**CSE 499B (Section 04)**

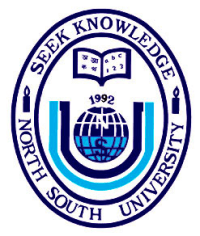
**Moder Tool Usage Report (CO3)**

**Project Title:** Plant Diseases Detection Using Image Processing

**Submitted To**

**Dr. Shazzad Hosain (SZZ)**

**Date: 04/05/2025**



**Group 3**

**Group Members**

|  |  |
| --- | --- |
| **ID** | **Name** |
| 2011892042 | Akib Hasan |
| 2011566042 | Tania Akter Lima |
| 2012959042 | Naima Homaira Khan |
| 2012464042 | Abroor Zahin Niloy |

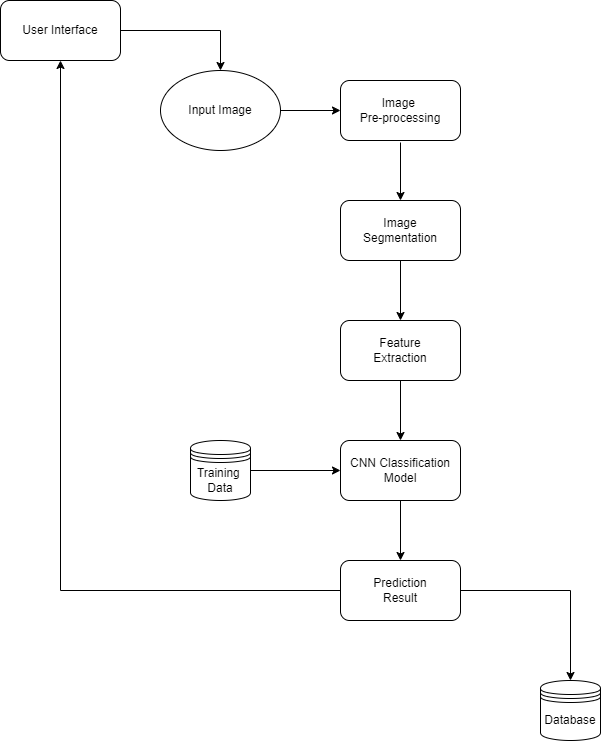
**1. Design Tools Used**

During the design phase of our project, we utilized online diagramming tools to effectively visualize and plan the system’s architecture, workflows, and interactions. The tools used for this purpose were:

* Draw.io
* Lucidchart

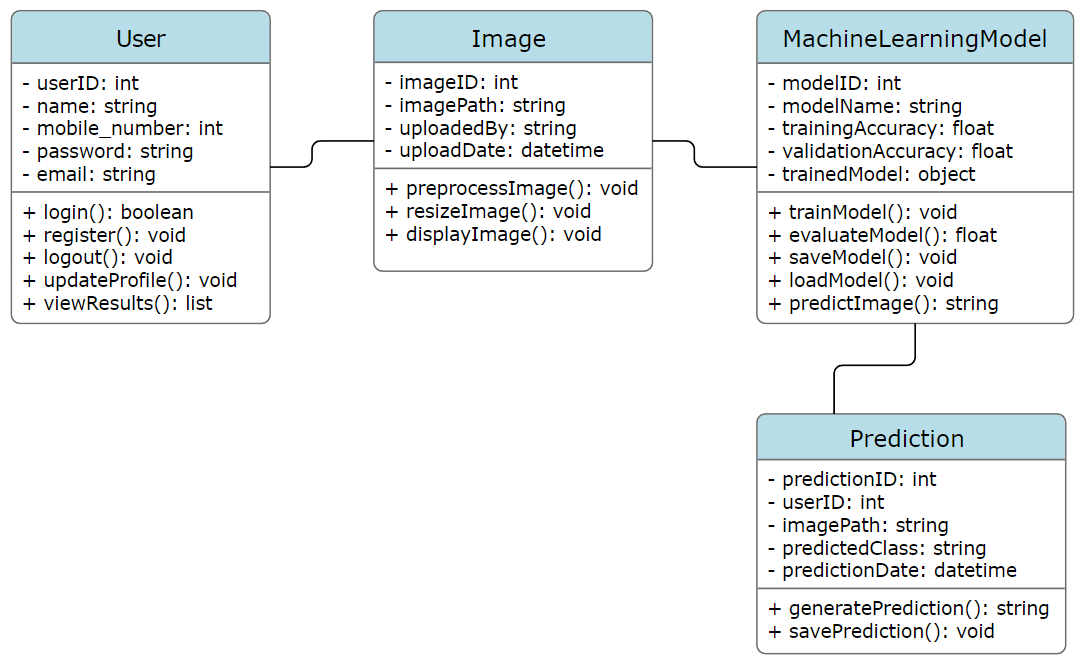
These tools allowed us to create a range of diagrams that supported system planning and helped align our team’s understanding. The following types of diagrams were developed:

**System Architecture & Workflow Diagram** – This combined diagram illustrated both the overall architecture and the logical flow of operations in the system. It mapped how the User Interface (Android app, web interface), machine learning backend (CNN model built with TensorFlow and Keras), and storage components interact. It also captured the step-by-step flow of user actions, image processing, prediction generation, and result display, providing a clear picture of the end-to-end process.



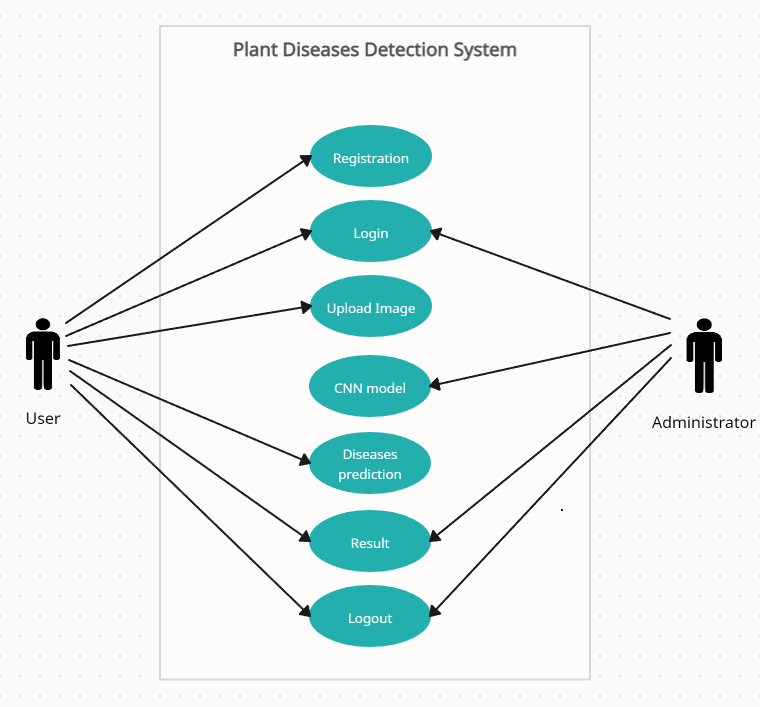
*Figure: System Design*

**Class Diagram** – Class diagrams were created to represent the object-oriented structure of both the mobile and web applications. These diagrams included the main classes, attributes, and methods, helping organize the project’s codebase and improve maintainability.

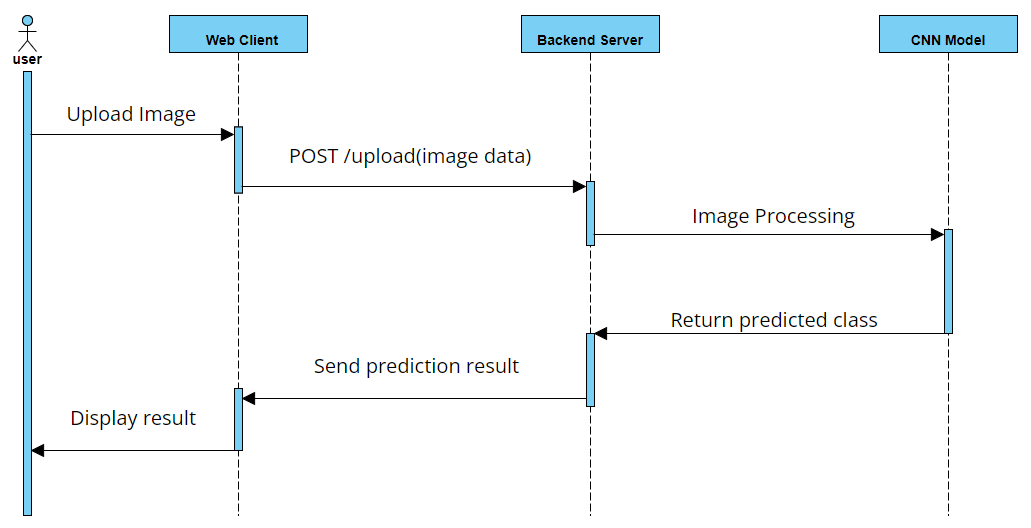


*Figure: Class Diagram*

**Use Case and Sequence Diagrams** – Use case diagrams highlighted different user roles and their interactions with system features. Sequence diagrams showed the order of communication between system components during key operations like disease detection and result display.

****

*Figure: Use Case Diagram*

****

*Figure: Sequence Diagram*

These diagrams served as essential planning tools that contributed to a smoother implementation phase, better communication within the team, and a clearer overall understanding of the project’s architecture and logic.

**2. Coding and Development Tools**

The development of our system involved multiple platforms and tools, each tailored to specific components of the project. These tools supported machine learning model development, web application design, and mobile app creation.

**Programming Languages & Libraries:**

* **Python** – Used for developing the machine learning model and implementing the backend logic of the web application.
* **Kotlin** – Used for building the Android native mobile application.
* **TensorFlow and Keras** – Used for developing, training, and deploying the Convolutional Neural Network (CNN) model for plant disease classification.

**Development Environments:**

* **Jupyter Notebook** – Used for building, training, testing, and evaluating the CNN model. It allowed easy visualization of training data and metrics.
* **Visual Studio Code** – Used for coding Python scripts related to backend logic and integration with the web interface.
* **Android Studio** – Used for designing and developing the Android mobile application using Kotlin and native Android components.

**Frameworks:**

* **Streamlit** – Used to develop a lightweight, interactive web interface. It allows users to upload plant images and receive real-time predictions from the trained model.

**Version Control:**

* **GitHub** – Used to manage the project’s source code, track changes, and collaborate across the team efficiently.

These tools enabled smooth development across different layers of the system and facilitated teamwork, modular development, and integration between the machine learning model and user applications.

**3. Test / Validation Tools Used**

Testing and validation played a critical role in ensuring the reliability and accuracy of our project. We used a combination of performance evaluation metrics for the machine learning model and manual testing for the user interfaces of both web and mobile applications. The tools and techniques used are as follows:

**Accuracy Metric, F1 Score, and Confusion Matrix** – To evaluate the performance of the Convolutional Neural Network (CNN) model, we used standard classification metrics. Accuracy measured the overall correctness of the predictions, while the F1 score provided a balanced evaluation of precision and recall, especially useful in handling class imbalance. The confusion matrix was instrumental in visualizing the model’s performance across different disease categories, helping us understand the number of correct and incorrect classifications for each class.

**Manual Testing (Web and Mobile Applications)** – For both the web application (built using Streamlit) and the Android mobile application (developed using Kotlin), we conducted manual testing to verify functionality and usability. This included checking the image upload and capture features, model prediction display, responsiveness across devices, and handling of unexpected inputs. The team members simulated real user interactions to identify and fix bugs and ensure smooth operation.

These validation approaches helped us confirm that the model performed effectively in identifying plant diseases and that the applications reliably delivered those results to the users. Regular testing during development allowed us to catch issues early, refine the system, and ensure an overall user-friendly and functional solution.