Standalone NodeMCU Relay Board

INTRODUCTION

A brief guide how to wirelessly control a relay board with a smartphone. A relay is used to switch the power on and off according to the user needs. The system consist of NodeMCU, single relay board and the Blynk application installed on a smartphone. In addition, the application can send tweets to Twitter. Blynk application can be downloaded for Android and iOS smartphones. Arduino IDE is used for programming the NodeMCU.

COMPONENTS

Blynk application

Blynk is a platform with iOS and Android apps to control Arduino, Raspberry Pi and the likes over the Internet. Blynk is used as the user interface. Blynk application can be downloaded from the link below.

http://www.blynk.cc/

NodeMCU

NodeMCU is an open source IoT platform based on ESP8266 Wi-Fi chip. NodeMCU is used as standalone controller. It connects to the Wi-Fi network and communicates with Blynk server.

Single relay board
A single relay is used to control the high voltage line current. Maximum current for one relay is listed 10 A at 230V.

Dupont wires

Dupont wires are used to connect the NodeMCU and the relay board.

CONSTRUCTION

Setting up Blynk application

After downloading the Blynk application and creating a new account, new project needs to be set up. Project name and hardware model fields can be left unmodified. These fields are set later. Auth token is needed, by clicking the E-mail button, the auth token is sent to the email address provided when account was created. By clicking the Create button, the project is created and it can be modified later on.
After the project is created the actual project set up can begin. By clicking the nut icon the app opens the Project Settings tab. For NodeMCU the hardware model is set to ESP8266.

As the relay is wished to be either on or off, button widget is added. For the tweeting option also Twitter widget is added. Terminal widget is added for debugging the code. Widgets are added by clicking the plus symbol. Button's output pin is set as a virtual and the port is set to V0, the button mode is set to Switch. For the tweeting option a Twitter account is needed. Twitter account is connected by clicking the Connect Twitter button. After authorizing Blynk to use the account the settings are set for tweeting. Terminal is set to virtual pin V1. Input line is enabled.
Setting up Arduino IDE

As the Blynk app has been set next step is to install the Arduino IDE and the libraries needed.

Arduino IDE can be downloaded from the link below

https://www.arduino.cc/en/Main/Software

After installing the Arduino IDE, the support for ESP8266 needs to be added. By opening the File menu and selecting the Preferences opens the preferences window. After adding the URL below to the Additional Boards Manager URL’s and clicking Ok button the support for ESP8266 is added.

http://arduino.esp8266.com/package_esp8266com_index.json
The support needs to be installed. By opening the Tools menu and selecting Board and Boards Manager the Boards Manager windows opens. By selecting Contributed from Type drop-down menu the different options are listed. By selecting the esp8266 by ESP8266 community forum and clicking Install button the ESP8266 board is installed.

For the board to work properly some libraries needs to be installed. Libraries for ESP8266 and Blynk can be downloaded from the links below.

https://github.com/esp8266/Arduino
https://github.com/blynkkk/blynk-library

Instructions for adding libraries can be found from Arduino's web page linked below.

https://www.arduino.cc/en/guide/libraries#toc2

Connecting NodeMCU and relay board
Programming NodeMCU on Arduino IDE
void setup()
{
    Serial.begin(9600);
    digitalWrite(4, HIGH); // "Hello world!"
    digitalWrite(4, LOW);  // "Hello world!"
    while (Serial.available() == 0)  // Wait until connected
    {
        digitalWrite(4, HIGH);
    }
    Serial.println("BLEZON v1.0 BINARY_VSLEW: Device started");
    Serial.println("--END--");
    digitalWrite(4, HIGH);  // Send a reply, or type
    Serial.println("anything else and get it printed back.");
    Serial.flush();
}

void loop()
{
    if (Serial.available() > 0)  // Read the serial input
    {
        Serial.readString();
        void loop()
    }
    else
    {
        digitalWrite(4, HIGH);  // Send a reply, or type
        Serial.println("COFFEE");
    }
}

void loop()
{
    if (Serial.available() > 0)  // Read the serial input
    {
        Serial.readString();
        void loop()
    }
    else
    {
        digitalWrite(4, HIGH);  // Send a reply, or type
        Serial.println("COFFEE");
    }
}