

Fertilizers Recommendation System for Disease Prediction

IBM

PROJECT REPORT

Submitted by

TEAM ID :PNT2022TMID46973

ISWARYA M (822119104015)

ARITHRA P(822119104006)

ARTHI S(822119104007)

SNEKA P (822119104039)

In partial fulfilment for the award of the degree Of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING



**UNIVERSITY COLLEGE OF ENGINEERING,
PATTUKOTTAI**

INDEX

1. INTRODUCTION

1.1 Project Overview

1.2 Purpose

2. LITERATURE SURVEY

2.1 Existing problem

2.2 References

2.3 Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

4.2 Non-Functional requirements

5. PROJECT DESIGN

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

6.3 Reports from JIRA

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1

7.2 Feature 2

7.3 Database Schema (if Applicable)

8. TESTING

8.1 Test Cases

8.2 User Acceptance Testing

9. RESULTS

9.1 Performance Metrics

10. ADVANTAGES & DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX

Source Code

GitHub & Project Demo Link

1.INTRODUCTION

1.1 Overview In this project, two datasets name fruit dataset and vegetable dataset are collected. The collected datasets are trained and tested with deep learning neural network named Convolutional Neural Networks (CNN). First, the fruit dataset is trained and then tested with CNN. It has 6 classes and all the classes are trained and tested. Second, the vegetable dataset is trained and tested. The software used for training and testing of datasets is Python. All the Python codes are first written in Jupyter notebook supplied along with Anaconda Python and then the codes are tested in IBM cloud. Finally, a webbased framework is designed with help Flask a Python library. There are 2 html files are created in templates folder along with their associated files in static folder. The Python program 'app.py' used to interface with these two webpages is written in Spyder-Anaconda python and tested.

1.2 Purpose This project is used to test the fruits and vegetables samples and identify the different diseases. Also, this project recommends fertilizers for predicted diseases.

2.LITERATURE SURVEY

2.1 Existing problem should proposed a method for leaf disease detection and suggest fertilizers to cure leaf diseases. But the method involves less number of train and test sets which results in poor accuracy. It proposed a simple prediction method for soilbased fertilizer recommendation system for predicted crop diseases. This method gives less accuracy and prediction. IoT based system for leaf disease detection and fertilizer recommendation which is based on Machine Learning techniques yields less 80 percentage accuracies.

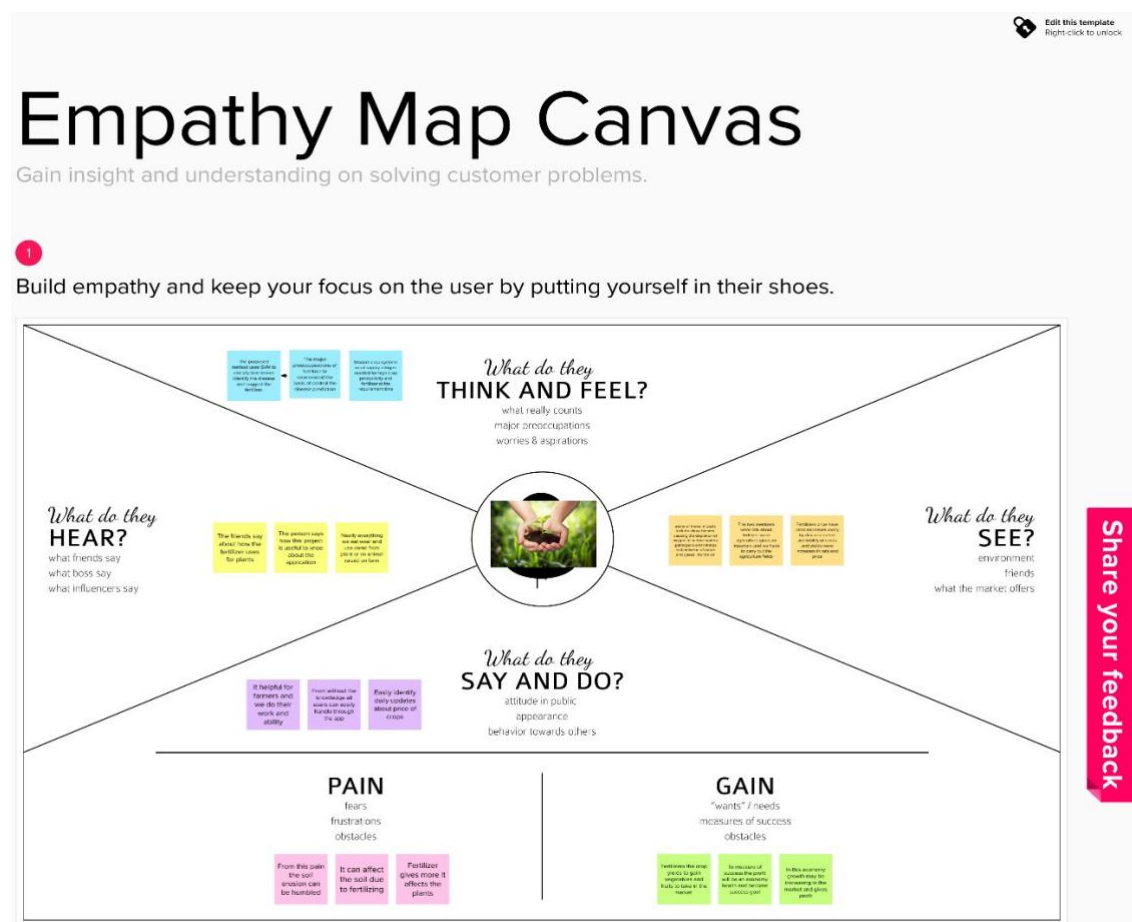
2.2 Neural Network Based Fertilizers Recommendation _System For Disorder Classification And Prediction In Petal Images .This methodology requires experts

who can recognize varieties in leaf shading. Ordinarily a similar malady is characterized by a few specialists as a different sickness. This arrangement is exorbitant, in light of the fact that it requires nonstop expert management

2.3 Agriculture is the most important sector in today's life. Most of the plants are affected by a wide variety of bacterial and fungal diseases. In agricultural aspects, if the plant is affected by leaf disease then it reduces the growth and productiveness. Generally, the plant diseases are caused by the abnormal physiological functionalities of plants

3.IDEATION & PROPOSED SOLUTION

3.1Empathy Map Canvas



3.2Ideation & Brainstorming



Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

- 🕒 10 minutes to prepare
- 👥 1 hour to collaborate
- 👤 2-8 people recommended

[Share template feedback](#)



Need some inspiration?
See a featured version of this template to kickstart your work.

[Open example](#)



Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

🕒 10 minutes

- Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.
- Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.
- Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#)



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

Fertilizer
Recommendation
System For Disease
Prediction
This may help to
avoid and control the
disease while using
the fertilizer



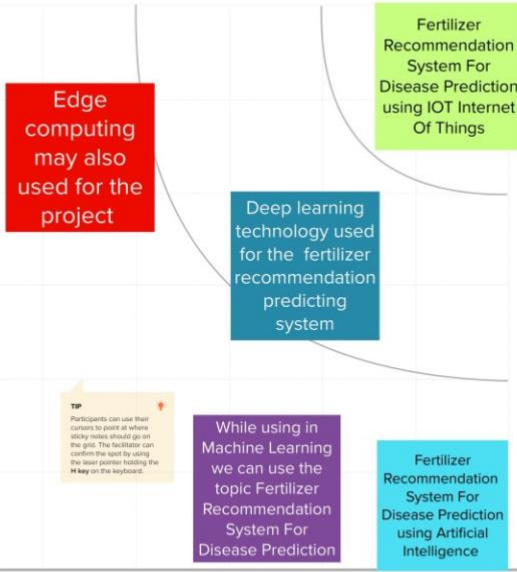
Key rules of brainstorming

To run a smooth and productive session

- 🗣️ Stay in topic.
- 💡 Encourage wild ideas.
- ⏸️ Defer judgment.
- 👂 Listen to others.
- 🗣️ Go for volume.
- 👁️ If possible, be visual.

the same page about what's important moving
in this grid to determine which ideas are important and

TIP
Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as they're added to your mural.



TIP
Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the H key on the keyboard.

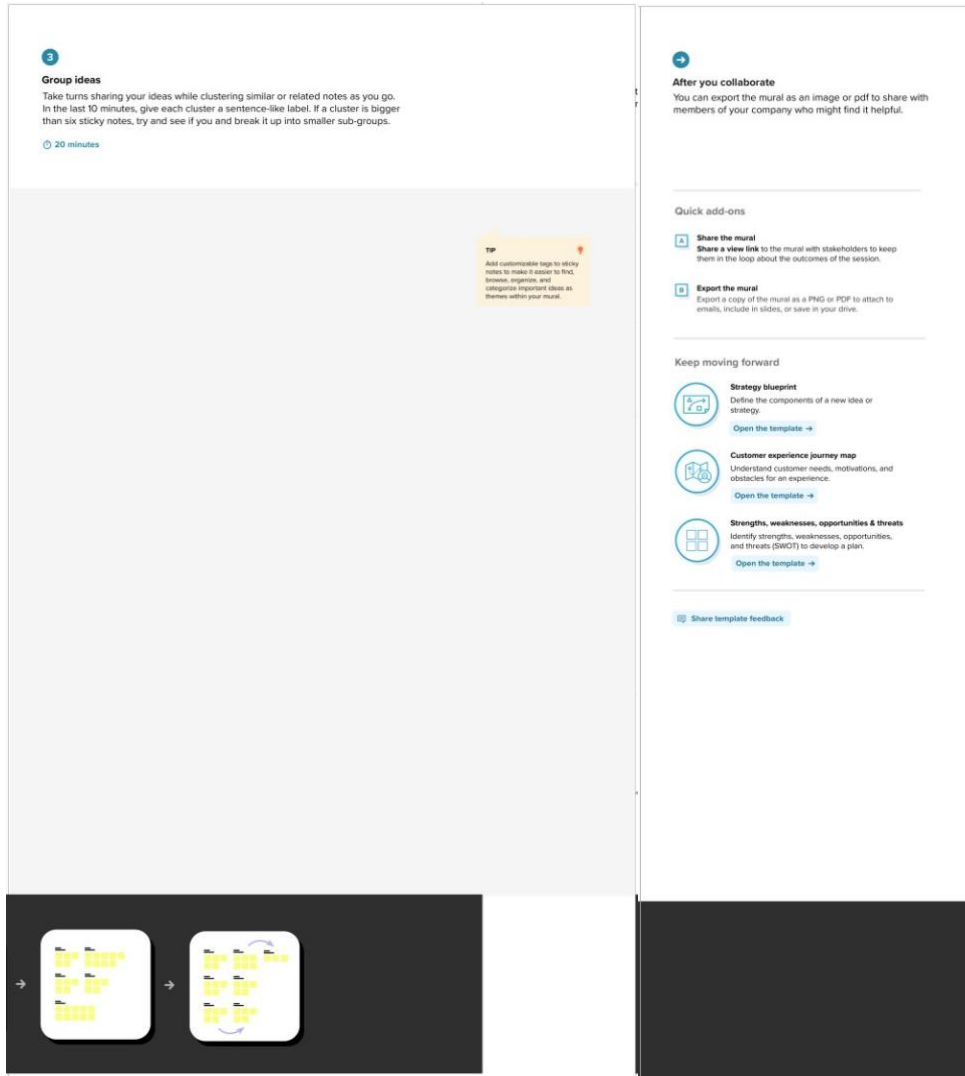
While using in Machine Learning we can use the topic Fertilizer Recommendation System For Disease Prediction

Fertilizer Recommendation System For Disease Prediction using Artificial Intelligence


Feasibility

Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)





3.3 Proposed Solution

S.No.	Parameter	Description
1	Problem Statement (Problem to be solved)	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.
2	Idea / Solution description	Develop an website for an farmers
3	Novelty / Uniqueness	The major agricultural products in India are rice, wheat, pulses, and spices. As our population is increasing rapidly the demand for agriculture products also increasing alarmingly. A huge amount of data are incremented from various field of agriculture.

4	Social Impact / Customer Satisfaction	The project helps us to know about what type of disease were attacked in our plants how to recover using fertilizers are the main concept.
5	Business Model (Revenue Model)	Increasing demand for agricultural production owing to the growing population. Widening information management systems as well as new advanced technologies adoption for improving crop productivity.
6	Scalability of the Solution	These tools are improving information about soils and vegetation that forms the basis for investments in management actions, provides early warning of pest and disease outbreaks, and facilitates the selection of sustainable cropland management practices.

3.4 Problem Solution fit

Agriculture is one field which has a high impact on life and economic status of human beings. Improper management leads to loss in agricultural products. Farmers lack the knowledge of disease and hence they produce less production. Farmers are unable to explain disease properly on call need to analysis the image of affected area of disease. Though, images and videos of crops provide better view and agro scientists can provide a better solution to resolve the issues related to healthy crop yet it not been informed to farmers. It is required to note that if the productivity of the crop is not healthy, it has high risk of providing good and healthy nutrition. Due to the improvement and development in technology where devices are smart enough to recognize and detect plant diseases. Recognizing illness can prompt faster treatment in order to lessen the negative impacts on harvest. This paper therefore focus upon plant disease detection using image processing approach. This work utilizes an open dataset of pictures of unhealthy and solid plants, where convolution system and semi supervised techniques are used to characterize crop species and detect the sickness. Farmers can interact with portal build interact with user interface to upload images of diseases leaf the image will be processing and train data to the algorithm.

These types of algorithm may evaluate the processing of image and predict to the user interface.

4.REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

FR-1	User registration	Registration through form Registration through Gmail
FR-2	User confirmation	Confirmation via OTP Confirmation via Email
FR-3	Capturing image	Capture the image of the leaf And check the parameter of the captured image .
FR-4	Image processing	Upload the image for the prediction of the disease in the leaf.
FR-5	Leaf identification	Identify the leaf and predict the disease in leaf.
FR-6	Image description	Suggesting the best fertilizer for the disease.
FR.NO	Functional requirement	Sub requirement (story/subtask)

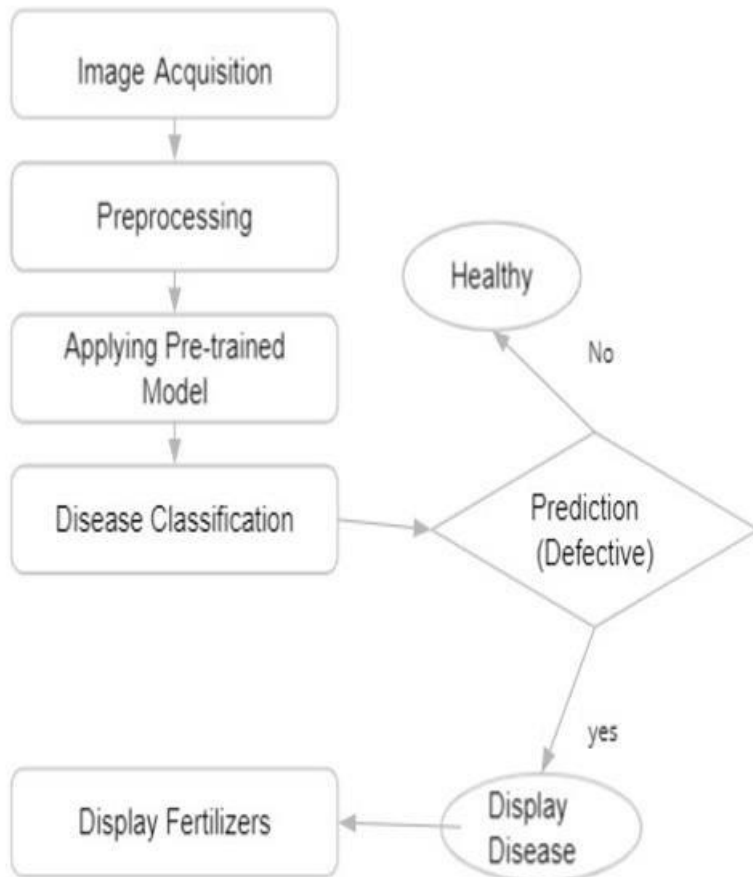
4.2 NON FUNCTIONAL REQUIREMENT

NFR.NO	Non-functional requirement	Description
NFR-1	Usability	Datasets of all the leaf is used to detecting the disease that present in the leaf.
NFR-2	Security	The information belongs to the user and leaf are secured highly.
NFR-3	Reliability	The leaf quality is important for the predicting the disease in leaf.
NFR-4	Performance	The performance is based on the quality of the leaf used for disease prediction
NFR-5	Availability	It is available for all user to predict the disease in the plant

NFR-6	Scalability	Increasing the prediction of the disease in the leaf
-------	-------------	--

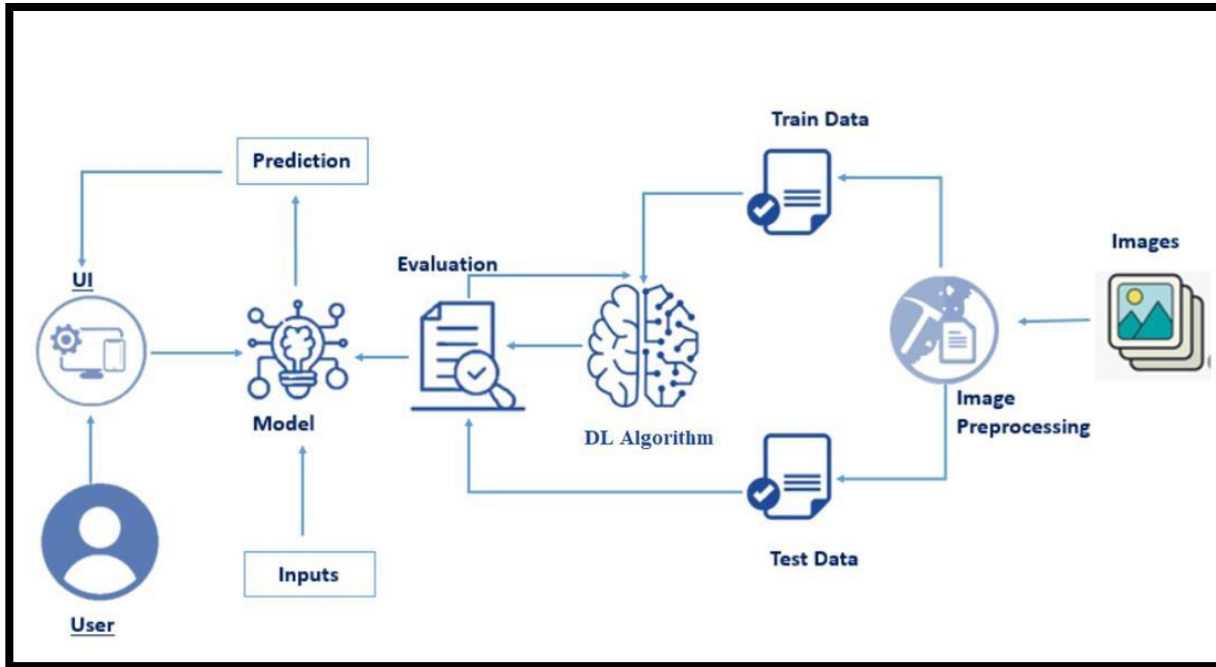
5. PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture

From this technical architecture and solution the user can take the picture of an image and the image will be processing the data the train data and test data are should be using in AI algorithm .The evaluated image could be predict the disease and gives an solution for the attacked leaf.



5.3 User Stories

The user can register in the account and may use mail id and password to login the page and make it useful for farmers. The image can be processed and data should be held by it. If you predict the images can be taken as a photo and a predict button should be made. Then the prediction will appear. If the leaf is healthy it shows healthy. If the leaf is not healthy it shows the prediction for fertilizer.

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points (Total)	Priority	Team Members
Sprint1	Model Creation and Training (Fruits)	FRSFDP-44	Create a model which can classify diseased fruit plants from given images. I also need to test the model and deploy it on IBM Cloud	8	High	Iswarya M
	Model Creation and Training (Vegetables)	FRSFDP-45	Create a model which can classify diseased vegetable plants from given images	2	Medium	Arithra

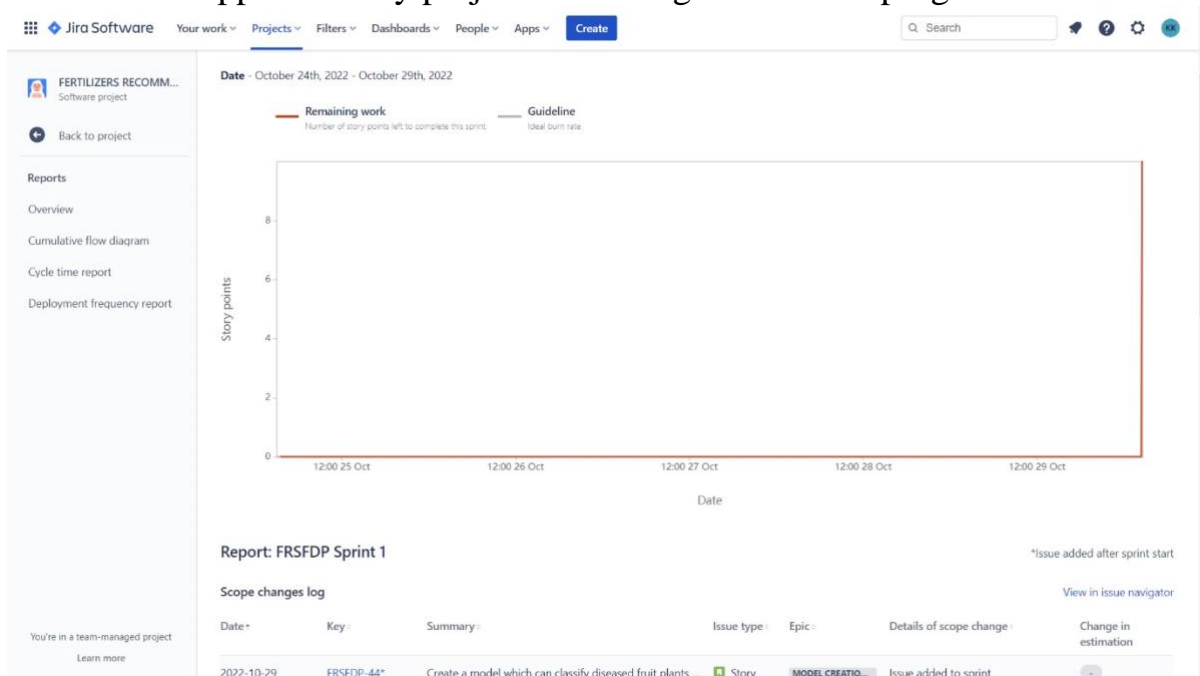
Sprint2	Model Creation and Training (Vegetables)	FRSFDP-46	Create a model which can classify diseased vegetable plants from given images and train on IBM Cloud	6	High	Arithra
	Registration	FRSFDP-47	As a user, I can register by entering my email, password, and confirming my password or via OAuth API	3	High	Sneka
	Upload page	FRSFDP-48	As a user, I will be redirected to a page where I can upload my pictures of crops	4	High	Iswarya
	Suggestion results	FRSFDP-49	As a user, I can view the results and then obtain the suggestions provided by the ML model	4	High	Arithra
	Base Flask App	FRSFDP-50	A base Flask web app must be created as an interface for the ML model	2	High	Arthi
Sprint3	Login	FRSFDP-51	As a user/admin/shopkeeper, I can log into the application by entering email & password	2	High	Sneka
	User Dashboard	FRSFDP-52	As a user, I can view the previous results and history	3	Medium	Sneka
	Integration	FRSFDP-53	Integrate Flask, CNN model with Cloudant DB	5	Medium	Iswarya
	Containerization	FRSFDP-54	Containerize Flask app using Docker	2	Low	Arithra
Sprint4	Dashboard (Admin)	FRSFDP-55	As an admin, I can view other user details and uploads for other purposes	2	Medium	Arthi
	Dashboard (Shopkeeper)	FRSFDP-56	As a shopkeeper, I can enter fertilizer products and then update the details	2	Low	Sneka
	Containerization	FRSFDP57	Create and deploy Helm charts using Docker Image made before	2	Low	Iswarya
	Logout	FRSFDP58	After finishing the process then logout	2	Low	Arithra

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	10	6 Days	24 Oct 2022	29 Oct 2022	10	30 Oct 2022
Sprint-2	15	6 Days	31 Oct 2022	05 Nov 2022	15	06 Nov 2022
Sprint-3	15	6 Days	07 Nov 2022	12 Nov 2022	15	13 Nov 2022
Sprint-4	12	6 Days	14 Nov 2022	19 Nov 2022	10	20 Nov 2022

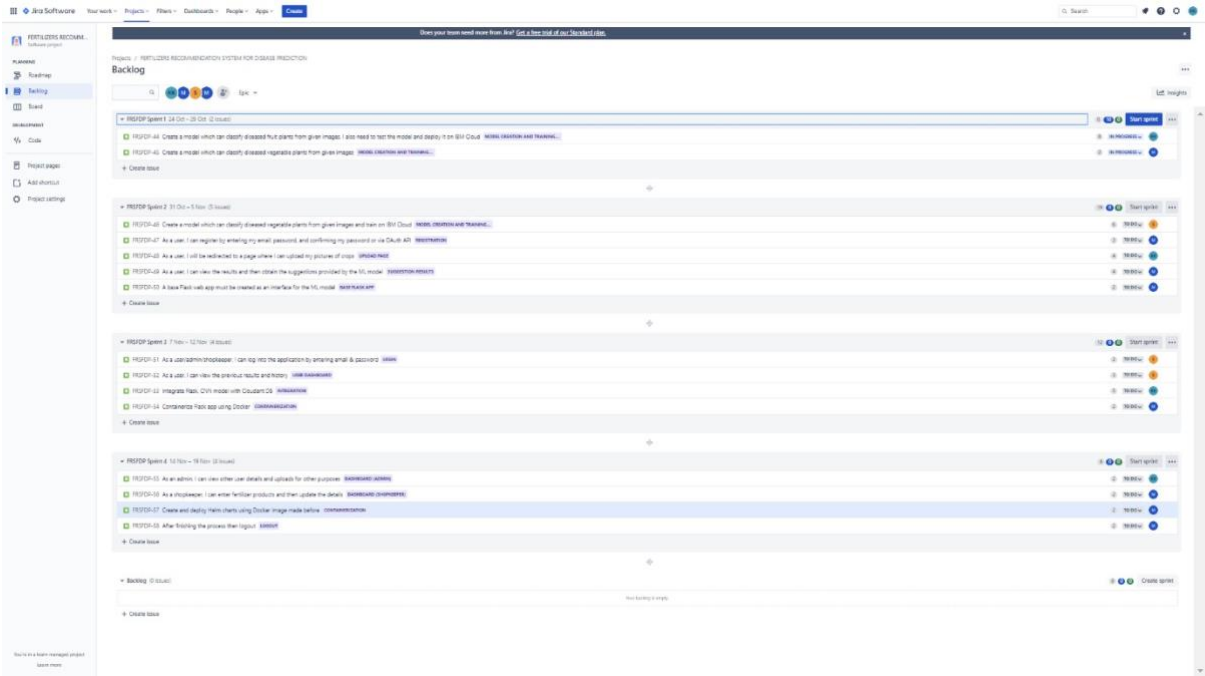
BURNDOWN CHART

A burn down 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.

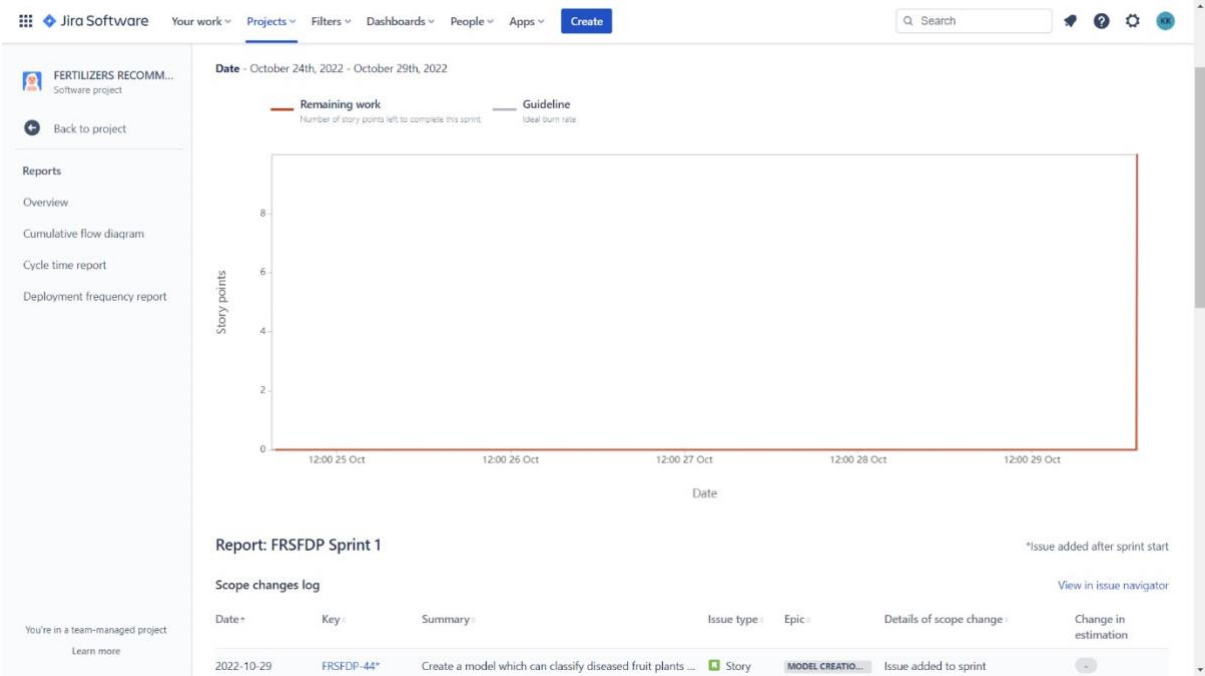


6.3 Reports from JIRA

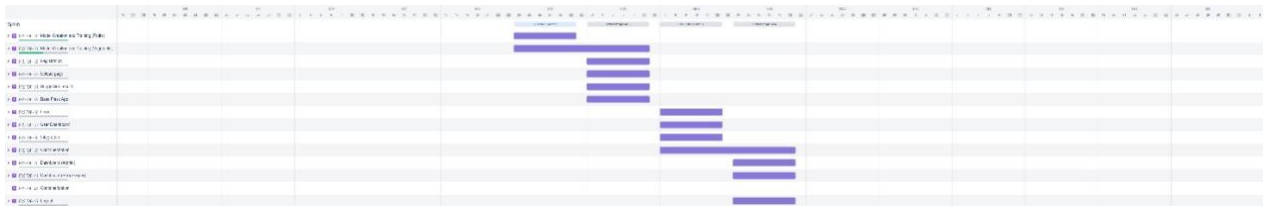
ACTIVITY LIST



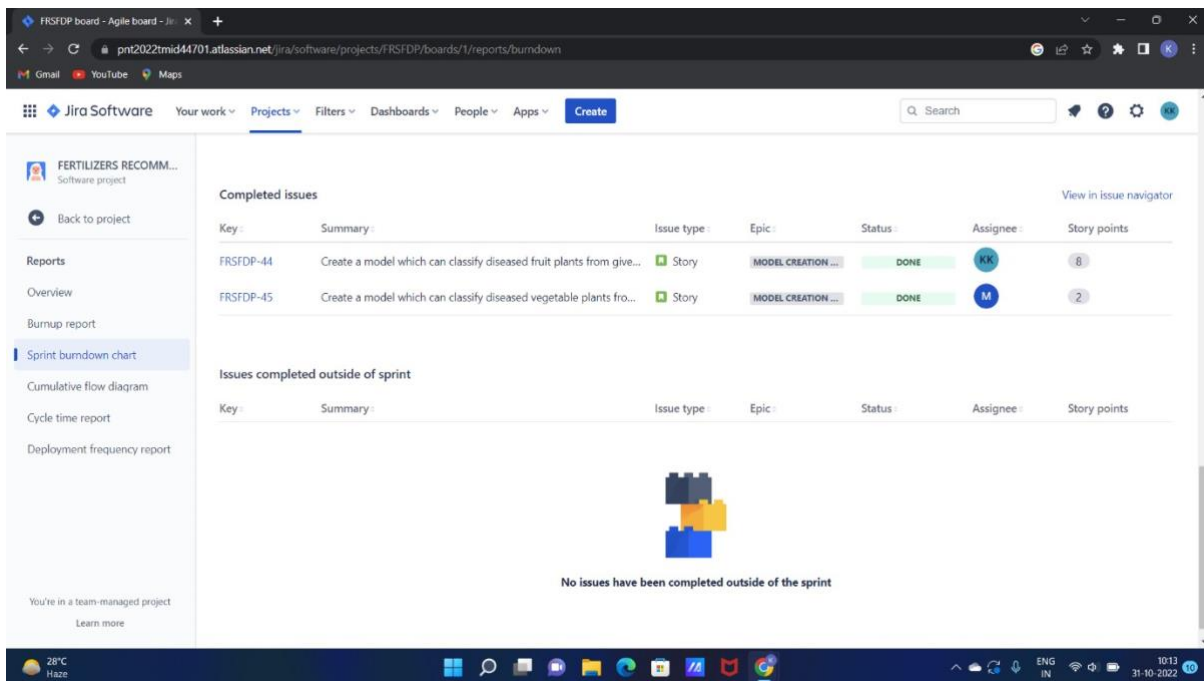
BURNDOWNCHART



ROAD MAP



SPRINT BURNDOWNCHART



7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1

```
<!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;
}

/* Change styles for span and cancel button on extra small screens */
@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 */  
.slideshow-container { max-  
width: 1000px; position:  
relative;  
margin: auto;  
}  
  
/* Caption text */  
.text { color:  
#f2f2f2; font-size:
```

```
15px; padding: 8px
12px; position:
absolute; bottom:
8px; width: 100%;
text-align: center;
}
/* The dots/bullets/indicators */
.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;
}

/* Fading animation */
.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}
}

/* On smaller screens, decrease text size */
@media only screen and (max-width: 300px) {
.text {font-size: 11px}
}
</style>
</head>
```

```
<body style="font-family:'Times New Roman', Times,
serif;backgroundcolor:#C2C5A8;">
```

```
<div class="header">
<div style="width:50%;float:left;font-size:2vw;text-align:left;color:white;
paddingtop: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;paddingright:30px;text-align:justify;">Agriculture is one of the major sectors
works 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>
<imgsrc="{{url_for('static',filename='images/12456.png')}}"
style="maxheight: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>

```

```

<!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://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css" rel="stylesheet">
<script src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>
<script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>
<script src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
<link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet' type='text/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'>
<link href="{ url_for('static', filename='css/final.css') }" 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;
}
```

```
.login{
margin-top:-70px;
}
body {
```

```

    background-color:#ffffff; background-
repeat: no-repeat; background-size:cover;
background-position: 0px 0px;
}
.login{
    margin-top:100px;
}

.container { margin-
top:40px;
padding: 16px;
} select
{
width: 100%;          margin-bottom: 10px;
background: rgba(255,255,255,255);
border: none;         outline:
none; padding: 10px;
font-size: 13px;
    color: #000000;
text-shadow: 1px 1px 1px rgba(0,0,0,0.3); border: 1px
solid rgba(0,0,0,0.3);
    border-radius: 4px;
    box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px rgba(255,255,255,0.2);
    -webkit-transition: box-shadow .5s ease;
    -moz-transition: box-shadow .5s ease;
    -o-transition: box-shadow .5s ease;      -ms-transition:
box-shadow .5s ease;
    transition: box-shadow .5s ease;

}

</style>
</head>

<body style="font-family:Montserrat;overflow:scroll;">

<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%;">

</div>
</div>
<div class="container">
<div id="content" style="margin-top:2em">
    <div class="container">
        <div class="row">

```

```
<div class="col-sm-6 bd" >

    <br>
    <imgsrc="{{url_for('static',filename='images/789.jpg')}}"
style="height:450px;width:550px"class="img-rounded" alt="Gesture">
    </div>
    <div class="col-sm-6">
        <div>
            <h4>Drop in the image to get the prediction </h4>
            <form action = "" id="upload-file" method="post" enctype="multipart/formdata">
                <select name="plant">

                    <option value="select" selected>Select plant type</option>
                    <option value="fruit">Fruit</option>
                    <option value="vegetable">Vegetable</option>                </select><br>
                    <label for="imageUpload" class="upload-label" style="background:
#28272c;">
                        Choose...
                    </label>
                    <input type="file" name="image" id="imageUpload" accept=".png,
.jpg, .jpeg">
                </form>

                <div class="image-section" style="display:none;">
                    <div class="img-preview">
                        <div id="imagePreview">
                        </div>
                    </div>
                    <div>
                        <button type="button" class="btn btn-info btn-lg " id="btnpredict"
style="background: #28272c;">Predict!</button>
                    </div>
                </div>

                <div class="loader" style="display:none;"></div>

                <h3>
                    <span id="result" style="font-size:17px; "></span>
                </h3>
            </div>
        </div>
    </div>
    </div>
    </div>
</div>
</body>
```

```
<footer>
<script src="{{ url_for('static', filename='js/main.js') }}"
type="text/javascript"></script></footer>
</html>
```

7.2 Feature 2

```
import requests from
tensorflow.keras.preprocessing import image from
tensorflow.keras.models import load_model
import numpy as np import pandas as pd import
tensorflow as tf
from flask import Flask, request, render_template, redirect, url_for import
os
from werkzeug.utils import secure_filename
from tensorflow.python.keras.backend import set_session

app = Flask(__name__)

#load both the vegetable and fruit models model
= load_model("vegetable.h5")
model1=load_model("fruit.h5")

#home page
@app.route('/') def home():  return
render_template('home.html')
#prediction page
@app.route('/prediction')
def prediction():

    return render_template('predict.html')

@app.route('/predict',methods=['POST'])
def predict():  if request.method ==
'POST':      # Get the file from post
request
    f = request.files['image']

    # Save the file to ./uploads
basepath = os.path.dirname(__file__)
file_path = os.path.join(
basepath, 'uploads', secure_filename(f.filename))
f.save(file_path)    img = image.load_img(file_path,
target_size=(128, 128))

    x = image.img_to_array(img)
```

```

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

plant=request.form['plant']
print(plant)
if(plant=="vegetable"):
    preds = model.predict(x)
    preds=np.argmax(preds)
    print(preds)
df=pd.read_excel('precautions - veg.xlsx')
print(df.iloc[preds]['caution'])    else:
    preds = model1.predict(x)        preds=np.argmax(preds)
df=pd.read_excel('precautions - fruits.xlsx')
print(df.iloc[preds]['caution'])

return df.iloc[preds]['caution']

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

```

8. TESTING

8.1 Test Cases

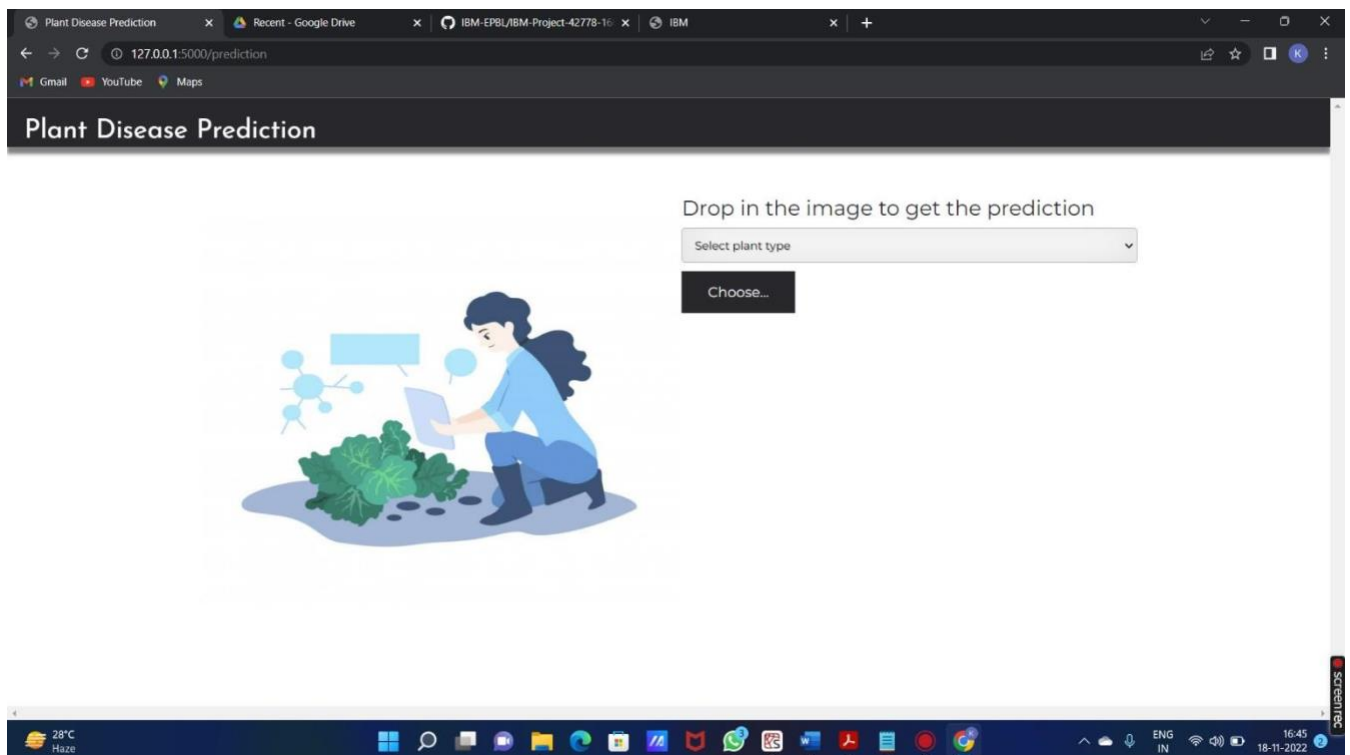
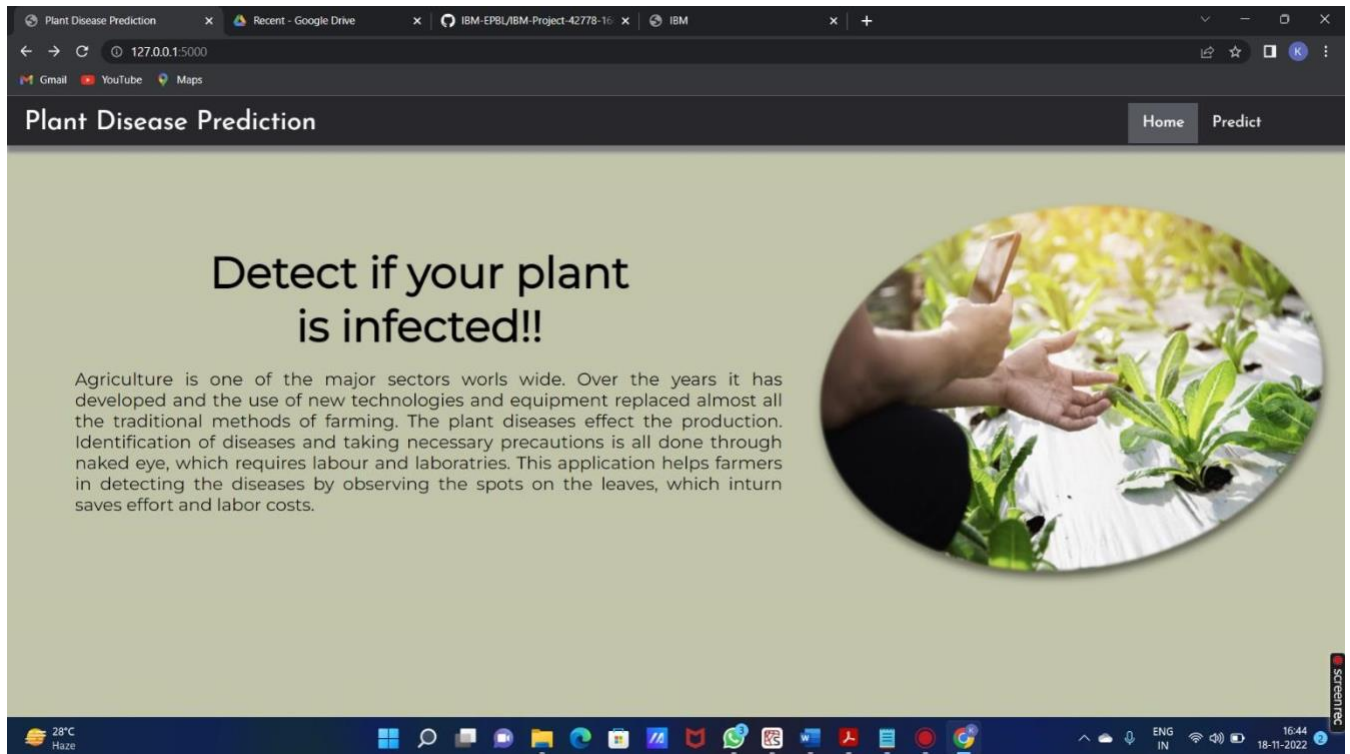
SECTION	TOTAL CASES	NOT TESTED	FAIL	PASS
Leaf spots	17	0	0	17
Mosaic Leaf Pattern	51	0	0	51
Misshapen Leaves	20	0	0	20
Yellow Leaves	7	0	0	7
Fruit Rots	9	0	0	9
Fruit Spots	4	0	0	4
Blights	2	0	0	2

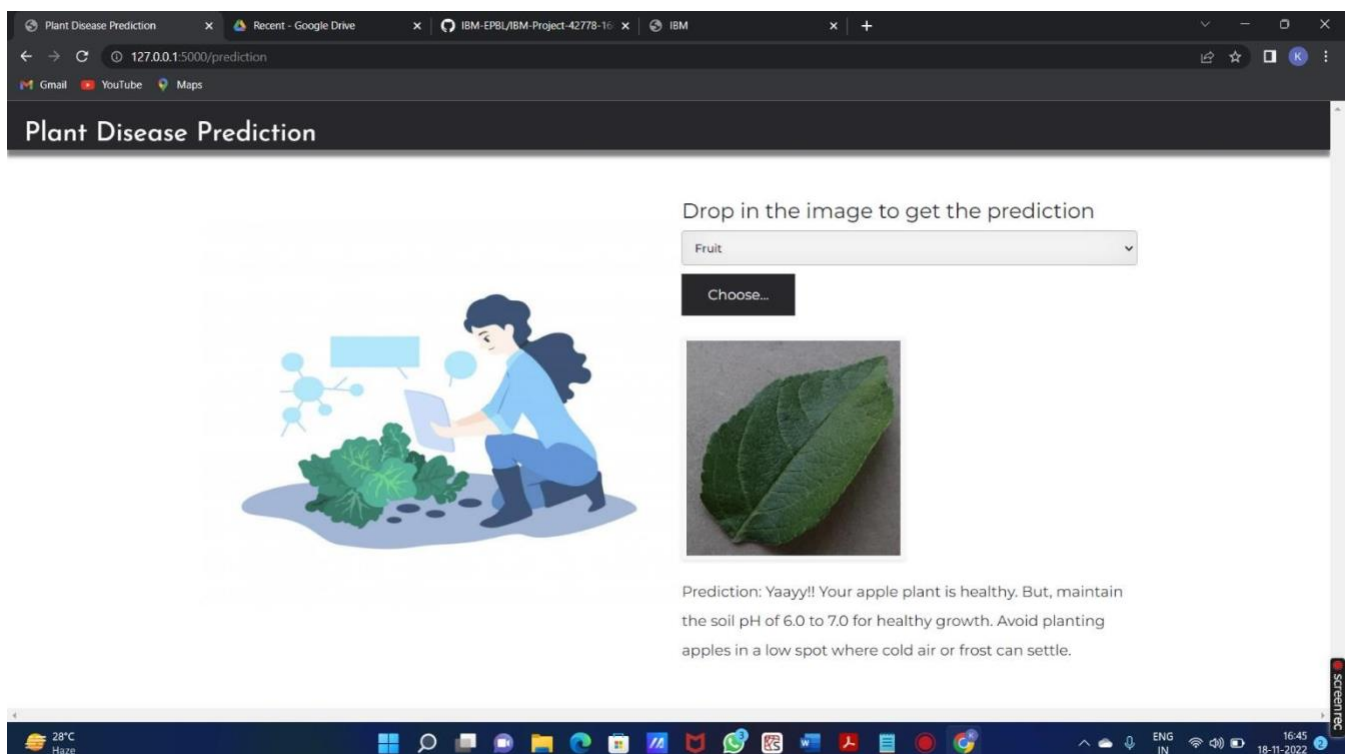
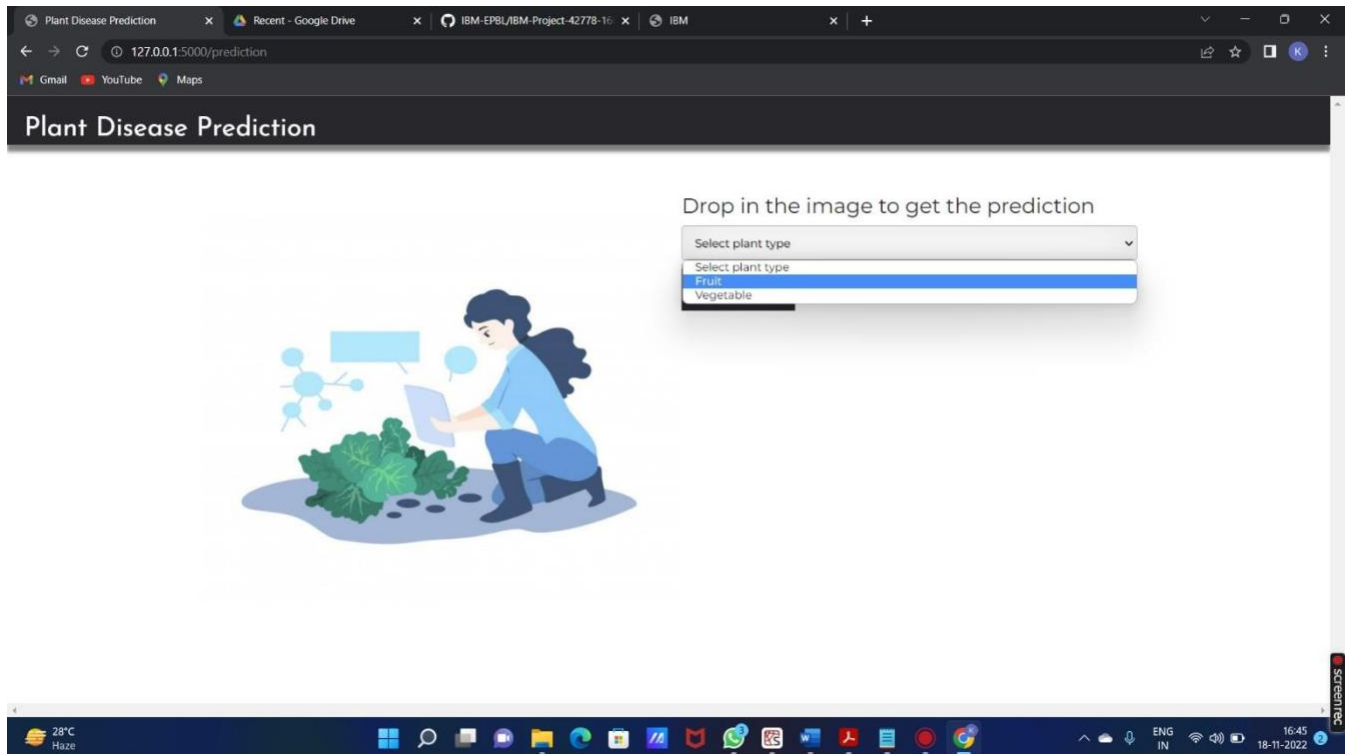
8.2 User Acceptance Testing

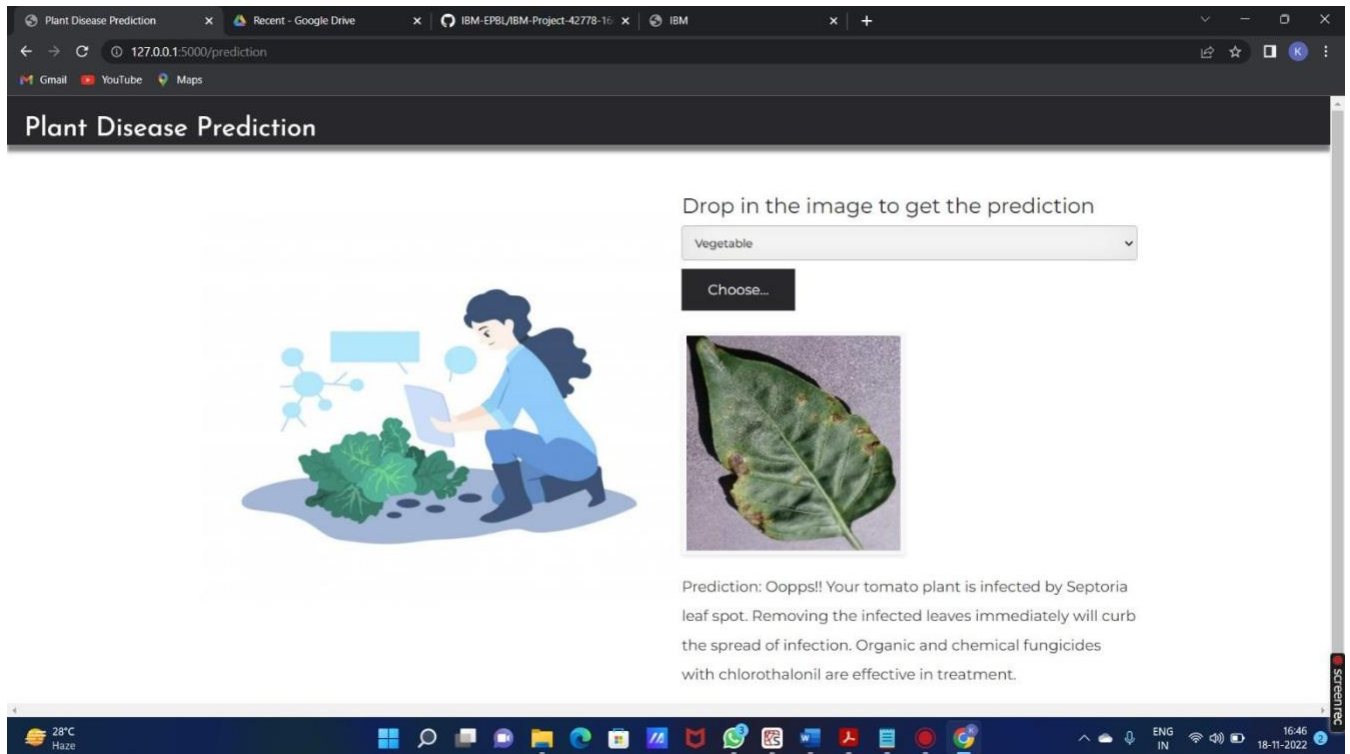
RESOLUTION	SEVERITY 1	SEVERITY 2	SEVERITY 3	SEVERITY 4	SUBTOTAL
Leaf spots	10	4	2	3	19
Mosaic Leaf Pattern	9	6	3	6	24
Misshapen Leaves	2	7	0	1	10
Yellow Leaves	11	4	3	20	38
Fruit Rots	3	2	1	0	6
Fruit Spots	5	3	1	1	10
Blights	4	5	2	1	12
Totals	44	31	13	32	119

9. RESULTS

9.1 Performance Metrics







10. ADVANTAGES & DISADVANTAGES

List of advantages

- The proposed model here produces very high accuracy of classification.
- Very large datasets can also be trained and tested.
- Images of very high can be resized within the proposed itself.

List of disadvantages

- For training and testing, the proposed model requires very high computational time.
- The neural network architecture used in this project work has high complexity.

11. CONCLUSION

The model proposed here involves image classification of fruit datasets and vegetable datasets. The following points are observed during model testing and training:

- The accuracy of classification increased by increasing the number of epochs.

- For different batch sizes, different classification accuracies are obtained.
- The accuracies are increased by increasing more convolution layers.
- The accuracy of classification also increased by varying dense layers.
- Different accuracies are obtained by varying the size of kernel used in the convolution layer output.
- Accuracies are different while varying the size of the train and test datasets.

12. FUTURE SCOPE

The proposed model in this project work can be extended to image recognition. The entire model can be converted to application software using python to exe software. The real time image classification, image recognition and video processing are possible with help OpenCV python library. This project work can be extended for security applications such as figure print recognition, iris recognition and face recognition.

13. APPENDIX

Source Code

```
import requests from
tensorflow.keras.preprocessing import image from
tensorflow.keras.models import load_model
import numpy as np import pandas as pd import
tensorflow as tf
from flask import Flask, request, render_template, redirect, url_for import
os
from werkzeug.utils import secure_filename
from tensorflow.python.keras.backend import set_session

app = Flask(__name__)

#load both the vegetable and fruit models model
= load_model("vegetable.h5")
model1=load_model("fruit.h5")
```

```

#home page
@app.route('/') def home():    return
render_template('home.html')

#prediction page
@app.route('/prediction') def
prediction():    return
render_template('predict.html')

@app.route('/predict',methods=['POST'])
    def predict():    if request.method
== 'POST':        # Get the file from post
request
    f = request.files['image']

    # Save the file to ./uploads
basepath = os.path.dirname(__file__)
file_path = os.path.join(
basepath, 'uploads', secure_filename(f.filename))
f.save(file_path)    img = image.load_img(file_path,
target_size=(128, 128))

    x = image.img_to_array(img)
    x = np.expand_dims(x, axis=0)

    plant=request.form['plant']
print(plant)
if(plant=="vegetable"):
preds = model.predict(x)
preds=np.argmax(preds)
    print(preds)
df=pd.read_excel('precautions - veg.xlsx')
    print(df.iloc[preds]['caution'])
else:
preds = model1.predict(x)    preds=np.argmax(preds)
df=pd.read_excel('precautions - fruits.xlsx')
print(df.iloc[preds]['caution'])

    return df.iloc[preds]['caution']

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

```

```

<!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;  
}
```

```
/* Change styles for span and cancel button on extra small screens */ @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 */  
.slideshow-container {  
max-width: 1000px;
```

```
position: relative; margin:
auto;
}
```

```
/* Caption text */
```

```
.text {
color: #f2f2f2;
font-size: 15px;
padding: 8px 12px;
position: absolute;
bottom: 8px; width:
100%; text-align:
center;
}
```

```
/* The dots/bullets/indicators */
```

```
.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;
}
```

```
/* Fading animation */
```

```
.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}
```

```

}

/* On smaller screens, decrease text size */
@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;
paddingtop: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;paddingright:30px;text-align:justify;">Agriculture is one of the major sectors
works 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>
<imgsrc="{ { url_for('static',filename='images/12456.png') } }"
style="maxheight:100%;max-width:100%;">

</div>
</div>

```

```
<div class="home">
```

```
<br>
```

```
</div>
```

```
<script>var slideIn
```

```
dex = 0;
```

```
showSlides();
```

```
function
```

```
showSlides() {
```

```
  vari;
```

```
  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>
```

```
<!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://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css"
rel="stylesheet">
<script src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>
<script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>
<script src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script><link
href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300'
rel='stylesheet' type='text/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'>
<link href="{ { url_for('static', filename='css/final.css') } }" 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;
}

.login{ margin-top:-70px;
} body
{
background-color:#ffffff; background-repeat:
no-repeat; background-size:cover;
background-position: 0px 0px;
}
.login{
margin-top:100px;
}

.container { margin-
top:40px; padding: 16px;
}
select {
width: 100%; margin-bottom: 10px;
background: rgba(255,255,255,255);
border: none; outline:
none; padding: 10px;
font-size: 13px; color:
#000000;
text-shadow: 1px 1px 1px rgba(0,0,0,0.3); border:
1px solid rgba(0,0,0,0.3); border-radius: 4px;
box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px rgba(255,255,255,0.2);
-webkit-transition: box-shadow .5s ease;

```

```
-moz-transition: box-shadow .5s ease;
-o-transition: box-shadow .5s ease;    -ms-transition:
box-shadow .5s ease;    transition: box-shadow .5s
ease;
}
```

```
</style>
</head>
```

```
<body style="font-family:Montserrat;overflow:scroll;">
```

```
<div class="header">
```

```
<div style="width:50%;float:left;font-size:2vw;text-align:left;color:white;
paddingtop:1%">Plant Disease Prediction</div>
```

```
<div class="topnav-right" style="padding-top:0.5%;">
```

```
</div>
```

```
</div>
```

```
<div class="container">
```

```
<div id="content" style="margin-top:2em">
```

```
    <div class="container">
```

```
        <div class="row">
```

```
            <div class="col-sm-6 bd" >
```

```
                <br>
```

```
                <imgsrc="{ {url_for('static',filename='images/789.jpg')}}" >
```

```
style="height:450px;width:550px"class="img-rounded" alt="Gesture">
```

```
            </div>
```

```
            <div class="col-sm-6">
```

```
                <div>
```

```
                    <h4>Drop in the image to get the prediction </h4>
```

```
                    <form action = "" id="upload-file" method="post" enctype="multipart/form-data">
```

```
                        <select name="plant">
```

```
                            <option value="select" selected>Select plant type</option>
```

```
                            <option value="fruit">Fruit</option>
```

```
                            <option value="vegetable">Vegetable</option>
```

```
                        </select><br>
```

```
                        <label for="imageUpload" class="upload-label" style="background:
```

```
#28272c;">
```

```
                            Choose...
```

```
        </label>
<input type="file" name="image" id="imageUpload" accept=".png, .jpg, .jpeg">
    </form>
```

```

    <div class="image-section" style="display:none;">
        <div class="img-preview">
            <div id="imagePreview">
            </div>
        </div>
        <div>
            <button type="button" class="btn btn-info btn-lg "
id="btn-predict" style="background: #28272c;">Predict!</button>
        </div>
    </div>
```

```
    <div class="loader" style="display:none;"></div>
```

```
    <h3>
        <span id="result" style="font-size:17px; "></span>
    </h3>
```

```

</div>
</div>
```

```

</div>
</div>
</div>
```

```
</div>
</body>
```

```
<footer>
<script src="{ { url_for('static', filename='js/main.js') } }" type="text/javascript"></script>
</footer>
</html>
.img-preview {
width: 256px; height:
256px; position: relative;
border: 5px solid #F8F8F8;
    box-shadow: 0px 2px 4px 0px rgba(0, 0, 0, 0.1);
margin-top: 1em; margin-bottom: 1em;
}
```



```
.img-preview>div { width:
100%; height: 100%;
background-size: 256px 256px;
background-repeat: no-repeat;
background-position: center;
}
```

```
input[type="file"] {
display: none;
}
```

```
.upload-label{
display: inline-block;
padding: 12px 30px;
background: #28272c;
color: #fff; font-size:
1em; transition: all
.4s; cursor: pointer;
}
```

```
.upload-label:hover{
background: #C2C5A8; color:
#39D2B4;
}
```

```
.loader { border: 8px solid #f3f3f3; /*
Light grey */ border-top: 8px solid
#28272c; /* Blue */ border-radius: 50%;
width: 50px; height: 50px;
animation: spin 1s linear infinite;
}
```

```
@keyframes spin {
0% { transform: rotate(0deg); }
100% { transform: rotate(360deg); }
}
```

GitHub & Project Demo Link
<https://github.com/IBM-EPBL/IBM-Project-5881-1658819040>

