**CHAPTER 1**

**INTRODUCTION**

A home automation system will control lighting, climate, entertainment systems, and appliances. 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](https://en.m.wikipedia.org/wiki/Internet_of_Things)

**Amazon Alexa**, known simply as **Alexa** is a [virtual assistant](https://en.m.wikipedia.org/wiki/Virtual_assistant) developed by [Amazon](https://en.m.wikipedia.org/wiki/Amazon_(company)). It is capable of voice interaction, music playback, making to-do lists, [setting alarms](https://en.m.wikipedia.org/wiki/Alarm_clock), streaming podcasts, playing audiobooks, and providing weather, traffic, sports, and other real-time information, such as [news](https://en.m.wikipedia.org/wiki/News).[[3]](https://en.m.wikipedia.org/wiki/Amazon_Alexa#cite_note-3) Alexa can also control several [smart devices](https://en.m.wikipedia.org/wiki/Smart_device) using itself as a [home automation](https://en.m.wikipedia.org/wiki/Home_automation) system. We are able to extend the Alexa capabilities by installing "skills" (additional functionality developed by third-party vendors, in other settings more commonly called [apps](https://en.m.wikipedia.org/wiki/Mobile_app) such as weather programs and audio features)

**NodeMCU** is an open source LUA based firmware developed for ESP8266 wifi chip. It is an open source IoT platform that includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. By default the NodeMCU firmware uses the Lua scripting language which is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects.

Our project is an attempt to utilize the benefits of NodeMCU ESP8266 and Amazon Alexa to improvise home automation so that it is more accessible and user friendly. We are also making customization easier and more efficient.

The usual and conventional methods of home automation utilises technologies like IFTT, ThingsSpeak and so on. But these methods require configuration on every platform and this makes the task little cumbersome and time taking. Also, coding part need some calculations for getting characters and decoding them. Therefore the uniqueness of our project involves the use of a library known as fauxmoESP which makes home automation more simpler and easier.

**CHAPTER 2**

**Hardware & software Requirements**

1. Amazon Alexa dot speaker

2. ESP8266-01 / Node MCU

3. Jumper Wires

4. Breadboard or Zero PCB

5. Arduino IDE

6. 4 channel relay

7. LED's

8. 3V battery

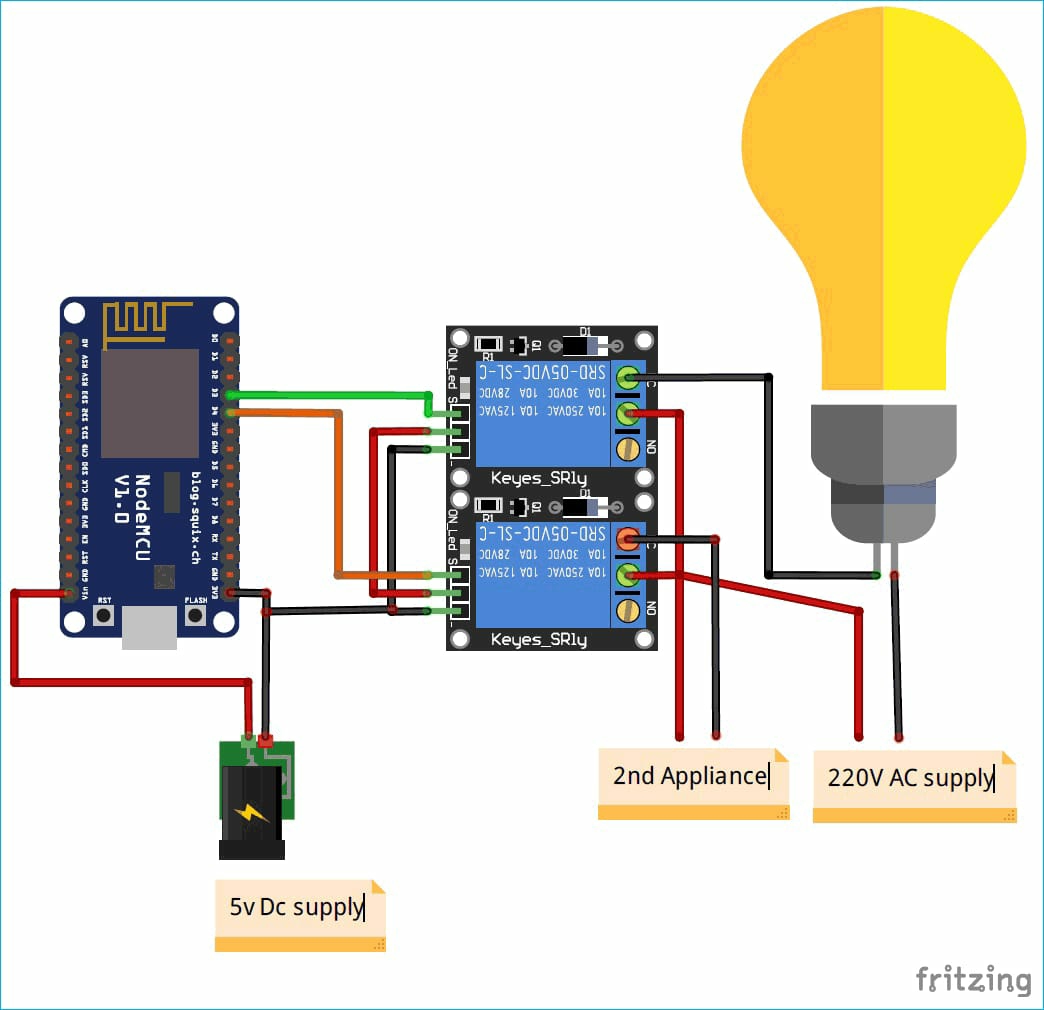
9. 6V battery

10. AC Supply

**CHAPTER 3**

**DESIGN METHOD**

* **BLOCK DIAGRAM / CIRCUIT DIAGRAM**

****

**CHAPTER 4**

**MODULE DESCRIPTION**

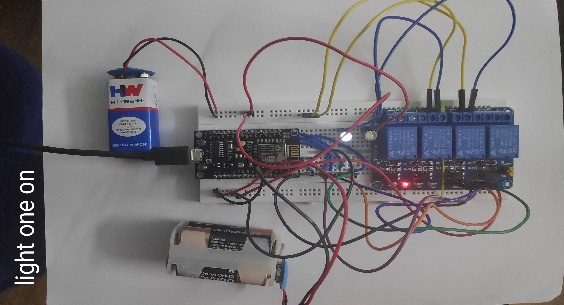
* **DIFFERENT MODULES USED :**

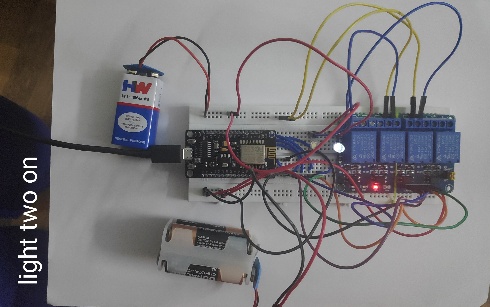
NodeMCU (ESP8266), Relay Module

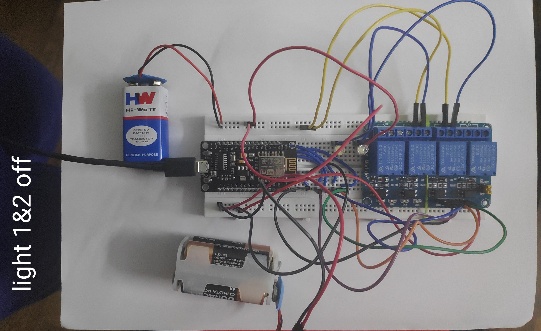
* **WORKING OF THE CIRCUIT** :
* The aim of our project is to switch on and off small devices (in our case 2 LED’s- light one and light two) . To do this first, we include important header files for ESP board and fauxmoESP and also, define pin number for the relays.
* We define baud rate of 115200 and Wi-Fi-SSID and Password
* Next we make a separate function for Wi-Fi setup so that it can be called in void setup function. We make the WiFi mode as station mode and pass the SSID and Password in WiFi.begin() function.
* In **void setup()** function, we pass the baud rate to serial.begin function and call the wifisetup function. We make relay pins high or low by default.
* Now, fauxmoESP has to create its own webserver, for this we pass true in createserver function, enable function and set port number as 80.
* Next, we add devices using fauxmo.addDevice() function.
* Now, we make a function when a command is received from Alexa. In this function we will compare the string with the device name when it is matched then change the state of  the device according to the command given.
* Similarly, we repeat the above steps for our second device.
* Finally we upload the complete code (given at the end) in NodeMCU after connecting the circuit as per the circuit diagram shown above.
* Lastly we check the working of the circuit by telling Alexa to turn on and off l ight one and light two respectively.

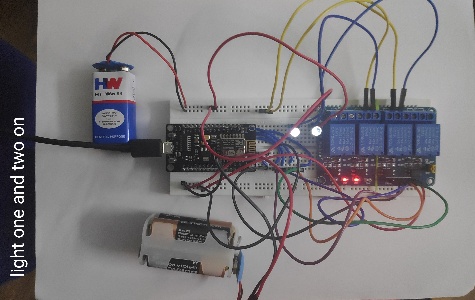
**CODING OF FUNCTIONAL MODULES**

/\* #include <Arduino.h>  
  #ifdef ESP32  
  #include <WiFi.h>  
  #define RELAY\_PIN\_2 12//D6  
  #define RELAY\_PIN\_3 14//D5  
  #else  
  \*/  
  #include <ESP8266WiFi.h>  
  //#define RF\_RECEIVER 5  
  #define RELAY\_PIN\_2 15 //D8    
  #define RELAY\_PIN\_3 3 //RX  
  // #endif  
#include "fauxmoESP.h"  
#define SERIAL\_BAUDRATE 115200  
  
#define WIFI\_SSID "~vx"  
#define WIFI\_PASS "12345678"  
#define app1 "light one"  
#define app2 "light two"  
fauxmoESP fauxmo;  
   
void wifiSetup() {  
  // Set WIFI module to STA mode  
  WiFi.mode(WIFI\_STA);  
  Serial.printf("[WIFI] Connecting to %s ", WIFI\_SSID);  
  WiFi.begin(WIFI\_SSID, WIFI\_PASS);  
  while (WiFi.status() != WL\_CONNECTED) {  
    Serial.print(".");  
    delay(100);  
  }  
  Serial.println();  
  Serial.printf("[WIFI] STATION Mode, SSID: %s, IP address: %s\n", WiFi.SSID().c\_str(), WiFi.localIP().toString().c\_str());  
}  
  
void setup() {  
  // Init serial port and clean garbage  
  Serial.begin(SERIAL\_BAUDRATE);  
  Serial.println();  
  
  // Wi-Fi connection  
  wifiSetup();  
  
  pinMode(RELAY\_PIN\_2, OUTPUT);  
  digitalWrite(RELAY\_PIN\_2, HIGH);  
  
  pinMode(RELAY\_PIN\_3, OUTPUT);  
  digitalWrite(RELAY\_PIN\_3, HIGH);  
   
  fauxmo.createServer(true);  
  fauxmo.setPort(80);  
  fauxmo.enable(true);  
  fauxmo.addDevice(app1);  
  fauxmo.addDevice(app2);  
  
  fauxmo.onSetState([](unsigned char device\_id, const char \* device\_name, bool state, unsigned char value) {      
    Serial.printf("[MAIN] Device #%d (%s) state: %s value: %d\n", device\_id, device\_name, state ? "ON" : "OFF", value);  
    if ( (strcmp(device\_name, app1) == 0) ) {  
      Serial.println("RELAY 2 switched by Alexa");  
      if (state) {  
        digitalWrite(RELAY\_PIN\_2, LOW);  
      } else {  
        digitalWrite(RELAY\_PIN\_2, HIGH);  
      }  
    }  
    if ( (strcmp(device\_name, app2) == 0) ) {  
      Serial.println("RELAY 3 switched by Alexa");  
      if (state) {  
        digitalWrite(RELAY\_PIN\_3, LOW);  
      } else {  
        digitalWrite(RELAY\_PIN\_3, HIGH);  
      }  
    }  
  });  
  
}  
  
void loop() {  
  fauxmo.handle();  
  
  static unsigned long last = millis();  
  if (millis() - last > 5000) {  
    last = millis();  
    Serial.printf("[MAIN] Free heap: %d bytes\n", ESP.getFreeHeap());  
  }  
}

**CHAPTER 5 - RESULTS**







**CHAPTER 6**

**CONCLUSION**

We have successfully designed a circuit that utilises NodeMCU and Amazon’s Alexa to turn devices ( light one and light two) on and off. We have emulated the concepts of IoT and Home Automation on a smaller scale.

**CHAPTER 7**

**FUTURE WORK**

Our future work on this project will be to implement this circuit on a larger scale ie.., to use this to turn on and off bigger appliances like room lights , fans, charging ports etc..,

The major aim of this project is to make everyday tasks easier by implementing technology efficiently. This idea will be useful for patients in hospitals, elderly people and the physically challenged by helping them do these small tasks easily**.**

**REFERENCES**

* [**https://iotdesignpro.com/projects/alexa-controlled-iot-home-automation-by-emulating-a-wemo-device-using-nodemcu**](https://iotdesignpro.com/projects/alexa-controlled-iot-home-automation-by-emulating-a-wemo-device-using-nodemcu)
* [**https://www.instructables.com/id/How-To-DIY-Home-Automation-With-NodeMCU-and-Amazon/**](https://www.instructables.com/id/How-To-DIY-Home-Automation-With-NodeMCU-and-Amazon/)