**Plant Disease Detection**

**Research Report I**

****

**Supervisor**

Mam Raheela Naseem

**Submitted by**

Horair Ahmad

AG # 2018-ag-8206

Mabtoor ul Shafiq

AG # 2018-ag-8208

Muhammad Waqas

AG # 2018-ag-8224

**BS [Computer Science]**

**Department of Computer Science**

**University of Agriculture, Faisalabad.**

Table of Contents

**1. INTRODUCTION**4

**1.1 Background**4

**1.2 Description**4

**1.3 Scope**4

**1.4 Objectives**4

**2. REQUIREMENTS**5

**2.1 Functional Requirements**5

**2.2 Non-Functional Requirements**5

**2.3 Hardware Requirements**5

**2.4 Software Requirements**5

**3. METHODOLOGY**6

**3.1 Tools & Technologies**7

**4. TIMELINE**8

List of Figures

**Figure 1: V-model Activities**  [6](#_Toc534026760)

**Figure 2: Tentative Timeline of the Project activities** [8](#_Toc534026760)

# **INTRODUCTION**

## **1.1 Background**

Human society needs to increase food production by an estimated 70% by 2050 to feed an expected population size that is predicted to be over 9 billion people. Currently, infectious diseases reduce the potential yield by an average of 40% with many farmers in the developing world experiencing yield losses as high as 100%. The widespread distribution of smartphones among crop growers around the world with an expected 5 billion smartphones by 2020 offers the potential of turning the smartphone into a valuable tool for diverse communities growing food. One potential application is the development of mobile disease diagnostics through machine learning and crowdsourcing.

## **Description**

The Goal is to minimize the yield loss caused by different diseases of plants. Objective is to create a mobile application to identify and detect diseases accurately and fastly. 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. 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. For this purpose, we will use deep learning model to classify different plants and diseases of the plants.

## **1.3 Scope**

First of all, data(images) of healthy and diseased leaves should be collected. Preprocessing of the data(images) which includes annotating and labelling of the data must be done. A model should be trained/developed on the training data and must be evaluated on test data. A mobile application should be developed which must be able to identify and detect diseased leaves of different plants.

## **1.4 Objectives**

* Goal: To build a mobile application to identify and detect the diseases of Plants.
* Objective 1: Data must be collected first.
* Objective 2: Data should be labelled and annotated before we pass it to model.
* Objective 3: Deep Learning model must be created.
* Objective 4: Model be evaluated using the test set of the data.
* Objective 5: Our trained model must be used in the mobile application.

# **REQUIREMENTS**

## **2.1 Functional Requirements**

### FR01: Take/select a picture

|  |  |
| --- | --- |
| FR01-01 | System should request to allow the camera. |
| FR01-02 | System should check the alignment of the image. |
| FR01-03 | System should request the user to allow the storage media. |

### FR02: Evaluate the Picture

|  |  |
| --- | --- |
| FR02-01 | System should send the captured picture to the database |
| FR02-02 | System should evaluate the picture either it is a healthy image or diseased. |

### FR03: Display Output

|  |  |
| --- | --- |
| FR03-01 | System should display the category of the Plant & disease. |
| FR03-02 | System should display the causes of the disease. |

## **2.2 Non- Functional Requirements**

**NFR01:** System shall remain available 24/7 to its users.

**NFR02:** System should take different image formats (jpeg, png)

**NFR03:** System shall process one image at a time.

**NFR04:** System should display error if the given image does not belong to the specified category of plants diseases.

## **2.3 Hardware Requirements**

Smartphone

RAM: 1gb or above

## **2.4 Software Requirements**

Android version: 4.0 or above

Internet Connectivity

# **METHODOLOGY**

The V model (Verification and Validation model) is an extension of the waterfall model. All the requirements are gathered at the start and cannot be changed. You have a corresponding testing activity for each stage. For every phase in the development cycle, there is an **associated testing phase.** The V model is highly disciplined, easy to understand, and makes project management easier. But it isn’t good for complex projects or projects that have unclear or changing requirements. This makes the V model a good choice for software where downtimes and failures are unacceptable.

We will use this process model because in our project requirements are completely specified in the beginning and there is no need to change them. As this model is sequential so it gives the option to complete one phase before we move to the next phase which is an ideal case to our project. As stated above, it gives us the option to test each phase before we move to next phase which gives us the flexibility.

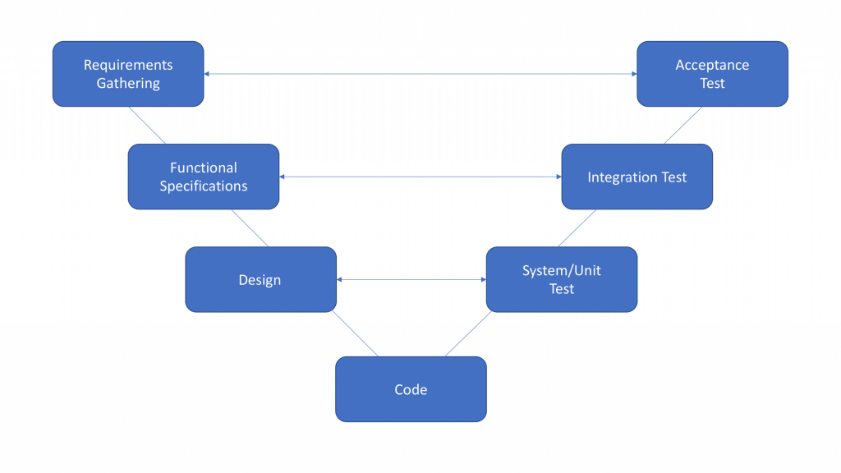


Figure 1:V-model Activities

## **Tools & Technologies**

**Labelimg**: A tool used for labeling and annotating images(data).

**Google colab**: Platform to create deep learning models.

**Vs code**: IDE to build mobile application.

**TensorFlow**: TensorFlow is a free and open-source software library for machine learning and artificial intelligence.

**NumPy**: NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices.

**Pandas**: Pandas is a software library written for the Python programming language for data manipulation and analysis.

**Matplotlib**: Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy.

**Keras**: Keras is an open-source software library that provides a Python interface for artificial neural networks.

**React-native**: Programming language to create a mobile app.

# **4. Timeline**

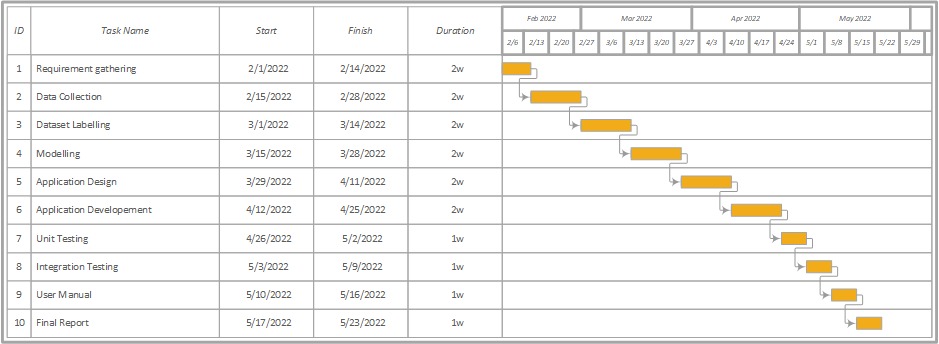
****

Figure 2: Tentative timeline of the project activities