JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA

विद्या तत्व ज्योतिसमः

2019-20

B.TECH, CSE

MINOR-2 PROJECT REPORT

INTEGRATING ADVANCE TECHNOLOGIES IN THE FIELD OF AGRICULTURE

SUBMITTED BY:-

Chitrank Mishra (17103103)

Aman Khare (17103048)

Dharmesh Pratap Singh (17103279)

MENTOR:-

Dr. Ashish Mishra

SUBMITTED TO:-

Mrs. Purtee Kohli

Dr. Arpita Jadhav Bhatt

ACKNOWLEDGEMENT

Aman Khare   **17103048**

Chitrank Mishra   **17103103**

Dharmesh Pratap Singh   **17103279**

<u>STUDENT DECLARATION</u>

We hereby declare that the presented report of Even Semester Minor Project titled "**INTEGRATING ADVANCE TECHNOLOGIES IN THE FIELD OF AGRICULTURE**" is an authentic record of work carried out under carried out inder the supervision of **Dr. Ashish Mishra**. We have not submitted this work elsewhere for any other degree or diploma. We are fully responsible to the contents of this report.

_____

Chitrank Mishra, Dharmesh Pratap Singh, Aman Khare

Bachelor of Technology

Department of Computer Science,

Jaypee Institute of Information Technology

SUPERVISOR CERTIFICATE

This is to certify that **Chitrank Mishra (17103103), Aman Khare (17103048), Dharmesh Pratap Singh (17103279),** students of Bachelor of Technology, Computer Science, **Jaypee Institute of Information Technology, Noida** worked under my supervision during the Even Semester 2020 for their Minor Project. I am pleased to state that they worked hard preparing the project and concerned reports. The information and findings in the report are authentic up to my knowledge.

The students possess a good moral character and pleasing personality individually. I wish them every success in life.

_____

Dr. Ashish Mishra

Assistant Professor

Department of Computer Science

Jaypee Institute of Information Technology, Noida

CONTENT LIST

1. INTRODUCTION

Agriculture is the root source of life for a greater part of the Indian population. 58% of the population of our country relies on agriculture. India is planned to increase the crop cultivation by a massive rise of 200% by 2020. These huge numbers and this level of production requires tremendous resources, the highest variety of seeds,   most modern and healthy methods of cultivation, and huge dependence on technological development.

Agriculture is still a growing field for research and development on technological upfront. India ranks second worldwide in farm outputs. As per 2018, agriculture employed 50% of the Indian workforce and contributed 17-18% to the country's GDP[1]. India ranks first in the world with the highest net cropped area followed by the US and China. India is ranked under the world's five largest producers of over 80% of agricultural produce items, including many cash crops such as coffee and cotton, in 2010[2].

Government of India has taken a huge initiative to create awareness among farmers to switch to the modern methods of farming, while they are investing in organizing hundreds of seminars to educate the cultivators about the growing technology, they are also organizing many programs nationwide to attract the brilliant minds to be a part of this initiative. Through such programs, the government is trying to identify and give up new classes of machinery and inventions to support sustainable farming.

Innovation is more important in agriculture than ever before. The industry requires modern tools to redesign the farming infrastructure and more data extraction to recognize patterns in various natural processes related to crop growth. Farm automation is a valuable asset in this domain. Farm automation focuses on making

farms more efficient ad automate the basic processes so that human labor can be utilized in performing tasks that are out of the league of machination.

Although these technologies are fairly young, the industry has seen a remarkable number of companies investing in these methods. But there is a long journey to travel to reach out to the very individuals, that are the farmers of the country. The main goal to drive modern methods to the very roots of the industry is still unfulfilled. Keeping that in mind, it is time to invent, to develop, and to implement thoughts into actions to bring out cheap and efficient methods of farming to start building again at the root level.

The various factors which affect the agriculture in India are explained in the pages further.

1.1 Irrigation

Indian irrigation infrastructure includes a network of major and minor canals from rivers, groundwater well-based systems, tanks, and other rainwater harvesting projects for agricultural activities. Of these, the groundwater system is the largest.

Of the 160 million hectares of cultivated land in India, about 39 million hectare can be irrigated by groundwater wells and an additional 22 million hectares by irrigation canals[3]. About 2/3rd cultivated land in India is dependent on monsoons. India has the world's largest groundwater well equipped irrigation system at 39mha (China with 19 mha is second, USA with 17 mha is third).

Incidents had been recorded that during the time of monsoons, the field which were already irrigated by the farmers, get destroyed due to the rains.

1.2 Temperature

India is a tropical country, having the line of Tropic of Cancer dividing India into two equal parts. The climate of the country is regularly warm throughout the year, demanding a lot of water for the newly sown seeds. Southern India falls under direct sun-rays throughout the year and hence receives the maximum amount of heat-waves. Soil in southern India is fairly suitable for the production of rice, which is also the staple food of the people living in that region.

Little of carelessness in the recording the temperature state can cause a lot of harm to the crops. Rice is very specific about irrigation conditions and it requires a careful observation throughout it's growth stage. Hence, it is very important to pre-calculate the weather conditions for the farmers in those region.

1.3 Inconvenience Faced By Farmers

India has not been a home to the smart technologies arriving to the world. Farmers in India have been using the tradition form of farming since ages. Though, there have been improvements in the type of seeds, type of irrigation, type of pesticides. But there is a huge requirement for the type of technology to automate the farm and control it's machinery from any place.

There are a lot of difficulties faced by a farmer in India. Major ones are listed below:-

1.3.1. Shortage of knowledge to grow crops optimally.

1.3.2. Shortage of energy resources.

1.3.3. Wastage of water by over-irrigation of the field.

1.3.4. Low and ill-patterned availability of Board Electricity.

Indian farmers lack the access to the researches going on in the field of agriculture and the access to the smart technology available in the market due to the huge price of implementation.

2. INSIGHTS OF THE REPORT

Following are a set of insights to the content of the report :-

2.1. Project Phase - I

This phase involved collecting data from various resources available online, from various research papers and surveys taken by various organizations.

2.2. Project Phase - II

This phase involves coupling the data extracted with the hardware and software implementation of the idea. The methodology is defined in the form of separate approaches i.e. hardware and software approaches.

2.3. Project Phase - III

This phase involves a trial to implement the prevailing machine learning technology to enhance the security methods of a farm.

3. PROBLEM STATEMENT

The major problem which Indian agriculture faces is the lack of use of technology to enable a healthy crop production. According to our observation, we have found the three basic causes of the problem being,

- Improper methods of water irrigation
- Unable to know the upcoming adverse weather conditions
- Overexposure to stray farm animals
- Water wastage due to unnecessary flooding of fields

In the view of above observations, we successfully designed a working prototype of a fully automated farm support system. Through it, we successfully achieved the following key points.

- Detecting crop-damaging animals and alerting the caretaker.
- Automate irrigation methods, based on outcome of current weather report and soil moisture content.
- Implementing a control panel to remotely control the machinery on farm.
- Successfully predicting the weather conditions and using them to regulate the quantity of water supplied to the fields.
- Identified the needs of different varieties of crops, including rice, wheat, which are the most staple crop in India, and grass which covers a major vegetation in city areas like parks, gardens, etc.
- Remotely monitor the current state of the farm or garden.

3.1 Problems Identified:

The problems which were found after achieving the above objectives are as follows :

- Current model works on a local server and client system.
- Current model alerts the farmer about the crop-damaging animals using message service through API, which causes the farmer to run to the farm causing inconvenience.
- Current model is not globally available to the hosting on a local platform.
- Current model is limited to a defined number of resources and is not scalable.

3.2 Current objectives:

The objectives which we are focusing now are as follows :

- Converting the platform to support global connectivity.
- Redefining the model to support as many number of resources as possible through the globally hosted platform.
- Implementing a channel to automate as many resources as required and set delays for automatic shut down.
- To counterfeit the inconvenience to the farmer, we are planning to implementing an ultrasonic sound security system, which will act like a smart scarecrow.
- Deploying the database and automation mechanism to the cloud network to allow for easy and hassle-free communication on a large scale.
- Reduce the traffic on the global server by dividing the control system between the local server and the global host.

4. IDEATION

The main idea is to redesign the existing limited system into a more resourceful product. Previous product was able to read moisture data from 2 sensors and could control 4 switches out of which 2 controlled the light, while two automated the irrigation motors. The system was limited and was not being used at the full potential.

The new design to fundamentally represents a control panel, which has various outlets to connect multiple types of machinery and control them remotely using the web platform. New automation mechanism is applied such that it will control the outlets which are marked to support automation. This can be remotely altered. That means, the user is not limited to connecting the irrigation motors and lights. They can also connect the thresher, cutter, winnower and all other machinery to the control panel.

The farmer can manage multiple farms from the same account, just by selecting the farm in the control panel. All the dependent controls and sensor data will be displayed on the interface. Farmers can also change the farm settings remotely from the interface.

To improve the security in the farm and remove the inconvenience for the farmer to arrive the farm every time, any intruder enters. It is known that sounds above a certain frequency are irritating to the receiver but are not harmful. We have designed an ultrasonic beeper system to scare away the intruder, i.e. crop-damaging animal, which will be detected using the cameras planted to detected the animals using CNN model running on the local system.

5. METHODOLOGY

The series of steps which we have used are described as under, along with the methods applied to achieve each one of them

5.1. Identification of intruding animals and alarm network.

A CNN based model will be placed with few cameras on the border of the field, which will be used to monitor the activity happening in the border nearby areas. The CNN model will be used to identify if there is a cow or buffalo in the farm nearby areas, as these animals causes the maximum damage to the crops. So it will differentiate between damage causing animals and non damage causing animals like a dog or cat. The dataset contains about 19000 photos of cats and dogs and 15000 photos of cows and buffalo.

Previously we were sending messages to alert the farmer, but now we have planned to implement an ultrasonic speaker to scare away the animals as the first step of security.

5.2. Cloud based database on MongoDB platform

MongoDB Atlas is the global cloud database service for modern applications. Complete database of the crop details and requirements, user details, user queries and automation system will be deployed on the cloud server and data will be fetched using custom APIs deployed on heroku platform.

5.3. Web server using heroku and Weather API.

This is the computing unit of our system. It will receive user input and corresponding API result as its own inputs and with some predefined constants it will calibrate the result. When the moisture content level falls below the threshold limit, an automated query will be made to fetch the time for running the motor.

The website we are using is *openweathermap.org*. It will provide us with:-

- Temperature
- Precipitation rate
- Humidity

6. PROJECT PHASE - I

6.1. DATA EXTRACTION

We have collected data from various government survey and research reports to make a concise and accurate set of variables to provide optimum irrigation needs to the germinating and growing crops. During our study, we found that there is a specific quantity of water required by the crop to be present in the soil all the time, during different stages of growth, for a healthy crop production.

We collected the volume of water per hectare, required by different crops and scaled them in our database in the terms of height as the land area will remain constant during one cultivation. This volume will vary from month to month and week to week, as per the growth stages of the crop. In the course of this project, we have so far studied the main staple crops, wheat and rice. We can also agree to include grass in our subjects to study as our model can be implement in gardens as well.

Complete database is uploaded to MongoDB Atlas Data Server, backed by AWS Server. MongoDB provides a scalable platform to upload data and access it from anywhere using various APIs. To access the database, we have designed our own system of APIs that readily fetches the data, updates it, uploads the required data, check for the weather situations, automate the hardware timely and format the details accordingly. Complete functionality of the API is described later.

The table in the database shown ahead shows the height of water in millimeters by each plant after a certain interval of time since the day it was sown in the field. Accordingly, the volume of water which should be allowed to enter the fields is calculated and the API returns the amount of time for which the motors should be turned on. The time period is scaled to minutes in correspondence with the scale set on the hardware model.

6.1.1. Study of Wheat:

1. Wheat is an annual plant of *Grammeae* family. It belongs to genus Triticum. Taking a close observation, we were able to identify the common needs of the wheat crop. The graph for the corresponding is shown on the next page[11].
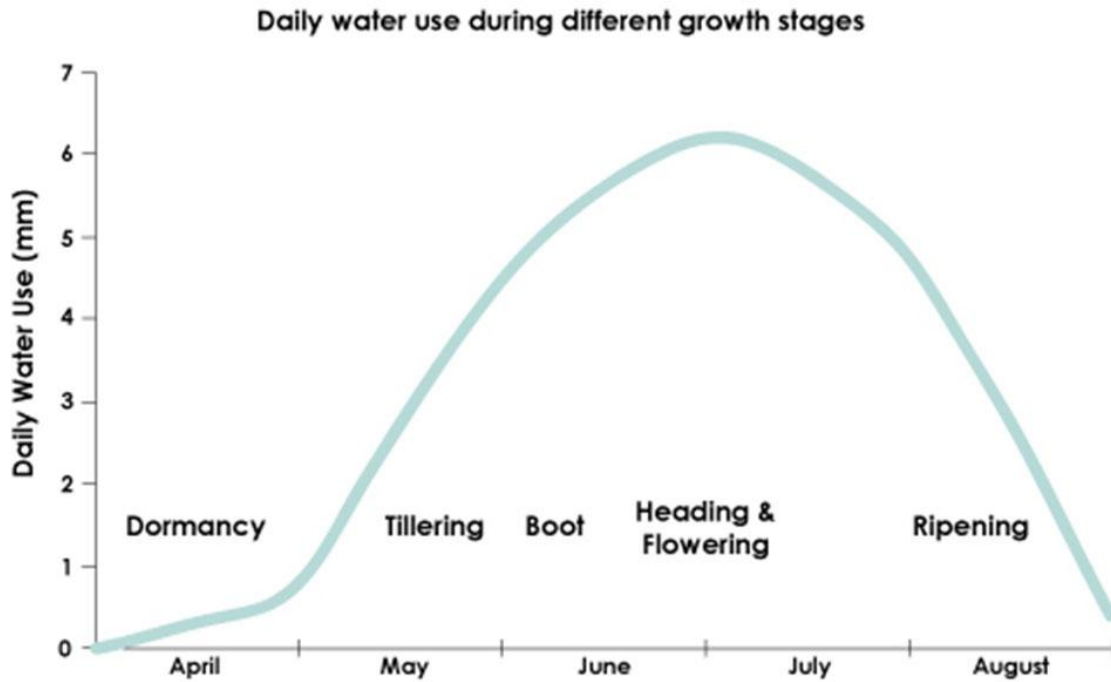

Fig 6.1. Lifetime of Wheat Crop

This graph is mapped into the following table. All the time periods are in reference to the minutes passed after sowing the seed.

month ▲ 1	monthend	min_height	max_height
0	10080	0.5	0.7
10081	20161	0.5	0.7
20161	30240	0.5	0.8
30241	40320	0.6	0.8
40321	50400	0.7	1
50401	60400	1	2
60481	70560	1	2.5
70561	80640	2	3.5
80641	90720	2	4
90721	100800	4	5.5
110881	120960	5	6
120961	131040	5	6
131041	141120	5	7
141121	151200	6	7
151201	161280	6	7
161281	171360	6	7
171361	181400	6.5	8
181440	191520	1	4
191521	201600	1	3.5
201601	211680	2	4
211681	221760	1	4
221761	10000000	1	4

Fig 6.2. Water requirement by wheat in mm per week scaled to minutes.

6.1.2. Study of Rice:

Rice is typically grown in bunded fields that are continuously flooded up to 7−10 days before harvest. Continuous flooding helps ensure sufficient water and control weeds. Lowland rice requires a lot of water. On average, it takes 1,432 liters of water to produce 1 kg of rice in an irrigated lowland production system[8]. Total seasonal water input to rice fields varies from as little as 400 mm in heavy clay soils with shallow groundwater tables to more than 2000 mm in coarse-textured (sandy or loamy) soils with deep groundwater tables[7].

After studying the irrigation practices for rice cultivation from various research papers and agriculture blogs. We were able to quantify the following water requirement table for rice crop.

month	monthend	min_height	max_height
0	10080	4	5
10081	20161	4	5
20161	30240	4	5
30241	40320	4	5
40321	50400	5	5
50401	60400	5	5
60481	70560	5	5
70561	80640	5	5
80641	90720	5	6
90721	100800	5	6
110881	120960	6	6
120961	131040	6	7
131041	141120	6	7
141121	151200	6	6
151201	161280	6	6
161281	171360	5	5
171361	181400	5	5
181440	191520	5	5
191521	201600	4	5
201601	211680	4	5
211681	221760	5	5
221761	10000000	5	5

Fig 6.3. Water requirement by rice in mm per week scaled to minutes.

6.1.3. Study of Grass:

Water plays a key role in lawn and garden care. Nature doesn't always supply enough water, so your landscape may need a little help from you. The signs of a well-cared for yard are a green, weed-free lawn, healthy trees and healthy shrubs. They are an investment in the value of your property and should be protected. The question of how much to water and how often has no single answer. It depends on weather conditions, soil composition and the plants themselves. Grass is particularly susceptible since 85% of its bulk is water[10]. A good drenching once or twice a week is better for your lawn than daily light sprinklings. This requires a long, thorough soaking of the soil, ideally to a depth of about 1 foot but at least 6 to 8 inches. A steady stream of water will run off. An even, sprinkling is best for deep penetration[12].

The following table structures the water requirement by lawn grass in various stages.

month	monthend	min_height	max_height
0	10080	0.5	0.7
10081	20161	0.5	0.7
20161	30240	0.5	0.8
30241	40320	0.6	0.7
40321	50400	0.8	1
50401	60400	0.8	1
60481	70560	0.8	1
70561	80640	0.8	1
80641	90720	1	1.2
90721	100800	1	1.2
110881	120960	1	1.2
120961	131040	1	1.4
131041	141120	1.2	1.4
141121	151200	1.2	1.6
151201	161280	2	2.5
161281	171360	2	2.5
171361	181400	2	2.5
181440	191520	2	2.5
191521	201600	2	2.5
201601	211680	2	2.5
211681	221760	2	2.5
221761	10000000	2.5	3

Fig 6.4. Water requirement by grass in mm per week scaled to minutes.

7. PROJECT PHASE - II

In this phase, we have redesigned the hardware architecture. The scale has been increased from two sensors to multiple sensors which can be modified by changing few variables. It also supports independent automated outlets, which are plug and play type. The user name simply add new machinery into the outlet and control it remotely using the interface.
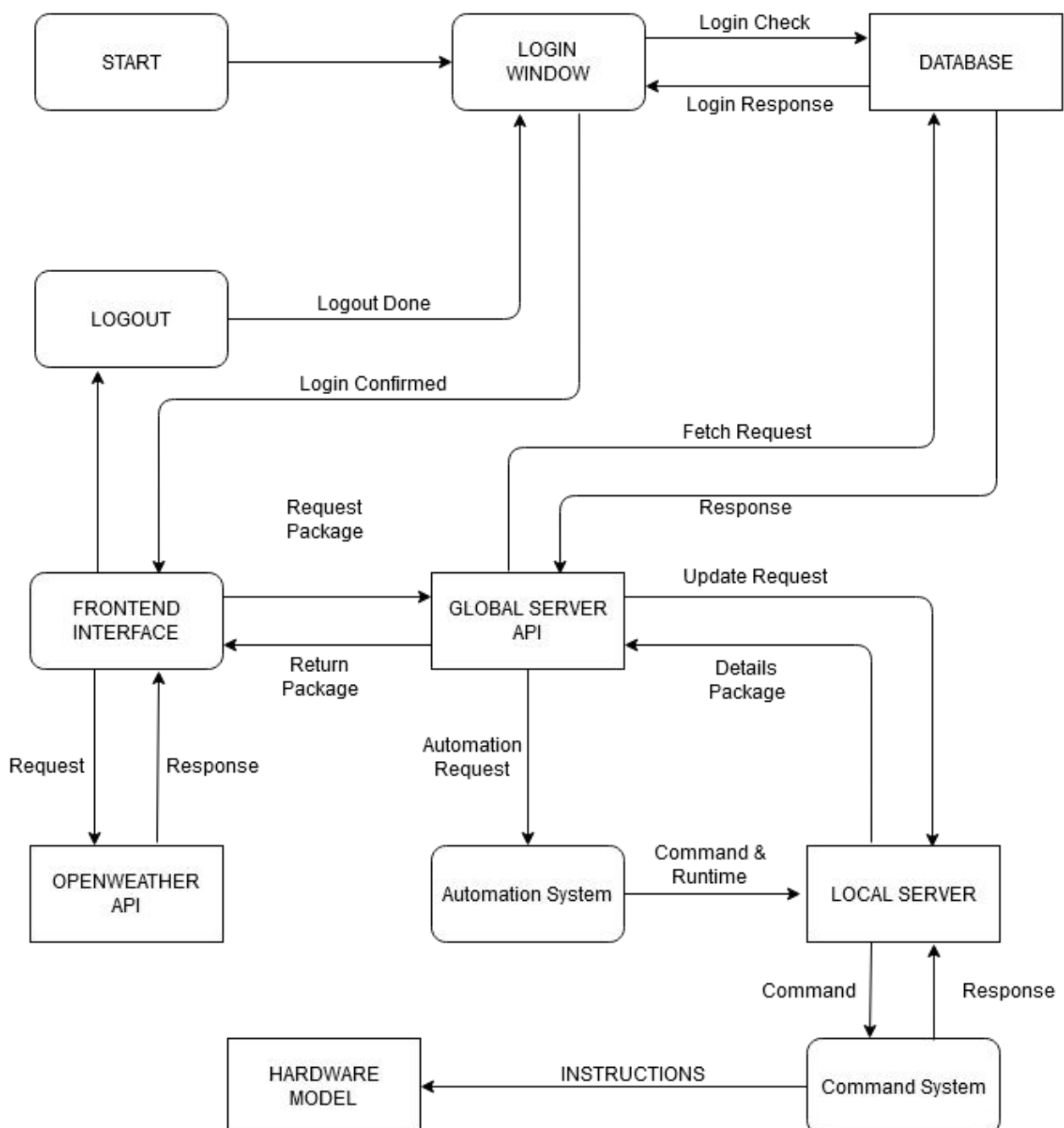

Fig 7.1.1. Flow Diagram

7.1. Hardware Implementation

7.1.1 Devices used :-

- NodeMCU v1.0 : This is a micro-controller motherboard enabled with esp8266 Wi-Fi Soc. MCU stand for micro-controller unit. The firmware used Lua scripting language. In this project, this device works as the core of connections. This devices acts as a client sending requests to the server containing various properties asked by the server.

- 8-bit Relay Module: This device is used to as a switch between various connections, which is operated by pushing down the activation pins to low voltage. It works on the principle of electromagnetic induction in a coil.

- Analog Soil Moisture Reader: It measures the average value of moisture in dry soil or partially humid soil, by calculating the Di-electric constant value between the two probes of the device.

- 12V Water suction pump: It is used to demonstrate how water will be pumped into the fields depending on the outcome of server's calculation of parameters.

- IC7805 and IC7809: IC7805 and IC7809 are used to convert the 12V from the adapter to 5V and 9V respectively. NodeMCU operates at 9V while relay operates at 5V.

- 12V single center shaft motor: FOr demostration purposes

- 12V Servo Motor: For Demostration purposes

- 12V ultrasonic buzzer: It creates a high pitch sound at ultrasonic frequency. This is requires to scare away the crop damaging animals. It works on receiving an analog signal from the microcontroller.

- 12V LED strips

- 12V Adapter

7.1.2 Code Structure:

The complete is available at :

NodeMCU restarts the code every time the power supply is resumed. It begins from the setup(). It assign the default values to the variables as are shown below and then initiates the network connection.

```
int switchCount=6;
int sensorCount=2;
int switchPins[] = { 4,0,2,14,12,13};
int sensorPins[] = { 16, 5};
int autoActive[] = { 0, 0, 0, 0, 0, 0};
long runtime[] = {- 1, -1, -1, -1, -1, -1};
String switchStatus[] = { "OFF", "OFF", "OFF", "OFF", "OFF", "OFF"};
int buzzer = 15;

int buzzerTimeout = 0;
long automationTimeout = 0;
String automationSwitchStatus = "OFF";
```

Fig 7.1.2. Default variables for the NodeMCU system.

It is observable that the code can be altered by changing the value of sensorCount and switchCount. For this device, we have set the sensorCount to 2 and switchCount to 6. Other variables define the state of different machines connected to the outlets. A very prime function of the device is to request the server to provide the time to automate the irrigation system depending on the value of all the sensors ad average moisture content of the farm.

```
if (automationTimeout % (10 * scale) == 0 && !automationFlag && moisture_percent < 50) {
    String apiURL = "https://thawing-coast-20586.herokuapp.com/calculate-runtime/";
    String postData = "VerificationKey=" + act_key;
    for(int i=0;i<sensorCount;i++){
        postData+="&Sensor"+String(i)+"=" + String(getSensorValue(i));
    }
    postData="&AvgMoisture=" + String(calculateMoisture());
    String motor_time = sendRequest(apiURL, postData);
    if (motor_time == "")
        motor_time = "15";
    Serial.println("Motor Time=" + motor_time);
    if (motor_time != "0") {
        turnOnAutomatedSwitches(motor_time.toInt());
    }
    automationTimeout = 1;
}
```

Fig 7.1.3 Automation function on NodeMCU system.

There are other utility function which are describes below with their corresponding functions:

Function Name	Function Description
connectWifi()	Connects to server to the global internet
random()	Returns a random number between a and b
getString()	Extracts the keyword obtained in the request received.
getNumber()	Extracts the number obtained in the request received
getRuntime()	Returns the value of minutes received in the request to turn the motors on.
getIndex()	Returns the index of switch received in the request.
getStatus()	Extracts the status of the switch to obtained in the request from the global API.
sendJSONData()	Send the values of sensors ans switch status as a JSON data packet to the global server
getSensorValue()	Returns the value of the moisture sensor at the index passed in parameter.
setSwitch()	Set the switch value to the value passed in the parameter.
turnOffAll()	Turns off all the machinery in the beginning of the code.
turnOnAutomatedSwitches()	Turn on the switches whose status is set for automation.
turnOffAutomatedSwitches()	Turn off the switches whose status is set for automation.
sendDetails()	Send all the details of the present state of the farm to the global server.
calculateMoisture()	Calculates the average moisture value at the farm.

Table 7.1. Functions on NodeMCU and their description

The request received by the local server are GET requests in format. These request contain keywords which are managed by the below function and accordingly the required functions from the above mentioned functions are called.

```
void manageRequest(String request){
    if (request.indexOf("get-details") != -1) {
        client.flush();
        sendDetails();
    }
    if (request.indexOf("buzzer") != -1) {
        client.println("Buzzer Timeout Set..!");
        buzzerTimeout = 20 * scale;
        digitalWrite(buzzer, HIGH);
    }
    if (request.indexOf("index") != -1) {
        client.flush();
        int index = getIndex(request, "index=");
        String status = getStatus(request, "status=");
        if (status == "ON") {
            long time = getRuntime(request);
            if (time != 0) {
                setSwitch(index, status);
                runtime[index] = time * scale;
            }
        }
        if (status == "OFF") {
            setSwitch(index, status);
            runtime[index] = 0;
        }
        sendDetails();
    }
    if (request.indexOf("automation-control") != -1) {
        String status = getStatus(request, "automation-control=");
        automationSwitchStatus = status;
        sendDetails();
    }
    if (request.indexOf("set-automation") != -1) {
        int index = getIndex(request, "set-automation=");
        String status = getStatus(request, "status=");
        if(status=="OFF")
            autoActive[index] = 0;
        if(status=="ON")
            autoActive[index] = 1;
        sendDetails();
    }
}
```

Fig 7.1.4. Request managing function on NodeMCU system

7.1.3 Circuit Diagram :-

The complete circuit diagram to support the code attached alongside is shown below. The below diagram defines the connection for managing 4 different types of motors, 2 light sources and 3 buzzers. It hosts 2 moisture sensors which sends data alternatively to the ESP8266 NodeMCU module, which acts as the local server.
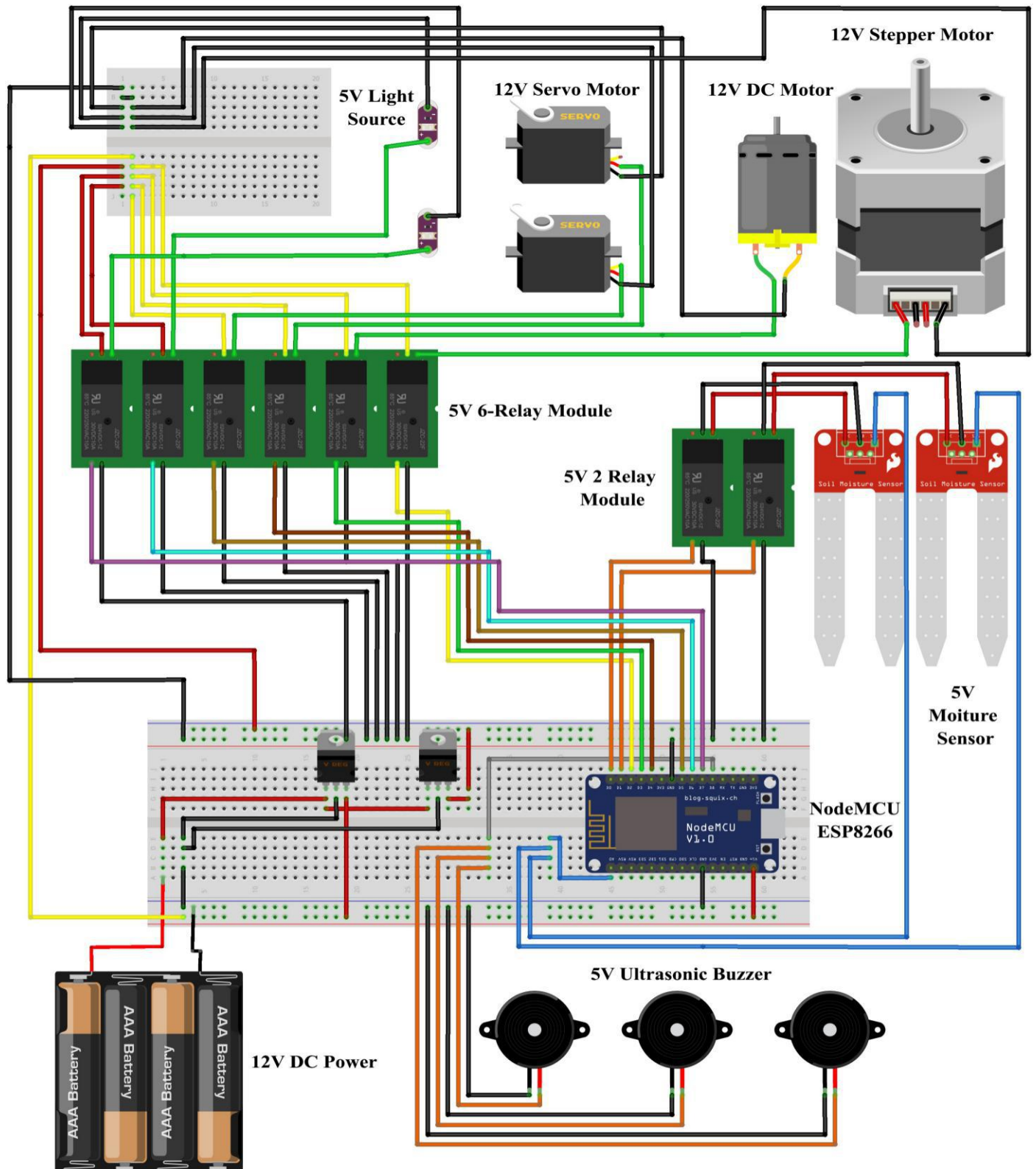

Fig 7.1.5.Circuit diagram for the Hardware system.

7.2. Software Implementation

The software implementation is divided into three sections:
- Frontend Interface
- Global Server on API
- Database

7.2.1 Frontend Interface:
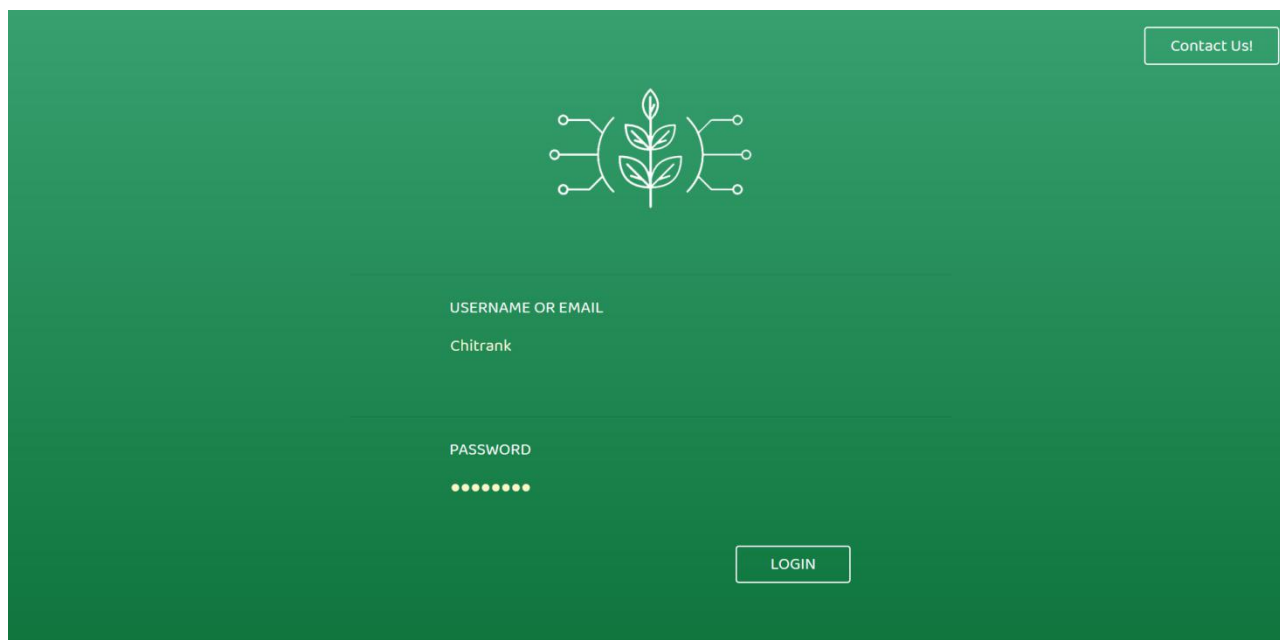
The frontend of the application hosts a dynamic UI which updates according to the needs of the user. The complete website is supported by HTML, CSS and Javascript. Bootstrap is used to make the website mobile-friendly. Javascript performs the backend operations and creates dynamic pages depending upon the user request and data received from the local client. The data from the local client is obtained by sending a request to return a packet of updated data, on user login. The benefit of having a responsive and dynamic UI is that it supports the realtime update of values from the local client.

The website is hosted on Github. The URL to the portal is
https://fallensage0614.github.io/Smart-Irrigation-Project/index.html

The website hosts a login page which logs in using a username/email or password. New users can contact the peers using the contact us button, by which they can raise a query for getting a new connection. The username/email entered is saved in cookies and helps in auto-login by the same IP until cookies expires. Timeout for the cookies is set at 1 day.

For demo login: Username: Chitrank Password: 12345678

Fig 7.2.1 Login page on System Portal.

On successful login, the user is redirected to the homepage. The homepage displays information about the product and a brief introduction. The navigation bar shows tabs to information related to the farm system.

Fig 7.2.2. User home page.

Each tab displays a functionality related to the mechanism. Control panel shows the controls to the farm machinery. Settings provides options to update the user and farm settings. Weather updates does as the name suggests. Contact provides option to raise a query. Each tab is explained in detail with the images.

Control Panel:- This page provides an option to select your farm in case of multiple farms owned by the user. On selecting the desired value. It sends a request to the local client. The sendDetails() function on the local client returns a JSON package to the page which contains the current status of sensors and switches. It displays the corresponding farm switches and sensor values on the page.
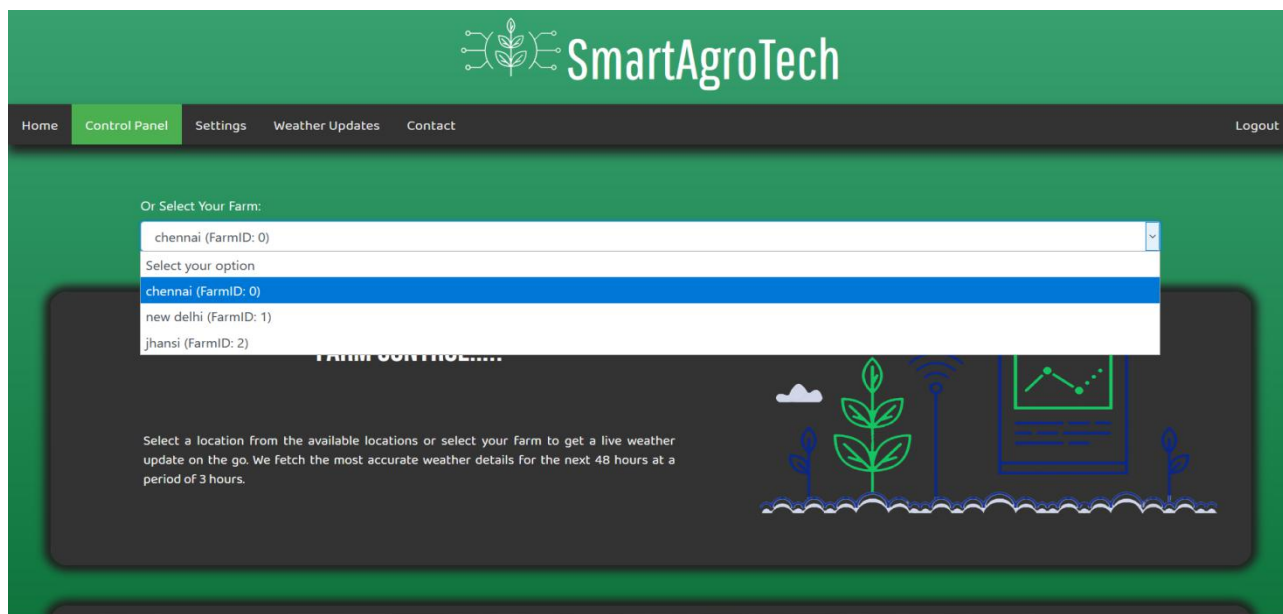

Fig 7.2.3 Control Panel

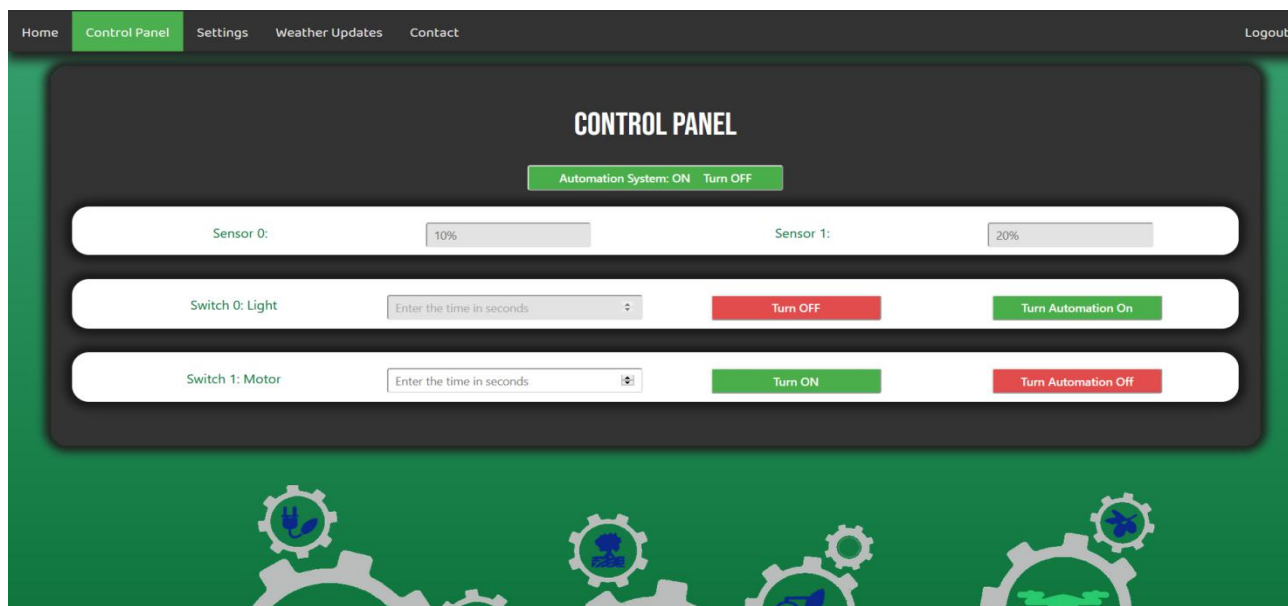For example, suppose we selected Chennai, the corresponding status shall appear in the panel below.

Fig 7.2.4. Control Switches

It is showing the values of the moisture in percent in the vicinity of nth sensor. The toggles for the switches appear in the tablets below. The automation button toggles the automation status for a switch. The switch in Automation: ON state signifies that these devices will operate according to the weather conditions.

Weather Updates:- This page shows the weather updates for the selected area. User can select from a list of farms or a list of cities as per the choice.
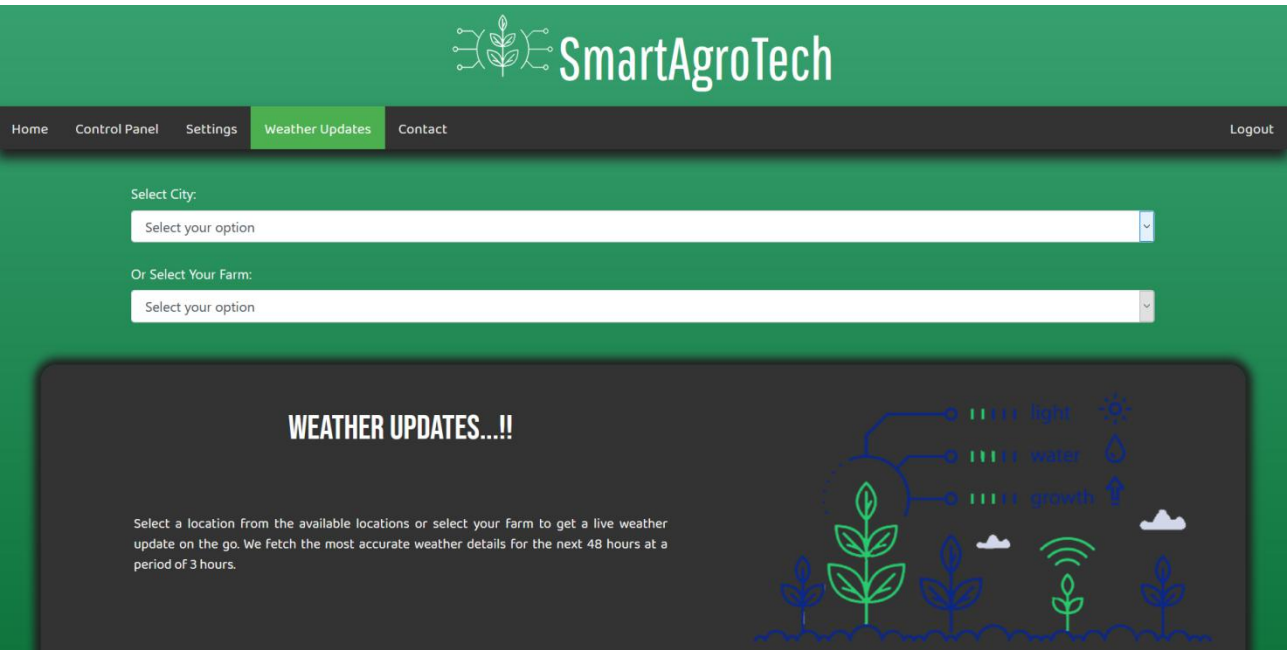

Fig 7.2.5. Weather Updates Page

As per the choice, the API will fetch the weather details from the openweather.org server and list the filtered data in a section below. The weather updates for the next 72 hours from now are displayed, at an interval of 3 hours.
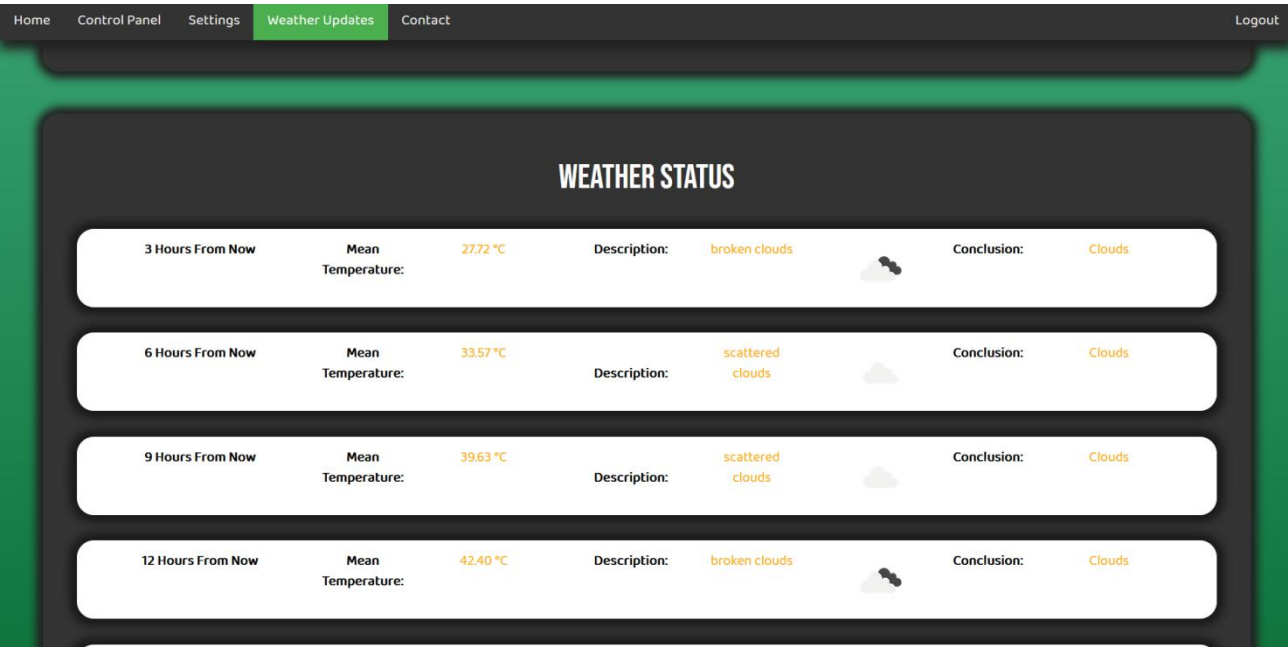

Fig 7.2.6. Weather Updates for a location.

openweather server is used by the automation API to fetch the time after which the rain will fall and accordingly regulate the water to allowed into the fields.

Settings:- This page displays the user data and farm settings. It gives an option to update the username, email, and password.
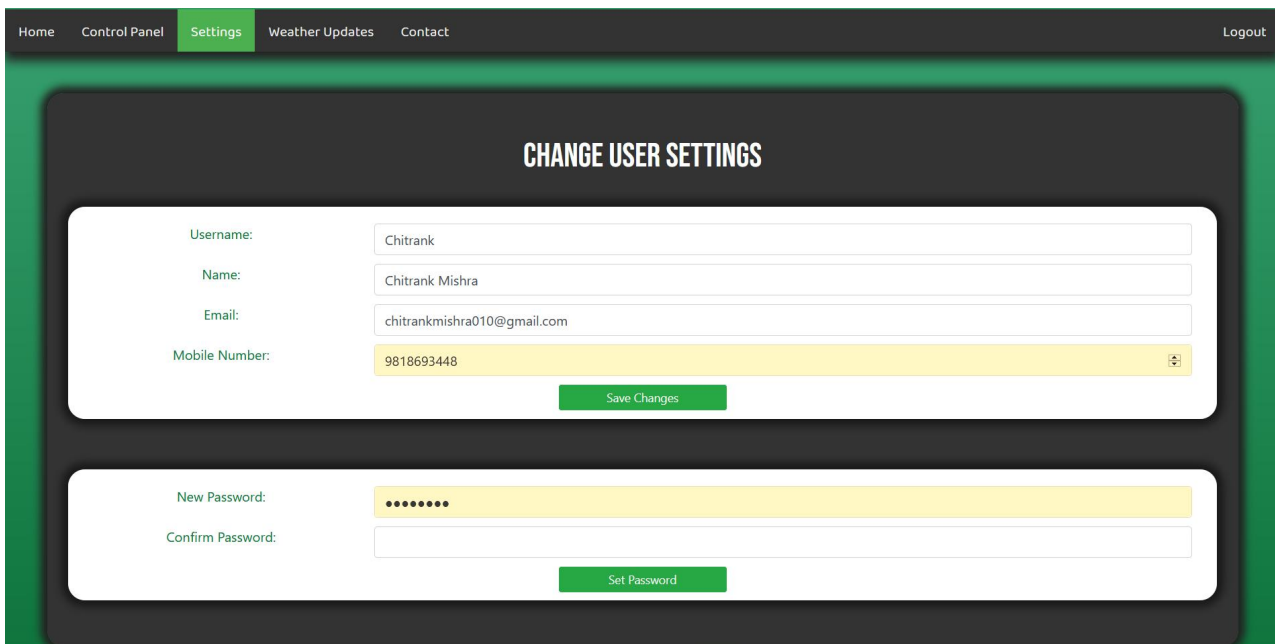

Fig 7.2.7. Settings Updates Page

It also gives and option to select the farm and update the settings for the farm which includes the farm's land area, the total capacity of water supply, the type of crop soil, the date of sowing the seeds.
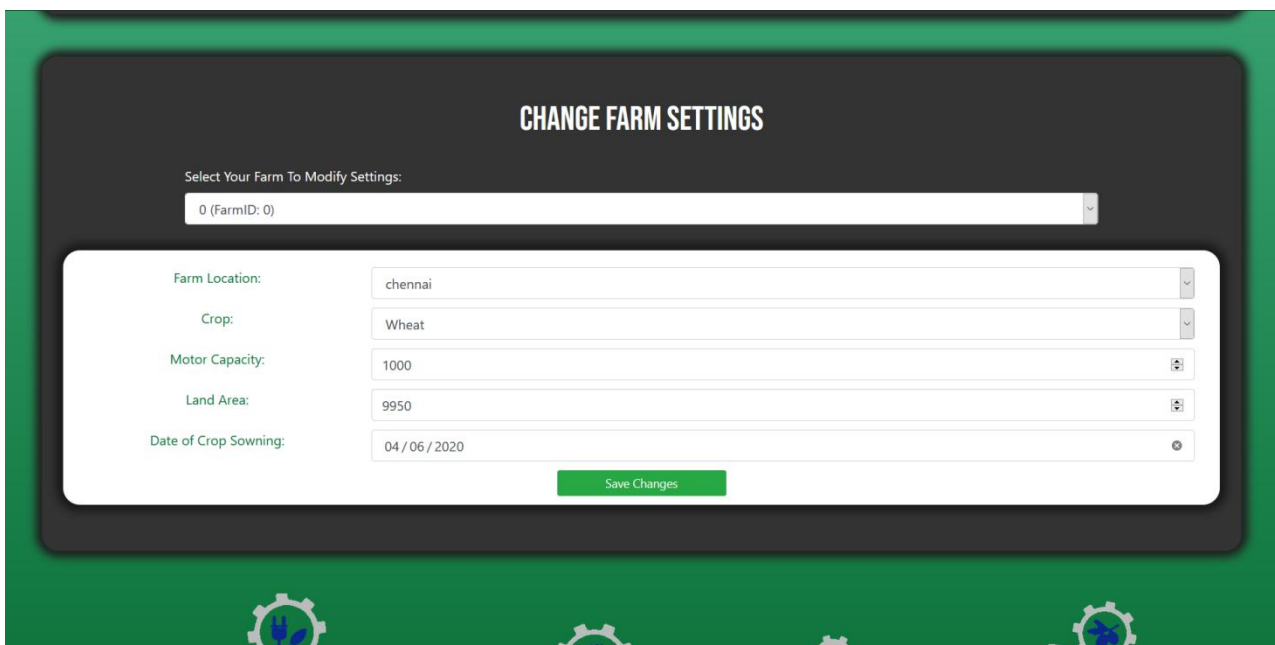

Fig 7.2.8. Section to update Farm Details.

Contact:- This is the last page. It shows the contact details and provides an option to raise any query by the user. The query raised will be saved in the database and responded timely.

7.2.2 Global Server API

As per the objective marked, the server is now globally hosted. It is a Flask based RESTFul application. The python based API is hosted on heroku and is available at https://thawing-coast-20586.herokuapp.com/<corresponding-path>

The API has complete access to the database and is the only way to communicate to the database. Heroku is a secular and reliable server for hosting web applications. The API is responsible for extracting all the information from the database and the local client. It obtains the IP address for a particular client and sends a get request asking for farm status as per the demand of the user.

Certain paths are used to calculate the required runtime values for the farm's irrigation system. The algorithm to calculate the runtime is described below:

- Fetch the IP address using the received FarmID from the database. Append the request to the IP address and call for a get request.

- Filter the data received and calculate the average moisture value of the soil on the farm. If the moisture content is below threshold then proceed.

- Fetch the weather server for the weather details. Filter the packet received and check for the chances of rain in the next 12 hours.

- Calculate the quantity of water needed by the crop in normal case, by the calculating the volume of water by multiplying the land area with the height of the water obtained from the database for the time stage the crop is in.

- Divide the volume by the per minute capacity of the motors installed in the farm. It will give the total time.

- If there are chances of rain in the coming 12 hours, reduces the time period as per the ratio.

- Then adjust the time period to replenish the quantity of water below threshold.

- Return the time period in minutes round off to 2 decimals.

All other functions of the application are managed by this API. It continuously runs on the heroku platform.

7.2.3. Database

The database for the complete application is hosted on MongoDB Atlas server. It can accessed using the API described in the previous section. Python library Pymongo is used extensively in accessing the database. The dashboard for the database is shown below:
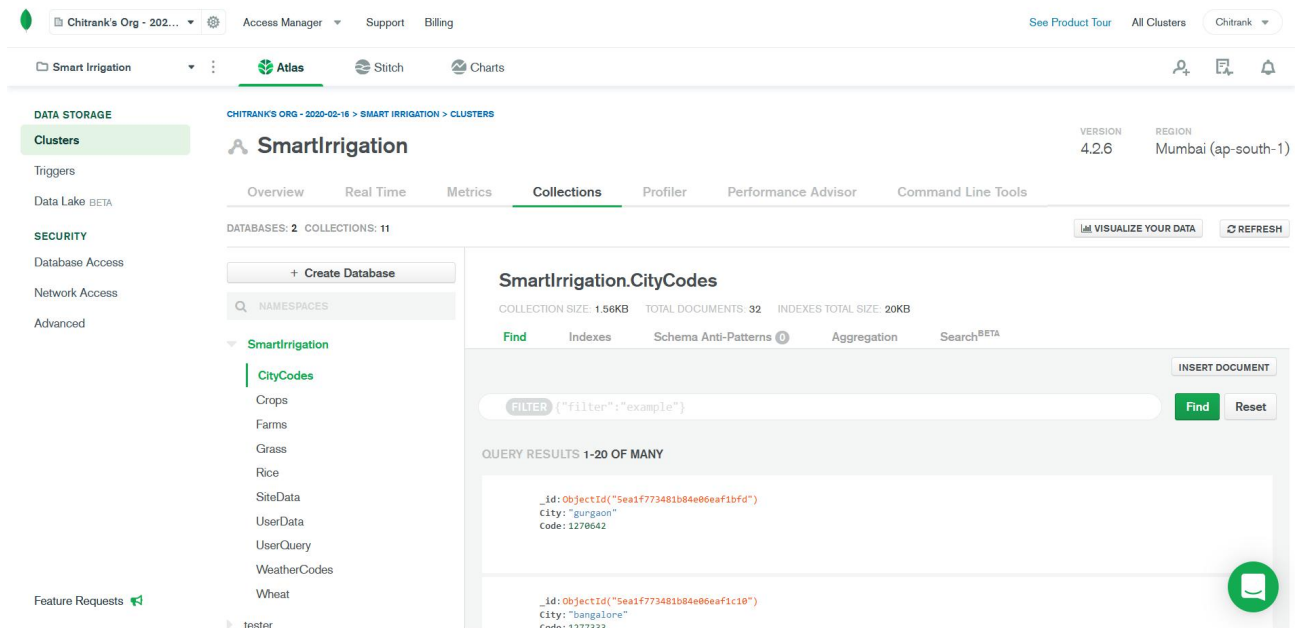

Fig 7.2.9. MongoDB Database Dashboard

The different collections shown above are listed with details about the data hosted.

Collection Name	Collection Details
CityCodes	Codes for all the cities, to fetch weather details.
Crops	All the crops supported by the application
Farms	All the details about each farm registered on the application.
Grass	The irrigation properties for grass fields
Rice	The irrigation properties for rice fields
SiteData	Specific data for the website.
UserData	All the details of all the users who have registered for the application.
UserQuery	User queries registered
WeatherCodes	The weather codes to filter details from the weather site.
Wheat	The irrigation properties for wheat fields

Table 7.1. Collection in the database and description.

8. PROJECT PHASE - III

Model Used: Convolutional Neural Net

CNNs are powerful image processing, artificial intelligence that use deep learning to perform both generative and descriptive tasks, often using machine vision that includes image and video recognition, along with recommendation systems and natural language processing. A CNN uses a system much like a multiple layer perceptron that has been designed for reduced processing requirements. The layers of a CNN consist of an input layer, an output layer and a hidden layer that includes multiple convolution layers, pooling layers, fully connected layers and normalization layers. The removal of limitations and increase in efficiency for image processing results in a system that is far more effective, simpler to trains limited for image processing and natural language processing.
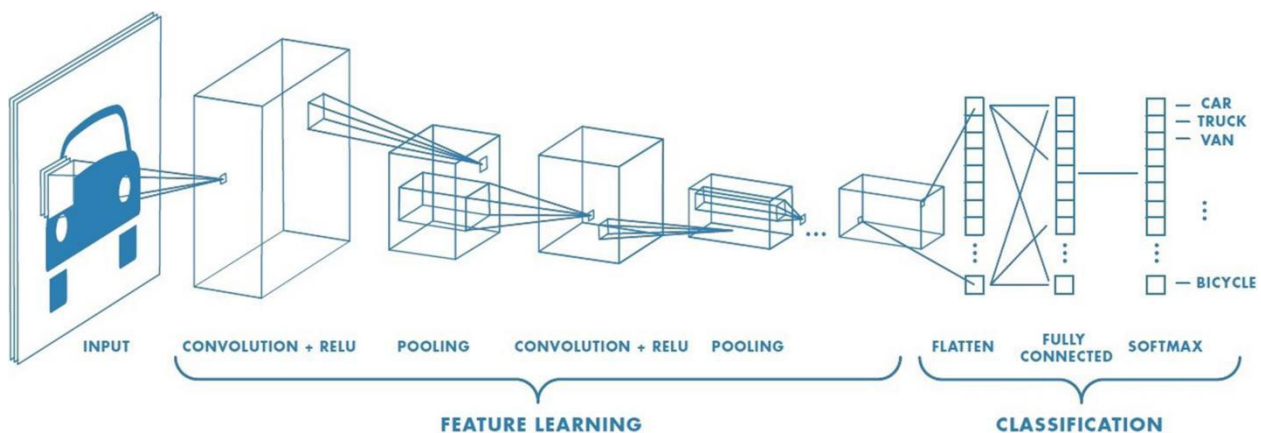

Fig 8.1.1. Preparing a Neural Net

Data Set:

As we know there is very limited research going on cattle, hence finding a dataset on platform like kaggle is not possible. So we did a research and then we scratch many websites to find a dataset containing more than 13000 images. The website we use is unsplash.com and we also used a dataset of cats and dogs from kaggle to train on non-cattle common animal. After downloading the dataset, we have reduced the size of images which at average was of 3MB to 300KB using various reduction technique of python CV2 library. After that, we labeled each images as cattle and non-cattle according to their types.

The code snippets are as follows:

```python
import urllib.request
import requests
from bs4 import BeautifulSoup
base_url = "https://unsplash.com/photos/"
urls_of_images = []
page = 0
import urllib
xyz="3150"
for x in range(83,100):
    URL = f"https://unsplash.com/napi/search/photos?query=cow&xp=&per_page=30&page={page}"
    res = requests.get(URL)
    res = res.json()
    page += 1

    array_of_image_data = res['results']

    for each_image in array_of_image_data:
        each_image_id = each_image['id']
        url=(f'{base_url}{each_image_id}/download')
        xyz=str(int(xyz)+1)
        (filename,headers)=urllib.request.urlretrieve(url, xyz)

l=1
print(len(urls_of_images))
for url in urls_of_images:
    #print(url)
    xyz=str(int(xyz)+1)
    (filename,headers)=urllib.request.urlretrieve(url, xyz)
#print(filename)
```

Fig 8.1.2. Downloading and sorting the Data Set.

```python
import os
path="C:\\Users\\hp\\Desktop\\web login\\New folder\\cats and dogs\\"
k=2
for i in os.listdir(".\\cats and dogs"):
    os.rename(path+i,path+'not_cattle'+str(k)+'.jpeg')
    k+=1
```

Fig 8.1.3. Changing the names of images

```python
from PIL import Image
import math
import os

arr=(os.listdir('.\\New folder\\New folder'))
path="C:\\Users\hp\\Desktop\\web login\\New folder\\New folder\\New folder\\"
for i in arr:
    print(i)
    try:
        image = Image.open(path+i)
        image.save(path+i,quality=8,optimize=True)
    except:
        pass
```

Fig 8.1.4. Reducing the size of Images.

After reduction of size of images, we have generated a greyscale of these images using python cv2 library and then resized the images to 50 * 50 image.

```python
def create_train_data():
    training_data = []
    for img in tqdm(os.listdir(TRAIN_DIR)):
        try:
            label = label_img(img)
            path = os.path.join(TRAIN_DIR,img)
            img = cv2.imread(path,cv2.IMREAD_GRAYSCALE)
            img = cv2.resize(img, (IMG_SIZE,IMG_SIZE))
            training_data.append([np.array(img),np.array(label)])
        except:
            pass
    shuffle(training_data)
    np.save('train_data.npy', training_data)
    return training_data
```

Fig 8.1.5. Creating the training data.

After this we have used a 2d neural network and used max pooling with 5 conv_2d layers in which layer contains 32,64,128,64,32 parameters. And then we use 2 fully connected layers with 1024 and 2 parameters respectively. The activation function used overall is relu with softmax only in last layer.

We have used 8600 images for our CNN model in which we have used 500 images in test and 8000 of images in training. In testing we have reached accuracy of 93% while in training we have reached accuracy of 99%.

```python
import tflearn
from tflearn.layers.conv import conv_2d, max_pool_2d
from tflearn.layers.core import input_data, dropout, fully_connected
from tflearn.layers.estimator import regression
tf.reset_default_graph()

convnet = input_data(shape=[None, IMG_SIZE, IMG_SIZE, 1], name='input')

convnet = conv_2d(convnet, 32, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = conv_2d(convnet, 64, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = conv_2d(convnet, 128, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = conv_2d(convnet, 64, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = conv_2d(convnet, 32, 5, activation='relu')
convnet = max_pool_2d(convnet, 5)

convnet = fully_connected(convnet, 1024, activation='relu')
convnet = dropout(convnet, 0.8)

convnet = fully_connected(convnet, 2, activation='softmax')
convnet = regression(convnet, optimizer='adam', learning_rate=LR, loss='categorical_crossentropy', name='targets')

model = tflearn.DNN(convnet, tensorboard_dir='log')
```

Fig 8.1.6. Preparing the convolution layer.

```
model.fit({'input': X}, {'targets': Y}, n_epoch=15, validation_set=({'input': test_x}, {'targets': test_y}),
    snapshot_step=500, show_metric=True, run_id=MODEL_NAME)

Training Step: 3809  | total loss: 0.01132 | time: 33.537s
| Adam | epoch: 015 | loss: 0.01132 - acc: 0.9965 -- iter: 8064/8099
Training Step: 3810  | total loss: 0.01326 | time: 34.839s
| Adam | epoch: 015 | loss: 0.01326 - acc: 0.9953 | val_loss: 0.39265 - val_acc: 0.9260 -- iter: 8099/8099
--
```

Fig 8.1.7. Output of training, Accuracy

We have used 15 epoch in creating our model.

After training we are currently using laptop we camera to capture images and predicting result using those images. For this w have used cv2 library of python

Before prediction we have converted the image again to the format of training data that is a greyscale with shape of (-1,50,50,1).

```
if os.path.exists('{}.meta'.format(MODEL_NAME)):
    model.load(MODEL_NAME)
    print('model loaded!')
key = cv2. waitKey(1)
webcam = cv2.VideoCapture(0)
check, frame = webcam.read()
print(check,"ll") #prints true as long as the webcam is running
print(frame) #prints matrix values of each framecd
cv2.imshow("Capturing", frame)
cv2.imwrite(filename='saved_img.jpg', img=frame)
webcam.release()
img_new = cv2.imread('saved_img.jpg', cv2.IMREAD_GRAYSCALE)
img_new = cv2.imshow("Captured Image", img_new)
cv2.waitKey(1650)
path="C:\\Users\\hp\\Documents\\saved_img.jpg"
img = cv2.imread(path,cv2.IMREAD_GRAYSCALE)
img = cv2.resize(img, (IMG_SIZE,IMG_SIZE))
x=model.predict(img.reshape(-1,50,50,1))
if(x[0][0]>x[0][1]):
    print("cattle")
else:
    print("not cattle")
```

Fig 8.1.8. Running the model for detecting the target using webcam.

After prediction of result we have also used an internet service named way2sms for sending message as an alert to the farm owner. In this we have used request library of python for doing an API request of way2sms API .

```
import requests
import json

URL = 'https://www.way2sms.com/api/v1/sendCampaign'

# get request
def sendPostRequest(reqUrl, apiKey, secretKey, useType, phoneNo, senderId, textMessage):
  req_params = {
  'apikey':"LD86DCYCTZTZJD18M074LIVISOSH9TPZ",
  'secret':"MUDJ7EAM6696ISV4",
  'usetype':'stage',
  'phone': "8299159085",
  'message':"Cattle spotted in farm",
  'senderid':"Dharmesh"
  }
  return requests.post(reqUrl, req_params)

# get response
response = sendPostRequest(URL, 'provided-api-key', 'provided-secret',
                          'prod/stage', 'valid-to-mobile',
                          'active-sender-id', 'message-text' )

# print response if you want
print (response.text)
```

Fig 8.1.9. Code sending a response of detecting the target.

On detecting any target animal in the vicinity, this program sends a request to the local client to activate the buzzers. The ultrasonic buzzers are planted in the field to scare away the animals which intend to cause damage. After the buzzer timeout, if the animal is still in the vicinity, then a message is sent to the owner with a warning.

```python
import requests
t=time.time()
if(t-time.time()>2000):
    try:
        t = time.time()
        print(ts)
        (requests.get("http://192.168.43.138/qazwsx/buzzer"))
    except:
        pass


# In[ ]:


import time
while(1):
    try:
        (requests.get("http://192.168.43.138/qazwsx/buzzer"))
    except:
        time.sleep(20)
        pass


# In[ ]:


(requests.get("http://192.168.43.138/qazwsx/buzzer"))
```

Fig 8.1.10. Code sending a request to the Local server to initiate the buzzer timer.

9. CONCLUSION

We would like to conclude our project work by defining the ways in which we are able to achieve our objectives.

- We have perfectly designed a system which can fetch the weather details and automate the farm irrigation activities.
- We have perfectly designed a system which can fairly control all the activities in the farm using internet.
- Our project uses the global internet facility to communicate between the local and global server.
- Our database is secured under the security facility of MongoDB Atlas and AWS Cloud Services.
- We have designed a system to identify harmful animals which can damage the crops and scare them away without causing any harm, if persistent then alert the farmer about it.
- Our project can be easily implemented into various other landforms like public parks, gardens etc, for controlling lights, sprinklers etc. They can also be automated.
- Our project provides life support to the farmers, for they do not need to go in dark of night to turn on the turbines, etc. They can fairly control the system from a distance.
- Our project also provide technical support to the farmer about the upcoming weather conditions.
- The total cost to automate the farm is the minimum for our system.
- The security parameters of the product are highly tested. On the wire, the requests being made to the client are supported only if they include the correct passkey which must match with the passkey of the client, else the request will fail.

10. FUTURE WORK

In the course of coming future, we are intended to remove potential vulnerabilities and support more variables to improve the prediction system and provide better farm support. Also, there will be an attempt to increase the number of available options like detecting the type of soil, to provide closer insight of the farming conditions.

We plan to design devices that can improve the communication between the peers and we plan to convert the sensors into a network of independent wireless modules.

We have planned to automate more of the agricultural structures and propose our idea for official funding, to help us implement it on a large scale.

11. REFERENCES

We would highlight the following sources of information to provide us with ample amount of data to achieve the project's purpose.

[1] "India economic survey 2018: Farmers gain as agriculture mechanisation speeds up, but more R&D needed". *The Financial Express. 29 January 2018*. Retrieved 8 January 2019.

[2] "FAOSTAT, 2014 data". *Faostat.fao.org*. Retrieved 17 September 2011.

[3] "Irrigation water management in paddy".*Agropedia.iitk.ac.in*, Retrieved 23 March 2009.

[4] Biology of rice, *Department of Biotechnology, Ministry of Science and Technology, India,* 2011

[5] *Tea (Camellia sinensis),* L. O. Kuntze. Tamil Nadu Tea Plantation Corporation, May 2014.

[6] "Irrigation Scheduling for Winter Wheat in Southern Alberta," Agri-Facts, Alberta Agriculture and Rural Development, Alberta Government, accessed September 25, 2013.
http://www1.agric.gov.ab.ca/$department/deptdocs.nsf/all/agdex13535/$file/112_56 1-1.pdf?OpenElement

[7] Journal article in scholarly journal, T.S. Krishnamoorthy, Durgesh Nandhini, "A Study on Sugarcane Production in India" International Journal of Advanced Research in Botany (IJARB)Volume 3, Issue 2, 2017, PP 13-17.
Available, https://www.arcjournals.org/pdfs/ijarb/v3-i2/3.pdf

[8] Journal article in scholarly journal, Sapna Grewal, Sonia Goel, "Current research status and future challenges to wheat production in India ", International journal of biotechnology, Vol-14, October 2015, PP 445-454, Available, https://pdfs.semanticscholar.org/d06b/023e250b8879c8e8aa f9bd2d1efefa574258.pdf

[9] Journal article in scholarly journal S. Sandhu "Status Paper on Rice", Directorate of Rice Developement,
Available, http://drdpat.bih.nic.in/Downloads/Status-Paper-on-Rice.pdf