

CHAPTER 1

INTRODUCTION TO IOT BASED CROP DISEASE DETECTION AND PESTICIDE RECOMMENDATION

Internet of things (IoT) has boomed in the past decade and is continuously disrupting the existing technologies. It connects millions and billions of smart devices to the Internet [1]. This work is about the detection of the disease of the crop and recommending pesticides for crop disease using sensors. The data collected will be uploaded to the cloud and is analysed by the analysis algorithm and detect the presence of the disease to the crop. If disease exist then the alert message is given to the farmer/user and suggest the best pesticides for that particular disease.

Pesticide recommendation for crop disease is a new innovative way of identifying the crop disease using the sensors and alerting the farmer about the diseases and suggest him the best pesticide to be used for that particular disease. The motivation behind developing this project is to reduce the farmer efforts of continuously monitoring the crop health by using the required sensors. The data obtained using the sensors are uploaded to the cloud. Machine learning algorithm is used to analyse this uploaded data to identify the disease. Machine learning provides system the ability to automatically adjust to conditions pertaining to environmental factors and improve from historic data without being explicitly programmed [3]. Traditional model of disease detection uses the complex process like image processing to identify the disease, which is more time consuming and also requires expensive sensors which only takes the good quality images [8]. This model will solve the above mentioned problem by decreasing the complexity in processing and the workload.

1.1 Purpose

The purpose of this project is to develop and propose the architecture to enhance the reliability and accuracy of an IOT network. Traditional IOT device of disease detection

uses some complex functionality like image processing and machine learning to identify the particular disease. The project is aimed at resolving this complex architecture of detection method by use of classification algorithms, color sensor and temperature sensor to help the crop producer to detect the disease and get suggestions of best pesticide for that particular disease by using the modern technology, Internet of Things (IOT).

1.2 Motivation

Agriculture development is providing the assistance to the crop producer by the help of various new technology and the agriculture resources. In this project a working model is developed to detect the disease of the crop using IOT and based on the data collected using sensors the best pesticides will be suggested to the crop producer for better yield.

1.3 Problem statement

To design and develop IOT model that:

1. Takes real time sensor data of the crop field and upload them to the cloud.
2. Analyse the obtained data using machine learning techniques and take accurate decisions on the cloud side. This drastically improves processing speed and reduce the complexity of the device compared to other architectures of similar kind.
3. Detect the crop disease and give the suggestion of pesticides to the crop producer by using Internet of Things.

1.4 Objectives

The primary objective of this project is to identify the disease of the crop from the data collected by the sensor.

1. Collect the crop data by sensors.
2. Upload the sensor data to the cloud.

3. Analyse the cloud data using classification algorithm and detect the disease of crop.
4. Suggestion of best pesticide to farmer if disease exist in the crop.

1.5 Scope and Relevance

The scope of this project is to demonstrate how IOT can be used to detect the disease of any crop by using classification algorithm of machine learning. This is achieved by continues monitoring of the crop by the sensors like color sensor, temperature sensor and humidity sensor. By using all these data collected from the sensors, the proposed system identifies the disease to the crop by use of machine learning model to assist real time decision making.

Traditional model of disease detection uses the complex process like image processing to identify the disease. It is more time consuming and also requires expensive sensors, and it should take good quality images. This model would solve the problem mentioned above by decreasing the complexity in processing by the use of color sensor and also reduces the work load of the model.

1.6 Literature Survey

1. According to the authors of the paper “IOT Based crop-field monitoring”, System is developed to monitor crop-field using sensors (soil moisture, temperature, humidity, light) & automation system. Technique of Eva transpiration is used, which can be used to schedule irrigation. Electromagnetic sensors are used. Web application is developed to analyse the data received & check the threshold values for the parameters and then do the action.
2. The paper published by Mohanrah I, Kirthika Ashokumar, Naren J named “Field monitoring and automation using IOT in Agriculture Domain” [6], This paper proposes the advantage of having ICT in Indian agriculture sector, which shows the path for rural farmers to replace some of the conventional techniques. A comparative study is made between the existing system and the developed systems. The system overcomes limitations of traditional agriculture procedures by utilizing water resources efficiency and also reducing the labor cost.
3. According to the authors from the paper “IOT Based smart crop-field monitoring and automation irrigation system” [2], System implemented on crop-field

- monitoring, was developed by smart phone operating by considering the sensors data via internet. Usage of ATMEL microcontrollers-based GSM operated sensors.
4. According to the authors of white published paper “IOT Based crop-field monitoring and irrigation automation”, Distributed in field sensors-based irrigation system to support site specific irrigation management. Temperature sensor, pH sensors are connected to ATMEL, mobile communication model, stable remote access to field condition & real time control and monitoring of the variable rate irrigation controller.
 5. According to the authors of the paper “IOT Based crop-field Monitoring and Irrigation Automation” [4], A system is developed to monitor the crop-field using sensors and automate the irrigation system. The notification is sent to farmers mobile periodically, the farmer can be able to monitor the field condition from anywhere. This system is 92% more efficient than the conventional approach.

1.7 Methodology and architecture Roadmap

The fig 1.1 represents the project-roadmap design based on the plan after commencement of the project title. The roadmap was adhered throughout the project with less than 15% deviation. And then the project was completed within the time allotted.

The entire project was divided into five phases. The first phase emphasized on the understanding of the problem statement, followed by the literature survey to explore the existing solution and concepts. The objective of this project is finalized. The second phase involves proposing the architecture of the project and detailed design of Unified Modelling (UML). In the third phase, the two models are developed and integrated for implementation. Two application were chosen for implementation. In the fourth phase, the result obtained were discussed. The scope for the future enhancement of the project was also provided. In the fifth and final phase of the project, documentation was done and final presentation and demonstration of the project was held as a part of the project symposium.

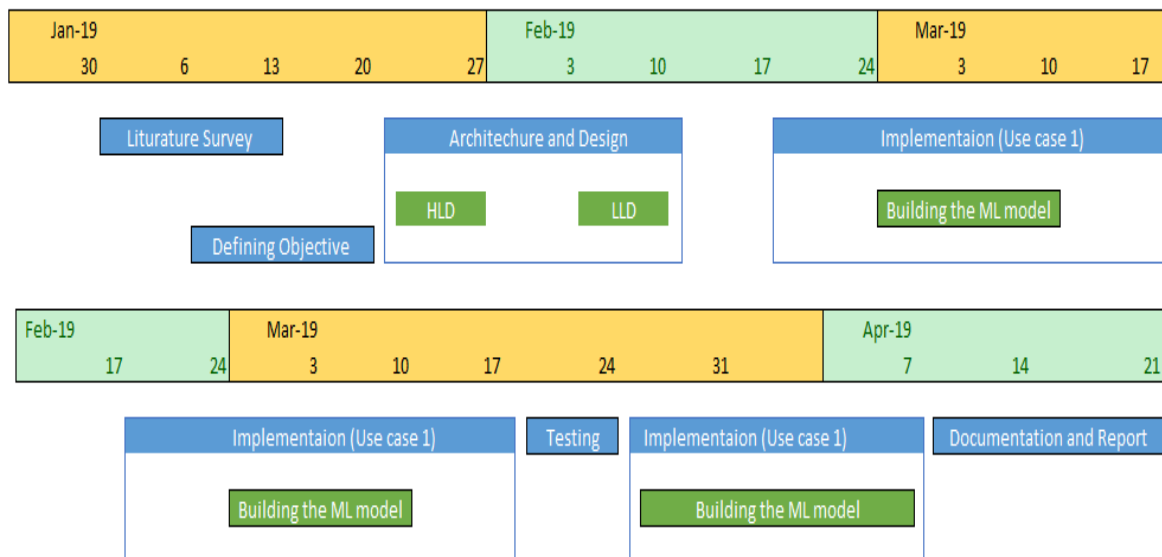


Fig:1.1 Road Map

1.8 Organization of thesis

This dissertation shows the model based on the concept of monitoring the crop field by using the sensors and detecting of the crop disease using the data collected, this reduces the processing speed and the burden to the server compared to the tradition model, where image processing is used to identify the disease. The organization of thesis is as follows.

Chapter 1 introduces our project IOT based crop disease detection and pesticide recommendation in detail. We first start by describing the motivation and purpose of the project followed by the problem statement of the model. The problem statement gives a brief description of the objectives and functionalities that are to be performed by the model. Furthermore, the link between objectives and functionalities is verified to point out the relevance of the project while relation to the real-world applications. We then list out all the journals and papers that have been referred to support foundation of the model. Detailed Methodology and Architectural road map are been discussed to aid the development of the project explained in this dissertation.

Chapter 2 provides a description of all the hardware, software, functional and non-functional requirements of the project. Functional and non-functional requirements

provide a brief description of what is expected from the model being developed. The software and the hardware requirements in the next section describes the essential elements that are used in order to achieve our goal.

Chapter 3 provides a detailed design of the working model. UML diagrams are used, providing the understanding of the model at each level of process. Use case diagrams are used to highlight the flow of processing out between sensing the data and making real time decisions using machine learning model. Further detailed design is discussed of the individual models.

Chapter 4 provides the implementation details are provided including the implementation requirements, tools and features and coding convention used. The various sensors and the other devices are described with its specifications and features.

Chapter 5 provides details on the software testing performed in the project. It gives the test plan for testing the different modules and focuses on three primary functional testing levels – Unit testing, Integration testing and the system testing.

Chapter 6 provides the results obtained are discussed. A comparative analysis is also done against the already existing method. The final chapter of the report concludes the project.

1.9 Summary

The concept of IOT in agriculture is still booming and young to explore this domain exponentially [7]. In essence to the points discussed in this report, disease detection of crop using IOT and sensors will do its work on continues monitoring of the crop this will reduce the farmer effort of monitoring the crop for its disease. By using the machine learning classification algorithms, the identification process of disease is made easy and faster compared to the traditional model where image processing model is used. The proposed model, which incorporates a machine learning model not only improves the scalability but also the scalability and the ease of maintenance.

CHAPTER 2

REQUIREMENT SPECIFICATION OF IOT BASED CROP DISEASE DETECTION AND PESTICIDE RECOMMENDATION

2.1 Functional Requirements

- 1 Collect the crop data by sensors.
- 2 Upload the sensor data to the cloud.
- 3 Analyse the cloud data and detect the disease of crop.
- 4 Suggestion of best pesticide if disease exist in the crop.

2.2 Non-Functional Requirements

- 1 Accessibility.
- 2 Performance.
- 3 Stability.
- 4 Reliability.

2.3 Hardware Requirements

- 1 Aurdino Board.
- 2 Sensors (Temperature, humidity and color sensors).
- 3 WiFi Module.
- 4 Battery.

2.4 Software Requirements

- 1 Arduino IDE
- 2 Hardware programming (aurdino C).
- 3 Server Back-end (python, mysql database).

CHAPTER 3**DESIGN OF IOT BASED CROP DISEASE
DETECTION AND PESTICIDE
RECOMMENDATION****3.1 High Level Design**

The architecture being proposed has two stage of operations that happen in different locations on different devices. The division of the system into stages further eases the process implementation.

3.1.1 System Architecture

The edge device which includes hardware and the sensors like color sensor, temperature sensor and humidity sensors. These sensors and the hardware are responsible for monitoring of the crop and sending the status of the crop to the cloud using the WiFi module. The architecture of this model is depicted in the figure 3.1 which shows how all the sub models of architecture are interconnected.

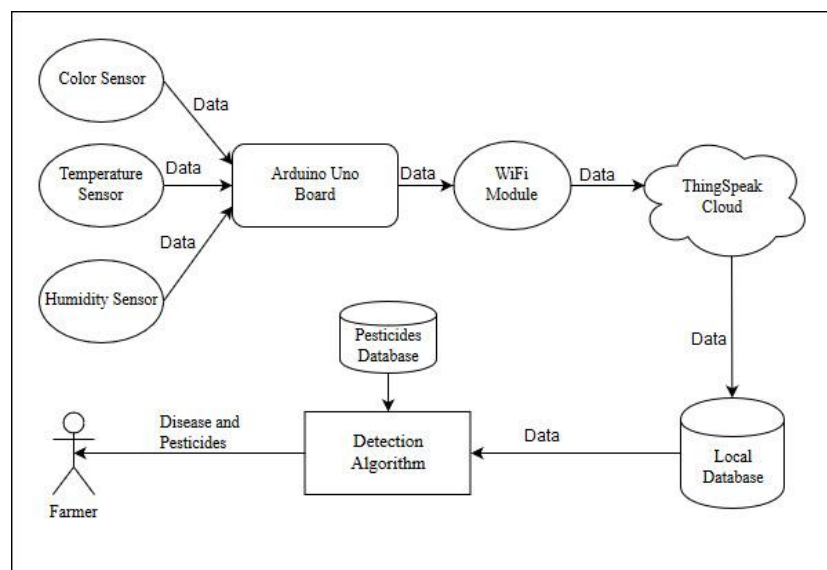


Fig:3.1 Architecture Diagram

In this model, the sensors like color sensor, temperature sensors and humidity sensors are connected to the Arduino Uno, which is intern connected to the ESP8255 WiFi module.

WiFi module collects the data from the Arduino and sends it to the cloud for further operations. The algorithm at the cloud side downloads the data from the cloud. The data obtained from the cloud are analysed by the algorithm if disease exist the farmer will get an alert message and the pesticide suggestions.

1.Design of decision Tree

The major part of the model is to design the algorithm such that it can identify the disease of the crop by using the data collected. The algorithm for identifying the disease is classification algorithm. To make a decision that the crop having disease or not is identified by decision tree algorithm. The figure 3.2 shows the decision tree for identifying the crop disease.

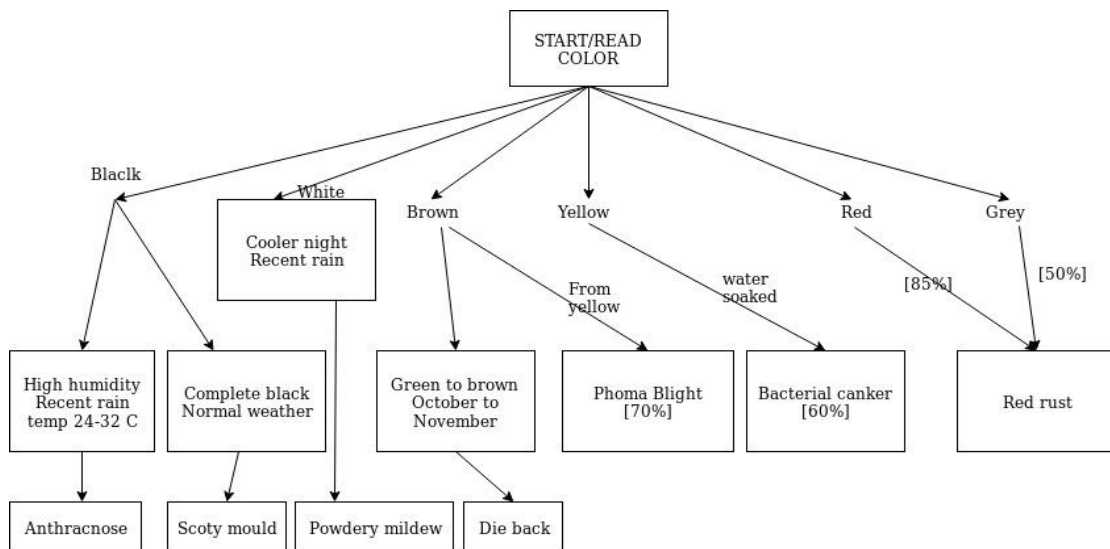


Fig: 3.2 Decision Tree

3.1.2 Sequence Diagram

This UML design represents the general sequence of events, once the architecture is into use on a specific application. It identifies four actors. Crop, sensors, server, detection algorithm and farmer. The sequence is in the order of time and shows which actor is active when. It also shows to an extent, how data is being passed from one subsystem to another.

The figure 3.3 shows the sequence diagram for the crop disease detection that is implemented in our application.

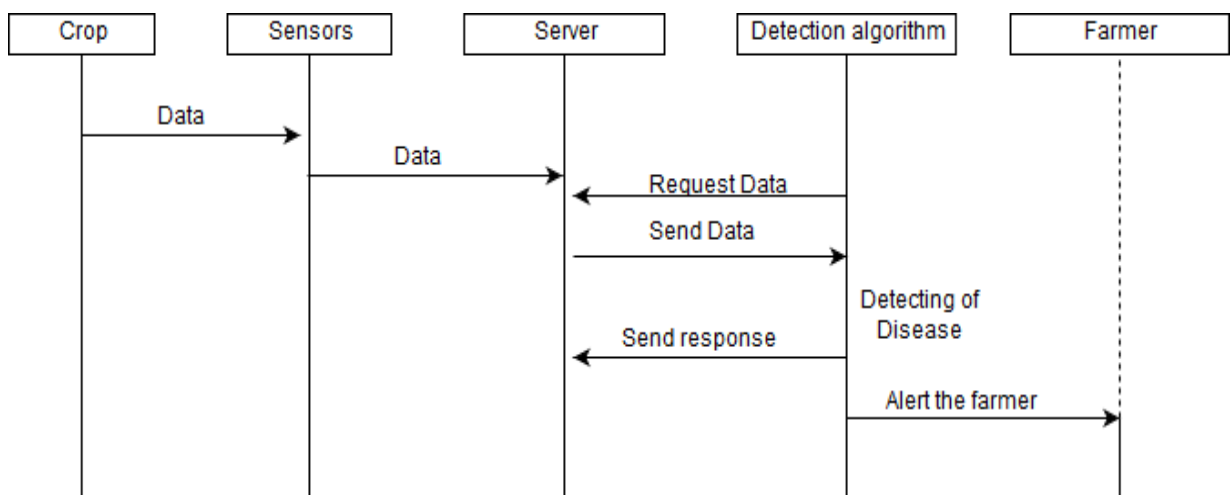


Fig:3.3 Sequence Diagram for crop-disease detection

3.1.3 Activity Diagram

The activity diagram in the figure 3.4 depicts the dynamic aspects of the system. It describes the different activities of the system for our architecture specific to crop-disease detection problem. The below figure 3.4 shows the activity diagram for the crop-disease detection application that implements our architecture.

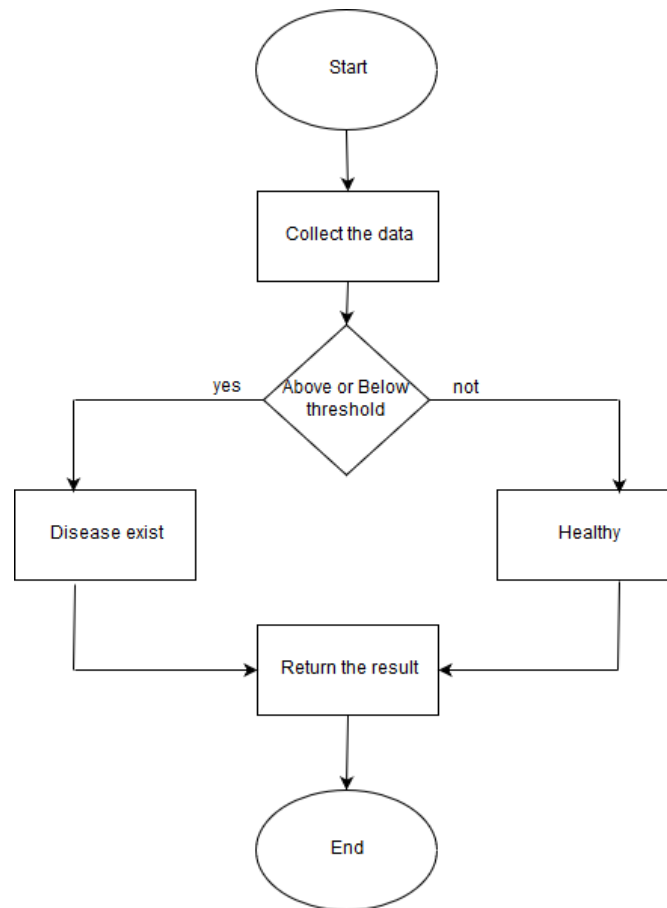


Fig: 3.4 Activity Diagram for crop-disease detection

3.1.4 Class Diagram

The diagram below represents the various classes and their methods for the application of crop-disease detection that uses our architecture. It identifies three distinct classes, Server, Sensors and Detection algorithm. Further the relationship between them are also understood by the diagram. Figure 3.5 below shows the class diagram for the crop-disease detection application.

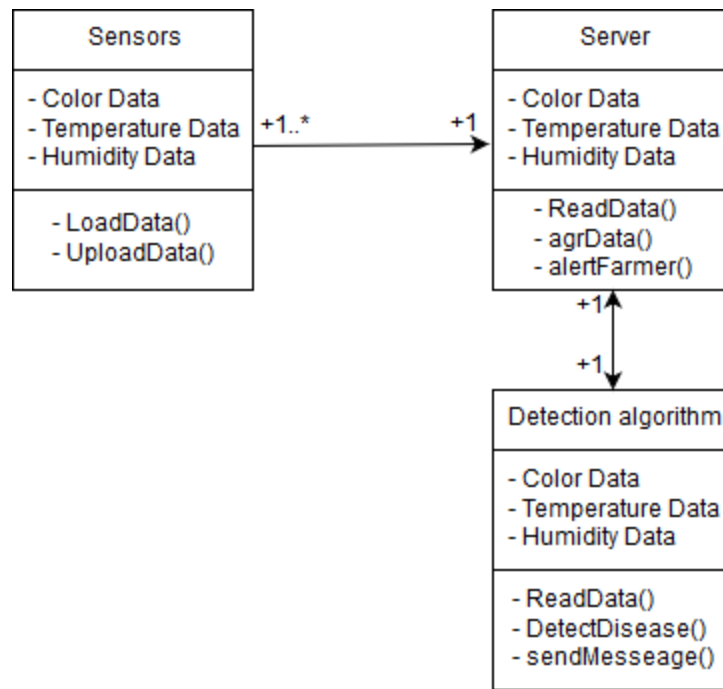


Fig:3.5 Class Diagram for crop-disease detection

3.2 Low Level Design

The architecture being proposed has two stage operation that happens in different physical locations on different device. Hence there are two different modules, which interact with each other. The server module and the sensors module. Cloud which is a server module it collects the data and stores it in the database.

The interaction between the server and hardware module is done by internet. Arduino board collects the data from the sensors and send it to the server. And it also sends the data to the cloud using WiFi module.

3.2.1 Server Module

In the server module, the computation intensive task takes place, which collects the sensor data, and aggregating them. By the use of the aggregated data and the classification algorithm, the server detects the disease of the crop and alerts the farmer. It also suggests the farmer with appropriate pesticides by the alert messages. The server database has three fields, collects the data from color, temperature and humidity sensors respectively.

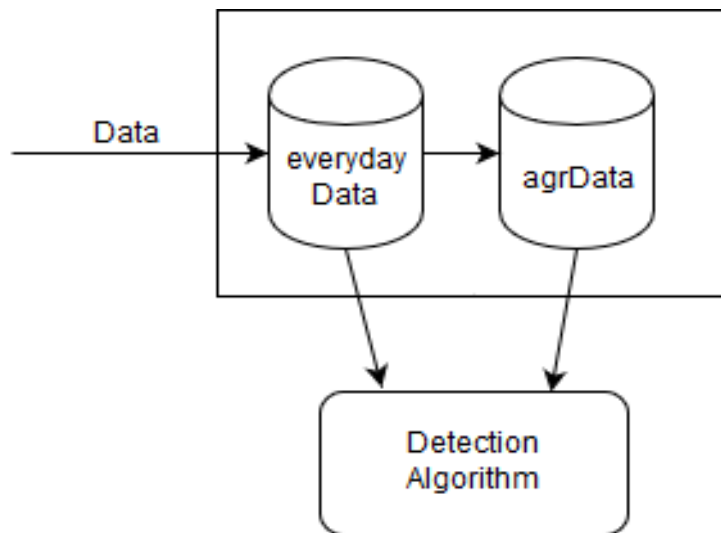


Fig:3.6 Server Module

3.2.2 Hardware Module

The hardware device, being responsible for continuous monitoring of the crop and collecting the data and sending them to the server. The data collected are color of leaf, temperature and humidity data of the crop. And another task is to send the data to the server by using the WiFi module. Arduino board is programmed in such a way that it collects all data from the sensors for every 30 min, and send these data to the cloud continuously. The time constant can be varied based on the requirement.

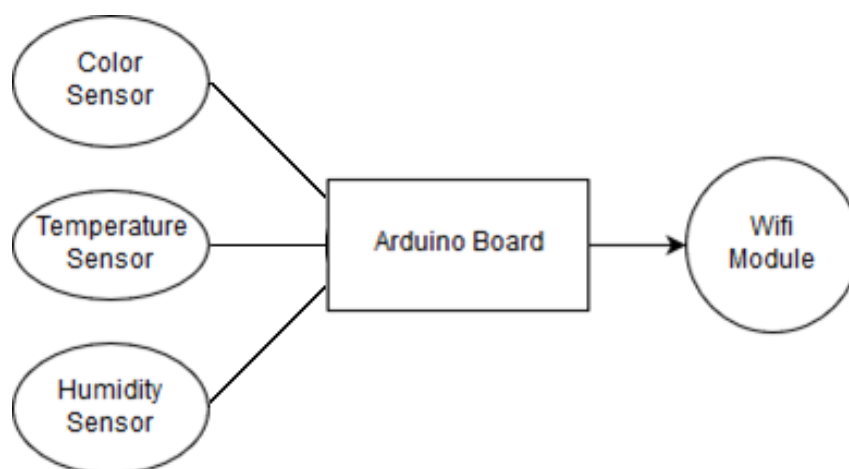


Fig:3.7 Hardware Module

3.2.3 Data Flow Diagram

A Data Flow Diagram (DFD) is a graphical representation of the “flow” of data through an information system. Data flow models describe how the data flows through a sequence of processing steps. DFD is composed of four elements, which are process, data flow, external entity and data store. With data flow diagram, user can easily visualize the operation within the system. DFDs provide the end users an abstract idea regarding the data that is given as input to the system, the effect that the input will ultimately have upon the whole system.

Level 0 - Data Flow Diagram

The level 0 DFD describes general operation of the system. It represents the data read by the system and action taken by the system. The system takes input from the sensors and alerts the farmer. The level 0 DFD is shown in the Figure 3.8

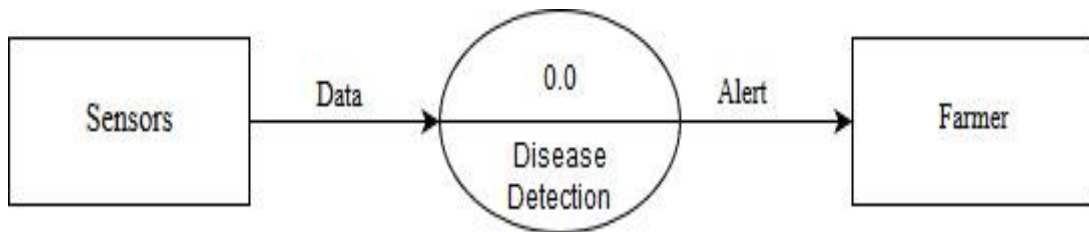


Fig:3.8 Level 0 DFD

Level 1 - Data Flow Diagram

The level 1 DFD describes the system more in detail than the Level 0 DFD. It specifies the main modules involved in the system. This is as shown in the figure 3.9.

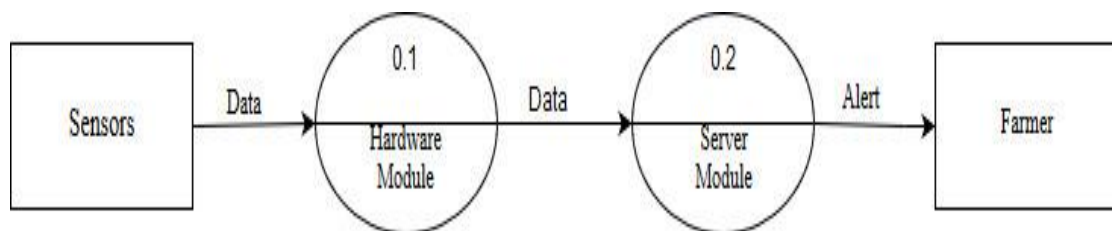


Fig:3.9 Level 1 DFD

The first module cloud which collects the sensor data form the sensors and stores in it. The second module detection algorithm which retrieve the cloud data and analyse it to verify the disease to the crop. If disease exists then it alerts the farmer, by sending alert message and appropriate pesticides.

Level 2 - Data Flow Diagram

The process in level 1 are expanded here. The level 2 DFD for IOT based crop disease detection and pesticide recommendation shown in the figure 3.10.

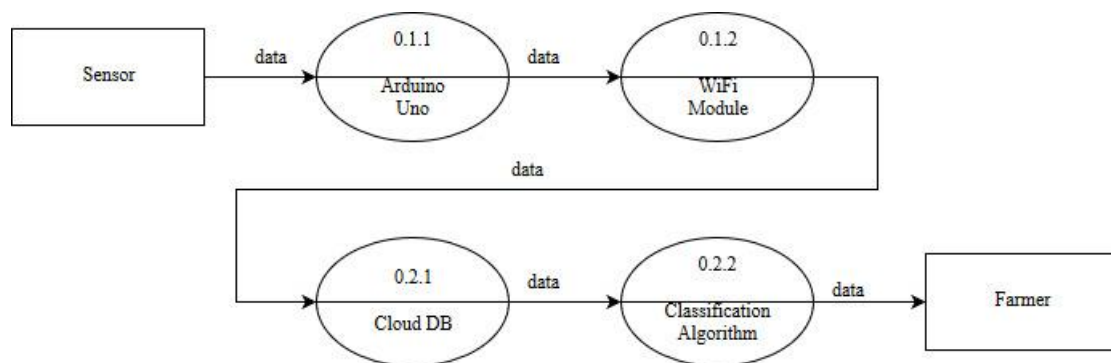


Fig:3.10 Level 2 DFD

Figure 3.10 shows level 2 DFD of this project. First module is split into two different modules. the data collected from the sensors are sent to Arduino Uno board, Arduino Uno board sends the data to cloud using WiFi module. Second module is also split into two different modules. Aggregated data and algorithm, the data obtained from hardware module are aggregated and stored in the database. Algorithm retrieves the aggregated data stored in database and analyse it, to verify it has a disease or not. If the disease exist then the algorithm will alert the farmer and suggest the appropriate pesticides.

CHAPTER 4

IMPLEMENTATION DETAILS OF IOT BASED CROP DISEASE DETECTION AND PESTICIDE RECOMMENDATION

4.1 Implementation Requirements

4.1.1 Hardware Requirements:

1. Arduino UNO:

The UNO is the widely used, documented and robust board in the whole Arduino family. It is based on the ATmega328P microcontroller. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It is a complete package consisting of everything that is needed to support the microcontroller. By simply connecting it to a computer with a USB cable or power it with an AC-to-DC adapter or battery, it can be started.

The figure 4.1 below shows an Arduino Uno board.

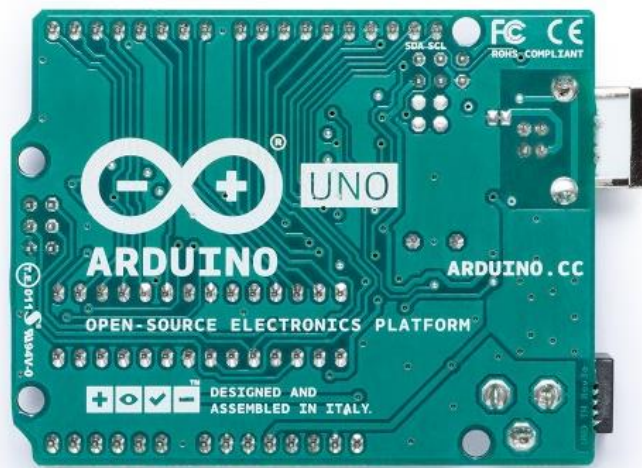


Fig 4.1 Arduino UNO board

2. Jumper Cables

Jumper wires or jumper cables are used for making connections between item on your breadboard and the Arduino header pin.

3. DHT11 Temperature-Humidity Sensor

The DHT11 is a sensor that detects water-vapor by measuring the electrical resistance between two electrodes. The humidity sensing component is a moisture holding substrate with electrodes applied to the surface. When water vapor is absorbed by the substrate, ions are released by the substrate which increases the conductivity between the electrodes. The change in resistance between the two electrodes is proportional to the relative humidity. Higher relative humidity decreases the resistance between the electrodes, while lower relative humidity increases the resistance between the electrodes.

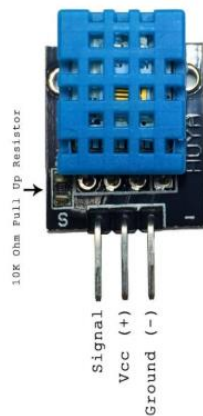


Fig:4.2 DHT 11 sensor

DHT11 measures temperature with a surface mounted NTC temperature sensor (thermistor) built into the unit.

4. ESP8266 Wifi Module

ESP8266 is Wi-Fi enabled system on chip (SoC) module developed by Espressif system. It is mostly used for development of IoT (Internet of Things) embedded applications. It comes with capabilities of,

- 2.4 GHz Wi-Fi (802.11 b/g/n, supporting WPA/WPA2),
- general-purpose input/output (16 GPIO),

- Inter-Integrated Circuit (I²C) serial communication protocol,
- analog-to-digital conversion (10-bit ADC)
- Serial Peripheral Interface (SPI) serial communication protocol,
- I²S (Inter-IC Sound) interfaces with DMA (Direct Memory Access) (sharing pins with GPIO),
- UART (on dedicated pins, plus a transmit-only UART can be enabled on GPIO2), and
- pulse-width modulation (PWM).

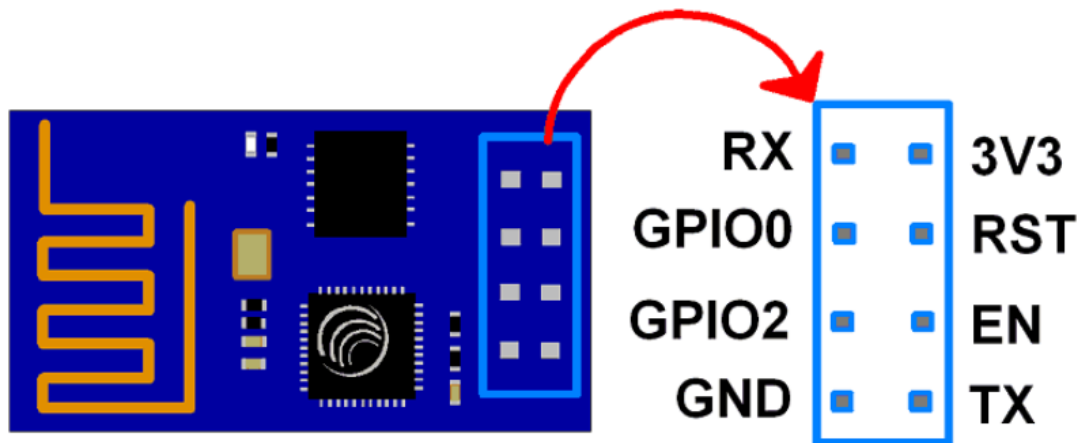


Fig:4.3 ESP8266 WiFi module pin configuration

4.1.2 Software Requirements

1.Arduino IDE (version 1.8.8)

The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards. The Arduino IDE supports the languages C and C++ using special rules of code structuring.

2.Python 3.0

Our project is aimed at running machine learning algorithms in the server and help take data driven decisions, Python is a simple, open-source programming language that

has a high data handling capacity. Also, it has a lot of libraries for easy development of any algorithm. Hence Python was chosen.

3.ThingSpeak cloud

ThingSpeak is an opensource Internet of Things application and provides API to store and retrieve data from ThingSpeak using HTTP protocol over the internet. It can be used to send sensor data privately to the cloud, analyse and visualize your data with MATLAB and also to trigger an action.

The data obtained from the sensors are sent to this cloud using the HTTP protocol over the internet provided by WiFi module. To send data to ThingSpeak and retrieve them, it requires the API key, which will be assigned by ThingSpeak itself. Every time new channel is created. To upload data to ThingSpeak data should send by HTTP request method. Request URL includes the data field and the API private key assigned by ThingSpeak. When HTTP request is made, the data fields are sent in the form of request are stored in the ThingSpeak database.

4.2 Implementation Tool Features

1.Aduino Uno

The major feature of Arduino UNO that separates it from the rest of the versions of Arduino board are:

- It is under constant innovation
- It can also be connected to the computer via a USB cable which does a dual purpose of supplying power and acting as a Serial port to interface the Arduino and the computer.
- It can be powered by a 9V-12V AC to DC adapter
- The ATmega328 chip is newly bought, removed and replaced if damaged which is not possible with other versions of Arduino
- Has a good customer support because of lots of projects are already implemented
- If is a very cost-effective

Table 4.1 Arduino UNO Specifications

Parameters	Values
Microcontroller	ATmega328P
Input Voltage	5V
Input Voltage(recommended)	7-12 V
Input Voltage(limit)	6-20 V
Digital I/O pins	14 (of which 6 provide PWM output)
PWM digital I/O pins	6
DC current per I/O pins	20 Ma
DC current for 3.3 V pin	50 Ma
Flash Memory	32 KB of which 0.5 KB is used by boot-loader
Clock speed	16 MHz
LED_BUILTIN	13

2. DHT11 Sensor

The specifications of the sensor have been summarized in Table 4.2

Tables 4.2 DHT11 Sensor Specifications

Parameter	Value
PCB size	22.0mm X 20.5mm X 1.6mm
Working Voltage	3.3 or 5 V DC
Voltage Operating	3.3 or 5V DC
Measurement range	20-95% RH; 0-50° C
Resolution	8 bit (temperature) ; 8 bit (humidity)
Compatible interfaces	2.54 3-pin interface and 4-pin grove interface

3.ESP8266 Wi-Fi module

The feature of ESP8266 Wi-Fi module are:

- It has Serial UART Interface
- It runs LWIP
- 802.11 bgn
- WiFi Direct (p2p), SOFT-AP
- Built-in TCP/IP
- The AT command is perfect efficient concise
- Support three modes: AP, STA and AP+STA coexistence mode
- UFL Antenna Connector
- Berg strip connectable
- Breadboard Compatible

4.3 Coding Conventions

4.3.1 Naming Conventions

The naming conventions used are described in the tables 4.3 below.

The 4.3 Naming Convention followed in source code

Type	Naming Convention	Examples
Function	Lowercase and uppercase words, CamelCase.	agrData() Check() CheckDisease()
Variable	Lower case letters, words, words separated by underscores and CamelCase.	X, y, z, DiseaseExist, Percentage
Class	All the word started with a capital letter and CamelCase.	DBdatabase

Constant	Uppercase single letter, word. or words, words separated with underscores.	-
Module	Short, lowercase word or words, words separated by underscore.	-
Package	Short, lowercase words, words not separated with underscores.	-

4.3.2 Comments

Documentation of the code is done by using the comments. Single line comments are been used to indicate the sections, action events, variables, functions, expression and descriptions. Multi-line comments have been used to group statements that have been tried out for obtaining the different types of output but are not necessarily required for the results. Further, multi-line comment has been used to maintain the author information as well as the source-code information.

4.4 Summary

The project has two modules – viz, server module and hardware module. The server module aggregates the data and check for the crop disease by the data obtained. The hardware module collects the crop status data by using the sensors it has that is color sensor, temperature sensor and humidity sensor. The hardware module senses these data and send them to the cloud using the WiFi module.

CHAPTER 5**SOFTWARE TESTING OF IOT BASED CROP
DISEASE DETECTION AND PESTICIDE
RECOMMENDATION****5.1 Testing process**

Software testing is done to verify whether the obtained results match the expected results and make sure that the software system is void of any defects. It is done by executing a software component or system component to evaluate one or more properties of interest. It aids in identifying errors and gaps in requirements specified. The testing process is divided into two main types – functional and non-functional testing. We will be focusing on functional testing in our project.

5.2 Functional Testing

In functional testing, the entire system is tested in comparison to requirement specifications. The features are tested by examining the output for the certain input. It ensures that the application satisfies the functional specification. This type of testing is result based and it is never concerned with how the process happens. It simulates actual system usage but doesn't make any system structure assumptions. Then it has different varieties of testing, viz, unit testing, integration testing, system testing, interface testing, regression testing etc.

5.2.1 Unit testing of modules**5.2.1 Module 1-Server Module (Disease Detection Code)**

Unit item	Line no	Input	Expected O/P	Actual O/P	Status
checkForColor()	23-44	Disease color value	Disease exist	Disease exist	Pass

checkForHumidity()	48-63	Disease Humidity	Probability of disease	Probability of disease	Pass
checkForTemperature()	72-89	Diseased Temperature	Probability of disease	Probability of disease	Pass
checkForDisease()	115-152	Diseased values	Disease exist	Disease exist	Pass
Check()	25-56	Call function	Check for disease	Check for disease	Pass

5.2.2 Module 2 – Hardware Module (Arduino code)

Unit item	Input	Expected O/P	Actual O/P	Status
Color sensor	Color	Color value	Color value	Pass
Temperature sensor	Temperature	Temperature value	Temperature value	Pass
Humidity sensor	Humidity	Humidity value	Humidity value	Pass

5.2.2 Integration testing

When different modules are integrated into one single unit and tested, it is referred to as integration testing. In this project we have two modules. This, integration testing is done by integrating the both modules together. Testing the communication between both modules is primarily tested.

5.2.2 Integration Testing Table

Item test	Test description	Test date	Status
Parameters passing	The parameters obtaining in the hardware module using sensors and sending them to the sever.	22-03-2019	Pass
Alert	The disease detected will be informed to the farmer by passing the disease	22-03-2019	Pass

	information to another package which alert the farmer.		
--	--	--	--

5.2.3 System testing

In system testing the complete integrated software is tested. The purpose of this test is to evaluate the systems compliance with the specific requirements.

5.2.3 System Testing Table

Item to test	Test description	Test date	Status
Data gathering	Sensors should gather data (color, temperature, humidity) continuously from the crop.	26-03-2019	Pass
Parameter passing	The parameters value, that is the data gathered must pass to the server using Wi-Fi module.	26-03-2019	Pass
Alert (Action)	After the disease detected the system should indicate or alert the farmer.	26-03-2019	Pass

5.3 Summary

Thus, different function testing was performed on the system to test the system fit for the functional usage. The functionalities that had been specified as a part of the requirement initially, has been met successfully. Since, it is an architecture being proposed, we are not focused on on-functional testing.

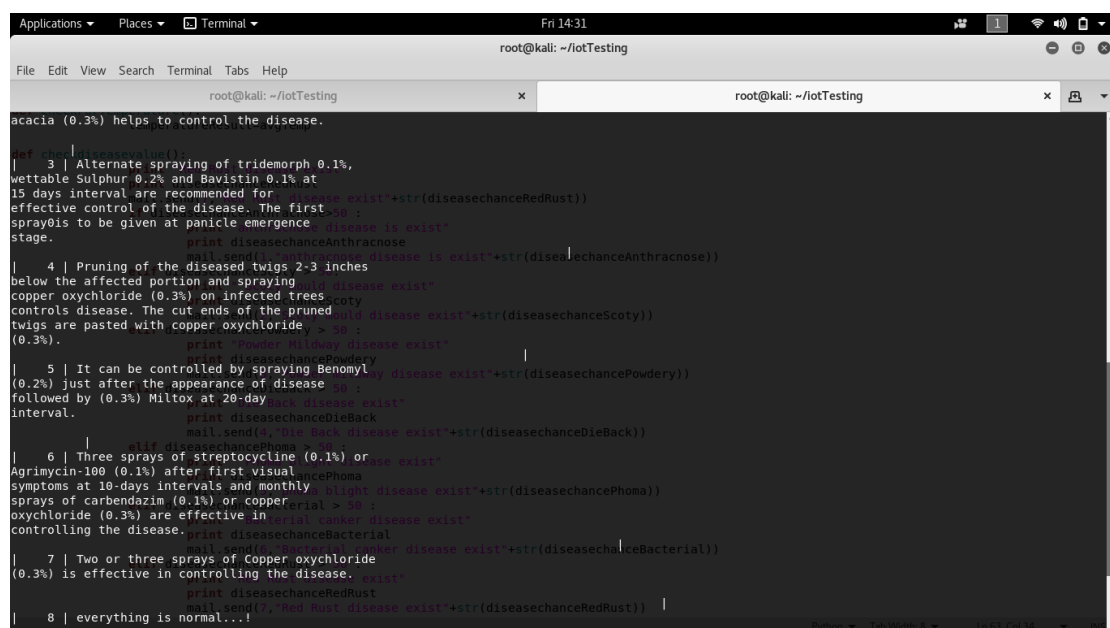
CHAPTER 6

RESULTS AND ANALYSIS

6.1 Results

The primary objective to design the architecture was successfully completed. In order to demonstrate the working model, several use-case were examined, implemented and tested. The server was a HP laptop with 8GB of RAM. But the Server can be hosted in lesser specification also depending of the number of client's model connected to the server. The device with light computational capabilities is been chosen, because it is cost-effective and easier for implementation for many applications. It has been decided to implement the architecture to detect these diseases.

1. Powdery mildew
2. Anthracnose
3. Die back
4. Phoma blight
5. Bacterial canker
6. Red rust

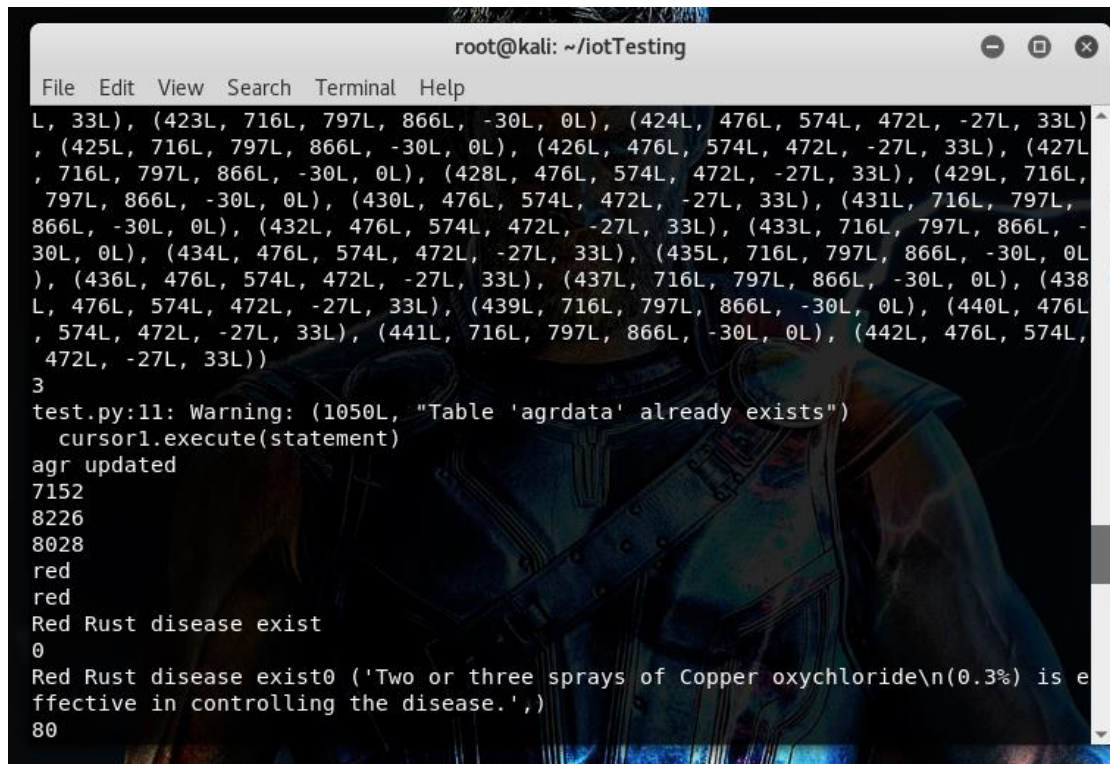


```

acacia (0.3%) helps to control the disease.
def check_diseasevalue():
| 3 | Alternate spraying of tridemorph 0.1%,
wetttable Sulphur 0.2% and Bavistin 0.1% at
15 days interval are recommended for Red Rust disease exist+str(diseasechanceRedRust))
effective control of the disease. The first spray
spray0is to be given at panicle emergence disease is exist"
stage.
print diseasechanceAnthracnose
| 4 | Pruning of the diseased twigs 2-3 inches
below the affected portion and spraying copper
copper oxychloride (0.3%) on infected trees
controls disease. The cut ends of the pruned
twigs are pasted with copper oxychloride
(0.3%).
print "Powdery Mildew disease exist"
print diseasechancePowdery
| 5 | It can be controlled by spraying Benomyl
(0.2%) just after the appearance of disease
followed by (0.3%) Milttox at 20-day
interval.
print diseasechanceDieBack
mail.send('Die Back disease exist'+str(diseasechanceDieBack))
| 6 | Three sprays of streptocycline (0.1%) or
Agrimycin-100 (0.1%) after first visual
symptoms at 10-days intervals and monthly
sprays of carbendazim (0.1%) or copper
oxychloride (0.3%) are effective in
controlling the disease.
print diseasechanceBacterial
| 7 | Two or three sprays of Copper oxychloride
(0.3%) is effective in controlling the disease.
print diseasechanceRedRust
mail.send('Red Rust disease exist'+str(diseasechanceRedRust))
| 8 | everything is normal....!
  
```

Fig: 6.1 Pesticides stored in database.

The figure 6.1 shows the snapshot of database which stores the pesticides to be suggested. The disease mentioned above and their pesticides are stored into the MySQL database. The indexing of the disease is as followed as the numbering of the disease list above. These pesticides are further used by the algorithm to give the appropriate pesticides to the farmer.



```

root@kali: ~/iotTesting
File Edit View Search Terminal Help
L, 33L), (423L, 716L, 797L, 866L, -30L, 0L), (424L, 476L, 574L, 472L, -27L, 33L)
, (425L, 716L, 797L, 866L, -30L, 0L), (426L, 476L, 574L, 472L, -27L, 33L), (427L
, 716L, 797L, 866L, -30L, 0L), (428L, 476L, 574L, 472L, -27L, 33L), (429L, 716L,
797L, 866L, -30L, 0L), (430L, 476L, 574L, 472L, -27L, 33L), (431L, 716L, 797L,
866L, -30L, 0L), (432L, 476L, 574L, 472L, -27L, 33L), (433L, 716L, 797L, 866L, -
30L, 0L), (434L, 476L, 574L, 472L, -27L, 33L), (435L, 716L, 797L, 866L, -30L, 0L
), (436L, 476L, 574L, 472L, -27L, 33L), (437L, 716L, 797L, 866L, -30L, 0L), (438
L, 476L, 574L, 472L, -27L, 33L), (439L, 716L, 797L, 866L, -30L, 0L), (440L, 476L
, 574L, 472L, -27L, 33L), (441L, 716L, 797L, 866L, -30L, 0L), (442L, 476L, 574L,
472L, -27L, 33L))
3
test.py:11: Warning: (1050L, "Table 'agrdata' already exists")
cursor1.execute(statement)
agr updated
7152
8226
8028
red
red
Red Rust disease exist
0
Red Rust disease exist0 ('Two or three sprays of Copper oxychloride\n(0.3%) is e
ffective in controlling the disease.',)
80
  
```

Fig:6.2 Server Detecting the disease

In the server side to verify the result of detecting the disease and to suggest the pesticide. The server program will also display the result values to the terminal with alerting the farmer.

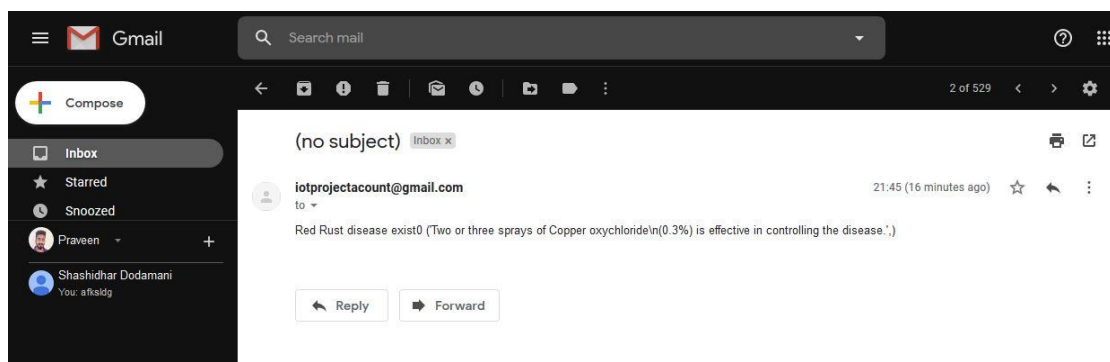


Fig: 6.3 Alerting the Farmer

The result value, that is the disease name and pesticides are suggested to the farmer using the farmer email ID. The figure 6.3 shows the screenshots of how the result message looks like. After getting this message the farmer can further proceed to apply the suggested pesticides in order to avoid the disease caused loss.

6.2 Comparative Analysis

To get the diseased leaf threshold value, analyses is been done. The values from the analysis are store in the database. In the disease detection algorithm, to detect the disease of the crop three parameters are been considered. The parameters are color of the leaf, temperature of the environment and humidity. The values we got from the analysis are provided in below table.

Table 6.1 Disease results

Disease	Color	Humidity range in %	Temperature range In degree Celsius
Powdery mildew	White	35-45	NA
Anthravnose	brown	32-45	NA
Die back	Red dots	NA	< 30
Phoma blight	yellow	35-45	NA
Bacterial canker	Grey	NA	NA
Red rust	Red	NA	NA

The disease can be detected by comparing the obtained values with the stored values of the parameters. Sometimes the disease can be detected only by the one parameter perhaps the greater the number of parameters the more accurate the result will be.

Different diseases have a different color combinations, temperature and humidity variations. For the diseases like Phoma blight, Bacterial canker, Redrust can be identified without considering the temperature parameter. For the diseases like Die back, Bacterial canker, Redrust can be identified without considering the humidity parameter. Different diseases have different parameters to be considered.

6.3 Summary

The architecture is thus dynamic and can be applied for several number of diseases to crop. The major requirement is the parameters to be consider and their threshold values. But the necessary security mechanisms are been implemented for the application specific.

CHAPTER 7**CONCLUSION**

The concept of crop disease detection using IOT is still young and the research to explore this domain is evolving exponentially. Our project proposes a solution to solve problems related to farmer. Continues monitoring of the crop field by the farmer and providing the appropriate pesticides to them is been automated using the IOT in this project. As per the results obtained, the number of diseases to the crop can be identified by the sensors like color sensor, humidity sensor and temperature sensor. The proposed method which incorporates a machine learning model not only improves the scalability but also increases adaptability and maintenance.

The objective of the project was fulfilled by developing the architecture that automates the detection of the disease of a crop and suggest the suitable pesticide to the farmer. The crop status is monitored continuously by the sensors of hardware module and are sent to the server. In the server side the data collected are analyzed and detect the disease to the crop.

7.1 Future Enhancements

1. The number of identification of disease using the apparatus can be increased by more number of parameters.
2. This architecture can also be developed to wide variety of crops.
3. The number of parameters we considering can also be increased to get the accurate results.
4. The integration of server and the sensors paradigms can be improved with respect to data transfer and the security of the data.
5. For applications involving sensitive data, data encryption can also be incorporated for privacy-preservation and security.

REFERENCES

- [1] Rajesh Yakkundimath, Girish Saunshi, Vishwanath Kamatar “Plant disease detection using IOT” International Journal of Engineering Science and Computing, Volume 8 Issue No.9, September 2018.
- [2] Zhang Chuanlei, Zhang Shanwen, Yang Jucheng, Shi Yancui, Chen Jia, “Apple Leaf Disease Identification using Genetic Algorithm and Correlation based feature selection method”, Int J Agric & Biol Eng, Vol. 10 No.2, March, 2017.
- [3] “Powdery mildew of mango” www.plantix.net/plant-disease/en.
- [4] Yun Shi, Zhen Wang, Xianfeng Wang, Shanwen Zhang, “Internet of Things Application to Monitoring Plant Diseases and Insect Pests”, International Conference on Applied Science and Engineering Innovation (ASEI) 2015.
- [5] Sai Vivek, P. Ponnammal, T Akash, Rithvikikaran, “Arduino Based Pest Control Using Real Time Environmental Monitoring Sensors”, International Journal of Advanced.
- [6] Mohanrah I, Kirthika Ashokumar, Naren J “Field monitoring and automation using IOT in Agriculture Domain” 6th International Conference on Advances in Computing & Communications, ICACC 2016, 6-8 September 2016, Cochin, India.
- [7] M. Vinoth, G. Vithiya “Farm field monitoring and irrigation automation using IOT” International Journal of Engineering & Technology, 2018.
- [8] Abhay Dutta, Saban Kumar K.C “IOT based Aquaponics Monitoring System” 1st KEC Conference on Engineering and Technology, 2018.
- [9] Rajesh yakkundimath, Girish saunshi, Vishwanath Kamatar “Plant Disease Detection Using IOT”, International Journal of Engineering and Computing, Volume 8 Issue No9, 2018 IJESC.
- [10] Prof. S. G. Galande, Prof. G. H. Agrawal, Mrs. Londhe Shalaka Rohan, “Internet of Things Implementation for Wireless Monitoring of Agricultural Parameters”, International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE) Volume 4, Issue 8, August 2015.

APPENDIX A

Code Snippets

Hardware code:

Reading sensor data Arduino:

```
void readSensors(void)
{
    temperature = DHT.humidity;
    temperature = (temperature-491.67)*5/9;
    humidity = DHT.temperature;
    digitalWrite(S2,LOW);
    digitalWrite(S3,LOW);
    colorRed = pulseIn(sensorOut, LOW);
    digitalWrite(S2,HIGH);
    digitalWrite(S3,HIGH);
    colorGreen = pulseIn(sensorOut, LOW);
    digitalWrite(S2, LOW);
    digitalWrite(S3, HIGH);
    colorBlue = pulseIn(sensorOut, LOW);
}
```

Sending data to cloud:

```
void GetThingspeakcmd(String getStr)
{
    String cmd = "AT+CIPSEND=";
    cmd += String(getStr.length());
    ESP8266.println(cmd);
    Serial.println(cmd);
    if(ESP8266.find(">"))
    {
        ESP8266.print(getStr);
        Serial.println(getStr);
        delay(500);
        String messageBody = "";
        while (ESP8266.available())
        {
            String line = ESP8266.readStringUntil('\n');
            if (line.length() == 1)
```



```
        {
            messageBody = ESP8266.readStringUntil('\n');
        }
    }
    Serial.print("MessageBody received: ");
    Serial.println(messageBody);
    return messageBody;
}
else
{
    ESP8266.println("AT+CIPCLOSE");
    Serial.println("AT+CIPCLOSE");
}
}
```

Server code:

```
def check():
    connection=MySQLdb.connect("localhost","root","root","IOT")
    cursorX=connection.cursor()
    statementX="SELECT * FROM agrdata ORDER BY id DESC LIMIT 1;"
    cursorX.execute(statementX)
    checkforcolor()
    avgResult=cursorX.fetchall()
    for row in avgResult:
        colorR=row[1]
        colorG=row[2]
        colorB=row[3]
        avgTemp=row[4]
        avgHumidity=avgResult[5]
    connection.commit()
    connection.close()
    checkForColor()
    checkForTemperature()
    checkForHumidity()
    checkfordisease()
```

APPENDIX B

SCREENSHOTS

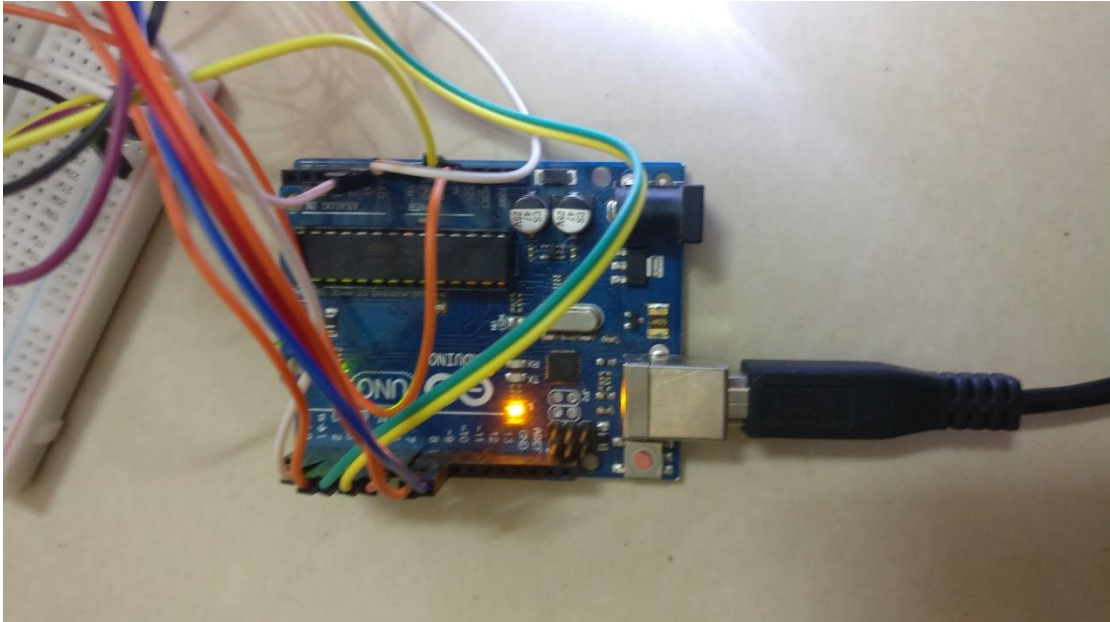


Fig: B1 Arduino Uno Board.

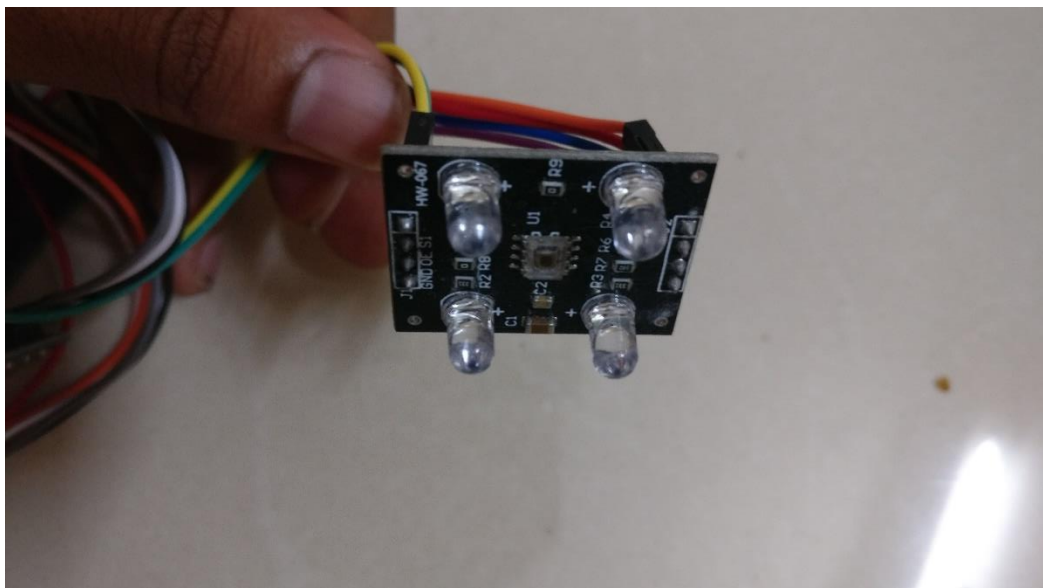


Fig: B2 Color Sensor.

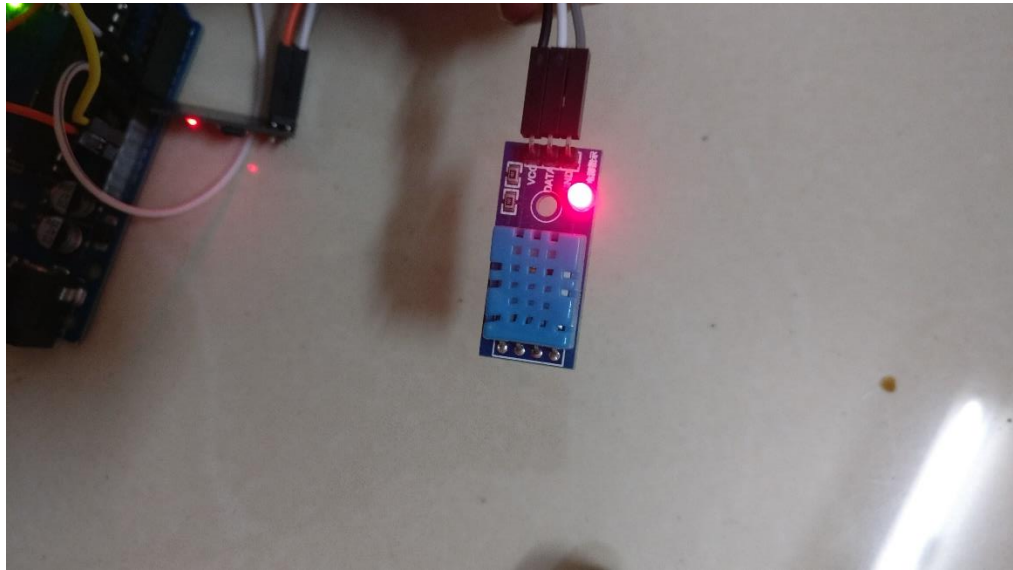


Fig: B3 Temperature and Humidity Sensor.

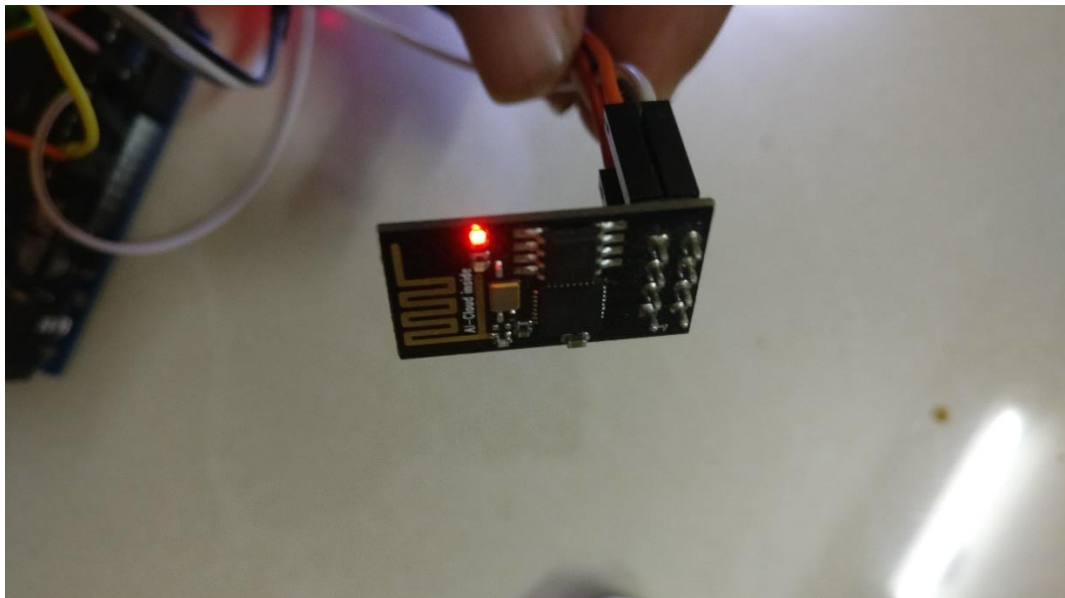


Fig: B4 WiFi Module.

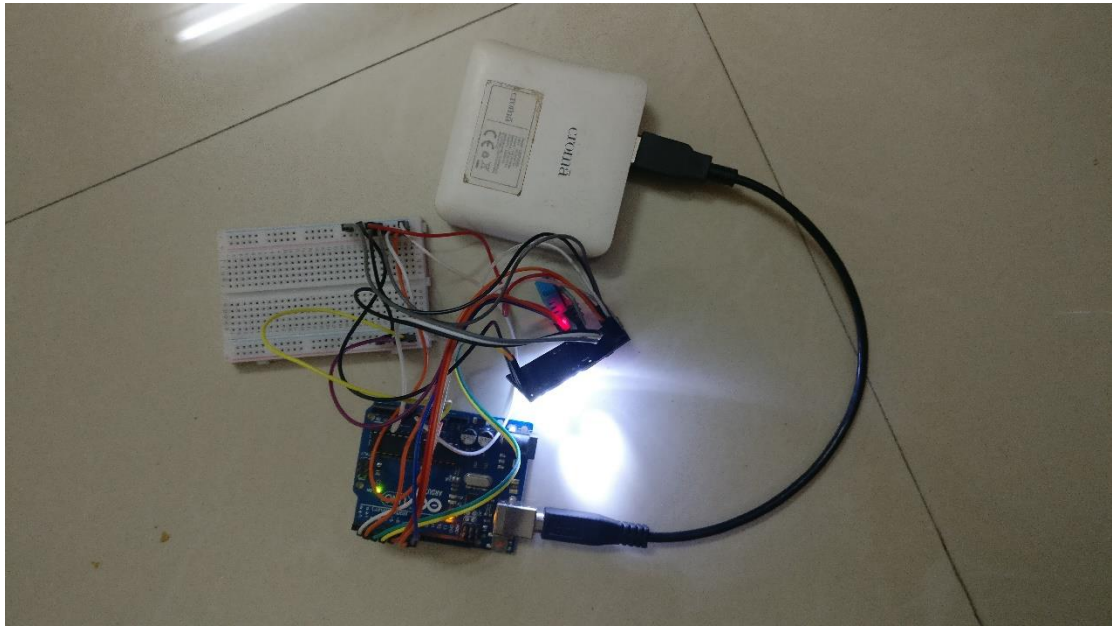


Fig: B5 Hardware Connections.



Fig:B6 Color Sensor Sensing Leaf color.

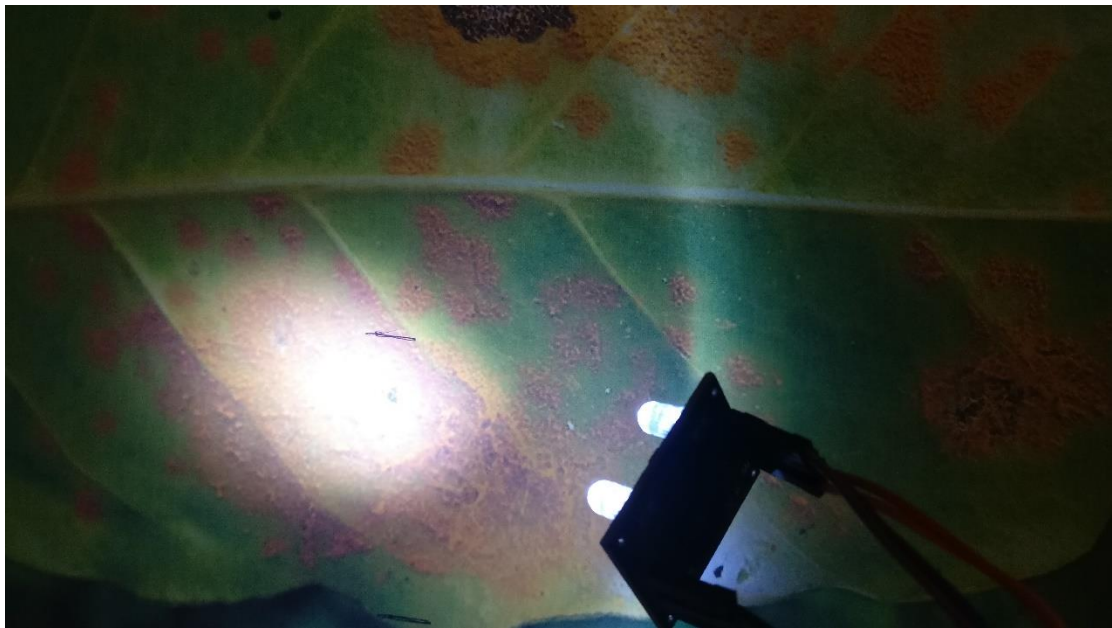


Fig:B7 Color Sensor Sensing Leaf color.

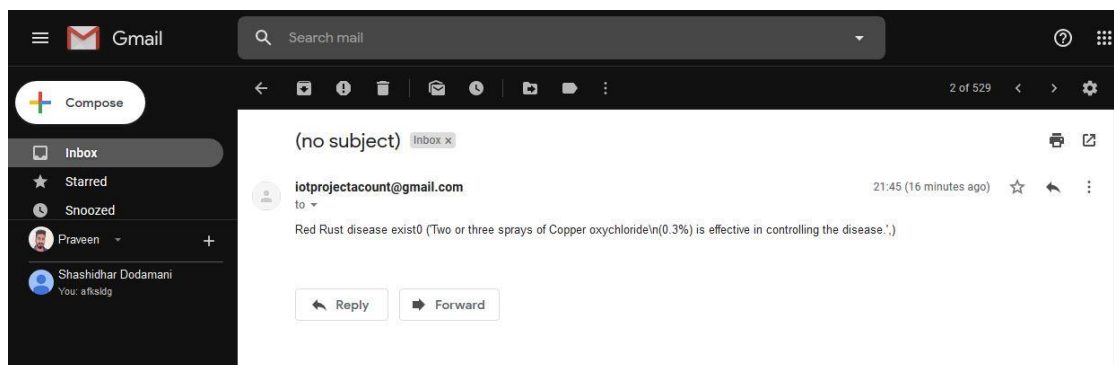


Fig: B8 Alert Message to the Farmer.