PROJECT REPORT

**Project Name : SMART FARMER-IOT ENABLED SMART**

**FARMING APPLICATION**

**Team ID: PNT2022TMID50617**

**Team : MARI MUTHU K (Leader)**

**THEN RAJAN (Mem 1) KANNAN T (Mem 2) SURESH G (Mem 3)**

# ESAKKI MUTHU P (Mem 4)

## INTRODUCTION

* 1. Project Overview
  2. Purpose

## LITERATURE SURVEY

* 1. Existing problem
  2. References
  3. Problem Statement Definition

## IDEATION & PROPOSED SOLUTION

* 1. Empathy Map Canvas
  2. Ideation & Brainstorming
  3. Proposed Solution
  4. Problem Solution fit

## REQUIREMENT ANALYSIS

* 1. Functional requirement

## PROJECT DESIGN

* 1. Data Flow Diagrams
  2. Solution & Technical Architecture

## PROJECT PLANNING & SCHEDULING

* 1. Sprint Planning, Schedule & Estimation

## CODING & SOLUTIONING (Explain the features added in the project along with code)

* 1. Feature 1
  2. Feature code 2
  3. Reports from JIRA

## TESTING

* 1. Test Cases
  2. User Acceptance Testing

## RESULTS

* 1. Performance Metrics

## ADVANTAGES & DISADVANTAGES

1. **CONCLUSION**

## FUTURE SCOPE

1. **APPENDIX**

Source Code, GitHub & Project Demo Link

# INTRODUCTION

* 1. **Project Overview**

IoT-based agriculture system helps the farmer in monitoring different parameters of his field

like soil moisture, Temperature, humidity using some sensors. Farmers can monitor all the sensor parameters by using a web or mobile application even if the farmer is not near his field. Watering the crop is one of the important tasks for the farmers. They can make the decision whether to water the crop or postpone it by monitoring the sensor parameters and controlling the motor pumps from the mobile application itself.



# Purpose

To improving the quality of agricultural services, enhancing value addition, granting market access and improving flow of farming inputs needed to farmers. They are

1. Increasing the amount of real-time data on the crop,
2. Remote monitoring and controlling of farmers,
3. Controlling water and other natural resources,
4. Improving livestock Management, 5)Accurate evaluation of soil and crops; 6)Improving agricultural production.

# LITERATURE SURVEY

**2.2 Existing problem**

The challenges of a smart farming system include the integration of these sensors and tying the sensor data to the analytics driving automation and response activities. When integrated, the use of data analytics can reduce the overall cost of agriculture and contribute to higher production from the same amount of area through precise control of water, fertilizer and light. Smart methods allow for farming on smaller and more distributed lands through remote monitoring, whether indoor or outdoor.

To successfully deploy a smart farming system, consider setting up a communications

network that can integrate a limited number of sensors across a large area of farmland. This will require third-party network provisioning or setting up a private network consisting of access points and uplinks to a private backhaul network, which channels all the data traffic to centralized monitoring software or an analytics head-end system

* It is not a secure system.
* There is no motion detection for protection of agriculture field.
* Automation is not available.

# References

1. ISSN No:-2456-2165 Volume 4, Issue 2 Feb – 2019: "Solars' Energy: - A safe and reliable, eco-friendly and sustainable Clean Energy Option for Future India: - A Review."
2. Universal Paper of advanced science and science and exploration technology. [2] GRD Journals- Global Research and Development Journal for Engineering | Volume 4 | Issue 3 | February (2019) ISSN: 2455-5703 “Design and Implementation of an Advanced Security System for Farm Protection from Wild animals".
3. [https://www.researchgate.net/publication/349141429\_Towards\_Climate\_Smart\_Farming](http://www.researchgate.net/publication/349141429_Towards_Climate_Smart_Farming-)-

A\_Reference\_Architecture\_for\_Integrated\_Farming\_Systems

1. International Journal of Innovationsin Engineering and Science, Impact Factor Value 4.046 e-ISSN: 2456-3463 Vol.4, No. 5, 2019 “Solar Powered Smart Fencing System for Agriculture Protection using GSM & Wireless Camera”.
2. International Journal of Management, Technology And Engineering ISSN NO : 2249-7455 Volume 8, Issue VII, JULY/2018”Protecting Crops From Birds, Using Sound Technology In

Agriculture”

1. American Journal of Engineering Research (AJER)2018 eISSN: 2320-0847 p

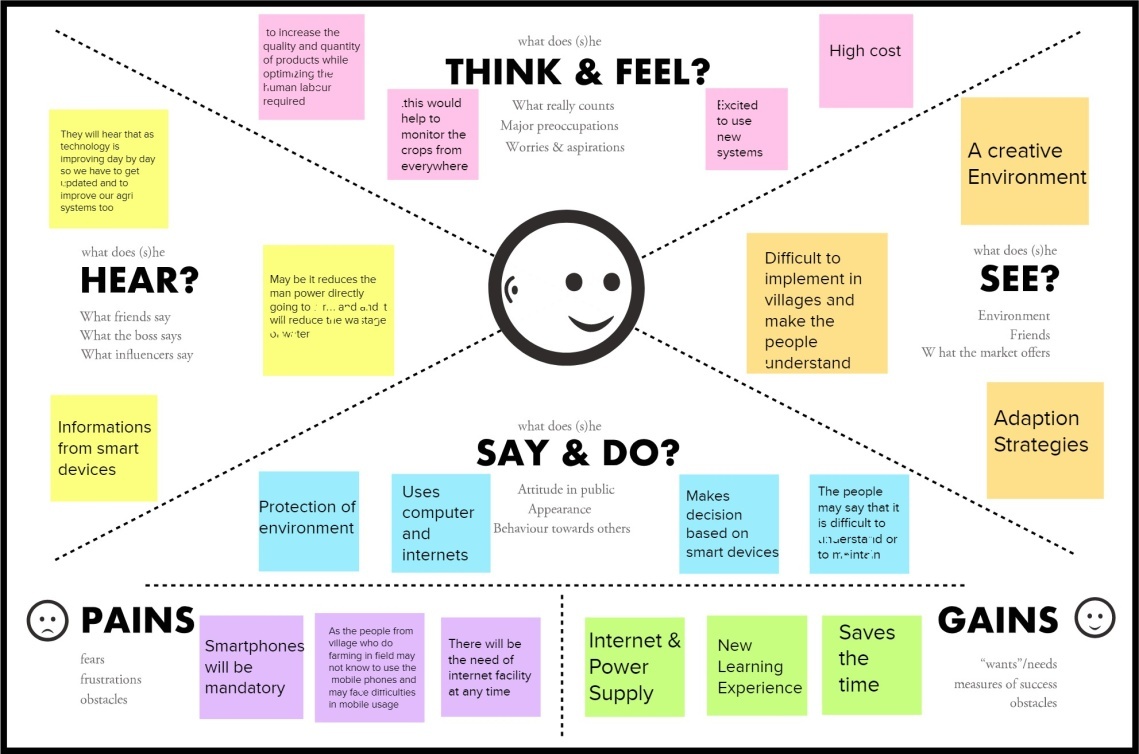
ISSN : 2320- 0936 Volume-7, Issue-7, pp-326-330 “Moisture Sensing Automatic Plant Watering System Using Arduino Uno”.

# Problem Statement Definition

The soil moisture sensor measures wetness content in the soil. The Arduino UNO microcontroller used to receive input from a various sensors and it can be controlled automatically. When soil moisture sensor goes low the water pump will be on and it exceeds defined levels of the water motor will turn off automatically. We can constantly monitor the growth of a crop using ultrasonic sensor. PIR sensor detects the motion or unusual movement in the agricultural land. This device his very helpful to the former to monitor and control environmental parameters at their field. The farmers did not go to theirfield, they can remotely monitor and control using cloud.

# IDEATION & PROPOSED SOLUTION

* 1. **Empathy Map Canvas**

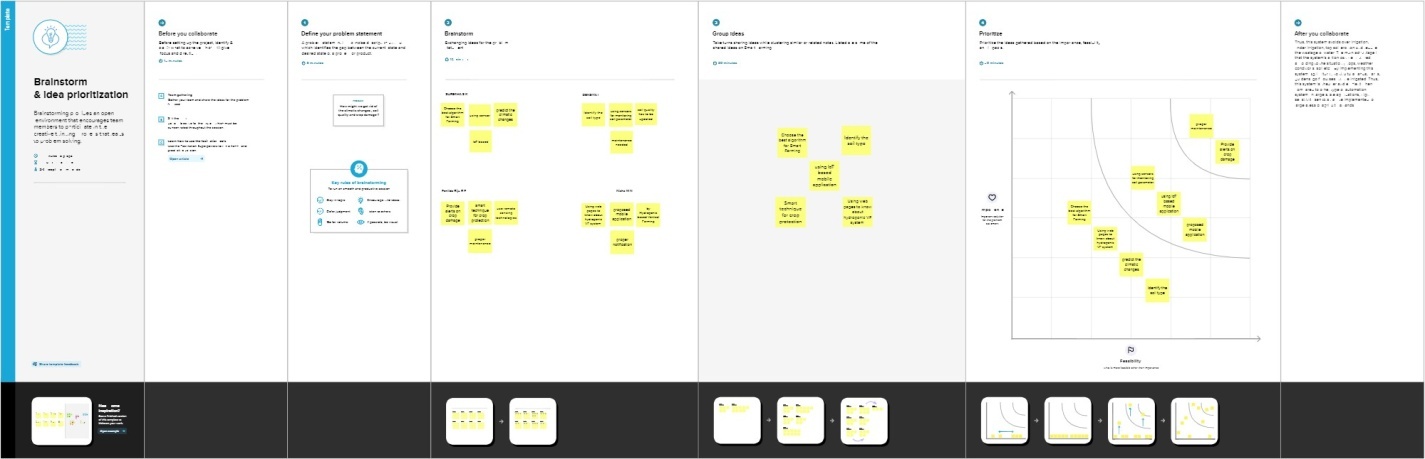


# Ideation and Brainstorming

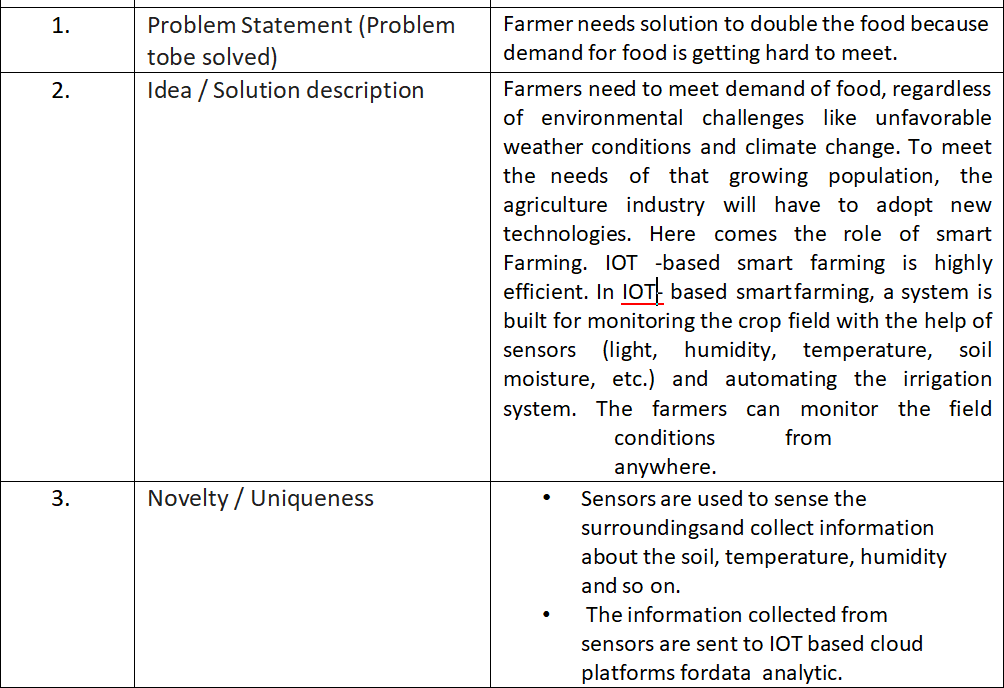
**Ideation:-**

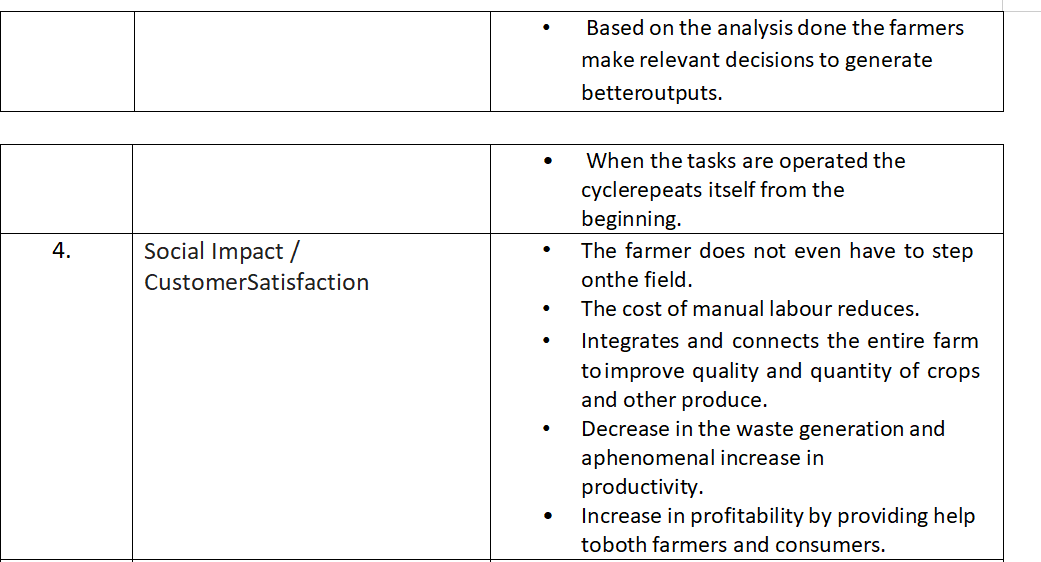


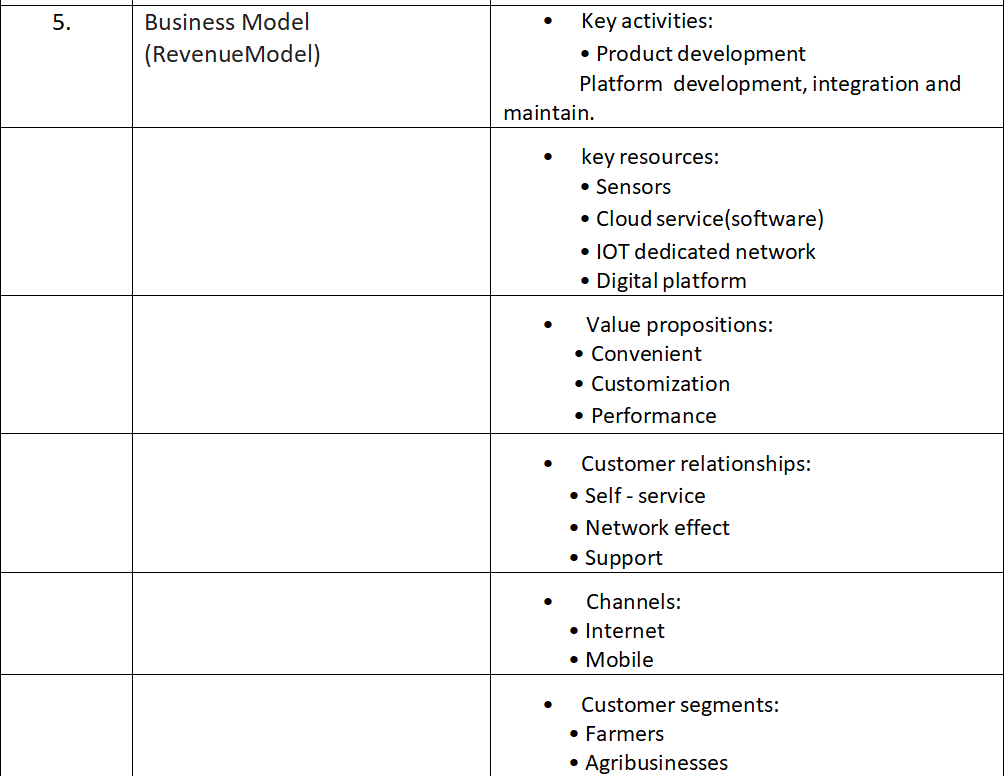
**Brainstorming :-**



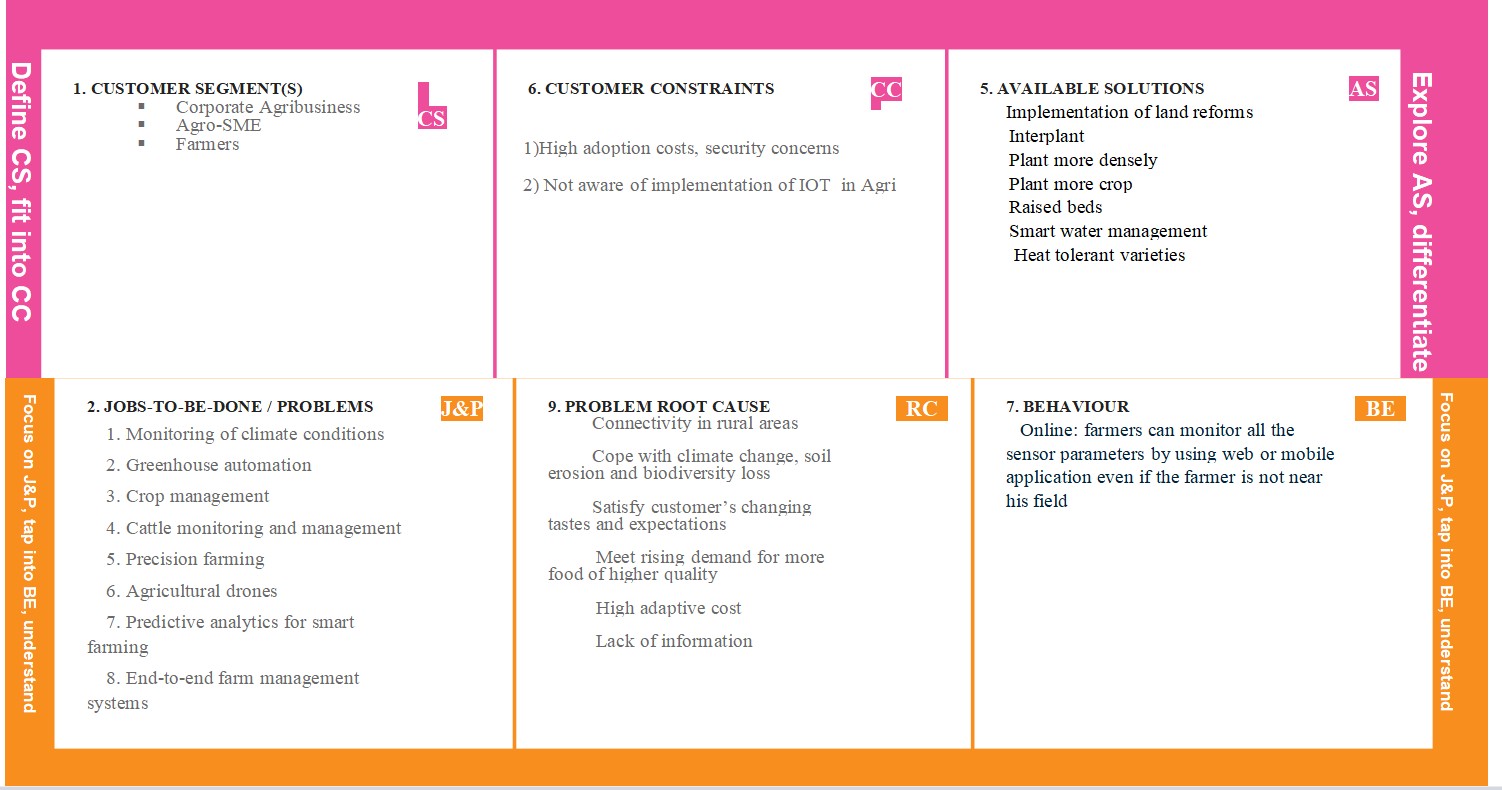
# Proposed Solution

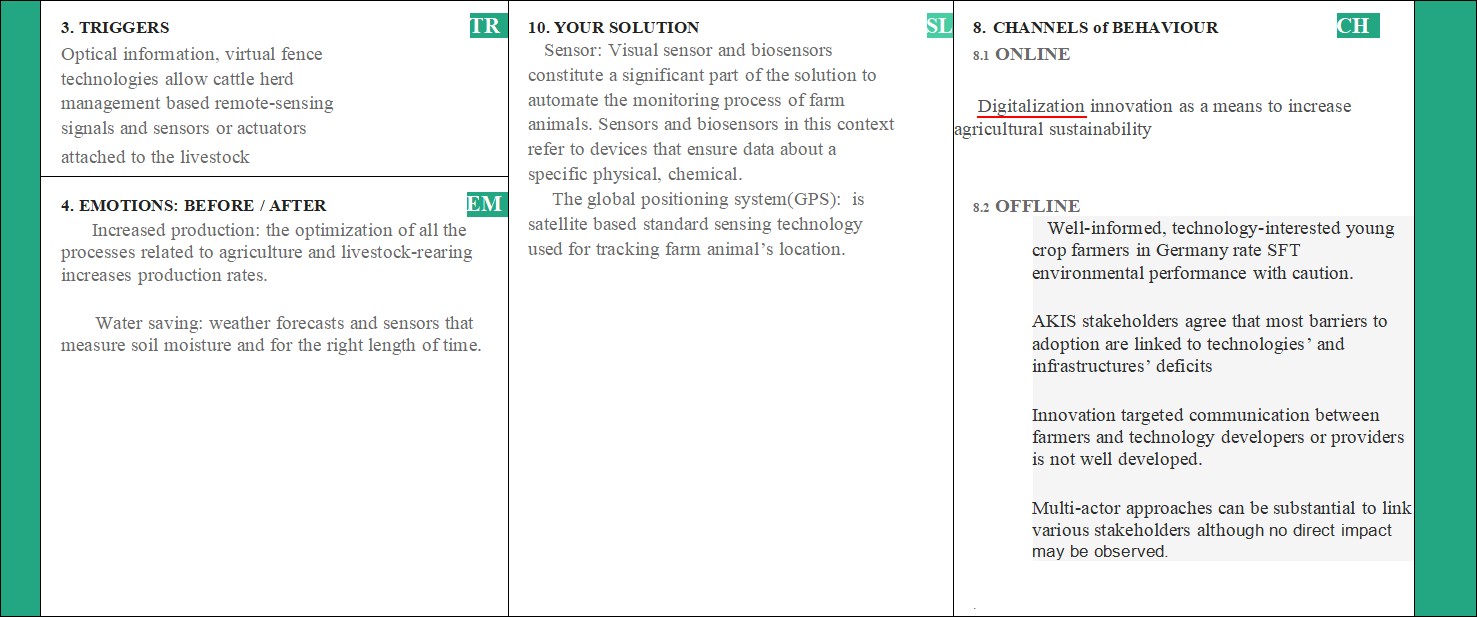






* 1. **Problem solution**





# REQUIREMENT ANALYSIS

* 1. **Functional Requirements:-**

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement**  **(Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | User Registration | Phone Application And Wifi Module |
| FR-2 | User Confirmation | Confirmation via Email  Confirmation via OTP |
| FR-3 | User Login/App | Check Id/ Username And Check Roles access |
| FR-4 | User Dashboard | Learn to access the application |
| FR-5 | Monitoring of climate  conditions | Using gadgets to map the climate conditions |
| FR-6 | Agricultural drones | Using gadgets to agriculture spraying, crop  monitoring |
| FR-7 | Greenhouse automation | Use of IOT sensors enables them to get accurate real-  time information |
| FR-8 | Log out | Exit |

# Non Functional Requirements :-

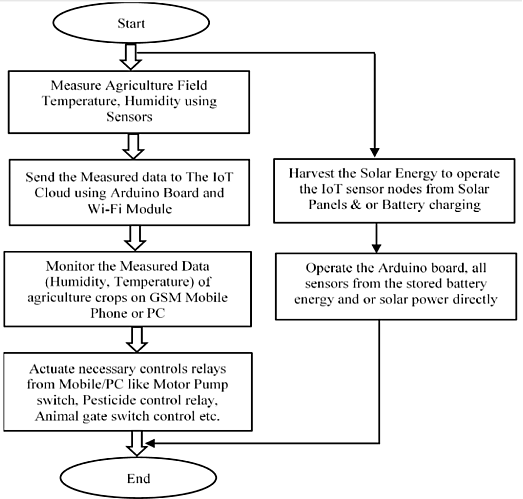
|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | Early detection and application of inputs only in the affected region, saving costs  Uses satellite imagery to detect the different zones in  farms |
| NFR-2 | **Security** | IOT and smart communication technologies introduce a vast exposure to cyber security threats  and vulnerabilities in smart farming environments |
| NFR-3 | **Reliability** | Reliable weather forecasts to maximize resource usage and minimize losses |
| NFR-4 | **Performance** | IOT devices and sensors capture various types of data from all over the field that can  then be analyzed through big data tools |
| NFR-5 | **Availability** | **Tanzania and Vietnam** are among the countries that will work towards climate smart agriculture – an approach aimed at transforming food systems |
| NFR-6 | **Scalability** | Scalability is the ability to increase available resources and system capability  without the need to a major system redesign or implementation, we can increase the capacity for data processing by increasing the cloud resources in the second layer and computation resources in the  third layer |

1. **PROJECT DESIGN**

# Data Flow Diagram

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs,

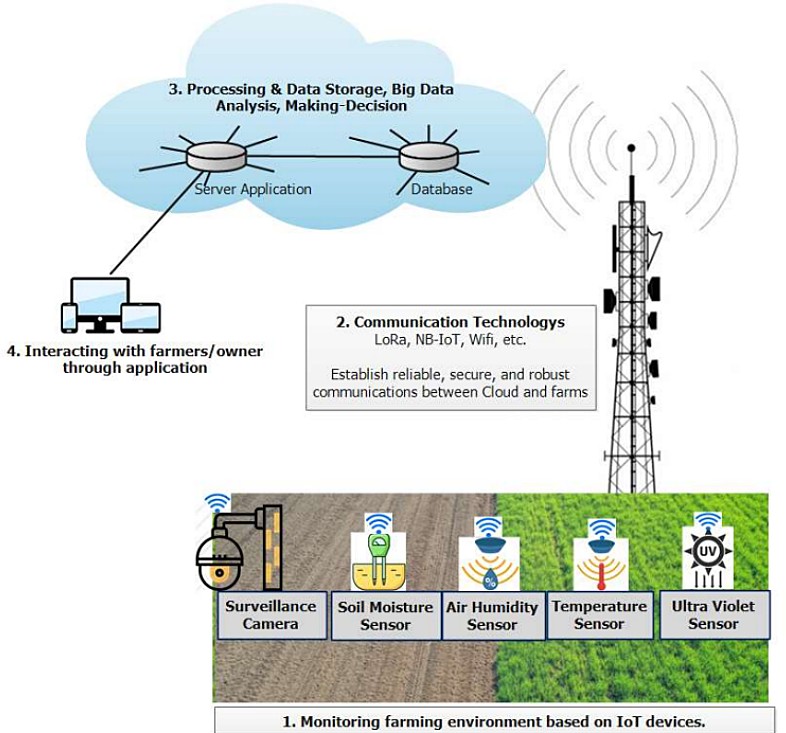
outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyze an existing system or model a new one. That’s why DFDs remain so popular after all these years. While they work well for data flow software and systems, they are less applicable nowadays to visualizing interactive, real-time or database-oriented software or systems.



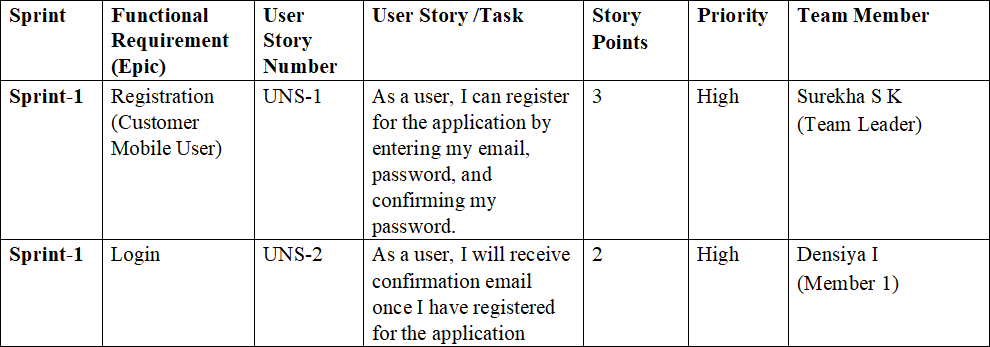
# Solution & Technical Architecture

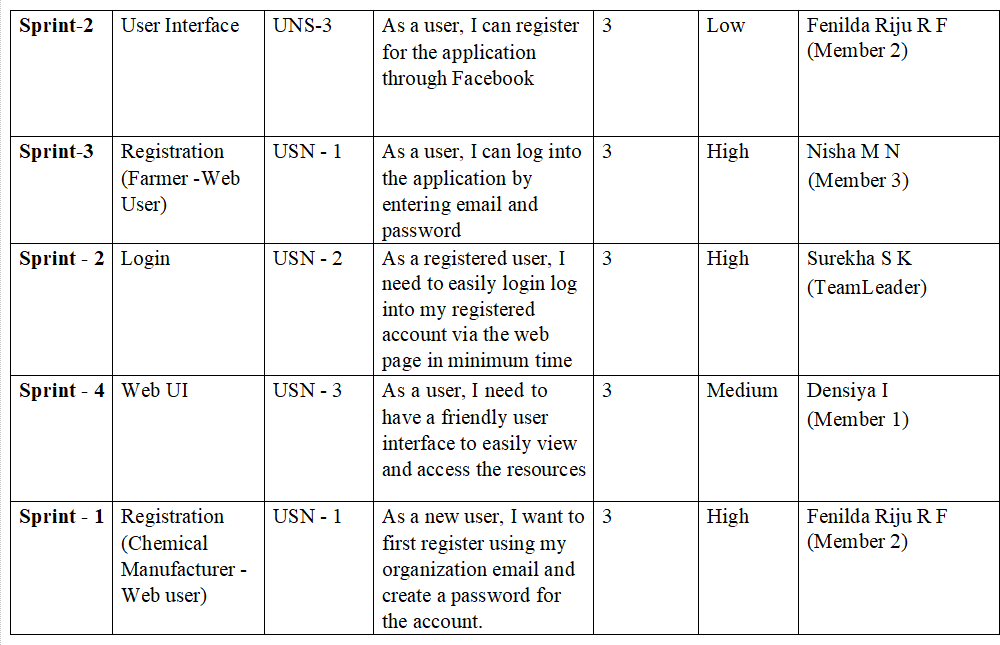
Technology architecture associates application components from application architecture with technology components representing software and hardware components. It provides a more concrete view of the way in which application components will be realized and deployed. It enables the migration problems that can arise between the different steps. It provides a more

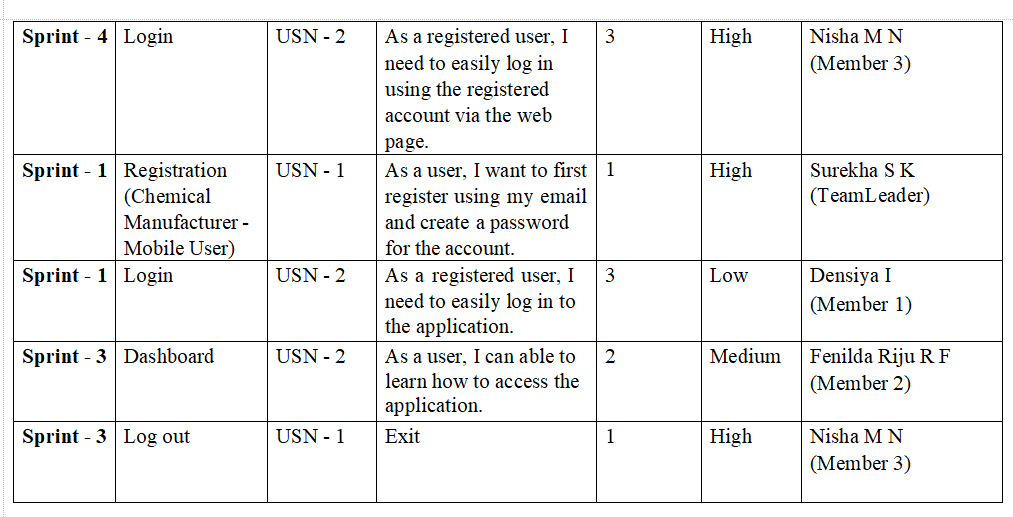
precise means of evaluating responses to constraints .



# PROJECT PLANNING AND SCHEDULING







1. **CODING & SOLUTIONING**
   1. **Feature Code 1**

import time import sys

import ibmiotf.application # to install pip install ibmiotf import ibmiotf.device

#Provide your IBM Watson Device Credentials organization = "1jk4ps"

deviceType = "NODEMCU1" deviceId = "12345"

authMethod = "token"authToken = "\*ev9TDcZDdKDBUA81@" def myCommandCallback(cmd): # function for Callback print("Command received: %s" % cmd.data)

if cmd.data['command']=='motoron': print("Motor On IS RECEIVED")

elif cmd.data['command']=='motoroff': print("Motor Off IS RECEIVED")

if cmd.command == "setInterval":

if 'interval' not in cmd.data:

print("Error - command is missing required information: 'interval'") else: interval = cmd.data['interval']

elif cmd.command == "print":

if 'message' not in cmd.data:

print("Error - command is missing required information: 'message'") else:

output=cmd.data['message'] print(output)

try:

deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth- method": authMethod, "auth-token": authToken}

deviceCli = ibmiotf.device.Client(deviceOptions) #..............................................

except Exception as e:print("Caught exception connecting device: %s" % str(e)) sys.exit()

# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times

deviceCli.connect() while True:

deviceCli.commandCallback = myCommandCallback # Disconnect the device and application from the cloud deviceCli.disconnect()

* 1. **Feature Code 2**

import time import sys

import ibmiotf.application import ibmiotf.device import random

#Provide your IBM Watson Device Credentials organization = "1jk4ps"

deviceType = "PNT2022TMID51719" deviceId = "Smart\_Farmer" authMethod = "token"

authToken = "l1\*53hCIhmEbf!&Es&" # Initialize GPIO

def myCommandCallback(cmd):

print("Command received: %s" % cmd.data['command']) status=cmd.data['command']

if status=="lighton":

print ("led is on") else :

print ("led is off") #print(cmd)

try:

deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}

deviceCli = ibmiotf.device.Client(deviceOptions) #..............................................

except Exception as e:

print("Caught exception connecting device: %s" % str(e)) sys.exit()

# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times

deviceCli.connect()while True: #Get Sensor Data from DHT11 temp=random.randint(0,100) Humid=random.randint(0,100)

data = { 'temp' : temp, 'Humid': Humid } #print data

def myOnPublishCallback():

print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % Humid, "to IBM Watson")

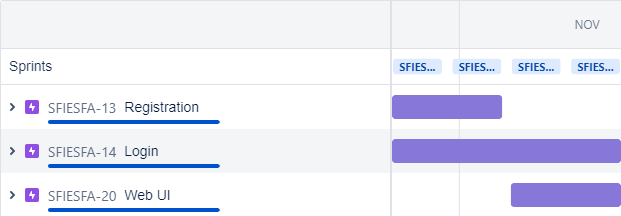
success = deviceCli.publishEvent("SDFRN", "json", data, qos=0, on\_publish=myOnPublishCallback)

if not success:

print("Not connected to IoTF") time.sleep(1)

deviceCli.commandCallback = myCommandCallback # Disconnect the device and application from the cloud deviceCli.disconnect()

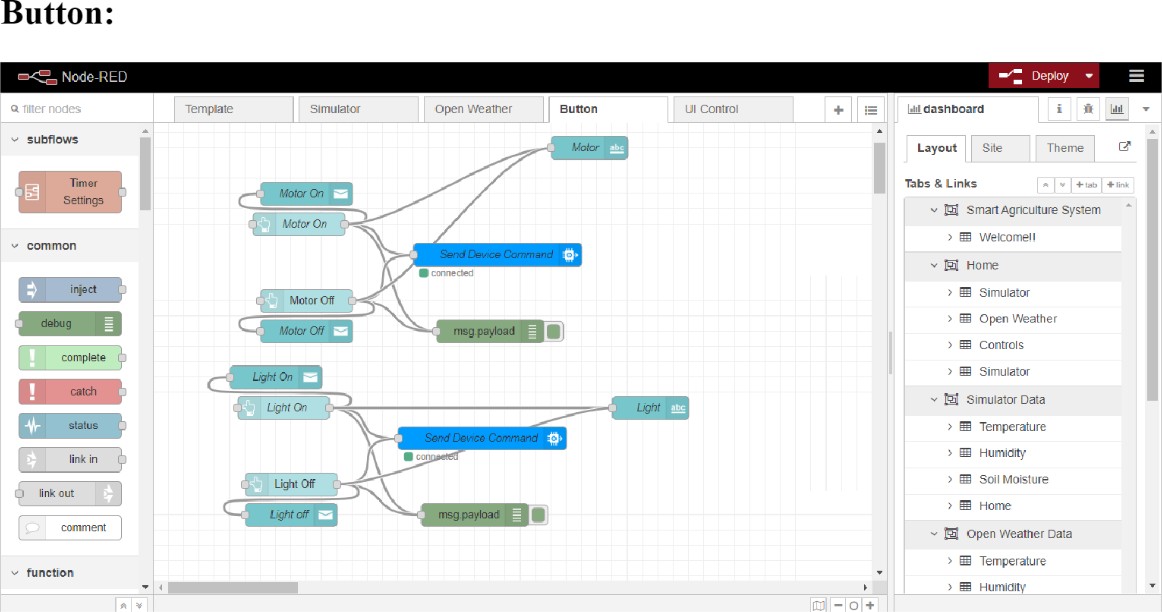
# Reports from JIRA

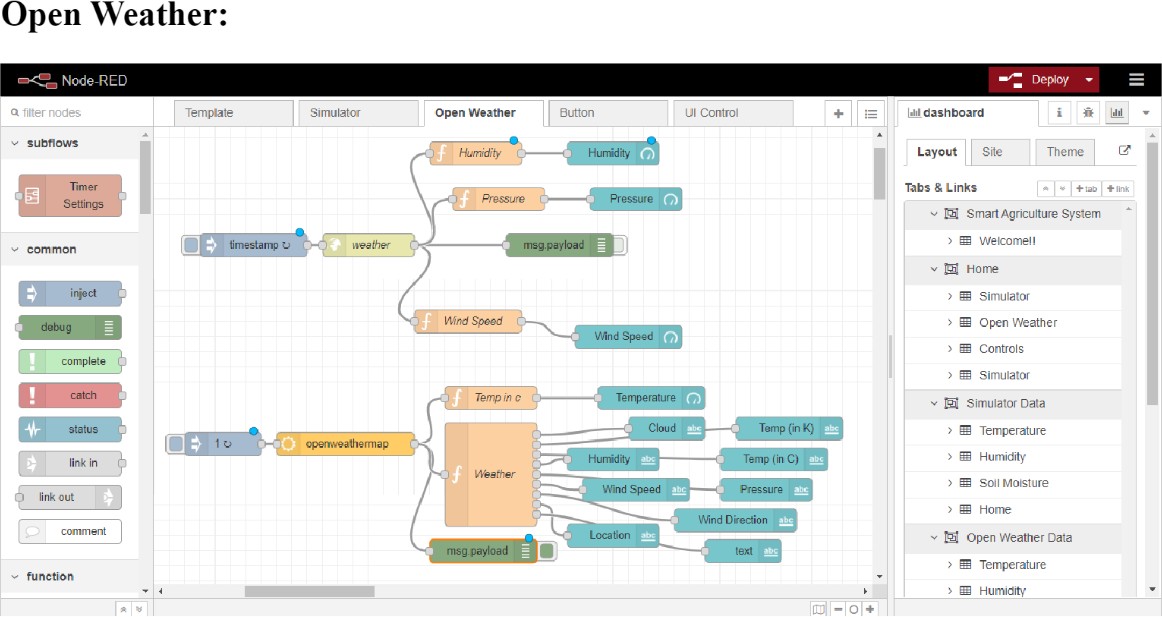


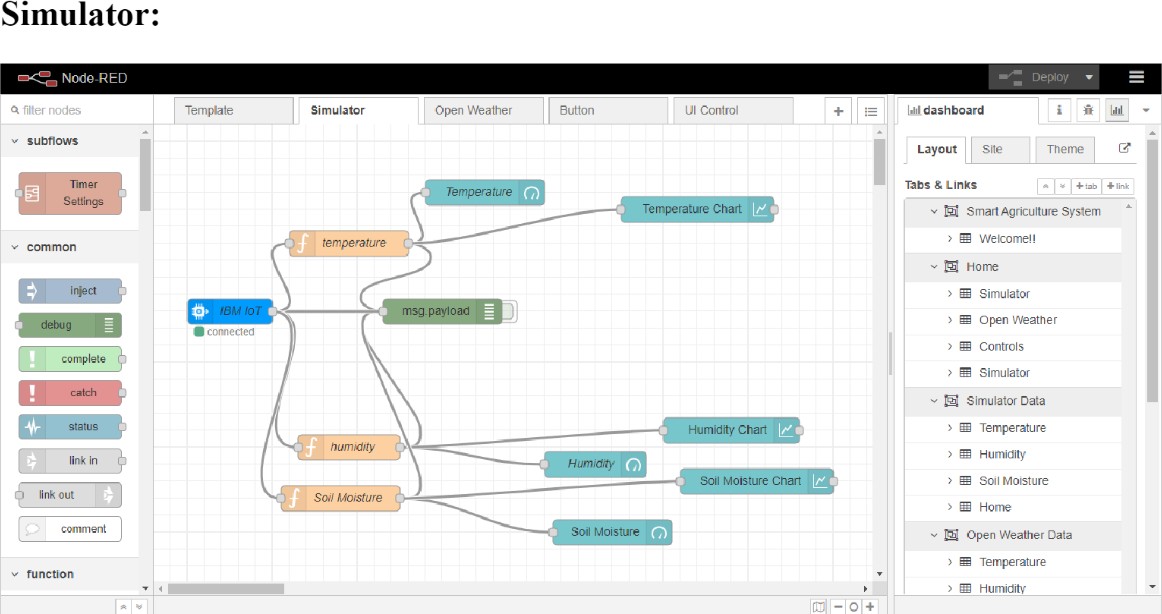
1. **TESTING**

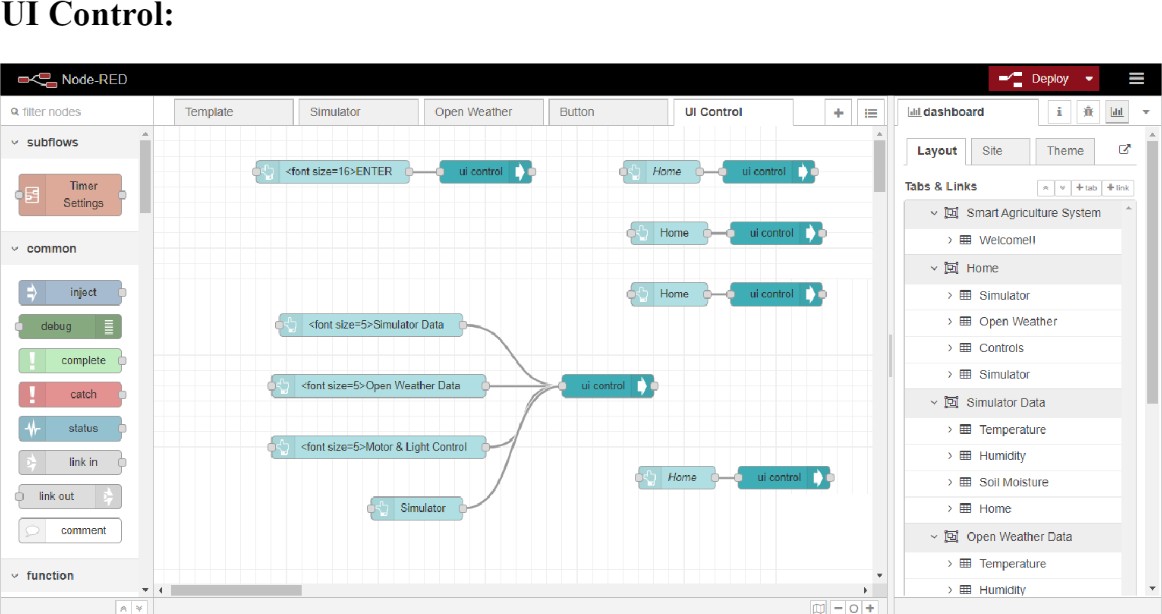
# Test case

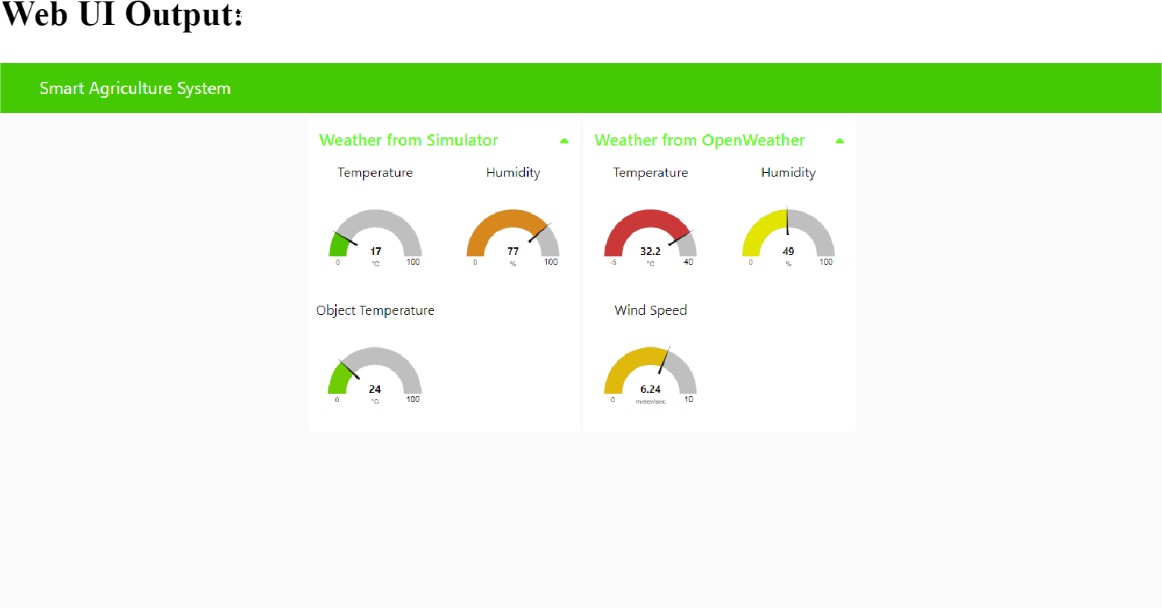
Web application using Node Red



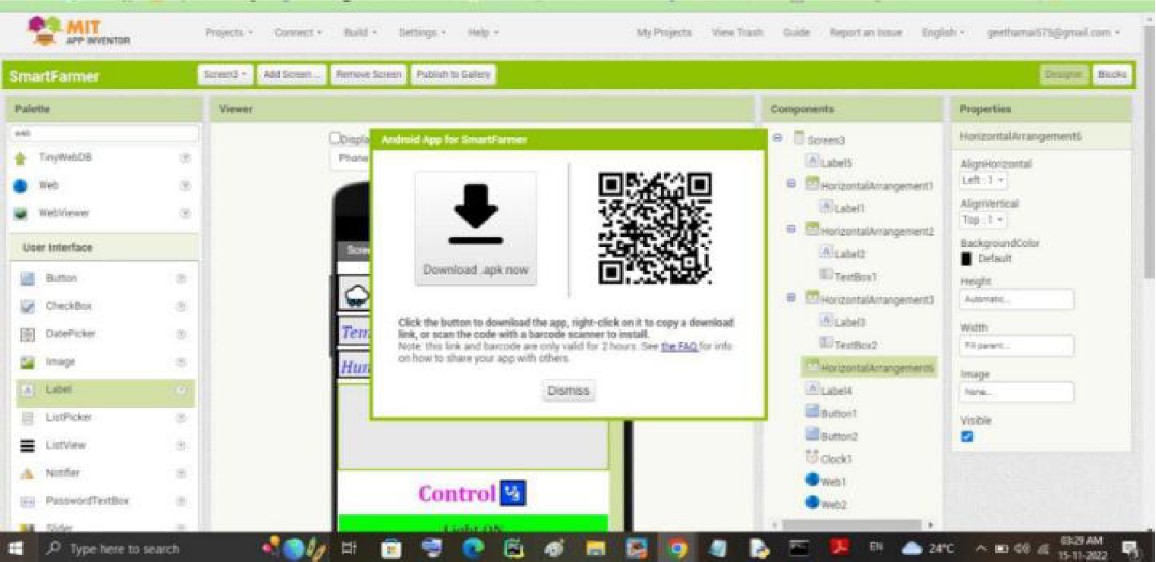


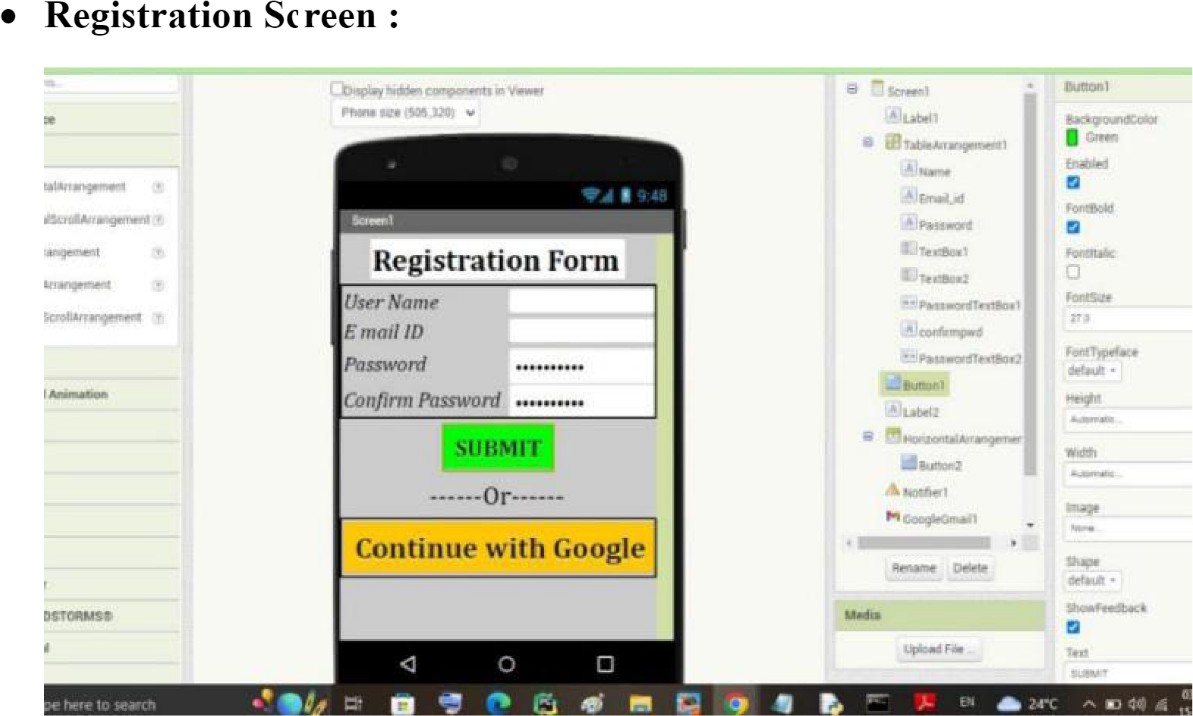




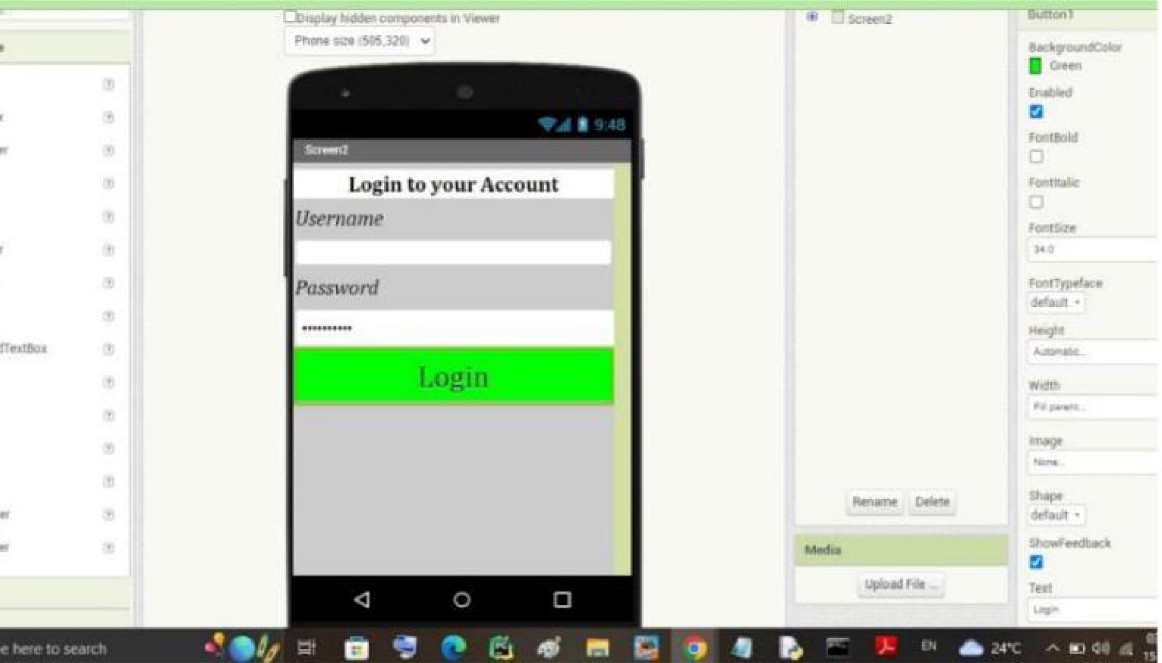


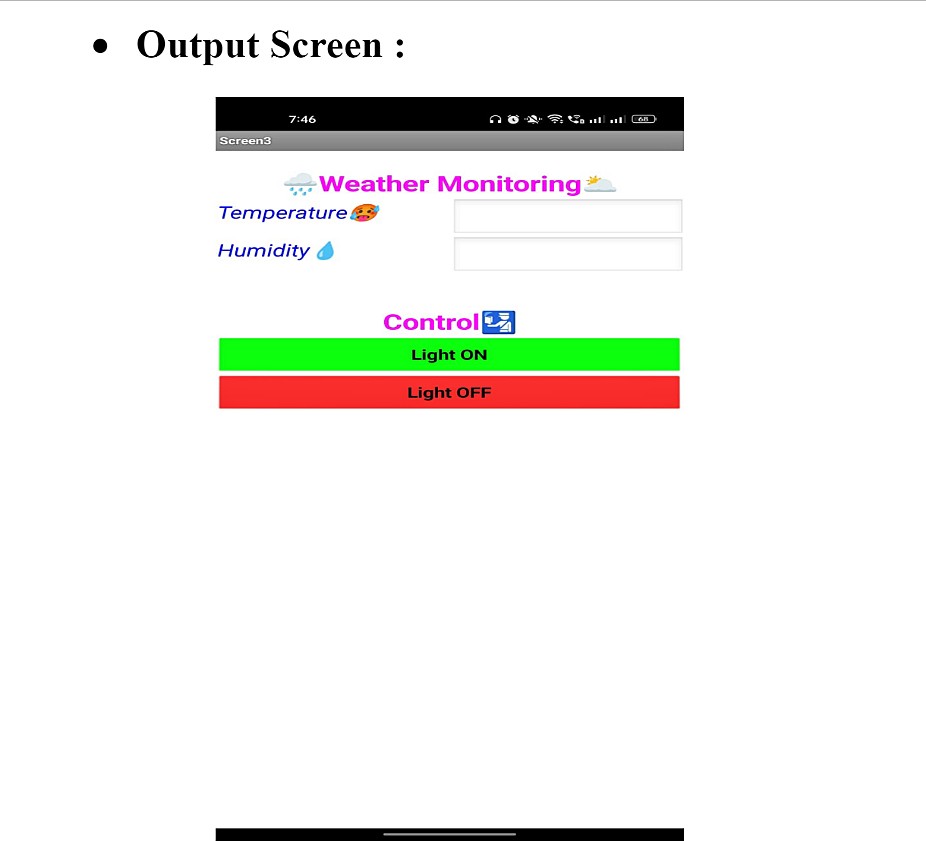
* 1. **User Acceptance Testing**
     + **APK screen :**





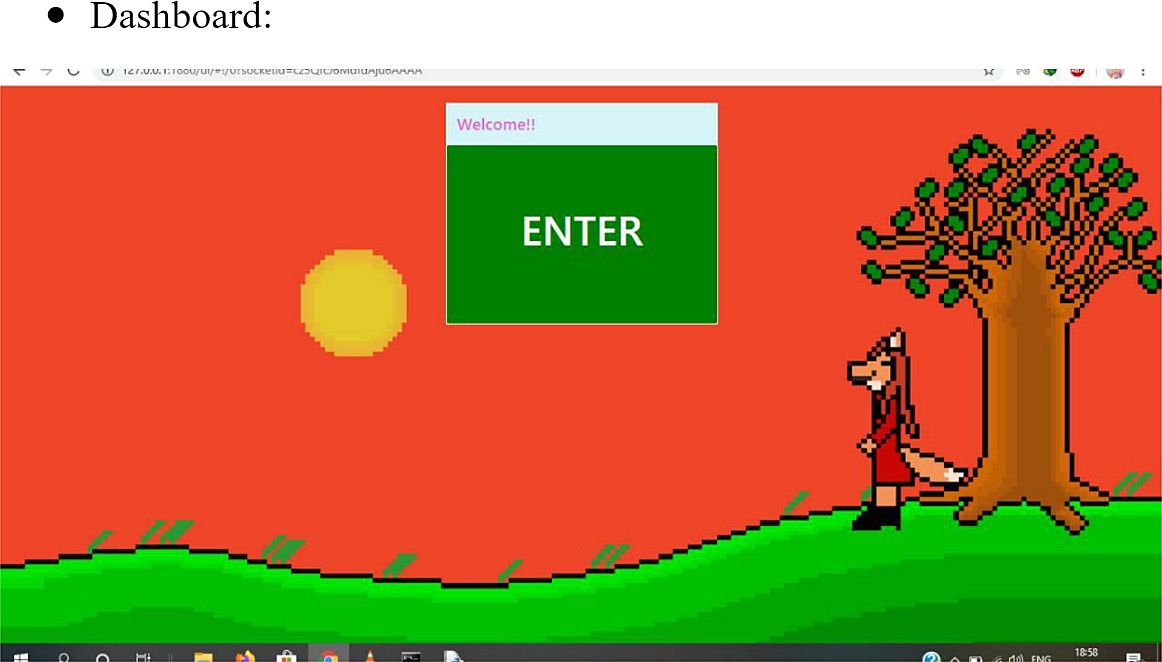
* **Login Screen :**

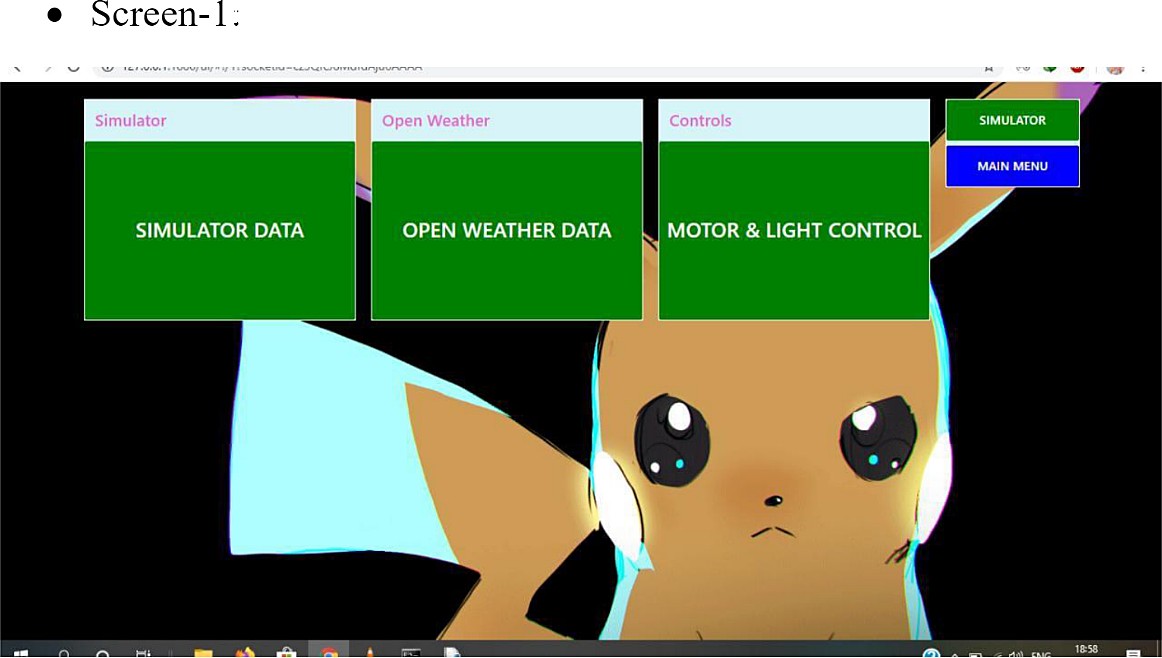


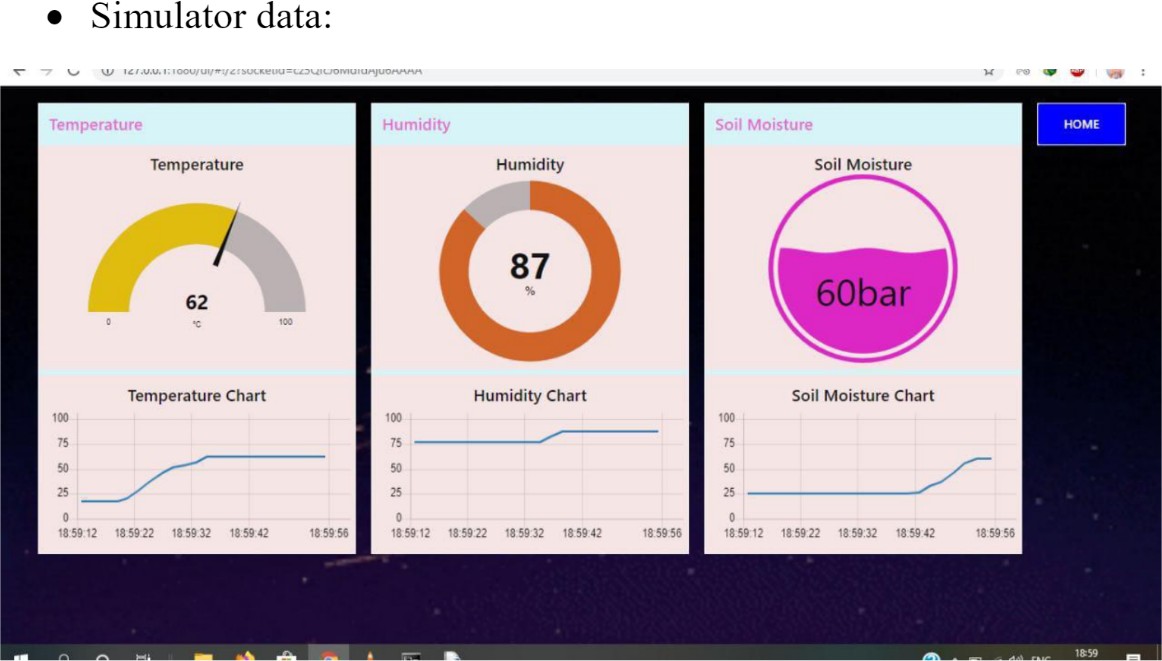


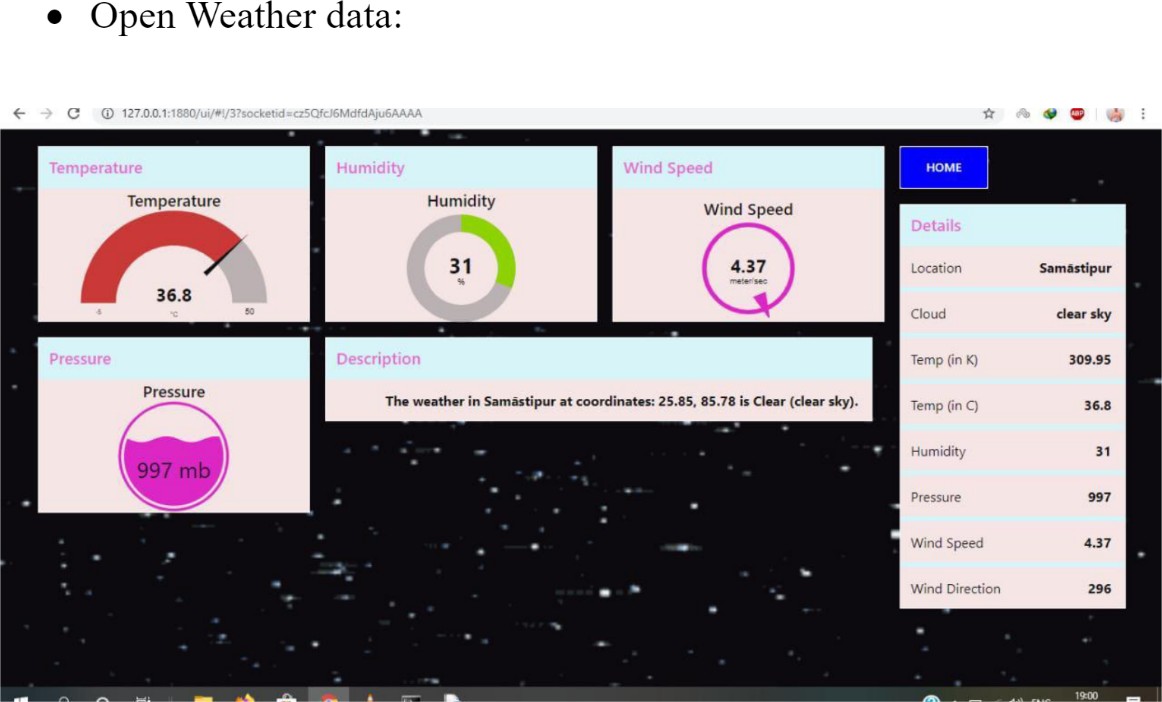
1. **RESULTS**

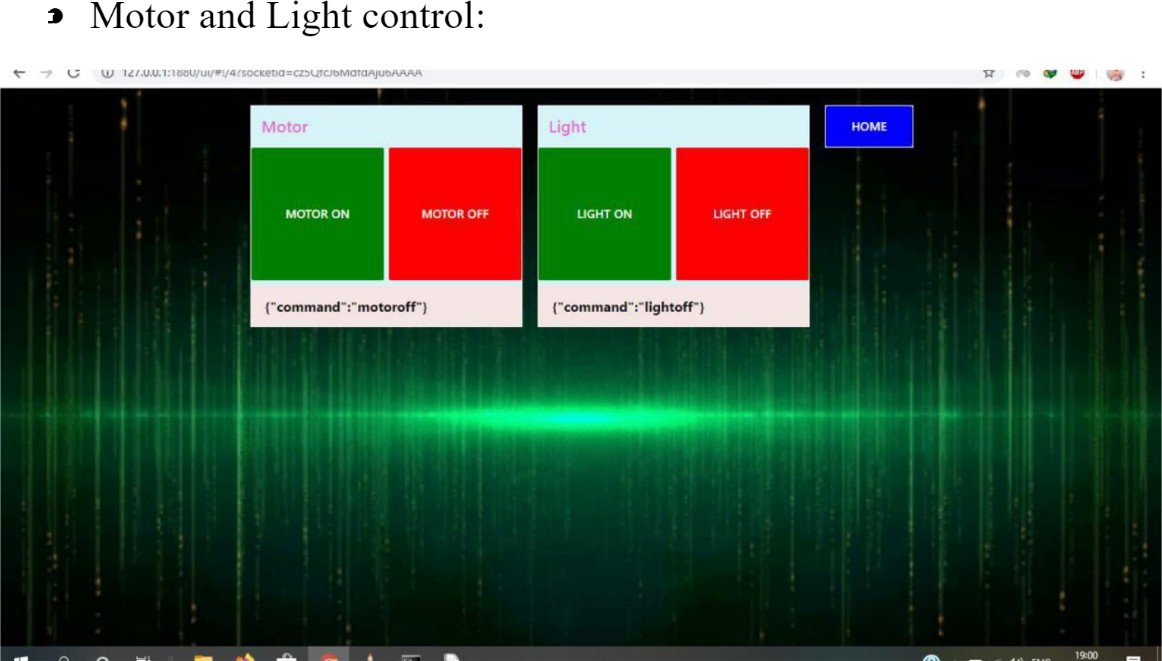
**9.1 Performance Metrics**

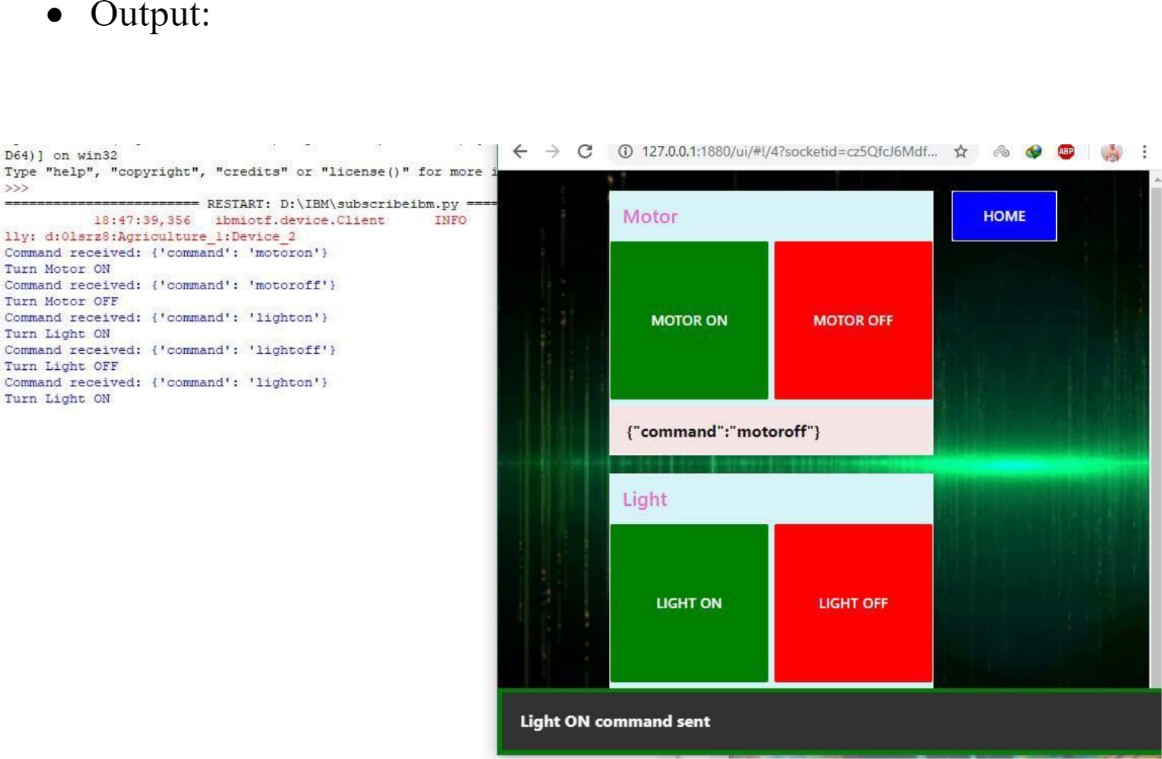












# ADVANTAGES & DISADVANTAGES

**Advantages:**

* A remote control system can help in working irrigation system valves dependent on schedule. Irrigating remote farm properties can be exceptionally troublesome and labor intensive. It gets hard to comprehend when the valves were started and whether the ideal measure of water was distributed.
* For situations where a quick reaction is required, manual valve actuation may not be conceivable constantly. Thus, remote observing and control of irrigation systems, generators or wind machines or some other motor-driven hardware become the next logical

step.

* Various solutions are available to monitor engine statistics and starting or stopping the engine. When the client chooses to begin or stop the motor, the program transmits a sign to

the unit within seconds by means of a mobile phone system.

* Submersible weight sensors or ultrasonic sensors can screen the degree of tanks, lakes, wells and different kinds of fluid stockpiling like fuel and compost. The product figures volume dependent on the tank or lake geometry after some time. It conveys alarms dependent on various conditions.

# Disadvantages:

* The smart agriculture needs availability of internet continuously. Rural part of most of the

developing countries do not fulfil this requirement. Moreover internet connection is

slower.

* The smart farming based equipment require farmers to understand and learn the use of technology. This is major challenge in adopting smart agriculture farming at large scale across the countries.

# CONCLUSION

Farmers can benefit greatly from an IoT-based smart agriculture system. As a result of the lack of irrigation, agriculture suffers. Climate factors such as humidity, temperature, and moisture can be adjusted dependent on the local environmental variables. This technology also detects animal invasions, which are a major cause of crop loss. This

technology aids in the scheduling of irrigation based on present data from the field and records from a climate source. It helps in deciding the farmer to whether to do irrigation or not to do. Continuous internet connectivity is required for continuous monitoring of data from sensors. This also can be overcome by using GSM unit as an alternative of mobile app. By GSM, SMS can be sent to farmers phone.

# FUTURE SCOPE

In the current project we have implemented the project that can protect and maintain the the crop. In this project the farmer monitor and control the field remotely. In future we can add or update few more things to this project

. • We can create few more models of the same project ,so that the farmer can have information of a entire.

* We can update the this project by using solar power mechanism. So that the power supply from electric poles can be replaced with solar panels. It reduces the power line cost. It will be a one time investment. We can add solar fencing technology to this project.
* We can use GSM technology to this project so that the farmers can get the information directly to his home through SMS. This helps the farmer to get information if there is a internet issues.
* We can add camera feature so that the farmer can monitor his field in real time. This helps in avoiding theft

# APPENDIX

**Source Code**

import wiotp.sdk.device import time

import os import datetime import random myConfig ={ "identity": {

"orgId": "0hzydu",

"typeId": "NodeMCU", "deviceId": "12345"

},

"auth": {

"token": "12345678"

}

}

client = wiotp.sdk.device.DeviceClient(config=myConfig,logHandlers=None) client.connect ()

def myCommandCallback (cmd) :

print("Message received from IBM IoT Platform: %s" %cmd.data['command']) m=cmd.data['command']

if (m=="motoron"):

print("Motor is switchedon") elif (m=="motoroff"):

print ("Motor is switchedOFF") print (" ")

while True:

moist =random.randint (0,100)

temp=random.randint (-20, 125)

hum=random.randint (0, 100)myData={'moisture':moist,'temperature':temp,'humidity':hum} client.publishEvent (eventId="status", msgFormat="json", data=myData, qos=0 , onPublish=None)

print ("Published data Successfully: %s",myData) time.sleep (2)

client.commandCallback =myCommandCallback client.disconnect ()

**Git hub link :-** [**https://github.com/IBM-EPBL/SI-GuidedProject-42881-1666511988**](https://github.com/IBM-EPBL/SI-GuidedProject-42881-1666511988)

## Demo video link :-

<https://drive.google.com/file/d/1rzG1hAwYHP-cqZz6yeKWnSlBb-qbNI2f/view?usp=drivesdk>