Plant Health Detection

T.E. mini-project report submitted in partial fulfilment of

the requirements of the degree of

Information Technology

by

Nikhil Shet EU2204019 (66)

Dhruv Goswami EU2204023 (56)

Dheeraj Khatiya EU2204024 (60)

Jeet Desai EU2204004 (63)

Under the guidance of

Mrs. Meenal Kate | Assistant Professor

Department of Information Technology

St. John College of Engineering and Management, Palghar

University of Mumbai

2023-24

CERTIFICATE

This is to certify that the B.E. mini-project entitled **"Plant Health Detection**" is a bonafide work of **"Nikhil Shet" (EU2204019) (45), "Dhruv Goswami"(EU2204023) (56), "Dheeraj Khatiya (EU2204024)" (60)** and **"Jeet Desai" (EU2204004) (63)** submitted to University of Mumbai in partial fulfilment of the requirement for the award of the degree of **"Information Technology Engineering"** during the academic year 2022- 23.

Mrs. Meenal Kate | Assistant Professor

Guide

Dr. Arun Saxena

Head of Department

Dr. G.V. Mulgund

Principal

Table of Contents

List of Figures

Chapter 1

Introduction

In order to stop or control the spread of plant diseases and pests, the process of recognizing, diagnosing, and monitoring the health of plants is referred to as "plant health detection." Any negative changes in a plant's health can have a big influence on the environment, agriculture, and the economy. Plant health is essential to their development and output. Visual inspection has historically been used to detect plant health, although this method can be labor- and time-intensive and subject to human error. However, because to technological advancements, a variety of tools and methods are now available for more precise and effective plant health diagnosis, including molecular diagnostics, imaging techniques, and remote sensing. Plant health detection is a field of study under the image recognition field of computer vision. Recognizing leaves is of utmost importance in biodiversity conservation. The major project, detection of diseases in laves, is also another important milestone in conserving not just biodiversity but also saving crops from disease spread. The algorithm PCA has aided the process of leaf detection, by the monitoring of some basic features of leaves and then comparing the values obtained with the available data set.

Keywords— diagnosing, human error, PCA algorithm, plant health

1.1 Motivation

The need to manage agricultural systems sustainably, provide food security, and safeguard the environment are some of the considerations that drive plant health detection. Following are some details on why plant health detection is crucial To ensure food security, keep in mind that plants are a major source of food for both people and animals, and that crop yields and quality can be greatly affected by plant diseases and pests. Finding and controlling plant diseases and pests with the aid of plant health detection can help prevent crop losses and guarantee that there is enough food to feed the world's expanding population. Keep the environment safe: Plant pests and diseases can seriously disrupt the environment by destroying ecosystems, contaminating streams, and harming wildlife. By identifying and controlling plant pests and diseases. Protect the environment: Plant diseases and pests can have significant environmental impacts, such as the destruction of ecosystems, contamination of waterways, and harm to wildlife. By detecting and managing plant diseases and pests.

1.2 Problem Statement

To increase the effectiveness and efficiency of the process, it is necessary to address a number of problems and issues that plant health detection faces. The following are a few of the main issue statements in plant health detection: Small- scale farmers and areas with limited resources typically cannot afford the sometimes expensive and specialized equipment and knowledge needed for planthealth detection methods and tools. The scalability and effectiveness of traditional plant health detection techniques, such as eye inspection, may be constrained by their labor- and time-intensive nature. Plant health detection procedures are not standardized, which makes it challenging to compare results and assess the potency of various approaches.

1.3 Objectives

The following are the goals of plant health detection:

- Determine the presence and severity of plant diseases and pests: The main goal of plant health detection is to precisely determine the presence and severity of plant diseases and pests.

- Monitor changes in plant health over time to identify possible issues before they become serious. This is another usage for plant health detection.

- Identify plant illnesses and pests effectively to prevent crop losses and the use of chemical pesticides. Accurate and prompt plant health detection can assist in the development of effective control methods for plant diseases and pests.

- Promote integrated pest control techniques, improve soil health, and lessen the usage of toxic pesticides are all ways that plant health detection can assist advance sustainable farming practises.

- Increase productivity and cut costs: Plant health detection can help agricultural systems become more productive by spotting issues before they become serious, removing the need for time-consuming human inspection, and lowering crop losses.

1.4 Scope

A plant health detection system is a technology that uses various methods to monitor and analyze the health status of plants. The system can detect early signs of diseases, nutrient deficiencies, and other factors that affect plant growth and development. The scope of a plant health detection system can vary depending on the specific needs and requirements of the users. It can be used in various settings, such as agricultural fields, greenhouse environments.

Chapter 2
Requirement Analysis

3.1 Hardware and Software Requirements

- **Hardware Requirements:**
    - Windows XP, Windows 7(32/64bit) or higher
    - Minimum 4GB RAM and higher
    - CPU @2.30GHz
    - 10GB available space on the hard disk
    - At least one Internet Browser e.g. Chrome, Firefox, Microsoft Edge etc.

- **Software Requirements:**
    - Windows 10 64-bit Operating System
    - Python 3.11.1
    - Jupyter Notebook

3.2 UML Diagram

3.2.1 Use Case Diagram

Following fig 3.2.1 shows Use Case Diagram

Fig 3.2.1 Use Case Diagram

3.2.2 Class diagram

Following fig 3.2.2 shows Class Diagram

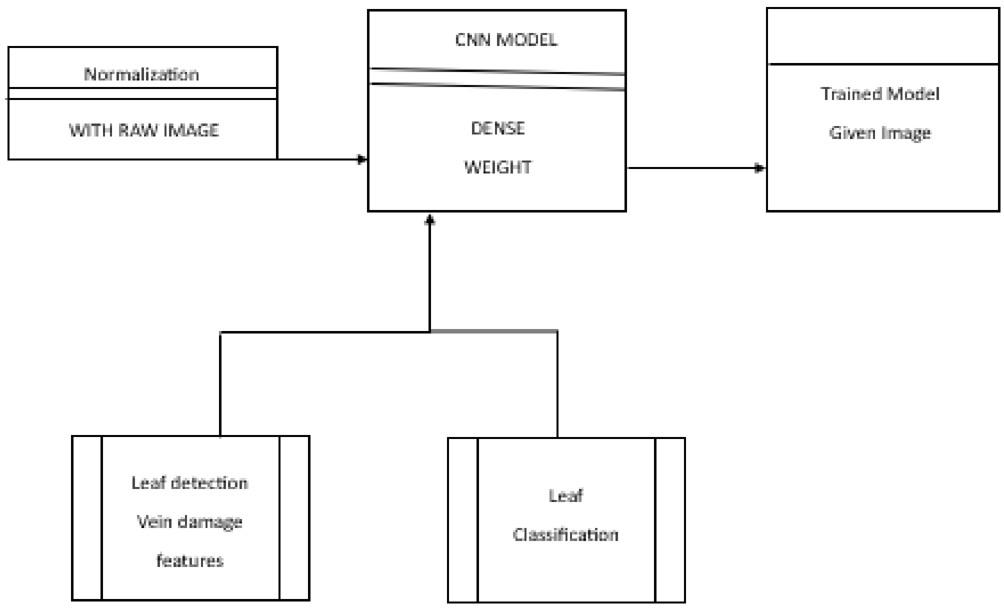

Fig 3.2.2 Class Diagram

3.2.3 Activity Diagram

Following Fig 3.2.3 shows Activity Diagram

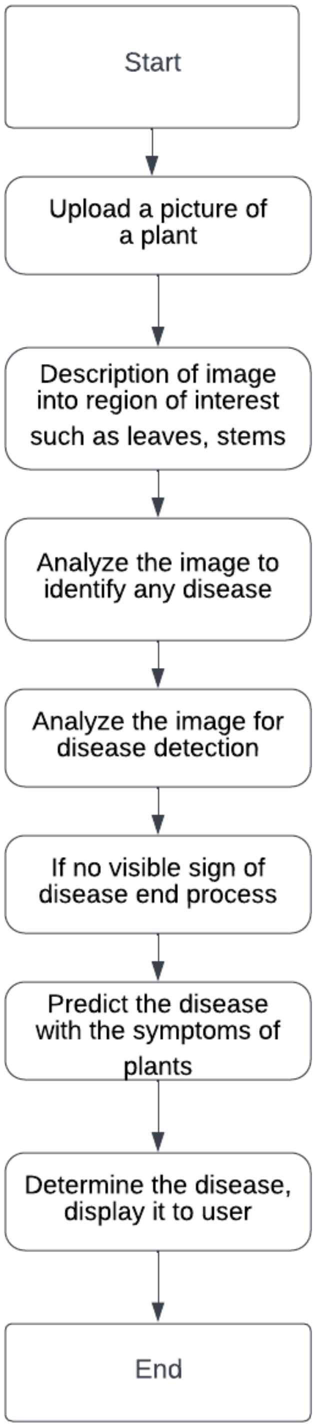

Fig 3.2.3 Activity Diagram

3.3 Timeline Chart

Following fig 3.3 shows Timeline Chart

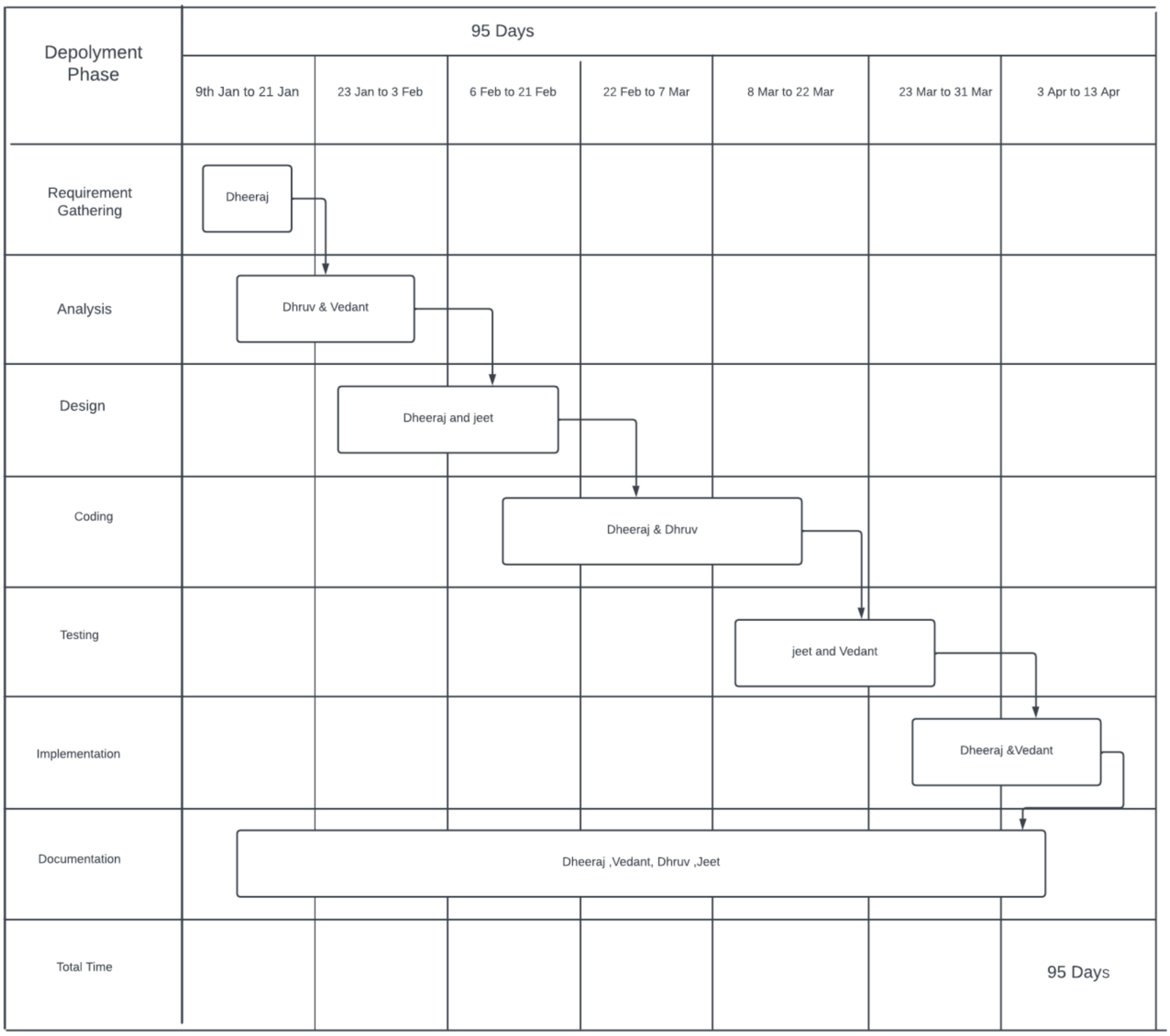

Fig 3.3 Timeline Chart

3.4 W.B.S

Following fig 3.4 shows W.B.S

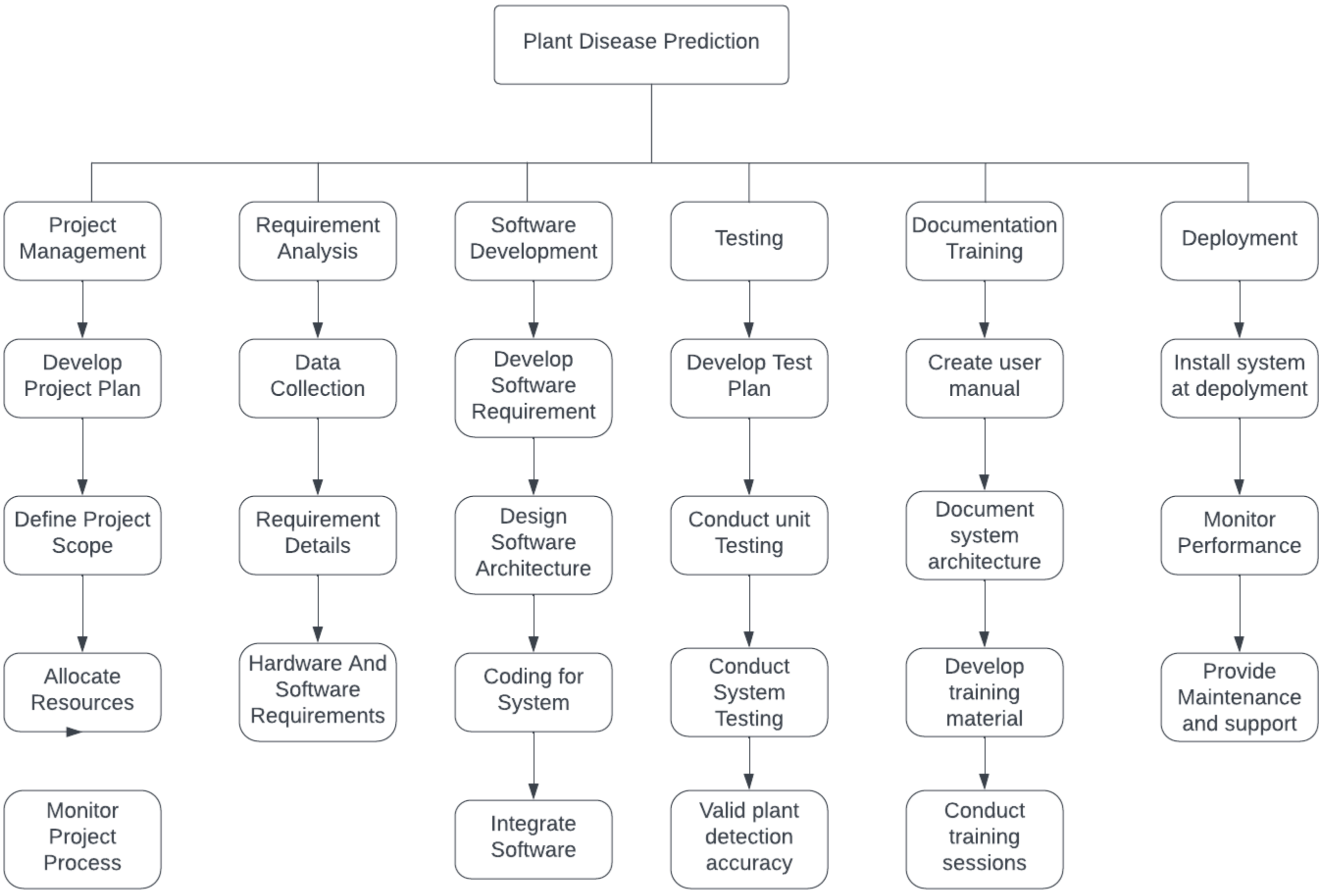

Fig 3.4 W.B.S Chart

3.4.2 Sequence Diagram

Following fig 3.4.3 shows Sequence Diagram

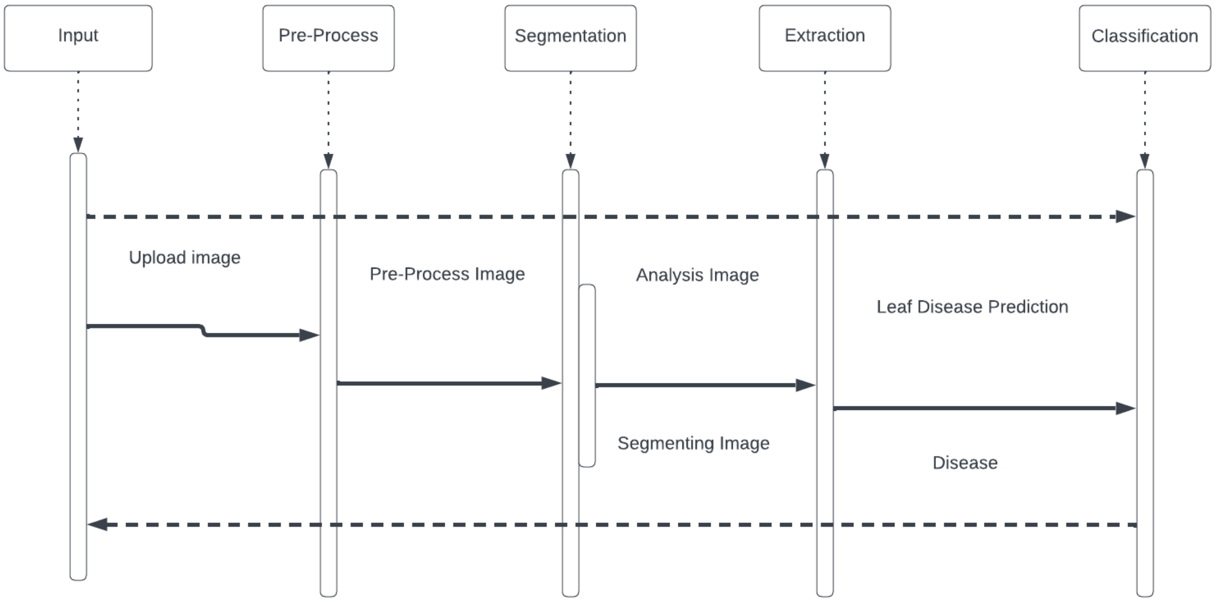

Fig 3.4.3 Sequence Diagram

Chapter 3

Report on Present Investigaion

4.1 Proposed System

4.1.1 Block diagram of Proposed System

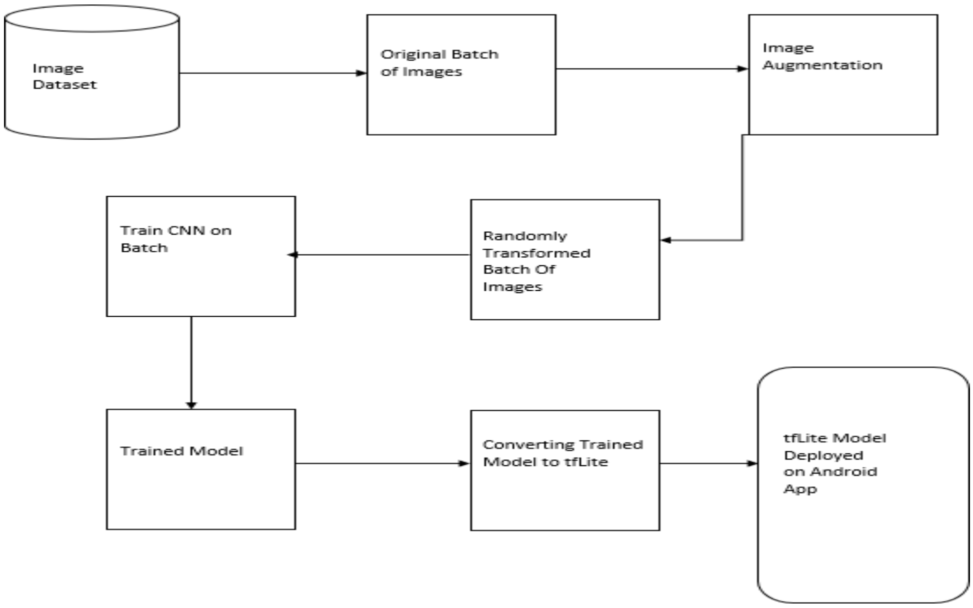

Fig 4.1.1 Block diagram of Proposed System

- **Plant Images**

  The data sets used in the research include the descriptions of the leaves before and after the diseases affect them.

- **Data Preprocessing**

  The use of computer algorithms to perform image processing on digital images is known as image pre-processing.

- **Model Training & Model Evaluation**

  The model is developed based for detection of leaf disease in presented in the section and is been evaluated.

- **Model Deployment**

  Owing to their evolutionary potential, plant pathogens are able to rapidly adapt to genetically controlled plant resistance, often resulting in resistance breakdown and major epidemics in agricultural crops.

4.2 Implementation

4.2.1 Algorithm/Flowchart

Following fig 4.2.1 shows Algorithm/Flowchart

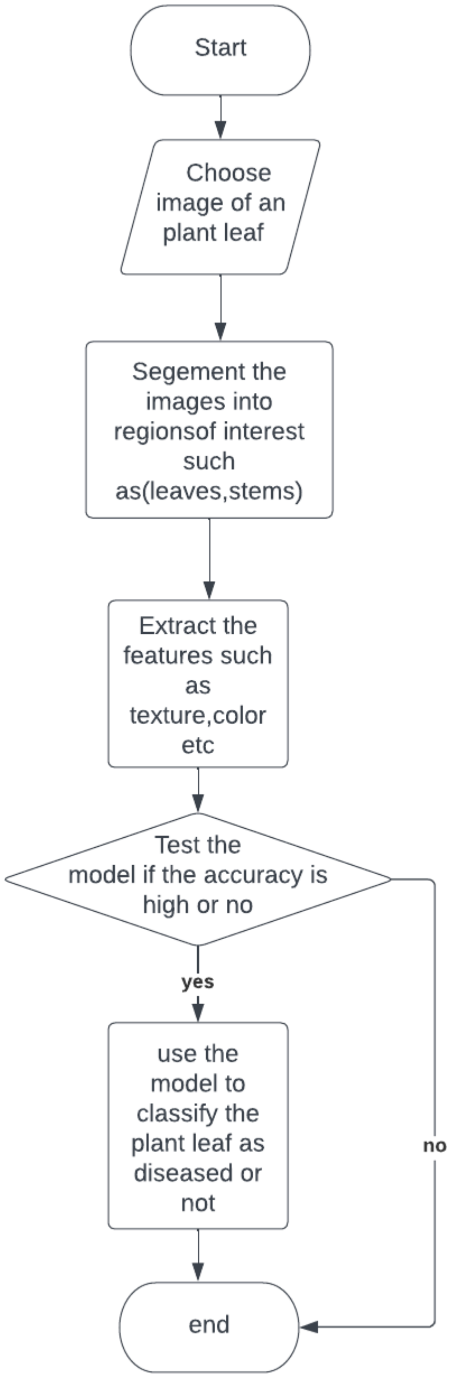

Fig 4.1.2 Algorithm/Flowchart

4.2.3 Pseudo code

import numpy as np

import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv) # Input data files are available in the read-only "../input/" directory

For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os

from PIL import Image import matplotlib.pyplot as plt

from matplotlib.image import imread import random plt.figure(figsize=(12,12))

path = "C:/Users/dhira/Downloads/Plant_images/Potato Early_blight" for i in range(1,17):

plt.subplot(4,4,i) plt.tight_layout()

rand_img = imread(path +'/'+ random.choice(sorted(os.listdir(path)))) plt.imshow(rand_img)

plt.xlabel(rand_img.shape[1], fontsize = 10)#width of image plt.ylabel(rand_img.shape[0], fontsize = 10)#height of image

def convert_image_to_array(image_dir):

try:

image = cv2.imread(image_dir) if image is not None :

image = cv2.resize(image, (256,256))

#image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY) return img_to_array(image)

else :

return np.array([]) except Exception as e:

heading_label.pack()

def showimage():

Load the image

img_path = filedialog.askopenfilename(initialdir=os.getcwd(), title="SELECTfile", filetypes=(("JPG File", ".jpg"), ("PNG File", ".png"), ("ALL Files", ".")))

img = Image.open(img_path)

Convert the image to a numpy array img_array = np.array(img)

Normalize the image img_array = img_array / 255.0

Add a new axis to the image array to make it compatible with the model img_array = np.expand_dims(img_array, axis=0)

Display the image in the GUI img = ImageTk.PhotoImage(img) lbl.configure(image=img) lbl.image=img

Predict the class of the image using the model y_pred = model.predict(img_array)[0]

Get the class label corresponding to the predicted class index

all_labels = ['Corn_(maize)_ Common_rust_','Potato Early_blight', 'Tomato Bacterial_spot']

predicted_label = all_labels[np.argmax(y_pred)]

Show a message box with the predicted class label messagebox.showinfo("Predicted : ", predicted_label)

heading_label1=Label(root,text="PREVENTIONS FOR PLANT DISEASES", font=("Helvetica", 18, "bold"))

heading_label1.pack()

4.2.4 Implementation

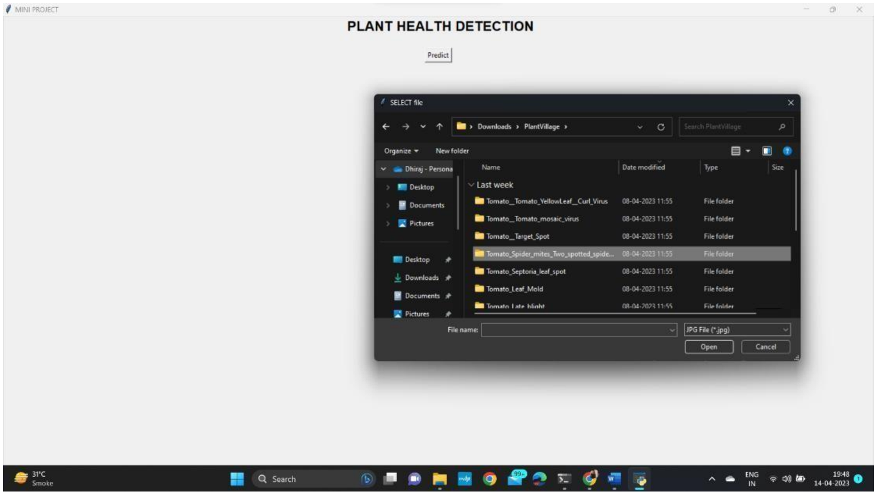

Fig: Uploading an image from the dataset

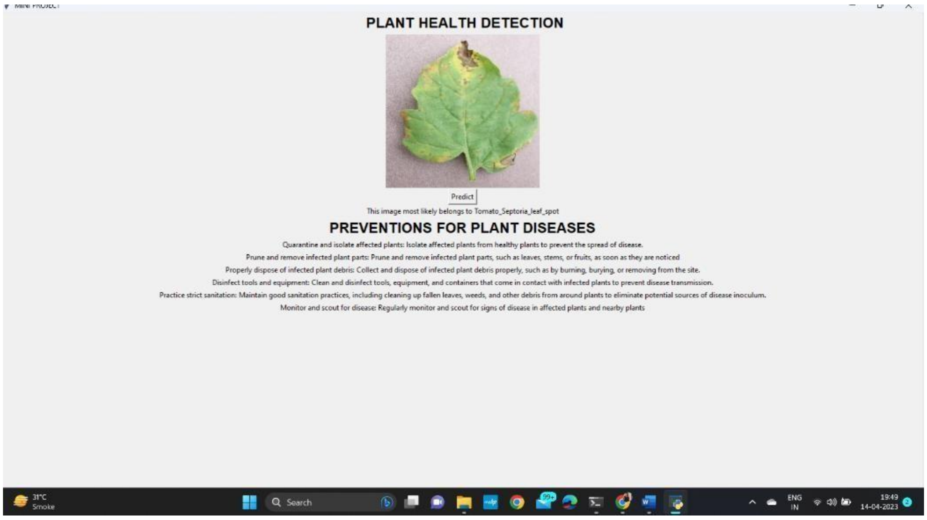

Fig: Final Output

In [33]:

```
IMG_SIZE = 64
import cv2
training = []
for lb in labels:
    img_path = os.path.join(fol_path, lb)
    class_num = labels.index(lb)
    for img in os.listdir(img_path):
        try:
            img_array = cv2.imread(os.path.join(img_path, img))
            new_array = cv2.resize(img_array, (IMG_SIZE, IMG_SIZE))
            training.append([new_array, class_num])
        except:
            continue
```

Fig

Above code appears to be loading images using OpenCV (cv2), resizing them to a specified size (IMG_SIZE), and then appending them to a training list along with their corresponding class number. The class number is determined based on the index of the label (lb) in a list of labels.

In [40]:

```
model = Sequential([
    Conv2D(16, (3,3), activation='relu',input_shape=(64,64,3)),
    MaxPooling2D((2,2),strides=2),
    Conv2D(16, (3,3), activation='relu'),
    MaxPooling2D((2,2),strides=2),
    Flatten(),
    Dense(64,activation='relu'),
    Dense(15, activation='softmax')

])
```

Fig

The model architecture consists of the following layer.

Conv2D: This is a convolutional layer with 16 filters, each of size (3,3), and the ReLU activation function. It takes an input shape of (64,64,3), where 64x64 is the size of the input images and 3 represents the number of color channels (RGB).

MaxPooling2D: This is a max pooling layer with a pool size of (2,2) and a stride of 2. It performs down-sampling by selecting the maximum value within each 2x2 patch of the feature maps.

Conv2D: Another convolutional layer with 16 filters of size (3,3) and ReLU activation.

MaxPooling2D: Another max pooling layer with a pool size of (2,2) and a stride of 2.

Flatten: This layer flattens the output of the previous layers into a 1D array, which is then fed as input to the fully connected layers.

Dense: The output layer with 15 units (matching the number of classes) and softmax activation, which is used for multi-class classification.

```
In [44]:  1 history = model.fit(X_train, y_train, batch_size= 16, epochs=20, verbose=1, validation_data=(X_test, y_test),callbacks=[call

Epoch 1/20
1032/1032 [==============================] - 53s 49ms/step - loss: 1.3638 - accuracy: 0.5707 - val_loss: 0.9027 - val_accuracy:
0.7066
Epoch 2/20
1032/1032 [==============================] - 50s 49ms/step - loss: 0.6907 - accuracy: 0.7717 - val_loss: 0.6162 - val_accuracy:
0.7997
Epoch 3/20
1032/1032 [==============================] - 48s 46ms/step - loss: 0.5091 - accuracy: 0.8280 - val_loss: 0.4489 - val_accuracy:
0.8484
Epoch 4/20
1032/1032 [==============================] - 48s 47ms/step - loss: 0.4117 - accuracy: 0.8615 - val_loss: 0.4385 - val_accuracy:
0.8549
Epoch 5/20
1032/1032 [==============================] - 49s 48ms/step - loss: 0.3369 - accuracy: 0.8867 - val_loss: 0.4381 - val_accuracy:
0.8542
Epoch 6/20
1032/1032 [==============================] - 50s 48ms/step - loss: 0.2866 - accuracy: 0.9023 - val_loss: 0.5806 - val_accuracy:
0.8210
Epoch 7/20
1032/1032 [==============================] - 49s 47ms/step - loss: 0.2406 - accuracy: 0.9170 - val_loss: 0.4036 - val_accuracy:
0.8702
Epoch 8/20
1032/1032 [==============================] - 49s 48ms/step - loss: 0.2074 - accuracy: 0.9294 - val_loss: 0.3938 - val_accuracy:
0.8687
Epoch 9/20
1032/1032 [==============================] - 49s 47ms/step - loss: 0.1750 - accuracy: 0.9419 - val_loss: 0.4073 - val_accuracy:
0.8716
Epoch 10/20
1032/1032 [==============================] - ETA: 0s - loss: 0.1469 - accuracy: 0.9507ur reached 90% accuracy
1032/1032 [==============================] - 50s 48ms/step - loss: 0.1469 - accuracy: 0.9507 - val_loss: 0.3813 - val_accuracy:
0.8767
```

Fig

This code snippet trains the previously defined model using the fit() method, which takes in the training data, batch size, number of epochs, and other parameters.

Chapter 5

Results and Discussion

We conducted our study using a combination of image-based techniques and machine learning-based techniques. For image-based techniques, we used hyperspectral imaging, and RGB imaging to capture images of plants. We then analysed these images using computer vision algorithms to id entify patterns or anomalies that indicated plant stress or disease.

For machine learning-based techniques, we used algorithms that could learn from data to identify patterns or anomalies that indicated plant stress or disease. These algorithms were trained using large datasets of plant health information and were able to predict plant health based on a set of input variables.

Our study showed that all three types of techniques were effective in detecting plant stress and disease. Image-based techniques were particularly effective in detecting visual changes in plant health, while sensor-based techniques provided more detailed information on specific parameters such as soil moisture and gas levels. Machine learning-based techniques provided a more holistic approach by incorporating data from various sources to predict plant health.

Chapter 6
Conclusion

6.1 Conclusion

Identifying plant health is a crucial part of sustainable agriculture that needs constant attention and innovation to solve current problems and satisfy future demands. We can advance food security, safeguard the environment, and sustainably manage agricultural systems by investing in plant health detection.The model is basically tested on some types of plant species with some types of plant diseases. As an extension to the project the number of classes of plants and its diseases will be increased. Also the model will be further improved by increasing the parameters for training and test. The proposed work summarizesmultiple studies regarding plant disease automation and identification through different ML methods.

6.2 Future Work

Since currently the system is trained using Plant Village dataset, the model is trained to detect only few types of plant diseases. We propose to train the system with much more data of various other plants and diseases to further increase the scope of the system. By adding images of many other plants, it will help in extracting many more features of the plants which certainly help in improving the accuracy of the system. The users using the system may also contribute to the system by capturing different types of plant images which can be added to the dataset.

References

[1] D. Singh, S. Singh, and V. K. Gupta (2021). Review of plant health monitoring. 24(1), 33–50; Journal of Crop Science and Biotechnology.

[2](2016). Kumar, A., B. Ganapathysubramanian, and A. K. Singh. High-throughput plant stress phenotyping using machine learning. 21(2): 110–124 in Trends in Plant Science.

[3] A. K. Mahlein, M. T. Kuska, S. Thomas, M. Wahabzada, and J. Behmann (2018). State-of-the-art hyperspectral sensors and imaging technology for phytopathology. 56, 535-558, Annual Review of Phytopathology.

[4](2010). Bock, C. H., Poole, G. H., Parker, P. E., and T. R. Gottwald. Visual inspection, digital photography, image analysis, and hyperspectral imaging are used to gauge the severity of plant diseases. 59–107 in Critical Reviews in Plant Sciences, 29(2).

[5] Simona E. Grigorescu, Nicolai Petkov, and Peter Kruizinga, Comparison of Texture Features Based on Gabor Filters , IEEE transaction on Image Processing, vol. 11, No. 10, pp-1160-1167, oct 2002 .

[6] Mohanty SP, Hughes DP and Salathé M (2016) Using Deep Learning for Image-Based Plant Disease Detection. Front. Plant Sci. 7:1419. doi: 10.3389/fpls.2016.01419

[7] R. M. Haralick, K. Shanmugam and I. Dinstein, "Textural Features for Image Classification," in IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-3, no. 6, pp. 610-621, Nov. 1973, doi: 10.1109/TSMC.1973.4309314.

[8] S. D. Khirade and A. B. Patil, "Plant Disease Detection Using Image Processing," 2015 International Conference on Computing Communication Control and Automation, 2015, pp. 768-771, doi: 10.1109/ICCUBEA.2015.153.

[9] S. C. Madiwalar and M. V. Wyawahare, "Plant disease identification: A comparative study," 2017 International Conference on Data Management, Analytics and Innovation (ICDMAI), 2017, pp. 13-18, doi: 10.1109/ICDMAI.2017.8073478.