# Smart Farmer - IoT Enabled Smart Farming Application

# **SPRINT-4**

Team ID	PNT2022TMID35924
Project name	Smart farmer- IoT
	enabled smart farming

In this session, we have developed the App for our project using MIT App Inventor, which displays the following field parameters – Moisture, Temperature, Humidity, Flame, Nitrogen, Phosphorous, Potassium. It also displays the Motor ON and OFF switches to control the water sprinklers located in the field from our Mobile itself. It consists of three screens whose screenshots are attached below:

# **SCREEN 1:**

# **Front End:**



# **Back End:**

```
when Clock1 · .Timer

do set Clock1 · . TimerEnabled · to false ·
open another screen screenName Screen2 ·
```

The above screenshot shows our screen 1. When the App is opened, this screen will be displayed for 3s then it moves to our login page. We have used Vertical Arrangement, Horizontal Arrangement and Clock components.

# **SCREEN 2:**

# **Front End:**



# **Back End:**

```
when Button1 v. Click

do if TextBox1 v. Text v = v * ibm v and v PasswordTextBox1 v. Text v = v * ibm v then call Notifier1 v. ShowAlert notice v Login successful v open another screen screenName vscreen3 v else call Notifier1 v. ShowAlert notice v Check your credentials v when Screen2 v. Initialize do set Clock1 v. TimerEnabled v to false v
```

This shows our screen 2. In this, we have designed our login page. When the user feeds the correct credentials, they will be moved to the next page. And if the login credentials are wrong, they will be stuck in the same page until they enter the correct details. We have used vertical and horizontal arrangements, labels, textbox, password textbox, button, notifier and clock components in this.

## **SCREEN 3:**

## **Front End:**



# **Back End:**

```
when Clock1 - .Timer
do set Web1 . Url to ( "http://169.51.205.41:32484/data "
   call Web1 .Get
when Web1 .GotText
[url] [responseCode] [responseType] [responseContent]
do set Label12 . Text to look up in pairs key potassium "
                                        pairs call Web2 .JsonTextDecode
                                                                  jsonText | get responseContent •
                                     notFound (not found)
   set Label13 . Text to look up in pairs key flame
                                        pairs call (Web2 · .JsonTextDecode
                                                                  jsonText ( get responseContent •
                                     notFound | " (not found "
   set Label16 . Text to look up in pairs key temp
                                        pairs call Web2 JsonTextDecode
                                                                 jsonText get responseContent •
                                     notFound " not found "
    set Label15 . Text to look up in pairs key "nitrogen"
                                        pairs call Web2 JsonTextDecode
                                                                 jsonText get responseContent •
                                     notFound | " uday "
    set Label5 . Text to look up in pairs key humidity
                                        pairs call Web2 JsonTextDecode
```

```
" uday '
 set Label5 . Text to look up in pairs key
                                               " (humidity
                                               call Web2 .JsonTextDecode
                                                                             get responseContent
                                    notFound
                                                " not found
 set (Label8 * ). Text * to | look up in pairs key |
                                               " moisture "
                                        pairs call Web2 JsonTextDecode
                                                                             get responseContent
                                                                   jsonText
                                               " not found "
                                    notFound
 set Label14 . Text to look up in pairs key
                                                " phosporus
                                                call Web2 .JsonTextDecode
                                                                              get responseContent
                                                                    jsonText
                                     notFound
                                                " not found "
when Button1 .Click
  set (Web2 * ). Url *
                             " http://169.51.205.41:32484/command?command=motoron
    call Web2 .Get
when Button2 .Click
do set Web2 . Url .
                             http://169.51.205.41:32484/command?command=motor.
    call Web2 .Get
```

This Screen is our final one. This displays all our field parameters, the inputs of which are delivered by the Node-RED Software. When the moisture reading drops below a certain level or the flame threshold is reached, an 'alert' SMS is sent to the user. The Motor ON/OFF switches are also used to control the field irrigation system. We have used vertical and horizontal arrangement, label, button, clock and web components.

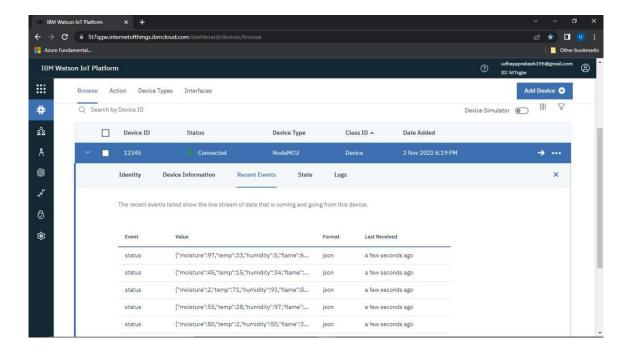
#### **OUTPUT:**

The following output flow is observed:

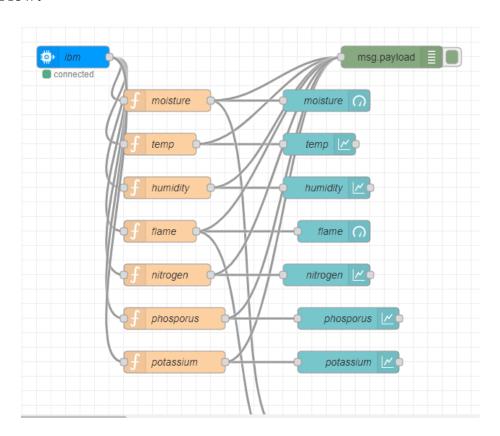
# **Python IDLE:**

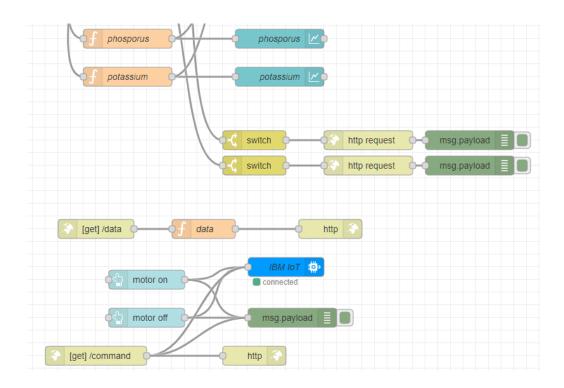
```
Python 3.8.0 Shell*
File Edit Shell Debug Options Window Help 13, phosporus: 03, pocassium: 30)
Published data Successfully: %s ('moisture': 5, 'temp': 35, 'humidity': 45, 'flame': 93, 'nitrogen': 3
9, 'phosporus': 5, 'potassium': 61)
Message received from IBM IoT Platform: motoron
Motor is switched ON
Published data Successfully: %s {'moisture': 45, 'temp': 54, 'humidity': 17, 'flame': 59, 'nitrogen':
0, 'phosporus': 73, 'potassium': 29}
Published data Successfully: %s {'moisture': 45, 'temp': 21, 'humidity': 83, 'flame': 16, 'nitrogen':
30, 'phosporus': 40, 'potassium': 22}
Published data Successfully: %s {'moisture': 89, 'temp': 23, 'humidity': 34, 'flame': 50, 'nitrogen':
54, 'phosporus': 31, 'potassium': 43}
Published data Successfully: %s ('moisture': 95, 'temp': 7, 'humidity': 0, 'flame': 73, 'nitrogen': 16
  'phosporus': 78, 'potassium': 89}
Published data Successfully: %s ('moisture': 80, 'temp': 62, 'humidity': 69, 'flame': 88, 'nitrogen':
32, 'phosporus': 16, 'potassium': 59}
Published data Successfully: %s {'moisture': 11, 'temp': 41, 'humidity': 69, 'flame': 43, 'nitrogen':
2, 'phosporus': 30, 'potassium': 6}
Published data Successfully: %s ('moisture': 78, 'temp': 19, 'humidity': 9, 'flame': 53, 'nitrogen': 1
4, 'phosporus': 91, 'potassium': 40}
Published data Successfully: %s {'moisture': 66, 'temp': 83, 'humidity': 55, 'flame': 0, 'nitrogen': 7
9, 'phosporus': 16, 'potassium': 79}
Published data Successfully: %s {'moisture': 60, 'temp': 50, 'humidity': 79, 'flame': 34, 'nitrogen':
    'phosporus': 86, 'potassium': 51}
Published data Successfully: %s ('moisture': 7, 'temp': 50, 'humidity': 39, 'flame': 100, 'nitrogen':
34, 'phosporus': 76, 'potassium': 55)
Published data Successfully: %s ('moisture': 17, 'temp': 54, 'humidity': 23, 'flame': 97, 'nitrogen':
44, 'phosporus': 92, 'potassium': 91}
Published data Successfully: %s ('moisture': 61, 'temp': 71, 'humidity': 92, 'flame': 67, 'nitrogen':
41, 'phosporus': 32, 'potassium': 19}
```

# **IBM Cloud:**

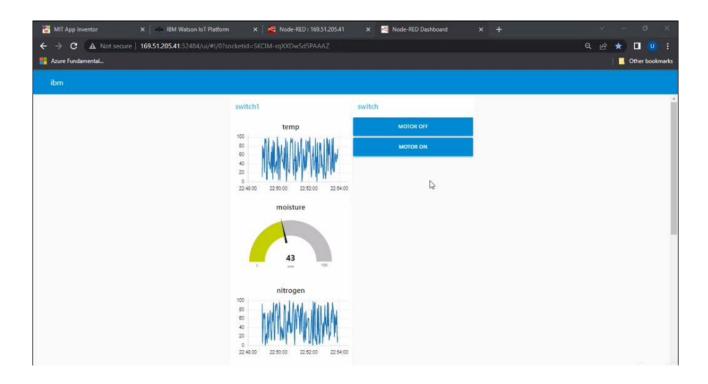


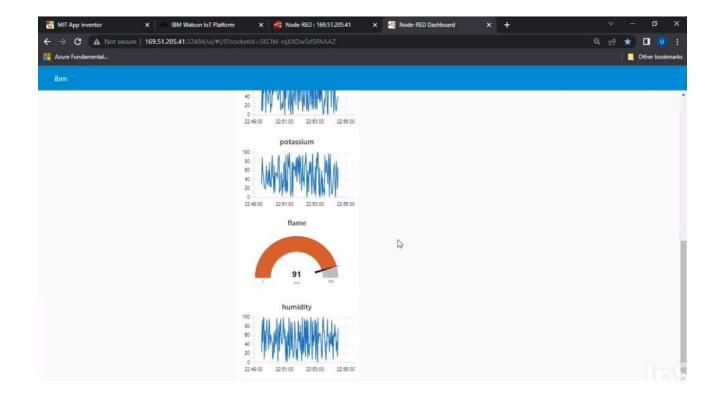
# **Node-RED Flow:**





# Web-UI:

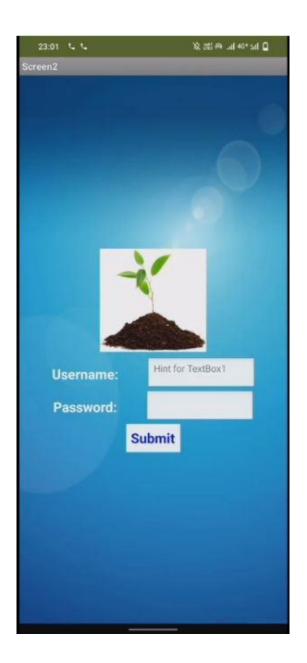




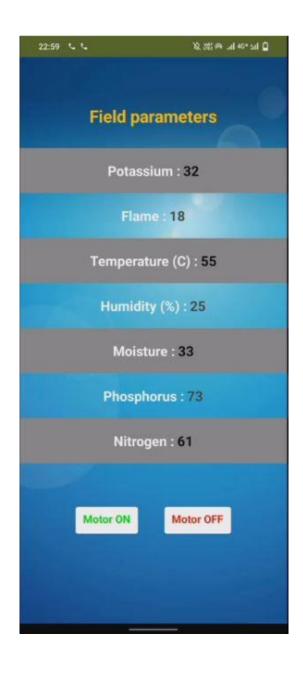
# App Display: Screen1:



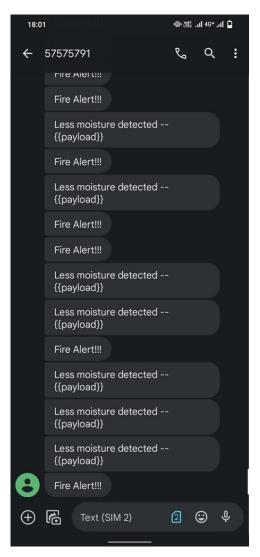
# Screen 2:



# Screen 3:



# SMS 'Alert':



# **Motor ON:**

Message received from IBM IoT Platform: motoron Motor is switched ON

# **Motor OFF:**

Message received from IBM IoT Platform: motoroff Motor is switched OFF