**A Project Status Report**

**On**

**Farmer Assistant Software**

**By**

|  |  |
| --- | --- |
| Dharmendra Bajpai | (1534810031) |
| Avinash patel | (1534810021) |
| Aditya Pandey | (1534810006) |
| Himanshu Tripathi | (1534813022) |

**Bachelor of Technology**

**in**

**Computer Science and Engineering**

**Under the Supervision of**

**Mr Gaurav Bajpai**

**(Dept. Of Computer Science & Engineering)**

**PSIT COLLEGE OF ENGINEERING, KANPUR**

**to the**



**Faculty of Computer Science and Engineering**

**Dr. A.P.J. Abdul Kalam Technical University, Lucknow**

**(Formerly Uttar Pradesh Technical University)**

**December, 2018**

**DECLARATION**

We hereby declare that the project entitled ‘FARMER ASSISTANT SOFTWARE’ submitted for the B.Tech. degree is our original work and the project has not formed the basis for the award of any other degree associate-ship, fellowship or any other similar titles.

**Signature :**

**Name** **: Dharmendra Bajpai**

**Roll No** **: 1534810031**

**Date** **:**

**Signature :**

**Name** **: Avinash Patel**

**Roll No** **: 1534810021**

**Date** **:**

**Signature :**

**Name** **: Aditya Pandey**

**Roll No** **: 1534810006**

**Date** **:**

**Signature :**

**Name** **: Himanshu Tripathi**

**Roll No** **: 1534813022**

**Date** **:**

**(Signature)**

**(Mr Gaurav Bajpai)**

**(Assistant Professor)**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  | Title Page  Declaration | i  ii |
|  | List of Figures | iii |
|  | List of Tables (optional) | iv |
|  | Timeline / Gantt Chart | v |
|  |  |  |
| **1.** | **INTRODUCTION** | 1 |
|  | * 1. Problem Definition   2. Project Overview/Specifications   3. Hardware Specification   4. Software Specification | 1  2  3 |
|  |  |  |
| **2.** | **LITERATURE SURVEY** | **5** |
|  | 2.1 Existing System  2.2 Proposed System  2.3 Feasibility Study | 5  6  7 |
|  |  |  |
| **3.** | **SYSTEM ANALYSIS** |  |
|  | 3.1 Present Scenario  3.2 Proposed System  3.3 Requirement Specification  3.3.1 Functional Requirements  3.3.2 Non-Functional Requirements | 9  10  12  16  17 |
|  |  |  |
| **4.** | **SYSTEM DESIGN**  4.1 Application Development  4.1.1 Creation of server  4.1.2 Uploading Image  4.1.3 View Message  4.2 Image Processing  4.2.1 Image preprocessing  4.2.2 Segmentation  4.2.3 Feature Extraction  4.2.4 Classification  4.3 UML Diagrams  4.3.1 Use Case Diagram  4.3.2 Sequence Diagram | **40** |
| **5.** | **CONCLUSIONS / RECOMMENDATIONS** | **47** |
| **6.** | **FUTURE ENHANCEMENTS** | **48** |
| **7.** | **REFERENCES** | **49** |
| **8.** | **APPENDICES** | **50** |

***"Do FARMING by making Farmers Smart."***

|  |  |
| --- | --- |
| **1.** | **INTRODUCTION** |

India is an agricultural country. Today, India ranks second worldwide in farm output. Now days, a new concept of smart farming has been introduced where the field conditions are controlled and monitored using the self operating systems. The self recognition of the disease is based on the identification of the symptoms of disease. So that information about the disease occurrence could be quickly and accurately provided to the farmers, experts and researchers. This in turn reduces the monitoring of large field by human being. In disease recognition from image the key is to extract the characteristic feature of the diseased region. According to the disease the features may vary. The features that are extracted from the image are color , shape, texture etc. Sometimes for detection of the disease more features are extracted and these extracted feature would increase the hardware as well as software cost. This further causes increase in the complexity and the computation time. Hence it is necessary to reduce the feature data.

The occurrence of the disease on the plant may result in significant loss in both quality as well as the quantity of agricultural product. This can produce the negative impact on the countries whose economies are primarily dependent on the agriculture. Hence the detection of the disease in the earlier stages is very important to avoid the loss in terms of quality, quantity and finance. Usually the methods that are adopted for monitoring and management of plant leaf disease are manual. One such major approach is naked eye observation. But the requirement of this method is continuous monitoring of the field by a person having superior knowledge about the plants and its corresponding diseases. Moreover, appointing such a person would may prove costly. Another approach is seeking advice from the expert which may add the cost. Also, the expert must be available in time otherwise it may results in loss. Diagnosis of disease on plant can also be done in laboratory testing. But this method requires satisfactory laboratory conditions along with professional knowledge. The pathogen detection methods can provide more accurate results. As the tests are carried out of field the cost may be high and could be time consuming.

This software suggests a system which can provide more accurate results related to the identification and classification of disease and the best solution of that disease. It tries to replace the need of the experts to certain extent.

Here, the captured image is first preprocessed and then converted to HSI color space format by using segmentation. The features such as major axis, minor axis, eccentricity are extracted from the image. In the last step, these features are given to the classifier to classify the disease occurred on the leaf .All the above processing is done by inception V3 .

**1.1 Problem Definition**

One of the major problems which are faced by farmers is “*Biological Problem” (problem present in leaves).*

They also do not get the result as they are expecting for ; which is a main reason that most of the people do not want to do farming.

**1.2 Project Overview/Specifications**

**1.2.1 Objective**

* To help farmers by identifying the disease present in their plants leafs and to provide the best possible solution for that disease.
* To enhance the productivity of farming and to develop the interest towards farming.
* To reduce the effort of farmer in terms of time and money .

**1.2.2 Goal**

The main goal of this software is to identify the disease present in plant leafs with best possible accuracy. This software will also give the solution to the disease which is present in the plant leafs. It will reduce the effort of farmers and will increase the productivity of the farmers.

**1.2.3 Functionality**

This project Farming Assistant will make farmers smart. It will assist farmers by taking care of every type of crops with the help of **new coming technologies like transfer learning and image processing**, it will detect if any disease is spreading in the crops along with this it will also tell the farmers that what treatment should be given to protect the crops.

Farming is not strictly about the primary production, but play a major role in improving the efficiency of entire supply chain and alleviating food security concerns.

It will manage the crop failure risk and boost feed efficiency in livestock production.

This software will take picture of diseased leaf as input which is sent by farmers for the disease identification.

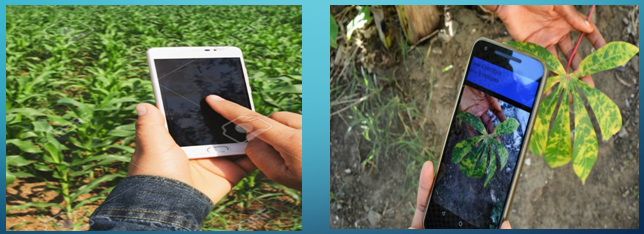


Fig.(1) Input to software.

Farmer can send the picture (input) by G-mail as well as by Whatsapp. After receiving the picture , this software will process the image as:

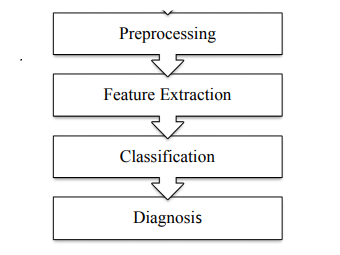


Fig (2)Input flow processing

The generated output will be send to the farmer through G-mail and sms text message. Farmer can easily use this software . The usage of this software is as simple as the use of mobile .

For the research scholar the **complete diagnosis** of the plant leaf can be provided which contain all type disease present in the plant leaf with its percentage.

The disease name and the solution of that disease will also be in the **Hindi Language.**

**1.3 Hardware Specification**

**1.4 Software Specification**

**2. LITERATURE REVIEW**

Various techniques of image processing and pattern recognition have been developed for detection of diseases occurring on plant leaves, stems, lesion etc. by the researchers. The sooner disease appears on the leaf it should be detected, identified and corresponding measures should be taken to avoid loss. Hence a fast, accurate and less expensive system should be developed. The researchers have adopted various methods for detection and identification of disease accurately. One such system uses thresholding and back propagation network. Input is grape leaf image on which thresholding is performed to mask green pixels. Using K-means clustering segmented disease portion is obtained. Then ANN is used for classification [1].The other method uses PCA and ANN.PCA is used to reduce the dimensions of the feature data. to reduce the no. of neurons in input layer and to increase speed of NN[2].Sometimes threshold cannot be fixed and object in the spot image cannot be located. Hence authors proposed LTSRG-algorithm for segmentation of image [3]. In cucumber leaf disease diagnosis, spectrum based algorithms are used [4].In the classification of rubber tree disease a device called spectrometer is used that measures the light intensity in electromagnetic spectrum. For the analysis SPSS is used [5].In citrus canker disease detection uses three level system. Global descriptor detects diseased lesion. To identify disease from similar disease based regions zone based local descriptor is used In last stage two level hierarchical detection structure identifies canker lesion [6]. For identification of disease on plant and stems first segmentation is carried using K-means clustering. Feature extraction is done by CCM method. Identification is done by using BPNN[7].With relevance to grapes, the fruit mostly suffer with tree types of diseases viz Powdery Mildew, Downy Mildew and Anthracnose. The two diseases are considered Powdery Mildew and Downy Mildew.

*Powdery Mildew:* Powdery mildew can infect all green parts of the grapevine. This disease is most easily recognized by the dusty appearance or white powdery growth occurring in patches on fruit or leaves. The white patches of powdery mildew produce millions of spores (conidia) which are spread by wind to cause more infections. Free moisture is not needed for secondary infection; temperature is the most important environmental factor. The image of grape leaf affected with late blight is shown below in Figure.

****

Fig () Diseased leaf

*Downy Mildew:* Early in the season, infected leaves develop yellowish-green lesions on their upper surfaces. As lesions expand, the affected areas turn brown, necrotic, or mottled. Severely infected leaves may curl and drop from the vine. The disease also attacks older leaves in late summer and autumn, producing a mosaic of small, angular, yellow to red-brown spots on the upper leaf surface. Downy mildew is favoured by warm, wet growing seasons.The image of leaf infected with early blight is shown below in Figure.



Fig () Diseased leaf

**2.1 Existing Method or System**

Earlier papers are describing to detection of the plant leaves diseases using various approaches are discussed below,

In [1], they discussed about automatic detection and classification of diseases. Plant disease spots are different in color but not in intensity. Thus color transform of RGB image is used for better segmentation of disease spots. Median filter is used for image smoothing and Otsu method is used to calculate threshold values to detect the disease spot. It doesn’t give accurate result for Dicot family plant.

P. Revathi and M. Hemalatha [4] investigated advance computing technology to assists the farmer in plant development process. This approach used mobile to capture infected cotton leaf images. RGB color feature segmentation is carried out to get disease spots. Edge detection technique is used for extraction of image features of spots to detect diseases. Neural network is used to categorize the diseases. The segmentation process is not suitable for Monocot family plant.

S. Dubey and R. Jalal [5] explored the concept of detection and classification of apple fruit diseases. The proposed approach is composed of three steps such as segmentation, feature extraction and classification. K-means clustering technique is used for the image segmentation. The features are extracted from the segmented image and images are classified based on a Multiclass Support Vector Machine (SVM). The proposed approach is specific to apple fruit diseases and cannot be extended to other fruit diseases.

In [6], the approach focused on Cercospora leaf spot detection in sugar beet using hybrid algorithms of template matching and support vector machine. The approach adopts three stages; first, a plant segmentation index of G-R is introduced to distinguish leaf parts from soil contained background for automatic selection of initial sub-templates. Second is robust template matching method adopted for continuous observation of disease development, foliar translation and dynamic object searching. Then, Support Vector Machine (SVM) is used to disease classification by a color features named two dimensional, xy color histogram. The segmentation process is not suitable for other Dicot family plant.

Yan, Han and Ming [7] proposed to select features of cotton disease leaf image by introducing fuzzy selection approach, fuzzy curves and fuzzy surfaces. The features which are extracted from fuzzy selection approach are used for diagnosing and identifying diseases. This approach removes the dependent features of image so as to reduce the number of features for classification.

Sannakki, Rajpurohit, Nargund and Kulkarni [8] proposed an approach to diagnose the disease using image processing and artificial intelligence techniques on images of grape plant leaf. The input image of grape leaf is complex at background. The thresholding is used to mask green pixels and image is processed to remove noise using anisotropic diffusion. Then, segmentation is done using K-means clustering technique. The diseased portion from segmented images is identified. The results were classified using back propagation neural network.

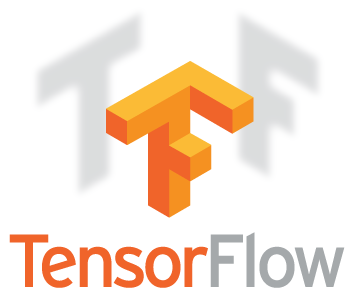
In [9], they investigated approach for automatic detection of chilies plant diseases. For that, CIELab color transformation model is used to extract color feature from infected image.

Compare the color feature for detection of disease. There is no effective work done in feature extraction. But it could yield more result accuracy if appropriate work would have been done.

Next paper [10] discussed about the monitoring of grapes and apples plant diseases. It suggests a solution to farmers for healthy yield and productivity. K-means clustering is used for segmentation and artificial neural network is used for classification of features. Also back propagation concept is used for counting the weight of mango. Morphology, color and texture features are extracted for classification.

**2.2 Proposed System**

We have created the system based upon image processing and machine learning with the help of tensor flow library.



TensorFlow is a multipurpose machine learning framework. TensorFlow can be used anywhere from training huge models across clusters in the cloud, to running models locally on an embedded system like your phone.

Install TensorFlow

Before we can begin the tutorial you need to install TensorFlow version 1.7, and PILLOW.

If you have a working python environment you can install them with:

pip install --upgrade "tensorflow==1.7.\*"

pip install PILLOW

If this doesn't work, follow the [instructions here](https://www.tensorflow.org/versions/r1.7/install/).

If you have the git repository from the first codelab

This codelab uses files generated during the [TensorFlow for Poets 1](https://codelabs.developers.google.com/codelabs/tensorflow-for-poets/index.html" \t "_blank)codelab. If you have not completed that codelab we recommend you go do it now. If you prefer not to, instructions for downloading the missing files are given in the next subsection.

In TensorFlow for Poets 1, you also cloned the relevant files for this codelab. Ensure that it is your current working directory, checkout the branch and check the contents, as follows:

cd tensorflow-for-poets-2

ls

This directory should contain three other subdirectories:

* The android/tflite directory contains all the files necessary to build a simple Android app using TFLite to classify images as it reads them from the camera. You will replace the model files with your customized versions.
* The scripts/ directory contains the python scripts you'll be using throughout the tutorial. These include scripts to prepare, test and evaluate the model.
* The tf\_files/ directory contains the files you should have generated in the first part. At minimum you should have the following files containing the retrained tensorflow program:

ls tf\_files/

retrained\_graph.pb retrained\_labels.txt

Otherwise (if you don't have the files from the first Codelab)

Clone the Git repository

The following command will clone the Git repository containing the files required for this codelab:

git clone https://github.com/googlecodelabs/tensorflow-for-poets-2

Now cd into the directory of the clone you just created. That's where you will be working for the rest of this codelab:

cd tensorflow-for-poets-2

The repo contains three directories: android/, scripts/, and tf\_files/

Checkout the files from the end\_of\_first\_codelab branch

git checkout end\_of\_first\_codelab

ls tf\_files

## Test the model

Next, verify that the model is producing reasonable results before starting to modifying it.

The scripts/ directory contains a simple command line script, label\_image.py, to test the network. Now we'll test label\_image.py on this picture of some daisies:



Image CC-BY, by [Fabrizio Sciami](https://www.flickr.com/photos/_fabrizio_/3475870145/" \t "_blank)

Now test the model. If you are using a different architecture you will need to set the "--input\_size" flag.

python -m scripts.label\_image \

--graph=tf\_files/retrained\_graph.pb \

--image=tf\_files/flower\_photos/daisy/3475870145\_685a19116d.jpg

The script will print the probability the model has assigned to each flower type. Something like this:

Evaluation time (1-image): 0.140s

daisy 0.7361

dandelion 0.242222

tulips 0.0185161

roses 0.0031544

sunflowers 8.00981e-06

This should hopefully produce a sensible top label for your example. You'll be using this command to make sure you're still getting sensible results as you do further processing on the model file to prepare it for use in a mobile app.

## Using TOCO

Mobile devices have significant limitations, so any pre-processing that can be done to reduce an app's footprint is worth considering. With TFLite a new graph converter is now included with the TensorFlow installation. This program is called the "[TensorFlow Lite Optimizing Converter](https://github.com/tensorflow/tensorflow/blob/master/tensorflow/contrib/lite/toco/README.md" \t "_blank)" or TOCO.

It is installed as a command line script, with TensorFlow, so you can easily access it. To check that toco is correctly installed on your machine, try printing the TOCO help, with the following command:

toco --help

We will use toco to optimize our model, and convert it to the TFLite format. toco can do this in a single step, but we will do it in two so that we can try out optimized model in between.

### Convert to model to TFLite format

While toco has advanced capabilities for dealing with [quantized graphs](https://www.tensorflow.org/performance/quantization), it also applies several optimizations that are still useful for our graph, (which does not use quantization). These include pruning unused graph-nodes, and performance improvements by joining operations into more efficient composite operations.

The pruning is especially helpful given that TFLite does not support training operations yet, so these should not be included in the graph.

While TOCO can be used to optimize regular graph.pb files, TFLite uses a different serialization format from regular TensorFlow. TensorFlow uses [Protocol Buffers](https://github.com/google/protobuf#protocol-buffers---googles-data-interchange-format), while TFLite uses [FlatBuffers](https://google.github.io/flatbuffers/" \t "_blank).

The primary benefit of FlatBuffers comes from the fact that they can be memory-mapped, and used directly from disk without being loaded and parsed. This gives much faster startup times, and gives the operating system the option of loading and unloading the required pages from the model file, instead of killing the app when it is low on memory.

We can create the TFLiteFlatBuffer with the following command:

IMAGE\_SIZE=224

toco \

--graph\_def\_file=tf\_files/retrained\_graph.pb \

--output\_file=tf\_files/optimized\_graph.lite \

--input\_format=TENSORFLOW\_GRAPHDEF \

--output\_format=TFLITE \

--input\_shape=1,${IMAGE\_SIZE},${IMAGE\_SIZE},3 \

--input\_array=input \

--output\_array=final\_result \

--inference\_type=FLOAT \

--input\_data\_type=FLOAT

This should output a "optimized\_graph.lite" in your "tf\_files" directory.

**2.3 Feasibility Study**

Plant diseases have turned into a nightmare as it can causesignificant reduction in both qualityand quantity ofagricultural products (Shen Weizheng, 2008influence the countries that primarily depend on agriculture inits economy (Prasad Babu, 2010). Consequently, detection of

plant diseases is an essential research topic as it may proveuseful in monitoring large fields of crops, and thusautomatically detect the symptoms of diseases as soon as theyappear on plant leafs. Monitoring crops for to detecting

diseases plays a key role in successful cultivationBabu, 2010 and Weizheng, 2008). The nakedeye observationof experts is the main approach adopted in practiceWeizheng, 2008). However,this requires continuousmonitoring of experts which might be prohibitively expensivein large farms. Further, in some developing countries, farmersmay have to go long distances to contact experts, this makesconsulting experts to very expensive and time consuming(Prasad Babu, 2010). Therefore; looking for a fast, automatic,less expensive and accurate method to detect plant diseasecases is of great realistic significance

**Techniques**

In this section, we consider the general flow of the varioussteps that are being performed in order to achieve the desiredresult. The proposed approach consists of four main steps:imageacquisition of leaves, extraction of leaves from complexbackground, statistical analysis and disease classification.The general flow of the disease detection system is illustrated



**Block diagram of disease detection system**



**Figure . Example of the output of Classification clustering for a leafthat is infected with early scorch disease. (a)The infected leafpicture. (b, c, d, e) the pixels of the 1st,2nd,3rd and 4thcluster,respectively.(f) a single gray-scale image with the pixel colour**

**based on cluster index.**

**3. SYSTEM ANALYSIS & DESIGN**

**3.1 PRESENT SCENARIO**

Plant diseases cause periodic outbreak of diseases which leads to large scale death and famine. It is estimated that the outbreak of helminthosporiose of rice in north eastern India in 1943 caused a heavy loss of food grains and death of a million people. Since the effects of plant diseases were devastating, some of the crop cultivation has been abandoned. It is estimated that 2007 plant disease losses in Georgia (USA) is approximately $653.06 million (Jean, 2009). In India no estimation has been made but it is more than USA because the preventive steps taken to protect our crops are not even one-tenth of that in USA.

The naked eye observation of experts is the main approach adopted in practice for detection and identification of plant diseases. But, this requires continuous monitoring of experts which might be prohibitively expensive in large farms. Further, in some developing countries, farmers may have to go long distances to contact experts, this makes consulting experts too expensive and time consuming and moreover farmers are unaware of non-native diseases.



**Fig 3.1: Block Diagram for Farmer’s portal.**

It is envisaged to make available relevant information and services to the farming community and private sector through the use of information and communication technologies, to supplement the existing delivery channels provided for by the department. Farmers‟ Portal is an endeavor in this direction to create one stop shop for meeting all information needs relating to agriculture of an Indian farmer. Once in the Farmers‟ Portal, a farmer will be able to get all relevant information on specific subjects around his village/block/district or state. Using the farmers‟ portal the farmer calls this service center and clears all his queries. The people present at the service search for relevant information or they are trained with all information to help farmers.

This also has a disadvantage as the people at the service center cannot see the exact problem the farmer is facing. They cannot imagine the severity of the disease completely and hence at times it may result in wrong disease detection. This may also lead in destruction of the crop.

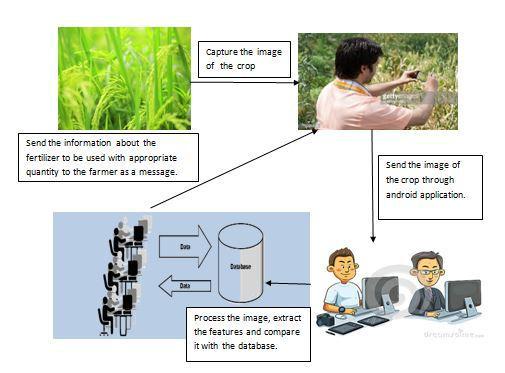
**3.2 PROPOSED SYSTEM**

In the proposed system at first the images are acquired from the farmer. The images are received from the farmer via the Android Application developed exclusively for the service of the farmer. The images are uploaded by the farmer by choosing the appropriate image of the leaf or the stem preferably from the Choose File option. On uploading a image the farmer receives a ID which has to be used later by him to check the pesticides for the affected disease.

The image uploaded by the farmer is processed by the MATLAB. Then image-processing techniques are applied to the acquired images to extract useful features that are necessary for further analysis. After that, several analytical techniques are used to classify the images according to the specific problem at hand.

The disease type is detected and displayed by the MATLAB. The affected area is also displayed to identify the severity of the disease. The pesticides for the detected disease and the amount to be given to the plant are entered into the database. The farmer in order to see the details has to click another button in the app which is View Message. On entering the ID which was previously displayed to the farmer the farmer can view the details that were uploaded.

Automatic detection of plant diseases is an important research topic as it may prove benefits in monitoring large fields of crops, and thus automatically detect the diseases from the symptoms that appear on the plant leaves. This enables machine vision that is to provide image based automatic inspection, process control and robot guidance. Comparatively, visual identification is labor intensive, less accurate and can be done only in small areas.



**Fig 3.2: Fertilization Management for crops.**

**3.3 REQUIREMENT SPECIFICATION**

**3.3.1 FUNCTIONAL REQUIREMENTS**

Functional requirements define the internal workings of the software: that is, the technical details, data manipulation and processing and other specific functionality that show how the use cases are to be satisfied. They are supported by non-functional requirements, which impose constraints on the design or implementation.

**SOFTWARE REQUIREMENTS**

Language

: PYTHON

OS

: Windows 10 (64 bit)

IDE

: Android Studio, MATLAB

**HARDWARE REQUIREMENTS**

Processor

: Above 1.5GHZ

Hard Disk

: 80GB

RAM

: 2GB

**3.3.2 NON-FUNCTIONAL REQUIREMENTS**

Non-functional requirements are requirements which specify criteria that can be used to judge the operation of a system, rather than specific behaviors. This should be contrasted with functional requirements that specify specific behavior or functions. Typical non-functional requirements are reliability, scalability, and cost. Non-functional requirements are often called the ilities of a system. Other terms for non-functional requirements are "constraints", "quality attributes" and "quality of service requirements".

**Reliability:** If any exceptions occur during the execution of the software it should be caught andthereby prevent the system from crashing.

**Scalability:** The system should be developed in such a way that new modules and functionalitiescan be added, thereby facilitating system evolution.

**Cost:** The cost should be low because a free availability of software package.

**4. SYSTEM DESIGN**

**4.1 APPLICATION DEVELOPMENT**

The android application for the farmer was developed using the android studio. **Android** **Studio** is the official Integrated Development Environment (IDE) forAndroidplatformdevelopment. The development of the app had various stages such as creation of server, storing images in the database, accessing images from database, updating information into the database and extracting message from the database. The procedure first was started by creating a server by creating account in Hostinger and a database and consequently a table for storing images was created. Later a php script was written for connecting to the server and the database to the Android Application.

The layout of Android app contained of three buttons which are Choose File, Upload Image and View Message. The View Message in turn would open a layout which contained a text box and a button called Get Message and a text area to display the message as shown above

**4.1.1 CREATION OF SERVER**

The first and foremost task is to create a server and this is done by creating a account. A relevant database is created in PHPMyadmin. The table images is subsequently created with two columns one being ID and the other being „image‟ which is of type BLOB. When an image is uploaded the image is uploaded into this table. Another table called message table is also created so that after processing image we can store values into the table.

**4.1.2 UPLOADING IMAGE**

When the farmer selects the Choose File option gallery folder opens so that the farmer can select image from the folder. On selection of the picture the picture gets displayed. On selection of right image and the image being displayed in the app the farmer can click on Upload image option. The image gets uploaded to the server and gets stored in the database. An ID is returned to the farmer with a message that image was uploaded successfully.

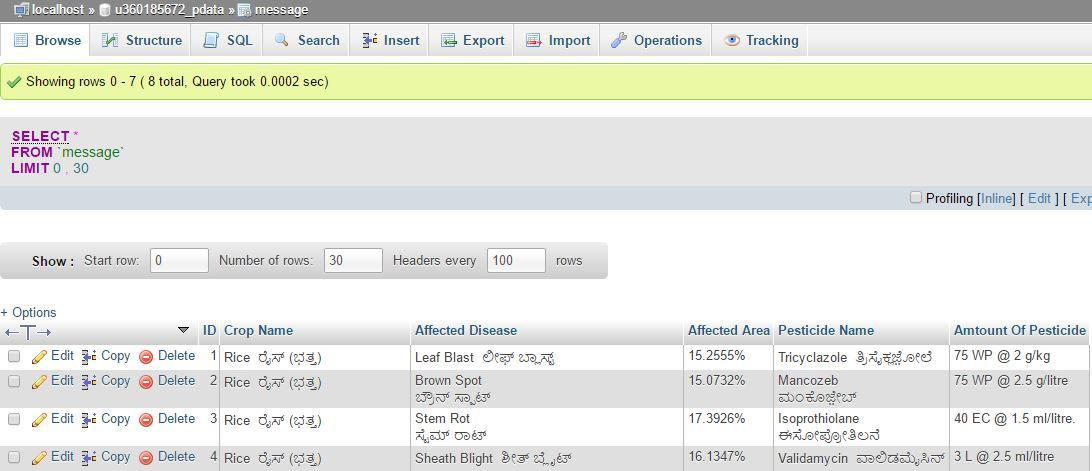
For uploading the image a php script was uploaded into the server. The script was given a link in Main Activity. So that on clicking upload image button in turn the php script is called. The Upload Image inserts the image into the table by using INSERT query. It displays “Successfully Uploaded Image” and the ID s displayed. The ID is displayed using the attribute LAST\_INSERT\_ID. The ID is auto incremented value which is made so while creating the table.

**4.1.3 VIEW MESSAGE**

The View Message is in turn another activity which gets opened. It has a text box where the ID that was generated for the farmer which is unique to every farmer has to be entered by him correctly.

A php script GETMESSAGE is been uploaded to the server. This script fetches the message from the database. In the database created earlier which had table „images‟ where the uploaded images are stored, another table called „message‟ was created. After analyzing the image in the MATLAB the details of the disease and the amount of fertilizer are updated into the table as per the ID of the image.

The message is fetched from the table and its displayed in the text area of the activity layout. Details such as ID, Crop Name, Disease name, Pesticide, Amount of pesticide, Affected Region are displayed.



**Fig 4.3: Message table in the server where the message contents are uploaded**

**4.2 IMAGE PROCESSING**

**Digital image processing** is the use of computeralgorithmsto performimageprocessing on digital images. It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and signal distortion during processing. Since images are defined over two dimensions (perhaps more) digital image processing may be model in the form of [multidimensional systems].

The image is acquired from the server by using another php script GetImage. It has query to access the image. The image can be accessed by MATLAB using the URL of the php file in the server. The GUI of MATLAB has an interface which consists of a button called Image Select. On clicking Image Select the image directly gets downloaded from server. The acquired image is preprocessed so that it can be used for further use.

The following steps are followed for detecting disease in crop:

* Image Preprocessing
* Segmentation
* Feature Extraction
* Classification

**4.2.1 IMAGE PREPROCESSING**

The image acquired is preprocessed. The preprocessing starts by converting the RGB image to L\*a\*b\* color space. The L\*a\*b\* color space consists of Luminosity layer L\*, chromacity layer a\* and b\*. All of the color information is stored in the layers a\* and b\*. It requires to make color form so that the RGB colored image is converted to L\*a\*b\* space. The function is makecform(), later the format is applied to the image that was acquired.

**4.2.2 SEGMENTATION**

There are several algorithms used for segmentation but one of the best methods used for detection of disease is k- means clustering. ***k*-means clustering** is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. *k*-means clustering aims to partition *n* observations into *k* clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells.*k*-means clustering tends to find clusters of comparable spatial extent, while the expectation-maximization mechanism allows clusters to have different shapes.

The algorithm has a loose relationship to the *k*-nearest neighbor classifier, a popular machine learning technique for classification that is often confused with *k*-means because of the *k* in the name. One can apply the 1-nearest neighbor classifier on the cluster centers obtained by *k*-means to classify new data into the existing clusters.

Classify the colors a\*b\* color space using k-means clustering. Since the image has 3 colors we create three clusters. Measure the distance Euclidean Distance Metric. Label every pixel in that image using results from K means.

Given a set of observations (**x**1, **x**2, …, **x***n*), where each observation is a *d*-dimensional real vector, *k*-means clustering aims to partition the *n* observations into *k* (≤ *n*) sets**S** = {*S*1, *S*2, …, *Sk*} so as to minimize the within-cluster sum of squares (WCSS) (sum of distance functions of each point in the cluster to the K center). In other words, its objective is to find:



…… (4.1)

where ***μ****i* is the mean of points in *Si*.

**Cluster analysis** or **clustering** is the task of grouping a set of objects in such a way thatobjects in the same group (called a **cluster**) are more similar (in some sense or another) to each other than to those in other groups (clusters).

In cluster analysis, the *k*-means algorithm can be used to partition the input data set into *k* partitions.

Label every pixel in the image using results from K means. Then a blank cell array is created to store the results of clustering. Followed by create RGB label using pixel\_labels. Selection of appropriate cluster is another important aspect. The cluster which displays the maximum disease affected part is to be selected. In the next step of feature extraction, the features of the selected cluster are extracted.

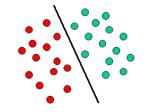
**4.2.3 FEATURE EXTRACTION**

The features of the selected cluster are extracted. The selected image is converted to grayscale since the image is in RGB format. At the next step the Gray Level Cooccurance Matrices (GLCM). The required statistics are derived from Gray level cooccurance Matrices (GLCM). The following 13 features that is extracted and evaluated:

Contrast, Corelation, Energy, Homogenity, Mean, Standard Deviation, Entopy, RMS. Variance, Smoothness, Kurtosis, Skewness. The thirteen features are stored in an array.

**4.2.4 CLASSIFICATION**

Support Vector Machines are based on the concept of decision planes that define decision boundaries. A decision plane is one that separates between a set of objects having different class memberships.

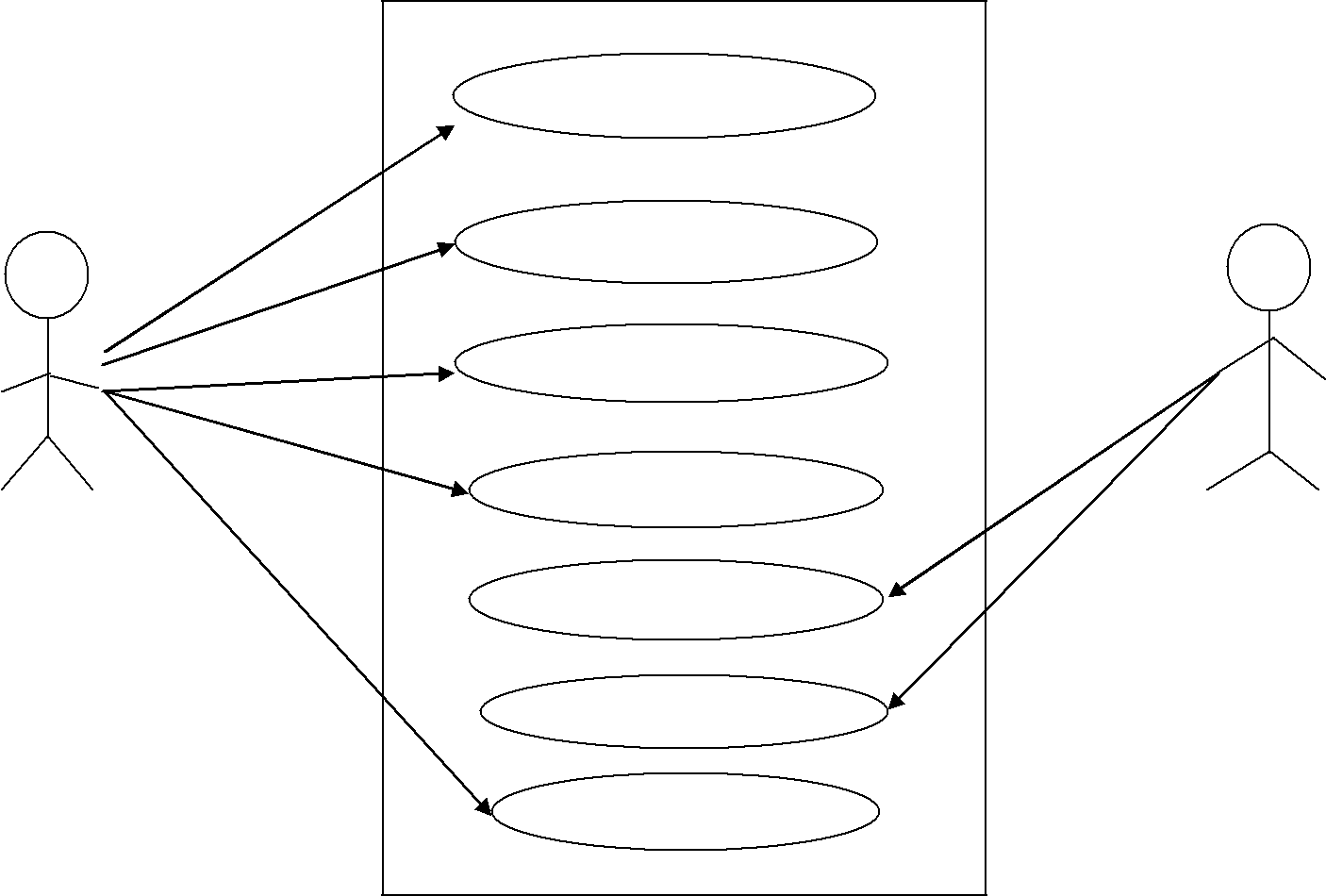


The above is a classic example of a linear classifier, i.e., a classifier that separates a set of objects into their respective groups (GREEN and RED in this case) with a line. Most classification tasks, however, are not that simple, and often more complex structures are needed in order to make an optimal separation, i.e., correctly classify new objects (test cases) on the basis of the examples that are available (train cases). This situation is depicted in the illustration below. Compared to the previous schematic, it is clear that a full separation of the GREEN and RED objects would require a curve (which is more complex than a line).

**4.3 UML DIAGRAMS**

**4.3.1 USE CASE DIAGRAM**

A behavioral diagram that shows a set of objects and their relationships.



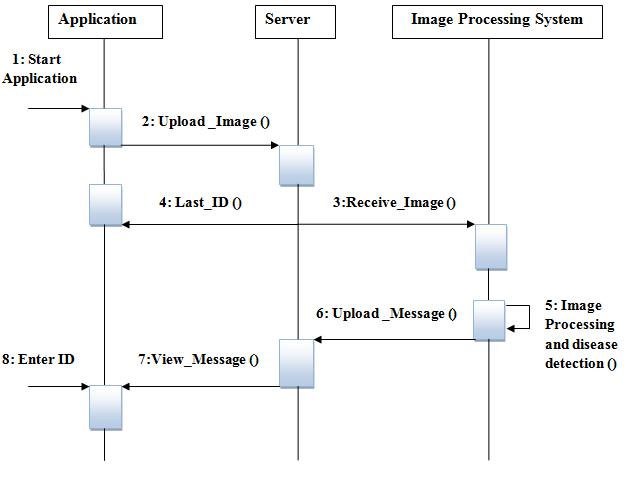
**Start Application**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Select Image** |  |  |
|  | **Upload Image** |  |  |
|  | **Receive Image** |  |  |
| **Farmer** | **Process the image** | **Service Center** |  |
|  |  |  |
|  | **Upload message** |  |  |
|  | **View message** |  |  |
|  | **Fig 4.7: Use Case Diagram** |  |  |

The above diagram represents the actors, Farmer and Service Center. The Use cases which drive the process to completion are Start Application, Select Image, Upload Image, Receive Image, Process the Image, Upload Message, View message.

**4.3.2 SEQUENCE DIAGRAM**

A behavioral diagram that shows an interaction, emphasizing the time ordering of messages.



**Fig 4.8: Sequence Diagram**

The above diagram represents interaction between various components. Application that is installed on farmer‟s phone is used to upload the image of crop. An Auto incremented ID is sent back to the farmer. The image received is processed and the message is uploaded. Using the ID the farmer retrieves the message.

**Fig 4.2: Image table in server where the images are uploaded**