

PLANT PULSE

A DESIGN PROJECT REPORT

submitted by

MADHUVADHANI S

RAKSHANA G

RANJANI S

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

in

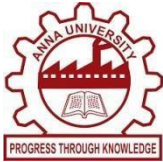
COMPUTER SCIENCE AND ENGINEERING

K RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai, Approved by AICTE, New Delhi)

Samayapuram – 621 112

NOVEMBER - DECEMBER, 2024



PLANT PULSE



A DESIGN PROJECT REPORT

submitted by

MADHUVADHANI S (811722104085)

RAKSHANA G (811722104119)

RANJANI S (811722104121)

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

K RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai, Approved by AICTE, New Delhi)

Samayapuram – 621 112

NOVEMBER - DECEMBER, 2024

K RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(AUTONOMOUS)

SAMAYAPURAM-621112

BONAFIDE CERTIFICATE

Certified that this project report titled “**PLANTPULSE**” is bonafide work of **MADHUVADHANI S (811722104085), RAKSHANA G (811722104119), RANJANI S (811722104121)** who carried out the project under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr. A Delphin Carolina Rani M.E., Ph.D.,

HEAD OF THE DEPARTMENT

PROFESSOR

Department of CSE

K Ramakrishnan College of Technology

(Autonomous)

Samayapuram-621 112

SIGNATURE

Mrs.A. Thenmozhi M.E.,

SUPERVISOR

Assistant Professor

Department of CSE

K Ramakrishnan College of Technology

(Autonomous)

Samayapuram-621 112

Submitted for the viva-voice examination held on.....

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION

We jointly declare that the project report on “**PLANT PULSE**” is the result of original work done by us and best of our knowledge, similar work has not been submitted to “**ANNA UNIVERSITY CHENNAI**” for the requirement of Degree of Bachelor of Engineering. This project report is submitted on the partial fulfilment of the requirement of the award of Degree of Bachelor of Engineering.

Signature

MADHUVADHANI S

RAKSHANA G

RANJANI S

Place: Samayapuram

Date :

ACKNOWLEDGEMENT

It is with great pride that we express our gratitude and indebtedness to our institution **“K RAMAKRISHNAN COLLEGE OF TECHNOLOGY”**, for providing us with the opportunity to do this project.

We are glad to credit honorable chairman **Dr. K RAMAKRISHNAN, B.E.**, for having provided for the facilities during the course of our study in college.

We would like to express our sincere thanks to our beloved Executive Director **Dr. S KUPPUSAMY, MBA, Ph.D.**, for forwarding our project and offering adequate duration to complete it.

We would like to thank **Dr. N VASUDEVAN, M.Tech., Ph.D.**, Principal, who gave opportunity to frame the project with full satisfaction.

We whole heartily thank **Dr. A DELPHIN CAROLINA RANI, M.E., Ph.D.**, Head of the Department, **COMPUTER SCIENCE AND ENGINEERING** for providing her support to pursue this project.

We express our deep and sincere gratitude and thanks to our project guide **Mrs. A.THENMOZHI, M.E.**, Department of **COMPUTER SCIENCE AND ENGINEERING**, for her incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

We render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course. We wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

ABSTRACT

Plant Pulse is an advanced agricultural technology that utilizes machine learning (ML) and deep learning (DL) to revolutionize leaf disease detection. This innovative system leverages image processing techniques to analyze and classify plant leaf diseases with high accuracy and efficiency. By providing a hands-free, real-time solution, Plant Pulse eliminates the need for traditional manual disease inspection, offering a scalable and intuitive interface that caters to both large-scale farmers and small agricultural setups. The technology is particularly beneficial for improving crop health, boosting agricultural productivity, and ensuring global food security. Powered by frameworks like TensorFlow, Keras, and Streamlit, Plant Pulse combines user-friendly design with cutting-edge algorithms to deliver precise and actionable insights. It supports sustainable farming practices, enabling early detection and treatment of diseases, thus minimizing crop losses. As Plant Pulse continues to evolve, it stands as a transformative tool for modern agriculture, making farming more efficient, sustainable, and accessible across the globe.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO.
	ABSTRACT	v
	LIST OF FIGURES	ix
	LIST OF ABBREVIATIONS	x
1	INTRODUCTION	1
	1.1 Background	1
	1.2 Overview	2
	1.3 Problem Statement	2
	1.4 Objective	3
	1.5 Implications	3
2	LITERATURE SURVEY	4
3	SYSTEM ANALYSIS	7
	3.1 Existing System	7
	3.2 Proposed System	8
	3.3 Block Diagram for Proposed System	9
	3.4 FlowChart	10
	3.5 Process Cycle	11
	3.6 Activity Diagram	12

4	MODULES	13
	4.1 Module Description	13
	4.1.1 Data Collection and Preprocessing	13
	4.1.2 Model Development	14
	4.1.3 User Interface	14
	4.1.4 Ethical Considerations	15
	4.1.5 Deployment and Sustainability	15
5	SYSTEM SPECIFICATION	16
	5.1 Hardware Requirements	16
	5.2 Software Requirements	16
	5.2.1 PYTHON	16
	5.2.2 Libraries and Framework	17
	5.2.2.1 Tensorflow	17
	5.2.2.2 Keras	17
	5.2.2.3 OpenCV	18
	5.3 VS Code	21
	5.4 Streamlit	22
6	METHODOLOGY	23
	6.1 Convolutional Neural Network Implementation	23
	6.2 Image Preprocessing and Augmentation	24
	6.3 Leaf Segmentation and ROI Extraction	24

6.4 Data Collection and Labelling	25
6.5 Model Evaluation and Optimization	25
6.6 Deployment Using Streamlit	26
7 CONCLUSION AND FUTURE ENHANCEMENT	27
7.1 Conclusion	27
7.2 Future Enhancement	27
APPENDIX-1	29
APPENDIX-2	35
REFERENCE	37

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO
1.1	Flow of visual analysis control	1
3.1	Block diagram – existing system	7
3.2	Usecase diagram	9
3.3	Flowchart	10
3.4	Process cycle of Plant Pulse	11
3.5	Activity Diagram of Plant Pulse	12
4.1	DataSet	14
5.1	Intensity Diagram	19

LIST OF ABBREVIATIONS

ABBREVIATION	FULL FORM
AI	Artificial Intelligence
CNN	Convolutional Neural Network
DL	Deep Learning
CV	Computer vision
HCI	Human Computer Interaction
IoT	Internet of Things
ML	Machine Learning
VR	Virtual Reality
RNN	Recurrent Neural Network
NLP	Natural Language Processing
RGB	Red, Green, Blue (color model for images)
API	Application Programming Interface
UI	User Interface
OpenCV	Open-Source Computer Vision Library
F1-Score	Harmonic Mean of Precision and Recall

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Early advancements in machine learning and computer vision laid the foundation for image-based analysis systems. With the integration of deep learning algorithms, it became possible to accurately classify and analyze visual data, revolutionizing various fields. The application of AI in image recognition has been enhanced through innovations in neural networks and feature extraction methods. Advancements in frameworks such as TensorFlow and Keras have significantly improved the accuracy, scalability, and efficiency of these systems.

Over the years, the feasibility of automated detection systems has been demonstrated across diverse industries, including agriculture, healthcare, and environmental monitoring. The use of real-time processing techniques and interactive interfaces has further streamlined the implementation of these technologies. The growing need for sustainable solutions and precision tools continues to drive innovation in AI-based analysis systems. Looking ahead, the potential of AI to improve efficiency, reduce manual intervention, and promote data-driven decision-making is vast and transformative.

Training Model	Pre-Processing	Feature Extraction	Recognition	Execution
-----------------------	-----------------------	---------------------------	--------------------	------------------

Figure 1.1: Flow of Visual Analysis Control

1.2 OVERVIEW

Farmers and agricultural workers often rely on manual inspection to monitor crop health, which can be time-consuming and prone to error. Advanced technologies now offer solutions that make this process more efficient and accurate. The proposed system uses a deep learning-based approach for real-time leaf disease detection and classification, making it an essential tool for modern agriculture.

Leveraging machine learning and image processing, the system analyzes plant leaf images to identify diseases. Using readily available hardware like cameras or smartphones, the system captures images of leaves, processes them, and provides detailed disease diagnoses. Recent advancements in AI and neural networks enable this system to achieve high accuracy and adaptability.

This technology reduces the reliance on manual observation, enabling farmers to quickly identify potential crop issues and take corrective action. The system's user-friendly interface allows seamless interaction, while its scalability ensures applicability across diverse agricultural settings. By supporting early detection and timely intervention, the solution promotes healthier crops, boosts productivity, and contributes to sustainable farming practices. Incorporating state-of-the-art tools and frameworks like TensorFlow and Streamlit, the system offers precision and efficiency. It eliminates the need for extensive technical expertise, making it accessible to a broad user base. This innovative approach to disease detection is set to transform agriculture by providing data-driven insights and promoting food security worldwide.

1.3 PROBLEM STATEMENT

With the emergence of ubiquitous computing, traditional methods of user interaction involving the keyboard, mouse, and etc are no longer adequate. The limitations of these devices restrict the range of instructions that can be executed. Direct usage of eye gestures and voice commands have the potential to serve as input devices for more natural and intuitive interaction, enabling users to perform everyday tasks with ease. Such methods can offer a more extensive instruction set and eliminate the need for direct physical contact with the computer, further enhancing the users experience.

1.4 OBJECTIVE

The primary objective is to enhance agricultural productivity by streamlining the process of leaf disease detection. The project aims to provide a real-time, scalable, and accessible solution that reduces the reliance on manual methods. Additionally, it emphasizes the importance of ethical data handling by ensuring transparency, consent, and compliance with data privacy regulations throughout its development and deployment.

1.5 IMPLICATION

Leaf disease detection leverages the power of image recognition and machine learning to analyze and interpret visual patterns on plant leaves. By utilizing cameras and advanced algorithms, the system captures, processes, and classifies leaf images to identify potential diseases. This technology translates visual cues, such as discolorations, spots, or abnormalities, into actionable insights, enabling precise disease diagnosis. Such an approach offers a more efficient, accurate, and scalable alternative to traditional methods, promoting timely intervention and enhancing crop health management. This innovation fosters a more sustainable agricultural practice while empowering users with data-driven insights for better decision-making.

CHAPTER 2

LITERATURE SURVEY

TITLE : Leaf Disease Detection Using Deep Learning
AUTHORS : Teenu Sahasra M, Sai Kumari S, Sai Meghana S, P. Rama Devi
YEAR : 2021

Leaf Disease Detection Using Deep Learning explores the application of advanced deep learning techniques to identify and classify diseases in plant leaves. The paper highlights the significance of early and accurate disease detection in improving agricultural productivity and reducing losses. Using Convolutional Neural Networks (CNNs) as the core methodology, the study demonstrates how automated systems can analyze images of plant leaves to detect disease symptoms with high precision. The authors emphasize the importance of creating a robust dataset and employing preprocessing techniques to enhance model performance. Their findings underscore the potential of deep learning to revolutionize plant health monitoring, offering scalable and efficient solutions for farmers and agricultural professional.

**TITLE : Plant Disease Prediction Using Deep Learning Techniques:
A Review**
AUTHORS : Hughes, D. P., & Salathé, M.
YEAR : 2019

This review paper summarizes the state of the art in deep learning for plant disease prediction, discussing the methodologies used, types of plant diseases commonly targeted, and datasets used in training the models. The paper also examines the real-world applicability of deep learning models in agricultural environments, emphasizing the use of CNNs for disease prediction tasks. Key challenges discussed include the acquisition of diverse and high-quality datasets, dealing with imbalanced classes and the computational challenges involved in training large-scale deep learning models. The paper also emphasizes the potential of integrating multiple data types to improve model accuracy and prediction.

TITLE : Deep Learning Models for Plant Disease Detection and Diagnosis
AUTHORS : Konstantinos P. Ferentinos
YEAR : 2018

This paper investigates the use of deep learning methodologies to identify and diagnose plant diseases through image analysis. The paper focuses on the effectiveness of Convolutional Neural Networks (CNNs) in classifying plant diseases from leaf images across various crops. Using a large dataset of annotated images, the study evaluates different CNN architectures, such as AlexNet, VGG, and GoogleNet, to assess their performance in terms of accuracy and computational efficiency. Ferentinos highlights the scalability of these models for real-world applications and discusses their potential in enabling precision agriculture. The research underscores the transformative role of deep learning in reducing dependency on manual inspections, improving early disease detection, and promoting sustainable farming practices.

TITLE : Deep Learning
AUTHORS : Ian Goodfellow, Yoshua Bengio, and Aaron Courville
YEAR : 2016

This seminal work "**Deep Learning**" by Ian Goodfellow, Yoshua Bengio, and Aaron Courville serves as a comprehensive guide to the principles, techniques, and applications of deep learning. This foundational text delves into the theoretical underpinnings of deep neural networks, offering insights into key topics such as gradient descent optimization, backpropagation, and regularization methods. It explores a wide array with applications spanning computer vision, natural language processing, and reinforcement learning. The book emphasizes the synergy between theoretical concepts and practical implementations, making it a valuable resource for both researchers and practitioners. By addressing challenges like overfitting, computational scalability, and the interpretability of deep models, it underscores the profound impact of deep learning in shaping artificial intelligence and its applications in diverse domains.

TITLE : Frontiers in Plant Science
AUTHORS : Mohanty S.P, Hughes D.P
YEAR : 2016

This article "**Frontiers in Plant Science**" by Hughes and Mohanty explores the transformative role of advanced technologies in addressing critical challenges in plant science. The authors focus on the integration of cutting-edge tools such as deep learning, machine vision, and genomic data analysis to improve crop productivity, disease management, and sustainable agriculture practices. Emphasizing the potential of computational advancements, the paper highlights their application in phenotyping, stress detection, and precision farming. The authors discuss the challenges associated with data acquisition and interpretation in plant science while presenting solutions such as high-throughput platforms and AI-driven analytics. This work underscores the importance of interdisciplinary approaches in enhancing our understanding of plant biology and advancing global agricultural sustainability.

TITLE : Application of Deep Learning in Plant Disease Detection and Classification
AUTHORS : Zhang, J., & Zhao, J.
YEAR : 2020

This article investigates the growing role of deep learning in plant disease detection, focusing on how deep learning methods are being applied to detect diseases in crops using image datasets. The authors review various neural network architectures, including CNNs, and their applications for classification tasks. The paper also discusses the challenges of large-scale data collection and the need for high-quality annotated datasets to train these deep learning models effectively. Additionally, the authors present several case studies where deep learning techniques have been implemented in detecting diseases in crops such as wheat, rice, and tomatoes. The performance metrics, including accuracy and sensitivity, are also analyzed.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

There are several existing systems and technologies for leaf disease detection, which are implemented with multiple algorithm

- **K-Nearest Neighbors (KNN):**

Classifies leaf diseases by measuring similarity to training images based on features like color and texture. It's simple but computationally expensive and less effective for complex patterns.

- **Support Vector Machine (SVM):**

Separates disease categories using hyperplanes based on extracted features. It's effective but requires manual feature engineering and may struggle with non-linear data.

- **Decision Trees:**

A rule-based classifier that uses extracted features like color and texture to build a tree structure. While simple to understand, it can overfit and may not perform well on complex datasets.

- **K-Means Clustering:**

An unsupervised algorithm that groups similar patterns in leaf images. It is used for segmentation but isn't ideal for precise disease classification, as it lacks supervision.

- **Image Processing Techniques:**

Methods like histogram thresholding and edge detection identify diseased areas by analyzing color intensity and boundary patterns. These are simple but limited in detecting complex or overlapping symptoms.

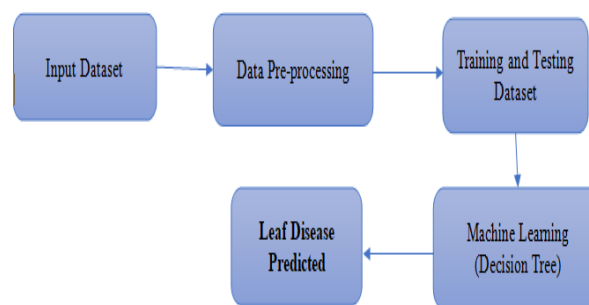


Fig: 3.1 Block Diagram – Existing System

3.2 PROPOSED SYSTEM

To design a leaf disease detection system, Plant Pulse uses advanced machine learning (ML) and deep learning (DL) techniques to identify and classify plant diseases based on leaf images. The system analyzes leaf conditions without the need for physical sensors, and can be deployed on standard computers or mobile devices.

The system is designed to be **user-friendly** and **accessible**, making it ideal for farmers, agricultural experts, and researchers. By leveraging the power of **Convolutional Neural Networks (CNNs)** and pre-trained models, Plant Pulse can accurately detect various diseases in plants through image analysis. For image processing and disease classification tasks, **OpenCV** is employed for image preprocessing, and **TensorFlow** and **Keras** are used for building and training deep learning models. The **Streamlit** framework is utilized to develop a simple graphical user interface (GUI), allowing users to easily upload leaf images, view the diagnosis, and receive disease information. The system offers **affordable**, **efficient**, and **real-time** disease detection solutions that can be accessed from anywhere, enabling quicker interventions and enhancing agricultural productivity. By analyzing and classifying diseases in plants, **Plant Pulse** helps to improve plant health management and supports **sustainable agriculture** practices for better food security.

3.3 BLOCK DIAGRAM OF PROPOSED SYSTEM

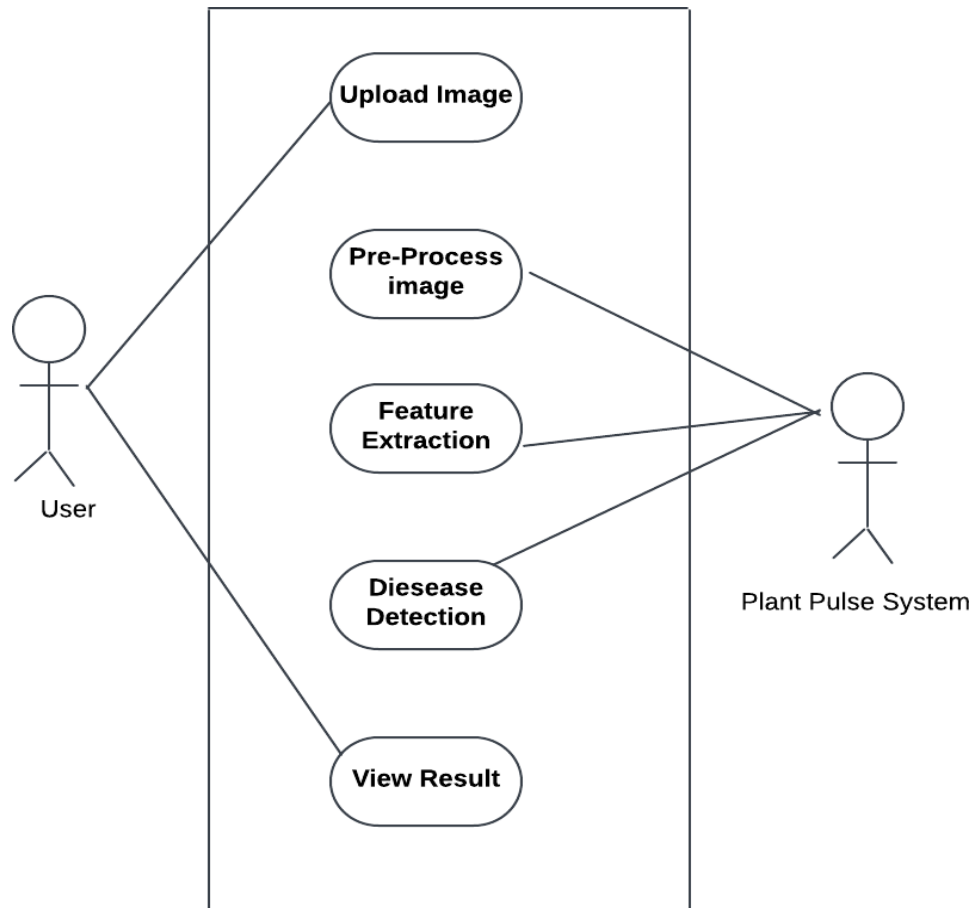


Fig 3.2: Usecase Diagram

3.4 FLOWCHART

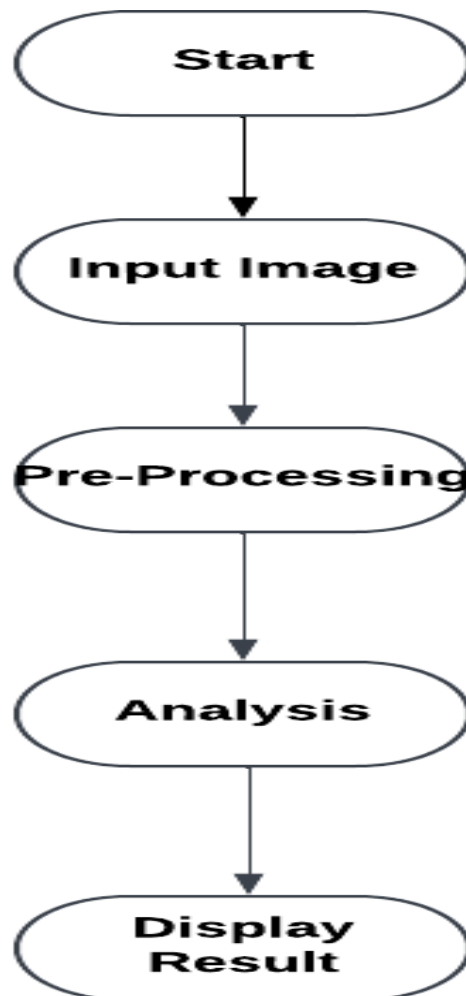


Fig: 3.3 FlowChart

3.5 PROCESS CYCLE

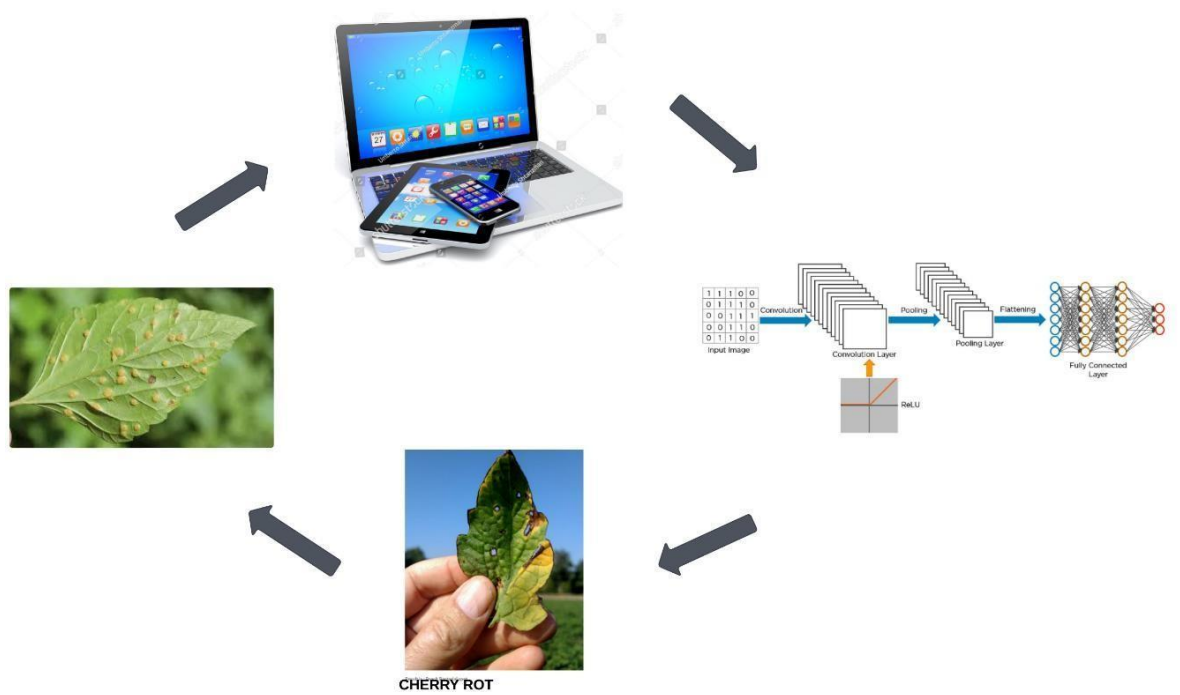


Fig 3.4 : Process Cycle of Plant Pulse

3.6 ACTIVITY DIAGRAM

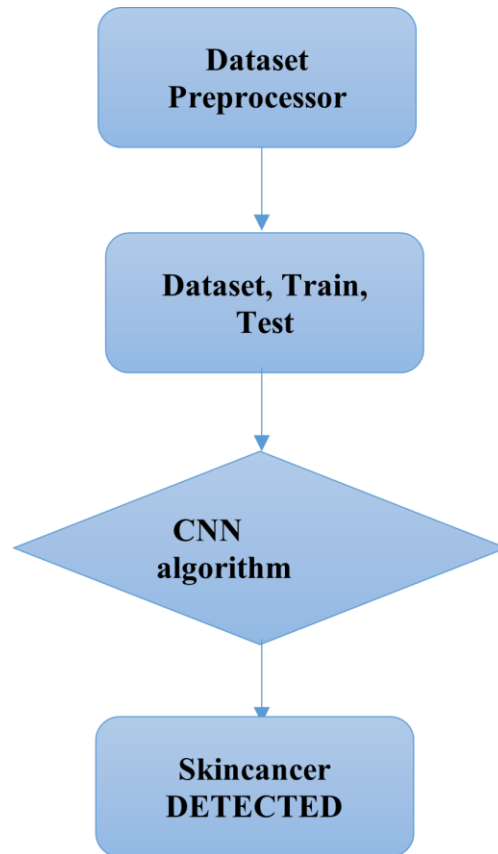


Fig: 3.5 Activity diagram of Plant Pulse

CHAPTER 4

MODULES

4.1 MODULE DESCRIPTION

The module aims to detect and classify the leaf disease, using a Convolutional Neural Network (CNN). Done by designing a CNN architecture suitable for image classification tasks. Common architectures include variations of Convolutional layers, MaxPooling layers, and fully connected layers. Metrics such as accuracy, precision, and F1 score can be used to evaluate the model's efficiency.

- Data Collection and Preprocessing
- Model Development
- User Interface
- Ethical Considerations
- Deployment and Sustainability

4.1.1 Data Collection and Preprocessing

In the project, we gathered every image that makes a appear to be diseased leaf. This is the project's most crucial step. Therefore, all of the visuals that we see come from real-time or recorded CCTV footage.

After gathering all the images, pre-processing is required. Thus not all images can convey information clearly. So that we may prepare the images by renaming, resizing, and labelling them. Once the procedure is complete, we can use the photos to train our deep learning model.

Data Augmentation: Augmenting the dataset by applying transformations like rotation, flipping, and zooming helps improve model generalization.

Normalization: Ensuring consistency in pixel values across images by scaling them to a standard range (e.g., [0, 1]) helps in faster convergence during training.

Resizing: Resizing images to a consistent resolution reduces computational load and ensures uniformity in input dimensions for deep learning models.

4.1.2 Model Development

During the training of CNN, the neural network is being fed with a large dataset of images being labelled with their corresponding class labels. The CNN network processes each image with its values being assigned randomly and then make comparisons with the class label of the input image.

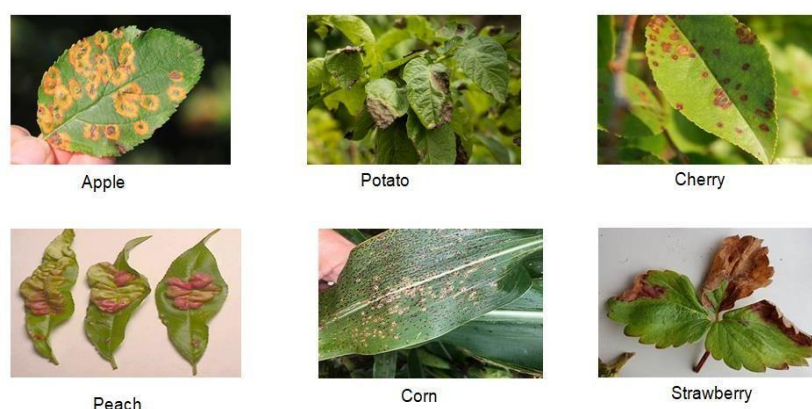


Fig 4.1 : DataSet (diseased leaves)

4.1.3 User Interface

In the User Interface (UI) module of the Plant Pulse project, the focus is on developing a simple and intuitive graphical user interface (GUI) that allows users to easily interact with the system. Using the Streamlit framework, the interface enables users to upload images of plant leaves directly into the system for analysis. Once an image is uploaded, the model processes the input, and the UI displays the diagnosis, including the identified disease and its severity. The UI is designed to be user-friendly, ensuring accessibility for farmers, agricultural experts, and researchers with varying levels of technical expertise. It also provides clear visual feedback and information, making it easy for users to understand the results and take necessary actions to manage plant health effectively.

4.1.4 Ethical Considerations

Focuses is on ensuring the responsible and ethical use of data and technology. The module emphasizes the importance of obtaining consent from users before collecting any plant image data, ensuring privacy and confidentiality. It also ensures that the collected data is used solely for the intended purpose of detecting and diagnosing plant diseases. Additionally, the system is designed to adhere to data privacy regulations, safeguarding users' personal information and any data related to plant health. The ethical considerations module promotes transparency, fairness, and accountability in the development and deployment of the **Plant Pulse** system, aiming to build trust with users while contributing to sustainable agricultural practices.

4.1.5 Deployment and Sustainability

This module focuses on ensuring the long-term viability and scalability of the system. The module aims to create a robust and efficient deep learning model that can be continuously improved over time as more data becomes available. It includes developing a flexible infrastructure that supports future updates and adaptation to different agricultural contexts globally. Additionally, the project emphasizes sustainability by using eco-friendly technologies and promoting its potential to enhance agricultural productivity in a way that supports global food security. The system is designed to be scalable, ensuring it can be deployed in both small-scale farms and large agricultural settings, with minimal resources and infrastructure. This module also highlights the importance of making the system cost-effective and accessible to farmers, ensuring that it has a positive long-term impact on sustainable agriculture.

CHAPTER 5

SYSTEM SPECIFICATION

5.1 HARDWARE REQUIREMENTS

- Processor – Intel corei5
- RAM – 8GB or Higher.
- Storage – 152GB or High

5.2 SOFTWARE REQUIREMENTS

- Programming Language : Python 3.7 or above
- Libraries & Frameworks : Tensorflow, Keras, Pandas, Open CV
- Development Environment : VS Code
- GUI Framework : Streamlit

5.2.1 PYTHON

- Python can be used on a server to create web applications.
- Python can be used on side software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle bigdata and perform complex mathematics.

5.2.2 LIBRARIES & FRAMEWORKS

5.2.2.1 TENSORFLOW



TensorFlow is an open-source Python library developed by Google for numerical computation and machine learning. It provides a flexible and comprehensive ecosystem of tools, libraries, and community resources to build, train, and deploy machine learning models. TensorFlow supports a wide range of applications, from deep learning to neural networks, and can efficiently run on both CPUs and GPUs. Its core functionality is based on data flow graphs, where nodes represent mathematical operations, and edges represent multi-dimensional data arrays (tensors). TensorFlow is widely used in research, industry, and production systems due to its scalability, performance, and ease of use in complex machine learning tasks.

5.2.2.2 KERAS

Keras is an open-source high-level neural networks API written in Python, designed for building and training deep learning models. Initially developed as an interface for TensorFlow, it is now tightly integrated with TensorFlow as its official high-level API. Keras simplifies the process of designing and experimenting with deep learning models by providing an easy-to-use interface for defining complex neural networks with just a few lines of code. It supports both convolutional networks (CNNs) and recurrent networks (RNNs), along with combinations of the two. Keras is known for its modularity, extensibility, and user-friendly approach, making it suitable for both beginners and experienced researchers in machine learning and artificial intelligence.

5.2.2.3 OPENCV



OpenCV is a Python open-source library, which is used for computer vision in Artificial intelligence, Machine Learning, face recognition, etc.

In OpenCV, the CV is an abbreviation form of a computer vision, which is defined as a field of study that helps computers to understand the content of the digital images such as photographs and videos.

The purpose of computer vision is to understand the content of the images. It extracts the description from the pictures, which may be an object, a text description, and three-dimension model, and so on. For example, cars can be facilitated with computer vision, which will be able to identify and different objects around the road, such as traffic lights, pedestrians, traffic signs, and so on, and acts accordingly.

Computer vision allows the computer to perform the same kind of tasks as humans with the same efficiency. There are a two main task which are defined below:

- **Object Classification** - In the object classification, we train a model on a dataset of particular objects, and the model classifies new objects as belonging to one or more of your training categories.
- **Object Identification** - In the object identification, our model will identify a particular instance of an object - for example, parsing two faces in an image and tagging one as ViratKohli and other one as Rohit Sharma.

RECOGNIZATION OF IMAGE IN COMPUTER:-

Human eyes provide lots of information based on what they see. Machines are facilitated with seeing everything, convert the vision into numbers and store in the memory. Here the question arises how computer convert images into numbers. So the answer is that the pixel value is used to convert images into numbers. A pixel is the smallest unit of a digital image or graphics that can be displayed and represented on a digital display device.

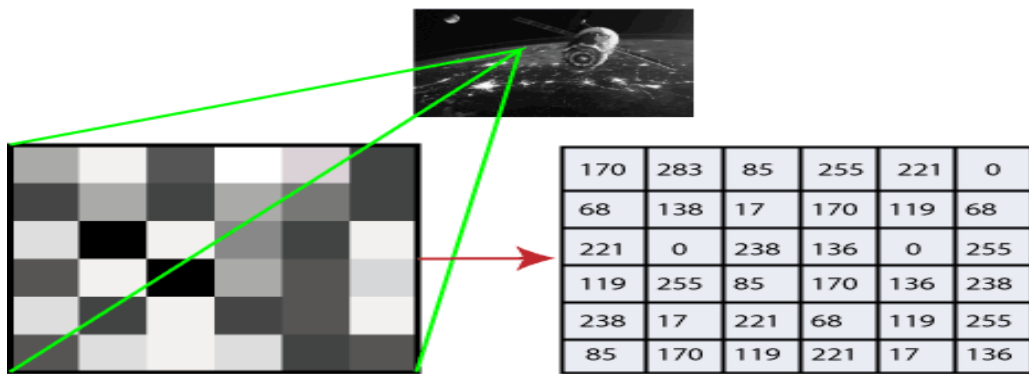


Fig.5.1 INTENSITY DIAGRAM

The picture intensity at the particular location is represented by the numbers. In the above image, we have shown the pixel values for a grayscale image consist of only one value, the intensity of the black color at that location.

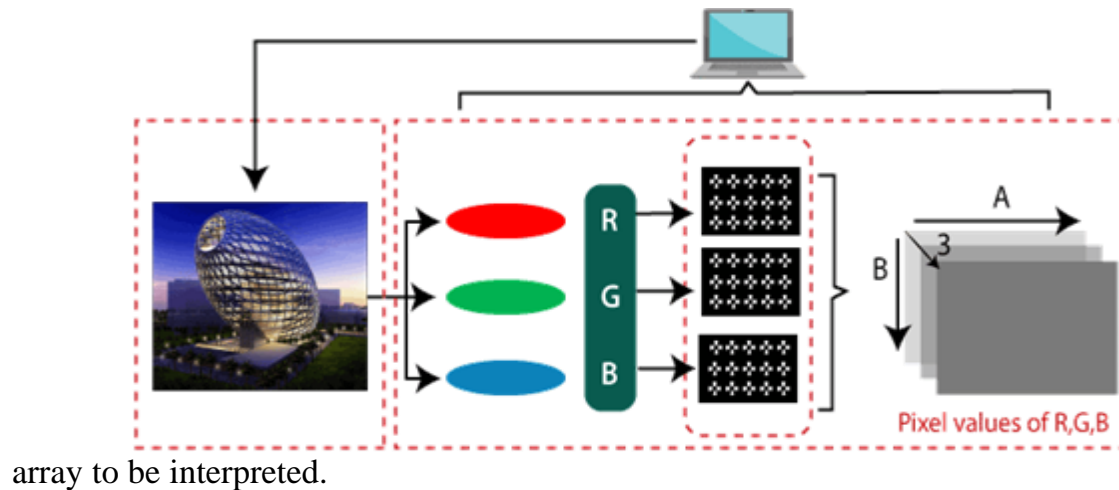
There are two common ways to identify the images:

1. Grayscale

Grayscale images are those images which contain only two colors black and white. The contrast measurement of intensity is black treated as the weakest intensity, and white as the strongest intensity. When we use the grayscale image, the computer assigns each pixel value based on its level of darkness.

2. RGB

An RGB is a combination of the red, green, blue color which together makes a new color. The computer retrieves that value from each pixel and puts the results in an



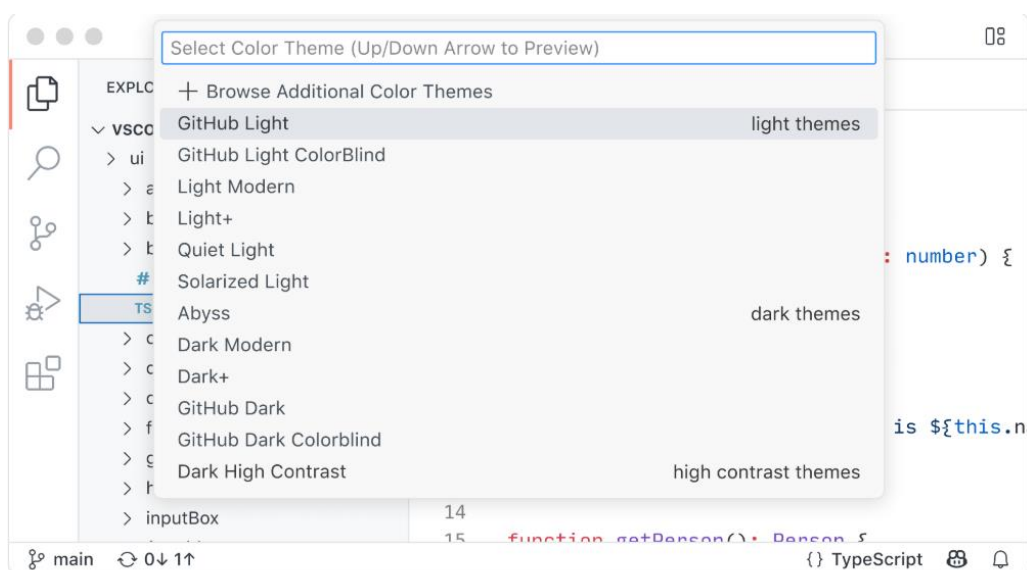
USAGE OF OPENCV :-

- OpenCV is available for free of cost.
- Since the OpenCV library is written in C/C++, so it is quite fast. Now it can be used with Python.
- It requires less RAM to use, it may be of 60-70 MB.
- Computer Vision is portable as OpenCV and can run on any device that can run on C.

5.3 VS CODE



Visual Studio Code (VS Code) is a free, open-source, and lightweight code editor developed by Microsoft. It is highly customizable, supporting a wide range of programming languages through extensions, including Python, JavaScript, C++, and more. VS Code offers powerful features like syntax highlighting, code completion, debugging tools, Git integration, and a rich ecosystem of plugins, making it suitable for both novice and experienced developers. Its performance is optimized for speed and responsiveness, and it provides an integrated terminal and version control tools, enhancing productivity in software development. With a strong community and continuous updates, VS Code has become one of the most popular code editors in the developer community.



5.4 STREAMLIT



Streamlit is an open-source Python library that enables the rapid creation of interactive web applications for data science and machine learning projects. It allows developers to build and deploy custom user interfaces (UIs) with minimal effort and without requiring extensive knowledge of front-end web technologies like HTML, CSS, or JavaScript. Streamlit's simplicity is reflected in its intuitive API, where users can create widgets, visualizations, and interactive controls using just Python code. It automatically updates the UI in response to changes in data or user inputs, making it ideal for creating real-time dashboards, prototypes, and data exploration tools. Streamlit is widely appreciated for its speed, ease of use, and seamless integration with popular data science libraries such as Pandas, Matplotlib, and Plotly.

CHAPTER 6

METHODOLOGY

6.1 CONVOLUTIONAL NEURAL NETWORK IMPLEMENTATION

CNN forms an integral part of deep learning as is applied to train data without using any image processing technique. In this experimental work, a new directory is created for each disease. Thirty three diseases are read to generate 8,150 image frames. The method to preprocess plant leaf images involves converting the image frames from RGB color to grayscale for simplified analysis and resizing them to 224×224 pixels to match the input requirements of deep learning models. Each plant leaf dataset contains a varying number of image frames, typically ranging between 3,000 to 4,000 frames per category, ensuring comprehensive training and evaluation.

In this project, the dataset of plant leaf images is divided into training and testing sets, with 70% (approximately 23 images) used for training and the remaining 30% (10 images) for testing. The architecture of the deep learning model consists of multiple layers, each with specific functionality:

- **Image Input Layer:** Size [224,224,3] to match the resized image dimensions and RGB channels.
- **Convolution 2D Layer:** Filter size [3,32] to extract features.
- **Rectified Linear Unit (ReLU Layer):** Activates non-linear transformations.
- **Max Pooling 2D Layer:** Pool size [2,2] for down sampling.
- **Fully Connected Layer:** Automatically determines the required size to link extracted features to output classes.
- **Softmax Layer:** Computes probabilities for each disease class.
- **Classification Output Layer:** Determines the final classification output.

The model is trained with hyperparameters configured for optimal performance. The number of epochs is set to 50, ensuring sufficient training iterations to fine-tune the weights and biases. This topology effectively supports the classification of plant leaf diseases with high accuracy.

6.2. IMAGE PREPROCESSING AND AUGMENTATION

In the preprocessing and augmentation stage of Plant Pulse, various transformations are applied to enhance the dataset and ensure the model's robustness in detecting leaf diseases. Images are resized to a consistent dimension of 224x224 pixels to standardize input for the deep learning model. The resizing ensures that the model can process data efficiently, regardless of the original image size. Augmentation techniques such as rotation, flipping, zooming, and color adjustments are employed to create variations of the original images. These transformations mimic real-world conditions, such as varying lighting, angles, and orientations, thereby increasing the diversity of the training data. For instance, a rotated leaf image might simulate how a plant appears in natural conditions when viewed from different angles, while a zoomed-in image helps the model focus on disease-specific features, such as lesions or discoloration.

In addition to geometric transformations, color adjustments are applied to replicate changes in lighting or shading in natural environments. This helps the model generalize better and recognize disease symptoms under various conditions. For example, a leaf with a light yellow tint caused by sunlight may be enhanced to identify and isolate specific disease-related features. Each augmented image is paired with its original label to maintain consistency in the dataset. This preprocessing and augmentation process ensures that the model becomes adept at identifying leaf diseases with high accuracy, even in diverse and unpredictable real-world scenarios.

6.3 Leaf Segmentation and ROI Extraction

This method is a crucial preprocessing step in Plant Pulse, focusing on isolating the leaf from the background for precise analysis. This involves utilizing image processing techniques, such as color thresholding and contour detection, to segment the leaf area from the input image. Advanced methods like edge detection or morphological operations may further refine the segmentation by removing noise and preserving leaf boundaries. Once the leaf is isolated, the Region of Interest (ROI) is extracted, highlighting areas with visible symptoms such as spots, discoloration, or lesions. This ensures that the subsequent deep learning model processes only the relevant parts of the image, enhancing the accuracy and efficiency of disease detection.

6.4 DATA COLLECTION AND LABELING

Involves gathering a diverse dataset of leaf images representing various plant species and disease types under different conditions, such as lighting, angles, and resolutions. Images are collected from multiple sources, including publicly available datasets, agricultural field surveys, and controlled environments. Each image is meticulously labeled with relevant metadata, including the plant species, disease type, and severity level, using expert annotations or existing references. Proper labeling is essential to ensure the dataset's quality and reliability, which directly impacts the training and performance of the deep learning models used for disease detection and classification.

6.5 MODEL EVALUATION AND OPTIMIZATION

The trained model is assessed using performance metrics such as accuracy, precision, and F1-score to evaluate its ability to correctly classify leaf diseases. Validation datasets are used to test the model's robustness and identify any overfitting or underfitting issues. Optimization techniques, including learning rate tuning, are employed to enhance the convergence of the model during training. Dropout regularization is applied to prevent overfitting by randomly deactivating neurons during training, improving the model's ability to generalize to unseen data. Cross-validation strategies ensure the model's adaptability across different datasets. Hyperparameter tuning, such as adjusting the number of layers or neurons, further refines the model's architecture. The evaluation phase provides actionable insights, allowing iterative improvements to achieve a balanced trade-off between accuracy and computational efficiency. This process ensures the system is both reliable and scalable for real-world applications.

6.6 DEPLOYMENT USING STREAMLIT

It focuses on creating an accessible platform for users to interact with the Plant Pulse system. Streamlit is used to develop a user-friendly web interface that allows seamless integration of machine learning models for real-time leaf disease detection. The interface enables users to upload images of plant leaves, which are then analyzed by the trained model to provide instant feedback on the detected disease type and severity. The platform's simplicity ensures that even non-technical users can easily access and utilize the system. Streamlit's flexibility allows for integrating visualizations like heatmaps or confidence scores to make predictions more interpretable. Furthermore, the interface is lightweight and deployable on cloud platforms, making it scalable for larger user bases. Regular updates and improvements can be easily managed to incorporate new features or model.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

The **Plant Pulse** system represents a significant step forward in automating the process of leaf disease detection. By leveraging advanced machine learning and deep learning techniques, the system offers accurate, efficient, and scalable solutions for agricultural monitoring. With its user-friendly interface and real-time feedback capabilities, Plant Pulse can empower farmers and agricultural experts to detect diseases at an early stage, ultimately contributing to improved crop health and higher yields. The use of convolutional neural networks (CNNs) and other deep learning models enables the system to classify diseases accurately, while the deployment through Streamlit makes it accessible for practical, real-world applications. The ethical considerations incorporated into the design ensure responsible data usage, and the project's sustainability aspects promise long-term benefits for global food security. Overall, Plant Pulse offers an innovative, cost-effective solution to a critical problem in agriculture, with the potential to revolutionize plant health management.

7.2 FUTURE ENHANCEMENT

The future of **Plant Pulse** technology holds immense potential for revolutionizing agricultural practices with even more advanced capabilities. As machine learning models continue to evolve, the accuracy of disease detection will improve, enabling the system to identify a wider range of plant diseases with higher precision. Future iterations of **PlantPulse** could incorporate **real-time multi-crop analysis**, allowing for simultaneous disease detection across different types of plants, providing farmers with more comprehensive insights. The system's integration with **Internet of Things (IoT)** devices will allow for continuous monitoring of environmental factors such as temperature, humidity, and soil conditions, making the disease detection process even more context-aware and accurate.

The use of **edge computing** could enable faster processing, reducing the need for constant internet connectivity and allowing for on-site disease detection in remote agricultural areas. **AI-based adaptive learning** will also improve the system's ability to analyze new, unseen diseases by dynamically updating the model with new data. **Augmented Reality (AR)** integration could provide farmers with an immersive experience, visualizing disease spread across crops and suggesting preventive measures in real-time.

Moreover, the system could be expanded to detect **pest infestations** and other crop health issues, making it an all-in-one tool for crop management. With enhanced user interfaces and support for **voice commands**, the system could become more accessible, enabling hands-free interaction for farmers in the field. As **Plant Pulse** becomes more efficient, the cost of implementation will decrease, making it accessible to smaller-scale farmers, contributing to **sustainable agriculture** globally and improving **food security**.

APPENDIX-1

SOURCE CODE

Main.py

```
import streamlit as st
import cv2 as cv
import numpy as np
import keras

label_name = ['Apple scab','Apple Black rot', 'Apple Cedar apple rust', 'Apple healthy',
'Cherry Powdery mildew',

'Cherry healthy','Corn Cercospora leaf spot Gray leaf spot', 'Corn Common rust', 'Corn
Northern Leaf Blight','Corn healthy',

'Grape Black rot', 'Grape Esca', 'Grape Leaf blight', 'Grape healthy','Peach Bacterial
spot','Peach healthy', 'Pepper bell Bacterial spot',

'Pepper bell healthy', 'Potato Early blight', 'Potato Late blight', 'Potato healthy',
'Strawberry Leaf scorch', 'Strawberry healthy',

'Tomato Bacterial spot', 'Tomato Early blight', 'Tomato Late blight', 'Tomato Leaf
Mold', 'Tomato Septoria leaf spot',

'Tomato Spider mites', 'Tomato Target Spot', 'Tomato Yellow Leaf Curl Virus',
'Tomato mosaic virus', 'Tomato healthy']

st.write("""The leaf disease detection model is built using deep learning
techniques, and it uses transfer learning to leverage the pre-trained knowledge of a
base model. The model is trained on a dataset containing images of 33 different
types of leaf diseases.""")

st.write("Please input only leaf Images of Apple, Cherry, Corn, Grape, Peach, Pepper,
Potato, Strawberry, and Tomato. Otherwise, the model will not work perfectly.")
model = keras.models.load_model('Training/model/Leaf Deases(96,88).h5')
uploaded_file = st.file_uploader("Upload an image")
```

```

if uploaded_file is not None:
    image_bytes = uploaded_file.read()

    img = cv.imdecode(np.frombuffer(image_bytes, dtype=np.uint8),
cv.IMREAD_COLOR)

    normalized_image = np.expand_dims(cv.resize(cv.cvtColor(img,
cv.COLOR_BGR2RGB), (150, 150)), axis=0)

    predictions = model.predict(normalized_image)
    st.image(image_bytes)

    if predictions[0][np.argmax(predictions)]*100 >= 80:
        st.write(f"This leaf is affected by {label_name[np.argmax(predictions)]}")
    else:st.write(f"Try Another Image")

```

Request.py

```

import requests

import numpy as np from keras.preprocessing.image

import load_img,img_to_array

url = http://127.0.0.1:5000/

img = img_to_array(load_img('DanLeaf2.jpg',target_size=(150,150,3))

r = requests.post(url, json={'img':img.tolist()})

print(f"\n\n{r.json()}\n\n")

```

Make API.py

```
from flask import Flask, request, jsonify

from tensorflow.keras.models

import load_model import numpy as np

leaf_deases_model = load_model('/home/shukur/Documents/Python Code/Tree
Deases/Leaf_Deases(95,88).h5')

label_name = ['Apple scab','Apple Black rot', 'Apple Cedar apple rust', 'Apple healthy',
'Cherry Powdery mildew', 'Cherry healthy','Corn Cercospora leaf spot Gray leaf spot',
'Corn Common rust', 'Corn Northern Leaf Blight','Corn healthy', 'Grape Black rot',
'Grape Esca', 'Grape Leaf blight', 'Grape healthy','Peach Bacterial spot','Peach healthy',
'Pepper bell Bacterial spot', 'Pepper bell healthy', 'Potato Early blight', 'Potato Late
blight', 'Potato healthy', 'Strawberry Leaf scorch', 'Strawberry healthy', 'Tomato
Bacterial spot', 'Tomato Early blight', 'Tomato Late blight', 'Tomato Leaf Mold',
'Tomato Septoria leaf spot', 'Tomato Spider mites', 'Tomato Target Spot', 'Tomato
Yellow Leaf Curl Virus', 'Tomato mosaic virus', 'Tomato healthy']

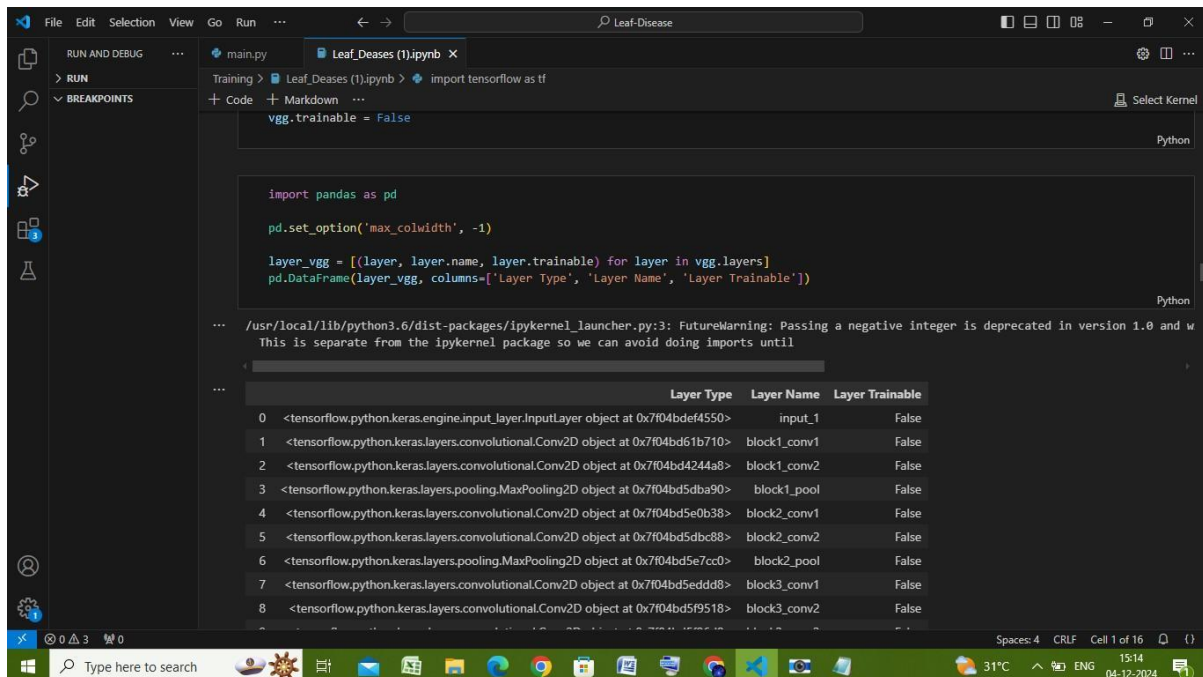
app = Flask(name)

@app.route("/",methods=['POST']) def just(): data = request.json img
= np.array(data['img'])

pridict_image = leaf_deases_model.predict(img.reshape((1,) + img.shape
)) return jsonify({"Label Name":label_name[np.argmax(pridict_image)],
Accuracy": pridict_image[0][np.argmax(pridict_image)]*100})

if name == "main": app.run(debug=True)
```

Trained Model

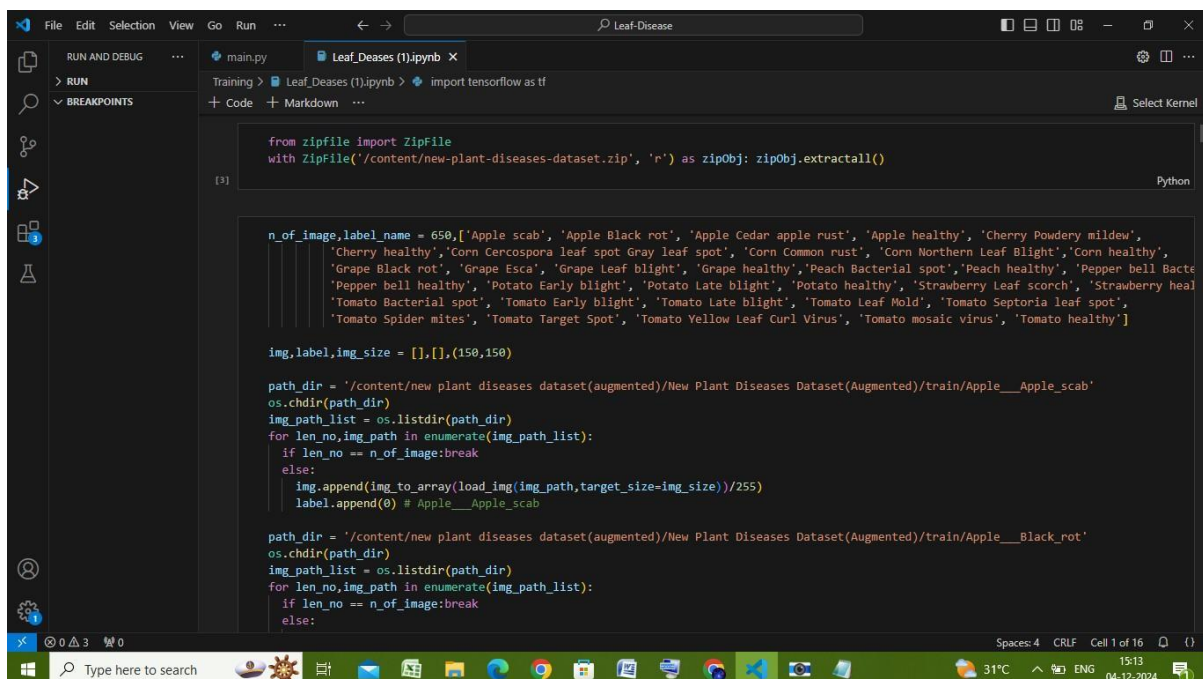


```
import tensorflow as tf
vgg.trainable = False

import pandas as pd
pd.set_option('max_colwidth', -1)

layer_vgg = [(layer, layer.name, layer.trainable) for layer in vgg.layers]
pd.DataFrame(layer_vgg, columns=['Layer Type', 'Layer Name', 'Layer Trainable'])
```

	Layer Type	Layer Name	Layer Trainable
0	<tensorflow.python.keras.engine.input_layer.InputLayer object at 0x7f04bdef4550>	input_1	False
1	<tensorflow.python.keras.layers.convolutional.Conv2D object at 0x7f04bd61b710>	block1_conv1	False
2	<tensorflow.python.keras.layers.convolutional.Conv2D object at 0x7f04bd4244a8>	block1_conv2	False
3	<tensorflow.python.keras.layers.pooling.MaxPooling2D object at 0x7f04bd5dba90>	block1_pool	False
4	<tensorflow.python.keras.layers.convolutional.Conv2D object at 0x7f04bd5e0b38>	block2_conv1	False
5	<tensorflow.python.keras.layers.convolutional.Conv2D object at 0x7f04bd5dbcb8>	block2_conv2	False
6	<tensorflow.python.keras.layers.pooling.MaxPooling2D object at 0x7f04bd5e7cc0>	block2_pool	False
7	<tensorflow.python.keras.layers.convolutional.Conv2D object at 0x7f04bd5edd8>	block3_conv1	False
8	<tensorflow.python.keras.layers.convolutional.Conv2D object at 0x7f04bd5f9518>	block3_conv2	False



```
from zipfile import ZipFile
with ZipFile('/content/new-plant-diseases-dataset.zip', 'r') as zipObj: zipObj.extractall()

n_of_image, label_name = 650, ['Apple scab', 'Apple Black rot', 'Apple Cedar apple rust', 'Apple healthy', 'Cherry Powdery mildew',
                                'Cherry healthy', 'Corn Cercospora leaf spot Gray leaf spot', 'Corn Common rust', 'Corn Northern Leaf Blight', 'Corn healthy',
                                'Grape Black rot', 'Grape Esca', 'Grape Leaf blight', 'Grape healthy', 'Peach Bacterial spot', 'Peach healthy', 'Pepper bell Bacte',
                                'Pepper bell healthy', 'Potato Early blight', 'Potato Late blight', 'Potato healthy', 'Strawberry Leaf scorch', 'Strawberry heal',
                                'Tomato Bacterial spot', 'Tomato Early blight', 'Tomato Late blight', 'Tomato Leaf Mold', 'Tomato Septoria leaf spot',
                                'Tomato Spider mites', 'Tomato Target Spot', 'Tomato Yellow Leaf Curl Virus', 'Tomato mosaic virus', 'Tomato healthy']

img, label, img_size = [], [], (150, 150)

path_dir = '/content/new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/train/Apple___Apple_scab'
os.chdir(path_dir)
img_path_list = os.listdir(path_dir)
for len_no, img_path in enumerate(img_path_list):
    if len_no == n_of_image: break
    else:
        img.append(img_to_array(load_img(img_path, target_size=img_size))/255)
        label.append(0) # Apple___Apple_scab

path_dir = '/content/new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/train/Apple___Black_rot'
os.chdir(path_dir)
img_path_list = os.listdir(path_dir)
for len_no, img_path in enumerate(img_path_list):
    if len_no == n_of_image: break
    else:
```

The screenshot shows a Jupyter Notebook titled 'Leaf_Deases (1).ipynb' in the VS Code editor. The code is in Python and imports TensorFlow and Keras. It defines a function to load and preprocess images from a dataset. The VGG16 model is loaded with 'imagenet' weights, and its summary is printed.

```

import tensorflow as tf

for len_no, img_path in enumerate(img_path_list):
    if len_no == n_of_image: break
    else:
        img.append(img_to_array(load_img(img_path, target_size=img_size))/255)
        label.append(32) # Tomato___healthy

img, label = np.array(img), np.array(label)

IMG_SHAPE = img_size
vgg = tf.keras.applications.vgg16.VGG16(weights='imagenet', include_top=False, input_shape=IMG_SHAPE)

vgg.summary()

```

The model summary for VGG16 is displayed below the code:

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 150, 150, 3)]	0
block1_conv1 (Conv2D)	(None, 150, 150, 64)	1792
block1_conv2 (Conv2D)	(None, 150, 150, 64)	36928
block1_pool (MaxPooling2D)	(None, 75, 75, 64)	0

The screenshot shows the same Jupyter Notebook with additional code to load multiple datasets from a directory. The code uses os.listdir to iterate through subdirectories and load images and labels for different plant diseases.

```

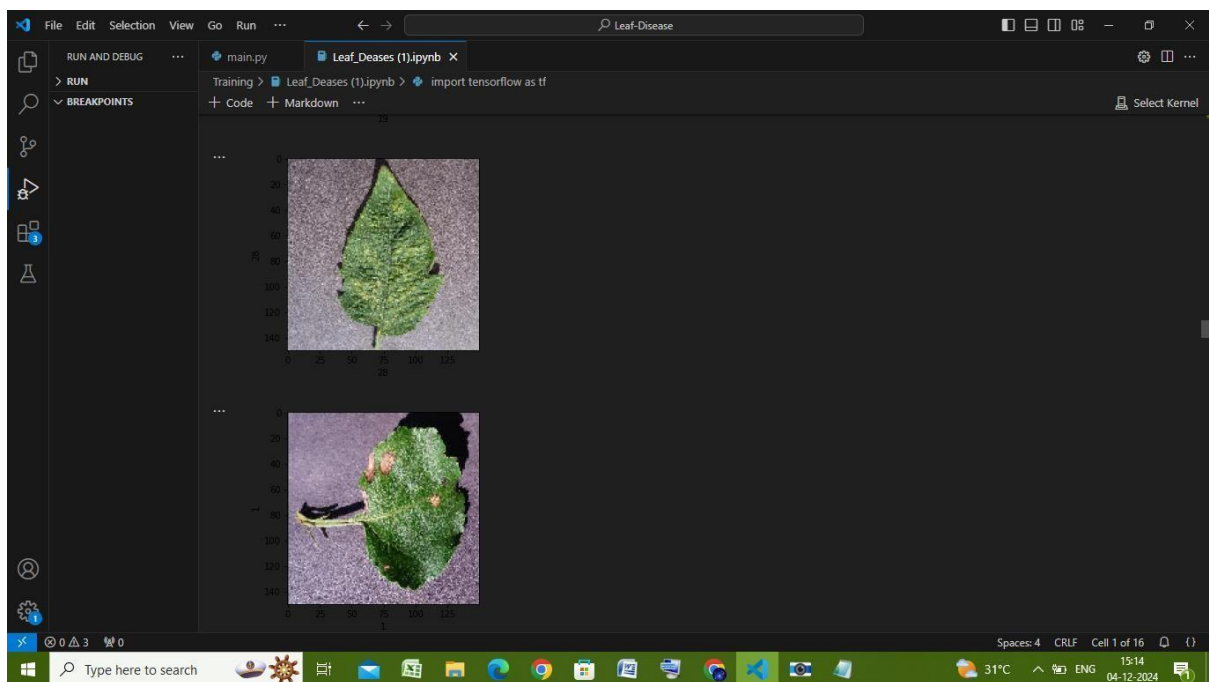
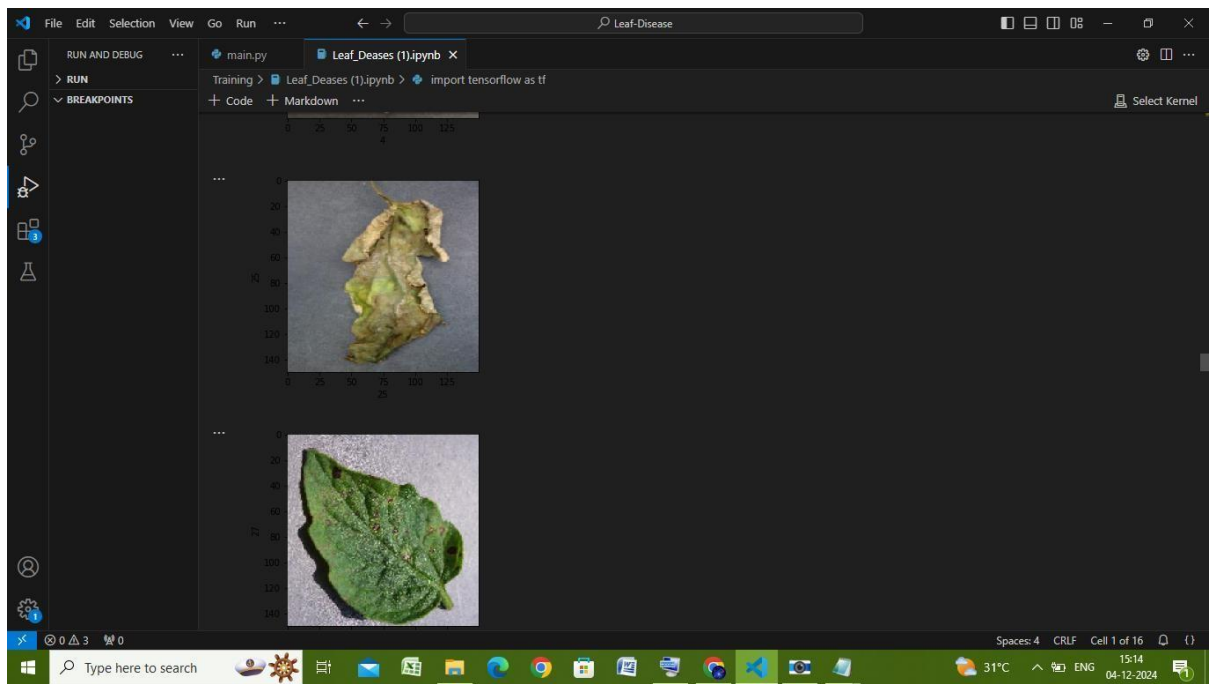
path_dir = '/content/new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/train/Apple___healthy'
os.chdir(path_dir)
img_path_list = os.listdir(path_dir)
for len_no, img_path in enumerate(img_path_list):
    if len_no == n_of_image: break
    else:
        img.append(img_to_array(load_img(img_path, target_size=img_size))/255)
        label.append(3) # Apple___healthy

path_dir = '/content/new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/train/Cherry_(including_sour)___Powdery_mildew'
os.chdir(path_dir)
img_path_list = os.listdir(path_dir)
for len_no, img_path in enumerate(img_path_list):
    if len_no == n_of_image: break
    else:
        img.append(img_to_array(load_img(img_path, target_size=img_size))/255)
        label.append(4) # Cherry_(including_sour)___Powdery_mildew

path_dir = '/content/new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/train/Cherry_(including_sour)___healthy'
os.chdir(path_dir)
img_path_list = os.listdir(path_dir)
for len_no, img_path in enumerate(img_path_list):
    if len_no == n_of_image: break
    else:
        img.append(img_to_array(load_img(img_path, target_size=img_size))/255)
        label.append(5) #Cherry_(including_sour)___healthy

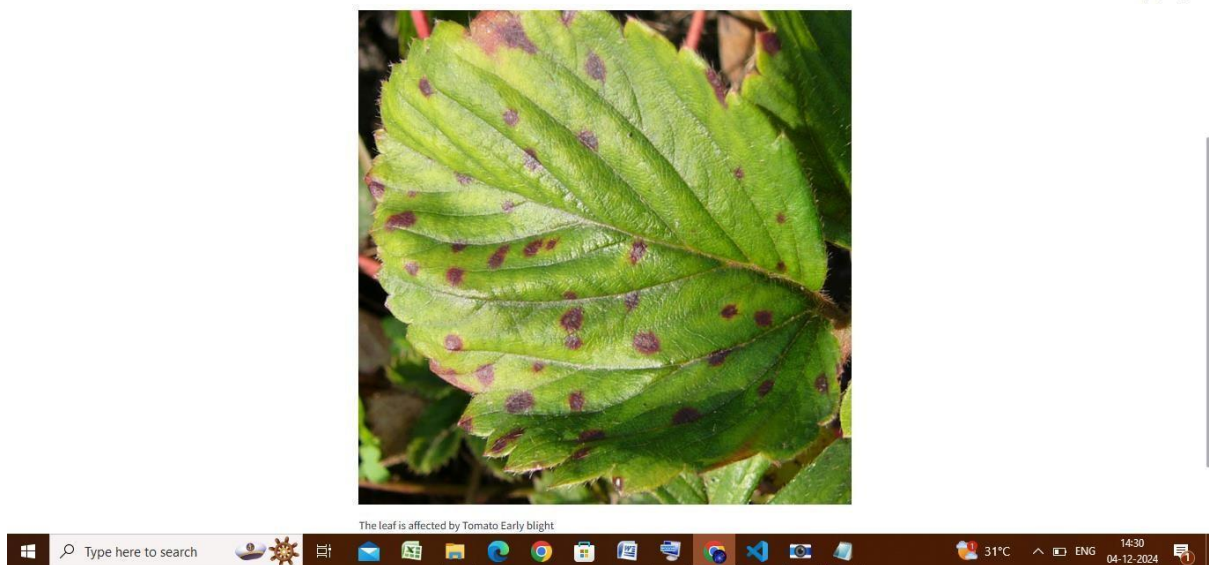
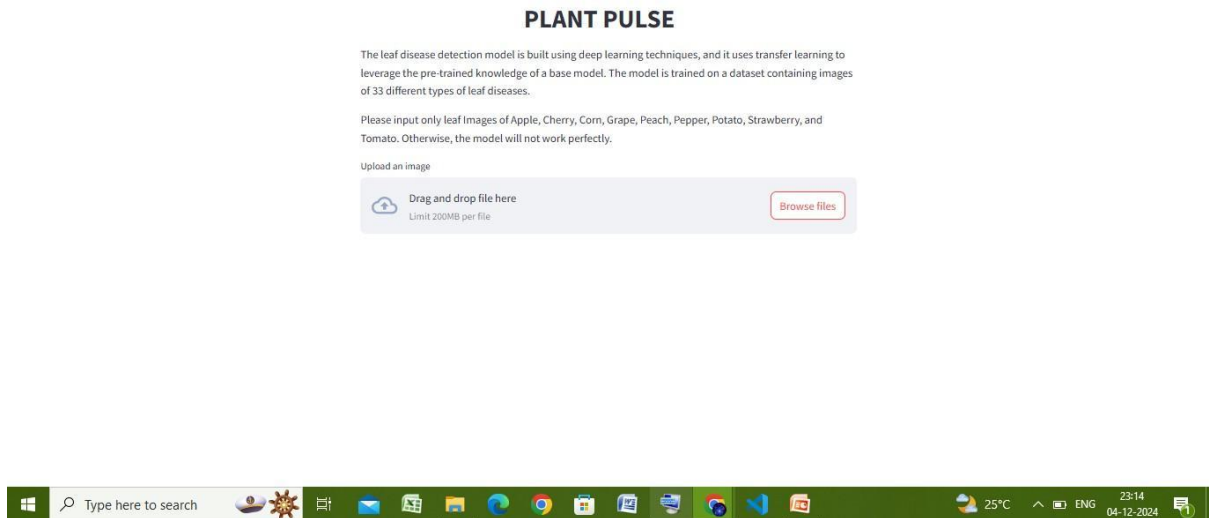
path_dir = '/content/new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/train/Corn_(maize)___Cercospora_leaf_spot'
os.chdir(path_dir)
img_path_list = os.listdir(path_dir)
for len_no, img_path in enumerate(img_path_list):

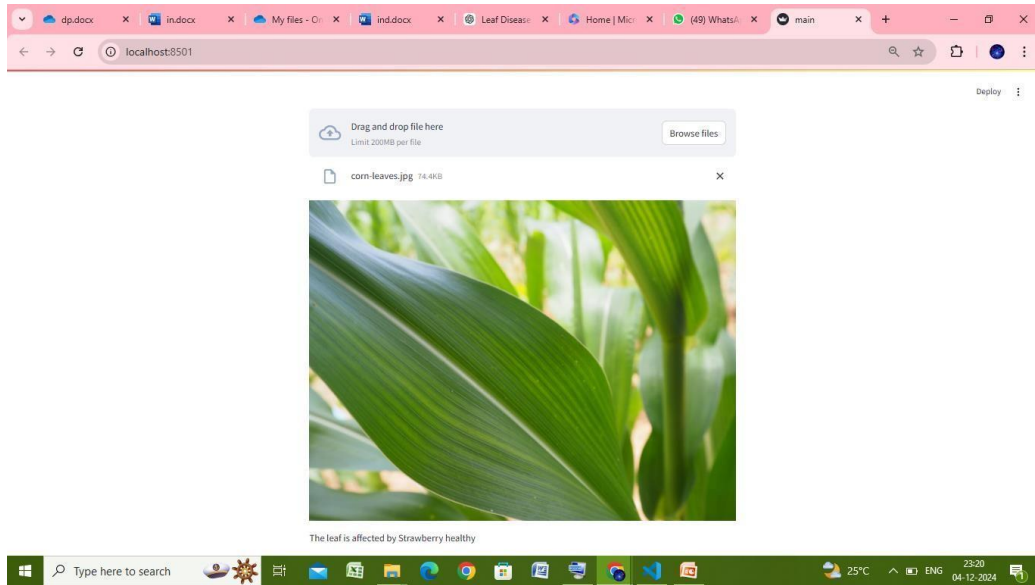
```



APPENDIX – 2

SCREENSHOTS





REFERENCES

1. Kumari, S., Devi, P. R., & Sahasra, T. M. (Year). *Leaf Disease Detection using Deep Learning*. Journal of Agricultural Technology and Innovations.
2. **Ferentinos, K. P. (Year). *Deep Learning Models for Plant Disease Detection and Diagnosis*. Computers and Electronics in Agriculture.**
3. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.
4. **Hughes, D. P., & Mohanty, S. P. (2017). *Frontiers in Plant Science: Automated Plant Disease Detection with Machine Learning*. Frontiers in Plant Science.**
5. **Chouhan, S. S., & Bharti, R. (2018). *Automatic Detection of Plant Diseases Using Convolutional Neural Networks*. Agricultural Science.**
6. **Zhang, J., & Zhao, J. (2020). *Application of Deep Learning in Plant Disease Detection and Classification*. Journal of Computational Biology.**
7. **Hughes, D. P., & Salathé, M. (2019). *Plant Disease Prediction Using Deep Learning Techniques: A Review*. Precision Agriculture.**