

# **UNIVERSITY OF MUMBAI**

A PROJECT REPORT ON

## **“PLANT DISEASE DETECTION USING CNN “**

SUBMITTED BY

**MR. DANIEL SIVAPRASAD  
MS. SHRUDHI BABU**

UNDER THE GUIDANCE OF

**PROF. CHANDANI PATEL**

**Late Shri. Vishnu Waman Thakur Charitable Trust's  
VIVA INSTITUTE OF TECHNOLOGY  
Shirgaon, Virar (East)  
2021-22**

Late Shri. Vishnu Waman Thakur Charitable Trust's  
**VIVA INSTITUTE OF TECHNOLOGY**  
Shirgaon, Virar(East)



**CERTIFICATE**

This is to certify that

**MR. DANIEL SIVAPRASAD**  
**&**  
**MS. SHRUDHI BABU**

Has satisfactorily completed the project entitled

**PLANT DISEASE DETECTION USING CNN**

Towards the partial fulfillment of the  
**MASTER OF COMPUTER APPLICATION (MCA)**  
As laid by University of Mumbai.

\_\_\_\_\_  
Principal

\_\_\_\_\_  
External Examiner

\_\_\_\_\_  
Internal Guide

## **DECLARATION**

We “Mr. Daniel Sivaprasad” and “Ms. Shrudhi Babu” hereby declare that we ourselves have completed the project under the guidance of **Prof. Chandani Patel**. We on our own have designed the system and have done all the programming required.

It may require some modifications in the future as per the user’s requirements. From practical implementation point of view, flexibility in the changes have incorporated in the package.

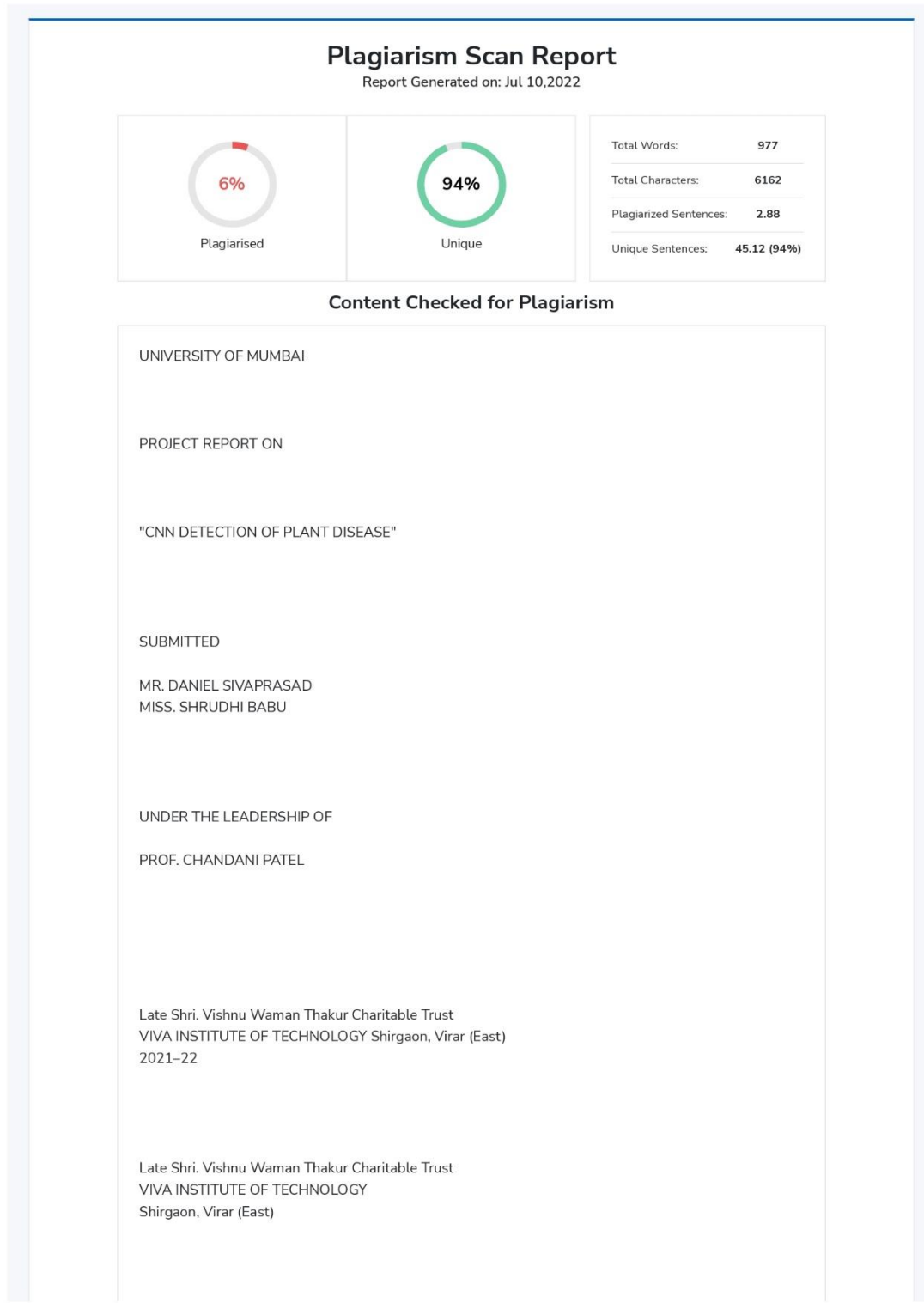
We are sure that we can do any kind of modification suggested while practical implementation by modifying file design or the program code if necessary.

**MR. DANIEL SIVAPRASAD**

**MS. SHRUDHI BABU**

**(author’s)**

# PLAGIARISM REPORT



Page 1 of 4

## **ACKNOWLEDGEMENT**

First and foremost. I bow my head to the Almighty for being my light and His showers of blessing during this mini project. I extend my gratitude to **Dr. ARUN KUMAR**, Principal. Viva Institute of Technology, to provide all the resources to do this mini project work. I record my sincere thanks and respect to our Head of Department, **Prof. Chandani Patel**, Department of Master of Computer Application, Viva Institute of Technology, with her good attention and outstanding guidance in all the work of my project.

I am indebted to my project coordinator and project guide **Prof. Chandani Patel**, Department of Master of Computer Application, Viva Institute of Technology for her guidance and important suggestions for all the ways to complete my project.

I extend my deepest gratitude to all the other staff for their support and all assistance during the study and course work of the project. Finally, I truly thank my parents and friends for giving us valuable and suggestion for the improvement of my project.

## **ABSTRACT**

The proposed system helps in identification of plant disease and provides remedies that can be used as a defense mechanism against the disease. The database obtained from the Internet is properly segregated and the different plant species are identified and are renamed to form a proper database then obtain test-database which consists of various plant diseases that are used for checking the accuracy and confidence level of the project. Then using training data, we will train our classifier and then output will be predicted with optimum accuracy. We use Convolution Neural Network (CNN) which comprises of different layers which are used for prediction. A prototype drone model is also designed which can be used for live coverage of large agricultural fields to which a high-resolution camera is attached and will capture images of the plants which will act as input for the software, based of which the software will tell us whether the plant is healthy or not. With our code and training model we have achieved an accuracy level of 97%. Our software gives us the name of the plant species with its confidence level and also the remedy that can be taken as a cure.

# INDEX

Sr. No.	Contents	Page No.
1	<b>Introduction</b>	
	<b>1.1 Introduction</b>	10
	1.1.1 Problem definition	11
	1.1.2 Objectives of Project	11
	1.1.3 Literature Survey	12
	1.1.3 Scope of Project	13
	<b>1.2 Technical Details</b>	
	1.2.1 Tools Used	14
2	<b>System Study and Planning</b>	15
	<b>2.1 System Study</b>	
	2.1.1 Existing System	16
	2.1.2 Disadvantages of Existing system	16
	2.1.3 Proposed System	17
	<b>2.2 System Planning and Schedule</b>	
	2.2.1 Activity Sheet	18
	2.2.2 S/W Development Model	19
	2.2.3 Waterfall Model	21
	2.2.4 Gantt Chart	22
3	<b>System Design</b>	23
	<b>3.1 Software Requirement Specification (SRS)</b>	
	3.1.1 Introduction of SRS	24
	3.1.1.1 Functional Requirements	25
	3.1.1.2 Non-Functional Requirements	25
	3.1.1.3 User Requirement	26
	3.1.1.4 Feasible Study	27
	3.1.2 Technology Requirements	28
	3.1.2.1 Software to be used	28
	3.1.2.2 Hardware to ne used	28
	<b>3.2 Detailed life Cycle of the Project</b>	
	3.2.1 Modules	29
	3.2.2 Diagrams	
	3.2.2.1 Use Case Diagram	31

	3.2.2.2 Sequence Diagram	32
	3.2.2.3 Activity Diagram	32
	3.2.2.4 Data Flow Diagram	33
	3.2.3 Methodology	35
	3.2.4 Screen Layout	37
	3.2.5 Lines of Code	39
	<b>Testing</b>	44
	4.1 Testing Objective	45
	4.2 White Box Testing	46
	4.3 Levels of Testing	46
	4.4 Integration Test Plan	48
	4.5 Unit Testing	48
	4.6 Output Testing	49
5	<b>Conclusion</b>	50
6	<b>Limitations</b>	51
7	<b>Future Enhancements</b>	52
8	<b>References</b>	53



# INTRODUCTION

## 1.1 INTRODUCTION

The primary occupation in India is agriculture. India ranks second in the agricultural output worldwide. Here in India, farmers cultivate a great diversity of crops. Various factors such as climatic conditions, soil conditions, various disease, etc affect the production of the crops. The existing method for plants disease detection is simply naked eye observation which requires more man labor, properly equipped laboratories, expensive devices, etc. And improper disease detection may led to inexperienced pesticide usage that can cause development of long term resistance of the pathogens, reducing the ability of the crop to fight back. The plant disease detection can be done by observing the spot on the leaves of the affected plant. The method we are adopting to detect plant diseases is image processing using Convolution neural network (CNN).

Machine learning is a computational way of detecting patterns in a given dataset in order to make inferences in another, similar dataset. In Indian Economy a Machine learning based recognition system will prove to be very useful as it saves efforts, money and time too. The approach given in this for feature set extraction is the color co-occurrence method. For automatic detection of diseases in leaves, neural networks are used. The approach proposed can significantly support an accurate detection of leaf, and seems to be important approach, in case of steam, and root diseases, putting fewer efforts in computation.

## 1.1.1

# PROBLEM DEFINITION

Plant diseases have turned into a dilemma as it can cause significant reduction in both quality and quantity of agricultural products. Automatic detection of plant diseases is an essential research topic as it may prove benefits in monitoring large fields of crops, and thus automatically detect the symptoms of diseases as soon as they appear on plant leaves. The proposed system is a software solution for automatic detection and classification of plant leaf diseases. The scheme consists of four main steps, first a color transformation structure for the input RGB image is created, then the green pixels are masked and removed using specific threshold value followed by segmentation process, the texture statistics are computed for the useful segments, finally the extracted features are passed through the classifier.

## 1.1.2

# OBJECTIVE OF PROJECT

We can reduce the attack of pests by using proper pesticides and remedies. We can reduce the size of the images by proper size reduction techniques and see to it that the quality is not compromised to a great extent. We can expand the projects of the earlier mentioned authors such that the remedy to the disease is also shown by the system. The main objective is to identify the plant diseases using image processing. It also, after identification of the disease, suggest the name of pesticide to be used. It also identifies the insects and pests responsible for epidemic. Apart from these parallel objectives, this drone is very time saving. The budget of the model is quite high for low scale farming purposes but will be value for money in large scale farming. It completes each of the process sequentially and hence achieving each of the output.

**Thus, the main objectives are:**

- To design such system that can detect plant disease and pest accurately.
- Create database of insecticides for respective pest and disease.
- To provide remedy for the disease that is detected.

### 1.1.3

## LITERATURE SURVEY

- ❖ The application of texture statistics for detecting the plant leaf disease has been explained the work done by Prof. Sanjay, B. Dhaygude. Firstly by color transformation structure RGB is converted into HSV space because HSV is a good color descriptor. Masking and removing of green pixels with pre-computed threshold level. Then in the next step segmentation is performed using 32X32 patch size and obtained useful segments. These segments are used for texture analysis by color co-occurrence matrix. Finally, if texture parameters are compared to texture parameters of normal leaf.
- ❖ In the work done by Amandeep Singh, Maninder Lal Singh, the most significant challenge faced during the work was capturing the quality images with maximum detail of the leaf color. It is very typical task to get the image with all the details within a processable memory. Such images are formed a through high resolution and thus are of 6-10MB of size. This was handled by using a Nikon made D5200 camera which served the task very well. Second challenge faced was to get rid of illumination conditions as from the start to the end of paddy crop season, illumination varies a lot even when the image acquiring time is fixed. However, the solution to this is variable user defined thresholding and making necessary adjustments to the shades of LCC.
- ❖ In the work done by Malvika Ranjan, Manasi Rajiv Weginwar, describes a diagnosis process that is mostly visual and requires precise judgment and also scientific methods. Image of diseased leaf is captured. As the result of segmentation Color HSV features are extracted. Artificial neural network (ANN) is then trained to distinguish the healthy and diseased samples. ANN classification performance is 80% better in accuracy.
- ❖ Work done by P.Revathi, M.Hemalatha is based on Image Edge detection Segmentation techniques in which, the captured images are processed for enrichment first. Then R, G, B color Feature image segmentation is carried out to get target regions (disease spots). Later, image features such as boundary, shape, color and texture are extracted for the disease spots to recognize diseases and control the pest recommendation. In this Research work consist three parts of the cotton leaf spot, cotton leaf color segmentation, Edge detection-based Image segmentation, analysis and classification of disease.

## 1.1.4

# SCOPE OF PROJECT

Plant diseases cause a major production and economic losses in the agricultural industry. The disease management is a challenging task. Usually, the diseases or its symptoms such as colored spots or streaks are seen on the leaves of a plant. In plants most of the leaf diseases are caused by fungi, bacteria, and viruses. The diseases caused due to these organisms are characterized by different visual symptoms that could be observed in the leaves or stem of a plant. Usually, these symptoms are detected manually. With the help of image processing, Automatic detection of various diseases can be detected with the help of image processing. Image processing plays a crucial role in the detection of plant diseases since it provides best results and reduces the human efforts.

The image processing could be used in the field of agriculture for several applications. It includes detection of diseased leaf, stem or fruit, to measure the affected area by disease, to determine the color of the affected area.

The use of image processing techniques in detection and identification of tomato plant diseases in the earlier stages and thereby the quality of the product could be increased. These systems monitor the plant such as leaves and stem and any variation observed from its characteristic features, variation will be automatically identified and also will be informed to the user.

## **1.2 TECHNICAL DETAILS**

### **1.2.1 TOOLS USED**

- ❖ Python based Computer Vision and Deep Learning libraries such as cv2, pickle, Label Binarizer, Convo2D, MaxPooling2D, Image Data Generator will be exploited for the development and experimentation of the project.
- ❖ Deep learning techniques, and in particular Convolutional Neural Networks (CNNs), have led to significant progress in image processing. Since 2016, many applications for the automatic identification of plant diseases have been developed. These applications could serve as a basis for the development of expertise assistance or automatic screening tools. Such tools could contribute to more sustainable agricultural practices and greater food production security.
- ❖ Tools such as Anaconda Python, and libraries such as OpenCV, will be utilized for this process. OpenCV (Open-Source Computer Vision) is a library that can be imported in almost all computer languages like python, C, Java etc. It contains optimized image processing tools. Using OpenCV in python boosts its abilities by incorporating numpy (Numerical Python). In image processing, images are dealt as large 3D arrays and numpy serves as a robust tool for numerical array computations.

# **SYSTEM STUDY AND PLANNING**

## **2.1 SYSTEM STUDY.**

### **2.1.1 EXISTING SYSTEM**

The existing method for plant disease detection is simply naked eye observation by experts through which identification and detection of plant diseases is done. For doing so, a large team of experts as well as continuous monitoring of plant is required, which costs very high when we do with large farms.

### **2.1.2 DISADVANTAGE OF EXISITING SYSTEM**

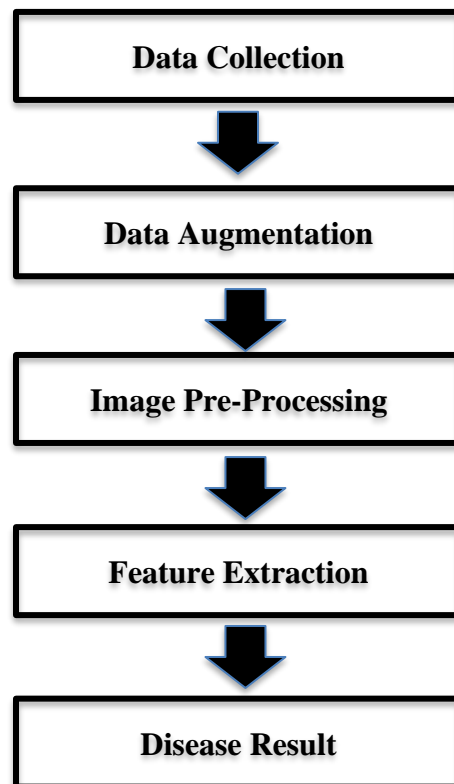
Plant disease identification by visual way is more laborious task and at the same time, less accurate and can be done only in limited areas. At the same time, in some countries, farmers do not have proper facilities or even idea that they can contact to experts. Due to which consulting experts even cost high as well as time consuming too. In such conditions, the suggested technique proves to be beneficial in monitoring large fields of crops. Automatic detection of the diseases by just seeing the symptoms on the plant leaves makes it easier as well as cheaper. This also supports machine vision to provide image based automatic process control, inspection, and robot guidance.



## 2.1.3

# PROPOSED SYSTEM

The plant disease detection system is considered as an image classification problem and can be performed using a deep learning algorithm. Hence the CNN architecture can be used to perform this disease detection system. To perform CNN the required data should be collected using Data collection, data augmentation, pre-processing, and feature extraction as shown in the below figure. Normally the feature extraction is performed using features of the trained data and based upon that data, the threshold values can be kept and then in testing part the value of the features will compare with the trained one to predict the image is diseased or not.



**Fig: Architecture diagram of the proposed system**

## 2.2 SYSTEM PLANNING AND SCHEDULE.

### 2.2.1 ACTIVITY SHEET

Sr. No.	Activity			Sign
		Planned Date	Actual Date	
1.	Preliminary Investigation	18/04/2022	21/04/2022	
2.	System Study and Analysis	25/04/2022	28/04/2022	
3.	System Coding and Report	16/05/2022	17/05/2022	
4.	System Testing	20/06/2022	25/06/2022	
5.	Project Submission	04/07/2022	08/07/2022	

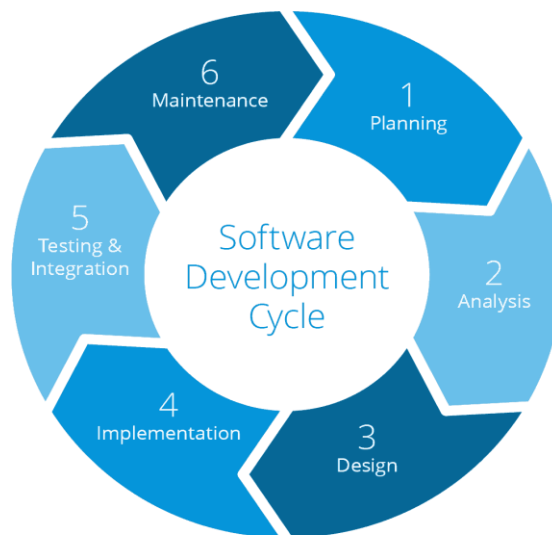
## 2.2.2

# SYSTEM DEVELOPMENT LIFE CYCLE (SDLC)

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality software. The SDLC aims to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

### What is SDLC?

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.



### 1. Planning/Requirement Analysis

Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry.

This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas

## 2. Defining

Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts.

This is done through an SRS (Software Requirement Specification) document which consists of all the product requirements to be designed and developed during the project life cycle

## 3. Design

The list of requirements that you develop in the definition phase is used to make design choices. In the design phase, one or more designs are created to achieve the project result.

Depending on the project subject, the design phase products include flow-charts, entry-relationship diagram, screen designs, photo impressions.

## 4. Implementation/Coding

In this stage of SDLC the actual development starts and the product is built. The programming code is generated during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.

## 5. Testing

This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

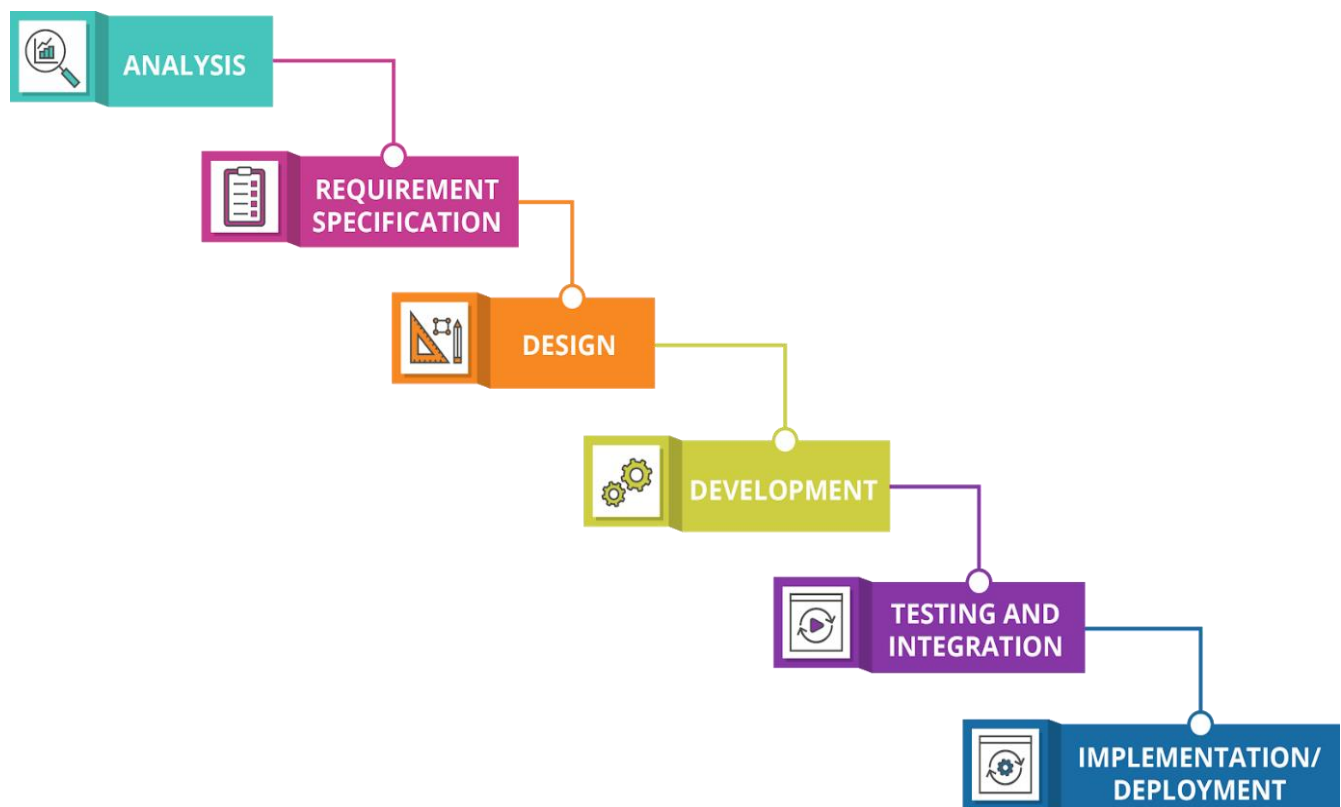
## 6. Maintenance

In this stage of the SDLC occurs after the product is in full operation. Maintenance of software can include software upgrades, repairs, and fixes of the software if it breaks. During the maintenance phase, errors or defects may exist, which would require repairs during additional testing of the software.

## 2.2.3













# WATERFALL MODEL

- ❖ Water fall model is used to develop this project.
- ❖ The waterfall model is a breakdown of project activities into linear sequential phases, where each phase depends on the deliverables of the previous one and corresponds to a specialization of tasks.
- ❖ The approach is typical for certain areas of engineering design



## 2.2.4

### GANTT CHART

<div>Period</div> <div>Process</div>	18/04/22 to 23/04/22 (Week 1)	25/04/22 to 14/05/22 (Week 2 - Week 4)	16/05/22 to 02/07/22 (Week 5 - Week 11)	04/07/22 to 08/07/22 (Week 12)
Preliminary Investigation	 			
System Analysis		 		
System Design		 		
System Coding			 	
Testing & Evaluation			 	
Project Submission				 

- Planned Date



- Actual Date



# SYSTEM DESIGN

## 3.1 SOFTWARE REQUIREMENT SPECIFICATION

### 3.1.1 INTRODUCTION OF SRS

Software (SRS) specification document captures a complete description of how the system is expected to operate. It is usually signed at the end of the needs engineering phase. The production of the requirements section of the software development process is Software Requirements Specifications (SRS) (also called the requirements document). This report lays the foundation for software engineering activities and builds on where all the requirements are requested and analyzed. SRS is an official report, acting as a representative software that enables customers to review whether (SRS) complies with their needs. Also, it includes system user requirements as well as detailed specification of system requirements.

The diagram below shows the different types of requirements captured during SRS.





### 3.1.1.1 Functional Requirements

- ❖ Functional requirements describe the system functionality, while the non-functional requirements describe system properties and constraints. Functional requirements capture the intended behavior of the system.
- ❖ This behavior may be expressed as services, tasks, or the functions the system is required to perform. This lays out important concepts and discusses capturing functional requirements in such a way they can drive architectural decisions and be used to validate the architecture.
- ❖ Features may be additional functionality, or differ from basic functionality along some quality attribute.
- ❖ In the proposed system, concert assesses the compliance of a workflow by analyzing the five established elements required to check for the rule adherence in workflows: activities, data, location, resources, and time limits.
- ❖ A rule describes which activities may, must or must not be performed on what objects by which roles. In addition, a rule can further prescribe the order of activities i.e., which activities have to happen before or after other activities.

### 3.1.1.2 Non-Functional Requirements

- ❖ **Security**  
System needs to control the user access and session It needs to store the data in a secure location and stored in a secure format It requires a secure communication channel for the data.
- ❖ **Concurrency and Capacity**  
System should be able to handle multiple computations executing simultaneously, and potentially interacting with each other.
- ❖ **Performance**  
Performance is generally perceived as a time expectation. This is one of the most important considerations especially when the project is in the architecture phase.

**❖ Reliability**

It is necessary to ensure and notify about the system transactions and processing as simple as keep a system log will increase the time and effort to get it done from the very beginning. Data should be transferred in a reliable way and using trustful protocols.

**❖ Maintainability**

Well-done system is meant to be up and running for long time. Therefore, it will regularly need preventive and corrective maintenance. Maintenance might signify scalability to grow and improve the system features and functionalities.

**❖ Usability**

End user satisfaction and acceptance is one of the key pillars that support a project success. Considering the user experience requirements from the project conception is a win bet, and it will especially save a lot of time at the project release, as the user will not ask for changes or even worst misunderstandings.

**❖ Documentation**

All projects require a minimum of documentation at different levels. In many cases the users might even need training on it, so keeping good documentation practices and standards will do this task spread along the project development; but as well this must be establish since the project planning to include this task in the list.

**3.1.1.3 User Requirement**

- ❖ The project is totally built at administrative end and thus only admin is guaranteed the access
- ❖ The only requirement for using this system is having machine with respect software in which the system was build

### 3.1.1.4 Feasible Study

- ❖ Feasibility Study in Software Engineering is a study to evaluate feasibility of proposed project or system. Feasibility study is one of stage among important four stages of Software Project Management Process.
- ❖ Feasibility study is carried out based on many purposes to analysis whether software product will be right in terms of development, implantation, contribution of project to the organization etc.

#### **Types of feasibility:**

- ❖ Technical feasibility
- ❖ Operational feasibility
- ❖ Economical feasibility

#### **Technical**

- Technical feasibility is used to find out whether the necessary technology for proposed equipment have the capacity to hold the data, which is used in the project.
- The technical feasibility issues usually raised during the feasibility stage of investigation.
- It is a study of functionality, performance and constraints that may affect the ability to achieve an acceptance system as described in the System Requirement & Specification (SRS) in System Defining stage.

#### **Operational**

- The implemented software replaces the traditional handwritten registers with automatic and convenient software which enables user to save time in, recording in and recording out of the product/commodities, sales and purchase.

- Provides easy storing of data in hard disk or servers, rather than the inconvenient and lofty registers which takes up physical space to be stored.

### **Economical**

- This is a very important aspect to be considered while developing a project. We decided the technology based on minimum possible cost factor.
- All hardware and software cost has to be borne by organization
- Over all we have estimated that the benefits the organization is going receive from the proposed system will surely overcome the initial costs and the later on running cost for system.

## **3.1.2 TECHNOLOGY REQUIREMENTS**

### **3.1.2.1 Software to be used:**

❖ Operating System	:	Windows 7 and above.
❖ Programing language	:	Python 2.7 and above.
❖ Platform	:	Jupyter Notebook
❖ Supporting libraries	:	Tensorflow, OpenCV, PIL, etc

### **3.1.2.2 Hardware to be used:**

❖ Processor	:	2.5 gigahertz (GHz) frequency or above.
❖ RAM	:	A minimum of 4 GB of RAM.
❖ Hard disk	:	A minimum of 20 GB of available space.
❖ Monitor	:	Minimum Resolution 1024 X 768

## **3.2. DETAILED LIFE CYCLE**

### **3.2.1 MODULES**

1. Image Acquisition
2. Image Preprocessing
3. Image Segmentation
4. Feature Extraction
5. Disease Classification

#### **1. Image Acquisition:**

Image acquisition means acquiring an image by means of camera from any real-life scene. In today's world, commonly used method is capturing photo by using digital camera. But other methods can also be used. In this project, images are taken from plant village dataset through which the images will be fetched and the algorithm will be trained and tested

#### **2. Image Preprocessing:**

Image pre-processing is used to increase the quality of image necessary for further processing and analysis. It includes color space conversion, image smoothing and image enhancement. The quality of input image is achieved by removing undesired distortion from the image. Image enhancement is performed to increase the contrast of image. Image clipping is done to get interested region. Smoothing filter is used for image smoothing.

#### **3. Image Segmentation:**

Image segmentation is the process of separating or grouping an image into different parts. Image segmentation are divided in to 3 categories

- Edge based
- Region based
- Clustering based

#### **4. Feature Extraction:**

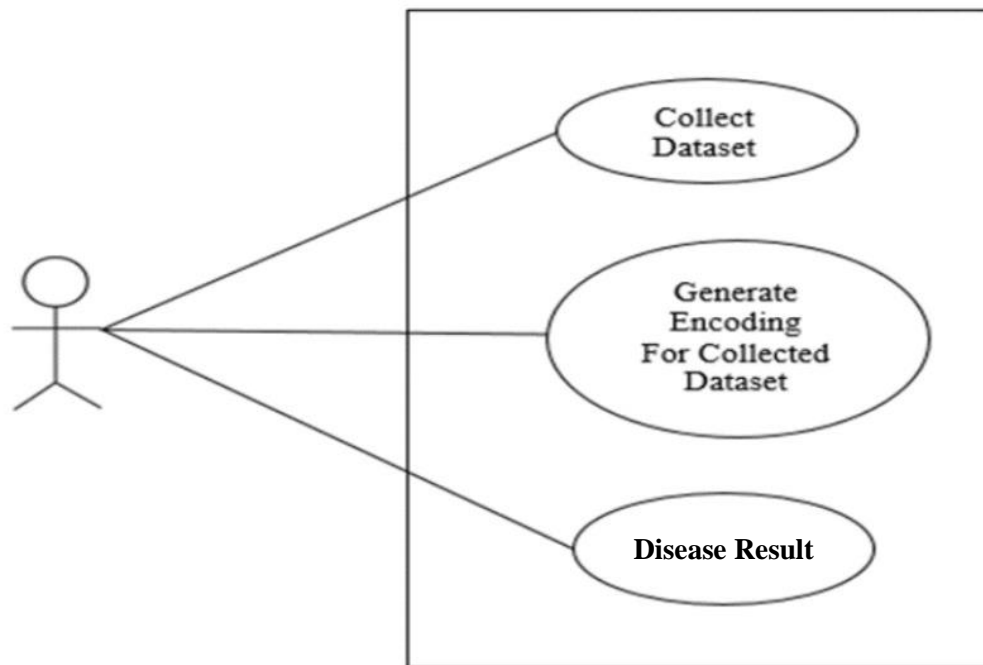
Extraction of features is the significant component of predicting the infected region gracefully. Extraction of feature involves reducing the amount of resource needed to describe large dataset. It is a method of identifying image characteristics and set of characteristics that will meaningfully represent significant classification and analysis data. It is expected that the extracted features will contain appropriate information from the input data, using this decreased representation instead of the full original data that the required job can do. Texture content counting is in main approach for region description.

#### **5. Disease Classification:**

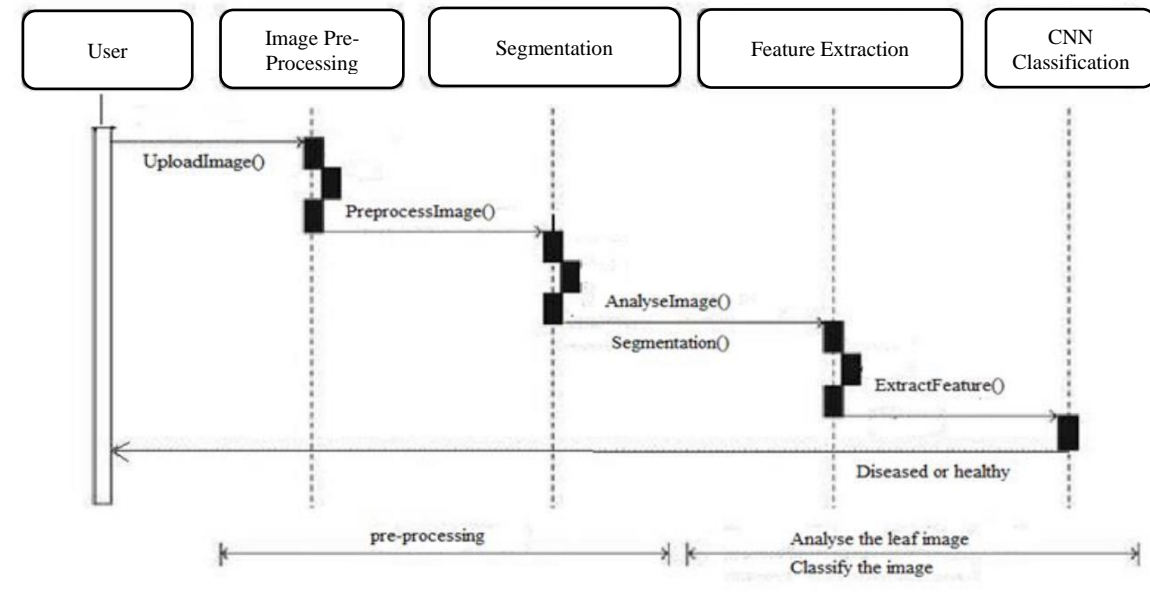
Convolutional neural networks are used in the automatic detection of leaves diseases. CNN is chosen as a classification tool due to its well-known technique as a successful classifier for many real applications. CNN architectures vary with the type of the problem at hand. The proposed model consists of three convolutional layers each followed by a max-pooling layer. The final layer is fully connected MLP. ReLu activation function is applied to the output of every convolutional layer and fully connected layer. The first convolutional layer filters the input image with 32 kernels of size 3x3. After max-pooling is applied, the output is given as an input for the second convolutional layer with 64 kernels of size 4x4. The last convolutional layer has 128 kernels of size 1x1 followed by a fully connected layer of 512 neurons. The output of this layer is given to Softmax function which produces a probability distribution of the four output classes. Gray Level Co-occurrence Matrix is created from gray scale images and used to describe the shape feature. The Gray Level Co-occurrence Matrix is based on the repeated occurrence of gray-level configuration in the texture. The spatial gray dependence matrix is used for texture analysis. A spatial gray dependence matrix is created based on hue, saturation and intensity. Run Length Matrix (RLM) is another type of matrix. Same gray pixel values are the part of run and those gray values forms a two-dimensional matrix.

## 3.2.2 DIAGRAMS

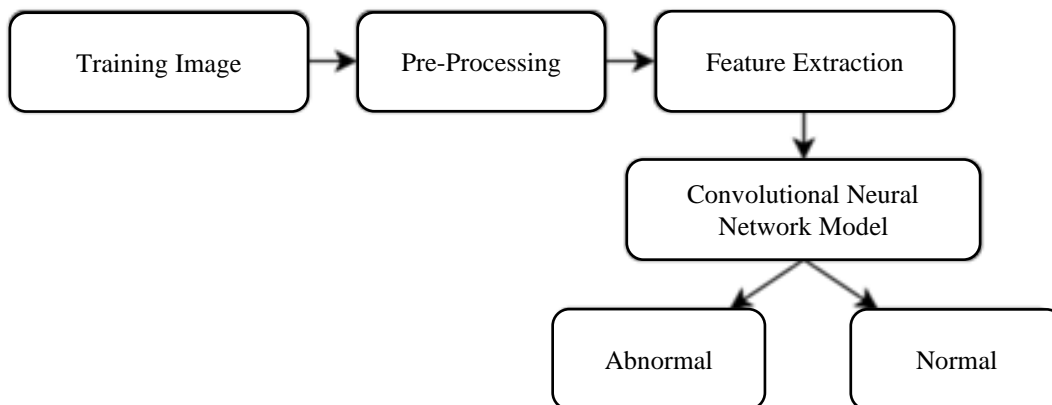
### 3.2.2.1 Use Case Diagram



### 3.2.2.2 Sequence Diagram



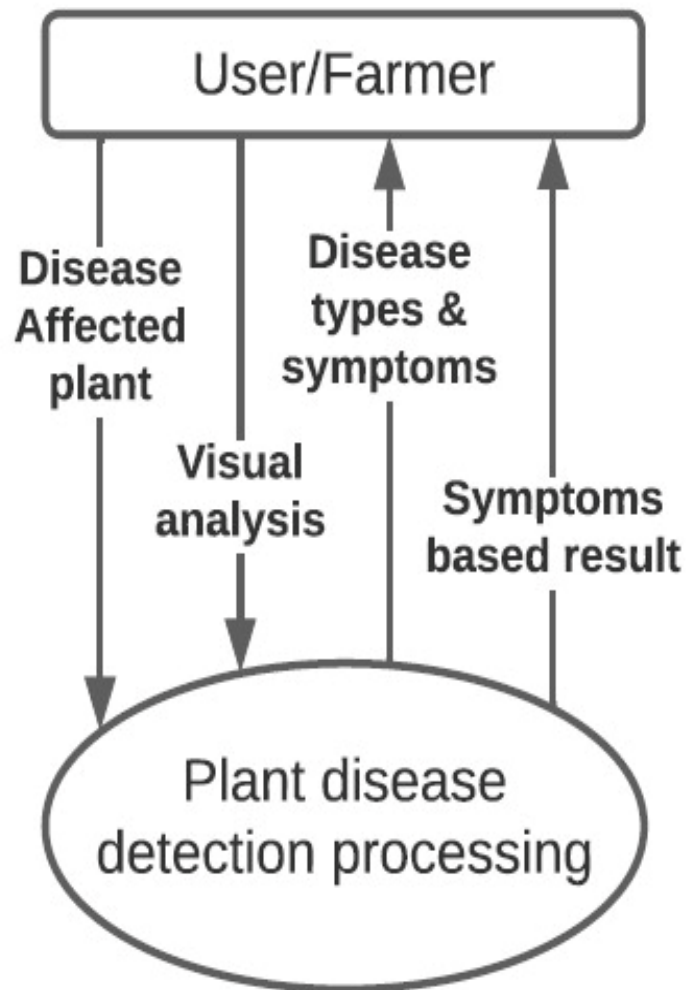
### 3.2.2.3 Activity Diagram

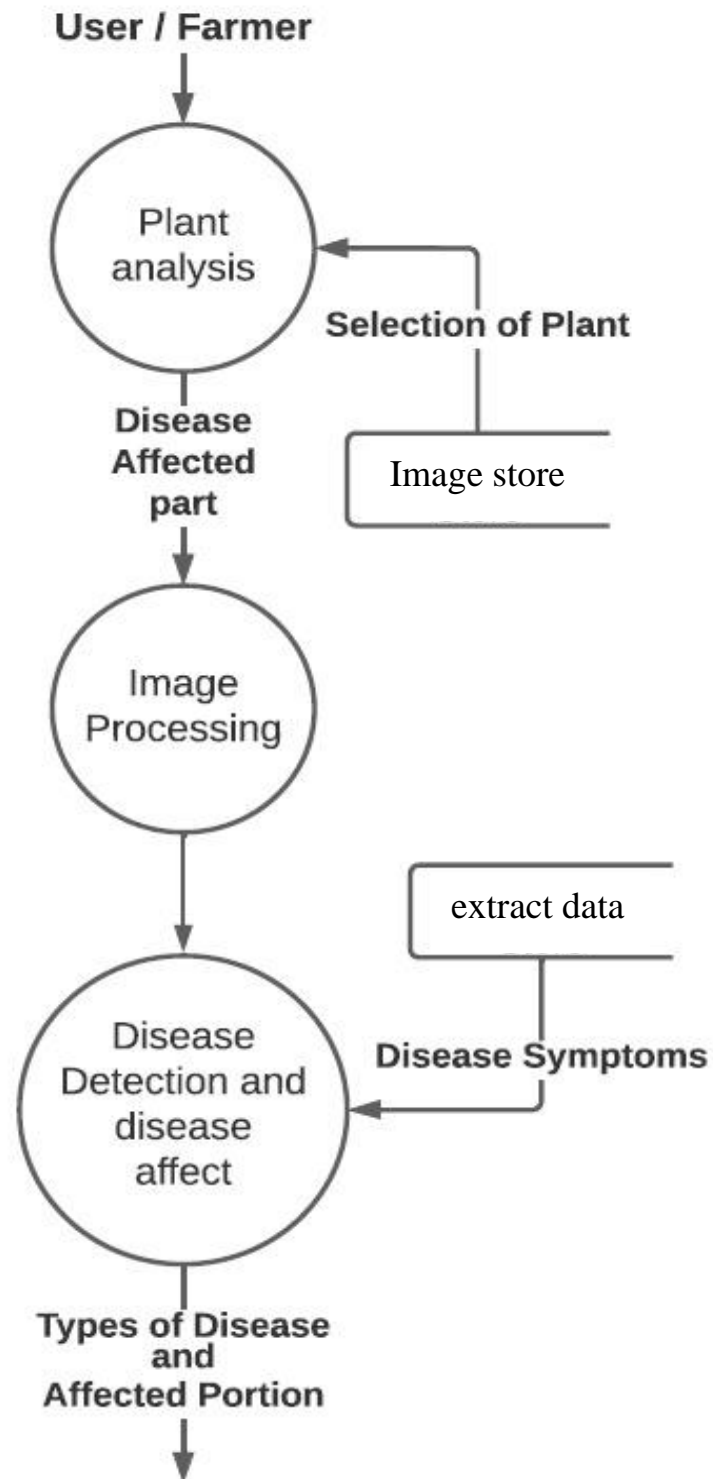




### 3.2.2.4 Data Flow Diagram

**Context Level:**



**Level 1:**

## 3.2.3

# METHODOLOGY

### Database collection:

Initial step for any image processing-based project is acquiring proper database which is valid. Most of the time the standard database is preferred but in certain circumstances we do not get proper database. So, in such conditions we can collect the images and can form our own database. The database is accessed from crowd AI which is plant disease classification challenge. Data available here is not labeled. So, the first task is to clean and label the database. There is a huge database so basically the images with better resolution and angle are selected. After selection of images, we should have deep knowledge about the different leaves and the disease they have. Huge research is done from plant village organization repository. Different types of plant images are studied and corresponding. After detail study, labeling is done by segregating the images and with different diseases.

#### Diseases of different plants database:

- Pepper\_\_\_bell\_\_\_Bacterial\_spot
- Pepper\_\_\_bell\_\_\_healthy.
- Potato\_\_\_Early\_blight.
- Potato\_\_\_healthy.
- Potato\_\_\_Late\_blight.
- Tomato\_\_\_Target\_Spot.
- Tomato\_\_\_Tomato\_mosaic\_virus.
- Tomato\_\_\_Tomato\_YellowLeaf\_\_\_Curl\_Virus
- Tomato\_Bacterial\_spot
- Tomato\_Early\_blight
- Tomato\_healthy
- Tomato\_Late\_blight
- Tomato\_Leaf\_Mold
- Tomato\_Septoria\_leaf\_spot
- Tomato\_Spider\_mites\_Two\_spotted\_spider\_mite.
- Tomato\_Spider\_mites\_Two\_spotted\_spider\_mite

**Preprocessing and training the model (CNN):**

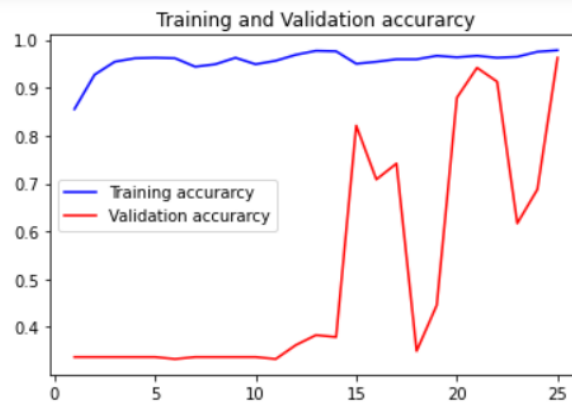
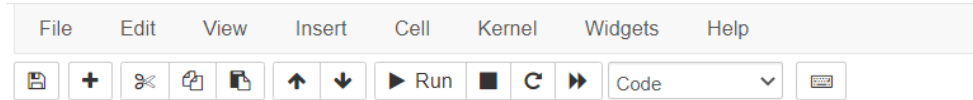
The database is Preprocessed such as Image reshaping, resizing and conversion to an array form. Similar processing is also done on the test image. A database consisting of about 32000 different plant species is obtained, out of which any image can be used as a test image for the software. The train database is used to train the model (CNN) so that it can identify the test image and the disease it has. CNN has different layers that are Dense, Dropout, Activation, Flatten, Convolution2D, MaxPooling2D. After the model is trained successfully, the software can identify the disease if the plant species is contained in the database. After successful training and preprocessing, comparison of the test image and trained model takes place to predict the disease.

## 3.2.4

# SCREEN LAYOUT

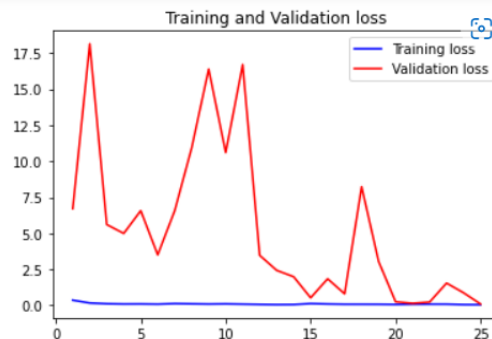
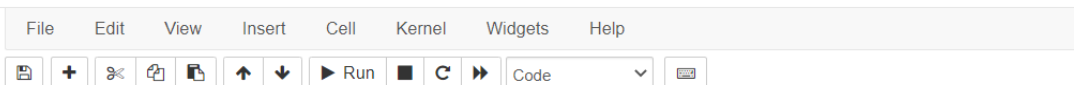
### 1. Training & Validation Accuracy

jupyter Plant disease detection Last Checkpoint: 11 hours ago (autosaved)



### 2. Training & Validation Loss

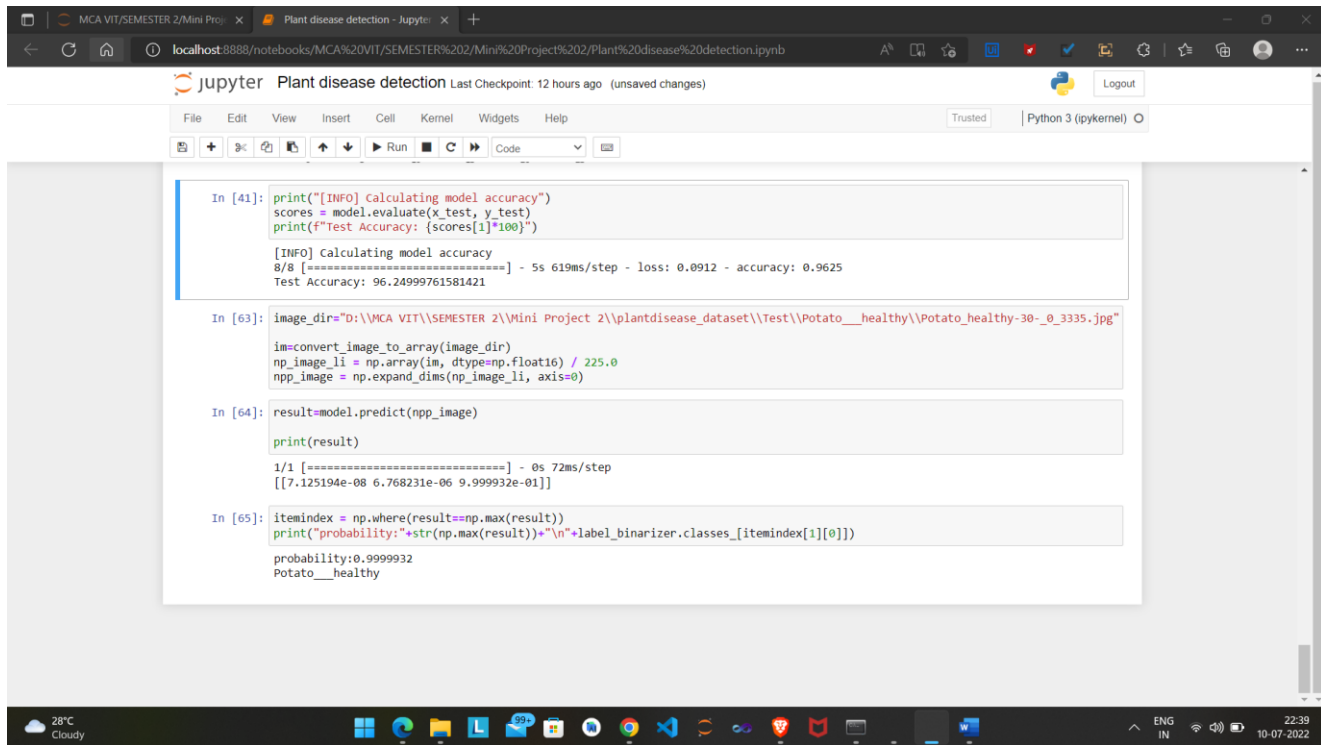
jupyter Plant disease detection Last Checkpoint: 11 hours ago (autosaved)



```
In [41]: print("[INFO] Calculating model accuracy")
scores = model.evaluate(x_test, y_test)
print(f"Test Accuracy: {scores[1]*100}")

[INFO] Calculating model accuracy
8/8 [=====] - 5s 619ms/step - loss: 0.0912 - accuracy: 0.9625
Test Accuracy: 96.24999761581421
```

### 3. Output



The screenshot displays a Jupyter Notebook interface with the following code and output:

```
In [41]: print("[INFO] calculating model accuracy")
scores = model.evaluate(x_test, y_test)
print(f"Test Accuracy: {scores[1]*100}")

[INFO] calculating model accuracy
8/8 [*****] - 5s 619ms/step - loss: 0.0912 - accuracy: 0.9625
Test Accuracy: 96.24999761581421
```

```
In [63]: image_dir="D:\\MCA VIT\\SEMESTER 2\\Mini Project 2\\plantdisease_dataset\\Test\\Potato__healthy\\Potato_healthy-30-_0_3335.jpg"
im=convert_image_to_array(image_dir)
np_image_li = np.array(im, dtype=np.float16) / 225.0
npp_image = np.expand_dims(np_image_li, axis=0)
```

```
In [64]: result=model.predict(npp_image)
print(result)

1/1 [*****] - 0s 72ms/step
[[7.125194e-08 6.768231e-06 9.999932e-01]]
```

```
In [65]: itemindex = np.where(result==np.max(result))
print("probability:"+str(np.max(result))+ "\n"+label_binarizer.classes_[itemindex[1][0]])

probability:0.9999932
Potato__healthy
```

The bottom of the image shows a Windows taskbar with the date 10-07-2022 and time 22:39.

## 3.2.5

# LINES OF CODE

```
!pip install opencv-python
```

```
conda install keras
```

```
conda deactivate
```

```
!pip install tensorflow
```

```
conda activate tf
```

```
import numpy as np
import pickle
import cv2
from os import listdir
from sklearn.preprocessing import LabelBinarizer
from keras.models import Sequential
from tensorflow.keras.layers import BatchNormalization
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D
from keras.layers.core import Activation, Flatten, Dropout, Dense
from keras import backend as K
from keras.preprocessing.image import ImageDataGenerator
from keras.optimizers import Adam
from keras.preprocessing import image
from tensorflow.keras.utils import img_to_array
from sklearn.preprocessing import MultiLabelBinarizer
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
```

```
EPOCHS = 25
```

```
INIT_LR = 1e-3
```

```
BS = 32
```

```
default_image_size = tuple((256, 256))
```

```
image_size = 0
```

```
directory_root = 'D:\MCA VIT\SEMESTER 2\Mini Project  
2\plantdisease_dataset'
```

```
width=256
```

```
height=256
depth=3
```

```
def convert_image_to_array(image_dir):
    try:
        image = cv2.imread(image_dir)
        if image is not None :
            image = cv2.resize(image, default_image_size)
            return img_to_array(image)
        else :
            return np.array([])
    except Exception as e:
        print(f"Error : {e}")
        return None
```

```
image_list, label_list = [], []
try:
    print("[INFO] Loading images ...")
    root_dir = listdir(directory_root)
    for directory in root_dir :
        # remove .DS_Store from list
        if directory == ".DS_Store" :
            root_dir.remove(directory)

    for plant_folder in root_dir :
        plant_disease_folder_list = listdir(f"{directory_root}/{plant_folder}")

        for disease_folder in plant_disease_folder_list :
            # remove .DS_Store from list
            if disease_folder == ".DS_Store" :
                plant_disease_folder_list.remove(disease_folder)

        for plant_disease_folder in plant_disease_folder_list:
            print(f"[INFO] Processing {plant_disease_folder} ...")
            plant_disease_image_list =
listdir(f"{directory_root}/{plant_folder}/{plant_disease_folder}/")

            for single_plant_disease_image in plant_disease_image_list :
                if single_plant_disease_image == ".DS_Store" :
                    plant_disease_image_list.remove(single_plant_disease_image)
```



```
for image in plant_disease_image_list[:200]:
    image_directory =
f"{directory_root}/{plant_folder}/{plant_disease_folder}/{image}"
    if image_directory.endswith(".jpg") == True or
image_directory.endswith(".JPG") == True:
        image_list.append(convert_image_to_array(image_directory))
        label_list.append(plant_disease_folder)
    print("[INFO] Image loading completed")
except Exception as e:
    print(f"Error : {e}")

image_size = len(image_list)

label_binarizer = LabelBinarizer()
image_labels = label_binarizer.fit_transform(label_list)
pickle.dump(label_binarizer,open('label_transform.pkl', 'wb'))
n_classes = len(label_binarizer.classes_)

print(label_binarizer.classes_)

np_image_list = np.array(image_list, dtype=np.float16) / 225.0

print("[INFO] Splitting data to train, test")
x_train, x_test, y_train, y_test = train_test_split(np_image_list,
image_labels, test_size=0.2, random_state = 42)

aug = ImageDataGenerator(
    rotation_range=25, width_shift_range=0.1,
    height_shift_range=0.1, shear_range=0.2,
    zoom_range=0.2, horizontal_flip=True,
    fill_mode="nearest")

model = Sequential()
inputShape = (height, width, depth)
chanDim = -1
if K.image_data_format() == "channels_first":
    inputShape = (depth, height, width)
    chanDim = 1
model.add(Conv2D(32, (3, 3), padding="same", input_shape=inputShape))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
```

```
model.add(MaxPooling2D(pool_size=(3, 3)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(Conv2D(64, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(128, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(Conv2D(128, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(1024))
model.add(Activation("relu"))
model.add(BatchNormalization())
model.add(Dropout(0.5))
model.add(Dense(n_classes))
model.add(Activation("softmax"))

model.summary()

opt = Adam(learning_rate=INIT_LR, decay=INIT_LR / EPOCHS)
# distribution
model.compile(loss="binary_crossentropy",
optimizer=opt,metrics=["accuracy"])
# train the network
print("[INFO] training network...")

history = model.fit_generator(
    aug.flow(x_train, y_train, batch_size=BS),
    validation_data=(x_test, y_test),
    steps_per_epoch=len(x_train) // BS,
    epochs=EPOCHS, verbose=1
)
```

```
accuracy = history.history['accuracy']
val_accuracy = history.history['val_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(accuracy) + 1)
#Train and validation accuracy
plt.plot(epochs, accuracy, 'b', label='Training accuracy')
plt.plot(epochs, val_accuracy, 'r', label='Validation accuracy')
plt.title("Training and Validation accuracy")
plt.legend()
plt.figure()
#Train and validation loss
plt.plot(epochs, loss, 'b', label='Training loss')
plt.plot(epochs, val_loss, 'r', label='Validation loss')
plt.title("Training and Validation loss")
plt.legend()
plt.show()

print("[INFO] Calculating model accuracy")
scores = model.evaluate(x_test, y_test)
print(f"Test Accuracy: {scores[1]*100}")

image_dir="D:\\MCA VIT\\SEMESTER 2\\Mini Project
2\\plantdisease_dataset\\Test\\Potato___healthy\\Potato_healthy-30-
_0_3335.jpg"
im=convert_image_to_array(image_dir)
np_image_li = np.array(im, dtype=np.float16) / 225.0
npp_image = np.expand_dims(np_image_li, axis=0)

result=model.predict(npp_image)
print(result)

itemindex = np.where(result==np.max(result))
print("probability:"+str(np.max(result))+ "\n"+label_binarizer.classes_[itemi
ndex[1][0]])
```

# TESTING

## 4. SYSTEM TESTING

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. There are various types of testing, explained in the following sections.

### 4.1 TESTING OBJECTIVE

- ❖ Testing is a process of executing a program with the intent of finding an error.
- ❖ A good test case is one that has a high probability of finding an as yet undiscovered error.
- ❖ A successful test one that uncovers an as –yet undiscovered error.

The above objectives imply a dramatic change in viewpoint. They move counter to the commonly held view that a successful test is one in which no errors are found. Testing cannot show the absence of detects; it can only show that software errors are present.

## 4.2

# WHITE BOX TESTING

White-box testing (also known as clear box testing, glass box testing, transparent box testing, or structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality as in Black Box Testing. Using white box testing methods, the software engineer can derive test cases that:

- Guarantee that all independent paths within a module have been exercised at least once.
- Exercise all logical decisions on their true and false sides.
- Execute all loops at their boundaries and within their validity.
- Logical errors and incorrect assumptions are inversely proportional to the probability that a program path will be executed.
- Often believe that logical path not likely to execute when, in fact, it may be executed on regular basis.

## 4.3

# LEVELS OF TESTING

The different levels of testing that are to be conducted are:

1. Code Testing
2. Program Testing
3. System Testing

### 1. Code Testing

The code test has been conducted to test the logic of the program. Here, we have tested with all possible combinations of data to find out logical errors. The code testing is done thoroughly with all possible data available with library

## **2. Program Testing**

Program testing is also called unit testing. The modules in the system are integrated to perform the specific function. The modules have been tested independently, later Assembled and tested thoroughly for integration between different modules.

## **3. System Testing**

System testing has been conducted to test the integration of each module in the system. It is used to find discrepancies between the system and its original objective. It is found that there is an agreement between current specifications and system documentation. Software Testing is carried out in three steps.

The first step includes unit testing where in each module is tested to provide his correctness, validity and also determine any missing operations. Errors are noted down and corrected immediately. Unit testing is the import and major part of the project. So, errors are rectified easily in particular module and program clarity is increased. In this project entire system is divided into several modules and is developed individually. So, unit testing is conducted to individual modules.

The second step includes integration testing. It need not be the case, the software whose modules when run individually and showing perfect results, will also show perfect results when run as a whole. The individual modules are clipped under this major module and tested again and the results are verified.

The final step involves validation and testing which determines the software functions as the user expected. Here also there may be some modifications. In the completion of the project, it is satisfied fully by the user.

## 4.4

# INTEGRATION TEST PLAN

Data can be lost across an interface, one module can have an adverse effort on the other sub functions, when combined, may not produce the desired major functions. Integrated testing is the systematic testing for constructing the uncover errors within the interface. The testing was done with sample data. The developed system has run successfully for this sample data. The need for integrated test is to find the overall system performance

## 4.5

# UNIT TESTING

In the unit testing we test each module individually and integrate with the overall system. Unit testing focuses verification efforts on the smallest unit of software design in the module. This is also known as module testing. The module of the system is tested separately. This testing is carried out during programming stage itself. In the testing step each module is found to work satisfactorily as regard to expected output from the module. There are some validation checks for fields also. For example, the validation check is done for varying the user input given by the user which validity of the data entered. It is very easy to find error debut the system

### Unit Testing Table:

Function Name	Tests Results
Uploading Image	Tested for uploading different types and sizes of images.
Detecting Disease	Tested for different images of plant leaves and diseases Bacterial Spot, yellow leaf curl virus.



## 4.6

# OUTPUT TESTING

After performance of the validation testing, the next step is output testing. The output displayed or generated by the system under consideration is tested by asking the user about the format required by system.

### Output Testing Table:

Functionality to be tested	Input	Tests Results
Working of change of file location	User convenience to access images stored	The folder in which the image is stored can be uploaded

## 5. CONCLUSION

The proposed system was developed taking in mind the benefits of the farmers and agricultural sector. The developed system can detect disease in plant and also provide the remedy that can be taken against the disease. By proper knowledge of the disease and the remedy can be taken for improving the health of the plant. The proposed system is based on python and gives an accuracy of around 97%. The accuracy and the speed can be increased by use of Googles GPU for processing. The system can be installed on Drones so that aerial surveillances of crop fields can be done.

## **6. LIMITATIONS**

1. This project only works for stand-alone it does not work on MAN and WAN.
2. This system should not be taken as tool to assist the vendors in selling or selecting of product in other words is not a type of expert system.
3. No direct interaction between customer and administrator.
4. No option to select other product which is not add this site. Only added product can be chosen.
5. Cost of product can be varied from customer to customer, no standard and fixed cost.

## **7. FUTURE ENHANCEMENTS**

1. In classifier the disease can be identified and solution for disease can be found. This information is sent to the farmer through GSM modem
2. To improve recognition rate of final classification process hybrid algorithms like Artificial Neural Network, Bayes classifier, Fuzzy Logic can also be used.
3. Mobile application can be developed which is handy and easy to use.
4. An extension of this work will focus on automatically estimating the severity of the detected disease.
5. As future enhancement of the project is to develop the open multimedia (Audio/Video) about the diseases and their solution automatically once the disease is detected.

## 8. REFERENCES

- ❖ <https://www.kaggle.com/emmarex/plant-disease-detection-using-keras>
- ❖ [https://www.researchgate.net/publication/344155605\\_Plant\\_Disease\\_Detection\\_using\\_CNN/link/5f56505c458515e96d372176/download](https://www.researchgate.net/publication/344155605_Plant_Disease_Detection_using_CNN/link/5f56505c458515e96d372176/download)
- ❖ Prof. Sanjay, B. Dhaygude, “Agricultural plant Leaf Disease Detection Using Image Processing”, International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering 2.1(2013):599-602.
- ❖ Amandeep Singh ,Maninder Lal Singh, “Automated Color Prediction of Paddy Crop Leaf using Image Processing”, International Conference on Technological Innovations in ICT for Agriculture and Rural Development (TIAR 2015).IEEE,2015.
- ❖ Malvika Ranjan, Manasi Rajiv Weginwar, “detection and classification of leaf disease using artificial neural network”, International Journal of Technical Research and Applications, 2015.
- ❖ P.Revathi, M.Hemalatha, “Advance Computing Enrichment Evaluation of Cotton Leaf Spot Disease Detection Using Image Edge detection.”,2012 Third International Conference on Computing, Communication and Networking Technologies(ICCNT'12).IEEE,2012.

