PROJECT REPORT

Fertilizers Recommendation System

For Disease Prediction

Team ID : PNT2022TMID26196

TEAM MEMBERS:

- **KAMALESHWARAN S (TEAM LEAD) [211519205068]**
- **JAGADEESWARAN K [211519205062]**
- **MANOJ M [211519205094]**
- **AKILAN D [211519205010]**

CONTENTS

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.

1.2 Purpose:

- To Detect and recognize the plant diseases and to recommend fertilizer, it is necessary to provide symptoms in identifying the disease at its earliest. Hence the authors proposed and implemented new fertilizers Recommendation System for crop disease prediction.

2. LITERATURE SURVEY:

2.1 Existing Problem:

- Adequate mineral nutrition is central to crop production. However, it can also exert considerable Influence on disease development. Fertilizer application can increase or decrease development of diseases caused by different pathogens, and the mechanisms responsible are complex, including effects of nutrients on plant growth, plant resistance mechanisms and direct effects on the pathogen. The effects of mineral nutrition on plant disease and the mechanisms responsible for those effects have been dealt with comprehensively elsewhere. In India, around 40% of land is kept and grown using reliable irrigation technologies, while the rest relies on the monsoon environment for water. Irrigation decreases reliance on the monsoon, increases food security, and boosts agricultural production.

- Most research articles use humidity, moisture, and temperature sensors near the plant's root, with an external device handling all of the data provided by the sensors and transmitting it directly to an external display or an Android application. The application was created to measure the approximate values of temperature, humidity and moisture sensors that were programmed into a microcontroller to manage the amount of water.

2.2 References:

- Soil health analysis and fertilizer prediction for crop image identification by Inception-V3 and random forest - ScienceDirect
- http:/ www.ijetajournal.org/volume-8/issue-2/IJETA-V8I2P1.pdf
- https:/ ieeexplore.ieee.org/document/8878781
- https:/ www.irjet.net/archives/V7/i10/IRJET-V7I1004.pdf
- https:/ www.sciencedirect.com/science/article/pii/S0168169921004245
- https:/ www.ijstr.org/final-print/nov2019/Fertilizers-Recommendation-System-For-Disease-Prediction-In-Tree-Leave.pdf
- https:/ arxiv.org/pdf/2204.11340.pdf

2.3 Problem Statement Definition:

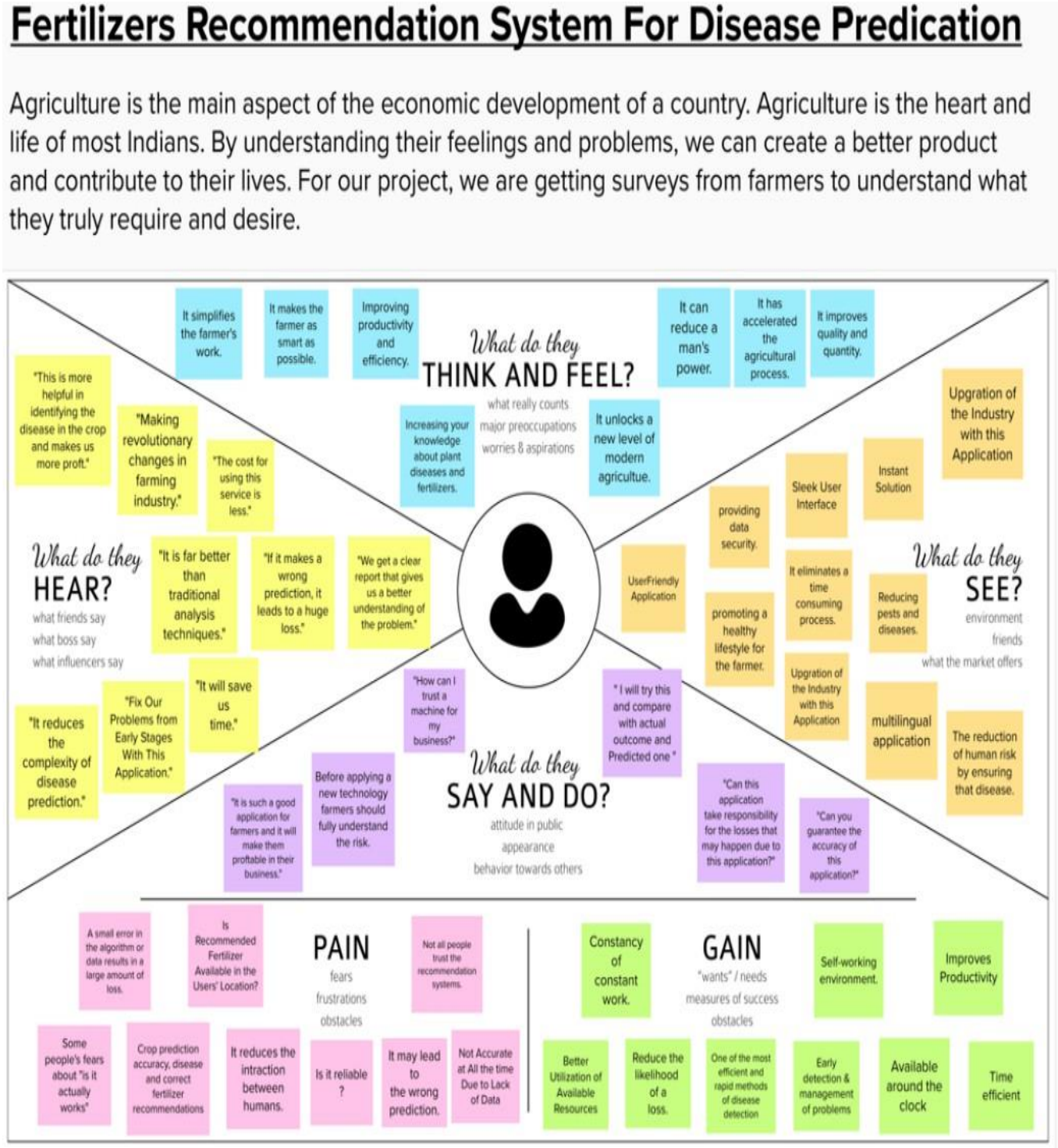
Mr. Surender is a 55-year-old man. He had an own farming land and do Agriculture for past 30 Years. In this 30 Years he had faced a problem in choosing fertilizers and controlling of plant disease. Surender wants to know the better recommendation for fertilizers for plants with the disease. He has faced huge losses for a long time. This problem is usually faced by most farmers. He needs to know the result immediately.

Mr. Surender is a 55-year-old man. He had an own farming land and do Agriculture for past 30 Years. In these 30 years he had facing a problem of plant diseases.	Surender wants to predict the plant disease and according to that he will buy the fertilizers. By this he will prevent his crops from the plant disease.	The presence of plant diseases on an agricultural farm cost Surender a lot of money. The traditional method of physically analysing particular aspects of leaves.	Increasing agricultural production, is a leading cause of nitrous oxide emissions from agriculture, contributing significantly to global warming.	Surender needs to deal with many problems, including how to: Cope with climate change, soil erosion and biodiversity loss. Meet rising demand for more food of higher quality.

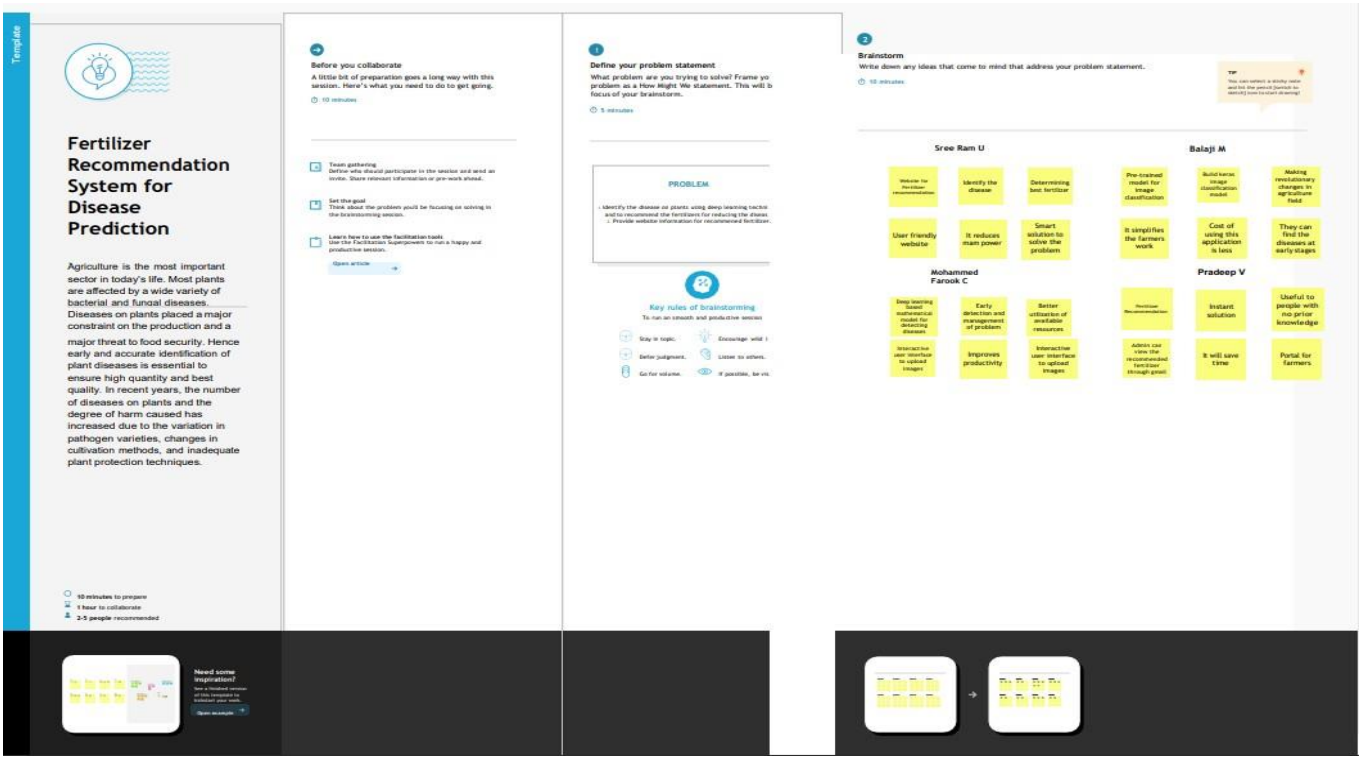
Who does the problem affect?	Persons who do Agriculture.
What are the boundaries of the problem?	People who Grow Crops and facing Issues of Plant Disease.
What is the issue?	In Agricultural aspects, if the plant is affected by leaf disease, then it reduces the growth and productiveness. Generally, the plant diseases are caused by the abnormal physiological functionalities of plants.
When does the issue occur?	During the development of the crops as they will be affected by various diseases.
Where does the issue occur?	The issue occurs in agriculture practicing areas, particularly in rural regions.
Why is it important that we fix the problem?	It is required for the growth of better-quality food products. It is important to maximise the crop yield.
What solution to solve this issue?	An Automated system is introduced to identify different diseases on plants by checking the symptoms shown on the leaves of the plant.
What methodology used to solve the issue?	Deep learning techniques are used to identify the diseases and suggest the precautions that can be taken for those diseases.

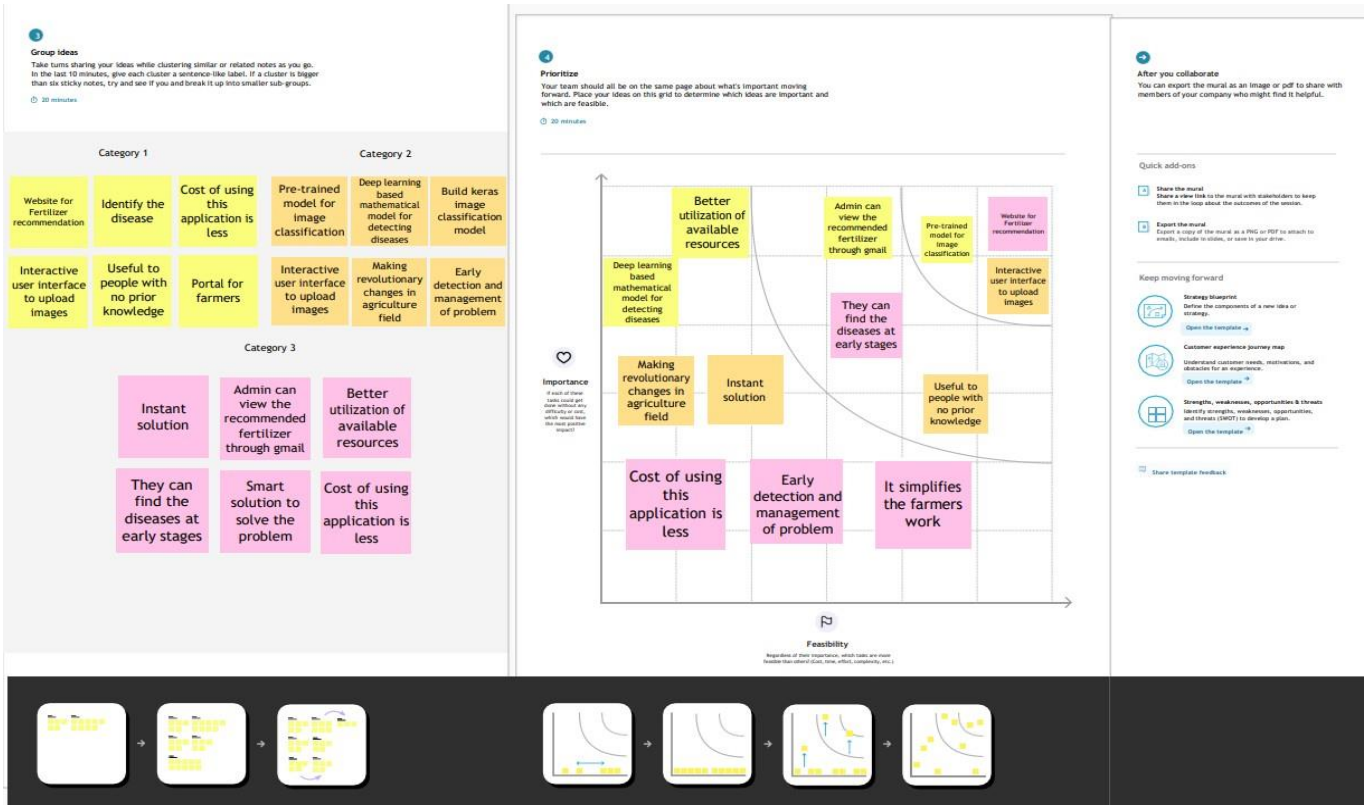
3. IDEATION & PROPOSED SOLUTION:

3.1 Empathy Map Canvas



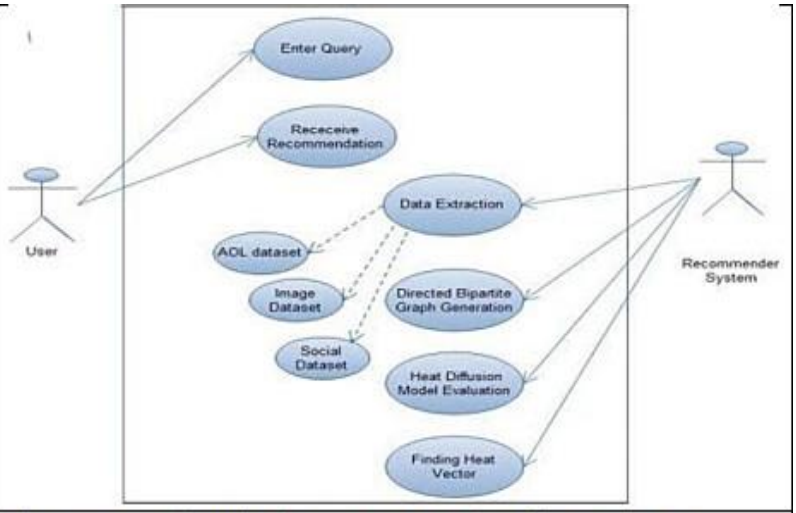
3.2 Ideation & Brainstorming:





3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Mostly, the plant leaf diseases are caused by Pathogens which are positioned on the stems of the plants. These different symptoms and diseases of leaves are predicted by different methods in image processing. These different methods include different fundamental processes like segmentation, feature extraction and classification and so on. Mostly, the prediction and diagnosis of leaf diseases are depending on the segmentation such as segmenting the healthy tissues from diseased tissues of leaves. Detection and recognition of plant diseases using machine learning are very efficient in providing symptoms of identifying diseases at its earliest.
2.	Idea / Solution description	Leaves are affected by bacteria, fungi, virus, and other insects. Support Vector Machine (SVM) algorithm classifies the leaf image as normal or affected. Vectors are constructed based on leaf features such as colour, shape, textures. Then hyperplane constructed with conditions to categorize the pre-processed leaves and also implement multiclass classifier, to predict diseases in leaf image with improved accuracy.
3.	Novelty / Uniqueness	Recommends the fertilizer for affected leaves based on severity level. Fertilizers may be organic or inorganic. Admin can store the fertilizers based on disease categorization with severity levels. The measurements of fertilizers are also suggested based on disease severity.
4.	Social Impact / Customer Satisfaction	Presently our farmers are not effectively using technology and analysis, so there may be a chance of wrong selection of fertilizer for crops that will reduce their income. To reduce those type of loses we have developed a farmer friendly system with GUI, that will predict which would be the best suitable fertilizer for particular crop disease. So, this makes the farmers to take right decision in selecting the fertilizer for crop disease such that agricultural sector will be developed by innovative idea.

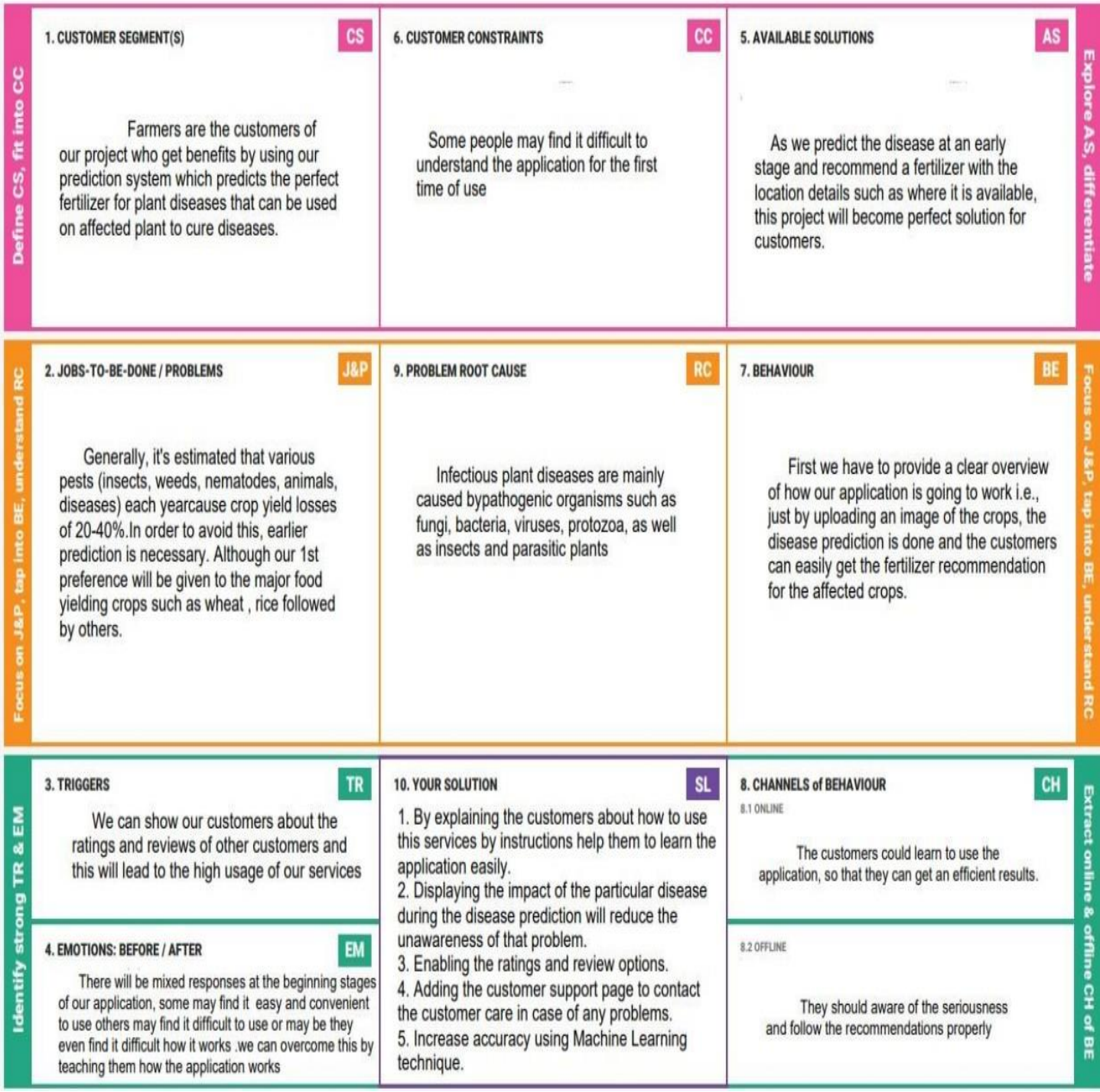
5.	Business Model (Revenue Model)	
6.	Scalability of the Solution	The proposed method uses SVM to classify tree leaves, identify the disease and suggest the fertilizer. The proposed method is compared with the existing CNN based leaf disease prediction. The proposed SVM technique gives a better result when compared to existing CNN. For the same set of images, F-Measure for CNN is 0.7and 0.8 for SVM, the accuracy of identification of leaf disease of CNN is 0.6 and SVM is 0.8.

So, we have built Web Application where:

- Farmers can easily interact with the portal that we have build.
- Interacts with the user interface to upload images of diseased leaf.
- Our model is analyses the disease and suggests the farmers with fertilizers are to beused in the crops or plants.
- Detection and recognition of plant diseases using machine learning are very efficient inproviding symptoms of identifying diseases at its earliest.
- It recommends the fertilizer for affected leaves based on severity level.
- This web application makes the farmers to take right decision in selecting the fertilizerfor crop disease such that agricultural sector will be developed by innovative idea.

3.4 Problem Solution fit:

Team ID : PNT2022TMID28620



4. REQUIREMENT ANALYSIS:

4.1 Functional requirement

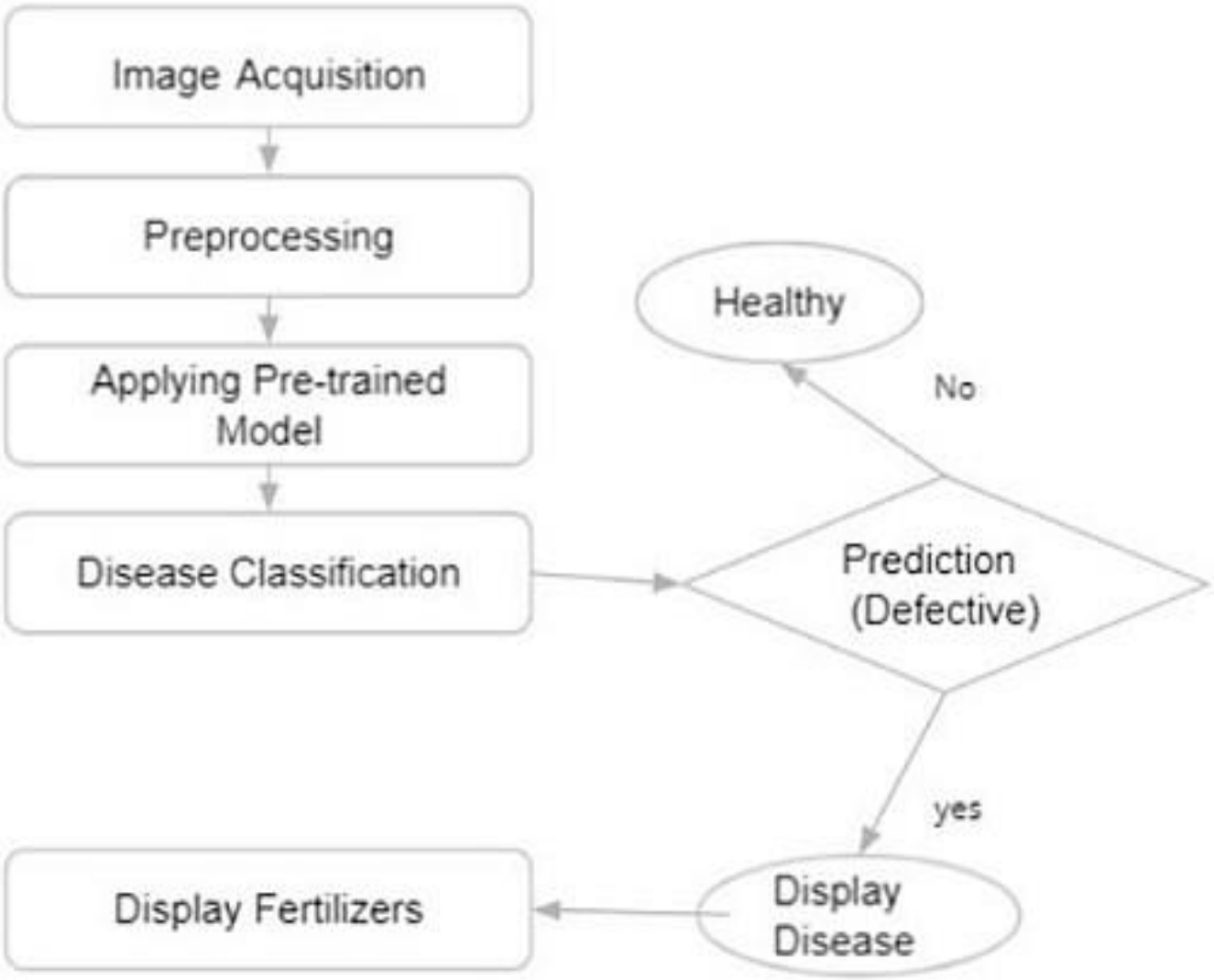
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Login	Login with user name Login with password
FR-4	Profile update	Update the user credentials Update the Contact details
FR-5	Uploading Images	Capture the image of the affected Crop Upload the image of the affected Crop This model will predict the disease of the affected Crop
FR-6	Recommendation	User will request the fertilizer Get the fertilizer recommendations
FR-7	Ratings and Reviews	Share their Experiences Give the Feedback

4.2 Non-Functional requirements

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	This service is designed and can be used on both website and mobile browsers so that the usability of this application is very efficient.
NFR-2	**Security**	This can be used only by users who have their proper login credentials
NFR-3	**Reliability**	In case of any issues such as the delay in the responses, it will be rectified to maintain its reliability.
NFR-4	**Performance**	Sometimes the wrong predictions occur due to the inaccuracy of the model at a rare point, in order to rectify this, this application will run the model more than one time to predict the exact result and recommends the fertilizer for that disease.
NFR-5	**Availability**	It will predict any type of new disease by learning from the available dataset and predict the disease accurately.
NFR-6	**Scalability**	It can be accessed by more number of users at the same time without any performance issues.

5. PROJECT DESIGN:

5.1 Data Flow Diagrams:



5.2 Solution & Technical Architecture:

1. DATA COLLECTION:

Data collection is the most efficient method for collecting and measure the data from different resources like kaggle and UCI machine learning repository. To get an approximate dataset for the system. This dataset must contain the following attributes i.) Images of fruit diseases ii.) Images of vegetable diseases etc., in which those parameters will consider for disease prediction.

2. DATA PRE-PROCESSING:

After collecting datasets from various resources. Dataset must be pre-processing before training to the model. The data pre-processing can be done by various stages, begins with reading the collected dataset the process continues to data cleaning. In data cleaning the datasets contain some redundant attributes, those attributes are not considering for disease prediction. So, we have to drop unwanted attributes and datasets containing some missing values we need to drop these missing values or fill with unwanted values in order to get better accuracy. Then define the target for a model. After data cleaning the dataset will be split into training and test set by using specific libraries.

3. DEEP LEARNING ALGORITHM FOR PREDICTION:

Machine learning predictive algorithms has highly optimized estimation has to be likely outcome based on trained data. Predictive analytics is the use of data, statistical algorithms and machine learning techniques to identify the likelihood of future outcomes based on historical data.

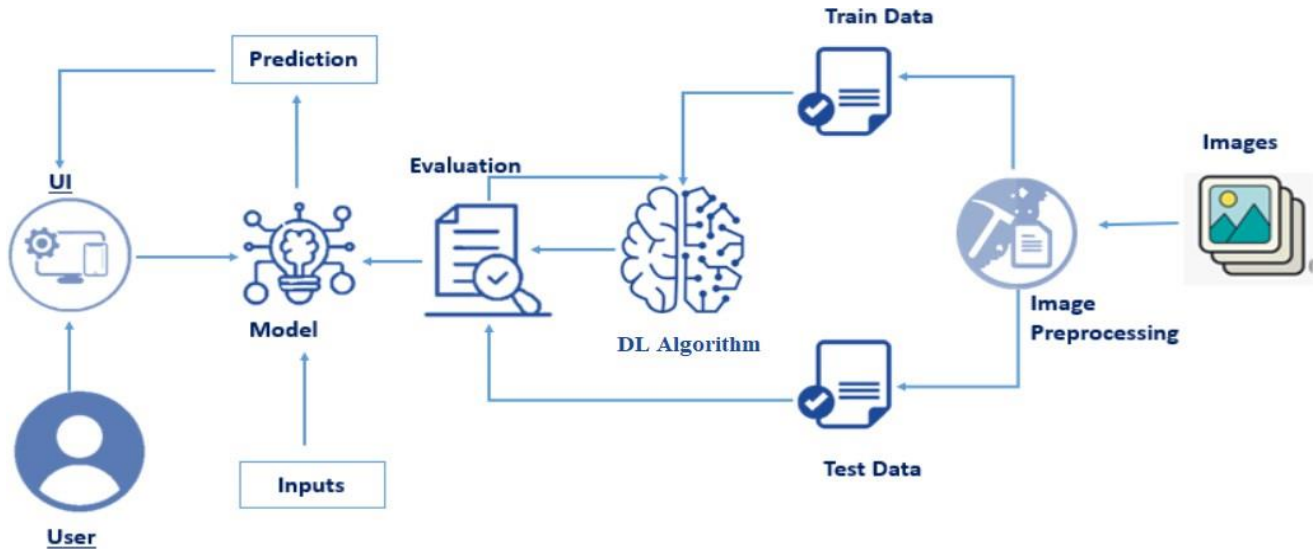
4. FERTILIZER RECOMMENDATION:

To Predict the particular fertilizer to be used, we use input parameters like N,P,K temperature, humidity, moisture and soil type and also crop to be grown. Fertilizer prediction process being with the loading the external fertilizers datasets. Once the dataset read then pre-processing will be done by various stages as discussed in Data Pre-processing section. After the data pre-processing, train the models using SVM, Random Forest classifier into training dataset. For a prediction of the fertilizers, we consider a various factor such as temperature, humidity, soil PH and predicted crop to be grown. Those are the input parameter for a system that can be entered by manually or taken from the sensors. Predicted crop and input parameter values will be appended in a list

RESULT:

The proposed system recommends the best suitable fertilizer for particular land by considering parameters such as various historical image data's. Among these parameters the crop disease is predicted by system itself by using previous data with SVM algorithm and other parameters are have to be entered by the user. In the output section the system displays a suitable fertilizer and various suggestions based on the severity level of the diseases.

Technical Architecture:



5.3 User Stories:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-1	Image Processing.	USN-1	As a user, I can retrieve useful information about the images.	1	Low
Sprint-2	Model Building for Fruit Disease Prediction.	USN-2	As a user, I can able to predict fruit disease using this model.	1	Medium
Sprint-2	Model Building for Vegetable Disease Prediction.	USN-3	As a user, I can able to predict vegetable disease using this model.	2	Medium
Sprint-3	Application Building.	USN-4	As a user, I can see a web page for Fertilizers Recommendation System for Disease Prediction	2	High
Sprint-4	Train The Model on IBM Cloud.	USN-5	As a user, I can save the information about Fertilizers and crops on IBM cloud	2	High

6. PROJECT PLANNING & SCHEDULING:

6.1 Sprint Planning & Estimation:

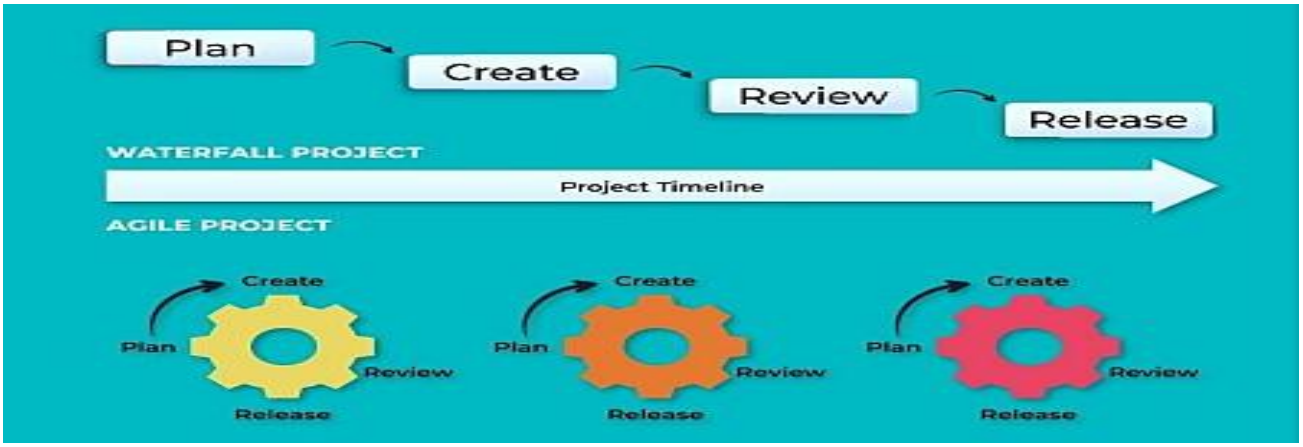
Sprint Planning:

- The performance of Artificial Intelligence (AI) models is being improved and increased in modern technology.
- Based Crop Yield Disease Prediction System would assist farmers in protecting their crops from a variety of diseases by identifying them during the process of taking an image at the plant and providing the afflicted disease's name to a machine learning algorithm.
- The best answer for the farmer will be provided in this project milestone, and he or she may find it on their own by using a web application with a completely user-friendly and straightforward user interface.
- Additionally, we intend to add a useful Module that is a fertilizer prescription for a certain disease to the process. It can propose both artificial and natural fertilizer in a similarway.

Estimation:

1. Planning is a crucial role in project management because it allows teammembersto schedule their time on the project.
2. This activity demonstrates how the team members assigned and completedvarious tasks!
3. In Project we can Split into the Four Step of Phrases are,

- Phrase 1: Information Collection and Requirement Analysis
- Phrase 2: Project Planning and Developing Modules
- Phrase 3: Implementing the High Accuracy Deep Learning Algorithm to Perform
- Phrase 4:Deploying the Model on Cloud and Testing the Model and UIPerformance



6.2 Sprint Delivery Schedule:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Image Processing.	USN-1	As a user, I can retrieve useful information about the images.	1	Low	Sree Ram U, Pradeep V
Sprint-2	Model Building for Fruit Disease Prediction.	USN-2	As a user, I can able to predict fruit disease using this model.	1	Medium	Sree Ram U, Mohammed Farook C
Sprint-2	Model Building for Vegetable Disease Prediction.	USN-3	As a user, I can able to predict vegetable disease using this model.	2	Medium	Sree Ram U, Balaji M
Sprint-3	Application Building.	USN-4	As a user, I can see a web page for Fertilizers Recommendation System for Disease Prediction	2	High	Sree Ram U, Mohammed Farook C, Balaji M, Pradeep V.
Sprint-4	Train The Model on IBM Cloud.	USN-5	As a user, I can save the information about Fertilizers and crops on IBM cloud	2	High	Sree Ram U, Mohammed Farook C.

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	26 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	30 Oct 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	05 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	10 Nov 2022

Velocity:

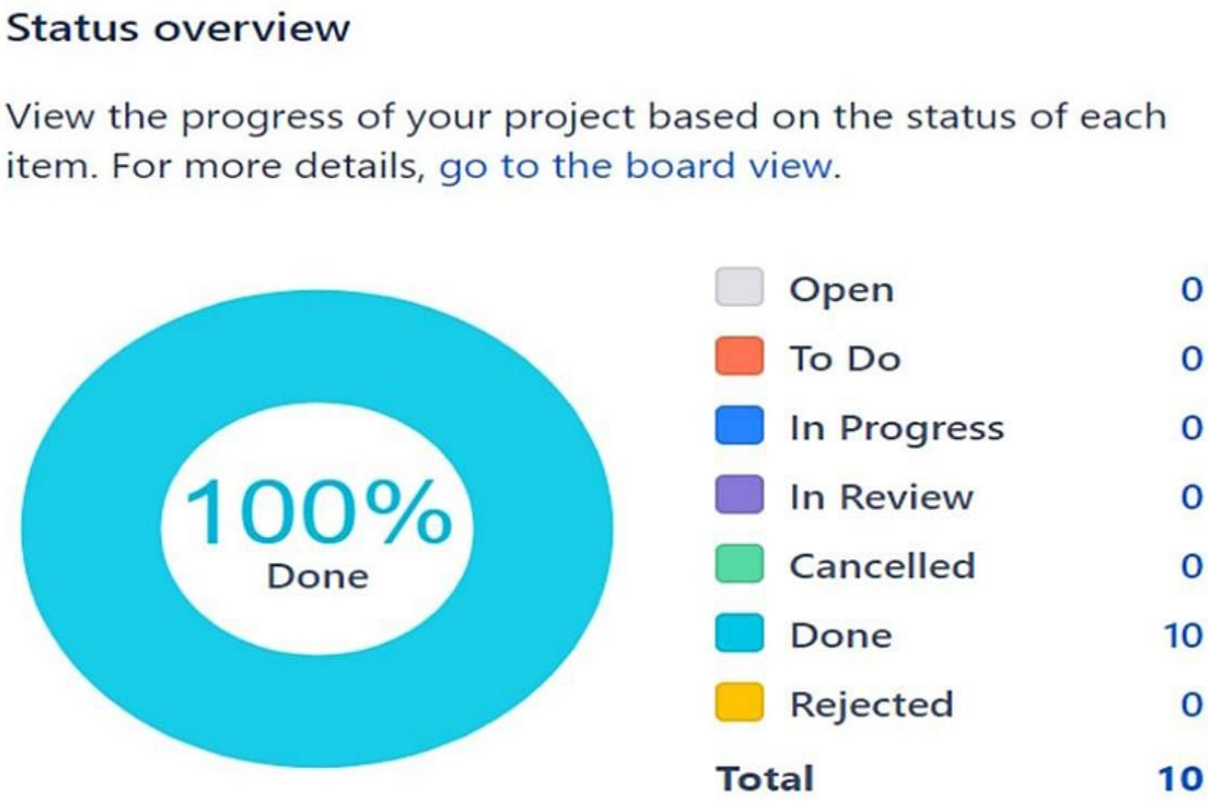
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

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

Average Velocity (AV) = 20/5 = 4

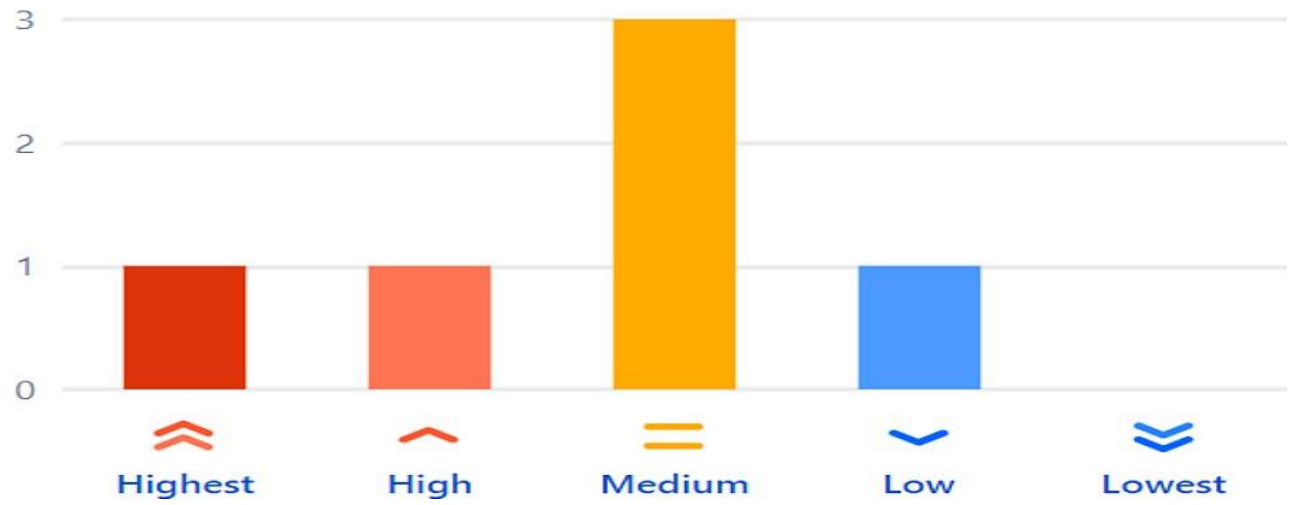
Burndown Chart:

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.

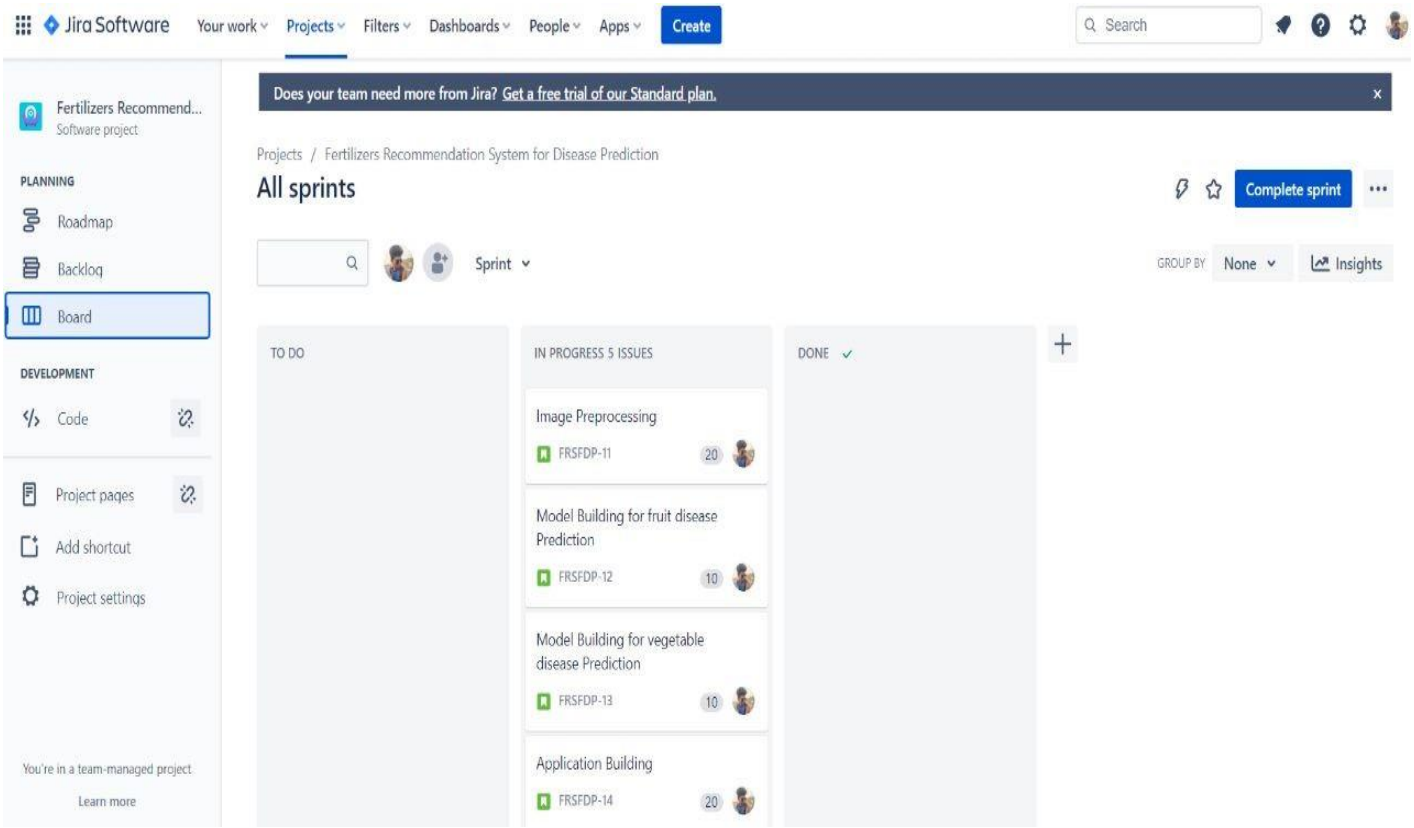


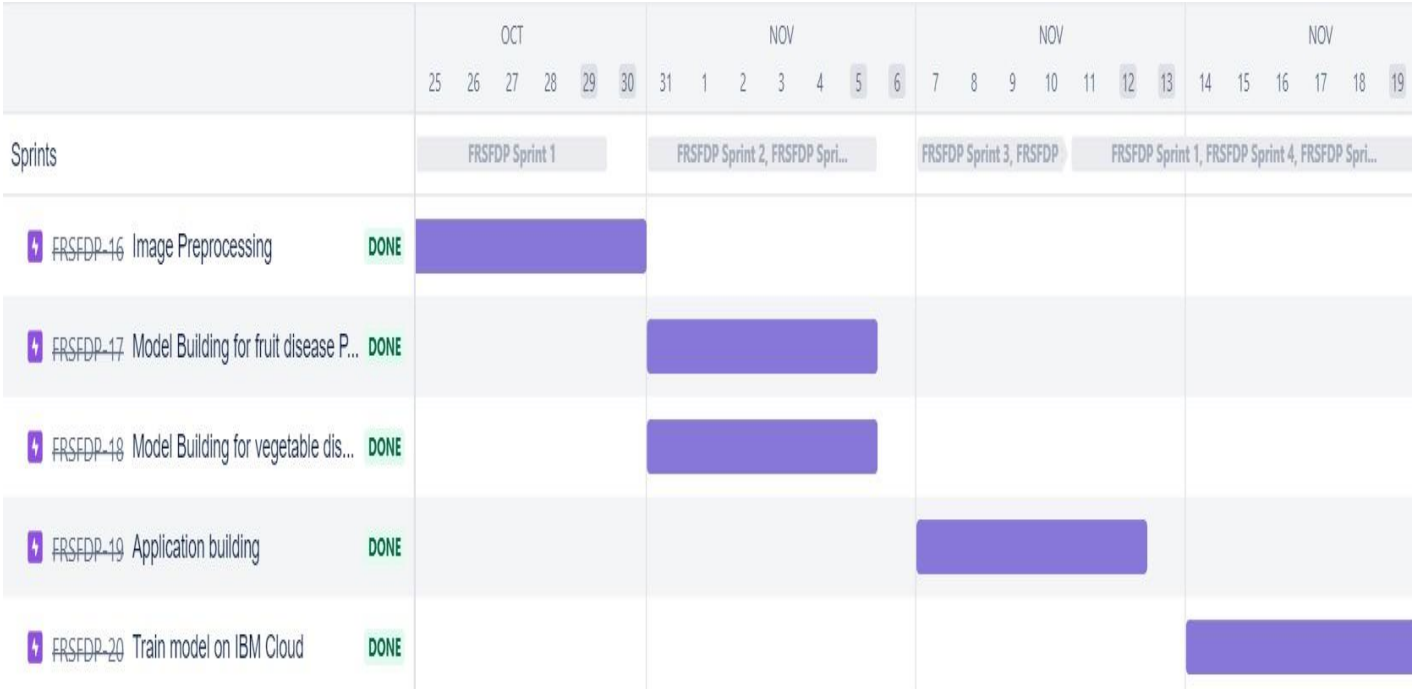
Priority breakdown

Get a holistic view of how work is being prioritized within your project. To check if the team's focusing on the right work, go to the list view.



6.3 Reports from JIRA:





7. CODING & SOLUTIONING:

7.1 Feature 1[Model Building]:

1. Import The Libraries

Import 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
```

2. Initializing The Model

Keras has 2 ways to define a neural network:

- Sequential
- Function API

The Sequential class is used to define linear initializations of network layers which then, collectively, constitute a model. In our example below, we will use the Sequential constructor tocreate a model, which will then have layers added to it using the add () method.

Now, will initialize our model.

Initialize the neural network layer by creating a reference/object to the Sequential class.

```
model=Sequential()
```

3. ADD CNNLayers

We will be adding three layers for CNN

- Convolution layer
- Pooling layer
- Flattening layer

Add Convolution Layer

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

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'))
```

Add the pooling layer

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

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

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

Add the flatten layer

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

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

4. Add Dense Layers

Now, let's add Dense Layers to know more about dense layers click below

Dense layers

The name suggests that layers are fully connected (dense) by the neurons in a network layer. Each neuron in a layer receives input from all the neurons present in the previous layer. Dense isused to add the layers.

Adding Hidden layers

This step is to add a dense layer (hidden layer). We flatten the feature map and convert it into a vector or single dimensional array in the Flatten layer. This vector array is fed it as an input to the neural network and applies an activation function, such as sigmoid or other, and returns the output.

- init is the weight initialization; function which sets all the weights and biases of a network to values suitable as a starting point for training.
- units/ output_dim, which denote is the number of neurons in the hidden layer.
- The activation function basically decides to deactivate neurons or activate them to get the desired output. It also performs a nonlinear transformation on the input to get better results on a complex neural network.
- You can add many hidden layers, in our project we are added two hidden layers. The 1st hidden layer with 40 neurons and 2nd hidden layer with 20neurons.

Adding the output layer

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

```
model.add(Dense(output_dim = 40 ,init = 'uniform',activation = 'relu'))
model.add(Dense(output_dim = 20 ,init = 'random_uniform',activation = 'relu'))
model.add(Dense(output_dim = 6,activation = 'softmax',init ='random_uniform'))
```

5. Train And Save The Model

Compile the model

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

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

Fit and save the model

Fit the neural network model with the train and test set, number of epochs and validation steps. Steps per epoch is determined by number of training images/ batch size, for validation steps number of validation images/ batch size.

```
model.fit_generator(x_train, steps_per_epoch = 168,epochs = 3,validation_data = x_test,validation_steps = 52)
```

Accuracy, Loss: Loss value implies how poorly or well a model behaves after each iteration of optimization. An accuracy metric is used to measure the algorithm's performance in an interpretable way. The accuracy of a model is usually determined after the model parameters and is calculated in the form of a percentage.

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.

6. Test The Model

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

Import the packages and load the saved model

Import the required libraries

```
from keras.preprocessing import image
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model
import numpy as np
```

Initially, we will be loading the fruit model. You can test it with the vegetable model in a similar way.

```
model = load_model("fruit.h5")
```

Load the test image, pre-process it and predict

Pre-processing the image includes converting the image to array and resizing according to the model. Give the pre-processed image to the model to know to which class your model belongsto.

```
img = image.load_img('apple_healthy.JPG',target_size = (128,128))
```

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

```
pred = model.predict_classes(x)
```

```
pred
```

```
[1]
```

The predicted class is 1.

7.2 Feature 2[Python Code]:

Build Python Code:

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

Activity 1: Build a flask application

Step 1: Load the required packages

```
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
```

Step 2: Initialize the flask app and load the model

An instance of Flask is created and the model is loaded using load_model from Keras.

```
app = Flask(__name__)

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

Step 3: Configure the home page

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

Step 4: Pre-process the frame and run

Pre-process the captured frame and give it to the model for prediction. Based on the prediction the output text is generated and sent to the HTML to display. We will be loading the precautions for fruits and vegetables excel file to get the precautions based on the output and return it to the HTML Page.

```
#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_classes(x)
            print(preds)
            df=pd.read_excel('precautions - veg.xlsx')
            print(df.iloc[preds[0]]['caution'])
        else:
            preds = model1.predict_classes(x)

            df=pd.read_excel('precautions - fruits.xlsx')
            print(df.iloc[preds[0]]['caution'])
        return df.iloc[preds[0]]['caution']
```

Run the flask application using the run method. By default, the flask runs on 5000 port. If the port is to be changed, an argument can be passed and the port can be modified.

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

8. TESTING:

8.1 Test Cases:

- Verify user is able to see the home page or not.
- Verify the UI elements in Home Page
- Verify user is able to redirect to predict page or not.
- Verify the UI elements in Predict Page
- Verify user is able to select the dropdown value or not.
- Verify user is able to upload the image or not.
- Verify whether the image is predicted correctly or not.

				Date	03-Nov-22								
				Team ID	PNT2022TMID28620								
				Project Name	Project - Fertilizers Recommendation System for Disease Prediction								
				Maximum Marks	4 marks								
Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	Executed By
HomePage_TC_OO1	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 M
HomePage_TC_OO2	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	-	Pradeep V
PredictPage_TC_OO 3	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 to 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	-	Mohammed Farook C
PredictPage_TC_OO 4	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	-	Sree Ram U, Pradeep V
PredictPage_TC_OO 5	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 to 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	-	Sree Ram U, Mohammed Farook C
PredictPage_TC_OO 6	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 to 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	Images to be Uploaded	Application should shows the uploaded image.	Working as expected	pass	Nil	N	-	Sree Ram U, Balaji M
PredictPage_TC_OO 7	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 to 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	-	Sree Ram U

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 Fertilizers Recommendation System for Disease Prediction 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	0	0	1	0	1
Duplicate	1	3	2	2	8
External	2	3	0	0	5
Fixed	4	4	4	4	16
Not Reproduced	0	0	0	1	1
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	7	10	7	7	31

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
Print Engine	1	0	0	1
Client Application	1	0	0	1
Security	1	0	0	1
Outsource Shipping	1	0	0	1
Exception Reporting	1	0	0	1
Final Report Output	1	0	0	1
Version Control	1	0	0	1

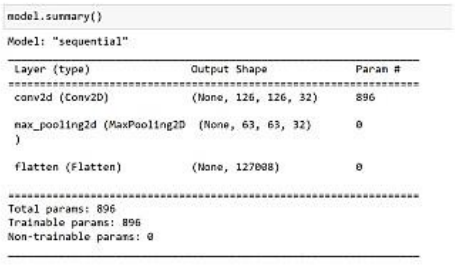
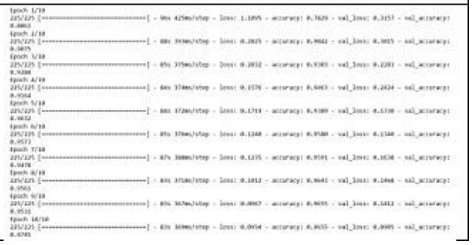
9. RESULTS:

9.1 Performance Metrics:

Date	10 November 2022
Team ID	PNT2022TMID28620
Project Name	Project - Fertilizers Recommendation System for Disease Prediction
Maximum Marks	10 Marks

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Total params: 896 Trainable params: 896 Non-trainable params: 0	
2.	Accuracy	Training Accuracy – 96.55 Validation Accuracy – 97.45	

Model Summary:

```
model.summary()
```

```
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

=================================================================
Total params: 896
Trainable params: 896
Non-trainable params: 0
_________________________________________________________________
```

Accuracy:

```
Epoch 1/10
225/225 [==============================] - 96s 425ms/step - loss: 1.1095 - accuracy: 0.7829 - val_loss: 0.3157 - val_accuracy:
0.8861
Epoch 2/10
225/225 [==============================] - 88s 393ms/step - loss: 0.2825 - accuracy: 0.9042 - val_loss: 0.3015 - val_accuracy:
0.9075
Epoch 3/10
225/225 [==============================] - 85s 375ms/step - loss: 0.2032 - accuracy: 0.9303 - val_loss: 0.2203 - val_accuracy:
0.9288
Epoch 4/10
225/225 [==============================] - 84s 374ms/step - loss: 0.1576 - accuracy: 0.9463 - val_loss: 0.2424 - val_accuracy:
0.9164
Epoch 5/10
225/225 [==============================] - 84s 372ms/step - loss: 0.1719 - accuracy: 0.9389 - val_loss: 0.1330 - val_accuracy:
0.9632
Epoch 6/10
225/225 [==============================] - 85s 376ms/step - loss: 0.1240 - accuracy: 0.9580 - val_loss: 0.1340 - val_accuracy:
0.9573
Epoch 7/10
225/225 [==============================] - 87s 388ms/step - loss: 0.1235 - accuracy: 0.9591 - val_loss: 0.1638 - val_accuracy:
0.9478
Epoch 8/10
225/225 [==============================] - 83s 371ms/step - loss: 0.1012 - accuracy: 0.9643 - val_loss: 0.1468 - val_accuracy:
0.9561
Epoch 9/10
225/225 [==============================] - 83s 367ms/step - loss: 0.0967 - accuracy: 0.9655 - val_loss: 0.1412 - val_accuracy:
0.9531
Epoch 10/10
225/225 [==============================] - 83s 369ms/step - loss: 0.0954 - accuracy: 0.9655 - val_loss: 0.0905 - val_accuracy:
0.9745
```

Locust report:

Locust Test Report

During: 11/17/2022, 5:24:47 PM - 11/17/2022, 5:34:15 PM

Target Host: http://127.0.0.1:5000

Script: locustfile.py

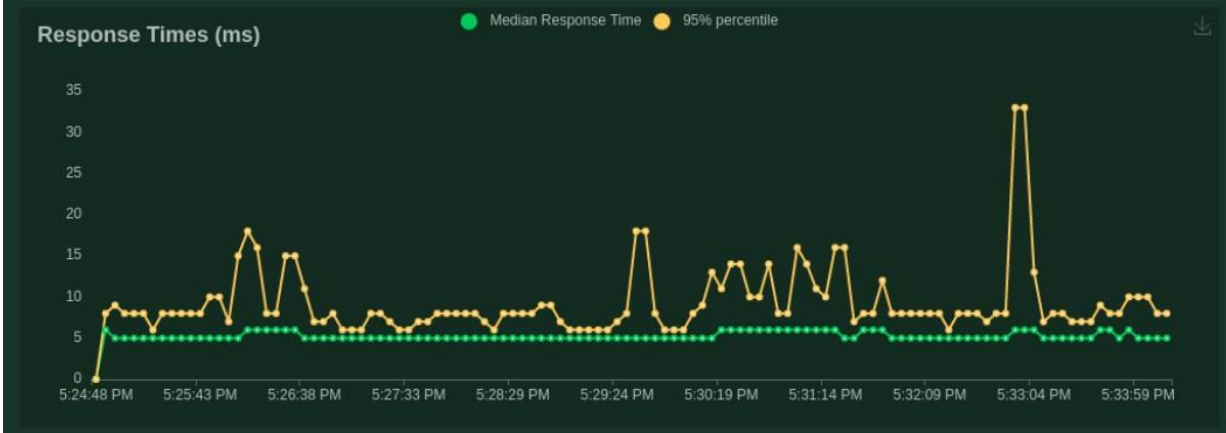
Request Statistics

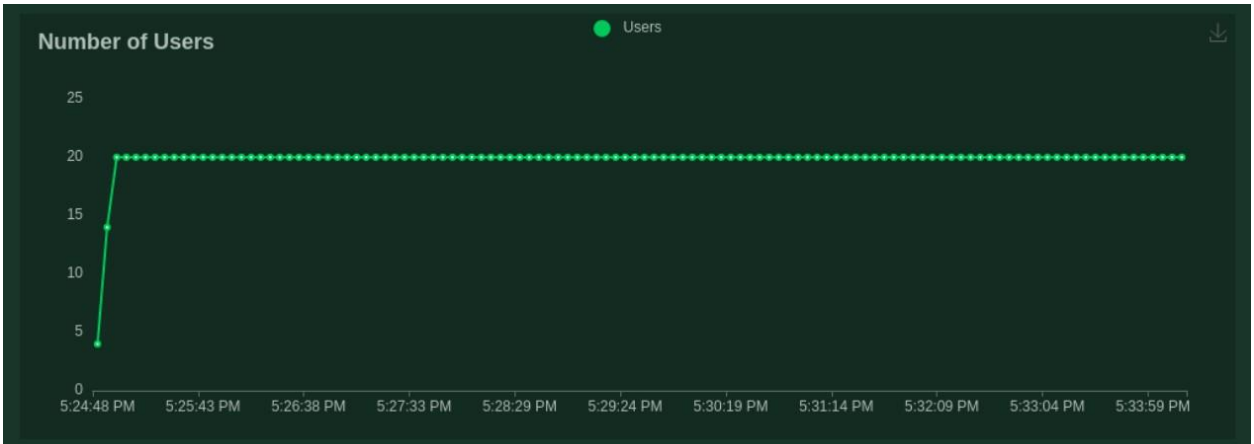
Method	Name	# Requests	# Fails	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	RPS	Failures/s
GET	/	1890	0	5	4	41	6381	3.3	0.0
GET	/prediction	1828	0	5	4	34	4484	3.2	0.0
	Aggregated	**3718**	**0**	**5**	**4**	**41**	**5448**	**6.5**	**0.0**

Response Time Statistics

Method	Name	50%ile (ms)	60%ile (ms)	70%ile (ms)	80%ile (ms)	90%ile (ms)	95%ile (ms)	99%ile (ms)	100%ile (ms)
GET	/	5	5	6	6	7	9	19	41
GET	/prediction	5	5	6	6	7	9	19	34
	Aggregated	**5**	**5**	**6**	**6**	**7**	**9**	**19**	**41**

Charts





Final ratio

Ratio per User class

- 100.0% AppUser
  - 50.0% home
  - 50.0% prediction

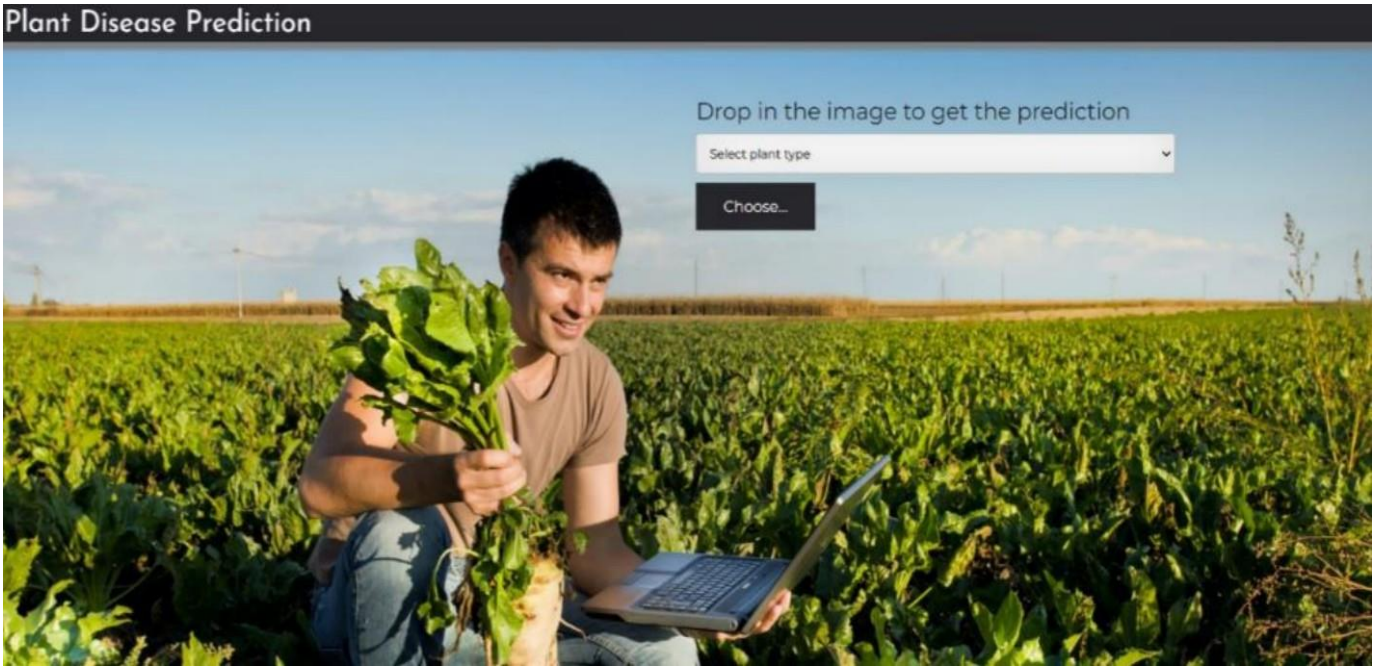
Total ratio

- 100.0% AppUser
  - 50.0% home
  - 50.0% prediction

OUTPUT:

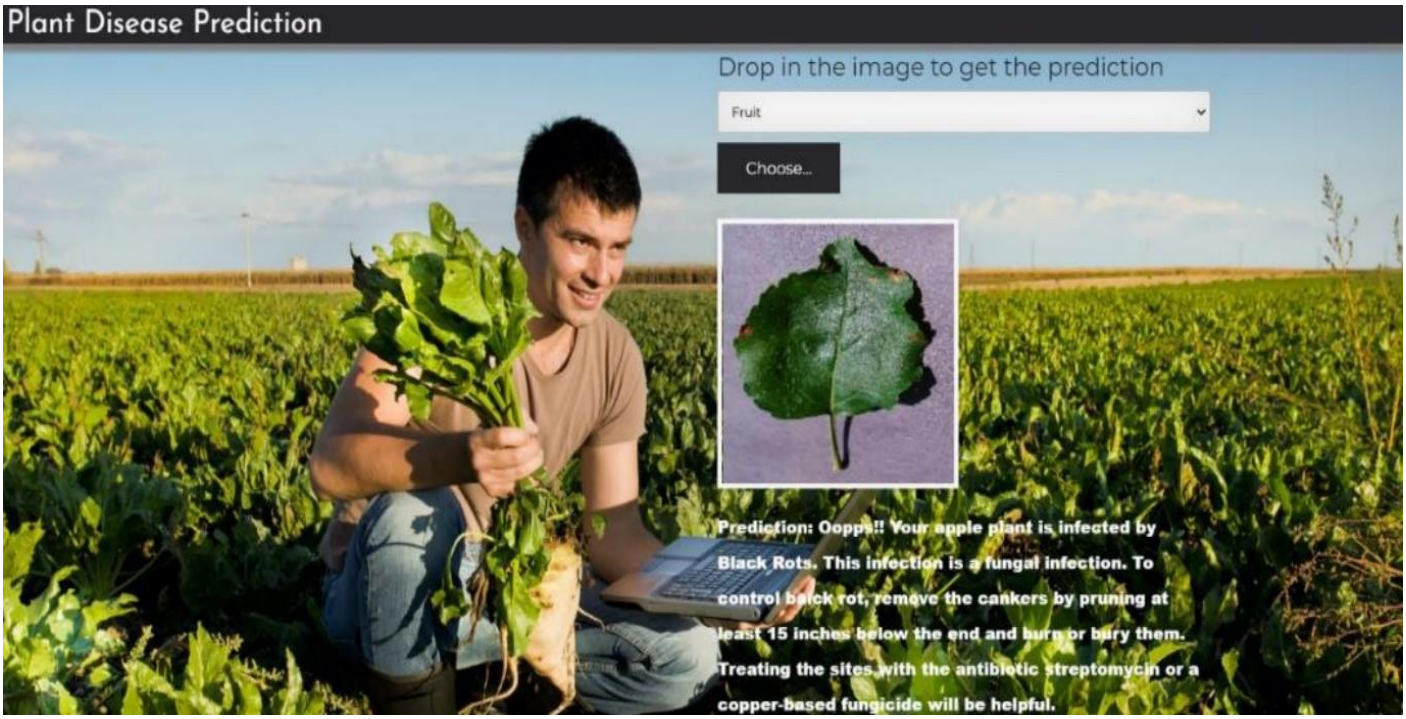
Home Page:

Prediction Page:

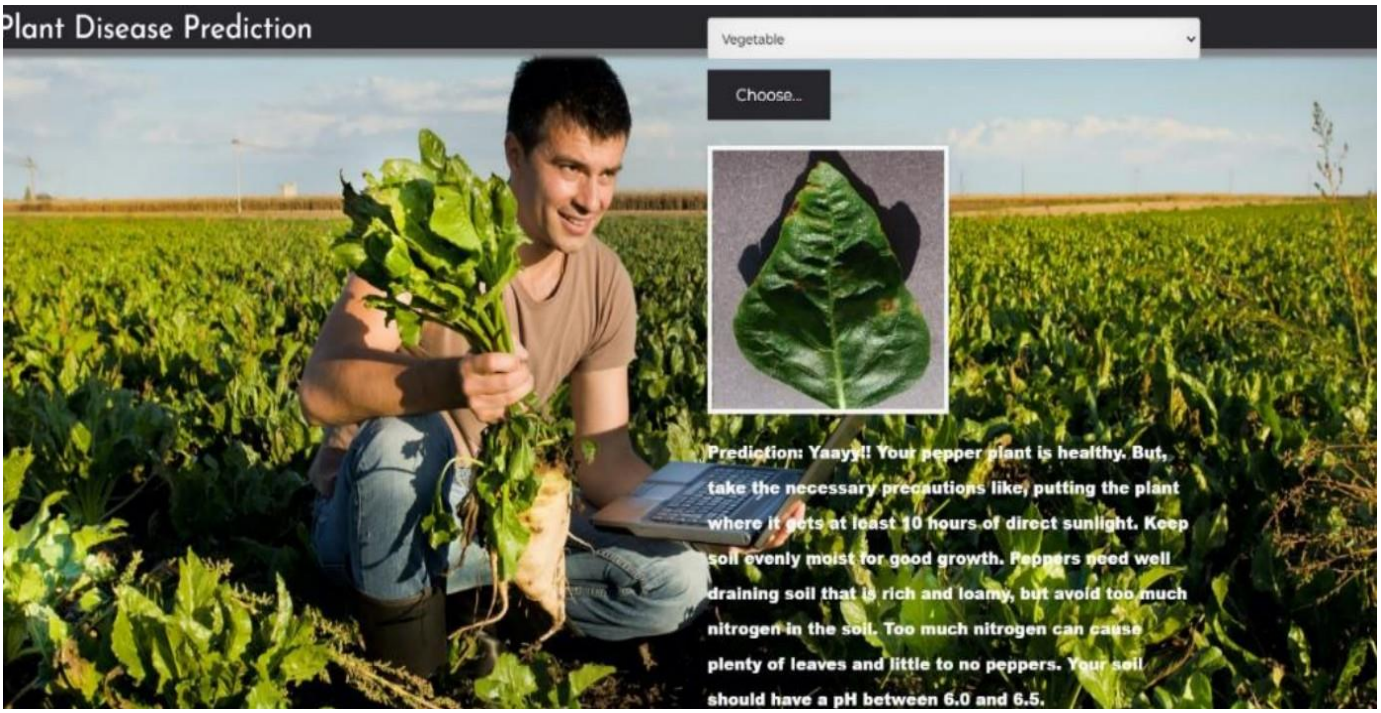


Result Page:

- **Fruit:**



- **Vegetable:**



10. ADVANTAGES & DISADVANTAGES:

ADVANTAGES:

- The proposed model could predict the disease just from the image of a particular plant.
- Easy to use UI.
- Model has some good accuracy in detecting the plant just by taking theinput(leaf).
- These kind of web applications can be used in the agricultural sector as well asfor small house hold plants as well.

Disadvantages:

- Prediction is limited to few plants as we havent trained all the plants.

11. CONCLUSION:

- 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 hasincreased due to the variation in pathogen varieties, changes in cultivation methods, and

  inadequate plant protection techniques.

- Usage of such applications could help the farmers to necessary precautions so that they don't face any loss as such.

12. FUTURE SCOPE:

- As of now we have just built the web application which apparently takes the input as an image and then predict the out in the near future we can develop an application which computer vision and AI techniques to predict the infection once you keep the camera near the plant or leaf this could make our project evenmore usable.
- This can be also done in Mobile applications like android, ios. It helps in many ways to improve the agriculture in cultivation of crops and predict the correct fertilizers to the crops.

13. APPENDIX:

Source Code[Python Code]:

```
import requests
from tensorflow.keras.preprocessing import imagefrom
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_)
global sess

global graph
graph=tf.compat.v1.get_default_graph()
model = load_model(r"C:\Users\Sree Ram\OneDrive\Desktop\IBM Project\fruit.h5")
model1=load_model(r"C:\Users\SreeRam\OneDrive\Desktop\IBM Project\vegetable.h5")
```

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

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

@app.route('/predict',methods=['POST']) def
predict():
     if request.method == 'POST':
        f = request.files['image']
       basepath = os.path.dirname(_file_)
       file_path = os.path.join(
       basepath, 'Dataset Plant Disease', 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)
```

[HTML Code]:

Home Page:

```
<!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 {
  font-family:'Times New Roman', Times, serif;
  background-image: url("../static/images/s1.jpg");
  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>

<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:40px;color:#013220;font-family:Montserrat;padding-left:20px;text-
align:center;padding-top:10%;">
<b>Fertilizers Recommendation System<br> For Disease Prediction!!</b>
</div><br>
<div         style="font-size:20px;color:#ffffff;font-family:Arial         Black;padding-left:70px;padding-
right:30px;text-align:justify;">Agriculture is one of the major sectors worls wide. Over theyears 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 labourcosts.

<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>
```

Prediction Page:

```
<!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-image: url("../static/images/s2.jpg");
  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>
           </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;color:white;font-family:Arial Black;">
</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>
```

Git-Hub Link :

https:/ github.com/IBM-EPBL/IBM-Project-26708-1660034316

Project Demonstration Link : https://www.youtube.com/embed/LsvBoJj3v8c