**COMSATS University Islamabad,   
Abbottabad Campus**

**SOFTWARE DESIGN DESCRIPTION   
(SDD DOCUMENT)**

**for**

**PLANT DISEASE DIAGNOSE SYSYTEM**  
Version 1.0

***By***

**ASAD KHAN FA21-BSE-081**

**FURQAN AHMAD FA21-BSE-089**

**ABDUL MUHAIMIN FA21-BSE-061**

***Supervisor*Sir Javed Raza**

***Bachelor of Science in Computer Science (20xx-20xx)***

**Table of Contents**

Contents

[**Introduction** 5](#_Toc195360221)

[**Design methodology and software process model** 5](#_Toc195360222)

[**System overview** 5](#_Toc195360223)

[**Architectural design** 6](#_Toc195360224)

[**Process flow/Representation** 7](#_Toc195360225)

[**Design models [along with descriptions]** 8](#_Toc195360226)

[Class Diagram 8](#_Toc195360227)

[Sequence Diagram 9](#_Toc195360228)

[State Transition Diagram 10](#_Toc195360229)

[Data Flow Diagram 10](#_Toc195360230)

[**Data design** 12](#_Toc195360231)

[**Data dictionary** 13](#_Toc195360232)

[**Human interface design** 14](#_Toc195360233)

[1. User Registration and Login 14](#_Toc195360234)

[2. Dashboard 14](#_Toc195360235)

[3. Plant Health Diagnosis 14](#_Toc195360236)

[4. Notifications and Alerts 14](#_Toc195360237)

[5. Settings 15](#_Toc195360238)

[6. Historical Data 15](#_Toc195360239)

[7. Support and Suggestions 15](#_Toc195360240)

[8. Admin Features 15](#_Toc195360241)

[9. Feedback for Errors and System Status 15](#_Toc195360242)

[10. Log Out 16](#_Toc195360243)

[**Screen images** 17](#_Toc195360244)

[**Appendix I** 21](#_Toc195360245)

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason for changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

**Application Evaluation History**

|  |  |
| --- | --- |
| **Comments (by committee)**  **\*include the ones given at scope time both in doc and presentation** | **Action Taken** |
|  |  |
|  |  |

**Supervised by**

**<Supervisor’s Name>**

Signature\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Introduction**

The Plant Diagnose System aims to assist farmers by predicting plant diseases using machine learning models and weather data integration. The system offers features such as disease detection, treatment suggestions, historical tracking, and real-time alerts.

Modules include:

* **User Management**: Authentication and role-based access.
* **Plant Disease Prediction**: Image analysis using machine learning.
* **Plant Monitoring**: Track plant health history and generate reports.
* **Notifications**: Alerts for critical plant health updates.
* **Community Forum**: Interaction and advice-sharing platform.

**Design methodology and software process model**

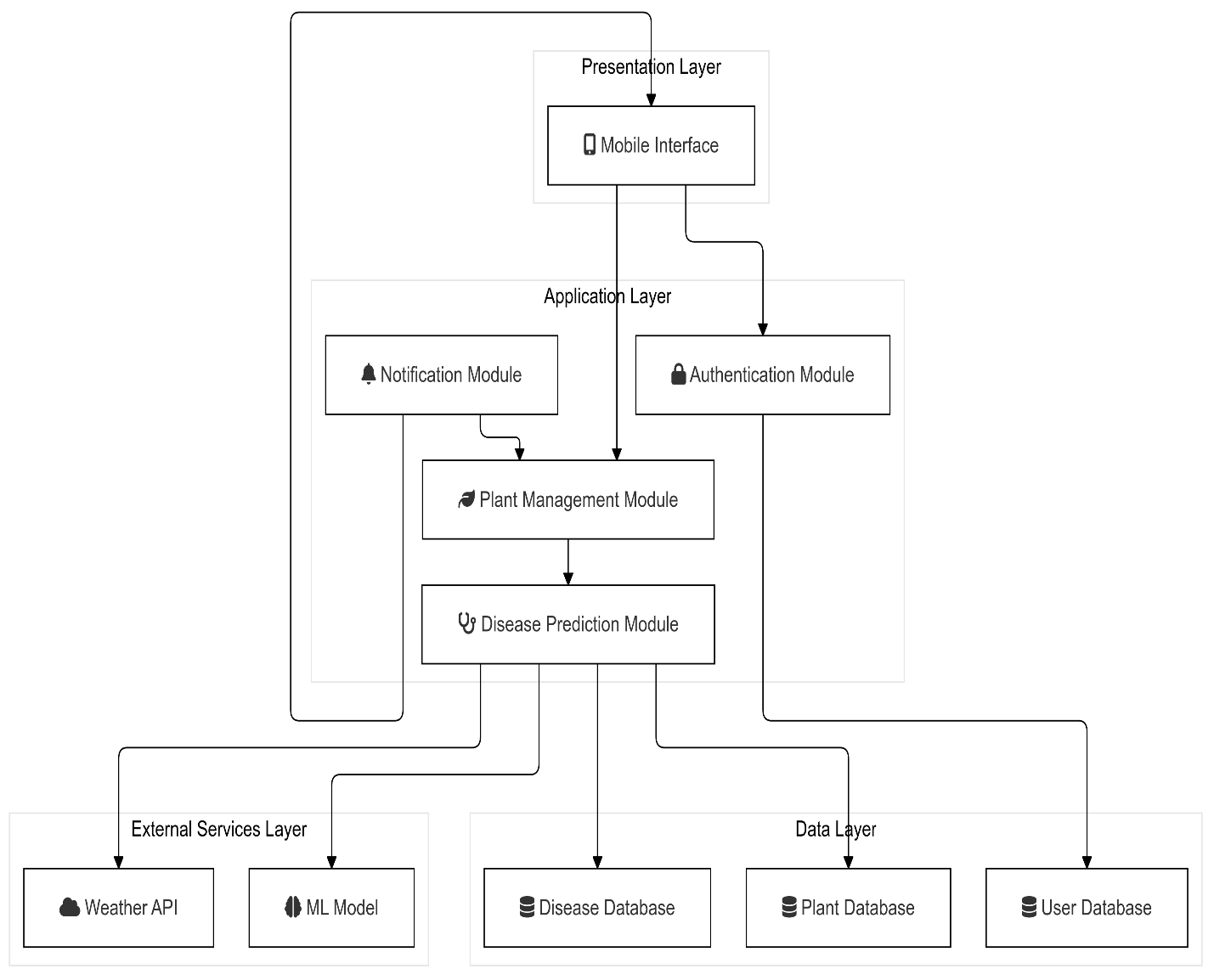
**Design Methodology**: Object-Oriented Design (OOD) is chosen for its modular and scalable approach, making the system easy to maintain and extend.

**Software Process Model**: Agile Process Model is selected to enable iterative development, continuous feedback, and adaptability to changing requirements. This aligns well with the evolving needs of the project and the stakeholders.

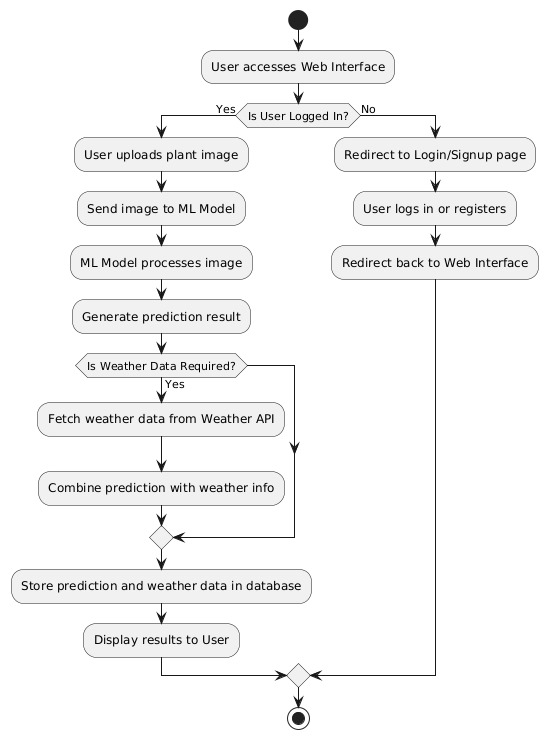
**System overview**

The Plant Diagnose System is a standalone mobile-base application that predicts plant diseases using machine learning. The system integrates weather data for enhanced accuracy and provides actionable insights for users.

**Architectural design**

****

**Process flow/Representation**



**Design models [along with descriptions]**

**The applicable models may include:**

## Class Diagram

A diagram of a computer flowchart

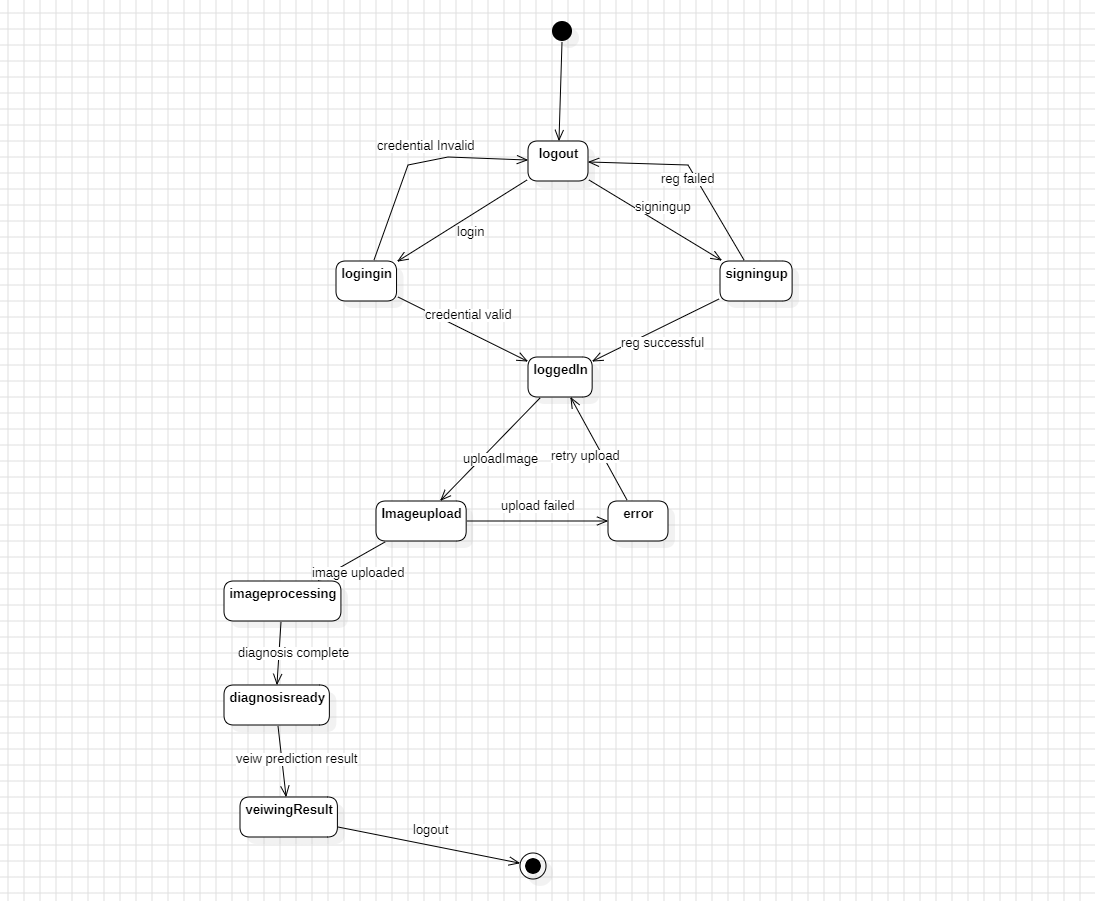
Description automatically generated

## Sequence Diagram

A close-up of a document

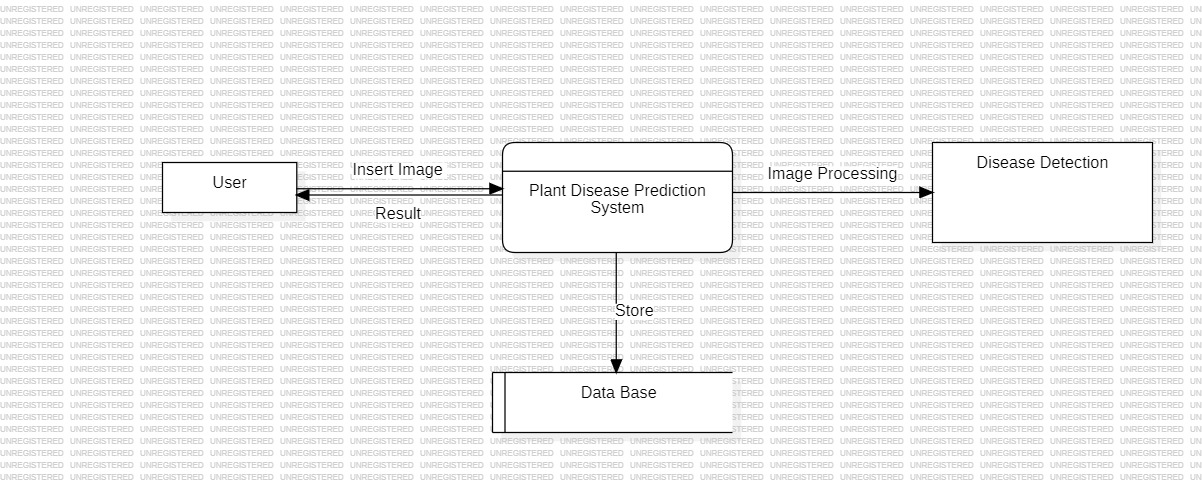
Description automatically generated

## State Transition Diagram



## Data Flow Diagram

**DFD Level 0**

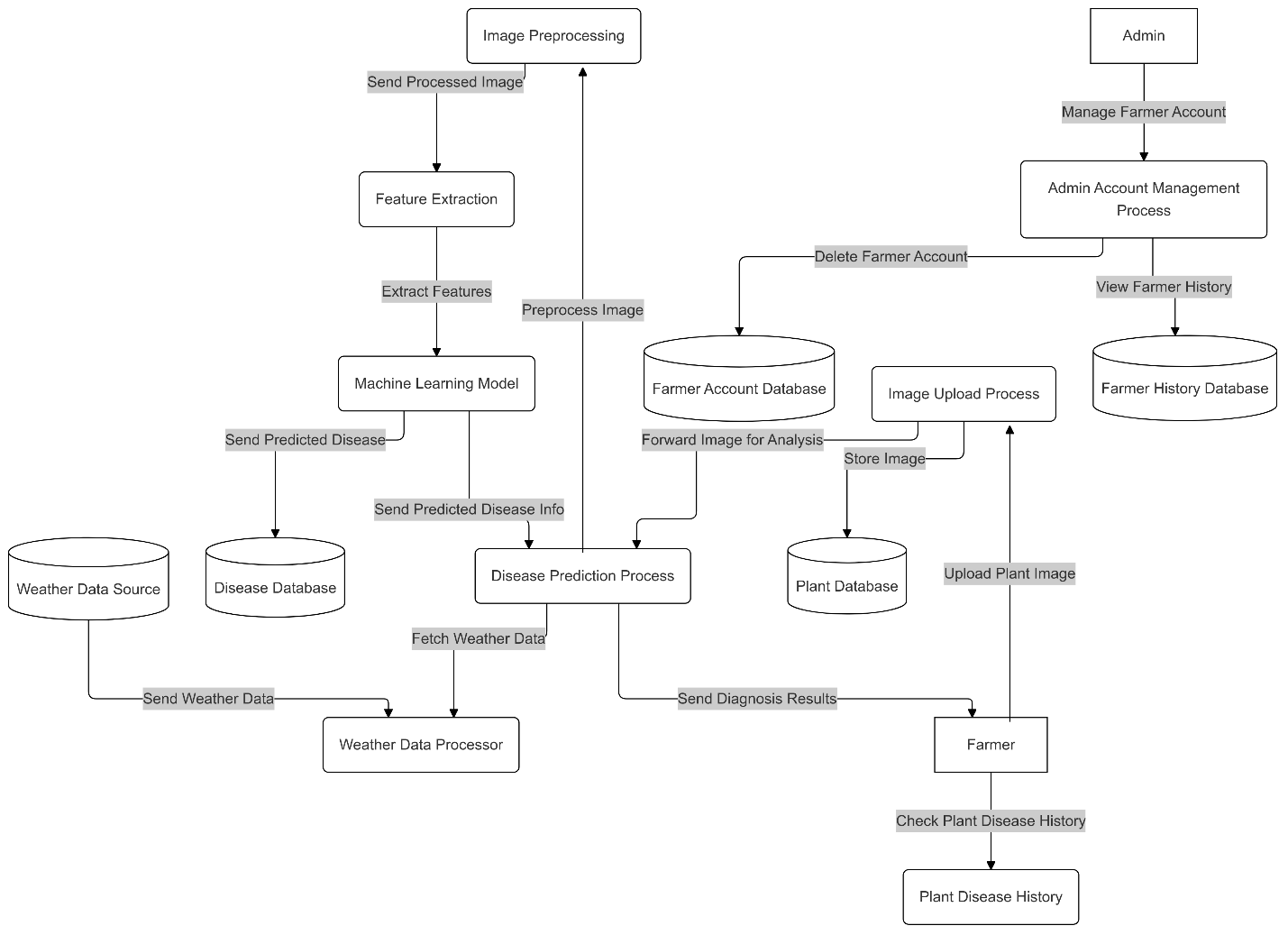


**DFD Level 1**

A diagram of a diagram

Description automatically generated

**DFD Level 3**

****

**Data design**

**Input Data**:

* Users provide inputs such as images, credentials, and plant data through the interface.
* Images are processed and linked to specific plant records.

**Processing Data**:

* Uploaded images are analysed using a machine learning model.
* The system cross-references result with disease information in the database.

**Output Data**:

* Results, treatment recommendations, and historical trends are retrieved from the database and displayed on the dashboard.

**Storage Mechanism**:

* An **SQL database** organizes and stores data entities, ensuring quick retrieval and scalability.
* Data tables are normalized to avoid redundancy and maintain integrity.

**Data dictionary**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| User ID | INT | Unique identifier for each user. |
| Username | |  | | --- | |  |   VARCHAR | User's login name. |
| Password | VARCHAR | Encrypted user password. |
| Email | VARCHAR | User's email address for communication and recovery. |
| Plant ID | INT | Unique identifier for each plant. |
| Plant name | VARCHAR | Name of the plant being monitored. |
| Plant type | VARCHAR | Type of plant (e.g., vegetable, fruit, ornamental). |
| Image path | VARCHAR | Path to the uploaded plant image stored in the system. |
| Disease name | VARCHAR   |  | | --- | |  | | Name of the plant disease. |
| Symptoms | TEXT   |  | | --- | |  | | Description of symptoms associated with the disease. |
| Causes | TEXT   |  | | --- | |  | | Known causes of the disease. |
| Treatment | TEXT | Recommended treatments for the disease. |
| Weather data | JSON | Weather conditions (temperature, humidity, etc.) retrieved from API. |

**Human interface design**

## 1. User Registration and Login

* **Functionality:**
  + A user will be able to register using their email, username, and password through the "Sign Up" page.
  + Registered users will log in using their credentials.
* **Feedback:**
  + Success messages for successful registration and login.
  + Error messages if credentials are invalid or if an account already exists.

## 2. Dashboard

* **Functionality:**
  + After logging in, the user will access a personalized dashboard.
  + Farmers will see options to upload images of plants, view plant health status, and monitor historical data.
  + Administrators will have access to manage users, view their activities, and monitor system metrics.
* **Feedback:**
  + Clear display of recent activities, notifications, and a summary of uploaded plant diagnostics.

## 3. Plant Health Diagnosis

* **Functionality:**
  + Farmers can upload images of plants suspected to have diseases.
  + The system will process the image and provide a diagnosis, including:
    - Detected disease (if any).
    - Severity of the disease.
    - Suggested treatments or preventive measures.
  + Integration of weather data to provide additional insights, like disease probability based on weather conditions.
* **Feedback:**
  + Display of diagnostic results with appropriate visuals (e.g., highlighted diseased areas).
  + Success or error messages if an image upload fails.

## 4. Notifications and Alerts

* **Functionality:**
  + Farmers will receive notifications about:
    - Plant health status changes.
    - Seasonal disease risks.
    - Personalized care tips for their plants.
* **Feedback:**
  + Pop-up or banner notifications with relevant details.
  + Alerts for critical conditions requiring immediate attention.

## 5. Settings

* **Functionality:**
  + Users can modify settings such as language, notification preferences, and access permissions.
  + Options to clear cache or autosave photos to an album.
* **Feedback:**
  + Confirmation prompts for actions like clearing cache or enabling/disabling features.

## 6. Historical Data

* **Functionality:**
  + Farmers can view a history of uploaded images, past diagnoses, and suggested treatments.
  + Data is organized by date and plant type for easy reference.
* **Feedback:**
  + Comprehensive logs displayed in a table or timeline format with filters for easy navigation.

## 7. Support and Suggestions

* **Functionality:**
  + Users can access help documentation, provide feedback, or suggest new features directly through the app.
  + Options like "Encourage Us," "Suggestion," and "How to Take a Picture" will guide users.
* **Feedback:**
  + Success messages after submitting feedback or suggestions.

## 8. Admin Features

* **Functionality:**
  + Administrators can:
    - Approve or delete farmer accounts.
    - Monitor user activities such as login history and uploaded images.
    - View system statistics like user count and performance metrics.
* **Feedback:**
  + Clear tables and visual summaries of user activities and system performance.
  + Confirmation messages for actions like account deletion.

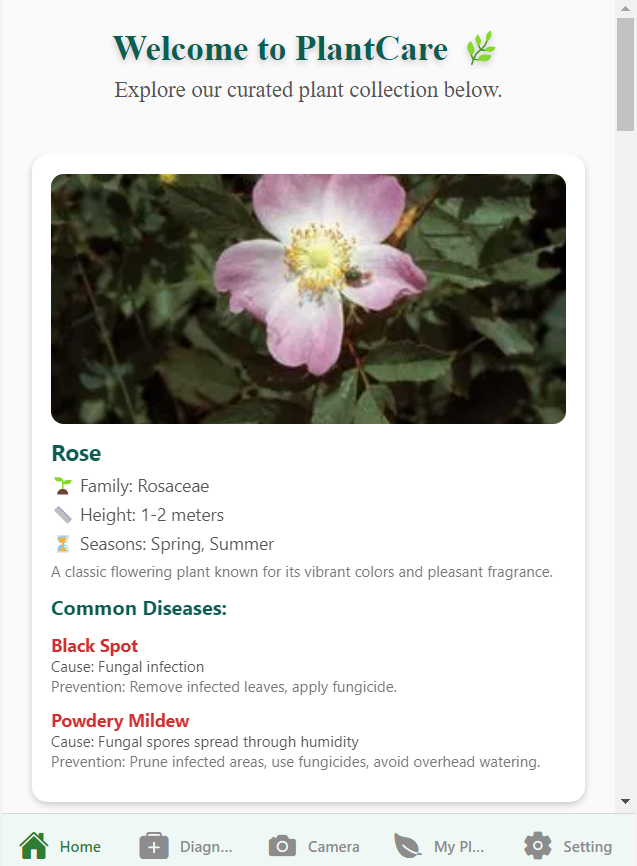
## 9. Feedback for Errors and System Status

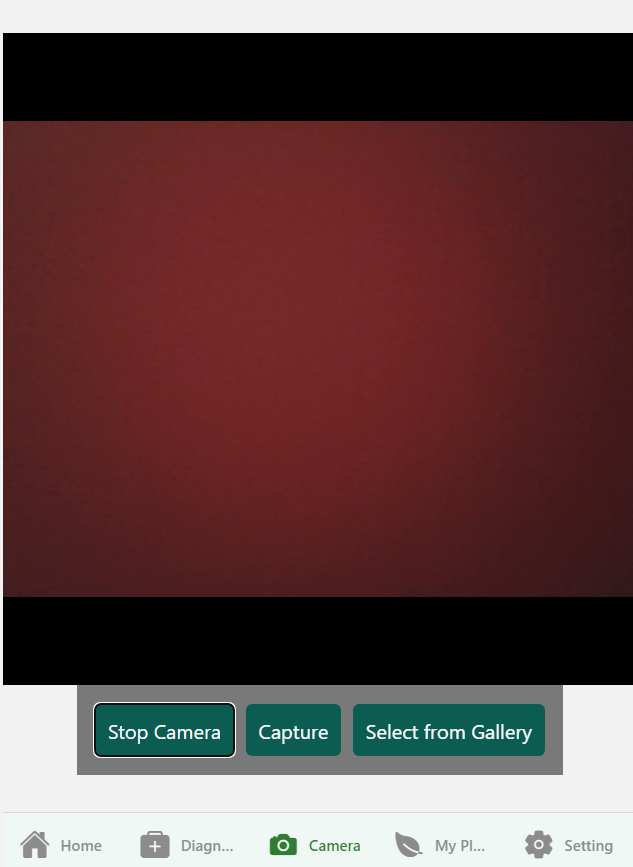
* **Functionality:**
  + The system will notify users in case of errors such as upload failures, invalid inputs, or server downtime.
* **Feedback:**
  + Descriptive error messages with suggested resolutions.
  + Indicators (like loading spinners or progress bars) during processing tasks.

## 10. Log Out

* **Functionality:**
  + Users can log out securely from the application.
* **Feedback:**
  + Confirmation prompts before logging out and success messages upon logging out.

**Screen images**





A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Appendix I**

* [UML Artifacts](http://agilemodeling.com/artifacts/)
* [Data Flow Diagrams](http://www.agilemodeling.com/artifacts/dataFlowDiagram.htm)
* [Software Engineering by Roger Pressman](https://example.com)
* Lucid Chart