IOT BASED SMART CROP PROTECTION SYSTEM FOR AGRICULTURE

A PROJECT REPORT

Submitted by

G.MAHALAKSHMI	**(952319106015)**
M.MEENA	**(952319106020)**
M.MUTHU PRIYA	**(952319106022)**
M.NARMATHA	**(952319106023)**
A.SRITHAMBIRATTI	**(952319106301)**

In partial fulfillment for the award of the degree

Of

BACHELOR OF ENGINEERING

In

ELECTRONICS AND COMMUNICATION ENGINEERING

PSN ENGINEERING COLLEGE, MELATHEDIYOOR

TIRUNELVELI-627152

AFFILIATED TO ANNA UNIVERSITY-CHENNAI 600 025

NOVEMBER 2022

BONAFIDE CERTIFICATE

Certified that this report titled "**IOT BASED SMART CROP PROTECTION SYSTEM FOR AGRICULTURE"** is the bonafide work of **"G.MHALAKSHMI (952319106015), M.MEENA (952319106020), M.MUTHU PRIYA (952319106022), M.NARMATHA (952319106023), A.SRITHAMBIRATTI (952319106301)"**, who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Mr.S.PITCHAIAH M.E.,	**Mr.S.PITCHAIAH M.E.,**
HEAD OF THE DEPARTMENT,	**IBM SPOC,**
Department of ECE,	Department of ECE,
Melathediyoor,	Melathediyoor,
Tirunelveli-627152	Tirunelveli-627152

Submitted for the Project work and Viva-Voce examination held on …………

MENTOR	**EVALUVATOR**

ABSTRACT

Internet of Things (IoT) plays a crucial role in smart agriculture. Smart farming is an emerging concept, because IoT sensors capable of providing information about their agriculture fields. The project aims making use of evolving technology i.e. IoT and smart agriculture using automation. Monitoring environmental factors is the major factor to improve the yield of the efficient crops. The feature of this project includes monitoring temperature, humidity and moisture in agricultural field through sensors DHT11, YL69. It will turn ON/OFF motor on the basis of soil moisture.

CONTENT

1.INTRODUCTION

1.1 PROJECT OVERVIEW:-

The main aim of this project is to help farmers automate their farms by providing them with a Web App through which they can monitor the parameters of the field like Temperature, soil moisture, humidity and etc and control the equipment like water motor and other devices remotely via internet without their actual presence in the field.

1.2 PURPOSE:

Agriculture has always been the most important economic sector in India. Despite the fact that agriculture employs the majority of India's population, farmers face numerous challenges. Deforestation happens as a result of overpopulation, depriving forest regions of water, food, and shelter. As a result, animal incursion into residential areas is increasing day by day, posing a threat to human life and property, as well as causing conflict between humans and animals. Agriculture is the backbone of the economy, but animal intrusion in agricultural land would result in massive crop loss. Elephants and other animals interacting with humans have a negative influence in a variety of ways, including crop devastation, damage to food stockpiles, water supplies, homes and properties, injuries, and human mortality. Conflict between human beings may also be a serious problem where large quantities of money are wasted and life is at risk. In recent times the numbers of those types of conflicts are increasing. Farmers in India has been facing serious threats from natural calam

ities, pests and damage by animals leading to lower yields. Ancient strategies are being followed by farmers aren't much effective and it's not being feasible to hire guards to keep an eye fixed on the crops and forestall the wild animals. Therefore this zone is to be monitored continuously to prevent entry of this kind of animals or the other unwanted intrusion. So, animal detection system is being vital in farm areas. Crops are mainly damaged by local animals like buffaloes, cows, pigs, goats, birds, and fire, etc. This leads to huge losses for the farmers. Agriculture farming is the main source of livelihood for many people in different parts of the world. Regrettably, farmers continue to rely on centuries-old ways. Crop yields are decreasing as a result of this. Furthermore, a number of factors lead to reduced agricultural yields, one of which being animal encroachment. In recent years, farmers all across the world have faced a unique challenge: wild animals. Wild creatures such as wild boars, elephants, deer, tigers, and monkeys, among others, trample crops by running across them. As a result, farmers are facing financial challenges. Farmers must physically irrigate a large quantity of agricultural land, which takes a long time.

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM:-

Animals are shocked by electric fences, and there is a risk of fire if plants or shrubs grow too close to the fence. If the fence is not adequately maintained, electromagnetic interference occurs, interfering with telephone and radio signals. Despite being the most often utilised farm protection measure, electric fencing is harmful to both animals and humans. Thorn fencing, which is also a widely used tactic, provides a similar effect to the prior technique.

2.2 REFERENCES:-

[1] M. De Clercq, A. Vats, and A. Biel, ‘‘Agriculture 4.0: The future of farming technology,’’ in Proc. World Government Summit, Dubai, UAE, 2018, pp. 11–13.

[2]

[3] Y. Liu, X. Ma, L. Shu, G. P. Hancke, and A. M. Abu-Mahfouz, ‘‘From industry 4.0 to agriculture 4.0: Current status, enabling technologies, and research challenges,’’ IEEE Trans. Ind. Informat., vol. 17, no. 6, pp. 4322–4334, Jun. 2021.

[4] M. S. Farooq, S. Riaz, A. Abid,K. Abid, and M. A. Naeem, ‘‘A survey on the role of IoT in agriculture for the implementation of smart farming,’’ IEEE Access, vol. 7, pp. 156237–156271, 2019.

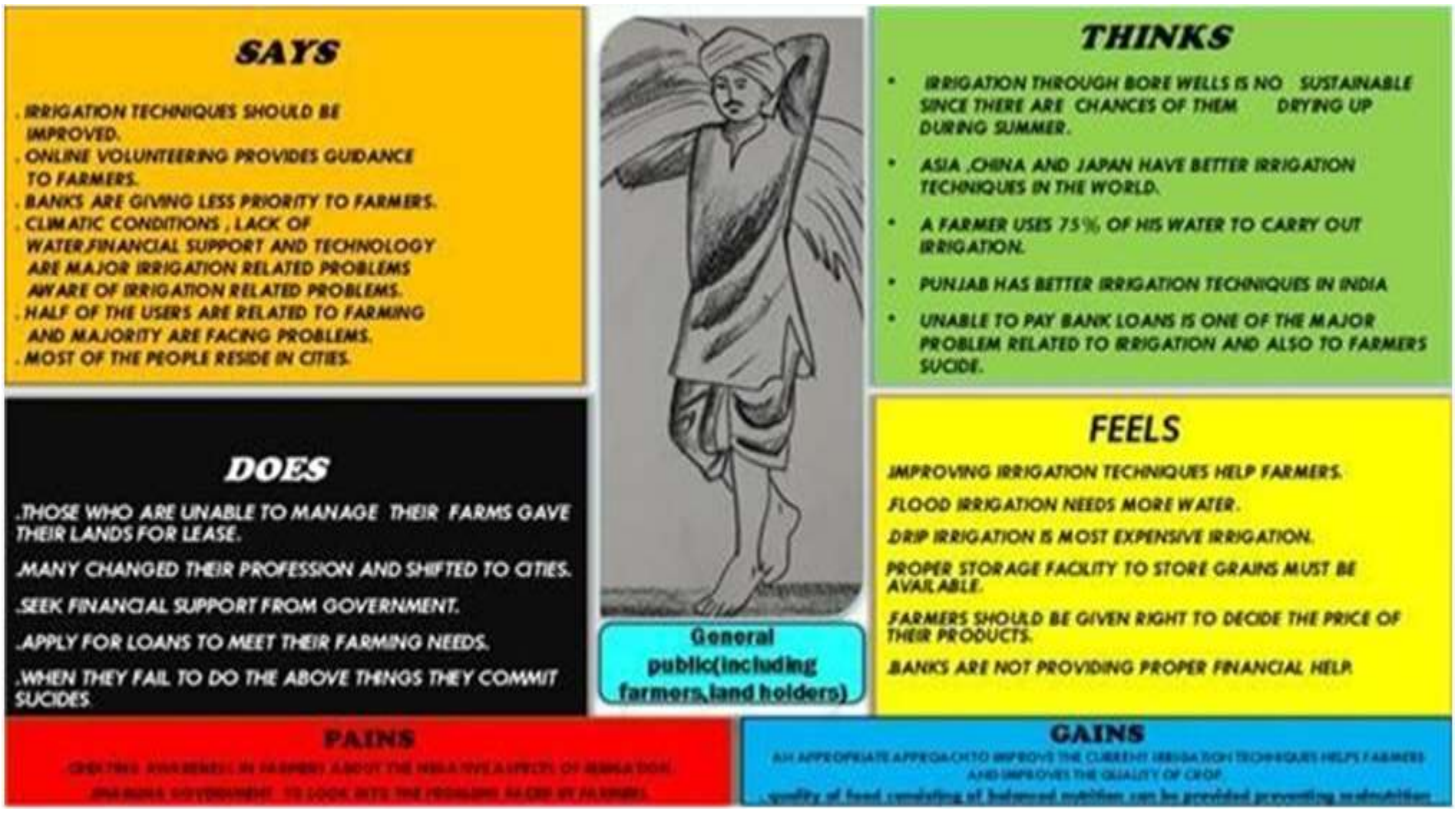
[5] K. Kirkpatrick, ‘‘Technologizing agriculture,’’ Commun. ACM, vol. 62, no. 2, pp. 14–16, Jan. 2019.

2.3 PROBLEM STATEMENT DEFINITION:-

Farmers are to be present at farm for its maintenance irrespective of the weather conditions. They have to ensure that the crops are well watered and the farm status is monitored by them physically. Farmer have to stay most of the time in field in order to get a good yield. In difficult times like in the presence of pandemic also they have to work hard in their fields risking their lives to provide food for the country. Animal attacks are a typical occurrence in India these days. These attacks kill villages and destroy their crops due to the lack of any detection system. These folks are helpless in the face of their fate due to a lack of sufficient safety precautions. As a result, a proper detection system could assist save their lives as well as the crops. Villagers' crops are also ruined as a result of animal meddling. The rapid loss of forests and encroaching crop area has resulted in an increase in animal invasion of fields, causing a significant shift in farmers' attitudes toward them. Harmony between a farmer and wild animals appears to be the next best thing to a miracle.

3.IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP:-



SAYS

.IRRIGATION TECHNIQUES SHOULD BE IMPROVED.
.ONLINE VOLUNTEERING PROVIDES GUIDANCE TO FARMERS.
.BANKS ARE GIVING LESS PRIORITY TO FARMERS.
.CLIMATIC CONDITIONS , LACK OF WATER,FINANCIAL SUPPORT AND TECHNOLOGY ARE MAJOR IRRIGATION RELATED PROBLEMS AWARE OF IRRIGATION RELATED PROBLEMS.
.HALF OF THE USERS ARE RELATED TO FARMING AND MAJORITY ARE FACING PROBLEMS.
.MOST OF THE PEOPLE RESIDE IN CITIES.

THINKS

- IRRIGATION THROUGH BORE WELLS IS NO SUSTAINABLE SINCE THERE ARE CHANCES OF THEM DRYING UP DURING SUMMER.
- ASIA ,CHINA AND JAPAN HAVE BETTER IRRIGATION TECHNIQUES IN THE WORLD.
- A FARMER USES 75% OF HIS WATER TO CARRY OUT IRRIGATION.
- PUNJAB HAS BETTER IRRIGATION TECHNIQUES IN INDIA
- UNABLE TO PAY BANK LOANS IS ONE OF THE MAJOR PROBLEM RELATED TO IRRIGATION AND ALSO TO FARMERS SUCIDE.

DOES

.THOSE WHO ARE UNABLE TO MANAGE THEIR FARMS GAVE THEIR LANDS FOR LEASE.
.MANY CHANGED THEIR PROFESSION AND SHIFTED TO CITIES.
.SEEK FINANCIAL SUPPORT FROM GOVERNMENT.
.APPLY FOR LOANS TO MEET THEIR FARMING NEEDS.
.WHEN THEY FAIL TO DO THE ABOVE THINGS THEY COMMIT SUCIDES.

General public(including farmers,land holders)

FEELS

.IMPROVING IRRIGATION TECHNIQUES HELP FARMERS.
.FLOOD IRRIGATION NEEDS MORE WATER.
.DRIP IRRIGATION IS MOST EXPENSIVE IRRIGATION.
.PROPER STORAGE FACILITY TO STORE GRAINS MUST BE AVAILABLE.
.FARMERS SHOULD BE GIVEN RIGHT TO DECIDE THE PRICE OF THEIR PRODUCTS.
.BANKS ARE NOT PROVIDING PROPER FINANCIAL HELP.

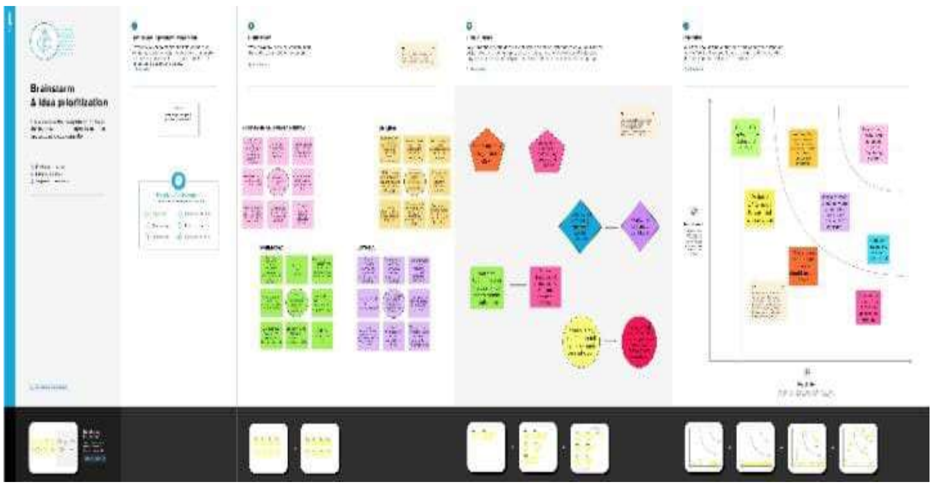
PAINS

CREATING AWARENESS IN FARMERS ABOUT THE NEGATIVE ASPECTS OF IRRIGATION.
ENGAGING GOVERNMENT TO LOOK INTO THE PROBLEMS FACED BY FARMERS.

GAINS

AN APPROPRIATE APPROACH TO IMPROVE THE CURRENT IRRIGATION TECHNIQUES HELPS FARMERS AND IMPROVES THE QUALITY OF CROP.
.quality of food consisting of balanced nutrition can be provided preventing malnutrition

3.2 IDEATION & BRAINSTORMING:-



3.3.PROPOSED SOLUTION:-

In order to improve the farmer’s working conditions and make them easier, weintroduce IoT services to him in which we use cloud services and internet to enable farmer to continue his work remotely via internet. He can monitor the field parameters and control the devices in farm.

3.4 PROBLEM SOLUTION FIT:-

Farmers are to be present at farm for its maintenance irrespective of the weather conditions. They have to ensure that the crops are well watered and the farm status is monitored by them physically. Farmer have to stay most of the time in field in order to get a good yield. In difficult times like in the presence of pandemic also they have to work hard in their fields risking their lives to provide food for the country. Animal attacks are a typical occurrence in India these days. These attacks kill villages and destroy their crops due to the lack of any detection system. These folks are helpless in the face of their fate due to a lack of sufficient safety precautions. As a result, a proper detection system could assist save their lives as well as the crops. Villagers' crops are also ruined as a result of animal meddling. The rapid loss of forests and encroaching crop area has resulted in an increase in animal invasion of fields, causing a significant shift in farmers' attitudes toward them. Harmony between a farmer and wild animals appears to be the next best thing to a miracle.

4.REQUIREMENT ANALYSIS

4.1 FUNCTION REQUIREMENT :

Following are the functional requirements of the proposed solution.

S.No	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Use many types of seeds	To grow the crops
FR-4	Growth perfect crop protection	Used sunlight ,irrigation,sand enrichment,natural Fertilizers
FR-5	Used more fertilizers	To yields many profits in agriculture

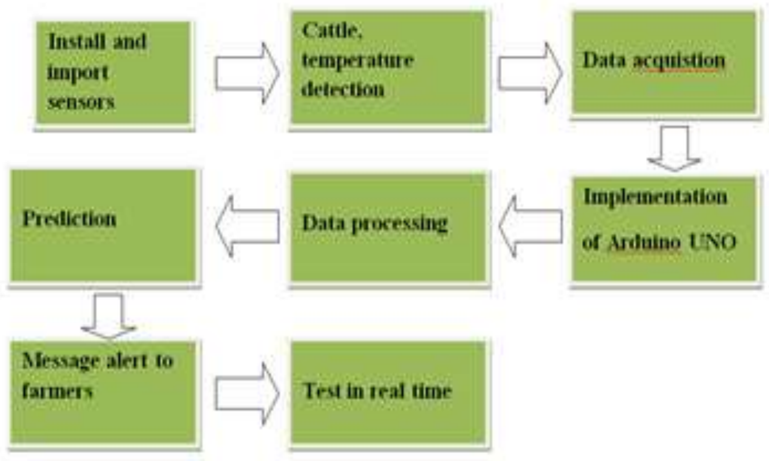
Non-functional Requirements:

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

S. No	Non-Functional Requirement	Description
NFR-1	**Usability**	To less hard work,to save time ,to increase more Yields
NFR-2	**Security**	To protect animal attacks
NFR-3	**Reliability**	To not affect weather conditions
NFR-4	**Performance**	High level perform
NFR-5	**Availability**	To decide human trace
NFR-6	**Scalability**	To produce easy crop maintaining

5.PROJECT DESION

5.1 DATA FLOW DIAGRAM


Install and import sensors
Cattle, temperature detection
Data acquistion
Implementation of Arduino UNO
Data processing
Prediction
Message alert to farmers
Test in real time

Flow:

- We start collecting data from cloud services and collect abunch of data from sensors.
- Save data in the form of numpy arrays
- We then implement arduino UNO with our stored data.
- The number of sensors for the model is determined by us, ifwe increase the number of sensors, the accuracy increases. But it requires much more time for implementing more sensors.
- Once detection is done, we can use this model for real time cattle

detection and simultaneously used to detect water level and temperature in the field.

5.2 SOLUTION & TECHNICAL ARCHITECTURE:-

Technical Architecture:

The Deliverable shall include the architectural diagram as below andthe information as per the table1 & table 2

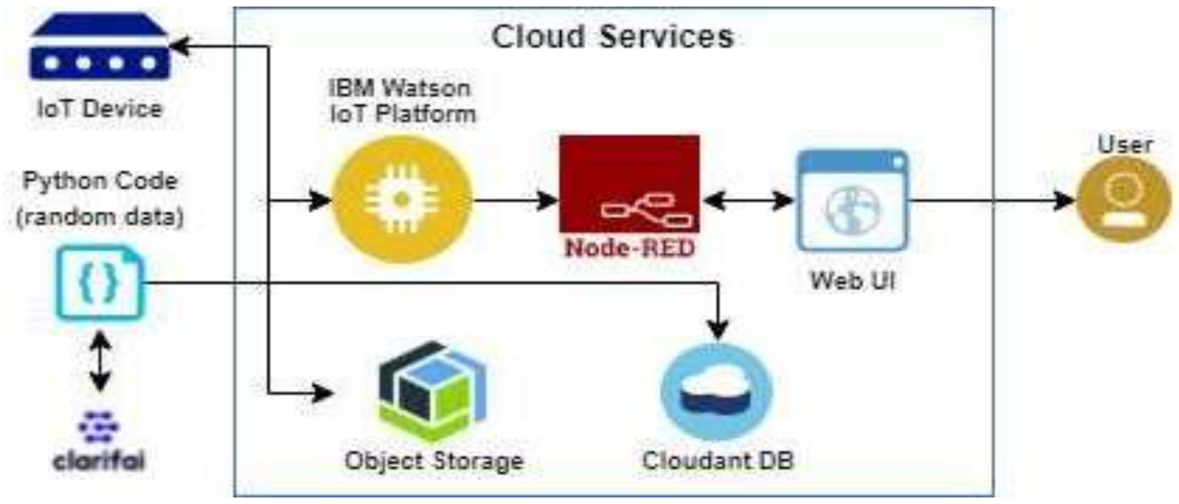


Table-1: Components &Technologies

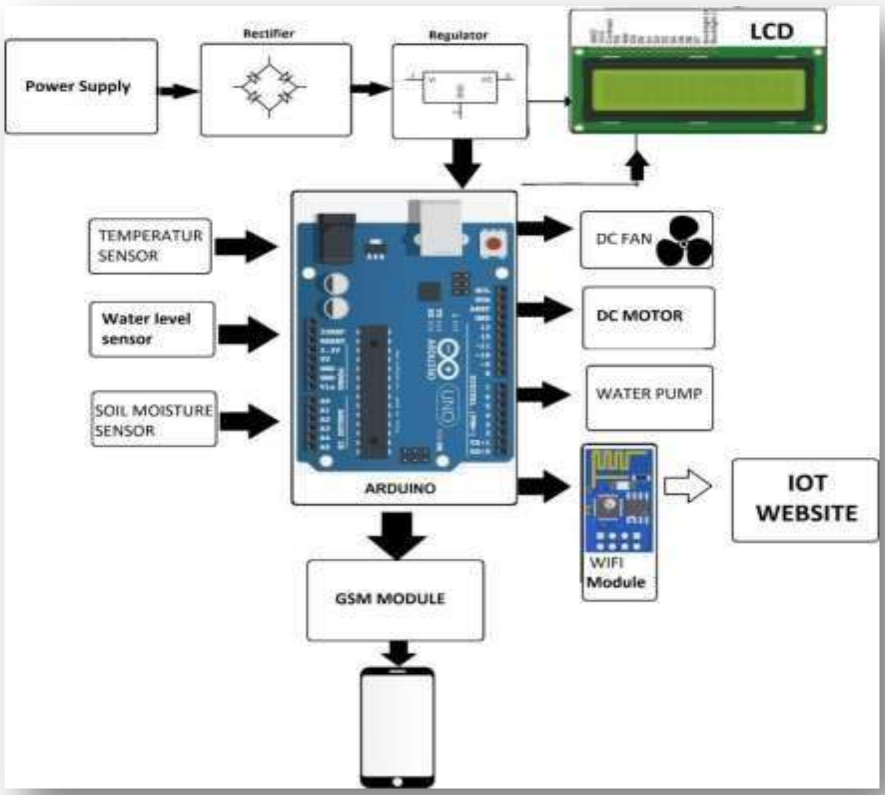
S.No	Component	Description	Technology
1.	User interface	How user interacts withthe web UI	HTML, CSS, JavaScript / AngularJs / React Js etc.
2.	Application Logic-1	Logic for a process in the application	Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson/node Red
4.	Application Logic-3	Logic for a process in the application	IBM Watson/node Red
5.	Database	Data Type,Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on cloud	IBM Cloudant

7.	File Storage	File storage requirements	IBM Block Storage
8.	Infrastructure (Server/cloud)	Application deploymenton Local System/CloudLocal Server configuration: Cloud sever configuration	Cloud Foundry

Table-2:Application Characteristics

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	The open-source frameworks used	Software
2.	Security implementation	List all the security/access controls implemented	Encryption process
3.	Scalable architecture	Justify the scalability of architecture(3-tier, micro-services)	Software
4.	Availability	Justify the availability of applications (eg. useof load balancers, distributed servers etc)	Software
5.	Performance	Design consideration for the performance of the application	Software

FLOW:



5.3 USER STORIES :-

User Type	Functional Requirement (Epic)	User Story Number	User Story/Task	Acceptance Criteria	Priority	Release
Developer	System Building	USN-1	Collect dataset	I can collect dataset	High	Sprint-1
		USN-2	Collectingdata from sensors	I can collect datafrom sensors	High	Sprint-1
		USN-3	Implementing arduino UNO from data collection		High	Sprint-2

		USN-4	Message alertto farmers	I can receive message	High	Sprint-3

6.PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION :-

TITLE	DESCRIPTION	DATE
Literature Survey on TheSelected Project and Information Gathering	A Literature Survey is a compilation summary of research done previously in the given topic. Literature survey can be taken from books, research paper online or from any source.	10 October 2022
Prepare Empathy Map	Empathy Map is a visualization tool which can be used to get a better insight of the customer	10 October 2022
Ideation-Brainstorming	Brainstorming is a group problem solving session where ideas are shared, discussed and organized among the team members.	10 October 2022
Define Problem Statement	A Problem Statement is a concise description of the problem or issues a project seeks to address. The problem statement identifies the current state, the desired future state and any gaps between the two.	10 October 2022
Data Flow Diagrams	Data Flow Diagram is a graphical or visual representation using a standardized set of symbols and notations to describe a business's operations through data movement.	13 October 2022
Technology Architecture	Technology Architecture is a more well defined version of solution architecture. It helps us analyze and understand various technologies that needs to be implemented in the project.	13 October 2022
Prepare Milestone & ActivityList	It helps us to understand and evaluate our own progress and accuracy so far.	1 November 2022

Spring Delivery Plan	Sprint planning is an event in scrum that kicks off the sprint. The purpose of sprint planning is to define what can be delivered in the sprint and how that work will be achieved.	1 November 2022
Problem Solution Fit	This helps us to understand the thoughts of the customer their likes, behavior, emotions etc.	10 October 2022
Proposed Solution	Proposed solution shows the current solution and it helps is going towards the desired result until it is achieved.	10 October 2022
Solution Architecture	Solution Architecture is a very complex process i.e it has a lot of sub-processes and branches. It helps in understanding the components and features to complete our project.	13 October 2022
Customer Journey	It helps us to analyze from the perspective of a customer, who uses our project.	13 October 2022
Functional Requirement	Here functional and non functional requirements are briefed. It has specific features like usability	13 October 2022

6.2 SPRINT DELIVERY SCHEDULE :-

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	MAHALAKSHMI MEENA MUTHU PRIYA NARMATHA SRITHAMBIRATTI
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	MAHALAKSHMI MEENA MUTHU PRIYA NARMATHA SRITHAMBIRATTI
Sprint-2		USN-3	As a user, I can register for the application through Facebook	2	Low	MAHALAKSHMI MEENA MUTHU PRIYA NARMATHA SRITHAMBIRATTI
Sprint-1		USN-4	As a user, I can register for the application through Gmail	2	Medium	MAHALAKSHMI MEENA MUTHU PRIYA NARMATHA SRITHAMBIRATTI

Sprint-1	Login	USN-5	As a user, I can log into the application byentering email & password	1	High	MAHALAKSHMI MEENA MUTHUPRIYA NARMATHA SRITHAMBIRATTI
Sprint-2	Dashboard	USN-6	The explored and visualized data aredisplayed in dashboard	2	High	MAHALAKSH MIMEENA MUTHU PRIYA NARMATHA SRITHAMBIRATTI

Project Tracker, Velocity & Burndown Chart: (4 Marks)

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	19	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	18	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

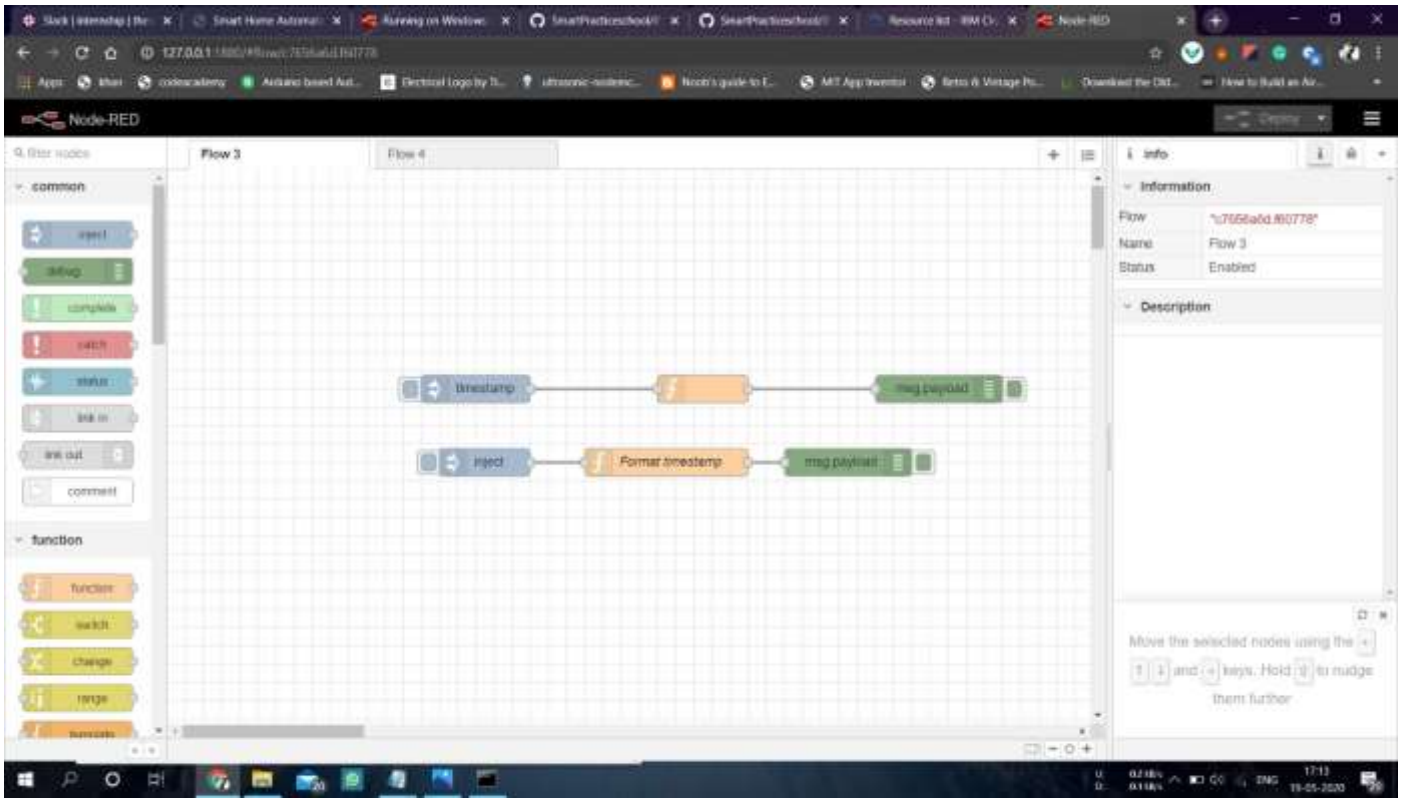
Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let’s calculate the team’s average velocity (AV) per iteration unit (story points per day)

7.CODING &SOLUTIONING

Node-RED is a flow-based development tool for visual programming developed originally by IBM for wiring together hardware devices, APIs and online services as part of the Internet of Things.

Node-RED provides a web browser-based flow editor, which can be used to create JavaScript functions.



Installation :

First install npm/node.js

Open cmd prompt

Type =>   npm install node-red

To run the application :

Open cmd prompt

Type=>   node-red

Then open http://localhost:1880/ in browser

Installation of IBM IoT and Dashboard nodes for Node-Red

In order to connect to IBM Watson IoT platform and create the Web App UI these nodes are required

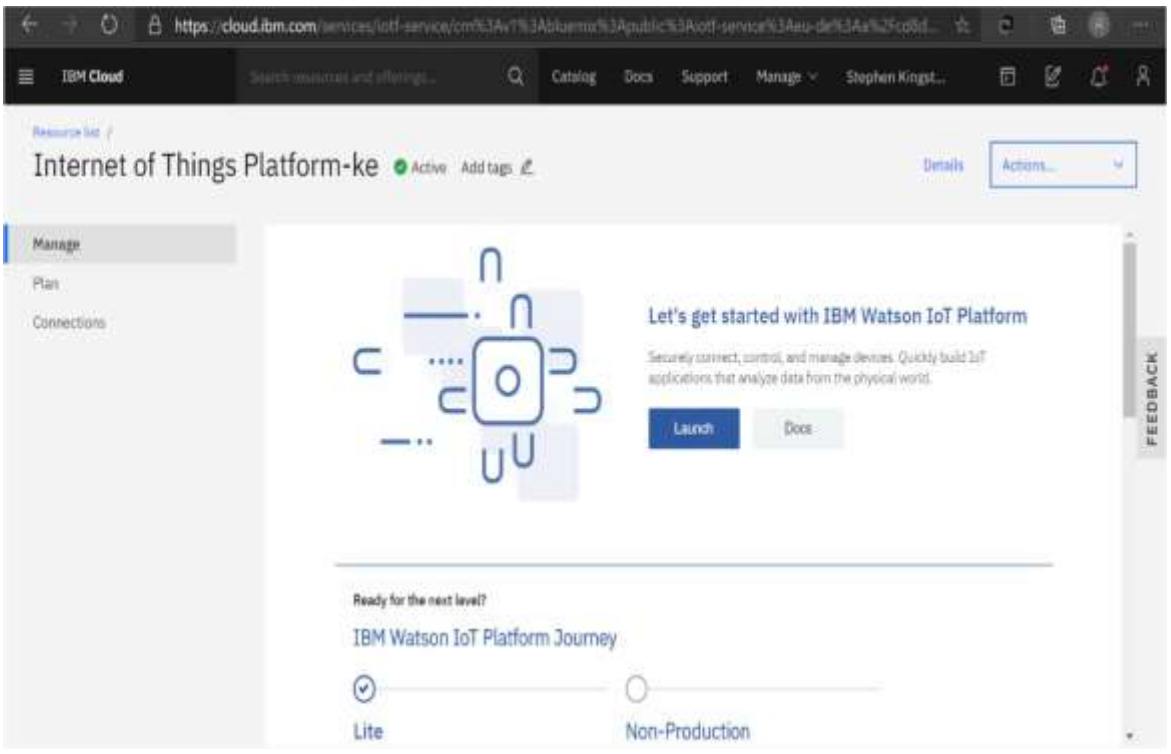
IBM IoT node

Dashboard node

IBM Watson IoT Platform

A fully managed, cloud-hosted service with capabilities for device registration, connectivity, control, rapid visualization and data storage.

IBM Watson IoT Platform is a managed, cloud-hosted service designed to make it simple to derive value from your IoT devices.



Securely connect, control, and manage devices. Quickly build IoT applications that analyze data from the physical world.

Steps to configure:

Create an account in IBM cloud using your email ID

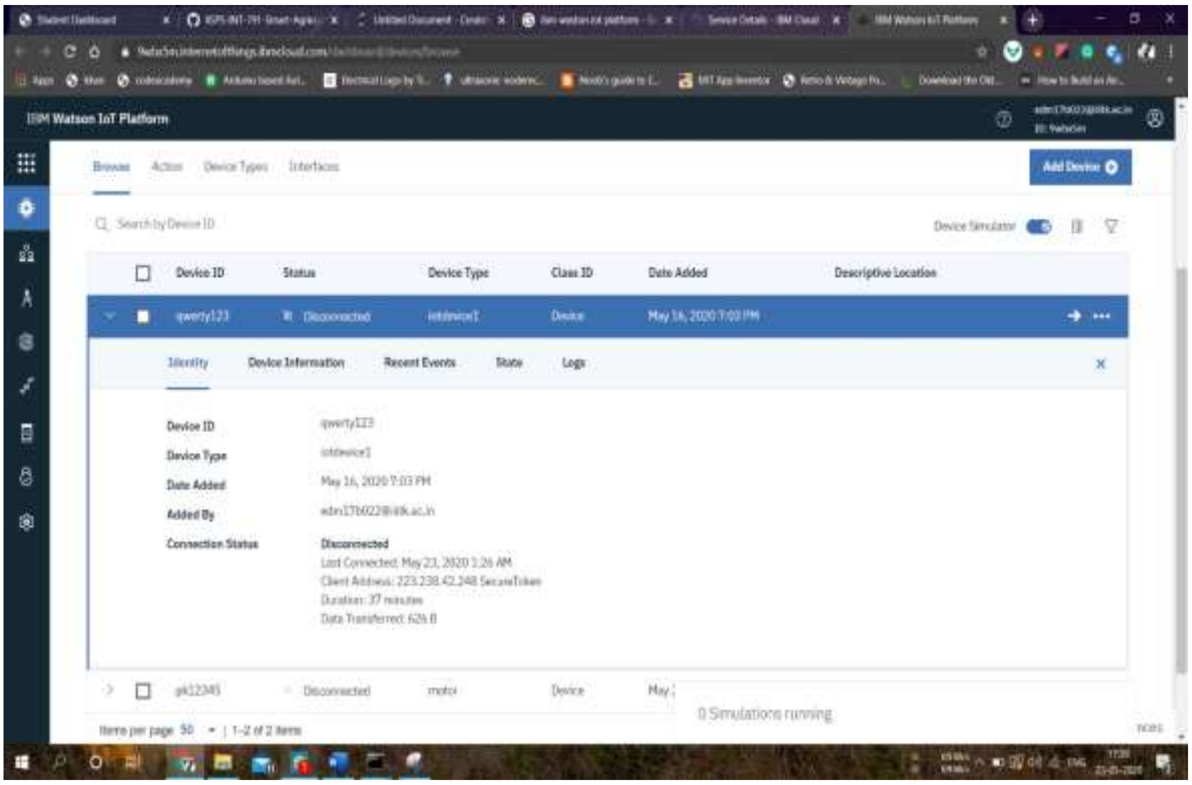
Create IBM Watson Platform in services in your IBM cloud account

Launch the IBM Watson IoT Platform

Create a new device

Give credentials like device type, device ID, Auth. Token

Create API key and store API key and token elsewhere.



Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
qwerty123	Disconnected	iotdevice1	Device	May 16, 2020 7:03 PM	
pk12345	Disconnected	motor	Device	May	

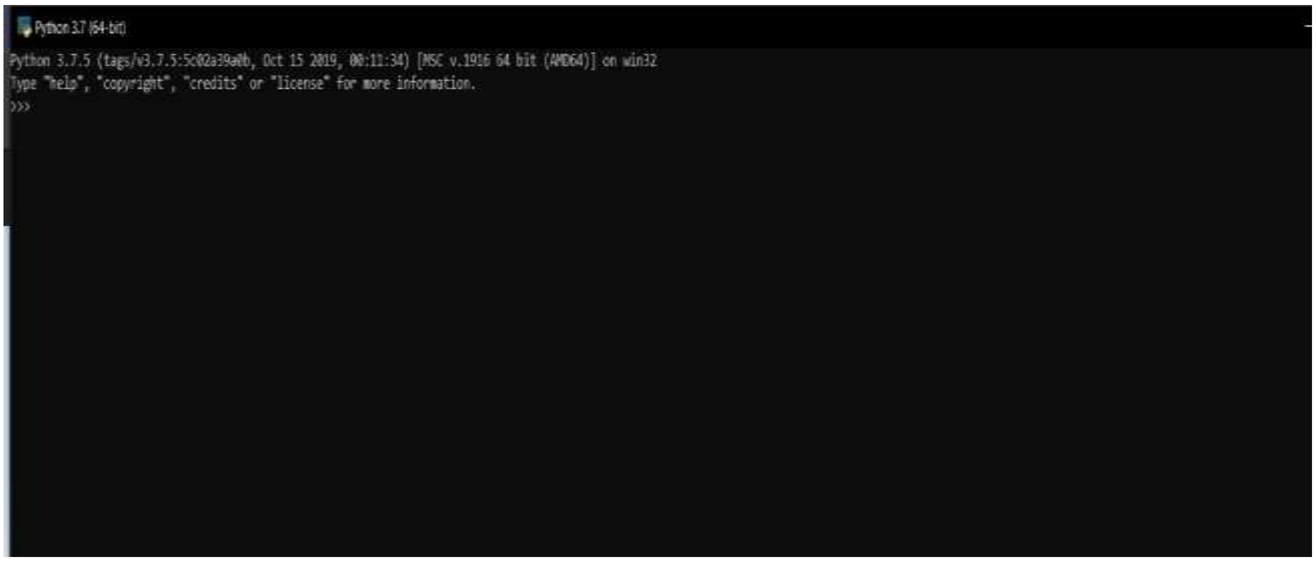
Device ID qwerty123
Device Type iotdevice1
Date Added May 16, 2020 7:03 PM
Added By edn17b022@iiitk.ac.in
Connection Status Disconnected
Last Connected: May 23, 2020 1:26 AM
Client Address: 223.238.42.248 SecureToken
Duration: 37 minutes
Data Transferred: 626 B

Items per page 50 | 1–2 of 2 items

Python IDE

Install Python3 compiler

Install any python IDE to execute python scripts, in my case I used Spyder to execute the code.



```
Python 3.7.5 (tags/v3.7.5:5c02a39a0b, Oct 15 2019, 00:11:34) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

IoT Simulator

In our project in the place of sensors we are going to use IoT sensor simulator which give randomreadings to the connected cloud.

The link to simulator: https://watson-iot-sensor-simulator.mybluemix.net/

We need to give the credentials of the created device in IBM Watson IoT Platform to connect cloud tosimulator.

OpenWeather API

OpenWeather Map is an online service that provides weather data.

It provides current weather data,forecasts and historical data to more than

2 million customer.

Website link: https://openweathermap.org/guide

Steps to configure:

Create account in OpenWeather

Find the name of your city by searching

Create API key to your account

Replace “city name” and “your api key” with your city and API key in below red text
`api.openweathermap.org/data/2.5/weather?q={city name}&appid={your api key}` Link I used in my project:

http://api.openweathermap.org/data/2.5/weather?q=Gudur,in&appid=62354068e45f41ffa6a5b164714145fe

Building Project

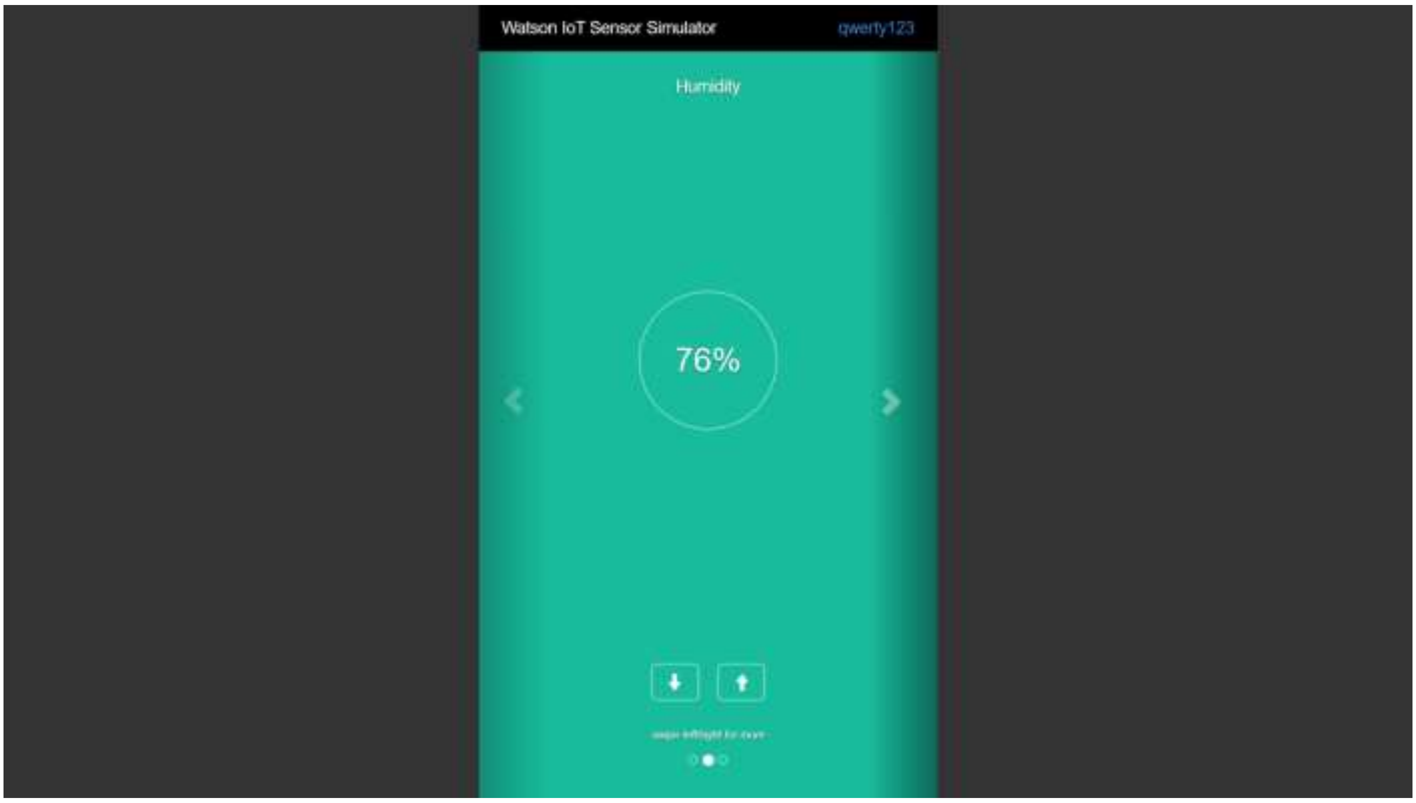
Connecting IoT Simulator to IBM Watson IoT Platform

Open link provided in above section 4.3

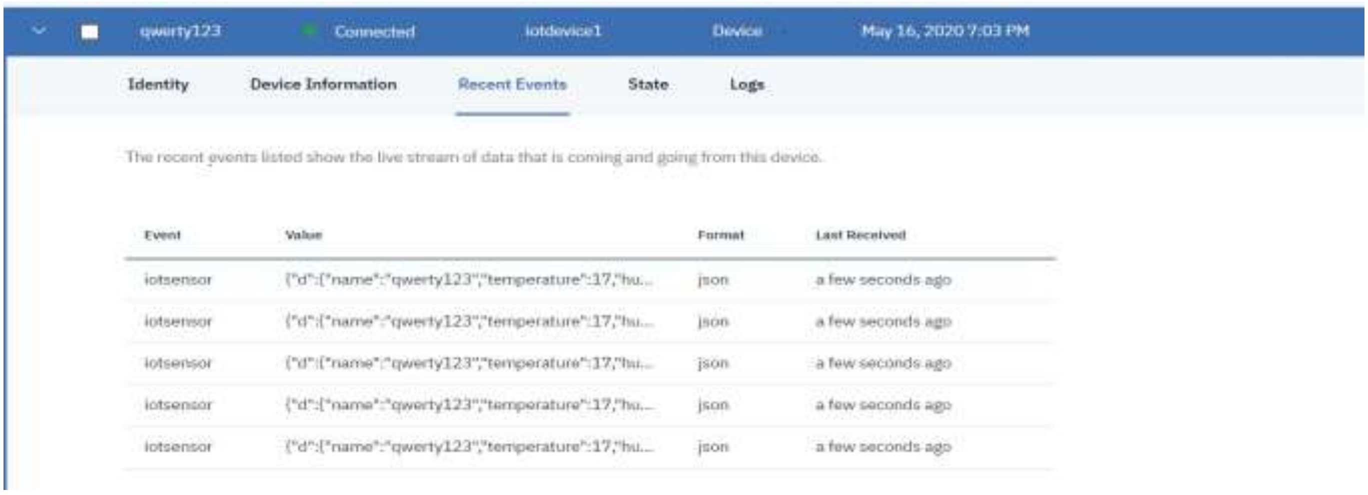
Give the credentials of your device in IBM Watson IoT Platform Click on connect

My credentials given to simulator are:

Org ID: 9wbx5	api: a-9wbx5m-1qfklrf7jl
Device type: iotdevice1	token: JcU(4(9Z37PdL!Rmz(Device ID : qwerty123
Device Token : johnyjohnyyespapa	



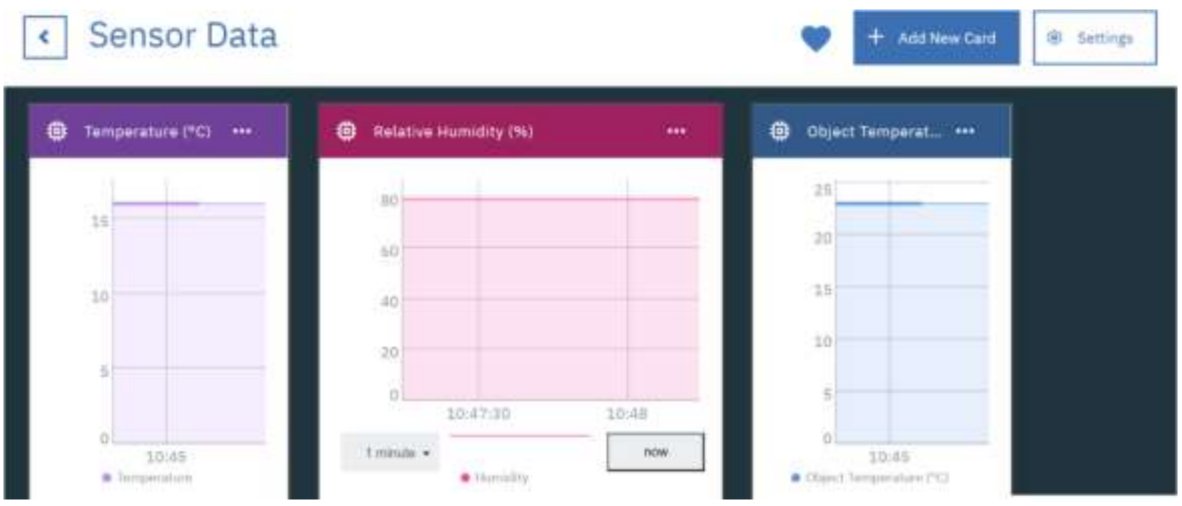
- You will receive the simulator data in cloud
- You can see the received data in Recent Events under your device
- Data received in this format(json)
- {
    - "d": {
        - "name": "qwerty123",
        - "temperature": 17,
        - "humidity": 76,
        - "objectTemp": 25
    - }
- }



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

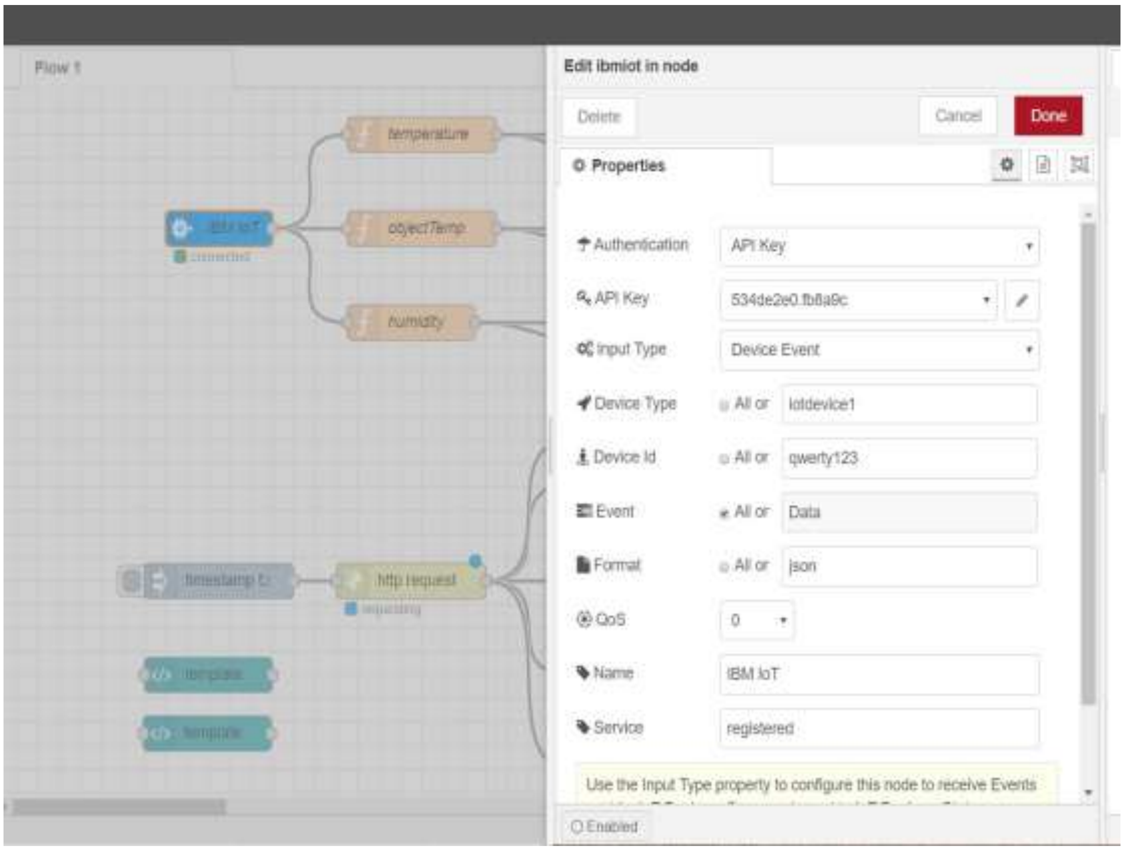
Event	Value	Format	Last Received
iotsensor	{"d":{"name":"qwerty123","temperature":17,"hu...	json	a few seconds ago
iotsensor	{"d":{"name":"qwerty123","temperature":17,"hu...	json	a few seconds ago
iotsensor	{"d":{"name":"qwerty123","temperature":17,"hu...	json	a few seconds ago
iotsensor	{"d":{"name":"qwerty123","temperature":17,"hu...	json	a few seconds ago
iotsensor	{"d":{"name":"qwerty123","temperature":17,"hu...	json	a few seconds ago

You can see the received data in graphs by creating cards in Boards tab



Configuration of Node-Red to collect IBM cloud data

The node IBM IoT App in is added to Node-Red workflow. Then the appropriate device credentials obtained earlier are entered into the node to connect and fetch device telemetry to Node-Red



Authentication: API Key
API Key: 534de2e0.fb8a9c
Input Type: Device Event
Device Type: All or iotdevice1
Device Id: All or qwerty123
Event: All or Data
Format: All or json
QoS: 0
Name: IBM IoT
Service: registered

Use the Input Type property to configure this node to receive Events

Once it is connected Node-Red receives data

from the device Display the data using debug

node for verification

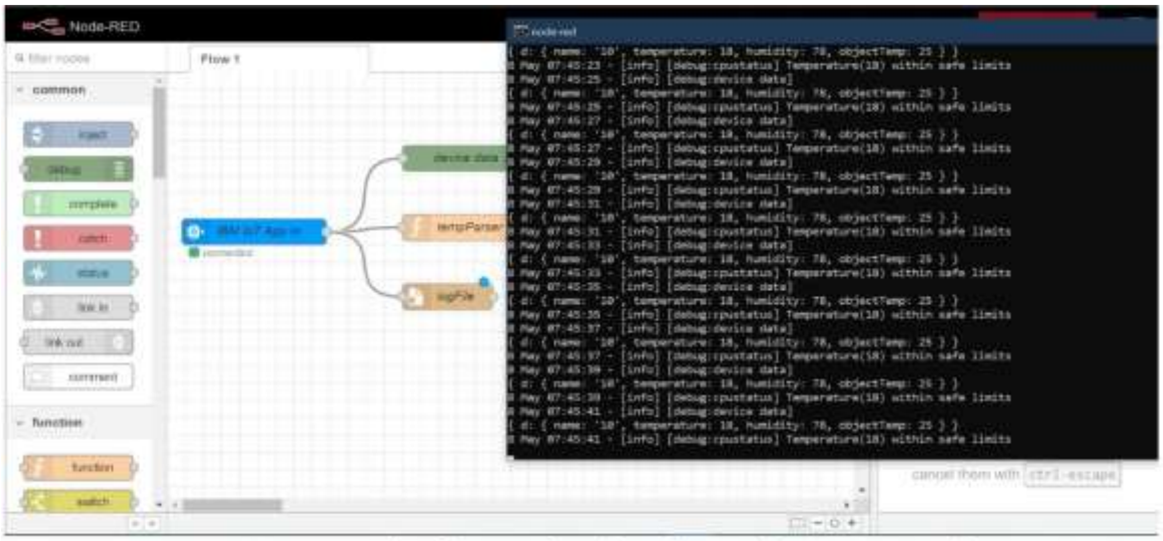
Connect function node and write the Java script code to get

each reading separately. The Java script code for the function

node is:

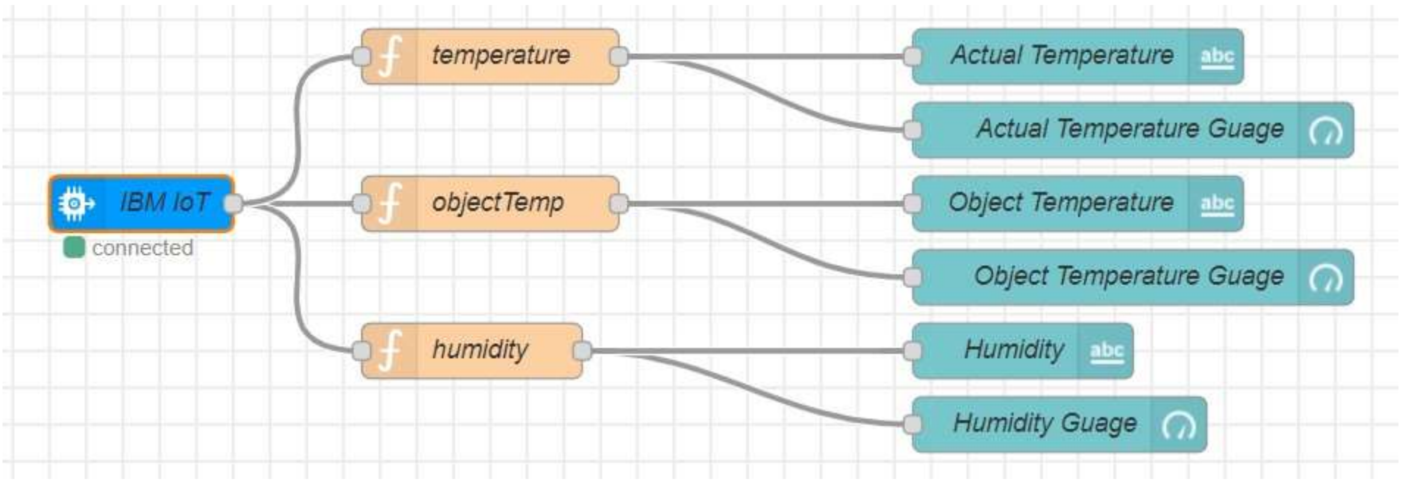
- msg.payload=msg.payload.d.te
- mperature return msg;

Finally connect Gauge nodes from dashboard to see the data in UI

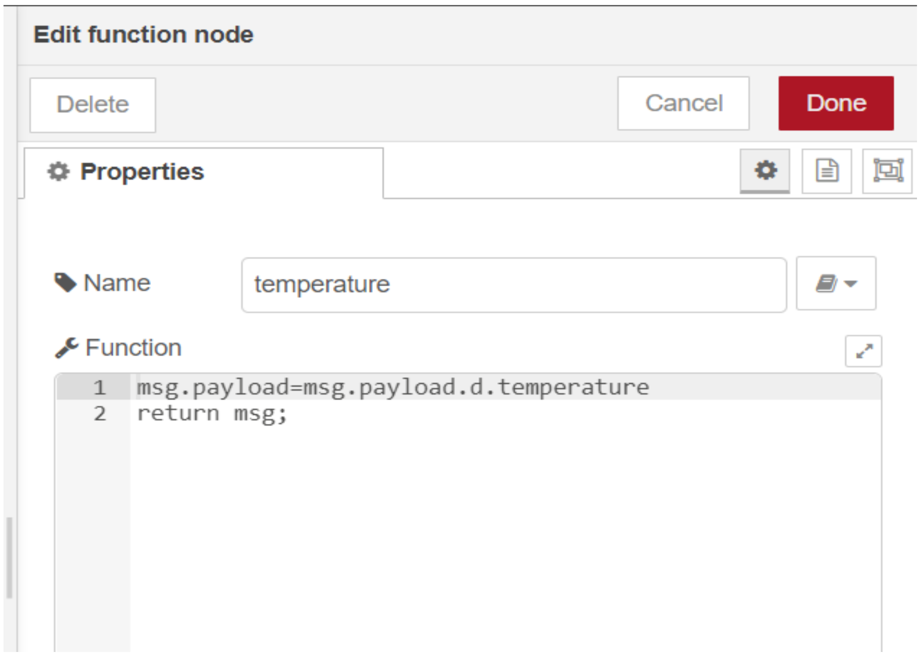


```
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
 May 07:45:23 - [info] [debug:cpustatus] Temperature(18) within safe limits
 May 07:45:25 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
 May 07:45:25 - [info] [debug:cpustatus] Temperature(18) within safe limits
 May 07:45:27 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
 May 07:45:27 - [info] [debug:cpustatus] Temperature(18) within safe limits
 May 07:45:29 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
 May 07:45:29 - [info] [debug:cpustatus] Temperature(18) within safe limits
 May 07:45:31 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
 May 07:45:31 - [info] [debug:cpustatus] Temperature(18) within safe limits
 May 07:45:33 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
 May 07:45:33 - [info] [debug:cpustatus] Temperature(18) within safe limits
 May 07:45:35 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
 May 07:45:35 - [info] [debug:cpustatus] Temperature(18) within safe limits
 May 07:45:37 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
 May 07:45:37 - [info] [debug:cpustatus] Temperature(18) within safe limits
 May 07:45:39 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
 May 07:45:39 - [info] [debug:cpustatus] Temperature(18) within safe limits
 May 07:45:41 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
 May 07:45:41 - [info] [debug:cpustatus] Temperature(18) within safe limits
```

Data received from the cloud in Node-RED console



Nodes connected in following manner to get each reading separately



Name: temperature

Function

```
msg.payload=msg.payload.d.temperature
return msg;
```

You can see the received data in graphs by creating cards in Boards tab

Configuration of Node-Red to collect data from OpenWeather

The Node-Red also receive data from the OpenWeather API by HTTP GET request. An inject trigger is addedto perform HTTP request for every certain interval.

HTTP request node is configured with URL we saved before in section 4.4

The data we receive from OpenWeather after request is in below JSON format:

{"coord":{"lon":79.85,"lat":14.13},"weather":[{"id":803,"main":"Clouds","description":"broken

clouds","icon":"04n"}],"base":"stations","main":{"temp":307.59,"feels_like":305.5,"temp_min":307.
59,"temp_max":307.59,"pressure":1002,"humidity":35,"sea_level":1002,"grnd_level":1000},"wind":{
"speed":6.23,"deg":170},"clouds":{"all":68},"dt":1589991979,"sys":{"country":"IN","sunrise":158993

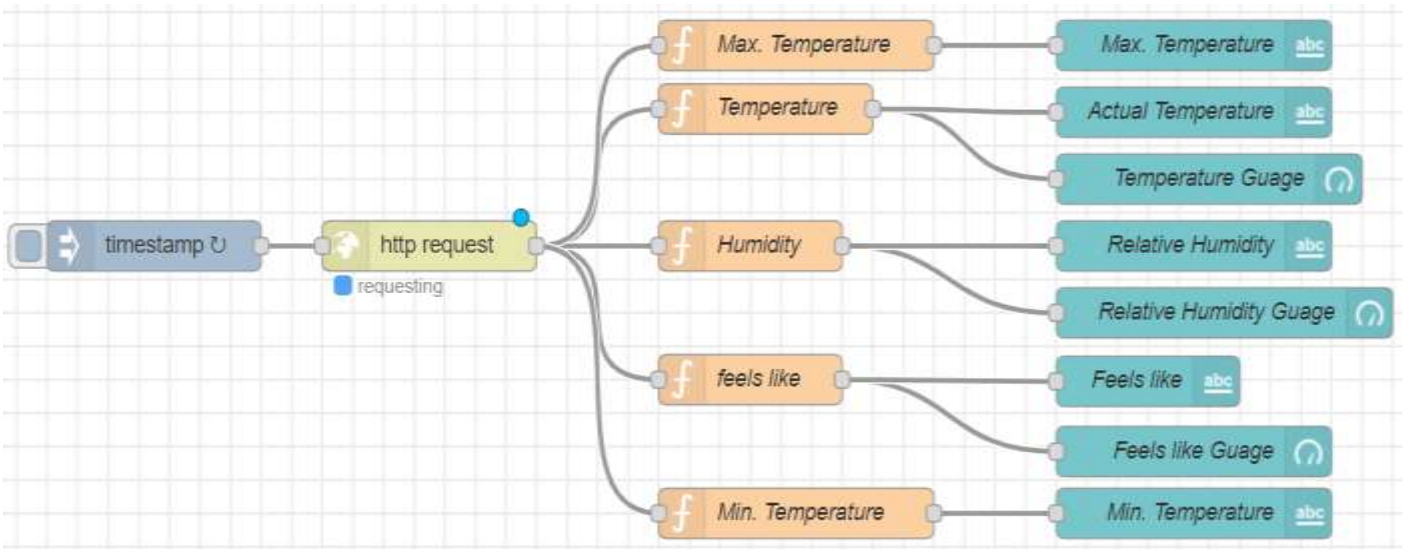
3553,"sunset":1589979720},"timezone":19800,"id":1270791,"name":"Gūdūr","cod":200}

In order to parse the JSON string we use Java script functions and get each parameters

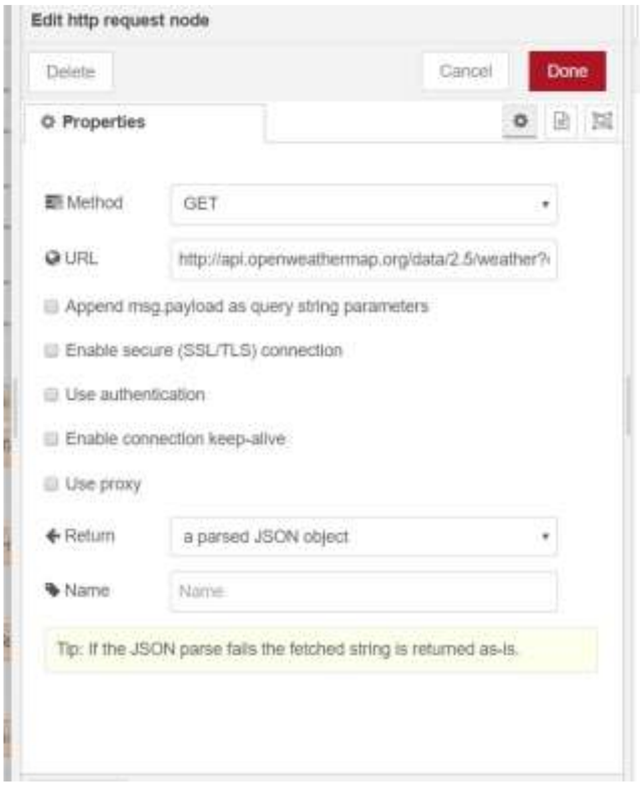
```
var temperature = msg.payload.main.temp;

temperature = temperature-273.15; return {payload : temperature.toFixed(2)};
```

In the above Java script code we take temperature parameter into a new variable and convert it from kelvin to Celsius

Then we add Gauge and text nodes to represent data visually in UI

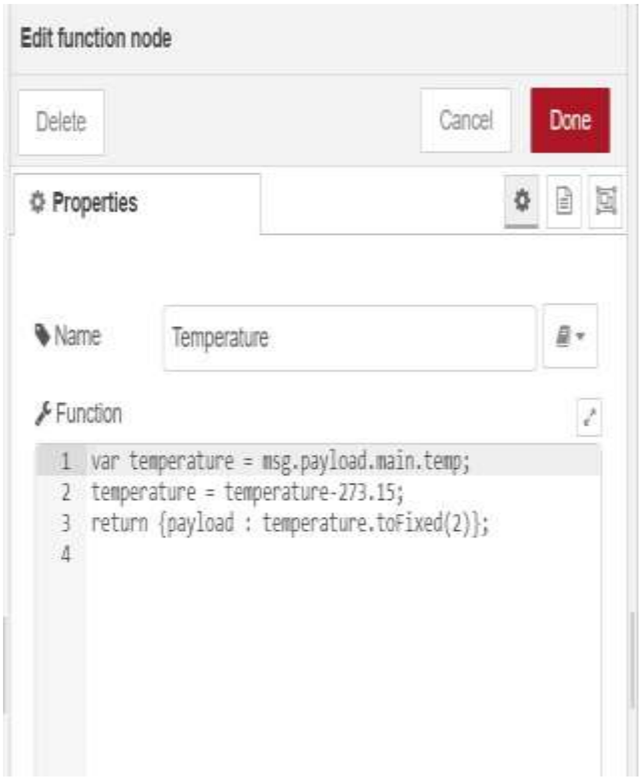


The above image has the program flow for receiving data from OpenWeather


Edit http request node

Method: GET
URL: http://api.openweathermap.org/data/2.5/weather?
Append msg.payload as query string parameters
Enable secure (SSL/TLS) connection
Use authentication
Enable connection keep-alive
Use proxy
Return: a parsed JSON object
Name: Name

Tip: If the JSON parse fails the fetched string is returned as-is.


Edit function node

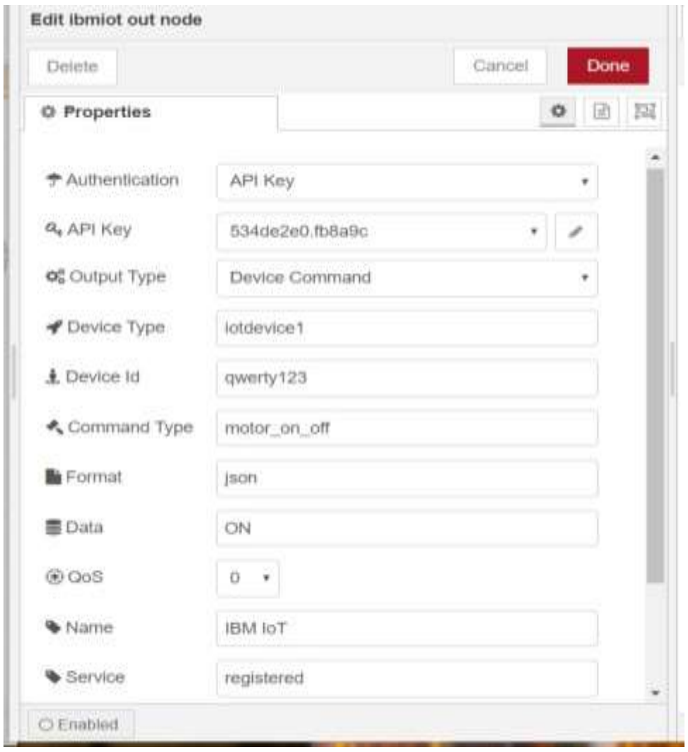
Name: Temperature
Function:

```
var temperature = msg.payload.main.temp;
temperature = temperature-273.15;
return {payload : temperature.toFixed(2)};
```

The above two images contain http request and function node data that needs to be filled.

Configuration of Node-Red to send commands to IBM cloud

ibmiot out node I used to send data from Node-Red to IBM Watson device. So, after adding it to the flow weneed to configure it with credentials of our Watson device.


Edit ibmiot out node

Authentication: API Key
API Key: 534de2e0.fb8a9c
Output Type: Device Command
Device Type: iotdevice1
Device Id: qwerty123
Command Type: motor_on_off
Format: json
Data: ON
QoS: 0
Name: IBM IoT
Service: registered

Here we add three buttons in UI which each sends a

number 0,1 and 2.0 -> for motor off

1 -> for motor on

2 -> for running motor continuously 30 minutes

We used a function node to analyse the data received and assign command to each number.

The Java script code for the analyser is:

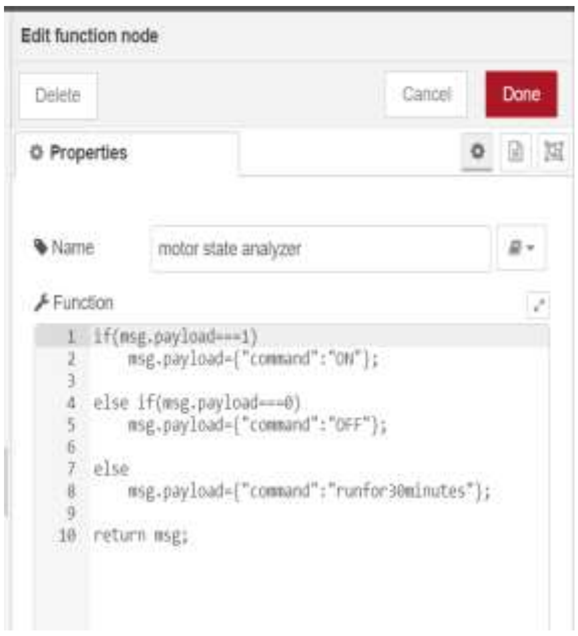
```
if(msg.payload===1)
 msg.payload={"command"
 :"ON"}; else
 if(msg.payload===0)
 msg.payload={"command"
 :"OFF"};else

 msg.payload={"command":"runfor30
minutes"};return msg;
```

Then we use another function node to parse the data and get the command and represent it visually with text node.

The Java script code for that function

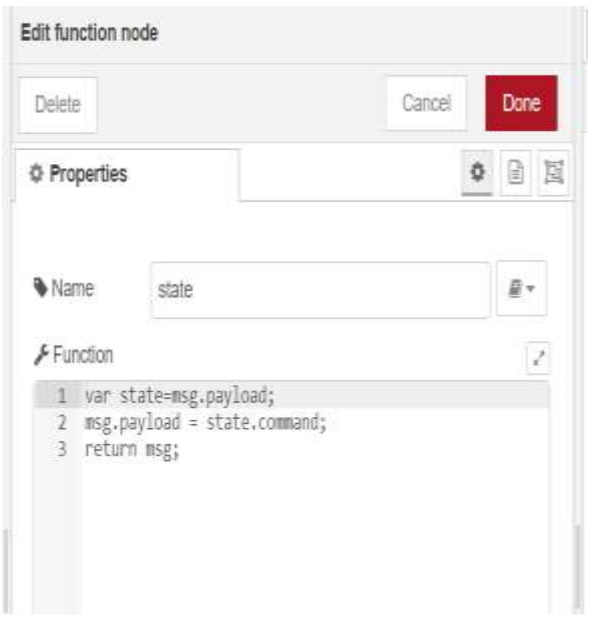
node is:

```
var state=msg.payload;

 msg.payload =
state.command;   return
msg;
```



Name: motor state analyzer

```
if(msg.payload===1)
    msg.payload={"command":"ON"};

else if(msg.payload===0)
    msg.payload={"command":"OFF"};

else
    msg.payload={"command":"runfor30minutes"};

return msg;
```

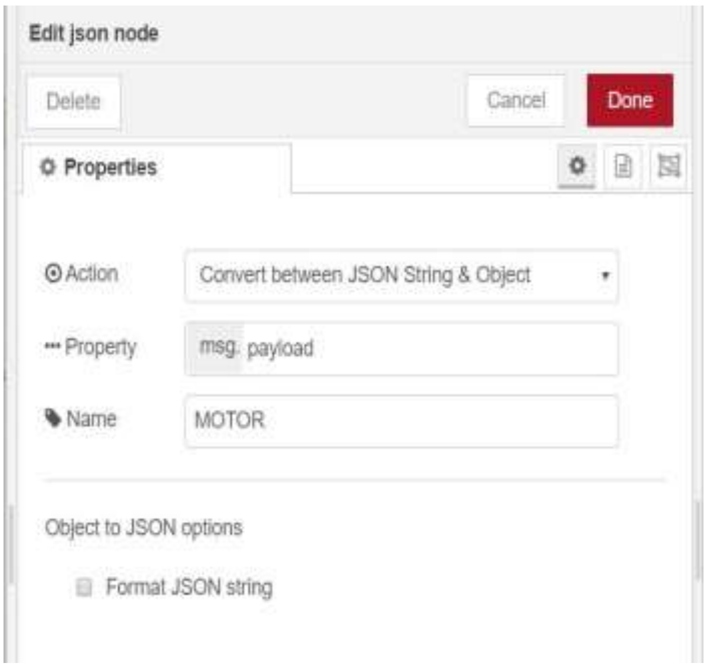


Name: state

```
var state=msg.payload;
msg.payload = state.command;
return msg;
```

The above images show the java script codes of analyser and state function nodes.

Then we add edit Json node to the conversion between JSON string & object and finally connect it to IBM IoTOut.



Action: Convert between JSON String & Object

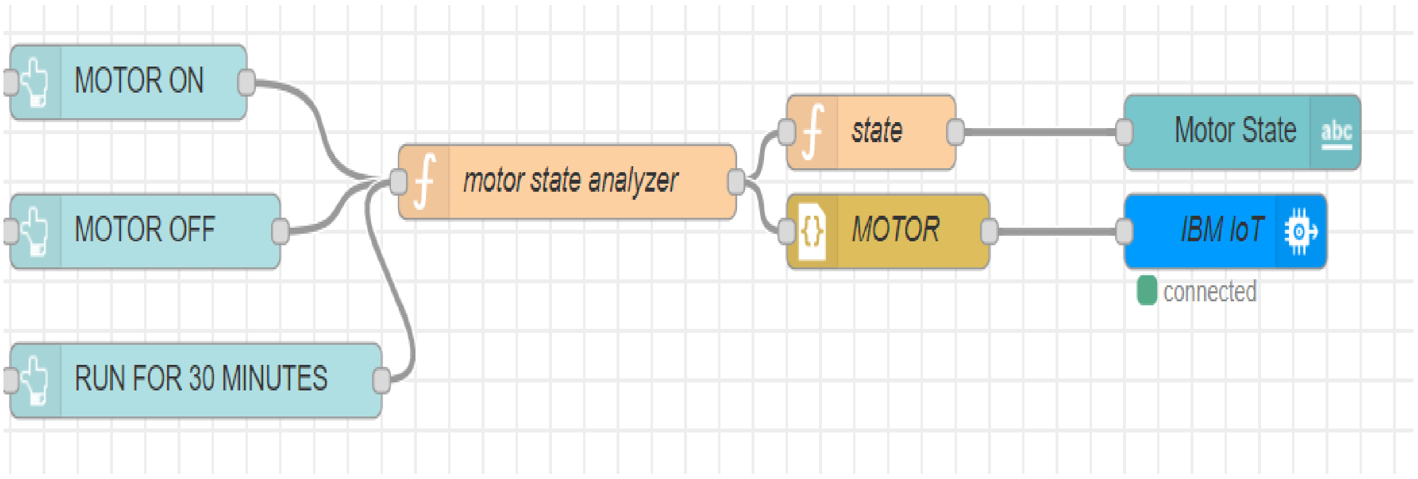
Property: msg. payload

Name: MOTOR

Object to JSON options

Format JSON string

Edit JSON node needs to be configured like



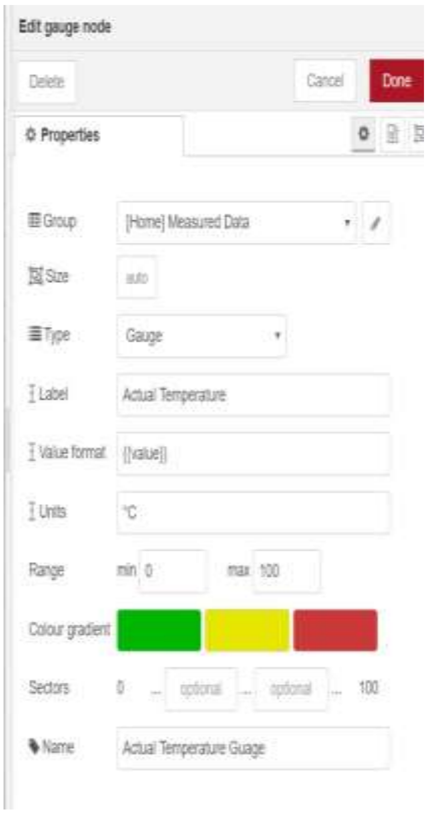
This is the program flow for sending commands to IBM cloud.

Adjusting User Interface

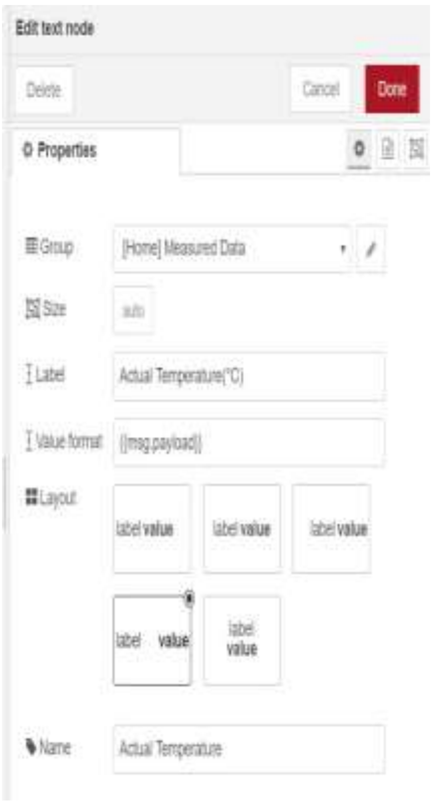
In order to display the parsed JSON data a Node-Red dashboard is created

Here we are using Gauges, text and button nodes to display in the UI and helps to monitor the parametersand control the farm equipment.

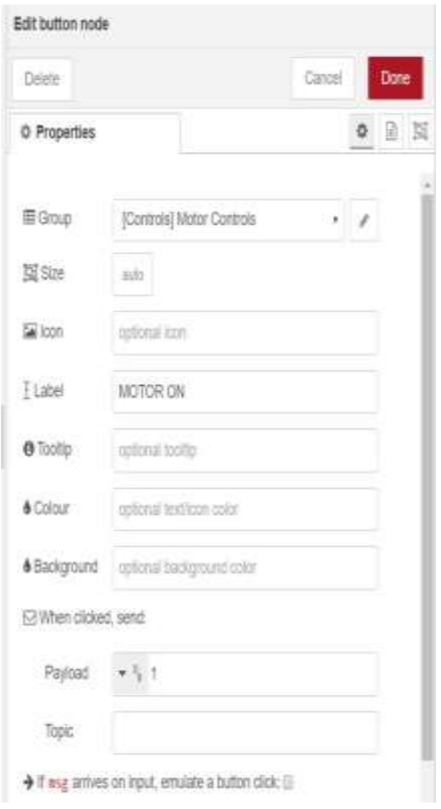
Below images are the Gauge, text and button node configurations



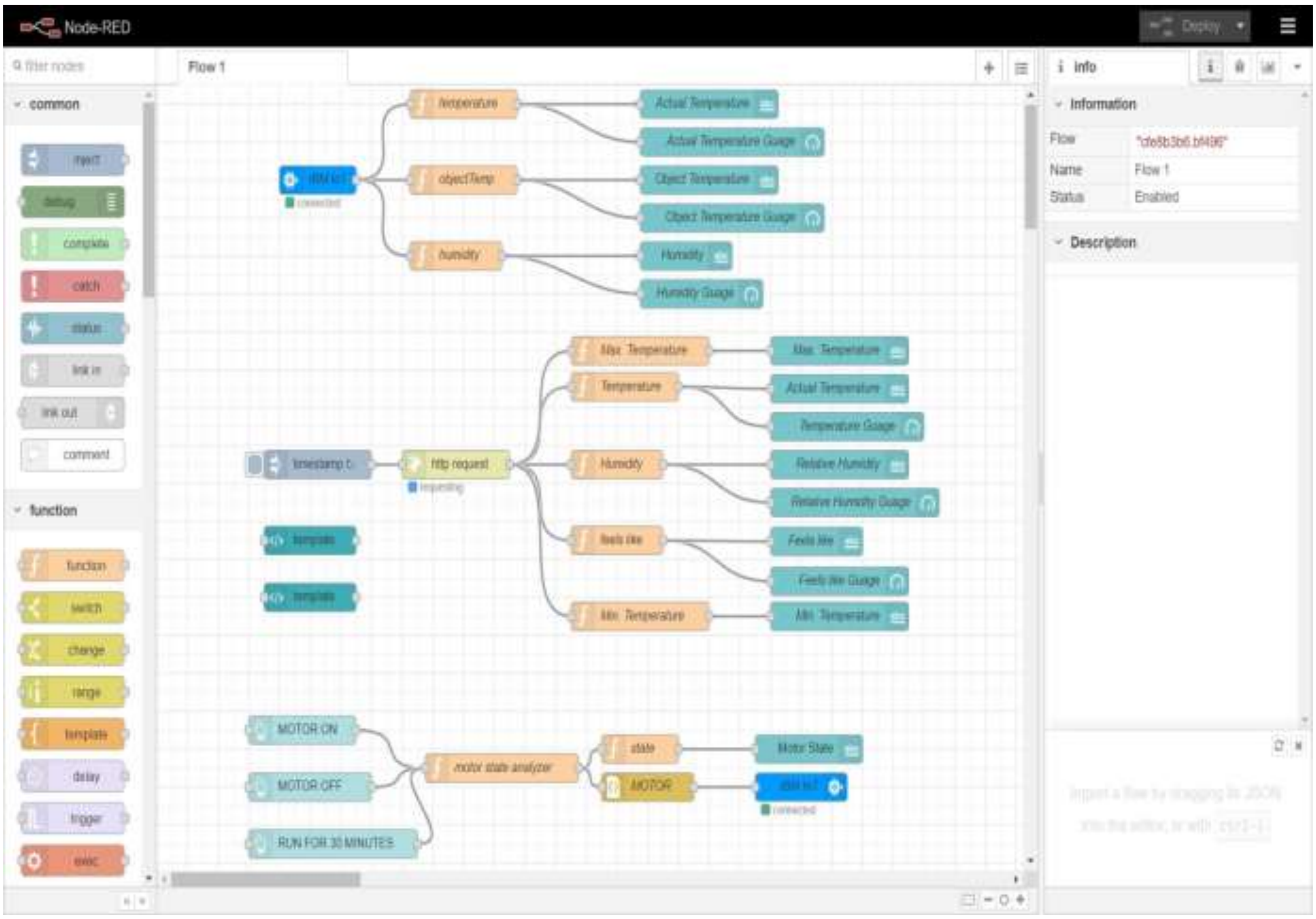
Group: [Home] Measured Data
Size: auto
Type: Gauge
Label: Actual Temperature
Value format: {{value}}
Units: °C
Range: min 0 max 100
Colour gradient: 🟩 🟨 🟥
Sectors: 0 ... optional ... optional ... 100
Name: Actual Temperature Guage



Group: [Home] Measured Data
Size: auto
Label: Actual Temperature(°C)
Value format: {{msg.payload}}
Layout
Name: Actual Temperature

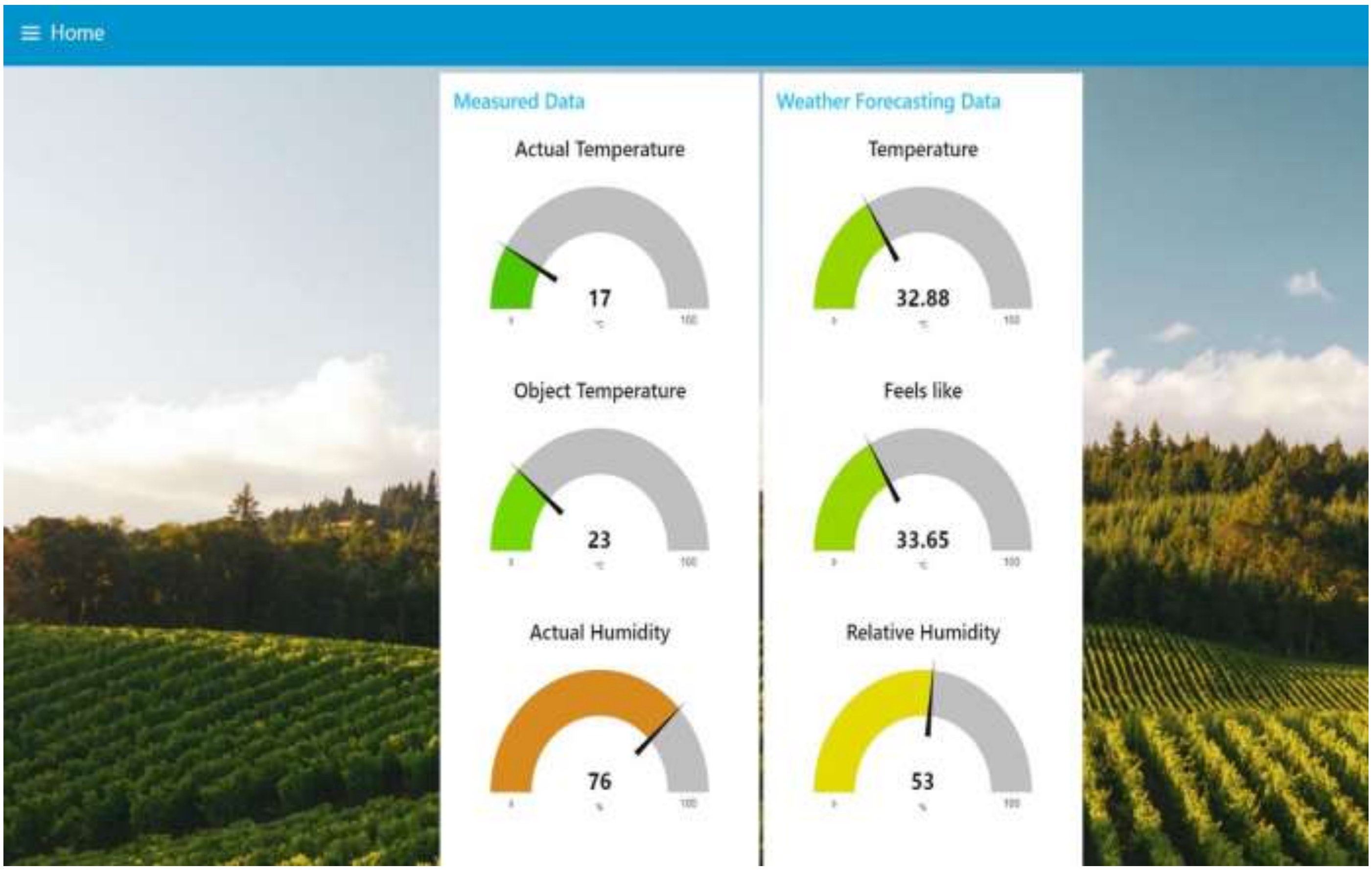


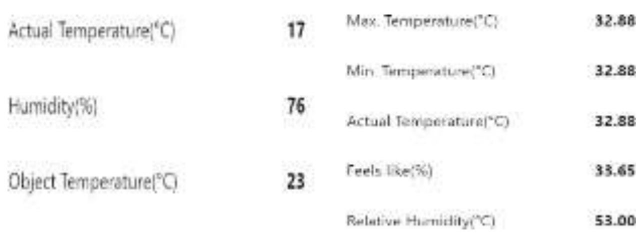
Group: [Controls] Motor Controls
Size: auto
Icon: optional icon
Label: MOTOR ON
Tooltip: optional tooltip
Colour: optional text/icon color
Background: optional background color
When clicked, send:
Payload: 1
Topic:
If msg arrives on input, emulate a button click:


Node-RED
Deploy
filter nodes
Flow 1
info
Information
Flow "cfe8b3b6.bf496"
Name Flow 1
Status Enabled
Description
common
inject
debug
complete
catch
status
link in
link out
comment
function
function
switch
change
range
template
delay
trigger
exec
IBM IoT
connected
temperature
objectTemp
humidity
Actual Temperature
Actual Temperature Guage
Object Temperature
Object Temperature Guage
Humidity
Humidity Guage
timestamp
http request
requesting
template
template
Max. Temperature
Temperature
Humidity
feels like
Min. Temperature
Max. Temperature
Actual Temperature
Temperature Guage
Relative Humidity
Relative Humidity Guage
Feels like
Feels like Guage
Min. Temperature
MOTOR ON
MOTOR OFF
RUN FOR 30 MINUTES
motor state analyzer
state
MOTOR
Motor State
IBM IoT
connected
Import a flow by dragging its JSON into the editor, or with ctrl-i

Web APP UI

Home Tab





Actual Temperature(°C)	17
Humidity(%)	76
Object Temperature(°C)	23

Max. Temperature(°C)	32.88
Min. Temperature(°C)	32.88
Actual Temperature(°C)	32.88
Feels like(%)	33.65
Relative Humidity(°C)	53.00

8.TESTING

Download the testset and trainset images from existing libraries which are available as Github repositories with each class consisting of 5000 images.

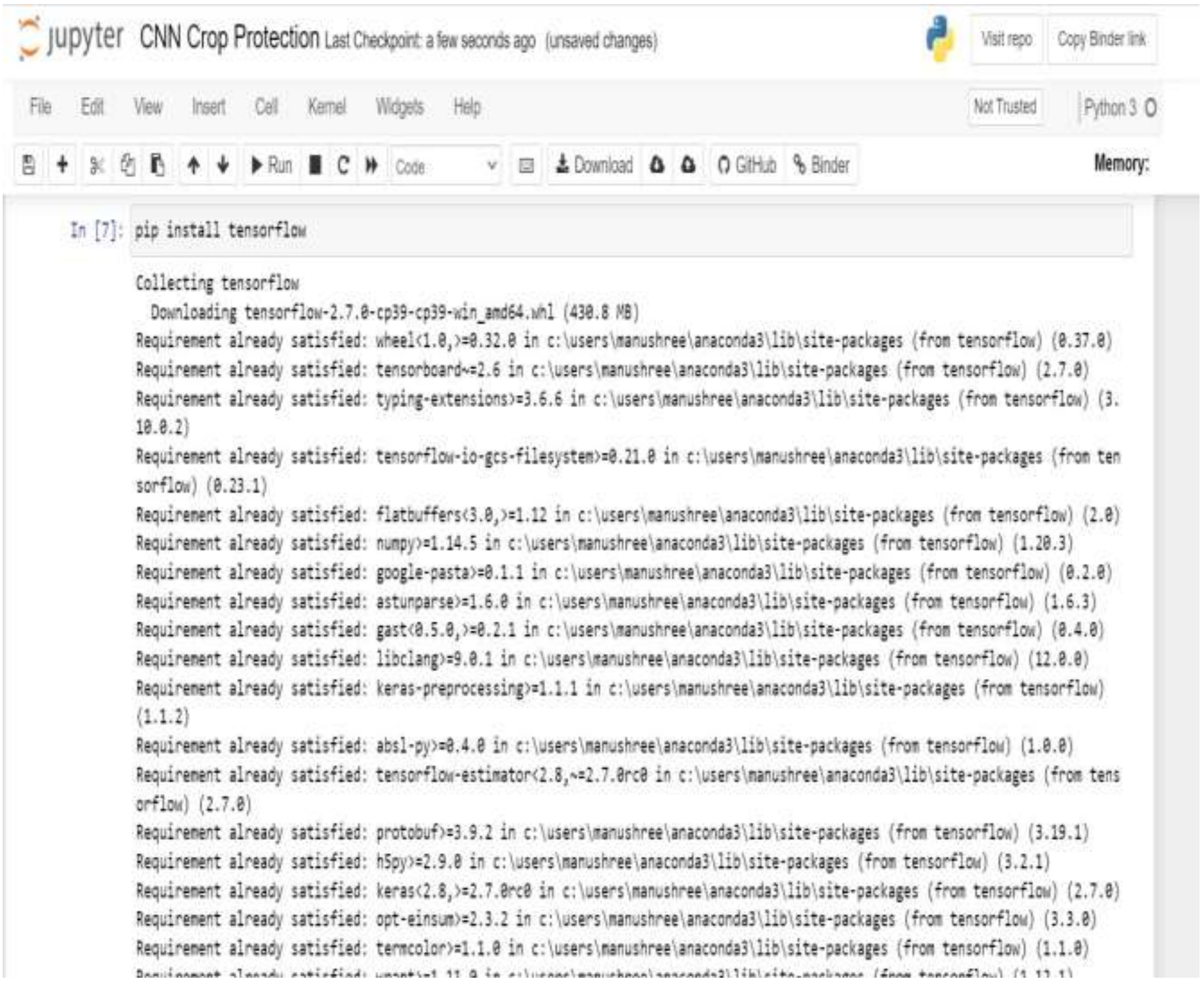
Class A: Bear

Class B: Elephant

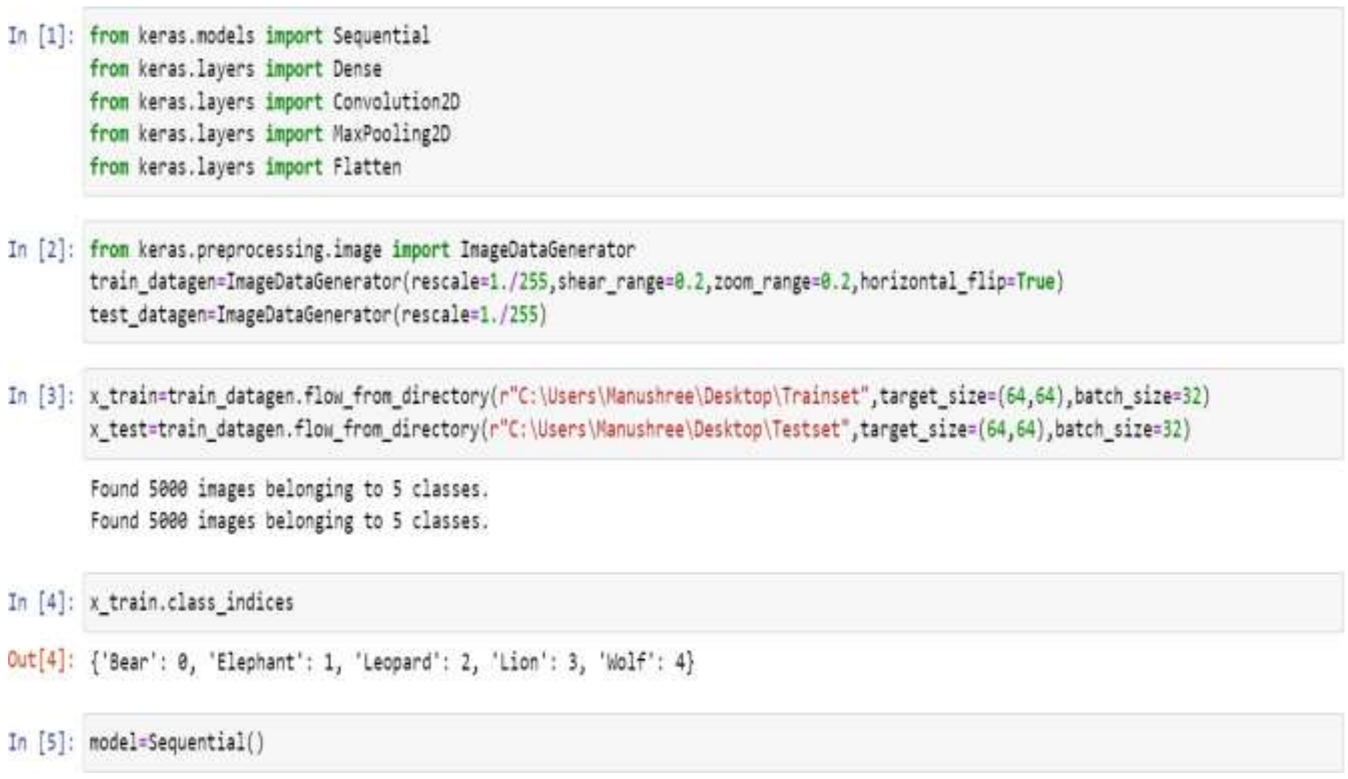
Class C: Leopard

Class D: Lion

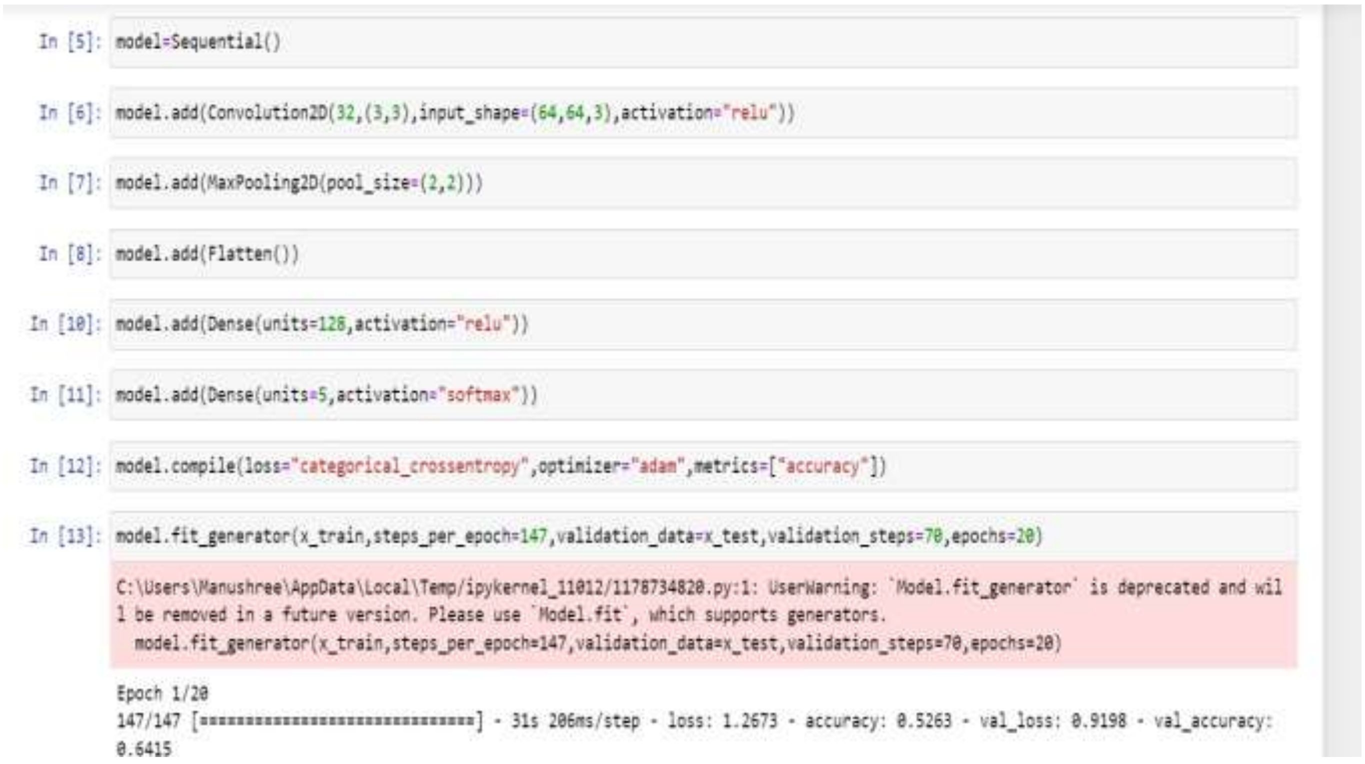
Class E: Wolf


jupyter CNN Crop Protection Last Checkpoint: a few seconds ago (unsaved changes)
Visit repo
Copy Binder link
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
Download
GitHub
Binder
Memory:

```
In [7]: pip install tensorflow

        Collecting tensorflow
          Downloading tensorflow-2.7.0-cp39-cp39-win_amd64.whl (430.8 MB)
        Requirement already satisfied: wheel<1.0,>=0.32.0 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (0.37.0)
        Requirement already satisfied: tensorboard~=2.6 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (2.7.0)
        Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (3.
        10.0.2)
        Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.21.0 in c:\users\manushree\anaconda3\lib\site-packages (from ten
        sorflow) (0.23.1)
        Requirement already satisfied: flatbuffers<3.0,>=1.12 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (2.0)
        Requirement already satisfied: numpy>=1.14.5 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (1.20.3)
        Requirement already satisfied: google-pasta>=0.1.1 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (0.2.0)
        Requirement already satisfied: astunparse>=1.6.0 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (1.6.3)
        Requirement already satisfied: gast<0.5.0,>=0.2.1 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (0.4.0)
        Requirement already satisfied: libclang>=9.0.1 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (12.0.0)
        Requirement already satisfied: keras-preprocessing>=1.1.1 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow)
        (1.1.2)
        Requirement already satisfied: absl-py>=0.4.0 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (1.0.0)
        Requirement already satisfied: tensorflow-estimator<2.8,~=2.7.0rc0 in c:\users\manushree\anaconda3\lib\site-packages (from tens
        orflow) (2.7.0)
        Requirement already satisfied: protobuf>=3.9.2 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (3.19.1)
        Requirement already satisfied: h5py>=2.9.0 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (3.2.1)
        Requirement already satisfied: keras<2.8,>=2.7.0rc0 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (2.7.0)
        Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (3.3.0)
        Requirement already satisfied: termcolor>=1.1.0 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (1.1.0)
        Requirement already satisfied: wrapt>=1.11.0 in c:\users\manushree\anaconda3\lib\site-packages (from tensorflow) (1.12.1)
```



```
In [1]: from keras.models import Sequential
        from keras.layers import Dense
        from keras.layers import Convolution2D
        from keras.layers import MaxPooling2D
        from keras.layers import Flatten

In [2]: from keras.preprocessing.image import ImageDataGenerator
        train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
        test_datagen=ImageDataGenerator(rescale=1./255)

In [3]: x_train=train_datagen.flow_from_directory(r"C:\Users\Manushree\Desktop\Trainset",target_size=(64,64),batch_size=32)
        x_test=train_datagen.flow_from_directory(r"C:\Users\Manushree\Desktop\Testset",target_size=(64,64),batch_size=32)

        Found 5000 images belonging to 5 classes.
        Found 5000 images belonging to 5 classes.

In [4]: x_train.class_indices

Out[4]: {'Bear': 0, 'Elephant': 1, 'Leopard': 2, 'Lion': 3, 'Wolf': 4}

In [5]: model=Sequential()
```

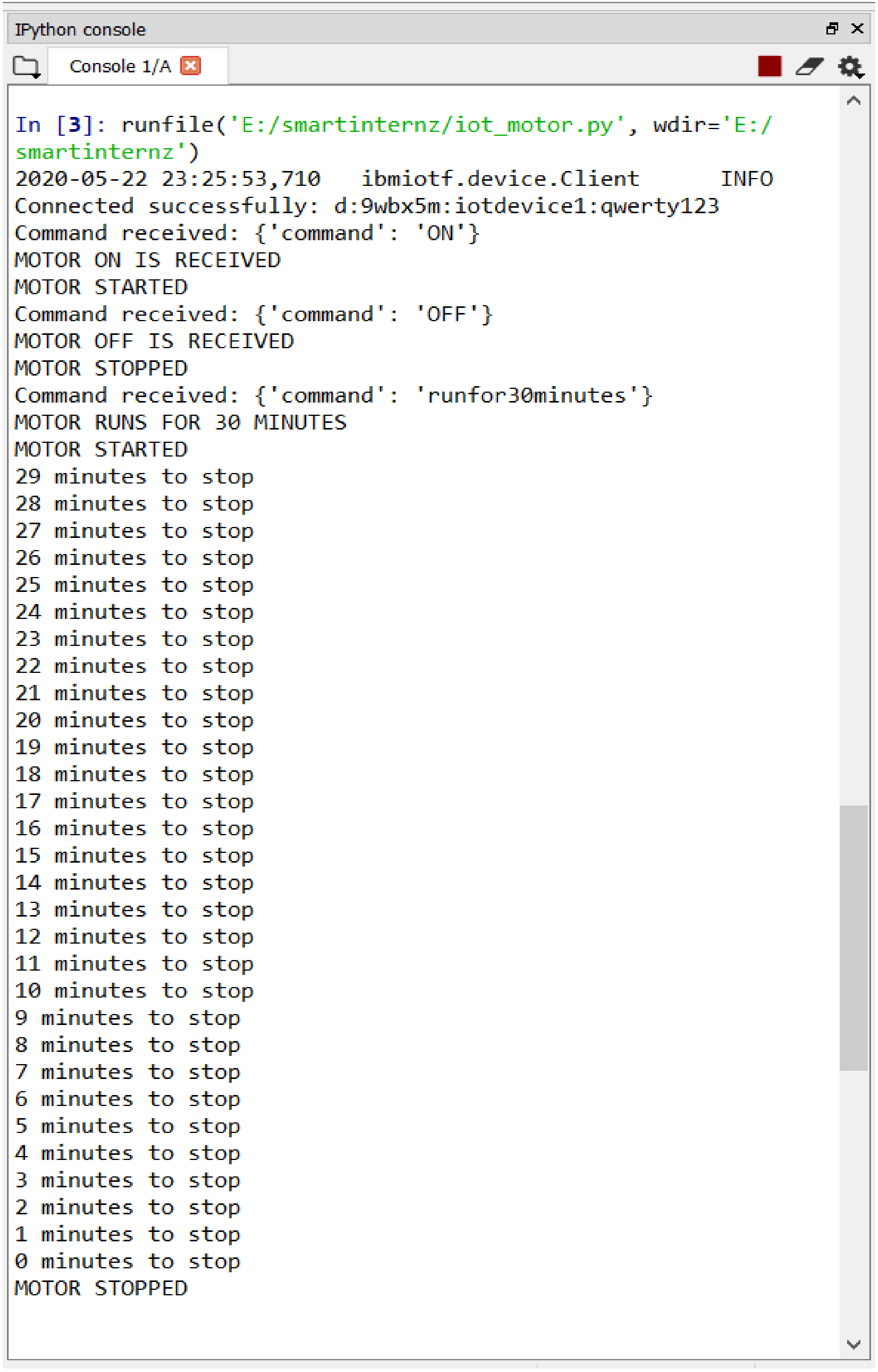


```
In [5]: model=Sequential()

In [6]: model.add(Convolution2D(32,(3,3),input_shape=(64,64,3),activation="relu"))

In [7]: model.add(MaxPooling2D(pool_size=(2,2)))

In [8]: model.add(Flatten())

In [10]: model.add(Dense(units=128,activation="relu"))

In [11]: model.add(Dense(units=5,activation="softmax"))

In [12]: model.compile(loss="categorical_crossentropy",optimizer="adam",metrics=["accuracy"])

In [13]: model.fit_generator(x_train,steps_per_epoch=147,validation_data=x_test,validation_steps=70,epochs=20)

C:\Users\Manushree\AppData\Local\Temp/ipykernel_11012/1178734820.py:1: UserWarning: `Model.fit_generator` is deprecated and wil
l be removed in a future version. Please use `Model.fit`, which supports generators.
  model.fit_generator(x_train,steps_per_epoch=147,validation_data=x_test,validation_steps=70,epochs=20)

Epoch 1/20
147/147 [==============================] - 31s 206ms/step - loss: 1.2673 - accuracy: 0.5263 - val_loss: 0.9198 - val_accuracy:
0.6415
```

Testing the model against the target image:

The target image used here was lion. The path of the image is given in the model and it is predicted.

9.RESULT



```
In [3]: runfile('E:/smartinternz/iot_motor.py', wdir='E:/
smartinternz')
2020-05-22 23:25:53,710   ibmiotf.device.Client      INFO
Connected successfully: d:9wbx5m:iotdevice1:qwerty123
Command received: {'command': 'ON'}
MOTOR ON IS RECEIVED
MOTOR STARTED
Command received: {'command': 'OFF'}
MOTOR OFF IS RECEIVED
MOTOR STOPPED
Command received: {'command': 'runfor30minutes'}
MOTOR RUNS FOR 30 MINUTES
MOTOR STARTED
29 minutes to stop
28 minutes to stop
27 minutes to stop
26 minutes to stop
25 minutes to stop
24 minutes to stop
23 minutes to stop
22 minutes to stop
21 minutes to stop
20 minutes to stop
19 minutes to stop
18 minutes to stop
17 minutes to stop
16 minutes to stop
15 minutes to stop
14 minutes to stop
13 minutes to stop
12 minutes to stop
11 minutes to stop
10 minutes to stop
9 minutes to stop
8 minutes to stop
7 minutes to stop
6 minutes to stop
5 minutes to stop
4 minutes to stop
3 minutes to stop
2 minutes to stop
1 minutes to stop
0 minutes to stop
MOTOR STOPPED
```

10.ADVANTAGES & DISADVANTAGES

Advantages:

- Farms can be monitored and controlled remotely.
- Increase in convenience to farmers.
- Less labour cost.
- Better standards of living.

Disadvantages:

- Lack of internet/connectivity issues.
- Added cost of internet and internet gateway infrastructure.
- Farmers wanted to adapt the use of WebApp.

11.CONCLUSION

Crop vandalism by wild animals and fire has evolved into a serious societal issue in recent years. It demands immediate attention because there is currently no viable answer to this problem. As a result, this initiative has a lot of societal significance because it tries to solve a problem. This initiative will assist farmers in securing their orchards and fields, preventing them from incurring large financial losses and saving them from the time and effort required to preserve their properties. This will also assist them in generating higher agricultural yields, which would improve their financial situation. By recognising the picture, an ultrasonic sensor may be triggered to drive the animal away. This approach is non-lethal to animals and does not waste powerThus the objective of the project to implement an IoT system in order to help farmers to control and monitor their farms has been implemented successfully.

12.APPENDIX

SOURCE CODE

```
import
timeimport
sys
import
ibmiotf.applicationimport
ibmiotf.device
organization = "9wbx5m"

deviceType =
"iotdevice1"deviceId =
"qwerty123" authMethod
= "token"

authToken =
"johnyjohnyyespapa"def
myCommandCallback(cmd):

    print("Command received: %s" %
    cmd.data)if cmd.data['command']=='ON':
    print("MOTOR ON IS RECEIVED")

    time.sleep(1)
    print("MOTOR
    STARTED")

    elif cmd.data['command']=='OFF':

   print("MOTOR OFF IS RECEIVED")

   time.sleep(1)

   print("MOTOR STOPPED")

   elif cmd.data['command']=='runfor30minutes':

   print("MOTOR RUNS FOR 30 MINUTES")
```

```
            print("MOTOR STARTED")
            for i in range(1,31):
            print("%d minutes to stop"%(30-i)) # use time.sleep(60) for delay of one minute
            print("MOTOR STOPPED")
   try:
            deviceOptions = {"org": organization, "type": deviceType, "id": deviceId,
                                                                                "auth-   method":
            authMethod, "auth-token": authToken}

           deviceCli = ibmiotf.device.Client(deviceOptions)
 except  Exception as e:
           print("Caught exception connecting device: %s" % str(e))sys.exit()
 deviceCli.connect()
 while True:
           deviceCli.commandCallback = myCommandCallback
 deviceCli.disconnect()
 from keras.models impoSequential
from keras.layers import Dense
 from keras.layers import Convolution2D
 from keras.layers import MaxPooling2D from
 keras.layers import Flatten
 from keras.preprocessing.image import ImageDataGenerator
 train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_
 flip=True) test_datagen=ImageDataGenerator(rescale=1./255)
 x_train=train_datagen.flow_from_directory(r"C:\Users\Manushree\Desktop\NewDataset\TrainSet",target_size
 =(64,64),batch_size=32)
 x_test=train_datagen.flow_from_directory(r"C:\Users\Manushree\Desktop\NewDataset\TestSet",
 target_size=( 64,64),batch_size=32)
 x_train.class_indices model=Sequential()
 model.add(Convolution2D(32,(3,3),input_shape=(64,64,3),activation="relu"))
 model.add(MaxPooling2D(pool_size=(2,2)))
 model.add(Flatten())
 model.add(Dense(units=128,init="uniform",activation="relu"))
 model.add(Dense(units=5,init="uniform",activation="softmax"))
 model.compile(loss="categorical_crossentropy",optimizer="adam",metrics=["accuracy"])
 model.fit_generator(x_train,steps_per_epoch=147,validation_data=x_test,validation_steps=70,epochs=20)
 from keras.models import load_model

 from keras.preprocessing import image

 img = image.load_img(r"C:\Users\Manushree\Desktop\chk imgs\.jpg",target_size = (64,64))
 import numpy as np

 x = image.img_to_array(img)

 x = np.expand_dims(x,axis = 0)
 x.shape
 ypred=model.predict_classes(x)

 index=['Bear','Bison','Deer','Elephant','Zebra'] print(index[ypred[0]])

 import cv2

 p = cv2.imread(r"C:\Users\Manushree\Desktop\chk imgs\d4.jpg")
 cv2.imshow("image",p)

 cv2.waitKey(0) cv2.destroyAllWindows()
 model.save("model.h5")
```

GITHUB & PROJECT DEMO LINK

https://github.com/IBM-EPBL/IBM-Project-51889-1660986419.git