HƯỚNG DẪN THỰC HÀNH
Lab 2-3: IoT Platform – Blynk

- Bài 6: Virtual Pin Reply trong Blynk
- Bài 7: Xử lý sự kiện trong Blynk
- Bài 8: Xây dựng hệ thống điều khiển relay bằng ứng dụng IoT Manager
- Bài 9: Hệ thống bật tắt relay bằng node-red
- Bài tập về nhà: Người dạy có thể cung cấp thêm một số bài tập về nhà cho học viên để hoàn thiện các kiến thức liên quan đến nội dung bài học.

Bài 6: Virtual Pin Reply trong Blynk

Bài này hướng dẫn cách gửi các giá trị lên Ứng dụng Blynk
Thiết lập dự án trong ứng dụng: Widget Hiển thị giá trị được gắn với V5 và được đặt ở bất kỳ tần suất đọc nào (ví dụ: 1 giây). Trong ứng dụng này, tần suất đọc của Widget phải được đặt thành PUSH.

```
// Hàm này được gọi khi có một Widget yêu cầu nhận dữ liệu từ Pin ảo (5)
BLYNK_READ(V5)
{
    Blynk.virtualWrite(V5, millis() / 1000);
}

void myTimerEvent()
{
        Blynk.virtualWrite(V5, millis() / 1000);
}
void setup() {
…
// Setup a function to be called every second
timer.setInterval(1000L, myTimerEvent);
}
```

Bài 7: Xử lý sự kiện (eventor) trong Blynk

- Bạn có thể sử dụng các quy tắc được xác định trước ở phía ứng dụng.
- Thiết lập dự án trong ứng dụng Blynk
- Widget eventor được thiết lập như sau:
    - Khi V0 bằng 1, thiết lập V1 là 255;
    - Khi V0 bằng 0, thiết lập V1 là 0;
- Widget LED ở pin V1

```
BlynkTimer timer;
boolean flag = true;
```

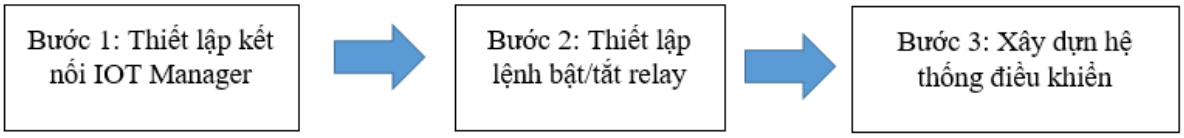
```
void sendFlagToServer() {
  if (flag) {
    Blynk.virtualWrite(V0, 1);
  } else {
    Blynk.virtualWrite(V0, 0);
  }
  flag = !flag;
}

BLYNK_WRITE(V1) {
  //ở đây ta nhận giá trị 0 hoặc 255
  int ledValue = param.asInt();
}
void setup()
{
  …
  // Hàm được gọi mỗi giây
  timer.setInterval(1000L, sendFlagToServer);
}
```

Bài 8: Xây dựng hệ thống điều khiển relay bằng ứng dụng IoT Manager

Hệ thống điều khiển dàn thổ canh bằng IoT được thực hiện bằng cách sử dụng một tài khoản trên CloudMQTT và tiến hành xây dựng kết nối giữa relay và tthiết bị điều khiển NodeMCU sau đó tiến hành xây dựng chương trình thiết lập giao diện, dữ liệu trạng thái và gửi dữ liệu lên Cloud để thiết lập hệ thống điều khiển thông qua ứng dụng trên thiết bị di động. Hệ thống này có giao diện và phần cứng như sau:

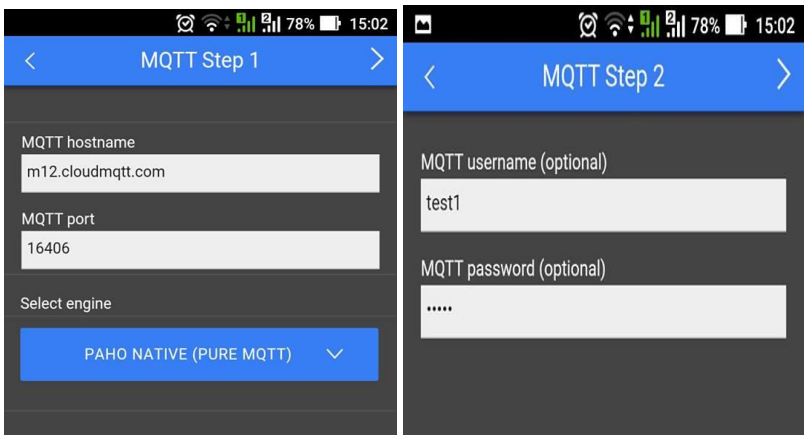
Hệ thống điều khiển relay bằng IoT Manager

Các bước thực hiện hệ thống điều khiển relay bằng ứng dụng IoT Manager:



Bước 1: tiến hành xây dựng các kết nối đến IoT Manager thông qua việc kết nối đến hệ thống server CloudMQTT.

Sau khi tiến hành tạo tài khoản thì tiến hành xây dựng ứng dụng trên thiết bị di động và lấy thông tin đã xây dựng instance mới điền vào iot manager đã cài đặt trên điên thoại. Trong đó cổng port đầu tiên sau đó tiến hành tạo một user mới, sau đó add them vào topic: chỗ patterm ghi "#" như vậy là chúng ta đã xong phần cài đặt ứng dụng lên thiết bị di động.



Thiết lập kết nối đến ứng dung IoT Manager

Sau khi thiết lập kết nối trên điện thoại di động chúng ta tiến hành thiết lập kết nối trên chương trình Aduino để gửi và nhận dữ liệu từ thiết bị Aduino vào hệ thống để thiết lập điều khiển.

```
const char* ssid = "cloudmqtt"; // địa chỉ/mật khẩu wifi
const char* password = "123456789";

const String mqttServer = "m12.cloudmqtt.com";// địa chỉ cloudmqtt
const int mqttPort = 16406;
const String mqttUser = "test1";
const String mqttPassword = "12345";
```

Bước 2: Thiết lập hệ thống điều khiển relay bằng IoT Manager

Tiếp theo chúng ta tiến hành nạp code cho ESP8266 để điều khiển cũng như thiết kế giao diện trên ứng dụng điện thoại. Để kết nối được với ứng dụng điều khiển iot manager cần khai báo để kết nối đến hệ thống kết nối đến wifi trong hàm look():

```
void loop() {
  if (WiFi.status() != WL_CONNECTED) {
    Serial.print("Connecting via WiFi to ");
    Serial.print(ssid);
    Serial.println("...");

    WiFi.begin(ssid, password);
    if (WiFi.waitForConnectResult() != WL_CONNECTED) {
      return;
    }
    Serial.println("");
    Serial.println("WiFi connect: Success");
    Serial.print("IP address: ");
    Serial.println(WiFi.localIP());
  }

  if (WiFi.status() == WL_CONNECTED) {
    if (!client.connected()) {
      Serial.println("Connecting to MQTT server ...");
      bool success;
      if (mqttUser.length() > 0) {
        success = client.connect( MQTT::Connect( deviceID ).set_auth(mqttUser, mqttPassword) );
```

Tiếp đó chúng ta thực hiện khai báo các tham số kết nối với trang web để tạo các giao diện iot manager. Các biến này gồm các biến lưu trữ tên, màu, chân điều khiển, kiểu điều khiển,… và khai báo một để gửi dữ liệu qua đám mây cũng như dữ liệu xuất hiện trên màn hình iot manager.

```
void initVar() {
  id    [0] = "0";
  page  [0] = "Vườn Rau";
  descr [0] = "Đèn-0";
  widget[0] = "toggle";
  pin[0] = D1;
  defaultVal[0] = 1;
  inverted[0] = true;
  sTopic[0]   = prefix + "/" + deviceID + "/light0";
  color[0]   = "\"color\":\"red\"";

  id    [1] = "1";
  page  [1] = "Vườn Rau";
  descr [1] = "Đèn-1";
  widget[1] = "toggle";
  pin[1] = D4;
  defaultVal[1] = 1;
  inverted[1] = true;
  sTopic[1] = prefix + "/" + deviceID + "/light1";
  color [1] = "\"color\":\"white\"";
```

```
  if (ids.length() == 0) {
     Serial.println("PUSH: ids not received, push failed");
     return;
  }
  if (!httpClient.connect(host, httpsPort)) {
     Serial.println("PUSH: connection failed");
     return;
  }
  String data = "{\"app_id\": \"8871958c-5f52-11e5-8f7a-c36f5770
  httpClient.println("POST " + url + " HTTP/1.1");
  httpClient.print("Host:");
  httpClient.println(host);
  httpClient.println("User-Agent: esp8266.Arduino.IoTmanager");
  httpClient.print("Content-Length: ");
  httpClient.println(data.length());
  httpClient.println("Content-Type: application/json");
  httpClient.println("Connection: close");
  httpClient.println();
  httpClient.println(data);
  httpClient.println();
  Serial.println(data);
  Serial.println("PUSH: done.");
}
```

✓ Tạo các trạng thái chạy cho bóng điện.

```
String setStatus ( String s ) {
  String stat = "{\"status\":\"" + s + "\"}";
  return stat;
}
String setStatus ( int s ) {
  String stat = "{\"status\":\"" + String(s) + "\"}";
  return stat;
}
```

Sau khi đã thiết lập các biến dữ liệu cũng như các khai báo kết nối đến điện toán đám mây thì chúng ta thiết lập hệ thống kết nối đến server CloudMQTT.

```
void pubConfig() {
  bool success;
  success = client.publish(MQTT::Publish(prefix, deviceID).set_qos(1));
  if (success) {
      delay(500);
      for (int i = 0; i < nWidgets; i = i + 1) {
        success = client.publish(MQTT::Publish(prefix + "/" + deviceID + "/config", thing_config[i]).set_qos(1));
        if (success) {
          Serial.println("Publish config: Success (" + thing_config[i] + ")");
        } else {
          Serial.println("Publish config FAIL! ("    + thing_config[i] + ")");
        }
        delay(150);
      }
  }
  if (success) {
     Serial.println("Publish config: Success");
  } else {
     Serial.println("Publish config: FAIL");
  }
  for (int i = 0; i < nWidgets; i = i + 1) {
      pubStatus(sTopic[i], stat[i]);
      delay(100);
  }
}
```

Bước 3: Xây dựng hệ thống điều khiển

Hệ thống điều khiển được thực hiện bằng cách vừa lấy dữ liệu và gửi dữ liệu lên server cùng một lúc. Tiến hành xây dựng một tập lệnh gửi dữ liệu đã được lấy ra từ thiết bị gửi lên giao diện iot manager để hiển thị lên màn hình.

```
if (client.connected()) {
  int chk= DHT.readl1(pin[3]);
  int x = DHT.temperature;
  int y = DHT.humidity;
  val = "{\"status\":\"" + String(x)+ "\"}";
  vall = "{\"status\":\"" + String(y)+ "\"}";
  client.publish(sTopic[3] + "/status", val );  // widget 3
  client.publish(sTopic[4] + "/status", vall );  // widget 3

  ///
  digitalWrite(pin[5],HIGH); //pin[6], HIGH);
  delay(500);
  sensorValuel = analogRead(A0);
  sensorValuel = constrain(sensorValuel, 530,840);
  sensorValuel = map(sensorValuel, 530, 840, 100, 0);
  digitalWrite(pin[5],LOW);
  val2 = "{\"status\":\"" + String(sensorValuel)+ "\"}";
  client.publish(sTopic[5] + "/status", val2 );  // widget 3
```

Tiếp tục xây dựng một hàm lấy dữ liệu từ các thiết bị để gửi dữ liệu đó xây dựng giao diện hiển thị chương trình trên thiết bị di động.

```
void callback(const MQTT::Publish& sub) {
  Serial.print("Get data from subscribed topic ");
  Serial.print(sub.topic());
  Serial.print(" => ");
  Serial.println(sub.payload_string());
  int chk= DHT.readl1(pin[3]);
  int i = DHT.temperature;
  int j = DHT.humidity;
  if (sub.topic() == sTopic[0] + "/control") {
    if (sub.payload_string() == "0") {
       newValue = 1; // inverted
       stat[0] = stat0;
    } else {
       newValue = 0;
       stat[0] = statl;
    }
    digitalWrite(pin[0],newValue);
    pubStatus(sTopic[0], stat[0]);
  } else if (sub.topic() == sTopic[1] + "/control") {
    if (sub.payload_string() == "0") {
       newValue = 1; // inverted
       stat[1] = stat0;
    } else {
```

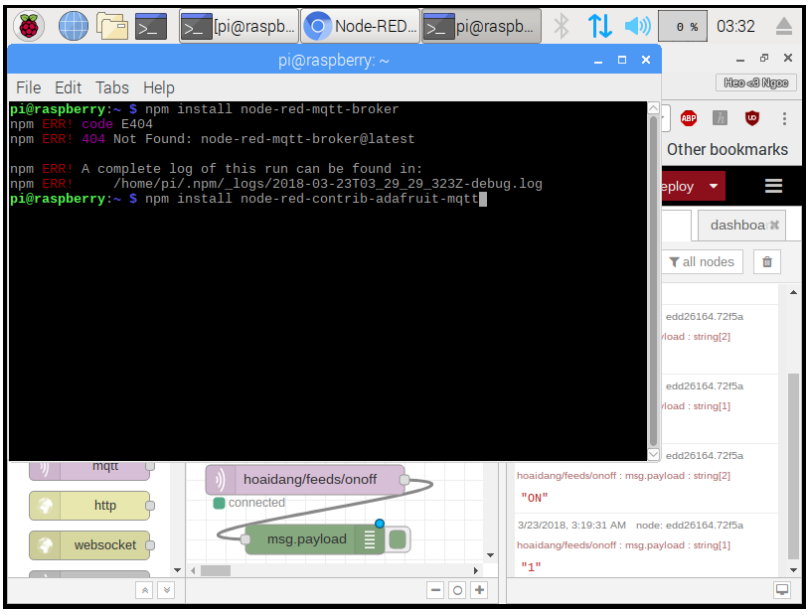
Sau khi xây dựng chương trình và tiến hành nạp dữ liệu cho thiết bị chúng ta mở ứng dunh iot manager trên điện thoại di động.

Kết quả: Chúng ta đã xây dựng được hệ thống điều khiên relay thông qua giao thức MQTT đến server CloudMQTT và sử dụng ứng dụng IoT Manager để điều khiển hệ thống đó. Hệ thống này sử dụng một ứng dụng trên thiết bị di động để trực tiếp điều khiển đến relay kết nối đến NodeMCU. Để lập trình xây dựng hệ thống đó chúng ta cần có một chút hiểu biết về ngôn ngữ lập trình web như HTML.

Bài 9: Xây dựng hệ thống bật/tắt relay bằng node-red

Node-red là một ứng đụng của điện toán đám mây dử dụng cloud Io.adafruit và dựa vào thông tin của các feeds đó để thiết lập giao diện điều khiển (phần 3.2). Hệ thống này được xây dựng trên hệ điều hành raspian sử dụng công cụ hỗ trợ node-red để bật tắt relay. Để kết nối với adafruit cần cài đặt 2 thư viện sử dụng trong node-red.

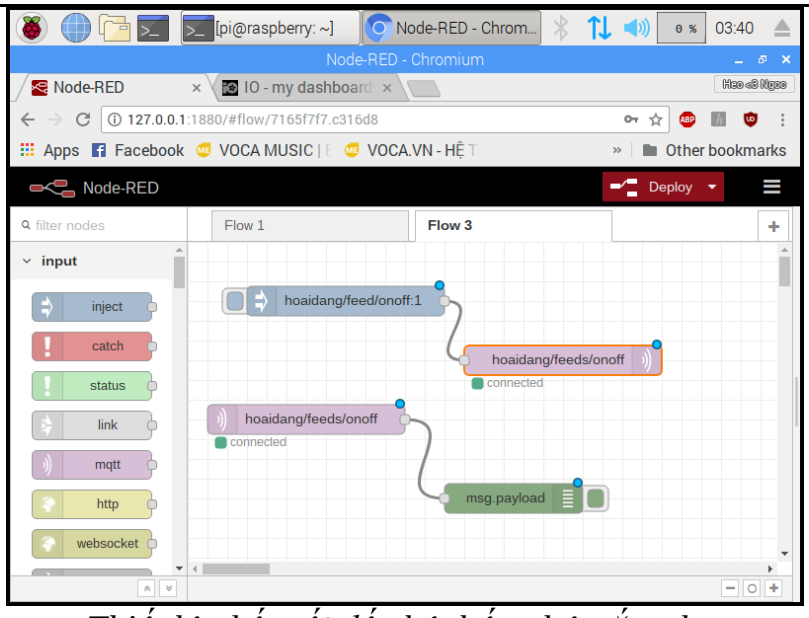
Node-red- mqtt-broker : Nhập lệnh → npm install node-red-mqtt-broker và Node-red-contrib-adafruit-mqtt bằng lệnh → npm install node-red-contrib-adafruit-mqtt



Cài đặt thư viện sử dụng Node-red

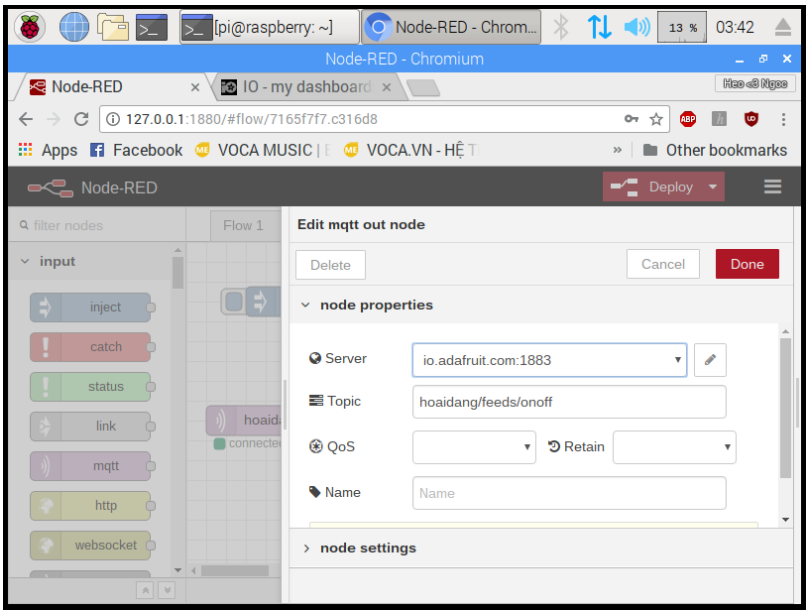
Sau cài đặt thư viện thì tiến hành thiết lập giao diện kết nối với io.adafruit bằng cách mở giao diện thiết kế node-red và tiến hành xây dựng giao diện thiết kế dưới đây hoặc là bạn có thể copy đoạn code này vào import flow thì sẽ không cần xây dựng giao diện nữa.

[{"id":"bdad81a8.4145d","type":"inject","z":"7165f7f7.c316d8","name":"","topic":"hoaidang/feed/onoff","payload":"1","payloadType":"num","repeat":"","crontab":"","once":false,"onceDelay":0.1,"x":120,"y":40,"wires":[["f7309c27.db324"]]},{"id":"f7309c27.db324","type":"mqtt out","z":"7165f7f7.c316d8","name":"","topic":"hoaidang/feeds/onoff","qos":"","retain":"","broker":"ebeedce.282c32","x":240,"y":100,"wires":[]},{"id":"7abf6e8a.0aab2","type":"mqtt in","z":"7165f7f7.c316d8","name":"","topic":"hoaidang/feeds/onoff","qos":"1","broker":"ebeedce.282c32","x":120,"y":240,"wires":[["edd26164.72f5a"]]},{"id":"edd26164.72f5a","type":"debug","z":"7165f7f7.c316d8","name":"","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","x":130,"y":300,"wires":[]},{"id":"ebeedce.282c32","type":"mqtt-broker","z":"","name":"io.adafruit.com:1883","broker":"io.adafruit.com","port":"1883","clientid":"","usetls":false,"compatmode":true,"keepalive":"60","cleansession":true,"willTopic":"","willQos":"0","willPayload":"","birthTopic":"","birthQos":"0","birthPayload":""}]



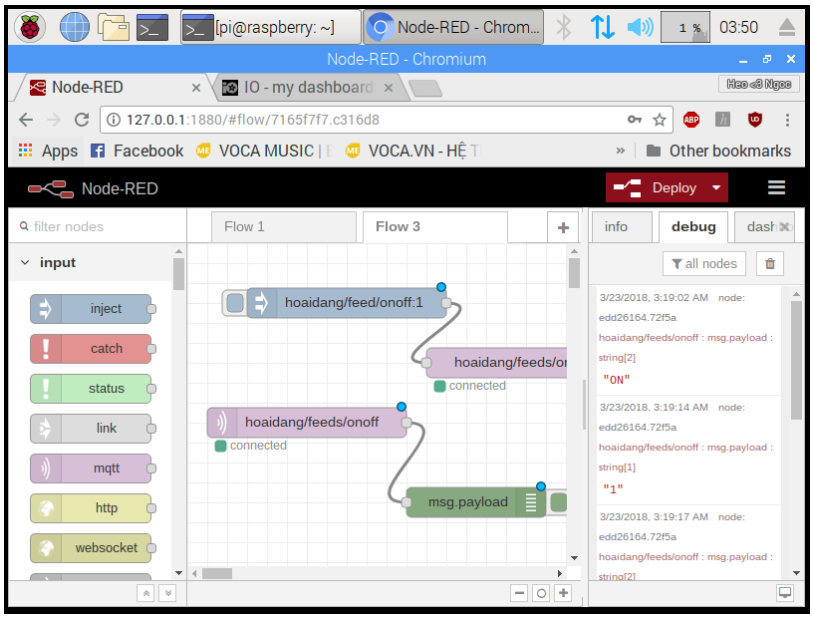
Thiết lập kết nối đến hệ thống bật tắt relay

Với một node inject để xuất dữ liệu kết nối với một mqtt out, trong đó topic đường link dẫn đến đối tượng mà bạn muốn gửi dữ liệu đến. Còn inject có thể thiết lập như sau: trong đó số 1 là dữ liệu muốn gửi đến cho onoff



Thiết lập cấu trúc hệ thống điều khiển relay

Sau khi đã thiết kế được giao diện ta chạy Deploy góc trên bên phải màn hình : thay đổi các dữ liệu trong io.adafruit.com thì dữ liệu sẽ được trả về trong debug như sau.



Giao diện kết nối hệ thống relay

Kết quả: Chúng ta xây dựng hệ thống điều khiển relay thông qua ứng dụng node-red trên hệ điều hành Raspian. Hệ thống này sử dụng các thiết lập trạng thái relay từ (phần 3.2) để gửi dữ liệu trạng thái đến Node-red và nhận dữ liệu từ các tác động ngược lại.