"Plant Disease Detection - Flauralysis" using Deep Learning

A Major Project Report Submitted to

Rajiv Gandhi Proudyogiki Vishwavidyalaya

Towards Partial Fulfillment for the Award of

Bachelor of Engineering in Computer Science Engineering

Submitted by:

Jayant Neema(0827CS201107)
Leena Ghatiya(0827CS201125)
Manas Dhaketa(0827CS201130)
Mandvee Vatsa(0827CS201132)

Guided by:

Prof. Anita Mahajan
Computer Science and Engineering

Acropolis Institute of Technology & Research, Indore

Jul - Dec 2023

EXAMINER APPROVAL

The Major Project entitled ***"Plant Disease Detection - Flauralysis"*** submitted by **Jayant Neema (0827CS201107), Leena Ghatiya (0827CS201125), Manas Dhaketa (0827CS201130), Mandvee Vatsa (0827CS201132)** has been examined and is hereby approved towards partial fulfillment for the award of **Bachelor of Technology degree in Computer Science Engineering** discipline, for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve the project only for the purpose for which it has been submitted.

(Internal Examiner) **(External Examiner)**

Date: **Date:**

RECOMMENDATION

This is to certify that the work embodied in this major project entitled "***Plant Disease Detection - Flauralysis***" submitted by **Jayant Neema (0827CS201107), Leena Ghatiya (0827CS201125), Manas Dhaketa (0827CS201130), Mandvee Vatsa (0827CS201132)** is a satisfactory account of the bonafide work done under the supervision of ***Prof. Anita Mahajan***, is recommended towards partial fulfillment for the award of the Bachelor of Technology (Computer Science Engineering) degree by Rajiv Gandhi Proudyogiki Vishwavidhyalaya, Bhopal

(Project Guide)

(Project Coordinator)

(Dean Academics)

STUDENTS UNDERTAKING

This is to certify that a major project entitled ***"Plant Disease Detection - Flauralysis"*** has been developed by us under the supervision of ***Prof. Anita Mahajan***. The whole responsibility of work done in this project is ours. The sole intention of this work is only for practical learning and research.

We further declare that to the best of our knowledge; this report does not contain any part of any work which has been submitted for the award of any degree either in this University or in any other University / Deemed University without proper citation and if the same work is found then we are liable for explanation to this.

Jayant Neema(0827CS201107)

Leena Ghatiya(0827CS201125)

Manas Dhaketa(0827CS201130)

Mandvee Vatsa(0827CS201132)

Acknowledgement

We thank the almighty Lord for giving us the strength and courage to sail out through the tough and reach on shore safely.

There are a number of people without whom this project's work would not have been feasible. Their high academic standards and personal integrity provided me with continuous guidance and support

We owe a debt of sincere gratitude, deep sense of reverence and respect to our project coordinator **Ms. Shraddha Sharma,** Professor, AITR and our mentor **Ms. Anita Mahajan,** Professor, AITR, Indore for her motivation, sagacious guidance, constant encouragement, vigilant supervision and valuable critical appreciation throughout this project work, which helped us to successfully complete the project on time.

We express profound gratitude and heartfelt thanks to **Dr. Kamal Kumar Sethi**, Professor & Head CSE, AITR Indore for his support, suggestion, and inspiration for carrying out this project. I am very much thankful to other faculty and staff members of IT Dept, AITR Indore for providing me all support, help and advice during the project. We would be failing in our duty if we do not acknowledge the support and guidance received from **Dr. S.C. Sharma**, Director, AITR, Indore whenever needed. We take the opportunity to convey my regards to the management of Acropolis Institute, Indore for extending academic and administrative support and providing us with all necessary facilities for the project to achieve our objectives.

We are grateful to **our parents and family members** who have always loved and supported us unconditionally. To all of them, we want to say **"Thank you"**, for being the best family that one could ever have and without whom none of this would have been possible.

Jayant Neema(0827CS201107)

Leena Ghatiya(0827CS201125)

Manas Dhaketa(0827CS201130)

Mandvee Vatsa(0827CS201132)

Executive Summary

Plant Disease Detection – Flauralysis

This project is submitted to Rajiv Gandhi Proudyogiki Vishwavidhyalaya, Bhopal (MP), India for partial fulfillment of Bachelor of Engineering in Computer Science branch under the sagacious guidance and vigilant supervision of ***Prof. Anita Mahajan.***

The project is based on object detection and analysis , which is a sub field of machine learning , concerned with algorithms inspired by efficient searching, recognition and analysis. In the project, Python is used for efficient working and libraries like Flask are used. This application is designed such that it supports different factors associated with plant disease detection. It is best suited for identifying plant disease and suggesting the possible causes and remedies. The primary goal of the project is to protect the agricultural area from diseased plants and to safeguard them by suggesting remedies.

Additionally, the effort strives to save agricultural produce. To obtain great accuracy while maintaining a low latency, we want to improve the proposed system's reaction time and efficiency.

Key words : Python, Flask, HTML, CSS, Deep Learning, CNN

"You have to get up and plant the seed and see if it grows, but you can't just wait around, you have to water it and take care of it."

- Bootsy Collins

List of Figures

List of Abbreviations

Abbr1: CSS- Cascading Style Sheets

Abbr2: HTML – HyperText Markup Language

Abbr3: FLOSS- Free/Libre and Open Software

Abbr4: CNN- Convolutional Neural Network

Abbr5: XML- eXtensible Markup Language

Abbr6: ML- Machine Learning

Abbr7: IoT- Internet of Things

Abbr8: HTTP- Hypertext Transfer Protocol

Abbr9: DB- Database

Abbr10: UML- Unified Modeling Language

Abbr11: WAV- Waveform Audio File Format

Abbr12: GUI- Graphical User Interface

Abbr13: SIANN- Shift-Invariant Artificial Neural Network

Table of Contents

Chapter 1. Introduction

Introduction

During the complete growth phase of plants, from sprout-seedling to vegetative-budding to flowering-ripening, diseases are one type of natural catastrophe that affects the normal growth of plants and even results in plant mortality. Flauralysis is a Deep Learning based model solution that will detect and predict most of the plant diseases which may be caused by pathogenic organisms such as fungi, bacteria, viruses, protozoa, as well as insects and parasitic plants. The productivity, quality and richness of plants are greatly influenced by diseases and pests. The most outrageous problem faced by farmers that causes loss on overall yields include plant diseases, and this has to be solved on an immediate basis to help crop producers separate out the infected plant after recognizing it from its symptoms.

This application will also give some useful tips on the basis of details entered by farmers and will display the possible diseases that may affect crops in the long run and give advice that can be used to prevent the same as “Prevention is always better than cure”. This can help farmers and researchers quickly detect and respond to outbreaks, potentially preventing crop losses and improving yields. [1]

Farmers often implement various management strategies to prevent plant disease, including the use of insecticides, pesticides, and repellents. However, these methods are not always effective, and some plants can become habituated to them, leading to continued deterioration. As such, it is essential for farmers to stay up-to-date on the latest research and best practices in plant disease prevention to protect their crops and livestock and maintain a sustainable farming operation.

1.1 Overview

Plant diseases result in significant reductions in both the quality and quantity of agricultural products as well as productivity and economic losses. In today's crop

monitoring of wide fields of crops, plant disease detection has attracted growing attention. Making the transition from one disease management strategy to another present's challenges for farmers. The conventional method used in practice for finding and identifying plant diseases is expert observation with the unaided eye. In this essay, we examine the requirement for an easy-to-use method for detecting plant-leaf diseases that would speed up agricultural improvements. Early disease diagnosis and crop health information can make it easier to manage illnesses through effective management techniques. Crop productivity will increase as a result of this method. This model also contrasts the advantages and drawbacks of different prospective strategies like Picture capture, image pre-processing, features extraction, and neural network-based classification are some of the procedures that are included in it.

1.2 Background and Motivation

Background:

The agricultural sector is the cornerstone of global food production, supporting economies and livelihoods worldwide. However, it faces a critical challenge: the timely identification and management of plant diseases. Plant diseases can significantly reduce crop yields, jeopardize food security, and result in economic losses for farmers and nations.

Traditional methods of disease detection often rely on manual inspection, which can be time-consuming and prone to errors. Timely intervention is essential for effective disease management. This necessitates a more efficient and accurate approach.

Motivation:

The motivation behind "Flauralysis" is to harness cutting-edge technology, such as artificial intelligence, image processing, and data analytics, to offer a practical solution to the problem of plant disease detection. By providing farmers and

agricultural professionals with a user-friendly tool to swiftly identify diseases, "Flauralysis" aims to mitigate crop damage and increase agricultural productivity.

The development of this plant disease detection system project will contribute to the advancement of plant prevention techniques, benefiting farmers worldwide. The system's affordability and ease of use will make it accessible to farmers of all sizes and resources, contributing to the overall sustainability and growth of the agricultural industry.

1.3 Problem Statement and Objectives

The problem addressed by "Flauralysis" is the inefficiency and inaccuracy of traditional plant disease detection methods. To tackle this problem, the system's objectives are as follows:

- Develop a user-friendly platform for capturing and uploading images of diseased plants.
- Implement advanced image processing and machine learning algorithms to analyze images.
- Identify the specific disease and assess its severity.
- Provide real-time disease management recommendations to end-users.
- Increase the overall efficiency of disease detection and crop management.

1.4 Scope of the Project

The scope of the "Flauralysis" Plant Disease Detection System project encompasses several key dimensions:

- **Disease Detection:** "Flauralysis" focuses on the detection of a wide range of plant diseases, including fungal, bacterial, and viral infections, as well as nutrient deficiencies.
- **User Interface:** The system's user interface is designed to be intuitive, enabling both experts and non-experts to use the application with ease.

- **Mobile Accessibility:** "Flauralysis" is accessible through mobile devices, making it a practical tool for farmers and agricultural professionals in various settings.
- **Data Analysis:** The system employs advanced data analysis techniques to provide accurate disease identification and severity assessment.
- **Recommendations:** "Flauralysis" offers disease management recommendations that include treatment options and preventative measures.

This system will be helpful in reducing the problems faced by farmers due to diseases on a large scale. This system will bring a remarkable change in the agriculture sector as farmers can check their crop health and take preventive measures without worrying about their crop deterioration. The size of the database may increase very rapidly so this system is made in such a way that it is scalable, reliable, easy to use, fast and efficient. The system will reduce damages done to crops by diseases and thus save the cultivation and avoid the loss to the farmers. This system will save time as compared to manually examining crops over the fields all the time of crop cultivation.[2]

1.5 Team Organization

Jayant Neema:

While doing preliminary investigation and understanding the limitations of the current system, I studied about the topic and its scope and surveyed various research papers related to the same. I also did Requirement Engineering, Elicitation and Designing and also the needed documentation for the same. The poster designing work was also done by me. I also worked on designing a dataset schema for the project.

Leena Ghatiya:

I investigated and found the right technology and studied it. For the implementation of the project , I collected the object data and designed it . Implementation logic for

the project objective and coding of internal functionalities is also done by me.

Manas Dhaketa:

I investigated and found the right technology and studied it. For the implementation of the project , I collected the object data and designed it . Implementation logic for the project objective and coding of internal functionalities is also done by me. I worked to make the application as dynamic as it stands today. also worked on the implementation of user friendly design ideas for the project.

Mandvee Vatsa :

Along with doing preliminary investigation and understanding the limitations of the current system, I studied about the topic and its scope and surveyed various research papers related to the same. I also did Requirement Engineering, Elicitation and Designing and also the needed documentation for the same. I also worked on designing a dataset schema via ER diagram for better efficiency of this Project.

1.6 Report Structure

The project **Plant Disease Detection - Flauralysis** is primarily concerned with detection, prediction, suggestions and analysis of plant diseases and the whole project report is categorized into five chapters.

Chapter 1: Introduction- introduces the background of the problem followed by rationale for the project undertaken. The chapter describes the objectives, scope and applications of the project. Further, the chapter gives the details of team members and their contribution in development of the project which is then subsequently ended with a report outline.

Chapter 2: Review of Literature- explores the work done in the area of Project undertaken and discusses the limitations of the existing system and highlights the issues and challenges of the project area. The chapter finally ends up with the requirement identification for present project work based on findings drawn from

reviewed literature and end user interactions.

Chapter 3: Proposed System - starts with the project proposal based on requirement identified, followed by benefits of the project. The chapter also illustrates the software engineering paradigm used along with different design representations. The chapter also includes a block diagram and details of major modules of the project. Chapter also gives insights of different types of feasibility study carried out for the project undertaken. Later it gives details of the different deployment requirements for the developed project.

Chapter 4: Implementation - includes the details of different Technology/ Techniques/ Tools/ Programming Languages used in developing the Project. The chapter also includes the different user interfaces designed in the project along with their functionality. Further it discusses the experiment results along with testing of the project. The chapter ends with evaluation of the project on different parameters like accuracy and efficiency.

Chapter 5: Conclusion - Concludes with objective wise analysis of results and limitation of present work which is then followed by suggestions and recommendations for further improvement.

Chapter 2. Review of Literature

Review of Literature

Plant disease detection plays a pivotal role in agriculture, impacting various aspects of farming, food production, and economic sustainability. Accurate and timely disease detection is essential for ensuring the health and productivity of crops. Plant disease detection is employed to identify diseases in plants and take appropriate measures to manage and control their spread. Here's why it is of paramount importance:

Ensuring Food Security: Plant disease detection is a linchpin of global food security. Crops are the primary source of food for human consumption and animal feed. Any disease outbreak in crops can lead to significant yield losses, affecting the availability of essential food products. This, in turn, can contribute to food shortages, malnutrition, and economic strain on both individual farmers and entire nations.

Preserving Environmental Sustainability: Plant diseases often necessitate the use of pesticides to control their spread. Excessive and indiscriminate use of pesticides can have detrimental effects on the environment. It can lead to soil and water pollution, harm non-target organisms, and disrupt ecological balance. Automated disease detection and targeted treatment can reduce the need for excessive pesticide application, contributing to sustainable and environmentally friendly agriculture.

Economic Impact: Plant diseases have a direct and severe economic impact on farmers. Disease outbreaks can result in reduced crop yields, increased expenses for disease management, and sometimes complete crop failure. The financial losses incurred by farmers not only affect their livelihood but also have a ripple effect on local and national economies. The economic stability of farming communities depends on effective disease detection and management.

Personal and Financial Consequences: Farmers and agricultural laborers often invest significant amounts of time, effort, and financial resources into crop cultivation. The failure to detect and manage plant diseases in a timely manner can lead to personal hardships and financial crises. Automated disease detection systems

can provide a lifeline for these individuals by preserving their investments and protecting their income.

Global Agricultural Sustainability: Sustainable agriculture is critical for meeting the ever-growing global demand for food. Plant disease detection is a key component of sustainable farming practices. It enables farmers to protect their crops, reduce environmental harm, and optimize resource utilization. By contributing to the sustainability of agriculture, disease detection systems support long-term food production and the well-being of farming communities worldwide.

2.1 Preliminary Investigation

2.1.1 Current System

Preliminary investigation in the context of plant disease detection using deep learning involves a comprehensive exploration of existing systems, their limitations, and the identification of requirements for the project. This section is critical for understanding the current landscape of plant disease detection and formulating the project's objectives.[3]

`

2.2 Limitations of Current System

The current landscape of plant disease detection relies heavily on manual visual inspection by agricultural experts and farmers. This approach is time-consuming, labor-intensive, and prone to errors, as it depends on the expertise of individuals. The limitations of this manual system include the following:

- **Subjectivity:** One of the significant limitations of the current system is its subjectivity. Disease identification depends on the expertise and experience of the individuals conducting the inspections. This subjectivity can lead to inconsistencies and misdiagnoses, impacting disease management strategies.
- **Time-Consuming:** Manual inspections are time-consuming, especially for larger agricultural areas. The delay in disease detection can lead to the rapid spread

of diseases and increased crop losses.

- **Expertise Dependency:** The current system heavily relies on the availability of agricultural experts. Many regions lack access to such experts, resulting in delayed disease identification and inadequate treatment.

2.3 Requirement Identification and Analysis for Project

To address the limitations of the current system, a project in the field of plant disease detection using deep learning is crucial. This section involves the analysis of a sample project and a research paper to identify the key requirements and objectives.

2.3.1 Sample Project Analysis

A sample project on plant disease detection using deep learning can provide valuable insights into the requirements and objectives of such an endeavor. Let's analyze a hypothetical project:

Sample Project Title: "Deep Learning-Based Plant Disease Detection for Tomato Crops"

Objective: The objective of this project is to develop an automated system capable of accurately identifying and classifying diseases in tomato crops based on leaf images. The project aims to provide a user-friendly interface for farmers and experts, enabling them to detect diseases swiftly.[4]

Key Components:

- **Data Collection:** The project requires a diverse dataset of tomato plant images, including healthy plants and those affected by various diseases.
- **Deep Learning Model:** Developing and training a deep learning model, potentially using CNN architectures like ResNet, to perform disease classification.
- **User Interface:** Creating a user-friendly interface for farmers and experts to interact with the system. This interface should allow users to upload images for analysis and receive disease reports.

- **Data Annotation:** An essential part of the project involves annotating the collected images with disease labels to create a ground truth dataset for model training and evaluation.
- **Performance Metrics:** The project should define evaluation metrics to assess the model's accuracy, precision, recall, and F1 score.
- **Technology Stack:** The project will rely on Python, deep learning frameworks like TensorFlow or PyTorch, and potentially web development technologies for the user interface.

2.3.2 Research Paper Analysis

Analyzing relevant research papers can provide further insights into the requirements and approaches for a plant disease detection project. Let's take a hypothetical research paper:

Research Paper Title: "A Deep Learning Approach for Automated Plant Disease Detection Using Multispectral Imaging"

Key Findings:

The paper focuses on using multispectral imaging, which captures information beyond visible light, enhancing disease detection accuracy. The importance of a comprehensive and diverse dataset is emphasized. It highlights the need for images of various plant species and diseases to train a robust model. Transfer learning from pre-trained models can significantly reduce the amount of data and training time required for custom models. Real-time disease detection capabilities can be crucial, especially for immediate action in the field.[5]

2.3.3 Conclusion

In conclusion, the preliminary investigation reveals that the current system for plant disease detection has significant limitations related to subjectivity, time

consumption, and expertise dependency. To address these limitations, a sample project and a research paper analysis highlight the essential requirements for a deep learning-based plant disease detection system. These requirements include data collection, model development, user interface, data annotation, performance metrics, and a suitable technology stack. Additionally, the use of multispectral imaging and transfer learning can enhance accuracy and efficiency in disease detection. The findings from this analysis will serve as a valuable foundation for the upcoming project in plant disease detection using deep learning.

Chapter 3. Proposed System

Proposed System

3.1 The Proposal

The proposal of **"Flauralysis"** aims to revolutionize the field of plant disease detection. It is designed as a comprehensive system that leverages modern technology, including artificial intelligence and image processing, to automate and enhance the process of identifying plant diseases accurately.

It will offer a user-friendly interface that allows farmers and agricultural professionals to capture images of diseased plants using smartphones or cameras.[6] The system will then process these images and provide real-time analysis, identifying the specific disease and its severity. The system will also offer recommendations for disease management and prevention.

3.2 Benefits of the Proposed System

The benefits of implementing "Flauralysis" are multi-faceted and include:

1. **Early Disease Detection:** The system can detect plant diseases at early stages, helping farmers take timely preventive measures to limit crop damage.

2. **Accuracy:** Utilizing advanced image analysis algorithms, "Flauralysis" offers highly accurate disease identification and severity assessment.

3. **Ease of Use:** The user-friendly interface ensures that both experts and non-experts can conveniently use the system.

4. **Cost Efficiency:** By enabling targeted and precise disease management, the system reduces the need for broad-spectrum treatments, saving on resources and minimizing environmental impact.

5. **Increased Yields:** Timely detection and intervention lead to higher crop

yields, improving overall agricultural productivity.

3.3 Block Diagram

The block diagram of **"Flauralysis"** provides a visual representation of the system's architecture. It outlines the various components and their interactions, including image input, processing modules, disease recognition, and user interface.

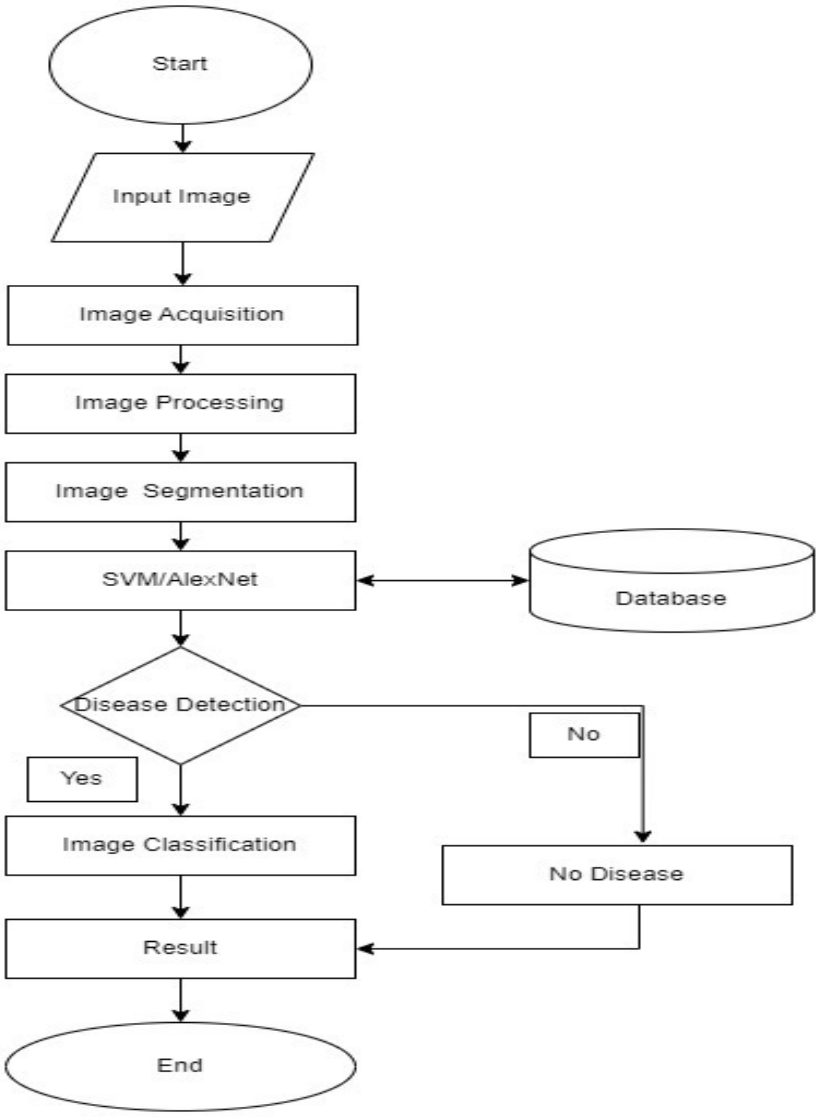


Figure 3-1 : Block Diagram

3.4 Feasibility Study

A feasibility study is an analysis of how successfully a system can be implemented, accounting for factors that affect it such as economic, technical and operational factors to determine its potential positive and negative outcomes before investing a considerable amount of time and money into it. A feasibility study is a crucial aspect of any system development. It assesses whether the proposed system is viable and practical from various perspectives. "Flauralysis" underwent a comprehensive feasibility study, covering three key aspects:

3.4.1 Technical

Technical feasibility focuses on the system's technological aspects. We evaluated whether the technology required for "Flauralysis" is readily available or could be developed within a reasonable timeframe. Our study confirmed that the necessary technologies, including image processing libraries and machine learning tools, are well-established and accessible.

3.4.2 Economic

Economic feasibility analyzes the cost-effectiveness of the system. "Flauralysis" is deemed economically feasible due to the potential return on investment. The system's cost of development, implementation, and maintenance is justified by the increased agricultural productivity, reduced resource usage, and crop yield improvements.

3.4.3 Operational

Operational feasibility assesses whether the proposed system can be seamlessly integrated into existing operations. We found that "Flauralysis" aligns well with existing agricultural practices. It is adaptable to diverse environments and can be operated by individuals with varying levels of technical expertise.

3.5 Design Representation

3.5.1 Data Flow Diagrams

A data-flow diagram is a way of representing a flow of data through a process or a system. The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow — there are no decision rules and no loops

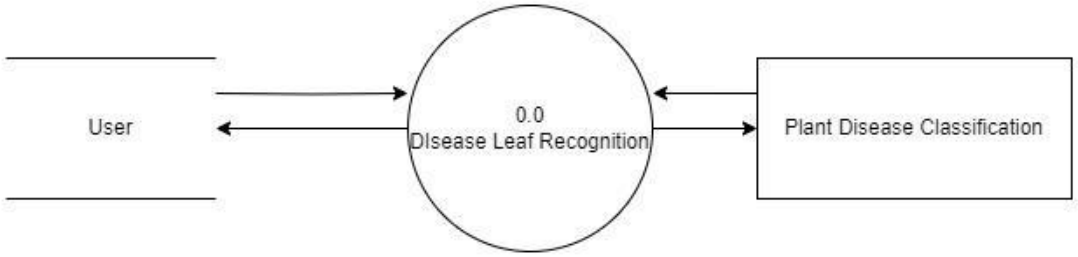


Figure 3-2 Data Flow Diagram Level 0

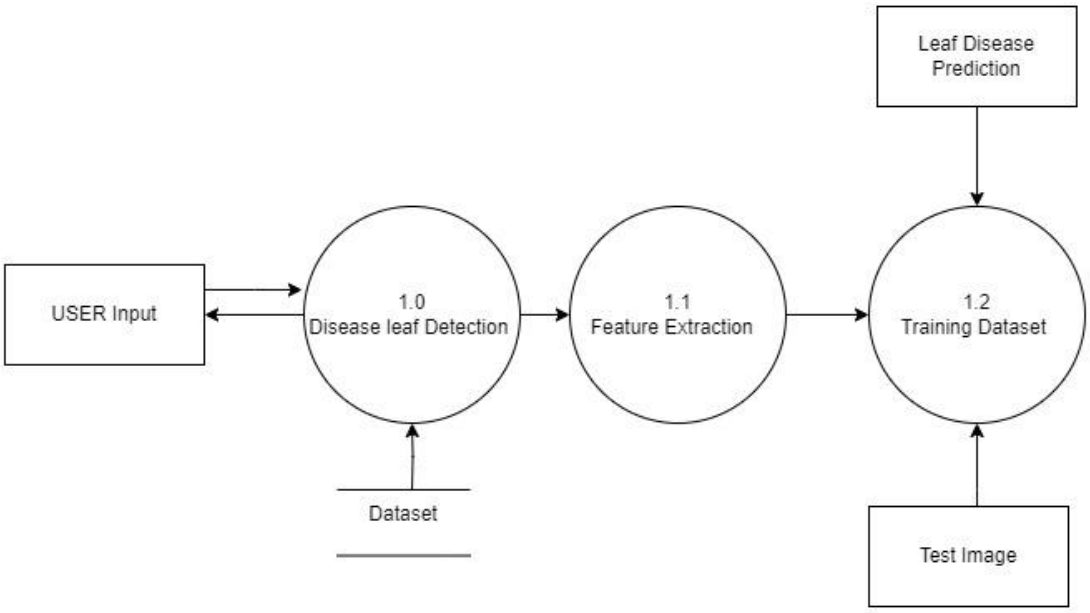


Figure 3-3 Data Flow Diagram Level 1

3.5.2 Sequence Diagram

A sequence diagram or system sequence diagram shows process interactions arranged in time sequence in the field of software engineering. It depicts the processes involved and the sequence of messages exchanged between the processes

needed to carry out the functionality.

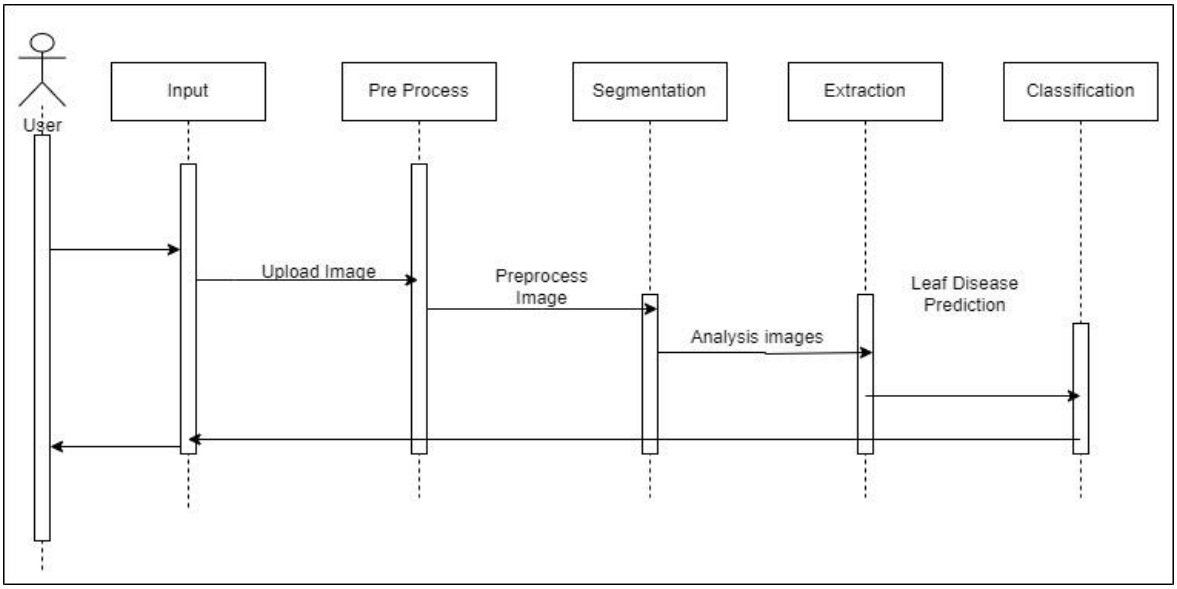


Figure 3-4 Sequence Diagram

3.5.3 Entity Relationship Diagram

An entity–relationship model describes interrelated things of interest in a specific domain of knowledge. A basic ER model is composed of entity types and specifies relationships that can exist between entities.

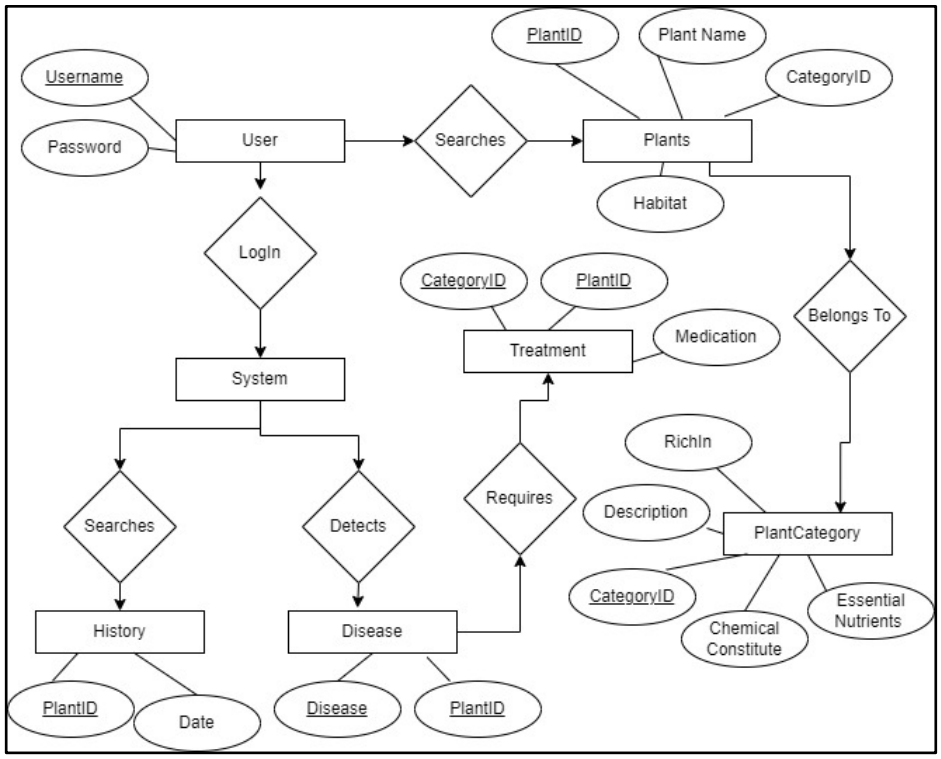


Figure 3-5 ER Diagram

3.5.4 UML Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e workflows), as well as the data flows intersecting with the related activities.Although activity diagrams primarily show the overall flow of control, they can also include elements showing the flow of data between activities through one or more data stores.

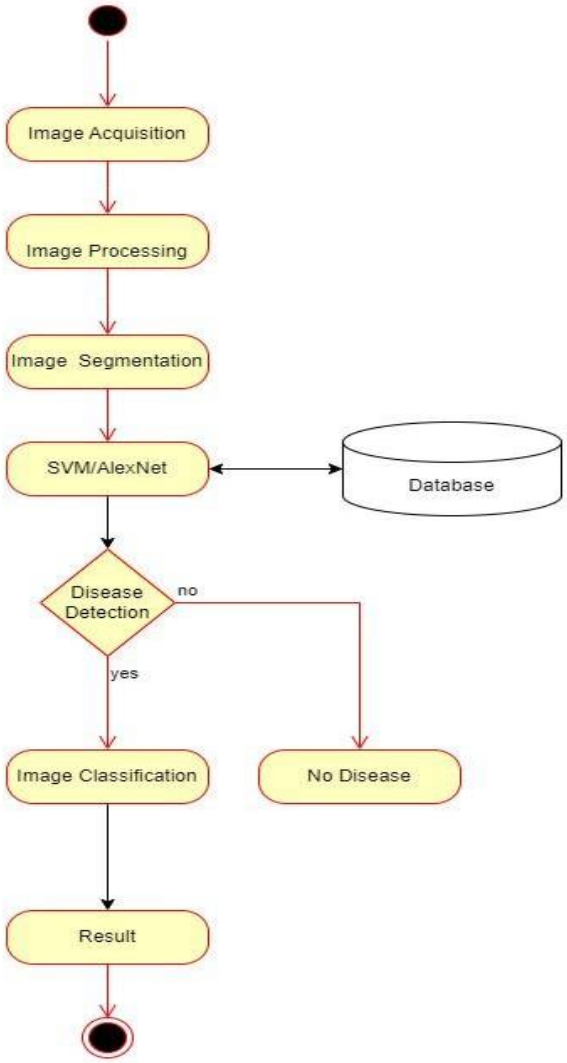


Figure 3-6 Activity Diagram

3.5.5 UML Class Diagram

In software engineering, a class diagram in the Unified Modeling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations, and the relationships among objects.

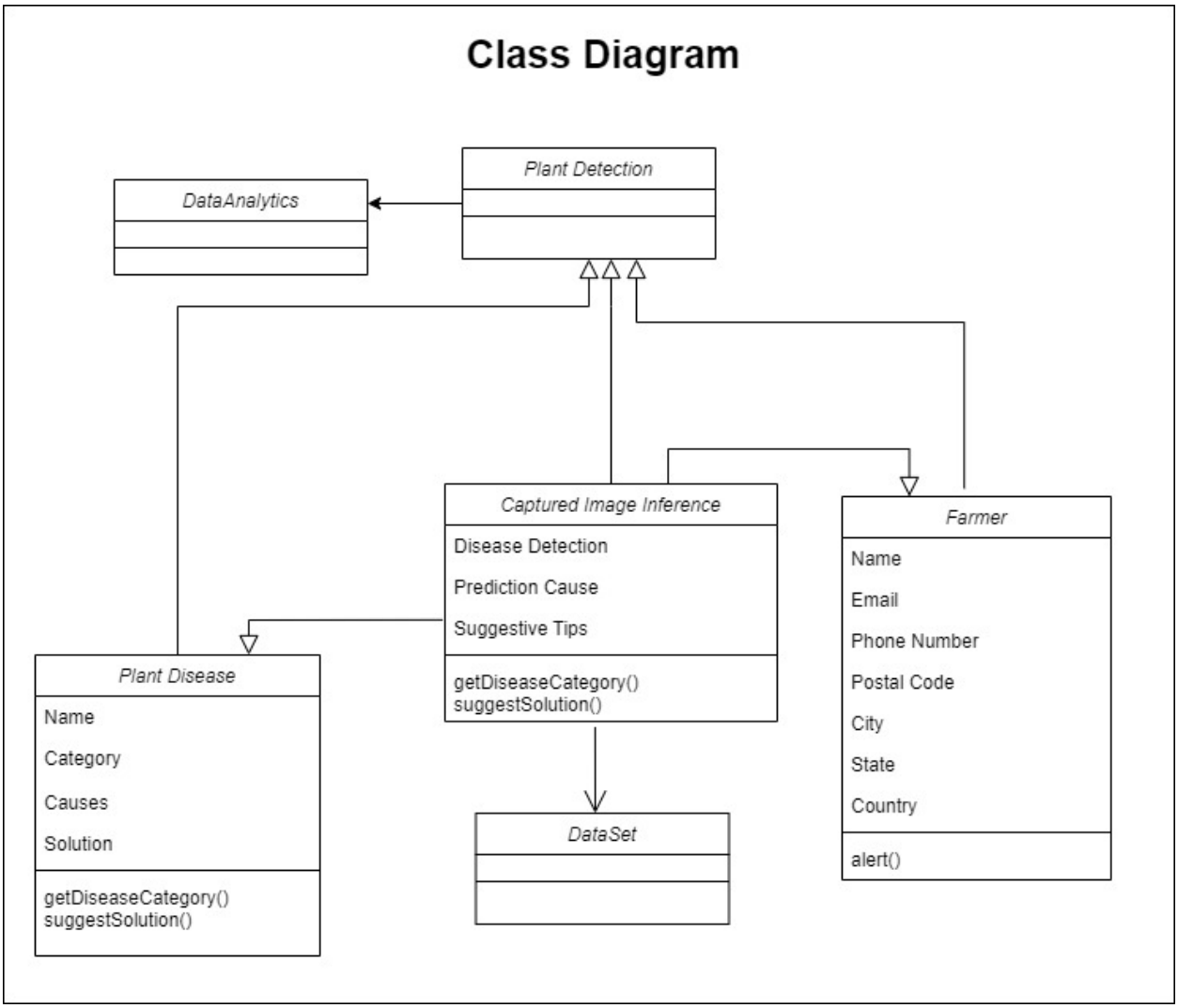


Figure 3-7 Class Diagram

3.5.6 UML Use Case Diagram

A use case diagram is used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and also tells how the user handles a system.

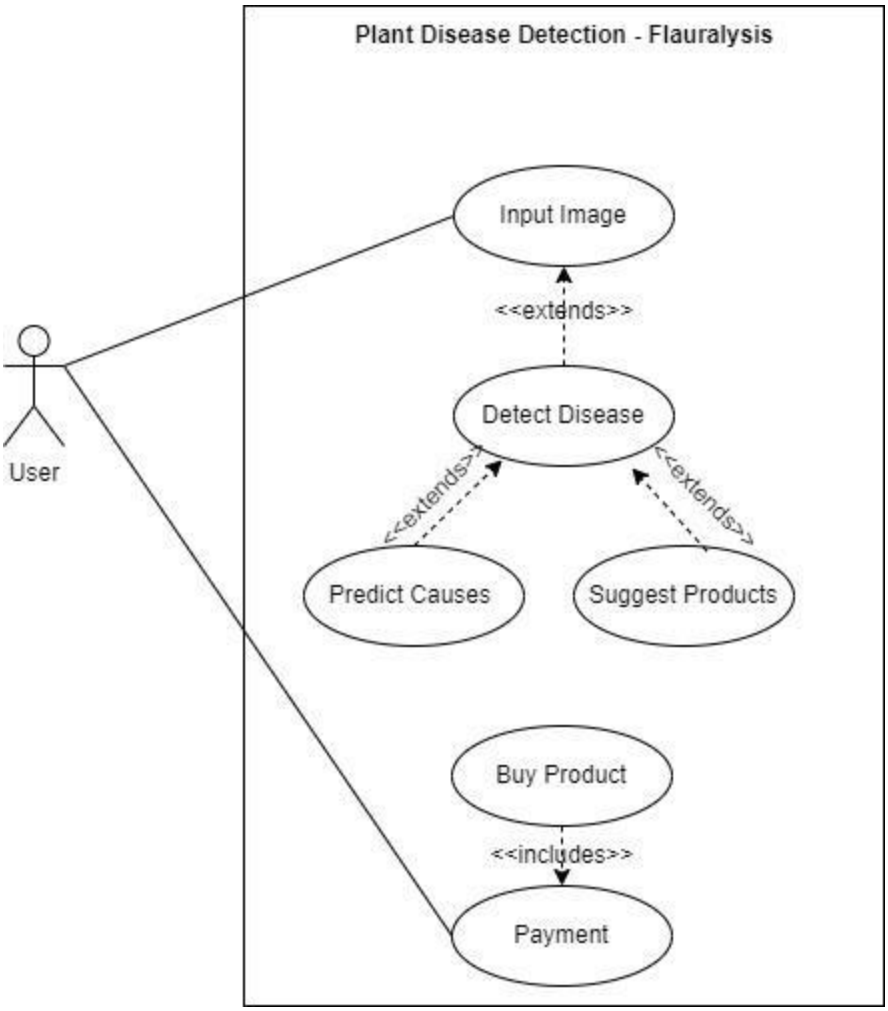


Figure 3-8 Use Case Diagram

3.5.7 Package Diagram

Package diagram, a kind of structural diagram, shows the arrangement and organization of model elements in a middle to large scale project. Package diagram can show both structure and dependencies between subsystems or modules, showing different views of a system, for example, as multi-layered (aka multi-tiered) application - multi-layered application model.

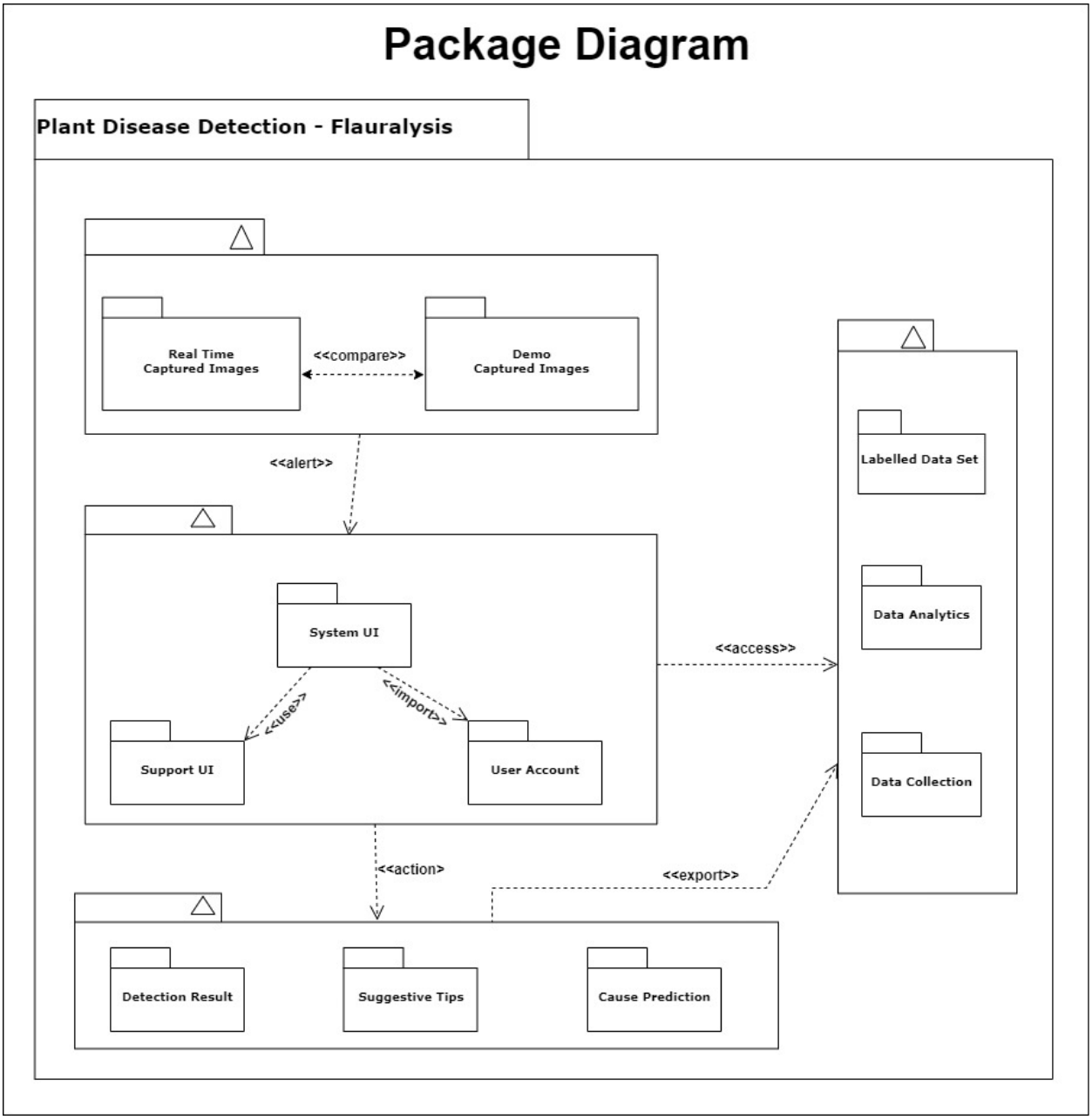


Figure 3-9 Package Diagram

3.5.8 Architectural Diagram

Software environments are complex, and they aren't static. New features are frequently added to accommodate growing customer needs and demands. Your team, even those team members who aren't immersed in the code every day, needs to understand your organization's software architecture so it can scale seamlessly.

This is where software architecture diagrams come in. They give the entire development team a visual overview making it easier to communicate ideas and key concepts in terms everyone can understand.

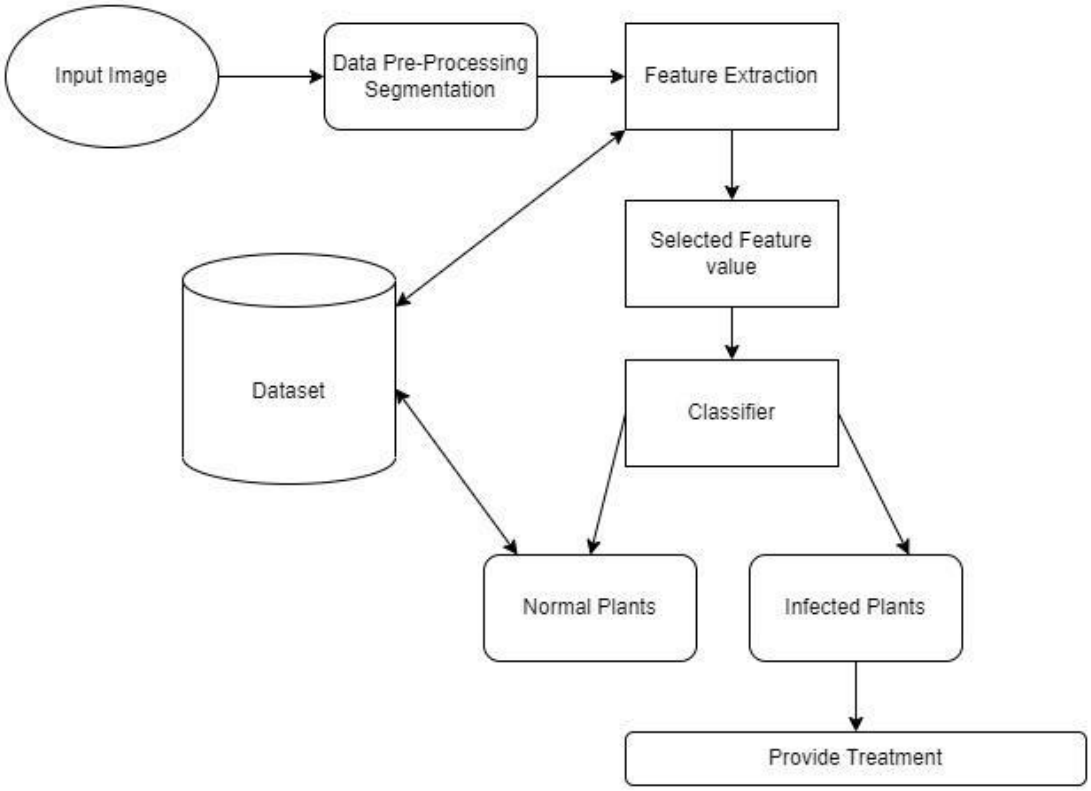


Figure 3-10 Architectural Diagram

3.6 Deployment Requirements

There are various requirements (hardware, software and services) to successfully deploy the system. These are mentioned below :

3.6.1 Hardware

- **Server Hardware:** The system should be hosted on robust servers with sufficient processing power and memory to handle image processing and machine learning tasks.
- **Storage Devices:** Adequate storage is essential for archiving image data and system backups.

- **Sensors and Cameras:** In agricultural settings, high-quality sensors and cameras may be necessary for capturing clear and detailed images of plants.

3.6.2 Software

- **Operating System:** The deployment environment should use a stable and secure operating system.
- **Databases:** Relational or NoSQL databases are required for storing user data, disease profiles, and historical information.
- **Framework:** The system's core components should be built using appropriate frameworks for web application development.
- **Machine Learning Libraries:** Utilize machine learning libraries like TensorFlow and Keras for disease detection models.
- **Web Servers:** A web server, such as Apache or Nginx, should be employed to host the user interface.
- **Image Processing Tools:** Software for image preprocessing, manipulation, and analysis is essential.

3.6.3 Communication Requirements

- **Network Infrastructure:** A stable internet connection is required for real-time data exchange between users and the server.
- **Encryption:** Implement strong encryption protocols to secure data transmission.
- ` **User Authentication:** Develop a secure authentication system to protect user accounts

Chapter 4. Implementation

Implementation

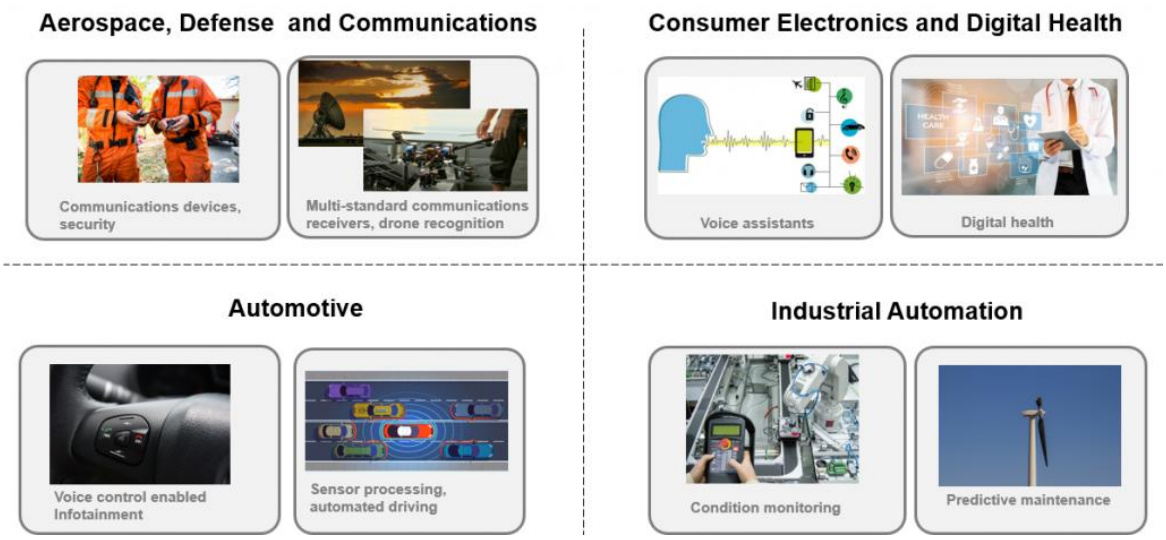
For the problem of farmers losing their crops due to diseases this application comes to the rescue. We built this application for the sole purpose of automation to help in detection of plant disease and use the reports further to reduce causes of disease. Concept of ML is used to make this application a huge success.

4.1 Technique Used

4.1.1 Deep Learning :

Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks. Deep learning (also known as deep structured learning or hierarchical learning) is part of a broader family of machine learning methods based on learning data representations, as opposed to task-specific algorithms. Learning can be supervised, semi-supervised or unsupervised. Deep learning models are loosely related to information processing and communication patterns in a biological nervous system, such as neural coding that attempts to define a relationship between various stimuli and associated neuronal responses in the brain.

Deep learning architectures such as deep neural networks, deep belief networks and recurrent neural networks have been applied to fields including computer vision, speech recognition, natural language processing, audio recognition, social network filtering, machine translation, bioinformatics and drug design, where they have produced results comparable to and in some cases superior to human experts.[7]



Courtesy- [Deep Learning for Signal Processing Applications » Artificial Intelligence - MATLAB & Simulink]

Figure 4-1 Deep Learning Applications

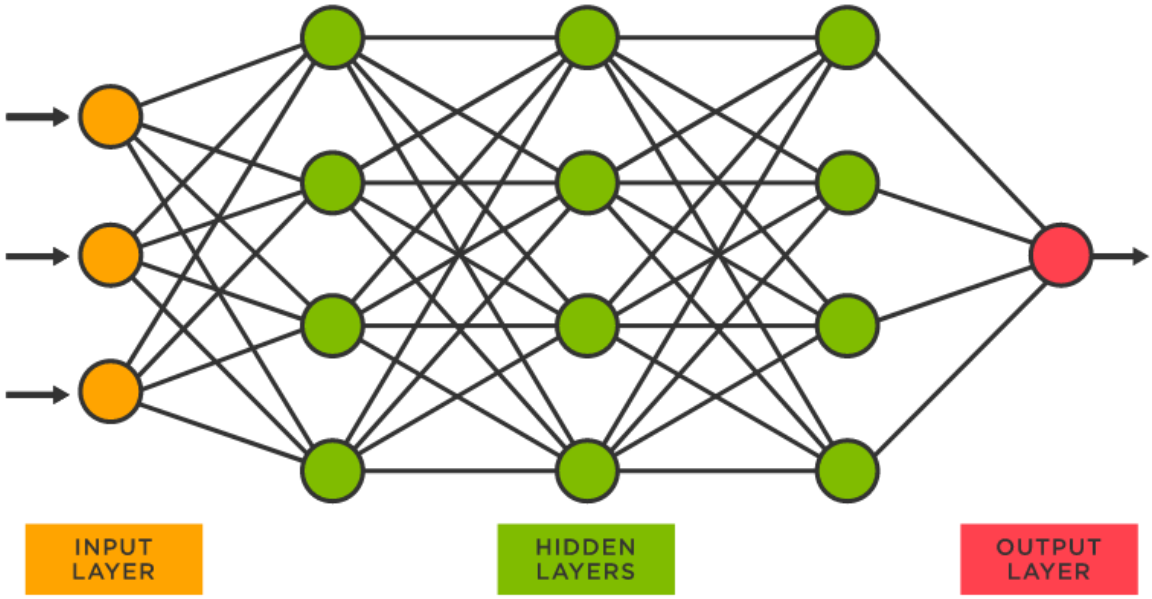
4.1.2 Neural Networks:

In machine learning, a convolutional neural network (CNN, or ConvNet) is a class of deep, feed-forward artificial neural networks that has successfully been applied to analyzing visual imagery.

CNNs use a variation of multilayer perceptrons designed to require minimal preprocessing. They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and translation invariance characteristics.

Convolutional networks were inspired by biological processes in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field. CNNs use relatively little pre-processing compared to other image classification algorithms. This means that the network learns the filters that in traditional algorithms were hand-engineered. This independence from prior knowledge and human effort in feature design is a major

advantage. They have applications in image and video recognition, recommender systems and natural language processing.



Courtesy-

Courtesy-TIBCO software

Figure 4-2 Neural Network

4.2 Tools Used

4.2.1 Flask Module

Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. It began as a simple wrapper around Werkzeug and Jinja and has become one of the most popular Python web application frameworks.

Flask offers suggestions, but doesn't enforce any dependencies or project layout. It is up to the developer to choose the tools and libraries they want to use. There are many extensions provided by the community that make adding new functionality easy.

4.2.2 Draw.io

Designed by Seibert Media, draw.io is proprietary software for making diagrams and charts. The software lets you choose from an automatic layout function, or create a custom layout. They have a large selection of shapes and hundreds of visual elements to make your diagram or chart one-of-a-kind. The drag-and-drop feature makes it simple to create a great looking diagram or chart. Draw.io has options for storing saved charts in the cloud, on a server, or network storage at a data center, depending on your needs

4.3 Language Used

4.3.1 Python

Python language is used in the system due to the following characteristics :

Simple :

Python is a simple and minimalistic language. Reading a good Python program feels almost like reading English (but very strict English!). This pseudocode nature of Python is one of its greatest strengths. It allows you to concentrate on the solution to the problem rather than the syntax i.e. the language itself.

Free and Open Source :

Python is an example of a FLOSS (Free/Libre and Open Source Software). In simple terms, you can freely distribute copies of this software, read the software's source code, make changes to it, use pieces of it in new free programs, and that you know you can do these things. FLOSS is based on the concept of a community which shares knowledge. This is one of the reasons why Python is so good - it has been created and improved by a community who just want to see a better Python.

Object Oriented :

Python supports procedure-oriented programming as well as object-oriented programming. In procedure-oriented languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In object-oriented languages, the program is built around objects which combine data and

functionality. Python has a very powerful but simple way of doing object-oriented programming, especially, when compared to languages like C++ or Java.[8]

Extensive Libraries :

The Python Standard Library is huge indeed. It can help you do various things involving regular expressions, documentation generation, unit testing, threading, databases, web browsers, CGI, ftp, email, XML, XML-RPC, HTML, WAV files, cryptography, GUI(graphical user interfaces) using Tk, and also other system-dependent stuff. Remember, all this is always available wherever Python is installed. This is called the "batteries included" philosophy of Python.

4.3.2 HTML (HyperText Markup Language): It provides the basic structure of sites, which is enhanced and modified by other technologies like CSS and JavaScript.rather than using a programming language to perform functions, HTML uses tags to identify different types of content and the purposes they each serve to the webpage.Every web page is made up of a bunch of these HTML tags denoting each type of content on the page. Each type of content on the page is "wrapped" in, i.e. surrounded by, HTML tags.

4.3.3 CSS (Cascading Style Sheets): It is used to control presentation, formatting, and layout.This programming language dictates how the HTML elements of a website should actually appear on the frontend of the page. Whereas HTML was the basic structure of your website, CSS is what gives your entire website its style. Those slick colors, interesting fonts, and background images? All thanks to CSS. This language affects the entire mood and tone of a web page, making it an incredibly powerful tool and an important skill for web developers to learn. It's also what allows websites to adapt to different screen sizes and device types.

4.4 Screenshots

The Following are the screenshots of the result of the project :

Figure 4-3 Flauralysis Website

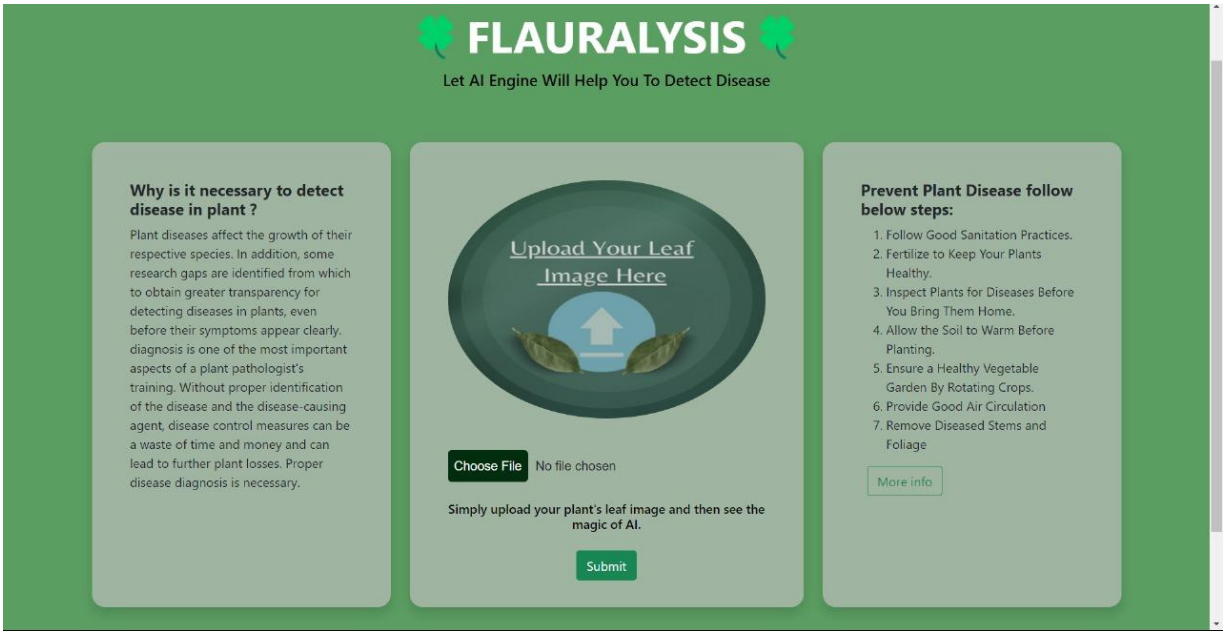


Figure 4-4 Page to Upload Image

4.5 Testing

Testing is the process of evaluation of a system to detect differences between given input and expected output and also to assess the features of the system. Testing assesses the quality of the product. It is a process that is done during the development process.

4.5.1 Strategy Used

A comprehensive testing strategy ensures that "Flauralysis" performs reliably and accurately :

- **Unit Testing:** Individual components, algorithms, and functions are tested for correctness.
- **Integration Testing:** The interactions between different system modules are evaluated to ensure smooth data flow.
- **System Testing:** The entire system, including the user interface and backend, is tested for end-to-end functionality. Tests can be conducted based on two approaches –
- **Functionality testing :** It is a type of software testing that validates the software system against the functional requirements/specifications. The purpose of Functional tests is to test each function of the software application, by providing appropriate input, verifying the output against the Functional requirements.
- **Implementation testing :** The texting method used here is Black Box Testing. It is carried out to test functionality of the program. It is also called 'Behavioral' testing. The tester in this case, has a set of input values and respective desired results. On providing input, if the output matches with the desired results, the program is tested 'ok', and problematic otherwise.
- **User Testing:** Real users participate in testing to provide insights into user-friendliness and system performance.

4.5.2 Test Case and Analysis

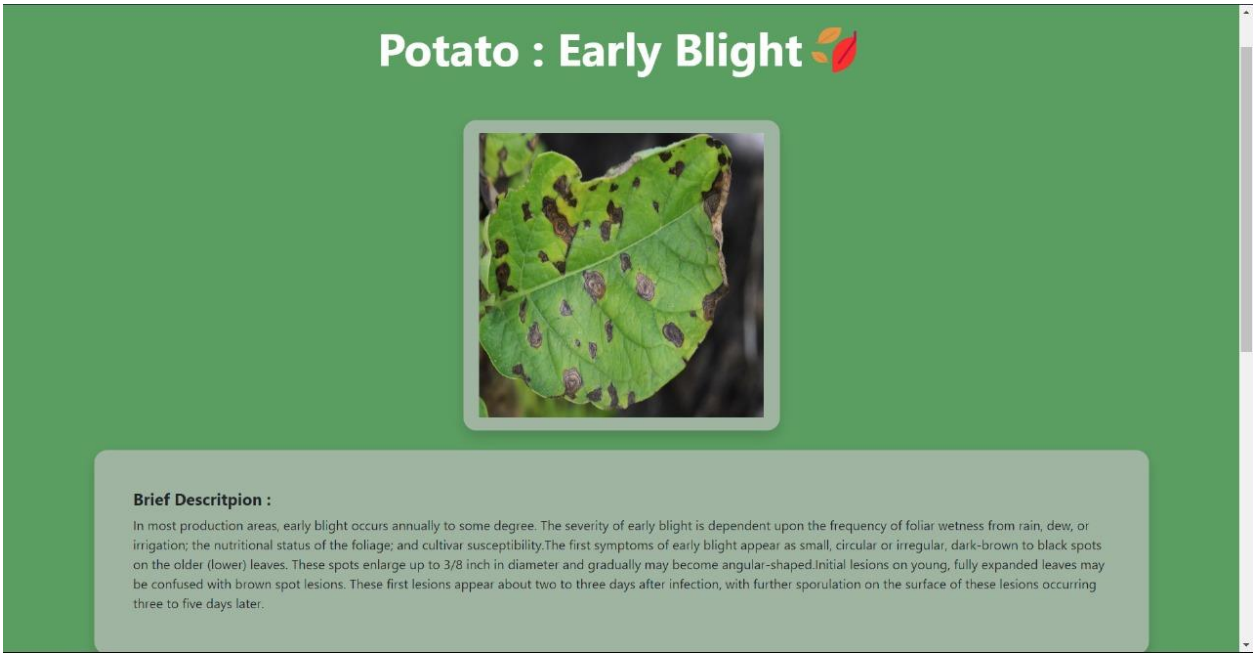


Figure 4-5 Plant Disease Testing

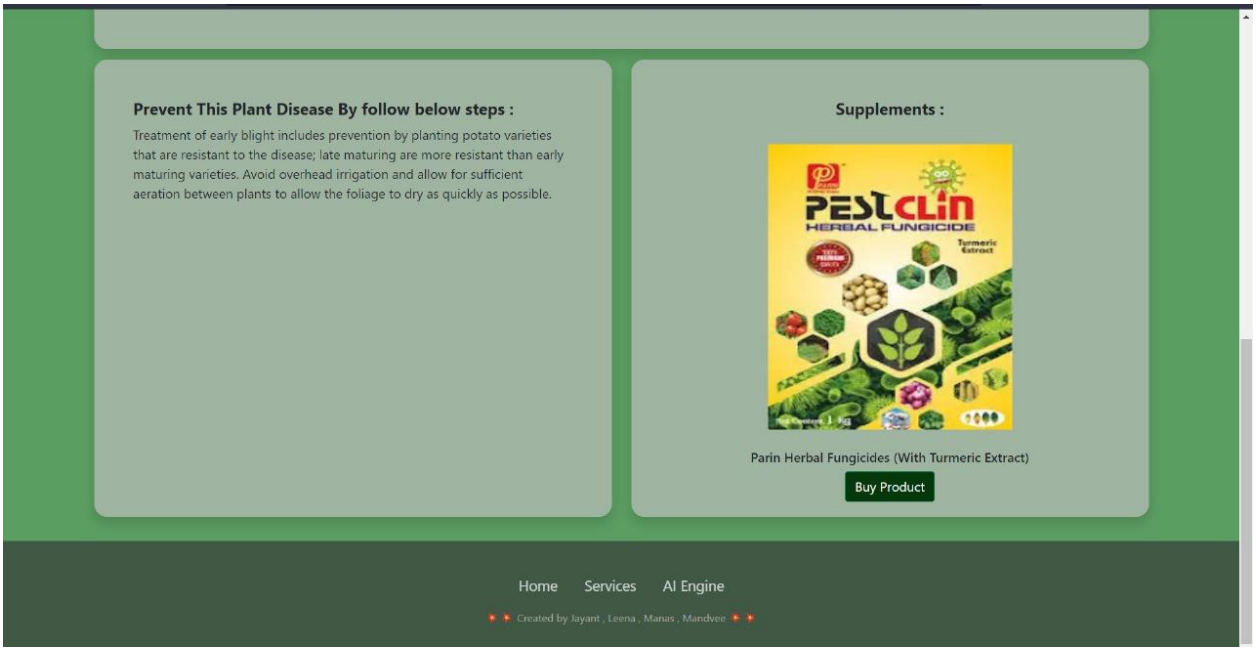


Figure 4-7 Suggestive Tips and Product

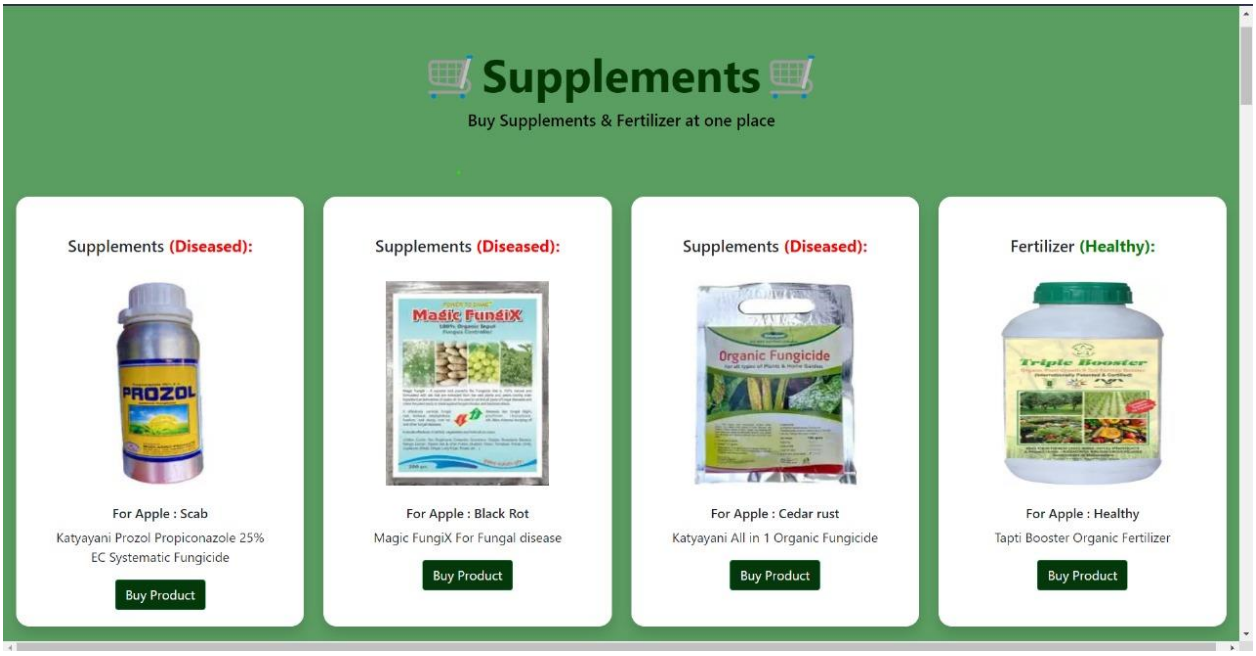


Figure 4-8 Product recommendations

Chapter 5. Conclusion

Conclusion

5.1 Conclusion

The implementation of "Flauralysis" represents a significant milestone in the field of plant disease detection. By harnessing deep learning and neural network technologies, the system can accurately identify diseases and assess their severity, offering practical recommendations for users. The use of Flask for web application development ensures a user-friendly experience.

5.2 Limitations of the Work

The system has certain limitations, including dependency on reliable internet connectivity, the need for high-quality image capture, and the requirement for constant updates to enhance disease detection accuracy.

- **Cost:** The cost of implementing and maintaining the system may be high, depending on the type and number of sensors used, the complexity of the software, and the manpower required to monitor and respond to alerts.
- **Power supply:** The system may require a continuous and reliable power supply, which may be challenging in remote or rural areas with limited access to electricity.

5.3 Suggestion and Recommendations for Future Work

Future work should focus on addressing limitations, expanding the database of plant diseases for enhanced detection capabilities, and incorporating additional features such as multi-language support. Continuous research into deep learning models is crucial to improving accuracy and performance.

"Flauralysis" is poised to be a transformative tool in the agricultural sector, aiding in the early detection and management of plant diseases, ultimately leading to increased crop yields and food security.

Here are some of the potential future scopes for this project:

- **Integration with IoT:** The plant disease detection system can be integrated with the Internet of Things (IoT) to enable real-time monitoring of the farm. The IoT integration can help in tracking various environmental factors such as temperature, humidity, and soil moisture levels that can affect the farm's yield.
- **Machine Learning:** Machine learning can be integrated into the plant disease detection system to improve the accuracy of disease detection algorithms. This integration can help to identify specific types of diseases, which can be useful in developing specific strategies to keep them away from the farm.
- **Mobile Application:** A mobile application can be developed to enable farmers to monitor their crops' security status remotely. This application can help in providing farmers with real-time alerts and enabling them to take prompt actions.
- **Integration with Drones:** Drones can be integrated with the plant disease detection system to provide a more comprehensive view of the farm. The drones can fly over the farm to provide real-time images and videos, which can help in identifying disease detection in real-time.
- **Integration with Smart Farming:** The plant disease detection system can be integrated with smart farming technologies to provide farmers with comprehensive insights into the health of their crops and soil. The integration can help in developing precise treatment plans to prevent crop damage due to animal intrusion.

In conclusion, the future scope of the plant disease detection project for farms is significant, and the project has the potential to be integrated with various technologies to improve its performance and effectiveness.

Bibliography

[1] H. Fujiyoshi, A. J. Lipton and R. S. Patil, "Moving Target Classification and Tracking from Real-time Video," in Fourth IEEE workshop, 1998.

[2] D. Comaniciu, R. Vishwanathan and P. Meer, "Real-Time Tracking of Non-Rigid Objects using Mean Shift," in IEEE, 2000.

[3] K. Levi and Y. Weiss, "Learning Object Detection from a Small Number of Examples: the Importance of Good Features," in IEEE Computer Society Conference, 2004.

[4] V. Lepetit, P. Lagger and P. Fua, "Randomized Trees for Real-Time Keypoint Recognition," in IEEE Computer Society Conference, 2005.

[5] T. Yang, S. Z. Li and J. Li, "Real-time Multiple Objects Tracking with Occlusion Handling in Dynamic Scenes," in IEEE Computer Society Conference, 2005.

[6] J. Zhou and J. Hoang, "Real Time Robust Human Detection and Tracking System," in IEEE Computer Society Conference, 2005.

[7] C. P. Papageorgiou, M. Oren and T. Poggio, "A General Framework for Object Detection," in Center for Biological and Computational Learning Artificial Intelligence Laboratory, Cambridge, 2005.

[8] R. D. Charette and F. Nashashibi, "Real TimeVisual Traffic Lights Recognition Based on Spot Light Detection andAdaptive Traffic Lights Template," in IEEE, 2009

Guide Interaction Sheet

Date	Discussion	Action Plan
09/08/23	Discussed the title of Project	Plant Disease Detection - Flauralysis was decided as the topic
12/08/23	Discussion on the technology to be used for object detection in real-time	Python, Flask, HTML, CSS and other tools were finalized
16/08/23	Discussion of the creation of synopsis of the project	Gathering of information for synopsis creation
23/08/23	Suggestions on how to do a literature survey and preliminary investigation on the topic	Many research papers were read , understood and their abstract were to be written.
06/09/23	Discussion on the implementation of the project and other work like presentation and diagrams	Understanding technology to be used and the model we are building
09/09/23	Discussion on the objective of the project (detecting animal in real-time on fields and alerting farmer using notification and sound system)	Decided to Include the logic of suggesting tips and probable causes.
20/09/23	Suggestion for counting the number of animals and detecting their species	Took steps for adding and modifying the program for suggesting solutions and predicting causes.
13/10/23	For generation of log files and storing the result, database was advised to be added	Action taken for worst case testing so that application works best for most of the cases.
30/10/23	Discussion on project documentation	Decided to write the content and integrate it in the proper format of the report

Source Code

```
import os
from flask import Flask, redirect, render_template, request
from PIL import Image
import torchvision.transforms.functional as TF
import CNN
import numpy as np
import torch
import pandas as pd

disease_info = pd.read_csv('disease_info.csv' , encoding='cp1252')
supplement_info = pd.read_csv('supplement_info.csv',encoding='cp1252')
model = CNN.CNN(39)
model.load_state_dict(torch.load("plant_disease_model_1_latest.pt"))
model.eval()
def prediction(image_path):
    image = Image.open(image_path)
    image = image.resize((224, 224))
    input_data = TF.to_tensor(image)
    input_data = input_data.view((-1, 3, 224, 224))
    output = model(input_data)
    output = output.detach().numpy()
    index = np.argmax(output)
    return index

app = Flask(_name_)
@app.route('/')
def home_page():
    return render_template('home.html')
@app.route('/contact')
def contact():
    return render_template('contact-us.html')
@app.route('/index')
```

```
def ai_engine_page():
   return render_template('index.html')
@app.route('/mobile-device')
def mobile_device_detected_page():
   return render_template('mobile-device.html')
@app.route('/submit', methods=['GET', 'POST'])
def submit():
   if request.method == 'POST':
      image = request.files['image']
      filename = image.filename
      file_path = os.path.join('static/uploads', filename)
      image.save(file_path)
      print(file_path)
      pred = prediction(file_path)
      title = disease_info['disease_name'][pred]
      description =disease_info['description'][pred]
      prevent = disease_info['Possible Steps'][pred]
      image_url = disease_info['image_url'][pred]
      supplement_name = supplement_info['supplement name'][pred]
      supplement_image_url = supplement_info['supplement image'][pred]
      supplement_buy_link = supplement_info['buy link'][pred]
      return render_template('submit.html' , title = title , desc = description , prevent =
prevent ,
                        image_url = image_url , pred = pred ,sname = supplement_name , simage =
supplement_image_url , buy_link = supplement_buy_link)
@app.route('/market', methods=['GET', 'POST'])
def market():
   return render_template('market.html', supplement_image =
list(supplement_info['supplement image']),
                     supplement_name = list(supplement_info['supplement name']), disease =
list(disease_info['disease_name']), buy = list(supplement_info['buy link']))
if _name_ == '_main_':
   app.run(debug=True)
```

```
{% extends 'base.html' %}
{% block pagetitle %}
Plant Disease Detection
{% endblock pagetitle %}
{% block body %}
<style>
 body{
  overflow-x: hidden;
 }
 @media only screen and (max-width: 830px) {
  .laptop{
   display: none;
  }
 }
 @media (min-width:830px) and (max-width: 1536px){
  .mobile{
   display: none;
  }
 }
</style>
<div class='mobile p-5'>
 <div class="row mb-5 text-center text-white">
  <div class="col-lg-10 mx-auto">
   <h1 class="display-4" style="padding-top: 2%;font-weight:
400;color:red;"><b>Note✍</b></h1>
   <p class="lead" style="font-weight: 500;color: black;">Please 🙏 Open This Site On
Laptop/PC 💻 Screen Only</p>
   <p class="lead" style="font-weight: 500;color: black;">Thank You 🤝 </p>
  </div>
</div>
</div>
<div class="row mb-5 text-center text-white laptop">
```

```
  <div class="col-lg-10 mx-auto">
    <h1 class="display-4" style="padding-top: 2%;font-weight: 400;color: rgb(255, 255,
255);"><b> 🍁 Plant Disease
        Detection 🍁 </b></h1>
    <p class="lead" style="font-weight: 500;color: rgb(255, 255, 255);">This AI Engine Will
Help To Detect Disease From Following Fruites And Veggies</p>
  </div>
</div>
<center class="laptop">
  <a href="/index">
  <button class = "bttn-pill bttn-lg btn-success"><b style="color:
black;">Flauralysis</b></button>
</a>
</center>
<div class="row p-5 laptop">
  <div class=" col">
    <div class="p-3 shadow rounded-lg" style="width: 90%;height: 92%; background-color:
#bdbdbdb3;">
      <img src="{{url_for('static', filename='/images/apple.webp')}} " width="250"
height="250">
      <center><p class="lead" style="font-weight: 500;color: black;">Apple</p></center>
    </div>
  </div>
  <div class="col">
    <div class="p-3 shadow rounded-lg" style="width: 90%;height: 92%; background-color:
#bdbdbdb3;">
      <img src="{{url_for('static', filename='/images/blueberries.png')}} " width="250"
height="250">
      <center><p class="lead" style="font-weight: 500;color:
black;">Blueberry</p></center>
    </div>
  </div>
  <div class=" col">
```

```
      <div class="p-3 shadow rounded-lg" style="width: 90%;height: 92%; background-color:
#bdbdbdb3;">
        <img src="{{url_for('static', filename='/images/cherry.webp')}} " width="250"
height="250">
        <center><p class="lead" style="font-weight: 500;color: black;">Cherry</p></center>
      </div>
    </div>
    <div class=" col">
      <div class="p-3 shadow rounded-lg" style="width: 90%;height: 92%; background-color:
#bdbdbdb3;">
        <img src="{{url_for('static', filename='/images/corn.jpg')}} " width="250"
height="250">
        <center><p class="lead" style="font-weight: 500;color: black;">Corn</p></center>
      </div>
    </div>
</div>
<div class="row p-5 laptop">
    <div class=" col">
      <div class="p-3 shadow rounded-lg" style="width: 90%;height: 92%; background-color:
#bdbdbdb3;">
        <img src="{{url_for('static', filename='/images/grapes.webp')}}" width="250"
height="250">
        <center><p class="lead" style="font-weight: 500;color: black;">Grape</p></center>
      </div>
    </div>
    <div class="col">
      <div class="p-3 shadow rounded-lg" style="width: 90%;height: 92%; background-color:
#bdbdbdb3;">
        <img src="{{url_for('static', filename='/images/orange.jpg')}} " width="250"
height="250">
        <center><p class="lead" style="font-weight: 500;color: black;">Orange</p></center>
      </div>
    </div>
```

```
  <div class=" col">
    <div class="p-3 shadow rounded-lg" style="width: 90%;height: 92%; background-color:
#bdbdbdb3;">
      <img src="{{url_for('static', filename='/images/peach.webp')}} "width="250"
height="250">
      <center><p class="lead" style="font-weight: 500;color: black;">Peach</p></center>
    </div>
  </div>
  <div class=" col">
    <div class="p-3 shadow rounded-lg" style="width: 90%;height: 92%; background-color:
#bdbdbdb3;">
      <img src="{{url_for('static', filename='/images/Bell_Peppers.jpg')}} " width="250"
height="250">
      <center><p class="lead" style="font-weight: 500;color: black;">Pepper
Bell</p></center>
    </div>
  </div>
</div>

<div class="row p-5 laptop">
  <div class=" col">
    <div class="p-3 shadow rounded-lg" style="width: 90%;height: 92%; background-color:
#bdbdbdb3;">
      <img src="{{url_for('static', filename='/images/potato.jpg')}} " width="250"
height="250">
      <center><p class="lead" style="font-weight: 500;color: black;">Potato</p></center>
    </div>
  </div>
  <div class="col">
    <div class="p-3 shadow rounded-lg" style="width: 90%;height: 92%; background-color:
#bdbdbdb3;">
      <img src="{{url_for('static', filename='/images/raspberries.webp')}} " width="250"
height="250">
```

```
        <center><p class="lead" style="font-weight: 500;color:
black;">Raspberry</p></center>
      </div>
    </div>
    <div class=" col">
      <div class="p-3 shadow rounded-lg" style="width: 90%;height: 92%; background-color:
#bdbdbdb3; ">
        <img src="{{url_for('static', filename='/images/soyabean.jpg')}} " width="250"
height="250">
        <center><p class="lead" style="font-weight: 500;color:
black;">Soybean</p></center>
      </div>
    </div>
    <div class=" col">
      <div class="p-3 shadow rounded-lg" style="width: 90%;height: 92%; background-color:
#bdbdbdb3;">
        <img src="{{url_for('static', filename='/images/squash.webp')}} " width="250"
height="250">
        <center><p class="lead" style="font-weight: 500;color: black;">Squash</p></center>
      </div>
    </div>
</div>
<center class="laptop">
<div class="row p-5">
    <div class=" col">
      <div class="p-3 shadow rounded-lg" style="width: 45%;height: 92%; background-color:
#bdbdbdb3;">
        <img src="{{url_for('static', filename='/images/strawberry.avif')}} " width="250"
height="250">
        <center><p class="lead" style="font-weight: 500;color:
black;">Strawberry</p></center>
      </div>
    </div>
```

```
  <div class="col">
    <div class="p-3 shadow rounded-lg" style="width: 45%;height: 92%; background-color:
#bdbdbdb3;">
      <img src="{{url_for('static', filename='/images/tomatoes.jpg')}}" width="250"
height="250">
      <center><p class="lead" style="font-weight: 500;color: black;">Tomato</p></center>
    </div>
  </div>
</div>
</center>
{% endblock body %}

[17:35, 05/11/2023] Manas Dhaketa Acro: <!doctype html>
<html lang="en">
<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <!-- Bootstrap CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/css/bootstrap.min.css" rel="stylesheet"
    integrity="sha384-
eOJMYsd53ii+scO/bJGFsiCZc+5NDVN2yr8+0RDqr0Ql0h+rP48ckxlpbzKgwra6"
crossorigin="anonymous">
  <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/ionicons/2.0.1/css/ionicons.min.css">

  <link rel = "stylesheet" href =
"https://cdnjs.cloudflare.com/ajax/libs/bttn.css/0.2.4/bttn.css">
  <title> {% block pagetitle %}
    {% endblock pagetitle %}</title>
  <style>
    .bg {
```

```
    background-image: url("images/background.jpeg");
    height: 100%;
    /* Center and scale the image nicely */
    background-position: center;
    background-repeat: no-repeat;
    background-size: cover;
}
.file-upload input[type='file'] {
    display: none;
}
body {
    background-color: #0f711aae;
    /* background: -webkit-linear-gradient(to right, #66a776, #043610);
    background: linear-gradient(to right, #5d9c6a, #059b0d); */
    height: 100vh;
}
.rounded-lg {
    border-radius: 1rem;
}
.custom-file-label.rounded-pill {
    border-radius: 50rem;
}
.custom-file-label.rounded-pill::after {
    border-radius: 0 50rem 50rem 0;
}
label {
    background-color: rgb(2, 46, 15);
    color: white;
    padding: 0.5rem;
    font-family: sans-serif;
    border-radius: 0.3rem;
    cursor: pointer;
    margin-top: 1rem;
```

```
}
#file-chosen {
   margin-left: 0.3rem;
   font-family: sans-serif;
}
.footer-basic {
   padding: 40px 0;
   background-color: #343434a7;
   color: #f1f3f5;
}
.footer-basic ul {
   padding: 0;
   list-style: none;
   text-align: center;
   font-size: 18px;
   line-height: 1.6;
   margin-bottom: 0;
}
.footer-basic li {
   padding: 0 10px;
}
.footer-basic ul a {
   color: inherit;
   text-decoration: none;
   opacity: 0.8;
}
.footer-basic ul a:hover {
   opacity: 1;
}
.footer-basic .social {
   text-align: center;
   padding-bottom: 25px;
}
```

```
      .footer-basic .social>a {
        font-size: 24px;
        width: 40px;
        height: 40px;
        line-height: 40px;
        display: inline-block;
        text-align: center;
        border-radius: 50%;
        border: 1px solid #ccc;
        margin: 0 8px;
        color: inherit;
        opacity: 0.75;
      }
      .footer-basic .social>a:hover {
        opacity: 0.9;
      }
      .footer-basic .copyright {
        margin-top: 15px;
        text-align: center;
        font-size: 13px;
        color: #aaa;
        margin-bottom: 0;
      }
    </style>
</head>
<body>
    <nav class="navbar navbar-expand-lg navbar-dark " style="background-color:
#343434a7;">
      <!-- <a class="navbar-brand" href="#">Navbar</a>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarNavDropdown" aria-controls="navbarNavDropdown" aria-
expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
```

```
      </button> -->
      <div class="collapse navbar-collapse justify-content-...
[17:35, 05/11/2023] Manas Dhaketa Acro: {% extends 'base.html' %}
{% block pagetitle %}
{{title}}
{% endblock pagetitle %}
{% block body %}
<div>
 <div class="container">
  <!-- For demo purpose -->
  <div class="row mb-5 text-center text-white">
   <div class="col-lg-10 mx-auto">
    <h1 class="display-4" style="padding-top: 2%;font-weight: 400;color: rgb(255, 255,
255);""><b>{{title}} 🍃</b></h1>
   </div>
  </div>
  <center>
  <div class=" col">
      <div class="p-3  shadow rounded-lg" style="width: 30%; background-color:
#bdbdbdb3;">
       <img src={{image_url}} width="350" height="350">
      </div>
   </div>
   </center>
   <br>
   <div class=" row ">
    <div class="col mx-auto">
      <div class="p-5  shadow rounded-lg" style="height: 95%; background-color:
#bdbdbdb3 ;">
       <h5><b>
          {% if pred==3 or pred==5 or pred==7 or pred==11 or pred==15 or pred==18 or
pred==20 or pred==23 or
          pred==24 or pred==25 or pred==28 or pred==38 %}
```

```
            Tips to Grow Healthy Plants :
            {% else %}
            Brief Descritpion :
            {% endif %}
          </b></h5>
        <p>{{desc}}
        </p>
       </div>
      </div>
    </div>
    <div class="row">
     <div class="col mx-auto">
      <div class="p-5 shadow rounded-lg" style="height:95% ;background-color:
#bdbdbdb3;">
        <h5><b>
          {% if pred==3 or pred==5 or pred==7 or pred==11 or pred==15 or pred==18 or
pred==20 or pred==23 or
          pred==24 or pred==25 or pred==28 or pred==38 %}
          Benefits :
          {% else %}
          Prevent This Plant Disease By follow below steps :
          {% endif %}
         </b></h5>
        <p>
         {{prevent}}
        </p>
       </div>
      </div>
      {% if pred!=4 %}
      <div class="col mx-auto"">
      <div class=" p-5  shadow rounded-lg" style="height: 95%; background-
color:#bdbdbdb3;">
       <center>
```

```
        <h5><b>
          {% if pred==3 or pred==5 or pred==7 or pred==11 or pred==15 or pred==18 or
pred==20 or pred==23 or
          pred==24 or pred==25 or pred==28 or pred==38 %}
          Fertilizer :
          {% else %}
          Supplements :
          {% endif %}
         </b></h5>
        <br>
        <img src={{simage}} width="300" height="350">
        <br>
        <br>
        <h6>{{sname}}</h6>
        <a target="_blank" href={{buy_link}}><button type="button" class="btn btn-success"
          style="background-color: #05380b;">Buy Product</button></a>
      </center>
    </div>
   </div>
   {% endif %}
  </div>
 </div>
</div>
{% endblock body %}
```