Green House Monitoring and Controlling System

Higher Diploma in Software Engineering22.2F

IOT Course Work

Source code

KAHDSE22.2F-020 L.H.M Fernando

School of Computing and Engineering

National Institute of Business Management

Colombo-7

Table of Content

Introduction

An automated IoT (Internet of Things) greenhouse is a technologically advanced agricultural facility designed to optimize and control the environment within the greenhouse to support better plant growth. It leverages a network of interconnected sensors, actuators, and control systems to monitor and adjust various environmental parameters.

Features of the automated greenhouse system

- Automated Lightning system
- Automated Security system
- Automated Temperature controlling system.
- Automated Watering system

Can include.

- Nutrition analyzing system.

Advantages

- Can change environment states for better plant growth.
  The greenhouse is equipped with sensors that monitor temperature and humidity. Based on the data collected, the system can automatically adjust the heating, cooling, and ventilation systems to maintain ideal conditions for plant growth. This includes regulating temperature and humidity to prevent stress and optimize photosynthesis.

- Reduce human involvement.

- Can control remotely.
  Using mobile applications or web interfaces, greenhouse operators can monitor and control the environment remotely. This allows for immediate response to changing conditions and the ability to adjust from anywhere.

- Real time data monitoring
  The greenhouse system can send real-time alerts.

- Data analysis and making decisions.
  IoT greenhouses collect vast amounts of data from various sensors. This data is analyzed to identify trends and patterns. Machine learning algorithms can then make predictions and recommendations for adjustments to the environmental parameters, helping to optimize plant growth further over time.

Disadvantages

- Cost

  The greenhouse can be designed with energy-efficient systems to reduce operational costs. For example, it might incorporate solar panels, energy-efficient heating and cooling systems, and rainwater harvesting for irrigation.

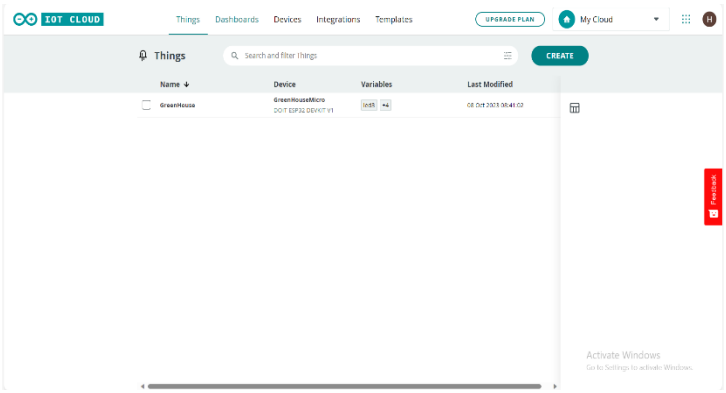
- Need better connection.

Used technologies.

Used language & software tool.

- Arduino IoT cloud

Arduino IoT Cloud is a cloud-based platform provided by Arduino for building and managing Internet of Things (IoT) projects. It offers a comprehensive ecosystem for connecting and controlling IoT devices, sensors, and actuators, allowing users to monitor and interact with their projects from anywhere with an internet connection.



Key Features and Components:

Device Management: Arduino IoT Cloud provides tools for registering and managing IoT devices. You can easily add new devices to your account, configure them, and monitor their status.

Dashboard: The web-based dashboard allows users to create customized interfaces to monitor and control their IoT devices. You can design user-friendly dashboards with widgets to display real-time data from sensors and control actuators remotely.

Device Communication: Arduino IoT Cloud uses MQTT (Message Queuing Telemetry Transport) for secure and efficient device communication. MQTT is a lightweight and reliable protocol for sending and receiving data.

Mobile App: Arduino offers a mobile app for both iOS and Android platforms, enabling users to access their IoT projects on the go. You can view sensor data, control devices, and receive notifications from your smartphone or tablet.

User Management: You can invite team members to collaborate on your IoT projects. The platform supports multi-user access with role-based permissions.

Secure Data: Arduino IoT Cloud places a strong emphasis on data security and privacy. Data is encrypted during transmission and storage to protect user information and project data.

Data Logging and Storage: The platform allows users to log and store historical data from sensors and actuators, making it easy to analyze trends and patterns over time.

Automation and Triggers: You can create automation rules and triggers to automate actions based on specific conditions. For example, you can set up rules to control devices based on sensor readings.

Integration with Arduino Boards: Arduino IoT Cloud seamlessly integrates with a variety of Arduino boards, including the Arduino MKR family and Arduino IoT hardware, making it easy to set up and connect devices.

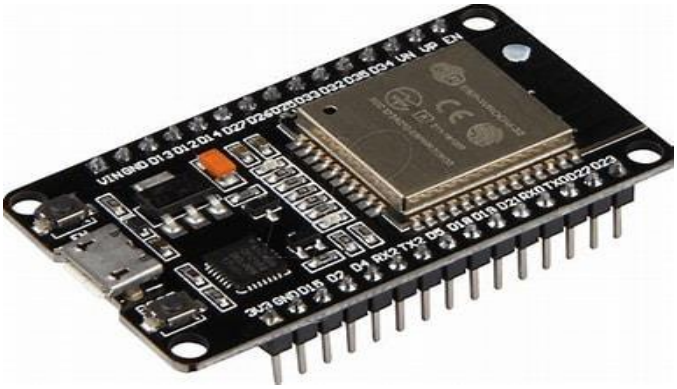
Why Arduino IoT cloud instead Blynk IoT

- Simplicity and Ease of Use
- IoT Device Management
- Arduino Ecosystem Integration
- Security
- Scalability
- Official Support and Documentation
- Integration with Other Arduino Services

MC

- ESP 32 DEVKITV1

  The ESP32 DEVKITV1 is a widely used development board based on the ESP32, a powerful Wi-Fi and Bluetooth microcontroller. The ESP32 is developed by Espressif Systems and is known for its versatility, performance, and low power consumption. The ESP32 DEVKITV1 board is designed to make it easy for developers to prototype and develop IoT and embedded projects.



Key Features and Components:

ESP32 Module: The core of the DEVKITV1 is the ESP32 module itself. The ESP32 is a dual-core microcontroller that features both Wi-Fi (802.11b/g/n) and Bluetooth (Bluetooth Low Energy or BLE) connectivity. It's based on the Tensilica Xtensa LX6 processor and has ample processing power and memory for a wide range of applications.

Integrated Antenna: The board typically comes with an integrated antenna for Wi-Fi and Bluetooth connectivity. This antenna is a printed trace on the PCB.

External Connectivity: The board has pins for connecting to various external components and sensors. These include GPIO pins, SPI, I2C, UART, and more, allowing you to interface with a wide range of devices.

USB-to-Serial Converter: The board usually includes a USB-to-serial converter (CP2102 or CH340) for easy programming and debugging via the USB port. This is particularly helpful for uploading code to the ESP32 and monitoring serial output.

Power Supply: The board can be powered through the micro-USB port, an external battery, or an external power supply. It typically supports a wide voltage range.

Reset and Boot Buttons: The board includes a reset button to restart the ESP32 and a boot button that can be used to enter programming mode.

LED Indicators: There are usually two onboard LEDs. One indicates power status, and the other can be controlled through GPIO pins in your code for various purposes.

Why ESP32 instead NODE MCU?

- More Processing Power
- More Memory
- Dual-Core Architecture
- Built-In Bluetooth

- Improved Power Efficiency

power connectors

Micro USB cable

5v DC power adapter

Jumper wires



power convertors

- Relay modules

  A relay module is an electrical component that acts as an electromechanical switch. It is used to control high-power or high-voltage electrical circuits using a low-power or low-voltage control signal. Relay modules are commonly employed in various applications to provide electrical isolation and control over loads, such as motors, lights, heaters, and other electrical devices.

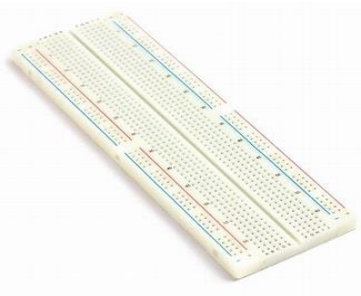
access technology

WIFI

other

Bread board

Screwdriver



Automated Lightning system

IoT greenhouses often use a combination of natural and artificial lighting. Sensors measure the intensity and duration of sunlight, and automated shading systems can be adjusted to control the amount of light reaching the plants. LED grow lights are also used to supplement natural light, and their intensity and spectrum can be adjusted to match the needs of different plant species.

1) For humans

2) For trees

1. Light control using virtual switch

2. Control light’s brightness using virtual meter

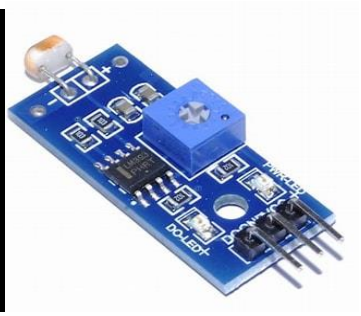
3. Light control using LDR

- LDR changes its resistance in response to changes in light intensity, we can use to light

control.

Need

- LED
- LDR sensor

Automated security system

A security system using a touch sensor is designed to detect and respond to physical touch or contact with an object or surface. It can be used to secure doors, windows, cabinets, or other access points by triggering an alarm or notification when unauthorized contact is made.

Need

- Touch sensor
- Buzzer

Automated watering system

Soil moisture sensors detect the moisture levels in the growing medium, ensuring that plants receive the right amount of water. Automated irrigation systems can adjust the frequency and duration of watering. Additionally, nutrient delivery systems can provide the necessary nutrients to the plants, ensuring they receive the right balance of essential elements.

Need

- Soil moisture sensor
- Pump
- Soil, water, Horse

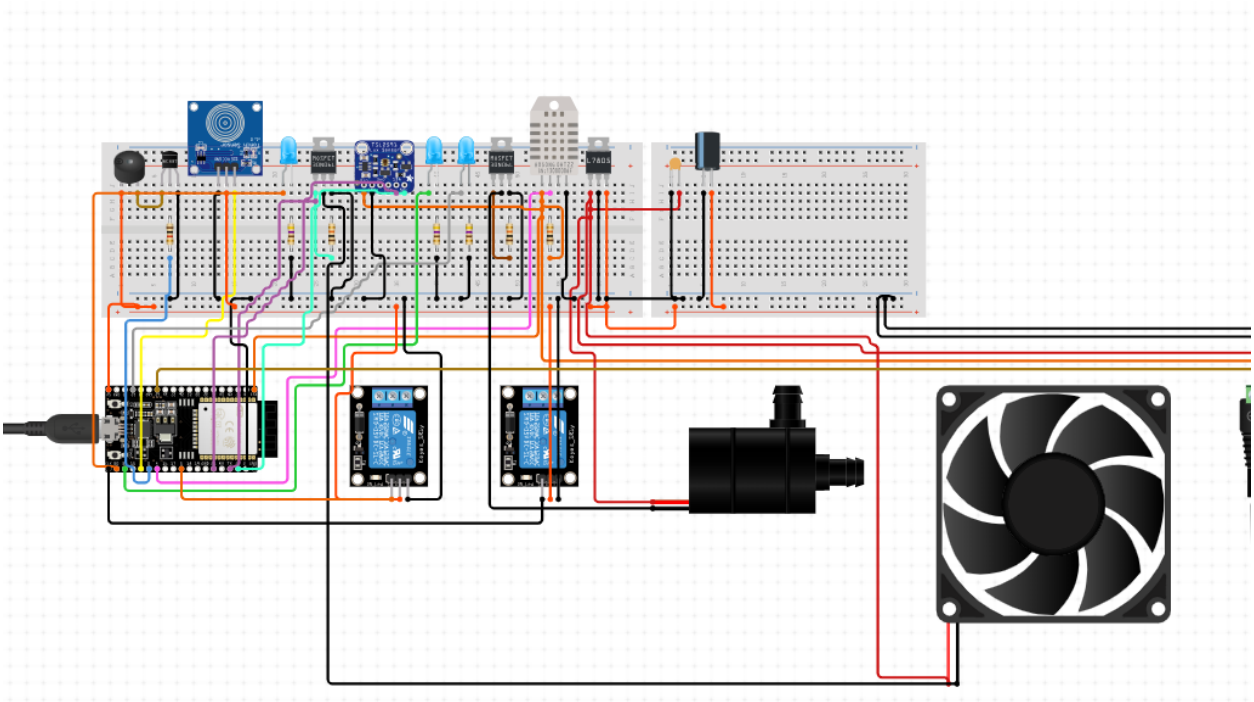
Automated temperature controlling system.

The greenhouse is equipped with sensors that monitor temperature and humidity. Based on the data collected, the system can automatically adjust the heating, cooling, and ventilation systems to maintain ideal conditions for plant growth. This includes regulating temperature and humidity to prevent stress.

Need

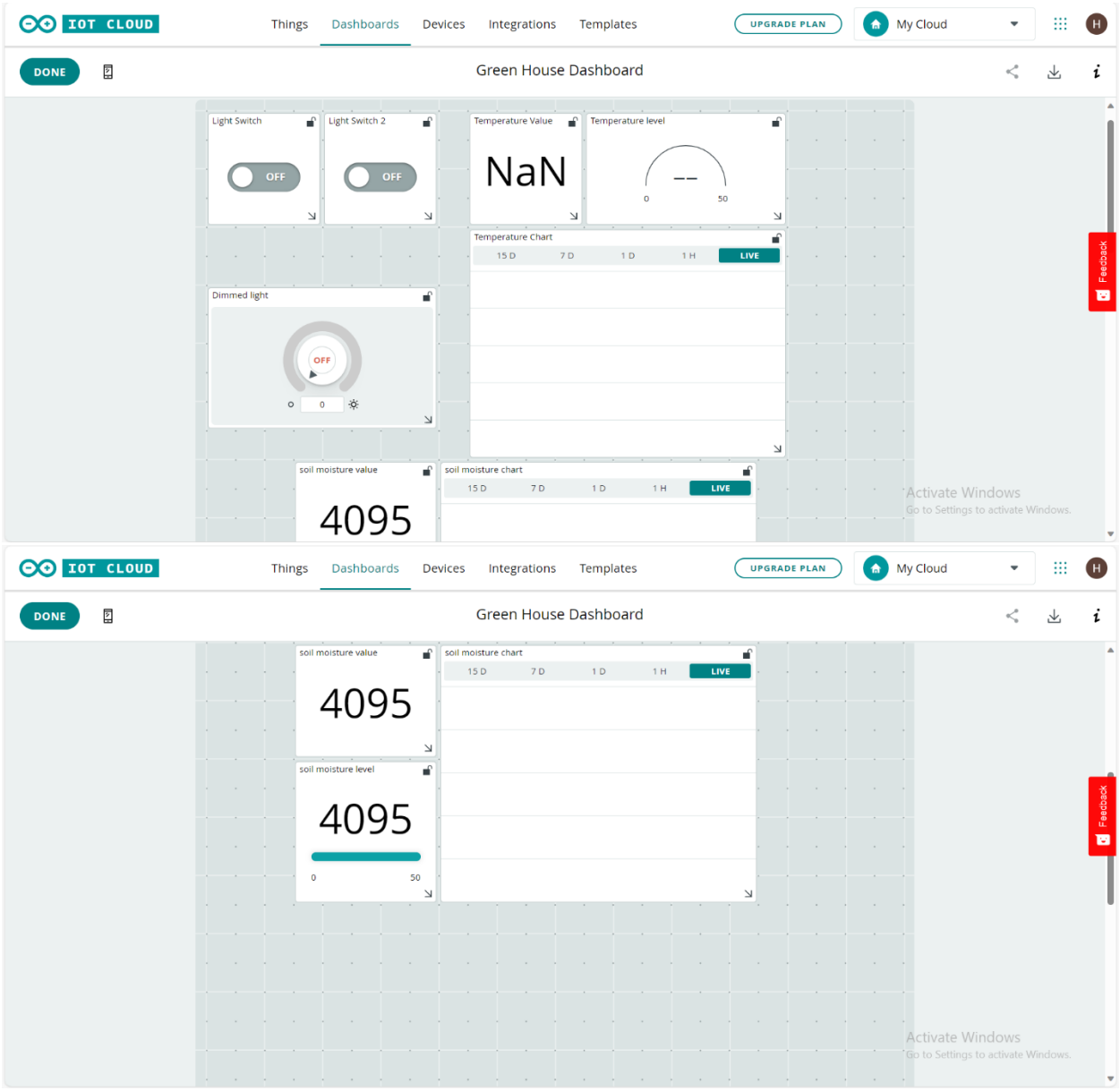
- DH11 temperature sensor
- 5V DC fan
- Heat element

Circuit diagram

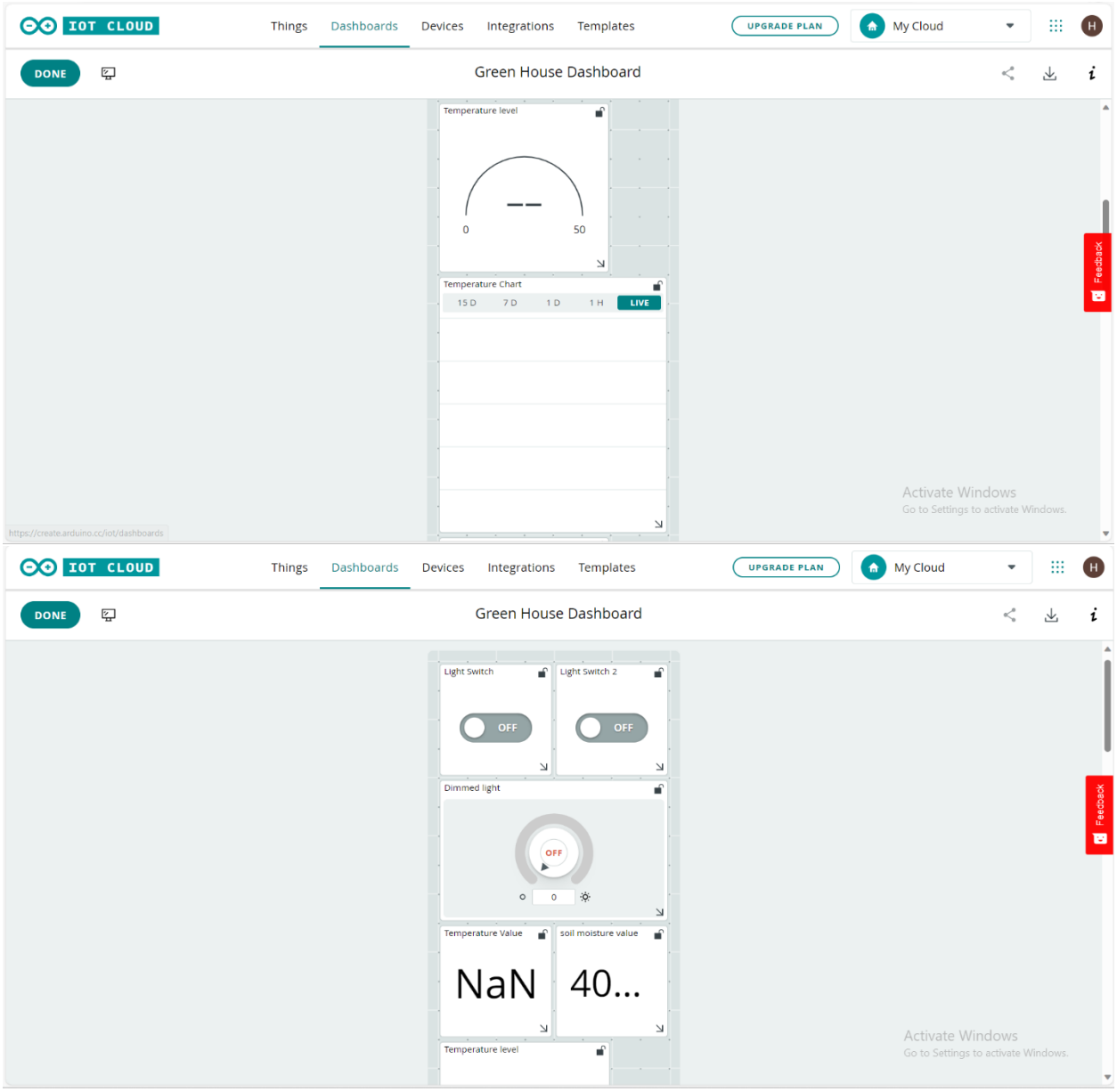


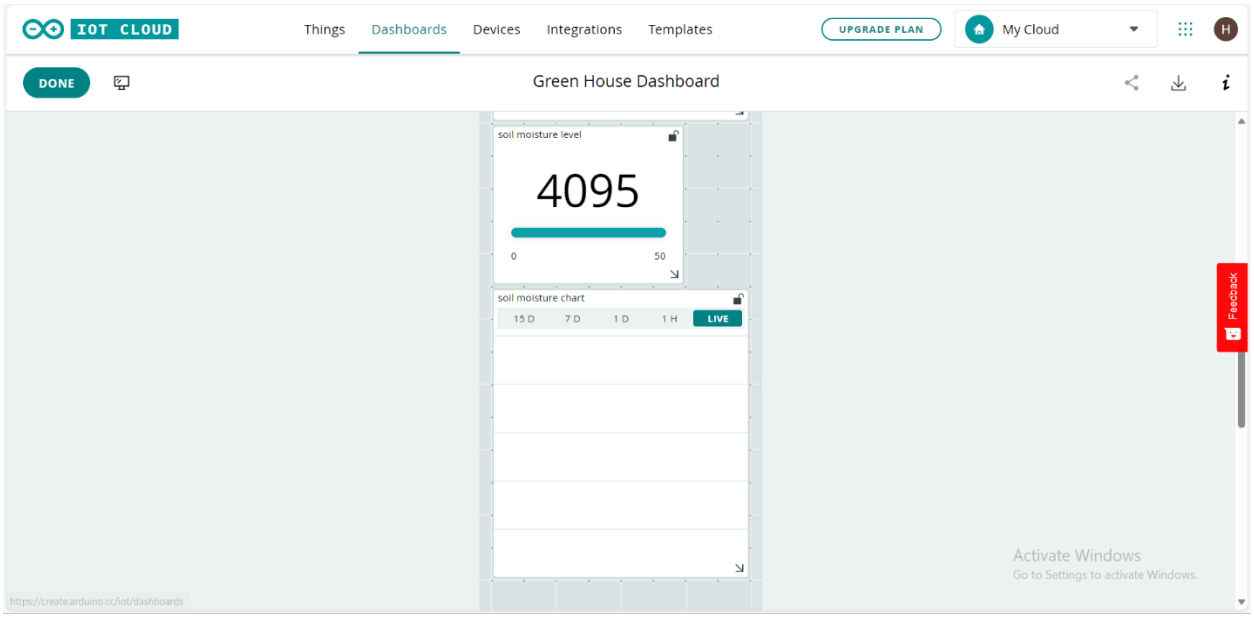
Dashboards

Web dashboard



Mobile dashboard




IOT CLOUD
Things
Dashboards
Devices
Integrations
Templates
UPGRADE PLAN
My Cloud
H
DONE
Green House Dashboard
soil moisture level
4095
0
50
soil moisture chart
15 D
7 D
1 D
1 H
LIVE
Feedback
Activate Windows
Go to Settings to activate Windows.
https://create.arduino.cc/iot/dashboards

source code

```
// DHT sensor library - Version: 1.4.4
#include <DHT.h>
#include <DHT_U.h>

/*
  Sketch generated by the Arduino IoT Cloud Thing "Untitled"
  https://create.arduino.cc/cloud/things/f9699b07-1951-483e-b8ee-37dfca0bded5

  Arduino IoT Cloud Variables description

  The following variables are automatically generated and updated when changes are made to the Thing

  CloudDimmedLight led3;
  CloudTemperatureSensor temperature;
  int soil_moisture;
  bool led;
  bool led2;

  Variables which are marked as READ/WRITE in the Cloud Thing will also have functions
  which are called when their values are changed from the Dashboard.
  These functions are generated with the Thing and added at the end of this sketch.
*/

#include "thingProperties.h"
int led_pin  = 2;
int led_pin2 = 4;
int led_pin3 = 5;
int brightness;
int ldr_pin  = 21;
int led_pin4 = 14;
#define DHT_SENSOR_PIN 13
#define DHT_SENSOR_TYPE DHT11
DHT dht_sensor(DHT_SENSOR_PIN,DHT_SENSOR_TYPE);
int relay1_pin = 23;
int touch_pin = 32;
int buzzer_pin = 18;
int soil_moisture_pin = 34;
int relay2_pin = 15;
int fan_pin = 19;
```

```
void setup() {
  // Initialize serial and wait for port to open:
  Serial.begin(9600);
  pinMode(led_pin  , OUTPUT);
  pinMode(led_pin2 , OUTPUT);
  pinMode(ldr_pin , INPUT);
  pinMode(led_pin3 , OUTPUT);
  pinMode(led_pin4 , OUTPUT);
  pinMode(DHT_SENSOR_PIN , INPUT);
  dht_sensor.begin();
  pinMode(relay1_pin , OUTPUT);
  pinMode(touch_pin , INPUT);
  pinMode(buzzer_pin , OUTPUT);
  pinMode(soil_moisture_pin , INPUT);
  pinMode(relay2_pin , OUTPUT);
  pinMode(fan_pin , OUTPUT);

  // This delay gives the chance to wait for a Serial Monitor without blocking if none is found
  delay(1500);

  // Defined in thingProperties.h
  initProperties();

  // Connect to Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);

  /*
     The following function allows you to obtain more information
     related to the state of network and IoT Cloud connection and errors
     the higher number the more granular information you'll get.
     The default is 0 (only errors).
     Maximum is 4
  */
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}

void loop() {
  ArduinoCloud.update();

  if(digitalRead(ldr_pin)==1)
  {
    digitalWrite(led_pin4, HIGH);
  }

  else
```

```
  {
    digitalWrite(led_pin4, LOW);
  }


  float humidity=dht_sensor.readHumidity();
  Serial.print("Humidity");
  Serial.print(humidity);
  temperature=dht_sensor.readTemperature();
  Serial.print("Tempereture");
  Serial.print(temperature);

   if(temperature<25)
 {
   digitalWrite(relay1_pin, HIGH);
 }
 else
 {
   digitalWrite(fan_pin, HIGH);
 }

 if(digitalRead(touch_pin)==1)
 {
   digitalWrite(buzzer_pin, HIGH);
 }

 soil_moisture = analogRead(soil_moisture_pin);
 if(soil_moisture>1200)
 {
   digitalWrite(relay1_pin, HIGH);
 }
  delay(2000);
}


/*
  Since Led is READ_WRITE variable, onLedChange() is
  executed every time a new value is received from IoT Cloud.
*/
void onLedChange()  {
  if (led)
  {
    digitalWrite(led_pin, HIGH);
  }
  else
  {
```

```
    digitalWrite(led_pin, LOW);
  }
}

/*
  Since Led2 is READ_WRITE variable, onLed2Change() is
  executed every time a new value is received from IoT Cloud.
*/
void onLed2Change()  {
  if (led2)
  {
    digitalWrite(led_pin2, HIGH);
  }
  else
  {
    digitalWrite(led_pin2, LOW);
  }
}

/*
  Since Led3 is READ_WRITE variable, onLed3Change() is
  executed every time a new value is received from IoT Cloud.
*/
void onLed3Change()  {
  brightness=led3.getBrightness();

  if(brightness>0)
  {
    analogWrite(led_pin3, brightness);
  }

}




/*
  Since Light is READ_WRITE variable, onLightChange() is
  executed every time a new value is received from IoT Cloud.
*/
void onLightChange()  {
  // Add your code here to act upon Light change
}
```

Conclusion

In summary, an automated IoT greenhouse has the potential to revolutionize agriculture by precisely controlling environmental factors, leading to better plant growth, increased crop yields, and reduced resource wastage. These smart greenhouses can significantly contribute to sustainable and efficient food production.

References

Chat GPT 3.5
YouTube