TEAM ID	PNT2022TMID27944
PROJECT NAME	IoT Based Smart Crop Protection System for Agriculture

Using Python IDLE

Source Code:

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
#Provide your IBM Watson Device Credentials
organization = "bx1po5"
deviceType = "abcd"
deviceId = "1234"
authMethod = "token"
authToken = "12345678"
# Initialize GPIO
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="motoron":
        print ("motor is on")
    else :
        print ("motor is off")
    #print(cmd)
```

```
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:
    #Get Sensor Data from DHT1
    temp=random.randint(0,100)
    Humid=random.randint(0,100)
    moisture=random.randint(0,100)
    data = { 'temp' : temp, 'Humid': Humid, 'moisture': moisture }
    #print data
    def myOnPublishCallback():
       print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % Humid, "moisture = %s %%" % moisture,"to IBM Watson")
    success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
    if not success:
       print("Not connected to IoTF")
    time.sleep(1)
    deviceCli.commandCallback = myCommandCallback
```

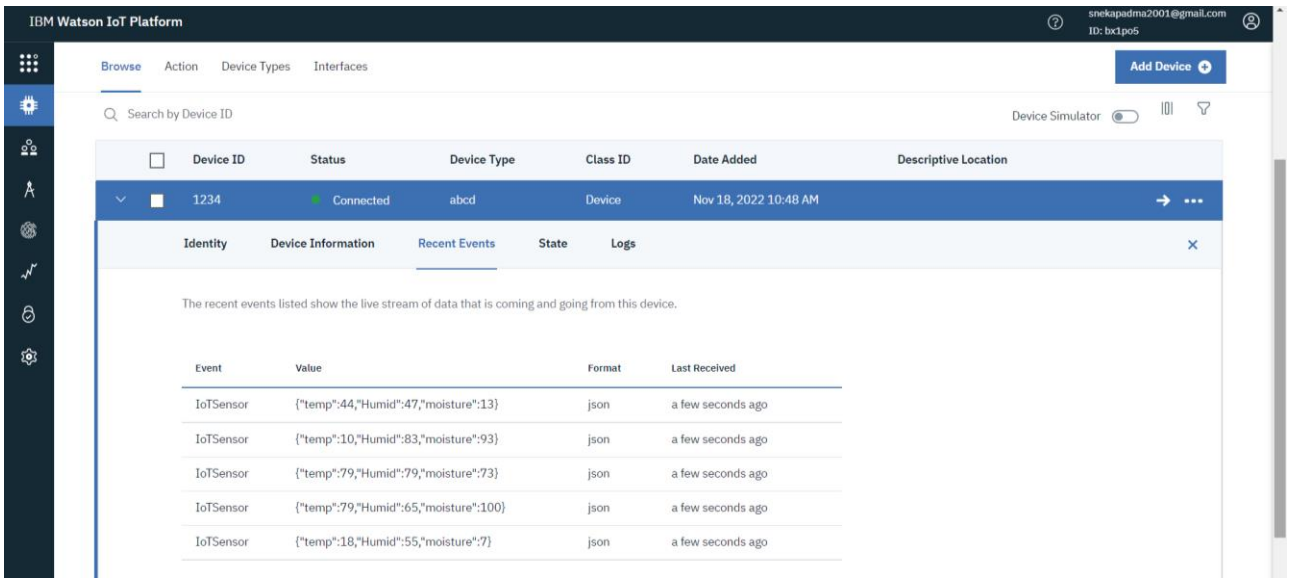
Disconnect the device and application from the cloud

deviceCli.disconnect()

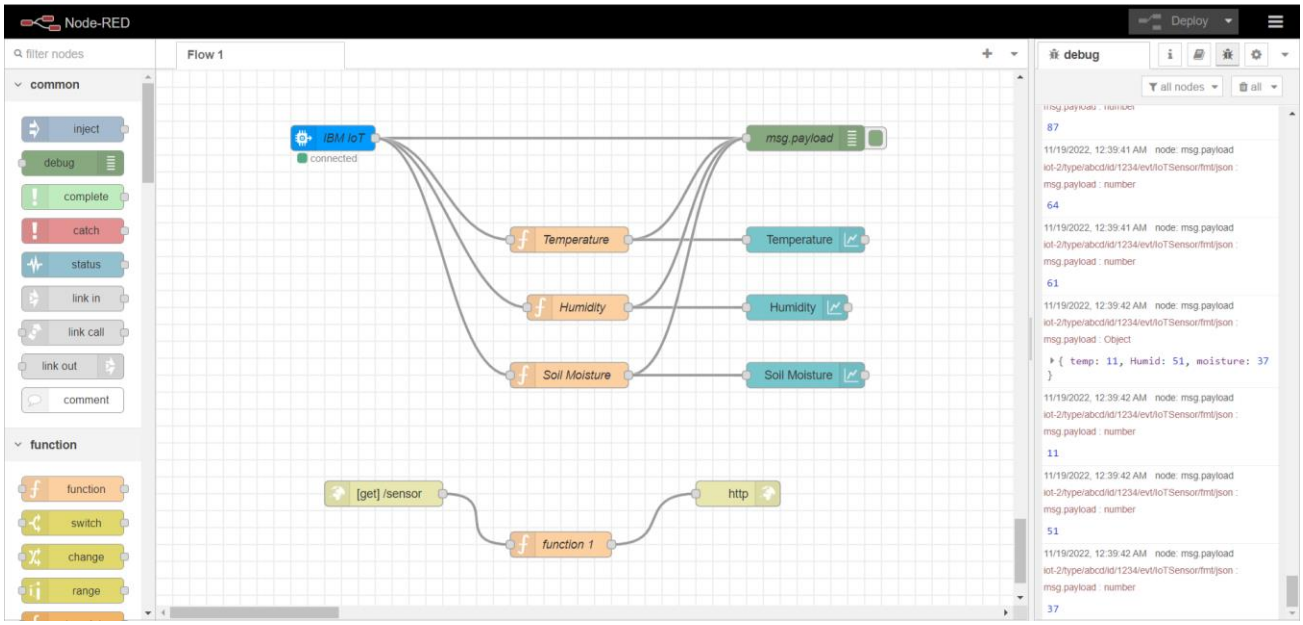
Output:

```
*Python 3.7.0 Shell*
File  Edit  Shell  Debug  Options  Window  Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
============= RESTART: C:/Users/sobi/OneDrive/Desktop/Sprint1.py =============
2022-11-18 22:50:00,333   ibmiotf.device.Client      INFO    Connected successfully: d:bx1po5:abcd:1234
Published Temperature = 82 C Humidity = 63 % moisture = 73 % to IBM Watson
Published Temperature = 68 C Humidity = 99 % moisture = 75 % to IBM Watson
Published Temperature = 82 C Humidity = 52 % moisture = 18 % to IBM Watson
Published Temperature = 71 C Humidity = 14 % moisture = 50 % to IBM Watson
Published Temperature = 19 C Humidity = 74 % moisture = 55 % to IBM Watson
Published Temperature = 19 C Humidity = 37 % moisture = 47 % to IBM Watson
Published Temperature = 84 C Humidity = 61 % moisture = 16 % to IBM Watson
Published Temperature = 97 C Humidity = 40 % moisture = 98 % to IBM Watson
Published Temperature = 88 C Humidity = 59 % moisture = 46 % to IBM Watson
Published Temperature = 54 C Humidity = 33 % moisture = 30 % to IBM Watson
Published Temperature = 71 C Humidity = 26 % moisture = 33 % to IBM Watson
Published Temperature = 64 C Humidity = 51 % moisture = 78 % to IBM Watson
Published Temperature = 64 C Humidity = 17 % moisture = 32 % to IBM Watson
Published Temperature = 19 C Humidity = 82 % moisture = 69 % to IBM Watson
Published Temperature = 69 C Humidity = 49 % moisture = 94 % to IBM Watson
Published Temperature = 48 C Humidity = 54 % moisture = 42 % to IBM Watson
Published Temperature = 55 C Humidity = 100 % moisture = 67 % to IBM Watson
Published Temperature = 35 C Humidity = 55 % moisture = 32 % to IBM Watson
Published Temperature = 76 C Humidity = 47 % moisture = 55 % to IBM Watson
Published Temperature = 83 C Humidity = 3 % moisture = 8 % to IBM Watson
Published Temperature = 58 C Humidity = 12 % moisture = 56 % to IBM Watson
Published Temperature = 11 C Humidity = 52 % moisture = 53 % to IBM Watson
Published Temperature = 64 C Humidity = 97 % moisture = 62 % to IBM Watson
Published Temperature = 29 C Humidity = 46 % moisture = 3 % to IBM Watson
```

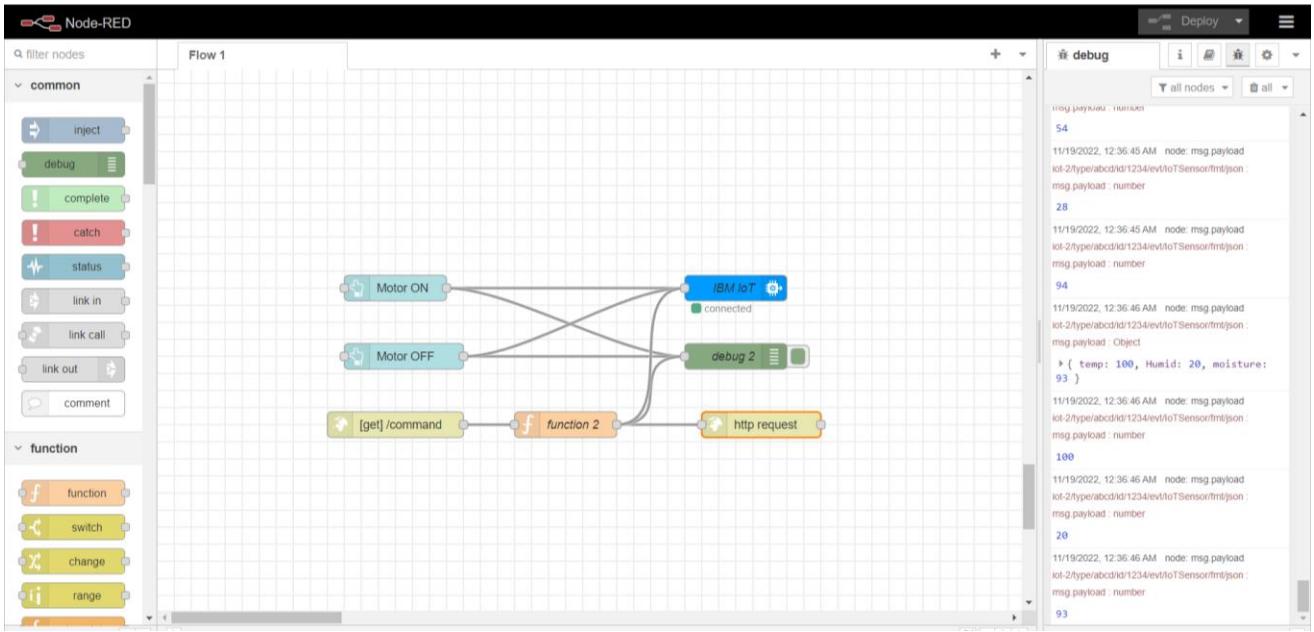
Python to IBM Watson IoT Platform:



Node-Red Flow Diagram for Sensor:



Node-Red Flow Diagram to configure with Button:



Source Code:

```
msg.payload = { "temp": global.get("t"),
                "Humid": global.get("h"),
                "moisture": global.get("m")}
return msg;
```

For Temperature:

```
msg.payload = msg.payload.temp
global.set("t",msg.payload)
return msg;
```

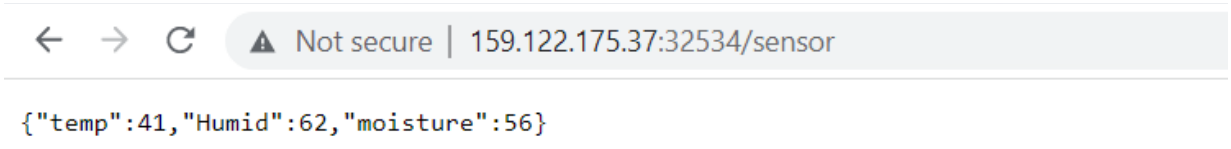
For Humidity:

```
msg.payload = msg.payload.Humid
global.set("h", msg.payload)
return msg;
```

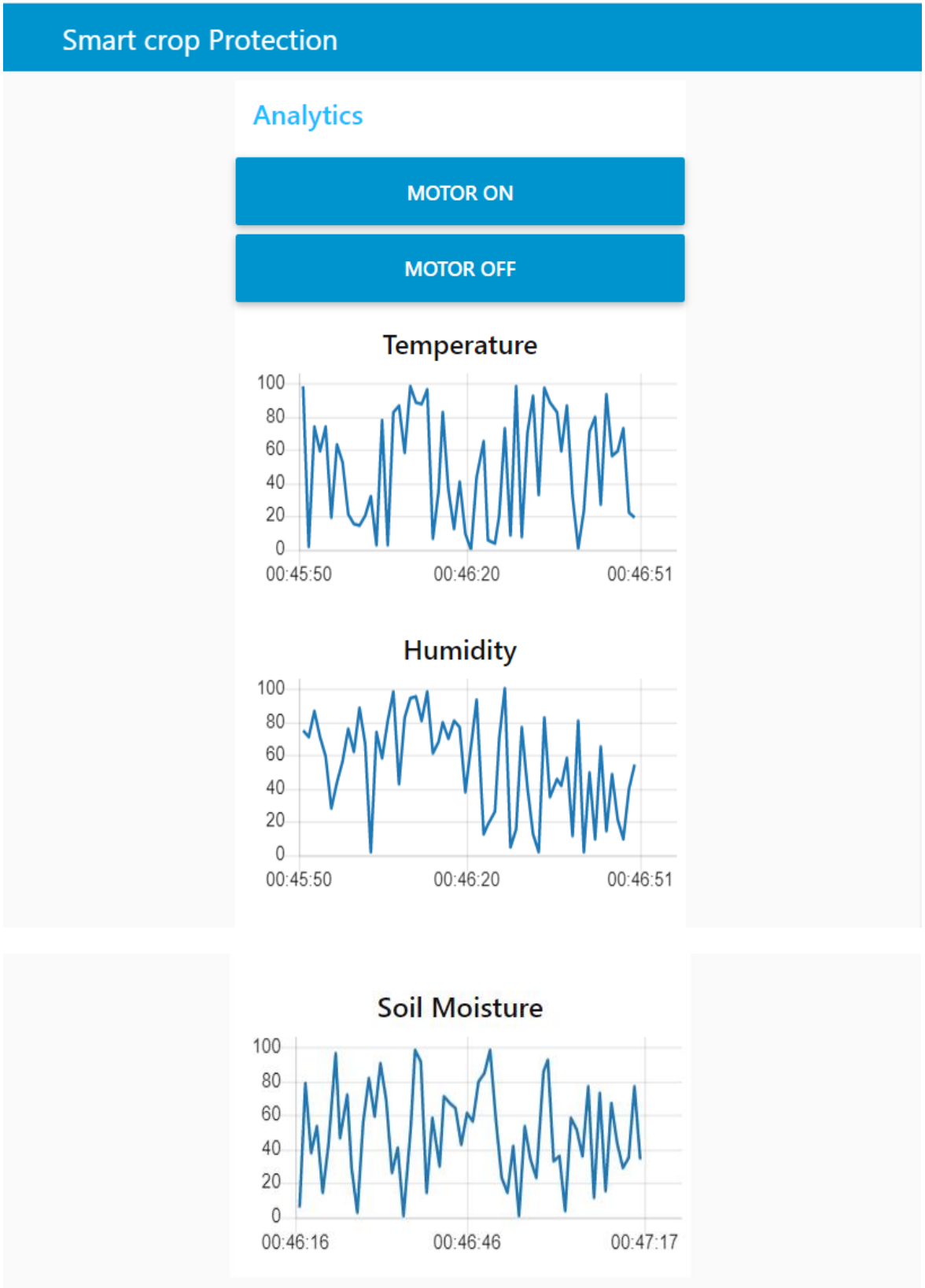
For Moisture:

```
msg.payload = msg.payload.moisture
global.set("m", msg.payload)
return msg;
```

HTTP Request using Node-Red:



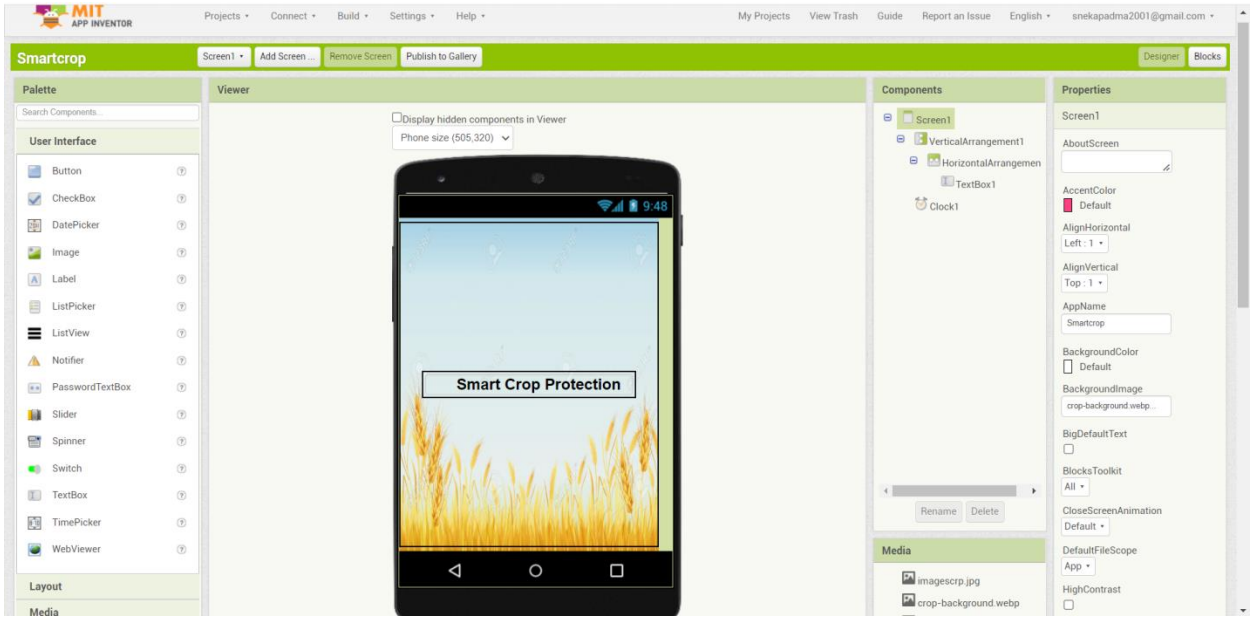
Generate the output for recent event:



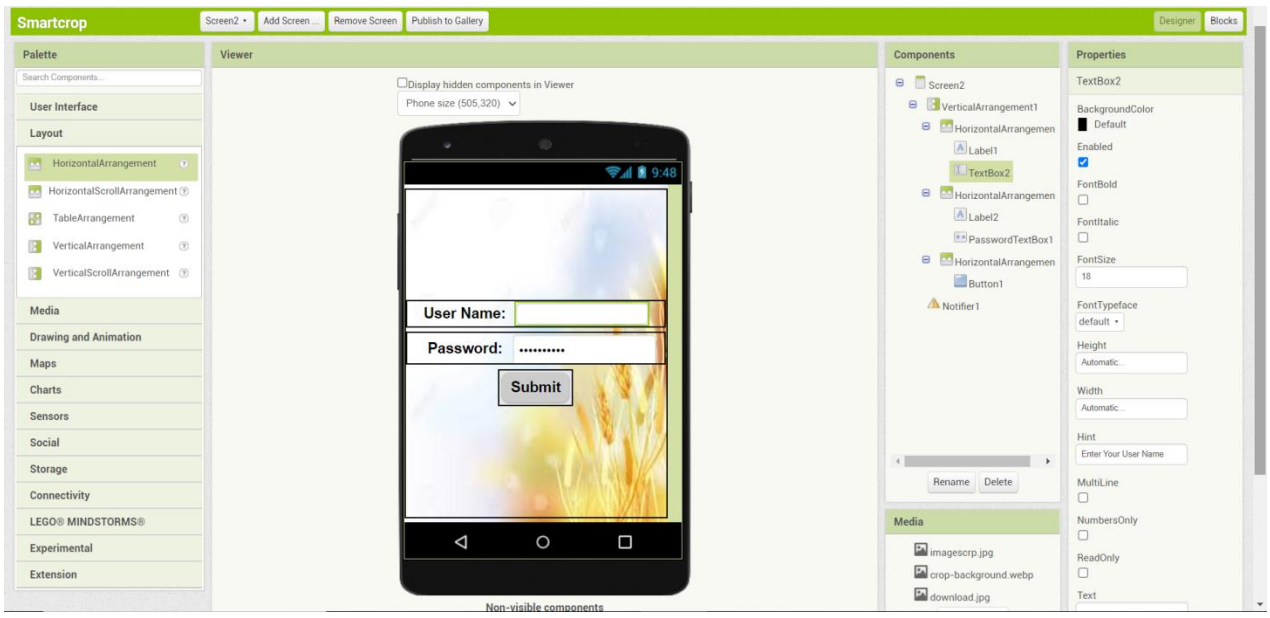
Mit Application Designer:

(FRONTEND)

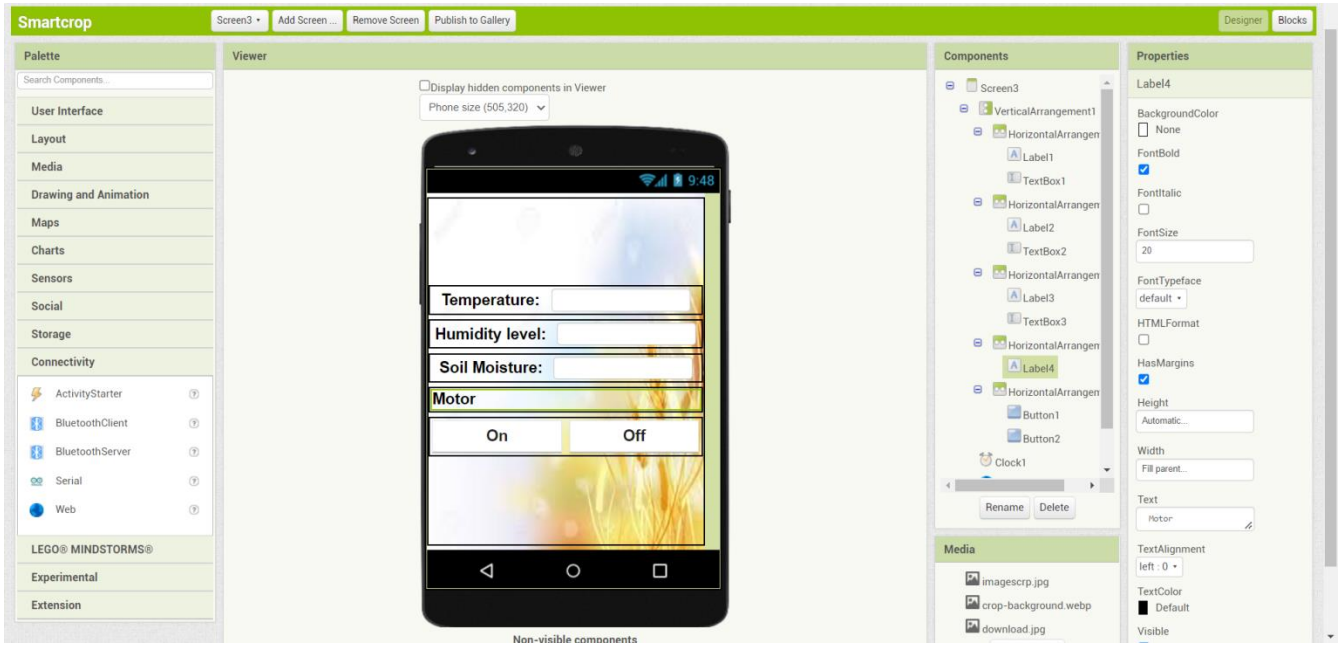
Screen-1:



Screen-2:



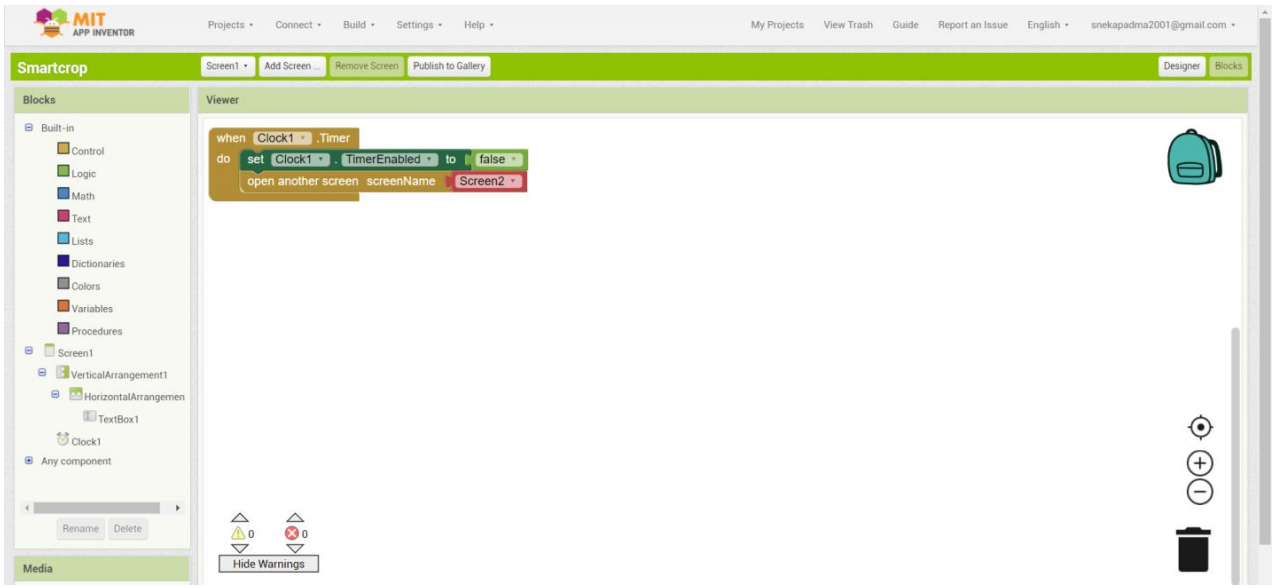
Screen-3:



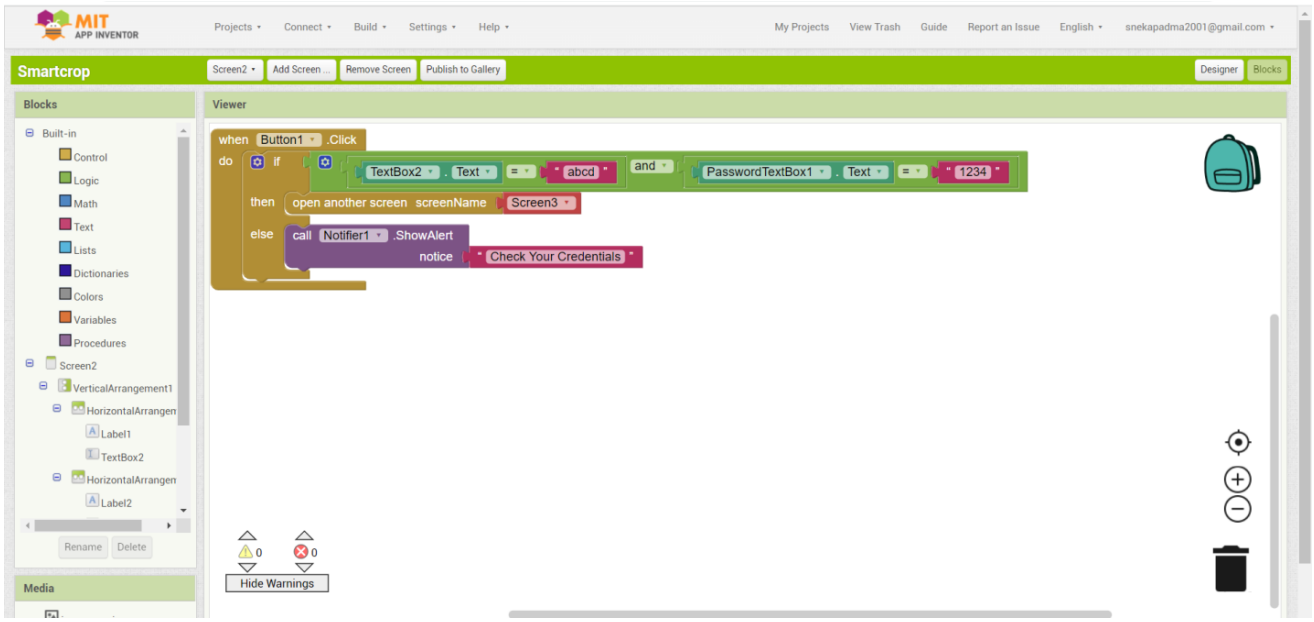
Customize the APP interface to display the values:

Blocks(BACKEND)

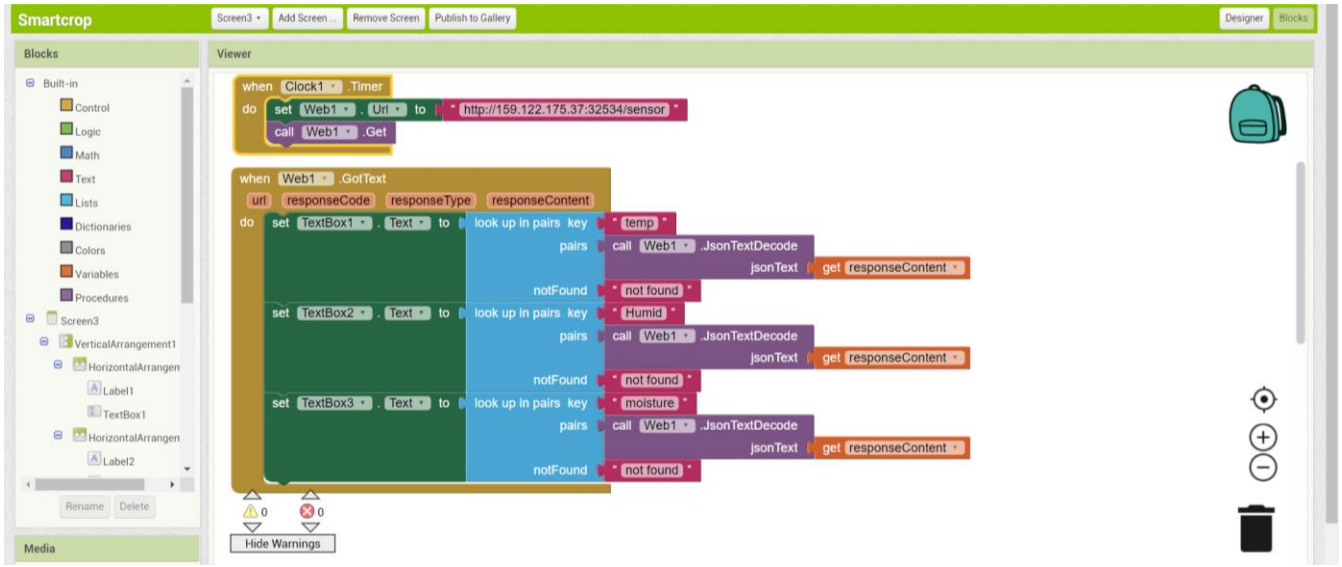
Screen-1:



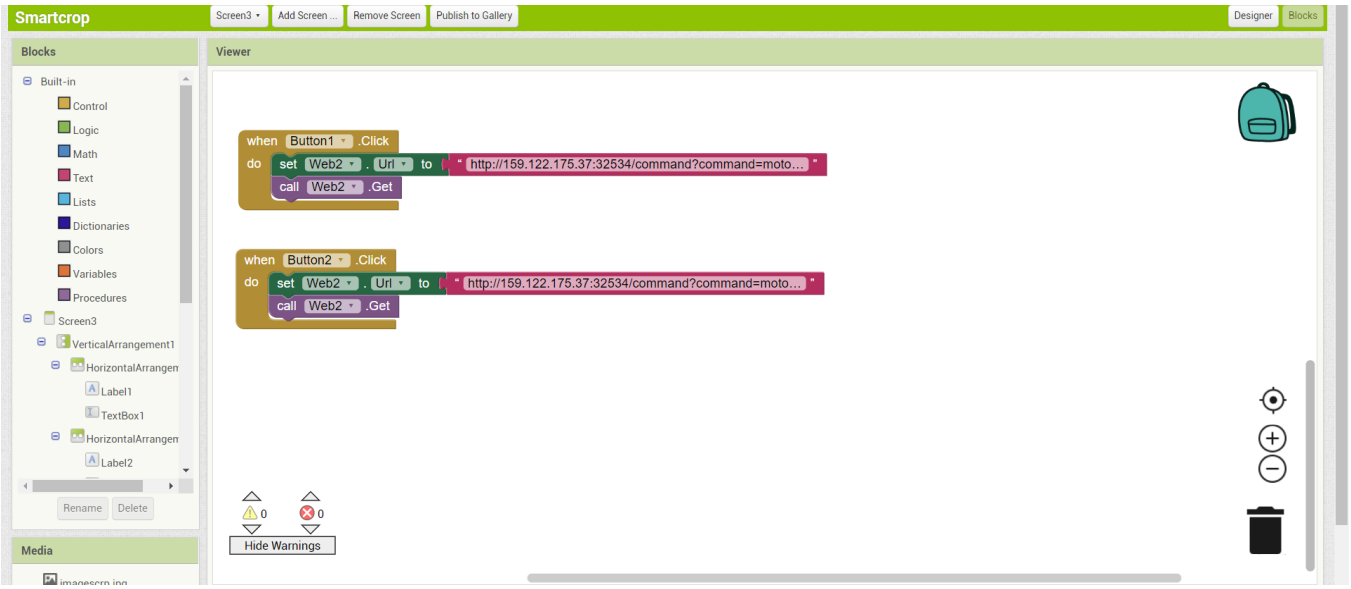
Screen-2:



Screen-3 :

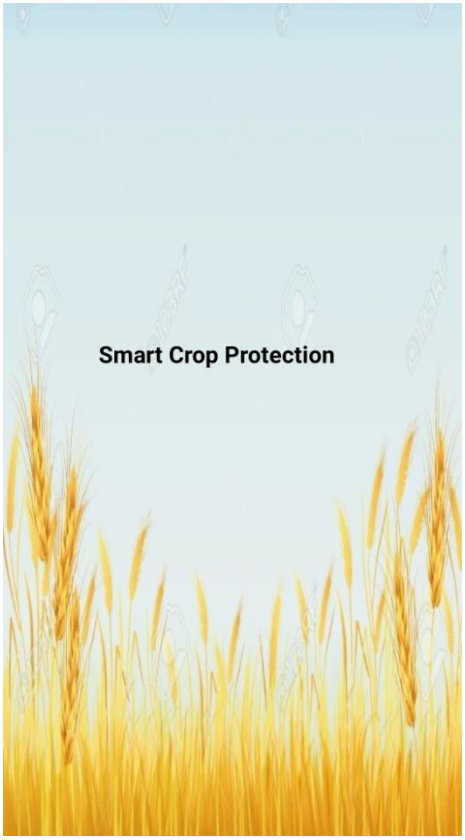


Screen-3 Blocks Button:



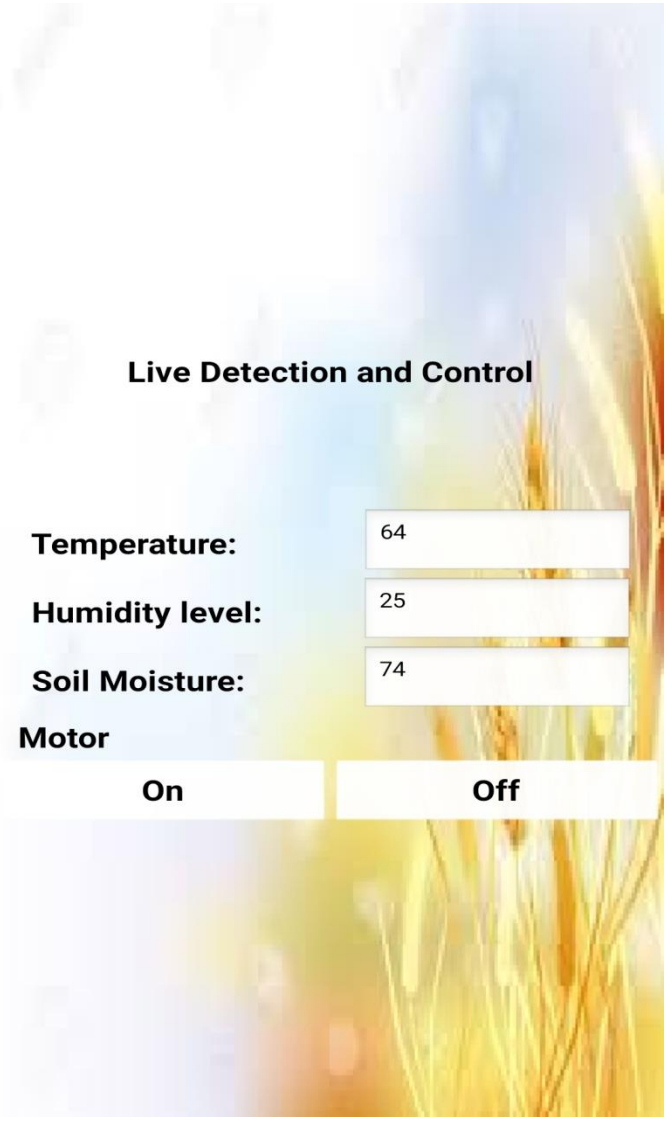
MIT App Invertor Output-Mobile Phone:

Screen-1:



Screen-2:

Screen-3:


Live Detection and Control
Temperature:
64
Humidity level:
25
Soil Moisture:
74
Motor
On
Off