AI CROP MANAGEMENT AND MONITORING SYSTEM

by

ANUPAM KUNWAR - 19BCE1369
DIVYANSH - 19BCE1610

A project report submitted to

Dr. GEETHA S

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

in partial fulfilment of the requirements for the course of

CSE1901

TECHNICAL ANSWERS FOR REAL WORLD PROBLEMS

in

B.Tech. COMPUTER SCIENCE AND ENGINEERING

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Vandalur – Kelambakkam Road

Chennai – 600127

MAY 2022

BONAFIDE CERTIFICATE

Certified that this project report entitled **AI CROP MANAGEMENT AND MONITORING SYSTEM,** is a bonafide work of **ANUPAM KUNWAR (19BCE1369)** and **DIVYANSH (19BCE1610)** who carried out the project work under my supervision and guidance.

Dr. Geetha S

Professor

School of Computer Science and Engineering (SCOPE),

VIT University, Chennai

Chennai-600127

ABSTRACT

Agriculture is the primary source of livelihood for about 58% of India's population. Indian farmers are the backbone of our nation's economy as the Indian economy is primarily Agriculture based. Some farmers in the agricultural industry manage so many acres of land, it's almost impossible to get prompt updates and alerts about potential problems without help from technology.

To get access to this technology, they need to spend money which is a huge burden to farmers. In the case of soil testing farmers need to take the sample and get it tested for free from a government lab, but now these governments are not at all taking care which is leading farmers to private companies where they meet cost expenses.

So to pull farmers out of this problem, in this project we are planning to design an IoT based smart agriculture monitor whose data is collected and further analyzed to predict the soil suitability for crop plantation which helps the farmers to make a better choice and improve the crop yield.

Data has become a fundamental aspect in modern agriculture to assist producers with essential decision-making, and current developments in data management are causing Smart Farming to increase tremendously. With objective information obtained by sensors with the goal of maximizing production and sustainability, significant advantages emerge. This type of data-driven farm management relies on data to boost efficiency while reducing resource waste and pollution of the environment. Data-driven agriculture, combined with robotic solutions that incorporate artificial intelligence approaches, lays the foundation for future sustainable agriculture.

In a growing busy life of people, taking care of plants has become a tough task. In recent years the need for automation is increasing day by day. Also leads to water wastage in many cases. There are many automated solutions in the market which requires time dependent motor activation control for watering the plants. In this project, a smart plant watering mechanism is introduced using IOT sensors and cloud-based databases. The sensors collect and evaluate data regarding changing weather and soil moisture levels before sending notifications to the user's phone and thing speak. The primary objective is to reduce human intervention and avoid water wastage by watering the plants as per the requirements. Output of the system will be satisfying the requirements of the plants by watering.

TABLE OF CONTENTS

INTRODUCTION

Agriculture is undergoing a revolution fueled by new technologies, which appears to be highly promising because it will allow this key sector to reach new heights of farm productivity and profitability. In our project we are focusing on precision agriculture, which entails applying inputs just when and where they are required, has emerged as the third wave of the contemporary agricultural revolution, and it is currently being boosted by the expansion of farm knowledge systems due to the increased availability of data.

In the agricultural environment, the Internet of Things (IoT) refers to the use of sensors and other devices to convert every element and action involved in farming into data. IoT technologies are one of the reasons why agriculture can generate such a big amount of valuable information, and the agriculture sector is expected to be highly influenced by the advances in these technologies. We have extensively used IoT concepts in our project keeping in mind that the main advantages of the use of IoT are achieving higher crop yields and less cost.

1.1 OBJECTIVE

To design and develop an Automatic Crop monitoring system where sensors are used to collect information in the agricultural field. The different sensors used are temperature and humidity sensor and soil moisture sensor. The information collected by the sensors is sent to the Arduino microcontroller ATmega328.The collected information can be displayed in an LCD display. The information collected by the sensors are updated periodically through Wi-Fi. A GSM module is connected with the microcontroller through which the message about the farm condition is sent to the authorized person.

- In order to maximize water consumption and maintain a green environment, it is important to irrigate more effectively as water supplies become increasingly scarce and polluted. The goal of this project is to create a smart plant watering

and monitoring system that analyses and records environmental factors to help plants thrive. The sensors collect and evaluate data regarding changing weather and soil moisture levels before sending notifications to the user's phone and thingspeak.

- Publishing the end result data on thingspeak .
- Making the plant monitoring easier by using IoT.
- To reduce the wastage of water by avoiding the unwanted watering of the plant.

1.2 BENEFITS

- In the context of Smart Farming, advanced management systems are delivering practical solutions. Furthermore, while some farmers have extensive knowledge accumulated over many years of field labor, technology may give a systematic method to uncover unforeseen problems that are difficult to discover through visual inspection on periodic checks.
- Farms that decide to be technology-driven in some way, show valuable advantages, such as saving money and work, having an increased production or a reduction of costs with minimal effort, and producing quality food with more environmentally friendly practices.
- A greater use of Smart Farming services is vital to not only improving a farm's financial performance, but also to meet the food needs of an expanding population.
- Yield will increase on an average as optimized crop treatment such as accurate planting, watering, pesticide application and harvesting directly affects production rates.
- Local and commercial farmers can use an internet connection to monitor many fields in numerous locations across the world. Decisions can be taken instantly and from any location.

1.3 APPLICATIONS

- - The project we be used to monitor the plant automatically.
- - Send alert notifications to the users/owner.
- - Checks the soil condition to help start the better growth of the plant.
- - Reduces water wastage.

1.4 FEATURES

- Real time monitoring of plant even in the absence of the owner.
- Regular notification updates about the plant conditions on owners mobile phone.
- End data is published on thingspeak to keep the track of past plant contitions.
- Motion around the plant can be detected if owner is not present and alert will be sent to the owner mobile phone.
- Rain alerts will also be sent to the user to avoid unwanted watering of the plant.
- Watering is done automatically whenever the soil moisture drops to zero threshold and vice versa.

LITERATURE SURVEY

SL. NO	Name of the research paper	Link for the paper	Inference
1.	A Model for Smart Agriculture Using IoT	https://ieeexplore.ieee.org/abstract/document/7955360	This Paper proposes a suitable alternative in the form of a wise agricultural model. The proposed model informs about some unfortunate events like weather patterns, soil conditions, epidemics etc. to the Farmers so that they can be ready to face and overcome unfortunate events. By using the proposed approach, received updated information allows the farmers to cope with and even benefit from these changes. It is a really challenging task that needs to provide such knowledge because of the highly localized nature of agriculture information, specifically distinct conditions.
2.	Field monitoring using IoT in agriculture	https://ieeexplore.ieee.org/abstract/document/8342777	This Paper proposes a field monitoring system using IoT. It helps the farmers to take corrective measures for the protection of crops and propose a method for better crop yield. Monitoring vegetation fields from distant locations not only helps in better utilization of the manpower but also quality of the crop. The data collected can be used for data analysis and provide a better measure and better alternative to deal with upcoming events.
3.	Integration of RFID and sensor in agriculture using IOT.	https://ieeexplore.ieee.org/abstract/document/8358372	This Paper proposes a framework that can work with less labor. The framework proposed supplies water just when it is needed. Besides in dry ranges where there is deficient precipitation, the water system winds up noticeably troublesome. Thus, we require a programmed framework that will decisively screen and control the water necessities in the field. Introducing Smart water system framework spares time and guarantees prudent use of water. So, this

			paper utilizes the sensors to consequently check the need of water or a need of low, high temperature and advise the agriculturist by means of RFID.
4.	Smart Farming – IoT in Agriculture	**https://ieeexplore.ieee.org/abstract/document/8597264**	This Paper proposes a suitable option for the farmers for more efficient and accurate yield, Water transport is measured by a smart device that works on soil condition so if we can control water wastage then we are automatically controlling electricity wastage also. Other sections in agriculture are insecticide, fertilizers and pesticides as in this paper we are proposing use of IoT in a poly house and poly house is a fully covered structure so there is almost no effect of outside factors like insects do not enter and cannot harm the crop so there will be less need of insecticides.
5.	A Research paper on smart agriculture using IoT	**https://www.irjet.net/archives/V7/i7/IRJET-V7I7479.pdf**	This paper brings insights to construct a framework for robust working on fields and easy for farmers. One of the main areas where IOT based research is going on and new products are launching on an everyday basis to make the activities smarter and efficient towards better production is "Agriculture". Agriculture sector is regarded as the most crucial sector globally for ensuring food security. Talking of India farmers, which are right now in huge trouble and are at a disadvantageous position in terms of farm size, technology, trade, government policies, climate conditions etc.
6.	IoT based Smart Agriculture	**https://ijarcce.com/upload/2016/june-16/IJARCCE%20188**	This Paper proposes a field monitoring system using IoT. After the research in the agricultural field, researchers found that the yield of agriculture is decreasing day by day. However, use of technology in the field of agriculture plays an important role in increasing the production as well as in reducing the extra man power efforts. Some of the research attempts are done for

			betterment of farmers which provides the systems that use technologies helpful for increasing the agricultural yield. The technological development in Wireless Sensor Networks made it possible to use in monitoring and control of greenhouse parameters in precision agriculture
7.	Advancement of mechanical automation in the agriculture sector and overview of IoT	https://iopscience.iop.org/article/10.1088/1742-6596/1399/4/044042	This Paper proposes a suitable alternative in the form of a wise agricultural model. This innovation has revolutionized the field of precision agriculture. Different types of high-quality effective sensors have been installed in the field for efficient use of irrigation water, fertilizers, fungicides and disease prevention in different crops. This collected data is sent to the cloud IoT framework through an internet connection that assesses the data and evaluates the results. After that, farmer mobile receives all the information about climate and requirements for the application of specific fungicide for the prevention of disease
8.	Evaluation of soil pH and soil moisture with different field sensors	https://www.sciencedirect.com/science/article/abs/pii/S161886671830116X	The purpose of this study was to compare field methods of measuring soil pH and VMC and identify the most accurate and precise method of determination for use in an urban site assessment. In order to evaluate these relationships, sensors were tested across a range of soil moisture contents and textures commonly found in the urban setting. Soil pH was evaluated using two glass electrode sensors and five metal electrode sensors. These data collected can be used to predict the best crop yield production which gives benefit to farmers and this increases the scope of IoT in the Agriculture sector.
9.	Internet-of-Things (IoT)-Based Smart Agriculture: Toward Making the Fields Talk	https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8784034	By implementing the latest sensing and IoT technologies in agriculture practices, every aspect of traditional farming method can be fundamentally changed. Currently, seamless integration of wireless sensors and the IoT in smart agriculture can raise agriculture to levels which were previously unimaginable. By following the practices of smart

			agriculture, IoT can help to improve the solutions of many traditional farming issues, like drought response, yield optimization, land suitability, irrigation, and pest control. major applications, services and wireless sensors being used for smart agriculture applications
10.	Internet Of Things Based Innovative Agriculture Automation Using AGRIBOT	**https://www.academia.edu/32737564/Internet_Of_Things_Based_Innovative_Agriculture_Automation_Using_AGRIBOT**	This paper is a small-scale effort but the same can be implemented with enormous results in a large scale that benefits all farmers of the world. The components used can be upgraded to a later version which will have more advanced options. The use of various sensors can be implemented in the future, so that various other works like filling the seed tank, water tank, mixing fertilizer and many other activities can also be automated. Apart from ploughing, seed dispensing, spraying fertilizer and harvesting other farming processes like weeding, bird watching etc. can also be implemented in one robot thus making the machine capable of multitasking.

PROPOSED WORK

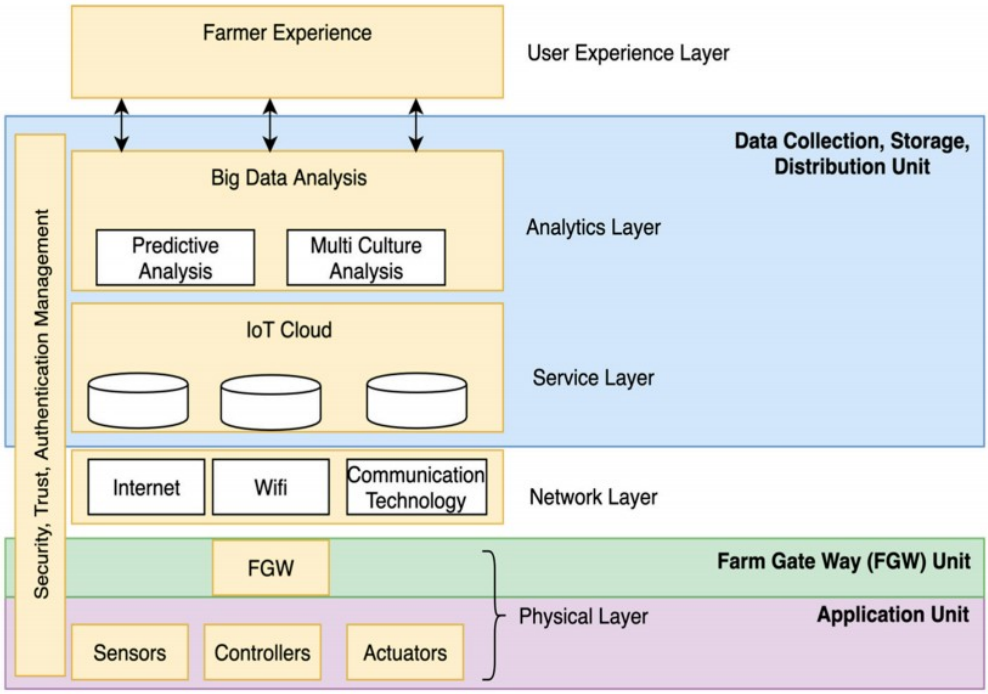
Considering the standard farming procedures, farmers need to visit the agriculture sites frequently throughout the crop life to have a better idea about the crop conditions. For this, the need for smart agriculture arises, as 70% of the farming time is spent monitoring and understanding the crop states instead of doing actual fieldwork. Considering the vastness of the agriculture industry, it incredibly demands technological and precise solutions with the aim of sustainability while leaving a minimum environmental impact.

3.1 REAL TIME SOLUTION

- The Crop Monitoring system will measure and record various soil parameters necessary for agriculture, like soil moisture, pH levels, temperature etc.
- The data collected from the sensor will be sent to a cloud server via a WIFI module, this data can be accessed by the service provider to perform the necessary analysis and draw efficient conclusions.
- Only collecting the data isn't important, we also need to make it ready for analysis. Data preprocessing is a very important first step for anyone dealing with data sets. That's because it leads to better data sets, that are cleaner and are more manageable, a must for any business trying to get valuable information from the data it gathers.
- After we have processed the collected data, we will decide what kind of output we want for which soil characteristics, for example qualitative, quantitative, predictive etc.
- As we manipulate and organize your data, we may need to traverse your steps again from the beginning, where we may need to modify our question, redefine parameters, and reorganize our data.
- Finally, we need to review if we have considered all parameters for making the decision, review if there is any hindering factor for implementing the decision. Choose data visualization techniques to communicate the message better. These visualization techniques may be charts, graphs, color coding, and more.

3.2 IMPACT AND FEASIBILITY

Sustainable agricultural development is expected to deliver more benefits to impoverished smallholder farmers and increase the resilience of communities that are highly vulnerable to extreme weather events associated with climate change.



Geographic feasibility: Topography relates to how difficult it is to till land, soil erosion and poor transportation networks and facilities. Agriculture can be mechanized depending on the topography of land to be used.

Crops thrive in rich, loamy soils with proper drainage. Crops absorb food and water through their roots from soil. They also enjoy plant support. Soils with poor texture and harsh chemicals are low in productivity. Therefore, poor soils inhibit plant growth and development. The complexity of the ‘smart tools’ should be adapted to the peculiarities of the farming and food safety in each selected geographic area.

Economic feasibility: With the global population increase, the demand for primary sector products has suffered an impulse that directly affects the economy. Due to this demand increase, management and traceability of production to prevent food loss or waste has become sometimes a difficult task. For that reason, the implementation of efficient processes for quality assurance and food loss avoidance is directly related to the economic revenues in each of the food chain states.

IoT implementations can be applied to improve the whole process and reach sustainable profitability through its efficiency. In order to evaluate the short/medium term economic benefits brought by the application of IoT, one should consider all investments incurred: development, deployment (installation and equipment), future updates, replacements, scalability, maintenance, etc.

Environmental feasibility: There is consensus on the benefits of introducing remote monitoring, control and application technologies to optimize input use efficiency, improve animal health and welfare, sustain product quality and safety, reduce the impact of machinery traffic on land, and promote effective delivery of environmental goods and services. Furthermore, the introduction of smart technologies in earlier stages of the agri-food chain can have important positive environmental effects across the whole chain. Moreover, available datasets such as those historically collected by cooperatives and unions during years (e.g., on yield mapping and recording, soil measurements, crop and

animal data) just for production control, and data from weather stations can be used to feed novel applications and improve production processes.

Societal feasibility: Societal acceptance of the new technologies, tools or processes tested should also be included as a dissemination objective in the cases where it is considered important for its success. Education and training aspects should be included to help end-users understand the use and usefulness of the new technologies. This would benefit from integrating initiatives already running for “education and farming”: they would represent a well-proven way to disseminate IoT culture among youngsters and the stakeholders of the food chain. In addition, concrete measures to enhance digital skills in along the agricultural value chain could be proposed, including academic partners into this process

Demographic feasibility: Agricultural production is affected by demographic changes such as population growth and the growth of the nuclear family system. Such an increase in nuclear families have resulted in a decrease in per capita land holding of households, and thus the trend of marginal landholdings is increasing in India. As per NSSO data 63rd Round, average land holding per household is around 1.4 hectares. Food security is a major issue in India as the annual population growth rate is 1.8 percent and relative food production rate is 1.2 percent. India needs an agricultural growth rate of 4.0 to 4.5 percent to reduce poverty and food insecurity significantly. This will make available more land and water resources for the cultivation of high value-added crops.

Political feasibility: The concept of a "trade-off" between growth and improved distribution is used not only to rationalize and perpetuate the current unequal distribution of agricultural land, but also to justify the intensive use (per acre) of scarce inputs like fertilizers and water on large farms and in developed regions, when the social product would be higher by spreading such inputs thinly over a larger area. Unlike lumpy investments in projects like major and medium irrigation, which are location-bound and have a protracted gestation time, Rural Works Programs provide instant benefits to a huge number of people across a broad area. Second, these programmes assist individuals in positions of power in disbursing patronage to a huge number of middlemen who must be compensated and whose future services must be guaranteed.

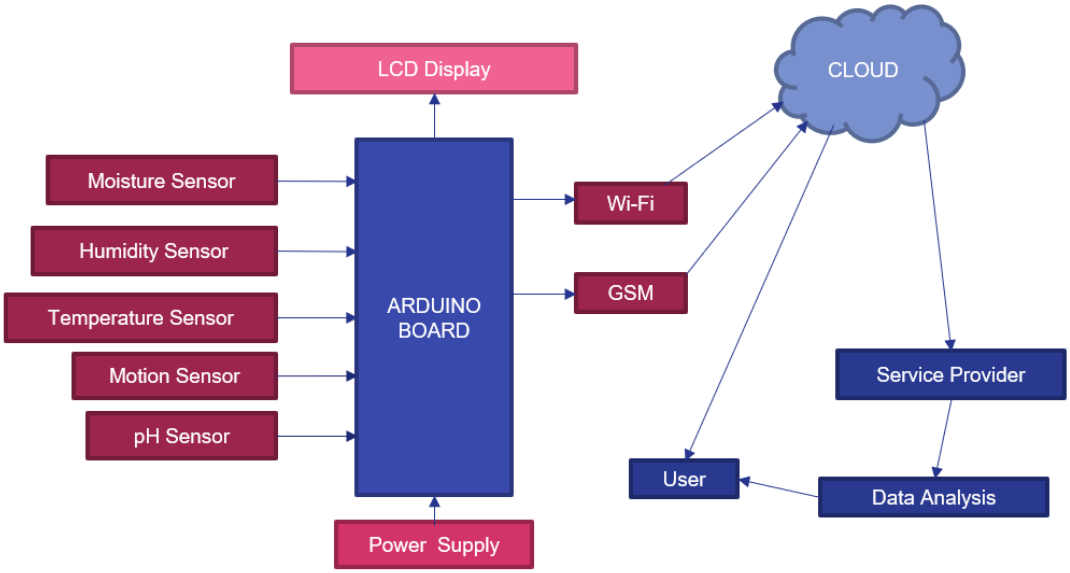
IMPLEMENTATION

The Design, technology choices, algorithms, implementation of the software of hardware for the project has been described in detail below:

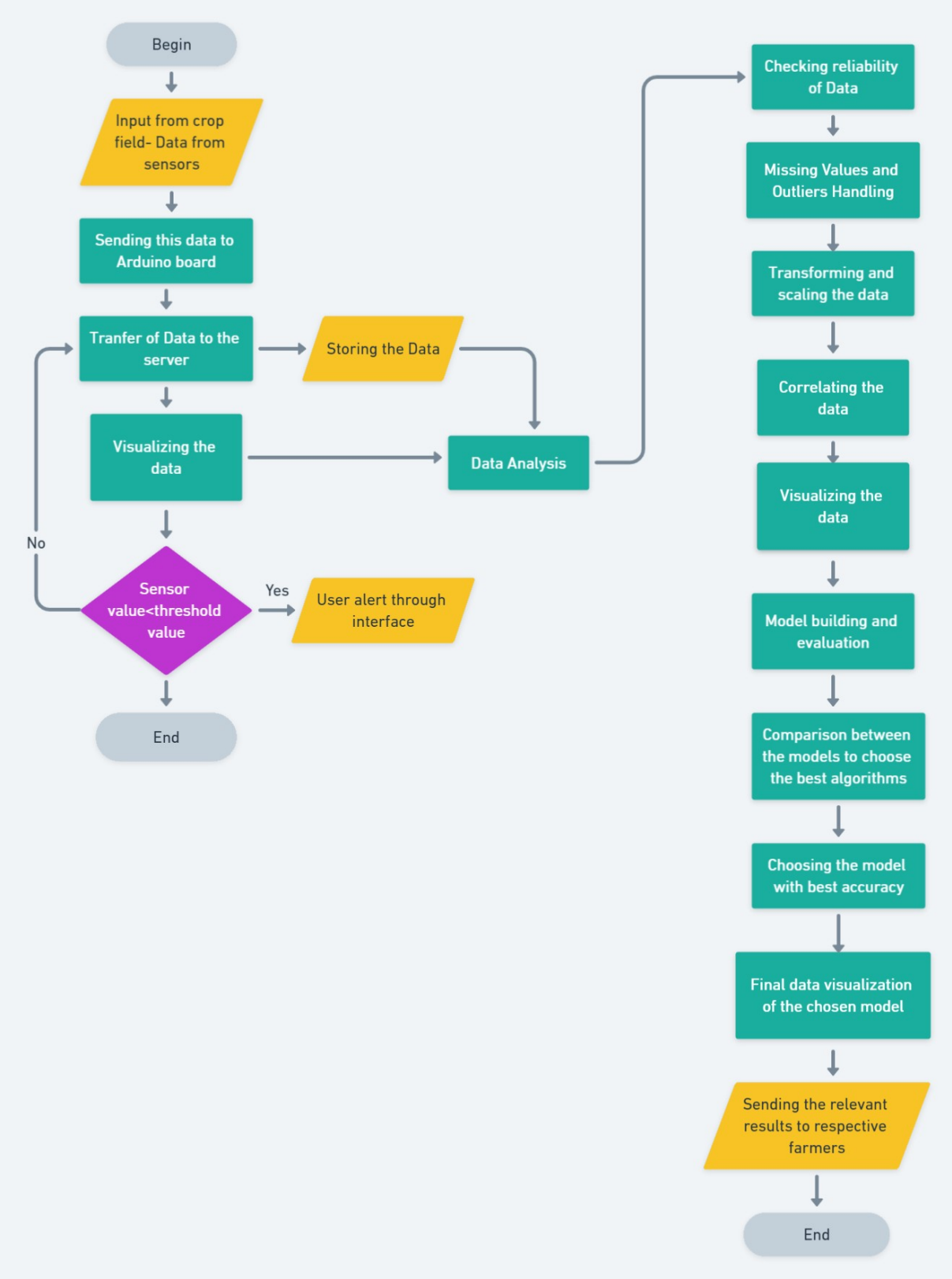
4.1 DESIGN APPROACH

- The application has been designed such that, initially the farmers should fill a registration form. As soon as the necessary entries have been made the Node-flows produce output according to the Json code from which the pH of the soil is predicted for the desired crop plantations.
- This is followed by the presentation of a detailed weather report. Based on the input (latitudes and longitudes) weather conditions are received from openweathermap.org website with a unique API ID. The temperature is predicted for the next five days along with the other relevant data (humidity, wind speed etc.) is displayed in a Node-red UI.
- pH_data CSV file is fed to the machine learning models. Machine learning concepts such as Decision tree, Random Forest, SVC and k-neighbors are implemented and accuracy is calculated. So that better model can be used for prediction. Initially we planned to use Naive Bayes and K-means but we had to go with the models available in the Node-red platform.
- Coming to the hardware part of our project, in real-time implementation live data would be received from the various sensors on the plantation field and that data would be fed to the machine learning models instead of data from the CSV file.
- The different sensors used are temperature and humidity sensor and soil moisture sensor. The information collected by the sensors is sent to the Arduino microcontroller ATmega328.
- The collected information can be displayed in an LCD display. The information collected by the sensors are updated periodically through Wi-Fi. A GSM module is connected with the microcontroller through which the message about the farm condition is sent to the authorized person.

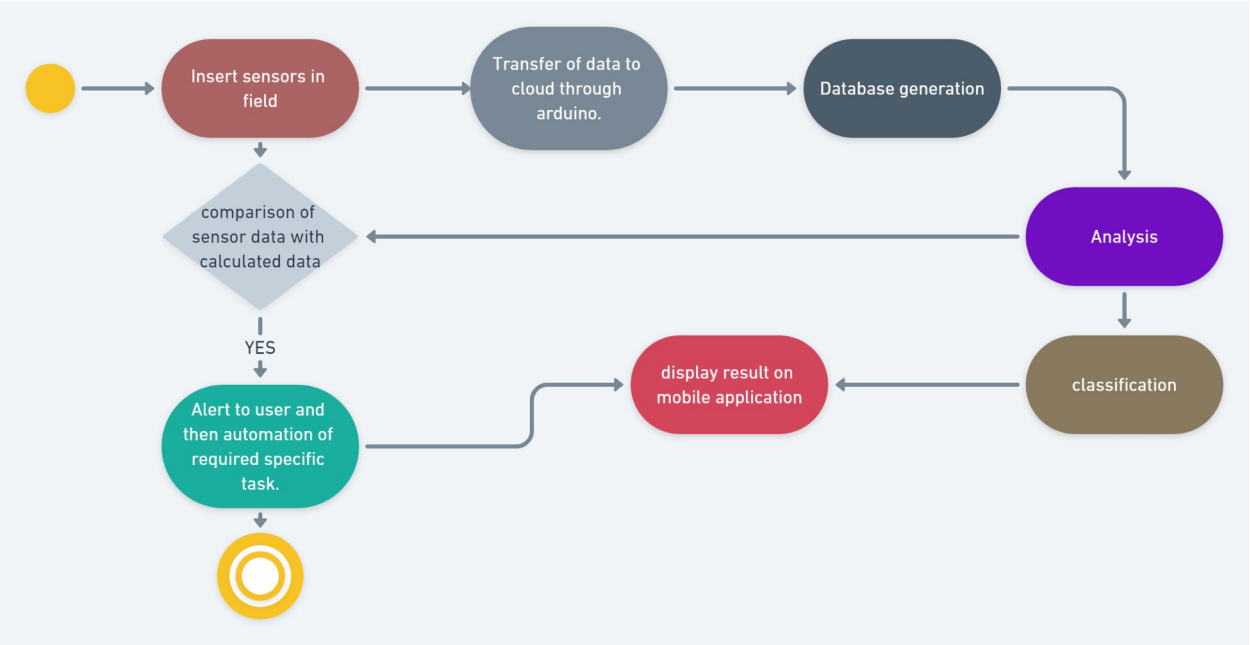
4.2 BLOCK DIAGRAM OF SYSTEM DESIGN



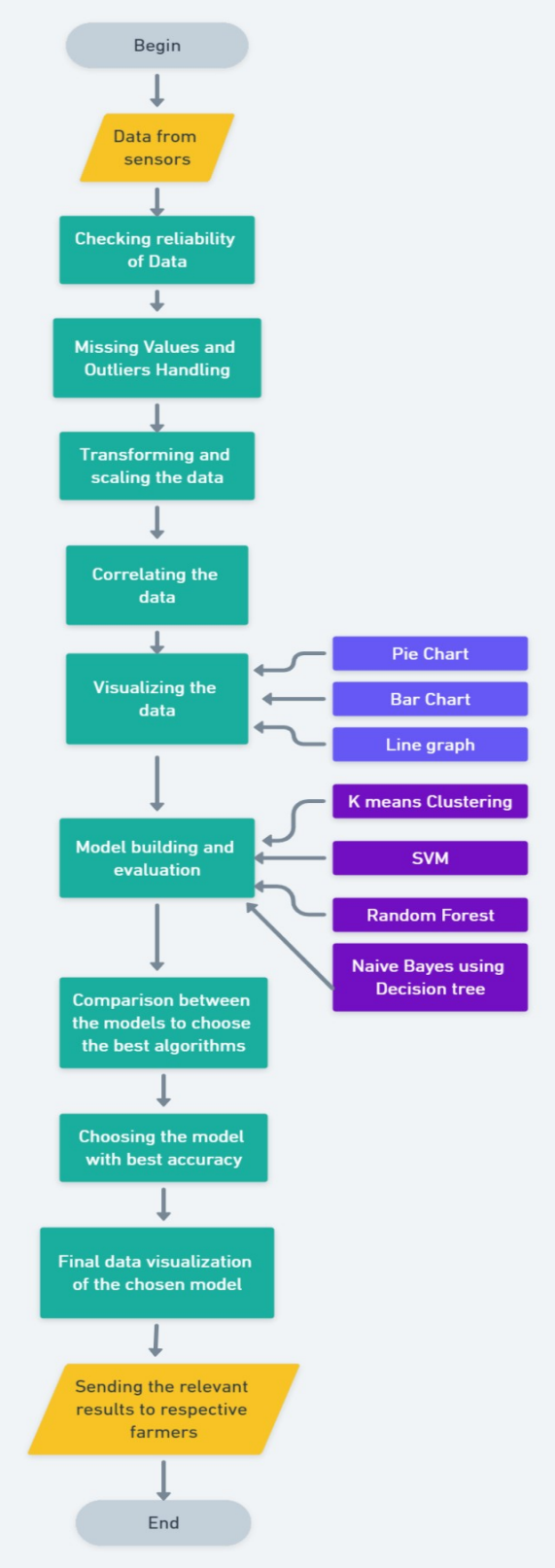
4.3 OVERALL FLOW DIAGRAMS



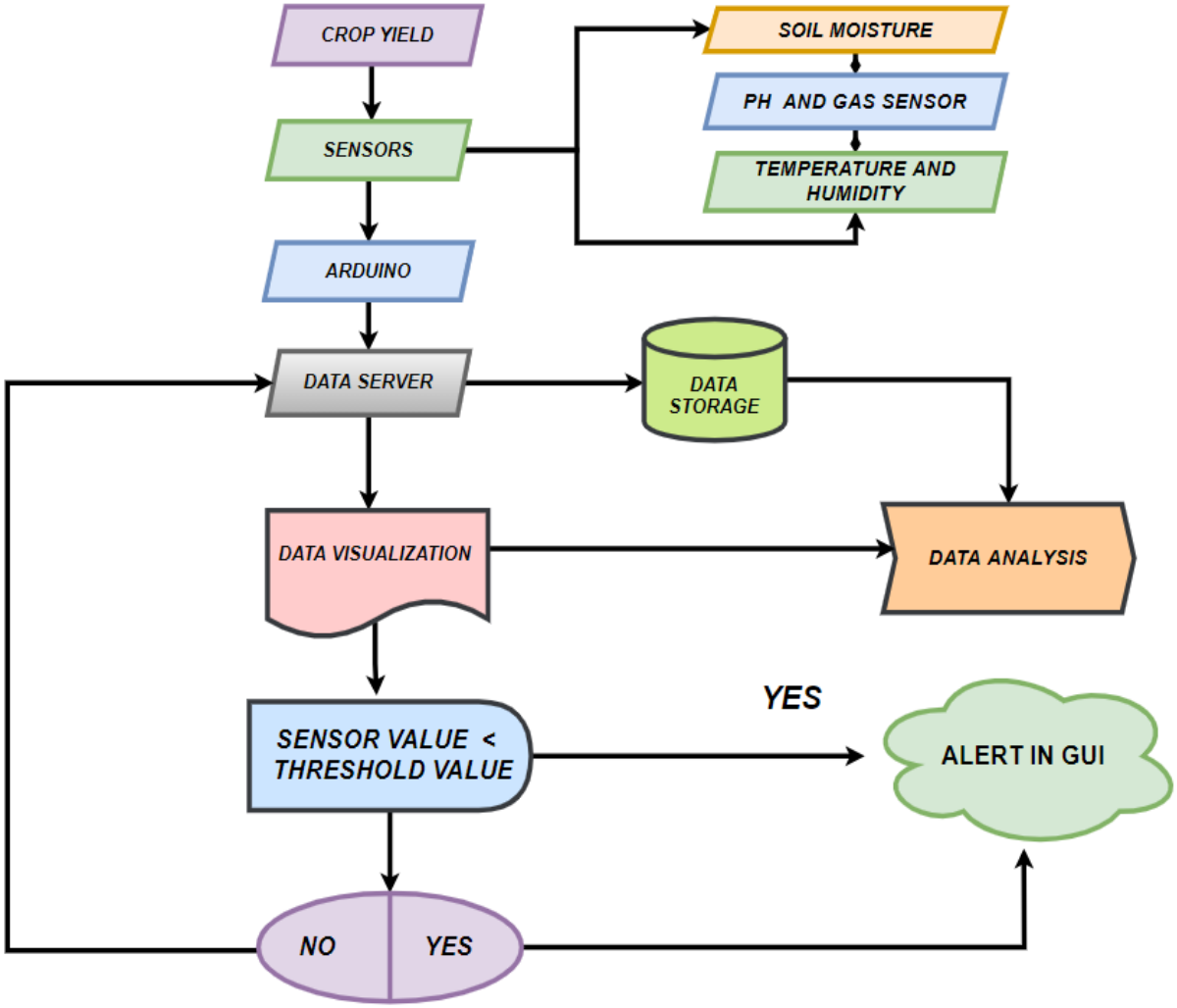
Overall activity diagram:



Overall software flow diagram:



Overall hardware flow diagram:



4.4 IMPLEMENTATION ALGORITHMS

Overall Algorithm for Software Implementation

1. Begin
2. Data <- enter sensor data
3. if data is received
   - 3.1. Check reliability of data
   - 3.2. Missing Values and Outliers Handling
   - 3.3. Data cleaning (remove outliers, data imputation etc)
   - 3.4. Data pre-processing (data reduction-PCA, data integration etc)
   - 3.5. Transforming and scaling the data
   - 3.6. Observe correlation between the values
   - 3.7. Visualizing the data
   - 3.8. Pie chart
   - 3.9. Boxplot
   - 3.10. Bar plot, histogram etc
   - 3.11. Model building and evaluation
   - 3.12. K Neighbours
   - 3.13. SVC
   - 3.14. Random Forest
   - 3.15. Decision tree
   - 3.16. Comparison between the models to choose the best model
   - 3.17. Choosing the model with the best accuracy
   - 3.18. Final data visualization of the chosen model
   - 3.19. Sending the relevant data to respective farmers
4. else
   - 4.1. Go to step 2
5. End

Overall Algorithm for Hardware Implementation

1. Begin
2. pH sensor <- input from the crop field
3. Moisture sensor <- Input from the crop field
4. Temperature sensor <- Input from the crop field
5. Send the received data to the Arduino for further processing
6. Send received data to the Data cloud
7. For further analysis, this data is stored in the cloud
8. This is followed by data visualization
9. Perform data analysis
10. Find threshold value
11. if (sensor value > threshold value)

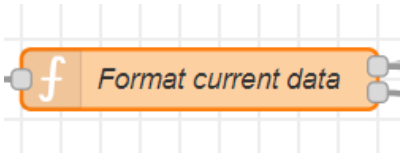
11.1. Go to step 3
12. else
12.1. Alert in GUI
13. End

4.5 CODING AND RESULTS

SOFTWARE IMPLEMENTATION:

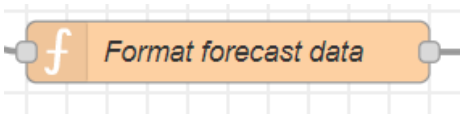
For fetching the weather data:

Code for formatting current and forecast data:




```
var icon = {};

var units = flow.get('units');
if (units === undefined) {units = 'imperial';}

function timeConvert(UNIX_timestamp){
  var dateObject = new Date(UNIX_timestamp * 1000);
  if (flow.get('hour12')) {  // 12 hour time format
       return  dateObject.toLocaleString('en',  {  timezone:  msg.payload.timezone,  hour12:  true,  hour:
'numeric', minute: '2-digit'}).toLowerCase();
  } else {  // 24 hour time format
       return  dateObject.toLocaleString('en',  {  timezone:  msg.payload.timezone,  hour12:  false,  hour:
'numeric', minute: '2-digit'});
  }
}

var degreesToCardinal = function(deg){
  if (deg>11.25 && deg<=33.75){return "NNE";}
  else if (deg>33.75 && deg<56.25){return "NE";}
  else if (deg>56.25 && deg<78.75){return "ENE";}
  else if (deg>78.75 && deg<101.25){return "E";}
  else if (deg>101.25 && deg<123.75){return "ESE";}
  else if (deg>123.75 && deg<146.25){return "SE";}
  else if (deg>146.25 && deg<168.75){return "SSE";}
  else if (deg>168.75 && deg<191.25){return "S";}
  else if (deg>191.25 && deg<213.75){return "SSW";}
  else if (deg>213.75 && deg<236.25){return "SW";}
  else if (deg>236.25 && deg<258.75){return "WSW";}
  else if (deg>258.75 && deg<281.25){return "W";}
  else if (deg>281.25 && deg<303.75){return "WNW";}
  else if (deg>303.75 && deg<326.25){return "NW";}
```

```
  else if (deg>326.25 && deg<348.75){return "NNW";}
  else {return "N";}
}

if (units == "imperial")
{
  msg.payload.current.temp = msg.payload.current.temp.toFixed() + ' °F';
  msg.payload.current.wind_speed = msg.payload.current.wind_speed.toFixed() + ' mph';
}
else  // metric units
{
  msg.payload.current.temp = msg.payload.current.temp.toFixed(1) + ' °C';
  msg.payload.current.wind_speed = msg.payload.current.wind_speed.toFixed(1) + ' m/s';
}

msg.payload.current.wind_cardinal = degreesToCardinal(msg.payload.current.wind_deg);

msg.payload.current.sunrise = timeConvert(msg.payload.current.sunrise);
msg.payload.current.sunset = timeConvert(msg.payload.current.sunset);

var iconString = 'wi-owm-' + msg.payload.current.weather[0].icon + ' wi-4x';
icon = {
    ui_control: {
        icon: iconString
    }
};

return [msg, icon];
```

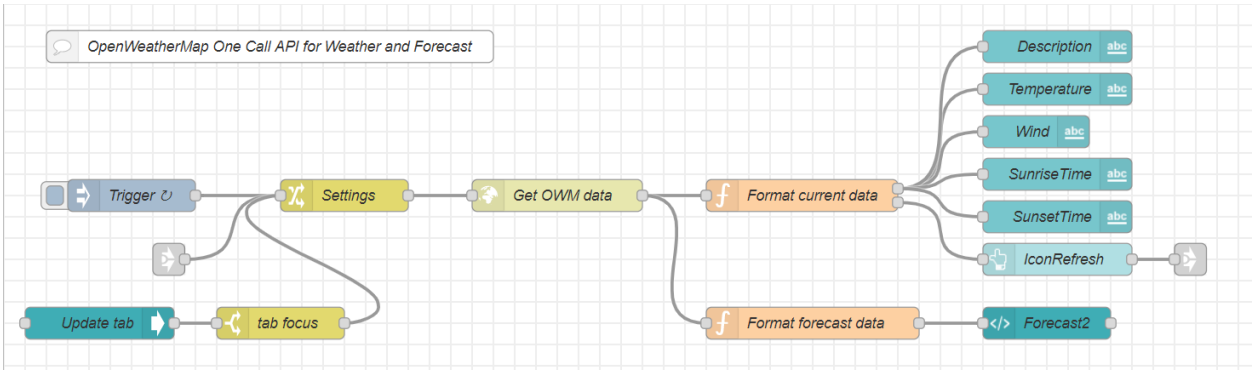



```
var fcdata = {};
var units=flow.get('units');
if (units===undefined)
{
  units="imperial";
}

function formatTemp(high, low){
    if (units == "imperial") {
        if (low){
          temp = parseFloat(high).toFixed() + '/' + parseFloat(low).toFixed()
        }
        else {
          temp = parseFloat(high).toFixed() + '°F'
        }
    }
    else { // metric
        if (low){
          temp = parseFloat(high).toFixed() + '/' + parseFloat(low).toFixed()
        }
        else {
          temp = parseFloat(high).toFixed() + '°C'
```

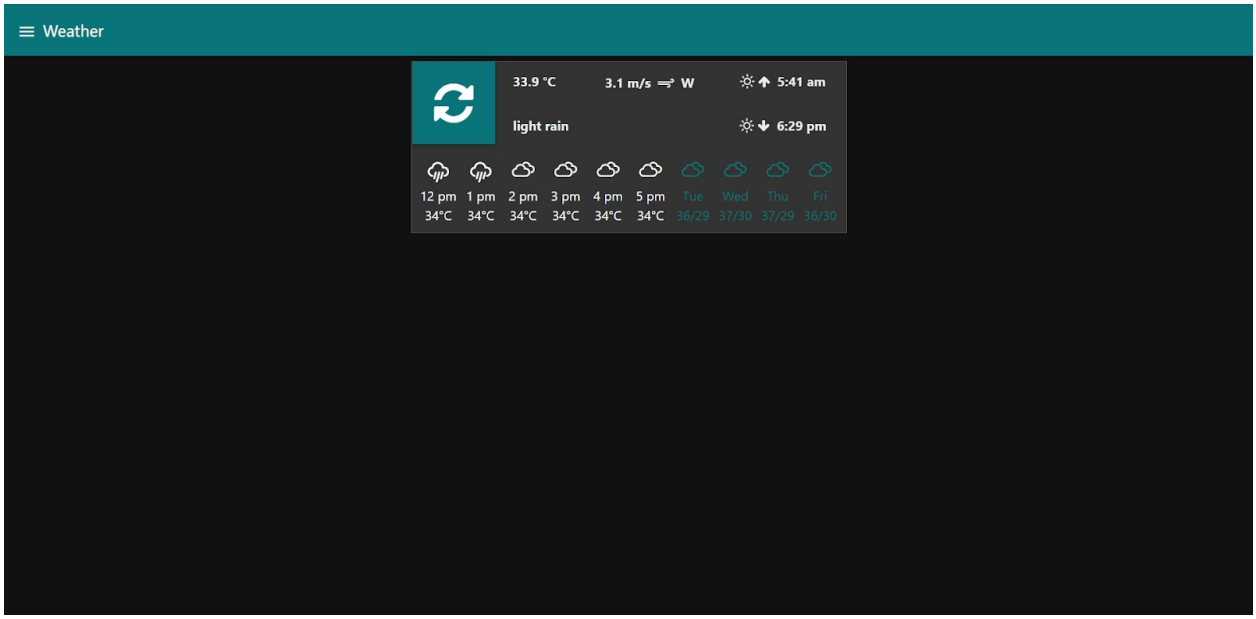
```
      }
    }
    return temp;
}

function dayName(unixTime){
  var dateObject = new Date(unixTime * 1000);
    return dateObject.toLocaleString(flow.get('lang'), { timezone: msg.payload.timezone, weekday: 'short'});
  // If the line above is not producing the correct short weekday names for the language set by 'lang',
  //   you can try upgrading to NodeJS version 13 or higher (when full international support was added for the toLocaleString function),
  //   or you can use the code below which is an example for French short weekday names.
/*
  switch (dateObject.toLocaleString('en', { timezone: msg.payload.timezone, weekday: 'short'})) {
    case 'Mon':
      return 'Lun';
    case 'Tue':
      return 'Mar';
    case 'Wed':
      return 'Mer';
    case 'Thu':
      return 'Jeu';
    case 'Fri':
      return 'Ven';
    case 'Sat':
      return 'Sam';
    case 'Sun':
      return 'Dim';
  }
*/
}

function timeConvert(UNIX_timestamp){
  var dateObject = new Date(UNIX_timestamp * 1000);
  if (flow.get('hour12')) {  // 12 hour time format
       return dateObject.toLocaleString('en', { timezone: msg.payload.timezone, hour12: true, hour: 'numeric'}).toLowerCase();
  } else {  // 24 hour time format
       return dateObject.toLocaleString('en', { timezone: msg.payload.timezone, hour12: false, hour: 'numeric'}) + ':00';
  }
}

// prepare forecast data for CSS based ui widget
fcdata.payload = {
  rowtext: {
            data01: {
      cell01: timeConvert(msg.payload.hourly[1].dt),
      cell02: timeConvert(msg.payload.hourly[2].dt),
      cell03: timeConvert(msg.payload.hourly[3].dt),
      cell04: timeConvert(msg.payload.hourly[4].dt),
      cell05: timeConvert(msg.payload.hourly[5].dt),
      cell06: timeConvert(msg.payload.hourly[6].dt),
      cell07: dayName(msg.payload.daily[1].dt),
```

```
    cell08: dayName(msg.payload.daily[2].dt),
    cell09: dayName(msg.payload.daily[3].dt),
    cell10: dayName(msg.payload.daily[4].dt),
        },
        data02: {
     cell01: formatTemp(msg.payload.hourly[1].temp),
     cell02: formatTemp(msg.payload.hourly[2].temp),
     cell03: formatTemp(msg.payload.hourly[3].temp),
     cell04: formatTemp(msg.payload.hourly[4].temp),
     cell05: formatTemp(msg.payload.hourly[5].temp),
     cell06: formatTemp(msg.payload.hourly[6].temp),
     cell07: formatTemp(msg.payload.daily[1].temp.max, msg.payload.daily[0].temp.min),
     cell08: formatTemp(msg.payload.daily[2].temp.max, msg.payload.daily[1].temp.min),
     cell09: formatTemp(msg.payload.daily[3].temp.max, msg.payload.daily[2].temp.min),
     cell10: formatTemp(msg.payload.daily[4].temp.max, msg.payload.daily[3].temp.min),
        }
 },
 rowicons: {
        data01: {
                cell01: msg.payload.hourly[1].weather[0].icon,
                cell02: msg.payload.hourly[2].weather[0].icon,
                cell03: msg.payload.hourly[3].weather[0].icon,
                cell04: msg.payload.hourly[4].weather[0].icon,
                cell05: msg.payload.hourly[5].weather[0].icon,
                cell06: msg.payload.hourly[6].weather[0].icon,
                cell07: msg.payload.daily[1].weather[0].icon,
                cell08: msg.payload.daily[2].weather[0].icon,
                cell09: msg.payload.daily[3].weather[0].icon,
                cell10: msg.payload.daily[4].weather[0].icon,
        }
 }
}

return fcdata;
```

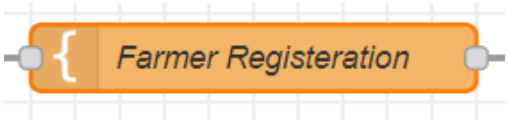
Node-flows:



Node-red dashboard-UI



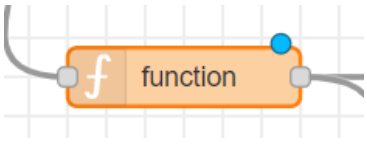
Code for farmer registration and crop inputs followed by pH prediction:




```
<!DOCTYPE html>
<html>
   <head>
      <h1 style="background-color:DogerBlue;">Farmer's Registration Form</h1>
   </head>
   <body>
<p style="background-color:Tomato;">

<form method ="post" action="/{{url}}">


<label for="name">Full name: </label><br>
<input type-"text" id-"fname" name-"fname"><br><br>

<label for="Contact">Contact Number:</label><br>
<input type="number" name="Contact" /><br><br>

<label for="email">Email:</label><br>
<input type="text" name="email" size="60" /><br><br>

<label for="age">Age:</label><br>
<input type="number" name="age" size="60" /><br>
<br>
<label for="state">State:</label><br>
<input type="text" name="place" size="60" /><br>
<br>
<label for="City">City:</label><br>
<input type="text" name="city" size="60" /><br>
<br>
```

```
<label for="area">Area in hectare:</label><br>
<input type="number" name="area" size="60" /><br>
<br>
<label for="Rabi crops">Rabi crops</label><br>
<input type="checkbox" id="Wheat" name="Wheat" value="Y">
<label for="Wheat"> wheat </label><br>
<input type="checkbox" id="Peas" name="Peas" value="Y">
<label for="Peas"> Peas </label><br>
<input type="checkbox" id="Barley" name="Barley" value="Y">
<label for="Barley"> Barley </label><br>
<input type="checkbox" id="Gram" name="Gram" value="Y">
<label for="Gram"> Gram </label><br>
<input type="checkbox" id="Mustard" name="Mustard" value="Y">
<label for="Mustard"> Mustard </label><br>
<br>
<label for="Kharif crops">Kharif crops</label><br>
<input type="checkbox" id="Rice" name="Rice" value="Y">
<label for="Rice"> Rice </label><br>
<input type="checkbox" id="Maize" name="Maize" value="Y">
<label for="Maize"> Maize </label><br>
<input type="checkbox" id="Jowar" name="Jowar" value="Y">
<label for="Jowar"> Jowar </label><br>
<input type="checkbox" id="Groundnut" name="Groundnut" value="Y">
<label for="Groundnut"> Groundnut </label><br>
<input type="checkbox" id="Cotton" name="Cotton" value="Y">
<label for="Cotton"> Cotton </label><br>
<input type="checkbox" id="Soyabean" name="Soyabean" value="Y">
<label for="soyabean"> Soyabean </label><br>
<br>
<label for="Zaid crops">Zaid crops</label><br>
<input type="checkbox" id="Seasonal_Fruits" name="Seasonal_Fruits" value="Y">
<label for="Seasonal_Fruits"> Seasonal_Fruits </label><br>
<input type="checkbox" id="Vegetables" name="Vegetables" value="Y">
<label for="Vegetables"> Vegetables </label><br>
<input type="checkbox" id="Fodder_Crops" name="Fodder_Crops" value="Y">
<label for="Fodder_Crops"> Fodder_Crops </label><br>
<br>
<input type="submit" value="Register">
<input type="reset" value="RESET">
   </body>
</html>
```

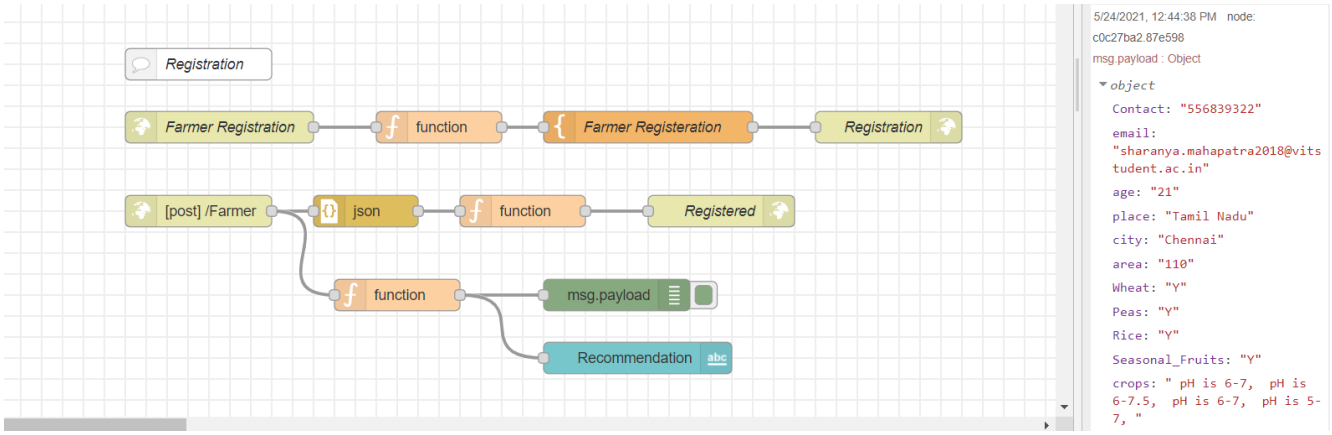
3rd function node



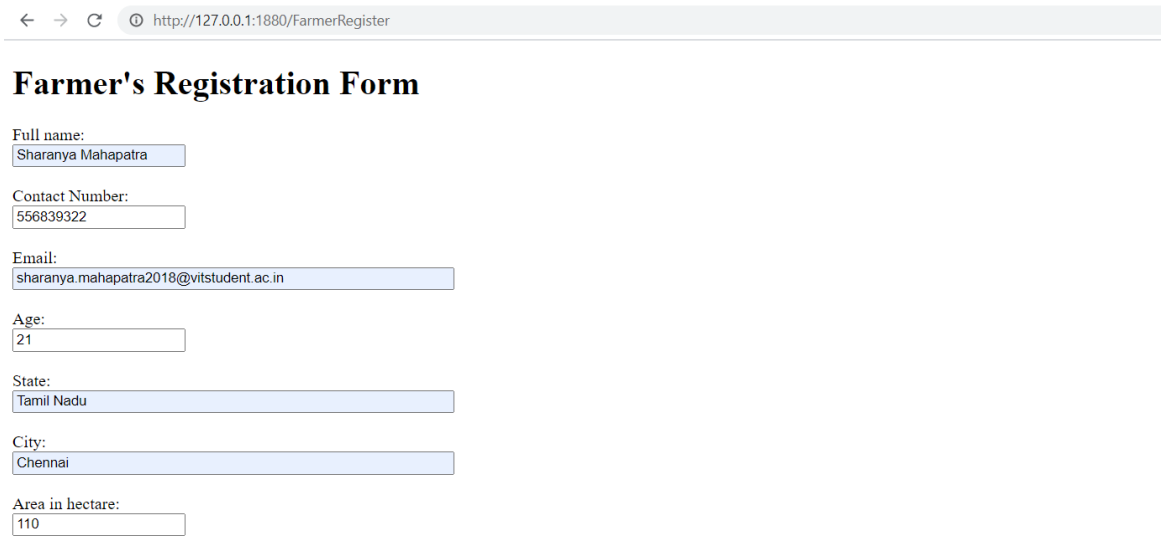

```
msg.payload.crops=" ";
if(msg.payload.Wheat=="Y")
{
   msg.payload.crops=msg.payload.crops+"pH is 6-7, ";
}
if(msg.payload.Peas=="Y")
{
```

```
    msg.payload.crops=msg.payload.crops+" pH is 6-7.5, ";
}
if(msg.payload.Barley=="Y")
{
    msg.payload.crops=msg.payload.crops+" pH is 5-7, ";
}
if(msg.payload.Gram=="Y")
{
    msg.payload.crops=msg.payload.crops+" pH less than 6, ";
}
if(msg.payload.Mustard=="Y")
{
    msg.payload.crops=msg.payload.crops+" pH is 3.5-6, ";
}
if(msg.payload.Rice=="Y")
{
    msg.payload.crops=msg.payload.crops+" pH is 6-7, ";
}
if(msg.payload.Maize=="Y")
{
    msg.payload.crops=msg.payload.crops+" pH is 6-7.2, ";
}
if(msg.payload.Jowar=="Y")
{
    msg.payload.crops=msg.payload.crops+" pH is 7.5, ";
}
if(msg.payload.Groundnut=="Y")
{
    msg.payload.crops=msg.payload.crops+" pH is 6-6.5, ";
}
if(msg.payload.Cotton=="Y")
{
    msg.payload.crops=msg.payload.crops+" pH is 5.5-8.2, ";
}
if(msg.payload.Soyabean=="Y")
{
    msg.payload.crops=msg.payload.crops+" pH is 6.3-6.5, ";
}
if(msg.payload.Seasonal_Fruits=="Y")
{
    msg.payload.crops=msg.payload.crops+" pH is 5-7, ";
}
if(msg.payload.Vegetables=="Y")
{
    msg.payload.crops=msg.payload.crops+" pH is 6-7, ";
}
if(msg.payload.Fodder_Crops=="Y")
{
    msg.payload.crops=msg.payload.crops+" pH is 6.6-7, ";
}
return msg;
```

Node-flows:



Browser window:



Rabi crops
- [x] wheat
- [x] Peas
- [ ] Barley
- [ ] Gram
- [ ] Mustard

Kharif crops
- [x] Rice
- [ ] Maize
- [ ] Jowar
- [ ] Groundnut
- [ ] Cotton
- [ ] Soyabean

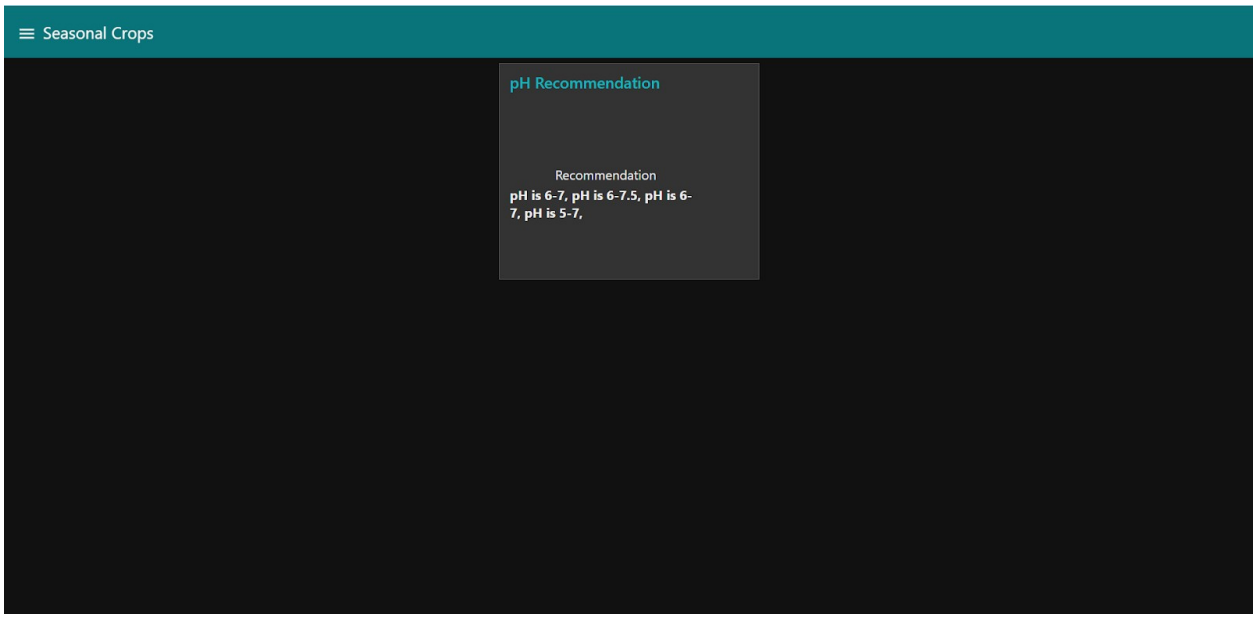
Zaid crops
- [x] Seasonal_Fruits
- [ ] Vegetables
- [ ] Fodder_Crops

Register RESET

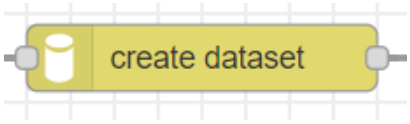
http://127.0.0.1:1880/Farmer

Data Submited and is available in debug window; {"Contact":"556839322","email":"sharanya.mahapatra2018@vitstudent.ac.in","age":"21","place":"Tamil Nadu","city":"Chennai","area":"110","Wheat":"Y","Peas":"Y","Rice":"Y","Seasonal_Fruits":"Y"}

Node-red dashboard-UI



Machine learning models to compare accuracy



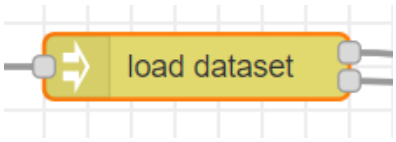
Json code:

```
module.exports = function(RED){
	function createDatasetNode(config){
		const path = require('path')
		const utils = require('../../../utils/utils')

		var node = this;

		//set configurations
		node.file = __dirname + '/create-dataset.py'
		node.config = {
			path: config.path,
			save: path.join(config.saveFolder, config.saveName),
			input: utils.listOfInt(config.input) || [0],
			output: parseInt(config.output) || undefined,
			trainingPartition: (Number(config.trainingPartition) || 80.0) / 100.0,
			shuffle: Boolean(config.shuffle),
			seed: parseInt(config.seed) || 0
		}

		utils.run(RED, node, config)
	}
	RED.nodes.registerType("create dataset", createDatasetNode)
}
```

Inbuilt python code:

```
while True:
	#wait request
	input()

	df = pandas.read_csv(config['path'], header=None)

	r, c = df.shape

	if config['shuffle']:
		df = df.sample(frac=1, random_state=config['seed'])

	x_train = df.iloc[: int(r * config['trainingPartition']), config['input']]
	x_test = df.iloc[int(r * config['trainingPartition']) :, config['input']]

	if 'output' in config:
		y_train = df.iloc[: int(r * config['trainingPartition']), config['output']]
		y_test = df.iloc[int(r * config['trainingPartition']) :, config['output']]
```

```
            x_train[x_train.shape[1]] = y_train
            x_test[x_test.shape[1]] = y_test

    if not os.path.isdir(config['save']):
            os.makedirs(config['save'], exist_ok=True)

    x_train.to_csv(config['save'] + '/train.csv', header=None, index=False)
    x_test.to_csv(config['save'] + '/test.csv', header=None, index=False)

    print('Dataset created.')
```



Json code:

```
import json
import pandas
import os

#read configurations
config = json.loads(input())

module.exports = function(RED){
    function loadDatasetNode(config){
            const path = require('path')
            const utils = require('../../../utils/utils')

            var node = this

            //set configurations
            node.file = __dirname + '/load-dataset.py'
            node.topic = 'real'
            node.config = {
                path:       path.join(config.datasetFolder,       config.datasetName,
config.partition),
                input: Boolean(config.input),
                output: Boolean(config.output)
            }

            utils.run(RED, node, config)
    }
    RED.nodes.registerType("load dataset", loadDatasetNode);
}
```

Inbuilt python code:

```
import sys
import json
```

```
#read configurations
config = json.loads(input())

while True:
        #wait request
    input()

    if config['input'] or config['output']:
        df = pandas.read_csv(config['path'], header=None)

        r, c = df.shape

        first_column = 0 if config['input'] else c - 1
        last_column = c if config['output'] else c - 1

        df = df.iloc[:, first_column:last_column]

        if config['output'] and not config['input']:
            res = json.dumps(df[df.columns[0]].tolist())
        else:
            res = df.to_json(orient='values')
        print(res)

    else:
        print('Nothing to load.', file=sys.stderr)
```

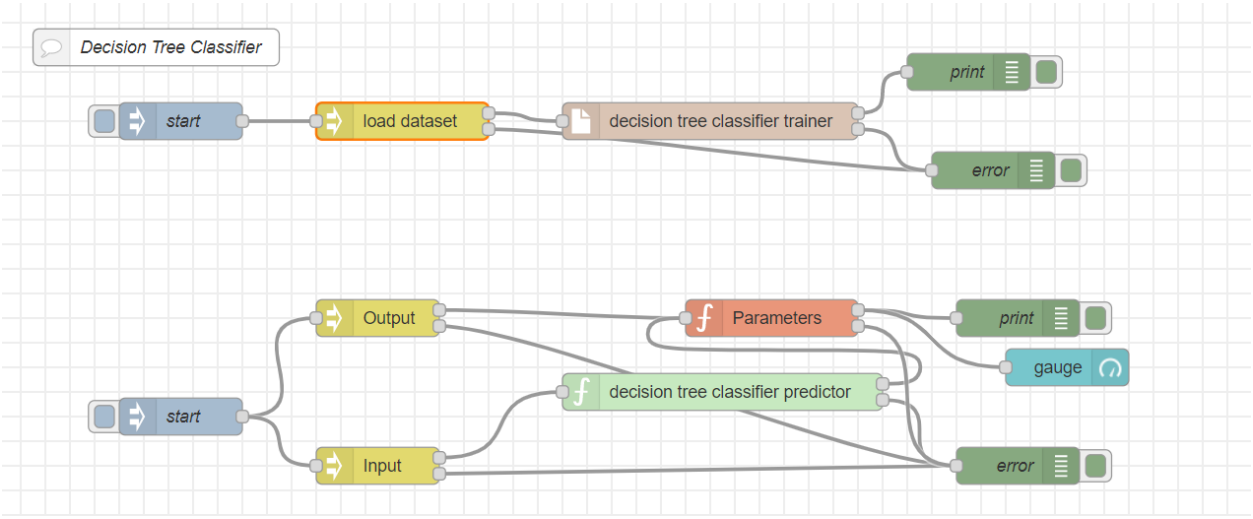
Machine learning classifiers trainer.py code:

```
import json
import pickle
import pandas
import os
import sys
sys.path.append(os.path.dirname(os.path.realpath(__file__)) + '/../../utils')
from sklw import SKLW

OUTLIER_DETECTORS = ['elliptic-envelope-classifier', 'isolation-forest-classifier', 'one-class-support-vector-classifier']

#read configurations
config = json.loads(input())
save = config['save']

while True:
        #read request
        data = input()
        try:
                #load data from request
                df = pandas.read_json(data, orient='values')
        except:
```

```
            #lead file specified in the request
            df = pandas.read_csv(json.loads(data)['file'], header=None)

        if config['classifier'] in OUTLIER_DETECTORS:
            x = df
            y = None
        else:
            x = df.iloc[:, :-1]
            y = df.iloc[:, -1]

        classifier = None

        if config['classifier'] == 'decision-tree-classifier':
            from sklearn.tree import DecisionTreeClassifier
            classifier = SKLW(path=save,
model=DecisionTreeClassifier(**config['kwargs']))
        elif config['classifier'] == 'deep-neural-network-classifier-tensorflow':
            from dnnctf import DNNCTF
            classifier = DNNCTF(path=save, del_prev_mod=True, **config['kwargs'])
        if config['classifier'] == 'elliptic-envelope-classifier':
            from sklearn.covariance import EllipticEnvelope
            classifier = SKLW(path=save, model=EllipticEnvelope(**config['kwargs']))
        if config['classifier'] == 'isolation-forest-classifier':
            from sklearn.ensemble import IsolationForest
            classifier = SKLW(path=save, model=IsolationForest(**config['kwargs']))
        elif config['classifier'] == 'k-neighbors-classifier':
            from sklearn.neighbors import KNeighborsClassifier
            classifier = SKLW(path=save,
model=KNeighborsClassifier(**config['kwargs']))
        elif config['classifier'] == 'multi-layer-perceptron-classifier':
            from sklearn.neural_network import MLPClassifier
            classifier = SKLW(path=save, model=MLPClassifier(**config['kwargs']))
        if config['classifier'] == 'one-class-support-vector-classifier':
            from sklearn.svm import OneClassSVM
            classifier = SKLW(path=save, model=OneClassSVM(**config['kwargs']))
        elif config['classifier'] == 'random-forest-classifier':
            from sklearn.ensemble import RandomForestClassifier
            classifier = SKLW(path=save,
model=RandomForestClassifier(**config['kwargs']))
        elif config['classifier'] == 'support-vector-classifier':
            from sklearn.svm import SVC
            classifier = SKLW(path=save, model=SVC(**config['kwargs']))

        try:
            #train model
            classifier.fit(x, y)
        except Exception as e:
            print(e)
            raise()
```

```
print(config['classifier'] + ': training completed.')
```

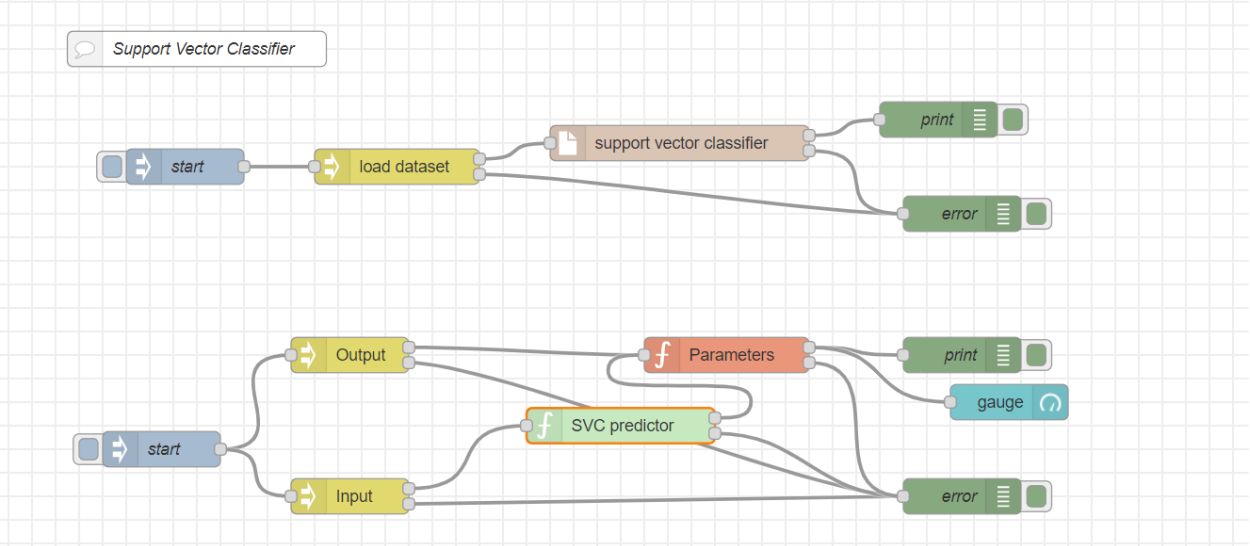
Decision tree classifier:



Json code:

```
module.exports = function(RED){
        function dTCNode(config){
                const path = require('path')
                const utils = require('../../../utils/utils')

                var node = this;

                //set configurations
                node.file = __dirname +  '/../trainer.py'
                node.config = {
                        classifier: 'decision-tree-classifier',
                        save: path.join(config.savePath, config.saveName),
                        kwargs: {
                                max_depth: parseInt(config.depth) || undefined,
                                criterion: config.criterion || undefined,
                                splitter: config.splitter || undefined
                        }
                }

                utils.run(RED, node, config)
        }
        RED.nodes.registerType("decision tree classifier", dTCNode)
}
```

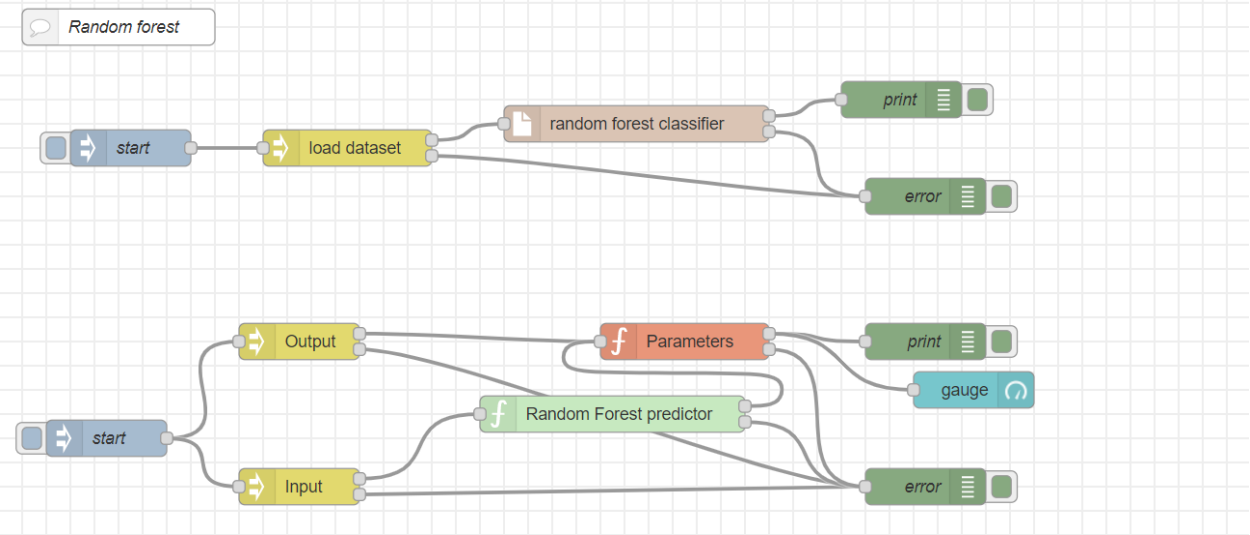
Support vector classifier:



Json code:

```
module.exports = function(RED){
        function sVCNode(config){
                const path = require('path')
                const utils = require('../../../utils/utils')

                var node = this;

                //set configurations
                node.file = __dirname +  '/../trainer.py'
                node.config = {
                        classifier: 'support-vector-classifier',
                        save: path.join(config.savePath, config.saveName),
                        kwargs: {
                                C: Number(config.c) || undefined,
                                kernel: config.kernel || undefined
                        }
                }

                utils.run(RED, node, config)
        }
        RED.nodes.registerType("support vector classifier", sVCNode);
}
```

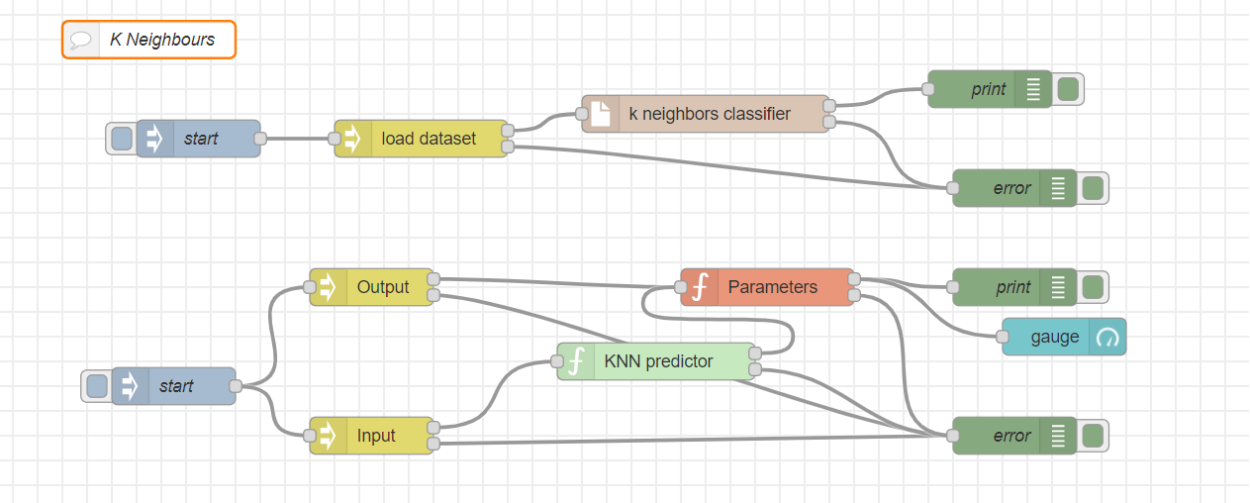
Random forest classifier:



Json code:

```
module.exports = function(RED){
        function rFCNode(config){
                const path = require('path')
                const utils = require('../../../utils/utils')

                var node = this;

                //set configurations
                node.file = __dirname +  '/../trainer.py'
                node.config = {
                        classifier: 'random-forest-classifier',
                        save: path.join(config.savePath, config.saveName),
                        kwargs: {
                                criterion: config.criterion,
                                max_depth: parseInt(config.maxDepth) || undefined,
                                n_estimators: parseInt(config.numTrees) || undefined
                        }
                }

                utils.run(RED, node, config)
        }
        RED.nodes.registerType("random forest classifier", rFCNode);
}
```

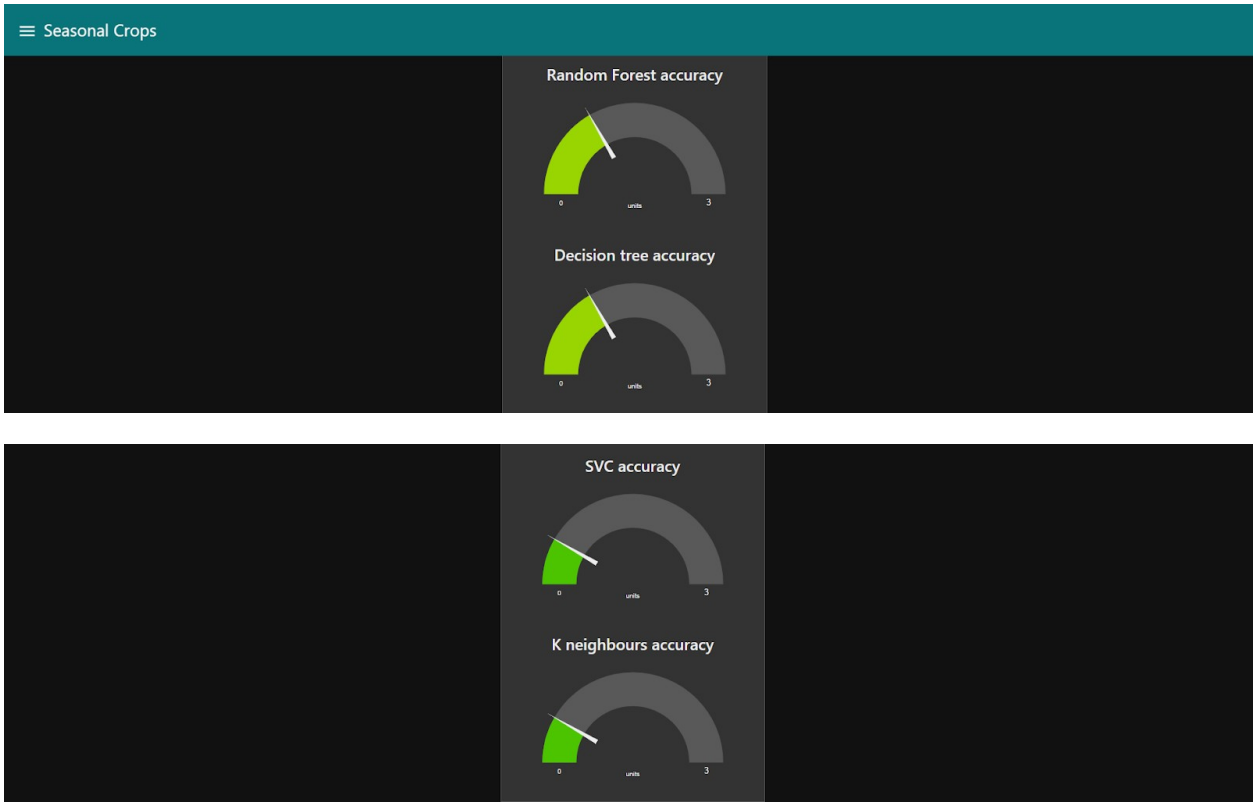
K Neighbors classifier:



Json code:

```
module.exports = function(RED){
        function kNCNode(config){
                const path = require('path')
                const utils = require('../../../utils/utils')

                var node = this;

                //set configurations
                node.file = __dirname +  '/../trainer.py'
                node.config = {
                        classifier: 'k-neighbors-classifier',
                        save: path.join(config.savePath, config.saveName),
                        kwargs: {
                                n_neighbors: parseInt(config.neighbors) || undefined,
                                weights : config.weights || undefined
                        }
                }

                utils.run(RED, node, config)
        }
        RED.nodes.registerType("k neighbors classifier", kNCNode);
}
```

Node-red dashboard-UI

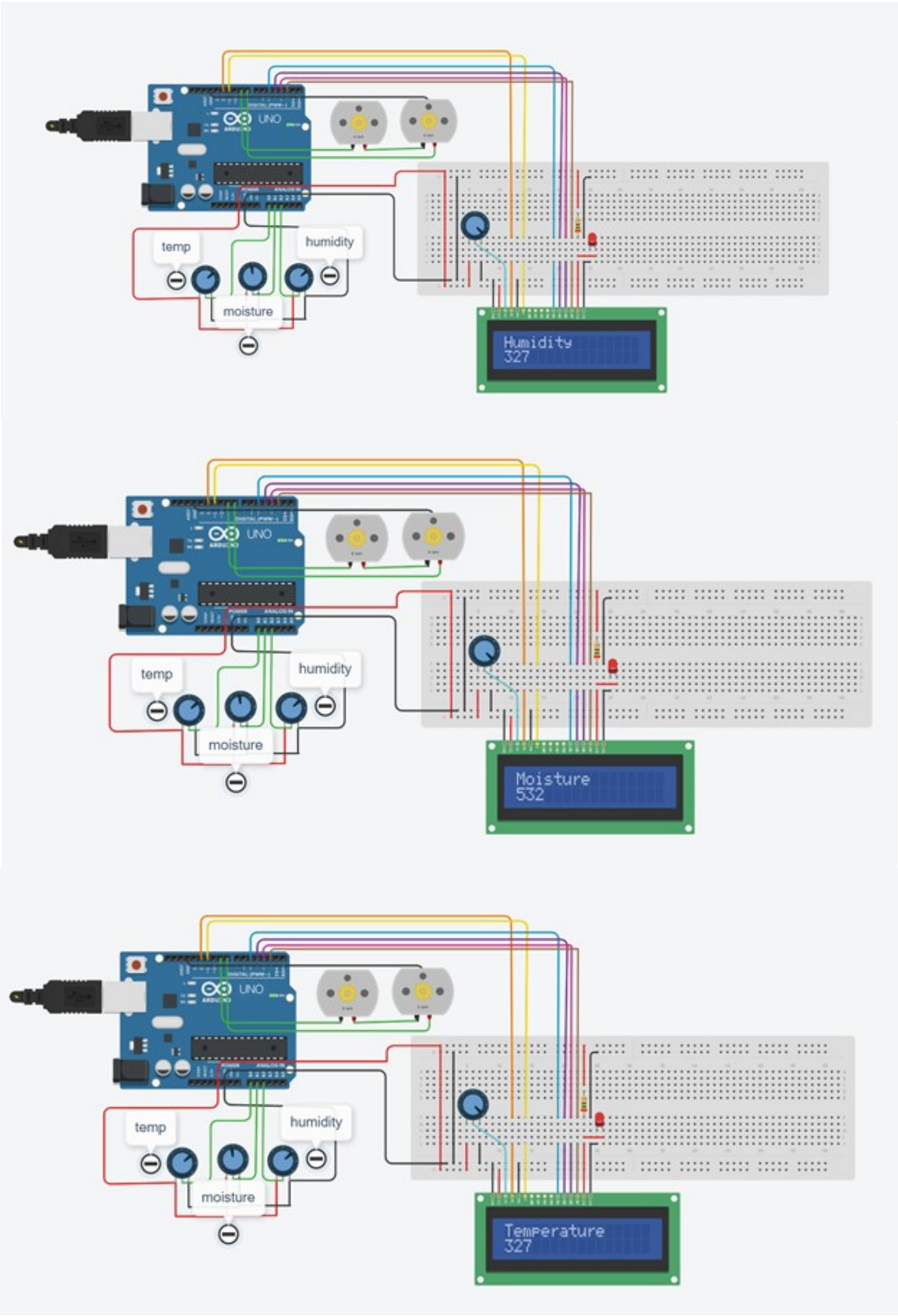


HARDWARE IMPLEMENTATION

Soil Moisture and Humidity Sensor

```
#include<LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
void setup() {
  pinMode(A0, INPUT);
  pinMode(A1, INPUT);
  pinMode(A2, INPUT);
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(10, OUTPUT);
  Serial.begin(9600);
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
}
void loop() {
  analogRead(A0);
  analogRead(A1);
  analogRead(A2);
  delay(100);
  Serial.print( "Temp Reading = ");
  Serial.println( analogRead(A0));
  lcd.setCursor(0, 0);
```

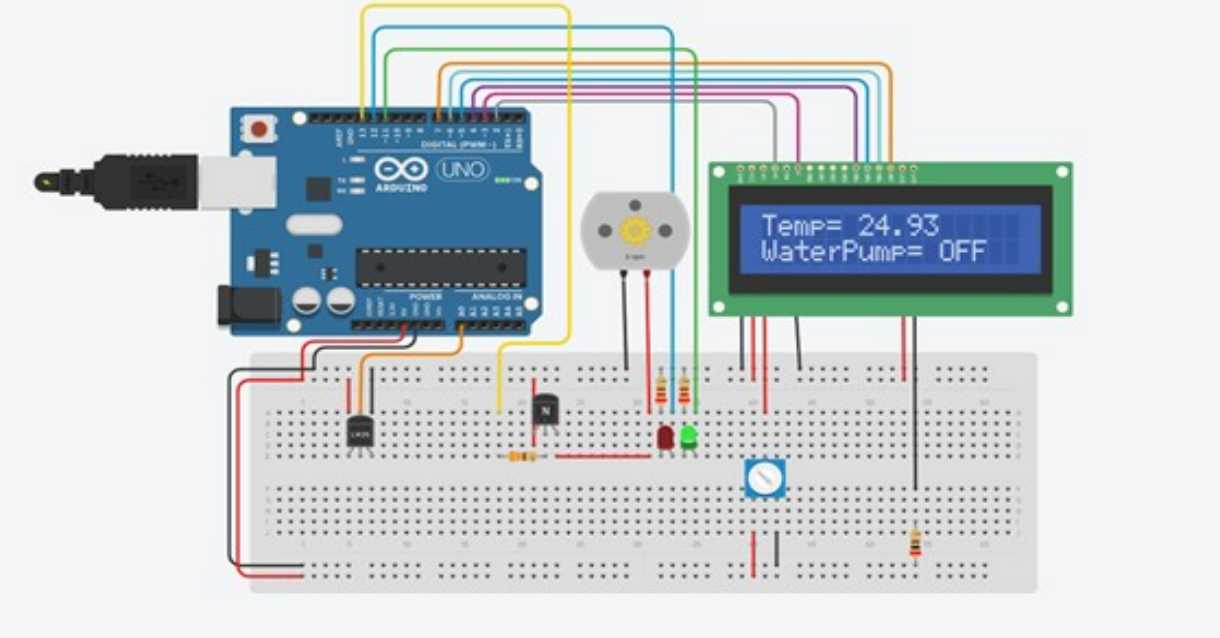
```
  lcd.print("Temperature");
  lcd.setCursor(0, 1);
  lcd.print(analogRead(A0));
  delay(1000);
  lcd.clear();
  if( analogRead(A0)>300)
  {
   digitalWrite(9,1);
   delay(1000);
  }
 digitalWrite(9,0);
  delay(1000);
Serial.print( "Moisture Reading = ");
Serial.println( analogRead(A1));
lcd.setCursor(0, 0);
  lcd.print("Moisture");
  lcd.setCursor(0, 1);
  lcd.print(analogRead(A1));
  delay(1000);
  lcd.clear();
if( analogRead(A1)>300)
 {
   digitalWrite(8,1);
   delay(1000);
 }
digitalWrite(8,0);
  delay(1000);
Serial.print( "Humidity Reading = ");
Serial.println( analogRead(A2));
lcd.setCursor(0, 0);
  lcd.print("Humidity");
  lcd.setCursor(0, 1);
  lcd.print(analogRead(A2));
  delay(1000);
  lcd.clear();
  delay(1000);
}
```

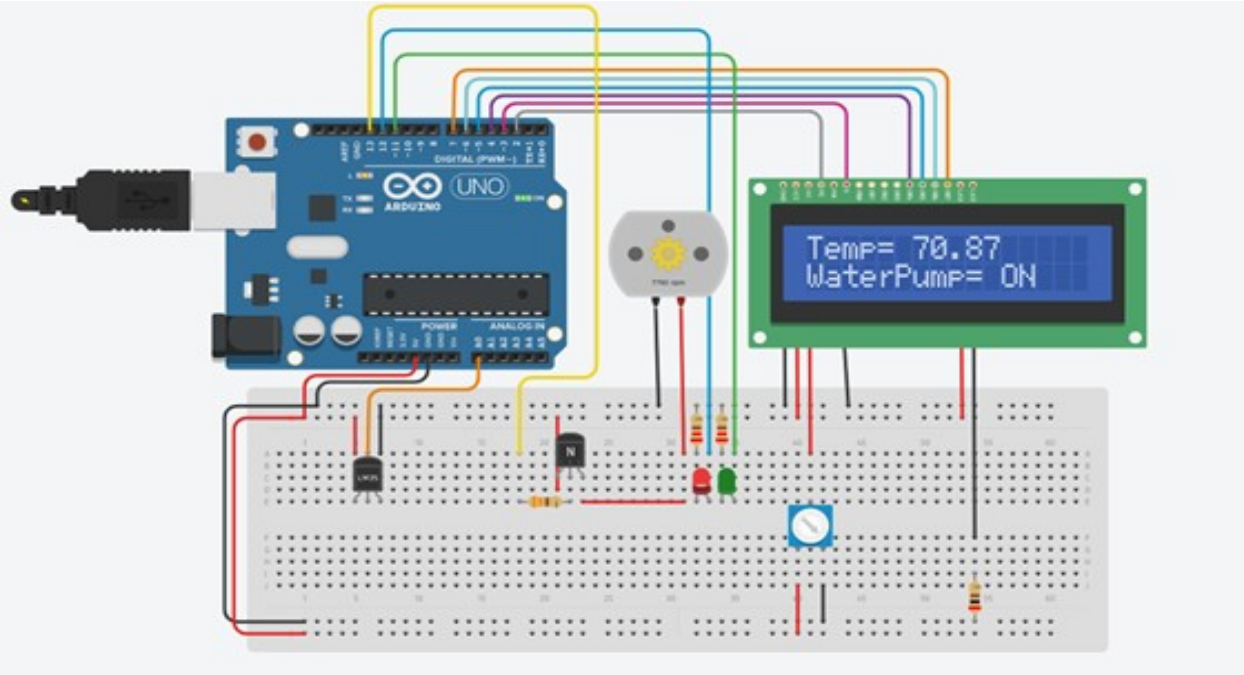


Automatic Water Pumping system by temperature reference:

```
#include <LiquidCrystal.h>
```

```
const int LM35 = A0;
const int motor = 13;
const int LedRed = 12;
const int LedGreen = 11;
LiquidCrystal lcd(2, 3, 4, 5, 6, 7);
void setup() {
  Serial.begin(9600);
  lcd.begin(16, 2);
  lcd.print("Automated Plant");
  lcd.setCursor(0,1);
  lcd.print("Watering System!");
  pinMode(motor, OUTPUT);
  pinMode(LedRed, OUTPUT);
  pinMode(LedGreen, OUTPUT);
  delay(2000);
  lcd.clear();
  lcd.print("Temp= ");
  lcd.setCursor(0,1);
  lcd.print("WaterPump= ");
}

void loop() {
  int value = analogRead(LM35);
  float Temperature = value * 500.0 / 1023.0;
  lcd.setCursor(6,0);
  lcd.print(Temperature);
  lcd.setCursor(11,1);
  if (Temperature > 50){
    digitalWrite(motor, HIGH);
    digitalWrite(LedRed, HIGH);
    digitalWrite(LedGreen, LOW);
    lcd.print("ON ");
  }
  else {
    digitalWrite(motor, LOW);
    digitalWrite(LedRed, LOW);
    digitalWrite(LedGreen, HIGH);
    lcd.print("OFF");
  }
delay(1000);
}
```

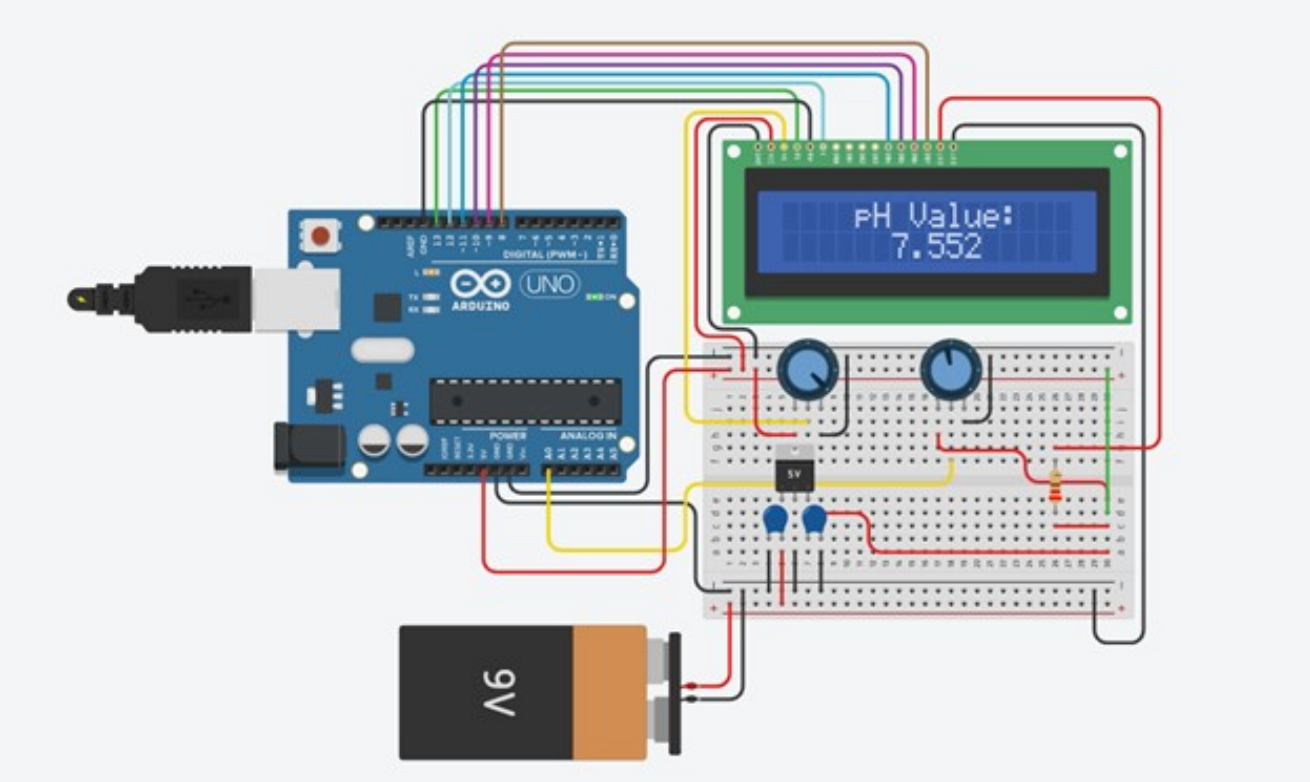




Soil pH-Value sensor

```
#include<LiquidCrystal.h>
const int rs =13,en = 12,d4 =11,d5 =10,d6 =9,d7 =8;
LiquidCrystal lcd(rs,en, d4,d5,d6,d7);
void setup()
{
  Serial.begin(9600);
  lcd.begin(16,2);
```

```
  lcd.setCursor(4,0);
  lcd.print("pH Value:");
}
void loop()
{
 int sensorValue = analogRead(A0);
 float ph = sensorValue * (14.0/1023.0);
 Serial.println(ph);
 lcd.setCursor(6,1);
 lcd.print (ph);
}
```

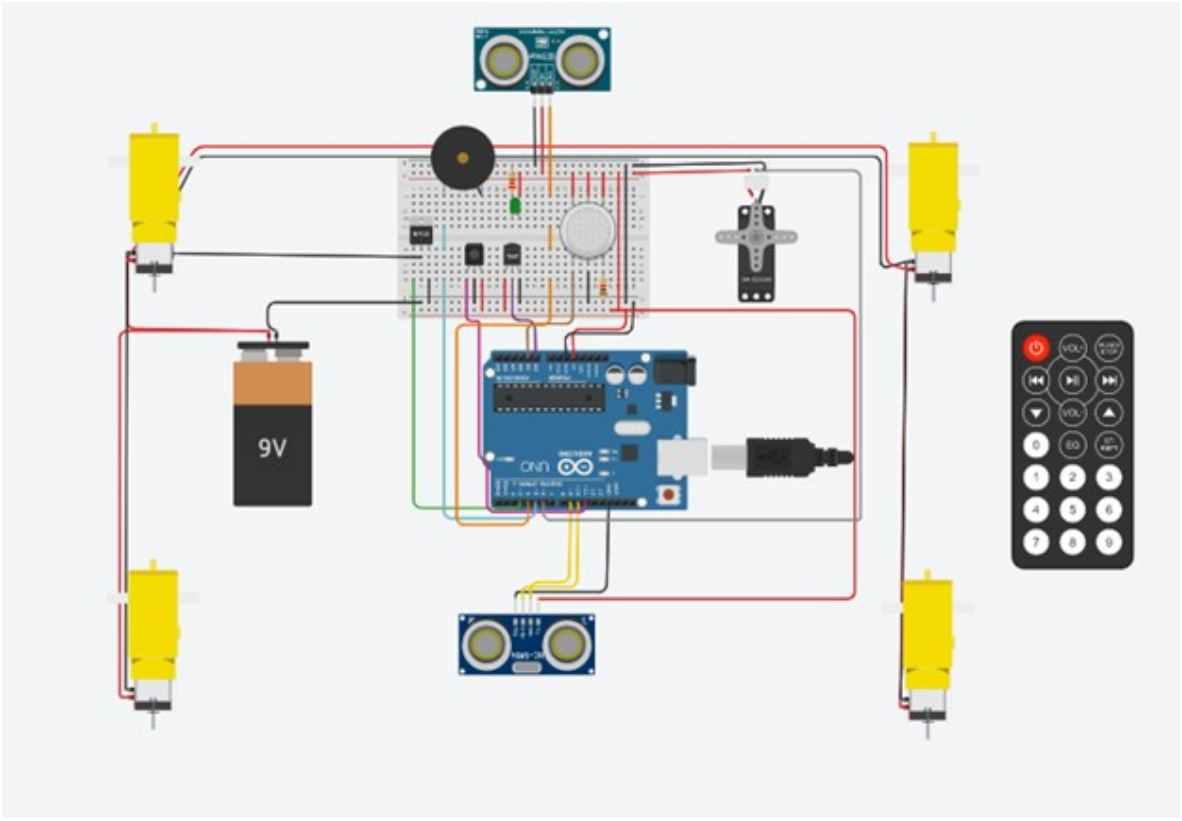


OVERALL COMBINED HARDWARE MODEL:

```
#include <IRremote.h>           //HEADER FILE - IR SENSOR
#include <Servo.h>
const int trigPin = 9;          //ULTRASONIC DECLARATION
const int echoPin = 10;
float duration, distance,temp;
int sensor_input,SensorRead;    //TEMP GAS DECLARATION
int buzzer = 5 ;
const int sensor = 4;           //PING SENSOR DECLARATION
int RECV_PIN = 11;
IRrecv irrecv(RECV_PIN);        //OBJECT FOR IR
```

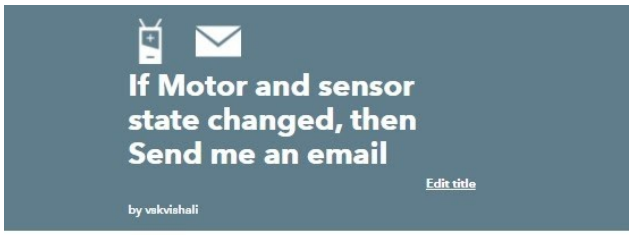
```
decode_results results;
int led = 7;                    //BULB DECLARATION
Servo servo_6;
void setup()
{
 pinMode(trigPin,OUTPUT);
 pinMode(echoPin,INPUT);
 pinMode(A1, INPUT);
 pinMode(buzzer, OUTPUT);
 irrecv.enableIRIn();           // Start the receiver
 Serial.begin(9600);
 servo_6.attach(6);
}
void loop()
{
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = (duration*.0343)/2;
  Serial.print("Distance: ");
  Serial.println(distance);
  delay(100);
  if(distance<30)
  {
  analogWrite(3,0);
  sensor_input = analogRead(A0);
  temp = (double)sensor_input/1024;
  temp = temp*5;
  temp = temp-0.5;
  temp = temp*100;
  Serial.print("CURRENT TEMPERATURE IS :");
  Serial.println(temp);
  if(temp<26)
  {
   Serial.println("Temperature level is Normal");
  }
  else
   {
   Serial.println("Temperature critical.Plant is suffering from disease");
   }
  SensorRead = analogRead(A1);
  if(SensorRead<100)
  {
   Serial.print(" CARBON DIOXIDE LEVEL IS NORMAL :");
   Serial.println(SensorRead);
   delay(10);
  }
```

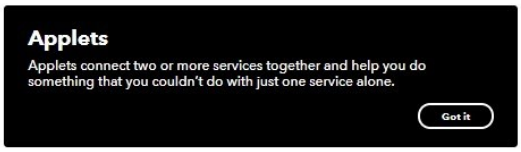
```
  else
  {
   Serial.println("CARBON DIOXIDE LEVEL:");
   Serial.println(SensorRead);
   Serial.println("HIGH!!");
  }
  }
  else if(distance>30)
  {
   analogWrite(3,255);
   digitalWrite(led,HIGH);
  }
  if (irrecv.decode(&results)) {
    Serial.println(results.value, HEX);
    irrecv.resume();                    //RESUME IR SENSING
    if(results.value==0xFD08F7)
    {
     long duration,cm;
     pinMode(sensor,OUTPUT);
     digitalWrite(sensor,LOW);
     delay(2);
     digitalWrite(sensor,HIGH);
     delay(5);
     digitalWrite(sensor,LOW);
     pinMode(sensor,INPUT);
     duration = pulseIn(sensor,HIGH);
     cm= microsecondsToCentimeters(duration);
     Serial.println(cm);
      if(cm<100)
      {
        digitalWrite(buzzer,HIGH);
        delay(500);
        digitalWrite(buzzer,LOW);
        delay(500);
      }
      else
      {
        digitalWrite(buzzer,LOW);
      }
  }
 }
  delay(100);
  analogWrite(6,255);
  delay(100);
}
long microsecondsToCentimeters(long microseconds)
{
  return microseconds/29;
}
```

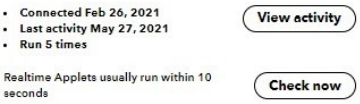


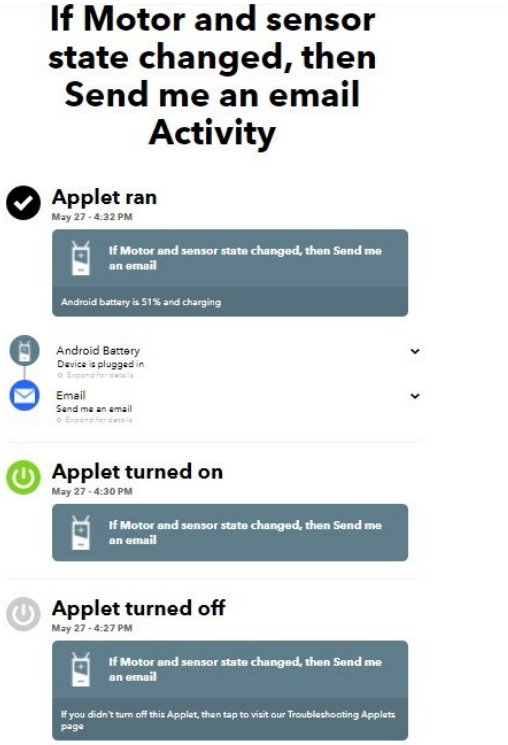

```
CURRENT TEMPERATURE IS :24.71
Temperature level is Normal
CARBON DIOXIDE LEVEL:
319
HIGH!!
Distance: 20.27
CURRENT TEMPERATURE IS :24.71
Temperature level is Normal
CARBON DIOXIDE LEVEL:
319
HIGH!!
Distance: 20.31
CURRENT TEMPERATURE IS :24.71
Temperature level is Normal
CARBON DIOXIDE LEVEL:
319
HIGH!!
Distance: 20.63
```

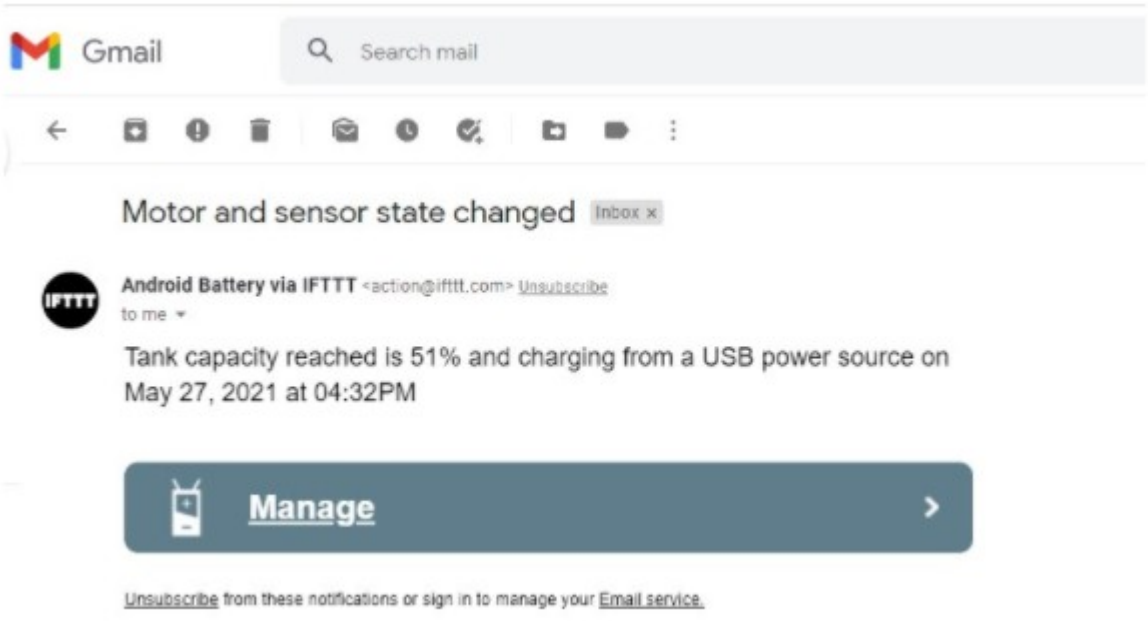
VISUALISATION OF DATA IN APP:











4.6 TESTING AND QUALITY ASSURANCE

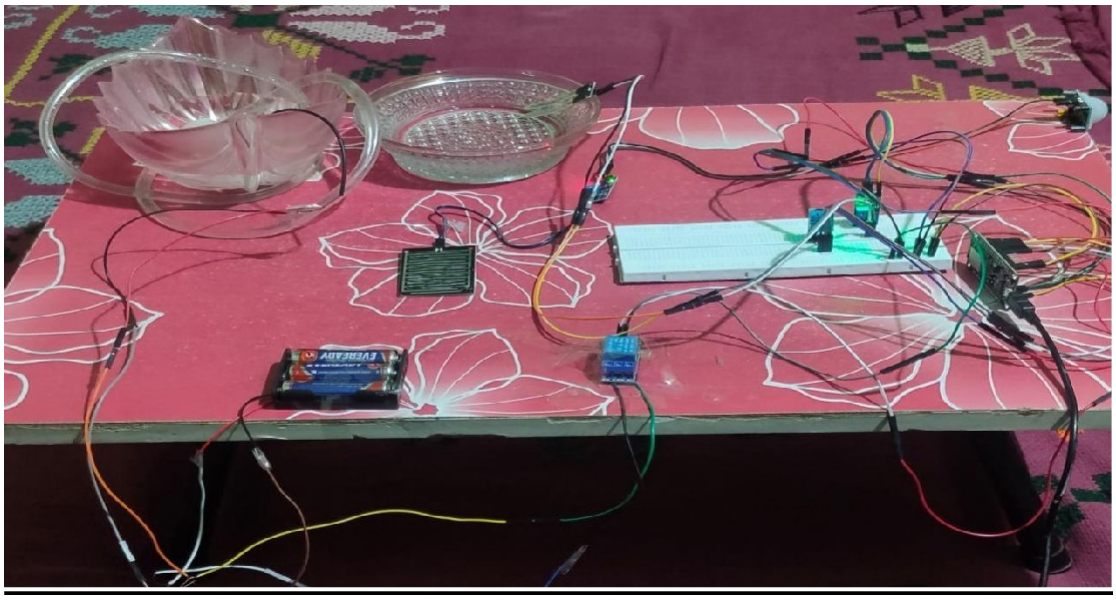
For fetching the weather data:
https://drive.google.com/file/d/15QfiH7OMkTeALUmrGpPsm-0edpAA-PNy/view?usp=sharing

For farmer registration and crop inputs followed by pH prediction:
https://drive.google.com/file/d/1naWSNhsIP38NAQR9l69ZGWOFm97IiqvE/view?usp=sharing

Machine learning models to compare accuracy:
https://drive.google.com/file/d/1NbF3YR4gHEcWHIdD3nTsmfUHykLquOPZ/view?usp=sharing

4.7 SNAPSHOTS-PROJECT , TEAM, RESULTS

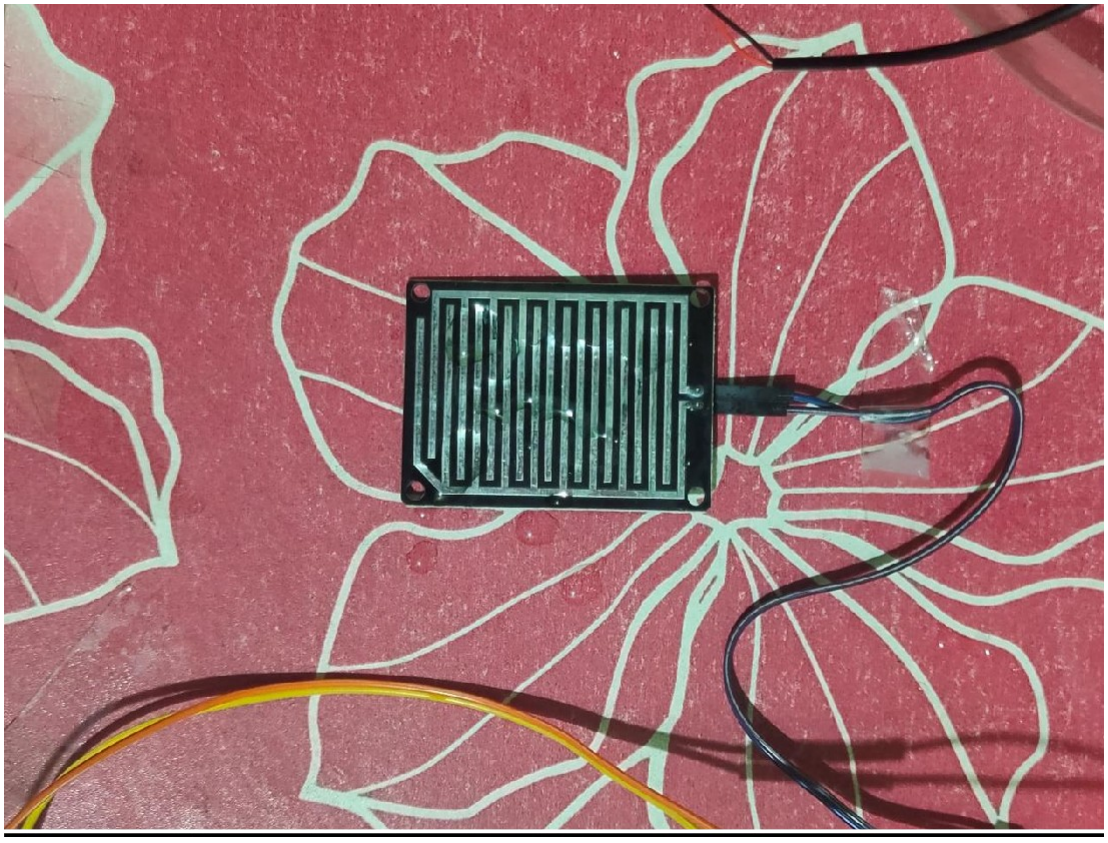
CIRCUIT



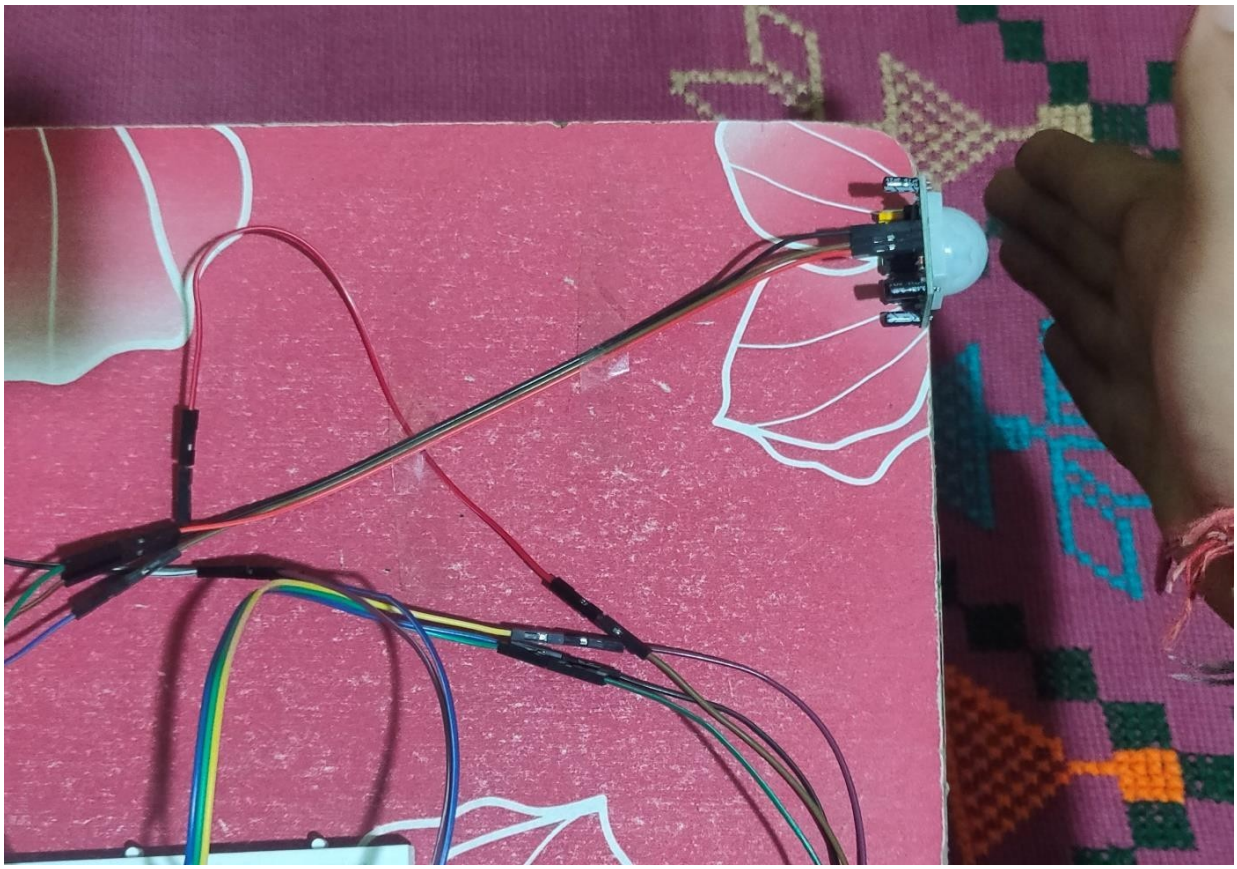
NOT WATERING:-

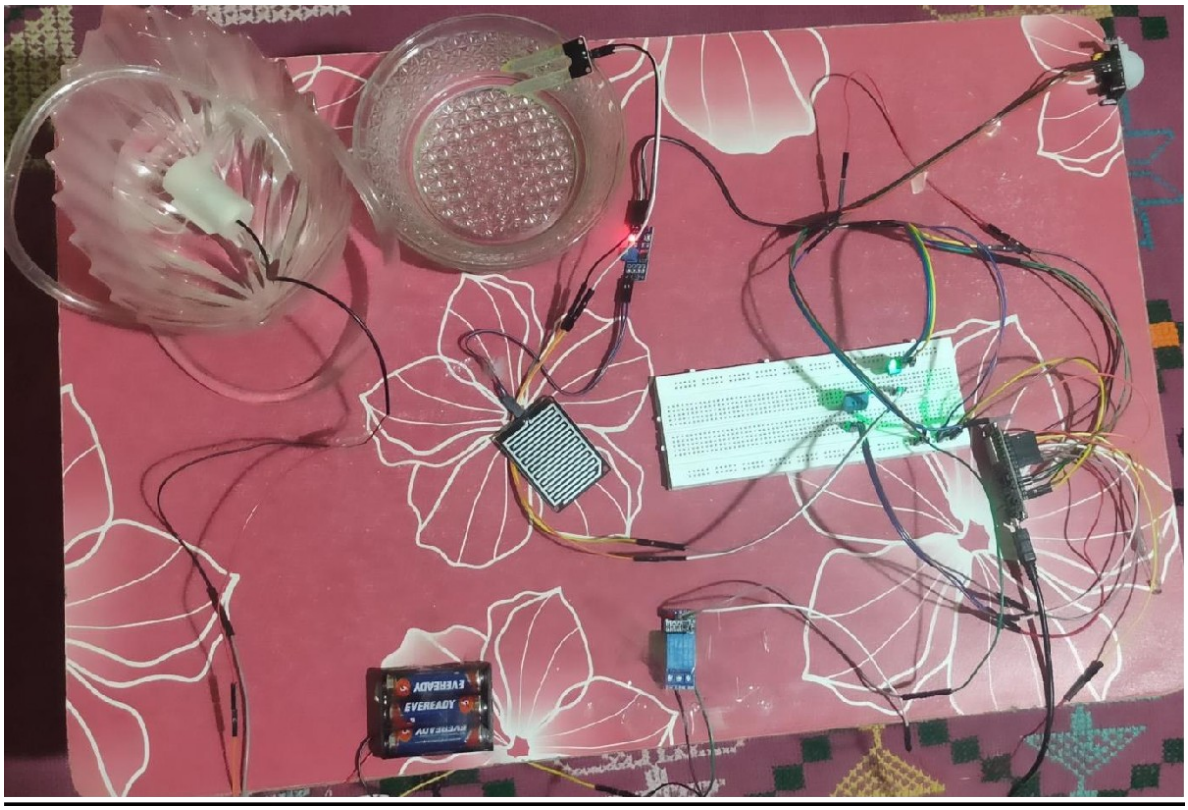
WATERING:-

RAINING:-



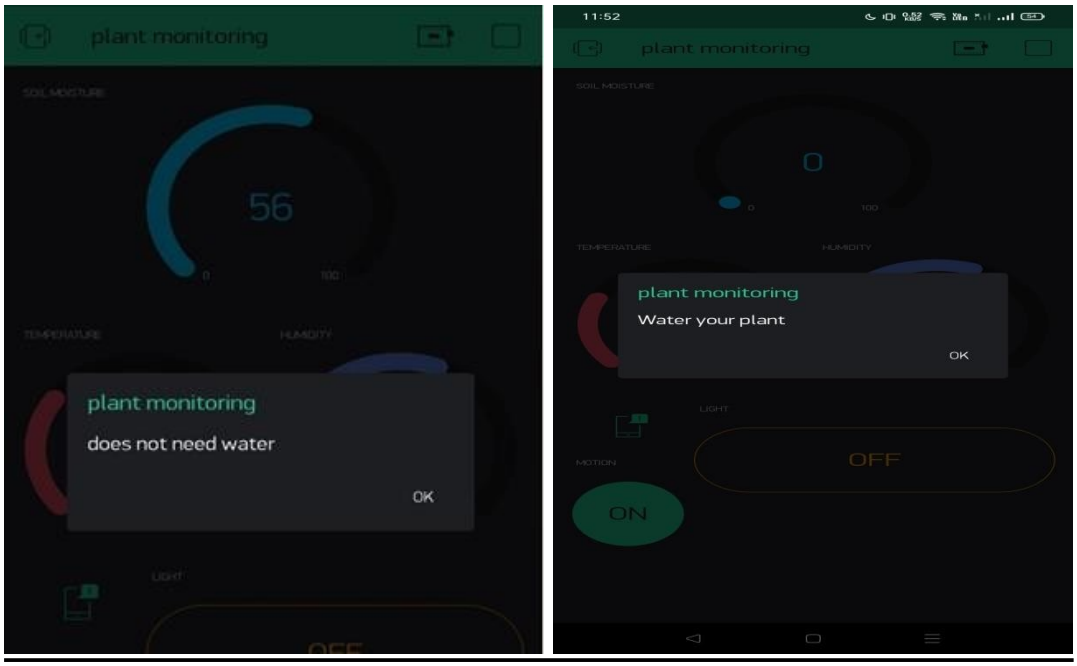
MOTION IN RANGE OF PIR SENSOR

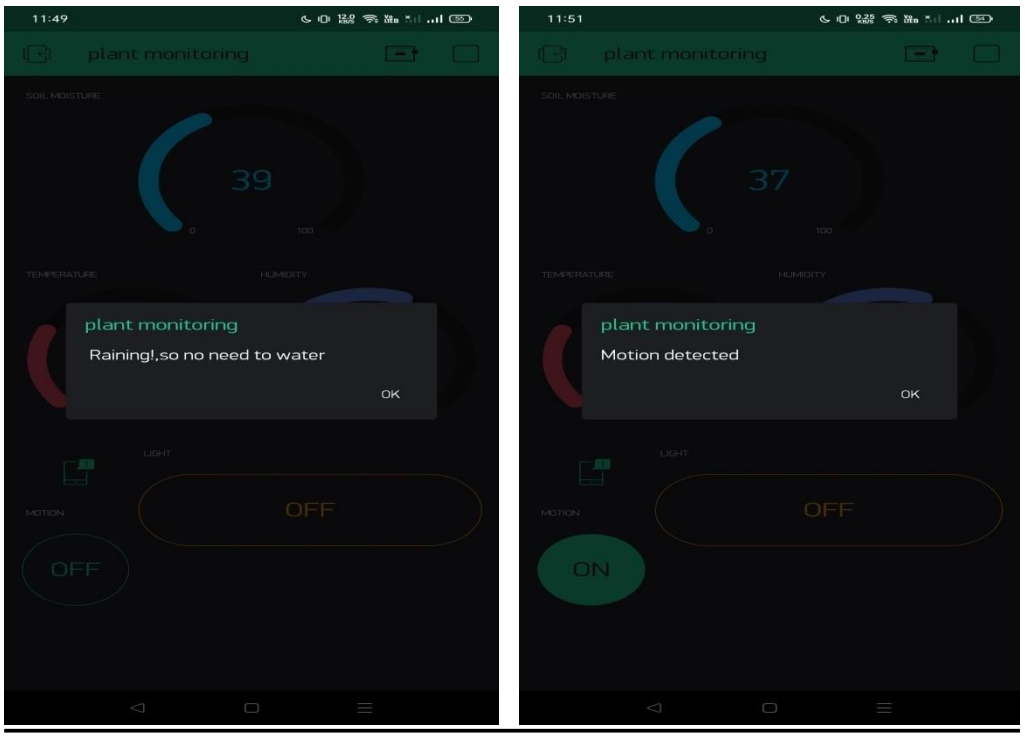


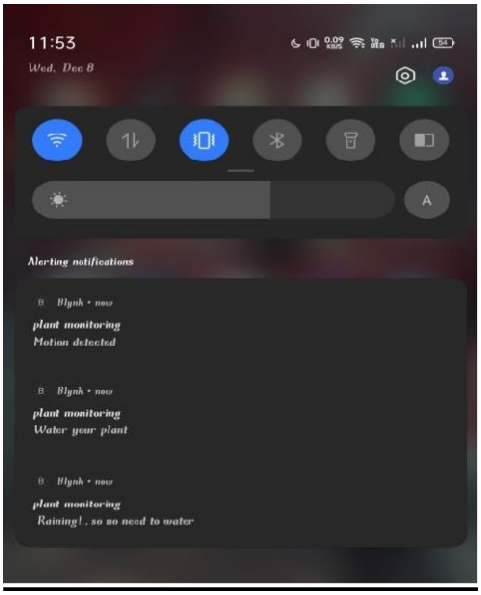


RESULT:-

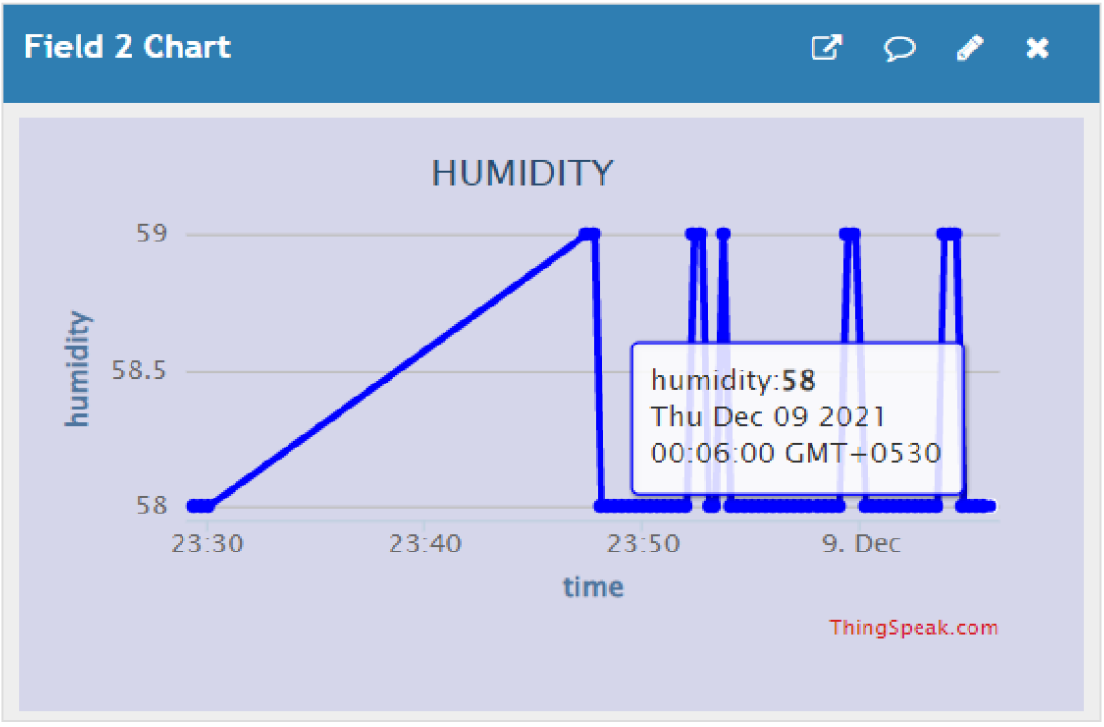
ON BLINK:-



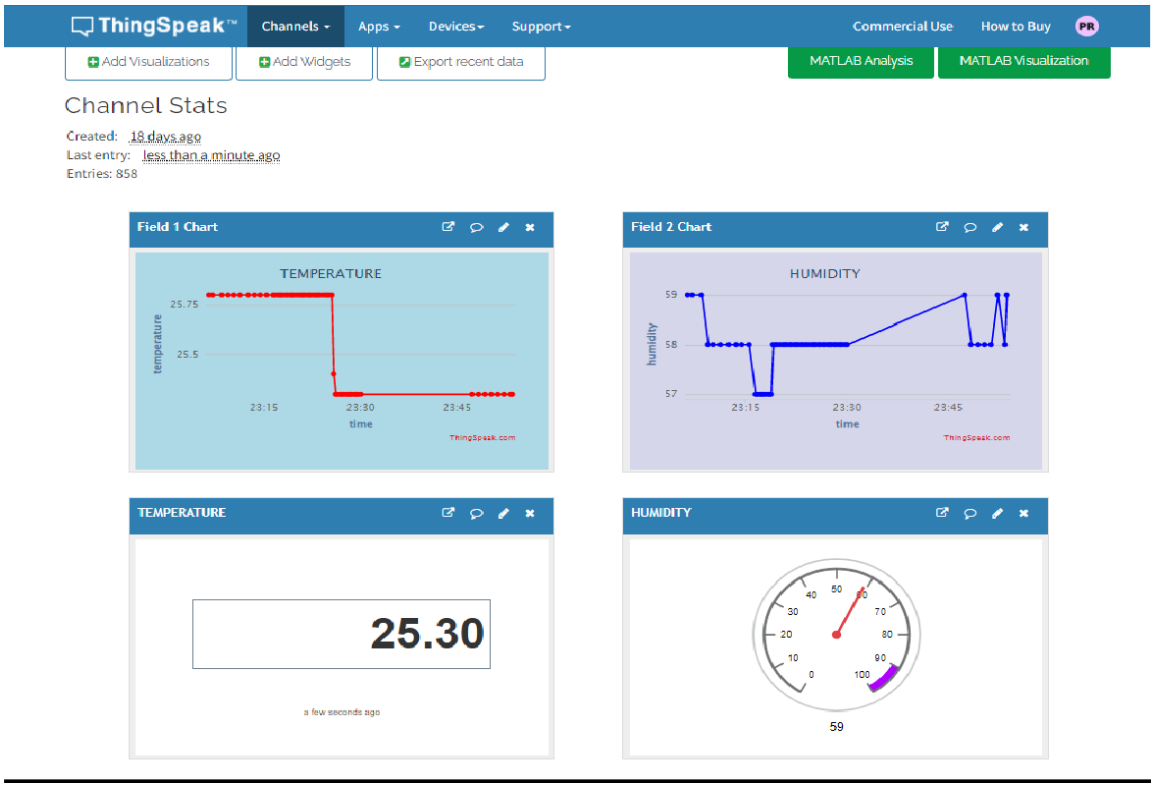


- 

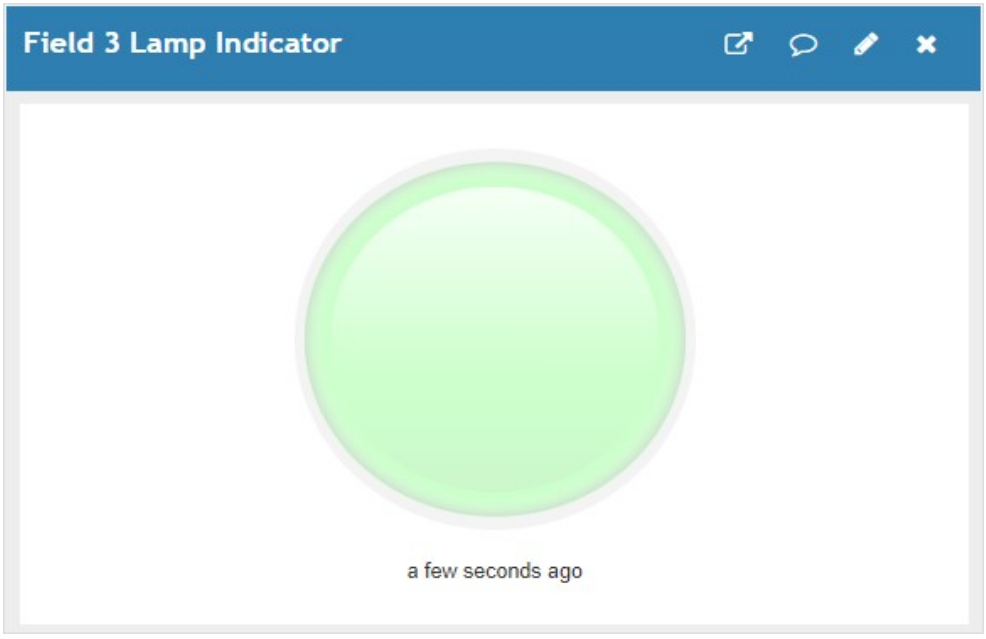
RESULT ON THINGSPEAK:-



-

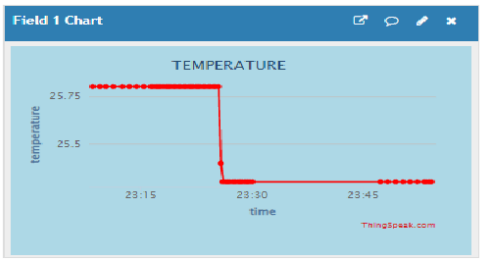

ThingSpeak
Channels
Apps
Devices
Support
Commercial Use
How to Buy
PR
Add Visualizations
Add Widgets
Export recent data
MATLAB Analysis
MATLAB Visualization
Channel Stats
Created: 18 days ago
Last entry: less than a minute ago
Entries: 858
Field 1 Chart
TEMPERATURE
temperature
25.75
25.5
23:15
23:30
23:45
time
ThingSpeak.com
Field 2 Chart
HUMIDITY
humidity
59
58
57
23:15
23:30
23:45
time
ThingSpeak.com
TEMPERATURE
25.30
a few seconds ago
HUMIDITY
0
10
20
30
40
50
60
70
80
90
100
59

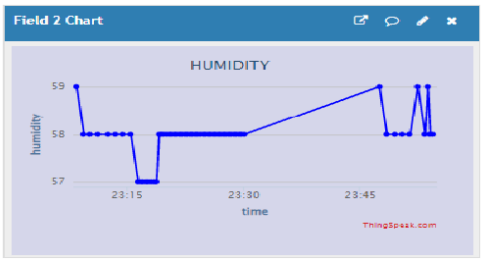
NO RAIN LAMP STATUS:-

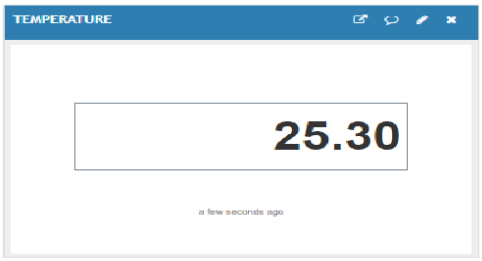


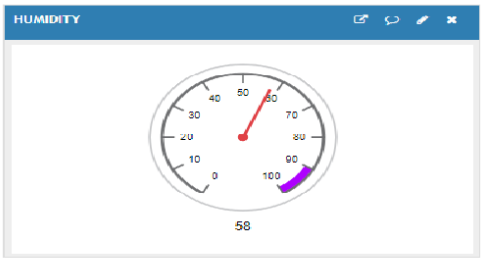
RAINING LAMP STATUS:-

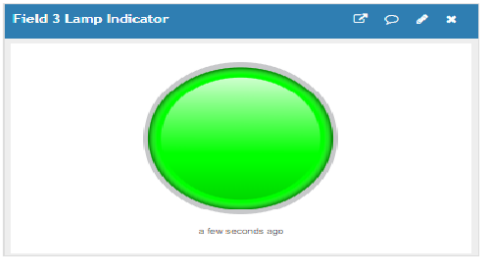
OVERALL THINGSPEAK RESULT:-











Working of Hardware:

https://drive.google.com/file/d/1-n1u9ykJ3z4K2LkjLvBgWG9FX4N7igk4/view?usp=sharing

4.8 SOFTWARE –CODING ON ARDUINO IDE AND ANALYSIS

```
#define BLYNK_PRINT Serial #include <OneWire.h> #include <SPI.h>
#include <BlynkSimpleEsp8266.h> #include <DHT.h>
#include <ESP8266WiFi.h> #include <DallasTemperature.h> #define ONE_WIRE_BUS D2
OneWire oneWire(ONE_WIRE_BUS); DallasTemperature sensors(&oneWire); String
apiKey = "NWDI3NPIBZJVJKFP";
char auth[] ="IM-hC4qKWItS_LbfxLXkdpmy2r9FD5SS"; char ssid[] = "iot";
char pass[] = "password";

const char* server = "api.thingspeak.com";


#define pirPin D1 int pirValue;
int pinValue; #define rainPin D5 #define DHTPIN 2
#define DHTTYPE DHT11 DHT dht(DHTPIN, DHTTYPE);
WiFiClient client; SimpleTimer timer;

const int pumpPin= 14 ;


BLYNK_WRITE(V0)

{

pinValue = param.asInt();

}

void sendSensor()

{

float h = dht.readHumidity(); float t = dht.readTemperature(); Serial.println("temperature
is:"); Serial.println(t); Serial.println("humidity is:"); Serial.println(h);
if (isnan(h) || isnan(t))

{

Serial.println("Failed to read from DHT sensor!"); return;
}

Blynk.virtualWrite(V5, h); //V5 is for Humidity Blynk.virtualWrite(V6, t); //V6 is for
Temperature
}

void setup()

{
```

```
pinMode(pumpPin, OUTPUT); Serial.begin(115200); delay(10);

Blynk.begin(auth, ssid, pass); pinMode(rainPin, INPUT); pinMode(pirPin, INPUT);
dht.begin(); Serial.println("Connecting to "); Serial.println(ssid); WiFi.begin(ssid, pass);
while (WiFi.status() != WL_CONNECTED)

{

delay(500); Serial.print(".");
}

Serial.println(""); Serial.println("WiFi connected"); timer.setInterval(1000L, sendSensor);
Serial.begin(115200); Blynk.begin(auth, ssid, pass); sensors.begin();
}

int sensor=0; int sensor1=0; int output=0;

void sendTemps()

{

sensor=analogRead(A0);

output=(100-map(sensor,0,1023,0,100));

sensors.requestTemperatures();

float temp = sensors.getTempCByIndex(0); Serial.print("moisture = "); Serial.print(output);
Serial.println("%"); Blynk.virtualWrite(V1, temp); Blynk.virtualWrite(V2,output);
delay(1000);
}

void getPirValue(void)          //Get PIR Data

{

pirValue = digitalRead(pirPin); if (pirValue)
{

Serial.println("Motion detected"); Blynk.notify("Motion detected");
}

}

void loop()

{

Blynk.run();
```

```
timer.run(); sendTemps();
if (output<=10) { digitalWrite(pumpPin,HIGH); Serial.println("needs water, send
notification"); Blynk.notify("Water your plant");

delay(100);

//send notification


}

else { digitalWrite(pumpPin,LOW);
Serial.println("does not need water");

// Blynk.notify("does not need water"); delay(100);
}


int rainState = digitalRead(rainPin); Serial.println(rainState);

if (rainState == 0) { Serial.println(" Raining!");
Blynk.notify(" Raining!,so no need to water"); delay(1000);
//send notification

}

else if (rainState == 1) { Serial.println("Not Raining"); delay(1000);
}

else { delay(100);
}

if (pinValue == HIGH)

{

getPirValue();

}

float h = dht.readHumidity(); float t = dht.readTemperature(); if (isnan(h) || isnan(t))
{

Serial.println("Failed to read from DHT sensor!"); return;
}

if (client.connect(server,80)) // "184.106.153.149" or api.thingspeak.com

{
```

```
String postStr = apiKey; postStr +="&field1="; postStr += String(t); postStr +="&field2=";
postStr += String(h); postStr +="&field3=";
postStr += String(rainState); postStr += "\r\n\r\n";
client.print("POST /update HTTP/1.1\n"); client.print("Host: api.thingspeak.com\n");
client.print("Connection: close\n");
client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n"); client.print("Content-Type:
application/x-www-form-urlencoded\n"); client.print("Content-Length: ");

client.print(postStr.length()); client.print("\n\n"); client.print(postStr);
Serial.print("Temperature: "); Serial.print(t);
Serial.print(" degrees Celcius, Humidity: "); Serial.print(h);
Serial.println("%. Send to Thingspeak."); Serial.print(" rain-staus "); Serial.print(rainState);
Serial.println("%. Send to Thingspeak.");

}

client.stop(); Serial.println("Waiting...");
// thingspeak needs minimum 15 sec delay between updates delay(1000);
}
```

SNALPSHOTS OF CODING AND RESULT

plant1 | Arduino 1.8.16 (Windows Store 1.8.51.0)

File Edit Sketch Tools Help

plant1

```
#define BLYNK_PRINT Serial
#include <OneWire.h>
#include <SPI.h>
#include <BlynkSimpleEsp8266.h>
#include <DHT.h>
#include <ESP8266WiFi.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS D2
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
String apiKey = "NWDI3NPIBZJVJKFP";
char auth[] ="IM-hC4qKWItS_LbfxLXkdpmy2r9FD5SS";
char ssid[] = "iot";
char pass[] = "password";
const char* server = "api.thingspeak.com";

#define pirPin D1
int pirValue;
int pinValue;
#define rainPin D5
#define DHTPIN 2
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);
WiFiClient client;
SimpleTimer timer;
const int pumpPin= 14 ;

BLYNK_WRITE(V0)
{
 pinValue = param.asInt();
}

void sendSensor()
```

plant1 | Arduino 1.8.16 (Windows Store 1.8.51.0)

File Edit Sketch Tools Help

plant1

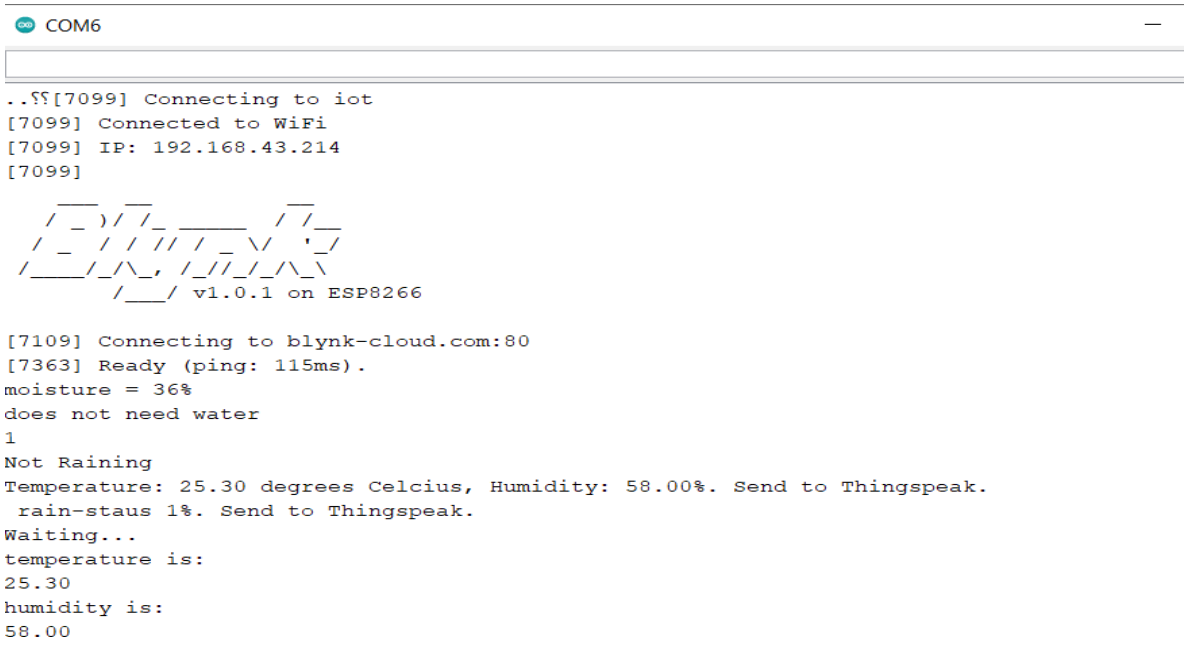
```
{
float h = dht.readHumidity();
float t = dht.readTemperature();
Serial.println("temperature is:");
Serial.println(t);
Serial.println("humidity is:");
Serial.println(h);
if (isnan(h) || isnan(t))
{
Serial.println("Failed to read from DHT sensor!");
return;
}

Blynk.virtualWrite(V5, h); //V5 is for Humidity
Blynk.virtualWrite(V6, t); //V6 is for Temperature
}

void setup()
{
  pinMode(pumpPin, OUTPUT);

Serial.begin(115200);
delay(10);
Blynk.begin(auth, ssid, pass);
 pinMode(rainPin, INPUT);
   pinMode(pirPin, INPUT);
dht.begin();
 Serial.println("Connecting to ");
       Serial.println(ssid);


       WiFi.begin(ssid, pass);
```

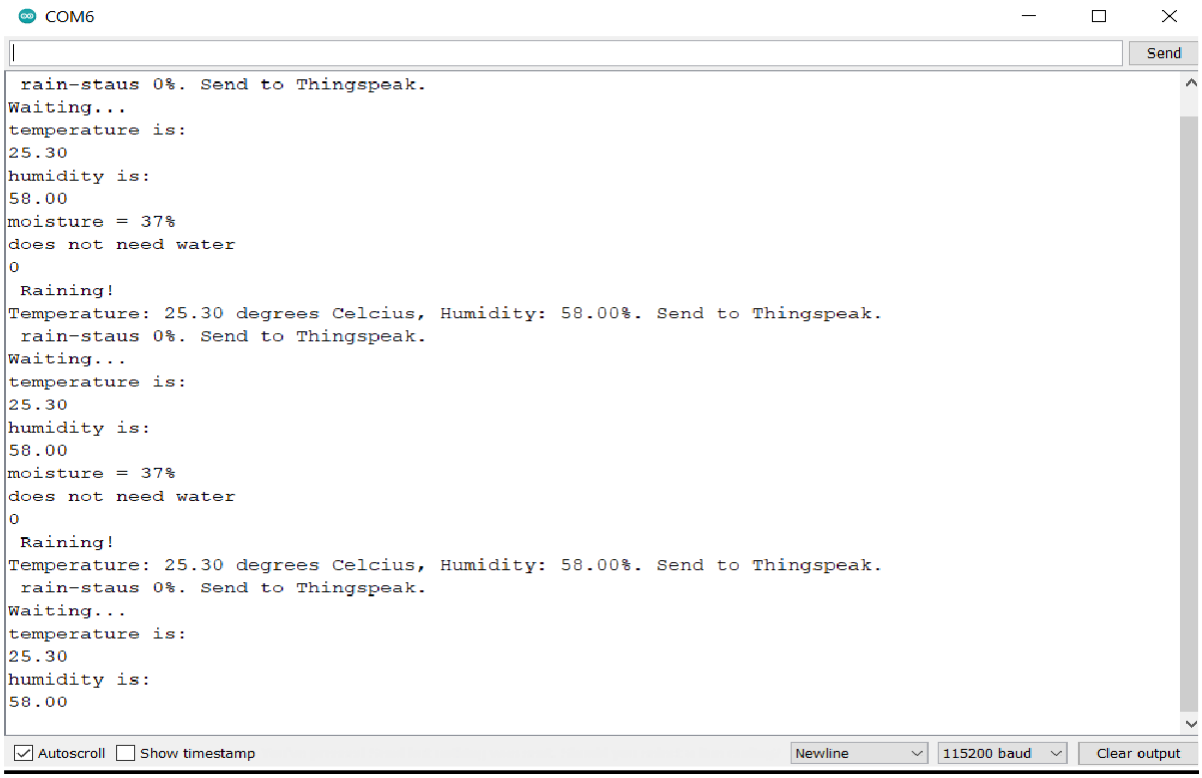
```
Serial.begin(115200);
delay(10);
Blynk.begin(auth, ssid, pass);
 pinMode(rainPin, INPUT);
   pinMode(pirPin, INPUT);
dht.begin();
 Serial.println("Connecting to ");
        Serial.println(ssid);


        WiFi.begin(ssid, pass);

       while (WiFi.status() != WL_CONNECTED)
      {
             delay(500);
             Serial.print(".");
      }
       Serial.println("");
       Serial.println("WiFi connected");
timer.setInterval(1000L, sendSensor);
Serial.begin(115200);
Blynk.begin(auth, ssid, pass);
sensors.begin();
}

int sensor=0;
int sensor1=0;
int output=0;


void sendTemps()
{
sensor=analogRead(A0);
output=(100-map(sensor,0,1023,0,100));
sensors.requestTemperatures();
float temp = sensors.getTempCByIndex(0);
Serial.print("moisture = ");
Serial.print(output);
Serial.println("%");
Blynk.virtualWrite(V1, temp);
Blynk.virtualWrite(V2,output);
delay(1000);
}

void getPirValue(void)          //Get PIR Data
  {
   pirValue = digitalRead(pirPin);
    if (pirValue)
     { Serial.println(pirValue);
       Serial.println("Motion detected");
       Blynk.notify("Motion detected");
     }
  }

void loop()
{
Blynk.run();
timer.run();
sendTemps();
if (output<=10) {
 digitalWrite(pumpPin,HIGH);
  Serial.println("needs water, send notification");
  Blynk.notify("Water your plant");
```

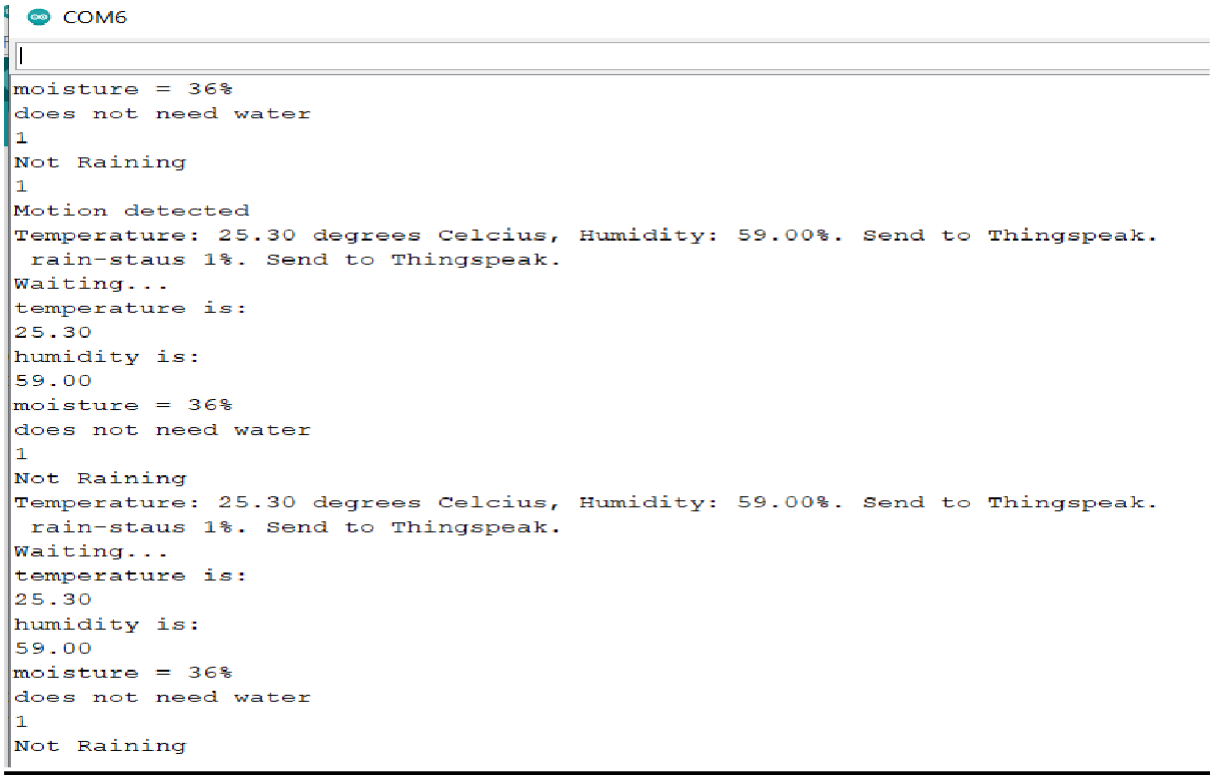
```
  delay(100);
//send notification

  }
  else {
    digitalWrite(pumpPin,LOW);
    Serial.println("does not need water");
    delay(100);
  }

int rainState = digitalRead(rainPin);
Serial.println(rainState);

  if (rainState == 0) {
  Serial.println(" Raining!");
  Blynk.notify(" Raining!,so no need to water");
  delay(1000);
//send notification

  }
  else if (rainState == 1) {
    Serial.println("Not Raining");
  delay(1000);
  }
  else {
    delay(100);
  }

  if (pinValue == HIGH)
       {
         getPirValue();
       }


       if (isnan(h) || isnan(t))
          {
              Serial.println("Failed to read from DHT sensor!");
               return;
          }

                 if (client.connect(server,80))   //   "184.106.153.149" or api.thingspeak.com
              {

                    String postStr = apiKey;
                    postStr +="&field1=";
                    postStr += String(t);
                    postStr +="&field2=";
                    postStr += String(h);
                    postStr +="&field3=";
                    postStr += String(rainState);
                    postStr += "\r\n\r\n";

                    client.print("POST /update HTTP/1.1\n");
                    client.print("Host: api.thingspeak.com\n");
                    client.print("Connection: close\n");
                    client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
                    client.print("Content-Type: application/x-www-form-urlencoded\n");
                    client.print("Content-Length: ");
                    client.print(postStr.length());
                    client.print("\n\n");
                    client.print(postStr);

                    Serial.print("Temperature: ");
                    Serial.print(t);
                    Serial.print(" degrees Celcius, Humidity: ");
                    Serial.print(h);
                    Serial.println("%. Send to Thingspeak.");
```

```
                                Serial.print("Temperature: ");
                                Serial.print(t);
                                Serial.print(" degrees Celcius, Humidity: ");
                                Serial.print(h);
                                Serial.println("%. Send to Thingspeak.");
                                Serial.print(" rain-staus ");
                                Serial.print(rainState);
                                Serial.println("%. Send to Thingspeak.");

                        }
          client.stop();

          Serial.println("Waiting...");

  // thingspeak needs minimum 15 sec delay between updates
  delay(1000);


}
```

RESULT ON SERIAL MONITOR:-







TECHNICAL ANALYSIS

Time complexity:

In the model we designed, different basic algorithms in software implementation and a usual algorithm for hardware implementation are used.

These algorithms decide the time complexity of the overall model, so we can achieve the final time complexity value of the overall model by calculating the time complexity of each algorithm and combining them to get the final one.

SOFTWARE:

- **KNN - n**= number of training examples, d= number of dimensions of the data= number of neighbors - **O(k*n*d)**
- **Decision Tree -** n= number of training examples, d= number of dimensions of the data= number of neighbors- **O(n*log(n)*d)**
- **Random Forest -** n= number of training examples, d= number of dimensions of the data= number of neighbors - **O(n*log(n)*d*k)**
- **SVM-** n= number of training examples, d= number of dimensions of the data= number of neighbors - **O(n^2)**

HARDWARE - n= number of training examples - **O(n)**

- **Overall time complexity of the model is -** n= number of training examples **O(n*log(n))**

Space Complexity:

In our project we implemented the hardware part individually i.e each sensor has been designed as a separate system and each system is sending data back for further analysis. This might lead to more transmission traffic and memory usage. So as part of the future work on this project we decided to design the whole circuit through one common board in order to minimize the space complexity of the model. This can be done using System-on-chip technology in real time implementation.

5.1 TECHNOLOGY USED

IOT

The Internet of Things (IoT) refers to a system of interrelated, internet-connected objects that are able to collect and transfer data over a wireless network without human intervention. The personal or business possibilities are endless. A 'thing' can refer to a connected medical device, a biochip transponder (think livestock), a solar panel, a connected automobile with sensors that alert the driver to a myriad of possible issues (fuel, tire pressure, needed maintenance, and more) or any object, outfitted with sensors, that has the ability to gather and transfer data over a network.

NODE-RED

Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways. It provides a browser-based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click. Node-RED provides a browser-based flow editor that makes it easy to wire together flows using the wide range of nodes in the palette. Flows can be then deployed to the runtime in a single-click. JavaScript functions can be created within the editor using a rich text editor. A built-in library allows you to save useful functions, templates or flows for re-use.

BIG DATA CONCEPTS:

- **DECISION TREE**
  Decision tree algorithms fall under the category of supervised learning. They can be used to solve both regression and classification problems. Decision trees use the tree representation to solve the problem in which each leaf node corresponds to a class label and attributes are represented on the internal node of the tree. We can represent any Boolean function on discrete attributes using the decision tree.
- **RANDOM FOREST**
  A Random Forest is an ensemble technique capable of performing both

regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as **bagging**. The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees.

- **SVM**

  Support Vector Machine (SVM) is a relatively simple **Supervised Machine Learning Algorithm** used for classification and/or regression. It is more preferred for classification but is sometimes very useful for regression as well. Basically, SVM finds a hyper-plane that creates a boundary between the types of data.

5.2 STANDARDS USED

There are many IoT communication protocols which are widely used in agriculture for the purpose of smart farming. By using these protocols farmers can communicate in a more convenient way and make more efficient decisions for smart farming to enhance and monitor the growth of crop. Most common wireless protocols which are being used named: IEEE 802.11 WIFI, 2G/3G/4 G-Mobile Communications Standards, LoraWan, WiMax, Low-Rate Wireless Personal Area Networks, Bluetooth, RFID, and ZigBee.

STANDARDS

Data Acquisition is further divided into two sub components namely: IoT data acquisition and standard data acquisition.

IoT data acquisition protocols

- Message Queuing Telemetry Transport (MQTT)
- Websocket
- Advanced Message Queuing Protocol (AMQP)
- Node
- Constrained Application Protocol (CoAP)
- Data Distribution Service (DDS)
- HyperText Transfer Protocol (HTTP)

Standard data acquisition protocols

- ZigBee
- WIFI
- Long Range Wide Area Network (LoraWan)
- ISOBUS
- 2G/3G/4G-MOBILE COMMUNICATIONS STANDARDS

STANDARDS USED

- IEEE 802.15.4
- IEEE 802.11(802.11a, 802.11b, 802.11g, 802.11n and 802.11ac)
- IEEE 802.16
- IEEE 802.15.1
- LoRaWAN R1.0

Arduino Uno

- It includes libraries “ADXL335.h” - to control Grove - 3-Axis Analog Accelerometer ADXL335, “Wire.h” - to communicate with I2C / TWI devices, “SPI.h” - to communicate with SPI devices, “Adafruit_BMP280.h” - for BMP280 pressure and altitude sensors, “dht.h” - for DHT11, DHT22, Temp & Humidity Sensors.
- The baud rate is set as “115200”.

MQTT

MQTT is a messaging protocol in IoT which is mainly designed for remote connections. It’s a bandwidth efficient protocol and uses little battery power. MQTT is used for continuous analysis and deploying a smart system for the agriculture sector. A low-cost web based IoT solution has been presented by using MQTT for monitoring, tracking and analyzing agricultural data and collecting knowledge from field ambience and improving environmental conditions. By using MQTT a low-cost irrigation system has been proposed for receiving and transmitting sensor information.

RFID

RFID works on the principal by assigning a unique number individually to each object in order to record information. RFID consists of readers, hosts and tags where tags receive and transmit radio waves due to which it is also known as a responder. RFI tags consist of active tags and passive tags which are available in different sizes and shapes. Passive tag is more advantageous as compared to active tag because it is cheaper than active tags. Tags have unique ID numbers and environmental information such as moisture level, temperature condition, and humidity etc. These tags are embedded and attached in multiple objects to identify that object.

2G/3G/4G – MOBILE COMMUNICATION

There are different generations of mobile communication standards including second generation (2G including GSM and CDMA), third generation (3Gincluding UMTS ad CDMA2000) and fourth generation (4G-including LTE). IoT devices based on these standards can communicate over cellular networks. Data rates for these standards rage from 9.6 Kb/s (2G).

ZIGBEE

ZigBee is on the top of IEEE 802 standards created by the ZigBee Alliance. It is a set of specifications for device-to-device networks having low power data rates. With the advancement of technology and increasing the demand of throughput there is a need for faster and low power consumption technology. These requirements are fulfilled by more established technologies which provide faster data transfer. In an agricultural environment IoT sensors sense the data and transfer it towards a remote server. After sensing, collected data is analyzed for decisions making.

802.11 – WiFi

IEEE 802.11 is a collection of Wireless Local Area Network (WLAN) communication standards. For example, 802.11a operates in the 5 GHz band, 802.11b and 802.11g operate in the 2.4 GHz band, 802.11n operates in the 2.4/5 GHz bands, 802.11ac operates in the 5 GHz band and 802.11ad operates in the

60 GHz band. These standards provide data rates from 1 Mb/s to 6.75 Gb/s. WiFi provides communication range in the order of 20 m (indoor) to 100 m (outdoor).

802.15.4 – LR-WPAN

IEEE 802.15.4 is a collection of Low-Rate Wireless Personal Area Networks (LR-WPAN) standards. These standards form the basis of specifications for high level communications protocols such as ZigBee. LR-WPAN standards provide data rates from 40 Kb/s to 250 Kb/s. These standards provide low-cost and low-speed communication with power constrained devices. It operates at 868/915 MHz and 2.4 GHz frequencies at low and high data rates, respectively. The specifications of 802.15.4 standards are available on the IEEE802.15 working group website (IEEE 802.15, 2014).

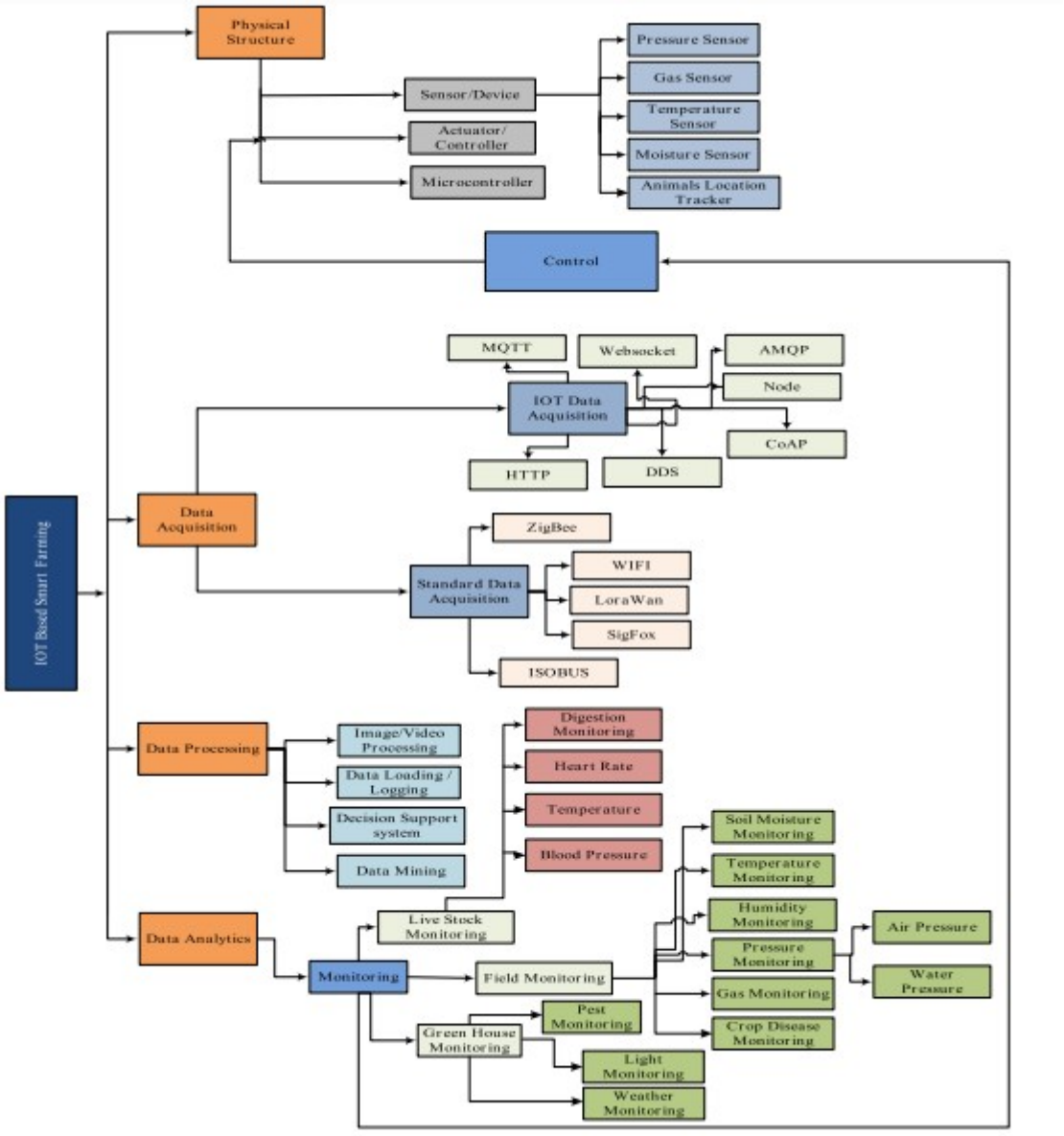
802.16 – WiMax

IEEE 802.16 is a collection of wireless broadband standards. WiMAX (Worldwide Interoperability for Microwave Access) standards provide data rates from 1.5 Mb/s to 1 Gb/s. The recent update (802.16m) provides a data rate of 100 Mb/s for mobile stations and 1 Gb/s for fixed stations. The specifications are readily available on the IEEE 802.16 working group website (IEEE 802.16, 2014).

802.15.1 – BlueTooth

Bluetooth is based on the IEEE 802.15.1 standard. It is a low power, low-cost wireless communication technology suitable for data transmission between mobile devices over a short range (8–10 m). The Bluetooth standard defines a personal area network (PAN) communication. It operates in the 2.4 GHz band. The data rate in various versions of the Bluetooth ranges from 1 Mb/s to 24 Mb/s. The ultra-low power, low-cost version of this standard is named as

Bluetooth Low Energy (BLE or Bluetooth Smart). Earlier, in 2010 BLE was merged with Bluetooth standard v4.0.



5.3 CONSTRAINTS

By adopting IoT in the agricultural sector we get numerous benefits, but still, there are challenges faced by IoT in agricultural sectors. The biggest challenges faced by IoT in the agricultural sector are lack of information, high adoption costs, and security concerns, etc. Most of the farmers are not aware of the implementation of IoT in agriculture. Major problem is that some of them are opposed to new ideas and they do not want to adopt even if it provides numerous benefits. The best thing that can be done to raise awareness of IoT's impact is to demonstrate farmers the use of IoT devices like drones, sensors and other technologies and they could provide them ease at work and accompanied by real-world examples.

Lack of Infrastructure – Even though the farmers embrace IoT technology, due to weak communication infrastructure they would not be able to take advantage of this technology. Farms are situated in rural areas and away from internet access. A farmer needs to have secure access to crop data from any place at any time, so link problems will make an advanced monitoring system in vain.

High Cost – In agriculture the equipment required to implement the IoT program is costly. Sensors are the least expensive part, but it would cost more than a thousand dollars to equip all the farmers 'fields to be with them.

Lack of Security– IoT devices communicate with older devices that have access to the internet network, so there is no assurance that they will be able to obtain data readings from drone mapping by using public connections. IoT agricultural systems gather large quantities of data that are difficult to secure.

Possibility for wrong Analysis of Weather Conditions - The bulk of the cycle is weather-specific in the case of agriculture. It is a natural phenomenon that can become unpredictable given the modified technologies. Climate conditions such as rain, sunshine, drought etc. are not influenced or regulated by any power. The value of natural occurrences cannot be reversed while the smart systems are in place. In intelligent agriculture, there is an issue in which computers may have a

detrimental effect on the environment. Because technology requires several computers, often the data will go wrong.

Wastage of supplies and exploitation of land - Since technology involves a lot of machines, there are chances where the data might get wrong at times. If there are faulty data processing equipment or sensors then it will lead to the situation where the wrong decisions are taken. This will lead to the overuse of resources like fertilizers or water. It might even lead to the over-application of fertilizers or pesticides on crops. This excessive use of chemicals might destroy the crop and reduce the richness of the land.

Location of the agricultural land - Smart agriculture needs the availability of the internet continuously. Rural parts of most of the developing countries do not fulfil this requirement. Moreover, internet connection is slower.

Technical faults-Most drones have less flight time and cover less area. Sensors help in mapping fields to understand their micro-scale characteristics. After mapping of crop yields, farmers could monitor and apply fertilizer and weed treatments only to It is a natural phenomenon which in spite of the updated technology can become unpredictable.

5.4 TRADE-OFFS

1) Replacing human labor with automation is a growing trend across multiple industries, and agriculture is no exception. Most aspects of farming are exceptionally labor-intensive, with much of that labor consisting of repetitive and standardized tasks—an ideal niche for robotics and automation.

2) IoT support in the farm's infrastructure is essential for the machines and the sensors to interface with the farmer, even as they operate autonomously and is the key feature to the "smart" farming system.

3)Farming has a wide scope of applications when it comes to the IoT. The Imminent use of technology has positively managed to minimize the risk and waste

experienced so far by the traditional farming methods. Farmers can now diagnose the areas detecting the fertility and conditions to carefully predict the possibility of the future yields.

4) Sowing seeds was once a laborious manual process. Modern agriculture improved on that with seeding machines, which can cover more ground much faster than a human. Effective seeding requires control over two variables: planting seeds at the correct depth, and spacing plants at the appropriate distance apart to allow for optimal growth.

5) Autonomous as well as innovative use of sensors and drones pulled together is what forms the Internet of Things or IoT. The IoT is the network of carefully outfitted physical devices with electrical connectivity that enables data exchange and aggregation. The use of development management software, actuators and sensors enable a more immediate integration of the physical world into electronically-based systems, resulting in efficiency, economic benefits, and reduced human labor.

6) The use of technology in agriculture makes smart agriculture a successful and much-needed initiative, of course with the increasing food supply demand. There is opportunity for intelligent farming, especially to understand and operate the equipment, to require certain skills. With technologies such as robots and computer-based intelligence for the operation of machines, it is highly unlikely that a typical farmer will acquire or even grow such skills. Such high-end innovations are not used by farmers. Computer language or artificial intelligence are not known.

7) While the use of intelligent technology is impressive in agriculture, it entails many costs. As mentioned earlier, it takes a great deal of money to convert this equipment if the machines are altered according to the farmer 's point. On the other hand, it will cost a lot of money to do the operation. As there are no higher profit levels for the agricultural industry, enormous investments are unlikely in this field. Even after altering the machinery, farmers may continue to operate the machinery improperly, causing harm or reparation. Because these devices are expensive now, it would cost a great deal of money to repair or upgrade them.

5.5 COST ANALYSIS

COMPONENT	COST(INR)
pH sensor	2000
Soil moisture sensor	180
Humidity sensor	400
Gas sensor	300
Temperature sensor	200
TMP36	250
IR sensor with remote	350
Resistors	250
Buzzer	130
LED	60
LCD display	320
Servo motor	230

Ultrasonic sensor	350
Hobby gear motor	470
9 V battery	260
Arduino Uno R3	460
GSM module	1200
Wi-Fi module-ESP8266	360
PCB Board	70
Wires	250
Jumper wires	200
Bread board	700 (175*4)
Relay module	150
USB cable	100
Hose pipe	250
Potentiometer	330
Water pump	4600[real time- for project submersible

	water pump-140]
Cost of software	0 [open source softwares are used]
Subscription cost	To be Determined
Total	14420

Subscription Plans:

PREMIUM
Overall subscription cost - 14420
[Inclusive of all the basic and advanced services and components]

CLASSIC
Overall subscription cost - 7820
[Without motor and pH sensor as mostly all the farmlands are preinstalled with a motor. While registration pH of the land is one of the important details collected from the farmers. Therefore, it is only necessary for the land which requires continuous monitoring of the pH which is a very rare case.]

STANDARD
Overall subscription cost - 6310
[Without motor, hose pipe, USB cable, TMP36, gas sensor, pH sensor and other basic components- Few components listed will be already available, therefore this subscription provides only the utmost service and technology required.]

CONCLUSION

IoT-based field monitoring is a dependable and quick technology that aids farmers in successfully monitoring their farms. This enables them to take remedial action in order to protect the crop and increase its production. Monitoring vegetation fields from afar not only aids in better staff utilization, but also improves crop quality. The information gathered could be utilized to develop strategies for achieving greater results in the future. By using sensors, the crop field that is connected to the internet, an appropriate decision can be taken.

In this project we have successfully designed an automated Smart Agriculture system which reduces the time and resources that is required while performing it manually. This system uses the technology of the Internet of Things. The system also measures moisture of soil and level of water in fields. This system works well in the ideal conditions and further improvement can be made when the conditions are not ideal like proper illumination or lightning. So, this innovation in the automation of agricultural devices will play an important role in precision agriculture and also help the farmers to increase the production to overcome the food scarcity of the world.

The proposed system can reduce the efforts of daily watering of plants. It also conserves water for irrigation by locating the sensor at the right position above the soil level. The plants can still sustain at low moisture levels when the temperature is moderate. The system is used to switch on/off the water pump according to the sensor readings there by automating the process of irrigation, which is one of the time-consuming activities. The system uses information from the sensors to irrigate soil which helps to prevent over irrigation (water wastage). Users can monitor the process online through a website. From this system it can be concluded that there can be considerable development in farming with the use of the Internet of Things.

FUTURE WORK

As a part of future work the following things can be included/added in our project:

- Deep learning, convolutional neural networks and computer vision can be used in later models of the Smart Agriculture system for plant disease recognition, pest control and irrigation control.
- In our project we implemented the hardware part individually i.e., each sensor has been designed as a separate system and each system is sending data back for further analysis. This might lead to more transmission traffic and memory usage. So as part of the future work on this project we decided to design the whole circuit through one common board in order to minimize the space complexity of the model. This can be done using System-on-chip technology in real time implementation.
- We can add a camera to look upon the growth of the plant on the live stream if not available in the house.
- We can make changes to the code to make the project work in plant specific conditions and monitor it.
- Cloud computing can be introduced to the project as it will facilitate the access of applications and data from any location worldwide and from any
- device with an internet connection. Cost savings; Cloud computing offers businesses with scalable computing resources hence saving them on the cost of acquiring and maintaining them. cloud storage provides various advantages as follows :
    - ✓ Data can be accessed from anywhere.
    - ✓ Hardware requirement and cost reduces.
    - ✓ Security of data increases.

REFERENCES

https://www.iotforall.com/iot-applications-in-agriculture

https://www.researchgate.net/publication/347453765_SMART_AGRICULTURE_USING_IOT

https://link.springer.com/article/10.1007/s41748-020-00146-

https://ieeexplore.ieee.org/abstract/document/8081906

https://www.rfwireless-world.com/Terminology/Advantages-and-uses-of-Agriculture-Sensors.html

https://developer.ibm.com/technologies/artificial-intelligence/tutorials/building-a-machine-learning-node-for-node-red-using-tensorflowjs/
https://rntlab.com/question/node-red-data-analysis/

INTERNATIONAL JOURNALS

- https://www.ijert.org/plant-monitoring-system
- https://ieeexplore.ieee.org/document/126197
- https://publons.com/journal/66441/ieee-international- conference-on-condition-monitor/

INTERNATIONAL CONFERENCES

- https://10times.com/e1zp-hf50-3x8s

REFERENCE JOURNALS:

- https://www.researchgate.net/publication/337656615_Internet_of_Things_and_Nodemcu_A_review_of_use_of_Nodemcu_ESP8266_in_IoT_products
- https://www.researchgate.net/publication/334207536_Humidity_and_temperature_monitoring
- https://www.researchgate.net/publication/326118051_Research_and_application_of_a_new_soil_moisture_sensor