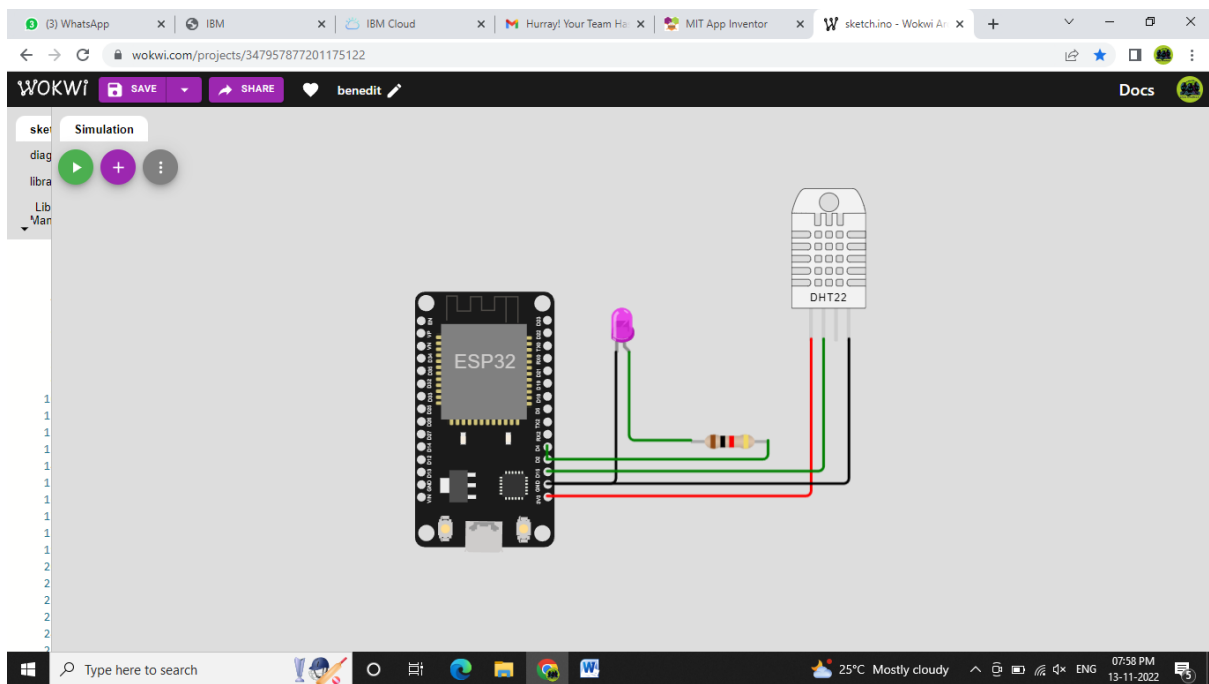


SPRINT DELIVERY-1

DATE	15 November 2022
TEAM ID	PNT2022TMID52158
PROJECT NAME	Smart Farmer - IoT Enabled Smart Farming Application

Connect sensors in ESP8266

▪ Circuit Diagram:



▪ This is the Connections of sensors in ESP8266

Develop a Python Code:

Code:

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

#Provide your IBM Watson Device Credentials
organization = "w9kxol"
deviceType = "123"
deviceId = "1234"
authMethod = "token"
authToken = "987654321"

# Initialize GPIO
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 an
event of type "greeting" 10 times
deviceCli.connect()

while True:
    #Get Sensor Data from DHT11

    temp=random.randint(90,110)
    Humid=random.randint(60,100)
    moist=random.randint(50,120)
    data = { 'temp' : temp, 'Humid': Humid , 'moist':moist}
    #print data
```

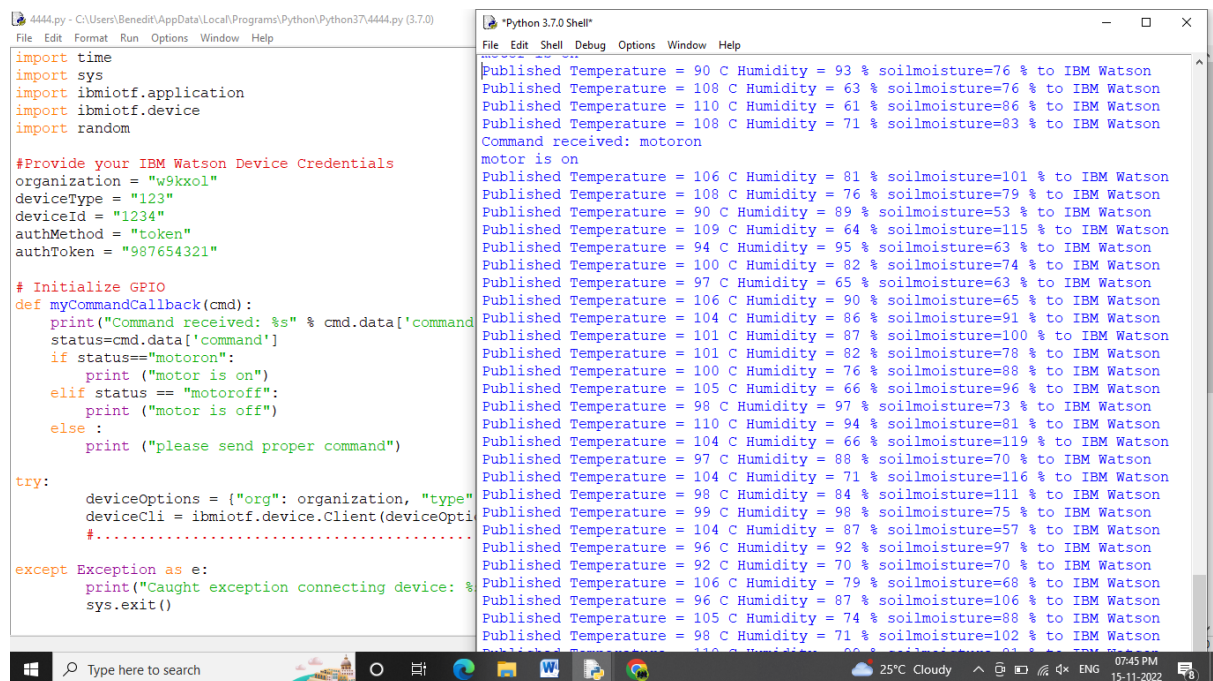
```
def myOnPublishCallback():
    print ("Published Temperature = %s C" % temp, "Humidity = %s %%" %
Humid,"soilmoisture=%s %%" % moist, "to IBM Watson")
```

```
    success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)
    if not success:
        print("Not connected to IoTF")
        time.sleep(10)
```

```
deviceCli.commandCallback = myCommandCallback
```

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

OUTPUT:



```
4444.py - C:\Users\Benedict\AppData\Local\Programs\Python\Python37\4444.py (3.7.0)
File Edit Format Run Options Window Help

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

#Provide your IBM Watson Device Credentials
organization = "w9kxol"
deviceType = "123"
deviceId = "1234"
authMethod = "token"
authToken = "987654321"

# Initialize GPIO
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"
deviceCli = ibmiotf.device.Client(deviceOpti
#.....
except Exception as e:
    print("Caught exception connecting device: %
sys.exit()
```

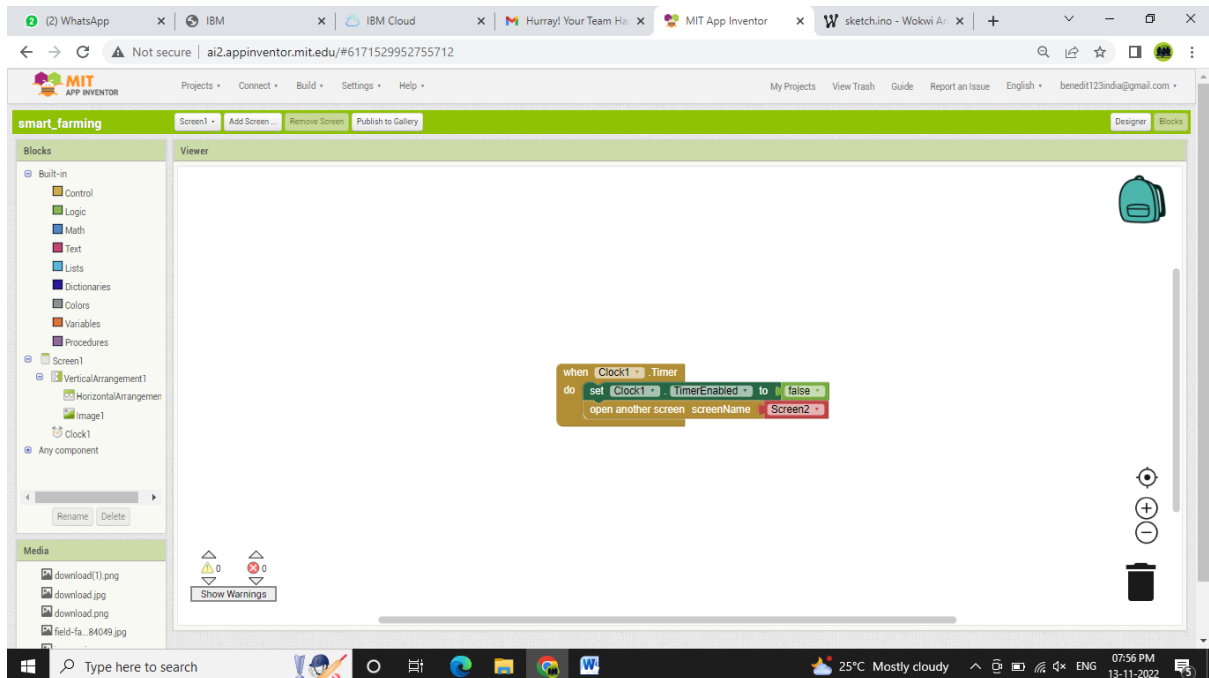
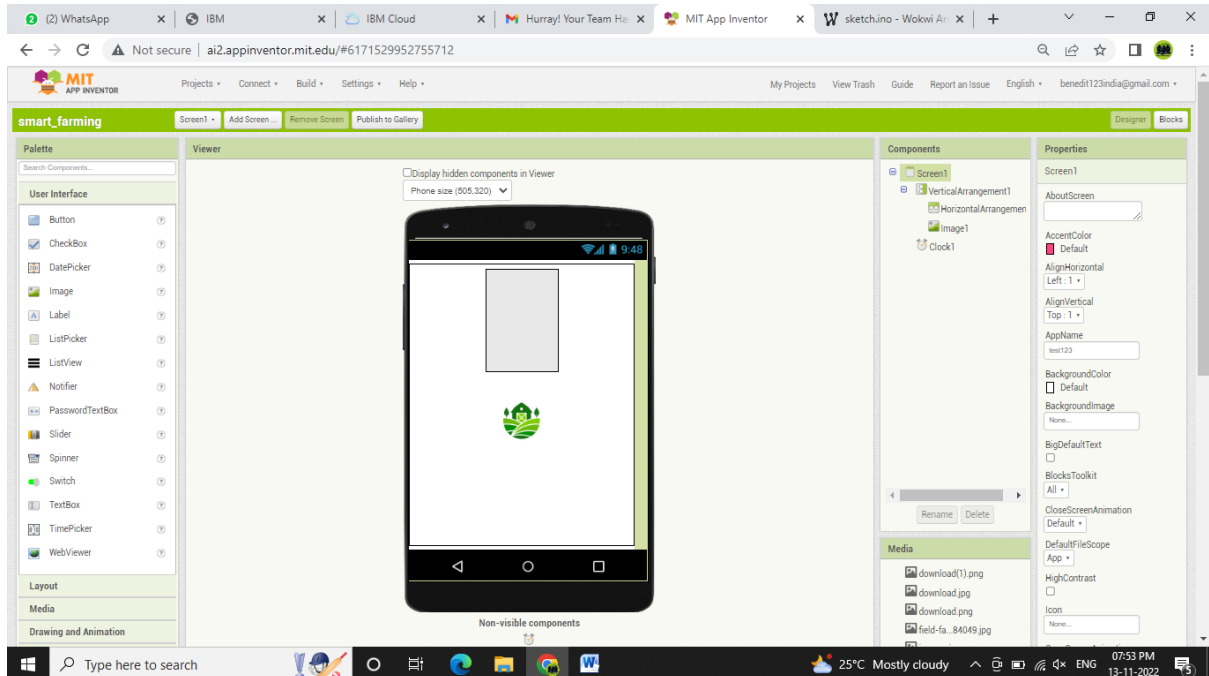
```
*Python 3.7.0 Shell*
File Edit Shell Debug Options Window Help

Published Temperature = 90 C Humidity = 93 % soilmoisture=76 % to IBM Watson
Published Temperature = 108 C Humidity = 63 % soilmoisture=76 % to IBM Watson
Published Temperature = 110 C Humidity = 61 % soilmoisture=86 % to IBM Watson
Published Temperature = 108 C Humidity = 71 % soilmoisture=83 % to IBM Watson
Command received: motoron
motor is on
Published Temperature = 106 C Humidity = 81 % soilmoisture=101 % to IBM Watson
Published Temperature = 108 C Humidity = 76 % soilmoisture=79 % to IBM Watson
Published Temperature = 90 C Humidity = 89 % soilmoisture=53 % to IBM Watson
Published Temperature = 109 C Humidity = 64 % soilmoisture=115 % to IBM Watson
Published Temperature = 94 C Humidity = 95 % soilmoisture=63 % to IBM Watson
Published Temperature = 100 C Humidity = 82 % soilmoisture=74 % to IBM Watson
Published Temperature = 97 C Humidity = 65 % soilmoisture=63 % to IBM Watson
Published Temperature = 106 C Humidity = 90 % soilmoisture=65 % to IBM Watson
Published Temperature = 104 C Humidity = 86 % soilmoisture=91 % to IBM Watson
Published Temperature = 101 C Humidity = 87 % soilmoisture=100 % to IBM Watson
Published Temperature = 101 C Humidity = 82 % soilmoisture=78 % to IBM Watson
Published Temperature = 100 C Humidity = 76 % soilmoisture=88 % to IBM Watson
Published Temperature = 105 C Humidity = 66 % soilmoisture=96 % to IBM Watson
Published Temperature = 98 C Humidity = 97 % soilmoisture=73 % to IBM Watson
Published Temperature = 110 C Humidity = 94 % soilmoisture=81 % to IBM Watson
Published Temperature = 104 C Humidity = 66 % soilmoisture=119 % to IBM Watson
Published Temperature = 97 C Humidity = 88 % soilmoisture=70 % to IBM Watson
Published Temperature = 104 C Humidity = 71 % soilmoisture=116 % to IBM Watson
Published Temperature = 98 C Humidity = 84 % soilmoisture=111 % to IBM Watson
Published Temperature = 99 C Humidity = 98 % soilmoisture=75 % to IBM Watson
Published Temperature = 104 C Humidity = 87 % soilmoisture=57 % to IBM Watson
Published Temperature = 96 C Humidity = 92 % soilmoisture=97 % to IBM Watson
Published Temperature = 92 C Humidity = 70 % soilmoisture=70 % to IBM Watson
Published Temperature = 106 C Humidity = 79 % soilmoisture=68 % to IBM Watson
Published Temperature = 96 C Humidity = 87 % soilmoisture=106 % to IBM Watson
Published Temperature = 105 C Humidity = 74 % soilmoisture=88 % to IBM Watson
Published Temperature = 98 C Humidity = 71 % soilmoisture=102 % to IBM Watson
```

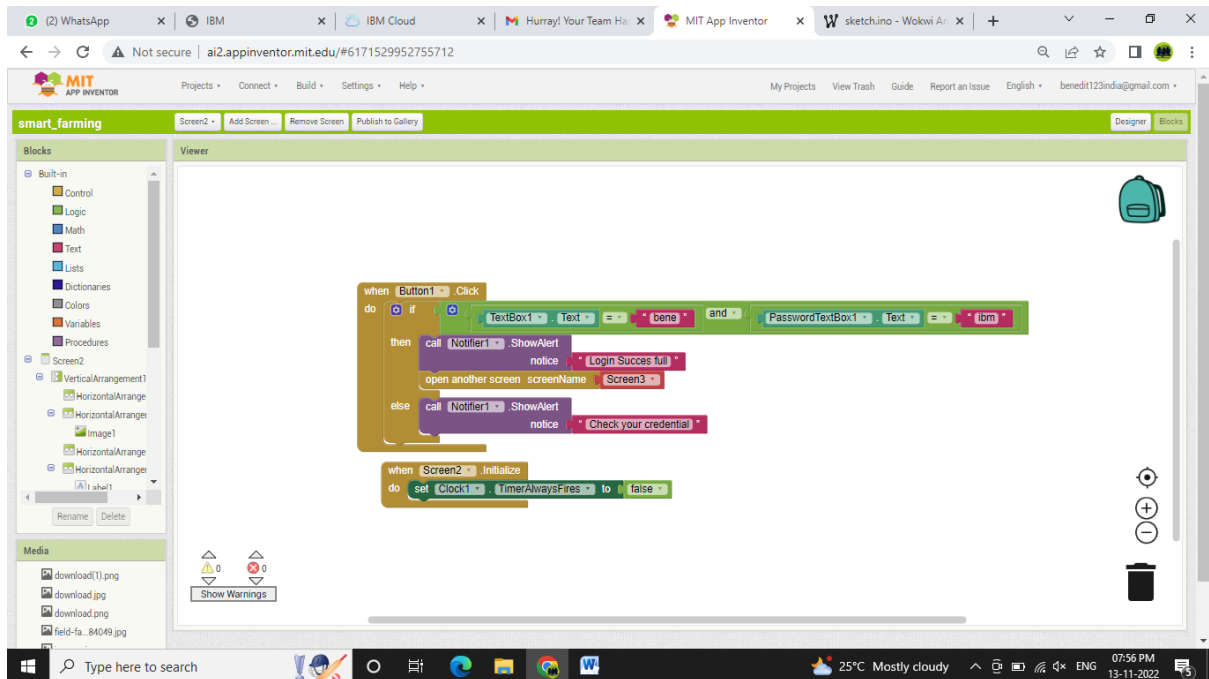
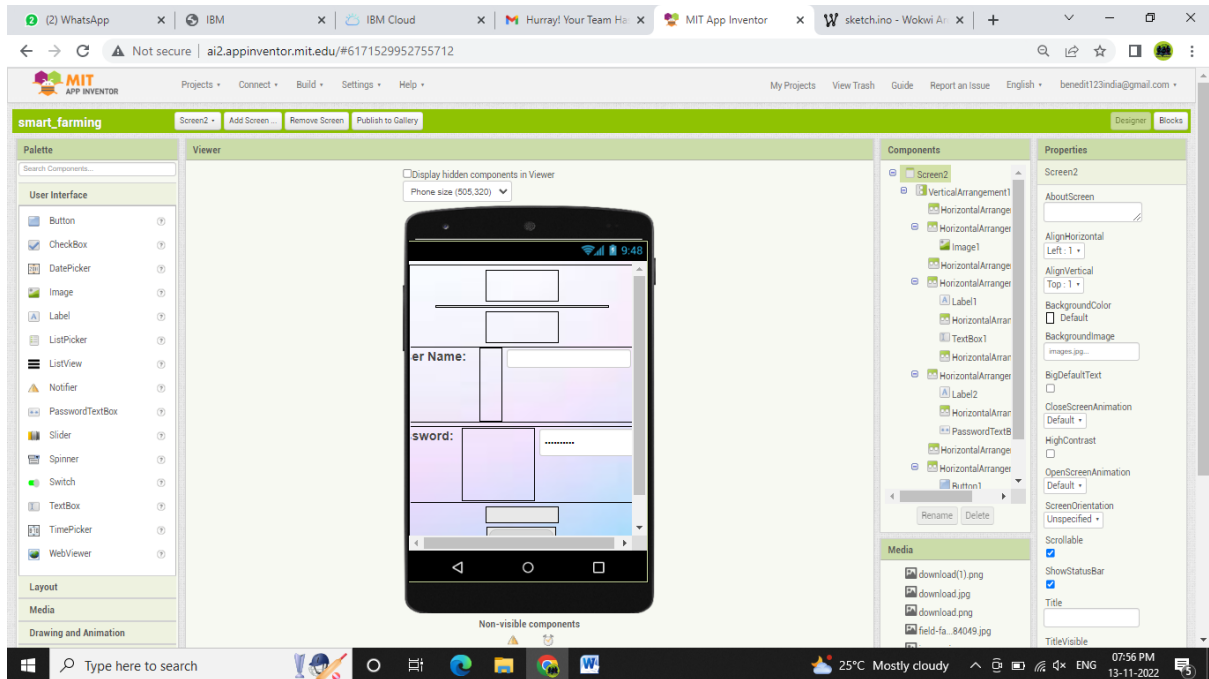
- This is the output for python code

Develop an application with MIT Appinventor:

■ Mobile App Opening page:



■ Mobile App Login page :



JIRA Software Sprint Planning:

The screenshot shows the Jira Software interface for a project named "Smart Farmer - IoT Enabled Smart Farming Application". The left sidebar contains navigation options: PLANNING (Roadmap, Backlog, Board), DEVELOPMENT (Code), and Project pages. The main area displays the "All sprints" view. It features a search bar, filters (Epic, Sprint), and a "GROUP BY" dropdown. The board is divided into columns: "TO DO 3 ISSUES", "IN PROGRESS 2 ISSUES", and "DONE 1 ISSUE". Issues are represented by cards with labels like "NODE RED" and "APPLICATION DEVELOPMENT". A "Quickstart" button is visible at the bottom. On the right, the "Insights" panel shows "Sprint progress" (100% done), "Sprint burndown", and "Epic progress". The bottom status bar indicates the system time as 01:25 PM on 13-11-2022.

The screenshot shows the Jira Software interface for the same project, now in the "Backlog" view. The left sidebar is identical. The main area displays the "Backlog" view with a search bar, filters (Epic), and a "GROUP BY" dropdown. The backlog is organized into sprints: "SFIEFA Sprint 1" (25 Oct - 29 Oct, 2 issues) and "SFIEFA Sprint 2" (31 Oct - 7 Nov, 2 issues). Issues are listed with their status (e.g., "DONE", "IN PROGRESS") and labels. A "Quickstart" button is visible at the bottom. On the right, the "Insights" panel shows "Sprint commitment" (16 points), "No average", and "Issue type breakdown". The bottom status bar indicates the system time as 01:26 PM on 13-11-2022.

IBM | Service Details | IBM Watson IoT | Node-RED: node | sketchino - Wol | WhatsApp | Smart Farmer - IoT En...

benben123456.atlassian.net/jira/software/projects/SFIEFA/boards/1/roadmap

Jira Software | Your work | Projects | Filters | Dashboards | People | Apps | Create

Smart Farmer - IoT En...
Software project

PLANNING

- Roadmap
- Backlog
- Board

DEVELOPMENT

- Code
- Project pages
- Add shortcut
- Project settings

You're in a team-managed project
Learn more

Projects / Smart Farmer - IoT Enabled Smart Farming Application

Roadmap

Give feedback | Share | Export

Search | BR | G | S | V | Status category | Epic

Sprints

- SFIEFA-8 Connection
- SFIEFA-9 Connection
- SFIEFA-10 Application development
- SFIEFA-11 Node Red
- SFIEFA-12 Testing
- + Create Epic

Today | Weeks | Months | Quarters | Quickstart

25°C Cloudy | 01:27 PM | 13-11-2022

