NALAIYA THIRAN PROJECT 2022

FERTILISER RECOMMENDATION SYSTEM FOR DISEASE PREDICTION

Batch : B1-1M3E
Team ID : PNT2022TMID26714
College : ST.JOSEPH COLLEGE OF ENGINEERING
Team Leader : SelvaSaravanan.L (212919205042)
Team Members : Vishwanath.S (212919205061)
Perumal.P (212919205032)
SuriyaPrakash.M (212919205051)

INDEX

1. **INTRODUCTION**
   - **1.1** Project Overview
   - **1.2** Purpose

2. **LITERATURE SURVEY**
   - **2.1** Existing problem
   - **2.2** References
   - **2.3** Problem Statement Definition

3. **IDEATION & PROPOSED SOLUTION**
   - **3.1** Empathy Map Canvas
   - **3.2** Ideation & Brainstorming
   - **3.3** Proposed Solution
   - **3.4** Problem Solution fit

4. **REQUIREMENT ANALYSIS**
   - **4.1** Functional requirement
   - **4.2** Non-Functional requirements

5. **PROJECT DESIGN**
   - **5.1** Data Flow Diagrams
   - **5.2** Solution & Technical Architecture
   - **5.3** User Stories

6. **PROJECT PLANNING & SCHEDULING**
   - **6.1** Sprint Planning & Estimation
   - **6.2** Sprint Delivery Schedule
   - **6.3** Reports from JIRA

7. CODING & SOLUTIONING

7.1 Feature 1
7.2 Feature 2

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

Plant disease prediction helps in the detection and recognition of the 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 onfertilizers, 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 fertilizersthat 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 ranking problems.	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.

	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

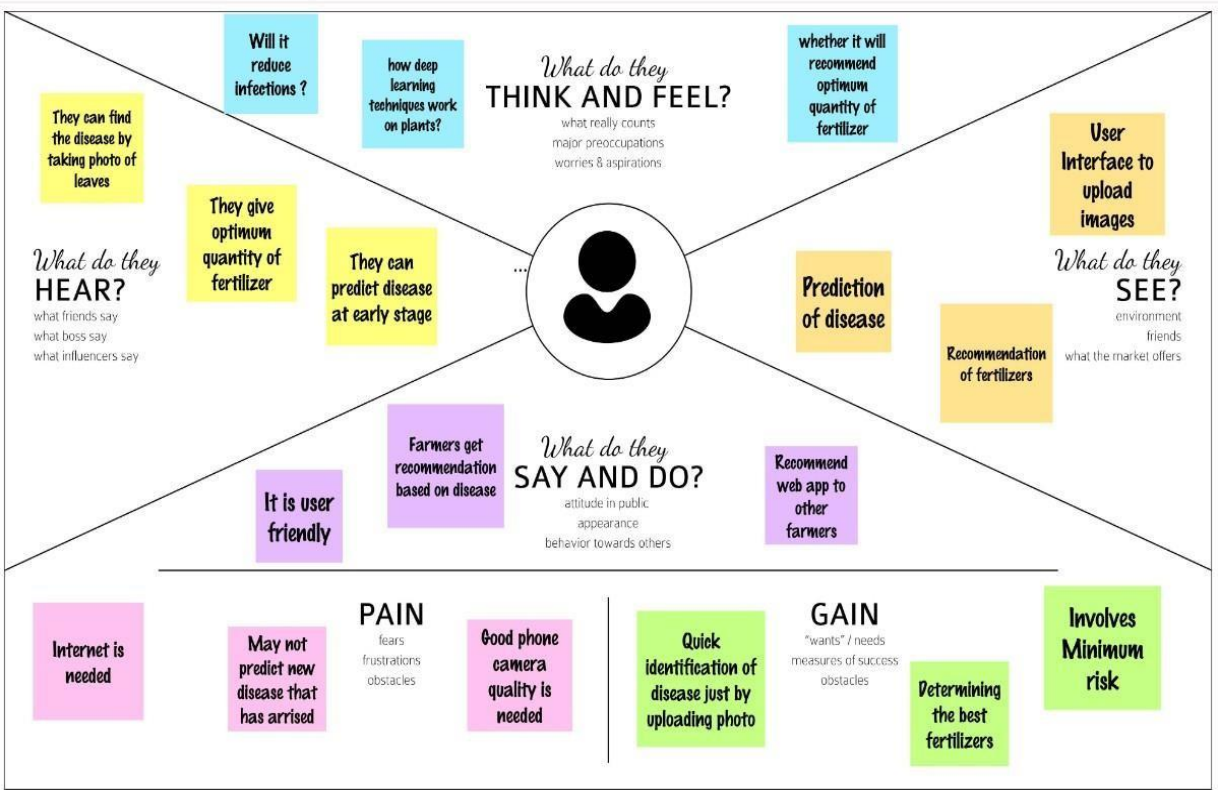
2.3 Problem Statement Definition

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

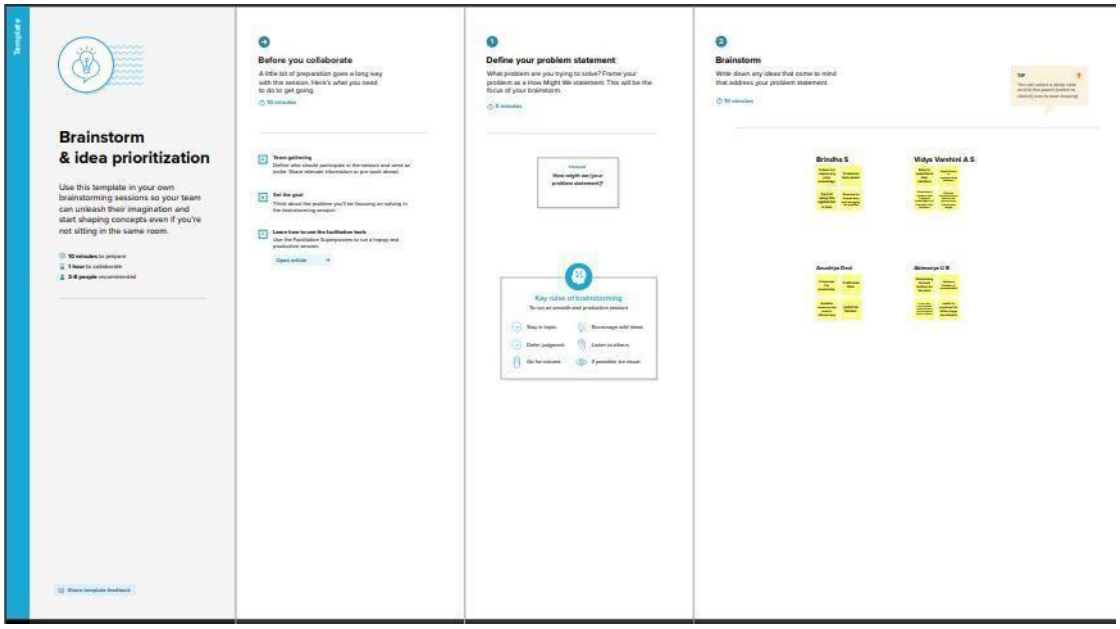
An empathy map is used to gain deeper insights on the customer's interaction with the system. It gives an idea on what the user feels and experiences while using the system, what fears the user has respective to the system, etc. It also specifies how supportive the system environment is and what the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

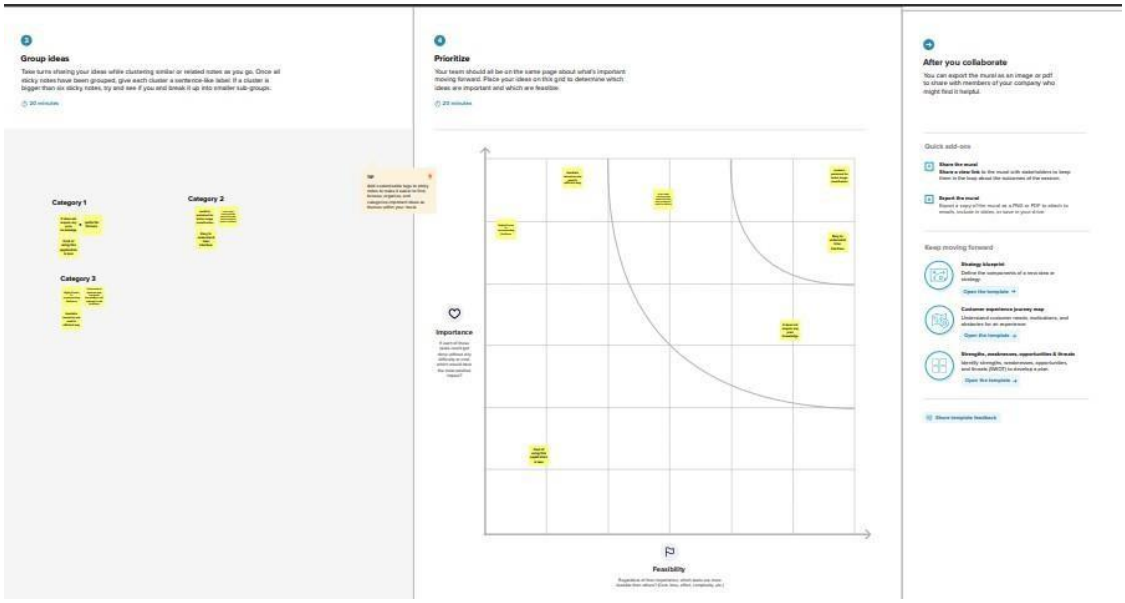


3.2 Ideation & Brainstorming

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 is built . Deep learning techniques are used to identify the diseases and suggest the fertilizes 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	PNT2022TMID15637
Project Name	Fertilizers Recommendation System for disease prediction
Maximum Marks	2 Marks

Proposed Solution Template:

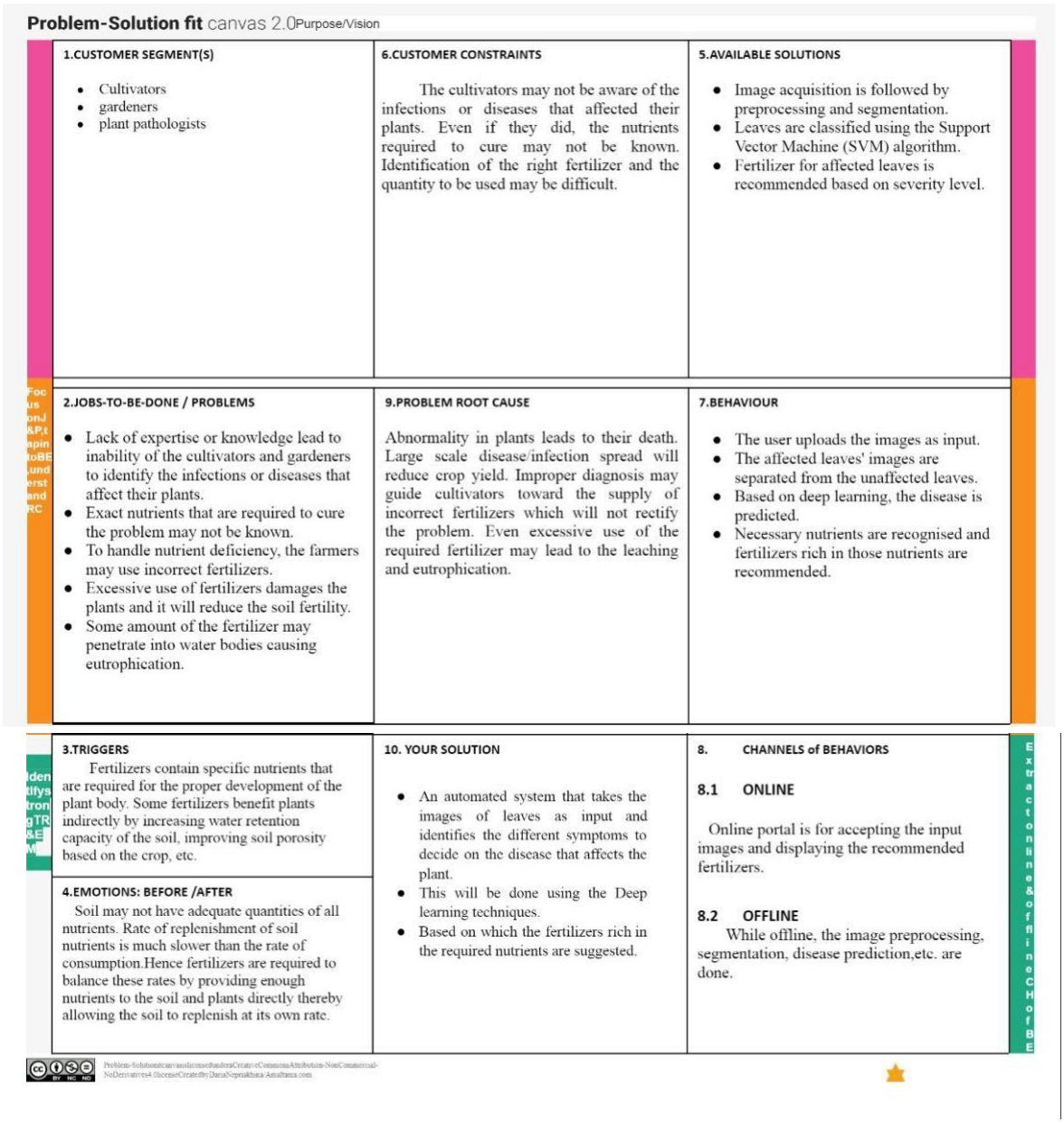
Project team shall fill the following information in proposed solution template.

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/carelessness, Lack of

		professional assistance, Inaccessibility to agro-tech solutions.Most of the diseases are detected in later stage that to manually which is time consuming and results in heavy loss so it is important to build an automated system that detects disesase at early stage and provides fertilizer recommendation accordingly.
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
		and the fertilizers that is required for 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.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.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.



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)

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	Registering through Gmail
FR-2	User confirmation	Confirmation is done through Email
FR-2	Image Capture	Take a picture of a leaf and verify that the leaf was captured using the specified criteria.
FR-3	Image Processing	Upload the image of the leaf for detecting the diseases that is present in the leaf.
FR-4	Leaf Prediction	Determine the parameter that should be taken into account for disease identification for identifying the leaf and predicting the disease in it.
FR-5	Image Description	Show the prescribed fertilizer that has to be used for the diseased leaf
FR-6	Providing Dataset	Training the datasets Testing the datasets
FR-7	Adding Datasets	Datasets for fruits and vegetables are added.

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.

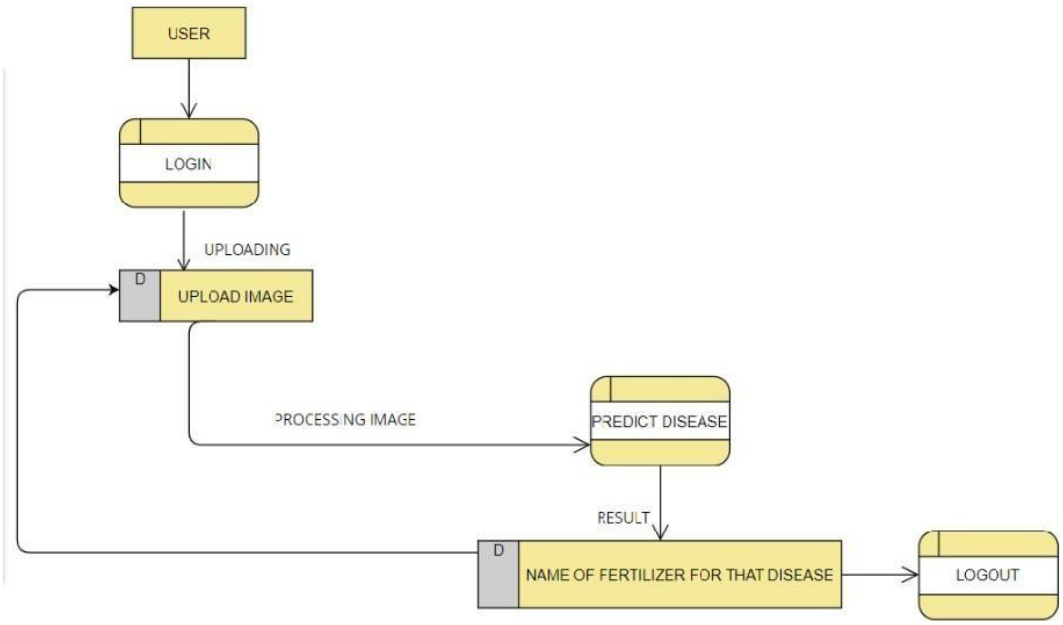
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Data sets can be prepared according to the leaf .Leaf datasets can be used for detection of all kind of leaf's Datasets can be reusable to detect diseases present in leaf.
NFR-2	**Security**	User information and leaf data are secured The 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.

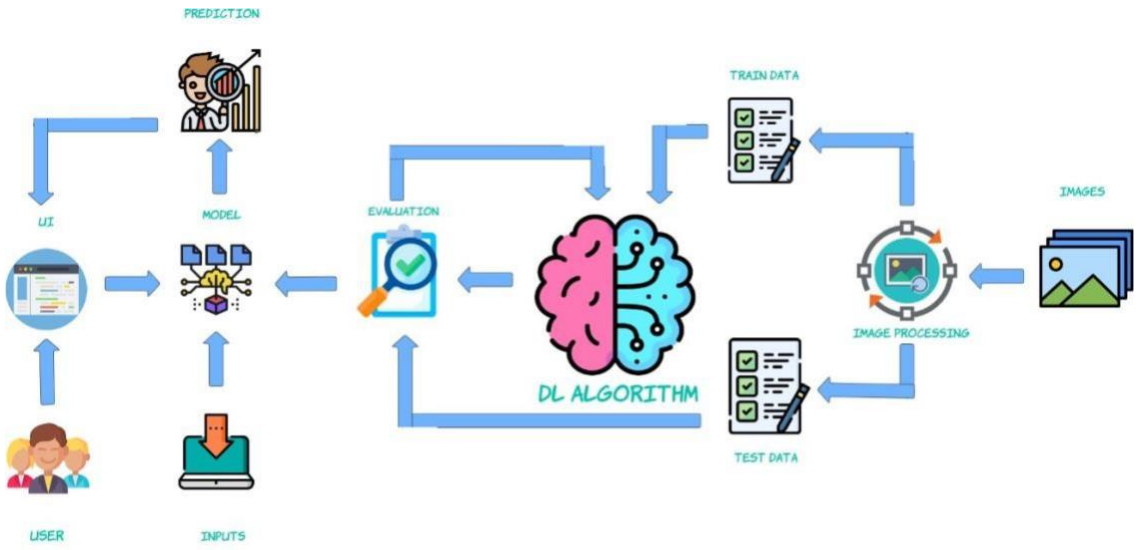
Data Flow Diagrams:



5.2 Solution & Technical Architecture

Solution Architecture:

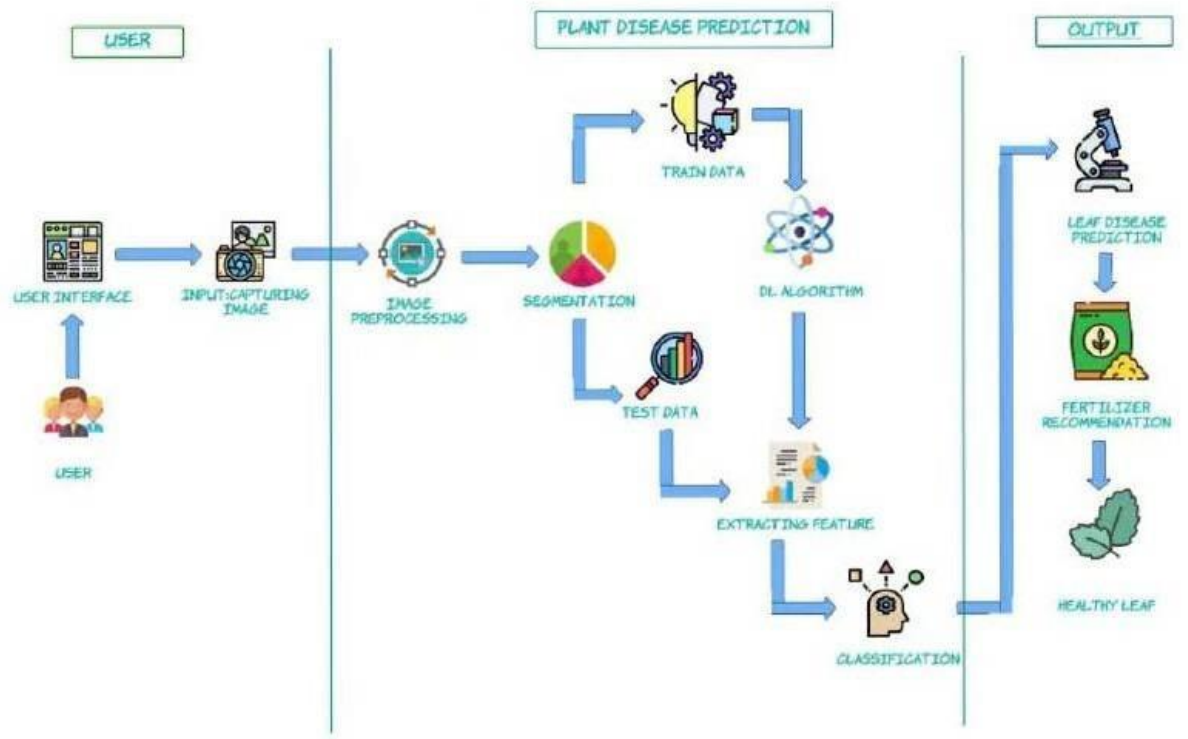
Solution architecture is the process of developing solutions based on predefined processes, guidelines and best practices with the objective that the developed solution fits within the enterprise architecture in terms of information architecture, system portfolios, integration requirements, etc.



FERTILIZERS RECOMMENDATION SYSTEM FOR DISEASE PREDICTION

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 donein 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 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:

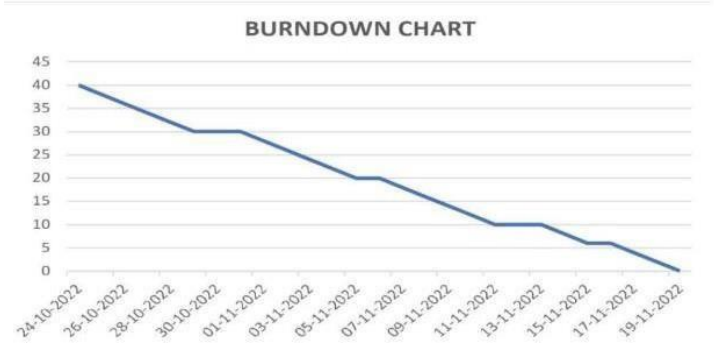
Sprint 1 Average Velocity:
Average Velocity = 20/2 = 10

Sprint 2 Average Velocity:
Average Velocity = 20/2 = 10

Sprint 3 Average Velocity:
Average Velocity = 20/1 = 20

Sprint 4 Average Velocity:
Average Velocity = 20/2 = 10

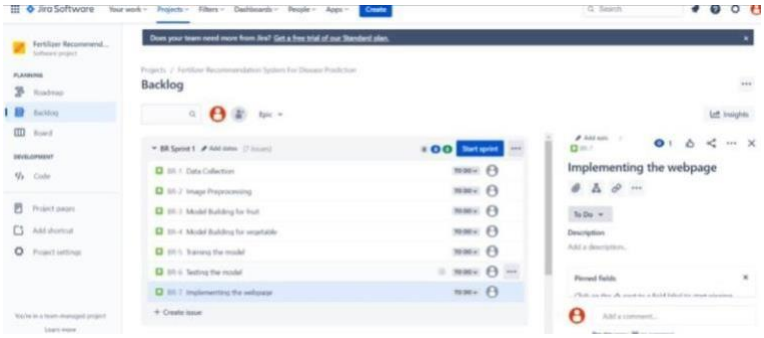
Burndown Chart:



6.3 Reports from JIRA

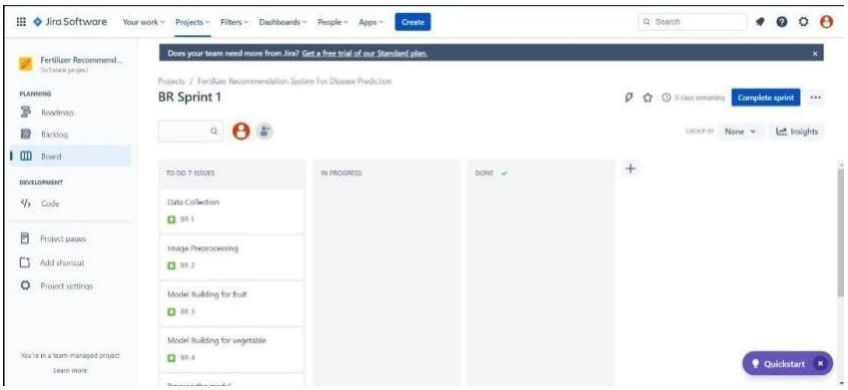
Backlog:

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.



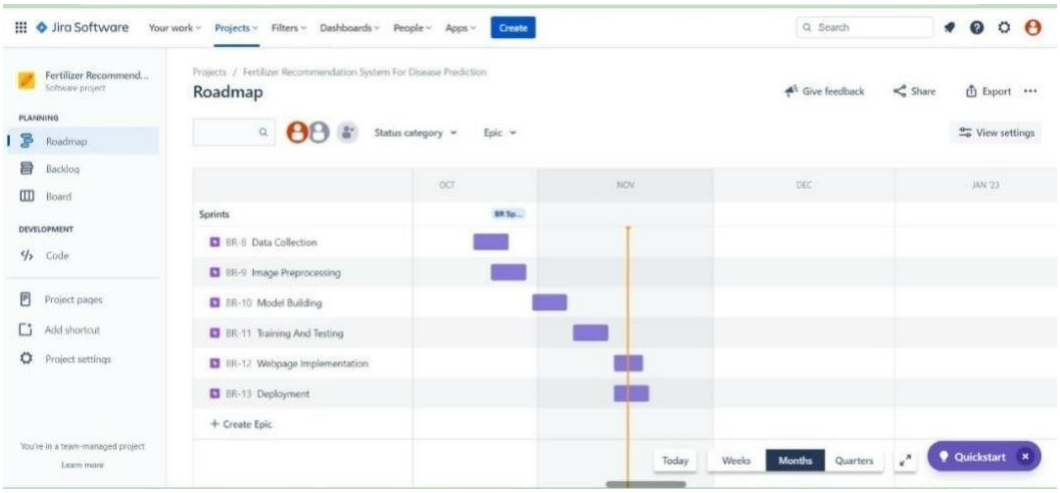
Board:

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 of 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:

```
import os
import numpy as np
import pandas as pd
from tensorflow.keras.models import load_model
# from tensorflow.keras.preprocessing import image
from werkzeug.utils import secure_filename

from flask import Flask, render_template, request

app = Flask(__name__)

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

#home page
```

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

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

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

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

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

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


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

if_name_== "_main_":
    app.run(debug=False)
s
```

Feature 1:

home.html:

```
<!DOCTYPE html>
<html >

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title> Plant Disease Prediction</title>
  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet'
type='text/css'>
<link href='https://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'>
<script type="text/javascript" src="https://gc.kis.v2.scr.kaspersky-
labs.com/FD126C42-EBFA-4E12-B309-
BB3FDD723AC1/main.js?attr=AMFGethlf4Q6r2IdpTrTqcDQGNLDU5Cbc3diYnUdLkg5mQrVB_td22O
HUAsBJSd0oo8OR0zM3rIPeFWfnEY4XCxQu4KOxMSqlshEoIBOzvYw0SsMYpyUv4fnvKEjmJoj_Y6cI4ov
-6AMOkz3Sh3epkfq0gltfnAPvvQBRdXqRmdqePVjlvvqL28ONZCiS0Qr5t0XGxJ0bSiWVT-
rH3cqaKCk05eP1Dx04mieTcjsA_TtFLx15PUu0ed6soaj-FOO6-
1d4OQxbJYBXUBefiUhzmOYCpsGIs1OyQvA0huo8AUYwYB72dvs07U3O2hq8BmYBv98h13sSo8iXKxyKx4
FUsOMkixjxYP6hu0wwi7yv1E2rei3GHtPl5YwHkWioQIPqvAmrlmaPtFZmF-
jE4_UUCi9IEKws8IduDiqQIFkxfO3YT_sUC9gWmxKSpGbiebwCgV-
wvdGEnbUxY18p9Db6jC6FVKRhqdMBianq63qv-
zZRMZbEpjzQT0DQAH3Yho4o4A00FIW2004q8Q80xt2kV928P_nBgS9HOgHI5EZxenbjfqANTs1rh8GGhB
d7RJaE8-
2AaqT6zbLf2tILJ8j4fk3bV1qsdw0fPmp6foJbDu4343XH36a0VGHsMLeVqcc30PSsE1pJbGE4_C_ExQd
0_uRSA40mRjnFwHdLo9SJc1qghyc5YGQil_utG48olMy9cC6z-iyKg1EeLKB43u-
q4SlUimRnuUsZW7drNWaijSfJPDmkm7lUJ0POwQXPfnLa2_spc3FisWCOZ7dFuIgDciIu0yF8rio2X0Pz
6pZkGQW4Fwl6vWKrLplmHagJElKXg58YSWwAT2DILilBjuSPiTwCHR9Ya_mAXW4C03v7xzJlaSK9jneEC
qctvKnH3RFgDS8ocfDcY65lXNRkq6v1hrcdv5sM2ek4Kjq4OFgX-wijr-0JdpSDpZlbIK00sPb4-
u1B8c7MaCqBcbJAhfmg4utLU67fn5GLoCX_-5TAWV0ID-_sC1Vs9glWRPkKmmktJMbVy98XqC5-
DhtE3yd5I9ZM1SEH1gGYLlRjxwzPjWwHE-YH1Nx9lm-
Esq27TK7M86uT8iAe7LgtviO2YsCB0buShHWmjh3RzwMGqNqeymFSxPRK_sDmTFoVjcaYpGa0kaMwhmmF
```

```
9AtPwGmFaGglv3rryVg0X0bGoXRetnrPpDG7jUoq5zQuXQSedBf9hmNwEqWsSZtI4zNTxjiEkxU0djhPX
qByZbnelp_3z6pqqniLzqj9jzAkvX6wDOW7ZycfDzOt-
zNgTxWdtf41P6ZjVu8EWSf65Wqgen5jD4IPXgXGtxkjrSbrqiX-
NxxxfKVJUOoOcEO0F6n3DWD0BMWS8UGOQO8gZZeXCfpuTIGYTD6okyD91kLk5AmhaNTJVKjkHO-
dHZqMHxikVhdK6C2PIfg4lEY0yuE3Fjj_5NNX5ZalIpOl3LN6YQ8Jqis_UmC_OXmjW2F5Y4p8VRRKc1HW
2DFaUxBrEgfSwe_keyaofodrjde_pfPuDQDryEgGy9DNIhpGUV_bQJ8jlPxRL7WSpmPU7-
IZ1mVN_onhqq2oI-WTl7ep-8w0GsJH3OhSRyyJC0XC9xtetqVjIHzcbKYFsxOaXT-
LLe7U9oHaXHzjDK3hn-ZNFYwzV_aoq8180eb" charset="UTF-8"></script><style>
.header {
      top:0;
      margin:0px;
      left: 0px;
      right: 0px;
      position: fixed;
      background-color: #28272c;
      color: white;
      box-shadow: 0px 8px 4px grey;
      overflow: hidden;
      padding-left:20px;
      font-family: 'Josefin Sans';
      font-size: 2vw;
      width: 100%;
      height:8%;
      text-align: center;
    }
    .topnav {
  overflow: hidden;
  background-color: #333;
}

.topnav-right a {
  float:  left;
  color: #f2f2f2;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 18px;
}

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

.topnav-right a.active {
```

```
  background-color: #565961;
  color: white;
}

.topnav-right {
  float: right;
  padding-right:100px;
}

body {

  background-color:#ffffff;
  background-repeat: no-repeat;
  background-size:cover;
  background-position: 0px 0px;
  }
  .button {
  background-color: #28272c;
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  border-radius: 12px;
}
.button:hover {
  box-shadow: 0 12px 16px 0 rgba(0,0,0,0.24), 0 17px 50px 0 rgba(0,0,0,0.19);
}
form {border: 3px solid #f1f1f1; margin-left:400px;margin-right:400px;}

input[type=text], input[type=password] {
  width: 100%;
  padding: 12px 20px;
  display: inline-block;
  margin-bottom:18px;
  border: 1px solid #ccc;
  box-sizing: border-box;
}

button {
  background-color: #28272c;
  color: white;
  padding: 14px 20px;
```

```
  margin-bottom:8px;
  border: none;
  cursor: pointer;
  width: 15%;
  border-radius:4px;
}

button:hover {
  opacity: 0.8;
}

.cancelbtn {
  width: auto;
  padding: 10px 18px;
  background-color: #f44336;
}

.imgcontainer {
  text-align: center;
  margin: 24px 0 12px 0;
}

img.avatar {
  width: 30%;
  border-radius: 50%;
}

.container {
  padding: 16px;
}

span.psw {
  float: right;
  padding-top: 16px;
}

/* Change styles for span and cancel button on extra small screens */
@media screen and (max-width: 300px) {
  span.psw {
     display: block;
     float: none;
  }
  .cancelbtn {
     width: 100%;
  }
```

```
}

.home{
  margin:80px;

  width:  84%;
  height: 500px;
  padding-top:10px;
  padding-left: 30px;

}
.login{
  margin:80px;
  box-sizing: content-box;
  width: 84%;
  height: 420px;
  padding: 30px;
  border: 10px solid blue;
}
.left,.right{
 box-sizing: content-box;
 height: 400px;
 margin:20px;
 border: 10px solid blue;
}

.mySlides {display: none;}
img {vertical-align: middle;}

/* Slideshow container */
.slideshow-container {
  max-width: 1000px;
  position: relative;
  margin: auto;
}

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

```
}
/* The dots/bullets/indicators */
.dot {
  height: 15px;
  width: 15px;
  margin: 0 2px;
  background-color: #bbb;
  border-radius: 50%;
  display: inline-block;
  transition: background-color 0.6s ease;
}

.active {
  background-color: #717171;
}

/* Fading animation */
.fade {
  -webkit-animation-name: fade;
  -webkit-animation-duration: 1.5s;
  animation-name: fade;
  animation-duration: 1.5s;
}

@-webkit-keyframes fade {
  from {opacity: .4}
  to {opacity: 1}
}

@keyframes fade {
  from {opacity: .4}
  to {opacity: 1}
}

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

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

<div class="header">
```

```
 <div style="width:50%;float:left;font-size:2vw;text-align:left;color:white;
padding-top:1%">Plant Disease Prediction</div>
  <div class="topnav-right"style="padding-top:0.5%;">

    <a class="active" href="{{ url_for('home')}}">Home</a>
    <a href="{{ url_for('prediction')}}">Predict</a>
  </div>
</div>

<div style="background-color:#ffffff;">
<div style="width:60%;float:left;">
<div style="font-size:50px;font-family:Montserrat;padding-left:20px;text-
align:center;padding-top:10%;">
<b>Detect if your plant<br> is infected!!</b></div><br>
<div style="font-size:20px;font-family:Montserrat;padding-left:70px;padding-
right:30px;text-align:justify;">Agriculture is one of the major sectors worls
wide. Over the years it has developed and the use of new technologies and
equipment replaced almost all the traditional methods of farming. The plant
diseases effect the production. Identification of diseases and taking necessary
precautions is all done through naked eye, which requires labour and laboratries.
This application helps farmers in detecting the diseases by observing the spots
on the leaves, which inturn saves effort and labor costs.</div><br><br>
</div>
</div>
<div style="width:40%;float:right;"><br><br>
<img src="{{url_for('static',filename='images/12456.png')}}" style="max-
height:100%;max-width:100%;">

</div>
</div>

<div class="home">

<br>

</div>

<script>
var slideIndex = 0;
showSlides();

function showSlides() {
  var i;
  var slides = document.getElementsByClassName("mySlides");
  var dots = document.getElementsByClassName("dot");
```

```
  for (i = 0; i < slides.length; i++) {
    slides[i].style.display = "none";
  }
  slideIndex++;
  if (slideIndex > slides.length) {slideIndex = 1}
  for (i = 0; i < dots.length; i++) {
    dots[i].className = dots[i].className.replace(" active", "");
  }
  slides[slideIndex-1].style.display = "block";
  dots[slideIndex-1].className += " active";
  setTimeout(showSlides, 2000); // Change image every 2 seconds
}
</script>
</body>
</html>
```

Feature 2 :

Predict.html:

```
<!DOCTYPE html>
<html >

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title> Plant Disease Prediction</title>
  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet'
type='text/css'>
<link href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css"
rel="stylesheet">
    <script type="text/javascript" src="https://gc.kis.v2.scr.kaspersky-
labs.com/FD126C42-EBFA-4E12-B309-
BB3FDD723AC1/main.js?attr=3wvf44XdejigWHFj22ANQmgfA-L5oa67wZhZwPtEITSot6t8o-
DPZwNcHRFhpa2tgGpDJGis4-1IHYyxyIAN2GE0-kSZKkCLRkbKttCLVN9mKhGFVtGJ3auoiiByn_jJ-
mA447x4TmdjGgz8XvMdLSPF4Gu5xwt0joGxWDXuOEF18Sa5usZGgj4TdDiTfDHpElX3P1eH-
lsevFhUJQEZe3981VXjRKYRn2FrxsYwXGSMBn0sRR9IYup35XYNQkvA6DLQV1lwLc4XuAo0BlJYAfI75R
4O5LwTWuT-uaft0DEQeuV_f3rKvkrcBkalcpWnyXVLeLyjMz5CqpZ1aSCy1MgVAzWxGb-
GX3eQb0F5qOksANddV_vhz1Ai4RgptuAfB8mVyuz0nWZzpmwam34lc4NL4tfyWGncKz2taMyGfsK4Mrn0
zfPlY9_n9FP0lMlAX0IQ8TfbVp4B1vbwnA-
```

```
RVJq8mxoTjgMgqhKhp6NQY_8gZULkbqqA0pqUMvfL3_fZC1PFipLNjCyCGe9YOaU9L7QF4CXeKsRhJXmI
898FhpxB1oI7z0xvndsDLPRsqbNuse_eGL9tz0Te5HLGhtoXSn5O8pHC99_XHYofrlismcByzZlmVqVkC
NfmbnMjaD9IQf6xAACyjkQ927AOvyDVCZKr-
tV6wRZyv_z7Z1J9AG7SGSLoB34AkMytkYXvpgGn21pGFNhvl3YSmyKYc2XJs89zHbp5fSyXsfasogSEYL
bpxCmuvzZKO4haaqouKDcLwBGMFp_Br095f-
AlhhWOdPDx1ezvTMx1NgS4QO97OmbyQCqHUFWWZLYNgjQ8zpfdBXB17L_v_lfmrUWhUiUVc9tRcJy-
lpchFJe8Gz7TUOKCRDjbIWtiqXryDeENrJgQ31laXp-
VVYpOI1L55pek2fgk5OCGNzVges5oG4PpMyCIXtJpv32E5rlPTktG4hD8eXmYQECVU1HvSmEiKvuY6T6i
9wdpqg_AnycRzUXmYdahFT3W7zToIn2RXzNfdOU0zbYBvtJ70TpR4PjfU75lJ0FsnphDuCnero3UYOak7
vYvGYD9YV2md5v-3AmP-eOor2m55JZRH_Hxpn28x-nDNCOHqVBC6leYuYFBVV_vL5l-
E8n92uWUqwMEzdZPZtAyRaCfz3D2Y0IYn-
ZrnfNTg2M_zVJePmUu1xdjYh7d1dx7nwclm7wJrBPb3JnX2kvEGYs9SM17MlwzoY1VJq4UzJ2D6oEvhQw
HvG4e1etlS6iLWzhy8RVMfBlTa4DPDOHmTlHhsKbn0UaMyFFCppe79rtIVRctcomnVmQysUwUOhjzlAq3
0-hXJCTqdCWJe2xnxjAuUHVqHSiHiZllZaoOWNCV5Ypx_eqzn-KyZS3u-
2_hGLHHNA2AVBWn_hF3Gz16dw6zA4QSmWZSfDUcNObLJGOSTaDS3Z8jPTloYPFmu8oES6TL1dLlEK5Yhc
SGaX4iv6o95drsZGb6bBcWgT7sNFHW6dVE9wdjoDFuBergPIAm0sKaZQ2Ex6j15OWCbE6UaPg-
VNfziA2FEPpJaI9hEPI2gdaSuHqovlEOt5mjuFBBOxpK0t8kOZRtsVzqUuJw3VcLjaP6SfG_KZfgX_g8T
Ps6CcFhlLRz63oXMQFPW6AA7eudWfygndazedq5B-
6DqSkOT04GTUJNqLcElg6KEEWqxd88BzoQoK28jrAf-xWHNIZv5HmQQYEnyX0U_cW8HX-
hde54TuY_fY3e5QYu4be-JxTkA4JxWLEagSa7-zs" charset="UTF-8"></script><script
src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>
    <script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>
    <script
src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
<link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300'
rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Merriweather'
rel='stylesheet'>
<link href='https://fonts.googleapis.com/css?family=Josefin Sans'
rel='stylesheet'>
<link href='https://fonts.googleapis.com/css?family=Montserrat' rel='stylesheet'>
<link href="{{ url_for('static', filename='css/final.css') }}" rel="stylesheet">
<style>
.header {
            top:0;
            margin:0px;
            left: 0px;
            right: 0px;
            position: fixed;
            background-color: #28272c;
            color: white;
            box-shadow: 0px 8px 4px grey;
            overflow: hidden;
            padding-left:20px;
            font-family: 'Josefin Sans';
```

```
            font-size: 2vw;
            width: 100%;
            height:8%;
            text-align: center;
        }
        .topnav {
  overflow: hidden;
  background-color: #333;
}

.topnav-right a {
  float:  left;
  color: #f2f2f2;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 18px;
}

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

.topnav-right a.active {
  background-color: #565961;
  color: white;
}

.topnav-right {
  float: right;
  padding-right:100px;
}

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

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

```
    margin-top:100px;
}

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


</style>
</head>

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

<div class="header">
 <div style="width:50%;float:left;font-size:2vw;text-align:left;color:white;
padding-top:1%">Plant Disease Prediction</div>
  <div class="topnav-right" style="padding-top:0.5%;">


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

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

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

                      <option value="select" selected>Select plant type</option>
                      <option value="fruit">Fruit</option>
                      <option value="vegetable">Vegetable</option>
        </select><br>


                <label for="imageUpload" class="upload-label" style="background:
#28272c;">
                    Choose...
                </label>
                <input type="file" name="image" id="imageUpload" accept=".png,
.jpg, .jpeg">


            </form>


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

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

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

        </div>
```

```
            </div>

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

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

final.css:

```
.img-preview {
    width: 256px;
    height: 256px;
    position: relative;
    border: 5px solid #F8F8F8;
    box-shadow: 0px 2px 4px 0px rgba(0, 0, 0, 0.1);
    margin-top: 1em;
    margin-bottom: 1em;
}

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

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

.upload-label{
    display: inline-block;
    padding: 12px 30px;
    background: #28272c;
    color: #fff;
    font-size: 1em;
```

```
    transition: all .4s;
    cursor: pointer;
}

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

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

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

main.js:

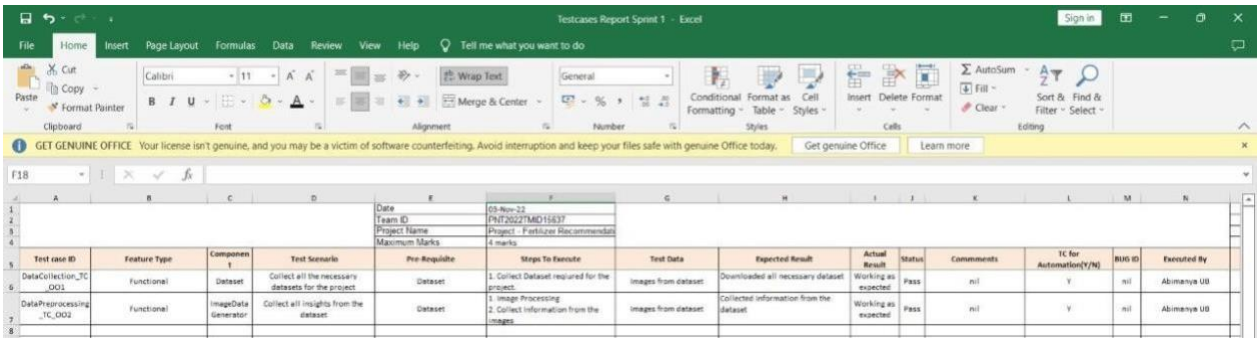
```
$(document).ready(function () {
    // Init
    $('.image-section').hide();
    $('.loader').hide();
    $('#result').hide();

    // Upload Preview
    function readURL(input) {
        if (input.files && input.files[0]) {
            var reader = new FileReader();
            reader.onload = function (e) {
                $('#imagePreview').css('background-image', 'url(' +
e.target.result + ')');
                $('#imagePreview').hide();
                $('#imagePreview').fadeIn(650);
            }
            reader.readAsDataURL(input.files[0]);
        }
    }
```

```
    $("#imageUpload").change(function () {
        $('.image-section').show();
        $('#btn-predict').show();
        $('#result').text('');
        $('#result').hide();
        readURL(this);
    });

    // Predict
    $('#btn-predict').click(function () {
        var form_data = new FormData($('#upload-file')[0]);

        // Show loading animation
        $(this).hide();
        $('.loader').show();

        // Make prediction by calling api /predict
        $.ajax({
            type: 'POST',
            url: '/predict',
            data: form_data,
            contentType: false,
            cache: false,
            processData: false,
            async: true,
            success: function (data) {
                // Get and display the result
                $('.loader').hide();
                $('#result').fadeIn(600);
                $('#result').text('Prediction: '+data);
                console.log('Success!');
            },
        });
    });



});
```

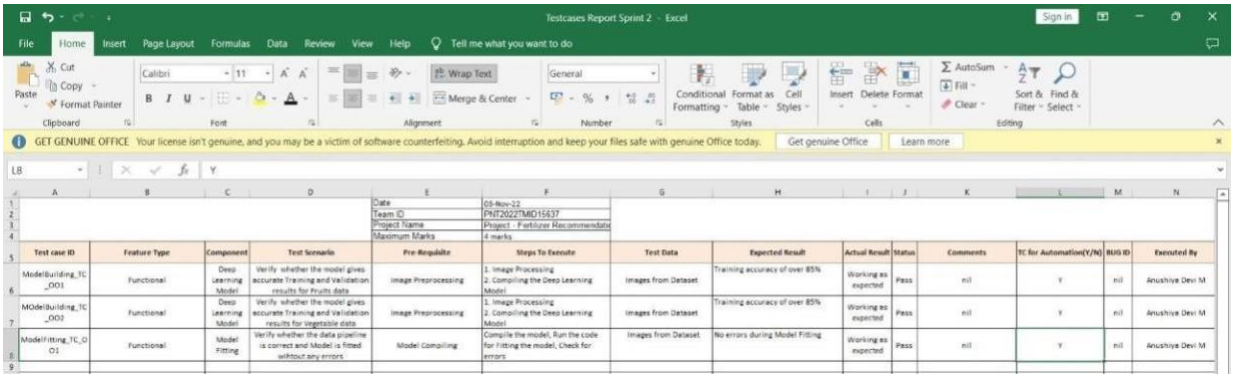
8. TESTING

8.1 Test Cases

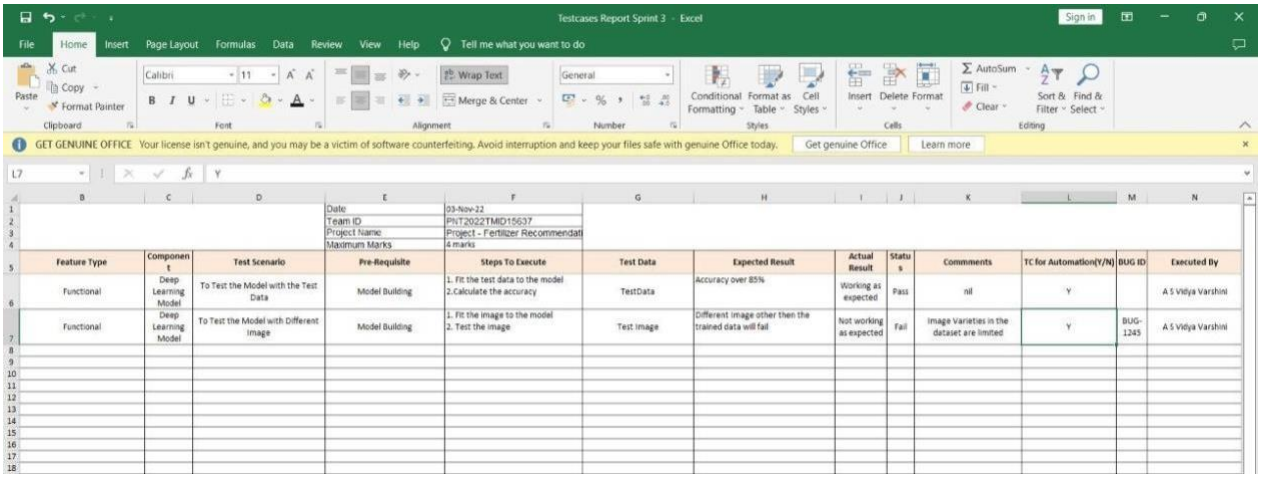
Test cases are a set of actions performed on a system to determine if it satisfies software requirements and functions correctly as it claimed to perform.



Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
DataCollection_TC_001	Functional	Dataset	Collect all the necessary datasets for the project	Dataset	1. Collect Dataset required for the project.	Images from dataset	Downloaded all necessary dataset	Working as expected	Pass	nil	Y	nil	Abimanya UB
DataPreprocessing_TC_002	Functional	ImageData Generator	Collect all insights from the dataset	Dataset	1. Image Processing 2. Collect Information from the images	Images from dataset	Collected information from the dataset	Working as expected	Pass	nil	Y	nil	Abimanya UB



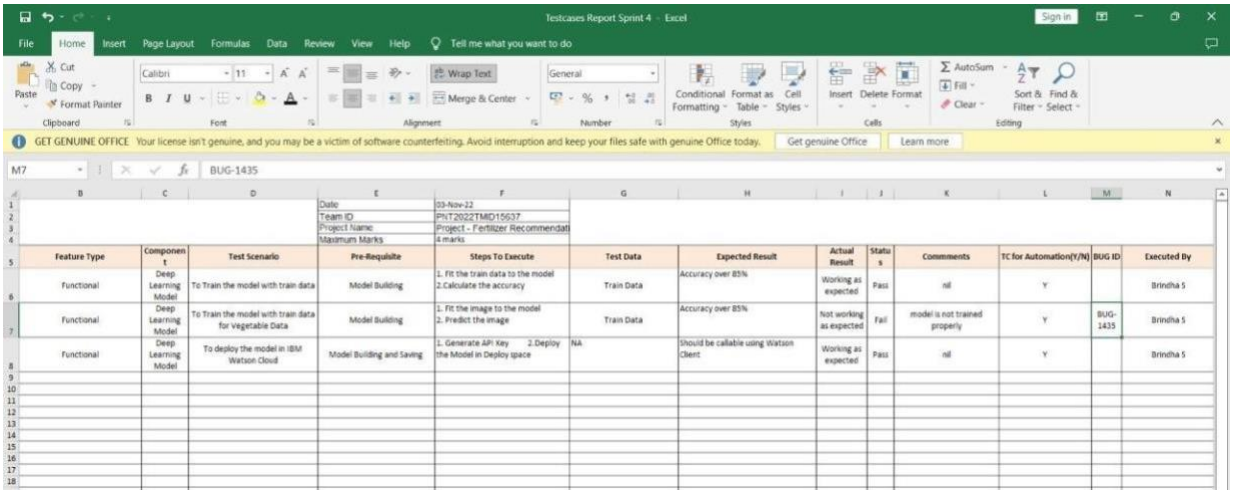
Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
ModelBuilding_TC_001	Functional	Deep Learning Model	Verify whether the model gives accurate Training and Validation results for fruits data	Image Preprocessing	1. Image Processing 2. Compiling the Deep Learning Model	Images from Dataset	Training accuracy of over 85%	Working as expected	Pass	nil	Y	nil	Anushiya Devi M
MOdelBuilding_TC_002	Functional	Deep Learning Model	Verify whether the model gives accurate Training and Validation results for Vegetable data	Image Preprocessing	1. Image Processing 2. Compiling the Deep Learning Model	Images from Dataset	Training accuracy of over 85%	Working as expected	Pass	nil	Y	nil	Anushiya Devi M
ModelFitting_TC_001	Functional	Model Fitting	Verify whether the data pipeline is correct and Model is fitted without any errors	Model Compiling	Compile the model, Run the code for fitting the model, Check for errors	Images from Dataset	No errors during Model Fitting	Working as expected	Pass	nil	Y	nil	Anushiya Devi M



Testcases Report Sprint 3 - Excel

Date	03-Nov-22
Team ID	PNT2022TMID15637
Project Name	Project - Fertilizer Recommendati
Maximum Marks	4 marks

Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
Functional	Deep Learning Model	To Test the Model with the Test Data	Model Building	1. Fit the test data to the model 2.Calculate the accuracy	TestData	Accuracy over 85%	Working as expected	Pass	nil	Y		A S Vidya Varshini
Functional	Deep Learning Model	To Test the Model with Different Image	Model Building	1. Fit the image to the model 2. Test the image	Test Image	Different image other then the trained data will fail	Not working as expected	Fail	Image Varieties in the dataset are limited	Y	BUG-1245	A S Vidya Varshini



Testcases Report Sprint 4 - Excel

Date	03-Nov-22
Team ID	PNT2022TMID15637
Project Name	Project - Fertilizer Recommendati
Maximum Marks	4 marks

Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
Functional	Deep Learning Model	To Train the model with train data	Model Building	1. Fit the train data to the model 2.Calculate the accuracy	Train Data	Accuracy over 85%	Working as expected	Pass	nil	Y		Brindha S
Functional	Deep Learning Model	To Train the model with train data for Vegetable Data	Model Building	1. Fit the image to the model 2. Predict the image	Train Data	Accuracy over 85%	Not working as expected	Fail	model is not trained properly	Y	BUG-1435	Brindha S
Functional	Deep Learning Model	To deploy the model in IBM Watson Cloud	Model Building and Saving	1. Generate API Key 2.Deploy the Model in Deploy space	NA	Should be callable using Watson Client	Working as expected	Pass	nil	Y		Brindha S

8.2 User Acceptance Testing

Before deploying the software application to a production environment the end user or client performs a type of testing known as user acceptance testing, or UAT to ensure whether the software functionalities serve the purpose of development.

Acceptance Testing
UAT Execution & Report Submission

Date	17 November 2022
Team ID	PNT2022TMID15637
Project Name	Fertilisers recommendation System for disease prediction
Maximum Marks	4 Marks

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
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	Fail	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

metrics are a baseline for performance tests. Monitoring the correct parameters will help you detect areas that require increased attention and find ways to improve them.

Project Development Phase
Model Performance Test

Date	10 November 2022
Team ID	PNT2022TMID15637
Project Name	Project - Fertilizers Recommendation System For Disease Prediction
Maximum Marks	10 Marks

Model Performance Testing:

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Total params: 45,221,754 Trainable params: 45,221,754 Non trainable params: 0	
2.	Accuracy	Training Accuracy – 97.55 Validation Accuracy – 96.45	

10. ADVANTAGES & DISADVANTAGES

Advantages:

- Early detection of plant diseases.
- Proper fertilizer recommendation to prevent or cure the plant infection or disease.
- No need to consult any specialists.
- Fully automated system.

Disadvantages:

- Requires training the system with large dataset.
- Works only on the pretrained diseases.
- When a plant is infected with multiple diseases the system may not predict all the diseases due to the mixed symptoms.
- Requires a good device connected to the internet.

11. CONCLUSION

Hence a system that takes in images as user input, analyses those for certain symptoms and identifies the disease, recommends the fertilizer to counter the deficiency of the nutrients is built and deployed.

12. FUTURE SCOPE

The system must be trained with numerous images of plant disease symptoms. In case of presence of multiple diseases, suitable classification must be done to predict each disease accurately and recommend separate fertilizers as a solution to each deficiency or infection.

13. APPENDIX

Source Code

Home.html:

```
<!DOCTYPE html>
<html >

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title> Plant Disease Prediction</title>
  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>
<link href='https://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'>
<script type="text/javascript" src="https://gc.kis.v2.scr.kaspersky-labs.com/FD126C42-EBFA-4E12-
B309-
BB3FDD723AC1/main.js?attr=AMFGethlf4Q6r2IdpTrTqcDQGNLDU5Cbc3diYnUdLkg5mQrVB_td
22OHUAsBJSd0oo8OR0zM3rIPeFWfnEY4XCxQu4KOxMSqlshEoIBOzvYw0SsMYpyUv4fnvKEjm
Joj_Y6cI4ov-
6AMOkz3Sh3epkfq0gltfnAPvvQBRdXqRmdqePVjlvvqL28ONZCiS0Qr5t0XGxJ0bSiWVT-
rH3cqaKCk05eP1Dx04mieTcjsA_TtFLx15PUu0ed6soaj-FOO6-
1d4OQxbJYBXUBefiUhzmOYCpsGIs1OyQvA0huo8AUYwYB72dvs07U3O2hq8BmYBv98h13sSo8
iXKxyKx4FUsOMkixjxYP6hu0wwi7yv1E2rei3GHtPl5YwHkWioQIPqvAmrlmaPtFZmF-
jE4_UUCi9IEKws8IduDiqQIFkxfO3YT_sUC9gWmxKSpGbiebwCgV-
wvdGEnbUxY18p9Db6jC6FVKRhqdMBianq63qv-
zZRMZbEpjzQT0DQAH3Yho4o4A00FIW2004q8Q80xt2kV928P_nBgS9HOgHI5EZxenbjfqANTs1r
h8GGhBd7RJaE8-
2AaqT6zbLf2tILJ8j4fk3bV1qsdw0fPmp6foJbDu4343XH36a0VGHsMLeVqcc30PSsE1pJbGE4_C_E
xQd0_uRSA40mRjnFwHdLo9SJc1qghyc5YGQil_utG48olMy9cC6z-iyKg1EeLKB43u-
q4SlUimRnuUsZW7drNWaijSfJPDmkm7lUJ0POwQXPfnLa2_spc3FisWCOZ7dFuIgDciIu0yF8rio2X
0Pz6pZkGQW4Fwl6vWKrLplmHagJElKXg58YSWwAT2DILilBjuSPiTwCHR9Ya_mAXW4C03v7x
zJlaSK9jneECqctvKnH3RFgDS8ocfDcY65lXNRkq6v1hrcdv5sM2ek4Kjq4OFgX-wijr-
0JdpSDpZlbIK00sPb4-u1B8c7MaCqBcbJAhfmg4utLU67fn5GLoCX_-5TAWV0ID-
_sC1Vs9glWRPkKmmktJMbVy98XqC5-DhtE3yd5I9ZM1SEH1gGYLlRjxwzPjWwHE-YH1Nx9lm-
Esq27TK7M86uT8iAe7LgtviO2YsCB0buShHWmjh3RzwMGqNqeymFSxPRK_sDmTFoVjcaYpGa0
kaMwhmmF9AtPwGmFaGglv3rryVg0X0bGoXRetnrPpDG7jUoq5zQuXQSedBf9hmNwEqWsSZtI4z
NTxjiEkxU0djhPXqByZbnelp_3z6pqqniLzqj9jzAkvX6wDOW7ZycfDzOt-
zNgTxWdtf41P6ZjVu8EWSf65Wqgen5jD4IPXgXGtxkjrSbrqiX-
NxxxfKVJUOoOcEO0F6n3DWD0BMWS8UGOQO8gZZeXCfpuTIGYTD6okyD91kLk5AmhaNTJV
KjkHO-
dHZqMHxikVhdK6C2PIfg4lEY0yuE3Fjj_5NNX5ZalIpOl3LN6YQ8Jqis_UmC_OXmjW2F5Y4p8VR
RKc1HW2DFaUxBrEgfSwe_keyaofodrjde_pfPuDQDryEgGy9DNIhpGUV_bQJ8jlPxRL7WSpmPU7
-IZ1mVN_onhqq2oI-WTl7ep-8w0GsJH3OhSRyyJC0XC9xtetqVjIHzcbKYFsxOaXT-
LLe7U9oHaXHzjDK3hn-ZNFYwzV_aoq8180eb" charset="UTF-8"></script><style>
.header {
                        top:0;
                        margin:0px;
```

```
                        left: 0px;
                        right: 0px;
                        position: fixed;
                        background-color: #28272c;
                        color: white;
                        box-shadow: 0px 8px 4px grey;
                        overflow: hidden;
                        padding-left:20px;
                        font-family: 'Josefin Sans';
                        font-size: 2vw;
                        width: 100%;
                        height:8%;
                        text-align: center;
                }
                .topnav {
  overflow: hidden;
  background-color: #333;
}

.topnav-right a {
  float: left;
  color: #f2f2f2;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 18px;
}

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

.topnav-right a.active {
  background-color: #565961;
  color: white;
}

.topnav-right {
  float: right;
  padding-right:100px;
}

body {

  background-color:#ffffff;
  background-repeat: no-repeat;
  background-size:cover;
  background-position: 0px 0px;
  }
  .button {
  background-color: #28272c;
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
```

```
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  border-radius: 12px;
}
.button:hover {
  box-shadow: 0 12px 16px 0 rgba(0,0,0,0.24), 0 17px 50px 0 rgba(0,0,0,0.19);
}
form {border: 3px solid #f1f1f1; margin-left:400px;margin-right:400px;}

input[type=text], input[type=password] {
  width: 100%;
  padding: 12px 20px;
  display: inline-block;
  margin-bottom:18px;
  border: 1px solid #ccc;
  box-sizing: border-box;
}

button {
  background-color: #28272c;
  color: white;
  padding: 14px 20px;
  margin-bottom:8px;
  border:  none;
  cursor: pointer;
  width: 15%;
  border-radius:4px;}

button:hover {
  opacity: 0.8;}

.cancelbtn {
  width: auto;
  padding: 10px 18px;
  background-color: #f44336;}

.imgcontainer {
  text-align: center;
  margin: 24px 0 12px 0;}

img.avatar {
  width: 30%;
  border-radius: 50%;}

.container {
  padding: 16px;}

span.psw {
  float: right;
  padding-top: 16px;}

/* Change styles for span and cancel button on extra small screens */
@media screen and (max-width: 300px) {
  span.psw {
     display: block;
```

```
    float: none;}

  .cancelbtn {
    width: 100%;}}

.home{
      margin:80px;
  width: 84%;
  height: 500px;
  padding-top:10px;
  padding-left: 30px;}

.login{
      margin:80px;
      box-sizing: content-box;
  width: 84%;
  height: 420px;
  padding: 30px;
  border: 10px solid blue;
}
.left,.right{
 box-sizing: content-box;
 height: 400px;
 margin:20px;
 border: 10px solid blue;
}

.mySlides {display: none;}
img {vertical-align: middle;}

/* Slideshow container */
.slideshow-container {
  max-width: 1000px;
  position: relative;
  margin: auto;
}

/* Caption text */
.text {
  color: #f2f2f2;
  font-size: 15px;
  padding: 8px 12px;
  position: absolute;
  bottom:  8px;
  width: 100%;
  text-align: center;
}
/* The dots/bullets/indicators */
.dot {
  height: 15px;
  width: 15px;
  margin: 0 2px;
  background-color: #bbb;
  border-radius: 50%;
  display: inline-block;
  transition: background-color 0.6s ease;
```

```
}

.active {
  background-color: #717171;
}

/* Fading animation */
.fade {
  -webkit-animation-name: fade;
  -webkit-animation-duration: 1.5s;
  animation-name: fade;
  animation-duration: 1.5s;
}

@-webkit-keyframes fade {
  from {opacity: .4}
  to {opacity: 1}
}

@keyframes fade {
  from {opacity: .4}
  to {opacity: 1}
}

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

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

<div class="header">
 <div style="width:50%;float:left;font-size:2vw;text-align:left;color:white; padding-top:1%">Plant Disease Prediction</div>
  <div class="topnav-right"style="padding-top:0.5%;">

    <a class="active" href="{{ url_for('home')}}">Home</a>
    <a href="{{ url_for('prediction')}}">Predict</a>
  </div>
</div>

<div style="background-color:#ffffff;">
<div style="width:60%;float:left;">
<div style="font-size:50px;font-family:Montserrat;padding-left:20px;text-align:center;padding-top:10%;">
<b>Detect if your plant<br> is infected!!</b></div><br>
<div style="font-size:20px;font-family:Montserrat;padding-left:70px;padding-right:30px;text-align:justify;">Agriculture is one of the major sectors worls wide. Over the years it has developed and the use of new technologies and equipment replaced almost all the traditional methods of farming. The plant diseases effect the production. Identification of diseases and taking necessary precautions is all done through naked eye, which requires labour and laboratries. This application helps farmers in detecting the diseases by observing the spots on the leaves, which inturn saves effort and labor costs.</div><br><br>
</div>
```

```
</div>
<div style="width:40%;float:right;"><br><br>
<img src="{{url_for('static',filename='images/12456.png')}}" style="max-height:100%;max-
width:100%;">
</div>
</div>
<div class="home">
<br>
</div>

<script>
var slideIndex = 0;
showSlides();

function showSlides() {
  var i;
  var slides = document.getElementsByClassName("mySlides");
  var dots = document.getElementsByClassName("dot");
  for (i = 0; i < slides.length; i++) {
    slides[i].style.display = "none";
  }
  slideIndex++;
  if (slideIndex > slides.length) {slideIndex = 1}
  for (i = 0; i < dots.length; i++) {
    dots[i].className = dots[i].className.replace(" active", "");
  }
  slides[slideIndex-1].style.display = "block";
  dots[slideIndex-1].className += " active";
  setTimeout(showSlides, 2000); // Change image every 2 seconds
}
</script>
</body>
</html>
```

Predict.html:

```
<!DOCTYPE html>
<html >

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title> Plant Disease Prediction</title>
  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>
<link href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css" rel="stylesheet">
    <script type="text/javascript" src="https://gc.kis.v2.scr.kaspersky-labs.com/FD126C42-EBFA-
4E12-B309-BB3FDD723AC1/main.js?attr=3wvf44XdejigWHFj22ANQmgfA-
L5oa67wZhZwPtEITSot6t8o-DPZwNcHRFhpa2tgGpDJGis4-1IHYyxyIAN2GE0-
kSZKkCLRkbKttCLVN9mKhGFVtGJ3auoiiByn_jJ-
mA447x4TmdjGgz8XvMdLSPF4Gu5xwt0joGxWDXuOEF18Sa5usZGgj4TdDiTfDHpElX3P1eH-
lsevFhUJQEZe3981VXjRKYRn2FrxsYwXGSMBn0sRR9IYup35XYNQkvA6DLQV1lwLc4XuAo0B
lJYAfI75R4O5LwTWuT-
```

uaft0DEQeuV_f3rKvkrcBkalcpWnyXVLeLyjMz5CqpZ1aSCy1MgVAzWxGb-
GX3eQb0F5qOksANddV_vhz1Ai4RgptuAfB8mVyuz0nWZzpmwam34lc4NL4tfyWGncKz2taMyGfs
K4Mrn0zfPlY9_n9FP0lMlAX0IQ8TfbVp4B1vbwnA-
RVJq8mxoTjgMgqhKhp6NQY_8gZULkbqqA0pqUMvfL3_fZC1PFipLNjCyCGe9YOaU9L7QF4CXe
KsRhJXmI898FhpxB1oI7z0xvndsDLPRsqbNuse_eGL9tz0Te5HLGhtoXSn5O8pHC99_XHYofrlismc
ByzZlmVqVkCNfmbnMjaD9IQf6xAACyjkQ927AOvyDVCZKr-
tV6wRZyv_z7Z1J9AG7SGSLoB34AkMytkYXvpgGn21pGFNhvl3YSmyKYc2XJs89zHbp5fSyXsfas
ogSEYLbpxCmuvzZKO4haaqouKDcLwBGMFp_Br095f-
AlhhWOdPDx1ezvTMx1NgS4QO97OmbyQCqHUFWWZLYNgjQ8zpfdBXB17L_v_lfmrUWhUiUV
c9tRcJy-lpchFJe8Gz7TUOKCRDjbIWtiqXryDeENrJgQ31laXp-
VVYpOI1L55pek2fgk5OCGNzVges5oG4PpMyCIXtJpv32E5rlPTktG4hD8eXmYQECVU1HvSmEiK
vuY6T6i9wdpqg_AnycRzUXmYdahFT3W7zToIn2RXzNfdOU0zbYBvtJ70TpR4PjfU75lJ0FsnphDu
Cnero3UYOak7vYvGYD9YV2md5v-3AmP-eOor2m55JZRH_Hxpn28x-
nDNCOHqVBC6leYuYFBVV_vL5l-E8n92uWUqwMEzdZPZtAyRaCfz3D2Y0IYn-
ZrnfNTg2M_zVJePmUu1xdjYh7d1dx7nwclm7wJrBPb3JnX2kvEGYs9SM17MlwzoY1VJq4UzJ2D6o
EvhQwHvG4e1etlS6iLWzhy8RVMfBlTa4DPDOHmTlHhsKbn0UaMyFFCppe79rtIVRctcomnVmQy
sUwUOhjzlAq30-hXJCTqdCWJe2xnxjAuUHVqHSiHiZllZaoOWNCV5Ypx_eqzn-KyZS3u-
2_hGLHHNA2AVBWn_hF3Gz16dw6zA4QSmWZSfDUcNObLJGOSTaDS3Z8jPTloYPFmu8oES6T
L1dLlEK5YhcSGaX4iv6o95drsZGb6bBcWgT7sNFHW6dVE9wdjoDFuBergPIAm0sKaZQ2Ex6j15O
WCbE6UaPg-
VNfziA2FEPpJaI9hEPI2gdaSuHqovlEOt5mjuFBBOxpK0t8kOZRtsVzqUuJw3VcLjaP6SfG_KZfgX_
g8TPs6CcFhlLRz63oXMQFPW6AA7eudWfygndazedq5B-
6DqSkOT04GTUJNqLcElg6KEEWqxd88BzoQoK28jrAf-xWHNIZv5HmQQYEnyX0U_cW8HX-
hde54TuY_fY3e5QYu4be-JxTkA4JxWLEagSa7-zs" charset="UTF-8"></script><script
src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>
<script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>
<script src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
<link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet'
type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Merriweather' rel='stylesheet'>
<link href='https://fonts.googleapis.com/css?family=Josefin Sans' rel='stylesheet'>
<link href='https://fonts.googleapis.com/css?family=Montserrat' rel='stylesheet'>
<link href="{{ url_for('static', filename='css/final.css') }}" rel="stylesheet">
<style>
.header {
top:0;
margin:0px;
left: 0px;
right: 0px;
position: fixed;
background-color: #28272c;
color: white;
box-shadow: 0px 8px 4px grey;
overflow: hidden;
padding-left:20px;
font-family: 'Josefin Sans';
font-size: 2vw;
width: 100%;
height:8%;
text-align: center;
}
.topnav {
overflow: hidden;
background-color: #333;
}

```
.topnav-right a {
  float: left;
  color: #f2f2f2;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 18px;
}

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

.topnav-right a.active {
  background-color: #565961;
  color: white;
}

.topnav-right {
  float: right;
  padding-right:100px;
}

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

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

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

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


</style>
</head>

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

<div class="header">
 <div style="width:50%;float:left;font-size:2vw;text-align:left;color:white; padding-top:1%">Plant
Disease Prediction</div>
  <div class="topnav-right" style="padding-top:0.5%;">


  </div>
</div>
<div class="container">
    <div id="content" style="margin-top:2em">
        <div class="container">
          <div class="row">
            <div class="col-sm-6 bd" >

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

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



            <div class="image-section" style="display:none;">
                <div class="img-preview">
                    <div id="imagePreview">
                    </div>
                </div>
                <div>
```

```
                                        <button type="button" class="btn btn-info btn-lg " id="btn-
predict" style="background: #28272c;">Predict!</button>
                                    </div>
                            </div>
                            <div class="loader" style="display:none;"></div>
                            <h3>
                            <span id="result" style="font-size:17px; "> </span>
                            </h3>
                    </div>
                            </div>

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


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

main.js:

```
$(document).ready(function () {
    // Init
    $('.image-section').hide();
    $('.loader').hide();
    $('#result').hide();

    // Upload  Previewfunction
    readURL(input) {
        if (input.files && input.files[0]) {
            var reader = new FileReader();
            reader.onload = function (e) {
                $('#imagePreview').css('background-image', 'url(' + e.target.result + ')');
                $('#imagePreview').hide();
                $('#imagePreview').fadeIn(650);
            }
            reader.readAsDataURL(input.files[0]);
        }
    }
    $("#imageUpload").change(function () {
        $('.image-section').show();
        $('#btn-predict').show();
        $('#result').text('');
        $('#result').hide();
        readURL(this);
    });

    // Predict
    $('#btn-predict').click(function () {
        var form_data = new FormData($('#upload-file')[0]);

        // Show loading animation
        $(this).hide();
```

```
        $('.loader').show();

        // Make prediction by calling api /predict
        $.ajax({
            type: 'POST',
            url: '/predict',
            data: form_data,
            contentType: false,
            cache: false,
            processData: false,
            async: true,
            success: function (data) {
                // Get and display the result
                $('.loader').hide();
                $('#result').fadeIn(600);
                $('#result').text('Prediction: '+data);
                console.log('Success!');
            },
        });
    });

});
```

Final.css:

```
.img-preview {
    width: 256px;
    height: 256px;
    position: relative;
    border: 5px solid #F8F8F8;
    box-shadow: 0px 2px 4px 0px rgba(0, 0, 0, 0.1);
    margin-top: 1em;
    margin-bottom: 1em;
}

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

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

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

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

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

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

Python – app.py:

```
import os
import numpy as np
import pandas as pd
from tensorflow.keras.models import load_model
# from tensorflow.keras.preprocessing import image
from werkzeug.utils import secure_filename

from flask import Flask, render_template, request

app = Flask(__name__)

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

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

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

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

      # Save the file to ./uploads
      basepath = os.path.dirname(__file__)
      file_path = os.path.join(
```

```
        basepath, 'uploads', secure_filename(f.filename))
    f.save(file_path)
    img = image.load_img(file_path, target_size=(128, 128))

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

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


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

if__name__== "_main_":
  app.run(debug=False)
```

DEPLOYMENT MODEL CODE:

Fruit model:

```
ls
sample_data/
pwd
'/home/wsuser/work'
!pip install keras==2.7.0
!pip install tensorflow==2.5.0
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab
wheels/public/simple/
Requirement already satisfied: keras==2.7.0 in /usr/local/lib/python3.7/dist-packages (2.7.0)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab
wheels/public/simple/
Requirement already satisfied: tensorflow==2.5.0 in /usr/local/lib/python3.7/dist-packages
 (2.5.0)
Requirement already satisfied: h5py~=3.1.0 in /usr/local/lib/python3.7/dist-packages (from
 tensorflow==2.5.0) (3.1.0)
Requirement already satisfied: protobuf>=3.9.2 in /usr/local/lib/python3.7/dist-packages (from
tensorflow==2.5.0) (3.19.6)
Requirement already satisfied: typing-extensions~=3.7.4 in /usr/local/lib/python3.7/dist packages
(from tensorflow==2.5.0) (3.7.4.3)
```

Requirement already satisfied: keras-nightly~=2.5.0.dev in /usr/local/lib/python3.7/dist packages (from tensorflow==2.5.0) (2.5.0.dev2021032900)
Requirement already satisfied: flatbuffers~=1.12.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0) (1.12)
Requirement already satisfied: gast==0.4.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0) (0.4.0)
Requirement already satisfied: absl-py~=0.10 in /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0) (0.15.0)
Requirement already satisfied: astunparse~=1.6.3 in /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0) (1.6.3)
Requirement already satisfied: tensorflow-estimator<2.6.0,>=2.5.0rc0 in /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0) (2.5.0) Requirement already satisfied: tensorboard~=2.5 in /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0) (2.9.1)
Requirement already satisfied: opt-einsum~=3.3.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0) (3.3.0)
Requirement already satisfied: six~=1.15.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0) (1.15.0)
Requirement already satisfied: google-pasta~=0.2 in /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0) (0.2.0)
Requirement already satisfied: grpcio~=1.34.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0) (1.34.1)
Requirement already satisfied: wrapt~=1.12.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0) (1.12.1)
Requirement already satisfied: termcolor~=1.1.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0) (1.1.0)
Requirement already satisfied: keras-preprocessing~=1.1.2 in /usr/local/lib/python3.7/dist packages (from tensorflow==2.5.0) (1.1.2)
Requirement already satisfied: wheel~=0.35 in /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0) (0.38.3)
Requirement already satisfied: numpy~=1.19.2 in /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0) (1.19.5)
Requirement already satisfied: cached-property in /usr/local/lib/python3.7/dist-packages (from h5py~=3.1.0->tensorflow==2.5.0) (1.5.2)
Requirement already satisfied: google-auth<3,>=1.6.3 in /usr/local/lib/python3.7/dist packages (from tensorboard~=2.5->tensorflow==2.5.0) (2.14.1)
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in /usr/local/lib/python3.7/dist-packages (from tensorboard~=2.5->tensorflow==2.5.0) (0.6.1)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in /usr/local/lib/python3.7/dist-packages (from tensorboard~=2.5->tensorflow==2.5.0) (1.8.1)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in /usr/local/lib/python3.7/dist-packages (from tensorboard~=2.5->tensorflow==2.5.0) (0.4.6)
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.7/dist-packages (from tensorboard~=2.5->tensorflow==2.5.0) (1.0.1)
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.7/dist-packages (from tensorboard~=2.5->tensorflow==2.5.0) (3.4.1)

Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.7/dist packages (from tensorboard~=2.5->tensorflow==2.5.0) (2.23.0)
Requirement already satisfied: setuptools>=41.0.0 in /usr/local/lib/python3.7/dist-packages (from tensorboard~=2.5->tensorflow==2.5.0) (57.4.0)
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.7/dist-packages (from google-auth<3,>=1.6.3->tensorboard~=2.5->tensorflow==2.5.0) (4.9) Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.7/dist packages (from google-auth<3,>=1.6.3->tensorboard~=2.5->tensorflow==2.5.0) (0.2.8) Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.7/dist packages (from google-auth<3,>=1.6.3->tensorboard~=2.5->tensorflow==2.5.0) (5.2.0) Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.7/dist packages (from google-auth-oauthlib<0.5,>=0.4.1->tensorboard~=2.5->tensorflow==2.5.0) (1.3.1)
Requirement already satisfied: importlib-metadata>=4.4 in /usr/local/lib/python3.7/dist packages (from markdown>=2.6.8->tensorboard~=2.5->tensorflow==2.5.0) (4.13.0) Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from importlib-metadata>=4.4->markdown>=2.6.8->tensorboard~=2.5->tensorflow==2.5.0) (3.10.0)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in /usr/local/lib/python3.7/dist packages (from pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorboard~=2.5->tensorflow==2.5.0) (0.4.8)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard~=2.5->tensorflow==2.5.0) (1.24.3)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard~=2.5->tensorflow==2.5.0) (2.10)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard~=2.5->tensorflow==2.5.0) (3.0.4) Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard~=2.5->tensorflow==2.5.0) (2022.9.24) Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.7/dist-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1->tensorboard~=2.5- >tensorflow==2.5.0) (3.2.2)

Image Augmentation

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
train_datagen=ImageDataGenerator(rescale=1./255,zoom_range=0.2,horizontal_flip=True,v
ertical_flip=False)
test_datagen=ImageDataGenerator(rescale=1./255)
ls
pwd
/content
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 crede
ntials.
# You might want to remove those credentials before you share the notebook.
client_4ff9f1114db24196a9abd4f5c1f0b60a = ibm_boto3.client(service_name='s3',
ibm_api_key_id='j4lNXssktSSxQiDx3pbNR_eFi1SMCDE6MFnBQ_EmNCDM',
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')
streaming_body_1 = client_4ff9f1114db24196a9abd4f5c1f0b60a.get_object(Bucket='trainm
odel-donotdelete-pr-cbqe37eh8gzesa', Key='fruit-dataset.zip')['Body']
# 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 possibil ities 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
'/home/wsuser/work'
import os
filenames = os.listdir('/home/wsuser/work/fruit-dataset/train')
x_train=train_datagen.flow_from_directory("/home/wsuser/work/fruit
dataset/train",target_size=(128,128),class_mode='categorical',batch_size=24) Found 5384
images belonging to 6 classes.
x_test=test_datagen.flow_from_directory(r"/home/wsuser/work/fruit
dataset/test",target_size=(128,128),
 class_mode='categorical',batch_size=24)
Found 1686 images belonging to 6 classes.
x_train.class_indices
{'Apple___Black_rot': 0, 'Apple___healthy': 1, 'Corn_(maize)___Northern_Leaf_Blight': 2,
 'Corn_(maize)___healthy': 3, 'Peach___Bacterial_spot': 4, 'Peach___healthy': 5}
```

CNN

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Convolution2D,MaxPooling2D,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.summary()
Model: "sequential_1"
```

_____Layer (type)
Output Shape Param #
==========
conv2d_1 (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
32*(3*3*3+1)
896
#Hidden Layers
model.add(Dense(300,activation='relu'))
model.add(Dense(150,activation='relu'))

Output Layers

model.add(Dense(6,activation='softmax'))
model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
len(x_train)
225
1238/24
51.583333333333336
model.fit_generator(x_train,steps_per_epoch=len(x_train),validation_data=x_test,validatio
n_steps=len(x_test),epochs=10)
/tmp/wsuser/ipykernel_164/1582812018.py:1: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.

model.fit_generator(x_train,steps_per_epoch=len(x_train),validation_data=x_test,validation
_steps=len(x_test),epochs=10)
Epoch 1/10
225/225 [==========] - 118s 520ms/step - loss: 0.8920 - accuracy: 0.8094 - val_loss: 0.2273 - val_accuracy: 0.9235
Epoch 2/10
225/225 [==========] - 116s 515ms/step - loss: 0.2367 - accuracy: 0.9179 - val_loss: 0.2056 - val_accuracy: 0.9324
Epoch 3/10
225/225 [==========] - 116s 517ms/step - loss: 0.1970 - accuracy: 0.9337 - val_loss: 0.4972 - val_accuracy: 0.8754
Epoch 4/10

```
225/225 [==============================] - 117s 521ms/step - loss: 0.1688 -
accuracy:  0.9422 - val_loss: 0.2279 - val_accuracy: 0.9217
Epoch 5/10
225/225 [==============================] - 116s 516ms/step - loss: 0.1438 -
accuracy:  0.9487 - val_loss: 0.1685 - val_accuracy: 0.9484
Epoch 6/10
225/225 [==============================] - 117s 518ms/step - loss: 0.1362 -
accuracy:  0.9556 - val_loss: 0.1176 - val_accuracy: 0.9662
Epoch 7/10
225/225 [==============================] - 116s 515ms/step - loss: 0.1282 -
accuracy:  0.9590 - val_loss: 0.5466 - val_accuracy: 0.8387
Epoch 8/10
225/225 [==============================] - 116s 514ms/step - loss: 0.1282 -
accuracy:  0.9597 - val_loss: 0.1194 - val_accuracy: 0.9620
Epoch 9/10
225/225 [==============================] - 116s 514ms/step - loss: 0.1141 -
accuracy:  0.9616 - val_loss: 0.1478 - val_accuracy: 0.9508
Epoch 10/10
225/225 [==============================] - 116s 516ms/step - loss: 0.0927 -
accuracy:  0.9695 - val_loss: 0.0772 - val_accuracy: 0.9751
<keras.callbacks.History at 0x7f71e8184070>
```

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
Train-model_new.tgz
```

IBM Cloud Deployment Model

```
!pip install watson-machine-learning-client –upgrade
Collecting watson-machine-learning-client
 Downloading watson_machine_learning_client-1.0.391-py3-none-any.whl (538 kB)
 |████████████████████████████████| 538 kB 21.2 MB/s eta 0:00:01
Requirement already satisfied: tqdm in /opt/conda/envs/Python-3.9/lib/python3.9/site packages
(from watson-machine-learning-client) (4.62.3)
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site packages
(from watson-machine-learning-client) (2022.9.24)
```

Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from watson-machine-learning-client) (2.26.0)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from watson-machine-learning-client) (0.8.9)
Requirement already satisfied: ibm-cos-sdk in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (2.11.0) Requirement already satisfied: pandas in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from watson-machine-learning-client) (1.3.4)
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from watson-machine-learning-client) (0.3.3)
Requirement already satisfied: boto3 in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from watson-machine-learning-client) (1.18.21)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from watson-machine-learning-client) (1.26.7)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client) (0.10.0)
Requirement already satisfied: s3transfer<0.6.0,>=0.5.0 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client) (0.5.0)
Requirement already satisfied: botocore<1.22.0,>=1.21.21 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client) (1.21.41)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from botocore<1.22.0,>=1.21.21->boto3->watson machine-learning-client) (2.8.2)
Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from python-dateutil<3.0.0,>=2.1->botocore<1.22.0,>=1.21.21->boto3->watson machine-learning-client) (1.15.0)
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from ibm-cos-sdk->watson-machine-learning-client) (2.11.0)
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from ibm-cos-sdk->watson-machine-learning-client) (2.11.0)
Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from requests->watson-machine-learning-client) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from requests->watson-machine-learning-client) (3.3) Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from pandas->watson-machine-learning-client) (2021.3) Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from pandas->watson-machine-learning-client) (1.19.5) Installing collected packages: watson-machine-learning-client
Successfully installed watson-machine-learning-client-1.0.391

```
from ibm_watson_machine_learning import APIClient
wml_credentials = {
 "url": "https://us-south.ml.cloud.ibm.com",
 "apikey":"0P3XkyCFYqABnc48BNG2ReoGAJy-oDXDRuULl4Y_zFxa"
 }
client = APIClient(wml_credentials)
def guid_from_space_name(client, space_name):
```

```
 space = client.spaces.get_details()
 return(next(item for item in space['resources'] if item['entity']["name"]==space_name)['m
etadata']['id'])
space_uid = guid_from_space_name(client, 'Trainmodel')
print("Space UID = " + space_uid)
Space UID = 616c7d74-e99b-4c09-9922-27394a62c2d0
client.set.default_space(space_uid)
'SUCCESS'
client.software_specifications.list()
NAME ASSET_ID TYPE
default_py3.6 0062b8c9-8b7d-44a0-a9b9-46c416adcbd9 base kernel-spark3.2-scala2.12
020d69ce-7ac1-5e68-ac1a-31189867356a base pytorch-onnx_1.3-py3.7-edt 069ea134-3346-
5748-b513-49120e15d288 base scikit-learn_0.20-py3.6 09c5a1d0-9c1e-4473-a344-
eb7b665ff687 base spark-mllib_3.0-scala_2.12 09f4cff0-90a7-5899-b9ed-1ef348aebdee base
pytorch-onnx_rt22.1-py3.9 0b848dd4-e681-5599-be41-b5f6fccc6471 base ai-function_0.1-py3.6
0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda base shiny-r3.6 0e6e79df-875e-4f24-8ae9-
62dcc2148306 base
tensorflow_2.4-py3.7-horovod 1092590a-307d-563d-9b62-4eb7d64b3f22 base pytorch_1.1-
py3.6 10ac12d6-6b30-4ccd-8392-3e922c096a92 base tensorflow_1.15-py3.6-ddl 111e41b3-
de2d-5422-a4d6-bf776828c4b7 base runtime-22.1-py3.9 12b83a17-24d8-5082-900f-
0ab31fbfd3cb base scikit-learn_0.22-py3.6 154010fa-5b3b-4ac1-82af-4d5ee5abbc85 base
default_r3.6 1b70aec3-ab34-4b87-8aa0-a4a3c8296a36 base pytorch-onnx_1.3-py3.6 1bc6029a-
cc97-56da-b8e0-39c3880dbbe7 base kernel-spark3.3-r3.6 1c9e5454-f216-59dd-a20e-
474a5cdf5988 base pytorch-onnx_rt22.1-py3.9-edt 1d362186-7ad5-5b59-8b6c-9d0880bde37f
base tensorflow_2.1-py3.6 1eb25b84-d6ed-5dde-b6a5-3fbdf1665666 base spark-mllib_3.2
20047f72-0a98-58c7-9ff5-a77b012eb8f5 base tensorflow_2.4-py3.8-horovod 217c16f6-178f-
56bf-824a-b19f20564c49 base runtime-22.1-py3.9-cuda 26215f05-08c3-5a41-a1b0-
da66306ce658 base do_py3.8 295addb5-9ef9-547e-9bf4-92ae3563e720 base autoai-ts_3.8-py3.8
2aa0c932-798f-5ae9-abd6-15e0c2402fb5 base tensorflow_1.15-py3.6 2b73a275-7cbf-420b-
a912-eae7f436e0bc base kernel-spark3.3-py3.9 2b7961e2-e3b1-5a8c-a491-482c8368839a base
pytorch_1.2-py3.6 2c8ef57d-2687-4b7d-acce-01f94976dac1 base spark-mllib_2.3 2e51f700-
bca0-4b0d-88dc-5c6791338875 base pytorch-onnx_1.1-py3.6-edt 32983cea-3f32-4400-8965-
dde874a8d67e base spark-mllib_3.0-py37 36507ebe-8770-55ba-ab2a-eafe787600e9 base spark-
mllib_2.4 390d21f8-e58b-4fac-9c55-d7ceda621326 base xgboost_0.82-py3.6 39e31acd-5f30-
41dc-ae44-60233c80306e base pytorch-onnx_1.2-py3.6-edt 40589d0e-7019-4e28-8daa-
fb03b6f4fe12 base default_r36py38 41c247d3-45f8-5a71-b065-8580229facf0 base
autoai-ts_rt22.1-py3.9 4269d26e-07ba-5d40-8f66-2d495b0c71f7 base autoai-obm_3.0
42b92e18-d9ab-567f-988a-4240ba1ed5f7 base pmml-3.0_4.3 493bcb95-16f1-5bc5-bee8-
81b8af80e9c7 base spark-mllib_2.4-r_3.6 49403dff-92e9-4c87-a3d7-a42d0021c095 base
xgboost_0.90-py3.6 4ff8d6c2-1343-4c18-85e1-689c965304d3 base pytorch-onnx_1.1-py3.6
50f95b2a-bc16-43bb-bc94-b0bed208c60b base autoai-ts_3.9-py3.8 52c57136-80fa-572e-8728-
a5e7cbb42cde base spark-mllib_2.4-scala_2.11 55a70f99-7320-4be5-9fb9-9edb5a443af5 base
spark-mllib_3.0 5c1b0ca2-4977-5c2e-9439-ffd44ea8ffe9 base autoai-obm_2.0 5c2e37fa-80b8-
5e77-840f-d912469614ee base spss-modeler_18.1 5c3cad7e-507f-4b2a-a9a3-ab53a21dee8b base
cuda-py3.8 5d3232bf-c86b-5df4-a2cd-7bb870a1cd4e base autoai-kb_3.1-py3.7 632d4b22-10aa-
5180-88f0-f52dfb6444d7 base pytorch-onnx_1.7-py3.8 634d3cdc-b562-5bf9-a2d4-
```

ea90a478456b base spark-mllib_2.3-r_3.6 6586b9e3-ccd6-4f92-900f-0f8cb2bd6f0c base
tensorflow_2.4-py3.7 65e171d7-72d1-55d9-8ebb-f813d620c9bb base spss-modeler_18.2
687eddc9-028a-4117-b9dd-e57b36f1efa5 base ----------

Note: Only first 50 records were displayed. To display more use 'limit' parameter.

```
software_space_uid = client.software_specifications.get_uid_by_name("tensorflow_rt22.1-
py3.9")
software_spec_uid
‘1eb25b84-d6ed-5dde-b6a5-3fbdf1665666'
ls
fruit-dataset/ fruit.h5 Train-model_new.tgz
model_details = client.repository.store_model(model= 'Train-model_new.tgz',  meta_props={
client.repository.ModelMetaNames.NAME:"CNN",
client.repository.ModelMetaNames.TYPE:"tensorflow_2.7",
client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_space_uid} )
model_id = client.repository.get_model_id(model_details)
model_id
'd0aeb6a2-e89c-4f8d-bf2f-a28ca4ea3cca'
ls
fruit-dataset/ fruit.h5 Train-model_new.tgz
```

Test The Model

```
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
model=load_model('fruit.h5')
#@title
img=image.load_img(r"C:\Users\LENOVO\Desktop\fruit-dataset\fruit dataset\test\00fca0da-
2db3-481b-b98a
9b67bb7b105c___RS_HL 7708.JPG",target_size=(128,128))
img
img=image.load_img(r"C:\Users\LENOVO\Desktop\ibm\Dataset Plant Disease\fruit
dataset\fruit-dataset\test\Apple___healthy\0adc1c5b-8958-47c0-a152- f28078c214f1___RS_HL
7825.JPG",target_size=(128,128))
img
```

```
x=image.img_to_array(img)
X
array([[[ 99., 86., 106.],
 [101., 88., 108.],
 [118., 105., 125.],
```

...,
[92., 83., 102.],
[93., 84., 103.],
[89., 80., 99.]],
[[96., 83., 103.],
[87., 74., 94.],
[102., 89., 109.],
...,
[88., 79., 98.],
[89., 80., 99.],
[83., 74., 93.]],
[[86., 73., 93.],
[88., 75., 95.],
[98., 85., 105.],
...,
[107., 98., 117.],
[96., 87., 106.],
[96., 87., 106.]],
...,
[[172., 175., 194.],
[173., 176., 195.],
[175., 178., 197.],
...,
[179., 180., 198.],
[184., 185., 203.],
[179., 180., 198.]],
[[172., 175., 194.],
[170., 173., 192.],
[173., 176., 195.],
...,
[178., 179., 197.],
[182., 183., 201.],
[178., 179., 197.]],
[[169., 172., 191.],
[166., 169., 188.],
[168., 171., 190.],
...,
[187., 188., 206.],
[185., 186., 204.],
[186., 187., 205.]]], dtype=float32) x=np.expand_dims(x,axis=0)
X
array([[[[99., 86., 106.],
[101., 88., 108.],
[118., 105., 125.],
...,
[92., 83., 102.],

```
 [ 93., 84., 103.],
 [ 89., 80., 99.]],
 [[ 96., 83., 103.],
 [ 87., 74., 94.],
 [102., 89., 109.],
 ...,
 [ 88., 79., 98.],
 [ 89., 80., 99.],
 [ 83., 74., 93.]],
 [[ 86., 73., 93.],
 [ 88., 75., 95.],
 [ 98., 85., 105.],
 ...,
 [107., 98., 117.],
 [ 96., 87., 106.],
 [ 96., 87., 106.]],
 ...,
 [[172., 175., 194.],
 [173., 176., 195.],
 [175., 178., 197.],
 ...,
 [179., 180., 198.],
 [184., 185., 203.],
 [179., 180., 198.]],
 [[172., 175., 194.],
 [170., 173., 192.],
 [173., 176., 195.],
 ...,
 [178., 179., 197.],
 [182., 183., 201.],
 [178., 179., 197.]],
 [[169., 172., 191.],
 [166., 169., 188.],
 [168., 171., 190.],
 ...,
 [187., 188., 206.],
 [185., 186., 204.],
 [186., 187., 205.]]]], dtype=float32)
y=np.argmax(model.predict(x),axis=1)
1/1 [==============================] - 0s 105ms/step
x_train.class_indices
{'Apple___Black_rot': 0, 'Apple___healthy': 1, 'Corn_(maize)___Northern_Leaf_Blight': 2,
 'Corn_(maize)___healthy': 3, 'Peach___Bacterial_spot': 4, 'Peach___healthy': 5}
index=['Apple___Black_rot','Apple___healthy','Corn_(maize)___Northern_Leaf_Blight','Corn
_(maize)___healthy','Peach___Bacterial_spot','Peach___healthy']
index[y[0]]
```

```
'Apple___healthy'
img=image.load_img(r"C:\LENOVO\Desktop\ibm\Dataset Plant Disease\fruit-dataset\fruit
dataset\test\Peach___healthy\0a2ed402-5d23-4e8d-bc98-
b264aea9c3fb___Rutg._HL 2471.JPG",target_size=(128,128))
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
y=np.argmax(model.predict(x),axis=1)
index=['Apple___Black_rot','Apple___healthy''Peach___Bacterial_spot','Peach___healthy']
index[y[0]]
1/1 [==============================] - 0s 26ms/step
'Peach___healthy'
import os
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
from flask import Flask,render_template,request
app=Flask(_name_)
model=load_model("fruit.h5")
@app.route('/')
def index():
 return render_template("index.html")
@app.route('/predict',methods=['GET','POST'])
def upload():
 if request.method=='POST':
 f=request.files['image']
 basepath=os.path.dirname('_file_')
 filepath=os.path.join(basepath,'uploads',f.filename)
 f.save(filepath)
 img=image.load_img(filepath,target_size=(128,128))
 x=image.img_to_array(img)
 x=np.expand_dims(x,axis=0)
 pred=np.argmax(model.predict(x),axis=1)
 index=['Apple___Black_rot','Apple___healthy',
,'Peach___Bacterial_spot','Peach___healthy']
 text="The Classified Fruit disease is : " +str(index[pred[0]])
 return text
if__name_=='_main_':
 app.run(debug=False)
```

vegetable model :

```
ls
sample_data/
pwd
'/home/wsuser/work'
!pip install keras==2.7.0
```

```
!pip install tensorflow==2.5.0
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab
wheels/public/simple/
Requirement already satisfied: keras==2.7.0 in /usr/local/lib/python3.7/dist-packages (2.7.0)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab
wheels/public/simple/
Requirement already satisfied: tensorflow==2.5.0 in /usr/local/lib/python3.7/dist-packages
 (2.5.0)
Requirement already satisfied: h5py~=3.1.0 in /usr/local/lib/python3.7/dist-packages (from
 tensorflow==2.5.0) (3.1.0)
Requirement already satisfied: protobuf>=3.9.2 in /usr/local/lib/python3.7/dist-packages (from
tensorflow==2.5.0) (3.19.6)
Requirement already satisfied: typing-extensions~=3.7.4 in /usr/local/lib/python3.7/dist packages
(from tensorflow==2.5.0) (3.7.4.3)
Requirement already satisfied: keras-nightly~=2.5.0.dev in /usr/local/lib/python3.7/dist packages
(from tensorflow==2.5.0) (2.5.0.dev2021032900)
Requirement already satisfied: flatbuffers~=1.12.0 in /usr/local/lib/python3.7/dist-packages
 (from tensorflow==2.5.0) (1.12)
Requirement already satisfied: gast==0.4.0 in /usr/local/lib/python3.7/dist-packages (from
 tensorflow==2.5.0) (0.4.0)
Requirement already satisfied: absl-py~=0.10 in /usr/local/lib/python3.7/dist-packages (from
tensorflow==2.5.0) (0.15.0)
Requirement already satisfied: astunparse~=1.6.3 in /usr/local/lib/python3.7/dist-packages (from
tensorflow==2.5.0) (1.6.3)
Requirement already satisfied: tensorflow-estimator<2.6.0,>=2.5.0rc0 in
 /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0) (2.5.0) Requirement already
satisfied: tensorboard~=2.5 in /usr/local/lib/python3.7/dist-packages (from tensorflow==2.5.0)
(2.9.1)
Requirement already satisfied: opt-einsum~=3.3.0 in /usr/local/lib/python3.7/dist-packages
 (from tensorflow==2.5.0) (3.3.0)
Requirement already satisfied: six~=1.15.0 in /usr/local/lib/python3.7/dist-packages (from
 tensorflow==2.5.0) (1.15.0)
Requirement already satisfied: google-pasta~=0.2 in /usr/local/lib/python3.7/dist-packages
 (from tensorflow==2.5.0) (0.2.0)
Requirement already satisfied: grpcio~=1.34.0 in /usr/local/lib/python3.7/dist-packages (from
tensorflow==2.5.0) (1.34.1)
Requirement already satisfied: wrapt~=1.12.1 in /usr/local/lib/python3.7/dist-packages (from
tensorflow==2.5.0) (1.12.1)
Requirement already satisfied: termcolor~=1.1.0 in /usr/local/lib/python3.7/dist-packages (from
tensorflow==2.5.0) (1.1.0)
Requirement already satisfied: keras-preprocessing~=1.1.2 in /usr/local/lib/python3.7/dist
packages (from tensorflow==2.5.0) (1.1.2)
Requirement already satisfied: wheel~=0.35 in /usr/local/lib/python3.7/dist-packages (from
 tensorflow==2.5.0) (0.38.3)
Requirement already satisfied: numpy~=1.19.2 in /usr/local/lib/python3.7/dist-packages (from
tensorflow==2.5.0) (1.19.5)
```

Requirement already satisfied: cached-property in /usr/local/lib/python3.7/dist-packages (from h5py~=3.1.0->tensorflow==2.5.0) (1.5.2)
Requirement already satisfied: google-auth<3,>=1.6.3 in /usr/local/lib/python3.7/dist packages (from tensorboard~=2.5->tensorflow==2.5.0) (2.14.1)
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in /usr/local/lib/python3.7/dist-packages (from tensorboard~=2.5->tensorflow==2.5.0) (0.6.1)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in /usr/local/lib/python3.7/dist-packages (from tensorboard~=2.5->tensorflow==2.5.0) (1.8.1)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in /usr/local/lib/python3.7/dist-packages (from tensorboard~=2.5->tensorflow==2.5.0) (0.4.6)
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.7/dist-packages (from tensorboard~=2.5->tensorflow==2.5.0) (1.0.1)
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.7/dist-packages (from tensorboard~=2.5->tensorflow==2.5.0) (3.4.1)
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.7/dist packages (from tensorboard~=2.5->tensorflow==2.5.0) (2.23.0)
Requirement already satisfied: setuptools>=41.0.0 in /usr/local/lib/python3.7/dist-packages (from tensorboard~=2.5->tensorflow==2.5.0) (57.4.0)
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.7/dist-packages (from google-auth<3,>=1.6.3->tensorboard~=2.5->tensorflow==2.5.0) (4.9) Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.7/dist packages (from google-auth<3,>=1.6.3->tensorboard~=2.5->tensorflow==2.5.0) (0.2.8) Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.7/dist packages (from google-auth<3,>=1.6.3->tensorboard~=2.5->tensorflow==2.5.0) (5.2.0) Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.7/dist packages (from google-auth-oauthlib<0.5,>=0.4.1->tensorboard~=2.5->tensorflow==2.5.0) (1.3.1)
Requirement already satisfied: importlib-metadata>=4.4 in /usr/local/lib/python3.7/dist packages (from markdown>=2.6.8->tensorboard~=2.5->tensorflow==2.5.0) (4.13.0) Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from importlib-metadata>=4.4->markdown>=2.6.8->tensorboard~=2.5->tensorflow==2.5.0) (3.10.0)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in /usr/local/lib/python3.7/dist packages (from pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorboard~=2.5->tensorflow==2.5.0) (0.4.8)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard~=2.5->tensorflow==2.5.0) (1.24.3)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard~=2.5->tensorflow==2.5.0) (2.10)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard~=2.5->tensorflow==2.5.0) (3.0.4) Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard~=2.5->tensorflow==2.5.0) (2022.9.24) Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.7/dist-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1->tensorboard~=2.5- >tensorflow==2.5.0) (3.2.2)

Image Augmentation

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
train_datagen=ImageDataGenerator(rescale=1./255,zoom_range=0.2,horizontal_flip=True,v
ertical_flip=False)
test_datagen=ImageDataGenerator(rescale=1./255)
ls
pwd
/content
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 crede
ntials.
# You might want to remove those credentials before you share the notebook.
client_4ff9f1114db24196a9abd4f5c1f0b60a = ibm_boto3.client(service_name='s3',
ibm_api_key_id='j4lNXssktSSxQiDx3pbNR_eFi1SMCDE6MFnBQ_EmNCDM',
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')
streaming_body_1 = client_4ff9f1114db24196a9abd4f5c1f0b60a.get_object(Bucket='trainm
odel-donotdelete-pr-cbqe37eh8gzesa', Key='vegetable-dataset.zip')['Body']
# 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 possibil ities 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
'/home/wsuser/work'
import os
filenames = os.listdir('/home/wsuser/work/vegetable-dataset/train')
x_train=train_datagen.flow_from_directory("/home/wsuser/work/vegetable
dataset/train",target_size=(128,128),class_mode='categorical',batch_size=24) Found 5384
images belonging to 6 classes.
x_test=test_datagen.flow_from_directory(r"/home/wsuser/work/vegetable
dataset/test",target_size=(128,128),
  class_mode='categorical',batch_size=24)
Found 1686 images belonging to 6 classes.
x_train.class_indices
```

{'Tomato___Blight': 0, 'Tomato___healthy': 1, 'Corn_(maize)___Northern_Leaf_Blight': 2,
'Corn_(maize)___healthy': 3, 'Potato___Blight': 4, 'Potato___healthy': 5}

CNN

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Convolution2D,MaxPooling2D,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.summary()
```

Model: "sequential_1"

_____Layer (type)
Output Shape Param #
==========
conv2d_1 (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

```
32*(3*3*3+1)
```

896

```
#Hidden Layers
model.add(Dense(300,activation='relu'))
model.add(Dense(150,activation='relu'))
```

Output Layers

```
model.add(Dense(6,activation='softmax'))
model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
len(x_train)
```

225

```
1238/24
```

51.583333333333336

```
model.fit_generator(x_train,steps_per_epoch=len(x_train),validation_data=x_test,validatio
n_steps=len(x_test),epochs=10)
```

/tmp/wsuser/ipykernel_164/1582812018.py:1: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.

model.fit_generator(x_train,steps_per_epoch=len(x_train),validation_data=x_test,validation_steps=len(x_test),epochs=10)
Epoch 1/10
225/225 [==============================] - 118s 520ms/step - loss: 0.8920 - accuracy: 0.8094 - val_loss: 0.2273 - val_accuracy: 0.9235
Epoch 2/10
225/225 [==============================] - 116s 515ms/step - loss: 0.2367 - accuracy: 0.9179 - val_loss: 0.2056 - val_accuracy: 0.9324
Epoch 3/10
225/225 [==============================] - 116s 517ms/step - loss: 0.1970 - accuracy: 0.9337 - val_loss: 0.4972 - val_accuracy: 0.8754
Epoch 4/10
225/225 [==============================] - 117s 521ms/step - loss: 0.1688 - accuracy: 0.9422 - val_loss: 0.2279 - val_accuracy: 0.9217
Epoch 5/10
225/225 [==============================] - 116s 516ms/step - loss: 0.1438 - accuracy: 0.9487 - val_loss: 0.1685 - val_accuracy: 0.9484
Epoch 6/10
225/225 [==============================] - 117s 518ms/step - loss: 0.1362 - accuracy: 0.9556 - val_loss: 0.1176 - val_accuracy: 0.9662
Epoch 7/10
225/225 [==============================] - 116s 515ms/step - loss: 0.1282 - accuracy: 0.9590 - val_loss: 0.5466 - val_accuracy: 0.8387
Epoch 8/10
225/225 [==============================] - 116s 514ms/step - loss: 0.1282 - accuracy: 0.9597 - val_loss: 0.1194 - val_accuracy: 0.9620
Epoch 9/10
225/225 [==============================] - 116s 514ms/step - loss: 0.1141 - accuracy: 0.9616 - val_loss: 0.1478 - val_accuracy: 0.9508
Epoch 10/10
225/225 [==============================] - 116s 516ms/step - loss: 0.0927 - accuracy: 0.9695 - val_loss: 0.0772 - val_accuracy: 0.9751
<keras.callbacks.History at 0x7f71e8184070>

Saving Model

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

ls -1
vegetable-dataset/
vegetable.h5
Train-model_new.tgz

IBM Cloud Deployment Model

!pip install watson-machine-learning-client –upgrade
Collecting watson-machine-learning-client
Downloading watson_machine_learning_client-1.0.391-py3-none-any.whl (538 kB)
|████████████████████████████████| 538 kB 21.2 MB/s eta 0:00:01
Requirement already satisfied: tqdm in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from watson-machine-learning-client) (4.62.3)
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from watson-machine-learning-client) (2022.9.24)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from watson-machine-learning-client) (2.26.0)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from watson-machine-learning-client) (0.8.9)
Requirement already satisfied: ibm-cos-sdk in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (2.11.0) Requirement already satisfied: pandas in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from watson-machine-learning-client) (1.3.4)
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from watson-machine-learning-client) (0.3.3)
Requirement already satisfied: boto3 in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from watson-machine-learning-client) (1.18.21)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from watson-machine-learning-client) (1.26.7)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client) (0.10.0)
Requirement already satisfied: s3transfer<0.6.0,>=0.5.0 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client) (0.5.0)
Requirement already satisfied: botocore<1.22.0,>=1.21.21 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client) (1.21.41)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from botocore<1.22.0,>=1.21.21->boto3->watson machine-learning-client) (2.8.2)
Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site packages (from python-dateutil<3.0.0,>=2.1->botocore<1.22.0,>=1.21.21->boto3->watson machine-learning-client) (1.15.0)
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from ibm-cos-sdk->watson-machine-learning-client) (2.11.0)
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from ibm-cos-sdk->watson-machine-learning-client) (2.11.0)

Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from requests->watson-machine-learning-client) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from requests->watson-machine-learning-client) (3.3) Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from pandas->watson-machine-learning-client) (2021.3) Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python 3.9/lib/python3.9/site-packages (from pandas->watson-machine-learning-client) (1.19.5) Installing collected packages: watson-machine-learning-client
Successfully installed watson-machine-learning-client-1.0.391
from ibm_watson_machine_learning import APIClient
wml_credentials = {
 "url": "https://us-south.ml.cloud.ibm.com",
 "apikey":"0P3XkyCFYqABnc48BNG2ReoGAJy-oDXDRuULl4Y_zFxa"
 }
client = APIClient(wml_credentials)
def guid_from_space_name(client, space_name):
 space = client.spaces.get_details()
 return(next(item for item in space['resources'] if item['entity']["name"]==space_name)['m
etadata']['id'])
space_uid = guid_from_space_name(client, 'Trainmodel')
print("Space UID = " + space_uid)
Space UID = 616c7d74-e99b-4c09-9922-27394a62c2d0
client.set.default_space(space_uid)
'SUCCESS'
client.software_specifications.list()
NAME ASSET_ID TYPE
default_py3.6 0062b8c9-8b7d-44a0-a9b9-46c416adcbd9 base kernel-spark3.2-scala2.12 020d69ce-7ac1-5e68-ac1a-31189867356a base pytorch-onnx_1.3-py3.7-edt 069ea134-3346-5748-b513-49120e15d288 base scikit-learn_0.20-py3.6 09c5a1d0-9c1e-4473-a344-eb7b665ff687 base spark-mllib_3.0-scala_2.12 09f4cff0-90a7-5899-b9ed-1ef348aebdee base pytorch-onnx_rt22.1-py3.9 0b848dd4-e681-5599-be41-b5f6fccc6471 base ai-function_0.1-py3.6 0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda base shiny-r3.6 0e6e79df-875e-4f24-8ae9-62dcc2148306 base
tensorflow_2.4-py3.7-horovod 1092590a-307d-563d-9b62-4eb7d64b3f22 base pytorch_1.1-py3.6 10ac12d6-6b30-4ccd-8392-3e922c096a92 base tensorflow_1.15-py3.6-ddl 111e41b3-de2d-5422-a4d6-bf776828c4b7 base runtime-22.1-py3.9 12b83a17-24d8-5082-900f-0ab31fbfd3cb base scikit-learn_0.22-py3.6 154010fa-5b3b-4ac1-82af-4d5ee5abbc85 base default_r3.6 1b70aec3-ab34-4b87-8aa0-a4a3c8296a36 base pytorch-onnx_1.3-py3.6 1bc6029a-cc97-56da-b8e0-39c3880dbbe7 base kernel-spark3.3-r3.6 1c9e5454-f216-59dd-a20e-474a5cdf5988 base pytorch-onnx_rt22.1-py3.9-edt 1d362186-7ad5-5b59-8b6c-9d0880bde37f base tensorflow_2.1-py3.6 1eb25b84-d6ed-5dde-b6a5-3fbdf1665666 base spark-mllib_3.2 20047f72-0a98-58c7-9ff5-a77b012eb8f5 base tensorflow_2.4-py3.8-horovod 217c16f6-178f-56bf-824a-b19f20564c49 base runtime-22.1-py3.9-cuda 26215f05-08c3-5a41-a1b0-da66306ce658 base do_py3.8 295addb5-9ef9-547e-9bf4-92ae3563e720 base autoai-ts_3.8-py3.8 2aa0c932-798f-5ae9-abd6-15e0c2402fb5 base tensorflow_1.15-py3.6 2b73a275-7cbf-420b-a912-eae7f436e0bc base kernel-spark3.3-py3.9 2b7961e2-e3b1-5a8c-a491-482c8368839a base

pytorch_1.2-py3.6 2c8ef57d-2687-4b7d-acce-01f94976dac1 base spark-mllib_2.3 2e51f700-bca0-4b0d-88dc-5c6791338875 base pytorch-onnx_1.1-py3.6-edt 32983cea-3f32-4400-8965-dde874a8d67e base spark-mllib_3.0-py37 36507ebe-8770-55ba-ab2a-eafe787600e9 base spark-mllib_2.4 390d21f8-e58b-4fac-9c55-d7ceda621326 base xgboost_0.82-py3.6 39e31acd-5f30-41dc-ae44-60233c80306e base pytorch-onnx_1.2-py3.6-edt 40589d0e-7019-4e28-8daa-fb03b6f4fe12 base default_r36py38 41c247d3-45f8-5a71-b065-8580229facf0 base autoai-ts_rt22.1-py3.9 4269d26e-07ba-5d40-8f66-2d495b0c71f7 base autoai-obm_3.0 42b92e18-d9ab-567f-988a-4240ba1ed5f7 base pmml-3.0_4.3 493bcb95-16f1-5bc5-bee8-81b8af80e9c7 base spark-mllib_2.4-r_3.6 49403dff-92e9-4c87-a3d7-a42d0021c095 base xgboost_0.90-py3.6 4ff8d6c2-1343-4c18-85e1-689c965304d3 base pytorch-onnx_1.1-py3.6 50f95b2a-bc16-43bb-bc94-b0bed208c60b base autoai-ts_3.9-py3.8 52c57136-80fa-572e-8728-a5e7cbb42cde base spark-mllib_2.4-scala_2.11 55a70f99-7320-4be5-9fb9-9edb5a443af5 base spark-mllib_3.0 5c1b0ca2-4977-5c2e-9439-ffd44ea8ffe9 base autoai-obm_2.0 5c2e37fa-80b8-5e77-840f-d912469614ee base spss-modeler_18.1 5c3cad7e-507f-4b2a-a9a3-ab53a21dee8b base cuda-py3.8 5d3232bf-c86b-5df4-a2cd-7bb870a1cd4e base autoai-kb_3.1-py3.7 632d4b22-10aa-5180-88f0-f52dfb6444d7 base pytorch-onnx_1.7-py3.8 634d3cdc-b562-5bf9-a2d4-ea90a478456b base spark-mllib_2.3-r_3.6 6586b9e3-ccd6-4f92-900f-0f8cb2bd6f0c base tensorflow_2.4-py3.7 65e171d7-72d1-55d9-8ebb-f813d620c9bb base spss-modeler_18.2 687eddc9-028a-4117-b9dd-e57b36f1efa5 base

---

Note: Only first 50 records were displayed. To display more use 'limit' parameter.

```
software_space_uid = client.software_specifications.get_uid_by_name("tensorflow_rt22.1-py3.9")
software_spec_uid
‘1eb25b84-d6ed-5dde-b6a5-3fbdf1665666'
ls
vegetable-dataset/ vegetable.h5 Train-model_new.tgz
model_details = client.repository.store_model(model= 'Train-model_new.tgz',  meta_props={
client.repository.ModelMetaNames.NAME:"CNN",
client.repository.ModelMetaNames.TYPE:"tensorflow_2.7",
client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_space_uid} )
model_id = client.repository.get_model_id(model_details)
model_id
'd0aeb6a2-e89c-4f8d-bf2f-a28ca4ea3cca'
ls
vegetable-dataset/ vegetable.h5 Train-model_new.tgz
```

Test The Model

```
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
model=load_model('vegetable.h5')
#@title
img=image.load_img(r"C:\Users\LENOVO\Desktop\vegetable-dataset\vegetable dataset\test\00fca0da-2db3-481b-b98a 9b67bb7b105c___RS_HL 7708.JPG",target_size=(128,128))
img
```

```
img=image.load_img(r"C:\Users\LENOVO\Desktop\ibm\Dataset Plant Disease\vegetable dataset\vegetable-dataset\test\Tomato___healthy\0adc1c5b-8958-47c0-a152-f28078c214f1___RS_HL 7825.JPG",target_size=(128,128))
img
```

```
x=image.img_to_array(img)
X
array([[[ 99., 86., 106.],
 [101., 88., 108.],
 [118., 105., 125.],
 ...,
 [ 92., 83., 102.],
 [ 93., 84., 103.],
 [ 89., 80., 99.]],
 [[ 96., 83., 103.],
 [ 87., 74., 94.],
 [102., 89., 109.],
 ...,
 [ 88., 79., 98.],
 [ 89., 80., 99.],
 [ 83., 74., 93.]],
 [[ 86., 73., 93.],
 [ 88., 75., 95.],
 [ 98., 85., 105.],
```

...,
[107., 98., 117.],
[96., 87., 106.],
[96., 87., 106.]],
...,
[[172., 175., 194.],
[173., 176., 195.],
[175., 178., 197.],
...,
[179., 180., 198.],
[184., 185., 203.],
[179., 180., 198.]],
[[172., 175., 194.],
[170., 173., 192.],
[173., 176., 195.],
...,
[178., 179., 197.],
[182., 183., 201.],
[178., 179., 197.]],
[[169., 172., 191.],
[166., 169., 188.],
[168., 171., 190.],
...,
[187., 188., 206.],
[185., 186., 204.],
[186., 187., 205.]]], dtype=float32) x=np.expand_dims(x,axis=0)
X
array([[[[99., 86., 106.],
[101., 88., 108.],
[118., 105., 125.],
...,
[92., 83., 102.],
[93., 84., 103.],
[89., 80., 99.]],
[[96., 83., 103.],
[87., 74., 94.],
[102., 89., 109.],
...,
[88., 79., 98.],
[89., 80., 99.],
[83., 74., 93.]],
[[86., 73., 93.],
[88., 75., 95.],
[98., 85., 105.],
...,
[107., 98., 117.],

```
 [ 96., 87., 106.],
 [ 96., 87., 106.]],
 ...,
 [[172., 175., 194.],
 [173., 176., 195.],
 [175., 178., 197.],
 ...,
 [179., 180., 198.],
 [184., 185., 203.],
 [179., 180., 198.]],
 [[172., 175., 194.],
 [170., 173., 192.],
 [173., 176., 195.],
 ...,
 [178., 179., 197.],
 [182., 183., 201.],
 [178., 179., 197.]],
 [[169., 172., 191.],
 [166., 169., 188.],
 [168., 171., 190.],
 ...,
 [187., 188., 206.],
 [185., 186., 204.],
 [186., 187., 205.]]]], dtype=float32)
y=np.argmax(model.predict(x),axis=1)
1/1 [==============================] - 0s 105ms/step
x_train.class_indices
{'Tomato___Blight': 0, 'Tomato___healthy': 1, 'Corn_(maize)___Northern_Leaf_Blight': 2,
 'Corn_(maize)___healthy': 3, 'Potato___Blight': 4, 'Potato___healthy': 5}
index=['Tomato___Blight','Tomato___healthy','Corn_(maize)___Northern_Leaf_Blight','Cor
n_(maize)___healthy','Potato___Blight','Potato___healthy']
index[y[0]]
'Tomato___healthy'
img=image.load_img(r"C:\LENOVO\Desktop\ibm\Dataset Plant Disease\vegetable
dataset\vegetable-dataset\test\Potato___healthy\0a2ed402-5d23-4e8d-bc98-
b264aea9c3fb___Rutg._HL 2471.JPG",target_size=(128,128))
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
y=np.argmax(model.predict(x),axis=1)
index=['Tomato___Blight','Tomato___healthy''Potato___Blight','Potato___healthy'] index[y[0]]
1/1 [==============================] - 0s 26ms/step
'Potato___healthy'
import os
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
from flask import Flask,render_template,request
```

```
app=Flask(_name_)
model=load_model("vegetable.h5")
@app.route('/')
def index():
 return render_template("index.html")
@app.route('/predict',methods=['GET','POST'])
def upload():
 if request.method=='POST':
 f=request.files['image']
 basepath=os.path.dirname('_file_')
 filepath=os.path.join(basepath,'uploads',f.filename)
 f.save(filepath)
 img=image.load_img(filepath,target_size=(128,128))
 x=image.img_to_array(img)
 x=np.expand_dims(x,axis=0)
 pred=np.argmax(model.predict(x),axis=1)
 index=['Tomato___Blight','Tomato___healthy', ,'Potato___Blight','Potato___healthy']
 text="The Classified Vegetable disease is : " +str(index[pred[0]]) return text
if__name__=='_main__':
 app.run(debug=False)
```

ibmapp.py

```
import requests
from tensorflow.keras.preprocessing import image from tensorflow.keras.models import
load_model
import numpy as np
import pandas as pd
import tensorflow as tf
from flask import Flask, request, render_template, redirect, url_for import os
from werkzeug.utils import secure_filename
app = Flask(_name_)
#load both the vegetable and fruit models
model = load_model("IBM-vegetable.h5")
model1=load_model("IBM-fruit.h5")
#home page
@app.route('/')
def home():
 return render_template('home.html')
#prediction page
@app.route('/prediction')
def prediction():
 return render_template('predict.html')
@app.route('/predict',methods=['POST'])
def predict():
 if request.method == 'POST':
 # Get the file from post request
 f = request.files['image']
```

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	Collecting dataset for pre-processing	10	High	SELVASARAVANAN.L PERUMAL.P SURIYAPRAKASH.M VISHWANATH.S
Sprint-1		USN-2	Data pre-processing-Used to transform the data into useful format.	10	Medium	SELVASARAVANAN.L PERUMAL.P SURIYAPRAKASH.M VISHWANATH.S
Sprint-2	Model Building	USN-3	Model building for fruit and vegetable disease prediction	10	High	SELVASARAVANAN.L PERUMAL.P SURIYAPRAKASH.M VISHWANATH.S
Sprint-2		USN-4	Splitting the data into training and testing from the entire dataset.	10	Medium	SELVASARAVANAN.L PERUMAL.P SURIYAPRAKASH.M VISHWANATH.S
Sprint-3	Training and Testing	USN-5	Training the model and testing the performance of the model	20	Medium	SELVASARAVANAN.L PERUMAL.P SURIYAPRAKASH.M VISHWANATH.S
Sprint-4	Implementation of Web page	USN-6	Implementing the web page for collecting the data from user	10	High	SELVASARAVANAN.L PERUMAL.P SURIYAPRAKASH.M VISHWANATH.S
Sprint-4		USN-6	Deploying the model using IBM Cloud and IBM Watson Studio	10	Medium	SELVASARAVANAN.L PERUMAL.P SURIYAPRAKASH.M VISHWANATH.S

```
# Save the file to ./uploads
basepath = os.path.dirname(_file_)
file_path = os.path.join(
basepath, 'uploads', secure_filename(f.filename)) f.save(file_path)
img = image.load_img(file_path, target_size=(128, 128))
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)

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

if _name_ == "_main_":
 app.run(debug=False)
```

GitHub Link:
https://github.com/IBM-EPBL/IBM-Project-23830-1659931745

Demo video link:

https://drive.google.com/file/d/15MNJcf-j9Yp1ryzjJWDTkTKpYjPRNO6a/view?usp=share_link