SMART FARMER- IoT ENABLED SMART FARMING APPLICATION

- BY

SRIVATSAN M

UDHAYAPRAKASH V

NIRANJANA VE

JAAJANA G

1. INTRODUCTION

1.1 Project overview

IoT-based agriculture system helps the farmer in monitoring different parameters of his field like soil moisture, temperature, and 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 purpose of doing this project is to allow the farmer to remotely monitor the field and to control the motor present in the field through the mobile application itself. IoT plays a crucial role in smart agriculture. IoT sensors are capable of providing information about agriculture fields. We have proposed an IoT and smart agriculture system. This IoT based Agriculture monitoring system makes use of wireless sensor networks that collects data from different sensors deployed at various nodes and sends it through the wireless protocol. By regular monitoring of these parameters, we can add the necessary things to the soil in order to make the crops grow without any difficulties. The temperature sensor gives the temperature value of the field. Similarly, the humidity sensor and soil moisture sensor returns the humidity & moisture content of the soil respectively. The flame sensor gives the intensity of flame , if any , present in the field. The NPK sensor gives the Nitrogen, Phosphorus & Potassium level of the soil. If any of these nutrient value is very less, then, a farmer can add that particular nutrient to the soil.

2. LITERATURE SURVEY

2.1 Existing Problem

- Manual monitoring of farm by the farmer appears to be a tedious job.

- Farmers are at the threat of facing financial risks when the crop yield is not better.

- Timely care should be given to the farm & all required components should be added to the soil if there is a need of any nutrient.

- Farmers are unaware of the new emerging technologies which help them in doing a smart agriculture.

2.2 References

[1] Bandara, Tharindu Madushan, Wanninayaka Mudiyanselage, and Mansoor Raza. "Smart farm and monitoring system for measuring the Environmental condition using wireless sensor network-IOT Technology in farming." In 2020 5th International Conference on Innovative Technologies in Intelligent Systems and Industrial Applications (CITISIA), pp. 1-7. IEEE, 2020.

This paper concentrates on implementing a smart farming system in order to facilitate farmers. They have implemented the proposed system using IoT, wireless sensor technologies and nodes to connect the entire network. The data collected and monitored from the field are soil moisture content and temperature. The designed system collect data from farms and transfer these data to central server using wireless technology and then, the server assigns task to each device present in the field based on the inputs received. They have also designed the mobile application and using this, farmers can set threshold value for each sensor.

The different modules used here are:

Soil moisture measuring system: Using this, the moisture data from the field is collected and is sent to central server through WSU (Wireless Sensor Unit) and the received data is compared with inbuilt threshold value. If the collected data is lesser than threshold value, then, the system sends an signal to actuators through WIU (Wireless Information Unit) to turn on watering system and this continues as long as data from field is greater than the threshold.

WSU (Wireless Sensor Unit): To transfer data to central server and to connect all sensors to Arduino board.

WIU (Wireless Information Unit): To communicate between sensors and WSU.

Temperature monitoring system: To monitor the temperature level in the farm.

Mobile applications to farmers: To check the moisture and temperature and to set threshold for all the parameters.

Sensor monitoring system: To check whether all the sensors are working properly or not. It is accessible only by developers.

[2] Sushanth, G., and S. Sujatha. "IOT based smart agriculture system." In 2018 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), pp. 1-4. IEEE, 2018.

This paper proposed to develop a Smart agriculture System that uses advantages of cutting edge technologies such as Arduino, IOT and Wireless Sensor Network. The paper aims at making use of evolving technology i.e. IOT and smart agriculture using automation. Monitoring environmental conditions is the major factor to improve yield of the efficient crops. The feature of this paper includes development of a system which can monitor temperature, humidity, moisture and even the movement of animals which may destroy the crops in agricultural field through sensors using Arduino board and in case of any discrepancy send a SMS notification as well as a notification on the application developed for the same to the farmer's smartphone using Wi-Fi/3G/4G. The system has a duplex communication link based on a cellular-Internet interface that allows for data inspection and irrigation scheduling to be programmed through an android application. Because of its energy autonomy and low cost, the system has the potential to be useful in water limited geographically isolated areas.

Components used

Arduino Uno R3 micro-controller board

LM 35 temperature sensor

Humidity sensor

Moisture sensor

Motion sensor

Wi-Fi module

ESP8266

GSM module

Implementation and its application :

The sensing phase involves the sensing of the physical parameters which includes temperature, moisture, humidity and motion. All these sensors are attached to the

Arduino Uno R3 micro-controller board. This board acts as the IoT gateway in the developed system as it has the capacity to transmit the data to the cloud. This transmission is done using Wi-Fi ESP8266 module.The processing phase takes place in the cloud. The cloud consists of a Web Server, a database where the sensed data is maintained and a decision logic which takes decisions based on the sensed data. In the information distribution phase, the output of the decision logic will be sent to the android application and then to the IOT gateway.

[3] Aditya Vadapalli, Swapna Peravali and Venkatarao Dadi, "Smart Agriculture system using IoT Technology" in 2020, International Journal of Advance Research in Science and Engineering.

Aim:

This paper highlights the disadvantages of conventional agricultural practices, how it drastically affects the Farmer’s life and how Smart Farming using IoT methods such as Precision farming, Efficient Water Management can solve many problems. They highlight how the foreign return Citizens without any job during the pandemic shifted to Smart Agriculture and refuse to leave for their jobs.

Components Used:

Soil Moisture Sensor

Raindrop Sensor

Temperature & Humidity Sensor (DHT11)

Arduino Uno Board

Method:

They make use of Wireless Sensor Networking System as collection process for information needed by the farmers for cultivation such as changes in environmental conditions like climate, hydrology, plant physiology, humidity, temperature, rain dampness of soil and others and also as Input feeder control system on agricultural machinery.Soil moisture sensors are fixed under the ground in field. Initially the water level reading is taken and decisions are made according to it. The temperature sensor (DTH11) is fixed at the centre of the field to get the overall reading of temperature of the soil. These sensors are connected to Arduino where we will get the readings. All sensors will send data to Arduino and data will be forwarded to WSN systems. The threshold value will be set according to the crop. The threshold value will be marked based on the requirement of the crop specified and predefined in the raspberry pi for every sensor. Whenever any sensor reaches a threshold value, message alert is sent to the user and action is taken according to it. The system has checked for the performance with the help of thing speak.com platform to check the Temperature, humidity rain and soil parameters.

[4] Harrika Pendyala, Ganeshkumar Roda, Annoja Mamidi, Madhavi Vangala, Sathyam Bonala and Keerthi Kumar Kolrapati "IoT Based Smart Agriculture Monitoring System" in 2020, International Journal of Scientific Engineering and Research (IJSER).

Aim:

This paper develops an IoT based smart agriculture monitoring system that uses the advantages of IoT is to monitor the agriculture by using the wireless sensor networks and collect the data from different sensors which are deployed at various no des and send by wireless protocol. By using IoT system the smart agriculture is powered by NodeMCU. It includes the humidity sensor, temperature sensor, moisture sensor and DC motor. This system starts to check the humidity and moisture level. The sensors are used to sense the level of water and if the level is below the range then the system automatically stars watering. According to the change in temperature level the sensor does its job. IoT also shows the information of humidity, moisture level by including date and time. The temperature level based on type of crops cultivated can also be adjusted.

Components used:

Soil moisture sensor

Temperature sensor (DHT-11)

Relay

Pump

IoT (WI-FI module ESP8266)

Power supply: 5V, 700mA Regulated power supply

Arduino IDE

Thingspeak website

Implementation and advantages:

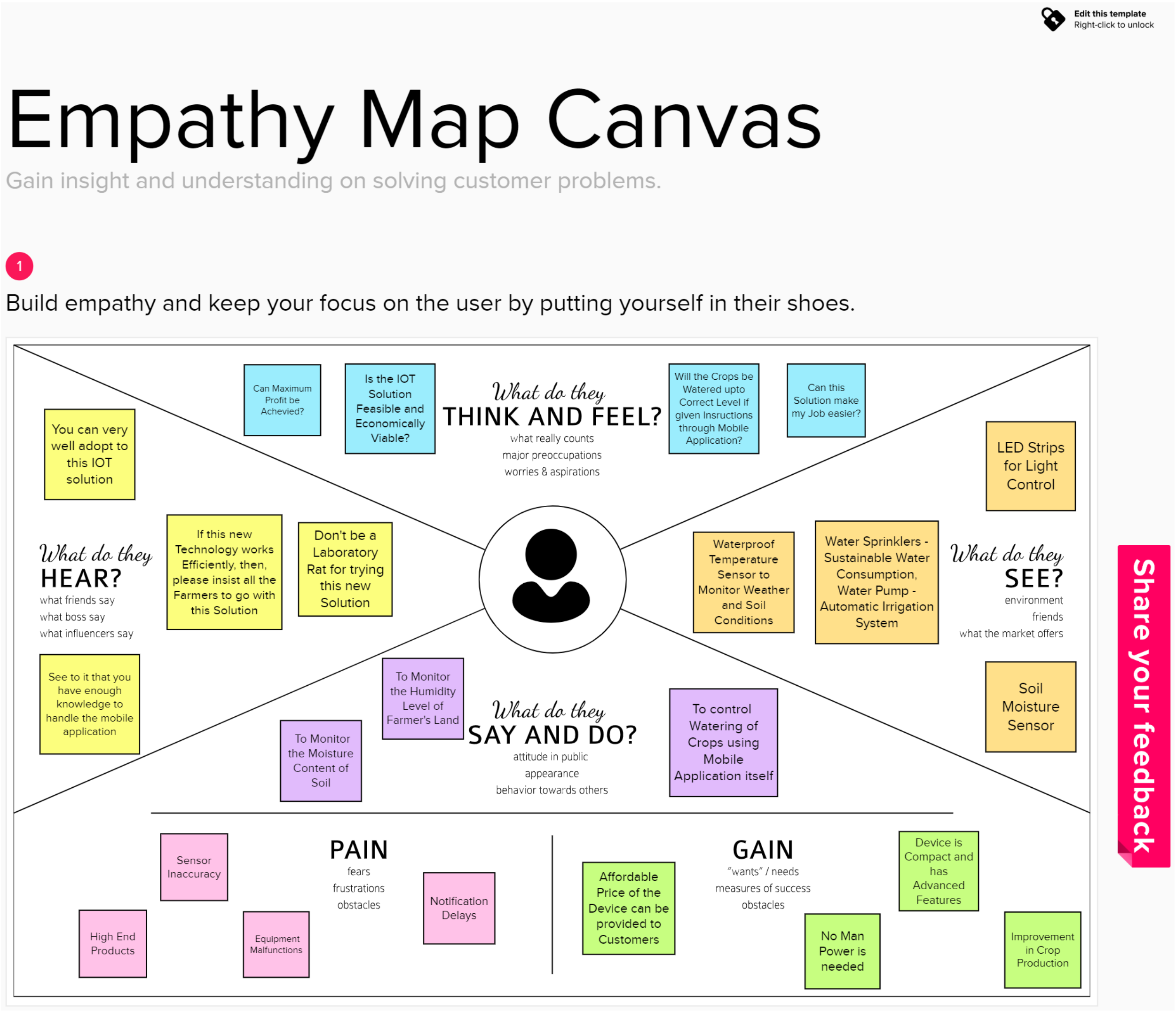
The working of the system includes soil moisture sensor and it is tested under various climatic conditions. The moisture output readings at different weather is taken and updated. Wi-Fi is used to achieve the wireless transmission. The sensed value is sent to micro-controller through NodeMCU and motor gets pump. The function of a power supply is to convert electric current from a source to the correct voltage, current and frequency to power up the load. As a result, power supplies are also referred to as electric power converters. ThingSpeak is an IoT analytics platform which is used to aggregate, visualize, and analyse live data streams in the cloud. When the data is sent to Thingspeak from the devices, it creates instant visualization of live data and sends an alert. It is easy to maintain and cost is reasonable to purchase. The components which are used are easily available. It has advantage to observe the status on smart phone or laptop using internet. The information is up to date even in absence of farmer.

2.3 Problem statement definition

Agriculture is considered as the basis of life for the human species as it is the main source of food grains and other raw materials. Hence there is need to implement modern science and technology in the agriculture sector for increasing the yield. We get better yield if we monitor the agriculture field carefully. Also, financial risk is one of the major problems of the farmers. If their crops are damaged by severe climatic conditions i.e. if the weather is too hot on one particular day, then, farmer should ensure that plants are watered sufficiently so that plants don’t die because of lack of water. Additionally, the farmer should periodically check the soil quality of his land and add required nutrients to the farmland. Then, farmer should also ensure that all the available resources are utilized efficiently. The only solution for all these problems is smart agriculture by modernizing the current traditional methods of agriculture. Hence, the project aims at making agriculture smart using automation and IoT technologies. Controlling of some operations like humidity level, temperature level monitoring will be done through any remote smart device or computer connected to Internet and the required operations will be performed by interfacing sensors and actuators with micro-controller. Data analytics tools are used for making predictions and decisions.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



Edit this template
Right-click to unlock

Empathy Map Canvas

Gain insight and understanding on solving customer problems.

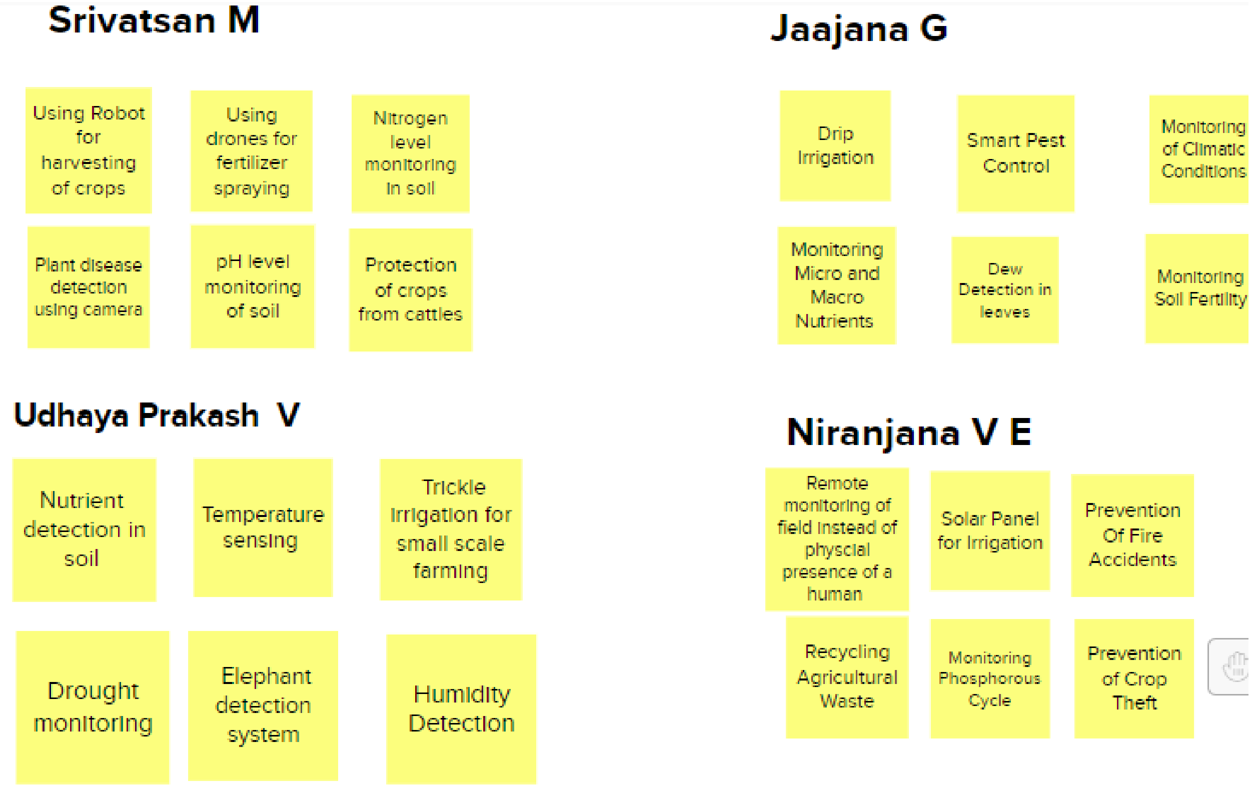
1

Build empathy and keep your focus on the user by putting yourself in their shoes.

What do they THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations
Can Maximum Profit be Achevied?
Is the IOT Solution Feasible and Economically Viable?
Will the Crops be Watered upto Correct Level if given Insructions through Mobile Application?
Can this Solution make my Job easier?
What do they HEAR?
what friends say
what boss say
what influencers say
You can very well adopt to this IOT solution
If this new Technology works Efficiently, then, please insist all the Farmers to go with this Solution
Don't be a Laboratory Rat for trying this new Solution
See to it that you have enough knowledge to handle the mobile application
What do they SEE?
environment
friends
what the market offers
LED Strips for Light Control
Waterproof Temperature Sensor to Monitor Weather and Soil Conditions
Water Sprinklers - Sustainable Water Consumption, Water Pump - Automatic Irrigation System
Soil Moisture Sensor
What do they SAY AND DO?
attitude in public
appearance
behavior towards others
To Monitor the Humidity Level of Farmer's Land
To Monitor the Moisture Content of Soil
To control Watering of Crops using Mobile Application itself
PAIN
fears
frustrations
obstacles
Sensor Inaccuracy
High End Products
Equipment Malfunctions
Notification Delays
GAIN
"wants" / needs
measures of success
obstacles
Affordable Price of the Device can be provided to Customers
No Man Power is needed
Device is Compact and has Advanced Features
Improvement in Crop Production
Share your feedback

3.2 Ideation & Brainstorming

Brainstorming:



Srivatsan M

Using Robot for harvesting of crops

Using drones for fertilizer spraying

Nitrogen level monitoring in soil

Plant disease detection using camera

pH level monitoring of soil

Protection of crops from cattles

Jaajana G

Drip Irrigation

Smart Pest Control

Monitoring of Climatic Conditions

Monitoring Micro and Macro Nutrients

Dew Detection in leaves

Monitoring Soil Fertility

Udhaya Prakash V

Nutrient detection in soil

Temperature sensing

Trickle irrigation for small scale farming

Drought monitoring

Elephant detection system

Humidity Detection

Niranjana V E

Remote monitoring of field instead of physcial presence of a human

Solar Panel for Irrigation

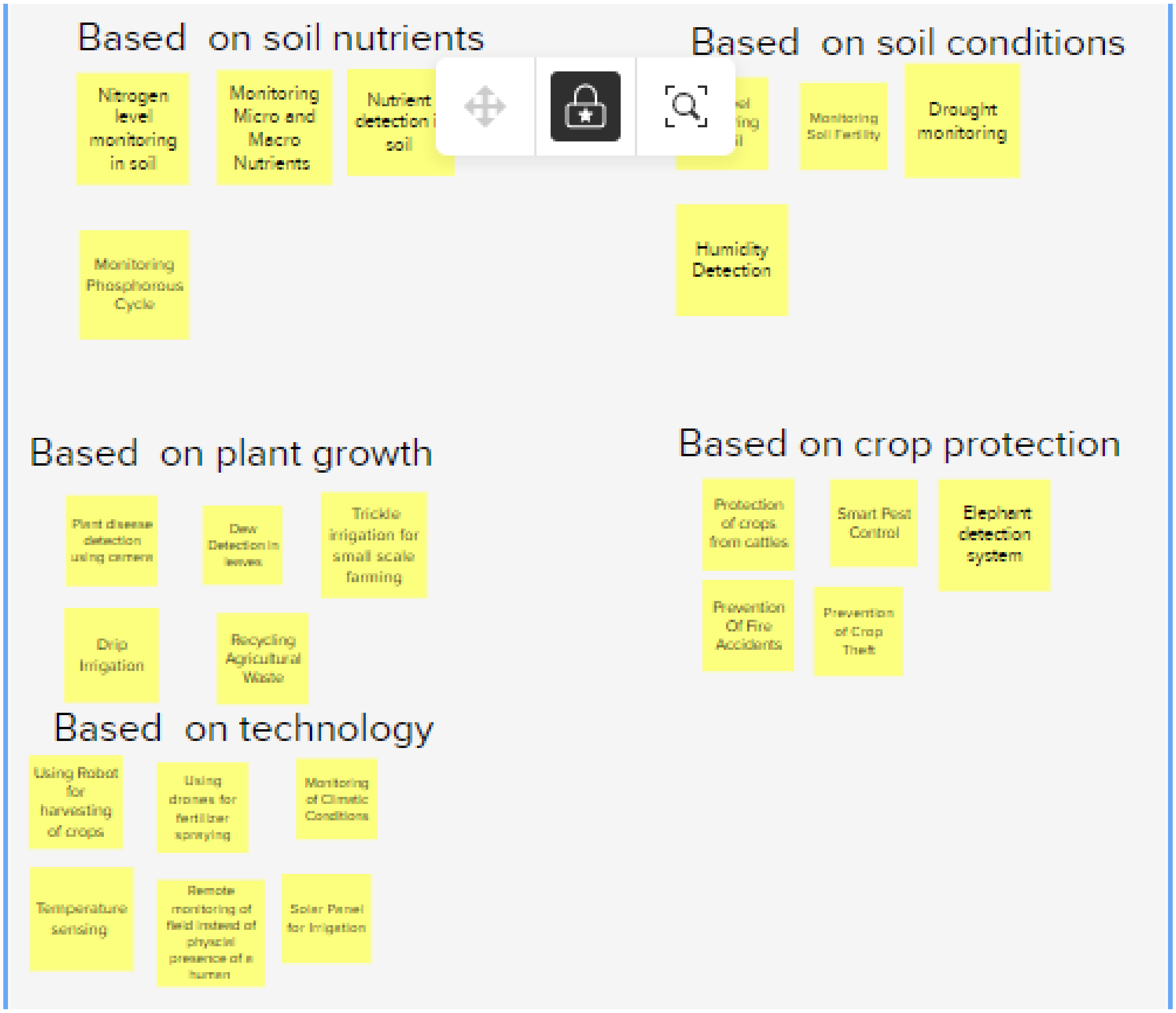
Prevention Of Fire Accidents

Recycling Agricultural Waste

Monitoring Phosphorous Cycle

Prevention of Crop Theft

Group ideas:



Based on soil nutrients

- Nitrogen level monitoring in soil
- Monitoring Micro and Macro Nutrients
- Nutrient detection i soil
- Monitoring Phosphorous Cycle

Based on soil conditions

- el ing il
- Monitoring Soil Fertility
- Drought monitoring
- Humidity Detection

Based on plant growth

- Plant disease detection using camera
- Dew Detection in leaves
- Trickle irrigation for small scale farming
- Drip Irrigation
- Recycling Agricultural Waste

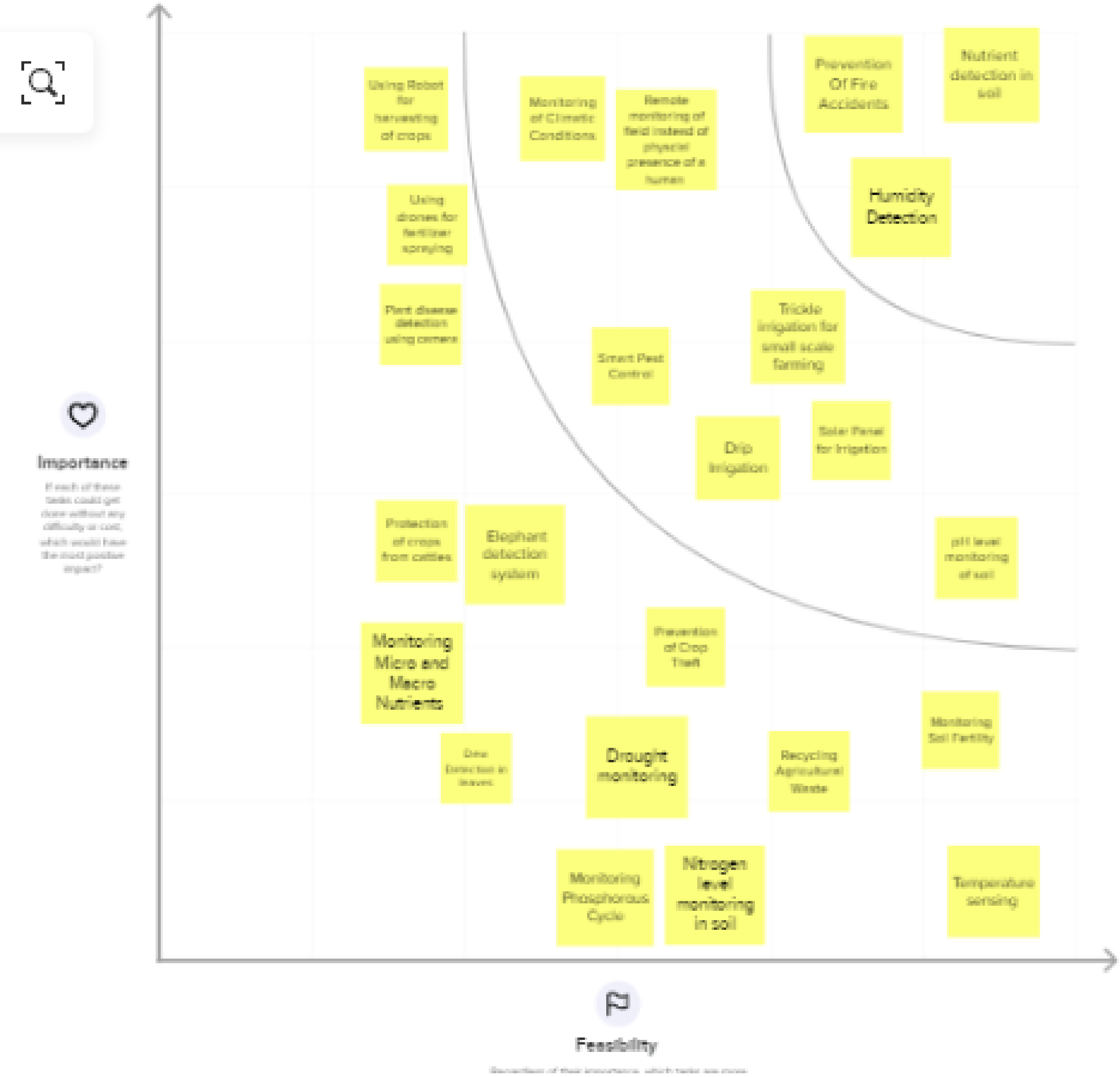
Based on crop protection

- Protection of crops from cattles
- Smart Pest Control
- Elephant detection system
- Prevention Of Fire Accidents
- Prevention of Crop Theft

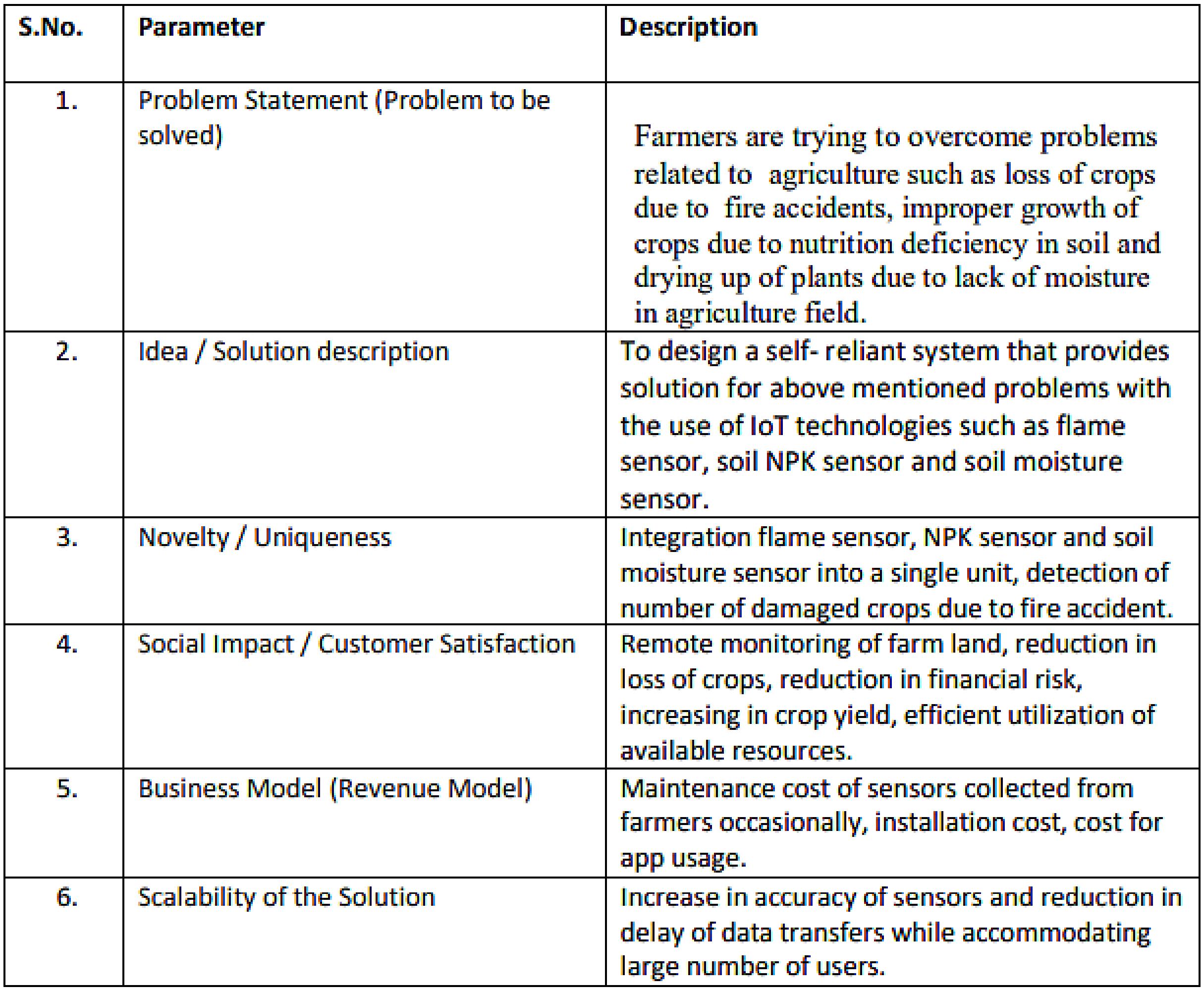
Based on technology

- Using Robot for harvesting of crops
- Using drones for fertilizer spraying
- Monitoring of Climatic Conditions
- Temperature sensing
- Remote monitoring of field instead of physcial presence of a human
- Solar Panel for Irrigation

Prioritization of idea:

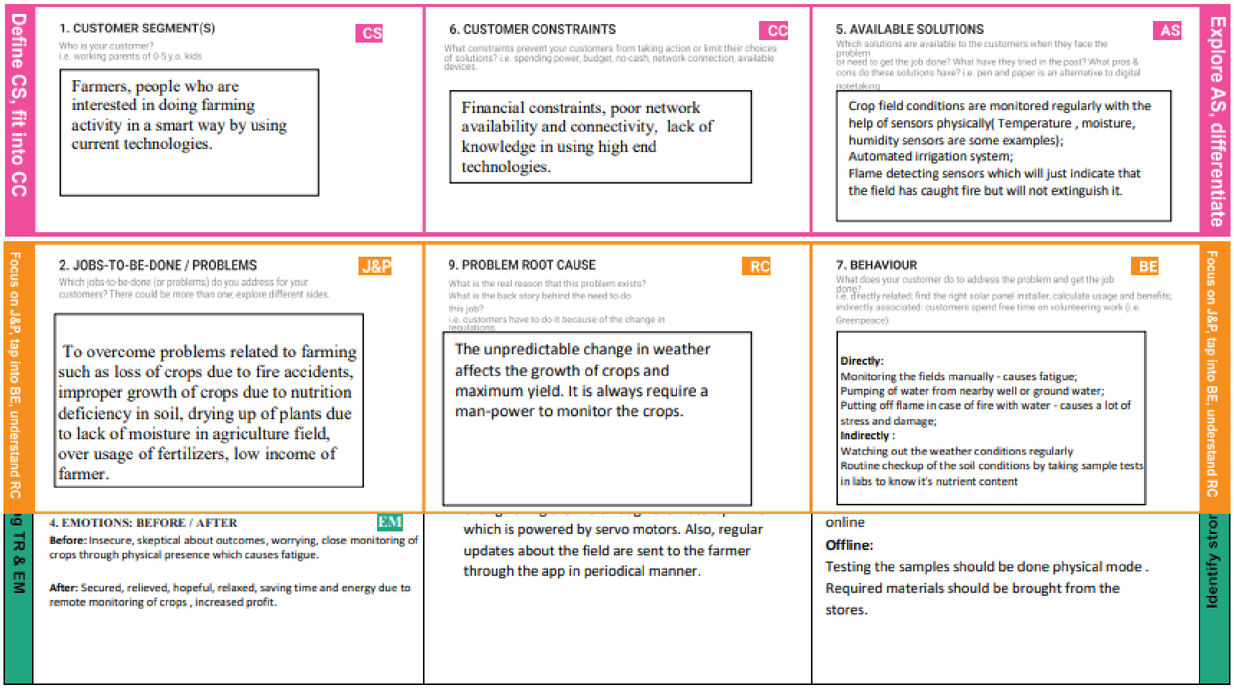

Using Robot for harvesting of crops
Monitoring of Climatic Conditions
Remote monitoring of field instead of physical presence of a human
Prevention Of Fire Accidents
Nutrient detection in soil
Using drones for fertilizer spraying
Humidity Detection
Plant disease detection using camera
Smart Pest Control
Trickle irrigation for small scale farming
Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
Drip Irrigation
Solar Panel for Irrigation
Protection of crops from cattles
Elephant detection system
pH level monitoring of soil
Prevention of Crop Theft
Monitoring Micro and Macro Nutrients
Monitoring Soil Fertility
Dew Detection in leaves
Drought monitoring
Recycling Agricultural Waste
Monitoring Phosphorous Cycle
Nitrogen level monitoring in soil
Temperature sensing
Feasibility
Regardless of their importance, which tasks are more

3.3 Proposed Solution



S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Farmers are trying to overcome problems related to agriculture such as loss of crops due to fire accidents, improper growth of crops due to nutrition deficiency in soil and drying up of plants due to lack of moisture in agriculture field.
2.	Idea / Solution description	To design a self- reliant system that provides solution for above mentioned problems with the use of IoT technologies such as flame sensor, soil NPK sensor and soil moisture sensor.
3.	Novelty / Uniqueness	Integration flame sensor, NPK sensor and soil moisture sensor into a single unit, detection of number of damaged crops due to fire accident.
4.	Social Impact / Customer Satisfaction	Remote monitoring of farm land, reduction in loss of crops, reduction in financial risk, increasing in crop yield, efficient utilization of available resources.
5.	Business Model (Revenue Model)	Maintenance cost of sensors collected from farmers occasionally, installation cost, cost for app usage.
6.	Scalability of the Solution	Increase in accuracy of sensors and reduction in delay of data transfers while accommodating large number of users.

3.4 Problem Solution fit



Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Who is your customer?
i.e. working parents of 0-5 y.o. kids

Farmers, people who are interested in doing farming activity in a smart way by using current technologies.

6. CUSTOMER CONSTRAINTS CC

What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.

Financial constraints, poor network availability and connectivity, lack of knowledge in using high end technologies.

5. AVAILABLE SOLUTIONS AS

Which solutions are available to the customers when they face the problem
or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking

Crop field conditions are monitored regularly with the help of sensors physically(Temperature , moisture, humidity sensors are some examples);
Automated irrigation system;
Flame detecting sensors which will just indicate that the field has caught fire but will not extinguish it.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.

To overcome problems related to farming such as loss of crops due to fire accidents, improper growth of crops due to nutrition deficiency in soil, drying up of plants due to lack of moisture in agriculture field, over usage of fertilizers, low income of farmer.

9. PROBLEM ROOT CAUSE RC

What is the real reason that this problem exists?
What is the back story behind the need to do this job?
i.e. customers have to do it because of the change in regulations.

The unpredictable change in weather affects the growth of crops and maximum yield. It is always require a man-power to monitor the crops.

7. BEHAVIOUR BE

What does your customer do to address the problem and get the job done?
i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)

Directly:
Monitoring the fields manually - causes fatigue;
Pumping of water from nearby well or ground water;
Putting off flame in case of fire with water - causes a lot of stress and damage;
Indirectly :
Watching out the weather conditions regularly
Routine checkup of the soil conditions by taking sample tests in labs to know it's nutrient content

Focus on J&P, tap into BE, understand RC

g TR & EM

4. EMOTIONS: BEFORE / AFTER EM

Before: Insecure, skeptical about outcomes, worrying, close monitoring of crops through physical presence which causes fatigue.

After: Secured, relieved, hopeful, relaxed, saving time and energy due to remote monitoring of crops , increased profit.

which is powered by servo motors. Also, regular updates about the field are sent to the farmer through the app in periodical manner.

online

Offline:

Testing the samples should be done physical mode . Required materials should be brought from the stores.

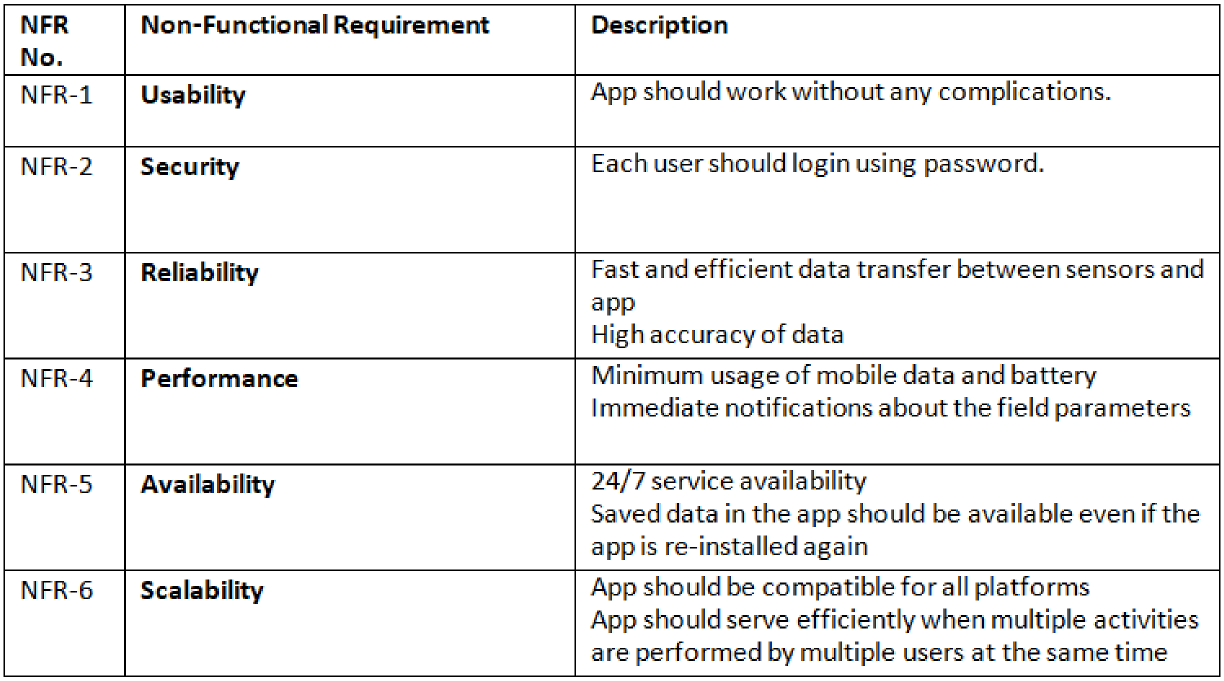
Identify stro

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration for app	Registration through App
FR-2	Real time monitoring of various parameters present in the field	Monitoring of soil moisture content Monitoring of fire Monitoring of temperature Monitoring of Soil nutrients

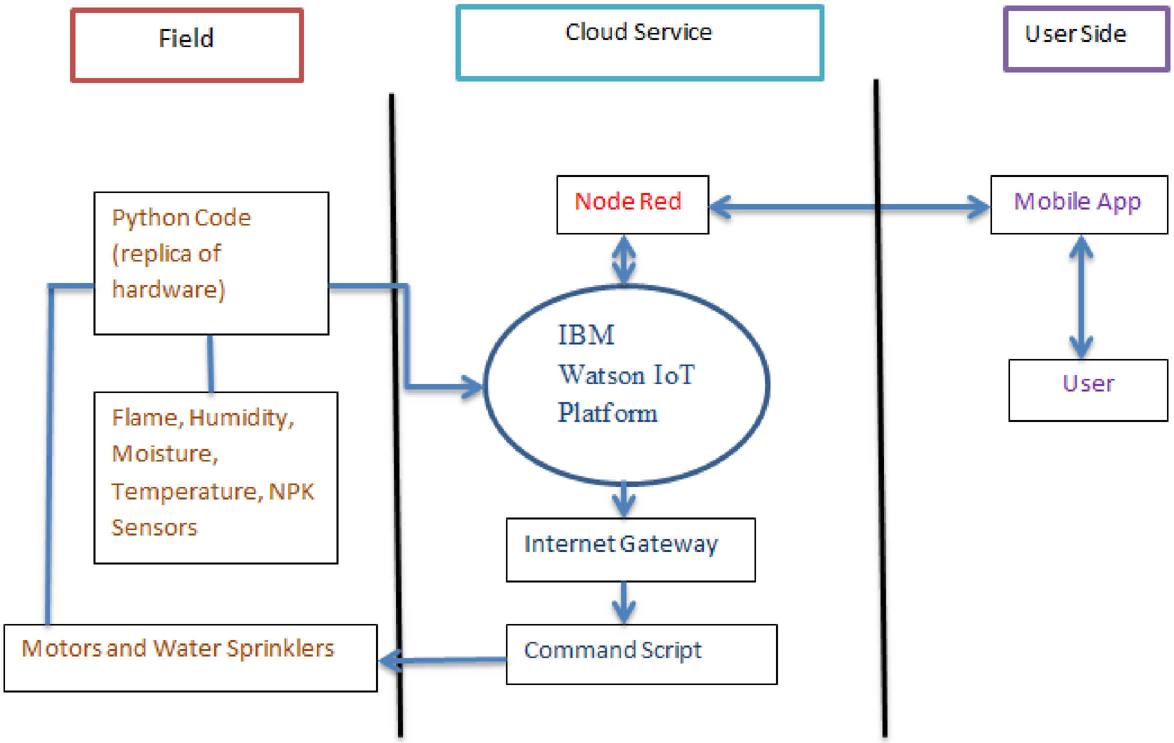
4.2 Non-Functional requirements



NFR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	App should work without any complications.
NFR-2	**Security**	Each user should login using password.
NFR-3	**Reliability**	Fast and efficient data transfer between sensors and app High accuracy of data
NFR-4	**Performance**	Minimum usage of mobile data and battery Immediate notifications about the field parameters
NFR-5	**Availability**	24/7 service availability Saved data in the app should be available even if the app is re-installed again
NFR-6	**Scalability**	App should be compatible for all platforms App should serve efficiently when multiple activities are performed by multiple users at the same time

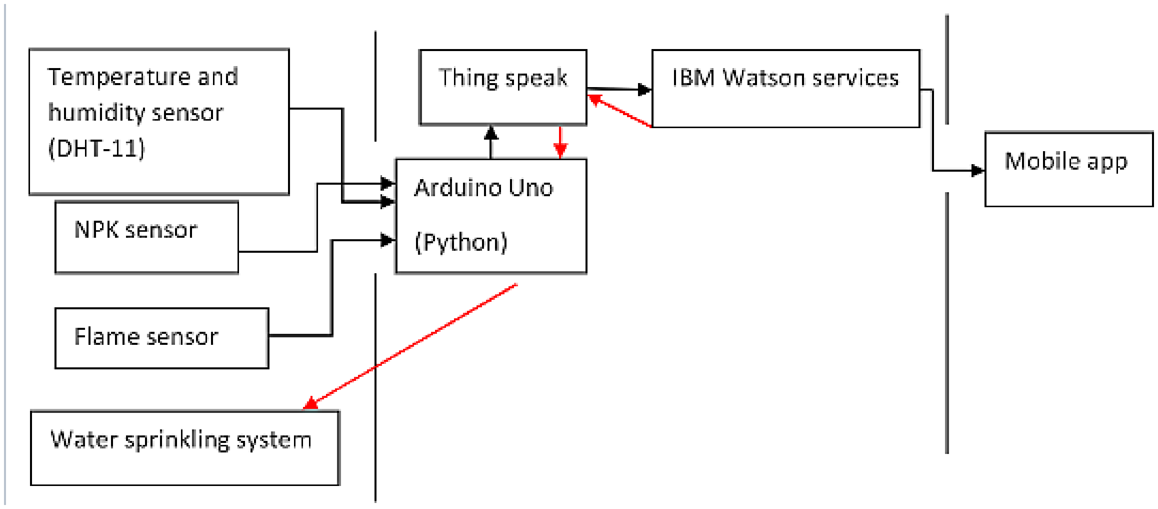
5. PROJECT DESIGN

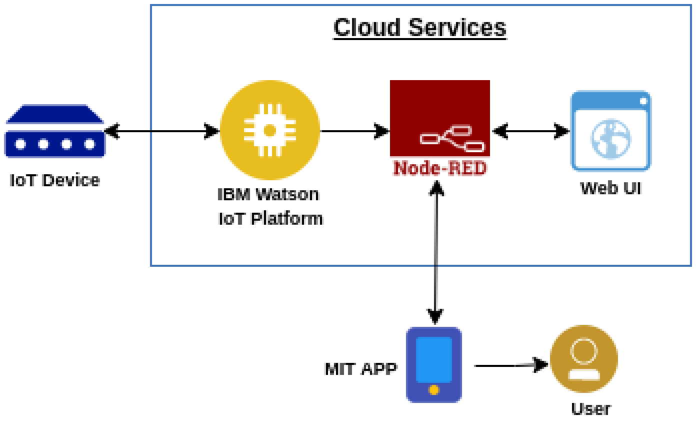
5.1 Data Flow Diagrams


Field
Cloud Service
User Side
Python Code (replica of hardware)
Flame, Humidity, Moisture, Temperature, NPK Sensors
Motors and Water Sprinklers
Node Red
IBM Watson IoT Platform
Internet Gateway
Command Script
Mobile App
User

- Flame, Nutrients, Moisture and Humidity Sensors are placed in the field and these values are generated using Python software.
- The Python software is connected to the IBM Cloud Platform through the Node Red Web GUI.
- The mobile application is developed using MIT app inventor.
- The field conditions can be checked regularly from the Mobile app which is connected through the same Node Red.
- The soil moisture sensor monitors various climatic conditions. The moisture output readings at different weather conditions is taken and updated.
- The flame detector works by detecting the UV radiation at the point of ignition. The sensor would become aware of it and produce a series ofthe pulses that are converted by detector electronics into an alarm output.
- NPK sensor is based on reflectance spectroscopy, which detects the level of energy absorbed or reflected by soil particles.
- All these information are collected regularly.
- Any changes or accidents are notified through the mobile app.
- Immediate action is also taken through the IBM cloud Platform.

5.2 Solution & Technical Architecture





5.3 User Stories

User Type	Functional Requirement(Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobileuser)	Registration	USN-1	As a user, I can register for the application byentering username and password.	I can access my account / dashboard	High	Sprint-1
Customer (Mobile user)	Login	USN-2	As a user, I can log into the application byentering username & Password.	I can login into the app upon giving correct credentials	High	Sprint-1
Customer (Mobileuser)	Establishing field parameter values	USN-1	As a user, I can want to ensure that basic field parameter values are updated correctly	The values are being published in Python IDLE	High	Sprint-2

User Type	**Functional Requirement(Epic)**	**User Story Number**	**User Story / Task**	**Acceptance criteria**	**Priority**	**Release**
Customer (Web UI user)	Dashboard	USN-1	As a user, I can enter into dashboard to view the field parameter values through Node-Red.	I can access the Dashboard to view the displaying of data	Medium	Sprint-3
Customer (Web UI user)	Establishing all the required field parameter values	USN -2	As a user, I can want to ensure that all the required field parameter values are updated correctly	The values are being published in Python IDLE	High	Sprint-3
Customer (Web UI user)	Graphical representation of data	USN -3	As a user, I want to view the field parameters in graphical form	The data can be viewed through chart and graph in Node- Red dashboard	Low	Sprint-3

User Type	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Acceptance criteria**	**Priority**	**Release**
Customer (Web UI user)	Controlling of motor	USN -4	As a user, I want to operate the motor present in the field remotely	The motor can be switched On/Off through the button present in dashboard	High	Sprint-3
Customer (Web UI user/Mobile user)	Alert messages	USN -5	As a user, I want to get notified when the field catches fire and when the soil moisture is below the threshed	The user will get alert message through the SMS service available in their phone when moisture level is below threshold & when field's fame level is above some threshold	High	Sprint-3
Customer (Mobile user)	Display all values in mobile app	USN -1	As a user, I want the field parameters to be displayed in mobile application	The parameter values are updated periodically in the app	High	Sprint-4

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Controlling of motor through app	USN -2	As a user, I want to operate the motor present in the field remotely	The motor can be switched on/off through the button present in the app	High	Sprint-4

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering username and password.	5	High	Srivatsan, Udhaya Prakash

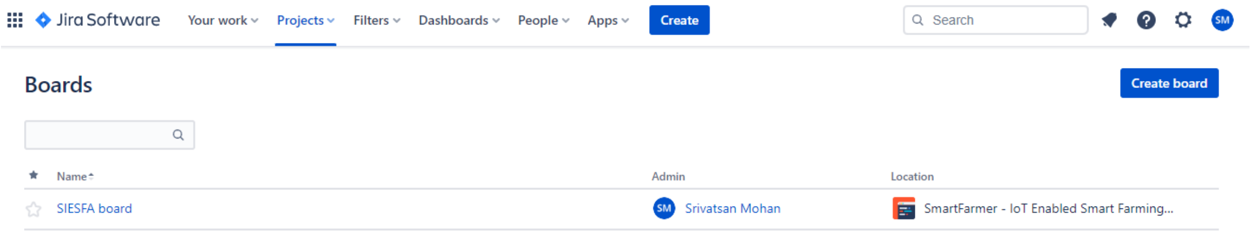
Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint-1	Login	USN-2	As a user, I can log into the application by entering username & Password.	5	High	Niranjana, Jaajana
Sprint-2	Establishing field parameter values	USN-1	As a user, I can want to ensure that basic field parameter values are updated correctly	3	High	Udhaya Prakash, Niranjana
Sprint-3	Dashboard	USN-1	As a user, I can enter into dashboard to view the field parameter values through Node-Red.	3	Medium	Niranjana, Srivatsan
Sprint-3	Establishing all the required field parameter values	USN -2	As a user, I can want to ensure that all the required field parameter values are updated correctly	3	High	Srivatsan, Jaajana
Sprint-3	Graphical representation of data	USN -3	As a user, I want to view the field parameters in graphical form	2	Low	Niranjana, jaajana

Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint-3	Controlling of motor	USN -4	As a user, I want to operate the motor present in the field remotely	5	High	Srivatsan, Niranjana
Sprint-3	Alert messages	USN -5	As a user, I want to get notified when the field catches fire and when the soil moisture is below the threshed	5	High	Udhaya prakash, Jaajana
Sprint-4	Display all values in mobile app	USN -1	As a user, I want the field parameters to be displayed in mobile application	5	High	Niranjana, Udhaya Prakash
Sprint-4	Controlling of motor through app	USN -2	As a user, I want to operate the motor present in the field remotely	5	High	Jaajana, Srivatsan

6.2 Sprint Delivery Schedule

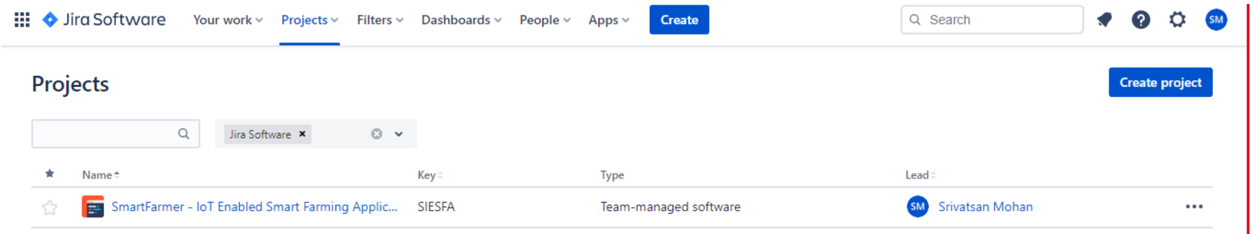
Sprint	Total Story Points	Duration	Sprint StartDate	Sprint End Date (Planned)	Story Points Completed (ason Planned End Date)	Sprint Release Date(Actual)
Sprint-1	10	6 Days	24 Oct 2022	29 Oct 2022	10	29 Oct 2022
Sprint-2	5	6 Days	31 Oct 2022	05 Nov 2022	8	05 Nov 2022
Sprint-3	18	6 Days	07 Nov 2022	12 Nov 2022	8	12 Nov 2022
Sprint-4	10	6 Days	14 Nov 2022	19 Nov 2022	6	19 Nov 2022

6.3 Reports from JIRA



Boards

★	Name	Admin	Location
☆	SIESFA board	SM Srivatsan Mohan	SmartFarmer - IoT Enabled Smart Farming...

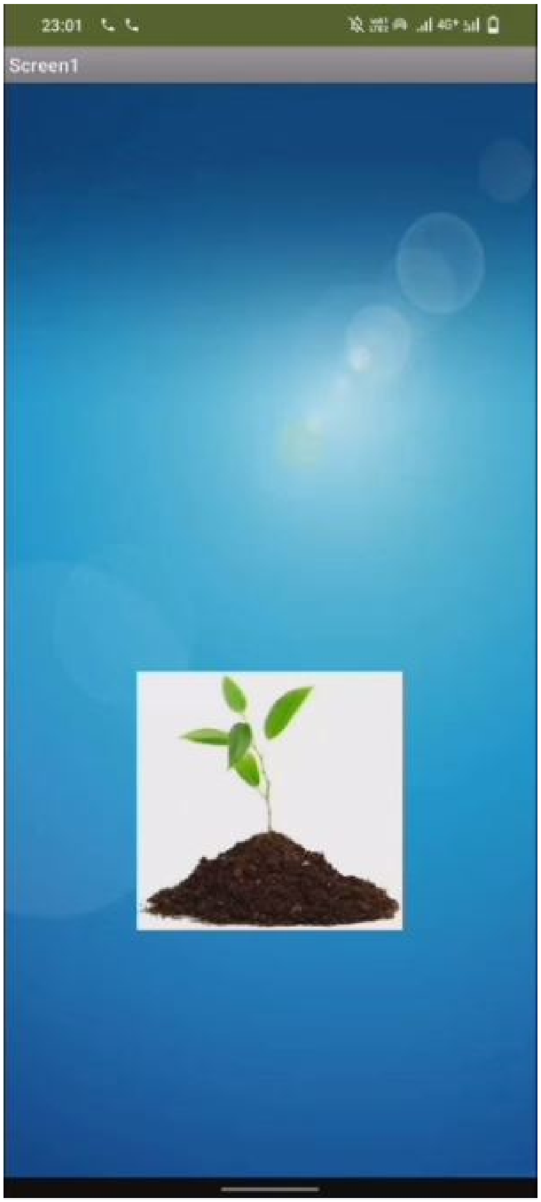


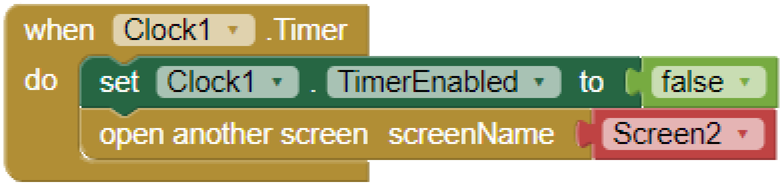
Projects

★	Name	Key	Type	Lead
☆	SmartFarmer - IoT Enabled Smart Farming Applic...	SIESFA	Team-managed software	SM Srivatsan Mohan

7. CODING & SOLUTIONING

7.1 Feature 1

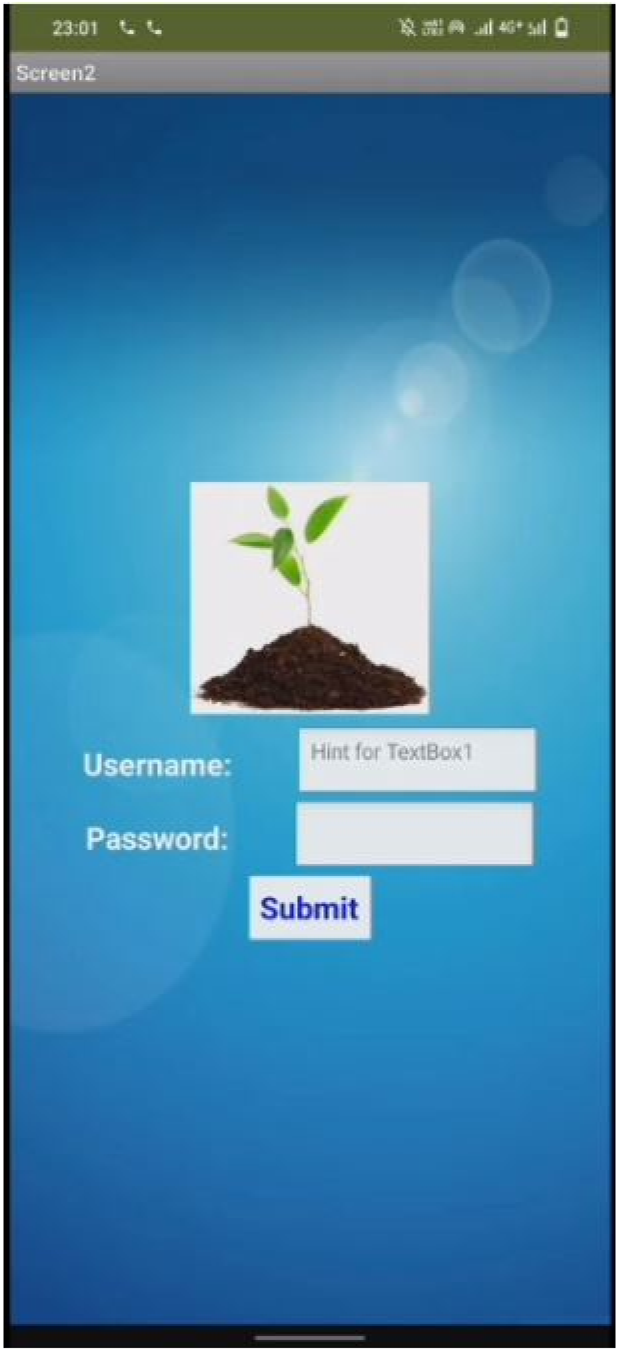


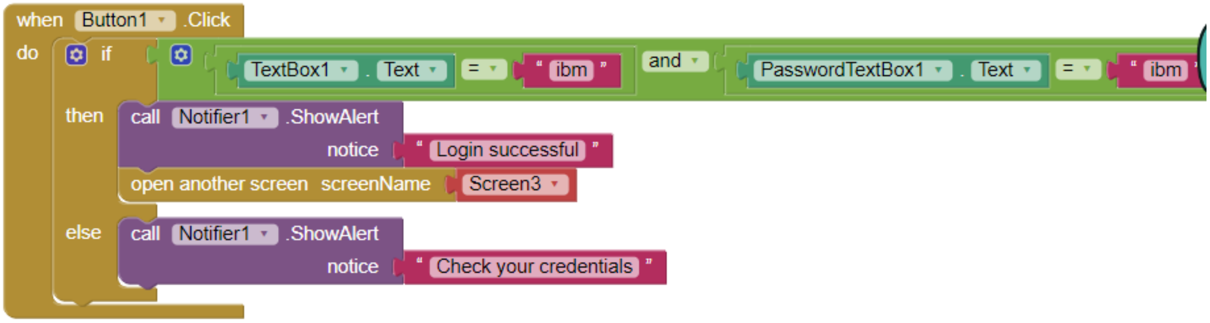


```
when Clock1 .Timer
do  set Clock1 . TimerEnabled to false
    open another screen  screenName Screen2
```

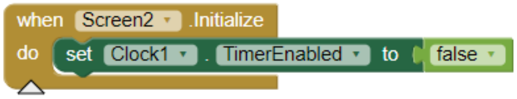
After scanning the QR code in MIT app Inventor,we can able to use the application in android.Here the picture of screen1, which is logo display.After 3 seconds, the screen 2 automatically appears.

7.2 Feature 2





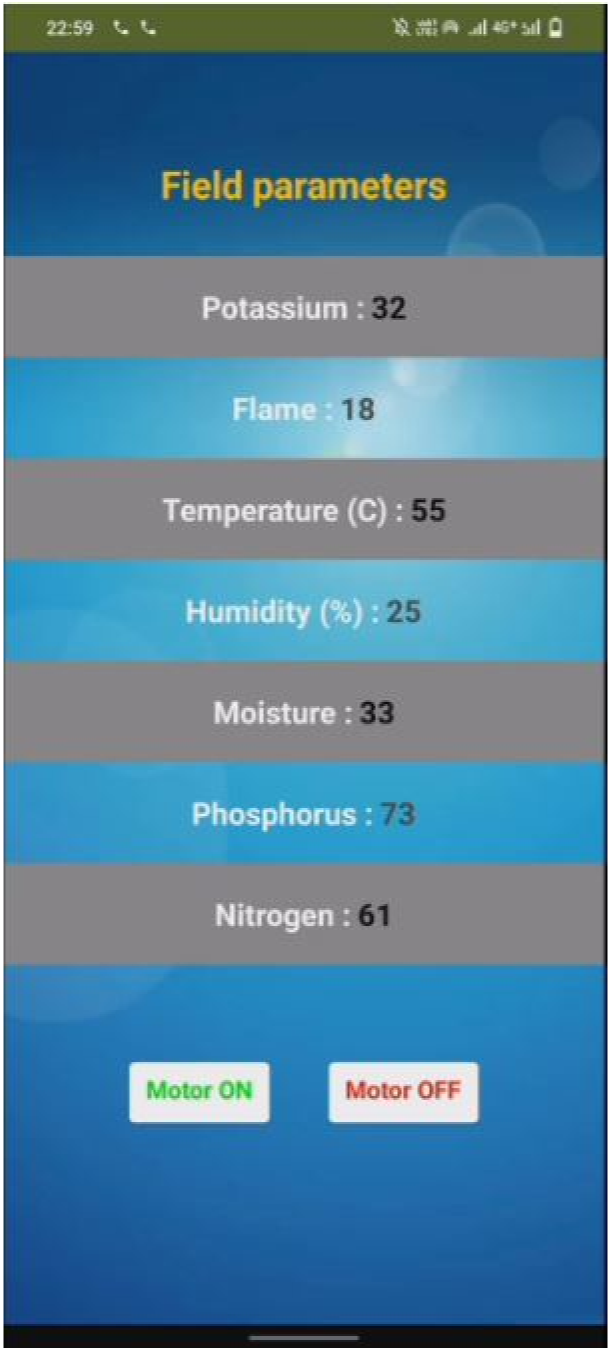
```
when Button1 .Click
do  if    TextBox1 . Text = " ibm "  and  PasswordTextBox1 . Text = " ibm
    then  call Notifier1 .ShowAlert
                              notice " Login successful "
          open another screen  screenName  Screen3
    else  call Notifier1 .ShowAlert
                              notice " Check your credentials "
```



```
when Screen2 .Initialize
do  set Clock1 . TimerEnabled to false
```

After 3 seconds from the first screen, we will be directed to the login page where we can login user credentials.

7.3 Feature 3



Field parameters

Potassium : 32

Flame : 18

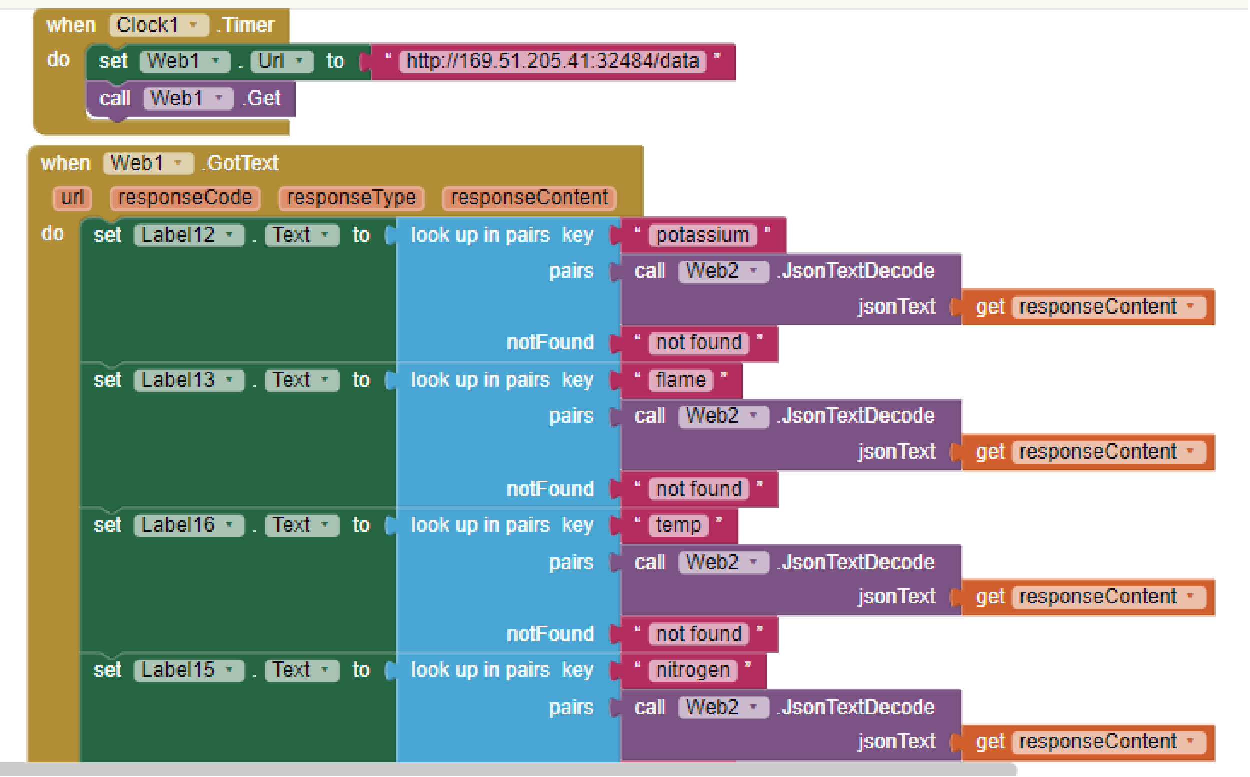
Temperature (C) : 55

Humidity (%) : 25

Moisture : 33

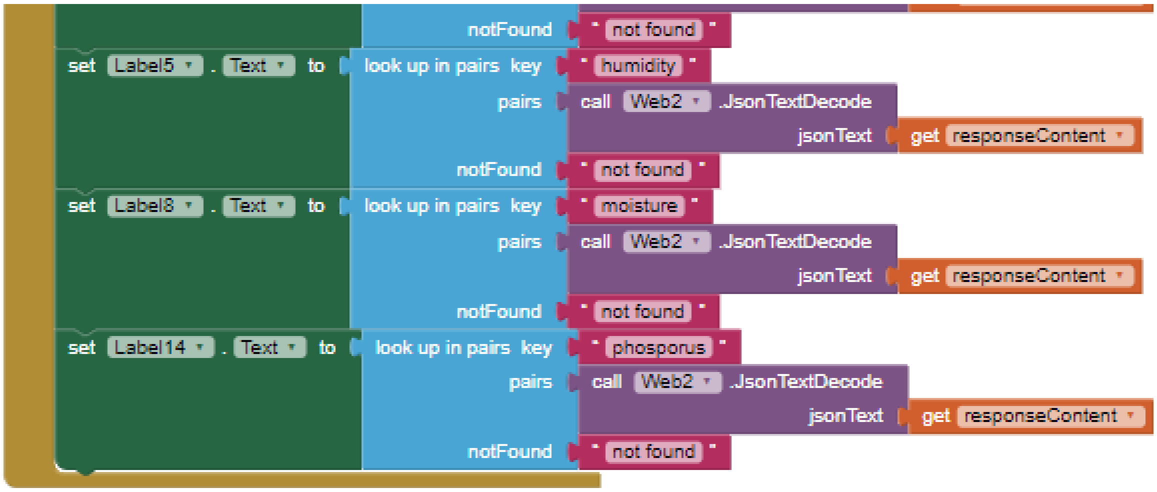
Phosphorus : 73

Nitrogen : 61

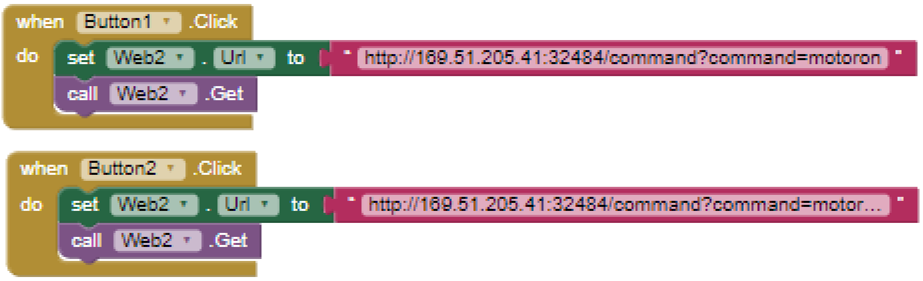


```
when Clock1 .Timer
do  set Web1 . Url to " http://169.51.205.41:32484/data "
    call Web1 .Get

when Web1 .GotText
 url  responseCode  responseType  responseContent
do  set Label12 . Text to  look up in pairs  key " potassium "
                                             pairs  call Web2 .JsonTextDecode
                                                              jsonText  get responseContent
                                          notFound " not found "
    set Label13 . Text to  look up in pairs  key " flame "
                                             pairs  call Web2 .JsonTextDecode
                                                              jsonText  get responseContent
                                          notFound " not found "
    set Label16 . Text to  look up in pairs  key " temp "
                                             pairs  call Web2 .JsonTextDecode
                                                              jsonText  get responseContent
                                          notFound " not found "
    set Label15 . Text to  look up in pairs  key " nitrogen "
                                             pairs  call Web2 .JsonTextDecode
                                                              jsonText  get responseContent
```



```
                                          notFound " not found "
    set Label5 . Text to  look up in pairs  key " humidity "
                                            pairs  call Web2 .JsonTextDecode
                                                             jsonText  get responseContent
                                         notFound " not found "
    set Label8 . Text to  look up in pairs  key " moisture "
                                            pairs  call Web2 .JsonTextDecode
                                                             jsonText  get responseContent
                                         notFound " not found "
    set Label14 . Text to  look up in pairs  key " phosporus "
                                             pairs  call Web2 .JsonTextDecode
                                                              jsonText  get responseContent
                                          notFound " not found "
```

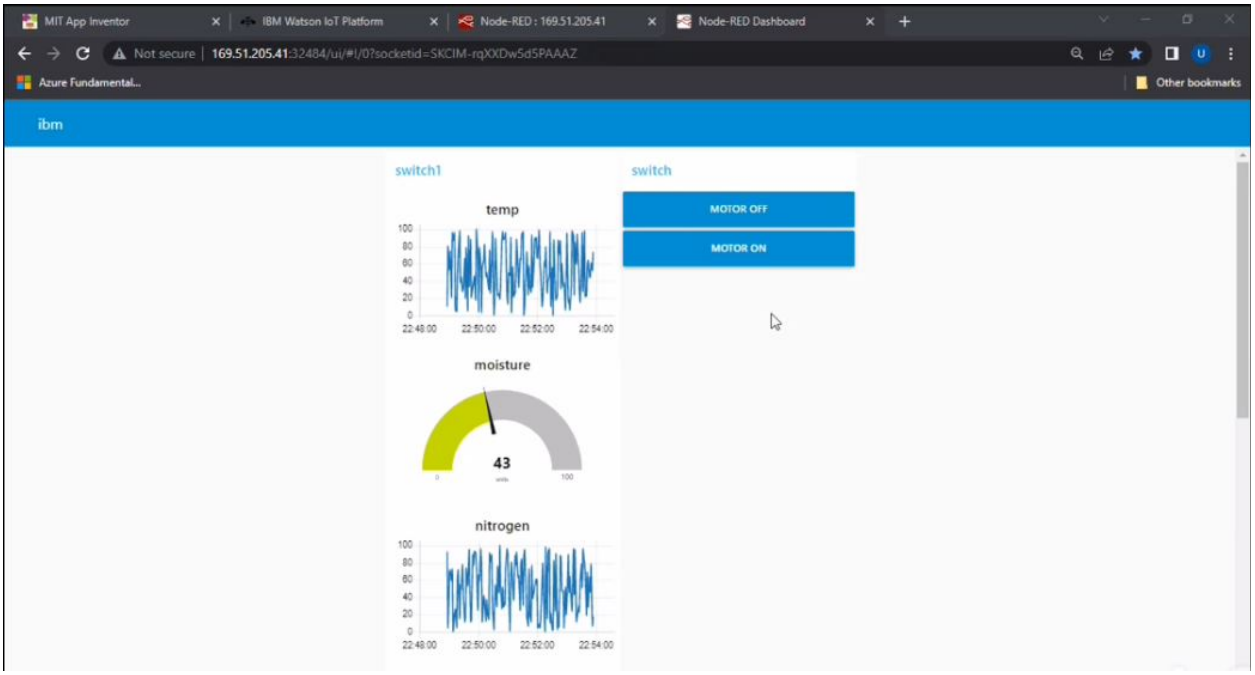


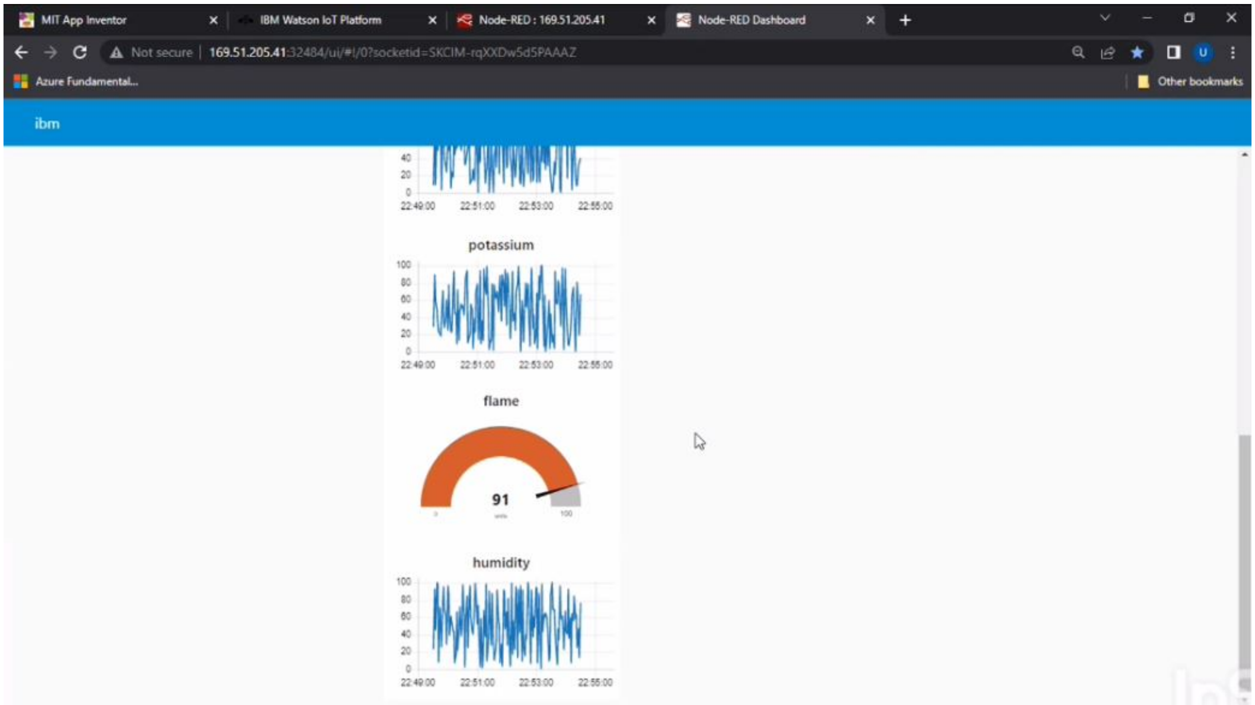
```
when Button1 .Click
do  set Web2 . Url to " http://169.51.205.41:32484/command?command=motoron "
    call Web2 .Get

when Button2 .Click
do  set Web2 . Url to " http://169.51.205.41:32484/command?command=motor... "
    call Web2 .Get
```

The field parameters is displayed in the screen3 after the user's successfull login and MOTOR ON and OFF buttons are also displayed.

7.4 Feature 4





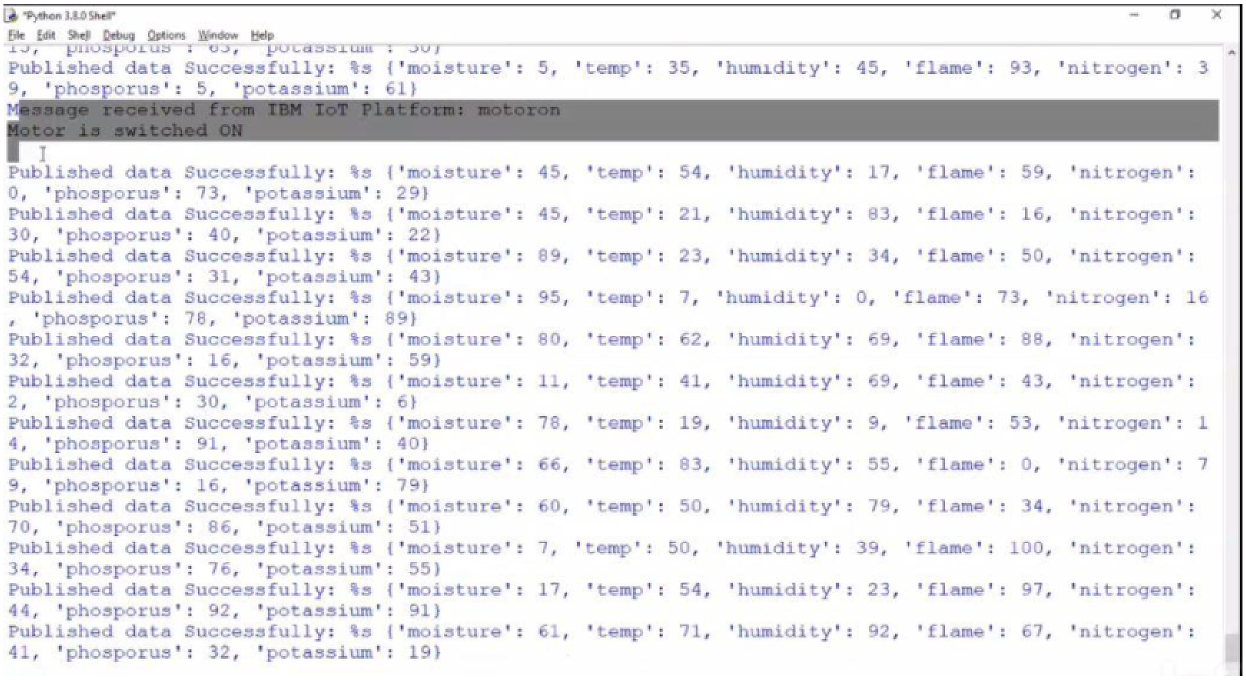
These is the Web UI which we have developed by the node-red.

8. TESTING

8.1 Test Cases

When the project is executed, we get the following results:

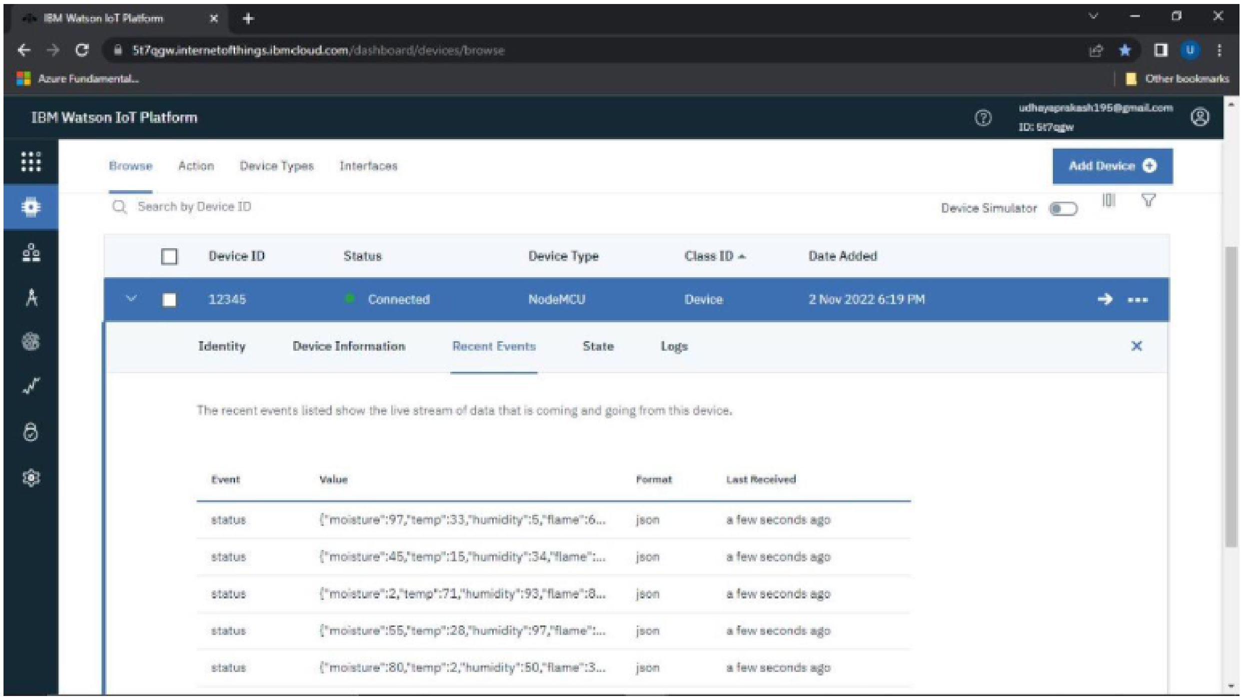
Python IDLE:



```
15, 'phosporus': 63, 'potassium': 30}
Published data Successfully: %s {'moisture': 5, 'temp': 35, 'humidity': 45, 'flame': 93, 'nitrogen': 3
9, 'phosporus': 5, 'potassium': 61}
Message received from IBM IoT Platform: motoron
Motor is switched ON

Published data Successfully: %s {'moisture': 45, 'temp': 54, 'humidity': 17, 'flame': 59, 'nitrogen':
0, 'phosporus': 73, 'potassium': 29}
Published data Successfully: %s {'moisture': 45, 'temp': 21, 'humidity': 83, 'flame': 16, 'nitrogen':
30, 'phosporus': 40, 'potassium': 22}
Published data Successfully: %s {'moisture': 89, 'temp': 23, 'humidity': 34, 'flame': 50, 'nitrogen':
54, 'phosporus': 31, 'potassium': 43}
Published data Successfully: %s {'moisture': 95, 'temp': 7, 'humidity': 0, 'flame': 73, 'nitrogen': 16
, 'phosporus': 78, 'potassium': 89}
Published data Successfully: %s {'moisture': 80, 'temp': 62, 'humidity': 69, 'flame': 88, 'nitrogen':
32, 'phosporus': 16, 'potassium': 59}
Published data Successfully: %s {'moisture': 11, 'temp': 41, 'humidity': 69, 'flame': 43, 'nitrogen':
2, 'phosporus': 30, 'potassium': 6}
Published data Successfully: %s {'moisture': 78, 'temp': 19, 'humidity': 9, 'flame': 53, 'nitrogen': 1
4, 'phosporus': 91, 'potassium': 40}
Published data Successfully: %s {'moisture': 66, 'temp': 83, 'humidity': 55, 'flame': 0, 'nitrogen': 7
9, 'phosporus': 16, 'potassium': 79}
Published data Successfully: %s {'moisture': 60, 'temp': 50, 'humidity': 79, 'flame': 34, 'nitrogen':
70, 'phosporus': 86, 'potassium': 51}
Published data Successfully: %s {'moisture': 7, 'temp': 50, 'humidity': 39, 'flame': 100, 'nitrogen':
34, 'phosporus': 76, 'potassium': 55}
Published data Successfully: %s {'moisture': 17, 'temp': 54, 'humidity': 23, 'flame': 97, 'nitrogen':
44, 'phosporus': 92, 'potassium': 91}
Published data Successfully: %s {'moisture': 61, 'temp': 71, 'humidity': 92, 'flame': 67, 'nitrogen':
41, 'phosporus': 32, 'potassium': 19}
```

IBM cloud:

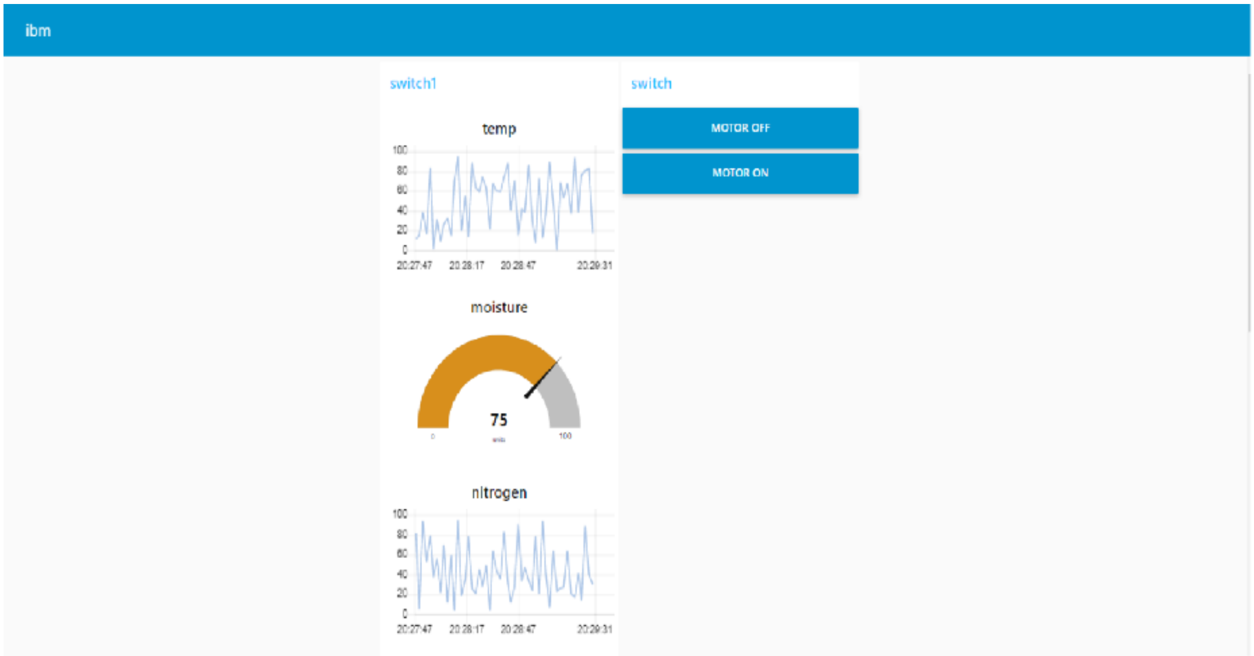


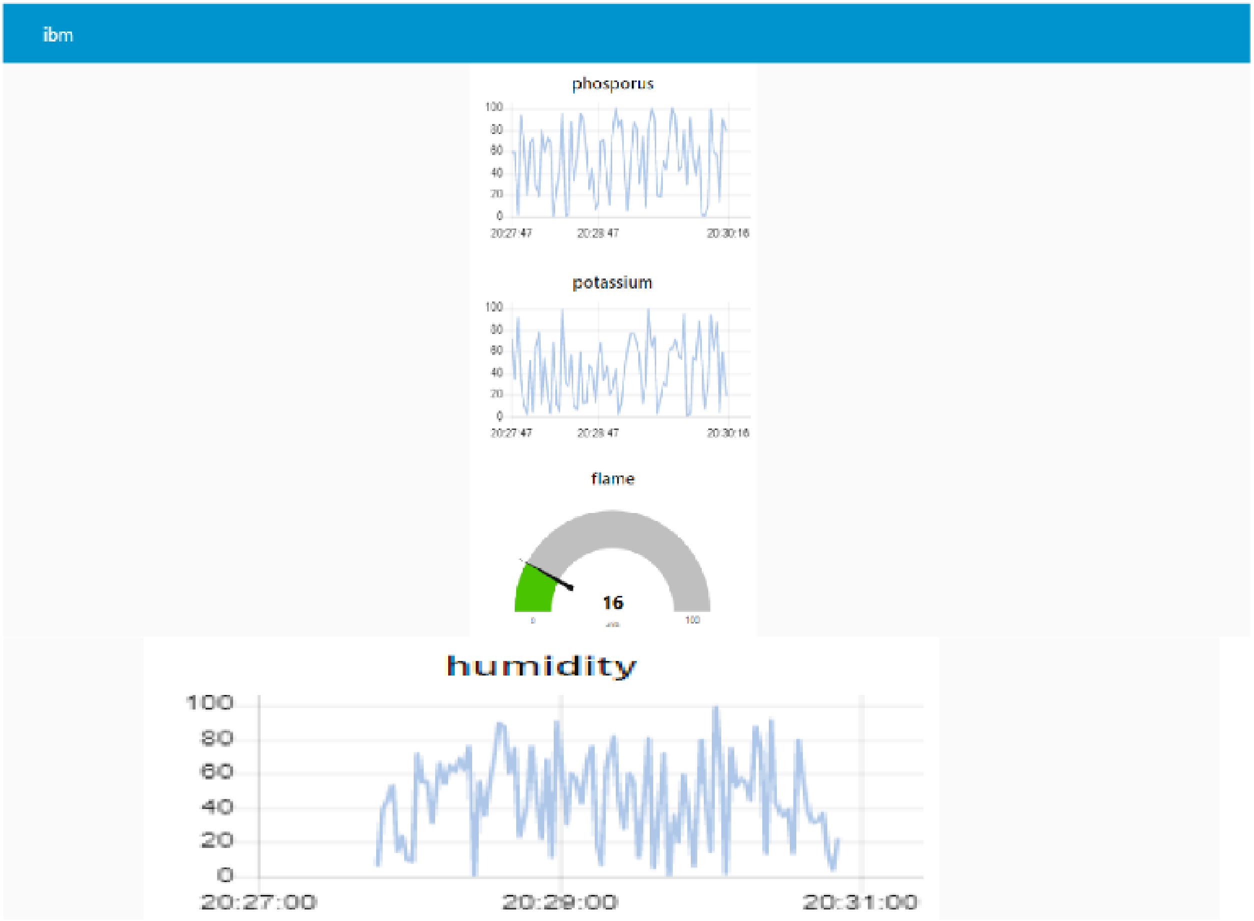
	Device ID	Status	Device Type	Class ID ▲	Date Added
☐	12345	● Connected	NodeMCU	Device	2 Nov 2022 6:19 PM

The recent events listed show the live stream of data that is coming and going from this device.

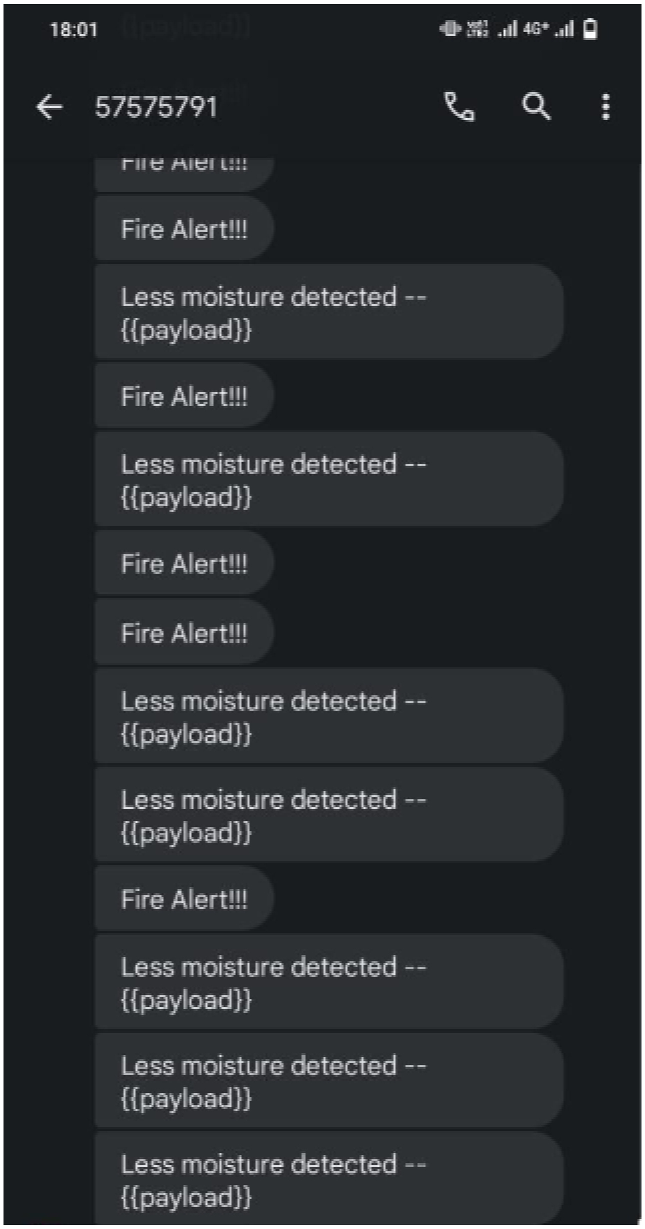
Event	Value	Format	Last Received
status	{"moisture":97,"temp":33,"humidity":5,"flame":6...	json	a few seconds ago
status	{"moisture":45,"temp":15,"humidity":34,"flame":...	json	a few seconds ago
status	{"moisture":2,"temp":71,"humidity":93,"flame":8...	json	a few seconds ago
status	{"moisture":55,"temp":28,"humidity":97,"flame":...	json	a few seconds ago
status	{"moisture":80,"temp":2,"humidity":50,"flame":3...	json	a few seconds ago

Web UI (Node- Red dashboard):




ibm
phosporus
100
80
60
40
20
0
20:27:47
20:28:47
20:30:16
potassium
100
80
60
40
20
0
20:27:47
20:28:47
20:30:16
flame
16
0
100
humidity
100
80
60
40
20
0
20:27:00
20:29:00
20:31:00

SMS alert when fire sensor & moisture sensor value is below threshold:



18:01

57575791

Fire Alert!!!

Fire Alert!!!

Less moisture detected -- {{payload}}

Fire Alert!!!

Less moisture detected -- {{payload}}

Fire Alert!!!

Fire Alert!!!

Less moisture detected -- {{payload}}

Less moisture detected -- {{payload}}

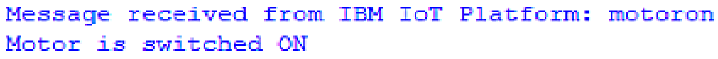
Fire Alert!!!

Less moisture detected -- {{payload}}

Less moisture detected -- {{payload}}

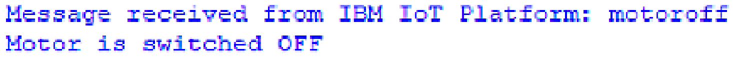
Less moisture detected -- {{payload}}

When "Motor ON" button is pressed in web UI, the output in Python is observed below:



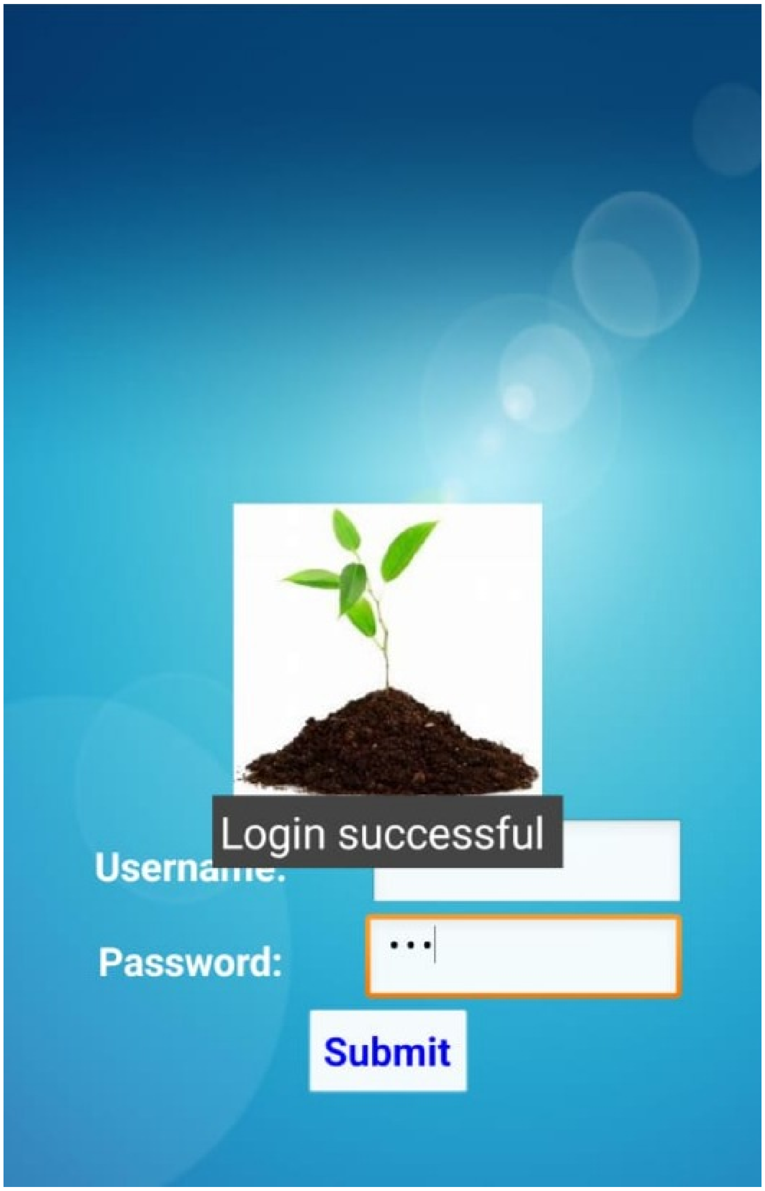
```
Message received from IBM IoT Platform: motoron
Motor is switched ON
```

When "Motor OFF" button is pressed in web UI, the output in Python is observed below:



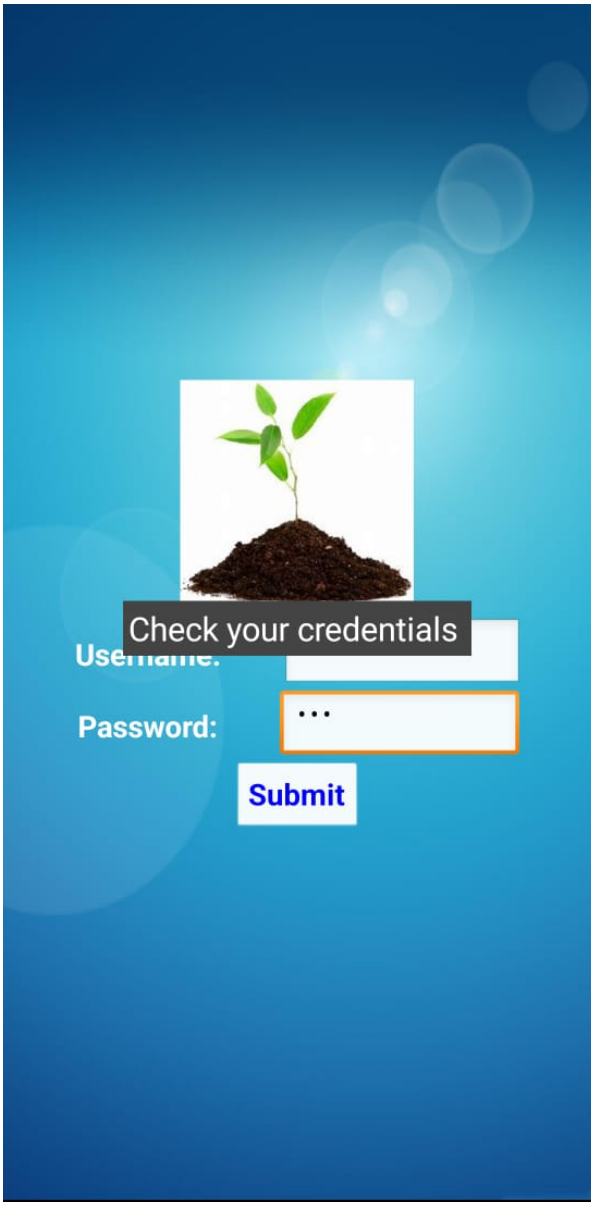
```
Message received from IBM IoT Platform: motoroff
Motor is switched OFF
```

When user credentials entered in the app are correct:



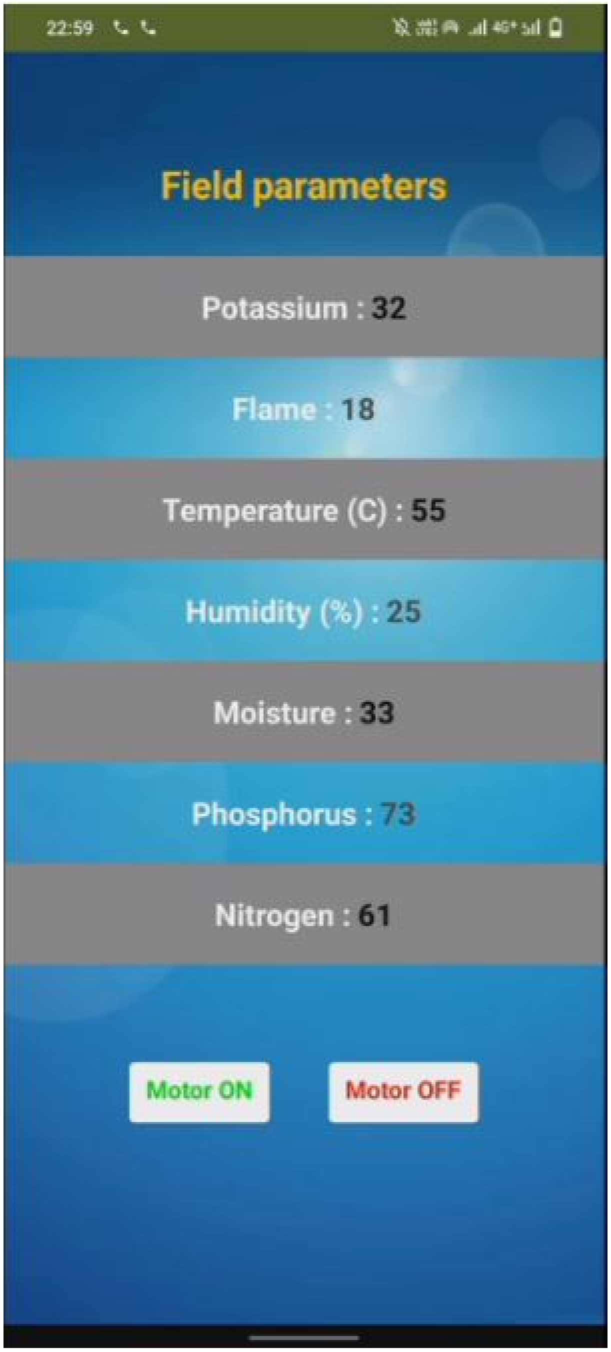
Login successful

When user credentials entered in the app are incorrect:



Check your credentials

App output:



Field parameters

Potassium : 32

Flame : 18

Temperature (C) : 55

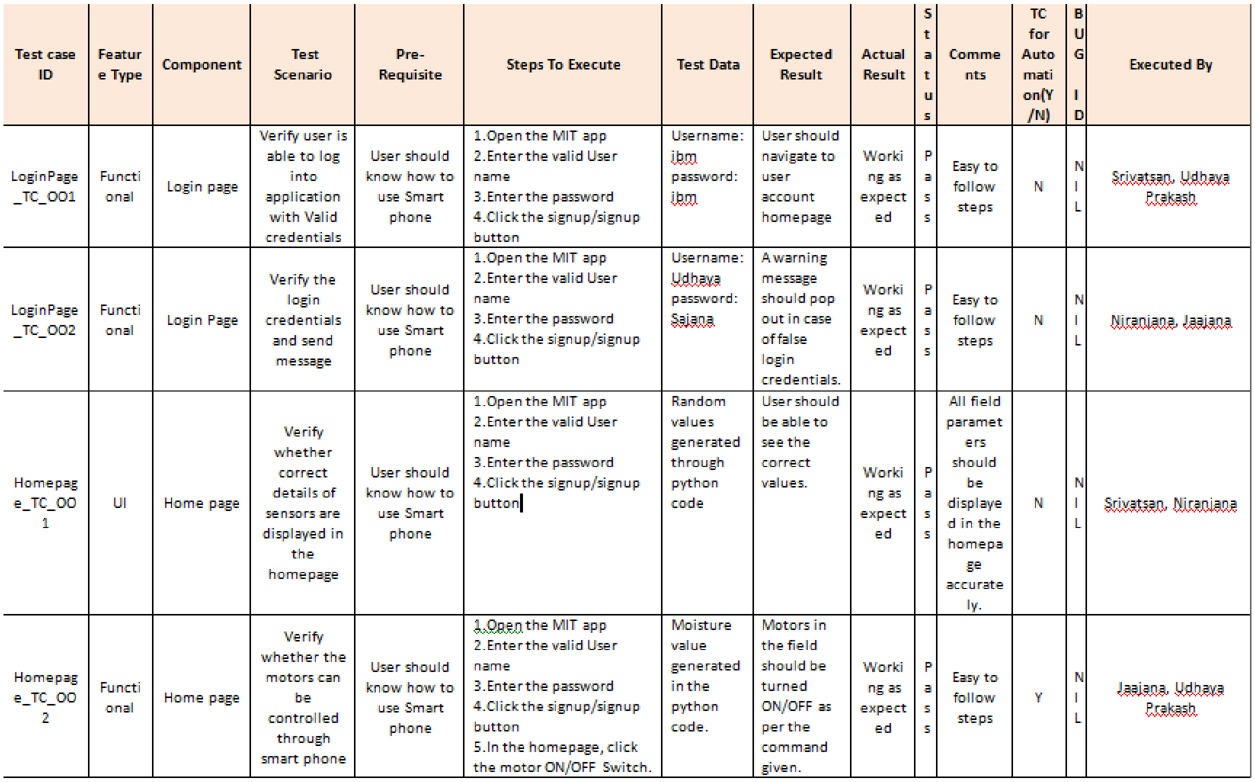
Humidity (%) : 25

Moisture : 33

Phosphorus : 73

Nitrogen : 61

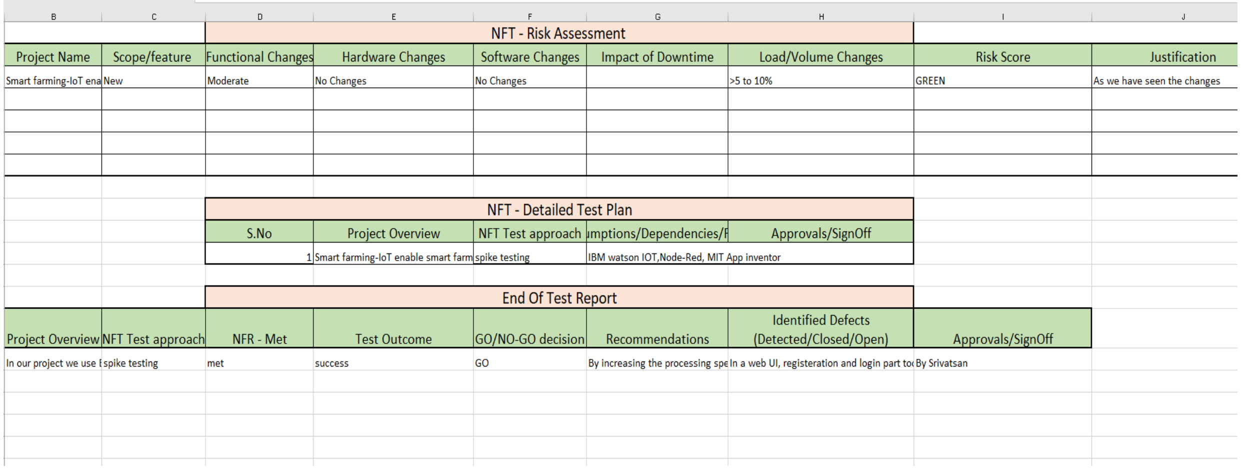
8.2 User Acceptance Testing



Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
LoginPage_TC_OO1	Functional	Login page	Verify user is able to log into application with Valid credentials	User should know how to use Smart phone	1.Open the MIT app 2.Enter the valid User name 3.Enter the password 4.Click the signup/signup button	Username: ibm password: ibm	User should navigate to user account homepage	Working as expected	Pass	Easy to follow steps	N	NIL	Srivatsan, Udhaya Prakash
LoginPage_TC_OO2	Functional	Login Page	Verify the login credentials and send message	User should know how to use Smart phone	1.Open the MIT app 2.Enter the valid User name 3.Enter the password 4.Click the signup/signup button	Username: Udhaya password: Sajana	A warning message should pop out in case of false login credentials.	Working as expected	Pass	Easy to follow steps	N	NIL	Niranjana, Jaajana
Homepage_TC_OO1	UI	Home page	Verify whether correct details of sensors are displayed in the homepage	User should know how to use Smart phone	1.Open the MIT app 2.Enter the valid User name 3.Enter the password 4.Click the signup/signup button	Random values generated through python code	User should be able to see the correct values.	Working as expected	Pass	All field parameters should be displayed in the homepage accurately.	N	NIL	Srivatsan, Niranjana
Homepage_TC_OO2	Functional	Home page	Verify whether the motors can be controlled through smart phone	User should know how to use Smart phone	1.Open the MIT app 2.Enter the valid User name 3.Enter the password 4.Click the signup/signup button 5.In the homepage, click the motor ON/OFF Switch.	Moisture value generated in the python code.	Motors in the field should be turned ON/OFF as per the command given.	Working as expected	Pass	Easy to follow steps	Y	NIL	Jaajana, Udhaya Prakash

9. RESULTS

9.1 Performance Metrics



NFT - Risk Assessment

Project Name	Scope/feature	Functional Changes	Hardware Changes	Software Changes	Impact of Downtime	Load/Volume Changes	Risk Score	Justification
Smart farming-IoT ena	New	Moderate	No Changes	No Changes		>5 to 10%	GREEN	As we have seen the changes

NFT - Detailed Test Plan

S.No	Project Overview	NFT Test approach	umptions/Dependencies/	Approvals/SignOff
1	Smart farming-IoT enable smart farm	spike testing	IBM watson IOT,Node-Red, MIT App inventor	

End Of Test Report

Project Overview	NFT Test approach	NFR - Met	Test Outcome	GO/NO-GO decision	Recommendations	Identified Defects (Detected/Closed/Open)	Approvals/SignOff
In our project we use	spike testing	met	success	GO	By increasing the processing spe	In a web UI, registeration and login part to	By Srivatsan

10. ADVANTAGES & DISADVANTAGES

Advantages:

In today’s world, technology has become a vital part in our day-to-day life. And it’s a pity if we don’t use it to help our very food providers – Our farmers. Our project makes agriculture much easier for our farmers. It enables them to control what’s going on in their field with much ease at their homes itself. The use of Moisture, Temperature, Humidity sensors make it known when to water the crops even if they are miles apart from their fields. It helps them to be prepared for other climatic conditions as well. The use of NPK sensor facilitates in keeping track of the soil fertility which is significant and unique to different crops. Thus a table of required nutrients for each crop is known from the app itself and this makes it much easier for the farmers to manage their land. And the use of flame sensors and water sprinklers makes it possible to act out on an emergency situation without the compulsion of the farmer being nearby.

Disadvantages:

One of the disadvantages is that it may take some time for data to be transmitted from the sensors to cloud, and then to the App. And since the sensors in the field are vulnerable, they make become less sensitive over a period of time or if any accident occurs. So regular maintenance of sensors is required. The farmers should be vigilant enough to turn ON the motor immediately after they receive an alert SMS when there is low moisture content or when fire accidents occur in the field.

11. CONCLUSION

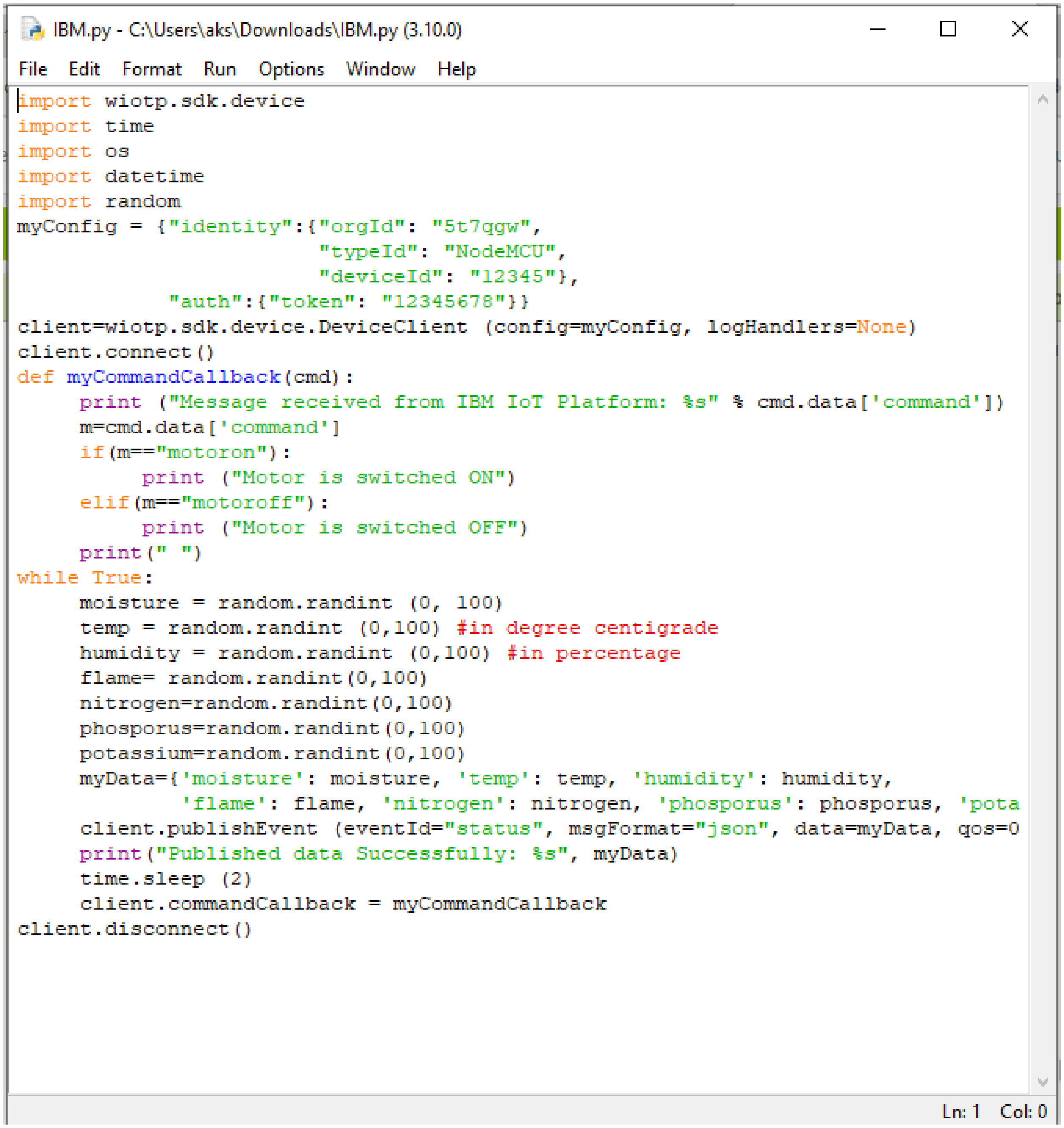
Thus, the project which we have designed mainly helps the farmer in increasing their crop yield by reducing their pain. This project allows remote monitoring of field parameters like soil moisture content, temperature, humidity and soil nutrient content. Also, an SMS alert is sent to the farmer when moisture level is below some threshold. In addition to this, SMS alert is also sent when the field catches fire i.e. when the value of flame sensor is above some threshold. So, according to the requirement, the farmer can switch ON and OFF the motor in his field. To replicate the real life application, we have used Python code simulation. It consists of Temperature sensor, Moisture sensor, Humidity sensor, NPK sensor(Nitogen, Phosphorus, Potassium sensor) and Flame sensor. All the above mentioned parameters are to be checked periodically so as to get a best yield from the farmland.

12. FUTURE SCOPE

1. Currently, only one user can access our app. In future works, we can make arrangements for multiple users to login/sign up simultaneously.
2. Many more sensors can also be included. For example, we can make room for pest control.
3. In the future, we can also move to automation. That is, when the moisture level decreases or a fire accident occurs, the water sprinklers get turned ON automatically.
4. We can also design drones to sprinkle fertilizers in accordance with NPK sensors.

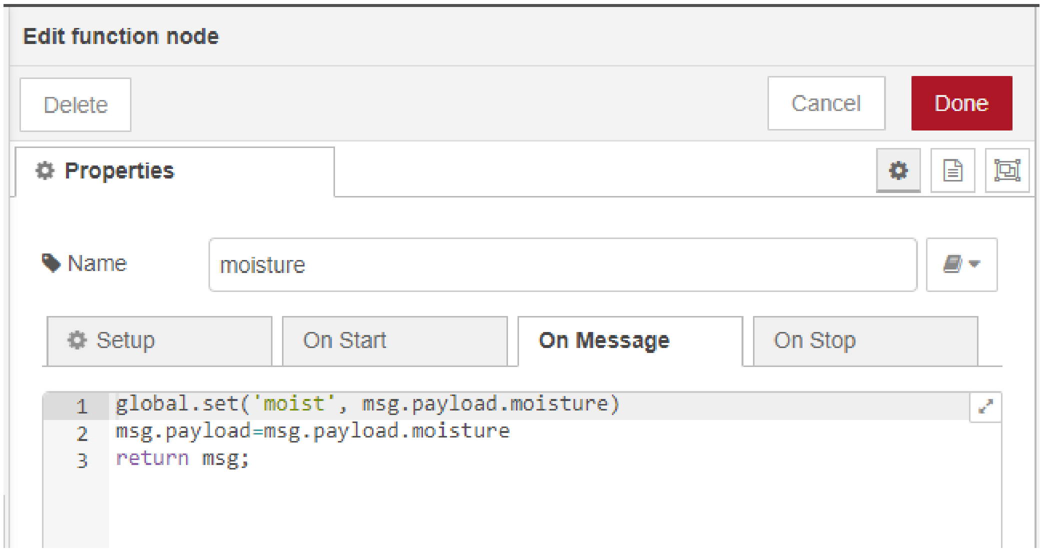
13. APPENDIX

Source Code


IBM.py - C:\Users\aks\Downloads\IBM.py (3.10.0)
File Edit Format Run Options Window Help
Ln: 1 Col: 0

```
import wiotp.sdk.device
import time
import os
import datetime
import random
myConfig = {"identity":{"orgId": "5t7qgw",
                        "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 switched ON")
    elif(m=="motoroff"):
        print ("Motor is switched OFF")
    print(" ")
while True:
    moisture = random.randint (0, 100)
    temp = random.randint (0,100) #in degree centigrade
    humidity = random.randint (0,100) #in percentage
    flame= random.randint(0,100)
    nitrogen=random.randint(0,100)
    phosporus=random.randint(0,100)
    potassium=random.randint(0,100)
    myData={'moisture': moisture, 'temp': temp, 'humidity': humidity,
            'flame': flame, 'nitrogen': nitrogen, 'phosporus': phosporus, 'pota
    client.publishEvent (eventId="status", msgFormat="json", data=myData, qos=0
    print("Published data Successfully: %s", myData)
    time.sleep (2)
    client.commandCallback = myCommandCallback
client.disconnect()
```

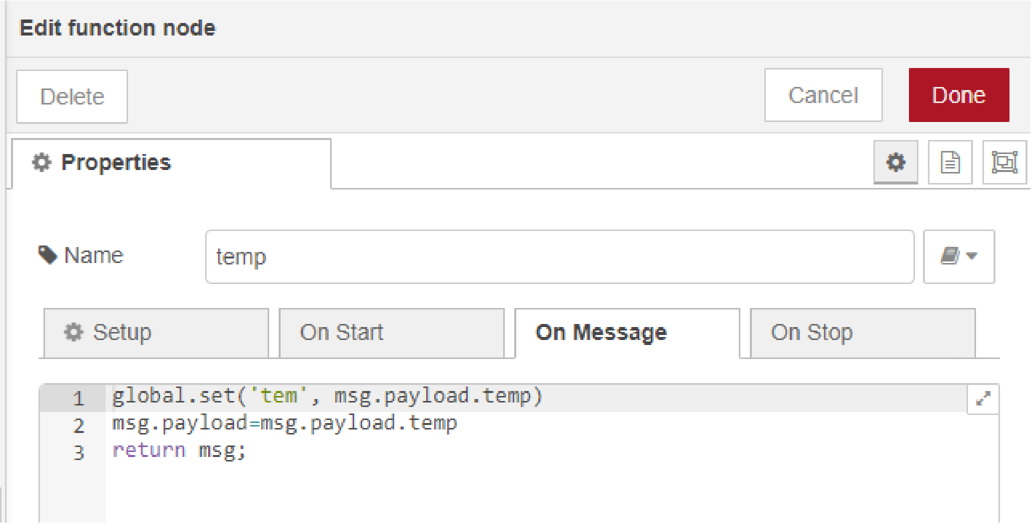
Moisture Sensor:



Name: moisture

```
global.set('moist', msg.payload.moisture)
msg.payload=msg.payload.moisture
return msg;
```

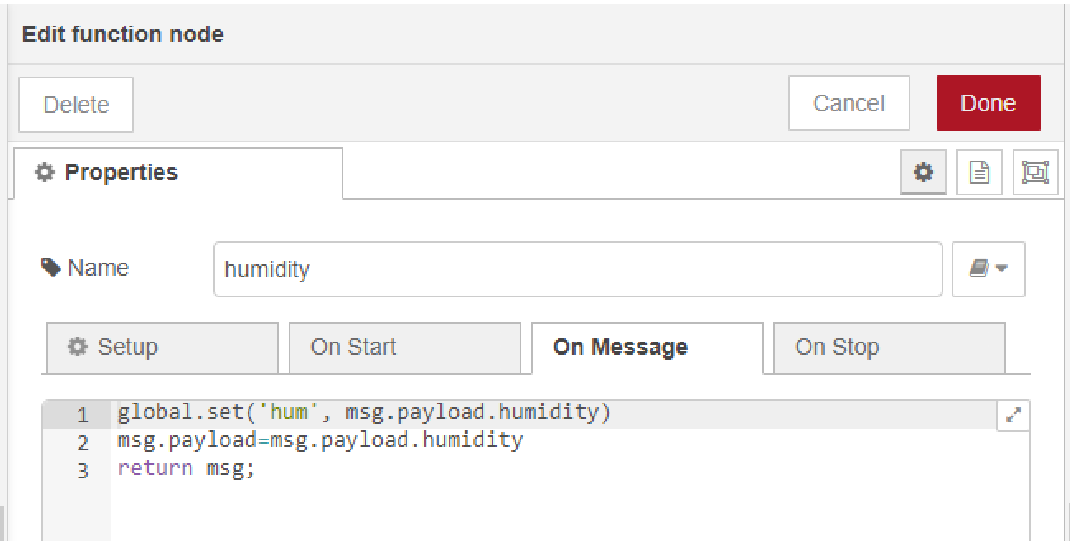
Temperature Sensor:



Name: temp

```
global.set('tem', msg.payload.temp)
msg.payload=msg.payload.temp
return msg;
```

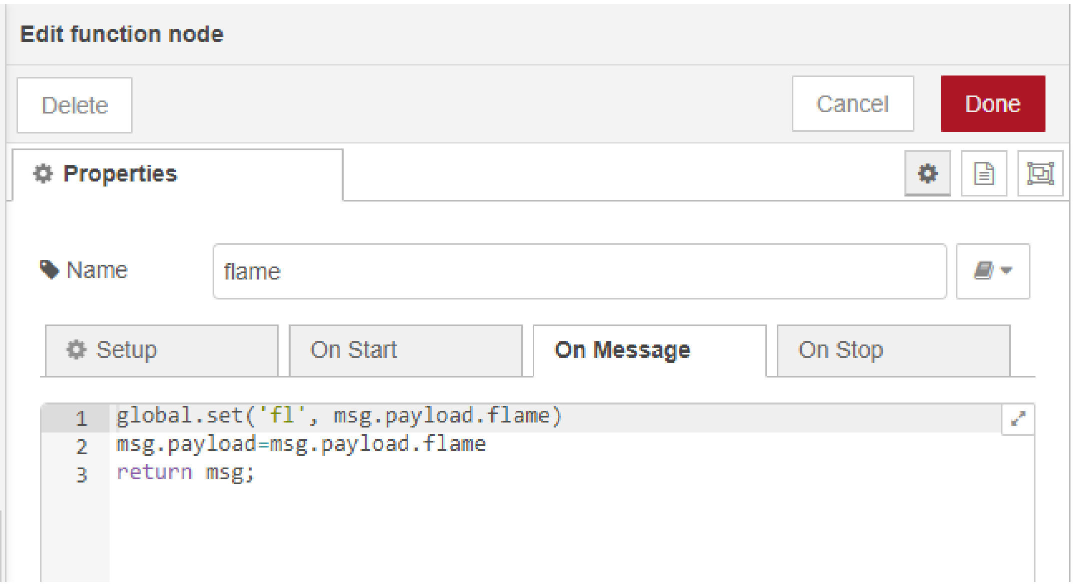
Humidity Sensor:



Name: humidity

```
global.set('hum', msg.payload.humidity)
msg.payload=msg.payload.humidity
return msg;
```

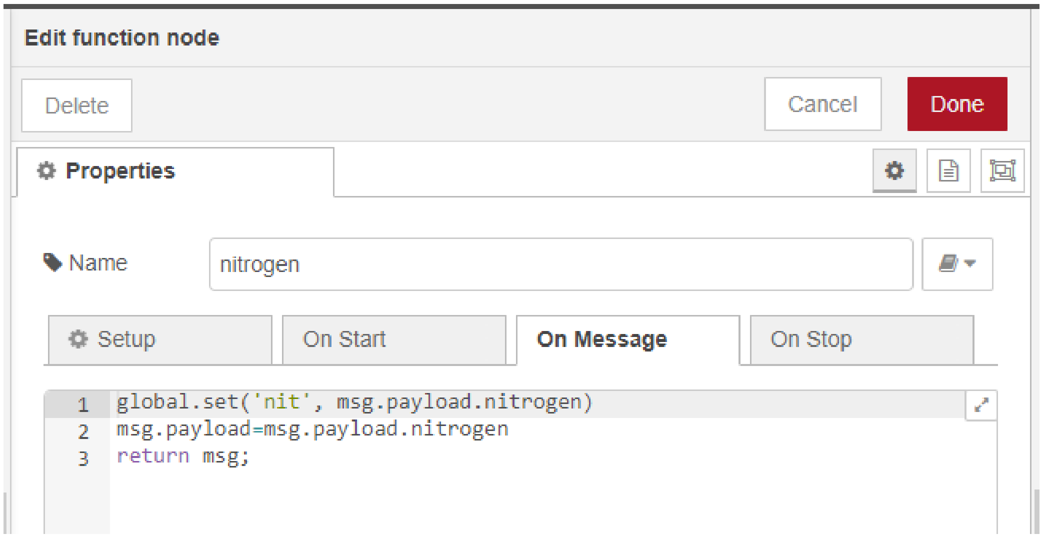
Flame Sensor:



Name: flame

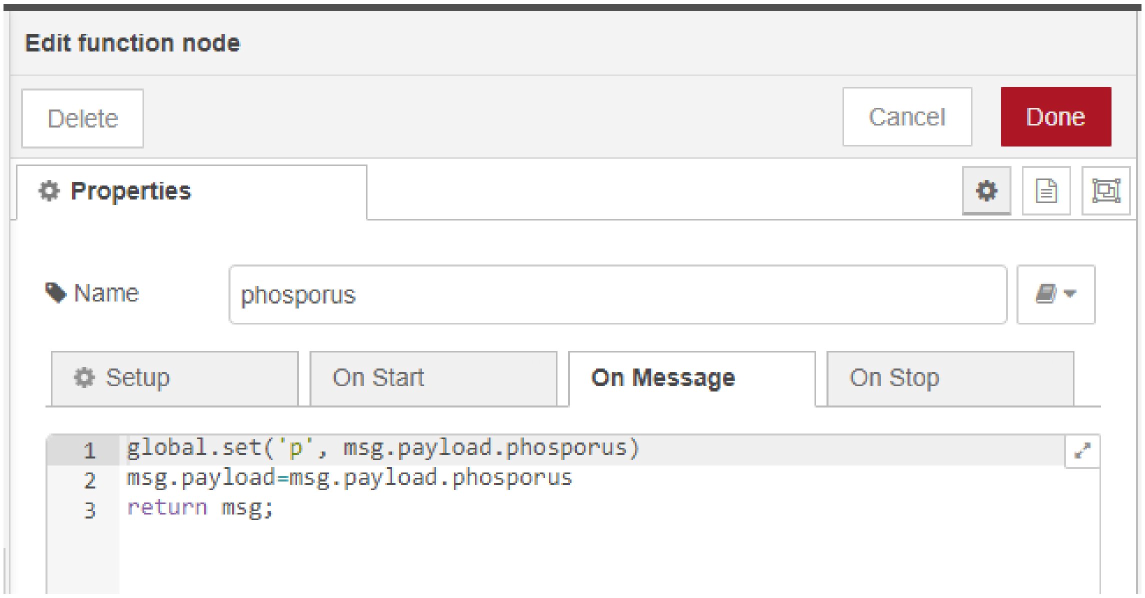
```
global.set('fl', msg.payload.flame)
msg.payload=msg.payload.flame
return msg;
```

NPK Sensor:



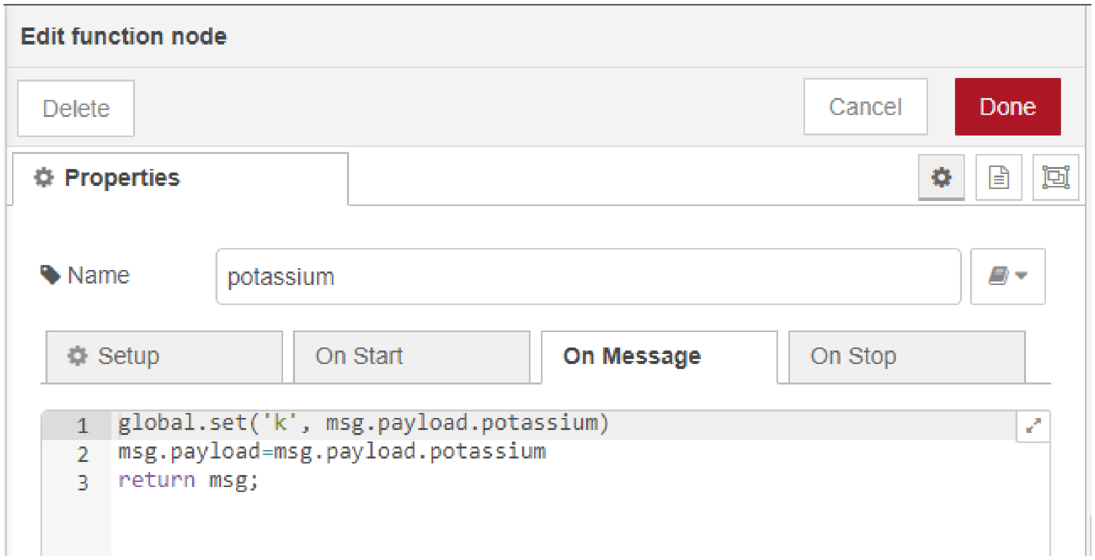
Name: nitrogen

```
global.set('nit', msg.payload.nitrogen)
msg.payload=msg.payload.nitrogen
return msg;
```



Name: phosporus

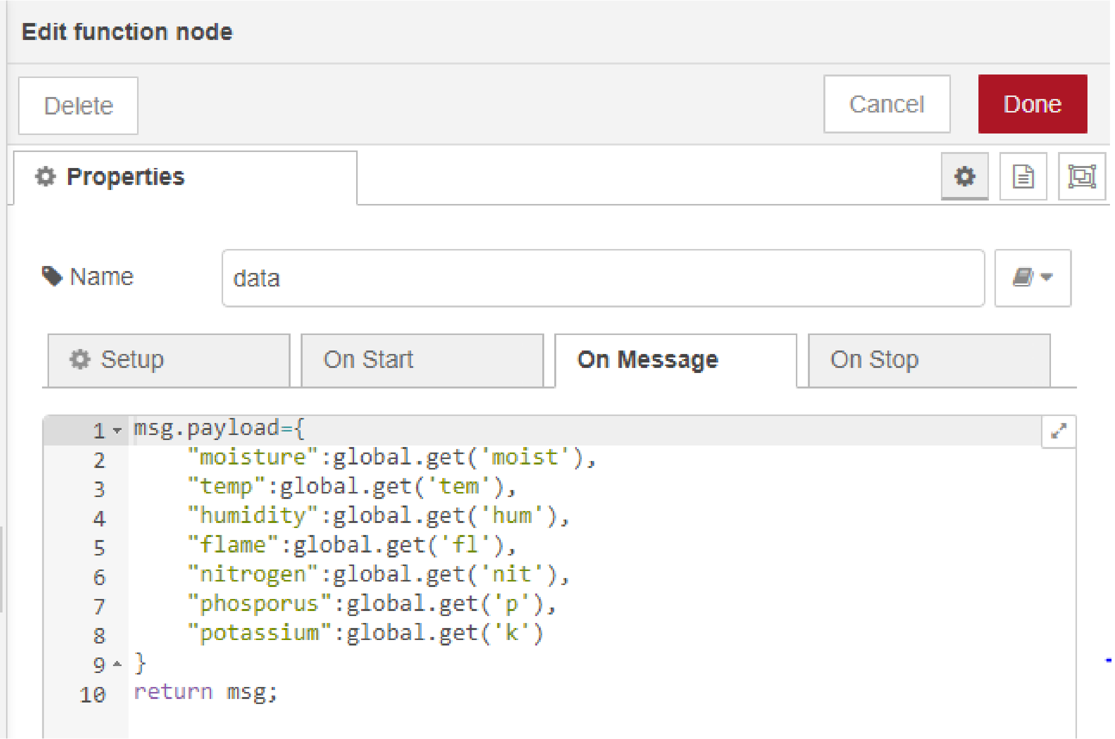
```
global.set('p', msg.payload.phosporus)
msg.payload=msg.payload.phosporus
return msg;
```



Name: potassium

```
global.set('k', msg.payload.potassium)
msg.payload=msg.payload.potassium
return msg;
```

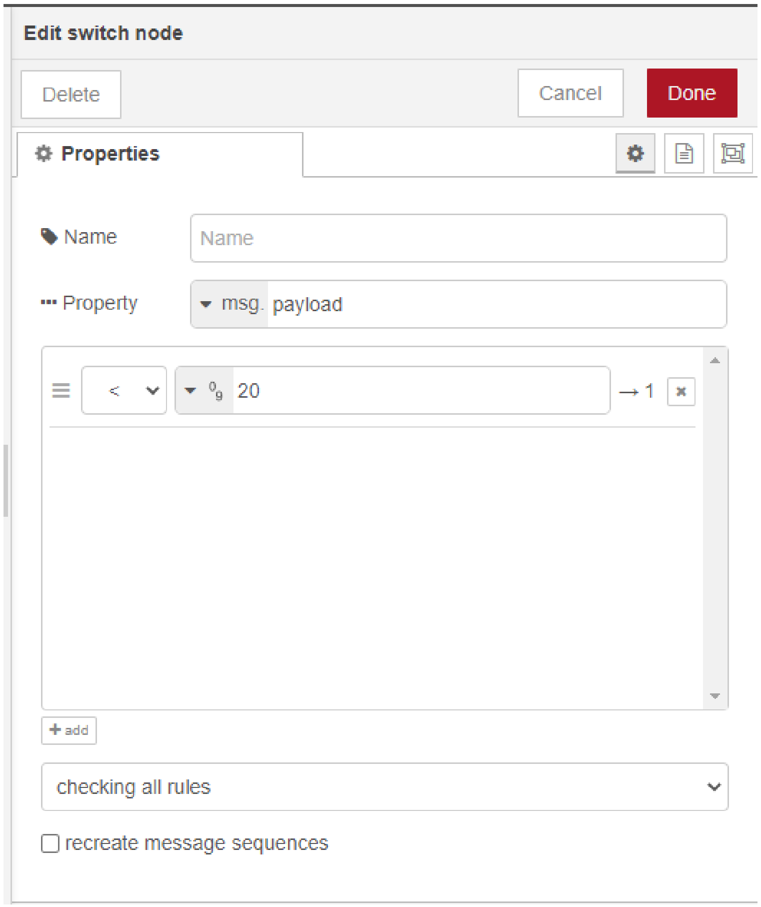
Data:



Name: data

```
msg.payload={
    "moisture":global.get('moist'),
    "temp":global.get('tem'),
    "humidity":global.get('hum'),
    "flame":global.get('fl'),
    "nitrogen":global.get('nit'),
    "phosporus":global.get('p'),
    "potassium":global.get('k')
}
return msg;
```

Motors:



Edit switch node

Name: Name

Property: msg. payload

< 20 → 1

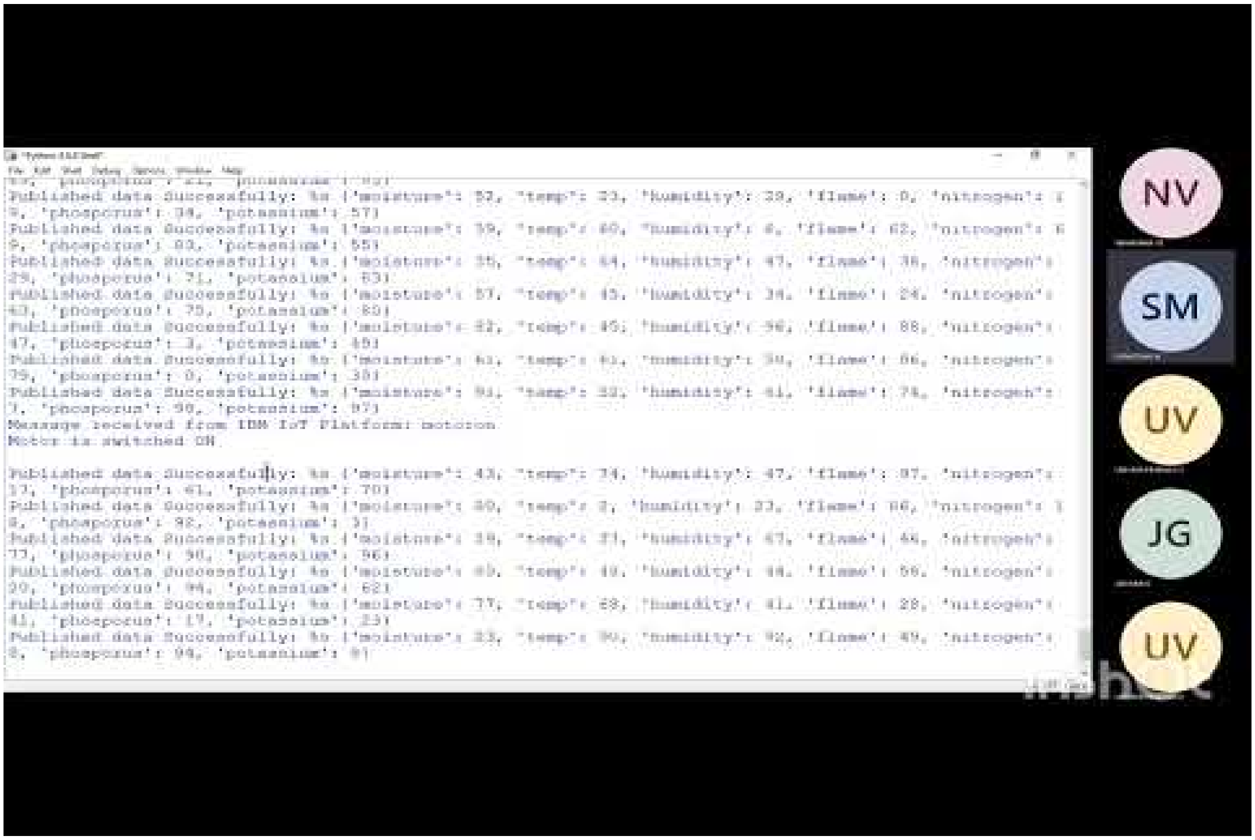
checking all rules

□ recreate message sequences

GitHub & Project Demo:

Github link: https://github.com/IBM-EPBL/IBM-Project-509-1658304420

Project demo link:



```
69, 'phosporus': 21, 'potassium': 85}
Published data Successfully: %s {'moisture': 52, 'temp': 23, 'humidity': 28, 'flame': 0, 'nitrogen': 1
9, 'phosporus': 34, 'potassium': 57}
Published data Successfully: %s {'moisture': 59, 'temp': 60, 'humidity': 6, 'flame': 62, 'nitrogen': 6
9, 'phosporus': 83, 'potassium': 55}
Published data Successfully: %s {'moisture': 35, 'temp': 64, 'humidity': 47, 'flame': 36, 'nitrogen':
29, 'phosporus': 71, 'potassium': 63}
Published data Successfully: %s {'moisture': 57, 'temp': 45, 'humidity': 38, 'flame': 24, 'nitrogen':
43, 'phosporus': 75, 'potassium': 83}
Published data Successfully: %s {'moisture': 32, 'temp': 45, 'humidity': 96, 'flame': 88, 'nitrogen':
47, 'phosporus': 3, 'potassium': 49}
Published data Successfully: %s {'moisture': 61, 'temp': 61, 'humidity': 30, 'flame': 86, 'nitrogen':
79, 'phosporus': 0, 'potassium': 38}
Published data Successfully: %s {'moisture': 91, 'temp': 32, 'humidity': 61, 'flame': 74, 'nitrogen':
7, 'phosporus': 50, 'potassium': 97}
Message received from IBM IoT Platform: motoron
Motor is switched ON

Published data Successfully: %s {'moisture': 43, 'temp': 74, 'humidity': 47, 'flame': 97, 'nitrogen':
17, 'phosporus': 61, 'potassium': 70}
Published data Successfully: %s {'moisture': 30, 'temp': 2, 'humidity': 23, 'flame': 86, 'nitrogen': 1
8, 'phosporus': 92, 'potassium': 3}
Published data Successfully: %s {'moisture': 19, 'temp': 21, 'humidity': 67, 'flame': 46, 'nitrogen':
77, 'phosporus': 50, 'potassium': 96}
Published data Successfully: %s {'moisture': 80, 'temp': 40, 'humidity': 48, 'flame': 56, 'nitrogen':
29, 'phosporus': 94, 'potassium': 62}
Published data Successfully: %s {'moisture': 77, 'temp': 28, 'humidity': 41, 'flame': 28, 'nitrogen':
41, 'phosporus': 17, 'potassium': 23}
Published data Successfully: %s {'moisture': 23, 'temp': 90, 'humidity': 92, 'flame': 49, 'nitrogen':
8, 'phosporus': 94, 'potassium': 8}
```