**PLANT SAGE**

**CAPSTONE PROJECT PHASE-1**

# Phase – I Report

## *Submitted by-*

|  |  |
| --- | --- |
| Aditya Mahaur | 21BCE11653 |
| Naman Roy | 21BCE11639 |
| Ayush Thakur | 21BCE11277 |
| Anmol Jain | 21BCE11416 |
| Anushka Gupta | 21BCE11309 |
|  |  |

***in partial fulfillment of the requirements for the degree of***

## *Bachelor of Engineering and Technology*



**VIT Bhopal University**

**Bhopal**

**Madhya Pradesh September, 2024**



# Bonafide Certificate

|  |  |
| --- | --- |
| Certified that this project report titled **“PLANT SAGE”** is the bonafide | |
| work of |  |
| “Aditya Mahaur | 21BCE11653, |
| Naman Roy | 21BCE11639, |
| Ayush Thakur | 21BCE11277, |
| Anmol Jain | 21BCE11416, |
| Anushka Gupta | 21BCE11309” who carried out the project work under |

my supervision. This project report of Phase-I is submitted for the DSN4095 Capstone Project ‘Review 1’ scheduled between 23.09.2024 and 29.09.2024.

**Mr. VIJAY KUMAR TRIVEDI**

**Supervisor**

**Mr. VISHAL SINGH**

**Reviewer 1**

**Mrs. PAVITHRA**

**Reviewer 2**



# Declaration of Originality

We hereby declare that this report entitled “PLANT SAGE” represents our original work carried out for the DSN Capstone project as a student of VIT Bhopal University and, to the of our knowledge, it contains no material previously published or written by another person, nor any material presented for the award of any other degree or diploma of VIT Bhopal University or any other institution. Works of the other authors cited in this report have been duly acknowledged under the section” References”.

# Acknowledgement

We would like to extend our heartfelt gratitude to our project supervisor, Mr. Vijay Kumar Trivedi, for his invaluable guidance, support, and encouragement throughout the course of this project. His expertise, insightful feedback, and unwavering commitment have been instrumental in shaping the direction of our work and helping us overcome numerous challenges along the way.

We also wish to express our sincere thanks to VIT Bhopal University for providing the necessary resources and fostering an environment that facilitated the successful completion of this project. Our appreciation also goes out to our reviewers, whose contributions and collaboration added valuable perspectives and enriched the quality of the project.

Furthermore, we are deeply grateful to our family and friends for their understanding, encouragement, and patience during this journey. Their support has been a constant source of motivation and inspiration.

Lastly, we would like to thank everyone who contributed, directly or indirectly, to the success of this project—whether through feedback, discussions, or encouragement. Your contributions have been instrumental in shaping the final outcome of this work.

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Topic** | **Page No.** |
| 1 | Introduction | 6 |
| 1.1 | Motivation | 6 |
| 1.2 | Objective | 6 |
| 2 | Existing Work | 7 |
| 3 | Technology Used | 8 |
| 4 | Methodology | 9 |
| 4.1 | Architecture Overview | 9 |
| 4.2 | System Components | 9 |
| 4.3 | Working Principle | 10 |
| 5 | Individual Contribution | 12 |
| 6 | Conclusion | 14 |
| 7 | References | 15 |

**Index**

## CHAPTER 1: INTRODUCTION

*Plant Sage* enables users like whether farmers, gardeners, or plant enthusiasts to quickly and accurately diagnose plant diseases. By simply capturing an image of the affected plant, users receive instant feedback on the disease, along with information on how to treat or prevent its spread. The machine learning models behind *Plant Sage* are trained on a vast dataset of plant diseases, ensuring high accuracy across various plant species.

The application is built using Flutter, allowing for a seamless and intuitive user experience across both Android and iOS platforms. Its cross-platform compatibility ensures that users can access the app regardless of their device, making it accessible to a wide audience.

## 1.1 Motivation

Plant diseases pose a significant threat to agriculture and home gardening, often resulting in reduced crop yields and compromised plant health. Timely identification of these diseases is crucial to minimize damage and take preventive measures.

Plant diseases not only affect crop yields but also threaten food security and the livelihoods of farmers. In many cases, delayed detection and lack of immediate action can lead to significant damage, resulting in economic loss and environmental challenges. In response to this challenge, *Plant Sage* has been developed as a user-friendly mobile application that utilizes machine learning to detect plant diseases from images of plant leaves.

## 1.2 Objective

*Plant Sage* aims to reduce the impact of plant diseases and contribute to more sustainable plant care practices. The objective of our project is to empower users to take swift and informed actions when dealing with plant diseases, thereby minimizing potential damage and promoting healthier plants. Early detection is key in managing plant diseases, as the sooner a disease is identified, the more effectively it can be treated or contained.

The user-friendly design of *Plant Sage*, developed using Flutter, ensures that the application is accessible to a wide range of users, from farmers to home gardeners. By simplifying the process of disease detection and offering timely solutions. With *Plant Sage*, users can capture an image of a plant showing signs of distress and receive an instant diagnosis, allowing them to respond quickly to the issue.

**CHAPTER 2: EXISTING WORK**

Machine learning models like convolutional neural networks (CNN) and deep learning techniques were employed to detect plant diseases using a dataset of 87,848 leaf images captured in both laboratory and real cultivation conditions. The CNN architectures used include AlexNet, AlexNetOWTBn, GoogLeNet, VGG, and OverFeat, with models trained using Torch7 and LuaJIT.

Three different approaches were explored:

1. **Approach 1:** The dataset was split into 80% training and 20% testing sets, ensuring a balanced distribution of laboratory and field images.
2. **Approach 2:** Images were preprocessed and cropped to 256x256 pixels while maintaining the same training/testing ratio.
3. **Approach 3:** Images were directly sent to a LLM like ChatGPT vision using a prompt to analyze and display the disease

The VGG model achieved the highest performance, with a 99.53% success rate in classifying previously unseen plant images. Models trained on original plant leaves outperformed those using processed images, with the VGG and AlexNetOWTBn architectures demonstrating the best results.

An approach of using deep learning method was explored in order to automatically classify and detect plant diseases from leaf images. The developed model was able to detect leaf presence and distinguish between healthy leaves and 13 different diseases, which can be visually diagnosed. The complete procedure was described, respectively, from collecting the images used for training and validation to image preprocessing and augmentation and finally the procedure of training the deep CNN and fine-tuning. Different tests were performed in order to check the performance of newly created model.

During the research we found that either the models were based only on CNN or only with an API call to a paid Large Language Model. We could not find a suitable open-source contribution in this field which also offers a prescriptive analysis, all the applications just detected the disease.

**CHAPTER 3: TECHNOLOGIES USED**

The *Plant Sage* project combines advanced technologies and deep learning models to develop an effective solution for plant disease detection. Key technologies and machine learning models employed in this project are:

1. **Flutter Framework:**

*Plant Sage* is developed using Flutter, a powerful open-source UI toolkit for building natively compiled applications from a single codebase. Flutter's cross-platform compatibility ensures that the app runs seamlessly on both Android and iOS devices. This makes the app user-friendly and accessible to a wide audience, including farmers and gardeners with varying technical expertise.

1. **Deep Learning Models:**

To classify plant diseases from leaf images, *Plant Sage* leverages deep learning techniques, specifically Convolutional Neural Networks (CNNs). CNNs are highly effective for image classification tasks due to their ability to automatically learn spatial hierarchies of features from input images.

1. **Computer Vision:**

Computer vision plays a pivotal role in the *Plant Sage* project by enabling the automated detection and classification of plant diseases through image analysis. The integration of computer vision techniques allows the application to process and interpret visual data from leaf images captured by users.

1. **Firebase:**

Integrating Firebase into the *Plant Sage* project enhances the app's functionality by providing real-time data synchronization, secure user authentication, scalable image storage, and robust analytics. By leveraging Firebase's various services, *Plant Sage* can deliver a more effective and user-friendly experience, ultimately empowering users to take timely actions for managing plant health and combating diseases. Firebase’s cloud-based architecture supports scalability and efficiency, making it an ideal choice for modern mobile applications in agriculture.

1. **Large Language Model:**

In order to enhance the capabilities of this application we are working with open-source large language models like Llama by meta and BERT by google, which will be given a prompt after detection of disease by the CNN model and in return a short prescriptive analysis of prevention of disease and future measures will be displayed to the user. These models work on an API call which is free of cost, hence reducing the computation and production cost of the application.

## CHAPTER 4: METHODOLOGY

The system design and architecture of the *Plant Sage* application are fundamental to its functionality, scalability, and performance. This section outlines the overall architecture, components, and working principles of the system.

* 1. **Architecture Overview**

The architecture of *Plant Sage* can be described using a layered approach that separates the concerns of the client-side (Flutter app) and the server-side (backend). The architecture includes:

* + Client Layer: This consists of the Flutter mobile application that interacts with users, allowing them to capture images, view results, and receive notifications.
  + Backend Layer: This layer includes the server that handles requests from the client, process images, manages data storage, and communicates with machine learning models and large language models.
  + Database Layer: This layer consists of the database services (Firebase or SQL) that store user data, images, and plant disease information.
  + Machine Learning Layer: This includes the trained machine learning models that analyze leaf images to detect diseases and LLMs to give prescriptive analysis.

**4.2 System Components**

1. **Flutter Mobile App:**

* User Interface (UI): Provides a user-friendly interface for capturing images, displaying results, and showing notifications.
* Image Capture Module: Uses the device camera to capture images of plant leaves.
* API Integration: Communicates with the backend using RESTful or GraphQL APIs for data exchange.

1. **Backend Server:**

* API Server: Handles incoming requests from the Flutter app, processes them, and returns responses. Built using Node.js, Python (Flask/FastAPI), or Firebase Functions.
* Authentication Module: Manages user registration, login, and authentication tokens.
* Image Processing Module: Receives images, preprocesses them, and sends them to the machine learning model for analysis.

1. **Database:**

* Firebase Realtime Database or Firestore: Stores user profiles, disease information, and historical records of analyzed images.
* Cloud Storage: Used to store images uploaded by users.

1. **Machine Learning Models:**

* CNN Models: Suitable Convolutional neural network architecture (e.g., MobileNetV2) will be deployed to analyze leaf images and detect diseases. These models can be hosted on cloud platforms or integrated with Firebase ML.

1. **Large Language Model:**

* A result-oriented prompt will be designed and given to the LLM with the disease detection result received from the machine learning model.
* The fine tuned LLM will asses the prompt and supply the application with a prescriptive analysis of the disease.
  1. **Working Principle**

The working principle of the *Plant Sage* application can be broken down into the following steps:

1. **User Interaction:**

* The user opens the *Plant Sage* app and navigates to the image capture interface.
* They take a picture of a plant leaf using the app's camera functionality.

1. **Image Upload:**

* The captured image is sent to the backend server via an API call.
* The image is temporarily stored in cloud storage (e.g., Firebase Cloud Storage).

1. **Image Processing:**

* The backend processes the image by applying preprocessing steps, such as resizing, normalization, and augmentation, to prepare it for the machine learning model.

1. **Disease Detection:**

* The preprocessed image is passed to the trained CNN model for analysis.
* The model evaluates the image and classifies it as healthy or identifies specific diseases, providing a confidence score for each prediction.

1. **Results Returned:**

* The backend sends the results back to the Flutter app, including the classification results and any recommended actions as a result from the large language model based on the detected diseases.

1. **User Feedback and Recommendations:**

* The app displays the results to the user, providing information about the detected disease and recommended treatments or preventive measures.
* Users can save their history of analyzed images and receive notifications about important updates.

1. **Data Storage and Analytics:**

* User data, including images and analysis results, are stored in the database for future reference and model training.
* Analytics tools track user interactions and app performance for ongoing improvement.

1. **Continuous Learning:**

* As more users interact with the app and provide feedback, the dataset can be expanded. This data can be used to retrain and fine-tune the machine learning models, improving accuracy and performance over time.

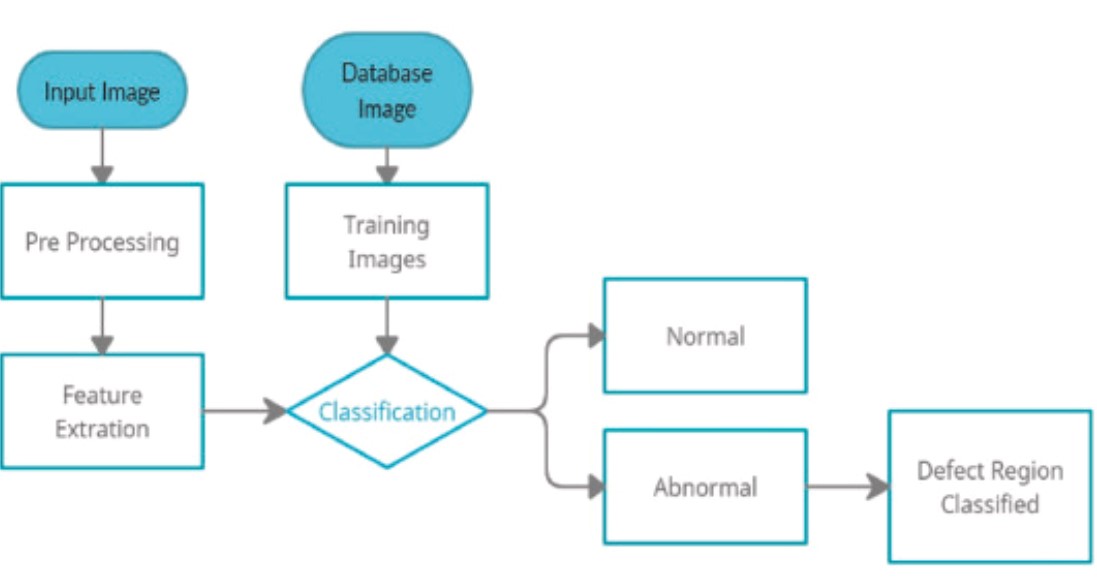


Fig. 1 Working Principle

**CHAPTER 5: INDIVIDUAL CONTRIBUTIONS**

1. **Aditya Mahur**

Aditya played a pivotal role in the project by focusing on the training and experimentation of multiple Large Language Models (LLMs). His responsibilities included:

* **Model Selection**: Aditya researched and evaluated various LLMs to determine which models would be most effective for our application. He assessed their performance in terms of accuracy and efficiency, ensuring they met the specific needs of farmers.
* **Deployment Methods**: He explored different deployment strategies to integrate LLMs into the Flutter application, optimizing the overall user experience. His experimentation led to the selection of methods that balanced performance and usability, crucial for end-users who may not be tech-savvy.
* **Collaborative Input**: Aditya collaborated closely with other team members, providing insights into how the LLM's output could be effectively tailored to meet user prompts, enhancing the clarity and accessibility of the generated prescriptions.

1. **Anushka Gupta**

Anushka made significant contributions through her extensive review of published research papers. Her efforts included:

* **Literature Review**: Anushka meticulously analyzed a wide range of academic papers and industry reports to identify potential models for our application. Her thorough understanding of the latest advancements in agricultural technology informed our model selection process.
* **Identifying Challenges**: She proactively sought out and documented problematic scenarios encountered in similar projects. By understanding these challenges, she helped the team anticipate potential pitfalls and devise effective solutions, ensuring a smoother development process.
* **Model Recommendations**: Based on her research, Anushka provided recommendations on which models to pursue further, facilitating informed decision-making within the team.

1. **Anmol Jain**

Anmol was responsible for the development of the Flutter front end, with key contributions including:

* **User Interface Design**: She designed an intuitive and user-friendly interface tailored to the needs of farmers. Anmol prioritized simplicity and ease of navigation, ensuring that users could easily capture images and access results without technical difficulties.
* **Database Integration**: Anmol will establish a robust connection to Firebase, enabling real-time image capture and secure storage of user data. Her work ensures that the app could handle multiple users simultaneously, which is essential for scalability.
* **Testing and Feedback**: She conducted user testing sessions to gather feedback on the front end, iteratively improving the design based on user experiences to enhance overall usability.

1. **Ayush Thakur**

Ayush focused on the backend integration of the Flutter application, making vital contributions such as:

* **Model Integration**: He will be integrating the Convolutional Neural Network (CNN) and the LLM into the Flutter app. This involved ensuring that data flow between the models and the app was seamless, allowing for real-time processing of user input.
* **Deployment Strategy**: Ayush excels in developing a comprehensive deployment strategy that included server management and monitoring. His approach ensures that the application remained stable and responsive, even under high usage conditions.
* **Collaboration with Frontend**: He has been working closely with Anmol to ensure that the backend processes aligned smoothly with the user interface, enabling a cohesive user experience from image capture to diagnosis and prescription.

1. **Naman Roy**

Naman has a critical role in the development of the CNN model, with contributions that included:

* **Data Curation**: He undertook extensive research to compile a high-quality dataset for training the CNN. This involved sourcing data from multiple reputable websites and ensuring it was well-labeled and relevant for plant disease detection.
* **Model Training and Optimization**: Naman trained the CNN using TensorFlow, employing various techniques to optimize its performance, such as fine-tuning hyperparameters and experimenting with different architectures. His expertise significantly improves the model’s accuracy in identifying plant diseases.
* **Validation and Testing**: He will be establishing a rigorous validation process to assess the CNN's performance, including testing against real-world scenarios to ensure reliability. His commitment to quality assurance ensures that the model produced is accurate and actionable results for farmers.

**CHAPTER 6: CONCLUSION**

The *Plant Sage* application demonstrates the integration of machine learning, generative AI and mobile technology to address the critical issue of plant disease detection. By leveraging advanced convolutional neural networks (CNNs), open-source Large Language Models (LLMs) and a user-friendly Flutter interface, the app enables quick and accurate identification of plant diseases from leaf images. The architecture supports scalability and efficient data management through robust backend services and cloud storage solutions. Through continuous learning and user feedback, *Plant Sage* can improve its performance over time, contributing to sustainable plant care practices. Ultimately, this project empowers users, including farmers and gardening enthusiasts, to make informed decisions, enhancing plant health and productivity.

**CHAPTER 7: REFERENCES**

[https://www.mdpi.com/2076-3417/13/17/9596](https://www.mdpi.com/2076-3417/13/17/9596%20) https://onlinelibrary.wiley.com/doi/10.1155/2016/3289801 https://www.sciencedirect.com/science/article/pii/S2665917422000757 https://www.sciencedirect.com/science/article/abs/pii/S0168169917311742

Ø <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0309468>

Ø <https://link.springer.com/article/10.1007/s10055-018-0338-3>

Ø

[https://books.google.co.in/books?hl=en&lr=&id=05lUDwAAQBAJ&oi=fnd&pg=PP1& dq=google+AR+Core&ots=QR1t5Qlorj&sig=aWXSLYakNAOooW1Iu6TzWfT0-nU&r edir\_esc=y#v=onepage&q&f=false](https://books.google.co.in/books?hl=en&lr=&id=05lUDwAAQBAJ&oi=fnd&pg=PP1&dq=google+AR+Core&ots=QR1t5Qlorj&sig=aWXSLYakNAOooW1Iu6TzWfT0-nU&redir_esc=y#v=onepage&q&f=false)

Ø <https://ieeexplore.ieee.org/document/9119208>

Ø

[https://books.google.co.in/books?hl=en&lr=&id=aO1mDwAAQBAJ&oi=fnd&pg=PP1](https://books.google.co.in/books?hl=en&lr=&id=aO1mDwAAQBAJ&oi=fnd&pg=PP1&dq=google+core+and+unity+&ots=c6pWfd0XWY&sig=rONv02E0Wjrp6Azn4ubAnSu9gmU&redir_esc=y#v=onepage&q&f=false)

[&dq=google+core+and+unity+&ots=c6pWfd0XWY&sig=rONv02E0Wjrp6Azn4ubAn](https://books.google.co.in/books?hl=en&lr=&id=aO1mDwAAQBAJ&oi=fnd&pg=PP1&dq=google+core+and+unity+&ots=c6pWfd0XWY&sig=rONv02E0Wjrp6Azn4ubAnSu9gmU&redir_esc=y#v=onepage&q&f=false)

[Su9gmU&redir\_esc=y#v=onepage&q&f=false](https://books.google.co.in/books?hl=en&lr=&id=aO1mDwAAQBAJ&oi=fnd&pg=PP1&dq=google+core+and+unity+&ots=c6pWfd0XWY&sig=rONv02E0Wjrp6Azn4ubAnSu9gmU&redir_esc=y#v=onepage&q&f=false)