Lab # 13 & 14

Control Home Appliances using MIT App and Google Firebase Real-time Database

Objectives

- Connect DH11 Humidity/Temperature Sensor to Local Server
- Use Smart Config to configure anonymous ESP32 with local Wifi
- Real time database creation using Google Firebase
- Connectivity of database with Android App and ESP32 Module
- Control of home appliances using ESP32

Tools

- Arduino
- ESP32 Board
- 2 LEDs
- DHT11 sensor
- Relay board
- 220V AC Light

Pre Lab

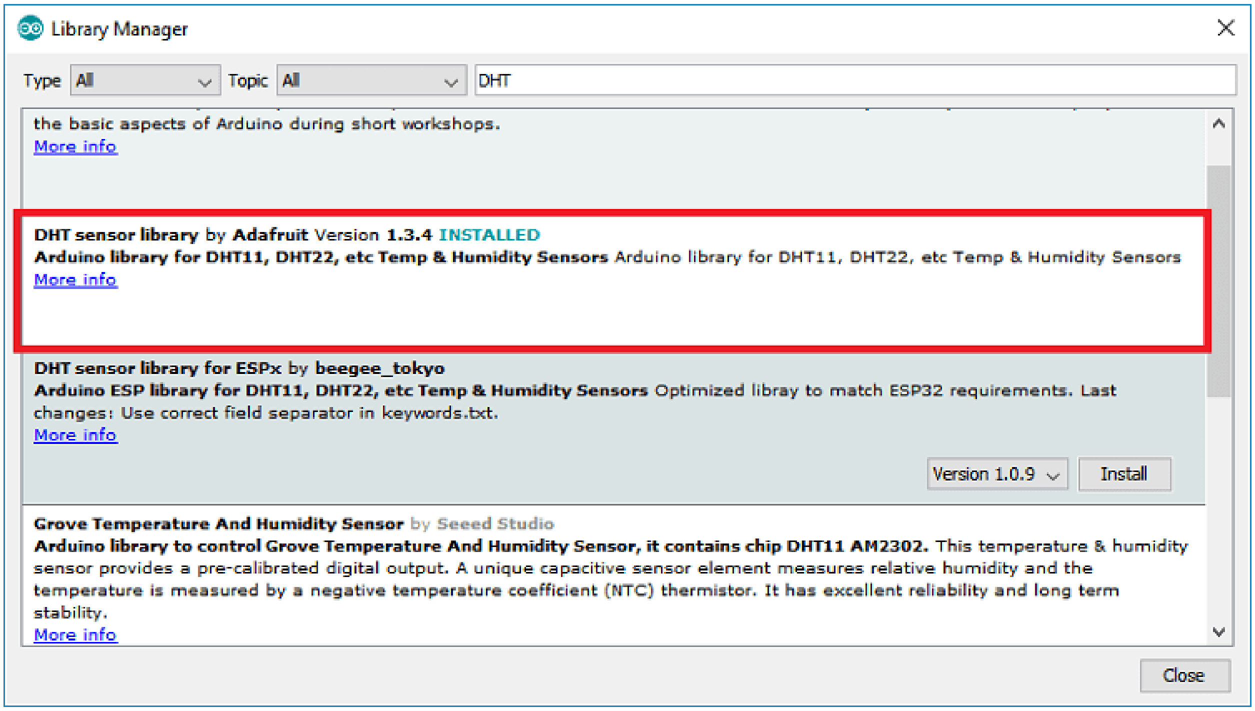
Please go through the data sheet of Expressif ESP32 development board.

Task 1:

Temperature/Humidity Sensor Read using DHT Sensor:

Step 1: Installing the DHT Library for ESP8266

1. Open your Arduino IDE and go to **Sketch** > **Include Library** > **Manage Libraries**. The Library Manager should open.
2. Search for "**DHT**" on the Search box and install the DHT library from Adafruit.



Library Manager

Type All Topic All DHT

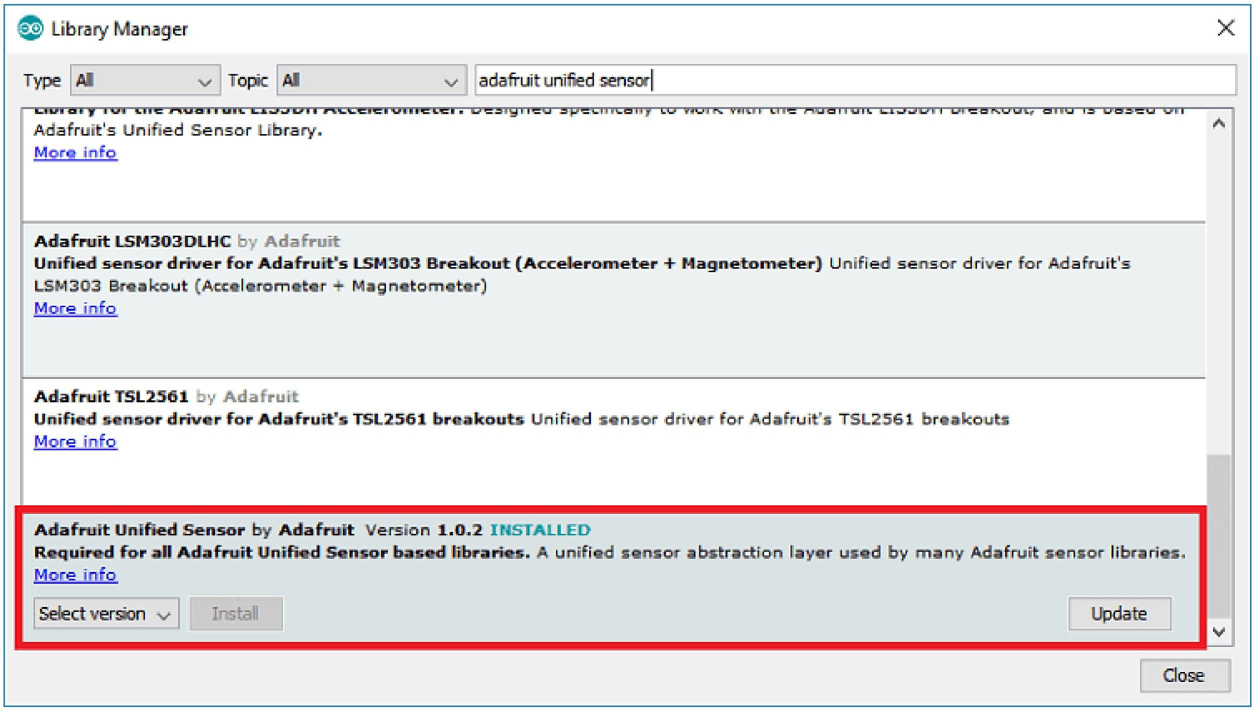
the basic aspects of Arduino during short workshops.
More info

DHT sensor library by **Adafruit** Version **1.3.4** INSTALLED
Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors
More info

DHT sensor library for ESPx by **beegee_tokyo**
Arduino ESP library for DHT11, DHT22, etc Temp & Humidity Sensors Optimized libray to match ESP32 requirements. Last changes: Use correct field separator in keywords.txt.
More info

Grove Temperature And Humidity Sensor by Seeed Studio
Arduino library to control Grove Temperature And Humidity Sensor, it contains chip DHT11 AM2302. This temperature & humidity sensor provides a pre-calibrated digital output. A unique capacitive sensor element measures relative humidity and the temperature is measured by a negative temperature coefficient (NTC) thermistor. It has excellent reliability and long term stability.
More info

3. After installing the DHT library from Adafruit, type “**Adafruit Unified Sensor**” in the search box. Scroll all the way down to find the library and install it.



Library Manager

Type All Topic All adafruit unified sensor

Library for the Adafruit LIS3DH Accelerometer. Designed specifically to work with the Adafruit LIS3DH Breakout, and is based on
Adafruit's Unified Sensor Library.
More info

Adafruit LSM303DLHC by Adafruit
Unified sensor driver for Adafruit's LSM303 Breakout (Accelerometer + Magnetometer) Unified sensor driver for Adafruit's LSM303 Breakout (Accelerometer + Magnetometer)
More info

Adafruit TSL2561 by Adafruit
Unified sensor driver for Adafruit's TSL2561 breakouts Unified sensor driver for Adafruit's TSL2561 breakouts
More info

Adafruit Unified Sensor by **Adafruit** Version **1.0.2** INSTALLED
Required for all Adafruit Unified Sensor based libraries. A unified sensor abstraction layer used by many Adafruit sensor libraries.
More info

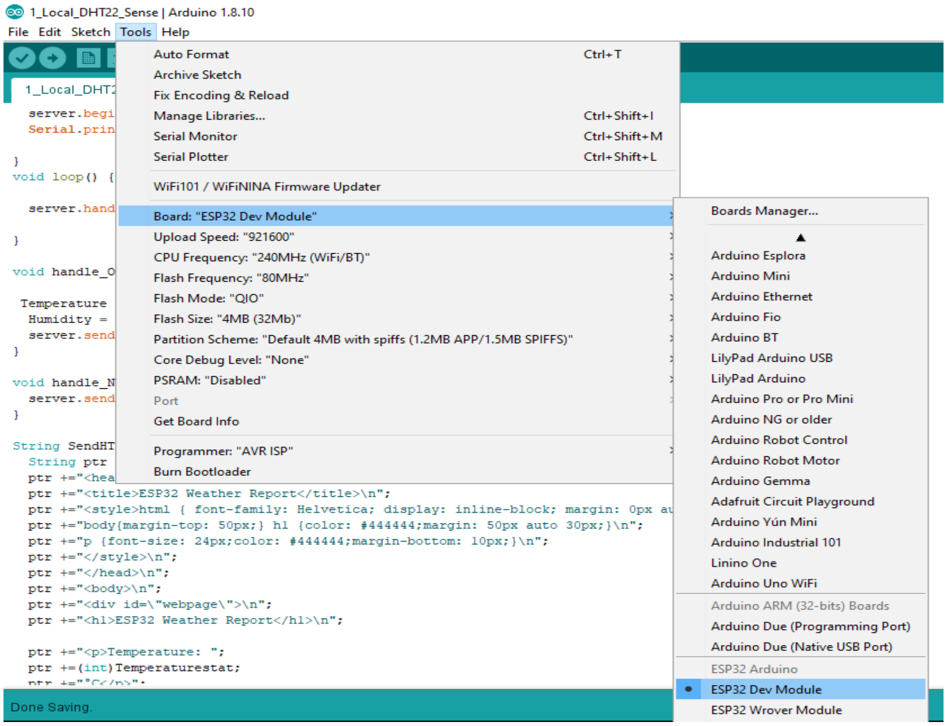
After installing the libraries, restart your Arduino IDE.

Step 2: Copy and Paste following Code

Example Code:

```
#include <WiFi.h>
#include <WebServer.h>
#include "DHT.h"

// Uncomment one of the lines below for whatever DHT sensor type you're using!
#define DHTTYPE DHT11   // DHT 11
//#define DHTTYPE DHT21   // DHT 21 (AM2301)
//#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321

/*Put your SSID & Password*/
const char* ssid = "Ahsan";  // Enter SSID here
const char* password = "12345678";  //Enter Password here


WebServer server(80);

// DHT Sensor
uint8_t DHTPin = 15;

// Initialize DHT sensor.
DHT dht(DHTPin, DHTTYPE);

float Temperature;
float Humidity;

void setup() {
  Serial.begin(115200);
  delay(100);

  pinMode(DHTPin, INPUT);
  dht.begin();

  Serial.println("Connecting to ");
  Serial.println(ssid);

  //connect to your local wi-fi network
  WiFi.begin(ssid, password);

  //check wi-fi is connected to wi-fi network
  while (WiFi.status() != WL_CONNECTED) {
  delay(1000);
  Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected..!");
  Serial.print("Got IP: ");  Serial.println(WiFi.localIP());

  server.on("/", handle_OnConnect);
  server.onNotFound(handle_NotFound);

  server.begin();
  Serial.println("HTTP server started");

}
void loop() {

  server.handleClient();

}

void handle_OnConnect() {

 Temperature = dht.readTemperature(); // Gets the values of the
temperature
  Humidity = dht.readHumidity(); // Gets the values of the humidity
  server.send(200, "text/html", SendHTML(Temperature,Humidity));
}

void handle_NotFound(){
  server.send(404, "text/plain", "Not found");
}

String SendHTML(float Temperaturestat,float Humiditystat){
  String ptr = "<!DOCTYPE html> <html>\n";
  ptr +="<head><meta name=\"viewport\" content=\"width=device-width,
initial-scale=1.0, user-scalable=no\">\n";
  ptr +="<title>ESP32 Weather Report</title>\n";
  ptr +="<style>html { font-family: Helvetica; display: inline-block; margin:
0px auto; text-align: center;}\n";
  ptr +="body{margin-top: 50px;} h1 {color: #444444;margin: 50px auto
30px;}\n";
  ptr +="p {font-size: 24px;color: #444444;margin-bottom: 10px;}\n";
  ptr +="</style>\n";
  ptr +="</head>\n";
  ptr +="<body>\n";
  ptr +="<div id=\"webpage\">\n";
  ptr +="<h1>ESP32 Weather Report</h1>\n";

  ptr +="<p>Temperature: ";
  ptr +=(int)Temperaturestat;
  ptr +="°C</p>";
  ptr +="<p>Humidity: ";
  ptr +=(int)Humiditystat;
  ptr +="%</p>";

  ptr +="</div>\n";
  ptr +="</body>\n";
  ptr +="</html>\n";
  return ptr;
}
```

Step 3 : Select Proper Board

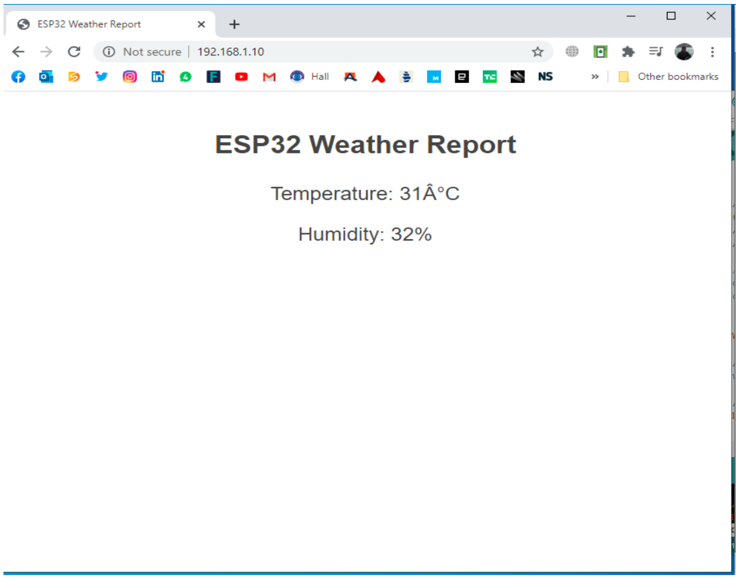


Board: "ESP32 Dev Module"
Upload Speed: "921600"
CPU Frequency: "240MHz (WiFi/BT)"
Flash Frequency: "80MHz"
Flash Mode: "QIO"
Flash Size: "4MB (32Mb)"
Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"
Core Debug Level: "None"
PSRAM: "Disabled"
Programmer: "AVR ISP"

```
  server.begi
  Serial.prin

}
void loop() {

  server.hand

}

void handle_O

 Temperature
  Humidity =
  server.send
}

void handle_N
  server.send
}

String SendHT
  String ptr
  ptr +="<hea
  ptr +="<title>ESP32 Weather Report</title>\n";
  ptr +="<style>html { font-family: Helvetica; display: inline-block; margin: 0px au
  ptr +="body{margin-top: 50px;} h1 {color: #444444;margin: 50px auto 30px;}\n";
  ptr +="p {font-size: 24px;color: #444444;margin-bottom: 10px;}\n";
  ptr +="</style>\n";
  ptr +="</head>\n";
  ptr +="<body>\n";
  ptr +="<div id=\"webpage\">\n";
  ptr +="<h1>ESP32 Weather Report</h1>\n";

  ptr +="<p>Temperature: ";
  ptr +=(int)Temperaturestat;
  ptr +="°C</p>";
```

Step 4: Compile and Run the Program

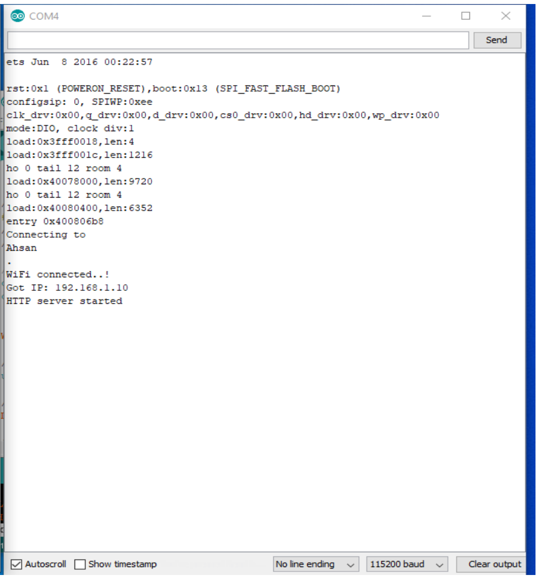
Output File:



ESP32 Weather Report

Temperature: 31Â°C

Humidity: 32%

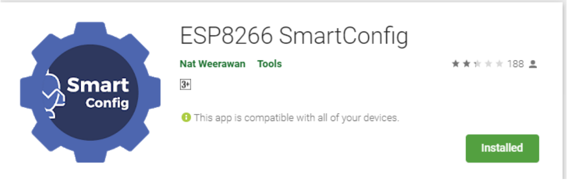


```
ets Jun  8 2016 00:22:57

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0018,len:4
load:0x3fff001c,len:1216
ho 0 tail 12 room 4
load:0x40078000,len:9720
ho 0 tail 12 room 4
load:0x40080400,len:6352
entry 0x400806b8
Connecting to
Ahsan
.
WiFi connected..!
Got IP: 192.168.1.10
HTTP server started
```

Task 2:

Use ESP8266 Smart Config App from Google Playstore to allocate IP to ESP32 by your mobile at Run-time

Step 1: Install the App from Play store



ESP8266 SmartConfig

This app is compatible with all of your devices.

Step 2: Use the following program and burn it in ESP32

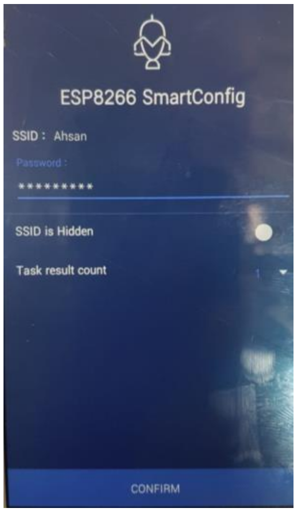
```
#include <WiFi.h>

#include <WebServer.h>



// Use ESP8266 Smart Config App from playstore to allocate ip to ESP32 by your mobile,

// Use Wifi SSID and Pwd, which will assign IP to ESP 32

// No need to define Password and SSID within your code.



WebServer server(80);



uint8_t LED1pin = 4;

bool LED1status = LOW;



uint8_t LED2pin = 5;

bool LED2status = LOW;



void setup() {

  Serial.begin(115200);

   //////////////////////////////////////////

  //Serial.begin(115200);



  //Init WiFi as Station, start SmartConfig
```

```
WiFi.mode(WIFI_AP_STA);
WiFi.beginSmartConfig();

//Wait for SmartConfig packet from mobile
Serial.println("Waiting for SmartConfig.");
while (!WiFi.smartConfigDone()) {
  delay(500);
  Serial.print(".");
}

Serial.println("");
Serial.println("SmartConfig received.");

//Wait for WiFi to connect to AP
Serial.println("Waiting for WiFi");
while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print(".");
}

Serial.println("WiFi Connected.");

Serial.print("IP Address: ");
Serial.println(WiFi.localIP());



//////////////////////////////////////////////
pinMode(LED1pin, OUTPUT);
pinMode(LED2pin, OUTPUT);

//WiFi.softAP(ssid, password);
//WiFi.softAPConfig(local_ip, gateway, subnet);
```

```
  delay(100);


  server.on("/", handle_OnConnect);
  server.on("/led1on", handle_led1on);
  server.on("/led1off", handle_led1off);
  server.on("/led2on", handle_led2on);
  server.on("/led2off", handle_led2off);
  server.onNotFound(handle_NotFound);


  server.begin();
  Serial.println("HTTP server started");


}
void loop() {
  server.handleClient();
  if(LED1status)
  {digitalWrite(LED1pin, HIGH);}
  else
  {digitalWrite(LED1pin, LOW);}


  if(LED2status)
  {digitalWrite(LED2pin, HIGH);}
  else
  {digitalWrite(LED2pin, LOW);}
}


void handle_OnConnect() {
  LED1status = LOW;
  LED2status = LOW;
  Serial.println("GPIO4 Status: OFF | GPIO5 Status: OFF");
  server.send(200, "text/html", SendHTML(LED1status,LED2status));
}
```

```
void handle_led1on() {
  LED1status = HIGH;
  Serial.println("GPIO4 Status: ON");
  server.send(200, "text/html", SendHTML(true,LED2status));
}

void handle_led1off() {
  LED1status = LOW;
  Serial.println("GPIO4 Status: OFF");
  server.send(200, "text/html", SendHTML(false,LED2status));
}

void handle_led2on() {
  LED2status = HIGH;
  Serial.println("GPIO5 Status: ON");
  server.send(200, "text/html", SendHTML(LED1status,true));
}

void handle_led2off() {
  LED2status = LOW;
  Serial.println("GPIO5 Status: OFF");
  server.send(200, "text/html", SendHTML(LED1status,false));
}

void handle_NotFound(){
  server.send(404, "text/plain", "Not found");
}

String SendHTML(uint8_t led1stat,uint8_t led2stat){
  String ptr = "<!DOCTYPE html> <html>\n";
  ptr +="<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1.0, user-scalable=no\">\n";
  ptr +="<title>LED Control</title>\n";

  ptr +="<style>html { font-family: Helvetica; display: inline-block; margin: 0px auto; text-align: center;}\n";

  ptr +="body{margin-top: 50px;} h1 {color: #444444;margin: 50px auto 30px;} h3 {color: #444444;margin-bottom: 50px;}\n";

  ptr +=".button {display: block;width: 80px;background-color: #3498db;border: none;color: white;padding: 13px 30px;text-decoration: none;font-size: 25px;margin: 0px auto 35px;cursor: pointer;border-radius: 4px;}\n";

  ptr +=".button-on {background-color: #3498db;}\n";

  ptr +=".button-on:active {background-color: #2980b9;}\n";

  ptr +=".button-off {background-color: #34495e;}\n";

  ptr +=".button-off:active {background-color: #2c3e50;}\n";

  ptr +="p {font-size: 14px;color: #888;margin-bottom: 10px;}\n";

  ptr +="</style>\n";

  ptr +="</head>\n";

  ptr +="<body>\n";

  ptr +="<h1>ESP32 Web Server</h1>\n";

  ptr +="<h3>Using Access Point(AP) Mode</h3>\n";


   if(led1stat)

  {ptr +="<p>LED1 Status: ON</p><a class=\"button button-off\" href=\"/led1off\">OFF</a>\n";}

  else

  {ptr +="<p>LED1 Status: OFF</p><a class=\"button button-on\" href=\"/led1on\">ON</a>\n";}


  if(led2stat)

  {ptr +="<p>LED2 Status: ON</p><a class=\"button button-off\" href=\"/led2off\">OFF</a>\n";}

  else

  {ptr +="<p>LED2 Status: OFF</p><a class=\"button button-on\" href=\"/led2on\">ON</a>\n";}


  ptr +="</body>\n";

  ptr +="</html>\n";

  return ptr;

}
```

Step 3: Open App and Enter Password against your Wifi Connection and enter CONFIRM



ESP8266 SmartConfig

SSID : Ahsan

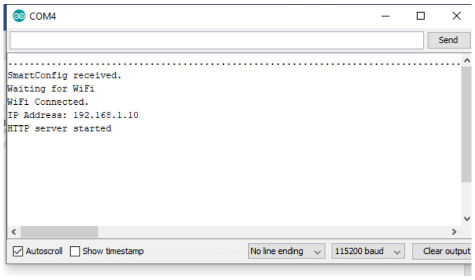
Password :

****

SSID is Hidden

Task result count 1

Step 4: Check Arduino Serial Output, it will show the Assigned IP from Wifi



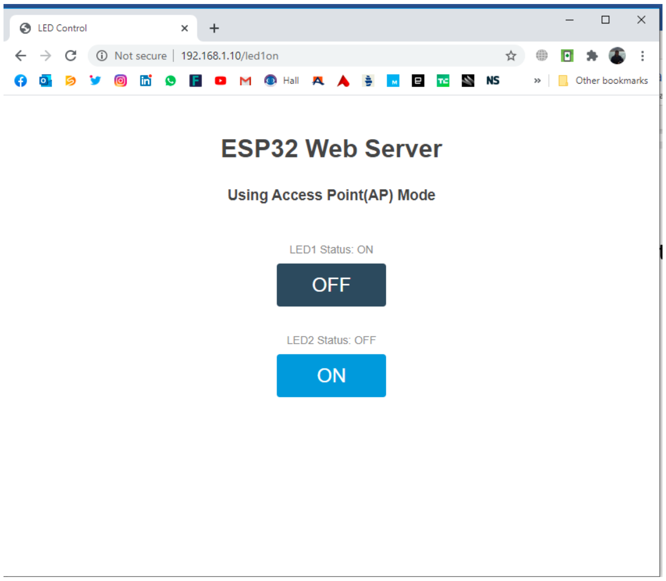
```
.................................................................................
SmartConfig received.
Waiting for WiFi
WiFi Connected.
IP Address: 192.168.1.10
HTTP server started
```

☑ Autoscroll □ Show timestamp

No line ending

115200 baud

Step 5: Enter the IP into your Web Browser and control the output LEDs conneted



192.168.1.10/led1on

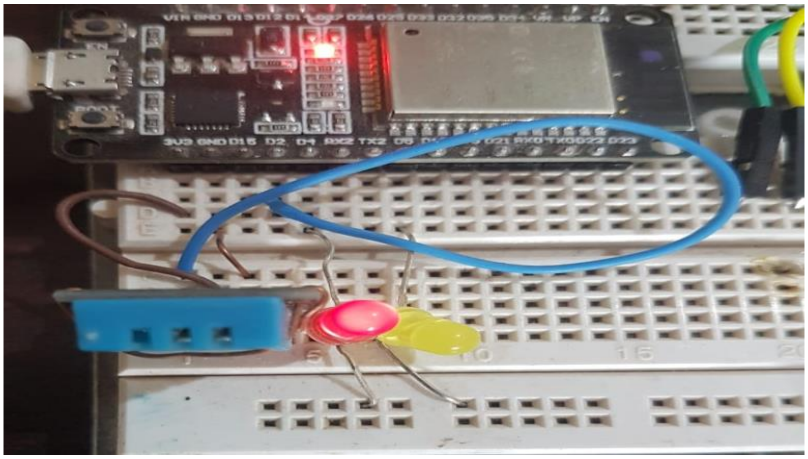
ESP32 Web Server

Using Access Point(AP) Mode

LED1 Status: ON

LED2 Status: OFF

Step 6: Check your Hardware

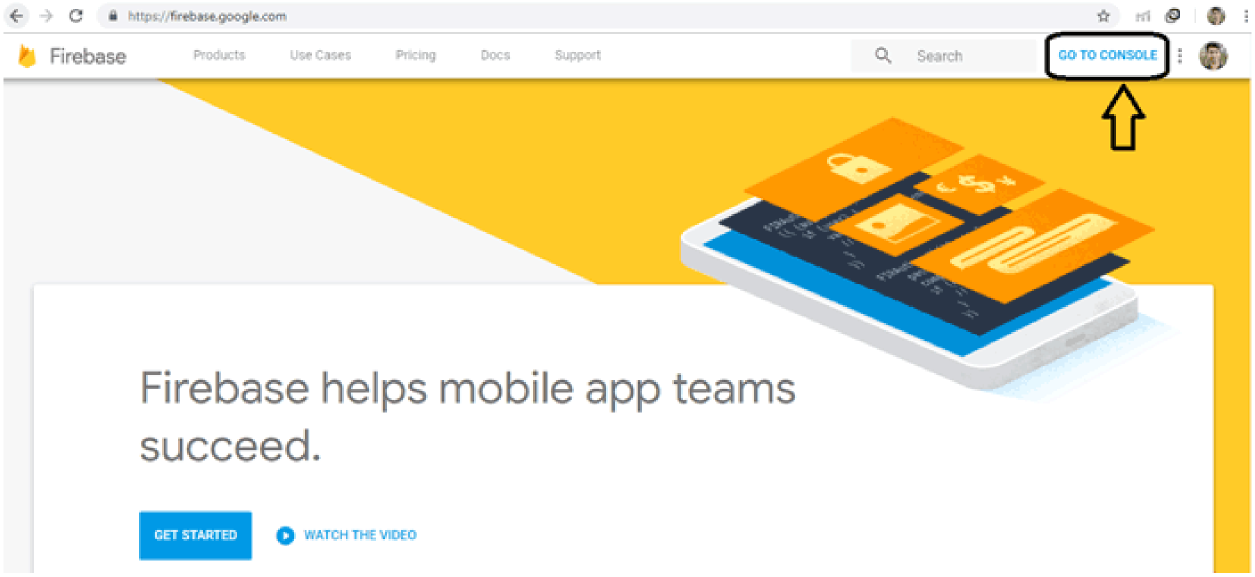


Task 3: LED control using Google Firebase and ESP32

Step 1: Setting up Firebase Console for ESP32

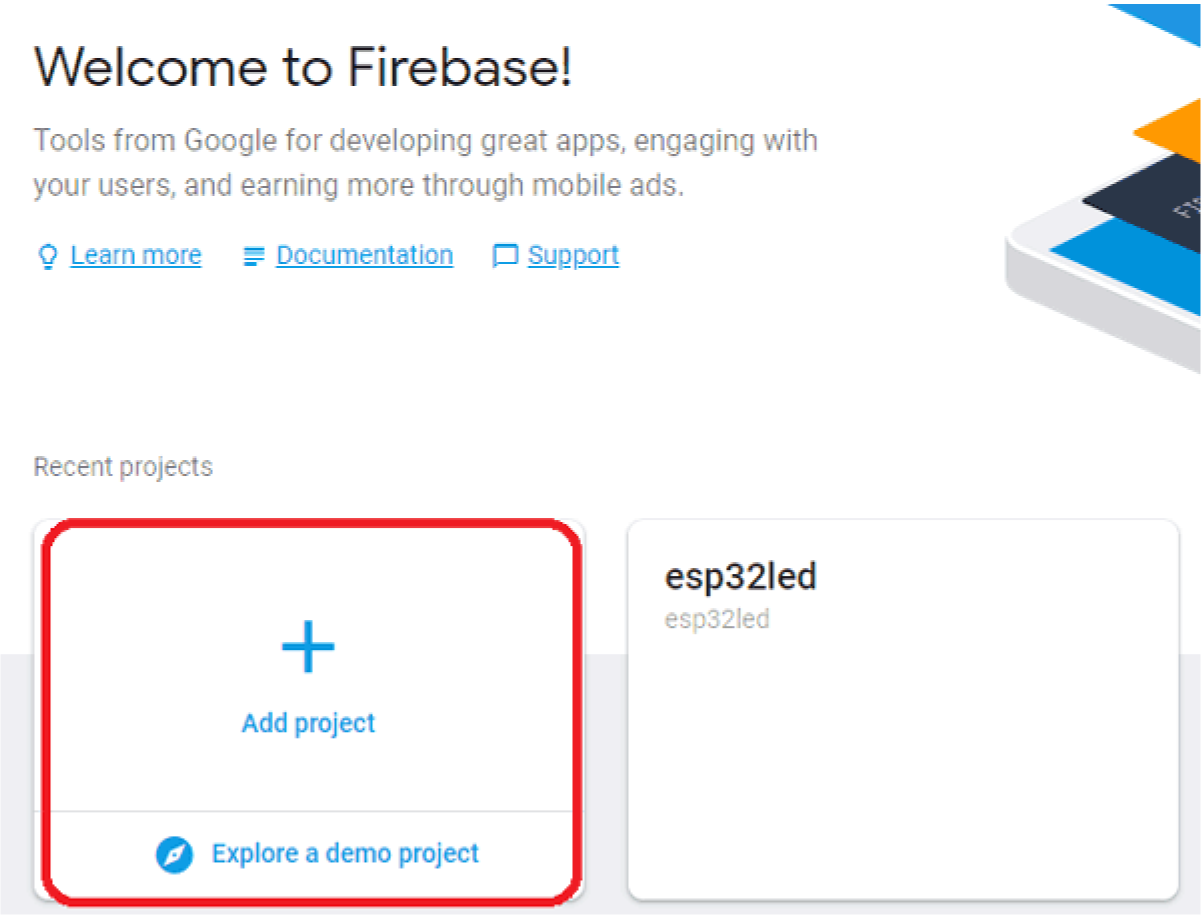
If you are using firebase for the first time then you have to create account for firebase or can directly signup using Google Account:

1. Open your browser and go for https://firebase.google.com
2. At the right top corner click on **"Go to Console".**



Firebase helps mobile app teams succeed.

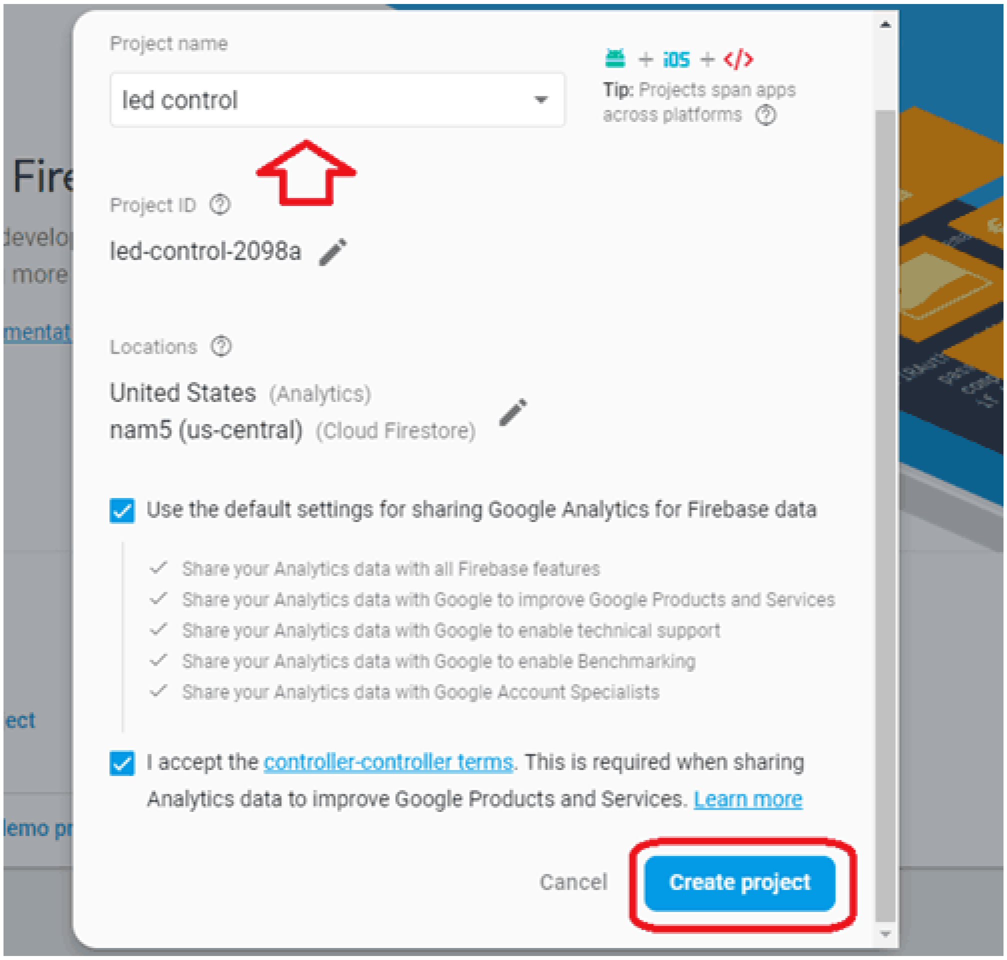
3. Click on "**Add Project**".



Welcome to Firebase!

Tools from Google for developing great apps, engaging with your users, and earning more through mobile ads.

4. Input your project name as you want and click on create project.



Project name

led control

Tip: Projects span apps across platforms

Project ID

led-control-2098a

Locations

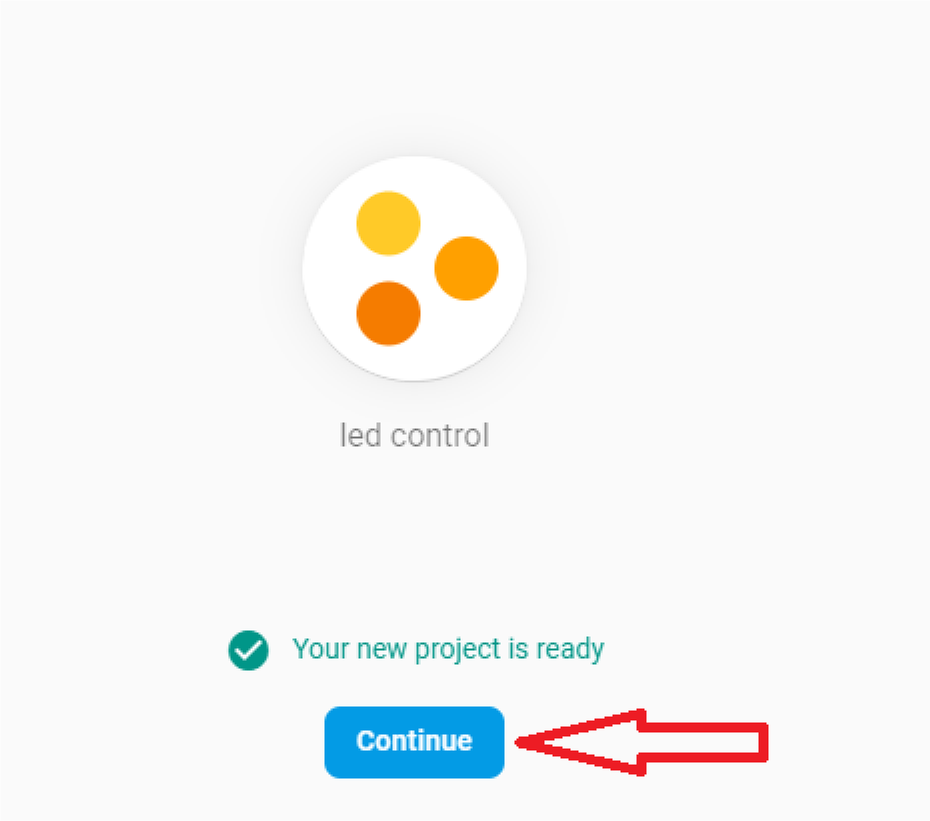
United States (Analytics)
nam5 (us-central) (Cloud Firestore)

☑ Use the default settings for sharing Google Analytics for Firebase data

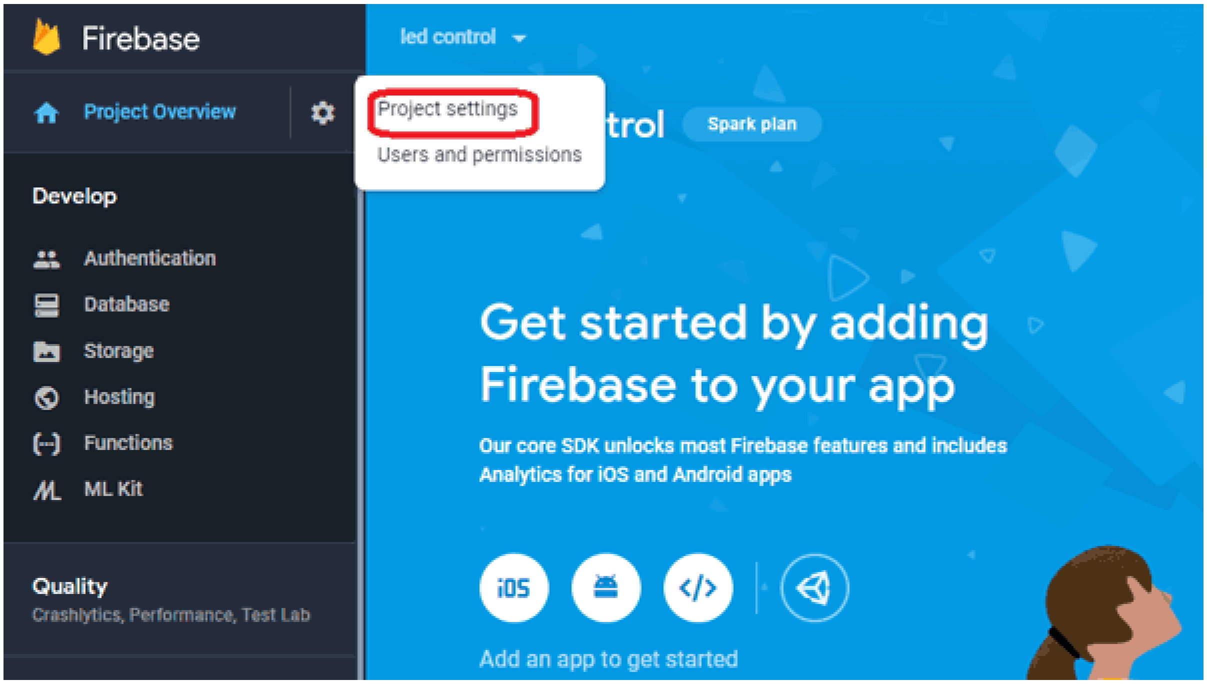
- ✓ Share your Analytics data with all Firebase features
- ✓ Share your Analytics data with Google to improve Google Products and Services
- ✓ Share your Analytics data with Google to enable technical support
- ✓ Share your Analytics data with Google to enable Benchmarking
- ✓ Share your Analytics data with Google Account Specialists

☑ I accept the controller-controller terms. This is required when sharing Analytics data to improve Google Products and Services. Learn more

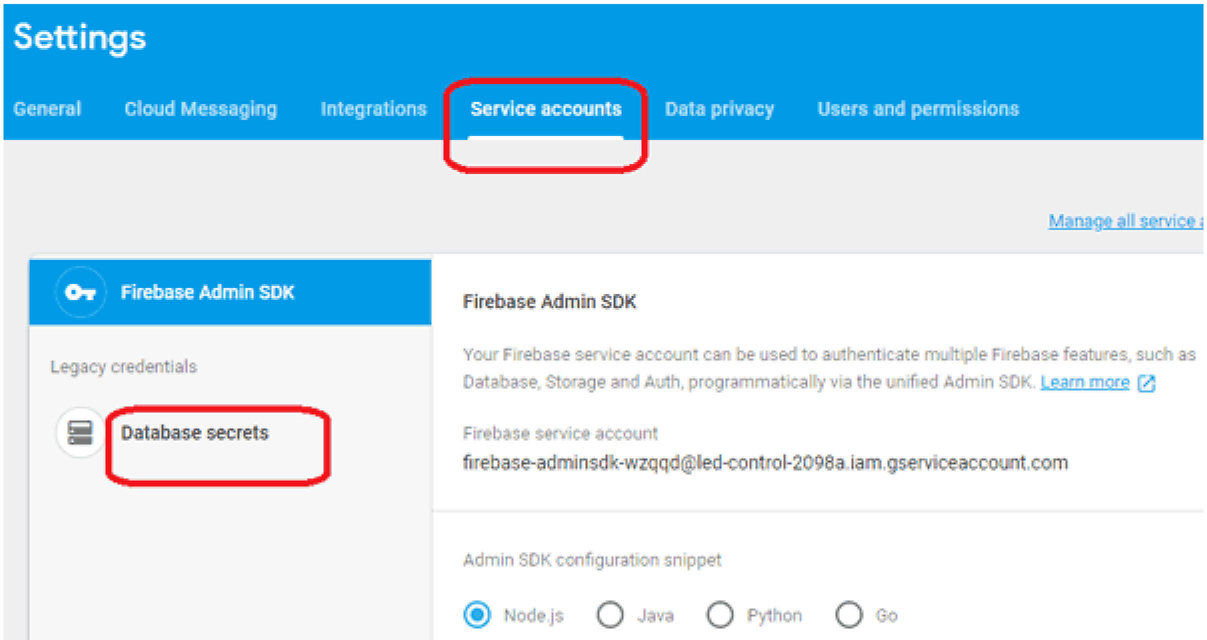
5. Now your project is created and click on “**Continue**”.



6. Now you will need host name and authorization key/secret key for this project while programming your ESP32; so now we will see how these parameters can be taken from this.
7. Go to setting icon and click on "**Project Setting**".


Firebase
led control
Project Overview
Project settings
Users and permissions
trol
Spark plan
Develop
Authentication
Database
Storage
Hosting
Functions
ML Kit
Quality
Crashlytics, Performance, Test Lab
Get started by adding Firebase to your app
Our core SDK unlocks most Firebase features and includes Analytics for iOS and Android apps
iOS
</>
Add an app to get started

8. Now click on **Service accounts** and then **Database secrets**.



Firebase Admin SDK

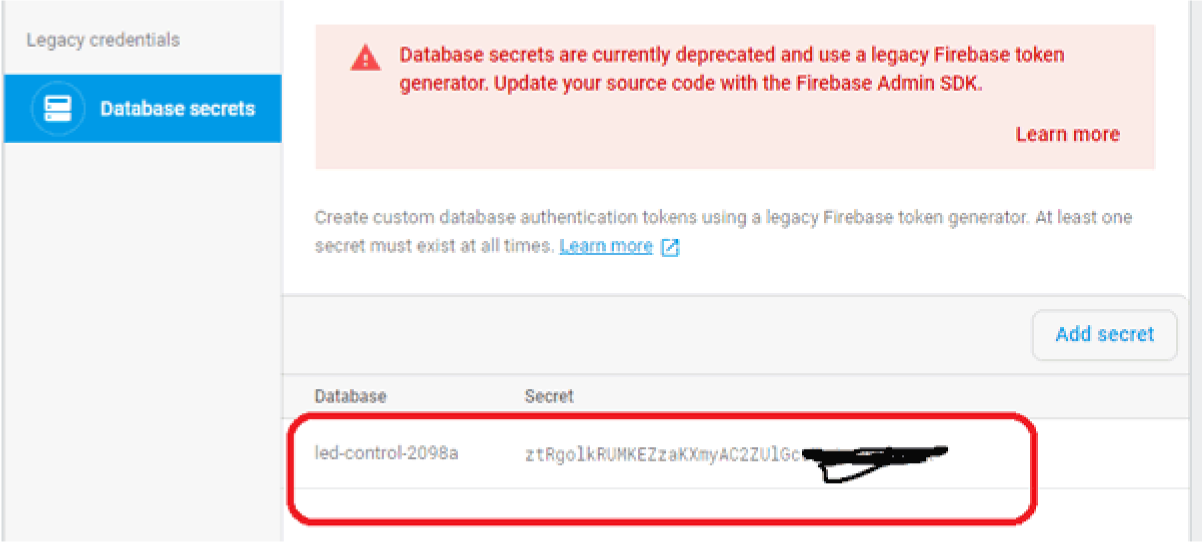
Your Firebase service account can be used to authenticate multiple Firebase features, such as Database, Storage and Auth, programmatically via the unified Admin SDK. Learn more

Firebase service account
firebase-adminsdk-wzqqd@led-control-2098a.iam.gserviceaccount.com

Admin SDK configuration snippet

● Node.js ○ Java ○ Python ○ Go

9. On clicking on *Database Secrets* you will find a secret key, copy this key and save it in notepad, this is your firebase authorization key which you will need later.



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

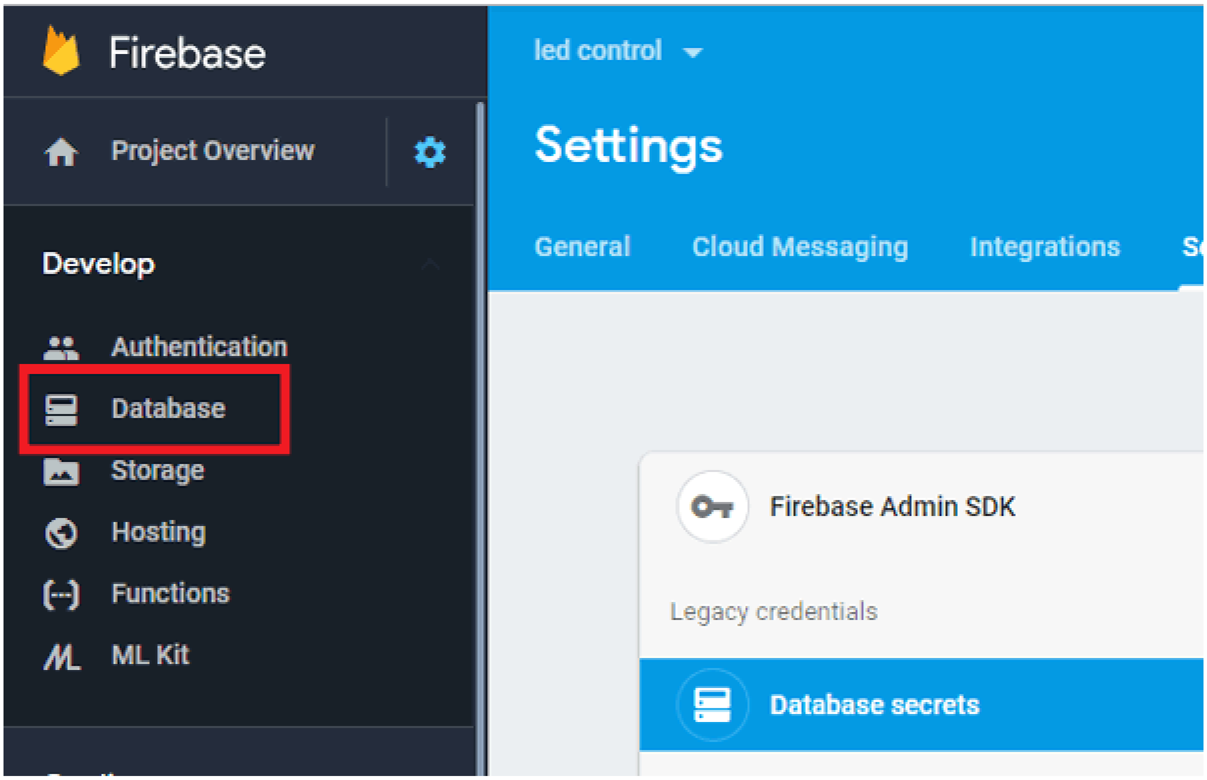
Learn more

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

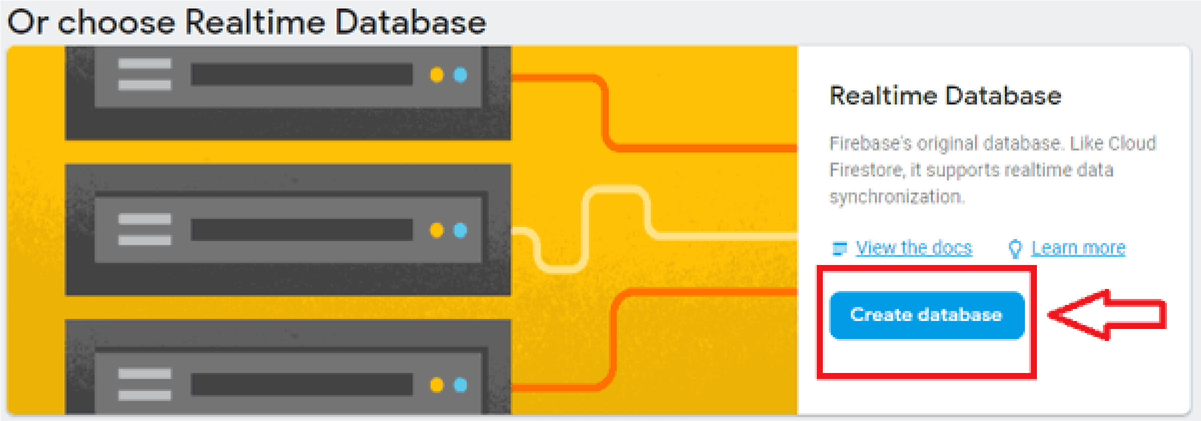
Add secret

Database	Secret
led-control-2098a	ztRgolkRUMKEZzaKXmyAC2ZUlGc[redacted]

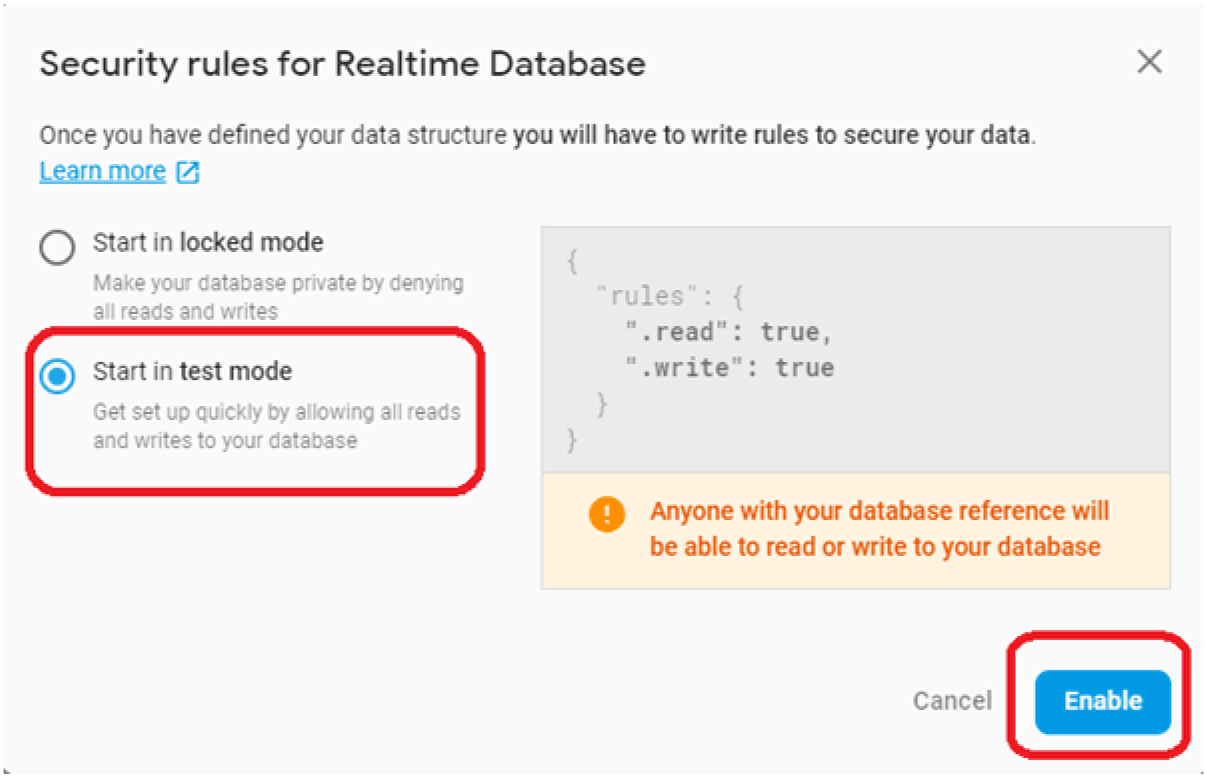
10. Now click on "**Database**" at left control bar.



11. Now scroll down and click on **"Create database".**


Or choose Realtime Database
Realtime Database
Firebase's original database. Like Cloud Firestore, it supports realtime data synchronization.
View the docs
Learn more
Create database

12. Now choose "**Start in test mode**" and click on **Enable**.



Security rules for Realtime Database

Once you have defined your data structure you will have to write rules to secure your data. Learn more

○ Start in locked mode

Make your database private by denying all reads and writes

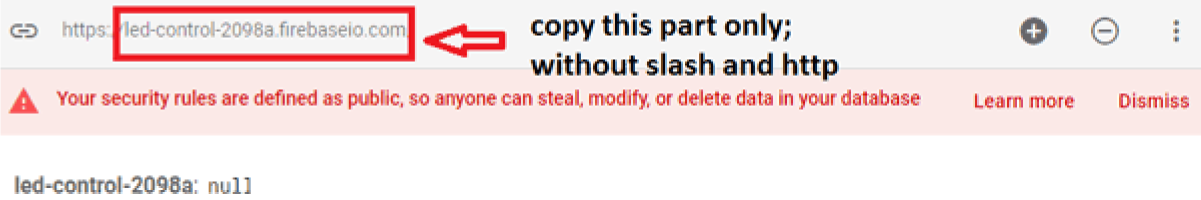
● Start in test mode

Get set up quickly by allowing all reads and writes to your database

```
{
  "rules": {
    ".read": true,
    ".write": true
  }
}
```

Anyone with your database reference will be able to read or write to your database

13. Now your database is created and you will have to come here again to control your LED, for now just copy the given URL without slash and http in notepad this is your **firebase host** which you will be required later.



https://led-control-2098a.firebaseio.com

copy this part only; without slash and http

Your security rules are defined as public, so anyone can steal, modify, or delete data in your database

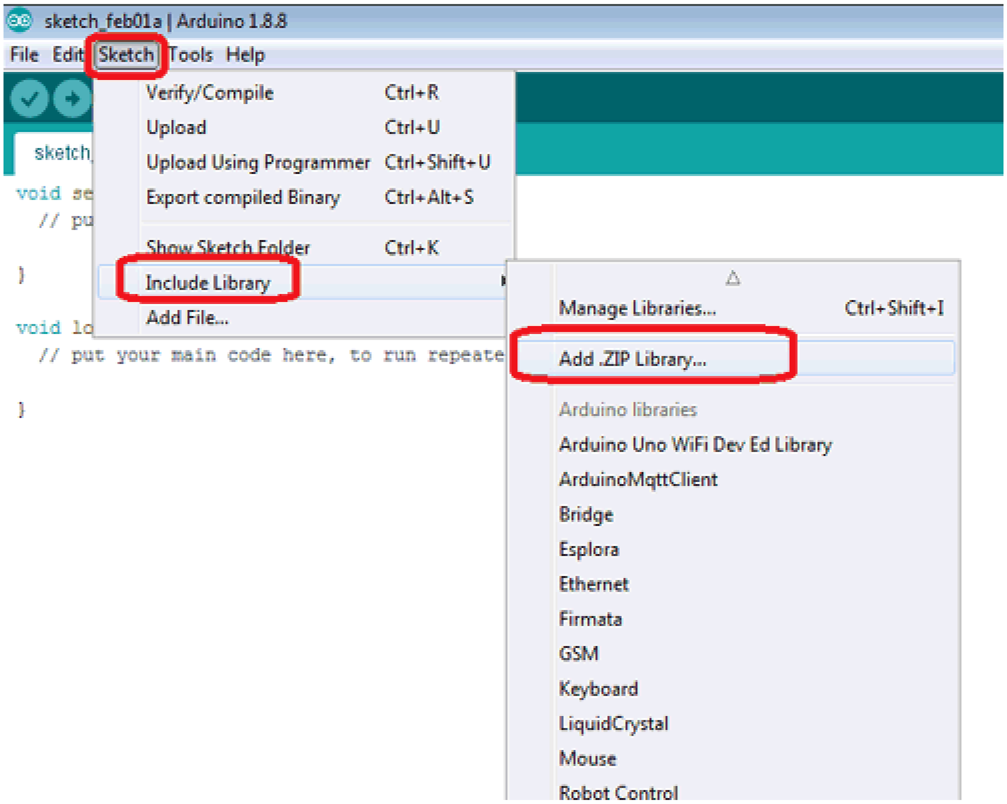
led-control-2098a: null

Step: Setting up ESP32 module with Firebase

To work with **ESP32 using Google firebase** you will need a firebase library so firstly download that library using below link and save it in Arduino library files.

https://github.com/ioxhop/IOXhop_FirebaseESP32

Now open your Arduino IDE and go to *Sketch---> include Library--> Add .ZIP library* and add the file you downloaded from the above link.


sketch_feb01a | Arduino 1.8.8
File Edit Sketch Tools Help
sketch
Verify/Compile Ctrl+R
Upload Ctrl+U
Upload Using Programmer Ctrl+Shift+U
Export compiled Binary Ctrl+Alt+S
Show Sketch Folder Ctrl+K
Include Library
Add File...
Manage Libraries... Ctrl+Shift+I
Add .ZIP Library...
Arduino libraries
Arduino Uno WiFi Dev Ed Library
ArduinoMqttClient
Bridge
Esplora
Ethernet
Firmata
GSM
Keyboard
LiquidCrystal
Mouse
Robot Control

```
void se
  // pu

}

void lo
  // put your main code here, to run repeate

}
```

After installing the library you are ready to work with Google firebase using ESP32.

Now go to Tools and select ESP32 Dev board and appropriate COM port and copy the code given below and edit it for network credentials, firebase secret key and firebase host. After editing the upload the code into ESP32 using Arduino IDE.

Step 3: Programming ESP32 for Google Firebase

Complete code for controlling LED using ESP32 is given at the end of the tutorial

Firstly include the libraries for using ESP32 and firebase.

```
#include <WiFi.h>

#include <IOXhop_FirebaseESP32.h>
```

Now enter your firebase host, secret key and network credentials.

```
#define FIREBASE_HOST "esp32led.firebaseio.com"

#define FIREBASE_AUTH "4QHdeFZquTh4fdXZkum2EPt2A50gXXXXXXXXX"

#define WIFI_SSID "XXXXXX"
```

```
#define WIFI_PASSWORD "XXXXXXXXXX"
```

In the *setup* function, define output pin, delay, baud rate and connect to your Wi-Fi.

```
void setup() {
  Serial.begin(9600);
  delay(1000);
  pinMode(2, OUTPUT);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to ");
  Serial.print(WIFI_SSID);
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }
```

The below statement tries to connect with the firebase server. If the host address and authorization key are correct then it will connect successfully.

```
Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
```

Now this is the class provided by firebase library to send string to firebase server. With the help of this we can change the status of LED.

```
Firebase.setString("LED_STATUS", "OFF");
```

After sending one status string to firebase path, write this statement to get the status of LED from same path and save it to variable.

```
fireStatus = Firebase.getString("LED_STATUS");
```

If received string is “ON” or “on” then just turn on the output LED.

```
if (fireStatus == "ON")
```

```
 {
Serial.println("Led Turned ON");
digitalWrite(2, HIGH);
 }
```

If received string is “OFF” or “off” then just turn off the output LED.

```
else if (fireStatus == "OFF") {
 Serial.println("Led Turned OFF");
 digitalWrite(2, LOW);
}
```

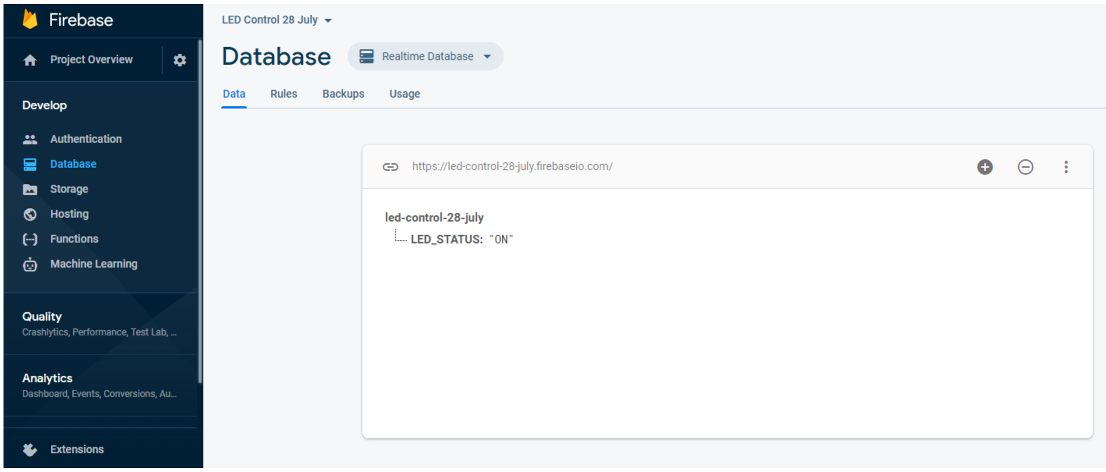
If received string is not any of these then just ignore and print some error message.

```
else {
Serial.println("Wrong Credential! Please send ON/OFF");
  }
```

The complete code is given at the end of this article, you can check from there, edit it and then upload it.

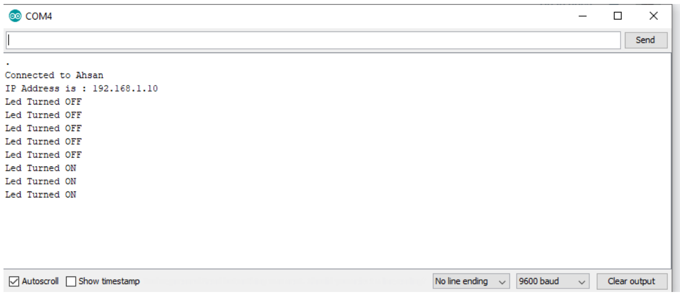
Step 4: Check your Output

After uploading the code open your serial monitor and in your browser open firebase and go to console and then select your project that you created earlier. In this go to database and it will show initially LED STATUS is OFF, from here you can change the LED status by writing ON here, your LED will change to ON state and you can see your LED state in your serial monitor also.



https://led-control-28-july.firebaseio.com/

```
led-control-28-july
└ LED_STATUS: "ON"
```



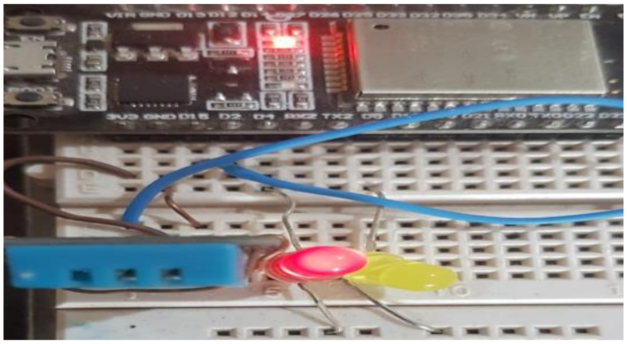
```
.
Connected to Ahsan
IP Address is : 192.168.1.10
Led Turned OFF
Led Turned OFF
Led Turned OFF
Led Turned OFF
Led Turned OFF
Led Turned ON
Led Turned ON
Led Turned ON
```

Step 5: Code

```
#include <WiFi.h>                                   // esp32 library


#include <IOXhop_FirebaseESP32.h>                             // firebase library


#define FIREBASE_HOST "led-control-28-july.firebaseio.com/"                // the project name address from
//firebase id


#define FIREBASE_AUTH "J6X1bbwAWGmmIxChCRnygFlJw6ReW63GGEK44K1g"           // the secret
//key generated from firebase


#define WIFI_SSID "Ahsan"                             // input your home or public wifi name


#define WIFI_PASSWORD "12345678"                            //password of wifi ssid


String fireStatus = "";                             // led status received from firebase


int led = 4;
```

```
void setup() {
  Serial.begin(9600);
  delay(1000);
  pinMode(4, OUTPUT);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);                          //try to connect with wifi
  Serial.print("Connecting to ");
  Serial.print(WIFI_SSID);
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }
  Serial.println();
  Serial.print("Connected to ");
  Serial.println(WIFI_SSID);
  Serial.print("IP Address is : ");
  Serial.println(WiFi.localIP());                                //print local IP address
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);                  // connect to firebase
  Firebase.setString("LED_STATUS", "OFF");                    //send initial string of led status


}


void loop() {


  fireStatus = Firebase.getString("LED_STATUS");          // get led status input from firebase


  if (fireStatus == "ON") {                 // compare the input of led status received from firebase


    Serial.println("Led Turned ON");


    digitalWrite(4, HIGH);                                // make output led ON


  }


  else if (fireStatus == "OFF") {        // compare the input of led status received from firebase


    Serial.println("Led Turned OFF");
    digitalWrite(4, LOW);                                 // make output led OFF
  }
  else {
    Serial.println("Wrong Credential! Please send ON/OFF");


  }
}
```

Step 6: Hardware Demo

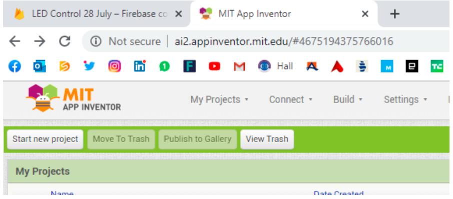


Task 4: LED control using Google Firebase, ESP32 and Mobile App

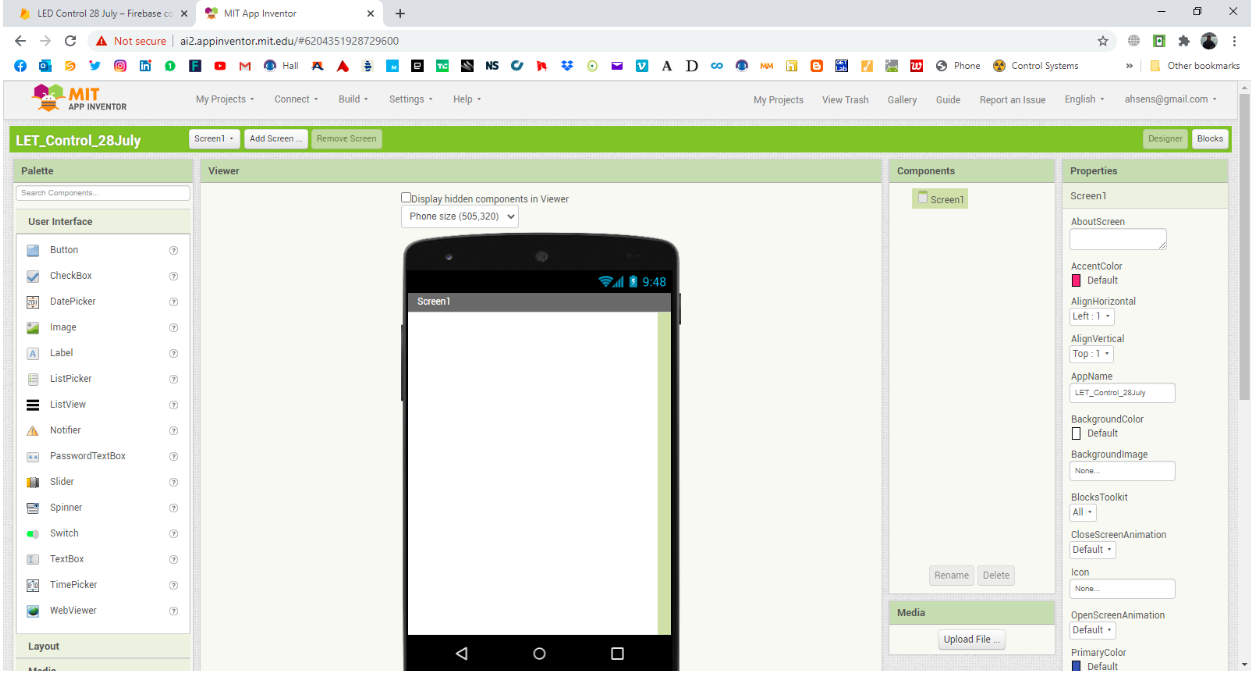
Step 1: Create App on MIT App Inventor

Goto link: http://ai2.appinventor.mit.edu/

1. **Create your account.**
2. **After Login start new project:**

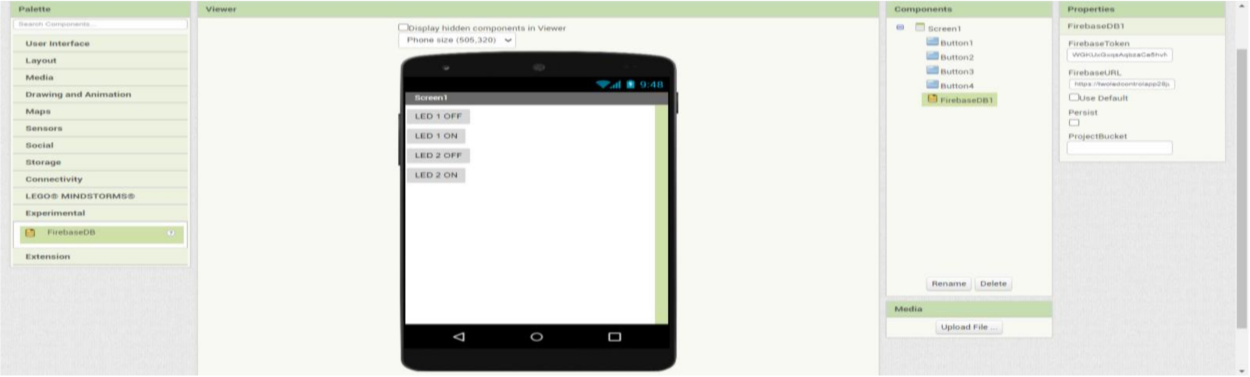


3. **Following screen appears**



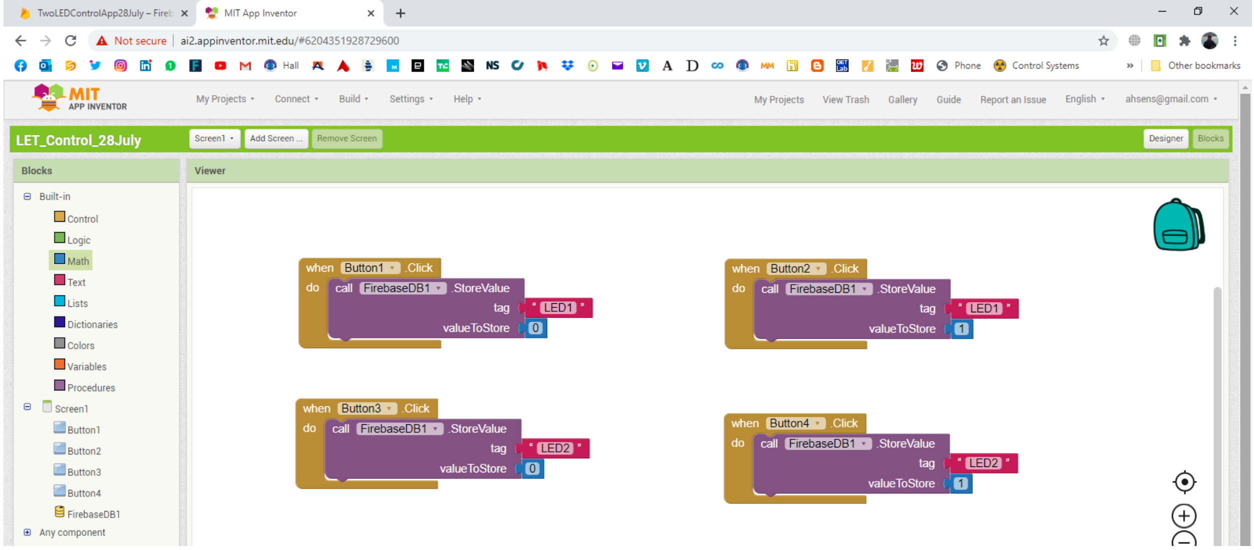
4. **Drag and drop four buttons and one firebase link as follows:**

Project Bucket on right side must be empty.



5: Enter firebase database link in Firebase URL and database secret key in in Firebase token

6: Goto Blocks and do following



when Button1.Click do call FirebaseDB1.StoreValue tag "LED1" valueToStore 0

when Button2.Click do call FirebaseDB1.StoreValue tag "LED1" valueToStore 1

when Button3.Click do call FirebaseDB1.StoreValue tag "LED2" valueToStore 0

when Button4.Click do call FirebaseDB1.StoreValue tag "LED2" valueToStore 1

7: Use Build to generate .apk file for your mobile and install this in your android mobile.

Or install MIT AI2 Companion app in your mobile and use Connect to see and simulate app in real time using your mobile.

Step 2: Create real-time database using steps mentioned in task above.

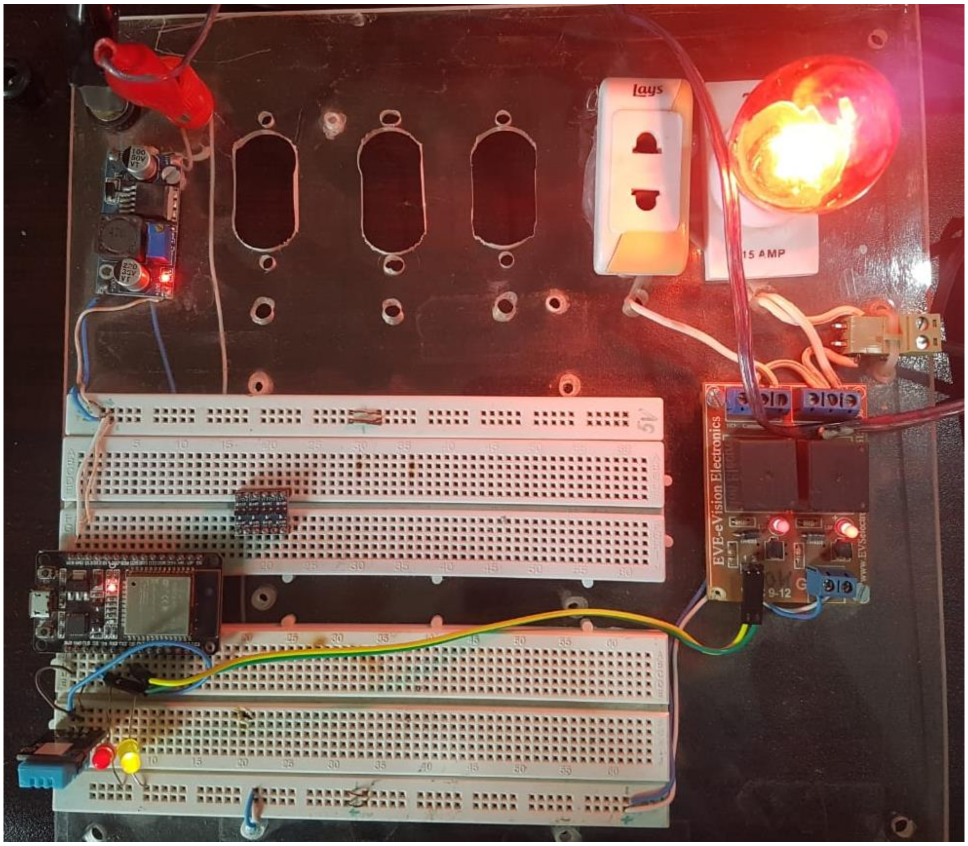
Step 3: Use following code and burn inside your ESP32 Controller.

```
#include <WiFi.h>                                    // esp32 library
#include <IOXhop_FirebaseESP32.h>                              // firebase library
#define FIREBASE_HOST "twoledcontrolapp28july.firebaseio.com/"              //Your
Firebase //Project URL goes here without "http:" , "\" and "/"
#define FIREBASE_AUTH "WGKUxGxqsAqbzaCe5hvNG9qC8gSekvvTLmFVFQPe"
//Your Firebase Database Secret goes here
```

```
#define WIFI_SSID "Ahsan"                    //your WiFi SSID for which yout NodeMCU
connects
#define WIFI_PASSWORD "12345678"                        //Password of your wifi network

String fireStatus = "";                                          // led status received from firebase

String fireStatus2 = "";

const int led1 = 4;
const int led2=5;

void setup() {

  Serial.begin(9600);

  delay(1000);

  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);

  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);                                   //try to connect with
wifi

  Serial.print("Connecting to ");

  Serial.print(WIFI_SSID);

  while (WiFi.status() != WL_CONNECTED) {

    Serial.print(".");

    delay(500);

  }

  Serial.println();

  Serial.print("Connected to ");
  Serial.println(WIFI_SSID);

  Serial.print("IP Address is : ");

  Serial.println(WiFi.localIP());                                  //print local IP address

  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);                           // connect to
firebase

  Firebase.setString("LED1", "0");                             //send initial string of led status
  Firebase.setString("LED2", "0");
}

void loop() {

  fireStatus = Firebase.getString("LED1");              // get led status input from firebase

  if (fireStatus == "1") {                  // compare the input of led status received from
firebase

    Serial.println("Led1 Turned ON");

    digitalWrite(4, HIGH);                                         // make output led ON

  }

  else if (fireStatus == "0") {            // compare the input of led status received from
firebase

    Serial.println("Led1 Turned OFF");

    digitalWrite(4, LOW);                                          // make output led OFF

  }

  else {
    Serial.println("Wrong Credential for LED1! Please send ON/OFF");

  }

  fireStatus2 = Firebase.getString("LED2");
  if (fireStatus2 == "1") {                    // compare the input of led status received from
 firebase

    Serial.println("Led2 Turned ON");

    digitalWrite(5, HIGH);                                   // make output led ON

  }

  else if (fireStatus2 == "0") {            // compare the input of led status received from
 firebase

    Serial.println("Led2 Turned OFF");

    digitalWrite(5, LOW);                                    // make output led OFF

  }

  else {

    Serial.println("Wrong Credential for LED2! Please send ON/OFF");

  }
}
```

Step 4: Control the LEDs using your Mobile App in real time.

Task 5: Control of Home appliances using MIT App and Firebase using ESP 32
Simple relay circuit to switch ON/OFF Bulb
Output:



Post-Lab Task :

Critical Analysis / Conclusion

(By Student about Learning from the Lab)

Lab Assessment				
Pre Lab			**/1**	**/10**
In Lab			**/5**	
Post Lab	**Data Analysis**	**/4**	**/4**	
	Data Presentation	**/4**		
	Writing Style	**/4**		
Instructor Signature and Comments				