Subject: Embedded System and RTOS	
Name of student:	
Class: BE (E&TC)	**Roll no.:**
Semester/Year: VII (2022-23)	**Exam no.:**
Date of performance:	**Date of submission:**
Examined by:	

Experiment No :

Construct IoT based Wireless Controlled Home Automation system using ESP8266.

AIM: Write a program to Construct IoT based Wireless Controlled Home Automation system using ESP8266

OBJECTIVES:

- To get acquaint with fundamental blocks of a Cloud-Ready IoT Application Implementation
- To understand the ESP8266 module and interfacing with Actuators.
- To understand the working of IoT platform (Blynk)
- To develop comprehensive approach towards building small low cost IoT application.

APPARATUS:

- ESP8266 DOIT DEVKIT V1
- Arduino IDE
- IOT Application- BLYNK

THEORY:

1. Introduction to IoT in Home Automation

By Home Automation we mean controlling lighting, climate, entertainment systems, and appliances without a manual switch. It may also include home security such as access control and alarm systems. When connected with the Internet, home devices are an important constituent of the Internet of Things ("IoT").

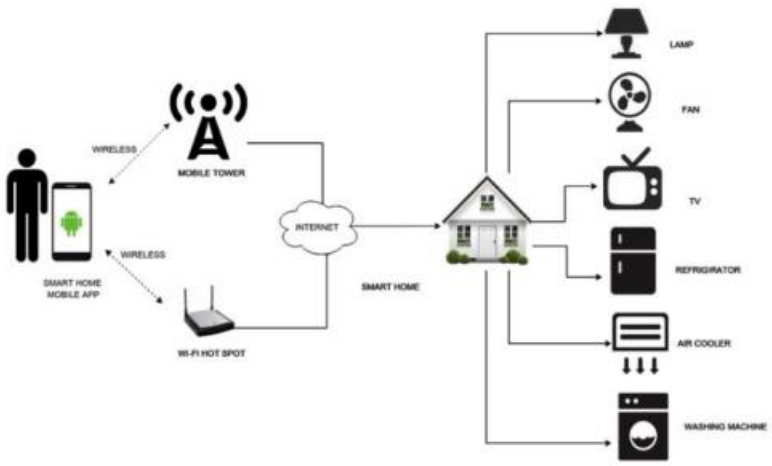


Figure 1: Architecture of IoT based Home Automation

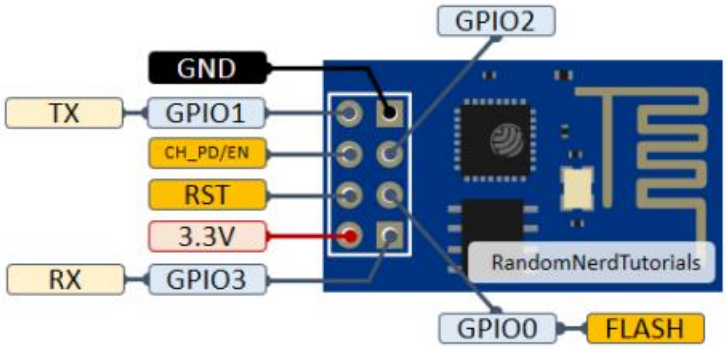
1.1 Network Connectivity Peripherals- ESP8266

The ESP8266 is a low-cost Wi-Fi microchip, with a full TCP/IP stack and microcontroller capability. It can be separately programmed and used directly as end node along with sensor and power supply, or it may be used as daughter board with additional microcontroller. In this experiment it is interfaced with Arduino Uno. Where through serial communication AT commands are sent to it for performing various tasks.

1.2 Features

- Processor: L106 32-bit RISC microprocessor core based on the Tensilica Xtensa Diamond Standard 106Micro running at 80 MHz
- Memory: 32 KB RAM
- Flash Memory: 512 KB-2MB
- IEEE 802.11 b/g/n Wi-Fi
- 10-bit ADC (successive approximation ADC)
- Supports, UART, SPI, I2C

Figure 2: Interfacing Diagram of ESP8266



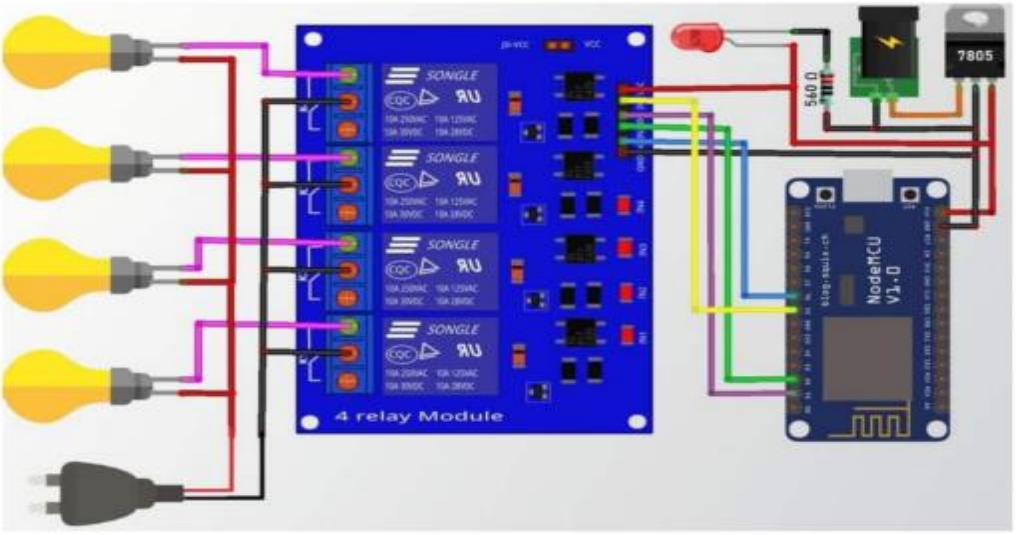


Figure 3: Circuit Diagram and Connection.

1.4 AT Commands

Following sequence of AT commands can be sent to ESP 8266 through Arduino via UART communication as shown in interfacing diagram

AT
AT+RST
AT+CWMODE=3
AT+CWLAP
AT+CWJAP="WIFI_ID","WIFI_PASSWORD"
AT+CIFSR
AT+CIPMUX=0
AT+CIPSTART="TCP","api.thingspeak.com",80
AT+CIPSEND=51
GET /update?api_key=XXXXXXXXXXXXXXXX&field1=255
AT+CIPCLOSE

Here, the 16 digit unique API write key is to be added along with sensor values in second last AT Command. The interpretation of AT commands is given at the end of this document.

2. IOT Platform (Blynk)

Blynk is an IoT platform for iOS or Android smartphones that is used to control Arduino, Raspberry Pi and NodeMCU via the Internet. This application is used to create a graphical interface or human machine interface (HMI) by compiling and providing the appropriate address on the available widgets.

Steps to setup Blynk

i. Create a new account using your Email ID.
ii. Click on "New Project".
iii. Follow the images below for the complete setup guide

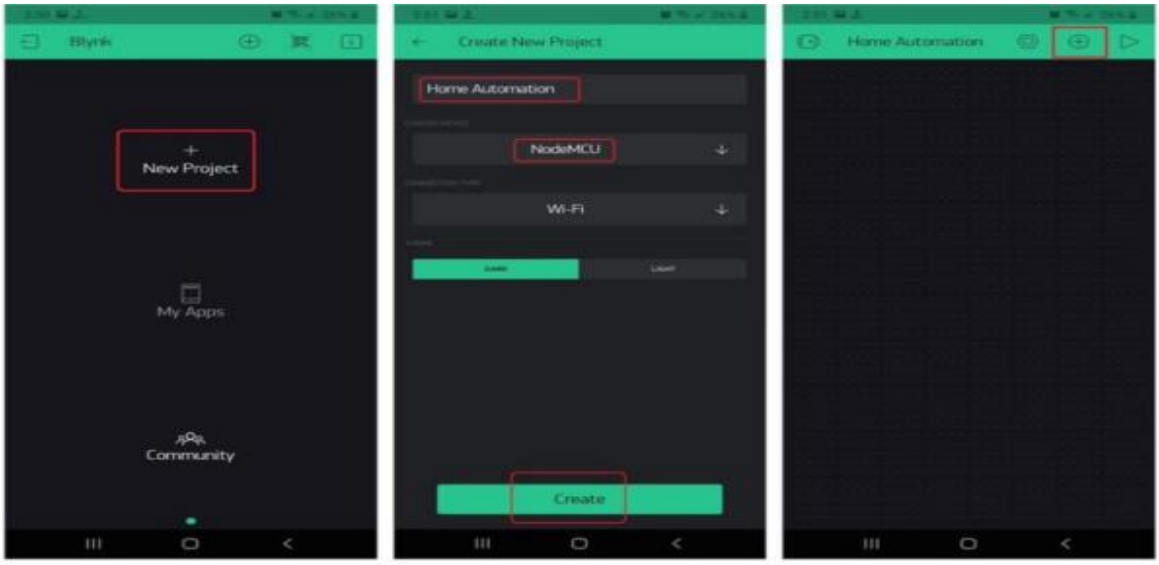


Figure 4: Steps for setting up blynk Application

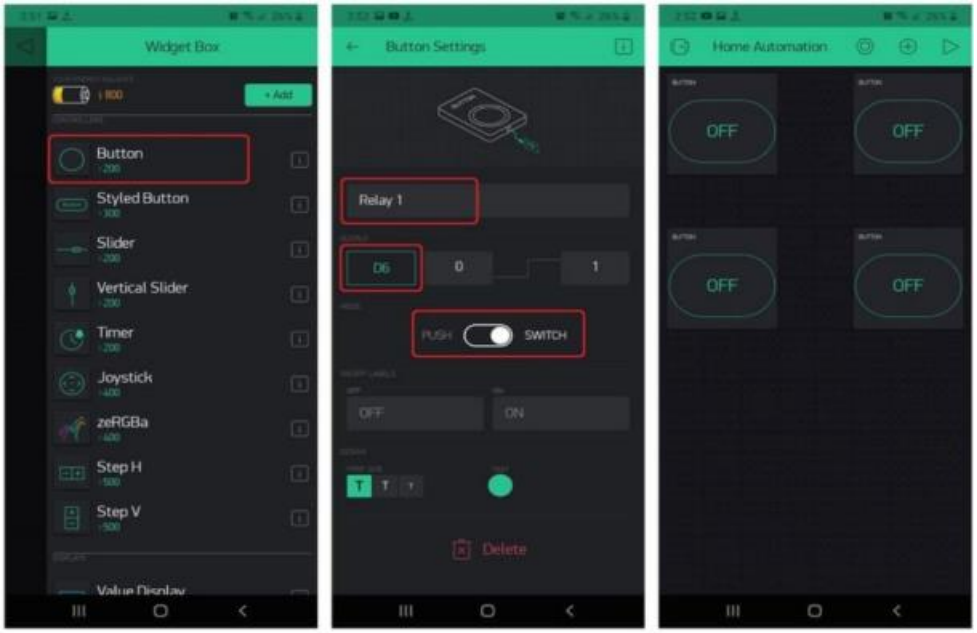


Figure 5: Steps for setting up blynk Application

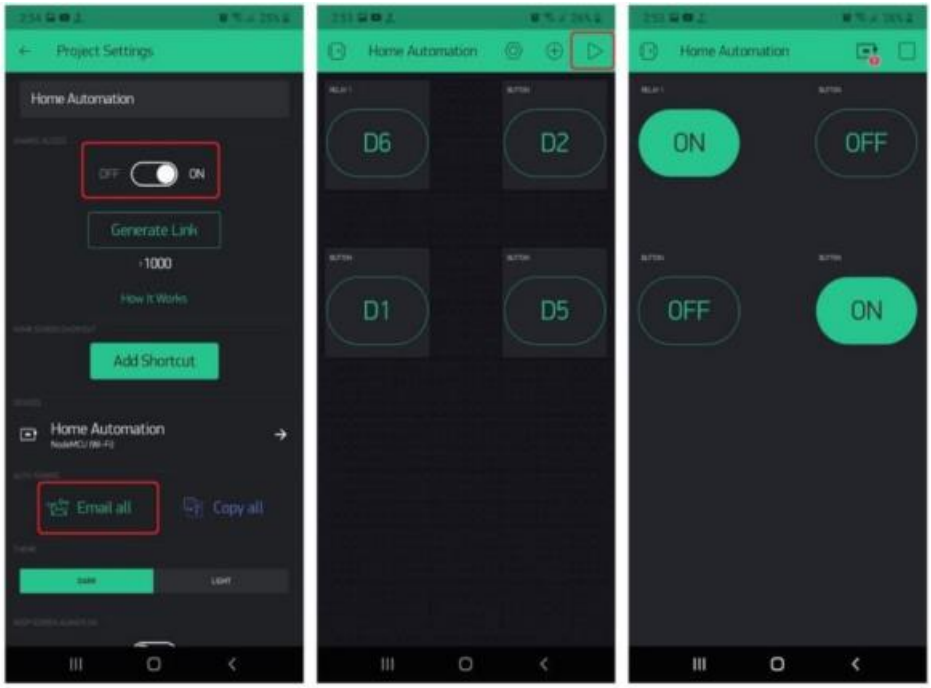
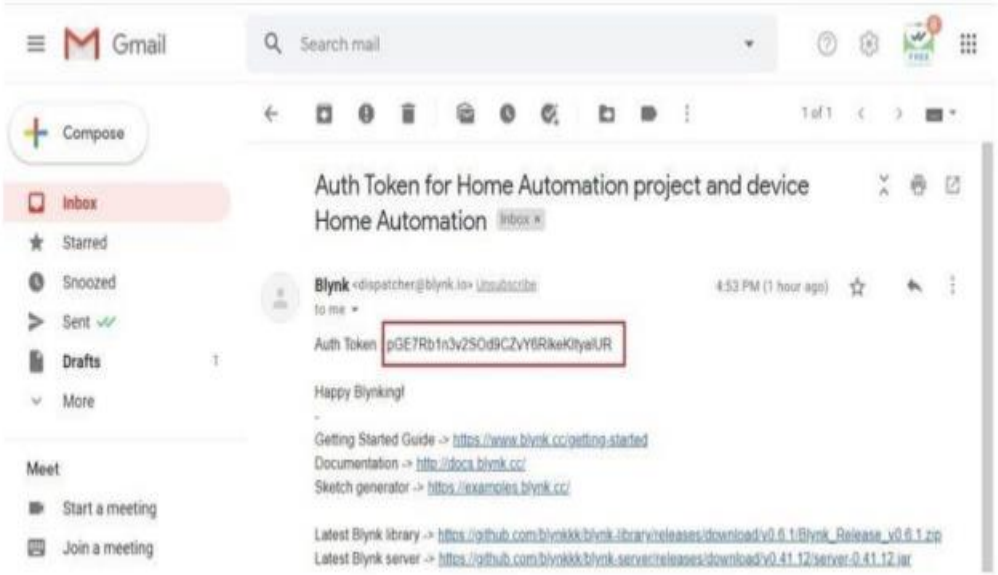


Figure 6: Steps for setting up blynk Application

iv. When you click on "Email All", an authentication token will be sent to your mail address. This token is very important as it is required in the program/code. So open your email ID & check for the token.



PROCEDURE:-

1. Make necessary connections as per Interfacing diagram.
2. Write code in Arduino IDE.
3. Verify Output

CONCLUSION:

REFERENCES:

a) ESP8266 Datasheet
b) https://docs.blynk.cc/
c) Wikipedia

SOURCE CODE:

```
/ Template ID, Device Name and Auth Token are provided by the Blynk.Cloud
// See the Device Info tab, or Template settings
#defin
e BLYNK_TEMPLATE_ID "TMPLiKV9fpaa"
#define BLYNK_DEVICE_NAME "Quickstart Device"
#define BLYNK_AUTH_TOKEN "JTIqzbGljMUty71F7q6W7dKh_O_YaYnp"
// Comment this out to disable prints and save space
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
char auth[] = BLYNK_AUTH_TOKEN;
// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "YourNetworkName";
char pass[] = "YourPassword";
BlynkTimer timer;
// This function is called every time the Virtual Pin 0 state changes
BLYNK_WRITE(V0)
{
 // Set incoming value from pin V0 to a variable
 int value = param.asInt();
 // Update state
 Blynk.virtualWrite(V1, value);
}
// This function is called every time the device is connected to the Blynk.Cloud
BLYNK_CONNECTED()
{
 // Change Web Link Button message to "Congratulations!"
 Blynk.setProperty(V3, "offImageUrl",
"https://static   image.nyc3.cdn.digitaloceanspaces.com/general/fte/congratulations.png");
 Blynk.setProperty(V3, "onImageUrl",
"https://static   image.nyc3.cdn.digitaloceanspaces.com/general/fte/congratulations_pressed.png"
);
 Blynk.setProperty(V3, "url", "https://docs.blynk.io/en/getting-started/what-do-i-need-
to   blynk/how-quickstart-device-was-made");
}
// This function sends Arduino's uptime every second to Virtual Pin 2.
void myTimerEvent()
{
 // You can send any value at any time.
 // Please don't send more that 10 values per second.
```

```
 Blynk.virtualWrite(V2, millis() / 1000);
}
void setup()
{
 // Debug console
 Serial.begin(115200);
 Blynk.begin(auth, ssid, pass);
 // You can also specify server:
 //Blynk.begin(auth, ssid, pass, "blynk.cloud", 80);
 //Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8080);
 // Setup a function to be called every second
 timer.setInterval(1000L, myTimerEvent);
}
void loop()
{
 Blynk.run();
 timer.run();
 // You can inject your own code or combine it with other sketches.
// Check other examples on how to communicate with Blynk. Remember
 // to avoid delay() function!
}
```

OUTPUT

Final Results of Controlling a Bulb with the help of Blynk Applicatio