

***IBM NALAIYATHIRAN
(HX8001)***

PROJECT REPORT

On

***Smart Farmer - IoT Enabled Smart Farming
Application***

TEAM ID: PNT2022TMID26544

TEAM MEMBERS:

<i>Sl.no</i>	<i>NAME</i>	<i>REGISTER.NO</i>
1.	SINGAREDDY MANEESH	211719106078
2.	PRAVEEN KUMAR	211719106060
3.	YOGESH SAI	211719106098
4.	VENKATESHAN	211719106094

in partial fulfillment for the award of the degree Of

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

RAJALAKSHMI INSTITUTE OF TECHNOLOGY

Chennai 600025.

INDEX

1. *INTRODUCTION*
2. *LITERATURE SURVEY*
3. *IDEATION & PROPOSED SOLUTION*
4. *REQUIREMENT ANALYSIS*
5. *PROJECT DESIGN*
6. *PROJECT PLANNING & SCHEDULING*
7. *CODING & SOLUTIONING*
8. *TESTING*
9. *RESULTS*
10. *ADVANTAGES & DISADVANTAGES*
11. *CONCLUSION*
12. *FUTURE SCOPE*
13. *APPENDIX*

INTRODUCTION

Internet of Things (IoT) technology has brought revolution to each and every field of common man's life by making everything smart and intelligent. IoT refers to a network of things which make a self configuring network. The development of Intelligent Smart Farming IoT based devices is day by day turning the face of agriculture production by not only enhancing it but also making it cost-effective and reducing wastage. The aim / objective of this report is to propose IoT based Smart Farming System assisting farmers in getting Live Data (Temperature, Soil Moisture) for efficient environment monitoring which will enable them to increase their overall yield and quality of products. The IoT based Smart Farming System being proposed via this report is integrated with Arduino Technology mixed with different Sensors and a Wifi module producing live data feed that can be obtained online using MIT app inventor.

LITERATURE SURVEY

1. TOPIC : IoT-Enabled Smart Agriculture

AUTHOR : Vu Khanh Quy , Nguyen Van Hau , Dang Van

DESCRIPTION : The growth of the global population coupled with a decline in natural resources, farmland, and the increase in unpredictable environmental conditions leads to food security is becoming a major concern for all nations worldwide. These problems are motivators that are driving the agricultural industry to transition to smart agriculture with the application of the Internet of Things (IoT) and big data solutions to improve operational efficiency and productivity. The IoT integrates a series of existing state-of-the-art solutions and technologies, such as

wireless sensor networks, cognitive radio ad hoc networks, cloud computing, big data, and end-user applications. This study presents a survey of IoT solutions and demonstrates how IoT can be integrated into the smart agriculture sector. To achieve this objective, we discuss the vision of IoT-enabled smart agriculture ecosystems by evaluating their architecture (IoT devices, communication technologies, big data storage, and processing), their applications, and research timeline. In addition, we discuss trends and opportunities of IoT applications for smart agriculture and also indicate the open issues and challenges of IoT application in smart agriculture.

2.TOPIC : Smart Farm Monitoring Using Raspberry Pi and Arduino

AUTHOR : Siwakorn Jindarat, Pongpisitt Wuttidittachotti

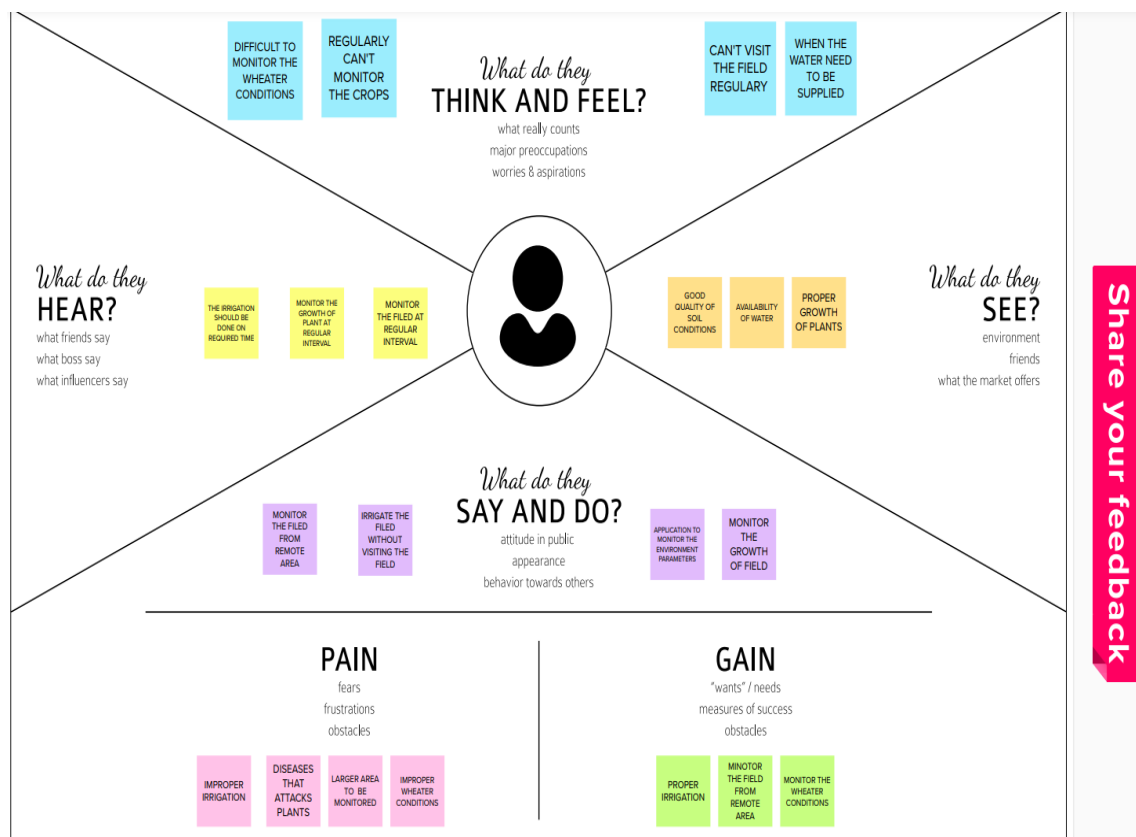
DESCRIPTION : This study aimed to investigate an establishment using an Intelligent System which employed an Embedded System and Smart Phone for chicken farming management and problem solving using Raspberry Pi and Arduino Uno. An experiment and comparative analysis of the intelligent system was applied in a sample chicken farm in this study. The findings of this study found that the system could monitor surrounding weather conditions including humidity, temperature, climate quality, and also the filter fan switch control in the chicken farm. The system was found to be comfortable for farmers to use as they could effectively control the farm anywhere at anytime, resulting in cost reduction, asset saving, and productive management in chicken farming.

PUBLISHED IN : 2015 IEEE 2015 International Conference on computer

IDEATION & PROPOSED SOLUTION

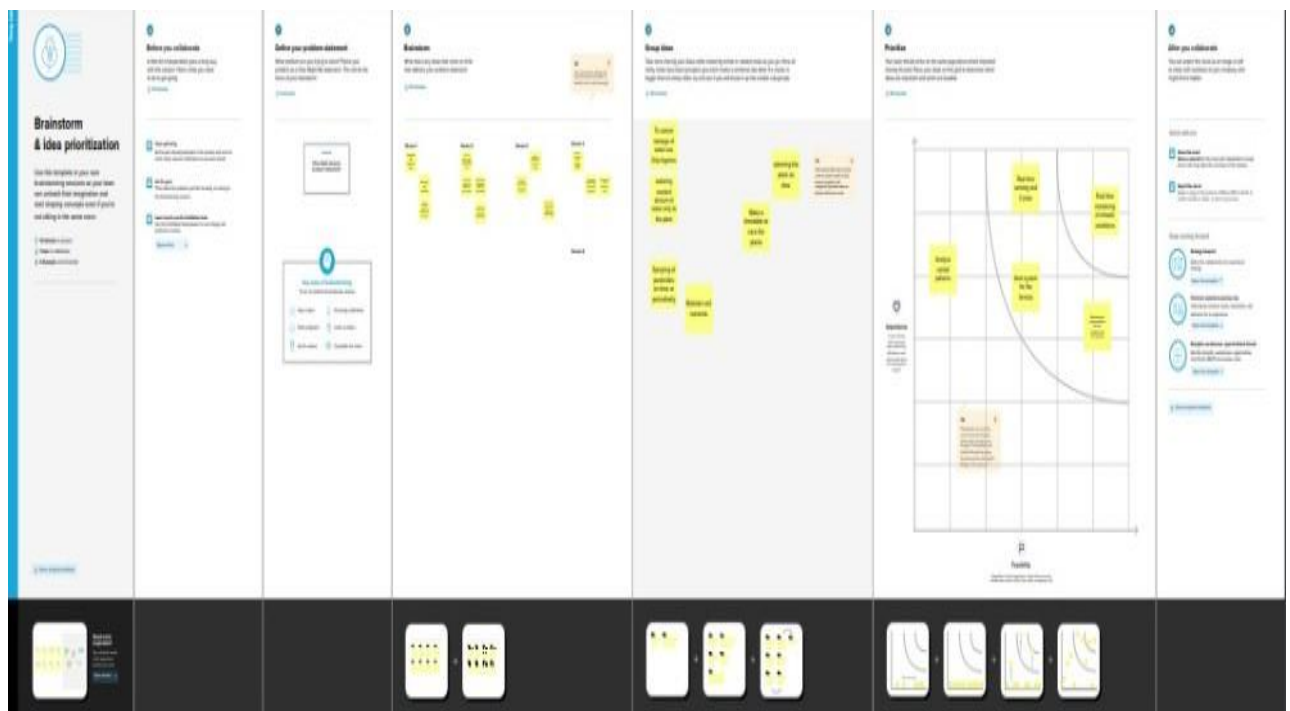
1. Empathy Map Canvas:

An empathy map is a collaborative tool teams can use to gain a deeper insight into their customers. Much like a user persona, an empathy map can represent a group of users, such as a customer segment. The empathy map was originally created by [Dave Gray](#) and has gained much popularity within the agile community



2. Ideation & Brainstorming:

Ideation essentially refers to the whole creative process of coming up with and communicating new ideas. Ideation is innovative thinking, typically aimed at solving a problem or providing a more efficient means of doing or accomplishing something. It encompasses thinking up new ideas, developing existing ideas, and figuring out means or methods for putting new ideas into practice. Ideation is similar to a practice known as brainstorming.



PROPOSED SOLUTION

To provide efficient decision support system using wireless sensor network which handle different activities of farm and gives useful information related to farm such as Soil moisture, Temperature and Humidity content . Smart Agricultural System solutions provide an integrated IoT platform in agriculture that allows farmers to use different types of sensors and used to collect the information of various parameter and analyse real- time data in order to make informed decisions. IoT based smart farming for Live Monitoring of Temperature and Soil Moisture has been proposed using Arduino and Cloud Computing . The System has high efficiency and accuracy in fetching the live data of temperature and soil moisture. The IoT based smart farming System being proposed via this report will assist farmers in increasing the agriculture yield

REQUIREMENT ANALYSIS

Functional Requirements:

Following are the functional requirements of the proposed solution.

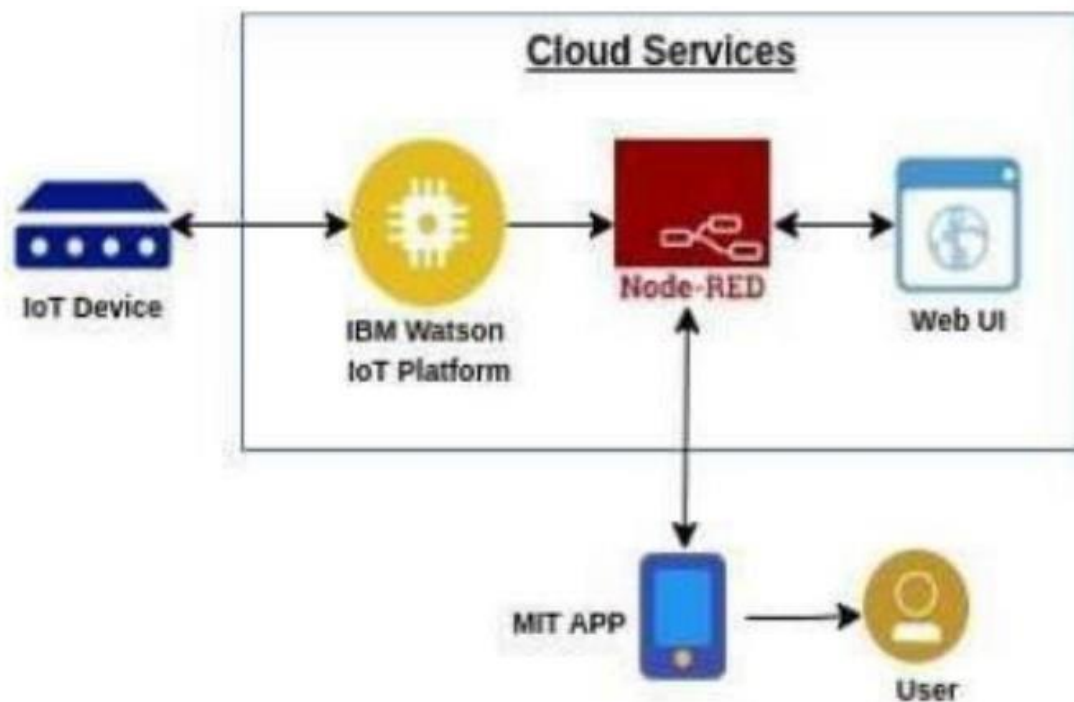
<i>FR No.</i>	<i>Functional Requirement (Epic)</i>	<i>Sub Requirement (Story / Sub-Task)</i>
<i>FR-1</i>	<i>IoT devices</i>	<i>Sensors and Wifi module.</i>
<i>FR-2</i>	<i>Software</i>	<i>Web UI, Node-red, IBM Watson, MIT app</i>

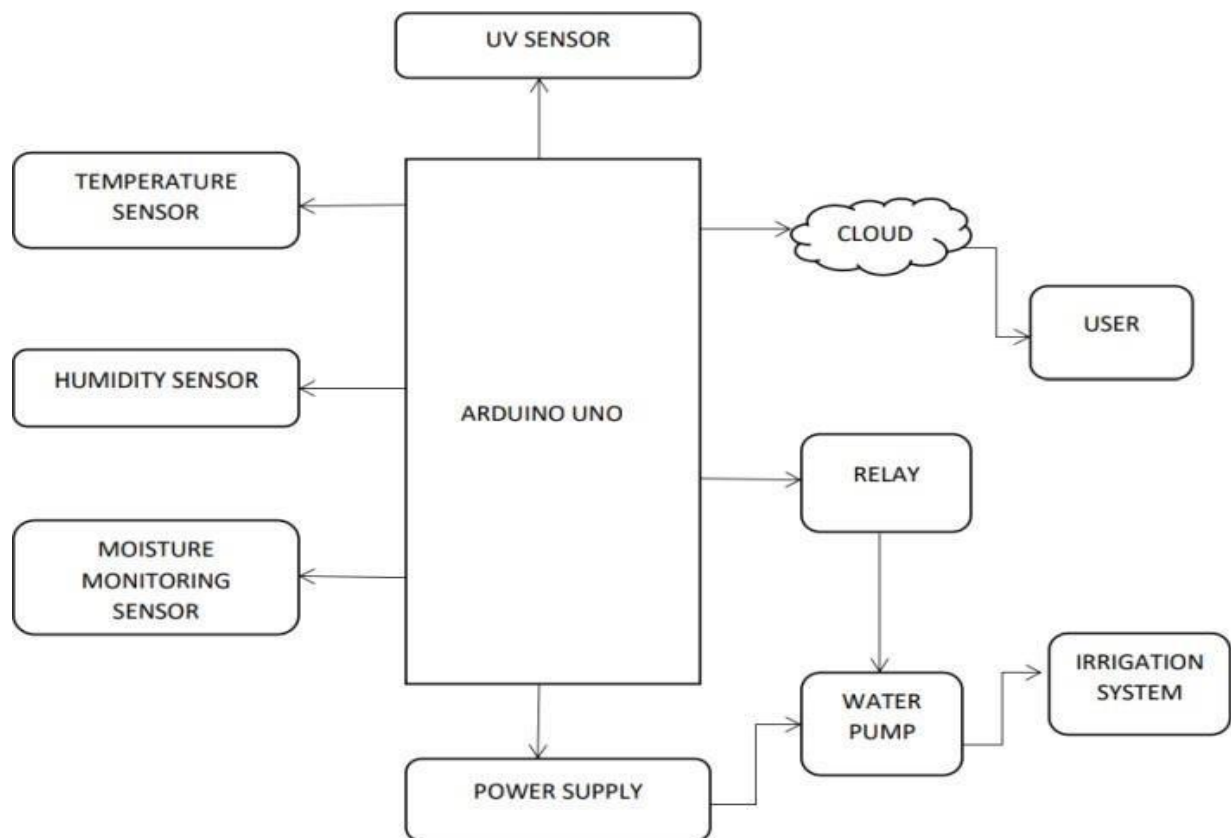
Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

<i>FR No.</i>	<i>Non-Functional Requirement</i>	<i>Description</i>
<i>NFR- 1</i>	<i>Usability</i>	<i>Time consumability is less, Productivity is high.</i>
<i>NFR- 2</i>	<i>Security</i>	<i>It has high level of security features due to integration of sensor data</i>
<i>NFR- 3</i>	<i>Reliability</i>	<i>Accuracy of data and hence it is Reliable.</i>
<i>NFR- 4</i>	<i>Performance</i>	<i>Performance is high and highly productive.</i>
<i>NFR- 5</i>	<i>Availability</i>	<i>With permitted network connectivity the application is accessible.</i>
<i>NFR- 6</i>	<i>Scalability</i>	<i>It is perfectly scalable and many new constraints can be added.</i>

PROJECT DESIGN





User Stories

Type	Requirement	User Story Number		Acceptance criteria	Priority	
(Mobile use)	Registration		As a user, I can register for the application by entering my email, password and confirming my password.	I can access my account/ dashboard	High	Spri
			As a user, I will receive confirmation email once I have registered for the application	confirmation email & click confirm.	High	Spri

(Web user)	Login		As a user, I can log into the application by entering email and password.	I can register & access the dashboard with Login	High	Sprint
	credentials		As a user, I can register for the application through mobile application	Temperature and Humidity details	Medium	Sprint
			dashboard and this dashboard include the check roles of access and then move to the manage modules.	dashboard in the smart farming application system.	Medium	Sprint
Executive			with the software.	store in cloud services.	High	Sprint
	IOT devices		As a user once view the manage modules this describes the manage system admins and Manage Roles of user and etc		Medium	Sprint
	Log out			Sign out	High	Sprint

PROJECT PLANNING & SCHEDULING

Product Backlog, Sprint Schedule, and Estimation


Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Interfacing Sensors and Motor Pump and IBM cloud	USN-1	Develop a python Code to Interface Sensors and Motor Pump and IBM cloud.	20	High	YOGESH SAI (Member 4)
Sprint-2	Node-Red	USN-2	Develop a web Application <u>Using a Node-Red</u> .	20	High	SINGAREDDY MANEESH (Leader)
Sprint-3	Mobile Application	USN-3	Develop a mobile Application using MIT-App Inventor.	20	High	VENKATESHAN R (Member 3)
Sprint-4	Integration & Testing	USN-4	Integrating Python Script, Web application & Mobile App	20	Medium	PRAVEEN KUMAR (Member 2)

Project Tracker, Velocity & Burndown Chart

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	11 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	17 Nov 2022

CODING & SOLUTIONING

Python Code:

- *For Connecting IBM Cloud*
- *For NODE RED*
- *Weather Map Information  MIT App Inventor*

Python Code

```
import wiotp.sdk.device
import time
import os
import datetime
import random
myconfig = {
    "identity": {
        "orgId": "9sg3zy",
        "typeId": "NodeMCU",
        "deviceId": "12345"
    },
    "auth": {
        "token": "IHk35N47uKF_vjA+C)"
```

```

    }
}

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 switched on")
    elif(m=="motoroff"):
        print("motor is switched OFF")
    print(" ")

while True:
    soil=random.randint(0,100)
    temp=random.randint(-20,125)
    hum=random.randint(0,100)

    myData={'soil_moisture':soil, '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.myCommandCallback = myCommandCallback

client.disconnect ()

```

```

import wiotp.sdk.device
import time
import os
import datetime
import random
myconfig = {
    "identity": {
        "orgId": "9sg3zy",
        "typeId": "NodeMCU",
        "deviceId": "12345"
    },
    "auth": {
        "token": "IHk55N47uRF_vjA+C)"
    }
}
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 switched on")
    elif(m=="motoroff"):
        print("motor is switched OFF")
    print(" ")
while True:
    soil=random.randint(0,100)
    temp=random.randint(-20,125)
    hum=random.randint(0,100)
    myData={'soil_moisture':soil, '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.myCommandCallback = myCommandCallback
client.disconnect ()

```

Running of programs

```

File Edit Shell Debug Options Window Help
Published data Successfully: %s {'soil_moisture': 74, 'temperature': 29, 'humidity': 98}
Published data Successfully: %s {'soil_moisture': 36, 'temperature': 7, 'humidity': 69}
Published data Successfully: %s {'soil_moisture': 20, 'temperature': 77, 'humidity': 56}
Published data Successfully: %s {'soil_moisture': 72, 'temperature': 103, 'humidity': 59}
Published data Successfully: %s {'soil_moisture': 77, 'temperature': -12, 'humidity': 12}
Published data Successfully: %s {'soil_moisture': 49, 'temperature': 46, 'humidity': 34}
Published data Successfully: %s {'soil_moisture': 13, 'temperature': 102, 'humidity': 29}
Published data Successfully: %s {'soil_moisture': 33, 'temperature': 12, 'humidity': 32}
Published data Successfully: %s {'soil_moisture': 47, 'temperature': 101, 'humidity': 86}
Published data Successfully: %s {'soil_moisture': 2, 'temperature': 94, 'humidity': 26}
Published data Successfully: %s {'soil_moisture': 81, 'temperature': 81, 'humidity': 73}
Published data Successfully: %s {'soil_moisture': 69, 'temperature': 18, 'humidity': 97}
Published data Successfully: %s {'soil_moisture': 96, 'temperature': 107, 'humidity': 20}
Published data Successfully: %s {'soil_moisture': 8, 'temperature': 84, 'humidity': 30}
Published data Successfully: %s {'soil_moisture': 77, 'temperature': 91, 'humidity': 99}
Published data Successfully: %s {'soil_moisture': 13, 'temperature': 78, 'humidity': 29}
Published data Successfully: %s {'soil_moisture': 0, 'temperature': 75, 'humidity': 26}
Published data Successfully: %s {'soil_moisture': 6, 'temperature': 105, 'humidity': 22}
Published data Successfully: %s {'soil_moisture': 72, 'temperature': 49, 'humidity': 16}
Published data Successfully: %s {'soil_moisture': 4, 'temperature': 113, 'humidity': 94}
Published data Successfully: %s {'soil_moisture': 14, 'temperature': 84, 'humidity': 82}
Published data Successfully: %s {'soil_moisture': 72, 'temperature': 20, 'humidity': 65}
Published data Successfully: %s {'soil_moisture': 73, 'temperature': 111, 'humidity': 58}
Published data Successfully: %s {'soil_moisture': 58, 'temperature': 97, 'humidity': 50}
Published data Successfully: %s {'soil_moisture': 48, 'temperature': -5, 'humidity': 62}
Published data Successfully: %s {'soil_moisture': 91, 'temperature': 107, 'humidity': 22}
Published data Successfully: %s {'soil_moisture': 39, 'temperature': 13, 'humidity': 83}
Published data Successfully: %s {'soil_moisture': 76, 'temperature': 38, 'humidity': 53}
Published data Successfully: %s {'soil_moisture': 43, 'temperature': 19, 'humidity': 17}
Published data Successfully: %s {'soil_moisture': 80, 'temperature': 118, 'humidity': 66}
Published data Successfully: %s {'soil_moisture': 93, 'temperature': -2, 'humidity': 6}
Published data Successfully: %s {'soil_moisture': 71, 'temperature': 19, 'humidity': 62}
Published data Successfully: %s {'soil_moisture': 45, 'temperature': 59, 'humidity': 84}
Published data Successfully: %s {'soil_moisture': 52, 'temperature': 101, 'humidity': 12}
Published data Successfully: %s {'soil_moisture': 33, 'temperature': 24, 'humidity': 74}
Published data Successfully: %s {'soil_moisture': 16, 'temperature': 11, 'humidity': 0}
Published data Successfully: %s {'soil_moisture': 93, 'temperature': 93, 'humidity': 59}
Published data Successfully: %s {'soil_moisture': 44, 'temperature': 96, 'humidity': 67}
Published data Successfully: %s {'soil_moisture': 54, 'temperature': 33, 'humidity': 52}

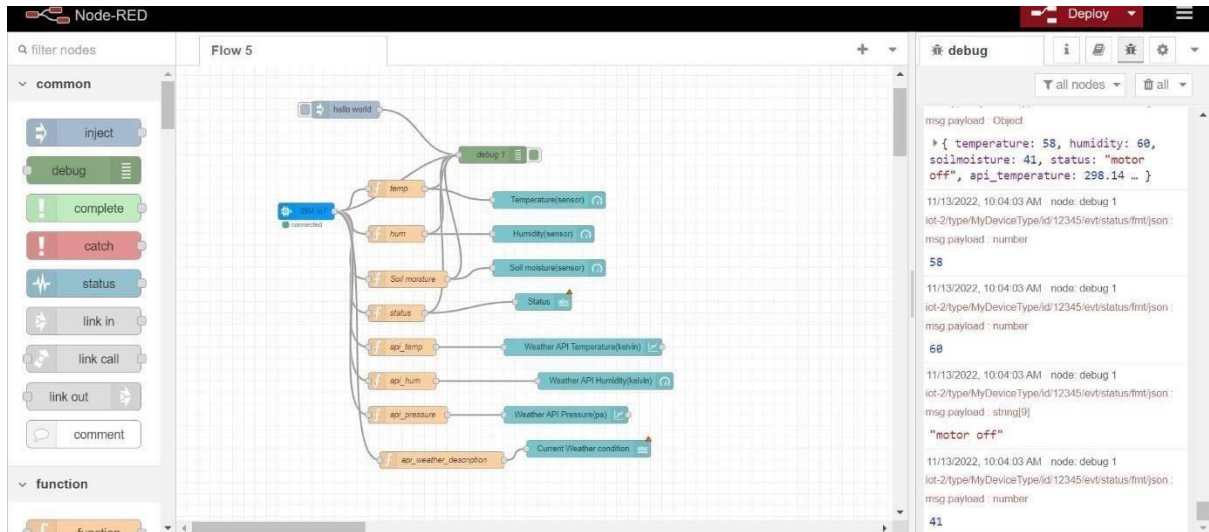
```

TESTING & RESULTS

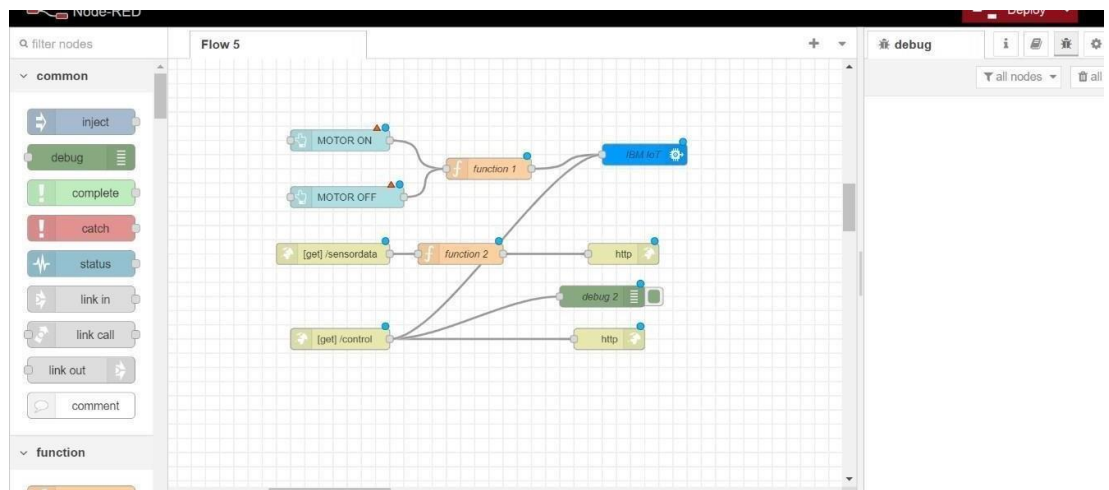
NODE RED Flow Connections

- *Interfacing IBM Cloud*
- *Intefacing & Getting Sensor Datas*
- *Connecting MIT App Inventor ☐ Weather Map Parameters*

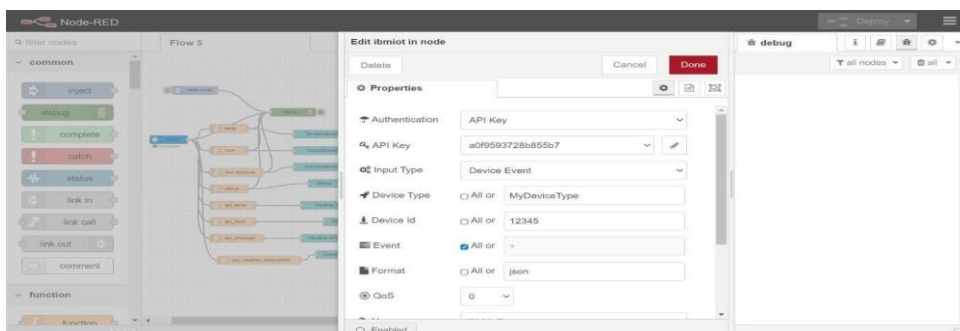
Flow:1



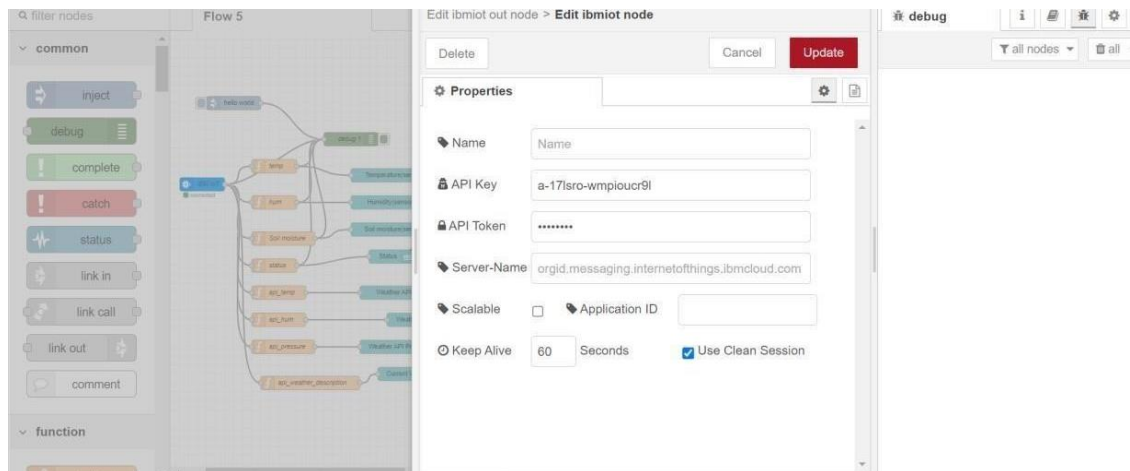
Flow:2



Flow:1 Configuring All Nodes With IBM IOT Platform



Flow:2 Configuring All Nodes With IBM IOT Platform



Execution of python program

```
erature': 298.14, 'api_pressure': 1013, 'api_humidity': 94, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 2, 'humidity': 93, 'soilmoisture': 52, 'status': 'motor off', 'api_tempe
rature': 298.14, 'api_pressure': 1013, 'api_humidity': 94, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 100, 'humidity': 100, 'soilmoisture': 63, 'status': 'motor off', 'api_te
mperature': 298.14, 'api_pressure': 1013, 'api_humidity': 94, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': -3, 'humidity': 9, 'soilmoisture': 28, 'status': 'motor off', 'api_tempe
rature': 298.14, 'api_pressure': 1013, 'api_humidity': 94, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 96, 'humidity': 93, 'soilmoisture': 24, 'status': 'motor off', 'api_tempe
rature': 298.14, 'api_pressure': 1013, 'api_humidity': 94, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': -5, 'humidity': 64, 'soilmoisture': 99, 'status': 'motor off', 'api_tempe
rature': 298.14, 'api_pressure': 1013, 'api_humidity': 94, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 8, 'humidity': 40, 'soilmoisture': 24, 'status': 'motor off', 'api_tempe
rature': 298.14, 'api_pressure': 1013, 'api_humidity': 94, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 15, 'humidity': 25, 'soilmoisture': 70, 'status': 'motor off', 'api_tempe
rature': 298.14, 'api_pressure': 1013, 'api_humidity': 94, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 116, 'humidity': 59, 'soilmoisture': 65, 'status': 'motor off', 'api_tem
perature': 298.14, 'api_pressure': 1013, 'api_humidity': 94, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 72, 'humidity': 71, 'soilmoisture': 13, 'status': 'motor off', 'api_tempe
rature': 298.14, 'api_pressure': 1013, 'api_humidity': 94, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 104, 'humidity': 82, 'soilmoisture': 90, 'status': 'motor off', 'api_tem
perature': 298.14, 'api_pressure': 1013, 'api_humidity': 94, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 63, 'humidity': 82, 'soilmoisture': 98, 'status': 'motor off', 'api_tempe
rature': 298.14, 'api_pressure': 1013, 'api_humidity': 94, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 27, 'humidity': 57, 'soilmoisture': 21, 'status': 'motor off', 'api_tempe
rature': 298.14, 'api_pressure': 1013, 'api_humidity': 94, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 107, 'humidity': 57, 'soilmoisture': 44, 'status': 'motor off', 'api_tem
perature': 298.14, 'api_pressure': 1013, 'api_humidity': 94, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': -15, 'humidity': 67, 'soilmoisture': 41, 'status': 'motor off', 'api_tem
perature': 298.14, 'api_pressure': 1013, 'api_humidity': 94, 'api_weather_description': 'mist'}
```

IBM Watson IoT Platform Device Connect & Live Data

The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A search bar and an 'Add Device' button are also present. The main table lists devices with columns for Device ID, Status, Device Type, Class ID, and Date Added. A device with ID 12345 is shown with a status of 'Disconnected' and a device type of 'NodeMCU'. Below the table, a detailed view of the device is shown, including its identity, device information, recent events, state, and logs. The device information section displays the following details:

Property	Value
Device ID	12345
Device Type	NodeMCU
Date Added	Nov 14, 2022 9:03 PM
Added By	211719106094@smartinternz.com
Connection Status	Disconnected

The bottom of the dashboard shows a taskbar with several open PDF files and a system tray indicating the time as 15:11 on 18/11/2022.

The screenshot shows the IBM Watson IoT Platform dashboard with a device with ID 12345 in a 'Connected' state. An 'Event Payload' window is open, displaying the following event details:

Event Name: status
Time Received: Nov 13, 2022 10:30 AM

The event payload is shown as a JSON object:

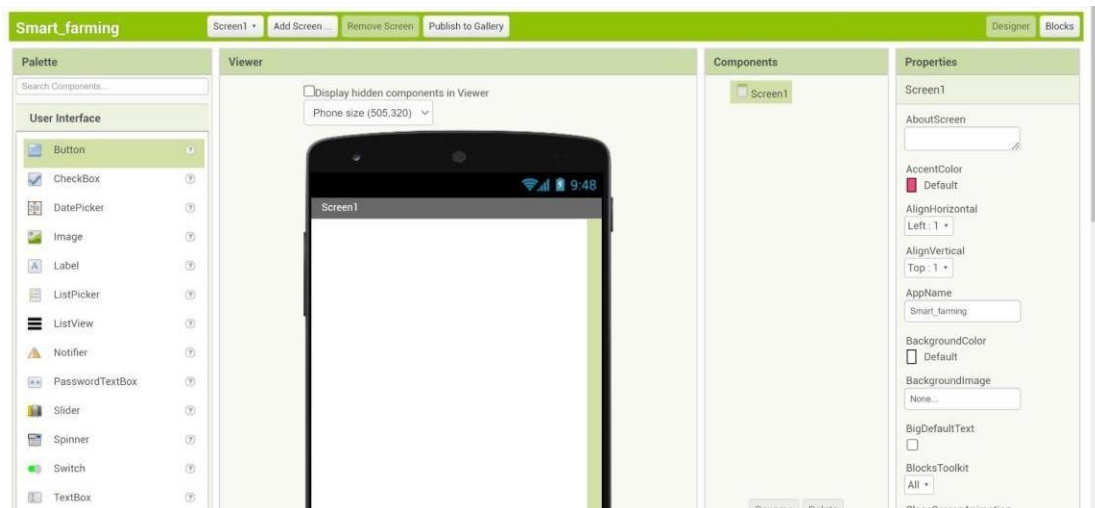
```
1 {  
2   "temperature": 118,  
3   "humidity": 84,  
4   "soilmoisture": 16,  
5   "status": "motor off",  
6   "api_temperature": 298.14,  
7   "api_pressure": 1013,  
8   "api_humidity": 94,  
9   "api_weather_description": "mist"  
10 }
```

The background dashboard shows a table of recent events for the device, with columns for Event and Value. The events listed are:

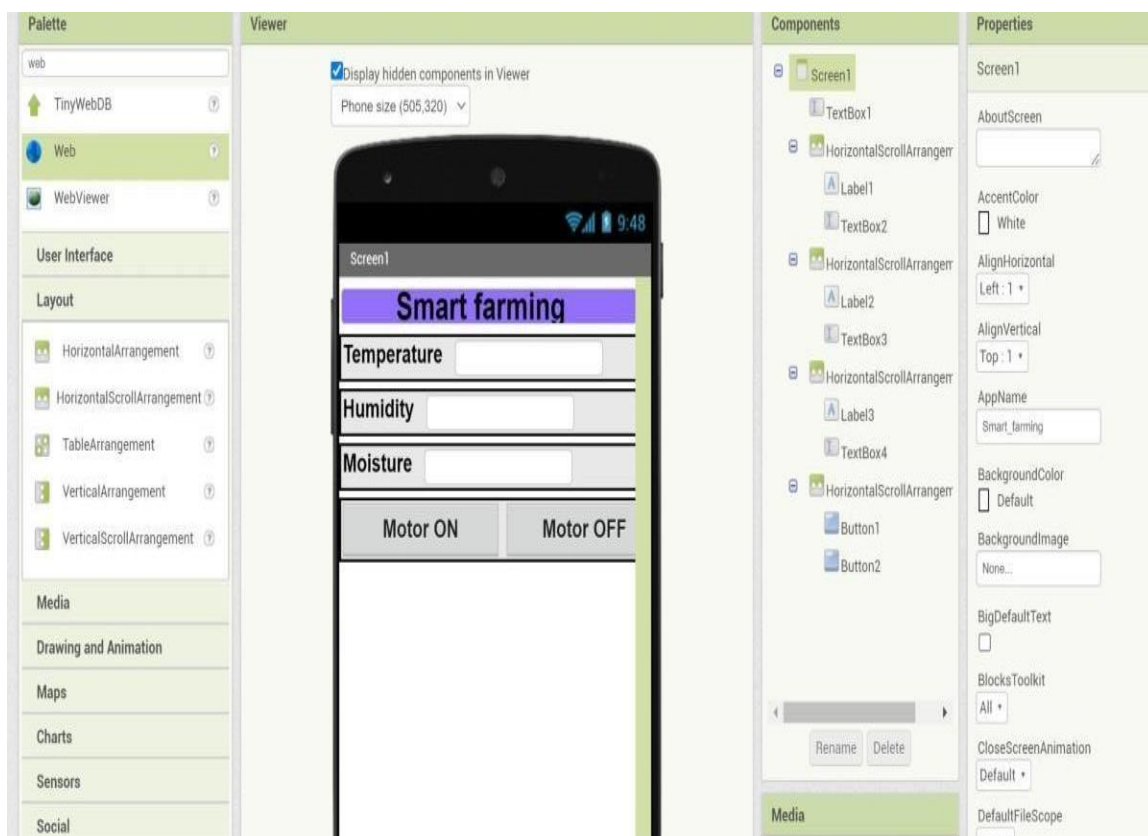
Event	Value
status	{"temperature":82,"humidity":84,"soilmoisture":16,"status":"motor off","api_temperature":298.14,"api_pressure":1013,"api_humidity":94,"api_weather_description":"mist"}
status	{"temperature":74,"humidity":84,"soilmoisture":16,"status":"motor off","api_temperature":298.14,"api_pressure":1013,"api_humidity":94,"api_weather_description":"mist"}
status	{"temperature":7,"humidity":84,"soilmoisture":16,"status":"motor off","api_temperature":298.14,"api_pressure":1013,"api_humidity":94,"api_weather_description":"mist"}
status	{"temperature":72,"humidity":84,"soilmoisture":16,"status":"motor off","api_temperature":298.14,"api_pressure":1013,"api_humidity":94,"api_weather_description":"mist"}
status	{"temperature":102,"humidity":84,"soilmoisture":16,"status":"motor off","api_temperature":298.14,"api_pressure":1013,"api_humidity":94,"api_weather_description":"mist"}

MIT APP INVENTOR

Step 1: Login Into MIT App Inventor



Step 2: Create Your User Interface By Using the Preset Tools



Step 3: Live Output In Mobile Application



The image shows a mobile application interface for smart farming. It features a purple header with the text "Smart farming". Below the header, there are three rows of data: "Temperature" with a value of "-5", "Humidity" with a value of "31", and "Moisture" with a value of "61". Each value is displayed next to a small input field. At the bottom of the interface, there are two buttons: "Motor ON" and "Motor OFF".

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 sensors will help to increase the productivity and customers can be benefited

Disadvantages:

- Cost
- Reliability
- Increased channel maintenance

CONCLUSION:

IoT based SMART FARMING SYSTEM for Live Monitoring of Temperature and Soil Moisture has been proposed using Arduino and Cloud Computing . The System has high efficiency and accuracy in fetching the live data of temperature and soil moisture. The IoT based smart farming System being proposed via this report will assist farmers in increasing the agriculture yield and take efficient care of food production as the System will always provide helping hand to farmers for getting accurate live feed of environmental temperature and soil moisture with more than 99% accurate results.

FUTURE SCOPE

Future work would be focused more on increasing sensors on this system to fetch more data especially with regard to Pest Control and by also integrating GPS module in this system to enhance this Agriculture IoT Technology to full-fledged Agriculture Precision ready product

APPENDIX

<https://github.com/IBM-EPBL/IBM-Project-22516-1659853453>