

Project Report on

Fertilizers Recommendation System For Disease Prediction

Team ID : PNT2022TMID47755

Submitted by

M. Sathyapriyan (911619104030)

L. Clitus (911619104005)

V. Periyathambinadan (911619104022)

S. Prakash (911619104024)

TABLE OF CONTENTS

1. **INTRODUCTION**
 - 1.1 Project Overview
 - 1.2 Purpose
2. **LITERATURE SURVEY**
 - 2.1 Existing Solution and References
 - 2.2 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 Project Overview :

Agriculture is the most important sector in today's life. Most plants are affected by a wide variety of bacterial and fungal diseases. Diseases on 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 on 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 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.

This System will allow the Farmers to upload the Images of leaves of plants and crops with disease to it. Then the deep learning model will predict the disease by using trained model and show the disease and also provide a preferable Fertilizer for it.

It will reduce the fear and struggle to farmers and can also improve yield to farmers. The interface is easy and user friendly. So, any common people can efficiently use it.

1.2 Purpose :

Farmers are feeling struggled for finding the disease and solution to it. An Online method can be effective in providing solution to the problem of Farmers. They can just know the disease by simply uploading the images of leaves.

2 - LITERATURE SURVEY

2.1 Existing Solution and References :

1. Data mining algorithms are used on agriculture data. The main criterion for this categorization is that if the pH value is greater than 8.5, the soil is unsuitable for crop cultivation; otherwise, it is. To overcome this problem the proposed system will give necessary suggestion to increase or decrease the pH value of soil.
2. Measuring pH using a glass electrode. principles of the glass-electrode method.
3. Plant Disease Detection and Classification using CNN Model with Optimized Activation Function S. Yegneshwar Yadhav; T. Senthilkumar; S. Jayanthi; J. Judeson Antony Kovilpillai.
4. Plant Leaf Disease Detection and Classification Based on CNN with LVQ Algorithm - Melike Sardogan, Adem Tuncer, Yunus Ozen.
5. Disease Detection and Classification in Agricultural Plants Using Convolutional Neural Networks — A Visual Understanding

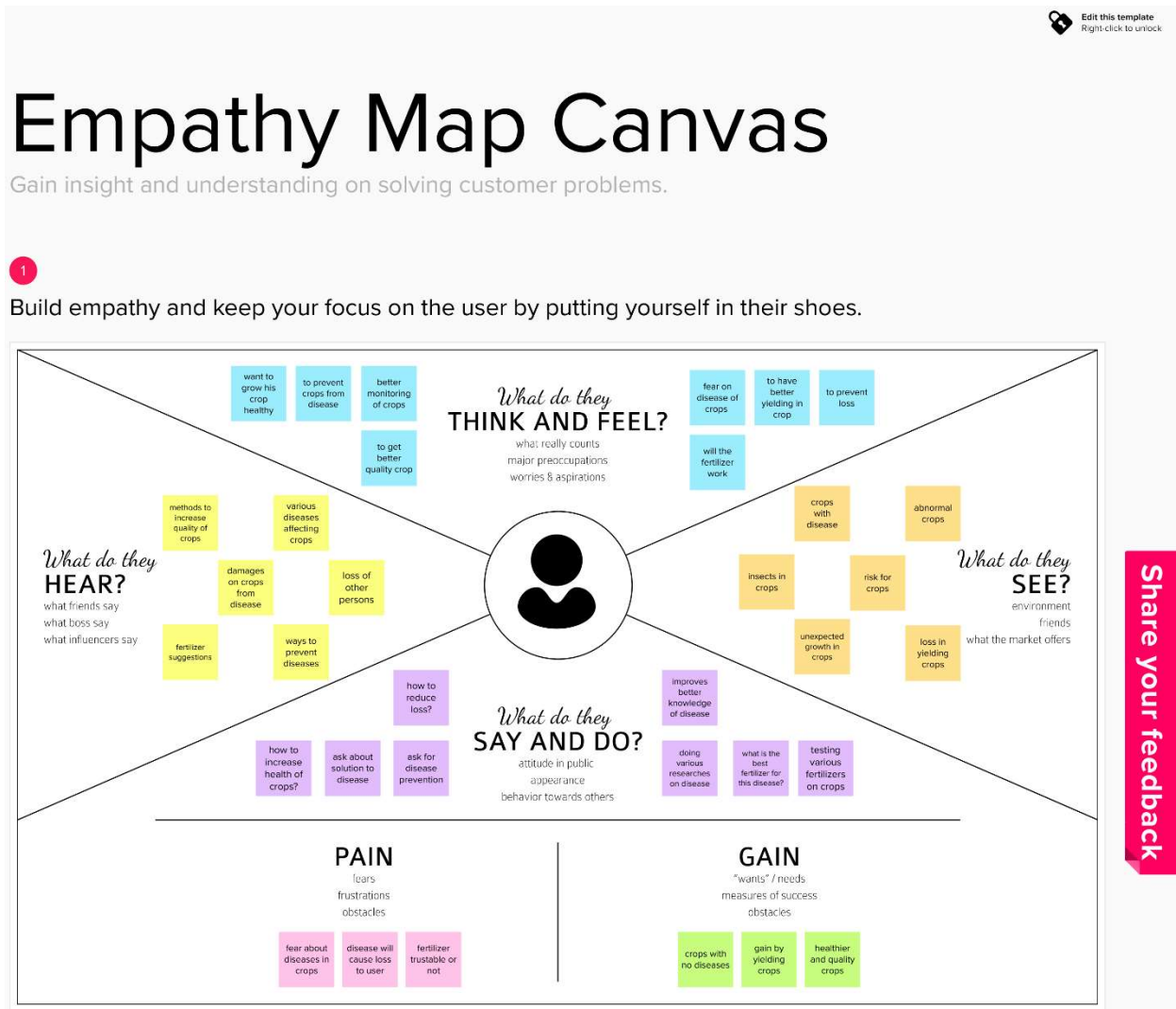
2.2 Problem Statement Definition :

I am **A Farmer**, I am Trying to **Grow some healthy crops**. But, **It is affected by diseases**. Because, **I can't find suitable Fertilizers** which makes me feel **Frustrated**

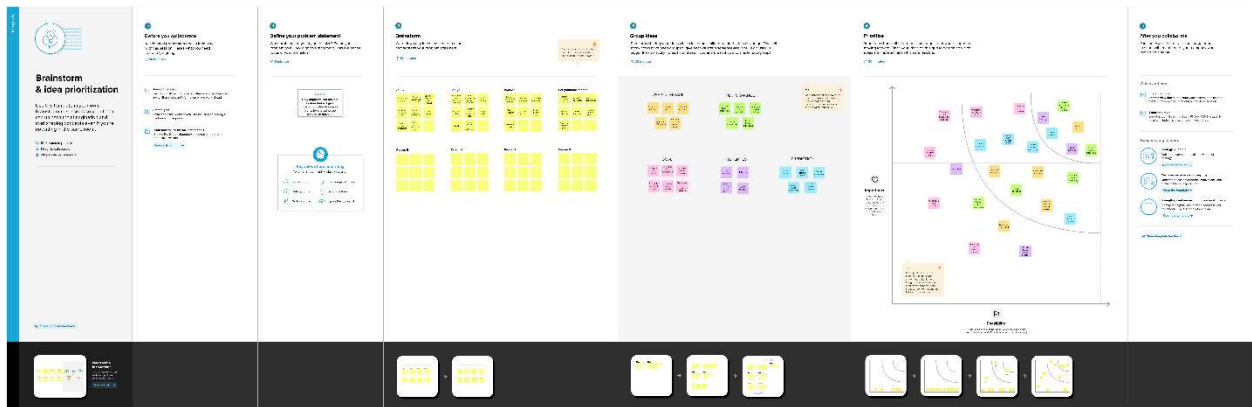


3 - IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



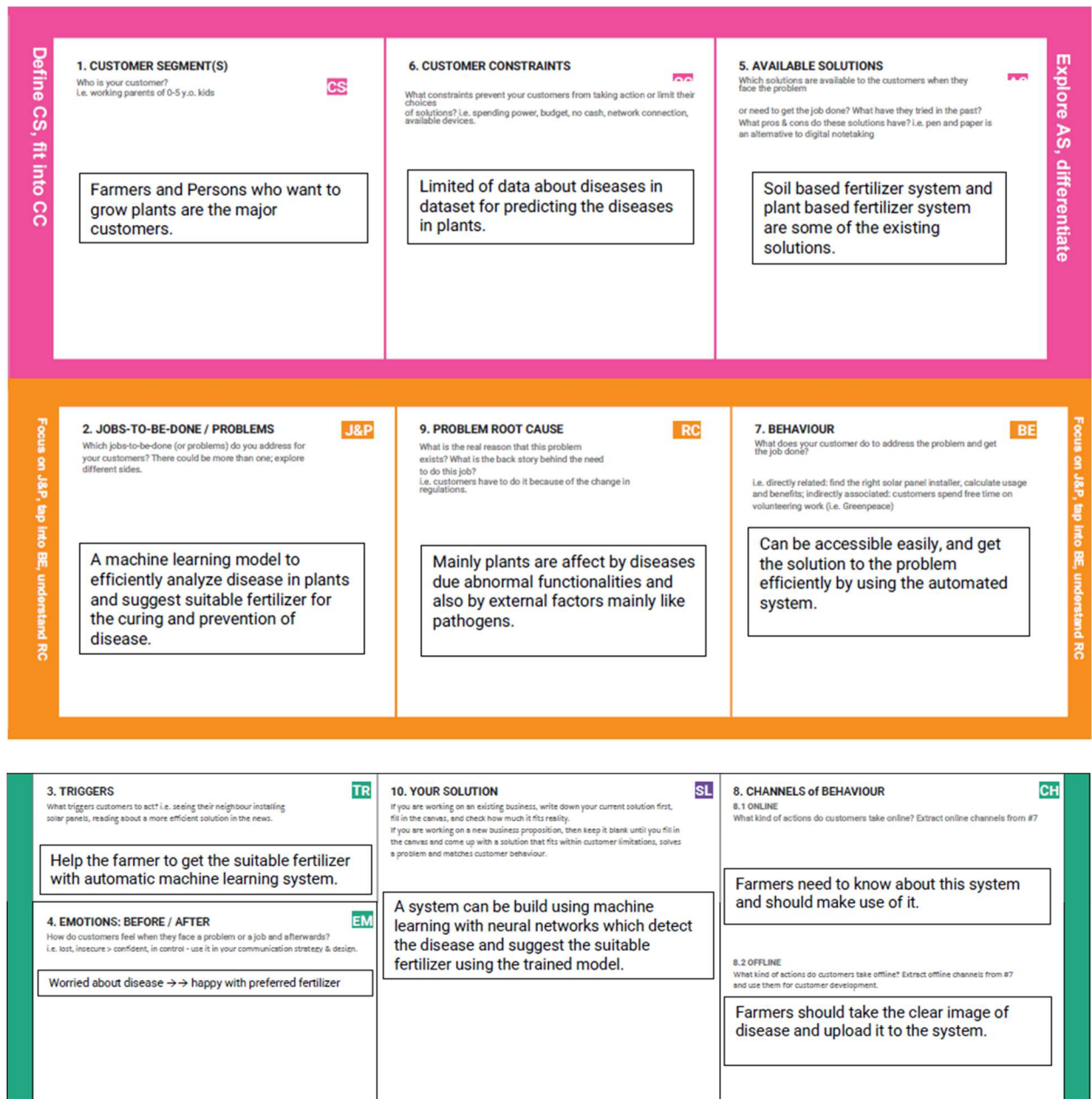
3.2 Ideation & Brainstorming



3.3 Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	<ul style="list-style-type: none"> Farmers cannot prevent the plants from affecting by disease. They are not knowing about suitable fertilizer for disease.
2.	Idea / Solution description	<ul style="list-style-type: none"> An automated machine learning system for predicting the disease from images that can be uploaded by farmers.
3.	Novelty / Uniqueness	<ul style="list-style-type: none"> More accurate in disease prediction Better suggestion of disease Dataset can upgrade as when required.
4.	Social Impact / Customer Satisfaction	<ul style="list-style-type: none"> Provide the complete solution through online Farmer can yield crops with good quality
5.	Business Model (Revenue Model)	<ul style="list-style-type: none"> Will increase the rate of buying fertilizers for industries. Will provide more quality crops.
6.	Scalability of the Solution	<ul style="list-style-type: none"> New diseases can be added to dataset for better prediction. Will be helpful to farmers who are not educated.

3.4 Problem Solution Fit :



4 - REQUIREMENT ANALYSIS

4.1 Functional Requirements:

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
FR-2	User Confirmation	Confirmation via Email
FR-3	User Profile	Fill the Profile page by providing required details
FR-4	Uploading Leaf Image	Image of leaf affected by disease want to be uploaded
FR-5	Requesting Solution	Image is processed and analysed by trained model and solution is generated
FR-6	Downloading Solution	The solution consists of disease and suggested fertilizer should be download in appropriate format.

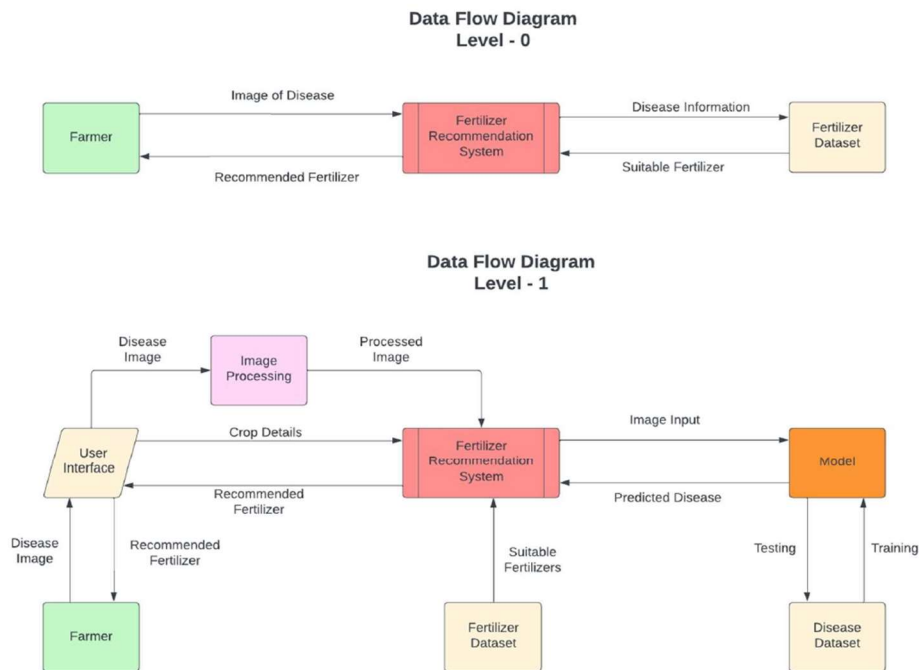
4.2 Non-functional Requirements:

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

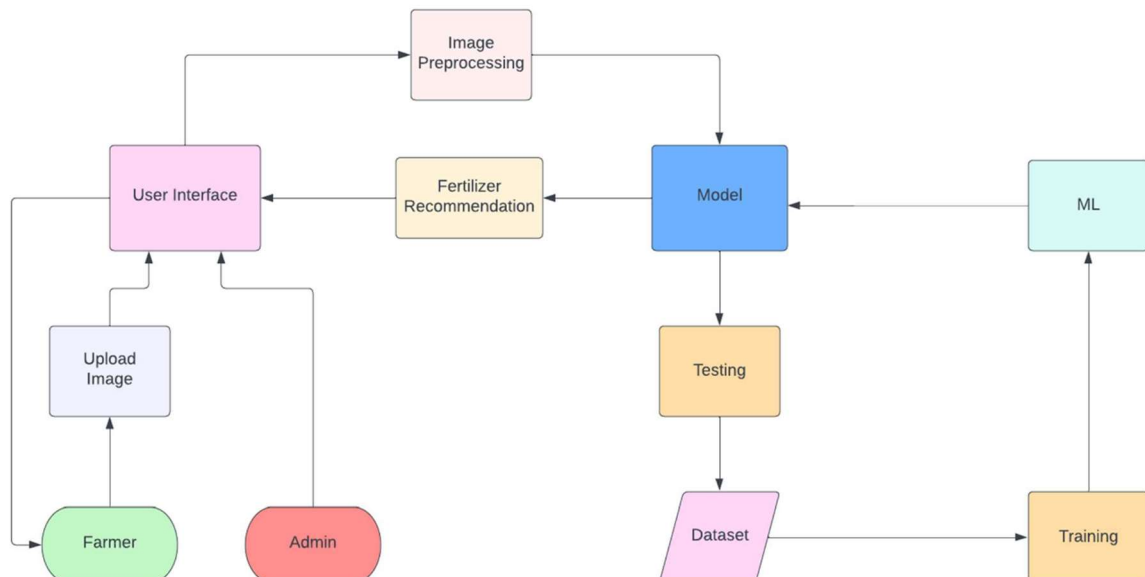
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Customer should access the system efficiently and easily.
NFR-2	Security	The data and information provided by the customer should be secure and prevent from any unauthorised user.
NFR-3	Reliability	The system must be reliable and trustable to the customer
NFR-4	Performance	To perform effectively and response must be fast
NFR-5	Availability	Should be available to all persons and accessible all the time
NFR-6	Scalability	Should be scalable in future

5 - PROJECT DESIGN

5.1 Data Flow Diagrams :



5.2 Solution and Technical Architecture :



Components & Technologies:

S.N o	Component	Description	Technology
1.	User Interface	How user interacts with application	HTML, CSS, JavaScript
2.	Application Logic-1	A Web page to upload Image of leaf with disease	Python
3.	Application Logic-2	Process the Image to given as input to Model	Python
4.	Application Logic-3	Trained Model gets the Image and use ML for prediction of disease	Python
5.	Database	Store Image data	MySQL
6.	Cloud Database	Database Service on Cloud Platform	IBM DB2
7.	File Storage	Store the data in a structure	Local Filesystem
8.	Machine Learning Model	A trained model is used for prediction of disease using Algorithms	Random Forest
9.	Infrastructure (Server / Cloud)	Application Deployment on Cloud Platform	IBM cloud

Application Characteristics:

S.N o	Characteristics	Description	Technology
1.	Open-Source Frameworks	A Web Framework written in python. It is Lightweight	Flask Web Framework
2.	Security Implementations	<ul style="list-style-type: none">Preventing from any malware attacksOnly accessible to Authorised Person	IBM cloud app ID
3.	Availability	Should be available at any time to every users	IBM cloud platform
4.	Performance	Should have a quick response time and userfriendly	Python, Flask

5.3 User Stories :

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Farmer	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
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Gmail	I can register & access the dashboard with Gmail ID	High	Sprint-1
	Login	USN-4	As a user, I can log into the application by entering email & password	I can Login and access my user account	High	Sprint-2
	Dashboard	USN-5	As a user, I can access the dashboard in the application and upload image of disease	I can access dashboard and upload images	High	Sprint-2
Fertilizer Shop Owner	Registration	USN-6	As a user, I can register for the website by entering my email, password, and confirming my password.	I can access my account / dashboard on website.	Medium	Sprint-3
	Login	USN-7	As a user, I can log into the website by entering email & password	I can Login into the web and access my user account	Medium	Sprint-3
	Dashboard	USN-8	As a user, I can access the dashboard in the website and upload image of disease	I can access web dashboard	Medium	Sprint-3
Administrator	Login	USN-9	As a admin, I can login for the website with my login credentials.	I can access my admin account	High	Sprint-4
	Dashboard	USN-10	As a admin, I can access the admin dashboard in the website	I can access admin dashboard on web	High	Sprint-4

6 - PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation :

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	Collect and downloading the dataset	2	High	Sathya priyan, Clitus, Periyathambi nadhan, Prakash
Sprint-1	Image Pre-processing	USN-2	Pre-process the images to applied for model	2	High	Sathya priyan, Clitus, Periyathambi nadhan, Prakash
Sprint-1	Model building for fruit disease prediction	USN-3	Building model to predict the disease in fruits	8	High	Sathya priyan, Clitus, Periyathambi nadhan, Prakash
Sprint-1	Model building for vegetable disease prediction	USN-4	Building model to predict the disease in vegetables	8	High	Sathya priyan, Clitus, Periyathambi nadhan, Prakash
Sprint-2	Train and testing The fruit disease prediction model	USN-5	Train the fruit disease detection model for prediction and perform testing	10	High	Sathya priyan, Clitus, Periyathambi nadhan, Prakash
Sprint-2	Train and testing The vegetable Disease prediction model	USN-6	Train the vegetable disease detection model for prediction and perform testing	10	High	Sathya priyan, Clitus, Periyathambi nadhan, Prakash

Sprint-3	Registration	USN-7	As a user, I can register to the website by entering email, password and confirming password	5	High	Sathya priyan, Clitus, Periyathambi nadhan, Prakash
Sprint-3	Login	USN-8	As a user, I can login into the website with my login credentials.	5	High	Sathya priyan, Clitus, Periyathambi nadhan, Prakash
Sprint-3	Dashboard	USN-9	As a user, I can access my dashboard in website and upload images.	10	High	Sathya priyan, Clitus, Periyathambi nadhan, Prakash
Sprint-4	Admin Login and Dashboard	USN-10	As an admin, I can access the admin dashboard to manage the website	10	Medium	Sathya priyan, Clitus, Periyathambi nadhan, Prakash
Sprint-4	Fertilizer shopkeeper Login and Dashboard	USN-11	As a shopkeeper, I can access my dashboard update the fertilizer details in the website	10	Medium	Sathya priyan, Clitus, Periyathambi nadhan, Prakash

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

6.3 Reports from JIRA :

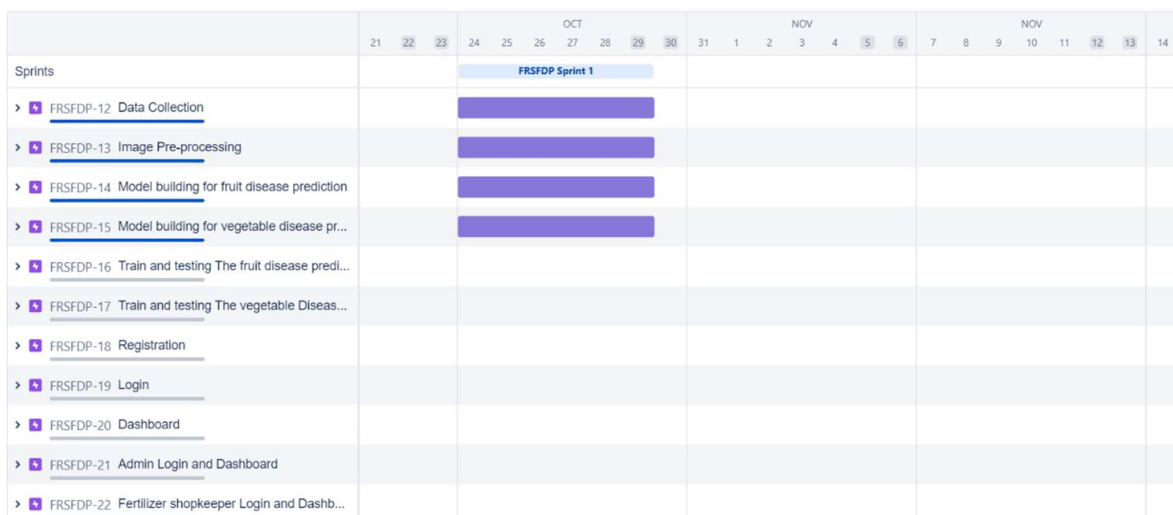
▼ Backlog (7 issues)		60	0	0	Create sprint
FRSFDP-5	Train the fruit disease detection model for prediction and perform testing	TRAIN AND TESTING THE FRUIT DI...	10	TO DO ▼	ST
FRSFDP-6	Train the vegetable disease detection model for prediction and perform testing	TRAIN AND TESTING THE VEGETA...	10	TO DO ▼	C
FRSFDP-7	As a user, I can register to the website by entering email, password and confirming password	REGISTRATION	5	TO DO ▼	S
FRSFDP-8	As a user, I can login into the website with my login credentials.	LOGIN	5	TO DO ▼	P
FRSFDP-9	As a user, I can access my dashboard in website and upload images.	DASHBOARD	10	TO DO ▼	C
FRSFDP-10	As an admin, I can access the admin dashboard to manage the website	ADMIN LOGIN AND DASHBOARD	10	TO DO ▼	ST
FRSFDP-11	As a shopkeeper, I can access my dashboard update the fertilizer details in the website	FERTILIZER SHOPKEEPER LOGIN A...	10	TO DO ▼	S
+ Create issue					

FRSFDP Sprint 1 24 Oct – 29 Oct (4 issues)

Data collection and model building

	Status	Assignee
FRSFDP-1 Collect and downloading the dataset DATA COLLECTION	In Review (2)	P
FRSFDP-2 Pre-process the images to applied for model IMAGE PRE-PROCESSING	In Review (2)	S
FRSFDP-3 Building model to predict the disease in fruits MODEL BUILDING FOR FRUIT DISE...	In Review (8)	ST
FRSFDP-4 Building model to predict the disease in vegetables MODEL BUILDING FOR VEGETABLE...	In Review (8)	C

+ Create issue



7 - CODING & SOLUTIONING

7.1 Feature 1:

- A Trained model is designed to be used to predict the diseases.
- Pre processing of Images for prediction by the model.

7.2 Feature 2:

- Home Page, Prediction Page and Result Page for showing prediction results.
- Register and Login page for Users.

7.3 Database Schema :

- IBM Cloudant DB

8 - TESTING

8.1 Test Cases:

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
HomePage_TC_001	Functional	Home Page	Verify user is able to see the home page or not.		1.Enter URL and click go. 2.verify whether the user is able to see the home page.	Enter URL and click go.	User able to see the home page.	Working as expected	Pass	Nil	N	-	Balaji T.S
HomePage_TC_002	UI	Home Page	Verify the UI elements in Home Page.		1.Enter URL and click go. 2.Verify the UI elements in Home Page.	Enter URL and click go.	Application should show below UI elements: Home Tab & Predict Tab	Working as expected	pass	Nil	N	-	C.J.Dinesh Kumar
PredictPage_TC_003	Functional	Predict page	Verify user is able to redirect to predict page or not.		1.Enter URL and click go. 2.Click on Predict button 3.Verify whether the user is redirect to predict page or not.	Click the predict button in home page	User should navigate to Predict page.	Working as expected	pass	Nil	N	-	S.Aparna
PredictPage_TC_004	UI	Predict page	Verify the UI elements in Predict Page.		1.Enter URL and click go. 2.Verify the UI elements in Predict Page.	Click the predict button and redirect to predict page.	Application should show below UI elements: Dropdown List , Upload file Button, Predict button.	Working as expected	pass	Nil	N	-	C.J.Dinesh Kumar
PredictPage_TC_005	Functional	Predict page	Verify user is able to select the dropdown value or not.		1.Enter URL and click go. 2.Click on Predict button 3.Verify whether the user is redirect to predict page or not. 4.Verify user is able to select the dropdown value or not.	Fruit or Vegetable.	Application should shows user to choose fruit or vegetable option in Dropdown list.	Working as expected	pass	Nil	N	-	P.Karthikeyan
PredictPage_TC_006	Functional	Predict page	Verify user is able to upload the image or not.		1.Enter URL and click go. 2.Click on Predict button 3.Verify whether the user is redirect to predict page or not. 4.Verify user is able to upload the images or not.	Images to be Uploaded	Application should shows the uploaded image.	Working as expected	pass	Nil	N	-	A.Muthusamy
PredictPage_TC_007	Functional	Predict page	Verify whether the image is predicted correctly or not.		1.Enter URL and click go. 2.Click on Predict button 3.Verify whether the user is redirect to predict page or not. 4.Verify user is able to select the dropdown value or not. 5.Verify user is able to upload the images or not. 6. Verify whether the image is predicted correctly or not.	Click the Predict Button	Application shows the predicted output	Working as expected	pass	Nil	N	-	T.R.S.PraveenRaj@Sankaran

8.2 User Acceptance Testing:

Defect Analysis

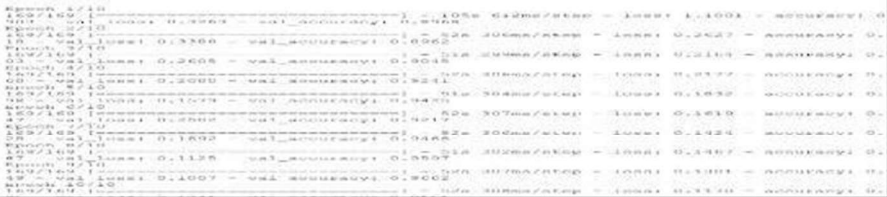
Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
Yellow Leaves	10	4	5	15	34
Blights	1	5	2	4	12
Fruit rots	3	1	0	2	6
Leaf spots	9	2	4	18	33
Mosaic leaf pattern	3	9	6	6	24
Fruit Spots	3	1	5	1	10
Leaves misshapen	0	7	2	1	10
Totals	29	29	24	47	129

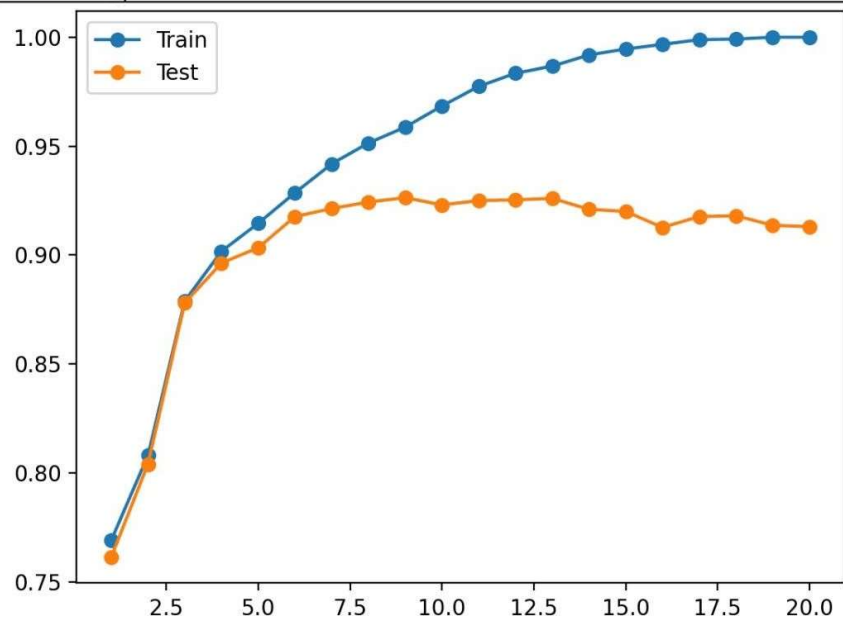
Test Case Analysis

Section	Total Cases	Not Tested	Fail	Pass
Yellow Leaves	20	0	0	20
Blights	43	0	0	43
Fruit rots	9	0	0	9
Leaf spots	5	0	0	5
Mosaic leaf pattern	19	0	0	19
Fruit Spots	2	0	0	2
Leaves misshapen	4	0	0	4

9 - RESULTS

9.1 Performance Metrics :

S.No.	Parameter	Values	Score												
1.	Model Summary	<div>Total Params:896 Trainable Params:896 Non- Trainable Params:0</div>	<div>model.summary()</div> <div>Model: "sequential"</div> <table><thead><tr><th>Layer (type)</th><th>Output Shape</th><th>Param #</th></tr></thead><tbody><tr><td>conv2d (Conv2D)</td><td>(None, 126, 126, 32)</td><td>896</td></tr><tr><td>max_pooling2d (MaxPooling2D)</td><td>(None, 63, 63, 32)</td><td>0</td></tr><tr><td>flatten (Flatten)</td><td>(None, 127008)</td><td>0</td></tr></tbody></table> <div>Total params: 896 Trainable params: 896 Non-trainable params: 0</div>	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
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													
2.	Accuracy	<div>Training Accuracy = 90.3</div> <div>Valuation Accuracy = 89.62</div>													
3.	Confidence Score (Only Yolo Projects)	<div>Class Detected - NA</div> <div>Confidence Score - NA</div>													



10 - ADVANTAGES & DISADVANTAGES

ADVANTAGES :

- Plant Disease can be detected and cure early as possible.
- Easy and Efficient solution to farmers.
- Quicker approach to find solution.
- Automated system with machine learning model.

DISADVANTAGES :

- Can detect only one disease in plants.
- Requires more time during Model Training.
- Performs only on pretrained diseases.

11 - CONCLUSION

Hence, the system can be used to identify Diseases in crops and find solution to it. So, it will reduce the fear and struggle to farmers and can also improve yield to farmers. This system is very efficient as it uses deep learning techniques to identify the diseases and suggest the precautions that can be taken for those diseases.

12 - FUTURE SCOPE

System can be upgraded in future by training the model with new datasets. Model can be trained with data of new diseases to predict the new diseases affects the crops.

Thus, the system is very scalable and reliable. Detection of multiple diseases can be made by using various classification techniques and suggests multiple fertilizers to individual diseases.

13 - APPENDIX

Source Code :

app.py

```
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
from cloudant.client import Cloudant

client = Cloudant.iam('1245c35d-e88f-4b11-82d6-ccee61b098ff-bluemix',
                     '0PYL_N3AdMnURBW1SvQPic81f0-4XdUQdkf0L1eiNA19',
                     connect=True)
my_database = client.create_database('my_database')

secretKey = os.urandom(12).hex()

app = Flask(__name__, template_folder='../templates',
            static_folder='../static')

basepath = os.path.dirname(__file__)
upload_path = os.path.join(os.path.dirname(basepath), 'static', 'uploads')

veg_model = load_model(os.path.join(basepath, "Vegetable.h5"))
fruit_model = load_model(os.path.join(basepath, "Fruit.h5"))

@app.route("/home")
def home():
    return render_template('home.html')
```

```
@app.route("/")
@app.route("/login")
def login():
    return render_template('login.html')

@app.route("/afterLogin", methods=['POST'])
def afterLogin():
    userid = request.form['_id']
    password = request.form['password']

    query = {'_id': {'$eq': userid}}
    docs = my_database.get_query_result(query)

    if(len(docs.all()) == 0):
        return render_template('login.html',msg="User name not found")
    else:
        if((userid == docs[0][0]['_id'] and password == docs[0][0]['password'])):
            return redirect(url_for('home'))
        else:
            return render_template('login.html',msg="Invalid User")

@app.route("/register")
def register():
    return render_template('register.html')

@app.route("/afterRegister", methods=['POST'])
def afterRegister():
    id = request.form['_id']
    name = request.form['name']
    email = request.form['email']
    password = request.form['password']
    confirmPassword = request.form['confirmPassword']
    data = {
        '_id': id,
        'name': name,
        'email': email,
        'password': password,
        'confirmPassword': confirmPassword,
    }

    query = {'_id': {'$eq': data['_id']}}
    docs = my_database.get_query_result(query)
```

```

        if(len(docs.all()) == 0):
            if(password == confirmPassword):
                url = my_database.create_document(data)
                return render_template('login.html')
            else:
                return render_template('register.html',msg="Password not matched")

        else:
            return render_template('login.html',msg="You are already a user. please Login !!!")

@app.route("/predict")
def predict():
    return render_template("predict.html")

@app.route("/prediction", methods=['Get', 'POST'])
def prediction():
    if request.method == 'POST':
        f = request.files['image']
        file_path = os.path.join(upload_path, 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']
        if(plant == "vegetable"):
            pred = veg_model.predict(x)
            df = pd.read_excel(os.path.join(basepath, 'precautions-veg.xlsx'))
            prec = df.iloc[np.argmax(pred),0]
            return render_template("result.html", msg=prec)
        else:
            pred = fruit_model.predict(x)
            df = pd.read_excel(os.path.join(
                basepath, 'precautions-fruit.xlsx'))
            prec = df.iloc[np.argmax(pred),0]
            return render_template("result.html", msg=prec)

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

```

login.html

```
<html>
<title>Plant Disease Prediction</title>
<head>
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet"
  integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"
  integrity="sha384-
OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJua0e923+mo//f6V8Qbsw3"
  crossorigin="anonymous"></script>
  <link rel="stylesheet" href="{{url_for('static',filename='css/login.css')}}"/>
</head>
<body>
  <div class="navbar">
    <h1>Plant Disease Prediction</h1>
    <ul>
      <li><a href="./home">Home</a></li>
      <li><a href="./register">Register</a></li>
    </ul>
  </div>
  <div class="content">
    <div class="inputs">
      {% if msg %}
      <div class="msg">
        <strong>Warning:</strong> {{ msg }}
      </div>
      {% endif %}
      <h2>Please Enter Login Details !!</h2>
      <form action="/afterLogin" method="POST">
        <div class="wrapper">
          </div>
          <div class="wrapper">
            <div class="input">
              <input required type="number" name="_id" />
            </div>
          </div>
        </div>
      </form>
    </div>
  </div>
</body>
</html>
```

```

        <div class="underline"></div>
        <label>Mobile Number</label>
    </div>
</div>
<div class="wrapper">
    <div class="input">
        <input required type="password" name="password" />
        <div class="underline"></div>
        <label>Password</label>
    </div>
</div>
<div class="wrapper">
    <div class="button">
        <input class="success" type="submit" value="Login" />
    </div>
</div>
</form>
</div>

    </div>
</body>

</html>

```

login.css

```

body {
    background-color: aquamarine;
}

img {
    border-radius: 50%;
    width: 700;
    height: 450;
}

.navbar {
    background-color: #393838;
    display: flex;
    justify-content: space-between;
    align-items: center;
}

```

```
.navbar h1 {
  color: white;
  font-size: 30px;
  margin-left: 10%;
}

.navbar ul {
  margin-right: 10%;
}

.navbar ul li {
  display: inline;
  margin: 0px 20px;
}

.navbar ul li a {
  color: white;
  text-decoration: none;
}

.navbar ul li a:hover {
  color: aquamarine;
  transition: 0.2s ease;
}

.content {
  margin-top: 50px;
  display: flex;
}

.inputs {
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  width: 65%;
  margin-top: 20px;
}

.msg{
  background-color: #ffadad;
  margin-bottom: 50px;
}
```



```
padding: 15px;
border-radius: 20px;
}

.wrapper {
  background-color: none;
  padding: 10px;
}

.input {
  height: 50px;
  width: 100%;
  position: relative;
  display: flex;
  justify-content: center;
}

.input input {
  height: 100%;
  width: 100%;
  padding: 20px 10px 10px 20px;
  border: none;
  border-radius: 20px;
  border-bottom: 2px solid silver;
  font-size: 17px;
  outline: none;
}

.underline {
  position: absolute;
  bottom: 0px;
  height: 2px;
  width: 85%;
}

.underline::before {
  position: absolute;
  content: "";
  height: 100%;
  width: 100%;
  background-color: green;
  border-radius: 0px 0px 20px 20px;
  transform: scaleX(0);
  transition: transform 0.3s ease;
```

```
}

.wrapper .input label {
  position: absolute;
  bottom: 10px;
  left: 10px;
  color: gray;
  pointer-events: none;
  transition: all 0.3s ease;
}

.wrapper .input input:focus~label,
.wrapper .input input:valid~label {
  transform: translateY(-25px);
  font-size: 12px;
  color: green;
}

.wrapper .input input:focus~.underline:before,
.wrapper .input input:valid~.underline:before {
  transform: scaleX(1);
}

.button{
  display: flex;
  justify-content: center;
}

.button input{
  background-color: coral;
  border: none;
  border-radius: 20px;
  height: 40px;
  width: 80px;
  color: white;
  font-size: 16px;
}
```

register.html

```
<html>
<title>Plant Disease Prediction</title>

<head>
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet"
    integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"
"
    integrity="sha384-
OERcA2EqJJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3"
    crossorigin="anonymous"></script>
  <link rel="stylesheet"
href="{{url_for('static',filename='css/register.css')}}"/>
</head>

<body>
  <div class="navbar">
    <h1>Plant Disease Prediction</h1>
    <ul>
      <li><a href="./home">Home</a></li>
      <li><a href="./login">Login</a></li>
    </ul>
  </div>
  <div class="content">
    <div class="inputs">
      {% if msg %}
      <div class="msg">
        <strong>Warning:</strong> {{ msg }}
      </div>
      {% endif %}
      <h2>Please Fill the following Details !!</h2>
      <form action="/afterRegister" method="POST">
        <div class="wrapper">
          <div class="input">
            <input required type="text" name="name" />
            <div class="underline"></div>
            <label>Name</label>
          </div>
```

```

        </div>
        <div class="wrapper">
            <div class="input">
                <input required type="number" name="_id" />
                <div class="underline"></div>
                <label>Mobile No</label>
            </div>
        </div>
        <div class="wrapper">
            <div class="input">
                <input required type="email" name="email" />
                <div class="underline"></div>
                <label>E-mail</label>
            </div>
        </div>
        <div class="wrapper">
            <div class="input">
                <input required type="password" name="password" />
                <div class="underline"></div>
                <label>Password</label>
            </div>
        </div>
        <div class="wrapper">
            <div class="input">
                <input required type="password" name="confirmPassword" />
                <div class="underline"></div>
                <label>Confirm Password</label>
            </div>
        </div>
        <div class="wrapper">
            <div class="button">
                <input type="submit" value="Register" />
            </div>
        </div>
    </form>
</div>


</div>
</body>

</html>

```

register.css

```
body {
  background-color: aquamarine;
}

img {
  border-radius: 50%;
  width: 700;
  height: 450;
}

.navbar {
  background-color: #393838;
  display: flex;
  justify-content: space-between;
  align-items: center;
}

.navbar h1 {
  color: white;
  font-size: 30px;
  margin-left: 10%;
}

.navbar ul {
  margin-right: 10%;
}

.navbar ul li {
  display: inline;
  margin: 0px 20px;
}

.navbar ul li a {
  color: white;
  text-decoration: none;
}

.navbar ul li a:hover {
  color: aquamarine;
  transition: 0.2s ease;
}
```

```
.content {
  margin-top: 50px;
  display: flex;
}

.inputs {
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  width: 65%;
  margin-top: 20px;
}

.msg{
  background-color: #ffadad;
  margin-bottom: 50px;
  padding: 15px;
  border-radius: 20px;
}

.wrapper {
  background-color: none;
  padding: 10px;
}

.input {
  height: 50px;
  width: 100%;
  position: relative;
  display: flex;
  justify-content: center;
}
```

```
.input input {
  height: 100%;
  width: 100%;
  padding: 20px 10px 10px 20px;
  border: none;
  border-radius: 20px;
  border-bottom: 2px solid silver;
  font-size: 17px;
  outline: none;
}

.underline {
  position: absolute;
  bottom: 0px;
  height: 2px;
  width: 85%;
}

.underline::before {
  position: absolute;
  content: "";
  height: 100%;
  width: 100%;
  background-color: green;
  border-radius: 0px 0px 20px 20px;
  transform: scaleX(0);
  transition: transform 0.3s ease;
}

.wrapper .input label {
  position: absolute;
  bottom: 10px;
  left: 10px;
  color: gray;
  pointer-events: none;
  transition: all 0.3s ease;
}
```

```
.wrapper .input input:focus~label,  
.wrapper .input input:valid~label {  
    transform: translateY(-25px);  
    font-size: 12px;  
    color: green;  
}  
  
.wrapper .input input:focus~.underline:before,  
.wrapper .input input:valid~.underline:before {  
    transform: scaleX(1);  
}  
  
.button{  
    display: flex;  
    justify-content: center;  
}  
  
.button input{  
    background-color: coral;  
    border: none;  
    border-radius: 20px;  
    height: 40px;  
    width: 80px;  
    color: white;  
    font-size: 16px;  
}  
  
.msg{  
    display: flex;  
    justify-content: center;  
    background-color: gray;  
}
```


home.html

```
<html>
<title>Plant Disease Prediction</title>

<head>
  <link rel="stylesheet" href="{{url_for('static',filename='css/home.css')}}"/>
</head>

<body>
  <div class="navbar">
    <h1>Plant Disease Prediction</h1>
    <ul>
      <li><a href="/predict">Predict</a></li>
      <li><a href="/login">Log Out</a></li>
    </ul>
  </div>
  <div class="content">
    <div class="description">
      <h1>Detect if your plant is infected!!</h1>
      <p>
        <span>Agriculture</span> is one of the major sectors world
        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 in turn saves effort and
        labour costs.
      </p>
      <div class="button">
        <a href="/predict">Predict Image</a>
      </div>
    </div>
    
</body>

</html>
```

home.css

```
body {
  background-color: aquamarine;
}

img{
  border-radius: 50%;
  width: 700;
  height: 450;
}

.navbar{
  background-color: #393838;
  display: flex;
  justify-content: space-between;
  align-items: center;
}

.navbar h1{
  color: white;
  font-size: 30px;
  margin-left: 10%;
}

.navbar ul{
  margin-right: 10%;
}

.navbar ul li{
  display: inline;
  margin: 0px 20px;
}

.navbar ul li a{
  color: white;
  text-decoration: none;
}

.navbar ul li a:hover{
  color: aquamarine;
  transition: 0.2s ease;
}
```

```
.content{
  margin-top: 50px;
  display: flex;
}

.content .name{
  text-align: left;
  margin-left: 20px;
}

h1{
  font-size: 40px;
  text-align: center;
}

p{
  padding: 25px;
  font-size: 20px;
}

p span{
  margin-left: 20px;
}

.button{
  display: flex;
  justify-content: center;
  align-items: center;
  margin-top: 50px;
}

.button a{
  text-decoration: none;
  color: white;
  height: 30px;
  width: 100px;
  background-color: coral;
  text-align: center;
  padding-top: 15px;
  border: 1px solid black;
  border-radius: 10px;
}
```

predict.html

```
<html>
<title>Plant Disease Prediction</title>
<head>
  <script src="https://code.jquery.com/jquery-1.12.4.js"></script>
  <link rel="stylesheet"
href="{{url_for('static',filename='css/predict.css')}}" />
</head>
<body>
  <div class="navbar">
    <h1>Plant Disease Prediction</h1>
    <ul>
      <li><a href="./home">Home</a></li>
      <li><a href="./login">Log out</a></li>
    </ul>
  </div>
  <div class="content">
    <div class="prediction">
      <div class="form">
        <h1>Drop the Images to Get the Prediction</h1>
        <form action="/prediction" method="POST" enctype="multipart/form-
data">

          <div class="form-row">
            <label for="plant">Select the Type of Plant :</label>
            <select name="plant">
              <option value="fruit">Fruit</option>
              <option value="vegetable">Vegetable</option>
            </select>
          </div>
          <div class="form-row">
            <label for="plant">Select the Image of Plant :</label>
            <input id="input" type="file" name="image" />
          </div>
          <div class="button-row">
            <input id="button" type="submit" value="Predict" />
          </div>
        </form>
      </div>
      <div>
        
      </div>
    </div>
  </body>
</html>
```

predict.css

```
body {
  background-color: aquamarine;
}

img{
  border-radius: 50%;
  width: 700;
  height: 450;
}

.navbar{
  background-color: #393838;
  display: flex;
  justify-content: space-between;
  align-items: center;
}

.navbar h1{
  color: white;
  font-size: 30px;
  margin-left: 10%;
}

.navbar ul{
  margin-right: 10%;
}

.navbar ul li{
  display: inline;
  margin: 0px 20px;
}
```

```
.navbar ul li a{
  color: white;
  text-decoration: none;
}

.navbar ul li a:hover{
  color: aquamarine;
  transition: 0.2s ease;
}

.content{
  margin-top: 30px;
  display: flex;
}

.prediction{
  width: 65%;
  display: flex;
  justify-content: center;
  align-items: center;
}

.prediction .form{
  display: flex;
  flex-direction: column;
  justify-content: space-between;
}

.prediction .form form{
  display: flex;
  flex-direction: column;
  justify-content: space-between;
  margin-top: 30px;
}
```

```
.prediction .form form .form-row label{
  font-size: 20px;
  margin-right: 20px;
}

.prediction .form form .form-row select{
  width: 100px;
  height: 30px;
  font-size: 15px;
  background-color: bisque;
  border-radius: 10px;
  margin-left: 10px;
  margin-bottom: 40px;
}

.prediction .form form .form-row #input{
  margin-bottom: 40px;
}

.prediction .form form .button-row{
  display: flex;
  justify-content: center;
  margin-top: 30px;
}

.prediction .form form .button-row #button{
  width: 100px;
  height: 30px;
  font-size: 15px;
  border-radius: 10px;
  border: none;
  background-color: coral;
}
```

result.html

```
<html>

<title>Plant Disease Prediction</title>

<head>

    <link rel="stylesheet" href="{{url_for('static',filename='css/result.css')}}"
/>

</head>

<body>

    <div class="navbar">
        <h1>Plant Disease Prediction</h1>
        <ul>
            <li><a href="./home">Home</a></li>
            <li><a href="./predict">Predict</a></li>
        </ul>
    </div>

    <div class="content">
        <div class="description">
            <h1>Prediction :</h1>
            <p>
                <span></span>{{msg}}
            </p>
        </div>

        
    </div>

</body>

</html>
```


result.css

```
body {  
  background-color: aquamarine;  
}  
  
img{  
  border-radius: 50%;  
  width: 700;  
  height: 450;  
}  
  
.navbar{  
  background-color: #393838;  
  display: flex;  
  justify-content: space-between;  
  align-items: center;  
}  
  
.navbar h1{  
  color: white;  
  font-size: 30px;  
  margin-left: 10%;  
}  
  
.navbar ul{  
  margin-right: 10%;  
}  
  
.navbar ul li{  
  display: inline;  
  margin: 0px 20px;  
}
```

```
.navbar ul li a{  
  color: white;  
  text-decoration: none;  
}
```

```
.navbar ul li a:hover{  
  color: aquamarine;  
  transition: 0.2s ease;  
}
```

```
.content{  
  margin-top: 50px;  
  display: flex;  
}
```

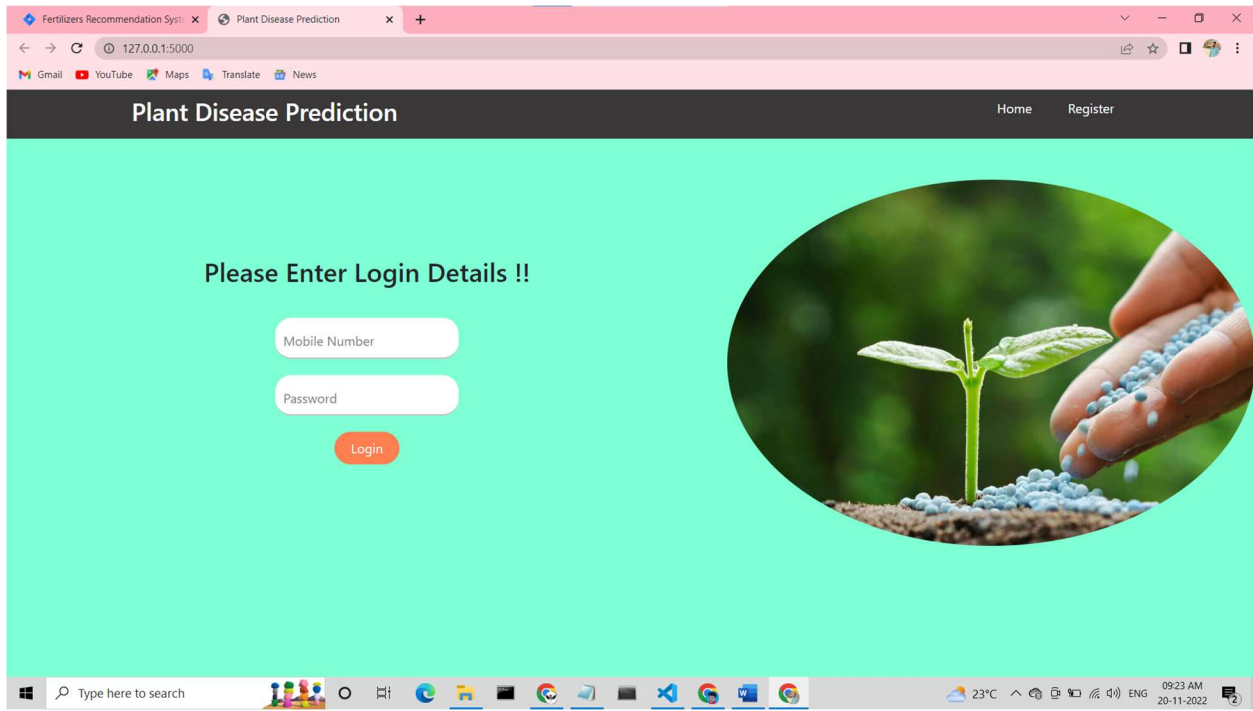
```
h1{  
  font-size: 40px;  
  text-align: center;  
}
```

```
p{  
  padding: 25px;  
  font-size: 20px;  
}
```

```
p span{  
  margin-left: 20px;  
}
```

Screenshots :

Login Page :



The screenshot shows a web browser window with two tabs: 'Fertilizers Recommendation Sys...' and 'Plant Disease Prediction'. The address bar shows '127.0.0.1:5000'. The page title is 'Plant Disease Prediction'. The navigation bar has 'Home' and 'Register' links. The main content area has a light blue background and a large circular image of a hand sowing seeds. The text 'Please Enter Login Details !!' is centered. Below it are two input fields: 'Mobile Number' and 'Password'. A red 'Login' button is at the bottom of the form.

Plant Disease Prediction

Home Register

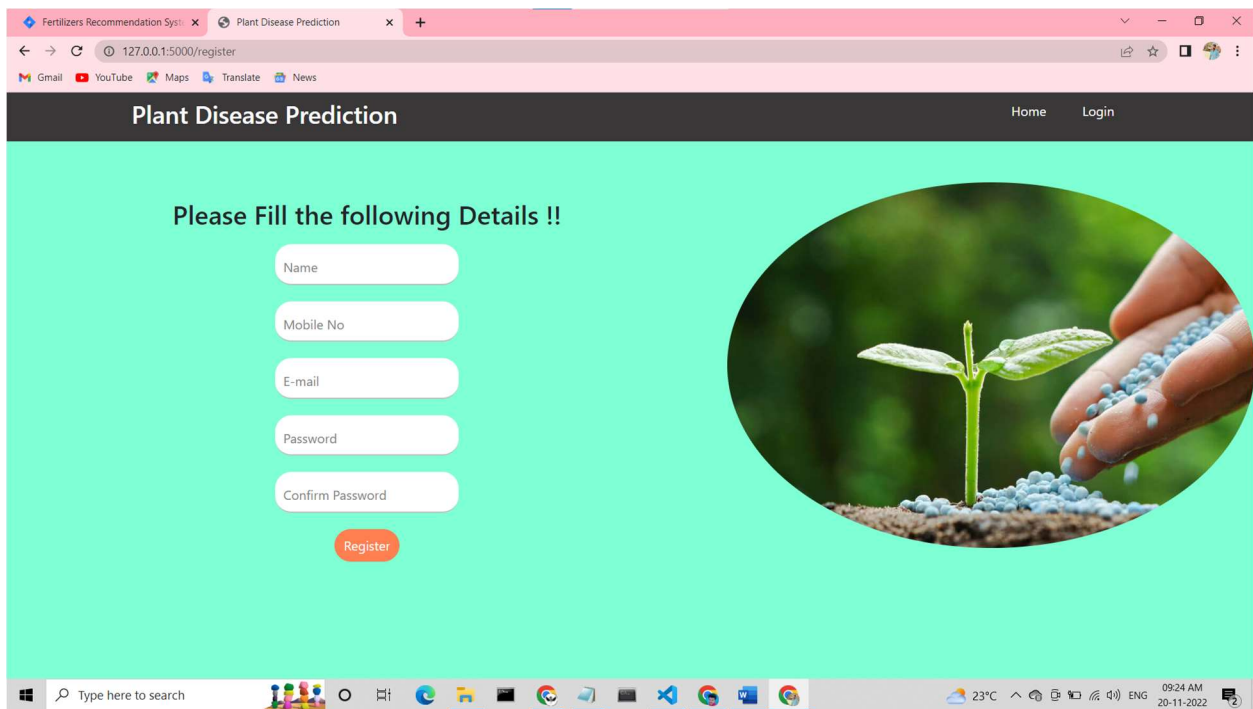
Please Enter Login Details !!

Mobile Number

Password

Login

Register Page :



The screenshot shows a web browser window with two tabs: 'Fertilizers Recommendation Sys...' and 'Plant Disease Prediction'. The address bar shows '127.0.0.1:5000/register'. The page title is 'Plant Disease Prediction'. The navigation bar has 'Home' and 'Login' links. The main content area has a light blue background and a large circular image of a hand sowing seeds. The text 'Please Fill the following Details !!' is centered. Below it are five input fields: 'Name', 'Mobile No', 'E-mail', 'Password', and 'Confirm Password'. A red 'Register' button is at the bottom of the form.

Plant Disease Prediction

Home Login

Please Fill the following Details !!

Name

Mobile No

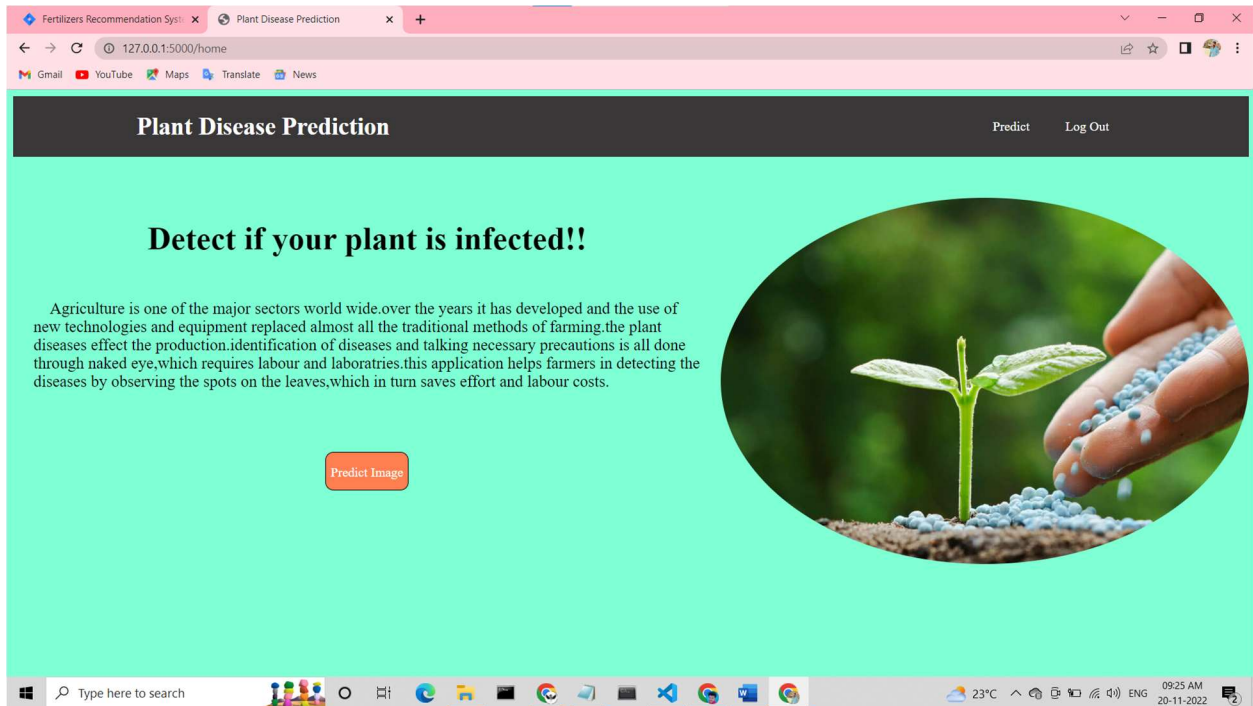
E-mail

Password

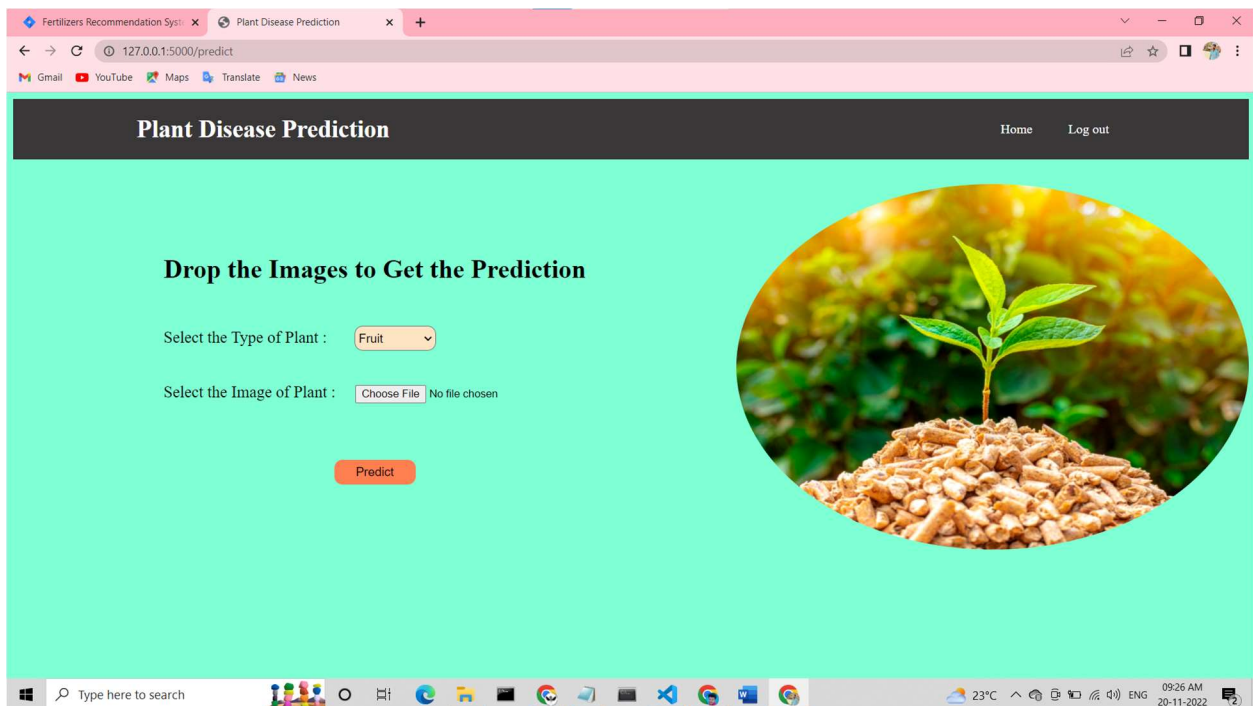
Confirm Password

Register

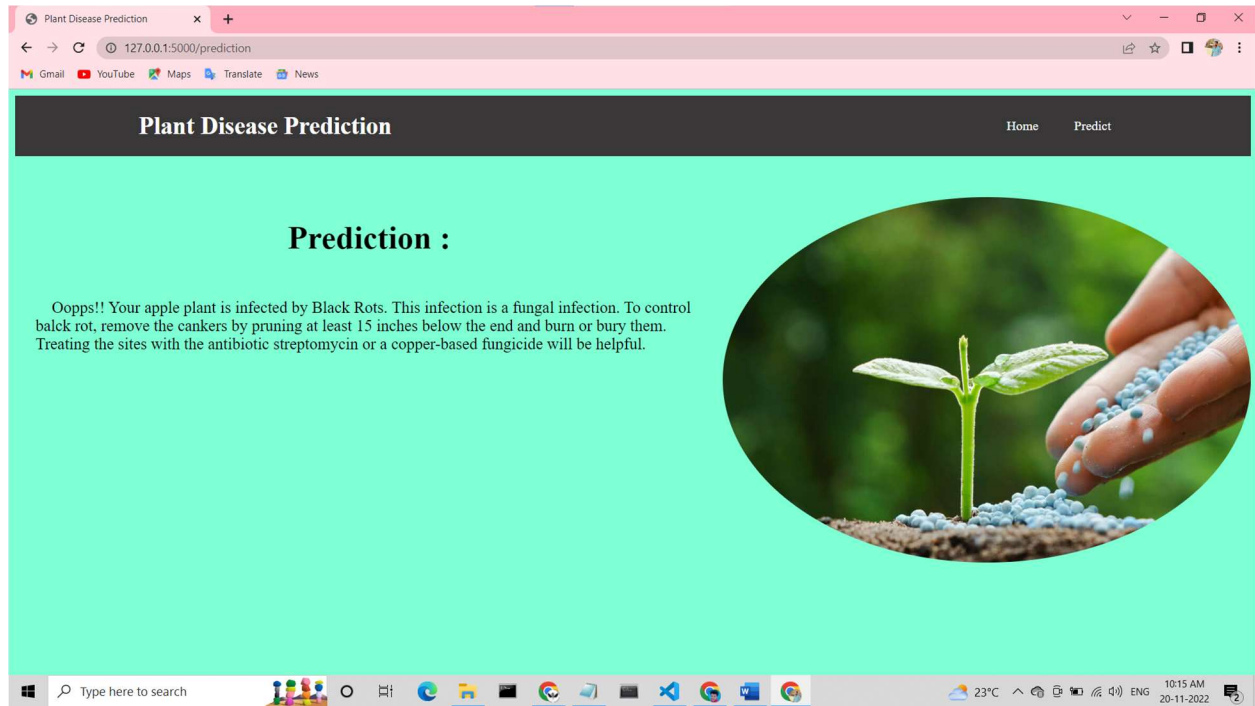
Home Page :



Predict Page :



Result Page :



GitHub & Project Demo Link:

- Github :

<https://github.com/IBM-EPBL/IBM-Project-49315-1660817823>

- Project Demo Video :

<https://youtu.be/yxFV2nlr9Nk>