IBM NALAIYATHIRAN

SMART FARMER-IOT ENABLED SMART FARMING APPLICATION

SPRINT 1

Title	Smart farmer-IoT enabled smart farming application
Domain	Internet of Things
Team ID	PNT2022TMID44170
Project Name	Project – Smart Farmer-IoT Enabled smartFarming Application

Introduction:

The main aim of this project is to help farmers to automate their farms by providing them with a Web App through which they can monitor the parameters 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.

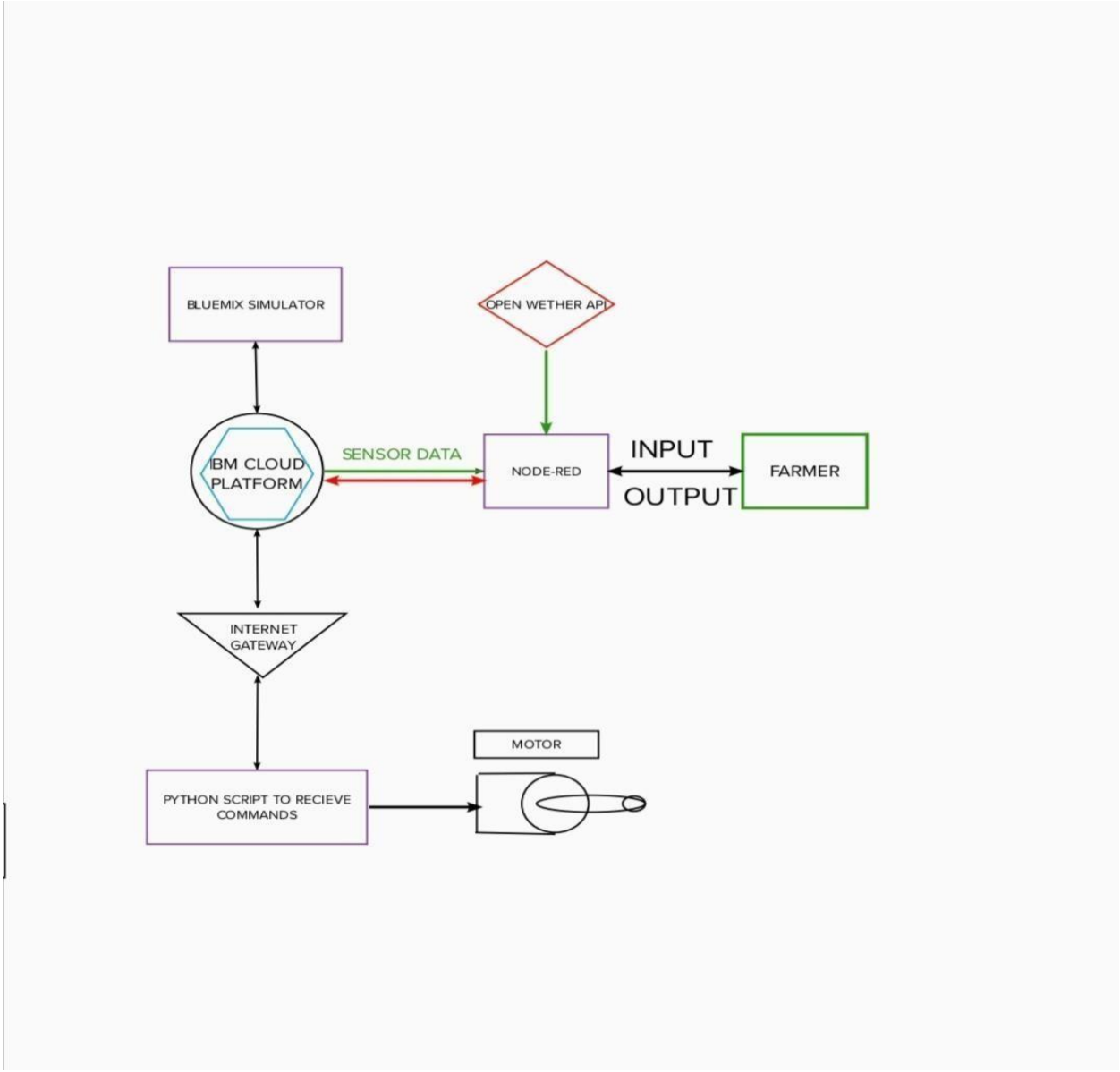
Proposed Solution:

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.

Theoretical Analysis:

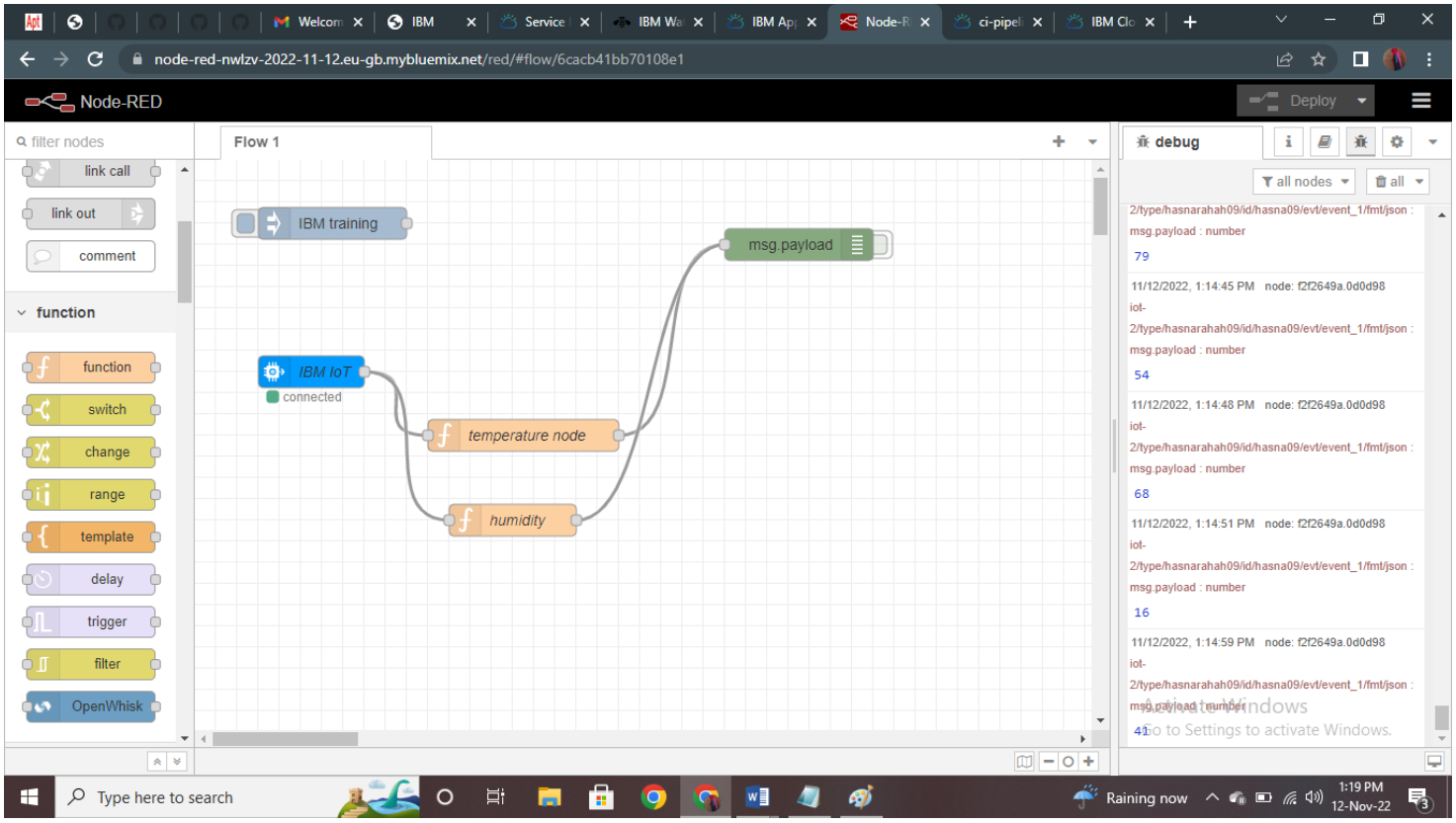
Block Diagram:

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



Required Software Installation
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.



Installation :

- First install npm/node.js
- Open cmd prompt
- Type => npm install node-red

To run the application :

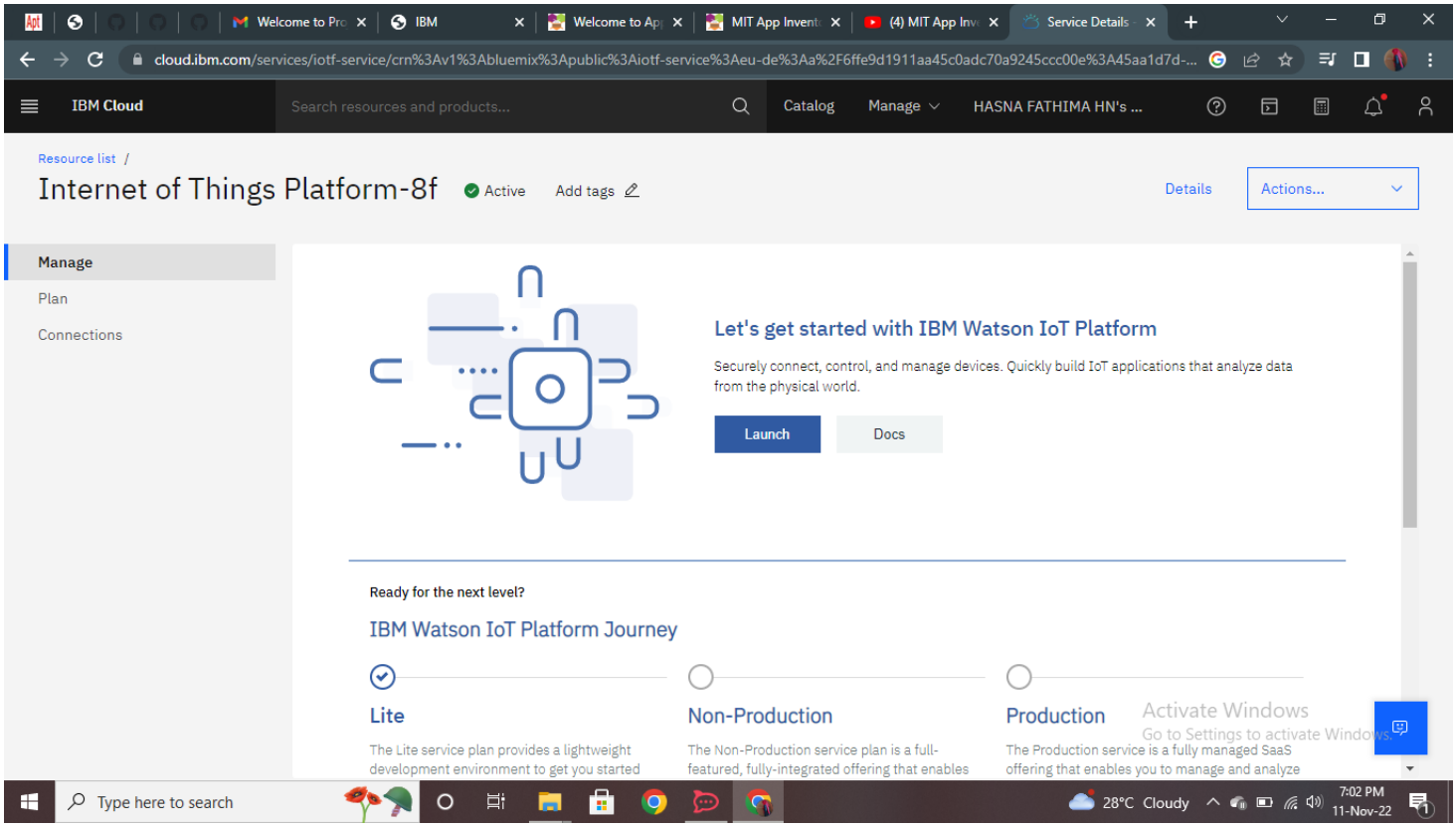
Installation of IBM IoT and Dashboard nodes for Node-Red

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

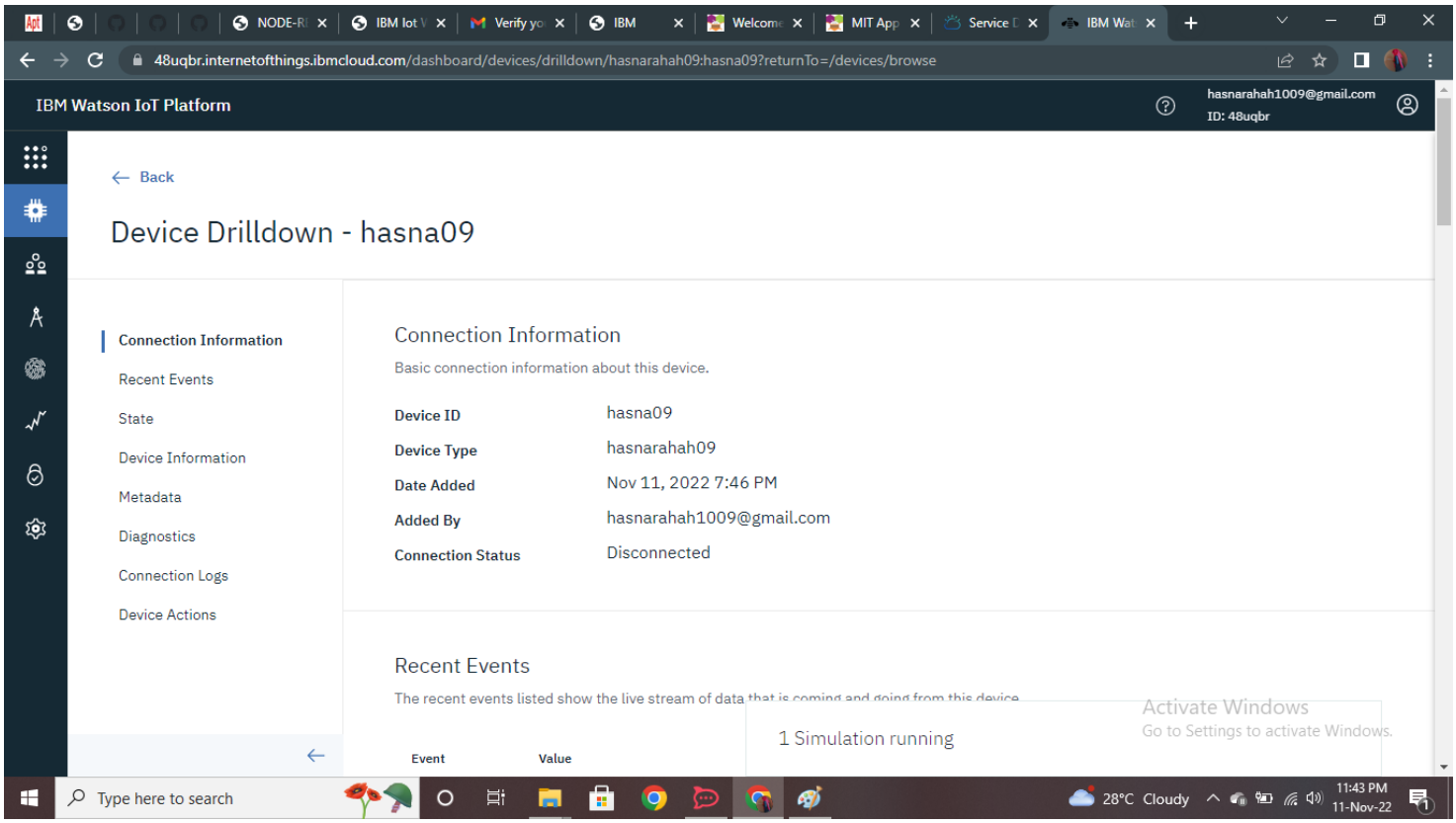
IBM Watson IoT Platform

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.


48uqbr.internetofthings.ibmcloud.com/dashboard/devices/drilldown/hasnarahah09:hasna09?returnTo=/devices/browse
IBM Watson IoT Platform
hasnarahah1009@gmail.com
ID: 48uqbr
Back
Device Drilldown - hasna09
Connection Information
Recent Events
State
Device Information
Metadata
Diagnostics
Connection Logs
Device Actions
Connection Information
Basic connection information about this device.
Device ID
hasna09
Device Type
hasnarahah09
Date Added
Nov 11, 2022 7:46 PM
Added By
hasnarahah1009@gmail.com
Connection Status
Disconnected
Recent Events
The recent events listed show the live stream of data that is coming and going from this device
Event
Value
1 Simulation running
Activate Windows
Go to Settings to activate Windows.
Type here to search
28°C Cloudy
11:43 PM
11-Nov-22

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 time import sys import ibmiotf.application import ibmiotf.device import random

#Provide your IBM Watson Device Credentials organization = " 48qubr" deviceType = "hasnarahah09" deviceId = "hasna09" authMethod = "token" authToken = "87654321"

Initialize GPI

```
        def myCommandCallback(cmd):
        print("Command  received:                    %s"  %
cmd.data['command'])
status=cmd.data['command']    if status=="motoron":
print     ("motor is on")          elif status == "motoroff":
print("motor is off")
else :print ("please send  proper command")
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
                anevent of type "greeting" 10 times deviceCli.connect()


                while True:
                        #Get Sensor Data from DHT11


                        temp=random.r
                        andint(90,110)
                        Humid=random
                        .randint(60,100
                        )
                        Mois=random.randint(20,120)


                    data = { 'temp' : temp, 'Humid': Humid, 'Mois'
                    :Mois}#print data    def
               myOnPublishCallback():
               print ("Published Temperature
               = %s C" % temp, "Humidity = %s
               %%" % Humid,
               "Moisture =%sdeg
               c" %Mois, "to IBM
               Watson")
                    success   =   deviceCli.publishEvent("IoTSensor",   "json",   data,   qos=0,
```

```
on_publish=myOnPublishCallback)                    if not success:
                                                        print("N
ot connectedto IoTF")  time.sleep(10)

        deviceCli.commandCallback = myCommandCallback

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

Connecting Sensors with Arduino using C++ code:

```
        #include
"Arduino.h" #include
"dht.h"
#include "SoilMoisture.h"
#define dht_apin A0
const int sensor_pin = A1; //soil moisture int
pin_out = 9;dht DHT; int c=0; void setup()
{
pinMode(2, INPUT); //Pin 2 as INPUT pinMode(3,
OUTPUT); //PIN3 as OUTPUT pinMode(9,
OUTPUT);//output for pump
}
 void
loop
()
{
  if (digitalRead(2) == HIGH)
  {
  digitalWrite(3, HIGH);          // turn the LED/Buzz ON
  delay(10000); // wait for 100 msecond digitalWrite(3, LOW);
  // turn theLED/Buzz OFF delay(100);
  }
```

```
  Serial.begin(9600);
    delay(1000);
   DHT.read11(dht_apin);  //temperature
                          floath=DHT.humidity;
float        t=DHT.temperature;
delay(5000);Serial.begin(9600);
float moisture_percentage; int
sensor_analog;
             sensor_analog       =
analogRead(sensor_pin);
moisture_percentage = ( 100 - (
(sensor_analog/1023.00) * 100 ) );float
m=moisture_percentage; delay(1000); if(m<40)//pump
{ while(m<40)
{
digitalWrite(pin_out,HIGH);       //open pump
sensor_analog =analogRead(sensor_pin);
moisture_percentage = ( 100 - ( (sensor_analog/1023.00) * 100 )
); m=moisture_percentage; delay(1000);
}
digitalWrite(pin_out,LOW);             //closepump
} if(c>=0)
{
mySerial.begin(9600);
                                    delay(15000)
;Serial.begin(9600); delay(1000);
Serial.print("\r"); delay(1000);
Serial.print((String)"update-
>"+(String)"Temprature="+t+(String)"Humidity="+h+(String
)"Moisture="+m); delay(1000);
```

```
    }
}
```

Circuit Diagram: