

**USMAN INSTITUTE OF TECHNOLOGY**

Affiliated with NED University of Engineering & Technology, Karachi

**Department of Computer Science**

B.S. Computer Science

FINAL YEAR PROJECT REPORT

**Batch-2019**

**Mango Disease Detector**

**By**

| M. Zeeshan Khan | 19B-048-CS |
| --- | --- |
| Javeria Batool | 19B-021-CS |
| Syed Abdul Ali | 19B-049-CS |
| Daniyal Aslam | 19B-056-CS |
| Yumna Rehman | 19B-123-CS |

**Supervised by**

Nasir-ud-Din

*ST-13, Block 7, Gulshan-e-Iqbal, Abul Hasan Isphahani Road, Opposite Safari Park, P.O. Box 75300,*

*Karachi, Pakistan. Phone: 34978274-5; 34994305; 34982476;* [*http://www.uit.edu*](http://www.uit.edu/)

**Submission Proforma**

(1) Muhammad Zeeshan Khan 19B-048-CS

(2) Javeria Batool 19B-021-CS

(3) Syed Abdul Ali 19B-049-CS

(4) Daniyal Aslam 19B-056-CS

(5) Yumna Rehman 19B-123-CS

1. [muhzkhan@students.uit.edu](mailto:muhzkhan@students.uit.edu)
2. [jabatool@students.uit.edu](mailto:jabatool@students.uit.edu)
3. [saali@students.uit.edu](mailto:saali@students.uit.edu)
4. [danaslam@students.uit.edu](mailto:danaslam@students.uit.edu)
5. [yurehman@students.uit.edu](mailto:yurehman@students.uit.edu)

Mango Disease Detector

Nasir ud Din

This report is submitted as required for the Project in accordance with the rules laid down by the Usman Institute of technology as part of the requirements for the award of the degree of Bachelor **Computer** **Science**. I declare that the work presented in this report is my own except where due reference or acknowledgement is given to the work of others.

Signatures of students Date

(1) …………………………….. ……………………….

(2) ……………………………. ……………………….

(3) …………………………….. ……………………….

(4) …………………………….. ……………………….

(5) …………………………….. ……………………….

**Acknowledgments**

We would like to express our deep gratitude and sincere appreciation to all those who have played an indispensable role in turning this project into a reality. The accomplishment we celebrate today is a true testament to the combined efforts, unwavering commitment, and valuable contributions of a diverse group of individuals. Our journey towards successfully completing this project has been a collaborative and enriching experience that we humbly acknowledge.

First and foremost, we extend our heartfelt thanks to the Divine Providence for bestowing us with blessings and guidance that have illuminated our path throughout this endeavor. It is through these blessings that we have been able to surmount challenges and attain our goals.

A special mention is reserved for our esteemed supervisor, Sir Nasir ud Din. His guidance, mentorship, and unwavering support have been pivotal in shaping the course of this project. His profound knowledge, constructive feedback, and insightful suggestions have consistently motivated us to exceed expectations and gain a deeper understanding of the project's scope and significance. Sir Nasir ud Din's steadfast belief in our capabilities and his dedication to our growth have been truly invaluable.

Additionally, we extend our heartfelt appreciation to UIT University for providing us with the platform to conceptualize and develop this project. The university's dedication to fostering innovation and academic excellence has empowered us to apply our theoretical knowledge to real-world challenges. We are grateful for the encouragement and resources provided by the university, which have facilitated our advancement and learning.

We would also like to acknowledge the supportive coordination of Sir Zeeshan Saleem Khan and Sir Syed Faisal Ali. Their assistance and guidance have been instrumental in our project's journey, and their encouragement has bolstered our spirits.

In conclusion, we remain profoundly grateful to all those who have contributed to this project's success, and we recognize that our achievements are a result of collective determination and shared vision.

**Abstract**

The agriculture sector is a vital component of many economies worldwide, and Pakistan is no exception. With a substantial portion of its economy relying on agriculture, the country faces various challenges in maintaining agricultural productivity and addressing issues such as plant malformation. Mangoes are one of Pakistan's key agricultural exports, but the presence of mango tree and farm malformation has led to reduced product yield and compromised quality, resulting in economic crises for farmers and the nation as a whole. To tackle this pressing issue, the Mango Disease Detector project has been launched with the main objective of developing a mobile and web application capable of detecting mango diseases and providing recommendations for curing affected plants or farms. Leveraging cutting-edge technologies like Convolutional Neural Networks (CNNs), this project aims to reduce human effort in monitoring mango farms while offering valuable insights to improve overall farm management. The core functionality of the Mango Disease Detector centers around a digital image processing and classification system. When a user encounters a mango plant that may be affected by disease, they can easily capture an image of the plant using their mobile phone and upload it to the application. Behind the scenes, the uploaded image undergoes various preprocessing techniques to enhance its quality and prepare it for disease analysis.

The next step involves labeling the image to indicate whether the mango plant is diseased or healthy. The system extracts features, such as color information and pixel characteristics, which serve as inputs to the CNN models responsible for classifying the image and detecting the presence of diseases. CNNs are particularly adept at image recognition tasks, making them ideal for detecting diseases based on visual cues. The CNN models used in the Mango Disease Detector have been trained on extensive datasets containing labeled images of healthy and diseased mango plants, ensuring accurate differentiation between the two, even when presented with new and unseen images. Upon completion of the analysis, the Mango Disease Detector provides a comprehensive report to the user. If the plant is found to be affected by a disease, the report specifies the type of mango disease detected and offers tailored recommendations and methods for curing the specific disease. Implementing the Mango Disease Detector offers numerous benefits for farmers and the agriculture sector in Pakistan. Early disease detection allows farmers to take timely actions to prevent further spread and minimize losses. The application's ability to suggest appropriate remedies facilitates effective disease management and ensures healthier mango fruits, thereby improving agricultural productivity and product quality. Moreover, the Mango Disease Detector's utilization of mobile and web technologies enhances accessibility for farmers across different regions of the country. As smartphones become increasingly prevalent, even in rural areas, the application becomes a powerful tool for reaching and assisting a wide user base.

**Table of contents**

Contents

[**Declaration 2**](#_meukdy)

[**Acknowledgments 3**](#_gjdgxs)

[**Abstract 4**](#_2s8eyo1)

[**Table of contents 5**](#_17dp8vu)

[**List of tables 8**](#_3rdcrjn)

[**List of figures 9**](#_36ei31r)

[**List of symbols and Units 10**](#_1ljsd9k)

[**1.**](#_26in1rg) **Introduction 11**

[1.1](#_lnxbz9) Project Overview 13

[1.2](#_35nkun2) Objective 13

[*1.2.1*](#_1ksv4uv) *Key Features of the Project 13*

[1.3](#_44sinio) System Diagram 14

[1.4](#_2jxsxqh) Algorithm & Techniques 15

[1.5](#_z337ya) Goal 17

[*1.5.1*](#_3j2qqm3) *Primary Goals to Achieve 17*

[1.6](#_1y810tw) Project Impact 18

[1.7](#_4i7ojhp) Species of Mangoes in Pakistan 18

[1.8](#_2xcytpi) Name of Diseases and their Symptoms 19

[1.9](#_1ci93xb) Chapter Conclusion 19

[**2.**](#_3whwml4) **Background and Literature Review 21**

[2.1](#_2bn6wsx) Introduction 21

[2.2](#_qsh70q) Disease Classification 22

[*2.2.1*](#_3as4poj) *Image Acquisition 22*

[*2.2.2*](#_1pxezwc) *Image Processing 22*

[*2.2.3*](#_49x2ik5) *Feature Extraction 23*

[*2.2.4*](#_2p2csry) *Convolutional Neural Network 23*

[2.3](#_3o7alnk) Similar Application 24

[*2.3.1*](#_23ckvvd) *Plant Disease & Treatment 24*

[*2.3.2*](#_ihv636) *Plant Disease Identification 25*

[*2.3.3*](#_32hioqz) *Angrio 26*

[*2.3.4*](#_1hmsyys) *LeafSnap Plants Identification 27*

[2.4](#_41mghml) Comparison of Similar Products 28

[2.5](#_2grqrue) Chapter Conclusion 29

[**3.**](#_vx1227) **Aim and Statement of Problem 30**

[3.1](#_3fwokq0) Problem Statement 30

[3.2](#_1v1yuxt) AIM 31

[3.3](#_4f1mdlm) Project Scope 32

[*3.3.1*](#_2u6wntf) *In Scope 32*

[*3.3.2*](#_19c6y18) *Out of Scope 33*

[3.4](#_3tbugp1) Chapter Conclusion 33

[**4.**](#_28h4qwu) **Hardware, Software analysis and requirements 34**

[4.1](#_nmf14n) Fact Finding Technique 34

[*4.1.1*](#_37m2jsg) *Research 34*

[*4.1.2*](#_1mrcu09) *Prototyping 35*

[4.2](#_46r0co2) Hardware Requirements 36

[4.3](#_2lwamvv) Software Requirements 37

[*4.3.1*](#_111kx3o) *For Training of Model 37*

[*4.3.2*](#_3l18frh) *For Web Application 37*

[*4.3.3*](#_206ipza) *For Mobile Application 38*

[4.4](#_4k668n3) Chapter Conclusion 38

[**5.**](#_2zbgiuw) **Software design and modeling 40**

[5.1](#_1egqt2p) Sequence Diagram 40

[*5.1.1*](#_3ygebqi) *Description 40*

[5.2](#_2dlolyb) State Diagram 42

[*5.2.1*](#_sqyw64) *Description 42*

[5.3](#_3cqmetx) Activity Diagram 44

[*5.3.1*](#_1rvwp1q) *Description 44*

[5.4](#_4bvk7pj) Communication Diagram 46

[*5.4.1*](#_2r0uhxc) *Description 46*

[5.5](#_1664s55) Component Diagram 48

[*5.5.1*](#_3q5sasy) *Description 48*

[5.6](#_25b2l0r) Class Diagram 50

[*5.6.1*](#_kgcv8k) *Description 50*

[5.7](#_34g0dwd) Use Case 52

[*5.7.1*](#_1jlao46) *User 52*

[*5.7.2*](#_43ky6rz) *System 54*

[*5.7.3*](#_2iq8gzs) *View Disease 56*

[5.8](#_xvir7l) Deployment Diagram 58

[*5.8.1*](#_3hv69ve) *Description 58*

[5.9](#_1x0gk37) `Operational Diagram 61

[*5.9.1*](#_4h042r0) *Description 61*

[**6.**](#_2w5ecyt) **Algorithm analysis and complexity 63**

[6.1](#_1baon6m) CNN Algorithm 63

[*6.1.1*](#_3vac5uf) *Pseudocode 64*

[6.2](#_2afmg28) Cases 66

[*6.2.1*](#_pkwqa1) *Best Case 66*

[*6.2.2*](#_39kk8xu) *Average Case 66*

[*6.2.3*](#_1opuj5n) *Worst Case 67*

[6.3](#_48pi1tg) Chapter Conclusion 67

[**7.**](#_2nusc19) **Implementation 69**

[7.1](#_1302m92) CNN Model 69

[**8.**](#_3mzq4wv) **Testing 76**

[8.1](#_2250f4o) Black Box Testing 76

[8.2](#_haapch) White Box Testing 77

[*8.2.1*](#_319y80a) *Unit Testing 78*

[8.3](#_1gf8i83) Gray Box Testing 79

[8.4](#_40ew0vw) Black Box Testing on Mango Disease Detector 79

[8.5](#_2fk6b3p) Unit Testing on Mango Disease Detector 82

[8.6](#_upglbi) Chapter Conclusion 85

[**9.**](#_45jfvxd) **Discussion (optional) 87**

[**10.**](#_2koq656) **Conclusions 89**

[**11.**](#_zu0gcz) **Future work 90**

[11.1](#_3ep43zb) APPENDICES A 90

[11.2](#_1tuee74) APPENDICES B: 91

[*11.2.1*](#_4du1wux) *UML DIAGRAMS 92*

[11.3](#_2szc72q) APPENDICE C 96

[11.4](#_184mhaj) APPENDICES D 98

[*11.4.1*](#_3s49zyc) *Plagiarism Report 99*

[**12.**](#_279ka65) **References 100**

# List of tables

Name of Disease & their symptoms…………………………………18

**List of figures**

1.1 System Diagram………………………………………………….13

2.5 Flow of Project…………………………………………………...28

5.1 Sequence Diagram………………………………………………. 39

5.2 State Diagram…………………………………………………….41

5.3 Activity Diagram…………………………………………………43

5.4 Communication Diagram……………………………...…………45

5.5 Component Diagram…………………………………...………...47

5.6 Class Diagram……………………………………………………49

5.7 User Case Diagram of User……………………………………...51

5.8 User Case Diagram of System…………………………………...53

5.9 User Case Diagram of View Disease…………………………….55

5.10 Deployment Diagram…………………………………………...57

5.11 Operational Diagram……………………………………………60

# Introduction

Agriculture has been the backbone of human civilization since time immemorial, providing sustenance and livelihood to communities around the world. In many countries, including Pakistan, the agriculture sector plays a vital role in the economic establishment, contributing significantly to the GDP, employment, and food security. Among the diverse range of agricultural products, mangoes hold a special place as one of Pakistan's major exports and a symbol of the country's rich agricultural heritage. However, the prevalence of diseases affecting mango trees has become a concerning challenge for farmers, leading to substantial economic losses and impacting the overall economy [9].

Mango cultivation in Pakistan holds immense cultural, social, and economic importance.The country's unique geographical location and favorable climate conditions create an ideal environment for growing high-quality mangoes . As a result, Pakistan is known for producing a wide variety of mango species, each with its distinctive characteristics . The success of the mango industry has not only fueled the local economy but also earned the country recognition as a significant exporter of this tropical fruit .

Despite the economic benefits, mango cultivation faces various threats, including pests, climate change, and most significantly, diseases. Mango trees are susceptible to several diseases caused by pathogens, such as fungi, bacteria, and viruses [9]. These diseases manifest in different forms, affecting various parts of the mango plant, including leaves, flowers, and fruits .Common mango diseases include anthracnose, die black, gall midge, powdery mildew, and sooty mold among others [9]. Due to these diseases, the quality and yield of mangoes are severely compromised, leading to devastating consequences for farmers and the national economy.

Traditionally, disease detection and management in agriculture have relied on human observation and expert knowledge. Farmers visually inspect their crops, looking for telltale signs of diseases, and based on their experience, take necessary actions to control the spread. However, this approach has limitations, as human inspection may not always be accurate, leading to delayed responses and ineffective disease management. Moreover, with the expansion of mango orchards and increasing demand for mangoes, manual inspection becomes time-consuming and impractical [9].

To address these challenges and improve disease detection accuracy, modern technologies, particularly computer vision and artificial intelligence, offer innovative solutions. Computer vision focuses on enabling machines to interpret and understand visual information from the world. By applying computer vision techniques, we can process digital images of mango trees, extract relevant features, and make automated disease diagnoses. This amalgamation of agriculture and technology has opened new avenues for sustainable agriculture, smart farming, and precision agriculture.

The Mango Disease Detector project aims to leverage the potential of computer vision and digital image processing to develop a web and mobile application that can accurately detect diseases in mango trees. By providing early and reliable disease detection, this application seeks to assist farmers in safeguarding their mango orchards and reducing economic losses [9]. Timely detection allows farmers to take proactive measures to control the spread of diseases, apply appropriate treatments, and ensure healthy mango harvests.

The central focus of this project is on mango diseases prevalent in Pakistan. Each region of the country has unique environmental conditions, which can influence disease occurrence and severity. By concentrating on local mango species and their specific diseases, the Mango Disease Detector is tailored to address the challenges faced by Pakistani farmers [9]. Additionally, this project aims to create a user-friendly interface, accessible to farmers across different regions and technological backgrounds.

The core technology powering the Mango Disease Detector is the Convolutional Neural Network (CNN). CNNs are a class of deep learning models specially designed for image recognition tasks. They can automatically learn and extract intricate patterns and features from images, making them well-suited for disease detection based on visual cues. By training the CNN model on a diverse dataset of mango leaf images, comprising both healthy and diseased samples, the system can learn to distinguish between the different disease patterns and accurately identify diseases [10].

In this project, the collection and curation of the dataset play a crucial role in achieving the desired accuracy. The dataset must encompass a wide variety of mango diseases, representing their varying severity and manifestations [10]. Furthermore, it should include images of mango leaves taken under different lighting conditions and angles to ensure the model's robustness.

The development of the Mango Disease Detector involves several stages, including data collection, preprocessing, model development, and application design.

In the preprocessing stage, the images undergo various transformations to standardize their dimensions, color contrast, and remove any artifacts that might affect the model's performance. Preprocessing ensures that the input data fed to the CNN model is consistent and suitable for training.

Designing an efficient and accurate model involves selecting appropriate architecture, hyperparameter tuning, and optimization.The CNN model undergoes an extensive training process, where it learns to recognize distinct disease patterns and classify mango leaves accordingly.

Once the CNN model is trained, it is integrated into the web and mobile application. The user-friendly interface allows farmers to capture images of mango leaves using their smartphones or upload images from their computers . The uploaded images are then processed by the trained CNN model, and the application generates a comprehensive report indicating the presence of diseases and their severity [9]. Alongside the diagnosis, the application also provides tailored recommendations and treatment options to address the specific disease detected.

The Mango Disease Detector project represents a pioneering initiative to bridge the gap between agriculture and technology. By utilizing the power of computer vision and artificial intelligence, this project aims to empower Pakistani farmers with a powerful tool to combat mango diseases effectively [9]. With the potential to revolutionize mango cultivation and enhance agricultural productivity, the Mango Disease Detector holds promise for a more sustainable and prosperous future for Pakistan's agricultural sector.

## Project Overview

Pakistan, known for its fertile lands and diverse agricultural products, heavily relies on the agriculture sector, which contributes to approximately 70% of the country's economy [14]. However, the sector faces numerous challenges, including the prevalence of diseases in fruits and vegetables that adversely affect their quality and productivity [15]. These diseases have significant implications for the national economy, leading to decreased yields and substantial financial losses. Notably, mangoes, a vital fruit crop for Pakistan, experienced a significant decline in productivity in 2019, resulting in the country incurring millions of dollars in losses due to the impact of diseases on mango trees.

The "Mango Disease Detector" project aims to address this critical issue by focusing on the agriculture sector and specifically targeting the detection of diseases on mango trees. By utilizing advanced technology, the project seeks to detect diseases based on the analysis of leaf images, providing farmers with crucial information regarding the presence and severity of diseases in their mango orchards [14]. This project is intended to aid farmers in making informed decisions and implementing appropriate measures to mitigate the impact of diseases, thereby boosting mango productivity and supporting the country's agricultural economy.

## Objective

The objective of this project is to build a simple and easy to use application that will make predictions of leaves that if they are affected or non-affected and to provide a summarized report of that, and give some suggestions based on the predictions.

### Key Features of the Project

#### Image Upload / Capture

Users can capture the image or upload it from their devices.

#### Image Pre-Processing

The uploaded or captured images undergo pre-processing techniques so that the quality of the image can be enhanced and can be easily processed by the model.

#### Image Classification

The pre-processed images are then used by the Convolutional Neural Network (CNN)

model. The model trained on the labeled data set of mango leaves.

#### Disease Severity Assessment

The system calculates the percentage of disease presence on the leaf based on the classified labels, providing an indication of the severity level.

#### Report Generation

A comprehensive report is generated, summarizing the results of the analysis.

#### User Interface

The application provides a user-friendly interface allowing users to navigate through different functionalities. The user interacts with the system through mobile application or web application.

#### Scalability

The system is designed to handle multiple captured or uploaded images concurrently, it provides high performance and scalability.

## System Diagram

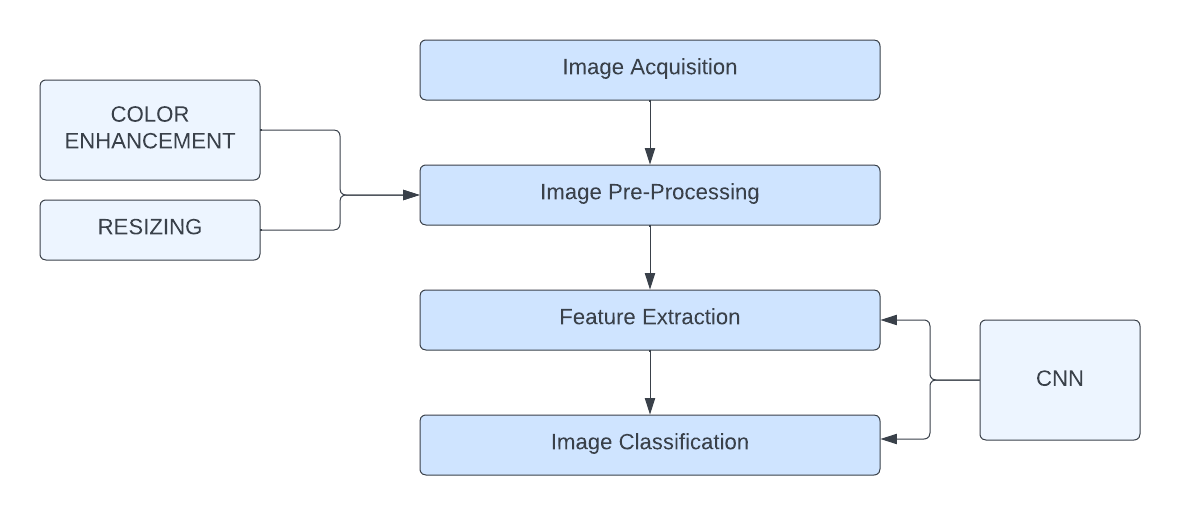


 Figure 1.1 System Diagram

In Figure 1.1, the first step is image acquisition in which an image is retrieved from a digital camera or being uploaded by the user or farmers, and then the second step is the preprocessing of the images. Preprocessing is essential to enhance the quality of the images and prepare them for further analysis. During this stage, several image processing techniques are applied to achieve clearer and more refined images. One of the main tasks in preprocessing is removing noise from the images. Noise in images can result from various factors, including lighting conditions, image acquisition devices, and environmental factors. By eliminating noise, the Mango Disease Detector ensures that the subsequent analysis is based on accurate and reliable information. Another vital aspect of preprocessing is color balancing and enhancement, the Mango Disease Detector can enhance the color composition of the images, bringing out important features and details that may otherwise remain obscured. This process aims to create standardized and consistent color representations across the dataset, ensuring that the disease detection model is not influenced by variations in image coloration. With the images now preprocessed and optimized, the Mango Disease Detector proceeds to the next crucial stage: feature extraction. Feature extraction is a fundamental process in image analysis, as it helps reduce the complexity of the dataset while preserving the essential information needed for disease detection. In this step, the initial set of raw image data is transformed into a more manageable and compact set of meaningful features. These features act as distinguishing characteristics that allow the disease detection model to recognize patterns and make accurate classifications. In the Mango Disease Detector project, feature extraction is particularly critical for achieving effective disease detection using the CNN algorithm. CNNs are a class of deep learning models inspired by the visual processing mechanisms of the human brain. They excel in learning intricate patterns and hierarchical representations from images, making them ideal for image classification tasks. However, the success of CNNs heavily relies on the quality of the features fed into the model.

The feature extraction process in the Mango Disease Detector project is driven by the CNN algorithm's ability to automatically learn and discern relevant features from the preprocessed images. The CNN model is trained on the preprocessed dataset, with each image labeled based on its disease status. During training, the model learns to detect patterns and features indicative of healthy mango leaves and leaves affected by various diseases. One of the key strengths of CNNs is their ability to recognize patterns irrespective of their orientation, position, or scale within an image. This property allows the Mango Disease Detector to be robust and reliable in detecting diseases, even when faced with images taken from different angles or under varying conditions.

Upon successful training, the Mango Disease Detector's CNN model is ready to classify new, unseen images. When a user captures an image of a mango leaf and uploads it to the application, the image undergoes the same preprocessing steps applied during the training phase. The preprocessed image is then fed into the trained CNN model, which analyzes the image's features and makes a disease classification.

In conclusion, the Mango Disease Detector project is a multifaceted endeavor that leverages cutting-edge technologies to automate and enhance disease detection in mango trees. By utilizing the power of the CNN algorithm, the application achieves accurate and efficient classification of mango leaf images, aiding farmers in preserving the health and productivity of their mango orchards. As the project progresses, continuous improvements and advancements in technology will further strengthen the Mango Disease Detector's capabilities, solidifying its role as a valuable tool in promoting sustainable agriculture and supporting the economy of Pakistan.

## Algorithm & Techniques

In the Mango Disease Detector project, Digital Image Processing (DIP) techniques play a pivotal role in the classification of diseases affecting mango trees [8]. Among the various algorithms utilized, the Convolutional Neural Network (CNN) stands out as a powerful deep learning model . CNNs have revolutionized image analysis and classification tasks, making them an ideal choice for accurately detecting diseases in mango leaf images .

Convolutional Neural Networks are a class of deep learning models inspired by the visual processing mechanisms of the human brain . They are particularly effective in tasks involving image recognition, object detection, and classification . The CNN algorithm is designed to automatically learn and extract intricate patterns and features from images, enabling it to make high-level decisions based on the visual information present in the input data [8].

One of the key components of CNNs is the convolutional layer, which performs the process of convolution . Convolution involves passing a set of learnable filters (also known as kernels) over the input image to detect various patterns and features [8]. The convolutional layer extracts local features from different regions of the input image, capturing essential characteristics that help the model discern between different classes of images.

By using multiple convolutional layers, CNNs are capable of learning hierarchical representations of the input data. These layers process the input image at different scales and levels of abstraction, creating a deeper understanding of the visual features present in the image. The CNN architecture allows it to recognize complex and non-linear patterns, making it highly suitable for tasks requiring sophisticated feature extraction, such as image classification [8].

In the Mango Disease Detector project, the CNN algorithm is applied to classify mango leaf images into different categories, such as healthy leaves or leaves affected by specific diseases [9]. During the training phase, a large dataset of mango leaf images is used to teach the CNN model to recognize disease-related patterns and features . Each image in the dataset is labeled based on its disease status, allowing the model to learn the correspondence between visual patterns and disease presence [9].

As the CNN model goes through the training process, it optimizes its internal parameters (weights and biases) to minimize the classification errors on the training data. This optimization process, known as backpropagation, fine-tunes the model's parameters to enhance its ability to recognize and classify patterns associated with diseases [8].

The power of CNNs lies in their ability to detect patterns and features irrespective of their position, orientation, or scale within the image[8]. This property ensures that the Mango Disease Detector remains robust and reliable in detecting diseases, even when presented with images taken from different angles or under varying lighting conditions .

Once the CNN model is trained, it is ready for real-world disease detection tasks. When a user captures an image of a mango leaf and uploads it to the Mango Disease Detector application, the image undergoes preprocessing to standardize its dimensions and eliminate noise, ensuring that the input data is consistent with the training data .

The preprocessed image is then fed into the trained CNN model for analysis. The model passes the image through its layers, performing convolutions and extracting relevant features [8]. Based on the learned patterns and features, the CNN algorithm makes a classification decision, identifying whether the uploaded image shows signs of disease and, if so, predicting the type and severity of the disease .

The Mango Disease Detector generates a comprehensive report for the user, providing crucial information about the health of their mango trees. This report empowers farmers to take proactive measures to manage diseases, prevent their further spread, and safeguard the productivity and quality of their mango orchards.

The utilization of the Convolutional Neural Network algorithm in the Mango Disease Detector project exemplifies the integration of cutting-edge deep learning techniques in agriculture. By leveraging the power of CNNs for image classification, the application achieves accurate and efficient disease detection, providing farmers with valuable insights to improve disease management strategies.

## Goal

Our goal is to develop an application that will become a helping hand to the farmers of Mangos. Our Project will help farmers to cure their orchids at an early stage. Our application is simple and accurate. Applications start from getting images from the user, that will be stored on the database. Then the AI/ML model will take pictures and start processing. The predictions will be stored on the Database. Based on those predictions a summarized report will be generated. And some suggestions will be provided to cure those diseases. Our application is a web application, so it will be deployed to a web server, and it will be accessible from everywhere and anyone can access it.

### Primary Goals to Achieve

* Enable the accurate detection of diseases in mango trees through the analysis of leaf images. The system should be capable of identifying common diseases that affect mango trees.
* Timely Intervention: Facilitate early detection of diseases to enable prompt intervention and appropriate treatment. Timely detection can help prevent the spread of diseases and minimize crop damage.
* Severity Assessment: Provide an assessment of the severity or extent of the diseases detected in mango trees. This information helps farmers prioritize management strategies and allocate resources accordingly.
* Recommendations and Precautions: Offer actionable recommendations and precautions to farmers based on the detected diseases and their severity. These recommendations can include appropriate treatments, cultural practices, and preventive measures to control the spread of diseases and maintain the health of the mango crop.

## Project Impact

The Mango Disease Detector project holds the potential to create a significant positive impact on Pakistan's agriculture sector and the national economy. By providing timely and accurate disease detection, the project will empower farmers to make informed decisions, implement appropriate disease management strategies, and safeguard their mango orchards. As a result, the project aims to increase mango productivity, improve product quality, and ultimately contribute to enhanced export revenues for the country.

## Species of Mangoes in Pakistan

* Almaas
* Alphonso
* Anmol
* Anwar Rataul
* BaganPali
* Chaunsa
* Chok Anan
* Collector
* Dusehri
* Desi Ada Pamato
* Desi Badam
* Desi Gola
* Desi Badshah
* Dilkash
* Fajri
* Gulab Janhu
* Gulab Khas,
* Lahoti,
* Lal Badshah,
* Langra
* Malda
* Muhammad Wole
* Nawab Puri
* Neelum
* Rani Phool
* Sindhri
* Saroli
* Sawarnarika
* Saleh Bhai
* Saib
* Shan-e-Khuda
* Taimuria
* Toofan
* Wanghi
* Zafran

## Name of Diseases and their Symptoms

| **Name of the Disease** | **Symptoms** | **Disease** |
| --- | --- | --- |
|  |  |  |
| Anthracnose | Anthracnose is a fungal disease that affects mango trees, causing dark lesions on leaves, flowers, and fruits, leading to premature fruit drop and reduced yield. |  |
|  |  |  |
| Sooty Mould | Growth of dark, soot-like patches on the leaves and fruits, often caused by insect infestations like aphids or whiteflies. |  |
|  |  |  |
| Die Black | Fungal infection characterized by black streaks on the trunk and branches, leading to wilting and death of the tree. |  |
|  |  |  |
| Powdery Mildew | A powdery white growth on leaves. |  |
|  |  |  |
| Gall Midge | Presence of wart like galls |  |

**Table01 Name of Disease & their symptoms**

## Chapter Conclusion

The Mango Disease Detector web application represents a significant leap forward in the field of agriculture, leveraging the capabilities of Artificial Intelligence and Machine Learning to address crucial challenges faced by mango farmers **[9]**. By providing an intuitive and user-friendly platform, the application empowers farmers to detect diseases in their mango trees swiftly and accurately. The ability to identify diseases at early stages can be a game-changer for mango cultivation, as it allows farmers to implement timely interventions and prevent further spread of infections. The AI/ML model, trained on a diverse dataset of mango leaf images, ensures reliable disease classification, guiding farmers towards informed decisions for effective disease management. The inclusion of tailored pesticide recommendations enhances the application's practicality, as it equips farmers with actionable solutions to combat specific diseases. As a result, the Mango Disease Detector can significantly reduce economic losses caused by disease outbreaks and promote sustainable agriculture practices.

Beyond its immediate impact on disease detection, the Mango Disease Detector holds great potential for the entire agriculture sector. As AI and ML technologies continue to advance, the application can be extended to cover other crops and plant diseases, supporting a broader range of farmers and ensuring food security on a larger scale [15]. Moreover, the wealth of data gathered through the application can contribute to valuable insights and research on disease patterns, allowing for proactive measures to prevent future outbreaks. Collaborating with agricultural experts, government agencies, and research institutions, the Mango Disease Detector can become a valuable tool in optimizing disease management strategies and promoting resilient farming practices.

The success of the Mango Disease Detector project is founded on collaborative efforts between technologists, researchers, and farmers. Involving farmers in the design and development process has been instrumental in creating a user-centric solution that aligns with their specific needs and challenges. Continuous engagement with the agricultural community will be vital in refining the application and adapting it to the evolving requirements of mango farming.

In a broader context, the Mango Disease Detector exemplifies the potential of technology to drive innovation in traditional industries. By embracing AI/ML solutions, agriculture can leap into the era of smart farming, where data-driven decisions, precision agriculture, and sustainable practices converge to optimize resource usage and increase productivity. The project demonstrates that technology can be harnessed not only for scientific advancements but also for practical solutions that impact people's lives directly.

The Mango Disease Detector web application epitomizes the fusion of technology and agriculture to tackle real-world challenges. As it moves from a pilot project to wider implementation, it has the potential to become a transformative force in mango cultivation, promoting resilience, sustainability, and prosperity for farmers across the region. The journey doesn't end here; rather, it opens up new possibilities for the intersection of AI, ML, and agriculture to revolutionize farming practices and contribute to global food security. With continued support and collaboration, the Mango Disease Detector is poised to become a beacon of innovation in the agricultural landscape, offering hope for a sustainable future for mango farmers and the broader agricultural community.

# Background and Literature Review

## Introduction

This literature review represents a comprehensive exploration of Digital Image Processing (DIP) techniques and their pivotal role in disease prevention within mango cultivation. The primary focus of this review is to discern the diverse range of DIP methodologies employed in safeguarding mango trees against diseases. The evaluation of these techniques encompasses a system analysis encompassing image acquisition, image pre-processing, feature extraction, and the subsequent classification of diseases, with each step playing a crucial role in disease detection [1].

In the broader context of agriculture, DIP techniques have emerged as indispensable tools for early disease detection and prevention across various crops and plants [1]. However, this literature review specifically hones in on the application of DIP within the realm of mango trees, acknowledging their unique challenges and requirements [2]. The overarching objective is to meticulously identify, analyze, and elucidate the various DIP methods deployed for the detection and management of diseases in mango cultivation [3]**.**

In the pursuit of sustainable mango orchard management, it becomes increasingly evident that technology integration is essential, given the challenges faced by mango farmers in disease control [4]. This review underscores the potential benefits derived from the seamless integration of DIP techniques into the disease detection process, thereby optimizing the overall management of mango orchards and promoting sustainability within the agricultural sector [5].

One remarkable example illustrating the potential of AI/ML-powered DIP techniques in mango disease prevention is the Mango Disease Detector project [6]. This innovative web application empowers users to capture images of mango trees and leverages a trained AI/ML model to predict disease presence with remarkable accuracy. Furthermore, it categorizes the specific disease type and offers tailored recommendations for pesticides and treatments, thereby enabling farmers to proactively protect their mango orchards and embrace sustainable agricultural practices.

By adopting and implementing DIP techniques, mango farmers stand to benefit significantly. These techniques empower them to detect diseases at their incipient stages, facilitating swift intervention and curtailing the spread of infections [2]. Moreover, the application of advanced algorithms guarantees precise disease classification, significantly reducing instances of erroneous diagnoses and optimizing resource allocation for disease management [4].

One of the primary advantages of incorporating DIP techniques is real-time monitoring of mango tree health, providing farmers with the ability to respond promptly to emerging threats. This ultimately contributes to heightened mango productivity and fosters economic growth within the agricultural sector. As the agriculture industry continues its evolution, embracing innovative technologies such as DIP is paramount for the realization of more efficient and sustainable farming practices.

In essence, this literature review serves as a testament to the transformative potential of DIP techniques in mitigating disease-related challenges inherent to mango cultivation. It not only sheds light on their immense value but also underscores the opportunities for further research and advancements in this critical field [5]. By harnessing the capabilities of DIP, mango farmers can fortify their orchards against diseases, thereby ensuring the continued prosperity of this vital sector.

## Disease Classification

### Image Acquisition

Image acquisition is a fundamental step in the process of obtaining digital images for analysis, and it plays a pivotal role in the success of disease detection in mango trees [1]. In this project, the image acquisition process involves capturing or uploading images of mangoes displaying disease symptoms, which are crucial for analysis by the AI/ML model. These images can be obtained through various means, including the use of digital cameras, smartphone cameras, or by accessing pre-existing datasets for training and testing the model [8].

The quality and resolution of these acquired images directly impact the accuracy of the subsequent disease detection and classification processes. Therefore, it is imperative to ensure that the acquired images are of high quality and consistent in size to facilitate effective analysis by the AI/ML model [1].

### Image Processing

#### Image Resizing

Image resizing is a critical aspect of image processing in mango disease detection [8]. It involves altering the dimensions of the acquired images, either by enlarging or reducing their size, to conform to a standardized format. This step ensures that all images fed into the AI/ML model are of the same size, a prerequisite for models like U-Net, which require uniform input dimensions [1].

Careful consideration is essential when resizing images, as improper handling can result in a loss of image quality and important information. Techniques such as interpolation or the application of specific filters can be employed to minimize the loss of data during the resizing process [8].

.

#### Filtering

Image filtering is another crucial component of image processing, serving to enhance and modify images to improve their quality and ease of analysis by the AI/ML model **[1] [9**]. Filters apply mathematical operations to image pixels, which can enhance features of interest and facilitate disease identification within mango trees.

#### Histogram Equalization

Histogram equalization is an image processing technique that plays a vital role in adjusting image contrast by modifying the intensity distribution of the histogram [8]. This process aims to create a linear trend in the cumulative probability function associated with the image, ultimately aiding in the detection of disease-related features.

#### Color Spacing

Color space conversion is an image processing operation that alters the representation and encoding of colors in an image [8]. Different color spaces have distinct ways of representing colors, and converting an image from one color space to another can impact how the image appears and influences the results of subsequent image processing operations. This step is essential for optimizing disease detection; as certain color spaces may be more suitable for specific disease-related feature extraction tasks [1].

.

### Feature Extraction

Feature extraction is a pivotal component in reducing the dimensionality of data, particularly in the context of large datasets, which is often the case in mango disease detection [8]. It involves selecting and combining variables into features that effectively reduce the volume of data while retaining essential information. Feature extraction is instrumental when dealing with extensive datasets, as it minimizes redundant data and streamlines the machine learning process [8][9].

### Convolutional Neural Network

For the detection of mango diseases, Convolutional Neural Network (CNN) algorithms are employed, representing a sophisticated approach to image analysis and classification [9]. CNNs process images in a manner similar to human visual perception, extracting key features from the image and recognizing patterns indicative of disease.

A CNN comprises multiple layers of artificial neurons, with each layer serving a distinct role in feature extraction and recognition [9]. In the initial layers, fundamental components such as edges within the image are identified. As data flows through the network, deeper layers progressively extract more intricate features, including disease-related patterns present in mango leaves [9] [19]**.**

Unlike traditional neural networks that rely on matrix multiplication, CNNs utilize a specialized technique known as convolution. Convolution involves taking the integral of the product of two functions after one has been reflected about the y-axis and shifted, enabling CNNs to effectively capture spatial information within images [10] [19].

Within a CNN, artificial neurons calculate parameters such as sums, weights, and activation values. Each layer within the network employs a unique activation function, and these functions are passed on to the next layer as data is processed [9].

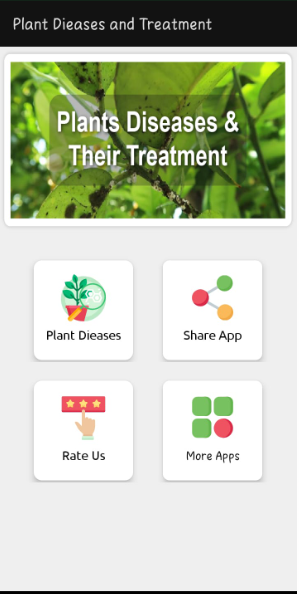
The initial layers of a CNN are adept at identifying basic components or features within an image, such as edges. As the output from these initial layers is propagated to subsequent layers, the network extracts more complex features, including combinations of edges and corners. Deeper layers in the network continue this process, extracting increasingly complex features such as the presence of specific objects or patterns related to diseases in mango leaves [19].

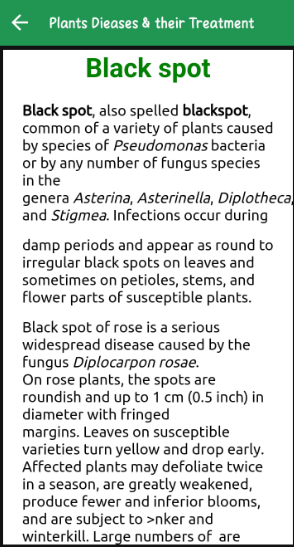
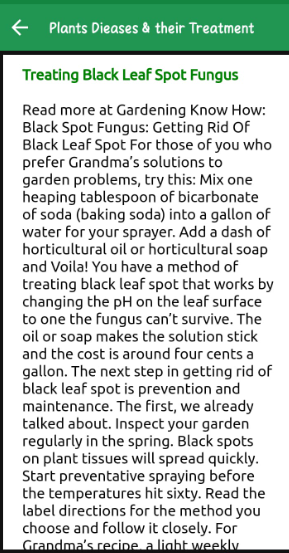
## Similar Application

### Plant Disease & Treatment

This mobile application contains information related to plant disease. It gives the information, why this disease occurs in the plant and how to cure this disease. It was released in the year 2021.

#### Interface

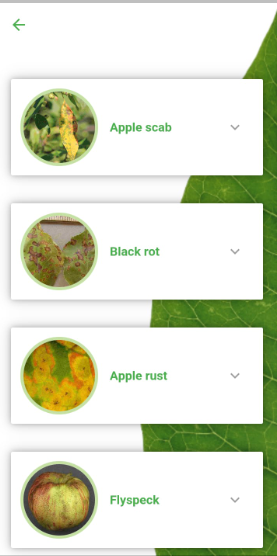
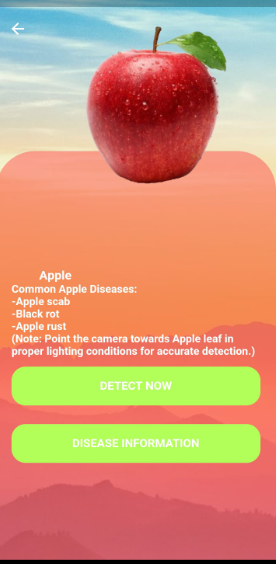


### Plant Disease Identification

It is a mobile application. Through this the user can get the information about different diseases found on different fruits, crops and vegetables. It also showed the prevention and control of the disease. It was released in the year 2020.

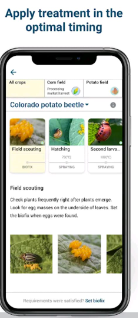
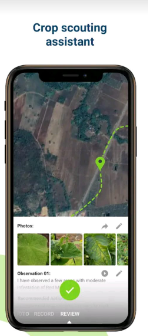
#### Interface

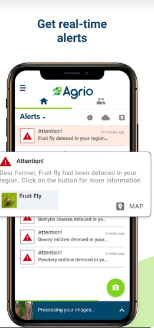
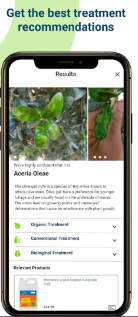


### Angrio

Angrio is a mobile application that helps the growers and crop advisor to forecast, identify, and treat plant diseases and pests, and nutrient deficiencies.

#### Interface

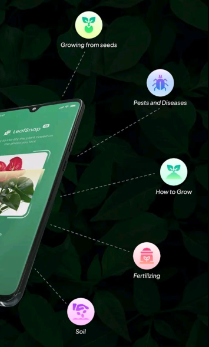
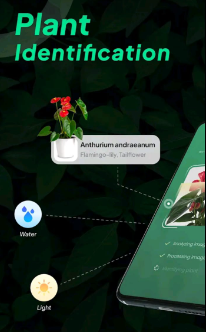


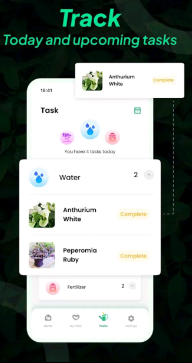
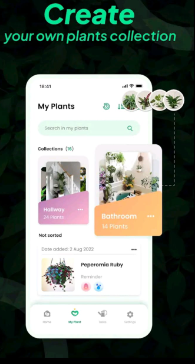


### LeafSnap Plants Identification

It is a mobile application, through which one can identify the plant disease and gives care reminders.

#### Interface





## Comparison of Similar Products

The Mango disease detector shares certain similarities with existing mobile applications that focus on disease prevention; however, it sets itself apart as a comprehensive web and mobile application offering a broader range of features. While the existing applications primarily provide precautions and preventive measures for diseases, our project goes beyond that by incorporating both web and mobile platforms. This dual-platform approach enhances accessibility and usability for users, allowing them to access the application from their preferred devices, be it smartphones or computers.

Moreover, the project incorporates a robust data management system, which sets it apart from the existing applications. User data and history are meticulously maintained, creating a personalized experience for each user. This feature ensures that the application can track the user's previous interactions, disease reports, and treatment history, enabling a seamless experience over time. With such data at hand, the application can provide more accurate and tailored recommendations for disease management and prevention, further enhancing the effectiveness of its services.

The core functionality of our project revolves around disease detection and classification. When a user uploads an image of a mango tree, the AI/ML-powered system analyzes the image to identify the presence of any disease. This analysis takes into account various image processing techniques, including segmentation and feature extraction, to accurately determine the disease's type and severity. The application then generates a detailed and informative report outlining the detected disease and its recommended treatment methods.

By providing disease-specific reports based on user-provided images, our project offers valuable insights to farmers and orchard owners. The detailed reports help users make informed decisions about disease management strategies, allowing them to take prompt and appropriate actions to prevent further spread and minimize the impact on mango productivity.

The integration of both web and mobile platforms ensures that users have the flexibility to access the application whenever and wherever they need it. Whether in the field tending to their mango trees or in the comfort of their homes, users can seamlessly utilize the application to monitor and manage the health of their orchards effectively.

In conclusion, while the project shares similarities with existing mobile applications in terms of disease prevention, its dual-platform approach and data management capabilities elevate its functionality. The focus on disease detection, classification, and personalized disease reports distinguish it as a valuable tool for mango farmers and orchard owners. By combining advanced image processing techniques with AI/ML algorithms, the project empowers users to make informed decisions and take proactive measures to protect and optimize their mango orchards, contributing to increased productivity and sustainable agricultural practices.

## Chapter Conclusion

In this project, the primary focus is to address approximately 4 to 5 different diseases that commonly affect mango trees. Our overarching goal is to develop a comprehensive web and mobile application that can generate detailed reports of mango farms, accurately identifying both healthy and unhealthy leaves. To achieve this, we will train our AI/ML model to recognize unhealthy leaves and, if present, detect the specific disease affecting those leaves from the list of pre-identified diseases.

Figure 2.3 outlines the flow of our project, depicting the sequential steps from input image to disease detection and report generation. Users will input images of mango leaves into the web application, which will undergo a series of preprocessing steps to enhance the image quality and remove any noise or imperfections. Histogram and color transformations will be applied, resulting in clearer and more refined images suitable for further analysis.

Next, we will employ the Convolutional Neural Network (CNN) algorithm, a powerful deep learning model, for disease detection. The CNN will initially classify the leaf as either healthy or unhealthy. If the leaf is identified as unhealthy, the model will proceed to analyze it further using the list of diseases it has been trained on. By effectively identifying unhealthy leaves and detecting the presence of any known diseases from our specified list, the model can provide valuable insights for farmers and orchard owners.

Our web and mobile application's ultimate output will be a comprehensive report summarizing the health status of the mango farm. This report will include detailed information about healthy and unhealthy leaves, along with any specific diseases detected. Armed with this data, farmers can take informed actions to address disease outbreaks, implement targeted treatments, and optimize their disease management strategies.

By incorporating image preprocessing techniques, and by using CNN algorithm, our project seeks to revolutionize disease detection and prevention in mango cultivation. The ability to identify and classify diseases accurately will empower farmers to proactively address potential threats and minimize the impact on mango production. Additionally, generating comprehensive reports will facilitate better decision-making, contributing to healthier mango orchards and ensuring sustainable agricultural practices. With these advancements, we aim to support mango farmers in their efforts to enhance productivity and secure a prosperous future for the mango industry.

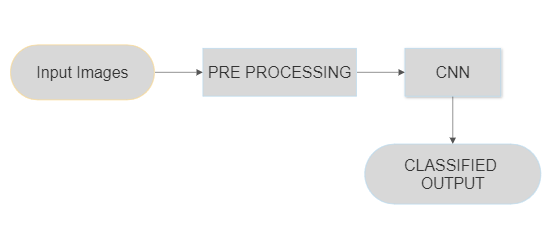


Figure 2.5 Flow of the Project

# Aim and Statement of Problem

## Problem Statement

Agriculture, being one of the most vital sectors in the global economy, plays a pivotal role in shaping the prosperity of nations. Pakistan, like many other countries, heavily relies on its agricultural sector as a significant contributor to its economic growth and sustenance. With a vast number of people associated with farming and agriculture-related activities, the well-being of this sector directly impacts the livelihoods of millions in the country.

One of the prominent agricultural products that Pakistan excels in is the production of mangoes. Known for their exquisite taste and premium quality, Pakistani mangoes are sought after worldwide. However, despite its success as an exporter, the mango industry faced a severe setback in 2019 due to two major factors: malformation and global warming.

Malformation of mango trees and farms resulted in significant economic losses for both farmers and the nation. Malformation is a condition where the natural growth of mango trees is disrupted, leading to abnormal shapes and structures in the fruits. This negatively affects the yield and quality of mangoes, making them unsuitable for international markets and diminishing their economic value. The loss of potential revenue due to malformation highlighted the urgent need to address this issue and safeguard the livelihoods of farmers and the country's economy.

Additionally, the adverse effects of global warming have been increasingly evident in the agricultural landscape. Climate change has disrupted traditional weather patterns, causing unpredictable and extreme weather events. Unseasonal rains, heatwaves, and erratic temperatures have had detrimental effects on mango orchards, leading to reduced yields and compromised fruit quality. The impact of global warming on the agricultural sector underscores the importance of implementing measures to mitigate its effects and build climate-resilient agricultural practices.

While natural disasters and global warming are beyond human control, the management of diseases affecting mango trees lies within our reach. Early detection of diseases in mango trees is crucial in curbing their spread and minimizing the damage to entire farms. Timely identification of diseases allows farmers to implement targeted disease management strategies, such as applying appropriate pesticides or adopting disease-resistant varieties, thus preventing the further escalation of the issue.

In light of the challenges faced by the mango industry, the development of a reliable disease detection system becomes paramount. The Mango Disease Detector project seeks to address this need by leveraging the power of Digital Image Processing (DIP) techniques and Artificial Intelligence (AI) . The application aims to provide an efficient and accurate means of detecting diseases in mango trees at their early stages, enabling prompt intervention and containment.

By utilizing DIP techniques, the application processes and enhances images of mango leaves provided by users. AI algorithms, particularly Convolutional Neural Networks (CNNs), then analyze these images to detect signs of diseases accurately. The AI-powered system classifies leaves as healthy or unhealthy, and if a disease is detected, it identifies the specific disease type from a list of 4 to 5 common mango diseases **[17]**. The application then generates comprehensive reports for users, presenting the health status of their mango farms and offering recommendations for disease management and treatment.

The agriculture sector is a lifeline for Pakistan's economy, and the mango industry holds significant importance as a major exporter. To protect the interests of farmers and ensure sustained economic growth, addressing issues like malformation and the effects of global warming becomes imperative . The Mango Disease Detector project presents a promising solution, aiming to empower farmers with early disease detection and effective disease management strategies. By harnessing the potential of AI and DIP, this project seeks to safeguard the quality and quantity of mango production, fortifying the resilience of the agricultural sector and contributing to the overall prosperity of the nation.

## AIM

The "Mango Disease Detector" project is a groundbreaking endeavor aimed at detecting diseases present in mango trees and providing targeted pesticide recommendations for effective disease management**.** By implementing this innovative web application, farmers can protect their orchards from potential damage caused by diseases, thereby directly impacting the nation's economy.

The web and mobile application will serve as a user-friendly platform where farmers can simply input images of mango trees for analysis. These images will undergo a series of critical steps to ensure accurate disease detection.

Image processing techniques will be applied to refine the image quality and remove any unwanted noise or artifacts. This step plays a crucial role in preparing the images for advanced analysis by eliminating any distortions that could potentially hinder disease detection accuracy.

The most important step involves the implementation of cutting-edge AI/ML algorithms. These algorithms, particularly Convolutional Neural Networks (CNNs), have shown exceptional capabilities in image recognition and classification tasks. By training the AI model on a diverse dataset containing both healthy and diseased mango leaf images, the system can learn to differentiate between healthy and unhealthy leaves accurately.

Upon completion of the AI/ML analysis, the application will display the results of the disease detection process to the users . If any diseases are detected, the system will further identify the specific type of disease from a predetermined list of 10-15 common mango diseases. This level of specificity is crucial in providing targeted treatment recommendations to farmers.

The generated report will offer farmers comprehensive insights into the health status of their mango farms . They will have access to details about the number of affected trees, the type of disease found, and the severity of the infections. Armed with this valuable information, farmers can take informed actions and implement appropriate treatment measures promptly.

Through the "Mango Disease Detector" project, farmers will have a powerful tool at their disposal to proactively manage their mango orchards. Early detection of diseases is instrumental in preventing the spread of infections and minimizing the economic losses associated with reduced fruit quality and yield.

Moreover, the application's ability to provide targeted pesticide recommendations is a significant advantage . By using the right pesticides for specific diseases, farmers can optimize resource utilization and minimize environmental impacts, contributing to sustainable agricultural practices.

Beyond its direct benefits to individual farmers, the "Mango Disease Detector" project holds broader implications for the agriculture sector. By enhancing disease management practices, the project can contribute to increased mango production, bolstering Pakistan's position as a major exporter of premium-quality mangoes.

The "Mango Disease Detector" web and mobile application is poised to revolutionize disease detection and management in the mango industry. By integrating image segmentation, processing, and advanced AI/ML algorithms, the application empowers farmers with timely, accurate disease diagnosis and tailored treatment recommendations. This powerful tool enables farmers to safeguard their mango orchards, protect the nation's economy, and promote sustainable agricultural practices. Through the successful implementation of this project, the agriculture sector can witness enhanced productivity and resilience, fostering a prosperous future for mango cultivation in Pakistan.

## Project Scope

The primary objective of the "Mango Disease Detector" project is to enable early identification of diseases affecting mango trees, providing farmers with valuable insights for effective disease management. The project encompasses several key components that fall within its scope:

### In Scope

* **Capturing and Storing Images:** The application allows users to capture images of mango leaves, which are then stored in a secure database. This step serves as the foundation for disease detection and analysis.
* **AI/ML Image Processing:** Once the images are available in the database, the AI/ML model takes over to process and analyze them. Advanced image processing techniques, including segmentation and feature extraction, are employed to enhance the accuracy of disease detection.
* **Predictions and Database Storage:** The AI/ML model generates predictions based on the analysis of the input images. The results are then stored in the database, creating a record of disease detections for each mango leaf image.

* **Summarized Report Generation:** The application compiles the disease detection results from the database and generates a comprehensive and summarized report. This report provides an overview of the health status of the entire mango farm, highlighting the presence and severity of any diseases.
* **Disease Management Suggestions:** In addition to the summarized report, the application offers disease management suggestions to farmers. Based on the specific diseases detected, targeted pesticide recommendations are provided to aid farmers in effectively treating and controlling infections.
* **Web Server Deployment:** The "Mango Disease Detector" application is deployed on a web server, ensuring universal accessibility. Farmers can access the application from anywhere and at any time, enhancing convenience and usability.

### Out of Scope

While the project addresses several critical aspects of disease detection and management, certain components fall outside its scope:

* **Pesticide Selection:** The application suggests pesticides for disease management; however, it does not encompass every pesticide available in the market. The focus is on recommending pesticides with established recommendations and positive reviews, ensuring farmers have access to reliable and trusted solutions.
* **Pesticide Usage Demonstration:** The "Mango Disease Detector" app does not provide a step-by-step demonstration of how pesticides should be used on fields or orchids. The application assumes that farmers are familiar with the appropriate methods of pesticide application or will seek guidance from agricultural experts for safe and effective use.

## Chapter Conclusion

In conclusion, the "Mango Disease Detector" project is a comprehensive endeavor aimed at empowering farmers with crucial disease management information. By capturing images of mango leaves and utilizing AI/ML image processing techniques, the application enables early disease detection, allowing farmers to take timely action and prevent the spread of infections. The provision of summarized reports and targeted pesticide recommendations further enhances the project's practicality and value to the agricultural community. While the app does not encompass every available pesticide or provide usage demonstrations, its primary goal of disease prevention remains paramount, contributing to improved mango productivity and supporting sustainable agricultural practices. Through the deployment of the application on a web server, farmers gain easy access to vital disease management insights, strengthening their ability to protect their mango farms and ensure long-term agricultural prosperity.

# Hardware, Software analysis and requirements

This chapter comprises the hardware, and the software requirements. It also includes fact-finding techniques, by using that we develop the application.

## Fact Finding Technique

The technique we use to analyze the main information related to our project at the initial stage are:

* Research
* Prototyping

### Research

The process of fact-finding is a crucial research technique that plays a vital role in understanding the scope and requirements of a project [5]. It involves gathering and analyzing relevant information to gain insights into various aspects of the project, enabling better decision-making and enhancing its overall effectiveness.

In the "Mango Disease Detector" project, fact-finding serves as the foundation for understanding the key challenges and goals of the application. It involves collecting data on the mango farming industry, prevalent diseases affecting mango trees, existing disease detection methods, and the needs and expectations of the farmers.

One of the primary benefits of fact-finding is that it provides a comprehensive view of the project's scope. By conducting thorough research and gathering data from various sources, the project team can identify the specific functionalities and features required to meet the farmers' needs effectively [5]. This helps in setting clear and realistic project goals, ensuring that the final product aligns with the expectations of the end-users.

Fact-finding also plays a crucial role in identifying the best algorithms and techniques to be implemented in the "Mango Disease Detector." Through research and analysis, the team can evaluate various algorithms used in disease detection and classification tasks [6]. This allows them to select the most suitable algorithm, such as the Convolutional Neural Network (CNN), known for its effectiveness in image recognition and classification. By understanding the strengths and limitations of different algorithms, the project team can make informed decisions that optimize the application's performance and accuracy [5].

Furthermore, fact-finding helps in understanding the specific challenges faced by mango farmers regarding disease detection and management. The fact-finding process also aids in determining the appropriate image preprocessing techniques for enhancing the quality and clarity of mango leaf images [16]. By researching and experimenting with various image processing methods, the team can identify the most effective techniques for noise reduction, contrast enhancement, and image normalization. This step is critical in ensuring that the CNN model receives high-quality input images for accurate disease detection [6].

Moreover, fact-finding supports the development of a user-friendly and accessible interface for the web and mobile application. By understanding the preferences and requirements of the end-users, the project team can design an intuitive and easy-to-navigate application that caters to the farmers' needs. The interface can be optimized to facilitate seamless image uploading or capturing, and the application can provide real-time feedback and guidance to the users during the disease detection process.

Fact-finding is an indispensable research technique that plays a pivotal role in shaping the "Mango Disease Detector" project[5]. By gathering and analyzing relevant information, the project team gains a comprehensive understanding of the project's scope, goals, and challenges. It facilitates the selection of the most suitable algorithms, the identification of optimal image preprocessing techniques, and the development of targeted disease management recommendations [6]. Through fact-finding, the project team can ensure that the application meets the needs of mango farmers, empowering them with timely and accurate disease detection and management tools. This approach ultimately contributes to enhanced mango productivity, improved fruit quality, and sustainable agricultural practices, benefiting both the farmers and the nation's economy.

### Prototyping

The prototyping technique of fact-finding is a valuable approach used in the analysis of all functional requirements of a project. By creating prototypes, which are simplified versions of the final product, stakeholders can gain a clear understanding of what the end product will look like and how it will function. This iterative process allows for early validation and refinement of the project's design and functionality, minimizing the chance of errors and ensuring a successful development stage.

Once the functional requirements are identified, the team begins the prototyping phase. Prototypes are early representations of the final product, often created using mock-ups or wireframes. These prototypes are not fully functional applications but serve as visual aids that demonstrate the flow, layout, and basic functionalities of the future application.

In the "Mango Disease Detector" project, the prototyping technique starts by creating mock-ups of the application's user interface (UI). After refining the UI design, the team proceeds to create interactive prototypes. These non-functional prototypes allow stakeholders to experience the application's user journey and explore different functionalities. For the "Mango Disease Detector," the interactive prototype may include features such as image uploading, preprocessing, disease classification, and displaying disease management recommendations.

The iterative nature of prototyping enables stakeholders to provide feedback at multiple stages of development. As the interactive prototypes are presented and evaluated, stakeholders can suggest improvements and modifications, which the team can quickly incorporate into subsequent iterations. This continuous feedback loop ensures that the final product aligns with the stakeholders' expectations and meets the project's functional requirements.

Furthermore, the use of non-functional prototypes is particularly advantageous in minimizing the chances of errors at the development stage. By simulating the application's behavior and user interactions, potential issues can be identified early on. For example, if the prototype reveals that certain user actions result in unexpected outcomes or errors, the team can address these issues before actual development begins. This proactive approach significantly reduces the likelihood of costly and time-consuming rework during the later stages of the project.

Moreover, the prototyping technique promotes collaboration and fosters a deeper understanding of the project's objectives among all stakeholders. By involving stakeholders in the prototyping process, they become active participants in shaping the final product. This collaborative approach enhances communication, mitigates misunderstandings, and ensures that the project's functional requirements are well-aligned with the stakeholders' needs and vision.

The prototyping technique of fact-finding is a powerful tool in the analysis of functional requirements for the "Mango Disease Detector" project. By creating interactive prototypes, the team can visualize the application's design and functionality, enabling stakeholders to provide valuable feedback and validate the product's suitability early in the development process. This iterative and collaborative approach minimizes the chances of errors, ensures the project's success, and delivers an application that empowers farmers with accurate disease detection and management capabilities. Through the use of prototyping, the "Mango Disease Detector" project can contribute to enhanced mango productivity, improved fruit quality, and sustainable agricultural practices, benefiting both farmers and the nation's agricultural sector as a whole..

## Hardware Requirements

The "Mango Disease Detector" is a versatile web and mobile application designed to be accessible from various devices with internet connectivity. Users can access the application through any laptop or computer that meets the specified requirements [18]. To download and run the application seamlessly, certain hardware and software specifications need to be met [2].

Firstly, for laptops and computers, the minimum requirement for RAM is 2GB [17]. RAM (Random Access Memory) is crucial for the application's smooth performance, as it allows the device to handle multiple tasks simultaneously and efficiently. With 2GB of RAM, the application can run without significant lag, ensuring a seamless user experience.

For mobile devices, the application is compatible with Android versions 10 or higher . This means that devices running Android 10, 11, or the latest Android versions can support the "Mango Disease Detector." Android 10 introduced various performance improvements and enhanced features, making it an optimal platform for running advanced applications like the "Mango Disease Detector."

In terms of storage, the application requires a certain amount of space on the device [2]. While the exact storage requirement may vary based on updates and additional features, a sufficient amount of available storage is necessary to download and install the application. Adequate storage ensures that users can store the application and use it without encountering space-related issues.

By meeting these specifications, users can download and install the "Mango Disease Detector" application on their laptops, computers, or Android devices with ease. The application's web-based platform ensures that it is accessible from any laptop or computer with an internet connection. Users can visit the application's website through their preferred web browsers and access its functionalities without needing to download or install any additional software [3].

For Android users, the application can be downloaded from the Google Play Store [18]. Devices with Android 10 or higher can seamlessly install the application and utilize its features for disease detection in mango trees. The application's compatibility with the latest Android versions ensures that users can take advantage of the latest technology advancements and features available on their devices.

In conclusion, the "Mango Disease Detector" application is designed to be user-friendly and accessible to a wide range of users [17]. By adhering to the specified hardware and software requirements, users can download and run the application on their laptops, computers, or Android devices [18]. With 2GB of RAM, Android version 10 or higher, and sufficient storage, users can leverage the application's disease detection capabilities to safeguard their mango orchards and enhance agricultural productivity. The application's web-based platform further extends its accessibility, enabling users to access its features conveniently through any laptop or computer with internet connectivity [3].

## Software Requirements

### For Training of Model

#### Language

The model will be implemented by using Python’s library of TensorFlow.

#### IDLE

Jupyter Notebook or PyCharm will be use in the training of the model.

### For Web Application

#### Language

Django Framework for developing the Web Application. This framework is based on Python.

#### IDLE

PyCharm will be use in the development of the web application.

### For Mobile Application

#### Language

For the development of the mobile application Java is being used.

#### IDLE

Android Studio is used in the development of mobile apps.

## Chapter Conclusion

In conclusion, the "Mango Disease Detector" project employs various fact-finding techniques to analyze and understand its scope and requirements. Research plays a vital role in gathering relevant information about the mango farming industry, prevalent diseases affecting mango trees, and the needs of farmers. This information forms the foundation for defining the project's functionalities and selecting suitable algorithms like the Convolutional Neural Network (CNN) for disease detection and classification.

Prototyping is another crucial fact-finding technique used in the project's analysis. By creating interactive prototypes, stakeholders can visualize the application's design and functionality, providing valuable feedback for refinement. The iterative nature of prototyping ensures that stakeholders can be actively involved in shaping the final product, reducing the chances of errors during development and enhancing the overall user experience.

Regarding the hardware requirements, the "Mango Disease Detector" is accessible through web and mobile applications. For laptops and computers, a minimum of 2GB RAM is required to ensure smooth application performance. For mobile devices, the application is compatible with Android version 10 or higher, making it accessible to a wide range of Android users. Adequate storage space is necessary to download and install the application without any storage-related issues.

The software requirements encompass the tools and frameworks utilized for different aspects of the project. Python's TensorFlow library is used for training the model, while Jupyter Notebook or PyCharm serves as the Integrated Development Environment (IDE) during the model training process.

For the web application, the Django framework, based on Python, is chosen for development, and PyCharm is used as the IDE for web application development. This allows for the creation of a user-friendly and accessible interface that caters to the needs of farmers, providing real-time disease detection and management guidance.

In the case of the mobile application, Java is employed as the programming language, and Android Studio is the IDE of choice. This combination ensures seamless development of the mobile app, allowing Android users to access the "Mango Disease Detector" on their devices conveniently.

The "Mango Disease Detector" project demonstrates a comprehensive approach to fact-finding, employing research and prototyping techniques to develop an effective and user-friendly solution for mango disease detection and management. By leveraging the selected algorithms and software tools, the application empowers farmers with timely disease detection, reducing potential crop losses and contributing to the enhancement of mango productivity. Through its web and mobile applications, the "Mango Disease Detector" aims to be an accessible and valuable tool in the agricultural sector, fostering sustainable practices and supporting the economy. With its focus on accurate disease identification and recommendations for effective management, the "Mango Disease Detector" project stands to benefit farmers and contribute to the advancement of modern agricultural practices. As technology continues to advance, such applications pave the way for innovative solutions to agricultural challenges, fostering a more sustainable and productive future for the agriculture industry.

# Software design and modeling

This chapter comprises the analysis of the system. It contains the class diagram, interaction diagram, state diagram, Data flow diagram, Activity diagram, Sequence diagram, and Use case diagram.

## Sequence Diagram

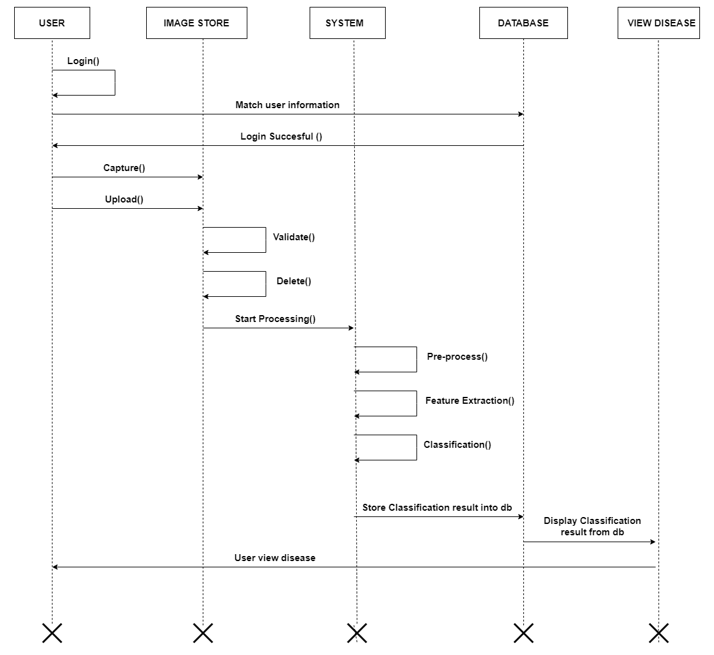


Figure 5.1 Sequence Diagram

### Description

The sequence diagram shown in Figure 5.1 illustrates the step-by-step flow of the "Mango Disease Detector" system, detailing the interactions between the user, the application, and the underlying database and AI/ML model.

The process begins with the user accessing the mobile application or web app and attempting to log in. The user enters their credentials, such as username and password, which are then verified against the data stored in the application's database. If the entered credentials match the database records, the user successfully logs in to the system and gains access to the application's functionalities.

After a successful login, the user can proceed to predict the disease in the mango leaves. The user has two options for inputting the mango leaf image: they can either upload a pre-captured picture from their device's gallery or capture an image directly from the application using their phone's camera . Whichever option they choose, the selected image is stored in the image store for further processing.

Once the image is acquired, the system validates the image to ensure its integrity and authenticity. The validation process checks for any potential issues with the image and ensures that it meets the required criteria for further processing. If the image passes the validation, it is considered suitable for disease prediction; otherwise, it may be rejected or deleted from the image store.

With the successful acquisition and validation of the image, the system now proceeds to process the image for disease prediction. The image is passed to the trained AI/ML model, which is responsible for disease detection. The model undergoes several key steps for accurate disease classification.

Firstly, the image goes through preprocessing, where various image enhancement techniques are applied to improve its quality and remove any potential noise or artifacts. Preprocessing ensures that the image is in a suitable format for feature extraction and classification.

Next, the system performs feature extraction on the preprocessed image. This step involves identifying relevant patterns, shapes, and characteristics in the image that are indicative of different diseases. The extracted features serve as the basis for the subsequent disease classification.

Finally, the AI/ML model classifies the uploaded image based on the extracted features. The model has been trained on a labeled dataset to recognize various diseases affecting mango leaves. Through the process of classification, the system determines the type of disease present in the uploaded image.

Once the classification process is completed, the processed image and disease prediction are stored in the database . The user can view the results of the disease prediction under the "View Disease" section of the application. The system displays the processed image along with the identified disease, providing the user with valuable insights into the health of their mango trees.

## State Diagram

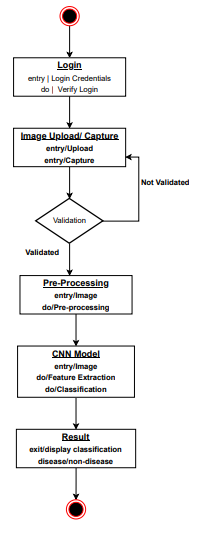


Figure 5.2 State Diagram

### Description

Figure 5.2 illustrates the state diagram of the "Mango Disease Detector" system, showcasing the various stages and interactions involved in the disease detection process. The diagram demonstrates the seamless flow of actions, starting from user login to the final disease classification result.

The process commences with the user logging into the application. Upon accessing the web or mobile app, the user is prompted to enter their credentials, such as a username and password. The system then performs a verification process by matching the entered credentials with the ones stored in the application's database. If the entered credentials match the database records, the user is granted access to the system and successfully logged in. After successful login, the user is presented with the option to upload an image from their device's image gallery or directly capture an image using the application's built-in camera. The selected image is then processed for further validation. The system performs a validation check on the uploaded or captured image. This validation function checks whether the size of the image is appropriate for processing. If the image size is correct, the system proceeds with further processing steps. However, if the image size is incorrect or does not meet the required criteria, the user is notified to upload or capture the image again. Upon successful validation of the image, the system proceeds with image preprocessing techniques. Image preprocessing involves applying various enhancement techniques to improve the image quality, remove noise, and standardize the image format. This preprocessing step ensures that the image is in a suitable state for the subsequent classification process. The core of the "Mango Disease Detector" system lies in the use of Convolutional Neural Network (CNN) for image classification. The preprocessed image is fed into the CNN model, which has been trained on a large dataset of mango leaf images with labeled disease categories. The CNN model analyzes the features and patterns present in the input image and performs disease classification based on its learning from the training data. After the classification process is completed, the system generates the result of the disease detection. The user is then presented with the output, indicating whether the mango leaf in the uploaded or captured image is diseased or not. The result is displayed to the user on the application's interface, providing valuable information about the health status of their mango trees.

## Activity Diagram

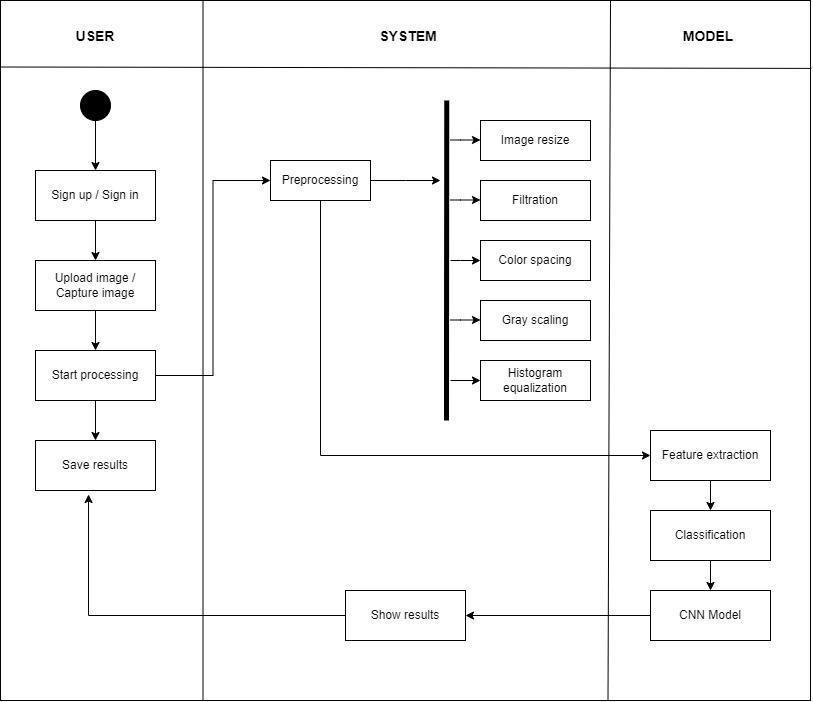


Figure 5.3 Activity Diagram

### Description

In Figure 5.3, the activity diagram presents a detailed representation of the interactions and processes involved in our Mango Disease Detector system. The user's activity initiates the entire workflow, and they have the option to either sign in if they already have an account or sign up if they are new to the system. This user authentication step ensures secure access to the system's features and functionalities.

Once the user is authenticated, they have two choices for inputting the image data. They can either upload an image from their system's storage or directly capture an image using the device's camera. This flexibility allows users to use the method most convenient for them, whether they have pre-existing images or need to capture images of mango crops on the spot.

After the image is selected, the user clicks on the "Start Process" button, triggering the start of the pre-processing step. In this crucial phase, the selected image undergoes various transformations to enhance its quality and suitability for analysis. Image resizing ensures that the image is in a standardized format, while filtration helps to remove any noise or unwanted elements that may interfere with the analysis process.

Furthermore, color spacing and grey scaling are applied to standardize the color representation, ensuring that the system is not biased towards certain color variations. Histogram equalization is also performed to improve the contrast and overall quality of the image, ensuring that important features are emphasized for better analysis.

Following pre-processing, the feature extraction phase comes into play. Feature extraction is a pivotal part of the dimensionality reduction process, where the raw data of the image is divided and reduced into more manageable and relevant groups of information. This extraction process is crucial for identifying the unique characteristics and patterns present in the mango crop images that can be indicative of different diseases.

In the classification phase, the powerful Convolutional Neural Network (CNN) algorithm takes center stage. The features extracted from the previous step are fed into the CNN, which is specifically designed for image recognition tasks. The CNN utilizes multiple convolutional layers to detect patterns, edges, and textures within the image. These learned features are then passed on to fully connected layers that perform the classification of the mango crop image into specific disease categories.

The classification process involves the use of a softmax activation function, which calculates the probabilities of the image belonging to different disease classes. The class with the highest probability is considered the detected mango disease. This robust and efficient classification process ensures accurate and reliable disease identification, enabling farmers and agricultural experts to make informed decisions for disease management and crop health.

Once the classification is completed, the results are generated based on the findings of the system. The results may include the specific disease identified, the severity level, and any additional recommendations or suggestions for managing the detected disease. These results are then presented to the user through a user-friendly interface, allowing them to access and interpret the information easily.

In conclusion, the activity diagram in Figure 5.3 outlines the step-by-step process of the Mango Disease Detector system, from user authentication to image input, pre-processing, feature extraction, classification, and result presentation. This comprehensive and well-structured workflow ensures the accurate detection and management of mango diseases, providing valuable support to farmers and agricultural experts in ensuring healthy and productive mango crops. The incorporation of advanced techniques, such as CNN and image pre-processing, reinforces the system's efficiency and reliability in disease detection, making it a valuable tool in modern agriculture practices.

## Communication Diagram

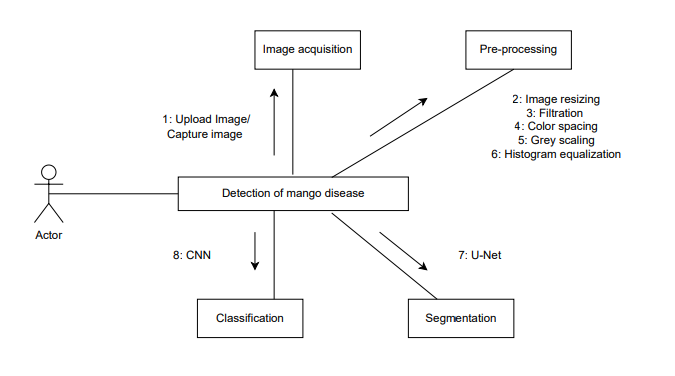


Figure 5.4 Communication Diagram

### Description

The communication diagram depicted in Figure 5.4 offers a comprehensive overview of the intricate process flow within our Mango Disease Detector system. At the heart of this system lies the user, represented as the actor, who acts as the primary initiator of the entire workflow. The user is granted two distinct choices to input image data, providing flexibility and convenience in accessing the system's functionalities.

The first option is to upload an image from their system's storage, enabling users to utilize pre-existing images of mango crops for analysis. This is particularly useful for situations where the user possesses a collection of images from previous crop assessments or research. The second option allows the user to directly capture an image using their device's camera, making it convenient for on-the-spot assessments of mango crops in the field. This versatility caters to a wide range of user needs and preferences, ensuring that the system is accessible and user-friendly.

Once the user selects and inputs the image, the process enters the image acquisition phase. During this phase, the system acquires and retrieves the image data for further analysis. The acquired image is then subjected to a series of essential preprocessing steps, aimed at optimizing its quality and suitability for disease detection.

The preprocessing phase includes several critical techniques that enhance the image's characteristics. Image resizing is performed to standardize the image dimensions, ensuring consistency and ease of analysis. Filtration techniques are applied to remove any noise or unwanted artifacts present in the image, which could potentially interfere with accurate disease identification. Color spacing normalization is executed to standardize the color representation across different images, mitigating any bias towards specific color variations. Additionally, grey scaling is employed to convert the image into a grayscale format, reducing computational complexity while preserving essential image information. Finally, histogram equalization is applied to enhance the image's contrast and overall quality, allowing for better visibility of crucial features during the subsequent analysis stages.

The next phase in the system's communication diagram is the classification phase. In this stage, the preprocessed image data is fed into the powerful Convolutional Neural Network (CNN) algorithm. The CNN is specifically designed for image recognition tasks, capable of extracting intricate patterns, edges, and textures from images. It does so by utilizing multiple convolutional layers, each responsible for identifying specific features in the image. These learned features are then passed on to fully connected layers, where the image is recognized and classified into specific disease categories.

The utilization of the softmax activation function plays a pivotal role in the classification process. This function computes the probabilities of the image belonging to different disease classes. Consequently, the class with the highest probability is designated as the detected mango disease. This accurate and reliable classification process empowers the system to precisely identify the presence of various diseases in mango crops, providing invaluable insights for farmers and agricultural experts.

Overall, the systematic workflow demonstrated in the communication diagram ensures seamless interaction between the user and the Mango Disease Detector system. The integration of advanced techniques, such as CNN and image preprocessing, guarantees the system's efficiency and reliability in disease detection. The outcome is a robust and user-friendly tool that aids farmers and agricultural experts in making informed decisions regarding disease management and crop health.

The impact of the Mango Disease Detector extends beyond the individual user level. By aiding in the timely and accurate identification of mango diseases, the system contributes to improved crop yield and overall agricultural productivity. This, in turn, supports sustainable agricultural practices, better resource allocation, and ultimately helps in addressing food security challenges.

## Component Diagram

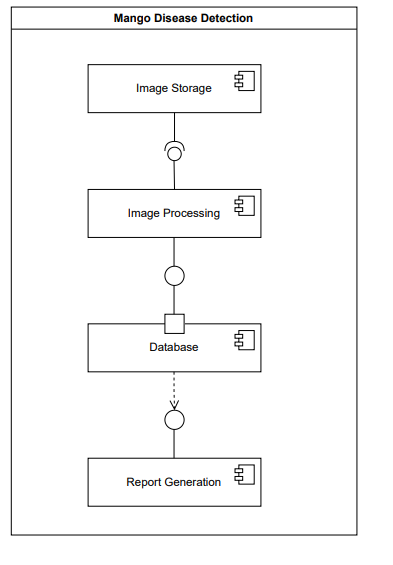


                            Figure 5.5 Component Diagram

### Description

Figure 5.5 presents a comprehensive component diagram that illustrates the intricate interactions between the various components within our Mango Disease Detector system. Each component plays a crucial role in enabling the system's functionalities and ensuring a smooth and efficient user experience. The diagram provides a visual representation of the relationships and dependencies between the components and the system as a whole.

The arrows in the diagram depict the dynamic relationships between the components and the system. These relationships are essential for the seamless flow of information and data exchange, ensuring that the system operates cohesively to deliver accurate and reliable disease detection results.

One of the significant aspects highlighted in the component diagram is the presence of a dotted arrow, representing a conditional access scenario. This signifies that certain components are accessible by the user under specific conditions, while at other times, access might be restricted. Specifically, if the user chooses to upload images for analysis, the system will actively engage and execute the disease detection process. However, in the absence of user image uploads, the system remains dormant, awaiting user input to initiate disease detection. This conditional access ensures that the system is resource-efficient and only activates its processing capabilities when required, optimizing its overall performance.

Additionally, the component diagram highlights the indispensable nature of certain components, represented by solid arrows. These components are critical for the system's functionality and are inherently intertwined with its core operations. For instance, the presence of a user interface is essential for enabling user interaction with the system. Without the user interface, the system would lack the means to receive user inputs and present the disease detection results effectively. The user interface serves as the primary gateway through which users can interact with the system and obtain valuable insights into their mango crops' health.

The user interface component serves as the platform through which the system generates and showcases the results of the disease detection process. Users can seamlessly navigate the user interface to access the information and analysis provided by the system. The user interface acts as the bridge between the technical complexities of the system's backend processes and the user's intuitive understanding, making the disease detection process accessible and user-friendly.

A critical aspect of the Mango Disease Detector system lies in its sophisticated use of image processing techniques and functions. These components work in harmony to preprocess the uploaded images, optimize their quality, and extract relevant features for disease classification. Image processing techniques, such as resizing, filtration, color spacing normalization, grey scaling, and histogram equalization, collectively enhance the image data's suitability for further analysis. This preparatory phase is vital for ensuring that the system accurately recognizes and classifies mango diseases, providing users with dependable results.

Furthermore, the system's reliance on a database component underscores the significance of data storage and management in disease detection. The database acts as a repository for storing the results generated during the disease detection process. This database preserves a historical record of detected diseases, facilitating data-driven decision-making and aiding in monitoring crop health trends over time. The database component ensures that the system can efficiently store and retrieve relevant information, enabling users to access past disease detection outcomes and make informed choices regarding disease management strategies.

In conclusion, the component diagram depicted in Figure 5.5 captures the interconnected nature of the Mango Disease Detector system. Its portrayal of conditional access, indispensable components, and the critical role of image processing and database functionalities collectively contribute to a holistic understanding of the system's operations. The diagram emphasizes the seamless interactions between components, facilitating an effective and user-friendly disease detection experience for farmers and agricultural experts. By leveraging these components' synergistic power, the Mango Disease Detector system empowers users with accurate disease detection and analysis, facilitating improved crop yield and supporting sustainable agricultural practices. As the system continues to evolve and incorporate advancements in technology, its potential to revolutionize disease management in mango crops and enhance agricultural productivity remains ever prominent.

## Class Diagram

A class diagram describes the structure of the system. It shows the relationship, attributes, and relationship between the system and user.

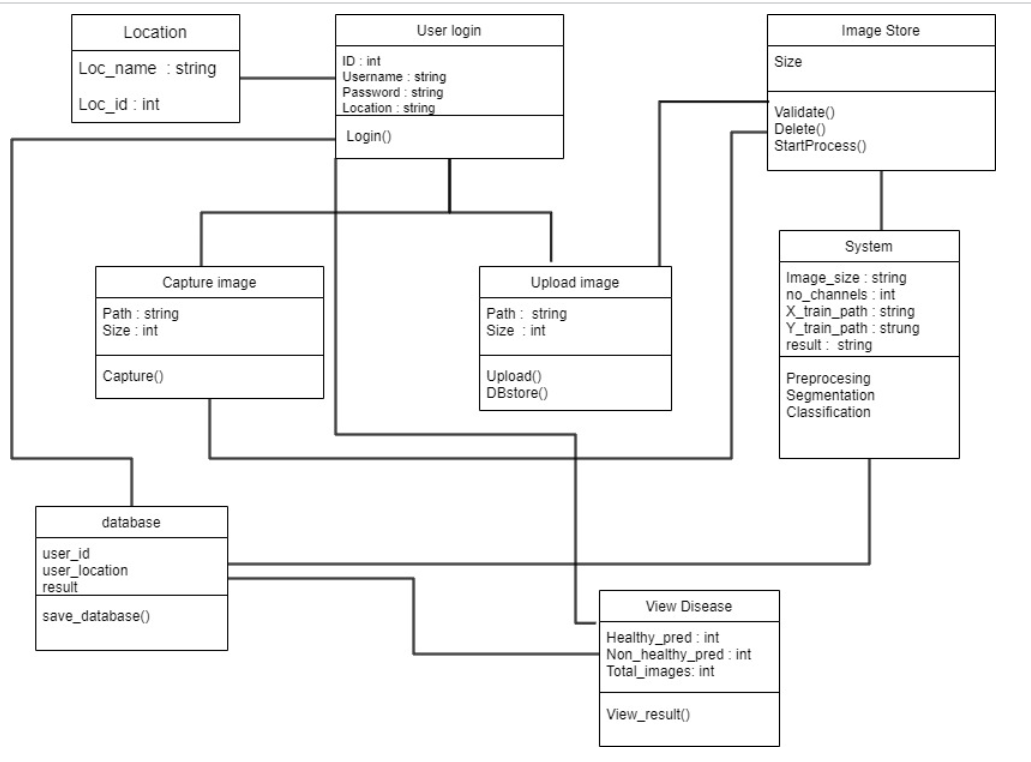


Figure 5.6 Class Diagram

### Description

In Figure 5.6, we present a detailed class diagram that encapsulates the essential components and interactions within our Mango Disease Detector system. The class diagram provides a comprehensive overview of the system's architecture and showcases how various classes interact to facilitate seamless disease detection and analysis.

The primary user interactions with the system are represented by the "User" class. Users can log in to the system by providing their credentials, including their name, location, password, and ID. Upon successful login, users gain access to the system's functionalities, enabling them to either upload images from their local system or capture images using their device's camera.

The "Image" class is responsible for handling the uploaded or captured images. It acts as a container that holds the image data and relevant attributes. When the user uploads an image, the system employs a validation function within the "Image" class to ensure the image's size meets the required criteria. This validation step helps in preventing errors and maintaining data integrity during the disease detection process.

Once the image is validated, users have the option to perform two essential functions. Firstly, they can choose to delete the uploaded image if they find it unsuitable or redundant. This functionality is represented by the "deleteImage()" method within the "Image" class. Secondly, users can proceed with image processing, wherein the "processImage()" method is called. This marks the initiation of the disease detection workflow, and the image is passed on to the AI/ML model or system responsible for further analysis.

The core of the disease detection process lies within the "AI/ML Model" class. This class incorporates various image processing techniques, image segmentation, and image classification algorithms. Through these methods, the AI/ML model effectively extracts meaningful features from the input image, identifies regions of interest, and classifies the image into different disease categories. The "AI/ML Model" class serves as the engine that powers accurate and reliable mango disease detection.

The "Image Preprocessing" class complements the AI/ML model by providing a set of techniques to optimize the input image's quality and suitability for analysis. These techniques may include resizing, filtering, color spacing normalization, grey scaling, and histogram equalization. By preprocessing the image, the system enhances its visual characteristics, enabling the AI/ML model to make more informed and accurate disease classification decisions.

The "Image Segmentation" class focuses on dividing the preprocessed image into meaningful regions or segments. This process is instrumental in identifying specific regions that exhibit signs of disease infection, thereby aiding in precise disease localization and assessment. The "Image Segmentation" class plays a pivotal role in enhancing the disease detection process's granularity and sensitivity.

Finally, the "Image Classification" class performs the critical task of categorizing the segmented regions into distinct disease classes. Leveraging powerful machine learning algorithms, the system identifies the disease type afflicting the mango crop in each segmented region. The classification results are then integrated to provide a comprehensive diagnosis of the mango crop's health.

In conclusion, the class diagram depicted in Figure 5.6 offers an insightful depiction of the Mango Disease Detector system's architecture. It showcases the interactions between the "User" class and the image-related classes, such as "Image," "AI/ML Model," "Image Preprocessing," "Image Segmentation," and "Image Classification." By employing these classes in a cohesive manner, the system ensures accurate, efficient, and user-friendly disease detection for mango crops. The integration of AI/ML algorithms, image preprocessing, and segmentation techniques empowers the system to provide valuable insights into the health of mango crops, enabling farmers and agricultural experts to make informed decisions and take proactive measures for disease management. The class diagram serves as a blueprint for the system's development and highlights the seamless interplay of its components to create a robust and effective Mango Disease Detector.

## Use Case

### User

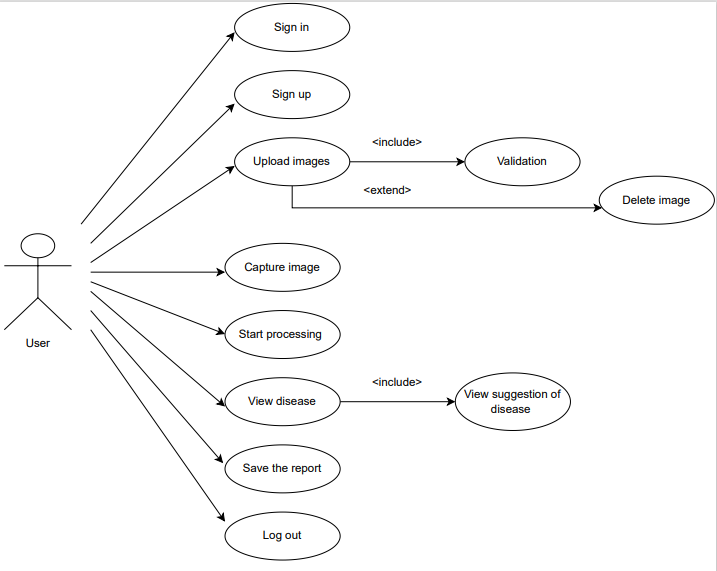


Figure 5.7 Use Case Diagram of User

#### Description

In Figure 5.7, we present the user activity diagram that showcases the various interactions and functionalities available to the user within our Mango Disease Detector system. The user plays a central role in the system's workflow, and the activity diagram outlines the sequence of actions they can perform to effectively detect and manage diseases in mango crops.

The user activity begins with two fundamental actions: sign up and sign in. New users have the option to sign up for the system by providing essential details such as their name, location, preferred password, and unique ID. This information is then stored securely in the system's database, creating a personalized user profile for future interactions. Existing users can sign in using their credentials to gain access to the system's features and disease detection capabilities.

Once successfully logged in, the user is presented with two primary options for disease detection: uploading an image from their local system or capturing an image using their device's camera. The flexibility to choose between these two methods allows users to work with images they have on hand or capture real-time images of mango crops for analysis.

When the user uploads an image, a validation process is initiated to ensure the image's size and format meet the system's requirements. If the image fails to pass the validation criteria, the system prompts the user to review the image and make any necessary adjustments. This validation step helps maintain data integrity and ensures that only appropriate images are processed for disease detection.

Upon successful validation, the user has the option to delete the uploaded image if they find it unsuitable for analysis or if they have uploaded multiple images and want to refine their selection. The "Delete Image" action provides users with control over their input, allowing them to make informed decisions during the disease detection process.

Once the user initiates image processing, the system employs advanced AI/ML algorithms and image processing techniques, as previously discussed in the class diagram, to analyze the image thoroughly. The system extracts meaningful features from the image, segments it into relevant regions, and classifies these regions into different disease categories. The results of the disease detection process are then displayed to the user, showcasing the identified diseases affecting the mango crop.

In addition to disease identification, the system also provides valuable information on disease precautions and management strategies. The user can access these precautionary measures for each detected disease, enabling them to take timely and appropriate actions to safeguard their mango crops from further damage.

To preserve the results and recommendations generated by the system, the user has the option to save the disease detection report. This report can be downloaded and stored for future reference, serving as a valuable resource for ongoing crop management and disease monitoring.

Finally, once the user has completed the disease detection process and saved the report, they can choose to log out of the system. The logout action ensures the user's privacy and security, preventing unauthorized access to their account and preserving the confidentiality of their data.

In conclusion, the user activity diagram depicted in Figure 5.7 outlines a seamless and user-centric disease detection process within our Mango Disease Detector system. The user is empowered to sign up or sign in to the system, upload or capture images of mango crops, and engage in disease detection with ease. The validation and deletion options provide users with control and flexibility during the image selection process. The system's AI/ML algorithms and image processing techniques work cohesively to generate accurate disease identification and precautionary measures, which users can access and save for future use. The ability to log out ensures user privacy and enhances the overall user experience. By facilitating efficient user interactions and delivering valuable disease detection insights, our Mango Disease Detector serves as a valuable tool for farmers and agricultural experts to enhance crop management practices and ensure the health and productivity of mango crops.

### System



Figure 5.8 Use Case Diagram of System

#### Description

In Figure 5.8, we are presented with a detailed illustration of the inner workings of our sophisticated Mango Disease Detector system. This diagram provides valuable insights into the systematic process flow, starting from image preprocessing to feature extraction and classification, all of which come together harmoniously to enable accurate and efficient mango disease detection.

The foundation of our system lies in image preprocessing, a crucial initial step that significantly impacts the quality of the subsequent analysis. As mango crop images are acquired through image upload or captured using the device's camera, they go through a series of preprocessing techniques to enhance their quality and prepare them for further analysis. These image preprocessing techniques include color spacing, resizing, and filtering.

Color spacing is an essential preprocessing technique that ensures all acquired images are standardized to a consistent color format. This standardization is vital for achieving consistent and reliable results during the disease detection process. By converting the images to a common color space, such as RGB or grayscale, the system can effectively handle images of varying formats, facilitating more accurate and streamlined analysis.

Resizing the images to a specific dimension is another key preprocessing step. This step is crucial for several reasons. Firstly, it optimizes computational resources by ensuring that all images have a consistent size, allowing for more efficient processing. Additionally, resizing ensures that images of different resolutions are transformed into a unified scale, making it easier for the system to extract meaningful features and patterns from the images.

Filtering techniques are also applied during image preprocessing to remove noise or unwanted elements from the images. Noise, such as artifacts or irrelevant information, can interfere with the accuracy of the disease detection process. By employing filtering techniques, the system can enhance the clarity and relevance of the images, focusing on the critical aspects that contribute to disease identification.

Once the images are preprocessed, the system advances to the feature extraction phase, a critical step in analyzing and understanding the content of the segmented regions. Feature extraction involves identifying relevant patterns, edges, and textures within the segmented regions, transforming the raw data into a more concise representation of the image content. These distinctive features serve as the foundation for disease identification and classification.

In the feature extraction phase, the system employs the Convolutional Neural Network (CNN) algorithm, a cutting-edge deep learning technique renowned for its unparalleled ability to automatically learn relevant features directly from images. CNNs are designed to mimic the human brain's visual processing, enabling them to recognize intricate patterns and relationships within images.

Through a series of convolutional layers, the CNN algorithm effectively captures complex and hierarchical features from the segmented regions. These learned features encode critical information about the presence of diseases, making the subsequent classification process more accurate and reliable. The feature extraction process acts as a powerful abstraction layer, enabling the system to distill the essence of the image content and focus solely on the relevant disease-related features.

The CNN's ability to learn and adapt to diverse image patterns and variations makes it an ideal choice for disease classification tasks. Its sophisticated architecture allows for robust and accurate mango disease detection, contributing to effective crop management and timely interventions.

In the classification phase, the extracted features are fed into fully connected layers, where the final classification decision is made. The system maps the learned features to specific disease classes, associating each segmented region with a corresponding disease category. The softmax activation function calculates the probabilities of the image belonging to different disease classes, and the class with the highest probability is identified as the detected mango disease.

The successful application of the CNN algorithm in mango disease detection highlights the potential of deep learning and artificial intelligence in transforming agricultural practices. By automating the detection process, our Mango Disease Detector offers a revolutionary approach to crop health management. Its accuracy and efficiency empower farmers and agricultural experts to make timely and informed decisions, ensuring the early identification and containment of diseases.

Moreover, the integration of image preprocessing techniques further enhances the system's robustness and adaptability to varying image conditions. Standardizing images through color spacing and resizing mitigates potential challenges arising from image disparities, guaranteeing consistent and reliable analysis outcomes.

The capability of our system to accurately segment regions of interest using the U-Net segmentation algorithm adds an extra layer of precision to disease detection. By isolating areas affected by diseases, the system can effectively focus on crucial regions and reduce false positives, contributing to a higher level of confidence in the diagnosis.

In conclusion, Figure 5.8 provides a comprehensive and detailed view of the inner workings of our Mango Disease Detector system. The seamless integration of image preprocessing, segmentation using U-Net, feature extraction through CNN, and disease classification underscores the power of AI-driven solutions in revolutionizing agricultural practices and ensuring sustainable crop production. Our Mango Disease Detector serves as a testament to the potential of advanced technologies in transforming the agricultural landscape and empowering farmers with cutting-edge tools for crop health management. By automating the detection of mango diseases, our system facilitates timely interventions, minimizes crop losses, and ultimately contributes to the enhancement of crop yield and overall agricultural productivity. As we continue to advance the field of agricultural technology, our Mango Disease Detector stands as a beacon of innovation and progress, striving to support farmers and safeguard crop health for a more sustainable future.

### View Disease

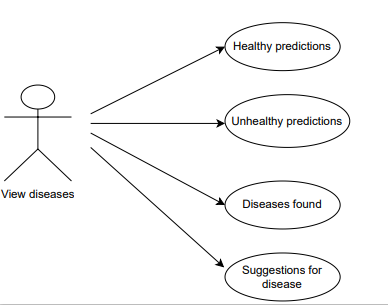


Figure 5.9 Use case of View Disease

#### Description

In Figure 5.9, we delve into the intricacies of the "View Disease" functionality, a critical component of our Mango Disease Detector system that empowers users to gain valuable insights into disease predictions and receive essential suggestions for managing crop health. This section provides users with a comprehensive overview of the disease detection results, including the number of healthy and unhealthy predictions and the specific diseases identified in the analyzed images. Additionally, it offers vital recommendations for addressing the detected diseases, enhancing the user's ability to make informed decisions and take timely action.

Upon accessing the "View Disease" section, users are presented with an intuitive and user-friendly interface that provides a summary of the disease detection outcomes. The first key element displayed is the number of healthy predictions, reflecting the count of images that were classified as disease-free. This information serves as a reassuring indicator of the overall crop health and provides users with an immediate assessment of the well-being of their mango crops.

The next vital piece of information offered is the number of unhealthy predictions, which represents the count of images exhibiting signs of diseases. This valuable insight highlights potential areas of concern in the crop, alerting users to the presence of diseases that require immediate attention. By quantifying the unhealthy predictions, users gain a quantitative understanding of the disease prevalence, enabling them to prioritize their response and allocate resources efficiently.

The core of the "View Disease" section lies in the detailed disease analysis, where the system showcases the specific diseases detected in the images. Each identified disease is presented along with its corresponding image(s), enabling users to visually verify the accuracy of the predictions. This visual representation fosters transparency and trust in the system's disease detection capabilities, bolstering user confidence in the results.

In addition to disease identification, the system provides comprehensive suggestions for managing and mitigating the detected diseases. These suggestions are tailored to each specific disease, taking into account its unique characteristics and recommended best practices for treatment and prevention. By providing customized guidance, our Mango Disease Detector empowers users with actionable steps to address the identified diseases effectively.

The disease suggestions draw from a vast repository of agricultural knowledge, incorporating the expertise of agricultural specialists and researchers. This wealth of information is curated and distilled into concise and actionable recommendations, making it easily accessible to users of varying levels of expertise. By integrating expert insights into the system, we ensure that users receive reliable and evidence-based guidance for disease management.

The "View Disease" section is designed to be interactive and dynamic, enabling users to explore the disease detection results comprehensively. Users can select individual disease entries to access more detailed information, including the symptoms, causes, and recommended treatments for each specific disease. This level of granularity empowers users to gain a deeper understanding of the identified diseases and the factors influencing their development.

Furthermore, the "View Disease" section offers the option to generate and save a comprehensive disease report. This report consolidates all disease detection results, suggestions, and additional information into a downloadable format, facilitating seamless data sharing and record-keeping. Users can use the report to track disease prevalence over time, assess the effectiveness of interventions, and make informed decisions for future crop management strategies.

By utilizing the power of data visualization, our Mango Disease Detector presents disease detection outcomes in an intuitive and accessible manner. Through charts, graphs, and infographics, users can gain valuable insights at a glance, enabling them to quickly identify disease patterns, trends, and areas requiring immediate attention. The visual representation of data simplifies complex information, making it easier for users to interpret and act upon the results.

In conclusion, Figure 5.9 showcases the remarkable capabilities of the "View Disease" functionality in our Mango Disease Detector system. By providing users with detailed disease detection results, suggestions for disease management, and a user-friendly interface, our system equips farmers and agricultural experts with the tools they need to proactively manage crop health and enhance agricultural productivity. The transparency and accuracy of the disease detection process, coupled with evidence-based recommendations, contribute to the system's reliability and effectiveness in supporting sustainable crop production. As we continue to refine and enhance our Mango Disease Detector, we remain committed to advancing agricultural technology and empowering farmers with innovative solutions for crop health management. With the "View Disease" functionality at their fingertips, users can navigate the challenges of disease detection and crop management with confidence, ensuring the health and prosperity of their mango crops for a more sustainable and food-secure future.

## Deployment Diagram

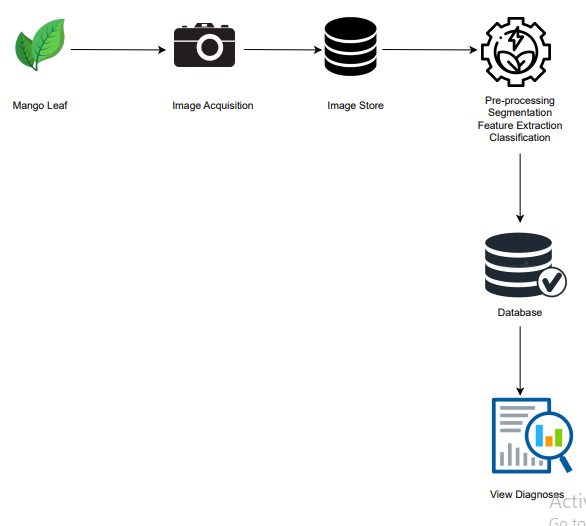


Figure 5.10 Deployment Diagram

### Description

The figure 5.10 illustrates the deployment diagram that provides a holistic view of the Mango Disease Detector system's architecture, illustrating the various components and how they interact to deliver a seamless disease detection and management experience. At the heart of this system lies the mango leaf image acquisition component, which serves as the primary interface between the user and the system. Users can upload images of mango leaves from their local system or capture images using the device's camera, initiating the disease detection process.

Once the images are acquired, they are passed to the image store component, which efficiently manages and stores the vast collection of mango leaf images. This component ensures that the images are organized and readily accessible for further processing and analysis. The image store is designed to handle large volumes of data, enabling the system to accommodate an extensive database of images from diverse sources.

The preprocess component plays a critical role in enhancing the quality and suitability of the acquired images for disease detection. It encompasses a range of image preprocessing techniques, including color spacing, resizing, and filtering. These techniques optimize the images for analysis by standardizing their format and reducing noise, ensuring consistent and reliable results during disease detection.

The database component acts as the central repository for storing critical information related to disease detection results, user profiles, and historical data. It efficiently manages data storage, retrieval, and manipulation, allowing the system to handle vast amounts of information generated from disease detection activities. The database serves as a valuable resource for users, enabling them to access historical disease data, track trends, and make data-driven decisions for crop management.

The view component serves as the user interface, providing users with a visually engaging and user-friendly platform to interact with the system. Users can view the results of disease detection, access disease reports, and explore disease diagnoses through intuitive visualizations and interactive elements. The view component is designed to be responsive, ensuring a seamless user experience across various devices, including desktops, tablets, and mobile phones.

The diagnoses component is the heart of the Mango Disease Detector system, encompassing the disease detection algorithms and machine learning models. It leverages advanced techniques such as Convolutional Neural Networks (CNN) to perform feature extraction and image classification. The diagnoses component analyzes the preprocessed images, identifies disease patterns, and generates disease predictions with high accuracy.

The deployment diagram showcases the efficient distribution of these components across multiple nodes, representing physical devices or servers. This distribution ensures optimal performance, scalability, and reliability of the Mango Disease Detector system. For instance, the image acquisition and view components may be deployed on user devices, allowing for real-time image uploads and immediate access to disease detection results.

The image store, preprocess, diagnoses, and database components may be deployed on dedicated servers or cloud infrastructure, offering the computational power and storage capacity needed to handle large-scale image processing and data management tasks. This scalable architecture enables the system to accommodate increasing data volumes and user traffic, making it suitable for deployment in diverse agricultural settings.

Furthermore, the deployment diagram demonstrates the communication paths and protocols utilized by the system's components. These communication paths ensure seamless data flow between the user interface, image processing, and database components. For example, the image acquisition component sends the acquired images to the preprocess component through secure communication channels, safeguarding user data and ensuring data integrity.

In conclusion, the deployment diagram of the Mango Disease Detector system provides a comprehensive overview of the system's architecture and interactions between its components. From image acquisition to disease detection and data management, each component plays a vital role in delivering accurate disease diagnoses and valuable insights for crop health management. The distribution of components across multiple nodes and the use of robust communication protocols ensure a reliable and scalable system suitable for use in diverse agricultural environments.

As agricultural technology continues to advance, the Mango Disease Detector system remains at the forefront, empowering farmers and agricultural experts with cutting-edge tools for disease detection and crop management. With its user-friendly interface, powerful disease detection algorithms, and efficient data management capabilities, the system contributes to enhanced crop productivity and sustainability. As we continue to enhance and refine the system, we remain committed to providing users with innovative solutions that promote a more food-secure future and support the global agricultural community in its efforts to address the challenges of crop diseases.

## `Operational Diagram

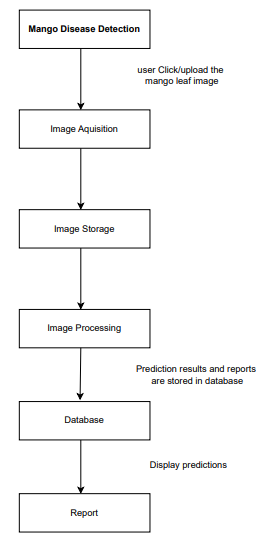


Figure 5.11 Operational Diagram

### Description

The operational diagram figure 5.11 offers a comprehensive view of the Mango Disease Detector system's day-to-day operations, illustrating how its key components interact to provide an efficient and effective disease detection process for mango leaves. At the core of this operational diagram lies the mango leaf image acquisition component, which serves as the gateway for users to input images into the system. Users have the option to either upload images from their local devices or capture images using the device's camera. This user-friendly interface ensures that farmers and agricultural experts can easily provide input to initiate the disease detection process.

Once the images are acquired, they are directed to the image store component. This vital part of the system handles the management and storage of the acquired images. The image store ensures that the images are appropriately organized and stored, facilitating easy retrieval and access for further processing. This seamless handling of the vast collection of mango leaf images is crucial for the overall efficiency and performance of the Mango Disease Detector system.

The preprocess component plays a critical role in optimizing the acquired images for disease detection. It employs various image preprocessing techniques to enhance the image quality and suitability for further analysis. Techniques such as color spacing, resizing, and filtering are applied to standardize the images and remove any noise or distortions that may affect the accuracy of disease detection. This preprocessing step is essential in ensuring consistent and reliable results during disease diagnosis.

The database component serves as the central repository for storing and managing critical information related to disease detection results and user data. It efficiently stores disease detection reports, user profiles, and historical data. The database enables users to access disease reports and track disease trends over time, providing valuable insights for crop management decisions. This secure and robust data management is instrumental in supporting evidence-based practices for agricultural experts and facilitating seamless communication between different components of the system.

The report component is a key feature that provides users with detailed disease detection reports. Once the disease detection process is completed, the system generates comprehensive reports that outline the detected diseases and their respective characteristics. The reports also include suggested measures and treatments for managing the identified diseases effectively. This information empowers users, allowing them to take prompt and informed actions to mitigate the impact of diseases on their mango crops.

The operational diagram demonstrates the flow of activities within the Mango Disease Detector system. It begins with image acquisition, where users input images into the system. The acquired images then undergo preprocessing to optimize their quality for disease detection. The preprocessed images are then passed through the disease detection algorithm, where advanced machine learning techniques, such as Convolutional Neural Networks (CNN), are employed for feature extraction and classification. The results of the disease detection process are stored in the database and used to generate detailed reports for user access.

Throughout the operational diagram, emphasis is placed on user-friendliness, efficiency, and accuracy. The system is designed to cater to users with varying levels of technical expertise, ensuring that farmers and agricultural experts can easily navigate the platform. Moreover, the integration of advanced image processing and machine learning algorithms guarantees precise and reliable disease detection results.

Overall, the operational diagram exemplifies the seamless interaction between the Mango Disease Detector system's components to deliver an integrated and effective disease detection solution for mango leaves. Through continuous monitoring, updates, and user feedback, the system strives to remain at the forefront of agricultural technology, empowering farmers and agricultural experts with innovative tools for disease management and improved crop productivity..

# Algorithm analysis and complexity

This chapter comprises the algorithms used in this project, their significance along with the pseudo code.

## CNN Algorithm

The algorithm used for the classification and detection of diseases in the "Mango Disease Detector" project is a Convolutional Neural Network (CNN) . CNNs are a type of Artificial Neural Network (ANN) specifically designed for processing and analyzing images [9]. This powerful deep learning algorithm has shown remarkable success in various computer vision tasks, including image classification, object detection, and segmentation [9].

The CNN architecture is inspired by the human visual system, where the brain processes visual information in a hierarchical manner, recognizing simple shapes and patterns before gradually building up to more complex objects and features [9]. Similarly, CNNs employ a series of interconnected layers, each responsible for different aspects of image analysis. These layers extract progressively higher-level features from the input images, enabling accurate and robust classification [9].

The first layer in a CNN is typically a convolutional layer. In this layer, a set of learnable filters, also known as kernels, convolve across the input image, performing element-wise multiplications and generating feature maps. These feature maps highlight various patterns and edges present in the image [9]. The process of convolution captures spatial relationships and local features, which are essential for understanding the image's content.

Following the convolutional layer, the CNN usually incorporates activation functions, such as ReLU (Rectified Linear Unit), to introduce non-linearity into the model [9]. Non-linearity is crucial as many real-world data patterns are not linearly separable, and activation functions allow the model to capture complex relationships within the data.

The next layer in the CNN is the pooling layer, which reduces the spatial dimensions of the feature maps while retaining the most relevant information [9]. Pooling operations, such as max-pooling, select the maximum value from a small region of the feature map and discard the rest. This downsampling process reduces the computational complexity of the model while preserving important features [9].

After several convolutional and pooling layers, the CNN includes one or more fully connected layers. These layers take the high-level features extracted from the previous layers and learn complex combinations of these features to make final predictions [9]. Fully connected layers integrate all the information extracted during the earlier stages of the CNN and perform classification based on the learned representations.

In the project, the sequential model of the CNN is designed using the TensorFlow library [19]. TensorFlow is an open-source deep learning framework that provides a flexible and efficient environment for building and training neural networks [19]. The sequential model in TensorFlow allows for the orderly arrangement of layers, creating a straightforward and easy-to-understand architecture [19].

By leveraging TensorFlow's capabilities, the CNN architecture for the "Mango Disease Detector" project is tailored to handle mango leaf images as input and output a prediction of whether the leaves are healthy or affected by a disease . During the training phase, the model learns to assign different weights and labels to various objects and aspects present in the input images, enabling it to make accurate disease classifications [19].

The training process involves optimizing the model's parameters to minimize the difference between predicted outputs and actual labels in a labeled dataset. This optimization is achieved through a process called backpropagation, where the model adjusts its weights and biases based on the calculated errors [18].

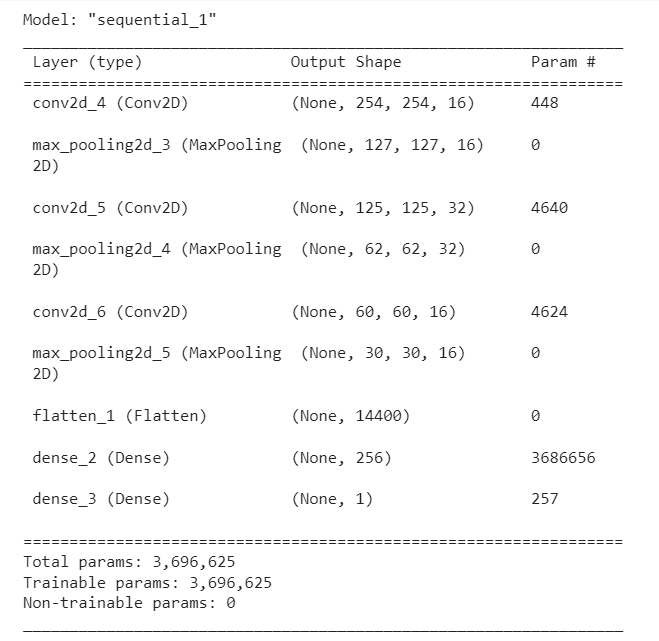
Once the CNN is fully trained, it can be deployed in the "Mango Disease Detector" application to analyze uploaded or captured mango leaf images . The sequential model takes these images as input, processes them through the convolutional layers, applies activation functions, performs pooling operations, and finally passes the high-level features to the fully connected layers for classification .

### Pseudocode

#### Building the Architecture of CNN Model

* From the tensor flow library import the Sequential, Conv2D, MaxPooling2D, Dense, Flatten, Dropout
* Model = Sequential ()
* Model.add (Conv2D (size of filter, (size matrix of kernel), activation function))
* Model.add (MaxPooling2D ())
* Model.add (Flatten ())
* Model.add (Dense (output\_channels \* ( input\_channels + 1)), activation function)
* Model.add (Dense (output\_channels \* ( input\_channels + 1)), activation function)

#### 6.1.1.2 Model Summary



#### Training of CNN Model

* Importing the Libraries
* Import Numpy, tensor flow, matplotlib
* Load the images of mango leaves into the model and preprocessed the images
* Feed the pre processed images into the model
* Perform Backpropagation by using adam optimizer
* Repeat this process for specified number of epochs.

* data = tf.keras.utils.image\_dataset\_from\_directory(“path”)
* data= data.map(lambda x,y:(x/255,y))
* model.compile('adam',loss,metrics = ['accuracy'])
* model.fit(train,epochs)

#### Evaluating the CNN Model

* Feed the unseen picture of mangoes in the model.
* Predict the model if it the leave is affected or not
* Evaluate the model by using metric accuracy

## Cases

### Best Case

In the best-case scenario, the time complexity of a CNN-based system for mango disease diagnosis is around O(n), where n is the size of the input mango image. Simple convolutional procedures and layers are used when dealing with low-complexity images or concentrating on fundamental features for disease identification. However, this is an idealized situation.

### Average Case

The average-case time complexity is O(n2) to O(n3). The depth of the network, the size of the filters, and the difficulty of feature extraction required for recognizing disease patterns in mango photos all have an impact on this range. The specific complexity relies on variables like network architecture and the amount of picture analysis needed for precise disease identification, while the presence of intermediate convolutional and pooling layers does lead to a quadratic or cubic expansion in computation relative to input size.

### Worst Case

The worst-case time complexity could be O(n4), especially if the network is deep, the filters are large, and the feature extraction is complicated. When processing huge and detailed mango photos for disease identification, a considerable computing cost may result from repetitive convolutions and non-linear operations across numerous layers.

## Chapter Conclusion

The "Mango Disease Detector" project leverages Convolutional Neural Network (CNN) technology as the core algorithm for disease classification and detection . CNNs, which are a type of Artificial Neural Network (ANN) specially designed for image processing and analysis, have demonstrated their effectiveness in various computer vision tasks, including image classification, object detection, and segmentation.

The architecture of the CNN in this project follows the hierarchical processing of visual information similar to the human brain. It comprises interconnected layers with specific roles in image analysis . The initial convolutional layers extract low-level features like edges and patterns from the input mango leaf images, while subsequent activation functions introduce non-linearity to capture complex data relationships.

Pooling layers come after the convolutional layers, reducing spatial dimensions while preserving crucial information. This downsampling optimizes computational efficiency without compromising the features necessary for accurate disease classification. Fully connected layers integrate high-level features to make final disease predictions based on learned representations .

The CNN model is implemented using the TensorFlow library in Python, which provides a flexible and efficient deep learning framework [19]. The sequential model arrangement simplifies layer organization, resulting in an intuitive and efficient architecture for disease classification.

Training the CNN involves using a labeled dataset and optimizing parameters through backpropagation to minimize prediction errors. Weights and biases are adjusted iteratively based on errors between predicted outputs and actual labels .

Once trained, the CNN is evaluated using unseen mango leaf images to predict whether the leaves are healthy or diseased, with performance assessed using metrics like accuracy.

In conclusion, the "Mango Disease Detector" project harnesses the capabilities of Convolutional Neural Network (CNN) technology to facilitate precise disease classification and detection in mango trees . This application, through its effective CNN architecture and training on labeled data, supports farmers in early disease identification, enabling timely intervention and disease management . By seamlessly integrating CNN technology with user-friendly interfaces, the "Mango Disease Detector" contributes to enhanced mango productivity, improved fruit quality, and sustainable agricultural practices. This project exemplifies the value of employing cutting-edge deep learning techniques to address real-world challenges, empowering farmers and bolstering the nation's economy .

# Implementation

## CNN Model

import pandas as pd

import os

import cv2

import numpy as np

import tensorflow as tf

from PIL import Image, ImageEnhance

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, Activation,GlobalAveragePooling2D

from tensorflow.keras.models import Sequential

from keras.models import Model

from keras.applications.inception\_v3 import InceptionV3,preprocess\_input

from tensorflow.keras.utils import load\_img

from tensorflow.keras.utils import  img\_to\_array

from sklearn.model\_selection import train\_test\_split

from tensorflow.keras import layers, models

from pathlib import Path

import matplotlib.pyplot as plt

import seaborn as sns

from PIL import Image

import random

from sklearn.metrics import classification\_report,confusion\_matrix

import itertools

from zipfile import ZipFile

file\_name = "/content/drive/MyDrive/External Images.zip"

with ZipFile(file\_name, 'r') as zip:

  zip.extractall()

  print('Done')

from zipfile import ZipFile

file\_name = "/content/drive/MyDrive/CS-06 MANGO DISEASE DETECTOR DATASET.zip"

with ZipFile(file\_name, 'r') as zip:

  zip.extractall()

  print('Done')

folder\_names = ["Anthracnose"  , "Die Black" , "Gall Midge" , "Healthy",

             "Powdery Mildew"  , "Sooty Mould"]

data\_dir = Path("/content/CS-06 MANGO DISEASE DETECTOR DATASET")

def filter\_subdirs(subdirs):

    return [subdir for subdir in subdirs if Path(subdir).name in folder\_names]

def count\_images\_per\_class(folder\_names, data\_dir):

    class\_counts = {}

    for folder in folder\_names:

        class\_path = data\_dir / folder

        num\_images = len(list(class\_path.glob('\*.jpg')))  # assuming the images are in jpg format

        class\_counts[folder] = num\_images

    return class\_counts

class\_counts = count\_images\_per\_class(folder\_names, data\_dir)

class\_counts\_df = pd.DataFrame.from\_dict(class\_counts, orient='index', columns=['count'])

plt.figure(figsize=(12, 6))

sns.barplot(data=class\_counts\_df.reset\_index(), x='index', y='count')

plt.xlabel('Class')

plt.ylabel('Number of Images')

plt.title('Distribution of Images Across Classes')

plt.xticks(rotation=45)

plt.show()

A graph of different colored rectangular shapes

Description automatically generated

def display\_sample\_images(folder\_names, data\_dir, num\_samples=3):

    fig, axes = plt.subplots(len(folder\_names), num\_samples, figsize=(12, 2 \* len(folder\_names)))

    for i, folder in enumerate(folder\_names):

        class\_path = data\_dir / folder

        image\_files = list(class\_path.glob('\*.jpg'))

        samples = random.sample(image\_files, num\_samples)

        for j, image\_file in enumerate(samples):

            img = Image.open(image\_file)

            axes[i, j].imshow(img)

            axes[i, j].set\_axis\_off()

            if j == 0:

                axes[i, j].set\_title(folder)

    plt.subplots\_adjust(wspace=0.05, hspace=0.25)

    plt.show()

display\_sample\_images(folder\_names, data\_dir)



from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Define the paths to your train, validation, and test folders

train\_dir = '/content/CS-06 MANGO DISEASE DETECTOR DATASET'

val\_dir = '/content/CS-06 MANGO DISEASE DETECTOR DATASET'

test\_dir = '/content/External Images'

# Define the image size and batch size

image\_size = (224, 224)

batch\_size = 32

# Data Augmentation and Preprocessing

data\_generator = ImageDataGenerator(

    rescale=1./255,

    rotation\_range=20,

    width\_shift\_range=0.2,

    height\_shift\_range=0.2,

    shear\_range=0.2,

    zoom\_range=0.2,

    horizontal\_flip=True

)

# Generating train dataset

train\_generator = data\_generator.flow\_from\_directory(

    train\_dir,

    target\_size=image\_size,

    batch\_size=batch\_size,

    class\_mode='categorical'

)

# Generating validation dataset

val\_generator = data\_generator.flow\_from\_directory(

    val\_dir,

    target\_size=image\_size,

    batch\_size=batch\_size,

    class\_mode='categorical'

)

# Generating test dataset (without augmentation)

test\_generator = data\_generator.flow\_from\_directory(

    test\_dir,

    target\_size=image\_size,

    batch\_size=batch\_size,

    class\_mode='categorical',

    shuffle=False

)

#Number of classes in your dataset

num\_classes = 6

# Load the pre-trained InceptionV3 model without the top (fully connected) layers

base\_model = InceptionV3(weights='imagenet', include\_top=False, input\_shape=(224, 224, 3))

# Add your own top layers for classification

x = base\_model.output

x = GlobalAveragePooling2D()(x)

x = Dense(1024, activation='relu')(x)

predictions = Dense(num\_classes, activation='softmax')(x)

# Create the final model

model = Model(inputs=base\_model.input, outputs=predictions)

# Freeze the base model layers

for layer in base\_model.layers:

    layer.trainable = False

# Compile the model

model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])

from keras.callbacks import ModelCheckpoint, EarlyStopping

mc = ModelCheckpoint(filepath = "./FYP.h5",

                     monitor = "accuracy",

                     verbose = 1,

                     save\_best\_only = True)

es= EarlyStopping(monitor = "accuracy",

                  min\_delta = 0.01,

                  patience=5,

                  verbose =1)

cb = [mc,es]

model.fit(

    train\_generator,

    steps\_per\_epoch=train\_generator.n // batch\_size,

    epochs=10,

    validation\_data=val\_generator,

    validation\_steps=val\_generator.n // batch\_size,

    callbacks = cb

)

test\_loss, test\_accuracy = model.evaluate(test\_generator)

print("Test Loss:", test\_loss)

print("Test Accuracy:", test\_accuracy)



predicted\_labels = model.predict(test\_generator)

y\_pred = np.argmax(predicted\_labels, axis=1)

y\_true = test\_generator.classes

correct\_predictions = np.sum(y\_true == y\_pred)

total\_predictions = len(y\_true)

accuracy = (correct\_predictions / total\_predictions) \* 100

print("Accuracy: ", accuracy, "%")



# Calculate classification report

report = classification\_report(y\_true, y\_pred)

# Print the classification report

print(report)

A screenshot of a graph

Description automatically generated

import seaborn as sns

import matplotlib.pyplot as plt

g\_dict = test\_generator.class\_indices

classes = list(g\_dict.keys())

# Confusion matrix

cm = confusion\_matrix(test\_generator.classes, y\_pred)

plt.figure(figsize=(10, 10))

sns.heatmap(cm, annot=True, fmt='d', cmap='Greens', cbar=False)

plt.title('Confusion Matrix')

plt.xlabel('Predicted Label')

plt.ylabel('True Label')

plt.xticks(ticks=np.arange(len(classes))+0.5, labels=classes)

plt.yticks(ticks=np.arange(len(classes))+0.5, labels=classes)

plt.show()

A green squares with white text

Description automatically generated

# Testing

Testing is a crucial and systematic process in software development that aims to evaluate and assess the quality, functionality, and performance of a software application. It involves executing the software with the intent of identifying defects, bugs, or errors, and ensuring that it meets the specified requirements and expectations. The primary objective of testing is to deliver a reliable and robust product that satisfies user needs and operates efficiently in different scenarios. Software testing encompasses various levels and methodologies, each serving a specific purpose in the development lifecycle. At the unit level, developers perform unit testing, which involves testing individual components or modules to verify their correctness and functionality in isolation. Integration testing, on the other hand, evaluates the interaction between integrated modules to identify any issues arising from their combination. System testing evaluates the entire software system to ensure that all components work together seamlessly, meet the functional and non-functional requirements, and comply with the desired behavior. User acceptance testing (UAT) allows end-users to validate the software's usability and suitability in a real-world environment before its final deployment. There are different testing approaches, such as black-box testing, which focuses on testing the software's external behavior without considering its internal implementation, and white-box testing, which involves examining the internal structure and logic of the software code to identify defects. Additionally, automated testing streamlines the testing process by using scripts and tools to execute test cases efficiently and repeatedly. Testing plays a vital role in software quality assurance by ensuring that defects are detected early in the development cycle, reducing the cost and effort required for fixing issues later. It enhances the software's reliability, security, and performance, thereby increasing user satisfaction and confidence in the product. Testing is an iterative process, and feedback from testing is used to refine and improve the software until it meets the desired level of quality and functionality. Ultimately, effective testing is essential for delivering software products that meet user needs, adhere to industry standards, and provide a seamless user experience.

## Black Box Testing

Black-box testing is a software testing methodology that examines the functionality of an application without delving into its internal code or structure. Testers treat the software as a "black box," focusing solely on its inputs, outputs, and behavior. This approach aims to validate whether the software functions as intended from the user's perspective, without any knowledge of its internal workings. Testers design test cases based on the software's specifications, requirements, and user interfaces, simulating real-world scenarios and user interactions. By doing so, they can identify issues related to usability, functionality, and user experience. Black-box testing promotes independence between testers and developers, as testers do not require access to the source code or programming knowledge. Instead, they rely on the software's documentation and external interfaces to design comprehensive test cases covering various scenarios and edge cases. Techniques such as equivalence partitioning, boundary value analysis, error guessing, and state transition testing are commonly used in black-box testing to ensure test coverage. Despite its advantages in early defect detection and objectivity, black-box testing may have limitations in test coverage and the ability to reproduce defects without knowledge of the code. Nonetheless, it remains a crucial testing approach to ensure that software meets user expectations and delivers a high-quality user experience. Its systematic and user-centric nature complements other testing methodologies, providing valuable insights into the software's functionality and behavior without requiring knowledge of its internal implementation.

## White Box Testing

White-box testing, also known as clear-box testing or structural testing, is a software testing technique that examines the internal structure and logic of an application. Testers have access to the application's source code and use this knowledge to design test cases that validate the correctness and completeness of the code's implementation. Unlike black-box testing, white-box testing focuses on testing the software's internal workings, including control flows, data flows, and the interaction of different components.

One of the primary objectives of white-box testing is to ensure that all code paths and branches are executed, providing maximum code coverage. Testers use various techniques such as statement coverage, branch coverage, and path coverage to measure the extent to which the source code has been exercised during testing. This helps identify areas of the code that have not been tested and may contain defects.

White-box testing is especially effective during the early stages of development when code is continuously evolving. It helps detect coding errors, logic flaws, and boundary cases that might not be apparent from a purely functional perspective. By catching these issues early, white-box testing can significantly reduce the cost of fixing defects in later stages of development.

One common technique used in white-box testing is unit testing, where individual units or components of the software are tested in isolation. Unit tests focus on testing small units of code, such as functions or methods, to ensure that they produce the expected output for a given set of inputs. Testers can use mock objects or test doubles to isolate the unit being tested from its dependencies, making it easier to control the test environment and focus on the specific functionality being tested.

Another important aspect of white-box testing is code review. During code review, developers or testers examine the source code to identify potential issues and provide feedback on coding practices and adherence to coding standards. Code reviews can uncover bugs, security vulnerabilities, and performance bottlenecks that might have been missed during other testing phases.

In addition to unit testing, white-box testing includes other techniques such as integration testing and path testing. Integration testing verifies the interactions between different components or modules to ensure they work correctly when integrated into the larger system. Path testing aims to test all possible execution paths through the source code to ensure that no logical errors or deadlocks exist in the application.

White-box testing is also used to assess the performance of the application. Testers can analyze the code to identify areas that might cause performance issues, such as excessive loops, inefficient algorithms, or memory leaks. By addressing these performance bottlenecks, developers can improve the overall speed and efficiency of the application.

One of the challenges of white-box testing is that it requires a deep understanding of the codebase and the ability to write test cases that cover various scenarios and edge cases. Testers must be familiar with the programming languages, design patterns, and architectural principles used in the application. Moreover, since white-box testing focuses on the implementation details, it might miss some user-centric defects that can be caught through black-box testing.

In conclusion, white-box testing is a valuable technique for ensuring the quality and reliability of software applications. By examining the internal structure and logic of the code, testers can identify defects, verify correctness, and achieve higher code coverage. White-box testing complements other testing methodologies and is an essential part of the software development lifecycle. It helps improve code quality, reduces the risk of defects, and enhances the overall user experience. When combined with black-box testing and other testing techniques, white-box testing forms a comprehensive testing strategy that ensures the delivery of high-quality software products to end-users.

### Unit Testing

Unit testing is a software testing technique where individual units or components of a software application are tested in isolation to ensure their correctness and functionality. The main objective of unit testing is to validate that each unit performs as intended and produces the expected output for a given set of inputs. Units in this context typically refer to functions, methods, or procedures, and unit testing is often performed during the early stages of the software development process.

Developers write test cases for each unit they want to test. These test cases consist of various scenarios with inputs and the corresponding expected outputs. Once the test cases are defined, they are executed automatically using unit testing frameworks, such as pytest for Python. These frameworks facilitate organizing and running test cases efficiently and reporting the results.

One of the essential principles of unit testing is isolation. Each unit should be tested independently of other units, and any external dependencies should be replaced with mock objects or test doubles. Mock objects mimic the behavior of real dependencies but allow testers to control their responses, making it easier to reproduce different scenarios during testing.

Unit testing provides numerous benefits to the software development process. Firstly, it helps catch bugs early in the development cycle, making it easier and less expensive to fix issues. Secondly, it provides a safety net for code refactoring, as developers can confidently modify code knowing that they have tests in place to detect regressions. Thirdly, unit tests serve as a form of documentation, describing the expected behavior of the code and making it easier for developers to understand its functionality. Additionally, unit testing promotes collaboration within development teams, as sharing and discussing test cases lead to a better understanding of the codebase and help identify potential design flaws.

## Gray Box Testing

Gray box testing is a software testing approach that combines elements of both black box testing and white box testing. In gray box testing, testers have partial knowledge of the internal workings of the software being tested, allowing them to design test cases that target specific components or functionalities while still considering the overall behavior of the system. This approach strikes a balance between the limited knowledge of external behavior in black box testing and the comprehensive understanding of internal code in white box testing.

In gray box testing, testers have access to certain information about the internal structure of the software, such as high-level design documents, data flow diagrams, or database schemas. This knowledge enables them to identify critical areas of the application and design test scenarios that focus on those areas. However, they do not have access to the complete source code or implementation details, which differentiates gray box testing from white box testing.

The main objective of gray box testing is to uncover defects in specific modules or functionalities of the software while evaluating its overall functionality and integration with other components. Testers use this approach to assess the interaction between different modules, data flow, and the accuracy of data processing within the application.

One of the primary advantages of gray box testing is its ability to provide a broader test coverage compared to black box testing. Testers can strategically select test cases based on their understanding of the internal workings of the software, ensuring that critical areas and potential vulnerabilities are thoroughly tested. This approach helps in identifying defects that might be missed by pure black box testing.

Gray box testing is also useful when working with large and complex systems where complete knowledge of the internal code may not be practical or feasible. By focusing on specific areas of the software, testers can efficiently allocate testing resources and achieve thorough test coverage.

Another benefit of gray box testing is its ability to simulate real-world scenarios that can reveal potential issues in the software. Testers can use their knowledge of the internal workings to design test cases that mimic various usage patterns and conditions, helping to assess the system's performance and reliability under different scenarios.

## Black Box Testing on Mango Disease Detector

Testing is a crucial phase in the software development lifecycle, and it plays a vital role in ensuring the quality, functionality, and reliability of the software product. In the context of the "Mango Disease Detector" project, testing techniques are employed to evaluate the front end of the application, focusing on the user interface (UI), user experience (UX), and overall usability. The objective of front-end testing is to identify and rectify any defects, bugs, or inconsistencies that might impact the user's interaction with the application.

| Test Case Title | Test Case Step | Test Case Data | Result Expected | Post-Condition | Result Actual | Pass/Fail |
| --- | --- | --- | --- | --- | --- | --- |
| Login - Valid Credentials | Verify the functionality of essential element on the login page | Open the login page and ensure the essential elements are present | Login page is accessible | User can get the access the application | Login page is accessible and functional. | Pass |
| Login with valid credentials | Enter valid Username and Password | Successful Login Redirect. | User redirect to home page | User login | Pass |
| Successful login process redirection | After successful user redirect to home page | User redirect to home page | User navigate to home page | User redirect to home page | Pass |
| Deny access to unregistered Users | Attempt to login with unregistered credentials | Appropriate error message | User receive error message | Appropriate error message | Pass |
| Login with invalid username | Attempt to login with invalid username | Appropriate error message | User receive error message | Appropriate error message | Pass |
| Login with invalid password | Attempt to login with invalid username | Appropriate error message | User receive error message | Appropriate error message | Pass |
|  |  |  |  |  |  |  |
| Sign Up | Verify Sign up functionality with valid inputs | Open sign up form and ensure its function working properly | Sign up form is accessible and functional | User can log in to the system | Sign up form is accessible and functional | Pass |
| Validation of mandatory field with valid input. | Enter valid user information into all fields. | Successful sign up message | User can log in to the system | Successful sign up message | Pass |
| Handling missing information in sign up | Leave one or more mandatory field empty | Error message display. | User receive error message | Error message display. | Pass |
|  |  |  |  |  |  |  |
| Home Page | Navigate to the Dashboard and access the different elements of the system. | Navigate to different elements, apply filters, and use search bars. | User can use different element and functions. | User can access all the functionality of the system | Home page function as intended. | Pass |
|  |  |  |  |  |  |  |
| Upload Image | Open gallery or browse image for prediction. | Click on image upload option and select the image | Image Upload successfully | User can predict the images | Image Upload successfully | Pass |
| Delete selected image and upload the new image | Select and delete image and upload the new one | Selected image is deleted and new image is uploaded | User can delete and upload images | Selected image is deleted and new image is uploaded | Pass |
| Perform disease detection after image upload | Select the image upload it and predicted image | Disease predicted display. | User can predict disease | Disease predicted display. | Pass |
|  |  |  |  |  |  |  |
| Capture Image | Activate device camera and capture the image | Click on capture image button and capture the image | Device camera activate | User can predict the captured image | Device camera activate | Pass |
| Verify captured images are store correctly | Capture an image and validate it if it store correctly. | Capture image store correctly | User can predict the captured image | Capture image store correctly | Pass |
|  |  |  |  |  |  |  |
| Cure | Enter the specific disease and check for its cure | Enter the disease that are stored into database and checks if it is correctly displayed with its cure, | Correct cure is displayed | User receive correct cure information | Correct cure is displayed | Pass |
| Enter the disease that is not present into the system | Enter the disease that is not present and checks for an appropriate error message | Error message is being displayed. | User receive an error message | Error message is being displayed. | Pass |
|  |  |  |  |  |  |  |
| Prediction report | When user have internet connectivity | Image uploaded/captured by user for prediction | System will show predicted report for the input leaf image along with percentages of disease present | User will be able to see and download the report | User will be able to see and download the report | Pass |
|  |  |  |  |  |  |  |
| Prediction report | When user have internet connectivity | Image uploaded/captured by user for prediction | System will show predicted report for the input leaf image along with percentages of disease present | User will be able to see and download the report | User will be able to see and download the report | Pass |
| When user doesn't have internet connectivity | Image uploaded/captured by user for prediction | System will show predicted report for the input leaf image along with percentages of disease present | User will only be able to see the report | User will be able to see the report but cannot download without internet connection | Pass |
|  |  |  |  |  |  |  |

* **Login**

The login page serves as the gateway to the application and requires rigorous testing to ensure its security and usability. Testers verify the presence and functionality of essential elements, such as the "Forgot Password" link or button, which allows users to reset their passwords if needed. Special attention is given to the login validation process, where testers simulate various scenarios of incorrect login credentials, ensuring that the appropriate error messages are displayed. This is critical to providing clear and meaningful feedback to users, enhancing their experience with the application.

The successful login process is also evaluated to validate that users are redirected to the home page seamlessly. Additionally, the testing team assesses how the application handles locked accounts resulting from multiple failed login attempts. The login cancellation functionality is examined to ensure that users have the option to abort the login process if needed. Finally, testers verify that the application denies access to unregistered users, enhancing the overall security of the application.

* **Sign Up**

The sign-up process is a crucial aspect of the application as it allows new users to register and gain access to the system. Testers conduct comprehensive testing to ensure that the sign-up form functions correctly. This includes validating that all mandatory fields are correctly marked and that the appropriate error messages are displayed when any required information is omitted. These validations are crucial to guide users in completing the sign-up process accurately. After successful sign-up, the application should provide users with a confirmation notification, indicating that their account has been created. This step is essential in enhancing user satisfaction and engagement. Proper testing of the sign-up process ensures that the application's user base can grow seamlessly and that all registered users can access the features and functionalities of the "Mango Disease Detector."

* **Home Page**

The home page serves as the central hub of the application, providing users with access to various features and functionalities. Testers ensure that all the specified functionalities are present and functioning as intended. These may include navigation buttons, search bars, filters, and other user interface elements. To validate the home page's functionality, testers simulate different user interactions and assess the application's response. This includes testing the navigation between different sections of the application and verifying that relevant information is displayed accurately. The responsiveness of the home page to user inputs is crucial for providing an optimal user experience.

* **Upload Page**

The upload page allows users to select images from their device's gallery for disease prediction. Testers conduct rigorous testing to verify that the gallery opens successfully when the user selects the image upload option. The selected image should be displayed correctly on the screen, ensuring that users can identify the image they wish to upload. To enhance user convenience, testers ensure that users can delete selected images and upload new ones as needed. The ability to upload multiple images without encountering errors is vital for smooth user interactions. After the image upload, the application performs disease prediction, and testers validate the accuracy of the predicted report. This ensures that users receive reliable and trustworthy information about the health status of their mango leaves.

* **Capture**

The capture feature is a valuable addition that allows users to capture images using the application's camera. Testers assess the functionality of the capture image button, ensuring that it activates the device's camera as intended. Additionally, the application should display the appropriate permission checkbox, seeking user consent for accessing the device's camera. After capturing an image, testers verify that it is correctly stored in the application's image storage. The application should handle image storage efficiently, ensuring that captured images are saved securely and can be accessed for further analysis and disease prediction. Testers also validate the process of uploading captured images for prediction, ensuring that users can make informed decisions about their mango plant's health based on the results.

* **Cure**

The cure feature is a valuable resource for users seeking remedies for specific diseases affecting their mango leaves. Testers validate that the correct cure is displayed when users enter a particular disease. The accuracy of this information is crucial for providing users with practical solutions to address any issues with their mango plants. To maintain the application's credibility, testers ensure that an appropriate error message is displayed when users enter a disease that is not supported by the application. This ensures that users are not provided with misleading or inaccurate information, further bolstering the application's trustworthiness.

* **Prediction report**

When the user has an internet connection, they can easily upload or capture an image of a mango leaf for disease prediction. The system then generates a comprehensive prediction report, complete with the percentages indicating the severity of any diseases detected. Users have the convenience of both viewing and downloading this detailed report, ensuring they have access to valuable information about their mango plants.

In cases where internet connectivity is available, the app still delivers a seamless experience. Users can upload or capture leaf images, and the system promptly generates a prediction report with disease percentage breakdowns. However, in this scenario, users can only view the report within the app; downloading is disabled without an internet connection.

Front-end testing is an integral part of the software development process, particularly for the "Mango Disease Detector" project. Through a systematic and thorough evaluation of the user interface, user experience, and overall usability, testers can identify and address potential issues, ensuring that the application meets user expectations and requirements. By validating the login, sign-up, home page, upload page, capture functionality, and cure feature, the testing team can provide valuable feedback to developers, leading to a refined and reliable application. Effective front-end testing enhances user satisfaction, promotes user engagement, and ultimately contributes to the success of the "Mango Disease Detector" project.

## Unit Testing on Mango Disease Detector

import unittest

import os

import tensorflow as tf

from tensorflow.keras.applications.inception\_v3 import InceptionV3, preprocess\_input

from tensorflow.keras.layers import Flatten, Dense

from tensorflow.keras.models import Model

from zipfile import ZipFile

file\_name = "/content/drive/MyDrive/External Images.zip"

with ZipFile(file\_name, 'r') as zip:

  zip.extractall()

  print('Done')

from zipfile import ZipFile

file\_name = "/content/drive/MyDrive/CS-06 MANGO DISEASE DETECTOR DATASET.zip"

with ZipFile(file\_name, 'r') as zip:

  zip.extractall()

  print('Done')

class ModelTesting(unittest.TestCase):

    @classmethod

    def setUpClass(cls):

        cls.train\_dir = '/content/CS-06 MANGO DISEASE DETECTOR DATASET'

        cls.test\_dir = '/content/External Images'

        cls.num\_classes = 6

        # Create ImageDataGenerator for training and testing data

        cls.data\_generator = tf.keras.preprocessing.image.ImageDataGenerator(

            preprocessing\_function=preprocess\_input,

            rotation\_range=20,

            width\_shift\_range=0.2,

            height\_shift\_range=0.2,

            shear\_range=0.2,

            zoom\_range=0.2,

            horizontal\_flip=True,

            rescale=1./255

        )

        # Generating train dataset

        cls.train\_generator = cls.data\_generator.flow\_from\_directory(

            cls.train\_dir,

            target\_size=(224, 224),

            batch\_size=32,

            class\_mode='categorical'

        )

        # Generating test dataset (without augmentation)

        cls.test\_generator = cls.data\_generator.flow\_from\_directory(

            cls.test\_dir,

            target\_size=(224, 224),

            batch\_size=32,

            class\_mode='categorical',

            shuffle=False

        )

    def test\_data\_preparation(self):

        # Test the number of batches in the train dataset

        self.assertEqual(len(self.train\_generator), 705)

        # Test the number of batches in the test dataset

        self.assertEqual(len(self.test\_generator), 10)

        # Test the number of classes

        self.assertEqual(self.train\_generator.num\_classes, self.num\_classes)

        self.assertEqual(self.test\_generator.num\_classes, self.num\_classes)

    def test\_model\_architecture(self):

        # Test the base model creation

        base\_model = InceptionV3(input\_shape=(224, 224, 3), include\_top=False)

        self.assertIsNotNone(base\_model)

        # Test the addition of custom classification layers

        x = Flatten()(base\_model.output)

        x = Dense(units=self.num\_classes, activation='softmax')(x)

        model = Model(base\_model.input, x)

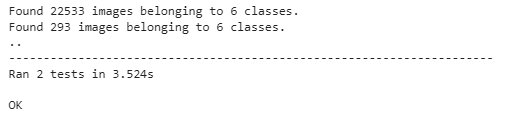
        self.assertIsNotNone(model)

        # Test the output shape of the model

        self.assertEqual(model.output\_shape, (None, self.num\_classes))

if \_\_name\_\_ == '\_\_main\_\_':

    unittest.main(argv=['first-arg-is-ignored'], exit=False)



The provided code is a unit test script written in Python using the `unittest` framework. Unit testing is a software testing method where individual units or components of a software system are tested in isolation to ensure they are functioning correctly. In this case, the unit test script is used to test the data preparation and model architecture for a Mango Disease Detector application.

The script uses the `ImageDataGenerator` from TensorFlow/Keras to preprocess and augment the image data. The `ImageDataGenerator` allows the script to read images from the specified directories, apply data augmentation techniques such as rotation, shifting, and flipping, and convert the images into batches for efficient training.

The `setUpClass` method is a special method that sets up the test environment before running any test cases. In this method, the script defines the directories for the training and testing datasets, specifies the number of classes (6 in this case), and creates the `ImageDataGenerator` objects for both datasets. The `ImageDataGenerator` is configured with various preprocessing and data augmentation options.

The `test\_data\_preparation` method is a test case that checks the data preparation steps. It first asserts the number of batches in the training and testing datasets. The script uses the `len` function on the `train\_generator` and `test\_generator` objects to get the number of batches. The number of batches is essential to ensure that all samples in the dataset are covered during training and testing.

Next, the test case checks the number of classes in the datasets using the `num\_classes` attribute of the `train\_generator` and `test\_generator` objects. The number of classes should match the specified number of classes (6 in this case), ensuring that the data generator properly handles the categorical labels.

The `test\_model\_architecture` method tests the creation of the model architecture. It starts by creating the base model using the `InceptionV3` model provided by TensorFlow/Keras. This model is pre-trained on the ImageNet dataset and can be used as a feature extractor for transfer learning. The test asserts that the base model is successfully created.

Next, the test case adds custom classification layers to the base model to create the final model architecture. It adds a `Flatten` layer to flatten the output of the base model and a `Dense` layer with the specified number of units (6 in this case) and a softmax activation function for multi-class classification. The test asserts that the final model is successfully created and that the output shape matches the expected shape.

Overall, this unit test script ensures that the data preparation and model architecture for the Mango Disease Detector application are working correctly. It verifies that the data generator generates batches with the correct number of samples and that the model architecture is constructed as intended. The test cases play a crucial role in ensuring the reliability and correctness of the application, providing developers with confidence that the system is functioning as expected.

## Chapter Conclusion

Software testing is a critical aspect of software development that aims to evaluate and assess the quality, functionality, and performance of a software application. It involves executing the software with the intent of identifying defects, bugs, or errors, and ensuring that it meets the specified requirements and expectations. The primary objective of testing is to deliver a reliable and robust product that satisfies user needs and operates efficiently in different scenarios.

In the context of the Mango Disease Detector application, testing is of utmost importance to ensure that the system accurately detects diseases in mango leaves, provides reliable results, and delivers a seamless user experience. The testing process encompasses various levels and methodologies, each serving a specific purpose in the development lifecycle.

The first level of testing to consider is black-box testing. Black-box testing involves examining the functionality of the application without delving into its internal code or structure. Testers treat the software as a "black box," focusing solely on its inputs, outputs, and behavior. The primary goal of black-box testing is to validate whether the software functions as intended from the user's perspective, without any knowledge of its internal workings.

For the Mango Disease Detector, black-box testing is crucial in evaluating the user interface (UI), user experience (UX), and overall usability of the application. Testers thoroughly examine the login and sign-up pages to ensure that users can access the system securely and conveniently. They verify the functionality of the home page, ensuring that users can navigate through different sections and access relevant information effortlessly. The upload and capture features are rigorously tested to confirm that users can input images accurately and proceed with disease prediction. Additionally, the cure feature is examined to ensure that users receive accurate and practical remedies for specific diseases affecting their mango plants. Black-box testing provides valuable insights into the user-centric aspects of the application and helps identify any issues that may impact user satisfaction.

On the other hand, white-box testing focuses on examining the internal structure and logic of the application. Testers have access to the source code and use this knowledge to design test cases that validate the correctness and completeness of the code's implementation. White-box testing is particularly useful in the early stages of development, where code is continuously evolving. It helps detect coding errors, logic flaws, and boundary cases that might not be apparent from a purely functional perspective.

White-box testing for the Mango Disease Detector involves unit testing, which evaluates individual units or components in isolation. Testers write test cases for each unit to ensure that they produce the expected output for various inputs. Unit testing is crucial in verifying the correctness of data preprocessing, image segmentation, and disease classification. It ensures that these components perform as intended and produce reliable results for further analysis.

In addition to unit testing, integration testing is performed to evaluate the interaction between different modules or components. Integration testing verifies that all integrated modules work correctly together and that data flows smoothly between them. For the Mango Disease Detector, integration testing ensures that the data preprocessing techniques seamlessly interface with the image segmentation and classification algorithms, providing a streamlined and accurate disease detection process.

Another important testing approach is gray box testing, which combines elements of both black-box and white-box testing. In gray box testing, testers have partial knowledge of the internal workings of the application, allowing them to design test cases that target specific components or functionalities while still considering the overall behavior of the system. Gray box testing strikes a balance between limited knowledge of external behavior and a comprehensive understanding of internal code.

Gray box testing for the Mango Disease Detector involves simulating real-world scenarios to assess the application's performance and reliability. Testers use their knowledge of the internal workings to design test cases that mimic various usage patterns and conditions. This approach helps identify potential issues in the software and provides valuable feedback to developers, leading to a refined and reliable application.

Automated testing is another crucial aspect of the testing process. Automated testing streamlines the testing process by using scripts and tools to execute test cases efficiently and repeatedly. For the Mango Disease Detector, automated testing ensures that data preprocessing, image segmentation, and classification algorithms consistently produce accurate results for different inputs. It also helps developers detect any regressions or performance bottlenecks during the development process, allowing for prompt corrective action.

Overall, testing is an iterative process that plays a vital role in software quality assurance. For the Mango Disease Detector, testing ensures that the application delivers accurate and reliable disease detection results to users. By combining black-box, white-box, and gray box testing approaches, testers gain comprehensive insights into the functionality, usability, and performance of the application. These testing methodologies, along with automated testing, contribute to the delivery of a high-quality and efficient Mango Disease Detector that meets user expectations and enhances agricultural productivity. Through continuous testing, feedback, and refinement, the Mango Disease Detector remains at the forefront of agricultural technology, supporting farmers and agricultural experts in managing crop diseases effectively and ensuring sustainable agricultural practices.

# Conclusions

The Mango Disease Detector project represents a significant step forward in leveraging modern technologies to address critical challenges faced by the agriculture sector, particularly in countries like Pakistan, where agriculture plays a pivotal role in the economy. By focusing on the detection of mango diseases and providing tailored recommendations for curing affected plants or farms, this mobile and web application has the potential to revolutionize mango farming practices, enhance productivity, and mitigate economic crises caused by malformation.

Throughout the development of the Mango Disease Detector, the primary objective was to create a user-friendly and effective platform that empowers farmers with the ability to detect diseases early on and take timely and targeted actions.

The core functionality of the application, centered around digital image processing and classification using Convolutional Neural Networks (CNNs), has proven to be a powerful tool in mango disease detection. By simply capturing an image of a mango plant and uploading it to the application, farmers can obtain a comprehensive analysis of the plant's health status. The implementation of image preprocessing techniques, such as color spacing, resizing, and filtering, ensures that the uploaded images are of optimal quality and prepared for disease analysis.

The system then extracts crucial features from the images, including color information and pixel characteristics. These extracted features serve as inputs to the trained CNN models responsible for classifying the images and detecting the presence of diseases.

The CNN models have been trained on vast datasets containing labeled images of healthy mango plants and those affected by various diseases. This training process has equipped the models to accurately differentiate between healthy and diseased mango plants, even when faced with new and previously unseen images. The outcome of the analysis is presented to the user through a comprehensive report, specifying the type of mango disease detected, if any. Furthermore, the application provides tailored recommendations and methods for curing the specific disease found on the plant. This personalized approach to disease management ensures that farmers receive precise guidance, enabling them to take appropriate measures to prevent further spread and minimize losses.

The implementation of the Mango Disease Detector promises a multitude of benefits for farmers and the agriculture sector in Pakistan. Early detection of diseases plays a crucial role in preventing extensive damage and reducing the economic burden on farmers. By providing timely insights, the application empowers farmers to make informed decisions and implement targeted interventions to protect their crops. Additionally, the ability to access the application through mobile and web technologies enhances its accessibility for farmers across different regions of the country.

With smartphones becoming increasingly prevalent, even in rural areas, the Mango Disease Detector becomes a powerful tool for reaching and assisting a wide user base. Furthermore, the project addresses the pressing issue of minimizing economic crises caused by mango malformation. By equipping farmers with a technology-driven solution, the Mango Disease Detector contributes to increased agricultural productivity and better-quality produce. Healthier mango crops translate into higher demand and more competitive market prices, positively impacting export revenues and bolstering the national economy. The application's reliance on artificial intelligence and deep learning technologies opens up opportunities for ongoing improvements and scalability.

Continuous updates and expansions can be implemented to include additional diseases, crop types, and agricultural regions, making the Mango Disease Detector an evolving and ever-improving tool for farmers.

# Future work

In the relentless pursuit of elevating agricultural productivity and bolstering the livelihoods of farmers, the Mango Disease Detector project has artfully laid the groundwork for an expansive horizon of future advancements. In our ongoing quest to amplify the impact of the application, forthcoming endeavors shall encompass an even broader spectrum of disease detection capabilities, spanning across a more comprehensive array of mango diseases. Additionally, our unwavering commitment extends to the extension of this cutting-edge technology, providing crucial support to a diverse range of crops beyond mangoes.

Intriguingly, the road ahead entails a strategic focus on tailoring region-specific adaptations that cater to the unique challenges faced by agricultural communities in various locales. This approach is further enriched by the seamless integration of real-time monitoring facilitated by Internet of Things (IoT) devices. Such an integration not only offers tailored and immediate intervention but also paves the way for a new era of precision-driven agriculture.

Recognizing the paramount importance of expertise, we eagerly anticipate a symbiotic collaboration with agricultural specialists. This dynamic partnership shall embrace the integration of remote sensing data, thereby augmenting the accuracy of disease detection and recommendations. This fusion of human knowledge and technological prowess holds the potential to reshape the very landscape of modern agriculture.

Our vision encompasses not only the technical prowess of the project but also its accessibility and usability. This will be realized through initiatives that foster robust user engagement, bolstered by multilingual support to cater to a global audience. Moreover, the project's enduring success hinges upon the implementation of stringent data privacy and security measures, ensuring that sensitive agricultural information remains safeguarded.

In essence, our journey continues with an unwavering commitment to revolutionize agricultural practices and empower farmers worldwide. The Mango Disease Detector project, while rooted in the present, stands poised to cultivate a future where technology and innovation harmonize to propel agriculture to new heights of prosperity and sustainability.

# Appendices

Following are the appendices in details.

## Appendices A

The precise details and overview of our project will be discussed in this chapter. The main purpose of our project is to identify a diseased mango leaf and also to provide the user with the suitable recommendations of cure based on the predicted disease to help control the disease to spread among other leafs or farms  or to completely eradicate the disease . For the detection of disease the image captured or uploaded by the user, image  will be first pre-processed , in the pre-processing stage image resizing ,filtration , color spacing , gray scaling as well as histogram equalization will be applied after the pre-processing stage is completed , the pre-processed image will be used for feature extraction after that classification will be performed on the already processed image , once the classification is done our system will show the results to the user containing the predicted disease as well as majors to be taken to control that disease and to yield better production of mangoes.

The major functionality of Mango Disease detection app revolves around digital image processing and classification system.

From users perspective our app will be easier to be use , as the user or the farmer who encounters a diseased mango leaf or the kind of leaf which looks different from normal mango leaf , our user can simply capture the image by giving access to our system for capturing image and upload it into our system to check whether the leaf is actually diseased or is healthy , the other way can be uploading the image , our user can directly upload the image into our system from their image gallery and once they click on predict our system will provide them with all the results derived from the processing of the leaf and will show the report whether the leaf is diseased or not . If the leaf is affected by any disease our system will also show recommendations to user to control that disease .

As far as our systems backend processing is concern , one the user or farmers captures the image or uploads the image our , system will perform pre-processing techniques on that acquired image , our preprocessing stage includes image resizing ,filtration , color spacing , gray scaling as well as histogram equalization to enhance the quality of image also to do better disease analysis. Once pre-processing is completed the image  must then have a label applied to it that specifies whether the mango plant/leaf  is ailing or healthy. The system extracts features, such as pixel properties and color information. This Information then will be used as an input for our trained CNN model ,which is used for classification of image .After that finally the report will be shown to the user predicting whether its healthy or is diseased along with the disease name and suggestions for improving the condition.

One of the major purpose of this app is to make farmers life easier , now farmers will just upload  the image and predictions will be made according also suggestions will b given according to the predicted  disease .

The detailed processing which will be done once the image is uploaded by user can be described as:

* **Upload Image**

The user can either capture the image by giving access to outsystem or can upload the image from photo gallery.

* **Pre-Processing**

Preprocessing stage includes image resizing, filtration, color spacing, gray scaling as well as histogram equalization to enhance the quality of image also to do better disease analysis Also to remove any noise the acquired image.

* **Feature Extraction**

Feature extraction is an important part to make accurate predictions, in this stage all the patterns on the leaf will be identified.

* **Classification**

Once all the feature extraction of the acquired image is done, classification will be done. The role of classification is to classify the leaf based on extracted features.

* **Results**

Once all the processing is done on the acquired image i.e. pre-processing, feature extraction and classification. Our system will show user the report containing disease predicted along with the suggestions to cure it. The results will also be stored in the database, creating a record of disease detections for each mango leaf image.

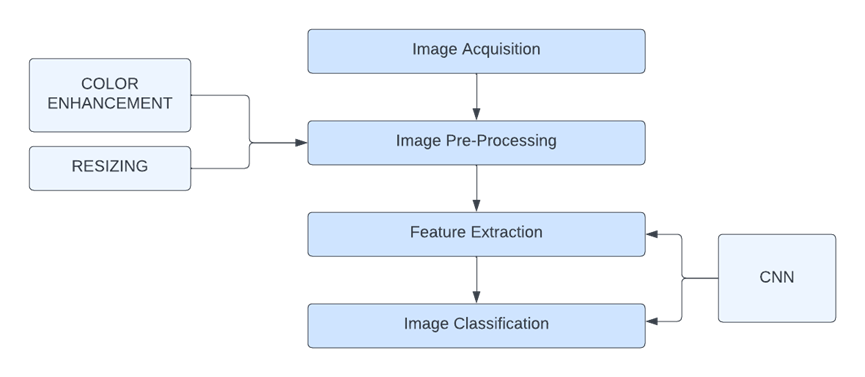
## Appendices B:

. Following diagram shows the basic flow of our project. Furthermore, we will explain the UML diagrams of our system for better explanation of our project’s Mango Disease Detections workflow.

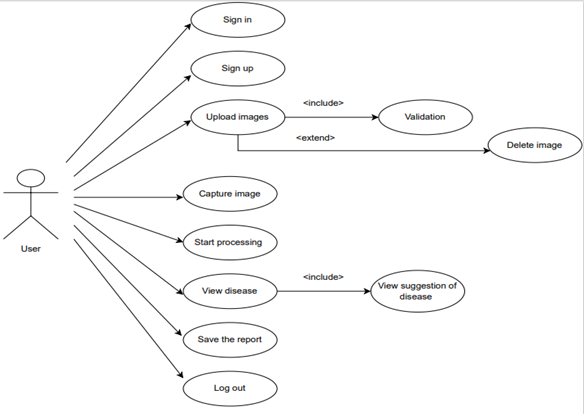
### UML Diagrams

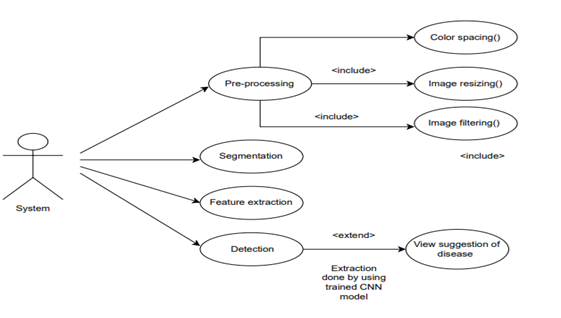
 Below are the UML diagrams of our project.

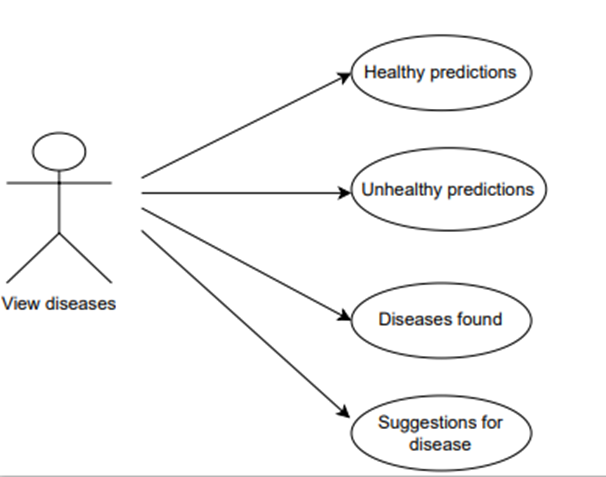
#### System Diagram



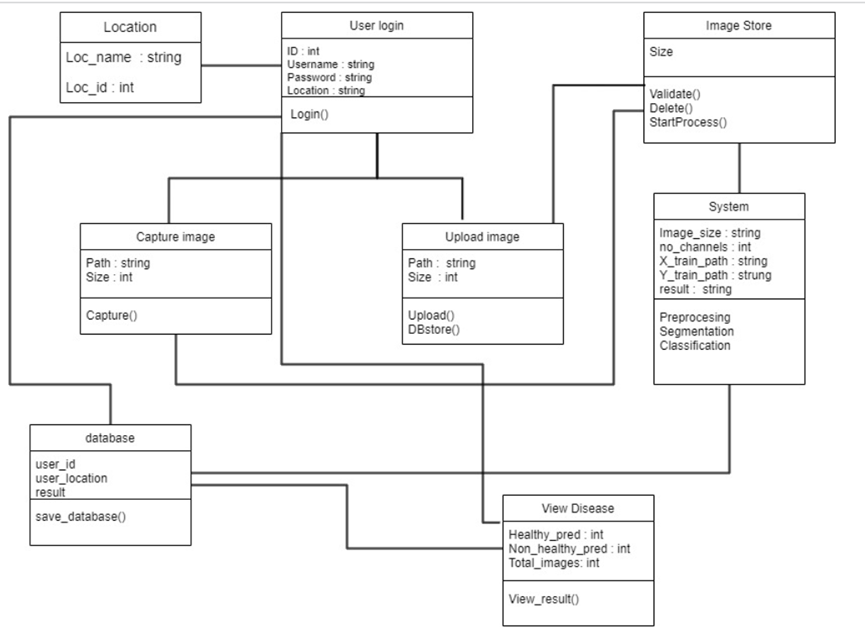
#### Use Case Diagram



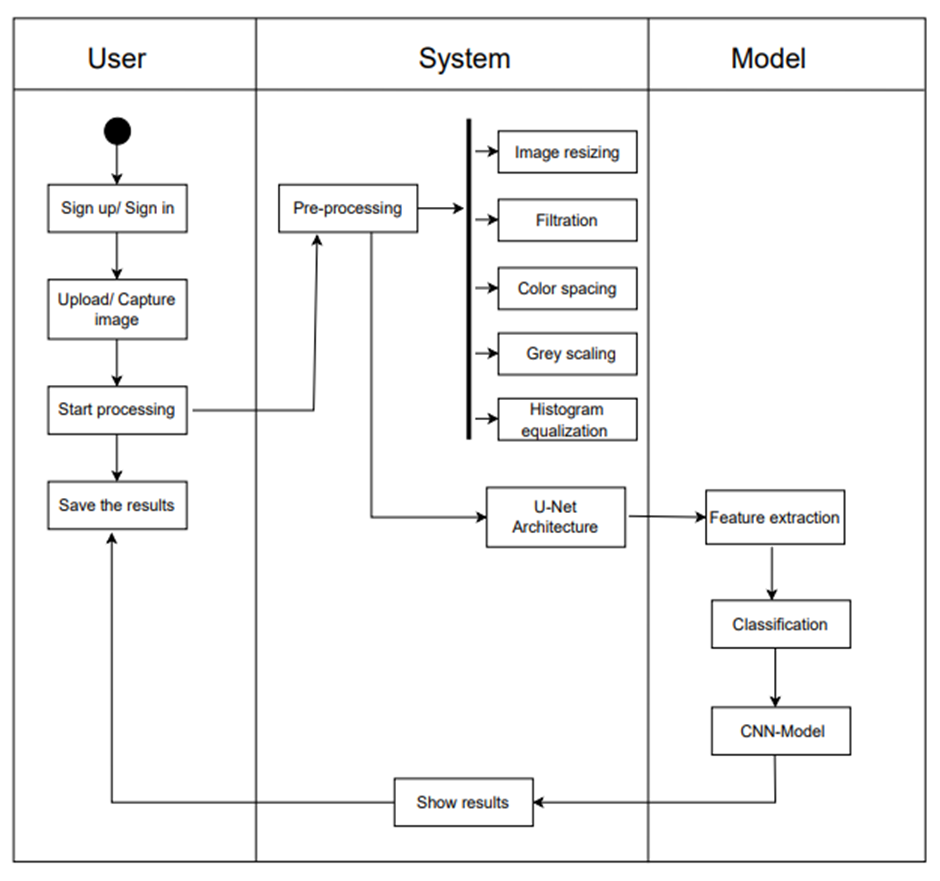




#### Class Diagram



#### Activity Diagram



## Appendices C

 Stake holders and their concerns about our project.

| **Attributes** | **Content** |
| --- | --- |
| Stakeholder | users / owner’s of farm |
| Goal | To view dashboard and reports |
| Task | How to  view dashboard and reports? |
| Concern | How to view details of different farms?  How to view details of a specific tree?  How to add users?  How to do announcement for other users? |

| **Attribute** | **Content** |
| --- | --- |
| Stakeholder | Local users / farmers |
| Goal | To view predict disease |
| Task | How to predict disease? |
| Concern | How to login?  How to click/ upload picture?  How to view previous reports? |

| **Attribute** | **Content** |
| --- | --- |
| Stakeholder | Maintenance manager |
| Goal | To maintain the system |
| Task | How to maintain the system?  How to troubleshoot the problems? |
| Concern | How is the communication carried out between different components of the system?  How to tackle version difference?  How to maintain a database? |

## Appendices D

 Detecting diseases in plants at early stages is extremely important, especially the fruit trees, because they are one of the products which are being exported to support the country's economy. The presence of any disease in these trees can promote lack of growth or mold fruits which will negatively affect the country’s export system.

Pakistan is one of the major exporters of mangoes and due to Pakistan extreme weather conditions some of the times the trees catch various diseases affecting the yield of mangoes, also their taste as well as can result in mouldy fruit. It can also spread among other trees if the precautions are not taken at early stages. Most of the time due to lack of resources for farmers, they are not able to take precautions at early stages of the disease which also might result in spoiling the other trees as well.

To facilitate farmers to take precautions at early stages of disease we created this system called mango disease detector, by using this farmer will simply clicks the picture of the leaf of the tree he thinks is infected and our system based on that image after performing some processing will show the results that whether the leaf is infected or is healthy. In case of diseased leaf detections our system will also provide farmers with the suggestions to stop spreading of the disease as well as recommendations on how these diseases can be eradicated completely.

**11.5 Appendices E**

### Plagiarism Report



# References

[1] Mango Quality Grading using Deep Learning Technique: Perspectives from Agriculture and Food Industry: Proceedings of the 21st Annual Conference on Information Technology Education.

<https://dl.acm.org/doi/abs/10.1145/3368308.3415370>

[2] Mango Leaf Diseases Recognition Using Neural Network and Support Vector,

<http://www.ijcse.net/docs/IJCSE19-08-01-010.pdf>

[3] Tech Science Press, Mango Leaf Disease Identification Using Fully Resolution Convolutional Network, August 24, 2021

<https://www.techscience.com/cmc/v69n3/44135/html>

[4] Mango Diseases Identification by a Deep Residual Network, IEEE Xplore

<https://ieeexplore.ieee.org/document/9214620>

[5] MDPI, Botanical Leaf Disease Detection and Classification Using Convolutional Neural Network: A Hybrid Metaheuristic Enabled Approach, May 20, 2022

<https://www.mdpi.com/2073-431X/11/5/82/html>

[6] Global Transitions Proceedings, Deep Learning Precision Farming: Grapes and Mango Leaf Disease Detection by Transfer Learning, August 12, 2021

<https://www.sciencedirect.com/science/article/pii/S2666285X21000303>

[7] MangoLeafBD: A Comprehensive Image Dataset to Classify, arXiv.org

<https://arxiv.org/pdf/2209.02377.pdf>

[8] MDPI, Mango Leaf Disease Recognition and Classification Using Novel Segmentation and Vein Pattern Technique, December 14, 2021

<https://www.mdpi.com/2076-3417/11/24/11901/htm>

[9] (PDF) Mango Disease Detection by using Image Processing

<https://www.researchgate.net/publication/335816862_Mango_Disease_Detection_by_using_Image_Processing>

[10] A Comparative Study of CNN and AlexNet for Detection of image classification- IEEE Xplore

<https://ieeexplore.ieee.org/abstract/document/8977648>

[11] SpringerLink, A Novel Fusion of Deep Learning and Android Application for Real-Time Mango Fruits Disease Detection, January 01, 1970

<https://link.springer.com/chapter/10.1007/978-981-15-5400-1_74>

[12] Global Transitions Proceedings, Deep Learning Precision Farming: Grapes and Mango Leaf Disease Detection by Transfer Learning, August 12, 2021

<https://www.sciencedirect.com/science/article/pii/S2666285X21000303>

[13] Postharvest Biology and Technology, Detection and discrimination of two fungal diseases of mango (cv. Keitt) fruits based on volatile metabolite profiles using GC/MS, February 20, 2007

<https://www.sciencedirect.com/science/article/abs/pii/S0925521407000245>

[14] Early Disease Classification of Mango Leaves, IEEE Xplore

<https://ieeexplore.ieee.org/abstract/document/9229136/>

[15] MDPI, Detection and Management of Mango Dieback Disease, October 20, 2017

<https://www.mdpi.com/1422-0067/18/10/2086>

[16] Indonesian Journal of Science and Technology, Detection of Mango Tree Varieties Based on Image Processing

<https://ejournal.upi.edu/index.php/ijost/article/view/3800>

[17] Journal of Electrical and Computer Engineering, Deep Learning-Based Image Processing for Cotton Leaf Disease and Pest Diagnosis, June 16, 2021

<https://www.hindawi.com/journals/jece/2021/9981437/>

[18] Journal of Computer and Communications, Mango Diseases Classification Solutions Using Machine Learning or Deep Learning: A Review, December 15, 2022

<https://www.scirp.org/journal/paperinformation.aspx?paperid=121856>

[19] MDPI, Computer Vision System for Mango Fruit Defect Detection Using Deep Convolutional Neural Network, November 02, 2022

<https://www.mdpi.com/2304-8158/11/21/3483>

[20] Detection and Localisation of Farm Mangoes using YOLOv5, EEE Xplore

<https://ieeexplore.ieee.org/abstract/document/9825078/>

[21] DISEASES OF MANGO AND THEIR MANAGEMENT – ResearchGate

<https://www.researchgate.net/publication/311715840_DISEASES_OF_MANGO_AND_THEIR_MANAGEMENT>

|  |
| --- |
|  |
|  |
|  |