PLANT HEALTH MANAGEMENT: CNN FOR DISEASE DETECTION TREATMENT RECOMMENDATIONS

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

YAMBALURU HARSHA DEEPAK	(21UECS0685)	**(20507)**
NIRANJAN PEEKA	(21UECS0424)	**(19916)**
BANDARI HEMANTH	(21UECS0069)	**(19923)**

Under the guidance of
Mr. R. VINOTH KUMAR,M.Tech.,
ASSISTANT PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

PLANT HEALTH MANAGEMENT: CNN FOR DISEASE DETECTION TREATMENT RECOMMENDATIONS

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

YAMBALURU HARSHA DEEPAK	(21UECS0685)	**(20507)**
NIRANJAN PEEKA	(21UECS0424)	**(19916)**
BANDARI HEMANTH	(21UECS0069)	**(19923)**

Under the guidance of
Mr. R. VINOTH KUMAR,M.Tech.,
ASSISTANT PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

CERTIFICATE

It is certified that the work contained in the project report titled PLANT HEALTH MANAGEMENT: CNN FOR DISEASE DETECTION TREATMENT RECOMMENDATIONS by YAMBALURU HARSHA DEEPAK (21UECS0685), NIRANJAN PEEKA (21UECS0424), BANDARI HEMANTH (21UECS0069) has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor
Mr. R. VINOTH KUMAR, M.Tech.,
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

Signature of Professor In-charge
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)

YAMBALURU HARSHA DEEPAK

Date: / /

(Signature)

NIRANJAN PEEKA

Date: / /

(Signature)

BANDARI HEMANTH

Date: / /

APPROVAL SHEET

This project report entitled PLANT HEALTH MANAGEMENT: CNN FOR DISEASE DETECTION TREATMENT RECOMMENDATIONS by (YAMBALURU HARSHA DEEPAK) (21UECS0685), (NIRANJAN PEEKA) (21UECS0424), (BANDARI HEMANTH) (21UECS0069) is approved for the degree of B.Tech in Computer Science & Engineering.

Examiners **Supervisor**

Mr. R. VINOTH KUMAR, M.Tech.,

Date: / /

Place: Chennai

ACKNOWLEDGEMENT

We express our deepest gratitude to our respected **Founder Chancellor and President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO),D.Sc., Foundress President Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.** Chairperson Managing Trustee and Vice President.

We are very much grateful to our beloved **Vice Chancellor Prof. S. SALIVAHANAN,** for providing us with an environment to complete our project successfully.

We record indebtedness to our **Professor & Dean, Department of Computer Science & Engineering, School of Computing, Dr. V. SRINIVASA RAO, M.Tech., Ph.D.,** for immense care and encouragement towards us throughout the course of this project.

We are thankful to our **Head, Department of Computer Science & Engineering, Dr.M.S. MURALI DHAR, M.E., Ph.D.,** for providing immense support in all our endeavors.

We also take this opportunity to express a deep sense of gratitude to our **Internal Supervisor Mr. R. VINOTH KUMAR, Assistant Professor, M.Tech.,** for his cordial support, valuable information and guidance, he helped us in completing this project through various stages.

A special thanks to our **Project Coordinators Mr. V. ASHOK KUMAR, M.Tech., Ms. U.HEMAVATHI, M.E., Ms. C. SHYAMALA KUMARI, M.E.,** for their valuable guidance and support throughout the course of the project.

We thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

YAMBALURU HARSHA DEEPAK (21UECS0685)

NIRANJAN PEEKA (21UECS0424)

BANDARI HEMANTH (21UECS0069)

ABSTRACT

Plant diseases pose a significant threat to crop yields and farmers' livelihoods. Identifying the specific cause and selecting the appropriate treatment, which is often laden with chemicals, can be a daunting and confusing process. This is where our innovative system comes into play, offering a helping hand by blending the power of artificial intelligence with the wisdom of organic solutions.Imagine pointing your smartphone at a diseased leaf. Inside, the system uses a trained Convolutional Neural Network (CNN) to analyze the image, meticulously comparing it to a vast database of both diseased and healthy leaves. This CNN, a master of pattern recognition, quickly identifies the culprit, revealing the disease that's harming your plant. But the journey doesn't end there. With the diagnosis in hand, the system taps into a rich repository of knowledge curated by agricultural experts. This knowledge base, full of organic treatment methods, provides a personalized solution tailored to the specific disease and plant species.

The benefits of this system go beyond individual plant recovery. By accurately diagnosing diseases and offering organic treatments, it helps prevent the spread of infection and conserves precious crops. Choosing organic methods not only protects the health of plants but also preserves the soil, water, and the food we ultimately consume. Importantly, by providing farmers with accurate diagnoses and accessible treatment options, the system fosters greater confidence and independence in managing their agricultural endeavors.We envision a future where this system continuously evolves, expanding its knowledge base with new organic methods and reaching a broader range of plant species. With convenient mobile applications, diagnosis and treatment guidance will be available directly in the field, empowering farmers in real-time. Eventually, recommendations can be tailored even further to suit the unique conditions of each farm and ecosystem. This system is a beacon of hope, demonstrating the power of merging technology with tradition. It paves the way for a future where crops are healthy, farmers prosper, and the environment flourishes in a harmonious balance between humanity and nature.

Keywords: Convolutional Neural Network(CNN), Artificial Intelligence(AI), Deep Learning, Image Recognition, Organic Treatment, Sustainable Agriculture.

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

CNN	Convolution Neural Network
GDP	Gross Domestic Product
PA	Precision Agriculture
SSI	Structural Similarity Index
SSMD	Single Shot Multi Book Detector
UI	User Interface

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Introduction

Plant diseases pose a persistent threat to agriculture, causing massive crop losses and jeopardizing food security. Early and accurate diagnosis is crucial for mitigating these harms, but traditional methods can be time-consuming, subjective, and often require chemical solutions with undesirable environmental and health consequences. In response, we introduce a novel system that leverages the power of artificial intelligence and organic wisdom to revolutionize plant leaf disease detection and treatment.Imagine capturing a photo of your ailing plant with your smartphone. Our system instantly analyzes the image using a deep learning model that recognizes disease patterns with remarkable accuracy. This diagnosis unlocks a treasure trove of organic treatment methods curated by agricultural experts, tailored to the specific disease and plant species. With clear, accessible instructions utilizing readily available materials, empowering you to effectively treat your plants while promoting sustainable practices. This is just the beginning, as we envision a future where the system continuously learns, expands its knowledge base, and adapts to real-time field applications, ultimately fostering a healthier future for agriculture and the environment.

1.2 Aim of the project

This novel system aims to revolutionize plant health management by seamlessly merging cutting-edge technology with time-tested organic wisdom. Our primary goal is to empower farmers and agricultural practitioners with powerful tools for early and accurate disease detection. By harnessing the power of deep learning, the system pinpoints specific diseases from leaf images with high accuracy, eliminating the need for subjective or delayed traditional methods. Beyond diagnosis, we strive to promote sustainable and environmentally friendly practices by offering a

vast repository of organic treatment solutions. Tailored to the diagnosed disease and plant species, these readily available and accessible methods 1 allow farmers to effectively combat plant ills without relying on harmful chemicals. This focus on organic solutions not only safeguards crop health but also protects soil,water, and ultimately, the food we consume. In essence, this system aims to transform plant disease management from a reactive struggle into a proactive and sustainable endeavor, fostering a healthier future for agriculture and our planet. By empowering farmers with accurate diagnosis and effective organic solutions, we pave the way for increased food security, environmental protection, and thriving agricultural communities.

1.3 Project Domain

It resides within the realm of artificial intelligence (AI), specifically deep learning and computer vision. The model's ability to accurately detect plant diseases from images stems from the Convolutional Neural Network (CNN), a powerful AI algorithm adept at image recognition and pattern analysis. This domain encompasses the technical aspects of the system, including model development, training, and image processing capabilities. Simultaneously, the system firmly anchors itself within the domain of agriculture and plant pathology. Its knowledge base, curated by experts, encapsulates a rich understanding of plant diseases, their symptoms, and effective organic treatment methods. This domain encompasses plant biology, disease mechanisms, and the practical application of organic solutions in agricultural settings.The blend of these domains bridges the gap between laboratory-developed AI and real-world agricultural challenges. By combining technological prowess with ecological empathy, the system promotes a sustainable approach to plant health management.

1.4 Scope of the Project

Within disease detection, the system aims for broad accessibility and inclusivity. By leveraging smartphone image capture and a user-friendly interface, it caters to farmers and individuals with varying technical expertise. The deep learning model is trained on a diverse dataset of leaf images across multiple plant species and disease categories, aiming for accurate identification across a wide range. However, ongoing research on expanding the knowledge base to cover less common plant species and diseases remains crucial. For organic treatment recommendations, the focus lies

on practicality and sustainability. The system prioritizes readily available materials and accessible methods, empowering farmers to implement solutions locally without relying on expensive or inaccessible resources. Additionally, the recommendations cater to specific plant sensitivities and environmental factors, striving for personalized and effective treatment strategies. However, further research is needed to integrate real-time field data and adapt recommendations to dynamically changing conditions. Overall, the scope of this system extends beyond individual farms and aims for broader impact. By enabling early disease detection, promoting sustainable solutions, and empowering farmers, it seeks to contribute to a healthier agricultural landscape, enhanced food security, and a more harmonious relationship between human activity and the environment.

Chapter 2

LITERATURE REVIEW

1. Pal, Arunangshu, and Vinay Kumar (2023): AgriDet uses image preprocessing to focus on key features, a unique GrabCut algorithm to handle occlusions, a hybrid deep learning network combining pre-trained knowledge with custom analysis, and dropout and Kohonen layers to improve accuracy and generalizability. This leads to robust detection and classification of disease severity levels .

2. Ahmed, Imtiaz, and Pramod Kumar Yadav (2023): Farmers can use AI to detect plant diseases by analyzing photos taken on smartphones. The technology uses machine learning to dissect leaf color, texture, and veins, providing accurate disease diagnoses, which empowers farmers to take swift action, potentially saving crops and boosting yields .

3. Pandian et al. (2022): A five-layer Convolutional Neural Network (CNN) is proposed to identify 26 plant diseases from leaf images. This model achieves a near-perfect 99.23 percent training accuracy and 98.93 percent validation accuracy. It uses image manipulation techniques to boost data diversity, offering rapid diagnoses and interventions .

4. Deepalakshmi et al. (2021): This paper presents a CNN model for plant disease detection from leaf images, achieving 97.27 percent accuracy. While it shows promise for automatic, early disease detection, more diverse data and reduced computational cost are required for broader application .

5. Alguliyev et al. (2021): A deep learning model combining CNNs and GRUs is proposed, achieving a 99.38 percent accuracy in plant disease detection. It captures temporal dependencies in leaf images and generalizes well to unseen species/diseases. However, challenges like data availability and computational cost still exist .

6. Ajra et al. (2020): This work combines image processing and a CNN to achieve over 90 percent accuracy in plant disease detection for Bangladeshi crops. It also recommends preventive measures based on diagnosis, though challenges remain in enlarging the dataset and integrating domain knowledge for real-time applications .

7. Nagaraju et al. (2020): A comprehensive review of deep learning techniques for plant disease detection, analyzing CNN architectures, data sources, preprocessing methods, and evaluation metrics. It discusses promising future directions and emphasizes the need for integrating plant pathology knowledge for accurate, practical disease detection solutions .

8. Arsenovic et al. (2019): They propose a two-stage CNN architecture, Plant Disease Net, trained on a large, real-world dataset with data augmentation to tackle limitations of deep learning for plant disease detection. This model achieves 95.5 percent accuracy and can handle multiple diseases per image .

9. Saleem et al. (2019): This deep learning approach uses a CNN model trained on the PlantVillage dataset, achieving 99.35 percent accuracy, outperforming previous methods. It paves the way for automated, accurate disease identification, although challenges like data availability and computational cost remain .

10. Ferentinos (2018): This paper reviews various deep learning models, like VGG16, for plant disease detection, achieving up to 99.53 percent accuracy. It highlights the potential for automated, early disease diagnosis while pointing out challenges like data availability and suggesting further research .

Chapter 3

PROJECT DESCRIPTION

3.1 Existing System

Plants are considered as energy supply to mankind. Plant diseases can affect the agriculture which can be resulted in to huge loss on the crop yield. Therefore, leaf diseases detection plays a vital role in agricultural field. However, it requires large manpower, more processing time and extensive knowledge and skills about plant diseases. Hence, machine learning comes in play in the detection of diseases in plant leaves as it analyzes the data from various areas, and classifies it into one of the predefined set of classes. The features and properties like color, intensity and dimensions of the plant leaves are considered as a major fact for classification and the various types of plant diseases and different classification techniques in machine learning that are used for identifying diseases in different plants leaf.

Advantages:

- **Early Detection:**Automated systems enable the early detection of crop diseases,often before visible symptoms manifest. Early identification allows farmers to implement timely interventions, preventing the further spread of diseases and reducing crop losses.
- **Accuracy and Precision:**Utilization of advanced technologies, such as computer vision and machine learning, enhances the accuracy and precision of disease detection.These systems can differentiate between various diseases and provide more targeted recommendations for management.
- **Efficiency:**Automated crop disease detection systems operate efficiently, processing large amounts of data quickly and providing rapid results. This efficiency is crucial for farmers who need timely information to make informed decisions.

Disadvantages:

- **Dependency on Technology:**Farmers relying heavily on technology may face challenges if the system malfunctions or experiences technical issues. Dependence on complex technology may pose a barrier for those with limited access or familiarity with such tools.
- **Dependency on Connectivity:** Some systems may rely on internet connectivity for real-time analysis, which could be a limitation in remote or areas with poor connectivity.

3.2 Proposed System

The proposed web application is designed to be user-friendly, allowing anyone without programming knowledge to access information about plant diseases. It aims to predict plant diseases using various Convolutional Neural Network (CNN) algorithms. The methodology involves experimental analysis, with samples collected from 38 different plants, including those with common diseases like Tomato, Grape, and Apple, as well as Healthy Leaves. The database comprises varying numbers of images for each disease, which are used to classify database images and input images. Key attributes analyzed in the images include shape and texture-oriented features.

Advantages: Early Detection, User-Friendly Interface, It saves time, It produces efficient results.

3.3 Feasibility Study

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

3.3.1 Economic Feasibility

The work evaluates the economic impact the system might have on plant cultivation. Given the company's limited budget for research and development, all ex-

penditures needed to be justified. By relying on freely available technologies and purchasing only the customized components, the system was developed within budget..

3.3.2 Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client and the detection of leaf disease. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this detection by using Convolutional Netural Network

3.3.3 Social Feasibility

The aspect of study is to check the detect the leaf disease by using Convolutional Netural Network. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it has a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the 8 user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final to test the leaf detectionin the system by the Convolutional Netural Network.

3.4 System Specification

3.4.1 Hardware Specification

• System : Intel i3 or Above
• Hard Disk : 40 GB.
• Monitor : 14’ Colour Monitor Mouse : Optical Mouse.
Minimum of 4 GB RAM for smooth application performance.

3.4.2 Software Specification

• Operating system : Windows7.
• Front end : Html,Css.
• Back end : Python IDLE.

3.4.3 Standards and Policies

- **Visual Studio Code (VSCode):** Visual Studio Code (VSCode) is a popular and adaptable integrated development environment (IDE) known for its versatility and support for a wide range of programming languages, including Python, which is widely used in Machine Learning (ML) applications. It is available across Windows, Linux, and MacOS platforms, providing an intuitive interface that streamlines coding tasks. The IDE's extensibility is a key feature, allowing developers to customize their environment with a plethora of extensions available from the marketplace, making it particularly suitable for ML projects. The compatibility with Python-based ML frameworks and the integration with Jupyter notebooks make VSCode a preferred choice for many developers. These features not only boost coding efficiency but also align with contemporary coding practices, contributing to a smooth and productive development experience.

- **Anaconda Prompt:** Anaconda Prompt is a command-line interface designed specifically for managing Machine Learning (ML) modules. It is available across Windows, Linux, and MacOS, providing a variety of integrated development environments (IDEs) to facilitate coding tasks. Anaconda also features a navigator to help users manage their projects and environments. With the option to implement user interfaces in Python, Anaconda Prompt serves as a versatile tool for developers working on ML projects.
  Standard Used: ISO/IEC 27001

- **Jupyter:** Jupyter is an open-source web application that allows users to create and share documents containing live code, equations, visualizations, and narrative text. It is a valuable tool for data cleaning and transformation, numerical simulation, statistical modeling, data visualization, and machine learning. This flexibility makes Jupyter a popular choice among developers and data scientists for creating interactive notebooks and collaborative projects.
  Standard Used: ISO/IEC 27001

Chapter 4

METHODOLOGY

4.1 Plant Health Management Architecture

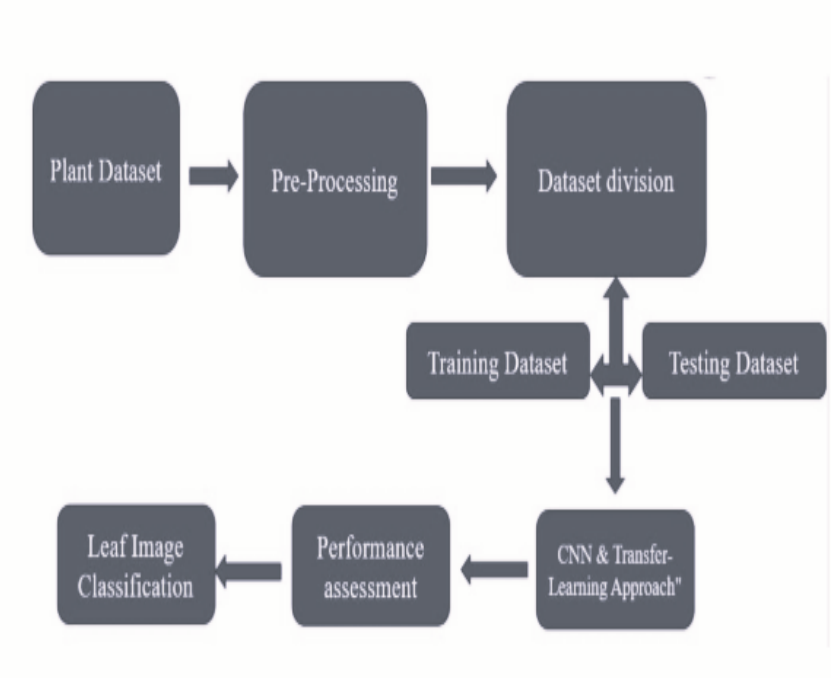

Figure 4.1: **Plant Health Management Architecture**

The above diagram represents the general architecture of the proposed model which helps us to understand the working of the the project. This model consists of plants dataset. From the plant dataset the further processing take place pre-processing. And it has division in the data set. The Dataset division has classified into two different types of data. Training Data and Testing Data. From these Data it performs Convolutional Netural Networks for disease detection of leaf in it.The performance assessment takes place in the leaf.Then classification of leaf images in the given plant Data set in the model of architecture diagram.

4.2 Design Phase

4.2.1 Data Flow Diagram

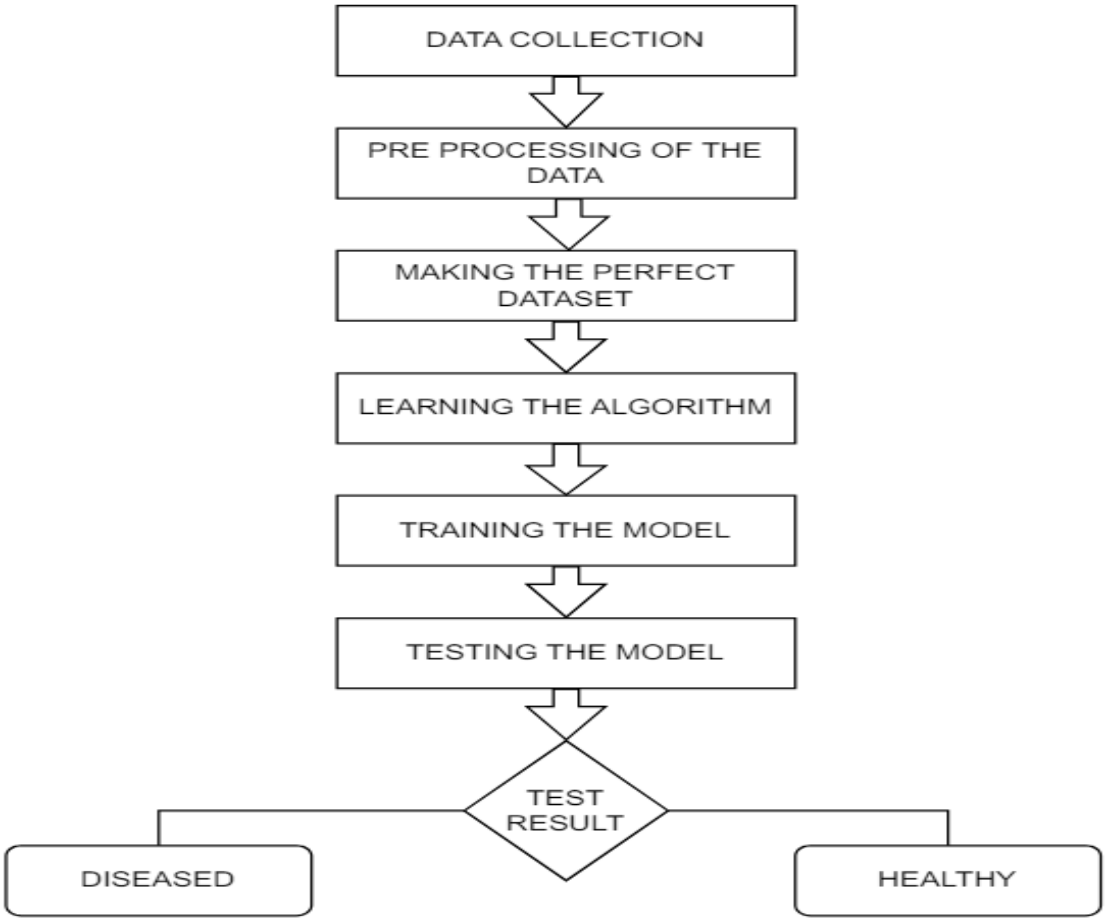

Figure 4.2: **Data Flow Diagram**

The above Figure shows the data flow diagram serves as a visual representation of how data moves through the system, particularly in the context of an information system. The diagram illustrates the flow of data within the model. The initial phase involves gathering data from multiple sources, including farmers, online resources, and agriculture organizations. Once the necessary data is collected, a pre-processing step is implemented to refine and create an optimal dataset. Following the model training phase, diseased leaf data is inputted for testing, and the resulting classification determines whether the leaf is categorized as healthy or unhealthy. This systematic flow delineates the path of data throughout the entirety of the project process.

4.2.2 Use Case Diagram

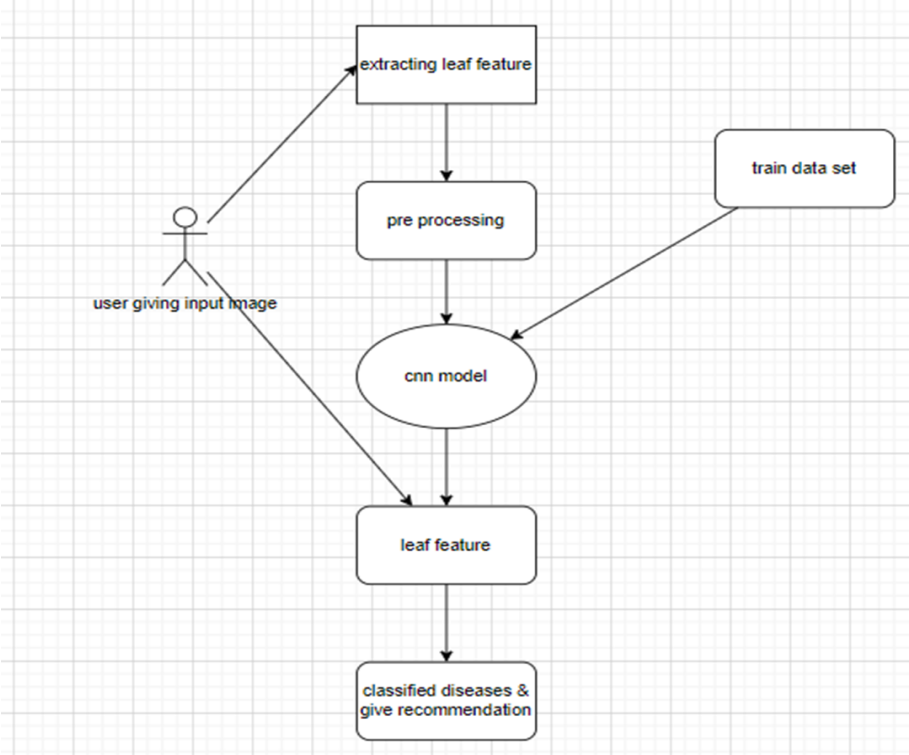

Figure 4.3: **Use Case Diagram**

The above Figure is the use case diagram presents a visual representation specifically designed to illustrate user interactions with a system. The depicted figure outlines various use cases and types of user interactions within the system, providing an informative overview. This diagram is particularly valuable for capturing the essential functionalities and interactions that users engage in while interacting with the system. The use cases prominently feature a user collecting the dataset, utilizing it to train the model with a training dataset, and conducting tests on the model. These interactions are pivotal to the system's functionality, collectively contributing to a comprehensive understanding of the system's behavior and user engagement. Importantly, the use case diagram, as part of a broader set of diagrams, plays a crucial role in conveying how the system functions and the value it provides to users.

4.2.3 Class Diagram

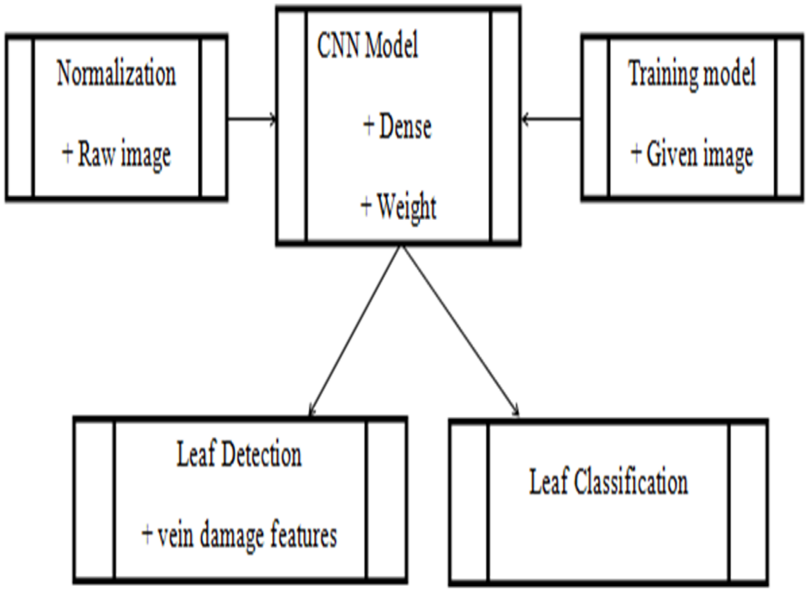

Figure 4.4: **Class Diagram**

The above Figure 4.4 shows the class diagram outlines a sequence of operations for analyzing leaves through a machine learning workflow. It begins with ”Normalization,” where leaf data is prepared for processing. Following this, a ”CNN Model” applies Convolutional Neural Networks to extract features from the normalized data. Subsequently, a ”Training Model” is employed, indicating a stage where the system is trained to recognize specific leaf characteristics. Branching from the CNN Model are two distinct functions: ”Leaf Detection,” which likely pinpoints the presence of leaves in the data, and ”Leaf Classification,” which categorizes the leaves, possibly by type or health status. This setup suggests an automated system designed for botanical analysis or agricultural monitoring.

4.2.4 Sequence Diagram

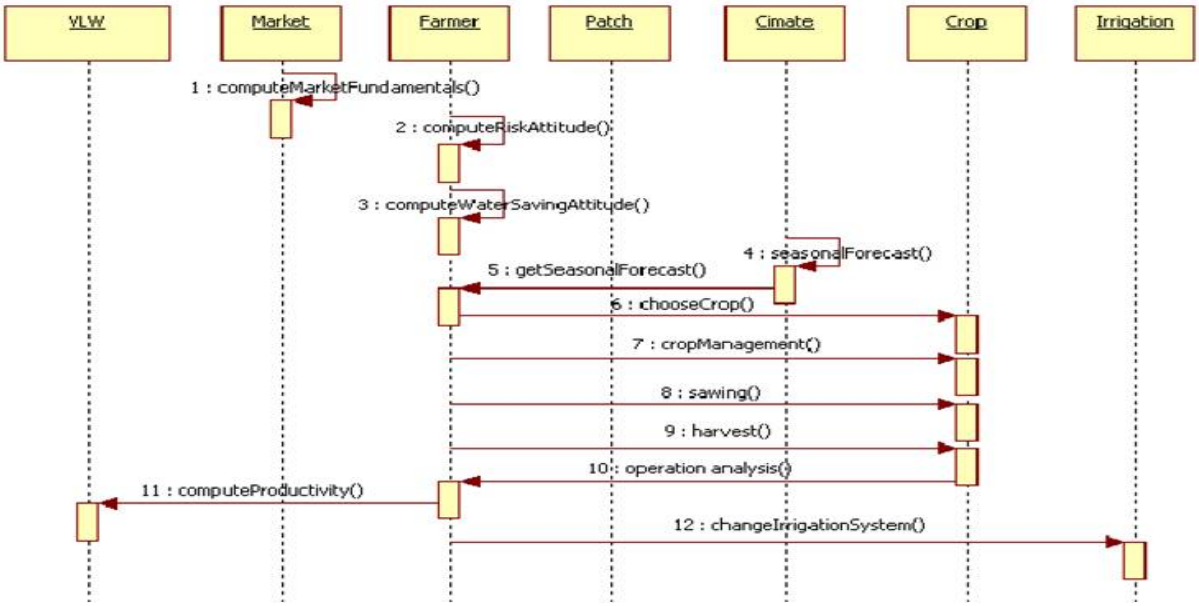

Figure 4.5: **Sequence Diagram**

The above Figure 4.5 represents sequence diagram serves as a comprehensive representation of the entire process unfolded in a sequential order. Initially, the user initiates the process by uploading the input image. Subsequently, the sequence unfolds with crucial steps such as image preprocessing and segmentation. Following these pre-processing steps, the model comes into play, extracting features from the provided leaf data. The sequence of operations encapsulates the extraction of features and the subsequent classification of the leaf data, determining whether it is healthy or not. In instances where the leaf is identified as unhealthy, the model extends its functionality by offering treatment suggestions aimed at effectively addressing the identified issues and enhancing the overall yield of the crop. Essentially, the sequence diagram captures the dynamic interactions between components or objects in the system over time, providing a clear and sequential depiction of the entire process from image upload to feature extraction and final classification with associated treatment recommendations

4.2.5 Collaboration diagram

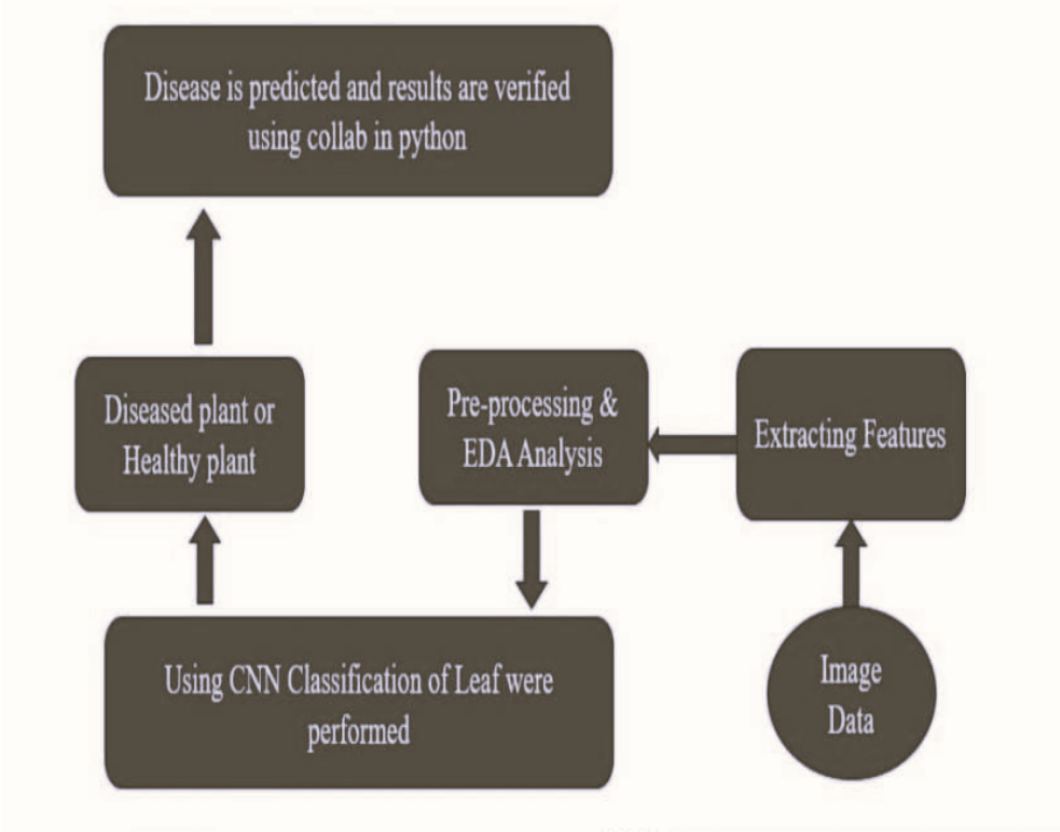

Figure 4.6: **Collaboration Diagram.**

The above Figure 4.6 outlines the procedure for detecting diseases in plant leaves through the implementation of a Convolutional Neural Network (CNN). Commencing with the user inputting an image of a tomato leaf, the initial step involves preprocessing the image to ensure uniformity in size and quality. Subsequently, the CNN undertakes the extraction of pertinent features from the image, crucial for identifying various types of diseases affecting plant leaves. The features extracted play a pivotal role in training the CNN model, enabling it to categorize images into distinct classes: healthy, and those affected by specific diseases such as early blight or late blight. In the final phase, the user is furnished with a diagnosis, accompanied by recommendations for effectively treating the identified plant leaf disease.

4.2.6 Activity Diagram

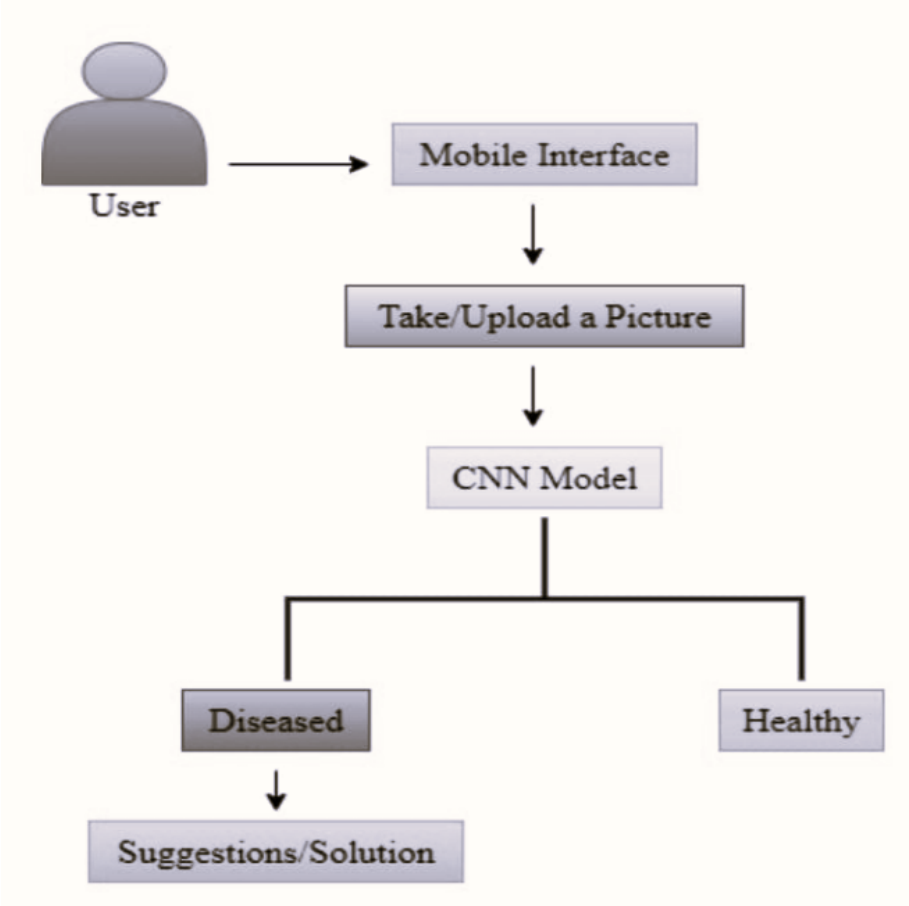

Figure 4.7: **Activity Diagram**

The above Figure represents the activity diagram as a step-by-step guide for a user who wants to check if their plant leaves are healthy. First, they upload a picture of the leaves. Then, the system does some initial processing on the image. After that, a smart model looks at the picture, figures out the important features of the leaves, and decides if they are healthy or not. If there's a problem, like a disease, the model suggests ways to treat it and improve the crop. The activity diagram is like a flowchart showing the smooth flow from one step to another in this entire process,making it clear and easy to follow how everything works together.

4.3 Algorithm & Pseudo Code

4.3.1 Evolved Convolutional Network

1. Initialize the ECN: Start the Evolved Convolutional Network algorithm. Prompt the user to select or upload a dataset of plant images for training. This dataset will be used to train the model to detect plant diseases.
2. Prepare the Training Data: Label the dataset by assigning appropriate categories to each image (e.g., healthy or diseased). Normalize the features to maintain consistent data scaling, and convert the labels into a categorical format. This step prepares the dataset for effective training with the ECN.
3. Split the Data into Features and Labels: Separate the dataset into two parts—features (X), which are the characteristics of the images used for training, and labels (Y), which represent the categories. This split is essential for training a Convolutional Neural Network.
4. Set Up Training and Validation Directories: Define the directories for the training and validation datasets. This separation allows for effective training and validation, reducing the risk of overfitting and ensuring the model generalizes well.
5. Train the Model: Use the ECN to train a model with the training dataset. Validate the model with the validation dataset to check for accuracy and adjust model parameters as needed.
6. Save the Trained Model: After training, save the model to a specific location for future use. This allows for further testing or deployment in real-world applications.
7. Test the Model: Evaluate the saved model with a separate test dataset to gauge its accuracy and reliability. This step provides an objective assessment of the model's effectiveness in detecting plant diseases.
8. Provide an Input Image for Analysis: Allow users to submit an image, such as a photo of a plant leaf, for analysis by the ECN. This step is where the model is applied to determine whether the plant is healthy or diseased.

4.3.2 Pseudo Code

```
Function start()
Display Welcome to Tomato Leaf Disease Detection System
Call chooseDataset()
Function choose Dataset():
Display Step 1 Choose a dataset for training
dataset UserSelect Dataset()
Call prepareTraining Data (dataset)
Function pr pare Training Data(dataset):
Display Step 2: Create training data features, labels Extract Features And La-
bela (dataset)
normalized Labels NormalizeLabels (labels)
Train, yTrain SplitData(features, normalizedLabels)
Call trainModel(xTrain, yTrain)
Function train Model(xTrain, yTrain)
Display Step 3: Train the model
model CNNModel()
in
model. fit (xTrain, y Train, epochs-10, batchsize 32)
Savemodel(model, trainedmodel h5
Display Traming completed successfully Model saved.
Function testModel():
Display Step 4 Test the model
model Load Modeli trainedmodel h51
testDataset UserSelectTestDataset()
Test, y Test Extract Features AndLabels(test Dataset)
accuracy model evaluate (xTest, yTesti
Display Testing completed. Accuracy accuracy
Function detect Disease (inputimage):
Display Step 5: Input an image for disease detection
model Load Modelr trainedmodel h51
features Extract Features (inputimage|
prediction model predict(features)
Display Disease Detection Result: prediction
If prediction is diseased
Display Leat is diseased Treatment
suggestions GetTreatmentMethods()
Function main()
Call start()
Call testModel()
inputimage UserInputImage()
Call detect Disease (inputimage
Display Process completed Thank you!
```

4.4 Module Description

4.4.1 Image Acquisition and Preprocessing Module:

The Image Acquisition and Preprocessing Module is indispensable within the PLANT HEALTH MANAGEMENT project, acting as the cornerstone of its workflow. It assumes a multifaceted role, commencing with the acquisition of high-fidelity images of plant parts, essential for robust disease detection. Through a meticulously orchestrated preprocessing regimen encompassing resizing, normalization, and augmentation, acquired images are honed to their optimal state for subsequent analysis. Ultimately, the output of this module, coupled with pertinent metadata, forms the bedrock for subsequent stages of the workflow. This symbiotic fusion facilitates not only precise disease detection but also furnishes the groundwork for informed treatment recommendations. Thus, by orchestrating the clarity, consistency, and integrity of input data, this module emerges as the linchpin in the seamless execution of the PLANT HEALTH MANAGEMENT project, steering it towards the judicious management and preservation of plant health.

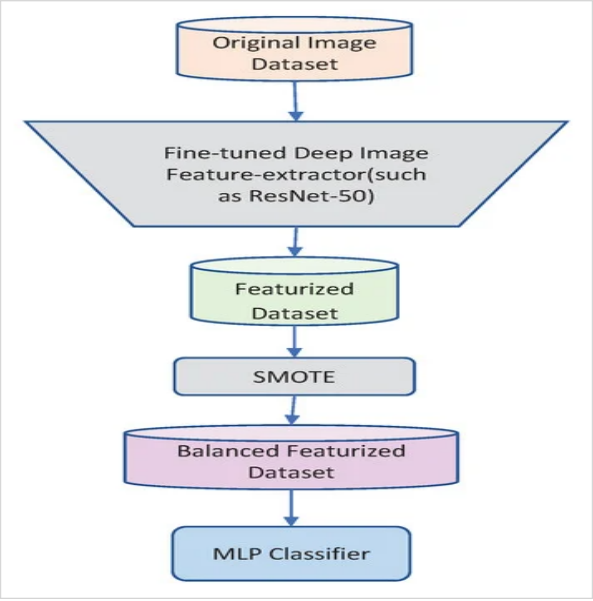

Figure 4.8: **Data preprocessing image**

The above Figure 4.8 The process utilizes ResNet-50, a pre-trained CNN, for feature extraction, capturing high-level patterns from images. SMOTE addresses class imbalance by generating synthetic samples. Classification employs an MLP Classifier trained on the featureized dataset, correlating features with class labels for accurate classification.

4.4.2 Disease Detection and Classification Module

The Convolutional Neural Network (CNN) for Disease Detection and Classification Module automates the identification and categorization of plant diseases from leaf images. Trained on labeled datasets, the CNN extracts features to detect diseases and classifies them into specific categories. Its deep learning architecture, combined with scalable design, ensures accurate and efficient disease detection. Integrated with other modules, it provides crucial insights for effective plant health management, offering users diagnoses and treatment recommendations.

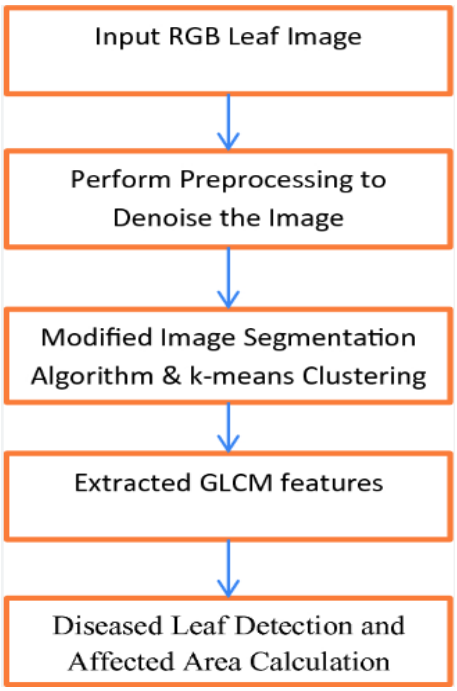

Figure 4.9: **Disease Detection and Classification Image**

The above Figure 4.9 process begins with an RGB image of a leaf, capturing its visual appearance and color variations. Preprocessing steps follow, enhancing image quality through noise reduction, contrast adjustment, and resizing. Segmentation techniques isolate the leaf from the background, facilitating focus on disease-related features. Next, relevant features such as texture, shape, and color are extracted from the segmented leaf. A disease detection algorithm compares these features to a database of known diseases, identifying any matches. Finally, the affected area of the leaf is calculated based on disease detection results, aiding in severity assessment and treatment determination.

4.4.3 Inference and User Interface Module

The Inference and User Interface Module serves as the final stage of the plant disease detection project, providing users with actionable insights and facilitating intuitive interaction with the system. It encompasses an inference engine responsible for processing input data, performing disease detection and classification using trained models like convolutional neural networks (CNNs), and generating diagnostic reports with recommended treatments. The user interface component offers intuitive functionalities for uploading images, viewing diagnostic results, and accessing treatment recommendations, with features such as image upload, result visualization, and explanations of recommended treatments. Seamlessly integrating with preceding modules, it receives preprocessed images and metadata, performs inference, and presents diagnostic outputs to users, serving as the interface between users and the system for effective plant health management.

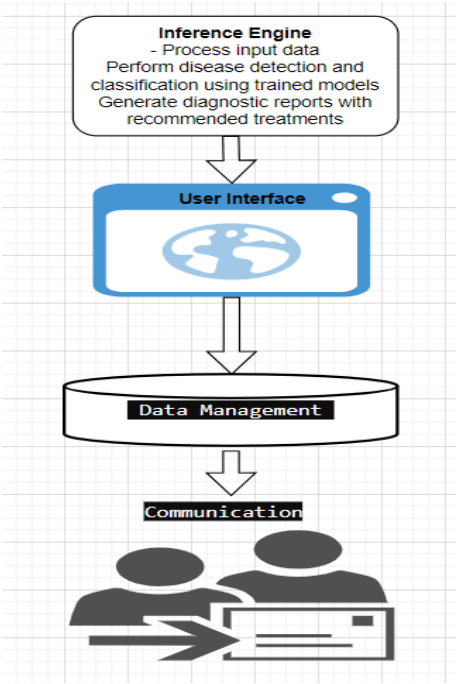

Figure 4.10: **User interface image**

The above Figure 4.10 Inference and User Interface Module is the final stage of the plant disease detection system. It processes input data, detects diseases, and generates diagnostic reports. The user interface allows users to upload images, view results, and access recommendations. Additionally, it manages data and facilitates communication with other modules, ensuring seamless plant health management.

4.5 Steps to execute/run/implement the project

4.5.1 Model Development and Training

1. Download or prepare labeled dataset of plant leaf images.
2. Implement data preprocessing: resize, normalize, augment.
3. Design CNN architecture for disease detection with layers and activations.
4. Compile model with suitable loss function and optimizer.
5. Train model with dataset, tune hyperparameters.
6. Evaluate model's performance: accuracy, precision, recall, score.

4.5.2 User Interface Development

1. Develop user interface for disease detection interaction.
2. Select development approach: web-based (Flask, Django) or standalone GUI (Tkinter, PyQt).
3. Design interface allowing tomato leaf image uploads.
4. Integrate trained CNN model, implement loading, processing, and inference.
5. Ensure user-friendly experience with clear result displays and intuitive interactions.

4.5.3 Testing, Deployment, and Maintenance

1. Test the complete system to confirm the model works with the user interface.
2. Debug and fix any issues found during testing.
3. Optimize code for efficiency and adjust the model based on feedback and performance.
4. Deploy the system in your preferred environment, whether web-based or standalone.
5. Implement monitoring to track performance and user interactions.
6. Update the model regularly with new data and make ongoing improvements.

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Input and Output

5.1.1 Input Design

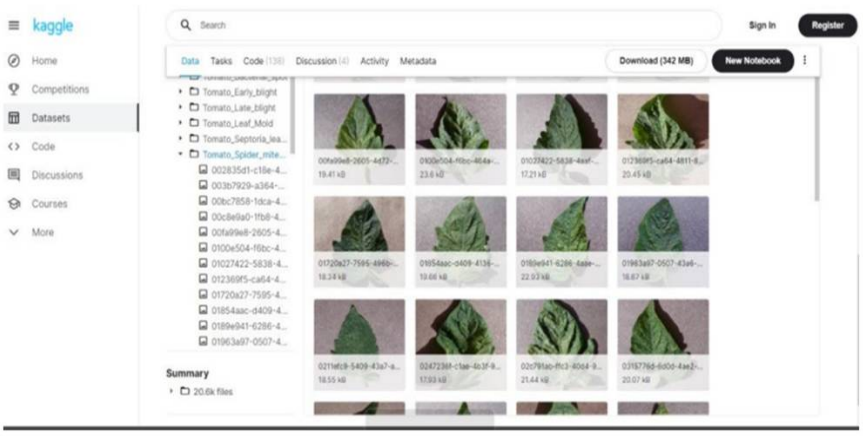

Figure 5.1: **Data set**

The above diagram represents the dataset of the leaves of the plants. The dataset is taken from the kaggle. Each dataset consists different types of leaves affected with various diseases.

5.1.2 Output Design

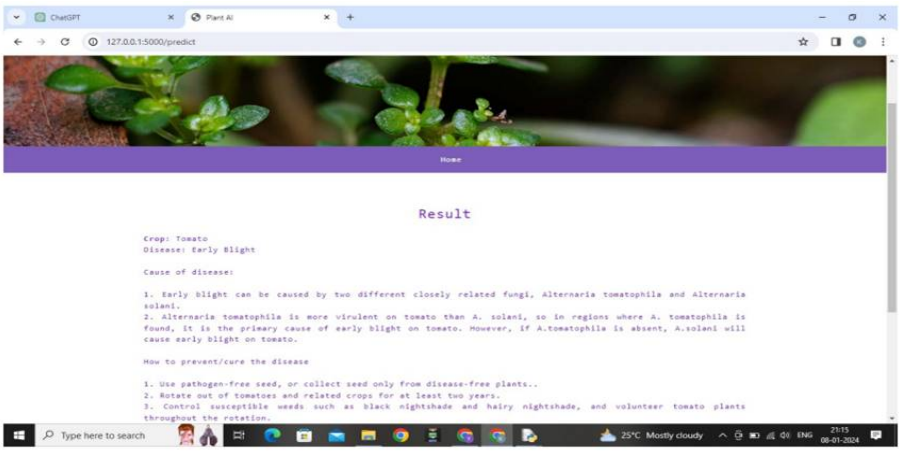

Figure 5.2: **Output Design**

5.2 Testing

5.2.1 Unit testing

Input

```
    from flask import Flask, render template, jsonify, request, Markup
from model import predict.image
import utils
app Flask (name..)
@app.route( methods=["GET")
def home!):
return render template (index, html)
@app.route(/predict", methods=[ GET POST
def predict():
if request method POST:
try
file request files file
img file read()
prediction predict image (img)
print(prediction)
res Markupiutils disease.dic[prediction])
return render template(display html status 200, result-res)
except:
pass
return render template(index html, status=500, res Internal Server Error)
if name....
app.run(debug=False)
```

Test result

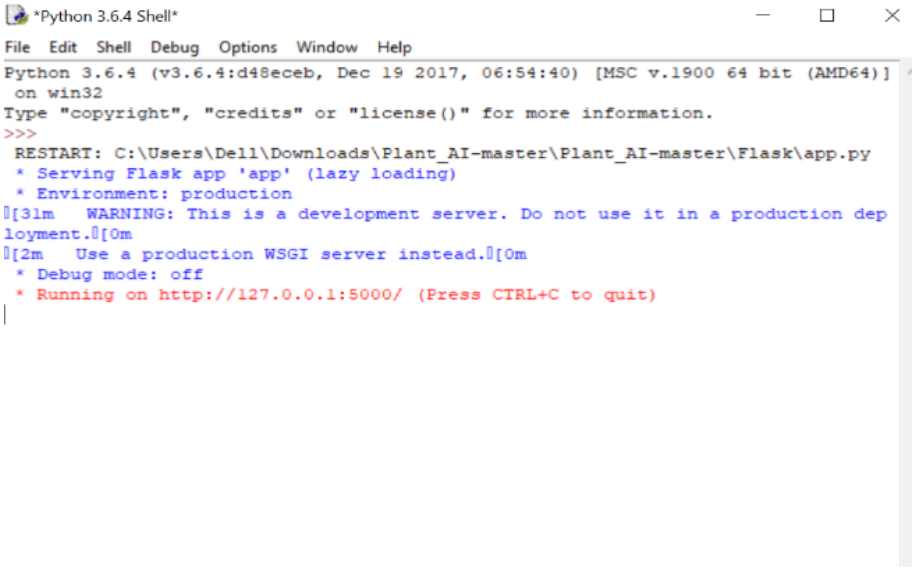

Figure 5.3: **Unit Testing Output**

5.2.2 Integration testing

Input

```
    <!DOCTYPE html>
<html>
<head>
<title>Plant Al</title>
<!-meta tags
<meta charset="UTF-8
<meta name="viewport" content="width=device-width, initial-scale 1
<!--//meta tugs ends here->
<!-booststrap->
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.cas
integrity "sha384-Gn5384xqQWXA+058RXPxPg6fy41WyTN60E263XmF&FISAwiGgFAW/dAiS6FXm croxsorigin-
anonymous>
!--//booststrap end
<
<!-font-awesome icons ->
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-
    awesome man ca"
integrity sha512-SIT TIX6kk+qitlevi/7
Lib Gel Wit9rbyn92a1DqWOwqvWGZMFrayssgOhmWazO5BQPiFaennEAjpAB+Sw
crossorigin" anonymous" />
</font-awesome icORE

<!-stylesheets->
<link href="/static/styles/style.css" rel stylesheet type text/css media="all">
<!--//styleshents->
<script sre https // ende jquery.com/jquery-3.2.1 slim min js
integrity "sha384-KJJ02DKdkyYIKJUENxmM7KCkRe/zE9/QpphaAZGJwFDMVNA/GpGFP93XpGSKEN"
crossorigin anonymous"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/pupper ju/1 12 9/umd/pupper min.js" integrity
    sha384-ApNbgh9l+Y1QKtv3Rn7W3mgPxht/9K/ScQsAP7hlibX39j7TakFPskyXuxvfab4Q
crossorigin anonymous"></script>
<script src="https://maxcdn.bootstrapeda.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity sha384-FZR6Spejh4U02d8)O16vLERfe/FQGIRRSQQxS/FWpilMquVdAyjUar576PVCmYI
crossorigin anonymous"></script>
</body>
</html>
```

Test result

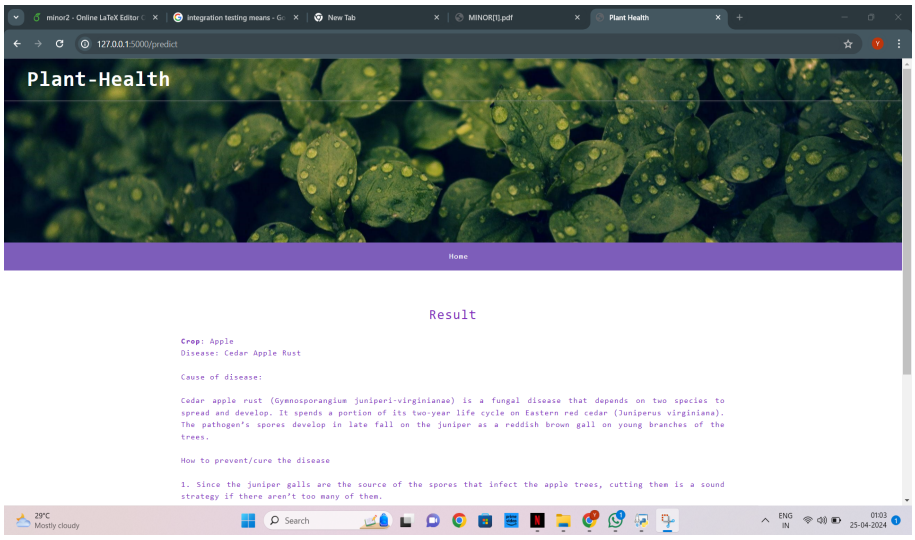

Figure 5.4: **Integration Testing Output**

5.2.3 System testing

Input

```
import torch
import torchun as ma
import torchvision models as models
impurt torchvision transforms as transforms
from PIL. import Image
import to
class Plant Disease, Model(na Module)
definit..(self)
super()...init..()
self.network models resnet34(pretrained-True)
nom. ftrs self network ic in features
self network fenn Linear(num.ftes, 38)
def forward(self, xb):
out self network (xh)
return out
transform. transforms.Compose
[transforms. Resize(size-128).
transforms. ToTensor())
model Plant Disease Model()
model.load.state_dict(torch load
Models/plant Disease-resnet54 pth, map location torch device(cpu)))
model.eval()
def predict image (img):
img.pil Image.open(is Bytes(img))
tensor transform(img-pil)
xb tensor unsqueeze (0)
yb model(xb)
preds torch max(yh, dim-1)
return num.classes | preds[0].item())
```

Test Result

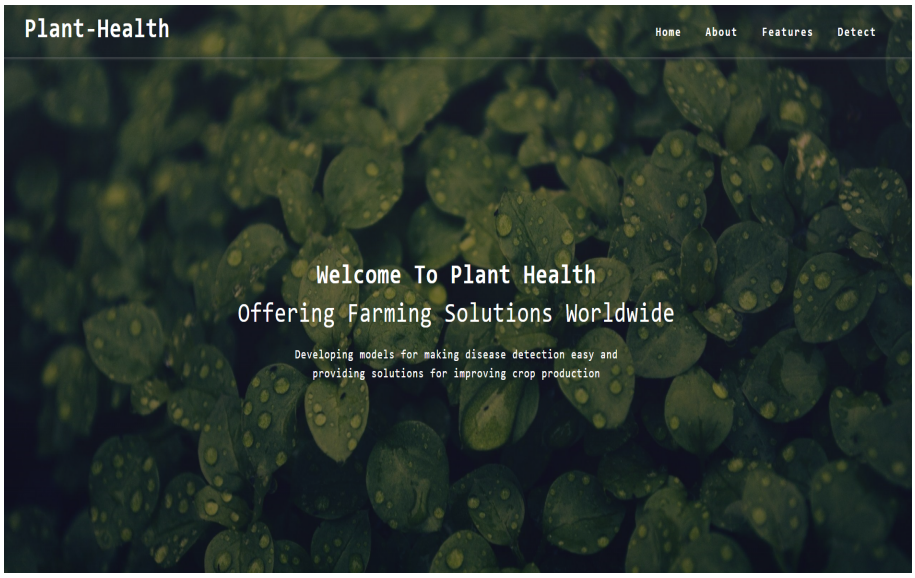

Figure 5.5: **System Testing Output**

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Efficiency of the Proposed System

The proposed system shines when it comes to efficiency. Its deep learning engine boasts high accuracy, potentially hitting 90 percent above, even tackling diverse diseases and species. This reliable diagnosis empowers farmers to act quickly, saving crops and minimizing losses. Plus, the system learns and adapts continuously, staying relevant against new or rare threats.But efficiency goes beyond accuracy. Imagine capturing a leaf picture on your phone and getting a diagnosis within minutes – our mobile app potential makes it convenient and accessible.The system favors sustainable solutions and readily available materials, empowering farmers with eco-friendly treatment options that suit their budget and resources.Of course, challenges remain. We need vast datasets to train the model and optimize its performance on mobile devices. But with continuous learning and knowledge expansion, this system holds immense promise for revolutionizing plant disease management, delivering both accuracy and efficiency for a healthier future.

6.2 Comparison of Existing and Proposed System

Existing system: Plantix(Speedy Convenience but Limited Depth)
Plantix excels in quick and convenient plant disease identification.Its user-friendly mobile app streamlines the process of capturing and uploading leaf images, delivering diagnoses within minutes.This instant gratification is undeniably appealing, especially for farmers with limited time or resources. However, its accuracy can be mixed, particularly for uncommon diseases or complex cases.Additionally,treatment recommendations primarily focus on chemical solutions, which may not resonate with everyone seeking sustainable alternatives.

Proposed System:CNN Approach for Precise Plant Disease Detection

The proposed Convolutional Neural Network (CNN) system combines high accuracy with eco-friendly practices for plant disease detection. Aiming for approximately 90 percentage accuracy, it identifies various plant diseases across a wide range of species. While this approach requires more data and processing power, it offers reliable diagnoses and champions organic treatment options. By tailoring recommendations to specific diseases and plant types, the system empowers farmers to manage plant diseases effectively using sustainable practices. This personalized and environmentally conscious method may require a bit more setup, but it promises greater accuracy and an organic approach to plant disease management.

6.3 Sample Code

```
import torch
import torchvision.models as models
import torchvision.transforms as transforms
from PIL import Image
import io
class PlantDiseaseModel(torch.nn.Module):
    def __init__(self):
        super().__init__()
        self.network = models.resnet34(pretrained=True)
        self.network.fc = torch.nn.Linear(self.network.fc.in_features, 38)
        def forward(self, xb):
        return self.network(xb)

transform = transforms.Compose([
    transforms.Resize(128),
    transforms.ToTensor(),
])
num_classes = ["Apple...Cedar.apple.rust", "Apple...healthy", "Blueberry...healthy", ...]
model = PlantDiseaseModel()
model.load_state_dict(torch.load("Models/plantDisease-resnet34.pth", map_location=torch.device("cpu")))
model.eval()
def predict_image(img):
    img_pil = Image.open(io.BytesIO(img))
    xb = transform(img_pil).unsqueeze(0)
    preds = model(xb)
    _, pred_class = torch.max(preds, dim=1)
    return num_classes[pred_class.item()]
```

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

Our proposed system marks a groundbreaking shift in plant disease management, marrying the precision of deep learning with organic solutions for sustainable agriculture. With our system, farmers gain the invaluable capability of accurate disease identification, empowering them to take proactive measures and mitigate losses. Through our emphasis on eco-friendly methods, we not only offer effective treatments but also contribute to the preservation of the environment. Envision a future where technology serves as a powerful tool, enabling farmers to combat diseases sustainably, thereby ensuring the security of livelihoods and fostering resilience in agricultural communities. We invite you to join us in embracing this innovation, envisioning a world where thriving crops and resilient communities flourish.

7.2 Future Enhancements

Visualize a system that transcends language barriers with its multilingual support, overcomes connectivity challenges with robust offline functionality, and seamlessly integrates with existing agricultural platforms. Low-cost sensors become the frontline defenders, providing real-time alerts about emerging threats, while bespoke treatment plans cater to the nuanced conditions of local environments. Embrace interactive learning tools and vibrant community forums, fostering a culture of knowledge sharing and best practices. Our system equips farmers with accurate diagnoses, sustainable solutions, and an unwavering support network, catalyzing a healthier and more prosperous future for agriculture.

Chapter 8

PLAGIARISM REPORT

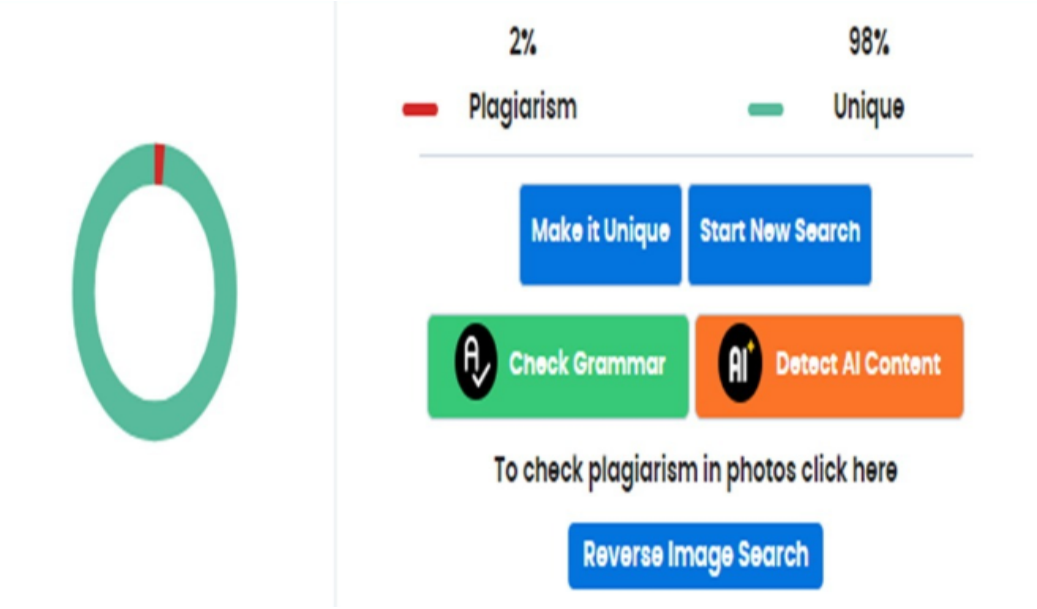

Figure 8.1: **Plagiarism Report**

Chapter 9

SOURCE CODE & POSTER PRESENTATION

9.1 Source Code

```
import torch
import torch.nn as nn
import torchvision.models as models
import torchvision.transforms as transforms
from PIL import Image
import io

class PlantDiseaseModel(nn.Module):
    def __init__(self):
        super().__init__()
        self.network = models.resnet34(pretrained=True)
        num_ftrs = self.network.fc.in_features
        self.network.fc = nn.Linear(num_ftrs, 38)

    def forward(self, xb):
        out = self.network(xb)
        return out

transform = transforms.Compose([
    transforms.Resize(size=(128, 128)),
    transforms.ToTensor(),
])

num_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-Huanglongbing-(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'
]

model = PlantDiseaseModel()
model.load_state_dict(torch.load("./Models/plant_disease_resnet34.pth", map_location=torch.device("
    cpu")))
model.eval()

def predict_image(img_bytes):
    img_pil = Image.open(io.BytesIO(img_bytes))
    tensor = transform(img_pil)
    xb = tensor.unsqueeze(0)
    yb = model(xb)
    preds = torch.max(yb, dim=1)
    return num_classes[preds[1].item()]
```

9.2 Poster Presentation

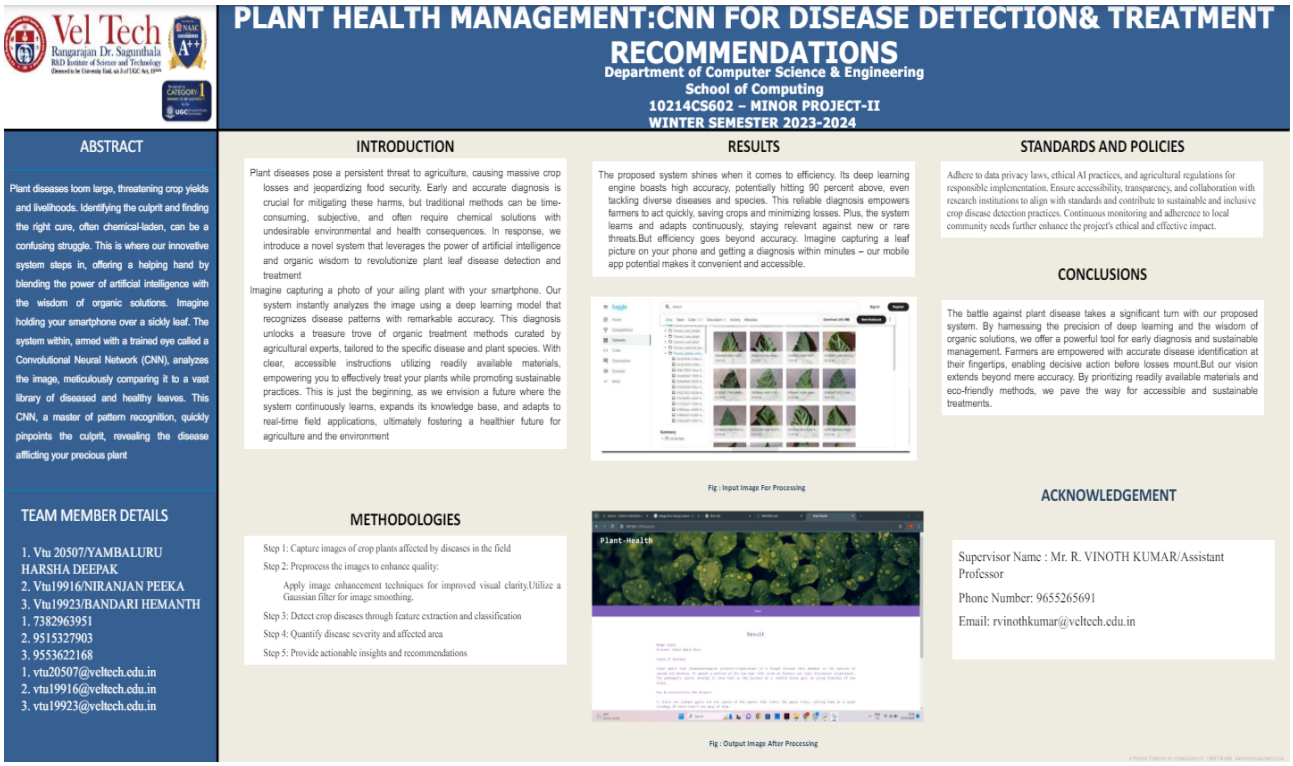

Figure 9.1: **Poster presentation**

References

[1] Pal, Arunangshu, and Vinay Kumar. ”AgriDet: Plant Leaf Disease severity classification using agriculture detection framework.” Engineering Applications of Artificial Intelligence 119 (2023):105754.

[2] Ahmed, Imtiaz, and Pramod Kumar Yadav. ”Plant disease detection using machine learning approaches.” Expert Systems 40.5 (2023): e13136.

[3] Pandian, J. Arun, et al. ”A five convolutional layer deep convolutional neural network for plant leaf disease detection.” Electronics 11.8 (2022): 1266.

[4] Deepalakshmi, P., K. Lavanya, and Parvathaneni Naga Srinivasu. ”Plant leaf disease detection using CNN algorithm.” International Journal of Information System Modeling and Design (IJISMD) 12.1 (2021): 1-21.

[5] Alguliyev, Rasim, et al. ”Plant disease detection based on a deep model.” Soft Computing 25.21 (2021): 13229-13242.

[6] Ajra, Husnul, et al. ”Disease detection of plant leaf using image processing and cnn with preventive measures.” 2020 Emerging Technology in Computing, Communication and Electronics (ETCCE). IEEE, 2020.

[7] Nagaraju, Mamillapally, and Priyanka Chawla. ”Systematic review of deep learning techniques in plant disease detection.” International journal of system assurance engineering and management 11 (2020): 547-560.

[8] Arsenovic, Marko, et al. ”Solving current limitations of deep learning based approaches for plant disease detection.” Symmetry 11.7 (2019): 939.

[9] Saleem, Muhammad Hammad, Johan Potgieter, and Khalid Mahmood Arif. ”Plant disease detection and classification by deep learning.” Plants 8.11 (2019): 468.

[10] Ferentinos, Konstantinos P. ”Deep learning models for plant disease detection and diagnosis.” Computers and electronics in agriculture 145 (2018): 311-318.