

ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

BÁO CÁO BÀI TẬP LỚN
Hệ nhúng

**Xây dựng hệ thống bật/tắt đèn, quạt tự động
sử dụng cảm biến ánh sáng, nhiệt độ**

Nhóm 38:

Họ tên	MSSV	Email	Lớp
Mai Xuân Ngọc	20204769	ngoc.mx204769@sis.hust.edu.vn	IT2 - 3
Vũ Minh Thành	20204790	thanh.vm204790@sis.hust.edu.vn	IT2 - 3

Giảng viên hướng dẫn:

TS. Đỗ Công Thuận

Bộ môn:

Kỹ thuật máy tính

HÀ NỘI, 08/2023

1	GIỚI THIỆU ĐỀ TÀI.....	4
1.1	Đặt vấn đề.....	4
1.2	Ý tưởng phát triển	4
1.3	Phần cứng sử dụng	5
1.3.1	Vi điều khiển ESP8266 – 12E.....	5
1.3.2	Giao tiếp I2C	6
1.3.3	Màn hình OLED I2C.....	7
1.3.4	Module DS1307 + AT24C32 Tiny RTC.....	8
1.3.5	Quang trở.....	10
2	PHÂN TÍCH THIẾT KẾ	11
2.1	Mô hình hệ thống	11
2.2	Thiết kế các giải pháp kết nối	12
2.2.1	Đặt vấn đề.....	12
2.2.2	Mô hình kết nối với mạng Wifi.....	13
2.3	Kết nối VDK với SioT Platform và nền tảng Blynk.....	14
2.4	Thiết kế các gói	14
2.5	Thiết kế sơ đồ mạch	15
3	PHÁT TRIỂN MÃ NGUỒN	16
3.1	Cài đặt kết nối Internet cho thiết bị nhúng.....	16
3.2	Kết nối với server SioT	17
3.3	Gói Debug	17
3.4	Gói Switch.....	18
3.5	Gói Sensor đọc dữ liệu	18
4	TRIỂN KHAI	19
4.1	Tạo thiết bị trên SioT	19
4.2	Tạo thiết bị trên Blynk	21
5	KẾT QUẢ THỰC HIỆN	25
5.1	Kết nối Internet	25
5.2	Các chế độ hoạt động	27
5.2.1	Chế độ tự động dừng cảm biến	27
5.2.2	Chế độ điều khiển bằng công tắc	28

5.3	Kết nối với Server SioT và đẩy dữ liệu từ lên server qua WebAPI	29
DOANH MỤC THAM KHẢO.....		31
PHỤ LỤC		31

1 GIỚI THIỆU ĐỀ TÀI

1.1 Đặt vấn đề

Trong bối cảnh cuộc cách mạng công nghiệp 4.0, các thiết bị thông minh được sản xuất và ra đời rất đa dạng, phong phú về chủng loại, mẫu mã có thể áp dụng để giải quyết một số nhu cầu cuộc sống về học tập, y tế, việc nhà, giao thông,...

Quan sát ở một số vùng nông thôn, em nhận thấy, đối với các hộ chăn nuôi gia súc, gia cầm, theo định kỳ hàng ngày sẽ phải đến từng chuồng để bật/tắt đèn, quạt cho phù hợp với trạng thái của vật nuôi. Đối với những hộ chăn nuôi nhỏ lẻ, việc này sẽ không quá vất vả; nhưng đối với những hộ chăn nuôi lớn, diện tích chăn nuôi lên đến hàng chục ha, thì việc đến từng chuồng gia súc, gia cầm để bật/tắt đèn, quạt sẽ tiêu tốn một lượng không nhỏ thời gian, công sức.

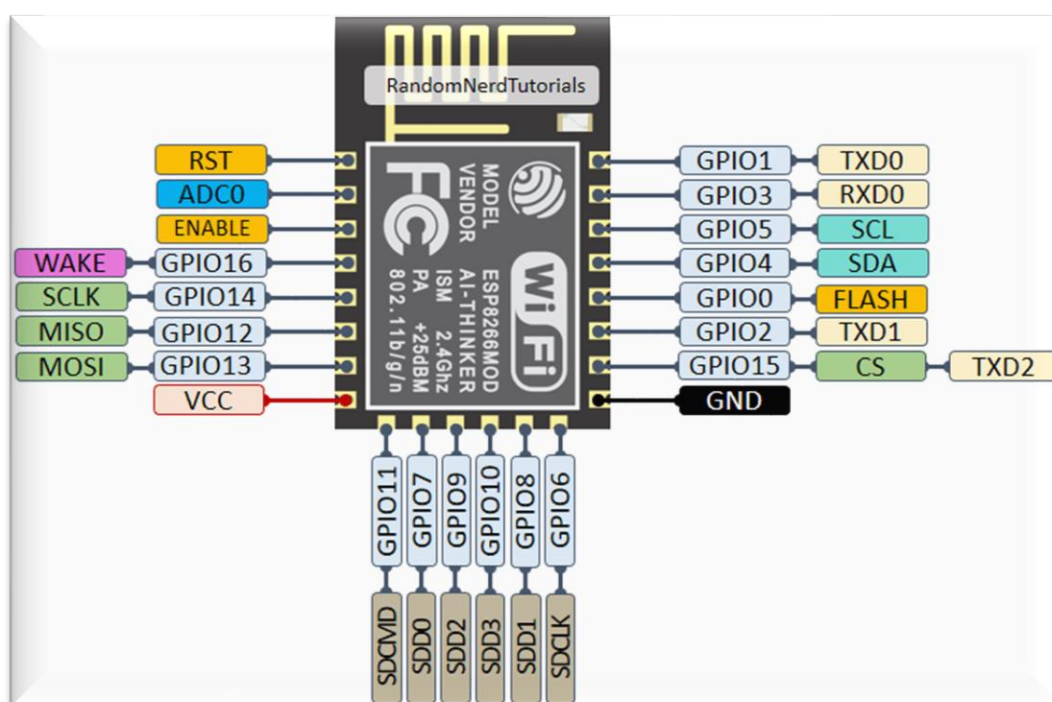
Nắm bắt được những khó khăn trên, với những kiến thức đã được học, chúng em đã xây dựng một hệ thống bật/tắt quạt, đèn tự động dựa trên điều kiện môi trường xung quanh bằng cách sử dụng cảm biến nhiệt độ, ánh sáng. Hệ thống trên có thể giúp các hộ chăn nuôi tiết kiệm được rất nhiều thời gian, công sức.

1.2 Ý tưởng phát triển

Ngoài chức năng tự động ở trên, chúng em tạo thêm chế độ giúp người dùng bật/tắt đèn, quạt từ xa để người dùng có thể chủ động kiểm soát trong một số tình huống đặc biệt.

1.3 Phần cứng sử dụng

1.3.1 Vi điều khiển ESP8266 – 12E



Sơ đồ chân Esp8266-12e

Thu phát wifi ESP8266 12E là module wifi giá rẻ và được đánh giá rất cao cho các ứng dụng liên quan đến Internet và Wifi cũng như các ứng dụng truyền nhận sử dụng thay thế cho các module RF khác.

ESP8266 là một chip tích hợp cao, được thiết kế cho nhu cầu của một thế giới kết nối mới, thế giới Internet of thing (IOT). Nó cung cấp một giải pháp kết nối mạng Wi-Fi đầy đủ và khép kín, cho phép nó có thể lưu trữ các ứng dụng hoặc để giảm tải tất cả các chức năng kết nối mạng Wi-Fi từ một bộ xử lý ứng dụng.

Mạch thu phát wifi ESP8266 12E có khả năng xử lý và lưu trữ mạnh mẽ cho phép nó được tích hợp với các bộ cảm biến, vi điều khiển và các thiết bị ứng dụng cụ thể khác thông qua GPIOs với một chi phí tối thiểu và một PCB tối thiểu.

ESP8266 12E có kích thước nhỏ gọn, ra chân đầy đủ của IC ESP8266, mạch được thiết kế và gia công chất lượng tốt với vỏ bọc kim loại chống nhiễu và anten Wifi PCB tích hợp cho khoảng cách truyền xa và ổn định.

Thông số kỹ thuật:

- IC chính: Wifi SoC ESP8266
- Điện áp sử dụng: 3.0V~3.6V (Optimal 3.3V)
- Working current: $\approx 70\text{mA}$ (170mA MAX), standby $< 200\mu\text{A}$

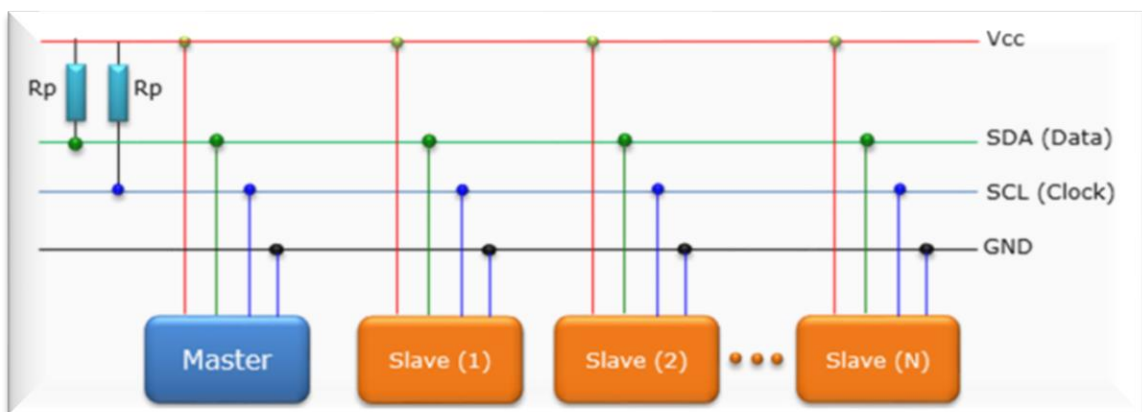
- Dòng tiêu thụ 10 μA , dòng điện năng chờ $< 5 \mu\text{A}$
- 30 chân (10 GPIO, every GPIO can be PWM, I2C, 1-wire)
- MCU Frequency: 80-160 MHz, 32-bit micro MCU
- SRAM size: 36 KB
- ROM size: 4 MB (SPI External Flash)
- Antena on PCB
- Transmission data rate: 110-460800bps
- 10bit precision ADC pinout on board (0~1V)
- WiFi @ 2.4 GHz, supports WPA / WPA2 security mode
- Wi-Fi Connectivity (802.11 b/ g/ n)
- Support UART/GPIO data communication interface
- Support STA/AP/STA+AP 3 working modes
- Built-in TCP/IP protocol stack, maximum 5 clients
- Nhiệt độ làm việc: $-40^{\circ}\text{C} \sim +125^{\circ}\text{C}$
- Kích thước: 24 x 16 x 3mm
- Trọng lượng: 4g

1.3.2 Giao tiếp I2C

I2C là tên viết tắt của cụm từ tiếng anh “Inter-Integrated Circuit”. Nó là một giao thức giao tiếp được phát triển bởi Philips Stoiiconductors để truyền dữ liệu giữa một bộ xử lý trung tâm với nhiều IC trên cùng một board mạch chỉ sử dụng hai đường truyền tín hiệu. Trên VDK sử dụng 2 đường SDA và SCL lần lượt là GPIO5 và GPIO4

Do tính đơn giản của nó nên loại giao thức này được sử dụng rộng rãi cho giao tiếp giữa vi điều khiển và mảng cảm biến, các thiết bị hiển thị, thiết bị IoT, EEPROMs, v.v ...

Đây là một loại giao thức giao tiếp nối tiếp đồng bộ: các bit dữ liệu được truyền từng bit một theo các khoảng thời gian đều đặn được thiết lập bởi một tín hiệu đồng hồ tham chiếu.



Sơ đồ BUS I2C

Theo sơ đồ trên có thể thấy, có thể kết nối nhiều thiết bị, cảm biến trên cùng một đường bus I2C, khi truyền chỉ cần phân biệt bằng địa chỉ của chúng.

1.3.3 Màn hình OLED I2C

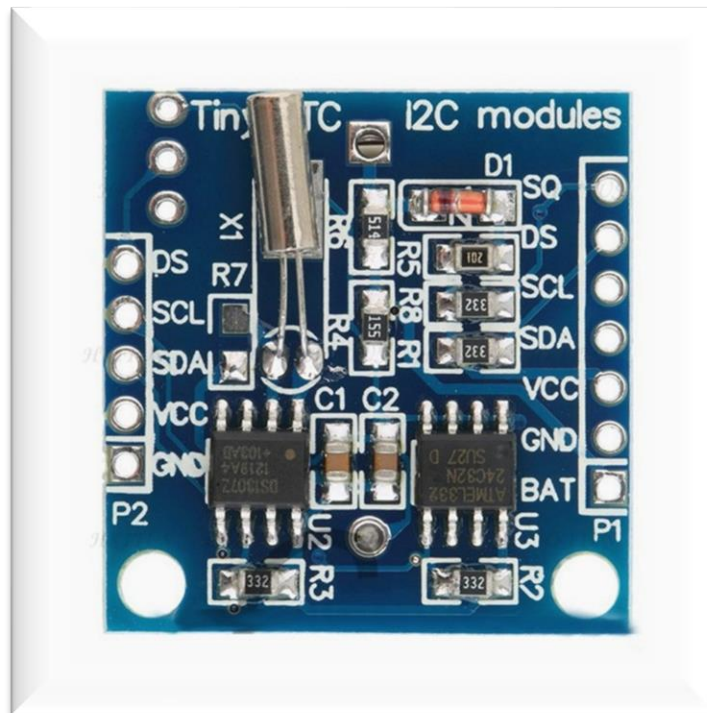


Màn hình OLED I2C

Là module màn hình đồ họa với độ phân giải 128x64. Module này nhỏ gọn và tiêu thụ năng lượng thấp, nhưng có chất lượng hiển thị tốt. Ngoài ra module sử dụng bus I2C nên chỉ cần 2 đường tín hiệu SCL và SDA, với tốc độ truyền nhận cao.

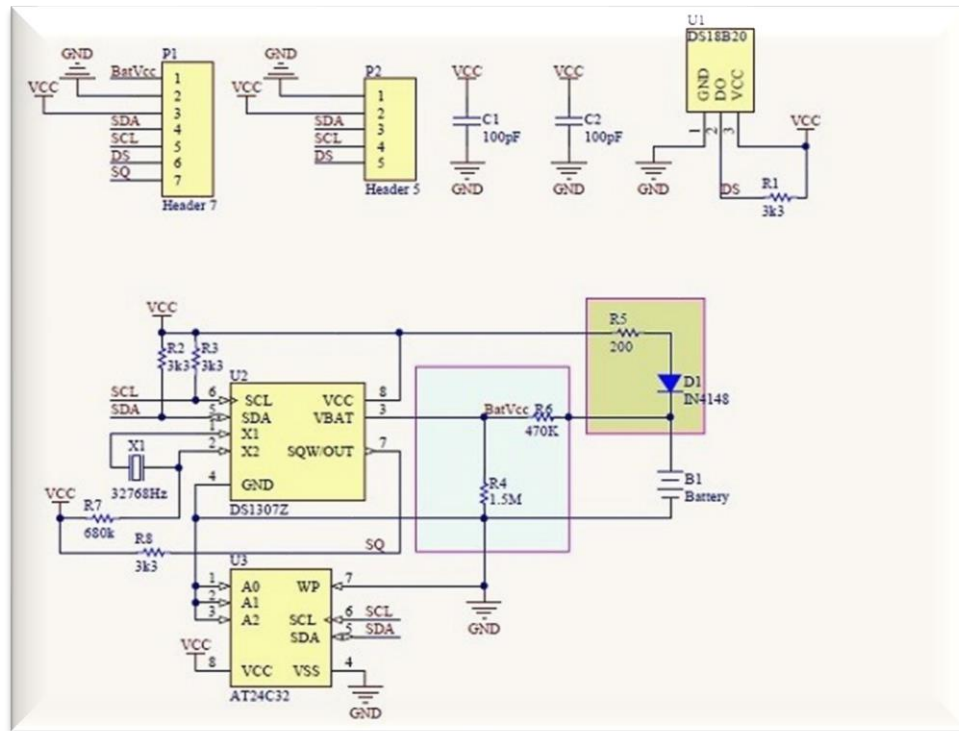
1.3.4 Module DS1307 + AT24C32 Tiny RTC

Là module nhỏ gọn tích hợp cả 3 IC gồm DS1307, AT24C32, và DS18B20 trên cùng một mạch. Cho phép thực hiện 3 chức năng cùng lúc: cung cấp thông tin thời gian thực, lưu trữ dữ liệu, và đo nhiệt độ



Module DS1307 + AT24C32 Tiny RTC

Sơ đồ mạch:



Sơ đồ mạch

3 IC này có thể đồng thời hoạt động đồng thời với nhau do chúng hoạt động độc lập và không xung đột về cổng giao tiếp. DS1307 sử dụng giao tiếp I2C, AT24C32 sử dụng giao tiếp I2C và DS18B20 sử dụng giao thức 1 – Wire. Điều này cho phép thực hiện cả 3 chức năng trên một mạch duy nhất, cung cấp thông tin thời gian thực, lưu trữ dữ liệu và đo nhiệt độ một cách đồng thời.

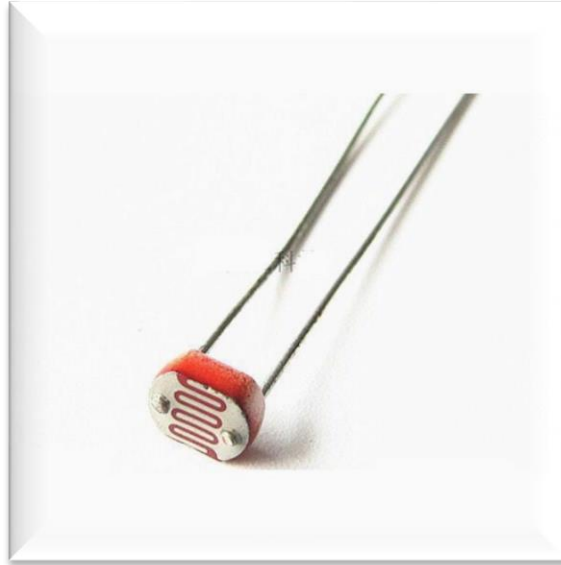
– Các chân tín hiệu:

- SDA (Serial Data Line) : Kết nối với chân dữ liệu (Data) trên mạch điều khiển chủ (ví dụ: Arduino) thông qua giao thức I2C
- SCL (Serial Clock Line): Kết nối với chân xung (Clock) trên mạch điều khiển chủ thông qua giao thức I2C
- INT/ SQW (Interrupt/Square Wave) : Chân này có thể được gán lại cho các chân khác trên mạch điều khiển chủ

Địa chỉ của DS1307 và AT24C32 tương ứng là:

- DS1307: 0x68
- AT24C32: 0x50

1.3.5 Quang trở



Quang trở

Thông số kỹ thuật:

- Điện áp tối đa: 150Vdc.
- Công suất tiêu thụ tối đa: 100mW.
- Nhiệt độ hoạt động: -30°C đến +70°C.
- Độ rọi: 20-45 K(ohm).
- Thời gian đáp ứng: 20ms.

2 PHÂN TÍCH THIẾT KẾ

2.1 Mô hình hệ thống

Thiết bị nhúng gồm những phần chính:

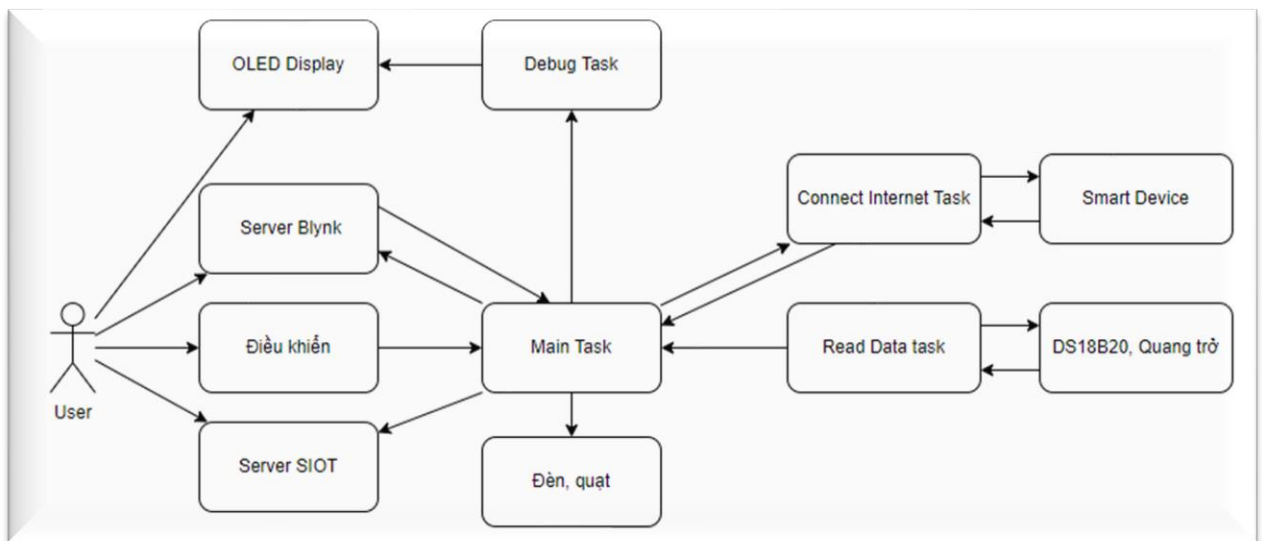
- Interface: các hàm giao diện để triệu gọi các dịch vụ từ đám mây (SIoT Spherem, Blynk).
- Core: các dịch vụ hạ tầng cơ bản cho thiết bị để khởi động hoạt động và kết thúc hoạt động.
- Appliances: Các ứng dụng hoạt động.

Hệ thống SIoT Sphere dịch vụ đám mây để lưu trữ tập trung mọi số liệu từ các thiết bị nhúng.

- Database: dữ liệu tập trung
- API Generator: sinh các WebAPI dựa trên các trường dữ liệu do lập trình viên khai báo.
- WebAPI: giao diện Restful kết nối với các thiết bị. Đề tài này chỉ liên quan tới SIoT Platform và được hợp chuẩn để kết nối với WebAPI của SIoT Sphere

Hệ thống Blynk cho phép tạo ra thiết bị ảo trên đám mây.

- Cung cấp các Datastream ánh xạ 1 – 1 với thiết bị vật lý.
- Cung cấp WebAPI để điều khiển.



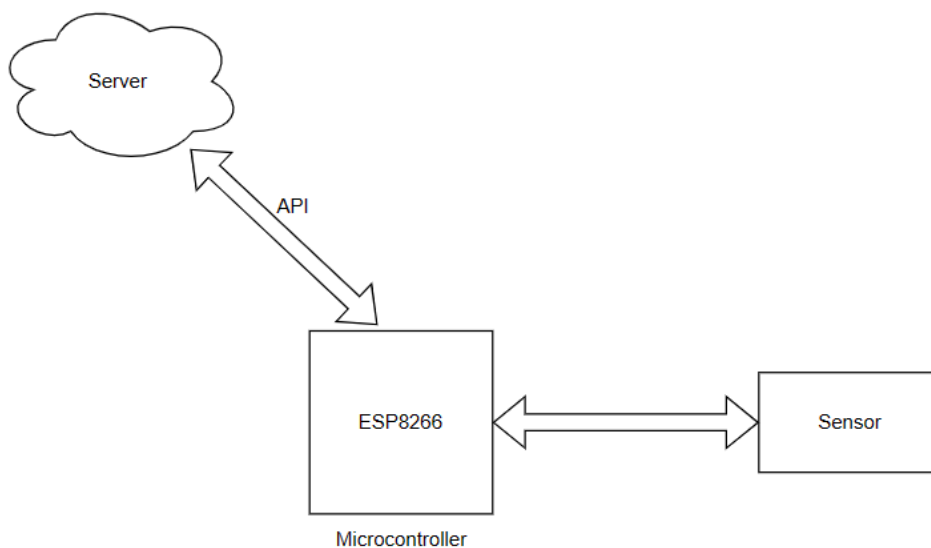
Sơ đồ hoạt động hệ thống

2.2 Thiết kế các giải pháp kết nối

2.2.1 Đặt vấn đề

Với mỗi thiết bị, để có thể chia sẻ dữ liệu trong mạng IOT đều cần phải kết nối vào Internet. Với Siot core sử dụng VĐK Esp8266 có hỗ trợ chuẩn kết nối Wifi ở băng tần 2,4GHz, vì vậy việc sử dụng thuận tiện và dễ dàng hơn. Tuy nhiên các yếu tố ảnh hưởng đến tốc độ đường truyền, smart config vẫn cần phải được xử lý.

Với những thiết bị được đặt ở vị trí cố định, cần phải đảm bảo có cơ chế tự động kết nối lại khi có các sự cố bất ngờ xảy ra như mất điện, mất mạng. Ngoài ra, với đường truyền kém, hoặc không ổn định cũng cần phải có cơ chế để thay đổi, lựa chọn đường truyền tốt hơn



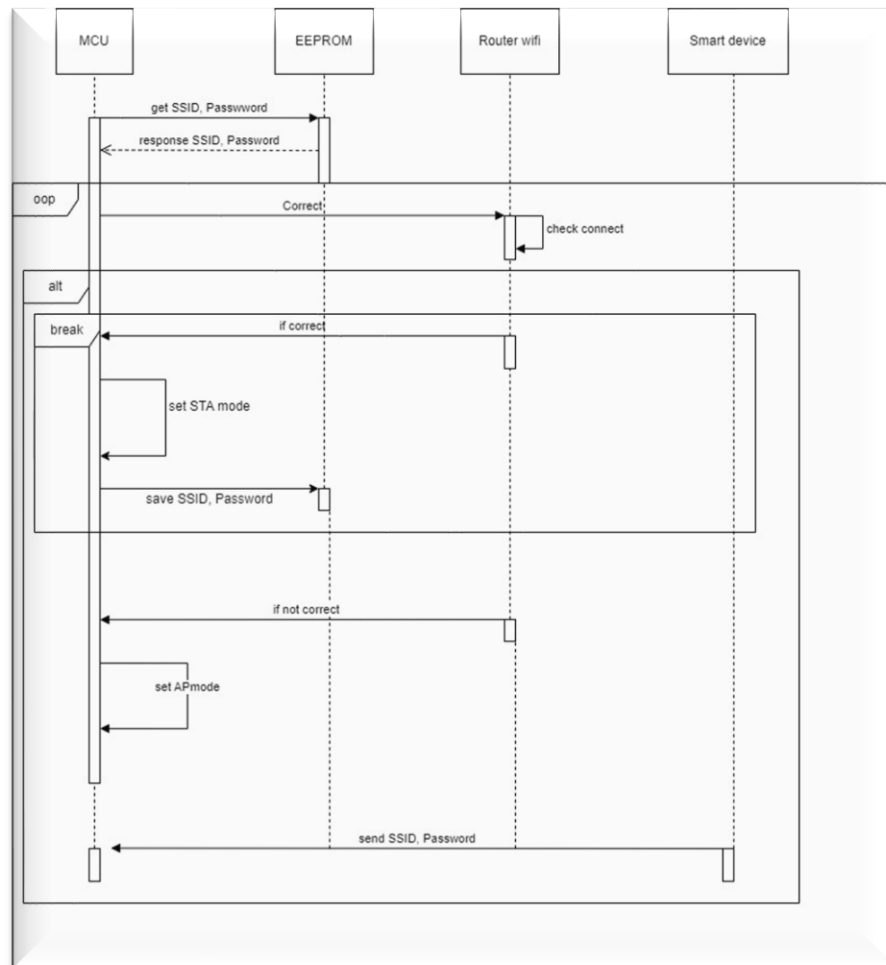
Mô hình chung của hệ thống

Mô hình trên gồm 3 phần chính:

- Khối core: Chức năng điều khiển chính của mạch, dùng vi điều khiển ESP8266 – là một dòng vi mạch Wifi giá rẻ sản xuất bởi Espressif.
- Khối ngoại vi, sensor: Khối này là các thiết bị, các loại cảm biến có thể giao tiếp với VĐK, vi điều khiển có thể đọc được giá trị từ chúng qua các chuẩn giao tiếp thông dụng như I2c, UART, SPI, USB, Analog, ...
- Khối server: Khối này sẽ giao tiếp với VĐK thông qua các hàm API, dữ liệu từ VĐK có thể đưa lên qua các phương thức GET, POST, UPDATE,...Dữ liệu như nhiệt độ, độ ẩm sẽ được lưu trữ cho tùy mục đích sử dụng

Ưu điểm của mô hình trên có thể dễ dàng nâng cấp mở rộng. Số lượng cảm biến, ngoại vi có thể thay đổi tùy ý, vì thế có thể đa dạng dữ liệu được thu thập. Ngoài ra, VĐK có thể dễ dàng đưa dữ liệu ra nhiều khối dữ liệu, server khác nhau, chỉ cần có API do phía server cung cấp

2.2.2 Mô hình kết nối với mạng Wifi



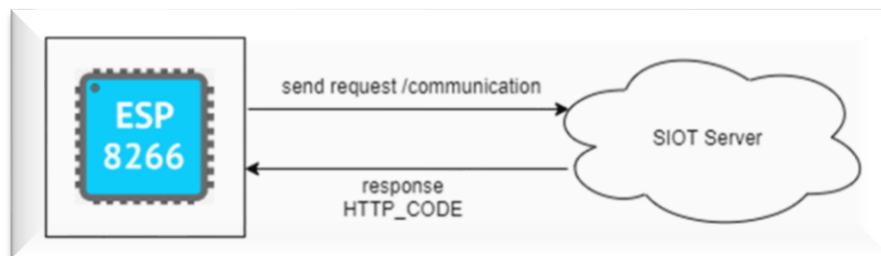
Sơ đồ kết nối internet

Mô hình trên gồm 4 đối tượng chính: VDK, Bộ nhớ, Router wifi, Smart device (Laptop, Smart Phone). Luồng hoạt động của mô hình như sau:

- Khi VDK khởi động chương trình để kết nối vào Internet, nó kiểm tra trong bộ nhớ (Flash/EEPROM) thông tin về SSID và Password của một mạng wifi đã tồn tại.
- Với thông tin lấy được, nếu kết nối thành công, VDK điều khiển sẽ log ra tên Wifi mà nó kết nối.
- Nếu bước 2 không thành công, VDK sẽ về chế độ STA mode, nó sẽ đóng vai trò là webbase-server cho phép các thiết bị khác kết nối vào, và đưa thông tin về SSID và Password để VDK có thể kết nối với 1 mạng phù hợp.
- Quá trình lặp lại cho đến khi kết nối thành công. Thông tin sẽ được cập nhật lại bộ nhớ. Vì vậy, đảm bảo được tính năng cần tự động kết nối lại (khi có sự cố xảy ra).

2.3 Kết nối VDK với SioT Platform và nền tảng Blynk

Sau khi VDK kết nối được với Internet, VDK sẽ được kết nối với Siot server để thực hiện các việc quan trọng như: lấy mã định danh, cập nhật các giá trị của sensor lên hệ thống, ... Dưới đây là mô hình của bài toán.



Mô hình kết nối VDK với Server

Việc kết nối với Blynk cũng tương tự. Khi kết nối thành công với Blynk, tín hiệu “Online” sẽ được hiển thị lên hệ thống.

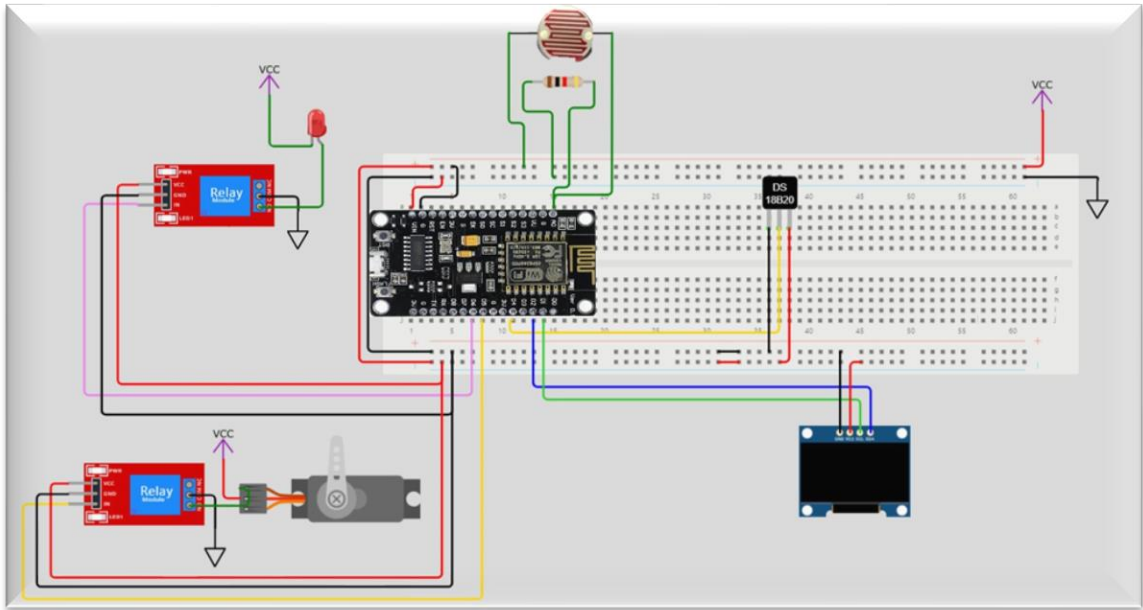
Khi kết nối thành công, các tác vụ khác với server mới diễn ra được, vì vậy cần kiểm tra kết nối trước mỗi lần sử dụng tài nguyên hay những tác vụ có liên quan đến các dịch vụ Internet.

2.4 Thiết kế các gói

- Common: chứa các hàm read, write eeprom để các class khác có thể sử dụng.
- ConnectInternet: cung cấp các hàm, các thuộc tính phục vụ việc kết nối vào mạng Wifi.
- ConnectServer: các hàm để truyền nhận, thông tin từ một server thông qua các đường dẫn URL, các API mà phía server đang quy định
- Debug: cung cấp các phương thức để hiển thị, log thuận tiện cho phát triển thông qua việc hiển thị ra màn hình OLED
- Sensor: gói làm việc với các hoạt động đọc, ghi dữ liệu từ các cảm biến
- Switch: gói quản lý các việc bật tắt các thiết bị (thật và ảo)

2.5 Thiết kế sơ đồ mạch

- Sơ đồ mạch nhúng:



Sơ đồ mạch

- Các Datastream trên Blynk:

Id	Name	Alias	Color	Pin	Data Type	Units	Is Raw	Min	Max
1	NHIETDO	NHIETDO	Red	V0	Double	°C	false	0	100
2	ANHSANG	ANHSANG	Light Blue	V1	Integer		false	0	400
3	BULL	BULL	Dark Green	V2	Integer		false	0	1
4	FAN	FAN	Dark Purple	V3	Integer		false	0	1
5	BUTTON_BULL	BUTTON BULL	Light Purple	V4	Integer		false	0	1
6	BUTTON_FAN	BUTTON FAN	Brown	V5	Integer		false	0	1
7	CHANGEMODE	CHANGEMODE	Purple	V6	Integer		false	0	1
8	Digital Pin 0	Digital Pin 0	Dark Green	0	Integer		false	0	1

Danh sách Datastream

3 PHÁT TRIỂN MÃ NGUỒN

3.1 Cài đặt kết nối Internet cho thiết bị nhúng.

Cấu trúc class ConnectInternet. Vai trò của class này là cung cấp giao thức để kết nối Internet.

Khoi gọi hàm connect() sẽ gọi đến phương thức autoConnect() của bộ thư viện WifiManager.

```
13 public:
14     char *ssid;
15     char *pass;
16     // Hàm khởi tạo
17     ConnectInternet();
18     ConnectInternet(char *ssid, char *pass);
19     // Kết nối và kiểm tra kết nối
20     void connect();
21     bool isConnected();
22     bool resetConnect();
23     // Get, set thuộc tính
24     void setSSID(char *ssid);
25     void setPass(char *pass);
26     String getSSID();
27     String getPass();
28     };
```

Cấu trúc class ConnectInternet

3.2 Kết nối với server SioT

Class ConnectServer có các phương để kết nối tới SioT server. Phương thức communicationServer dùng để lấy dữ liệu từ phía thiết bị gửi lên server SioT.

```
enum{
    GET,
    POST,
    UPDATE,
    PUT,|
    DELETE
};

class ConnectServer
{
private:
    /* data */
public:
    ConnectServer(/* args */);
    ~ConnectServer();
    bool isServerConnected(void);
    int getVersion(void);
    String getBoardId();
    int pushData();
    int communicationSever(String URI, String value, String& response,int method);
};
```

Class ConnectServer

3.3 Gói Debug

Hiện thị thông tin ra màn hình OLED I2C

```
class Debug
{
public:
    Debug();
    ~Debug();
    static void LOG_TO_SCREEN(int x, int y, String string);
    static void LOG_PROGRESS_UPDATE(int current, int sum);
    static void SCREEN_WELCOME();
    // Data members

private:
    // Data members
};
```

Class Debug

3.4 Gói Switch

Kết nối với hệ thống Blynk và điều khiển các công tắc. Cũng như việc giám sát dữ liệu real – time.

```
23  class Switch
24  {
25  public:
26      Switch();
27      ~Switch();
28      void connectToBlynk();
29      void sendDataToBlynk(float temp, float light);
30      void changeDeviceBull(int level);
31      void changeDeviceFan(int level);
32      void changeMode(int level);
33  };
```

Class Switch

3.5 Gói Sensor đọc dữ liệu

```
22  class Sensor
23  {
24  public:
25      Sensor();
26      ~Sensor();
27      void init();
28      char *convertDayToStr(SensorData data);
29      char *convertTimeToStr(SensorData data);
30      char *convertAllToStr(SensorData data);
31      float readTemperature();
32      SensorData readDataSensor();
33      void sendDataToPC(SensorData data);
34  };
```

Gói Sensor

3.6 Cấu trúc bộ nhớ EEPROM

4 TRIỂN KHAI

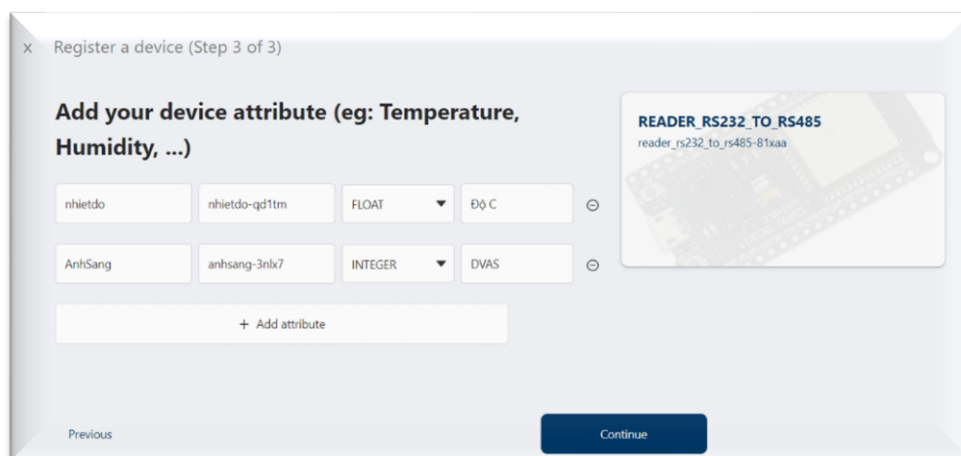
4.1 Tạo thiết bị trên SioT

Để có thể đẩy dữ liệu cập nhật lên server trước hết cần tạo một thiết bị mới trên cơ sở dữ liệu. Để có thể thêm thiết bị cần truy cập vào đường dẫn: <http://siot.soict.ai/> cần tạo tài khoản nếu chưa có, chọn vào mục “Add device”



Giao diện tạo thiết bị mới trên siot core

Điền các thông tin cần thiết, sau đó tạo các trường thông tin cho các dữ liệu cảm biến cần thiết của thiết bị.



Các thông tin cảm biến của thiết bị









Theo ví dụ ở trên, thiết bị trên gồm 2 giá trị gồm:

- Giá trị về nhiệt độ, kiểu số thực, đơn vị là độ C
- Giá trị về cường độ ánh sáng, kiểu số nguyên, đơn vị là DVAS

Sau khi hoàn thành các bước, các thông tin về API, authorization được hiển thị.

4.2 Tạo thiết bị trên Blynk

- Các Datastream trên Blynk

Id	Name	Alias	Color	Pin	Data Type	Units	Is Raw	Min	Max
1	NHIETDO	NHIETDO		V0	Double	°C	false	0	100
2	ANHSANG	ANHSANG		V1	Integer		false	0	400
3	BULL	BULL		V2	Integer		false	0	1
4	FAN	FAN		V3	Integer		false	0	1
5	BUTTON_BULL	BUTTON BULL		V4	Integer		false	0	1
6	BUTTON_FAN	BUTTON FAN		V5	Integer		false	0	1
7	CHANGEMODE	CHANGEMODE		V6	Integer		false	0	1
8	Digital Pin 0	Digital Pin 0		0	Integer		false	0	1

Danh sách Datastream

- Token bên server Blynk

FIRMWARE CONFIGURATION

```
#define BLYNK_TEMPLATE_ID "TMPL6a4dLgmmy"  
#define BLYNK_TEMPLATE_NAME "Ngoc Thanh"  
#define BLYNK_AUTH_TOKEN "iIvspXRjmJ4-s7Gb9DW2oyJo7QHxFyEp"
```

Template ID, Device Name, and AuthToken should be declared at the very top of the firmware code.

- Dán vào trong file Switch.h

```
#ifndef _SWITCH_H_  
#define _SWITCH_H_  
  
#define BLYNK_TEMPLATE_ID "TMPL6a4dLgmmy"  
#define BLYNK_TEMPLATE_NAME "Ngoc Thanh"  
#define BLYNK_AUTH_TOKEN "iIvspXRjmJ4-s7Gb9DW2oyJo7QHxFyEp"
```

- Ảnh xạ chân ảo Blynk và thiết bị nhúng

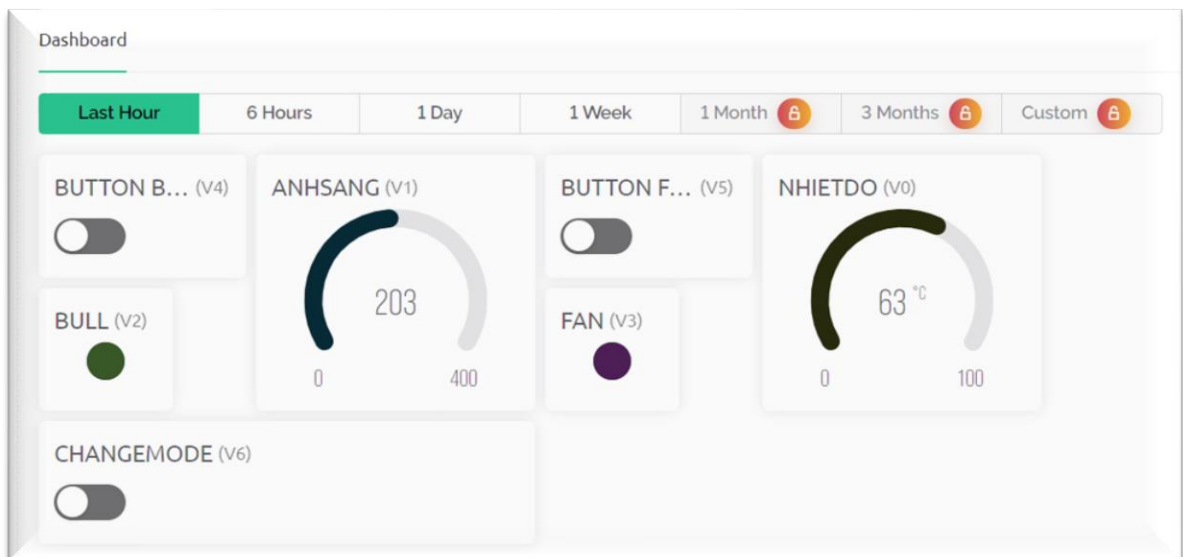
```

Switch.h
8 #define NHIETDO V0 // Dữ liệu nh
9 #define ANHSANG V1 // Dữ liệu ả
10 #define BULL V2 // Bóng đèn
11 #define FAN V3 // Quạt
12 #define BUTTON_BULL V4 // Nút đèn ả
13 #define BUTTON_FAN V5 // Nút quạt ả
14 #define BUTTON_CHANGE_MODE V6
15 #define BULB_PIN D5
16 #define FAN_PIN D6
17 #define MODE_ADDR 8
18
19 // Khai báo địa chỉ trên EEPROM
20 #define STATE_FAN_ADDR 0
21 #define STATE_BULL_ADDR 4

Sensor.h
8 #define ONE_WIRE_BUS 2
9 #define DLS_PIN A0
10 #define BULB_PIN D5
11 #define FAN_PIN D6
12 #define CHANGE_MODE D7
13
14 #define MIN_ADDR_DATA 12
15 #define MAX_ADDR_DATA 172
16
17 typedef struct
18 {
19     uint8_t light;
20     float temp;
21 } SensorData;
  
```

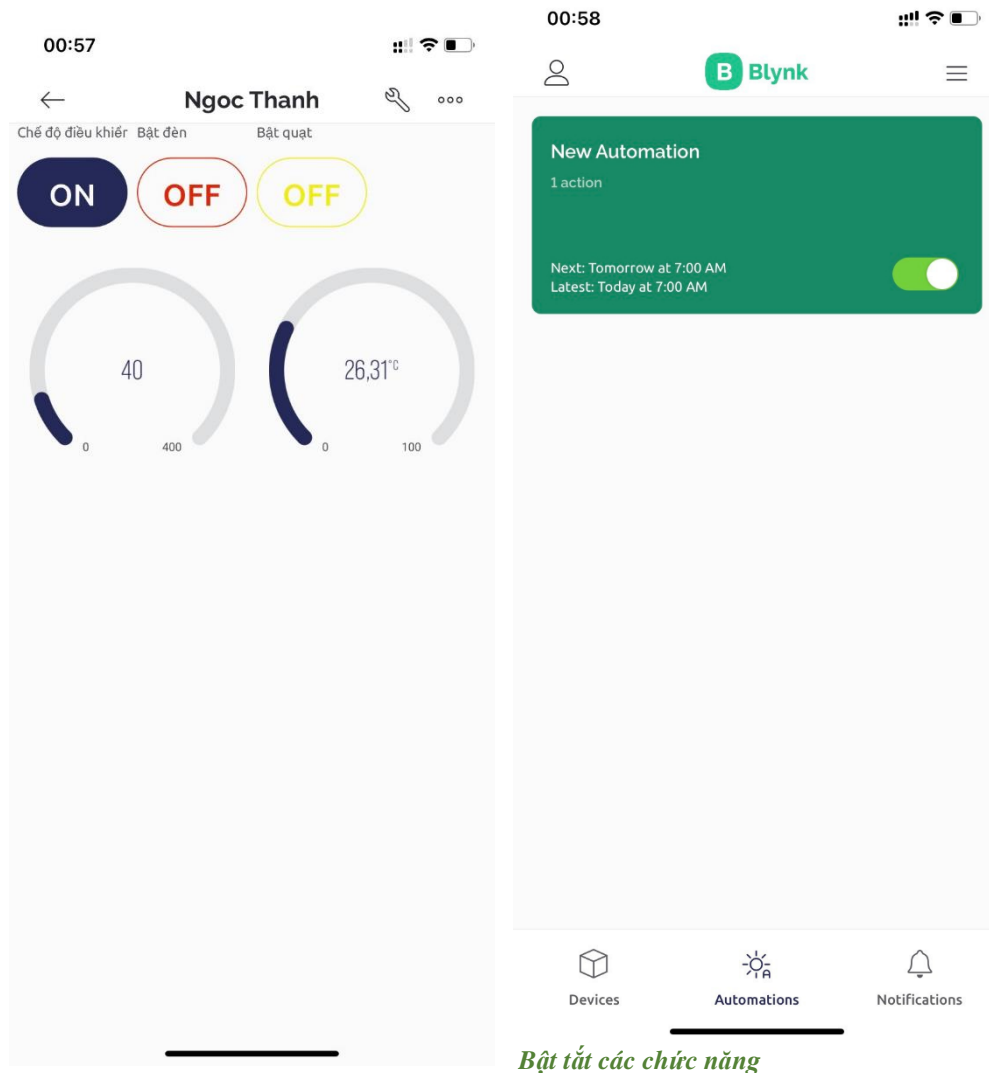
Ảnh xạ chân Blynk sang thiết bị

- Thiết kế Web Dashboard

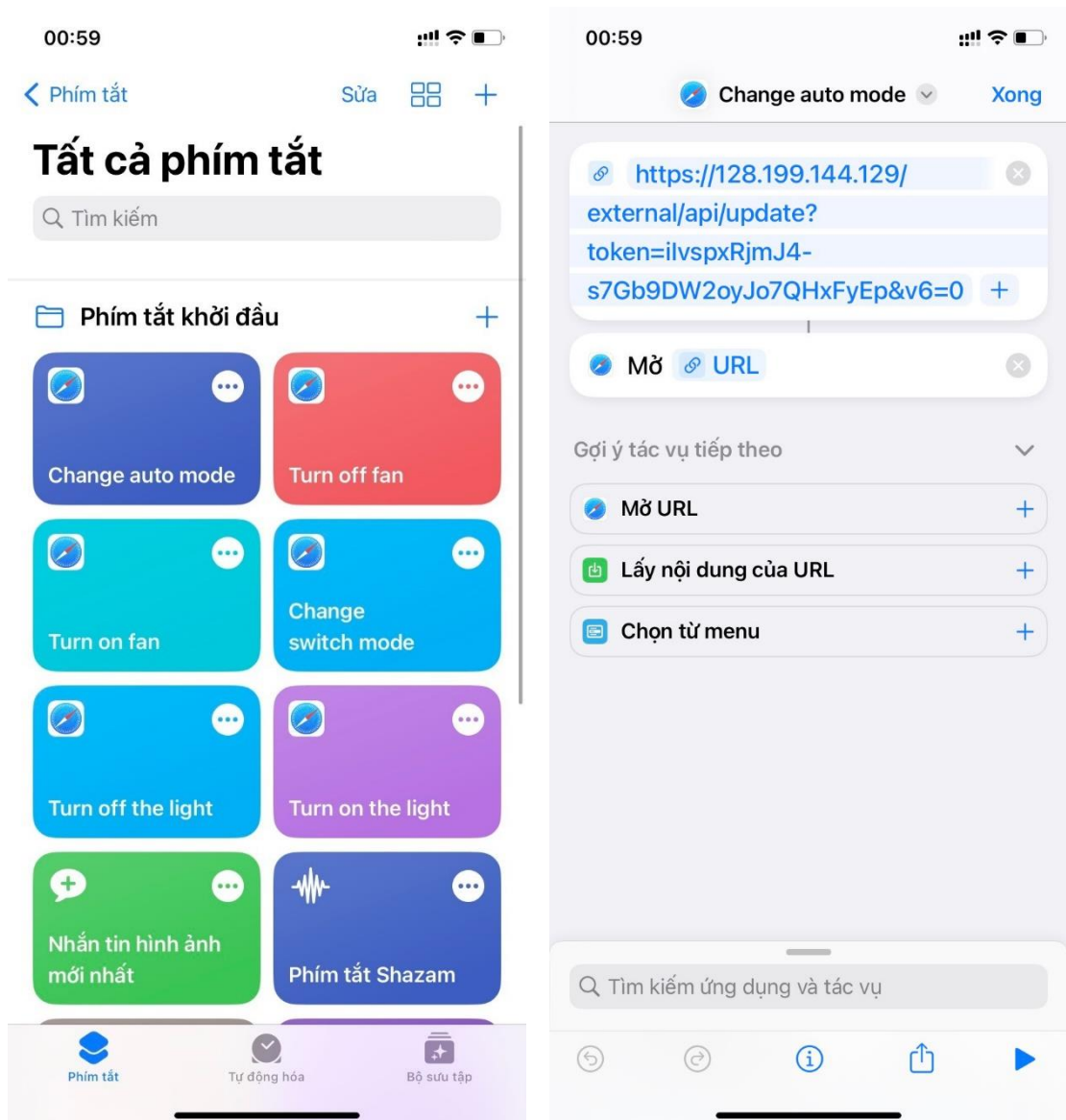


Giao diện web

- Thiết kế Mobile Dashboard và Tạo kịch bản tự động



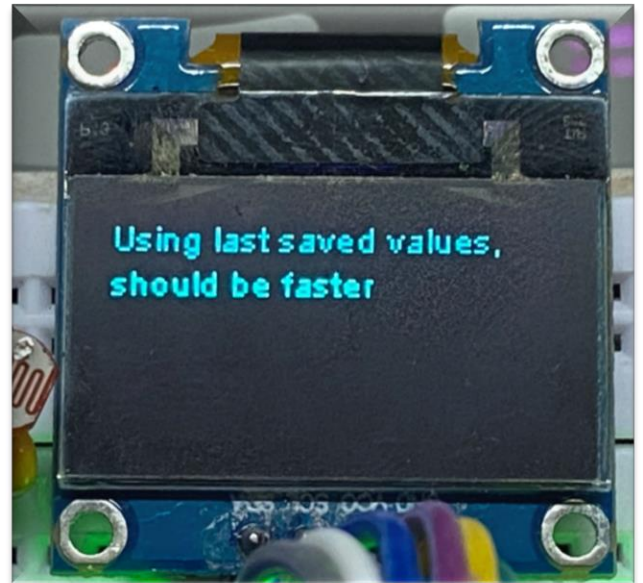
Dùng API do Blynk cung cấp để tích hợp vào trợ lý ảo Siri:



5 KẾT QUẢ THỰC HIỆN

5.1 Kết nối Internet

- Màn hình Welcome, sau đó sẽ tự động dò tìm và kết nối với mạng đã được lưu trong EEPROM từ trước

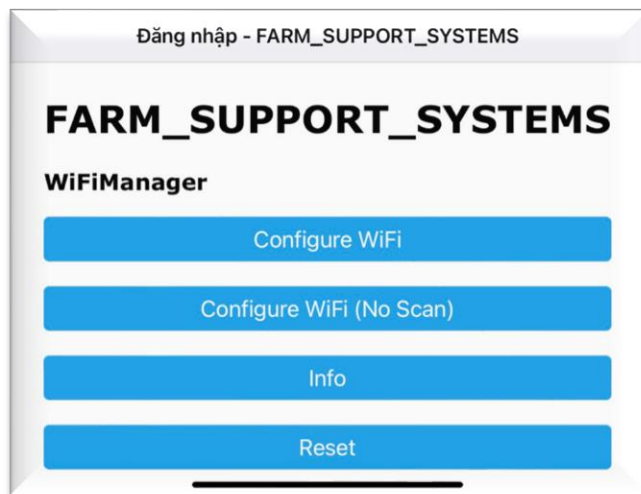


- Nếu không kết nối được, VDK sẽ bật HTTP server với SSID DEFAULT



FARM_SUPPORT_SYSTEMS		📶
Ngan Duong	🔒	📶
Phong 206	🔒	📶
Tầng 3A	🔒	📶

- Giao diện kết nối Internet, sau khi kết nối thành công sẽ chuyển qua hoạt động.

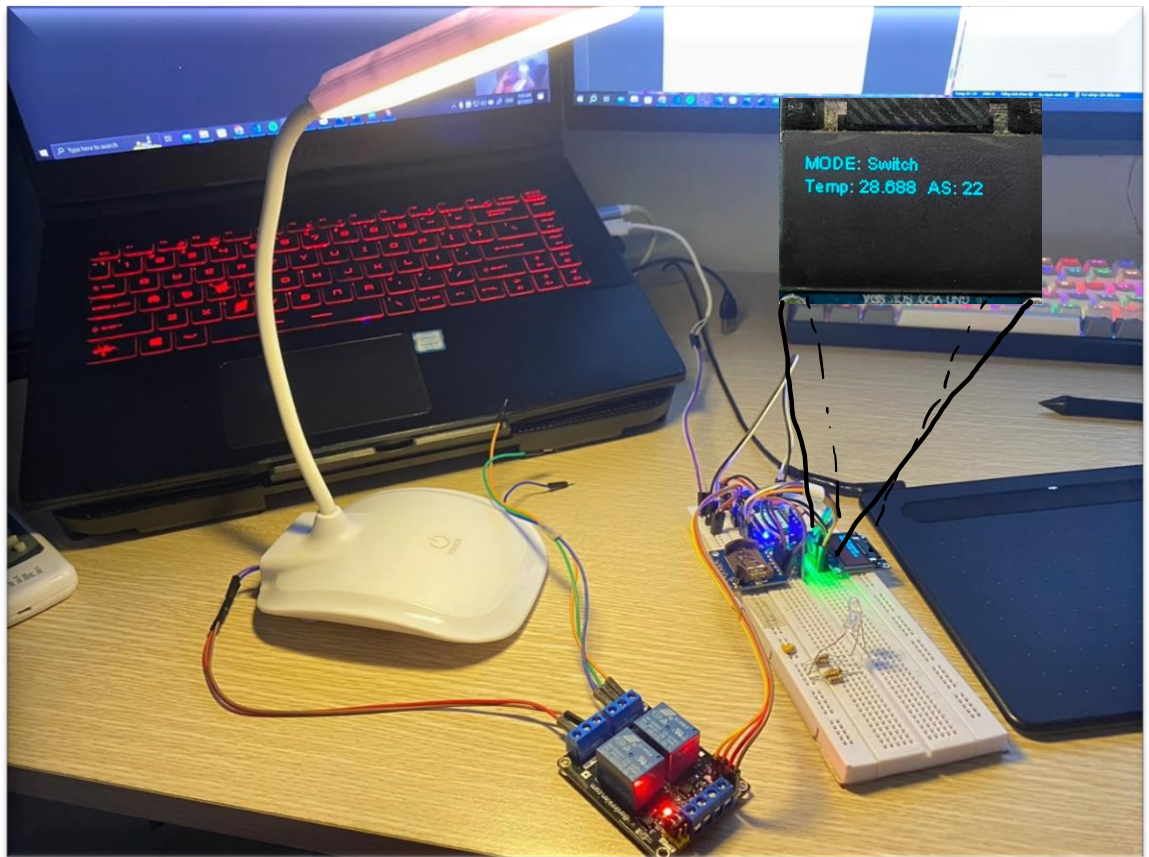


5.2 Các chế độ hoạt động

5.2.1 Chế độ tự động dùng cảm biến

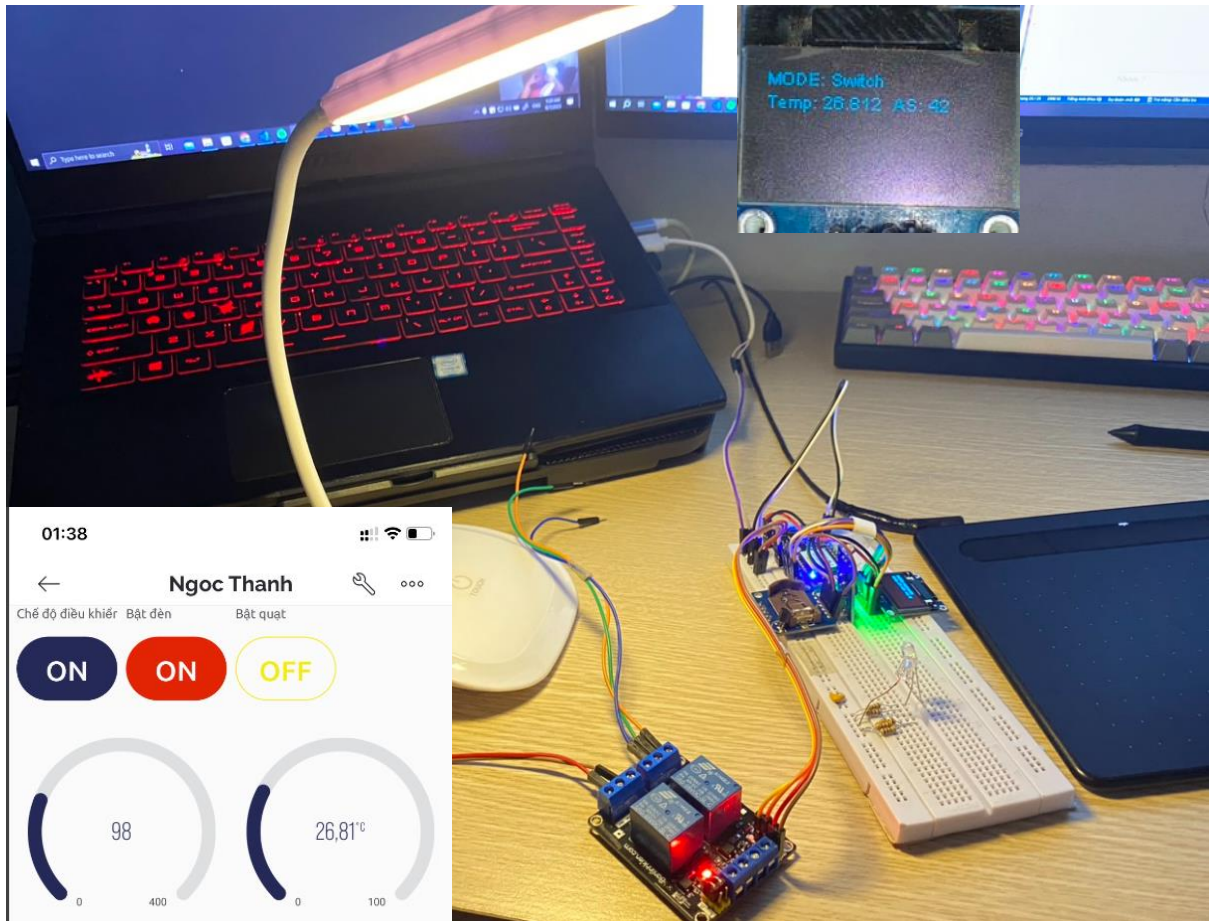
- Đọc dữ liệu từ cảm biến và lưu vào EEPROM (21 giá trị). Sau đó tính giá trị trung bình. Khi vượt ngưỡng với độ sai lệch ($0,25^{\circ}C$ với nhiệt độ, 5 DVAS với quang trở) thì tiến hành thay đổi trạng thái của thiết bị.
- Khi thiết bị thay đổi trạng thái thì trạng thái này cũng được đồng bộ với hệ thống Blynk.

Ví dụ: Chế độ tự động, cường độ ánh sáng thấp, đèn bật lên:



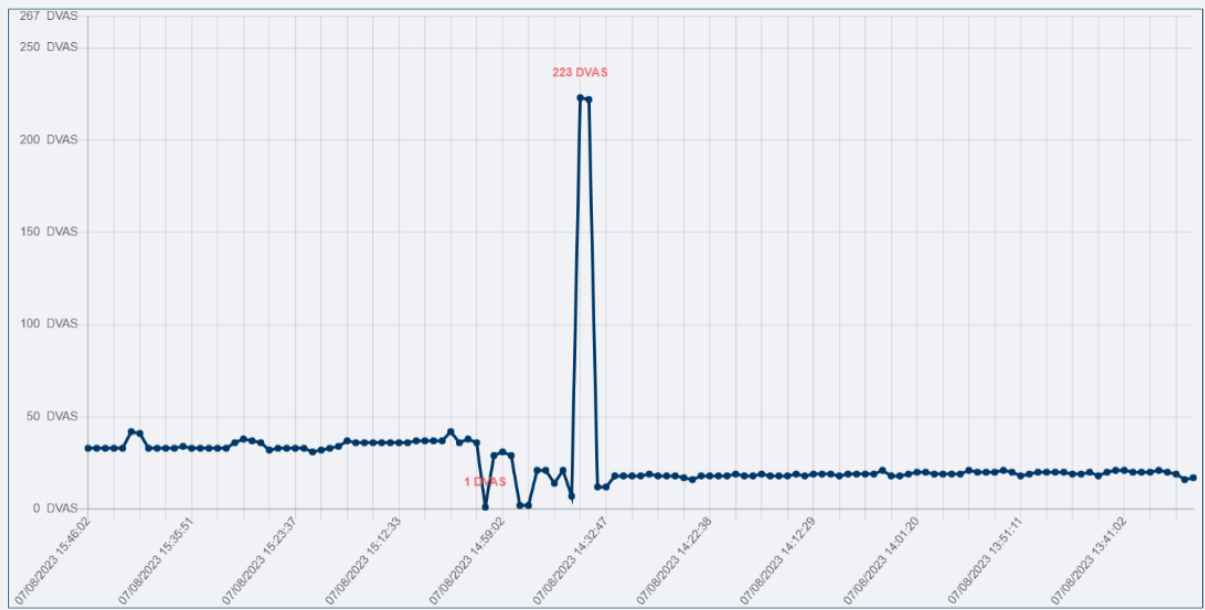
5.2.2 Chế độ điều khiển bằng công tắc

- Ở chế độ này, chúng ta có thể chủ động tắt, bật các thiết bị qua ứng dụng Blynk, chủ động hẹn giờ các thiết bị.



5.3 Kết nối với Server SioT và đẩy dữ liệu từ lên server qua WebAPI

Kết nối thành công với Siot server. Dữ liệu về nhiệt độ, ánh sáng sử dụng cảm biến DS18B20, được VDK đọc và update lên Siot Server, với tần suất 50s/lần.



Đồ thị dữ liệu ánh sáng theo thời gian

Name: AnhSang
Created at: 30/07/2023 16:27:33
Max value: 1024 DVAS Min value: 0 DVAS
Last active: 07/08/2023 15:46:02

2023-08-07 00:00:00 → 2023-08-07 23:59:59

Total: 129 in 6781 total records

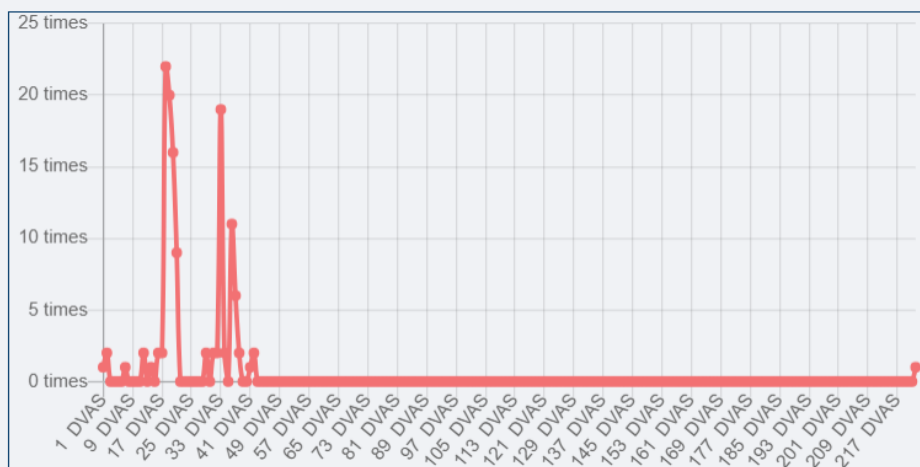
	Value (DVAS)	Created date
1	33	07/08/2023 15:46:02
2	33	07/08/2023 15:45:12
3	33	07/08/2023 15:44:21
4	33	07/08/2023 15:43:31
5	33	07/08/2023 15:42:40
6	42	07/08/2023 15:41:49
7	41	07/08/2023 15:40:56
8	33	07/08/2023 15:40:05

Bảng dữ liệu theo ngày

Frequently data:

2023-08-07 00:00:00

→ 2023-08-07 23:59:59



Đồ thị phân bố các giá trị về ánh sáng

DOANH MỤC THAM KHẢO

- [1] N. N. Thành, "Xây dựng hệ quản trị dữ liệu tập trung SIoT Sphere cho hệ thống nhúng," Hà Nội, 12/2020.
- [2] Blynk Incorporated, "https://docs.blynk.io/en/," 8 5 2023. [Online]. Available: <https://docs.blynk.io/en/>. [Accessed 8 8 2023].
- [3] Đ. C. Thuận, Bài giảng Hệ nhúng, Hà Nội, 2023.
- [4] DALLAS SEMICONDUCTOR, "DS18B20 Programmable Resolution 1-Wire® Digital Thermometer".
- [5] SOLOMON SYSTECH, "SSD1306 Advance Information 128 x 64 Dot Matrix OLED/PLED Segment/Common Driver with Controller," 2008.

PHỤ LỤC

Mã nguồn chương trình: <https://github.com/mxngocqb/Embedded-Systems>