PROJECT REPORT

Project Name:

SMARTFARMER- IOT ENABLED SMART FARMINGAPPLICATION

Team ID:
PNT2022TMID36144

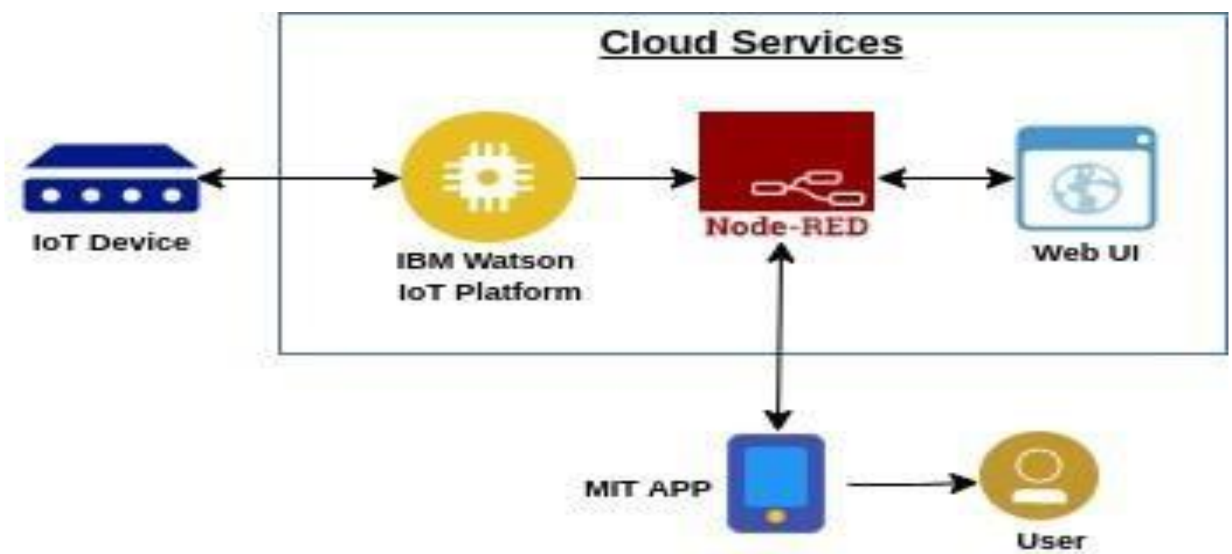
Team:

- **SATHISHKUMAR G**
- **HEMANTH R**
- **ARAVINDHAN A**
- **UMAPATHY A V**

1. INTRODUCTION

1.1 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.



1.2 PURPOSE

The smart agriculture model main aim **to avoid water wastage in the irrigation process**. It is low cost and efficient system Is shown below. It includes NodeMCU, Arduino Nano, sensors like soil moisture and Dht11, solenoid valves, relays.

2.LITERATURE SURVEY

2.2 Existing problem

The challenges of a smart agriculture 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 agriculture 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.

2.2 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] 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”.

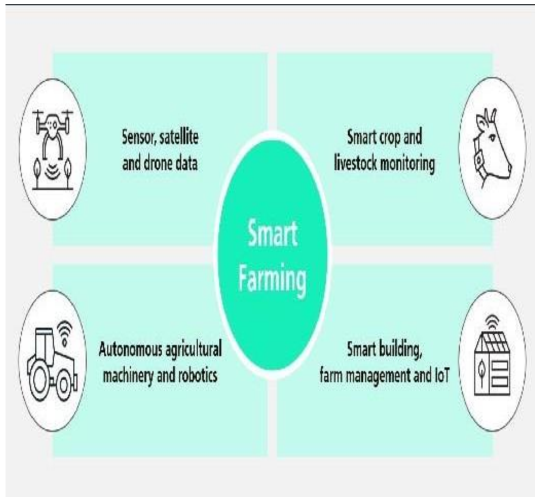
[4] International Journal of Management, Technology And Engineering ISSN NO : 2249-7455 Volume 8, Issue VII, JULY/2018”Protecting Crops From Birds, Using Sound Technology In Agriculture” [5] American Journal of Engineering Research (AJER)2018 eISSN: 2320-0847 pISSN : 2320- 0936 Volume-7, Issue-7, pp-326-330 “Moisture Sensing Automatic Plant Watering System Using Arduino Uno”.

2.3 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.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation and Brainstorming

Introduction on Internet of Things (IoT), application of IoT in agricultural field to improve the yield and quality by reducing the cost is provided. The sensors which are used in the architecture are discussed briefly and the process of transmission of data from the agriculture field to the central system is explained. The proposed system advantages are included. In addition, open research issues, challenges, and future of IoT in agricultural field are highlighted. The concept is basically developed on an idea, where there are numerous things or objects - such as Arduino, sensors, GSM models, LCD display, etc., that are connected with the Internet. Each of the objects has a different address and is able to interact with other items. The things or objects co-operate with each other to reach a common goal.

We are going to construct a smart agricultural monitoring system which can collect crucial agricultural data and send it to an IoT platform called Thingspeak in real time where the data can be logged and analyzed. The logged data on Thingspeak is in graphical format, a botanist or a reasonably knowledged farmer can analyze the data (from anywhere in the world) to make sensible changes in the supplied resources (to crops) to obtain high quality yield.

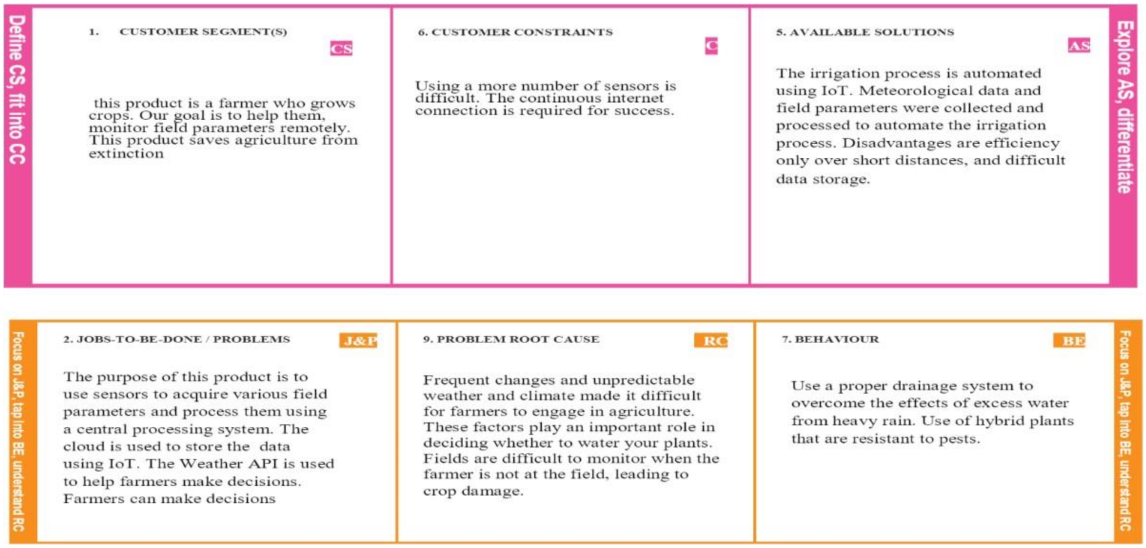
Smart agriculture monitoring system or simply smart farming is an emerging technology concept where data from several agricultural fields ranging from small to large scale and its surrounding are collected using smart electronic sensors. The collected data are analyzed by experts and local farmers to draw short term and long-term conclusion on weather pattern, soil fertility, current quality of crops, amount of water that will be required for next week to a month etc.

We can take smart farming a step further by automating several parts of farming, for example smart irrigation and water management. We can apply predictive algorithms on microcontrollers or SoC to calculate the amount of water that will be required today for a particular agriculture field. Say, if there was rain yesterday and the quantity of water required today is going to be less. Similarly, if humidity was high the evaporation of water at upper ground level is going to be less, so water required will be less than normal, thus reducing water usage.

3.3 Proposed Solution

S.No	Parameter	Description
1.	Problem Statement (Problem to be solved)	• Our project will give the solution to overcome theseproblems with help of IOT. • In agriculture, there are two major problems one is unpredictable climate change and another one is the yields of the crops that have been damaged by improper irrigation.
2.	Idea / Solution description	• It collects the data from different types of sensors and it sends the value to the main server. • It also collects the weather data from the weather API. • The ultimate decision, whether to water the crop or not is taken by the farmer using mobile application.
3.	Novelty / Uniqueness	• It depends on IOT thus eliminating the need of physical work of farmers and thus increasing the productivity in every possible manner. • The weather data are taken from the reliable source.
4.	Social Impact / Customer Satisfaction	• The information collected are from reliable sources and hence the farmer could make more precise decision, thereby the productivity increases.
5.	Business Model (Revenue Model)	• Smart farming is an advanced and innovative way to get maximum cultivation and minimize the human efforts.
6.	Scalability of the Solution	• Automatic farming equipment adjustment is made feasible by integrating information such as crops/weather and equipment to automatically alter temperature, humidity, andso on. • With the use of sensors, it has enabled Farmers to reduce waste and increase output.

3.4 Problem solution fit



4. Requirement Analysis

4.1 Functional Requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Software	MIT app, Web UI, Node-red, IBM Watson
FR-2	IoT devices	Node mcu, Required sensors
FR-3	3D printer	Make the cover case

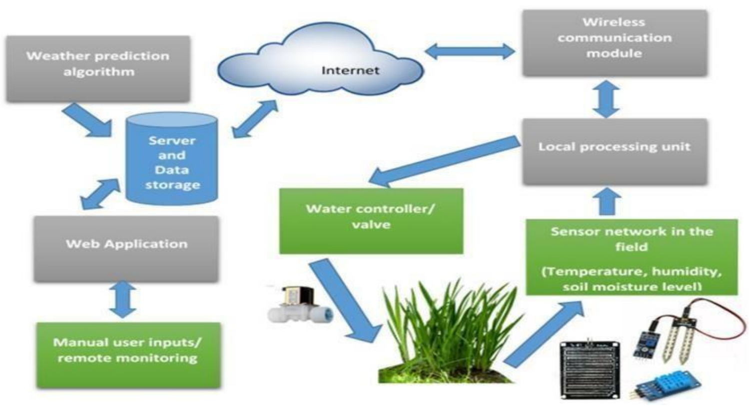
5. PRODUCT DESIGN

5.1 Data flow diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

- The different soil parameters temperature, soil moistures and then humidity are sensed using different sensors and obtained value is stored in the IBM cloud.

- Arduino UNO is used as a processing Unit that process the data obtained from the sensors and whether data from the weather API.
- NODE-RED is used as a programming tool to write the hardware, software, and APIs. The MQTT protocol is followed for the communication.
- All the collected data are provided to the user through a mobile application that was developed using the MIT app inventor. The user could plan through an app, weather to water the crop or not depending upon the sensor values. By using the app they can remotely operate to the motor switch.



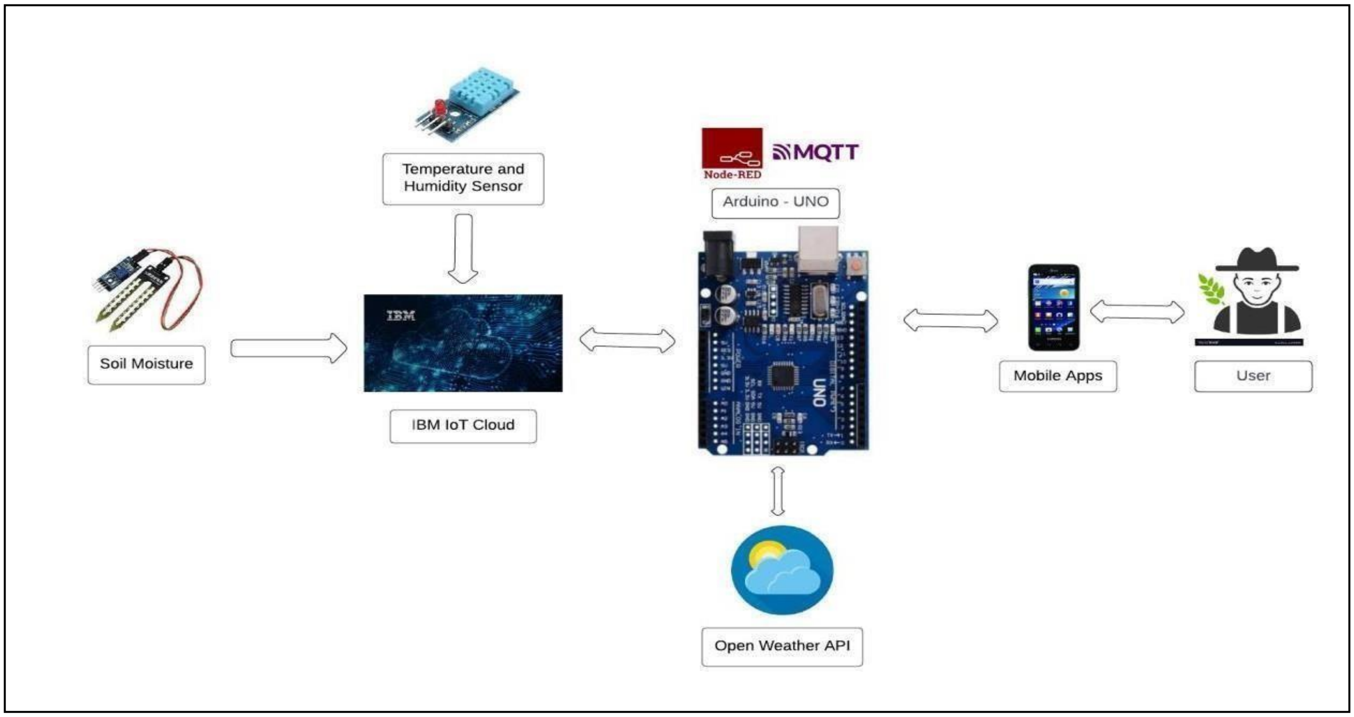
5.2 Solution and Technical Architecture

The different soil parameters (temperature, humidity, Soil Moisture) are sensed using different sensors, and the obtained value is stored in the IBM cloud.

Arduino UNO is used as a processing unit that processes the data obtained from sensors and weather data from weather API.

Node-red is used as a programming tool to wire the hardware, software, and APIs.

The MQTT protocol is followed for communication.



6. PROJECT PLANNING AND SCHEDULING

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer	IoT devices	USN-1	Node mcu kit, Sensors	User can setup easly	High	Sprint-1
Customer	Software	USN-2	IBM Watson IoT platform, Workflows for IoT scenarios using NodeMCU	User know the parameter	High	Sprint-1
Customer	Web UI	USN-3	Web UI Login setps	User can access the app for the data services	High	Sprint-2
Customer	MIT app	USN-4	To develop an application user interface using MIT		High	Sprint-1

7. CODING AND SOLUTIONING

7.1 Feature

```
import time

import sys

import ibmiotf.application

import ibmiotf.device

import random


#Provide your IBM Watson Device Credentials

organization = "w1v28e"

deviceType = "raspberrypi"

deviceId = "sk40"

authMethod = "token"

authToken = "110319106040"


def myCommandCallback (cmd):

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

    status=cmd.data['command']

    if status== "motoron":

        print ("motor is on")

    elif status == "motorff":

        print ("motor is off")

    else:
```

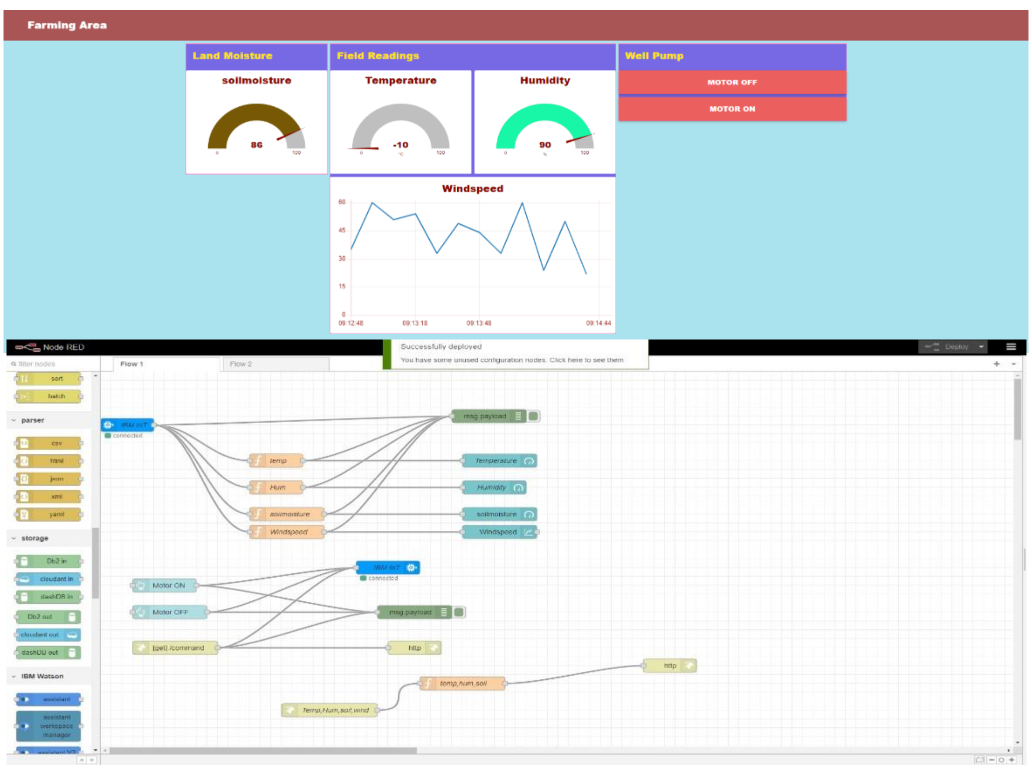
```
        print ("please send proper command")


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 evention connecting device: %s" % str(e))
    sys.exit()



deviceCli.connect()
while True:
    temp=random.randint (-10,100)
    Humid=random.randint (40,100)
    soilmoisture=random.randint (10,100)
    Windspeed_kmh=random.randint (15,60)
    data = {'temp': temp,'Humid': Humid,'soilmoisture': soilmoisture,'Windspeed_kmh':
Windspeed_kmh}
    def myonPublishCallback():
        print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % Humid,"soilmoisture
= %s" % soilmoisture,"Windspeed_kmh = %s NTU" % Windspeed_kmh, "to IBM Watson")
    success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myonPublishCallback)
```

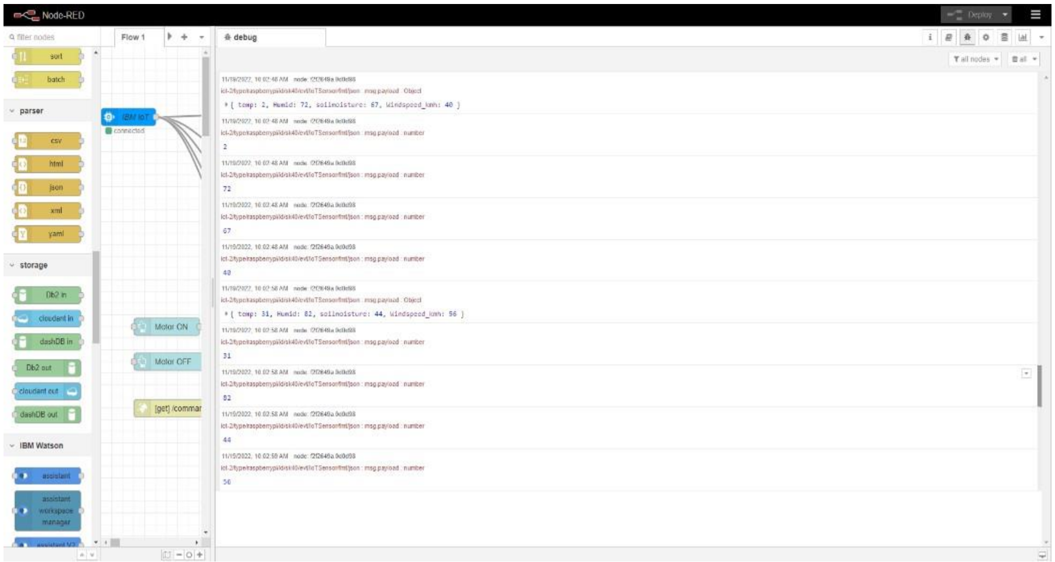
```
    if not success:
        print("Not connected to IOTF")
    time.sleep (10)
    deviceCli.commandCallback = myCommandCallback
deviceCli.disconnect()
```

8. TESTING

8.1 Test case

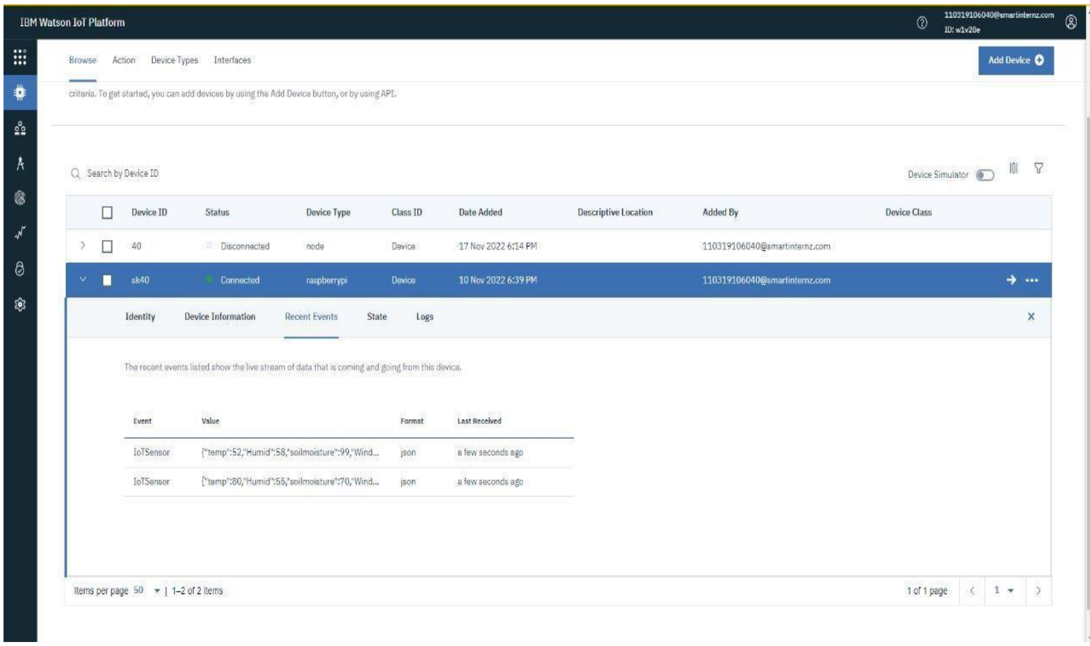
Web application using Node Red

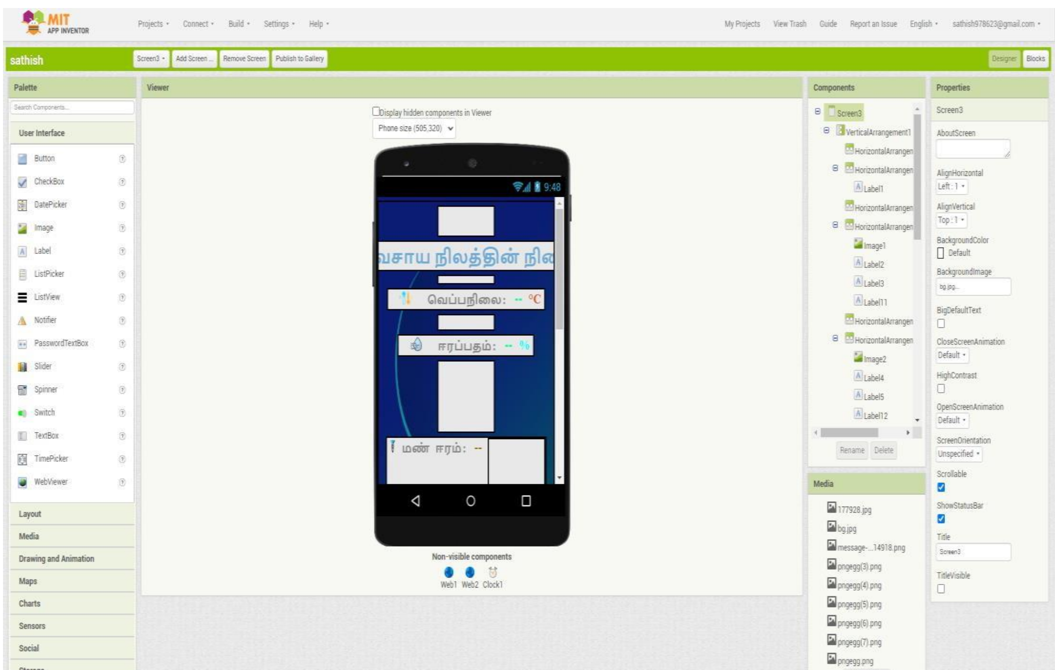


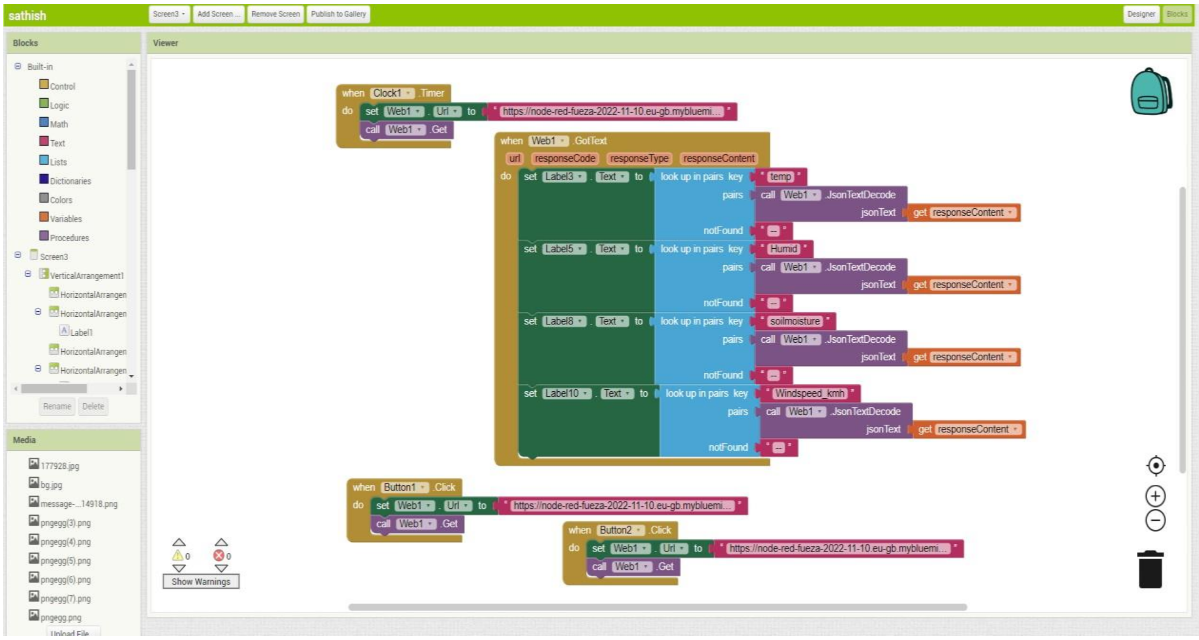


Node-RED
Deploy
filter nodes
Flow 1
debug
sort
batch
parser
csv
html
json
xml
yaml
storage
Db2 in
cloudant in
dashDB in
Db2 out
cloudant out
dashDB out
IBM Watson
assistant
assistant workspace manager
IBM IoT
connected
Motor ON
Motor OFF
[get] /commar
all nodes
all

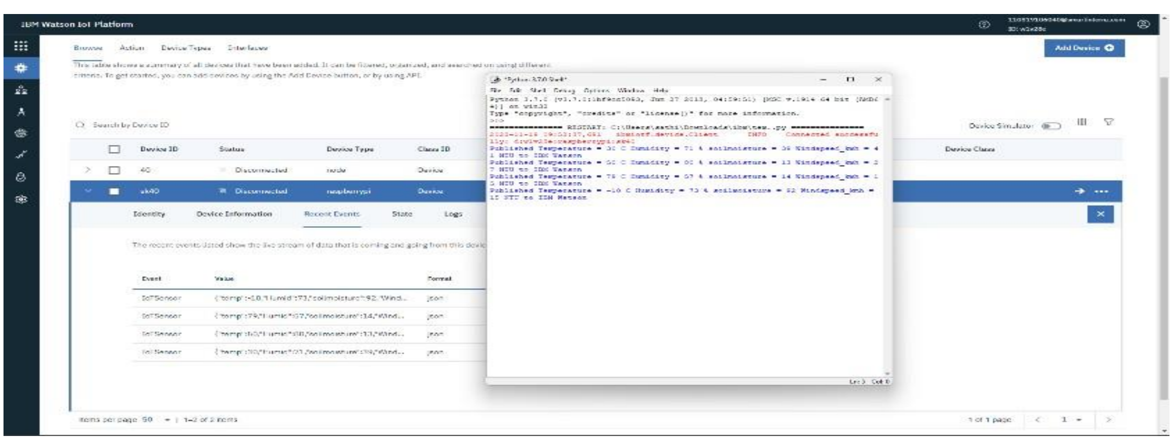
tem.py - C:\Users\sathi\Downloads\ibm\tem.py (3.7.0)
File Edit Format Run Options Window Help
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
#Provide your IBM Watson Device Credentials
organization = "wlv28e"
deviceType = "raspberrypi"
deviceId = "sk40"
authMethod = "token"
authToken = "110319106040"
def myCommandCallback (cmd):
print ("Command received: %s" % cmd.data['command'])
status=cmd.data['command']
if status== "motoron":
print ("motor is on")
elif status == "motorff":
print ("motor is off")
else:
print ("please send proper command")
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()
deviceCli.connect()
while True:
temp=random.randint (-10,100)
Humid=random.randint (40,100)
soilmoisture=random.randint (10,100)
Windspeed_kmh=random.randint (15,60)
data = {'temp': temp,'Humid': Humid,'soilmoisture': soilmoisture,'Windspeed_kmh': Windspeed_kmh}
def myonPublishCallback():
print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % Humid,"soilmoisture = %s" % soilmoisture,"Windspeed_kmh = %s NTU" % Windspeed_kmh, "to IBM Watson")
success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myonPublishCallback)
if not success:
print("Not connected to IOTF")
time.sleep (10)
deviceCli.commandCallback = myCommandCallback
deviceCli.disconnect()
Ln: 1 Col: 0


IBM Watson IoT Platform
Browse
Action
Device Types
Interfaces
Add Device
criteria. To get started, you can add devices by using the Add Device button, or by using API.
Search by Device ID
Device Simulator
Device ID | Status | Device Type | Class ID | Date Added | Descriptive Location | Added By | Device Class
40 | Disconnected | node | Device | 17 Nov 2022 6:14 PM | 110319106040@smartinternz.com
sk40 | Connected | raspberrypi | Device | 10 Nov 2022 6:39 PM | 110319106040@smartinternz.com
Identity
Device Information
Recent Events
State
Logs
The recent events listed show the live stream of data that is coming and going from this device.
Event | Value | Format | Last Received
IoTSensor | {"temp":52,"Humid":58,"soilmoisture":99,"Wind... | json | a few seconds ago
IoTSensor | {"temp":80,"Humid":55,"soilmoisture":70,"Wind... | json | a few seconds ago
Items per page 50 | 1–2 of 2 items
1 of 1 page


MIT APP INVENTOR
Projects
Connect
Build
Settings
Help
My Projects
View Trash
Guide
Report an Issue
English
sathish
Screen3
Add Screen...
Remove Screen
Publish to Gallery
Designer
Blocks
Palette
User Interface
Button
CheckBox
DatePicker
Image
Label
ListPicker
ListView
Notifier
PasswordTextBox
Slider
Spinner
Switch
TextBox
TimePicker
WebViewer
Layout
Media
Drawing and Animation
Maps
Charts
Sensors
Social
Storage
Viewer
Display hidden components in Viewer
வசாய நிலத்தின் நில
வெப்பநிலை: -- °C
ஈரப்பதம்: -- %
மண் ஈரம்: --
Non-visible components
Web1
Web2
Clock1
Components
Screen3
VerticalArrangement1
Label1
Image1
Label2
Label3
Label11
Image2
Label4
Label5
Label12
Rename
Delete
Media
177928.jpg
bg.jpg
pngegg(3).png
pngegg(4).png
pngegg(5).png
pngegg(6).png
pngegg(7).png
pngegg.png
Properties
Screen3
AboutScreen
AlignHorizontal
AlignVertical
BackgroundColor
BackgroundImage
BigDefaultText
CloseScreenAnimation
HighContrast
OpenScreenAnimation
ScreenOrientation
Scrollable
ShowStatusBar
Title
TitleVisible

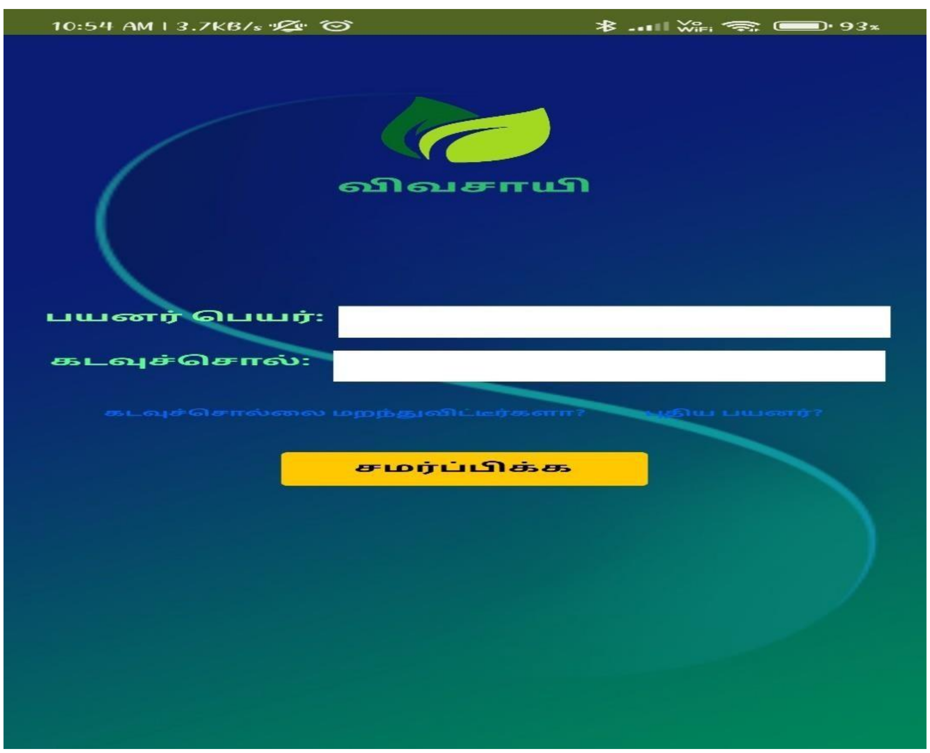


8.2 User Acceptance Testing



9.RESULTS

9.1 Performance Metrics



10. Advantages and 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 laborintensive. 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.

11. 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.

12. 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 thefts.

13. Appendix Source Code

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random

#Provide your IBM Watson Device Credentials
organization = "w1v28e"
deviceType = "raspberrypi"
deviceId = "sk40"
authMethod = "token"
authToken = "110319106040"

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

```
    if status== "motoron":
        print ("motor is on")
    elif status == "motorff":
        print ("motor is off")
    else:
        print ("please send proper command")

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 evention connecting device: %s" % str(e))
    sys.exit()



deviceCli.connect()
while True:
    temp=random.randint (-10,100)
    Humid=random.randint (40,100)
    soilmoisture=random.randint (10,100)
    Windspeed_kmh=random.randint (15,60)
```

```
    data = {'temp': temp,'Humid': Humid,'soilmoisture': soilmoisture,'Windspeed_kmh': Windspeed_kmh}

    def myonPublishCallback():

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

    success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myonPublishCallback)

    if not success:

        print("Not connected to IOTF")

    time.sleep (10)

    deviceCli.commandCallback = myCommandCallback

deviceCli.disconnect()
```

Github link:

https://github.com/IBM-EPBL/IBM-Project-9726-1659070690

Project Demo link:

https://drive.google.com/file/d/1R0rVuZ92n4gOFCRapc7hozJV5emOLd8u/vie

w?usp=share_link