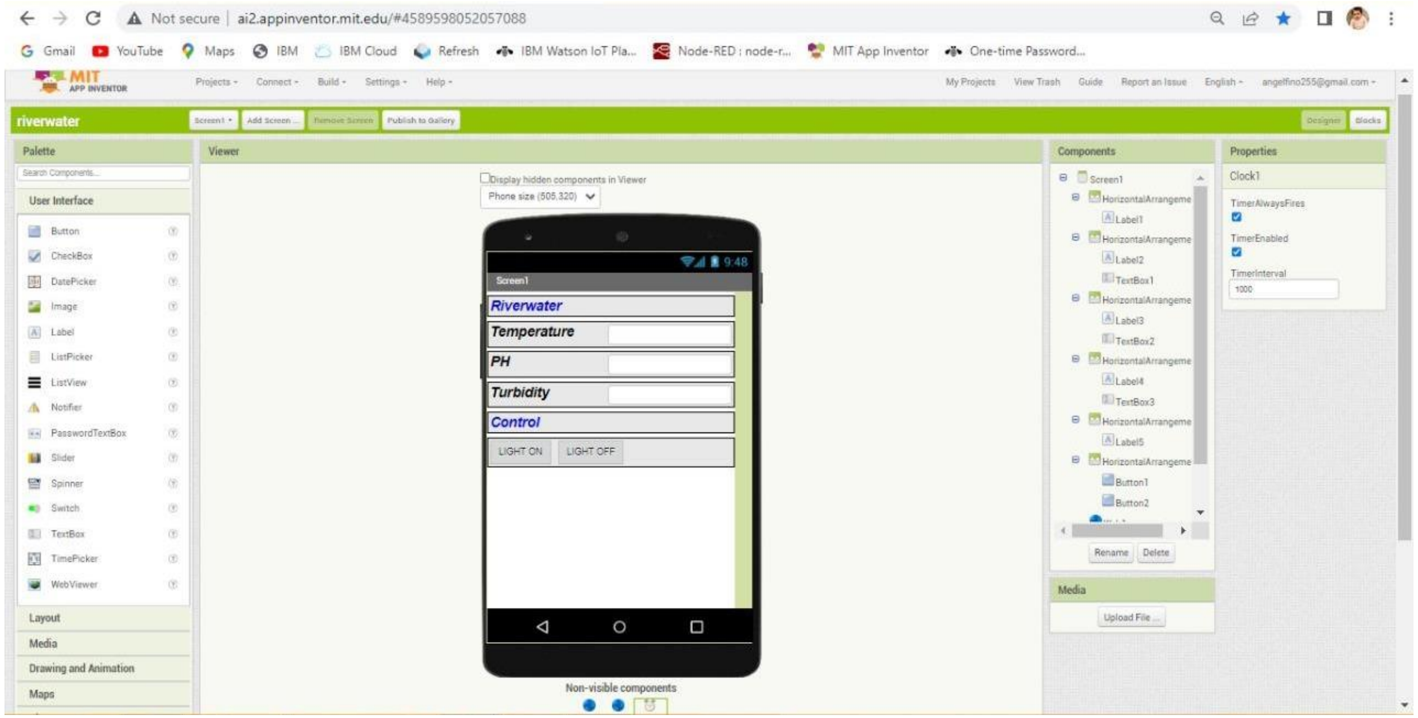
PROJECT DEVELOPMENT PHASE
SPRINT-3

DATE	12 NOVEMBER 2022
TEAM ID	PNT2022TMID05914
PROJECT TITLE	Real-time river water quality monitoring and control system

USN-11

As a user ,I can design the front end in MIT app inventor.



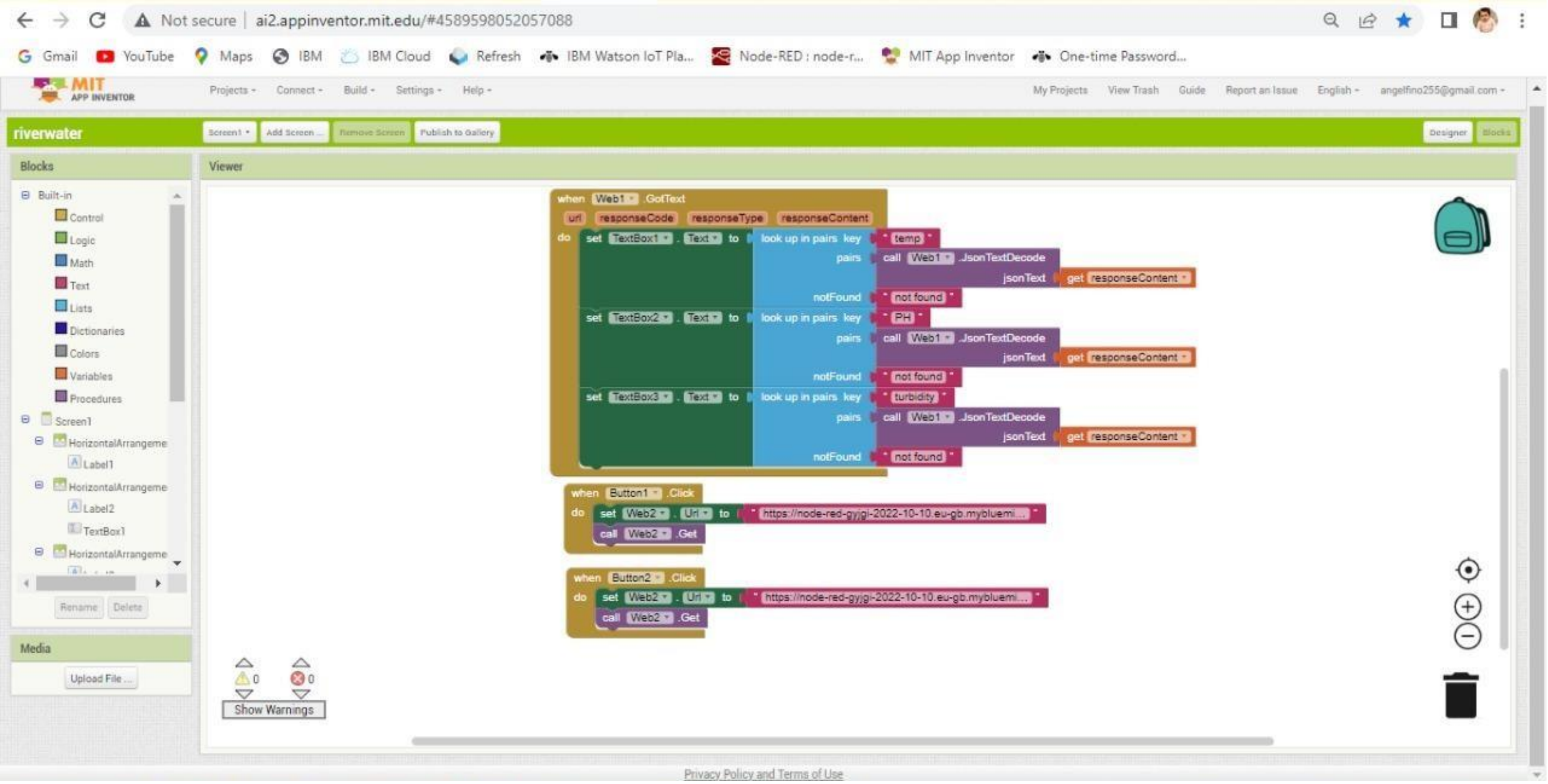
TimerAlwaysFires ☑

TimerEnabled ☑

TimerInterval 1000

USN-12

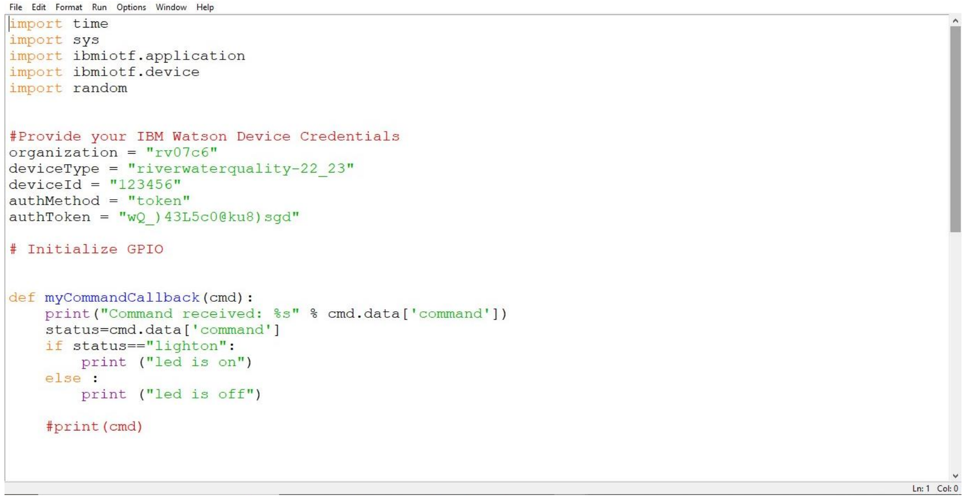
As a user ,I can design the back end(blocks) in MIT app inventor



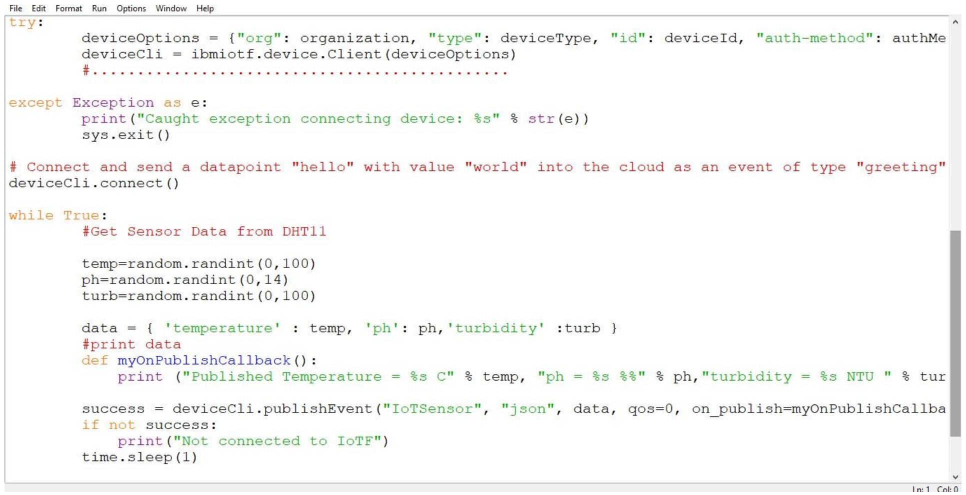
```
when Web1 .GotText
  url  responseCode  responseType  responseContent
do  set TextBox1 . Text to  look up in pairs  key  " temp "
                                               pairs  call Web1 .JsonTextDecode
                                                                  jsonText  get responseContent
                                               notFound  " not found "
    set TextBox2 . Text to  look up in pairs  key  " PH "
                                               pairs  call Web1 .JsonTextDecode
                                                                  jsonText  get responseContent
                                               notFound  " not found "
    set TextBox3 . Text to  look up in pairs  key  " turbidity "
                                               pairs  call Web1 .JsonTextDecode
                                                                  jsonText  get responseContent
                                               notFound  " not found "

when Button1 .Click
do  set Web2 . Url to  " https://node-red-gyjgi-2022-10-10.eu-gb.mybluemi... "
    call Web2 .Get

when Button2 .Click
do  set Web2 . Url to  " https://node-red-gyjgi-2022-10-10.eu-gb.mybluemi... "
    call Web2 .Get
```

USN-13

As a user ,I can develop the python script.



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


#Provide your IBM Watson Device Credentials
organization = "rv07c6"
deviceType = "riverwaterquality-22_23"
deviceId = "123456"
authMethod = "token"
authToken = "wQ_)43L5c0@ku8)sgd"

# Initialize GPIO


def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="lighton":
        print ("led is on")
    else :
        print ("led is off")

    #print(cmd)
```



```python
try:
        deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMe
        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"
deviceCli.connect()

while True:
        #Get Sensor Data from DHT11

        temp=random.randint(0,100)
        ph=random.randint(0,14)
        turb=random.randint(0,100)

        data = { 'temperature' : temp, 'ph': ph,'turbidity' :turb }
        #print data
        def myOnPublishCallback():
            print ("Published Temperature = %s C" % temp, "ph = %s %%" % ph,"turbidity = %s NTU " % tur

        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallba
        if not success:
            print("Not connected to IoTF")
        time.sleep(1)
```

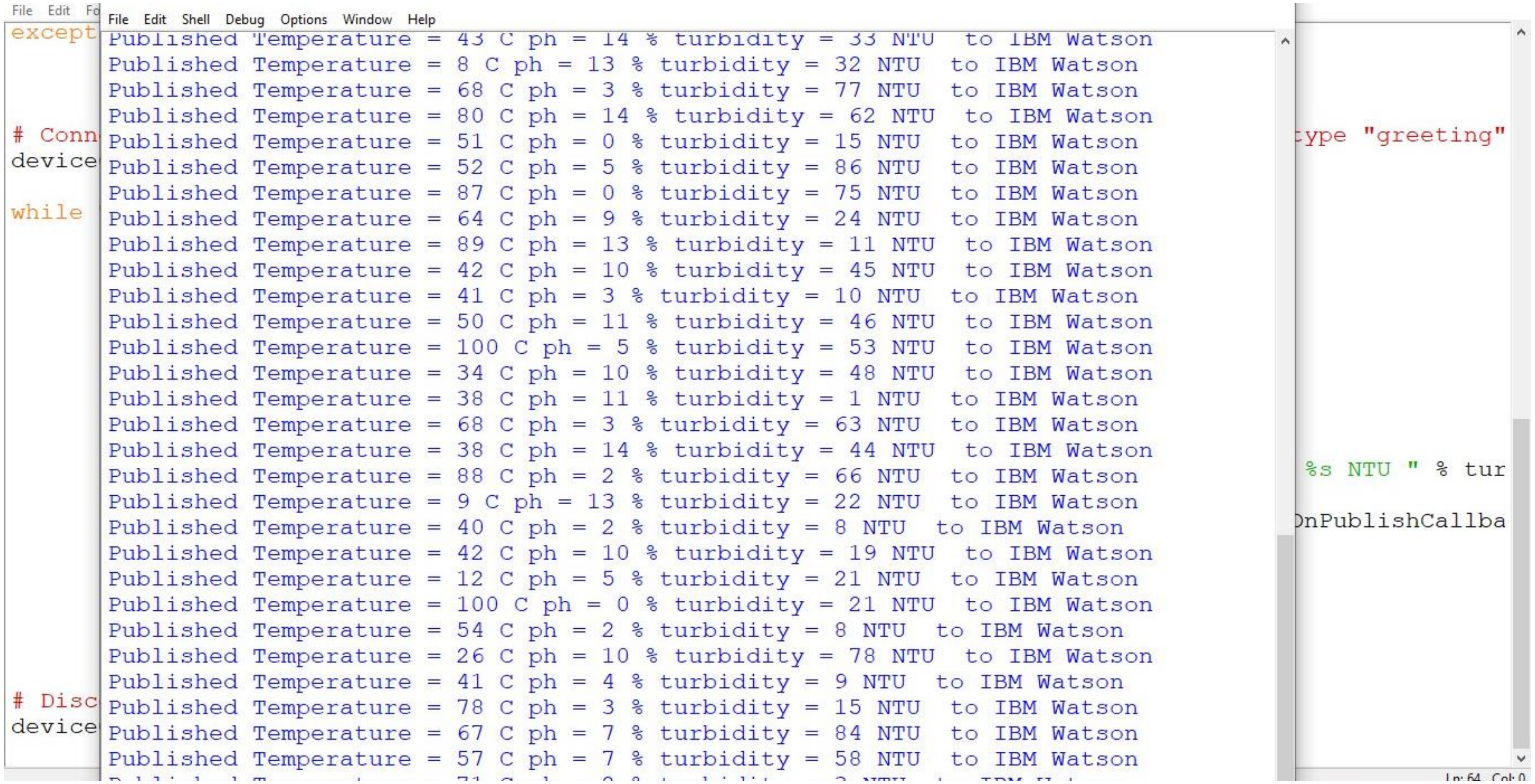
PROGRAM:

```
import ibmiotf.device
import random

#Provide your IBM Watson Device Credentials
organization = "rv07c6"
deviceType = "riverwaterquality-22_23"
deviceId = "123456"
authMethod = "token"
authToken = "wQ_)43L5c0@ku8)sgd"

# Initialize GPIO


def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="lighton":
        print ("led is on")
    else :
        print ("led 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 DHT11

    temp=random.randint(0,100)
    ph=random.randint(0,14)
    turb=random.randint(0,100)

    data = { 'temperature' : temp, 'ph': ph,'turbidity' :turb }
    #print data
    def myOnPublishCallback():
      print ("Published Temperature = %s C" % temp, "ph = %s %%" % ph,"turbidity = %s NTU " % turb ,"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()
```

USN-14

As a user,I can get the output of the program with the parameters

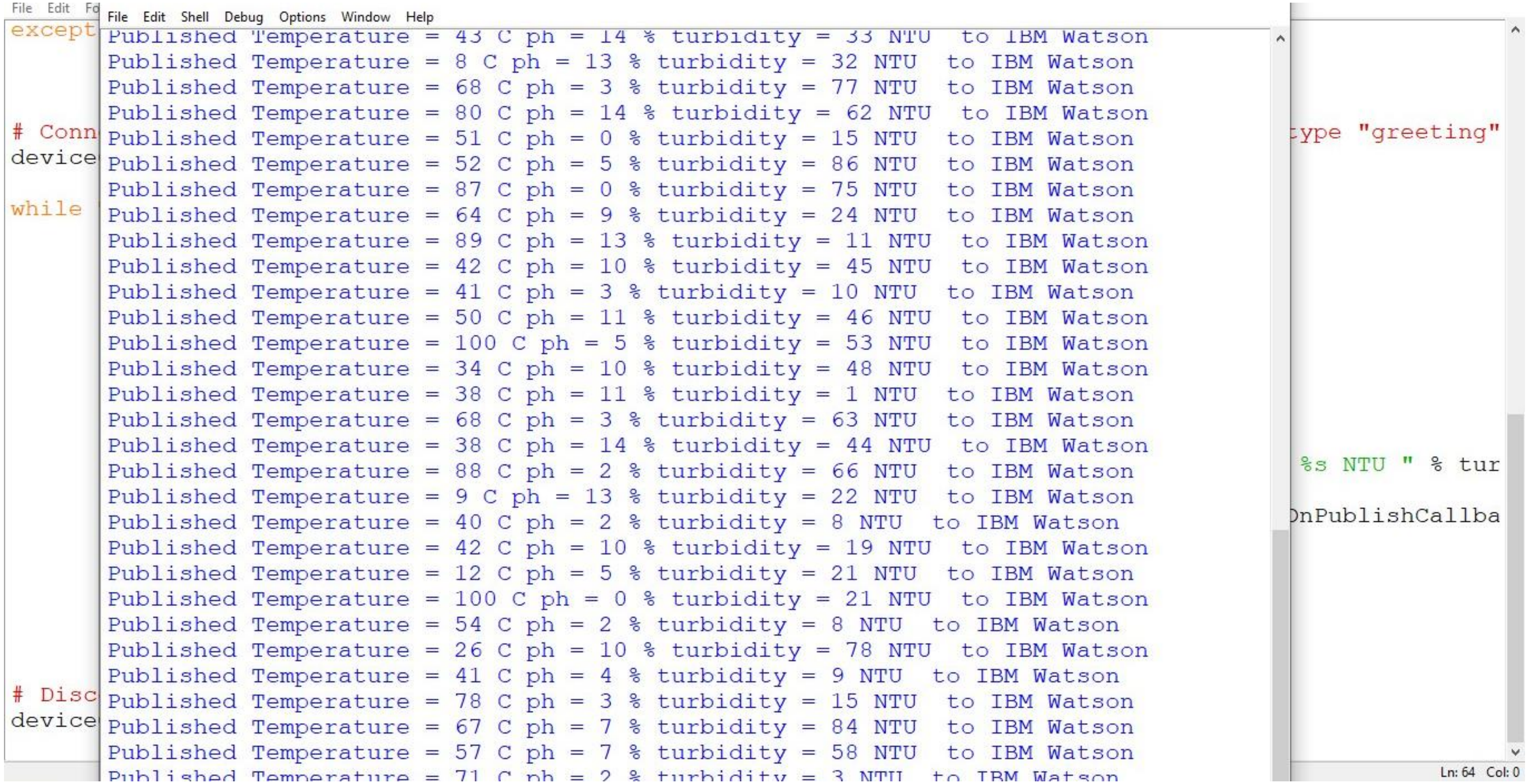


```
except
# Conn
device
while
# Disc
device
type "greeting"
%s NTU " % tur
OnPublishCallba
```

```
Published Temperature = 43 C ph = 14 % turbidity = 33 NTU  to IBM Watson
Published Temperature = 8 C ph = 13 % turbidity = 32 NTU  to IBM Watson
Published Temperature = 68 C ph = 3 % turbidity = 77 NTU  to IBM Watson
Published Temperature = 80 C ph = 14 % turbidity = 62 NTU  to IBM Watson
Published Temperature = 51 C ph = 0 % turbidity = 15 NTU  to IBM Watson
Published Temperature = 52 C ph = 5 % turbidity = 86 NTU  to IBM Watson
Published Temperature = 87 C ph = 0 % turbidity = 75 NTU  to IBM Watson
Published Temperature = 64 C ph = 9 % turbidity = 24 NTU  to IBM Watson
Published Temperature = 89 C ph = 13 % turbidity = 11 NTU  to IBM Watson
Published Temperature = 42 C ph = 10 % turbidity = 45 NTU  to IBM Watson
Published Temperature = 41 C ph = 3 % turbidity = 10 NTU  to IBM Watson
Published Temperature = 50 C ph = 11 % turbidity = 46 NTU  to IBM Watson
Published Temperature = 100 C ph = 5 % turbidity = 53 NTU  to IBM Watson
Published Temperature = 34 C ph = 10 % turbidity = 48 NTU  to IBM Watson
Published Temperature = 38 C ph = 11 % turbidity = 1 NTU  to IBM Watson
Published Temperature = 68 C ph = 3 % turbidity = 63 NTU  to IBM Watson
Published Temperature = 38 C ph = 14 % turbidity = 44 NTU  to IBM Watson
Published Temperature = 88 C ph = 2 % turbidity = 66 NTU  to IBM Watson
Published Temperature = 9 C ph = 13 % turbidity = 22 NTU  to IBM Watson
Published Temperature = 40 C ph = 2 % turbidity = 8 NTU  to IBM Watson
Published Temperature = 42 C ph = 10 % turbidity = 19 NTU  to IBM Watson
Published Temperature = 12 C ph = 5 % turbidity = 21 NTU  to IBM Watson
Published Temperature = 100 C ph = 0 % turbidity = 21 NTU  to IBM Watson
Published Temperature = 54 C ph = 2 % turbidity = 8 NTU  to IBM Watson
Published Temperature = 26 C ph = 10 % turbidity = 78 NTU  to IBM Watson
Published Temperature = 41 C ph = 4 % turbidity = 9 NTU  to IBM Watson
Published Temperature = 78 C ph = 3 % turbidity = 15 NTU  to IBM Watson
Published Temperature = 67 C ph = 7 % turbidity = 84 NTU  to IBM Watson
Published Temperature = 57 C ph = 7 % turbidity = 58 NTU  to IBM Watson
Published Temperature = 71 C ph = 0 % turbidity = 3 NTU  to IBM Watson
```

USN-15

As a user , I can get the commands in the output when the buttons are pressed.

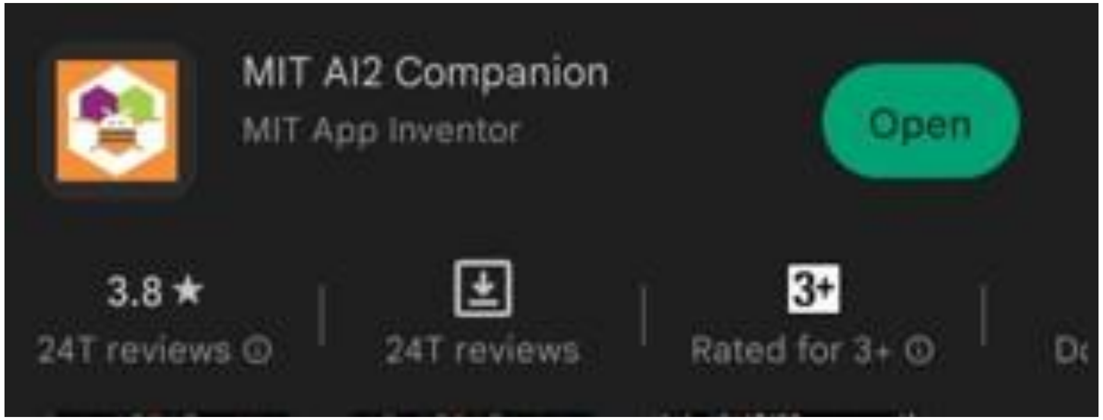


```
except
# Conn
device
while
# Disc
device
ype "greeting"
%s NTU " % tur
OnPublishCallba
```

```
Published Temperature = 43 C ph = 14 % turbidity = 33 NTU  to IBM Watson
Published Temperature = 8 C ph = 13 % turbidity = 32 NTU  to IBM Watson
Published Temperature = 68 C ph = 3 % turbidity = 77 NTU  to IBM Watson
Published Temperature = 80 C ph = 14 % turbidity = 62 NTU  to IBM Watson
Published Temperature = 51 C ph = 0 % turbidity = 15 NTU  to IBM Watson
Published Temperature = 52 C ph = 5 % turbidity = 86 NTU  to IBM Watson
Published Temperature = 87 C ph = 0 % turbidity = 75 NTU  to IBM Watson
Published Temperature = 64 C ph = 9 % turbidity = 24 NTU  to IBM Watson
Published Temperature = 89 C ph = 13 % turbidity = 11 NTU  to IBM Watson
Published Temperature = 42 C ph = 10 % turbidity = 45 NTU  to IBM Watson
Published Temperature = 41 C ph = 3 % turbidity = 10 NTU  to IBM Watson
Published Temperature = 50 C ph = 11 % turbidity = 46 NTU  to IBM Watson
Published Temperature = 100 C ph = 5 % turbidity = 53 NTU  to IBM Watson
Published Temperature = 34 C ph = 10 % turbidity = 48 NTU  to IBM Watson
Published Temperature = 38 C ph = 11 % turbidity = 1 NTU  to IBM Watson
Published Temperature = 68 C ph = 3 % turbidity = 63 NTU  to IBM Watson
Published Temperature = 38 C ph = 14 % turbidity = 44 NTU  to IBM Watson
Published Temperature = 88 C ph = 2 % turbidity = 66 NTU  to IBM Watson
Published Temperature = 9 C ph = 13 % turbidity = 22 NTU  to IBM Watson
Published Temperature = 40 C ph = 2 % turbidity = 8 NTU  to IBM Watson
Published Temperature = 42 C ph = 10 % turbidity = 19 NTU  to IBM Watson
Published Temperature = 12 C ph = 5 % turbidity = 21 NTU  to IBM Watson
Published Temperature = 100 C ph = 0 % turbidity = 21 NTU  to IBM Watson
Published Temperature = 54 C ph = 2 % turbidity = 8 NTU  to IBM Watson
Published Temperature = 26 C ph = 10 % turbidity = 78 NTU  to IBM Watson
Published Temperature = 41 C ph = 4 % turbidity = 9 NTU  to IBM Watson
Published Temperature = 78 C ph = 3 % turbidity = 15 NTU  to IBM Watson
Published Temperature = 67 C ph = 7 % turbidity = 84 NTU  to IBM Watson
Published Temperature = 57 C ph = 7 % turbidity = 58 NTU  to IBM Watson
Published Temperature = 71 C ph = 2 % turbidity = 3 NTU  to IBM Watson
```

USN-16

As a user, I can download MIT AI2 companion app in my mobile



MIT AI2 Companion

MIT App Inventor

3.8 ★

24T reviews ⓘ

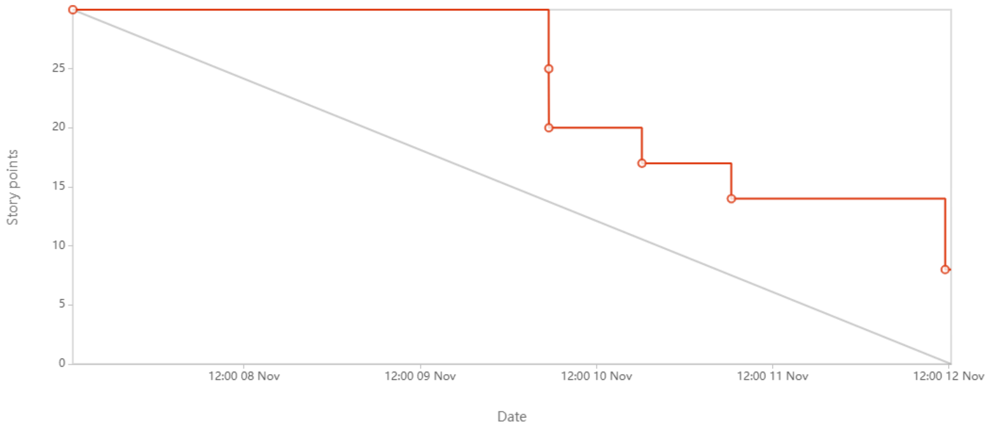
24T reviews

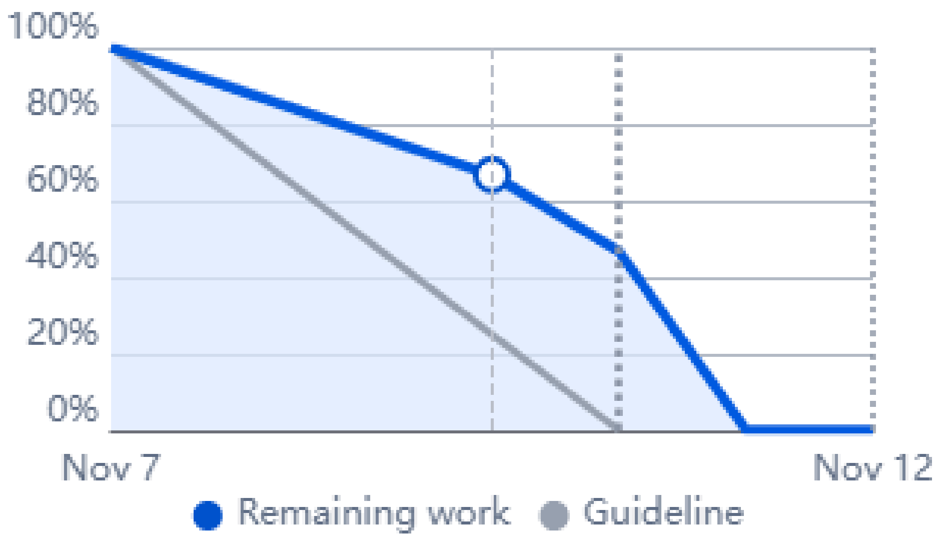
3+

Rated for 3+ ⓘ

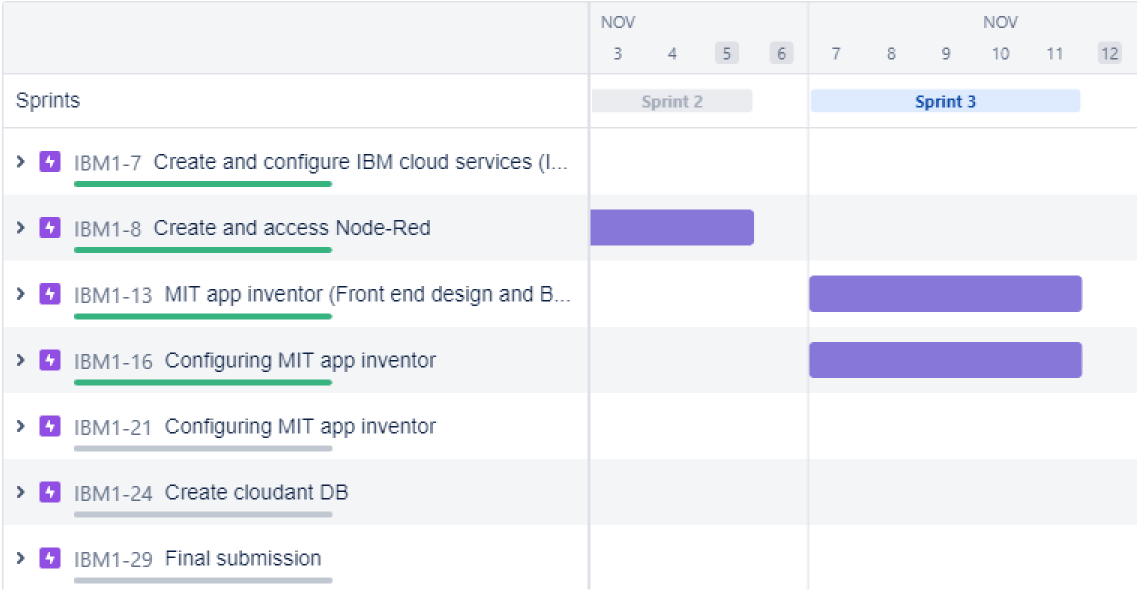
Dc

SPRINT BURNDOWN CHART:





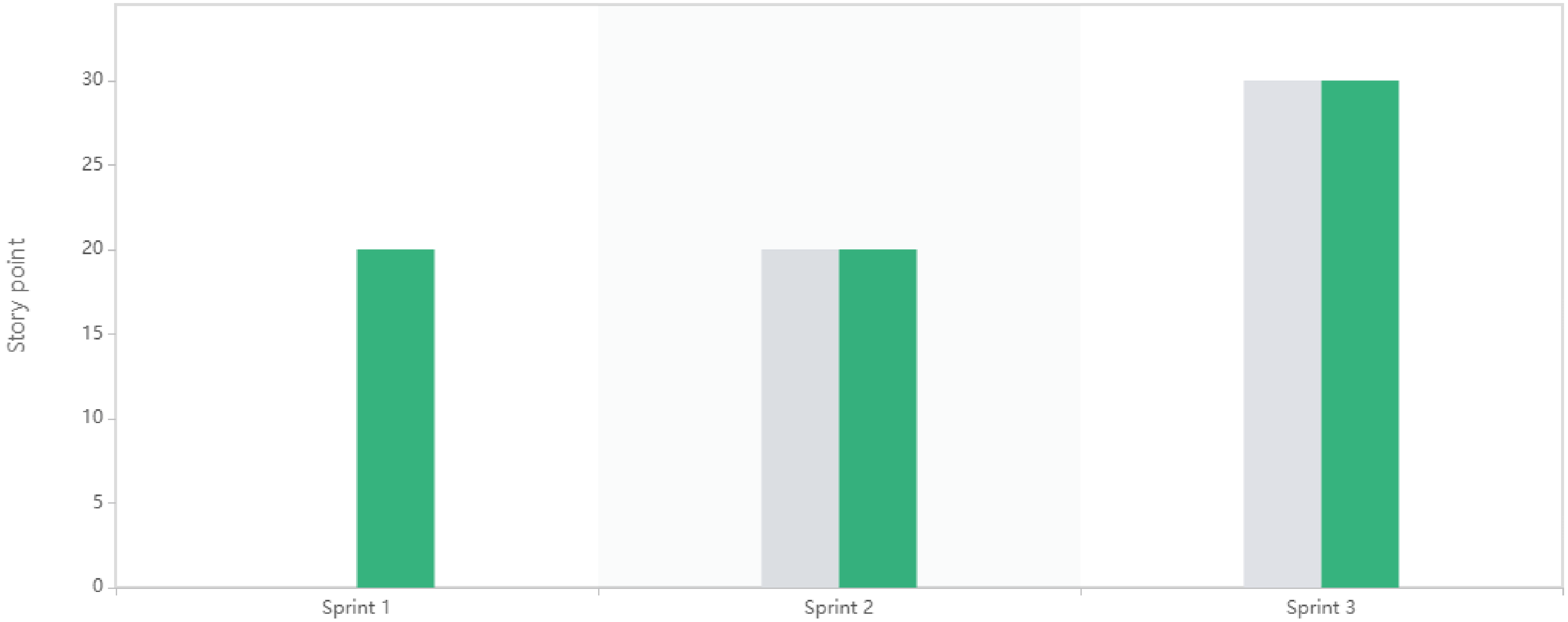
ROAD MAP:



Sprints

- IBM1-7 Create and configure IBM cloud services (I...
- IBM1-8 Create and access Node-Red
- IBM1-13 MIT app inventor (Front end design and B...
- IBM1-16 Configuring MIT app inventor
- IBM1-21 Configuring MIT app inventor
- IBM1-24 Create cloudant DB
- IBM1-29 Final submission

VELOCITY CHART:


Story point
Sprint 1
Sprint 2
Sprint 3