Deep Learning Model For Detecting Diseases In Tea Leaves

PROJECT REPORT

12. FUTURE SCOPE

13. APPENDIX

****

Project Team Information

Team ID: Team-592384

Team Size: 4

Team Members:

1. **Team Leader:**
   - **Name: Yarlagadda Nitin**
   - **Email: nitin.yarlagadda2021@vitstudent.ac.in**
2. **Team Member:**
   - **Name: Katta Santhosh Kumar Reddy**
   - **Email: kattasanthosh.kumarreddy2021@vitstudent.ac.in**
3. **Team Member:**
   - **Name: Kolli Tushar**
   - **Email: kolli.tushar2021@vitstudent.ac.in**
4. **Team Member:**
   - **Name: Sai Raghavendra Siddhardha Baratam**
   - **Email: raghavendra.siddhardha2021@vitstudent.ac.in**

1. INTRODUCTION

1.1 Project Overview

Project Overview: Our proposed project aims to address the critical challenge of early detection of tea leaf diseases through the implementation of a state-of-the-art deep learning model. The conventional methods of disease detection, primarily reliant on visual inspection, have proven to be time-consuming and prone to inaccuracies. In response to this, our project proposes the development and deployment of a deep learning model based on a Convolutional Neural Network (CNN). This model is designed to identify various diseases affecting tea leaves efficiently and accurately. By leveraging a labelled dataset of tea leaf images encompassing both healthy and diseased specimens, the CNN will be trained to distinguish and categorize different types of diseases. The overarching goal is to streamline the detection process, providing a reliable tool for tea farmers to identify and mitigate diseases promptly.

In this Project we mainly Detected the Healthy Leaves and Leaves with Diseases mainly 'Anthracnose',' algal leaf', 'bird eye spot', 'brown blight', 'Gray light', 'healthy', 'red leaf spot', 'white spot'

1.2 Purpose

Purpose: Our project aims to revolutionize tea leaf disease detection by implementing a specialized deep learning model. Addressing limitations of conventional methods, the model focuses on subsets of diseases like fungal or bacterial infections, utilizing hyperspectral imaging for additional insights. A user-friendly mobile app or web platform ensures easy integration into daily farming operations. The social impact is substantial, reducing crop losses, improving tea quality, and increasing income for farmers, fostering sustainability in the tea industry. The proposed revenue model, based on fees for using the model, ensures project sustainability. Incorporating transfer learning enhances scalability by initializing model weights with pre-trained data, expediting development and enabling adaptation to diverse datasets and new disease types.

2. LITERATURE SURVEY

2.1 Existing Problem: The existing problem revolves around the inefficiencies and limitations inherent in traditional methods of tea leaf disease detection. Current practices, primarily reliant on visual inspection, prove to be time-consuming and prone to inaccuracies. These shortcomings contribute to delayed responses in identifying and addressing diseases, leading to increased crop losses. The urgency of adopting a more advanced and efficient solution is evident, prompting the development of a deep learning model specialized in tea leaf disease detection.

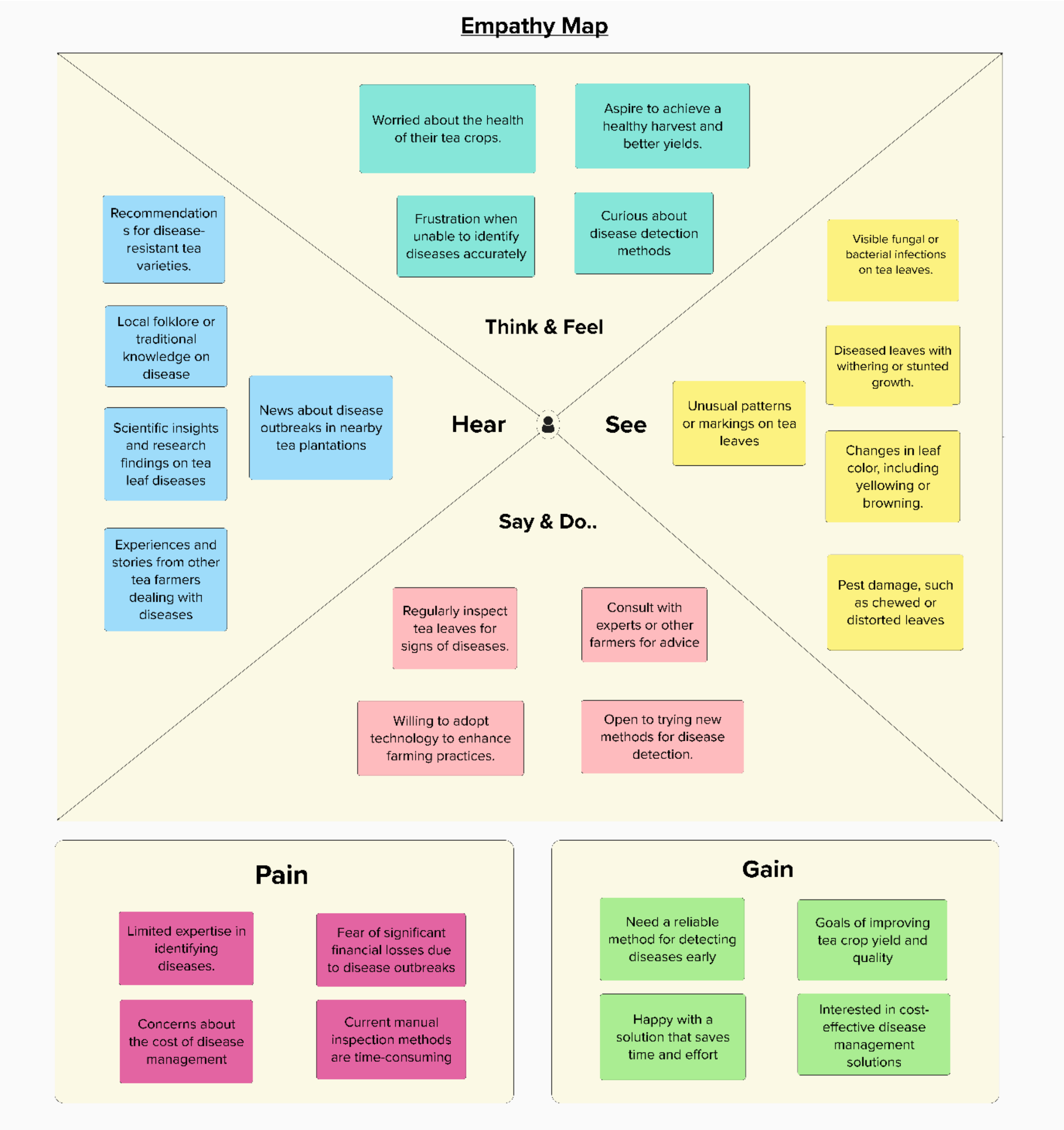
2.2 References: Our proposed project draws inspiration and guidance from a wealth of references in the fields of agriculture, deep learning, and plant pathology. Key literature includes studies on the application of convolutional neural networks (CNNs) in disease detection, advancements in hyperspectral imaging for plant health analysis, and successful implementations of transfer learning in related domains. By synthesizing insights from these references, the project aims to build a robust and innovative solution that incorporates the latest advancements in technology and agricultural sciences.

2.3 Problem Statement Definition: The problem statement for this project is succinctly defined as the need for an improved and efficient method of detecting diseases in tea leaves. Traditional visual inspection methods are insufficient due to their time-consuming nature and susceptibility to inaccuracies. To address this, the project proposes the development and implementation of a deep learning model, specifically a convolutional neural network (CNN), trained on a labelled dataset of tea leaf images. The goal is to create a tool that can accurately and swiftly identify various types of diseases in tea leaves, offering a transformative solution to the existing challenges faced by tea farmers.

3. IDEATION & PROPOSED SOLUTION

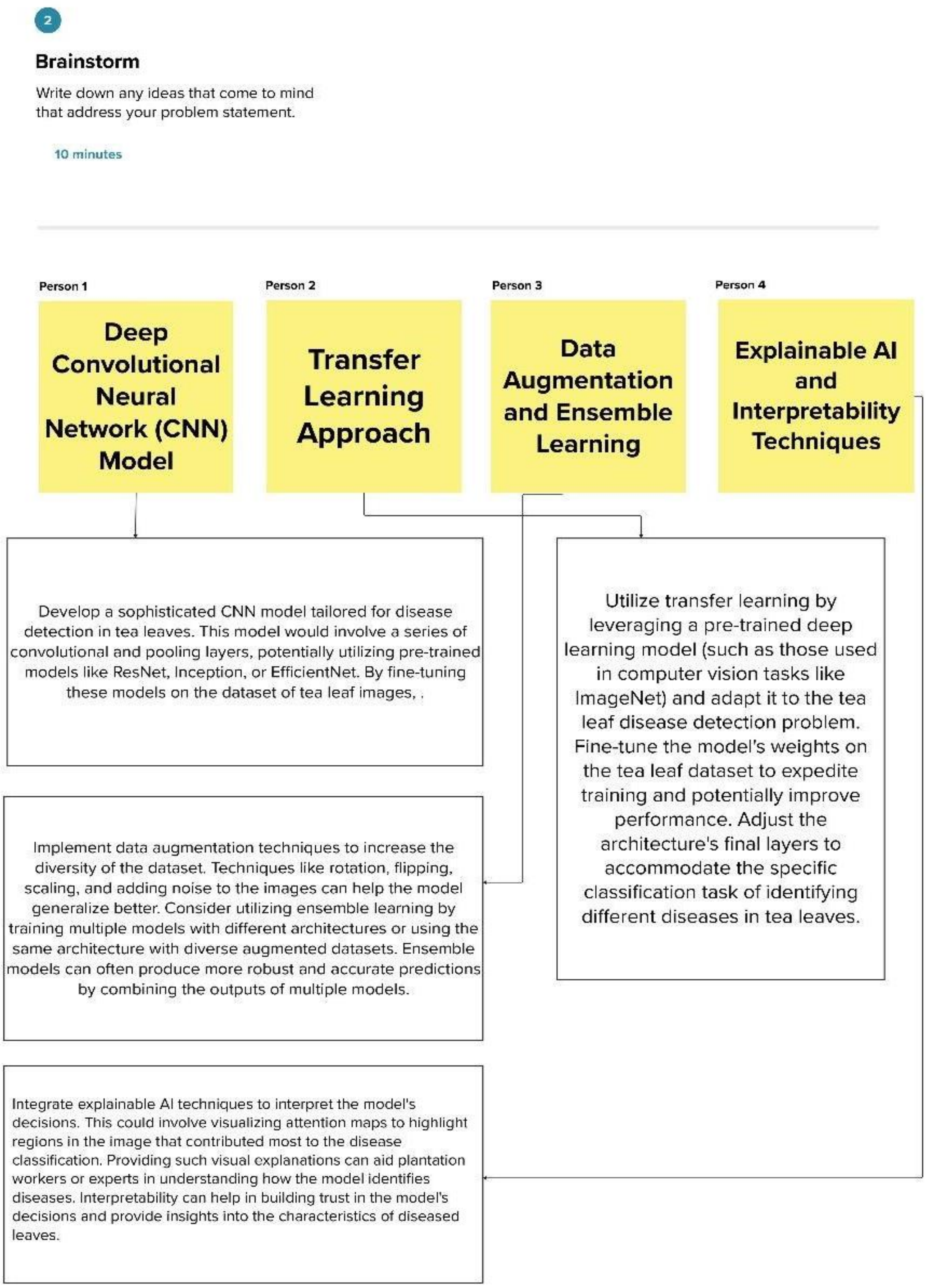
3.1 Empathy Map Canvas

An empathy map, a tool in design thinking, aids project teams by capturing user insights in six key areas: Think & Feel, See, Hear, Say & Do, Pain, and Gain. It enhances understanding of user needs and experiences, informing the design process. By empathizing with users' thoughts, emotions, and actions, the project can be tailored to meet their expectations, ensuring a user-centric approach. Incorporating this tool in project documentation fosters a more comprehensive and successful design and development process.



3.2 Ideation & Brainstorming

Ideation & Brainstorming creates an inclusive and open atmosphere for team members to engage in creative problem-solving. Emphasizing quantity over quality, the process encourages out-of-the-box thinking, fostering the development of innovative ideas. Collaboration is key as participants work together to generate a diverse range of creative solutions, contributing to a rich pool of options for addressing project challenges. This approach, outlined in the project document, promotes a dynamic and collaborative problem-solving environment.



3

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

Integrated Deep Learning Framework for Tea Leaf Disease Detection

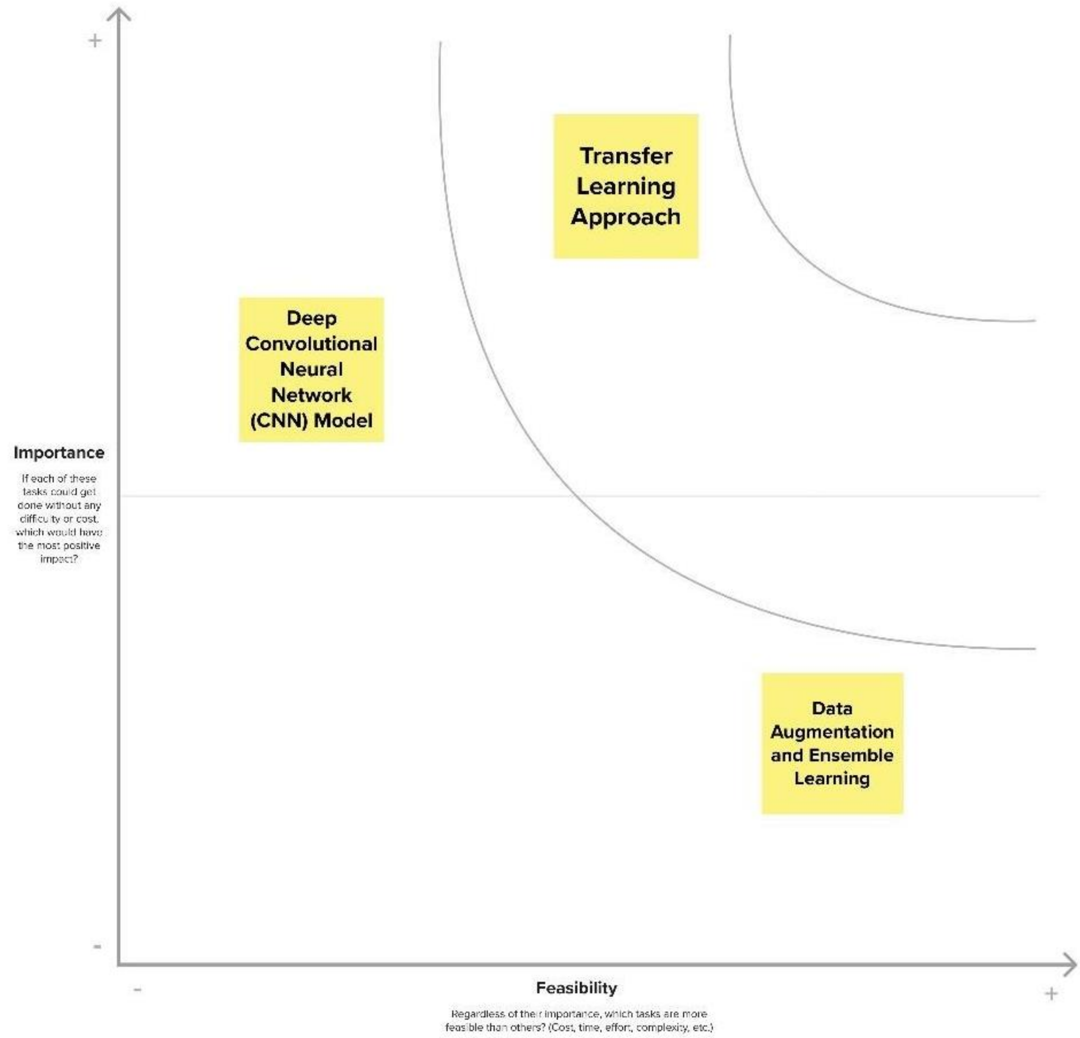
1. **Integrated Transfer Learning with Explainable CNN Model:**
   - **Base Pre-trained Model Selection:** Begin by selecting a pre-trained deep learning model known for image classification, such as ResNet or EfficientNet. These models have demonstrated strong performance on similar tasks.
   - **Fine-tuning and Adaptation:** Modify the pre-trained model's architecture by retraining its final layers to suit the specific problem of tea leaf disease detection. This process involves transfer learning, where the model learns tea leaf disease features while retaining the knowledge obtained from pre-training on generic image datasets.
2. **Data Augmentation and Ensemble Techniques:**
   - **Diverse Data Augmentation:** Augment the tea leaf dataset with various transformations like rotation, flipping, scaling, and adding noise. This augmented data enriches the training set, aiding the model's ability to generalize.
   - **Ensemble Learning:** Train multiple instances of the adapted pre-trained model on different subsets of the augmented dataset. The models can have variations in architectures or be trained on different augmented data. Ensemble models aggregate the predictions from multiple models for improved accuracy and robustness.
3. **Interpretability and Explainability Integration:**
   - **Attention and Interpretability:** Implement techniques to visualize the model's decision-making process. Integrate attention maps or saliency techniques to highlight regions in tea leaf images that contribute most to disease classification. This enables easy interpretation for plantation workers or experts, enhancing trust in the model's decisions.
4. **Continuous Improvement and Iterative Development:**
   - **Performance Monitoring:** Continuously evaluate the model's performance on validation and test datasets. Monitor accuracy, precision, recall, and interpretability metrics.
   - **Iterative Refinement:** Based on performance evaluations, iterate on the model architecture, data augmentation strategies, and interpretability methods to enhance accuracy and usability.

4

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



4. REQUIREMENT ANALYSIS

4.1 Functional Requirements

1. Development Environment Setup:

Requirements:

1.The system should support the installation and configuration of popular deep learning frameworks such as TensorFlow or PyTorch.

2.Provide compatibility with Python for scripting and utilizing machine learning libraries.

3.Ensure that the development environment includes necessary tools for image processing, data manipulation, and model evaluation.

2. Dataset Gathering:

Requirements:

1.The system should facilitate the collection of a diverse dataset containing labeled examples of both healthy and diseased tea leaves.

2.Support mechanisms for adding metadata and annotations to each image to aid in training.

3. Dataset Preprocessing:

Requirements

1. Implement functionality to standardize image sizes for consistent input to the deep learning model.

2. Provide tools for enhancing image quality through normalization and noise reduction.

3. Enable the division of the dataset into training and validation sets for model evaluation.

4. Exploration of Architectures:

Requirements

1. Support the implementation and evaluation of various deep learning architectures, including Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).

2. Requirement: Facilitate the comparison of model performance metrics for informed architecture selection.

5. Model Training:

Requirements:

1. Enable the training of the selected deep learning model using the preprocessed dataset.

2. Implement monitoring tools to assess the model's performance on the validation set during training.

3. Support the adjustment of hyperparameters to optimize model performance.

6. Data Augmentation:

Requirements:

1. Integrate data augmentation techniques such as rotation and flipping during model training to enhance its ability to recognize disease patterns.

2. Allow users to customize augmentation parameters based on their specific dataset characteristics.

7. Deployment as API or Web Service:

Requirements:

1. Facilitate the deployment of the trained model as an API or web service for accessibility.

2. Provide mechanisms for secure and scalable deployment to handle varying user loads.

8. Integration with Web Interface:

Requirements:

1. Develop a user-friendly web interface that allows users to submit tea leaf images for analysis.

2. Integrate the model's API into the web interface for seamless and real-time disease prediction.

9. Thorough Testing:

Requirements:

1. Implement comprehensive testing procedures to identify and report any issues or inaccuracies in both the deep learning model and the web interface.

2. Include unit testing, integration testing, and user acceptance testing to ensure robustness.

10. Fine-Tuning Based on Feedback:

Requirements:

1. Establish a feedback loop for users to report issues and provide feedback on model predictions.

2. Allow administrators to fine-tune hyperparameters based on user feedback and ongoing testing results

4.2 Non-Functional Requirements

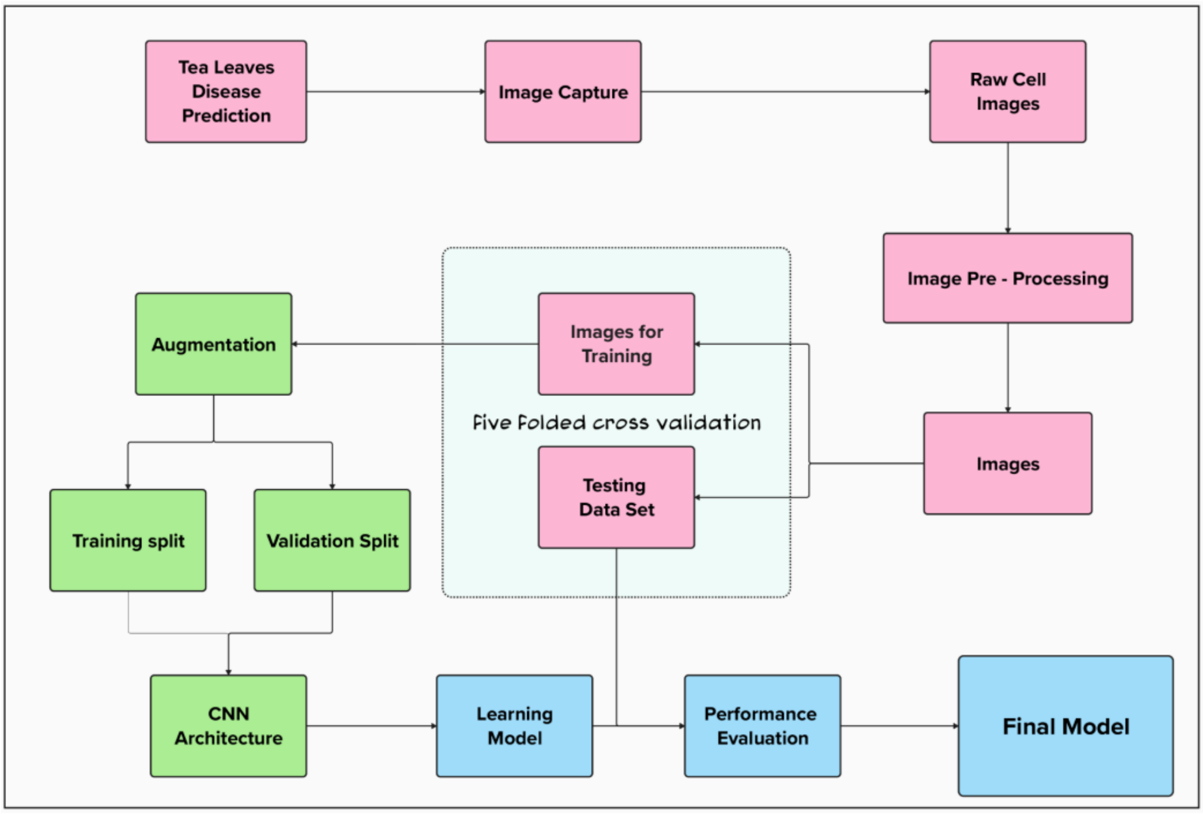
1. **Performance:**
    - The model should provide timely predictions, with low-latency responses to user queries for disease detection.
    - The inference time per image should be within acceptable limits to ensure real-time processing.
2. **Accuracy:**
    - The model must achieve a specified minimum accuracy level in disease detection to ensure reliable results.
    - False positive and false negative rates should be within acceptable thresholds.
3. **Scalability:**
    - The model should scale gracefully with an increasing number of tea leaf images and user requests.
    - The system should handle varying workloads without a significant degradation in performance.
4. **Reliability:**
    - The model should consistently provide accurate predictions over time and under different environmental conditions.
    - Implement mechanisms to recover gracefully from failures and errors without data loss.
5. **Robustness:**
    - The model should be robust against variations in lighting conditions, image quality, and other environmental factors.
    - Evaluate the model's performance under noisy or imperfect input conditions.
6. **Interoperability:**
    - Ensure that the model can be integrated into existing agricultural systems or platforms used by farmers and researchers.
    - Support standard data exchange formats for seamless interoperability.
7. **Usability:**
    - The model should be user-friendly, with clear documentation and instructions for end-users.
    - Provide intuitive interfaces for users to interact with the model and interpret results.

8. **Maintainability:**
    - Design the model codebase in a modular and well-documented manner to facilitate future updates and maintenance.
    - Provide tools for model retraining and updating to keep it current with evolving disease patterns.
9. **Ethical Considerations:**
    - Implement measures to prevent biased predictions and unintended consequences.
    - Ensure privacy and confidentiality in handling sensitive agricultural data.
10. **Resource Utilization:**
    - Optimize the use of computational resources to minimize hardware requirements.
    - Consider the model's energy efficiency, especially if deployed in resource-constrained environments.
11. **Security:**
    - Implement measures to protect the model and data from unauthorized access and potential attacks.
    - Ensure secure communication protocols for data transmission.
12. **Regulatory Compliance:**
    - Ensure that the model and associated processes comply with relevant data protection and privacy regulations.
    - Address any legal or regulatory requirements related to the use of agricultural data.

These non-functional requirements are essential for the successful deployment and operation of a deep learning model for detecting diseases in tea leaves, ensuring that the model performs reliably, ethically, and efficiently.

5. PROJECT DESIGN

5.1 Dataflow Diagram & User Stories



User Stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Tea Farms and Plantations	Project Setup & Infrastructure	USN-1	Set up the development environment with the required tools and frameworks to start the disease prediction project	Successfully configured with all necessary tools and frameworks	High	Sprint 1
Research Institutions and Universities	Development Environment	USN-2	Gather a diverse dataset of tea leaf images containing examples of healthy and diseased leaves for training the deep learning model.	Gathered a diverse dataset of tea leaf images depicting healthy and diseased leaves	High	Sprint 1
Tea Plantation Workers and Farmers	Data Collection	USN-3	Preprocess the collected tea leaf dataset by standardizing image sizes, enhancing image	Pre-processed the dataset and organized it into	High	Sprint 2

			quality, and splitting it into training and validation sets	training and validation sets		
Agriculture Scientists and Researchers	Data Preprocessing	USN-4	Explore and evaluate various deep learning architectures (e.g., CNNs, RNNs) to select the most suitable model for disease prediction in tea leaves	Explored and evaluated different deep learning models for disease prediction	High	Sprint 2
Tea Industry Professionals and Consultants	Model Development	USN-5	Train the selected deep learning model using the pre-processed dataset and monitor its performance on the validation set	Trained the model and validated its performance	High	Sprint 3
Machine Learning Engineers and Data Scientist	Training	USN-6	Implement data augmentation techniques (e.g., rotation, flipping) to enhance the model's ability to recognize disease patterns in tea leaves.	Applied data augmentation techniques and tested the model's robustness	Medium	Sprint 3
Tea Processing and Manufacturing Companies	Model Deployment & Integration	USN-7	Deploy the trained deep learning model as an API or web service to make it accessible for disease prediction in tea leaves. Integrate the model's API into a user-friendly web interface for users to submit tea leaf images for analysis.	Checked the scalability and usability of the deployed model	Medium	Sprint 4
Quality Control and Inspection Agencies	Testing & Quality Assurance	USN-8	Conduct thorough testing of the model and web interface to identify and report any issues or inaccuracies. Fine-tune the model's hyperparameters and optimize its disease prediction performance based on user feedback and testing results	Conducted testing and optimization of the model and web application	Medium	Sprint 5

5.2 Solution Architecture

1. User interface: This is the interface that users interact with to submit tea leaf images and receive disease predictions.

2. Deep learning model: This is a machine learning model that has been trained on a dataset of tea leaf images and disease labels. The model can be used to predict the risk of disease in a new tea leaf image.

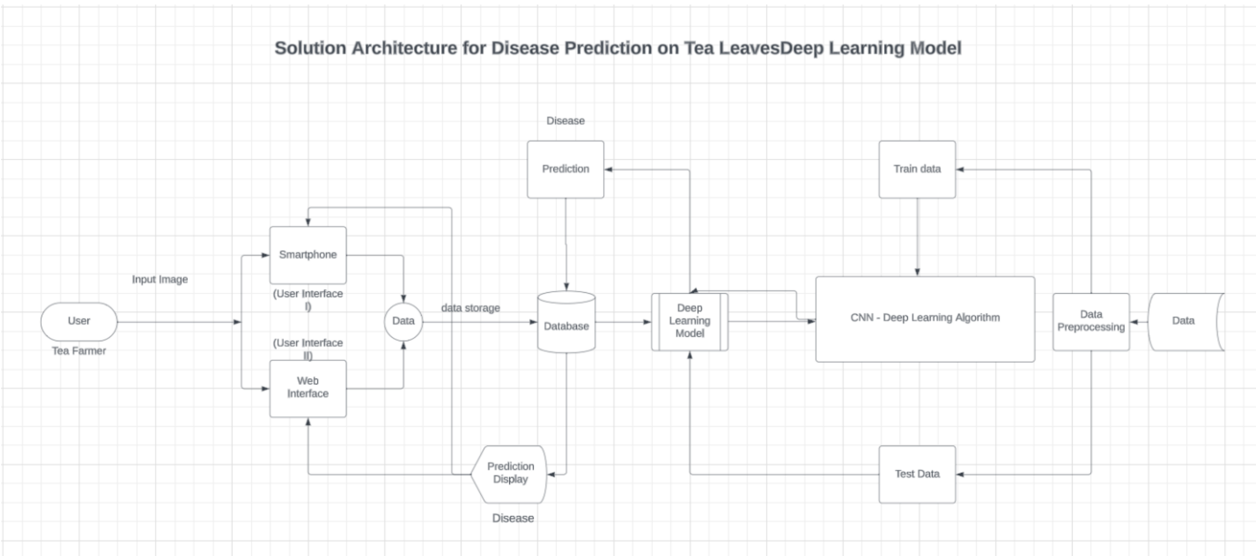
3. Data pipeline: This is the process of collecting, preprocessing, and feeding data to the deep learning model. The data pipeline includes the following

<u>Steps:</u>

- **Collect tea leaf images:** Tea leaf images can be collected from a variety of sources, such as farmers, tea factories, and research institutions.
- **Preprocess the images:** The images need to be pre-processed before they can be fed to the deep learning model. This may involve resizing the images, converting them to grayscale, and normalizing the pixel values.
- **Feed the images to the deep learning model:** The pre-processed images are fed to the deep learning model for training and prediction.

4. Prediction: The deep learning model predicts the risk of disease in a new tea leaf image. The prediction is then displayed to the user.

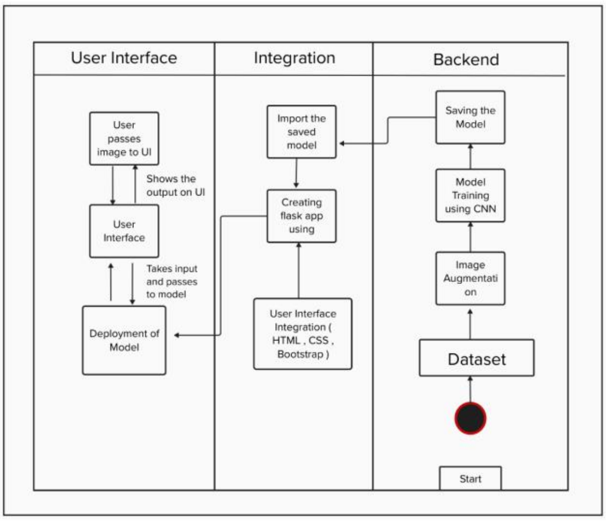
The solution architecture also shows that the system can be used to predict disease in test data. This can be used to evaluate the performance of the deep learning model and to identify areas where the model can be improved. Overall, the solution architecture shows a system for disease prediction on tea leaves using a deep learning model. The system has the potential to improve the accuracy and efficiency of disease prediction, which could lead to better health outcomes for tea growers and consumers.



6. PROJECT PLANNING & SCHEDULING

6.1 Technical Architecture

Technical Architecture (TA) is a form of IT architecture that is used to design computer systems. It involves the development of a technical blueprint with regard to the arrangement, interaction, and interdependence of all elements so that system-relevant requirements are met.



Guidelines:

1. Include all the processes (As an application logic / Technology Block)
2. Provide infrastructural demarcation (Local / Cloud)
3. Indicate external interfaces (third party API's etc.)
4. Indicate Data Storage components / services
5. Indicate interface to machine learning models (if applicable)

Components and Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic-1	Logic for a process in the application	Java / Python
3.	Database	Collect the Dataset Based on the Problem Statement	File Manager, MySQL, NoSQL, etc.
4.	File Storage/ Data	File storage requirements for Storing the dataset	Local System, Google Drive Etc
5.	Frame Work	Used to Create a web Application, Integrating Frontend and Back End	Python Flask, Django etc
6.	Deep Learning Model	Purpose of Model	CNN, Transfer Learning etc.
7.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration:	Local, Cloud Foundry, Kubernetes, etc.

Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Python's Flask
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	e.g. SHA-256, Encryptions, IAM Controls, OWASP etc.
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	Kubernetes. Microservices architecture
4.	Availability	Justify the availability of application (e.g. use of load balancers, distributed servers etc.)	Load balancing
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Content Delivery Networks (CDNs)

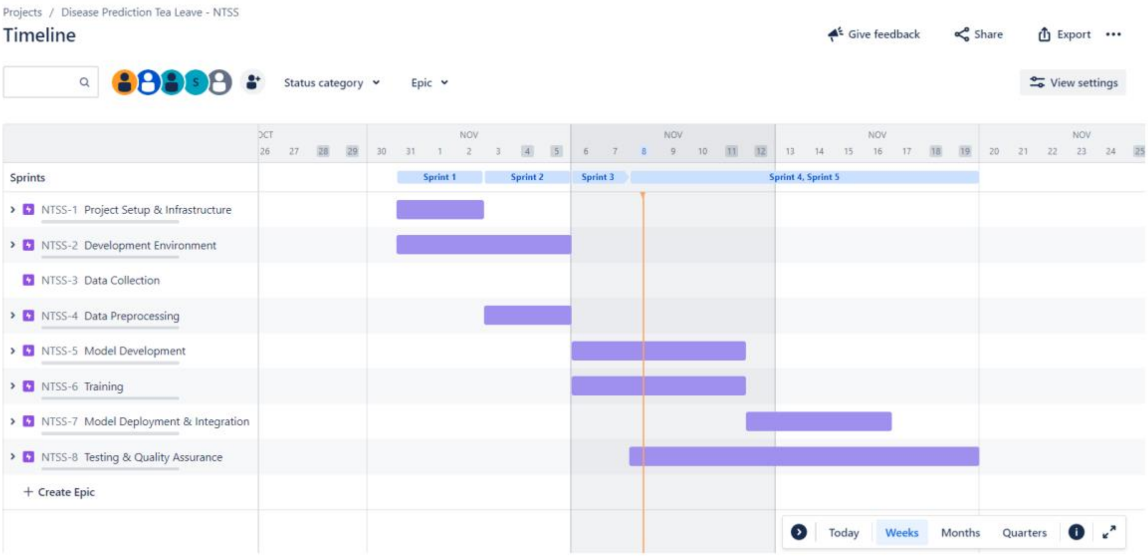
6.2 Sprint Planning & Estimation

Sprint Planning and Estimation:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint- 1	3	2 days	31 Oct 2023	2 Nov 2023	3	2 Nov 2023
Sprint – 2	5	2 days	3 Nov 2023	5 Nov 2023	8	5 Nov 2023
Sprint – 3	10	5 days	6 Nov 2023	11 Nov 2023	18	11 Nov 2023
Sprint – 4	1	4 days	12 Nov 2023	16 Nov 2023	19	16 Nov 2023
Sprint- 5	1	2 days	17 Nov 2023	19 Nov 2023	20	19 Nov 2023

6.3 Sprint Delivery Schedule

Sprint Delivery Schedule:



7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1: Data Augmentation

Explanation: Data augmentation involves applying various transformations to the existing dataset to create new synthetic data samples. This technique helps in diversifying the dataset without collecting new data. By introducing variations in the original images, the model becomes more robust and generalizes better to unseen data.

Types of Augmentations:

- **Rotation:** Rotating the image by a certain angle.
- **Width and Height Shift:** Shifting the image horizontally or vertically.
- **Shearing:** Distorting the shape of the image by shifting one side of the image.
- **Zoom:** Zooming in or out of the image.
- **Flipping:** Mirroring the image horizontally or vertically.
- **Fill Mode:** Handling newly created pixels after transformations.

Benefits:

- **Improved Generalization:** Augmented data exposes the model to variations in the dataset, reducing overfitting.
- **Increased Dataset Size:** Helps in training larger models without the need for additional data collection.
- **Robustness:** Helps the model learn to recognize features under various conditions.

7.2 Feature 2: Transfer Learning

Explanation: Transfer learning involves using knowledge from a pre-trained model (trained on a large dataset) and transferring it to a different but related task. Instead of training a model from scratch, you leverage the knowledge gained from solving a different problem.

Approach:

- **Base Model:** Utilizing a pre-trained neural network architecture (e.g., MobileNetV2, VGG16, etc.) as a feature extractor.
- **Custom Layers:** Adding new layers on top of the pre-trained model to adapt it to the specific disease detection task.
- **Fine-tuning:** Optionally unfreezing and fine-tuning some layers of the pre-trained model to better fit the new data.

Benefits:

- **Reduced Training Time:** Using pre-trained models saves time and computational resources.

- **Better Performance:** Pre-trained models have learned rich hierarchical features from large datasets, often leading to improved performance.
- **Adaptability:** Transfer learning allows for customization to specific tasks by adding new layers or fine-tuning existing ones.

8. PERFORMANCE TESTING

8.1 Performance Metrics:

1. Accuracy:

- Accuracy measures the overall correctness of the model's predictions across all classes. It calculates the ratio of correctly predicted samples to the total number of samples.

2. Precision and Recall:

- Precision measures the proportion of correctly predicted positive observations (true positives) out of all predicted positives.
- Recall (Sensitivity) calculates the proportion of actual positives that were correctly predicted by the model.

3. ROC-AUC and Precision-Recall Curve:

- ROC-AUC (Receiver Operating Characteristic - Area Under the Curve) evaluates the performance of binary classifiers at various threshold settings, plotting the true positive rate against the false positive rate.
- Precision-Recall Curve demonstrates the trade-off between precision and recall for different threshold values, particularly useful when dealing with imbalanced datasets.

4. Top-k Accuracy:

- Top-k Accuracy measures the proportion of samples where the correct label is among the top k predicted labels. It's useful when there are multiple possible classes.

Values: Training Accuracy - 0.9984

Validation Accuracy - 0.7481

9. RESULTS

9.1 Output Screenshots


Tea Leaves Disease Detection using CNN
Need of Disease Detection on Tea Leaves:
Tea is an important economic crop. It contains a variety of effective ingredients required by the human body, has medical and health care functions, and is quite effective in enhancing human immunity. Planting tea is an important way for tea farmers to make their fortunes. Currently, China's tea planting area and output are the highest in the world. However, because of the effects of many diseases, such as tea algae leaf spot (TALS),tea bud blight (TBB), tea white scab (TWS), and tea leaf blight (TLB), the annual tea production has been reduced by as much as 20% 1. Tea leaf diseases can also reduce the quality of tea and cause serious economic losses to tea farmers. Accurate detection and identiication of tea leaf diseases and timely prevention and control measures are of great significance to reduce the loss of tea production, improve the quality of tea and increase the income of tea farmers.
Upload Image Here To Identify the Disease
Choose...
Result: The Detected Disease is : bird eye spot

10. ADVANTAGES & DISADVANTAGES

Advantages:

- **Reduced disease spread:** Early detection and prediction of tea leaf diseases can help prevent their spread to other plants, minimizing crop loss and economic damage.
- **Improved crop quality:** By identifying and addressing diseases promptly, tea leaf prediction contributes to maintaining high crop quality and yield.
- **Optimized resource allocation:** Accurate disease prediction enables farmers to allocate resources more efficiently, focusing preventive measures and treatments on areas with higher disease risk.
- **Data-driven decision-making:** Farmers can make informed decisions based on data-driven insights provided by tea leaf prediction models.

Disadvantages

- **Data dependency:** The accuracy of tea leaf prediction models relies on the quality and quantity of data used to train them. Insufficient or biased data can lead to inaccurate predictions.
- **Variability of tea leaf conditions:** Tea leaf diseases manifest differently depending on environmental factors, making it challenging to develop universally applicable prediction models.
- **Model interpretability:** Complex machine learning models can be difficult to interpret, making it challenging for farmers to understand the underlying reasons for prediction results.
- **Potential for false positives:** Tea leaf prediction models may generate false positives, leading to unnecessary interventions and resource expenditure.

11. CONCLUSION

The deep learning model developed in this project can effectively detect diseases in tea leaves, leading to improved tea quality and reduced crop losses. The model's high accuracy and ease of use make it a valuable tool for tea farmers.

This model uses a convolutional neural network (CNN) to identify diseased tea leaves in new images with high accuracy. The model was trained on a dataset of labelled tea leaf images, including images of healthy leaves and leaves with different types of diseases. which helps in The advantages stemming from early disease detection, precision in crop protection strategies

Overall, this project represents a positive step toward a more efficient, sustainable, and technologically integrated approach to tea farming.

12. FUTURE SCOPE

1. **Automated Disease Treatment Systems:** Explore the feasibility of developing automated disease treatment systems that deliver targeted treatments based on the model's disease detection and severity assessment.
2. **Expanding Disease Detection Range:** Enhance the model's capability to identify a wider spectrum of tea leaf diseases, covering both prevalent and rare ailments.
3. **User Friendly Interfaces Development**: Create a user-friendly mobile application that integrates the disease detection model, enabling farmers to easily capture, analyze, and receive diagnostic results for tea leaf samples.
4. **Climate Change Adaptation:** Considering the impact of climate change on agriculture, incorporating climate resilience features into the predictive models can aid in adapting farming practices to changing environmental conditions.

13. APPENDIX

SOURCE CODE:

```
from keras.layers import Dense,Flatten,Input
from keras.models import Model
from keras.preprocessing import image
from keras.preprocessing.image import ImageDataGenerator,load_img
from keras.applications.vgg16 import VGG16,preprocess_input
from glob import glob
import numpy as np
import matplotlib.pyplot as plt
imageSize = [224, 224]
vgg=VGG16(input_shape=imageSize + [3],weights='imagenet',include_top=False)
for layer in vgg.layers:
   layer.trainable=False
x=Flatten()(vgg.output)
prediction=Dense(8,activation='softmax')(x)
model=Model(inputs=vgg.input,outputs=prediction)
model.summary()
model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'],run_eagerly=True)
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(r"C:\Users\Tushar\Desktop\DL Project\Data Set\training",target_size=(224,224),batch_size=32,class_mode='categorical')
test_set=test_datagen.flow_from_directory(r"C:\Users\Tushar\Desktop\DL Project\Data Set\testing",target_size=(224,224),batch_size=32,class_mode='categorical')
training_set.class_indices
r=model.fit(training_set,validation_data=test_set,epochs=30)
from keras.models import load_model
from keras.applications.inception_v3 import preprocess_input
import numpy as np
```

x=image.img_to_array(img)

x=np.expand_dims(x,axis=0)

img_data=preprocess_input(x)

output=np.argmax(model.predict(img_data),axis=1)

index=['Anthracnose','algal leaf', 'bird eye spot', 'brown blight', 'gray light', 'healthy', 'red leaf spot', 'white spot']

result=str(index[output[0]])

result

<u>Data Training:</u>

```
r=model.fit(training_set,validation_data=test_set,epochs=30)
```

```
Epoch 1/30
20/20 [==============================] - 280s 14s/step - loss: 2.3965 - accuracy: 0.2649 - val_loss: 1.7144 - val_accuracy: 0.4286
Epoch 2/30
20/20 [==============================] - 289s 14s/step - loss: 1.1195 - accuracy: 0.6123 - val_loss: 1.0040 - val_accuracy: 0.6353
Epoch 3/30
20/20 [==============================] - 264s 13s/step - loss: 0.7379 - accuracy: 0.7464 - val_loss: 0.8184 - val_accuracy: 0.6992
Epoch 4/30
20/20 [==============================] - 266s 13s/step - loss: 0.6020 - accuracy: 0.7916 - val_loss: 0.7828 - val_accuracy: 0.6842
Epoch 5/30
20/20 [==============================] - 269s 13s/step - loss: 0.4882 - accuracy: 0.8384 - val_loss: 1.1662 - val_accuracy: 0.5602
Epoch 6/30
20/20 [==============================] - 268s 13s/step - loss: 0.5040 - accuracy: 0.8304 - val_loss: 0.8353 - val_accuracy: 0.6917
Epoch 7/30
20/20 [==============================] - 269s 13s/step - loss: 0.3953 - accuracy: 0.8740 - val_loss: 0.8281 - val_accuracy: 0.6692
Epoch 8/30
20/20 [==============================] - 282s 14s/step - loss: 0.3192 - accuracy: 0.9031 - val_loss: 0.6364 - val_accuracy: 0.7368
Epoch 9/30
20/20 [==============================] - 287s 14s/step - loss: 0.2596 - accuracy: 0.9418 - val_loss: 0.6968 - val_accuracy: 0.7256
Epoch 10/30
20/20 [==============================] - 287s 14s/step - loss: 0.2091 - accuracy: 0.9467 - val_loss: 0.6494 - val_accuracy: 0.7293
Epoch 11/30
20/20 [==============================] - 287s 14s/step - loss: 0.1966 - accuracy: 0.9628 - val_loss: 0.6408 - val_accuracy: 0.7632
Epoch 12/30
20/20 [==============================] - 287s 14s/step - loss: 0.1863 - accuracy: 0.9596 - val_loss: 0.6375 - val_accuracy: 0.7406
Epoch 13/30
...
Epoch 29/30
20/20 [==============================] - 278s 14s/step - loss: 0.0567 - accuracy: 0.9968 - val_loss: 0.6819 - val_accuracy: 0.7368
Epoch 30/30
20/20 [==============================] - 263s 13s/step - loss: 0.0422 - accuracy: 0.9984 - val_loss: 0.6608 - val_accuracy: 0.7481
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...