CONTENTS

ABSTRACT

Agriculture is the main aspect of the country’s development. Many people lead their life in the agriculture field, which gives fully related to agricultural products. Plant disease, especially on leaves, is one of the major factors of reductions in both the quality and quantity of 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 in agriculture preservation. Hence, 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 in plants by checking the symptoms shown on the leaves of the plant. Here, we are aiming to use Deep Learning for plant disease detection based on images of a leaf of the plant. Deep learning has a convolution neural network that is used to find features from the leaf of the plant and matches the input image with the dataset. The goal of our application is to develop a system that recognizes crop diseases for the user by taking an image of a plant leaf, Image processing starts with the digitized color image of the diseased leaf. The disease prediction is done based on the accuracy of the training model, when the disease is predicted our application provides a detailed description of the disease and then recommends the suitable fertilizers required to counterattack that disease.

CHAPTER 1
INTRODUCTION

1.1 PROJECT OVERVIEW

In recent years, the growth of the internet has increased tremendously and the applications of the internet has increased, this results in the availability of huge amounts of various forms of data across the internet and usage of that data for the analysis of several applications has been increased and the technology is also growing on side by side .The various forms of the data is been collected as a big data and information is generated from that big data and it's been processed and analyzed for a particular purpose. Machine Learning techniques are used. Machine Learning (ML) is a technology which is a subset of artificial intelligence in which the computer machine is trained based on the labeled data set and works according to the trained set of data and the predictions will occur or the algorithm of the machine is trained in the way in which it will learn from the from the examples of the environment in which it is been deployed, makes experience and provides the predictions of that environment, without explicitly programming computers. Machine learning focuses on the development of computer programs that can access data and use it to learn and derive the relevant functions according to the dataset.

As the Technological industry grows, the development of technology in agriculture has its own phase and modern agricultural practices have been invoked in farming by educated farmers. Agriculture plays a major role in human life and civilization and part of the economic growth of the nation depends on the trades of agriculture of that particular nation. There are several factors that influence agriculture and their practices. One of the main factors which influence agriculture is Disease Prediction. Compared to all other factors required for successful agriculture practice, disease prediction plays an important role than all of the others, due to current environmental conditions and human habits. As the environmental conditions change it is a challenging task to maintain a good and healthy plant in farming lands so disease prediction plays a major role in agriculture to prevent its extremity, Agriculture needs as much attention as any other field in the current situation because it leads to loss of crops and starvation.

Plant disease is caused by several pathogenic organisms such as bacteria, fungus, nematodes, viruses which grow on the plants and lead to hamper the growth and development of the plants. Disease are transmitted through the nature factors such as wind, pests, water, humans, animals and birds, contaminated equipment, infected planting materials, physical contact between the affected plant and healthy plant. This leads to the minimization of the quality and quantity of the plant products and the plant will die prematurely.

There are some of the examples of the symptoms of disease in plants they are, Nutrient deficiency symptoms(Blossom end rot is a calcium deficiency), Leaf spots caused by bacteria or fungi Discoloration (yellowing; mottling- colored spots, chlorosis) may be caused by viruses, Fertilizer burn, Pesticide burn

The project which comprises two datasets named fruit dataset and vegetable dataset are collected. The fruit data set contains six [6] classes in which the fruits dataset are classified into healthy data set and disease affected dataset and contains Apple, Corn, Peach. The vegetable data set contains six [9] classes in which the vegetable dataset are classified into healthy data set and disease affected dataset and contains Pepper bell, Potato, Tomato. The collected datasets are trained and tested with a deep learning neural network named
Convolutional Neural Networks (CNN). First, the fruit dataset is trained and then tested with CNN. It has 6 classes and all the classes are trained and tested. Second, the vegetable dataset is trained and tested. The software used for training and testing of datasets is Python. All the
Python codes are first written in Jupyter notebook supplied along with Anaconda Python and then the codes are tested in the IBM cloud. Finally a web based framework is designed with help of Flask, a Python library. There are 2 html files created in templates folder along with their associated files in the static folder. The Python program 'python.py' used to interface with these two web pages is written in Spyder-Anaconda python and prediction for the disease has been taken out.

1.2 PURPOSE

The main goal of the project is to identify the various types of disease in the plants by identifying the symptoms which are shown on the leaves of the plants before the disease gets extremity and loss of the crops by performing the above mentioned tasks the diseases in the plants can be minimized and controlled by providing the precautions for the diseases and fertilizers is been recommended for the growth of the plants. Notification of government subsidies which are related to agriculture has been provided to the farmer. This helps in providing the fertilizers and the subsidies which are related to the plants in being given to the farmer.

CHAPTER 2
LITERATURE SURVEY

2.1 EXISTING PROBLEM

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

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

2.2 REFERENCES

[1] Reyes Angie .K, Juan C. Caicedo, and Jorge E.Camargo, "Fine-tuning Deep Convolutional Networks for Plant Recognition", In CLEF (Working Notes),2015.

[2] Hamrouni .L, Aiadi .O, Khaldi .B and Kherfi .M.L, "Plants Species Identification using Computer Vision Techniques", Revue des Bioressources 7, no. 1,2018.

[3] Dimitrovski, Ivica, GjorgjiMadjarov, DragiKocev, and PetreLameski, "Maestra at LifeCLEF 2014 Plant Task: Plant Identification using Visual Data", In CLEF (Working Notes), pp. 705-714, 2014.

[4] Sue Han, CheeSeng Chan, Paul Wilkin, and Paolo Remagnino, "Deep-plant: Plant identification with convolutional neural networks", In Image Processing (ICIP),2015 IEEE International Conference on, pp.452-456,IEEE ,2015.

[5] Kaur, Lakhvir, and Vijay Laxmi, "A Review on Plant Leaf Classification and Segmentation", International Journal Of Engineering And Computer Science 5, no. 8, 2016.

[6] Kadir, Abdul, Lukito Edi Nugroho, AdhiSusanto, and Paulus InsapSantosa, "Leaf classification using shape, color, and texture features", arXiv preprint arXiv:1401.4447, 2013.

[7] Lee, Sue Han, CheeSeng Chan, Simon Joseph Mayo, and Paolo Remagnino, "How deep learning extracts and learns leaf features for plant classification", Pattern Recognition 71, pp: 1-13, 2017

[8] Zeiler, Matthew D., and Rob Fergus, "Visualizing and understanding convolutional networks", In European conference on computer vision, pp. 818-833. Springer, Cham, 2014.

[9] Satnam Singh and Manjit Singh Bhamrah, "Leaf identification using feature extraction and neural network", IOSR Journal of Electronics and Communication Engineering 5, pp: 134-140, 2015.

[10] Vijayashree .T and Gopal .A, "Authentication of Leaf Image Using Image Processing Technique", ARPN Journal of Engineering and Applied Sciences 10, no. 9, pp: 4287-4291, 2015. [11] Mohanty, Sharada P., David P. Hughes, and Marcel Salathé, "Using deep learning for imagebased plant disease detection", Frontiers in plant science 7, pp: 1419, 2016.

[12]Pujari, Devashish, Rajesh Yakkundimath, and Abdul Munaf S. Byadgi. "SVM and ANN based classification of plant diseases using feature reduction techniques." IJIMAI 3, no. 7 (2016): 6-14. [13]Lee, Sue Han, Chee Seng Chan, and Paolo Remagnino. "Multi-organ plant classification based on convolutional and recurrent neural networks." IEEE Transactions on Image Processing 27, no. 9 (2018): 4287-4301.

[14] Ramesh, S., and D. Vydeki. "Application of machine learning in the detection of blast disease in South Indian rice crops." Journal of Phytology (2019).

[15] Yalcin, Hulya, and Salar Razavi. "Plant classification using convolutional neural networks."In 2016 Fifth International Conference on Agro-Geoinformatics (Agro-Geoinformatics), pp. 1-5.IEEE, 2016.

[16] Lee, Sue Han, Chee Seng Chan, Simon Joseph Mayo, and Paolo Remagnino. "How deep learning extracts and learns leaf features for plant classification." Pattern Recognition 71 (2017): 1-13.

2.3 PROBLEM STATEMENT DEFINITION

This Defines the Problem statement to understand Customer’s point of view. The Customer Problem Statement template helps us focus on what matters to create experiences people will love.

A well-articulated Customer problem statement allows us and our team to find the ideal solution for the challenges your customers face. Throughout the process, you’ll also be able to empathize with our Customers, which helps us better understand how they perceive our product or service.

CHAPTER 3

IDEATION & PROPOSED SOLUTION

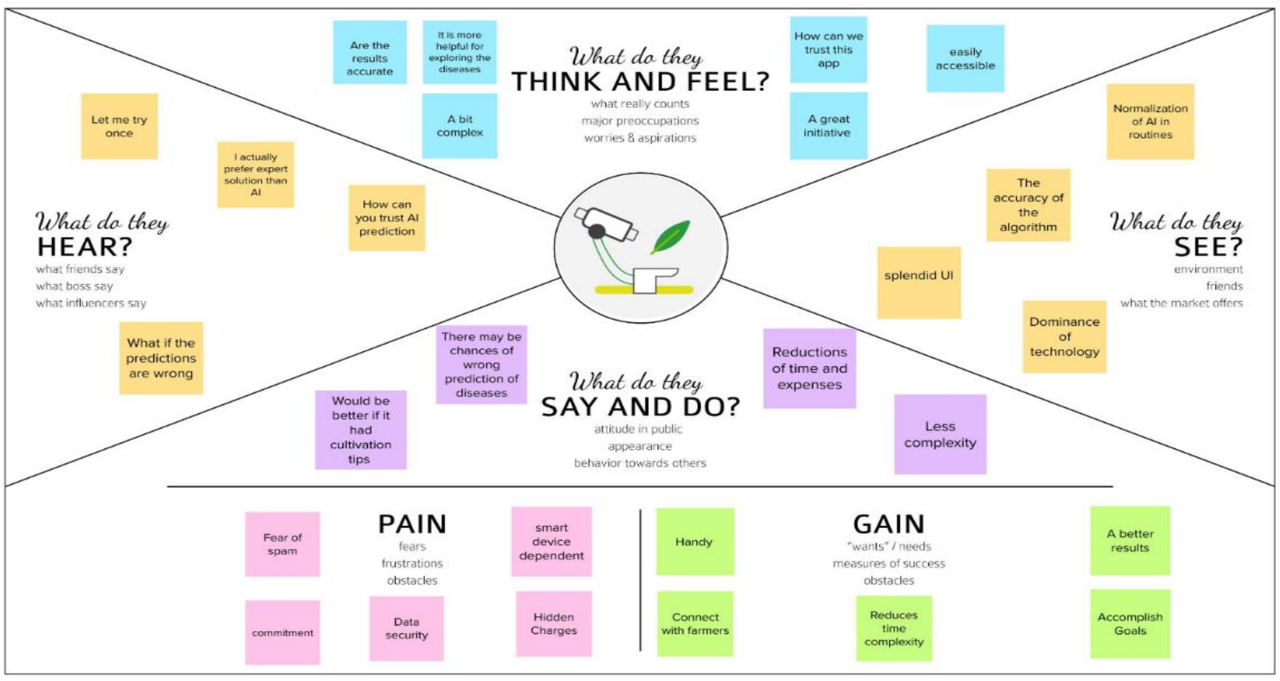
3.1 EMPATHY MAP CANVAS

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviors and attitudes. It is a useful tool to help teams better understand their users. Empathy mapping is a simple workshop activity that can be done with stakeholders, marketing and sales, product development, or creative teams to build empathy for end users. For teams involved in the design and engineering of products, services, or experiences, an empathy mapping session is a great exercise for groups to "get inside the heads" of users.

Empathy maps are most useful at the beginning of the design process after user research but before requirements and concepting. The mapping process can help synthesize research observations and reveal deeper insights about a user's needs The benefits include:

- Better understanding of the user
- Distilled information in one visual reference
- Callouts of key insights from research
- Fast and inexpensive
- Easily customizable based on available information and goals
- Common understanding among teams

The maps can also be used throughout the design process and revised as new data becomes available. A sparsely populated map or a session that reveals more questions than answers indicates where more user research needs to be done.



3.2 IDEATION & BRAINSTORMING

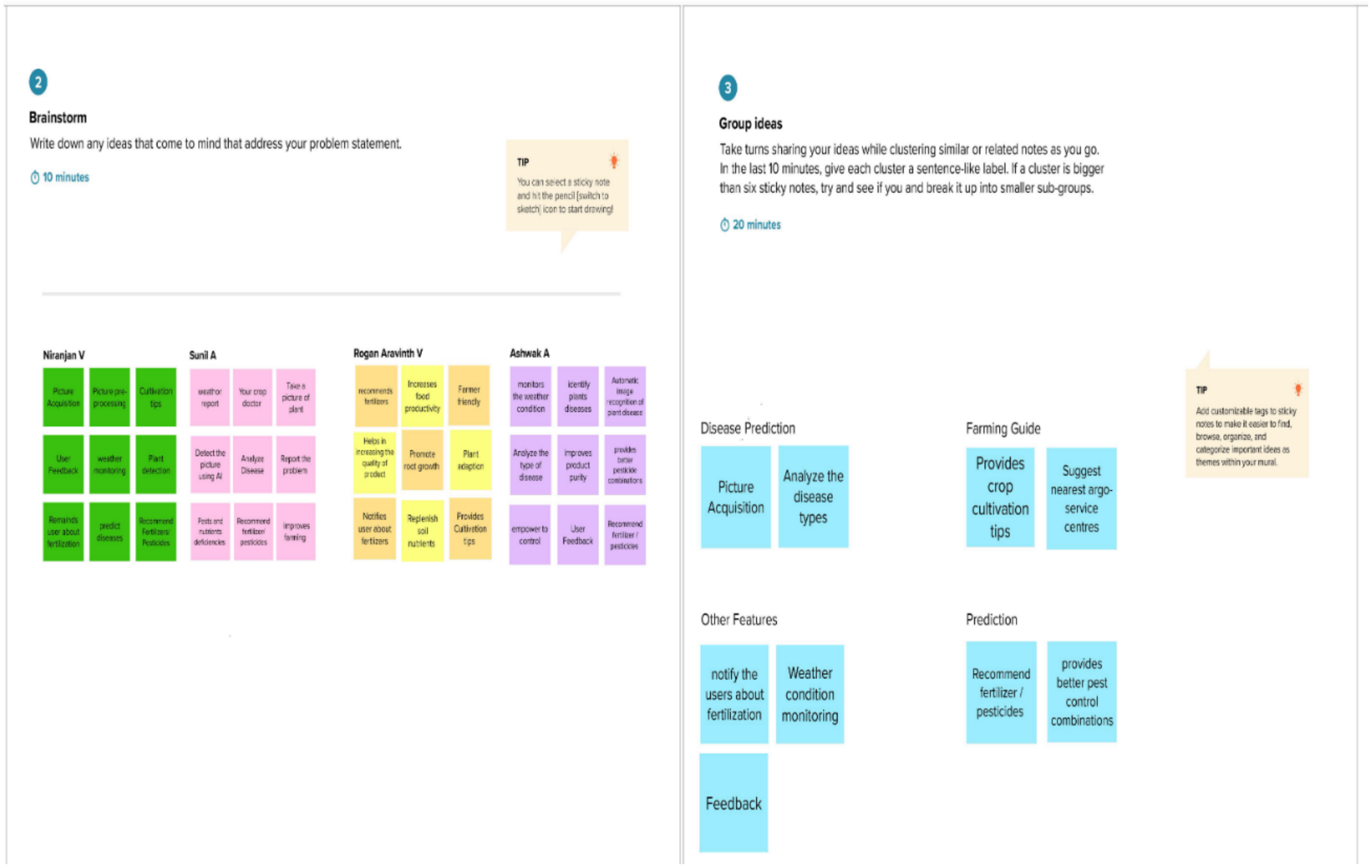
Ideation is often closely related to the practice of brainstorming, a specific technique that is utilized to generate new ideas. A principal difference between ideation and brainstorming is that ideation is commonly more thought of as being an individual pursuit, while brainstorming is almost always a group activity. Brainstorming is usually conducted by getting a group of people together to come up with either general new ideas or ideas for solving a specific problem or dealing with a specific situation.

For example, a major corporation that recently learned it is the object of a major lawsuit may want to gather together top executives for a brainstorming session on how to publicly respond to the lawsuit being filed.

Participants in a brainstorming session are encouraged to freely toss out whatever ideas may occur to them. The thinking is that by generating a large number of ideas, the brainstorming group is likely to come up with a suitable solution for whatever issue they are addressing.

The lines between ideation and brainstorming have become a bit more blurred with the development of several brainstorming software programs, such as Bright idea and Idea wake. These software programs are designed to encourage employees of companies to generate new ideas for improving the companies' operations and, ultimately, bottom-line profitability.

The programs often combine the processes of ideation and brainstorming in that individual employees can use them, but companies may simulate brainstorming sessions by having several employees all utilize the software to generate new ideas intended to address a specific purpose.



2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

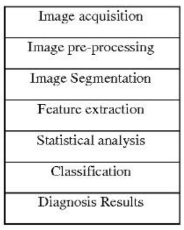
TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

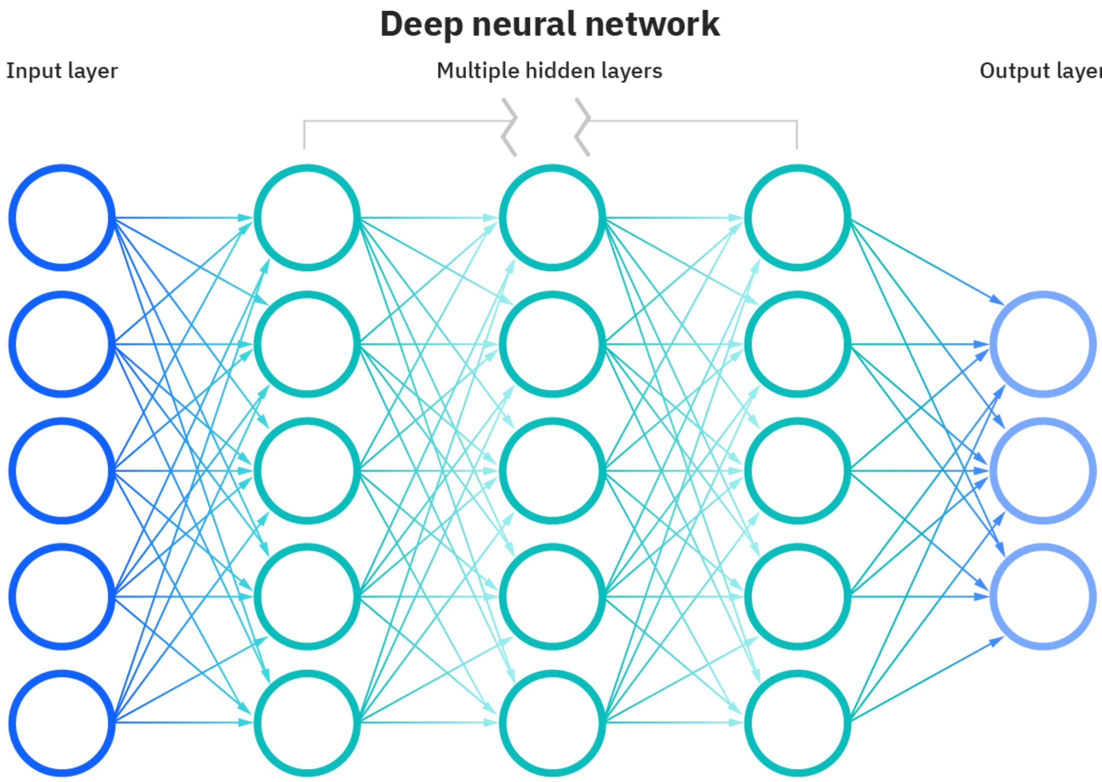
Niranjan V
Picture Acquisition
Picture pre-processing
Cultivation tips
User Feedback
weather monitoring
Plant detection
Remainds user about fertilization
predict diseases
Recommend Fertilizers/ Pesticides
Sunil A
weathor report
Your crop doctor
Take a picture of plant
Detect the picture using AI
Analyze Disease
Report the problem
Pests and nutrients deficiencies
Recommend fertilizer/ pesticides
improves farming
Rogan Aravinth V
recommends fertilizers
Increases food productivity
Farmer friendly
Helps in increasing the quality of product
Promote root growth
Plant adaption
Notifies user about fertizers
Replenish soil nutrients
Provides Cultivation tips
Ashwak A
monitors the weather condition
identify plants diseases
Automatic image recognition of plant disease
Analyze the type of disease
improves product purity
provides better pesticide combinations
empower to control
User Feedback
Recommend fertilizer / pesticides
Disease Prediction
Picture Acquisition
Analyze the disease types
Farming Guide
Provides crop cultivation tips
Suggest nearest argo-service centres
Other Features
notify the users about fertilization
Weather condition monitoring
Feedback
Prediction
Recommend fertilizer / pesticides
provides better pest control combinations

3.3 PROPOSED SOLUTION

In this project a system is introduced. The main purpose of an introduced system is to detect the diseases on the plants by checking the symptoms on the leaves using image processing techniques such as image acquisition, segmentation and feature extraction methods where features such as shape, color and texture are taken into consideration.



Convolutional Neural Network (CNN), a machine learning technique is used in classifying the plant leaves into health or diseased plants. If the classified plant leaf is affected by a disease, CNN will give the name of that particular disease and suggests the precautions for that particular disease is made. The suitable fertilizer is recommended for that particular diseases which will help in growing healthy plants and improve the productivity.



The notification feature is introduced in this system to notify users about the information related to the fertilizer subsidies provided by the government.

3.4 PROBLEM SOLUTION FIT

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. It helps entrepreneurs, marketers and corporate innovators identify behavioral patterns and recognize what would work and why **Purpose:**

- Solve complex problems in a way that fits the state of your customers.
- Succeed faster and increase your solution adoption by tapping into existing mediums and channels of behavior.
- Sharpen your communication and marketing strategy with the right triggers and messaging.
- Increase touch-points with your company by finding the right problem-behavior fit and building trust by solving frequent annoyances, or urgent or costly problems.
- Understand the existing situation in order to improve it for your target group.

1. Customer Segment(s)

The farmers who grow crops in the fields are the Customer as per our Solution. They themselves can scan the leaf of the plant they are growing, if they found it to be infected by any kind of leaf disease. if they are in need of any kind of solution regarding how and what can be done to avoid such disease , they can use Our Solution.

2. Jobs To Be Done / Problems

Farmers are concerned about various pathogen diseases that will occur in different stages of plants of both fruits and vegetables. Once they find that their plants are yet to get infected from any of such diseases, they would be able to use our System as a Solution for their problem.

3. Triggers

Finding Suitable Fertilizer and Safe precautions for the growth of the Plants tend to be a trigger for our system. It is possible by making our System into existence so that that would help the users to get rid of such problems.

4. Emotions: Before / After

If they faced a problem , they could confidently handle the problem and ease access , without being an user farmers will not face any such issues further.

5. Available Solutions

We have introduced various Solutions to get rid of the above mentioned Issues. Some of them are

- Disease
- Detection
- Fertilizer Recommendation
- Subsidy Information

- Cultivation Tips

Disease detection:

Once the user uploads the image of a leaf that he finds to be infected , the trained system would find the type of disease by which it is infected. This is first done by checking if the leaf is infected or not , Once the System finds that the leaf is infected , it detects the Disease by which the plant gets affected.

Fertilizer Recommendation :

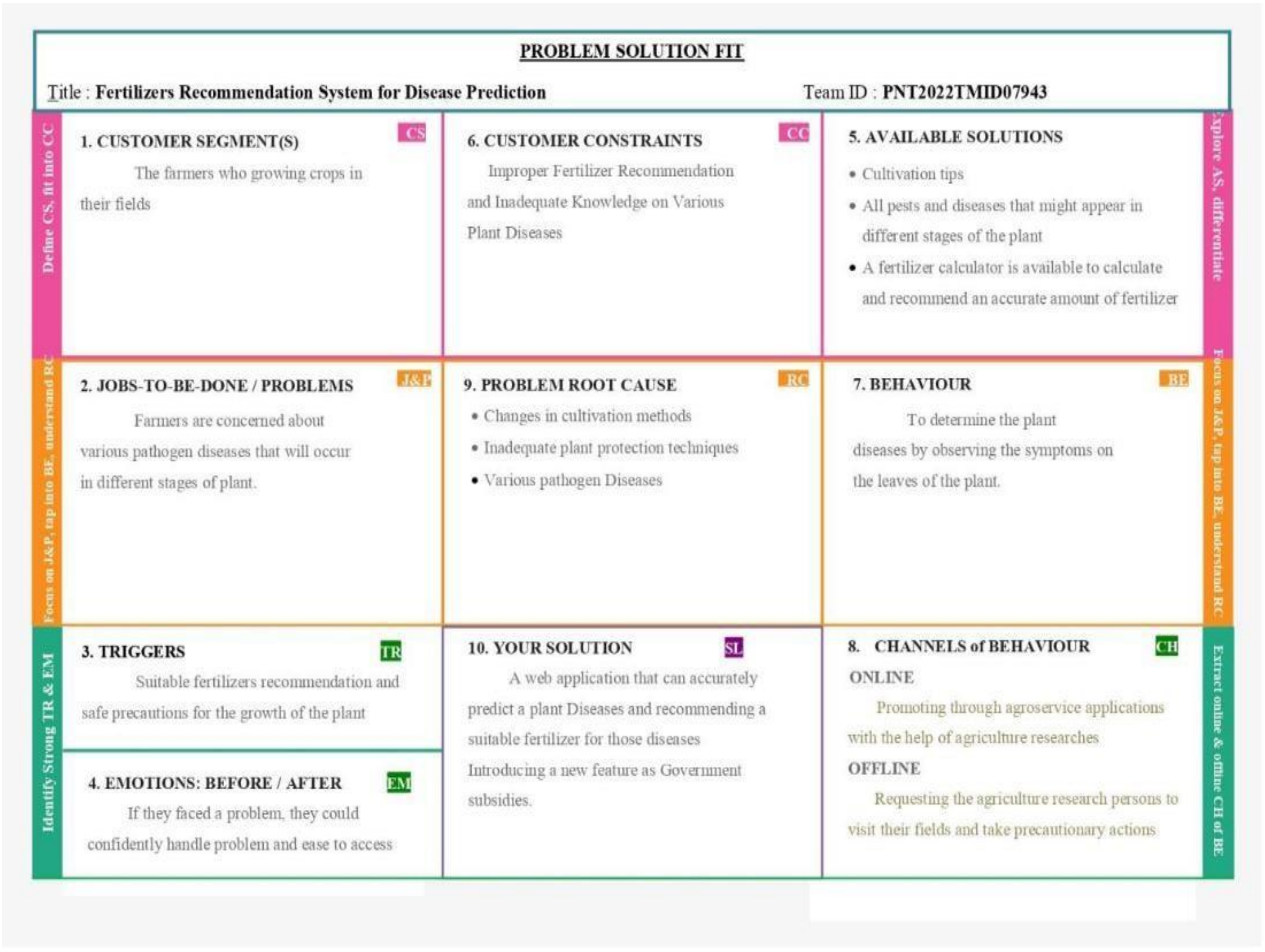
Once the system finds the disease it could check for all the types of Plant diseases that the particular leaf can get affected , based on the disease the leaf can get affected , our system would predict the suitable fertilizer that could prevent further Disease.

Subsidy Information :

Users will be notified about all the recent Informations about the Fertilizer subsidy that is announced by the Government, Since Many of the Farmers are not aware of any such Subsidies , this solution shall be useful in many such kinds.

Cultivation Tips :

Apart from the Subsidy informations , Users will be notified about some of the Cultivation tips that would help the farmer to know better about the type of farming in which they are practicing



PROBLEM SOLUTION FIT

Title : **Fertilizers Recommendation System for Disease Prediction** Team ID : **PNT2022TMID07943**

Define CS, fit into CC	**1. CUSTOMER SEGMENT(S)** CS The farmers who growing crops in their fields	**6. CUSTOMER CONSTRAINTS** CC Improper Fertilizer Recommendation and Inadequate Knowledge on Various Plant Diseases	**5. AVAILABLE SOLUTIONS** • Cultivation tips • All pests and diseases that might appear in different stages of the plant • A fertilizer calculator is available to calculate and recommend an accurate amount of fertilizer	Explore AS, differentiate
Focus on J&P, tap into BE, understand RC	**2. JOBS-TO-BE-DONE / PROBLEMS** J&P Farmers are concerned about various pathogen diseases that will occur in different stages of plant.	**9. PROBLEM ROOT CAUSE** RC • Changes in cultivation methods • Inadequate plant protection techniques • Various pathogen Diseases	**7. BEHAVIOUR** BE To determine the plant diseases by observing the symptoms on the leaves of the plant.	Focus on J&P, tap into BE, understand RC
Identify Strong TR & EM	**3. TRIGGERS** TR Suitable fertilizers recommendation and safe precautions for the growth of the plant **4. EMOTIONS: BEFORE / AFTER** EM If they faced a problem, they could confidently handle problem and ease to access	**10. YOUR SOLUTION** SL A web application that can accurately predict a plant Diseases and recommending a suitable fertilizer for those diseases Introducing a new feature as Government subsidies.	**8. CHANNELS of BEHAVIOUR** CH **ONLINE** Promoting through agroservice applications with the help of agriculture researches **OFFLINE** Requesting the agriculture research persons to visit their fields and take precautionary actions	Extract online & offline CH of BE

CHAPTER 4
SYSTEM REQUIREMENTS

4.1 FUNCTIONAL REQUIREMENTS

Functional requirements define the internal workings of the software: that is, the technical details, data manipulation and processing and other specific functionality that show how the use cases are to be satisfied. They are supported by non-functional requirements, which impose constraints on the design or implementation.

HARDWARE REQUIREMENTS

Component	Minimum Requirement
Processor	64-bit, four-core, 2.5 GHz minimum per core
RAM	8 GB for developer or evaluation use 16 GB for production use
Hard disk	20 GB

SOFTWARE REQUIREMENTS

- Machine Learning Algorithms
- Microsoft Excel 2019
- Python IDE
- Jupyter Tool (Anaconda Python Distribution)
- VS Code Editor

Machine Learning Algorithms

Machine Learning (ML) is a technology which is a subset of artificial intelligence in which the computer machine is trained based on the labeled data set and works according to the trained set of data and the predictions will occur or the algorithm of the machine is trained in the way in which it will learn from the from the examples of the environment in which it is been deployed, makes experience and provides the predictions of that environment, without explicitly programming computers. The two main processes of machine learning algorithms are classification and regression.

Broadly, there are 3 types of Machine Learning Algorithms

i. Supervised Learning

This algorithm consists of a target / outcome variable (or dependent variable) which is to be predicted from a given set of predictors (independent variables). Using these sets of variables, we generate a function that maps inputs to desired outputs. The training process continues until the model achieves a desired level of accuracy on the training data. Examples of Supervised Learning: Regression, Decision Tree, Random Forest, KNN, Logistic Regression etc.

ii. Unsupervised Learning

In this algorithm, we do not have any target or outcome variable to predict / estimate. It is used for clustering populations in different groups, which is widely used for segmenting customers in different groups for specific intervention. Examples of Unsupervised Learning: Apriori algorithm, K-means.

iii. Reinforcement Learning

Using this algorithm, the machine is trained to make specific decisions. It works this way: the machine is exposed to an environment where it trains itself continually using trial and error. This machine learns from past experience and tries to capture the best possible knowledge to make accurate business decisions.

Example of Reinforcement Learning: Markov Decision Process

Common Machine Learning Algorithms

The list of commonly used machine learning algorithms. These algorithms can be applied to almost any data problem:

- Linear Regression
- Logistic Regression
- Decision Tree
- SVM
- Naive Bayes
- kNN
- K-Means
- Random Forest
- Dimensionality Reduction Algorithms
- Gradient Boosting algorithms

Microsoft Excel

Microsoft Excel has the basic features of all spreadsheets, using a grid of *cells* arranged in numbered *rows* and letter-named *columns* to organize data manipulations like arithmetic operations. It has a battery of supplied functions to answer statistical, engineering, and financial needs. In addition, it can display data as line graphs, histograms and charts, and with a very limited three- dimensional graphical display. It allows sectioning of data to view its dependencies on various factors for different perspectives (using *pivot tables* and the *scenario manager*). A PivotTable is a powerful tool that can save time in data analysis. It does this by simplifying large data sets via PivotTable fields It has a programming aspect, *Visual Basic for Applications*, allowing the user to employ a wide variety of numerical methods, for example, for solving differential equations of mathematical physics, and then reporting the results back to the spreadsheet.

It also has a variety of interactive features allowing user interfaces that can completely hide the spreadsheet from the user, so the spreadsheet presents itself as a so-called *application*, or *decision support system* (DSS), via a custom-designed user interface, as a design tool that asks the user questions and provides answers and reports. In a more elaborate realization, an Excel application can automatically poll external databases and measuring instruments using an update schedule, and analyze the results.

Python IDE

An IDE (or Integrated Development Environment) is a program dedicated to software development. As the name implies, IDEs integrate several tools specifically designed for software development. These tools usually include:

- An editor designed to handle code (with, for example, syntax highlighting and auto-completion)
- Build, execution, and debugging tools

Most IDEs support many different programming languages and contain many more features. They can, therefore, be large and take time to download and install. You may also need advanced knowledge to use them properly.

In contrast, a dedicated code editor can be as simple as a text editor with syntax highlighting and code formatting capabilities. Most good code editors can execute code and control a debugger. The very best ones interact with source control systems as well. Compared to an IDE, a good dedicated code editor is usually smaller and quicker, but often less feature rich.

Jupyter Tool

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. It is a server-client application that allows editing and running notebook documents via a web browser. The *Jupyter Notebook App* can be executed on a local desktop requiring no internet access (as described in this document) or can be installed on a remote server and accessed through the internet. The Jupyter Notebook has a notebook *kernel* which is a “computational engine” that executes the code contained in a Notebook document. The *ipython kernel*, referenced in this guide, executes python code. The uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, etc.

4.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements are requirements which specify criteria that can be used to judge the operation of a system, rather than specific behaviors. This should be contrasted with functional requirements that specify specific behavior or functions. Typical non-functional requirements are reliability, scalability, and cost. Non-functional requirements are often called the ilities of a system. Other terms for non-functional requirements are "constraints", "quality attributes" and "quality of service requirements".

Reliability: The diseases can be predicted within a short period of time before the diseases spread to other crops and the farmer can access the portal at any time based on the requirement by accessing through the internet.

Scalability: The system should be developed in such a way that new modules and functionalities can be added, thereby facilitating system evolution.

Cost: The cost of the application is low because of the free availability of software packages.

Performance: The system's performance is based on the quality of the leaf used for disease prediction and the results for those predictions are obtained within seconds for the diseases and certain fertilizers are recommended for those diseases within a period of time.

Usability: The user/farmer can use the application in a structured manner and there is flow of execution of the requirements across the user interface, redirecting to the pages will be easy flow while performing actions on the buttons in the web page.

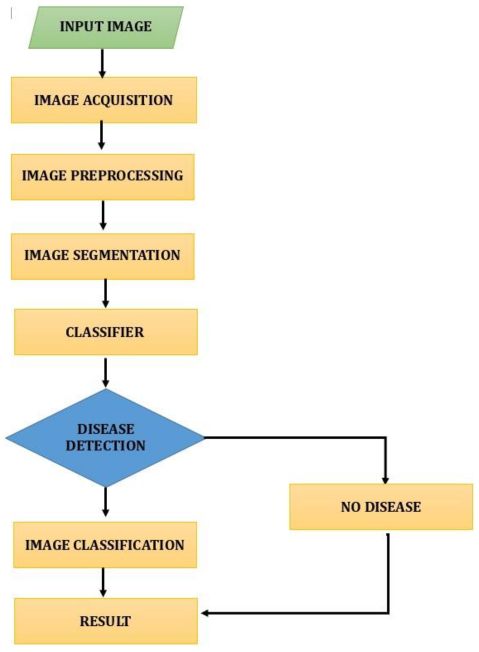
Security: The system provides security for the user information by login to their profile using the login credentials.

CHAPTER 5
PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS

Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation.

Data flow diagrams can be divided into logical and physical. The logical data flow diagram describes flow of data through a system to perform certain functionality of a business. The physical data flow diagram describes the implementation of the logical data flow.



Identification is especially needed to predict both the quality and quantity of the First segmentation step primarily based on a mild polygonal leaf model is first achieved and later used to guide the evolution of an energetic contour. Combining global shape descriptors given by the polygonal model with local curvature based features, the leaves are then classified overleaf datasets. In this research work introduce a method designed to deal with the obstacles raised by such complex images, for simple and plant leaves. A first segmentation step based on graph-cut approach is first performed and later used to guide the evolution of leaf boundaries, and implement classification algorithm to classify the diseases and recommend the fertilizers to affected leaves.

Input Image:

A digital camera or similar devices are used to take images of different types, and then those are used to identify the affected area in leaves. Then different types of image-processing techniques are applied to them, the process those images, to get different and useful features needed for the purpose of analyzing later-Plant leaf disease

Image Classification Steps :

Image acquisition: To get the image of a leaf so that evaluation in the direction of a class can be accomplished.

Image Preprocessing: The purpose of image preprocessing is improving image statistics so that undesired distortions are suppressed and image capabilities which are probably relevant for similar processing are emphasized. The preprocessing receives an image as input and generates an output image as a grayscale, an invert and a smoothed one.

Image Segmentation: Implements Guided active contour method. Unconstrained active contours applied to the difficult natural images. Dealing with unsatisfying contours, which would try and make their way through every possible grab cut in the border of the leaf. The proposed solution uses the polygonal model obtained after the first step not only as an initial leaf contour but also as a shape prior that will guide its evolution towards the real leaf boundary.

Disease Prediction:

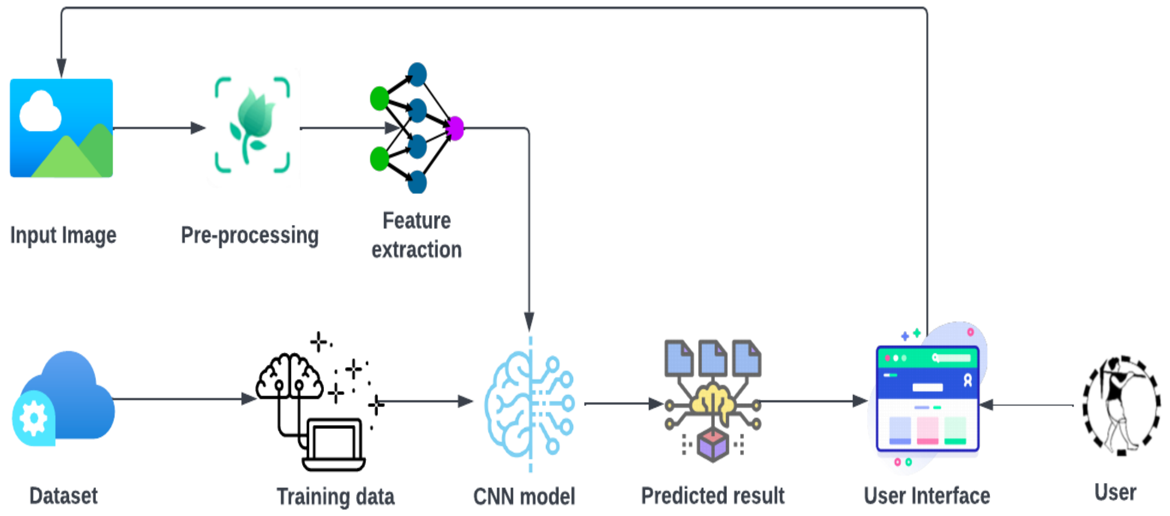
Leaves are affected by bacteria, fungi, virus, and other insects. Support Vector Machine (SVM) algorithm classifies the leaf image as normal or affected. Vectors are constructed based on leaf features such as color, shape, textures. Then hyperplanes are constructed with conditions to categorize the preprocessed leaves and also implement a multiclass classifier, to predict diseases in leaf image with improved accuracy.

Fertilizer Recommendation:

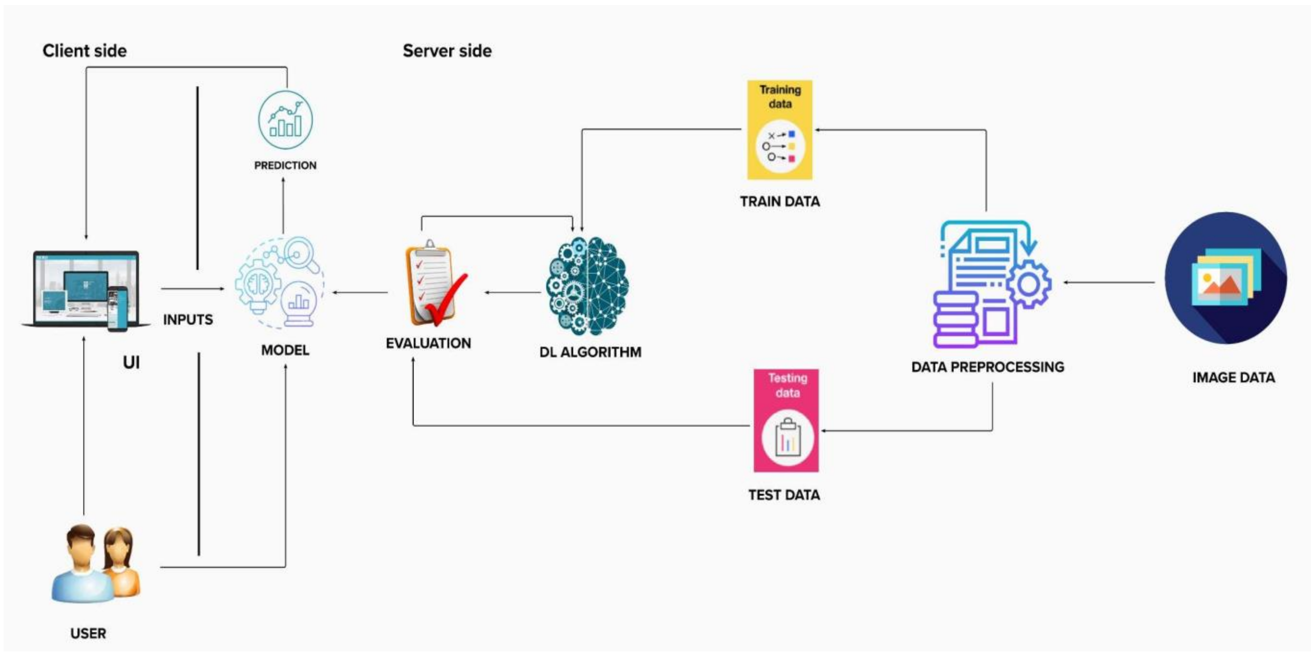
Recommend the fertilizer for affected leaves based on severity level. Fertilizers may be organic or inorganic. Admin can store the fertilizers based on disease categorization with severity levels. The measurements of fertilizers suggested based on disease severity.

5.2 SOLUTION & TECHNICAL ARCHITECTURE

SOLUTION ARCHITECTURE



TECHNICAL ARCHITECTURE



The figure shown above represents the Solution Architecture that we made use of in our Project. the flow starts from the leaf Image dataset where two datasets will be used, we will be creating two models, one to detect vegetable leaf diseases like tomato, potato, and pepper plants and the second model would be for fruits diseases like corn, peach, and apple. We use this data to train the model . Before training the model we have to preprocess the images and then feed them on to the model for training. We make use of **Keras Image Data Generator** class for image preprocessing.

Image Pre-processing includes the following main tasks:

- Import ImageDataGenerator Library.
- Configure ImageDataGenerator Class.
- Applying ImageDataGenerator functionality to the train set and test set.

The ImageDataGenerator accepts the original data, randomly transforms it, and returns only the new, transformed data.

Model Building for Fruit Disease Prediction

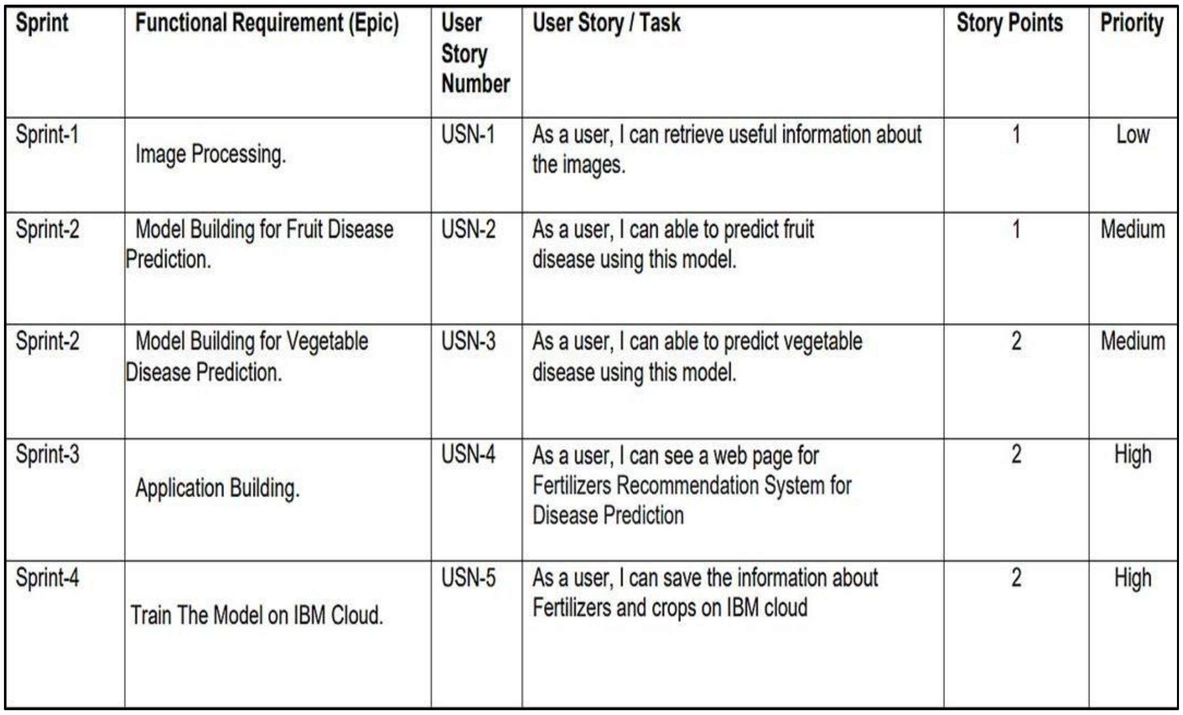
Once we have augmented and pre-processed image data, we shall begin our model building, this activity includes the following steps

- Import the model building Libraries
- Initializing the model
- Adding CNN Layers

- Adding Hidden Layer
- Adding Output Layer
- Configure the Learning Process
- Training and testing the model
- Saving the model

Once the model is successfully trained and tested, it is now ready to predict the type of Disease by which the scanned leaf is infected, this can be viewed by the user(farmers) using the User interface(Web Application).

5.3 USER STORIES



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

CHAPTER 6
PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

The delivery plan of project deliverables is a strategic element for every Project Manager. The goal of every project is, in fact, to produce a result that serves a specific purpose. With the word “purpose “, we can mean the most disparate goals: a software program, a chair, a building, a translation, etc.… In Project Spirit Delivery Planning is one of the processes of completing the project and Showcasing the TimeLine of the Project Planning. This Delivery plan helps to understand the process and Work Flow of the Project working by the Team Mates. Every Single Module is assigned to the teammates to showcase their work and contribution of developing the Project and improve the efficiency of the progress.

Modern technology is increasing and optimizing the performance of the Artificial Intelligence (AI) Model. The Crop Yield Disease Prediction System is helpful for farmers to prevent the crop from various diseases which can identify the Disease within a process of capturing the image at the plant and Machine Learning Algorithm will give affected Disease names. In this Project Milestone will be given the Best Solution for the farmer using the complete friendly and Simple user friendly interface web application for fetching the solution by own. In addition, we are planning to add a valid Module that is Fertilizer Recommendation for the Specific Disease, It can give both Artificial Fertilizer and Natural Fertilizer in Suggestion manner.

6.2 SPRINT DELIVERY SCHEDULE

In project Management, Planning is an important task to schedule the phrase of the project to the Team member. In this Activity represents the various activities allocated and are done by the Team Members. This can be subdivided into four main steps:

Phrase 1:

Information Collection and Requirement Analysis. This is a section where all the required details of the project are analyzed and the requirements are gathered. Once all the requirements are gathered , we make a cross check of the gathered information , if we find that any of the required information is not yet gathered , those components shall be gathered and are used for further use.

Phrase 2:

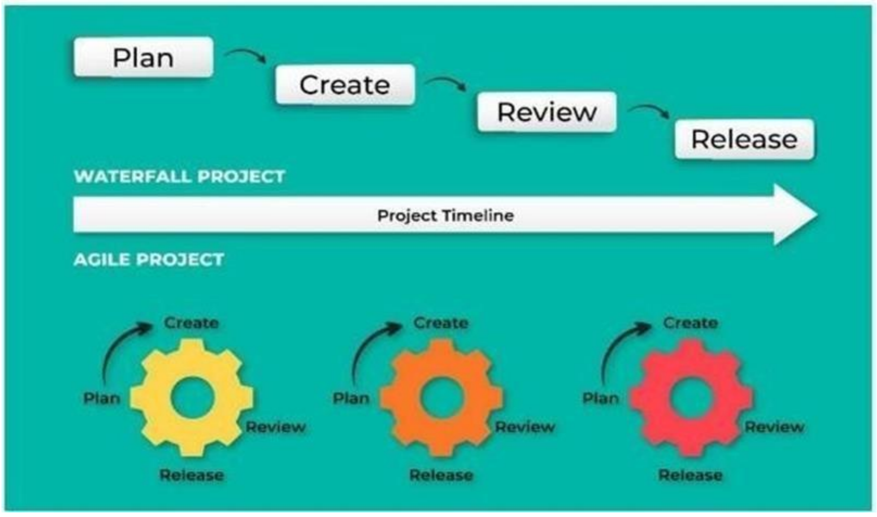
Project Planning and Developing Modules, This is a section where the whole project is subdivided into various modules. This is done so that each work can be split to each member of the team. Once all the modules are completed, they will be integrated to make it a whole project.

Phrase 3:

Implementing the High Accuracy Deep Learning Algorithm, This is done so that the, model gets trained well with the algorithm, to find the accurate result of the expected plant Disease

Phrase 4:

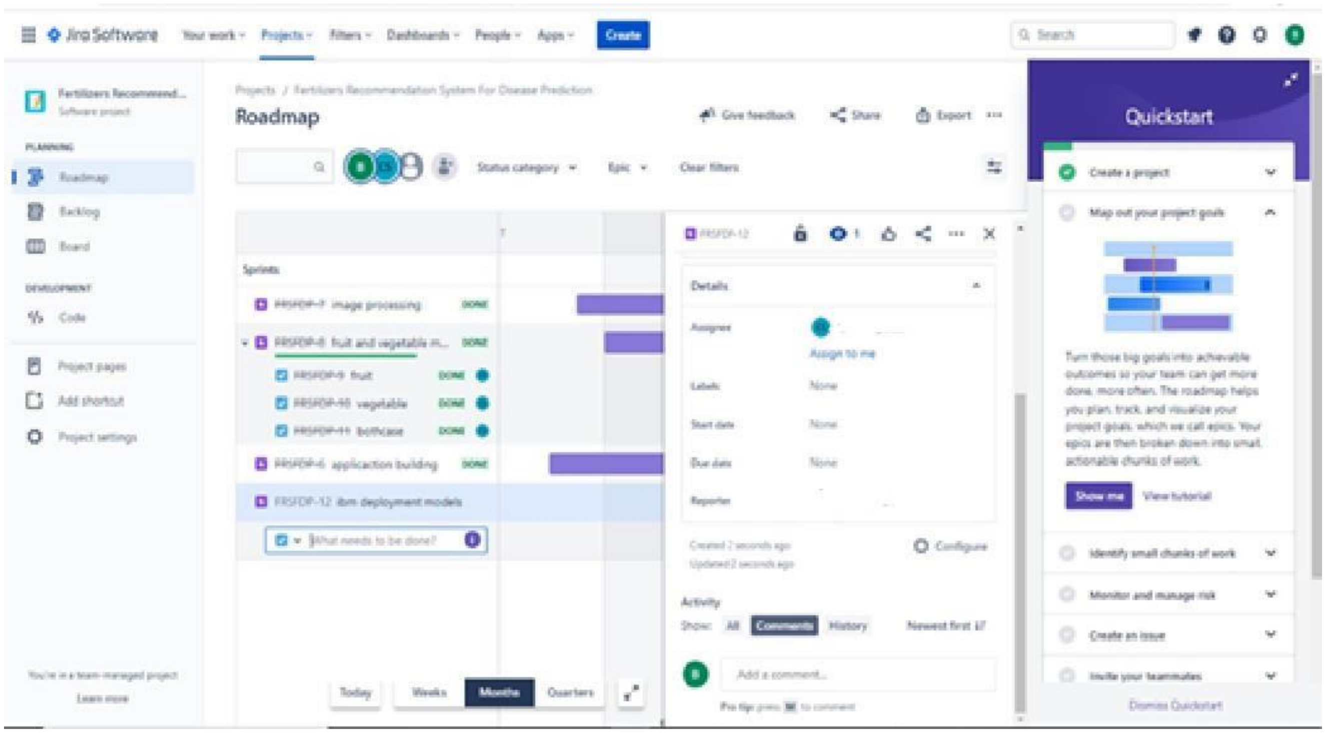
Deploying the Model on Cloud and Testing the Model and UI Performance, This is done to make our solution available anywhere and anytime this makes our solution more efficient and reliable.



6.3 REPORTS FROM JIRA:

Jira Software is part of a family of products designed to help teams of all types manage work. Originally, Jira was designed as a bug and issue tracker. But today, Jira has evolved into a powerful work management tool for all kinds of use cases, from requirements and test case management to agile software development.

Jira is one of the best open-source tools for planning and tracking in Agile methodology. Development teams use Jira for tracking bugs and projects, managing Scrums, and visualizing workflows with Kanban boards. Workflows in Jira make it easy to plan, track, release, and report on software.



Fertilizers Recommend...
Software project

PLANNING

DEVELOPMENT

You're in a team-managed project
Learn more

Projects / Fertilizers Recommendation System For Disease Prediction

Roadmap

Sprints

FRSFDP-7 image processing	DONE
FRSFDP-8 fruit and vegetable m...	DONE
FRSFDP-9 fruit	DONE
FRSFDP-10 vegetable	DONE
FRSFDP-11 bothcase	DONE
FRSFDP-6 application building	DONE
FRSFDP-12 ibm deployment models	

What needs to be done?

FRSFDP-12

Details

Assignee	Assign to me
Labels	None
Start date	None
Due date	None
Reporter	

Created 2 seconds ago
Updated 2 seconds ago

Configure

Activity
Show: All Comments History Newest first

Add a comment...

Pro tip: press M to comment

Quickstart

Create a project

Map out your project goals

Turn those big goals into achievable outcomes so your team can get more done, more often. The roadmap helps you plan, track, and visualize your project goals, which we call epics. Your epics are then broken down into small, actionable chunks of work.

Identify small chunks of work

Monitor and manage risk

Create an issue

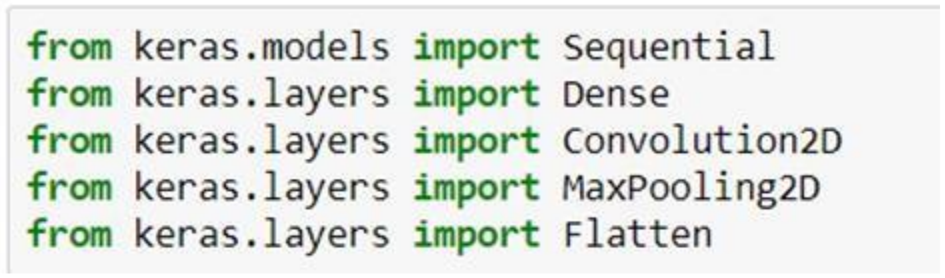
Invite your teammates

CHAPTER 7
CODING & SOLUTIONING

7.1 FEATURE 1[MODEL BUILDING]:

Import the Libraries

Import the libraries that are required to initialize the neural network layer, and create and add different layers to the neural network model.



```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
```

Initializing The Model

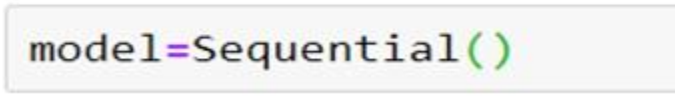
Keras has 2 ways to define a neural network:

· Sequential
· Function API

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

Now, will initialize our model.

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



```
model=Sequential()
```

Add CNN Layers

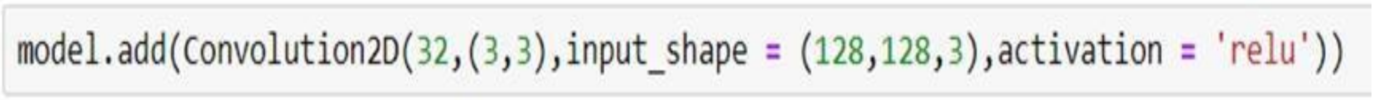
We will be adding three layers for CNN

- Convolution layer
- Pooling layer
- Flattening layer

Add Convolution Layer

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

Activation Function: These are the functions that help us to decide if we need to activate the node or not. These functions introduce non-linearity in the networks.

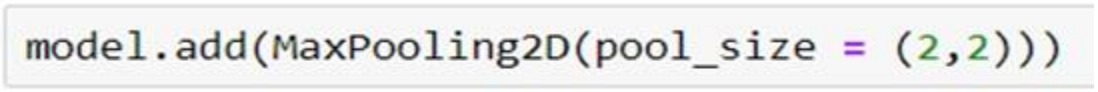


```
model.add(Convolution2D(32,(3,3),input_shape = (128,128,3),activation = 'relu'))
```

Add the pooling layer

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

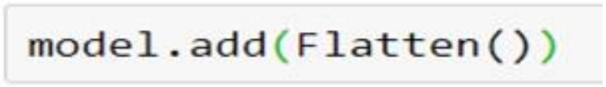
After the convolution layer, a pooling layer is added. Max pooling layer can be added using MaxPooling2D class. It takes the pool size as a parameter. Efficient size of the pooling matrix is (2,2). It returns the pooled feature maps. (Note: Any number of convolution layers, pooling and dropout layers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



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

Add the flatten layer

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



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

Adding Hidden layers

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

Adding the output layer

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

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

Train And Save The Model Compile the model

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

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

Fit and save the model

Fit the neural network model with the train and test set, number of epochs and validation steps.

Steps per epoch is determined by number of training

The weights are to be saved for future use. The weights are saved in as .h5 file using save().

```
model.fit_generator(x_train, steps_per_epoch = 168,epochs = 3,validation_data = x_test,validation_
```

```
model.save("fruit.h5")
```

model.summary() can be used to see all parameters and shapes in each layer in our models. **Test The Model**

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

Import the packages and load the saved model

Import the required libraries

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

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

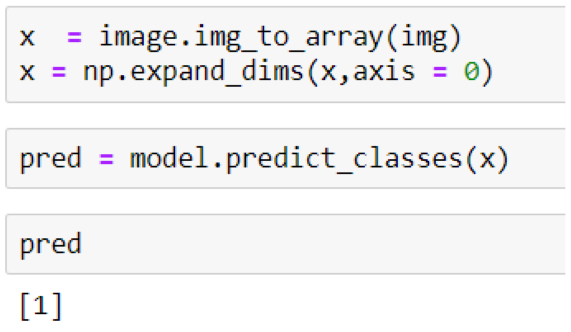
```
model = load_model("fruit.h5")
```

Load the test image, pre-process it and predict

Pre-processing the image includes converting the image to array and resizing according to the model.

Give the pre-processed image to the model to know to which class your model belongs to.

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



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

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

```
pred
```

```
[1]
```

The predicted class is 1.

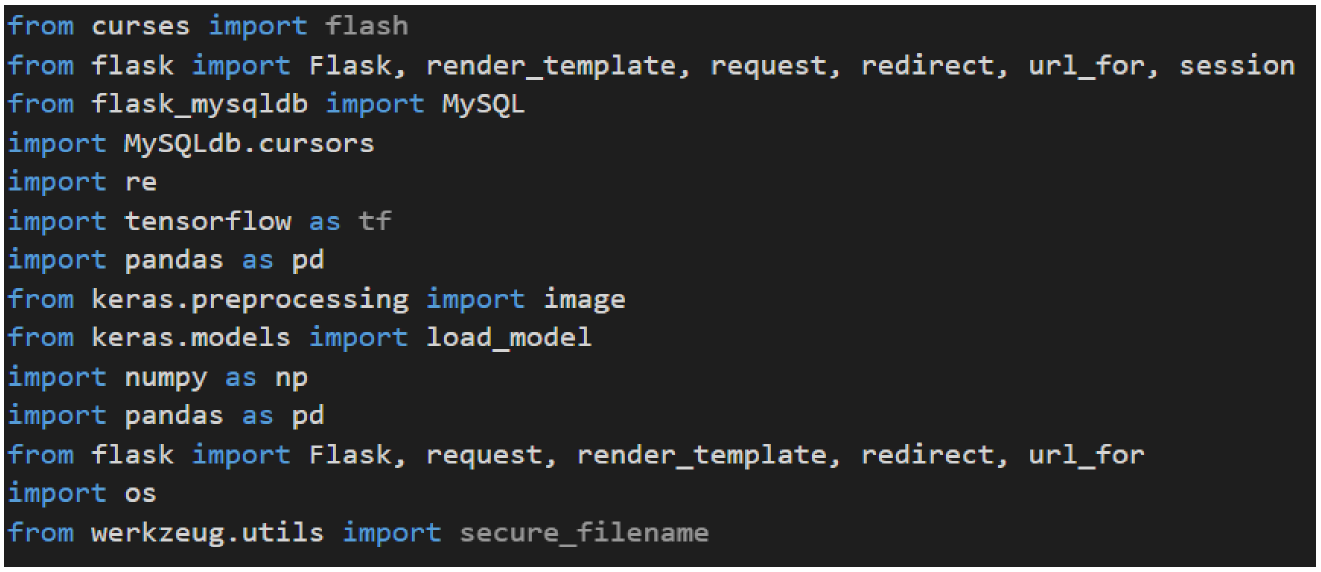
7.2 FEATURE 2[PYTHON CODE]:

Build Python Code:

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

Activity 1: Build a flask application

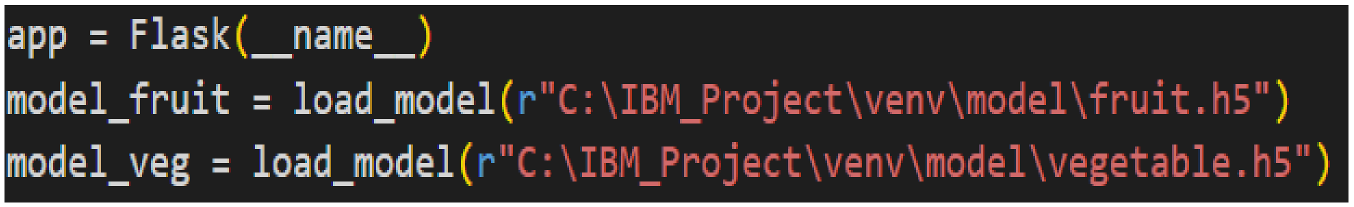
Step 1: Load the required packages



```
from curses import flash
from flask import Flask, render_template, request, redirect, url_for, session
from flask_mysqldb import MySQL
import MySQLdb.cursors
import re
import tensorflow as tf
import pandas as pd
from keras.preprocessing import image
from keras.models import load_model
import numpy as np
import pandas as pd
from flask import Flask, request, render_template, redirect, url_for
import os
from werkzeug.utils import secure_filename
```

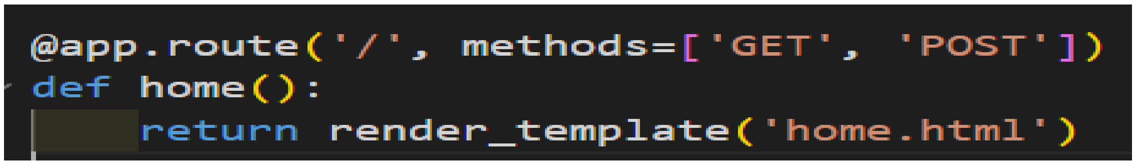
Step 2: Initialize the flask app and load the model

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



```
app = Flask(__name__)
model_fruit = load_model(r"C:\IBM_Project\venv\model\fruit.h5")
model_veg = load_model(r"C:\IBM_Project\venv\model\vegetable.h5")
```

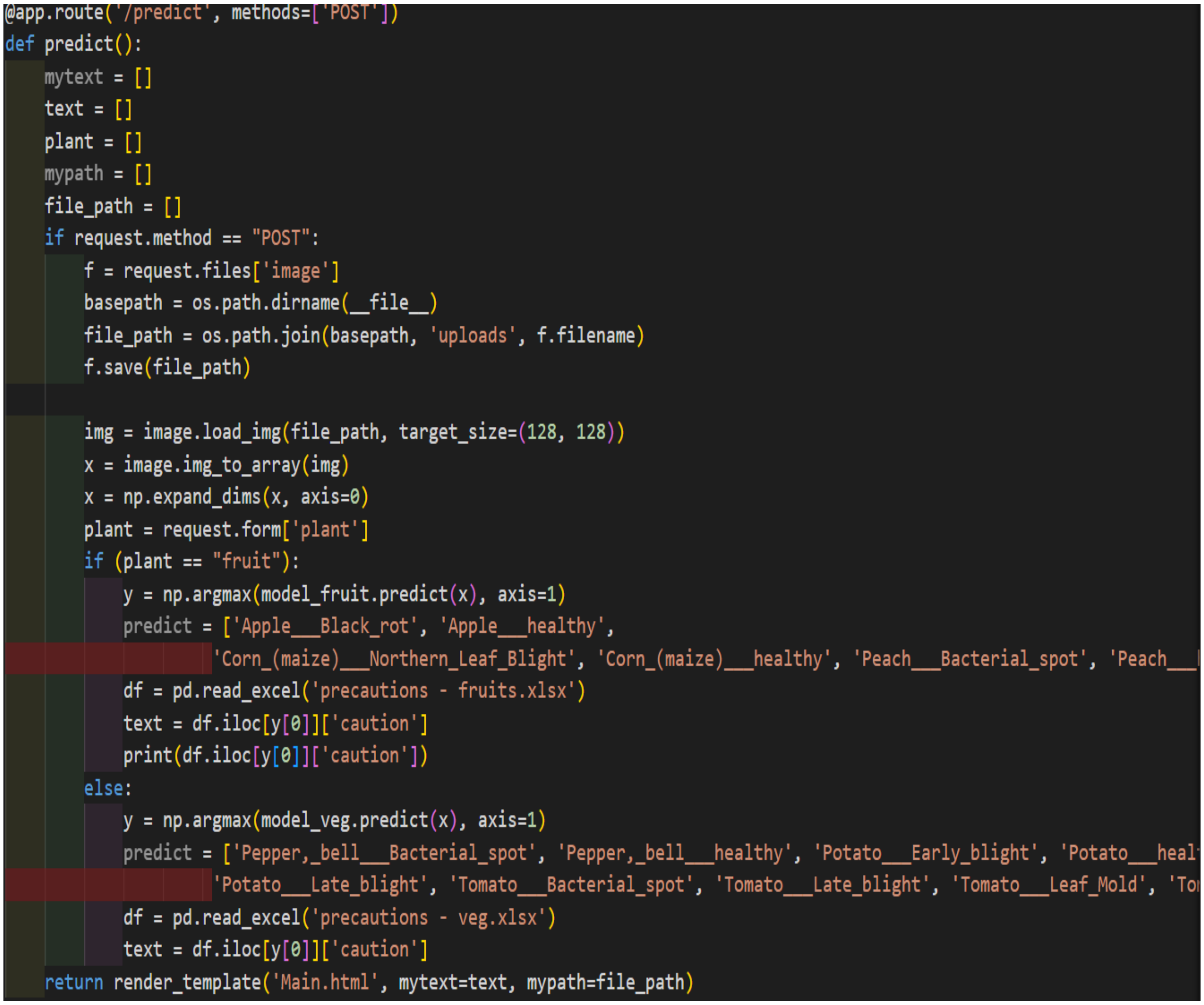
Step 3: Configure the home page



```
@app.route('/', methods=['GET', 'POST'])
def home():
    return render_template('home.html')
```

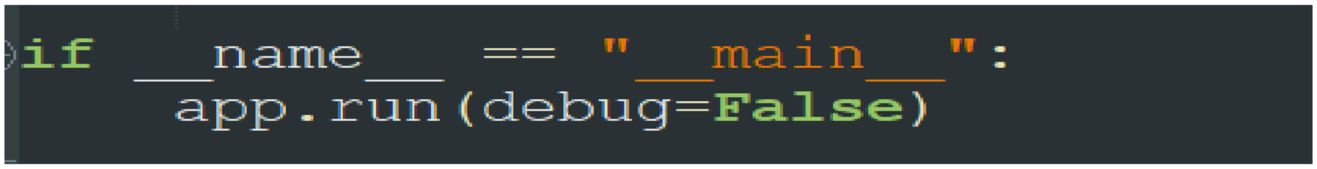
Step 4: Pre-process the frame and run

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



```
@app.route('/predict', methods=['POST'])
def predict():
    mytext = []
    text = []
    plant = []
    mypath = []
    file_path = []
    if request.method == "POST":
        f = request.files['image']
        basepath = os.path.dirname(__file__)
        file_path = os.path.join(basepath, 'uploads', f.filename)
        f.save(file_path)

        img = image.load_img(file_path, target_size=(128, 128))
        x = image.img_to_array(img)
        x = np.expand_dims(x, axis=0)
        plant = request.form['plant']
        if (plant == "fruit"):
            y = np.argmax(model_fruit.predict(x), axis=1)
            predict = ['Apple___Black_rot', 'Apple___healthy',
                     'Corn_(maize)___Northern_Leaf_Blight', 'Corn_(maize)___healthy', 'Peach___Bacterial_spot', 'Peach___
            df = pd.read_excel('precautions - fruits.xlsx')
            text = df.iloc[y[0]]['caution']
            print(df.iloc[y[0]]['caution'])
        else:
            y = np.argmax(model_veg.predict(x), axis=1)
            predict = ['Pepper,_bell___Bacterial_spot', 'Pepper,_bell___healthy', 'Potato___Early_blight', 'Potato___heal
                     'Potato___Late_blight', 'Tomato___Bacterial_spot', 'Tomato___Late_blight', 'Tomato___Leaf_Mold', 'To
            df = pd.read_excel('precautions - veg.xlsx')
            text = df.iloc[y[0]]['caution']
    return render_template('Main.html', mytext=text, mypath=file_path)
```

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



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

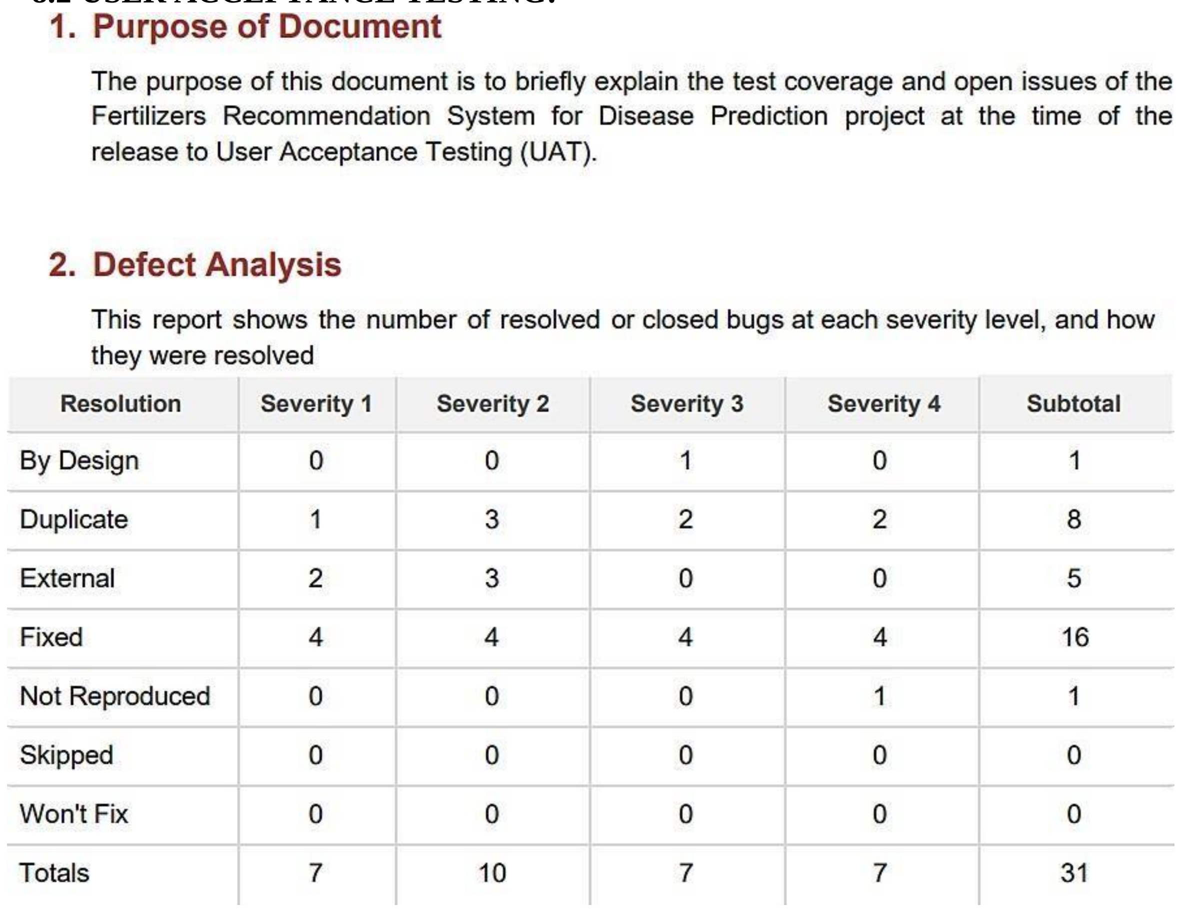
CHAPTER 8

TESTING

8.1 TEST CASES:

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

8.2 USER ACCEPTANCE TESTING:



1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Fertilizers Recommendation System for Disease Prediction project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	0	0	1	0	1
Duplicate	1	3	2	2	8
External	2	3	0	0	5
Fixed	4	4	4	4	16
Not Reproduced	0	0	0	1	1
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	7	10	7	7	31

3. Test Case Analysis

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

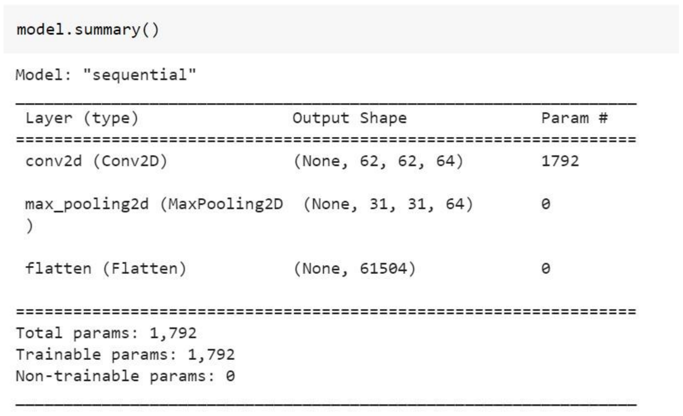
Section	Total Cases	Not Tested	Fail	Pass
Print Engine	1	0	0	1
Client Application	1	0	0	1
Security	1	0	0	1
Outsource Shipping	1	0	0	1
Exception Reporting	1	0	0	1
Final Report Output	1	0	0	1
Version Control	1	0	0	1

CHAPTER 9
RESULTS

9.1 PERFORMANCE METRICS:

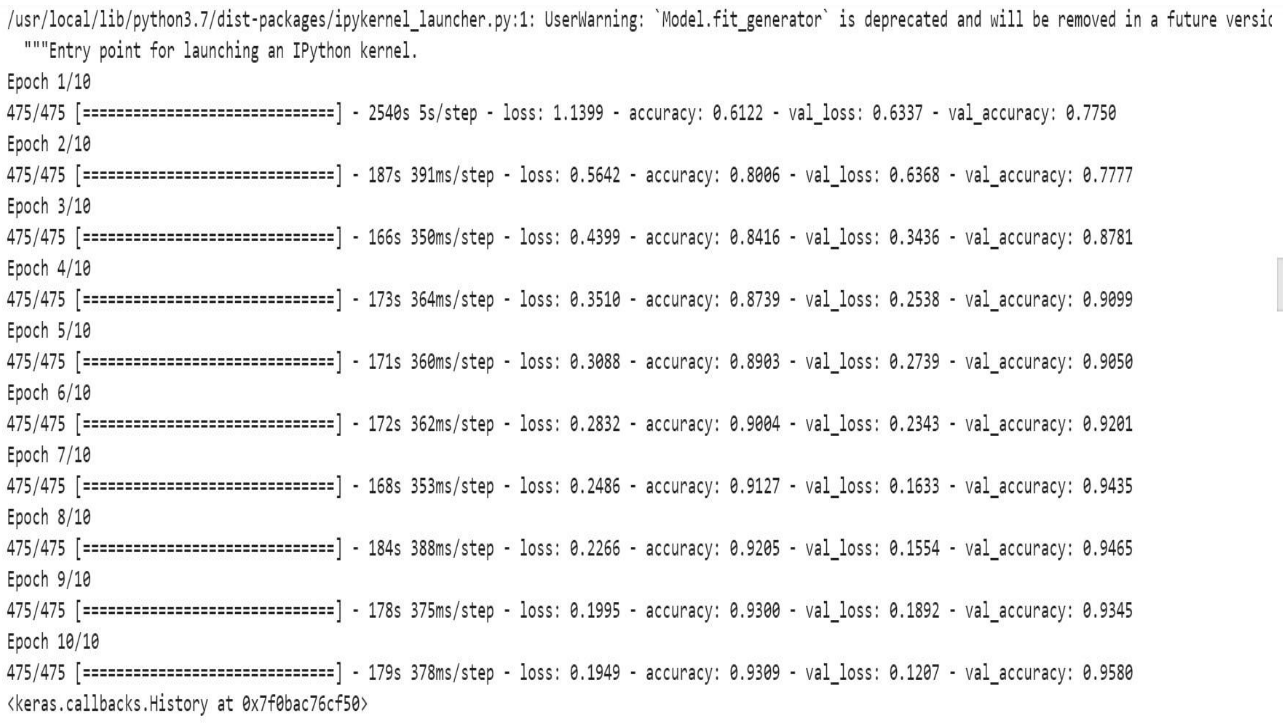
Vegetable.h5

Model Summary:



```
model.summary()

Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 64)        1792

 max_pooling2d (MaxPooling2D  (None, 31, 31, 64)       0
 )

 flatten (Flatten)           (None, 61504)             0

=================================================================
Total params: 1,792
Trainable params: 1,792
Non-trainable params: 0
_________________________________________________________________
```

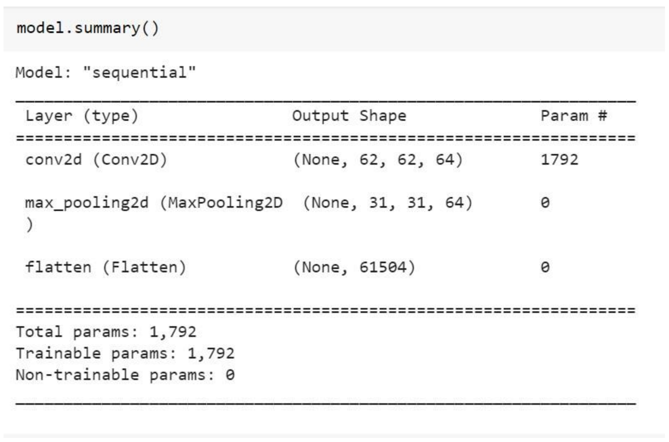
Accuracy:



```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future versic
  """Entry point for launching an IPython kernel.
Epoch 1/10
475/475 [==============================] - 2540s 5s/step - loss: 1.1399 - accuracy: 0.6122 - val_loss: 0.6337 - val_accuracy: 0.7750
Epoch 2/10
475/475 [==============================] - 187s 391ms/step - loss: 0.5642 - accuracy: 0.8006 - val_loss: 0.6368 - val_accuracy: 0.7777
Epoch 3/10
475/475 [==============================] - 166s 350ms/step - loss: 0.4399 - accuracy: 0.8416 - val_loss: 0.3436 - val_accuracy: 0.8781
Epoch 4/10
475/475 [==============================] - 173s 364ms/step - loss: 0.3510 - accuracy: 0.8739 - val_loss: 0.2538 - val_accuracy: 0.9099
Epoch 5/10
475/475 [==============================] - 171s 360ms/step - loss: 0.3088 - accuracy: 0.8903 - val_loss: 0.2739 - val_accuracy: 0.9050
Epoch 6/10
475/475 [==============================] - 172s 362ms/step - loss: 0.2832 - accuracy: 0.9004 - val_loss: 0.2343 - val_accuracy: 0.9201
Epoch 7/10
475/475 [==============================] - 168s 353ms/step - loss: 0.2486 - accuracy: 0.9127 - val_loss: 0.1633 - val_accuracy: 0.9435
Epoch 8/10
475/475 [==============================] - 184s 388ms/step - loss: 0.2266 - accuracy: 0.9205 - val_loss: 0.1554 - val_accuracy: 0.9465
Epoch 9/10
475/475 [==============================] - 178s 375ms/step - loss: 0.1995 - accuracy: 0.9300 - val_loss: 0.1892 - val_accuracy: 0.9345
Epoch 10/10
475/475 [==============================] - 179s 378ms/step - loss: 0.1949 - accuracy: 0.9309 - val_loss: 0.1207 - val_accuracy: 0.9580
<keras.callbacks.History at 0x7f0bac76cf50>
```

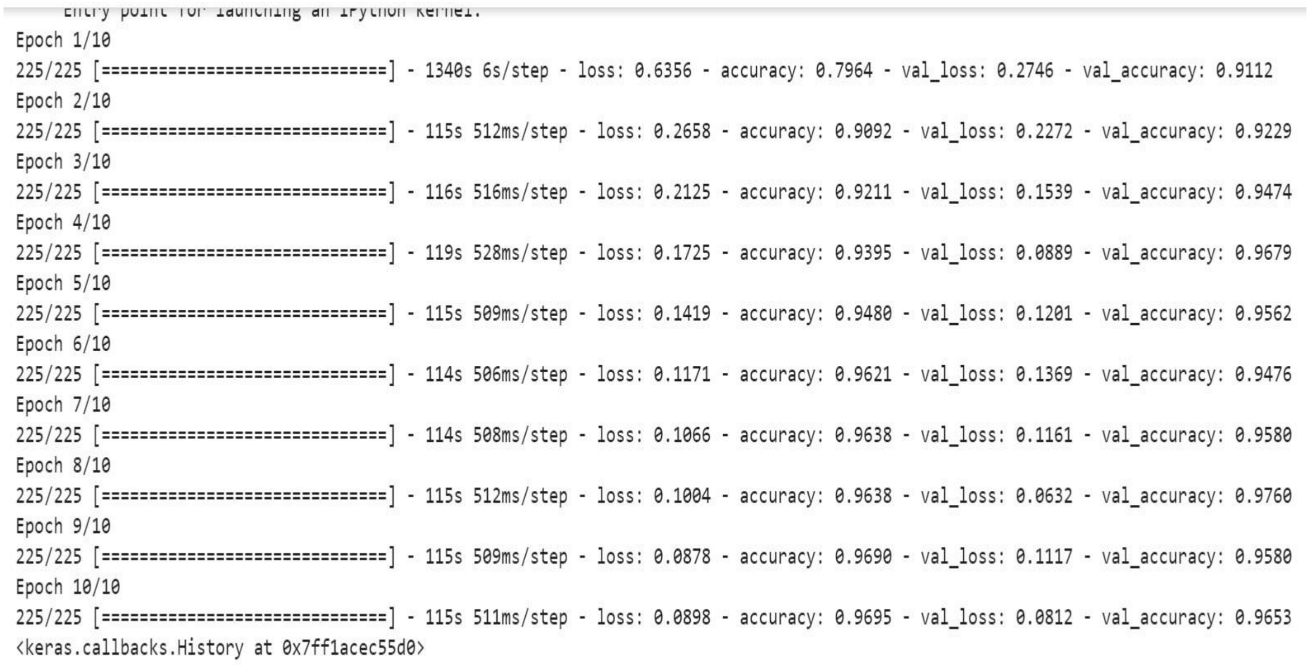
Fruit.h5

Model Summary:



```
model.summary()

Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 64)        1792

 max_pooling2d (MaxPooling2D  (None, 31, 31, 64)       0
 )

 flatten (Flatten)           (None, 61504)             0

=================================================================
Total params: 1,792
Trainable params: 1,792
Non-trainable params: 0
_________________________________________________________________
```

Accuracy:



```
    Entry point for launching an IPython kernel.
Epoch 1/10
225/225 [==============================] - 1340s 6s/step - loss: 0.6356 - accuracy: 0.7964 - val_loss: 0.2746 - val_accuracy: 0.9112
Epoch 2/10
225/225 [==============================] - 115s 512ms/step - loss: 0.2658 - accuracy: 0.9092 - val_loss: 0.2272 - val_accuracy: 0.9229
Epoch 3/10
225/225 [==============================] - 116s 516ms/step - loss: 0.2125 - accuracy: 0.9211 - val_loss: 0.1539 - val_accuracy: 0.9474
Epoch 4/10
225/225 [==============================] - 119s 528ms/step - loss: 0.1725 - accuracy: 0.9395 - val_loss: 0.0889 - val_accuracy: 0.9679
Epoch 5/10
225/225 [==============================] - 115s 509ms/step - loss: 0.1419 - accuracy: 0.9480 - val_loss: 0.1201 - val_accuracy: 0.9562
Epoch 6/10
225/225 [==============================] - 114s 506ms/step - loss: 0.1171 - accuracy: 0.9621 - val_loss: 0.1369 - val_accuracy: 0.9476
Epoch 7/10
225/225 [==============================] - 114s 508ms/step - loss: 0.1066 - accuracy: 0.9638 - val_loss: 0.1161 - val_accuracy: 0.9580
Epoch 8/10
225/225 [==============================] - 115s 512ms/step - loss: 0.1004 - accuracy: 0.9638 - val_loss: 0.0632 - val_accuracy: 0.9760
Epoch 9/10
225/225 [==============================] - 115s 509ms/step - loss: 0.0878 - accuracy: 0.9690 - val_loss: 0.1117 - val_accuracy: 0.9580
Epoch 10/10
225/225 [==============================] - 115s 511ms/step - loss: 0.0898 - accuracy: 0.9695 - val_loss: 0.0812 - val_accuracy: 0.9653
<keras.callbacks.History at 0x7ff1acec55d0>
```

CHAPTER 10

ADVANTAGE & DISADVANTAGE

List of advantages

- The proposed model here produces very high accuracy of classification of the disease in various plants by the symptoms shown on the leaves and the diseases can be identified earlier and those can be controlled in an ordered manner.

- Fertilizers are recommended for those diseases shown in plants and the diseases in the plants are controlled before spreading to other crops.

- A large amount of plant leaves are trained and tested as datasets and farmers can use this web application for the various types of plants which are grown in the fields without depending on other systems.

- Government subsidies which are provided for the fertilizers have been notified to the farmers. It leads to better cooperation between the farmers and the government for the growth of agriculture.
- As the farmers use this web application, they will learn how to treat a plant for a particular disease and what the precautions taken for those diseases can be known to farmers as the application works as user friendly.

List of disadvantages

- The algorithm which has been deployed for the prediction of plant diseases works inappropriately, leads to wrong predictions of the diseases and causes failure of the system.

- The system can predict the diseases only for the trained set of plant dataset and it will not predict the disease for the non-trained set of plant dataset.

- Farmers will not use this web application for the wrong predictions of plant diseases and leads to decrease in disease control cycle of plants.

CHAPTER 11

CONCLUSION

In agriculture disease prediction for the plants plays a major role in all kinds of agricultural crops. Most of the plants are affected with a variety of pathogenic diseases. Due to the diseases in plants the quality of the agricultural products in the market has been minimized. Diseases on plants will lead to loss of crops and results in starvation. Hence, detection of plant diseases are necessary to obtain large quantities and best quality. As the growth of the technology increases in recent years, the number of diseases on plants and intensity of harm caused has increased due to the variation in bacterial and fungal varieties, poor cultivation methods and inadequate plant protection techniques.

A web application is introduced through which the farmer can identify the plant diseases by taking the images of the signs which are shown on the leaves of the plants. Deep learning techniques are used for the identification of the plant diseases and suggest several precautions that can be taken for those diseases and provide fertilizers for the growth of the plant and the control of the plant diseases.

CHAPTER 12

FUTURE SCOPE

The proposed model in this project work can be extended to image recognition. The entire model can be converted to application software using python to exe software. The real time image classification, image recognition and video processing are possible with help of OpenCV python library. This project work can be extended for security applications such as figure print recognition, iris recognition and face recognition.

APPENDIX A

CODING SNIPPETS

app.py

```
from curses import flash
from flask import Flask, render_template, request, redirect, url_for, session
from flask_mysqldb import MySQL
import MySQLdb.cursors
import re
import tensorflow as tf
import pandas as pd
from keras.preprocessing import image
from keras.models import load_model
import numpy as np
import pandas as pd
from flask import Flask, request, render_template, redirect, url_for
import os
from werkzeug.utils import secure_filename

UPLOAD_FOLDER = '/path/to/the/uploads'
ALLOWED_EXTENSIONS = {'txt', 'pdf', 'png', 'jpg', 'jpeg', 'gif'}

app = Flask(__name__)
model_fruit = load_model(r"C:\IBM_Project\venv\model\fruit.h5")
model_veg = load_model(r"C:\IBM_Project\venv\model\vegetable.h5")

# Change this to your secret key (can be anything, it's for extra protection)
app.secret_key = 'asdfghjkl'

# Enter your database connection details below
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'Admin@1026'
app.config['MYSQL_DB'] = 'pythonlogin'

# Intialize MySQL
mysql = MySQL(app)


@app.route('/', methods=['GET', 'POST'])
def home():
    return render_template('home.html')



@app.route('/login', methods=['GET', 'POST'])
def login():
    # Output message if something goes wrong...
    msg = 'Please provide correct credentials'
    # Check if "username" and "password" POST requests exist (user submitted form)
    if request.method == 'POST' and 'username' in request.form and 'password' in request.form:
        # Create variables for easy access
        username = request.form['username']
        password = request.form['password']
        # Check if account exists using MySQL
        cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute(
            'SELECT * FROM accounts WHERE username = %s AND password = %s', (username, password,))
        # Fetch one record and return result
        account = cursor.fetchone()
        # If account exists in accounts table in out database
        if account:
            # Create session data, we can access this data in other routes
            session['loggedin'] = True
            session['id'] = account['id']
            session['username'] = account['username']
            # Redirect to home page
            return render_template('Main.html', msg=msg)
        else:
            # Account doesnt exist or username/password incorrect
            msg = 'Incorrect username/password!'
    # Show the login form with message (if any)
    return render_template('index.html', msg=msg)
# this will be the logout page


@app.route('/logout')
def logout():
    # Remove session data, this will log the user out
    session.pop('loggedin', None)
    session.pop('id', None)
    session.pop('username', None)
    # Redirect to login page
    return render_template('index.html', msg='')

# this will be the registration page, we need to use both GET and POST requests


@app.route('/register', methods=['GET', 'POST'])
def register():
    # Output message if something goes wrong...
    msg = ''
    # Check if "username", "password" and "email" POST requests exist (user submitted form)
    if request.method == 'POST' and 'username' in request.form and 'password' in request.form and 'email' in
request.form:
        # Create variables for easy access
        username = request.form['username']
        password = request.form['password']
        email = request.form['email']
        # Check if account exists using MySQL
        cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute(
            'SELECT * FROM accounts WHERE username = %s', (username,))
        account = cursor.fetchone()
        # If account exists show error and validation checks
        if account:
            msg = 'Account already exists!'
        elif not re.match(r'[^@]+@[^@]+\.[^@]+', email):
            msg = 'Invalid email address!'
        elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'Username must contain only characters and numbers!'
        elif not username or not password or not email:
            msg = 'Please fill out the form!'
        else:
            # Account doesnt exists and the form data is valid, now insert new account into accounts table
            cursor.execute(
                'INSERT INTO accounts VALUES (NULL, %s, %s, %s)', (username, password, email,))
            mysql.connection.commit()
            msg = 'You have successfully registered!'
    elif request.method == 'POST':
        # Form is empty... (no POST data)
        msg = 'Please fill out the form!'
    # Show registration form with message (if any)
    return render_template('register.html', msg=msg)

# this will be the home page, only accessible for loggedin users


@app.route('/Main', methods=['GET', 'POST'])
def Main():

    # Check if user is loggedin
    if 'loggedin' in session:
        # User is loggedin show them the home page
        return render_template('Main.html', username=session['username'])

    # User is not loggedin redirect to login page
    return render_template('index.html', msg='')

# this will be the profile page, only accessible for loggedin users



@app.route('/profile')
def profile():
    # Check if user is loggedin
    if 'loggedin' in session:
        # We need all the account info for the user so we can display it on the profile page
        cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute('SELECT * FROM accounts WHERE id = %s',
                    (session['id'],))
        account = cursor.fetchone()
        # Show the profile page with account info
        return render_template('profile.html', account=account)
    # User is not loggedin redirect to login page
    return redirect(url_for('login'))



@app.route('/predict', methods=['POST'])
def predict():
    mytext = []
    text = []
    plant = []
    mypath = []
    file_path = []
    if request.method == "POST":
        f = request.files['image']
        basepath = os.path.dirname(__file__)
        file_path = os.path.join(basepath, 'uploads', f.filename)
        f.save(file_path)

        img = image.load_img(file_path, target_size=(128, 128))
        x = image.img_to_array(img)
        x = np.expand_dims(x, axis=0)
        plant = request.form['plant']
        if (plant == "fruit"):
            y = np.argmax(model_fruit.predict(x), axis=1)
            predict = ['Apple___Black_rot', 'Apple___healthy',
                    'Corn_(maize)___Northern_Leaf_Blight', 'Corn_(maize)___healthy', 'Peach___Bacterial_spot',
'Peach___healthy']
            df = pd.read_excel('precautions - fruits.xlsx')
            text = df.iloc[y[0]]['caution']
            print(df.iloc[y[0]]['caution'])
        else:
            y = np.argmax(model_veg.predict(x), axis=1)
            predict = ['Pepper,_bell___Bacterial_spot', 'Pepper,_bell___healthy', 'Potato___Early_blight',
'Potato___healthy',
                    'Potato___Late_blight', 'Tomato___Bacterial_spot', 'Tomato___Late_blight',
'Tomato___Leaf_Mold', 'Tomato___Septoria_leaf_spot']
            df = pd.read_excel('precautions - veg.xlsx')
            text = df.iloc[y[0]]['caution']
    return render_template('Main.html', mytext=text, mypath=file_path)



if __name__ == '__main__':
    app.run(debug=True)
```

home.html

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

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Fertilizer Recommendation System</title>
    <link rel="stylesheet" href="{{url_for('static',filename='style.css')}}">
</head>

<body>
    <div>
        <ul id="navbar">
            <li><a href="{{ url_for('home') }}"><span>HOME</span> </a> </li>
            <a href="{{url_for('login')}}"><button class="button1">LOGIN</button></a>
            <a href="{{url_for('register')}}"><button class="button1">REGISTER</button></a>
        </ul>

    </div>

    <div class="section c-store-home-wrap">
        <div class="intro-header">
            <div class="intro-content cc-homepage">
                <div class="intro-text">
                    <div class="heading-jumbo">Fertilizer<br> Recommendation<br>System</div>
                    <div class="text2">"Weeds are Flowers too, when you get to know them" </div>
                    <a href="/about" class="button2">
                        <div>Learn More</div>
                    </a>
                </div>


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

</html>
```

Index.html

```
<!DOCTYPE html>
<html>
    <head>
            <meta charset="utf-8">
            <title>Login</title>
            <link rel="stylesheet" href="{{ url_for('static', filename='reg_style.css') }}">
            <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.1/css/all.css">
    </head>
    <body>
            <div class="login">
                    <h1>Login</h1>
                    <div class="links">
                            <a href="{{ url_for('login') }}" class="active">Login</a>
                            <a href="{{ url_for('register') }}">Register</a>
                    </div>
                    <form action="{{ url_for('login') }}" method="post">
                            <label for="username">
                                    <i class="fas fa-user"></i>
                            </label>
                            <input type="text" name="username" placeholder="Username" id="username"
required>
                            <label for="password">
                                    <i class="fas fa-lock"></i>
                            </label>
                            <input type="password" name="password" placeholder="Password" id="password"
required>
                            <div class="msg">{{ msg }}</div>
                            <input type="submit" value="Login">
                    </form>
            </div>
    </body>
</html>
```

register.html

```
<!DOCTYPE html>
<html>
    <head>
            <meta charset="utf-8">
            <title>Register</title>
            <link rel="stylesheet" href="{{ url_for('static', filename='reg_style.css') }}">
            <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.1/css/all.css">
    </head>
    <body>
            <div class="register">
                    <h1>Register</h1>
                    <div class="links">
                            <a href="{{ url_for('login') }}">Login</a>
                            <a href="{{ url_for('register') }}" class="active">Register</a>
                    </div>
                    <form action="{{ url_for('register') }}" method="post" autocomplete="off">
                            <label for="username">
                                    <i class="fas fa-user"></i>
                            </label>
                            <input type="text" name="username" placeholder="Username" id="username"
required>
                            <label for="password">
                                    <i class="fas fa-lock"></i>
                            </label>
                            <input type="password" name="password" placeholder="Password" id="password"
required>
                            <label for="email">
                                    <i class="fas fa-envelope"></i>
                            </label>
                            <input type="email" name="email" placeholder="Email" id="email" required>
                            <div class="msg">{{ msg }}</div>
                            <input type="submit" value="Register">
                    </form>
            </div>
    </body>
</html>
```

profile.html

```
<html>
<head>
    <link rel="stylesheet" href="{{ url_for('static', filename='reg_style.css') }}">
    <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.1/css/all.css">
</head>
<body><div>
    <nav class="navtop">
        <div>
            <h1>Website Title</h1>
            <a href="{{ url_for('Main') }}"><i class="fas fa-home"></i>Home</a>
            <a href="{{ url_for('profile') }}"><i class="fas fa-user-circle"></i>Profile</a>
            <a href="{{ url_for('logout') }}"><i class="fas fa-sign-out-alt"></i>Logout</a>
        </div>
    </nav></div>
    <div class="content">
        <h2>Profile Page</h2>
        <div>
            <p>Your account details are below:</p>
            <table>
                <tr>
                    <td>Username:</td>
                    <td>{{ account['username'] }}</td>
                </tr>
                <tr>
                    <td>Password:</td>
                    <td>{{ account['password'] }}</td>
                </tr>
                <tr>
                    <td>Email:</td>
                    <td>{{ account['email'] }}</td>
                </tr> </table>
        </div>
    </div>
</body>
</html>
```

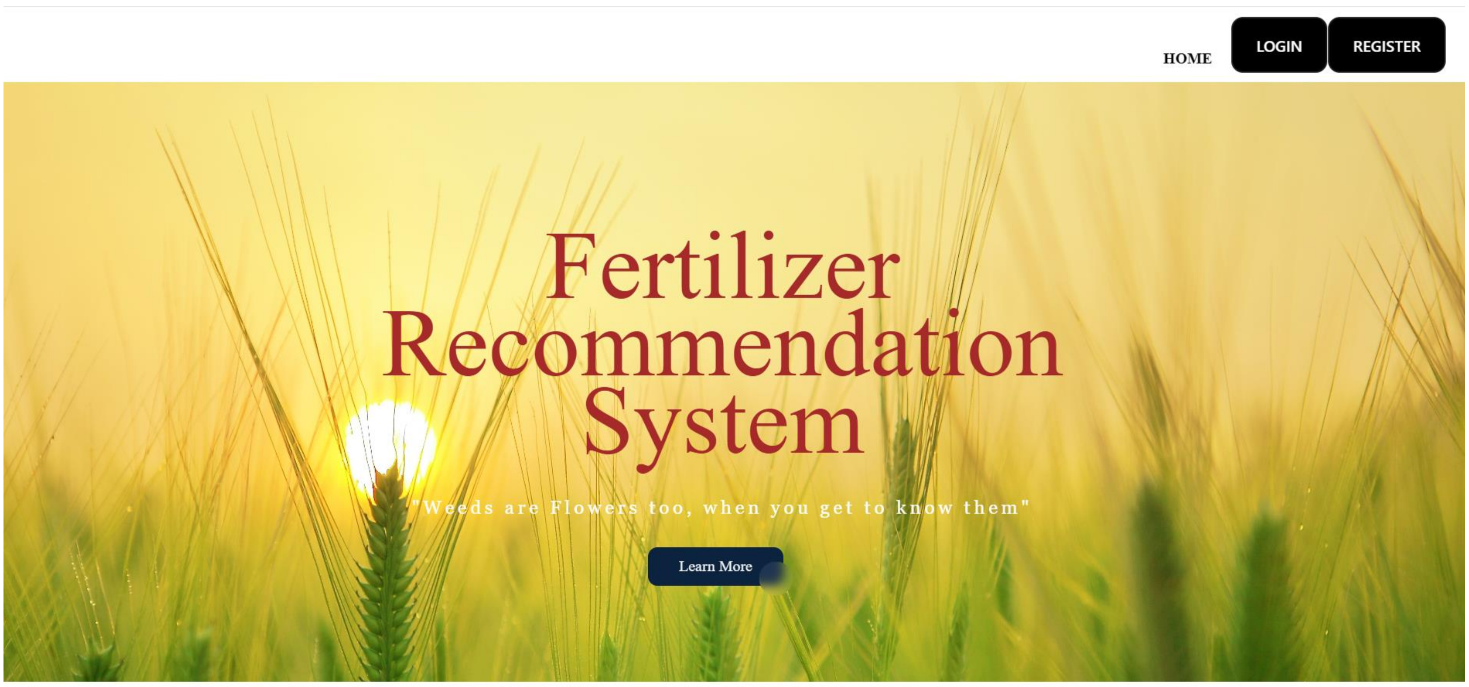
Main.html

```
<!DOCTYPE html>
<html>

<head>
    <meta charset="utf-8">
    <title>{% block title %}{% endblock %}</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='reg_style.css') }}">
    <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.1/css/all.css">
<style>
    #output-image{
            size: 100px;;
    }
</style>

</head>

<body class="loggedin">
    <nav class="navtop">
            <div>
                    <h1>Website Title</h1>
                    <a href="{{ url_for('Main') }}"><i class="fas fa-home"></i>Home</a>
                    <a href="{{ url_for('profile') }}"><i class="fas fa-user-circle"></i>Profile</a>
                    <a href="{{ url_for('logout') }}"><i class="fas fa-sign-out-alt"></i>Logout</a>
            </div>
    </nav>
    <div class="split left">
            <div class="centered">
                    <img src="{{ url_for('static', filename='Animated_img.jpg') }}">
            </div>
    </div>
    <form method=post enctype=multipart/form-data>
            <div class="split right">
                    <div class="centered">
                            <h2 class="h4 mb-3 font-weight-normal">Upload Your Image</h2>
                            <input    type="file"    name="file"    class="form-control-file"    id="inputfile"
onchange="preview_image(event)"
                                    style="font-weight: bold;">
                    <br>
                    <br>
                    <img id="output-image" />
                    <div class="Submit_button">
                        <a href="{{url_for('predict')}}"><button class="button3">PREDICT</button></a>
                    </div>
                    <!---<div>
                    <label for="images" class="drop-container">
                        <input type="file" name="file" onchange="preview_image(event)" accept="image/*" required>
                        <img id="output-image" class="rounded mx-auto d-block" />
                    </label>
                </div>
                --->
    </form>

    </div>
    </div>
    <script type="text/javascript">
        function preview_image(event) {
            var reader = new FileReader();
            reader.onload = function () {
                var output = document.getElementById('output-image')
                output.src = reader.result;
            }
            reader.readAsDataURL(event.target.files[0]);
        }
    </script>
</body>

</html>
```

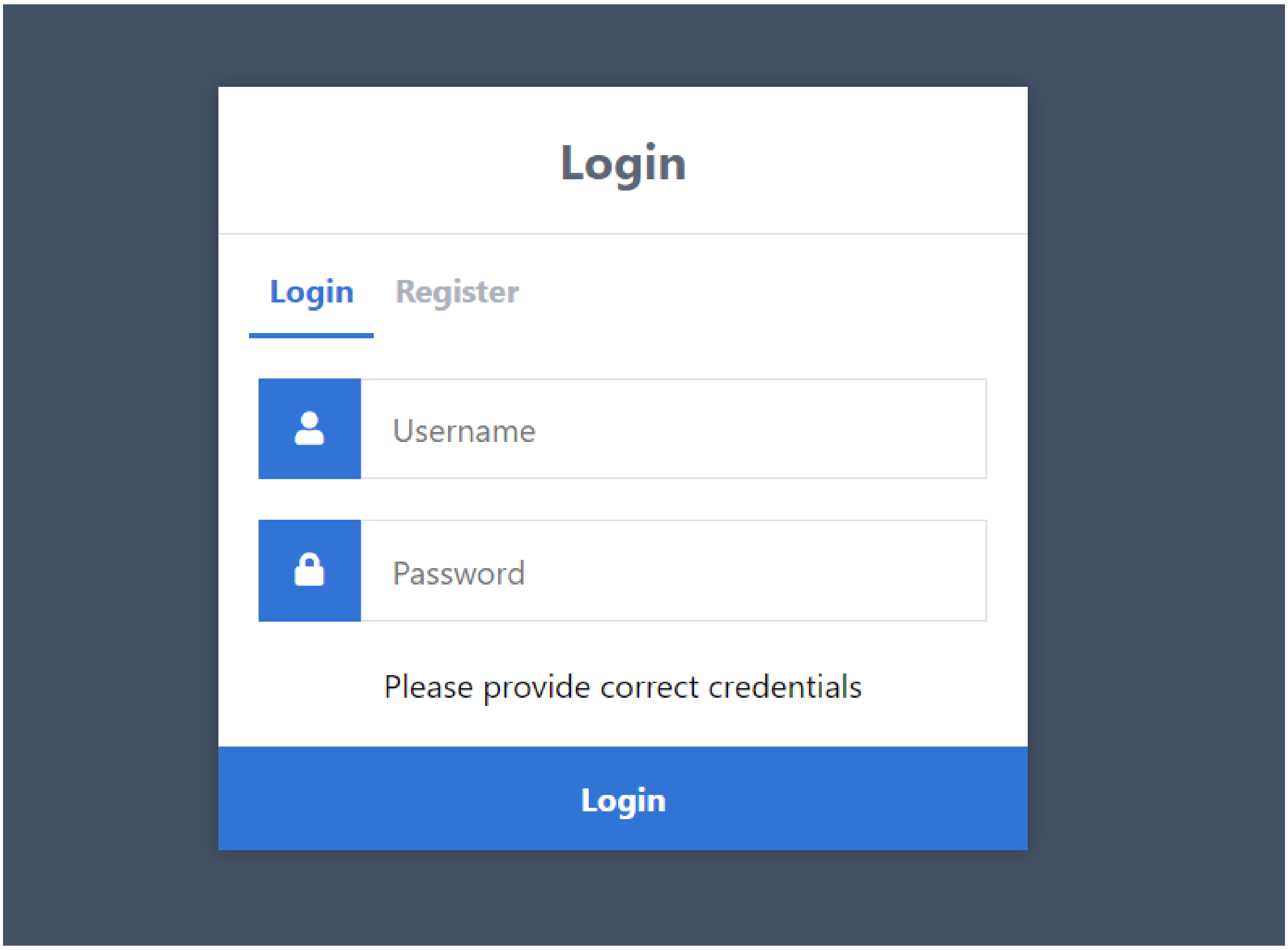
APPENDIX B

RESULT PAGE SCREENSHOTS

Home Page

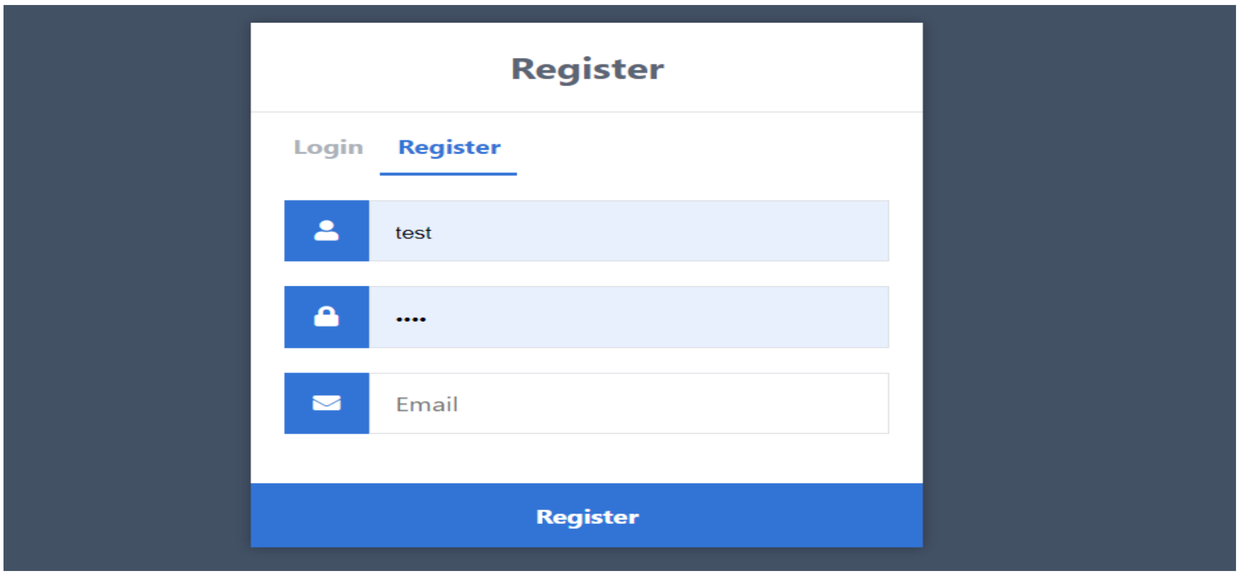

HOME
LOGIN
REGISTER
Fertilizer Recommendation System
"Weeds are Flowers too, when you get to know them"
Learn More

Login Page



Please provide correct credentials

Register

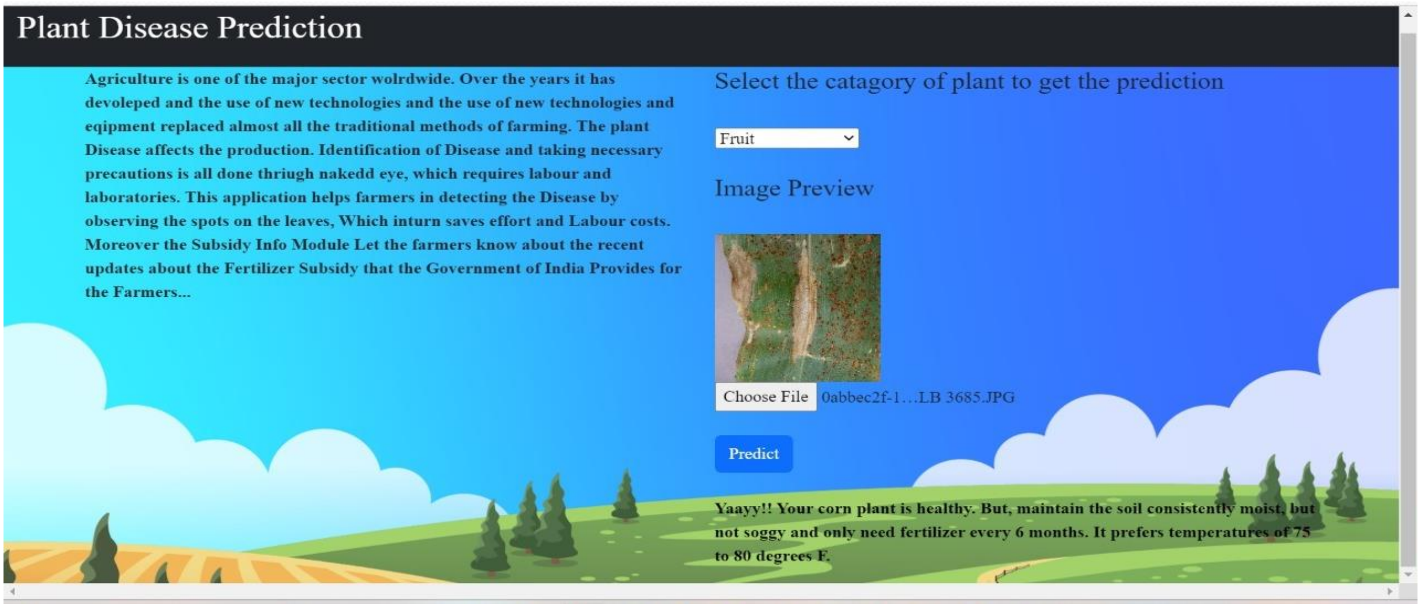

Register

test

••••

Email

Fruit Disease Detection and Fertilizer Recommendation


Plant Disease Prediction

Agriculture is one of the major sector wolrdwide. Over the years it has devoleped and the use of new technologies and the use of new technologies and eqipment replaced almost all the traditional methods of farming. The plant Disease affects the production. Identification of Disease and taking necessary precautions is all done thriugh nakedd eye, which requires labour and laboratories. This application helps farmers in detecting the Disease by observing the spots on the leaves, Which inturn saves effort and Labour costs. Moreover the Subsidy Info Module Let the farmers know about the recent updates about the Fertilizer Subsidy that the Government of India Provides for the Farmers...

Select the catagory of plant to get the prediction

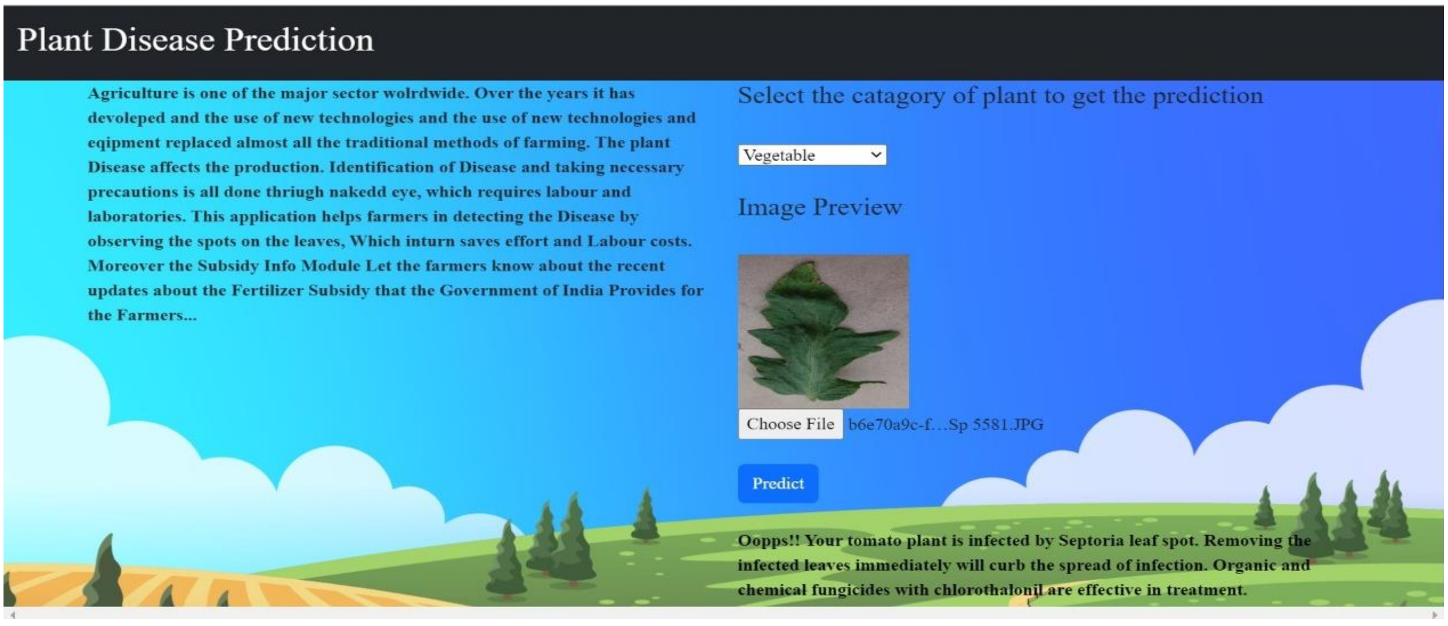
Fruit

Image Preview

Choose File 0abbec2f-1…LB 3685.JPG

Yaayy!! Your corn plant is healthy. But, maintain the soil consistently moist, but not soggy and only need fertilizer every 6 months. It prefers temperatures of 75 to 80 degrees F.

Vegetable Disease Detection and Fertilizer Recommendation


Plant Disease Prediction

Agriculture is one of the major sector wolrdwide. Over the years it has devoleped and the use of new technologies and the use of new technologies and eqipment replaced almost all the traditional methods of farming. The plant Disease affects the production. Identification of Disease and taking necessary precautions is all done thriugh nakedd eye, which requires labour and laboratories. This application helps farmers in detecting the Disease by observing the spots on the leaves, Which inturn saves effort and Labour costs. Moreover the Subsidy Info Module Let the farmers know about the recent updates about the Fertilizer Subsidy that the Government of India Provides for the Farmers...

Select the catagory of plant to get the prediction

Vegetable

Image Preview

Choose File b6e70a9c-f…Sp 5581.JPG

Oopps!! Your tomato plant is infected by Septoria leaf spot. Removing the infected leaves immediately will curb the spread of infection. Organic and chemical fungicides with chlorothalonil are effective in treatment.

APPENDIX C

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-10022-1659088839