## Project-1: Room Automation Using an Android App.

# Introduction:

This project is based on the concept of Room Automation using the android app and the Firebase Database. Here I am using ESP8266 or NodeMCU, which is the most common WiFi development board. ( If you don't have an idea about that kind of development board, visit this **link.** ) So, the basic concept is that we are building an android app and a Hardware Box ( contains Relays, Power Module And NodeMCU ). In the Android App, you will find buttons for individual lights. The most important part of this project is to design a circuit that can be controlled in both ways (i.e. Manually and Via Internet ). Here I am using Firebase Database to establishing a connection in between our Android App and NodeMCU.

# Requirements:

| | |
|---|---|
| 1. ESP8266 x1 | Buy From Amazon |
| 3. Relay Module x1 | Buy From Amazon |
| 4. Power Module x1 | Buy From Amazon |
| 3. Lights x2 | Buy From Amazon |
| 4. Jumper Wires | Buy From Amazon |
| 5. Arduino IDE | Download |

# How It's Made?

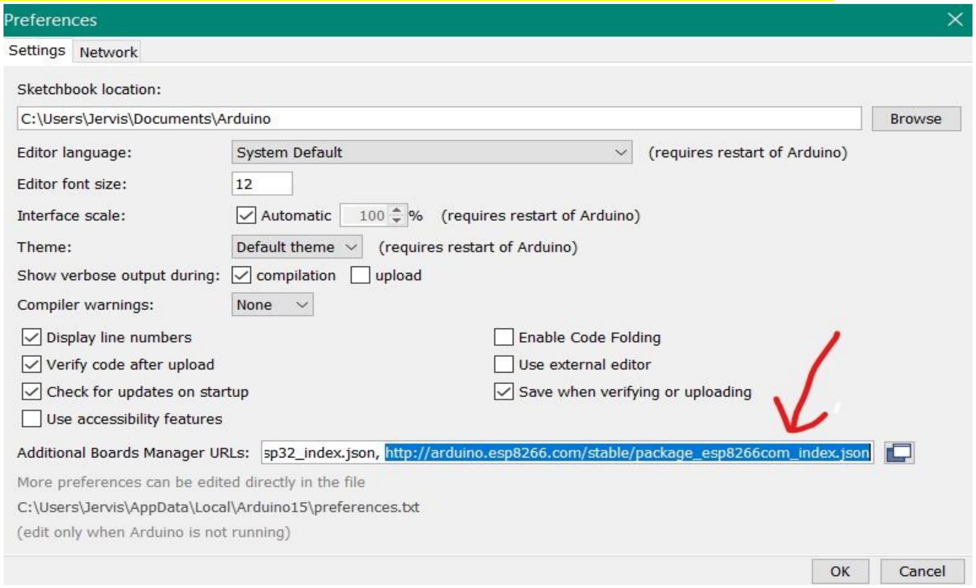
I have divided this section into five parts. i.e.

A. Setup the NodeMCU And Install the Libraries.
B. Setup the Firebase Database And Connect to the NodeMCU.
C. Simulate the Circuit in Proteus And Wiring up with the NodeMCU.
D. Link Android Studio to the Firebase Database.
E. Create the App in Android Studio.

# A.Setup the NodeMCU And Install the Libraries:

### I. Installing ESP8266 Board Manager:

NodeMCU is an ESP8266 based development board which is very easy to program with the help of the Arduino IDE. You need to Install Arduino IDE to program the NodeMCU. I have attached the Download Link for Arduino IDE. The installation of this software is simple. Just follow the instructions that are mentioned in the installation window. Now you have to add a JSON link in the preferences. Just open the IDE and go to the File -> Preferences. A preferences window will open. You will find an input box labeled as **"Additional Boards Manager URLs".** Copy this link ( http://arduino.esp8266.com/stable/package_esp8266com_index.json ) into that input box



Preferences

Settings | Network

Sketchbook location:
C:\Users\Jervis\Documents\Arduino [Browse]

Editor language: System Default (requires restart of Arduino)

Editor font size: 12

Interface scale: ☑ Automatic 100 % (requires restart of Arduino)

Theme: Default theme (requires restart of Arduino)

Show verbose output during: ☑ compilation □ upload

Compiler warnings: None

☑ Display line numbers
☑ Verify code after upload
☑ Check for updates on startup
□ Use accessibility features
□ Enable Code Folding
□ Use external editor
☑ Save when verifying or uploading

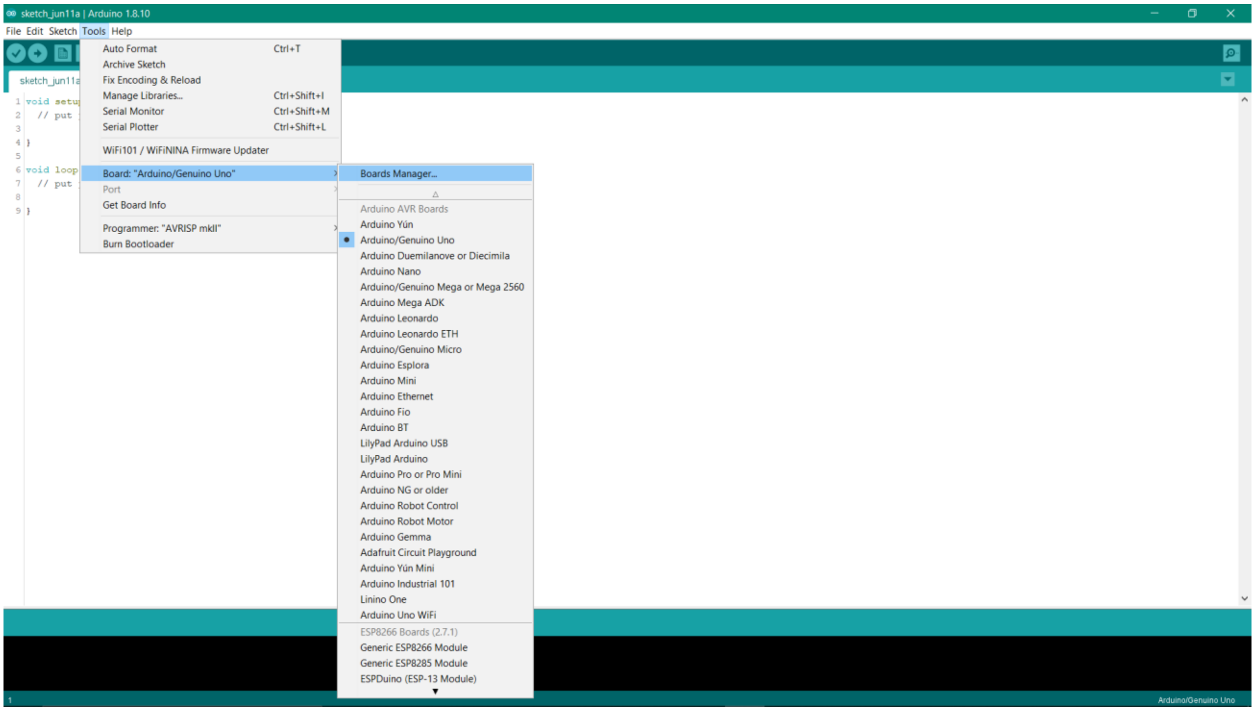
Additional Boards Manager URLs: sp32_index.json, http://arduino.esp8266.com/stable/package_esp8266com_index.json

More preferences can be edited directly in the file
C:\Users\Jervis\AppData\Local\Arduino15\preferences.txt
(edit only when Arduino is not running)

[OK] [Cancel]

and restart the IDE. Now you are ready to install the required Board Manager for your NodeMCU board.

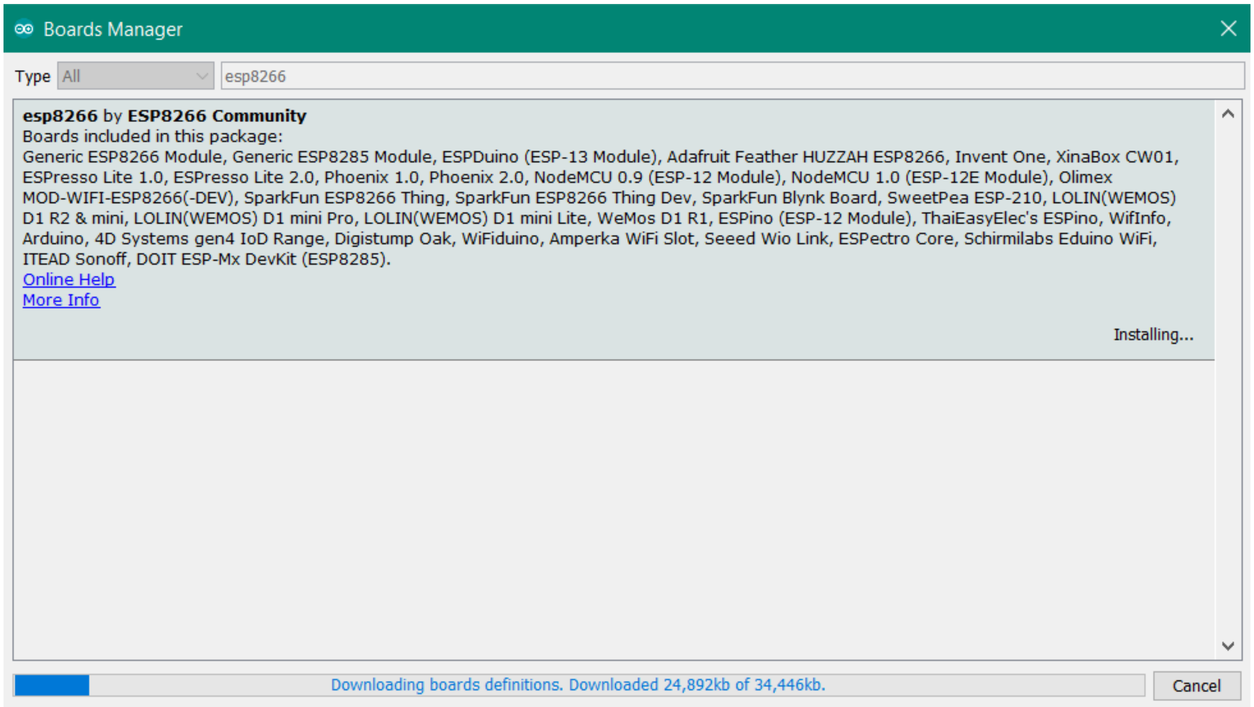
To install the ESP8266 Board Manager just take your cursor to the Tools -> Boards -> Board Manager. When you click on the Board manager window will pop up on you screen. Here you will



```
void setup() {
  // put
}

void loop() {
  // put
}
```

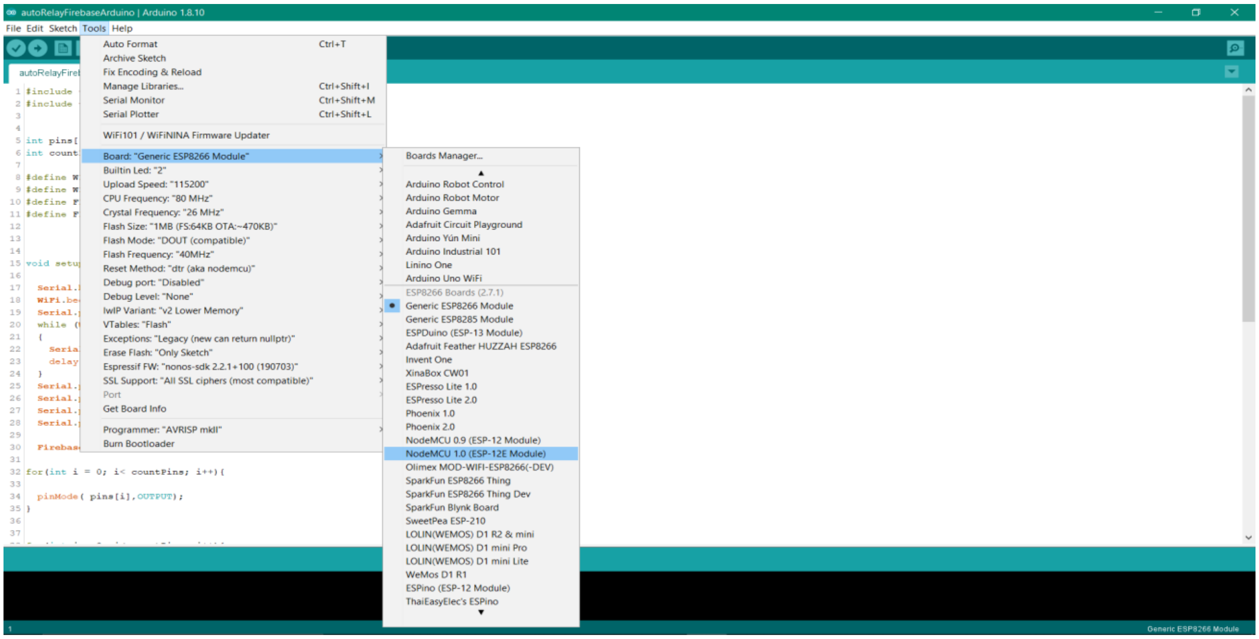
find all of the development boards that are used in Arduino IDE so far.  Just search for ESP8266 and click install. Again restart your IDE



**esp8266** by **ESP8266 Community**
Boards included in this package:
Generic ESP8266 Module, Generic ESP8285 Module, ESPDuino (ESP-13 Module), Adafruit Feather HUZZAH ESP8266, Invent One, XinaBox CW01, ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Olimex MOD-WIFI-ESP8266(-DEV), SparkFun ESP8266 Thing, SparkFun ESP8266 Thing Dev, SparkFun Blynk Board, SweetPea ESP-210, LOLIN(WEMOS) D1 R2 & mini, LOLIN(WEMOS) D1 mini Pro, LOLIN(WEMOS) D1 mini Lite, WeMos D1 R1, ESPino (ESP-12 Module), ThaiEasyElec's ESPino, WifInfo, Arduino, 4D Systems gen4 IoD Range, Digistump Oak, WiFiduino, Amperka WiFi Slot, Seeed Wio Link, ESPectro Core, Schirmilabs Eduino WiFi, ITEAD Sonoff, DOIT ESP-Mx DevKit (ESP8285).

Downloading boards definitions. Downloaded 24,892kb of 34,446kb.

So, now you have your required reference files for ESP8266 chip. Before you move forward please go to the board in tool section again. You will find a section for ESP8266. Hare you don't need to confuse as there so many boards are available for selection in this section. So, Connect your NodeMCU Board and select the NodeMCU 1.0 (ESP-12E Module) (Note: If you have 12F Module, select the generic ESP8266 option from that board menu).



Board: "Generic ESP8266 Module"
Builtin Led: "2"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Crystal Frequency: "26 MHz"
Flash Size: "1MB (FS:64KB OTA:~470KB)"
Flash Mode: "DOUT (compatible)"
Flash Frequency: "40MHz"
Reset Method: "dtr (aka nodemcu)"
Debug port: "Disabled"
Debug Level: "None"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
Exceptions: "Legacy (new can return nullptr)"
Erase Flash: "Only Sketch"
Espressif FW: "nonos-sdk 2.2.1+100 (190703)"
SSL Support: "All SSL ciphers (most compatible)"
Programmer: "AVRISP mkII"

```
for(int i = 0; i< countPins; i++){

  pinMode( pins[i],OUTPUT);
}
```

### II. Installing the Libraries that required in this project:

Now you need to install the required libraries in this. Download our project folder and search for a folder named as ‘Required Library’. You will find another folder inside this that is ‘firebase-arduino-master’. Copy this folder and paste this where your Arduino libraries folder is located. If do not know the location just go to the preferences under the file tab in Arduino IDE. There You will find a path named as Sketchbook location. Open this path in your file manager and search for the ‘libraries’ folder. Paste the ‘firebase-arduino-master’ folder in the ‘libraries’ folder. Now restart your IDE again. Bingo, you are now ready to move towards the programming part of the NodeMCU board.

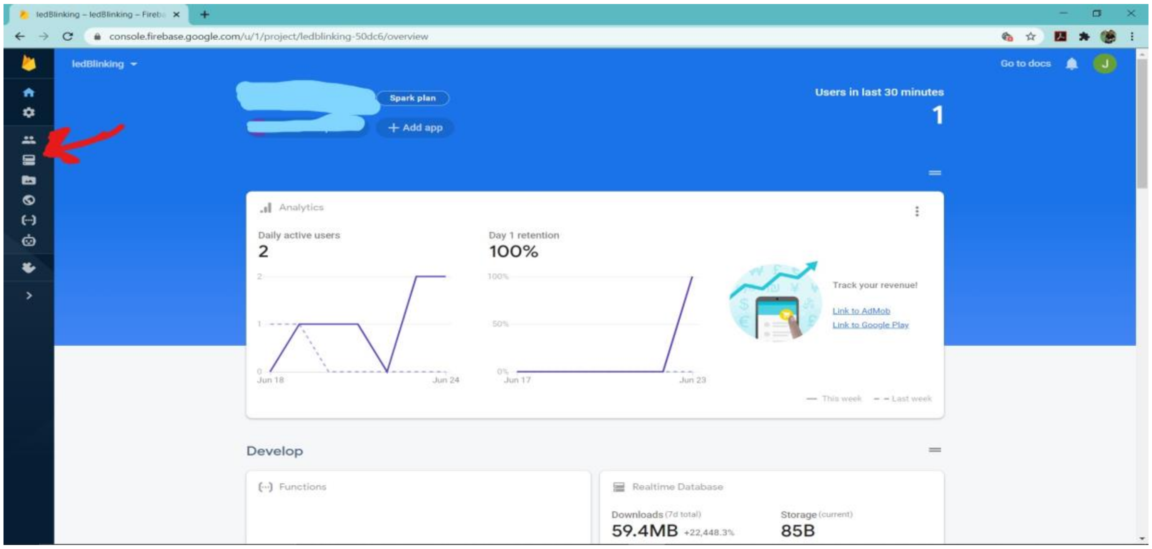
## B. Setup the Firebase Database And Connect to the NodeMCU:

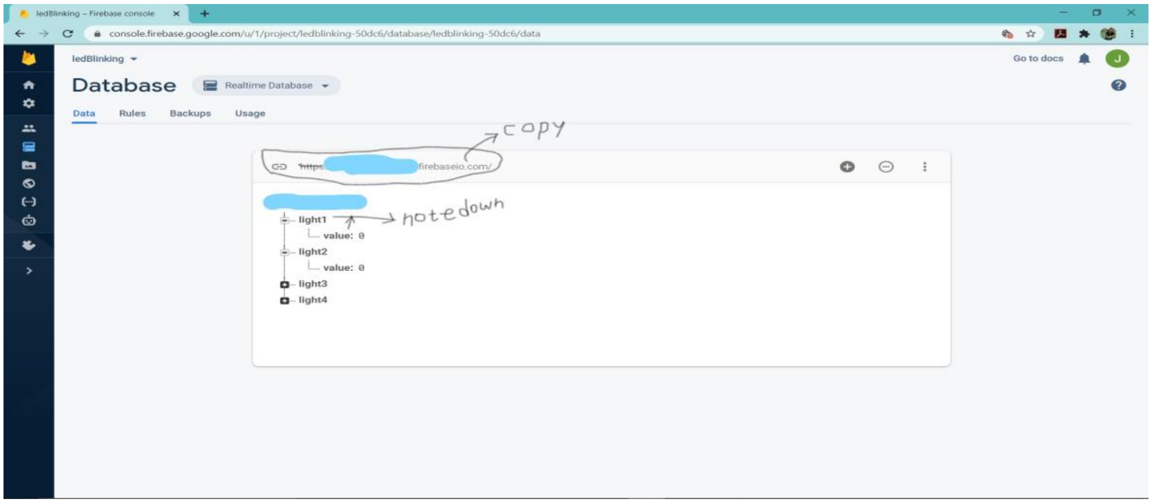
### I. Create a New Project in Firebase console:

Open your browser and search for firebase console. (Make sure that you have a Gmail account) Then open your firebase console account by signing in to your google account. You will find a new project tab. Just click on that and enter your project name or details and hit create project button. Great now you have created your first firebase project.

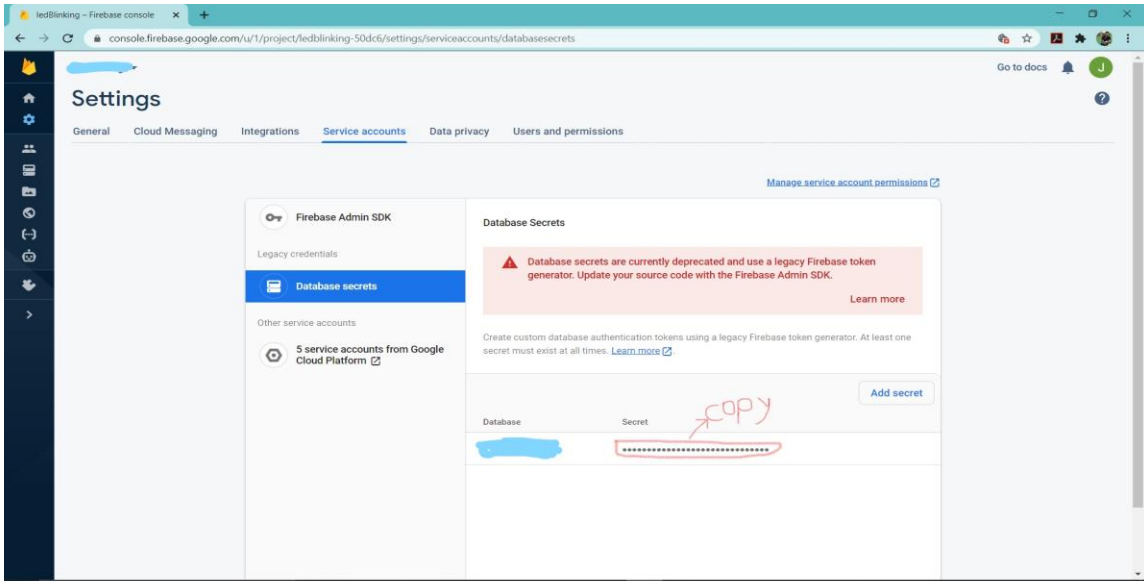
### II. Create your Firebase Project and Setup Real time Database:

You will find a Database tab on your firebase project dashboard. Click on that and you will be redirected to your database section. Find realtime database and hit the create a database button under this section. ( Make sure that you have check on the read and write permission) Now you have your Realtime Database panel. So, let’s create your database keys or parameters. In my case I have created my own parameters and values as per the figure shows. ( i.e. light1,light2,light3) you are free to create your own values and parameters but don’t forget to write down those heads because we are going to call this parameters on our nodemcu board. One more last step remaining that is the database link and secret key. You will find your database link just above the parameter section. Copy that link and paste on your notebook.(make sure that you are copying this without https:// part.) Now go to the Settings -> Service Accounts -> Database Secrets. Here you will find your database secrets. Copy this secrets on the same notebook. Superb now you are all set to move towards your nodemcu programing part.




ledBlinking – Firebase console
console.firebase.google.com/u/1/project/ledblinking-50dc6/database/ledblinking-50dc6/data
ledBlinking
Go to docs
Database
Realtime Database
Data
Rules
Backups
Usage
copy
https [redacted] firebaseio.com/
notedown

```
[redacted]
 ├─ light1
 │   └─ value: 0
 ├─ light2
 │   └─ value: 0
 ├─ light3
 └─ light4
```



Database secrets are currently deprecated and use a legacy Firebase token generator. Update your source code with the Firebase Admin SDK.

Create custom database authentication tokens using a legacy Firebase token generator. At least one secret must exist at all times. Learn more

| Database | Secret |
|---|---|
| [redacted] | •••••••••••••••••••••••••••••• |

### III. Prepare the Program on NodeMCU:

Here is the github link for the code. Clone this repo on your desktop and open the roomautomation.ino file. You will find that there are some ***** field in the code section. Replace those with your own data as follows. FIREBASE_HOST = “the database link you copied on your notepad”, FIREBASE_KEY= “Database secret key ”, WIFI_SSID= “ Your WiFi SSID ”, WIFI_PASSWORD = “ WiFi Password ”. Don’t worry about that section as I have explained every single line of codes through comments. Just follow those comments. So now finally the setup section and code section of hardware end is over. Now let’s move to the design of the circuit section.

## C. Simulate the circuit on Proteus Software and Wiring up with the NodeMCU:

I have discuss this section in detail on my youtube channel. Just click on this link.