FERTILIZERS RECOMMMENDATION FOR DISEASE PREDICTION

IBM PROJECT REPORT

SUBMITTED BY

TEAM ID:PNT2022TMID31284

SNEHA.K(621519205040)

SURUTHI.G(621519205045)

VAISHNAVI.R(621519205047)

VIJAYARASI.V(621519205050)

YUVASRI.S(621519205052,)

MAHENDRA EDUCATIONAL INSTITUTIONS

MAHENDRA COLLEGE OF ENGINEERING SALEM (ANNA UNIVERSITY)

BACHELOR OF ENGINEERING

IN

INFORMATION TECHNOLOGY

MINNAMPALLI

SALEM

ANNA UNIVERSITY: CHENNAI 600 025

NOV 2022

S.NO	TABLE OF CONTENTS	PAGE NO

1. INTRODUCTION

1.1 PROJECT OVERVIEW

Agriculture is the most important sector in today's life. Most plants are affected by a wide variety of bacterial and fungal diseases . Plant diseases cause yield reductions that have a direct influence on the domestic and international food production systems and lead to financial losses. The Food and Agriculture Organization (FAO) of the United Nations International Plant Protection Convention (2017) estimates that plant diseases and pests cause a 20% to 40% loss in worldwide food production. An estimated 13% of the global crop yield loss is attributable to plant diseases. These figures demonstrate how crucial it is to recognise plant diseases in order to reduce production losses. But first, it's essential to comprehend the causes of plant illnesses. Three factors aid disease formation in plants: the host, a favorable environment, and the pathogen. These factors create the plant disease triangle shown in Fig. 1. In most cases, diseases begin to show symptoms and affect the plant from the bottom up. The triangle of plant disease is formed by these elements. The majority of the time, illnesses start to manifest symptoms and affect a plant from the ground up. After infection, many plant diseases spread throughout the entire crop. Crops must be periodically checked since early disease control can stop the spread of the disease. Many times, after pollination, plant illnesses also manifest themselves later in the growing season. Different forms of plant diseases damage various plant organs. Plant pathologists can most easily detect foliar diseases, or plant diseases that manifest symptoms on leaves, by looking at these diseases' distinctive characteristics. Up to 50% of yield losses are specifically attributable to fungal infections.

HOST

DISEASE

PATHOGEN ENVIRONMENT

Therefore, the majority of contemporary studies employ computer vision, machine learning and deep learning methods to recognise illnesses in photos of plant leaves. Effective plant disease diagnosis will include early-season plant disease identification, identification of multiple diseases in various crops and multiple concurrent diseases, estimation of the disease severity, estimation of the proper volume of pesticide to apply, and practical actions to take for managing the disease to limit its spread. Plant phenotyping and precision agriculture both need the diagnosis of plant diseases. Both of these disciplines rely heavily on data, information, and technology. Because they require a manual visual inspection and are expensive, time-consuming, and dependent on experts, conventional plant disease diagnosis and monitoring techniques are inefficient for precision agriculture. Additionally, these methods are probably influenced by human bias and weariness, resulting in decreased accuracy. Studies have looked into how image processing methods might be used to plant photos in order to overcome these unreliable methods for disease identification. One of the first studies to use classical image processing for plant diseases was in 1983, utilising black and white photos and films of leaves from potted tomato and blackened plants. The severity of corn streak disease was also quantified using image processing techniques, and it was shown that computer based approaches were more precise than conventional visual examination. The ability of conventional image processing methods to assist in the objective diagnosis of diseases has made them increasingly popular over the past 30 years. However, because these methods rely on manual feature extraction, which takes time and is vulnerable to individual bias because various research could give different traits varying degrees of importance. Developed Technologies have provided the ability to produce sufficient food to meet the demand of society. The food's and the crops' safety and security, however, remained unachieved. Farmers face difficulties due to factors such as climate change, a decrease in pollinators, plant diseases, and other issues. These qualities require a strong foundation, which needs to be accomplished as soon as possible. The utilisation of analysis and detection techniques employing current technology aids farmers in solving such issues. The country depends on modern technologies in pandemic scenarios like COVID 19 to handle problems and stop the spread of the diseases. Because they can cause famines and droughts, plant diseases pose a serious threat to human survival

1.2 PURPOSE

Agriculture provides food to all the human beings even in case of rapid increase in the population. It is recommended to predict the plant diseases at their early stage in the field of agriculture is essential to cater the food to the overall population. But it is unfortunate to predict the diseases at the early stage of the crops. The idea behind the project is to bring awareness amongst the farmers about the cutting-edge technologies to reduce diseases in plant leaf. Several factors associated with disease diagnosis in plants using deep learning techniques must be considered to develop a robust system for accurate disease management. However, despite the range of applications, several gaps within plant disease research are yet to be addressed to support disease management on farms. Thus, there is a need to establish a knowledge base of existing applications and identify the challenges and opportunities to help advance the development of tools that address farmers' needs.

The aim of our project is to identify

- what disease does our crop have,
- what is the cause of disease,
- how to prevent the disease,
- how to cure the disease,
- fertilizer recommendation to cure the disease

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

[1] This method used datasets to find diseased and healthy plant leaves. we introduced a deep convolutional neural network to identify crop series and diseases that may not be present in the plant tissue. The model trained on the test set has an accuracy of 99.35%. This process is enabled by deep learning, machine learning and digital epidemiologyA neural network associates images of diseased plants and crops as a pair. A neural network node is a mathematical function that receives numerical inputs from input edges and provides numerical outputs as output edges. We analyze 54,306 images of plant leaves that have been assigned a variance of 38 class labels. We resize the images to 256x256 pixels and perform both model optimization and prediction on these reduced images.

Adavantages: this system identifies the diseases that may not be present in the

plant issue. **Disadvantages:** analysing all the images of plant might be difficult

and time consuming. **Algorithm used:** deep convolutional neural network

Technologies Used: deep learning, machine learning and digital epidemiology

[2] This system explains about Plant identification system developed by computer vision researchers to know plant diseases. In this article, A network (CNN) can be used to gain an intuition of selected features based on a deconvolution network (DN) approach. Different order of veins is the best representative feature. We observe a multi-level representation of leaf data compared to that of contour shapes, showing hierarchical transformation of features. From a lower abstraction to a higher abstraction corresponding to the seed class. These insights gave us insight into the design of new hybrid feature extraction models that can continue to improve.

The uniqueness of the plant classification system.

Advantages: Features learned using deep learning can improve plant recognition performance

Disadvantages: defining featuresparts or patterns of an object in an image that help to identify.

Keywords: Plant recognition, deep learning, feature visualisation.

[3] This explains about the several ways to recognize plant medical condition. Some diseases have no visible symptoms, or takes effect too late to act, and Advanced analytics require Changes in symptoms exhibited by diseased plants. Evaluate the performance of the detection algorithm. To distinguish between diseased and healthy leaves, another class was added to the dataset.The source was removed using a developed Python script comparison procedure. Script will remove duplicates and Compare image metadata (name, size, date).

Advantages: datasets were introduced to detect each area of the leaf

(size,veins,thickness). **Diadavantages:** resolving image size less than 500px is not

considered.

[4] This proposed system explains about the water needs of plants vary from place to place due to changes in soil content, texture, climatic factors, and more. In addition to water requirements, plant diseases can also cause plants not to grow properly. In this article, we proposed a new intelligent irrigation system that can automatically control irrigation using an Android mobile application. In addition, photos of plant leaves are captured and sent to the cloud server. This is further processed and compared with images of diseased plant leaves in the cloud database. Based on the comparison, a list of suspected plant diseases is displayed to the user via an Android mobile application.

[5] The proposed method makes use of soil and PH samples as input and helps predict plants that can be recommended for soil and fertilizer that can be suitable. Information on the ground is collected by sensors and the data is transmitted from the Arduino via Zigbee and WSN (Wireless Sensor Network) to MATLAB. Analysis and processing of soil data are performed using ANN (Artificial Neural Neural Networks) and crop recommendations are carried out using SVMs (Support Vector Machines).

Advantage: It helps to improve production at field and income rates, improved crop prediction. **Disadvantages:** Crop sicknesses cannot be detected and prevented at earlier stage.

Algorithms used: ANN (Artificial Neural Network), SVM (Support Vector Machine) . **Hardware and Software:** Arduino, Zigbee, MATLAB, WSN.

[6] This paper presents a methodology for classifying three major leaf diseases of banana using local textural characteristics. Disease-affected regions are identified using image enhancement and color segmentation. The segmented image is transformed into one transform domain using three Image transforms (DWT, DTCWT, and ranklet transforms). Feature vectors are extracted from transform-domain images using LBP and its variants (ELBP, MeanELBP, and MedianELBP). Experimental results showed the best classification performance of ELBP features extracted from the DTCWT domain (accuracy 95.4%, accuracy 93.2%, sensitivity 93.0%, Fscore 93.0%, and specificity 96.4%).

Advantage: Compared with conventional methods of trait extraction, this new method of merging DTCWT with ELBP traits achieved a high level of accuracy in accurately detecting and classifying banana fungal diseases at an early stage.

Disadvantage: The plants which are at specific Euclidean distance can only be classified. **Algorithms used:** DWT (Discrete wavelet transform), LBP (Linear Binary Pattern)

[7] In this paper, the disease classification was initially performed by the International Center for Tropical Agriculture (CIAT) with banana images as input, which was transferred to the primary processing technique. A hybrid segmentation called the generalized variation fuzzy mean sum (TGVFCMS) was used segment the affected leaf area. After segmentation, the data is passed to CNN for final review classification. It has database more than 18,000 real photos of bananas in the CIAT image gallery. The dataset includes dry/aged leaves (DOL),HP and 700 balanced images of 5 major diseases such as banana Fusarium wilt of Banana (FWB), Black Sigatoka (BS), Xanthoma wilt or bacterial banana wilt (BBW),Yellow Sigatoka (YS) and banana pustulosis.

Advantage: It provides high accuracy and for disease diagnosis, these technologies reduce and eliminates the amount of time and money required to complete the project.

Disadvantage: Thereis no improved algorithm for an early warning by collecting the climatic changes.

Algorithm used: Convolutional Neural Network (CNN).

[8] The purpose of the paper is to raise awareness among farmers about cutting-edge technology that can prevent plant leaf disease. The approaches of machine learning and image processing with an accurate algorithm are identified to detect the leaf illnesses in the tomato plant as tomato is only a readily available vegetable. The K-means Clustering is introduced to divide the data space into Voronoi cells. Leaf sample borders are extracted with contour tracking. Multiple descriptors i.e., Discrete Wavelet Transform, Principal Component Analysis A gray-level co occurrence matrix is used to extract informative features of leaf samples. finally, the extracted features are classified using machine learning approaches such as support vector machines (SVM), Convolutional Neural Networks (CNN) and K-Nearest Neighbors (KNN). The accuracy of the proposed model is It was tested using SVM (88%), K-NN (97%) and CNN (99.6%) on disordered tomato samples.

Advantage: It provide better accuracy and automatically detects the leaf disease.**Disadvantage:** There is no huge amount of datasets for other leaf samples and the disease is not predicted at the early stage of operation.

2.2 REFERENCES

[1] Using Deep Learning for Image-Based Plant Disease Detection

S. Sankaran, A. Mishra, R. Ehsani, and C. Davis, “A review of advanced techniques for detecting plant diseases,” Computers and Electronics in Agriculture.

[2] How Deep Learning Extracts and Learns Leaf Features for Plant Classification

Sue Han Leea, Chee Seng Chan, corresponding authora, Simon Joseph Mayob, Paolo Remagninoc.

[3] Deep Neural Networks Based Recognition of Plant Diseases by Leaf ImageClassification Srdjan Sladojevic , 1 Marko Arsenovic, Andras Anderla, Dubravko Culibrk , and Darko

Stefanovic. Department of Industrial Engineering and Management , Faculty of Technical Sciences University of Novi Sad , Trg Dositeja Obradovica 6 , 21000 Novi Sad, Serbia .

[4] Cloud based automated irrigation and plant leaf disease detection system using an android application

O. T. Zaheema ,Department of Computer Science Engineering, Eranad Knowlededge City Technical Campus, Manjeri, Kerala, India

[5] Agro based crop and fertilizer recommendation system using machine learning Preethi G , Rathi Priya V , Sanjula S M ,Lalitha S D , Vijaya Bindhu B.

Final Year CSE, R.M.K Engineering College , rath16309.cs@rmkec.ac.in ,

Assistant Professor , R.M.K. Engineering College , sdi.cse@rmkec.ac.in ,

Cognizant Technology Solutions - 2020.

[6] Foliar fungal disease classification in banana plants using elliptical local binary pattern on multiresolution dual tree complex wavelet transform domain

Deepthy Mathew, C. Sathish Kumar, K. Anita Cherian. Department of Electronics and Communication Engineering , Rajiv Gandhi Institute of Technology , APJ Abdul Kalam Technological University, Kottayam 686501, India . Department of Electronics and Communication Engineering , Government Engineering College, Idukki 685603, India .

Department of Plant Pathology , College of Horticulture , Kerala Agricultural University, Thrissur 680656, India-2020.

[7] An automated segmentation and classification model for banana leaf disease detection V. Gokula Krishnan1 , J. Deepa , Pinagadi Venkateswara Rao , V. Divya , S. Kaviarasan.Associate Professor , CSIT Department , CVR College of Engineering , Hyderabad , India.Assistant Professor , CSE Department , EaswariEngineering College , Chennai ,India 2022

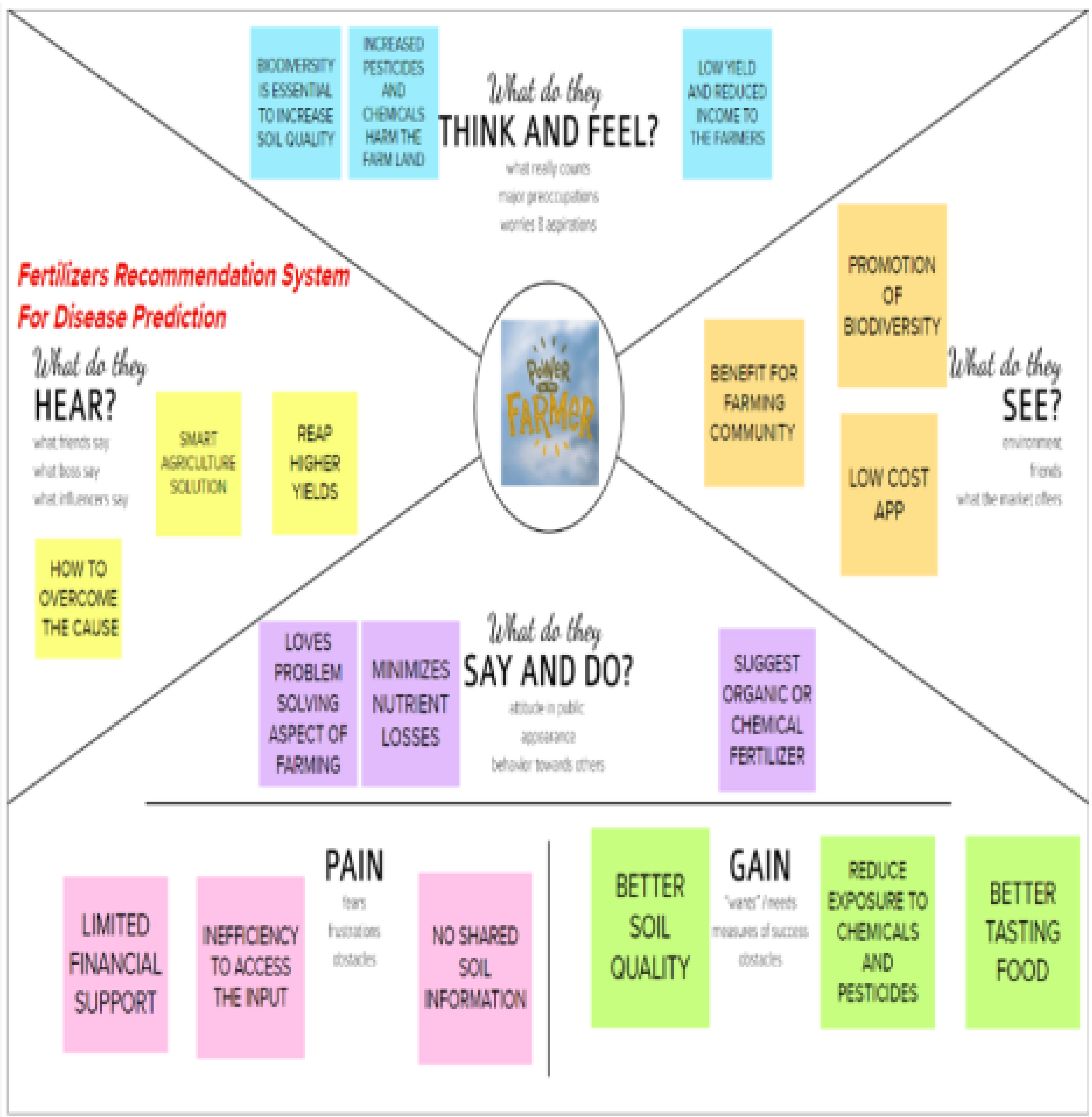
[8] Plant leaf disease detection using computer vision and machine learning algorithms Sunil S.Harakannanavar, Jayashri M. Rudagi , Veena I Puranikmath ,AyeshaSiddiqua ,Pramodhini.R.Department of ECE, Nitte MeenakshiInstitute,Yelahanka, Bangalore, Karnataka.

2.3 PROBLEM STATEMENT DEFINITION

QUESTION	DESCRIPTION
WHO DOES THE PROBLEM AFFECT ?	FARMERS,CUSTOMERS,ORGANIZATIONS
WHAT ARE THE BOUNDARIES OF THE PROBLEM ?	GEOGRAPHICAL LOCATIONS,COUNTRIES,FARM LANDS
WHAT IS THE ISSUE ?	PLANT DISEASES ARE THE COMMON REASON FOR LOW YIELD AND REDUCED INCOME TO THE FARMERS
WHEN DOES THE ISSUE OCCUR ?	DUE BACTERIAL INFECTION AND IMBALANCE OF A NUTRIENT IN THE SOIL
WHERE IS THE ISSUE OCCURING ?	FARMING LANDS,GRADENING,CROP FIELDS
WHY IS IT IMPORTANT THAT WE FIX THE PROBLEM ?	EASY IDENTIFICATION OF DISEASES IMPROVES THE QUALITY OF FOOD AND HIGH CROP YIELDS AND PREVENTS THE DISEASE
HOW DOES IT HELP TO FARMERS ?	HELPS TO YIELD MAXIMUM PRODUCTION OF CROPS

3. IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP



12

3.2 IDEATION AND BRAINSTROMING

1

Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

1.Identify the disease on plants using deep learning techniques and to recommend the fertilizers for reducing the diseases.
2.Provide website information for recommened fertilizer.

Key rules of brainstorming

To run an smooth and productive session

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

Mummaneni sravani

- Website for Fertilizer recommendation
- Identify the disease
- Determining best fertilizer
- User friendly website
- It reduces mam power
- Smart solution to solve the problem

Nalina.V

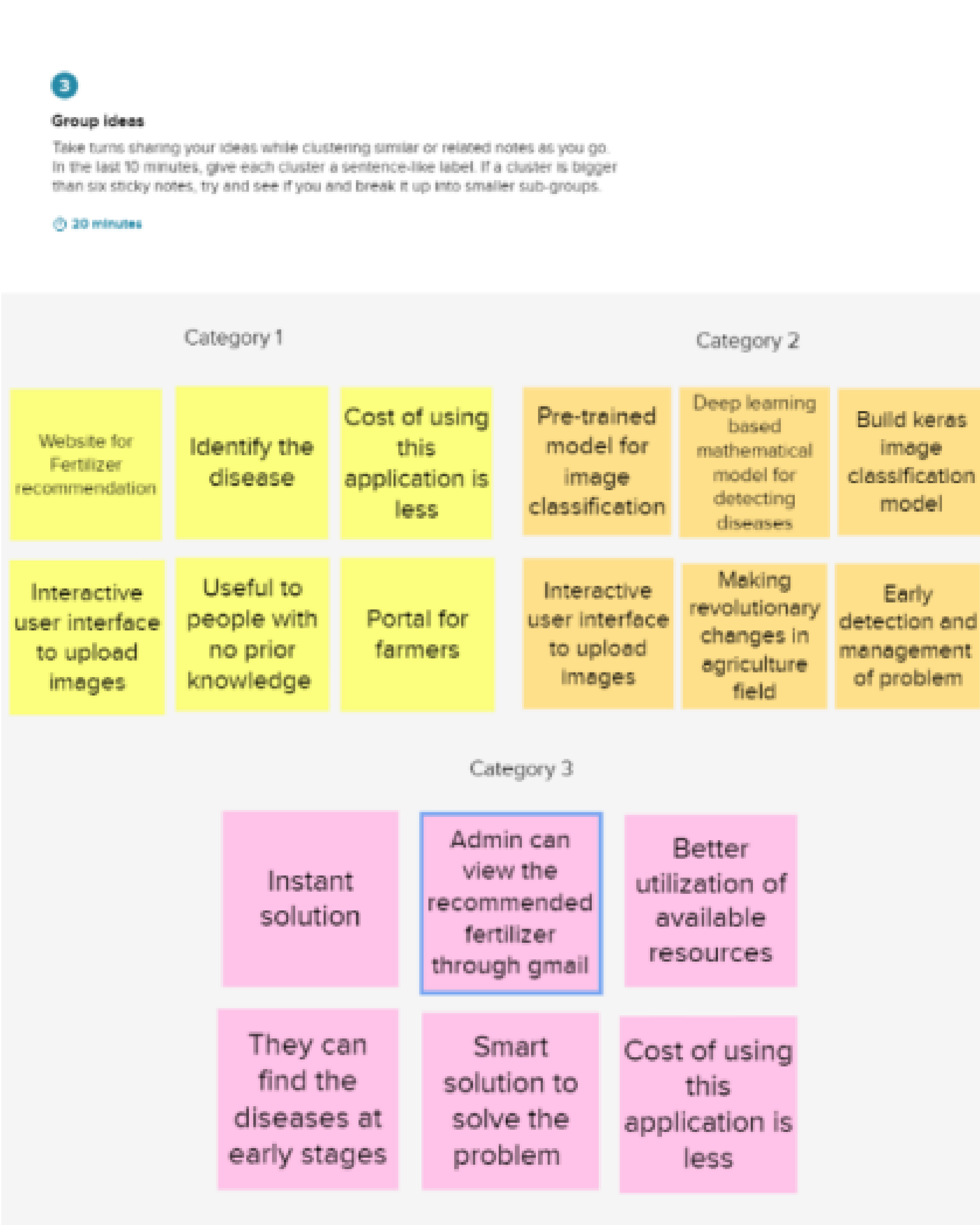
- Pre-trained model for image classification
- Build keras image classification model
- Making revolutionary changes in agriculture field
- It simplifies the farmers work
- Cost of using this application is less
- They can find the diseases at early stages

Konduru Theja Sree

- Deep learning based mathematical model for detecting diseases
- Early detection and management of problem
- Better utilization of available resources
- Interactive user interface to upload images
- Improves productivity
- Interactive user interface to upload images

Nandhini .M

- Fertilizer Recommendation
- Instant solution
- Useful to people with no prior knowledge
- Admin can view the recommended fertilizer through gmail
- It will save time
- Portal for farmers


3
Group ideas
Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.
20 minutes
Category 1
Website for Fertilizer recommendation
Identify the disease
Cost of using this application is less
Interactive user interface to upload images
Useful to people with no prior knowledge
Portal for farmers
Category 2
Pre-trained model for image classification
Deep learning based mathematical model for detecting diseases
Build keras image classification model
Interactive user interface to upload images
Making revolutionary changes in agriculture field
Early detection and management of problem
Category 3
Instant solution
Admin can view the recommended fertilizer through gmail
Better utilization of available resources
They can find the diseases at early stages
Smart solution to solve the problem
Cost of using this application is less

16

3.3 PROPOSED SOLUTION

S.No.	Parameter	Description

1.	Problem Statement (Problem to be solved)	In agricultural aspects, if the plant is aff ected by leaf disease, then it reduces the gro wth and productivity. Generally, the plant diseas es are caused by the abnormal physiological functionalities of plants. The issue occ urs in agriculture practicing areas, particular ly in rural regions.
2.	Novelty / Uniqueness	1. We have combined the features of CNN and a pre-trained model result ed in an improved performance in the prediction. 2. Data is fed to the CNN. And, its output is sent as the input to our p re trained model ResNet50. This increased our model's prediction accuracy to be above 85%
3.	Social Impact / Customer Satisfaction	• By recommending the appropriate fertilizers for the diseases predic ted, the quality of food products impr oves. It also controls the disease in pla nts.

• This also maximizes the crop yield by using the land efficiently.

4.	Business Model (Revenue Model)	• Helps farmers to grow more food using fewer resources by reducing the damage caused by irrelevant fertilizers and diseases attacked. • With the proposed model crop yield, farm efficiency, agricultural product output will be increased • A high gain can be seen in the agricultural output and profit will be increased.
5.	Scalability of the Solution	• Deep learning technique is used to identify the diseases and better fertilizer suggestions that can be recommended for those diseases. • Using Deep Learning techniques for recommendation reduces the time taken to detect diseases than other traditional methods.

3.4 PROBLEM SOLUTION FIT

1.CUSTOMER SEGMENTS

Farmers are the customers who are going to use this application. Farmers can interact with the portal build. Interacts with the user interface to upload images of diseased leaf. Our model-built analyses the Disease and suggests the farmer with fertilizers are to be used.

6.CUSTOMER LIMITATIONS

It may lead to wrong prediction Recommended fertilizer may not be available in the user's location

5.AVAILABLE SOLUTIONS

Non efficient image processing algorithms were used in earlier systems.This traditional approach gives lower accuracy and is time consuming. This drawback of the existing system propelled us towards the idea for developing a system that could ease this effort

2.PROBLEM/PAINS

The existing system only identifies the disease but does not recommend the remedy to be taken for the disease. It leads to wrong prediction. Recommended fertilizer may not be available in the user's location It may lead to wrong prediction.

9.PROBLEM ROOT/CAUSE

Infected seed, soil crop debris Infectious plant disease are caused by pathogenic organisms such as fungi, bacteria, viruses as well as insects Through the movement of contaminated soil, machinery, animals and other plant material

7.BEHAVIOR

Non efficient image processing algorithms were used in earlier systems. This traditional approach gives lower accuracy and is time consuming. In our project we identify the plant diseases using CNN with ResNET50 we have used. Then it recommends the fertilizer to be used. Comparing to other projects our project's accuracy is more because we are using CNN with ResNET50

3.TRIGGERS TO ACT

We have combined the features of CNN and a pre-trained model resulted in an improved performance in the prediction. Data is fed to the CNN and, its output is sent as the input to our pre- trained model ResNet50. This increased our model's prediction accuracy to be above 85%.

4.EMOTIONS

We are going to develop a userfriendly web application. Our algorithm gives the best accuracy in identifying the plant disease and recommending the fertilizer.

10.YOUR SOLUTION

In other projects it detects disease of only one color using basic CNN. In our project we identify the plant diseases using CNN with ResNET50 we have used. Then it recommends the fertilizer to be used. Comparing to other projects our project's accuracy is more because we are using CNN with ResNET50

8.CHANNELS OF BEHAVIOR

We have combined the features of CNN and a pre-trained model resulted in an improved performance in the prediction. Data is fed to the CNN and, its output is sent as the input to our pre- trained model ResNet50.
This increased our model's prediction accuracy to be above 85%
So, it helps to identify the disease in the earlier stages itself which reduces the huge impact on economic loss.

4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

Following are the functional requirements of the proposed solution.

FR No.	Functional	Sub Requirement

	Requirement	
FR-1	User Registration	Registration through form Registration through Gmail Registration through LinkedIn
FR-2	Image Capture	Take image of a leaf Check the leaf is captured under given parameters
FR-3	Image Processing	Upload the leaf image Click the predict button
FR-4	Updated Native Language	Languages can be changed according to the user, which he is more understandable with. (Ex: English, Hindi, Tamil)
FR-5	Leaf Prediction	Add the pesticides and fertilizers to be used for an unhealthy leaf
FR-6	Image Description	Show the prescribed fertilizer and description of the disease for curing a unhealthy leaf
FR-7	Providing Datasets	Training datasets Tes
FR-8	Adding Datasets	Fruit datasets for fruits Vegetable datasets for vegetables

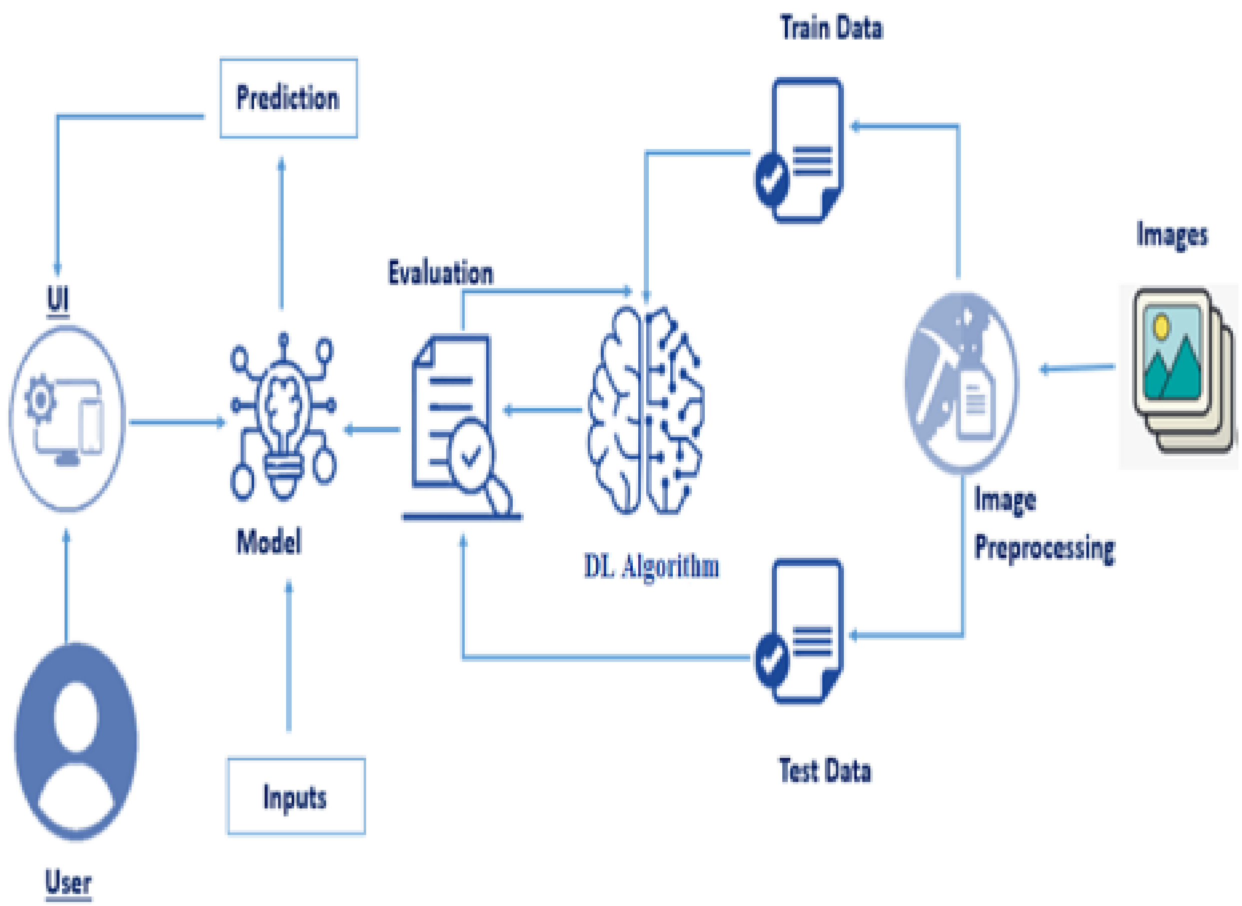
4.2 NON - FUNCTIONAL REQUIREMENTS

Following are the non-functional requirements of the proposed solution.

NFR No.	Non-Functional Requirement	Description
NFR-1	Usability	Leaf datasets can be used for detection of all kind of leafs Datasets can be reusable Data sets can be prepared according to the leaf
NFR-2	Security	User information and leaf data are secured The algorithms used are more secure
NFR-3	Reliability	The leaf quality is more The datasets and image capturing performs consistently well
NFR-4	Performance	Leaf problem defines once the leaf is detected Performs well according to the quality of leaf provides certain cure to it.
NFR-5	Availability	Quality of leaf will be used again for detection Available and easy access of datasets provided
NFR-6	Scalability	Increase in growth of predicting the results and defining a leaf

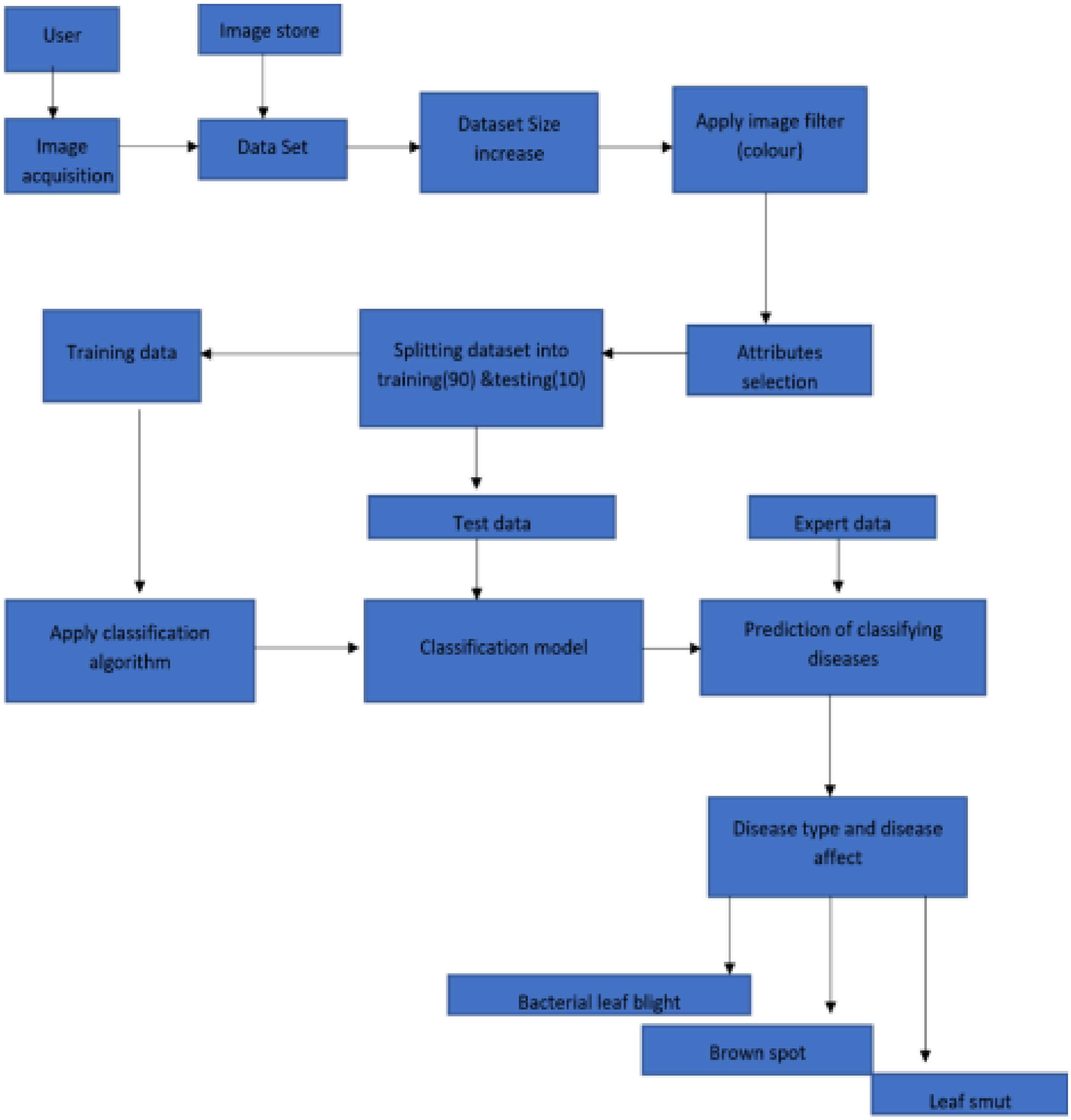
5.PROJECT DESIGN

An automated system is introduced to identify different diseases on plants by checking the symptoms shown on the leaves of the plant. Deep learning techniques are used to identify the diseases and suggest the precautions that can be taken for those diseases.



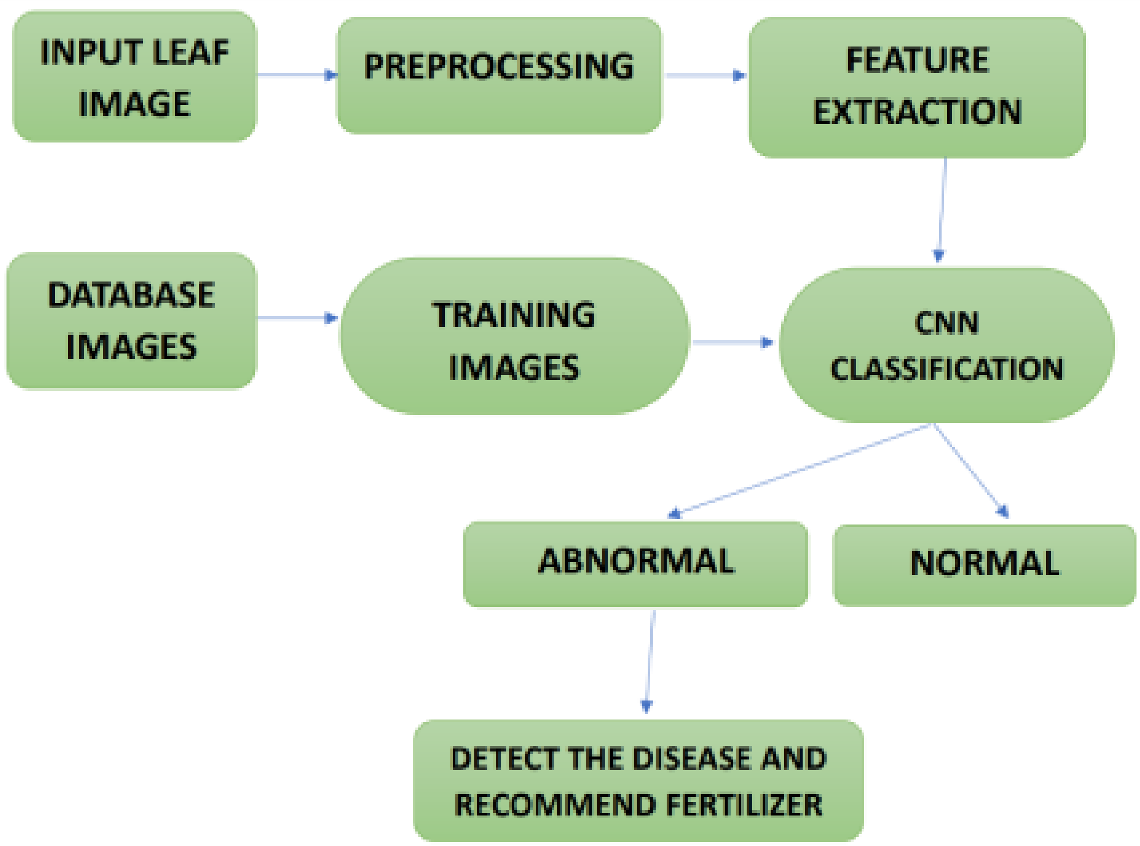
22

5.1 DATA FLOW DIAGRAMS



5.2 SOLUTION & TECHNICAL ARCHITECTURE

SOLUTION ARCHITECTURE:



TECHNICAL ARCHITECTURE:

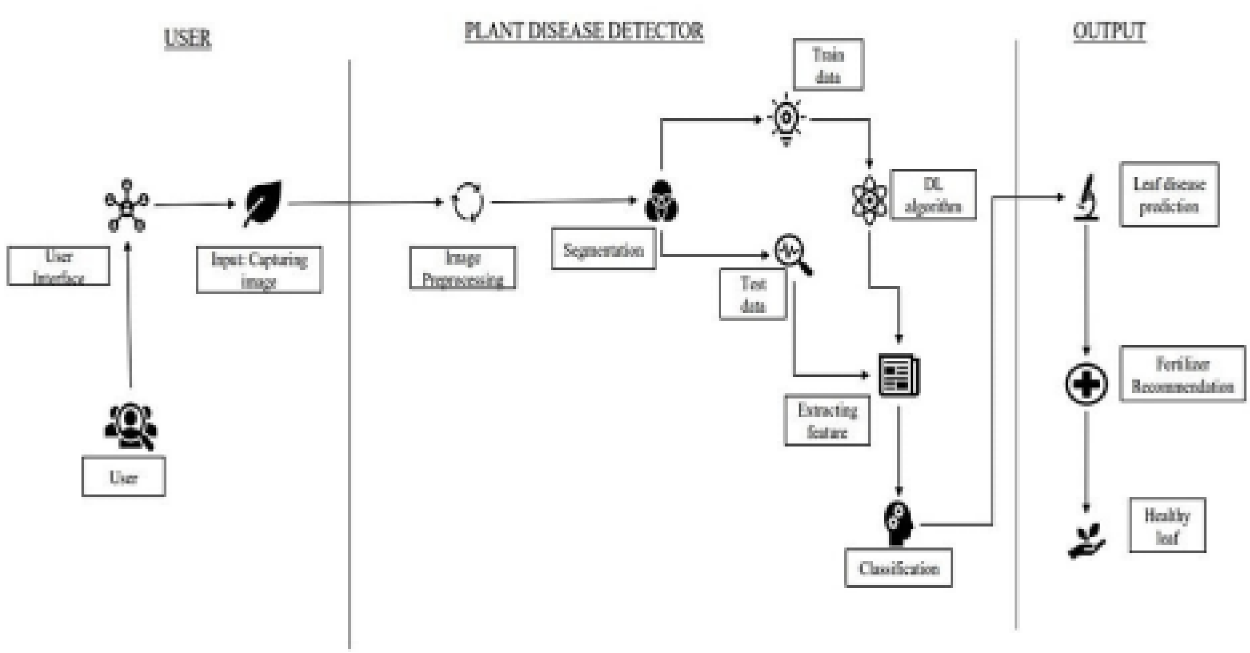


TABLE -1: COMPONENTS & TECHNOLOGIES:

S.NO	Component	Description	Technology
1,	User Interface	How user interacts with the website.	HTML,CSS, etc,.
2,	Disease Prediction	Here the disease in the leaf is predicted	Keras,CNN.
3.	Fertilizer Recommendation	The fertilizer is recommended for the predicted disease	User interface, HTML, CSS.
4.	Dataset	The training and testing data are collectively stored	Kaggle.com, data.gov, UCI machine learning repository, etc.
5.	File Storage	File storage requirements	IBM, Local File system.
6,	Modules	Purpose of deep learning modules	Image Recognition Modules,etc.
7.	Infrastructure(Server)	Application development on Local System-local server configuration:	Local File system.

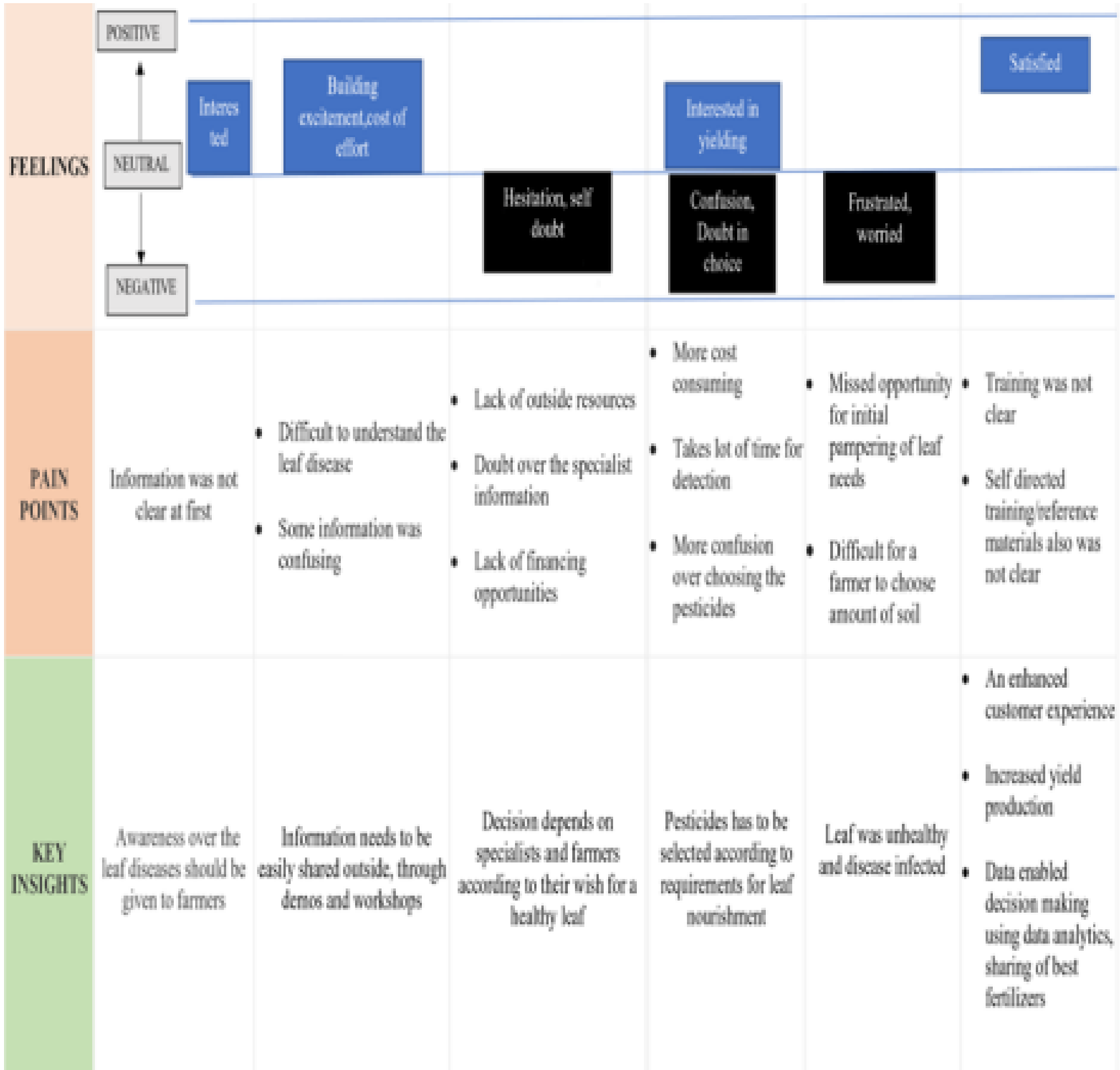
TABLE – 2: APPLICATION CHARACTERISTICS:

S.NO	Characteristics	Description	Technology
1.	Opensource Framewo rk	List of the opensource framework used	Open source-PyChar m, anaconda navigator, flask framework.
2.	Login	List of the access co ntrol implementation	Security - OWASP
3.	Scalable Architecture	Justify the scalable architecture	PyCharm
4.	Availability	Justify the availabilit	Web application ac

		y of website	cess to all.
5.	Performance	Design consideration fo r the performance of the web site	Convolutional Ne ural Networks.

5.3 USER STORIES

STAGES	AWARENESS	INFORMATION GATHERING	DECISION MAKING	PESTICIDE SELECTION	BEFORE DETECTION	AFTER DETECTION
GOALS	Understand what type of leaf disease possibilities exist	Learning	Setting criteria for Healthy leaf	Complete knowledge about pesticides and achieve high yield production	Leaf with high possibility of diseases	A well treated and healthy leaf without any disease
ACTIONS	Sees a demo leaf with high infection which has to be treated	Know about all the healthy and unhealthy leaf and talk to the specialist	• Compares healthy leaf possibilities to the unhealthy one and makes a decision • Refer to the leaf family	Knowledge about which leaf should be treated with what kind of fertilizers	• Check leaf condition • Check the weather condition • Check the soil condition	• Treats the leaf with suitable fertilizer as suggested • Makes sure of the suitable soil and weather condition
TOUCH POINTS	• Information provided at research • Interactions with the specialists at the research center	Verify the information provided at research	Information that can be asked/known with others for good healthy leaf production	Checking the pesticide quality and cost	Get to know the knowledge about leaf and its diseases	Training all leafs with good references or by using good learning materials



6. PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data collection and preproces	USN-1	Collecting plant disease dataset	2	Low	Mummaneni Sravani

	sing					
Sprint-1		USN-2	Labelling the dataset according to class	3	Medium	Nalina.V
Sprint-1		USN-3	38 types of plant diseases is labeled accordingly	2	Medium	Mummanen i Sravani
Sprint-1		USN-4	Data set Will contain both healthy and diseased data	1	Low	Nandhini.M
Sprint-1	Preprocessin g	USN-5	To prepare raw data in a format that the network can accept	2	High	Nalina.V
Sprint-1		USN-7	Shear range image will be distorted along an axis, mostly to create or rectify the perception angle	3	High	Nalina.V, Nandhini .M

Sprint-1 USN-8 Zoom Augmentation will randomly zoom the image and adds new pixels for the image

3 High Konduru Theja sree

Sprint-1		USN-9	Flipping the entire pixels of an image horizontally	3	High	Nalina . V
Sprint-2	Training , Testing and Creating a model	USN-10	Start initiating the model	3	Medium	Mummanen i Sravani
Sprint-2		USN-11	Adding different layers of cnn(convolution, pooling dense , flatten)	2	Medium	Nalina.V
Sprint-2		USN-12	Creating/compil ing with adam optimizer	1	Medium	Nandhini.M
Sprint-2		USN-13	Keras - Categorical Cross Entropy Loss Function for multi class classification	2	Medium	Konduru Theja sree
Sprint-2		USN-14	creating metrics	2	Medium	Nandhini .M
Sprint-2		USN-15	train the data with 20 epoch	3	High	Konduru Theja sree
Sprint-2		USN-16	testing the model	5	High	Mummanen i Sravani Konduru Theja sree

Sprint-2 USN-17 save the model 2 Medium Nalina .V

Sprint-3	Flask and Frame workdesign	USN-18	Creating backend framework with flask	8	High	Konduru Theja sree , Nandhini.M
Sprint-3		USN-19	importing the model file	3	Medium	Nandhini.M
Sprint-3		USN-20	Create route to link htmlRoutes and View Functions in Flask Framework index file	5	High	Mummanen i Sravani
Sprint-3		USN-21	Server Startup, requests and services in a loop	4	Medium	Konduru Theja sree ,
Sprint-4	Front end web application developmen t	USN-22	creating a html template with css file	8	High	Mummanen i Sravani, , Nalina.V
Sprint-4		USN-23	user can import diseased plant leaf in web page	2	Medium	Mummanen i Sravani , Konduru Theja sree
Sprint-4		USN-24	predicting what is the type of disease	2	Medium	Nalina.V , Nandhini.M

			occurred for the given input			
Sprint-4		USN-25	User can classify as healthy or diseased	2	Medium	Mummaneni Sravani
Sprint-4		USN-26	if plant has disease then suggest fertilizer and pesticides	3	Medium	Mummaneni Sravani

Sprint-4 USN-27 alert the admin about the prediction with the gmail 3 Medium Nalina.V

SCHEDULE

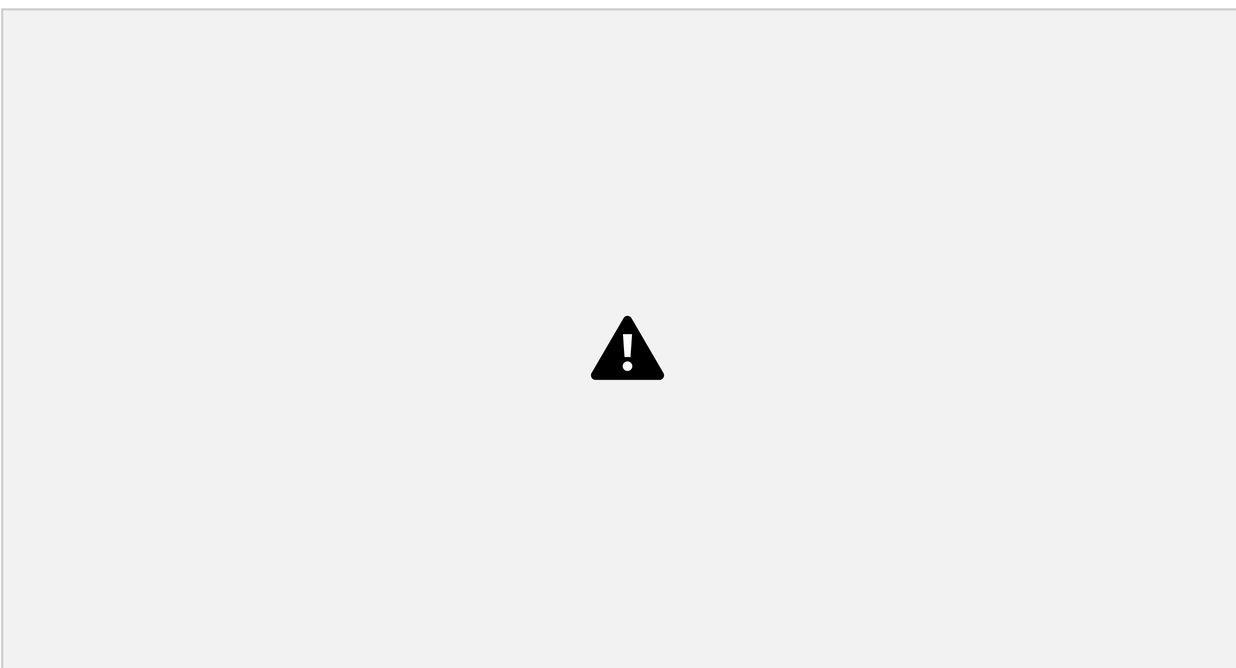
6.2 SPRINT DELIVERY

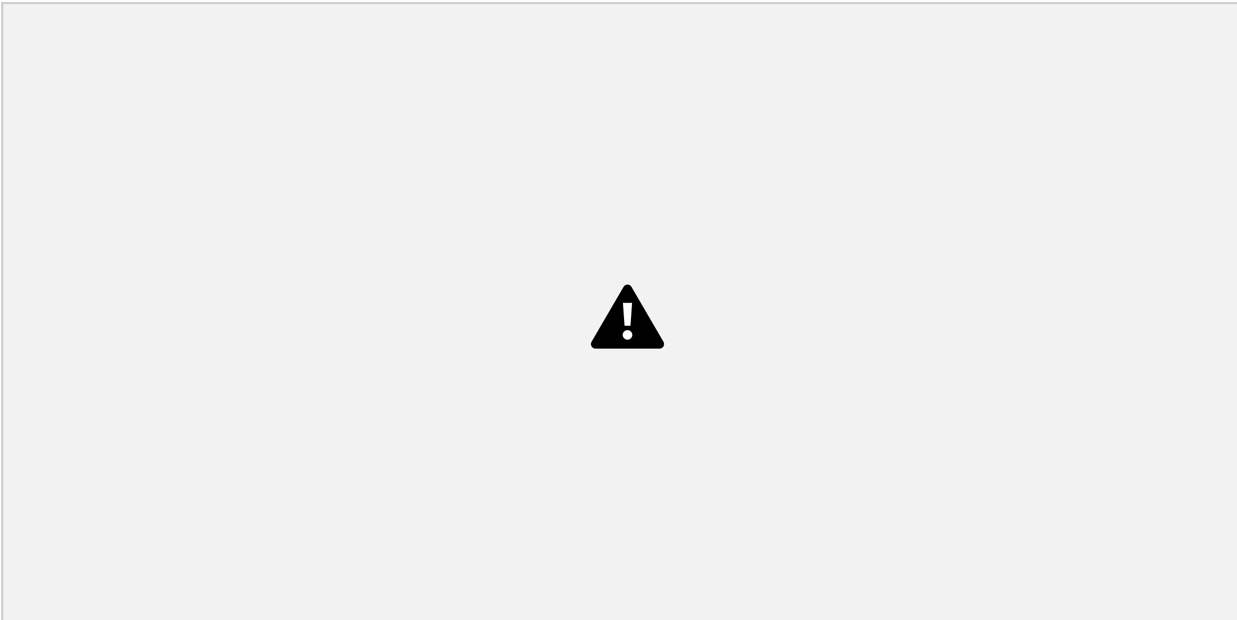
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)

Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	27 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	3 Nov 2022
Sprint-3	20	6 Days	07Nov 2022	12 Nov 2022	20	10 Nov 2022
Sprint-4	20	6 Days	14Nov 2022	19 Nov 2022	20	17 nov 2022

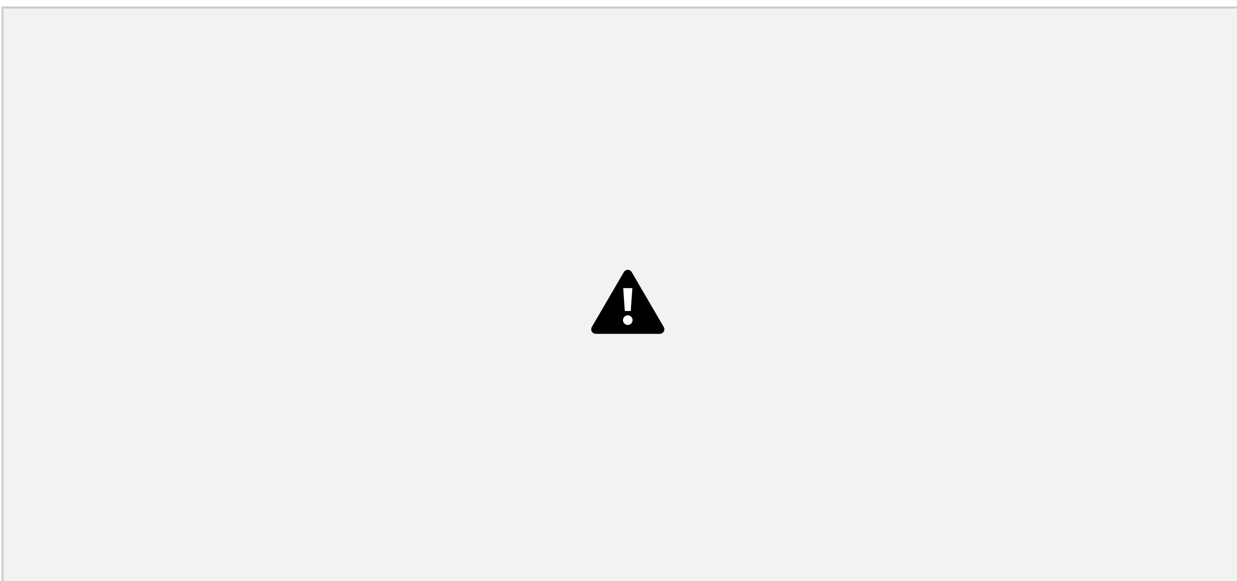
6.3 REPORTS FROM JIRA

TEAM JIRA ROADMAP





32



TEAM JIRA BOARD

33

34

7. CODING & SOLUTIONING

7.1 FEATURE 1

DATASET

Two datasets will be used, we will be creating two models one to detect vegetable leaf diseases like tomato, potato, and pepper plants and the second model would be for fruits diseases like corn, peach, and apple.

Downloading the Plant Disease dataset from the below

link

https://drive.google.com/file/d/1fxs7ptI6zh7NTbCOZARKZ7AmYKjnprrY/view

IMAGE PREPROCESSING

Before training the model, you have to pre-process the images and then feed them on to the model for training. We make use of Keras ImageDataGenerator class for image pre-processing.

Image Pre-processing includes the following main tasks

• Import ImageDataGenerator Library.

• Configure ImageDataGenerator Class.

• Applying ImageDataGenerator functionality to the trainset and test set.

Image data augmentation is a technique that can be used to artificially expand the size of a training dataset by creating modified versions of images in the dataset.

The Keras deep learning neural network library provides the capability to fit models using image data augmentation via the ImageDataGenerator class.

There are five main types of data augmentation techniques for image data;

specifically: • Image shifts via the width_shift_range and height_shift_range arguments. • The image flips via the horizontal_flip and vertical_flip arguments.

• The image rotates via the rotation_range argument

• Image brightness via the brightness_range argument.

• The image zooms via the zoom_range argument.

An instance of the ImageDataGenerator class can be constructed for train and test.

Image agumentation

```
from keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(rescale = 1./255, shear_range = 0.2,zoom_range
= 0.2,horizontal_flip =True)
test_datagen = ImageDataGenerator(rescale = 1)

x_train=train_datagen.flow_from_directory('/content/drive/MyDrive/fruit-dataset/fruit
dataset/train',batch_size=32,target_size=(128,128),color_mode='rgb',class_mode='ca
tegorical')

x_test=test_datagen.flow_from_directory('/content/drive/MyDrive/fruit-dataset/fruit
dataset/test',batch_size=32,target_size=(128,128),color_mode='rgb',class_mode='c
ategorical ')
```

OUTPUT:

Found 5384 images belonging to 6 classes.

Found 1686 images belonging to 6 classes.

MODEL BUILDING FOR FRUIT DISEASE PREDICTION

For model building we are following the below steps

- Import the libraries
- Initializing the model
- Add CNN layers
- Add dense layer
- Train and Save the model

IMPORT THE LIBRARIES

Here we have Imported the libraries that are required to initialize the neural network layer, and create and add different layers to the neural network model.

```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
```

INITIALIZING THE MODEL

Keras has 2 ways to define a neural network:

• Sequential
• Function API

Here we are using Sequential class . The Sequential class is used to define linear initializations onetwork layers which then, collectively, constitute a model.

We will use the Sequential constructor to create a model, which will then have layers added to it using the add () method.Now, will initialize our model.

Initialize the neural network layer by creating a reference/object to the Sequential class. **model=Sequential()**

ADD CNN LAYERS

We will be adding three layers for CNN
• Convolution layer
• Pooling layer
• Flattening layer

ADD CONVOLUTION LAYER

The first layer of the neural network model, the convolution layer will be added. To create a convolution layer, Convolution2D class is used. It takes a number of feature detectors, feature detector size, expected input shape of the image, and activation function as arguments. This layer applies feature detectors on the input image and returns a feature map (features from the image).

Activation Function: These are the functions that help us to decide if we need to activate the node or not. These functions introduce non-linearity in the networks.

model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activatio n='relu'))

Add the pooling layer

Max Pooling selects the maximum element from the region of the feature map covered by the filter. Thus, the output after the max-pooling layer would be a feature map containing the most prominent features of the previous feature map.

After the convolution layer, a pooling layer is added. Max pooling layer can be added using MaxPooling2D class. It takes the pool size as a parameter. Efficient size of the pooling matrix is (2,2). It returns the pooled feature maps. (Note: Any number of convolution layers, pooling and dropout layers can be added

model.add(MaxPooling2D(pool_size = (2,2)))

Add the flatten layer

The flatten layer is used to convert n-dimensional arrays to 1-dimensional arrays. This 1D array will be given as input to ANN layers.

model.add(Flatten()

ADD DENSE LAYERS

This step is to add a dense layer (output layer) where you will be specifying the number of classes your dependent variable has, activation function, and weight initializer as the arguments. We use the add () method to add dense layers. the output dimensions here is 6

```
model.add(Dense(40, 'relu'))
model.add(Dense(20, 'relu'))
model.add(Dense(6, 'softmax', ))
```


TRAIN AND SAVE THE MODEL

COMPILE THE MODEL

After adding all the required layers, the model is compiled, for this step, loss function, optimizer and metrics for evaluation can be passed as arguments

```
model.compile(optimizer='adam', loss ="categorical_crossentropy" ,
metrics =['accuracy'])
```

FIT AND SAVE THE MODEL

Fit the neural network model with the train and test set

```
model.fit(x_train,epochs=20,steps_per_epoch=89,validation_data =
x_test, validation_steps = 27)
```

The weights are to be saved for future use. The weights are saved in as .h5 file using save(). **model.save("fruit.h5")**

model.summary() can be used to see all parameters and shapes in each layer in our models

OUTPUT:

____

______ Layer (type) Output Shape Param #

===

=== conv2d (Conv2D) (None, 126, 126, 32) 896

max_pooling2d (MaxPooling2D (None, 63, 63, 32) 0
)

flatten (Flatten) (None, 127008) 0

dense (Dense) (None, 40) 5080360

dense_1 (Dense) (None, 20) 820

dense_2 (Dense) (None, 6) 126

===

=== Total params: 5,082,202

Trainable params: 5,082,202

Non-trainable params: 0

Epoch 1/20

89/89 [==============================] - 717s 8s/step - loss: 1.3023 - accuracy: 0.5609 - val_loss: 59.3136 - val_accuracy: 0.7199

Epoch 2/20

89/89 [==============================] - 354s 4s/step - loss: 0.6571 - accuracy: 0.7882 - val_loss: 60.1567 - val_accuracy: 0.7824

Epoch 3/20

89/89 [==============================] - 183s 2s/step - loss: 0.4134 - accuracy: 0.8615 - val_loss: 124.2583 - val_accuracy: 0.6863

Epoch 4/20

89/89 [==============================] - 108s 1s/step - loss: 0.3113 - accuracy: 0.8982 - val_loss: 615.5879 - val_accuracy: 0.4329

Epoch 5/20

89/89 [==============================] - 75s 836ms/step - loss: 0.2583 - accuracy: 0.9129 - val_loss: 541.0003 - val_accuracy: 0.4641

Epoch 6/20

89/89 [==============================] - 60s 673ms/step - loss: 0.2481 - accuracy: 0.9112 - val_loss: 663.6074 - val_accuracy: 0.4630

Epoch 7/20

89/89 [==============================] - 54s 599ms/step - loss: 0.2167 - accuracy: 0.9252 - val_loss: 504.1471 - val_accuracy: 0.4850

Epoch 8/20

89/89 [==============================] - 52s 584ms/step - loss: 0.2076 -

accuracy: 0.9274 - val_loss: 554.8959 - val_accuracy: 0.4618

Epoch 9/20

89/89 [==============================] - 51s 574ms/step - loss: 0.2308 -

accuracy: 0.9200 - val_loss: 591.8171 - val_accuracy: 0.4618

Epoch 10/20

89/89 [==============================] - 50s 564ms/step - loss: 0.1834 -

accuracy: 0.9402 - val_loss: 927.3312 - val_accuracy: 0.4028

Epoch 11/20

89/89 [==============================] - 50s 558ms/step - loss: 0.1923 -

accuracy:

0.9361 - val_loss: 726.0818 - val_accuracy: 0.4560

Epoch 12/20

89/89 [==============================] - 50s 556ms/step - loss: 0.1800 -

accuracy: 0.9417 - val_loss: 971.6089 - val_accuracy: 0.4120

Epoch 13/20

89/89 [==============================] - 50s 555ms/step - loss: 0.1912 -

accuracy: 0.9364 - val_loss: 635.0948 - val_accuracy: 0.5255

Epoch 14/20

89/89 [==============================] - 49s 553ms/step - loss: 0.1511 -

accuracy: 0.9479 - val_loss: 644.2640 - val_accuracy: 0.4965

Epoch 15/20

89/89 [==============================] - 50s 560ms/step - loss: 0.1573 -

accuracy: 0.9473 - val_loss: 631.8009 - val_accuracy: 0.5961

Epoch 16/20

89/89 [==============================] - 50s 556ms/step - loss: 0.1383 -

accuracy: 0.9508 - val_loss: 679.6030 - val_accuracy: 0.5475

Epoch 17/20

89/89 [==============================] - 50s 556ms/step - loss: 0.1452 -

accuracy: 0.9515 - val_loss: 903.8804 - val_accuracy: 0.4271

Epoch 18/20

89/89 [==============================] - 50s 556ms/step - loss: 0.1486 - accuracy: 0.9498 - val_loss: 661.1855 - val_accuracy: 0.6146

Epoch 19/20

89/89 [==============================] - 49s 553ms/step - loss: 0.1251 - accuracy: 0.9547 - val_loss: 1024.8682 - val_accuracy: 0.4225

Epoch 20/20

89/89 [==============================] - 50s 557ms/step - loss: 0.1289 - accuracy: 0.9568 - val_loss: 722.6586 - val_accuracy: 0.4907

MODEL BUILDING FOR VEGETABLE DISEASE PREDICTION

For model building we are following the below steps

- Import the libraries
- Initializing the model
- Add CNN layers
- Add dense layer
- Train and Save the model

IMPORT THE LIBRARIES

Here we have Imported the libraries that are required to initialize the neural network layer, and create and add different layers to the neural network model.

```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
```

from keras.layers **import** Flatten

INITIALIZING THE MODEL

Keras has 2 ways to define a neural network:

- Sequential
- Function API

Here we are using Sequential class . The Sequential class is used to define linear initializations of network layers which then, collectively, constitute a model.

We will use the Sequential constructor to create a model, which will then have layers added to it using the add () method.Now, will initialize our model.

Initialize the neural network layer by creating a reference/object to the Sequential class. **model=Sequential()**

ADD CNN LAYERS

We will be adding three layers for CNN

- Convolution layer
- Pooling layer
- Flattening layer

model.add(Convolution2D(32,(3,3),input_shape = (128,128,3),activation = 'relu'))

model.add(MaxPooling2D(pool_size = (2,2)))

```
model.add(Flatten())
```

ADD DENSE LAYERS

This step is to add a dense layer (output layer) where you will be specifying the number of classes your dependent variable has, activation function, and weight initializer as the arguments. We use the add () method to add dense layers. the output dimensions here is 6


```
model.add(Dense(300, 'relu'))

model.add(Dense(150, 'relu'))

model.add(Dense(75, 'relu'))

model.add(Dense(9, 'softmax', ))
```

TRAIN AND SAVE THE MODEL

COMPILE THE MODEL

After adding all the required layers, the model is compiled, for this step, loss function, optimizer and metrics for evaluation can be passed as arguments

```
model.compile(optimizer='adam', loss = "categorical_crossentropy" ,
metrics =['accuracy'])
```

FIT AND SAVE THE MODEL

Fit the neural network model with the train and test set

model.fit(x_train,epochs=20,steps_per_epoch=89,validation_data = x_test, validation_steps = 27)

The weights are to be saved for future use. The weights are saved in as .h5 file usi

ng save(). **model.save("veg.h5")**

TEST BOTH THE MODEL

Now that we have trained both the models, testing both the models by loading the saved models.

TEST THE MODEL

The model is tested with different images to know if it is working correctly

Import the packages and load the saved model

Import the libraries

from tensorflow.keras.preprocessing **import** image

from tensorflow.keras.preprocessing.image **import**

img_to_array **from** tensorflow.keras.models **import** load_model

import numpy **as** np

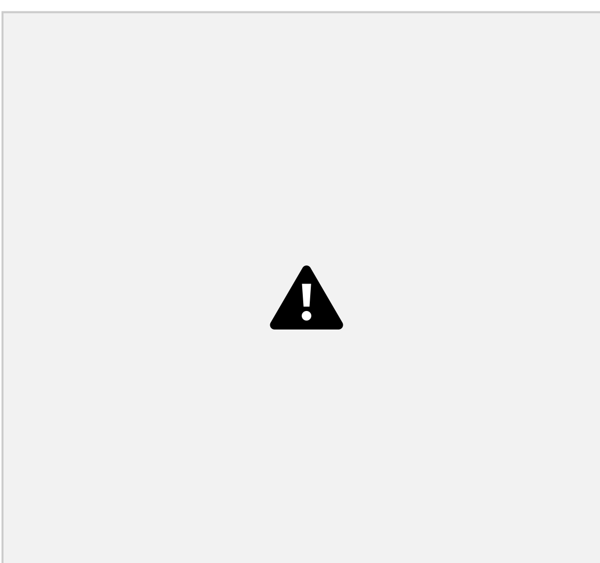
Initially, we will be loading the fruit model and test it with the vegetable model in a

similar way. model = load_model(**'fruit.h5'**)

Pre-processing the image includes converting the image to array and resizing according to the model.The pre-processed image is given to the model to know to which class your model belongs to.

img = image.load_img(r"/home/wsuser/work/Dataset Plant Disease/fruit-dataset/fruit dataset/test/Apple___healthy/0a553fc0-fc2c-4598-baba-3bc10191447c___RS_HL 5969.JPG", **target_size = (128,128))**

OUTPUT:



x=image.img_to_array(img)

```
x=np.expand_dims(x,axis=0)
result=model.predict(img)
print(result)
```

OUTPUT

The predicted class is 1

48

7.2 FEATURE 2

Application Building

After the model is built, we will be integrating it into a web application so that normal users can also use it. The new users need to initially register in the portal. After registration users can log in to browse the images to detect the disease.

In this section, you have to build

- HTML pages - front end
- Python script - Server-side script

Build Python Code

After the model is built, we will be integrating it into a web application so that normal users can also use it. The user needs to browse the images to detect the disease.

Activity 1: Build a flask application

Step 1: Load the required packages

```
from __future__ import division, print_function
```

```
import os

import numpy as np

import cv2

# Keras

from tensorflow.keras.models import load_model

from tensorflow.keras.preprocessing.image import img_to_array 49


# Flask utils

from flask import Flask, request, render_template

from werkzeug.utils import secure_filename
```

Step 2: Initializing the flask app and loading the model

flask applications must create an application instance. The web server passes all the requests it receives from clients to objects for handling using a protocol for WSG from flask import Flask app = Flask (__name__) (An application instance is an object of class Flask.)

```
app = Flask(__name__)

MODEL_PATH = 'fruit.h5'
```

MODEL LOADING

```
model = load_model(MODEL_PATH)

model.make_predict_function()
```

```
default_image_size = (128, 128)

abels=["Apple___Black_rot","Apple___healthy","Corn_(maize)___hea
lthy",
"Corn_(maize)___Northern_Leaf_Blight","Peach___Bacterial_spot","
Peach ___healthy"]

def convert_image_to_array(image_dir):

  try:

    image = cv2.imread(image_dir)
```



```
    if image is not None:

    image = cv2.resize(image, default_image_size)

  return img_to_array(image)

else:

   return np.array([])

except Exception as e:

print(f"Error : {e}")

return None

def model_predict(file_path, model):

  x = convert_image_to_array(file_path)

  x = np.expand_dims(x, axis=0)
```

```
preds = model.predict(x)
return preds
```

Step 3: Configure the home page

Routes and View Functions in Flask Framework Instance

Clients send requests to the webserver, in turn, sends them to the Flask application instance. The instance needs to know what code needs to run for each URL requested and map URLs to Python functions. The association between a URL and the function that handles it is called a route. The most convenient way to define a route in a Flask application is through the (app.route). Decorator exposed by the application instance, which registers the 'decorated

function,' decorators are python feature that modifies the behavior of a

function.

```
@app.route("/", methods=['GET'])
def index():
    return render_template("index.html", query="")
```

Step 4: Pre-process the frame and run

Pre-process the captured frame and given it to the model for prediction. Based on the prediction the output text is generated and sent to the HTML to display.

Request

To process incoming data in Flask, you need to use the request object, including mime-type, IP address, and data. HEAD: Un-encrypted data sent to server w/o response.

GET

Sends data to the server requesting a response body.

POST

Read form inputs and register a user, send HTML data to the server are methods handled by the route. Flask attaches methods to each route so that different view functions can handle different request methods to the same URL.

```
@app.route("/", methods=['GET', 'POST'])

def upload():

    if (request.method == 'POST'):

        f = request.files['file']
```



```
        basepath = os.path.dirname(__file__)

        file_path=os.path.join(basepath,'uploads',secure_filename(f.file

         name)) f.save(file_path)

         preds = model_predict(file_path, model)

         preds = np.argmax(preds)

          result = labels[preds]

         return render_template('index.html',

      prediction_text=result) return None
```

Server Startup - The application instance has a 'run' method that launches flask's integrated development webserver –

```
if __name__ == "__main__":
 app.run(debug=True)
```

Output:

```
* Serving Flask app 'app'

* Debug mode: on

* Running on http://127.0.0.1:500
```


Build HTML Pages

Html files and CSS files code documents related to front end design has been attached below.

Index.html

```
{% extends 'layout.html' %}
{% block body %}
<!-- banner -->
<section class="banner_w3lspvt" id="home">
        <div class="csslider infinity" id="slider1">
        <div class="banner-top1">
```

```
            <div class="overlay">

            <div class="container">

            <div class="w3layouts-banner-info text-center">

            <h3 class="text-wh">MyCrop-Plant Disease Prediction</h3>

         <h4 class="text-wh mx-auto my-4"><b>Get informed decisions about your f
arming strategy.<br>In Your Own Language.</b></h4>

         <h4 class="text-wh mx-auto my-4"><strong> Here are some questio
ns we'll answer</strong></h4>

 <p class="text-li mx-auto mt-2">

            1. Which disease do your crop have? <br>

            2. What cause the disease to plant? <br>

            3. How to prevent the disease?<br>

            4. How to cure the disease?<br>

           5.Fertilizer Recommended</p>

                                        </div>

                                 </div>
```



```
                    </div>

            </div></div>

</section>


<!-- //banner -->

<!-- core values -->

<section class="core-value py-5">

<div class="container py-md-4">

<h3 class="heading mb-sm-5 mb-4 text-center"> About Us</h3>

            <div class="row core-grids">

            <div class="col-lg-6 core-left"><br>
```

```
<img src="{{ url_for('static', filename='images/13.jpg') }}" class="img-fluid" alt="" /></d
iv> <div class="col-lg-6 core-right">
<h3 class="mt-4">Improving Agriculture, Improving Lives, Cultivating Crops To Make Fa
rmers Increase Profit.</h3>
<p class="mt-3">We use state-of-the-art machine learning and deep learning technologi
es to
help you to guide through the entire farming process. Make informed decisions to under
stand the
demographics of your area, understand the factors that affect your crop and keep them
healthy for a super awesome successful yield.</p>
                    </div>
                 </div>
        </div>
</section>
<!-- //core values -->
<!-- Products & Services -->
<section class="blog py-5">
<div class="container py-lg-5">
<h3 class="heading mb-sm-5 mb-4 text-center"> Our Services</h3>
```



```
<div class="row blog-grids">
<div class="col-lg-4 col-md-6 blog-left mb-lg-0 mb-md-5 pb-md-5 pb
-5"> <a href="{{ url_for('home') }}">
<img src="{{ url_for('static', filename='images/s35.jpg') }}" class="img-fluid" alt="
" /> <div class="blog-info">
<h4>Crop Disease</h4>
<p class="mt-1">Predicting the name of the disease through the plant l
eaf</p>                                  </div></a>
                 <br><br><br>
```

```
</div><div class="col-lg-4 col-md-6 blog-middle mb-lg-0 mb-md-5 pb-md-5 p
b-5"> <a href="{{ url_for('home') }}">
<img src="{{ url_for('static', filename='images/s6.jpg') }}" class="img-fluid" alt="
" /> <div class="blog-info">
<h4>Fertilizer Recommendation and Prevention</h4>
<p class="mt-1">Recommendation about the prevention step to the user to prevent the
disease in future.</p>
                                </div>
                            </a>
                    <br>
                <br>
    </div>
<div class="col-lg-4 col-md-6 blog-right mb-lg-0 mb-sm-5 pb-lg-5 pb-m
d-5"> <a href="{{ url_for('disease_prediction') }}">
<img src="{{ url_for('static', filename='images/s7.jpg') }}" class="img-fluid" alt=
""> <div class="blog-info">
<h4>Cause of Disease</h4>
<p class="mt-1">Predicting the cause of disease to the plant</p>
```



```
                                </div>
                            </a>
                        </div>
                    </div>
            </section>
        <style>
</style>
```

```
<!-- //Products & Services -->
</html>
{% endblock %}
```

Disease.html

```
{% extends 'layout.html' %} {% block body %}
<style>
  html body {
    background-color: rgb(206, 206, 228);
  }
</style>
<br />
<br />
  <h2 style="text-align: center; margin: 0px; color: black">
    <b>Find out which disease has been caught by your pla
nt</b>   </h2>
  <br />
```



```
  <br>
<div style="
    width: 350px;
    height: 50rem;
    margin: 0px auto;
    color: black;
    border-radius: 25px;
```

```
    padding: 10px 10px;
    font-weight: bold;
  ">
  <form class="form-signin" method=post enctype=multipart/form-data>
  <h2 class="h4 mb-3 font-weight-normal"><b>Please Upload The Image</b
></h2>
    <input type="file" name="file" class="form-control-file" id="inputfil
e" onchange="preview_image(event)" style="font-weight: bold;">
    <br>
    <br>
    <img id="output-image" class="rounded mx-auto d-block" />
    <button class="btn btn-lg btn-primary btn-block" type="submit" style="font-w
eight: bold;">Predict</button>
  </form>
</div>
<script type="text/javascript">
  function preview_image(event) {
    var reader = new FileReader();
    reader.onload = function () {
      var output = document.getElementById('output-image')
      output.src = reader.result;}
```



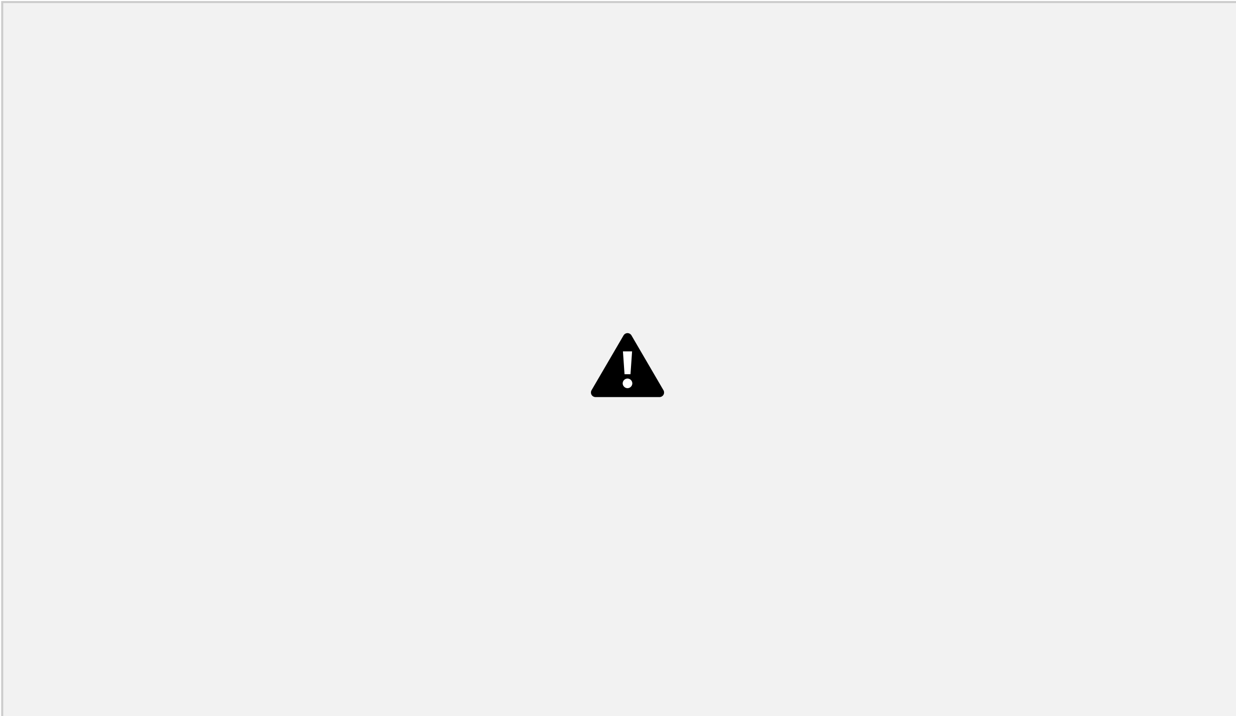
```
    reader.readAsDataURL(event.target.files[0]);}
</script>
</div>
{% endblock %}
```

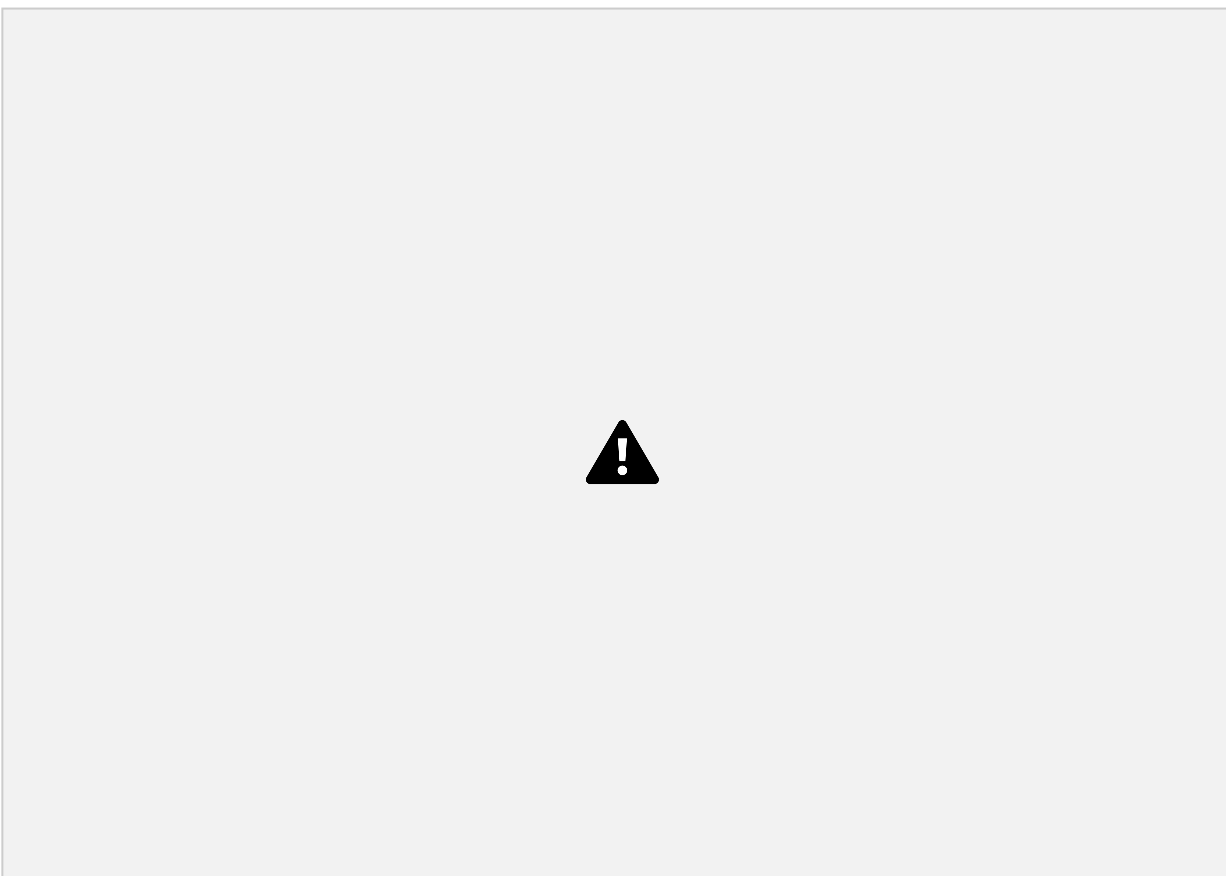
Disease-result.html

```
{% extends 'layout.html' %}
 {% block body %}
<div class="container py-2 mx-auto my-50 h-10 " style="margin: 9
rem;"   <div class="row">
     <div class="col-sm py-2 py-md-3">
        <div class="card card-body" style="justify-content: center; backg
round color:blanchedalmond">
          <p class="text-center" style="color: black; font-size: 22px;">{{ predicti
on }}              </p>
        </div>
     </div>
  </div>
</div>
{% endblock %}
```

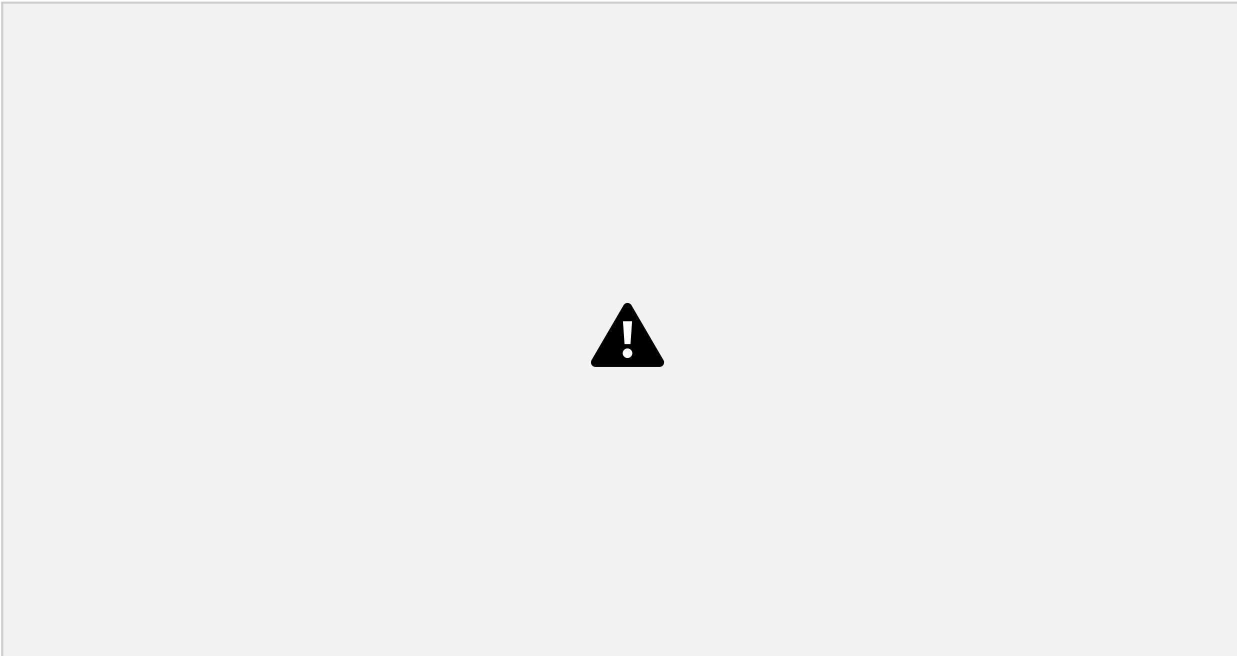

Build the UI where a home page will have details about the application, a prediction page where a user is allowed to browse an image and get the predictions

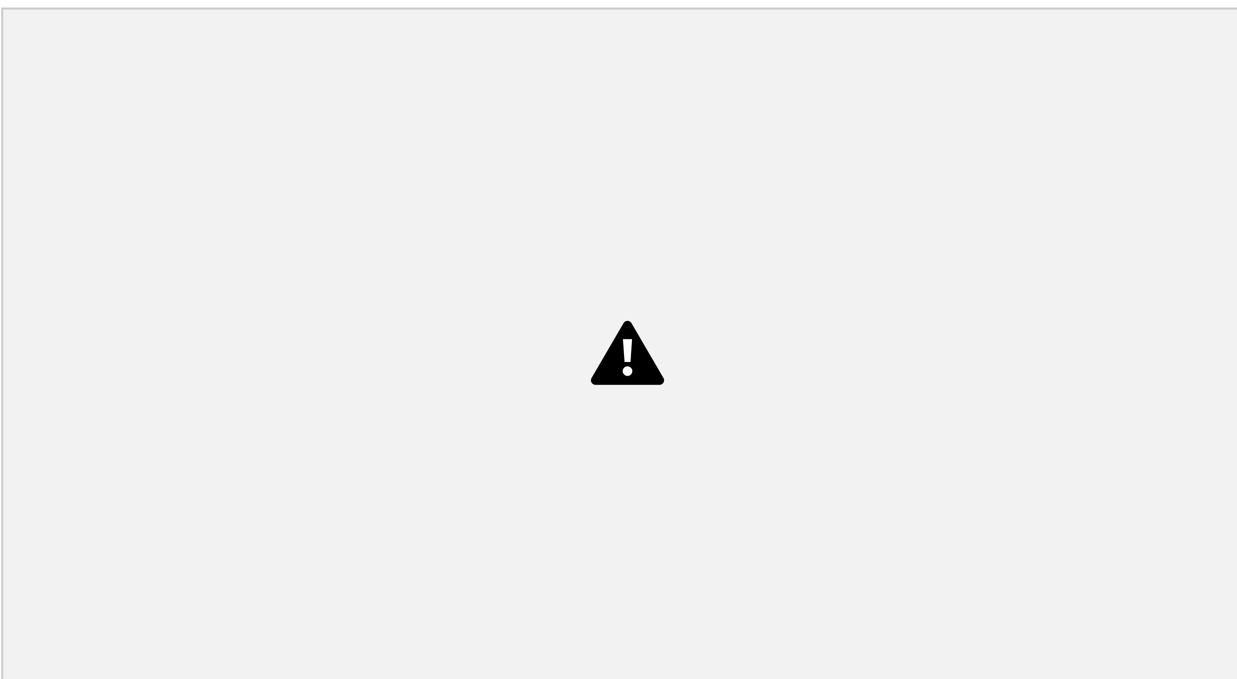
[WEB APPLICATION SCREEN SHOTS]





After clicking on disease button, you will be redirected to the find out which disease has been caught by your plant page where you can browse the images.





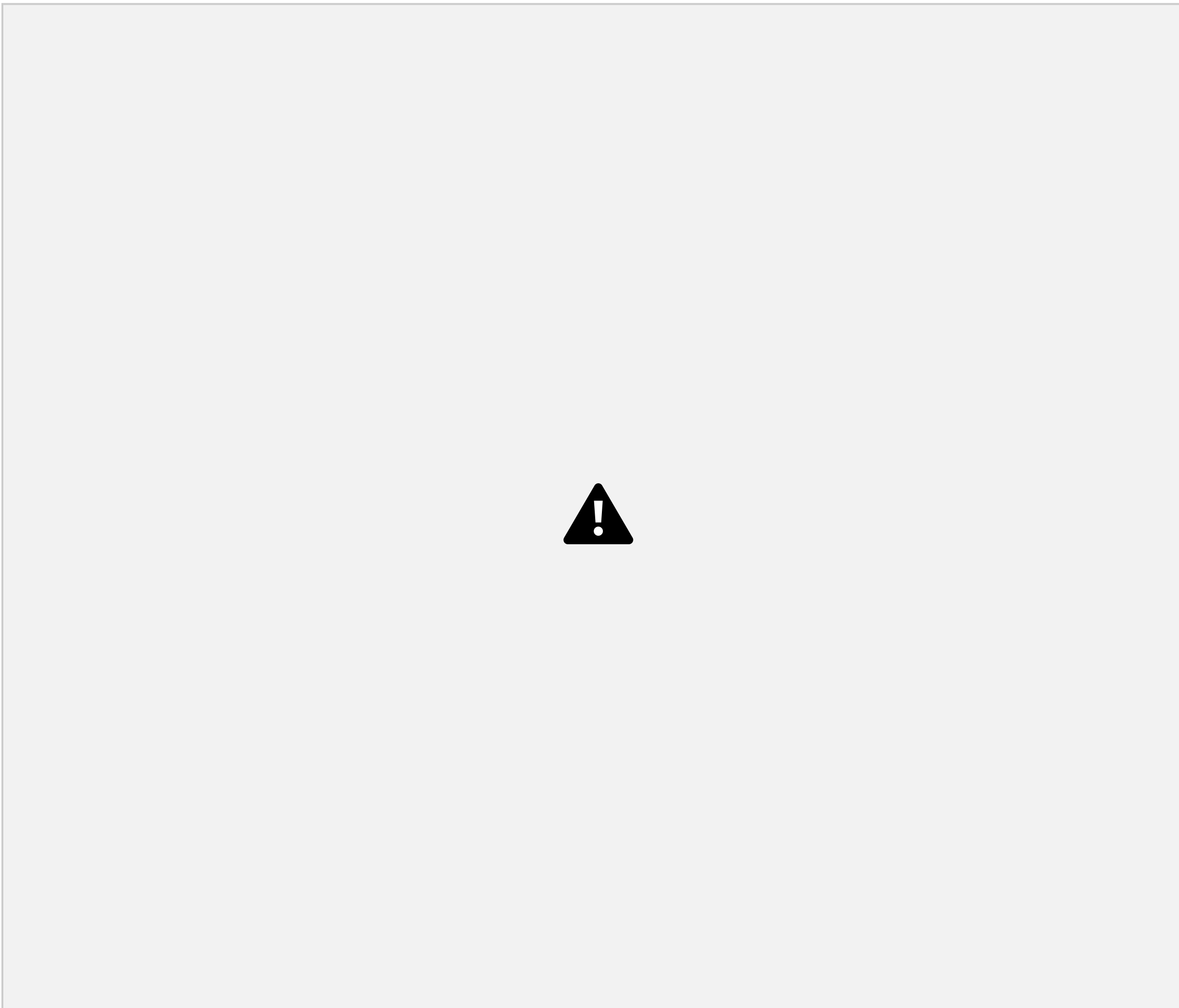
RUN THE CODE

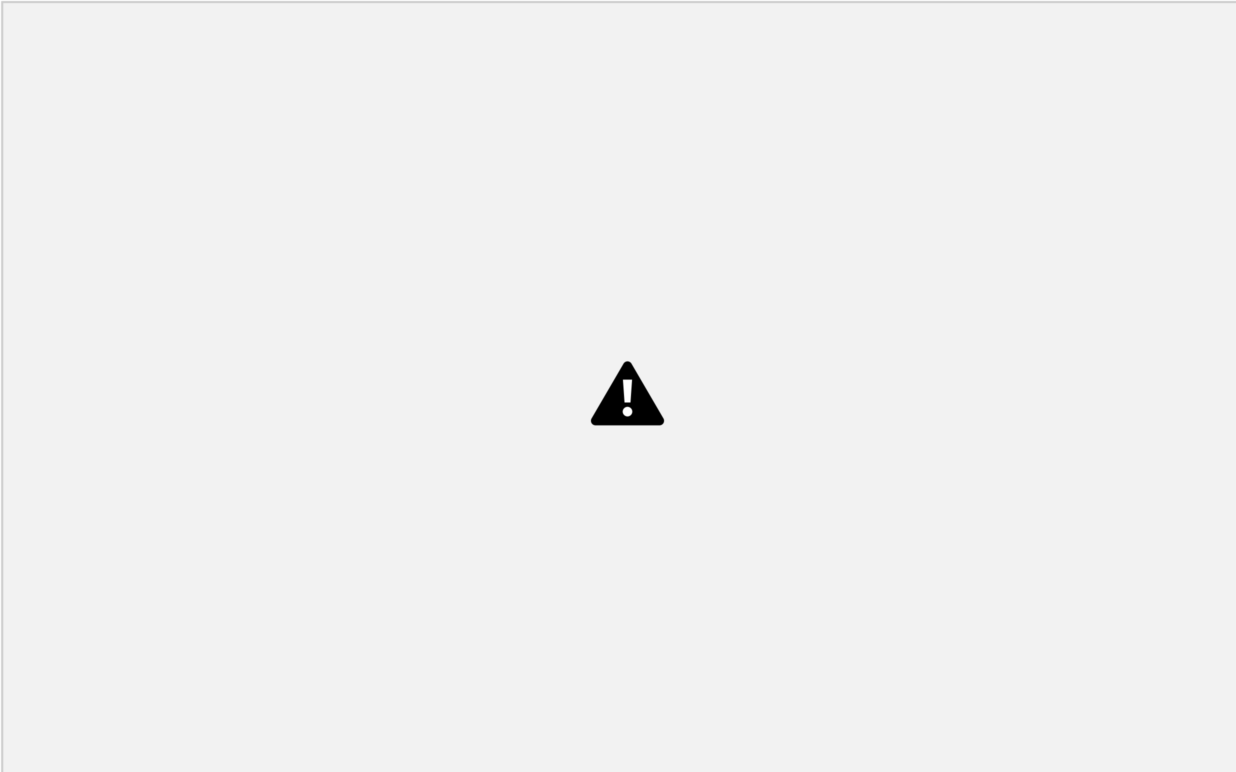
Step 1: Run the application.

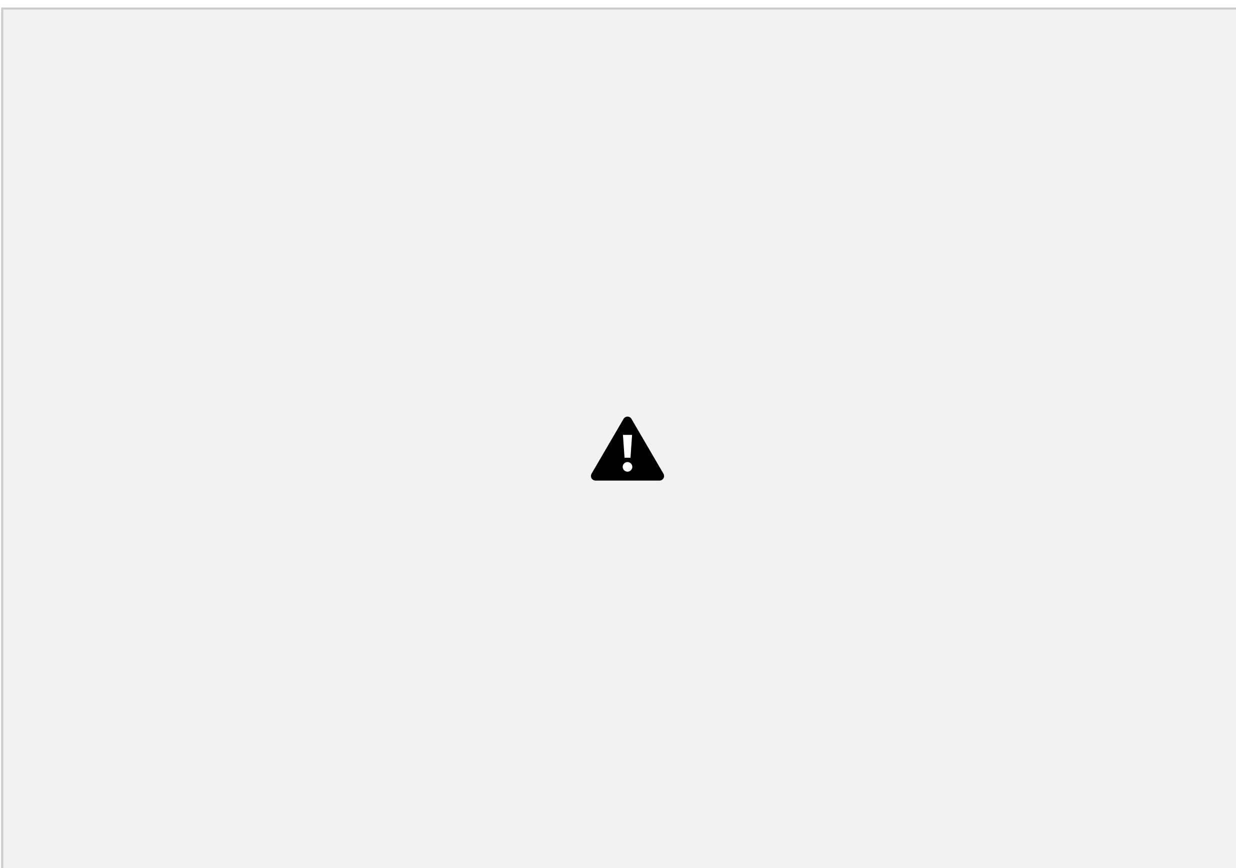
Open the browser and navigate to localhost to check your web application **Running**

the application on http://127.0.0.1:5000

Step 2: The home page looks like this







After clicking on disease button, you will be redirected to the find out which disease has been caught by your plant page where you can browse the images

8. TESTING

8.1 TEST CASES

TEST SCENARI	STEPS TO EXECUTE	TEST DATA	EXPECTED	ACTUAL RESULT	STATUS

O			RESULT		
Verify user is able to run theapplication by login to the home page	1. Click on the run.app 2. A link will be generated 3. click on the link provided to visit the home page	http://127.0.0.1:5000	Home page is displayed	Home page is displayed	pass
Verify the user can see the home page and see the diseases	1.Go to the homepage 2.Click on to the diseases 3. A predict button will be displayed to check the leaf diseases	http://127.0.0.1:5000	Predict button page will be displayed	Predict button page will be displayed	pass
Verify the user can see the leaf images by clicking the	1.Click the predict button 2. A list of images will be displayed	http://127.0.0.1:5000	Images of the diseased leaves has to be displayed	Images of the diseased leaves has to displayed	pass

predict button	3.Select a l eaf image that has to be predic ted 4.After the l eaf is predicted, the leaf has to determine the diseases.				
Verify the leaf disease is predicted correctly	1.The information has to provide correct disease 2.if the disease is correct test case is passed, or else the test case is fail.	http://127.0.0.1:5000	Successfully predicted the disease and displays the fertilizer recommended	Have successfully predicted the disease and correctly recommended the fertilizer.	pass

8.2 User Acceptance Testing

1.Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [ProductName] project at the time of the release to User Acceptance Testing (UAT).

2.Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	6	4	2	3	15
Duplicate	1	0	3	0	4
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	1	2	2	5

Totals 7 5 9 6 27

3.Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Client Application	5	0	0	5
Security	2	0	0	2
Final Report Output	4	0	0	4

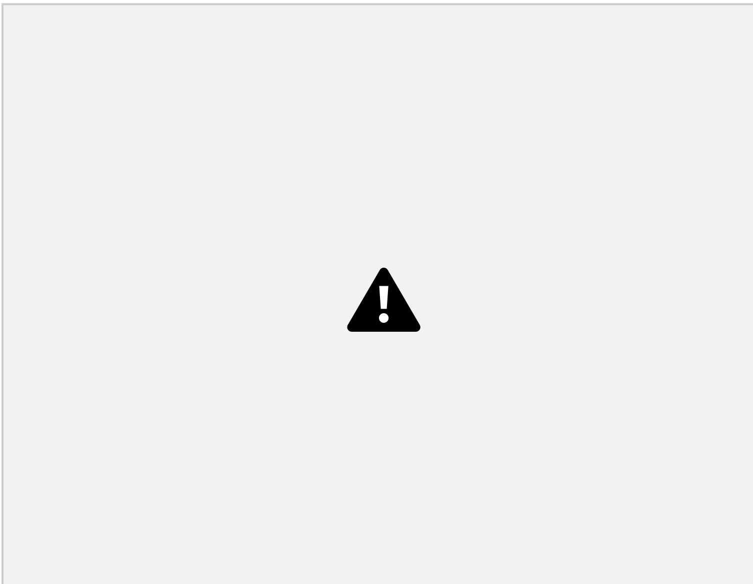
Version Control 2 0 0 2 **69**

RESULTS

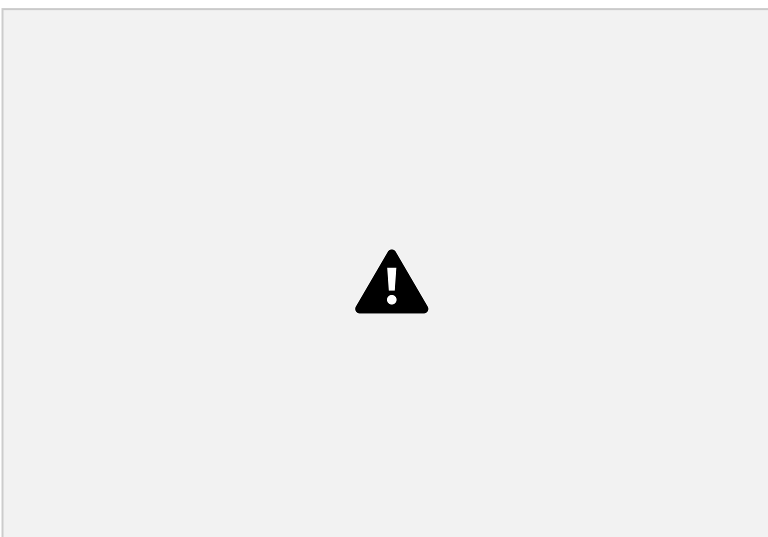
Performance Metrics

1. PARAMETER: MODEL SUMMARY

VEGETABLE



FRUIT

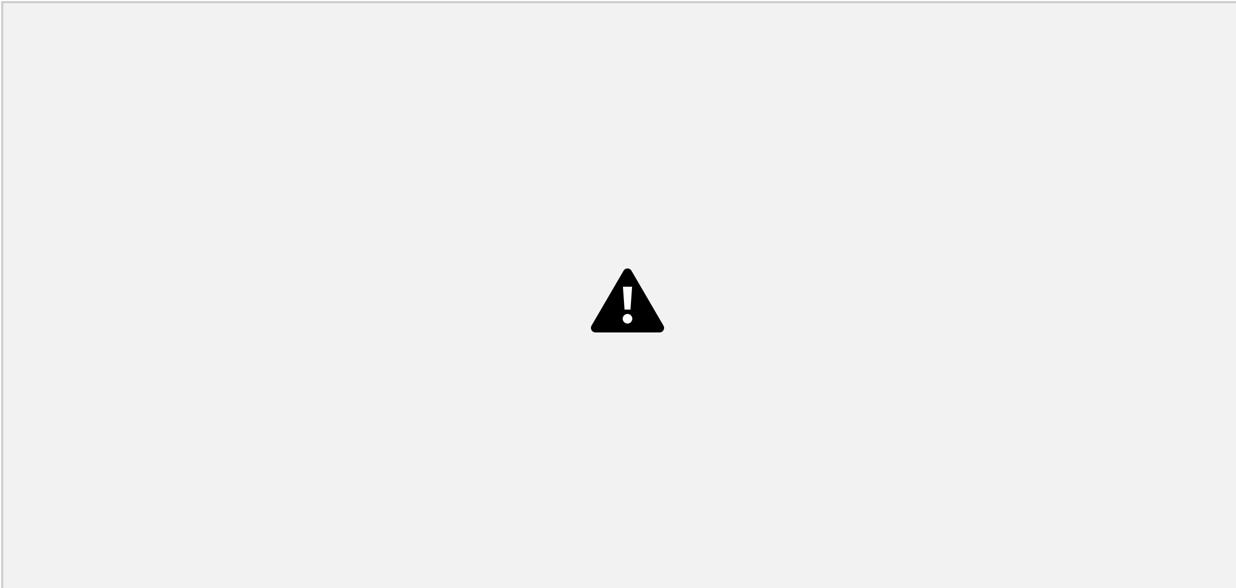


2.PARAMETER: ACCURACY

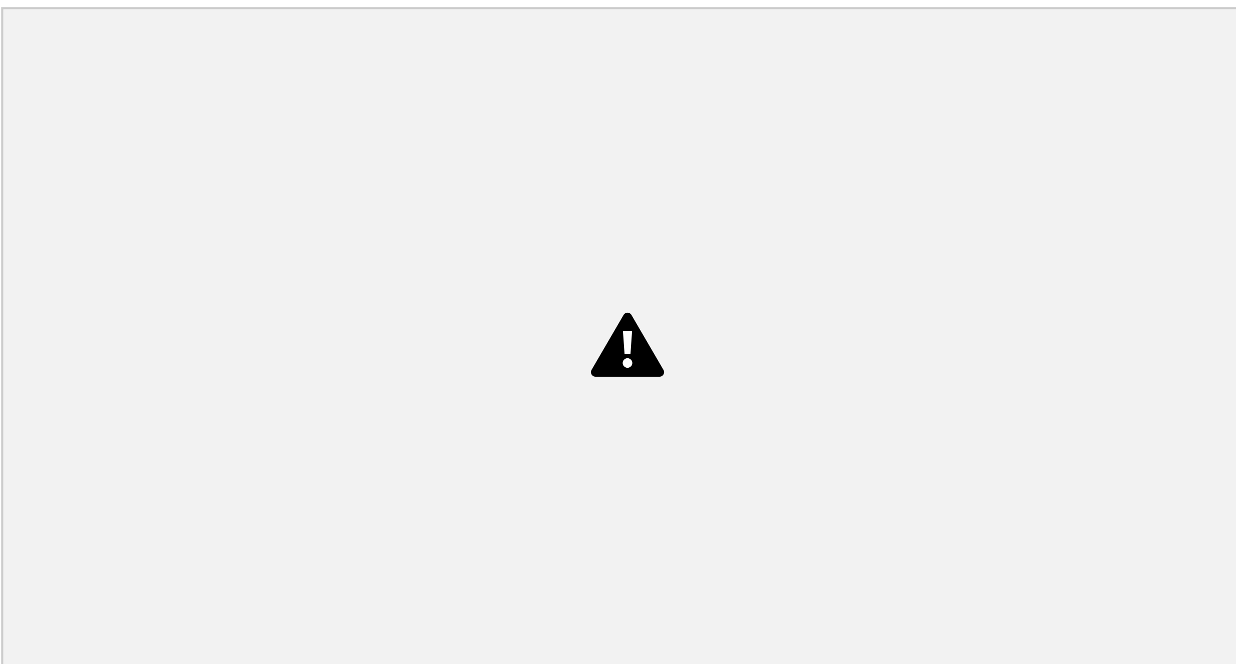
Training Accuracy -

Validation Accuracy -

VEGETABLE



FRUIT



3. Confidence Score (Only Yolo Projects) - NOT A YOLO PROJECT

OUTPUT SCREENSHOTS:

10.ADVANTAGES & DISADVANTAGES

ADVANTAGES

Farmers can interact with the portal build

• Interacts with the user interface to upload images of diseased leaf

• Our model-built analyses the Disease and suggests the farmer with fertilizers are to be used • It is easy to maintain.

• It is user-friendly.

• The system can easily detect the leaf from the image.

• It will also detect which type of leaf it is.

The following are the areas where the plant disease detection system is used

They are, **1. Agriculture**

2. Research and study

1.AGRICULTURE

Crop disease are a major threat to food security, but their rapid identification remains difficult in many parts of the world due to the lack of the necessary infrastructure. The combination of increasing global smartphone penetration and recent advanced in computer vision made possible by deep learning has paved the way for smartphone-assisted disease diagnosis.

Overall, the approach of training deep learning models on increasingly large and publicly available image datasets presents a clear path toward smartphone-assisted crop disease diagnosis on a massive global state.

2.RESEARCH AND STUDY

In future, large dataset and different types of disease can be found easily by the growing machine learning technologies. Students and biologists can research about the new diseases in plants and help them to provide remedies and treatments based on the type of disease.

DISADVANTAGES

1. More training samples **-** more speed of computing distances sensitive to irrelavant inputs so expensive testing everytime.

2. It is slower in execution speed long training time .

3. Sometimes it can predict the wrong disease which may cause difficulty to

farmers. 4. Recommending wrong fertilizers can damage the crops.

11.CONCLUSION

We have proposed ban automated system to identify and classify the disease caused in plants at an earlier stage with pest management.to detect and identification of various diseases, we use the convolutional neural network (CNN) and deep learning. The result from can be used to identify the disease with high accurate and suggest solution . High performance model is obtained by using best hyper parameters and good training data . The final model will give high accuracy for the given data. An application to detect , controls , and monitor the plantdisease helps the farmer to reduce their work as well as time.This application helps the farmer to reduce their effort, and also helps in increasing the farm of production. The proposed method helps to find the plant disease and in monitoring the several environmental conditions the status of the leaf has been identified with the help of neural network classification . Then the environment circumstances such as temperature, humidity and moisture has been monitored the environmental condition is abnormal, then the pump will automatically.This project gives the executed results on different diseases classification techniques that can be used for plant leaf disease detection a. Therefore, related diseases for these plants were taken for identification. With very less computational efforts the optimum results were obtained, which also shows the efficiency of the proposed algorithm in recognition and classification of the leaf diseases. Another advantage of using this method is that the plant diseases can be identified at an early stage or the initial stage. By using this concept, the disease identification is done for all kinds of leafs and also the user can know the affected area of leaf in percentage by identifying the disease properly the user can rectify the problem very easy.

12. FUTURE SCOPE

- This system can be enhanced in future by using the trained model in android apps to make more feasible and efficiently.

- In future, use of more advanced algorithms can beimplemented into the system to show high accuracy and less process time.

- Using the camera we canimplement the system in continuous monitoring of crops and plants for detecting the texture of plants for more early detection of plants.

- After the leaf undergoes detection, the disease is identified and check whether the leaf can be cured at certain conditions or not and fertilizersare recommended according to the leaf.

13.APPENDIX

Source Code

Image Pre-processing

```
from keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale = 1./255, shear_range = 0.2,zoom_range
= 0.2,horizontal_flip =True)

test_datagen = ImageDataGenerator(rescale = 1)
```

FRUIT

```
x_train= train_datagen.flow_from_directory('/content/drive/MyDrive/fruit-dataset/fruit
```

```
dataset/train',batch_size=32,target_size=(128,128),
                                        color_mode='rgb',class_mode='categorical')
x_test = test_datagen.flow_from_directory('/content/drive/MyDrive/fruit-dataset/fruit
dataset/test',batch_size=32,target_size=(128,128),
                                        color_mode='rgb',class_mode='categorical')
```

VEGETABLE

```
x_train= train_datagen.flow_from_directory('/content/drive/MyDrive/Veg-dataset/Veg
dataset/train_set',batch_size=32,target_size=(128,128),
                                        color_mode='rgb',class_mode='categorical')
x_test = test_datagen.flow_from_directory('/content/drive/MyDrive/Veg-dataset/Veg
dataset/test_set',batch_size=32,target_size=(128,128),
                                        color_mode='rgb',class_mode='categorical')
```