Fertilizers Recommendation System For Disease Prediction

IBM PROJECT REPORT SUBMITTED BY

Team ID : PNT2022TMID01288

Team Leader : GOKUL

Team member : AKASH DEEP

Team member : GIRIDHARAN

Team member : GOKULAN

PANIMALAR ENGINEERING COLLEGE

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

POONAMALLEE

ANNA UNIVERSITY: CHENNAI 600 025

CHAPTER-1

INTRODUCTION

1.1 OVERVIEW OF THE PROJECT

Agriculture is the maximum vital region in today's life. Most flowers are suffering froma huge type of bacterial and fungal sicknesses. Diseases in flowers located a prime constrainton the manufacturing and a prime hazard to meal security. Hence, early and correct identification of plant sicknesses is crucial to make the certain excessive amount and first-class quality. In latestyears, the number of sicknesses on flowers and the diploma of damage precipitated has expanded because of the variant in pathogen varieties, modifications in cultivation methods, and insufficient plant safety strategies. A computerized gadget is added to pick out special sicknesses on flowers via way of means of checking the signs and symptoms proven on the leaves of the plant. Deep gaining knowledge of strategies is used to pick out the sicknesses and recommend the precautions that may be taken for the one's sicknesses.

Machine learning is particularly effective in detecting and recognizing plant illnesses, and it can provide early disease sign identification. Plant disease specialists can examine the digital photos processed with digital image processing to identify blights on plants. computer vision and image processing applications Processing methods merely help farmers throughout all regions. about agriculture. In most cases, plant diseases are brought on by plant physiological functions that are aberrant. as a result, the generation of distinctive symptoms is based on distinguishing between typical physiological functions and abnormalities in the way that plants function physiologically. Typically, the pathogens that cause plant leaf diseasesare put in place on the plant's stems.

These are distinct Different factors that can predict the signs and diseases of leaves processing methods for images. These many approaches make use of various core techniques like segmentation, feature extraction, and classification, among others. Most often,segmentation is used to distinguish between healthy and diseased tissues of leaves in order to forecast and diagnose leaf diseases.

1.2 PURPOSE

The main purpose of this project is used to test the leaves and fruits of the plant's sample and identify the diseases. Then provide the recommended fertilizer for that disease. The process starts with the user has to take an image of the affected leaves and then uploading that image. It scans the leaves with the help of the CNN layer and machine learning technique. Machine learning is particularly effective in detecting and recognizing plant illnesses, and it can provide early disease sign identification. Plant disease specialists can examine the digital photos processed with digital image processing to identify blights on plants. computer vision and image processing applications Processing methods merely help farmers throughout all regions. It detects the type of that disease and finds the recommended fertilizer which should be used for that disease.

Traditional approaches depend on experts, encounters, and guides, but the bulk of them are expensive, time-consuming, and labor-intensive, and it might be challenging to precisely identify them. As a result, it seems crucial for trade and biology in agriculture that a quick and accurate method be used to identify plant infections. If the illness is not correctly detected, disease control measures could be a waste of time and money and result in further plant loss. A deep learning-based model is what our project suggests, and it will be trained using images of crop leaves that are both healthy and diseased that are taken from a dataset. The model will accomplish its objective by grouping images of leaves into harmful categories based on flaw patterns.

CHAPTER-2

LITERATURE SURVEY

2.1 EXISTING PROBLEM

Indumathi proposed a method for leaf disease detection and suggest Fertilizers to cure leaf diseases [1]. But the method involves in less number of training and test sets which results in poor accuracy. Pandi Selvi [2] proposed a simple prediction method for soil-based fertilizer recommendation systems for predicted crop diseases. This method gives less accuracy and prediction. Shiva reddy [3] proposed an IoT-based system for leaf disease detection and fertilizer recommendation which is based on Machine Learning techniques and yields less than 80 percent accuracy.

2.2 REFERENCES

[1] R Indumathi.; N Sagari.; V Thejuswini.; R Swarnareka.,"Leaf Disease Detection and Fertilizer Suggestion", IEEE International Conference on System, Computation, Automation and Networking (ICSCAN), 29-30 March 2019, DOI:10.1109/ICSCAN.2019.8878781.

[2] P. Pandi Selvi, P. Poornima, "Soil Based Fertilizer Recommendation System for Crop Disease Prediction System", International Journal of Engineering Trends and Applications (IJETA)–Volume 8 Issue 2, Mar-Apr2021.

[3] Dimitrovski, Ivica, GjorgjiMadjarov, DragiKocev, and PetreLameski, "Maestra at LifeCLEF 2014 Plant Task: Plant Identification using Visual Data", In CLEF (Working Notes), pp. 705-714, 2014.

[4] Naresh, Y. G., and H. S. Nagendraswamy, "Classification of medicinal plants: an approachusing modified LBP with symbolic representation", Neurocomputing 173, pp: 1789-1797, 2016.

[5] Sue Han, CheeSeng Chan, Paul Wilkin, and Paolo Remagnino, "Deep-plant: Plant identification with convolutional neural networks", In Image Processing (ICIP), 2015 IEEE International Conference on, pp. 452-456, IEEE, 2015.

[6] Kaur, Lakhvir, and Vijay Laxmi, "A Review on Plant Leaf Classification and Segmentation", International Journal Of Engineering And Computer Science 5, no. 8, 2016.

[7] Kadir, Abdul, Lukito Edi Nugroho, AdhiSusanto, and Paulus InsapSantosa, "Leaf classification using shape, color, and texture features", arXiv preprint arXiv:1401.4447, 2013.

[8] Lee, Sue Han, CheeSeng Chan, Simon Joseph Mayo, and Paolo Remagnino, "How deep learning extracts and learns leaf features for plant classification", Pattern Recognition 71, pp: 1-13, 2017.

2.3 PROBLEM STATEMENT DEFINITION

Agriculture is the most important sector in today's life. Most plants are affected by a wide variety of bacterial and fungal diseases. Diseases in plants placed a major constraint on the production and a major threat to food security. Hence, early and accurate identification of plant diseases is essential to ensure high quantity and best quality. In recent years, the number of diseases in plants and the degree of harm caused has increased due to the variation in pathogen varieties, changes in cultivation methods, and inadequate plant protection techniques.

An automated system is introduced to identify different diseases in 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. The disease can be found easily in the early stage by looking at changes in the color of the leaves. So, without knowing about the correct disease they use some fertilizers and it doesn't cure the disease properly. This fertilizer recommendation system helps to find the accurate disease and helps them to cure the disease and increase in the growth of plants.

CHAPTER-3

IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes.

It is a useful tool to help teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it along with his or her goals and challenges.

Fig 3.1 Empathy Map

3.2 IDEATION AND BRAINSTORMING

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem-solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Step-1: Team Gathering, Collaboration and Select the Problem Statement

Fig 3.2 Brainstorming step-1

Step-2: Brainstorm, Idea Listing and Grouping

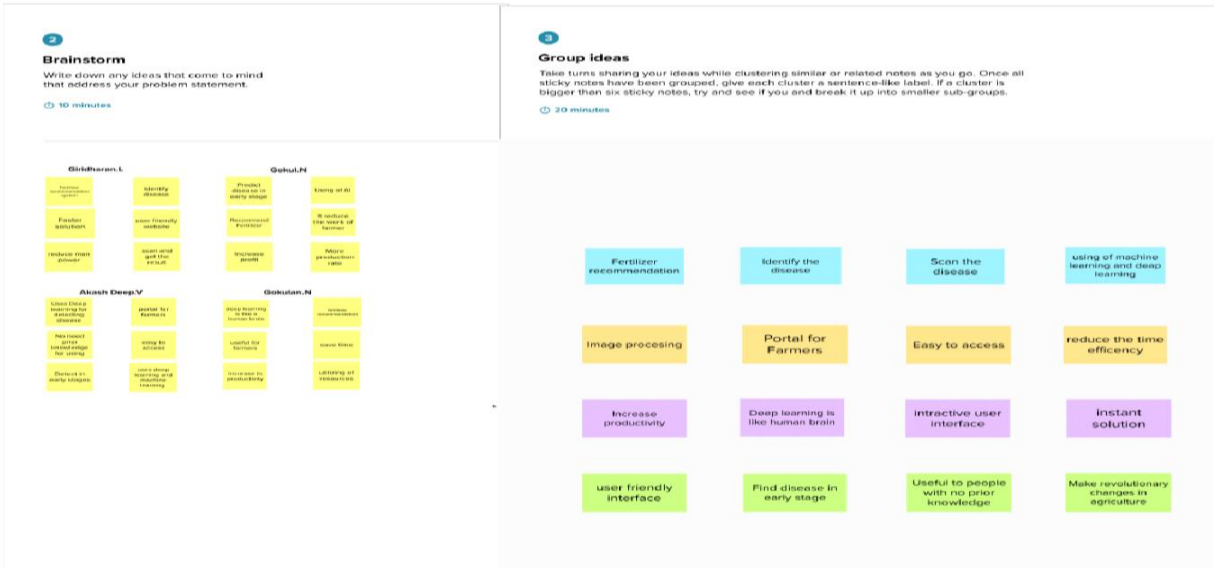


Fig 3.3 step-2

Step-3: Idea Prioritization

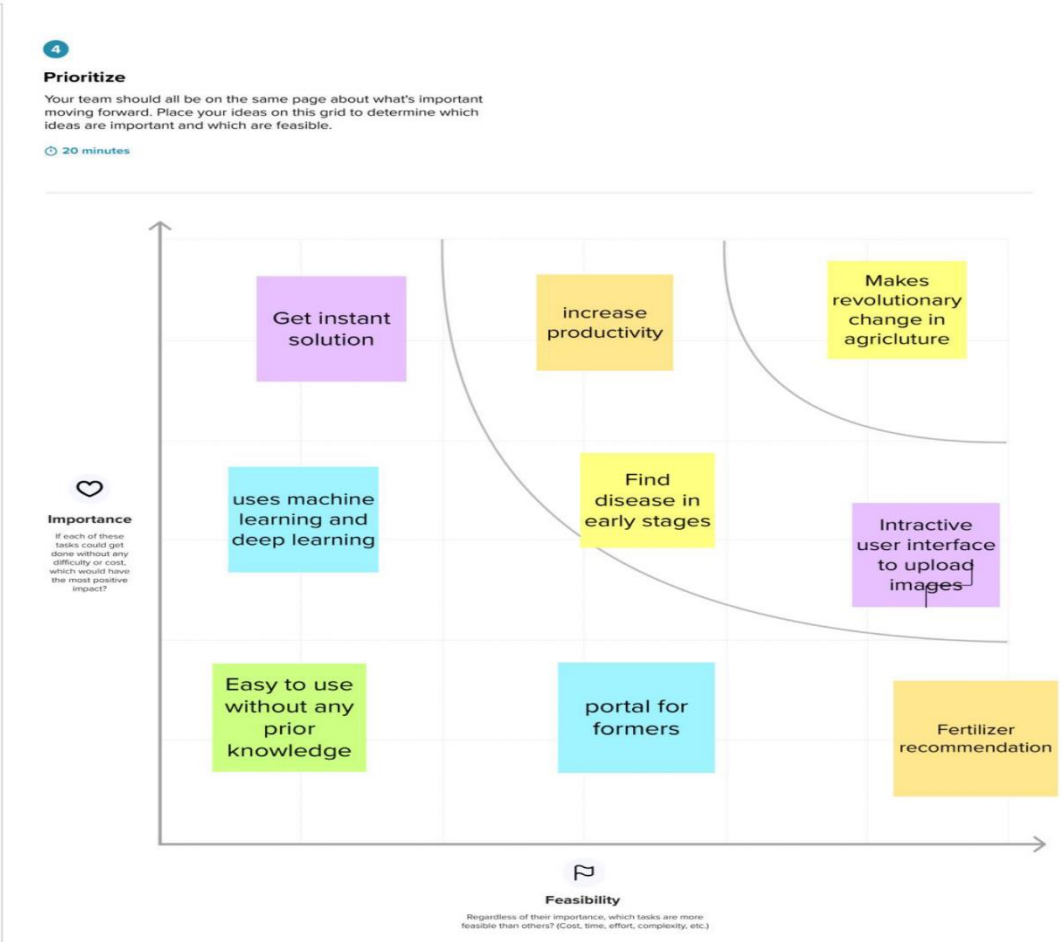


Fig 3.4 step-3

3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
•	Problem Statement (Problem to be solved)	Disease in plants reduced the quantity and quality of the plant's productivity. Identifying the disease in plants is hard to find.
•	Idea / Solution description	One of the solutions to the problem is to identify the disease in its early stage and use the correct fertilizer.
•	Novelty / Uniqueness	This application can suggest good fertilizer for the disease in the plant by recognizing the images.
•	Social Impact / Customer Satisfaction	It helps the farmer by identifying the disease in the early stage and increasing the quality and quantity of crops inefficiently way.
•	Business Model (Revenue Model)	The application is recommended to farmers on a subscription basis.
•	Scalability of the Solution	This application can be improved by introducing online purchases of crops, fertilizer easily.

3.4 PROBLEM SOLUTION FIT

Fig 3.5 solution fit

CHAPTER-4

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Image Uploading	Upload from local storage
FR-4	Image Pre-processing	Evaluating using DL Algorithm
FR-5	Displaying result	Display results got from the model
FR-6	Feedback	Give feedback through forms

4.2 NON-FUNCTIONAL REQUIREMENT

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

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	We propose a user-friendly web application system based on machine learning. So, the user can provide the input using forms on our user interface and quickly get their results. The proposed method is also found to perform better and produce a higher number of yields.
NFR-2	**Reliability**	More farmers get benefited from this system as they simply have to upload an image to get the fertilizer recommendation. Using the proposed model, crop yield production increased and gave the super ability to decide the right combination of different types of available resources. This will help farmers and agriculture experts to adopt the method for other crops.
NFR-3	**Performance**	Deep learning techniques are used to identify the diseases and suggest the precautions that can be taken for those diseases. So, it provides better performance and recommends fertilizers in a quick manner.

CHAPTER-5

PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS

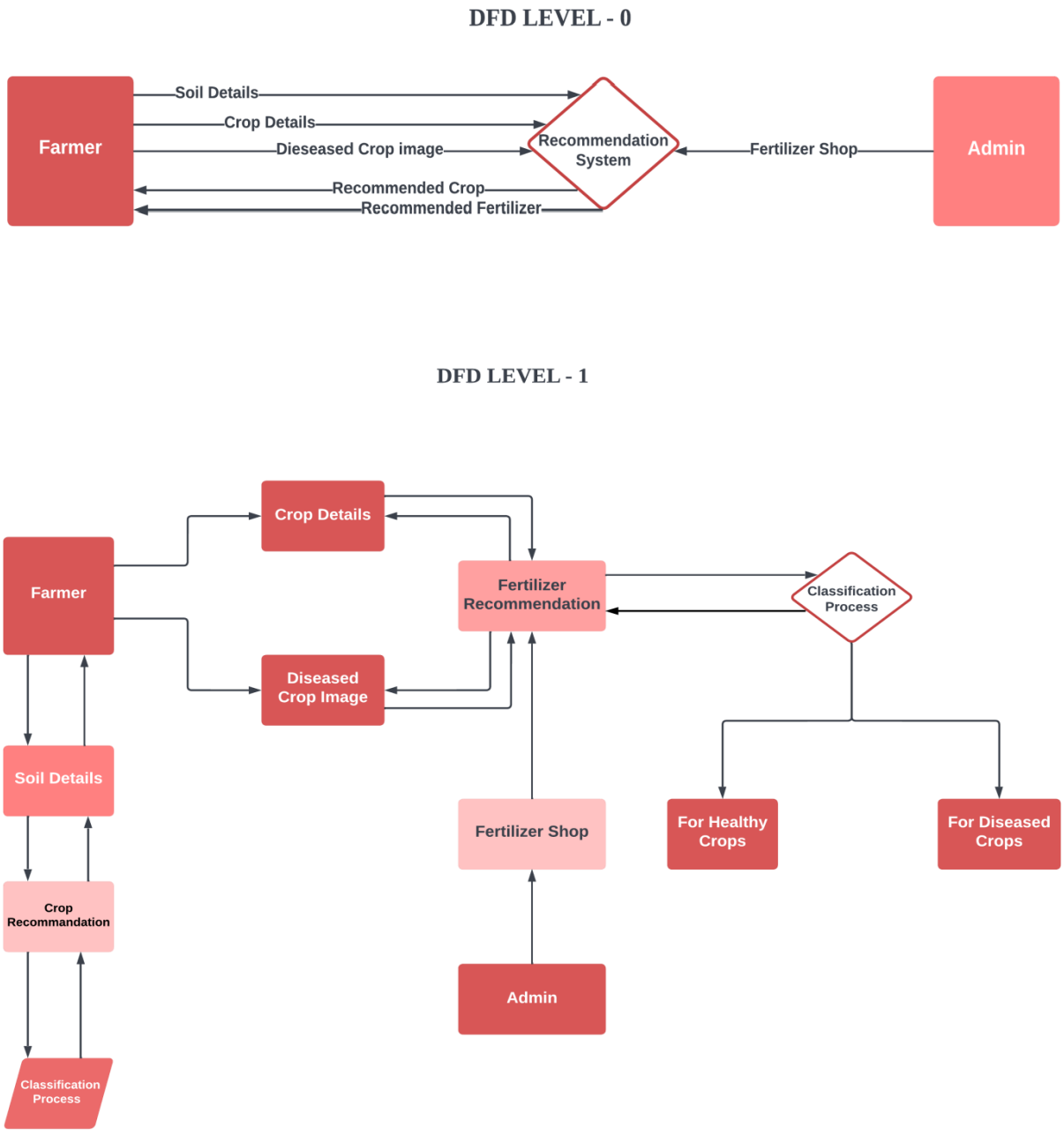


Fig 5.1 data flow diagram

5.2 SOLUTIONS AND TECHNICAL ARCHITECTURE

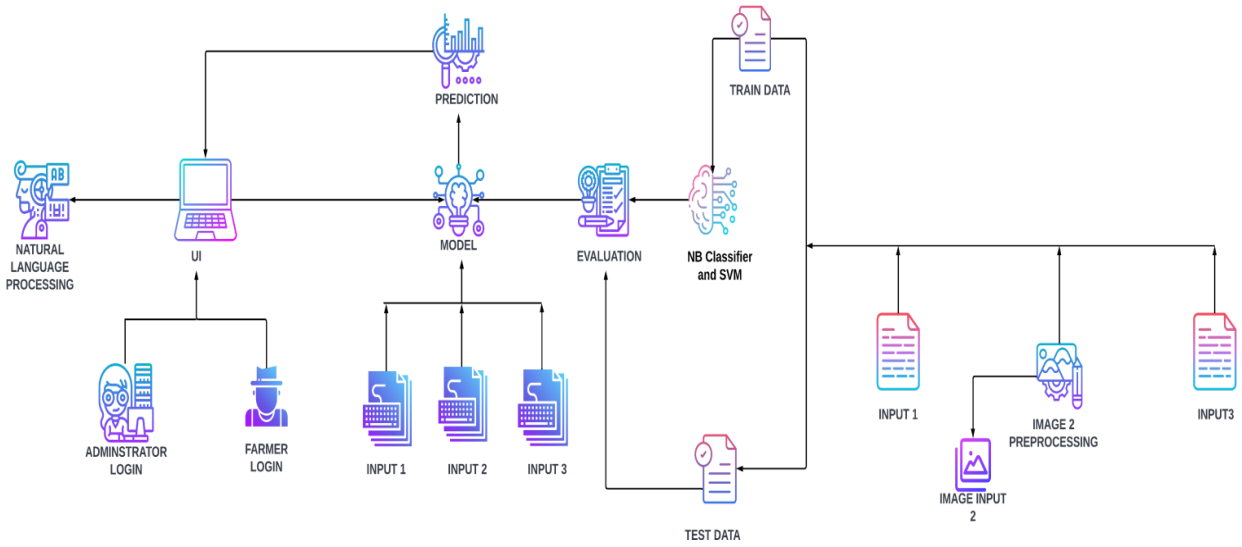


Fig 5.2 technical architecture

Table-1: Components & Technologies:

S. No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript
2.	Application Logic-1	A page to upload images as input	Python
3.	Application Logic-2	To use the Machine Learning model and predicting the result	Python
4.	Database	Structured data-images	MySQL
5.	Cloud Database	Database that typically runs on a cloud computing platform and access to the database is provided as-a- service	IBM Cloud Databases for MySQL
6.	File Storage	To store data in a hierarchical structure	Local File system
7.	Machine Learning Model	We use a Support Vector Machine Algorithm that is used widely in Classification and Regressionproblems.	Random Forest ,XG Boost

Table 2: Application Characteristics:

S. No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Flask micro web framework	Written in Python. It is classified as a micro frame work because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions
2.	Security Implementations	With all aspects of the job, including detecting malicious attacks, analysing the network endpointprotection and vulnerability assessment, Sign in encryption	IBM Cloud App ID Services
3.	Scalable Architecture	It can expand according to plant diseases andfertilizer recommendation system	-
4.	Availability	Available for all data size	-
5.	Performance	Can extend the storage according to our needs	Python, AngularJS

5.3 USER STORIES

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
	Login	USN-2	As a user, I can log into the application by entering email & password	I can log in using my Email ID account or user credentials	High	Sprint-1
	Dashboard	USN-3	As a user, I can view the page of the application where I can upload my images and the fertilizer should be recommended.	I can access my account/ dashboard	High	Sprint-2
Customer (Web user)	Registration	USN-4	As a user, I can log in to web Dashboard just Like website dashboard.	I can register using my username and password.	High	Sprint-3
	Login	USN-5	As a user, I can log in to my web dashboard	I can log in using my user credentials.	High	Sprint-3

			with the login credentials			
	Dashboard	USN-6	As a user, I can view the web application where I can upload myimages and the fertilizer should be recommended.	I can access my account/ dashboard.	High	Sprint-4
		USN-7	As the user, the fertilizer recommendedshould be of higher accuracy	I can access my account/ dashboard.	High	Sprint-4
Administrator	Login	USN-8	As an admin, Ican log in to the website, using my login credentials.	I can log into the website using my login credentials.	High	Sprint-5
	Dashboard	USN-9	As an admin, Ican view the dashboard of the application.	I can accessmy dashboard	High	Sprint-5

CHAPTER-6

PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can sign up and register respective sites to access the required detailsand data. And import the required libraries forthe processes.	2	High	Gokul.N Akashdeep.V Giridharan.L Gokulan.N
Sprint-2	Login	USN-2	As a user, I will access the page and test andtrain the CNN model to predict or detect the plant disease.	2	High	Gokul.N Akashdeep.V Giridharan.L Gokulan.N
Sprint-3	Customer Service	USN-3	As a customer care executive , I am availableto the customers . so if the customers have any issues or in need of any assistance they will get help and solve them.	1	Medium	Gokul.N Akashdeep.V Giridharan.L Gokulan.N
Sprint-4	Dashboard	USN-4	As a user, I will have the access to knowabout the activities in the plant.	2	High	Gokul.N Akashdeep.V Giridharan.L Gokulan.N

6.2 SPRINT DELIVERY SCHEDULE

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	04 Nov 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022		06 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022		09 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022		12 Nov 2022

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's averagevelocity (AV) per iteration unit (story points per day)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

AV:

Sprint 1 = 20/6= 3.33,

Sprint 2 = 20/6= 3.33,

Sprint 3 = 20/6= 3.33,

Sprint 4 = 20/6= 3.33.

A burndown chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as scrum. However, burn-down charts can be applied to any project containing measurable progress over time.

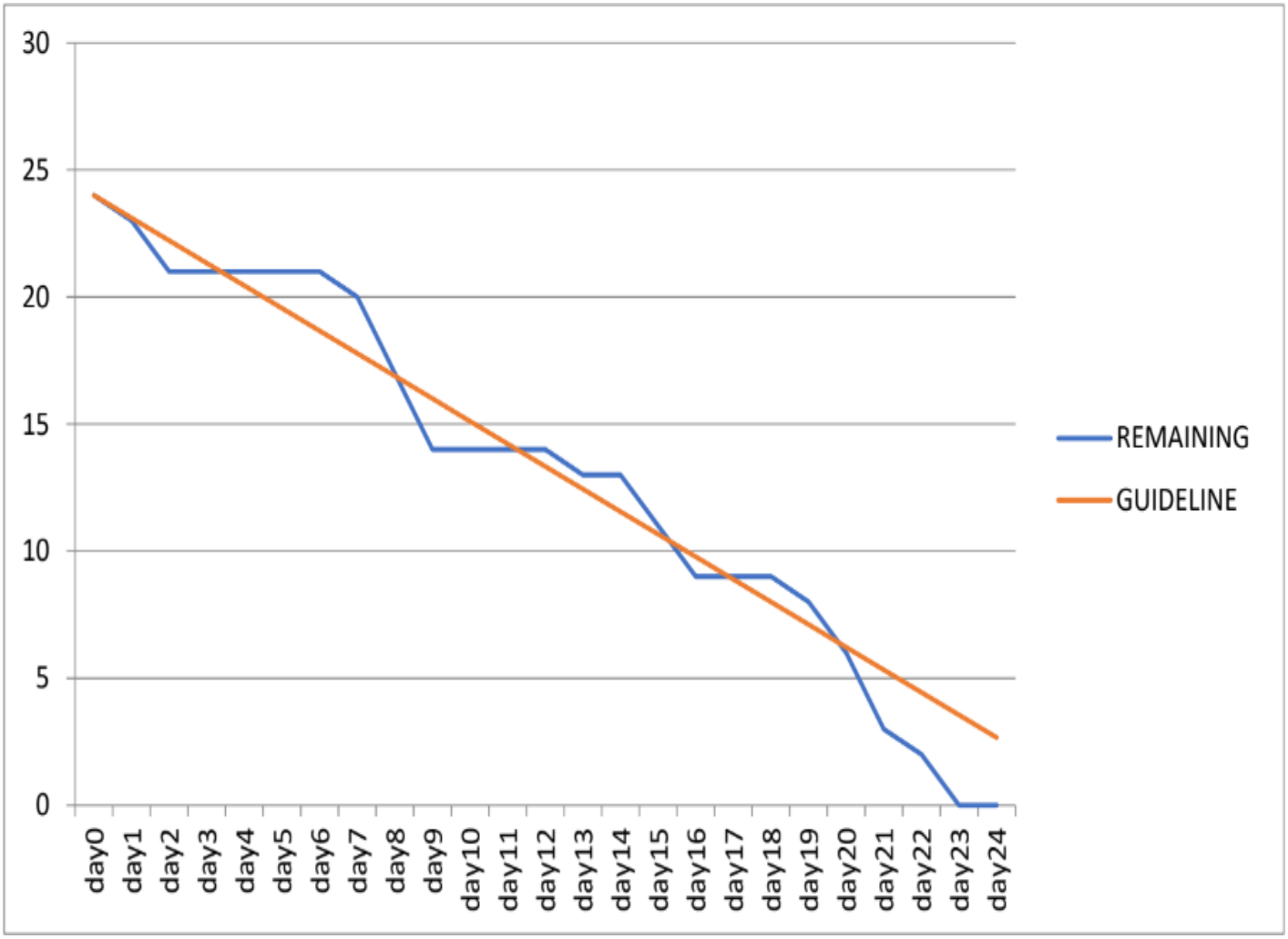


Fig 6.1 Burn-down chart

CHAPTER-7

CODING & SOLUTIONING

7.1 FEATURE 1

7.1.1 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 fruit diseases like corn, peach, and apple.

7.1.2 IMAGE PROCESSING

Before training the model, you have to pre-process the images and then feed them onto themodel for training. We make use of the 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 atraining dataset by creating modified versions of images in the dataset. The Keras deep learning neural network library provides the capability to fit models using imagedata augmentation via the ImageDataGenerator class.

7.1.3 MODEL BUILDING FOR 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

7.1.4 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
```

7.1.5 ADD CNN AND CONVOLUTION LAYER

We will be adding three layers for CNN

- Convolution layer
- Pooling layer

Flattening 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 layerapplies 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),activation='relu'))
```

7.1.6 TRAIN AND SAVE THE MODEL

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

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

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

7.1.7 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 [==============================] - 4s/step - loss: 0.6571 - accuracy: 0.7882 - val_loss: 60.1567 - val_accuracy: 0.7824

Epoch 3/20

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

Epoch 4/20

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

Epoch 5/20

89/89 [==============================] - 5s 836ms/step - loss: 0.2583 - accuracy: 0.9129 - val_loss: 541.0003 - val_accuracy: 0.4641

Epoch 6/20

89/89 [==============================] - 0s 673ms/step - loss: 0.2481 - accuracy: 0.9112 - val_loss: 663.6074 - val_accuracy: 0.4630

Epoch 7/20

89/89 [==============================] - 4s 599ms/step - loss: 0.2167 - accuracy: 0.9252 - val_loss: 504.1471 - val_accuracy: 0.4850

Epoch 8/20

89/89 [==============================] - 2s 584ms/step - loss: 0.2076 - accuracy: 0.9274 - val_loss: 554.8959 - val_accuracy: 0.4618

Epoch 9/20

89/89 [==============================] - 1s 574ms/step - loss: 0.2308 - accuracy: 0.9200 - val_loss: 591.8171 - val_accuracy: 0.4618

Epoch 10/20

89/89 [==============================] - 0s 564ms/step - loss: 0.1834 - accuracy: 0.9402 - val_loss: 927.3312 - val_accuracy: 0.4028

Epoch 11/20

89/89 [==============================] - 0s 558ms/step - loss: 0.1923 - accuracy:

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 - accurac

[==============================] - ep : - y:

0.9402 - val_loss: 927.3312 - val_accuracy:

0.4028

Epoch 11/20

89/89 50s 558ms/st - loss 0.1923 accurac

[==============================] - ep : - y:

7.2 FEATURE 2

7.2.2 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 login to browse the images to detect the disease.

In this section, you have to build

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

7.2.3 BUILD PYTHON CODE

After the model is built, we will be integrating it into a web application so that normal users canalso 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_functionimport os

import numpy as np

import cv2

Keras

from tensorflow.keras.models import load_model

from tensorflow.keras.preprocessing.image import img_to_array

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)___healthy", "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 toPython 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 predictionthe 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, IPaddress, 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.filename))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 integrateddevelopment webserver –

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

Output:

* Serving Flask app 'app'
* Debug mode: on
* Running on http://127.0.0.1:500

7.2.4 BUILD HTML PAGES

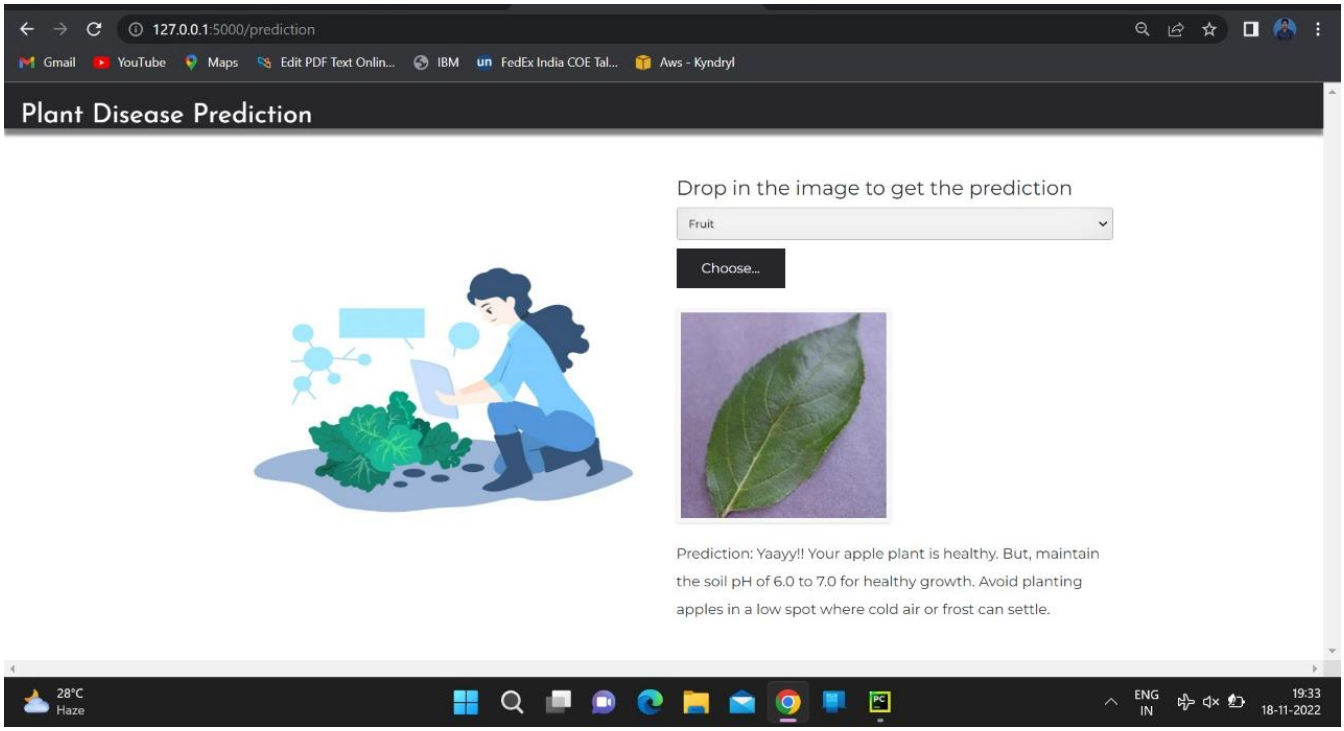
```
{% extends 'layout.html' %}
{% block body %}
<!-- banner -->
<section class="banner_w3lspvt" id="home">
    <div class="csslider infinity" id="slider1">
    <div class="banner-top1">
    <div class="overlay">
    <div class="container">
    <div class="w3layouts-banner-info text-center">
```

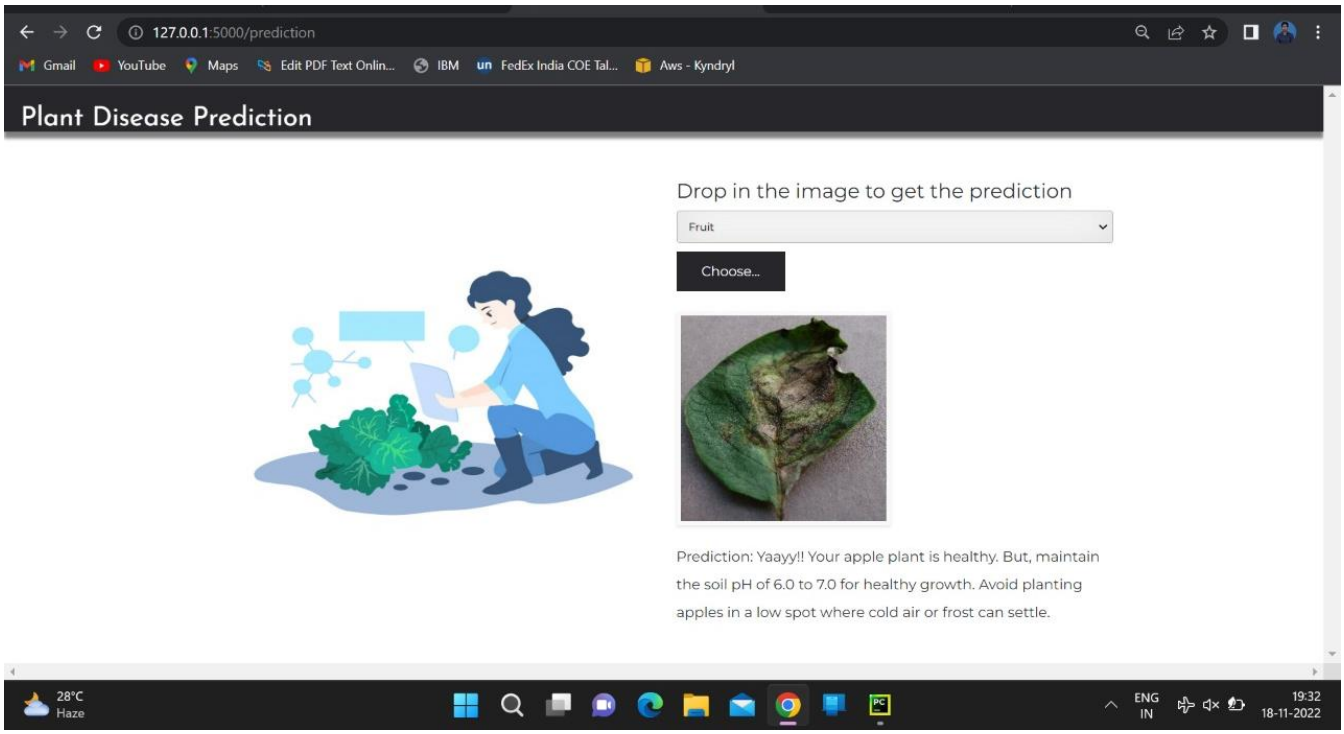
```
        <h3 class="text-wh">MyCrop-Plant Disease Prediction</h3>
  <h4 class="text-wh mx-auto my-4"><b>Get informed decisions about your
      farmingstrategy.<br>In Your Own Language.</b></h4>
  <h4 class="text-wh mx-auto my-4"><strong> Here are some questions
      we'llanswer</strong></h4>
   <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=""
      /></div>
```

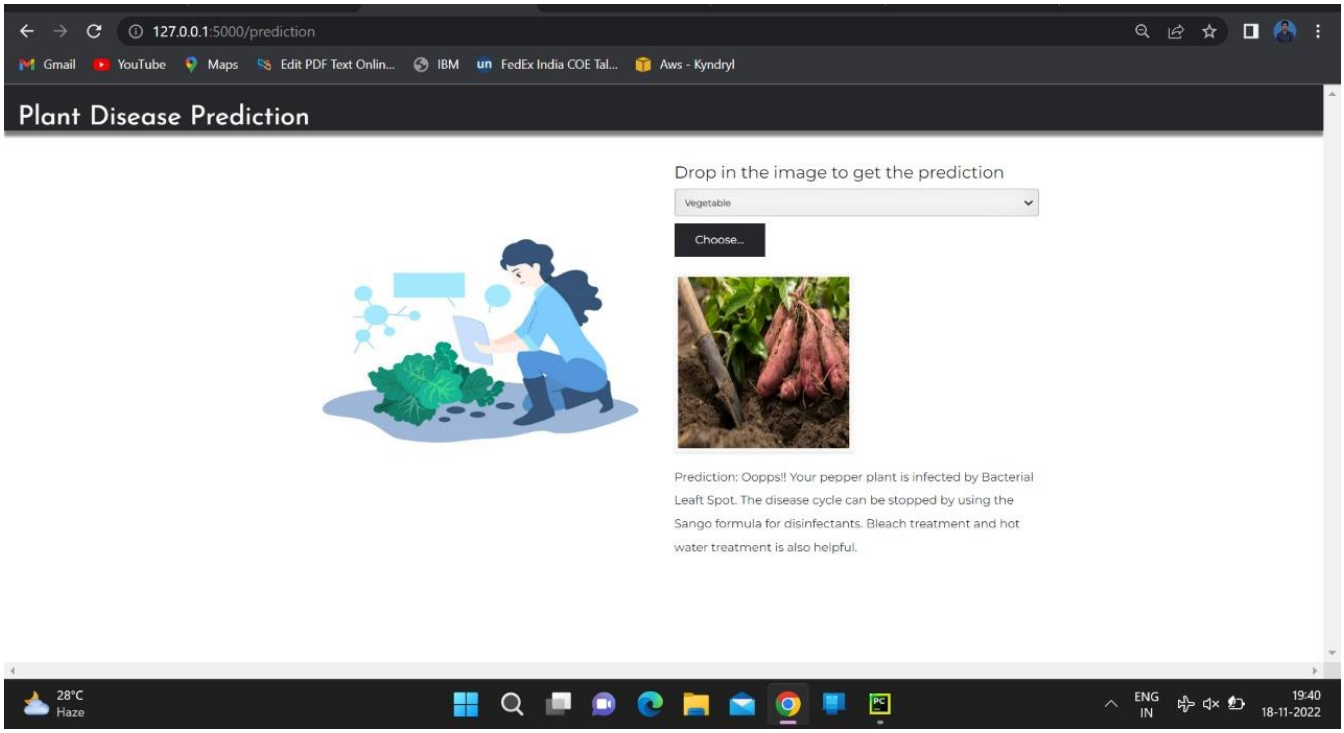
```
    <div class="col-lg-6 core-right">
  <h3 class="mt-4">Improving Agriculture, Improving Lives, Cultivating Crops To Make
      FarmersIncrease Profit.</h3>
  <p class="mt-3">We use state-of-the-art machine learning and deep learning
      technologies to help you to guide through the entire farming process. Make informed
      decisions to understand thedemographics 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 leaf</p>
</div></a>
```

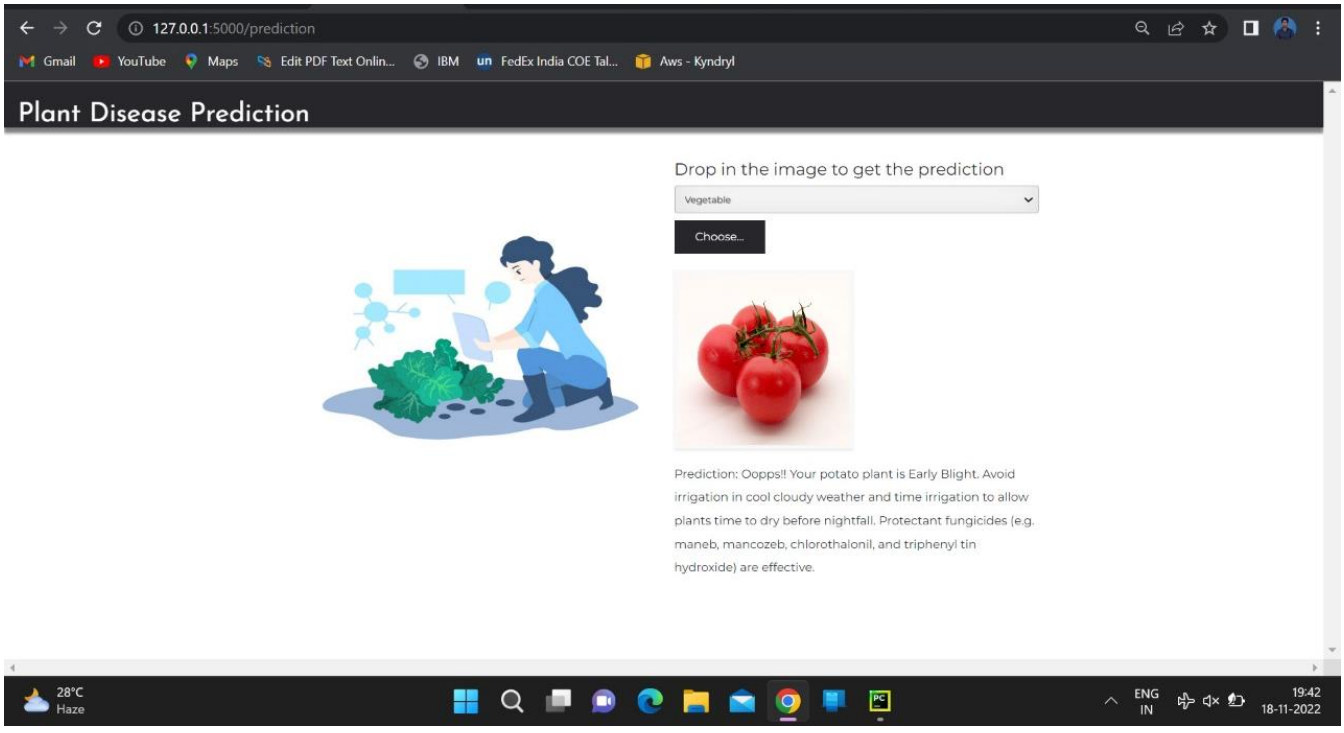
```
                    <br><br><br>
  </div><div class="col-lg-4 col-md-6 blog-middle mb-lg-0 mb-md-5 pb-md-5 pb-5">
<a href="{{ url_for('home') }}">
  <img src="{{ url_for('static', filename='images/s6.jpg') }}" class="img-fluid" alt="" />
  <div class="blog-info">
  <h4>Fertilizer Recommendation and Prevention</h4>
  <p class="mt-1">Recommendation about the prevention step to the user to prevent the
      disease infuture.</p>

  <div class="col-lg-4 col-md-6 blog-right mb-lg-0 mb-sm-5 pb-lg-5 pb-md-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 %}
```


127.0.0.1:5000/prediction
Gmail
YouTube
Maps
Edit PDF Text Onlin...
IBM
FedEx India COE Tal...
Aws - Kyndryl
Plant Disease Prediction
Drop in the image to get the prediction
Fruit
Choose...
Prediction: Yaayy!! Your apple plant is healthy. But, maintain the soil pH of 6.0 to 7.0 for healthy growth. Avoid planting apples in a low spot where cold air or frost can settle.
28°C
Haze
ENG
IN
19:33
18-11-2022


127.0.0.1:5000/prediction
Gmail
YouTube
Maps
Edit PDF Text Onlin...
IBM
FedEx India COE Tal...
Aws - Kyndryl
Plant Disease Prediction
Drop in the image to get the prediction
Fruit
Choose...
Prediction: Yaayy!! Your apple plant is healthy. But, maintain the soil pH of 6.0 to 7.0 for healthy growth. Avoid planting apples in a low spot where cold air or frost can settle.
28°C
Haze
ENG
IN
19:32
18-11-2022


127.0.0.1:5000/prediction
Gmail
YouTube
Maps
Edit PDF Text Onlin...
IBM
FedEx India COE Tal...
Aws - Kyndryl
Plant Disease Prediction
Drop in the image to get the prediction
Vegetable
Choose...
Prediction: Oopps!! Your pepper plant is infected by Bacterial Leaft Spot. The disease cycle can be stopped by using the Sango formula for disinfectants. Bleach treatment and hot water treatment is also helpful.
28°C
Haze
ENG
IN
19:40
18-11-2022


127.0.0.1:5000/prediction
Gmail
YouTube
Maps
Edit PDF Text Onlin...
IBM
FedEx India COE Tal...
Aws - Kyndryl
Plant Disease Prediction
Drop in the image to get the prediction
Vegetable
Choose...
Prediction: Oopps!! Your potato plant is Early Blight. Avoid irrigation in cool cloudy weather and time irrigation to allow plants time to dry before nightfall. Protectant fungicides (e.g. maneb, mancozeb, chlorothalonil, and triphenyl tin hydroxide) are effective.
28°C
Haze
ENG
IN
19:42
18-11-2022

CHAPTER-8

TESTING

8.1 TEST CASE

TEST SCENARIO	STEPS TO EXECUTE	TEST DATA	EXPECTED RESULT	ACTUAL RESULT	STATUS
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 thelink provided to visit the home page	http://127.0.0.1:5000	Home page isdisplayed	Home page is displayed	pass
Verify theuser can see the homepage and see the diseases	1.Go to the homepage 2.Click on to thediseases 3. A predict button will bedisplayed 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 theuser 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 thediseased leaves has tobe displayed	Images of the diseased leaves has to displayed	pass

predict button	3.Select a leaf image that has to be predicted 4.After the leaf is predicted, theleaf has to determine the diseases.				
Verify the leafdisease i spredicted 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 thefertilizer.	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

CHAPTER-9

RESULT

9.1 PERFORMANCE METRICS

VEGETABLE

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_1 (Conv2D)           (None, 126, 126, 32)      896

 max_pooling2d_1 (MaxPooling  (None, 63, 63, 32)       0
 2D)

 flatten_1 (Flatten)         (None, 127008)            0

 dense_7 (Dense)             (None, 300)               38102700

 dense_8 (Dense)             (None, 150)               45150

 dense_9 (Dense)             (None, 75)                11325

 dense_10 (Dense)            (None, 9)                 684

=================================================================
Total params: 38,160,755
Trainable params: 38,160,755
Non-trainable params: 0
_________________________________________________________________
```

FRUIT

```
Model: "sequential"
_________________________________________________________________
 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
```

PARAMETER ACCURACY

Training Accuracy

Validation Accuracy

VEGETABLE

```
model.fit(x_train,epochs=10,steps_per_epoch=89,validation_data = x_test, validation_steps = 27)


model.save("veg.h5")
```

```
Epoch 1/10
89/89 [==============================] - 718s 8s/step - loss: 1.4285 - accuracy: 0.5103 - val_loss: 398.4555 - val_accuracy: 0.3611
Epoch 2/10
89/89 [==============================] - 512s 6s/step - loss: 0.9607 - accuracy: 0.6552 - val_loss: 883.0972 - val_accuracy: 0.2697
Epoch 3/10
89/89 [==============================] - 361s 4s/step - loss: 0.7985 - accuracy: 0.7229 - val_loss: 664.0219 - val_accuracy: 0.3275
Epoch 4/10
89/89 [==============================] - 290s 3s/step - loss: 0.6901 - accuracy: 0.7598 - val_loss: 870.4464 - val_accuracy: 0.2859
Epoch 5/10
89/89 [==============================] - 208s 2s/step - loss: 0.6114 - accuracy: 0.7802 - val_loss: 709.7632 - val_accuracy: 0.3542
Epoch 6/10
89/89 [==============================] - 183s 2s/step - loss: 0.5603 - accuracy: 0.7978 - val_loss: 842.9805 - val_accuracy: 0.2384
Epoch 7/10
89/89 [==============================] - 148s 2s/step - loss: 0.5167 - accuracy: 0.8195 - val_loss: 1794.7992 - val_accuracy: 0.1296
Epoch 8/10
89/89 [==============================] - 118s 1s/step - loss: 0.4628 - accuracy: 0.8385 - val_loss: 1593.1969 - val_accuracy: 0.1516
Epoch 9/10
89/89 [==============================] - 103s 1s/step - loss: 0.4795 - accuracy: 0.8304 - val_loss: 1793.0253 - val_accuracy: 0.1551
Epoch 10/10
89/89 [==============================] - 94s 1s/step - loss: 0.3958 - accuracy: 0.8575 - val_loss: 1651.8546 - val_accuracy: 0.1505
```

FRUIT

```
model.fit(x_train,epochs=10,steps_per_epoch=89,validation_data = x_test, validation_steps = 27)


model.save("fruit.h5")
```

```
Epoch 1/10
89/89 [==============================] - 945s 11s/step - loss: 1.1761 - accuracy: 0.6246 - val_loss: 66.9958 - val_accuracy: 0.7940
Epoch 2/10
89/89 [==============================] - 477s 5s/step - loss: 0.5927 - accuracy: 0.8090 - val_loss: 129.4430 - val_accuracy: 0.6516
Epoch 3/10
89/89 [==============================] - 234s 3s/step - loss: 0.4787 - accuracy: 0.8441 - val_loss: 227.8628 - val_accuracy: 0.5243
Epoch 4/10
89/89 [==============================] - 122s 1s/step - loss: 0.3456 - accuracy: 0.8835 - val_loss: 233.2232 - val_accuracy: 0.5359
Epoch 5/10
89/89 [==============================] - 85s 959ms/step - loss: 0.2847 - accuracy: 0.9040 - val_loss: 633.7368 - val_accuracy: 0.3704
Epoch 6/10
89/89 [==============================] - 68s 767ms/step - loss: 0.2261 - accuracy: 0.9235 - val_loss: 681.6103 - val_accuracy: 0.3993
Epoch 7/10
89/89 [==============================] - 59s 663ms/step - loss: 0.2459 - accuracy: 0.9125 - val_loss: 233.5868 - val_accuracy: 0.6343
Epoch 8/10
89/89 [==============================] - 52s 587ms/step - loss: 0.2116 - accuracy: 0.9245 - val_loss: 600.8589 - val_accuracy: 0.4167
Epoch 9/10
89/89 [==============================] - 51s 572ms/step - loss: 0.1742 - accuracy: 0.9431 - val_loss: 729.3225 - val_accuracy: 0.4167
Epoch 10/10
89/89 [==============================] - 52s 587ms/step - loss: 0.1638 - accuracy: 0.9437 - val_loss: 778.6277 - val_accuracy: 0.3681
```

CHAPTER-10

ADVANTAGES & DISADVANTAGES

10.1 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.
- It will suggest the recommended fertilizer for that disease quickly with in a minute of time.

10.2 DISADVANTAGES

1. More training samples - more speed of computing distances sensitive irrelevant inputsso expensive test every time.
2. It is slower in execution speed and long training time.
3. Sometimes it can predict the wrong disease which may cause difficulty for farmers.
4. Recommending the wrong fertilizers can damage crops.
5. It requires more samples to prepare the application and if any wrong updates that make crop damage.
6. Previously yield is predicted on the bases of the farmers prior experience but now weather conditions may change drastically so they cannot guess the yield.

CHAPTER-11

CONCLUSION

We have proposed an 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 usethe convolutional neural network (CNN) and deep learning. The result from can be used to identify the disease with a highly accurate and suggested solution. A high-performance model is obtained by using the best hyperparameters and good training data. The final model will give high accuracy for the given data. An application to detect, control, and monitor plant disease 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 environmental circumstances such as temperature, humidity, and moisture have been monitored the environmental condition is abnormal, then the pump will automatically. This project gives the executed results on different disease 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 effort the optimum results were obtained, which also shows the efficiency of the proposed algorithm in the recognition and classification of the leaf diseases. Another advantage of using this method is that plant diseases can be identified at an early stage or the initial stage. By using this concept, disease identification is done for all kinds of leaves and also the user can know the affected area of the leaf in percentage by identifying the disease properly the user can rectify the problem very easily.

CHAPTER-12
FUTURE SCOPE

- This system can be enhanced in the future by using the trained model in android apps tomake it more feasible and efficient.
- In the future, the use of more advanced algorithms can be implemented into the system to showhigh accuracy and less process time.
- Using the camera we can implement 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 checked whether the leaf can be cured under certain conditions or not, and fertilizers are recommended according to the leaf.

CHAPTER-13

APPENDIX

13.1 SOURCE CODE

Html Code:

```
<!DOCTYPE html>
<html >
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title> Plant Disease Prediction</title>
  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet' type='text/css'>
<link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">
<link href='https://fonts.googleapis.com/css?family=Merriweather' rel='stylesheet'>
<link href='https://fonts.googleapis.com/css?family=Josefin Sans' rel='stylesheet'>
<link href='https://fonts.googleapis.com/css?family=Montserrat' rel='stylesheet'>
<style>
.header {
                        top:0;
                        margin:0px;
                        left: 0px;
                        right: 0px;
```

```
                position: fixed;
                background-color: #28272c;
                color: white;
                box-shadow: 0px 8px 4px grey;
                overflow: hidden;
                padding-left:20px;
                font-family: 'Josefin Sans';
                font-size: 2vw;
                width: 100%;
                height:8%;
                text-align: center;
            }
            .topnav {
  overflow: hidden;
  background-color: #333;
}
.topnav-right a {
  float: left;
  color: #f2f2f2;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 18px;
}

.topnav-right a:hover {
  background-color: #ddd;
```

```
  color: black;
}
.topnav-right a.active {
  background-color: #565961;
  color: white;
}
.topnav-right {
  float: right;
  padding-right:100px;
}
body {
  background-color:#ffffff;
  background-repeat: no-repeat;
  background-size:cover;
  background-position: 0px 0px;
  }
  .button {
  background-color: #28272c;
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  border-radius: 12px;
}
```

```
.button:hover {
  box-shadow: 0 12px 16px 0 rgba(0,0,0,0.24), 0 17px 50px 0 rgba(0,0,0,0.19);
}
form {border: 3px solid #f1f1f1; margin-left:400px;margin-right:400px;}
input[type=text], input[type=password] {
  width: 100%;
  padding: 12px 20px;
  display: inline-block;
  margin-bottom:18px;
  border: 1px solid #ccc;
  box-sizing: border-box;
}
button {
  background-color: #28272c;
  color: white;
  padding: 14px 20px;
  margin-bottom:8px;
  border: none;
  cursor: pointer;
  width: 15%;
  border-radius:4px;
}
button:hover {
  opacity: 0.8;
}
.cancelbtn {
  width: auto;
```

```
  padding: 10px 18px;
  background-color: #f44336;
}
.imgcontainer {
  text-align: center;
  margin: 24px 0 12px 0;
}
img.avatar {
  width: 30%;
  border-radius: 50%;
}
.container {
  padding: 16px;
}
span.psw {
  float: right;
  padding-top: 16px;
}
@media screen and (max-width: 300px) {
  span.psw {
     display: block;
     float: none;
  }
  .cancelbtn {
     width: 100%;
  }
}
```

```
.home{
        margin:80px;
  width: 84%;
  height: 500px;
  padding-top:10px;
  padding-left: 30px;
}
.login{
        margin:80px;
        box-sizing: content-box;
  width: 84%;
  height: 420px;
  padding: 30px;
  border: 10px solid blue;
}
.left,.right{
 box-sizing: content-box;
 height: 400px;
 margin:20px;
 border: 10px solid blue;
}
.mySlides {display: none;}
img {vertical-align: middle;}
.slideshow-container {
  max-width: 1000px;
  position: relative;
```

```
  margin: auto;
}
.text {
  color: #f2f2f2;
  font-size: 15px;
  padding: 8px 12px;
  position: absolute;
  bottom: 8px;
  width: 100%;
  text-align: center;
}
.dot {
  height: 15px;
  width: 15px;
  margin: 0 2px;
  background-color: #bbb;
  border-radius: 50%;
  display: inline-block;
  transition: background-color 0.6s ease;
}
.active {
  background-color: #717171;
}
.fade {
  -webkit-animation-name: fade;
  -webkit-animation-duration: 1.5s;
  animation-name: fade;
```

```
  animation-duration: 1.5s;
}
@-webkit-keyframes fade {
  from {opacity: .4}
  to {opacity: 1}
}
@keyframes fade {
  from {opacity: .4}
  to {opacity: 1}
}
@media only screen and (max-width: 300px) {
  .text {font-size: 11px}
}
</style>
</head>
<body style="font-family:'Times New Roman', Times, serif;background-color:#C2C5A8;">
<div class="header">
 <div style="width:50%;float:left;font-size:2vw;text-align:left;color:white; padding-top:1%">Plant Disease Prediction</div>
  <div class="topnav-right"style="padding-top:0.5%;">
    <a class="active" href="{{ url_for('home')}}">Home</a>
    <a href="{{ url_for('prediction')}}">Predict</a>
  </div>
</div>
<div style="background-color:#ffffff;">
<div style="width:60%;float:left;">
```

```
<div style="font-size:50px;font-family:Montserrat;padding-left:20px;text-align:center;padding-top:10%;">

<b>Detect if your plant<br> is infected!!</b></div><br>

<div style="font-size:20px;font-family:Montserrat;padding-left:70px;padding-right:30px;text-align:justify;">Agriculture is one of the major sectors worls wide. Over the years it has developed and the use of new technologies and equipment replaced almost all the traditional methods of farming. The plant diseases effect the production. Identification of diseases and taking necessary precautions is all done through naked eye, which requires labour and laboratries. This application helps farmers in detecting the diseases by observing the spots on the leaves, which inturn saves effort and labor costs.</div><br><br>

</div>

</div>

<div style="width:40%;float:right;"><br><br>

<img src="{{url_for('static',filename='images/12456.png')}}" style="max-height:100%;max-width:100%;">

</div>

</div>

<div class="home">

<br>

</div>

<script>

var slideIndex = 0;

showSlides();

function showSlides() {

  var i;

  var slides = document.getElementsByClassName("mySlides");

  var dots = document.getElementsByClassName("dot");

  for (i = 0; i < slides.length; i++) {

    slides[i].style.display = "none";
```

```
  }
  slideIndex++;
  if (slideIndex > slides.length) {slideIndex = 1}
  for (i = 0; i < dots.length; i++) {
    dots[i].className = dots[i].className.replace(" active", "");
  }
  slides[slideIndex-1].style.display = "block";
  dots[slideIndex-1].className += " active";
  setTimeout(showSlides, 2000); // Change image every 2 seconds
}
</script>
</body>
</html>
```

SOURCE CODE LINK:

https://drive.google.com/drive/folders/1l2LgV--jBIyuyeZH4amimFZpK7SyML0n

13.2 GITHUB AND PROJECT DEMO LINK

GITHUB LINK: https://github.com/IBM-EPBL/IBM-Project-5212-1658751622

PROJECT DEMO LINK:
https://drive.google.com/file/d/1PoqUUViwdEEHxv4pG7--E82MVRsT-uZS/view?usp=sharing