Plant Disease Detection and Classification using Android Application

Authors

Mubashar Ali (UET/17-CS-16)

Muhammad Tanveer (UET/17-CS-17)

Supervised By

Mr. Muhammad Wakeel Ahmad

Department of Computer Science

UNIVERSITY OF ENGINEERING AND TECHNOLOGY TAXILA

DEDICATION

Dedicated to

“The Teacher of the Universe”

(Peace be Upon Him)

With whose existence and
by having the charity of His knowledge
the universe got illuminated with the light of insight
and wisdom
and
the journey of human enlightenment was made possible.

ACKNOWLEDGEMENT

All praise to Allah Almighty, Lord of all the worlds, the Merciful and the most Beneficent, who gave us all strengths, thoughts and co-operative people to make us to accomplish this goal and fulfill all the required functionalities.

This was all not possible without the proper guidance, continuous appreciation and moral support by our honorable Supervisor **Mr. Muhammad Wakeel Ahmad.** He was always there whenever we need his help and guidance. We are really thankful to him who made our concepts clear and also thankful to Chairman of Computer Science department, **Dr. Syed Aun Irtiza** and **Dr. Syed Adnan** for helping us.

On the end, we would like to acknowledge all of the assistance and influences of UNIVERSITY OF ENGINEERING AND TECHNOLOGY, TAXILA for supporting us with all that is needed starting from the books, and ending with the full attention that it is providing us with, to help us to be professionals in the field of Computer Science.

DECLARATION

We hereby declare that we have developed this project and accompanied report entirely on the basis of our personal efforts and knowledge. Not any of the portion of the application work presented has been submitted of any application for any other prerequisite or degree of this or any other university or institute of learning.

Students Name & Signature

Student 1	Student 2
M. Tanveer	Mubashar Ali
_____	_____

CERTIFICATE OF APPROVAL

It is to certify that the final year project of BS Computer Science "**Plant Disease Detection and Classification using Android Application"** was developed by **Muhammad Tanveer and Mubashar Ali** under the supervision of honorable "**Mr. Muhammad Wakeel Ahmad**" and that in his opinion, it is in scope, fully adequacy and quality of the degree of Bachelors of Science in Computer Sciences.

External Examiner

Supervisor

Mr. Rao Wakeel Ahmad Lecturer,
Computer Science Department UET,
Taxila

Chairman of Department

Dr. Syed Aun Irtiza
Department of Computer Science UET,
Taxila

ABSTRACT

Plant diseases causes many significant damages and losses in crops all around in the world. Some suitable measures on disease detection should be introduced to prevent damages and minimize fatalities. Early Detection of Disease helps in increasing the crop productivity as well as in minimizing expense. Technical methods using machine learning and computer vision are actively researched to achieve intelligence farming by early detection on plant diseases. The accuracy of object detection and recognition systems has been drastically improved by the recent development in Deep Neural Networks. By using these systems and implementation of computer vision and machine learning techniques, plant diseases can be detected. Here we have used transfer learning-based approach to diagnose diseases of different plants using their images captured by camera devices of smartphone. Our aim is to build a market-oriented product for Plant Disease Detection, a smartphone application compatible with smartphone camera. The target group of the user is those who want a quick diagnosis on common leaf diseases at any time of the day i.e. Farmers, agricultural industries, agricultural consultants and Government Agencies & Departments.

Table of Contents

List of Figures

List of Tables

CHAPTER #1

INTRODUCTION

1 INTRODUCTION

In this first chapter, we will provide introduction of this application, software tools and techniques, problem statement, goals, future scope of this application, proposed model of application, motivation of this proposed model, relevance to courses and all tools & techniques which are used to implement this application.

1.1 Introduction & History

In agriculture sector, plants diseases and pests are a major challenge. A precise and fast recognition of plants diseases could help to perform an early treatment method though at the same time significantly reducing economic losses.

Techniques based on image processing are used to detect plant diseases without causing any secondary impact in plant. However, the accuracy is still a challenge in Computer Vision. In intelligent systems, most important issue is the use of proper dataset. Our technique is applicable on all kinds of plants but for testing, we focus on the recognition and identification of diseases that affect tomato plants (we are not working on pest detection due to lack of dataset). Economically, tomato and wheat is the most important vegetable crop around the world, and its production has been significantly increased through the years. Several diseases that affect tomato plants are the main part of this study. Testing is done on some samples of our collected dataset which contains different diseases of different plants. The diseases of plants on which testing is done are late blight, bacterial spot, early blight and many more approximately ten diseases.

It is a machine learning based technique in order to identify diseases in different plants smartphone camera. It will help the Farmers, agricultural industries, agricultural consultants and Government Agencies and Departments to perform fast diagnosis on common leaf disease at any time. Our system uses images of diseased plants taken from different tomato farms and fields thus, we don't need to analyze samples in laboratory. We collected dataset of images from fields and greenhouses using android mobile camera. Techniques like image annotation, image augmentation are applied on images in order to increase the dataset and accuracy of images. For training those images, an open-source software library called Tensor Flow is used. Farmers and Horticulturists will be able to take advantage from this application when they want early detection of plants diseases, so they can do it easily by using android application.

This App has following features:

1. Detect plant disease more precisely.
2. Save time required for infield scouting and actions.
3. Help the farmer to identify crop's progress accurately and in replanting decisions.
4. Help in preventing unnecessary waste of financial resources.
5. Smart phone diagnostic app will keep farmer updated about crop at a single click.

Since this project is android based also, so users can access this tool from multiple android devices to recognize different image targets. This application just requires images captured by various camera devices with different tenacities, such as mobile phone and other digital cameras. The goal of this project is to develop a market-oriented product for Plant Disease Diagnosis, a smartphone app compatible with both smartphone camera and digital camera. Moreover, it can easily deal with different size of objects, different illumination conditions and background differences contained in the surrounding areas of diseased plants. This is not much time consuming as compared to older ways of plant disease detection techniques. The idea is productive and as well as innovative. The Plant Disease Detection App provides a practical and real time market-oriented product that can be used in different fields without employing any complex and expensive technology. It also deliberates the possibility that a plant can be concurrently affected by more than one disease in the same plant.

1.2 Statement of the problem

The economy of agricultural countries heavily depends upon number of crops grown in the country. Moreover, the rapidly increasing population (especially those residing in rural areas) also depends upon agriculture for subsistence. In Pakistan, agriculture contributes to 24 percent of national GDP. It feeds whole rural and urban population.

But crops diseases have been a continuous threat to farmer's livings and equally endanger the food security of the world. Diseases on plants leads to the greater number of decreases in the terms of both quality and quantity of agricultural products. The reason behind this failure is, the naked eye observation of the expert/farmer is in practice, which is time consuming and not feasible for larger fields.

Transfer learning-based detection and identification of plant diseases provide hints to detect the disease in early stages. Relatively, detecting the plant diseases visually is affluent, difficult and inefficient. In addition to, it involves the expertise of trained agricultural experts and botanists.

1.3 Introduction to the Software tools and Technologies

In order to develop this final year project, following software tools are used:

1. Python
2. Tensorflow
3. Flutter
4. Google Colab
5. Android Studio

1.3.1 Why Python is used?

Python is easy to understand and its packages are free to use under GNU license. Many machine learning engineers use python. So we chose python for it.

1.3.2 Why Android Studio is used?

Android Studio is an official integrated development environment for android development. It provides code management, one-click solution for many problems faced by android developers and it also provides support for version controlling. So, we chose android studio for developing our android app.

1.3.3 Why TensorFlow is used?

Tensorflow open source and fast machine learning system that operates at large scale, having a lot of community support. It is also python based and as we decided to develop our website using python so using Tensor flow for ML was the best choice to gain more exposure of the language. [1] [1-6]

1.3.4 Why Google Colab is used?

Google Colab is an open-source service of Google which we can use to train model of Machine learning and we can access GPU by using Google Colab. it can easily run the Python language code. We can save and upload our working file on Google Drive by using Colab. It is a wonderful tool for Machine learning.

1.3.5 Why Flutter is used?

Flutter SDK is a Google's UI toolkit for making beautiful, natively amassed applications for mobile, website, and desktop from a single code. We can deploy the application of flutter on Android as well as the IOS.

1.4 Objectives

The objectives of a project are important to achieve goal. The main objectives our product- oriented project is:

1. To develop a practical, reliable and inexpensive real time application that can be used in fields.
2. To develop a market-oriented product for Plant Disease Detection, a smartphone app compatible with smartphone camera
3. It considers the possibility that there can be more than one disease same plant is affected.
4. Moreover, android application will update the user about weather forecast daily and hourly, so that user can prepare work plan carefully.

The diseases in the plants may show various physical characteristics i.e. difference in colors, shapes and forms etc. We are using a transfer learning-based approach using Object Detection API and Faster RCNN pre-trained modal for distinguishing different diseases. Based on these facts [5, 7, 9, 10], following characteristics are analyzed:

1. **Location's symptom:** Diseases affect not only leaves of the plant but other parts like stems or fruits also.
2. **Disease patterns:** Visible variations are shown on either front or back side of leaves by the symptoms of diseases.

3. **Fungus type:** Some diseases can easily be differentiated by the identification of type of fungus.

1.5 Proposed Solution

In this project we are proposed to develop smart phone-based system which would help farmers to diagnose diseases of crops earlier. Machine learning based technology will assist farmers and agricultural agencies to scout areas of fields. Farmers and Horticulturist can use smartphone as they like and monitor the fields in less time. This will also save the cost of labor associated with physical infield scouting. An early recognition of affected parts will also decrease pesticide application rates and cost. This application will increase the overall growth of crops and no need for identifying the diseases in long time. This application has user friendly interface. All communications between all the users is done through an android device. An android app is the cost-effective solution without expensive hardware or software requirement. The images acquired by Smart phone will be processed on an online server using Tensor flow and Google Colab.

A transfer learning-based approach with the help of TensorFlow and Object Detection API is used for this purpose [4]. A visual representation of it is shown below in **Figure 1.1**

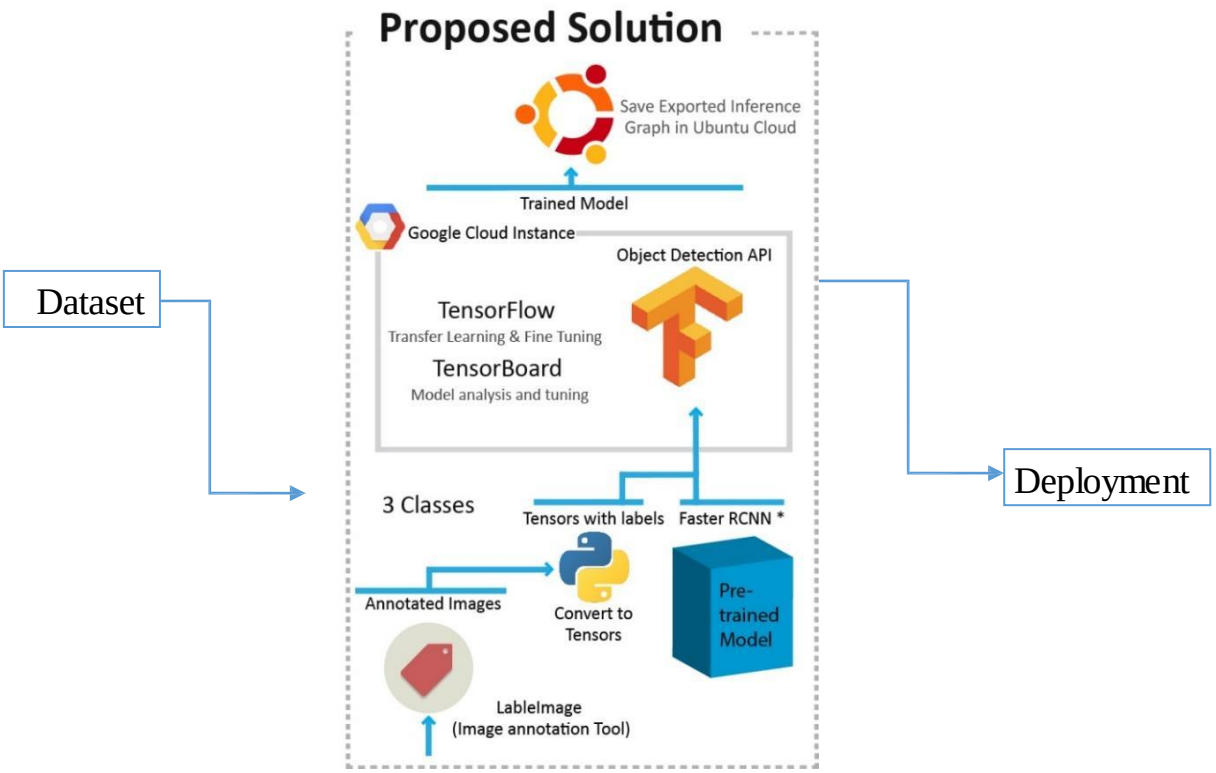


Figure 1.1 Proposed Solution

https://www.tensorflow.org/

1.6 Motivation

Plant Disease Detection App saves significant amount of time and money of farmers and agricultural industries. Despite the existence of many learning Applications, the following features are the motivational source of our project:

1. Cost optimization in terms of developing and deploying the overall application.
2. This application is really very helpful and easy to use for any layman farmer to detect diseases of plants.
3. We are developing a market-oriented product i.e. an android application
4. Android application also show weather forecasting details.
5. All the update is given to the farmer on one click via mobile app.

1.7 Scope of Proposed Solution

This project is designed to assist individuals (farmers and agricultural specialists) to get informs about their fields by using machine learning and artificial intelligence techniques. Someone is doing this from this product whenever Farmers and Horticulturist have shortage of time and they get all the informs about their fields using this product. It is a market oriented real time product that is less costly and easy to implement. It can work with any image through various camera devices or upload from the system with different resolutions. Moreover, it can effectively handle background variation and different illumination conditions. It considers the possibility that there can be more than one disease same plant is affected.

1.8 Relevance to Courses

1.8.1 Software Design Project-I (CS-400)

This course aided us to evaluate our development methodologies which are used in our project.

1.8.2 Introduction to Database Systems (CS-203)

This course helped us in database management.

1.8.3 Smart Application Development (CS-311)

This course helped us in developing android smart application.

1.8.4 Digital Image Processing (CSC271)

This course helped us in analysis of image.

1.8.5 Computer Vision (CS-309)

This Course Helped in deployment and practical analysis of Computer Vision based Application.

1.8.6Software Engineering (CS-201)

This Corse helped in management of project and quality assurance.

1.9 Tools & Techniques

"Plant Diseases Detection" in an Android based application and a market-oriented product consists of software technologies. Software details are given in section 1.9.1. As it is a product for the Kitchen Gardening and basically work on the images of the Tomato and Wheat. We use Deep learning model for the deployment of image classification and detection. Moreover, we use Flutter for the deployment of Android Application as it has effective interface options. We use Google Colab as it is an open-source product of Google for machine learning and we can access the GPUs for the training of the Deep learning model.

Flutter is used for the interface of android application. The model we used for the plant's diseases detection is an unsupervised model for the detection of diseases in ten plants. The camera or gallery image of the leaves are uploaded to the application and the model predict the diseases along with it also provides the name of pesticide for the treatment of the disease. There is another awesome functionality which is weather forecast. It is also necessary for the farmers or horticulturist to know about the weather before spray the pesticide because if there is rain then the use of pesticide will not effective.

1.9.1 Software Details

1. Google Colab
2. Tensor flow
3. Android Studio
4. Python
5 Flutter
5. Cloud
6. SQLite
7. Excel
8. Mongo DB

CHAPTER # 2

Literature Review

2 Literature Review

In this chapter, we will focus on what is Tensor Flow? What is Machine Learning? What is Artificial Intelligence? How we'll implement this application in real world? Why we need it, if other applications exist? What is the pros and cons of existing applications and proposed application?

2.1 What is Tensor Flow?

According to official website of tensor flow

Tensor Flow is an open-source software library for high recital numerical computation. Its flexible architecture allows easy deployment of computation across a diversity of platforms (CPUs, GPUs, TPUs), and from desktops to groups of servers to mobile and edge devices. Initially developed by researchers and engineers from the Google Brain team within Google's AI organization, it comes with solid support for machine learning and deep learning and the flexible arithmetic computation core is used across many other scientific domains.

2.2 What is Machine Learning?

According to medium.com

Machine learning is subfield of artificial intelligence. Its objective is to allow computers to learn on their own. A machine's learning algorithm permits it to identify patterns in experiential data, build models that elucidate the world, and predict things without having overt pre-programmed rules and models.

2.3 What is Artificial Intelligence?

According to medium.com

Artificial intelligence is the examination of agents that observe the world around them, form plans, and make choices to achieve their goals. Its foundations contain mathematics, logic, philosophy, probability, linguistics, neuroscience, and decision theory. Many fields fall under the sunshade of AI, such as computer vision, robotics, machine learning, and natural language processing.

2.4 Implementation in real world

This application can be implemented by Farmers, agricultural industries, agricultural consultants, Horticulturist and Government Agencies & Departments to perform rapid diagnosis on common leaf diseases at any time. Anyone will be able to download the android version of this application from Play Store after some time. Users can easily use this application by having friendly user interface. User will find it efficient and less time consuming.

2.5 Existing Applications

2.5.1 Assess

There is an expensive and commercial software, called Assess, which can quickly and easily tell the quantity of disease's symptoms of plants.

2.5.2 Leaf snap

There is an excellent iPhone app, called Leaf snap, which help the user to identify plant species from images of leaves using visual recognition.

2.5.3 Leaf Doctor

This is an android app, which performs quantitative assessments for plant diseases on diseased leaves. User collect snaps of diseases plants and calculate the percentage of diseased tissue. The algorithm employs user-specific values for up to 8 colors of healthy tissues in picture. The color of each pixel is then evaluated for its distance from the healthy colors and assign a status of healthy or diseased leaf. The assessment data may be sent by email to the recipients.

2.6 Drawbacks of Existing Applications

Drawbacks in above mentioned existing application are as follows:

1. Low accuracy.
2. First capture image and upload it.
3. Hardware dependent requires a computer system, which is inconvenient for the user to use any time.
4. Weather Forecast is not available.

2.7 The Necessity

There are some necessities for this market-oriented product, android application to conduct meetings are as follows:

2.7.1 Dataset

Must need to have images as the developer has used in project. We have collected a large number images of tomato plants, which contain the diseases depending on the farm where they were taken from. Some farms presented more infected plants by canker, early blight, late blight and plague, and others by leaf mold, bacterial spot.

Figure 1.1 Tomato late Blight

https://extension.umn.edu/diseases/late-blight

farm but in less presence. Our technique is applicable on all kinds of plats and its diseases, but testing is done on the tomato plants and wheat plants affected by late blight, bacterial spot early blight and eight other diseases of Tomato, Grapes, Apple and eight more vegetables and fruits. The infections of Late blight produce firm lesions, dark brown as shown in **Figure 2.1** which may destroy the entire tomato fruit.

2.7.2 Accurate Results

Plant Disease Detection app must show accurate results.

2.7.3 Better and quick Analysis

Plant Disease Detector App must provide the facility of quick and efficient analysis and detection of diseased plants as compared to the previous conventional ways of analysis.

2.8Application Software Architecture

The problems which are befallen in the existing application are overwhelmed in proposed application. In this application we are implemented all of the functionality by using any mobile device. The proposed application supports fast and efficient detection and analysis of diseased images as compared to existing applications.

2.8.1 Application Architecture

Our App consists of five features:

1. Machine vision-based technology using transfer learning and fine tuning of Inception-V3.
2. A smartphone based low-cost android application for monitoring small fields.
3. App Detect the Diseases of different plants almost 5 plants and 10 Diseases.
4. Android application also forecasts weather daily & hourly and predicts accurate weather forecasting.
5. Reduction in the time and work load.

How we achieved these is explained in **Figure 2.2** below.

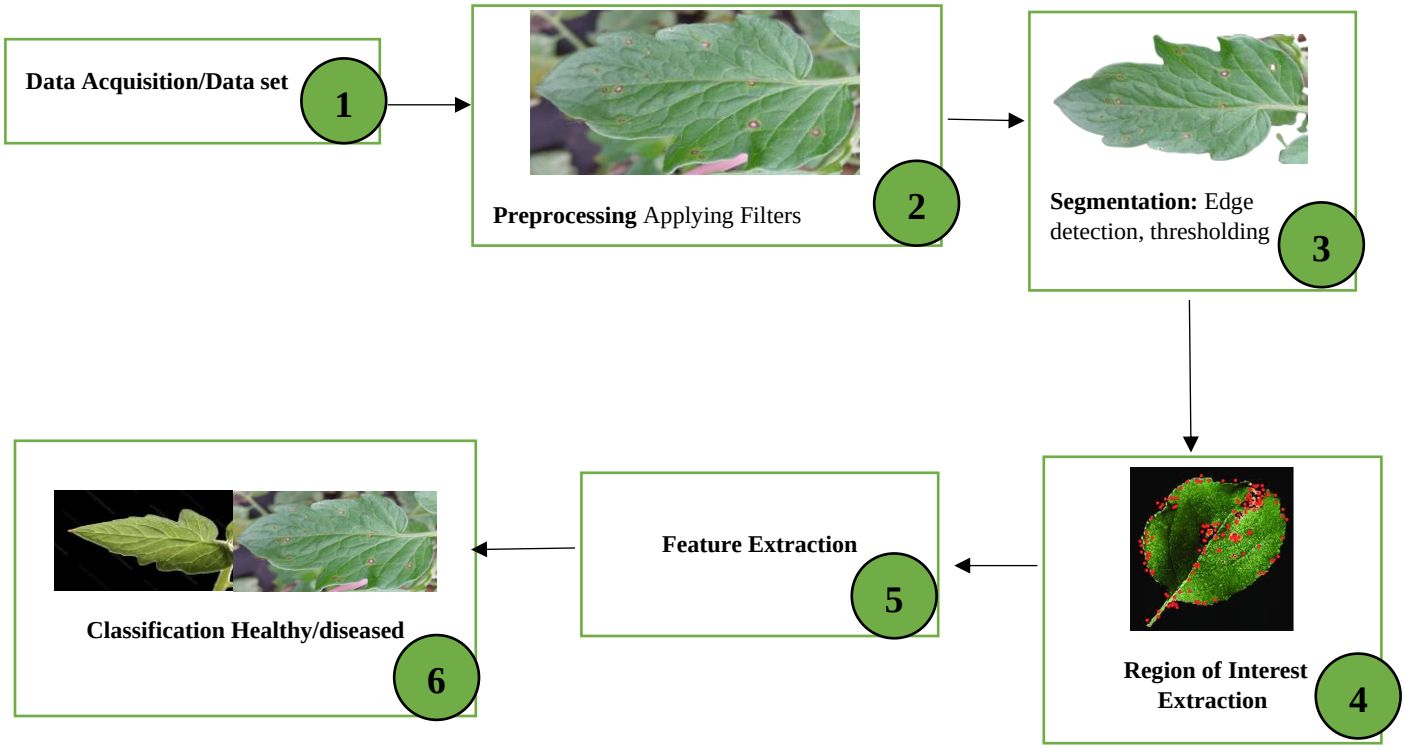


Figure 2.2 Application Architecture

CHAPTER # 3

REQIREMETNS ANALYSIS

3 REQUIREMENTS ANALYSIS

In this chapter requirements exploration, feasibility study, planning, forecasting, modeling, scheduling and design of the project is discussed. For the deployment of any project, the major problem is requirement congregation. We will also emphasis on functional and non-functional requirements. The process for gathering requirements has its personal defined procedure according to the complexity of the application. To describe project schedule and processing, different models and techniques also absorbed on this chapter.

3.1 Stake holders

Stake holders are the final users of project. This project is basically for the kitchen gardening so our stake holders are following

- Farmers
- Horticulturists
- People doing Kitchen Gardening

3.2 Requirements Gathering Methods

A requirement can be defined as a condition or ability that must be processed by a product or an application. Methods that can be used for collecting requirements are as follows:

- By analysis and observations
- Using software tools
- Using techniques of Artificial Intelligence and Machine Learning

The methods we have used to collect requirements are observations and knowing about AI new techniques.

3.3 Requirement Analysis

Requirement's analysis is the procedure of development, forecasting and studying the overall former requirements of the application requirements. Requirement's analysis is divided into two parts based on our project:

1. Functional Requirements
2. Non-Functional Requirements

3.3.1 Functional Requirements

Following are the functional requirements of our project:

RQ-F1: Must provide disease diagnosis feature natively using smartphone camera.
RQ-F2: Must show weather forecast.
RQ-F3: Ability for the user to read about different diseases and counter measures.
RQ-F4: Must purpose the medicine for the disease.
RQ-F5: Must login and register new user

3.3.2 Non-Functional Requirements

Non-functional requirements are the restrictions or checks on the services and functions provided by an application such as constraints on the development standards/process and restrictions of time etc. Non-Functional requirements of App are as follows:

RQ-NF1: Application must provide better response and performance.
RQ-NF2: Application must provide good quality results
RQ-NF3: Application must be effectual.
RQ-NF4: Application must be user interactive.
RQ-NF5: Application shall detect plant diseases more precisely
RQ-NF6: Application must be less time consuming.

3.4 Application Quality Attribute

3.4.1 Availability

Application must be approachable and available at every time.

3.4.2 Maintainability

Making modifications or upgrade-potential in the utility will not be so much difficult. A little

knowledge about machine learning, transfer learning, python and Tensor flow will be enough to make changes or upgrading the software/utility.

3.4.3 Consistency

When a developer is updating information, consistency must be there.

3.4.4 Portability

Plant Disease Detection android based mobile application there is no problem in portability to use it cross the android versions and systems for processing.

3.3.5 Database Requirements

In this section, we are using MySQL and SQLite for storing the user's data and classification results.

3.5 Use Cases

We draw several use cases for analysis of our project. Some of the use case are as follows

3.5.1 UC1: Starting of Application

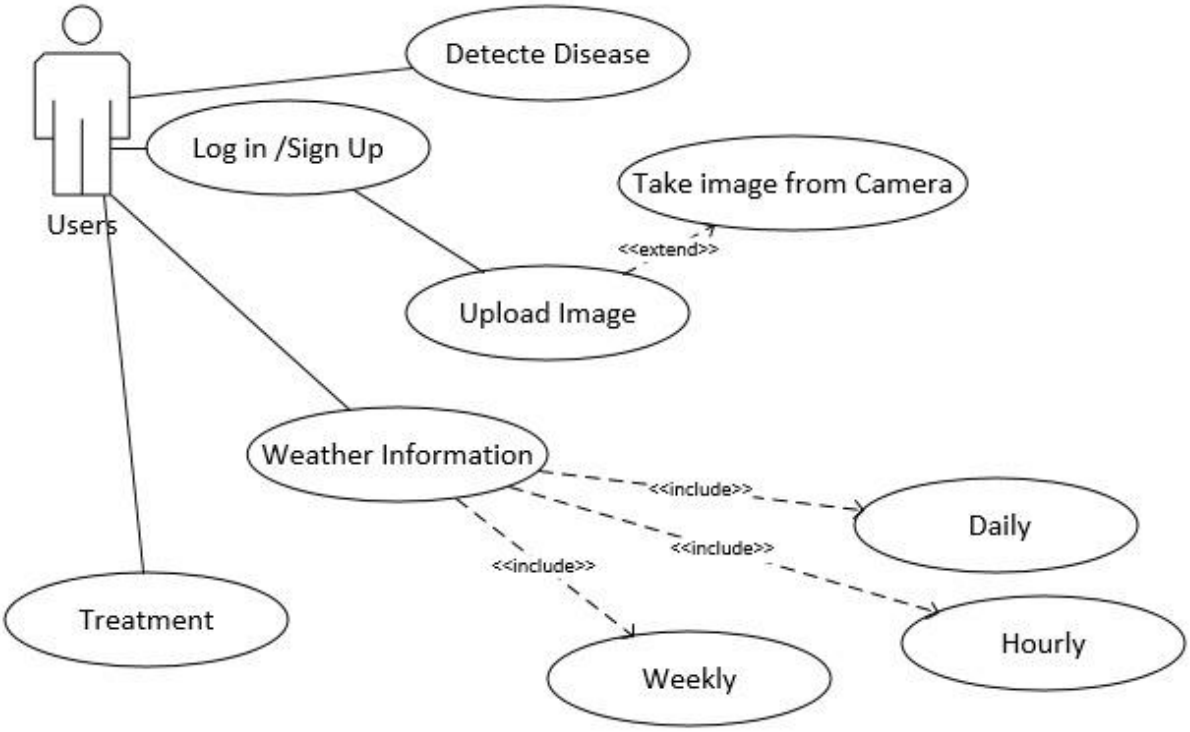


Figure 1.1 Use Case Diagram

3.6 Sequence Diagram

In Sequence diagram we represent the requirement of current system and how user interact with the system. It also represents how different interact with each other. Like login how our system authenticates a user.

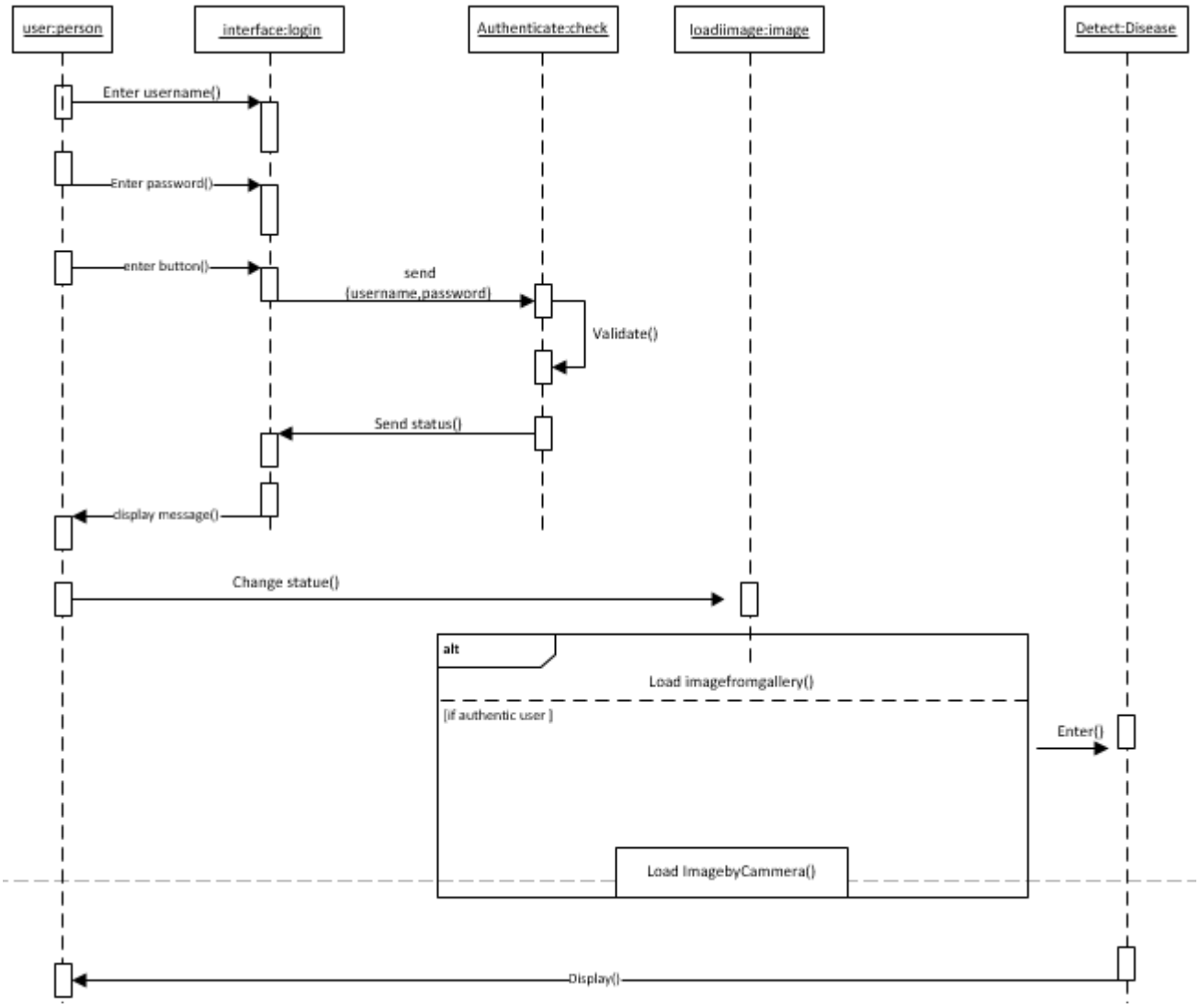


Figure 3.2 Sequence Diagram

3.7 Activity Diagram

An activity diagram is alike to a flowchart or a dataflow diagram showing a series of actions or flow of control in a system. An explanation use case step can also be drawn. It can consist of sequential and concurrent activity model.

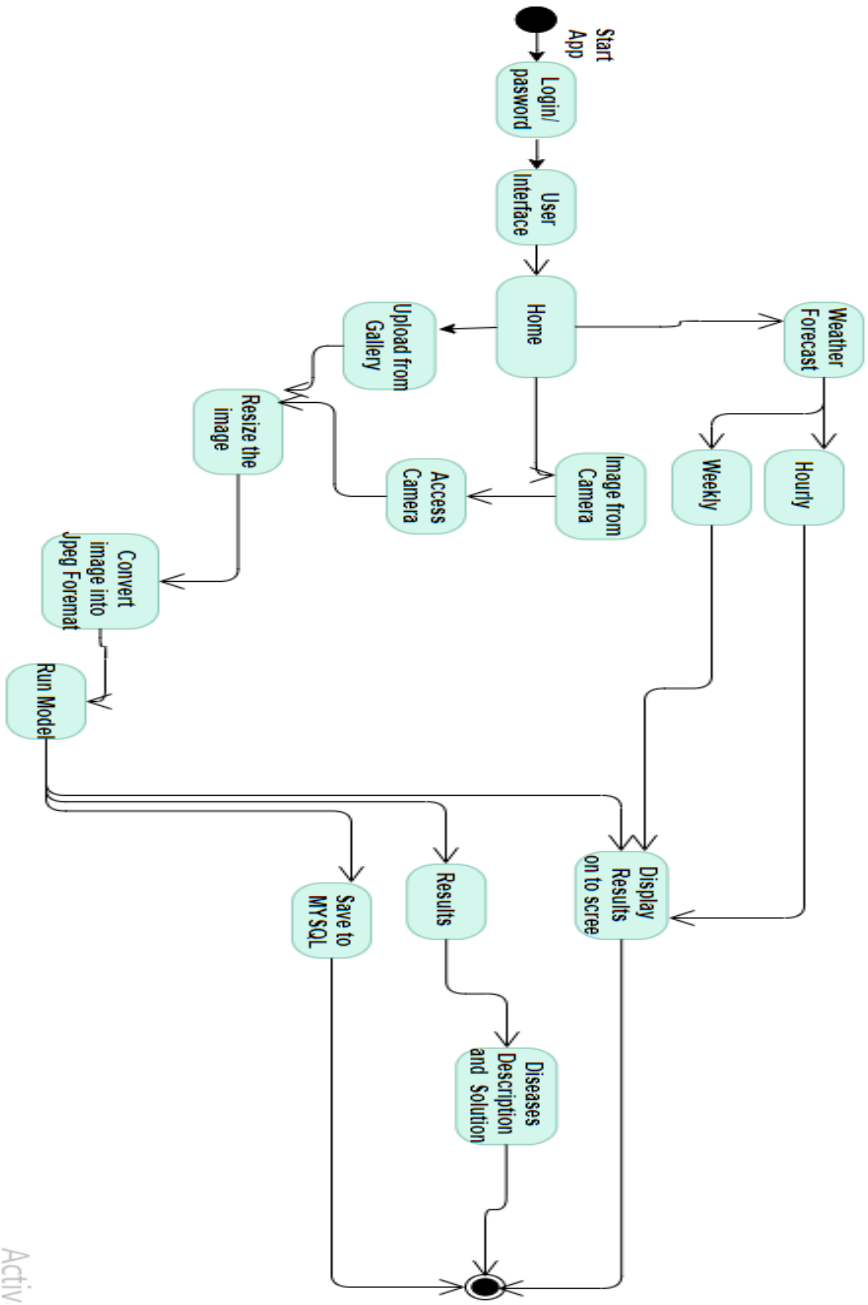


***Figure 3.3** Activity Diagram*

3.8 Data Flow Diagram

A DFD displays the flow of data across the app. The information about process timing and whether it is background process or not, doesn't defined in DFD.

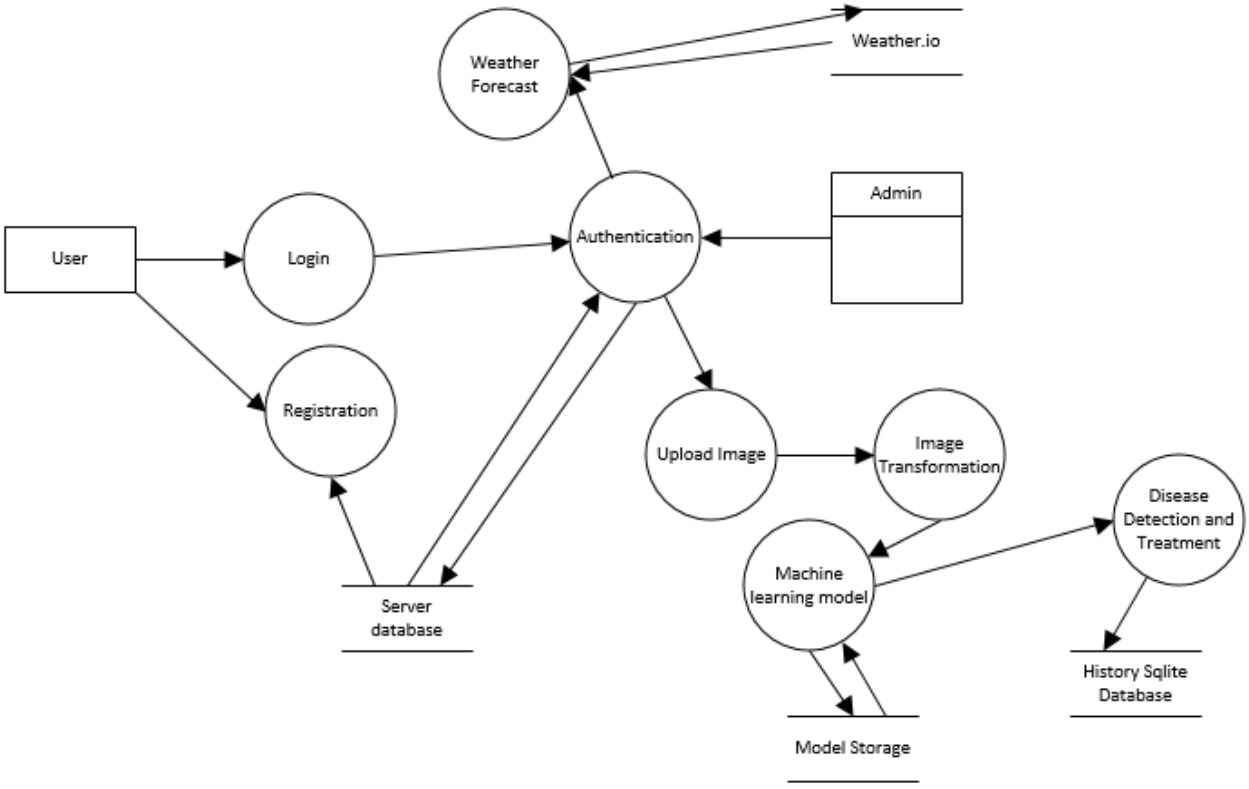


Figure 3.4 DFD Diagram

3.9 Class Diagram

A class diagram defines the relationships and code dependencies between the classes in unified modeling language. Methods and variables are defined by classes and act as a specific entity in a program. Basically, class diagram is more useful for object-oriented programming (oop). This class diagram is defining our mobile application classes and characteristics.

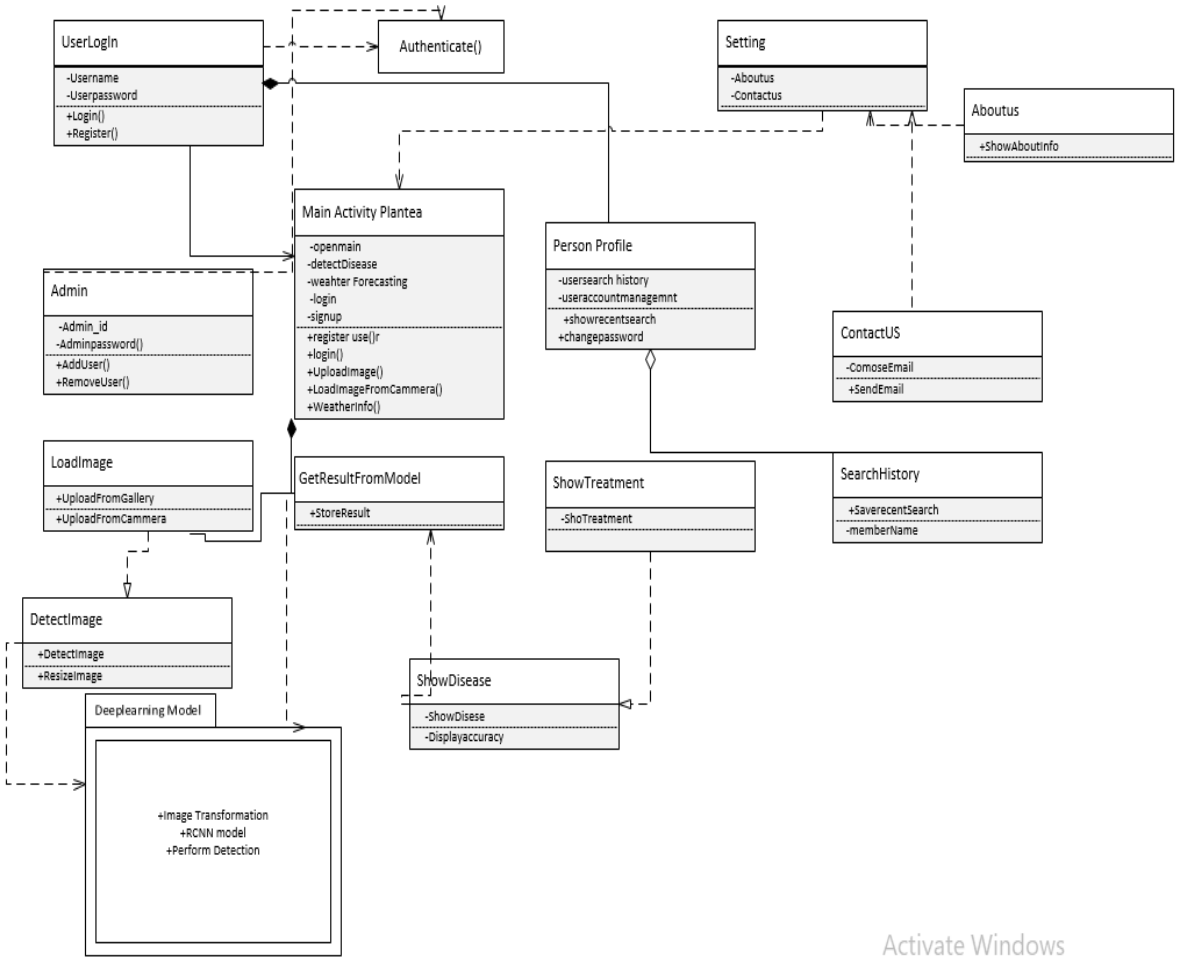


Figure 3.5 *Class Diagram*

3.9.1 Class Diagram for Weather Forecasting Module

This part of class diagram is representing the weather forecasting module, it's classes and characteristics.

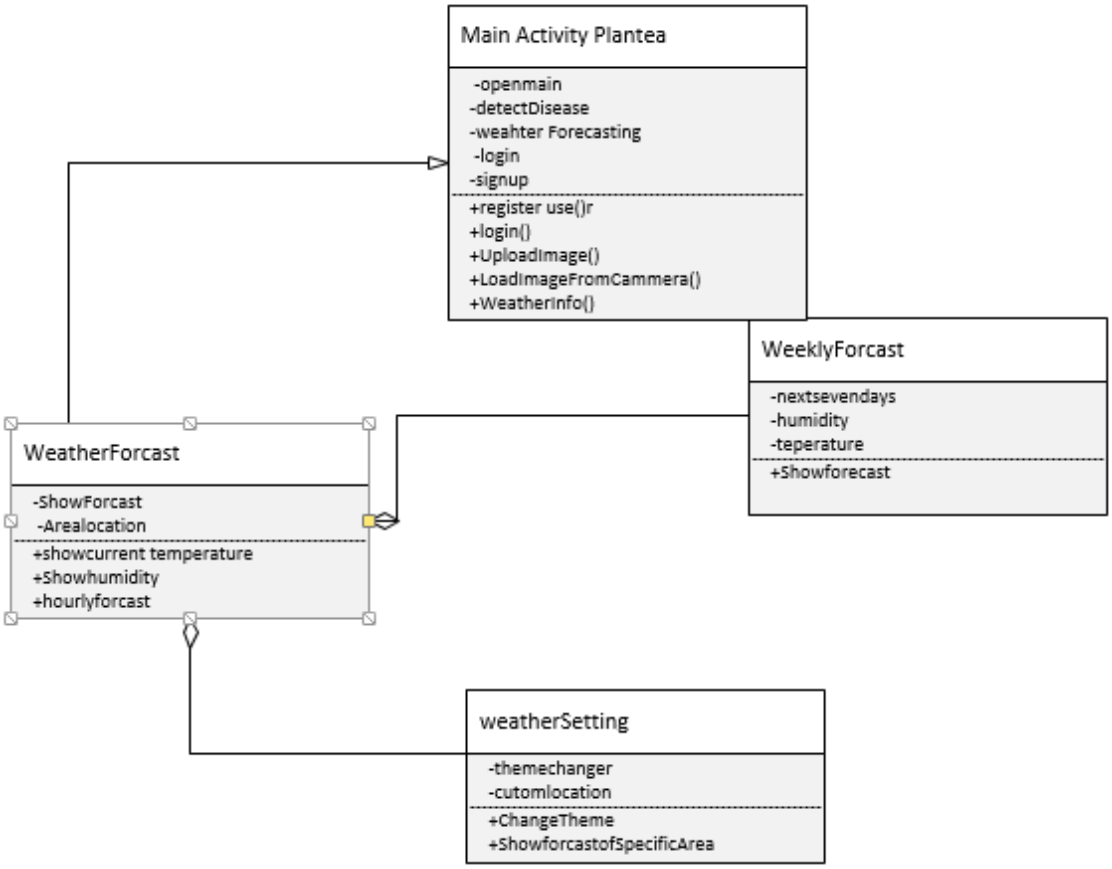


Figure 3.6.2 Use Case Weather

3.10 Software Life Cycle:

Software Development Life Cycle (SDLC) is a process adopted by the software industry to design, deploy and test high quality soft wares. The SDLC aims to produce a high-quality software that can meets or exceeds customer expectations, reaches completion within times and cost estimations. In our project we first analyze our requirements like image acquisition, effective model training, flutter applications interface and the weather forecast. We then properly gather requirements. Then we adopted the policy of step-by-step design like we first deploy a single module. Then test it after that we move to the next module. First, we train the model next we test the module after that we design the interface. Then move to next phase. In stabilization phase we test all the modules and remove all the ambiguities. In deployment phase we deploy the application on the Google Play Store and check the reviews of the downloaders (end user). At last phase which is maintenance we update the application

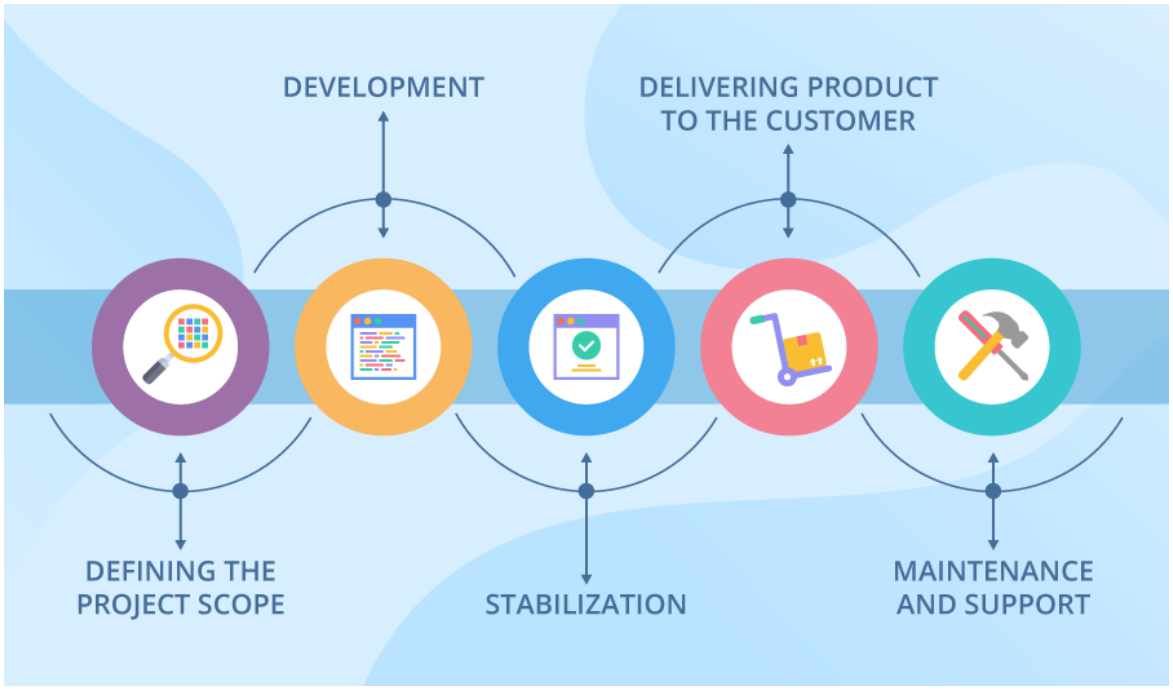


Figure 3.7 Software life cycle

Chapter # 4
System Design

4 SYSTEM DESIGN

In this chapter, we will discuss that what are the existing techniques and which one we have chosen for deployment of this project in an effective way, also we will discuss advantages of adopted methodology [14, 15].

4.1 What is methodology and why we need it?

When a small or large project have started to deploy, first thing all of programmers required is method. Methodology is a technique of developing a project, in which all of the programmers collect the user's requirements, design project, deploy it, and after all this testing and maintenance of the project, in a fulfilment of user and according to the project requirements.

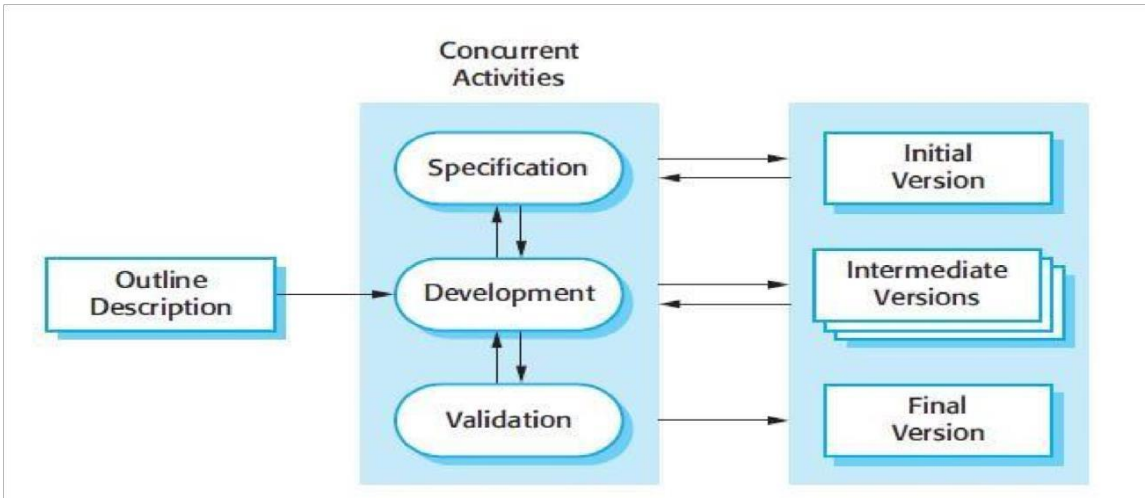


Figure 1.1 Agile Method

4.2 Adopted Methodology

Incremental model is used to deploy this project, in which we divided our work in multiple modules. All these modules are more divided into more easily managed modules which made up the authentic implementation of the requirements.

Reason behind this architecture:

- It is easy to test and debug the product during the iteration
- We incrementally develop our project and handled changes recommend by the supervisor

4.3 Project Time Allocation

Gantt chart shows task that we will perform against time. We have divided the project in small parts and select time against these parts in which we will perform specific task. Gantt chart is also describing the progress in time with graphical preview. Dividing a large project in parts make it easy to understand and interpret.

Task name	Start date	End date	Assigned	Progress	Month 1	Month 2	Month 3	Month 4	Month 5	Month 6	Month 7	Month 8	Month 9	Month 10
Complete Project	**01/10/2020**	**30/07/2021**	**Mubashar and Tanveer**	**100%**										
Idea Selection	01/10/2020	15/10/2020	Parallel	**5%**										
Review Concepts	16/10/2020	20/10/2020	Parallel	**7%**										
Check Compliance of all parts	21/10/2020	25/10/2020	Parallel	**8%**										
Final Approval	26/10/2020	20/11/2020	Parallel	**10%**										
Data Acquisition	21/11/2020	5/12/2020	M. Tanveer	**17%**										
Mobile Application Design	06/12/2020	30/12/2020	Mubashar Ali	**25%**										
Pre-Processing	01/01/2021	31/01/2021	M. Tanveer	**35%**										
Segmentation	01/02/2021	15/03/2021	Mubashar Ali	**45%**										
Region of Interest Extraction	16/03/2021	15/04/2021	M. Tanveer	**60%**										
Feature Extraction	16/04/2021	15/05/2021	Mubashar Ali	**70%**										
Classification	16/05/2021	01/06/2021	M. Tanveer	**80%**										
Combine and Integrate parts	02/06/2021	21/06/2021	Mubashar Ali	**90%**										
Review Final App Working	21/06/2021	30/06/2021	Parallel	**95%**										
Project Submission	01/07/2021	31/07/2021	Parallel	**100%**										

Figure 4.1 *Project Timeline Allocation 1*

4.4 Key Milestones

Key Milestones			
Sr. No	Time Spent	Milestones	Deliverables
1.	1 Months (September 18,2020- October 18,2020)	Analysis & Requirements	Research Proposal
2.	5 Days (October 19,2020 –October 24, 2020)	Planning, scheduling	Report
3.	1 Month (October 25, 2020 -November 25, 2020)	Modeling & design	Report
4.	6 & half Months (December 20, 2020-June 25, 2021)	Coding & testing	Software System
5.	7 Days (June 26, 2021-July 3, 2021)	Deployment	Software System

Table 4.1 Key Milestones

4.5 Roles and Responsibilities

Project development team is comprising of two members. To achieve a goal, documentation and development is equally dispersed among them and each member work on parallel to avoid wastage of time.

All the members are hardworking and divide the tasks to avoid the time wastage of time. The major aim behind is to deploy the product within time slot. Like one-member works on Machine learning part and other works on model deployment and integration. Day to day meeting also helps to discuss the challenges and the progress of the project.

Teamwork is also an important task in the field of Computer Science so we also understand and learn how to deal with the team. With the collaboration of the team it is very easy to deploy effective and useful product. Throughout the project the team collaboration was good and both members are cooperative and helpful.

4.6 Work break Down Structure Diagram

As it is mentioned earlier that we use agile technique for the development of our system. So we break down the work into different modules each member performs his task for the relevant module here is the diagram for WBS.

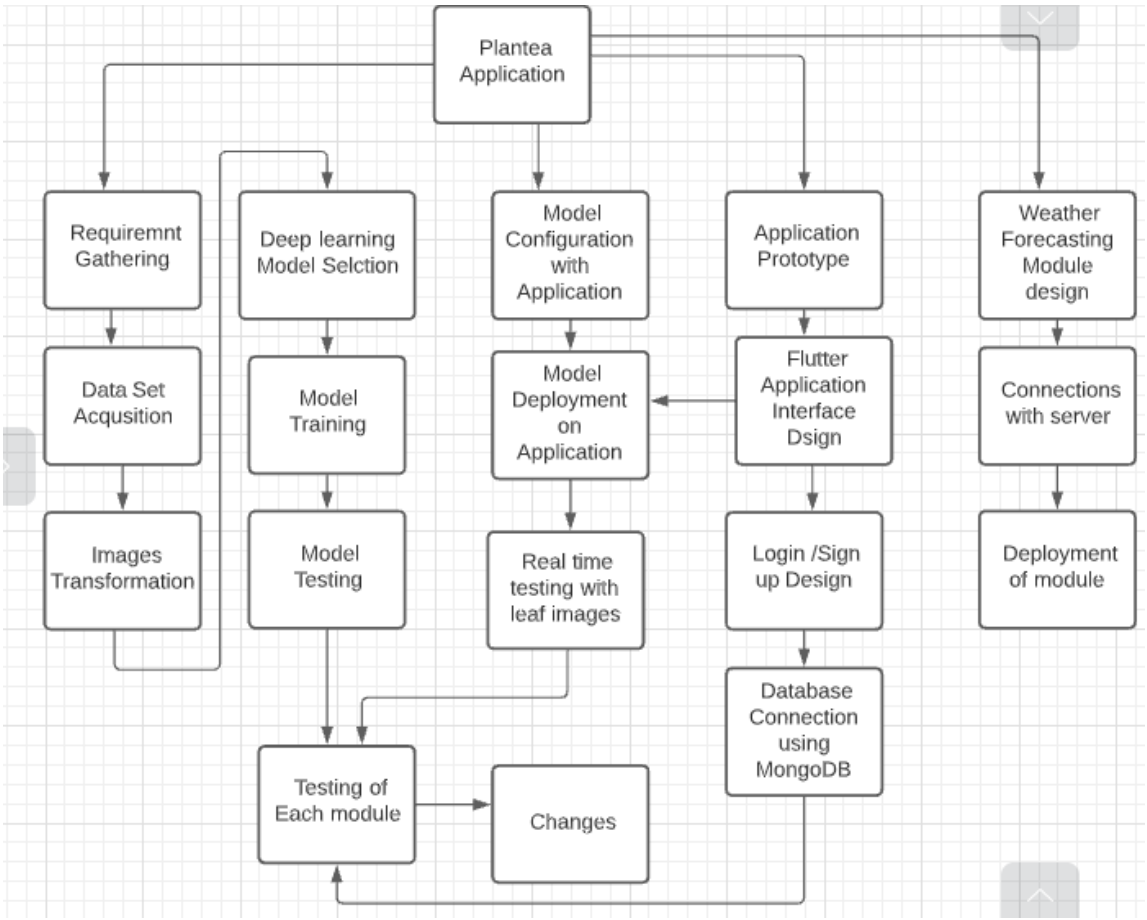


Figure 4.2 WBS Diagram

4.7 Data Base Design Diagram:

Data Base Design diagram represents how different entities interact with each other. Like how the data of user store in the database and which functions user can perform. Like user can save his recent searches but admin should not have access to these pictures in terms of privacy. Similarly, only admin can add new users no user should have access to the admin section.

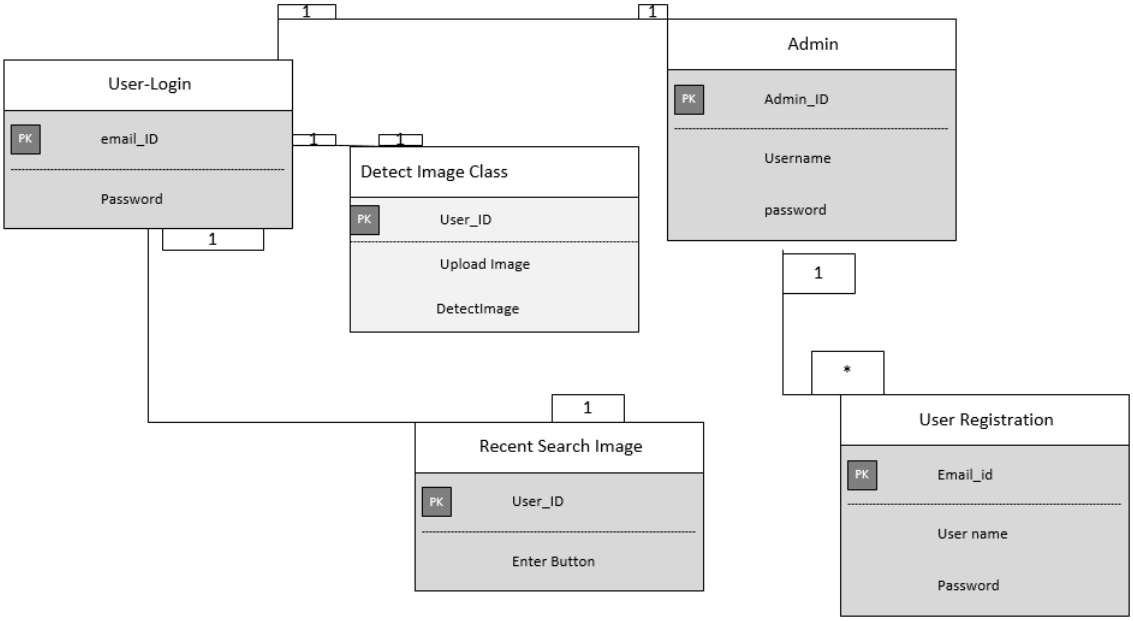


Figure 4.3 Data Base Diagram

CHAPTER # 5

SYSTEM IMPLEMENTATION

5 SYSTEM IMPLEMENTATION

In this chapter, we'll focus on an implementation of "Plantae" application. Where user can detect diseases of plants and get cure.

5.1 Introduction

This section discusses the phase in which the application is developed. Work to be done in the phase is much straightforward and stepwise due to the planning and designing that went into the project before this phase. The design requirements and specifications will be implemented in an executable program. Because designing has been completed, developers now have an idea of how the application should be implemented and what should be its look and feel. Now all that remains is for the developers to put together all the components and designs, so that the project can be easily implemented and understood.

5.2 Training

Object detection generally is a term used to label computer vision techniques for locating objects and labeling them. Object detection techniques can be applied both to static images and moving images.

Open-source libraries such as OpenCV's DNN library and TensorFlow Object detection API offers easy-to-use, opensource frameworks where pre-trained models for object detection reach high accuracy in detecting various objects from humans to tv monitors. However, the image taken from camera or image selection from mobile storage will be of different sizes. That's why, we have to resize the image according to the size of images that we will use for training of our model. This made it important that we train the model to work with specific data. For this project, we choose Tensor Flow object detection API due to its popularity in object detection sand for its easy- to-approach.

We are using 54000 images for the training of model. We are using 70% images for training purposes and 30% for testing purpose. There is total 10 plants including fruits and vegetables which are used for detection of these diseases and there is total 38 diseases which we can detect using this mobile application.

```
apple apple scab
apple black rot
apple cedar apple rust
apple healthy
blueberry healthy
cherry including sour powdery mildew
cherry including sour healthy
corn maize cercospora leaf spot gray leaf spot
corn maize common rust
corn maize northern leaf blight
corn maize healthy
grape black rot
grape esca black measles
grape leaf blight isariopsis leaf spot
grape healthy
orange haunglongbing citrus greening
peach bacterial spot
peach healthy
pepper bell bacterial spot
pepper bell healthy
potato early blight
potato late blight
potato healthy
raspberry healthy
soybean healthy
squash powdery mildew
strawberry leaf scorch
strawberry healthy
tomato bacterial spot
tomato early blight
tomato late blight
tomato leaf mold
tomato septoria leaf spot
tomato spider mites two spotted spider mite
tomato target spot
tomato tomato yellow leaf curl virus
tomato tomato mosaic virus
tomato healthy
```

In the project, TensorFlow object detection API is used and tested for detection of plant diseases. The model used is inception_v3.

```
[ ] module_selection = ("inception_v3", 299, 2048) #@param ["(\"mobilenet_v2\", 224, 1280)",
    handle_base, pixels, FV_SIZE = module_selection
    MODULE_HANDLE ="https://tfhub.dev/google/tf2-preview/{}/feature_vector/2".format(handle_
    IMAGE_SIZE = (pixels, pixels)
    BATCH_SIZE = 64 #@param {type:"integer"}
```

module_selection: ("inception_v3", 299, 2048)

BATCH_SIZE: 64

Figure 1.1 Inception_V3 code

We already know that

1. For working with given data, a pre-trained model needs to be fine-tuned.
2. After fine tuning a per-trained model can and will work efficiently with given data.
3. Fine-tuned data can have some problems with camera images as it cannot provide the equal size image of a plant.
4. Using a training data taken at different times will improve the results with new images.

1.2.1 R-CNN functions

Originally **R-CNN** method worked by running a neural net classifier on samples cropped from images using externally computed box proposals. This approach is very computationally expensive due to many crops. **Fast RCNN** reduced the computation by doing the feature extraction only once to the whole image and using cropping on the lower layer i.e. feature extraction is done only once to the whole image and samples are cropped with externally computed box proposals. **Faster-RCNN** goes beyond and used extracted features to create class-agnostic box proposals i.e. no externally computed box proposals. R-FCN is like Faster RCNN, but the feature cropping is done on a separate layer to increase efficiency.

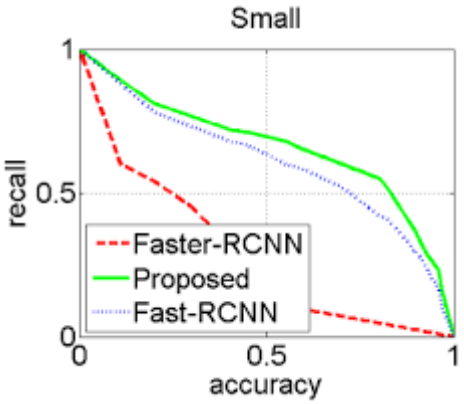


Figure 5.2 Faster RCNN model

5.2.2 Activation Function

Activation Function control how well the network model learns the training dataset. There are two activation functions that we are using for training of our model.

5.2.2.1 ReLu

The first one is "**ReLu**" Rectified Linear Unit. The operation of ReLu is closer to biological neurons work. ReLu is non-linear and the advantage of ReLu is not having backpropagation errors unlike the sigmoid function. For large neural networks, the speed of building models based on ReLu is very fast.

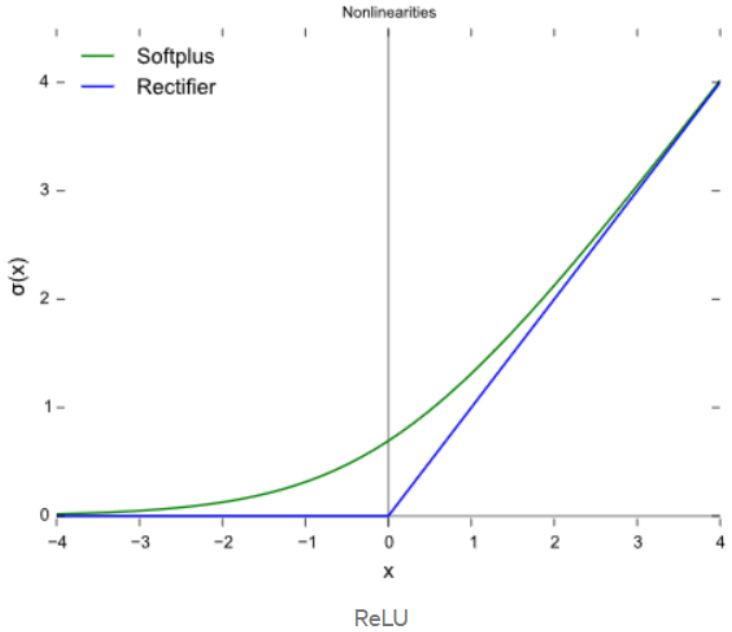


Figure 5.3 ReLu Function Graph

5.2.2.2 Softmax

The second one is "**Softmax**". It is usually used in the final layer of neural network-based classifier. This type of networks are commonly trained under a log loss regime, giving a non-linear variant of multinormal logistic regression. The output of Softmax is a probability distribution.

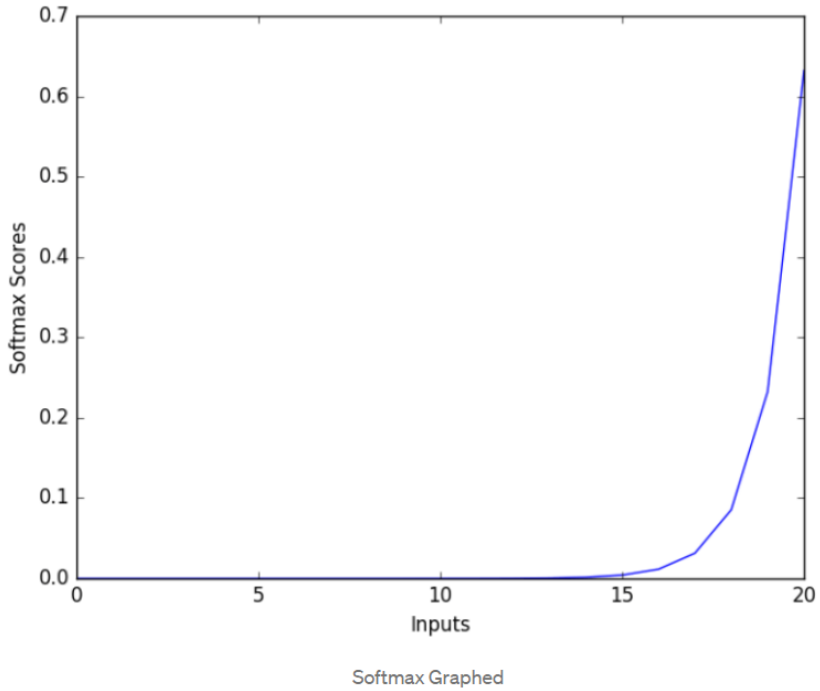


Figure 5.4 Softmax Graph

5.2.3 Training of Data Set

The object detection API is a framework built on top of TensorFlow that helps to make it easy to 'develop, train and deploy object detection models. To make this possible, the TensorFlow Object Detection API makes available to the user multiple pre-trained object detection models. In this work, we used Faster RCNN model with Inception v3. We have our own dataset of 54000 images.

The pre-trained F-RCNN model was fine-tuned for our dataset manually. A provided faster_rcnn_inception_v2_dataset.config file was used as a basis for the model configuration. The provided checkpoint for faster_rcnn_inception_v2_dataset was used as a starting point for the fine-tuning process. The training was stopped after 20 epochs. During the process the loss function eventually evened out to a very small value after that many steps.

The training took over 72 hours on server with CPU with 8 cores. The total loss was greatly decreased over time from the pre-trained model.

```
[ ]  EPOCHS=20 #@param {type:"integer"}
     STEPS_EPOCHS = train_generator.samples//train_generator.batch_size
     VALID_STEPS=validation_generator.samples//validation_generator.batch_size
     history = model.fit_generator(
               train_generator,
               steps_per_epoch=STEPS_EPOCHS,
               epochs=EPOCHS,
               validation_data=validation_generator,
               validation_steps=VALID_STEPS)
```

EPOCHS: 20

Figure 5.5 Training of Dataset

5.2.4 Model Accuracy

Model accuracy means the number of classifications a model correctly predicts divided by the total number of predictions made. It's a way of assessing the performance of a model, but certainly not the only way.

After performing 20 epochs, our model gives 89% accuracy and 29% percent loss values. The results for testing are very good.

```
Epoch 18/20
286/286 [==============================] - 380s 1s/step - batch: 142.5000 - size: 63.9196 - loss: 0.3143 - accuracy: 0.8900
Epoch 19/20
286/286 [==============================] - 380s 1s/step - batch: 142.5000 - size: 63.9196 - loss: 0.2920 - accuracy: 0.8983
Epoch 20/20
286/286 [==============================] - 379s 1s/step - batch: 142.5000 - size: 63.9196 - loss: 0.2902 - accuracy: 0.8979
```

Figure 5.6 Accuracy Calculation

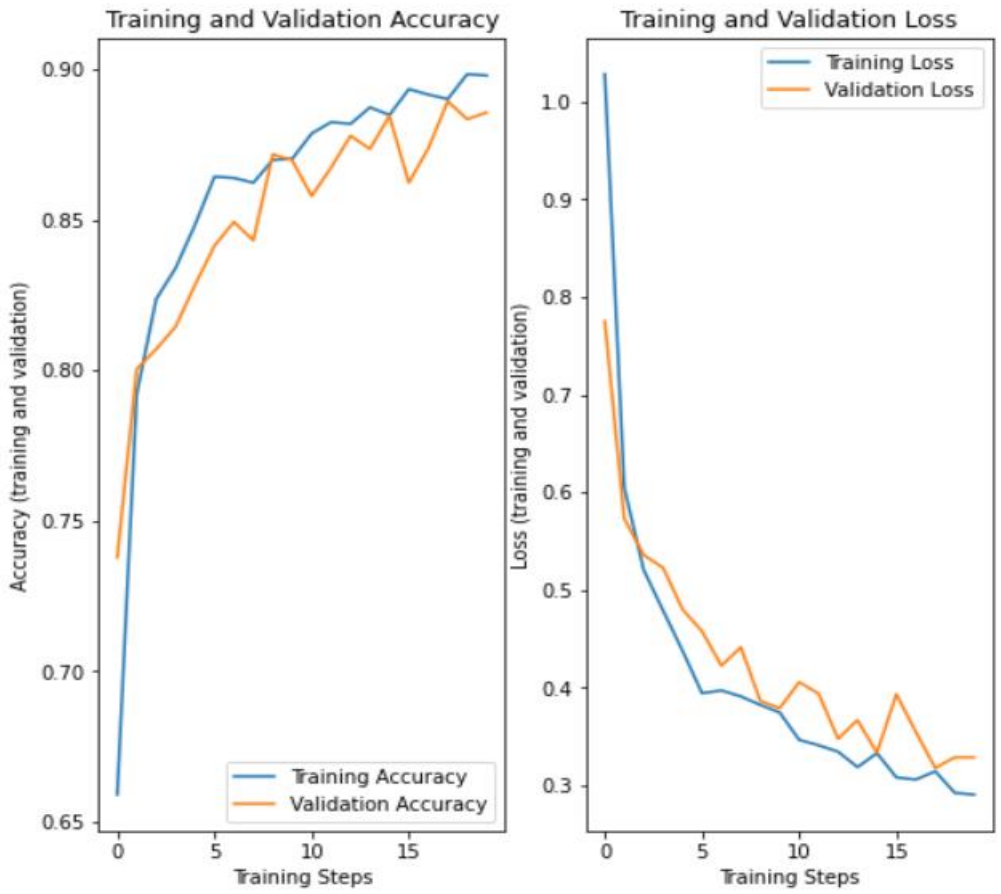
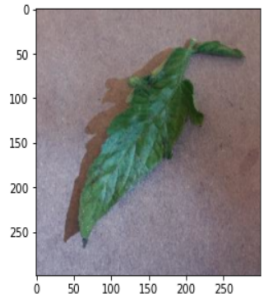


Figure 5.7 Training Validation and Loss

5.2.5 Model Testing

Here we test the model by using 30% validation images which we have separated from original data. In model testing, we perform testing validations test to check weather our model performs accurate detection or not at run time. It will also provide us an idea that how our final product responses on different images. We perform some tests on different images at runtime and it provide us good results as shown in figures

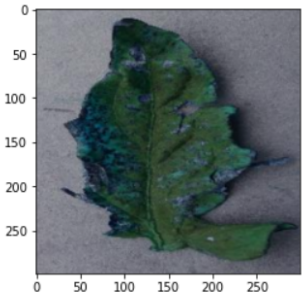
```
SOURCE: class: Tomato___Leaf_Mold, file: Tomato___Leaf_Mold/fe0bcd7f-a7de-4155-95a5-79043190eef7___Crnl_L.Mold 7030_180deg.JPG
/usr/local/lib/python3.7/dist-packages/tensorflow/python/keras/engine/training.py:2325: UserWarning: `Model.state_updates` will be removed in a future version.
  warnings.warn('`Model.state_updates` will be removed in a future version. '
PREDICTED: class: Tomato___Leaf_Mold, confidence: 0.999866
```




```
<Figure size 432x288 with 0 Axes>
```

Figure 5.8 Bacterial Spot Detection

```
SOURCE: class: Tomato___Bacterial_spot, file: Tomato___Bacterial_spot/35f5191b-02dc-4358-8b35-808f2e1dc580___GCREC_Bact.Sp 5590.JPG
PREDICTED: class: Tomato___Bacterial_spot, confidence: 0.743969
```




```
<Figure size 432x288 with 0 Axes>
```

Figure 5.9 Leaf Mold Detection

```
SOURCE: class: Tomato___Leaf_Mold, file: Tomato___Leaf_Mold/ed52b0a7-f569-4fdb-b04c-a624d8793092___Crnl_L.Mold 8880.JPG
PREDICTED: class: Tomato___Leaf_Mold, confidence: 0.977153
```

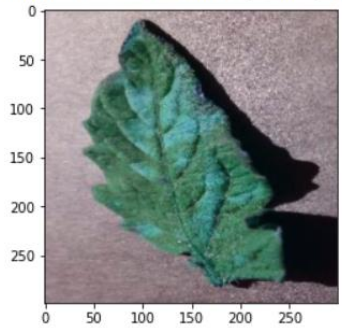
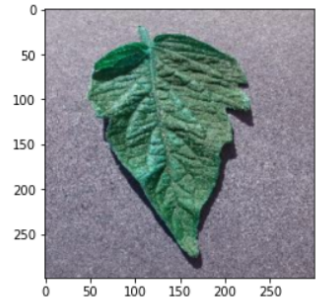


Figure 5.10 Leaf Mold Detection

```
SOURCE: class: Tomato___Spider_mites Two-spotted_spider_mite, file: Tomato___Spider_mites Two-spotted_spider_mite/3a54bf13-626b-4c
PREDICTED: class: Tomato___Tomato_mosaic_virus, confidence: 0.830973
```




```
<Figure size 432x288 with 0 Axes>
```

Figure 5.11 *Mosaic Virus Detection*

5.2.6 Model Conversion

Then at the end we get our model in the form of plant_disease_model.m5 file. We save this model in our directory. We convert this model to plant_disease_model. tflite file because. tflite we are using TensorFlow for training the model and. tflite is also easy to integrate in flutter. It works perfectly.

```
!mkdir "tflite_models"
TFLITE_MODEL = "tflite_models/plant_disease_model.tflite"


# Get the concrete function from the Keras model.
run_model = tf.function(lambda x : reloaded(x))

# Save the concrete function.
concrete_func = run_model.get_concrete_function(
    tf.TensorSpec(model.inputs[0].shape, model.inputs[0].dtype)
)

# Convert the model to standard TensorFlow Lite model
converter = tf.lite.TFLiteConverter.from_concrete_functions([concrete_func])
converted_tflite_model = converter.convert()
open(TFLITE_MODEL, "wb").write(converted_tflite_model)
```

Figure 5.12 *Tflite Conversion*

5.2.7 Model Integration into App

```
void initState() {
  super.initState();
  _busy = true;

  loadModel().then((val) {
    setState(() {
      _busy = false;
    });
  });
}

Future loadModel() async {
  try {
    await Tflite.loadModel(
      model: "assets/plant_disease_model.tflite",
      labels: "assets/plant_labels.txt",
    );
  } on PlatformException {
    print('Failed to load model.');
  }
}

Future recognizeImage(File image) async {
  var recognitions = await Tflite.runModelOnImage(
    path: image.path,
    numResults: 6,
    threshold: 0.05,
    imageMean: 127.5,
    imageStd: 127.5,
  );
  setState(() {
    _busy = false;
    _recognitions = recognitions;
  });
```

Figure 5.13 *Model Integration in Application*

5.3 Mobile Application

5.3.1 Introduction

This section provides an overview of the working of the mobile app of disease detection project. Some the problems encountered during the development and deployment will also be discussed in this section.

5.3.2 Description

The plantae app aims to provide a platform for the users where they can log in, upload a picture or capture a picture of the plants that they need to get diagnosed. The app detects if there are any infected plants from the footage that is provided by the user. The app also tells weather details hourly and weekly. The application is a Mobile app that has been implemented using the following

- Python
- Flutter
- Android Studio
- SQLite
- NodeJS
- MongoDB
- Visual Studio Code
- TensorFlow

5.3.3 Features of Mobile Application

Followings are the main features of our mobile application.

Figure 5.14 Splash Screen

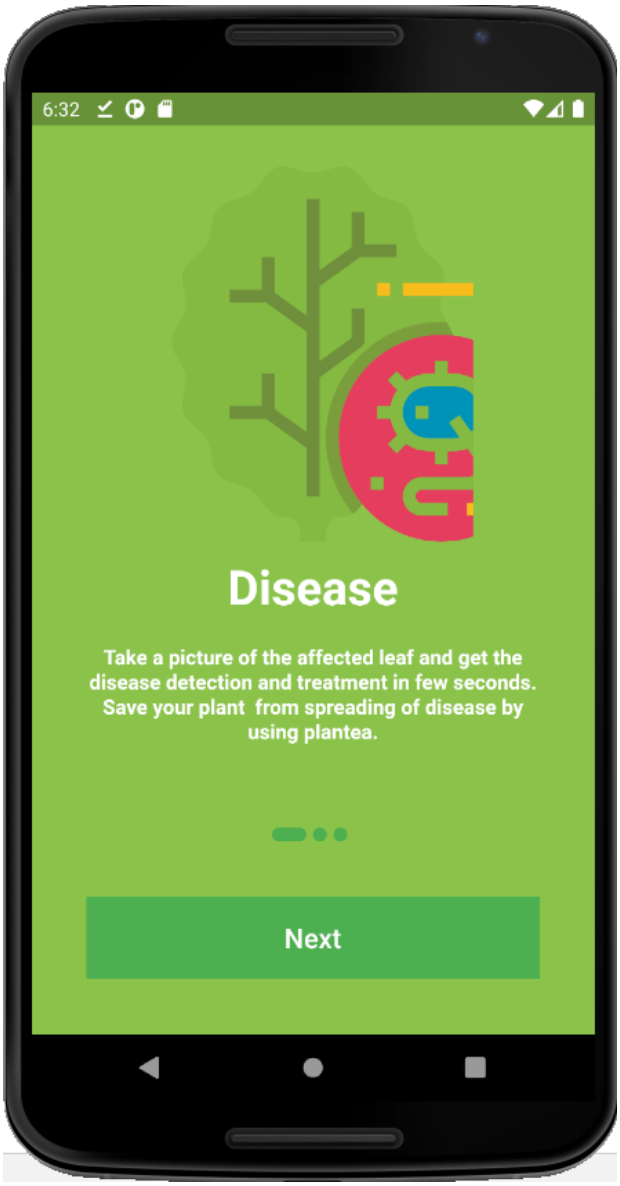


Figure 5.15 Onboarding Screen 1

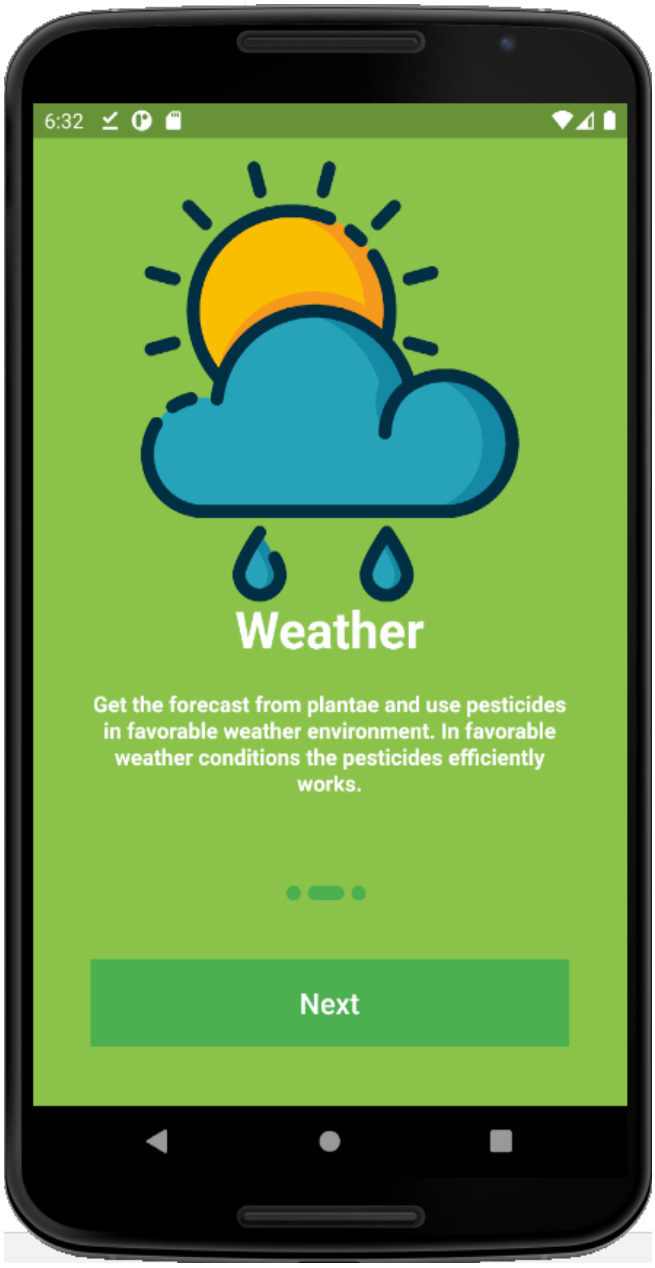


Figure 5.16 Onboarding Screen 2

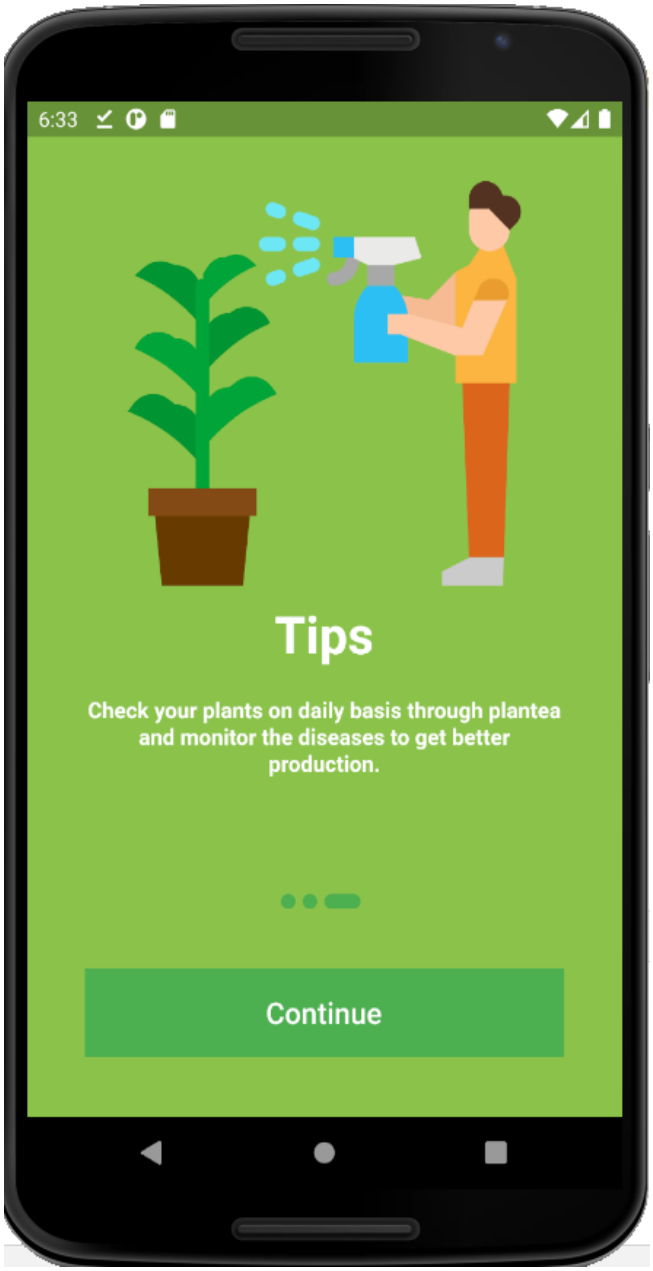


Figure 5.17 Onboarding Screen 3

5.3.3.1 Registration

The user inputs their information, we verify that information with conditions on input and insert into database and return message with a success notification. If the information is not valid, we'll remain on registration page and show their EROR to user. The inputs are given below:

- Name-letters only which will be submitted
- Email-Valid Email format
- Password

5.3.3.2 Login

When the user initially registers, we would log them in automatically, but the process of "logging in" is simply just verifying that the email and password the user is providing matches up with one of the records that we have in our database table for users.

- Email
- Password -also password will be checked with corresponding email.

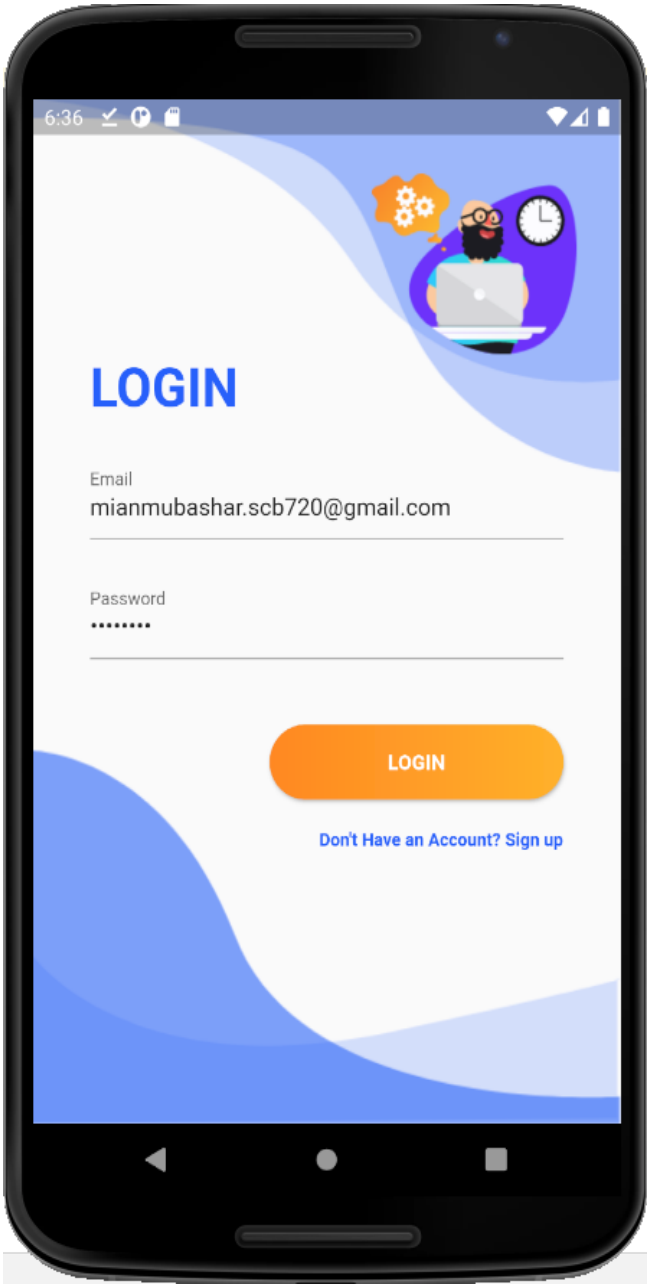


Figure 5.18 Login Screen

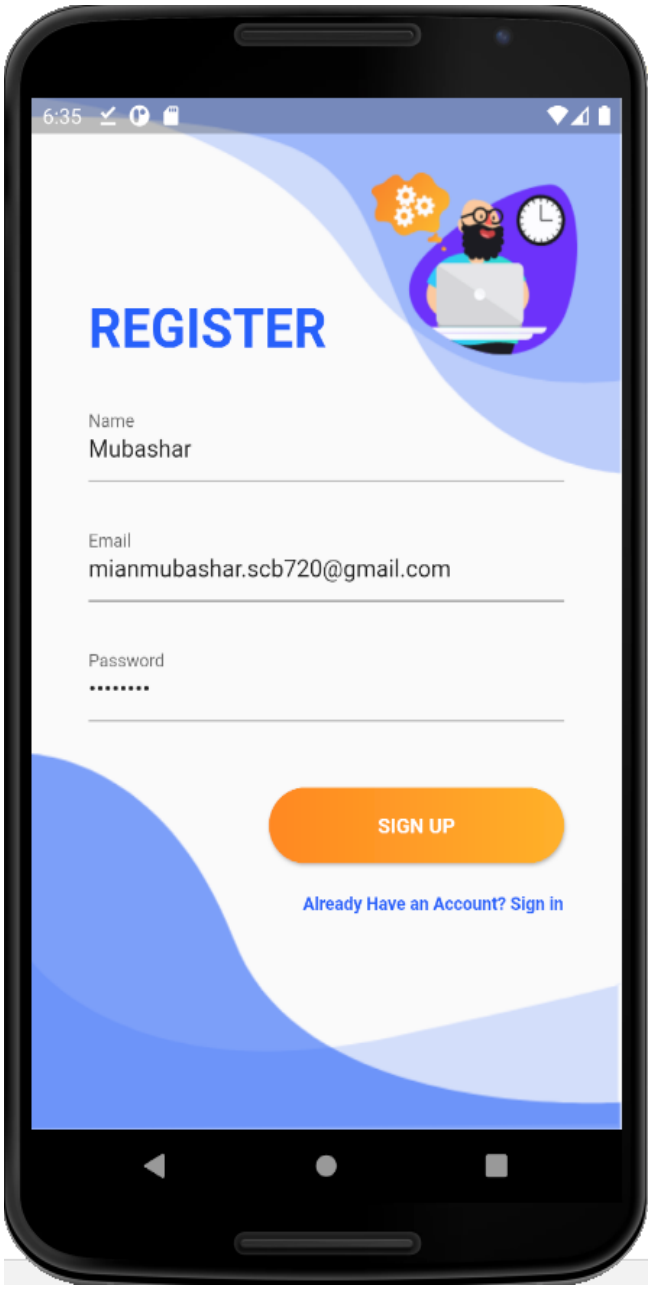


Figure 5.19 Onboarding Screen 1

5.3.3.4 How to keep track of them once they have logged in?

It's using session! We can create a session variable that holds the user's id. From our study in Database Design, we know that if we have the id of any table, we can gather the rest of the information that is associated with that id. Storing a single session variable with the user's id is all we need to access all the information associated with that user.

Once we have already identified the places on our site that we wish to be dynamic for users that are logged in, then we just need to check to see if that session variable has been set and display the content accordingly.

What's happening!

1. A basic login and registration form
2. Redirect user to a home page on successful login and register
3. Display error messages if either login or registration validations fail.

```
void navigateUser() async{
  SharedPreferences prefs = await SharedPreferences.getInstance();
  var status = prefs.getBool('isLoggedIn') ?? false;
  print(status);
  if (status) {
    Navigation.pushReplacement(context, "/Home");
  } else {
    Navigation.pushReplacement(context, "/Login");
  }
}
```

Figure 5.20 Login Authentication 1

5.3.3.5 Static Disease Detection

The main functionality of the tool is the detection of a disease in an image of a plant. The app provides the user with an interface where user can upload a picture and the app detects disease in plant leaf. Complete process of detection is done on the server. Once the detection script has run, the user is directed to a new page that displays the details about detected diseases in the plants and its remedy. The image will be stored in previous search history from where a user can see which images he has searched before.

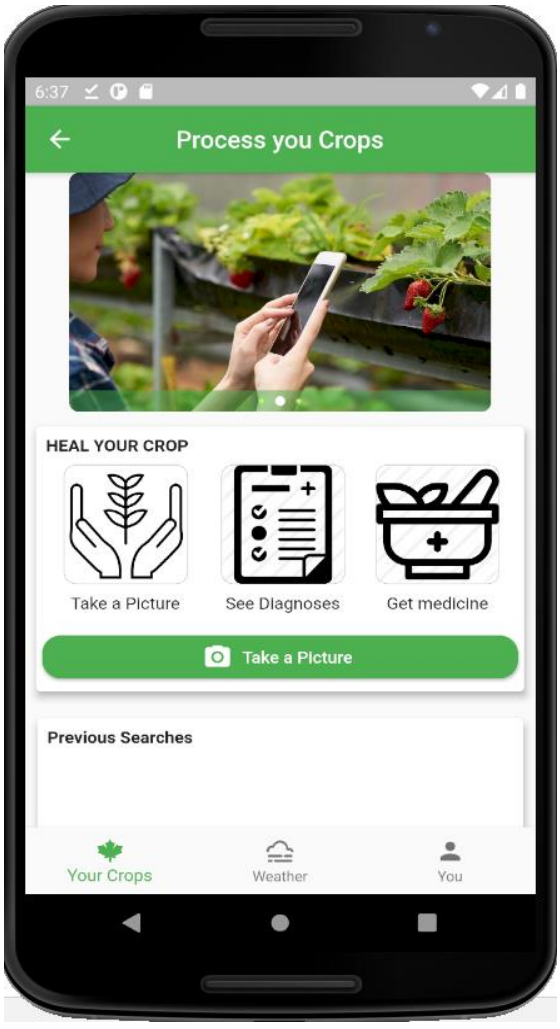


Figure 5.21 Home Screens

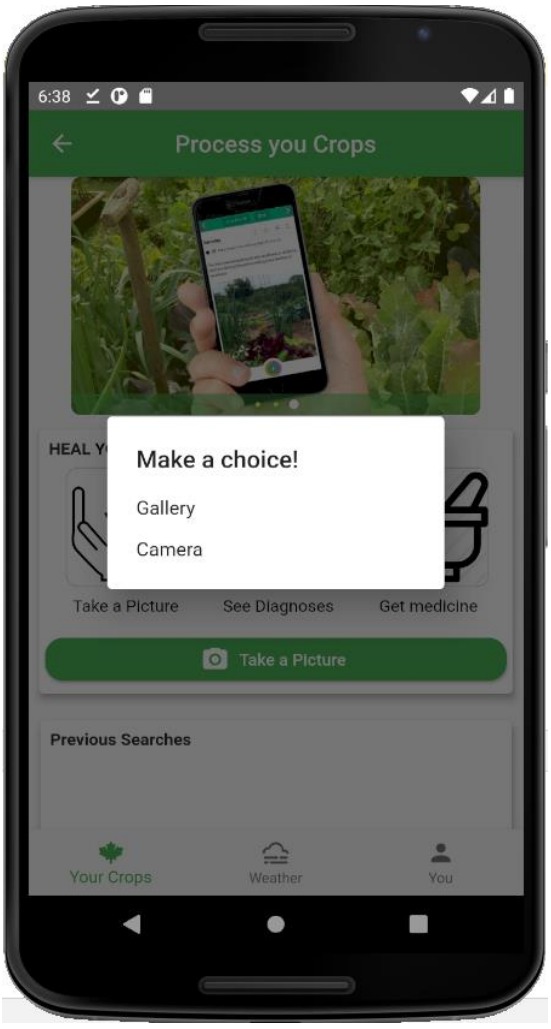


Figure 5.22 Choice screen

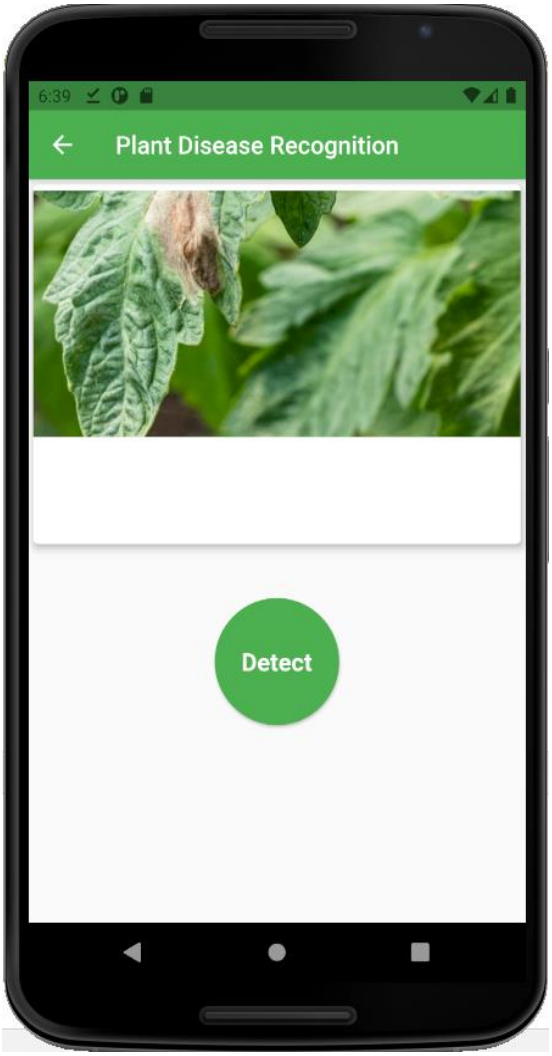


Figure 5.22 Detection Screen

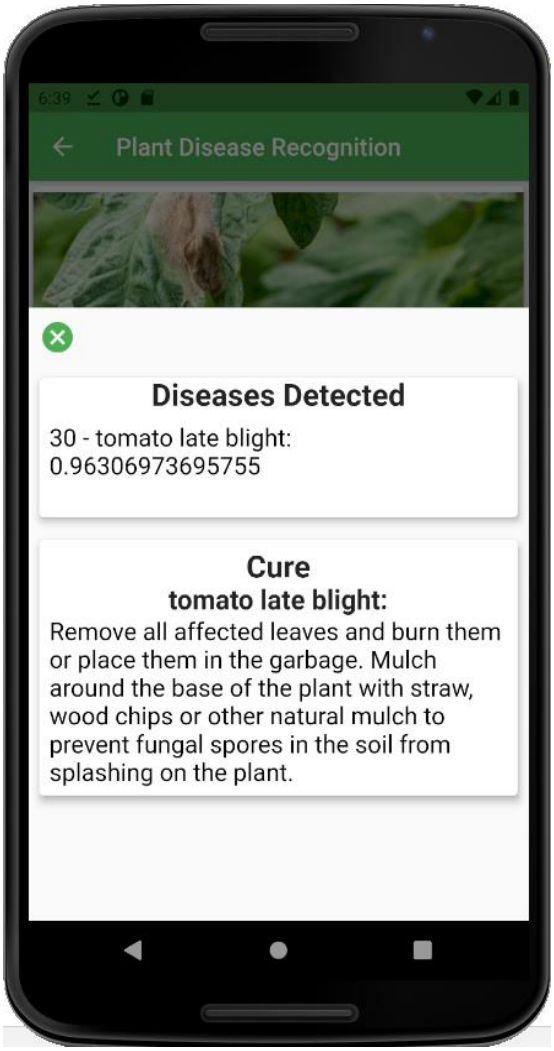


Figure 5.23 Detect and Cure Screen

5.3.3.6 History

The app provides a history tab to the user where they can see which images he has searched before in the previous detections.

5.3.5 Setting

Here user can read information and app use from about me button. User can also contact with the developer or owner of the app by going into the contact us card.

Figure 5.24 History Screen

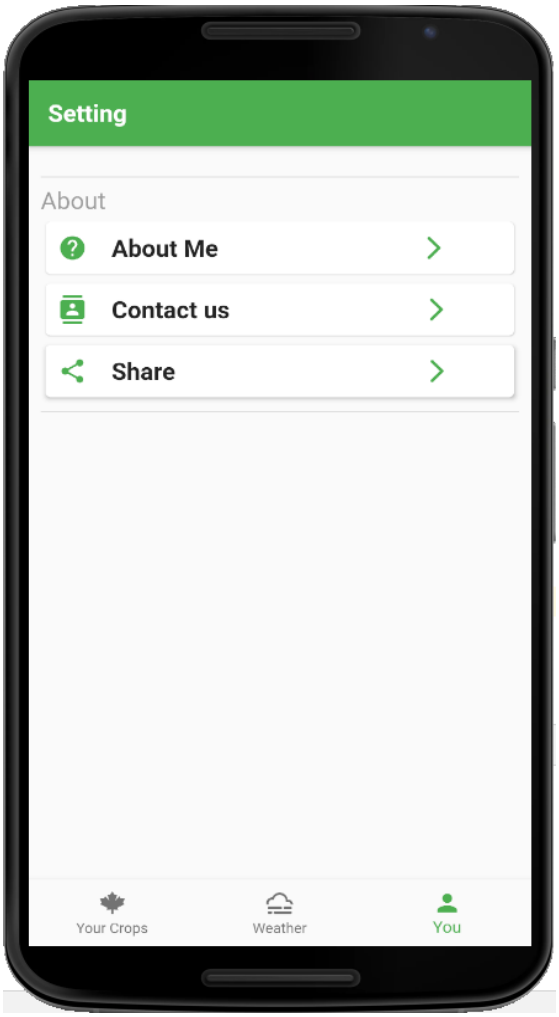


Figure 5.25 Settings Screen

5.3.3.8 Validation

The app has authentication-based access. It allows only the registered users to access all the features. If the user has no account for the app, he can sign up for it in the signup section provided.

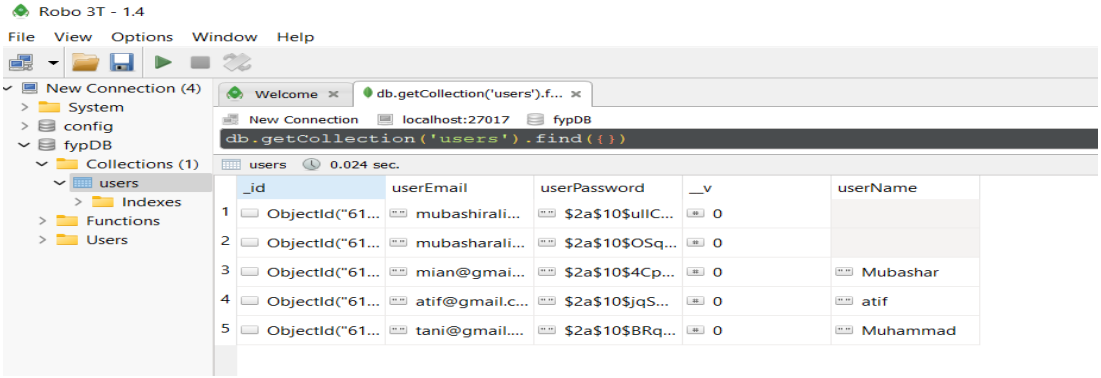


Figure 5.26 *SQL Lite Data Base*

5.3.3.8.1 Login Authentication

```
static userLogin(dynamic context,dynamic userEmail,dynamic userPassword) async {
//  provider.setLoadingState(true);
  var res = await http.post(Uri.parse(ApiUtils.loginUrl), body: {
    "userEmail": userEmail,
    "userPassword": userPassword,
  });
  if (res.statusCode == 200) {
    print("success");
    //provider.setLoadingState(false);
    Navigator.pushReplacement(
        context, MaterialPageRoute(builder: (_) => Home1()));
  } else {
    var data = jsonDecode(res.body);
    var message = data['msg'];
    SnakBar.snackBar(context, message);
    //provider.setLoadingState(false);
  }
}
```

Figure 5.27 Login Authentication Code

5.3.3.8.2 Signup Process

```
static userRegistration(
    //AuthProvider provider,
    dynamic context,dynamic userEmail,dynamic userName,dynamic userPassword) async {
//  provider.setLoadingState(true);
  var res = await http.post(Uri.parse(ApiUtils.registerUrl), body: {
    "userName":userName,
    "userEmail": userEmail,
    "userPassword": userPassword
  });
  if (res.statusCode == 200) {
   // provider.setLoadingState(false);
    print("success");
    Navigator.pushReplacement(
        context, MaterialPageRoute(builder: (_) => Home1()));
  } else {
    print("erro");
    var data = jsonDecode(res.body.toString());
    //print(data);
    var message = data['msg'];
    SnakBar.snackBar(context, message);
   // provider.setLoadingState(false);
  }
}
```

Figure 5.28 Signup Code

5.3.4 Weather Forecasting

In weather forecasting, it shows the detail of weather of the city in which user is using the app. The app asks for location access, if user give access than app fetch the details of the app and if user not give access of location, then its app will not show weather details. We can also search the weather details of interested city. The weather module also shows the detail of weather condition hourly and weekly.

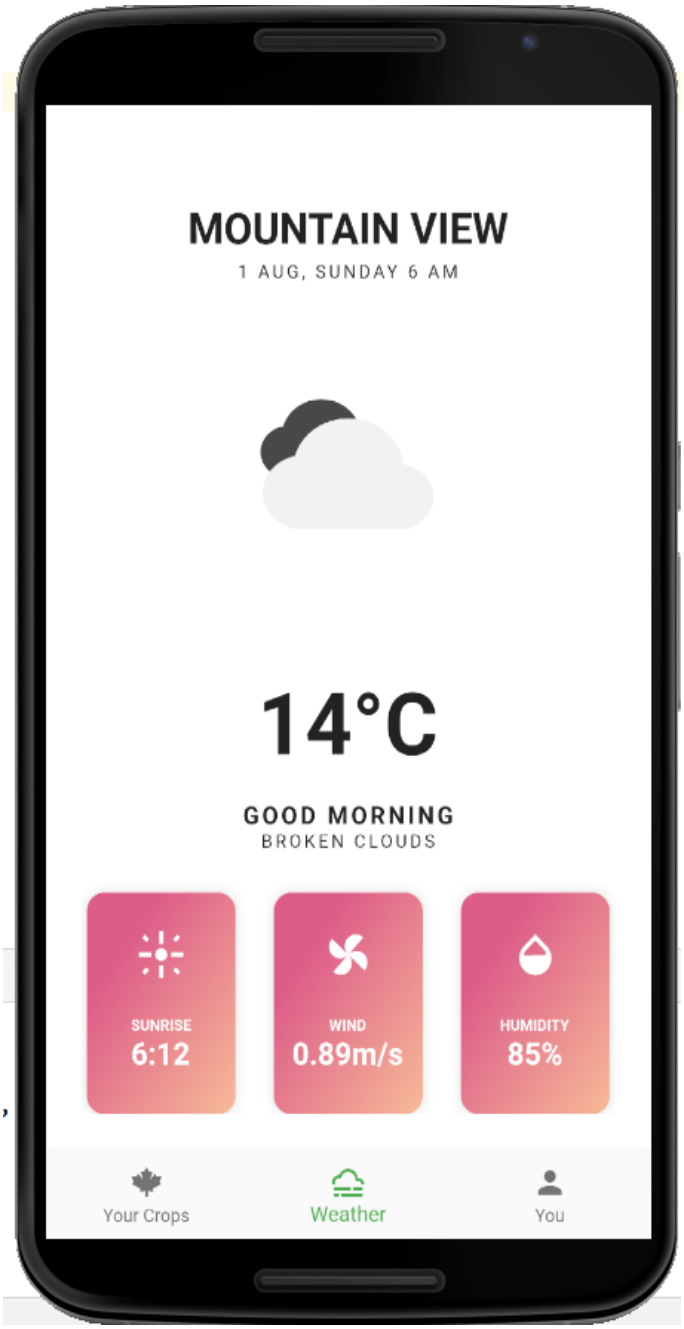


Figure 5.29 Weather Forecast Screens

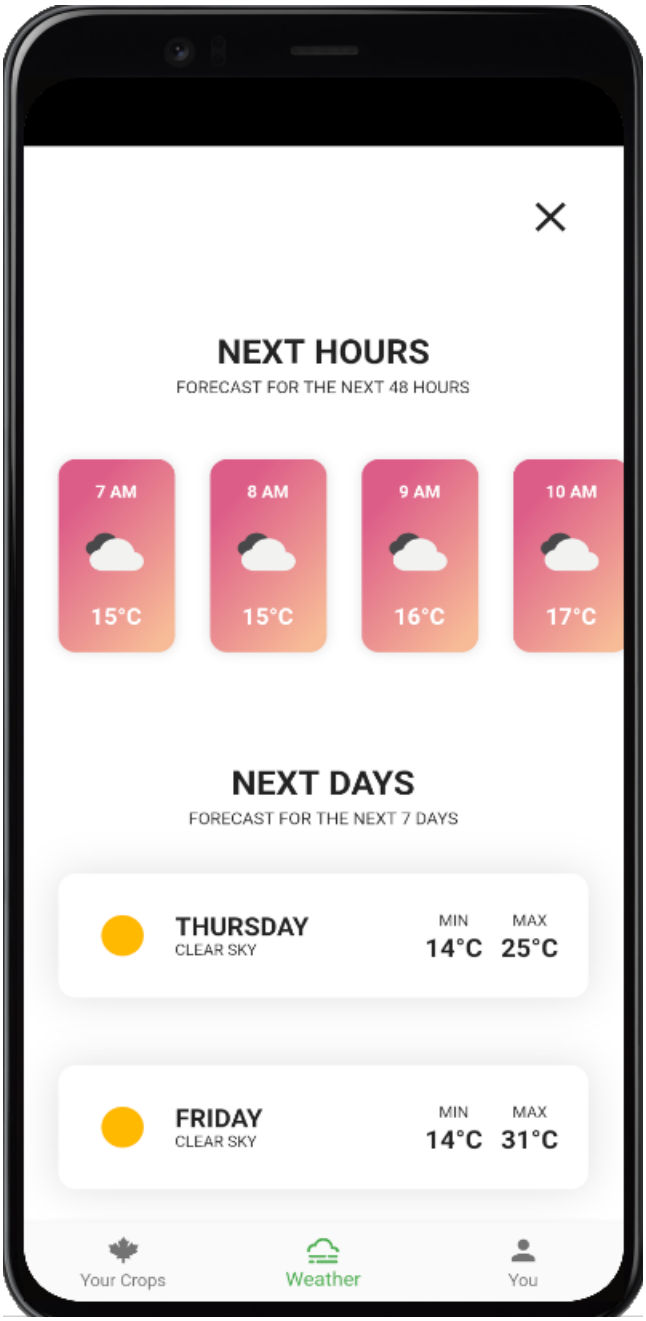


Figure 5.30 Weather Forecast Screens

5.3.6 How does it work?

The mobile app is developed using flutter language on android studio. For database, we are using SQLite for local storage and NodeJS and MongoDB for main storage. When user login, the app authenticates weather the user is registered or not. If the user is not registered, then the app will show ERROR. New users can register themselves by going to signup page. When user register, his data comes in the storage which admin can access and see.

The detection of diseases is done by running the images against the trained detection models that have been trained on the dataset containing images from different sources of different plants that are healthy as well as infected with diseases.

CHAPTER # 6

SYSTEM TESTING

6 SYSTEM TESTING

In this chapter, we will confer the testing phase of developed application "**Plants Disease Detection Application"** in different manner to know that how much efficient and effective application is.

6.1 Introduction

System testing is a procedure of executing an application or program with the expectation of discovering mistakes and whether the application is satisfying client needs. It can also be described as the capacity of a program in meeting the required or wanted outcomes. Generally, in the field of software engineering, a separate phase is carried out which is called testing phase it is done after the implementation phase has been completed. This approach has a benefit that it is difficult for an individual to look for his mistakes, but a different perspective and fresh eye can readily find the observable errors much quicker than the individual that has written or read the said material several times.

6.2 Testing Plan

A process of performing as application or program with the intention of finding errors and whether the application is fulfilling user needs.

6.2.1 Unit Testing

The software units in an application are modules and subroutines that are incorporated and integrated to play a particular function. Unit testing first of all focuses around modules, irrespective of one another's functionalities, to find errors. This enables the developers to recognize errors in the code and the logic within each individual module. This testing includes performing practically. All the feature controls are tested to make sure that each performs its action as required.

Commonly used method is White-Box Testing method. Every time a component of the program is changed, it can be run for testing that is the biggest and famous benefit of this testing phase. Issues that are arises during this phase, allowing to be resolved as quickly as

possible. Unit testing is familiar by software developers. It allows them to test their application units before moving them to testers for formal testing.

6.2.2 System Testing

To test the complete “Plants Diseases Detection Application”, system testing has been used. It is beneficial to check whether the application meets its requirements and fulfill Quality Standards.

6.2.3 Integration Testing

Integration testing allows the software developers to integrate all the components of the “Plants Diseases Detection Application” within one program and then test them in a group. Basically, this testing level is used. To catch the defects in the user interface between the function’s modules. It is useful to determine how logically and efficiently all the units’ components are running together. This testing helps to make sure that the application is integrated efficiently and is a functional unit that has a smooth transition of run time working.

6.2.4 User Acceptance Testing

User acceptance of “Plants Diseases Detection Application” is the key factor for the success of our application. The application under consideration has been tested for user compliance by frequently keeping in touch with the application users during developing period and making required changes.

6.3 Test Cases:

6.3.1 Register:

In this project we have tested the following test cases.

Test Case #: 1	Test Case Name: Register User
System: Plants Diseases Detection Application	Subsystem: Register
Designed by: Muhammad Tanveer	Design Date: 27/07/2021
Executed by: Muhammad Tanveer	Execution Date: 27/07/2021
Short Description: Register new user	
Pre-conditions	
User has opened the register page and have not register already with correct input and put wrong input	

Figure 6.1 Test Case Registration

Step	Action	Expected System Response	Pass /Fail	Comment
1	Enter Name, Email, password and Confirm Password	System Required to enter Password, Name, Email, Confirm Password	✓	All values correctly entered
2	Click to Submit	System Main Interface of the	✓	Display/ Home Page
3	**Check post-condition 1**			
4	Repeat steps 1 and put confirm password wrong	System Required to enter, Password, Name, Email, Confirm Password	✓	Confirm Password Wrong
5	Click on Submit	System show error password does not match	✓	Displayed Error
6	**Check post-condition 2**			
7	Repeat step1 and enter wrong email	Wrong Email	✓	Displayed Error
8	**Check post-condition 3**			
9	Repeat step 1 with Enter wrong CNIC	Display Wrong Name	✓	Displayed Error

Table 6.2 Register Test Case Table

Post-conditions

1. Password does not match
2. Display error.
3. Display error.

ID	Input								Expected Result
	Username		Password		Special Character		Length		
	Valid Class	Valid Class	Valid Class	Valid Class	Valid Class	Valid Class	Valid Class	Valid Class	
1	Ali	-	123#ABCd	-	#	-	>7	-	Register Successfully
2	-	Ali	123#ABCd	-	#	-	>7	-	Invalid Username
3	Ali	-	123#ABC	-	#	-	>7	-	Invalid Username
4	Ali	-	123#?ABCd	-	-	#?	>7	-	Invalid Username

Table 6.3 Register Test Cases Valid or Invalid Class

6.3.2Login:

Pre-conditions

User has opened the login page and have not login already with correct input and put wrong input

Test Case 2

Test Case Name: Login

System: Plant Diseases Detection | Subsystem: Login

Designed by: Tanveer | Design Date: 27/07/2021

Executed by: Mubashar | Execution Date: 27/07/2021

Short Description: Login user

Figure 6.4 Test Case for login

Step	Action	Expected System Response	Pass/Fail	Comment
1	Enter Email Password	System Required to enter Email Password	✓	All Values Correctly Entered
2	Click on Submit	System show Main Screen of App	✓	Display Home Screen
3	**Check post-condition 1**			
4	Repeat steps 1 and put wrong password	Wrong input	✓	Password or Email Wrong
5	Click on Submit	System shows error password wrong input	✓	Displayed Error
6	**Check post-condition 2**			
7	Repeat step1 and enter wrong Email	Wrong Email	✓	Displayed Error
8	Click on Submit	System shows error password wrong input	✓	Displayed Error

Table 6.5 Login Test Case Table

6.3.3 Disease Detection:

Test Case #: 3 Test Case Name: Detection

System: Plants Disease Detection Subsystem: Disease Detection

Designed by: Mubashar Design Date: 27/07/2021

Executed by: Tanveer Execution Date: 27/07/2021

Short Description: Detect Disease of Plants

Pre-conditions

User has opened the login page and have not login already with correct input and put wrong input

Step	Action	Expected System Response	Pass/Fail	Comment
1	Upload JPG File	System Required image with jpg, png or jpeg format	✓	Correct file
2	Click on Submit	Image passed	✓	Display Disease
3	**Check post-condition 1**			
4	Repeat steps 1 and put wrong file format	Choose correct file format	✓	Wrong file format
5	Click on Submit	System shows error password wrong input	✓	Displayed Error
5	**Check post-condition 2**			
6	Repeat step1 and put compress or blur image	Lower quality image	✓	Displayed Error
8	Click on Submit	System shows error password wrong input	✓	Displayed Error

Table 6.6 Disease Detection Using valid and Invalid class Table

Post-conditions

1. Wrong file format
2. Lower Quality Image

6.3.4 Contact Us:

Test Case #: 4	Test Case Name: Contact Us
Subsystem: Contact Us	By: Mubashar Ali
Design Date: 27/07/2021	
Executed by: Tanveer	Execution Date: 27/07/2021

Pre-conditions

User have opened contact us page to contact with admin by submitting his/her issue

Step	Action	Expected System Response	Pass/Fail	Comment
1	Fill Form Correctly	Fill form Correctly	✓	Correctly
2	Click on Submit	Form Submitted Notification	✓	Successfully submitted
3	**Check post-condition 1**			
5	**Check post-condition 2**			
6	Repeat step1 and titled field required don't put title	Lower quality image	✓	Displayed Error

Table 6.7 Contact Us Test Case Table

Post-conditions

1. Email Required
2. Title Field Required

ID	Input								Expected Result
	Email		Subject		Phone Number		Length		
	Valid Class	Valid Class	Valid Class	Valid Class	Valid Class	Valid Class	Valid Class	Valid Class	
1	Mian1@gmail.com	-	123#ABCd	-	+92315-0001320	-	13	-	Submitted Successfully
2	-	Mian1@gmail.com	123#ABCd	-	+92315-0001320	-	13	-	Invalid email
3	Mian1@gmail.com	-	123#ABC	-	+92315-0001320	-	13	-	Invalid number
4	Mian1@gmail.com	-	123#?ABCd	-	-	+92315-0001320	13	-	Invalid number
5	-	Mian1@gmail.com	123#?ABCd		-	+92315-0001320	13	>13 Or <13	Invalid email and number

Table 6.8 Contact Us Test Cases Valid or Invalid Class Table

Email	Subject	Phone Number	Length
mianmubashar720@gmail.com	123#AB Cd	+923150001320	>13 Or <13
mianmubashar720@gmail.com	123#AB Cd	+923150001320	>13 Or <13
mianmubashar720@gmail.com	123#AB Cd	+923150001320	>13 Or <13
mianmubashar720@gmail.com	123#AB Cd	+923150001320	>13 Or <13
mianmubashar720@gmail.com	123#AB Cd	+923150001320	>13 Or <13

Table 6.9 Contact Us Test Cases Valid or Invalid Class Example Table

6.4 Objectives of Testing

Test Cases	Objectives
1	To make sure that application is easy to understand and is in working order.
2	To make sure that the user can work on application with ease.
3	To make sure that user can view data correctly.
4	To make sure that Detection of Disease is done on run time on Android App.
5	To make sure that all the detection are according to disease.
6	To make sure that server is picking the video and showing detection
7	To make sure that all modules are working properly.
8	To make sure that the application run at cross android versions successfully.
9	To make sure that android camera able to upload image

Table 6.10 Objectives -Test Case

6.5 Testing Results

Criteria	Test Status	REMARKS
All the graphical user interface options display successfully.	Test successful	OK
Disease Detection is working on run time.	Test successful	OK
Detections are working properly according to their targeted image	Test successful	OK
Label showing exact data	Test successful	OK
Android camera is streaming video to server	Test successful	OK

Table 6.11 Testing Result

CHAPTER #7

CONCLUSION
&
FUTURE WORK

7 CONCLUSIO & FUTURE WORK

In this chapter, we will discuss the results and discussions of this framework "**Plant Diseases Detection**" with conclude remarks and will also discuss related future work of this application.

7.1 Conclusion

"Plant **Diseases Detection**" is developed for multiple DJI versions. Application and product are specially designed for farmers, agricultural industries, agricultural consultants and Government Agencies & Departments. The main objective of this application is to provide the facility of quick disease detection with modern techniques with the development of a practical, reliable and inexpensive real time application that can be used in fields. It is a market-oriented product for Plant Disease Detection, a smartphone app compatible with both smartphone camera and drone camera. User just needs to install android version of it once and then use it one clicks away. It will improve the performance agricultural crops.

7.2 Future Work

In next our first preference is to enhance this application by providing more new features that are as follows:

1. Synchronization of Android and Web application.
2. Integration of Raspberry PI.
3. Launching across customized drones (no DJI drone involved).

8 REFERENCES

1. Rampasek, L. and A. Goldenberg, *TensorFlow: Biology's Gateway to Deep Learning?* Cell Syst, 2016. **2**(1): p. 12-4.
2. Alzantot, M., et al., *RSTensorFlow: GPU Enabled TensorFlow for Deep Learning on Commodity Android Devices.* MobiSys, 2017. **2017**: p. 7-12.
3. Lite, T., *Android to launch tensorflow lite for mobile machine learning*. 2017.
4. Mulfari, D., A. Palla, and L. Fanucci, *Embedded Systems and TensorFlow Frameworks as Assistive Technology Solutions.* Stud Health Technol Inform, 2017. **242**: p. 396-400.
5. Tran, D., *How to train your own Object Detector with TensorFlow's Object Detector API.* URI: https://medium. com/towards-data-science/how-to-train-yourown-object-detector-with-TensorFlows-object-detector-api-bec72ecfe1d9. Cited November, 2017. **5**: p. 2017.
6. Wongsuphasawat, K., et al., *Visualizing Dataflow Graphs of Deep Learning Models in TensorFlow*. IEEE Trans Vis Comput Graph, 2018. **24**(1): p. 1-12.
7. Fuentes, A., et al., *A Robust Deep-Learning-Based Detector for Real-Time Tomato Plant Diseases and Pests Recognition.* Sensors (Basel), 2017. **17**(9).
8. Bradski, G. and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*. 2008: " O'Reilly Media, Inc.".
9. Huang, J., et al. *Speed/accuracy trade-offs for modern convolutional object detectors*. in *IEEE CVPR*. 2017.
10. Ren, S., et al. *Faster r-cnn: Towards real-time object detection with region proposal networks*. in *Advances in neural information processing systems*. 2015.
11. Lu, J., et al., *Detection of multi-tomato leaf diseases (late blight, target and bacterial spots) in different stages by using a spectral-based sensor.* Sci Rep, 2018. **8**(1): p. 2793.
12. Schaefer, S.C., et al., *Enhanced resistance to early blight in transgenic tomato lines expressing heterologous plant defense genes.* Planta, 2005. **222**(5): p. 858-66.
13. Xie, C., et al., *Detection of early blight and late blight diseases on tomato leaves using hyperspectral imaging.* Sci Rep, 2015. **5**: p. 16564.
14. Behmann, J., et al., *Specim IQ: Evaluation of a New, Miniaturized Handheld Hyperspectral Camera and Its Application for Plant Phenotyping and Disease Detection.* Sensors (Basel), 2018. **18**(2).
15. Bergenti, F., M.-P. Gleizes, and F. Zambonelli, *Methodologies and software engineering for agent systems: the agent-oriented software engineering handbook*. Vol. 11. 2006: Springer Science & Business Media.
16. Howard, A.G., et al., *Mobilenets: Efficient convolutional neural networks for mobile vision applications.* arXiv preprint arXiv:1704.04861, 2017.
17. Hoo-Chang, S., et al., *Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning.* IEEE transactions on medical imaging, 2016. **35**(5): p. 1285.

18. Krizhevsky, A., I. Sutskever, and G.E. Hinton. *Imagenet classification with deep convolutional neural networks*. in *Advances in neural information processing systems*. 2012.
19. Szegedy, C., et al. *Rethinking the inception architecture for computer vision*.
20. http://cocodataset.org/
21. Greenberg, M., *Development of web applications using Flask in Python*. Moscow: DMK, 2014.
22. Grinberg, M., *Designing a RESTful API with Python and Flask*. 2013.
23. https://developer.dji.com/mobile-sdk/documentation/sample-code/index.html