IOT ENABLED SMART FARMING APPLICATION.

Sprint Delivery – 1

TEAMID : PNT2022TMID29852

1. Introduction

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.

2. Problem Statement

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.

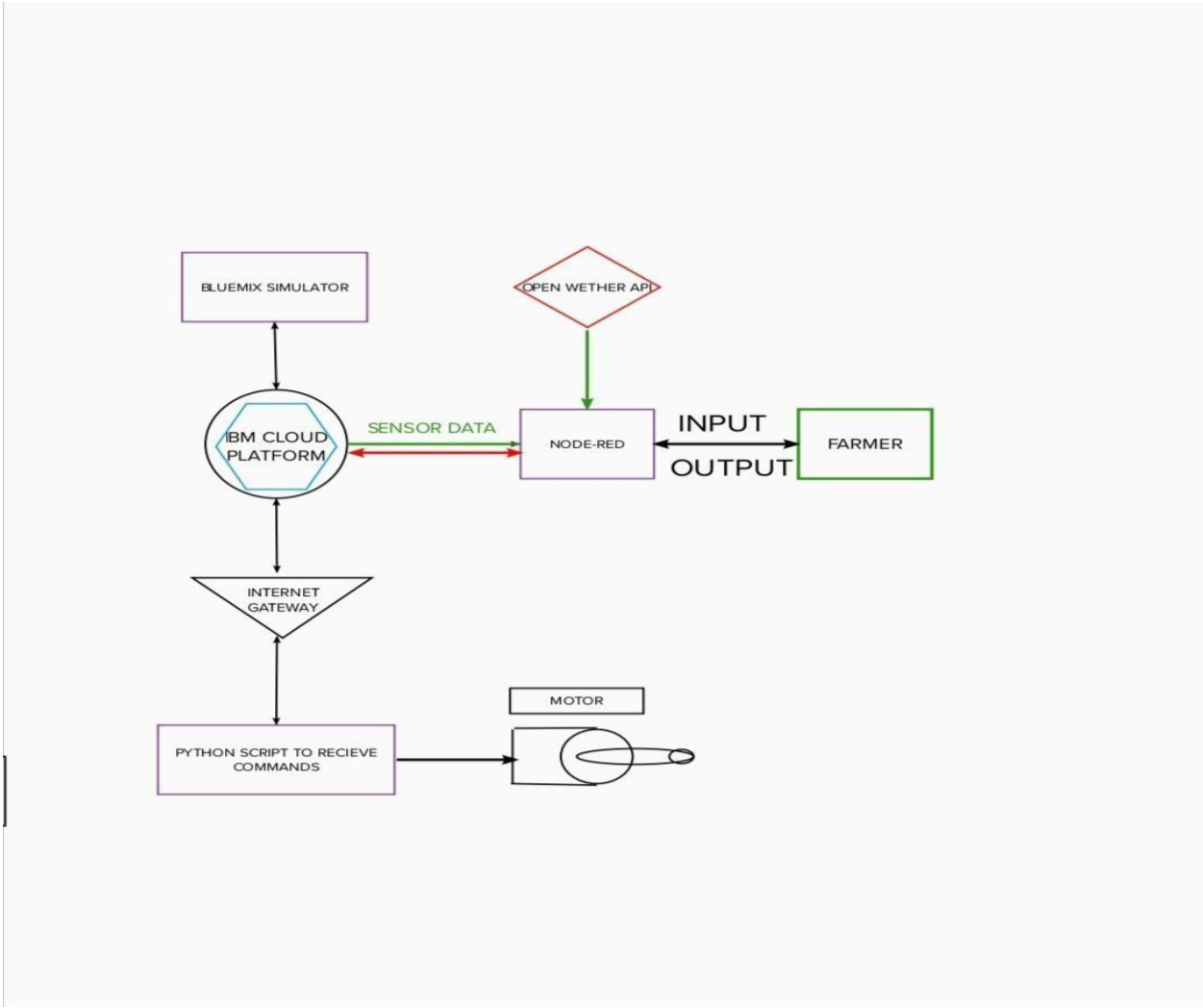
3. Proposed Solution

In order to improve the farmer's working conditions and make them easier, we introduce 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.

4. Theoretical Analysis

4.1 Block Diagram

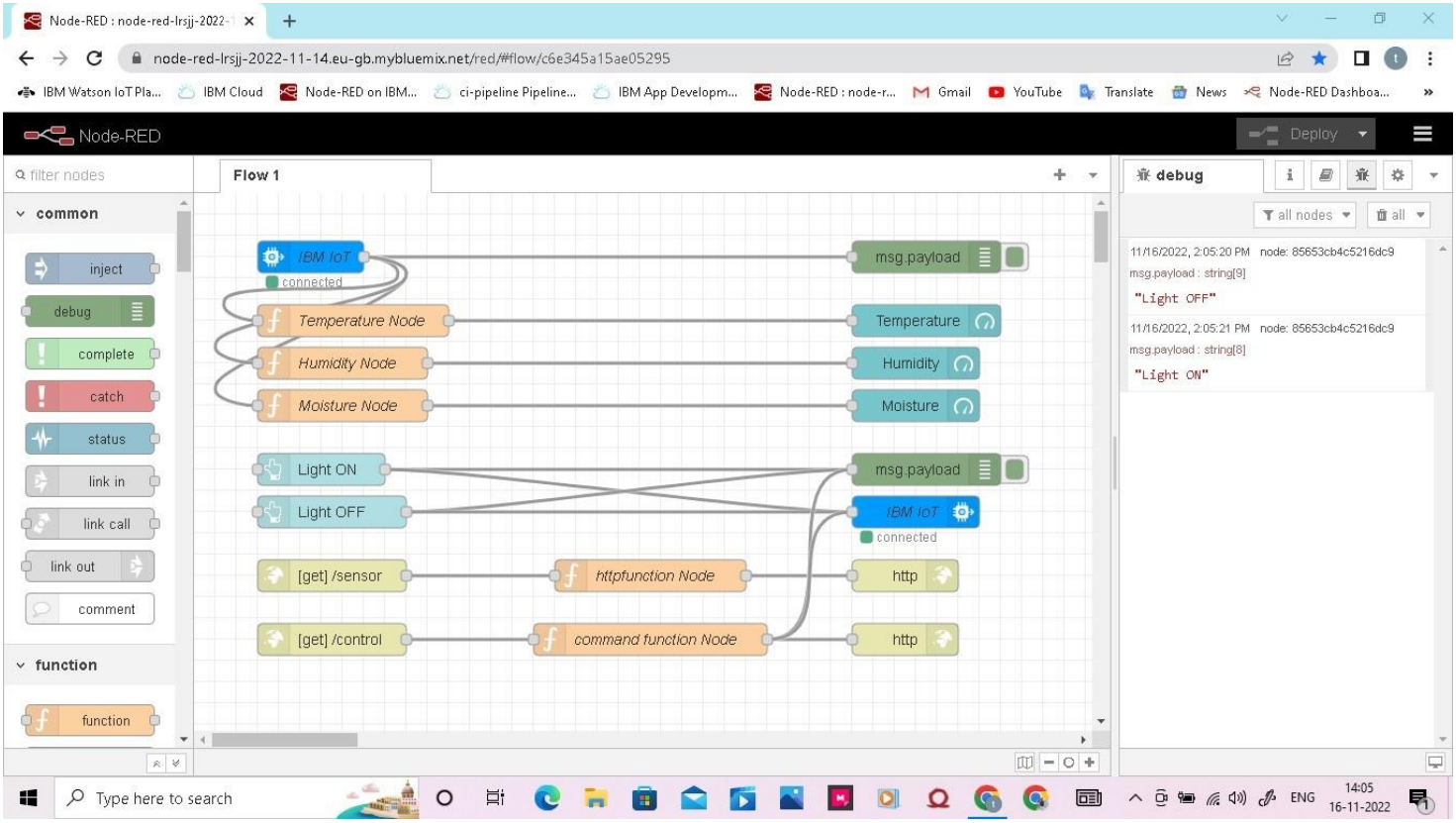
In order to implement the solution, the following approach as shown in the block diagram is used



4.2 Required Software Installation

4.2.A Node-Red

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

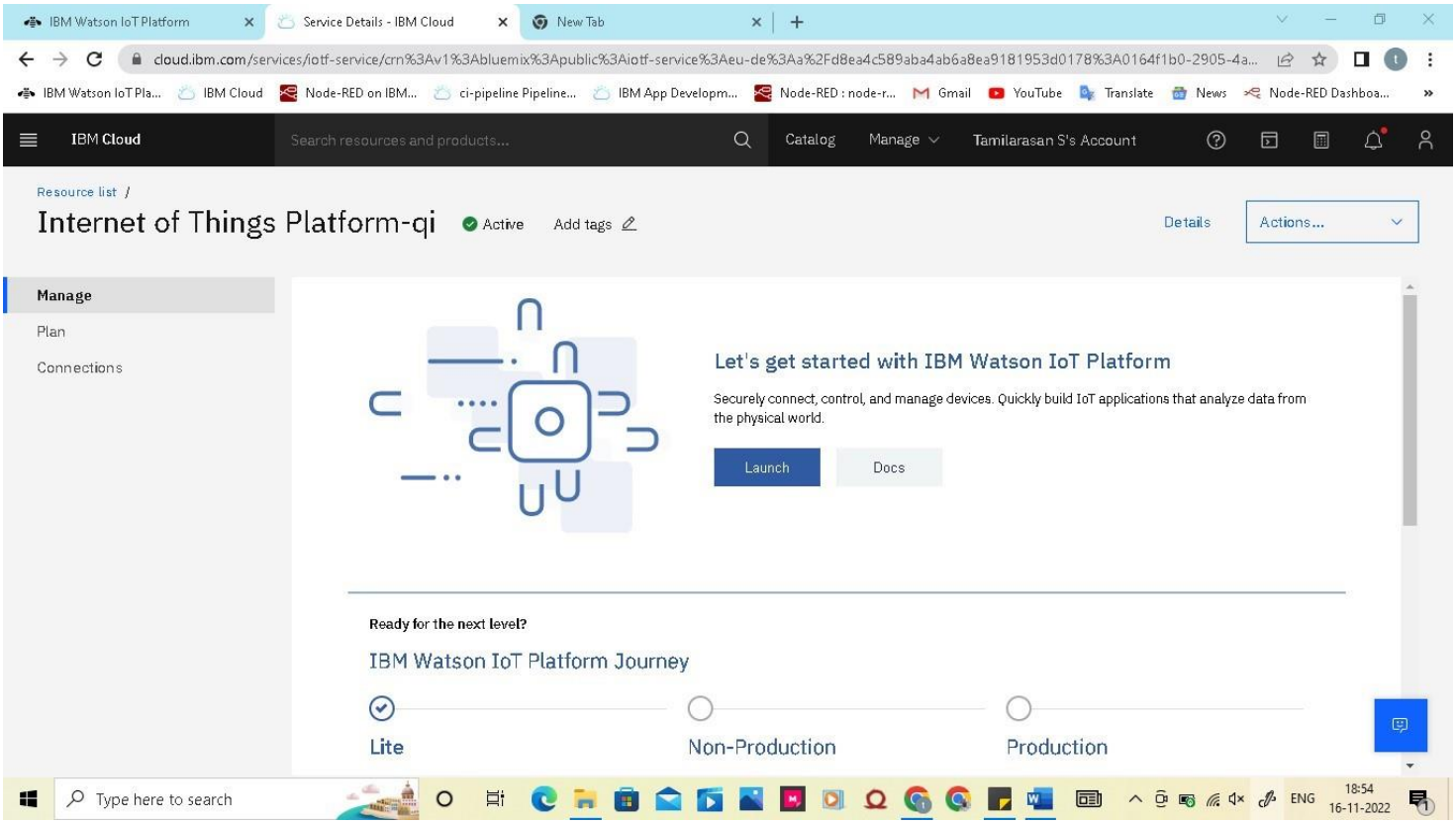
Installation of IBM IoT and Dashboard nodes for Node-Red

In order to connect to IBM Watson IoT platform and create the Web App UI these nodes are required 1. IBM IoT node

2. Dashboard node

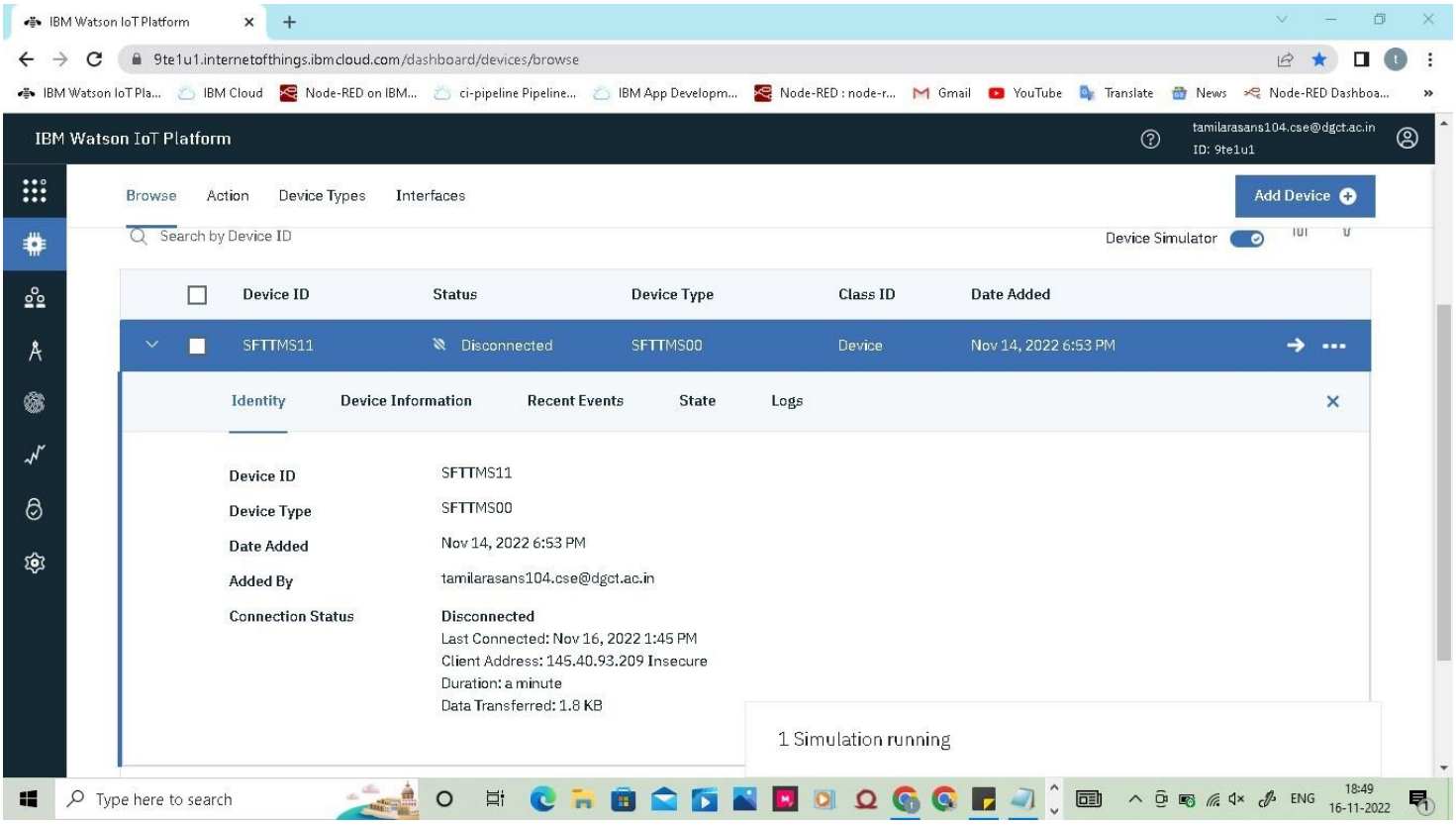
4.2.B IBM Watson IoT Platform

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.



Steps to configure:

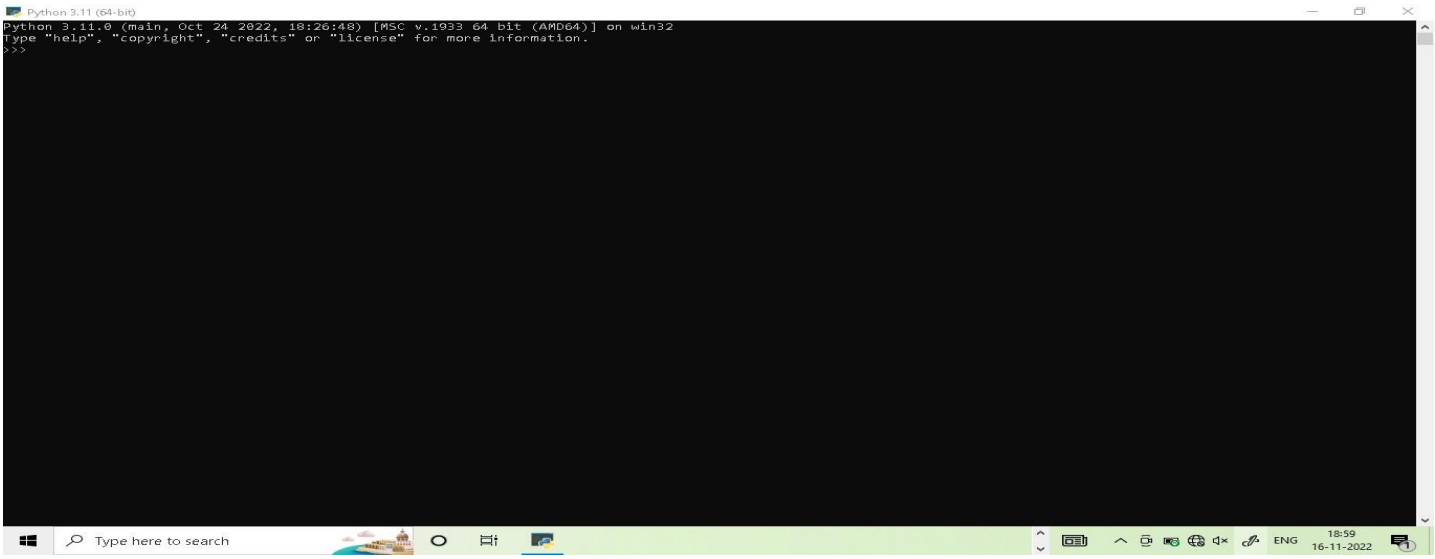
- Create an account in IBM cloud using your email ID
- Create IBM Watson Platform in services in your IBM cloud account
- Launch the IBM Watson IoT Platform
- Create a new device
- Give credentials like device type, device ID, Auth. Token
- Create API key and store API key and token elsewhere.



4.2.C Python IDE

Install Python3 compiler

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



Code:

```
import random

import sys

import time

import ibmiotf.application

import ibmiotf.device


#provide Your IBM Watson
Device Credentials

organization = "9te1u1"

deviceType = "SFTTMS00"

deviceID = "SFTTMS11"

authMethod = "token"

authToken = "PNTIBMSb18"



#Initialize GPIO

def myCommandCallback(cmd):

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

   status=cmd.data['command']
```

```
    if status=="lighton":
        print  ("led is on")
    elif status == "lightoff":
        print ("led is off")
    else:
      print ("please send proper
command")
try:
      deviceOptions =
{'org':organization,'type':deviceT
ype,'id':deviceID,'auth-
method':authMethod, 'auth-
token': authToken}
      deviceCli =
ibmiotf.device.Client(deviceOptio
ns)

#.....

except Exception as e:
       print("caught exception
connecting device:%s" % str(e))
       sys.exit()
```

```
# connect and send a datapoint
"hello"with value "world" info
the cloud as an event of
type"greetings"10 times

deviceCli.connect()


while True:

        #Get sensor Data from
DHT11



temp=random.randint(0,100)

Humid=random.randint(0,100)

soilmoisture=random.randint(0,1
00)

        data = { 'temp' : temp,
'Humid': Humid,
'soilmoisture':soilmoisture}

        #print data
```

```
        def myOnPublishCallback():
            print ("published
Temperature = %s C" %  temp,
"Humidity = is %s %%" % Humid,
"soilmoisture= is %s %%" %
soilmoisture,"to IBM Watson")



        success =
deviceCli.publishEvent("IOTSenso
r",
"json",data,qos=0,on_publish=m
yOnPublishCallback)
        if not success:
            print("Not connected to
IOTF")
        time.sleep(5)



        deviceCli.commandCallback
= myCommandCallback
# Disconnect the device and
application from the cloud
deviceCli.disconnect()
```