# FERTILISER RECOMMENDATION SYSTEM FOR DISEASE PREDICTION

# A PROJECT REPORT

## SUBMITTED BY

## TEAM ID: PNT2022TMID31637


| | |
|---|---|
| **PRAVEEN SOORAJ** | **711719104067** |
| **ROSHAN J** | **711719104077** |
| **SATHASIVAM T** | **711719104085** |
| **THAAFIA BEGUM** | **711719104099** |


In partial fulfillment for the award of the degree

Of

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**


**KGISL INSTITUTE OF TECHNOLOGY, SARAVANAMPATTI**

**ANNA UNIVERSITY :: CHENNAI 600 025**

# INDEX

## 12.FUTURE SCOPE

## 13.APPENDIX

Source Code

GitHub & Project Demo Link

# 1. INTRODUCTION

## 1.1 Project Overview

Plant disease prediction helps in the detection and recognition of plant diseases. The images of plants are captured and analyzed for certain symptoms using Computer vision and image processing. By identifying the disease, the deficit nutrients that lead to the disease are found. Based on the available data on fertilizers, the necessary nutrient rich fertilizers are recommended.

## 1.2 Purpose

The plant diseases may lead to abnormal functionalities which may end up with the death of the plant. The project aims at recognizing the symptoms at the early stages. The project also aims at guiding the farmers with the proper choice of the fertilizers that are required to counter the deficiency of the nutrients that cause the disease.

# 2. LITERATURE SURVEY

## 2.1 Existing problem

| Project Title | Algorithms used | Advantages | Disadvantages |
|---|---|---|---|
| Plant Infection Detection Using Image Processing | Infections are detected based on K-means clustering which uses hue estimation method for dividing and clustering the image and GLCM techniques that is used for texture analysis. | This system was capable of identifying the infection and classifies them accordingly with 98.27% of accuracy. This automated system reduces time of detection and labor cost | The farmers must afford mobile phones or digital camera to take images of infected leaves of different plants. |
| Prediction of crop yield and fertilizer recommendation using machine learning algorithms | Random Forest and Support Vector Machine algorithms are used for the classification of the soil to classify, display confusion matrix, Precision, Recall, predict crop based on the given inputs, etc. | It recommends fertilizer suitable for every particular crop. | Requires Third Party applications to display information on weather, temperature, humidity, atmospheric pressure, etc. |
| Plant Disease Detection Using Image | Random Forest classifier, a combination of | Accuracy scores were 93% which is nearly equal to f1 | The proposed system is able to detect 20 different diseases |

| | | | |
|---|---|---|---|
| Processing and Machine Learning | multiple decision trees is used where each tree is trained by using different subsets of the whole dataset to reduce the overfitting and improves the accuracy of the classifier. | scores. It requires less time for prediction than other deep learning-based approaches since it uses statistical machine learning and image processing algorithm. | only. |
| Fertilizers Recommendation System for Disease Prediction in Tree Leaves | Support Vector Machine (SVM) algorithm classifies the leaf image as normal or affected. And it is used to identify a function Fx which obtain the hyper-plane. | Recommend the fertilizer for affected leaves and its measurement or quantity are suggested based on severity level of the disease. | The proposed algorithm cannot be used to identify the disease that affects the other plant organs such as stems and fruits. |
| Farmer's Assistant: A Machine Learning Based Application for Agricultural Solutions | Extreme Gradient Boosting (XGBoost), is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library. It provides parallel tree boosting and is the leading machine learning library for regression, classification, and | It is expected that boosting (Random Forest) and bagging (XG Boost) models will usually perform and generalize better than non-ensemble methods. | This model performs well only on the images which are from those classes that the model already knows and it will not be able to detect the correct class for any data that is out of the domain. |

| | | | |
|---|---|---|---|
| | ranking problems. | | |

| | | | |
|---|---|---|---|
| | Random forest algorithm is also used. | | |
| Cloud Based Automated Irrigation and Plant Leaf Disease Detection System Using an Android Application. | K-means clustering is used for feature extraction. | It is simple and cost-effective system for plant leaf disease detection. | Any H/w failures may affect the system performance. |
| Detection of Leaf Diseases and Classification using Digital Image Processing. | K-Means Clustering used for image segmentation and then system extract the GLCM features from disease detected images. The disease classification done through the SVM classifier. | The system detects the diseases on citrus leaves with 90% accuracy. | System only able to detect the disease from citrus leaves. |

## 2.2 References

[1]. G. Preethi, P. Rathi, S. M. Sanjula, S. D. Lalitha, B. V. Bindhu, “Agro based crop and fertilizer recommendation system using machine learning”,

European Journal of Molecular & Clinical Medicine, 7, 4, 2020, 2043-2051
https://deepai.org/publication/farmer-s-assistant-a-machine-learning-based-application-for-agricultural-solutions

[2]. International Journal of Engineering Applied Sciences and Technology, 2019 Vol. 4, Issue 5, ISSN No. 2455-2143, Pages 371-376
https://www.ijeast.com/papers/371-376,Tesma405,IJEAST.pdf

[3]. Plant Disease Detection Using Image Processing and Machine Learning Pranesh Kulkarni1 , Atharva Karwande1 , Tejas Kolhe1 , Soham Kamble1 , Akshay Joshi1 , Medha Wyawahare1 1 Department of Electronics and Telecommunication, Vishwakarma Institute of Technology. https://arxiv.org/ftp/arxiv/papers/2106/2106.10698.pdf

[4]. Plant Infection Detection Using Image Processing - Senthilkumar Meyyappan, Nalla Malla Reddy Engineering college, Corresponding Author: Dr. Sridhathan C
https://www.researchgate.net/publication/326803995_Plant_Infection_Detection_Using_Image_Processing

[5]. Plant Disease Detection Using Image Processing DOI-10.1109/ICCUBEA.2015.153
https://ieeexplore.ieee.org/document/7155951

[6]. Metrics for Performance Measurements
https://www.mathworks.com/matlabcentral/answers/418986-how-to-calculate-true-positive-true-negative-false-positive-and-false-negative-as-we-have-segment

[7]. International journal of scientific & technology research volume 8, issue 11, November 2019 ISSN 2277-8616 3343 Fertilizers Recommendation System for Disease Prediction in Tree Leaf
http://www.ijstr.org/final-print/nov2019/Fertilizers-Recommendation-System-For-Disease-Prediction-In-Tree-Leave.pdf

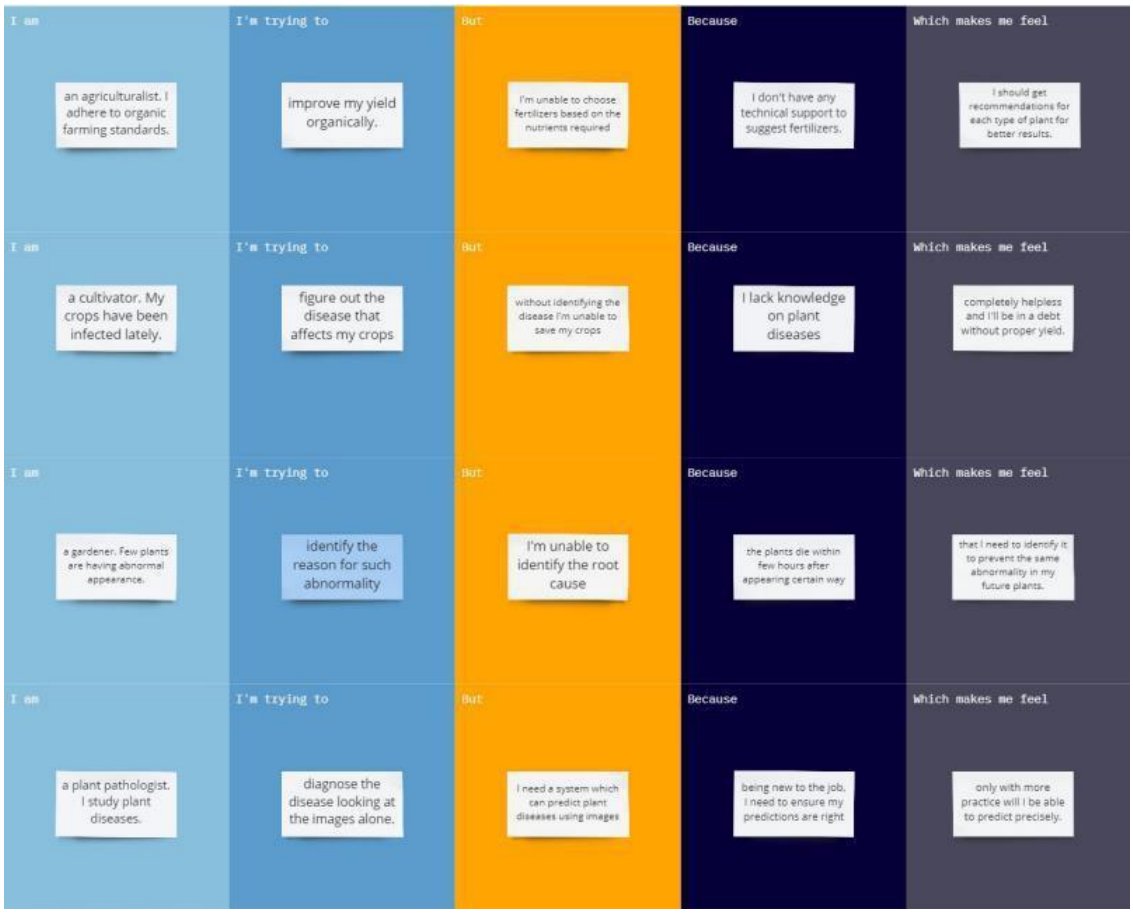
[8]. Farmer's Assistant: A Machine Learning Based Application for Agricultural Solutions - Shloka Gupta, Nishit Jain, Akshay Chopade, Aparna Bhonde, Department of Information Technology Datta Meghe College of Engineering Navi Mumbai, India.
https://arxiv.org/pdf/2204.11340.pdf

[9]. S. D. Khirade, A. B. Patil, "Plant Disease Detection Using Image Processing", 2015 International Conference on Computing Communication Control and Automation, 2015, pp. 768-771, doi: 10.1109/ICCUBEA.2015.153
https://www.semanticscholar.org/paper/Plant-Disease-Detection-Using-

Image-Processing-Khirade-Patil/575467ca9dc8d7f687fe2f490f6b18932b5c45b

## 2.3 Problem Statement Definition

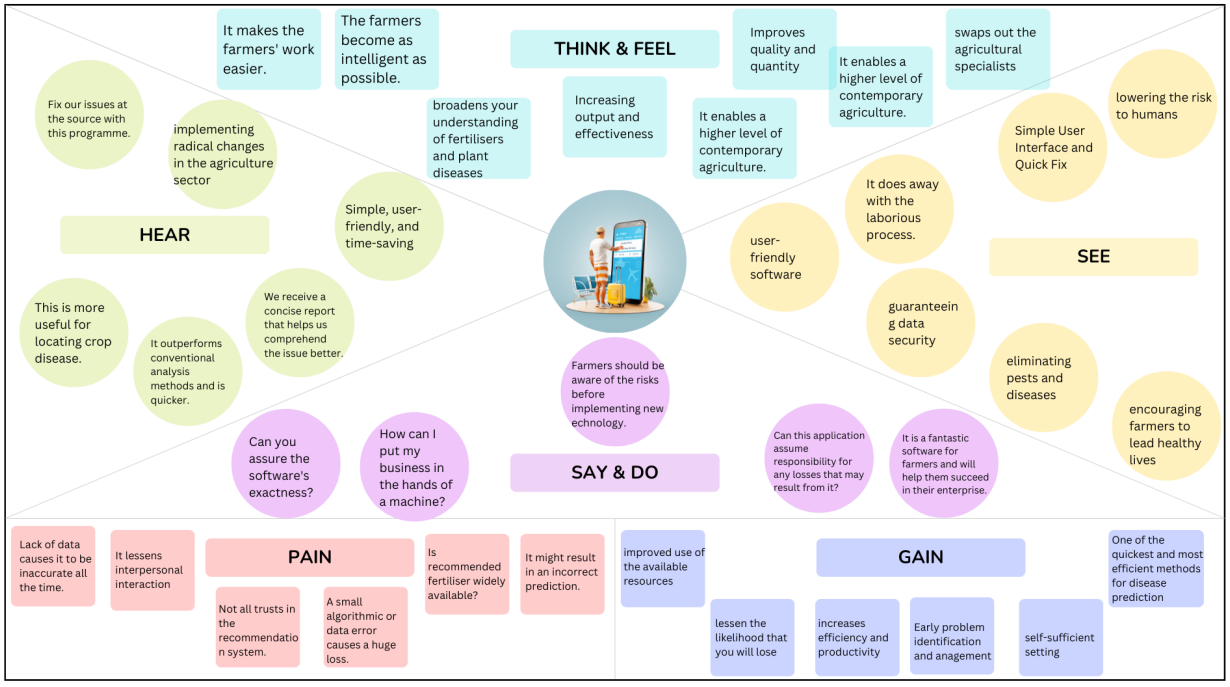
This project aims at providing a system to support the cultivators in choosing the right fertilizers for their plants to counter the deficiency of nutrients that cause various infections and diseases. The below blocks define the problems faced by the different users and the solutions that are provided by the system.



# 3. IDEATION & PROPOSED SOLUTION

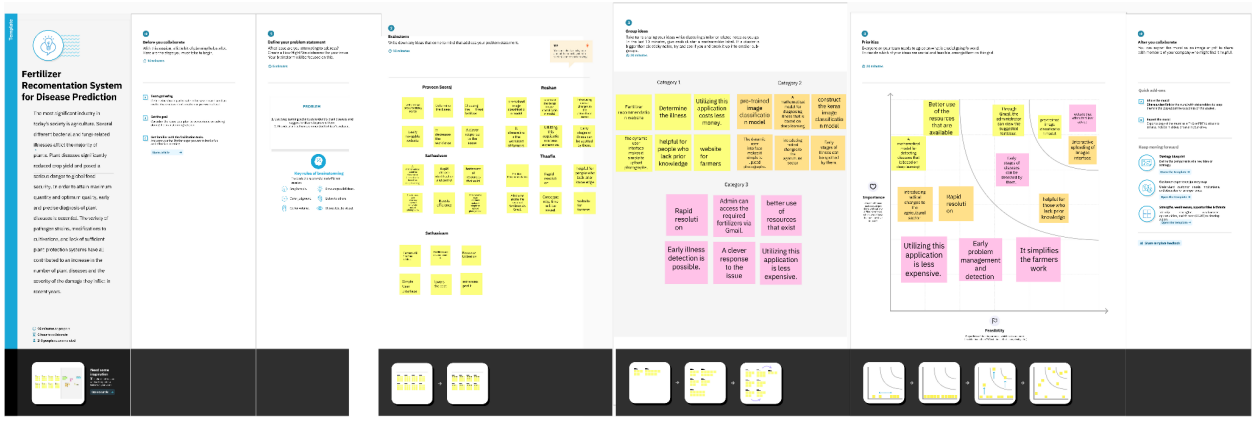
## 3.1 Empathy Map Canvas

An empathy map is used to gain deeper insights on the customer's interaction with the system. It gives an idea on what the user feels and experiences while using the system, what fears the user has respective to the system, etc. It also specifies how supportive the system environment is and what the users are likely to hear from the people around them regarding the usage of the system.



## 3.2 Ideation & Brainstorming

Ideation and Brainstorming are performed to generate ideas and solutions. Brainstorming is a group activity unlike ideation.




## 3.3 Proposed Solution

An automated system that takes the images of plant parts as input identifies 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 fertilizers that can help cure the disease. The user need not consult any specialist for identification of diseases that affected the leaves or for the recommendation of the fertilizers.

**PROJECT DESIGN PHASE - I**
**PROPOSED SOLUTION TEMPLATE**

| | |
|---|---|
| **DATE** | 24 SEPTEMBER 2022 |
| **TEAM ID** | PNT2022TMID31637 |
| **PROJECT NAME** | FERTILISERS RECOMMENDATION SYSTEM FOR DISEASE PREDICTION |
| **MAXIMUM MARKS** | 2 MARKS |

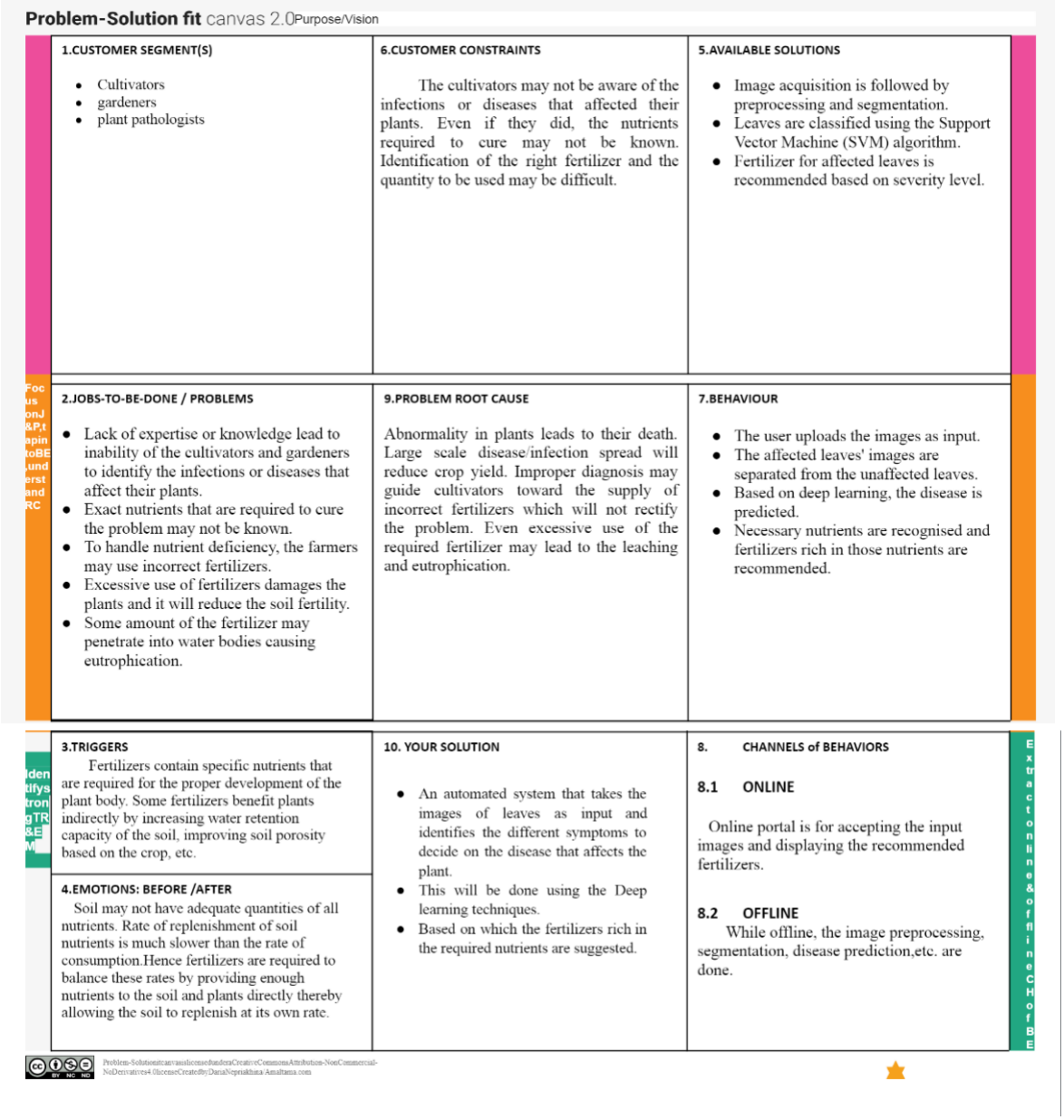
**PROPOSED SOLUTION TEMPLATE:**
Proposed team shall fill the following information in proposed solution template.

| S.NO | PARAMETER | DESCRIPTION |
|---|---|---|
| 1 | Problem Statement (Problem to be solved) | Agriculture is having a great impact on the country's economy. Different diseases effect plant that reduces their production and is a major threat to food security. The major problems that the farmers of our country are currently facing includes Crop Failure, Lack of adequate knowledge, Crop damage due to ignorance/careless, Lack of professional assistance, Inaccessibility to agro-tech Solutions. |
| 2. | Idea/Solution description | An automated system is built that takes the input as picture of leaves which is uploaded by the user, identifies 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 fertilizer needed for the plant. |
| 3. | Novelty/Uniqueness | It doesnot require user to consult any specialist for identification of diseases that affected the leaves<br><br>and the fertilizers that is required for<br><br>the same. It detects Plant disease at their early stage |

| | | |
|---|---|---|
| 4. | Social Impact/Customer Satisfaction | The whole process of identifying disease and recommendation of fertilizer happens just by uploading image so it is user friendly. It helps farmers to get good yield out of the crop<br>People will get good quality food products |
| 5. | Business Model (Revenue Model) | Social media is the best way to spread the word about our application. And with the influencers we can reach out to people<br>Clustering and targeting the farmers for identifying diseases on their plants and recommending them fertilizers for the same |
| 6. | Scalability of the solution | It can be used in research areas to study about the diseases in plant and the best fertilizer that can be recommended for it among the list of fertilizers available. It can be used by anyone in the world. |

## 3.4 Problem Solution fit

The Problem-Solution Fit means that the solution that is realized can actually solve the problem that the customer faces.



**Problem-Solution fit** canvas 2.0 Purpose/Vision

**1.CUSTOMER SEGMENT(S)**

- Cultivators
- gardeners
- plant pathologists

**6.CUSTOMER CONSTRAINTS**

The cultivators may not be aware of the infections or diseases that affected their plants. Even if they did, the nutrients required to cure may not be known. Identification of the right fertilizer and the quantity to be used may be difficult.

**5.AVAILABLE SOLUTIONS**

- Image acquisition is followed by preprocessing and segmentation.
- Leaves are classified using the Support Vector Machine (SVM) algorithm.
- Fertilizer for affected leaves is recommended based on severity level.

**2.JOBS-TO-BE-DONE / PROBLEMS**

- Lack of expertise or knowledge lead to inability of the cultivators and gardeners to identify the infections or diseases that affect their plants.
- Exact nutrients that are required to cure the problem may not be known.
- To handle nutrient deficiency, the farmers may use incorrect fertilizers.
- Excessive use of fertilizers damages the plants and it will reduce the soil fertility.
- Some amount of the fertilizer may penetrate into water bodies causing eutrophication.

**9.PROBLEM ROOT CAUSE**

Abnormality in plants leads to their death. Large scale disease/infection spread will reduce crop yield. Improper diagnosis may guide cultivators toward the supply of incorrect fertilizers which will not rectify the problem. Even excessive use of the required fertilizer may lead to the leaching and eutrophication.

**7.BEHAVIOUR**

- The user uploads the images as input.
- The affected leaves' images are separated from the unaffected leaves.
- Based on deep learning, the disease is predicted.
- Necessary nutrients are recognised and fertilizers rich in those nutrients are recommended.

**3.TRIGGERS**

Fertilizers contain specific nutrients that are required for the proper development of the plant body. Some fertilizers benefit plants indirectly by increasing water retention capacity of the soil, improving soil porosity based on the crop, etc.

**4.EMOTIONS: BEFORE /AFTER**

Soil may not have adequate quantities of all nutrients. Rate of replenishment of soil nutrients is much slower than the rate of consumption.Hence fertilizers are required to balance these rates by providing enough nutrients to the soil and plants directly thereby allowing the soil to replenish at its own rate.

**10. YOUR SOLUTION**

- An automated system that takes the images of leaves as input and identifies the different symptoms to decide on the disease that affects the plant.
- This will be done using the Deep learning techniques.
- Based on which the fertilizers rich in the required nutrients are suggested.

**8. CHANNELS of BEHAVIORS**

**8.1 ONLINE**

Online portal is for accepting the input images and displaying the recommended fertilizers.

**8.2 OFFLINE**

While offline, the image preprocessing, segmentation, disease prediction,etc. are done.

# 4. REQUIREMENT ANALYSIS

## 4.1 Functional requirement

Functional Requirements specify the features and functions of the proposed system.

**Project Design Phase-II**
**Solution Requirements (Functional & Non functional)**

| | |
|---|---|
| **Date** | **17 October 2022** |
| **Team ID** | **PNT2022TMID31637** |
| **Project Name** | **Project - Fertilizers Recommendation System for Disease Prediction** |
| **Maximum Marks** | **4 Marks** |

**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 | Utilizing a Form for Registration |
| FR-2 | User Confirmation | Email confirmation required |
| FR-3 | User Profile | after logging in, completing the profile page |
| FR-4 | Uploading Dataset (Leaf) | The leaves' pictures must be uploaded. |
| FR-5 | Requesting solution | The pre-defined model is contrasted to the photographs, and a remedy is generated. |
| FR-6 | Downloading Solution | The answer in PDF format, which includes fertilizer suggestions and potential diseases. |

**Non-functional Requirements:**

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

| FR No. | Non-Functional Requirement | Description |
|---|---|---|
| NFR-1 | Usability | The system enables the user to carry out the tasks quickly, effectively, and easily. |
| NFR-2 | Security | ensuring that no unauthorized access or malware will be used to access any of the system's data. |
| NFR-3 | Reliability | Due to the application running on a single server, the website takes time to recover from errors. |
| NFR-4 | Performance | Response and network processing times are quick. |
| NFR-5 | Availability | Approximately 98% of the time, the system will be accessible. |
| NFR-6 | Scalability | The website can be scaled |

## 4.2 Non-Functional requirements

Non functional requirements specify the general properties of the proposed system.

**Non-functional Requirements:**

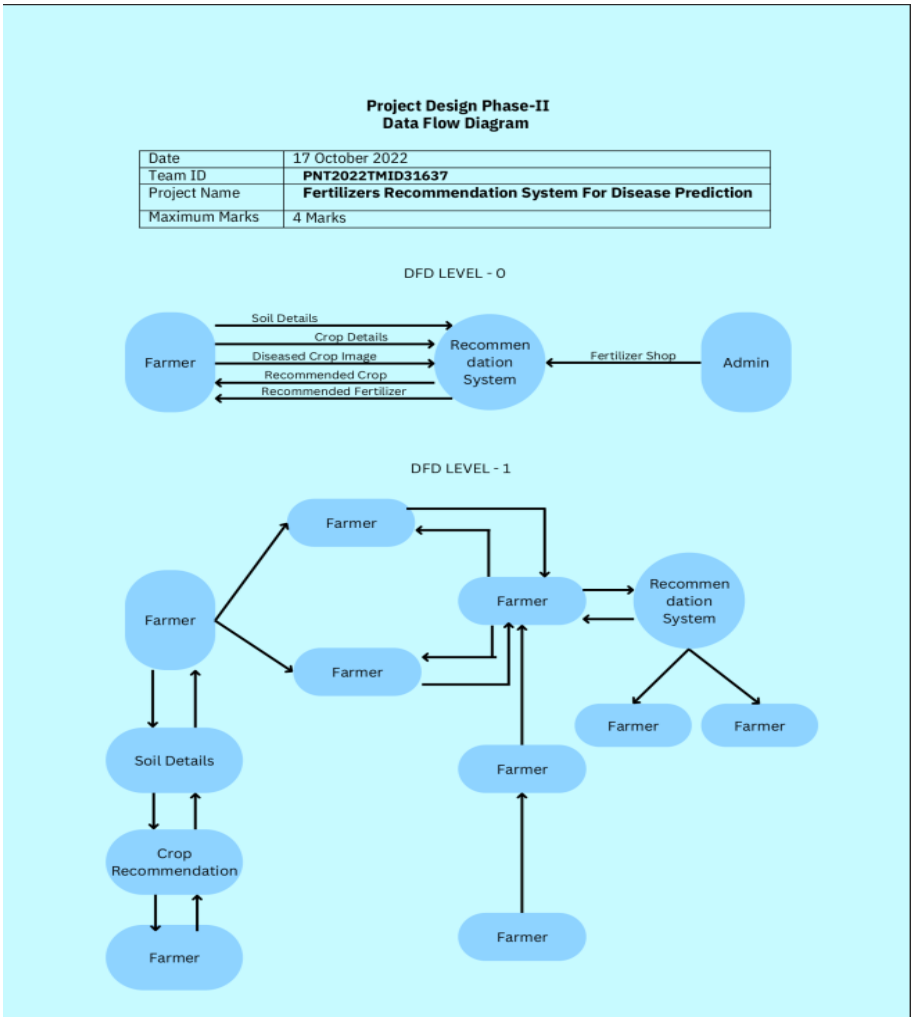
Following are the non-functional requirements of the proposed solution.

| FR No. | Non-Functional Requirement | Description |
|---|---|---|
| NFR-1 | **Usability** | Data sets can be prepared according to the leaf .Leaf datasets can be used for detection of all kind of leaf's Datasets can be reusable to detect diseases present in leaf. |
| NFR-2 | **Security** | User information and leaf data are secured The employed algorithms are more secure. |
| NFR-3 | **Reliability** | The leaf quality is more for predicting the disease in leaf. The datasets and image capture consistently performs well. |
| NFR-4 | **Performance** | The leaf problem is specified when the leaf is detected. Performs well according to the quality of the leaf and provides a specific cure to it by showing recommendation of fertilizer. |
| NFR-5 | **Availability** | The quality of the leaf will be used again for detection. Datasets will be made available and easily accessible. It is available to all users to predict plant disease. |
| NFR-6 | **Scalability** | Increasing the accuracy of disease prediction in the leaf. |

# 5. PROJECT DESIGN

## 5.1 Data Flow Diagrams

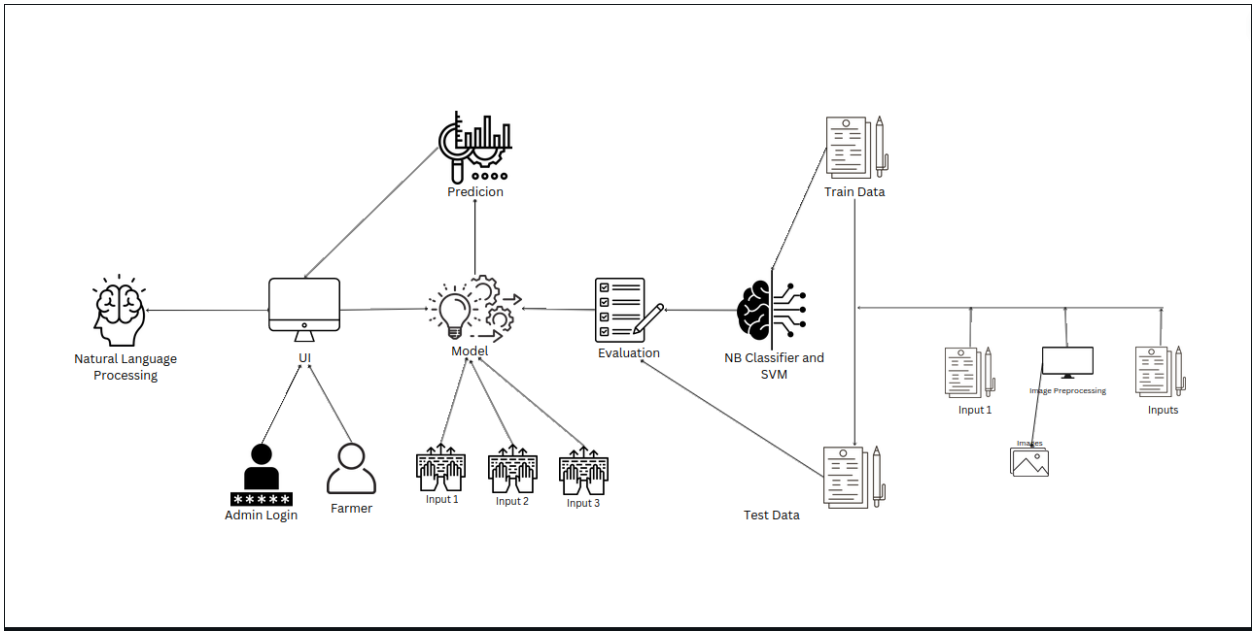
A data flow diagram or DFD(s) maps out the flow of information for any process or system. DFDs help you better understand process or system operation to discover potential problems, improve efficiency, and develop better processes.



## 5.2 Solution & Technical Architecture

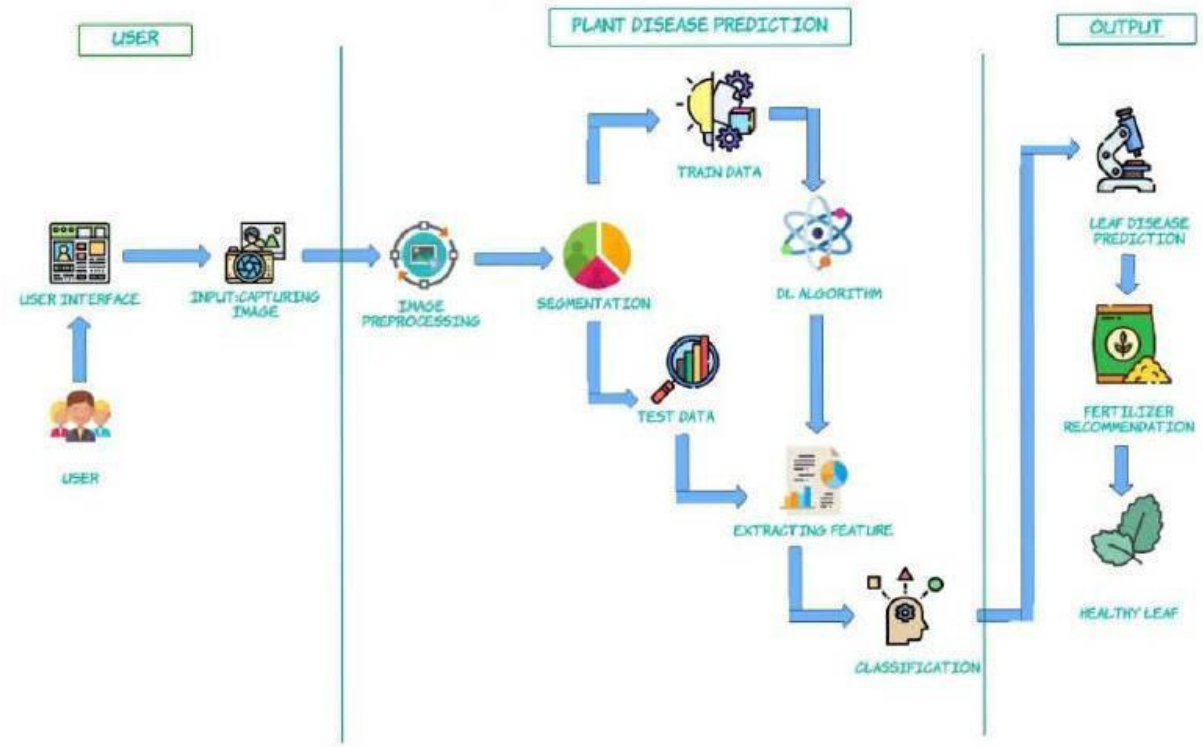
## Solution Architecture:

Solution architecture is the process of developing solutions based on predefined processes, guidelines and best practices with the objective that the developed solution fits within the enterprise architecture in terms of information architecture, system portfolios, integration requirements, etc.



## Technical Architecture:

Technical architecture involves the development of a technical blueprint regarding the arrangement, interaction, and interdependence of all elements so that system-relevant requirements are met.



## 5.3 User Stories

An informal, generic explanation of a software feature written from the viewpoint of the end user is known as a user story. Its objective is to explain how a software feature will benefit the user.

**USER STORIES:**

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

| User Type | Functional Requirement (Epic) | User Story Number | User Story / Task | Acceptance criteria | Priority | Release |
|---|---|---|---|---|---|---|
| Customer (Mobile user) | Registration | USN-1 | As a user, I can register for the application by providing my email address, password, and confirming my password . | I have access to my profile/dashboard. | High | Sprint-1 |
| | | USN-2 | Once I have registered for the application, I will receive a confirmation email. | I can receive a confirmation email and click the confirm button. | High | Sprint-1 |
| | | USN-3 | As a user, I can sign up for the application using Gmail. | I can use Gmail to access the application. | Medium | Sprint-1 |
| | Login | USN-4 | As a user, I can access the application by entering my email address and password. | I can make use of the Application for Disease Prediction | High | Sprint-1 |
| Customer (Web user) | Registration | USN-5 | As a Web user, I can register on the System with a User ID. | I can access the app like a website. | High | Sprint-1 |
| Customer Care Executive | Customer Support | USN-6 | As a supporter, I can see how customers use the product. | I can develop Customer Guidelines and Practices. | Low | Sprint-2 |
| Administrator | Analyst | USN-7 | As an admin, I can update several datasets about plant diseases. | I can store a significant amount of data. | High | Sprint-1 |
| Customer Purpose | Prediction | USN-8 | It use artificial intelligence to identify plant diseases in captured photographs and provides a live view of prediction. | I can predict plant disease. | High | Sprint-1 |

# 6. PROJECT PLANNING & SCHEDULING

## 6.1 Sprint Planning & Estimation

The purpose of sprint planning is to define what can be delivered in the sprint and how that work will be achieved. Sprint planning is done in collaboration with the whole team.

## 6.2 Sprint Delivery Schedule

Agile sprints typically last from one week to one month. The goal of sprints is to put pressure on teams to innovate and deliver more quickly, hence the shorter the sprint, the better.

**PROJECT TRACKER, VELOCITY & BURNDOWN CHART: (4 MARKS)**

| 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 OCT 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 |

**VELOCITY:**

Sprint 1 Avg Velocity:

Avg Velocity = 20/2 = 10

Sprint 2 Avg Velocity:

Avg Velocity = 20/2 = 10

Sprint 3 Avg Velocity:

Avg Velocity = 20/1 = 20

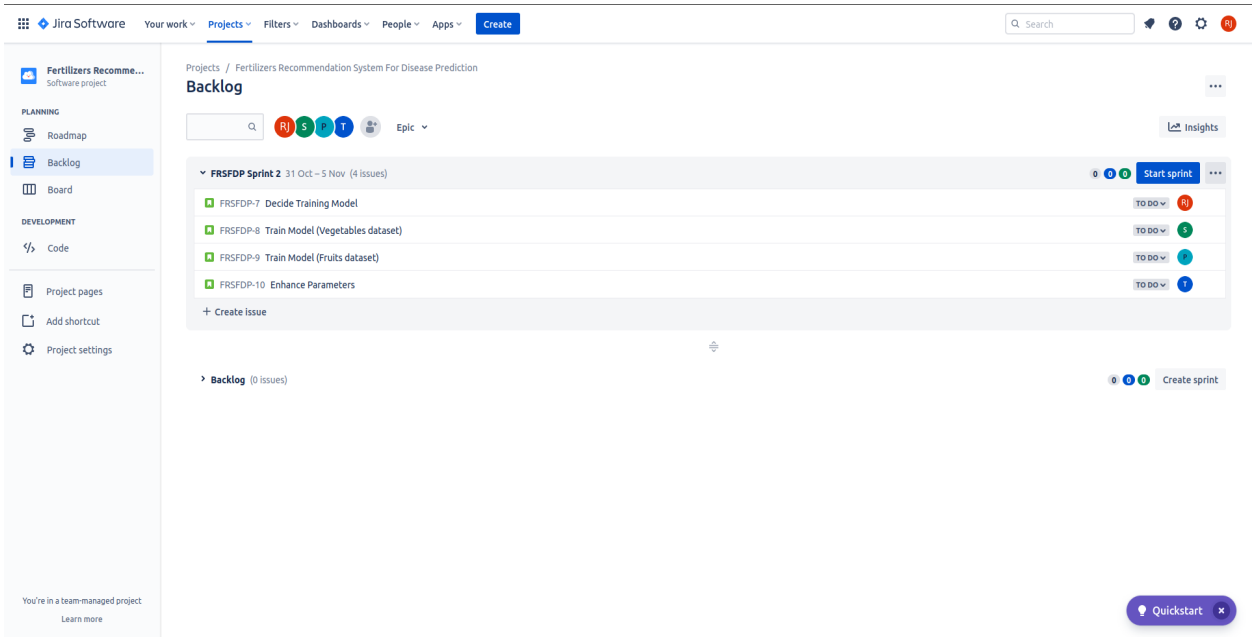
Sprint 4 Avg Velocity:

Avg Velocity = 20/2 = 10

## 6.3 Reports from JIRA

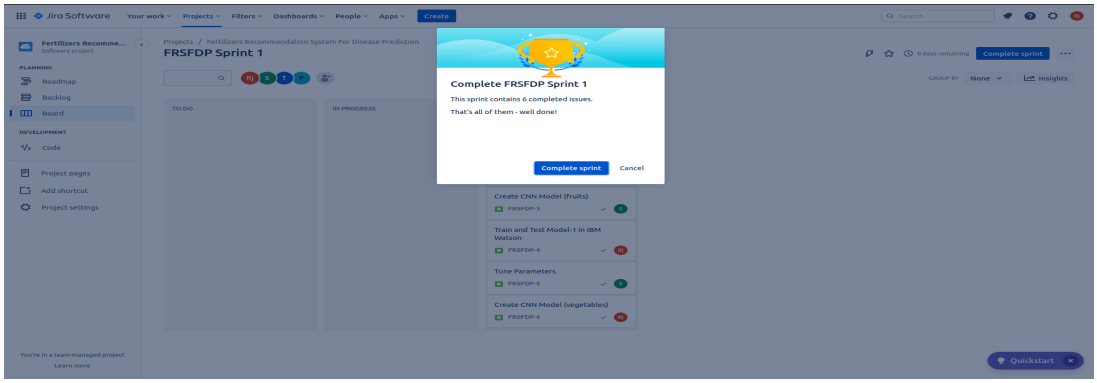
## Backlog:

A backlog is a list of issues that's related to the project and the functions of the system. It makes it simple to make, store, manage a variety of problems including the ones the team is working on.



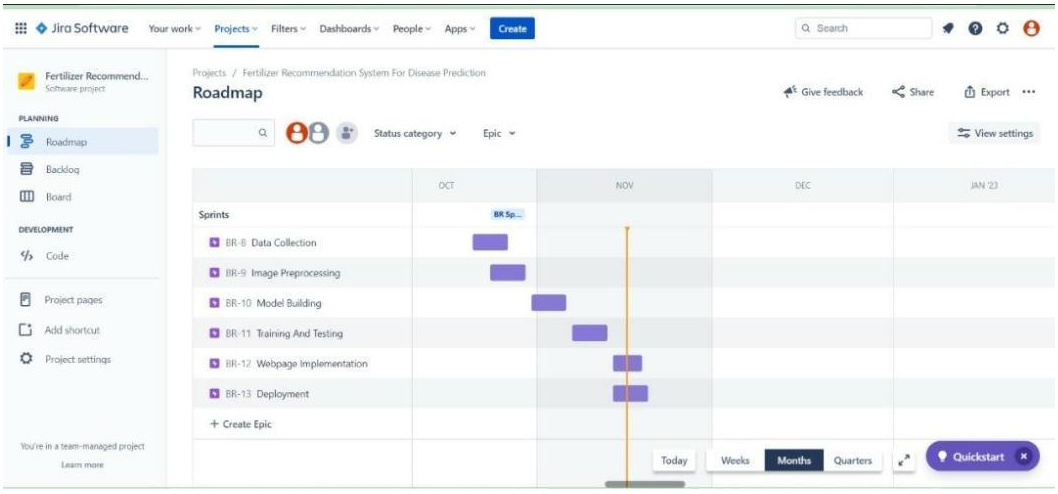
## Board:

A board reflects your team's process, tracking the status of work. The columns on the board represent the status of your team's issues. The visual representation of the work helps in discussing and tracking the progress of the project from start to finish.




## Roadmap:

A roadmap offers quick and easy planning that helps teams better manage their dependencies and track progress on the big picture in real-time.



# 7. CODING & SOLUTIONING

## Python – app.py:

```
# flask imports
from flask import Flask, request, jsonify,
make_response,render_template,redirect
from flask_sqlalchemy import SQLAlchemy
import uuid # for public id
from werkzeug.security import generate_password_hash, check_password_hash
# imports for PyJWT authentication
import jwt
from datetime import datetime, timedelta
from functools import wraps

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
import os
from werkzeug.utils import secure_filename
from tensorflow.python.keras.backend import set_session



# creates Flask object
app = Flask(__name__)
# configuration
# NEVER HARDCODE YOUR CONFIGURATION IN YOUR CODE
# INSTEAD CREATE A .env FILE AND STORE IN IT
# app.config['SECRET_KEY'] = 'your secret key'
# # database name
# app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///Database.db'
# app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = True
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///site.db'



# Model saved with Keras model.save()
```

```
# Load your trained model
model =
load_model('/home/dell/Documents/ibm-final/IBM-Project-13546-1659520952/Pr
oject Development Phase/Sprint_3/models_ibm/fruit.h5')
model1 =
load_model('/home/dell/Documents/ibm-final/IBM-Project-13546-1659520952/Pr
oject Development Phase/Sprint_3/models_ibm/fruit.h5')


print('Model loaded. Check http://127.0.0.1:5000/')

# creates SQLALCHEMY object
db = SQLAlchemy(app)

app.app_context().push()


# Database ORMs
class User(db.Model):
   id = db.Column(db.Integer, primary_key = True)
   public_id = db.Column(db.String(50), unique = True)
   name = db.Column(db.String(100))
   email = db.Column(db.String(70), unique = True)
   password = db.Column(db.String(80))

# decorator for verifying the JWT
def token_required(f):
   @wraps(f)
   def decorated(*args, **kwargs):
       token = None
       # jwt is passed in the request header
       if 'x-access-token' in request.headers:
           token = request.headers['x-access-token']
       # return 401 if token is not passed
       if not token:
           return jsonify({'message' : 'Token is missing !!'}), 401

       try:
```

```
            # decoding the payload to fetch the stored details
            data = jwt.decode(token, app.config['SECRET_KEY'])
            current_user = User.query\
                .filter_by(public_id = data['public_id'])\
                .first()
        except:
            return jsonify({
                'message' : 'Token is invalid !!'
            }), 401
        # returns the current logged in users contex to the routes
        return f(current_user, *args, **kwargs)

    return decorated

# User Database Route
# this route sends back list of users
@app.route('/user', methods =['GET'])
@token_required
def get_all_users(current_user):
    # querying the database
    # for all the entries in it
    users = User.query.all()
    # converting the query objects
    # to list of jsons
    output = []
    for user in users:
        # appending the user data json
        # to the response list
        output.append({
            'public_id': user.public_id,
            'name' : user.name,
            'email' : user.email
        })

    return jsonify({'users': output})




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

```
def dash():
   if request.method == 'POST':
       # Get the file from post request
       f = request.files['file']
       # 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']
       plant = "leaf"
       print(plant)
       if (plant=="vegetable"):
           preds = model1.predict(x)
           print(preds)
           pred = np.argmax(preds)
           df=pd.read_excel('precautions - veg.xlsx')
           print(df.iloc[pred]['caution'])
       else:
           preds = model1.predict(x)
           print(preds)
           pred = np.argmax(preds)
           df=pd.read_excel('precautions - fruits.xlsx')
           print(df.iloc[pred]['caution'])

       # return df.iloc[pred]['caution']
       return render_template("base.html")
   return render_template("base.html")




# route for logging user in
@app.route('/login', methods =['POST','GET'])
def login():
```

```
   if request.method == 'POST':
       # creates dictionary of form data
       auth = request.form

       if not auth or not auth.get('email') or not auth.get('password'):
           # returns 401 if any email or / and password is missing
           return make_response(
               'Could not verify',
               401,
               {'WWW-Authenticate' : 'Basic realm ="Login required !!"'}
           )

       user = User.query\
           .filter_by(email = auth.get('email'))\
           .first()

       if not user:
           # returns 401 if user does not exist
           return make_response(
               'Could not verify',
               401,
               {'WWW-Authenticate' : 'Basic realm ="User does not exist
!!"'}
           )

       if check_password_hash(user.password, auth.get('password')):
           # generates the JWT Token
           token = jwt.encode({
               'public_id': user.public_id,
               'exp' : datetime.utcnow() + timedelta(minutes = 30)
           }, app.config['SECRET_KEY'])

           return render_template('base.html')

       # returns 403 if password is wrong
       return make_response(
           'Could not verify',
           403,
           {'WWW-Authenticate' : 'Basic realm ="Wrong Password !!"'}
       )
```

```
    else:
        return render_template("login.html")

# signup route
@app.route('/signup', methods =['POST','GET'])
def signup():
    if request.method == 'POST':
        # creates a dictionary of the form data
        name = request.form["name"]
        email = request.form["email"]
        password = request.form["name"]

        # data = request.form

        # # gets name, email and password
        # name, email = data.get('name'),
        # data.get('email')
        # password = data.get('password')

        # checking for existing user
        user = User.query\
            .filter_by(email = email)\
            .first()
        if not user:
            # database ORM object
            user = User(
                public_id = str(uuid.uuid4()),
                name = name,
                email = email,
                password = generate_password_hash(password)
            )
            # insert user
            db.session.add(user)
            db.session.commit()

            # return make_response('Successfully registered.', 201)
            return render_template("success.html")

        else:
            # returns 202 if user already exists
```

```
            return make_response('User already exists. Please Log in.',
202)
    else:
        return render_template("register.html")



if __name__ == "__main__":
    # setting debug to True enables hot reload
    # and also provides a debugger shell
    # if you hit an error while running the server
    app.run(debug = True)
```

s

## Feature 1:

## Base.html:

```
<!DOCTYPE html>
<html lang="en">

<head>
 <meta charset="utf-8">
 <meta content="width=device-width, initial-scale=1.0"
name="viewport">

 <meta content="" name="description">
 <meta content="" name="keywords">

 <!-- Favicons -->
 <link href="assets/img/favicon.png" rel="icon">
 <link href="assets/img/apple-touch-icon.png"
rel="apple-touch-icon">

 <!-- Google Fonts -->
 <link rel="preconnect" href="https://fonts.googleapis.com">
 <link rel="preconnect" href="https://fonts.gstatic.com"
crossorigin>
 <link
href="https://fonts.googleapis.com/css2?family=Open+Sans:ital,wght@0
,300;0,400;0,500;0,600;0,700;1,300;1,400;1,600;1,700&family=Amatic+S
C:ital,wght@0,300;0,400;0,500;0,600;0,700;1,300;1,400;1,500;1,600;1,
700&family=Inter:ital,wght@0,300;0,400;0,500;0,600;0,700;1,300;1,400
;1,500;1,600;1,700&display=swap" rel="stylesheet">

 <!-- Vendor CSS Files -->
 <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstra
p.min.css" rel="stylesheet"
integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65V
ohhpuuCOmLASjC" crossorigin="anonymous">
```

```
 <!-- Template Main CSS File -->
 <link rel="stylesheet" href="{{ url_for('static',
filename='styles/main.css') }}">


</head>

<body>

 <!-- ======= Header ======= -->
 <header id="header" class="header fixed-top d-flex
align-items-center">
   <div class="container d-flex align-items-center
justify-content-between">

     <a href="/" class="logo d-flex align-items-center me-auto
me-lg-0">
       <!-- Uncomment the line below if you also wish to use an
image logo -->
       <!-- <img src="assets/img/logo.png" alt=""> -->
       <h1>Leafy<span>.</span></h1>
     </a>

     <nav id="navbar" class="navbar">
       <ul>
         <li><a href="/">Home</a></li>
         <li><a href="signup">Sign up</a></li>
         <li><a href="login">Sign in</a></li>
       </ul>
     </nav><!-- .navbar -->

     <i class="mobile-nav-toggle mobile-nav-show bi bi-list"></i>
     <i class="mobile-nav-toggle mobile-nav-hide d-none bi
bi-x"></i>

   </div>
 </header><!-- End Header -->

 {% block body %}
```

```
 <!-- ======= Hero Section ======= -->
 <section id="hero" class="hero d-flex align-items-center
section-bg">
   <div class="container">
     <div class="row justify-content-between gy-5">
       <div class="col-lg-5 order-2 order-lg-1 d-flex flex-column
justify-content-center align-items-center align-items-lg-start
text-center text-lg-start">
         <h2 data-aos="fade-up">Plant the TREE<br>Save the
Planet</h2>
         <p data-aos="fade-up" data-aos-delay="100" >Fertilizers
Recommendation System For Disease
           Prediction.</p>
         <div class="d-flex" data-aos="fade-up"
data-aos-delay="200">
           <form method="post" action="/"
enctype="multipart/form-data">
             <div class="input-group mb-3">
               <input type="file" class="form-control
btn-book-a-table" accept="image/*" id="file-upload" name="file"
onchange="previewImage(event)" required>
             </div>
             <button type="submit" style="background-color: blue;"
class="btn btn-primary btn-book-a-table w-50">Predict</button>
           </form>
         </div>
       </div>
       <div class="col-lg-5 order-1 order-lg-2 text-center
text-lg-start">
         <img id="preview-selected-image" class="img-fluid" alt=""
data-aos="zoom-out" data-aos-delay="300">
       </div>
     </div>
   </div>
 </section><!-- End Hero Section -->

 <main id="main">

   <!-- ======= About Section ======= -->
```

```
   <section id="about" class="about">
     <div class="container" data-aos="fade-up">

       <div class="section-header">
         <h2>About Us</h2>
         <p>Fertilizers Recommendation System For <span>Disease
           Prediction</span></p>
       </div>

       <div class="row gy-4">
         <div class="col-lg-6 position-relative about-img"
style="background-image:
url(https://img.freepik.com/free-photo/red-autumn-leaf_1161-255.jpg)
; background-repeat: no-repeat;" data-aos="fade-up"
data-aos-delay="150">

         </div>
         <div class="col-lg-6 d-flex align-items-end"
data-aos="fade-up" data-aos-delay="300">
           <div class="content ps-0 ps-lg-5">
             <ul>
               <li style="font-size: 20px;">Detection and
recognition of plant diseases using machine
                   learning are very efficient in providing symptoms
of identifying
                   diseases at its earliest. Plant pathologists can
analyze the
                   digital images using digital image processing for
diagnosis of
                   plant diseases. Application of computer vision
and image
                   processing strategies simply assist farmers in
all of the regions
                   of agriculture. Generally, the plant diseases are
caused by the
                   abnormal physiological functionalities of plants.
Therefore, the
                   characteristic symptoms are generated based on
the
                   differentiation between normal physiological
functionalities and
```

```
                    abnormal physiological functionalities of the
plants. 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.</li>
                </ul>



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

        </div>
      </section><!-- End About Section -->

      <!-- ======= Contact Section ======= -->
      <!-- <section id="contact" class="contact">
        <div class="container" data-aos="fade-up">

          <div class="section-header">
            <h2>Contact</h2>
            <p>Need Help? <span>Contact Us</span></p>
          </div>


          <form action="forms/contact.php" method="post" role="form"
class="php-email-form p-3 p-md-4">
            <div class="row">
              <div class="col-xl-6 form-group">
                <input type="text" name="name" class="form-control"
id="name" placeholder="Your Name" required>
              </div>
              <div class="col-xl-6 form-group">
                <input type="email" class="form-control" name="email"
id="email" placeholder="Your Email" required>
              </div>
```

```
          </div>
          <div class="form-group">
            <input type="text" class="form-control" name="subject"
id="subject" placeholder="Subject" required>
          </div>
          <div class="form-group">
            <textarea class="form-control" name="message" rows="5"
placeholder="Message" required></textarea>
          </div>
          <div class="my-3">
            <div class="loading">Loading</div>
            <div class="error-message"></div>
            <div class="sent-message">Your message has been sent.
Thank you!</div>
          </div>
          <div class="text-center"><button type="submit">Send
Message</button></div>
        </form> -->
        <!--End Contact Form -->

      <!-- </div>
    </section> -->
    <!-- End Contact Section -->

  </main><!-- End #main -->

  {% endblock %}


  <a href="#" class="scroll-top d-flex align-items-center
justify-content-center"><i class="bi bi-arrow-up-short"></i></a>

  <div id="preloader"></div>

  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.
bundle.min.js"
integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+
JcXn/tWtIaxVXM" crossorigin="anonymous"></script>
```

```
 <!-- Template Main JS File -->
 <script src="{{url_for('static',
filename='js/main.js')}}"></script>


</body>

</html>
```

## Feature 2 :

## login.html:

```
{% extends 'base.html' %}

{% block head %}
   <title>Leafy-Sign In</title>
{% endblock %}


{% block body %}

<!-- ======= Hero Section ======= -->
<main id="main">

   <section id="contact" class="contact">
       <div class="container" data-aos="fade-up">
          <div class="section-header mt-5">
           <p>Sign<span>In</span></p>
         </div>

         <form method="post" role="form" class="php-email-form p-3
p-md-4">


             <div class="form-group">
               <input type="email" class="form-control" name="email"
id="email" placeholder="Your Email" required>
             </div>
```

```
            <div class="form-group">
              <input type="password" class="form-control"
name="subject" id="subject" placeholder="Password" required>
            </div>


          <div class="text-center"><button type="submit">Sign In
</button></div>
        </form>
        <!--End Contact Form -->
       </div>
     </section><!-- End Contact Section -->

</main>
{% endblock %}
```

## Register.html:

```
{% extends 'base.html' %}

{% block head %}
   <title>Leafy-Sign In</title>
{% endblock %}


{% block body %}

<!-- ======= Hero Section ======= -->
<main id="main">

   <section id="contact" class="contact">
       <div class="container" data-aos="fade-up">
          <div class="section-header mt-5">
           <p>Sign<span>In</span></p>
         </div>
```

```
        <form method="post" role="form" class="php-email-form p-3
p-md-4">


            <div class="form-group">
              <input type="email" class="form-control" name="email"
id="email" placeholder="Your Email" required>
            </div>
            <div class="form-group">
              <input type="password" class="form-control"
name="subject" id="subject" placeholder="Password" required>
            </div>


          <div class="text-center"><button type="submit">Sign In
</button></div>
        </form>
        <!--End Contact Form -->
       </div>
    </section><!-- End Contact Section -->

</main>
{% endblock %}
```

## Success.html:

```
{% extends 'base.html' %}

{% block head %}
   <title>Leafy-Sign Up</title>
{% endblock %}


{% block body %}

<!-- ======= Hero Section ======= -->
<main id="main">
```

```
 <section id="hero" class="hero d-flex align-items-center section-bg">
   <div class="container">
     <div class="row justify-content-between gy-5">
       <div class="col-lg-12 d-flex flex-column justify-content-center
align-items-center align-items-lg-start text-center text-lg-start">
         <h2 data-aos="fade-up">Success<br>Save the Planet</h2>
         <p data-aos="fade-up" data-aos-delay="100">Fertilizers
Recommendation System For Disease
           Prediction.</p>
         <div class="d-flex" data-aos="fade-up" data-aos-delay="200">
           <a href="#book-a-table" class="btn-book-a-table">Book a
Table</a>
           <a href="https://www.youtube.com/watch?v=LXb3EKWsInQ"
class="glightbox btn-watch-video d-flex align-items-center"><i class="bi
bi-play-circle"></i><span>Watch Video</span></a>
         </div>
       </div>

     </div>
   </div>
 </section><!-- End Hero Section -->

</main>
{% endblock %}
```

Main.js

```
/**
* Template Name: Yummy - v1.2.1
* Template URL:
https://bootstrapmade.com/yummy-bootstrap-restaurant-website-template/
* Author: BootstrapMade.com
* License: https://bootstrapmade.com/license/
*/
document.addEventListener('DOMContentLoaded', () => {
   "use strict";
    /**
    * Preloader
    */
   const preloader = document.querySelector('#preloader');
```

```
  if (preloader) {
    window.addEventListener('load', () => {
      preloader.remove();
    });
  }
  /**
  * Sticky header on scroll
  */
  const selectHeader = document.querySelector('#header');
  if (selectHeader) {
    document.addEventListener('scroll', () => {
      window.scrollY > 100 ? selectHeader.classList.add('sticked') :
selectHeader.classList.remove('sticked');
    });
  }
  /**
  * Navbar links active state on scroll
  */
  let navbarlinks = document.querySelectorAll('#navbar a');
  function navbarlinksActive() {
    navbarlinks.forEach(navbarlink => {
      if (!navbarlink.hash) return;
      let section = document.querySelector(navbarlink.hash);
      if (!section) return;
      let position = window.scrollY + 200;
      if (position >= section.offsetTop && position <=
(section.offsetTop + section.offsetHeight)) {
        navbarlink.classList.add('active');
      } else {
        navbarlink.classList.remove('active');
      }
    })
  }
  window.addEventListener('load', navbarlinksActive);
  document.addEventListener('scroll', navbarlinksActive);
  /**
  * Mobile nav toggle
  */
  const mobileNavShow = document.querySelector('.mobile-nav-show');
  const mobileNavHide = document.querySelector('.mobile-nav-hide');
  document.querySelectorAll('.mobile-nav-toggle').forEach(el => {
```

```
    el.addEventListener('click', function(event) {
      event.preventDefault();
      mobileNavToogle();
    })
  });
   function mobileNavToogle() {
    document.querySelector('body').classList.toggle('mobile-nav-active');
    mobileNavShow.classList.toggle('d-none');
    mobileNavHide.classList.toggle('d-none');
  }
   /**
   * Hide mobile nav on same-page/hash links
   */
  document.querySelectorAll('#navbar a').forEach(navbarlink => {
     if (!navbarlink.hash) return;
     let section = document.querySelector(navbarlink.hash);
    if (!section) return;
     navbarlink.addEventListener('click', () => {
      if (document.querySelector('.mobile-nav-active')) {
        mobileNavToogle();
      }
    });
   });
   /**
   * Toggle mobile nav dropdowns
   */
  const navDropdowns = document.querySelectorAll('.navbar .dropdown >
a');
   navDropdowns.forEach(el => {
    el.addEventListener('click', function(event) {
      if (document.querySelector('.mobile-nav-active')) {
        event.preventDefault();
        this.classList.toggle('active');
        this.nextElementSibling.classList.toggle('dropdown-active');
         let dropDownIndicator =
this.querySelector('.dropdown-indicator');
        dropDownIndicator.classList.toggle('bi-chevron-up');
        dropDownIndicator.classList.toggle('bi-chevron-down');
      }
    })
  });
```

```
    /**
     * Scroll top button
     */
    const scrollTop = document.querySelector('.scroll-top');
    if (scrollTop) {
      const togglescrollTop = function() {
        window.scrollY > 100 ? scrollTop.classList.add('active') :
scrollTop.classList.remove('active');
      }
      window.addEventListener('load', togglescrollTop);
      document.addEventListener('scroll', togglescrollTop);
      scrollTop.addEventListener('click', window.scrollTo({
        top: 0,
        behavior: 'smooth'
      }));
    }
    /**
     * Initiate glightbox
     */
    const glightbox = GLightbox({
      selector: '.glightbox'
    });
    /**
     * Initiate pURE cOUNTER
     */
    new PureCounter();
    /**
     * Init swiper slider with 1 slide at once in desktop view
     */
    new Swiper('.slides-1', {
      speed: 600,
      loop: true,
      autoplay: {
        delay: 5000,
        disableOnInteraction: false
      },
      slidesPerView: 'auto',
      pagination: {
        el: '.swiper-pagination',
        type: 'bullets',
        clickable: true
```

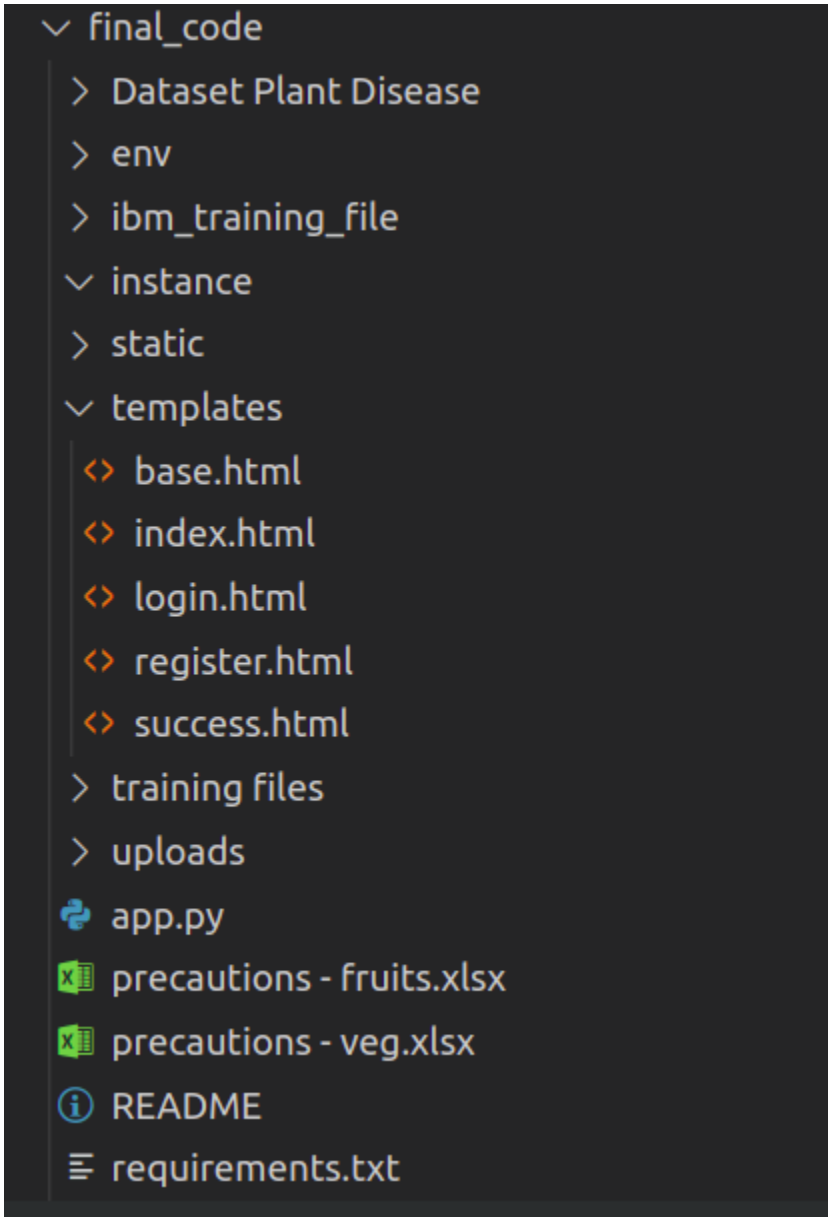
```
  },
  navigation: {
    nextEl: '.swiper-button-next',
    prevEl: '.swiper-button-prev',
  }
});
 /**
 * Init swiper slider with 3 slides at once in desktop view
 */
new Swiper('.slides-3', {
  speed: 600,
  loop: true,
  autoplay: {
    delay: 5000,
    disableOnInteraction: false
  },
  slidesPerView: 'auto',
  pagination: {
    el: '.swiper-pagination',
    type: 'bullets',
    clickable: true
  },
  navigation: {
    nextEl: '.swiper-button-next',
    prevEl: '.swiper-button-prev',
  },
  breakpoints: {
    320: {
      slidesPerView: 1,
      spaceBetween: 40
    },
     1200: {
      slidesPerView: 3,
    }
  }
});
 /**
 * Gallery Slider
 */
new Swiper('.gallery-slider', {
  speed: 400,
```

```
    loop: true,
    centeredSlides: true,
    autoplay: {
      delay: 5000,
      disableOnInteraction: false
    },
    slidesPerView: 'auto',
    pagination: {
      el: '.swiper-pagination',
      type: 'bullets',
      clickable: true
    },
    breakpoints: {
      320: {
        slidesPerView: 1,
        spaceBetween: 20
      },
      640: {
        slidesPerView: 3,
        spaceBetween: 20
      },
      992: {
        slidesPerView: 5,
        spaceBetween: 20
      }
    }
  });
   /**
   * Animation on scroll function and init
   */
  function aos_init() {
    AOS.init({
      duration: 1000,
      easing: 'ease-in-out',
      once: true,
      mirror: false
    });
  }
  window.addEventListener('load', () => {
    aos_init();
  });
```

```
  });

const previewImage = (event) => {
   const imageFiles = event.target.files;
   const imageFilesLength = imageFiles.length;
   if (imageFilesLength > 0) {
       const imageSrc = URL.createObjectURL(imageFiles[0]);
       const imagePreviewElement =
document.querySelector("#preview-selected-image");
       imagePreviewElement.src = imageSrc;
       imagePreviewElement.style.display = "block";
   }
};
```

## File Structure



# 8. TESTING

## 8.1 Test Cases

| | | |
|---|---|---|
| | | **Test Scenarios** |
| 4 | 1 | Verify COS creation |
| 5 | 2 | Verify dataset loading |
| 6 | 3 | Verify if Model is deployed |
| 7 | 4 | Verify predictions |
| | | |
| | | |
| | | **Search** |
| 11 | 1 | Check the cloud storgae |
| 12 | 2 | Check the dataset |
| 13 | 3 | Check the DL Model |
| 14 | 4 | Check the predictions |

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Date | 5 Nov 22 | | | | | | | | |
| | | | | Team ID | PNT2022TMID43580 | | | | | | | | |
| | | | | Project Name | Crude Oil Price Prediction | | | | | | | | |
| **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** |
| SPRINT_01_TC01 | Functional | Authentication | Verify New User Details | Database | Get user data from register page and store in db using flask | From user | Account is created | Working as expected | Pass | | Y | 1 | Purujit KG |
| SPRINT_01_TC02 | Functional | Authentication | Verify Login | Login form, DB | Accept username and password and check if that combination is present in the database | From user | User is logged in | Working as expected | Pass | | Y | 2 | Vasanth R |
| SPRINT_02_TC03 | Functional | Model | Verify if Model is Predicting values | Tensorflow | Train an LSTM model and feed it values | From user | Model predicts values | Working as expected | Pass | | Y | 3 | Raaj Visaanth MS |
| SPRINT_03_TC04 | Functional | Model | Verify model deployement | Loaded Model | Predict using model | Input data | Prediction is displayed in UI | Working as expected | Pass | | Y | 4 | Tarun K Kumar |

# 8.2 User Acceptance Testing

**ACCEPTANCE TESTING**
**UAT EXECUTION & REPORT SUBMISSION**

| | |
|---|---|
| DATE | 17 NOVEMBER 2022 |
| TEAM ID | PNT2022TMID31637 |
| PROJECT NAME | FERTILIZERS RECOMMENDATION SYSTEM FOR DISEASE PREDICTION |
| MAXIMUM MARKS | 4 MARKS |

1. **PURPOSE OF DOCUMENT:**

The purpose of the document is to briefly explain the least coverage and open issues of the Fertilizers recommendation system for disease prediction project at the time of the release to Use 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** |
|---|---|---|---|---|---|
| Leaf spots | 10 | 4 | 2 | 3 | 19 |
| Mosaic Leaf Pattern | 9 | 6 | 3 | 6 | 24 |
| Blights | 4 | 5 | 2 | 1 | 12 |
| Yellow leaves | 11 | 4 | 3 | 20 | 38 |
| Fruit rots | 3 | 2 | 1 | 0 | 6 |
| Misshapen leaves | 2 | 7 | 0 | 1 | 10 |
| Fruit spots | 5 | 4 | 1 | 1 | 11 |
| **Totals** | **44** | **31** | **13** | **32** | **120** |

3. **TEST CASE ANALYSIS:**
   This report shows the number of test cases that have passed, failed and untested

| SECTION | TOTAL CASES | NOT TESTED | FALL | PASS |
|---|---|---|---|---|
| Leaf spots | 18 | 0 | 0 | 18 |
| Fruit spots | 5 | 0 | 0 | 5 |
| Mosaic leaf pattern | 43 | 0 | 0 | 43 |
| Blights | 2 | 0 | 0 | 2 |
| Misshapen leaves | 25 | 0 | 0 | 25 |
| Yellow leaves | 7 | 0 | 0 | 7 |
| Fruit rots | 9 | 0 | 0 | 9 |

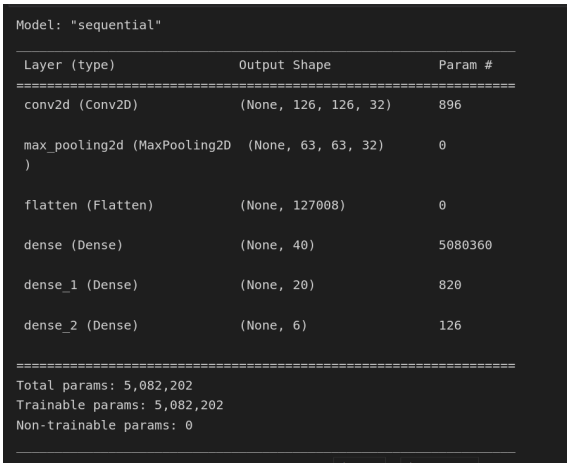
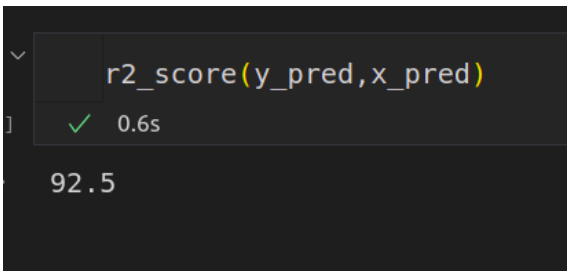
# 9. RESULTS

## Performance Metrics

**PROJECT DEVELOPMENT PHASE**
**MODEL PERFORMANCE TEST**

| | |
|---|---|
| **DATE** | 10 NOVEMBER 2022 |
| **TEAM ID** | PNT2022TMID31637 |
| **PROJECT NAME** | Fertilizers Recommendation System For Disease Prediction |
| **MAXIMUM MARKS** | 10 Marks |

**MODEL PERFORMANCE TESTING:**

| S.NO | PARAMETER | VALUES | SCREENSHOT |
|---|---|---|---|
| 1. | Model Summary | Total params: 5,082,202<br>Trainable params: 5,082,202<br>Non trainable params: 0 |  |
| 2. | Accuracy | Training Accuracy - 92.5 |  |

## 10. ADVANTAGES & DISADVANTAGES

**Advantages:**

- Detecting plant diseases early.
- The right fertilizer should be recommended to treat or prevent plant infections and diseases.
- There is no need to contact any experts.
- Complete automation.

**Disadvantages:**

- Requires using a sizable dataset to train the system.
- Works only on disorders that have been taught.
- Due to the mixed signs of several illnesses when a plant is afflicted, the system might not be able to forecast every disease.
- Need a reliable internet-connected device

## 11. CONCLUSION

As a result, a system that accepts user-provided images, analyses them for specific symptoms, recognises the disease, and then suggests fertiliser to make up for nutrient deficiencies is developed and put into use.

## 12. FUTURE SCOPE

Several photos of plant disease signs must be used to train the system. If there are several illnesses present, it is necessary to classify them properly in order to precisely anticipate each condition and suggest different fertilisers as treatments for each infection or deficiency.

# 13. APPENDIX

## Source Code

**Base.html:**

```
<!DOCTYPE html>
<html lang="en">

<head>
 <meta charset="utf-8">
 <meta content="width=device-width, initial-scale=1.0" name="viewport">

 <meta content="" name="description">
 <meta content="" name="keywords">

 <!-- Favicons -->
 <link href="assets/img/favicon.png" rel="icon">
 <link href="assets/img/apple-touch-icon.png" rel="apple-touch-icon">

 <!-- Google Fonts -->
 <link rel="preconnect" href="https://fonts.googleapis.com">
 <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
 <link
href="https://fonts.googleapis.com/css2?family=Open+Sans:ital,wght@0,300;0
,400;0,500;0,600;0,700;1,300;1,400;1,600;1,700&family=Amatic+SC:ital,wght@
0,300;0,400;0,500;0,600;0,700;1,300;1,400;1,500;1,600;1,700&family=Inter:i
tal,wght@0,300;0,400;0,500;0,600;0,700;1,300;1,400;1,500;1,600;1,700&displ
ay=swap" rel="stylesheet">

 <!-- Vendor CSS Files -->
 <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.
css" rel="stylesheet"
integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65Vohhpuu
COmLASjC" crossorigin="anonymous">


 <!-- Template Main CSS File -->
```

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


</head>

<body>

 <!-- ======= Header ======= -->
 <header id="header" class="header fixed-top d-flex align-items-center">
   <div class="container d-flex align-items-center
justify-content-between">

     <a href="/" class="logo d-flex align-items-center me-auto me-lg-0">
       <!-- Uncomment the line below if you also wish to use an image logo
-->
       <!-- <img src="assets/img/logo.png" alt=""> -->
       <h1>Leafy<span>.</span></h1>
     </a>

     <nav id="navbar" class="navbar">
       <ul>
         <li><a href="/">Home</a></li>
         <li><a href="signup">Sign up</a></li>
         <li><a href="login">Sign in</a></li>
       </ul>
     </nav><!-- .navbar -->

     <i class="mobile-nav-toggle mobile-nav-show bi bi-list"></i>
     <i class="mobile-nav-toggle mobile-nav-hide d-none bi bi-x"></i>

   </div>
 </header><!-- End Header -->

 {% block body %}



 <!-- ======= Hero Section ======= -->
 <section id="hero" class="hero d-flex align-items-center section-bg">
   <div class="container">
```

```
    <div class="row justify-content-between gy-5">
      <div class="col-lg-5 order-2 order-lg-1 d-flex flex-column
justify-content-center align-items-center align-items-lg-start text-center
text-lg-start">
        <h2 data-aos="fade-up">Plant the TREE<br>Save the Planet</h2>
        <p data-aos="fade-up" data-aos-delay="100" >Fertilizers
Recommendation System For Disease
          Prediction.</p>
        <div class="d-flex" data-aos="fade-up" data-aos-delay="200">
          <form method="post" action="/" enctype="multipart/form-data">
            <div class="input-group mb-3">
              <input type="file" class="form-control btn-book-a-table"
accept="image/*" id="file-upload" name="file"
onchange="previewImage(event)" required>
            </div>
            <button type="submit" style="background-color: blue;"
class="btn btn-primary btn-book-a-table w-50">Predict</button>
          </form>
        </div>
      </div>
      <div class="col-lg-5 order-1 order-lg-2 text-center text-lg-start">
        <img id="preview-selected-image" class="img-fluid" alt=""
data-aos="zoom-out" data-aos-delay="300">
      </div>
    </div>
  </div>
 </section><!-- End Hero Section -->

 <main id="main">

  <!-- ======= About Section ======= -->
  <section id="about" class="about">
    <div class="container" data-aos="fade-up">

      <div class="section-header">
        <h2>About Us</h2>
        <p>Fertilizers Recommendation System For <span>Disease
          Prediction</span></p>
      </div>

      <div class="row gy-4">
```

```
          <div class="col-lg-6 position-relative about-img"
style="background-image:
url(https://img.freepik.com/free-photo/red-autumn-leaf_1161-255.jpg);
background-repeat: no-repeat;" data-aos="fade-up" data-aos-delay="150">

          </div>
          <div class="col-lg-6 d-flex align-items-end" data-aos="fade-up"
data-aos-delay="300">
            <div class="content ps-0 ps-lg-5">
              <ul>
                <li style="font-size: 20px;">Detection and recognition of
plant diseases using machine
                    learning are very efficient in providing symptoms of
identifying
                    diseases at its earliest. Plant pathologists can
analyze the
                    digital images using digital image processing for
diagnosis of
                    plant diseases. Application of computer vision and
image
                    processing strategies simply assist farmers in all of
the regions
                    of agriculture. Generally, the plant diseases are
caused by the
                    abnormal physiological functionalities of plants.
Therefore, the
                    characteristic symptoms are generated based on the
                    differentiation between normal physiological
functionalities and
                    abnormal physiological functionalities of the plants.
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.</li>
              </ul>



            </div>
```

```
        </div>
      </div>

    </div>
  </section><!-- End About Section -->

  <!-- ======= Contact Section ======= -->
  <!-- <section id="contact" class="contact">
    <div class="container" data-aos="fade-up">

      <div class="section-header">
        <h2>Contact</h2>
        <p>Need Help? <span>Contact Us</span></p>
      </div>


      <form action="forms/contact.php" method="post" role="form"
class="php-email-form p-3 p-md-4">
        <div class="row">
          <div class="col-xl-6 form-group">
            <input type="text" name="name" class="form-control" id="name"
placeholder="Your Name" required>
          </div>
          <div class="col-xl-6 form-group">
            <input type="email" class="form-control" name="email"
id="email" placeholder="Your Email" required>
          </div>
        </div>
        <div class="form-group">
          <input type="text" class="form-control" name="subject"
id="subject" placeholder="Subject" required>
        </div>
        <div class="form-group">
          <textarea class="form-control" name="message" rows="5"
placeholder="Message" required></textarea>
        </div>
        <div class="my-3">
          <div class="loading">Loading</div>
          <div class="error-message"></div>
          <div class="sent-message">Your message has been sent. Thank
you!</div>
```

```
          </div>
          <div class="text-center"><button type="submit">Send
Message</button></div>
       </form> -->
       <!--End Contact Form -->

     <!-- </div>
   </section> -->
   <!-- End Contact Section -->

 </main><!-- End #main -->

 {% endblock %}


 <a href="#" class="scroll-top d-flex align-items-center
justify-content-center"><i class="bi bi-arrow-up-short"></i></a>

 <div id="preloader"></div>

 <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle
.min.js"
integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/t
WtIaxVXM" crossorigin="anonymous"></script>


 <!-- Template Main JS File -->
 <script src="{{url_for('static', filename='js/main.js')}}"></script>


</body>

</html>
```

**login.html:**

```
      {% extends 'base.html' %}

{% block head %}
```

```
   <title>Leafy-Sign In</title>
{% endblock %}


{% block body %}

<!-- ======= Hero Section ======= -->
<main id="main">

   <section id="contact" class="contact">
       <div class="container" data-aos="fade-up">
          <div class="section-header mt-5">
           <p>Sign<span>In</span></p>
         </div>

         <form method="post" role="form" class="php-email-form p-3
p-md-4">


             <div class="form-group">
               <input type="email" class="form-control" name="email"
id="email" placeholder="Your Email" required>
             </div>
             <div class="form-group">
               <input type="password" class="form-control" name="subject"
id="subject" placeholder="Password" required>
             </div>


           <div class="text-center"><button type="submit">Sign In
</button></div>
         </form>
         <!--End Contact Form -->
        </div>
     </section><!-- End Contact Section -->

</main>
{% endblock %}
```

**Register.html**

```
{% extends 'base.html' %}

{% block head %}
   <title>Leafy-Sign Up</title>
{% endblock %}


{% block body %}

<!-- ======= Hero Section ======= -->
<main id="main">

   <section class="contact">
       <div class="container" data-aos="fade-up">
          <div class="section-header mt-5">
           <p>Sign<span>Up</span></p>
         </div>

         <form method="post" class="php-email-form p-3 p-md-4">

             <div class="form-group">
               <input type="text" name="name"
class="form-control" id="name" placeholder="Your Name" required>
             </div>
             <div class="form-group">
               <input type="email" class="form-control"
name="email" id="email" placeholder="Your Email" required>
             </div>

           <div class="form-group">
             <input type="password" class="form-control"
name="password" id="password" placeholder="Create Password"
required>
           </div>

           <div class="my-3">
             <div class="loading">Loading</div>
             <div class="error-message"></div>
```

```
            <div class="sent-message">Your message has been
sent. Thank you!</div>
          </div>
          <div class="text-center"><button type="submit">Sign
Up</button></div>
        </form>
        <!--End Contact Form -->
       </div>
    </section><!-- End Contact Section -->

</main>
{% endblock %}
```

**Main.js**

```
/**
* Template Name: Yummy - v1.2.1
* Template URL:
https://bootstrapmade.com/yummy-bootstrap-restaurant-website-template/
* Author: BootstrapMade.com
* License: https://bootstrapmade.com/license/
*/
document.addEventListener('DOMContentLoaded', () => {
  "use strict";
   /**
   * Preloader
   */
  const preloader = document.querySelector('#preloader');
  if (preloader) {
    window.addEventListener('load', () => {
      preloader.remove();
    });
  }
   /**
   * Sticky header on scroll
   */
  const selectHeader = document.querySelector('#header');
  if (selectHeader) {
    document.addEventListener('scroll', () => {
```

```
      window.scrollY > 100 ? selectHeader.classList.add('sticked') :
selectHeader.classList.remove('sticked');
    });
  }
   /**
   * Navbar links active state on scroll
   */
  let navbarlinks = document.querySelectorAll('#navbar a');
   function navbarlinksActive() {
    navbarlinks.forEach(navbarlink => {
       if (!navbarlink.hash) return;
       let section = document.querySelector(navbarlink.hash);
      if (!section) return;
       let position = window.scrollY + 200;
       if (position >= section.offsetTop && position <=
(section.offsetTop + section.offsetHeight)) {
        navbarlink.classList.add('active');
      } else {
        navbarlink.classList.remove('active');
      }
    })
  }
  window.addEventListener('load', navbarlinksActive);
  document.addEventListener('scroll', navbarlinksActive);
   /**
   * Mobile nav toggle
   */
  const mobileNavShow = document.querySelector('.mobile-nav-show');
  const mobileNavHide = document.querySelector('.mobile-nav-hide');
   document.querySelectorAll('.mobile-nav-toggle').forEach(el => {
    el.addEventListener('click', function(event) {
      event.preventDefault();
      mobileNavToogle();
    })
  });
   function mobileNavToogle() {
    document.querySelector('body').classList.toggle('mobile-nav-active');
    mobileNavShow.classList.toggle('d-none');
    mobileNavHide.classList.toggle('d-none');
  }
   /**
```

```
    * Hide mobile nav on same-page/hash links
    */
   document.querySelectorAll('#navbar a').forEach(navbarlink => {
      if (!navbarlink.hash) return;
      let section = document.querySelector(navbarlink.hash);
     if (!section) return;
      navbarlink.addEventListener('click', () => {
       if (document.querySelector('.mobile-nav-active')) {
         mobileNavToogle();
       }
     });
    });
    /**
    * Toggle mobile nav dropdowns
    */
   const navDropdowns = document.querySelectorAll('.navbar .dropdown >
a');
    navDropdowns.forEach(el => {
     el.addEventListener('click', function(event) {
       if (document.querySelector('.mobile-nav-active')) {
         event.preventDefault();
         this.classList.toggle('active');
         this.nextElementSibling.classList.toggle('dropdown-active');
          let dropDownIndicator =
this.querySelector('.dropdown-indicator');
         dropDownIndicator.classList.toggle('bi-chevron-up');
         dropDownIndicator.classList.toggle('bi-chevron-down');
       }
     })
   });
    /**
    * Scroll top button
    */
   const scrollTop = document.querySelector('.scroll-top');
   if (scrollTop) {
     const togglescrollTop = function() {
       window.scrollY > 100 ? scrollTop.classList.add('active') :
scrollTop.classList.remove('active');
     }
     window.addEventListener('load', togglescrollTop);
     document.addEventListener('scroll', togglescrollTop);
```

```
  scrollTop.addEventListener('click', window.scrollTo({
    top: 0,
    behavior: 'smooth'
  }));
}
 /**
 * Initiate glightbox
 */
const glightbox = GLightbox({
  selector: '.glightbox'
});
 /**
 * Initiate pURE cOUNTER
 */
new PureCounter();
 /**
 * Init swiper slider with 1 slide at once in desktop view
 */
new Swiper('.slides-1', {
  speed: 600,
  loop: true,
  autoplay: {
    delay: 5000,
    disableOnInteraction: false
  },
  slidesPerView: 'auto',
  pagination: {
    el: '.swiper-pagination',
    type: 'bullets',
    clickable: true
  },
  navigation: {
    nextEl: '.swiper-button-next',
    prevEl: '.swiper-button-prev',
  }
});
 /**
 * Init swiper slider with 3 slides at once in desktop view
 */
new Swiper('.slides-3', {
  speed: 600,
```

```
    loop: true,
    autoplay: {
      delay: 5000,
      disableOnInteraction: false
    },
    slidesPerView: 'auto',
    pagination: {
      el: '.swiper-pagination',
      type: 'bullets',
      clickable: true
    },
    navigation: {
      nextEl: '.swiper-button-next',
      prevEl: '.swiper-button-prev',
    },
    breakpoints: {
      320: {
        slidesPerView: 1,
        spaceBetween: 40
      },
       1200: {
        slidesPerView: 3,
      }
    }
  });
   /**
   * Gallery Slider
   */
  new Swiper('.gallery-slider', {
    speed: 400,
    loop: true,
    centeredSlides: true,
    autoplay: {
      delay: 5000,
      disableOnInteraction: false
    },
    slidesPerView: 'auto',
    pagination: {
      el: '.swiper-pagination',
      type: 'bullets',
      clickable: true
```

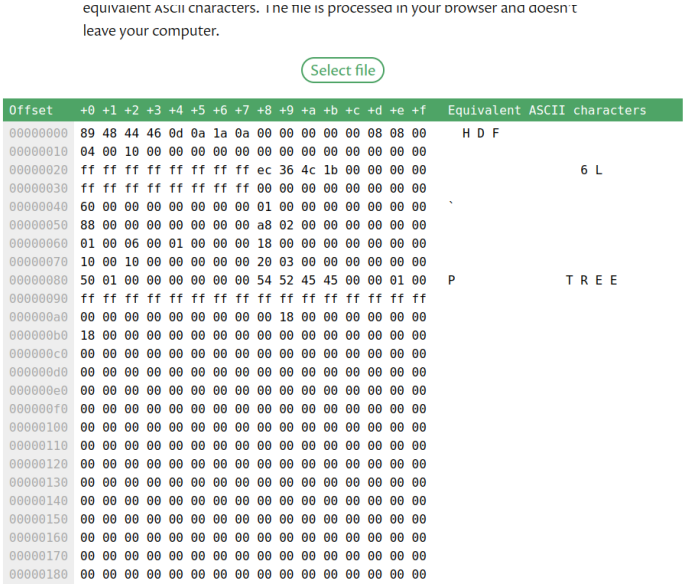
```
    },
    breakpoints: {
      320: {
        slidesPerView: 1,
        spaceBetween: 20
      },
      640: {
        slidesPerView: 3,
        spaceBetween: 20
      },
      992: {
        slidesPerView: 5,
        spaceBetween: 20
      }
    }
  });
   /**
   * Animation on scroll function and init
   */
  function aos_init() {
    AOS.init({
      duration: 1000,
      easing: 'ease-in-out',
      once: true,
      mirror: false
    });
  }
  window.addEventListener('load', () => {
    aos_init();
  });
 });

const previewImage = (event) => {
  const imageFiles = event.target.files;
  const imageFilesLength = imageFiles.length;
  if (imageFilesLength > 0) {
      const imageSrc = URL.createObjectURL(imageFiles[0]);
      const imagePreviewElement =
document.querySelector("#preview-selected-image");
      imagePreviewElement.src = imageSrc;
      imagePreviewElement.style.display = "block";
```

```
    }
};
```

# DEPLOYMENT MODEL CODE:

## Vegetable model:

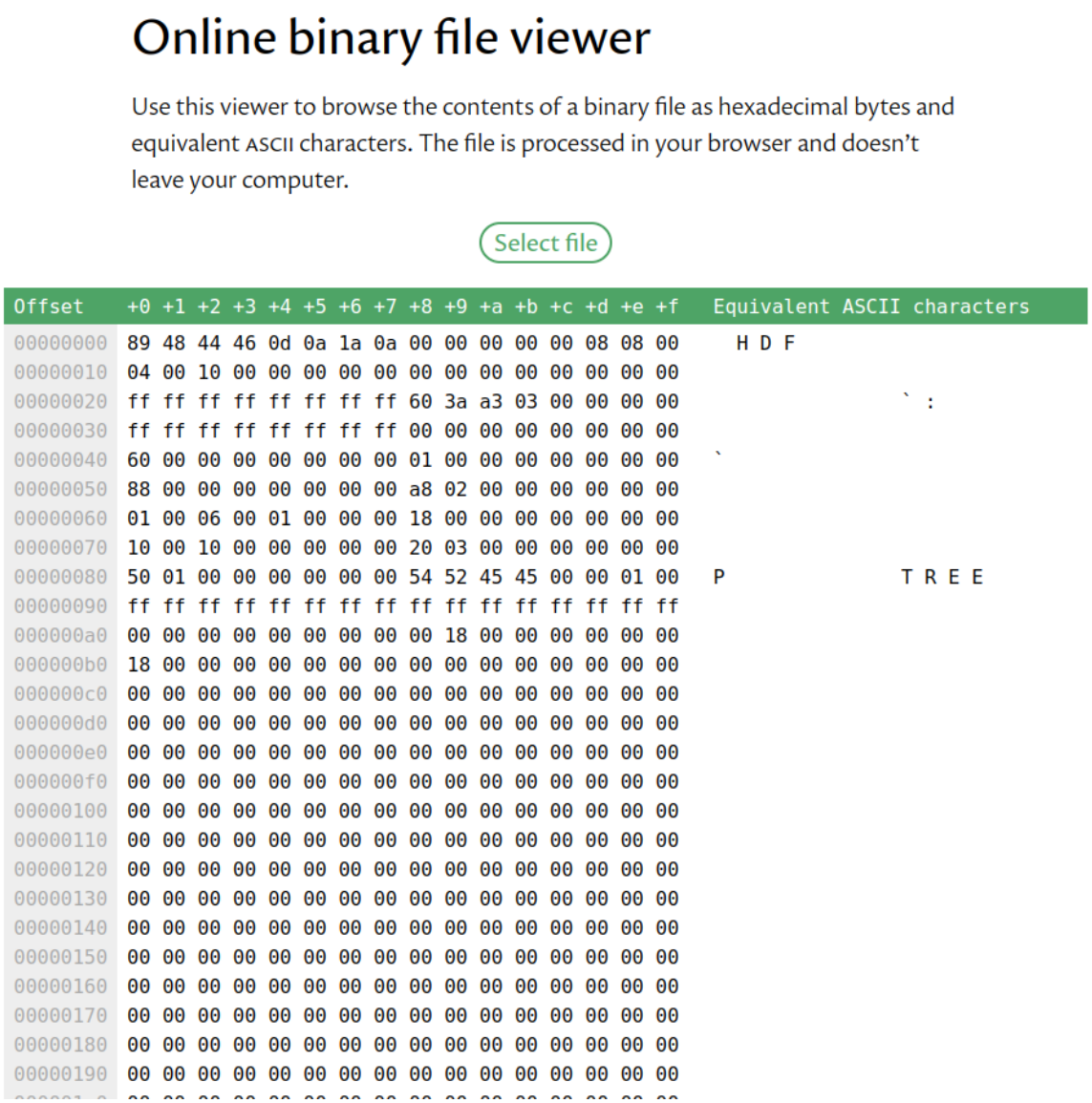


equivalent ASCII characters. The file is processed in your browser and doesn't leave your computer.

Select file

| Offset | +0 | +1 | +2 | +3 | +4 | +5 | +6 | +7 | +8 | +9 | +a | +b | +c | +d | +e | +f | Equivalent ASCII characters |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 00000000 | 89 | 48 | 44 | 46 | 0d | 0a | 1a | 0a | 00 | 00 | 00 | 00 | 00 | 08 | 08 | 00 | H D F |
| 00000010 | 04 | 00 | 10 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000020 | ff | ff | ff | ff | ff | ff | ff | ff | ec | 36 | 4c | 1b | 00 | 00 | 00 | 00 | 6 L |
| 00000030 | ff | ff | ff | ff | ff | ff | ff | ff | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000040 | 60 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 01 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | ` |
| 00000050 | 88 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | a8 | 02 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000060 | 01 | 00 | 06 | 00 | 01 | 00 | 00 | 00 | 18 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000070 | 10 | 00 | 10 | 00 | 00 | 00 | 00 | 00 | 20 | 03 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000080 | 50 | 01 | 00 | 00 | 00 | 00 | 00 | 00 | 54 | 52 | 45 | 45 | 00 | 00 | 01 | 00 | P T R E E |
| 00000090 | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | |
| 000000a0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 18 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 000000b0 | 18 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 000000c0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 000000d0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 000000e0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 000000f0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000100 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000110 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000120 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000130 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000140 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000150 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000160 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000170 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000180 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |

**Fruit model**



Online binary file viewer

Use this viewer to browse the contents of a binary file as hexadecimal bytes and equivalent ASCII characters. The file is processed in your browser and doesn't leave your computer.

Select file

| Offset | +0 | +1 | +2 | +3 | +4 | +5 | +6 | +7 | +8 | +9 | +a | +b | +c | +d | +e | +f | Equivalent ASCII characters |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 00000000 | 89 | 48 | 44 | 46 | 0d | 0a | 1a | 0a | 00 | 00 | 00 | 00 | 00 | 08 | 08 | 00 | H D F |
| 00000010 | 04 | 00 | 10 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000020 | ff | ff | ff | ff | ff | ff | ff | ff | 60 | 3a | a3 | 03 | 00 | 00 | 00 | 00 | ` : |
| 00000030 | ff | ff | ff | ff | ff | ff | ff | ff | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000040 | 60 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 01 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | ` |
| 00000050 | 88 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | a8 | 02 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000060 | 01 | 00 | 06 | 00 | 01 | 00 | 00 | 00 | 18 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000070 | 10 | 00 | 10 | 00 | 00 | 00 | 00 | 00 | 20 | 03 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000080 | 50 | 01 | 00 | 00 | 00 | 00 | 00 | 00 | 54 | 52 | 45 | 45 | 00 | 00 | 01 | 00 | P T R E E |
| 00000090 | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | ff | |
| 000000a0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 18 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 000000b0 | 18 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 000000c0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 000000d0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 000000e0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 000000f0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000100 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000110 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000120 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000130 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000140 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000150 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000160 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000170 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000180 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 00000190 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |

# Image Augmentation

```
from keras.preprocessing import image
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import load_model
import numpy as nps
model=load_model(r'../Model_Building/Model_Building/Model_Building_For_Veg
etable_Disease_Prediction/vegetable.h5')
import numpy as np
```

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

import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It
includes your credentials.
# You might want to remove those credentials before you share the
notebook.
cos_client = ibm_boto3.client(service_name='s3',
   ibm_api_key_id='z_pQzwYPsnwgHgzAF2ZhxF5BM0HC70n05v6cANDZ3HaT',
   ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
   config=Config(signature_version='oauth'),

endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'fertilizersrecommendationsystemfo-donotdelete-pr-cerofklwxzbgci'
object_key = 'Dataset Plant Disease.zip'

streaming_body_1 = cos_client.get_object(Bucket=bucket,
Key=object_key)['Body']
if not
hasattr(streaming_body_1,"__iter__"):streaming_body_1.__iter__=types.Metho
Type(__iter__,streaming_body_1)


# Your data file was loaded into a botocore.response.StreamingBody object.
# Please read the documentation of ibm_boto3 and pandas to learn more
about the possibilities to load the data.
# ibm_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/
# pandas documentation: http://pandas.pydata.org/

from io import BytesIO
```

```
import zipfile
unzip = zipfile.ZipFile(BytesIO(streaming_body_1.read()),'r')
file_paths = unzip.namelist()
for path in file_paths:
   unzip.extract(path)
pwd
import os
filenames=os.listdir("/home/wsuser/work/Dataset Plant
Disease/fruit-dataset/fruit-dataset/train")
x_train=train_datagen.flow_from_directory('/home/wsuser/work/Dataset Plant
Disease/fruit-dataset/fruit-dataset/train',target_size=(128,128),batch_siz
e=2,class_mode='categorical')
x_test=test_datagen.flow_from_directory('/home/wsuser/work/Dataset Plant
Disease/fruit-dataset/fruit-dataset/test',target_size=(128,128),batch_size
=2,class_mode='categorical')
x_train.class_indices
```

# CNN

```
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
model=Sequential()
model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'
))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
model.add(Dense(units=40,kernel_initializer='uniform',activation='relu'))
model.add(Dense(units=20,kernel_initializer='random_uniform',activation='r
elu'))
model.add(Dense(units=6,kernel_initializer='random_uniform',activation='so
ftmax'))
model.compile(loss='categorical_crossentropy',optimizer="adam",metrics=["a
ccuracy"])
```

Output exceeds the size limit. Open the full output data in a text editor
Epoch 1/20
89/89 [==============================] - 8s 80ms/step - loss: 1.6727 - accuracy: 0.3090 - val_loss: 130.4584 - val_accuracy: 0.3333
Epoch 2/20
89/89 [==============================] - 7s 76ms/step - loss: 1.6646 - accuracy: 0.3090 - val_loss: 102.0820 - val_accuracy: 0.4444
Epoch 3/20
89/89 [==============================] - 7s 75ms/step - loss: 1.3100 - accuracy: 0.5056 - val_loss: 154.1361 - val_accuracy: 0.5926
Epoch 4/20
89/89 [==============================] - 7s 80ms/step - loss: 0.9234 - accuracy: 0.6798 - val_loss: 69.0112 - val_accuracy: 0.7593
Epoch 5/20
89/89 [==============================] - 7s 78ms/step - loss: 0.8374 - accuracy: 0.6854 - val_loss: 90.7320 - val_accuracy: 0.6667
Epoch 6/20
89/89 [==============================] - 7s 76ms/step - loss: 0.8152 - accuracy: 0.6910 - val_loss: 36.6560 - val_accuracy: 0.7407
Epoch 7/20
89/89 [==============================] - 6s 73ms/step - loss: 0.8015 - accuracy: 0.7247 - val_loss: 65.8309 - val_accuracy: 0.7407
Epoch 8/20
89/89 [==============================] - 7s 73ms/step - loss: 0.6968 - accuracy: 0.7303 - val_loss: 98.7893 - val_accuracy: 0.6852
Epoch 9/20
89/89 [==============================] - 7s 76ms/step - loss: 0.7190 - accuracy: 0.7191 - val_loss: 71.7151 - val_accuracy: 0.7222
Epoch 10/20
89/89 [==============================] - 7s 77ms/step - loss: 0.6637 - accuracy: 0.7472 - val_loss: 80.6197 - val_accuracy: 0.6296
Epoch 11/20
89/89 [==============================] - 7s 77ms/step - loss: 0.6850 - accuracy: 0.7584 - val_loss: 122.2742 - val_accuracy: 0.6667
Epoch 12/20
89/89 [==============================] - 7s 78ms/step - loss: 0.7138 - accuracy: 0.7360 - val_loss: 56.3075 - val_accuracy: 0.7593
Epoch 13/20
...
Epoch 19/20
89/89 [==============================] - 7s 75ms/step - loss: 0.3732 - accuracy: 0.8764 - val_loss: 199.4278 - val_accuracy: 0.7037
Epoch 20/20
89/89 [==============================] - 7s 77ms/step - loss: 0.5277 - accuracy: 0.7697 - val_loss: 36.3950 - val_accuracy: 0.7963

<keras.callbacks.History at 0x7f35ca1a80d0>

## Saving Model

```
ls
fruit-dataset/
model.save('fruit.h5')
!tar -zcvf Train-model_new.tgz fruit.h5
fruit.h5
ls -1
fruit-dataset/
fruit.h5
```

## GitHub Link:

https://github.com/IBM-EPBL/IBM-Project-13546-1659520952