EEE 310 | Project Report

Implementation of an MQTT Based Home Automation System

L-3, T-1

Section: A2

Submission Date: 27th July, 2021

Submitted to,

Dr. Md. Forkan Uddin

Professor, Dept of EEE, BUET

Tashfiq Ahmed

Lecturer, Dept of EEE, BUET

Submitted by,

Name: Fariza Siddiqua
ID: 1706043

Name: Shoaib Ahmed
ID: 1706046

Name: A F M Mahfuzul Kabir
ID: 1706045

Name: S. M. Monzurul Hoque Chowdhury
ID: 1706056

Table of Contents:

(1) Introduction:

Nowadays, we have remote controls for television sets and other electronic systems, which have made our lives really easy. Have you ever wondered about home automation which would give the facility of controlling tube lights, fans and other electrical appliances at home using a remote control or even our smartphone? The answer may be affirmative, but the main problem is that the available options are mostly cost-ineffective. Our project is a direct solution to this problem. In our project, we have implemented a wireless home automation system both in software platform using proteus, Blynk & some necessary applications and in practical using a smartphone-based system where we used Blynk app to control the appliances.

Automation is the most frequently spelled term in the field of electronics. The hunger for automation brought many revolutions in the existing technologies. These had greater importance than any other technologies due to its user-friendly nature. If we talk about automation, home automation must be the hottest cake of this arena. The "Home Automation" concept has existed for many years. The terms "Smart Home", "Intelligent Home" followed and has been used to introduce the concept of networking appliances and devices in the house.

In our software based wireless home automation project, we have built a home-like environment in proteus software. We used some lights & fans and there was "Arduino Uno" component in the home-like circuit. Moreover, we used Blynk app to control the environment with our smartphone or computer emulator. WIFI has been used as the connecting channel between Proteus & Blynk app. The key components used in this software-based application are:

1. Proteus
2. Arduino IDE
3. Arduino Library for Proteus
4. Arduino UNO
5. Fan
6. Relay Board
7. Light Bulb
8. Switch
9. LD Player (Android Emulator)
10. VSPE (Virtual Serial Ports Emulator)
11. Blynk App
12. 12V DC Line

In our practical implementation as well, we have used the Blynk app to control the proceedings from our smartphone. In this case, the key components are:

1. NodeMCU ESP8266
2. 4 or 2 channel relay modules
3. Blynk App
4. Connecting Wires
5. Light Bulb
6. Connecting Wires
7. 5V DC Supply

(2) Theory:

The IOT (Internet of Things) stands for the communication systems/protocol between everyday object over the internet. For example, when we can control our home appliances using our smart devices, such as smart phone, PC or tablet computer, its IOT. Our home automation system is nothing but digitalizing our home appliances by controlling them through smart devices. The most popular protocol used for home automation systems currently in the world is the MQTT (Message Queuing Telemetry Transport) protocol. MQTT architecture is very simple. It combines of a broker and some clients. Clients can work both as subscriber or publisher. Each client can control some appliances depending on their hardware strength. An MQTT network architecture looks something like this:

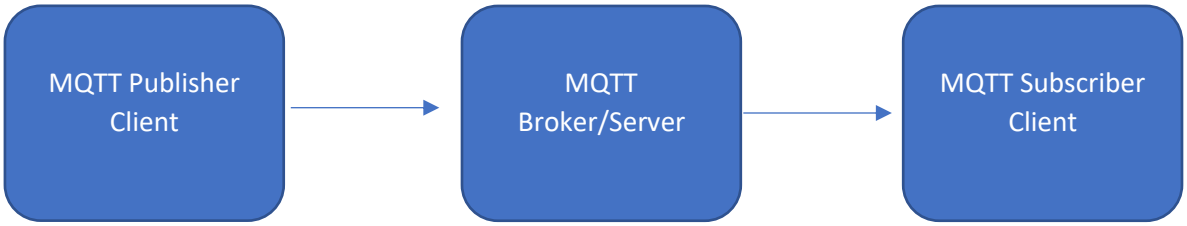


Figure: Simple MQTT protocol

The server/broker is the main controlling unit of an IOT system. It controls the whole network by running published or subscribed data like a mini computer. The publisher client on the other hand, can send (publish) message to the broker, that way the broker knows at what stage is the device connected to a client. The broker can be controlled wired or wirelessly. When we want to turn a device on or off, we send our signal to the broker. The broker then sends the message to a subscriber. The subscriber client then turns on or off the device and then publishes a feedback to the broker. That way, a single client can work as both publisher and subscriber.

The most common devices used as the MQTT clients are Arduino, ESP and etc. microcontroller circuits. Their job is simple, to turn on or off the devices connected to them. The broker however needs to be a bit more powerful than a simple Arduino or ESP. Usually Raspberry Pie is used to build a broker. However, there are many cloud broker available now-a-days that can control the client devices over the cloud. Blynk is one of them and for our project we'll be using Blynk cloud server to control our clients.

As for controlling the home appliances such as light bulbs, fans etc., a special kind of switch was used. This is called relay. Relay can be represented simply as the connection between electronic devices and electrical appliances. As most home appliances require a higher level of power (usually 12V or more than that), an Arduino or ESP or any other electronic device simply can't handle that much of voltage. So, we simply used relay, which is a switch that can be controlled using microcontrollers. And once the switch is closed, it can contain higher level of power that's required to run the home appliances.

As for our project, we both demonstrated it in simulation platform, as well as in hardware platform. For simulation platform we used Arduino as client devices while for hardware platform we used ESP as client devices. For both platform Blynk cloud server was used as a broker.

(3) Simulation:

In this part of our project we used various simulation software to build and apply our home automation system. The whole simulation was done in a simulation software, which is named Proteus.

Proteus: Proteus is a simulation software widely used for building and simulating electrical circuits. Once a circuit is built, the simulation can be turned on simply by turning on the play/simulate button. A special library had to included in the proteus platform which is the Arduino library. Arduino is not built in proteus; thus, the inclusion of Arduino library is a must.

VSPE: Virtual Serial Ports Emulator or VSPE is a software used to build a virtual port in the computer environment. As in real life, the Arduino was to be connected using a virtual port, the simulation environment doesn't have that. So, VSPE does that job and creates two virtual port (a pair of ports) which is used for communication.

Blynk: As mentioned earlier, Blynk is an online cloud server, widely used for IOT projects. Blynk uses MQTT protocol to control devices via the internet. Simply installing the Blynk app in an android or apple phone one can turn the device into an IOT control device. The Blynk app can be used to control the IOT connected devices if the device is connected to the cloud.

LD Player: LD player is an android emulator. For our project demonstration over the internet, simulating the Blynk app from our computer device was necessary. LD player simply contained the Blynk app which can control our simulation circuit without a problem.

Arduino IDE: Arduino IDE is simply the software that's used to configure the codes for the microcontroller we used. The Arduino used in our simulation contained the program and helped us to control our devices.

Circuit: The following circuit was built in the proteus simulation software.

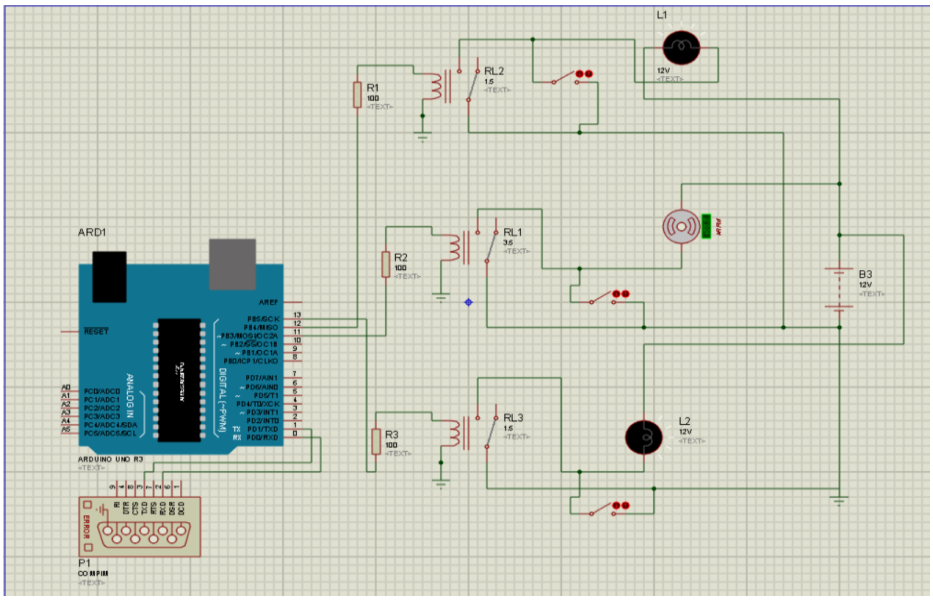


Figure: The circuit used for simulation.

As can be seen, we're using two light bulbs and one single fan as our home appliances to be controlled. The VSPE software was used to build a pair of ports (COM1 and COM2) for communication between Arduino and blynk.

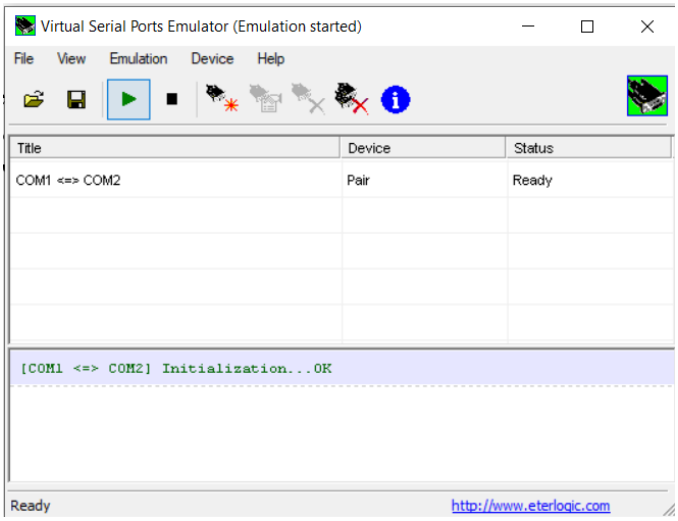


Figure: COM1 and COM2 ports emulated using VSPE

The Blynk app was used to create three switches that will control the three appliances. The Blynk app itself was emulated in LD Player emulator. The D11, D12 and D13 actually represents the Arduino pin the appliances are connected to through relay switch.

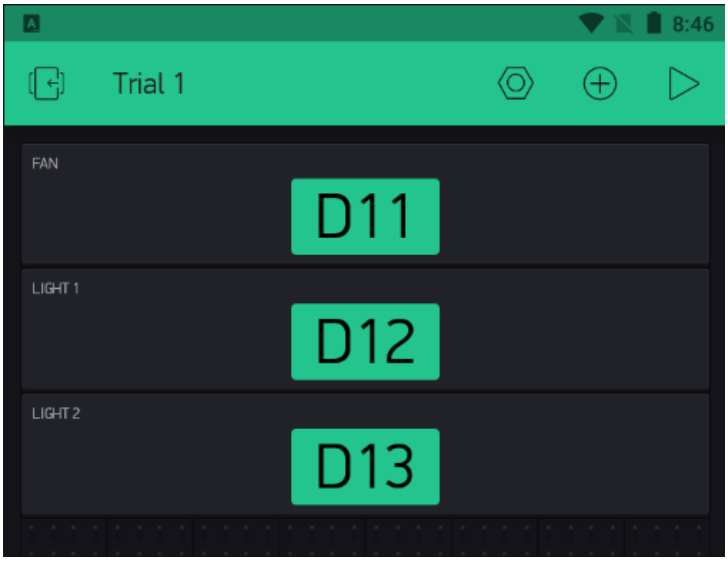


Figure: Blynk app in LD Player platform.

The Arduino in the proteus software had to be loaded with the code we mentioned earlier. The code is available in the appendix. The code had the authorization code that we got from the Blynk app. This authorization code is unique for every single project and anyone who has the code can control the system, so it's very much sensitive and must be held with caution.

Demonstration: Once the Proteus play button is pressed, the whole circuit comes to life. However, at this point all three appliances are turned off as the Arduino code was set such that initially the output pins are set at Low. Virtual ports were emulated using VSPE. A simple command window in the Arduino library is used to activate the VSPE ports and connect them with the Blynk app over the internet.

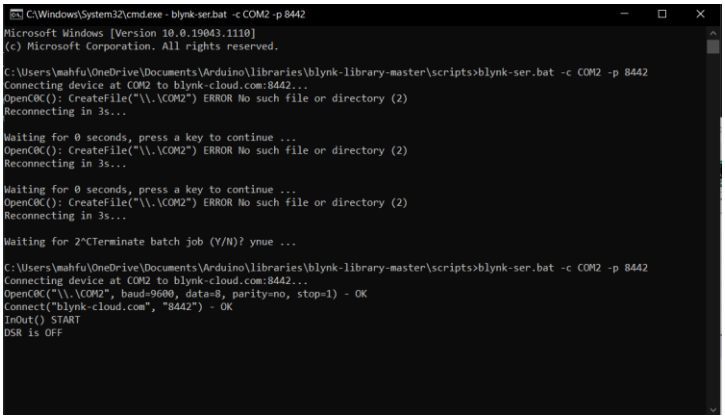


Figure: Command window to activate communication.

As can be seen in the figure below, the Blynk app is connected to the circuit. You can see in the upper right corner of the Blynk app, no warning or error is seen. So, the app is well connected and fully functional. As all three switches are turned off, the relay switch is at an open position and the appliances are turned off as well.

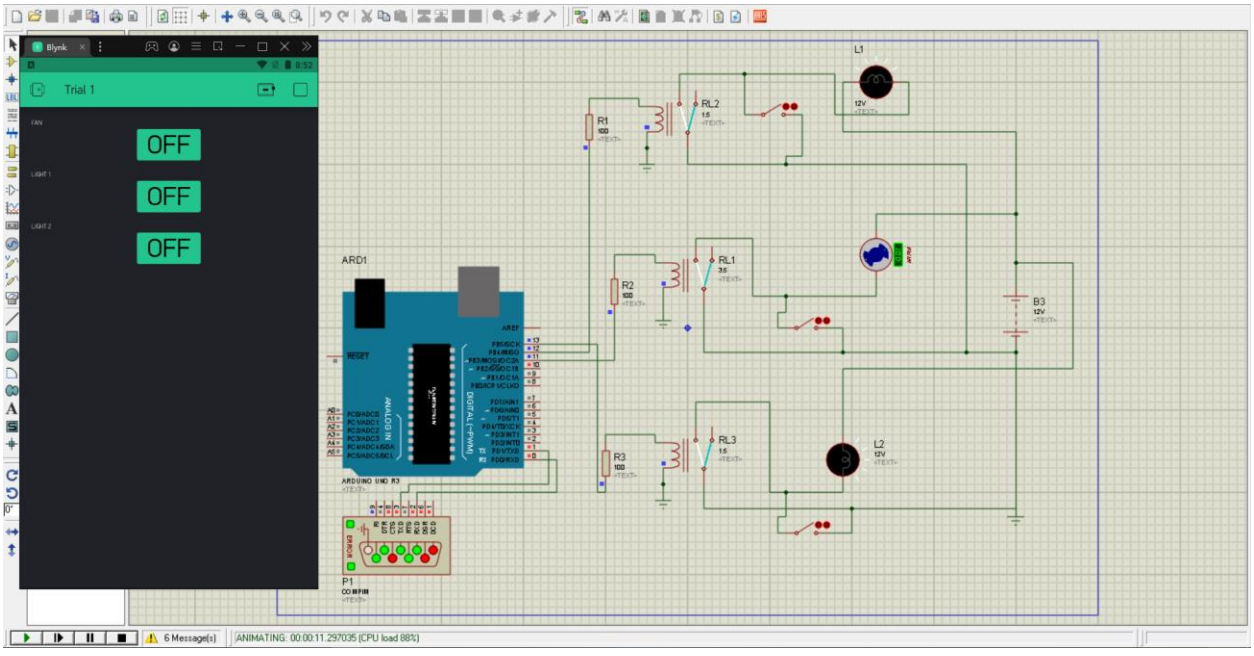


Figure: The Blynk app connected to the proteus circuit for home automation system.

Once we turn on the switches in the Blynk app, the Arduino pins are set at High and the relay switches close. The lights and fans turn on at the same time.

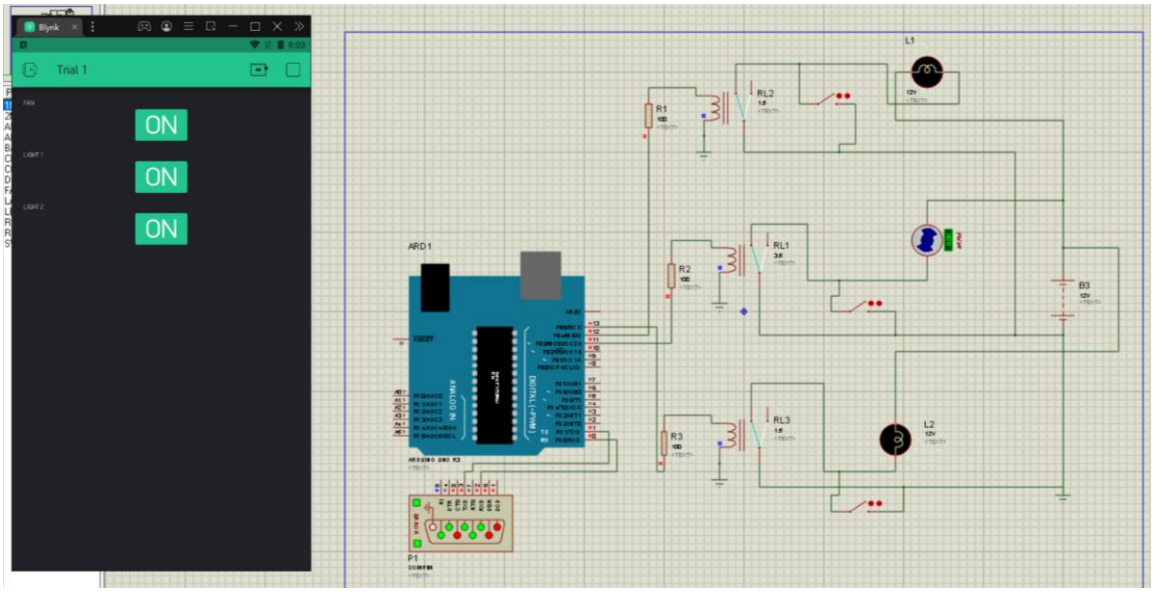


Figure: The appliances are functioning as soon as the switches in Blynk app is turned on.

There is also a physical switch connected in parallel with each appliance for emergency. The reason is simple, if the network is somehow down or the Arduino or any component is malfunctioning, the appliances can still be controlled using the physical switch.

(4) Implementation in Hardware:

We have also implemented our home automation system in hardware. We connected the device directly to the switch board of a particular room with one light and one fan. As for implementing in our home, we used ESP8266 microcontroller as client. The reason is that in practical, Arduino is not much powerful and ESP results in better performance.

Procedure:

1. Upload the code to NodeMCU board:

I. Open Arduino IDE→ Go to files → Preference → Select ESP8266 → Click OK
II. Go to tools → Library Manager → Search for blynk → Download the blynk library
III. Go to tools → Board Manager → Search for ESP8266 → Install the board
IV. Go to tools → Board "ESP8266" → Select NodeMCU 1.0

Then we wrote the code attached to the report and run it. It's also a built-in code in NodeMCU. In the code, the token number got from BLYNK app and wifi name and password should be given. Finally, we selected a particular Bluetooth port from tools and uploaded the code to the NodeMCU board.

2. Connect the element in following order.

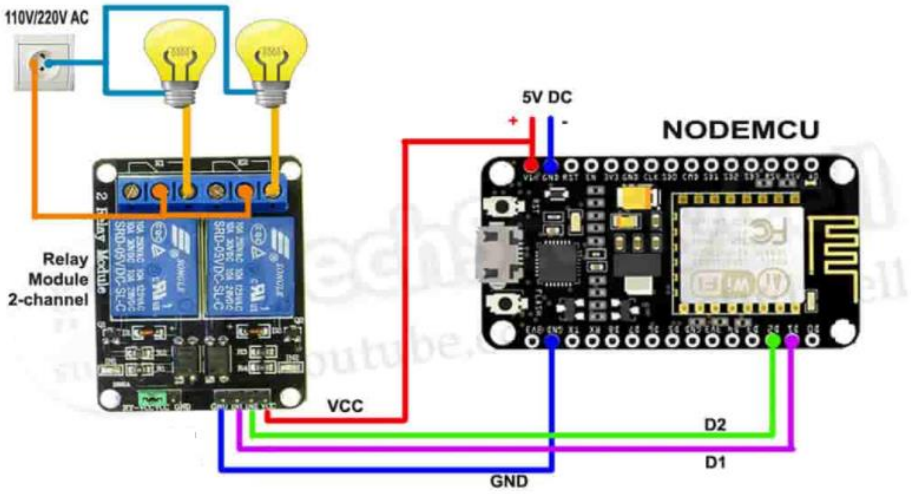


Figure: NodeMCU- Relay Module Circuit

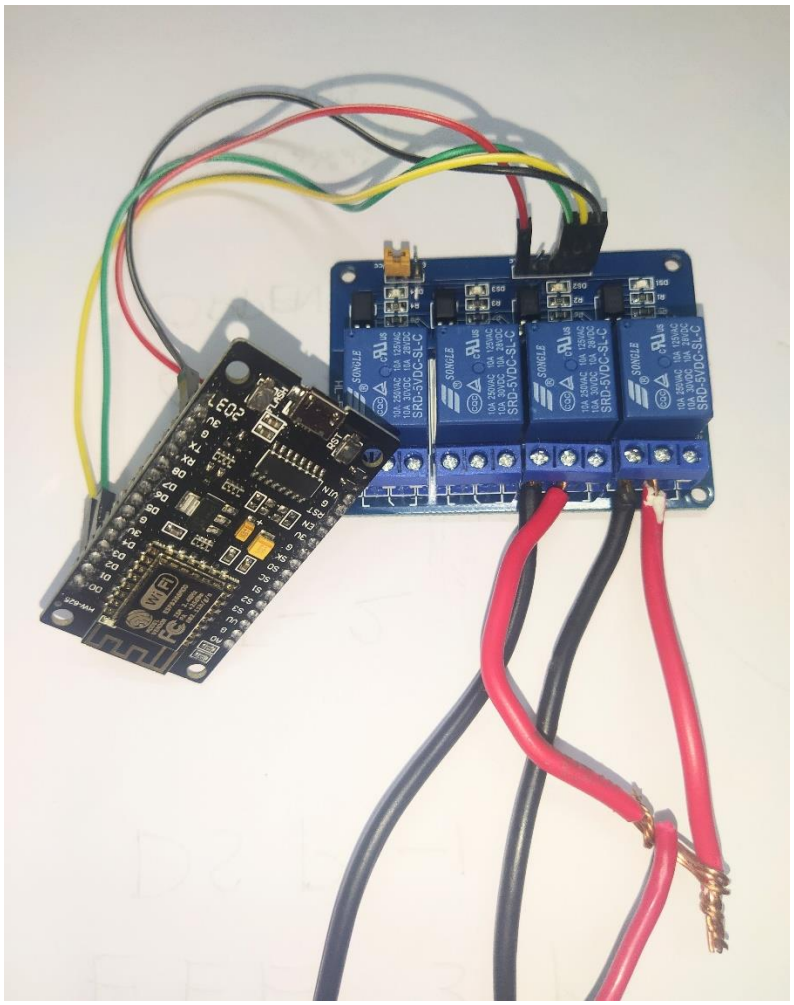


Figure: Practical Connection Between NodeMCU and Relay Module

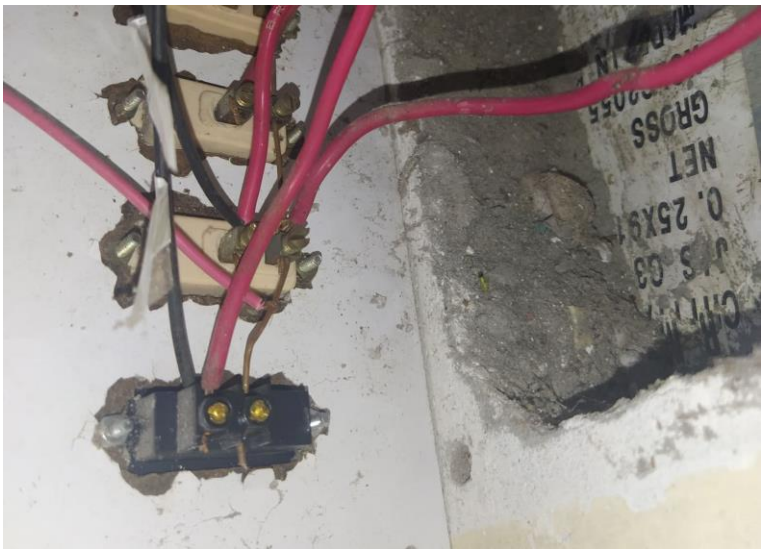
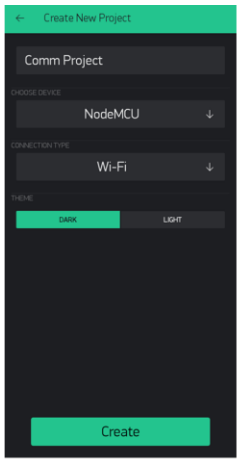


Figure: Practical Connection Behind Switch Board

3. Connect the Relay Module according to the circuit diagram in figure 1.

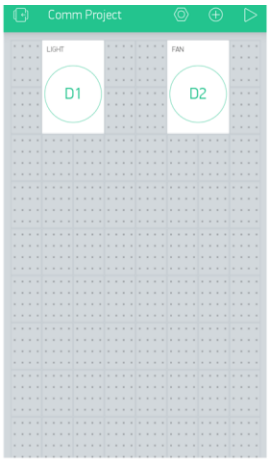
4. Build the switches in BLYNK app.



1. Create a new project in blynk selecting NodeMCU

2. By tapping on the "+" button, create a button for light

3. Create another button for fan in the same way



4. To connect blynk with NodeMCU, connect the charger of the NodeMCU and tap on the play button.

After creating the Blynk project, a token number was given to our email address. We copied the token and pasted it in the code. The final output is presented in a video file. The video file can be accessed using this link. The result was successful. The Blynk app successfully connected to the ESP over the Blynk cloud server and was able to control the home appliances without any problem.

(5) Room for Improvements:

1. Build a server on our own instead of cloud server. Raspberry pie and mosquito OS can be used in that purpose.
2. Control the fan speed using smart phone.
3. Find a better way to control the home appliances using both physical switch and smartphone.
4. Make the connection between broker and client more secure using password.

(6) Discussion:

Home automation has surely brought in a lot of hype in the recent era. The convenience factor of controlling all home devices from one place is enormous. Being able to keep all of the technology in your home connected through one interface is a massive step forward for technology and home management. All you just need is to press "on" from your smartphone and turning that device that on will certainly come to lessen our efforts. Moreover, bringing newer features of IoT into home automation may help us ensure home security and provide great level of flexibility in using various devices in near future. In our software project, we used a basic home-like environment and implemented home automation. In our hardware application though, we provided with a model that is really cost effective, simple & easy-to-use.

Here, we used a very simple model with simple components and used cloud server as a fundamental building block. There is a slight issue with that. The cloud server is quite slow and unsafe. So, building a server of our own & using it in our application would have been really handy to meet our goals & provide better service. Again, we could see that it took quite some time for the fan to turn off after switching in the software-based application. This is because our arduino has the ability to control the relay which is a simple thing to control the on-off operation. As a result, we had hardly any control over the fan speed. These are some issues related to our wireless home automation project. If can be fixed, our built model can be really impactful in bringing home automation to households of even mid class people.

(7) Appendix:

Code Used at Arduino IDE (simulation part):

```
#define BLYNK_PRINT SwSerial

#include <SoftwareSerial.h>

SoftwareSerial SwSerial(10, 11); // RX, TX

#include <BlynkSimpleStream.h>

char auth[] = "8HtWpwV6n5aon-*******-AUIeEfwA";  // Part of the authentication code is hidden intentionally
```

```
void setup()
{
  // Debug console
  SwSerial.begin(9600);

  // Blynk will work through Serial
  // Do not read or write this serial manually in your sketch
  Serial.begin(9600);
  Blynk.begin(Serial, auth);
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);
    digitalWrite(11,LOW);
  digitalWrite(13,LOW);
  digitalWrite(12,LOW);
 }

 void loop()
 {
   Blynk.run();
 }
```

Code used at Arduino IDE (hardware part):

```
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
char auth[] = "JhKD7TBr5gkwuIpuAPr3O4XOR8lywwxD";
// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "Google";
char pass[] = "22446688";
```

```
void setup()
 {
 // Debug console
Serial.begin(9600);
Blynk.begin(auth, ssid, pass);
// You can also specify server: //
Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 80);
//Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8080);
}
void loop()
{
Blynk.run();
}
```