



TUGAS PROYEK AKHIR IOT

SMART HOME ASSISTANT

Kelompok 14

5027201019 - Muhammad Alwan

5027201042 - Ilham Muhammad Sakti

5027201061 - Gennaro Fajar Mennde

Dosen Pengampu

Ir. Muchammad Husni, M.Kom

DEPARTEMEN TEKNOLOGI INFORMASI

Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

Surabaya 2022

DAFTAR ISI

Daftar isi	2
Smart home assistant	3
Komponen smart home assistant	3
Rangkaian smart home assistant	9
Dokumentasi	10
Lampiran	22

I. Smart home assistant

Smart home merupakan teknologi yang mengacu pada pengaturan rumah yang nyaman di mana peralatan dan perangkat dapat dikontrol secara otomatis dari jarak jauh atau dari mana saja dengan koneksi internet. Sistem ini menggunakan perangkat seluler atau perangkat jaringan lainnya. Teknologi ini memungkinkan pengguna untuk mengontrol fungsi seperti akses keamanan ke rumah, pengaturan pencahayaan, home theater dari jarak jauh, dan masih banyak yang lainnya. Sistem ini bertujuan agar pemilik rumah menjadi lebih mudah dan nyaman mengatur rumahnya dan juga membantu penghematan biaya.

Pada tugas proyek akhir mata kuliah internet of things kami mengangkat judul, yaitu **“Smart Home Assistant”**. Kami membuat simulasi terkait *smart home* menggunakan alat trainer IOT kit ESP32 serta komponen tambahan yang lainnya. Sistem yang kami buat dapat menjalankan tugas seperti menghidupkan dan mematikan lampu 1, 2, 3 serta menghidupkan dan mematikan ketiga lampu tersebut secara bersamaan baik menggunakan aplikasi via mobile, web, serta melalui bantuan google assistant. Selain mengoperasikan lampu sistem kami dapat menghidupkan buzzer atau alarm rumah, menghidupkan dan mematikan tv, serta pintu otomatis yang akan terbuka ketika terdapat orang yang akan masuk ke rumah lewat pintu tersebut.

II. Komponen smart home assistant

Berikut komponen yang kami gunakan pada system **Smart Home Assistant**.

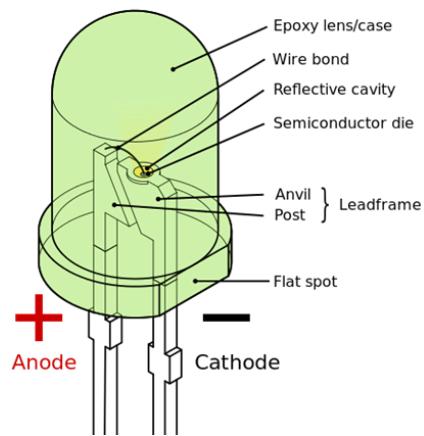
1. ESP32

ESP32 adalah mikrokontroler berharga rendah dan hemat energi dengan wifi dan dual-mode bluetooth terintegrasi. Generasi ESP32 menggunakan mikroprosesor Tensilica Xtensa LX6 sebagai inti. Baik dalam mode single-core maupun dual-core. ESP32 dibuat oleh Espressif Systems, perusahaan berbasis di Shanghai, Tiongkok. Pada sistem kami ESP32 bertindak sebagai kontroler dan penghubung antara komponen aktuator dengan komponen Adafruit.



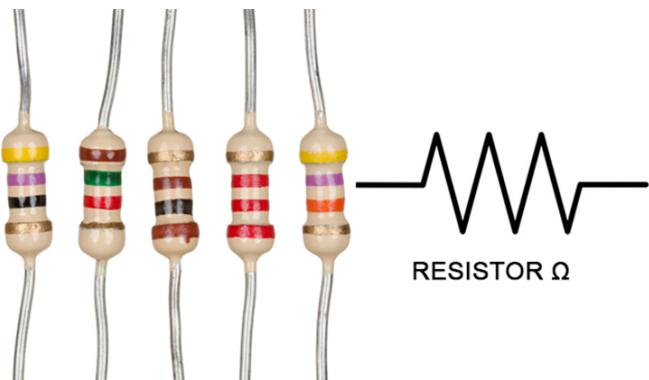
2. Led

LED merupakan kependekan dari Light Emitting Diode, yakni salah satu dari banyak jenis perangkat semikonduktor yang mengeluarkan cahaya ketika arus listrik melewatiinya. Pada sistem kami LED diibaratkan sebagai lampu-lampu yang ada di rumah. Jadi, jika LED dihidupkan berarti lampu yang ada di rumah ini dihidupkan dan begitu sebaliknya.



3. Resistor

Resistor merupakan komponen elektronik yang memiliki dua pin dan didesain untuk mengatur tegangan listrik dan arus listrik. Resistor pada sistem kami berfungsi sebagai pengaman LED. Jadi, apabila beban berlebih LED tidak akan langsung rusak.



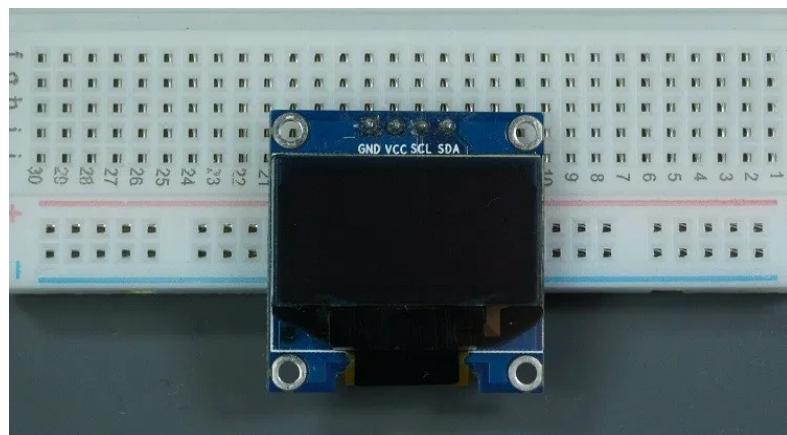
4. Buzzer

Buzzer adalah komponen elektronika yang dapat menghasilkan getaran suara dalam bentuk gelombang bunyi. Buzzer lebih sering digunakan karena ukuran penggunaan dayanya yang minim. Pada sistem kami buzzer berfungsi sebagai bel pintu yang ada di rumah.



5. OLED

OLED adalah singkatan dari Organic Light-Emitting Diode atau organic LED. Teknologi ini digunakan untuk menghasilkan cahaya atau gambar yang baik dan berkualitas tinggi. Dalam hal ini, layar OLED harus menggerakkan arus listrik melalui bahan electroluminescent organik. OLED yang kami gunakan adalah tipe 128x64-I2C. OLED pada sistem kami diibaratkan sebagai televisi yang dapat dihidupkan otomatis ke layanan ***Smart Home Assistant***.



6. Sensor Ultrasonic

Sensor ultrasonic yang berfungsi untuk merubah besaran fisis (suara) menjadi besaran listrik maupun sebaliknya yang dikonversi menjadi jarak. Sensor yang kami gunakan pada sistem ini adalah sensor ultrasonic HC-SR04 yang mempunyai gelombang ultrasonik yaitu gelombang bunyi yang mempunyai frekuensi sangat tinggi yaitu 20.000 Hz. Bunyi ultrasonik tidak dapat didengar oleh telinga manusia. Bunyi ultrasonik dapat didengar oleh anjing, kucing, kelelawar, dan lumba-lumba. Bunyi ultrasonik bisa merambat melalui zat padat, cair dan gas. Reflektivitas bunyi ultrasonik di permukaan zat padat hampir sama dengan reflektivitas bunyi ultrasonik di permukaan zat cair. Akan tetapi, gelombang bunyi ultrasonik akan diserap oleh tekstil dan busa. Pada sistem kami, sensor ini berfungsi untuk mendeteksi objek yang ada di depan pintu rumah. Jika objek di depan pintu rumah maka, pintu akan terbuka otomatis dan begitu pula sebaliknya.



7. Motor Servo

Motor servo adalah komponen elektronika yang berupa motor yang memiliki sistem feedback guna memberikan informasi posisi putaran motor aktual yang diteruskan pada rangkaian kontrol mikrokontroler. Pada dasarnya motor servo banyak digunakan sebagai aktuator yang membutuhkan posisi putaran motor yang presisi. Komponen ini pada sistem ini berfungsi sebagai penggerak pintu atau pintu akan dibuka ketika terdapat objek di depan pintu dan begitu sebaliknya.



8. Kabel Jumper

Kabel jumper adalah kawat listrik, atau kelompoknya dalam kabel, dengan konektor atau pin di setiap ujungnya, yang biasanya digunakan untuk menghubungkan komponen papan tempat memotong roti atau prototipe lain atau rangkaian uji, secara internal atau dengan peralatan atau komponen lain tanpa solder. Pada sistem kami kabel jumper berfungsi sebagai penghubung komponen hardware yang ada.



9. Adafruit IO

Adafruit IO adalah platform yang dirancang untuk menampilkan, merespons, dan berinteraksi dengan data proyek terkait *internet of things*. Pada sistem kami, Adafruit IO bertindak sebagai kontrol aktuator secara online baik melalui web ataupun mobile apps. Jadi, dapat dikatakan IFTTT disini sebagai *IoT middleware* karena bertugas sebagai *interface* antar komponen IoT untuk saling berkomunikasi satu sama lain.

The screenshot shows a dark-themed Adafruit IO dashboard. At the top, there's a navigation bar with links for Shop, Learn, Blog, Forums, LIVE!, AdaBox, IO, Account, and a shopping cart icon showing '0'. Below the navigation is a header with the user name 'lilhamssakti27', the path 'Dashboards > lampu2', and a settings gear icon. The main area contains a 4x3 grid of controls. The first column has three rows: 'allLamp' (top), 'LED 2' (middle), and 'LED 3' (bottom). The second column has two rows: 'BUZZER' (top) and 'L4' (bottom). The third column has three rows: 'L6' (top), 'L7' (middle), and 'LED 4' (bottom). There is also a single row for 'L5' and 'OLED' which only have one control each. Each control is represented by a red circular button with a small number '0' to its right. The entire interface is designed for managing various IoT components through a web-based interface.

10. IFTTT

IFTTT berfungsi sebagai penghubung dan penerjemah ke adafruit sehingga kita dapat mengontrol komponen yang ada di *smart home* melalui google assistant.

Start connecting your world

Save time and money by making the internet work for you! We believe you might like...

Log notes in a Google Drive spreadsheet

•: Google

• 14.2k

Create an event on your iPhone's Calendar with Google Home

•: Google

• 53.8k

Say "OK Google, add __ by __" to add a song to a Spotify playlist

by thomas_s87

• 7.2k

•: 14

11. Google Assistant

Google Assistant adalah asisten virtual yang didukung oleh kecerdasan buatan dan dikembangkan oleh Google yang terutama tersedia di perangkat seluler dan perangkat rumah pintar. Tidak seperti Google Now, Google Assistant dapat terlibat dalam percakapan dua arah. Pada sistem yang kami buat, kita dapat mengontrol semua perangkat dengan menggunakan perintah sesuai yang kita inginkan dengan catatan perintah tersebut telah diprogram terlebih dahulu sebelumnya.

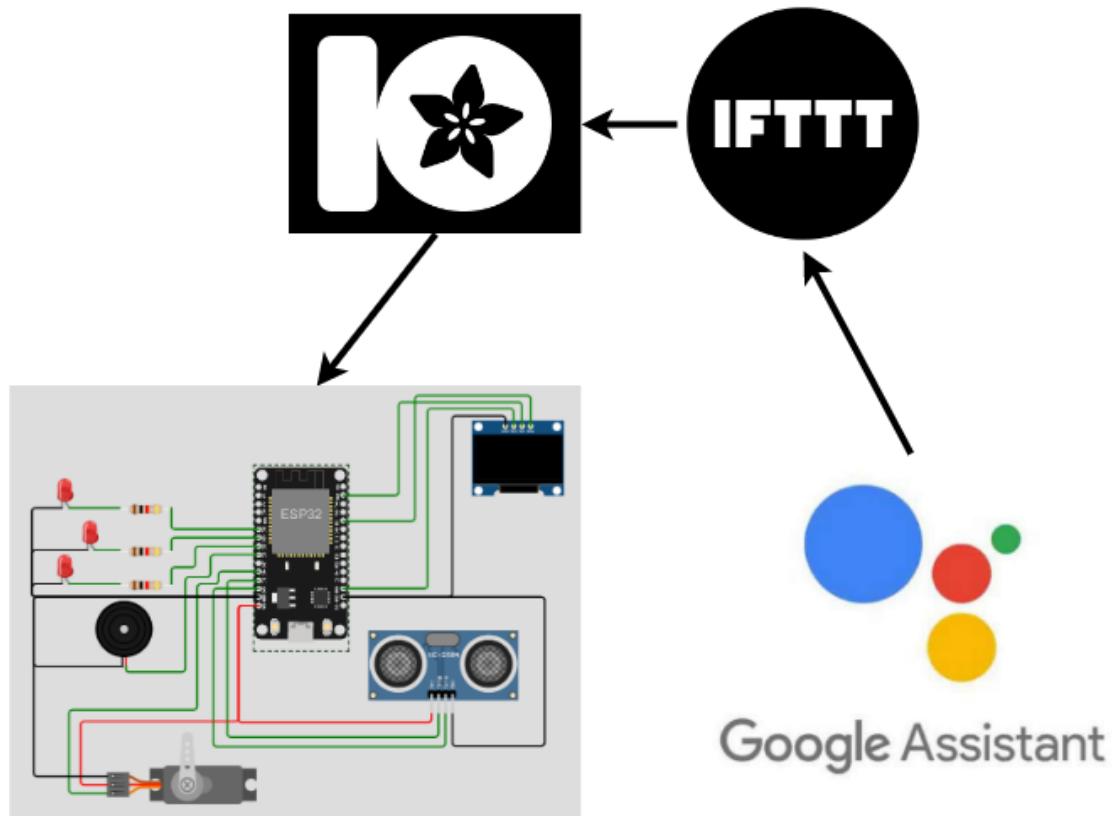


12. Handphone

Pada sistem kami handphone berfungsi sebagai tempat mengontrol lampu dan buzzer melalui aplikasi dan menjalankan perintah seperti menghidupkan melalui suara dengan bantuan google assistant serta microphone yang ada handphone sebagai sensor untuk menerima inputan suara dari pengguna.

III. Rangkaian smart home assistant

Rangkaian



Cara Kerja

Sistem **Smart Home Assistant** yang kami buat terdiri dari sensor ultrasonic dan microphone yang ada di Hp. Aktuator yang kita gunakan ada led, buzzer, motor servo dan oled. Google assistant kami gunakan sebagai *artificial intelligence (AI)* agar pengguna dapat mengontrol aktuator menggunakan suara melalui microphone Hp. Adafruit IO digunakan sebagai interface antara ESP32 dengan google assistant serta tempat untuk mengontrol aktuator melalui website. IFTTT berfungsi sebagai penerjemah perintah dari google assistant ke Adafruit IO.

Cara kerjanya, pengguna dapat memberikan perintah melalui google assistant terkait instruksi yang telah diprogram sebelumnya. Perintah tersebut sebagai berikut.

- LAMP 1 ON/ OFF
- LAMP 2 ON/ OFF
- LAMP 3 ON/ OFF
- ALL LAMP ON/ OFF
- BUZZER ON/ OFF
- OLED ON/ OFF

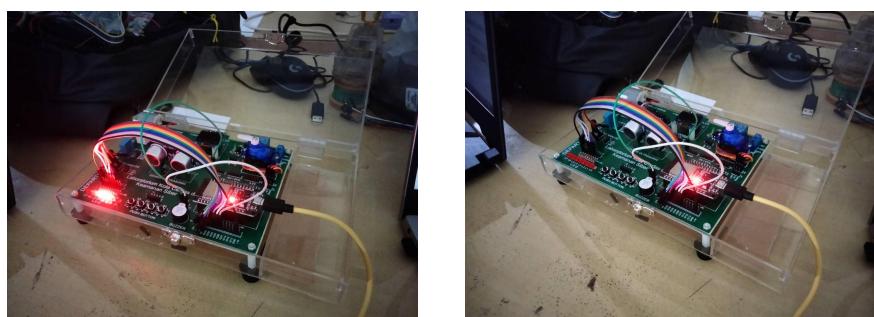
Selain dari google assistant, kita dapat melakukan perintah tersebut melalui website yaitu io.adafruit.com atau melalui aplikasi Mqtt dashboard. Setelah pengguna menginstruksikan perintah melalui google assistant maka data instruksi tersebut akan diteruskan ke Adafruit IO yang sebelumnya instruksi google assistant diterjemahkan oleh IFTTT. Kemudian, Adafruit IO akan memberikan perintah ke ESP32 sesuai perintah yang diinputkan sebelumnya melalui google assistant. Sedangkan untuk sensor ultrasonik berfungsi sebagai pendekripsi objek. Jadi, jika ada objek di depan pintu maka pintu yang digerakkan menggunakan motor servo akan terbuka.

IV. Dokumentasi

Langkah kerja

1. Rangkailah semua komponen sesuai rangkaian smart home assistant.
2. Lakukan instalasi library yang dibutuhkan, yaitu
 - `#include <WiFi.h>`, berfungsi agar ESP32 dapat terhubung dengan internet melalui wifi.
 - `#include <WebServer.h>`, berfungsi agar ESP32 dapat terhubung dengan internet.
 - `#include "Adafruit_MQTT.h"`, berfungsi agar ESP32 dapat berkomunikasi dengan Adafruit IO.
 - `#include "Adafruit_MQTT_Client.h"`, berfungsi agar ESP32 dapat berkomunikasi dengan Adafruit IO.
 - `#include "SSD1306Wire.h"`, library ini berfungsi sebagai driver untuk OLED agar OLED dapat menampilkan text ataupun gambar.
 - `#include <ESP32Servo.h>`, library ini berfungsi untuk mengontrol servo menggunakan mikrokontroler.
3. Buat akun di ADAFRUIT IO (<https://io.adafruit.com/>) dan IFTTT (<https://ifttt.com/>).
4. Buat button di ADAFRUIT IO sebanyak 5 button dengan data input 0 dan 1.
5. Buat trigger pada google assistant trigger IFTTT sebanyak yang dibutuhkan dan set trigger ke button ADAFRUIT IO dengan ADAFRUIT MQTT.
6. Sambungkan wifi ke laptop yang terhubung dengan ESP32.
7. Masukkan program di Arduino IDE dan hubungkan dengan ADAFRUIT IO melalui tokennya.

Foto Rangkaian



Source Code

```
#include <WiFi.h>
#include <WebServer.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"
#include "SSD1306Wire.h"
#include <ESP32Servo.h> //Library untuk Servo

#define LED2      32          // for 1st lamp
#define LED3      33          // for 2nd lamp
#define LED4      25          // for 3rd lamp
#define BUZZER    26          // for Buzzer

//OLED related variables
#define OLED_ADDR 0x3c
#define OLED_SDA 21    //4 //TTGO board without SD Card has OLED SDA connected to
pin 4 of ESP32
#define OLED_SCL 22    //15 //TTGO board without SD Card has OLED SCL connected to
pin 15 of ESP32
#define OLED_RST 16    //Optional, TTGO board contains OLED_RST connected to pin
16 of ESP32
#define OLED_VCC 15

const int trigPin = 12;
const int echoPin = 13;
//define sound speed in cm/uS
```

```

#define SOUND_SPEED 0.034

#define CM_TO_INCH 0.393701

long duration;

float distanceCm;

float distanceInch;

Servo myservo; //Buat object 1 buah motor servo


#define WLAN_SSID      "ITS-ASRAMA-E"           // Your SSID
#define WLAN_PASS      "itssurabaya"           // Your password

/**************** Adafruit.io Setup *******/

#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883                  // use 8883 for SSL
#define AIO_USERNAME    "ilhamssakti27"        // Replace it with your username
#define AIO_KEY         "aio_Dbtp17tcTEkX0Y2cvG9Vp7UnsKmC" // Replace with your
Project Auth Key

/**************** Global State (you don't need to change this!) *******/

// Create an ESP8266 WiFiClient class to connect to the MQTT server.

WiFiClient client;

// or... use WiFiClientSecure for SSL

//WiFiClientSecure client;

```

```
// Setup the MQTT client class by passing in the WiFi client and MQTT server and login details.
```

```
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT,  
AIO_USERNAME, AIO_KEY);
```

```
*****  
***** Feeds  
******/
```

```
// Setup a feed called 'onoff' for subscribing to changes.
```

```
Adafruit_MQTT_Subscribe lampu2 = Adafruit_MQTT_Subscribe(&mqtt,  
AIO_USERNAME"/feeds/lampu2"); // FeedName --- FOR LED2
```

```
Adafruit_MQTT_Subscribe lampu3 = Adafruit_MQTT_Subscribe(&mqtt,  
AIO_USERNAME"/feeds/lampu3"); // FeedName --- FOR LED3
```

```
Adafruit_MQTT_Subscribe lampu4 = Adafruit_MQTT_Subscribe(&mqtt,  
AIO_USERNAME"/feeds/lampu4"); // FeedName --- FOR LED2
```

```
Adafruit_MQTT_Subscribe lampu5 = Adafruit_MQTT_Subscribe(&mqtt,  
AIO_USERNAME"/feeds/lampu"); // FeedName --- FOR ALL LED
```

```
Adafruit_MQTT_Subscribe buzzer = Adafruit_MQTT_Subscribe(&mqtt,  
AIO_USERNAME"/feeds/lampu5"); // FeedName --- FOR BUZZER
```

```
Adafruit_MQTT_Subscribe oled = Adafruit_MQTT_Subscribe(&mqtt,  
AIO_USERNAME"/feeds/lampu6"); // FeedName --- FOR OLED
```

```
void MQTT_connect();
```

```
SSD1306Wire display(OLED_ADDR, OLED_SDA, OLED_SCL);
```

```
int counter = 0;
```

```
void initOLED() {
```

```

pinMode(OLED_RST, OUTPUT);
pinMode(OLED_VCC, OUTPUT);
//Give a low to high pulse to the OLED display to reset it
//This is optional and not required for OLED modules not containing a reset pin
digitalWrite(OLED_RST, LOW);
delay(20);
digitalWrite(OLED_RST, HIGH);

}

void showOLEDMessage(String line1, String line2, String line3) {
    display.init();          // clears screen
    display.setFont(ArialMT_Plain_16);
    display.drawString(0, 0, line1);    // adds to buffer
    display.drawString(0, 20, line2);
    display.drawString(0, 40, line3);
    display.display();        // displays content in buffer
}

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

    pinMode(LED2, OUTPUT);
    pinMode(LED3, OUTPUT);
    pinMode(LED4, OUTPUT);
    pinMode(BUZZER, OUTPUT);
    initOLED();
}

```

```
Serial.begin(115200); // Starts the serial communication  
pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output  
pinMode(echoPin, INPUT); // Sets the echoPin as an Input  
myservo.attach(14); //Set servo pada pin PWM 14  
  
// Connect to WiFi access point.  
Serial.println(); Serial.println();  
Serial.print("Connecting to ");  
Serial.println(WLAN_SSID);  
  
WiFi.begin(WLAN_SSID, WLAN_PASS);  
while (WiFi.status() != WL_CONNECTED) {  
    delay(500);  
    Serial.print(".");  
}  
Serial.println();  
  
Serial.println("WiFi connected");  
Serial.println("IP address: ");  
Serial.println(WiFi.localIP());  
  
// Setup MQTT subscription for onoff feed.  
mqtt.subscribe(&lampu2);  
mqtt.subscribe(&lampu3);
```

```

        mqtt.subscribe(&lampu4);

        mqtt.subscribe(&lampu5);

        mqtt.subscribe(&buzzer);

        mqtt.subscribe(&oled);

    }

void loop() {

    // Clears the trigPin

    digitalWrite(trigPin, LOW);

    delayMicroseconds(2);

    // Sets the trigPin on HIGH state for 10 micro seconds

    digitalWrite(trigPin, HIGH);

    delayMicroseconds(10);

    digitalWrite(trigPin, LOW);

    // Reads the echoPin, returns the sound wave travel time in microseconds

    duration = pulseIn(echoPin, HIGH);

    // Calculate the distance

    distanceCm = duration * SOUND_SPEED/2;

    // Convert to inches

    distanceInch = distanceCm * CM_TO_INCH;
}

```

```
// Prints the distance in the Serial Monitor
Serial.print("Distance (cm): ");
Serial.println(distanceCm);
Serial.print("Distance (inch): ");
Serial.println(distanceInch);

if(distanceCm<=15) // Jarak (Cm) dapat anda sesuaikan
{
    myservo.write(0); //Posisi servo pada 90 derajat
}

else{ //Jika jarak lebih dari yang ditentukan
    myservo.write(180); //Posisi servo pada 90 derajat
}

delay(1000);

MQTT_connect();

Adafruit_MQTT_Subscribe *subscription;
while ((subscription = mqtt.readSubscription(5000))) {
```

```
// mulai with button adafruit

if (subscription == &lampu2) {

    Serial.print(F("Got: "));

    Serial.println((char *)lampu2.lastread);

    int Light1_State = atoi((char *)lampu2.lastread);

    digitalWrite(LED2, (Light1_State));


}

if (subscription == &lampu3) {

    Serial.print(F("Got: "));

    Serial.println((char *)lampu3.lastread);

    int Light2_State = atoi((char *)lampu3.lastread);

    digitalWrite(LED3, (Light2_State));


}

if (subscription == &lampu4) {

    Serial.print(F("Got: "));

    Serial.println((char *)lampu4.lastread);

    int Light3_State = atoi((char *)lampu4.lastread);

    digitalWrite(LED4, (Light3_State));


}
```

```
if (subscription == &lampu5) {  
    Serial.print(F("Got: "));  
    Serial.println((char *)lampu5.lastread);  
    int Light4_State = atoi((char *)lampu5.lastread);  
    digitalWrite(LED2, (Light4_State));  
    digitalWrite(LED3, (Light4_State));  
    digitalWrite(LED4, (Light4_State));  
}  
}
```

```
if (subscription == &buzzer) {  
    Serial.print(F("Got: "));  
    Serial.println((char *)buzzer.lastread);  
    int Buzzer_State = atoi((char *)buzzer.lastread);  
    digitalWrite(BUZZER, (Buzzer_State));  
}  
}
```

```
if (subscription == &oled) {  
    Serial.print(F("Got: "));  
    Serial.println((char *)oled.lastread);  
    int OLED_State = atoi((char *)oled.lastread);  
    if(OLED_State == 1){  
        showOLEDMassage("Ilham", "Gennaro", "Alwan");  
        digitalWrite(OLED_VCC, HIGH);  
    }  
}
```

```

delay(1000);

}

if(OLED_State == 0){

    digitalWrite(OLED_VCC, LOW);

    showOLEDMessage("Num seconds is: ", String(counter), " 02");

    delay(1000);

}else {

    showOLEDMessage("Ilham", "Gennaro", "Alwan");

    digitalWrite(OLED_VCC, HIGH);

    delay(1000);

    continue;

}

}

void MQTT_connect() {

    int8_t ret;

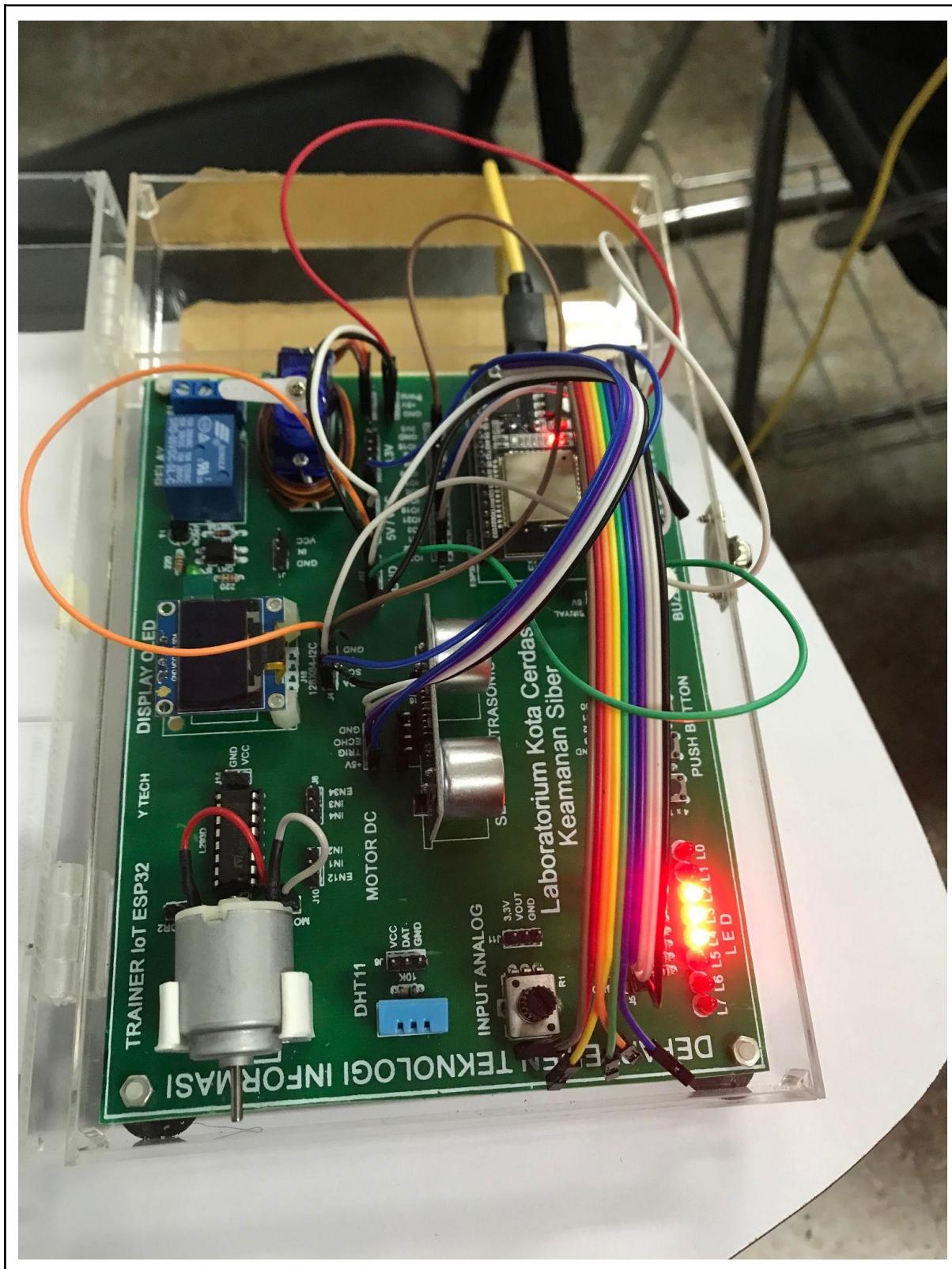
    // Stop if already connected.

    if (mqtt.connected()) {

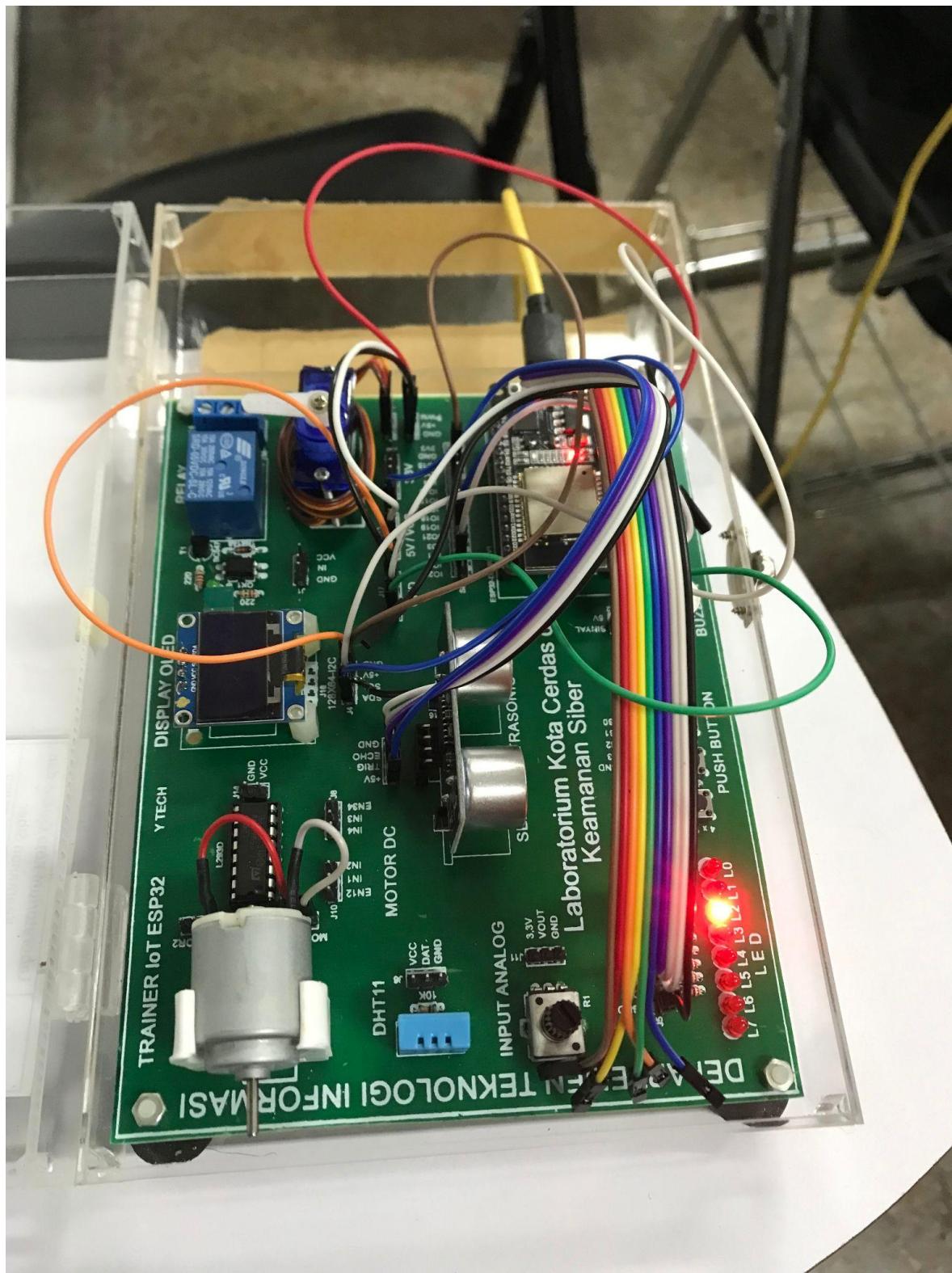

```

```
return;  
}  
  
Serial.print("Connecting to MQTT... ");  
  
uint8_t retries = 3;  
  
while ((ret = mqttt.connect()) != 0) { // connect will return 0 for connected  
    Serial.println(mqttt.connectErrorString(ret));  
    Serial.println("Retrying MQTT connection in 5 seconds...");  
    mqttt.disconnect();  
    delay(5000); // wait 5 seconds  
    retries--;  
    if (retries == 0) {  
        // basically die and wait for WDT to reset me  
        while (1);  
    }  
}  
Serial.println("MQTT Connected!");  
}
```

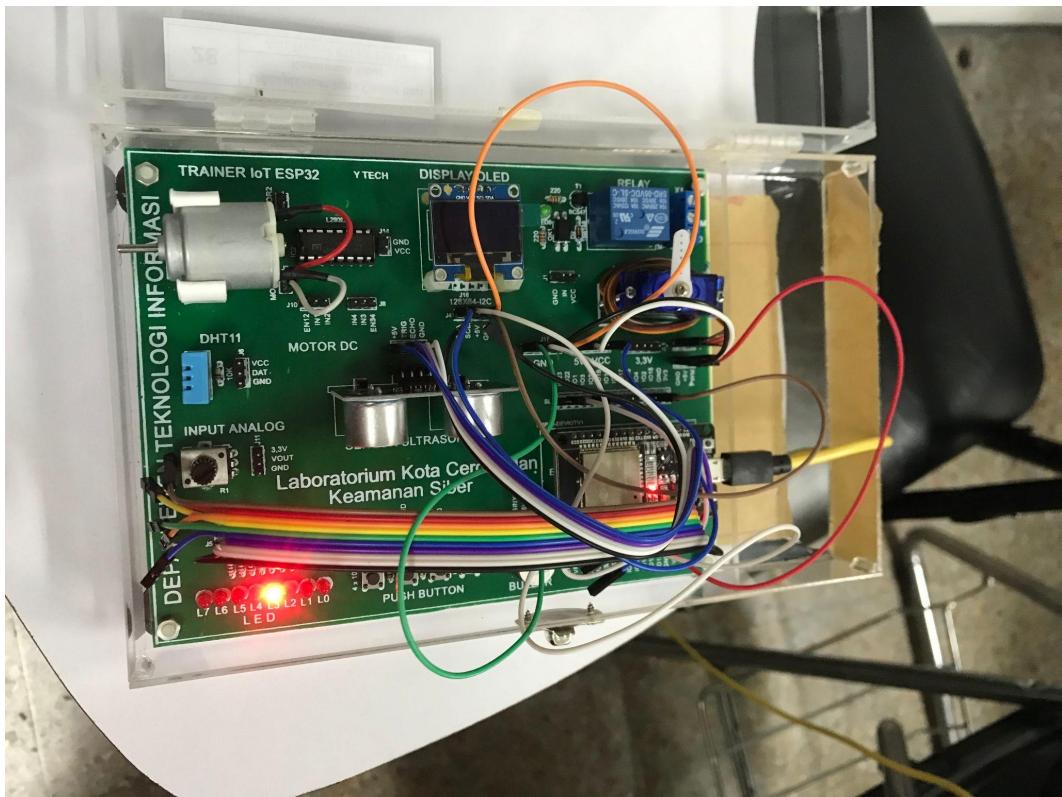
V. Lampiran



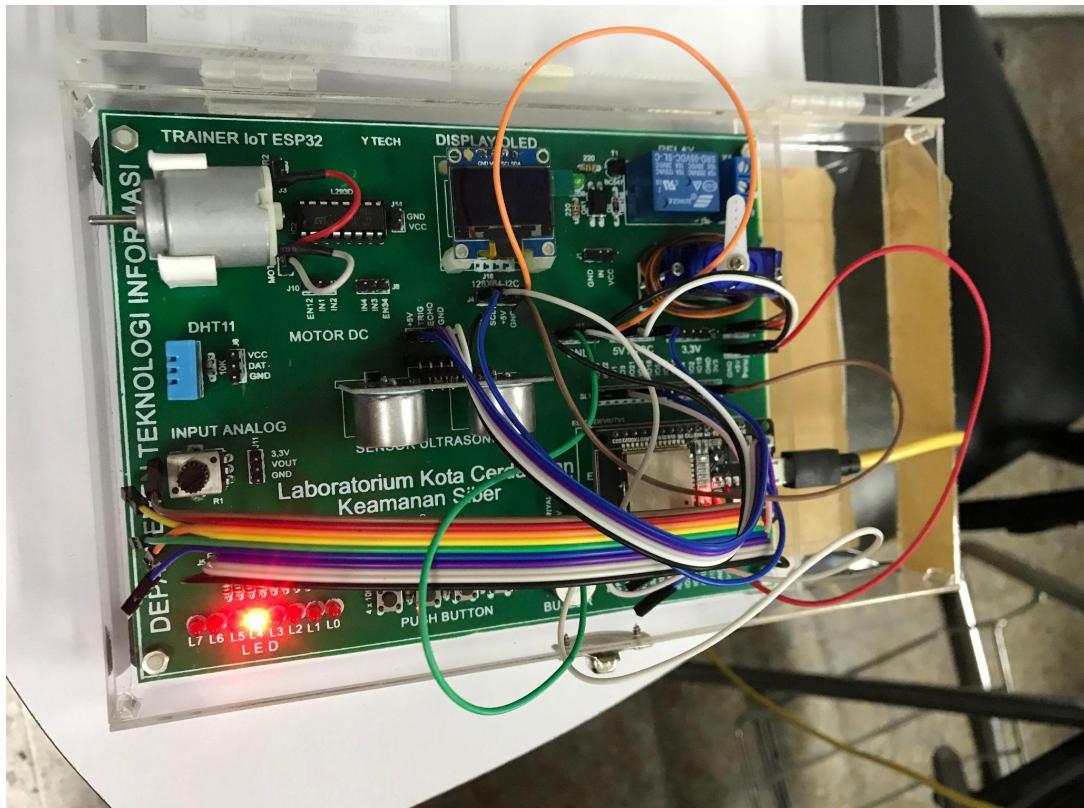
allLamp



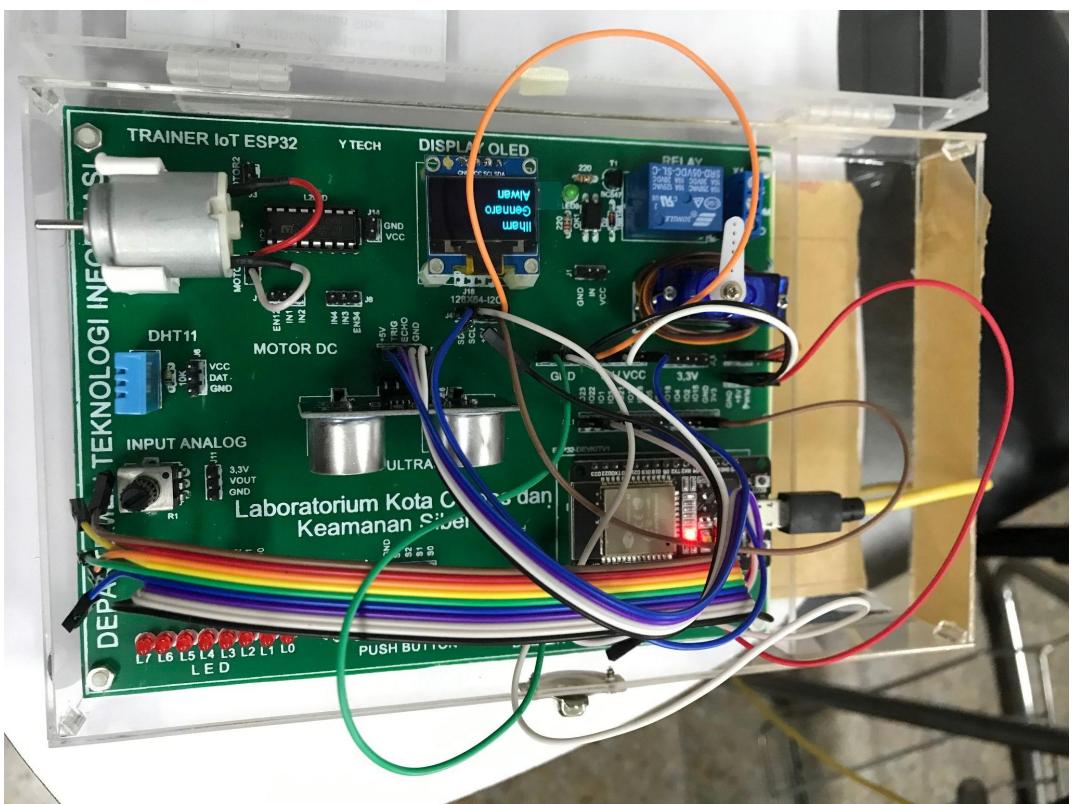
Lampu 1



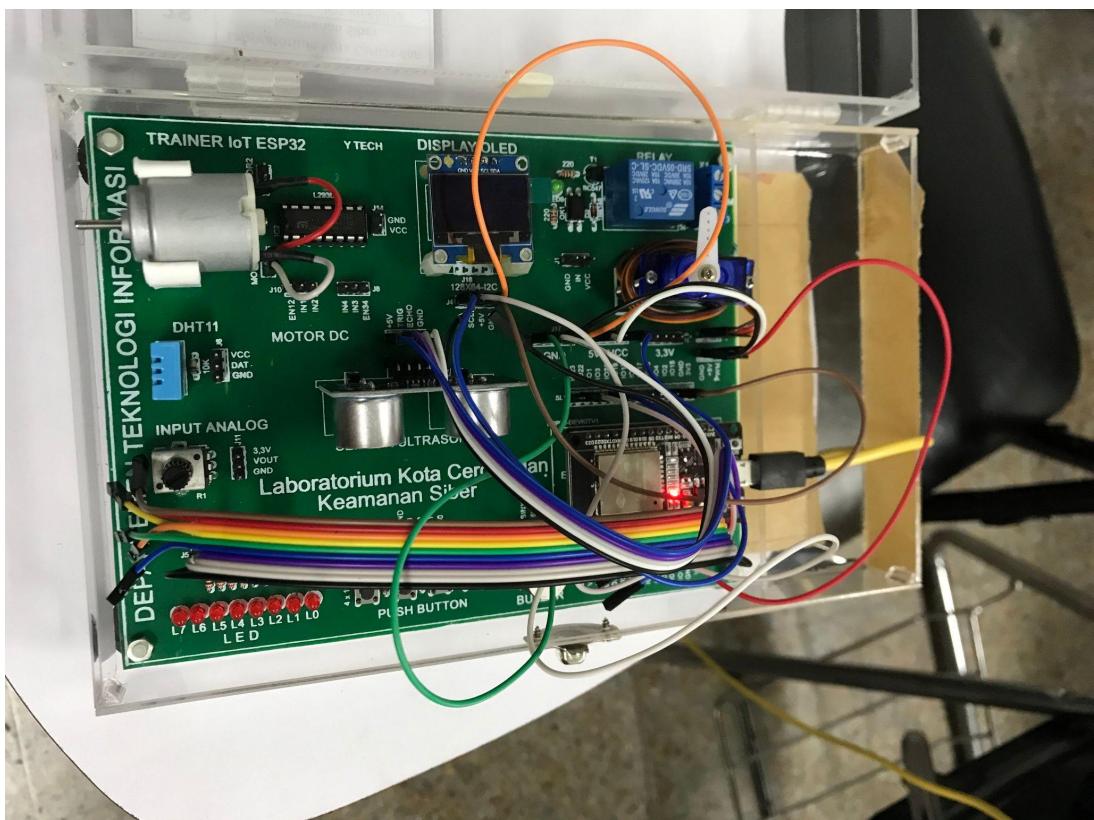
Lampu 2



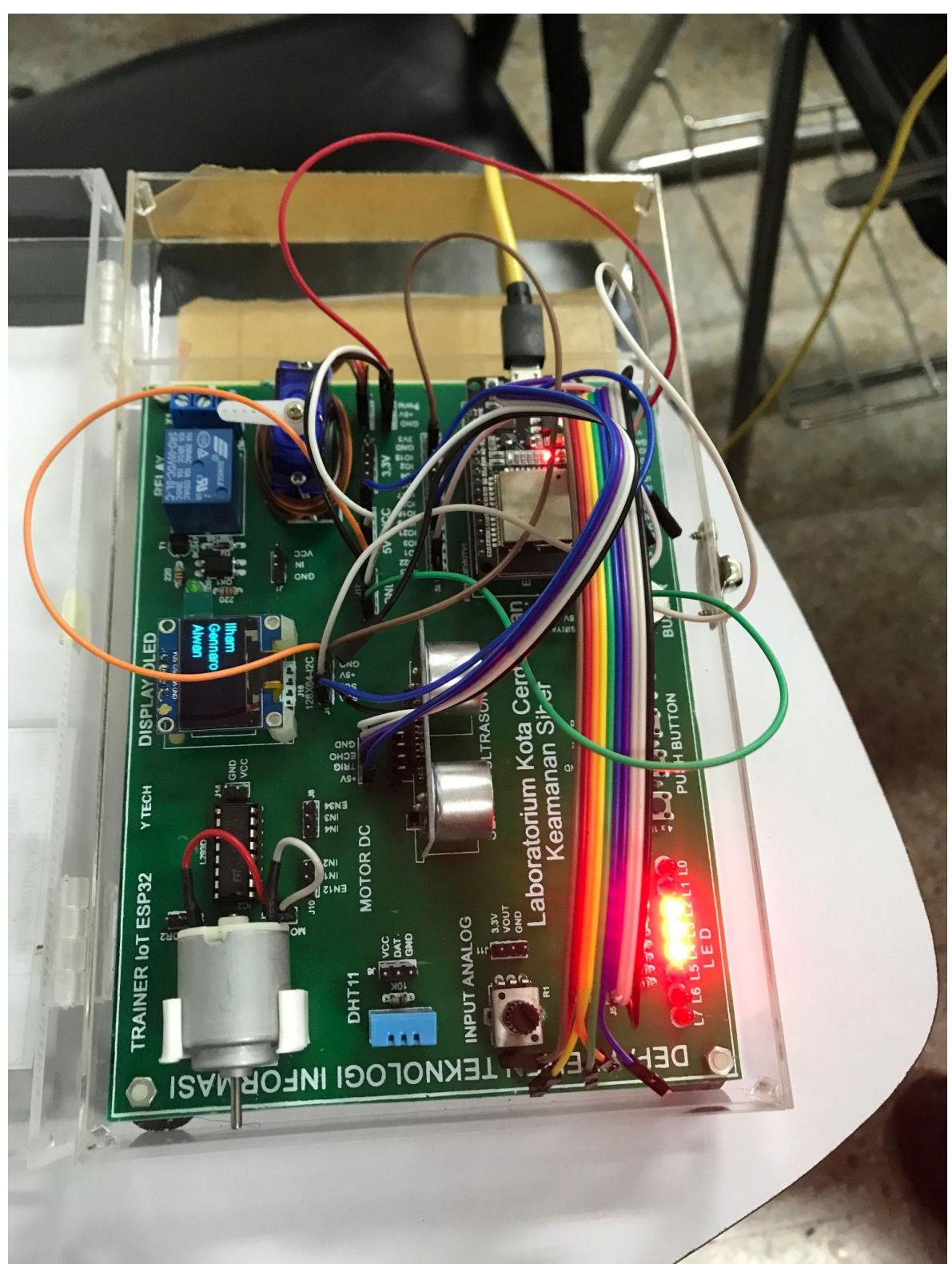
Lampu 3



OLED



BUZZER



OLED, ALL LAMP, BUZZER

DAFTAR PUSTAKA

<https://www.youtube.com/watch?v=izeQRuEkQKM&t=489s>

<http://www.boarduino.web.id/2015/11/mengontrol-servo-dengan-sensor.html>