FERTILIZERS RECOMMENDATION SYSTEM FOR DISEASE PREDICTION

Submitted by

ID: PNT2022TMID10907

R.Prathibha(811519104084)

T.Yogaasri(811519104124),

I.Sruthika (811519104106),

G.Thivya Priya (811519104112)

In partial fulfillment of the award of the degree

Of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

K.RAMAKRISHNAN COLLEGE OF ENGINEERING

Project Report Format

1. **INTRODUCTION**

1.INTRODUCTION

All the present college interaction web portals are static in natural and not very interactive with the user and even for the developer. Static Web Portals have limitations over extensibility where the web designer has to modify the whole content of the web designer has to modify the whole content of the web site for any enhancement. So there is a need for developinga Dynamic Portal.

This Dynamic College Interaction Web Portal provides an on line solution for the day-to-day operation of an organization. This dynamic environment is provided an easy way of updatingthe information about the college News, Announcements and various activities of the institution anywhere anytime on the network. It has to be very interactive with the user and operators on a data that is stored in a central database on the server. It must be able to install on any web server hosting company. The portal must contain college details like events that are going to be held, like to different details and forum where in the user can share their ideas through discussion.

1.1 PROJECT OVERVIEW

Most plants are affected by a wide variety of bacterial and fungal diseases. Early and accurate identification of plant diseases is essential to ensure high quantity and best quality. An automated system is introduced to identify different diseases on plants. Deep learning techniques are used to identify the diseases and suggest the precautions.

The farmer needs a way to choose the best fertilizers to increase crop growth and cure diseases.

1.2 PURPOSE

To maximize the symptoms. Advise the farmers to verify that. The farmers will be notified of where the fertilizers are available locally using their location. If the product is not available locally, the link to e-commerce websites can be provided. Depending upon the disease spread in the plant, the amount of fertilizer needed to be used should be specified.The application should support English and Tamil languages. The application should be handy and operate on all devices.Prevention steps can be mentioned. If the disease is not identifed then it should be added to the dataset.

The percentage of disease spread in the crop can be predicted.Tour guide of application can be in voice of local languages for easy understanding of the application. Farmers can share their crop's condition and search results. The farmer can also directly search for the disease and find precautions to prevent

the disease before affecting the crop.The mode of transmission of fertilizers can be shown(spray or sprinkle).The related diseases can also be shown in bottom of the predicted disease.
The cost of fertilizers can be mentioned.

2. LITERATURE SURVEY

1. Fertilizers Recommendation System For Disease Prediction In Tree LeaveS

Authors : R. Neela, P. Nithya

Agriculture is the main aspect of country development. Many people lead their life from agriculture field, which gives fully related to agricultural products. Plant disease, especially on leaves, is one of the major factors of reductions in both quality and quantity of the food crops. In agricultural aspects, if the plant is affected by leaf disease then it reduces the growth of the agricultural level. Finding the leaf disease is an important role of agriculture preservation. After pre-processing using a median filter, segmentation is done by Guided Active Contour method and finally, the leaf disease is identified by using Support Vector Machine. The disease-based similarity measure is used for fertilizer recommendation.

Advantage:

It is advantageous to find plant diseases using an automatic technique because it lessens the amount of work required to monitor large crop farms and finds disease symptoms at an early stage, when they first appear on plant leaves.

2. Soil Based Fertilizer Recommendation System for Crop Disease Prediction System

Authors : Dr.P. Pandi Selvi, P. Poornima

Smart analysis and Comprehensive prediction model in agriculture helps the farmer to yield right crop at the right time. The main benefits of the proposed system are as follows: Yield right crop at the right time, Balancing the crop production, control plant disease, Economic growth, and planning to reduce the crop scarcity. Hence to Detect and recognize the plant diseases and to recommend fertilizer it is necessary to provide symptoms in

identifying the disease at its earliest. Hence the authors proposed and implemented new fertilizers Recommendation System for crop disease prediction.

Advantage:

Good potential with ability to detect plant leaf disease.

Disadvantage:

Require more time.

3. Plant disease detection by Image Processing

Author : Monishanker Halder

To promote sustainable development, the smart city implies a global vision that merges artificial intelligence, big data, decision making, information and communication technology (ICT), and the Internet-of-Things (IOT). These processes above are related for solving real life problems. Food is one of the basic needs of human being. World population is increasing day by day. So it has become important to grow sufficient amount of crops to feed such a huge population. But with the time passing by, plants are affected with various kinds of diseases, which cause great harm to the agricultural plant productions. Beside that many countries economy greatly depends on agricultural productivity and it's also a need for a country to attain agricultural productivity of basic agricultural product for the people of that particular country. Detection of plant disease through some automatic technique is beneficial as it requires a large amount of work of monitoring in big farm of crops, and at very early stage itself it detects symptoms of diseases means where they appear on plant leaves. In this paper surveys on different disease classification techniques that can be used for plant leaf disease detection.

Advantage:

High speed, preferable in limited precision.

Disadvantage:

Dimensionality (Large no. the features), Poor performance.

4. Plant leaf diseases detection using image processing techniques

Authors : K.Narsimha Reddy, B.Polaiah, N.Madhu

This paper provides survey on different classification techniques that can be used for plant leaf diseases classification. Identification of symptoms of disease by naked eye is difficult for farmer. Crop protection in large frames is done by using computerized image processing technique that can detect diseased leaf using color information of leaves. There are so many classification techniques such as k-Nearest Neighbor Classifier, Probabilistic Neural Network, Genetic Algorithm, Support Vector Machine, and Principal Component Analysis, Artificial neural network, Fuzzy logic. Selecting a classification method is always a difficult task because the quality of result can vary for different input data. Plant leaf disease classifications have wide applications in various fields such as in biological research, in Agriculture etc. This paper provides an overview of different classification techniques used for plant leaf disease classification.

Advantage:

Much faster and more accurate.

Disadvantage:

Require large storage space.

5. Crop disease identification and classification using pattern recognition and digital image processing techniques

Authors : Goutum Kambale , Dr.Nitin Bilgi

Agricultural scientists play an important role in detecting and finding cure for plant diseases. Sometimes manual identification of disease is time consuming and laborious process. One of the most important factors contributing to low yield is disease attack. Many studies show that quality of agricultural products may be reduced due to various factors of plant diseases. In banana plant, diseases which are commonly observed are panama wilt, yellow sigatoka, black sigatoka, banana streak virus and banana bunchy top virus . The banana plant leaf diseases not only restrict the growth of the plant but also destroy the crop. Banana plant leaf diseases must be identified early and accurately as it can prove detrimental to the yield. Hence, a machine learning method is required to identify the affected leaf images in timely manner. The images required for this work are captured from the fields using digital camera. The captured images are then processed on computer using pattern recognition and digital image processing techniques. These techniques will help in identifying banana plant diseases thereby increasing the yield of banana . This a survey paper on disease identification and classification of banana crops. A summary of various techniques for disease identification and classification is also done.

Advantage:

Prediction accuracy is high, robust working when training example contain errors.

Disadvantage:

Involve long training time, difficult to understand learned function. large no. of support vectors used from the training set to perform classification task.

6.**Survey on plants disease detection using Machine Learning**

Authors : Preetha S , Musqan

Arshad Agriculture is a significant source of income for Indian people. Experts do the manual method of detecting disease in a plant. For this, a large team was required, and continuous monitoring was required; that was a complicated task when we do this with a large number of crops. In some places, farmers were unaware of the experts, and they do not have proper facilities. In such conditions, one technique can be beneficial in keeping track of and monitoring a large number of crops. This technique is known as Automatic Detection. This technique makes it much easier and cheaper to detect disease. Machine Learning can provide a method and algorithm to detect the disease. There should be the training of images of all types of leaves that include the ones that are healthy and disease leaf images.

Advantage:

Easy to implement and quite good in result.

Disadvantage:

Slow learner, not robust to the noise data in large training example.

7.Plant Disease Detection Techniques

Author : Dr. Rajbir Kaur

Plant diseases cause major losses in terms of production, economy, quality and quantity of agricultural products. Since, 70% of Indian economy is dependent on agricultural yield, there is a need to control the loss incurred by plant diseases. The plants need to be monitored from a very initial stage of their lifecycle to avoid such diseases. The traditional method being followed for this supervision is naked eye observation which is more time-consuming, expensive and a lot of expertise is required. So, in order to speed up this process there is a need to automate the disease detection system. The disease detection system needs to be developed using image processing techniques. Many researchers have developed systems based on various techniques of image processing. This paper reviews the potential of the methods of plant leaves disease detection system that facilitates the advancement in agriculture. It includes various phases such as the image acquisition, image segmentation, feature extraction and classification.

Advantage:

Plant disease detection using any automated technology is advantageous since it eliminates a considerable amount of monitoring work in large crop farms and detects disease signs at an early stage, i.e. when they occur on plant leaves.

Disadvantages:

Require extensive training.

Hard to comprehend taught function.

8.**Disease Detection in Tomato Leaf using Deep Learning Techniques**

Authors : Hepzibah Elizabeth David; K. Ramalakshmi; Hemalatha Gunasekaran; R. Venkatesan

Tomatoes are the most common vegetable crop widely cultivated in the agricultural fields in India. The tropical climate is ideal for its growth, however certain climatic conditions and various other factors affect the normal growth of tomato plants. Apart from these climatic conditions and natural disasters, plant disease is a major crisis in crop production and results in economic loss. The traditional disease detection methods for tomato crops could not produce the expected outcome and the detection period for diseases was slow. The early detection of diseases can give better results than the existing detection models. Thus, computer vision-based technology deep learning techniques could be implemented for earlier disease detection. This paper introduces a comprehensive analysis of the disease classification and detection techniques implied for tomato leaf disease identification. This paper also reviews the merits and drawbacks of the methodologies proposed. This paper finally proposes the early disease detection technique to identify tomato leaf disease using hybrid deep-learning architecture

Advantage:

Prediction accuracy is high and having robust working when training example have error in them.

Disadvantages:

Require long training time.

Difficult to understand learned function

2.2 PROBLEM STATEMENT DEFINITION

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

major threat to food security. Hence, early and accurate identification of plant diseases is essential to ensure high quantity and best quality. In recent years, the number of diseases on plants and the degree of harm caused has increased due to the variation in pathogen varieties, changes in cultivation methods,

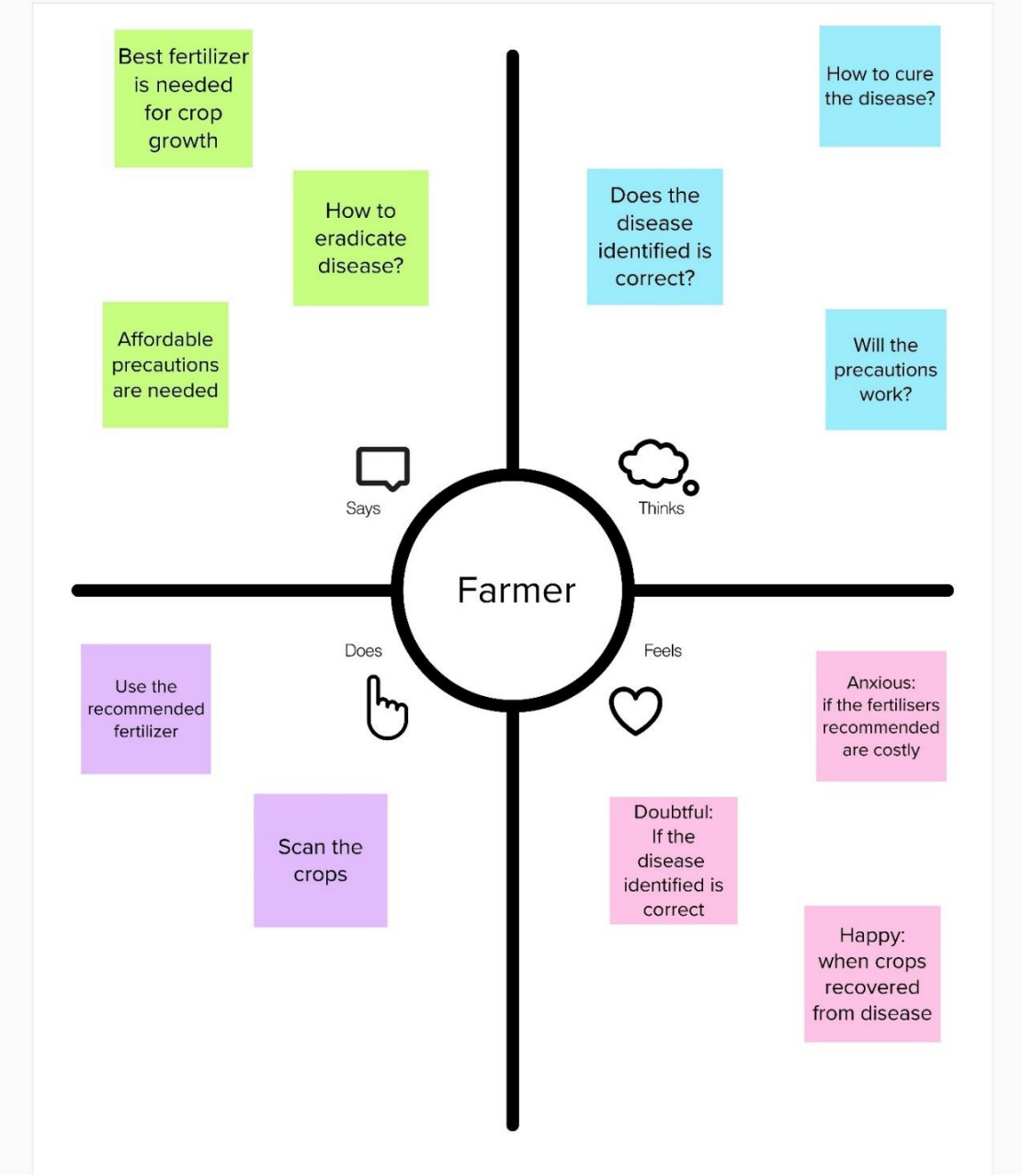
and inadequate plant protection techniques. An automated system is introduced to identify different diseases on plants by checking the symptoms shown on the leaves of the plant.

Deep learning techniques are used to identify the diseases and suggest the precautions that can be taken for those diseases.

3. IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

- Users pains and gain was captured to prepare empathy map and list of problem statement was prepared.
- An empathy map is a collaborative tool teams can use to gain a deeper insight into their customers.
- Much like a user persona, an empathy map can represent a group of users, such as a customer segment.

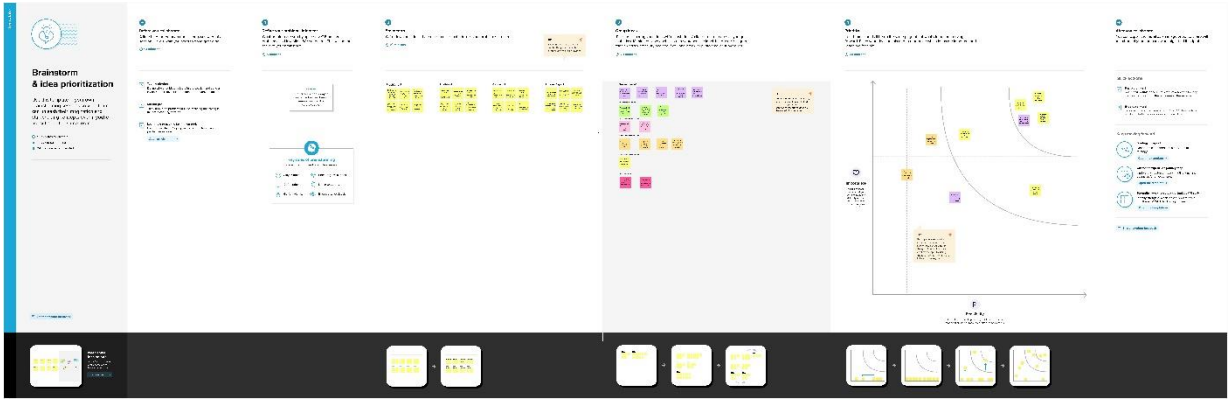


3.2 **IDEATION AND BRAINSTOMING**

- Various brainstorming ideas are organised and based on the feasibility and importance top three ideas were prioritised.

- **Ideation** is the creative process of generating, developing, and communicating new ideas, where an idea is understood as a basic element of thought that can be either visual, concrete, or abstract. Ideation comprises all stages of a thought cycle, from innovation, to development, to actualization.

- **Brainstorming** is a method of generating ideas and sharing knowledge to solve a particular commercial or technical problem, in which participants are encouraged to think without interruption.



3.3 PROPOSED SOLUTIONS

- Prepare the proposed solution document, which includes the novelty, feasibility of idea, business model, social impact, scalability of solution, etc.

Project Design Phase-I Proposed Solution Template

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)	The farmer needs a path to analyse the diseaseon the crop and to choose the best fertilizer to increase the yield.
2.	Idea / Solution description	The farmer can rectify the disease directly and prevent the disease by taking suitable fertilizer and precautions to harvest healthy crops.
3.	Novelty / Uniqueness	Plant disease can be deducted by leveraging the power of deep learning. Users can share their crop's status and find the quick remedy.
4.	Social Impact / Customer Satisfaction	By producing healthy and disease-less crops, society consumes hygiene crops, and stays healthy and happy.

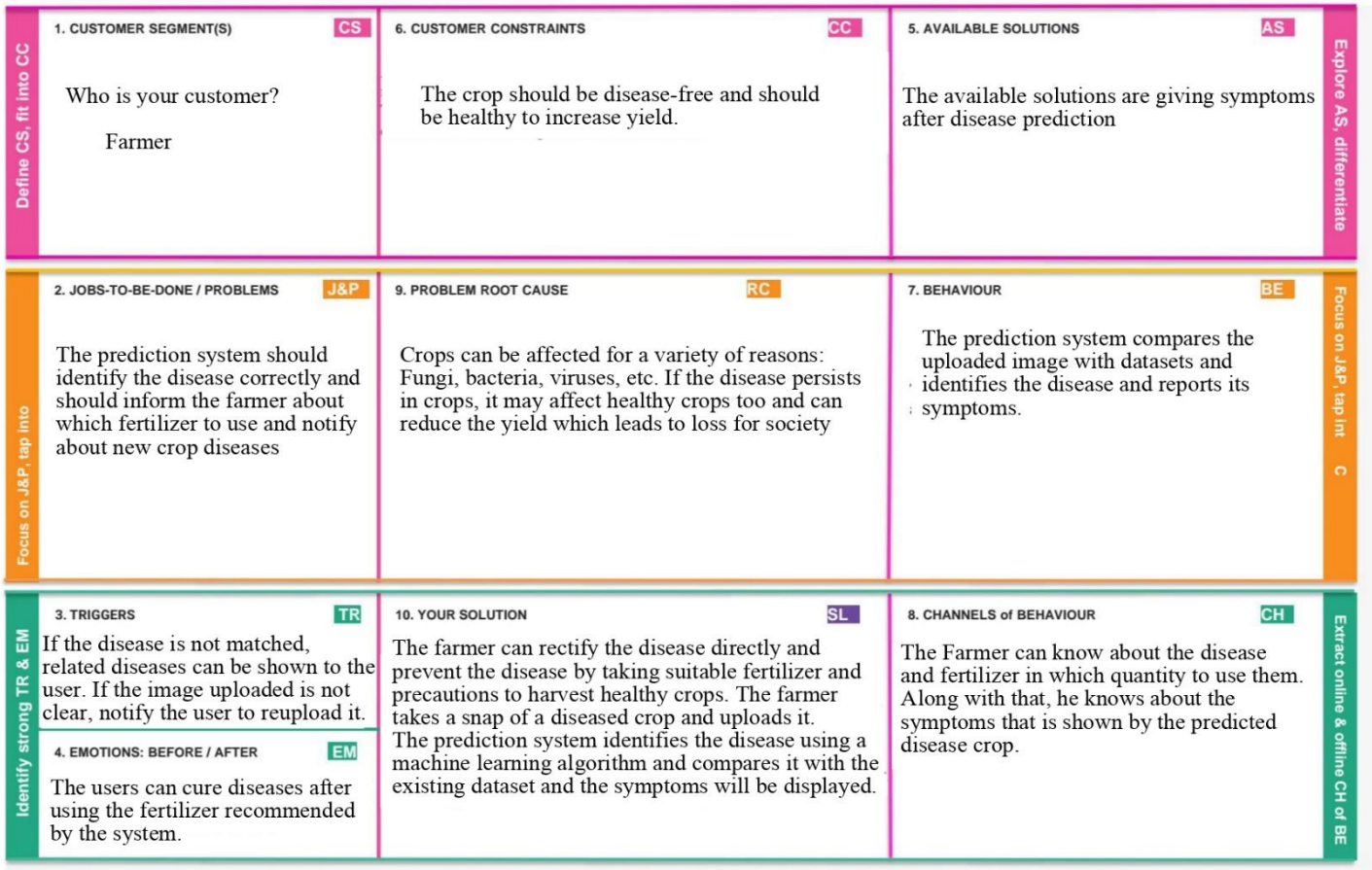
5.	Business Model (Revenue Model)	The revenue can be generated by recommending fertilizers from local markets and E-shopping links can be provided.
6.	Scalability of the Solution	The goal is to create a smart system for monitoring agricultural land and detecting plant diseases. Setups can be created near agricultural lands.

3.4 PROBLEM SOLUTION FIT

- Prepare problem - solution fit document.

- The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem.

Problem-Solution Fit Fertilizers Recommendation System For Disease Prediction



4.REQUIREMENT ANALYSIS

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

4.1 FUNCTIONAL REQUIREMENTS

- Functional requirements may involve calculations, technical details, data manipulation and processing, and other specific functionality that define what a system is supposed to accomplish.

- Behavioral requirements describe all the cases where the system uses the functional requirements, these are captured in use cases.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User details storage	Storing user details via database
FR-4	Scanning crop images	Uploading images of disease via camera
FR-5	Predicting crop symptoms, fertilizers	Display the disease and its symptoms by analysing using dataset.
FR-6	Search results of diseases	Searching the symptoms, prevention steps.

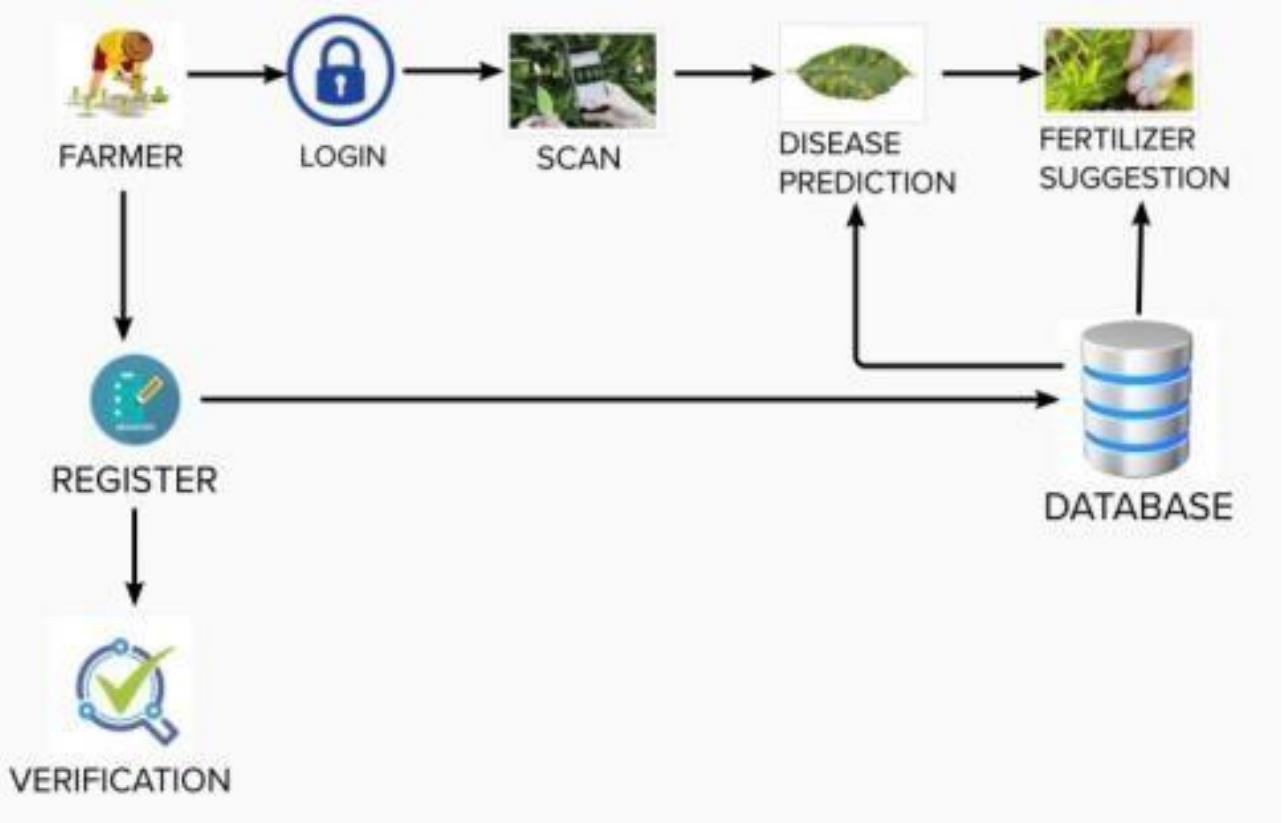
4.2 NON – FUNCTIONAL REQUIREMENTS

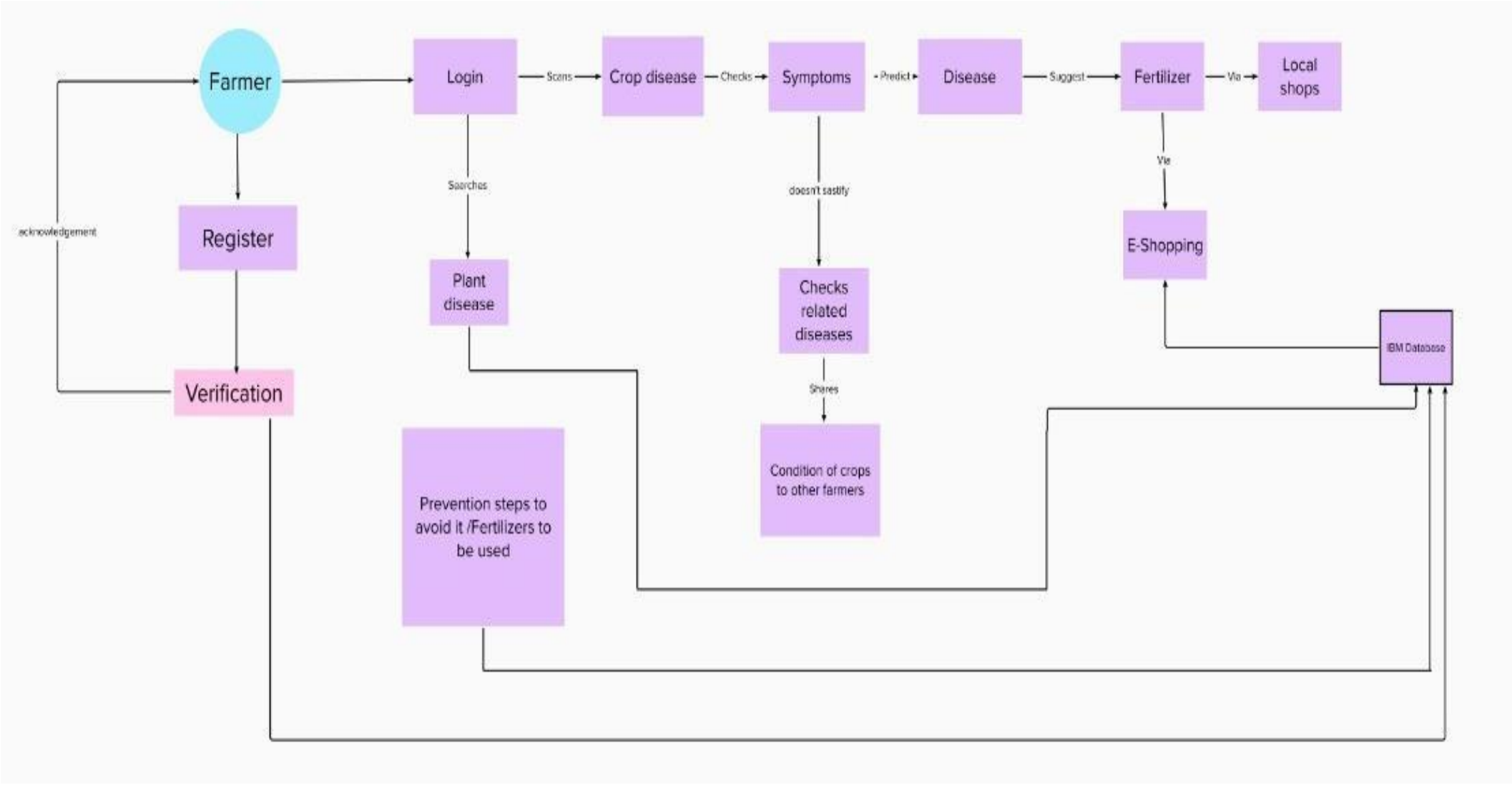
- Nonfunctional Requirements (NFRs) define system attributes such as security, reliability, performance, maintainability, scalability, and usability.

- They serve as constraints or restrictions on the design of the system across the different backlogs.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Scanning images and use the suggested fertilizers.
NFR-2	**Security**	Encrypt the user details and use a firewall to store user details securely.
NFR-3	**Reliability**	Predicting the disease and its symptoms. If symptoms don’t match, display related diseases.
NFR-4	**Performance**	Displaying immediate search results.
NFR-5	**Availability**	Can be accessible easily at any time from database.
NFR-6	**Scalability**	Accurately predict disease even given many images.

5. DATA FLOW DIAGRAMS

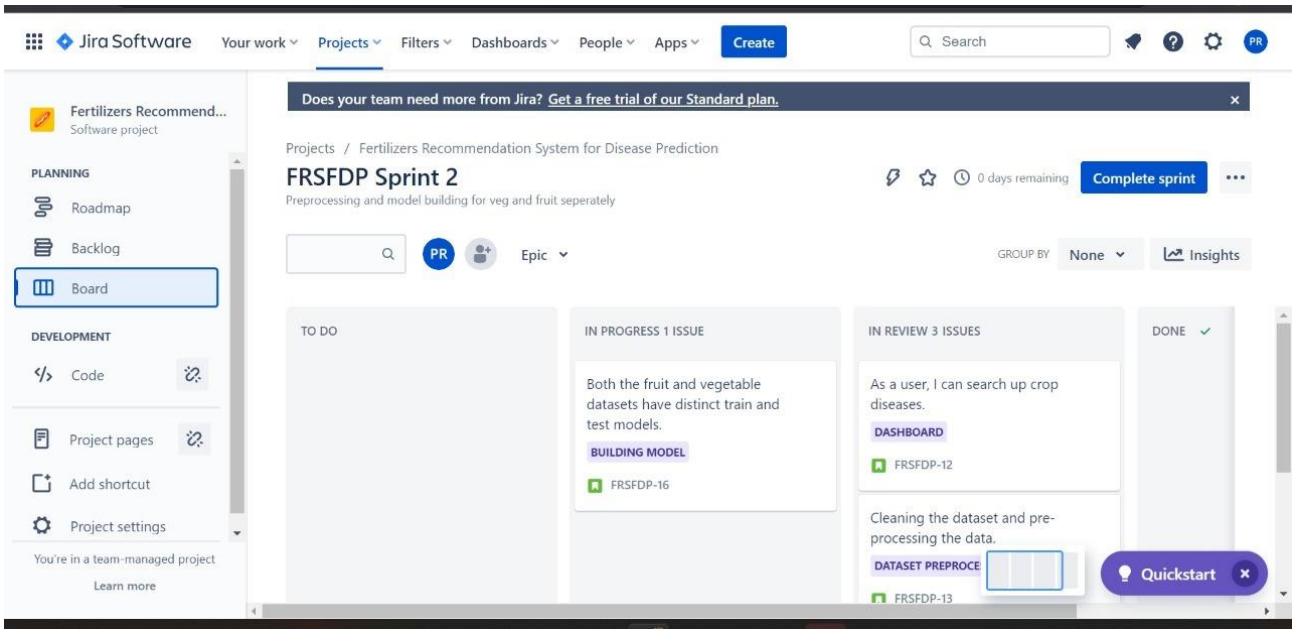



Farmer
Login
Scans
Crop disease
Checks
Symptoms
Predict
Disease
Suggest
Fertilizer
Via
Local shops
acknowledgement
Register
Verification
Searches
Plant disease
doesn't sastify
Checks related diseases
Shares
Condition of crops to other farmers
Prevention steps to avoid it /Fertilizers to be used
Via
E-Shopping
IBM Database

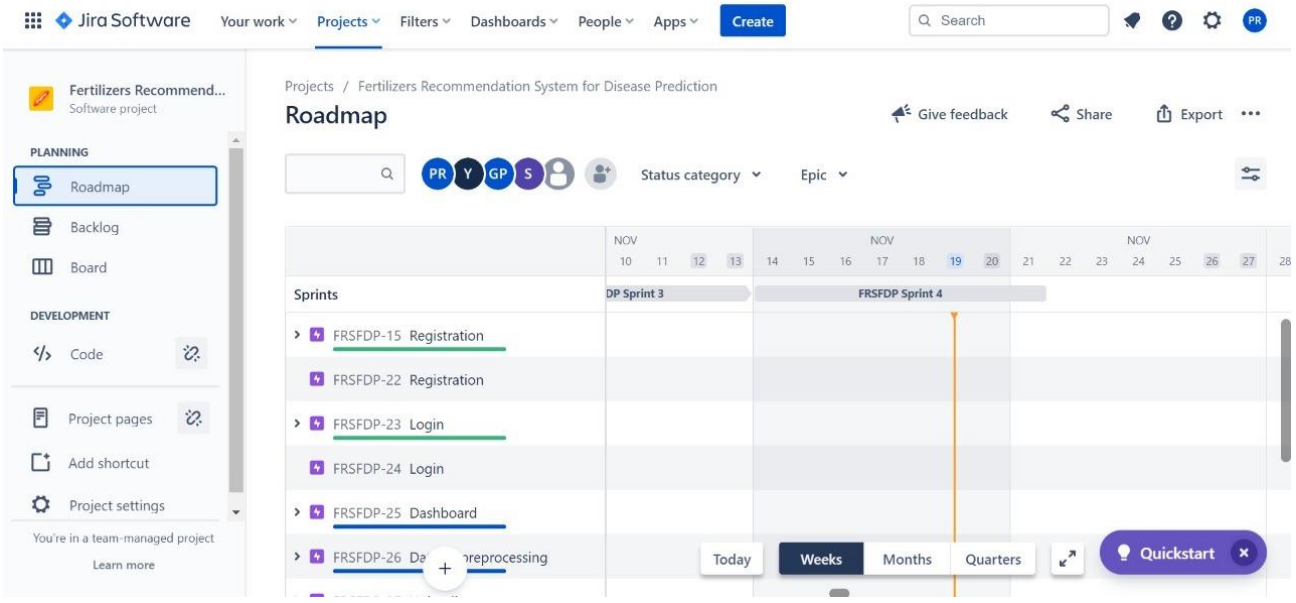
User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account /dashboard		
		USN-2	As a user, I will receive confirmation email or OTP once I have registered for the application	I can receive confirmationemail & click confirm		
		USN-3	As a user, I can register for the applicationthrough the form	I can register & access thedashboard by logging in with the password		
	Login	USN-4	As a user, I can log into the application byentering my email & password	If the password and email match then the user is authenticated.		
	Dashboard	USN-5	As a user, I can search for disease	The disease can be searched directly or crop disease image is uploaded using the phone camera		
Customer (Webuser)	Dashboard	USN-6	As a user, I can search for fertilizer	The disease could be searched directly or crop disease images can beuploaded		

6.PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION



6.2 SPRINT DELIVERY SCHEDULE



7 CODING & SOLUTIONING

DATA COLLECTION AND PRE-PROCESSING

FRUIT

In [94]:

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remou
nt=True).
```

In [ ]:

```
# Import Libraries
import os
import glob
import matplotlib.pyplot as plt
import numpy as np
# Keras API
import keras
from keras.models import Sequential
from keras.layers import Dense,Dropout,Flatten
from keras.layers import Conv2D,MaxPooling2D,Activation,AveragePooling2D,BatchNormalization
from keras.preprocessing.image import ImageDataGenerator
```

In [ ]:

```
# My data is in google drive.
f_train_dir ="/content/drive/MyDrive/Project/Dataset Plant Disease/fruit-dataset/fruit-dataset/train"
f_test_dir="/content/drive/MyDrive/Project/Dataset Plant Disease/fruit-dataset/fruit-dataset/test"
```

In [ ]:

```
# function to get count of images
def get_files(directory):
  if not os.path.exists(directory):
    return 0
  count=0
  for current_path,dirs,files in os.walk(directory):
    for dr in dirs:
      count+= len(glob.glob(os.path.join(current_path,dr+"/*")))
  return count
```

In [ ]:

```
f_train_samples =get_files(f_train_dir)
f_num_classes=len(glob.glob(f_train_dir+"/*"))
f_test_samples=get_files(f_test_dir)
print(f_num_classes,"Classes")
print(f_train_samples,"Train images")
print(f_test_samples,"Test images")
```

```
6 Classes
5334 Train images
1679 Test images
```

In [ ]:

```
# Pre-processing data with parameters.
f_train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
f_test_datagen=ImageDataGenerator(rescale=1./255)
```

In [ ]:

```
# set height and width and color of input image.
img_width,img_height =256,256
input_shape=(img_width,img_height,3)
batch_size =32
```

```
f_train_generator=f_train_datagen.flow_from_directory(f_train_dir,target_size=(img_width,img_height),batch_size=bat
ch_size)
f_test_generator=f_test_datagen.flow_from_directory(f_test_dir,shuffle=True,target_size=(img_width,img_height),batc
h_size=batch_size)
Found 5334 images belonging to 6 classes.
Found 1679 images belonging to 6 classes.
In [ ]:
f_train_generator.class_indices
Out[ ]:
{'Apple___Black_rot': 0,
 'Apple___healthy': 1,
 'Corn_(maize)___Northern_Leaf_Blight': 2,
 'Corn_(maize)___healthy': 3,
 'Peach___Bacterial_spot': 4,
 'Peach___healthy': 5}
```

Model Building

```
In [ ]:
from keras_preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Convolution2D,MaxPool2D,Flatten
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import glob
from keras.layers import Dense,Dropout,Flatten
from keras.layers import Conv2D,MaxPooling2D,Activation,AveragePooling2D,BatchNormalization
from keras.preprocessing import image
from tensorflow.keras.utils import load_img
from tensorflow.keras.utils import img_to_array
import numpy as np
import matplotlib.pyplot as plt
from tensorflow import keras
In [ ]:
# CNN building.
model = Sequential()
model.add(Conv2D(32, (5, 5),input_shape=input_shape,activation='relu'))
model.add(MaxPooling2D(pool_size=(3, 3)))
model.add(Conv2D(32, (3, 3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, (3, 3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(512,activation='relu'))
model.add(Dropout(0.25))
model.add(Dense(128,activation='relu'))
model.add(Dense(f_num_classes,activation='softmax'))
model.summary()
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 252, 252, 32)      2432

 max_pooling2d (MaxPooling2D  (None, 84, 84, 32)       0
```

)

conv2d_1 (Conv2D) (None, 82, 82, 32) 9248

max_pooling2d_1 (MaxPooling 2D) (None, 41, 41, 32) 0

conv2d_2 (Conv2D) (None, 39, 39, 64) 18496

max_pooling2d_2 (MaxPooling 2D) (None, 19, 19, 64) 0

flatten (Flatten) (None, 23104) 0

dense (Dense) (None, 512) 11829760

dropout (Dropout) (None, 512) 0

dense_1 (Dense) (None, 128) 65664

dense_2 (Dense) (None, 6) 774

===
Total params: 11,926,374
Trainable params: 11,926,374
Non-trainable params: 0
_____

In [ ]:

```
from keras.preprocessing import image
from tensorflow.keras.utils import load_img
from tensorflow.keras.utils import img_to_array
import numpy as np
img1 = load_img('/home/8e0669bb-1959-4f28-b98d-da3c2d85396c___RS_NLB 3678.JPG')
plt.imshow(img1);
#preprocess image
img1 = load_img('/home/8e0669bb-1959-4f28-b98d-da3c2d85396c___RS_NLB 3678.JPG',target_size=(256, 256))
img = img_to_array(img1)
img = img/255
img = np.expand_dims(img, axis=0)
```

In [ ]:

```
validation_generator = f_train_datagen.flow_from_directory(f_train_dir,target_size=(img_height,
img_width),batch_size=batch_size)
opt=keras.optimizers.Adam(lr=0.001)
model.compile(optimizer=opt,loss='categorical_crossentropy',metrics=['accuracy'])
nb_epoch = 10
train=model.fit_generator(f_train_generator,epochs=nb_epoch,steps_per_epoch=f_train_generator.samples//batch_size,
validation_data=validation_generator,validation_steps=validation_generator.samples // batch_size,verbose=1)
```

Found 5334 images belonging to 6 classes.
/usr/local/lib/python3.7/dist-packages/keras/optimizers/optimizer_v2/adam.py:110: UserWarning: The `lr` argument is d
eprecated, use `learning_rate` instead.
super(Adam, self).__init__(name, **kwargs)
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: UserWarning: `Model.fit_generator` is deprecated and
will be removed in a future version. Please use `Model.fit`, which supports generators.
"""
Epoch 1/10
166/166 [==============================] - 1385s 8s/step - loss: 0.7518 - accuracy: 0.7212 - val_loss: 0.3305
- val_accuracy: 0.8869

Epoch 2/10
166/166 [==============================] - 676s 4s/step - loss: 0.2807 - accuracy: 0.9002 - val_loss: 0.1838 - val_accuracy: 0.9386
Epoch 3/10
166/166 [==============================] - 675s 4s/step - loss: 0.2477 - accuracy: 0.9113 - val_loss: 0.1738 - val_accuracy: 0.9405
Epoch 4/10
166/166 [==============================] - 676s 4s/step - loss: 0.1957 - accuracy: 0.9334 - val_loss: 0.1340 - val_accuracy: 0.9516
Epoch 5/10
166/166 [==============================] - 675s 4s/step - loss: 0.1485 - accuracy: 0.9464 - val_loss: 0.2552 - val_accuracy: 0.9228
Epoch 6/10
166/166 [==============================] - 664s 4s/step - loss: 0.1325 - accuracy: 0.9562 - val_loss: 0.0999 - val_accuracy: 0.9644
Epoch 7/10
166/166 [==============================] - 672s 4s/step - loss: 0.1044 - accuracy: 0.9645 - val_loss: 0.0792 - val_accuracy: 0.9723
Epoch 8/10
166/166 [==============================] - 669s 4s/step - loss: 0.0923 - accuracy: 0.9704 - val_loss: 0.0489 - val_accuracy: 0.9838
Epoch 9/10
166/166 [==============================] - 676s 4s/step - loss: 0.0759 - accuracy: 0.9757 - val_loss: 0.0545 - val_accuracy: 0.9812
Epoch 10/10
166/166 [==============================] - 672s 4s/step - loss: 0.0824 - accuracy: 0.9723 - val_loss: 0.0769 - val_accuracy: 0.9740
In [ ]:

```
# Save model
from keras.models import load_model
model.save('fruit.h5')
```

IBM Deployment

IBM Deployment

In [ ]:

```
!pip install ibm_watson_machine_learning
```

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting ibm_watson_machine_learning
  Downloading ibm_watson_machine_learning-1.0.257-py3-none-any.whl (1.8 MB) 1.8 MB 5.2 MB/s
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.7/dist-packages (from ibm_watson_machine_learning) (4.13.0)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.7/dist-packages (from ibm_watson_machine_learning) (1.24.3)
Collecting ibm-cos-sdk==2.7.*
  Downloading ibm-cos-sdk-2.7.0.tar.gz (51 kB)51 kB 740 kB/s
Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-packages (from ibm_watson_machine_learning) (21.3)
Requirement already satisfied: certifi in /usr/local/lib/python3.7/dist-packages (from ibm_watson_machine_learning) (2022.9.24)
Requirement already satisfied: pandas<1.5.0,>=0.24.2 in /usr/local/lib/python3.7/dist-packages (from ibm_watson_machine_learning) (1.3.5)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from ibm_watson_machine_learning) (2.23.0)
Requirement already satisfied: tabulate in /usr/local/lib/python3.7/dist-packages (from ibm_watson_machine_learning) (0.8.10)
Collecting lomond
  Downloading lomond-0.3.3-py2.py3-none-any.whl (35 kB)
Collecting ibm-cos-sdk-core==2.7.0

```
  Downloading ibm-cos-sdk-core-2.7.0.tar.gz (824 kB) 824 kB 52.4 MB/s
Collecting ibm-cos-sdk-s3transfer==2.7.0
  Downloading ibm-cos-sdk-s3transfer-2.7.0.tar.gz (133 kB)133 kB 61.3 MB/s
Collecting jmespath<1.0.0,>=0.7.1
  Downloading jmespath-0.10.0-py2.py3-none-any.whl (24 kB)
Collecting docutils
  Downloading docutils-0.15.2-py3-none-any.whl (547 kB)|547 kB 64.8 MB/s
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /usr/local/lib/python3.7/dist-packages (from ibm-cos-sdk-core==2.7.0->ibm-cos-sdk==2.7.*->ibm_watson_machine_learning) (2.8.2)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.7/dist-packages (from pandas<1.5.0,>=0.24.2->ibm_watson_machine_learning) (1.21.6)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (from pandas<1.5.0,>=0.24.2->ibm_watson_machine_learning) (2022.6)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from python-dateutil<3.0.0,>=2.1->ibm-cos-sdk-core==2.7.0->ibm-cos-sdk==2.7.*->ibm_watson_machine_learning) (1.15.0)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests->ibm_watson_machine_learning) (2.10)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests->ibm_watson_machine_learning) (3.0.4)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from importlib-metadata->ibm_watson_machine_learning) (3.10.0)
Requirement already satisfied: typing-extensions>=3.6.4 in /usr/local/lib/python3.7/dist-packages (from importlib-metadata->ibm_watson_machine_learning) (4.1.1)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/dist-packages (from packaging->ibm_watson_machine_learning) (3.0.9)
Building wheels for collected packages: ibm-cos-sdk, ibm-cos-sdk-core, ibm-cos-sdk-s3transfer
  Building wheel for ibm-cos-sdk (setup.py) ... done
  Created wheel for ibm-cos-sdk: filename=ibm_cos_sdk-2.7.0-py2.py3-none-any.whl size=72563 sha256=8de5211f749cce223db4e4361c865c7aca195f7315169eda46b82c15299da3b2
  Stored in directory: /root/.cache/pip/wheels/47/22/bf/e1154ff0f5de93cc477acd0ca69abfbb8b799c5b28a66b44c2
  Building wheel for ibm-cos-sdk-core (setup.py) ... done
  Created wheel for ibm-cos-sdk-core: filename=ibm_cos_sdk_core-2.7.0-py2.py3-none-any.whl size=501013 sha256=cbae045d61e3ab8c2ad10b9fc4abc68d24f8de92982a7fee262a6480270d9678
  Stored in directory: /root/.cache/pip/wheels/6c/a2/e4/c16d02f809a3ea998e17cfd02c13369281f3d232aaf5902c19
  Building wheel for ibm-cos-sdk-s3transfer (setup.py) ... done
  Created wheel for ibm-cos-sdk-s3transfer: filename=ibm_cos_sdk_s3transfer-2.7.0-py2.py3-none-any.whl size=88622 sha256=6af880aabeb9b2100f20ed8667c00d8d375b8d9f6ccafd1454dd501096fa3480
  Stored in directory: /root/.cache/pip/wheels/5f/b7/14/fbe02bc1ef1af890650c7e51743d1c83890852e598d164b9da
Successfully built ibm-cos-sdk ibm-cos-sdk-core ibm-cos-sdk-s3transfer
Installing collected packages: jmespath, docutils, ibm-cos-sdk-core, ibm-cos-sdk-s3transfer, lomond, ibm-cos-sdk, ibm-watson-machine-learning
  Attempting uninstall: docutils
    Found existing installation: docutils 0.17.1
    Uninstalling docutils-0.17.1:
      Successfully uninstalled docutils-0.17.1
Successfully installed docutils-0.15.2 ibm-cos-sdk-2.7.0 ibm-cos-sdk-core-2.7.0 ibm-cos-sdk-s3transfer-2.7.0 ibm-watson-machine-learning-1.0.257 jmespath-0.10.0 lomond-0.3.3
In [ ]:
from ibm_watson_machine_learning import APIClient
In [ ]:
wml_credentials = {
   "url" : "https://us-south.ml.cloud.ibm.com",
   "apikey" : "oWwMptIWYMhIEjIrCdapePv5oA13N4YrsxfqbO9xC22y"
}
In [ ]:
client = APIClient(wml_credentials)
Python 3.7 and 3.8 frameworks are deprecated and will be removed in a future release. Use Python 3.9 framework instead.
```

In [ ]:
client
Out[ ]:
In [ ]:
client.spaces.list()
Note: 'limit' is not provided. Only first 50 records will be displayed if the number of records exceed 50

ID NAME CREATED
cd3c5446-b337-4425-bbca-31ad7f4ec0be Fertilizers Recommendation System Deployment 2022-11-18T23:19:38.565Z

In [ ]:
space_uid = "cd3c5446-b337-4425-bbca-31ad7f4ec0be"
In [ ]:
client.set.default_space(space_uid)
Out[ ]:
'SUCCESS'
In [ ]:
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
autoai-kb_rt22.2-py3.10 125b6d9a-5b1f-5e8d-972a-b251688ccf40 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
autoai-ts_rt22.2-py3.10 396b2e83-0953-5b86-9a55-7ce1628a406f 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
pytorch-onnx_rt22.2-py3.10 40e73f55-783a-5535-b3fa-0c8b94291431 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
runtime-22.2-py3.10-xc        5e8cddff-db4a-5a6a-b8aa-2d4af9864dab  base
autoai-kb_3.1-py3.7           632d4b22-10aa-5180-88f0-f52dfb6444d7  base
----------------------------  ------------------------------------  ----
Note: Only first 50 records were displayed. To display more use 'limit' parameter.
```

In [ ]:

```
software_spec_uid = client.software_specifications.get_uid_by_name("tensorflow_rt22.1-py3.9")
```

In [ ]:

```
software_spec_uid
```

Out[ ]:

```
'acd9c798-6974-5d2f-a657-ce06e986df4d'
```

Test the Model

Test the Model

In [ ]:

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

In [ ]:

```
test_dir="/content/drive/MyDrive/Project/Dataset Plant Disease/fruit-dataset/fruit-dataset/test"
```

In [ ]:

```
test_dir
```

Out[ ]:

```
'/content/drive/MyDrive/Project/Dataset Plant Disease/fruit-dataset/fruit-dataset/test'
```

In [ ]:

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

In [ ]:

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

In [ ]:

```
model = tf.keras.models.load_model("/content/fruit.h5")
```

In [ ]:

```
test_datagen_1=ImageDataGenerator(rescale=1)
test_generator_1=test_datagen_1.flow_from_directory(
    test_dir,
    target_size=(128,128),
    batch_size=20,
    class_mode='categorical'
)
```

```
Found 1679 images belonging to 6 classes.
```

In [ ]:

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

In [ ]:

```
img=load_img('/content/9f0564fa-2184-4df2-b0b9-bec3f36294e8___R.S_HL 0651 copy 2.jpg')
```

In [ ]:

```
img
Out[ ]:

In [ ]:
x=img_to_array(img)
x=np.expand_dims(x,axis=0)
In [ ]:
img=image.load_img(r'/content/9f0564fa-2184-4df2-b0b9-bec3f36294e8___R.S_HL 0651 copy 2.jpg')
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', 'Corn_(maize)___healthy', 'Corn_(maize)___Northern_Leaf_Blight',
'Peach___Bacterial_spot', 'Peach___healthy']
index[y[0]]
1/1 [==============================] - 0s 44ms/step
Out[ ]:
'Corn_(maize)___Northern_Leaf_Blight'
In [ ]:
model.evaluate(f_test_generator,steps=50)
50/50 [==============================] - 355s 7s/step - loss: 0.1048 - accuracy: 0.9675
Out[ ]:
[0.104802206158638, 0.9674999713897705]
```

APP.PY

```
# -- coding: utf-8 --

"""app.ipynb

Automatically generated by Colaboratory.

Original file is located at

    https://colab.research.google.com/drive/1dfHEhozcH-XTnw7QkWknrW_zVOilv2pP

"""


import requests

from tensorflow.keras.preprocessing import image

from tensorflow.keras.models import load_model

import numpy as np

import pandas as pd

import tensorflow as tf

from flask import Flask, request, render_template, redirect, url_for

import os

from werkzeug.utils import secure_filename

from tensorflow.python.keras.backend import set_session
```

```
app=Flask(_name_)
model = load_model("vegetable.h5")
model1 = load_model("fruit.h5")

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

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

@app.route('/predict',methods=['GET','POST'])
def predict():
    if request.method == 'POST':
        #getting file from post request
        f=request.files['image']
        #save the files to uploads
        basepath = os.path.dirname(os.getcwd())
        file_path = os.path.join(basepath, '\\Users\\Inspiron15 3000\\Desktop\\Application Building\\uploads', secure_filename(f.filename))
        f.save(file_path)
        img = image.load_img(file_path, target_size=(256,256))
        x=image.img_to_array(img)
        x=np.expand_dims(x,axis=0)
        plant=request.form['plants']
        print(plant)
        p = ""
```

```
        disease = ""
        caution = ""
        if(plant=="Vegetable"):
            y = np.argmax(model.predict(x),axis=1)
            df=pd.read_excel('precautions - veg.xlsx')
            caution = df.iloc[y[0]]['caution']
            p = df.iloc[y[0]]['plant']
            disease = df.iloc[y[0]]['disease']
        else:
            y = np.argmax(model1.predict(x),axis=1)
            df=pd.read_excel('precautions - fruits.xlsx')
            caution = df.iloc[y[0]]['caution']
            p = df.iloc[y[0]]['plant']
            disease = df.iloc[y[0]]['disease']

        return render_template('predict.html', plant=p , disease = disease , caution = caution)

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

8.RESULT

PERFORMANCE:

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	3	19
Duplicate	9	6	3	6	24
External	2	7	0	1	10
Fixed	11	4	3	20	38
Not Reproduced	3	2	1	0	6
Skipped	5	3	1	1	10
Won't Fix	4	5	2	1	12
Totals	44	31	13	32	119

3. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	17	0	0	17
Client Application	51	0	0	51
Security	21	0	0	21
Outsource Shipping	7	0	0	7
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

10. ADVANTAGES

1.Achieving the maximum crop at minimum yield is the ultimate Aim of the project.

2.Early detection of problems and management of that problems can help the farmers for better crop yield.

3.For the better understanding of the crop yield, we need to study of the huge data with the help of machine learning algorithm so it will give the accurate yield for that crop and suggest the farmer for a better crop.

11. CONCLUSION AND FUTURE SCOPE

The prediction of crop yield based on location and proper implementation of algorithms have proved that the higher crop yield can be achieved. From above work I conclude that for soil classification Random Forest is good with accuracy 86.35% compare to Support Vector Machine. For crop yield prediction Support Vector Machine is good with accuracy 99.47% compare to Random Forest algorithm. The work can be extended further to add following functionality. Mobile application can be build to help farmers by uploading image of farms. Crop diseases detection using image processing in which user get pesticides based on disease images. Implement Smart Irrigation System for farms to get higher yield.

12. REFERENCES

http://www.ijstr.org/final-print/nov2019/Fertilizers-Recommendation-System-For-Disease-Prediction-In-Tree-Leave.pdf

https://extension.sdstate.edu/sites/default/files/2019-03/P-00039_0.pdf

13. APPENDIX

https://github.com/IBM-EPBL/IBM-Project-8421-1658918783