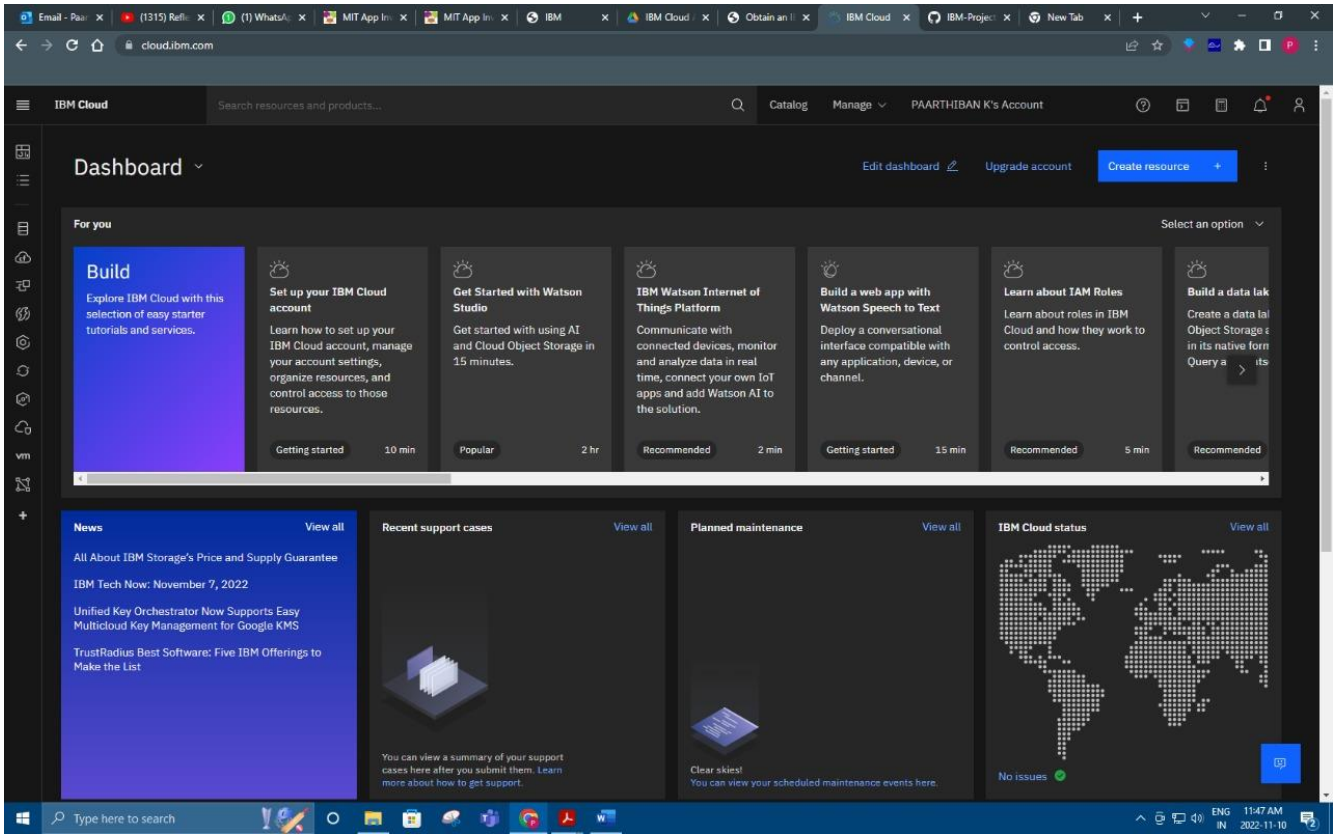
SPRINT 1

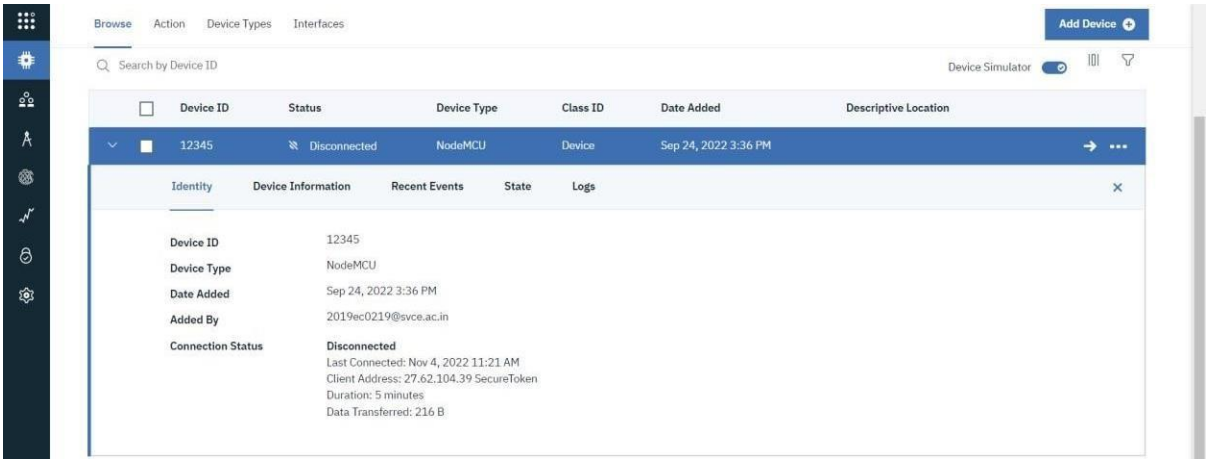
Date	29 October 2022
Team ID	PNT2022TMID01891
Project Name	Smart Farmer – IOT Enabled Smart Farming Application

Configuration:

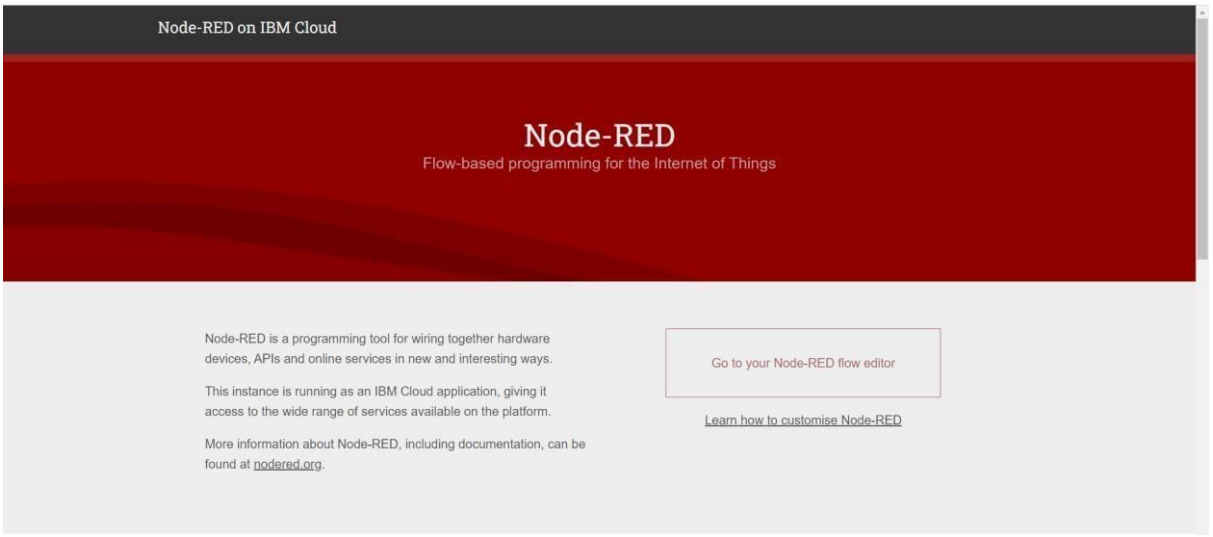
An account has been created on the required platforms (IBM Cloud, IBM Watson, NodeRed, MIT App Inventor) **IBM Cloud:**

IBM Watson:

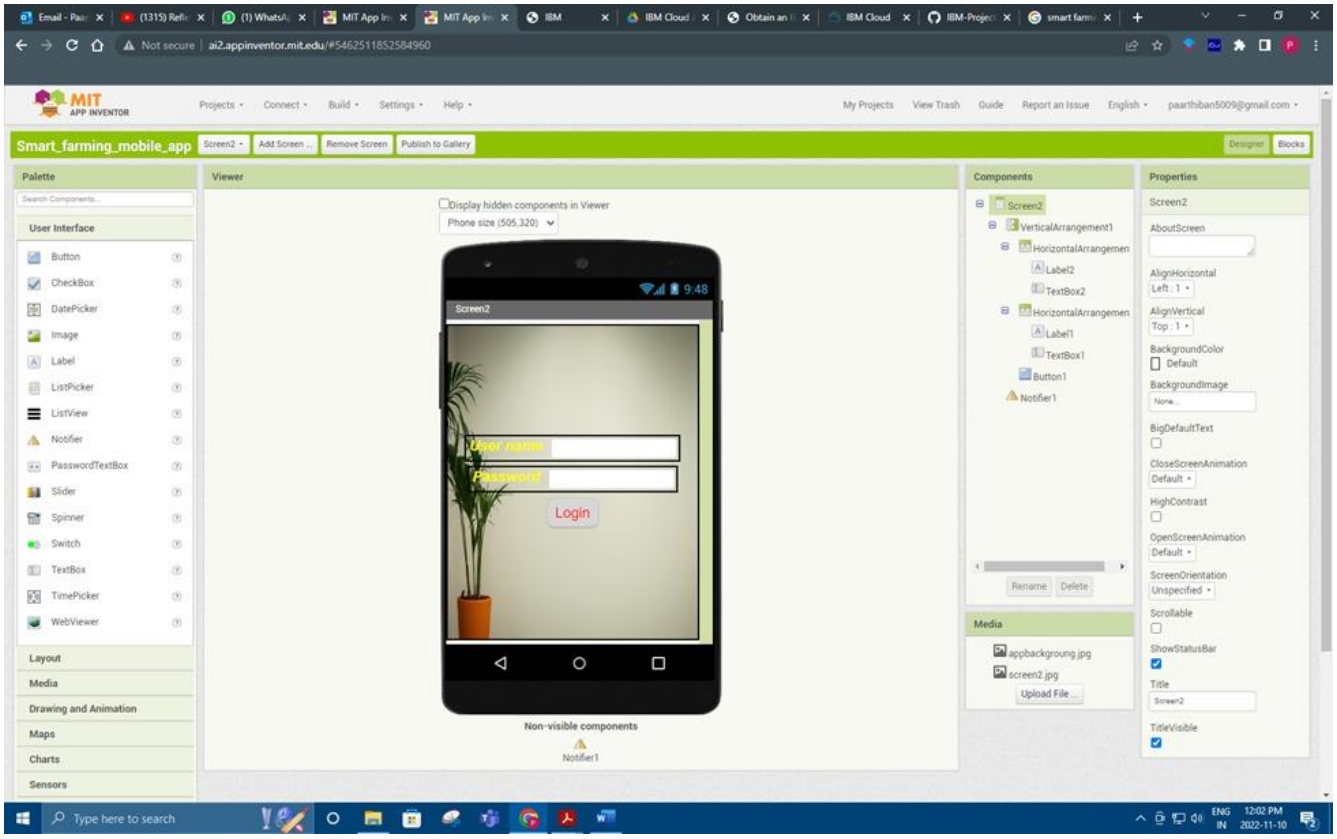



Browse
Action
Device Types
Interfaces
Add Device
Search by Device ID
Device Simulator
Device ID
Status
Device Type
Class ID
Date Added
Descriptive Location
12345
Disconnected
NodeMCU
Device
Sep 24, 2022 3:36 PM
Identity
Device Information
Recent Events
State
Logs
Device ID
12345
Device Type
NodeMCU
Date Added
Sep 24, 2022 3:36 PM
Added By
2019ec0219@svce.ac.in
Connection Status
Disconnected
Last Connected: Nov 4, 2022 11:21 AM
Client Address: 27.62.104.39 SecureToken
Duration: 5 minutes
Data Transferred: 216 B

Node-Red:


Node-RED on IBM Cloud
Node-RED
Flow-based programming for the Internet of Things
Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways.
This instance is running as an IBM Cloud application, giving it access to the wide range of services available on the platform.
More information about Node-RED, including documentation, can be found at nodered.org.
Go to your Node-RED flow editor
Learn how to customise Node-RED

MIT App Inventor:



Simulation:

Program:

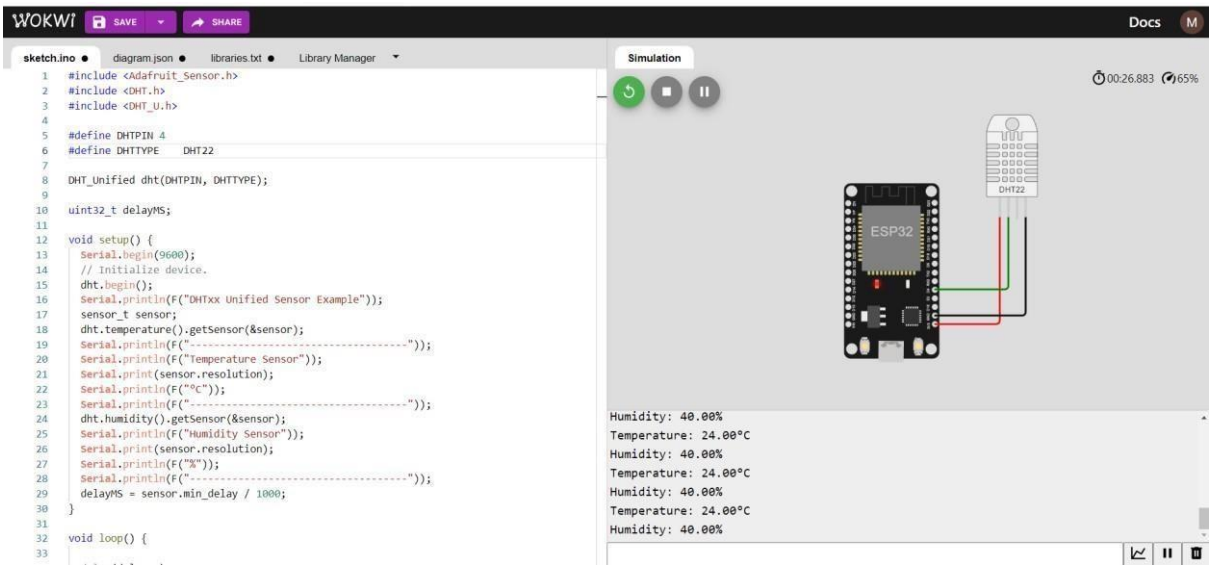
```
#include <Adafruit_Sensor.h>
#include <DHT.h> #include
<DHT_U.h>

#define DHTPIN 4
#define DHTTYPE DHT22

DHT_Unified dht(DHTPIN, DHTTYPE); uint32_t
delayMS; void
setup() {
Serial.begin(960 0);
//
Initialize
device.
  dht.begin();
  Serial.println(F("DHTxx Unified Sensor Example"));
sensor_t sensor;
```

```
  dht.temperature().getSensor(&sensor);
  Serial.println(F(" ----------------------------------
  "));
  Serial.println(F("Temperature Sensor"));
  Serial.print(sensor.resolution);
  Serial.println(F("°C"));
  Serial.println(F("————————————————————————————————————-
"));                        dht.humidity().getSensor(&sensor);
Serial.println(F("Humidity Sensor"));
  Serial.print(sensor.resolution); Serial.println(F("%"));
  Serial.println(F("------------------------------------------------------------------"));
delayMS = sensor.min_delay / 1000;
}  void  loop()  {  delay(delayMS);
sensors_event_t              event;
dht.temperature().getEvent(&event);
if (isnan(event.temperature)) {
    Serial.println(F("Error reading temperature!"));
  } else
{
    Serial.print(F("Temperature: "));
    Serial.print(event.temperature);
    Serial.println(F("°C"));
  }  dht.humidity().getEvent(&event);  if
(isnan(event.relative_humidity)) {
    Serial.println(F("Error reading humidity!"));
  } else
{
    Serial.print(F("Humidity: "));
    Serial.print(event.relative_humidity);
    Serial.println(F("%"));
  }
}
```

OUTPUT:




```
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

#define DHTPIN 4
#define DHTTYPE    DHT22

DHT_Unified dht(DHTPIN, DHTTYPE);

uint32_t delayMS;

void setup() {
  Serial.begin(9600);
  // Initialize device.
  dht.begin();
  Serial.println(F("DHTxx Unified Sensor Example"));
  sensor_t sensor;
  dht.temperature().getSensor(&sensor);
  Serial.println(F("------------------------------------"));
  Serial.println(F("Temperature Sensor"));
  Serial.print(sensor.resolution);
  Serial.println(F("°C"));
  Serial.println(F("------------------------------------"));
  dht.humidity().getSensor(&sensor);
  Serial.println(F("Humidity Sensor"));
  Serial.print(sensor.resolution);
  Serial.println(F("%"));
  Serial.println(F("------------------------------------"));
  delayMS = sensor.min_delay / 1000;
}

void loop() {
```