AUSTON

Wi-Fi Remote Control Switch

Presented by

Ma May Kar Yar Lay

Mg Wai Yar Aung

21.4.2023

Introduction

- In this project , we use mobile application and wi-fi network to learn how to remotely control LED on/off .
- We can control not only in the wi-fi area but also far away from the host via cloud method.

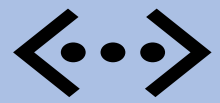
Objectives

- To understand how to remotely control home appliances (refers to LED in this project) via wi-fi module fundamentally.

- To accomplish above objective, we are using ESP8266 wi-fi module and installing software application on smart phones as user interface to control LED on/off via a wi-fi network.

Wi-Fi

Wi-Fi is a wireless networking technology that uses radio waves to provide wireless high-speed Internet access and allows devices such as computers (laptops and desktops), mobile devices (smart phones and wearables), and other equipment (printers and video cameras) to interface with the Internet.
It allows these devices and many more to exchange information with one another, creating a network.



Components

Hardware

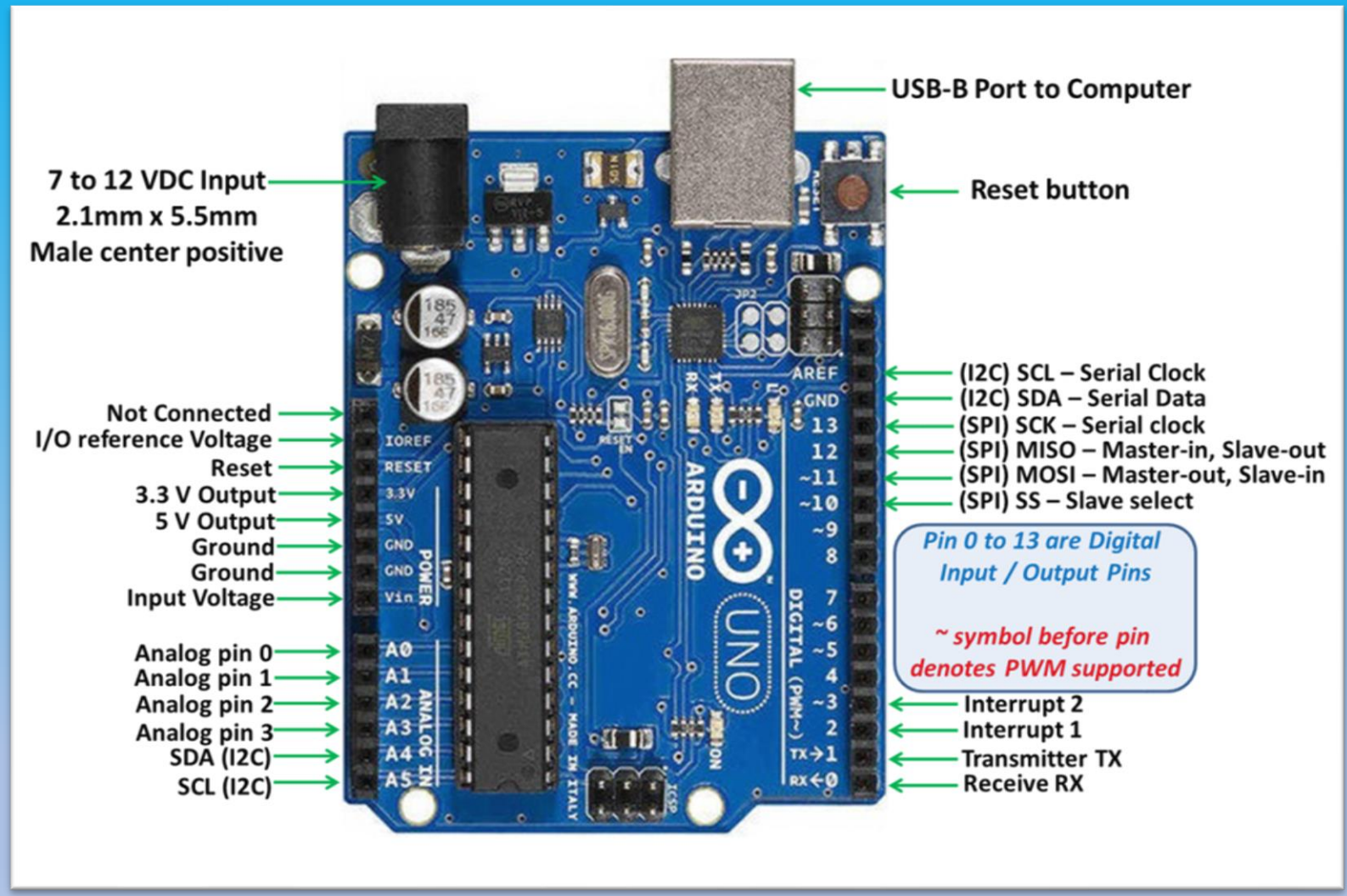
- Arduino UNO
- ESP8266 Wi-Fi Module
- LEDs
- Resistors
- Jumpers
- Breadboard
- Smart Phone

Software

- Remote XY

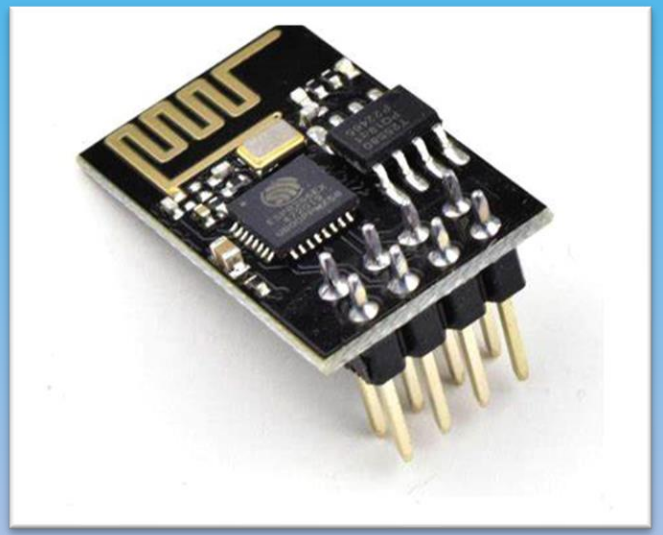
Arduino Uno

Microcontroller: ATmega328P
Operating Voltage: 5V
Input Voltage (recommended): 7-12V
Inout Voltage (limit): 6-20V
Digital I/O Pins: 14 (of which 6 provide PWM output)
PWM Digital I/O Pins: 6
Analog Input Pins: 6
DC Current per I/O Pin: 20 mA
DC current for 3.3V Pin: 50 mA
Flash Memory: 32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM: 2 KB (ATmega328P)
EEPROM: 1 KB (ATmega328P)
Clock Speed: 16 MHz
LED_BUILTIN: 13
Length: 68.6 mm
Width: 58.4 mm
Weight: 25 g



ESP8266-01 Wi-fi Module

- Wi-fi modules or wi-fi microcontrollers are used to send and receive data over Wi-Fi. They can also accept commands over the Wi-Fi. They are used for communications between devices. They are most commonly used in the field of Internet of Things.
- ESP8266 is the most widely used Wi-Fi module. It is a low-cost microchip with a full TCP/IP stack and microcontroller capability, produced by Espressif Systems.
- This small module allows microcontrollers to connect to a Wi-Fi network and make simple TCP/IP connections.
- The layers of the TCP/IP model: network access, internet, transport, and application. Used together, these layers are a suite of protocols.
- It can also work in two modes: station mode, where it connects to an existing wi-fi hotspot, or access point mode, where it creates its own wi-fi network.



ESP8266-01 Wi-fi Module Specification

- 2.4 GHz Wi-Fi (802.11 b/g/n, supporting WPA/WPA2).
- General-purpose input/output (16 GPIO).
- Inter-Integrated Circuit (I^2C) serial communication protocol.
- Analog-to-digital conversion (10-bit ADC).
- Serial Peripheral Interface (SPI) serial communication protocol.
- I^2S (Inter-IC Sound) interfaces with DMA(Direct Memory Access) (sharing pins with GPIO).
- UART (on dedicated pins, plus a transmit-only UART can be enabled on GPIO2).
- Pulse-width modulation (PWM)

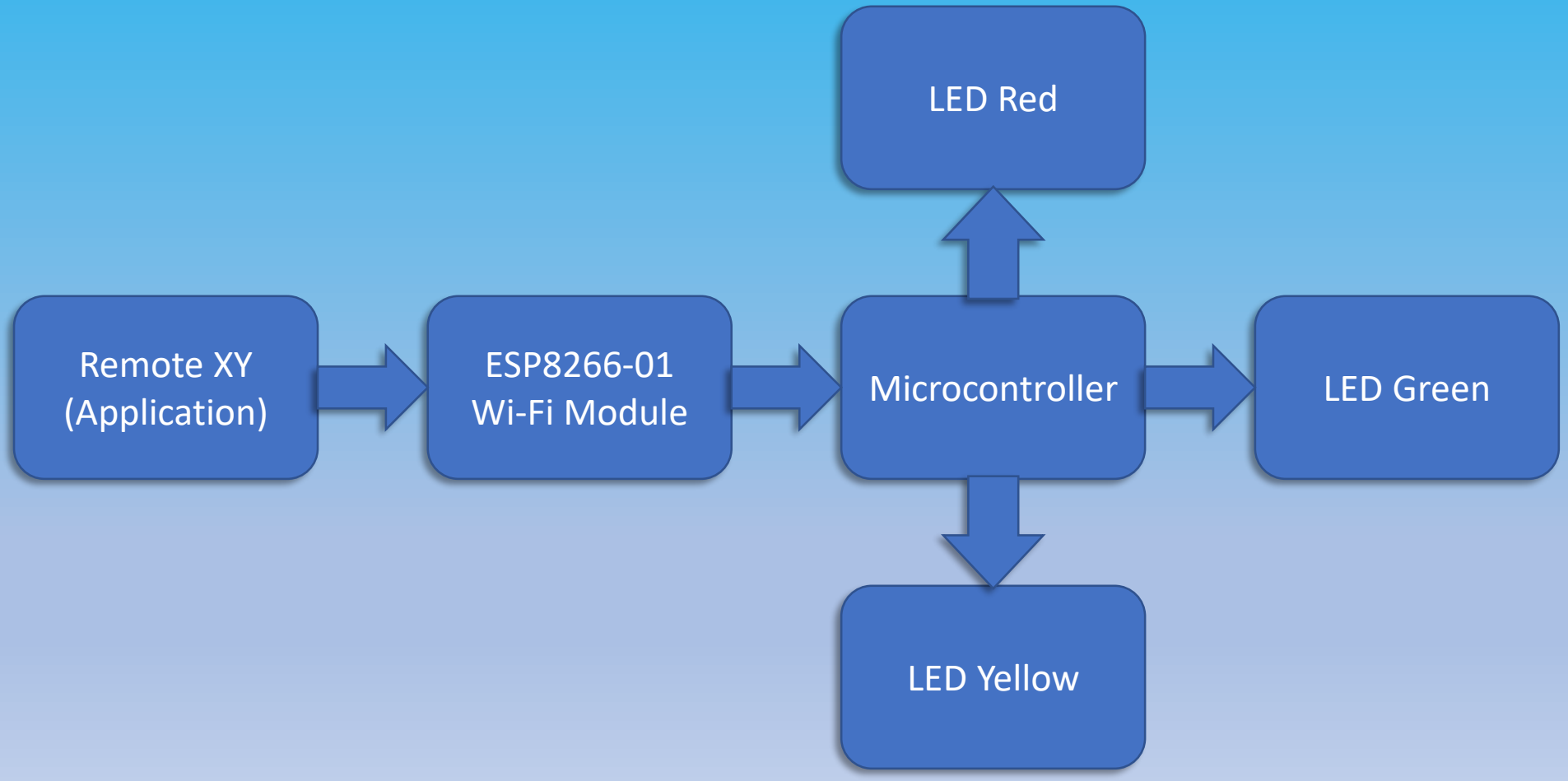
ESP8266-01 Wi-fi Module Pin Description

- VCC – Power Pin(+3.3V;can handle up to 3.6V)
- GND: – Ground Pin
- RST: – Reset Pin
- EN(CH_PD): – Enable Pin.
- TX: – Serial Transmit Pin of UART.
- RX: – Serial Receive Pin of UART.
- GPIO0 & GPIO2: – General Purpose I/O Pins. It also known as TX/RX pins are used for Programming the module or for serial I/O purpose.

ESP8266-01 Wi-fi Module Application

- (1) Internet of Things
- (2) Network consumer equipment
- (3) Measurement
- (4) Building automation
- (5) Home automation
- (6) Smart home gateway
- (7) Intelligent lighting
- (8) Smart Plugs and Lights
- (9) Baby Monitor
- (10) Sensor Network
- (11) Industrial Control

Block Diagram

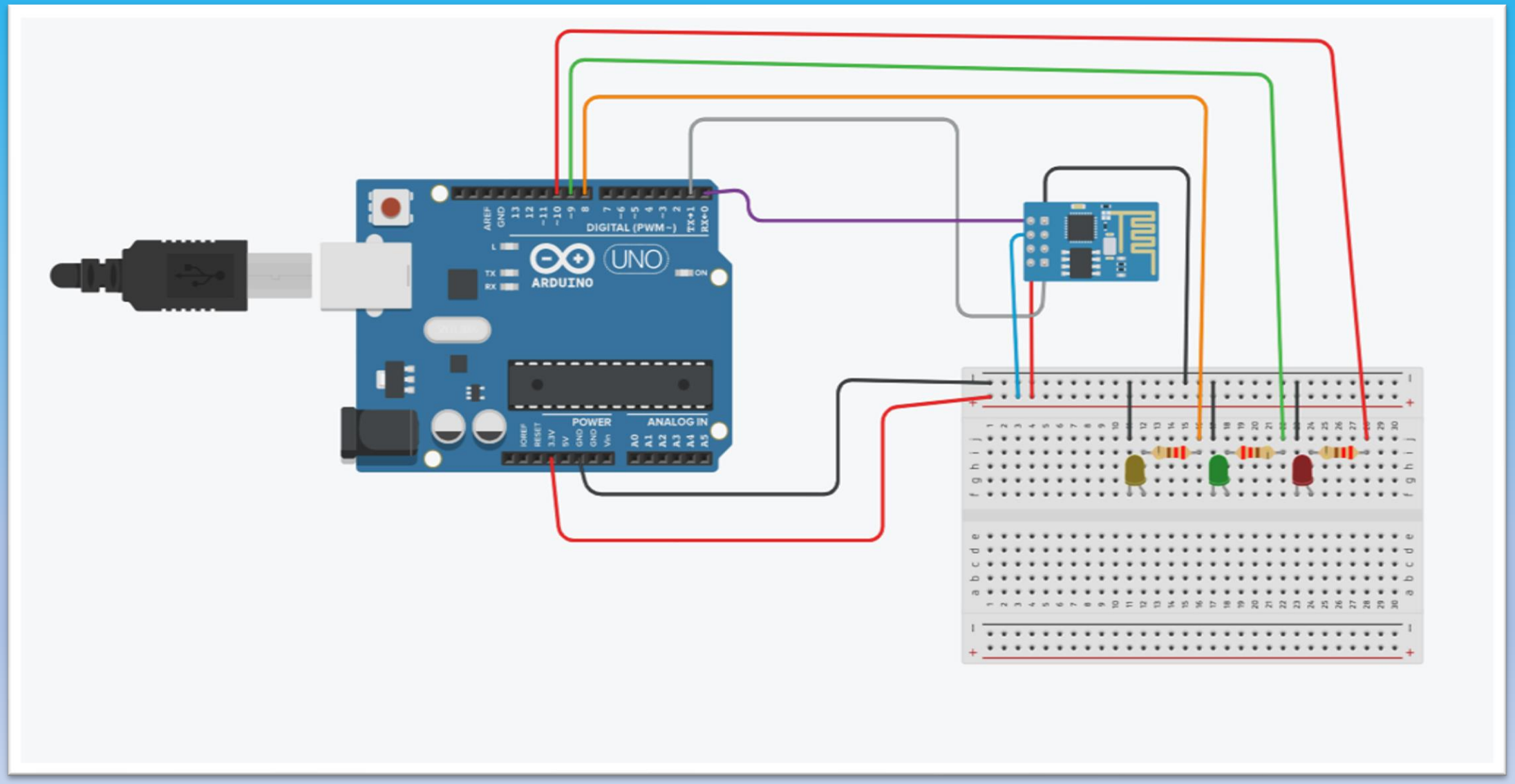


Pin Connection

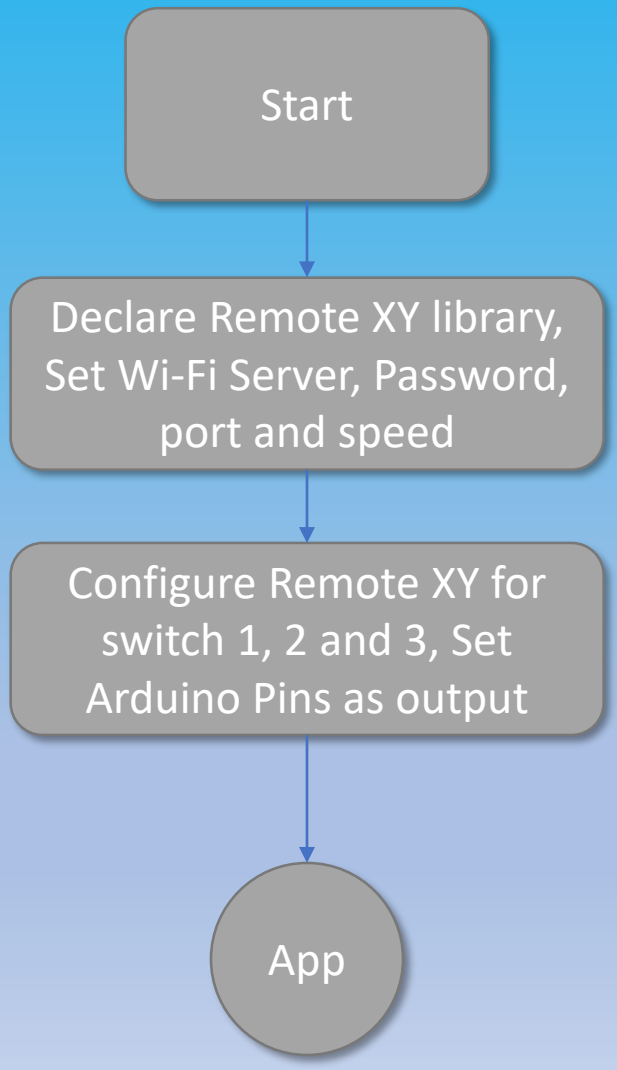
- LED1 (yellow) → 8
- LED2(green) → 9
- LED3(red) → 10
- Tx (ESP8266) → 0 (Rx: Arduino)
- Rx (ESP8266) → 1 (Tx: Arduino)
- VCC + EN → 3.3 V (Arduino)
- GND → GND (Arduino)

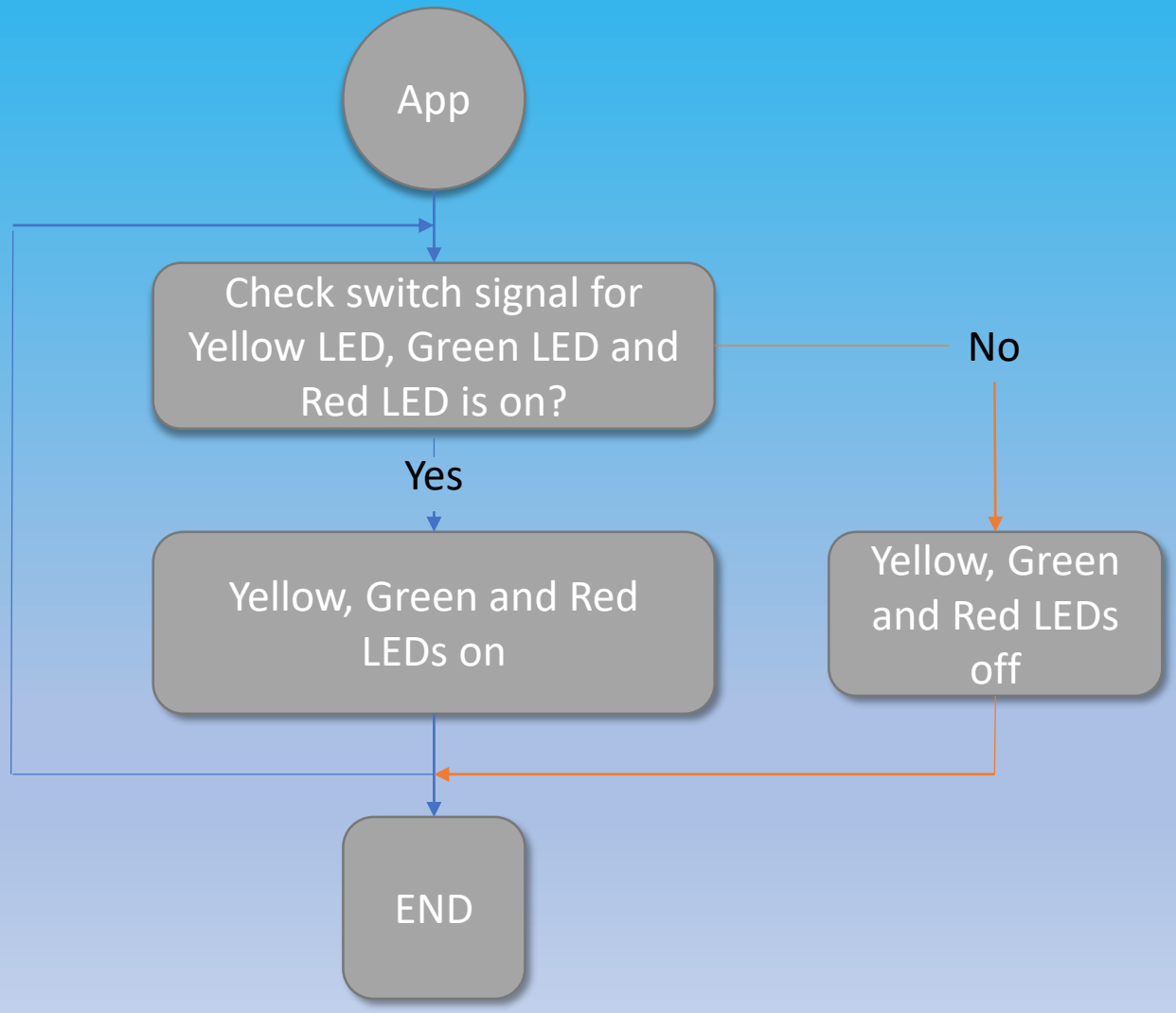
➢ Disconnect the Tx and Rx pins of Wi-Fi module while uploading the sketch to Arduino.

Circuit Diagram



Flow Chart


Start
Declare Remote XY library, Set Wi-Fi Server, Password, port and speed
Configure Remote XY for switch 1, 2 and 3, Set Arduino Pins as output
App


App
Check switch signal for Yellow LED, Green LED and Red LED is on?
No
Yes
Yellow, Green and Red LEDs on
Yellow, Green and Red LEDs off
END

Software Implementation

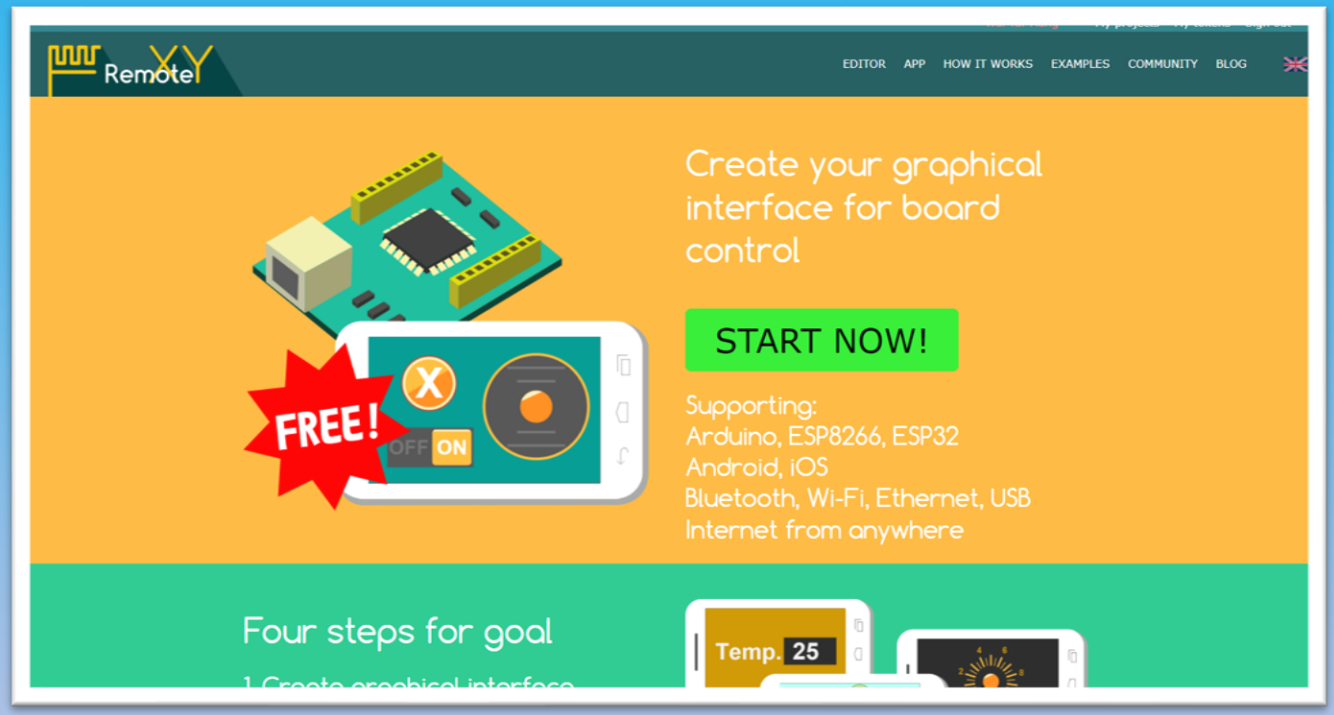
Wi-Fi Access Point

The software setting is shown step by step as follows.

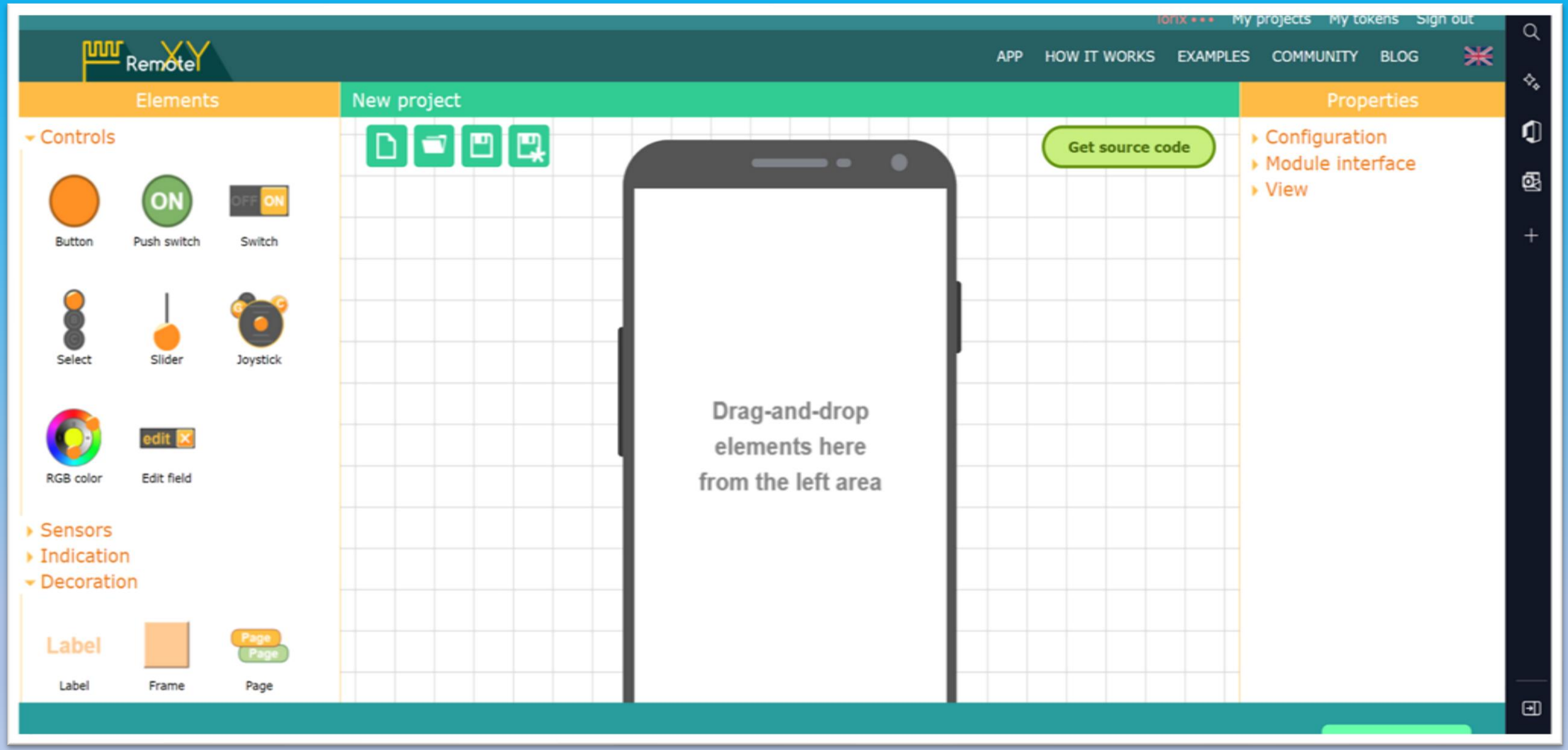
1.Go to the website "Remote XY".

https://remotexy.com

2.Then click " start now".



3. You will see home page of Remote XY. Then, go to configuration.

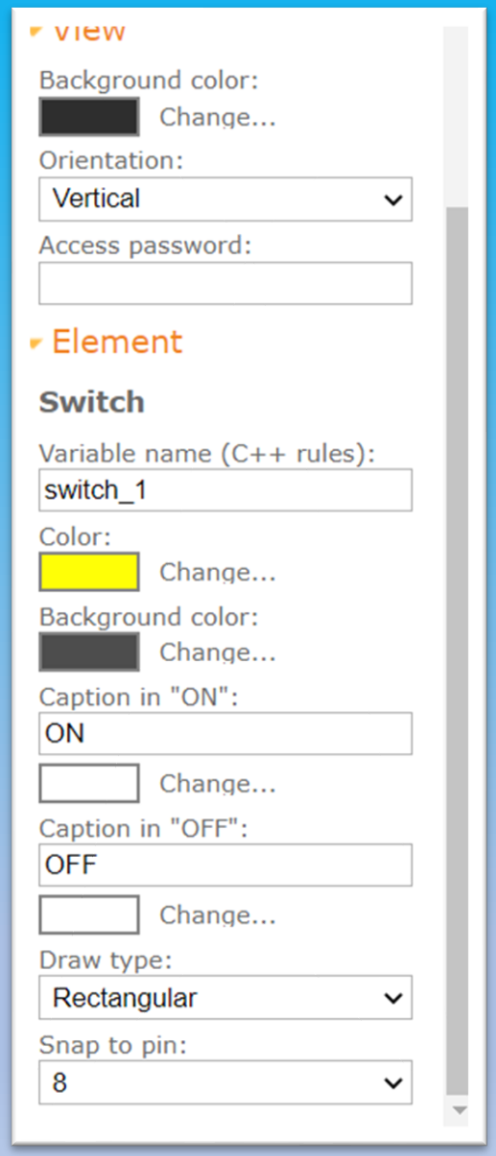


Select as follows. After that, click “Apply”.

4.

In module interface, set serial port to 0(RX) and 1(TX), baud rate to 115200.

Name your wi-fi and set password. Port can be set as your preference.

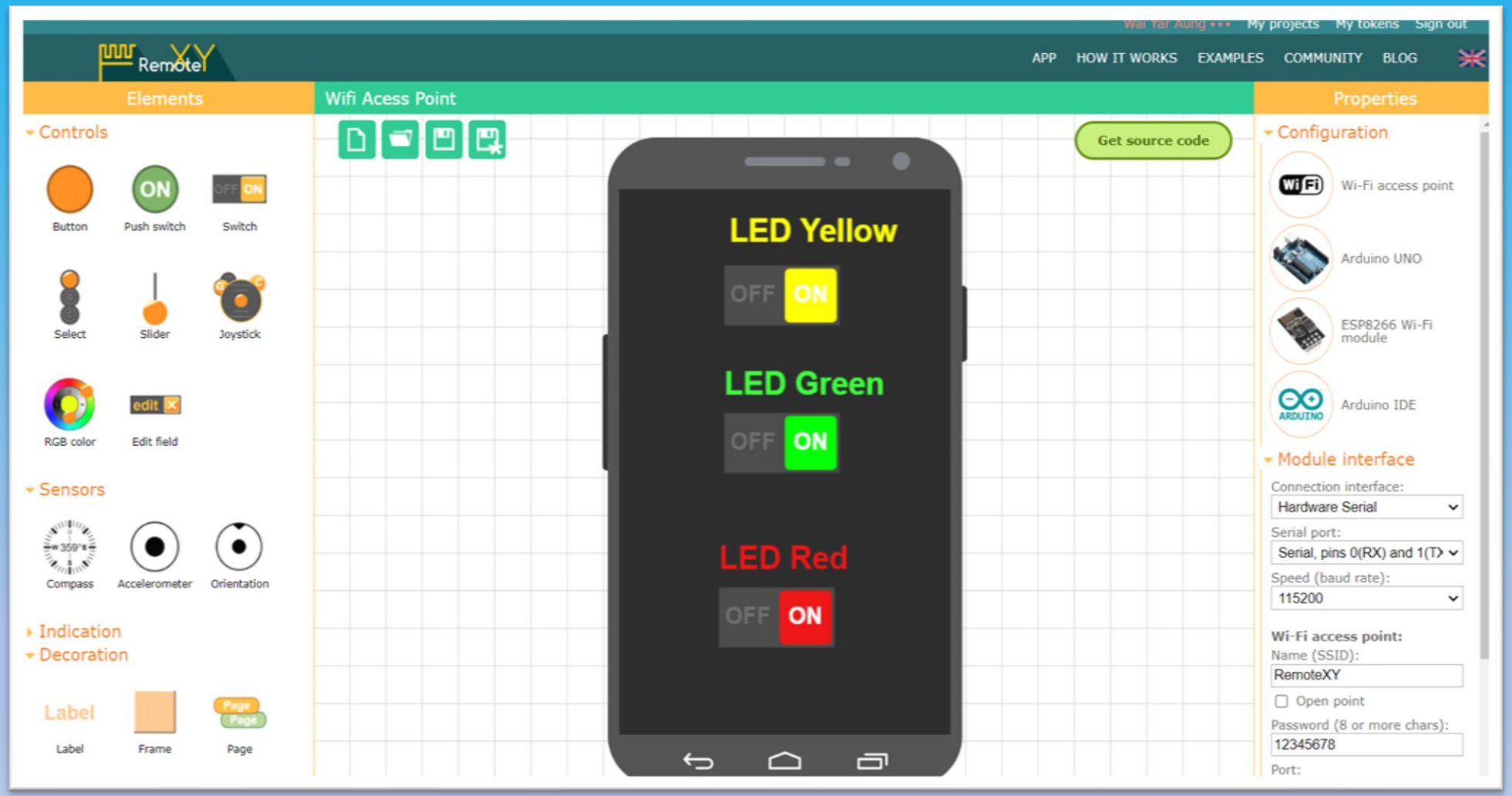


5. In View, you can decorate. If you set access password, you will need to enter password when connecting to wi-fi on Remote XY app.

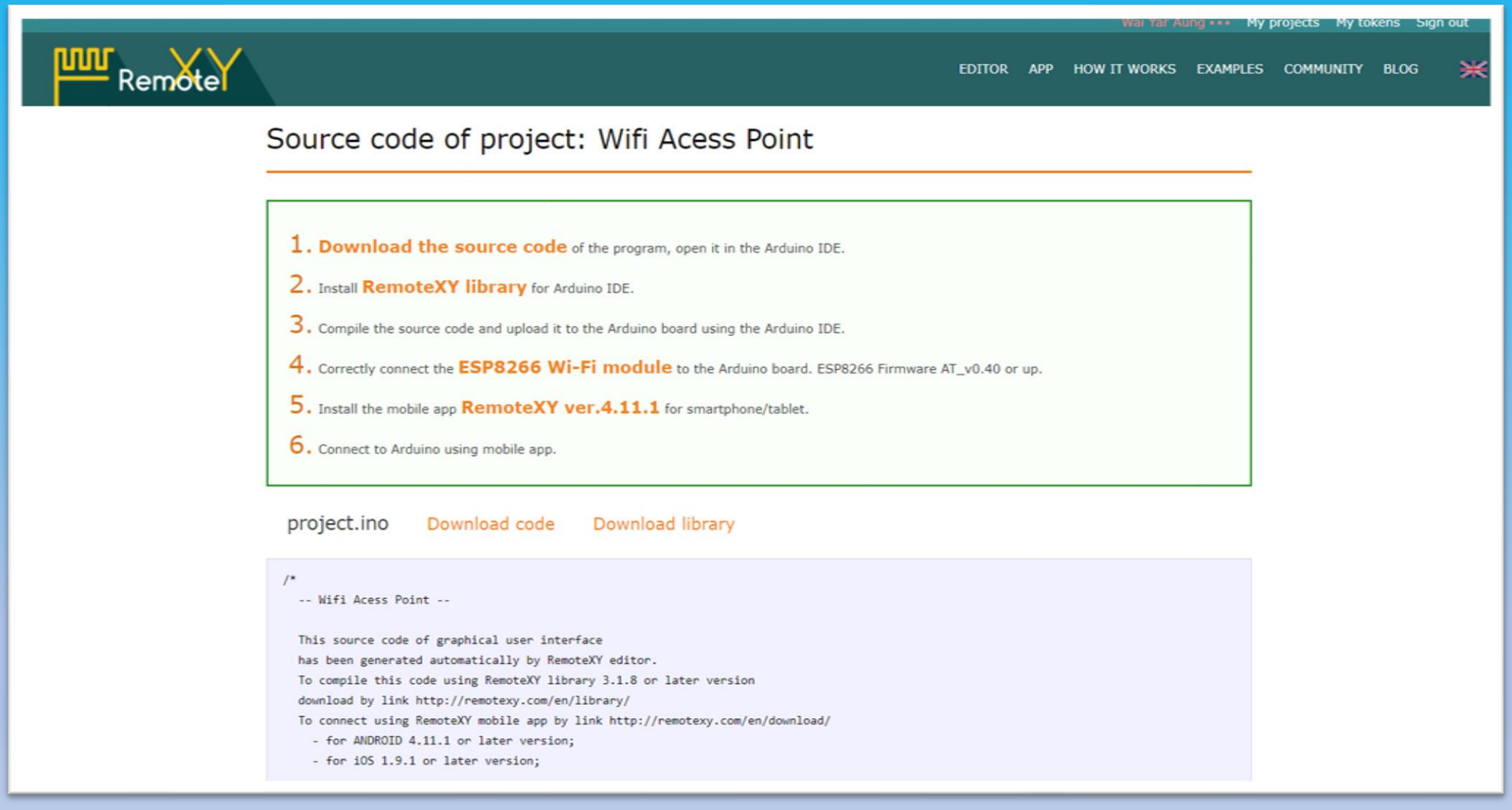
6. In Controls, take switch or push button. In this project, switch is used and name the elements, LED Yellow, LED Green and LED Red.

7.Click the switch and set pin in “Snap to pin”. In this project, set the pins according to the circuit diagram.
LED yellow is set to pin 8.
LED green is set to pin 9.
LED red is set to pin 10.

8. This is the final result. After that, don't forget to save the project. Then, click " Get source code".



- The Remote XY will generate arduino code automatically.
- You can download the code or just simply copy the code.
- Download library which needs to be installed before running the code.

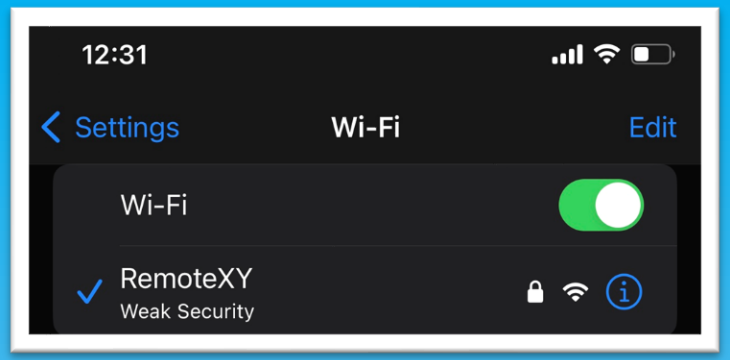


➢ Disconnect the Tx and Rx pins of Wi-Fi module while uploading the sketch to Arduino.

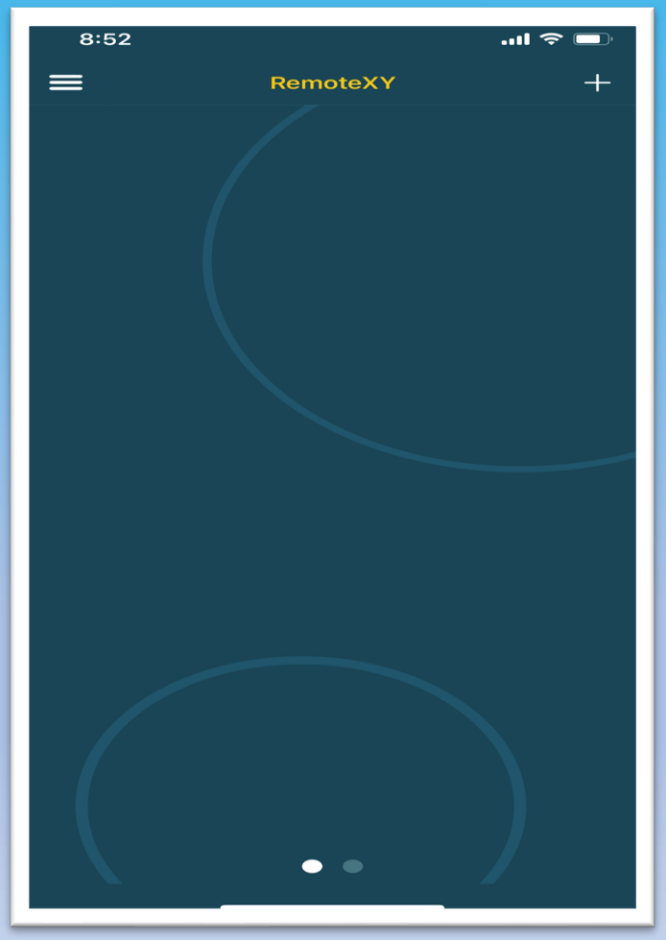
Download Remote XY: Arduino control app from Google play store or App store to your mobile phone.

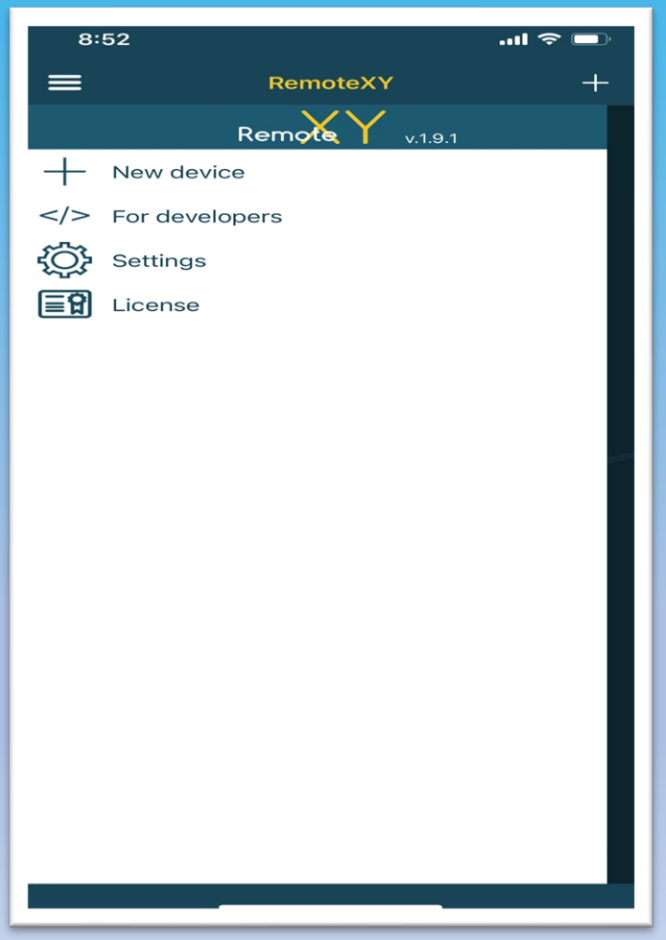
https://apps.apple.com/app/id1168130280

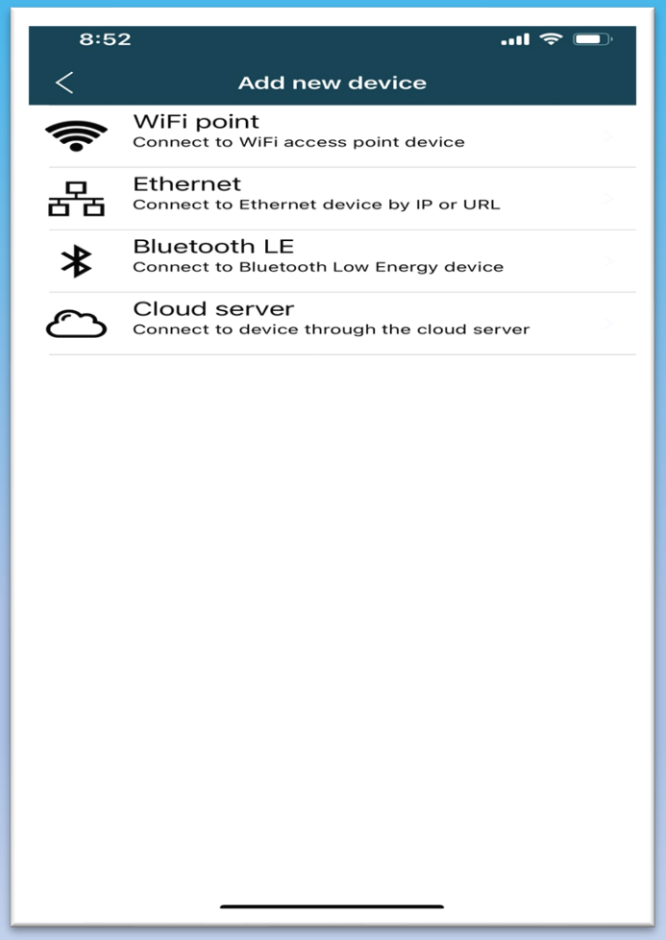
https://play.google.com/store/apps/details?id=com.shevauto.remotexy.free

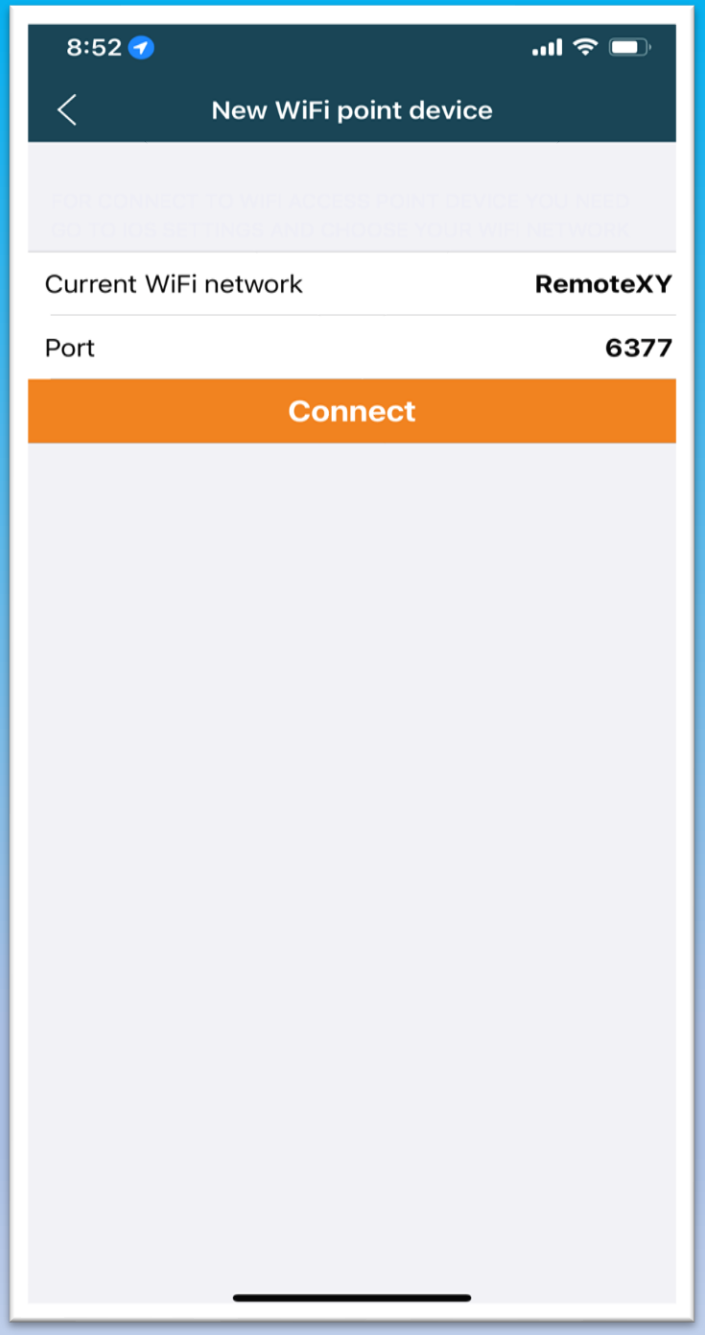


- Connect your phone to Remote XY in the Wi-Fi setting.
- Open your Remote XY app and add the new device.
- Add new device > Wi-Fi point > Remote XY > Enter password and connect.



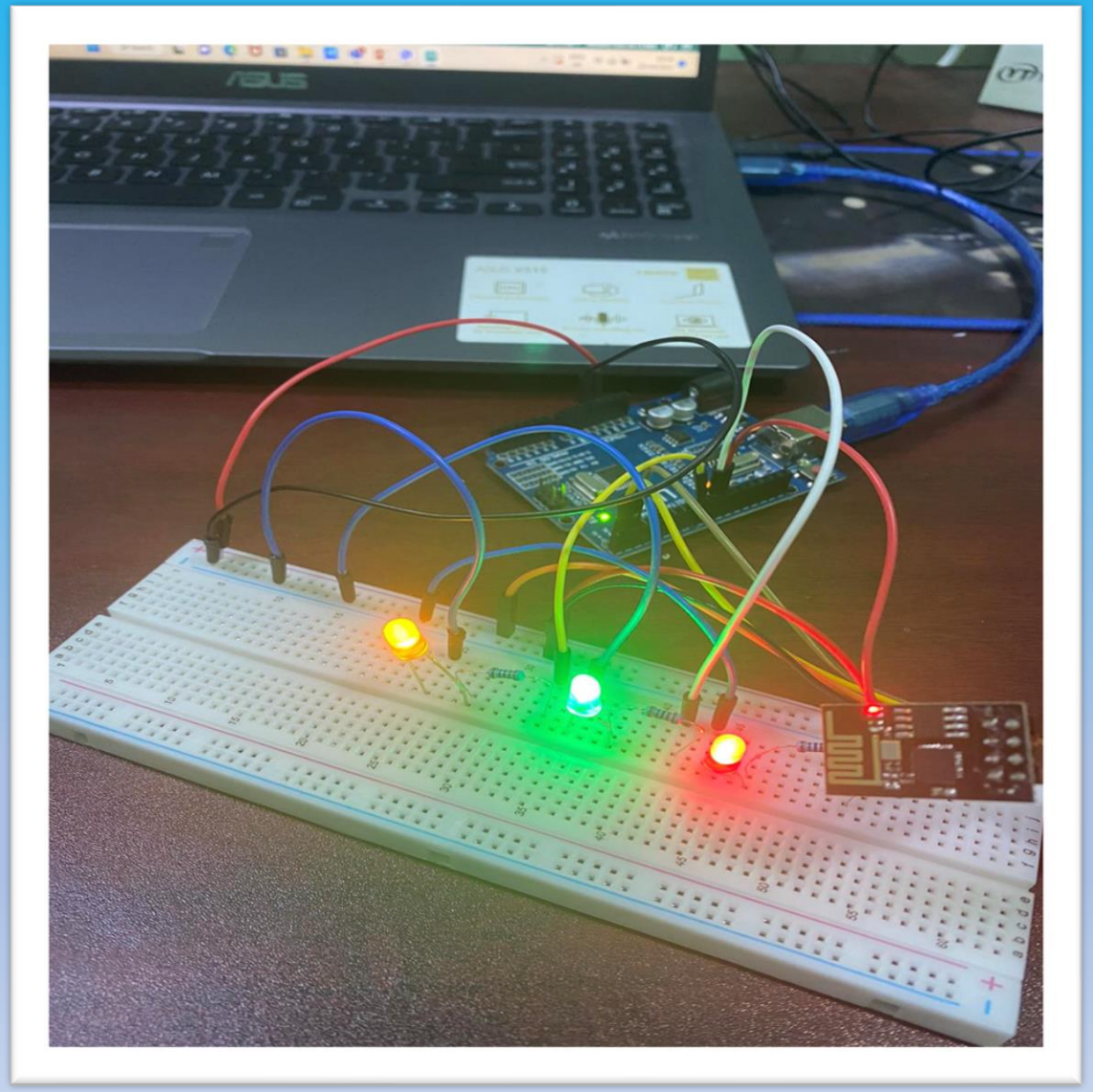





8:52
New WiFi point device
FOR CONNECT TO WIFI ACCESS POINT DEVICE YOU NEED
GO TO IOS SETTINGS AND CHOOSE YOUR WIFI NETWORK
Current WiFi network
RemoteXY
Port
6377
Connect


8:53
Close
RemoteXY
Stop
LED Yellow
OFF
ON
LED Green
OFF
ON
LED Red
OFF
ON

Tests and Results



Program Sketch

Wifi_access_point.ino

```
#define REMOTEXY_MODE__ESP8266_HARDSERIAL_POINT

#include <RemoteXY.h>

// RemoteXY connection settings
#define REMOTEXY_SERIAL Serial
#define REMOTEXY_SERIAL_SPEED 115200
#define REMOTEXY_WIFI_SSID "RemoteXY"
#define REMOTEXY_WIFI_PASSWORD "12345678"
#define REMOTEXY_SERVER_PORT 6377


// RemoteXY configurate
#pragma pack(push, 1)
uint8_t RemoteXY_CONF[] =   // 111 bytes
  { 255,3,0,0,0,104,0,16,8,1,2,0,20,14,22,11,94,26,31,31,
  79,78,0,79,70,70,0,2,0,20,41,22,11,135,26,31,31,79,78,0,
  79,70,70,0,2,0,19,73,22,11,1,26,31,31,79,78,0,79,70,70,
  0,129,0,19,65,24,6,1,76,69,68,32,82,101,100,0,129,0,20,33,
  30,6,136,76,69,68,32,71,114,101,101,110,0,129,0,21,5,18,6,94,
  76,69,68,32,89,101,108,108,111,119,0 };

// this structure defines all the variables and events of your control interface
struct {

    // input variables
  uint8_t switch_1; // =1 if switch ON and =0 if OFF
  uint8_t switch_2; // =1 if switch ON and =0 if OFF
  uint8_t switch_3; // =1 if switch ON and =0 if OFF

    // other variable
  uint8_t connect_flag;  // =1 if wire connected, else =0

} RemoteXY;
#pragma pack(pop)

//////////////////////////////////////////////
//           END RemoteXY include           //
//////////////////////////////////////////////

#define PIN_SWITCH_1 8
#define PIN_SWITCH_2 9
#define PIN_SWITCH_3 10


void setup()
{
  RemoteXY_Init ();

  pinMode (PIN_SWITCH_1, OUTPUT);
  pinMode (PIN_SWITCH_2, OUTPUT);
  pinMode (PIN_SWITCH_3, OUTPUT);

  // TODO you setup code

}

void loop()
{
  RemoteXY_Handler ();

  digitalWrite(PIN_SWITCH_1, (RemoteXY.switch_1==0)?LOW:HIGH);
  digitalWrite(PIN_SWITCH_2, (RemoteXY.switch_2==0)?LOW:HIGH);
  digitalWrite(PIN_SWITCH_3, (RemoteXY.switch_3==0)?LOW:HIGH);

  // TODO you loop code
```

```
39  //////////////////////////////////////////////
40
41  #define PIN_SWITCH_1 8
42  #define PIN_SWITCH_2 9
43  #define PIN_SWITCH_3 10
44
45
46  void setup()
47  {
48    RemoteXY_Init ();
49
50    pinMode (PIN_SWITCH_1, OUTPUT);
51    pinMode (PIN_SWITCH_2, OUTPUT);
52    pinMode (PIN_SWITCH_3, OUTPUT);
53
54    // TODO you setup code
55
56  }
57
58  void loop()
59  {
60    RemoteXY_Handler ();
61
62    digitalWrite(PIN_SWITCH_1, (RemoteXY.switch_1==0)?LOW:HIGH);
63    digitalWrite(PIN_SWITCH_2, (RemoteXY.switch_2==0)?LOW:HIGH);
64    digitalWrite(PIN_SWITCH_3, (RemoteXY.switch_3==0)?LOW:HIGH);
65
66    // TODO you loop code
67    // use the RemoteXY structure for data transfer
68    // do not call delay()
69
70
71  }
```

Cloud Server

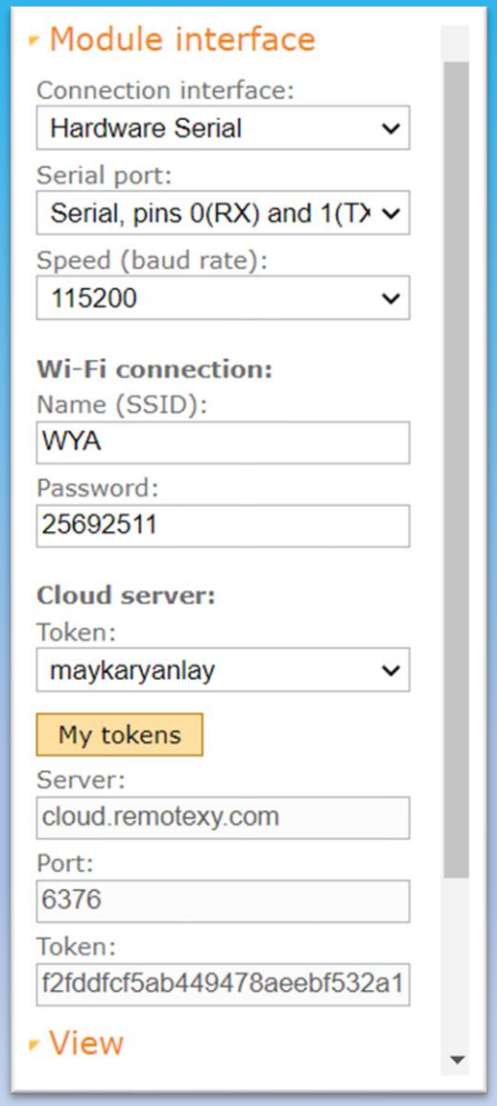
For cloud server,

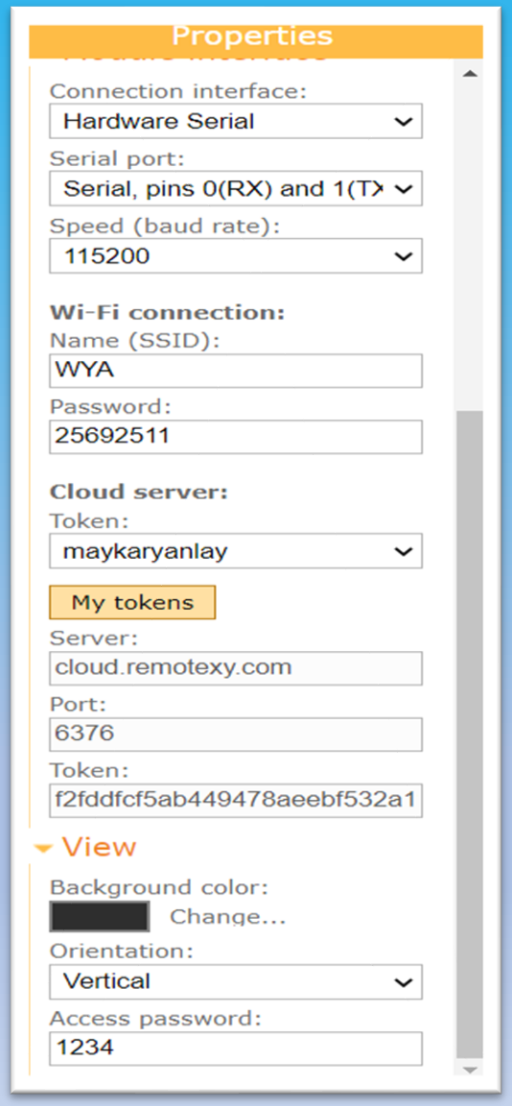
Step 1 and 2 are the same as above.

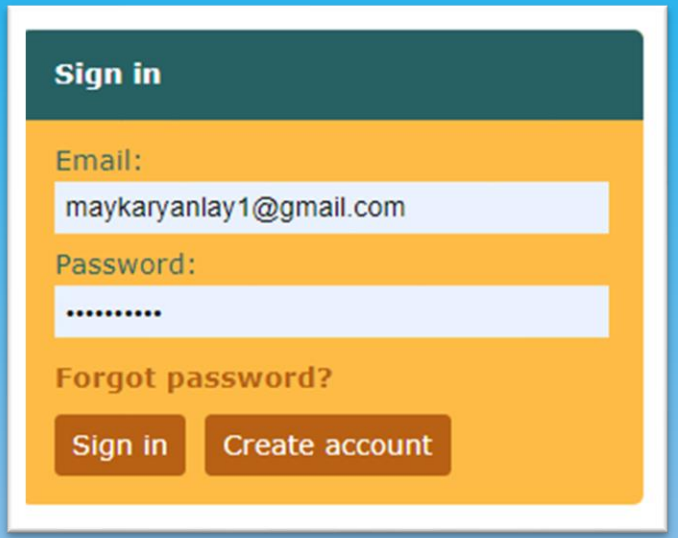
Then, choose cloud server of “connection” section in step 3.

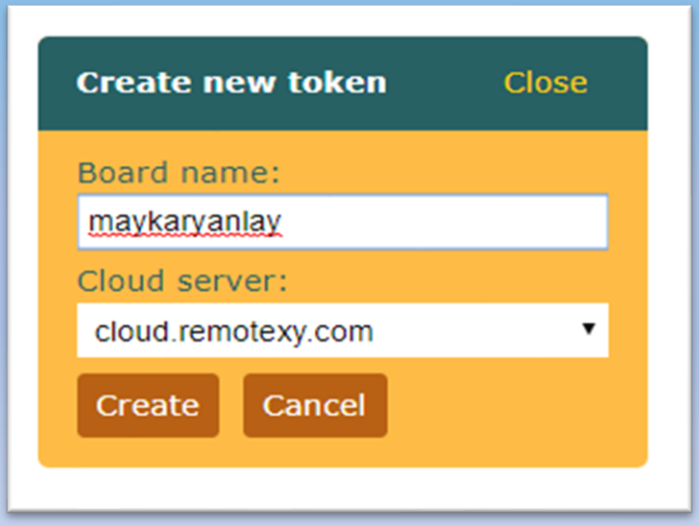
4.

In module interface, set serial port to 0(RX) and 1(TX), baud rate to 115200. Name your wi-fi and set password you would like to connect. Then, create a token to build a cloud server.

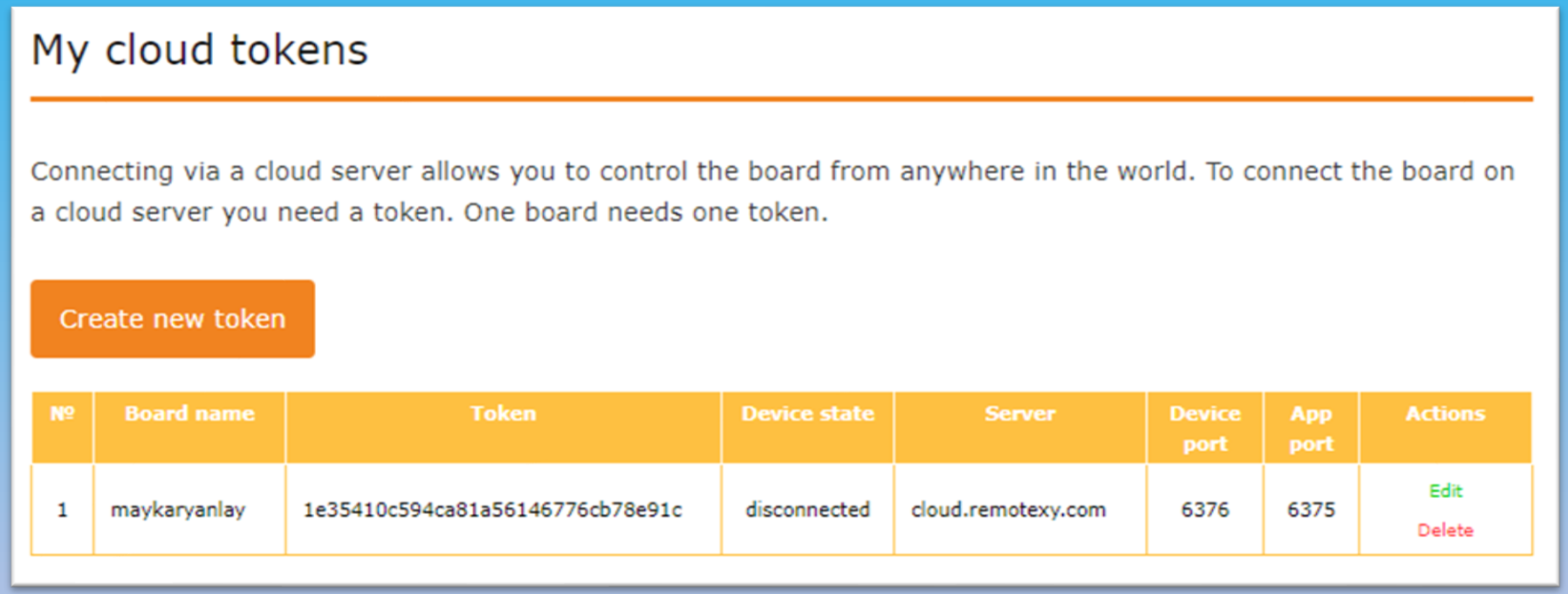








5. Copy the token.



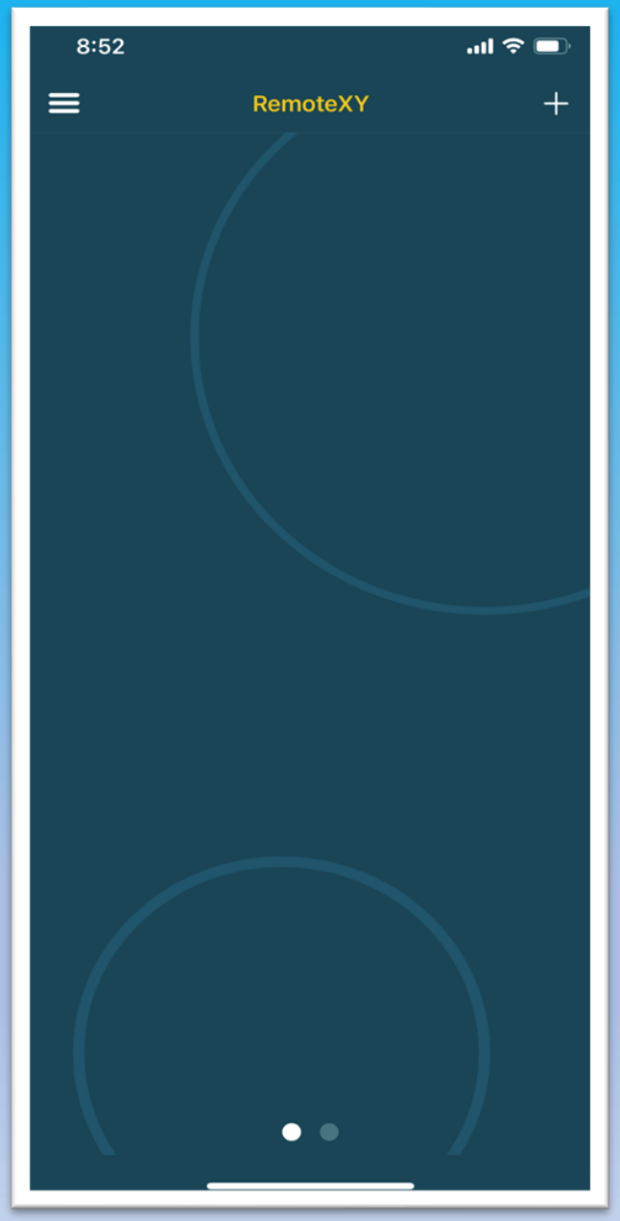
My cloud tokens

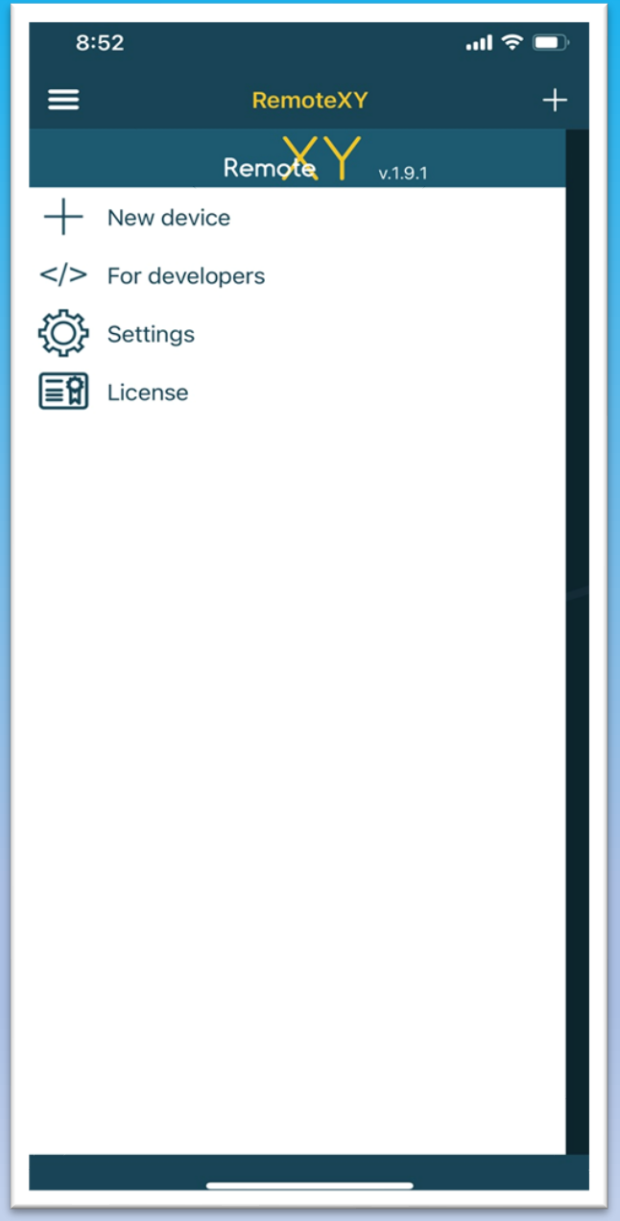
Connecting via a cloud server allows you to control the board from anywhere in the world. To connect the board on a cloud server you need a token. One board needs one token.

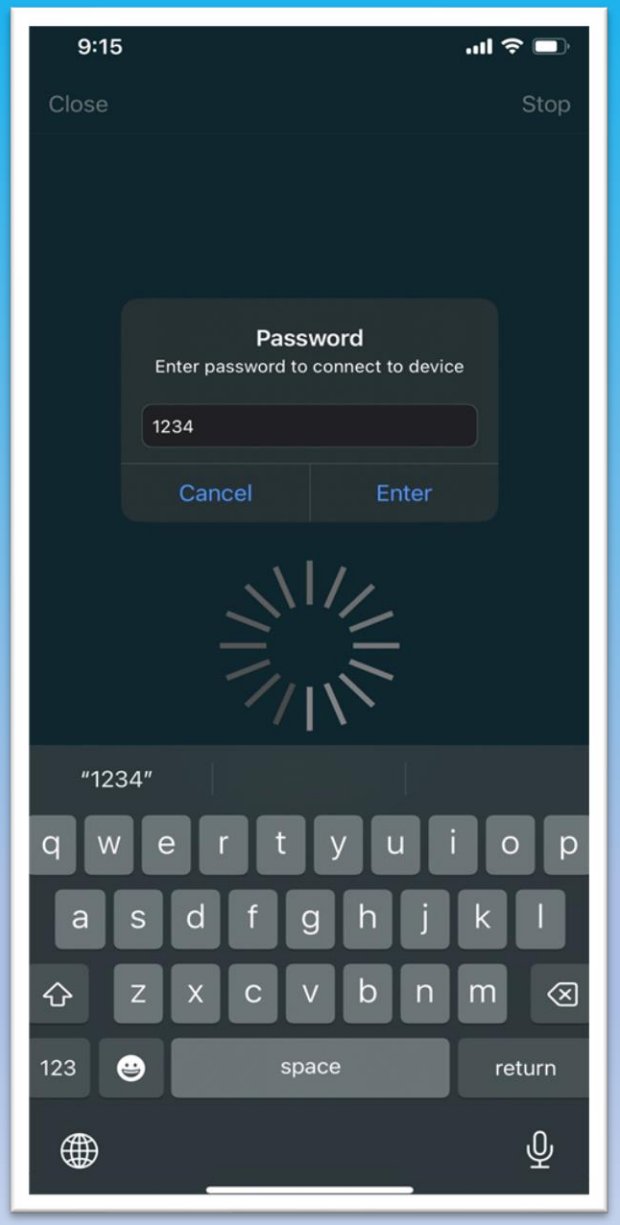
Create new token

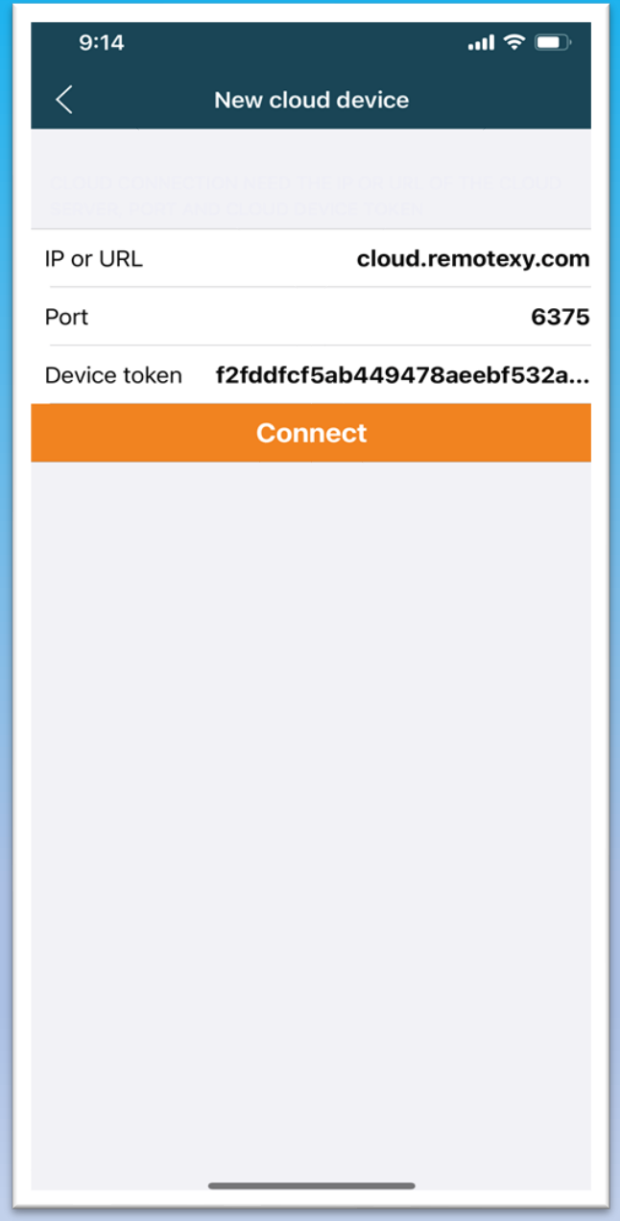
№	Board name	Token	Device state	Server	Device port	App port	Actions
1	maykaryanlay	1e35410c594ca81a56146776cb78e91c	disconnected	cloud.remotexy.com	6376	6375	Edit Delete

Do as follows and fill your cloud server in IP or URL, port number in port and your token in Device token.

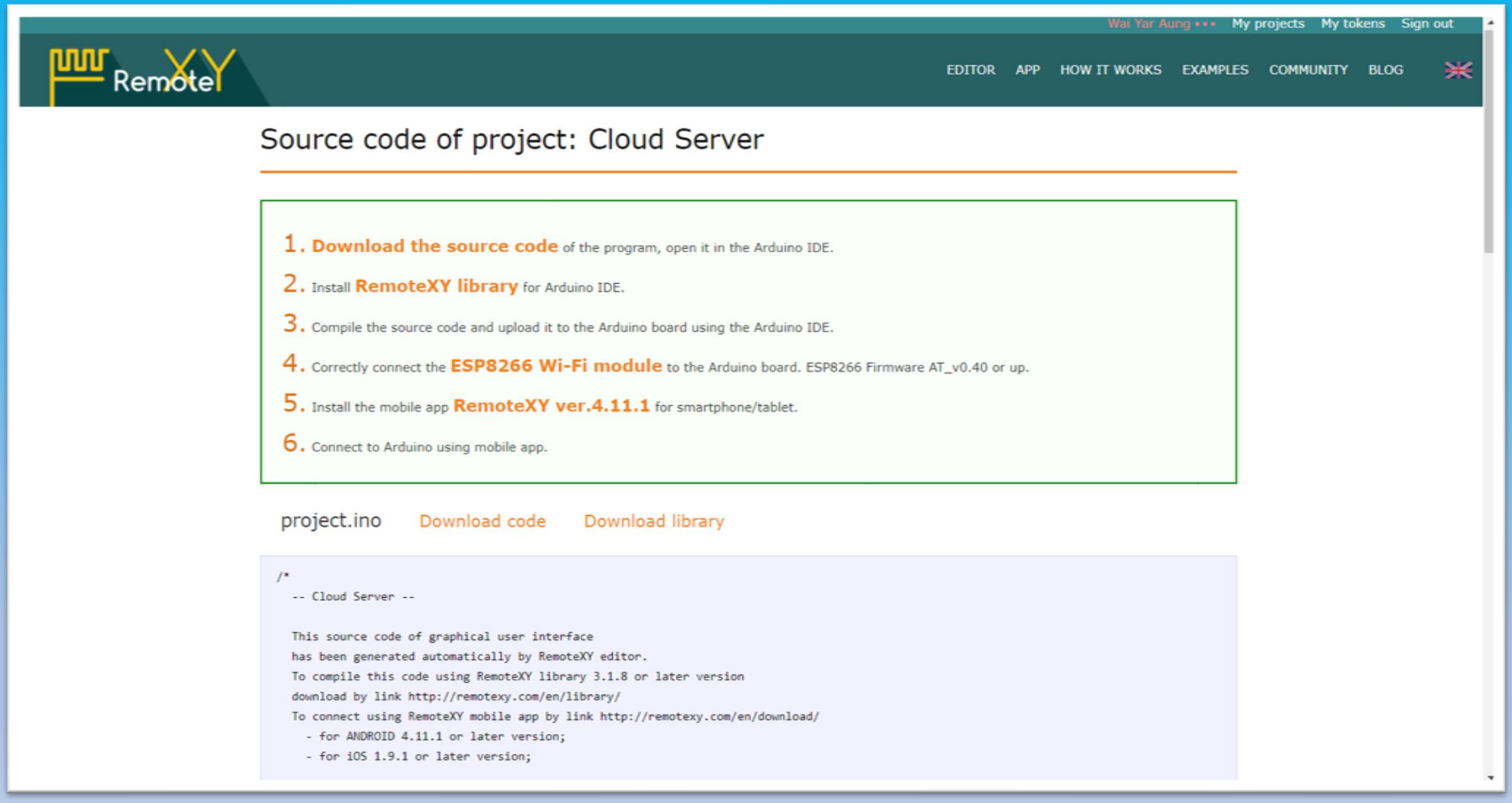








Then, get source codes and run it in your Arduino.



Program Sketch

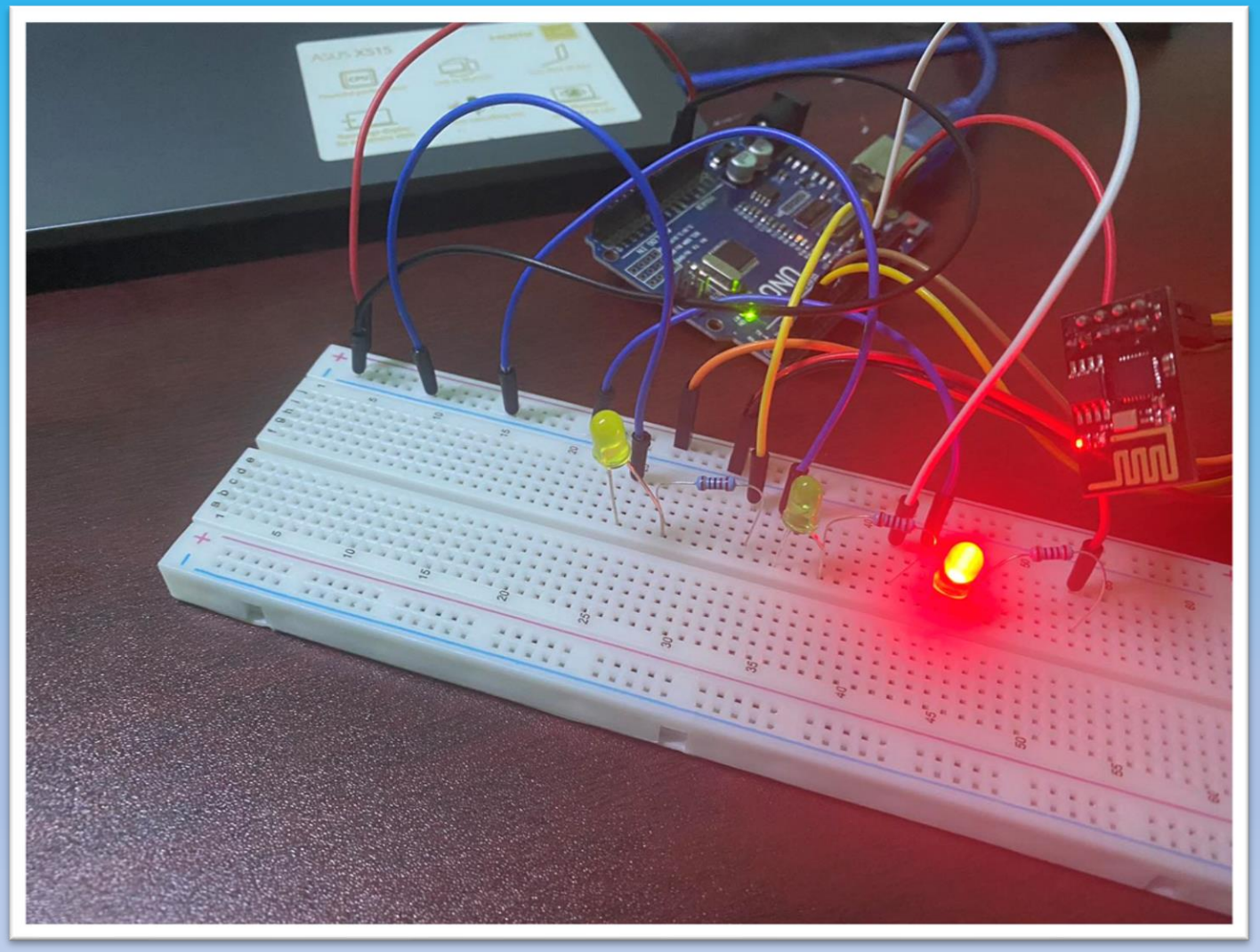
cloud_server.ino

```
#define REMOTEXY_MODE__ESP8266_HARDSERIAL_CLOUD

#include <RemoteXY.h>

// RemoteXY connection settings
#define REMOTEXY_SERIAL Serial
#define REMOTEXY_SERIAL_SPEED 115200
#define REMOTEXY_WIFI_SSID "WYA"
#define REMOTEXY_WIFI_PASSWORD "25692511"
#define REMOTEXY_CLOUD_SERVER "cloud.remotexy.com"
#define REMOTEXY_CLOUD_PORT 6376
#define REMOTEXY_CLOUD_TOKEN "f2fddfcf5ab449478aeebf532a11cc58"
#define REMOTEXY_ACCESS_PASSWORD "1234"


// RemoteXY configurate
#pragma pack(push, 1)
uint8_t RemoteXY_CONF[] =   // 111 bytes
  { 255,3,0,0,0,104,0,16,8,1,2,0,20,14,22,11,94,26,31,31,
  79,78,0,79,70,70,0,2,0,20,41,22,11,135,26,31,31,79,78,0,
  79,70,70,0,2,0,19,73,22,11,1,26,31,31,79,78,0,79,70,70,
  0,129,0,19,65,24,6,1,76,69,68,32,82,101,100,0,129,0,20,33,
  30,6,136,76,69,68,32,71,114,101,101,110,0,129,0,21,5,18,6,94,
  76,69,68,32,89,101,108,108,111,119,0 };

// this structure defines all the variables and events of your control interface
struct {

    // input variables
  uint8_t switch_1; // =1 if switch ON and =0 if OFF
  uint8_t switch_2; // =1 if switch ON and =0 if OFF
  uint8_t switch_3; // =1 if switch ON and =0 if OFF
```

```
    // other variable
  uint8_t connect_flag;  // =1 if wire connected, else =0

} RemoteXY;
#pragma pack(pop)

/////////////////////////////////////////////
//           END RemoteXY include          //
/////////////////////////////////////////////

#define PIN_SWITCH_1 8
#define PIN_SWITCH_2 9
#define PIN_SWITCH_3 10


void setup()
{
  RemoteXY_Init ();

  pinMode (PIN_SWITCH_1, OUTPUT);
  pinMode (PIN_SWITCH_2, OUTPUT);
  pinMode (PIN_SWITCH_3, OUTPUT);

  // TODO you setup code

}

void loop()
{
  RemoteXY_Handler ();

  digitalWrite(PIN_SWITCH_1, (RemoteXY.switch_1==0)?LOW:HIGH);
  digitalWrite(PIN_SWITCH_2, (RemoteXY.switch_2==0)?LOW:HIGH);
```

```
42  //////////////////////////////////////////////
43
44  #define PIN_SWITCH_1 8
45  #define PIN_SWITCH_2 9
46  #define PIN_SWITCH_3 10
47
48
49  void setup()
50  {
51    RemoteXY_Init ();
52
53    pinMode (PIN_SWITCH_1, OUTPUT);
54    pinMode (PIN_SWITCH_2, OUTPUT);
55    pinMode (PIN_SWITCH_3, OUTPUT);
56
57    // TODO you setup code
58
59  }
60
61  void loop()
62  {
63    RemoteXY_Handler ();
64
65    digitalWrite(PIN_SWITCH_1, (RemoteXY.switch_1==0)?LOW:HIGH);
66    digitalWrite(PIN_SWITCH_2, (RemoteXY.switch_2==0)?LOW:HIGH);
67    digitalWrite(PIN_SWITCH_3, (RemoteXY.switch_3==0)?LOW:HIGH);
68
69    // TODO you loop code
70    // use the RemoteXY structure for data transfer
71    // do not call delay()
72
73
74  }
```

Tests and Results



Discussion

- When connecting the ESP8266 wi-fi module for too long, it gets hot.
- Wi-fi module can disconnect sometime, uploading the code one more time can solve this problem.
- In this project, Remote XY is used but there are many other websites and platforms to approach ESP8266 wi-fi module.
- There are numerous breakout boards available based on ESP8266 Wi-fi Module like ESP8266 Node MCU V3. Because of its compact size, its most importantly used in autonomous project.
- 2.4GHz frequency can reach up to 150 feet (46 m) indoors and 300 feet (92 m) outdoors. The 2.4GHz band uses longer waves, which makes it better suited for longer ranges or transmission through walls and other solid objects.

References

https://remotexy.com/

https://www.electronicwings.com/sensors-modules/esp8266-wifi-module

https://erc-bpgc.github.io/handbook/electronics/Modules/wifi_module/

Thank you.