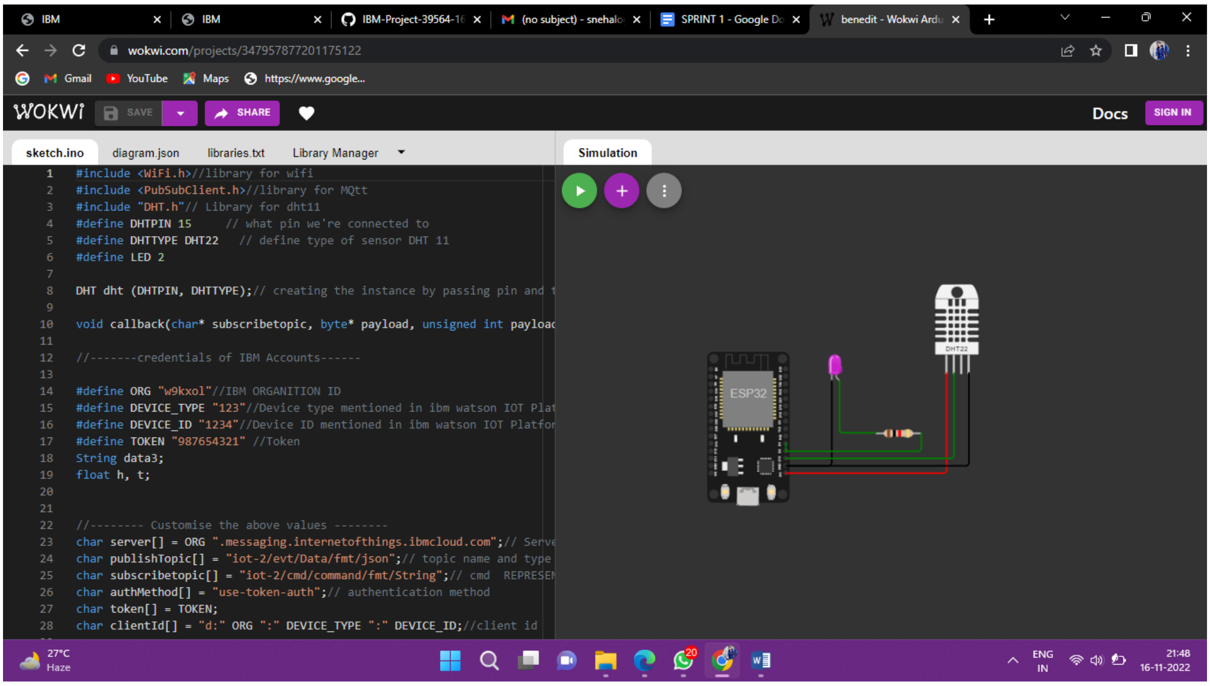
SMART FARMER – IOT ENABLEDD SMART FARMING APPLICATION

PROJECT DEVELOPMENT – DELIVERY OF SPRINT - 1

DATE	16 NOVEMBER 2022
TITLE	SMART FARMER – IOT ENABLEDD SMART FARMING APPLICATION
TEAM ID	PNT2022TMID33748
TEAM LEADER NAME	SUBIKA M
TEAM MEMBER NAME	PEMALATHA S SELENA CLARA M SNEHA L

Connect Sensor in ESP8266

CIRCUIT DIAGRAM:



```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include "DHT.h"// Library for dht11
#define DHTPIN 15     // what pin we're connected to
#define DHTTYPE DHT22   // define type of sensor DHT 11
#define LED 2

DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and t

void callback(char* subscribetopic, byte* payload, unsigned int payload

//-------credentials of IBM Accounts------

#define ORG "w9kxol"//IBM ORGANITION ID
#define DEVICE_TYPE "123"//Device type mentioned in ibm watson IOT Plat
#define DEVICE_ID "1234"//Device ID mentioned in ibm watson IOT Platfor
#define TOKEN "987654321" //Token
String data3;
float h, t;


//-------- Customise the above values --------
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Serve
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd  REPRESEN
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
```

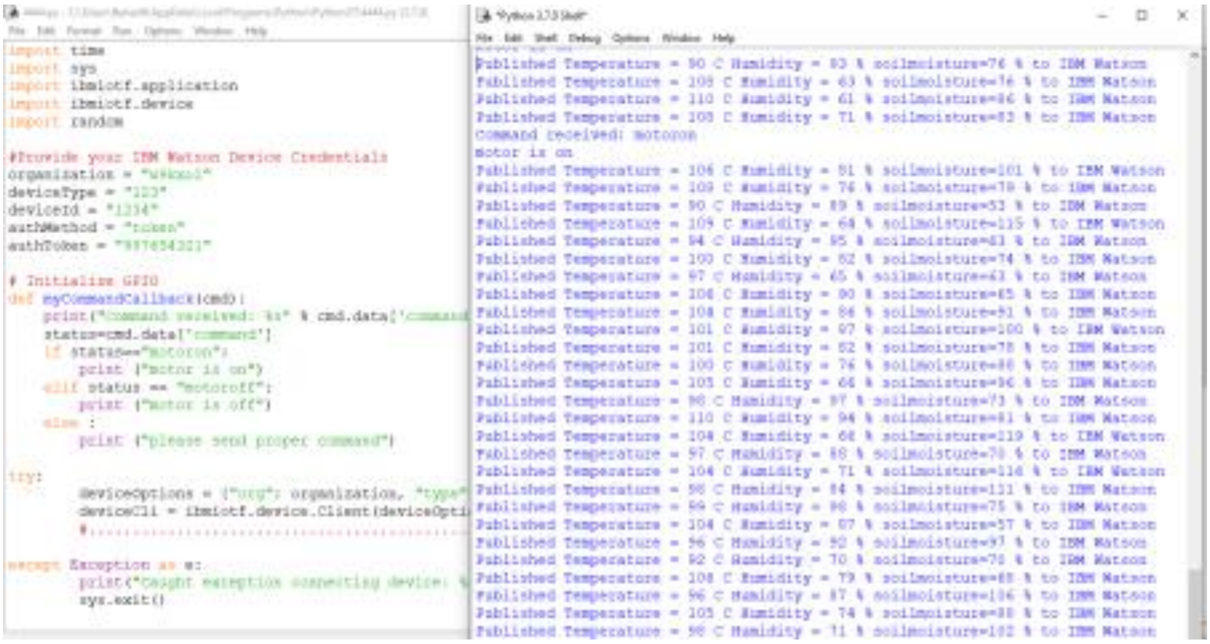
Develop a Python Code:

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:

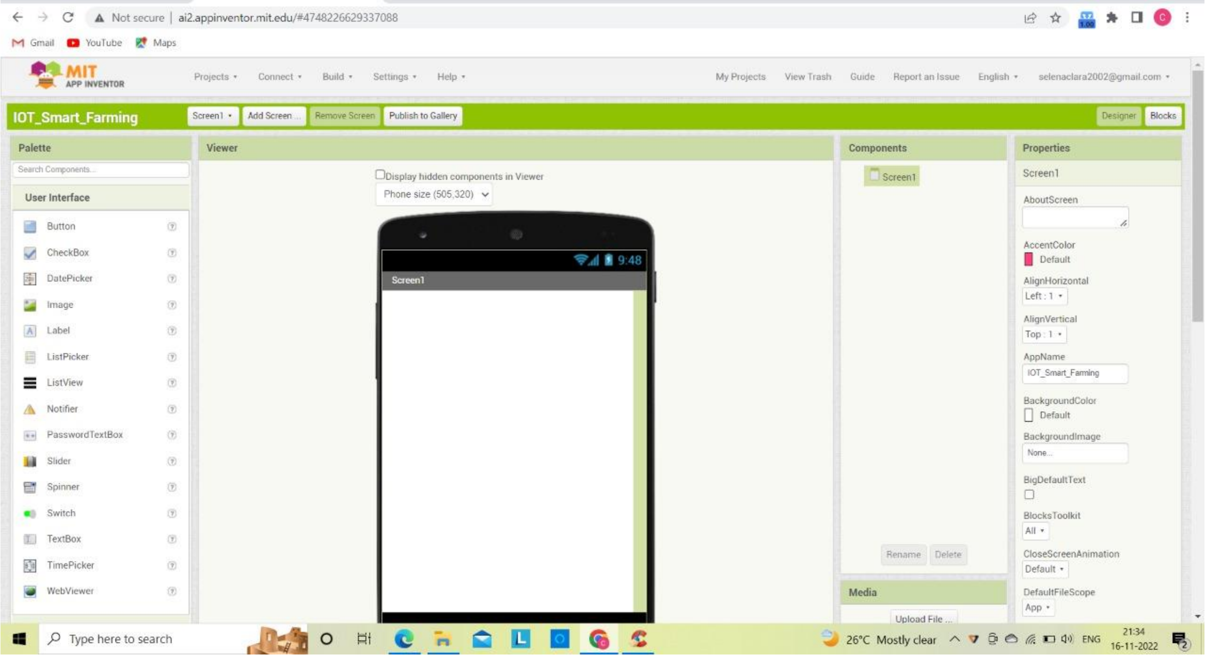


```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random

#Provide your IBM Watson Device Credentials
organization = "u8kxo1"
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()
```

```
Published Temperature = 90 C Humidity = 83 % soilmoisture=76 % to IBM Watson
Published Temperature = 109 C Humidity = 63 % soilmoisture=76 % to IBM Watson
Published Temperature = 110 C Humidity = 61 % soilmoisture=86 % to IBM Watson
Published Temperature = 103 C Humidity = 71 % soilmoisture=83 % to IBM Watson
command received: motoron
motor is on
Published Temperature = 106 C Humidity = 51 % soilmoisture=101 % to IBM Watson
Published Temperature = 109 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=83 % to IBM Watson
Published Temperature = 100 C Humidity = 62 % 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 = 108 C Humidity = 84 % soilmoisture=91 % to IBM Watson
Published Temperature = 101 C Humidity = 97 % soilmoisture=100 % to IBM Watson
Published Temperature = 101 C Humidity = 52 % 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 = 68 % soilmoisture=119 % to IBM Watson
Published Temperature = 97 C Humidity = 88 % soilmoisture=78 % 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 = 67 % soilmoisture=57 % to IBM Watson
Published Temperature = 96 C Humidity = 92 % soilmoisture=97 % to IBM Watson
Published Temperature = 92 C Humidity = 70 % soilmoisture=75 % to IBM Watson
Published Temperature = 108 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
```

Develop an application with MIT APP inventor:

Mobile App opening page:



Not secure | ai2.appinventor.mit.edu/#4748226629337088

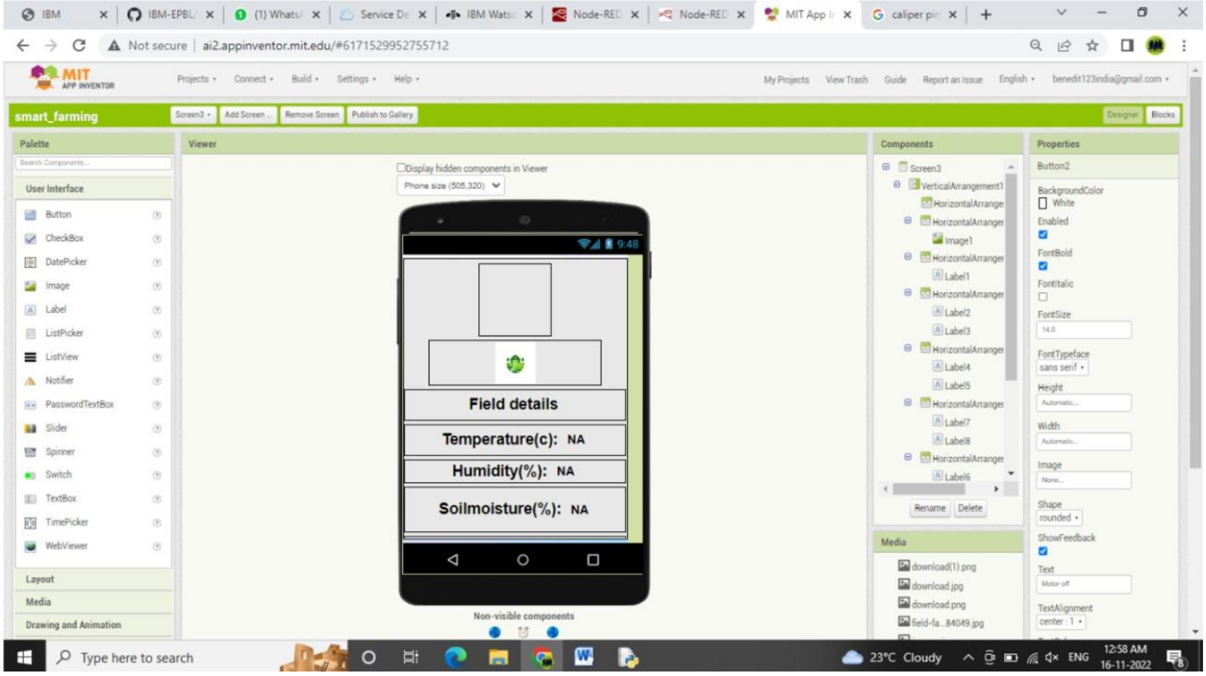
IOT_Smart_Farming

Display hidden components in Viewer

Phone size (505,320)

AppName: IOT_Smart_Farming

Mobile App Log in Page:



ai2.appinventor.mit.edu/#6171529952755712

Field details

Temperature(c): NA

Humidity(%): NA

Soilmoisture(%): NA

Button2

BackgroundColor: White

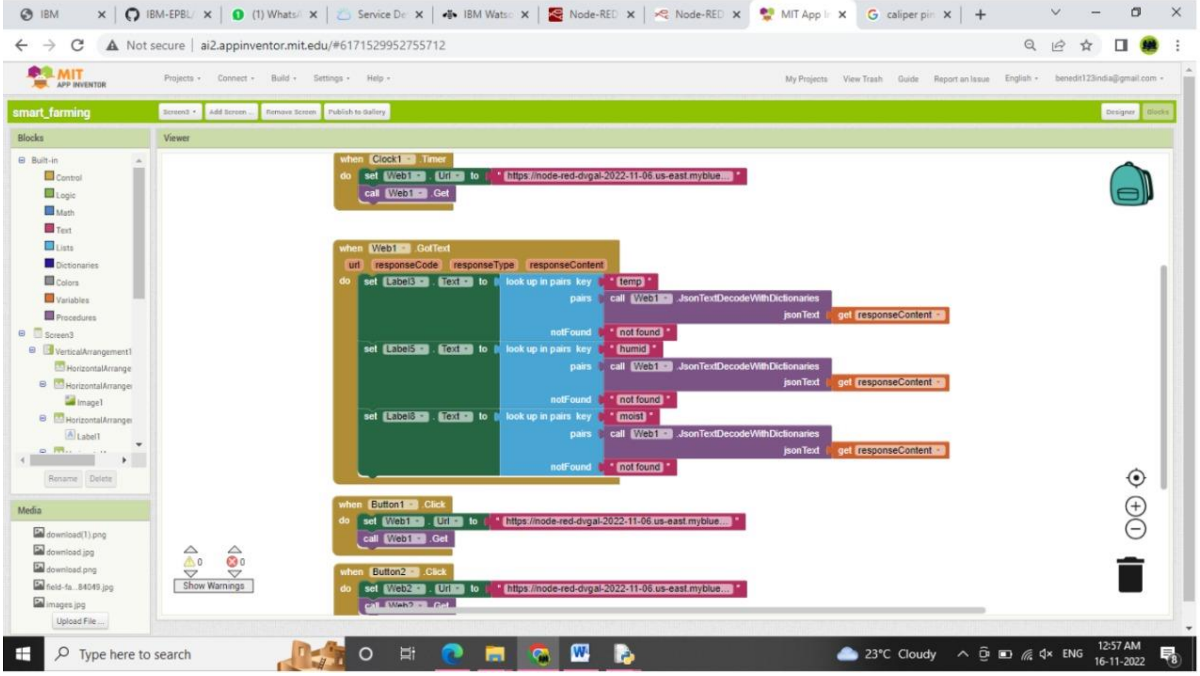
FontSize: 14.0

FontTypeface: sans serif

Shape: rounded

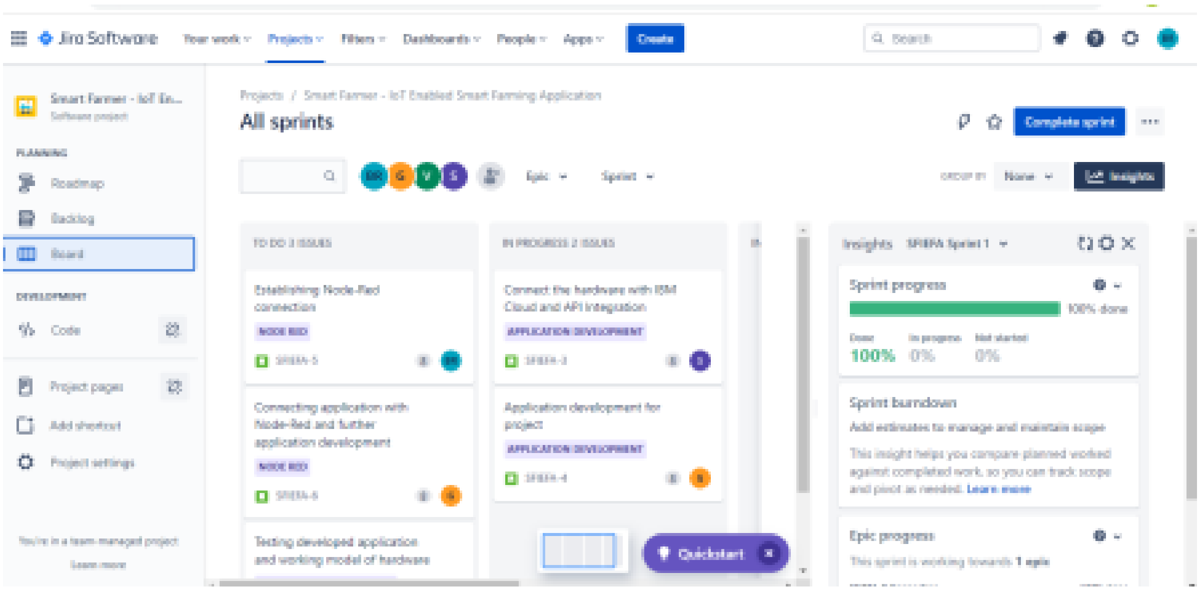
Text: Motor off

TextAlignment: center : 1



```
when Clock1 .Timer
do  set Web1 . Url to "https://node-red-dvgal-2022-11-06.us-east.myblue..."
    call Web1 .Get

when Web1 .GotText
  url  responseCode  responseType  responseContent
do  set Label3 . Text to  look up in pairs  key "temp"
                                             pairs  call Web1 .JsonTextDecodeWithDictionaries
                                                                jsonText  get responseContent
                                             notFound "not found"
    set Label5 . Text to  look up in pairs  key "humid"
                                             pairs  call Web1 .JsonTextDecodeWithDictionaries
                                                                jsonText  get responseContent
                                             notFound "not found"
    set Label8 . Text to  look up in pairs  key "moist"
                                             pairs  call Web1 .JsonTextDecodeWithDictionaries
                                                                jsonText  get responseContent
                                             notFound "not found"

when Button1 .Click
do  set Web1 . Url to "https://node-red-dvgal-2022-11-06.us-east.myblue..."
    call Web1 .Get

when Button2 .Click
do  set Web2 . Url to "https://node-red-dvgal-2022-11-06.us-east.myblue..."
    call Web2 .Get
```

JIRA Software Sprint Planning:



Projects / Smart Farmer - IoT Enabled Smart Farming Application

All sprints

TO DO 3 ISSUES

Establishing Node-Red connection
NODE RED
SFIEFA-5

Connecting application with Node-Red and further application development
NODE RED
SFIEFA-6

Testing developed application and working model of hardware

IN PROGRESS 2 ISSUES

Connect the hardware with IBM Cloud and API integration
APPLICATION DEVELOPMENT
SFIEFA-3

Application development for project
APPLICATION DEVELOPMENT
SFIEFA-4

Insights SFIEFA Sprint 1

Sprint progress
100% done
Done 100%
In progress 0%
Not started 0%

Sprint burndown
Add estimates to manage and maintain scope
This insight helps you compare planned worked against completed work, so you can track scope and pivot as needed. Learn more

Epic progress
This sprint is working towards 1 epic