# AgroDrone: Crop Disease Detection Using Drone
## A Project Report

Submitted in partial fulfillment for the award of the degree

of

**Bachelor of Technology**

in

**Computer Science and Engineering**

of

**Cochin University of Science and Technology**

by

**Basim Rauf(12151203)**

**Melvin Isaac(12151230)**

**Nilufar K P(12151231)**

**Greeshma Girish(14151204)**

**Ranjitha Menon(14151207)**

Department of Computer Science and Engineering

College of Engineering, Karunagappally - 690523

April, 2018

# COLLEGE OF ENGINEERING KARUNAGAPPALLY
# KOLLAM - 690523, KERALA

## BONAFIDE CERTIFICATE

Certified that this project report on **AgroDrone: Crop Disease Detection Using Drone** is a bonafide work of the major project carried out at our department by BASIM RAUF, MELVIN ISAAC, NILUFAR K P, GREESHMA GIRISH, RANJITHA MENON in 8th Semester in partial fulfilment of the requirements for the award of Bachelor of Technology Degree in Computer Science and Engineering by Cochin University of Science and Technology.

PROJECT CO-ORDINATORS

**Prof. MANOJ RAY D**
ASSOCIATE PROFESSOR
Dept. of Computer
Science and Engineering

**Mrs. REMYA R S**
ASSISTANT PROFESSOR
Dept. of Computer
Science and Engineering

PROJECT GUIDE

**Mrs. REMYA R S**
ASSISTANT PROFESSOR
Dept. of Computer
Science and Engineering

HEAD OF THE DEPARTMENT

**Dr. BINU V P**
ASSOCIATE PROFESSOR
Dept. of Computer
Science and Engineering

# Declaration

I hereby declare that the project titled **AgroDrone: Crop Disease Detection Using Drone**, under the guidance of Mrs. Remya R S (Assistant Professor, Dept. of Computer Science), Mr. Manoj Ray D (Assistant Professor, Dept. of Computer Science), College of Engineering Karunagappally for the partial fulfilment of B.Tech (Computer Science and Engineering) degree course of the Cochin University of Science and Technology is my original and true work. The findings in this report are based on the data collected by me. Any part of this report has not been reproduced or copied from any other report of the university.

Place:Karunagappally

Basim Rauf

Melvin Isaac

Nilufar K P

Greeshma Girish

Ranjitha Menon

# Acknowledgement

Success and happiness are directly related to the achievements of the predetermined goals. The success of this project work is due to the varied contribution of several distinguished personalities. We take this wonderful opportunity to thank each and every one of them. It is our privilege to extend our sincere gratitude to all those who helped our team in the fruition of the project entitled **AgroDrone: Crop Disease Detection Using Drone** .

First of all we thank God Almighty for being the guide light throughout the project and helped us to complete it within stipulated time.

We remember with grateful thanks, the encouragement and support rendered by our principal Dr. Jaya V L. We express our deepest sense of gratitude to the head of the department, Dr. Binu V P for his valuable guidance.

We also express our heartfelt gratitude to our project coordinators,Prof. Manoj Ray D (Associate Professor of Department of Computer Science, College of Engineering, Karunagappally), Mrs. Remya R S (Assistant Professor of Department of Computer Science, College of Engineering, Karunagappally) and our project guide Mrs. Remya R S (Assistant Professor of Department of Computer Science, College of Engineering, Karunagappally) for timely suggestions and encouragement given for the successful completion of the project. We would always oblige for the helping hands of all other staff members of the department who directly or indirectly contributed in this venture.

Our overrunning debt also lays with family members, friends and well-wishers who helped in bringing out this project successfully.

# Abstract

Disease can negatively impact plant health before any visible signs like leaf discolouration are shown. While these stresses are invisible to the naked eye, cameras using special filters we could detect these subtle changes. The drone will use multi-spectral cameras, which uses special filters to capture reflected light from selected regions of the large crop field. Stressed plants typically display a 'spectral signature' that distinguishes them from healthy plants. The innovation of this project involves developing a solution to detect and classify infection in the crop. By understanding how different stresses impact each other and their 'spectral signatures' are used to develop solutions to better map and forecast crop diseases. This project can be implemented using drones and their multi-spectral sensors, as well as developing tools to train a computer program to analyse the images and classify them based on disease progression.

# Contents

# List of Figures

# Chapter 1

# INTRODUCTION

Agricultural drones represent a new way to collect field-level data. The most compelling reason for using drones is that the results are on-demand; whenever and wherever needed, the drone can be easily and quickly deployed. It's hard to beat the immediacy and convenience of planning the mission, collecting the data, and getting near real-time results; only drones over these benefits. Drones are adorable, requiring a very modest capital investment when compared to most farm equipment. They can pay for themselves and start saving money within a single growing season. They're safe and reliable. They are easy to integrate into the regular crop-scouting workout; while visiting a field to check for pests or other ground issues, the drone can be deployed to collect aerial data. Yet, the real advantages of drones are not about the hardware; the value is in the convenience, quality and utility of the final data product. Your time is valuable.

# Chapter 2

# SYSTEM ANALYSIS

## 2.1 Existing System

Thousands of farmers are killing themselves in India every year. They make this ultimate sacrifice not just because the weather gods have been brutal. It is also because they have been failed by crop insurance, their primary protection from climatic flippancy.

## 2.2 Limitations of Existing System

- Now a days farmers manually check their field for detection of any possible disease outbreaks.
- If the land area is very large then it's difficult to examine whole of the crops.
- It is physically tiring and exhausting.
- Time consuming and costly.

## 2.3 Study of Proposed System

Agricultural drones represent a new way to collect field-level data. The most compelling reason for using drones is that the results are on-demand; whenever and wherever needed, the drone can be easily and quickly deployed. It's hard to beat the im-

mediacy and convenience of planning the mission, collecting the data, and getting near real-time results; only drones over these benefits.

**Advantages of Proposed System** To provide the results on demand whenever and wherever needed.

- The drone can be easily and quickly deployed.
- Drones are affordable, only requiring a very modest capital investment when compared to most farm equipment
- Visiting acres to check for pests or other ground issues. The drone can be collect aerial data.
- Less timing. Flying through the air and collecting data takes less time.

Basic idea :

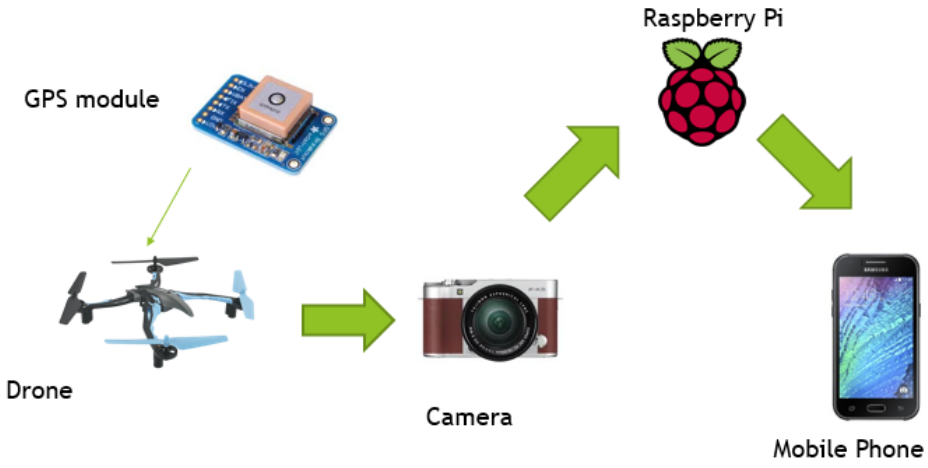


Figure 2.1: Basic Architecture of the System

## 2.4 Requirement Specification

### 2.4.1 Software Requirement Specification

[1] OPERATING SYSTEM : Windows 8 or above

### 2.4.2 Hardware Requirement Specification

[1] OPERATING SYSTEM : Intel(R)Core(TM)i3@1.90GHz or above

[2] RAM: 1 GB or more

[3] STORAGE CAPACITY: 40 GB

### 2.4.3 System Development Tool

[1] FRONT END:JAVA

[2] BACK END:MYSQL

## 2.5 About Java Technology

Java technology is both a programming language and a platform. Java is a set of computer software and specifications developed by Sun Microsystems, which was later acquired by the Oracle Corporation, that provides a system for developing application software and deploying it in a cross-platform computing environment. Java is used in a wide variety of computing platforms from embedded devices and mobile phones to enterprise servers and supercomputers. While they are less common than standalone Java applications, Java applets run in secure, sandboxed environments to provide many features of native applications and can be embedded in HTML pages.

Writing in the Java programming language is the primary way to produce code that will be deployed as byte code in a Java Virtual Machine (JVM); byte code compilers are also available for other languages, including Ada, JavaScript, Python, and Ruby. In addition, several languages have been designed to run natively on the JVM, including Scala, Clojure and Apache Groovy. Java syntax borrows heavily from C and C++, but object-oriented features are modeled after Smalltalk and Objective-C.[11] Java eschews certain low-level constructs such as pointers and has a very simple memory model where every object is allocated on the heap and all variables of object types are references. Memory management is handled through integrated automatic garbage collection performed by the JVM.

Java is

[1] Object Oriented In Java, everything is an Object. Java can be easily extended since it is based on the Object model.

[2] Platform Independent Unlike many other programming languages including C and C++, when Java is compiled, it is not compiled into platform specific machine, rather into platform independent byte code. This byte code is distributed over the web and interpreted by the Virtual Machine (JVM) on whichever platform it is being run on.

[3] Simple Java is designed to be easy to learn. If you understand the basic concept of OOP Java, it would be easy to master.

[4] Secure With Java's secure feature it enables to develop virus-free, tamper-free systems. Authentication techniques are based on public-key encryption.

[5] Architecture-neutral Java compiler generates an architecture-neutral object file format, which makes the compiled code executable on many processors, with the presence of Java runtime system.

[6] Portable Being architecture-neutral and having no implementation dependent aspects of the specification makes Java portable. Compiler in Java is written in ANSI C with a clean portability boundary, which is a POSIX subset.

[7] Robust Java makes an effort to eliminate error prone situations by emphasizing mainly on compile time error checking and runtime checking.

[8] Multithreaded With Java's multithreaded feature it is possible to write programs that can perform many tasks simultaneously. This design feature allows the developers to construct interactive applications that can run smoothly.

[9] Interpreted Java byte code is translated on the fly to native machine instructions and is not stored anywhere. The development process is more rapid and analytical since the linking is an incremental and light-weight process.

[10] High Performance With the use of Just-In-Time compilers, Java enables high performance.

[11] Distributed Java is designed for the distributed environment of the internet.

[12] Dynamic Java is considered to be more dynamic than C or C++ since it is designed to adapt to an evolving environment. Java programs can carry extensive amount of run-time information that can be used to verify and resolve accesses to objects on run-time.

## 2.6 Overview of the Software Development Process

### 2.6.1 The Java Platform

The Java platform is a suite of programs that facilitate developing and running programs written in the Java programming language. A Java platform will include an execution engine (called a virtual machine), a compiler and a set of libraries; there may also be additional servers and alternative libraries that depend on the requirements. Java is not specific to any processor or operating system as Java platforms have been implemented for a wide variety of hardware and operating systems with a view to enable Java programs to run identically on all of them. Different platforms target different classes of device and application domains:

Java Card: A technology that allows small Java-based applications (applets) to be run securely on smart cards and similar small-memory devices.
Java ME (Micro Edition): Specifies several different sets of libraries (known as profiles) for devices with limited storage, display, and power capacities. It is often used to develop applications for mobile devices, PDAs, TV set-top boxes, and printers.
Java SE (Standard Edition): For general-purpose use on desktop PCs, servers and similar devices.
Java EE (Enterprise Edition): Java SE plus various APIs which are useful for multi-tier clientserver enterprise applications.
The Java platform consists of several programs, each of which provides a portion of its overall capabilities. For example, the Java compiler, which converts Java source code into Java bytecode (an intermediate language for the JVM), is provided as part of the Java Development Kit (JDK). The Java Runtime Environment (JRE), complementing the JVM with a just-in-time (JIT) compiler, converts intermediate bytecode into native machine code on the fly. The Java platform also includes an extensive set of libraries.

The essential components in the platform are the Java language compiler, the libraries, and the runtime environment in which Java intermediate bytecode executes according to the rules laid out in the virtual machine specification.

Java Virtual Machine

Main article: Java Virtual Machine

The heart of the Java platform is the concept of a ”virtual machine” that executes Java bytecode programs. This bytecode is the same no matter what hardware or operating system the program is running under. There is a JIT (Just In Time) compiler within the Java Virtual Machine, or JVM. The JIT compiler translates the Java bytecode into native processor instructions at run-time and caches the native code in memory during execution.

The use of bytecode as an intermediate language permits Java programs to run on any platform that has a virtual machine available. The use of a JIT compiler means that Java applications, after a short delay during loading and once they have ”warmed up” by being all or mostly JIT-compiled, tend to run about as fast as native programs.[17][18][19] Since JRE version 1.2, Sun's JVM implementation has included a just-in-time compiler instead of an interpreter.

Although Java programs are cross-platform or platform independent, the code of the Java Virtual Machines (JVM) that execute these programs is not. Every supported operating platform has its own JVM.

In most modern operating systems (OSs), a large body of reusable code is provided to simplify the programmer's job. This code is typically provided as a set of dynamically loadable libraries that applications can call at runtime. Because the Java platform is not dependent on any specific operating system, applications cannot rely on any of the pre-existing OS libraries. Instead, the Java platform provides a comprehensive set of its own standard class libraries containing many of the same reusable functions commonly found in modern operating systems. Most of the system library is also written in Java. For instance, the Swing library paints the user interface and handles the events itself, eliminating many subtle differences between how different platforms handle components.

The Java class libraries serve three purposes within the Java platform. First, like other standard code libraries, the Java libraries provide the programmer a well-known set of functions to perform common tasks, such as maintaining lists of items or perform-

ing complex string parsing. Second, the class libraries provide an abstract interface to tasks that would normally depend heavily on the hardware and operating system. Tasks such as network access and file access are often heavily intertwined with the distinctive implementations of each platform. The java.net and java.io libraries implement an abstraction layer in native OS code, then provide a standard interface for the Java applications to perform those tasks. Finally, when some underlying platform does not support all of the features a Java application expects, the class libraries work to gracefully handle the absent components, either by emulation to provide a substitute, or at least by providing a consistent way to check for the presence of a specific feature.

### 2.6.2 JSP and J2EE Edition

The ”Java Framework” (Java 2 Platform) is composed of three editions, each desgined for different purposes:

[1] J2ME: Java 2 Micro Edition is intended for development of embedded applications, for PDAs and mobile terminals.

[2] J2SE: Java 2 Standard Edition is designed for development of applications for personal computers.

[3] J2EE: Java 2 Enterprise Edition, designed for professional use (implementation on servers).

Each edition provides a complete environment for the development and execution of Java-based applications and includes a JVM (Java virtual machine) and a set of classes.

J2EE (Java 2 Enterprise Edition) is a standard proposed by Sun, supported by a consortium of international companies, to define a standard for developing multi-level (component-based) enterprise applications.

The services (API) offered and runtime infrastructure, is generally referred as the ”J2EE platform” and it includes:

[1] Specifications for the application server, that is to say, the execution environment: J2EE defines the roles/interfaces for the applications and the environment in which

they are executed. These recommendations allow third-party companies to develop application servers conform to the specifications defined without having to re-develop the main services.

[2] Services (through API), that is to say independent Java extensions to provide a number of standard features. Sun provides a minimal implementation of these API, called as J2EE SDK (J2EE Software Development Kit).

J2EE relies entirely on Java, it enjoys the advantages and disadvantages of this language, especially the portability and maintainability of code.

In addition, the J2EE architecture based on discrete, interchangeable and distributed components, which implies the following:

[1] easily extend the architecture.

[2] a system based on J2EE mechanisms can have high availability to ensure a good quality of service.

[3] the maintainability of applications is facilitated.

Java Platform, Enterprise Edition or Java EE is a widely used computing platform for development and deployment of enterprise software (network and web services).Java EE was formerly known as Java 2 Platform, Enterprise Edition or J2EE.

The platform uses the object-oriented Java programming language. It is part of the Java software-platform family. Java EE extends the Java Platform, Standard Edition (Java SE), providing an API for object-relational mapping, distributed and multitier architectures, and web services. The platform incorporates a design based largely on modular components running on an application server. The platform emphasizes convention over configuration and annotations for configuration. Optionally XML can be used to override annotations or to deviate from the platform defaults.

### 2.6.3 MySQL

MySQL is the most popular Open Source Relational SQL database management system. MySQL is one of the best RDBMS being used for developing web-based software

applications. This tutorial will give you quick start with MySQL and make you comfortable with MySQL programming.

## MySQL DATABASE

MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. MySQL is developed, marketed, and supported by MySQL AB, which is a Swedish company. MySQL is becoming so popular because of many good reasons:

[1] MySQL is released under an open-source license. So you have nothing to pay to use it.

[2] MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.

[3] MySQL uses a standard form of the well-known SQL data language.

[4] MySQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.

[5] MySQL works very quickly and works well even with large data sets.

[6] MySQL is very friendly to PHP, the most appreciated language for web development.

[7] MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4 GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).

[8] MySQL is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

MySQL provides a suite of tools for developing and managing MySQL- based business critical applications on Windows. ”My Sequel” is (as of 2008) the world’s most used open source relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases. The SQL phrase stands for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under

a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation. MySQL is a popular choice of database for use in web applications, and is a central component of the widely used LAMP open source web application software.

### 2.6.4 Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by metaprogramming and metaobjects (magic methods)). Many other paradigms are supported via extensions, including design by contract and logic programming.

Rather than having all of its functionality built into its core, Python was designed to be highly extensible. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach

Most Python implementations (including CPython) include a readevalprint loop (REPL), permitting them to function as a command line interpreter for which the user enters statements sequentially and receives results immediately.

As well as standard desktop integrated development environments, there are Web browser-based IDEs; SageMath (intended for developing science and math-related Python programs); PythonAnywhere, a browser-based IDE and hosting environment; and Canopy IDE, a commercial Python IDE emphasizing scientific computing

### 2.6.5 System Implementation

Implementation is the stage in the project where the theoretical design is turned into working system and is giving confidence on the new system for the users that it will work efficiently and effectively. It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the change over, an evaluation of change over methods. Apart from planning major task of preparing the implementation are education and training of users. The more complex system is being implemented, the more involved will be the system analysis and design effort required just for implementation. An implementation co-ordination committee based on politics of individual organization has been appointed. The implementation process begins with preparing a plan for the implementation of the system. According to this plan, the activities are to be carried out, discussions made regarding the equipment and resources and the additional equipment has to be acquired to implement the new system. The implementation plan includes a description of all activities that must occur to implement the system and to put it into operation. It indicates the personal responsible for the activities and prepares a time chart for implementing the system. The implementation plan consists of the following step:

[1] List all files required for implementation.

[2] Identify all data required to build new files during the implementation.

[3] List all new documents and procedures that go into the new system.

The implemented system has the following features:

[1] Reduced data redundancy.

[2] Ease of use.

[3] Controlled flow.

[4] Simplifies the management activities.

### 2.6.6 Planning and Scheduling

This phase mainly deals with how we can plan and organize different stages for each project. These are different stages for each project. A good software engineer must go

through these phases. Otherwise, chances for failure are very high and also it is difficult to correct. A good programmer must go through these following phases such as system study, data collection, design, coding, testing and implementation.

# Chapter 3

# SYSTEM DESIGN

System design is the process of defining the architecture, components, modules, interfaces and data for a system to satisfy specified requirements. Object-oriented analysis and design methods are the most widely used methods for computer system design. It translates the system requirements into ways of making them operational. The design phase focuses on the detailed implementation of the system recommended in the feasibility study. Design goes through logical and physical stages of development. The characteristics of a well defined system are:

- Security
- Practicality
- Efficiency
- Acceptability
- Flexibility
- Economy
- Reliability
- Simplicity

System design contains Logical Design and Physical Designing. Logical designing describes the structure and characteristics or features, like output, input, files, database and procedures. The physical design follows the logical design, actual software and a

working system. There will be constraints like Hardware, Software, Cost, Time and Interfaces.

## 3.1 Process Design

The term system analysis refers to an orderly-structured process for identifying and solving problems. A system is a combination of resources working together to converts to usable outputs. It is an orderly grouping of interdependent components like together according to a plan achieve a specific objective. The underlying information needs and potential of interactions between various data elements has to capture. Analysis is a detailed study of each of these components in their operation and relationship with the outside of the system. The study of system concept has three implications.

- A system must be designed to achieve a pre-determined objective.
- Inter relationship and interdependent must exit among the components.
- Higher priority is given to the objective of the operations.

## 3.2 Input Design

In the input design, user-oriented inputs are converted into a computer based system format. It also includes determining the record media, method of input, speed of capture and entry on to the screen. Online data entry accepts commands and data through a keyboard. The major approach to input design is the menu and the prompt design. In each alternative, the user's options are predefined. The data ow diagram indicates logical data ow, data stores, source and destination. Input data are collected and organized into a group of similar data. Once identifed input media are selected for processing. In this software, importance is given to develop Graphical User Interface (GUI), which is an important factor in developing effcient and user-friendly software. For inputting user data, attractive forms are designed. User can also select desired options from the menu, which provides all possible facilities. Also the important input format is designed in such a way that accidental errors are avoided. The user has to input only just the minimum data required, which also helps in avoiding the errors that the users may

make. Accurate designing of the input format is very important in developing effcient software. The goal or input design is to make entry as easy, logical and free from errors.

## 3.3 Output Design

In the output design, the emphasis is on producing a hard copy of the infor- mation requested or displaying the output on the CRT screen in a predetermined format. Two of the most output media today are printers and the screen. Most users now access their reports from a hard copy or screen display. Computer's output is the most important and direct source of information to the user, efficient, logical, output design should improve the systems relations with the user and help in decision-making. As the outputs are the most important source of information to the user, better design should improve the system's relation and also should help in decision- making. The output device's capability, print capability, print capability, response time requirements etc should also be considered form design elaborates the way output is presented and layout available for capturing information. It's very helpful to produce the clear, accurate and speedy information for end users.

## 3.4 Functional Requirements

Functional requirements are those that refers to the functionality of the system. i.e., what service it will provide to the user. A use case in software engineering and system engineering is description of a systems behavior as it responds to a request that originates from outside of that system. Use cases describe the interaction between one or more actors and the system itself, represented as a sequence of simple steps. Actors are something Or Someone which exists outside the system (black box) under study, and the take part in a sequence of activities in a dialogue with the system to achieve some goal. Actors may be end users, other systems, or hardware devices.

## 3.5 Non-Functional Requirements

There are requirements that are not functional in nature. Specifically, these are the, constrain system must work within. The nonfunctional requirement include performance requirements. There are two type of performance requirement static and dynamic. Static requirements include a number of terminals supported, number of simultaneous users to be supported, number of file system has to process and their size etc. Dynamic requirements includes execution time behavior of the system such as throughput, response time, expected time for completion of operation etc. The nonfunctional requirements include design constraints, logical database requirements, and standard compliance and so on.

## 3.6 Database Design

Database files are the key source of information in to system. It is the process of designing database files, which are the key source of information to the system. The files should be properly designed and planned for collection, accumulation, editing and retrieving the required information. This database contains tables, where each table corresponds to one particular type of information. Each piece of information in the table is called a field or column. A table also contains records, which is a set of fields. All records in a table have the same set of fields with different information. There are primary key fields that uniquely identify a record in a table. The MySQL has been chosen for developing the relevant databases.

## 3.7 Data Flow Diagram

Larry Constantine first developed the DFD as a way of expressing system requirements in a graphical form that led to module design. A DFD also known as bubble chart has the purpose of clarifying system requirements and identify major transformations that will become programs in system design. So it is the starting point of design phase that functionally decomposes the requirement specification down to the lowest levels of details. A DFD consists of series of bubbles joined by lines. The bubbles represent data flow in the system.

A DFD describes what data flow rather than how they are processed. So it does not depend on negation. The key question we are trying to answer is: what major transformation must occur for input to be correctly transformed to output hardware, software and data structure or file. The creation of the designed system takes place in the implementation phase. Development phase overview, preparing of implementation, computer program development phase report and overview. It also performs activities like writing, testing, debugging and documentation

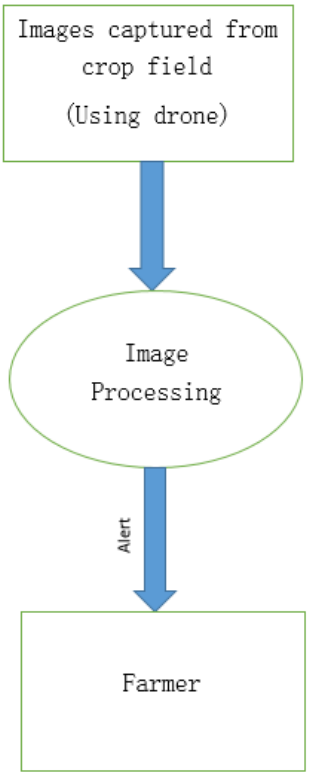


Figure 3.1: Level 0 DFD

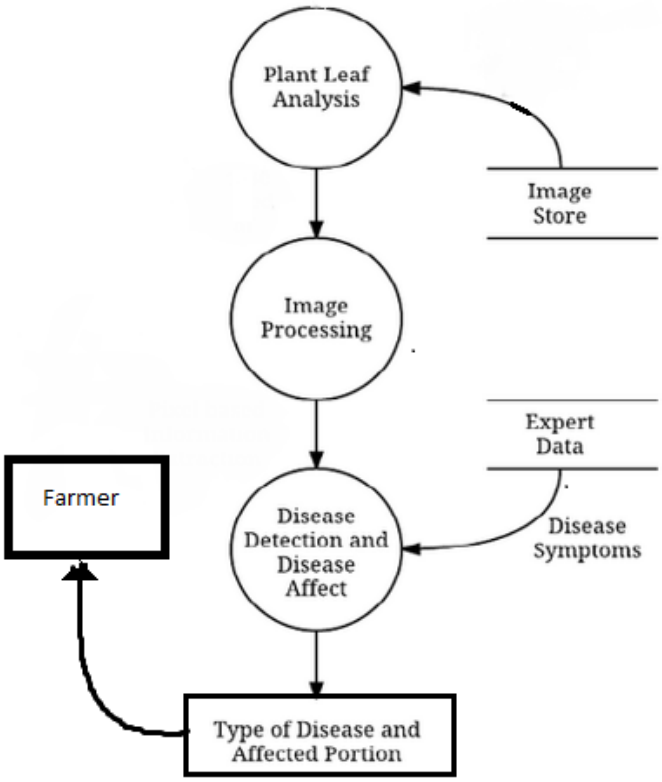


Figure 3.2: Level 1 DFD


Figure 3.3: Level 2 DFD

# Chapter 4

# IMPLEMENTATION

## 4.1 Implementation Approach

A crucial phase in the system life cycle is the successful implementation of the new system design. Implementation is the project when the theoretical design is turned into a working system. If the implementation stage is not properly planned and controlled it can cause chaos.Thus it can be considered to be the most crucial stage in achieving a successful new system.

Normally this stage involves setting up a coordinating committee,which will act as a sounding board for idea ,complaints and problem .The first task is implementation planning.The next task in preparing for implementation stage is training the staff in new skills with which they can use the system.Evaluation and maintenance is done to bring the new system to require standards .the implementation phase comprises of implementation planning.Education and training,system training.

## 4.2 Screenshots

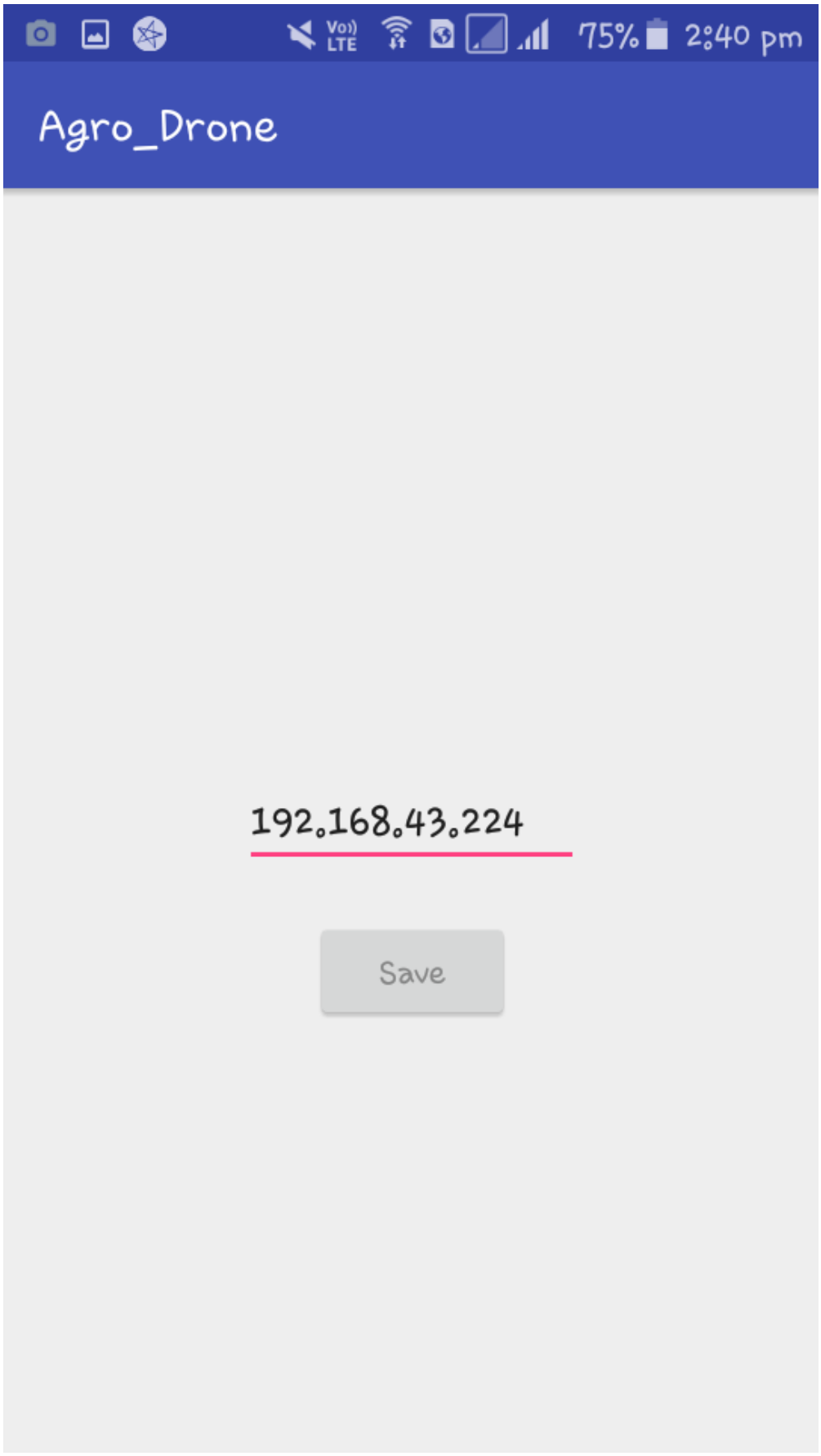


Figure 4.1: App Screen 1


Figure 4.2: App Screen 2


Figure 4.3: Detected Image with Location

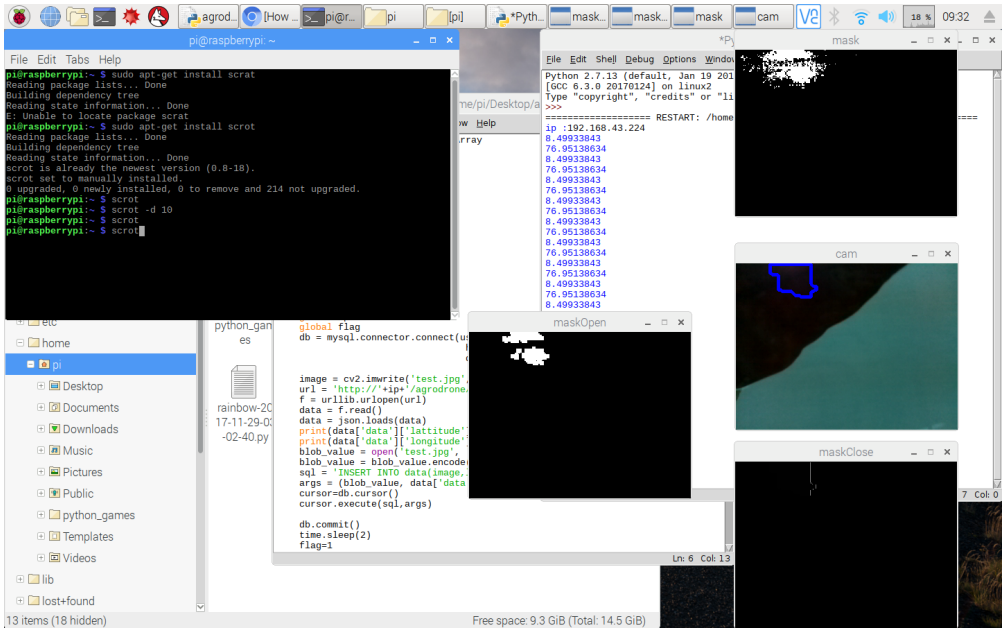


Figure 4.4: RaspberryPi Screen 1

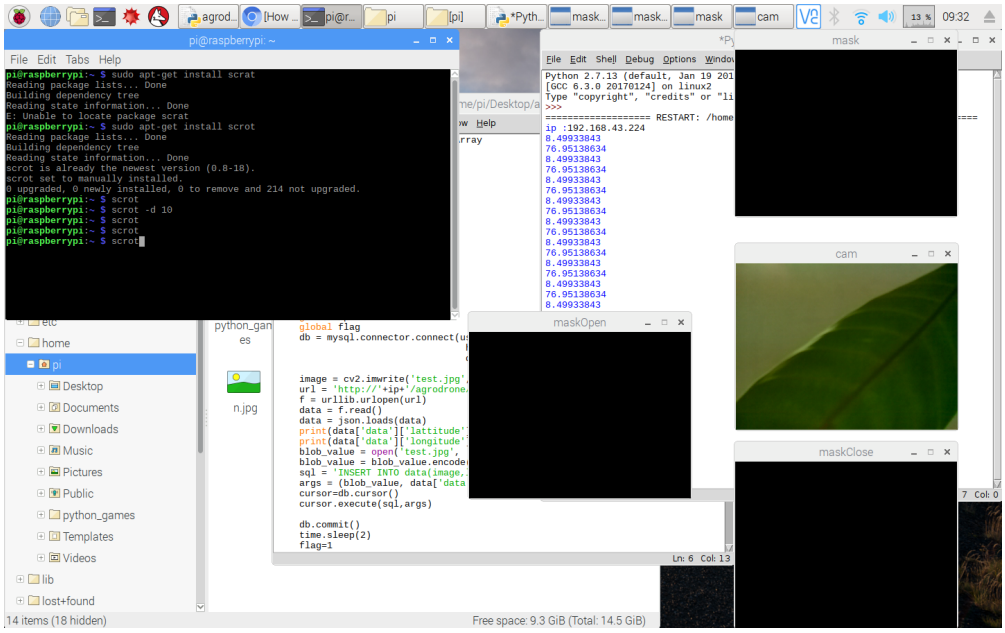


Figure 4.5: RaspberryPi screen 2

# Chapter 5

# TESTING

## 5.1 Unit Testing

Unit testing is performed by selecting each unit of operation for testing. Taking a sample search code for searching a file in database.Tested that it worked successfully. After execution of the uploading code the database and server system is checked to see the upload is there.Unit testing focuses verification effort on the smallest unit of software design that is the module. Unit testing exercises specific path in a modules control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit hence, the name unit testing.

## 5.2 Integration Testing

Integration testing addresses the issues associated with the dual problems of verification and program instruction.After the software has been integrated as a set of higher order tests are conducted. The main objective in this testing process is to take unit tested modules and build a program structure that has been dictated by design. The following are the types of integration testing:

1. Top down integration

2.Bottom up integration TOP DOWN INTEGRATION:

This method is an incremental approach to the construction of program structure. Modules are integrated by moving down through the control hierarchy, beginnings with the

main program module are incorporated into the structure in the either depth first or breadth first manner. BOTTOM UP INTEGRATION: This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated form the bottom up, processing for the modules subordinates to a given level is always available and the need for the stubs is eliminated. The bottom-up integration strategy may be implemented with the following steps:

[1] The low level modules are combined into clusters that perform a specific software sub-function.

[2] A driver i.e. the control program for testing is written to coordinate test case input and output.

[3] The cluster is tested.

[4] Drivers are removed and clusters are combined moving upwards in the program structure. Project aspect: Using integrated test plans prepared in the design phase of the system development, integration test was carried out. All the errors found in the system were corrected for the next testing steps.

## 5.3 Validation Testing

Validation testing is performed in each form which input users data using text box or radio button or selection box. The validation is performed for name, phone no etc.The name field must be character. The mobile number should have 10 digit. The signup is done only when all validation password matching is performed. Validation testing test if all the input data is in correct format and correct input data. The user can go format only when the validation of inputs data is satisfied. Validation test can be defined in many ways, but the simple definition is that is reasonably expected by customer. Software validation is achieved through a series of black box test that demonstrate conformity with requirements. Validation succeeds when the software functions in a manner which user wishes. Project aspect: Proposed system consideration has been tested by using validation testing and found to be working satisfactorily.

## 5.4 System Testing

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic.

As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware system. The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together or between any of the assemblages and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

## 5.5 Regression Testing

Regression testing is a type of software testing which verifies that software, which was previously developed and tested, still performs correctly after it was changed or interfaced with other software. Changes may include software enhancements, patches, configuration changes, etc. During regression testing, new software bugs or regressions may be uncovered. Sometimes a software change impact analysis is performed to determine what areas could be affected by the proposed changes. These areas may include functional and non-functional areas of the system.

The purpose of regression testing is to ensure that changes such as those mentioned above have not introduced new faults. One of the main reasons for regression testing is to determine whether a change in one part of the software affects other parts of the software.

Common methods of regression testing include re-running previously completed tests and checking whether program behavior has changed and whether previously fixed faults have re-emerged. Regression testing can be performed to test a system efficiently by systematically selecting the appropriate minimum set of tests needed to adequately cover a particular change.

## 5.6 Blackbox Testing

Black box testing is the Software testing method which is used to test the software without knowing the internal structure of code or program.

Most likely this testing method is what most of tester actual perform and used the majority in the practical life.

The main purpose of the Black Box is to check whether the software is working as per expected in requirement document and whether it is meeting the user expectations or not.

There are different types of testing used in industry. Each testing type is having its own advantages and disadvantages. So fewer bugs cannot be find using the black box testing or white box testing.

## 5.7 Whitebox Testing

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality (i.e. black-box testing). In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT). White-box testing can be applied at the unit, integration and system levels of the software testing process. Although traditional testers tended to think of white-box testing as being done at the unit level, it is used for integration and system testing more frequently today. It can test paths within a unit, paths between units during integration, and between subsystems during a system level test. Though this method of test design can uncover many errors or problems, it has the potential to miss unimplemented parts of the specification or missing requirements.

White box testing is a method of testing the application at the level of the source code. These test cases are derived through the use of the design techniques mentioned above: control flow testing, data flow testing, branch testing, path testing, statement coverage and decision coverage as well as modified condition decision coverage. White box

testing is the use of these techniques as guidelines to create an error free environment by examining any fragile code. These White box testing techniques are the building blocks of white box testing, whose essence is the careful testing of the application at the source code level to prevent any hidden errors later on. These different techniques exercise every visible path of the source code to minimize errors and create an error free environment.

## 5.8 Test plan

A test plan documents the strategy that will be used to verify and ensure that a product or system meets its design specifications and other requirements. A test plan is usually prepared by or with significant input from test engineers.

Depending on the product and the responsibility of the organization to which the test plan applies, a test plan may include a strategy for one or more of the following:

[1] Design Verification or Compliance test - to be performed during the development or approval stages of the product, typically on a small sample of units.

[2] Manufacturing or Production test - to be performed during preparation or assembly of the product in an ongoing manner for purposes of performance verification and quality control.

[3] Acceptance or Commissioning test - to be performed at the time of delivery or installation of the product.

[4] Service and Repair test - to be performed as required over the service life of the product.

[5] Regression test - to be performed on an existing operational product, to verify that existing functionality didn't get broken when other aspects of the environment are changed (e.g., upgrading the platform on which an existing application runs).

A complex system may have a high level test plan to address the overall requirements and supporting test plans to address the design details of subsystems and components.

# Chapter 6

# CONCLUSION AND FUTURE WORK

The advancement of drone technology has seen many emerging use cases including the expanding use of drones in agriculture. The availability of imaging sensors provide farmers with new opportunities to increase crop yields, minimise crop losses, and thereby maximise their prots. The generated map is key to improving crop performance and reducing costs. It shows exactly which areas of the crop needs further attention. Farmers can spend more time treating their plants, and less time scouting.

# Chapter 7

# REFERENCE


- ”SVM Classifier Based Grape Leaf Disease Detection”,2016 conference on advance in signal processing(CASP),Pune.jun 9-11,2016

- www.dronezon.com

- www.sensefly.com

- www.ncbi.nlm.nih.gov

- https.//phys.org/news/2017-04-drones-early disease crops.html