

1. INTRODUCTION

1.1 Project Overview

1.2 Purpose

2. IDEATION & PROPOSED SOLUTION

2.1 Problem Statement Definition

2.2 Empathy Map Canvas

2.3 Ideation & Brainstorming

2.4 Proposed Solution

3. REQUIREMENT ANALYSIS

3.1 Functional requirement

3.2 Non-Functional requirements

4. PROJECT DESIGN

4.1 Data Flow Diagrams

4.2 Solution & Technical Architecture

4.3 User Stories

5. CODING & SOLUTIONING

5.1 Feature 1

5.2 Feature 2

5.3 Database Schema (if Applicable)

6. RESULTS

6.1 Performance Metrics

7. ADVANTAGES & DISADVANTAGES

8. CONCLUSION

9. FUTURE SCOPE

10. APPENDIX

Source Code

GitHub & Project Video Demo Link

1. INTRODUCTION

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 identification 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.

Tea leaf diseases can be identified by observing the leaves condition like color and spots on the leaves. Strange spots & colors on the leaves may be an indication of disease. Experts and farmers can identify the type of disease by observing the leaves manually.

At present, the diagnosis of tea leaf diseases relies on the manual method. Most tea trees grow in rugged mountainous areas. Thus, it is time-consuming and costly for experts to go to the tea garden for diagnosis. However, results are largely subjective when farmers rely on their own experience to distinguish the types of tea diseases. The user interacts with the UI (User Interface) to choose the image. The chosen image analyzed by the model which is integrated with flask application.

The following software's, concepts, and packages are required Anaconda Navigator is a free and open-source distribution of the Python and R programming languages for data science and machine learning related applications. It can be installed on Windows, Linux, and macOS. Conda is an open-source, cross-platform, package management

system. Anaconda comes with so very nice tools like Jupyter Lab, Jupyter Notebook, QtConsole, Spyder, Glueviz, Orange, Rstudio, Visual Studio Code. For this project, we will be using Jupyter notebook and Spyder.

Nowadays, artificial intelligence techniques like machine learning and various deep learning algorithms have been mainly used in a wide variety of agricultural operations. Among these techniques, deep learning achieves better developments in the agricultural field of research because of its automatic feature extraction capability and also consists of several deep learning algorithm.

1.1 PROJECT OVERVIEW

Tea leaf diseases can be identified by observing the leaves condition like color and spots on the leaves. Strange spots & colors on the leaves may be an indication of disease. Experts and farmers can identify the type of disease by observing the leaves manually.

At present, the diagnosis of tea leaf diseases relies on the manual method. Most tea trees grow in rugged mountainous areas. Thus, it is time-consuming and costly for experts to go to the tea garden for diagnosis. However, results are largely subjective when farmers rely on their own experience to distinguish the types of tea diseases.

To overcome the above problem we are building a model which is used for the prevention and early detection of tea leaves disease. Basically tea leaves disease diagnosis depends on the different characteristics like color, spots, texture etc. Here the person can capture the images of the tea leaves and then the image will be sent to the trained model. The model analyzes the image and detects whether the tea leaves are having any disease or not and its type.

1.2 PROJECT PURPOSE

The main purpose of the project is to develop a website capable of detecting diseases in tea leaves accurately and efficiently. And to achieve the following:

1.Enhance disease identification accuracy: Improve the accuracy of disease detection algorithms by leveraging machine learning techniques and incorporating user feedback for model refinement.

2.Expand disease coverage: Extend the range of diseases that the website can identify and classify in tea leaves, covering both common and rare ailments.

3.User-friendly interface: Create an intuitive and user-friendly interface that allows users to easily upload tea leaf images, view disease predictions, and access additional information and recommendations.

4.Disease prevention and management: Assist tea farmers in early detection and effective management of diseases in tea plants, thereby improving crop yield and quality.

5.Knowledge dissemination: Educate tea farmers and enthusiasts about tea leaf diseases, fostering a better understanding of these ailments and promoting best practices for disease management in the tea industry.

6.Accessibility and reach: Make disease detection technology easily accessible to tea farmers, regardless of their geographical location or level of technical expertise, by providing a user-friendly web-based platform.

2. IDEATION AND PROPOSED SOLUTION

Ideation refers to the process of generating, developing, and exploring new ideas. It involves brainstorming and coming up with creative solutions, concepts, or strategies to address a particular problem or challenge.

In this ideation phase , the following things are included:

- Problem statement definition
- Empathy map
- Ideation and Brainstorming

2.1 PROBLEM STATEMENT DEFINITION

Tea farmers and producers face significant challenges in detecting and managing diseases in tea leaves. Currently, there is a lack of accessible and accurate tools to identify and classify diseases, resulting in reduced crop yield, compromised tea quality, and economic losses. Existing methods for disease detection are often time-consuming, subjective, and require specialized expertise, making them impractical for widespread adoption.

The problem primarily affects tea farmers and producers who encounter difficulties in detecting diseases in tea leaves, leading to negative impacts on their crop yield and financial sustainability. There is a scarcity of user-friendly and reliable tools specifically designed for tea leaf disease detection. The existing methods are limited in their accessibility, accuracy, and efficiency, hindering effective disease management. Without timely and accurate disease detection, tea farmers and producers struggle to implement appropriate preventive measures and targeted treatments. This results in decreased crop yield and compromised tea quality, affecting their profitability and market competitiveness.

Current disease detection methods rely heavily on manual observation and subjective judgment, making the process time-consuming and prone to errors. Tea farmers often lack

the necessary expertise and training required to accurately identify and classify diseases.

| I am | I'm trying to | But | Because | Which makes me feel |
|---|---|--|---|--|
| A tea farmer who is struggling to identify diseases in your tea leaves. You are determined to find a solution that will help you detect these diseases early and save your crops. | Create a deep learning model that can accurately detect diseases in tea leaves. You're researching different data sources and algorithms to find the best approach. | I have some challenges along the way. One challenge is finding high-quality data to train your model. Another challenge is choosing the right algorithm that will be effective for detecting diseases in tea leaves. | Important to find a solution because detecting diseases early is crucial for the health of your tea crops and the success of your business. You also want to help other tea farmers who may be facing similar challenges. | This challenge makes you feel determined to find a solution and motivated to learn more about deep learning. It also makes you feel a sense of responsibility to your tea crops and your fellow tea farmers. |

mirr

Figure 2.1 Problem statement definition

Based on this problem statement, the proposed tea leaf disease detection project aims to develop a solution that overcomes the limitations of existing methods, providing tea farmers and producers with an accessible, accurate, and efficient tool to detect diseases in tea leaves.

2.2 EMPATHY MAP CANVAS

An empathy map is a tool used to gain a deeper understanding of a specific target audience or user group. It helps in developing empathy by exploring and documenting the users' thoughts, feelings, behaviors, and needs. By creating an empathy map, you can gain insights into the users' perspectives, motivations, and pain points, which can inform the design and development of products or solutions tailored to their needs.

The traditional empathy map consists of four quadrants, each focusing on different aspects of the user experience:

1. Says: This quadrant captures the explicit statements or quotes from the user. It includes what the user says, their goals, aspirations, and any relevant quotes or phrases that express their thoughts and desires.

2. Thinks: In this quadrant, you outline the internal thoughts and assumptions of the user. It involves exploring what the user is thinking, their beliefs, fears, expectations, and any underlying thoughts that drive their behavior.

3. Feels: This quadrant delves into the emotional state of the user. It explores the user's emotions, desires, frustrations, and any other relevant feelings or sensations they may experience throughout their journey.

4. Does: Here, you focus on the observable actions and behaviors of the user. This includes their physical actions, interactions, habits, and behaviors in relation to the problem or context you are addressing.

By creating an empathy map, you can synthesize user research, interviews, or observations into a visual representation that helps the project team gain a deeper understanding of the target audience. This understanding can inform decision-making, guide product design, and enable the creation of user-centric solutions.



Empathy map

Use this framework to develop a deep, shared understanding and empathy for other people. An empathy map helps describe the aspects of a user's experience, needs, and pain points, to quickly understand your users' experience and mindset.

Share template feedback



Build empathy

The information you add here should be representative of the observations and research you've done about your users.

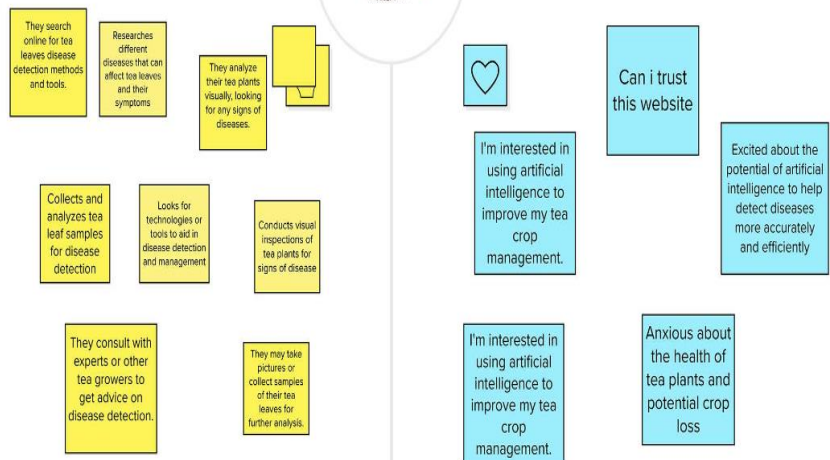
Says

What have we heard them say?
What can we imagine them saying?



Thinks

What are their wants, needs, hopes, and dreams? What other thoughts might influence their behavior?



Does

What behavior have we observed?
What can we imagine them doing?

Feels

What are their fears, frustrations, and anxieties? What other feelings might influence their behavior?

2.3. IDEATION AND BRAINSTORMING

Ideation refers to the process of generating, developing, and exploring new ideas. It involves brainstorming and coming up with creative solutions, concepts, or strategies to address a particular problem or challenge.

Brainstorming is a group problem-solving method that involves the spontaneous contribution of creative ideas and solutions. This technique requires intensive, freewheeling discussion in which every member of the group is encouraged to think aloud and suggest as many ideas as possible based on their diverse knowledge.



Brainstorm & idea prioritization

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

🕒 10 minutes to prepare

🕒 1 hour to collaborate

👤 2-8 people recommended

💬 [Share template feedback](#)



Before you collaborate

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

🕒 10 minutes

A

Team gathering

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

B

Set the goal

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

C

Learn how to use the facilitation tools

Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#) →

1

Define your problem statement

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

 5 minutes

PROBLEM

At present, the diagnosis of tea leaf diseases relies on the manual method. Most tea trees grow in rugged mountainous areas. Thus, it is time-consuming and costly for experts to go to the tea garden for diagnosis. However, results are largely subjective when farmers rely on their own experience to distinguish the types of tea diseases.



Key rules of Brainstorming

To run an smooth and productive session



Stay in topic.



Encourage wild ideas.



Defer judgment.



Listen to others.



Go for volume.



If possible, be visual.

2

Brainstorm

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

🕒 10 minutes

TIP

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

Person 1

Research and gather data on different types of diseases that commonly affect tea plants. Explore existing methods for detecting diseases in tea leaves and identify their limitations

Identify different data sources that can be used to train a deep learning model for detecting diseases in tea leaves. Determine the most effective deep learning algorithm to use for this particular application

Person 2

Consider the hardware and software requirements for building a deep learning model for detecting diseases in tea leaves. Explore existing methods for detecting diseases in tea leaves and identify their limitations

Develop a user-friendly interface for the deep learning model that can be used by tea planters to diagnose diseases in their plants. Identify potential partnerships with tea plantations or agricultural organizations for data collection and model testing

Person 3

Consider the ethical implications of using a deep learning model for diagnosing diseases in tea leaves and how to address potential concerns. Evaluate the impact that a deep learning model for detecting diseases in tea leaves could have on the tea industry and the environment.

1. Define the problem: The problem is how to create a deep learning model that can accurately detect diseases in tea leaves.
2. Set a time limit: Decide how much time you want to spend on the brainstorming session. It could be anywhere from 10 minutes to an hour.

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 can break it up into smaller sub-groups.

🕒 20 minutes

TIP

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

Research and gather data on different types of diseases that commonly affect tea plants. Explore existing methods for detecting diseases in tea leaves and identify their limitations

Consider the hardware and software requirements for building a deep learning model for detecting diseases in tea leaves. Explore existing methods for detecting diseases in tea leaves and identify their limitations

Identify different data sources that can be used to train a deep learning model for detecting diseases in tea leaves. Determine the most effective deep learning algorithm to use for this particular application

1. Define the problem: The problem is how to create a deep learning model that can accurately detect diseases in tea leaves.
2. Set a time limit: Decide how much time you want to spend on the brainstorming session. It could be anywhere from 10 minutes to an hour.

Consider the ethical implications of using a deep learning model for diagnosing diseases in tea leaves and how to address potential concerns. Evaluate the impact that a deep learning model for detecting diseases in tea leaves could have on the tea industry and the environment.

Develop a user-friendly interface for the deep learning model that can be used by tea planters to diagnose diseases in their plants. Identify potential partnerships with tea plantations or agricultural organizations for data collection and model testing

4

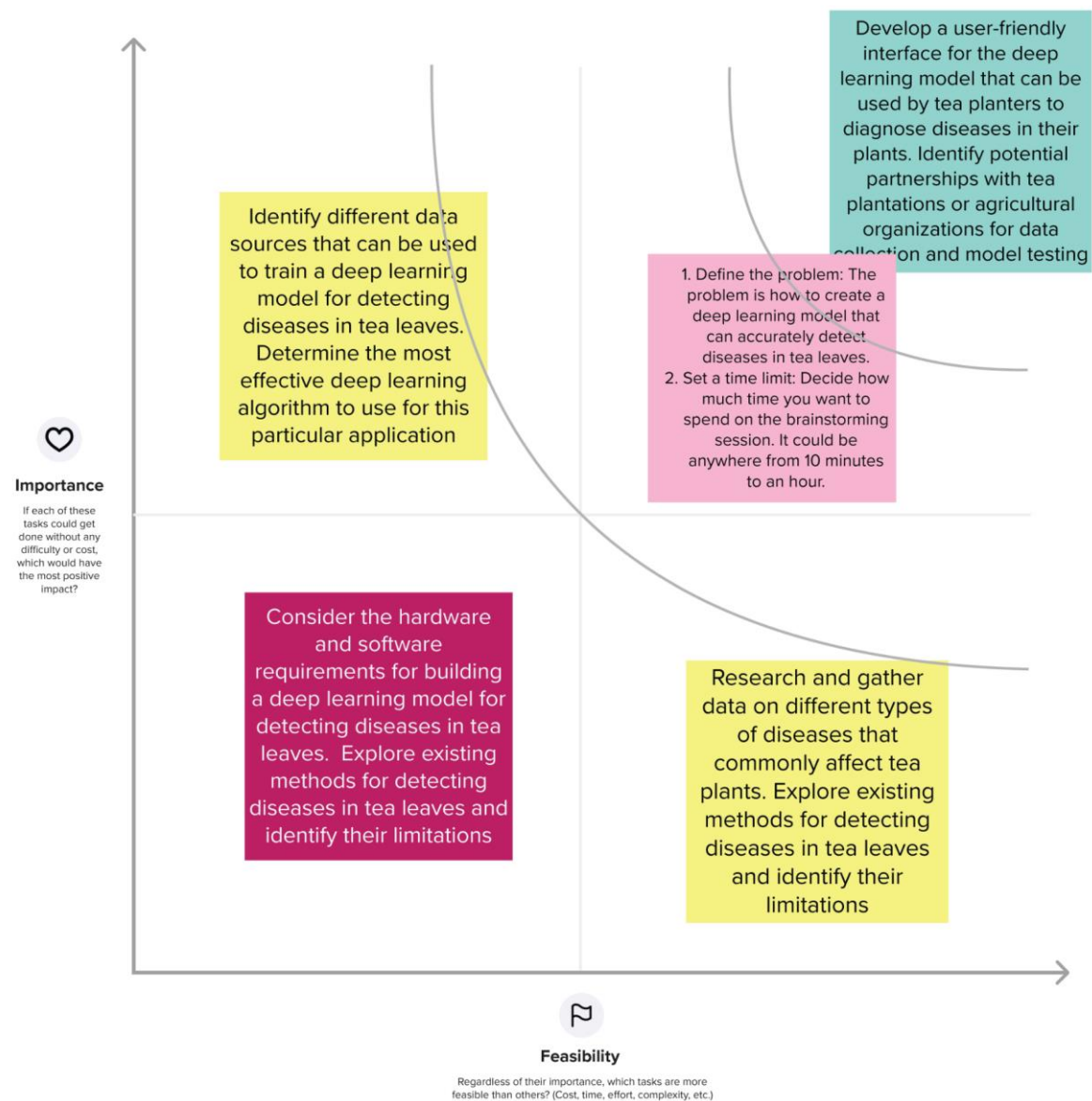
Prioritize

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

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H** key on the keyboard.



2.4. PROPOSED SOLUTION

The proposed solution is to develop a comprehensive tea leaf disease detection website that utilizes deep learning model to accurately identify and classify diseases in tea leaves. The website will serve as a valuable tool for tea farmers and researchers, providing timely and accurate information for disease prevention and management.

1. Image Acquisition and Pre-processing :

- Enable tea farmers to capture tea leaf images using smartphones or digital cameras.
- Implement preprocessing techniques to optimize image quality, remove noise, and enhance the input for the deep learning model.

2. Deep Learning Model Development:

- Utilize deep learning techniques, such as convolutional neural networks (CNNs), to develop a powerful and accurate disease detection model.
- Train the deep learning model using a diverse dataset of tea leaf images, covering various disease types and severities.
- Employ transfer learning to leverage pre-trained models, such as VGGNet or ResNet, to enhance the performance and efficiency of the disease detection.

3. User-Friendly Interface:

- Design an intuitive and user-friendly website interface that allows tea farmers to easily interact with the application.
- Create an image upload feature that enables users to submit tea leaf images for disease detection using the deep learning model.

4. Real-Time Disease Detection:

- Integrate the trained deep learning model into the backend of the website.

- Develop algorithms and infrastructure to preprocess user-uploaded images, pass them through the deep learning model, and generate real-time disease detection results.

5. Disease Classification and Recommendations:

- Classify the detected diseases into relevant categories using the deep learning model's predictions.

- Provide detailed information about each identified disease, including symptoms, causes, and recommended management practices.

- Offer tailored recommendations and suggested treatments based on the detected diseases.

By focusing on utilizing a deep learning model, your tea leaf disease detection website will leverage the power of artificial intelligence to accurately identify and classify diseases in tea leaves. This solution will provide tea farmers with a valuable tool for timely disease detection, enabling them to take proactive measures and ensure the health and productivity of their tea crops.

3. REQUIREMENT ANALYSIS

Requirement analysis is a systematic process used in software engineering and other fields to identify, clarify, and define the needs and expectations of stakeholders for a particular system, product, or project. It involves gathering and understanding the requirements, constraints, and objectives of the stakeholders to ensure that the final solution meets their needs effectively. The goal of requirement analysis is to establish a clear and unambiguous understanding of what the system should accomplish, how it should behave, and what features and functions it should have. This process helps in defining the scope of the project, identifying potential risks and challenges, and providing a foundation for the subsequent phases of development. By conducting a thorough requirement analysis, software development teams can minimize misunderstandings, avoid costly rework, and increase the likelihood of delivering a system that satisfies the stakeholders' needs and expectations.

Requirement analysis includes analysis of both functional and non functional requirements of the system. Functional requirements define what a system, product, or software application should do or the specific functions it should perform to fulfill its intended purpose. These requirements describe the desired behavior, features, and capabilities of the system. Non-functional requirements are the specifications that describe the characteristics, constraints, and qualities of a system or software application, rather than its specific functionality. They define how the system should perform, behave, and interact with its environment, rather than what tasks it should accomplish. Non-functional requirements are important for evaluating the overall performance, usability, security, maintainability, and other aspects of a system.

The functional and non-functional requirements of our system is listed below

3.1. FUNCTIONAL REQUIREMENTS

Functional requirements define what a system, product, or software application should do or the specific functions it should perform to fulfill its intended purpose. These requirements describe the desired behavior, features, and capabilities of the system. They outline the functionality that the system must possess to meet the needs of its users and stakeholders. Functional requirements provide a clear and detailed description of what the system should do, guiding the development process and serving as a basis for testing and verification. They help ensure that the final product meets the expectations and needs of its users.

| FR No. | Functional Requirement (Epic) | Sub Requirement (Story / Sub-Task) |
|--------|-------------------------------|--|
| FR-1 | Home Page | In this Web Page the main home page of the website is displayed with some basic design with 3 buttons on it, which redirects us to the next 3 pages respectively. |
| FR-2 | About Page | In this page, the about of the project is described in a effective way with buttons with link to other pages including home page. |
| FR-3 | Tea home Page | In this page, tha general information about the tea diseases and about something about the precautions and treatments will be displayed. |
| FR-4 | Disease prediction page | Uploading the images of tea leaves could be performed in this page. Once the report is uploaded prediction is performed on backend and output is shown on the same page as the name of the disease and the first aid treatment to be done. |

3.2. NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements are the specifications that describe the characteristics, constraints, and qualities of a system or software application, rather than its specific functionality. They define how the system should perform, behave, and interact with its environment, rather than what tasks it should accomplish. Non-functional requirements are important for evaluating the overall performance, usability, security, maintainability, and other aspects of a system. Non-functional requirements are essential for ensuring that a system meets the desired quality attributes and aligns with the stakeholders' expectations. They guide the design, implementation, and evaluation of the system from a broader perspective beyond its core functionality.

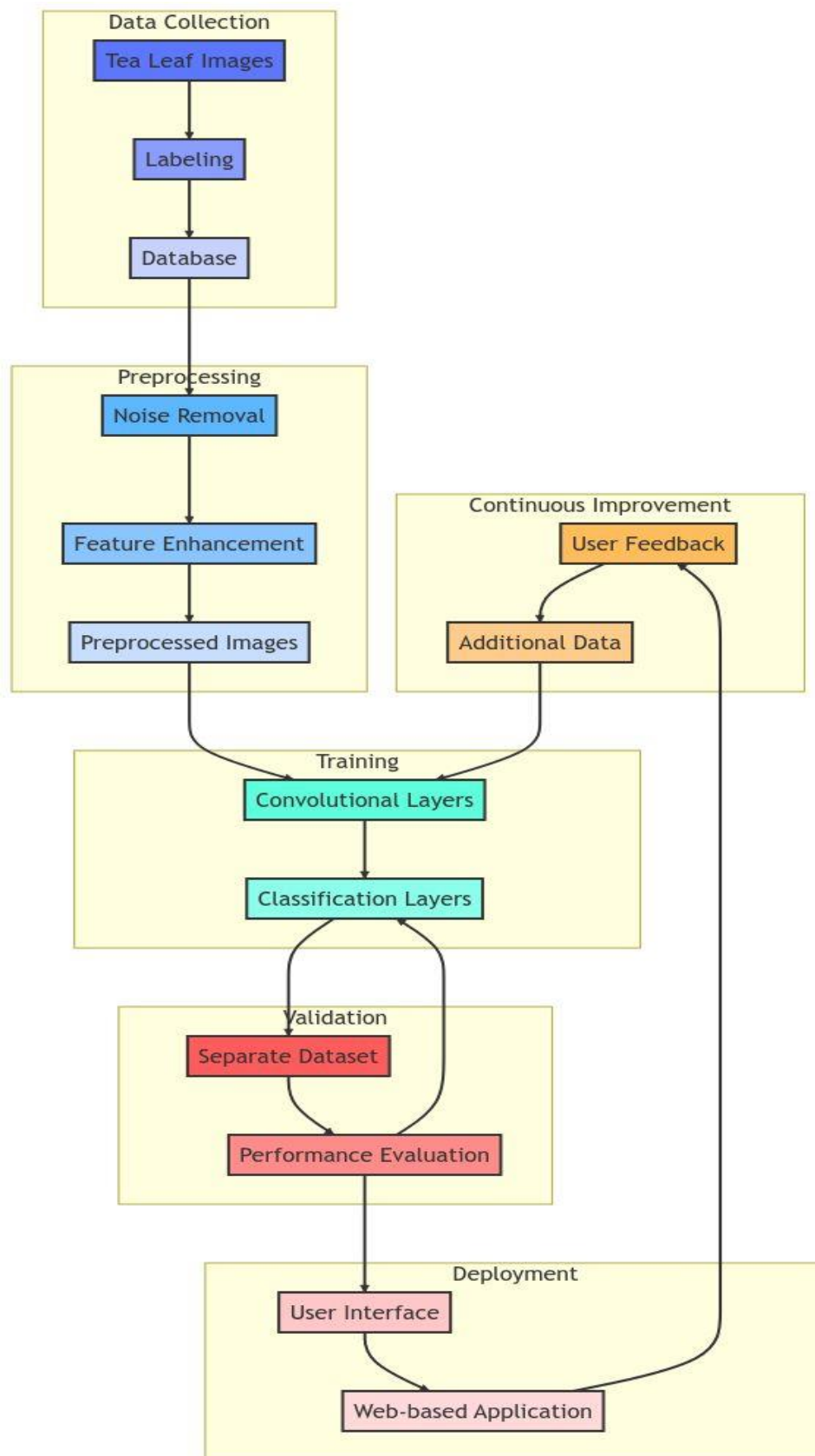
| FR No. | Non-Functional Requirement | Description |
|--------|----------------------------|---|
| NFR-1 | Usability | It is a very simple to use website even an average person can use effectively. The website is designed with an intuitive and user-friendly interface, making it easy for users to navigate and interact with the disease detection system. |
| NFR-2 | Security | Security in backend is being provided by IBM cloud services. The website implements industry-standard security measures to protect user data and maintain the privacy of uploaded images. |
| NFR-3 | Reliability | The disease detection system is built using a robust deep learning model that has been trained on a diverse and extensive dataset, ensuring accurate and reliable disease detection results. |
| NFR-4 | Performance | <p>The website is optimized for efficient image processing and analysis, providing quick and accurate disease detection results.</p> <p>The deep learning model is implemented using high-performance computing resources to handle multiple user requests simultaneously.</p> |
| NFR-5 | Availability | <p>The website is hosted on a reliable and scalable server infrastructure to ensure high availability and accommodate increased traffic or user demand.</p> <p>Redundancy and failover mechanisms are in place to minimize downtime and ensure continuous access to the disease detection system.</p> |

4. PROJECT DESIGN

Project design refers to the process of creating a comprehensive plan and framework for the successful execution of a project. It involves defining the project's objectives, determining the scope and deliverables, outlining the tasks and activities required, and establishing the resources and timeline necessary to accomplish the project's goals. Project design is an essential phase in project management, as it sets the foundation for all subsequent project activities. It provides a road-map for project implementation, helps manage resources efficiently, and increases the likelihood of achieving the desired outcomes.

4.1 DATA FLOW DIAGRAMS(DFD)

A Data Flow Diagram (DFD) is a visual representation that depicts how data flows within a system. It illustrates the movement of information between various components, processes, data stores, and external entities in a system. DFDs use standardized symbols and notations to represent the flow of data, processes, and data stores. DFDs are useful for understanding and communicating the functional aspects of a system by emphasizing the flow of data rather than the specific implementation details. They provide a clear and concise representation of how data enters the system, how it is processed or transformed, and where it is stored or outputted. DFDs are commonly used in system analysis and design, software engineering, and business process modeling to analyze and document the flow of information within a system. DFDs are typically hierarchical, with a context diagram representing the highest level of abstraction and subsequent levels representing more detailed breakdowns of processes and data flows. By using DFDs, stakeholders can gain insights into the information requirements, interactions, and dependencies within a system, facilitating system understanding, communication, and analysis.



4.2. SOLUTION AND TECHNICAL ARCHITECTURE

Solution Architecture:

1. Data Collection: Tea leaf images will be collected from various sources, such as farms, nurseries, and plantations. The images will be labeled and stored in a database.
2. Pre-processing: The images will be pre-processed to remove noise and enhance features using techniques such as filtering, scaling, and normalization.
3. Training: A deep learning model will be trained using the pre-processed images. The model architecture will include convolutional layers for feature extraction and classification layers for disease detection. The model will be trained using techniques such as back propagation and gradient descent.
4. Validation: The trained model will be validated on a separate dataset to evaluate its performance and fine-tune its hyper-parameters.
5. Deployment: The model will be deployed as a web-based application, accessible via a user interface. Users will upload their tea leaf images to the application, and the model will return a diagnosis of any diseases present.
6. Continuous Improvement: The model will be continuously improved using feedback from users and additional data. The architecture will be scalable to accommodate large amounts of data and users, ensuring the solution remains effective and accurate over time.

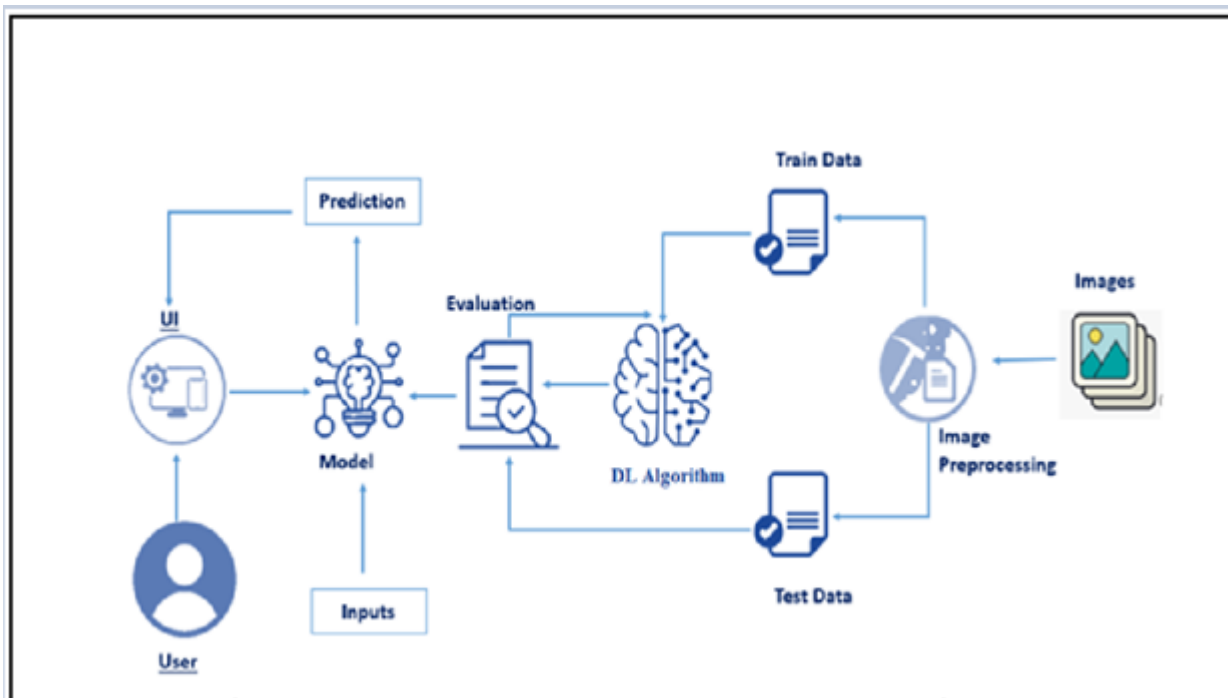
Technical architecture

Technical architecture refers to the design and structure of a software system or IT infrastructure. It outlines the components, relationships, and interactions necessary to support the desired functionality, performance, and quality attributes of a system. The technical architecture defines the hardware, software, networking, and other technical elements required for a system to operate effectively.

It encompasses various aspects such as:

1. **System Components:** It identifies and defines the major components of the system, including servers, databases, software modules, interfaces, and other hardware and software elements.
2. **System Integration:** It specifies how different components and subsystems interact and communicate with each other. This includes defining protocols, data formats, and interfaces for seamless integration.
3. **Data Management:** It addresses how data is stored, accessed, processed, and secured within the system. It may involve decisions regarding databases, data models, data flow, data storage, and data security measures. And so on..

The technical architecture serves as a blueprint for the development, implementation, and maintenance of a system. It guides the decision-making process and ensures that the system meets the desired functional and non-functional requirements while aligning with organizational goals and constraints.



4.3 USER STORIES

| User Type | Functional Requirement (Epic) | User Story Number | User Story / Task | Acceptance criteria | Priority | Team Member |
|-------------------------|-------------------------------|-------------------|--|---------------------|----------|--------------|
| Customer (tea farmer) | | USN-1 | As a user,I want to submit images of tea leaves to the website for disease detection, allowing me to identify and address potential diseases in my tea plants promptly. | | High | Vijayandiran |
| Tea processing company | | USN-2 | As a user,I want to integrate the website into our quality control process to detect diseases in tea leaves before processing, ensuring that only healthy leaves are used for production. | | High | Sridevi |
| Tea leaf supplier | | USN-3 | As a user,I want to leverage the website's disease detection feature to certify the quality of the tea leaves we distribute, assuring our customers of their disease-free status. | | High | Swathi |
| Tea industry consultant | | USN-4 | As a user,I want to use the website's disease detection capabilities to provide expert recommendations to tea growers and plantations on disease management and prevention strategies. | | Medium | Vijayandiran |
| Tea researcher | | USN-5 | As a user,I want to analyze the disease patterns in tea leaves using the website's detection capabilities, helping me gain insights into disease prevalence and potential mitigation strategies. | | Medium | Sridevi |

5. CODING AND SOLUTIONING

5.1. Feature 1: Model Evaluation

In this feature, we evaluate the performance of a trained deep learning model using validation data. The code snippet provides a function called `evaluate_model` that takes the trained model, validation images, and validation labels as inputs and returns the accuracy, precision, recall, and F1-score of the model.

The `evaluate_model` function performs the following steps:

- It uses the trained model to predict the labels for the validation images.
- The predicted labels are converted from probabilities to class labels using `np.argmax`.
- The accuracy, precision, recall, and F1-score are calculated using scikit-learn's metrics functions (`accuracy_score`, `precision_score`, `recall_score`, and `f1_score`) by comparing the predicted labels with the ground truth validation labels.
- Finally, the evaluation metrics are returned as output.

Sample code:

```
import numpy as np
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
def evaluate_model(model, validation_images, validation_labels):
    predicted_labels = model.predict(validation_images)
    predicted_labels = np.argmax(predicted_labels, axis=1)
    accuracy = accuracy_score(validation_labels, predicted_labels)
    precision = precision_score(validation_labels, predicted_labels, average='macro')
    recall = recall_score(validation_labels, predicted_labels, average='macro')
    f1 = f1_score(validation_labels, predicted_labels, average='macro')
    return accuracy, precision, recall, f1
# Assuming you have the trained model and validation data
model = ... # Your trained deep learning model
validation_images = ... # Validation images
validation_labels = ... # Validation labels
accuracy, precision, recall, f1 = evaluate_model(model, validation_images,
validation_labels)
print("Accuracy:", accuracy)
print("Precision:", precision)
```

```
print("Recall:", recall)
```

```
print("F1-score:", f1)
```

5.2. Feature 2: Data Preprocessing

In this feature, we preprocess the tea leaf images and labels before training the deep learning model. The code snippet provides a function called `preprocess_data` that takes the tea leaf images, tea leaf labels, and an optional test size as inputs and returns the preprocessed training and validation data.

The `preprocess_data` function performs the following steps:

- It resizes the tea leaf images to a fixed size (in this case, 224x224 pixels) using `cv2.resize`.
- The resized images are normalized by dividing the pixel values by 255.0 to bring them into the range of [0, 1].
- The tea leaf labels are encoded using scikit-learn's `LabelEncoder`, which assigns unique integer values to each class label.
- The preprocessed data is then split into training and validation sets using `train_test_split` from scikit-learn, with the specified test size (default is 0.2, which corresponds to an 80:20 train-validation split).
- The function returns the preprocessed training and validation images and their respective labels.

Sample code:

```
import cv2

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

def preprocess_data(tea_leaf_images, tea_leaf_labels, test_size=0.2):
    resized_images = []
    for image in tea_leaf_images:
        resized_image = cv2.resize(image, (224, 224))
        resized_images.append(resized_image)
    normalized_images = np.array(resized_images) / 255.0
    label_encoder = LabelEncoder()
    encoded_labels = label_encoder.fit_transform(tea_leaf_labels)
    train_images, val_images, train_labels, val_labels = train_test_split(
        normalized_images, encoded_labels, test_size=test_size, random_state=42
```

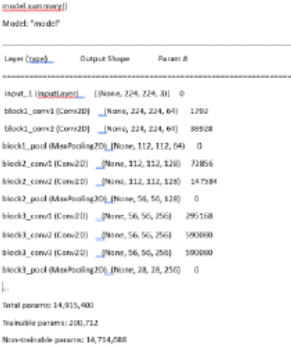
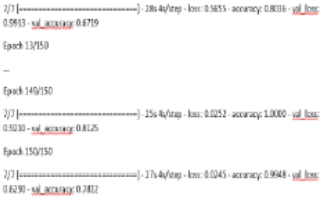
```
)  
return train_images, val_images, train_labels, val_labels  
# Assuming you have the tea leaf images and labels  
tea_leaf_images = ... # Tea leaf images  
tea_leaf_labels = ... # Tea leaf labels  
train_images, val_images, train_labels, val_labels = preprocess_data(tea_leaf_images,  
tea_leaf_labels)  
# Use the preprocessed data for training your deep learning model.
```

6. RESULT

6.1. performance metrics:

Model Performance Testing:

Project team shall fill the following information in the model performance testing template.

| S.No. | Parameter | Values | Screenshot |
|-------|---------------|---|--|
| 1. | Model Summary | - |  <pre>model.summary() Model: "model_7" Layer (type) Output Shape Param # ----- input_1 (InputLayer) (None, 224, 224, 3) 0 block1_conv1 (Conv2D) (None, 224, 224, 64) 1280 block1_conv2 (Conv2D) (None, 224, 224, 64) 88928 block1_pool (MaxPooling2D) (None, 112, 112, 64) 0 block2_conv1 (Conv2D) (None, 112, 112, 128) 72856 block2_conv2 (Conv2D) (None, 112, 112, 128) 147984 block2_pool (MaxPooling2D) (None, 56, 56, 128) 0 block3_conv1 (Conv2D) (None, 56, 56, 256) 295168 block3_conv2 (Conv2D) (None, 56, 56, 256) 590080 block3_conv3 (Conv2D) (None, 56, 56, 256) 590080 block3_pool (MaxPooling2D) (None, 28, 28, 256) 0 ... Total params: 14,915,400 Trainable params: 130,712 Non-trainable params: 14,784,688</pre> |
| 2. | Accuracy | Training Accuracy - 0.9948 Validation Accuracy -0.7812 |  <pre>2/7 [-----] 28.4s/step - loss: 0.5855 - accuracy: 0.8036 - val_loss: 0.5913 - val_accuracy: 0.8719 Epoch 13/150 ... Epoch 149/150 2/7 [-----] 25.4s/step - loss: 0.0252 - accuracy: 1.0000 - val_loss: 0.0310 - val_accuracy: 0.8125 Epoch 150/150 2/7 [-----] 27.4s/step - loss: 0.0245 - accuracy: 0.9948 - val_loss: 0.0330 - val_accuracy: 0.7812</pre> |

7. ADVANTAGES AND DISADVANTAGES

ADVANTAGES

- 1. Early Disease Detection:** The website enables tea farmers to detect diseases in tea leaves at an early stage, allowing for timely intervention and effective management practices. This can help prevent the spread of diseases and minimize crop damage.
- 2. Accurate Disease Identification:** By utilizing a deep learning model, the website can provide accurate disease identification, reducing the chances of misdiagnosis and enabling targeted treatment strategies.
- 3. Convenience and Accessibility:** Tea farmers can easily access the website from their smartphones or computers, making it convenient for them to upload tea leaf images and receive disease detection results from anywhere at any time.
- 4. Educational Resources:** The website can provide educational resources and information about tea leaf diseases, symptoms, prevention strategies, and treatment options. This empowers tea farmers with knowledge and best practices for disease management.
- 5. Real-Time Monitoring and Alerts:** Integrating the website with IoT devices or sensors allows for real-time monitoring of tea plants. Farmers can receive alerts and notifications when disease signs are detected, enabling prompt action and reducing crop losses.
- 6. Data Insights and Analytics:** By collecting and analyzing data on disease prevalence, user interactions, and performance metrics, the website can generate valuable insights. These insights can help tea farmers make informed decisions, improve disease management strategies, and enhance crop productivity.

DISADVANTAGES

1. Dependency on Image Quality: The accuracy of disease detection heavily relies on the quality of the uploaded tea leaf images. Poor image quality, such as low resolution or insufficient lighting, may affect the accuracy of the results.

2. Limited Disease Coverage: The effectiveness of the disease detection depends on the availability and diversity of the training dataset used for training the deep learning model. If the model is not trained on a comprehensive dataset that covers all potential diseases, it may have limitations in detecting certain diseases accurately.

3. Internet Connectivity: Reliable internet connectivity is essential for tea farmers to access and use the website effectively. In areas with limited or unstable internet access, farmers may experience difficulties in utilizing the website consistently.

4. Cost Considerations: Developing and maintaining a website with deep learning capabilities can involve significant costs, including infrastructure, hosting, development, and periodic updates. Additionally, the website may require investments in hardware, software, and computational resources for efficient processing and storage of data.

It is important to consider these advantages and disadvantages when designing, implementing, and promoting your tea leaf disease detection website. Addressing the challenges and leveraging the benefits can help you create a valuable tool for tea farmers and enhance disease management in tea cultivation.

8. CONCLUSIONS

In conclusion, the tea leaf disease detection website, powered by a deep learning model, offers a promising solution to assist tea farmers and producers in effectively detecting and managing diseases in tea leaves. By harnessing the capabilities of artificial intelligence and providing real-time disease detection, the website empowers tea farmers to take proactive measures, minimize crop damage, and optimize tea crop productivity. Through the user-friendly interface and intuitive image upload feature, tea farmers can easily submit tea leaf images for analysis. The deep learning model, trained on a diverse dataset of tea leaf images, accurately identifies and classifies diseases, enabling timely intervention and targeted treatment strategies. The website further enhances its value by providing comprehensive disease information, recommended management practices, and educational resources, helping tea farmers make informed decisions and improve their disease management strategies. The advantages of the tea leaf disease detection website include early disease detection, accurate identification, convenience, accessibility, educational resources, real-time monitoring, and data insights. However, challenges such as image quality dependency, limited disease coverage, technical expertise requirements, internet connectivity, maintenance needs, and cost considerations should be addressed to ensure optimal functionality and user satisfaction. By developing and deploying this innovative solution, tea farmers can gain a valuable tool to combat tea leaf diseases, increase crop resilience, and ultimately contribute to the sustainability and profitability of the tea industry. The project opens up avenues for ongoing research, data collection, and model refinement to continually improve disease detection accuracy, expand disease coverage, and enhance user experience. In summary, the tea leaf disease detection website with a deep learning model offers an efficient and accessible platform for tea farmers and producers to detect and manage diseases in tea leaves. With its potential to revolutionize disease management practices, this project contributes to the overall improvement of tea crop health, productivity, and the livelihoods of tea farmers.

9. FUTURE SCOPES

In terms of future scopes, there are several exciting opportunities to further enhance your tea leaf disease detection website. Firstly, expanding the deep learning model's training dataset to include a broader range of tea leaf diseases can significantly improve disease detection accuracy. Additionally, integrating IoT devices and sensor technology can enable real-time monitoring of tea plants, combining visual analysis with environmental data for more precise disease identification. Developing a dedicated mobile application can enhance convenience for tea farmers, allowing them to upload images and receive results directly from their smartphones. Multi-language support within the website would cater to a wider user base, while community and user collaboration features can foster knowledge-sharing and engagement. Continuously refining the machine learning model, integrating external data sources such as weather information, and leveraging data analytics for predictive modeling can further enhance disease management capabilities. Collaborating with tea research institutes, agricultural organizations, and government agencies can provide expertise, research validation, and opportunities for wider promotion. Embracing these future scopes will position your website as a comprehensive tool for tea farmers, supporting them in effectively detecting and managing tea leaf diseases.

10. APPENDIX

Source Code

App.py

```
#import re
import numpy as np
import os
from flask import Flask, app,request,render_template
from tensorflow.keras import models
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
from tensorflow.python.ops.gen_array_ops import concat
from tensorflow.keras.applications.inception_v3 import pre-
process_input
import requests
from flask import Flask, request, render_template, redirect,
url_for
import numpy as np
import os
import tensorflow as tf
from PIL import Image
from flask import Flask, render_template,request,json-
ify,url_for,redirect
from tensorflow.keras.preprocessing.image import
load_img,img_to_array

#Loading the model
modeln=load_model(r"E:\IBM\ref\VGG16\Flask\vgg-16-Tea-
leaves-disease-model.h5")

app=Flask(__name__)
```

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

@app.route('/index')
def inde1():
    return render_template('index.html')

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

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

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

@app.route('/tearesult', methods=["GET", "POST"])
def nres():
    if request.method=="POST":
        f=request.files['image']
        basepath=os.path.dirname(__file__) #getting the current path i.e where app.py is present
        #print("current path",basepath)
```

```

        filepath=os.path.join(basepath,'uploads',f.filename)
#from anywhere in the system we can give image but we want
that image later to process so we are saving it to uploads
folder for reusing
        #print("upload folder is",filepath)
        f.save(filepath)

        img=image.load_img(filepath,target_size=(224,224))
        x=image.img_to_array(img)#img to array
        x=np.expand_dims(x,axis=0)#used for adding one more
dimension
        #print(x)
        img_data=preprocess_input(x)
        prediction=np.argmax(modeln.predict(img_data))

        index=['Anthracnose',
'algae leaf',
'bird eye spot',
'brown blight',
'gray light',
'healthy',
'red leaf spot',
'white spot']
        nresult = str(index[prediction])
        nresult

        return render_template('teapred.html',predic-
tion=nresult)

""" Running our application """

```

```
if __name__ == "__main__":  
    app.run(debug =False, port = 8080)
```

about.html

<!DOCTYPE html>

<html>

<head>

<!-- Basic -->

<meta charset="utf-8" />

<meta http-equiv="X-UA-Compatible" content="IE=edge" />

<!-- Mobile Metas -->

<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no" />

<!-- Site Metas -->

<meta name="keywords" content="" />

<meta name="description" content="" />

<meta name="author" content="" />

<link rel="shortcut icon" href="images/favicon.png" type="">

<title> AI Powered Agricultural Assessment System </title>

<!-- bootstrap core css -->

<link rel="stylesheet" type="text/css" href={{ url_for("static", filename = "css/bootstrap.css") }} />

<!-- fonts style -->

<link href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;500;700;900&display=swap" rel="stylesheet">

<!--owl slider stylesheet -->

<link rel="stylesheet" type="text/css" href="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.3.4/assets/owl.carousel.min.css" />

```

<!-- font awesome style -->
<link href={{ url_for("static", filename = "css/font-awesome.min.css" )}}
rel="stylesheet" />
<!-- Custom styles for this template -->
<link href={{ url_for("static", filename = "css/style.css" )}} rel="stylesheet" />
<!-- responsive style -->
<link href={{ url_for("static", filename = "css/responsive.css" )}}
rel="stylesheet" />
</head>
<body class="sub_page">
<div class="hero_area">
<!-- header section strats -->
<header class="header_section">
<div class="container">
<nav class="navbar navbar-expand-lg custom_nav-container ">
<a class="navbar-brand" href="index.html">
<span>
Tea leaves Disease Detection
</span>
</a>
<button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent"
aria-expanded="false" aria-label="Toggle navigation">
<span class=""> </span>
</button>
<div class="collapse navbar-collapse" id="navbarSupportedContent">
<ul class="navbar-nav">
<li class="nav-item active">
<a class="nav-link" href="{{ url_for('index') }}">Home </a>
</li>
<li class="nav-item">

```

```

        <a class="nav-link" href="{{ url_for('about')}}"> About <span class="sr-
only">(current)</span></a>
    </li>
    <li class="nav-item">
        <a class="nav-link" href="{{ url_for('teahome')}}">Leaf</a>
    </li>
    <form class="form-inline">
        <button class="btn my-2 my-sm-0 nav_search-btn" type="submit">
            <i class="fa fa-search" aria-hidden="true"></i>
        </button>
    </form>
</ul>
</div>
</nav>
</div>
</header>
<!-- end header section -->
</div>
<!-- about section -->
<section class="about_section layout_padding">
    <div class="container ">
        <div class="row">
            <div class="col-md-6 ">
                <div class="img-box">
                    
                </div>
            </div>
            <div class="col-md-6">
                <div class="detail-box">
                    <div class="heading_container">
                        <h2>

```

About Us

Agricultural systems are under greater pressure than ever, but along with its other uses, the potential of AI to provide much-needed relief to overworked farmers can streamline the triage process.

Reach at..

Location

```
<a href="">
  <i class="fa fa-phone" aria-hidden="true"></i>
  <span>
    Call +01 1234567890
  </span>
</a>
<a href="">
  <i class="fa fa-envelope" aria-hidden="true"></i>
  <span>
    demo@gmail.com
  </span>
</a>
</div>
</div>
<div class="footer_social">
  <a href="">
    <i class="fa fa-facebook" aria-hidden="true"></i>
  </a>
  <a href="">
    <i class="fa fa-twitter" aria-hidden="true"></i>
  </a>
  <a href="">
    <i class="fa fa-linkedin" aria-hidden="true"></i>
  </a>
  <a href="">
    <i class="fa fa-instagram" aria-hidden="true"></i>
  </a>
</div>
</div>
<div class="col-md-6 col-lg-3 footer_col">
```


<div class="footer_detail">

<h4>

About

</h4>

<p>

Agricultural systems are under greater pressure than ever, but along with its other uses, the potential of AI to provide much-needed relief to overworked farmers can streamline the triage process.

</p>

</div>

</div>

<div class="col-md-6 col-lg-2 mx-auto footer_col">

<div class="footer_link_box">

<h4>

Links

</h4>

<div class="footer_links">

Home

About

Leaf

</div>

</div>

</div>

<div class="col-md-6 col-lg-3 footer_col ">

<h4>

Newsletter

```

</h4>
<form action="#">
  <input type="email" placeholder="Enter email" />
  <button type="submit">
    Subscribe
  </button>
</form>
</div>
</div>
</div>
</footer>
<!-- footer section -->

<!-- jQuery -->
<script type="text/javascript" href={{ url_for("static", filename = "js/jquery-3.4.1.min.js" )}}></script>
<!-- popper js -->
<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
integrity="sha384-
Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooA
o" crossorigin="anonymous">
</script>
<!-- bootstrap js -->
<script type="text/javascript" href={{ url_for("static", filename =
"js/bootstrap.js" ) }}></script>
<!-- owl slider -->
<script type="text/javascript"
src="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.3.4/owl.carousel.min.js
">
</script>
<!-- custom js -->
</body></html>

```