Smart Agriculture System based on IOT

Introduction:

This project is build to reduce the efforts of farmers by developing a smart agriculture system with the help of IBM cloud IOT platform service, IBM IOT sensor and Node-red service. By this system farmer can operate the motor that is ON/OFF based on moisture level even when he is far away from the farm.

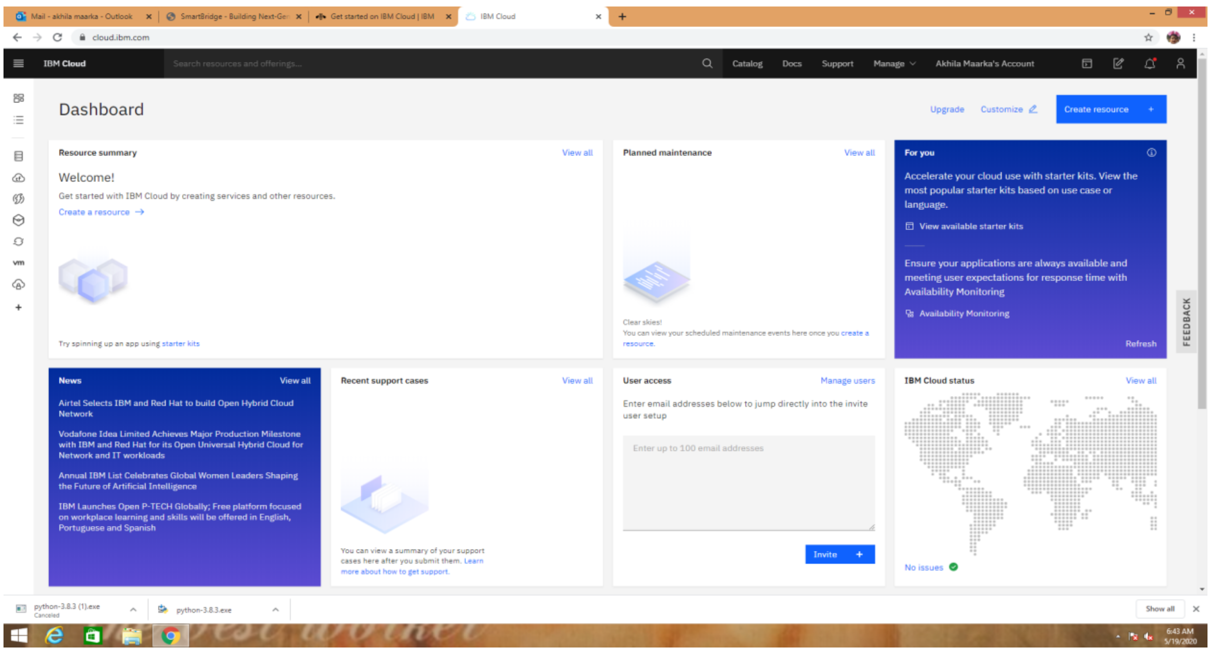
Project Requirements:

- IBM cloud Watson IOT platform service
- Node-Red service
- Python IDLE
- IBM IOT simulator

Methodology:

Smart Agriculture System is constructed by first creating IBM cloud account and accessing Watson service in it. A device is created so that it receives the data which is sent by the IBM IOT simulator. IBM IOT simulator contains the data of parameters like temperature, humidity and soil moisture of a particular farm. Then the data which is received by the IBM cloud is integrated to the Node-Red. A web application is build in Node-Red by using nodes in it to give user interface according to the data given by IBM IOT sensor. Weather API which is an open source gives us the information of weather of a particular region which is mentioned is also integrated in to the Node-Red. Finally a python code written in python IDLE to subscribe to IBM IOT platform and get the commands when the motor is ON/OFF.

IBM Cloud Account:



Mail - akhila maarka - Outlook | SmartBridge - Building Next-Gen | Get started on IBM Cloud | IBM | IBM Cloud

cloud.ibm.com

IBM Cloud | Search resources and offerings... | Catalog | Docs | Support | Manage | Akhila Maarka's Account

Dashboard

Upgrade | Customize | Create resource +

Resource summary View all

Welcome!

Get started with IBM Cloud by creating services and other resources.

Create a resource →

Try spinning up an app using starter kits

Planned maintenance View all

Clear skies!
You can view your scheduled maintenance events here once you create a resource.

For you

Accelerate your cloud use with starter kits. View the most popular starter kits based on use case or language.

View available starter kits

Ensure your applications are always available and meeting user expectations for response time with Availability Monitoring

Availability Monitoring

Refresh

FEEDBACK

News View all

Airtel Selects IBM and Red Hat to build Open Hybrid Cloud Network

Vodafone Idea Limited Achieves Major Production Milestone with IBM and Red Hat for its Open Universal Hybrid Cloud for Network and IT workloads

Annual IBM List Celebrates Global Women Leaders Shaping the Future of Artificial Intelligence

IBM Launches Open P-TECH Globally; Free platform focused on workplace learning and skills will be offered in English, Portuguese and Spanish

Recent support cases View all

You can view a summary of your support cases here after you submit them. Learn more about how to get support.

User access Manage users

Enter email addresses below to jump directly into the invite user setup

Enter up to 100 email addresses

Invite +

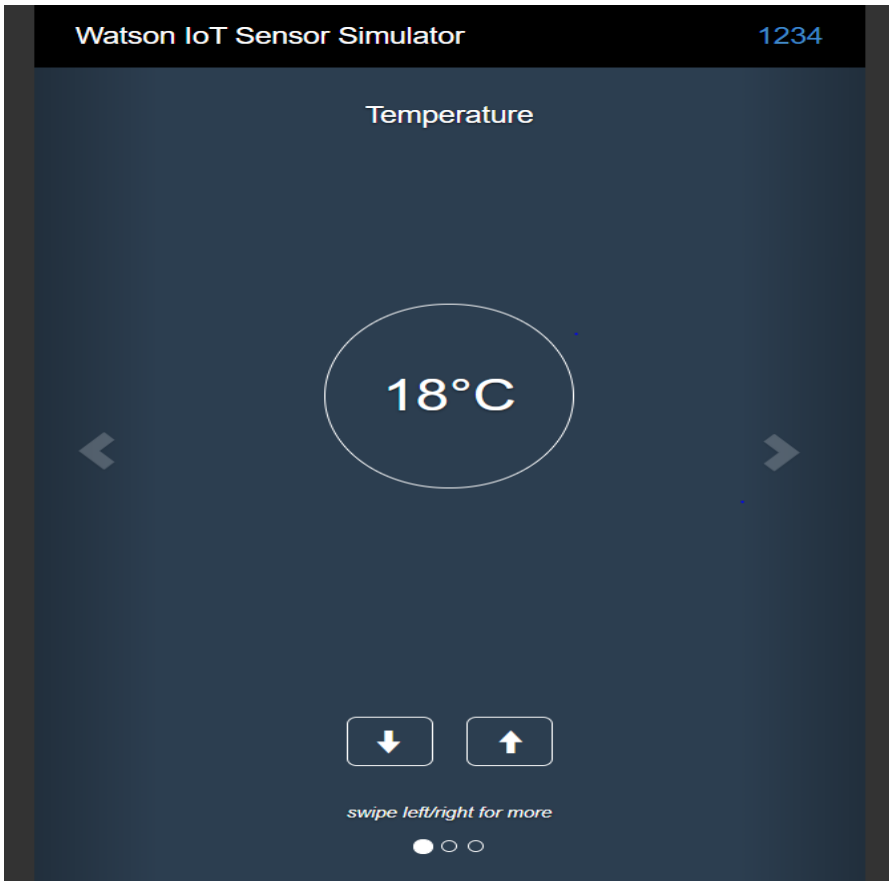
IBM Cloud status View all

No issues

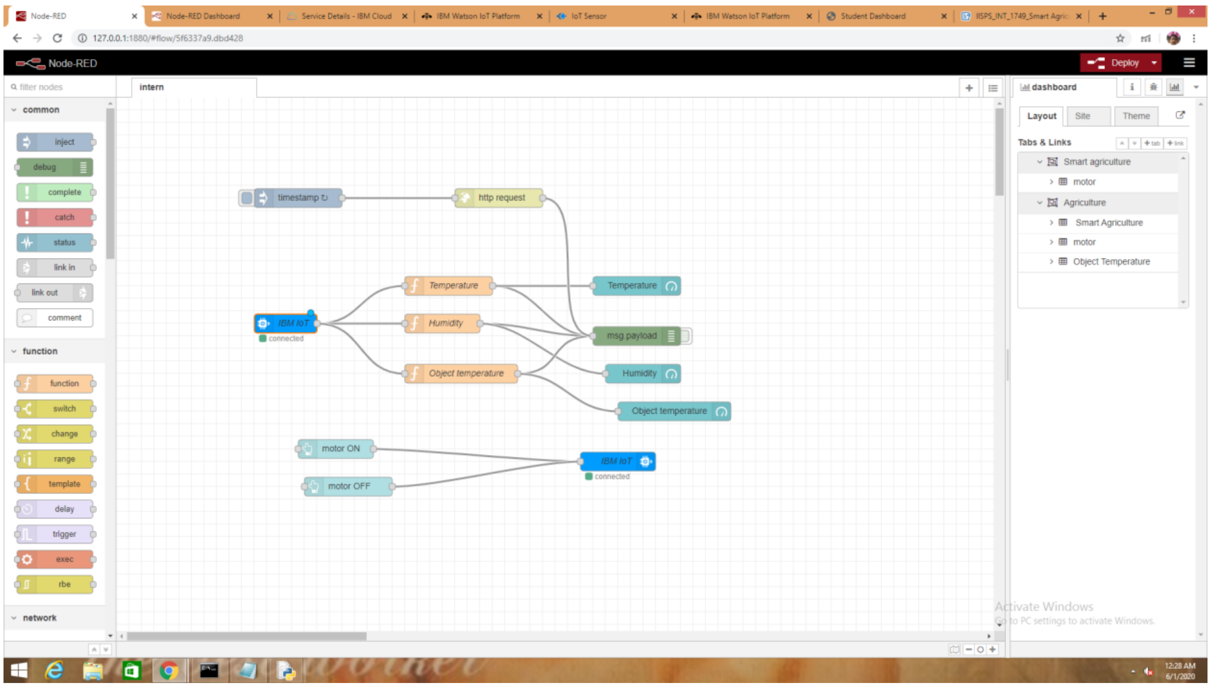
python-3.8.3 (1).exe Canceled | python-3.8.3.exe | Show all

6:43 AM 5/19/2020

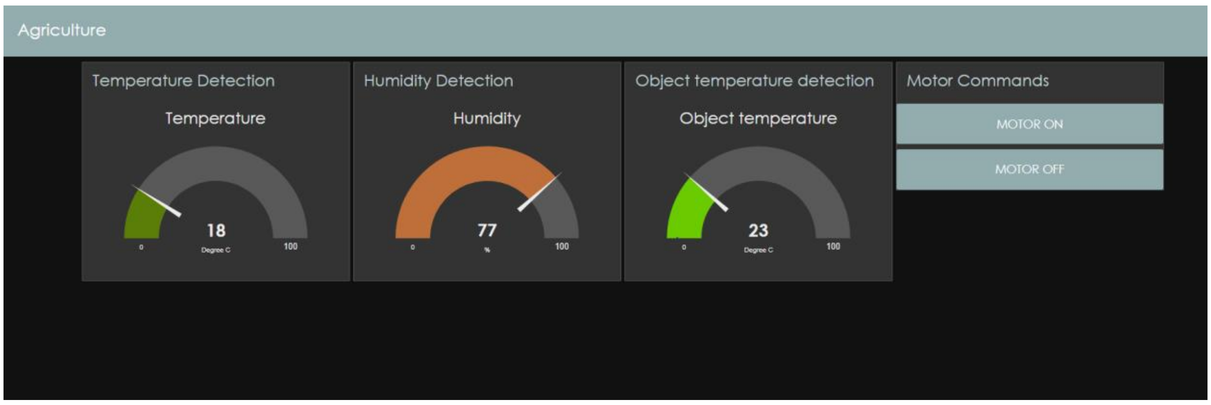
IBM IOT simulator



Node -Red:



User Interface:



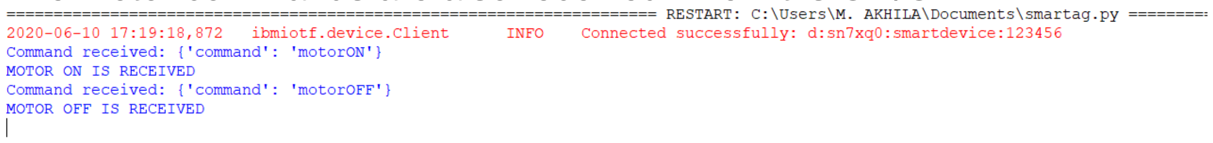
Python Code:

```
import time
import sys
import ibmiotf.application # to install pip install ibmiotf
import ibmiotf.device

#Provide your IBM Watson Device Credentials
organization = "sn7xq0" #replace the ORG ID
deviceType = "smartdevice"#replace the Device type
deviceId = "123456"#replace Device ID
authMethod = "token"
authToken = "9441718473" #Replace the authtoken

def myCommandCallback(cmd): # function for Callback
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()
```

Result:

This 'Smart Agriculture System based on IOT' enabled farmer to reduce his daily tasks and to save his time instead of going to the farm daily. The motor commands are also received when it is ON/OFF.



```
==================================================================== RESTART: C:\Users\M. AKHILA\Documents\smartag.py =========
2020-06-10 17:19:18,872   ibmiotf.device.Client      INFO    Connected successfully: d:sn7xq0:smartdevice:123456
Command received: {'command': 'motorON'}
MOTOR ON IS RECEIVED
Command received: {'command': 'motorOFF'}
MOTOR OFF IS RECEIVED
```