**PROJECT REPORT**

1. INTRODUCTION

1.1 OVERVIEW

The objectives of this report is to proposed IOT based Smart Farming System which will enable farmers to have live data of soil moisture, environment temperature and humidity at very low cost so that live monitoring can be done.

1.2 PURPOSE

* To provide a service using which farmers can monitor the temperature, humidity and soil moisture parameters along with weather forecasting details so that they can grow and yield a good crop.
* Based on all the parameters he can water his crop by controlling the motors using the mobile application.
* Even if the farmer is not present near his crop he can water his crop by controlling the motors using the mobile application from anywhere.

2. LITERATURE SURVEY

2.1 Existing problem

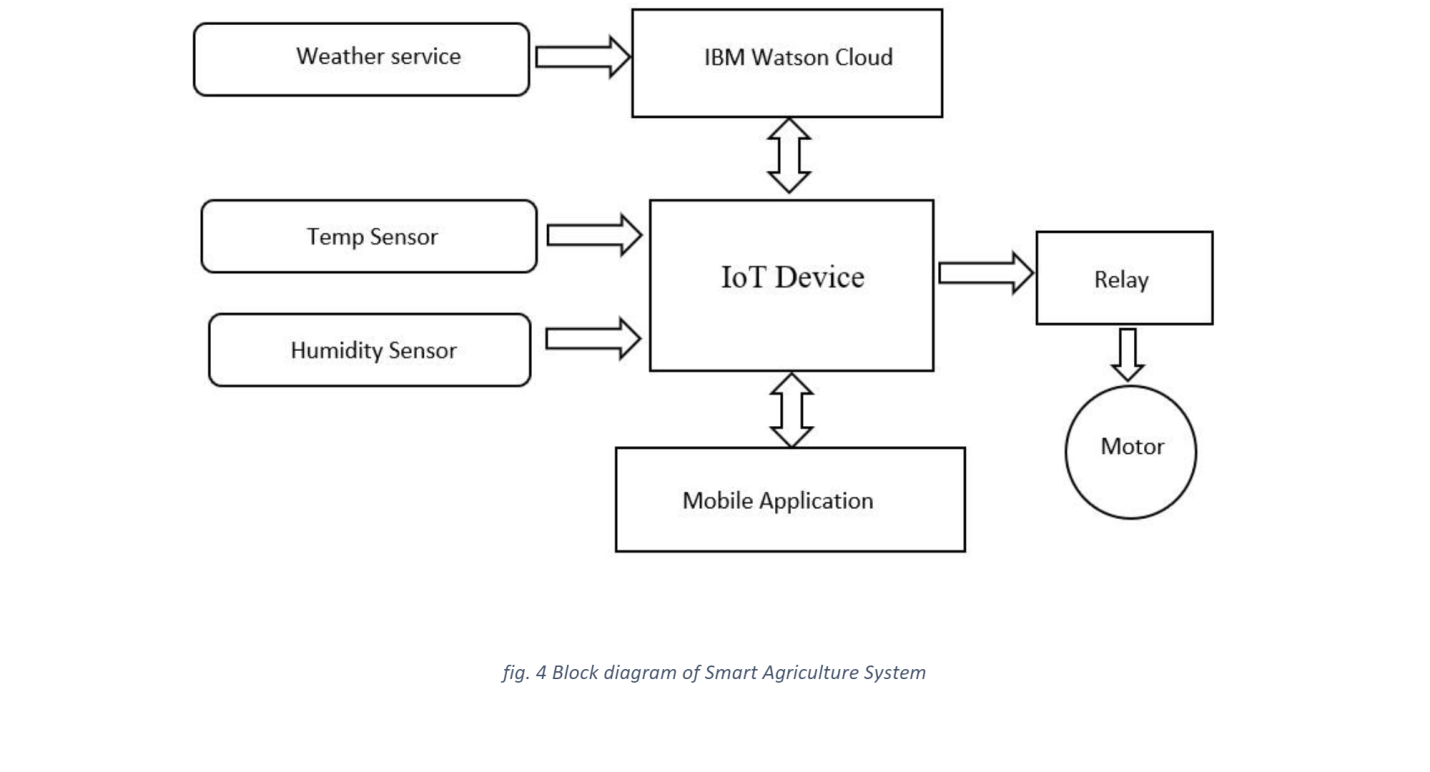
* Farmers cannot control their equipment/motor on off buttons remotely and have to go to the farm to operate and monitor the equipment.
* The farmers have stay at their farms to take care of their crops and they are unable to get information about the crops remotely.
* They do not have scientific values about the requirement of the crops like how much water is needed by the crop. Excess usage of water leads to wastage of water and less usage of water can affect the growth of the crops

2.2 Proposed solution

* Smart Agriculture System based on IoT can monitor soil moisture and climatic conditions to grow and yield a good crop.
* The farmer can also get the realtime weather forecasting data by using external platforms like Open Weather API.
* Farmer is provided a mobile app using which he can monitor the temperature, humidity and soil moisture parameters along with weather forecasting details.
* Based on all the parameters he can water his crop by controlling the motors using the mobile application.
* Even if the farmer is not present near his crop he can water his crop by controlling the motors using the mobile application from anywhere.
* Here we are using the Online IoT simulator for getting the Temperature,Humidity and Soil Moisture values.

3. THEORITICAL ANALYSIS

3.1 Block diagram



3.2 Hardware / Software designing: We use different softwares for designing

1. **IBM Cloud Platform:** To use services like IBM Watson IOT Platform

2. **IBM Watson IOT Platform:** To create an IOT device and to use IBM Watson Sensor Simulator

3. **IBM Watson Sensor Simulator:**  To receive data as it act as a virtual IOT device 

4. **Node-Red**: To create node-red dashboard (web app / UI) and to receive data from Open Weather API

5. **Open Weather API:** To receive weather related data of a particular location

4. EXPERIMENTAL INVESTIGATIONS

First task to build the web app is to get the sensor data in the cloud. I am using IBM cloud for this. We need an IBM account for the same. Steps to take sensor data in the cloud:

1. Sign up for IBM Academic Initiative Account using this [link](https://my15.digitalexperience.ibm.com/b73a5759-c6a6-4033-ab6b-d9d4f9a6d65b/dxsites/151914d1-03d2-48fe-97d9-d21166848e65/home/).
2. After this Sign up for IBM Cloud using [link](https://cloud.ibm.com/login).
3. Go to IBM Watson IOT Platform by searching IOT platform in the catalogue in IBM Cloud as below

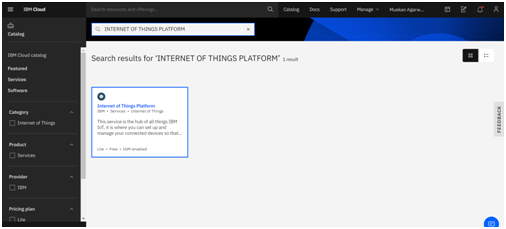
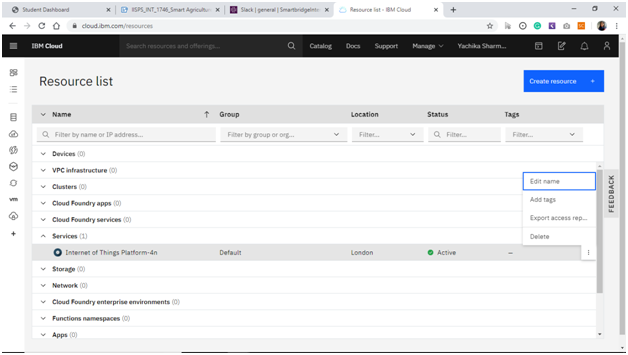


Figure 1: Internet of Things Platform service on IBM Cloud

Go to the IOT Platform and now we will create a device here.

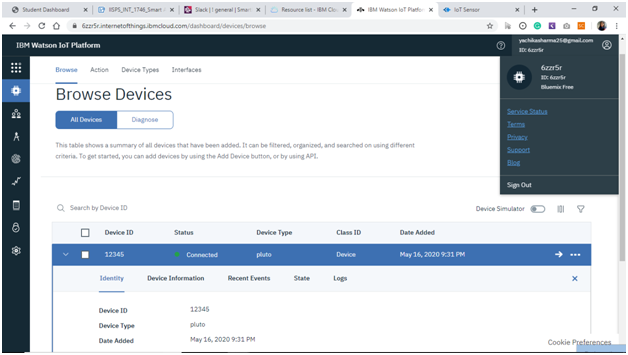
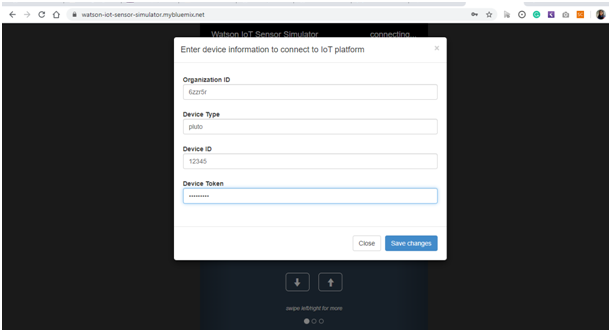
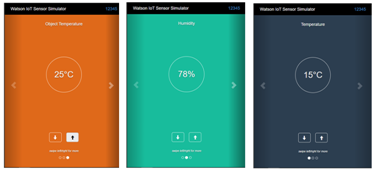


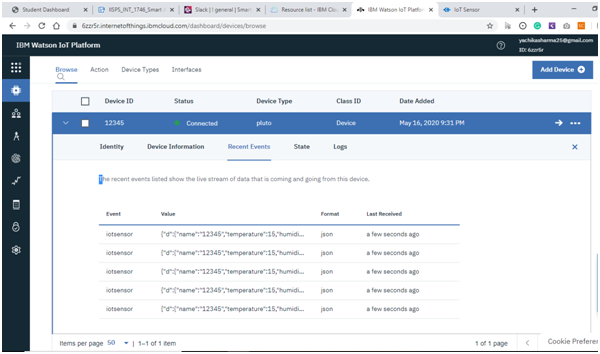
Figure 2: The device which we have created is showing status as “Connected” on IBM Watson IOT Platform

You will get Device credentials save them in a notepad so that we connect to IOT simulator. Go to [link](https://watson-iot-sensor-simulator.mybluemix.net/) for IOT simulator.

Figure 3: Watson IOT Sensor Simulator login page

The following screen appears once your simulator get’s connected.

Figure 4: Watson IOT Sensor Simulator is connected and sending data to the Cloud

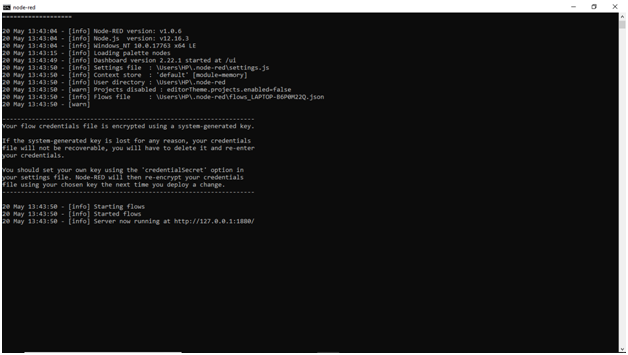
Figure 5: We are receiving data from the Watson IOT Sensor Simulator

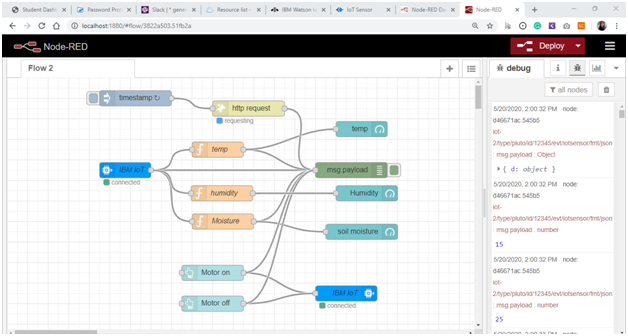
Now once we got the data in the cloud, we will use node red to get the data in a web app. To install node-red in windows follow this [link](https://nodered.org/docs/getting-started/windows#3-run-node-red). After this we would need to externally install IBM iot node in node red using the below code.

**Node-red-contrib-scx-ibmiotapp**

We would need 3 flows:

1. To take the weather data from OpenWeather API.
2. To take sensor data from the IBM cloud.
3. Finally, to transfer the motor control data to the cloud.

Figure 6: Running Node-RED locally

Figure 7: Watson IOT Sensor Simulator data is being received on Node-RED using IBM IOT Input node

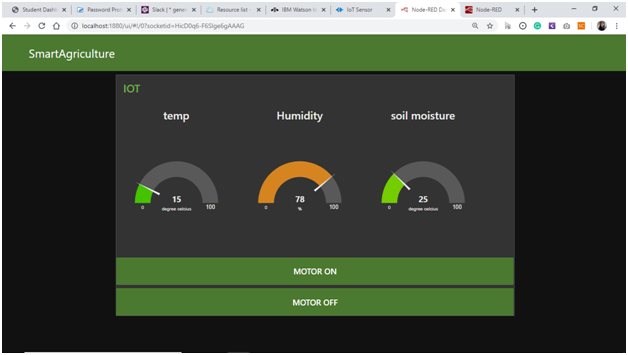
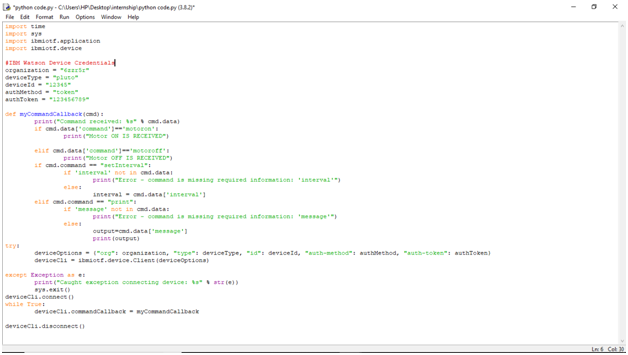


Figure 8: Node-RED Dashboard / UI / WEB APP showing

Figure 9: Python code to receive motor ON/OFF commands via IBM IOT Output node

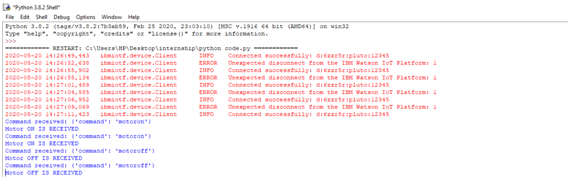
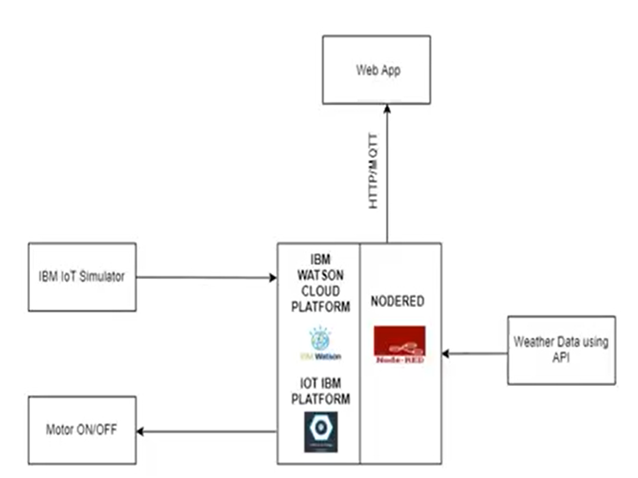


Figure 10: Device is receiving motor ON/OFF commands via IBM IOT Output node

5. FLOWCHART



6. RESULT

We have successfully designed the Smart agricultural system based on iot as we are getting weather data from the Open Weather API and the Watson IoT Sensor Simulator data. Also we are able to send motor on/off commands to the device

7. ADVANTAGES & DISADVANTAGES

The following are the advantages of IoT in Agriculture:

* Allows farmers to continuously monitor soil parameters.
* Weather forecasting of crop location
* Farmers can control their motor from anywhere

The following are the advantages of IoT in Agriculture:

* Network error can cause device to disconnect.
* Weather forecast may not be accurate.

8. APPLICATIONS

With the adoption of IoT in various areas like Industry, Homes and even Cities, huge potential is seen to make everything Intelligent and Smart. Even the Agricultural sector is also adopting IoT technology these days and this in turn has led to the development of “AGRICULTURAL Internet of Things (IoT)”

9. CONCLUSION

By using this system farmers can effectively produce more yield and can save water from wastage. With help of weather forecast service farmer can water their land as per weather. He can also turn off motor when water content in soil is sufficient

10. FUTURE SCOPE

* With help of artificial intelligence, we can suggest farmers to grow a particular crop based on soil data from the sensors.
* We can also control the water supply to crops with help of artificial learning based on soil moisture.

11. BIBILOGRAPHY

1. https//:www.ibm.com

2. https//:www.youtube.com

APPENDIX

API AND DEVICE CREDENTIALS

Device Credentials:

Organization ID: 6zzr5r

Device Type: pluto

Device ID: 12345

Authentication Method: use-token-auth

Authentication Token: 123456789

Open Weather API details:

API call: api.openweathermap.org/data/2.5/weather?q={city name}&&appid={your api key}

API Key on open weather: 1d548196be099afcd3b8e13ce888160a

City: Dīnānagar,IN URL:

api.openweathermap.org/data/2.5/weather?q=Dīnānagar,IN&appid=1d548196be099afcd3b8e13ce888160a

A. Source code

import time

import sys

import ibmiotf.application

import ibmiotf.device

#IBM Watson Device Credentials

organization = "6zzr5r"

deviceType = "pluto"

deviceId = "12345"

authMethod = "token"

authToken = "123456789"

def myCommandCallback(cmd):

print("Command received: %s" % cmd.data)

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()