Indian Institution of Information Technology Surat

Software Engineering (CS 401)

Project on Plant Disease Prediction

FloraWise

Submitted By

Full Name	Student ID
Bhumil Rangholiya	UI22CS66

Submitted To

Mrs. Jiby Babin

Mr. Rishi Sharma

May 16, 2024

Acknowledgements

I express my heartfelt gratitude to the Faculty Coordinators, Mr.Rishi Sharma and Mrs.Jiby Babin, whose unwavering guidance and wisdom illuminated my path throughout this journey. Their mentorship, keen insights and encouragement infused my efforts with direction and purpose, shaping FloraWise into its finest form.
I extend my sincere appreciation to all contributors who lent their expertise, whether through direct involvement or indirect support. Your valuable input, feedback and assistance played an integral role in the project's development and success.
My deepest thanks go to my friends and family for their unwavering support, understanding and patience. Their belief in me served as a constant source of motivation, sustaining me through every challenge and triumph.
Lastly, I acknowledge the generous resources, facilities and productive environment provided by the Indian Institute of Information Technology, which facilitated the project's execution and the realization of its goals.
In closing, I am humbled and honored to have been part of this endeavor and I am grateful for the collective effort and collaboration that brought FloraWise to fruition.

Abstract

In botany, identifying and classifying plant species is crucial, often relying on leaf structures such as shape, texture, vein patterns and color. FloraWise addresses the challenges of plant identification, especially in remote areas, by developing an automatic system where users can upload leaf images to a website. Using Convolutional Neural Networks (CNNs) with tools like TensorFlow and Keras, FloraWise accurately identifies plant species from single leaf images. Additionally, it swiftly detects and classifies plant diseases, which is essential for crop health and growth, overcoming the tedious and error-prone traditional methods. By leveraging advanced image processing and AI, FloraWise enhances crop yield, minimizes disease-related losses and supports agricultural sustainability and food security.

Declaration

I declare that this is to certify...

- This report comprises my original work towards the degree of Bachelor of Technology in Computer Science and Engineering at Indian Institute of Information Technology (IIIT) Surat and has not been submitted elsewhere for a degree.
- Due acknowledgement has been made in the text to all other material used.

UI22CS66
(Bhumil Rangholiya)

Contents

Contents

1 | Introduction

1.1 | Project Overview

FloraWise is an innovative application designed to enhance agricultural practices by utilizing deep learning technology for plant recognition and disease detection. By providing accurate and timely information about plant species and diseases based on input leaf images, FloraWise aims to support botanists, farmers, gardeners, agricultural researchers, and enthusiasts in identifying plant species and diagnosing plant diseases.

1.2 | Problem Statement

In agriculture, early detection of plant diseases is critical to preventing crop loss and ensuring healthy growth. Traditional methods of plant disease detection are time-consuming and often require expert knowledge, making them inaccessible to many. FloraWise addresses this challenge by offering a user-friendly tool that leverages deep learning to recognize plant species and detect diseases from leaf images.

1.3 | Objectives

1. Plant Species Recognition: Accurately classify plant species from uploaded leaf images.
2. Disease Detection: Identify and classify diseases affecting the plants based on leaf images.
3. User Accessibility: Provide a simple and intuitive interface for users to upload images and receive diagnostic results.

1.4 | Intended Audience

FloraWise is intended for a diverse audience, including:

1. Botanists: For research and educational purposes.
2. Farmers: To monitor and maintain the health of their crops.
3. Gardeners: For maintaining healthy plants in their gardens.
4. Agricultural Researchers: To study plant diseases and improve agricultural practices.
5. Plant Enthusiasts: For personal knowledge and interest in plant care.

1.5 | System Overview

FloraWise employs Convolutional Neural Networks (CNNs) to analyze leaf images, extract features, and classify them into different plant species and disease categories. The system is built using TensorFlow and Keras, ensuring robust and scalable performance.

1.6 | Key Features

1. Image Upload: Users can easily upload leaf images for analysis.
2. Real-time Analysis: The system processes images and provides results quickly.
3. Accuracy: High accuracy in plant species recognition and disease detection, validated through extensive training and evaluation.
4. User-friendly Interface: An intuitive interface guiding users through the image upload and result viewing process.

2 | Tools and Technologies

This section provides an overview of the software project management aspects for FloraWise, a plant recognition and disease detection system. FloraWise uses deep learning techniques to identify plant species and detect diseases from leaf images. The assignment will cover identification of tools, technologies, and languages for implementing this project.

2.1 | Tools, Technologies and Languages

The FloraWise project relies on several key technologies, tools & programming languages, including:

- Kaggle : for dataset storage and retrieval.
- Python : The primary language for deep learning implementation with extensive libraries and community support.
- Deep Learning Frameworks
  - TensorFlow : Used to create and train the deep learning model.
  - Keras : Provides a high-level interface for building neural networks.
- Matplotlib & Seaborn(Data visualization) : For visualizing training results, accuracy and confusion matrices.
- Pandas (Data Management) : Used for data manipulation and processing.
- Google colab : Used for interactive development and documentation.
- Streamlit(Deployment Platform) : Used to deploy the trained model as a web application.

2.2 | Dataset & Model overview

- Dataset
  The dataset for this project is the ”New Plant Diseases Dataset (Augmented)” from Kaggle. It contains a large collection of labeled images depicting various plant species and plant diseases. The dataset is divided into training, validation and testing sets, requiring some preprocessing steps like resizing, normalization and augmenting to enhance the model.
- Model(CNN) Architecture
  The deep learning model for FloraWise is a Convolutional Neural Network (CNN) with multiple convolutional layers for feature extraction, followed by dense layers for classification. Here's an outline of the architecture:
  - TensorFlow : Used to create and train the deep learning model.
  - Keras : Provides a high-level interface for building neural networks.
- Convolutional Layers : The model uses several convolutional layers with ReLU activation and max-pooling for feature extraction.
- Dense Layers : Following the convolutional layers, dense layers are used to classify plant species and diseases.
- Dropout Layers : To prevent overfitting, dropout layers are added.
- Output Layer : The output layer uses softmax activation for multi-class classification.

2.3 | Training and Evaluation

- Data Preprocessing : Preprocessing the dataset to create training and validation sets.
- Model Training : Training the model using backpropagation and Adam optimizer with a learning rate of 0.0001.
- Evaluation Metrics : The model's performance is evaluated using metrics like accuracy, loss, confusion matrix, and classification report.

2.4 | Results

After training for 10 epochs, the model achieved a high accuracy(Training accuracy: 96.6683% & Validation accuracy: 93.4498%) on both training and validation sets, indicating good learning and generalization. Challenges faced during the process included preventing overfitting and ensuring a balanced dataset. To address these issues, dropout layers were added and the training set was shuffled.

3 | Requirement Analysis and Specification

3.1 | Introduction

The purpose of this Software Requirement Specification (SRS) document is to outline the functional and nonfunctional requirements for the development of a search engine project for a college website. The search engine will enable users to efficiently find relevant information within the college website, enhancing user experience and accessibility.

3.1.1 | Scope

The scope of this project includes the following activities
Implementing a search functionality within the college website.
Providing search results in a structured and user-friendly manner.
Allowing users to search for specific information, including academic programs, faculty details, departmental information, events, and other relevant content.

3.1.2 | Out of Scope

The following activities are considered out of scope for this project
Integration of external search engines.
Advanced search features beyond basic keyword-based search.
Search engine optimization (SEO) for external search engines.

3.2 | Functional Requirements

3.2.1 | User Interface

The search engine shall provide a user-friendly interface for users to input search queries.
The interface shall be integrated seamlessly within the college website's existing design and navigation structure.

3.2.2 | Search Functionality

The search engine shall support keyword-based search queries.
The search engine shall return relevant results based on the search query.
Users shall be able to enter search queries in a text box and submit them for processing.

3.2.3 | Search Results

The search results shall be displayed in a structured format, with relevant titles, descriptions, and links to the corresponding pages or documents.
The search engine shall prioritize results based on relevance and importance, using appropriate algorithms.

3.2.4 | Filtering and Sorting

Users shall have the option to filter and sort search results based on various criteria, such as date, relevance, and category.

3.3 | Non-Functional Requirements

3.3.1 | Performance

The search engine shall have low latency and provide quick responses to user queries.
The system shall be able to handle multiple concurrent search requests without significant performance degradation.

3.3.2 | Security

The search engine shall employ secure authentication mechanisms to protect user information.
User search queries and results shall be encrypted during transmission to prevent unauthorized access.

3.3.3 | Reliability

The search engine shall be available and operational 24/7, with minimal downtime for maintenance and updates.
The system shall have backup and recovery mechanisms in place to ensure data integrity and availability.

3.3.4 | Usability

The search engine interface shall be intuitive and easy to use, catering to users with varying levels of technical proficiency.

Error messages and prompts shall be provided to guide users in case of incorrect inputs or search queries.

3.3.5 | Scalability

The system architecture shall be designed to accommodate future expansion and increased usage.

The search engine shall be able to handle a growing volume of content and users without sacrificing performance or responsiveness.

3.4 | Introduction

3.4.1 | Product scope

This system aims to recognize plant species and detect diseases through deep learning technology. It will provide accurate and timely information about plant species and diseases based on input leaf images.

3.4.2 | Product value

Enhances agricultural practices by facilitating early disease detection, promoting plant health and aiding in the accurate identification of plant species.

3.4.3 | Intended audience

Botanists, farmers, gardeners, agricultural researchers and botany enthusiasts who require assistance in identifying plant species and diagnosing the plant diseases.

3.4.4 | Intended use

The system is intended to be used as a tool for uploading leaf images, analyzing them using deep learning models and providing users with relevant information about plant species and diseases.

3.4.5 | General description

The system utilizes deep learning models, specifically CNNs, to analyze leaf images and extract features for plant recognition and disease detection.

3.5 | Functional requirements

- The system shall allow users to upload leaf images for analysis.
- The system shall recognize and classify plant species from uploaded leaf images.
- The system shall detect and classify diseases in plants from uploaded leaf images.
- The system shall provide users with accurate and timely results regarding plant species and diseases.

3.6 | External interface requirements

3.6.1 | User interface requirements

The user interface shall be intuitive and user-friendly, allowing users to easily upload leaf images and view analysis results.The interface shall provide clear instructions and feedback to guide users through the process.

3.6.2 | Hardware interface requirements

The system shall be compatible with standard web browsers and smartphone cameras for image capture.

3.6.3 | Software interface requirements

The system shall integrate with deep learning frameworks such as TensorFlow and Keras for image analysis and classification.

3.7 | Communication interface requirements

Secure data transmission protocols (e.g., HTTPS) shall be employed to ensure the confidentiality and integrity of user data.

3.8 | Non-functional requirements

3.8.1 | Security

User data shall be encrypted and stored securely to prevent unauthorized access.Access controls shall be implemented to restrict user access to sensitive information.

3.8.2 | Capacity

The system shall be capable of handling a large number of concurrent user requests without significant degradation in performance.

3.8.3 | Compatibility

The system shall be compatible with major web browsers (e.g Chrome, Firefox, Safari).

3.8.4 | Reliability

The system shall provide accurate results with minimal errors or false positives in plant species recognition and disease detection.

3.8.5 | Scalability

The system architecture shall be designed to scale horizontally to accommodate increasing user demands.

3.8.6 | Maintainability

The system shall be modular and well-documented to facilitate easy updates and maintenance as technology evolves.

3.8.7 | Usability

The user interface shall be intuitive and accessible, with clear instructions and guidance for users of all skill levels.

3.8.8 | Performance

The system shall provide fast response times for image analysis and result retrieval to ensure a seamless user experience.
This detailed SRS outlines the functional and non-functional requirements for the development of a plant recognition and disease detection system, aiming to provide valuable assistance to users in the field of agriculture and botany.

4 | Design

4.1 | Structural Diagrams(DFDs)

4.1.1 | Data Flow Diagram Level-0

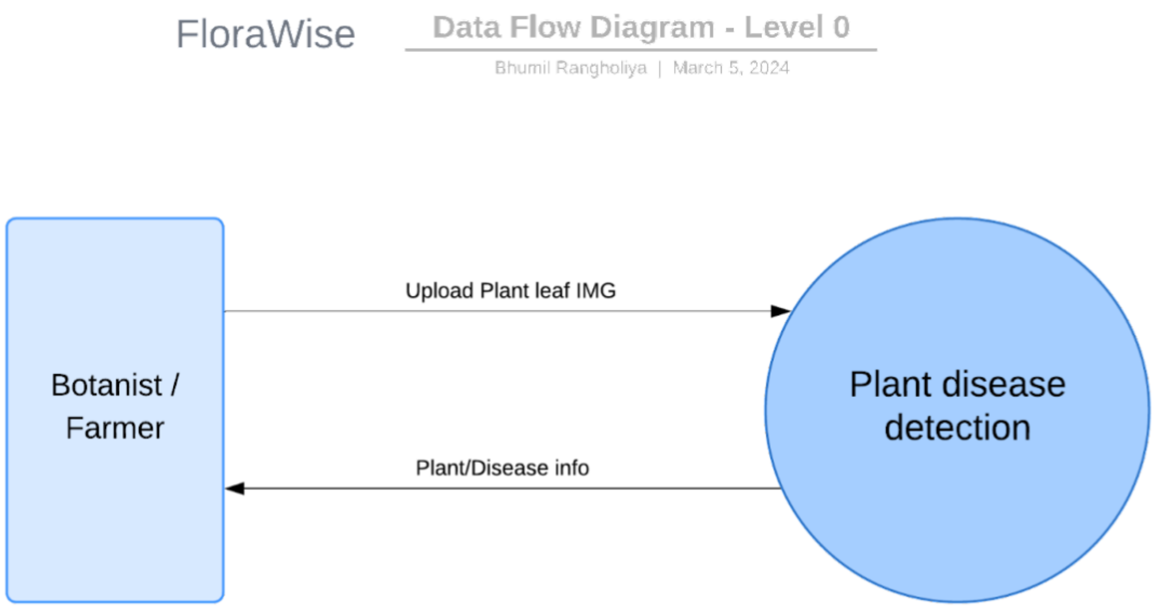

Figure 4.1: Data Flow Diagram Level-0

4.1.2 | Data Flow Diagram Level-1

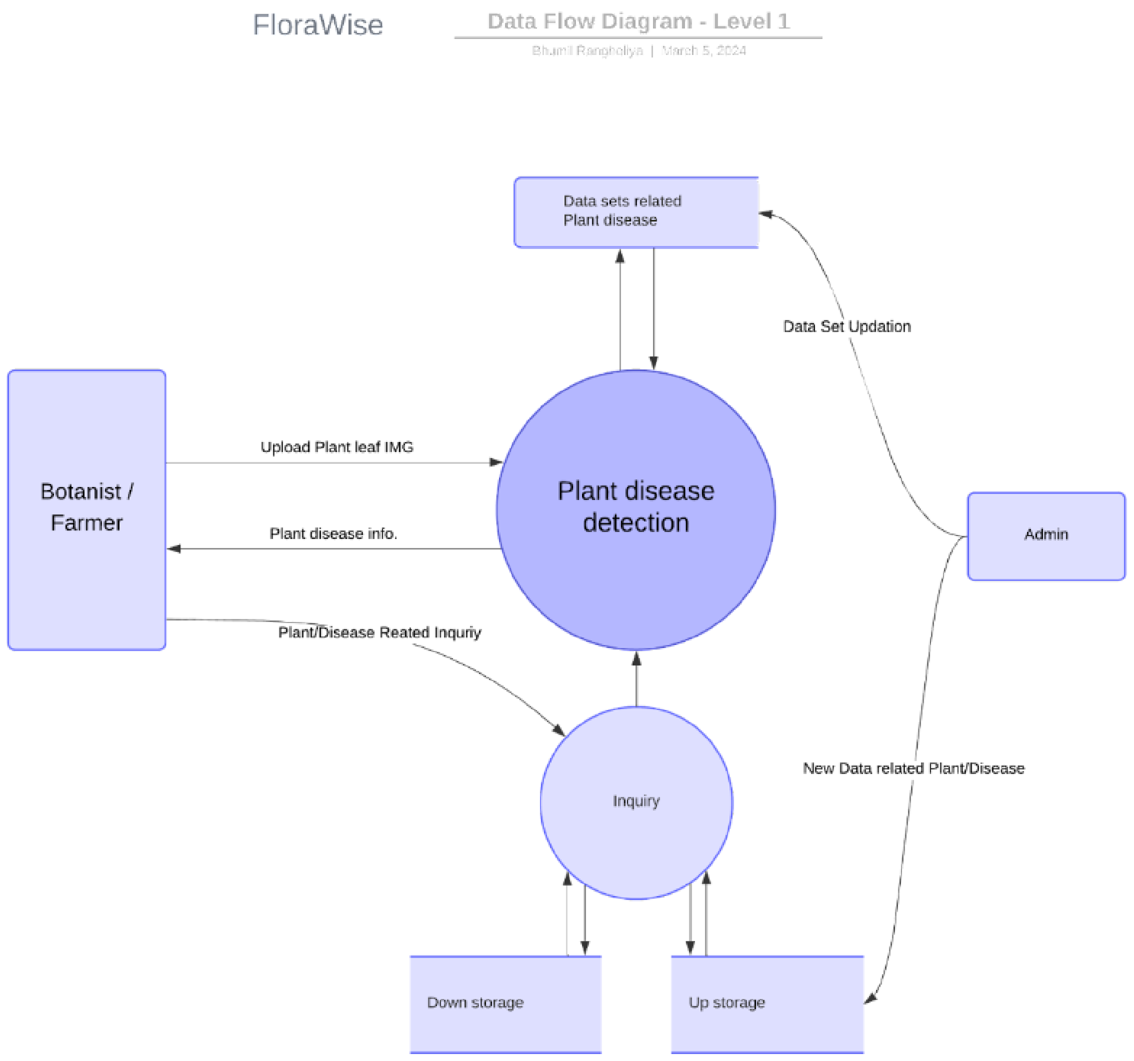
FloraWise
Data Flow Diagram - Level 1
Bhumil Rangholiya | March 5, 2024
Data sets related Plant disease
Data Set Updation
Botanist / Farmer
Upload Plant leaf IMG
Plant disease info.
Plant disease detection
Admin
Plant/Disease Reated Inquriy
Inquiry
New Data related Plant/Disease
Down storage
Up storage

Figure 4.2: Data Flow Diagram Level-1

4.1.3 | Data Flow Diagram Level-2

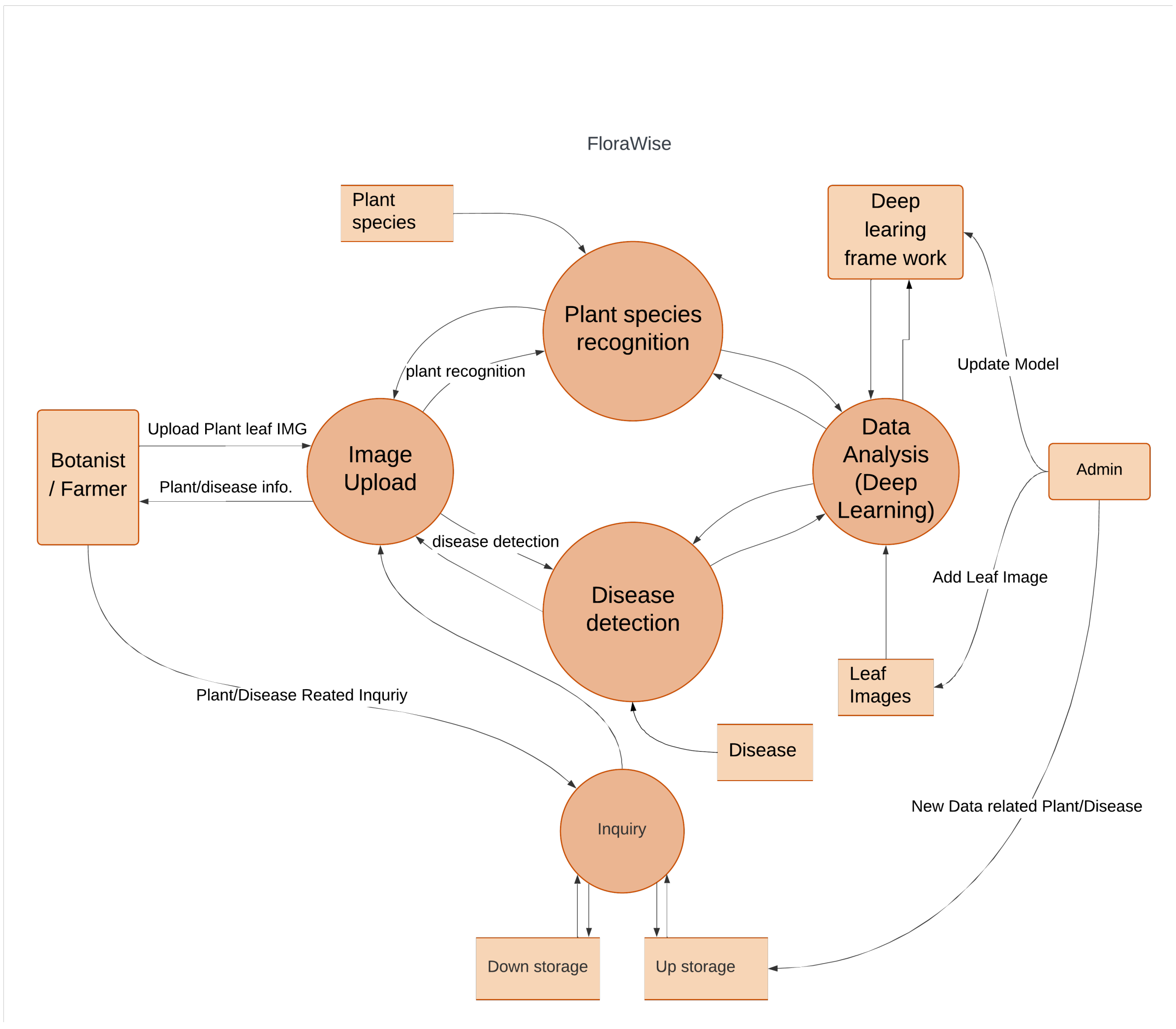

Figure 4.3: Data Flow Diagram Level-2

4.2 | UML Diagrams

4.2.1 | Use Case Diagram

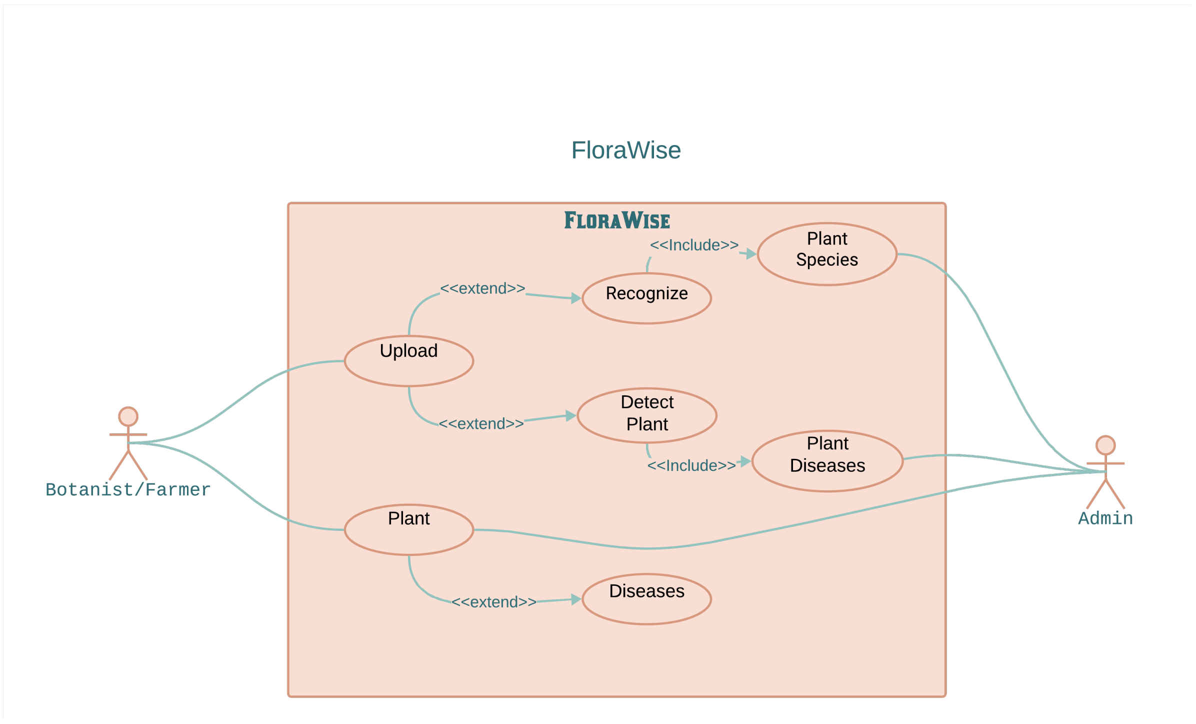

Figure 4.4: Use Case Diagram

4.2.2 | Activity Diagram

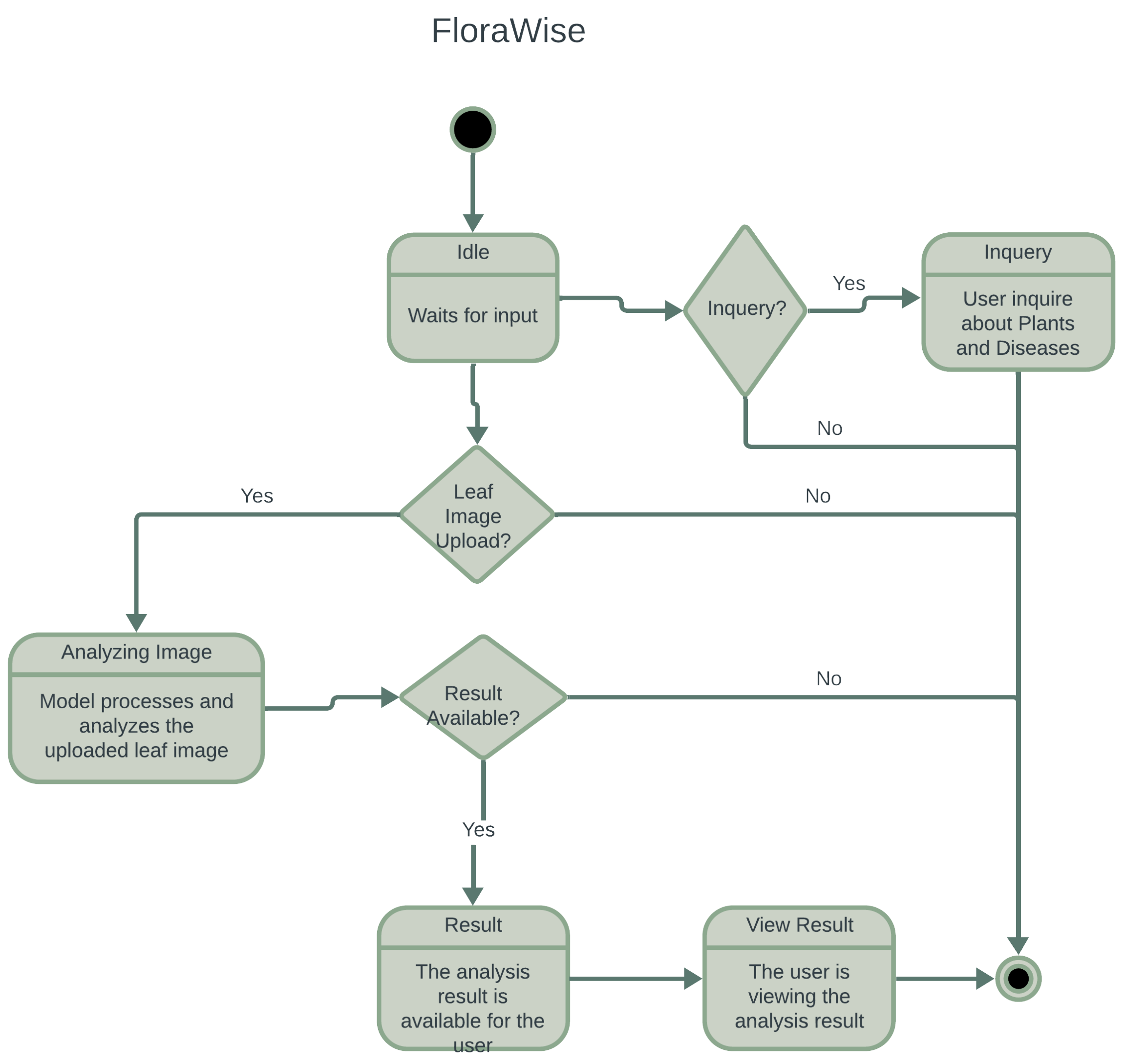
FloraWise
Idle
Waits for input
Inquery?
Yes
Inquery
User inquire about Plants and Diseases
No
Yes
Leaf Image Upload?
No
Analyzing Image
Model processes and analyzes the uploaded leaf image
Result Available?
No
Yes
Result
The analysis result is available for the user
View Result
The user is viewing the analysis result

Figure 4.5: Activity Diagram

4.2.3 | State Chart Diagram

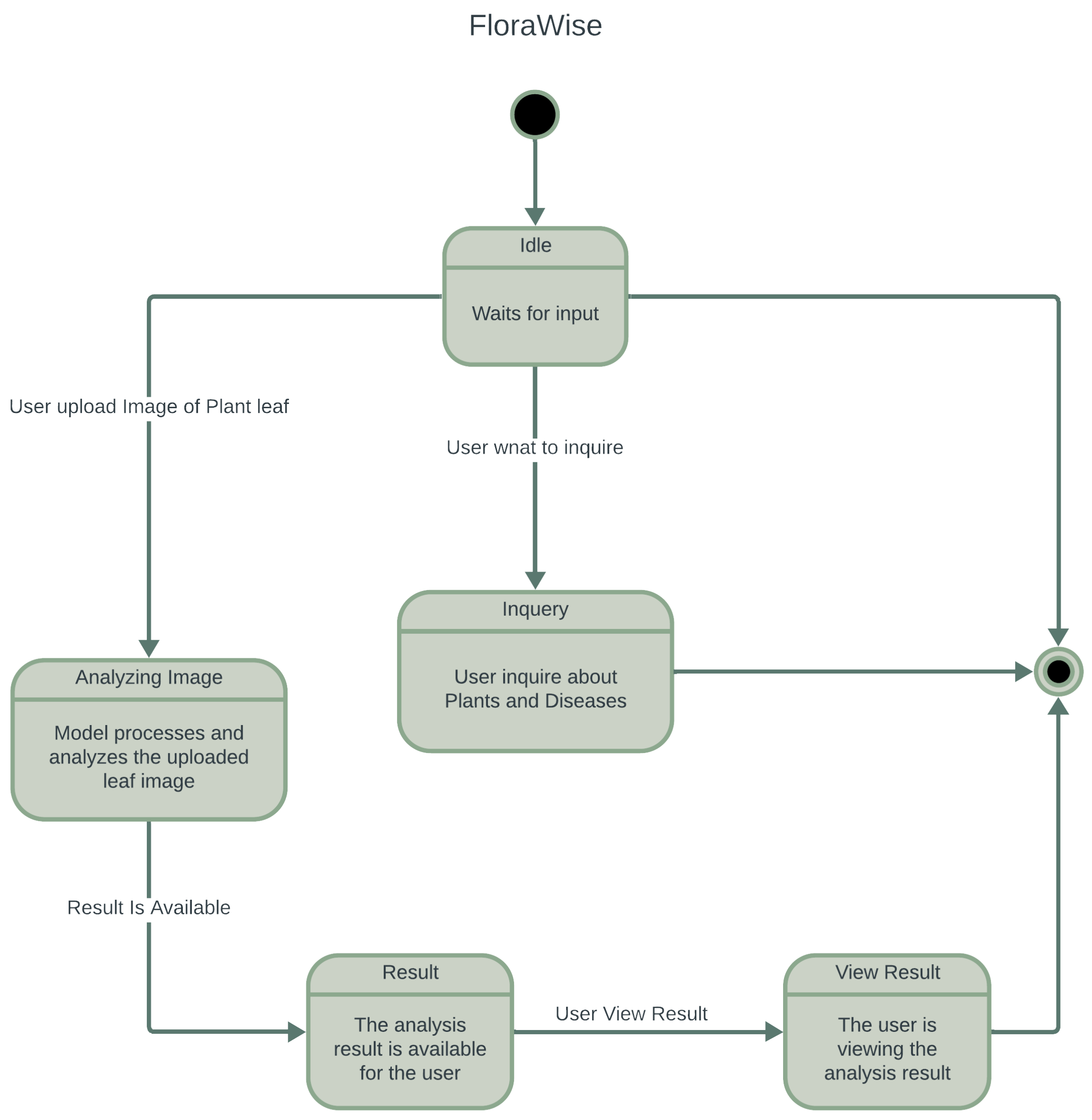
FloraWise
Idle
Waits for input
User upload Image of Plant leaf
User wnat to inquire
Inquery
User inquire about Plants and Diseases
Analyzing Image
Model processes and analyzes the uploaded leaf image
Result Is Available
Result
The analysis result is available for the user
User View Result
View Result
The user is viewing the analysis result

Figure 4.6: State Chart Diagram

5 | Testing Strategies

5.1 | Test Planning and Organization

Define Test Objectives: Establish clear objectives for the testing phase. For FloraWise, the objectives include validating the accuracy of plant recognition and disease detection, ensuring system scalability and confirming seamless user interactions in the Streamlit deployment.

Test Scope: Determine the extent and boundaries of testing. For FloraWise, this includes unit testing, integration testing, performance testing and security testing.

5.2 | Unit Testing Strategy

Identify Test Cases: Create test cases to cover all critical functionalities within the project. Each test case should have a clear description, expected outcome and input data.

Testing Framework: Use Pytest to conduct unit tests on individual components of the project. This ensures that each function and module works as intended in isolation.

Automated Testing: Implement automated unit tests to streamline the testing process and ensure consistency in test execution.

5.3 | Integration Testing Strategy

Identify Integration Points: Determine where different components interact within FloraWise. This includes the deep learning model, data preprocessing and Streamlit deployment.

Testing Framework: Use Selenium to automate integration tests, simulating user interactions with the Streamlit interface. Additionally, employ Postman to test API endpoints and communication between components.

Regression Testing: Establish a regression testing framework to ensure that changes in the codebase do not introduce new defects.

5.4 | Performance Testing Strategy

Load and Stress Testing: Use JMeter to simulate high loads and test FloraWise's response under stress. This helps assess the system's scalability and robustness.

Concurrent User Testing: Employ Locust to simulate multiple concurrent users, ensuring the system can handle the expected volume of traffic.

Resource Monitoring: Monitor resource usage (CPU, memory, network) during performance testing to identify potential bottlenecks.

5.5 | Security Testing Strategy

Data Security: Ensure data is encrypted and transmitted securely. Validate secure communication protocols (e.g., HTTPS).

Access Controls: Test access control mechanisms to ensure that unauthorized users cannot access sensitive data.

Vulnerability Scanning: Use security tools to scan for common vulnerabilities and weaknesses in the system.

5.6 | Continuous Integration and Deployment (CI/CD) Strategy

Automate Testing: Use JMeter to automate the execution of unit and integration tests with each code change. This ensures continuous validation throughout the development process.

Deployment Automation: Automate deployment to Streamlit, ensuring that new builds are deployed and tested quickly.