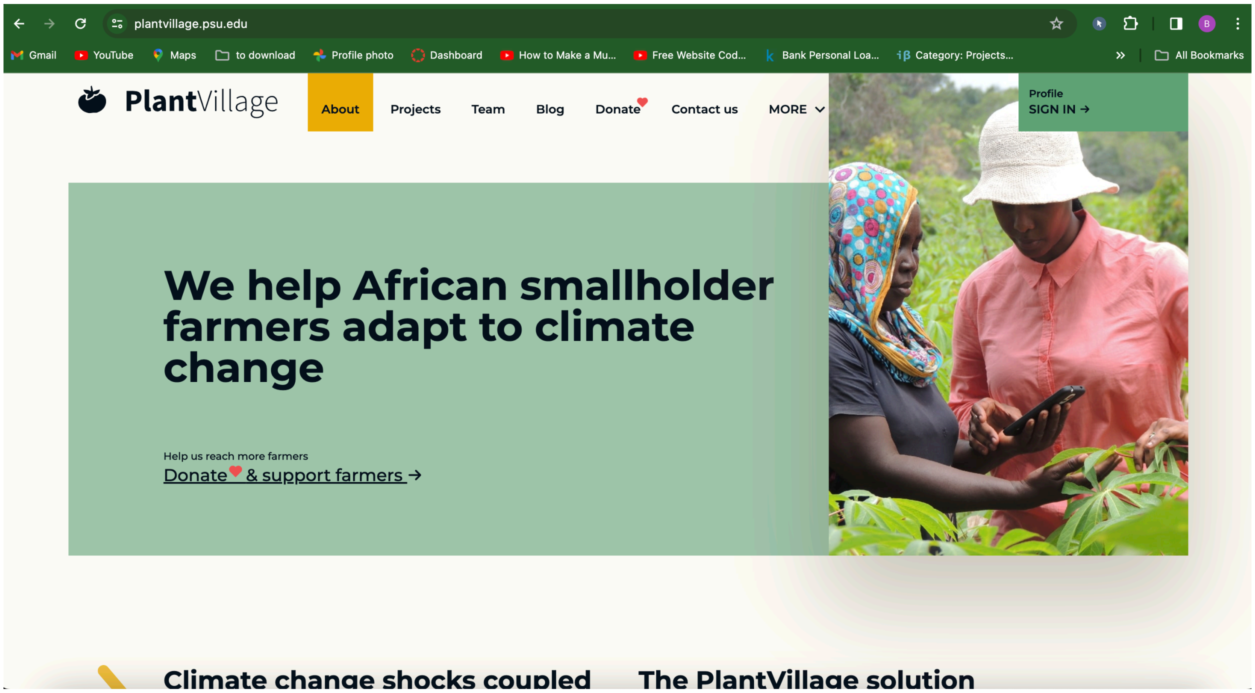



*Figure 62: Plant Village*

*PlantVillage* (no date). Available at: https://plantvillage.psu.edu/.

PlantVillage has developed a triple A model (Algorithmic Agricultural Advice) that works to increase the yield and profitability for millions of farmers. It is our goal to reach hundreds of millions in partnership with an ecosystem of farmer facing organizations and the farmers themselves. Our algorithms come from our integration of AI, satellite technology and our unique field force (the Dream Team). Once a farmer inputs 3 critical details (crop type, location, planting date) the algorithms within the PlantVillage engine can send out advice via smartphone, SMS, TV or real world social networks.

## 7.8 Progress Review -- Appendix

The "Plant Disease Classification System" project has made significant strides since its initiation, demonstrating substantial progress across various facets. This section provides a comprehensive overview of the achievements, challenges encountered, and the roadmap for the upcoming phases.

### 7.8.1 4.1 Accomplishments

**Prototype Development**

The project successfully reached the prototype development phase, featuring a basic CNN-based model and simplified web and mobile interfaces. This phase laid the foundation for subsequent iterations and enhancements based on user feedback.

- **User Feedback and Iteration:** The team actively collected feedback from users during this phase, allowing for iterative cycles of refinement. Valuable insights were gathered, guiding improvements in both the classification model and user interfaces.

**Testing and Validation**

Rigorous testing and validation procedures were implemented throughout the prototype phase to ensure the accuracy and reliability of the system. This phase marked a crucial step in refining the disease classification model and addressing any unforeseen issues.

- **Development of Improved Prototypes:** Improved prototypes were successfully developed in iterative cycles, incorporating enhancements and providing users with a more refined and feature-rich system.

**Ongoing User Engagement**

Continuous user engagement remained a priority, fostering a dynamic feedback loop with end-users. This approach ensures that the project aligns closely with user requirements and expectations.

- **Comprehensive Guides and Resources:** To facilitate user adoption, the team developed comprehensive guides and resources, enabling users to maximize the features of the application.

**Challenges Encountered**

**Data Quality and Quantity**

Ensuring the quality and diversity of the training dataset posed challenges, requiring dedicated efforts in data cleaning and augmentation.

**Technical Challenges**

Technical issues, including model instability and software bugs, were encountered during the development phase, necessitating additional time for debugging and refinement.

**Integration Complexity**

The integration of the disease database, sensor data, and multiple applications introduced complexities, leading to compatibility issues that required careful resolution.

## 7.9 Considered Mythodology

1. **Waterfall Model:**

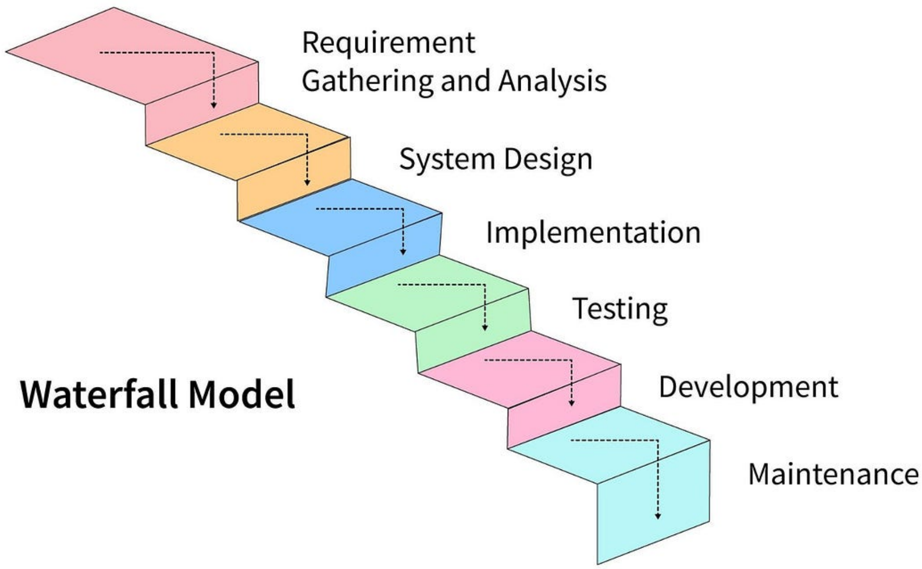



*Figure 63: WaterFall Model*

1. The **Waterfall Model** is a traditional and linear software development approach. It follows a sequential progression through distinct phases, where the output of one phase becomes the input for the next. The typical phases in the Waterfall Model are:
2. **Requirements:** Gathering and documenting project requirements.
3. **Design:** Creating a detailed system design based on the requirements.
4. **Implementation:** Developing the actual software based on the design.
5. **Testing:** Conducting comprehensive testing to identify and fix errors.
6. **Deployment:** Deploying the software for use by end-users.
7. **Maintenance:** Addressing any issues or enhancements after deployment.
8. The Waterfall Model is well-structured and easy to understand, making it suitable for projects with well-defined and unchanging requirements. However, its rigidity can be a drawback when dealing with evolving project needs.

2. **Prototype Model:**

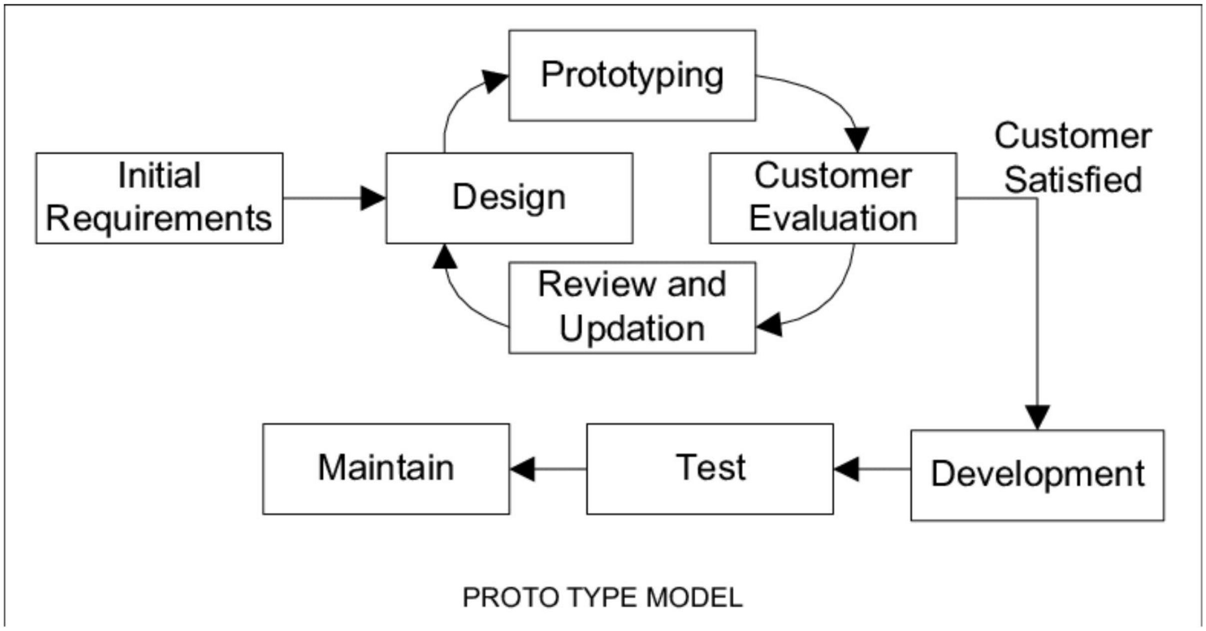



*Figure 64: Prototype Model*

The **Prototype Model** is an iterative development approach focused on creating prototypes or early versions of a system. Instead of waiting until the end of the development cycle to get user feedback, the Prototype Model encourages the creation of smaller, functional prototypes early in the process. Key steps in the Prototype Model include:

1. **Initial Requirements:** Gathering initial requirements for the system.
2. **Developing a Prototype:** Creating a preliminary version of the system based on the initial requirements.
3. **User Evaluation:** Seeking feedback from users on the prototype.
4. **Refinement:** Iteratively refining the prototype based on user feedback.
5. **Full System Development:** Once the prototype is approved, developing the full system.
9. This model is beneficial when requirements are unclear or likely to change. It allows for early user involvement, ensuring that the final system meets user expectations.

3. **Agile Methodology:**

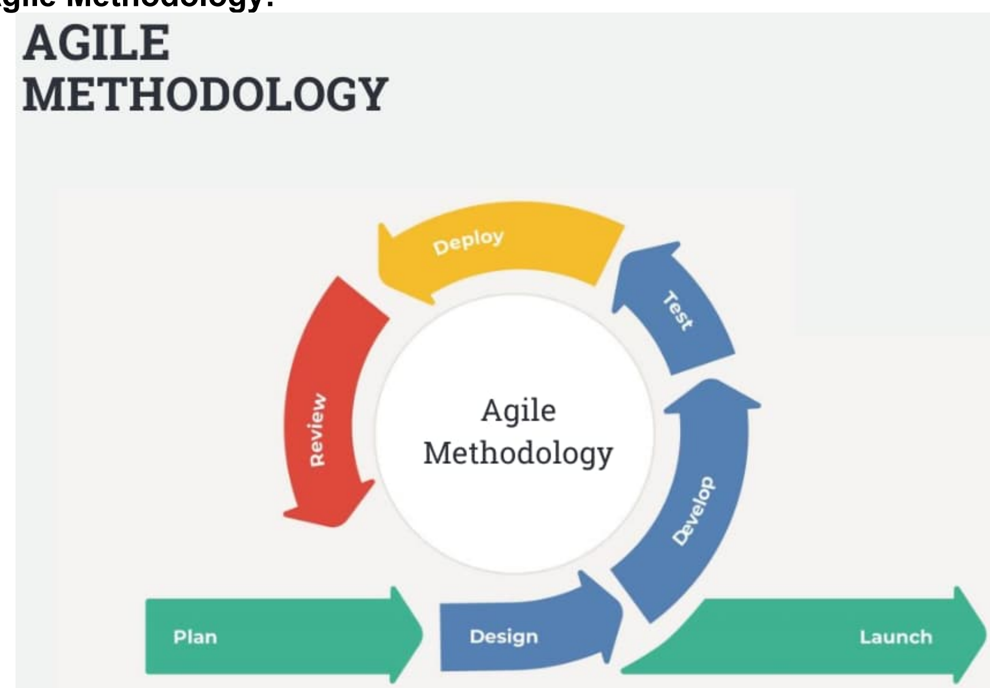


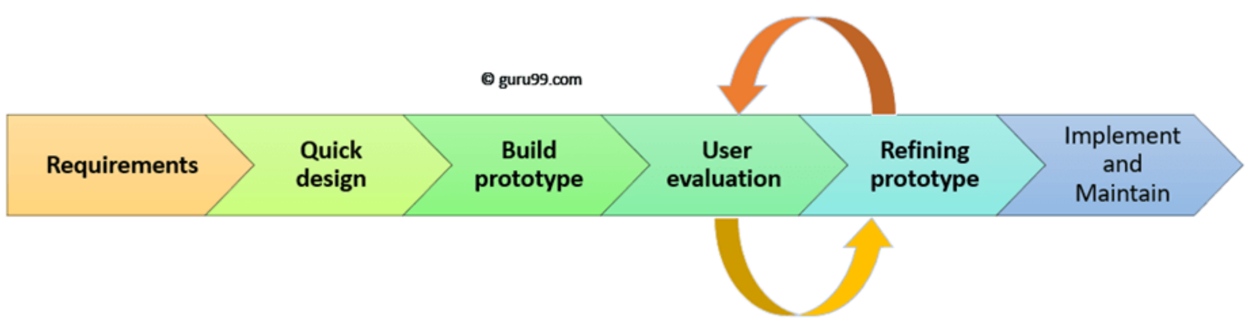

*Figure 65: Agile Model*

**Agile Methodology** is an iterative and flexible approach to software development that prioritizes adaptability and customer satisfaction. It emphasizes collaboration, frequent deliveries of small, functional increments, and responsiveness to changing requirements. Key principles and practices of Agile include:

1. **Iterative Development:** Delivering a working product in small, incremental cycles called iterations.
2. **Customer Collaboration:** Involving customers and stakeholders throughout the development process.
3. **Adaptability:** Embracing changes in requirements even late in the development process.
4. **Individuals and Interactions:** Valuing communication and collaboration among team members.
5. **Working Software:** Prioritizing the delivery of functional software at the end of each iteration.

10. Agile is well-suited for projects where requirements are expected to evolve, and rapid responses to changes are crucial. It promotes a customer-centric and adaptive development environment. Common Agile frameworks include Scrum and Kanban.

## 7.10 Selected Methodology: Prototype Models




i. **Iterative Development:**

The academic nature of this yearlong project necessitates a development approach that accommodates evolving user requirements within defined time constraints. The

Prototype Model aligns with this requirement as it emphasizes iterative cycles, allowing for the refinement of the system based on user feedback.

ii. **User Involvement:**

Frequent meetings with supervisors to showcase project progress and receive feedback are integral to this academic project. The Prototype Model facilitates this by providing tangible prototypes that can be presented and refined based on supervisor feedback. This iterative process ensures continuous improvement.

iii. **Efficient Resource Utilization:**

In an academic setting where time is a valuable resource, the Prototype Model offers efficiency. Code developed for one component can be reused in another, optimizing development time and resources. This is particularly crucial in a project with bounded time constraints.

iv. **Adaptability to Changes:**

The academic project involves continuous testing and refinement to control changes in the developing system. The Prototype Model, with its iterative nature, enables continuous testing at each phase of development. This proactive approach helps detect errors early and facilitates controlled changes in the software.

v. **Comprehensive Documentation:**

Documentation is pivotal in academic projects, and the Prototype Model ensures that documentation tasks are carried out at each phase. From requirement gathering in the inception phase to system deployment and maintenance, the Prototype Model includes systematic documentation, aligning with the academic project's documentation needs.

The Prototype Model, with its user-centric and iterative characteristics, is deemed suitable for this project to ensure a responsive and adaptable development process.

Continuing with the detailed description of the Prototype Model for the selected methodology:

vi. **User-Centric Design:**

1. The Prototype Model places a strong emphasis on user involvement throughout the development process. In the context of this project, which aims to create a Plant Disease Classification System, understanding and accommodating user feedback is critical. The

iterative cycles allow for the creation of tangible prototypes that users can interact with, providing valuable insights for refining the system.

vii. **Visual Representation:**

1. One of the distinctive features of the Prototype Model is its focus on creating visual representations of the system. For a project involving image-based disease detection, visual prototypes are particularly effective. These prototypes not only aid in communicating design concepts but also help users visualize the functionality of the system.

viii. **Early Detection of Issues:**

1. The Prototype Model facilitates the early detection of issues and challenges in the development process. By presenting working prototypes, potential problems can be identified and addressed at an early stage. This proactive approach contributes to the overall quality of the final system.

ix. **Flexible and Adaptive:**

1) The flexibility of the Prototype Model allows for adaptability to changes in requirements. As the project progresses and user needs evolve, the iterative nature of the model enables developers to incorporate changes efficiently. This flexibility is crucial for an academic project where requirements may evolve based on feedback and learning.

x. **Risk Mitigation:**

1. The Prototype Model supports risk mitigation by allowing developers to address potential risks and challenges incrementally. Early prototypes act as risk reduction mechanisms, providing opportunities to test critical functionalities and identify areas that may pose challenges later in the development process.

xi. **Enhanced Communication:**

- The visual nature of prototypes enhances communication among project stakeholders. For an academic project involving external and internal supervisors, clear communication of design concepts and progress updates is vital. The Prototype Model's visual representations facilitate effective communication.

In summary, the Prototype Model is well-suited for an academic project like the "Plant Disease Classification System." Its user-centric, iterative, and visual approach aligns with the project's goals of creating a responsive and adaptable system for accurate plant disease detection.

## 7.11 SRS Document

**System Description**

The Plant Disease Classification System comprises a CNN model trained on a diverse dataset, a web application for image uploads and instant diagnosis, and a mobile application for on-the-go access. The system's core functionality lies in accurately classifying diseases in 38 plant categories.

**Functional Requirements**

1. **Admin Login**
   a. The system should provide a secure login mechanism for administrators.
2. **Admin Dashboard**
   a. The admin dashboard should provide a centralized view of system activities and statistics.
3. **User Login**
   a. The system should provide a secure login mechanism for users.
4. **User Registration**
   a. The system should allow users to register for an account.
5. **User Dashboard**
   a. The user dashboard should provide a personalized view of the user's activities and results.
6. **User Plant Disease Detection**
   a. Users should be able to initiate plant disease detection using the system.
7. **User Image Upload**
   a. Users should have the ability to upload images for disease detection.

8. **Logout**
   a. Users and admins should be able to securely log out of the system.
9. **Deployment**
   a. The system should be deployable on web servers and mobile platforms.

**Non Functional Requirements**

**Usability**

**User-Friendly Interface**: The web and mobile applications should have an intuitive and user-friendly interface to enhance user experience.

**Reliability**

**System Availability**: The system should strive for 99% availability, ensuring users and administrators can access the platform reliably.

**Data Accuracy**: The accuracy of plant disease detection results should be above 95% to provide reliable information to users.

**Performance**

**Response Time**: The system should respond to user interactions within 2 seconds, providing a seamless and responsive experience.

**Security**

**Data Encryption**: All sensitive data, including user credentials and uploaded images, should be encrypted during transmission and storage.

**Access Control**: Role-based access control should be implemented, ensuring that users and administrators only have access to the functionalities relevant to their roles.

**Portability**

**Cross-Browser Compatibility**: The web application should be compatible with major web browsers, including Chrome, Firefox, and Safari.

**Cross-Platform Compatibility**: The mobile application should be compatible with both Android and iOS platforms.

**Scalability**

**System Scalability**: The system architecture should be scalable to accommodate future growth in terms of users and features.

**Maintainability**

**Code Maintainability**: The codebase should be well-documented and structured, facilitating ease of maintenance and future development.

**Compliance**

**Data Privacy Regulations**: The system should comply with relevant data privacy regulations, ensuring the protection of user information.

**Documentation**

**Technical Documentation**: Comprehensive technical documentation should be provided to assist developers, system administrators, and users in understanding and using the systems

## 7.12 Entity Relationship Diagrams - appendix

More about this in the Appendix: Entity Relationship Diagram

### 7.12.1 Possible Entities

- Users
- Profiles
- Classifications
- Recommendations
- Plants
- Images

### Relationships between relationships

- One user can have one profile.
- One profile can belong to one user.
- One user can have many classifications.
- One classification can belong to many users.
- One profile can have many recommendations.
- One recommendation can belong to one profile.
- One classification can have many plants.
- One plant can belong to one classification.
- One plant can have many images.
- One image can belong to one plant.

The diagram shows nine classes grouped into four sections: "Tokens", "Users", "Classification", and "Recommendation". These sections interact with an additional class called "Images".

**Within the "Tokens" section, the class "Token" has four attributes:**

- Token_ID (PK): This is likely the primary key for the "Token" class and uniquely identifies each token.
- Access Token: This attribute may store a token that allows access to the application.
- Refresh Token: This attribute may store a token that can be used to obtain a new access token.

**There are two additional classes in the "Tokens" section:**

- View Token: This class may represent a token that allows access to specific views within the application.
- Profile: This class may represent a user's profile information.

**The "Users" section includes the class "Users" with the following attributes:**

- User_ID (PK): This is likely the primary key for the "Users" class and uniquely identifies each user.
- Username: This attribute stores the username for each user.
- Password: This attribute stores the password for each user.
- Phone Number: This attribute stores the phone number for each user.
- Country: This attribute stores the country for each user.
- Email: This attribute stores the email address for each user.

**The "Users" section also has a method called "Login()". This method may be used to log in a user to the application.**

The "Classification" section includes the class "Classification" with the following attributes:

- Classification ID (PK): This is likely the primary key for the "Classification" class and uniquely identifies each classification.
- Model(): This method may return a machine learning model used for classification.

**There is also a class called "Plants" in the "Classification" section. This class may represent different types of plants.**

**The "Recommendation" section includes the class "Recommendation" with the following methods:**

- Plant_rec(): This method may recommend plants to users based on their classifications.

Finally, the "Images" section contains the class "Images" with the following attributes:

- Image_ID (PK): This is likely the primary key for the "Images" class and uniquely identifies each image.
- Users_ID (FK): This attribute may be a foreign key that references the "Users" class, indicating which user uploaded the image.
- Image_name: This attribute stores the name of the image file.

**There is also a method called "Upload_img()" in the "Images" section. This method may be used to upload an image to the application.**

## 7.13 Data Dictionary -appendix

**Table Name: Token**

| Column Name | Data Type | Description |
|---|---|---|
| Token_ID (PK) | Integer | Unique identifier for each token |
| Access Token | String | Token that allows access to the application |
| Refresh Token | String | Token that can be used to obtain a new access token |

*Table 17: Data Dictionary of Token*

**Table Name: Users**

| Column Name | Data Type | Description |
|---|---|---|
| User_ID (PK) | Integer | Unique identifier for each user |
| Username | String | Username for each user |
| Password | String | Password for each user |
| Phone Number | String | Phone number for each user |
| Country | String | Country for each user |
| Email | String | Email address for each user |

*Table 18: Data Dictionary of Users*

**Table Name: Classification**

| Column Name | Data Type | Description |
|---|---|---|
| Classification ID (PK) | Integer | Unique identifier for each classification |
| Model | String | Name or identifier of the machine learning model used for classification |

**Table Name: Plants**

| Column Name | Data Type | Description |
|---|---|---|
| Plant_ID (PK) | Integer | Unique identifier for each plant |
| Classification ID (FK) | Integer | Foreign key referencing the Classification table, indicating the classification of the plant |
| Plant_name | String | Name of the plant |
| (Other attributes as needed for plant details) | | |

**Table Name: Images**

| Column Name | Data Type | Description |
|---|---|---|
| Image_ID (PK) | Integer | Unique identifier for each image |
| User_ID (FK) | Integer | Foreign key referencing the Users table, indicating which user uploaded the image |
| Image_name | String | Name of the image file |

**Table Name: Profile**

| Column Name | Data Type | Description |
|---|---|---|
| Profile_ID (PK) | Integer | Unique identifier for each profile |
| User_ID (FK) | Integer | Foreign key referencing the Users table, indicating the user associated with the profile |
| (Other attributes as needed for profile information) | | |

**Table Name: Recommendation**

| Column Name | Data Type | Description |
|---|---|---|
| Recommendation_ID (PK) | Integer | Unique identifier for each recommendation |
| User_ID (FK) | Integer | Foreign key referencing the Users table, indicating the user who received the recommendation |
| Plant_ID (FK) | Integer | Foreign key referencing the Plants table, indicating the recommended plant |