PROJECT REPORT

IOT BASED SMART CROP PROTECTION SYSTEM FOR AGRICULTURE

TEAM ID : : PNT2022TMID52016

TEAM LEADER: KANJANA S K

TEAM MEMBER 1: AARTHI G S

TEAM MEMBER 2: AATHITH J G

TEAM MEMBER 3:DURGAI VEERAPANDIAN S

Content

1. Introduction :

Agriculture is a huge contributor for our economy . Nowadays the biggest challenge faced by farmers in agriculture is the “ Crop Depredation “ due to wildlife interference and various other environmental factors like climate change . These factors leads to huge yield loss for farmers. With the changing of climate, agriculture faces increasing problems with extreme weather events leading to considerable yield losses of crops .Due to climate changes the farming pattern will also change.. The crops are also affected in a large scale due to animals and bird attacks. So in order to increase the yield and protect the crops , there is a urgent need to address the above mentioned issues.

1.1 Project Overview :

Agriculture nowadays faces many issues which cause huge capital and yield loss to the farmers. But in our project we mainly focus on 2 issues – Climatic change and crop attack due to animals and birds. In order to face this issues in an efficient manner , we are going to create a “Crop Protection System” which is going to be IOT based. In our project the main component is Arduino Uno. We are going to sense the moisture content of soil using humidity sensor. The climate at an particular area is going to be read from website “ openweathermap.org”. Also we are going to use PIR and ultra sonic sensor for detecting motion and measuring the distance. The information from the IOT device is going to send to cloud storage and notifications will be sent to farmer’s mobile and necessary actions will be taken.

1.2 Purpose :

The main purpose of this system is :

- To predict the climate of the area.
- To monitor the soil moisture level.
- To prevent the animals and birds from attacking the field.

2. Literature Survey :

For our project we have analysed so many research papers related to our concept. The information obtained from the survey is given below :

2.1 Existing Problem :

The existing problem is that the agriculture sector faces many problems. For example animals and birds attack climate change , improper irrigation pattern etc. In order to tackle these issues they use many methods. Farmers will predict the climate and they plan accordingly but this can be inappropriate . Because nowadays the climate conditions are unpredictable . If there is excess water in the field and if the farmers are far away from their field , it will lead to huge losses. Also they appoint guards for protecting the farms at night time. But this approach will not be that feasible as it may be an threat to human life. Also if no one is there in the farm then animals may affect the crop. Also electric fences are being used which will lead to excess electricity bills. Also the existing system use buzzers for shooing off the animals and birds. But this is no very efficient when there is a group of animals and birds.

2.2 References :

- IOT Based Crop Protection System against Birds and Wild Animal Attacks - P.Navaneetha , R.Ramiya Devi , S.Vennila , P.Manikandan , Dr.S.Saravana
- IoT based Raspberry Pi Crop Vandalism Prevention system - P.A.Harsha Vardhini, N.Koteswaramma, K.Murali Chandra Babu
- The new era of Technological Farming: An Emerging Agronomics - Neha A. Rathi , Pranav M. Patil , Aniket S. Marwade , Mohit K. Popat
- Smart Crop Protection System from Animals and Fire using Arduino - Srikanth N , Aishwarya , Kavita H M , Rashmi Reddy

2.3 Problem Statement Definition :

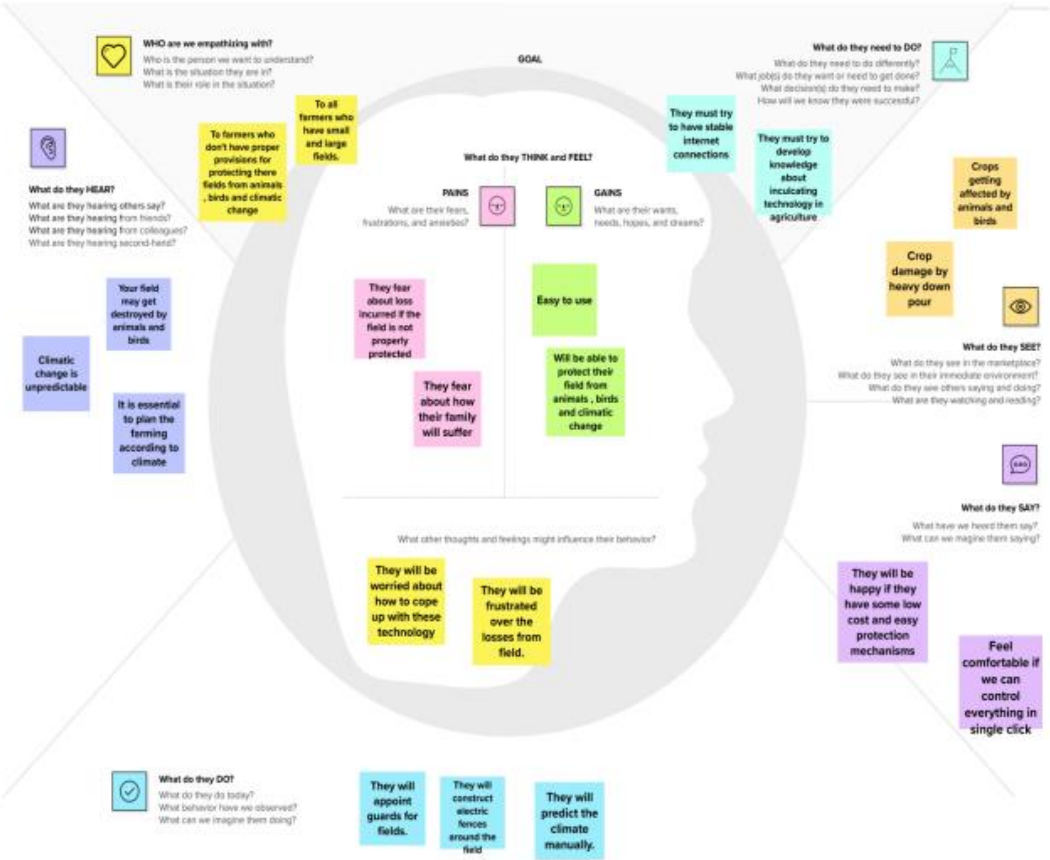
Agriculture is a huge contributor for our economy . Nowadays the biggest challenge faced by farmers in agriculture is the " Crop Depredation " due to wildlife interference. Many farmers have reported decrease in crop yield due to wild life attacks on crops. There is an urgent need to address this issue for the well being of farmers. Traditional methods followed by farmers are not that effective and it is not feasible to hire guards to keep an eye on crops and prevent wild animals , Since safety of both human and animal is equally vital. So there is need of advanced solution to address this issue. To overcome this issue , newer technologies like IOT and Wireless Sensor Networks are being used . The hardware implementation is done using Arduino and Raspberry pi.

3. Ideation & Proposed Solution :

The ideation and proposed solution is given below :

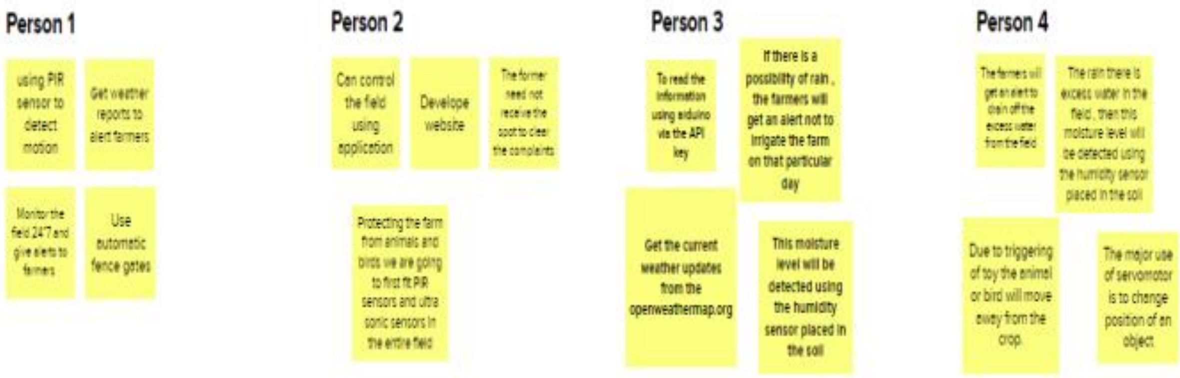
3.1 Empathy Map Canvas :

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes. It is a useful tool to helps teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.



3.2 Ideation & Brainstorming :

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions. Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.



3.3 Proposed Solution :

For the climate change issue , we are going to monitor the current climate of area and notify the farmer based on that information. For this we are going to use an web application called " **openweathermap.org**". From this application we are going to read the information using arduino via the API key. If there is a possibility of rain , the farmers will get an alert not to irrigate the farm on that particular day. If after the rain there is excess water in the field , then this moisture level will be detected using the humidity sensor placed in the soil . Then the farmers will get an alert to drain off the excess water from the field. Next for protecting the farm from animals and birds we are going to first fit PIR sensors and ultra sonic sensors in the entire field. PIR sensor will be used to detect the motion and ultra sonic sensor for measuring the distance of that animal from crop. The farmers will get an alert if the animal is within the range. As a means of protecting the farm from animals and birds the scarecrow toy attached to the servomotor will be triggered. The major use of servomotor is to change position of an object etc. Due to triggering of toy the animal or bird will move away from the crop.

Scalability of the Solution :

This project is very much efficient because it is based on IOT. As the whole process is going to be controlled by technology , there will be much less probability for error. The farmers will be able to control everything from there place itself and the actions will be quick. As we are using ultra sonic sensor along with PIR sensor , we will be able to measure the distance and continuous monitoring of the field , the farmers will be able to find the spot at which animal is and can trigger toy at that particular spot. Also as the weather is being monitored continuously , the farmers will be able to plan accordingly. Even the 16 days weather report can be already known , which makes the process more efficient.

3.4 Problem Solution Fit :

In this part of the project various questions asked about the project are answered. The questions are asked about the customer segments , problem to be solved , triggers , customer constraints , problem root cause , our proposed solution , available solution , behaviour , channels of behaviour , emotions of customer before & after the implementation of idea. After answering the above questions , the solution becomes more clear.

4. Requirement Analysis :

The various functional and non-functional requirements of the project are discussed below :

4.1 Functional Requirements :

The various functional requirements of project are ;

- **User Registration** : Registration through Form , Registration through G-mail.
- **User Confirmation** : Confirmation via E-mail , Confirmation via OTP.

- **openweathermap.org App :** From this application we are going to read the weather information using arduino via the API key.
- **Humidity sensor** : Moisture level will be detected using the humidity sensor placed in the soil . Then the farmers will get an alert to drain off the excess water from the field.
- **PIR sensor and Ultra sonic sensor** : PIR sensor will be used to detect the motion and ultra sonic sensor for measuring the distance of that animal from crop. The farmers will get an alert if the animal is within the range.
- **Servomotor :** The scarecrow toy attached with the servomotor will be triggered. The servomotor is generally used to change position of the objects.

4.2 Non-Functional Requirements :

The various non-functional requirements of the project are :

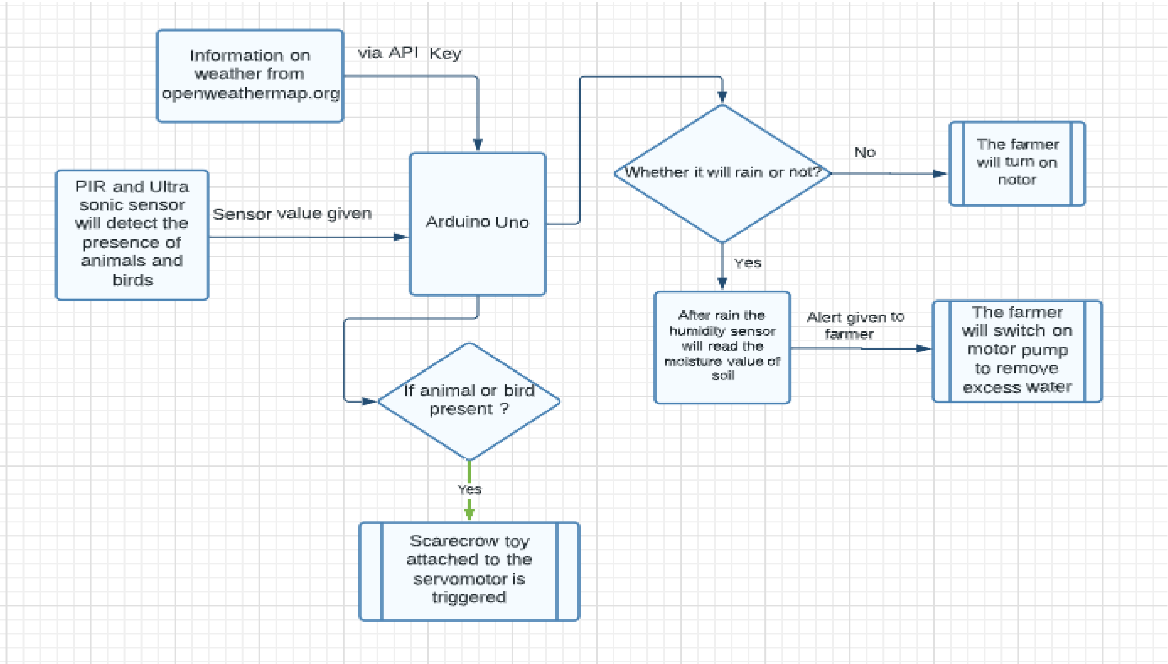
- **Usability :** The farmers will be able to control the operations in the field even from a longer distance.
- **Security :** This is used to protect farm lands from animals and birds and also climatic changes.
- **Reliability :** The farmers are capable of predicting and planning their farming practices in a more efficient way.
- **Performance :** It sends SMS to farmers when sensor detects the animal or birds movements and also alert them about the moisture in the soil.
- **Availability :** Through the development and deployment of software, we empower farmers to manage farmland.
- **Scalability:** This project is very much efficient because it is based on IOT.

5. Project Design :

This section concerns with the design of flow diagrams , solution and technical architectures and user stories.

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.



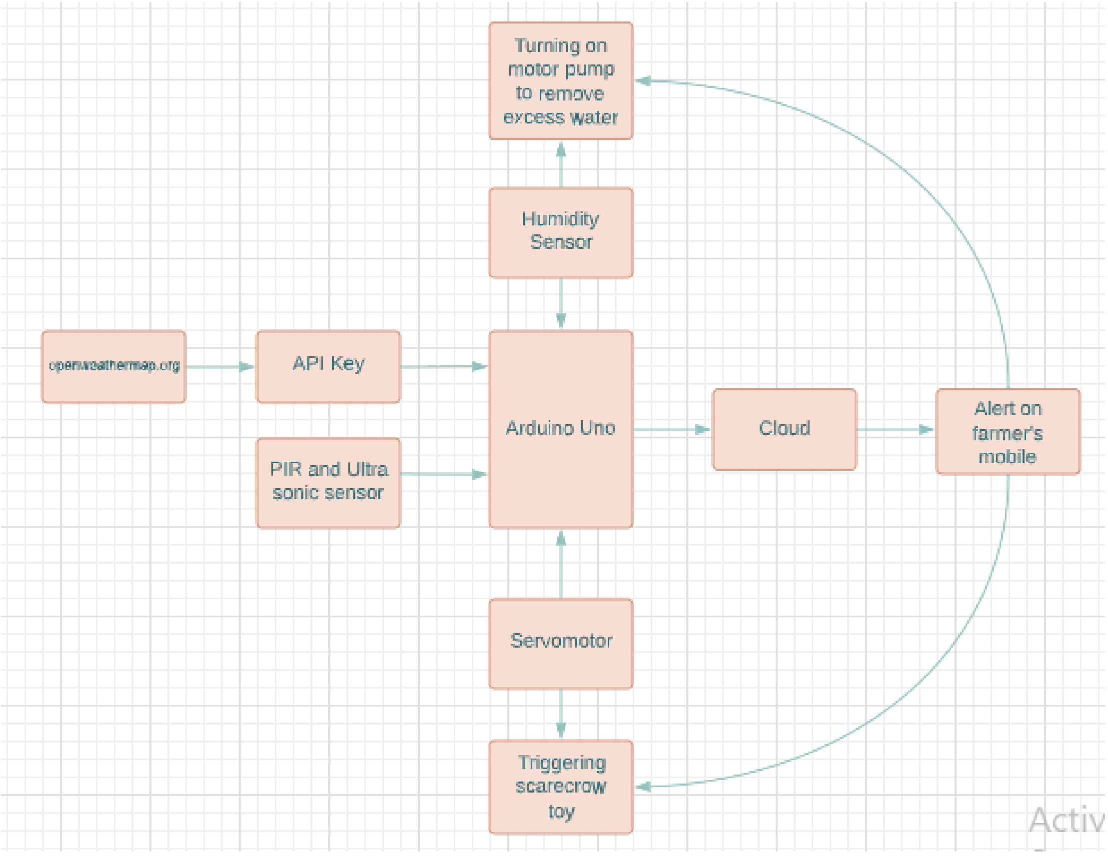
In this project the arduino uno is the processor used , it will detect the presence of animals and birds using PIR and Ultrasonic sensor , it will give the alert to farmers after the signal received. If there is a presence of animals or birds the scarecrow toy attached to the servomotor is triggered .

And information on weather is received with the help of openweathermap.org , after the message received it send a notification to the farmers mobile whether it will rain or not . If no the farmer will turn on the motor , if yes it will sent the moisture level of the soil and after the alert given to the farmer he will turn on the motor pump to remove the excess water in the farm land.

5.2 Solution & Technical Architecture :

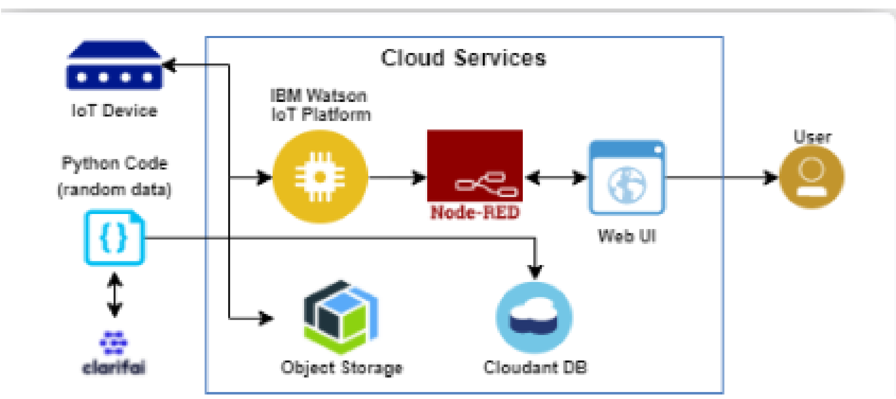
Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behaviour, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.



Technical Architecture :

- In Technical Architecture IOT device act as the hardware device which sends the message to the user using cloud services .
- In cloud services it contains some interfaces such as:
  - IBM Watson IOT Platform
  - Node-Red
  - Web UI
  - Object storage
  - Cloudant DB



In this above diagram it shows the process of technical architecture . That it shows that the IOT device will send the data to the cloud services and the IBM Watson IOT platform eill reseive the data and it passes to the node red and then it ir received by the web UI after it is received by the user . Python code (random data) is stored in cloudant DB . And object storage is will store the process from IOT device.

5.3 User Stories :

The various user stories are discussed here :

- As a user, I can register for the application by entering my email, password, and confirming my password.
- As a user, I will receive confirmation email once I have registered for the application
- As a user, I can register for the application through Gmail
- As a user, I can log into the application by entering email & password
- Weather forecast for that day can be seen.
- Humidity sensor value can be seen.
- It is used to control the motor pump
- The presence of animals and birds in the field is detected.
- Upon the presence of animal or bird , the toy at that spot is triggered

6. Project Planning & Scheduling :

This section contains the project planning and the schedules for various tasks.

6.1 Sprint Planning & Estimation :

Sprint 1 :

- As a user, I can register for the application by entering my email, password, and confirming my password.
- As a user can login using the email and password.
- Create the IBM Cloud services which are being used in this project.
- Configure the IBM Cloud services which are being used in completing this project.

Sprint 2 :

- IBM Watson IOT platform acts as the mediator to connect the web application to IOT devices, so create the IBM Watson IOT platform.
- In order to connect the IOT device to the IBM cloud, create a device in the IBM Watson IOT platform and get the device credentials.
- Configure the connection security and create API keys that are used in the Node-RED service for accessing the IBM IOT Platform.

Sprint 3 :

- To create a web application create a Node-RED service.
- Launch the cloudant DB and create a database to store the image URL.

Sprint 4 :

- Create a cloud object storage service, create a bucket to store the images, and configure the bucket settings.
- Develop a python script.
- Develop a python script to publish random sensor data such as temperature, moisture, soil and humidity to the IBM IOT platform.

6.2 Sprint Delivery Schedule :

The sprint delivery schedule is shown below :

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	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

Burn-Down Chart :

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

7. Coding :

7.1 Features :

In this project we send the weather data through IOT Simulator shown in instead of real soil and temperature conditions. Simulator passes the data through IBM Cloud to the web application. The data is displayed on the Dash board . Web Application is build using Node-RED. Web Application is also used to control the devices further like motor, pumps, or any other devices in the agricultural field. In this project the output is passed using python code and the control action is displayed in python code console window.

8. Result :

Thus we implemented the project using the IBM Cloud configuration services and python script and the project is executed successfully.

9. Advantages & Disadvantages :

Advantages :

- As the project is going to be IOT based , it is undeniably beneficial for saving the environment , reducing the cost and boosting the efficiency.
- The manpower used will be very much reduced , so that the farmer can save the labour cost. The farmer will also be able to save energy and time.
- The components used in this process will cost effective and will consume only less power.
- Weather is being monitored continuously , the farmers will be able to plan accordingly. Even the 16 days weather report can be already known , which makes the process more efficient.
- The farmers will be able control the activities in the field in a single click from anywhere in the world.
- This inclusion of technology will make the works simpler and it will pave way for the development of agricultural sector.

Disadvantages :

- Continuous and stable internet connection will be needed for proper working of system.
- The farmers may require an additional support to handle the system as the learning process is little tough.
- Technical failure may take place.

10. Conclusion :

In conventional methods the farmer will predict the climate and they plan accordingly but this can be inappropriate . Because nowadays the climate conditions are unpredictable . If there is excess water in the field and if the farmers are far away from their field , it will lead to huge losses. Also they appoint guards for protecting the farms at night time. But this approach will not be that feasible as it may be an threat to human life. Also if no one is there in the farm then animals may affect the crop. But due to inclusion of technology in agriculture the farmers will be able to control everything in a single click. This help them in saving a lot of time and energy. As this project will be based on IOT the farmers will be able to monitor their farm even if they are far away from the field. The details will also be more accurate than a human prediction. The whole process can also be automated so that the process can be more fast.

11. Applications :

- Precision Farming that is farming processes can be made more controlled and accurate.
- Live monitoring can be done of all the processes and the conditions on the agricultural field
- All the controls can be made just on the click.
- Quality can be maintained.

12. Future Scope :

In future due to more demand of good and more farming in less time, for betterment of the crops and reducing the usage of extravagant resources like electricity and water IoT can be implemented in most of the places.

13. Appendix :

Source Code :

Motor.py

import time

import sys

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

Provide your IBM Watson Device Credentials organization = "8gyz7t" # replace the ORG ID

deviceType = "weather_monitor" # replace the Device type deviceId = "b827ebd607b5" # replace Device ID authMethod = "token"

authToken = "LWVpQPaVQ166HWN48f" # Replace the authtoken

def myCommandCallback(cmd): # function for Callback if cmd.data['command'] == 'motoron':

print("MOTOR ON IS RECEIVED")

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

if cmd.command == "setInterval": if 'interval' not in cmd.data:

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

else:

interval = cmd.data['interval'] elif cmd.command == "print":

if 'message' not in cmd.data:

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

else:

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

try:

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

"auth-token": authToken}

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

except Exception as e:

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

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

deviceCli.connect()

while True:

deviceCli.commandCallback = myCommandCallback

Disconnect the device and application from the cloud deviceCli.disconnect()

Sensor.py

import time

import sys

import ibmiotf.application

import ibmiotf.device

```
import random

# Provide your IBM Watson Device Credentials organization = "8gyz7t"

# replace the ORG ID

deviceType = "weather_monitor"

# replace the Device type deviceId = "b827ebd607b5"

# replace Device ID authMethod = "token"

authToken = "LWVpQPaVQ166HWN48f"

# Replace the authtoken

def myCommandCallback(cmd):

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

try:

deviceOptions = {"org": organization, "type": deviceType, "id": deviceId,

"auth-method": authMethod, "auth-token": authToken} deviceCli =
ibmiotf.device.Client(deviceOptions) #...................

...........................

except Exception as e:

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

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

deviceCli.connect()

while True:

temp=random.randint(0,100) pulse=random.randint(0,100) soil=random.randint(0,100)

data = { 'temp' : temp, 'pulse': pulse ,'soil':soil}

#print data

def myOnPublishCallback():

print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % pulse,"Soil
Moisture = %s %%" % soil,"to IBM Watson")
```

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

if not success:

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

deviceCli.commandCallback = myCommandCallback

Disconnect the device and application from the cloud deviceCli.disconnect()

Node-RED Flow :

[

{ "id":"625574ead9839b34",

"type":"ibmiotout", "z":"630c8601c5ac3295",

"authentication":"apiKey",

"apiKey":"ef745d48e395ccc0",

"outputType":"cmd",

"deviceId":"b827ebd607b5",

"deviceType":"weather_monitor",

"eventCommandType":"data",

"format":"json",

"data":"data",

"qos":0, "name":"

IBM IoT",

"service":"registered",

"x":680,

"y":220,

"wires":[]

},

{

"id":"4cff18c3274cccc4", "type":"ui_button", "z":"630c8601c5ac3295", "name":"", "group":"716e956.00eed6c", "order":2,

"width":"0",

"height":"0", "passthru":false, "label":"MotorON",

"tooltip":"",

"color":"",

"bgcolor":"",

"className":"",

"icon":"", "payload":"{\"command\":\"motoron\"}", "payloadType":"str",

"topic":"motoron",

"topicType":"str", "x":360,

"y":160, "wires":[["625574ead9839b34"]]},

{

"id":"659589baceb4e0b0", "type":"ui_button", "z":"630c8601c5ac3295",

"name":"", "group":"716e956.00eed6c",

"order":3,

"width":"0",

"height":"0", "passthru":true, "label":"MotorOFF",

"tooltip":"",

"color":"",

"bgcolor":"",

"className":"",

"icon":"", "payload":"{\"command\":\"motoroff\"}", "payloadType":"str",

"topic":"motoroff",

"topicType":"str", "x":350,

"y":220, "wires":[["625574ead9839b34"]]},

{"id":"ef745d48e395ccc0", "type":"ibmiot", "name":"weather_monitor", "keepalive":"60",

"serverName":"", "cleansession":true, "appId":"", "shared":false},

{"id":"716e956.00eed6c",

"type":"ui_group",

"name":"Form", "tab":"7e62365e.b7e6b8", "order":1,

"disp":true, "width":"6", "collapse":false},

{"id":"7e62365e.b7e6b8",

"type":"ui_tab",

"name":"contorl",

"icon":"dashboard", "order":1, "disabled":false, "hidden":false}

]

[

{

"id":"b42b5519fee73ee2", "type":"ibmiotin", "z":"03acb6ae05a0c712", "authentication":"apiKey", "apiKey":"ef745d48e395ccc0", "inputType":"evt", "logicalInterface":"", "ruleId":"", "deviceId":"b827ebd607b5", "applicationId":"",

"deviceType":"weather_monitor", "eventType":"+",

"commandType":"",

"format":"json",

"name":"IBMIoT", "service":"registered", "allDevices":"", "allApplications":"", "allDeviceTypes":"", "allLogicalInterfaces":"", "allEvents":true, "allCommands":"",

"allFormats":"",

"qos":0,

"x":270,

"y":180,

"wires":[["50b13e02170d73fc","d7da6c2f5302ffaf","a949797028158f3f","a71f164bc378bcf1"]]

},

{ "id":"50b13e02170d73fc",

"type":"function", "z":"03acb6ae05a0c712", "name":"Soil Moisture",

"func":"msg.payload = msg.payload.soil;\nglobal.set('s',msg.payload);\nreturn msg;",
"outputs":1,

"noerr":0, "initialize":"",

"finalize":"",

"libs":[], "x":490,

"y":120,

"wires":[["a949797028158f3f","ba98e701f55f04fe"]]

},

{

"id":"d7da6c2f5302ffaf", "type":"function", "z":"03acb6ae05a0c712", "name":"Humidity",

"func":"msg.payload = msg.payload.pulse;\nglobal.set('p',msg.payload)\nreturn msg;",
"outputs":1,

"noerr":0, "initialize":"",

"finalize":"",

"libs":[], "x":480,

"y":260, "wires":[["a949797028158f3f","70a5b076eeb80b70"]]

},

{ "id":"a949797028158f3f",

"type":

"debug",

"z":"03acb6ae05a0c712",

"name":"IBMo/p",

"active":true,

"tosidebar":true,

"console":false,

"tostatus":false,

"complete":"payload",

"targetType":"msg",

"statusVal":"", "statusType":"auto", "x":780,

"y":180,

"wires":[]

},

{

"id":"70a5b076eeb80b70", "type":"ui_gauge", "z":"03acb6ae05a0c712", "name":"", "group":"f4cb8513b95c98a4", "order":6,

"width":"0",

"height":"0",

"gtype":"gage",

"title":"Humidity",

"label":"Percentage(%)",

"format":"{{value}}", "min":0, "max":"100",

"colors":["#00b500","#e6e600","#ca3838"],

"seg1":"",

"seg2":"",

"className":"", "x":860,

"y":260,

"wires":[]

},

{

"id":"a71f164bc378bcf1", "type":"function", "z":"03acb6ae05a0c712", "name":"Temperature",

"func":"msg.payload=msg.payload.temp;\nglobal.set('t',msg.payload);\nreturn msg;", "outputs":1,

"noerr":0, "initialize":"",

"finalize":"",

"libs":[], "x":490,

"y":360,

"wires":[["8e8b63b110c5ec2d","a949797028158f3f"]]

},

{

"id":"8e8b63b110c5ec2d", "type":"ui_gauge", "z":"03acb6ae05a0c712", "name":"", "group":"f4cb8513b95c98a4", "order":11,

"width":"0",

"height":"0",

"gtype":"gage", "title":"Temperature", "label":"DegreeCelcius",

"format":"{{value}}", "min":0, "max":"100",

"colors":["#00b500","#e6e600","#ca3838"], "seg1":"",

"seg2":"",

"className":"", "x":790,

"y":360,

"wires":[]

},

{

"id":"ba98e701f55f04fe", "type":"ui_gauge", "z":"03acb6ae05a0c712", "name":"", "group":"f4cb8513b95c98a4", "order":1,

"width":"0",

"height":"0",

"gtype":"gage", "title":"Soil Moisture", "label":"Percentage(%)",

"format":"{{value}}", "min":0, "max":"100",

"colors":["#00b500","#e6e600","#ca3838"], "seg1":"",

"seg2":"",

"className":"", "x":790,

"y":120,

"wires":[]

},

{

"id":"a259673baf5f0f98", "type":"httpin", "z":"03acb6ae05a0c712", "name":"",

"url":"/sensor",

"method":"get", "upload":false, "swaggerDoc":"", "x":370,

"y":500,

"wires":[["18a8cdbf7943d27a"]]

},

{

"id":"18a8cdbf7943d27a", "type":"function", "z":"03acb6ae05a0c712",
"name":"httpfunction",

"func":"msg.payload{\"pulse\":global.get('p'),\"temp\":global.get('t'),\"soil\":global.get(
's')};\nreturn msg;",

"outputs":1,

"noerr":0, "initialize":"",

"finalize":"",

"libs":[], "x":630,

"y":500, "wires":[["5c7996d53a445412"]]

},

{ "id":"5c7996d53a445412",

"type":"httpresponse", "z":"03acb6ae05a0c712", "name":"",

"statusCode":"",

"headers":{}, "x":870,

"y":500,

"wires":[]

},

{

"id":"ef745d48e395ccc0", "type":"ibmiot", "name":"weather_monitor", "keepalive":"60",

"serverName":"", "cleansession":true, "appId":"",

"shared":false},

{

"id":"f4cb8513b95c98a4", "type":"ui_group",

"name":"monitor", "tab":"1f4cb829.2fdee8", "order":2,

"disp":true, "width":"6", "collapse":false, "className":""

},

{

"id":"1f4cb829.2fdee8",

"type":"ui_tab",

"name":"Home",

"icon":"dashboard", "order":3, "disabled":false, "hidden":false }

Github link : https://github.com/IBM-EPBL/IBM-Project-43427-1660716854