FERTILIZERS RECOMMENDATION SYSTEM FOR DISEASE PREDICTION

TEAM ID:PNT2022TMID30525

TEAM MEMBERS:

AKALYA.S-613019104002

DEVIKA P-613019104016

DHIVYADHARSHNI S-613019104022

MANJU V-613019104045

VIVEKANANDHA COLLEGE OF TECHNOLOGY FOR WOMEN

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

MENTOR : K.STEEPHEN
Assistant Professor ,
CSE Department

Abstract

Agriculture is one among the most important sector of Indian Economy. More than50% of population depends on agriculture as their source of income. Growing crops over thousands of years without caring about replenishing has led to depletion and exhaustion of soil nutrients resulting in their low productivity. To improve the production, chemical fertilizers were added. But excessive use of these not only makes the plants dependent on artificial fertilizers but also erodes the natural quality of land**.**

It is therefore important to make sure that only the sufficient number of fertilizers is added so that yield can be increased and at the same time, the natural quality of soil remain intact. For this, it is essential to know the soil nutrient levels. Our project aims at finding the soil nutrient richness and predict the fertility of a given soil sample in real time. Based on the result obtained, the system will also be giving recommendation on the type of fertilizer and in what quantity It is to be used in the soil sample so that the yield can be maximized. Detection and recognition of plant diseases using machine learning are very efficient in providing symptoms of identifying diseases at its earliest.

1.INTRODUTION

India is an agricultural country and depends on agricultural products for their wellbeing. It is agriculture that promotes the economic growth and the development of our country. But recently many problems have been faced by the farmers due to certain natural calamities. Apart from these major calamities, they were also in lack of sufficient knowledge about the nutrients present in their soil. Not only the soil type, the climatic condition and the usage of fertilizer also play a major role. Certain varieties of crops can be cultivated based on the climatic condition in their locality and accordingly the fertilizers can be preferred. In certain situation the usage of fertilizers also affects the cultivation. Under a climatic condition the cultivation of a right crop, usage of right fertilizer to the soil gives better yield. Mostly, fertilizers were recommended based on the nutrients present in the soil. On using chemical fertilizer the quality or the nutrients present in the soil was degraded, that promotes a decrease in the nutritive value of the soil. Another major factor to be considered is the disease in the crop cultivated. Identifying the disease in the plants and preferring appropriate fertilizer by the agriculturist to the farmers plays a major role. In earlier days, all these process were carried out manually. But with the advancement of technology the entire system was digitalized. But even then there exist various problems that need careful attention.

1. 1 PROJECT OVERVIEW:

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

1.2 PURPOSE:

The purpose of analyzing later-Plant leaf disease 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. 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 emphasize.

2.LITERATURE SURVEY

2. 1 EXISTING PROBLEM:

[1]The author proposed a method which helps us to find the crop affected by disease and recommend a fertilizer for the crop based on the severity level. The fertilizers may be organic or inorganic. The algorithm that they used to train the model is Support Vector machine. 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 hyperplane constructed with conditions to categorize the preprocessed leaves and also implement multiclass classifier, to predict diseases in leaf image.

Advantages: The proposed SVM technique gives a better result. For the same set of images, F-Measure for CNN is 0.7and 0.8 for SVM, the accuracy of identification of leaf disease of CNN is 0.6 and SVM is 0.8.

Disadvantages: The accuracy of this algorithm can further be improved and does not provide accurate result for all diseased crops.

[2] The main objective of this paper is image analysis & classification techniques for detection of leaf diseases and classification. The leaf image is firstly preprocessed and then does the further work. K-Means Clustering used for image segmentation and

then system extract the GLCM features from disease detected images. The disease classification done through the SVM classifier.

Algorithm used: Gray-Level Co-Occurrence Matrix (GLCM) features, SVM, K-Means Clustering .

Advantages: The system detects the diseases on citrus leaves with 90% accuracy.
Disadvantages: System only able to detect the disease from citrus leaves.

[3] The proposed method uses SVM to classify tree leaves, identify the disease and suggest the fertilizer. The proposed method is compared with the existing CNN based leaf disease prediction. The proposed SVM technique gives a better result when compared to existing CNN. For the same set of images, F-Measure for CNN is 0.7and 0.8 for SVM, the accuracy of identification of leaf disease of CNN is 0.6 and SVM is 0.8.

Advantages: The prediction and diagnosing of leaf diseases are depending on the segmentation such as segmenting the healthy tissues from diseased tissues of leaves. **Disadvantages:** This further research is implementing the proposed algorithm with the existing public datasets. Also, various segmentation algorithms can be implemented to improve accuracy. The proposed algorithm can be modified further to identify the disease that affects the various plant organs such as stems and fruits.

2. 2 REFERENCES :

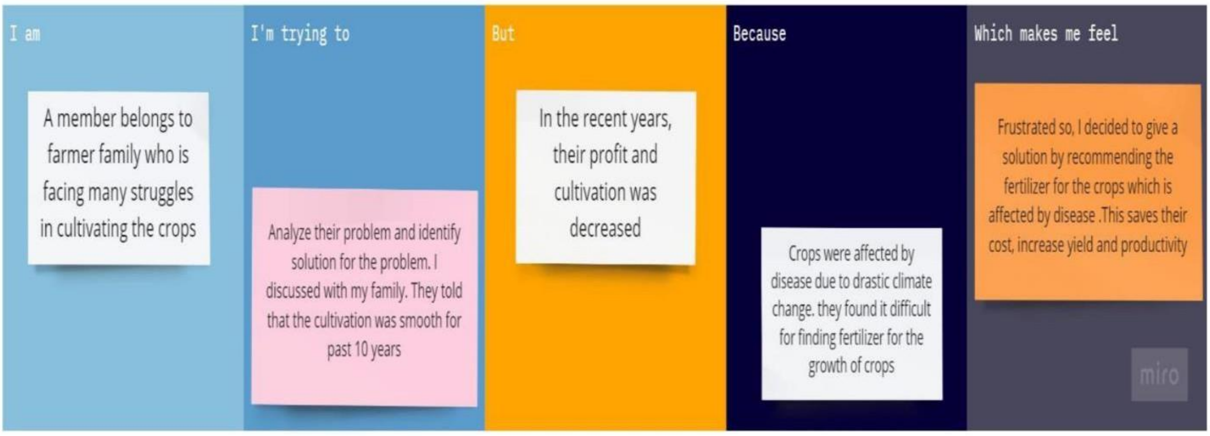
[1] Luca Bencini, Davide Di Palma, Giovanni Collodi, G. Manes and Antonio Manes, "Agricultural monitoring based on wireless sensor network technology: Real long life deployments for physiology and pathogens control.". Third International Conference on Sensor Technologies and Applications. IEEE, 2009.

[2] International Journal of Computer Science and Informatics. Jay Gholap, Anurag Ingole, Jayesh Gohil, Shailesh Gargade and Vahida Attar, "Soil Data Analysis Using Classification Techniques and Soil Attribute Prediction", IJCSI, Vol. 9, Issue 3, No 3, ISSN: 1694-0814,May2012.

[3] 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.

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.



I am

A member belongs to farmer family who is facing many struggles in cultivating the crops

I'm trying to

Analyze their problem and identify solution for the problem. I discussed with my family. They told that the cultivation was smooth for past 10 years

But

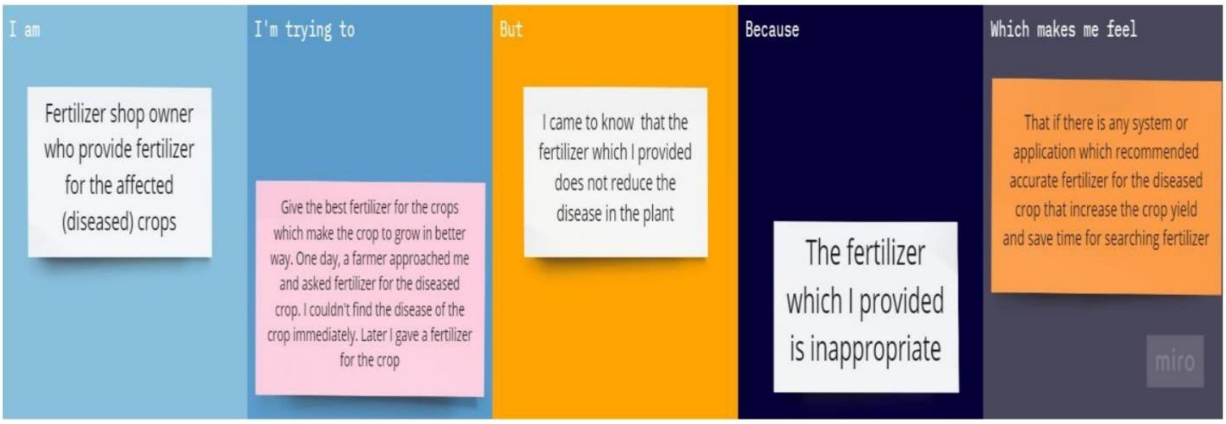
In the recent years, their profit and cultivation was decreased

Because

Crops were affected by disease due to drastic climate change. they found it difficult for finding fertilizer for the growth of crops

Which makes me feel

Frustrated so, I decided to give a solution by recommending the fertilizer for the crops which is affected by disease .This saves their cost, increase yield and productivity



I am

Fertilizer shop owner who provide fertilizer for the affected (diseased) crops

I'm trying to

Give the best fertilizer for the crops which make the crop to grow in better way. One day, a farmer approached me and asked fertilizer for the diseased crop. I couldn't find the disease of the crop immediately. Later I gave a fertilizer for the crop

But

I came to know that the fertilizer which I provided does not reduce the disease in the plant

Because

The fertilizer which I provided is inappropriate

Which makes me feel

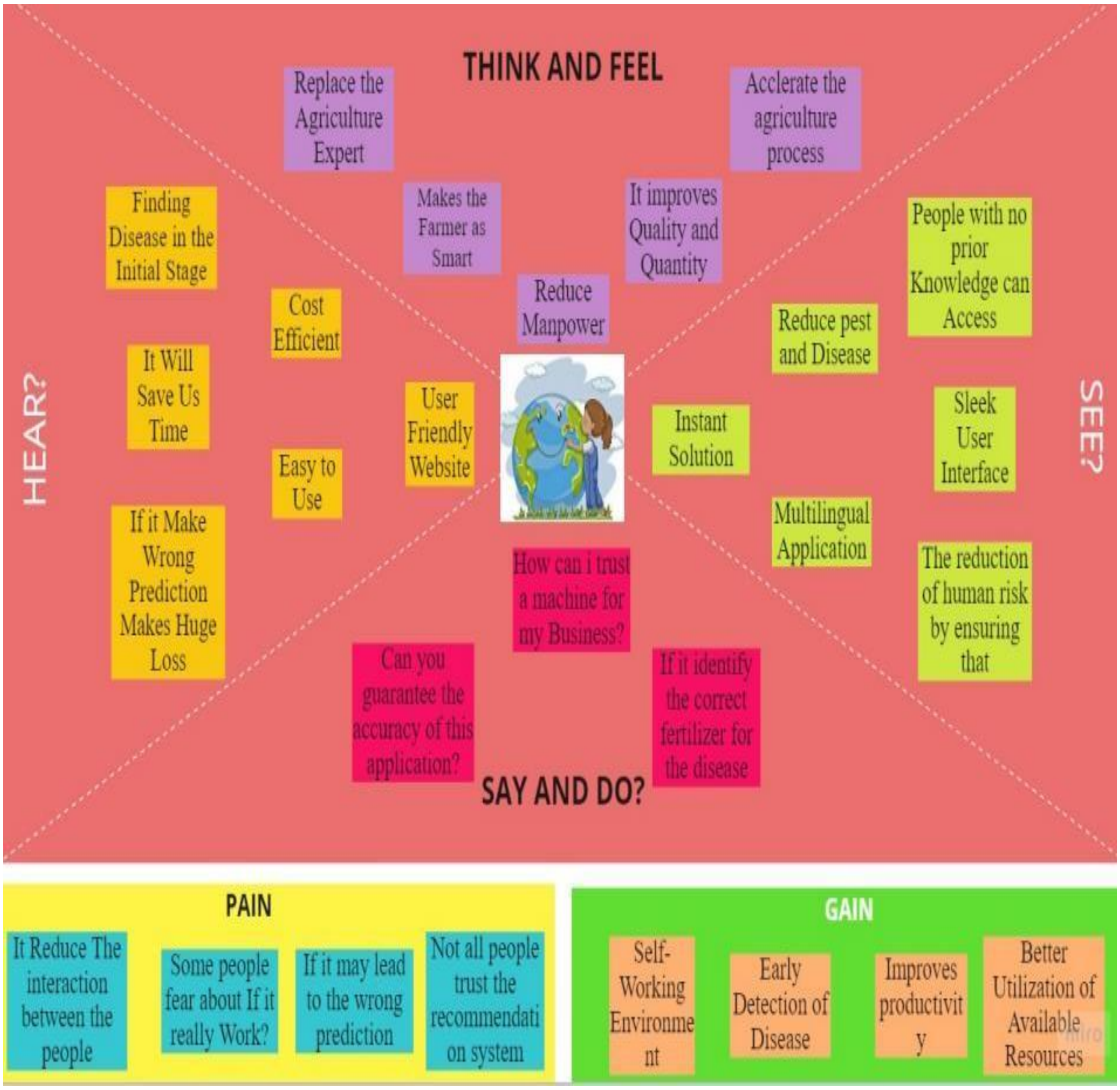
That if there is any system or application which recommended accurate fertilizer for the diseased crop that increase the crop yield and save time for searching fertilizer

Problem Statement (PS)	**I am (Customer)**	**I'm trying to**	**But**	**Because**	**Which makes mefeel**
PS 1	A member belongs to farmer family who is facing many struggles in cultivating the crops	Analyze their problem and identify solution for the problem. I discussed with my family. They told that the cultivation was smooth for past 10 years	In the recent years, their profit and cultivation was decreased	Crops were affected by disease due to drastic climate change. they found it difficult for finding fertilizer for the growth of crops	Frustrated so, I decided to give a solution by recommending the fertilizer for the crops which is affected by disease .This saves their cost, increase yield and productivity
PS 2	Fertilizer shop owner who provide fertilizer for the affected (diseased) crops	Give the best fertilizer for the crops which make the crop to grow in better way. One day, a farmer approached me and asked fertilizer for the diseased crop. I couldn't find the disease of the crop immediately. Later I gave a fertilizer for the crop	I came to know that the fertilizer which I provided does not reduce the disease in the plant	The fertilizer which I provided is inappropriate	That if there is any system or application which recommended accurate fertilizer for the diseased crop that increase the crop yield and save time for searching fertilizer

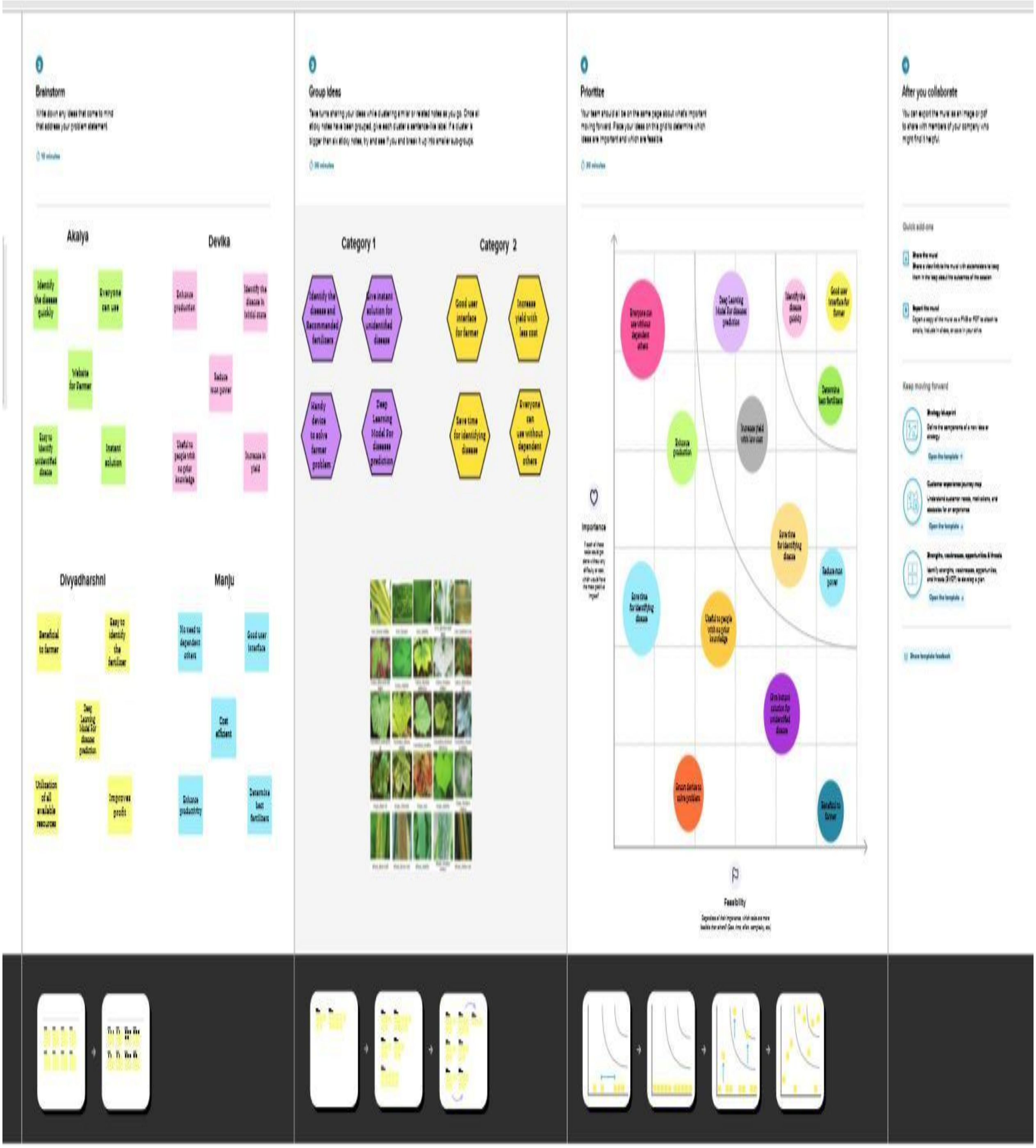
3.IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS:

A fast and exact approach to identify plant infections appears so critical for the good thing about trade and biology to agriculture. Disease control procedures can be a waste of money and time if the disease isn’t properly identified and it can lead to more plant loss. Our project proposes a deep on the model will achieve its goal by categorising photos of leaves into unhealthy categories based on defect patterns**.**


THINK AND FEEL
Replace the Agriculture Expert
Acclerate the agriculture process
Makes the Farmer as Smart
It improves Quality and Quantity
Reduce Manpower
HEAR?
Finding Disease in the Initial Stage
Cost Efficient
It Will Save Us Time
Easy to Use
User Friendly Website
If it Make Wrong Prediction Makes Huge Loss
SEE?
People with no prior Knowledge can Access
Reduce pest and Disease
Sleek User Interface
Instant Solution
Multilingual Application
The reduction of human risk by ensuring that
SAY AND DO?
How can i trust a machine for my Business?
Can you guarantee the accuracy of this application?
If it identify the correct fertilizer for the disease
PAIN
It Reduce The interaction between the people
Some people fear about If it really Work?
If it may lead to the wrong prediction
Not all people trust the recommendation system
GAIN
Self-Working Environment
Early Detection of Disease
Improves productivity
Better Utilization of Available Resources
miro

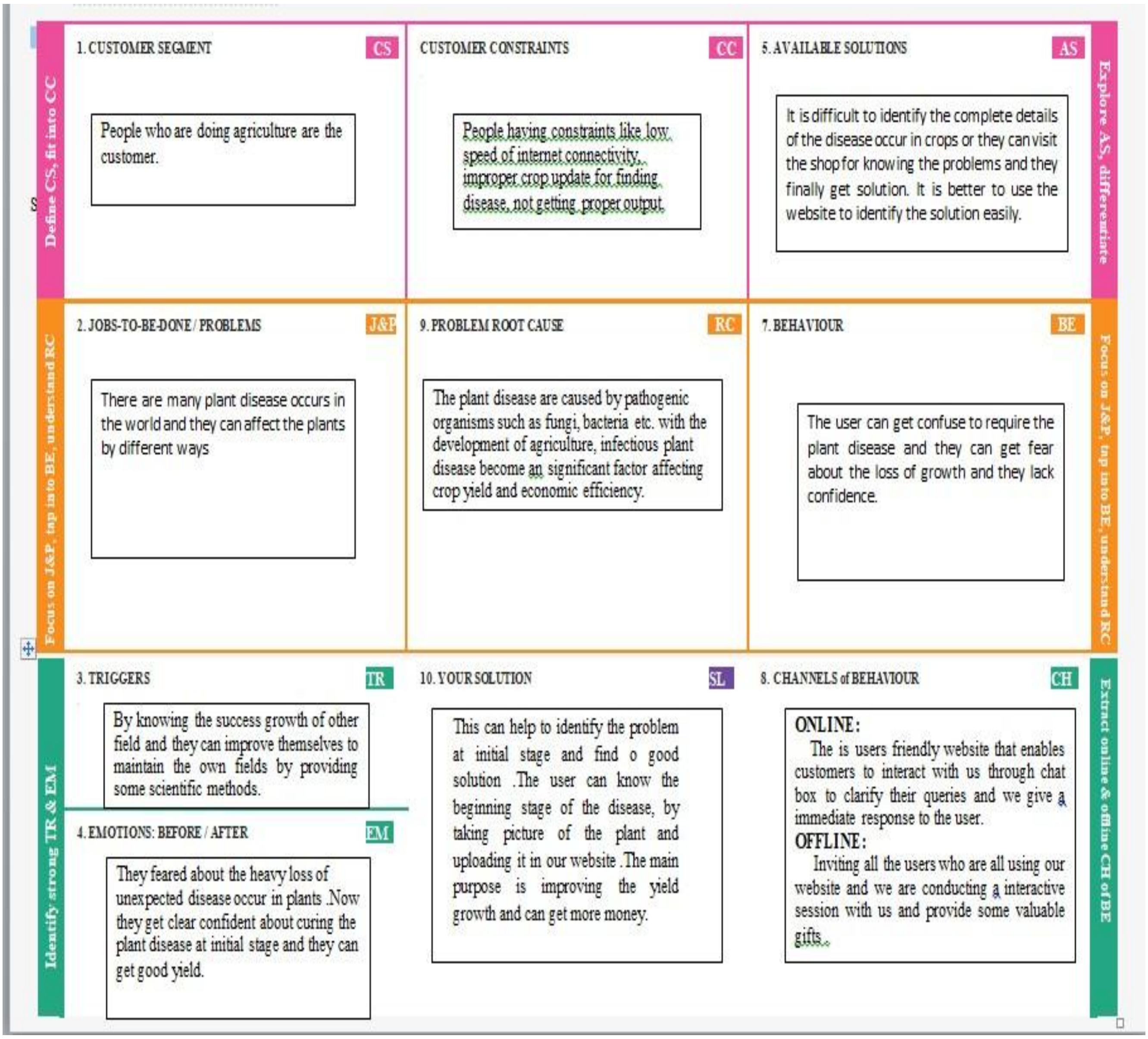
3.2 IDEATION & BRAINSTORMING


Brainstorm
Write down any ideas that come to mind that address your problem statement.
10 minutes
Akalya
Identify the disease quickly
Everyone can use
Website for farmer
Easy to identify unidentified disease
Instant solution
Devika
Enhance production
Identify the disease in initial stage
Reduce man power
Useful to people with no prior knowledge
Increase in yield
Divyadharshni
Beneficial to farmer
Easy to identify the fertilizer
Deep Learning Model for disease prediction
Utilization of all available resources
Improves profit
Manju
No need to dependent others
Good user interface
Cost efficient
Enhance productivity
Determine best fertilizers
Group ideas
Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, 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
Identify the disease and Recommended fertilizers
Give instant solution for unidentified disease
Handy device to solve farmer problem
Deep Learning Model for disease prediction
Category 2
Good user interface for farmer
Increase yield with less cost
Save time for identifying disease
Everyone can use without dependent others
Prioritize
Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.
20 minutes
Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
Everyone can use without dependent others
Deep Learning Model for disease prediction
Identify the disease quickly
Good user interface for farmer
Determine best fertilizers
Enhance production
Increase yield with less cost
Save time for identifying disease
Reduce man power
Save time for identifying disease
Useful to people with no prior knowledge
Give instant solution for unidentified disease
Handy device to solve problem
Beneficial to farmer
Feasibility
Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)
After you collaborate
You can export the mural as an image or pdf to share with members of your company who might find it helpful.
Quick add-ons
Share the mural
Export the mural
Keep moving forward
Strategy blueprint
Open the template
Customer experience journey map
Open the template
Strengths, weaknesses, opportunities & threats
Open the template
Share template feedback

3.3 PROPOSED SOLUTION:

S.NO	Parameter	Description
1.	Problem Statement (Problem to be solved)	Agriculture is an extremely risky industry and our farmers are at the forefront of the industry. Farmer's face several problems which include crops get affected by diseases, the soil is not being nutritious enough for the plant to grow, etc. More than 15% of the crops get wasted in India due to diseases and hence it has become one of the major concerns to be resolved.
2.	Idea / Solution description	The solution for problem is by using Deep learning technique, which recommended the fertilizer for the affected crops. The fertilizer recommendation model provide the best recommendation for the crop based on their nutrition value, the most important issue is plant gets easily affected by diseases. We also create a website for using this application as this application is used anywhere by anyone. We have tried many different algorithms for finding better accuracy to recommend right fertilizer for affected crop.
3.	Novelty / Uniqueness	The combination of increasing global Smartphone penetration in the rural areas and recent advances in computer vision made possible by deep learning has paved the way for web assisted disease diagnosis also we have created an open source easy to use web application to address the issues which help to improve crop production.
4.	Social Impact / Customer Satisfaction	The farmer who uses this application has ability to minimize the manpower needed for recommendation. Using this model, the agriculture domain will have a great impact on the environment and there is an increase in the productivity and yield. The buyer and seller will also be more beneficial by attaining more profit
5.	Business Model (Revenue Model)	The usages of this application for predicting the fertilizers, analyzing the disease in a touch make the farmers easy with minimal subscription. This action adds a lot of value to the company and business in society
6.	Scalability of the Solution	The machine learning model which is trained using CNN have the capacity to scale when needed and operated with same speed. High model accuracy will improve for more testing and training data.

3.4.PROBLEM SOLUTION FIT:



Define CS, fit into CC

1. CUSTOMER SEGMENT CS

People who are doing agriculture are the customer.

CUSTOMER CONSTRAINTS CC

People having constraints like low speed of internet connectivity, improper crop update for finding disease, not getting proper output.

5. AVAILABLE SOLUTIONS AS

It is difficult to identify the complete details of the disease occur in crops or they can visit the shop for knowing the problems and they finally get solution. It is better to use the website to identify the solution easily.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

There are many plant disease occurs in the world and they can affect the plants by different ways

9. PROBLEM ROOT CAUSE RC

The plant disease are caused by pathogenic organisms such as fungi, bacteria etc. with the development of agriculture, infectious plant disease become an significant factor affecting crop yield and economic efficiency.

7. BEHAVIOUR BE

The user can get confuse to require the plant disease and they can get fear about the loss of growth and they lack confidence.

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

By knowing the success growth of other field and they can improve themselves to maintain the own fields by providing some scientific methods.

4. EMOTIONS: BEFORE / AFTER EM

They feared about the heavy loss of unexpected disease occur in plants .Now they get clear confident about curing the plant disease at initial stage and they can get good yield.

10. YOUR SOLUTION SL

This can help to identify the problem at initial stage and find o good solution .The user can know the beginning stage of the disease, by taking picture of the plant and uploading it in our website .The main purpose is improving the yield growth and can get more money.

8. CHANNELS of BEHAVIOUR CH

ONLINE:

The is users friendly website that enables customers to interact with us through chat box to clarify their queries and we give a immediate response to the user.

OFFLINE:

Inviting all the users who are all using our website and we are conducting a interactive session with us and provide some valuable gifts..

Extract online & offline CH of BE

4.REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT :

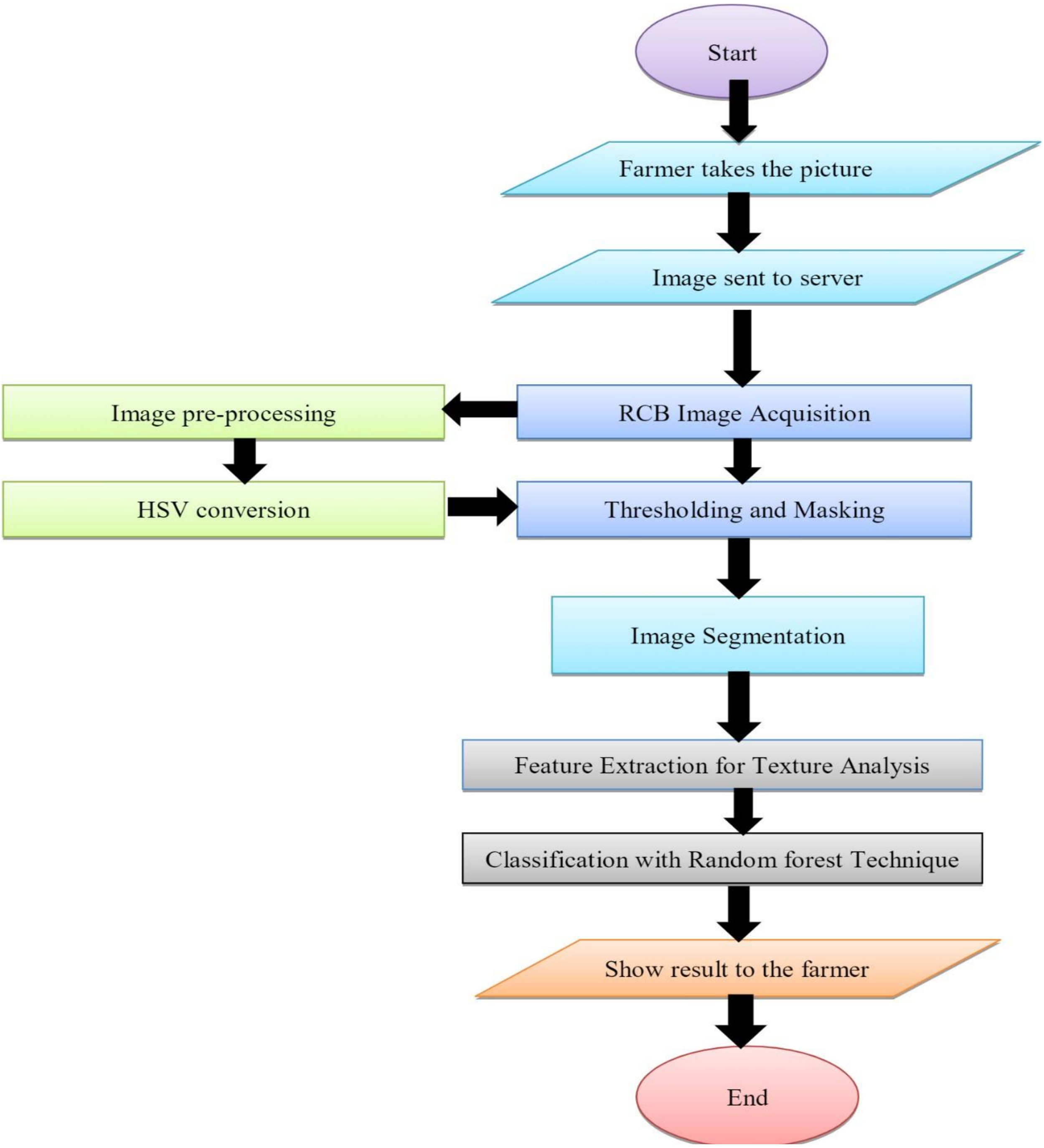
FR NO	**Functional Requirement (Epic)**	**Sub Requirement (Story / Sub-Task)**
FR – 1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR - 2	User Confirmation	Confirmation via Email Confirmation via OTP
FR - 3	Upload Images for Prediction	User need to upload images for the prediction of disease in the image
FR - 4	User Requirement	The user simply give a inputs disease leaf image. The software will instantly generate and accurate disease of leaf to the farmer and also provide the suitable fertilizer to the disease leaf.

4.2 NON FUNCTIONAL REQUIREMENT:

FR No.	**Non-Functional Requirement**	**Description**
NFR-1	**Usability**	User can easily understand the application. They satisfied with the result.
NFR-2	**Security**	With the help of the user name and password it provides more security and the data are private
NFR-3	**Reliability**	This website must perform without failure in 98 percentage of use case
NFR-4	**Performance**	This application supporting 2,000 users per hour must provide 4 seconds or less response time in a desktop browse
NFR-5	**Availability**	User can access any time
NFR-6	**Scalability**	The website must be scalable enough to support 5,000 visits at the same time while maintaining optimal performance

5. PROJECT DESIGN

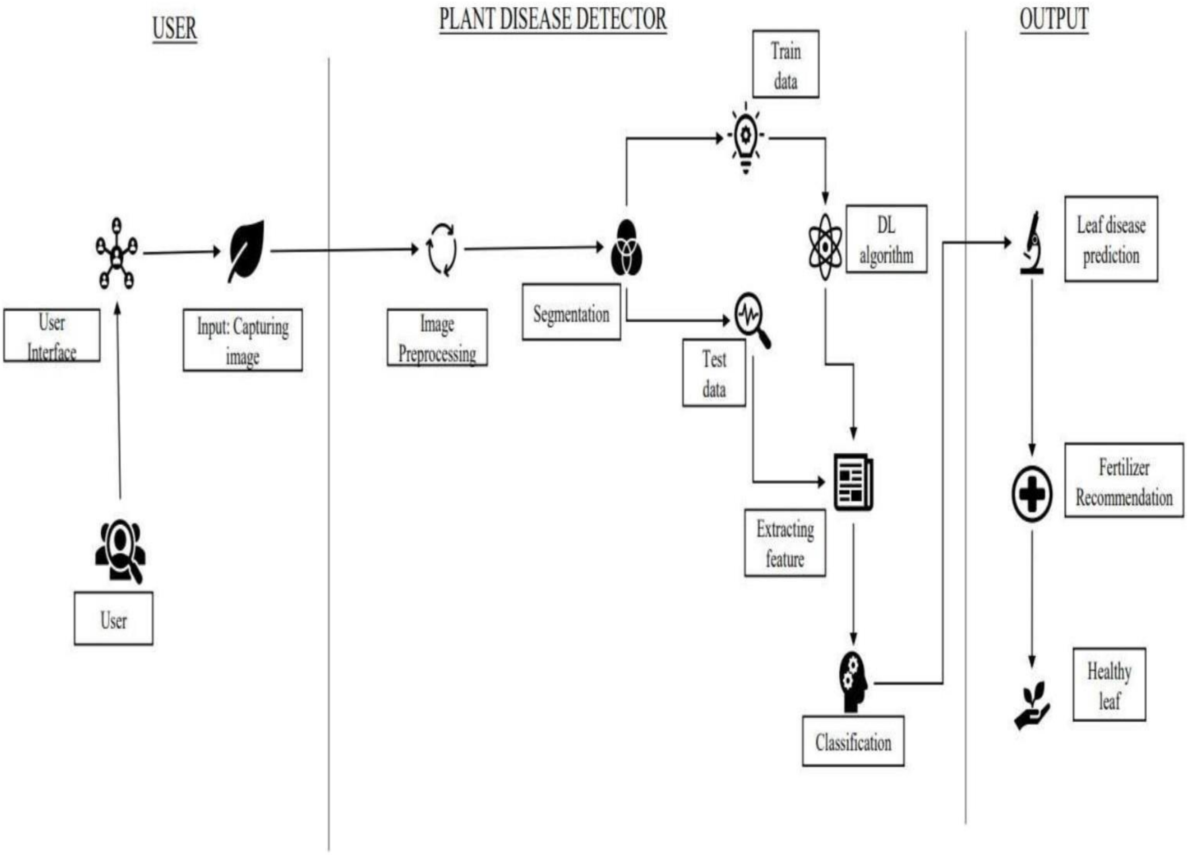
5.1 DATA FLOW DIAGRAM :



5.2 SOLUTION & TECHNICAL ARCHITECTURE:

Solution architecture is a complex process-with many sub-process-that bridges the gapbetween business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software toproject stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed anddelivered.



5.3 USER STORIES :

USER TYPE	FUNCTIONAL REQUIREMENT	USER STORY NUMBER	USER STORY / TASK	ACCEPTANCE CRITERIA	PRIORITY	RELEASE
Customer	Registration	User -1	I can enter into the website and click on the register form and fill the details given in that form	I can get my profile after registering	High	High Speed
Farmer	Login	User-2	As a user ,I can login by using email ,password and confirming password	I can receive confirmation email & click confirm	High	High Speed
Agricultural workers	Sign in	User -3	As a user ,I can sign in by filling some details like name, address , phone number in that chat	After sign in I have a new account	Low	Take a minute

6.PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION:

Sprint Planning:

The performance of Artificial Intelligence (AI) models is being improved and increased in modern technology.

➤ Based Crop Yield Disease Prediction System would assist farmers in protecting their crops from a variety of diseases by identifying them during the process of taking an image at the plant and providing the afflicted disease's name to a machine learning algorithm.

➤ The best answer for the farmer will be provided in this project milestone, and he or she may find it on their own by using a web application with a completely user-friendly and straightforward user interface.

➤ Additionally, we intend to add a useful Module that is a fertilizer prescription for a certain disease to the process. It can propose both artificial and natural fertilizer in a similar way.

Estimation:

1. Planning is a crucial role in project management because it allows team Members to schedule their time on the project.

2. This activity demonstrates how the team members assigned and completed various tasks!

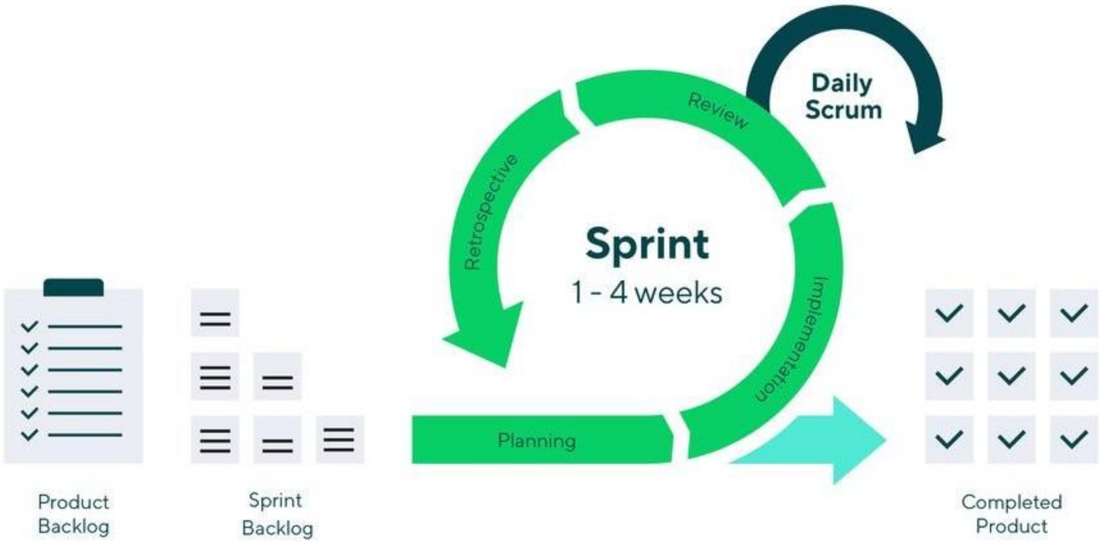
3. In Project we can Split into the Four Step of Phases are,

● Phase 1: Information Collection and Requirement Analysis

● Phase 2: Project Planning and Developing Modules

● Phase 3: Implementing the High Accuracy Deep Learning Algorithm to Perform

● Phase 4: Deploying the Model on Cloud and Testing the Model and UI Performance.



6.2 SPRINT DELIVERY SCHEDULE

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points (Total)	Priority	Team Members
Sprint-1	Collection of Dataset		Collecting all the required dataset that arer used to train and test the model	4	High	Akalya.S, Devika.P, Dhivyadharshini.S, Manju.V
	Model Creation and Training (Fruits)		Create a model which can classify diseased fruit plants from healthy plants from the images. I also need to test the model and deploy it on IBM Cloud.	4	High	Akalya.S, Devika.P, Dhivyadharshini.S, Manju.V
	Model Creation and Training (Vegetables)		Create a model which can classify diseased vegetable plants from healthy plants from the given images and also testing the model by deploying it in IBM Cloud.	4	High	Akalya.S, Devika.P, Dhivyadharshini.S, Manju.V

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points (Total)	Priority	Team Members
Sprint-2	Model Training and testing in IBM Cloud		Create a model which can classify diseased vegetable plants from given images and train on IBM Cloud	6	High	Akalya.S, Devika.P, Dhivyadharshini.S, Manju.V
	Registration	USN-1	As a user, I can register by entering my email, password, and confirming my password or via OAuth API	3	Medium	Akalya.S, Devika.P, Dhivyadharshini.S, Manju.V
	Upload page	USN-2	As a user, I will be redirected to a page where I can select based on my requirement whether to upload my pictures of crops for disease prection or entering my soil details for crop recommendation or fertilizer recommendation.	4	High	Akalya.S, Devika.P, Dhivyadharshini.S, Manju.V
	Suggestion results	USN-3	As a user, I can view the results and then obtain the suggestions provided by the ML model	4	High	Akalya.S, Devika.P, Dhivyadharshini.S, Manju.V
	Base Flask App		A base Flask web app must be created as an interface for the ML model to interact.	2	High	Akalya.S, Devika.P, Dhivyadharshini.S Manju.V
Sprint-3	Login	USN-4	As a user/admin/shopkeeper, I can log into the application by entering email & password	2	High	Akalya.S, Devika.P, Dhivyadharshini.S, Manju.V
	User Dashboard	USN-5	As a user, I can view the previous results and history which saves the user's time.	3	Medium	Akalya.S, Devika.P, Dhivyadharshini.S, Manju.V
	Integration		Integrate Flask, CNN model with Cloud and Database.	5	Medium	Akalya.S, Devika.P, Dhivyadharshini.S, Manju.V

	Containerization		Containerize Flask app using Docker	5	Low	Akalya.S, Devika.P, Dhivyadharshini.S, Manju.V
Sprint-4	Testing and Documentation		Finally the project is tested and further improvements is made based on user feedback. Documentation is also made in order to make better user experience.	4	Medium	Akalya.S, Devika.P, Dhivyadharshini.S, Manju.V
	Dashboard	USN-6	As a shopkeeper, I can enter fertilizer products and then update the details if any	4	Low	Akalya.S, Devika.P, Dhivyadharshini.S, Manju.V
	Containerization		Create and deploy Helm charts using Docker Image made before.	2	Low	Akalya.S, Devika.P, Dhivyadharshini.S, Manju.V

Project Tracker, Velocity & Burndown Chart:

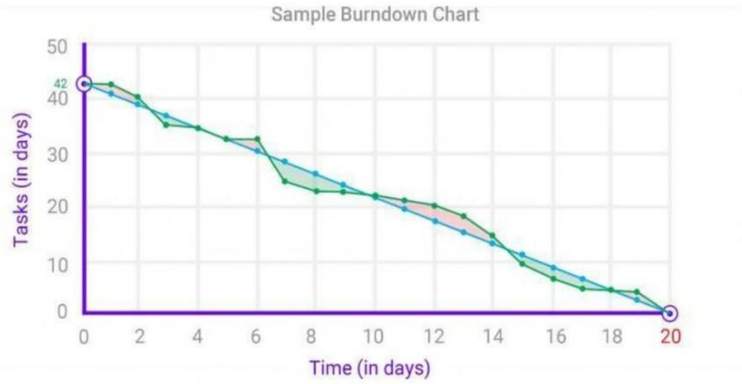
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	10	6 Days	24 Oct 2022	29 Oct 2022	10	30 Oct 2022
Sprint-2	15	6 Days	31 Oct 2022	05 Nov 2022	15	07 Nov 2022
Sprint-3	15	6 Days	07 Nov 2022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

VELOCITY:

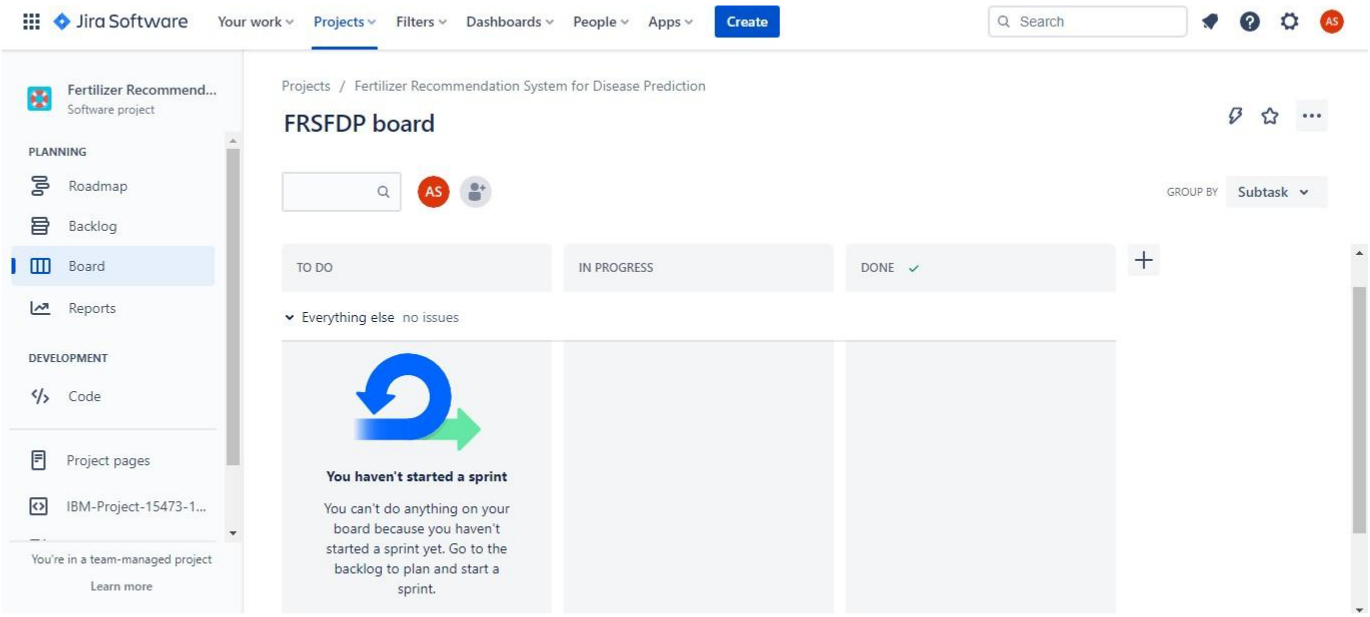
Imagine we have a 10 days sprint duration, and the velocity of the team is 20. Let’s calculate the team’s average velocity per iteration unit.

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

BURNDOWN CHART:



ROADMAP:



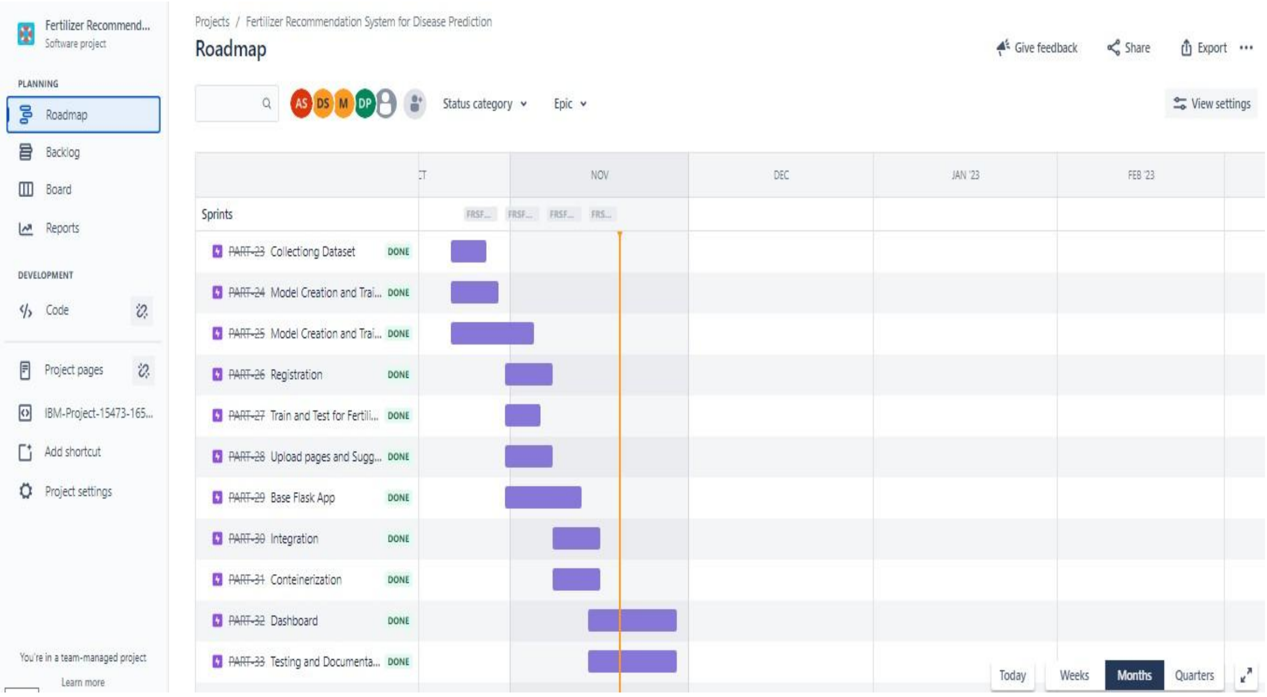
You haven't started a sprint

You can't do anything on your board because you haven't started a sprint yet. Go to the backlog to plan and start a sprint.

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.

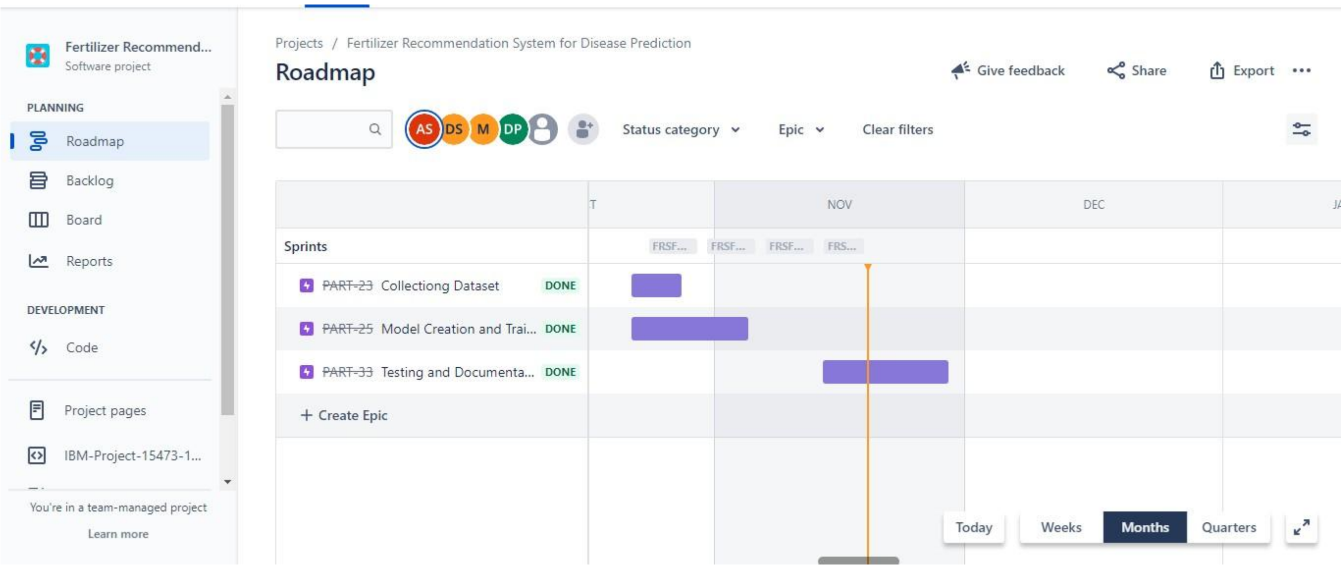


Projects / Fertilizer Recommendation System for Disease Prediction

Roadmap

Sprints

~~PART-23~~ Collectiong Dataset DONE
~~PART-24~~ Model Creation and Trai... DONE
~~PART-25~~ Model Creation and Trai... DONE
~~PART-26~~ Registration DONE
~~PART-27~~ Train and Test for Fertili... DONE
~~PART-28~~ Upload pages and Sugg... DONE
~~PART-29~~ Base Flask App DONE
~~PART-30~~ Integration DONE
~~PART-31~~ Conteinerization DONE
~~PART-32~~ Dashboard DONE
~~PART-33~~ Testing and Documenta... DONE

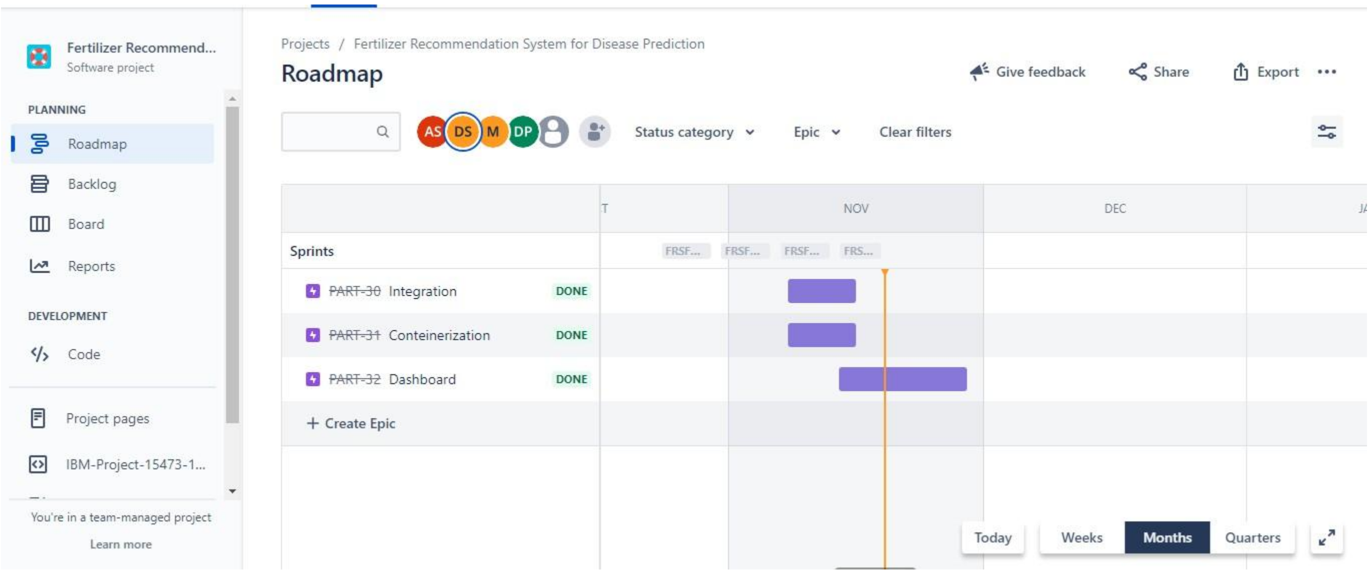


Projects / Fertilizer Recommendation System for Disease Prediction

Roadmap

Sprints

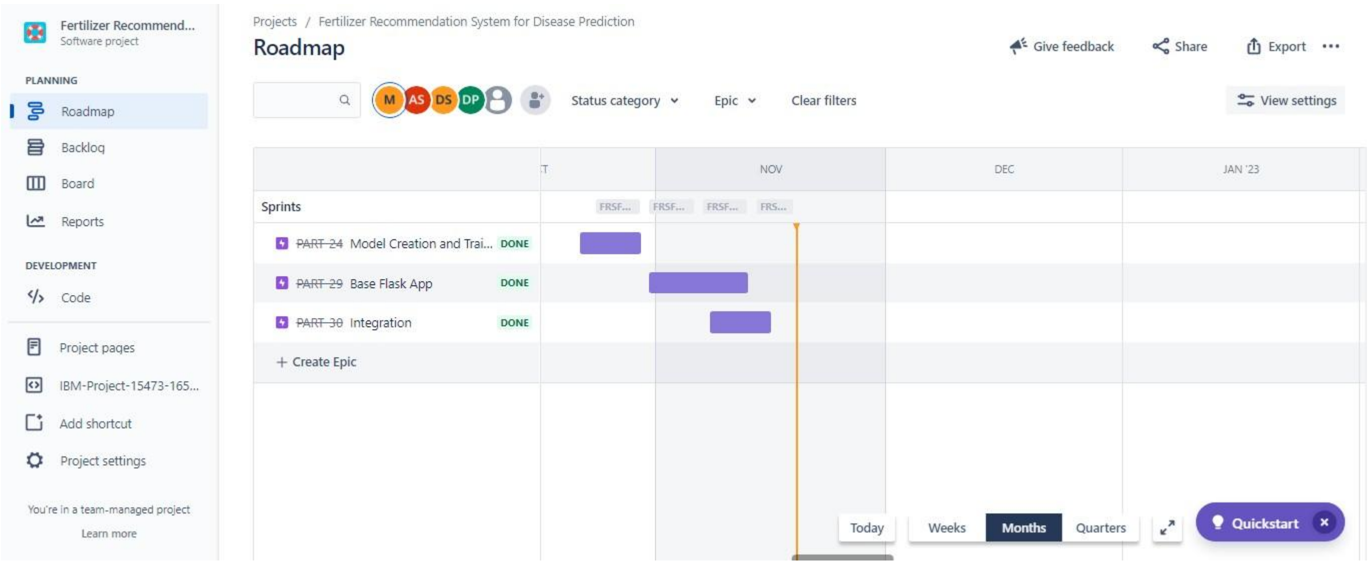
~~PART-23~~ Collectiong Dataset DONE
~~PART-25~~ Model Creation and Trai... DONE
~~PART-33~~ Testing and Documenta... DONE
+ Create Epic



~~PART-30~~ Integration DONE

~~PART-31~~ Conteinerization DONE

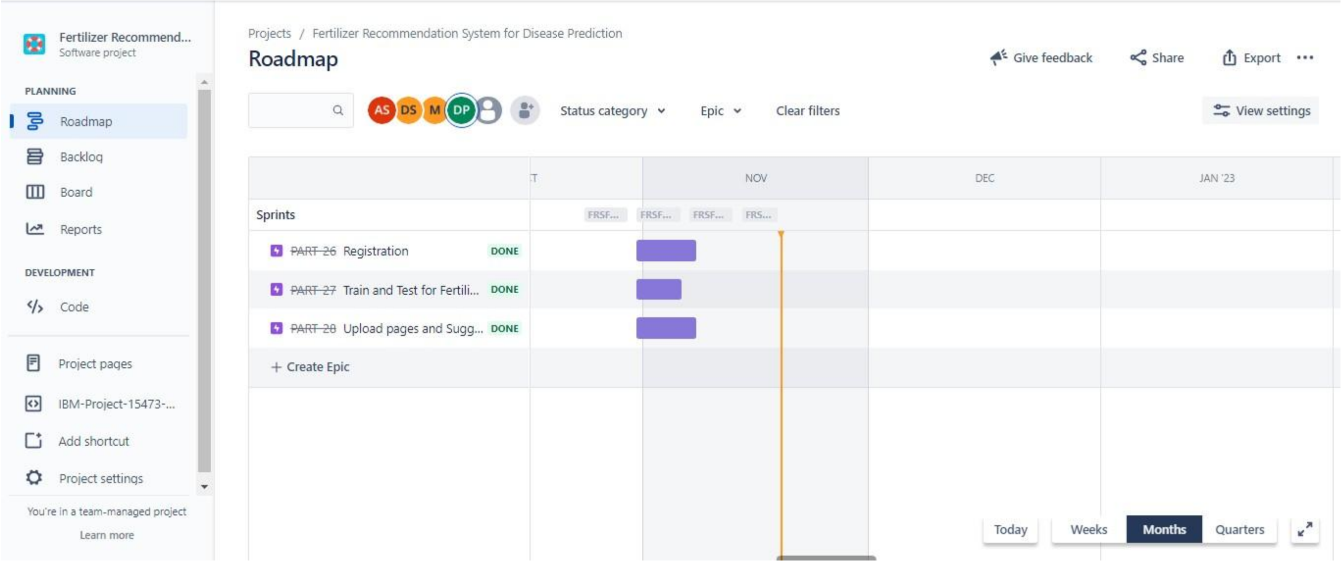
~~PART-32~~ Dashboard DONE



~~PART-24~~ Model Creation and Trai... DONE

~~PART-29~~ Base Flask App DONE

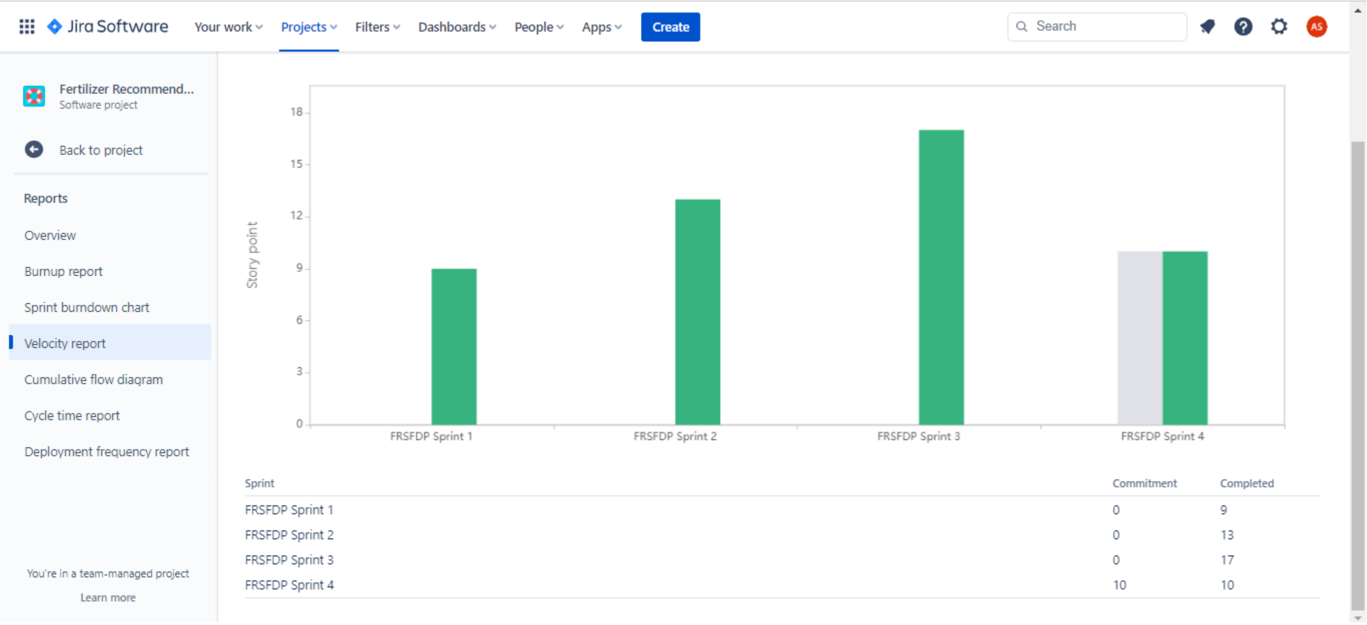
~~PART-30~~ Integration DONE



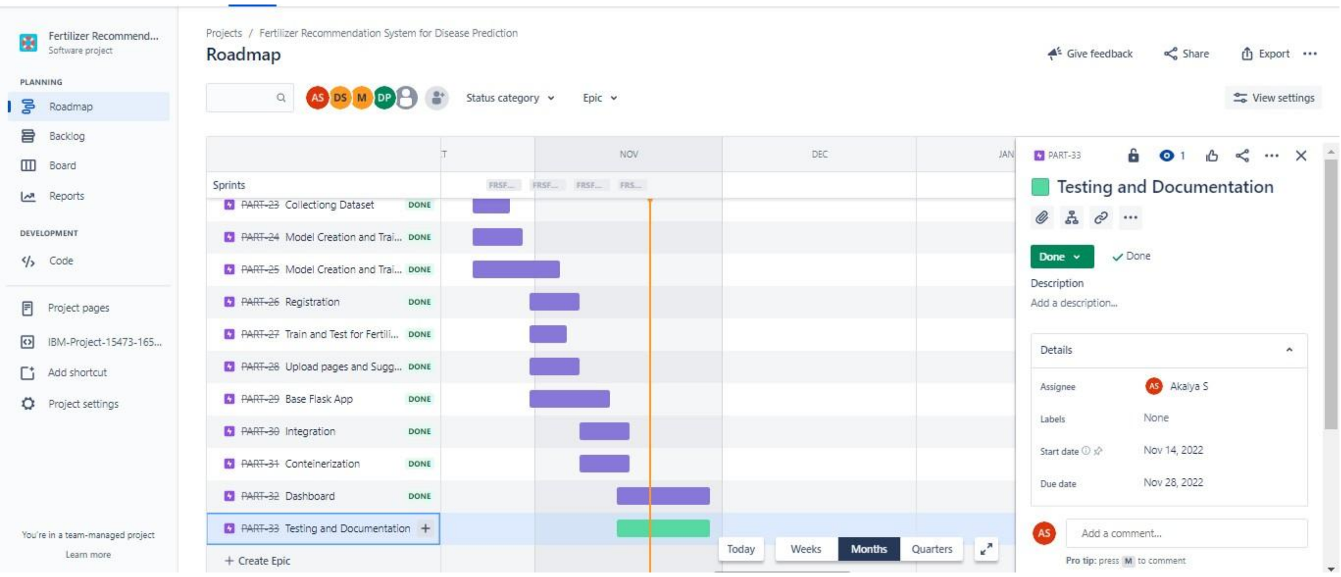
~~PART-26~~ Registration DONE

~~PART-27~~ Train and Test for Fertili... DONE

~~PART-28~~ Upload pages and Sugg... DONE



Sprint	Commitment	Completed
FRSFDP Sprint 1	0	9
FRSFDP Sprint 2	0	13
FRSFDP Sprint 3	0	17
FRSFDP Sprint 4	10	10



Projects / Fertilizer Recommendation System for Disease Prediction

Roadmap

Sprints

~~PART-23~~	Collectiong Dataset	DONE
~~PART-24~~	Model Creation and Trai...	DONE
~~PART-25~~	Model Creation and Trai...	DONE
~~PART-26~~	Registration	DONE
~~PART-27~~	Train and Test for Fertili...	DONE
~~PART-28~~	Upload pages and Sugg...	DONE
~~PART-29~~	Base Flask App	DONE
~~PART-30~~	Integration	DONE
~~PART-31~~	Conteinerization	DONE
~~PART-32~~	Dashboard	DONE
~~PART-33~~	Testing and Documentation	

+ Create Epic

PART-33

Testing and Documentation

Done ✓ Done

Description

Add a description...

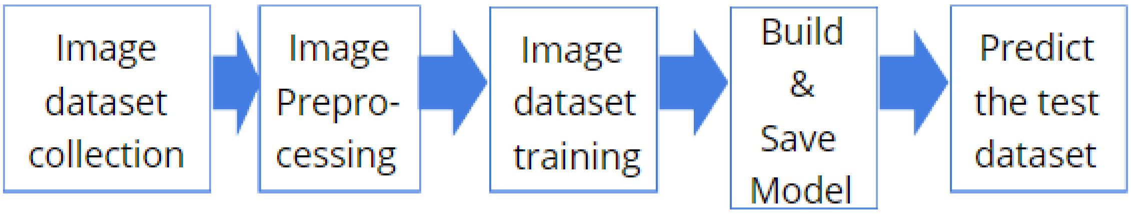
Details

Assignee	AS Akalya S
Labels	None
Start date	Nov 14, 2022
Due date	Nov 28, 2022

AS Add a comment...

Pro tip: press M to comment

7.CODING AND SOLUTIONING



The block diagram of the entire project is shown in figure. First step is the image dataset collection followed by image preprocessing. The third step is the training of image datasets with initializing different hyper parameters. Then build the model and save the model file with .pth format. The final stage is the testing of existing or new datasets using the trained model.

HARDWARE/SOFTWARE DESIGNING

The software used for training and testing the dataset is Python. The Jupyter notebook (Notebook of IBM cloud also) is used for python programming. The neural network used for training and testing the model is Convolutional Neural Network(CNN). The CNN has following layers:

- Convolutional layer (32x32 kernal (3x3))
- Max-pool layer (kernel(2x2))
- Flatten layer
- Dense layer (different layers with different size)
- Final output dense layer(size 6x1 for fruit dataset and 9x1 for Vegetable dataset)

In the preprocessing step, images are normalized to 1 and then resized to 128x128. The images are arranged in different batch sizes. Then train set and test set are formed from the collected datasets. In order to do the above steps in Python, the following Python libraries must be imported before starting the process:

- ➢ Numpy
- ➢ TensorFlow
- ➢ Keras
- ➢ Matplotlib (optional for data visualization)

The following activation functions used in the CNN training:

- RELU at the end of convolution layer and Max Pool layer
- Softmax at the end of output dense layer
- For testing the dataset argumax is used, its an optional

DATASETLINK:

Dataset: https://www.kaggle.com/datasets/vipoooool/new-plant-diseases-dataset

IMAGE PREPROSSESSING

Image preprocessing is the steps taken to format images before they are used by model training and inference. This includes, but is not limited to, resizing, orienting, and color corrections.Image preprocessing may also decrease model training time and increase model inference speed. If input images are particularly large, reducing the size of these images will dramatically improve model training time without significantly reducing model performance. For example, the standard size of images on iPhone 11 is 3024×4032. The machine learning model Apple uses to create masks and apply Portrait Mode performs on images half this size before its output is rescaled back to full size.

Preprocessing is an essential step to clean image data before it is ready to be used in a computer vision model. There are both technical and performance reasons why preprocessing is essential.Fully connected layers in convolution neural networks, a common architecture in computer vision, require that all images are the same sized arrays. If your images are not in the same size, your model may not perform as expected. If you are building a model in code using a library like Tensor flow, you are likely to encounter an error if your image is not the same size.

FEATURE 1

IMAGE PREPOSSESSING CODING FOR FRUIT

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

```
  train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)


test_datagen = ImageDataGenerator(rescale = 1./255)
training_set =
train_datagen.flow_from_directory('/content/drive/MyDrive/plant_dataset/fruit_
dataset/train',target_size=(224,224),batch_size=32,class_mode='categorical')
test_set =
train_datagen.flow_from_directory('/content/drive/MyDrive/plant_dataset/fruit_
dataset/test',target_size=(224,224),batch_size=32,class_mode='categorical')
```

IMAGE PREPROSSESSING CODING FOR VEGETABLE

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


test_datagen = ImageDataGenerator(rescale = 1./255)
training_set =
train_datagen.flow_from_directory('/content/drive/MyDrive/plant_dataset/fruit_
dataset/train',target_size=(224,224),batch_size=32,class_mode='categorical')
```

```
test_set = train_datagen.flow_from_directory('/content/drive/MyDrive/plant_dataset/fruit_dataset/test',target_size=(224,224),batch_size=32,class_mode='categorical')
```

MODEL BULIDING:

```
    ls
   cd /content/drive/MyDrive/full_dataset
   ls
```

Mounting google drive

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

Installing torch

```
 !pip install torchsummary
```

Import necessary libraries

```
import os
import numpy as np
import pandas as pd
import torch
import matplotlib.pyplot as plt
import torch.nn as nn
from torch.utils.data import DataLoader
from PIL import Image
import torch.nn.functional as F
from torchvision import datasets, transforms, models
import torchvision.models as models
from torchvision.utils import make_grid
from torchvision.datasets import ImageFolder
from torchsummary import summary
from matplotlib.ticker import FormatStrFormatter
%matplotlib inline
```

Unzip the file

```
!unzip archive.zip
```

Exploring the data

```
data_dir = "/content/drive/MyDrive/full_dataset/New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)"
train_dir = data_dir + "/train"
valid_dir = data_dir + "/valid"
diseases_plant = os.listdir(train_dir)
diseases_plant
plants = []
No_Of_Diseases = 0
for plant in diseases_plant:
    if plant.split('___')[0] not in plants:
        plants.append(plant.split('___')[0])
    if plant.split('___')[0] != 'healthy':
        No_Of_Diseases += 1
print(plants)
len(plants)
format(No_Of_Diseases)
nums = {}
for disease in diseases_plant:
    nums[disease] = len(os.listdir(train_dir + '/' + disease))
    img_per_class = pd.DataFrame(nums.values(), index=nums.keys(), columns=["noumber_of_images"])
img_per_class
X_train = 0
for value in nums.values():
    X_train += value
print(X_train)
```

datasets for validation and training

```
train = ImageFolder(train_dir, transform=transforms.ToTensor())
valid = ImageFolder(valid_dir, transform=transforms.ToTensor())
img, label = train[0]
print(img.shape, label)
len(train.classes)
```

for checking some images from training dataset

```
def show_image(image, label):
    print("Label :" + train.classes[label] + "(" + str(label) + ")")
    plt.imshow(image.permute(1, 2, 0))
show_image(*train[101])
show_image(*train[30500])
show_image(*train[50000])
random_seed = 7
torch.manual_seed(random_seed)
```

setting the batch size

```
batch_size = 32
```

DataLoaders for training and validation

```
train_dl = DataLoader(train, batch_size, shuffle=True, num_workers=2, pin_memory=True)
```

```
valid_dl = DataLoader(valid, batch_size, num_workers=2, pin_memory=True)
def show_batch(data):
    for images, labels in data:
        fig, ax = plt.subplots(figsize=(30, 30))
        ax.set_xticks([]); ax.set_yticks([])
        ax.imshow(make_grid(images, nrow=8).permute(1, 2, 0))
        break
```

Images for first batch of training

```
show_batch(train_dl)
```

for moving data into GPU (if available)

```
def get_default_device():
    """Pick GPU if available, else CPU"""
    if torch.cuda.is_available:
        return torch.device("cuda")
    else:
        return torch.device("cpu")


#for moving data to device (CPU or GPU)
def to_device(data, device):
    """Move tensor(s) to chosen device"""
    if isinstance(data, (list,tuple)):
        return [to_device(x, device) for x in data]
    return data.to(device, non_blocking=True)


# for loading in the device (GPU if available else CPU)
class DeviceDataLoader():
    """Wrap a dataloader to move data to a device"""
    def___init__(self, dl, device):
        self.dl = dl
        self.device = device

    def___iter__(self):
        """Yield a batch of data after moving it to device"""
        for b in self.dl:
            yield to_device(b, self.device)

    def___len__(self):
        """Number of batches"""
        return len(self.dl)
device = get_default_device()
device
# Moving data into GPU
train_dl = DeviceDataLoader(train_dl, device)
valid_dl = DeviceDataLoader(valid_dl, device)
class SimpleResidualBlock(nn.Module):
    def___init__(self):
        super().__init__()
        self.conv1 = nn.Conv2d(in_channels=3, out_channels=3, kernel_size=3,
stride=1, padding=1)
        self.relu1 = nn.ReLU()
        self.conv2 = nn.Conv2d(in_channels=3, out_channels=3, kernel_size=3,
stride=1, padding=1)
        self.relu2 = nn.ReLU()
    def forward(self, x):
        out = self.conv1(x)
        out = self.relu1(out)
        out = self.conv2(out)
        return self.relu2(out) + x
# for calculating the accuracy
def accuracy(outputs, labels):
    _, preds = torch.max(outputs, dim=1)
    return torch.tensor(torch.sum(preds == labels).item() / len(preds))
```

base class for the model

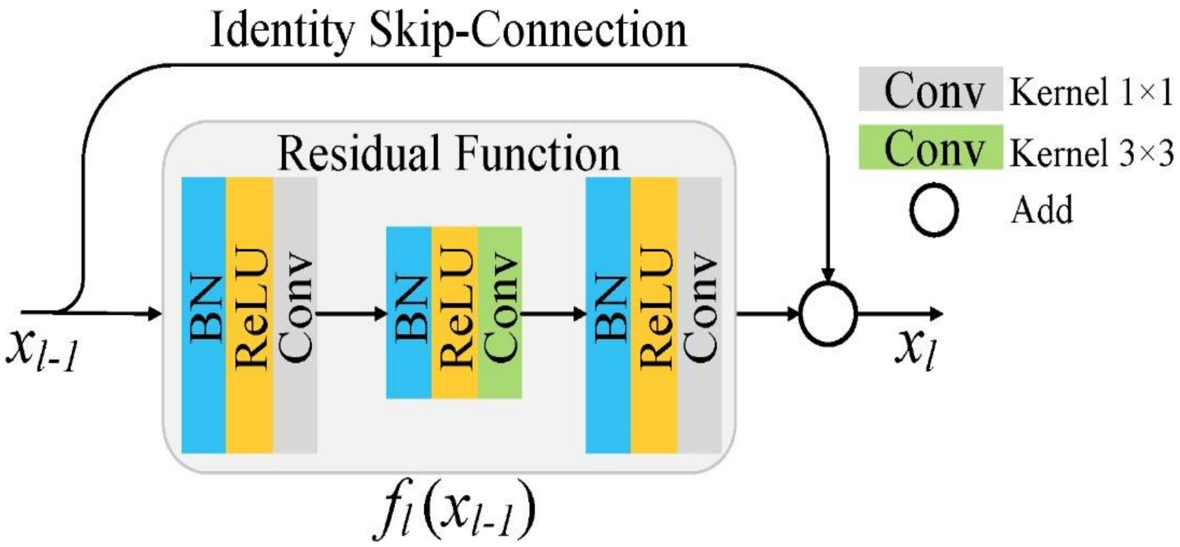
```
class ImageClassificationBase(nn.Module):
    def training_step(self, batch):
        images, labels = batch
        out = self(images)                  # Generate predictions
        loss = F.cross_entropy(out, labels) # Calculate loss
        return loss
    def validation_step(self, batch):
        images, labels = batch
        out = self(images)                   # Generate prediction
        loss = F.cross_entropy(out, labels)  # Calculate loss
        acc = accuracy(out, labels)          # Calculate accuracy
        return {"val_loss": loss.detach(), "val_accuracy": acc}
    def validation_epoch_end(self, outputs):
        batch_losses = [x["val_loss"] for x in outputs]
        batch_accuracy = [x["val_accuracy"] for x in outputs]
        epoch_loss = torch.stack(batch_losses).mean()       # Combine loss
        epoch_accuracy = torch.stack(batch_accuracy).mean()
        return {"val_loss": epoch_loss, "val_accuracy": epoch_accuracy} # Combine accuracies
    def epoch_end(self, epoch, result):
        print("Epoch [{}], last_lr: {:.5f}, train_loss: {:.4f}, val_loss: {:.4f}, val_acc: {:.4f}".format(
            epoch, result['lrs'][-1], result['train_loss'], result['val_loss'], result['val_accuracy']))
```

Architecture for training

```
# convolution block with BatchNormalization
def ConvBlock(in_channels, out_channels, pool=False):
    layers = [nn.Conv2d(in_channels, out_channels, kernel_size=3, padding=1),
              nn.BatchNorm2d(out_channels),
              nn.ReLU(inplace=True)]
    if pool:
        layers.append(nn.MaxPool2d(4))
    return nn.Sequential(*layers)
```

Resnet architecture:

We are going to use ResNet, which have been one of the major breakthrough in computer vision since they were introduced in 2015. In ResNets, unlike in traditional neural networks, each layer feeds into the next layer, we use a network with residual blocks, each layer feeds into the next layer and directly into the layers about 2–3 hops away, to avoid over-fitting (a situation when validation loss stop decreasing at a point and then keeps increasing while training loss still decreases). This also helps in preventing vanishing gradient problem and allow us to train deep neural networks. Here is a simple residual block:



```
class ResNet9(ImageClassificationBase):
    def___init__(self, in_channels, num_diseases):
        super().__init__()
        self.conv1 = ConvBlock(in_channels, 64)
        self.conv2 = ConvBlock(64, 128, pool=True) # out_dim : 128 x 64 x 64
        self.res1 = nn.Sequential(ConvBlock(128, 128), ConvBlock(128, 128))
        self.conv3 = ConvBlock(128, 256, pool=True) # out_dim : 256 x 16 x 16
        self.conv4 = ConvBlock(256, 512, pool=True) # out_dim : 512 x 4 x 44
        self.res2 = nn.Sequential(ConvBlock(512, 512), ConvBlock(512, 512))
        self.classifier = nn.Sequential(nn.MaxPool2d(4),
                                        nn.Flatten(),
                                        nn.Linear(512, num_diseases)
    def forward(self, xb): # xb is the loaded batch
        out = self.conv1(xb)
        out = self.conv2(out)
        out = self.res1(out) + out
        out = self.conv3(out)
        out = self.conv4(out)
        out = self.res2(out) + out
        out = self.classifier(out)
        return out
```

for training

```
@torch.no_grad()
def evaluate(model, val_loader):
    model.eval()
    outputs = [model.validation_step(batch) for batch in val_loader]
    return model.validation_epoch_end(outputs)
def get_lr(optimizer):
    for param_group in optimizer.param_groups:
        return param_group['lr']
def fit_OneCycle(epochs, max_lr, model, train_loader, val_loader, weight_decay=0,
                grad_clip=None, opt_func=torch.optim.SGD):
    torch.cuda.empty_cache()
    history = []
    optimizer = opt_func(model.parameters(), max_lr, weight_decay=weight_decay)
    sched = torch.optim.lr_scheduler.OneCycleLR(optimizer, max_lr, epochs=epochs, steps_per_epoch=len(train_loader))
    for epoch in range(epochs):
        # Training
        model.train()
        train_losses = []
        lrs = []
        for batch in train_loader:
            loss = model.training_step(batch)
            train_losses.append(loss)
            loss.backward()
            # gradient clipping
            if grad_clip:
                nn.utils.clip_grad_value_(model.parameters(), grad_clip)
            optimizer.step()
            optimizer.zero_grad()
          recording and updating learning rates
            lrs.append(get_lr(optimizer))
            sched.step()
        validation
        result = evaluate(model, val_loader)
        result['train_loss'] = torch.stack(train_losses).mean().item()
        result['lrs'] = lrs
        model.epoch_end(epoch, result)
        history.append(result)
    return history
%%time
history = [evaluate(model, valid_dl)]
history
epochs = 2
max_lr = 0.01
grad_clip = 0.1
weight_decay = 1e-4
opt_func = torch.optim.Adam
```

calculating accuracy

```
%%time
history += fit_OneCycle(epochs, max_lr, model, train_dl, valid_dl,
                             grad_clip=grad_clip,
                             weight_decay=1e-4,
                             opt_func=opt_func)
def plot_accuracies(history):
    accuracies = [x['val_accuracy'] for x in history]
    plt.plot(accuracies, '-x')
    plt.xlabel('epoch')
    plt.ylabel('accuracy')
    plt.title('Accuracy vs. No. of epochs')
def plot_lrs(history):
    lrs = np.concatenate([x.get('lrs', []) for x in history])
    plt.plot(lrs)
    plt.xlabel('Batch no.')
    plt.ylabel('Learning rate')
    plt.title('Learning Rate vs. Batch no.');
plot_accuracies(history)
plot_lrs(history)
```

Testing

```
test_dir = "/content/drive/MyDrive/full_dataset/New Plant Diseases Dataset(Augmented)/New Plant Diseases Dataset(Augmented)/valid"
test = ImageFolder(test_dir, transform=transforms.ToTensor())
test_images = sorted(os.listdir(test_dir)) # since images in test folder are in alphabetical order
test_images
def predict_image(img, model):
    """Converts image to array and return the predicted class
        with highest probability"""
    # Convert to a batch of 1
    xb = to_device(img.unsqueeze(0), device)
    yb = model(xb)

    # Pick index with highest probability

    _, preds = torch.max(yb, dim=1)

    # Retrieve the class label

    return train.classes[preds[0].item()]

# predicting first image

img, label = test[0]

plt.imshow(img.permute(1, 2, 0))

print('Label:', test_images[0], ', Predicted:', predict_image(img, model))

# getting all predictions (actual label vs predicted)

for i, (img, label) in enumerate(test):

    print('Label:', test[i], ', Predicted:', predict_image(img, model))
```

saving to the kaggle working directory

```
PATH = 'plant-disease-model.pth'

torch.save(model.state_dict(), PATH)
```

saving the entire model to working directory

```
PATH = '/content/drive/MyDrive/full_dataset/plant-disease-model-complete.pth'

torch.save(model, PATH)
```

FEATURE 2

APPLICATION BUILDING

```
from flask import Flask, render_template, request, Markup
import numpy as np
import pandas as pd
from utils.disease import disease_dic
from utils.fertilizer import fertilizer_dic
import requests
import config
import pickle
import io
import torch
from torchvision import transforms
from PIL import Image
from utils.model import ResNet9
import os
import requests,json
```

-----------------------LOADING THE TRAINED MODELS --------------------

Loading plant disease classification model:

```
disease_classes = ['Apple___Apple_scab',
                  'Apple___Black_rot',
                  'Apple___Cedar_apple_rust',
                  'Apple___healthy',
                  'Blueberry___healthy',
                  'Cherry_(including_sour)___Powdery_mildew',
                  'Cherry_(including_sour)___healthy',
                  'Corn_(maize)___Cercospora_leaf_spot Gray_leaf_spot',
                  'Corn_(maize)___Common_rust_',
                  'Corn_(maize)___Northern_Leaf_Blight',
                  'Corn_(maize)___healthy',
                  'Grape___Black_rot',
                  'Grape___Esca_(Black_Measles)',
                  'Grape___Leaf_blight_(Isariopsis_Leaf_Spot)',
                  'Grape___healthy',
                  'Orange___Haunglongbing_(Citrus_greening)',
                  'Peach___Bacterial_spot',
                  'Peach___healthy',
                  'Pepper,_bell___Bacterial_spot',
                  'Pepper,_bell___healthy',
                  'Potato___Early_blight',
                  'Potato___Late_blight',
                  'Potato___healthy',
                  'Raspberry___healthy',
                  'Soybean___healthy',
                                  'Squash___Powdery_mildew',
                                  'Strawberry___Leaf_scorch',
                                  'Strawberry___healthy',
                                  'Tomato___Bacterial_spot',
                                  'Tomato___Early_blight',
                                  'Tomato___Late_blight',
                                  'Tomato___Leaf_Mold',
                                  'Tomato___Septoria_leaf_spot',
                                  'Tomato___Spider_mites Two-spotted_spider_mite',
                                  'Tomato___Target_Spot',
                                  'Tomato___Tomato_Yellow_Leaf_Curl_Virus',
                                  'Tomato___Tomato_mosaic_virus',
                                  'Tomato___healthy']
                  disease_model_path = r'E:/IBM/main project/full project/app/models/plant-disease-model.pth'
                  disease_model = ResNet9(3, len(disease_classes))
                  disease_model.load_state_dict(torch.load(
                      disease_model_path, map_location=torch.device('cpu')))
                  disease_model.eval()
```

Loading crop recommendation model:

```
crop_recommendation_model_path = 'models/RandomForest.pkl'
crop_recommendation_model = pickle.load(
    open(crop_recommendation_model_path, 'rb'))
```

Custom functions for calculations:

```
def weather_fetch(city_name):
    """
    Fetch and returns the temperature and humidity of a city
    :params: city_name
    :return: temperature, humidity
    """
    api_key = "9d7cde1f6d07ec55650544be1631307e"
    base_url = "http://api.openweathermap.org/data/2.5/weather?"
    complete_url = base_url + "appid=" + api_key + "&q=" + city_name
    response = requests.get(complete_url)
    x = response.json()
    if x["cod"] != "404":
        y = x["main"]
        temperature = round((y["temp"] - 273.15), 2)
        humidity = y["humidity"]
        return temperature, humidity
    else:
        return None
def predict_image(img, model=disease_model):
    """
    Transforms image to tensor and predicts disease label
    :params: image
    :return: prediction (string)
    """
    transform = transforms.Compose([
        transforms.Resize(256),
        transforms.ToTensor(),
    ])
    image = Image.open(io.BytesIO(img))
    img_t = transform(image)
    img_u = torch.unsqueeze(img_t, 0)
```

Get predictions from model:

```
    yb = model(img_u)
    # Pick index with highest probability
    _, preds = torch.max(yb, dim=1)
    prediction = disease_classes[preds[0].item()]
    # Retrieve the class label
    return prediction
```

______________________________FLASK APP ____________________________________

```
app = Flask(__name__)
```

render home page:

```
@ app.route('/')
```

```
def home():
    title = 'Heck of Harvest- Home'
    return render_template('index.html', title=title)
```

render crop recommendation form page:

```
@ app.route('/crop-recommend')
def crop_recommend():
    title = 'Heck of Harvest - Crop Recommendation'
    return render_template('crop.html', title=title)
```

render fertilizer recommendation form page

```
@ app.route('/fertilizer')
def fertilizer_recommendation():
    title = 'Heck of Harvest - Fertilizer Suggestion'
    return render_template('fertilizer.html', title=title)
```

render disease prediction input page

RENDER PREDICTION PAGE

render crop recommendation result page:

```
@ app.route('/crop-predict', methods=['POST'])
def crop_prediction():
    title = 'Heck of Harvest - Crop Recommendation'
    if request.method == 'POST':
        N = int(request.form['nitrogen'])
        P = int(request.form['phosphorous'])
        K = int(request.form['pottasium'])
        ph = float(request.form['ph'])
        rainfall = float(request.form['rainfall'])
        # state = request.form.get("stt")
        city = request.form.get("city")
        if weather_fetch(city) != None:
            temperature, humidity = weather_fetch(city)
            data = np.array([[N, P, K, temperature, humidity, ph, rainfall]])
            my_prediction = crop_recommendation_model.predict(data)
            final_prediction = my_prediction[0]
            return render_template('crop-result.html', prediction=final_prediction, title=title)
        else:
            return render_template('try-again.html', title=title)
```

render fertilizer recommendation result page:

```
@ app.route('/fertilizer-predict', methods=['POST'])
def fert_recommend():
    title = 'Heck of Harvest - Fertilizer Suggestion'
    crop_name = str(request.form['cropname'])
    N = int(request.form['nitrogen'])
```

```
P = int(request.form['phosphorous'])

K = int(request.form['pottasium'])

# ph = float(request.form['ph'])

df = pd.read_csv('data/fertilizer.csv')

nr = df[df['Crop'] == crop_name]['N'].iloc[0]

pr = df[df['Crop'] == crop_name]['P'].iloc[0]

kr = df[df['Crop'] == crop_name]['K'].iloc[0]

n = nr - N

p = pr - P

k = kr - K

temp = {abs(n): "N", abs(p): "P", abs(k): "K"}

max_value = temp[max(temp.keys())]

if max_value == "N":

    if n < 0:

        key = 'NHigh'

    else:

        key = "Nlow"

elif max_value == "P":

    if p < 0:

        key = 'PHigh'

    else:

        key = "Plow"
    else:
        if k < 0:
            key = 'KHigh'
        else:
            key = "Klow"
    response = Markup(str(fertilizer_dic[key]))
    return render_template('fertilizer-result.html', recommendation=response,
title=title)
```

render disease prediction result page:

```
@app.route('/disease-predict', methods=['GET', 'POST'])
def disease_prediction():
    title = 'Heck of Harvest - Disease Detection'
    if request.method == 'POST':
        if 'file' not in request.files:
            return redirect(request.url)
        file = request.files.get('file')
        if not file:
            return render_template('disease.html', title=title)
        try:
            img = file.read()
            prediction = predict_image(img)
            prediction = Markup(str(disease_dic[prediction]))
            return render_template('disease-result.html', prediction=prediction, title=title)
        except:
            pass
    return render_template('disease.html', title=title)
if__name__== '__main__':
    app.debug = True
    app.run()
```

```
a
p
p
.
r
u
n
(
)
```

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 redirect to predict page or not.
- Verify the UI elements in Predict Page
- Verify user is able to select the dropdown value or not.
- Verify user is able to upload the image or not.
- Verify whether the image is predicted correctly or not.

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result St		Commnets	TC Automation(B	ID
HomePage_TC_OO1	Functional	Home Page	Verify user is able to see the home page or not.		1. Enter URL and click go 2. verify whether the user is able to see the home page.	Enter URL and click go	User able to see the home page	Working as expected	Pass	Nil	N	_
HomePage_TC_OO2	UI	Home Page	Verify the UI elements in Home Page		1. Enter URL and click go 2. Verify the UI elements in Home Page.	Enter URL and click go	Application should show below UI elements: Home Tab & Predict Tab	Working as expected	pass	Nil	N	_
PredictPage_TC_OO3	Functional	Predict page	Verify user is able to redirect to predict page or not.		1. Enter URL and click go 2. Click on Predict button 3. Verify whether the user to redirect to predict page or not.	Click the predict button in home page	User should navigate to Predict page	Working as expected	pass	Nil	N	_
PredictPage_TC_OO4	UI	Predict page	Verify the UI elements in Predict Page		1. Enter URL and click go 2. Verify the UI elements in Predict Page.	Click the predict button and redirect to predict page	Application should show below UI elements: Dropdown List , Upload file Button, Predict button.	Working as expected	pass	Nil	N	_
PredictPage_TC_OO5	Functional	Predict page	Verify user is able to select the dropdown value or not.		1. Enter URL and click go 2. Click on Predict button 3. Verify whether the user to redirect to predict page or not. 4.Verify user is able to select the dropdown value or not.	Fruit or Vegetable	Application should shows user to choose fruit or vegetable option in dropdown list.	Working as expected	pass	Nil	N	_
PredictPage_TC_OO6	Functional	Predict page	Verify user is able to upload the image or not.		1. Enter URL and click go 2. Click on Predict button 3. Verify whether the user to redirect to predict page or not. 4.Verify user is able to select the dropdown value or not. 5.Verify user is able to upload the images or not	Images to be Uploaded	Application should shows the uploaded image.	Working as expected	pass	Nil	N	_
PredictPage_TC_OO7	Functional	Predict page	Verify whether the image is predicted correctly or not		1. Enter URL and click go 2. Click on Predict button 3. Verify whether the user to redirect to predict page or not. 4.Verify user is able to select the dropdown value or not. 5. Verify user is able to upload the images or not 6. Verify whether the image is predicted correctly or not	Click the Predict Button	Application shows the predicted output	Working as expected	pass	Nil	N	

8.2 USER ACCEPTANCE TESTING

Defect Analysis

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
Leaf spots	14	4	2	3	19
Mosaic leaf pattern	13	6	3	6	28
Misshapen leaves	2	7	0	1	10
Yellow leaves	11	4	3	20	38
Fruit rots	3	2	1	0	6
Fruit spots	5	3	1	1	10
Blights	4	5	2	1	12
Totals	44	31	13	32	11

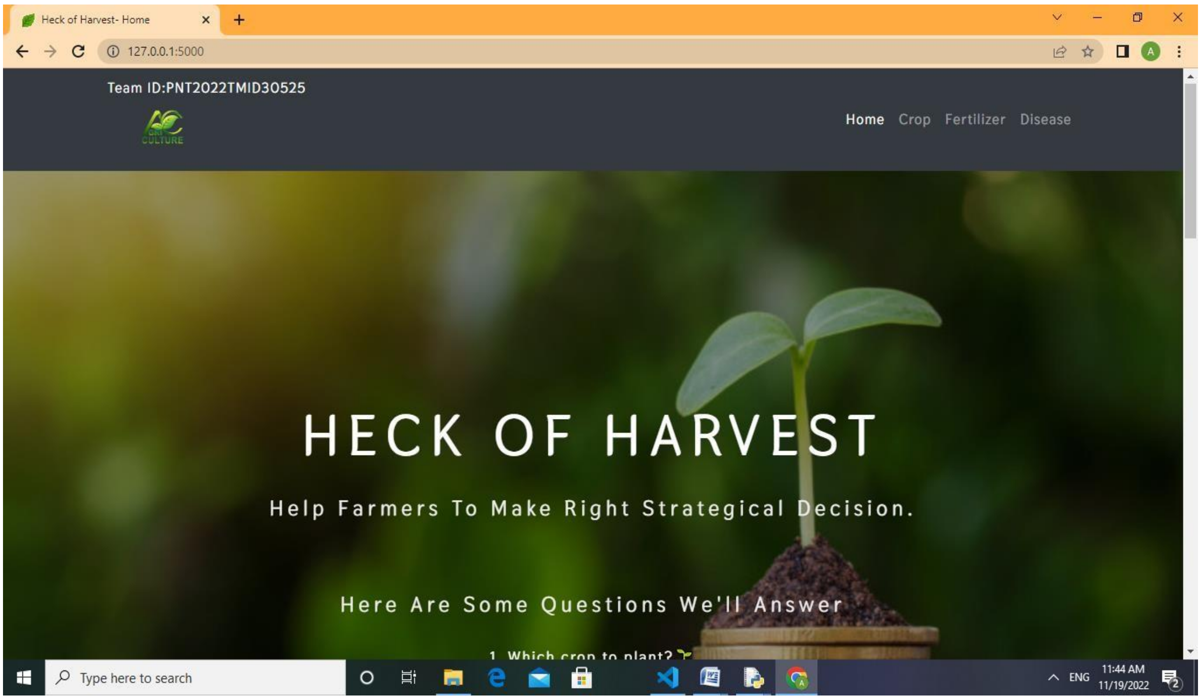
TEST CASE ANALYSIS

Test case analysis helps to understand the number of test case passed failed and untested.

section	Total Cases	Not Tested	Fail	Pass
Leaf spots	17	0	0	17
Mosaic leaf pattern	51	0	0	51
Misshapen leaves	20	0	0	20
Yellow leaves	7	0	0	7
Fruit rots	9	0	0	9
Fruit spots	4	0	0	4
Blights	2	0	0	2

9.RESULTS

HOME PAGE:



Team ID:PNT2022TMID30525

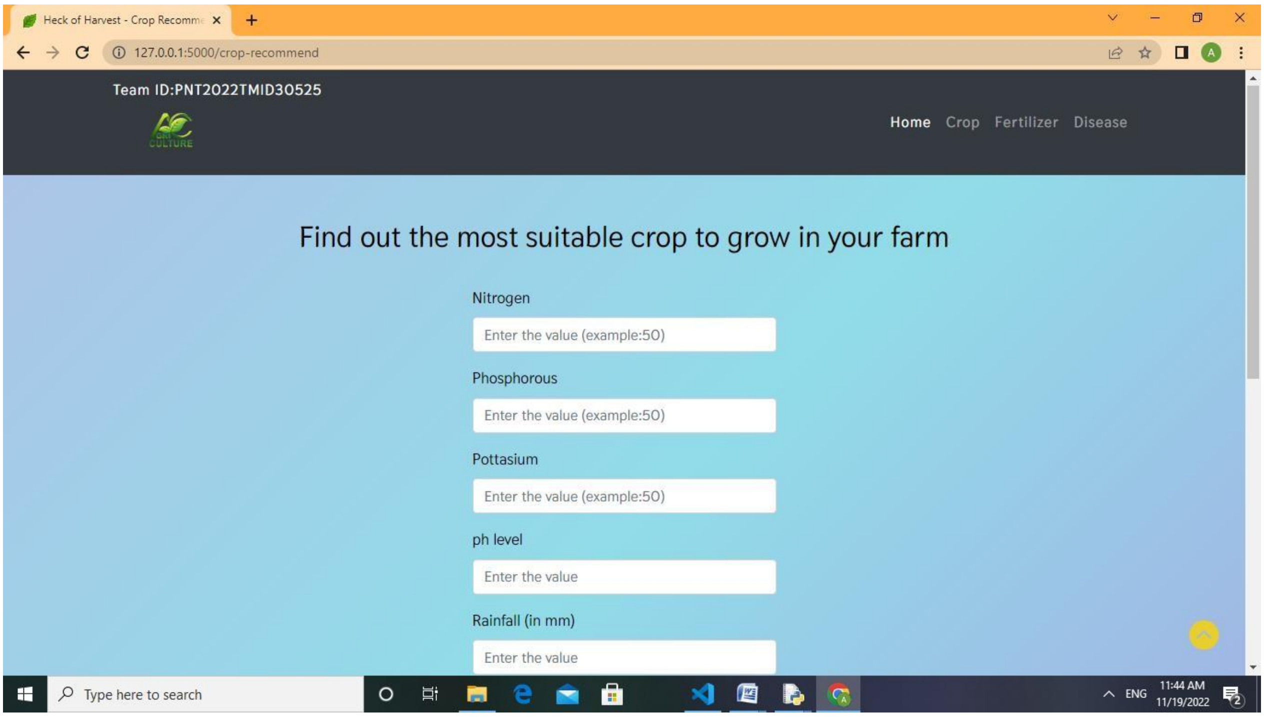
HECK OF HARVEST

Help Farmers To Make Right Strategical Decision.

Here Are Some Questions We'll Answer

1. Which crop to plant?

CROP RECOMMENDATION PAGE:



Team ID:PNT2022TMID30525

Find out the most suitable crop to grow in your farm

Nitrogen

Enter the value (example:50)

Phosphorous

Enter the value (example:50)

Pottasium

Enter the value (example:50)

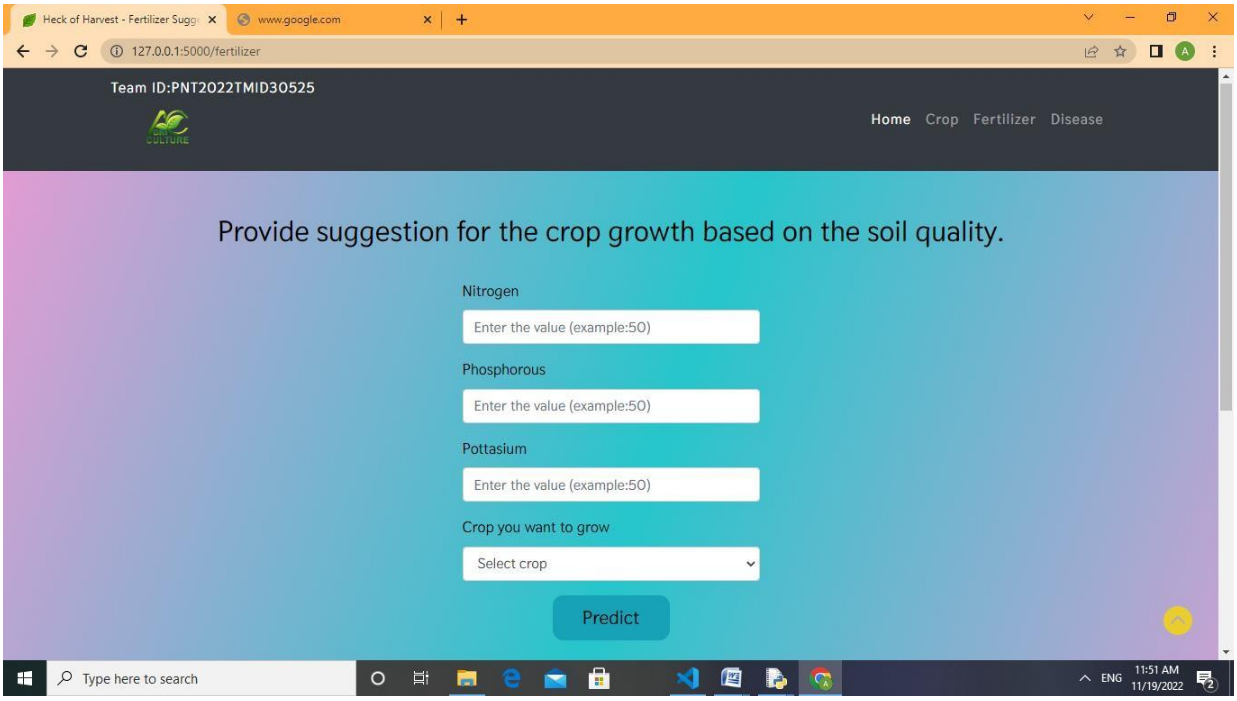
ph level

Enter the value

Rainfall (in mm)

Enter the value

FERTILIZER RECOMMENDATION PAGE:



Provide suggestion for the crop growth based on the soil quality.

Nitrogen

Enter the value (example:50)

Phosphorous

Enter the value (example:50)

Pottasium

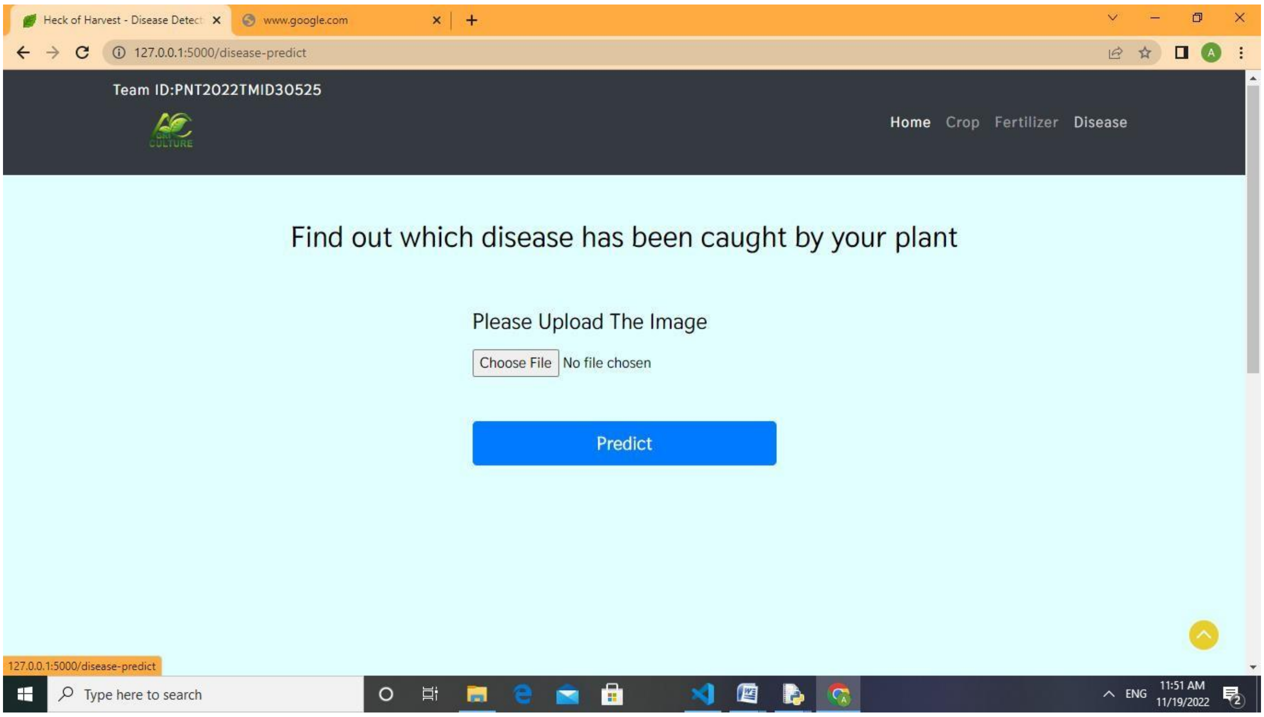
Enter the value (example:50)

Crop you want to grow

Select crop

Predict

DISEASE PREDICTION:



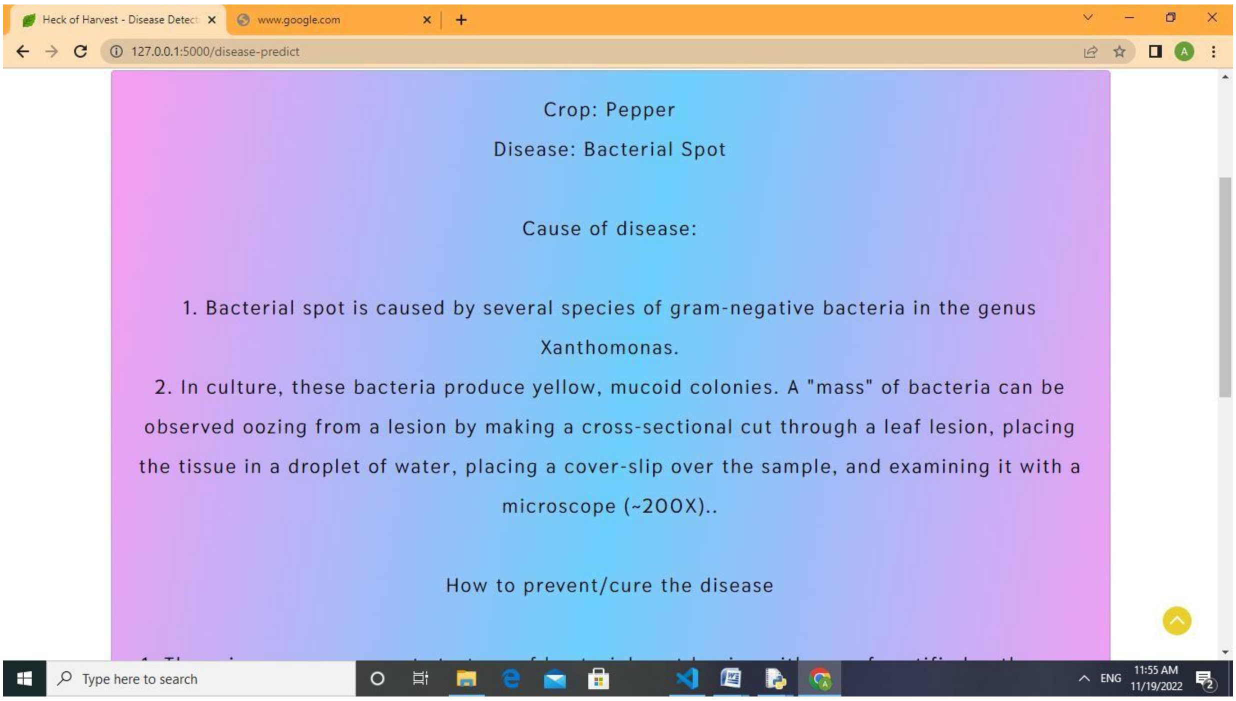
Find out which disease has been caught by your plant

Please Upload The Image

Choose File No file chosen

Predict

DISEASE PREDICTION RESULT:


Heck of Harvest - Disease Detect
www.google.com
127.0.0.1:5000/disease-predict
Type here to search
11:55 AM
11/19/2022

Crop: Pepper

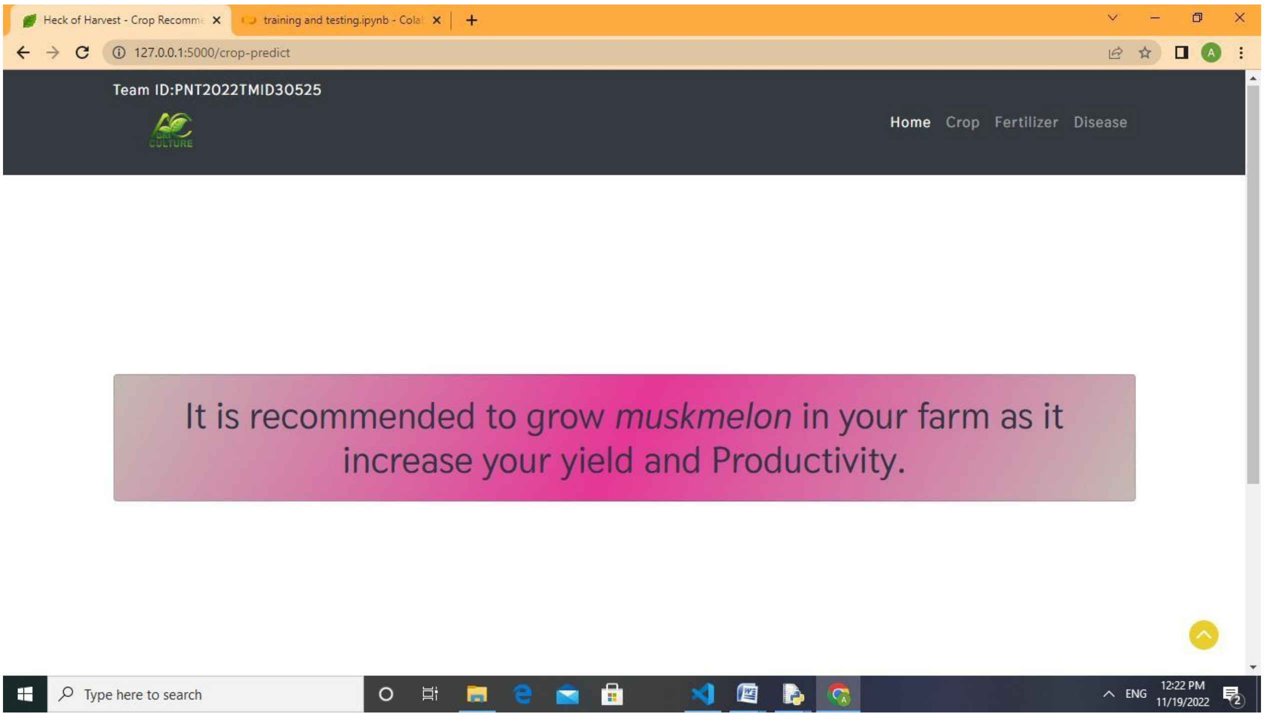
Disease: Bacterial Spot

Cause of disease:

1. Bacterial spot is caused by several species of gram-negative bacteria in the genus Xanthomonas.
2. In culture, these bacteria produce yellow, mucoid colonies. A "mass" of bacteria can be observed oozing from a lesion by making a cross-sectional cut through a leaf lesion, placing the tissue in a droplet of water, placing a cover-slip over the sample, and examining it with a microscope (~200X)..

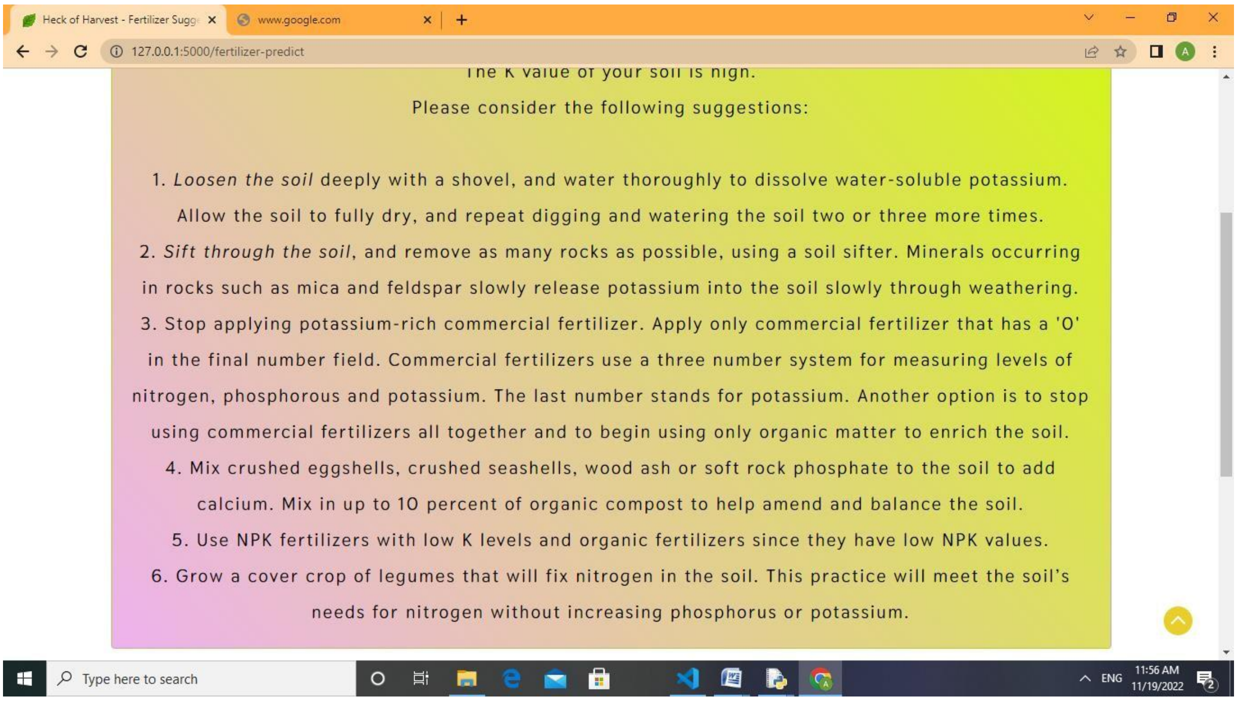
How to prevent/cure the disease

CROP RECOMMENDATION RESULT:



It is recommended to grow *muskmelon* in your farm as it increase your yield and Productivity.

FERTILIZER RECOMMENDATION RESULT:


Heck of Harvest - Fertilizer Sugg
www.google.com
127.0.0.1:5000/fertilizer-predict
Type here to search
ENG
11:56 AM
11/19/2022

The K value of your soil is high.

Please consider the following suggestions:

1. *Loosen the soil* deeply with a shovel, and water thoroughly to dissolve water-soluble potassium. Allow the soil to fully dry, and repeat digging and watering the soil two or three more times.
2. *Sift through the soil*, and remove as many rocks as possible, using a soil sifter. Minerals occurring in rocks such as mica and feldspar slowly release potassium into the soil slowly through weathering.
3. Stop applying potassium-rich commercial fertilizer. Apply only commercial fertilizer that has a '0' in the final number field. Commercial fertilizers use a three number system for measuring levels of nitrogen, phosphorous and potassium. The last number stands for potassium. Another option is to stop using commercial fertilizers all together and to begin using only organic matter to enrich the soil.
4. Mix crushed eggshells, crushed seashells, wood ash or soft rock phosphate to the soil to add calcium. Mix in up to 10 percent of organic compost to help amend and balance the soil.
5. Use NPK fertilizers with low K levels and organic fertilizers since they have low NPK values.
6. Grow a cover crop of legumes that will fix nitrogen in the soil. This practice will meet the soil's needs for nitrogen without increasing phosphorus or potassium.

9.1.PERFORMANCE METRICS

PERFORMANCE EVALUATION:

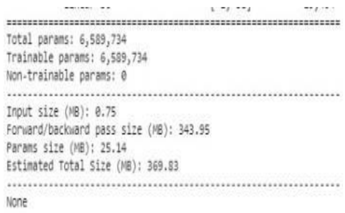
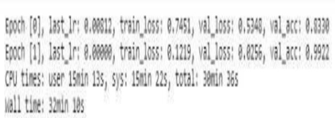
Performance testing is the practice of evaluating how a system performs in terms of responsiveness and stability under a particular workload. Performance tests are typically executed to examine speed, robustness, reliability, and application size.

The performance tests you run will help ensure your software meets the expected levels of service and provide a positive user experience. They will highlight improvements you should make to your applications relative to speed, stability, and scalability before they go into production. Applications released to the public in absence of testing might suffer from different types of problems that lead to a damaged brand reputation, in some cases, irrevocably.

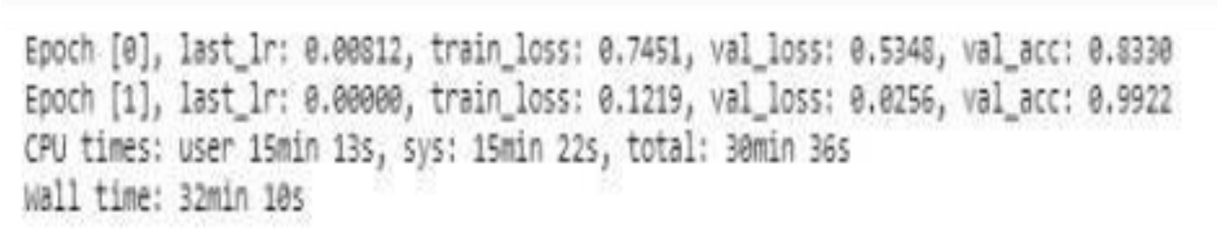
The adoption, success, and productivity of applications depends directly on the proper implementation of performance testing.

While resolving production performance problems can be extremely expensive, the use of a continuous optimization performance testing strategy is key to the success of an effective overarching digital strategy.

Thus it helps in our project to detect the speed and accuracy of different data. In this we have deployed our project in IBM Watson cloud while the performance testing helps to evaluate the robustness and scalability.

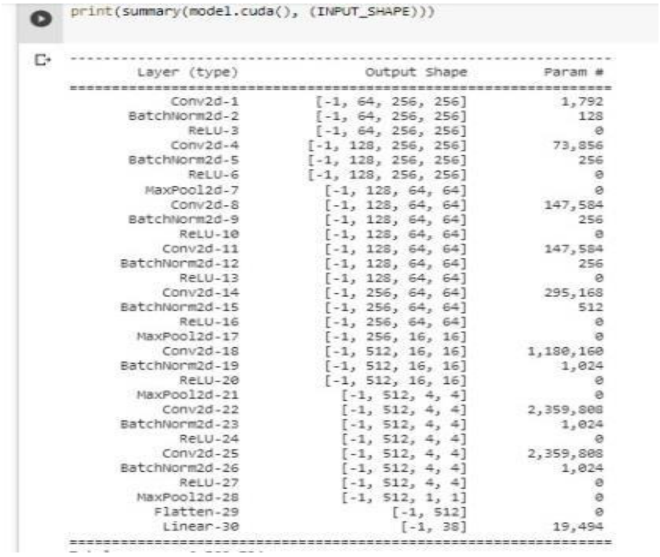
S.No.	**Parameter**	**Values**	**Screenshot**
1.	Model Summary	Total params:6,589,734 Trainable params: 6,589,734 Non-Trainable params:0	 Total params: 6,589,734 Trainable params: 6,589,734 Non-trainable params: 0 Input size (MB): 0.75 Forward/backward pass size (MB): 343.95 Params size (MB): 25.14 Estimated Total Size (MB): 369.83 None
2.	Accuracy	Training Accuracy – 83% Validation Accuracy -99%	 Epoch [0], last_lr: 0.00812, train_loss: 0.7451, val_loss: 0.5348, val_acc: 0.8330 Epoch [1], last_lr: 0.00000, train_loss: 0.1219, val_loss: 0.0256, val_acc: 0.9922 CPU times: user 15min 13s, sys: 15min 22s, total: 30min 36s Wall time: 32min 10s

ACCURACY:



```
Epoch [0], last_lr: 0.00812, train_loss: 0.7451, val_loss: 0.5348, val_acc: 0.8330
Epoch [1], last_lr: 0.00000, train_loss: 0.1219, val_loss: 0.0256, val_acc: 0.9922
CPU times: user 15min 13s, sys: 15min 22s, total: 30min 36s
Wall time: 32min 10s
```

MODEL SUMMARY:



```
print(summary(model.cuda(), (INPUT_SHAPE)))
```

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1        [-1, 64, 256, 256]           1,792
       BatchNorm2d-2        [-1, 64, 256, 256]             128
              ReLU-3        [-1, 64, 256, 256]               0
            Conv2d-4       [-1, 128, 256, 256]          73,856
       BatchNorm2d-5       [-1, 128, 256, 256]             256
              ReLU-6       [-1, 128, 256, 256]               0
         MaxPool2d-7         [-1, 128, 64, 64]               0
            Conv2d-8         [-1, 128, 64, 64]         147,584
       BatchNorm2d-9         [-1, 128, 64, 64]             256
             ReLU-10         [-1, 128, 64, 64]               0
           Conv2d-11         [-1, 128, 64, 64]         147,584
      BatchNorm2d-12         [-1, 128, 64, 64]             256
             ReLU-13         [-1, 128, 64, 64]               0
           Conv2d-14         [-1, 256, 64, 64]         295,168
      BatchNorm2d-15         [-1, 256, 64, 64]             512
             ReLU-16         [-1, 256, 64, 64]               0
        MaxPool2d-17         [-1, 256, 16, 16]               0
           Conv2d-18         [-1, 512, 16, 16]       1,180,160
      BatchNorm2d-19         [-1, 512, 16, 16]           1,024
             ReLU-20         [-1, 512, 16, 16]               0
        MaxPool2d-21           [-1, 512, 4, 4]               0
           Conv2d-22           [-1, 512, 4, 4]       2,359,808
      BatchNorm2d-23           [-1, 512, 4, 4]           1,024
             ReLU-24           [-1, 512, 4, 4]               0
           Conv2d-25           [-1, 512, 4, 4]       2,359,808
      BatchNorm2d-26           [-1, 512, 4, 4]           1,024
             ReLU-27           [-1, 512, 4, 4]               0
        MaxPool2d-28           [-1, 512, 1, 1]               0
          Flatten-29                  [-1, 512]               0
           Linear-30                   [-1, 38]          19,494
================================================================
```

ADVANTAGES:

The system comes with a model to be precise and accurate in predicting crop yield and deliver the end user with proper recommendations about required fertilizer ratio based on atmospheric and soil parameters of the land which enhance to increase the crop yield and increase farmer revenue. The prediction of crop yield based on location and proper implementation of algorithms have proved that the higher crop yield can be achieved. From above work I conclude that for soil classification Random Forest is good with accuracy 86.35% compare to Support Vector Machine. For crop yield prediction Support Vector Machine is good with accuracy 99.47% compare to Random Forest algorithm. The work can be extended further to add following functionality. Mobile application can be build to help farmers by uploading image of farms. Crop diseases detection using image processing in which user get pesticides based on disease images. Implement Smart Irrigation System for farms to get higher yield.

- Fertilizers have all nutrients required for plants growth.
- It is soluble and easily absorbed by plants.
- It enhances the metabolism of plants.
- It is easily available in the market.
- Highly needed for large production.

DISADVANTAGES:

- Fertilizers are more expensive than manure.
- Over fertilization can damage the plants.
- It is toxic and can harm humans.
- It affected the environment and echo system.
- Long term use reduce soil quality.

CONCLUSIONS:

The core strategy of this project is to predict the crop based on the soil nutrient content and the location where the crop is growing. This system will help he farmers to choose the right crop for their land and to give the suitable amount of fertilizer to produce the maximum yield. The Support Vector Machine algorithm helps to predict the crop the precisely based on the pre-processed crop data. This system will also help the new comers to choose the crop which will grow in their area and produce them a good profit.

A decent amount of profit will attract more people towards the agriculture. Also, the crop growth is based on the climate conditions in the particular area and the seasonal monsoons happens now are unpredictable, hence it is easy for the farmers when the prediction result is also based on the climatic conditions. Live weather prediction will also help the users to predict the crop water needs and also it will help the farmers to decrease the crop damage due to the rain or drought.

The prediction of crop yield based on soil data and proper implementation of algorithms have proved that a higher crop yield can be achieved. From the above work, we conclude that for soil classification Random Forest is a suitable algorithm with an accuracy of 99.09% compare to Gaussian Naive Bayes. The work can be extended further to add the following functionality. Building a Website can be built to help farmers by uploading an image of farms. Crop diseases detection uses image processing in which users get pesticides based on disease images and Fertilizer prediction based onsoil condition.

By categorizing the soil samples according to the soil type, land type and macro nutrients Nitrogen (N), Phosphorus (P) and Potassium (K) present in the soil the suitable crop along with its appropriate fertilizer is suggested to the agricultural stakeholder. The month in which the yield will be high is also suggested to the user. The yield calculation is also provided for the crop selected by the farmer. The proposedcrop recommendation system provides 82% of accuracy.

FUTURE SCOPE:

The future work is to implement Machine Learning Algorithms like Ensemble Classifiers to predict the crop yield and recommend the crop with appropriate fertilizer. In the existing system only soil characteristics were considered to provide crop recommendations. In the future work the climatic parameters will also be taken into account to provide crop recommendations. Also the method can be extended to include diverse varieties of crop to be cultivated and to analyze it's performance.

This further research is implementing the proposed algorithm with the existing public datasets. Also, various segmentation algorithms can be implemented to improve accuracy. The proposed algorithm can be modified further to identify the disease that affects the various plant organs such as stems and fruits.

DEMO LINK:

https://drive.google.com/file/d/1fbkuCAV9rl2a4oknHqLrLvD3O5qsXd3Z/view?usp=share_link