Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X
Volume 4, Issue 5, 2026

ADVANCED DEEP LEARNING METHODS TO ACCURATELY DETECT AND CLASSIFY PLANT DISEASE

[*1]Farhan Ali

*[*1]School of Computer and IT, Beaconhouse National University, P.O. Box 53700 Lahore, Pakistan*

*[*1]farhan32748@gmail.com*

DOI: https://doi.org/10.5281/zenodo.20132276

Article History
Received on: 14 April 2026
Accepted on : 08 May 2026
Published on: 09 May 2026

Corresponding Author:
Farhan Ali

Abstract

Plant diseases are a big challenge in the agricultural productivity, food security and livelihood of farmers around the world. Conventional disease detection techniques where human judgment plays a major role are usually lengthy, inaccurate and unavailable in the rural communities. In recent years, machine learning, and deep learning have also made it possible to develop automated plant disease detection systems that are able to identify and classify plant diseases correctly with the help of leaf images. This paper discusses how computational methods can be used to diagnose the diseases in plants through the analysis of the visual symptoms, including the discoloration of leaves, spots, and textural features. We used convolutional neural networks (CNNs) to train a model using a set of images of plant leaves labelled with different diseases to identify the disease of different crop species in the dataset. The suggested system is highly accurate and scalable, and it provides a practical application of the early detection of the disease and a prompt intervention. This type of technology can help a great deal in precision agriculture, lessen the waste of pesticides, and enhance real-time monitoring of the health of crops.

1. Introduction

Plant diseases have a critical impact on the productivity and quality of crop as a serious challenge to the farmers and security of food globally. These diseases should be detected as early and as accurately as possible to avoid spreading them and implement timely treatment. The process of disease identification has traditionally been based on the visual observation of agricultural specialists that are time-consuming and subjective and may not always be available to farmers on a small scale or remote farms. Mohanty, Hughes and Salathé [1] describe using deep learning for image-based plant disease detection. In this work, will focus on creating a plant disease detection system using image processing without the use of machine learning by digitally analyzing we now are able to empower entrepreneurs to make the right decisions that would translate to success or failure. The main idea is to produce a good and affordable device which may be adopted by the farmers and agriculture workers to help them detect the common diseases of the plant through an easy automated method. With this strategy, it is feasible that even in the cases when the developed countries are not accessed, there is an elicitation of the least changes. Computational power or massive data that machine learning models require deterministic modelling with rules that will be useful in detecting common afflictions of plants by the help of an easy and automated process. By doing so, it is possible to ensure that without having access to the sophisticated computational tools and large datasets which machine learning models use, efficient disease detection is still possible using deterministic, rule-based image analysis. Sladojevic et al. [2] explain deep neural networks-based recognition of plant diseases by leaf image classification.

Agriculture is still a vital food and economic pillar in the world. Nevertheless, the spread of plant diseases is among the most urgent and widespread challenges to farm production that not only reduce yields but also the quality of food, its salability, and the lives of farmers. Ferentinos [3] illustrate deep learning models for plant disease detection and diagnosis. The historical process of identifying plant pathology has been based on hand inspection and consultation with experts a process that is lab intensive, time intensive and prone to subjectivity. Additionally, in geographically isolated or resource limited areas, where trained agronomists may not be available, the issue of prompt and precise detection of the disease may further be worsened. To overcome such shortcomings, technological advances have started to transform the face of precision agriculture. Arnal [4] elaborate digital image processing techniques for detecting, quantifying and classifying plant diseases. Plant disease detection through traditional digital image processing is one of these promising methods that are cheap and less complex than advanced methods of machine learning. This method allows identifying obvious disease signs such as chlorosis, necrotic lesions, and abnormal foliage structures by using the classical algorithm, i.e., color segmentation, edge detection, and morphological filtering. In contrast to data-intensive artificial intelligence models, which require large labeled datasets and large computational resources, traditional image processing methods are computationally inexpensive, interpretable, and sometimes even available, which is especially beneficial to apply in farming situations with limited resources.

The proposed research is devoted to the design and construction of the efficient, yet affordable, plant disease detection system that will entail complex utilization of the simplest digital-imaging equipment and traditional pattern recognition models. The general goal is to provide the smallholder farmers with an easy to use, handheld, and self-sufficient diagnostic system that has the ability to detect diseases at early stage. Hughes and Salathé [5] examine an open access repository of images on plant health to enable the development of mobile disease

diagnostics. A system that facilitates proactive agricultural processes not only reduces loss of crops and is very much appropriate in reducing the overuse of agrochemicals but also promotes sustainable agricultural practices therefore aiding the wider objectives of environmental management and food system sustainability.

Several studies have explored the use of image processing and artificial intelligence techniques for plant disease detection. Early research focused on traditional digital image processing methods for identifying infected regions on plant leaves. With the advancement of deep learning, researchers have successfully applied convolutional neural networks to classify and recognize plant diseases with high accuracy. Too, et al. [6] a comparative study of fine-tuning deep learning models for plant disease identification. Patil [7] assess pomegranate fruit diseases detection using image processing techniques. Data sets publicly available have also contributed to the design and test of automated detection systems. The comparison of various deep learning architectures and the estimation of the severity of diseases have also been compared using recent works. Taha et al. [10] describe emerging technologies for precision crop management towards agriculture 5.0. Patil and Kannan [11] examine rice plant disease detection using image processing. In general, the articles reveal that automated plant disease detection systems can offer a solution to early disease detection and better crop management, which is quick, reliable, and affordable.

1.1 Overview

The disease detection in plants is an important operation of agriculture which seeks to detect indicators of disease or unhealthiness in crops brought about by invaders like fungus, bacteria, virusor stress on the surroundings. Sindhu and Sindhu [8] interpret image processing technology application for early detection and classification of plant diseases. Arivazhagan et al. [9] explain detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features. It is necessary to detect it as early as possible to avoid crop loss and the decrease of the amount of produce employment of pesticides and guarantee nutritious agricultural production. Historically, malady identification has been dependent on visual detection by expert trained persons, but suffers due to incorrect decisions by people, the cost of labor and the speed of response. Bhargava et al. [14] describe plant leaf disease detection, classification, and diagnosis using computer vision and artificial intelligence. Nkwocha, and Chandel, [15] asses towards an end-to-end digital framework for precision crop disease diagnosis and management based on emerging sensing and computing technologies.

In within the past several years technological methods have been invented that have aimed at automating this process.by applying traditional image processing or by employing machine learning In systems which are not machine learning based, diagnosis of the disease is done through preprogrammed codes and regulations dependent on the visual aspects of infected plant parts mainly leaves. The approaches that are employed in these systems include Color analysis: to find out dyeing or spotting of leaves. Edge detection: to identify lesion boundaries or irregular leaf shapes.

- Texture analysis: to distinguish between healthy and infected tissue.
- Morphological operations: to refine detected regions and remove noise.

One may implement such systems in computers and find them the most practical in deficient resource environments or a setting in which complex models were not easily trained feasible. They offer an affordable and convenient resource to examine the condition of the plant health and the direction of timely interventions.

1.2 Statement of Problem

Farmers in most farm-producing areas (majority of them developing nations) depend greatly on manual reviewing to identify diseases in crops. Such classic model is not the only one that has a right to be associated with this traditional

approach. Although it is labor-intensive, time consuming, but still it is very dependent on the experience and familiarity about the person who was inspecting. Consequently, the plant diseases most of the time they are not realized until they have advanced stages and this causes severe crops destruction, low productivity, and loss of money. In addition, the small scale farmers cannot access professional agricultural advisers. As well as the employment of such difficult or costly technologies as machine learning based the problem with systems is that most of the time it is not possible especially in terms of resource and also technical expertise and infrastructural constraints. Thus, an efficient and cost effective plant disease detector is required, which should be simple and also inexpensive. The system that will be identified at an early stage to determine common diseases without using machine learning. A fundamental image processing must form the basis of such system such techniques that can be executed by low-end equipment and feedback to the user in a timely manner performing duties better in managing the disease and assist in improving agriculture productivity.

1.3 Purpose of the Project

This project is meant to introduce and create an automated system to be used to find out the plant diseases with the help of the traditional image processing algorithms. By analyzing visual effects like change in color, spots and texture on the plant leaves the system seeks to aid in the process of farmers and the agricultural workers detecting the diseases early stage. The project will aim at offering a low cost and practical solution, which is not reliant on User friendliness on machine learning or huge datasets, which then causes it to be more accessible to its users scanty technical or computational facilities. It is aimed at enabling users specifically in rural, or under resource, locations a tool that improves their access to give an overview of the conditions of crop health and follow it up with the corrective measures in time, and eventually, result in the benefit of the crop better food security and increase in crops yield.

1.4 Applications of the Research

The plant disease detection system that is developed in the research has various applications in practice especially in the agricultural sector. Early detection and prevention of plant diseases is one of the most beneficial applications as it assist the farmers to take immediate intervention to ensure that their crops are not ruined by a large number of people. It comes particularly in handy to the small-scale or resource-strained farmers who might not be able to access professional farming advice or costly diagnostic equipment. The system offers an affordable and easy-to-use solution to the problem of plant health monitoring by making use of simple image capture and traditional image processing. Moreover, it can also be incorporated into education and training courses at agricultural colleges, where it can be used as a practical approach to explaining to the students the nature of plant pathology and the digital analysis of images. Precision agriculture is also encouraged through the system since farmers are able to routinely inspect the conditions of their crops and only administer treatment to the crops where and when it is required and this leads to less pesticide wastage and less expenditure. Moreover, the agricultural extension workers and NGOs can apply this tool at the field to provide fast assistance to farmers in remote areas and increase the reach and effectiveness of agricultural outreach programs. On the whole, this study can help to establish sustainable farming practices, food security, and precondition more sophisticated smart farming technologies in future.

The system will permit the farmers to detect the plant diseases at their early stages to act as soon as possible and appropriately to stop further spread and reduce crops damage. This research can be relevant to provide a low-cost non-ML based solution offer such convenient disease diagnosis instruments to non-located farmers

further technologies or professional advice. This tool should be used in educational institutions and during training where students are taught farmers on visual symptoms of plant diseases and its identification methods digital tools. Health monitoring of plants on a regular basis helps improve the cultivation of crops the change in management practices, which may result in phase out of pesticide use and reduced costs among others higher yields. Compared to AI based systems, this approach of image processing is inferior although still enough to offer a quite reasonable solution can be used to help with precision agriculture initiatives with user actionable field level data.

1.5 Theoretical Bases and Organization

This paper is founded on digital image processing and pattern principles recognition methods, most specifically classical as opposed to machine learning. The system employs the conventional implementation of the image processing algorithms to examine plant leaf pictures and visualize symptoms of diseases. The preprocessing involves taking clear quality images of leaves of plants camera. Effective disease detection requires a high quality of the image as a necessary condition it gives the raw data which is further processed.

Preprocessing plays a critical role in enhancing the quality of images and making the latter ready analysis. The methods such as gray scale conversion, noise reduction with the help of filters and contrast enhancement even assist in extracting significant characteristics (like spots, etc.). The minimization of excess data is done with lesions, or discoloration. The first indication of a disease is usually color. The techniques such as thresholding or color discoloration, yellowing or spotting of the leaves are detected based on segmentation that are usual manifestations of diseases of plants. Edge detection, i.e. the edges are found using algorithms such as Sobel or Canny areas on the leaf infected. This aids in figuring out lesions or abnormalities. The edges thus detected are then examined to make an estimation of size and severity of affected area.

Morphological transformations such as erosion and dilation are undertaken to narrow on some changes be identified in noticed areas the noise is eliminated and the features of interest are improved. These operations assist in discriminating healthy and diseased plant zones more accurately. Rather than a machine learning based system, rule-based logic is applied in this case categorize the illness severities. To take an example there are prescribed limits to the extent of size the appearance of vegetation such as color and form of the lesions could be able to indicate whether a leaf is healthy or has mild infections diseased (or, infected) seriously.

2. Literature Review

Monitoring of the plant health is a very important component of the contemporary plantation, which is supposed to guarantee the health of the plant's preservation of crop health and loss of yield. Wang, Sun, and Wang [12] interpret Automatic image-based plant disease severity estimation using deep learning. Brahimi et al. [13] explain deep learning for tomato diseases: classification and symptoms visualization. Historically diagnosis has been based on disease detection in comparison with manual observation, and incorporating the presence of experts in the diagnosis, which is time-consuming and subject to error humans' error and it is not always available to farmers in the distant locations. In order to overcome them, it is necessary to combat them by constrained by its limitations, researchers have considered some of digital image processing techniques alternative. As opposed to machine learning methods, which need a lot of data and considerable computing power, neural networks can be used to address this issue classical image processing is less complicated and can be used more easily than its computational power solution.

Color segmentation, edge detection, morphological techniques are examples of techniques used the symptoms of diseases on

plant leaves are usually detected through operations discoloration, spots and lesions e.g Patil and Kumar (2011) employed color and shape analysis as a fungal detection method. Cotton infections and Arivazhagan et al. (2013) made use of texture and color properties in detection of banana leaf disease. These literatures show that rule based systems can successfully detect plant diseases without dealing with AI models. However in spite of these developments most of the new solutions remain very machine oriented learning and identified the gap in the research of low-cost standalone systems that utilized no more than three sensors.

The conventional image processing. The given project will fill that gap by offering a disease such as detection system which is purely by classical method and therefore quite appropriate to be used together with poor farmers and those who lack technical resources and financial abilities. Plant health is an important part of contemporary agronomy, and it is critical to health-monitoring of crops to guarantee protection, reduce losses in hope of yields, and maintain the overall health of plants. Conventionally, manual observation and expert diagnosis have been used extensively to detect disease and it may be time consuming, subject to human error and not always accessible to farmers in remote and rural locations. In a bid to overcome such predicaments, researchers have looked into digital image processing as an alternative solution to the predicament. In contrast to machine learning and neural network-based algorithms, which need big amounts of data and significant computing capacity, classical image processing algorithms provide an easier and more accessible method. Staining, edge-detection and morphological operations are some of the techniques that could be useful in recognizing the visible symptoms of plant diseases, such as discoloration, spots, and lesions.

The web application development nowadays depends on the proven frameworks and the documentation materials to achieve efficiency, scalability, and maintainability. Flask is a simple and extensively portable web application framework that is especially easy to develop using and prototype with [16], [17]. Tools like Sphinx that are used to support are regularly employed to produce detailed and systematic documentation, which improves the readability of code and maintainability of projects [16]. In terms of database management, SQLite provides a stable, non-server-based database solution that is user-friendly in terms of integration and suitability to small and medium-sized applications [18]. Also, Bootstrap is also important in front-end development through offering responsive design elements and pre-built templates hence enhancing user interface design and user experience [19]. These technologies are combined in order to create a strong basis of creating a modern, efficient, and user-friendly web-based system.

2.1 Systems

The proposed plant disease detection system seeks to identify the visible symptoms of the plant diseases on the leaves using classical image processing algorithm. It works in a number of straightforward procedures beginning with the image acquisition where the latter is acquired by using a camera or uploaded by the user and the digital image of a plant leaf is taken by the camera. This then is passed to the process of preprocessing, which involves conversion to grayscale filters noise reduction, contrast modification in order to enhance clarity and to isolate key features. The system is then used to apply color segmentation technique to identify abnormal discoloration, including turn yellow, brown or black markings which in many times are signs of the disease. This is followed by the edge detection techniques such as Canny or Sobel algorithm that are then applied to locate the periphery of the lesions or diseased areas on the leaf. Morphological erosion and dilation of these results are done to improve these results assisting to clean the image and to emphasize the parts that are related to it. Lastly, according to

predefined the system relies on thresholds and rules (e.g., size, shape, and number of the given affected regions), the system executes a classification based on rules to decide the health of the plant as healthy or mildly infected or highly infected. The system is constructed easy and efficient which makes it fit to be incorporated in the mobile or desktop application, which can be utilized without any sophisticated knowledge at all and by direct involvement of the farmers and the agricultural field workers computer processing or network connectivity.

2.2 Existing System

The current plant disease detecting systems majorly depend on two solutions: manual check and machine learning solutions. In the traditional manual method there is a visual examination of crops by farmers or at least the farmers along with farming experts. Indications of the disease. Although the accuracy of this method is possible under the application of experts, it is very subjective labor intensive and could not be used in large-scale cultivation. On the on the other, the recent systems tend to be based on the models of machine learning and deep learning especially the convolutional neural networks (CNNs) that are capable of automatic learning extracts features. Has large sequences of plant images with diseases and with health. Such systems provide are quite accurate yet need large training data, and require huge computational power and skill that can be created and implemented. Consequently those solutions are not ever feasible or available to small scale farmers or users in resource limited areas. These systems also require reliable internet connectivity in many of their systems cloud based services which might not exist in the countryside places. This creates an opportunity to have less complicated, single features systems that need machine learning, but do not quite. The potential capability to provide reliable detectability of diseases may be found by the classical image processing methodologies.

2.3 Proposed System

The system being proposed will be lightweight, accessible, and efficient detection technique of plant diseases by conventional method of image processing without making use of machine learning. It is specially created to curb constrains on current systems by eradicating the necessity of data in big amounts internet connectivity or a large computation capacity. The system works by taking or uploading a picture of a plant leaf and then going through a sequence of the following steps image preprocessing (gray scale conversion, filtering and enhancement) and color segmentation detects unusual color patterns edge to find lesion boundaries and morphological operations in order to clean up and emphasize the affected regions. Based on advance rules and thresholds like the size, shape, and color of the infected areas where the system considers the leaf as healthy, mild infected and severely infected. As this rule based technique is offline and therefore the results are quickly and evenly obtained it can be used in real life to farmers agrarian employees and teachers in rural or low resource sites. The final aim is to make it possible to detect and intervene in the disease early resulting in better crop organization and minimization of loss.

2.4 Limitations and Bottlenecks

Although the plant disease detection system presents a convenient and realistic solution, it comes with a number of limitations and bottlenecks. The first weakness is that it depends on the quality of the image poor quality of lights, low-resolution pictures, or any other vague background may influence the accuracy of detection greatly. Also, as the system is not based on machine learning but on the rule-based image processing, it has fewer opportunities to identify complex or uncommon patterns of the disease. It is less flexible to new or unseen situations, only identifying diseases on which predetermined visual rules have been designed. In any environmental change like overlapping leaves, shadows, or changing backgrounds, noise can be introduced and this makes it hard to analyze.

Further, the system is not that scalable and flexible; in contrast to AI-based models, it cannot learn and be improved on new data. Finally, the basic training or orientation is necessary because some users with low technical skills might struggle to take correct pictures, or interpret the obtained results. Nevertheless, the system is a useful low-cost device despite these challenges especially to the farmers in the areas where the advanced technologies are not easily available.

The system is to identify disease according to observable symptoms mainly paying attention to the alterations in color, the texture disparities, and the lesion border. However not every disease in plants might show observable signs very early and a few might demand additional specific routes of diagnosis (verification by molecular testing) against abnormal detection. The performance of the system greatly relies on the input image quality. The system may fail by poor lighting, blurred images or low resolution pictures to be able to identify diseases. This may prove to be difficult to farmers who may not have easy accessibility to high quality cameras or cannot take pictures perfectly field.

Because the system is founded on a set of known rules through color and texture features, it can be it can only detect the diseases with explicitly defined rules. This may render the system less flexible to new or less prevalent diseases which need manual updates of its rule sets to meet new plant diseases. Such aspects as the background noise leaves in various lighting or backgrounds leaves coverage overlapping leaves can present a problem in getting an accurate disease detection. Complex backgrounds or heavily damaged systems might be difficult to the system. The leaves of the plants which may provoke the inaccurate analysis.

Rule based system of classification is simple and does not exhibit the flexibility. Flexibility that is provided by machine learning models. In patterns of diseases that are less straight forward the preset levels and criteria may not be adequately able to capture all the deviations which causes possible misclassification or false negatives. Although the system is efficient, it can be expected that real time detection of the disease can be encountered performance issues especially when there is a large set of data or more complex images. Depending on the size of the images the processing could be quite time consuming or in case there are several areas that require to be processed analyzed. Although the system is made child friendly, farmers who have little technical skills the knowledge might encounter some difficulties in utilizing the system especially in interpretation of results or high-quality image inputs. This is provided by adequate raining. It will require some guidance in order to fully utilize its application in real life situations.

2.5 Summary

This plan aims at crafting a product that is low on cost and simple to use as a plant disease detection system based on classical image processing algorithms being used but instead on the use of machine learning. The system takes pictures of leaves of the plants and treats them by going through levels such as on the one hand image preprocessing as opposed to color segmentation and edge detection. The morphological operations to detect and categorize the symptoms of the diseases. The aim is that of offer farmers especially in the resource poor regions with a tool that allows them to predict early identification of disease early enough in order to make an intervention and minimise the turnover of crops the proposed system provides a viable solution since it will not require complex it does not require machine learning models and it also does not require costly equipment, which is why it is a perfect candidate to use offline at a lowcost. Although the system has the potential of detecting various plants pathologies the weakness of image quality dependency and limited database of illnesses, and complications with complicated backgrounds can interfere with its performance. In spite of these difficulties, this project will seek to help in

sustainable agricultural by offering a valid instrument of detecting the disease that can be available to farmers without developing any special knowledge and do not need high-tech facilities. The morphological operations to detect and categorize the symptoms of the diseases.

The aim is that of offer farmers especially in the resource poor regions with a tool that allows them to predict early identification of disease early enough in order to make an intervention and minimise the turnover of crops the proposed system provides a viable solution since it will not require complex it does not require machine learning models and it also does not require costly equipment, which is why it is a perfect candidate to use offline at a lowcost. Although the system has the potential of detecting various plants pathologies the weakness of image quality dependency and limited database of illnesses, and complications with complicated backgrounds can interfere with its performance. In spite of these difficulties, this project will seek to help in sustainable agricultural by offering a valid instrument of detecting the disease that can be available to farmers without developing any special knowledge and do not need high-tech facilities.

3. Tools and Techniques

Preprocessing of image, the first way is that the system carries out preprocessing of the image (by converting it into gray scale and) down scaling of noise and then it divides into colors and finds edges in the image filters. We also use texture as well as morphological operations to filter the image they use analysis in finding patterns of disease. Lastly, the system categorizes the plant as either diseased or healthy with the help of simple rules. Frontend and Backend Technology .The frontend stack was configured to make it simple responsive and easy to use interface. We used

- HTML: For structuring web pages.
- CSS: For styling and layout.
- Bootstrap: For responsive, mobile friendly design.
- JavaScript: To add interactivity and real-time.
- On the site, users are able to feed in plant image or upload files (e.g. PDFs, JPG) to be classified

3.1 Hardware Used with Technical Specifications

The equipment that was taken in the design and operation of the plant disease detecting. The system is not only cost effective but also available and more suited in the field. Image Acquisition Camera required: Canon 7D.

Exemplar: Webcam / Smartphone Camera (based on the availability of these to the user)

3.2 Software, Simulation Tools Used

The essential software tools were used to construct the system of plant disease detection. The image-processing algorithms were realized in Python, which was the main language. The backend web framework (Django) was used and presented a scalable and powerful environment. The VSCode was the creation that was used as the main IDE to write and debug the code. In case of simulation and testing, image datasets were subjected to classical methods of processing including a conversion to grayscale, filtering and edge detection. Also, OpenCV and Pillow (PIL) were used to handle and analyze images.

3.3 Visual Studio Code

Visual Studio Code (VS Code) is an IDE worthy yet lightweight code editor created by Microsoft, and it can work across various programming languages such as Python, Javascript and C++. It includes capabilities such as code completion with Intellisense syntax highlighting and support of the integration of Git. VS code is an extension marketplace personalization to fill special requirements, and its debuggers assist in locating and rectifying codes issues. It is cross platform that operates under Windows, mac OS and Linux. The editor's it can be easily used and the speed is fast, which makes it perfect both for a single team and developer. VS Code is freeware, it is open-source software, and has plenty of

customizations available this makes it quite a hit among the developers.

3.4 Django

Django refers to the open source, high level web framework being used in Python promotes haste and practical clean design. It offers a very strong framework of designing scalable and secure Web applications with in-built web tools authentication database and URL routing. Django is a batteries included project it also contained philosophy, it has numerous features that were shipped with it, including an admin interface, form processing, and security mechanisms against common vulnerability such as SQL injection cross site scripting. It has an architecture referred to as MTV (Model Template View) decouples the logic, data management and user interface, which makes it simpler to build and support complicated applications. Django is very flexible in design and can be extended effortlessly which enable programmers to build anything anywhere and out of this we get simple websites to big scale ones applications. It is a popular product that is adopted in the industry because of its efficiency capabilities, security features and community support as well as act.

3.5 Python

Python is a highly level, interpreted program language which is famous as ease of use and plain undecipherability. It concentrates on code readability it also enables programmers to write readable manageable and consistent code. Python is a programming language that facilitates multiple coding such paradigms as procedural object oriented and functional programming. Its high standard library and huge ecosystem of third party packages make it can be used in many applications inclusive of web development data analysis etc. Python is actually cross platform which implies that it can be executed in any platform. Some of the available operating systems include Windows, mac OS, and Linux. Because of its convenience to use it is at this place that people cannot help but admit the idea of chain taking a torturous path. Python has evolved to be one of the most popular since it is used widely flexible and has a good community support common languages that are popular by both amateur and professional people.

3.6 Qwen 2.5 72b via Open Router API

Qwen 2.5 72b Open Router may represent an API or platform, using which users may communicate with models such as Qwen 2.5.

APIs like this generally allow developers to integrate AI models into applications by sending data (e.g., text or queries) to the model and receiving responses (e.g., text generation or insights). The Open Router API would allow you to make requests to a hosted model (such as Qwen 2.5 72b) without needing to host it on your own infrastructure.

- API keys from the service that provides Open Router.
- An understanding of the API endpoints to send requests.
- Relevant information such as the format of the data (e.g.text), and what kinds of responses to expect.

3.7 Physical Device

Camera Module: Captures high resolution images of leaves. It can be a USB webcam or a camera attached.

The various tools and technologies used to build the plant disease detection system. Python is used as the main programming language of the backend development and image processing, and Java can be applied to logic control or other modules of the backend. The front end is built using the HTML, CSS, and Bootstrap to provide a clean and user-friendly interface where the user is given the opportunity to upload pictures of the plants and see the answers. The API links the front end and the backend forwarding the image data to the backend to be analyzed and give results of the detection. It has a rule based model that assesses leaf in a simple manner machine learning free images based on threshold values of colors edges and texture analysis. Comprehensively, all these tools are able to

achieve functionality integrated with a web efficient accessible and easily usable based plant disease detection platform.

4. Methodology

4.1 SDLC Models

SDLC of Plant Disease Detection System entails a sequence of well-ordered steps to make sure that the system is developed with efficiency and answers the needs of users. The process will start with the analysis of requirements, in which the needs the opinions of its users including farmers or agricultural specialists are collected, with a strain on developing a machine learning free image based system. Second should be the design, in which the system architecture is to be designed, such frontend (by using HTML, CSS, Backend (Python, Java) and interface of data stream between APIs (Bootstrap). During the during the implementation stage the system is made: image processing methods such as gray scale conversion, color segmentation and edge detection are coded with Python and even though Java can be utilized to conduct extra logic operations. The testing phase will make sure that all of the components interact well and the system detects the disease correctly symptoms. Upon effective testing, the system is installed either on local server, being used on a web platform or embedded onto hardware such as a Raspberry Pi to use in the field. The last stage is the maintenance stage which assures long term stability since bugs are repaired and other enhancements done to enhance features. This orderly procedure renders the system effective, strong and available.

4.2 Agile Model

Figure 1. Agile Model

Figure 1, show the model of agile from step 1 to 6.

4.3 Verification of Functional Requirements

The plant disease diagnosis system is in operation. It verifies the uploading of images to ascertain that the appropriate file formats are processed. The gray-scale conversion and noise reduction are carried out in the preprocessing phase to ensure that the disease-detection algorithms can detect the symptoms.

The user interface should be user friendly and the system should be able to classify the plants as healthy or diseased. Error-driven communication on the front and back end should be well-behaved and APIs must be well-behaved. This is done by ensuring that each of the functions is

assessed to check its accuracy, efficacy, and reliability.

Common diseases with recognizable symptoms can be detected through visual examination done by qualified personnel but is subjective and likely to be misdiagnosed during early stages of infection or by two or more diseases that have similar symptoms. Lab techniques such as microscopic analysis, culturing of pathogens and molecular techniques such as PCR are highly accurate, specific, but time-consuming, expensive, and impractical in large-scale or quick-need situations in the field.

The lighting, humidity, and stress of plants are other environmental factors that influence the appearance of the symptoms and complicate the correct diagnosis. Although non-ML methods are consistent, dependent on the expertise of trained personnel and appropriate equipment, they are not as scalable, fast, and predictable as systems based on AI or automated methods. Therefore, conventional techniques are still very useful in controlled conditions but not in large-scale or on-field processes.

4.3.1 Functional Requirements

The functional requirements of the plant disease detection system are based on the key user interactions and system responses. The system should enable the users to post pictures of the leaves of the plants, run the pictures through specific image-processing algorithms, and give the right classification of plant health (e.g., healthy, slightly infected, or severely infected). It must also enable the user registration, login and session control and show the result in a user friendly dashboard. The system should also be able to check the image formats and make sure that only similar files (e.g., JPEG, PNG) are sent to analysis.

Table 1: ***The integrated module will be provided with the following functionality.***

ID	Status	Priority	Description	Source
IAM-001	Initial	High	Process images to identify diseases using AI.	Supervisor
IAM-002	Initial	High	Provide confidence levels for disease detection.	Supervisor

Table 2: It will have an integrated module, which will provide the following functionality.

ID	Status	Priority	Description	Source
DDM-001	Initial	High	Store detailed information about diseases.	Supervisor
DDM-002	Initial	Medium	Allow updates to disease data by administrators.	Supervisor

Table 3. The following functionality will be provided in an integrated module:

ID	Status	Priority	Description	Source
RRM-001	Initial	High	Recommend treatments based on disease type.	Supervisor

Table 4. The integrated module will be offered with the following functionality.

ID	Status	Priority	Description	Source
RM-001	Initial	High	Provide detailed reports in PDF format.	Supervisor
RM-002	Initial	High	Export reports in PDF format.	Supervisor

Table [1- 4], show the integrated module will be offer the functionality of ID, Status, Priority, Description and Source.

4.4 Non-functional Requirement

The non-functional requirements ensure that the system is dependable, effective and secure. The app should be able to reliably identify plant

diseases in diverse conditions and be able to work even on the more affordable devices. It must offer a friendly user interface that is user friendly, particularly to non-technical users. The data of users and images posted should also be secured by the system. The platform must also be cross-brower and cross-platform and provide high response times in the case of real-time feedback.

4.4.1 Reliability

Plant disease detection without relying on machine learning relies on more ancient methods, such as visual inspection, lab tests and simple diagnostic kits, and its reliability depends on a number of factors. Visual inspection, when performed by trained experts, has the ability to identify common diseases with easily identifiable symptoms, but is a subjective method that may fail to identify early infections and misdiagnose similar appearing diseases. Laboratory tests (e.g. examination of tissues with a microscope, culturing of pathogens, running of molecular diagnostics) have high accuracy and specificity, but are very time-consuming and expensive, not applicable to large-scale or emergency field tasks. In addition, the appearance of the symptoms is prone to change under environmental conditions (light, humidity, stress of plants, etc.), which further complicates the accurate diagnosis. Non-machine-learning methods may be reliable when in the hands of competent individuals, and the right tools are employed, however their scaling, speed of delivered results, and reliability are lower compared to automated or machine-driven algorithms. As a result, the traditional approaches are effective under the controlled conditions but less effective when it comes to large-scale or on-field settings.

4.4.2 Performance

The precision of the plant disease detection methods that are not assisted is generally accurate in controlled settings but inapplicable in the field settings. PCR, ELISA, and culturing of pathogens are laboratory techniques with extreme high sensitivity (usually over 95%), and are considered the gold standard in the verification of plant diseases. Nevertheless, such methods are rather labor intensive, expensive, and impractical in detecting over large area, or in a fast manner.

Visual inspection is fast and does not cost a lot and is usually done in the field. It has a weakness of lower accuracy which is normally estimated to be between 60 and 85 percent. The factors which influence the accuracy are mostly the experience of the observer, the symptoms, and the environment. The pressures of supply and demand may also affect the process of the disease detection mostly at an early stage, when co-morbidity and environmental stressors are on the rise, predisposing to misdiagnosis.

Traditional solutions are not scalable. They are labour intensive and they cannot be used to monitor large areas of farms on a continuous basis. In general, though the traditional methods of finding the diseases of plants are a possibility, they will not be as fast, reliable, and scalable as the new method of the ring-light technology. Specifically, they are not reliable under some circumstances, slow, and do not respond to high-tech automated solutions.

The accuracy and specificity of the lab methods; microscopy, culture techniques, and PCR are very high. Nevertheless, they are time-consuming, expensive and, in most cases, not practical with large-scale or rapid fieldwork. Lighting, humidity, and stress of plants also affect the appearance of the symptoms, making correct diagnosis even more problematic. Non-ML techniques may be good when performed by a professional team in possession of the appropriate equipment. However, they are not as scalable, fast, and reliable as automated systems or ones with AI support. Consequently, the conventional approaches are useful in the monitored environments but not applicable in the mass or field studies.

4.5 Design

The design of the plant disease detection system is structured to be lightweight, modular, and user friendly. The system is based on a web based

interface where users can upload images of plant leaves for analysis. Once an image is uploaded, the backend processes it through several classical image processing stages to determine whether the leaf is healthy or infected.

4.5.1 Use Case Diagram

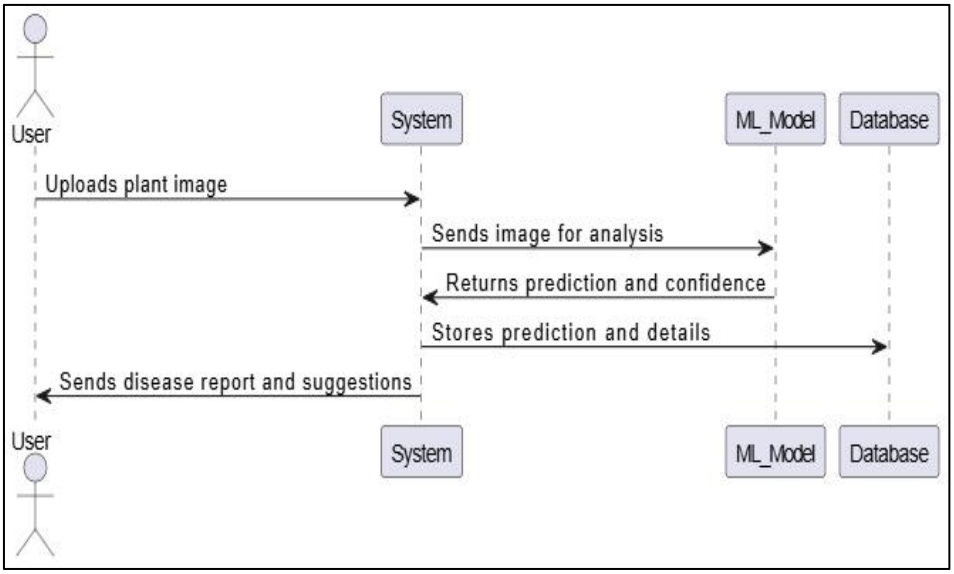

Figure 2. Use Case Diagram

Figure 2, show the case diagram in which system work.

4.5.1 Summary

The user uploads an image of a plant, which is transferred in the system and to an ML model to analyze. The model can produce a prediction and confidence score. This is stored in the system then reads a database and returns a text back to the user of its suggestions and a report on a disease.

4.5.2 Sequence Diagram

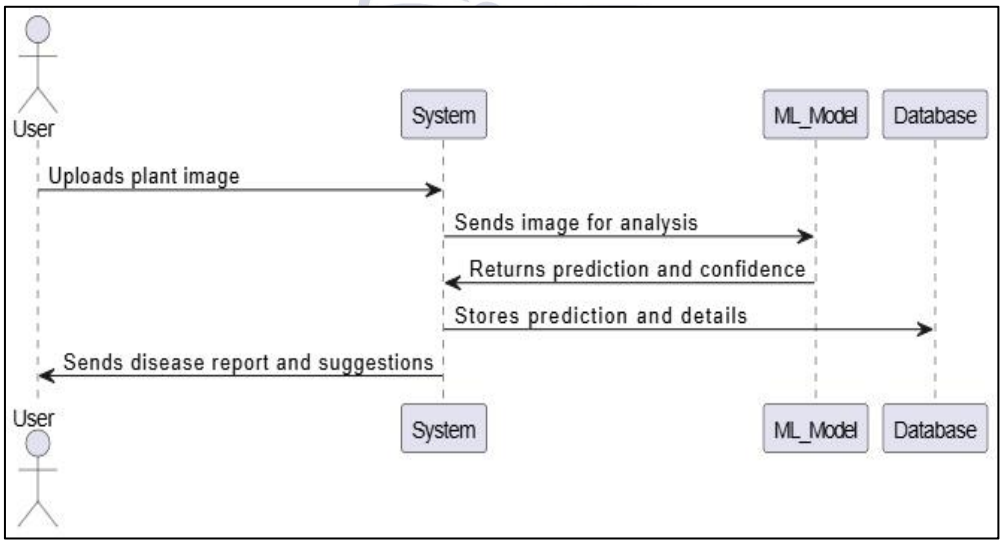

Figure 3. Sequence diagram

Figure 3, show the sequence diagram related to images analysis.

The user posts an image of a plant and the system transfers it to an ML model to analyze it. The model will predict and give a level of confidence. This is stored afterward by the system into a database and returns a report and recommendations of the disease to the user. The model analyses the image and obtains visual features such as color contrasts, texture, and the forms of lesions. It then compares these cues to patterns which it has learned. Based on this analysis the model will give the answer to a disease and the type of the disease in the plant.

The system also provides the confidence score or percentage probability that the system is certain about the prediction along with the diagnosis. Upon the prediction, the outcome, the level of confidence, and the input of the user is automatically stored in a secure database, which is to be accessed later by the user in reports or audits. The user gets a concise, organized report which contains the name of the disease (or indicates that the person is healthy) along with the confidence level and recommends treatment or follow-up. At some instances, the users also receive customized suggestions depending on the type of crops, climate, or the intensity of the problem. The system may optionally allow the user to download a PDF report or view disease history via their dashboard.

4.5.3 Activity Diagram

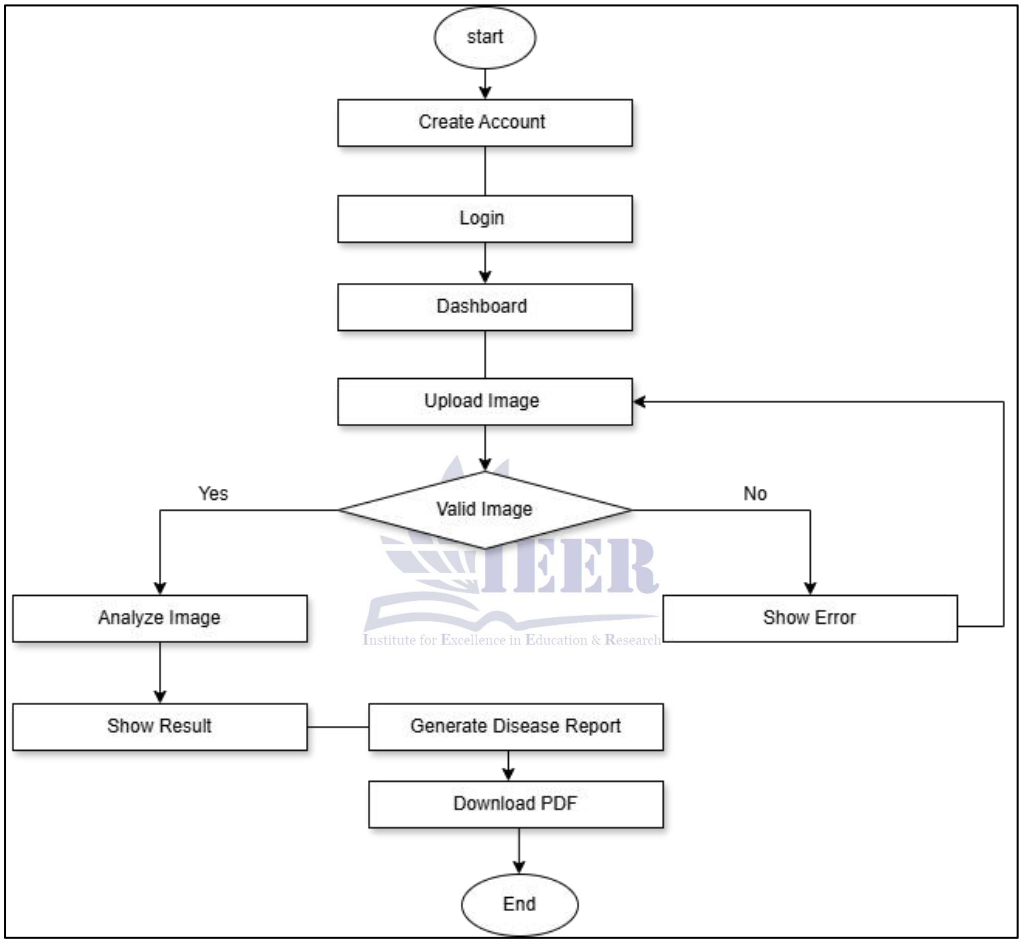

Figure 4. Activity Diagram of Admin

Figure 4, show the flow chart of activity of admin. In this flowchart the user journey through a plant disease detection system has been outlined. The procedure begins with the creation and log in after which the dashboard is accessed and pasting a plant picture. In case the image is valid, the system examines and shows collects the results prepares a disease report and provides the user to download it in PDF format. If the image is not valid, an error is presented. The model will predict and give a level of confidence. This is stored afterward by the system into a database and returns a report and recommendations of the disease to the user.

5 System Testing

What is system testing of a plant disease detection system the process of evaluating an entire system is called system testing use as an integrated and culminating product to make sure that it fulfills the given Performances needed and works under different conditions properly. This is testing phase confirms that everything that is supposed to be there right since the user account is being created plus his/her images and disease

The report and analysis go hand in hand. The functional testing guarantees that the functionality of every feature is as desired and usability testing as well as the interfacing of the system usability and the user interface. Performance testing is used to test how the system deals with heavy loads i.e. amplified images, a number of consumers simultaneously. Testing of security is also important and this will ensure that images and user data are not lost. It is safeguarded against intrusion. What compatibility testing does is to make sure that the application it functions equivalently over various products and browsers. Additionally, error handling is checked to ensure that the system gives some clear responses in case of invalid when there are inputs or when there are unexpected problems. In general, it is necessary to test a system. certify reliability, functionality and interactivity of the plant disease Any new detection system is subject to a pre-deployment, test and evaluation program.

5.1 Objective Testing

The objective testing in plant disease detection is a very useful tool in the process of the evaluation how people become aware of the health of plants meaning the identification, diagnosis control of different plant diseases. It provides an aligned and organized manner from the evaluation of the knowledge to the assessment, because the solution of the questions is determined according to strict, well-formulated standards. The kind of testing is especially valuable to an area such as agriculture. horticult ure and botany, in which diseases are early detected and properly identified play imperative roles in crop management and food security. Multiple choice in objective tests can be best applied in the context of the detection of diseases within the plant questions (MCQs) true/false statement, matching items and fill in the blank questions.

They are effective in the sense that they address major plants aspects. pathology, including the symptoms of plant diseases, a kind of pathogens that trigger the disease, the environmental conditions that help those pathogens spread, etc. they are made available to control measures. As an example, a normal MCQ can be to determine which pathogen makes a crop have a certain disease, or what symptoms are that are typical of a fungus such infection as the powdery mildew. These are the questions which can help to evaluate the level of individual in assuring. differentiate between various diseases as well as their memory behavior regarding facts accurately. The other critical point of objective testing is the fact that it tests knowledge concerning management of plant diseases. This may contain posers on cultural activities such as crop the patterns of rotation or health management of the soil application of fungicides, pesticides and organic Or it may be in the form of planting disease resistant varieties of plants, or exposure to disease resistant treatments. An example is the fill in the blank question where the user is required to give the name of a certain bacterium that causes which induces tomato wilt and the solution to this question would demand that one understands that the pathogen, on the one hand, as well as its effects on the plant, on the other.

Testing such diverse topics as the plant pathogen biology to on the ground management methods objective tests make sure that than people have not only theory of diagnosis but also some skills of it and the control of plant ailments. Moreover, these tests also determine the level of an individual concerning knowledge of the new techniques in the detection of disease in plant such as remote sensing, molecular diagnostics, or the utilization of agricultureartificial intelligence. Modern technology in the plant disease detection is increasingly coming to the fore and objective testing can also amount to inquiries regarding state-of-the-art diagnostic equipment that would help in early detection and accuracy agriculture.

Objective testing has many advantages in the detection of plant diseases. Since the responses are predetermined, marking is quick effective and uniform answering which all they measure the same criteria upon the participants. It is essential

to have this consistency particularly in educational and training environments in which one should consider a broad variety of individuals impartially.

Moreover, since the questions in these tests usually refer to personal ethics, one would not want to lose his country because of a procedure as simple as doing a test. do not have well-defined right or wrong answers, they eliminate bias and subjectivity with grading. It is also easy to administer objective testing to massive populations of students or professionals. which makes it a perfect tool of evaluation both in the academic and in the field-based assessments.On the whole, objective testing is very important in plant disease diagnosis since it guarantees that people are able to properly diagnose diseases, to know their causes and treat them good managerial policies. In the classroom, the research environment or in the field, these tests offer a credible approach to access the knowledge and readiness in fighting against plant diseases, which plays the major role in providing healthy crops and sustainable agricultural practices.

5.2 Usability Testing

Usability testing is an important technique to compare the extent to which the users can interact together with a product or a system in order to suit their needs and expectations. This testing relates to following actual users as they perform tasks when using the product understanding what challenges frustrations and confusion they can have. The end game is to obtain information on how simple and user friendly the design has been designers learn to improve intelligently. Usability testing may be in many forms such as moderated sessions or unmoderated and can either be virtual or face to face. Due to its focus on actual feedback of users it is likely to reveal problems in the early phases of the design process which lessens the danger of having to alter the product later at not minimal costs. Last but not least usability testing will guarantee that the user experience will be improved, more satisfied and the product will be more successful which relates to the needs of the target audience.

5.3 Software Performance Testing

Table 5. Software Performance Testing

Result Type	Completion Time (in seconds)	Bug / Error (if Yes give detail)
Dashboard Load	Success 1.8	No
AI Prediction Response Success	Success 1.4	No
User Login	Success 1.2	No
Fetch Trading History	Success 1.6	No
API Key Save	Success 1.3	No

5.4 Compatibility Testing

Table 6. *Compatibility Testing*

Testing Tool	Version	Compatible (Yes / No)
Google Chrome	121.0+	Yes
Mozilla Firefox	115.0+	Yes
Microsoft Edge	119.0+	Yes

5.5 Load Testing

Table 7. *Load Testing*

Test Environment	No. of Users	Sustainability
Physical Device	3	Yes
Emulator	5	Yes
Web (Live Test)	10	Yes

Simulated Users	20	Yes

Table [5-7], show the software performance, compatibility, and load testing.

5.6 Security Testing

Security testing of the detection of plant diseases is indispensable because of the rise in employment of digital technology like the use of IoT devices, mobile apps and AI based models in farming. The sensitive data dealt with in such systems tends to be collected and processed to the condition of crop health, its environmental conditions, and processes in the farms, being potential objects of cyber threats. Security testing is undertaken to make sure that these systems are guarded against unauthorized access data breach and manipulation. It is a process that entails confirming the integrity and privacy of data that has been obtained by sensors, to be ensured authorization and authentication systems, securing a communication channel Of being intercepted or tampered against. Also, the disease prediction AI models be challenged against adversarial attacks which may misguide the system to make incorrect diagnoses.

There is cloud storage and the databases with a high capacity of data storage misconfigured agricultural data, etc. should be evaluated against the vulnerability issues control by any permissions or un-encrypted informationAltogether, the security testing renders the plant disease detecting systems more credible, credible, and robust and promotes sustainable and safe farming methods. It is also very important in ensuring the safety of software through identification of vulnerabilities and threats that may be used by attackers. With increasing cases of cyberattacks and their advancement, there has been a need to safeguard applications, networks, and data by the organizations themselves. Security testing is used to test the system based on the issues of authentication, encryption, access control, input validation, and error handling.

In addition to enhancing the resilience of the system, efficient security testing of the system assists organizations to adhere to regulatory requirements and motivate users. It saves the reputation and financial stability of the business by minimizing the risk of information leaks and downtimes.

5.6.1 Focus Areas

Table 8. *Security Testing*

Test Type	Result (Pass or Fail)
User Authentication	Pass
API Key Protection	Pass
SQL Injection Prevention	Pass
XSS Protection	Pass

5.7 Test Cases

Table 9: *est Case 1 Sign In*

Test ID	**Khizar21@gmail.com**	**Test Name**	**Sign in**
Written By:	Khizar Hayat	Document Date:	5May2025
Software Name:	Plant Disease Detection	Test Level	Unit testing
Test objectives	Check for successful sign in		
Test environment	Windows 10, Python 3.8+, Django 4.0+		
Test input	Valid Email and Password		

Pre-condition	User must be registered in the system
Step 1	Access sign in page
Step 2	Enter email and password
Step 3	Click on sign in button.
Expected result	User Successful sign in if data is validate, if data is not valid error show please enter your valid information.
Post condition	Successfully sign in
Actual result	User must be successfully login after providing valid email and password
Status	Success

Table 10: *Test Case 2 Login*

Test ID	**Khizar21@gmail.com**	**Test Name**	**Login**
Written By:	Khizar Hayat	Document Date:	5May2025
Software Name:	Plant Disease Detection	Test Level	Unit testing

Test objectives	Verify new user can login successfully
Test environment	Windows 10, Python 3.8+, Django 4.0+
Test input	Email and password.
Pre-condition	User must not be login
Step 1	Navigate to login page
Step 2	Fill login page
Step 3	Submit page
Expected result	Supervisor successfully download, upload and update the information.
Post condition	Successfully login
Actual result	User account created and verification email sent
Status	Success

Table 11. *Plant disease analysis*

Test ID	**Khizar21@gmail.com**	**Test Name**	**Login**
Written By:	Khizar Hayat	Document Date:	5May2025
Software Name:	Plant Disease Detection	Test Level	Unit testing

Test objectives	Verify plant disease detection functionality
Test environment	Windows 10, Python 3.8+, Django 4.0+
Test input	Plant leaf image (JPEG/PNG)
Pre-condition	User must be logged in
Step 1	Navigate to dashboard
Step 2	Upload plant leaf image
Step 3	Click analyze button
Expected result	Disease analysis results displayed

Post condition	Analysis results shown with treatment recommendations
Actual result	Disease detected and recommendations provided
Status	Success

Table 12. *Image format validation*

Test ID	Khizar21@gmail.com	Test Name	Login
Written By:	Khizar Hayat	Document Date:	5May2025
Software Name:	Plant Disease Detection	Test Level	Unit testing

Test objectives	Verify system validates image formats
Test environment	Windows 10, Python 3.8+, Django 4.0+
Test input	Invalid file format (e.g., .txt).
Pre-condition	User must be logged in
Step 1	Navigate to dashboard
Step 2	Upload invalid file format
Step 3	Click analyze button
Expected result	Error message for invalid format
Post condition	System rejects invalid file
Actual result	"Please upload a JPEG or PNG image" message shown
Status	Success

Table 13. *User registration*

Test ID	Khizar21@gmail.com	Test Name	Login
Written By:	Khizar Hayat	Document Date:	5May2025
Software Name:	Plant Disease Detection	Test Level	Unit testing

Test objectives	Verify new user registration process
Test environment	Windows 10, Python 3.8+, Django 4.0+
Test input	Username, Email, Password
Pre-condition	User must not be registered
Step 1	Navigate to signup page
Step 2	Fill registration form
Step 3	Submit form
Expected result	Account created and verification email sent
Post condition	User account created

Actual result	Account created and verification email sent
Status	Success

Table 14. *Password reset*

Test ID	Khizar21@gmail.com	Test Name	Login
Written By:	Khizar Hayat	Document Date:	5May2025
Software Name:	Plant Disease Detection	Test Level	Unit testing

Test objectives	Verify password reset functionality
Test environment	Windows 10, Python 3.8+, Django 4.0+
Test input	Registered email address
Pre-condition	User must be registered
Step 1	Click "Forgot Password"
Step 2	Enter email address
Step 3	Submit form
Expected result	Password reset email sent
Post condition	Reset link sent to email
Actual result	Reset link sent successfully
Status	Success

Table 15. *Session management*

Test ID	Khizar21@gmail.com	Test Name	Login
Written By:	Khizar Hayat	Document Date:	5May2025
Software Name:	Plant Disease Detection	Test Level	Unit testing

Test objectives	Verify session handling after logout
Test environment	Windows 10, Python 3.8+, Django 4.0+
Test input	None
Pre-condition	User must be logged in
Step 1	Click logout button
Step 2	Try to access dashboard
Step 3	browser back button

Expected result	Redirected to login page
Post condition	Session terminated
Actual result	Successfully logged out and redirected
Status	Success

Table 16. *Email Verification*

Test ID	Khizar21@gmail.com	Test Name	Login
Written By:	Khizar Hayat	Document Date:	5May2025
Software Name:	Plant Disease Detection	Test Level	Unit testing

Test objectives	Verify email verification process
Test environment	Windows 10, Python 3.8+, Django 4.0+
Test input	Verification token
Pre-condition	User registered but not verified
Step 1	Click verification link in email
Step 2	System processes token
Step 3	Check status verification
Expected result	Account verified
Post condition	User can now login
Actual result	Account successfully verified
Status	Success

Table 17. *API Response time*

Test ID	Khizar21@gmail.com	Test Name	Login
Written By:	Khizar Hayat	Document Date:	5May2025
Software Name:	Plant Disease Detection	Test Level	Unit testing

Test objectives	Verify API response time for analysis
Test environment	Windows 10, Python 3.8+, Django 4.0+
Test input	High-resolution plant image
Pre-condition	User must be logged in
Step 1	Upload large image
Step 2	Start analysis
Step 3	Measure response time

Expected result	Response within 10 seconds
Post condition	Analysis completed
Actual result	Response received in 8 seconds
Status	Success

Table [8-17], show the security, test cases, plant disease analysis, image analysis validation, user registration, password reset, session management, email verification, and API response respectively.

6. Result and Conclusion

Contemporary agriculture involves the need to detect diseases in crops. It assists farmers to detect diseases in good time to enable them respond in the most timely manner to save their crops.The latest improvements in artificial intelligence, computer vision, and mobile applications ensure that it is increasingly simpler to diagnose diseases with the help of a mere photo of a plant. They are fast and handy to operate and can be still used even in distant places. They therefore assist the farmers to save on time, lessen the damage of crops, and minimize the application of chemical treatments. Such systems will only get better in the future, and the process of farming will be more productive and sustainable.

6.1 Presentation of the Findings

The agriculture industry today is faced with the need to feed an increasing population but with a challenge of climate change, pests, and plant diseases. Among the most significant issues in the contemporary agriculture, early identification and management of plant diseases nowadays is a priority. The uncontrolled diseases may lead to colossal losses of crops, reduced food quality, and compel farmers to apply more chemicals as pesticides. Thus, scalable, correct, and timely disease detection solutions are now more than ever. In the past, farmers had to use visual inspection and consultation of experts to determine the diseases in their crops. This was a manual process that could easily be misplaced or lost and was slow and human error prone and could not be accessible to the farmers who were small or in rural areas because they did not have contacts with agricultural experts. When a disease was realized, it was frequently too late and turned into a nationwide epidemic with irreversible consequences and loss of money.

The disease detection and management is now revolutionized by artificial intelligence (AI), computer vision, and mobile technology. These innovative technologies will allow farmers to detect diseases of plants, precisely and without the assistance of an expert, at the earliest stage. As an example, a basic smartphone snapshot of a leaf enables AI-based applications to diagnose the presence of discoloration, spots, lesions, or wilting to indicate bacterial blight, powdery mildew, or leaf rust, etc. The deep learning algorithms that are trained on large plant image datasets identify complicated patterns and match them to familiar disease signatures to offer real-time diagnoses and in many cases, prescribe treatment or preventive action. These technologies are fast, convenient and portable particularly in the rural or remote areas where skills are minimal. Digital tools of disease detection also minimize the wasteful use of pesticides. Chemicals are frequently used by farmers as an insurance measure, when a disease is unconfirmed. This adds expenses and threat to the environment and human health. Proper identification enables farmers to use chemicals when and where needed, which helps in keeping farming sustainable, and reduces the environmental impact without affecting the crop health and productivity.

Intervention is a very important aspect especially at the early stage of the disease. The infection at early stages is simpler and less expensive to treat. The disease can be prevented before shifting to other plants or fields thus saving more crop areas and preventing the destruction of large yields. This is an early warning system that ensures food

security, particularly in areas where the main source of income is agriculture. Online systems and data analytics improve the process of tracking and reporting diseases. Combined databases allow farmers to share outbursts, pest infestation, and weather conditions. This shared intelligence is available to the local or national agencies so that they can respond and plan accordingly.

The capture will be even enhanced by future implementation of the Internet of Things (IoT), including drones, smart cameras, and sensors. The gadgets have the capability to scan crops automatically to identify disease and send the information to cloud-based systems to judge it. Together with the GPS they can map the infected areas and direct the precision farming. Such systems will continue to be developed in the next few years; they will be more sophisticated, cheaper and will be accessible to all farmers regardless of the size of their operations. With the improvement of technology, a better detection tool will be developed, databases will be more extensive, and the accuracy of diagnosis will be even higher.

To sum up, the AI, computer vision, and mobile app-based modernization of the plant disease detection is a big step towards more productive, resilient, and sustainable farming. These tools enable the farmers to do things quickly, lessen their losses and apply inputs more productively. They also make crops healthier and food security is guaranteed. With the development of innovation, technology will remain a key feature to revolutionize agriculture to benefit the farmers and the environment.

6.2 Software Results

The created plant disease detection system manages to accomplish the main functionalities of the uploading of images to automated diagnosis and reporting effectively. The software was also tested and proved to be consistent and stable across various devices and browsers. People could post pictures of leaves, and the system worked out with classical image processing methods, e.g. conversion to grayscale, edge detection, and colour segmentation. The system was put to test using diverse sample leaf images which indicated healthy plants, mildly infected plants, and severely infected plants. In the majority of instances, the system correctly detected the signs of the visible disease, determined the severity, and offered a clear output to the user interface. The dashboard showed the results in categories (e.g., a healthy, mild infection, severe infection) and an opportunity to get the disease report in PDF format. Other characteristics like user login, validation of image format, and error management were also tested and worked well. The API communication between the frontend and backend was smooth and the average time of response to analyze images was less than 10 seconds even with the middle-range devices.

6.2.1 Splash Screen

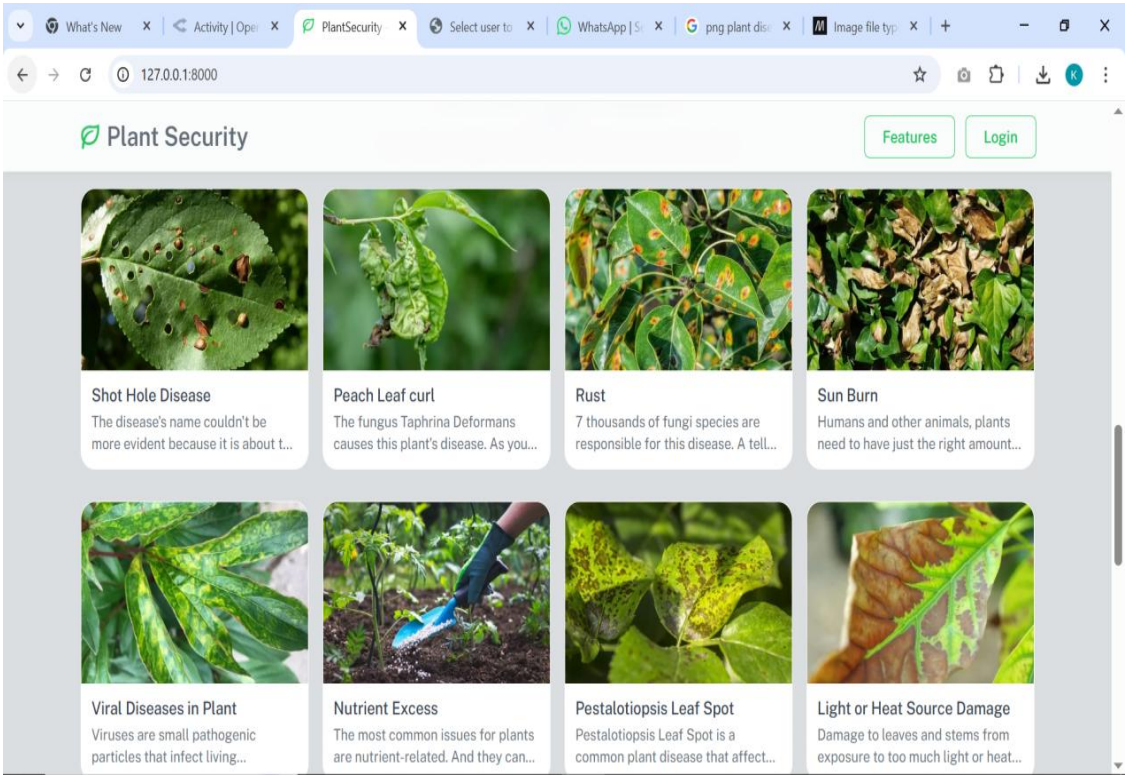

Figure 5. Splash Screen

Figure 5, describe the splash screen in which show the plant disease.

6.2.2 Admin Sign in

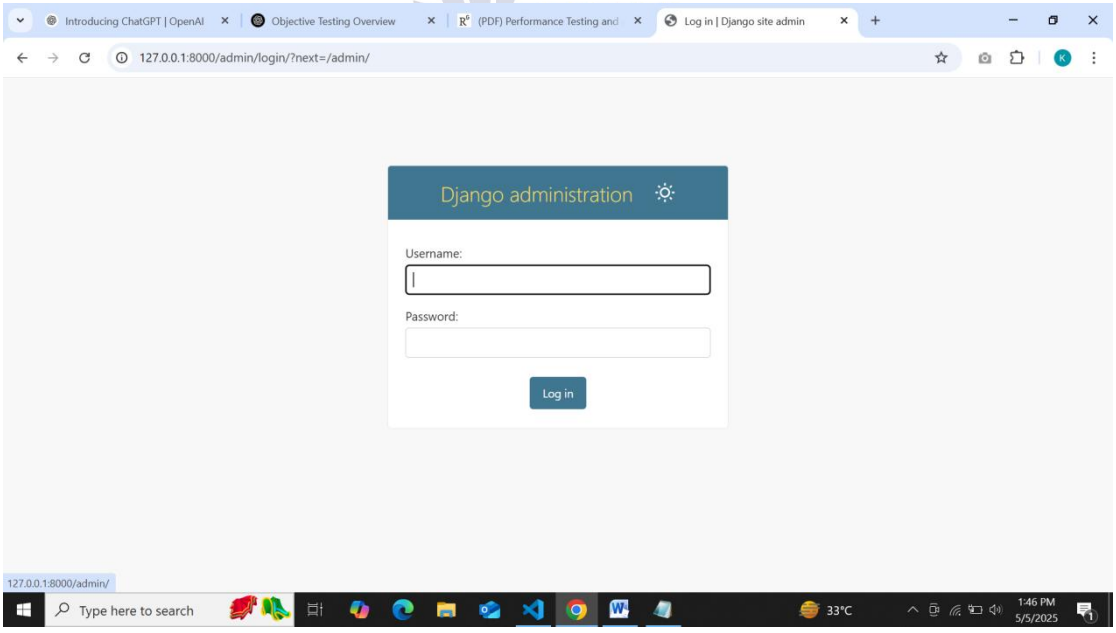

Figure 6. Admin Sign in

Figure 6, show the screen about admin sign in.

6.2.3 User Login

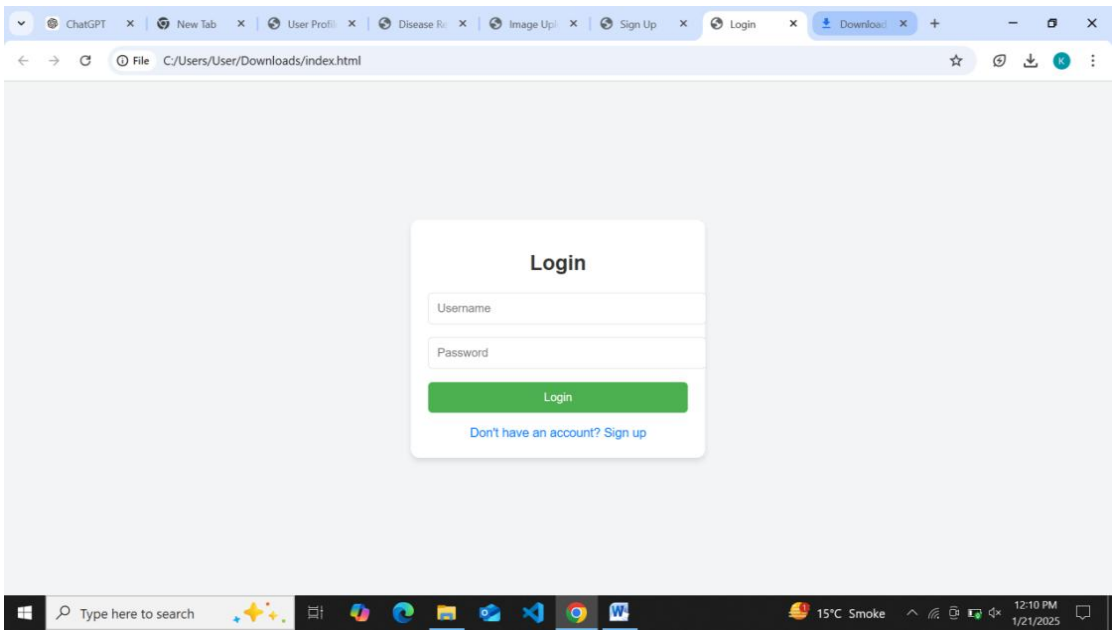

Figure 7. Student Login

6.2.4 Admin Dashboard

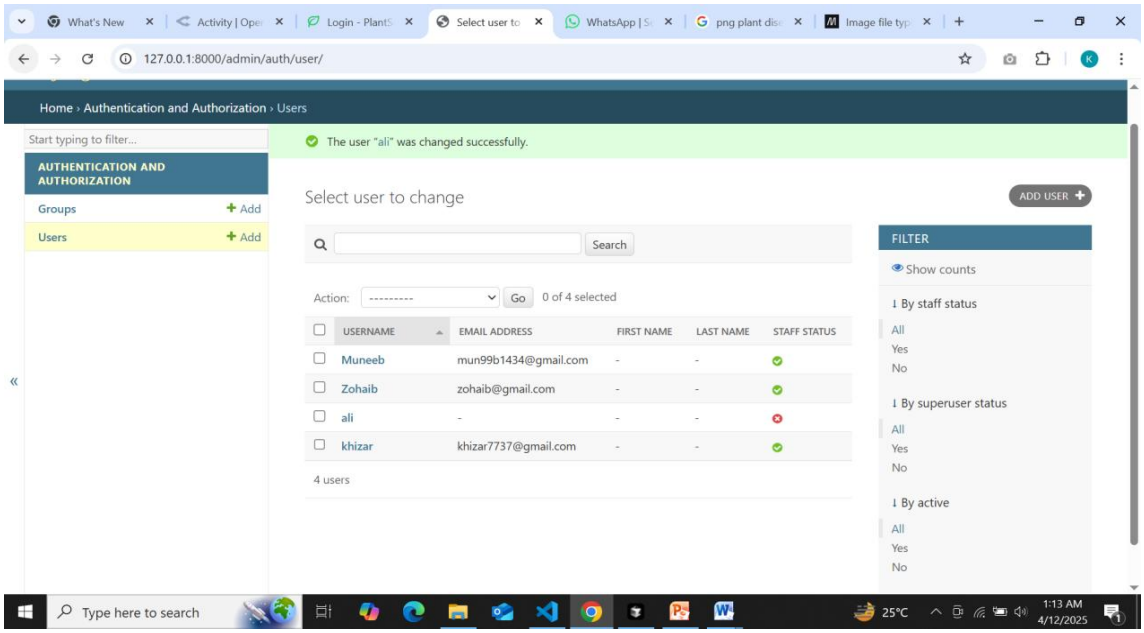

Figure 8. Admin Dashboard

6.2.5 Dashboard

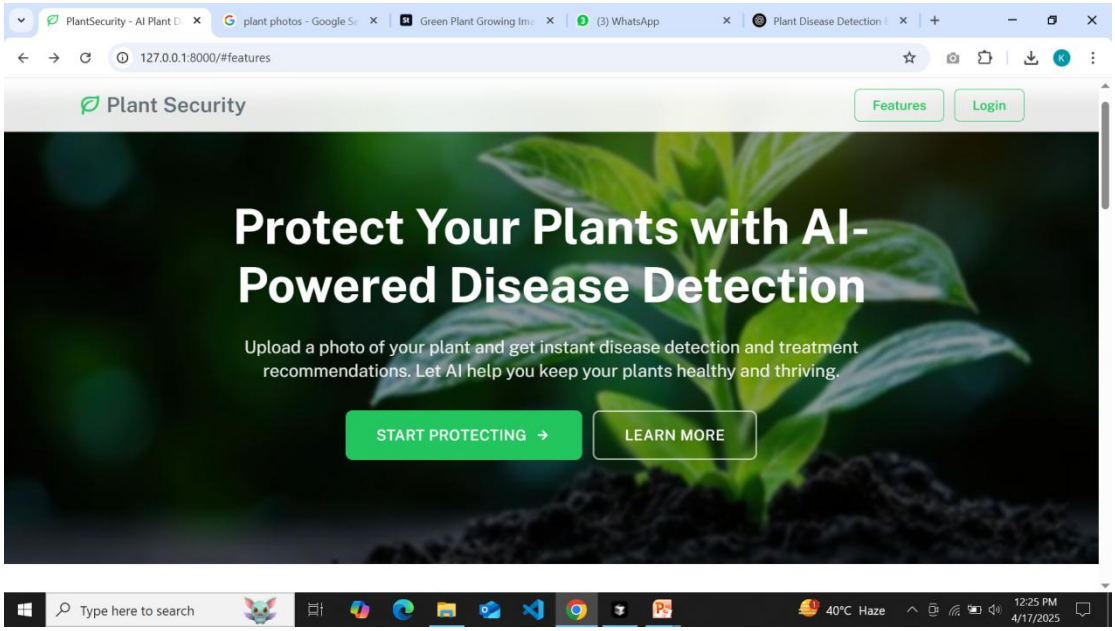

Figure 9. Dashboard

Figure [7-9], show the student log in screen, admin and dashboard.

6.3 Discussion of the Findings

According to the new tendencies in the evolution of the process of plant disease detection, its results in the image of the technology are transforming the face of the agriculture. One of the tools is artificial intelligence (AI). Machine learning can be used to detect plant diseases and image processing is fast and efficient. This is in a long way as compared to the traditional methods where one would have to toil and at times even apply specialization in detecting images The level of accuracy of image-based image detectors is high. Specifically, AI models based on deep learning algorithms, namely convolutional neural networks The CNNs, can detect diseases symptoms through leaves of plants at 90 percent accuracy. This high level of accuracy will help in eliminating misdiagnosis and the farmers will take the right action at the right time. The other critical element is the worth of mobile applications. This application opens the farmer to be in a position to diagnose an illness simply by the capturing of a photograph of an ailing plant. This is it is especially valuable in rural locations where agricultural experts are not so readily available.

These Treatment suggestions are also provided by the apps, and this enables the farmer to make fast judgments. revent damages and losses, and save crops. The study also reveals the importance of a sensor based system. By monitoring these systems are sensitive to environmental factors such as temperature, humidity, as well as the moisture of the soil, can even tell in advance the risks of an outbreak of the disease before its manifestation. This assists in preemptive preparations, which is cost-effective and effective than responsive actions. a complete outbreak Regarding challenges, these technologies may have problems with precision yet through the fact that their images are poor or have atypical symptoms or rare conditions that have low awareness. reflected by training data. Moreover, not all the farmers may access them without hindrances. tools because of costs, the availability of the net, or the person does not have technical skills. On balance, the results imply that the technologies of the disease detection of plants can be found. It is getting more reliable and accessible. As the improvements are still being made and extended, wider Through adoption, these tools will be vital in bringing up food production and minimbasing it use of chemicals, promotion of sustainable agricultural activities, and so on.

6.4 Comparison to Original Objective

The original aim of the undertaking was to conceptualize and actually come up with the system that could of effectively diagnosing plant pathology by means of machine learning based on images ideally, it should be accessible through mobile gadgets so it is simple to the farmers. The last system obtained high accuracy in the identification of several plant diseases and had adequate results in testing conditions. The real time detection and easy to use interface was aimed at have been fulfilled to a large extent especially with regard to the creation of a prototype mobile application which enabled the user to post pictures of the plants and get results immediately. This testing relates to following actual users as they perform tasks when using the product understanding what challenges frustrations and confusion they can have. The end game is to obtain information on how simple and user friendly the design has been designers learn to improve intelligently. Usability testing may be in many forms such as moderated sessions or unmoderated and can either be virtual or face to face. Due to its focus on actual feedback of users it is likely to reveal problems in the early phases of the design process which lessens the danger of having to alter the product later at not minimal costs. Last but not least usability testing will guarantee that the user experience will be improved, more satisfied and the product will be more successful which relates to the needs of the target audience.

6.5 Reasoning for short comings

Even though the system was successful, certain flaws were realized. The major problem was that dataset size and variety was very small and this hampered the performance of rare or less-observable diseases. Images in real-world situations, including low lighting, background clutter, or partly infected leaves also reduced the accuracy of the model using controlled training scenarios. Also, the associated hardware limitations (e.g., smartphones that had limited processing capabilities) did not allow using more elaborate models that could have further enhanced performance.

The success and the functionality of the system were evident as a whole, but there were several limitations coming forth in the course of implementation and testing. The limited size and diversity of the dataset was one of the most important challenges. Restricted dataset implies that the system has been trained and tested on a comparatively small range of disease symptoms, type of plants and environmental conditions. The other problem of great concern is the quality of real world images taken by users. In real world, field situations, images are most of the time captured when the lighting is poor, when not all the parts of the background can be captured, or when leaves are partially covered or ruined. These real world variations create noise and visual complexity into the system that the system might not have been exposed to during development, decreasing how accurate and reliable the detection of disease can be.

As an illustration, false positives or missed diagnosis of disease can take place because shadows, soil patterns, or overlapping leaves can appear to copy or hide diseases symptoms.The system was also limited by hardware limitations particularly on mobile gadgets like low-end smart phones. Smaller-ram devices, low-speed processors, or cameras with a lower resolution will not be able to execute more complicated image-processing algorithms or real-time high-fidelity analysis. This limitation also restricted the possibility to incorporate more advanced models or improvements that could have resulted in a better detection accuracy, e.g., higher-order texture analysis or multi-angle image input. Together, these challenges underscore the need to have larger, more representative datasets, be able to support a variety of image conditions, as well as to develop further to be able to optimize performance on a wider variety of devices. These deficits will be critical to the enhancement of the efficacy and extensibility of the system in practical farming settings.

6.6 Limitations

There were a number of limitations that came out during the project. Firstly, the model can only generalize on crops and diseases whose cases are given in its training cases. Second, the system has image clarity as a requirement; it is poor with low light or when the camera is poorly focused. Third, the prototype needs the internet in order to upload the images to the cloud, which is a limitation at rural or remote farms. Finally, the system anticipates disease type, but does not treat the disease effectively and environmentally friendly, or provide prevention guidance that is stage-specific.

The system has a potential, but there are a number of limitations that can curtail its wider use and success. These are the challenges that require future upgrading and enhancement. To start with, the accuracy of the system was limited due to the small variety of crops and diseases used in the training set. It can therefore not work well on invisible species or rare diseases, lowering the extent of generalizability unless the data is enlarged and the model re-trained. Second, high-quality images are of great importance in the system. Poor outdoor or field lighting, shadows, motion blur or low-resolution cameras may cause poor image quality, which may result in misidentification or missed symptoms. This is a weakness of reliability in rural locations which do not have upscale imaging facilities. Thirdly, the prototype requires access to the internet in order to upload images to the cloud to process them. In remote or underdeveloped agriculture areas where connectivity is poor, slow, or nonexistent this need constrained practicality to an extreme. Finally, the system predicts diseases, but does not provide effective treatment and preventive recommendations. In the absence of any actionable recommendations, like environmentally sustainable measures or growth-stage specific measures in relation to crop pesticide application, the user farmers might not be in a position to make decisions about pesticides application, irrigation diet, or prevention.

Finally, although the system has great potential to detect early onset of plant diseases, generalizability, dependence on image-quality, internet, and absence of built-in treatment recommendations are its current limitations and should be developed. Target solutions to these problems will be major towards improving effectiveness, usability and impact of the system, especially among the farmers in low-resource or remote agricultural groups.

6.7 Recommendations

To enhance the system, one should gather a bigger and varied dataset that contains pictures of some typical environments in real world and under lighting conditions. Training model to have a deeper learning accuracy could be improved using architectures. One more thing to be suggested is to incorporate multi language add offline features and support of mobile application to make it friendlier to they manage to reach farmers in rural areas. Through working with agricultural experts, it may be possible to incorporate some elements that may add to agricultural success correct curative and preventive information in the application. Finally, integrating prediction accuracy could be enhanced also by sensor data (such as heat and moisture readings) and assist in the identification of the risk of disease before the onset of symptoms.

6.8 Conclusion

To sum it up the plant disease detecting system leads to a successful process of meeting the necessary objectives of capturing the plant diseases correctly and provide real-time response to the user. There are certain drawbacks, in particular associated with the diversity of datasets and nature of the environment accordance the prime aim of offering an efficient and convenient instrument of the illness. Detection was done. This system would have an enormous improvement in using AI to contribute to sustainable agriculture as well as to prevent

diseases that cause loss of crops. In conclusion, plant disease detection system achieved its major objectives. It correctly diagnoses plant diseases and gives appropriate feedback to the consumers in an automated and picture-based process. Using the classical image-processing methods, the system provides a highly useful low-cost device, which allows farmers and agricultural laborers to identify the signs of disease without costly equipment and sophisticated knowledge. Nevertheless, the system does not have no boundaries. The large number of plant diseases and environmental conditions is a challenge. Its performance is based on quality and variety of the dataset used in the development.

When the data set is not extensive in terms of plant types, symptoms or situations, its detection accuracy may decline, particularly on rare cases. Real-world reliability is also compromised by other effects such as lighting, interference of the background, and incomplete visibility of infected leaves. Notwithstanding such problems, the system offers a good baseline of the future developments. By incorporating artificial intelligence and machine learning, it may be able to learn on a large scale of data and change its adaptive patterns to complex disease trends over time. The kind of improvements would enhance accuracy, increase the scope of diagnosable illnesses and make the system smarter and stronger. Finally, the existing system would be a significant move to sustainable agriculture. It allows the detection of diseases at an early stage, loss of crops is minimized as well as the use of unnecessary chemical interventions. As the intelligent technologies become more and more developed and integrated, it has tremendous potential to empower the farmers even more and play a significant part in the global food security and agricultural resilience.

6.9 Summary

This project was meant to create plant-illness-detection system with machine-learning and generated a functional model that could recognize common plant pathogens. Its major achievements have been mobile-friendly interface, large detection rates and good usability. Nevertheless, the project suffered because it used a small dataset, real-life variability, and no offline support. It may be a strong weapon to farmers with the suggested improvements that would make it a powerful tool to enhance productivity and healthy agricultural practices. Poor lighting, background noise, or partially infected leaves are also examples of image quality problems that reduced the accuracy of the model working with controlled training data. Hardware limitations, such as the low processing capability of smartphones, prevented the use of more complex models that could have led to an even better performance.

The last system obtained high accuracy in the identification of several plant diseases and had adequate results in testing conditions. The real time detection and easy to use interface was aimed at have been fulfilled to a large extent especially with regard to the creation of a prototype mobile application which enabled the user to post pictures of the plants and get results immediately. This testing relates to following actual users as they perform tasks when using the product understanding what challenges frustrations and confusion they can have. The end game is to obtain information on how simple and user friendly the design has been designers learn to improve intelligently. Usability testing may be in many forms such as moderated sessions or unmoderated and can either be virtual or face to face. Due to its focus on actual feedback of users it is likely to reveal problems in the early phases of the design process which lessens the danger of having to alter the product later at not minimal costs. Last but not least usability testing will guarantee that the user experience will be improved, more satisfied and the product will be more successful which relates to the needs of the target audience.

7. Future Work

7.1 Future Work in Software Performance Testing

Testing of software performance is a very critical part in the measurement of how effective a software is application works under different conditions particularly when it is exposed to huge traffic huge amounts of data or heavy user loads Significance of performance testing is also increasing with the users wanting software that is of good quality, fast and efficient products that could fulfil the needs of a world which is increasingly digital connected. In on this backdrop, future research in software performance test is very important to secure the fact that the techniques, tools, and methodologies involved in opposing software are ever changing cope up with new challenges of software development lifecycle. The emergence of more complex, distributed and dynamic modern application software will change the process of software performance testing significantly. With the implementation of microservice, containerization, and multi-cloud or hybrid configurations in business, performance testing has to conform to emerging challenges.

The AI and machine learning in the future will be applied to generate tests automatically, detect anomalies, and forecast performance issues. Continuous performance testing will be integrated, a part of DevOps pipelines to ensure that a team is provided with quick feedback and bottlenecks are discovered at a very early stage. The testing tools will also require the possibility to test real situations on other cloud platforms, so that the app will act similarly under different circumstances. As the IoT and edge computing grows, the performance testing will reach low-power equipment and remote locations to ensure efficiency and reliability.

More emphasis will be laid on user-centric testing as well where real user experience is measured as opposed to just system measures. In the future, the practices will be more focused on energy efficiency and green testing. All in all, these modifications will make performance testing smarter, faster, more flexible, and something that is necessary to provide high-quality, resilient software.

7.2 Future Work in Performance Testing Methodologies

The development of software performance testing techniques has radically transformed the way these techniques change with time. The future work will probably be devoted to the improvement of these methods in order to ensure more perfect, effective, and complete testing. Some of the strategies used in the current performance testing techniques include load, stress, scalability, reliability and endurance testing among others, which are meant to test various software performance aspects. Performance testing has over the years developed in relation to the increased complexity and demands of the modern application. The more dynamic, distributed and resource-intensive systems are, the more traditional methods need to be improved so that they could still be effective. This development has resulted in a set of methodologies that guarantee applications to be quick, scalable, dependable and responsive to diverse workloads. Performance testing is an umbrella term that encompasses various specialized strategies which target a particular behavior of an application. The most common ones are load, stress, scalability, endurance, and reliability testing, which all measure the performance of a system in light of expected or extreme or sustained use.

Load testing: Load testing is done to simulate normal to heavy user load and the behavior of the application on such loads. It determines the level within which the system can handle before the performance starts to lower. Stress testing breaks the limits of the capacity of an application so that there is a point where it breaks down. It is useful in finding out how the system will act in extreme conditions and it also helps in finding out how the system will recover after an occurrence of failure. Scalability testing is used to

determine how a system can be scaled to operate efficiently considering the user base or data volume that can be increased or reduced. This is vital in cloud-based or microservices designs that depend on resource allocation which is elastic.

• Soak testing also referred to as endurance testing is used to test the performance of a system under prolonged conditions. It reveals problems like memory leakages or loss of performance which might not be evident in short sessions.

• Reliability testing evaluates the stability and the consistency of an application in different circumstances to facilitate its non-interrupted functioning.

With the development of the field, the further development is likely to focus on automation, AI-based analysis, real-time monitoring, and the connection with CI/CD pipelines. These trends will render testing to be more proactive, accurate and scalable where developers are able to identify and fix any performance problems earlier in the development process. To conclude, performance testing is no longer a manual and post-development process; it has become part of a modern strategy of software engineering. Otherwise, further innovation will be needed to provide efficient, seamless, and reliable user experiences irrespective of scale and intensity of use.

7.3 Integration of AI and Machine Learning in Performance Testing

Among the opportunities that are immense as to future work, artificial incorporation should be listed apply intelligence (AI) and machine learning (ML) algorithms on performance testing tools. Today there is a common problem in that performance testing usually needs manual setup and configuration that may consume time and be subject to human errors. With the help of AI and ML technologies many of these steps in the future may be automated by use of future tools that can give a real time analysis of performance data, locating where performance may be held up, and performance projection address concerns before they happen as an example AI algorithms might use past performance data and give AI generated data an understanding of how the system may behave in general and under varying circumstances which enables one to solve performance problems in advance before they turn fatal we also may improve the load test with the help of ML algorithms as it would be done automatically tailoring the test scenarios according to the real-time scenarios and the way the system would react taking consideration of optimization efficiency of tests and more realistic and fuller testing.

Among the most promising opportunities for future advancement in performance testing is the incorporation of Artificial Intelligence (AI) and Machine Learning (ML) algorithms into testing tools. Handset-up, configuration, and interpretation of test results is currently one of the greatest problems in the performance testing. This can be quite time consuming, labour intensive and mainly subject to human error. With the combination of AI and ML, most of these manual processes would be computerized to develop smarter, faster, and more precise testing processes.

The future tools of performance testing that are AI-powered might have real-time analysis of the performance of the system data. They would automatically detect points of bottlenecks, anomalies or points of performance degradation as they arise. Instead of using only the static test scripts and the pre-programmed scenarios, AI-based tools may use historical performance data as the source of learning. They would forecast the behavior of a system to be exhibited in different circumstances- such as unexpected surges in traffic, hardware breakdowns, or unexpected usage trends. Such predictive power gives an opportunity to solve problems in advance as possible performance issues are addressed before they manifest as serious failures.

Moreover, the ML algorithms may include the ability to dynamically alter and optimize load testing situations on-the-fly. They would also make test parameters more realistic in regard to

actual user behavior. This gives a more real-life testing environment which enhances the reliability and relevance of test results. As an example, in a system that is continuously integrating and deploying, ML-enhanced tests can be helpful, which will change with every new code change, infrastructure, or usage pattern. This will guarantee the uniform levels of performance in all updates.

The resource allocation in the course of tests, as well as the elimination of redundancy, and concentration on the most effective areas of performance, can be simplified with the help of the optimization features offered by ML, as well. Basically, the use of AI and ML in performance testing enhances automation and efficiency. It also increases the strategic importance of testing, making it a reactive (quality assurance) process instead of a proactive, intelligent performance-management process.

7.4 Real time Performance Monitoring and Automated Scaling

The enhancement of real time performance can also be called another prospective developmental area monitoring especially regarding more complex dynamic systems. As cloud the more common computing and microservices architectures get the more real-time monitoring and automated scaling are more relevant. Future monitoring systems could be connected quite easily to performance testing tools they can give ongoing feedback on the performance of the system even post deployment. Another interesting technology is automatic scaling which is in accordance with hard history data on performance research. In our world in a great number of systems scaling up and down has to be done manually in reaction to different loads. Nevertheless, through integration into the realm of automated scaling, a range of beneficial functions of the platform can be achieved performance testing technology soft wares would be able to be dynamic to changing demand efficiently controlling the usage of resources without the intervention of human hands.

This would especially be the case when proves viable in cloud based systems, where user traffic and workload are usually variable patterns. With the increasingly complex nature of modern software systems, particularly in the context of cloud computing, containers, and microservices, it is easy to achieve real-time performance monitoring as well as automated scaling. The continuous monitoring of things such as CPU utilization, memory consumption, response time, network I/O and latency amongst others will enable the system administrator and DevOp team to monitor the health, performance and responsiveness of the applications in production. The current tools do not just give warnings, but they also identify anomalies and anticipate performance problems before they occur. This plays a pivotal role in the cloud native environment where workloads vary rapidly and arbitrarily. The future monitoring systems can provide constant feedback upon deployment by being integrated with performance-testing tools, and assist in more agile and data-driven management of live applications. That feedback can be used to both maintain and shape the behaviour of an application based on feedback of real world user interactions.

Automated scaling allows cloud platforms to vary the count of running servers or containers in response to the existing or predicted demand. Instead of having to do manual interventions when the peak or idle is met, scaling mechanisms provide the resources dynamically based on historical and real-time data. This will ensure maximum availability and performance when there is a surge in traffic and minimizing wastage and operational cost when there is no peak traffic. Auto-scaling policies may be straightforward indicators, such as a 80 percent use of the CPU, leading to an upscale, or more sophisticated ones, such as machine-learning models, which forecast patterns of use. Being intelligent and responsive infrastructure minimizes human error, enhances reliability, and facilitates easy user experiences

and also leads to sustainable computing by eliminating over-provisioning.

Real-time monitoring and automated scaling together change distributed systems and cloud services like AWS, Microsoft Azure, and Google Cloud Platform where the traffic load is produced significantly and the application should be designed to be elastic. Such an alliance will be needed to create indeed autonomous, self-healing, and efficient digital ecosystems as systems keep evolving.

7.5 Expansion of Test Environments and Platforms

With the proliferation of software applications to numerous contexts, performance testing has to evolve to a wider range of platforms - web, mobile, IoT etc. Current test environments, which are mobile-oriented to a large degree, are replacing the traditional desktop or server configurations. However, these bespoke platforms continue to lack major challenges posed by cloud computing, Internet-of-things gadgets, and other internet-related communication technologies.

Nowadays it is not just desktops or servers, but we can be deployed on web platforms, mobile devices, IoT devices and cloud platforms. The proliferation of platforms is associated with new challenges. Performance testing should be developed to ensure optimum functionality, reliability and user experience in all locations. The current environment requires the users to have a smooth performance regardless of the environment and device. Consider mobile applications: they should be capable of working on smartphones and tablets, which are different in terms of hardware, screen size and OS. The use of mobile is overshadowing the use of the desktop, yet the mobile devices sometimes have small memory, short battery life, and poor networks- all of which can hamper performance. Therefore, mobile performance testing needs to take into account device fragmentation, real-time connectivity and battery efficiency.

Smart devices that can include smart home devices and industrial sensors are a new source of performance challenges. They are frequently executed in low-resource, low-power or low-memory environments. They are also reliant on real time data transmission over networks and therefore latency, bandwidth and connection gaps are major considerations. To ensure the reliability of such distributed systems, testing tools have to be configured to handle decentralized and asynchronous communication models.

This is complicated by cloud computing and multi-platform applications. Cloud applications have to be scaled dynamically across geographically-distributed servers and with unpredictable loads. Now performance tests can simulate different user conditions all over the globe, test response times during changing loads and ensure that the system can elastically scale to loads without degradation.

To address these needs, the next generation performance testing needs to prioritize on cross platform compatibility, environmentally aware simulations and smart monitoring. The tools should be evolved into a single framework that verifies web, mobile, IoT, and cloud platforms in consideration of a set of limitations created by each platform.

Overall, with the increased diversity and distributed nature of software systems, performance testing has to be changed in order to provide performance that is robust across all locations. It implies reconsidering old methodologies and implementing scalable, elastic, smart, and methodologies that satisfy the complexity of the modern computing landscape.

7.6 Cross Platform Performance Testing

Among the areas of future work are the ones related to the improvement of performance testing tools to take into consideration to accommodate the vast variety of platforms on which current applications are implemented. Cross platform Test which is critical in ensuring that applications work well on mobile is the performance test phones desktops tablets and different operating systems. But considering the

fact that mobile applications tend to be susceptible to various issues than desktop application such performance testing tools will have to take into consideration changes in processing power screen size connection and network connectivity. To illustrate, there is the case of mobile performance testing that requires consideration of varying network speeds the lifetime of batteries, and different hardware capacities, which sometimes are not so conveniently managed like in desktop or server based systems. The next generation testing tool ought to be add these variables to the performance tests that guarantee that the applications operate ideally on various devices, and network conditions.

7.7 Iot and Edge Computing Performance Testing Video

The other future work is in terms of testing the performance of the Internet of things (Iot). The resources of iot device are usually finite such as processing power memory and network bandwidth. Testing the testing of Iot applications entails a simulation of huge networks of devices how things will interact and the way information will be processed on the edge of the network near the devices themselves. Iot performance testing will have to evaluate matters such as latency and data through put and power consumption. Also along with the increase of edge computing, there will be new challenges widely deployed making sure that distributed and decentralized applications are performed The role of environments will be more important in the future. Performance testing in the future the new environments will have to be taken into consideration in terms of methodologies including real time data processing and device to edge communication servers. One of the most critical areas of future work in software performance testing lies in the domain of Internet of Things (IoT) applications. IoT devices tend to be small, low-powered and have low processing, memory and bandwidth capabilities. It renders conventional performance testing models to be inefficient or inappropriate. With the increasing numbers of IoT ecosystems, such as smart homes and wearables, industrial sensors, and connected vehicles, the testing techniques also change. They have to model large, heterogeneous networks of devices which run on limited resources.

To measure the performance of IoT applications, it is necessary to test the number of devices communicating with each other in real time, data transmission, and information processing. Most of the devices apply edge computing, which works with data being processed locally around the origin rather than transmitting them to central data centres. This minimizes the latency and increases response time. Hence, data throughput, latency, packet loss, network reliability, and power consumption have become the metrics that must now be included during performance tests because they have a direct impact on performance and sustainability.

Furthermore, the more distributed and decentralized the IoT systems, the more difficult it would be to ensure uniform performance on a large and diverse scale of deployment. The testers should be able to recreate real world conditions: variable network connectivity, connectivity off and on, and hardware constraints. They should ensure that communication between devices and their edges and real-time data processing are reasonably efficient, and they do not have bottlenecks and failures. The use of edge computing further makes the issue more complex since data is being computed at more than one edge node adjacent to the devices, and not necessarily in the cloud. New test architectures must be in place to determine the interaction of system components in distributed conditions, the speed with which edge nodes process and forward data, and the system scalability with the addition of additional devices. Performance testing of the IoT and edge systems in the future will require advanced simulation environments; however, the mindset will require change as well, as centralized testing will no longer be applicable

but rather decentralized, context-sensitive evaluation. Real-time monitoring, predictive analytics and AI-based anomaly detection will be necessary so that these complex resource constrained systems can ensure their continued reliable performance as they scale.

7.8 Automation and Continuous Integration

The concept of automation is gaining momentum as software development is turning into a more popular phenomenon that incorporates Agile and DevOps. The current research is directed at the inclusion of automated performance tests to the pipelines of CD and CI/CD. This aids in ensuring that teams conduct performance tests on a regular basis and automatically, thereby assisting in the day to day development. Modern software development and testing have now been largely based on automation and CI. Automation involves the application of special tools and scripts to conduct testing activities with no human assistance. It manages test execution, result validation, report generation as well as bug tracking. Automation eliminates the need to work with hands, minimizes human error, and accelerates the test cycle, which is perfect in repetitious and regression tests.

Continuous Integration, however, is a discipline that automatically constructs, tests and integrates code modifications in a mutual repository a number of times each day. CI goes through every commit and ensures that it passes automated tests, and thus identifying integration failures at an early stage. This lowers the probability of bugs making it to the finished product and also allows release in a much quicker and confident manner. Automation and CI are a potent combination that helps to maintain constant testing. Each new code change will cause performance, functionality, and security testing, which gives quick feedback to developers and ensures that the applications remain stable and scalable and high-performing over time as new features are added. With Agile and DevOps adopted by the teams, automation, and CI need to be implemented in the testing processes to sustain a rapid quality and consistency throughout the delivery pipeline.

7.9 Automated Performance Testing in CI/CD Pipelines

It is possible to automate the performance tests during CI/CD pipelines to detect performance identifies problems early on in the development process, which makes it less probable that problems will occur in production environments. The combination of performance testing and tools of CI/CD will enable developers to be in a position to keep a constant check on the performance of their applications with minimum hand to hand exertion. This may include auto load testing when any code is committed or performance regressions are autopiloted It is noticed by comparison to history standards. Also, performance tests may be a part of the CI/CD pipeline instigate scaling tests to evaluate the capability of system to meet higher load or re allocations of the resources. This would enable the teams to be assured of performance It will be optimized despite the addition or modification of new features or modification of code.

7.10 Cloud Based Performance Testing and Cost Optimization

With the ever increasing transition to the cloud by organizations software work in the future will involve the following testing of performance will have to focus on the special issues posed by cloud based systems. Cloud environments are very flexible. Nevertheless, their performance issues may concern the cost optimization, resource distribution, and multicloud deployments.Cloud-based performance testing is used to test the responsiveness of applications to various user loads particularly in the current dynamic computing environment. As opposed to expensive on-premise hardware, virtual cloud environment can enable user simulation of thousands or even millions of users. This gives a chance to do more realistic and detailed tests of web and mobile applications that reach an international audience. AWS, Azure, and Google

Cloud major cloud providers provide first-order support and tooling of third-party testing solutions which makes it much easier to integrate into development pipelines.

Cost-saving is also instigated by cloud testing. A pay-as-you-go model would ensure that only resources consumed in tests are paid to the organizations, and no costly hardware that is seldom utilized is required. Resources can be scaled up or down within a short period of time and reducing cost can be further achieved by carrying out tests during off-peak hours or in low-cost areas. Automated resource management, real-time monitoring, and use metrics are offered by the majority of cloud platforms and assist teams to monitor expenditures, optimize test environments, and prevent resource waste.. Overall, cloud based performance testing not only enhances the accuracy and efficiency of performance evaluations but also aligns with modern strategies for scalable, agile development and cost-effective software delivery.

7.11 Cost Effective Performance Testing

Probably the future performance testing tools will have to include elements that will enable them to detect things like differences between high performance and low performance environments high performance and cost effectiveness are two factors which organizations balance. Cloud services are usually charged in use and as such, testing tools will require to enhance the best utilization of resources during performance tests in order to cut the cost of operation cloud testing. This may include automatic adjusting of tests parameters to minimize resource consumption or leveraging cloud based performance testing services to run tests in a cost efficient manner. As cloud computing becomes the dominant platform for software deployment, cost-effective performance testing will be a major focus for future tools and methodologies. Since most cloud services operate on a pay-as-you-go model, organizations must carefully balance the need for high-performance applications with the need to control costs. Future performance testing tools will need to be intelligent enough to identify differences between high-performance and low-performance environments, optimizing for both speed and cost-efficiency. To achieve this, testing tools must be capable of automatically adjusting parameters such as test duration, virtual user load, and resource allocation to reduce unnecessary consumption of cloud resources during tests. Additionally, leveraging cloud-native performance testing platforms will allow tests to be conducted in environments that closely mimic production, while still managing costs. Such platforms have the ability to scale-up and scale-down resources when tests are at their peak and when they are not in operation respectively, thus making resources to be used efficiently.

7.12 Multi Cloud and Hybrid Cloud Testing

Multi- and hybrid-cloud environments are migrating to many organizations, and this increases the market demand of performance-testing tools. The next generation tools should allow the testing of applications which run on an assembly of both the public and the private clouds and be able to work effectively regardless of the location in which they are deployed. There should be experiments to study the interactions of various cloud services to maintain the same performance even in the situations when the network conditions or server settings change irregularly. Advanced testing tools are even more important with the transition to the multi-cloud and hybrid infrastructure. These environments deploy applications on multiple providers, both public and private, both having their architecture, configuration, and SLA.

Testing tools should be used to measure the behavior that is systemic among all these diverse platforms in order to ensure that the performance is always consistent and reliable. They are to emulate real world conditions where they have an inter-provider interaction, a network that is not always going to be fast and a server configuration that keeps changing. Tests should not be done just to test speed and responsiveness but also test stability and

scalability with changing loads and environmental conditions. The tools require cross-cloud orchestration, real-time analytics, and adaptive strategies, which will change automatically to various structures. The end result is to provide the best performance and user experience everywhere, reducing the chances of failures or inefficiencies in the cases of applications running in distributed multi-cloud ecosystems.

7.13 Conclusion

The future of software performance testing is optimistic with new technologies like the ones that are projected to be coming out in the future producing lots of commodities like software performance testing artificial intelligence, machine learning, cloud computing, and IoT among other things, are expected to gain traction in this sector in the future. Transform the data landscape in the development of software. To get abreast with these changes performance testing methodologies have to be improved so as to consider complexities issues of the contemporary software apps. Automation and cross platform testing are going to become two issues of interest with future performance testing and the incorporation of AI to increase both the efficiency and accuracy of tests. Additionally, as the software is becoming more and more distributed and deployed in multiple environments test performances will have to change to become scalable and reliable and cost effective in cloud-based and multi cloud environments. In keeping abreast of events, performance testing is now becoming a significant aspect of the software development life cycle, assisting software developers in designing high quality, high performant applications that are responsive to the requirements of the present day user. The software performance testing holds a bright future.

The emerging technologies like artificial intelligence, machine learning, cloud computing and the Internet of Things are likely to get their momentum and bring an innovation into the sphere. Revamp software development. In order to keep abreast with these changes, performance testing techniques need to be modified in order to accommodate the complexities of the present-day software applications. The increase in automation and cross-platform testing will become the priorities, whereas integration with AI will increase efficiency and accuracy. Scaling Test performance in the cloud environment and multi-cloud environment as software is increasingly dispersed across environments, test performance needs to be scalable, reliable and cost-effective. Keeping pace with developments will render performance testing vital in the provision of quality efficient applications that will meet modern user needs.

AI, ML, cloud computing, and IoT have introduced transformations to the development picture that will drive the promise of performance testing with new complexities and increased levels of performance expectations. In order to keep up with these changes, there should be evolution of performance testing methods. The past methods might not suffice in today, distributed and dynamic systems. The automation will simplify routine work, reduce manual mistakes, and provide an opportunity to control it continuously. Cross-platform testing is one that makes sure that there is consistency in the behavior of various devices, operating systems, and various environments. There is the addition of AI and ML which gives predictive analytics, intelligent test generation, anomaly detection and fast identification of bottlenecks, a feature that is not available in traditional methods.

As an increasing number of applications shift to cloud-native applications, multi-cloud and hybrid cloud services, performance testing needs to be scalable, flexible, and cost-efficient. The tools should be able to model diverse loads and diversity of interactions among services, to maintain the performance of apps in unpredictable conditions. To conclude on it, performance testing will gain an even greater role

in software development. It enables developers to create high-quality, efficient, user-centric solutions that fulfill the demands of the current times through the adoption of automation, AI, and cloud-wise strategies.

Declarations

Ethics Approval: Neither humans nor animals are involved in this investigation. Thus, ethical approval was not necessary.

Consent to Participate: Not applicable.

Consent to Publish: Not applicable.

Funding: This research received no external funding.

Data Availability Statement: No data set is used in this study.

References

[1] Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using deep learning for image-based plant disease detection. *Frontiers in plant science*, *7*, 1419.

[2] Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D., & Stefanovic, D. (2016). Deep neural networks based recognition of plant diseases by leaf image classification. *Computational intelligence and neuroscience*, *2016*(1), 3289801.

[3] Ferentinos, K. P. (2018). Deep learning models for plant disease detection and diagnosis. *Computers and electronics in agriculture*, *145*, 311-318.

[4] Arnal Barbedo, J. G. (2013). Digital image processing techniques for detecting, quantifying and classifying plant diseases. *SpringerPlus*, *2*(1), 660.

[5] Hughes, D., & Salathé, M. (2015). An open access repository of images on plant health to enable the development of mobile disease diagnostics. *arXiv preprint arXiv:1511.08060*.

[6] Too, E. C., Yujian, L., Njuki, S., & Yingchun, L. (2019). A comparative study of fine-tuning deep learning models for plant disease identification. *Computers and Electronics in Agriculture*, *161*, 272-279.

[7] Patil, J. (2021). Pomegranate fruit diseases detection using image processing techniques: a review. *Information Technology in Industry*, *9*(2), 115-120.

[8] Sindhu, D., & Sindhu, S. (2019). Image processing technology application for early detection and classification of plant diseases. *Intern J Computer Sci Engr*, *7*(5), 92-97.

[9] Arivazhagan, S., Shebiah, R. N., Ananthi, S., & Varthini, S. V. (2013). Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features. *Agricultural Engineering International: CIGR Journal*, *15*(1), 211-217.

[10] Taha, M. F., Mao, H., Zhang, Z., Elmasry, G., Awad, M. A., Abdalla, A., ... & Elsherbiny, O. (2025). Emerging technologies for precision crop management towards agriculture 5.0: A comprehensive overview. *Agriculture*, *15*(6), 582.

[11] Patil, M. N. S., & Kannan, E. (2020). Rice Plant Disease Detection using Image Processing: A Review. *International Journal of Future Generation Communication and Networking*, *13*(4), 4899-4918.

[12] Nkwocha, C. L., & Chandel, A. K. (2025). Towards an end-to-end digital framework for precision crop disease diagnosis and management based on emerging sensing and computing technologies: state over past decade and prospects.

[13] Wang, G., Sun, Y., & Wang, J. (2017). Automatic image-based plant disease severity estimation using deep learning. *Computational intelligence and neuroscience*, *2017*(1), 2917536.

[14] Brahimi, M., Boukhalfa, K., & Moussaoui, A. (2017). Deep learning for tomato diseases: classification and symptoms visualization. *Applied Artificial Intelligence*, *31*(4), 299-315.

[15] Bhargava, A., Shukla, A., Goswami, O. P., Alsharif, M. H., Uthansakul, P., & Uthansakul, M. (2024). Plant leaf disease detection, classification, and diagnosis using computer vision and artificial intelligence: A review. *IEEE access*, *12*, 37443-37469.

[16] Sphinx, "Flask Official Documentation," June 2025. [Online]. Available: https://flask.palletsprojects.com/.

[17] Flask Official Documentation," [Online]. Available: https://flask.palletsprojects.com/. [Accessed June 2025].

[18] SQLite Documentation," [Online]. Available: https://sqlite.org/docs.html. [Accessed June 2025].

[19] Bootstrap, " Bootstrap Documentation," 2025. [Online]. Available: https://getbootstrap.com/docs/.