FERTILIZERS RECOMMENDATION SYSTEM FOR DISEASE PREDICTION

TEAM ID: PNT2022TMID08559

TEAM MEMBERS:

VISHWANANTHA R	(810419205054)
SATHISHKUMAR R	(810419205040)
DEVA K	(810419205011)
JAYARAJ K	(810419205020)

CONTENTS

1. INTRODUCTION

1.1 PROJECT OVERVIEW

Agriculture serves as a means of supplying food to a population that is always expanding, as well as a significant source of energy and a means of combating global warming. Plant diseases are very important because they can have a negative impact on the quality and quantity of crops produced in agriculture. Early detection of plant diseases is crucial for their treatment and management. Typically, illnesses are identified using the naked eye technique. Experts who can recognise variations in leaf colour are involved in this process. This method requires a lot of work, takes a while, and is not appropriate for fields with a lot of space. The same ailment is frequently classified differently by various experts. Costly specialist monitoring is required for this procedure, which makes it pricey. Plant diseases can drive up the cost of agricultural production and, if left untreated at an early stage, could spell complete financial ruin for a producer. In order to stop the spread of a plant disease at a low cost and save the majority of the production, farmers must keep an eye on their crops and recognise the first symptoms. It may be expensive to hire experienced agriculturists, particularly in remote, isolated geographic areas .Various experts regularly assign multiple classifications to the same illness. This operation is costly because it calls for pricy professional supervision. Plant diseases can increase the cost of agricultural production and, if not promptly treated, could result in a producer's total financial ruin. Farmers must keep an eye on their crops and be able to spot the first symptoms in order to stop the spread of a plant disease at a low cost and save the majority of the production. Agriculturists with experience may be expensive to hire, especially in isolated, distant locations.

1.2 PURPOSE

They forecast plant disease and recommend fertiliser for the damaged plants. This frequently involves a range of methods for assessing the qualities of the herbs that largely influence the plants. These complex systems that contain a large amount of datasets are forecasted using the neural network. By using artificial intelligence, complex manual systems' working models can be made simpler and more precise.

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

Leaves are the most obvious and widespread choice for tree species recognition, even though the botanical classification was not built upon their properties. They can be found almost all year long, are easy to photograph, and their shapes present well studied specificities that make the identification, if not trivial, possible. Our goal with the Folia application is then to build a system for leaf shape analysis that processes, unlike what has been done to date, pictures in a natural environment. With the aim of being an educational tool, it relies on high-level geometric criteria inspired by those used by botanists, that make a semantic interpretation possible, to classify a leaf into a list of species. Digital image processing will improve the quality of the image by removing noise & other unwanted pixels and obtain more information from image. Image segmentation is a mid-level processing technique used to analyze the image and can be used to classify or cluster an image into several disjoint parts by grouping the pixels to form a region of homogeneity based on the pixel characteristics like gray level, color, texture, intensity and other features. The main purpose of the segmentation process is to get more information about the image, the region we are interested in and to clearly differentiate the object and the background in an image. The criteria for segmenting the image is very hard to decide as it varies from image to image and also varies significantly on the modal quality of image. In some cases interactive methods can be laborious and time consuming and in some cases manual interaction to segment the image may be error-prone while the fully automated approach can give error output.

2.1.1LIMITATIONS OF EXISTING SYSTEM:

- Suffer in the local minima problems.
- Dimensionality is high to produce large number of irrelevant features.
- User defined segmentation can be done.
- Does not recommend the fertilizers to leaves diseases.

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] Naresh, Y. G., and H. S. Nagendraswamy, "Classification of medicinal plants: an approach using modified LBP with symbolic representation", Neurocomputing 173, pp: 1789-1797, 2016.

[5] 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.

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

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

[8] 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.

[9] Joly, Alexis, HervéGoéau, HervéGlotin, ConcettoSpampinato, Pierre Bonnet, Willem-Pier Vellinga, Julien Champ, Robert Planqué, Simone Palazzo, and Henning Müller, "LifeCLEF 2016: multimedia life species identification challenges", In International Conference of the Cross-Language Evaluation Forum for European Languages, pp. 286- 310, Springer, Cham, 2016.

[10] Zeiler, Matthew D., and Rob Fergus, "Visualizing and understanding convolutional networks", In European conference on computer vision, pp. 818-833. Springer, Cham, 2014.
[11] 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.

[12] 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.

[13] Nguyen, ThiThanhNhan, ThiLan Le Van Tuan Le, Hai Vu, NataponPantuwong, and Yasushi Yagi, "Flower species identification using deep convolutional neural networks", 2015.

[14] Mohanty, Sharada P., David P. Hughes, and Marcel Salathé, "Using deep learning for image-based plant disease detection", Frontiers in plant science 7, pp: 1419, 2016.

[15] Chaulagain, Basanta, Bhuwan Bhatt, BipinKhatiwada, and BishalChaulagain. "Plant Leaf Recognition", 2013.

[16] Pujari, Devashish, Rajesh Yakkundimath, and Abdulmunaf S. Byadgi. "SVM and ANN based classification of plant diseases using feature reduction technique." IJIMAI 3, no. 7 (2016): 6-14.

[17] 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.

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

[19] Akbarzadeh, Saman, AriePaap, SelamAhderom, BeniaminApopei, and Kamal Alameh. "Plant discrimination by Support Vector Machine classifier based on spectral reflectance." Computers and electronics in agriculture 148 (2018): 250-258.https://www.sciencedirect.com/science/article/pii/ S0168169917310268.

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

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

[22] 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.

[23] Uzal, Lucas C., Guillermo L. Grinblat, Rafael Namías, Mónica G. Larese, J. S. Bianchi, E. N. Morandi, and Pablo M. Granitto. "Seed-per-pod estimation for plant breeding using deep learning." Computers and electronics in agriculture 150 (2018): 196-204.

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

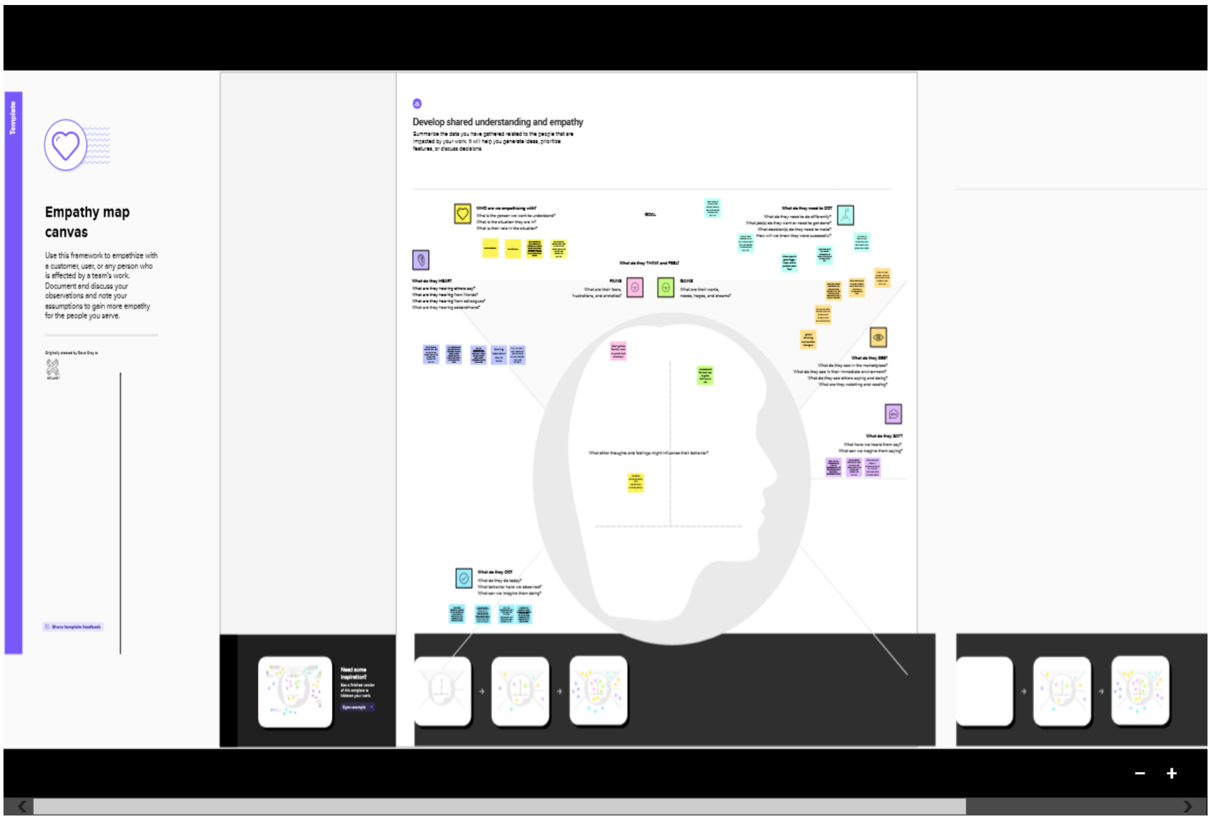
[25] Sladojevic, Srdjan, Marko Arsenovic, AndrasAnderla, DubravkoCulibrk, and DarkoStefanovic. "Deep neural networks based recognition of plant diseases by leaf image classification." Computational intelligence and neuroscience2016 (2016).

2.3 PROBLEM STATEMENT DEFINITION

Farmers' conventional methods of agricultural cultivation are ineffective. It does not make proper use of all available resources. Farmers are unable to detect crop diseases due to a lack of knowledge and old practices, which often result in soil nutrient deterioration and exhaustion. As a result, crop failure occurs. Growing only certain crops depletes the soil, and if the crops are harmed by illnesses, farmers are uninformed of how to recover such crops. Food needs cannot be met until and unless efficient resource management and use is implemented.

3. IDEATION & PROPOSED SOLUTION

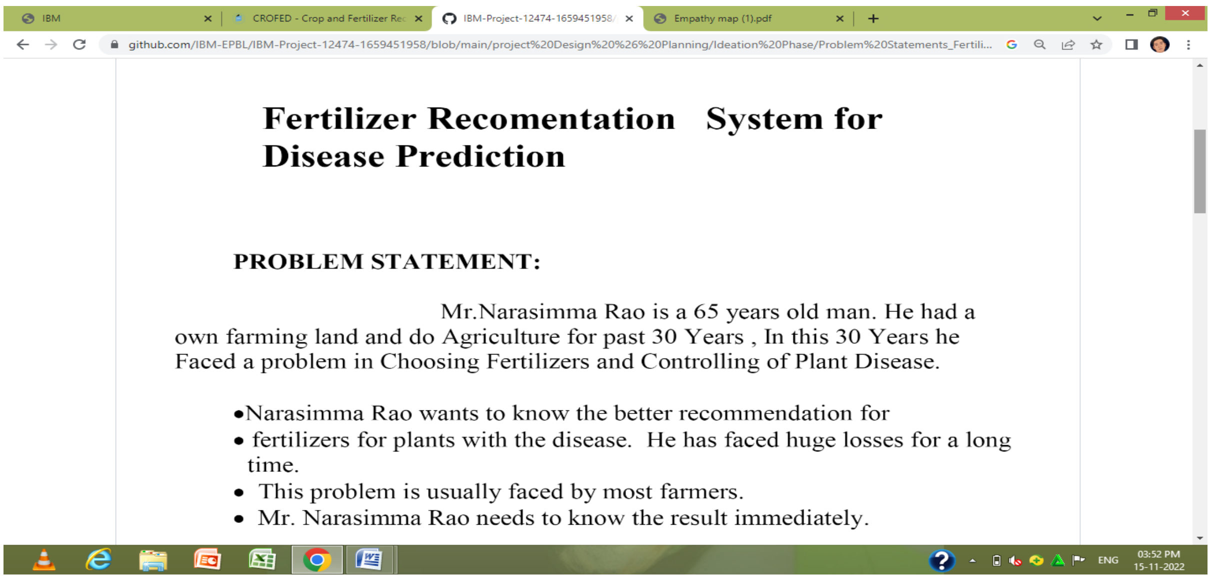
3.1 EMPATHY MAP CANVAS



Template
Empathy map canvas
Use this framework to empathize with a customer, user, or any person who is affected by a team's work. Document and discuss your observations and note your assumptions to gain more empathy for the people you serve.
Develop shared understanding and empathy
Summarize the data you have gathered related to the people that are impacted by your work. It will help you generate ideas, prioritize features, or discuss decisions.
What do they THINK and FEEL?
What are their fears, frustrations, and anxieties?
What are their wants, needs, hopes, and dreams?
What are they hearing others say?
What are they hearing from friends?
What are they hearing from colleagues?
What are they hearing second-hand?
What do they need to do differently?
How will we know they were successful?
What do they see in the marketplace?
What do they see in their immediate environment?
What do they see others saying and doing?
What are they watching and reading?
− +

3.2 IDEATION & BRAINSTORMING

3.2.1 IDEATION :

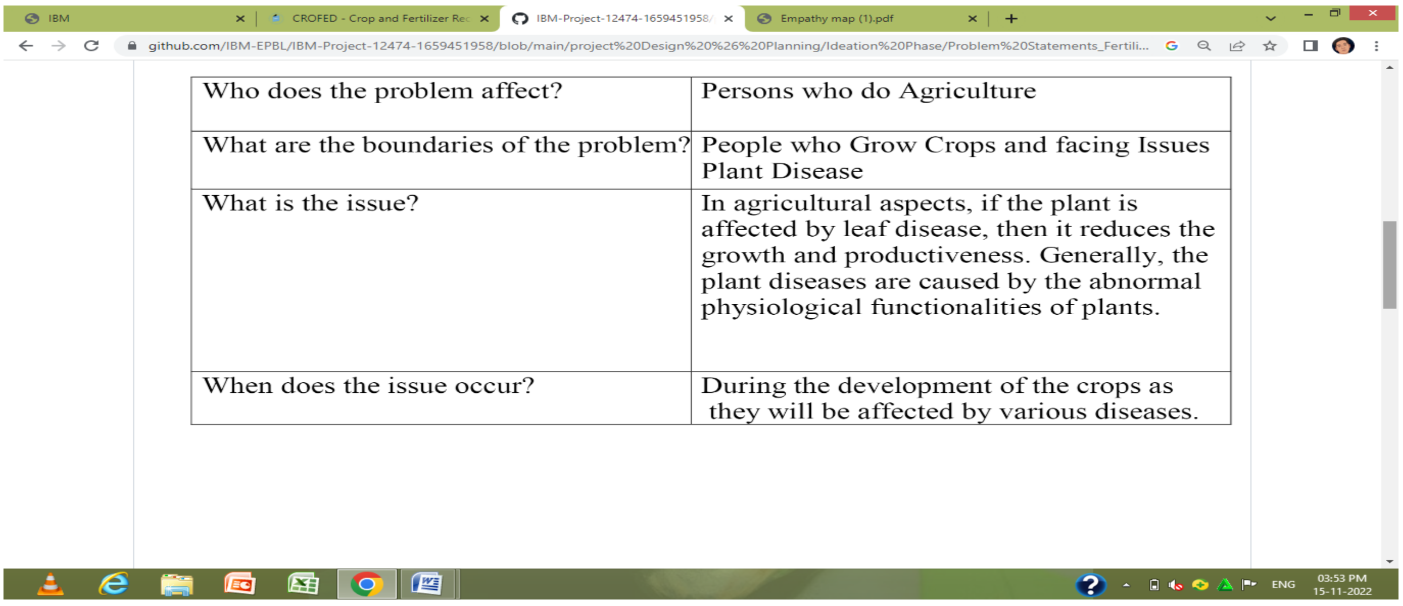


Fertilizer Recomentation System for Disease Prediction

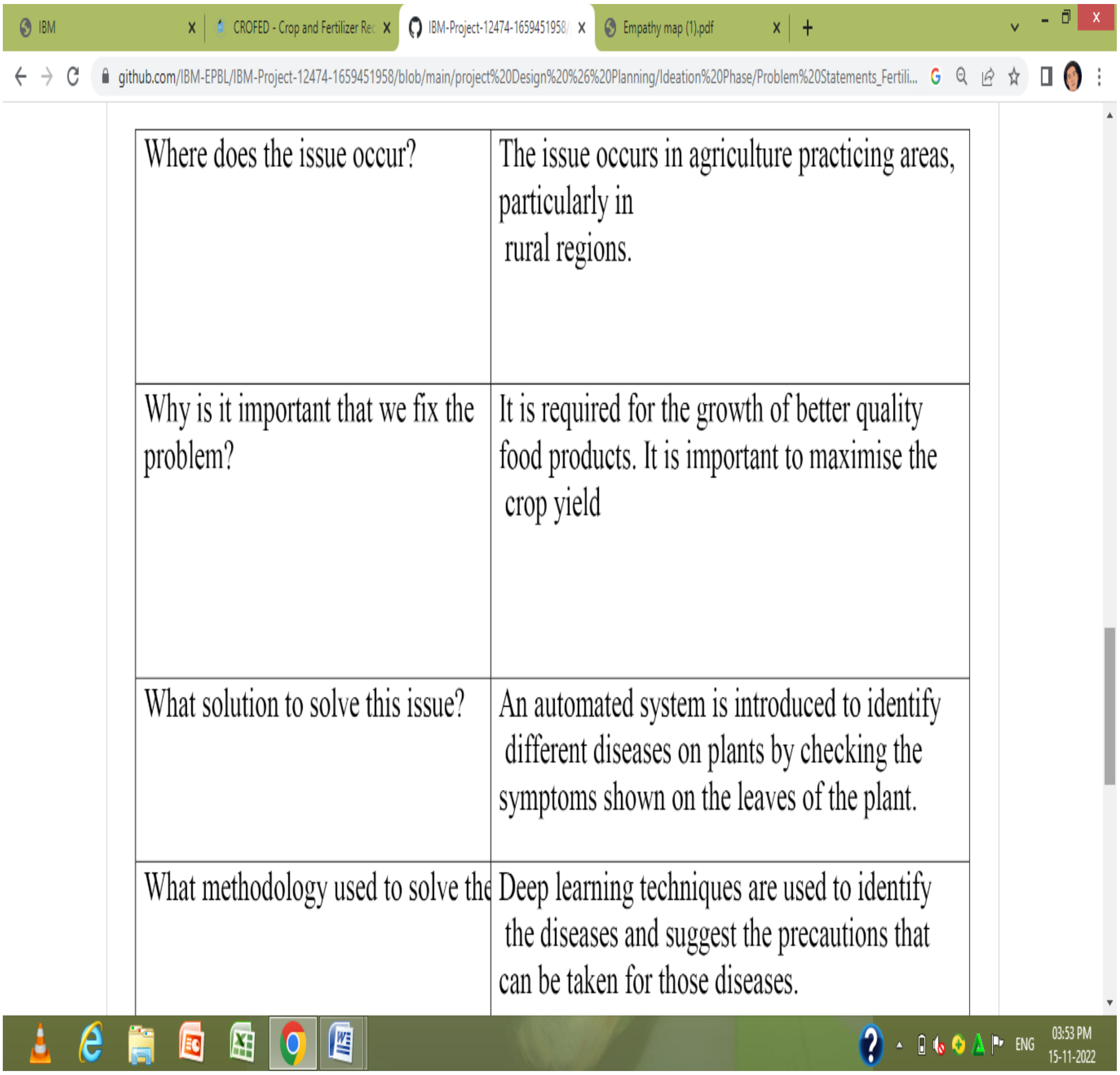
PROBLEM STATEMENT:

Mr.Narasimma Rao is a 65 years old man. He had a own farming land and do Agriculture for past 30 Years , In this 30 Years he Faced a problem in Choosing Fertilizers and Controlling of Plant Disease.

- Narasimma Rao wants to know the better recommendation for
- fertilizers for plants with the disease. He has faced huge losses for a long time.
- This problem is usually faced by most farmers.
- Mr. Narasimma Rao needs to know the result immediately.



Who does the problem affect?	Persons who do Agriculture
What are the boundaries of the problem?	People who Grow Crops and facing Issues Plant Disease
What is the issue?	In agricultural aspects, if the plant is affected by leaf disease, then it reduces the growth and productiveness. Generally, the plant diseases are caused by the abnormal physiological functionalities of plants.
When does the issue occur?	During the development of the crops as they will be affected by various diseases.

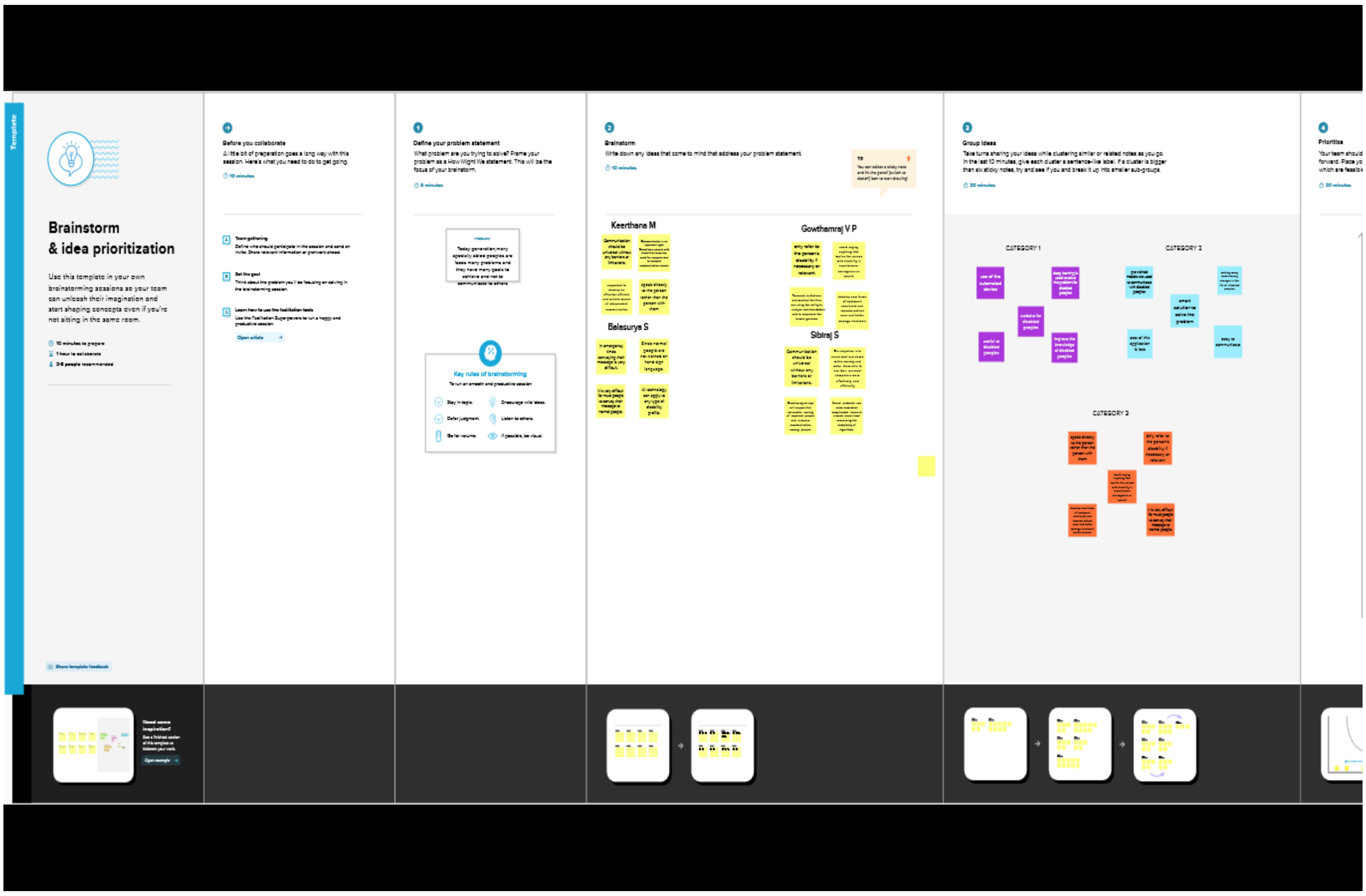


IBM | CROFED - Crop and Fertilizer Rec | IBM-Project-12474-1659451958/ | Empathy map (1).pdf

github.com/IBM-EPBL/IBM-Project-12474-1659451958/blob/main/project%20Design%20%26%20Planning/Ideation%20Phase/Problem%20Statements_Fertili...

Where does the issue occur?	The issue occurs in agriculture practicing areas, particularly in rural regions.
Why is it important that we fix the problem?	It is required for the growth of better quality food products. It is important to maximise the crop yield
What solution to solve this issue?	An automated system is introduced to identify different diseases on plants by checking the symptoms shown on the leaves of the plant.
What methodology used to solve the	Deep learning techniques are used to identify the diseases and suggest the precautions that can be taken for those diseases.

3.2.2 BRAINSTORMING :


Template

Brainstorm
& idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

10 minutes to prepare
1 hour to collaborate
2-8 people recommended

Before you collaborate
A little bit of preparation goes a long way with this session. Here's what you need to do to get going.
10 minutes

A Team gathering
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B Set the goal
Think about the problem you'll be focusing on solving in the brainstorming session.

C Learn how to use the facilitation tools
Use the Facilitation Superpowers to run a happy and productive session.

Open article

1 Define your problem statement
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.
5 minutes

PROBLEM
Today generation, many specially abled peoples are faces many problems and they have many goals to achieve and not to communicate to others.

Key rules of brainstorming
To run an smooth and productive session

Stay in topic.
Encourage wild ideas.
Defer judgment.
Listen to others.
Go for volume.
If possible, be visual.

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!

Keerthana M

Communication should be universal without any barriers or limitations.

[illegible]

[illegible]

speak directly to the person rather than the person with them.

Gowthamraj V P

only refer to the person's disability if necessary or relevant.

[illegible]

[illegible]

[illegible]

Balasurya S

In emergency times conveying their message is very difficult.

Since normal people are not trained on hand sign language.

It is very difficult for mute people to convey their message to normal people.

AI technology can apply to any type of disability profile.

Sibiraj S

Communication should be universal without any barriers or limitations.

[illegible]

[illegible]

[illegible]

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

CATEGORY 1

use of the automated devices

[illegible]

website for disabled peoples

useful to disabled peoples

improve the knowledge of disabled peoples

CATEGORY 2

[illegible]

[illegible]

smart solution to solve the problem

ease of this application is less

easy to communicate

CATEGORY 3

speak directly to the person rather than the person with them

only refer to the person's disability if necessary or relevant.

[illegible]

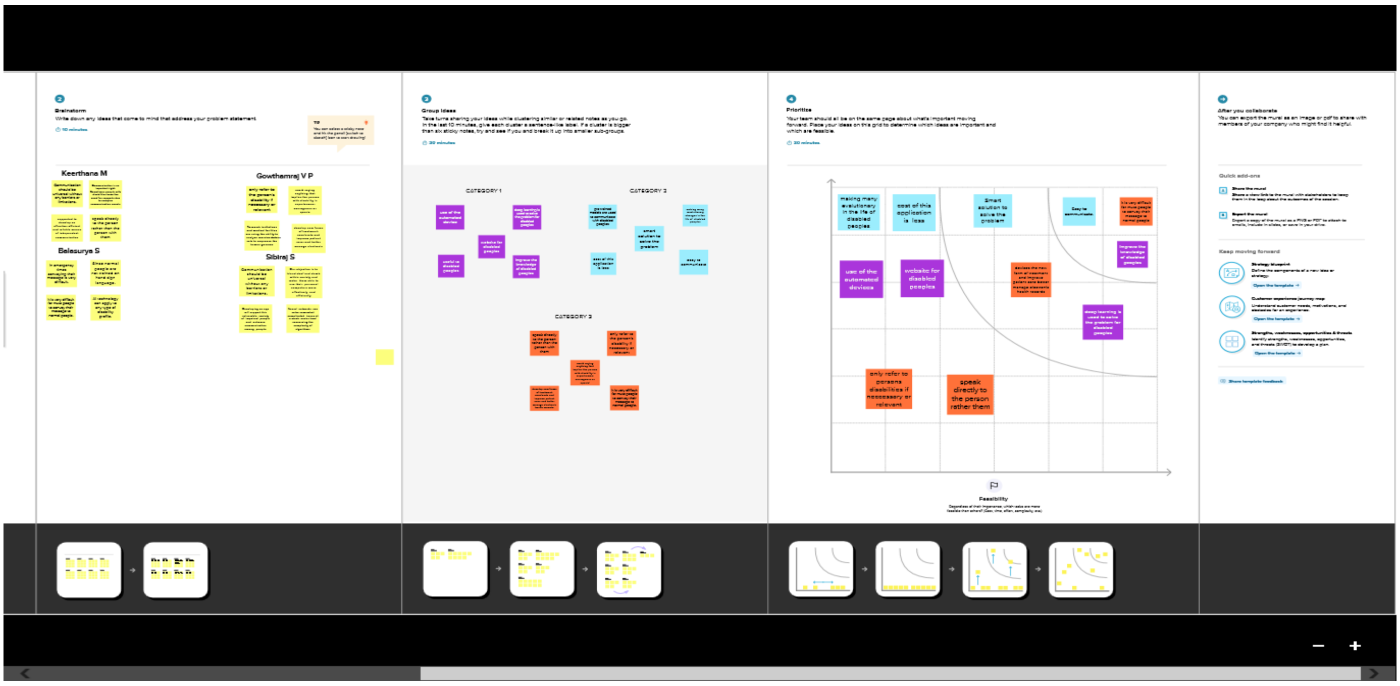
[illegible]

It is very difficult for mute people to convey their message to normal people.

4 Prioritize
Your team should … forward. Place yo… which are feasib…
20 minutes

Share template feedback

Need some inspiration?
See a finished version of this template to kickstart your work.
Open example



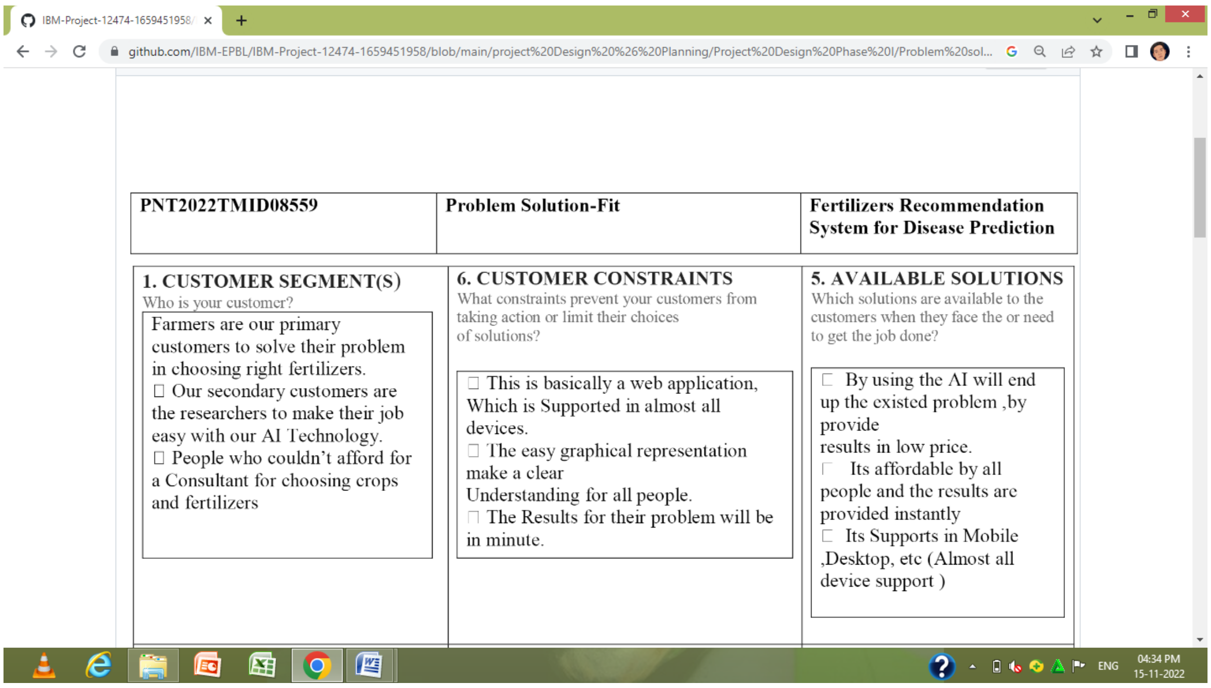
3.3 PROPOSED SOLUTION :

Even when considering trees only, leaves show an impressively wide variety in shapes. It is however necessary to come up with a representation of what a leaf is, that is accurate enough to be fitted to basically any kind of leaf. The general shape of a leaf is a key component of the process of identifying a leaf. Botanists have a whole set of terms describing either the shape of a simple leaf, of the lobes of a palmate leaf, or of the leaflets of a compound leaf. Here present a study on segmentation of leaf images restricted to semi-controlled conditions, in which leaves are photographed against a solid light-colored background. Such images can be used in practice for plant species identification, by analyzing the distinctive shapes of the leaves. The most important of these are: the variety of leaf shapes, inevitable presence of shadows and specularities, and the time constraints required by interactive species identification applications. The identification of species is the first and essential key to understand the plant environment. In this project introduce a method designed to deal with the obstacles raised by such complex images, for simple and lobed tree leaves. A first segmentation step based on a light polygonal leaf model is first performed, and later used to guide the evolution of an active contour. Combining global shape descriptors given by the polygonal model with local curvature-based features, the leaves are then classified over leaf datasets. In this project we introduce a method designed to deal with the obstacles raised by such complex images, for simple and lobed tree 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.

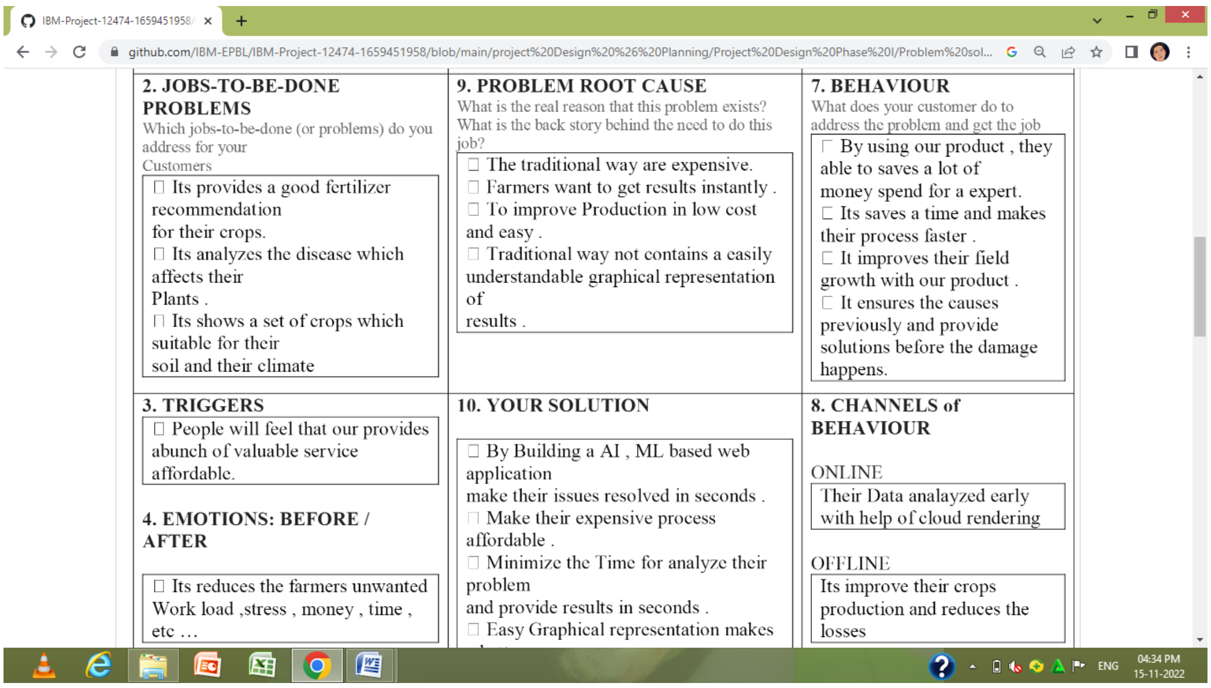
3.3.1ADVANTAGE OF PURPOSED SYSTEM:

- Segmentation can be done easily to spilt the tree parts.
- Classify the affected parts in leaves.
- Eliminate redundant features of images.
- Provide improved accuracy rate.

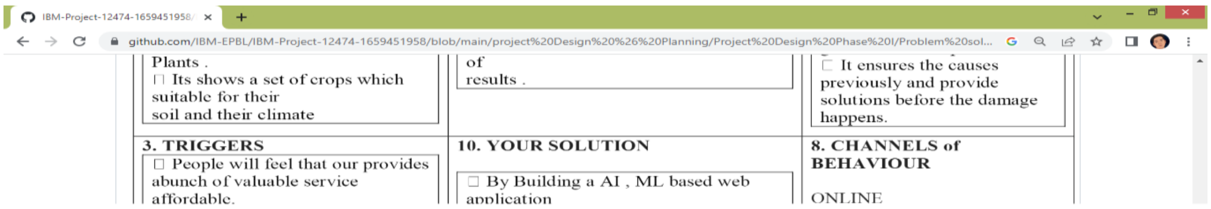
3.4 PROBLEM SOLUTION FIT :


IBM-Project-12474-1659451958/
github.com/IBM-EPBL/IBM-Project-12474-1659451958/blob/main/project%20Design%20%26%20Planning/Project%20Design%20Phase%20I/Problem%20sol...
ENG
04:34 PM
15-11-2022

PNT2022TMID08559	**Problem Solution-Fit**	**Fertilizers Recommendation System for Disease Prediction**
1. CUSTOMER SEGMENT(S) Who is your customer? Farmers are our primary customers to solve their problem in choosing right fertilizers. □ Our secondary customers are the researchers to make their job easy with our AI Technology. □ People who couldn't afford for a Consultant for choosing crops and fertilizers	**6. CUSTOMER CONSTRAINTS** What constraints prevent your customers from taking action or limit their choices of solutions? □ This is basically a web application, Which is Supported in almost all devices. □ The easy graphical representation make a clear Understanding for all people. □ The Results for their problem will be in minute.	**5. AVAILABLE SOLUTIONS** Which solutions are available to the customers when they face the or need to get the job done? □ By using the AI will end up the existed problem ,by provide results in low price. □ Its affordable by all people and the results are provided instantly □ Its Supports in Mobile ,Desktop, etc (Almost all device support)


IBM-Project-12474-1659451958/
github.com/IBM-EPBL/IBM-Project-12474-1659451958/blob/main/project%20Design%20%26%20Planning/Project%20Design%20Phase%20I/Problem%20sol...
ENG
04:34 PM
15-11-2022

2. JOBS-TO-BE-DONE PROBLEMS Which jobs-to-be-done (or problems) do you address for your Customers □ Its provides a good fertilizer recommendation for their crops. □ Its analyzes the disease which affects their Plants . □ Its shows a set of crops which suitable for their soil and their climate	**9. PROBLEM ROOT CAUSE** What is the real reason that this problem exists? What is the back story behind the need to do this job? □ The traditional way are expensive. □ Farmers want to get results instantly . □ To improve Production in low cost and easy . □ Traditional way not contains a easily understandable graphical representation of results .	**7. BEHAVIOUR** What does your customer do to address the problem and get the job □ By using our product , they able to saves a lot of money spend for a expert. □ Its saves a time and makes their process faster . □ It improves their field growth with our product . □ It ensures the causes previously and provide solutions before the damage happens.
3. TRIGGERS □ People will feel that our provides abunch of valuable service affordable. **4. EMOTIONS: BEFORE / AFTER** □ Its reduces the farmers unwanted Work load ,stress , money , time , etc …	**10. YOUR SOLUTION** □ By Building a AI , ML based web application make their issues resolved in seconds . □ Make their expensive process affordable . □ Minimize the Time for analyze their problem and provide results in seconds . □ Easy Graphical representation makes	**8. CHANNELS of BEHAVIOUR** ONLINE Their Data analayzed early with help of cloud rendering OFFLINE Its improve their crops production and reduces the losses


IBM-Project-12474-1659451958/
github.com/IBM-EPBL/IBM-Project-12474-1659451958/blob/main/project%20Design%20%26%20Planning/Project%20Design%20Phase%20I/Problem%20sol...

Plants . □ Its shows a set of crops which suitable for their soil and their climate	of results .	□ It ensures the causes previously and provide solutions before the damage happens.
3. TRIGGERS □ People will feel that our provides abunch of valuable service affordable.	**10. YOUR SOLUTION** □ By Building a AI , ML based web application	**8. CHANNELS of BEHAVIOUR** ONLINE

4. REQUIREMENT ANALYSIS

REQUIREMENT SPECIFICATION

The technical specification requirement for the software products is the requirement specification. It lists the functional, performance, and security requirements for a specific software system. Additionally, usage scenarios from a user, operational, and administrative standpoint are provided in the requirements. A thorough overview of the software project is what the software requirements specification is meant to do. The target audience is given a description of the project's parameters, goals, user interface, hardware, and software needs.

4.1 FUNCTIONAL REQUIREMENT :

• Operating system	: Windows OS
• Front End	: C#.NET
• Back End	: SQL SERVER
• Application	: Windows Application
• Tool	: Visual Studio 2010

4.2 NON-FUNCTIONAL REQUIREMENT :

• Processor	: Dual core processor 2.6.0 GHZ
• RAM	: 2GB
• Hard disk	: 160 GB
• Compact Disk	: 650 Mb
• Keyboard	: Standard keyboard
• Monitor	: 15 inch color monitor

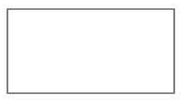
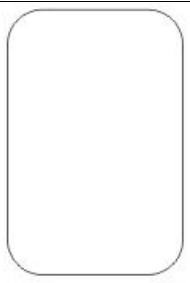
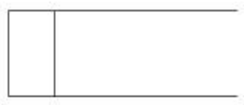
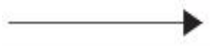
5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAM :

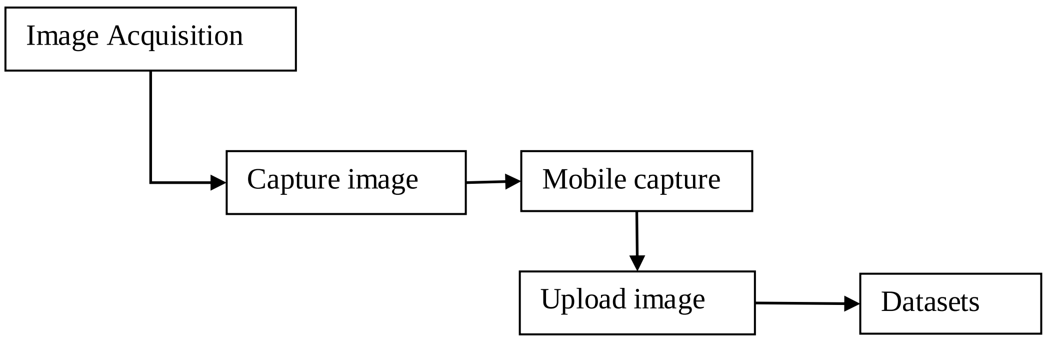
FLOW DIAGRAM

A two-dimensional diagram explains how data is processed and transferred in a system. The graphical depiction identifies each source of data and how it interacts with other data sources to reach a common output. Individuals seeking to draft a data flow diagram must identify external inputs and outputs, determine how the inputs and outputs relate to each other, and explain with graphics how these connections relate and what they result in. This type of diagram helps business development and design teams visualize how data is processed and identify or improve certain aspects.

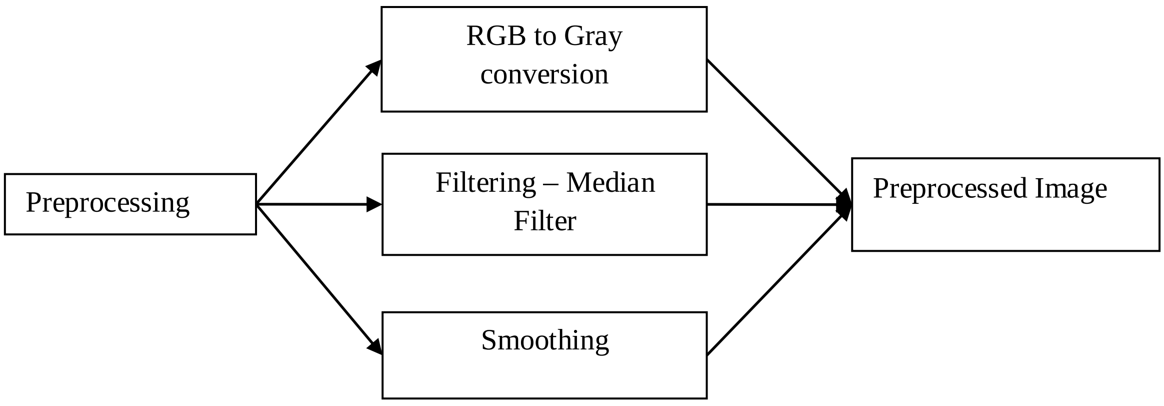
Data flow Symbols:

Symbol	Description
	An **entity**. A source of data or a destination for data.
	A **process** or task that is performed by the system.
	A **data store**, a place where data is held between processes.
	A **data flow**.

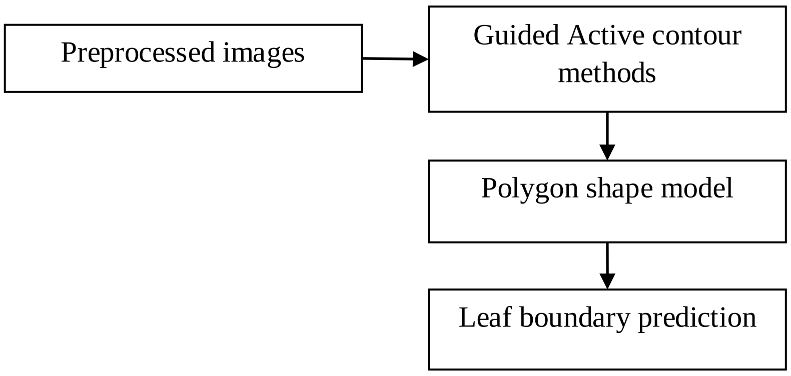
Level 0



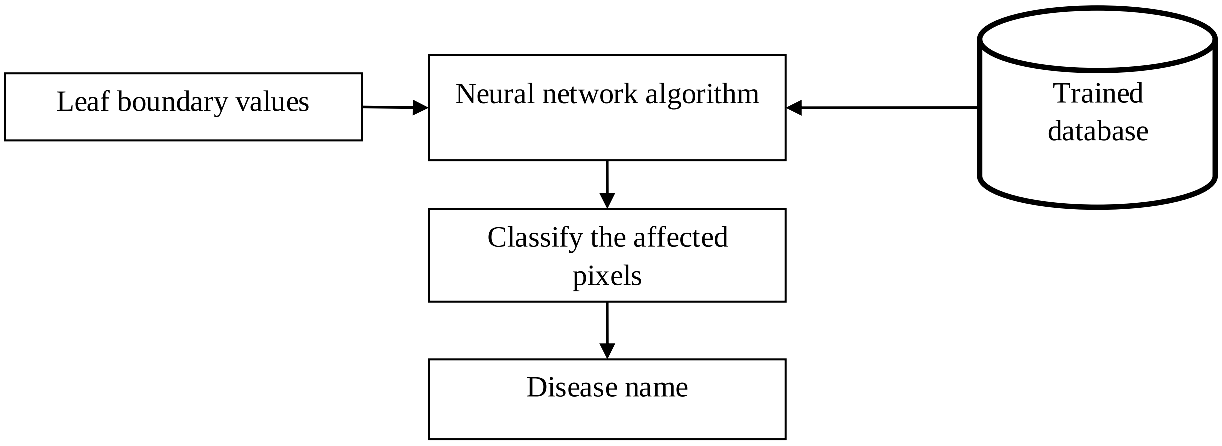
Level 1



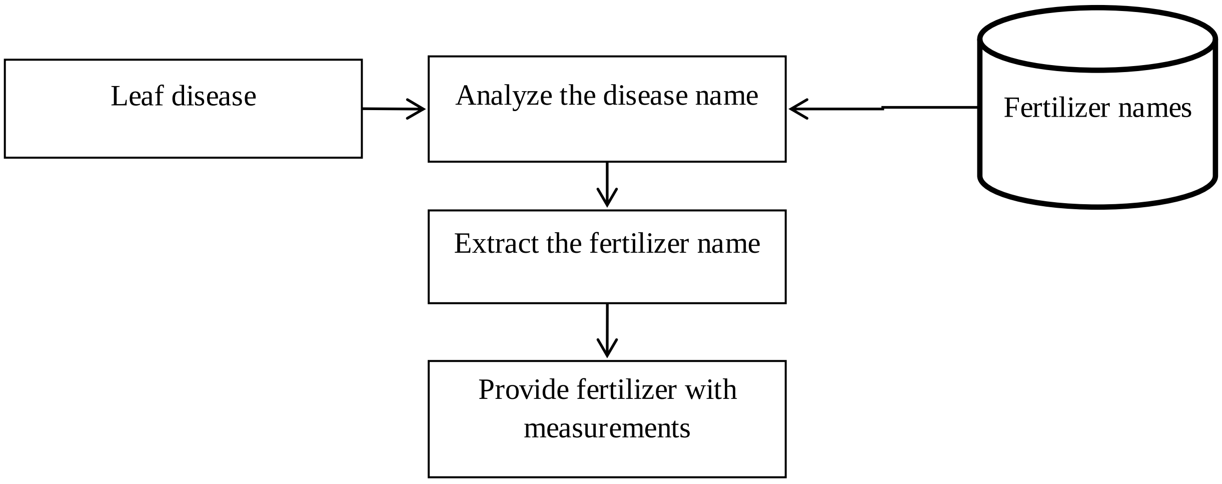
Level 2



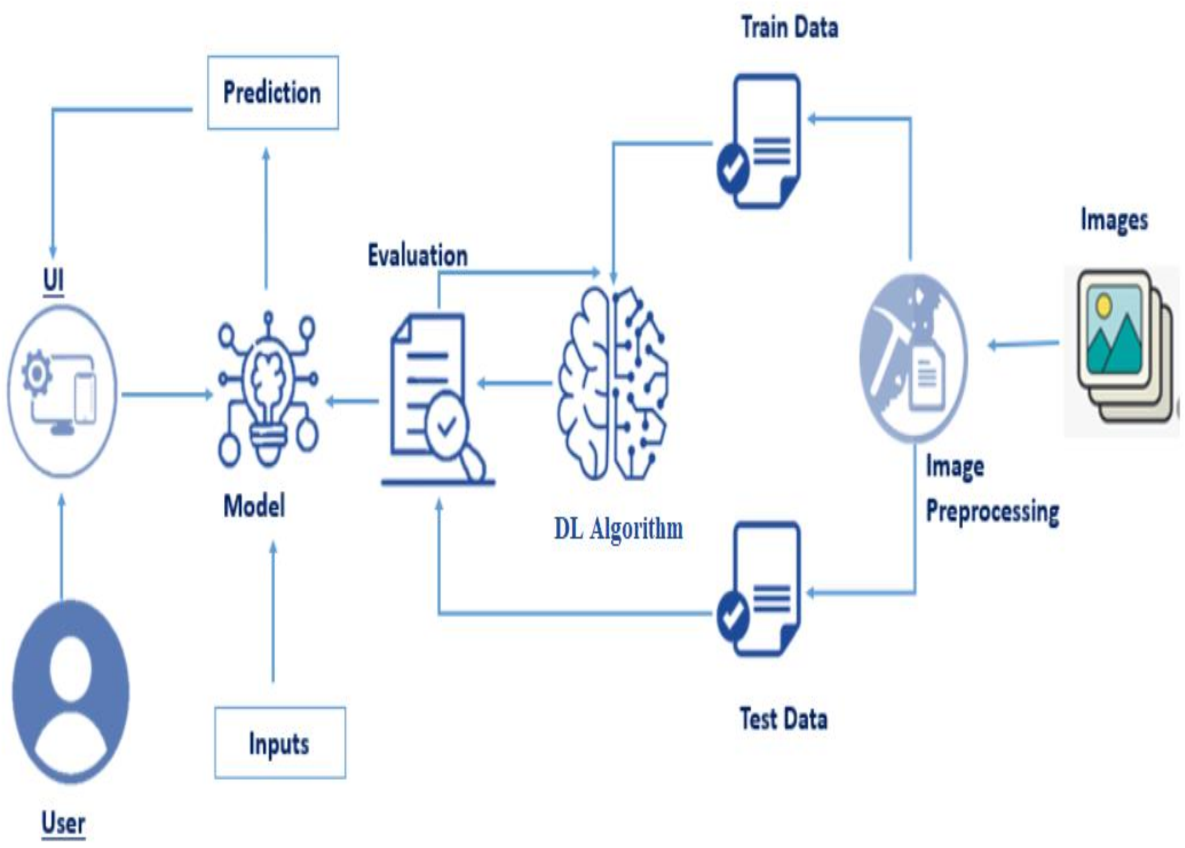
Level 3



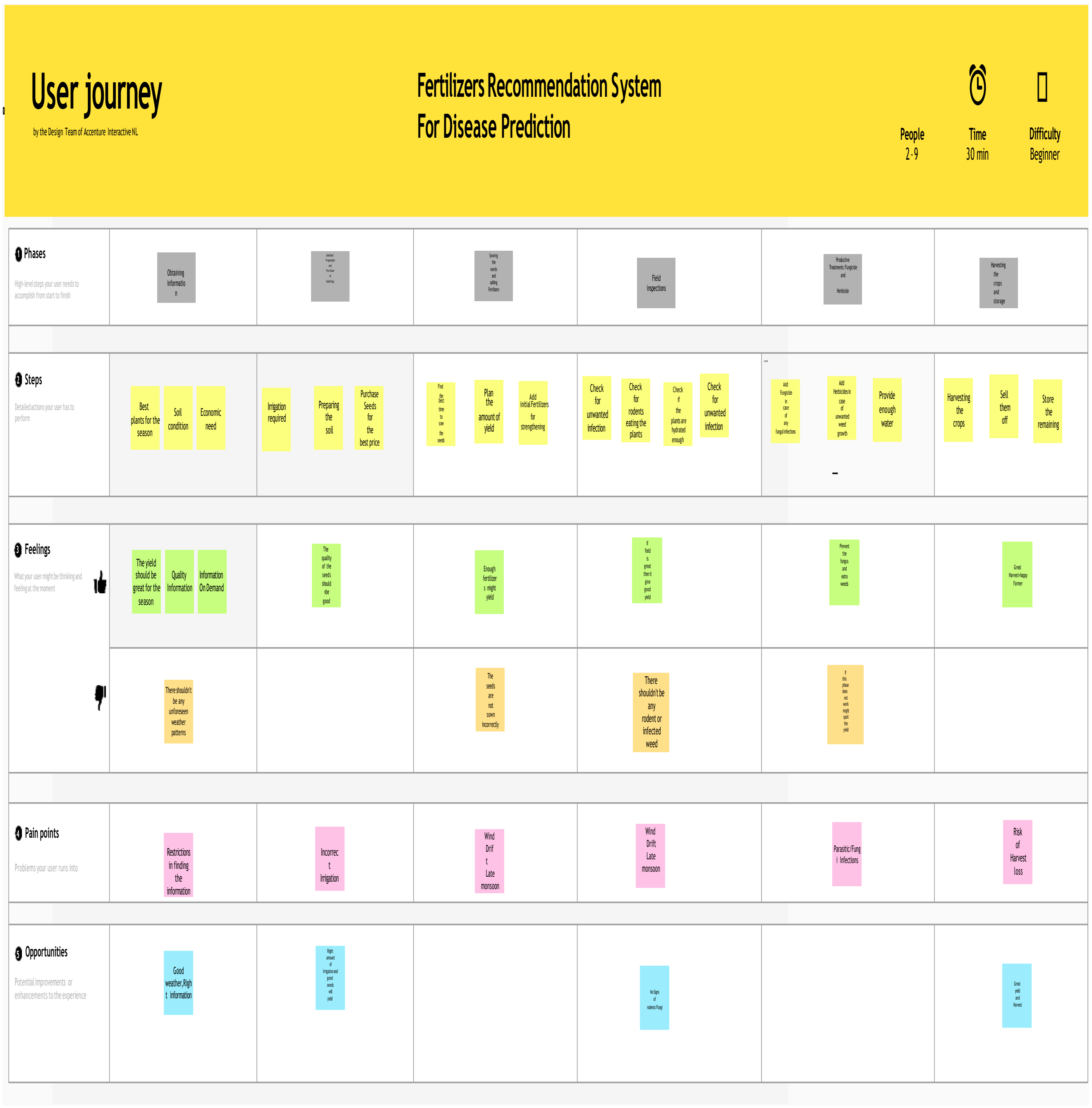
Level 4


Leaf disease
Analyze the disease name
Fertilizer names
Extract the fertilizer name
Provide fertilizer with measurements

5.2 SOLUTION & TECHNICAL ARCHITECTURE :



5.3 USER STORIES :



User journey

by the Design Team of Accenture Interactive NL

Fertilizers Recommendation System For Disease Prediction

People 2-9 | Time 30 min | Difficulty Beginner

1 Phases High-level steps your user needs to accomplish from start to finish	Obtaining informatio n	Seedbed Preparation and Purchase of Seedlings	Sowing the seeds and adding Fertilizers	Field inspections	Productive Treatments: Fungicide and Herbicide	Harvesting the crops and storage
2 Steps Detailed actions your user has to perform	Best plants for the season; Soil condition; Economic need	Irrigation required; Preparing the soil; Purchase Seeds for the best price	Find the best time to sow the seeds; Plan the amount of yield; Add initial Fertilizers for strengthening	Check for unwanted infection; Check for rodents eating the plants; Check if the plants are hydrated enough; Check for unwanted infection	Add Fungicide in case of any fungal infections; Add Herbicides in case of unwanted weed growth; Provide enough water	Harvesting the crops; Sell them off; Store the remaining
3 Feelings What your user might be thinking and feeling at the moment 👍	The yield should be great for the season; Quality Information; Information On Demand	The quality of the seeds should vbe good	Enough fertilizer s might yield	If field is great then it give good yield	Prevent the fungus and extra weeds	Great Harvest=happy Farmer
👎	There shouldn't be any unforeseen weather patterns		The seeds are not sown incorrectly	There shouldn't be any rodent or infected weed	If this phase does not work might spoil the yield	
4 Pain points Problems your user runs into	Restrictions in finding the information	Incorrec t Irrigation	Wind Drif t Late monsoon	Wind Drift Late monsoon	Parasitic/Fung i Infections	Risk of Harvest loss
5 Opportunities Potential improvements or enhancements to the experience	Good weather,Righ t information	Right amount of Irrigation and good seeds will yield		No Signs of rodents/Fungi		Great yield and Harvest

6. PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION :

PROJECT PLANNING PHASE

MILESTONE & ACTIVITY LIST

DATE	9 NOVEMBER 2022
TEAM ID	PNT2022TMID08559
PROJECT NAME	FERTILIZER RECOMMENDATION SYSTEM FOR PLANT DISEASEPREDICTION

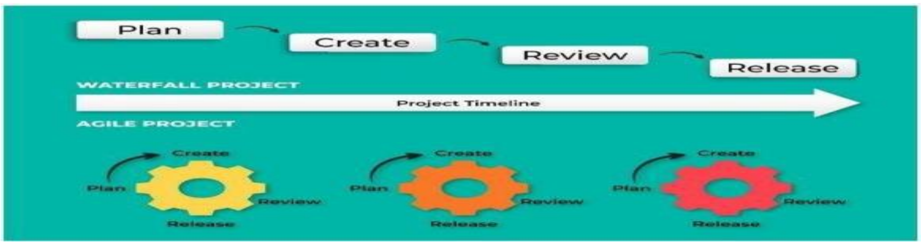
Milestone:

Modern Technology are increasing and optimizing the Performance of the Artificial Intelligences (AI) Model. Based Crop Yield Disease Prediction System, is helpful for farmersto prevent the crop from the various Disease which can identify the Disease with in a processof capturing the Image at the plant and Machine Learning Algorithm will give affected Disease Name. In this Project Milestone will be given the Best Solution for the farmer using the complete friendly and simple user interface web application to fetching the solution by own. In addition, process we are planned 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.

Activity List:

In Project Management Planning is an important task to scheduling the phases of the project to the Team Member. In this Activity can shows the various activity are allocated and doneby the Team Members. The phases are

Phase 1: Literature survey and information gathering
Phase 2: Prepare empathy map
Phase 3: Ideation
Phase 4: Proposed solution
Phase 5: Proposed solution fit
Phase 6: Solution architecture
Phase 7: Customer journey
Phase 8: Functional requirement
Phase 9: Data flow diagrams
Phase 10: Technology architecture
Phase 11: prepare milestone activity and list
Phase 12: Spirit delivery plan
Phase 13: Project development-Delivery


Plan
Create
Review
Release
WATERFALL PROJECT
Project Timeline
AGILE PROJECT
Create
Plan
Review
Release
Create
Plan
Review
Release
Create
Plan
Review
Release

6.2 SPRINT DELIVERY SCHEDULE :

Project Planning Phase Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

DATE	10 NOVEMBER
TEAM ID	PNT2022TMIDO8559
PROJECT TITLE	Fertilizers Recommendation System For Disease Prediction

Product Backlog, Sprint Schedule, and Estimation :

Sprint	Function Requirement (Epic)	User Story · Number	User Story / Task	Story Points (Total)	Priority	Team Members
Sprint-1	Model Creation and Training (Fruits)		Create a model which can classify diseased fruit plants from given images. I also need to test the model and deploy it on IBM Cloud	8	High	Vishwanantha, sathishkumar, deva, jaya raj
	Model Creation and Training (Vegetables)		Create a model which can classify diseased vegetable plants from given images	2	High	Vishwanantha, sathishkumar, deva, jaya raj

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points (Total)	Priorit y	Team Members
Sprint-2	Model Creation and Training (Vegetables)		Create a model which can classify diseased vegetable plants from given images and train on IBM Cloud	6	high	Vishwanantha, sathishkumar, deva, jaya raj
	Registration	USN-1	As a user, I can register by entering my email, password, and confirming my password or via O Auth API	3	high	Vishwanantha, sathishkumar, deva, jaya raj
	Upload page	USN-2	As a user, I will be redirected to a page where I can upload my pictures of crops	4	high	Vishwanantha, sathishkumar, deva, jaya raj
	Suggestion results	USN-3	As a user, I can view the results and then obtain the suggestions provided by the ML mode	4	high	Vishwanantha, sathishkumar, deva, jaya raj
	Base Flask App		A base Flask web app must be created as an interface for the ML model	2	high	Vishwanantha, sathishkumar, deva, jaya raj

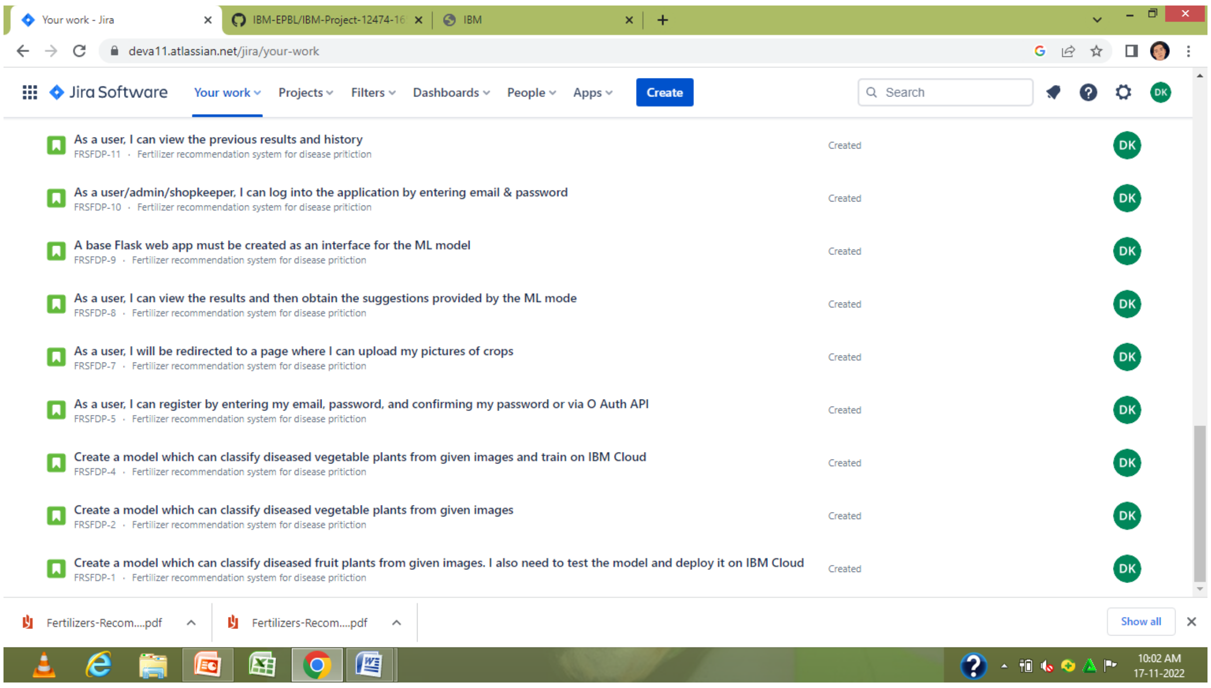
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points (Total)	Priority	Team Members
Sprint-3	Login	USN-4	As a user/admin/shopkeeper, I can log into the application by entering email & password	2	high	Vishwanantha, sathishkumar, deva, jaya raj
	User Dashboard	USN-5	As a user, I can view the previous results and history	3	Medium	Vishwanantha, sathishkumar, deva, jaya raj
	Integration		Integrate Flask, CNN model with Cloud ant DB	5	Medium	Vishwanantha, sathishkumar, deva, jaya raj
	Containerization		Containerize Flask app using Docker	2	low	Vishwanantha, sathishkumar, deva, jaya raj

Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points (Total)**	**Priority**	**Team Members**
Sprint-4	Dashboard (Admin)	USN-6	As an admin, I can view other user details and uploads for other purposes	2	Medium	Vishwanantha, sathishkumar, deva, jaya raj
	Dashboard (Shopkeeper)	USN-7	As a shopkeeper, I can enter fertilizer products and then update the details if any	2	low	Vishwanantha, sathishkumar, deva, jaya raj
	Containerization		Create and deploy Helm charts using Docker Image made before	2	low	Vishwanantha, sathishkumar, deva, jaya raj

Project Tracker, Velocity & Burn down Chart :

Sprint	**Total Story Points**	**Duration**	**Sprint Start Date**	**Sprint End Date (Planned)**	**Story Points Completed (as on Planned End Date)**	**Sprint Release Date (Actual)**
Sprint-1	10	6 Days	24 Oct 2022	29 Oct 2022	10	30 Oct 2022
Sprint-2	15	6 Days	31 Oct 2022	05 Nov 2022	15	06 Nov 2022
Sprint-3	15	6 Days	07 Nov 2022	12 Nov 2022	15	13 Nov 2022
Sprint-4	12	6 Days	14 Nov 2022	19 Nov 2022	10	20 Nov 2022

6.3 REPORTS FROM JIRA :



As a user, I can view the previous results and history
FRSFDP-11 · Fertilizer recommendation system for disease priction — Created — DK

As a user/admin/shopkeeper, I can log into the application by entering email & password
FRSFDP-10 · Fertilizer recommendation system for disease priction — Created — DK

A base Flask web app must be created as an interface for the ML model
FRSFDP-9 · Fertilizer recommendation system for disease priction — Created — DK

As a user, I can view the results and then obtain the suggestions provided by the ML mode
FRSFDP-8 · Fertilizer recommendation system for disease priction — Created — DK

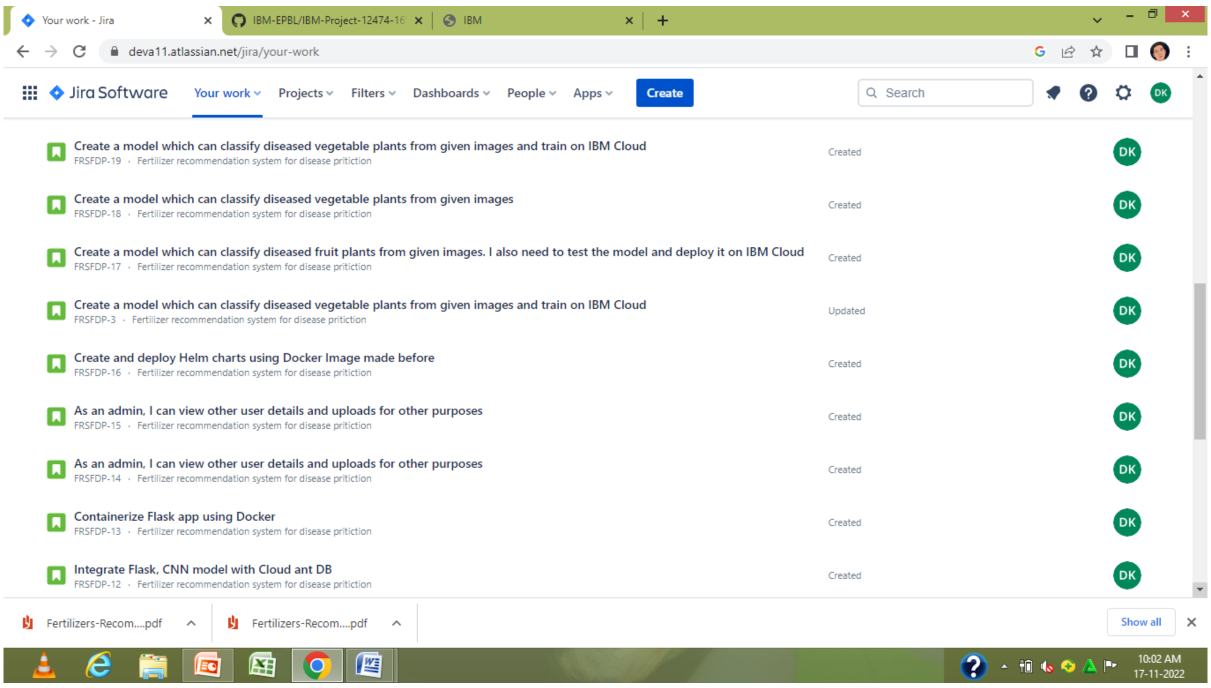
As a user, I will be redirected to a page where I can upload my pictures of crops
FRSFDP-7 · Fertilizer recommendation system for disease priction — Created — DK

As a user, I can register by entering my email, password, and confirming my password or via O Auth API
FRSFDP-5 · Fertilizer recommendation system for disease priction — Created — DK

Create a model which can classify diseased vegetable plants from given images and train on IBM Cloud
FRSFDP-4 · Fertilizer recommendation system for disease priction — Created — DK

Create a model which can classify diseased vegetable plants from given images
FRSFDP-2 · Fertilizer recommendation system for disease priction — Created — DK

Create a model which can classify diseased fruit plants from given images. I also need to test the model and deploy it on IBM Cloud
FRSFDP-1 · Fertilizer recommendation system for disease priction — Created — DK



Create a model which can classify diseased vegetable plants from given images and train on IBM Cloud
FRSFDP-19 · Fertilizer recommendation system for disease priction — Created — DK

Create a model which can classify diseased vegetable plants from given images
FRSFDP-18 · Fertilizer recommendation system for disease priction — Created — DK

Create a model which can classify diseased fruit plants from given images. I also need to test the model and deploy it on IBM Cloud
FRSFDP-17 · Fertilizer recommendation system for disease priction — Created — DK

Create a model which can classify diseased vegetable plants from given images and train on IBM Cloud
FRSFDP-3 · Fertilizer recommendation system for disease priction — Updated — DK

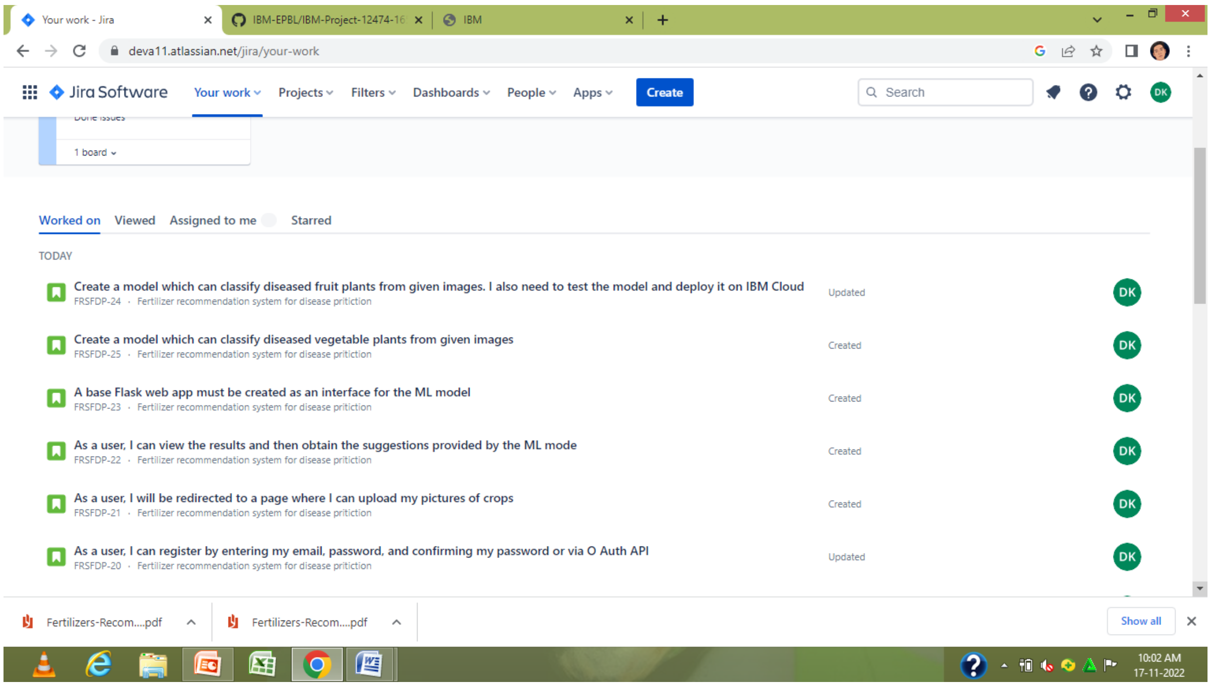
Create and deploy Helm charts using Docker Image made before
FRSFDP-16 · Fertilizer recommendation system for disease priction — Created — DK

As an admin, I can view other user details and uploads for other purposes
FRSFDP-15 · Fertilizer recommendation system for disease priction — Created — DK

As an admin, I can view other user details and uploads for other purposes
FRSFDP-14 · Fertilizer recommendation system for disease priction — Created — DK

Containerize Flask app using Docker
FRSFDP-13 · Fertilizer recommendation system for disease priction — Created — DK

Integrate Flask, CNN model with Cloud ant DB
FRSFDP-12 · Fertilizer recommendation system for disease priction — Created — DK



TODAY

Create a model which can classify diseased fruit plants from given images. I also need to test the model and deploy it on IBM Cloud
FRSFDP-24 · Fertilizer recommendation system for disease prittion — Updated — DK

Create a model which can classify diseased vegetable plants from given images
FRSFDP-25 · Fertilizer recommendation system for disease prittion — Created — DK

A base Flask web app must be created as an interface for the ML model
FRSFDP-23 · Fertilizer recommendation system for disease prittion — Created — DK

As a user, I can view the results and then obtain the suggestions provided by the ML mode
FRSFDP-22 · Fertilizer recommendation system for disease prittion — Created — DK

As a user, I will be redirected to a page where I can upload my pictures of crops
FRSFDP-21 · Fertilizer recommendation system for disease prittion — Created — DK

As a user, I can register by entering my email, password, and confirming my password or via O Auth API
FRSFDP-20 · Fertilizer recommendation system for disease prittion — Updated — DK

7. CODING & SOLUTIONING

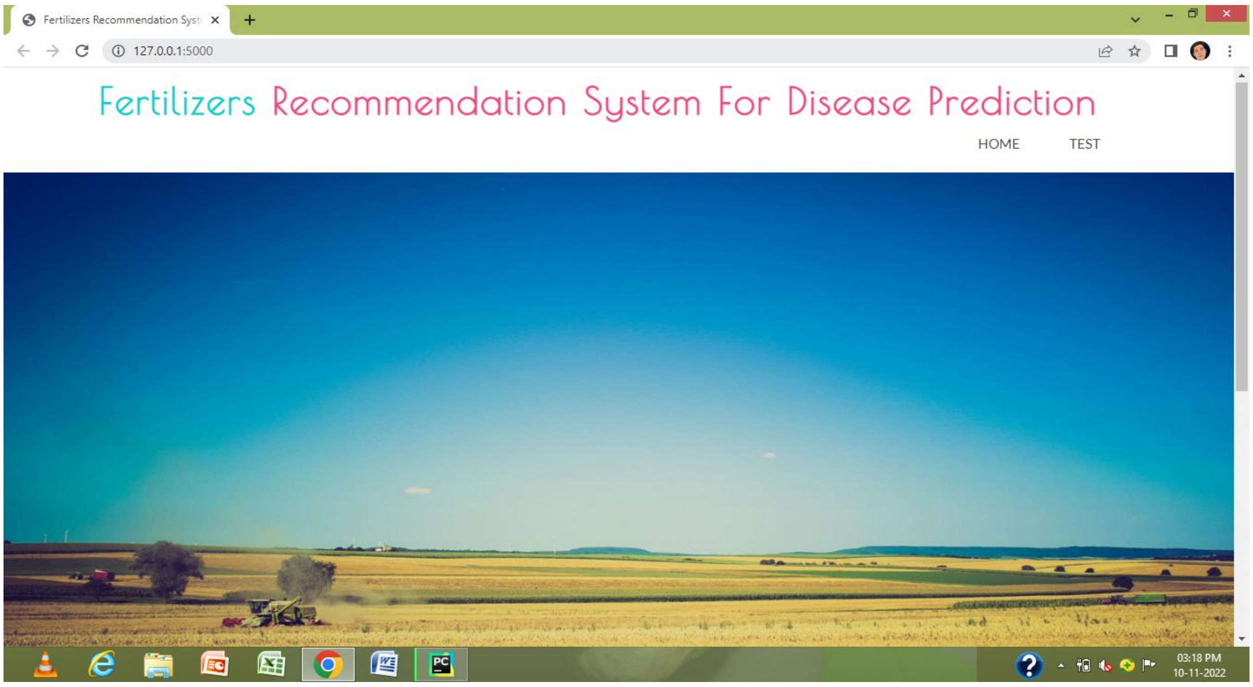
7.1 FEATURE 1 : CODING SAMPLE

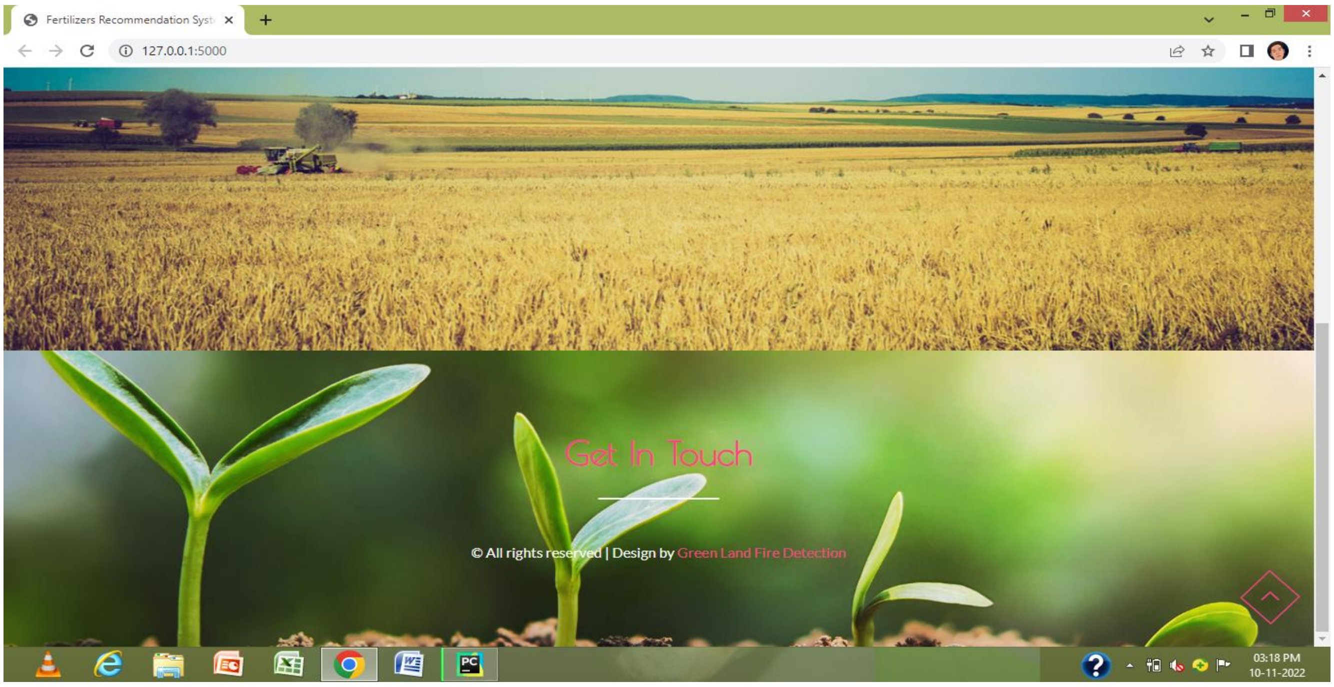
```
mport tensorflow as tf
import numpy as np

from tkinter import *
import os
from tkinter import filedialog
import cv2
import time
from matplotlib import pyplot as plt
from tkinter import messagebox




def endprogram():
   print ("\nProgram terminated!")
   sys.exit()
```

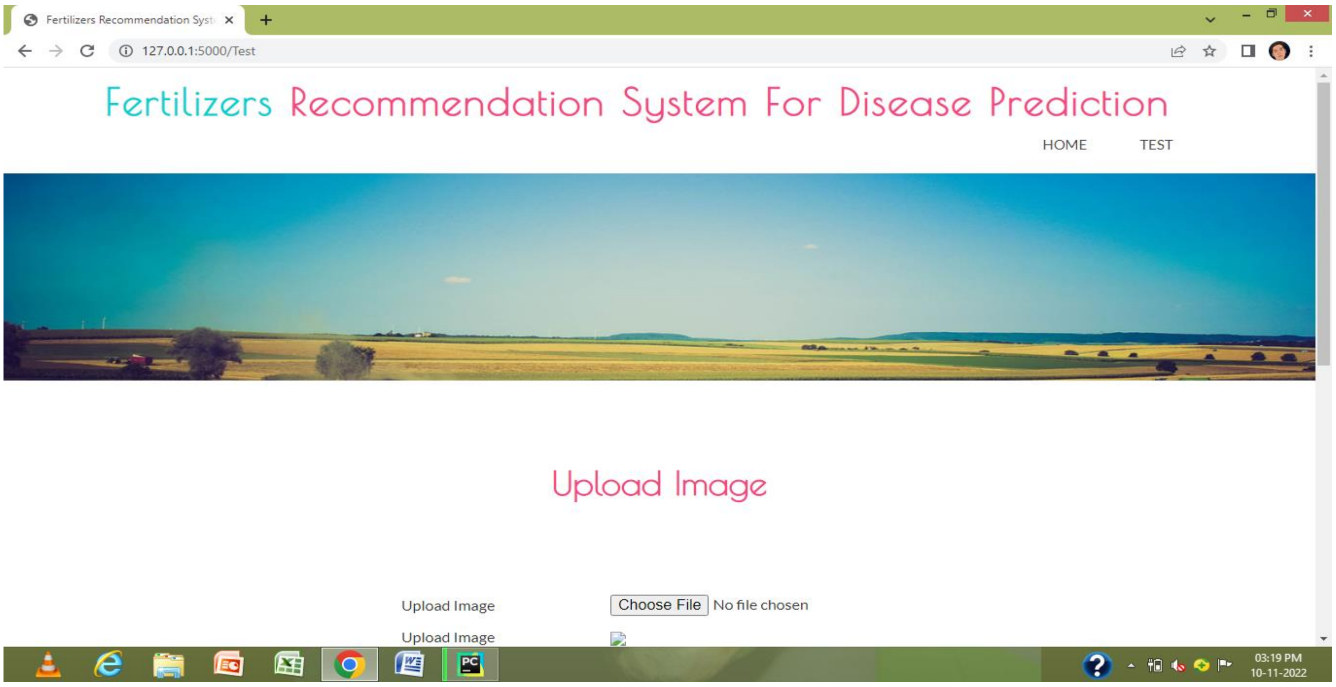
```python
def fulltraining():
    import model as mm





def testing():
    global testing_screen
    testing_screen = Toplevel(main_screen)
    testing_screen.title("Testing")
    # login_screen.geometry("400x300")
    testing_screen.geometry("600x450+650+150")
    testing_screen.minsize(120, 1)
    testing_screen.maxsize(1604, 881)
    testing_screen.resizable(1, 1)
    testing_screen.configure(bg='green')
    # login_screen.title("New Toplevel")

    Label(testing_screen, text='''Upload Image''', disabledforeground="#a3a3a3",
          foreground="#000000", width="300", height="2", font=("Calibri",
16)).pack()
    Label(testing_screen, text="").pack()
    Label(testing_screen, text="").pack()
    Label(testing_screen, text="").pack()
    Button(testing_screen, text='''Upload Image''', font=(
        'Verdana', 15), height="2", width="30", command=imgtest).pack()


global affect
def imgtest():


    import_file_path = filedialog.askopenfilename()

    image = cv2.imread(import_file_path)
    print(import_file_path)
    filename = 'Output/Out/Test.jpg'
    cv2.imwrite(filename, image)
    print("After saving image:")
    #result()

    #import_file_path = filedialog.askopenfilename()
    print(import_file_path)
    fnm = os.path.basename(import_file_path)
    print(os.path.basename(import_file_path))

   # file_sucess()

    print("\n*********************\nImage : " + fnm + "\n*********************")
    img = cv2.imread(import_file_path)
    if img is None:
        print('no data')

    img1 = cv2.imread(import_file_path)
    print(img.shape)
    img = cv2.resize(img, ((int)(img.shape[1] / 5), (int)(img.shape[0] / 5)))
    original = img.copy()
    neworiginal = img.copy()
    cv2.imshow('original', img1)
    gray = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)

    img1S = cv2.resize(img1, (960, 540))

    cv2.imshow('Original image', img1S)
    grayS = cv2.resize(gray, (960, 540))
    cv2.imshow('Gray image', grayS)

    dst = cv2.fastNlMeansDenoisingColored(img1, None, 10, 10, 7, 21)
    cv2.imshow("Nosie Removal", dst)

    thresh = 127
    im_bw = cv2.threshold(grayS, thresh, 255, cv2.THRESH_BINARY)[1]
    #cv2.imshow("affect Removal", im_bw)
    number_of_black_pix = np.sum(im_bw == 0)
    #print(number_of_black_pix)
    #if(number_of_black_pix<5000):
        #affect =


    result()








def result():
    import warnings
    warnings.filterwarnings('ignore')

    import tensorflow as tf
    classifierLoad = tf.keras.models.load_model('firemodel.h5')

    import numpy as np
    from keras.preprocessing import image

    test_image = image.load_img('Output/Out/Test.jpg', target_size=(200, 200))
    img1 = cv2.imread('Output/Out/Test.jpg')
    # test_image = image.img_to_array(test_image)
    test_image = np.expand_dims(test_image, axis=0)
    result = classifierLoad.predict(test_image)

    out = ''
    pre=''
    if result[0][0] == 1:

        out="Fire"

    elif result[0][1] == 1:

        out="Nofire"




    messagebox.showinfo("Result", "Classfication Rssult : "+str(out))








def main_account_screen():
    global main_screen
    main_screen = Tk()
    width = 600
    height = 600
    screen_width = main_screen.winfo_screenwidth()
    screen_height = main_screen.winfo_screenheight()
    x = (screen_width / 2) - (width / 2)
    y = (screen_height / 2) - (height / 2)
    main_screen.geometry("%dx%d+%d+%d" % (width, height, x, y))
    main_screen.resizable(0, 0)
    # main_screen.geometry("300x250")
    main_screen.configure(bg='green')
    main_screen.title("Forest Fire Detection  ")

    Label(text="Forest Fire Detection", width="300", height="5", bg='green',
font=("Calibri", 16)).pack()



    Button(text="Training", font=(
        'Verdana', 15), height="2", width="30", bg='green',command=fulltraining,
highlightcolor="black").pack(side=TOP)
    Label(text="").pack()

    Button(text="Testing", font=(
        'Verdana', 15), height="2", width="30",bg='green',
command=testing).pack(side=TOP)

    Label(text="").pack()

    main_screen.mainloop()
```

```
main_account_screen()
```

7.2 FEATURE 2 : OUTPUT SAMPLE


Fertilizers Recommendation Syst
127.0.0.1:5000
Fertilizers Recommendation System For Disease Prediction
HOME
TEST
03:18 PM
10-11-2022


Fertilizers Recommendation Syst
127.0.0.1:5000
Get In Touch
© All rights reserved | Design by Green Land Fire Detection
03:18 PM
10-11-2022



Fertilizers Recommendation Syst

127.0.0.1:5000/Test

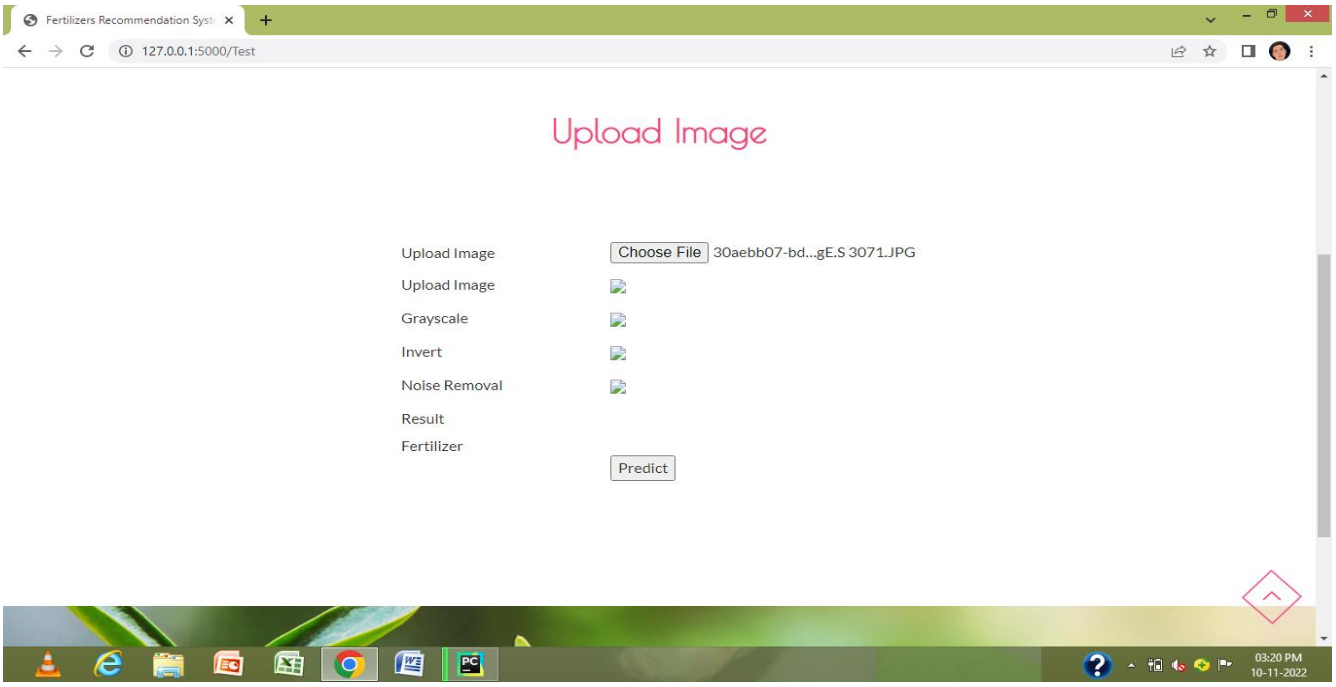
Fertilizers Recommendation System For Disease Prediction

HOME TEST

Upload Image

Upload Image | Choose File No file chosen

Upload Image



Fertilizers Recommendation Syst

127.0.0.1:5000/Test

Upload Image

Upload Image | Choose File 30aebb07-bd...gE.S 3071.JPG

Upload Image

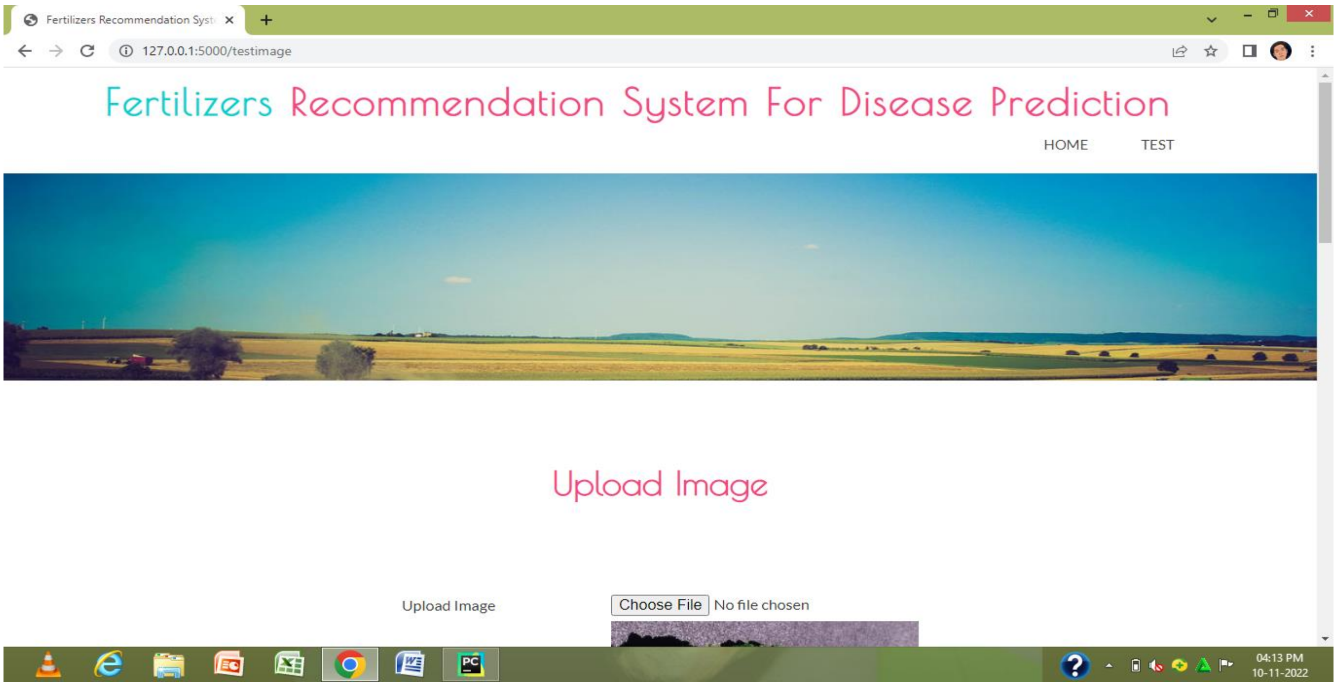
Grayscale

Invert

Noise Removal

Result

Fertilizer



Fertilizers Recommendation Syst

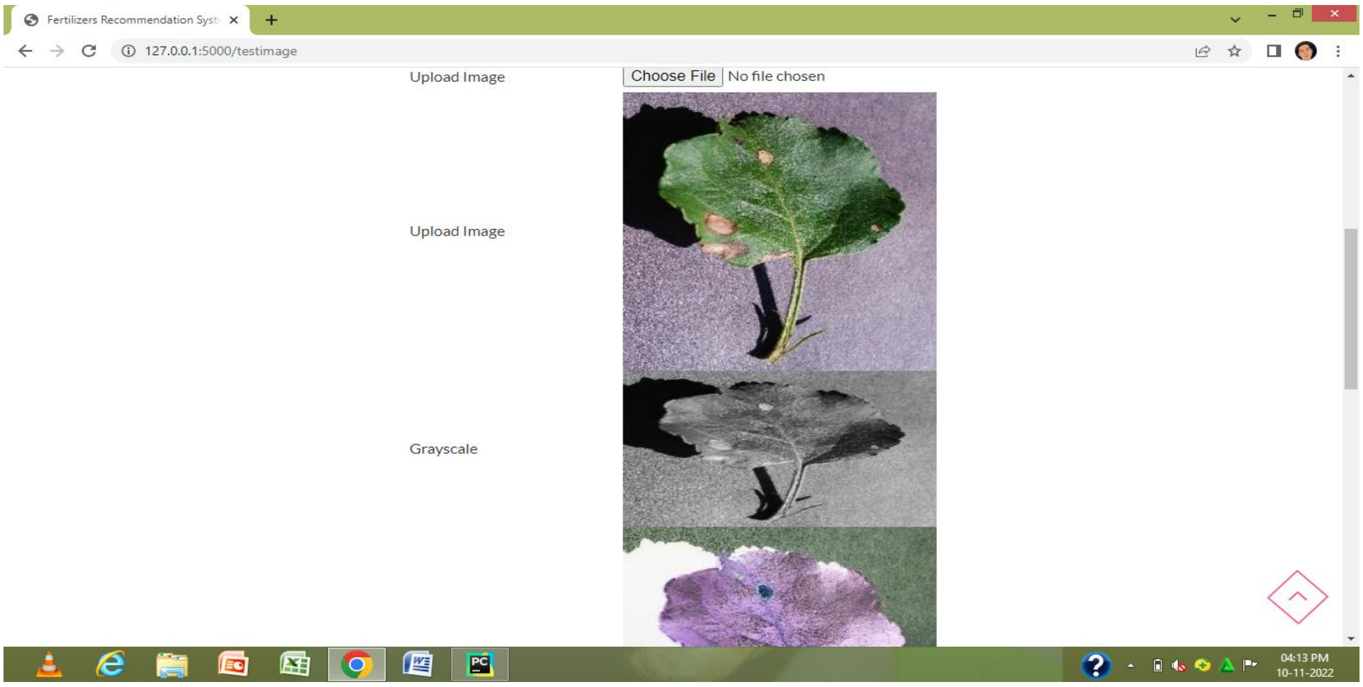
127.0.0.1:5000/testimage

Fertilizers Recommendation System For Disease Prediction

HOME TEST

Upload Image

Upload Image Choose File No file chosen



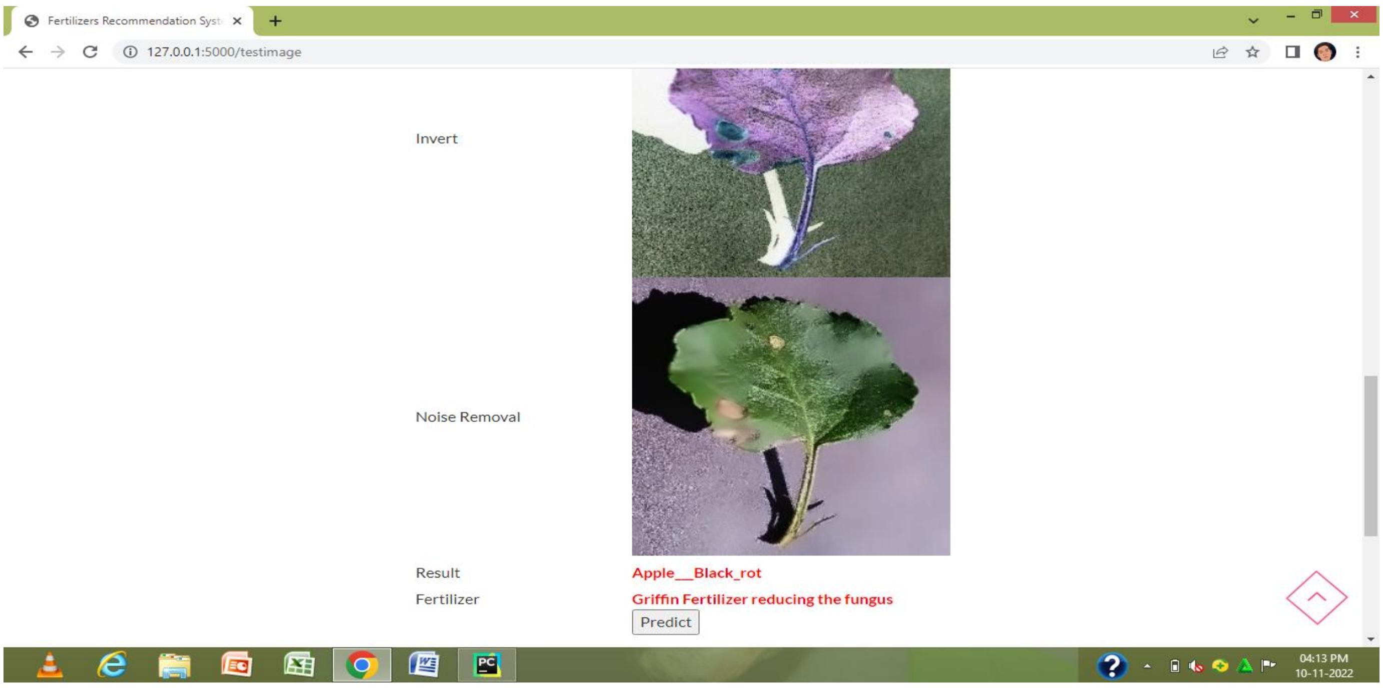
Fertilizers Recommendation Syst

127.0.0.1:5000/testimage

Upload Image Choose File No file chosen

Upload Image

Grayscale



Result Apple__Black_rot

Fertilizer Griffin Fertilizer reducing the fungus

8. TESTING

8.1 TEST CASE :

8.1 TEST CASES

A test case has components that describe input, action and an expected response, in order to determine if a feature of an application is working correctly. A test case is a set of instructions on "HOW" to validate a particular test objective/target, which when followed will tell us if the expected behavior of the system is satisfied or not.

Characteristics of a good test case:

- Accurate: Exacts the purpose.

- Economical: No unnecessary steps or words.
- Traceable: Capable of being traced to requirements.
- Repeatable: Can be used to perform the test over and over.

S.NO	FUNCTION	DESCRIPTION	EXPECTED OUTPUT	ACTUAL OUTPUT	STATUS
1	Framework construction	Generate the GUI for admin and user	Individual page for admin and user	Individual page for admin and user	Success
2	Read the comments	Comments analysis	Comments in text format	Comments in text format	Success
3	Classification	Classify the datasets	Negative comments	Negative comments	Success
4	Rules implementation	Block the comments and friends	Block the users	Block the users	Success

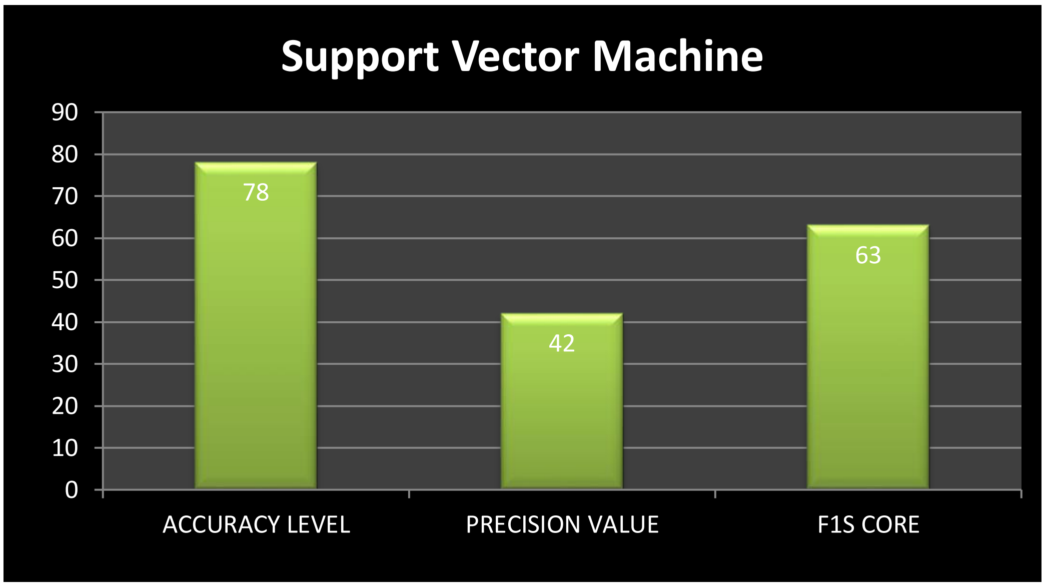
8.2 USER ACCEPTANCE TESTING :

Acceptance testing can be defined in many ways, but a simple definition is the succeeds when the software functions in a manner that can be reasonable expected by the customer. After the acceptance test has been conducted, one of the two possible conditions exists. This is to fine whether the inputs are accepted by the database or other validations. For example accept only numbers in the numeric field, date format data in the date field. Also the null check for the not null fields. If any error occurs then show the error messages. The function of performance characteristics to specification and is accepted. A deviation from specification is uncovered and a deficiency list is created. User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

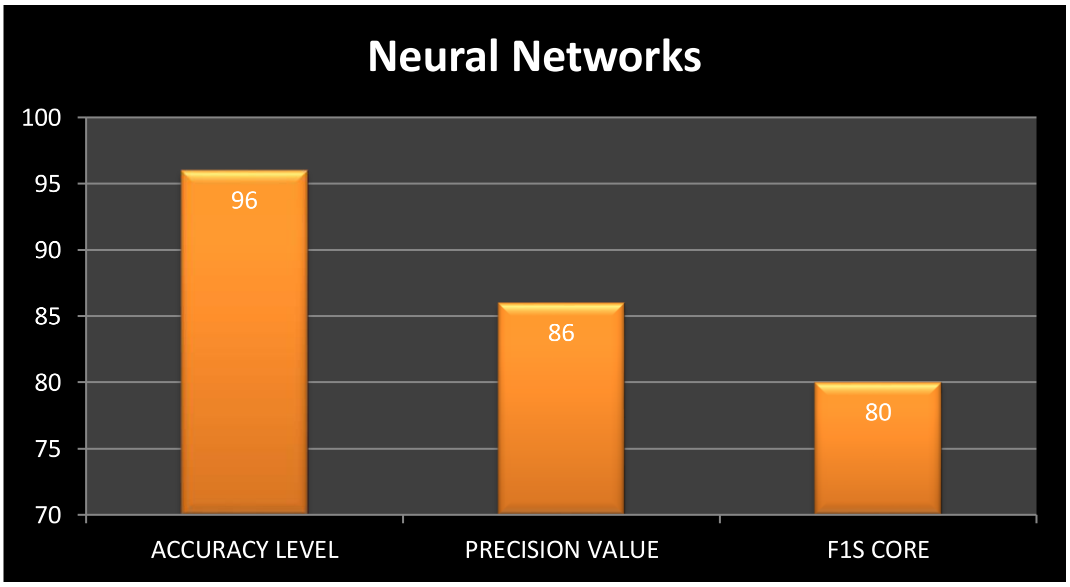
9. RESULT

9.1 PERFORMANCE METRICS

Existing Algorithm



Proposed Algorithm



10. ADVANTAGES AND DISADVANTAGES

DISADVANTAGES

- Suffer in the local minima problems.
- Dimensionality is high to produce large number of irrelevant features.
- User defined segmentation can be done.
- Does not recommend the fertilizers to leaves diseases.
- Manual approach is used

ADVANTAGES

- Segmentation can be done easily to spilt the tree parts.
- Classify the affected parts in leaves.
- Eliminate redundant features of images.
- Provide improved accuracy rate

11. CONCLUSION

CONCLUSION

We presented a machine learning approach for crop yield prediction, which demonstrated superior performance in Crop Challenge using large datasets of products. The approach used deep neural networks to make yield predictions (including yield, check yield, and yield difference) based on genotype and environment data. The carefully designed deep neural networks were able to learn nonlinear and complex relationships between genes, environmental conditions, as well as their interactions from historical data and make reasonably accurate predictions of yields for new hybrids planted in new locations with known weather conditions. Performance of the model was found to be relatively sensitive to the quality of weather prediction, which suggested the importance of weather prediction techniques. We trained two deep neural networks, one for yield and the other for check yield, and then used the difference of their outputs as the prediction for yield difference. This model structure was found to be more effective than using one single neural network for yield difference, because the genotype and environment effects are more directly related to the yield and check yield than their difference. In modern era, the deep neural network is the prominent tool in agricultural industry for providing support to farmers in monitoring crop yield based on multiple parameters. Thus, the machine learning model provides high accuracy in detecting the suitable crop identification compared to other methodologies.

12. FUTURE SCOPE

FUTURE WORK

This project describes crop yield prediction ability of the algorithm. In future we can determine the efficient algorithm based on their accuracy metrics that will helps to choose an efficient algorithm for crop yield prediction

13. APPENDIX

SOURCE CODE :

```
import tensorflow as tf
import time
import numpy as np
import os

start = time.time()
#try:
# Total iterations
final_iter = 1000

# Assign the batch value
batch_size = 20

# 20% of the data will automatically be used for validation
validation_size = 0.2
img_size = 128
num_channels = 3
```

```
train_path = r'data\Train'

# Prepare input data
if not os.path.exists(train_path):
print("No such directory")
raise Exception
classes = os.listdir(train_path)
num_classes = len(classes)

# We shall load all the training and validation images and labels into memory
using openCV and use that during training
data = dataset.read_train_sets(train_path, img_size, classes,
validation_size=validation_size)

# Display the stats
print("Complete reading input data. Will Now print a snippet of it")
print("Number of files in Training-set:\t\t{}".format(len(data.train.labels)))
print("Number of files in Validation-set:\t{}".format(len(data.valid.labels)))
session = tf.compat.v1.Session()
x = tf.compat.v1.placeholder(tf.float32, shape=[None, img_size, img_size,
num_channels], name='x')

## labels
y_true = tf.compat.v1.placeholder(tf.float32, shape=[None, num_classes],
name='y_true')
y_true_cls = tf.argmax(y_true, dimension=1)

##Network graph params
filter_size_conv1 = 3
num_filters_conv1 = 32

filter_size_conv2 = 3
num_filters_conv2 = 32

filter_size_conv3 = 3
num_filters_conv3 = 64

fc_layer_size = 128


def create_weights(shape):
return tf.Variable(tf.random.truncated_normal(shape, stddev=0.05))


def create_biases(size):
return tf.Variable(tf.constant(0.05, shape=[size]))



def make_generator_model(input,
                               num_input_channels,
                               conv_filter_size,
                               num_filters):
## We shall define the weights that will be trained using create_weights function.
weights = create_weights(shape=[conv_filter_size, conv_filter_size,
num_input_channels, num_filters])
## We create biases using the create_biases function. These are also trained.
biases = create_biases(num_filters)
```

```
## Creating the convolutional layer
layer = tf.nn.conv2d(input=input,
filter=weights,
strides=[1, 1, 1, 1],
padding='SAME')

    layer += biases

## We shall be using max-pooling.
layer = tf.nn.max_pool(value=layer,
ksize=[1, 2, 2, 1],
strides=[1, 2, 2, 1],
padding='SAME')
## Output of pooling is fed to Relu which is the activation function for us.
layer = tf.nn.relu(layer)

return layer


# Function to create a Flatten Layer
def create_flatten_layer(layer):
# We know that the shape of the layer will be [batch_size img_size img_size
num_channels]
    # But let's get it from the previous layer.
layer_shape = layer.get_shape()

## Number of features will be img_height * img_width* num_channels. But we shall
calculate it in place of hard-coding it.
num_features = layer_shape[1:4].num_elements()

## Now, we Flatten the layer so we shall have to reshape to num_features
layer = tf.reshape(layer, [-1, num_features])

return layer


# Function to create a Fully - Connected Layer
def create_fc_layer(input,
                    num_inputs,
                    num_outputs,
                    use_relu=True):
# Let's define trainable weights and biases.
weights = create_weights(shape=[num_inputs, num_outputs])
    biases = create_biases(num_outputs)

# Fully connected layer takes input x and produces wx+b.Since, these are matrices,
we use matmul function in Tensorflow
layer = tf.matmul(input, weights) + biases
if use_relu:
        layer = tf.nn.relu(layer)

return layer


# Create all the layers
layer_conv1 = make_generator_model(input=x,
num_input_channels=num_channels,
conv_filter_size=filter_size_conv1,
num_filters=num_filters_conv1)
```

```
layer_conv2 = make_generator_model(input=layer_conv1,
num_input_channels=num_filters_conv1,
conv_filter_size=filter_size_conv2,
num_filters=num_filters_conv2)

layer_conv3 = make_generator_model(input=layer_conv2,
num_input_channels=num_filters_conv2,
conv_filter_size=filter_size_conv3,
num_filters=num_filters_conv3)

layer_flat = create_flatten_layer(layer_conv3)

layer_fc1 = create_fc_layer(input=layer_flat,
num_inputs=layer_flat.get_shape()[1:4].num_elements(),
num_outputs=fc_layer_size,
use_relu=True)

layer_fc2 = create_fc_layer(input=layer_fc1,
num_inputs=fc_layer_size,
num_outputs=num_classes,
use_relu=False)

y_pred = tf.nn.softmax(layer_fc2, name='y_pred')

y_pred_cls = tf.argmax(y_pred, dimension=1)
session.run(tf.compat.v1.global_variables_initializer())
cross_entropy = tf.nn.softmax_cross_entropy_with_logits_v2(logits=layer_fc2,
labels=y_true)
cost = tf.reduce_mean(cross_entropy)
optimizer = tf.compat.v1.train.AdamOptimizer(learning_rate=1e-4).minimize(cost)
correct_prediction = tf.equal(y_pred_cls, y_true_cls)
accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))

session.run(tf.compat.v1.global_variables_initializer())


# Display all stats for every epoch
def show_progress(epoch, feed_dict_train, feed_dict_validate, val_loss,
total_epochs):
    acc = session.run(accuracy, feed_dict=feed_dict_train)
    val_acc = session.run(accuracy, feed_dict=feed_dict_validate)
    msg = "Training Epoch {0}/{4} --- Training Accuracy: {1:>6.1%}, Validation
Accuracy: {2:>6.1%},  Validation Loss: {3:.3f}"
print(msg.format(epoch + 1, acc, val_acc, val_loss, total_epochs))


total_iterations = 0

saver = tf.compat.v1.train.Saver()

print("")


# Training Function
def train(num_iteration):
global total_iterations

for i in range(total_iterations,
                   total_iterations + num_iteration):
```

```
        x_batch, y_true_batch, _, cls_batch = data.train.next_batch(batch_size)
        x_valid_batch, y_valid_batch, _, valid_cls_batch =
data.valid.next_batch(batch_size)

        feed_dict_tr = {x: x_batch,
                        y_true: y_true_batch}
        feed_dict_val = {x: x_valid_batch,
                         y_true: y_valid_batch}

        session.run(optimizer, feed_dict=feed_dict_tr)

if i % int(data.train.num_examples / batch_size) == 0:
            val_loss = session.run(cost, feed_dict=feed_dict_val)
            epoch = int(i / int(data.train.num_examples / batch_size))
# print(data.train.num_examples)
            # print(batch_size)
            # print(int(data.train.num_examples/batch_size))
            # print(i)

total_epochs = int(num_iteration / int(data.train.num_examples / batch_size)) + 1
show_progress(epoch, feed_dict_tr, feed_dict_val, val_loss, total_epochs)
            saver.save(session, 'trained_model')

    total_iterations += num_iteration


train(num_iteration=final_iter)


#except Exception as e:
    #print("Exception:",e)

# Calculate execution time
end = time.time()
dur = end-start
print("")
if dur<60:
print("Execution Time:",dur,"seconds")
elif dur>60 and dur<3600:
    dur=dur/60
print("Execution Time:",dur,"minutes")
else:
    dur=dur/(60*60)
print("Execution Time:",dur,"hours")
from flask import Flask, render_template, flash, request, session,send_file
from flask import render_template, redirect, url_for, request
import warnings
import datetime
import cv2
import tensorflow as tf
import numpy as np

from tkinter import *
import os


app = Flask(__name__)
app.config['DEBUG']
app.config['SECRET_KEY'] = '7d441f27d441f27567d441f2b6176a'

@app.route("/")
def homepage():

return render_template('index.html')





@app.route("/Test")
def Test():
return render_template('Test.html')




@app.route("/train", methods=['GET', 'POST'])
def train():
if request.method == 'POST':
import model as model

return render_template('Tranning.html')





@app.route("/testimage", methods=['GET', 'POST'])
def testimage():
if request.method == 'POST':


        file = request.files['fileupload']
        file.save('data/alien_test/Test.jpg')


img = cv2.imread('data/alien_test/Test.jpg')



        train_path = r'data\train'
if not os.path.exists(train_path):
print("No such directory")
raise Exception
# Path of testing images
dir_path = r'data\alien_test'
if not os.path.exists(dir_path):
print("No such directory")
raise Exception

# Walk though all testing images one by one
for root, dirs, files in os.walk(dir_path):
for name in files:

print("")
                image_path = name
                filename = dir_path + '\\' + image_path
```

```
print(filename)
                image_size = 128
num_channels = 3
images = []

if os.path.exists(filename):

# Reading the image using OpenCV
image1 = cv2.imread(filename)

                    import_file_path = filename

                    image = cv2.imread(import_file_path)
                    fnm = os.path.basename(import_file_path)
                    filename = 'Test.jpg'
cv2.imwrite(filename, image)
# print("After saving image:")

print("\n********************\nImage : " + fnm + "\n********************")
                    img = cv2.imread(import_file_path)
if img is None:
print('no data')

                    img1 = cv2.imread(import_file_path)
print(img.shape)
                    img = cv2.resize(img, ((int)(img.shape[1] / 5),
(int)(img.shape[0] / 5)))
                    original = img.copy()
                    neworiginal = img.copy()
                    cv2.imshow('original', img1)
                    gray = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)

                    cv2.imshow('Original image', img1)
                    orimage = 'static/Out/Test.jpg'
cv2.imwrite(orimage, img1)


                    cv2.imshow('Gray image', gray)

                    gry = 'static/Out/gry.jpg'

cv2.imwrite(gry, gray)


                    p = 0
for i in range(img.shape[0]):

for j in range(img.shape[1]):
                            B = img[i][j][0]
                            G = img[i][j][1]
                            R = img[i][j][2]
if (B >110 and G >110 and R >110):
                                p += 1

totalpixels = img.shape[0] * img.shape[1]
                    per_white = 100 * p / totalpixels
if per_white >10:
                        img[i][j] = [500, 300, 200]
                        cv2.imshow('color change', img)
```

```
# Guassian blur
blur1 = cv2.GaussianBlur(img, (3, 3), 1)
# mean-shift algo
newimg = np.zeros((img.shape[0], img.shape[1], 3), np.uint8)
                    criteria = (cv2.TERM_CRITERIA_EPS +
cv2.TERM_CRITERIA_MAX_ITER, 10, 1.0)
                    img = cv2.pyrMeanShiftFiltering(blur1, 20, 30, newimg, 0,
criteria)
                    cv2.imshow('means shift image', img)

                    noise = 'static/Out/noise.jpg'

cv2.imwrite(noise, img)


# Guassian blur
blur = cv2.GaussianBlur(img, (11, 11), 1)

                    blur = cv2.GaussianBlur(img, (11, 11), 1)
# Canny-edge detection
canny = cv2.Canny(blur, 160, 290)
                    canny = cv2.cvtColor(canny, cv2.COLOR_GRAY2BGR)
# contour to find leafs
bordered = cv2.cvtColor(canny, cv2.COLOR_BGR2GRAY)
                    contours, hierarchy = cv2.findContours(bordered,
cv2.RETR_TREE, cv2.CHAIN_APPROX_NONE)
                    maxC = 0
for x in range(len(contours)):
if len(contours[x]) > maxC:
                            maxC = len(contours[x])
                            maxid = x
perimeter = cv2.arcLength(contours[maxid], True)
# print perimeter
Tarea = cv2.contourArea(contours[maxid])
                    cv2.drawContours(neworiginal, contours[maxid], -1, (0, 0,
255))
                    cv2.imshow('Contour', neworiginal)
# cv2.imwrite('Contour complete leaf.jpg',neworiginal)
                    # Creating rectangular roi around contour
height, width, _ = canny.shape
                    min_x, min_y = width, height
                    max_x = max_y = 0
frame = canny.copy()
# computes the bounding box for the contour, and draws it on the frame,
for contour, hier in zip(contours, hierarchy):
                        (x, y, w, h) = cv2.boundingRect(contours[maxid])
                        min_x, max_x = min(x, min_x), max(x + w, max_x)
                        min_y, max_y = min(y, min_y), max(y + h, max_y)
if w >80 and h >80:
# cv2.rectangle(frame, (x,y), (x+w,y+h), (255, 0, 0), 2)   #we do not draw the
rectangle as it interferes with contour later on
roi = img[y:y + h, x:x + w]
                            originalroi = original[y:y + h, x:x + w]
if (max_x - min_x >0 and max_y - min_y >0):
                        roi = img[min_y:max_y, min_x:max_x]
                        originalroi = original[min_y:max_y, min_x:max_x]
                        cv2.rectangle(frame, (min_x, min_y), (max_x, max_y), (255,
0, 0),
2)  # we do not draw the rectangle as it interferes with contour
```

```
cv2.imshow('ROI', frame)

                    roi12 = 'static/Out/roi.jpg'

cv2.imwrite(roi12, frame)



                    cv2.imshow('rectangle ROI', roi)
img = roi
# Changing colour-space
                    # imghsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
imghls = cv2.cvtColor(roi, cv2.COLOR_BGR2HLS)
                    cv2.imshow('HLS', imghls)
                    imghls[np.where((imghls == [30, 200, 2]).all(axis=2))] = [0,
200, 0]
                    cv2.imshow('new HLS', imghls)
# Only hue channel
huehls = imghls[:, :, 0]
                    cv2.imshow('img_hue hls', huehls)
# ret, huehls = cv2.threshold(huehls,2,255,cv2.THRESH_BINARY)
huehls[np.where(huehls == [0])] = [35]
                    cv2.imshow('img_hue with my mask', huehls)
# Thresholding on hue image
ret, thresh = cv2.threshold(huehls, 28, 255, cv2.THRESH_BINARY_INV)
                    cv2.imshow('thresh', thresh)
# Masking thresholded image from original image
mask = cv2.bitwise_and(originalroi, originalroi, mask=thresh)
                    cv2.imshow('masked out img', mask)

# Resizing the image to our desired size and preprocessing will be done exactly as
done during training
image = cv2.resize(image1, (image_size, image_size), 0, 0, cv2.INTER_LINEAR)
                    images.append(image)
                    images = np.array(images, dtype=np.uint8)
                    images = images.astype('float32')
                    images = np.multiply(images, 1.0 / 255.0)

# The input to the network is of shape [None image_size image_size num_channels].
Hence we reshape.
x_batch = images.reshape(1, image_size, image_size, num_channels)

# Let us restore the saved model
sess = tf.compat.v1.Session()
# Step-1: Recreate the network graph. At this step only graph is created.
saver = tf.compat.v1.train.import_meta_graph('models/trained_model.meta')
# Step-2: Now let's load the weights saved using the restore method.
saver.restore(sess, tf.train.latest_checkpoint('./models/'))

# Accessing the default graph which we have restored
graph = tf.compat.v1.get_default_graph()

# Now, let's get hold of the op that we can be processed to get the output.
                    # In the original network y_pred is the tensor that is the
prediction of the network
y_pred = graph.get_tensor_by_name("y_pred:0")

## Let's feed the images to the input placeholders
x = graph.get_tensor_by_name("x:0")
```

```
                    y_true = graph.get_tensor_by_name("y_true:0")
                    y_test_images = np.zeros((1, len(os.listdir(train_path))))

# Creating the feed_dict that is required to be fed to calculate y_pred
feed_dict_testing = {x: x_batch, y_true: y_test_images}
                    result = sess.run(y_pred, feed_dict=feed_dict_testing)
# Result is of this format [[probabiliy_of_classA probability_of_classB ....]]
print(result)

# Convert np.array to list
a = result[0].tolist()
                    r = 0

# Finding the maximum of all outputs
max1 = max(a)
                    index1 = a.index(max1)
                    predicted_class = None

# Walk through directory to find the label of the predicted output
count = 0
for root, dirs, files in os.walk(train_path):
for name in dirs:
if count == index1:
                                predicted_class = name
                            count += 1

# If the maximum confidence output is largest of all by a big margin then
                    # print the class or else print a warning
for i in a:
if i != max1:
if max1 - i < i:
                                r = 1

out = ''

pre = ""
if r == 0:
print(predicted_class)

if (predicted_class == "Black spot"):
                            out = predicted_class

                            pre = 'Griffin  Fertilizer  reducing the fungus'

elif (predicted_class == "canker"):
                            out = predicted_class
                            pre = 'sprayed with Bordeaux mixture 1.0 per cent.'


elif (predicted_class == "greening"):
                            out =  predicted_class
                            pre =  'Mn-Zn-Fe-B micronutrient fertilizer'

elif (predicted_class == "healthy"):
                            out =  predicted_class
# messagebox.showinfo("Uses", '')
elif (predicted_class == "Melanose"):
                            out = predicted_class
                            pre =  'strobilurin fungicide'
```

```
else:

                        out = 'Could not classify with definite confidence'


else:
print("File does not exist")







        org = 'static/Out/Test.jpg'
gry ='static/Out/gry.jpg'
noise = 'static/Out/noise.jpg'
roi12 = 'static/Out/roi.jpg'





return
render_template('Test.html',result=out,org=org,gry=gry,inv=noise,noi=roi12,fer=pre
)



def sendmsg(targetno,message):
import requests

requests.post("http://smsserver9.creativepoint.in/api.php?username=fantasy&passwor
d=596692&to=" + targetno + "&from=FSSMSS&message=Dear user  your msg is " +
message + " Sent By FSMSG
FSSMSS&PEID=1501563800000030506&templateid=1507162882948811640")










if __name__ == '__main__':
    app.run(debug=True, use_reloader=True)
import cv2
import os
import glob
from sklearn.utils import shuffle
import numpy as np
```

```
def load_train(train_path, image_size, classes):
    images = []
    labels = []
    img_names = []
    cls = []

print('Going to read training images')
for fields in classes:
        index = classes.index(fields)
print('Now going to read {} files (Index: {})'.format(fields, index))
        path = os.path.join(train_path, fields, '*g')
        files = glob.glob(path)
for fl in files:
            image = cv2.imread(fl)
            image = cv2.resize(image, (image_size, image_size),0,0,
cv2.INTER_LINEAR)
            image = image.astype(np.float32)
            image = np.multiply(image, 1.0 / 255.0)
            images.append(image)
            label = np.zeros(len(classes))
            label[index] = 1.0
labels.append(label)
            flbase = os.path.basename(fl)
            img_names.append(flbase)
            cls.append(fields)
    images = np.array(images)
    labels = np.array(labels)
    img_names = np.array(img_names)
    cls = np.array(cls)

return images, labels, img_names, cls


class DataSet(object):

def __init__(self, images, labels, img_names, cls):
self._num_examples = images.shape[0]

self._images = images
self._labels = labels
self._img_names = img_names
self._cls = cls
self._epochs_done = 0
self._index_in_epoch = 0

@property
def images(self):
return self._images

@property
def labels(self):
return self._labels

@property
def img_names(self):
return self._img_names

@property
```

```
def cls(self):
return self._cls

@property
def num_examples(self):
return self._num_examples

@property
def epochs_done(self):
return self._epochs_done

def next_batch(self, batch_size):
"""Return the next `batch_size` examples from this data set."""
start = self._index_in_epoch
self._index_in_epoch += batch_size

if self._index_in_epoch >self._num_examples:
# After each epoch we update this
self._epochs_done += 1
start = 0
self._index_in_epoch = batch_size
assert batch_size <= self._num_examples
    end = self._index_in_epoch

return self._images[start:end], self._labels[start:end],
self._img_names[start:end], self._cls[start:end]


def read_train_sets(train_path, image_size, classes, validation_size):
class DataSets(object):
pass
data_sets = DataSets()

  images, labels, img_names, cls = load_train(train_path, image_size, classes)
  images, labels, img_names, cls = shuffle(images, labels, img_names, cls)

if isinstance(validation_size, float):
    validation_size = int(validation_size * images.shape[0])

  validation_images = images[:validation_size]
  validation_labels = labels[:validation_size]
  validation_img_names = img_names[:validation_size]
  validation_cls = cls[:validation_size]

  train_images = images[validation_size:]
  train_labels = labels[validation_size:]
  train_img_names = img_names[validation_size:]
  train_cls = cls[validation_size:]

  data_sets.train = DataSet(train_images, train_labels, train_img_names,
train_cls)
  data_sets.valid = DataSet(validation_images, validation_labels,
validation_img_names, validation_cls)

return data_sets
import tensorflow as tf
import numpy as np

from tkinter import *
```

```
import os
from tkinter import filedialog
import cv2
import time
from matplotlib import pyplot as plt
from tkinter import messagebox







def endprogram():
print ("\nProgram terminated!")
   sys.exit()










def training():

import Training as tr







def imgtraining():
    import_file_path = filedialog.askopenfilename()

    image = cv2.imread(import_file_path)
    filename = 'Test.jpg'
cv2.imwrite(filename, image)
print("After saving image:")

    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    cv2.imshow('Original image', image)
    cv2.imshow('Gray image', gray)
# import_file_path = filedialog.askopenfilename()
print(import_file_path)
    fnm = os.path.basename(import_file_path)
print(os.path.basename(import_file_path))

from PIL import Image, ImageOps

    im = Image.open(import_file_path)
    im_invert = ImageOps.invert(im)
    im_invert.save('lena_invert.jpg', quality=95)
    im = Image.open(import_file_path).convert('RGB')
    im_invert = ImageOps.invert(im)
    im_invert.save('tt.png')
    image2 = cv2.imread('tt.png')
    cv2.imshow("Invert", image2)

""""-----------------------------------------------"""

img = image

gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    cv2.imshow('Original image', img)
#cv2.imshow('Gray image', gray)
dst = cv2.fastNlMeansDenoisingColored(img, None, 10, 10, 7, 21)
    cv2.imshow("Nosie Removal", dst)

gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)



print("\n*********************\nImage : " + fnm + "\n*********************")
    img = cv2.imread(import_file_path)
if img is None:
print('no data')

    img1 = cv2.imread(import_file_path)
print(img.shape)
    img = cv2.resize(img, ((int)(img.shape[1] / 5), (int)(img.shape[0] / 5)))
original = img.copy()
neworiginal = img.copy()
    cv2.imshow('original', img1)
    gray = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)

    cv2.imshow('Original image', img1)
# cv2.imshow('Gray image', gray)
p = 0
for i in range(img.shape[0]):

for j in range(img.shape[1]):
            B = img[i][j][0]
            G = img[i][j][1]
            R = img[i][j][2]
if (B >110 and G >110 and R >110):
                p += 1

totalpixels = img.shape[0] * img.shape[1]
    per_white = 100 * p / totalpixels
if per_white >10:
        img[i][j] = [500, 300, 200]
        cv2.imshow('color change', img)
# Guassian blur
blur1 = cv2.GaussianBlur(img, (3, 3), 1)
# mean-shift algo
newimg = np.zeros((img.shape[0], img.shape[1], 3), np.uint8)
    criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 10, 1.0)
    img = cv2.pyrMeanShiftFiltering(blur1, 20, 30, newimg, 0, criteria)
    cv2.imshow('means shift image', img)
# Guassian blur
blur = cv2.GaussianBlur(img, (11, 11), 1)
    cv2.imshow('Noise Remove', blur)
    corners = cv2.goodFeaturesToTrack(gray, 27, 0.01, 10)
    corners = np.int0(corners)
```

```
# we iterate through each corner,
    # making a circle at each point that we think is a corner.
for i in corners:
        x, y = i.ravel()
        cv2.circle(image, (x, y), 3, 255, -1)

    plt.imshow(image), plt.show()




def testing():
global testing_screen
    testing_screen = Toplevel(main_screen)
    testing_screen.title("Testing")
# login_screen.geometry("400x300")
testing_screen.geometry("600x450+650+150")
    testing_screen.minsize(120, 1)
    testing_screen.maxsize(1604, 881)
    testing_screen.resizable(1, 1)
# login_screen.title("New Toplevel")

Label(testing_screen, text='''Upload Image''', background="#d9d9d9",
disabledforeground="#a3a3a3",
foreground="#000000", bg="turquoise", width="300", height="2", font=("Calibri",
16)).pack()
    Label(testing_screen, text="").pack()
    Label(testing_screen, text="").pack()
    Label(testing_screen, text="").pack()
    Button(testing_screen, text='''Upload Image''', font=(
'Verdana', 15), height="2", width="30", command=imgtest).pack()

def imgtest():
    import_file_path = filedialog.askopenfilename()

    image = cv2.imread(import_file_path)
print(import_file_path)
    filename = 'data/alien_test/Test.jpg'
cv2.imwrite(filename, image)
print("After saving image:")




def main_account_screen():
from PIL import Image, ImageTk
global main_screen
    main_screen = Tk()
    width = 600
height = 600
screen_width = main_screen.winfo_screenwidth()
    screen_height = main_screen.winfo_screenheight()
    x = (screen_width / 2) - (width / 2)
    y = (screen_height / 2) - (height / 2)
    main_screen.geometry("%dx%d+%d+%d" % (width, height, x, y))
    main_screen.resizable(0, 0)
# main_screen.geometry("300x250")
main_screen.title("Leaf Disease classification")

    Label(text="Leaf Disease classification", bg="turquoise", width="300",
height="5", font=("Calibri", 16)).pack()
    Label(text="").pack()
    Label(text="").pack()

    image = ImageTk.PhotoImage(Image.open('gui/12344.jpg'))

    Label(main_screen, text='Hello', image=image, compound='left', height="100",
width="200",).pack()

    Button(text="Training", font=(
'Verdana', 15), height="2", width="30", command=training,
highlightcolor="black").pack(side=TOP)
    Label(text="").pack()
    Button(text="Testing", font=(
'Verdana', 15), height="2", width="30", command=testing).pack(side=TOP)

    Label(text="").pack()

    main_screen.mainloop()


main_account_screen()
```

GitHub &Project Demo link :

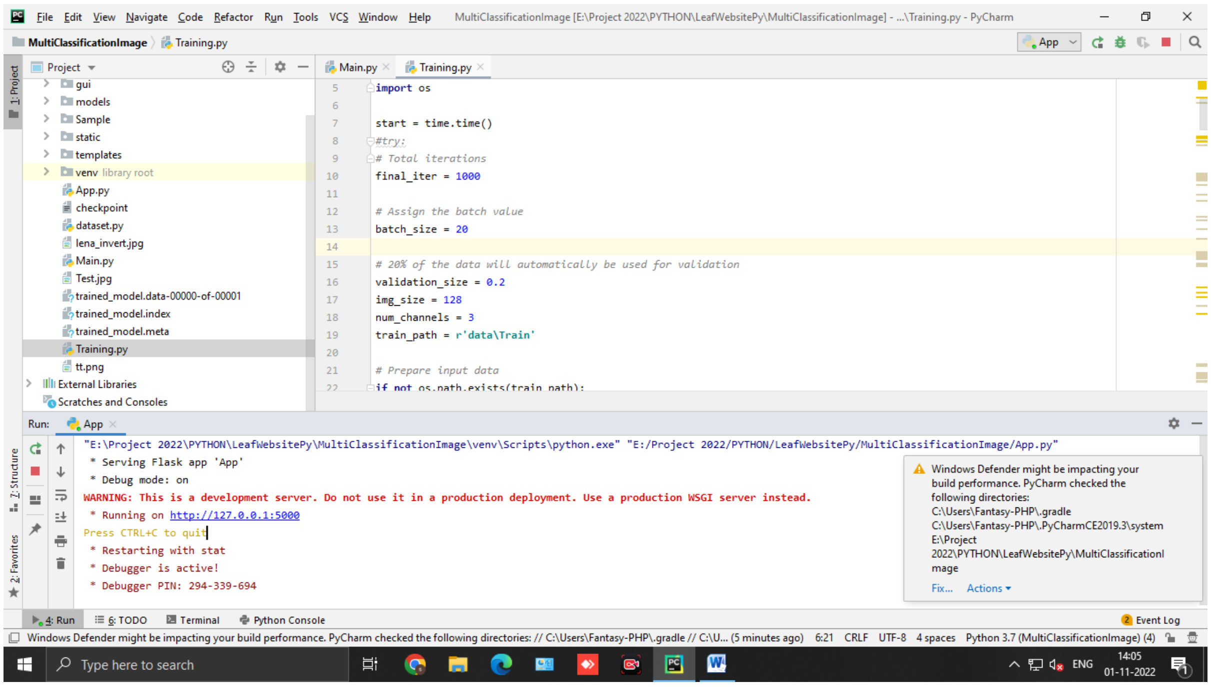
GitHub :

IBM-EPBL/IBM-Project-12474-1659451958

Project Demo link :

https://www.youtube.com/watch?v=CMWIfIJVYyI

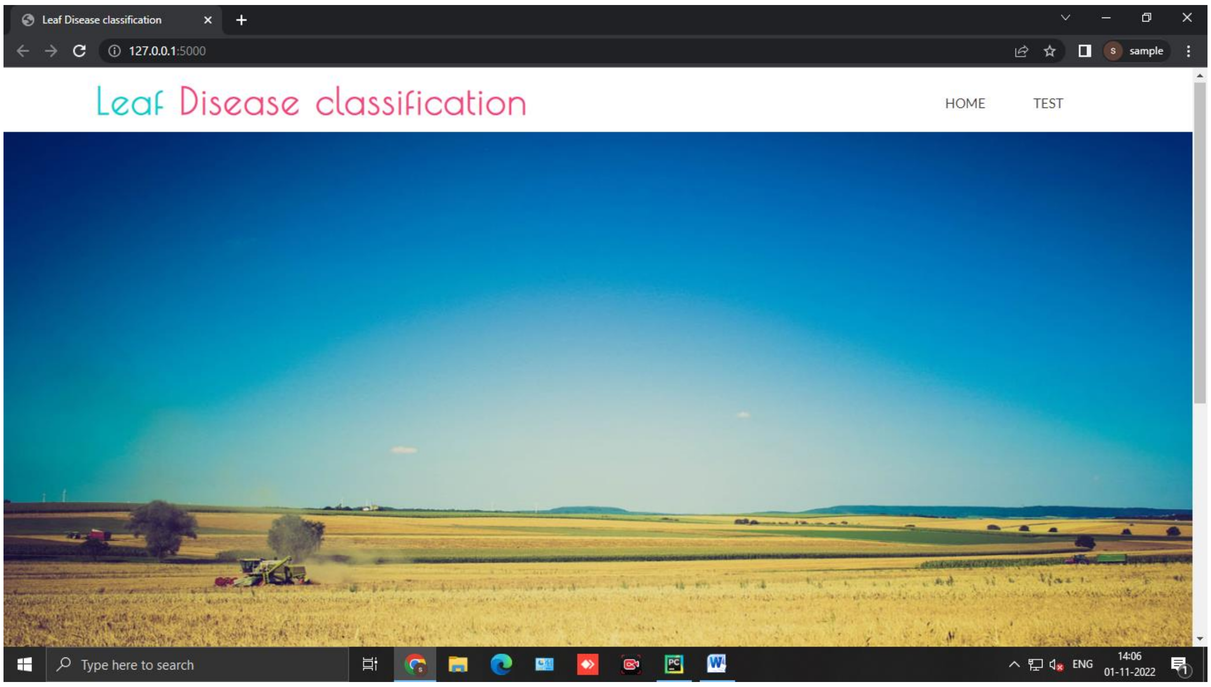
OUTPUT :

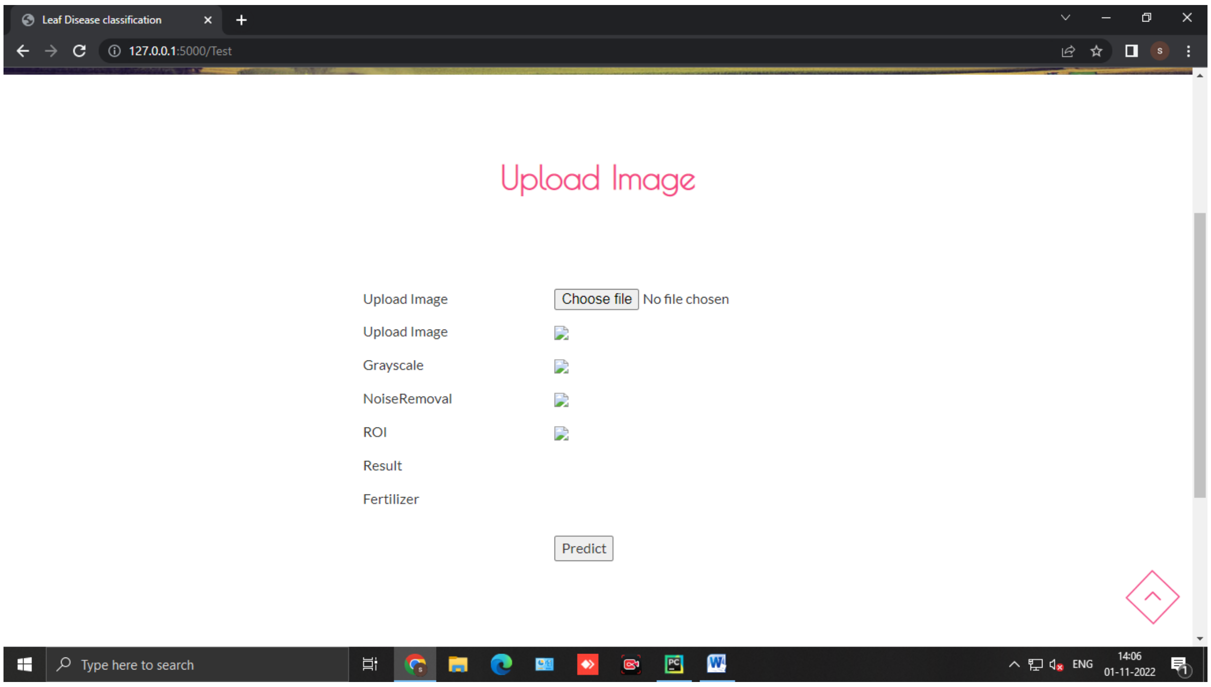


```python
import os

start = time.time()
#try:
# Total iterations
final_iter = 1000

# Assign the batch value
batch_size = 20

# 20% of the data will automatically be used for validation
validation_size = 0.2
img_size = 128
num_channels = 3
train_path = r'data\Train'

# Prepare input data
if not os.path.exists(train_path):
```

```
"E:\Project 2022\PYTHON\LeafWebsitePy\MultiClassificationImage\venv\Scripts\python.exe" "E:/Project 2022/PYTHON/LeafWebsitePy/MultiClassificationImage/App.py"
 * Serving Flask app 'App'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 294-339-694
```

Windows Defender might be impacting your build performance. PyCharm checked the following directories:
C:\Users\Fantasy-PHP\.gradle
C:\Users\Fantasy-PHP\.PyCharmCE2019.3\system
E:\Project 2022\PYTHON\LeafWebsitePy\MultiClassificationImage




Leaf Disease classification

127.0.0.1:5000/Test

Upload Image

Upload Image Choose file No file chosen

Upload Image

Grayscale

NoiseRemoval

ROI

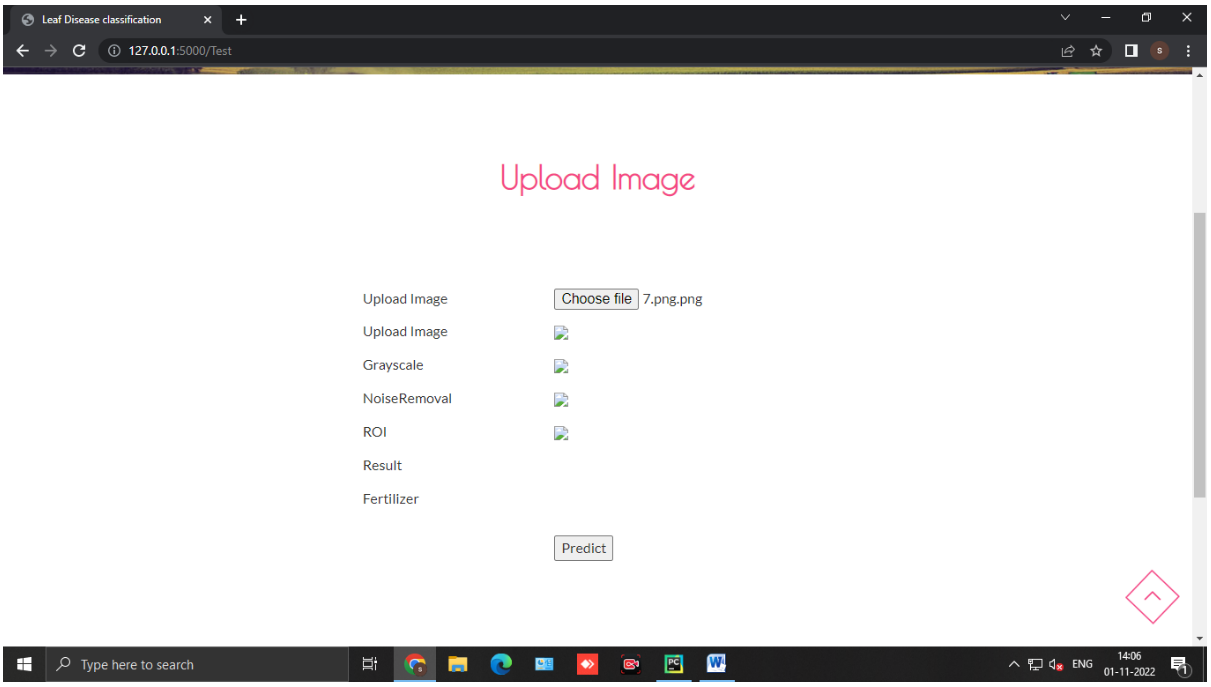
Result

Fertilizer

Predict

Type here to search

ENG 14:06 01-11-2022


Leaf Disease classification

127.0.0.1:5000/Test

Upload Image

Upload Image Choose file 7.png.png

Upload Image

Grayscale

NoiseRemoval

ROI

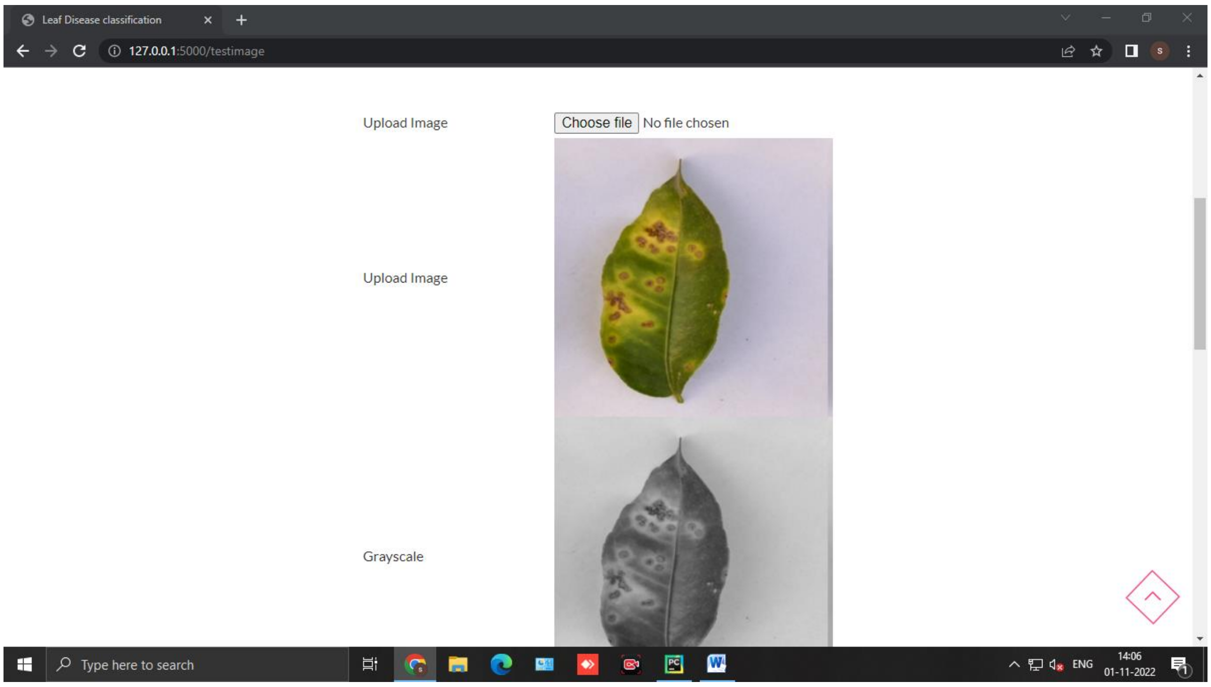
Result

Fertilizer

Predict

Type here to search

ENG 14:06 01-11-2022

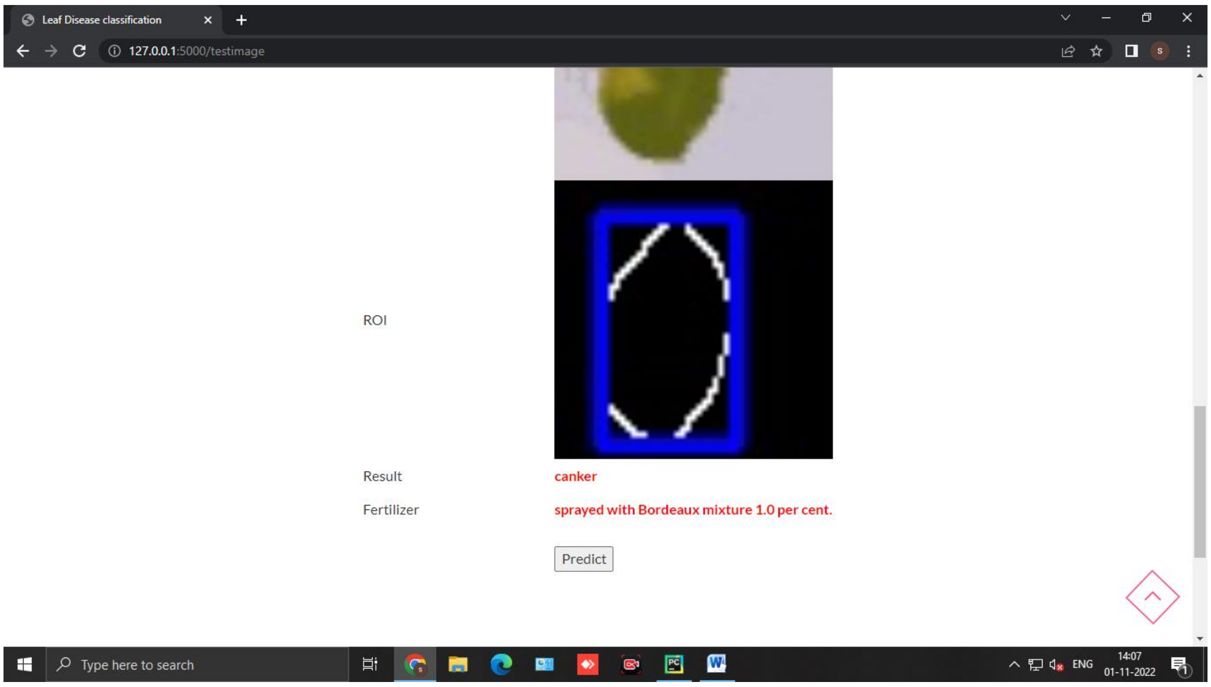

Leaf Disease classification

127.0.0.1:5000/testimage

Upload Image — Choose file No file chosen

Upload Image

Grayscale


Leaf Disease classification

127.0.0.1:5000/testimage

ROI

Result — canker

Fertilizer — sprayed with Bordeaux mixture 1.0 per cent.

Predict