**گزارش کار پروژه اینترنت اشیاء**

استاد : **اصغر کریم پور گمش آبادی.**

گروه داشنجویان :

1. یاسین ابراهیم نژادیان
2. امید اعظامی
3. آرش حسین پور

موضوع: **پروژه شماره 1** کنترل سرعت فن بر اساس مقدار سنسور رطوبت و نمایش ان در LCD و سرور MQTT

تاریخ: **1403/10/17**

دانشگاه : **ملی مهارت تبریز**

فهرست

[قدم0: پیش نیاز ها 5](#_Toc187062769)

[وسایل بکار رفته 5](#_Toc187062770)

[شکل کامل مدار 7](#_Toc187062771)

[کد های کامل برنامه. 7](#_Toc187062772)

[توابع loop و setup 7](#_Toc187062773)

[چطور بورد رو ریست کنیم. 8](#_Toc187062774)

[اضافه کردن برد esp32 به Arduino IDE 8](#_Toc187062775)

[کدام برد رو انتخاب کنیم. 10](#_Toc187062776)

[قدم1: خواندن اطلاعات سنسور رطوبت 11](#_Toc187062777)

[پایه های سنسور رطوبت 11](#_Toc187062778)

[سیم کشی 12](#_Toc187062779)

[کد خواندن اطلاعات سنسور رطوبت 12](#_Toc187062780)

[توضیحات کوتاه درباره کد 13](#_Toc187062781)

[قدم2: تنظیم سرعت فن 14](#_Toc187062782)

[توضیحات کوتاه درباره فن ۴ سیم 14](#_Toc187062783)

[سیم کشی 15](#_Toc187062784)

[کد کنترل فن 16](#_Toc187062785)

[توضیحات ساده درباره عملکرد کد 17](#_Toc187062786)

[قدم3: تنظیم سرعت فن بر اساس میزان رطوبت 18](#_Toc187062787)

[تابع تنظیم سرعت 18](#_Toc187062788)

[تابع loop تا اینجای کار 18](#_Toc187062789)

[تابع getActualFanSpeed 19](#_Toc187062790)

[قدم4: نمایش اطلاعات روی LCD 20](#_Toc187062791)

[سیم کشی پایه های مثبت LCD 20](#_Toc187062792)

[سیم کشی پایه های منفی LCD 20](#_Toc187062793)

[سیم کشی پایه های متصل به برد 21](#_Toc187062794)

[کد مربوط به کنترل LCD 21](#_Toc187062795)

[پارامتر های LiquidCrystal 22](#_Toc187062796)

[نمایش اطلاعات روی LCD 22](#_Toc187062797)

[تابع refreshLCD 22](#_Toc187062798)

[ارور ها یا مشکلات LCD که حل کردیم. 23](#_Toc187062799)

[قدم5:ارسال اطلاعات با MQTT 23](#_Toc187062800)

[اتصال به WiFi 23](#_Toc187062801)

[اضافه کردن لایبرری PubSubClient 23](#_Toc187062802)

[چطور فایل zip لایبرری رو وارد برنامه کنیم. 24](#_Toc187062803)

[اتصال به سرور یا بروکر MQTT 25](#_Toc187062804)

[ارسال اطلاعات سنسور ها در MQTT 26](#_Toc187062805)

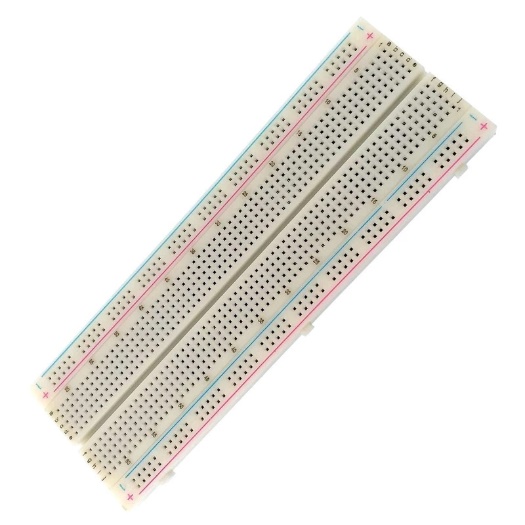
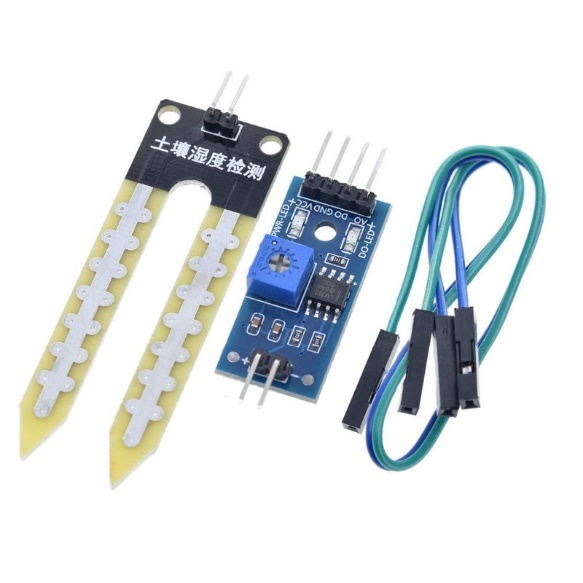
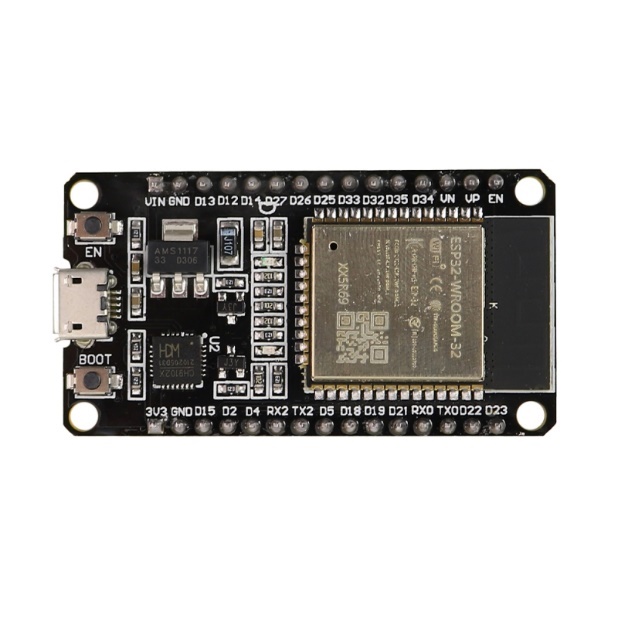
[تابع publishMqttData 27](#_Toc187062806)

[!!!: تابع loop تا اینجای کار 28](#_Toc187062807)

[راه اندازی برنامه MQTT Dash 29](#_Toc187062808)

# قدم0: پیش نیاز ها

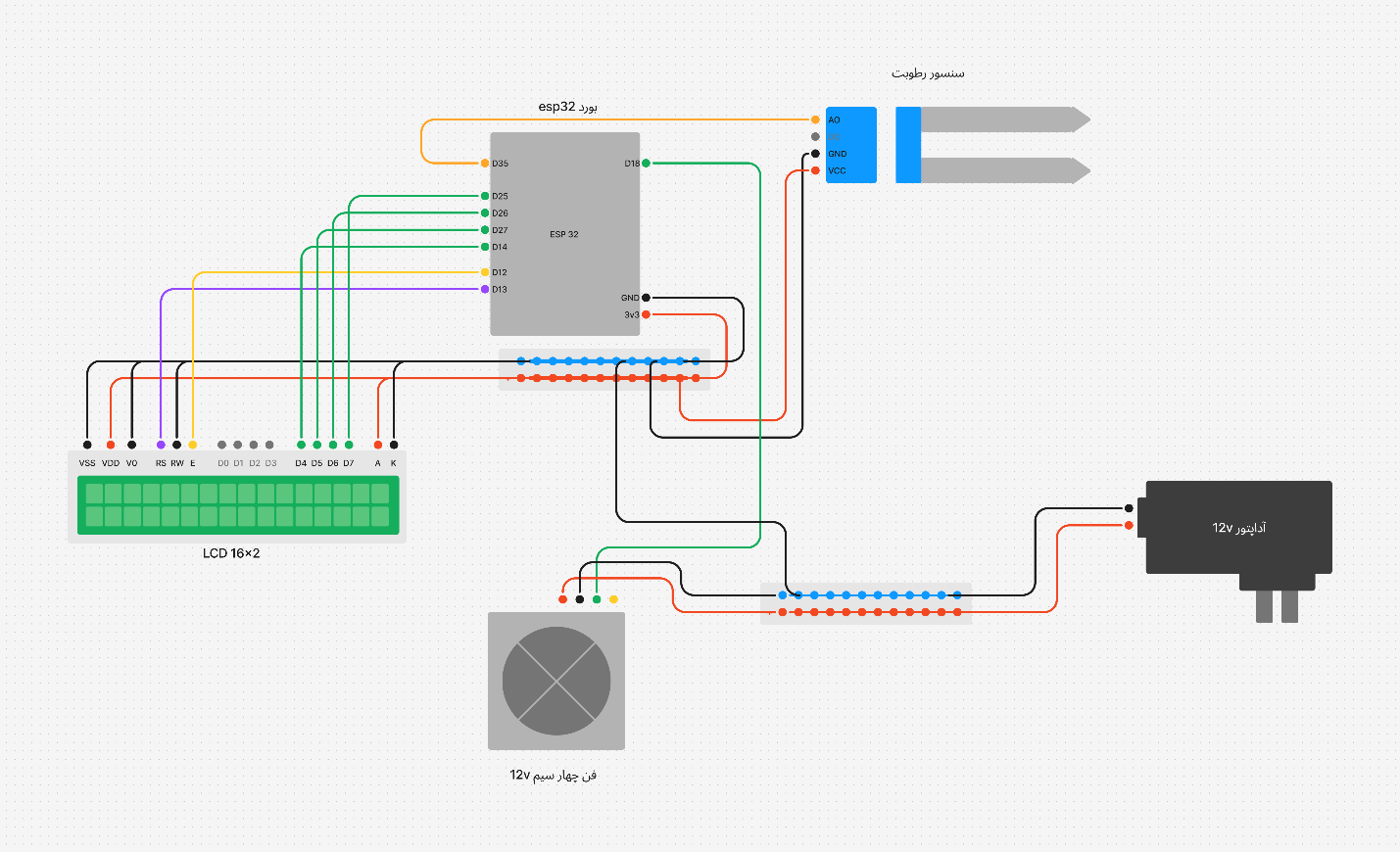
## وسایل بکار رفته



لیست لوازم:

1. بورد esp32 wroom 30 pin
2. Lcd 16x2 بدون i2c
3. سنسور رطوبت خاک یا soil moisture
4. فن 4 سیم brushless 12V
5. اداپتور 12V
6. سیم جامپر
7. برد بورد

## شکل کامل مدار



## کد های کامل برنامه.

کد های کامل و تمام فایل ها در ریپازیتوری زیر قرار دارند.

<https://github.com/nezhadian/IOT2024>

## توابع loop و setup

**1.تابع setup**:

**کاربرد**: این تابع **یک** بار در ابتدای اجرای برنامه فراخوانی می‌شود.

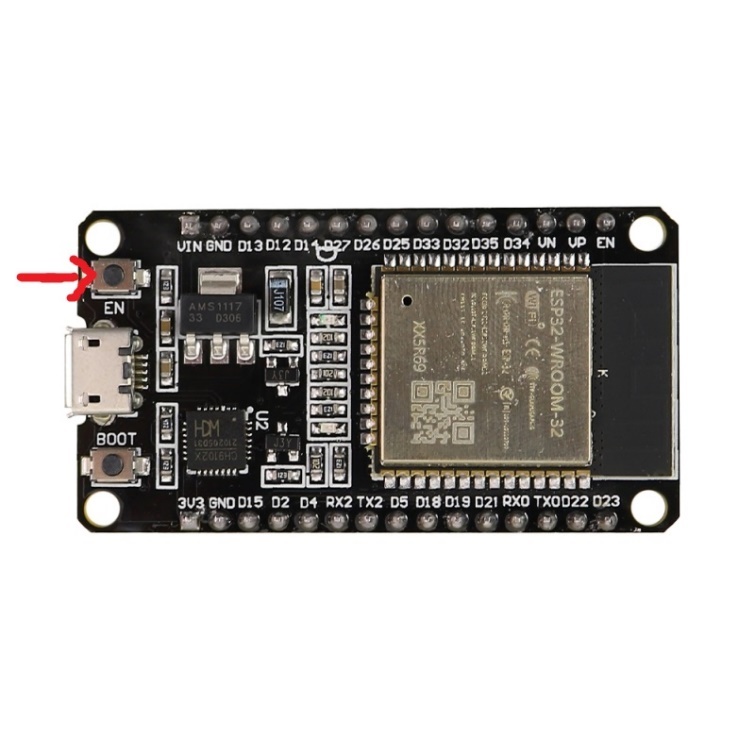
**هدف**: برای انجام تنظیمات اولیه و آماده‌سازی سخت‌افزار و نرم‌افزار استفاده می‌شود.

**۲. تابع loop:**

**کاربرد**: این تابع به طور مداوم و **بی‌پایان** تکرار می‌شود.

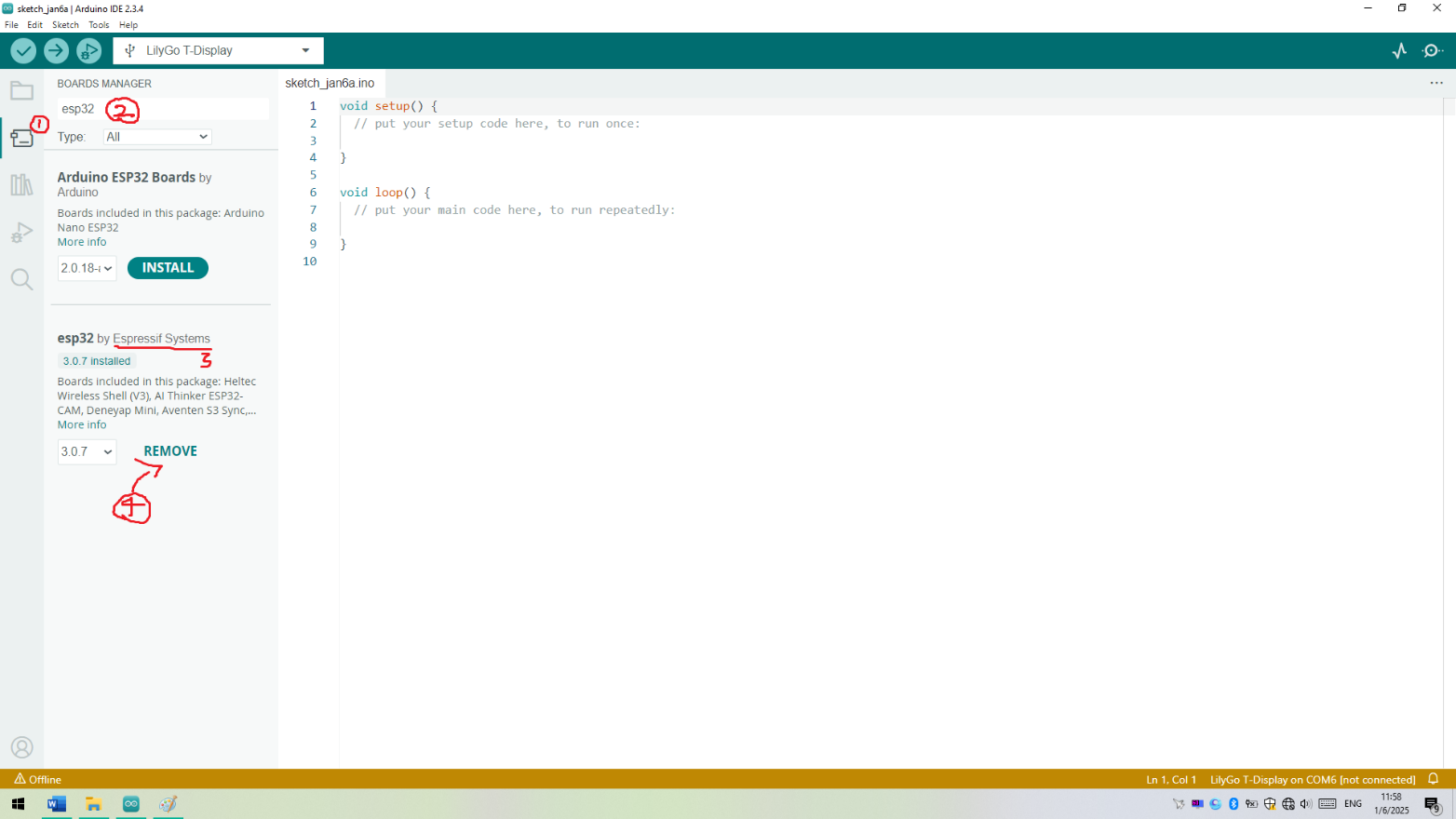
**هدف**: برای اجرای کدهای اصلی برنامه (مانند خواندن سنسورها، کنترل موتورها، ارسال داده‌ها و ...) استفاده می‌شود.

## چطور بورد رو ریست کنیم.



با استفاده از این دکمه روی برد میتوانیم برد رو ریست کنیم. با اینکار روند اجرای برنامه از اول شروع میشه.

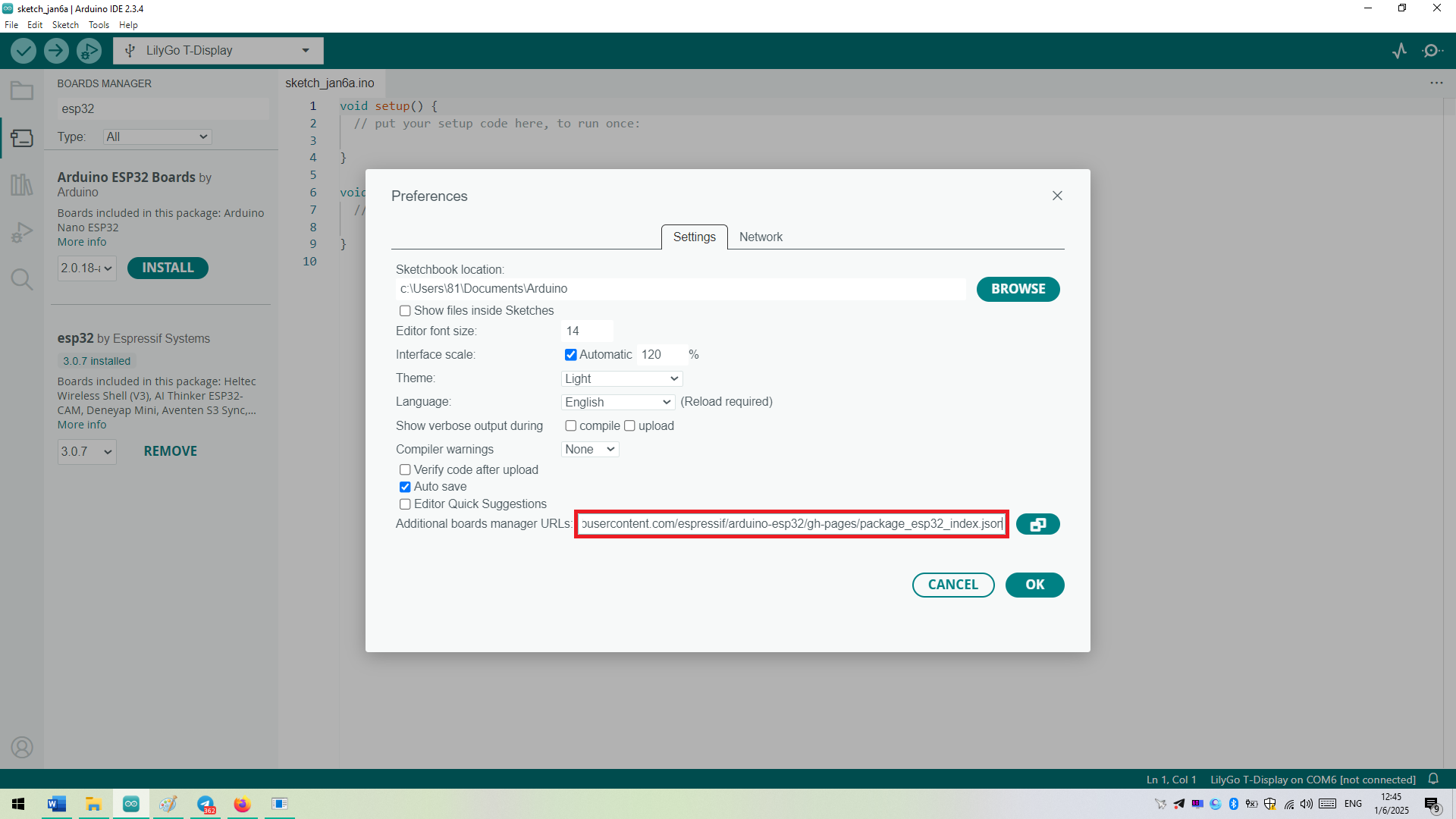
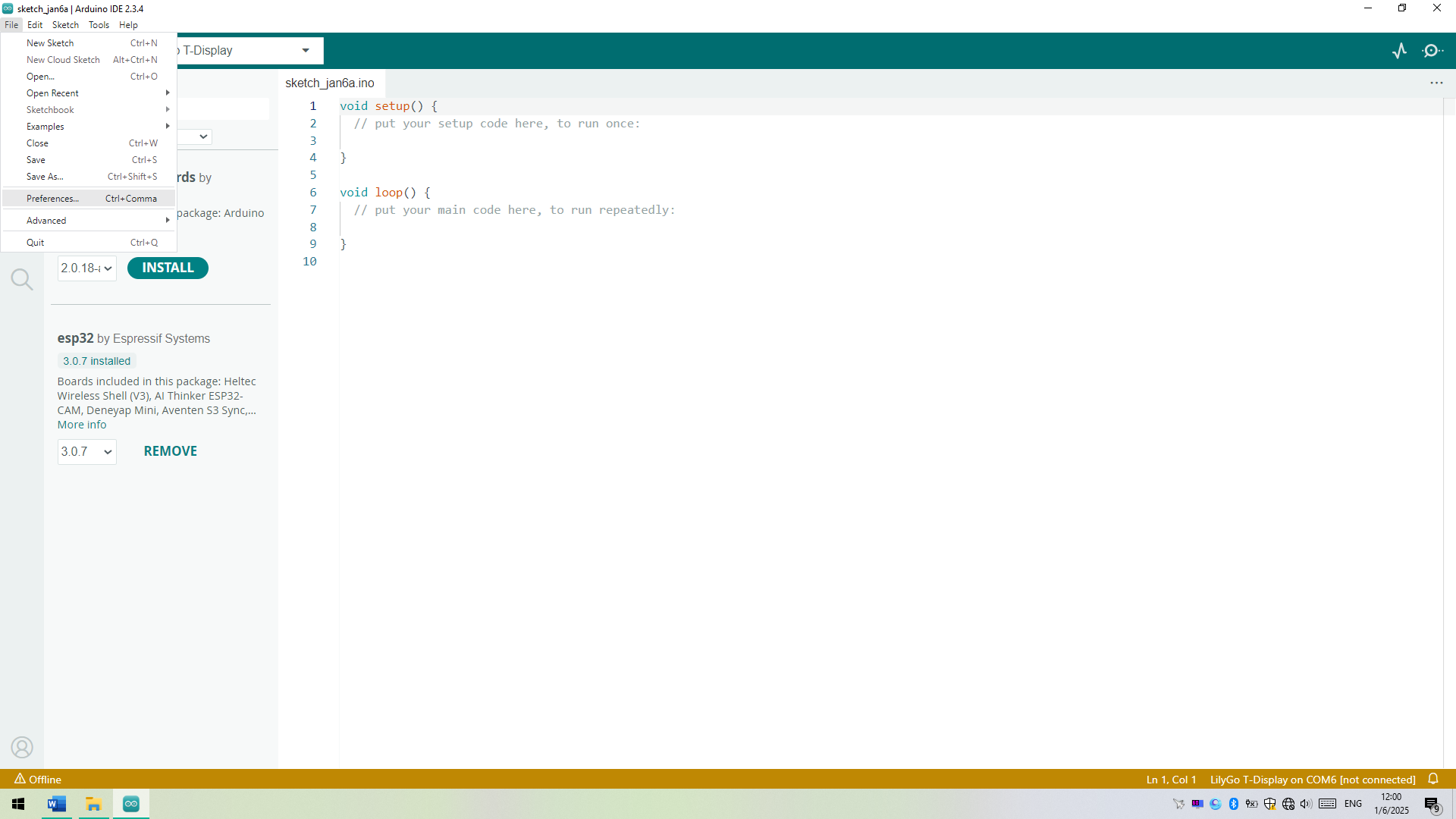
## اضافه کردن برد esp32 به Arduino IDE



برای استفاده از برد esp32 لازمه که برد اون رو در Arduino IDE نصب کنیم.

1. روی Boards Manager کلیک میکنیم.
2. Esp32 رو سرچ میکنیم.
3. مطمئن میشیم که بورد مورد نظر از طرف Espressif Systems است.
4. اون رو نصب میکنیم.

اگر برد esp32 در Boards Manager موجود نبود چیکار کنیم؟

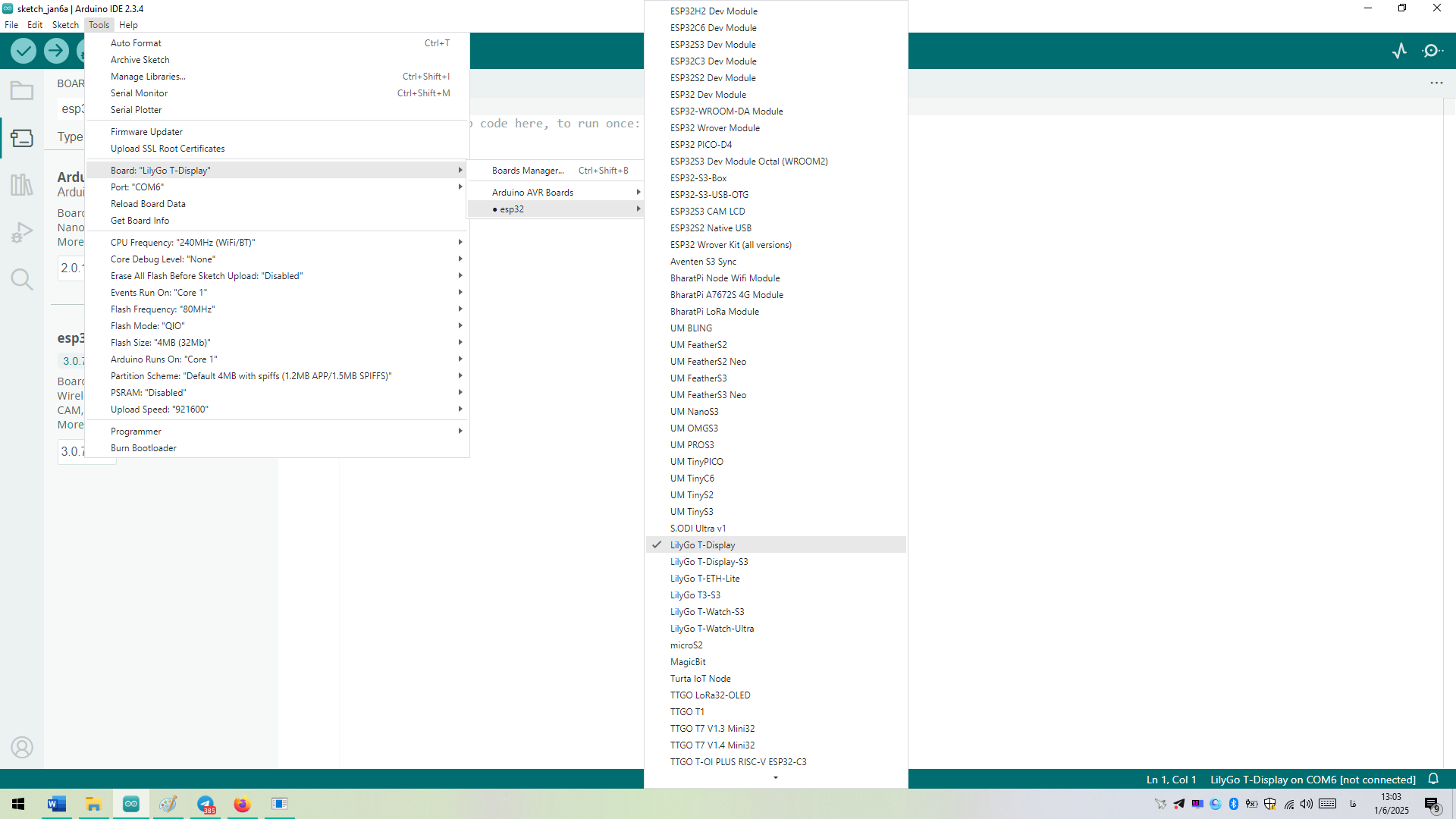


و این ادرس رو مینویسیم تا برنامه برد را پیدا کند.

https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package\_esp32\_index.json

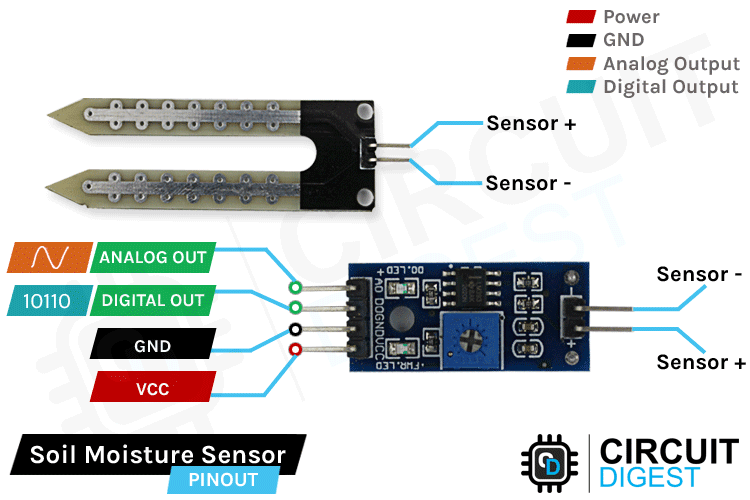
## کدام برد رو انتخاب کنیم.

برای این برد ما گذینه ی LilyGo T-Display سازگار است.



# قدم1: خواندن اطلاعات سنسور رطوبت

## پایه های سنسور رطوبت



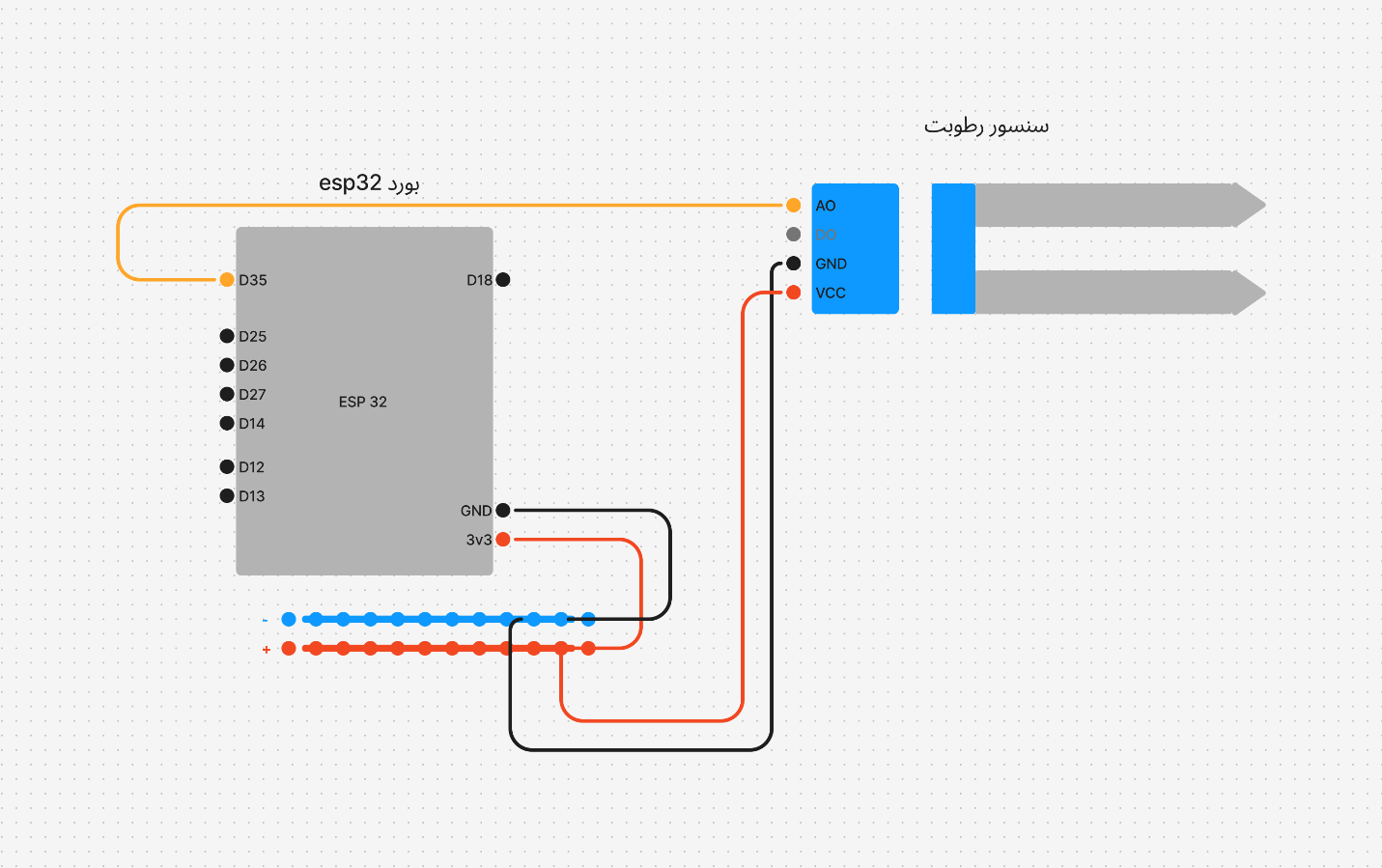
**VCC**: پایه مثبت ولتاژ معمولا 3.3v تا 5v

**GND**: پایه منفی

**DO**: خروجی دیجیتال که در صورت رسیدن به آستانه رطوبت، سیگنال HIGH یا LOW می‌دهد.

**AO**: خروجی آنالوگ که مقدار دقیق رطوبت خاک را ارائه می‌دهد.

## سیم کشی



بر اساس تصویر سیم کشی را انجام میدهیم.

## کد خواندن اطلاعات سنسور رطوبت

// تعریف پایه سنسور رطوبت خاک

const int soilPin = 35;

void setup() {

  // شروع ارتباط سریال برای نمایش اطلاعات

  Serial.begin(115200);

  // تنظیم پایه سنسور به عنوان ورودی

  pinMode(soilPin, INPUT);

}

// تابع خواندن مقدار رطوبت خاک

int readSoilMoistureValue() {

  // خواندن مقدار آنالوگ از سنسور

  int analog = analogRead(soilPin);

  // تبدیل مقدار آنالوگ به درصد رطوبت (۰ تا ۱۰۰)

  int soilValue = 100 - map(analog, 0, 4095, 0, 100);

  // نمایش مقدار آنالوگ و درصد رطوبت در سریال مانیتور

  Serial.print("Debug: soil analog ");

  Serial.println(analog);

  Serial.print("Debug: soil value ");

  Serial.println(soilValue);

  return soilValue;

}

void loop() {

  // خواندن مقدار رطوبت خاک

  int soilValue = readSoilMoistureValue();

  // نمایش مقدار رطوبت خاک در سریال مانیتور

  Serial.print("Sensor: Soil ");

  Serial.println(soilValue);

  // تاخیر ۱ ثانیه‌ای قبل از خواندن مجدد

  delay(1000);

}

## توضیحات کوتاه درباره کد

1. **تابع analogRead**
   * این تابع مقدار آنالوگ سنسور رطوبت خاک را می‌خواند و یک عدد بین **۰ تا ۴۰۹۵** برمی‌گرداند.
   * **۰** نشان‌دهنده حداکثر رطوبت (خاک کاملاً خیس).
   * **۴۰۹۵** نشان‌دهنده حداقل رطوبت (خاک کاملاً خشک).
2. **دستور map**
   * این تابع یک عدد از یک بازه (مثلاً ۰ تا ۴۰۹۵) را به بازه‌ی دیگری (مثلاً ۰ تا ۱۰۰) تبدیل می‌کند.
   * در این کد، map(analog, 0, 4095, 0, 100) مقدار آنالوگ را به یک درصد (۰ تا ۱۰۰) تبدیل می‌کند.
3. **چرا خروجی map را منهای ۱۰۰ کرده‌ایم؟**
   * چون مقدار سنسور با رطوبت رابطه معکوس دارد (هرچه خاک خشک‌تر باشد، عدد بزرگ‌تر است).
   * با منهای کردن از ۱۰۰، این رابطه معکوس می‌شود
     + **۰** خاک کاملاً خشک (۰٪ رطوبت).
     + **۱۰۰** خاک کاملاً خیس (۱۰۰٪ رطوبت).
   * این کار باعث می‌شود خروجی قابل‌درکتر باشد و نشان‌دهنده درصد واقعی رطوبت خاک باشد.

# قدم2: تنظیم سرعت فن



## توضیحات کوتاه درباره فن ۴ سیم

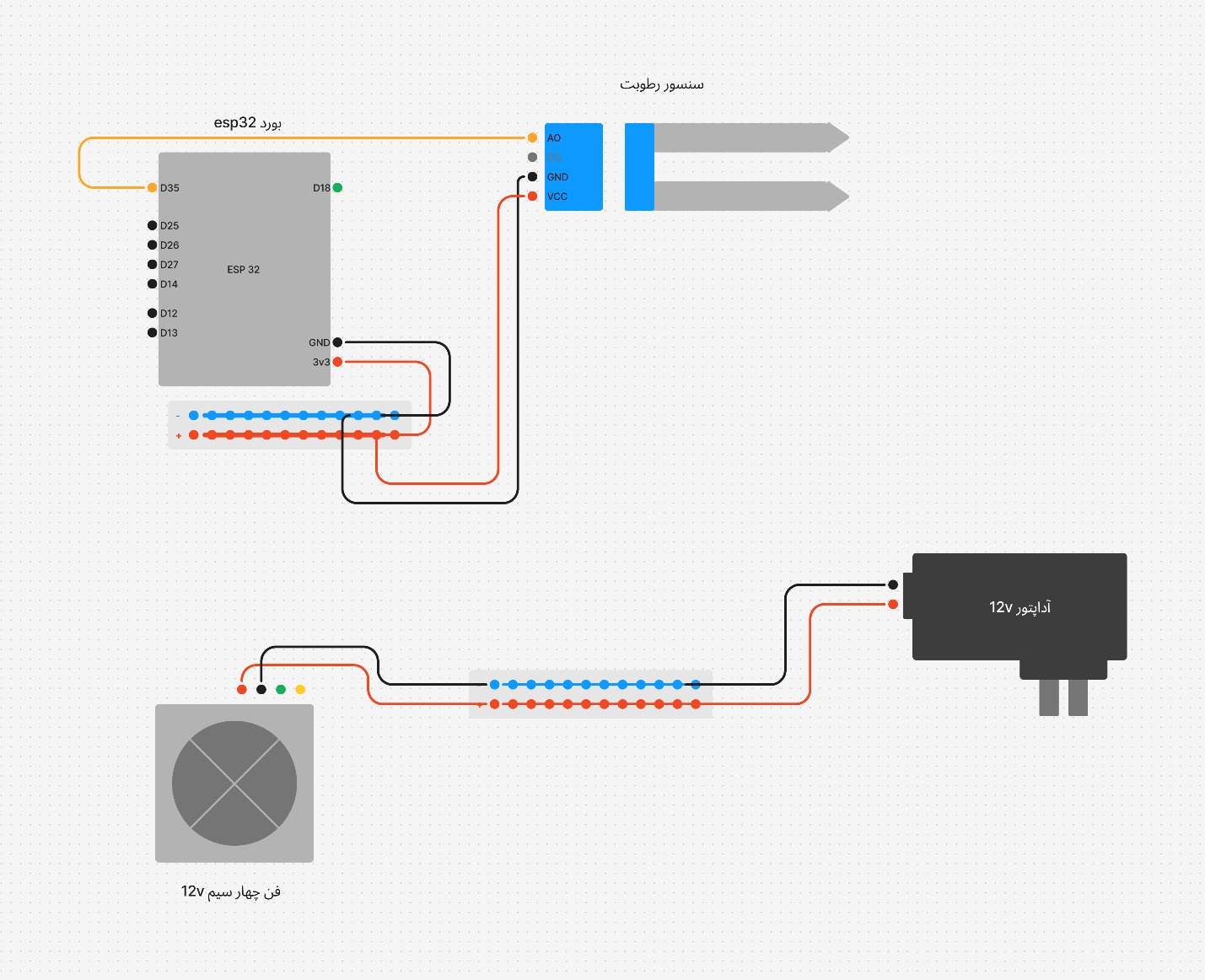
**سیم قرمز (+)**  
این سیم به منبع تغذیه **۱۲ ولت** متصل می‌شود تا برق مورد نیاز فن تأمین شود. این سیم قرمز به هیچ وجه نباید با بورد ارتباط داشته باشد. چون درجا بورد رو میسوزاند.

**سیم مشکی (GND)**  
این سیم به **زمین (GND)** مدار متصل می‌شود تا مدار کامل شود.

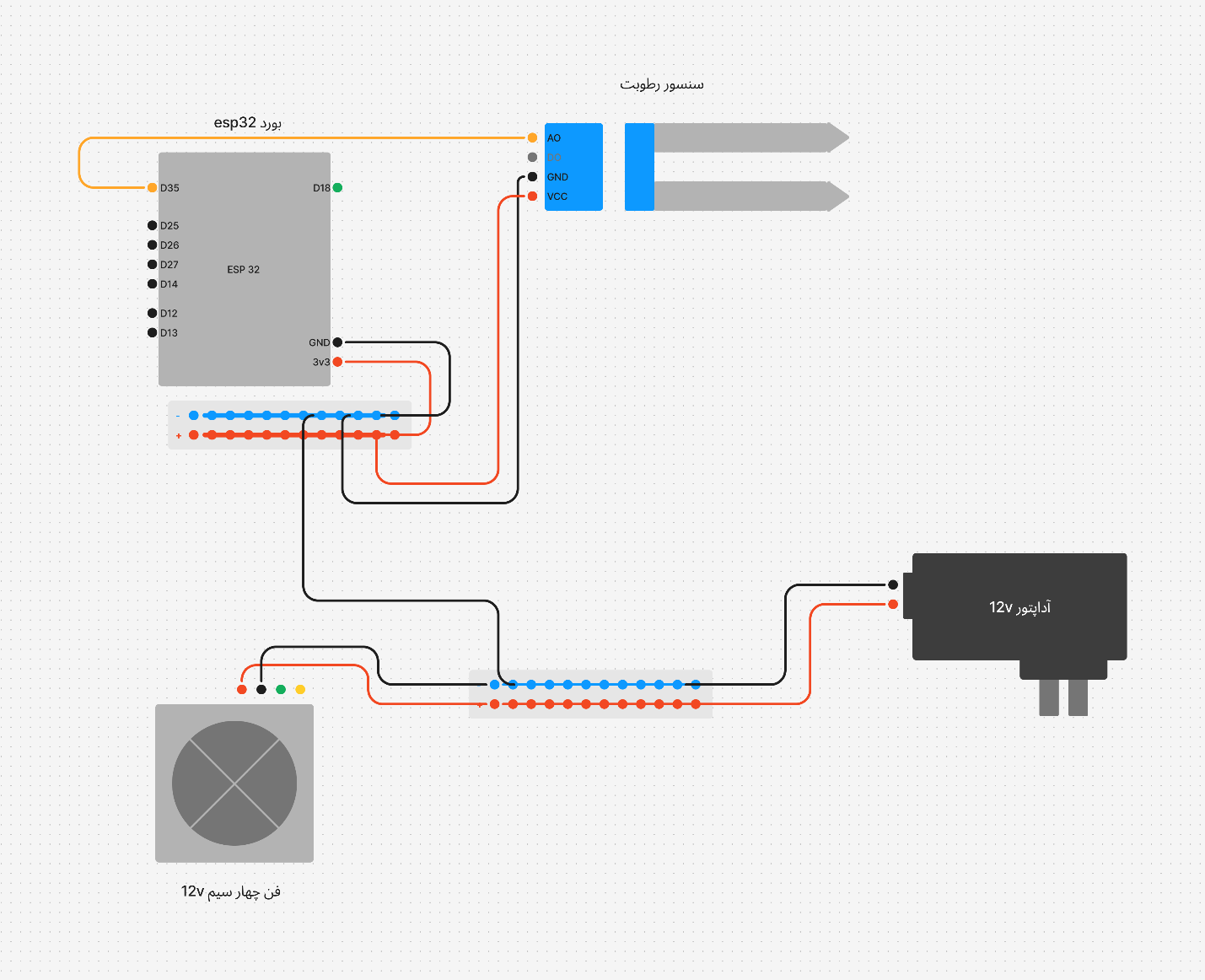
**سیم سبز (PWM)**  
این سیم برای **کنترل سرعت فن** استفاده می‌شود. با ارسال سیگنال PWM (مدولاسیون عرض پالس) به این سیم، می‌توان سرعت فن را تنظیم کرد.  
فرکانس معمول PWM برای فن‌های ۴ سیم، حدود **۲۵ کیلوهرتز** است.

**سیم زرد (Tachometer)**  
این سیم برای **خواندن سرعت فن** استفاده می‌شود. که ما ازش استفاده نمیکنیم.

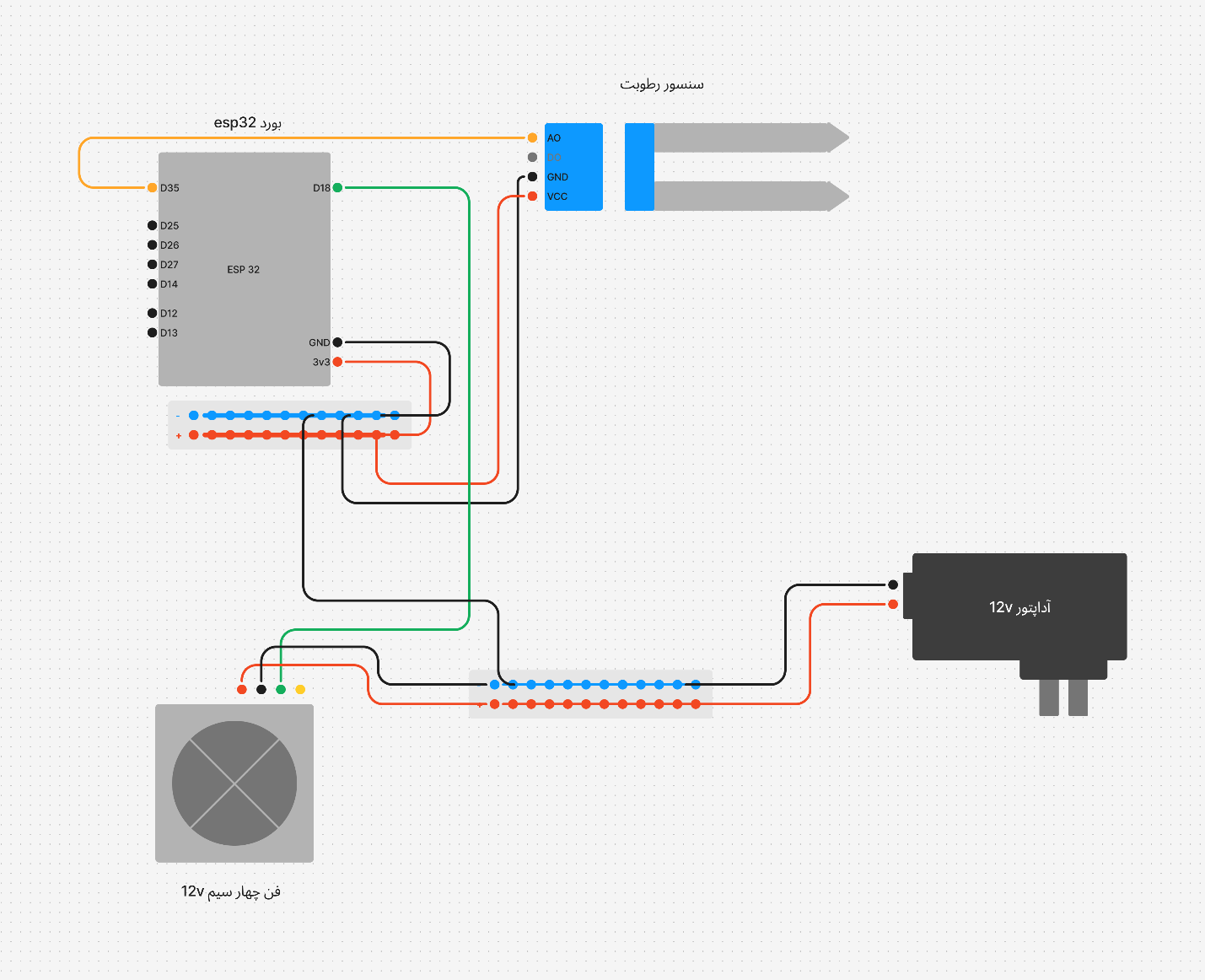
## سیم کشی



فن باید به این شیوه به اداپتور 12v وصل شود.



**نکته خیلی مهم:** GND یا پایه منفی اداپتور باید به پایه GND برد متصل شود در غیر اینصورت سرعت فن کنترل نمیشود.



در اخر هم سیم سبز رو به پایه‌ی D18 وصل میکنیم.

## کد کنترل فن

// تعریف پایه فن

const int fanPin = 18;

long curFanSpeed = 0;

void setup() {

  // شروع ارتباط سریال برای نمایش اطلاعات

  Serial.begin(115200);

  // تنظیم PWM برای کنترل سرعت فن

  const int pwmFreq = 20000;   // فرکانس PWM (۲۰ کیلوهرتز)

  const int pwmChannel = 0;    // کانال PWM

  const int pwmResolution = 8; // رزولوشن PWM (۸ بیت = مقادیر ۰ تا ۲۵۵)

  ledcAttachChannel(fanPin, pwmFreq, pwmResolution,8); // اتصال پایه فن به کانال PWM

  // تنظیم سرعت اولیه فن

  setFanSpeed(255); // فن خاموش

}

// تابع تنظیم سرعت فن

void setFanSpeed(int speed) {

  ledcWrite(fanPin, speed);  // تنظیم سرعت فن با استفاده از PWM

  curFanSpeed = speed;       // ذخیره سرعت فعلی فن

}

void loop() {

  // تغییر سرعت فن به صورت تستی

  setFanSpeed(0);   // سرعت حداکثر

  delay(2000);      // تاخیر ۲ ثانیه

  setFanSpeed(128); // سرعت متوسط

  delay(2000);      // تاخیر ۲ ثانیه

  setFanSpeed(255); // توقف فن

  delay(2000);      // تاخیر ۲ ثانیه

}

## توضیحات ساده درباره عملکرد کد

۱. تابع ledcAttachChannel

**کاربرد**: این تابع پایه‌ی مورد نظر (fanPin) را به یک کانال PWM متصل می‌کند. و آن را روی فرکانس 20k hz تنظیم میکند. این مقدار بر اساس اطلاعات فن تنظیم شده است.

۲. تابع setFanSpeed

**کاربرد**: این تابع سرعت فن را تنظیم می‌کند.

**پارامترها**:

Speed: مقدار سرعت فن (بین ۰ تا ۲۵۵).

اگر speed = 0 باشد، فن با حداکثر سرعت می‌چرخد.

اگر speed = 255 باشد، فن خاموش می‌شود.

**عملکرد**:

با استفاده از ledcWrite، مقدار speed را به کانال PWM اعمال می‌کند.

مقدار speed را در متغیر curFanSpeed ذخیره می‌کند.

# قدم3: تنظیم سرعت فن بر اساس میزان رطوبت

## تابع تنظیم سرعت

اگر بخواهیم بر اساس چیزی که در توضیحات پروژه نوشته شده سرعت فن را تنظیم کنیم باید یه همچین کدی بنویسیم.

void setFanSpeedBasedOnSoilMoisture(int soilValue){

    if(soilValue > 30)

      setFanSpeed(0);

    else if(soilValue < 24)

      setFanSpeed(255);

}

اما چون این فن ما پتانسیل بیشتری داشت. ما کد هوشمندانه تری رو برایش نوشتیم.

void setFanSpeedBasedOnSoilMoisture(int soilValue){

    if(soilValue < 30)

      setFanSpeed(255);

    else if(soilValue < 70)

      setFanSpeed(128);

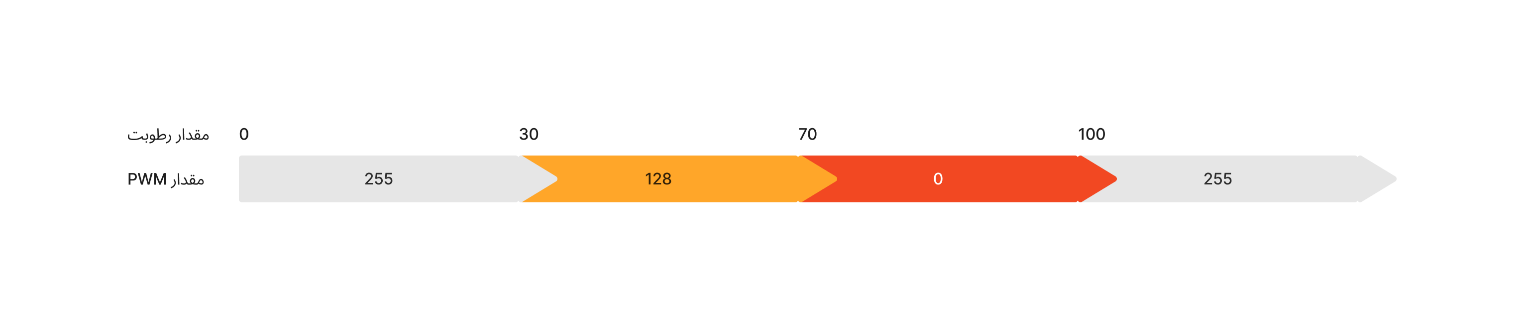
    else if(soilValue <= 100)

      setFanSpeed(0);

    else

      setFanSpeed(255);

}



## تابع loop تا اینجای کار

در اخر این کد ما برای تابع loop است که مقدار رطوبت رو میخونه و بر اساس اون سرعت فن رو تنظیم میکنه.

void loop() {

    // خواندن مقدار رطوبت خاک از سنسور

    float soilValue = readSoilMoistureValue();

    // نمایش مقدار رطوبت خاک در سریال مانیتور

    Serial.print("Sensor: Soil ");

    Serial.println(soilValue);

    // تنظیم سرعت فن بر اساس مقدار رطوبت خاک

    setFanSpeedBasedOnSoilMoisture(soilValue);

    // محاسبه سرعت واقعی فن (بر اساس RPM) از مقدار PWM فعلی

    int actualFanSpeed = getActualFanSpeed(curFanSpeed);

    // نمایش سرعت فن در سریال مانیتور

    Serial.print("Sensor: Fan speed ");

    Serial.println(actualFanSpeed);

    // تاخیر ۵۰۰ میلی‌ثانیه قبل از تکرار حلقه

    delay(500);

}

## تابع getActualFanSpeed

int getActualFanSpeed(int value){

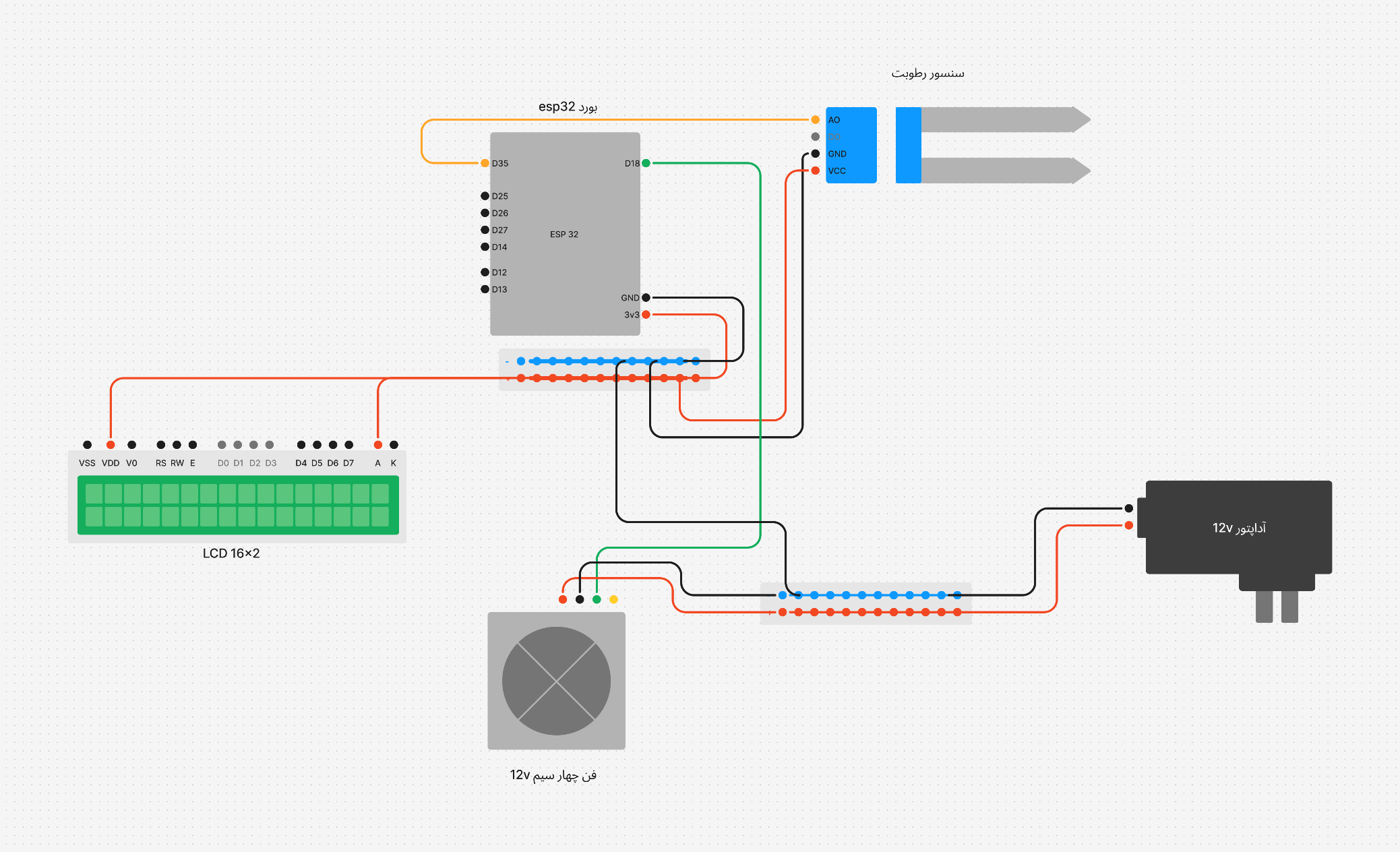
  return 12000 - map(value,0,255,0,12000);

}

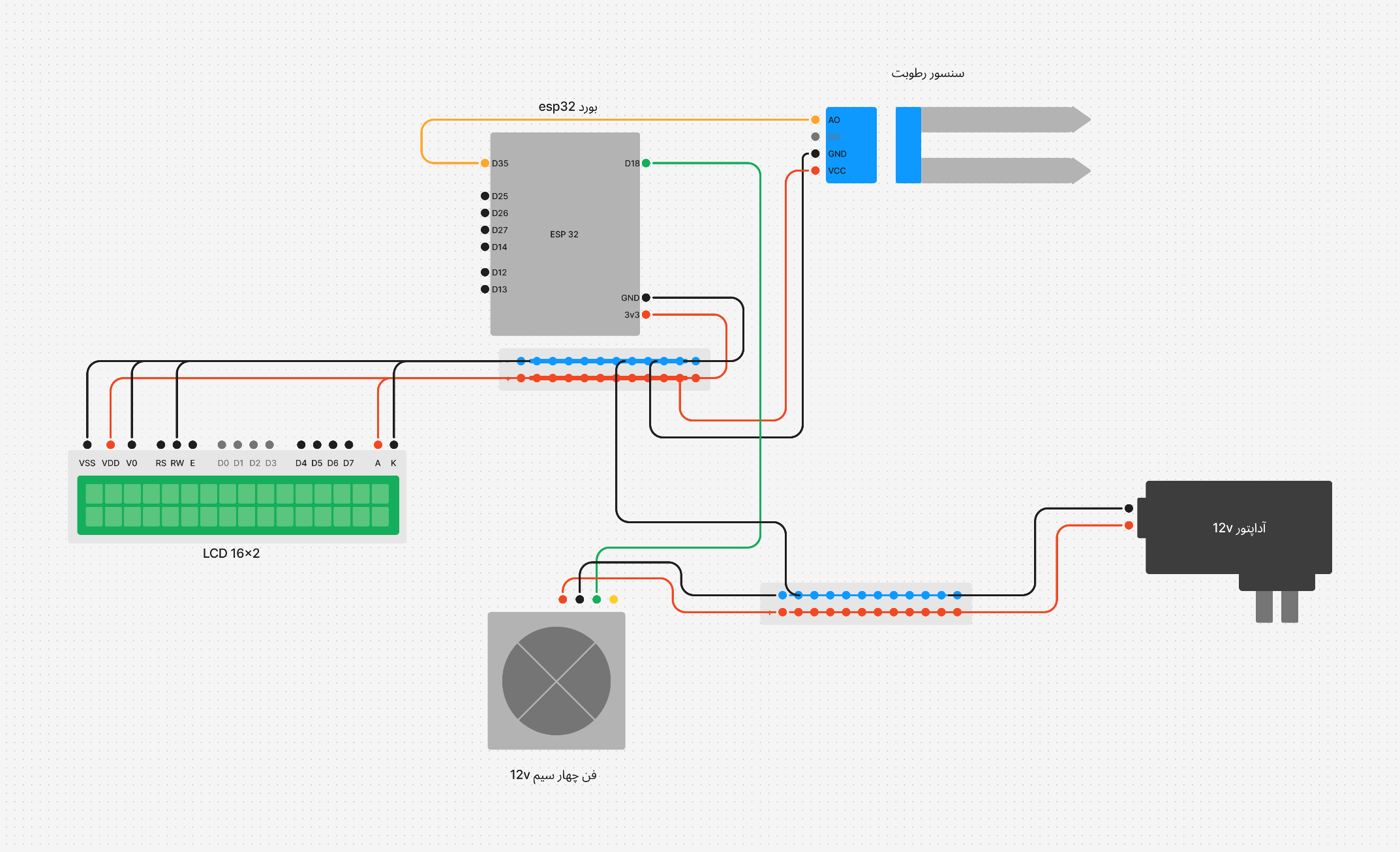
این تابع سرعت واقعی فن رو بر اساس PWM محاسبه میکند. که توی مستندات اصلی آن سرعت آن 12000 دور در دقیقه است که ما بر اساس نسبت PWM اون رو محاسبه میکنیم. البته این مقدار ممکنه با سرعت واقعی چرخش فن متفاوت باشه اما در حالت عادی تقریبا درسته.

# قدم4: نمایش اطلاعات روی LCD

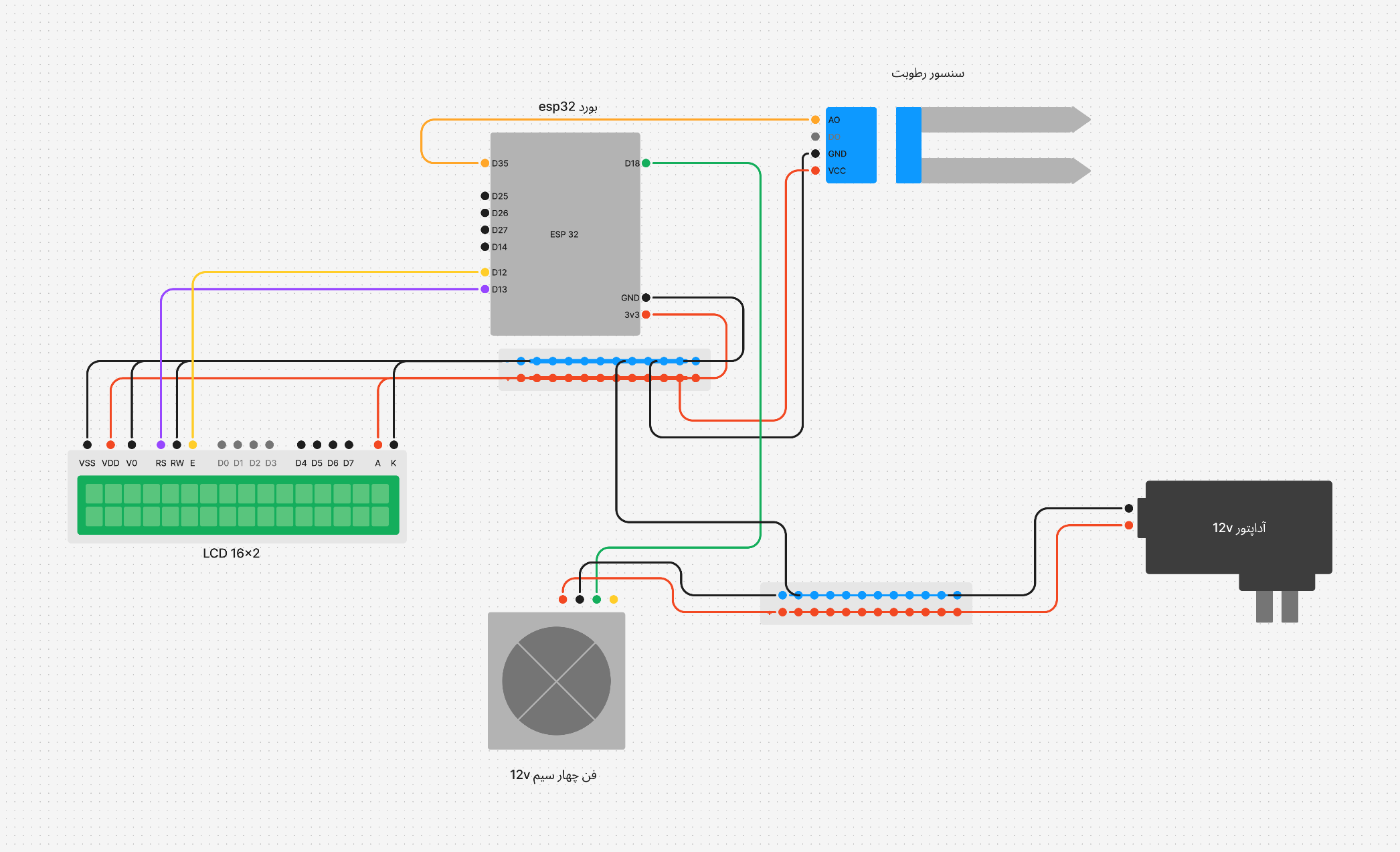
## سیم کشی پایه های مثبت LCD

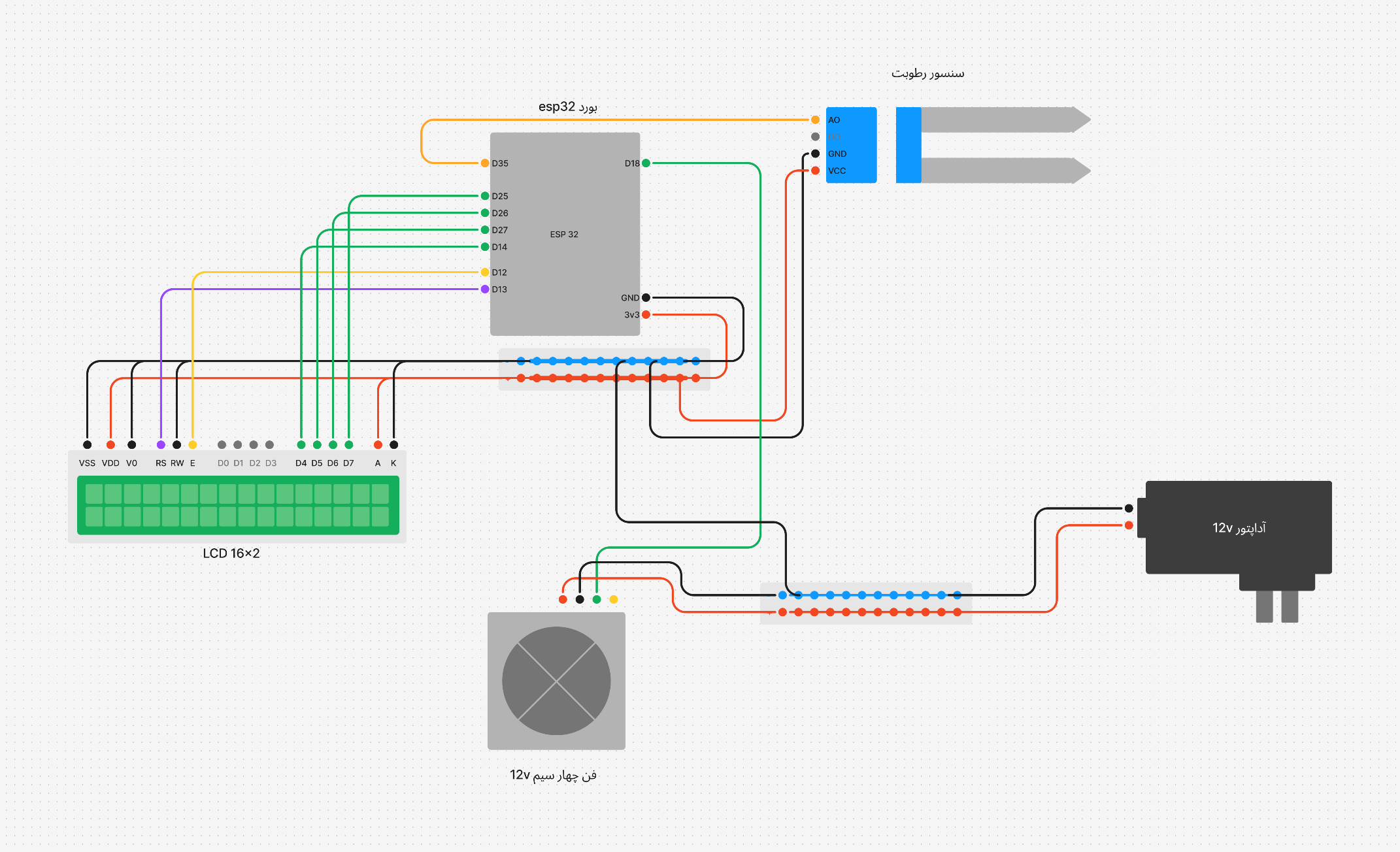


## سیم کشی پایه های منفی LCD



## سیم کشی پایه های متصل به برد





## کد مربوط به کنترل LCD

//اضافه کردن کتابخانه

#include <LiquidCrystal.h>

// پیکربندی پایه‌های LCD

LiquidCrystal lcd(13, 12, 14, 27, 26, 25);

void setup() {

  // تنظیم ابعاد LCD که برای ما

  //2 سطر روی هم هستند

  // و هر سطر 16 خانه دارد.

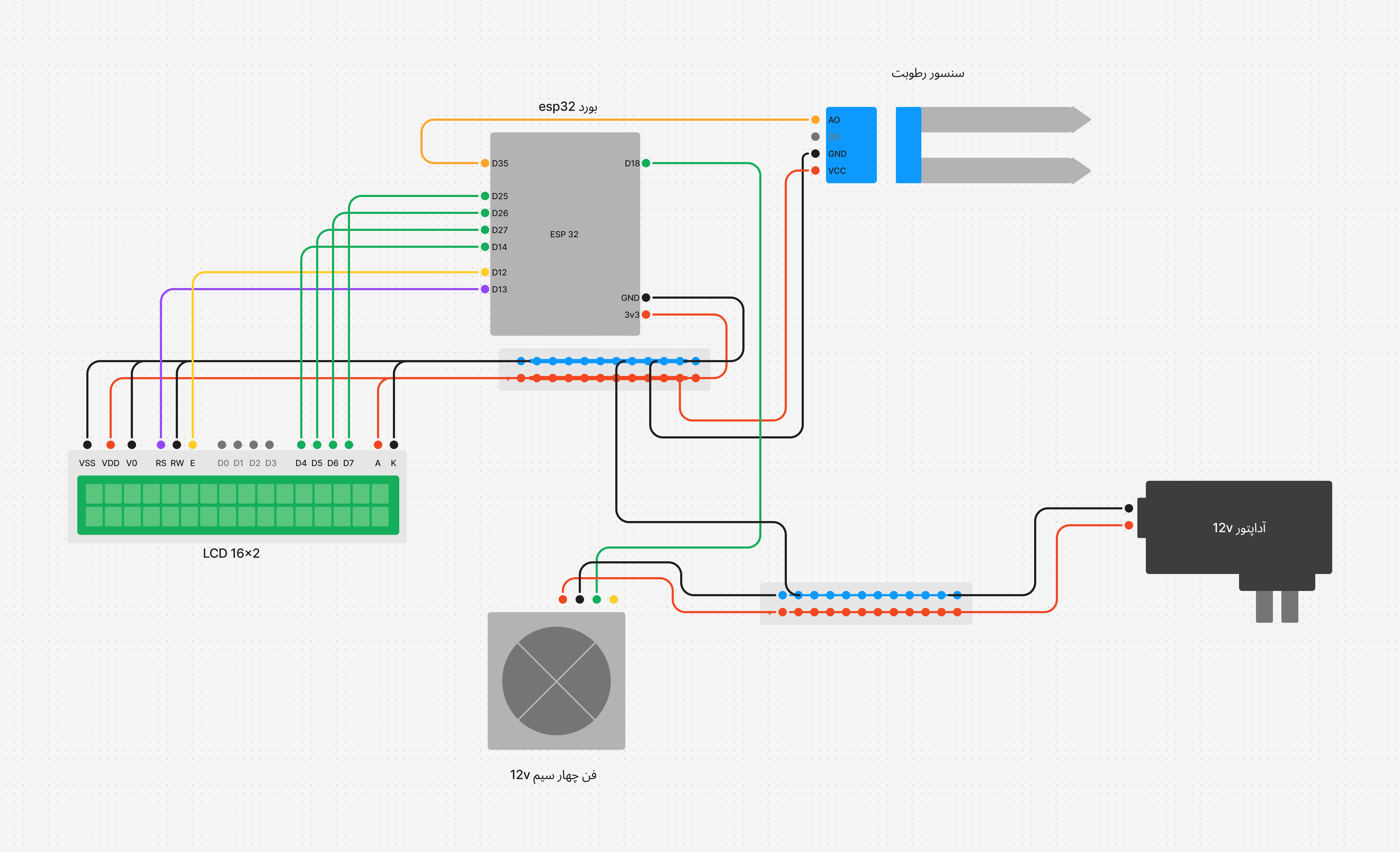
  lcd.begin(16, 2);

}

## پارامتر های LiquidCrystal

هر کدوم از این پارامتر ها مربوط به این پین ها روی LCD میباشند.

LiquidCrystal lcd(RS, E, D4, D5, D6, D7);



## نمایش اطلاعات روی LCD

تا اینجای کار تابع loop به این شیوه درآمده است اول مقدار رطوبت خوانده میشود بعد سرعت فن بر اساس مقدار رطوبت تنظیم میشود و حالا باید سرعت فن و مقدار رطوبت روی LCD نمایش داده شود.

void loop() {

    float soilValue = readSoilMoistureValue();

    Serial.print("Sensor: Soil ");

    Serial.println(soilValue);

    setFanSpeedBasedOnSoilMoisture(soilValue);

    int actualFanSpeed = getActualFanSpeed(curFanSpeed);

    Serial.print("Sensor: Fan speed ");

    Serial.println(actualFanSpeed);

    refreshLCD(soilValue, actualFanSpeed);

    delay(500);

}

## تابع refreshLCD

void refreshLCD(int soilValue, int speed) {

  lcd.clear(); // پاک کردن صفحه LCD

  lcd.setCursor(0, 0); // قرار دادن کرسر در ستون ۰ و سطر ۰ (سطر اول)

  lcd.print("Soil  "); // نمایش متن "Soil  " روی LCD

  lcd.print(soilValue); // نمایش مقدار رطوبت خاک

  lcd.setCursor(0, 1); // قرار دادن کرسر در ستون ۰ و سطر ۱ (سطر دوم)

  lcd.print("RPM  "); // نمایش متن "RPM  " روی LCD

  lcd.print(speed); // نمایش مقدار سرعت فن

}

## ارور ها یا مشکلات LCD که حل کردیم.

1. **نمایشگر فقط مربع نشون میداد** => پایه ولتاژ + به VIN وصل شده باید به 3v3 وصل میشد.
2. **نمایشگر کاراکتر های چرت و پرت چاپ میکرد** => یکی از سیم های اطلاعات D4,D5,D6,D7 به خوبی وصل نشده بود.

# قدم5:ارسال اطلاعات با MQTT

## اتصال به WiFi

برای اتصال به WiFi از لایبرری WiFi استفاده میکنیم.

#include <WiFi.h>

بعد از وارد کردن لایبرری نام و رمز WiFiرا تعریف میکنیم.

const char\* ssid = "Galaxy Note 12";

const char\* password = "12345678";

و حالا با استفاده از تابع begin برنامه شروع به جستجوی شبکه مدنظر میکند و هر زمان در محدوده قرار بگیرد. به ان وصل میشود.

void setup() {

  WiFi.begin(ssid, password);

}

حالا چطور بفهمیم WiFi وصل شده یا ن؟ با استفاده از این کد.

if(WiFi.status() != WL\_CONNECTED){

      Serial.println("MQTT: Wifi is not connected");

      return;

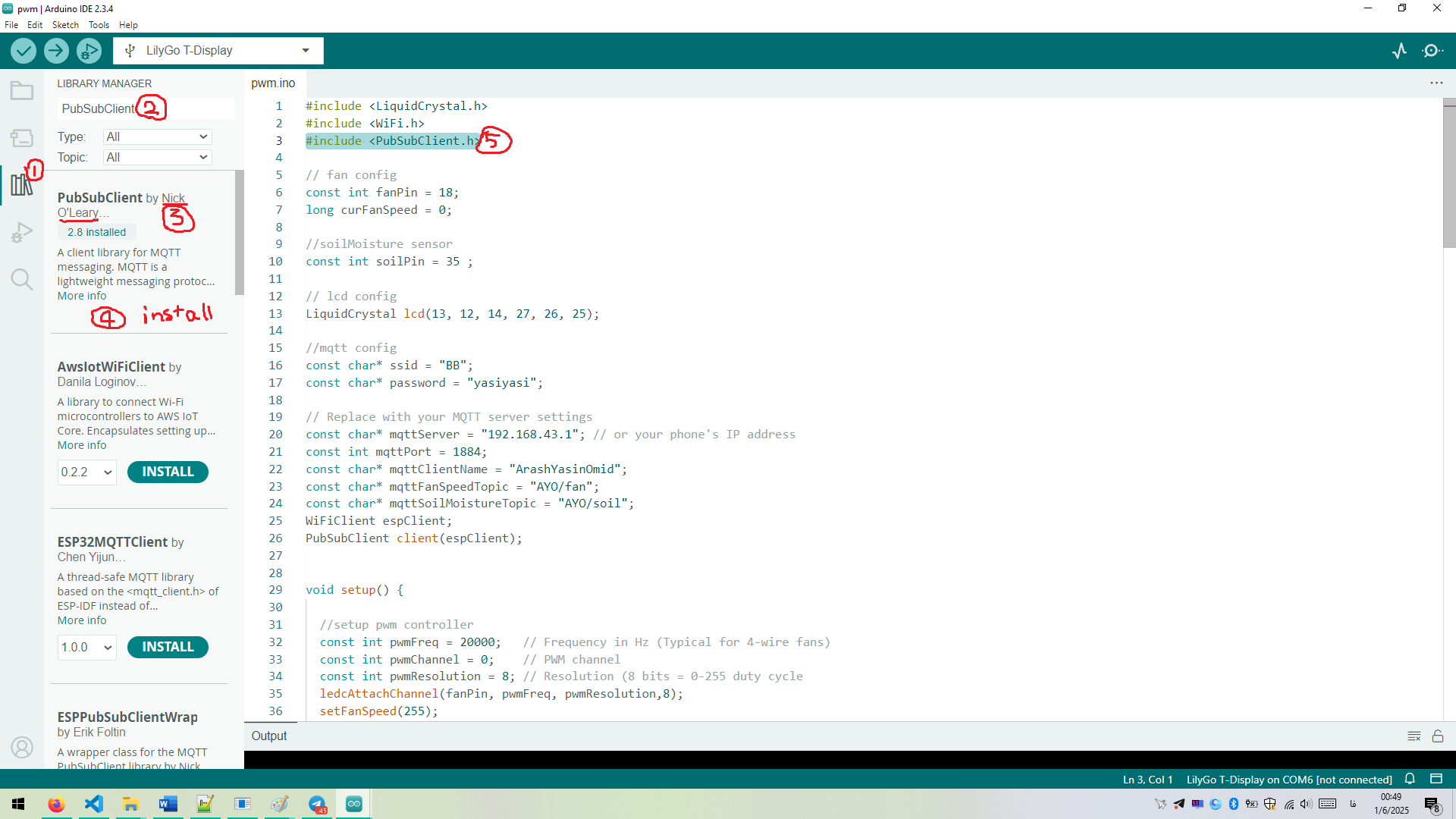
    }

تابع status() وضعیت فعلی اتصال را بر میگرداند که اگر برابر WL\_CONNECTED باشد. یعنی WiFi وصل است

**قطعه کد بالا قطع بودن اتصال رو بررسی میکند که در ادامه از آن نیز استفاده میکنیم.**

## اضافه کردن لایبرری PubSubClient

برای استفاده از پروتکل MQTT ما از لایبرری PubSubClient استفاده میکنیم در ادامه نحوه اضافه کردن آن توضیح داده شده.



اول به قسمت Library Manager میرویم.

اسم کتابخانه PubSubClient رو مینویسم.

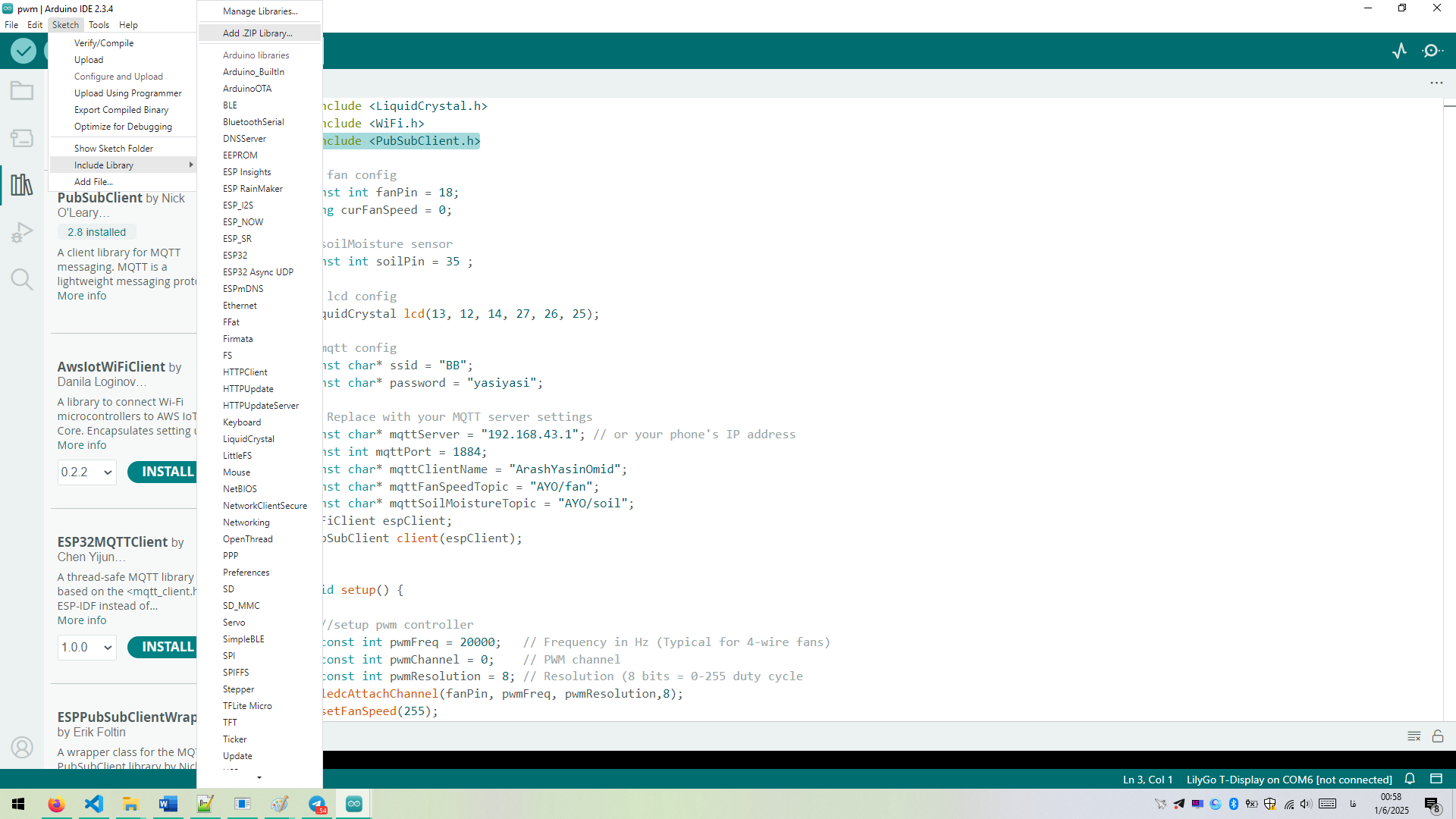
مطمئن میشیم که توسعه دهنده آن Nick O’Leary هست.

و دکمه install رو میزنیم.

بعد میتونیم با دستور include ازش استفاده کنیم.

## چطور فایل zip لایبرری رو وارد برنامه کنیم.

گاهی اوقات لایبرری مورد نظر ما در لیست Library Manger پیدا نمیشود. و به جاش فایل zip انرا از وبسایت اصلی دانلود میکنیم. حالا چطور میتونیم این فایل zip رو وارد Arduino IDE کنیم.



به این مسیر میرویم. Sketch>Include Library>Add .ZIP Library… و بعد در پنجره ای که باز میشه. فایل zip که دانلود کردیم رو میتونیم اضافه کنیم.

## اتصال به سرور یا بروکر MQTT

بعد از اتصال به WiFi این بار نوبت آن میرسد که به سرور متصل بشیم.

اول لایبرری رو وارد برنامه میکنیم.

#include <PubSubClient.h>

حالا اطلاعات اولیه رو تعریف میکنیم.

const char\* mqttServer = "192.168.4.1";

const int mqttPort = 1883;

const char\* mqttClientName = "ArashYasinOmid";

**mqttServer** :

همان ادرس سرور ماست. که در این مورد از IPسرور خود کلاس دانشگاه استفاده کردیم. همچنین میتونیم از test.mosquitto.org هم استفاده کنیم.

**mqttPort**:

معمولا ثابته روی 1883 هست.

**mqttClientName**:

اسمیه که باهاش خومون رو به سرور معرفی میکنیم.

WiFiClient espClient;

PubSubClient client(espClient);

حالا باید یک کلاینت pubSub تعریف کنیم که با استفاده از آن اطلاعات رو به سرور میفرستیم.

و در تابع setup هم اون رو پیکربندی میکنیم.

void setup() {

    client.setServer(mqttServer, mqttPort);

}

حالا بعد از تعریف Client باید از وصل بودن به سرور مطمئن شوییم.

if(!client.connected()){

         client.connect(mqttClientName);

         Serial.println("MQTT: Attempting to connect to broker...");

         return;

    }

این کد چک میکنه که آیا به سرور وصل هستیم که اگر وصل نباشد. سعی میکنه وصل بشه.

**client.connected():**

این تابع وضعیت اتصال رو بر میگردونه.

**client.connect(mqttClientName):**

این تابع سعی میکنه. به سرور وصل بشه.

**در ادامه از این کد استفاده میکنیم.**

## ارسال اطلاعات سنسور ها در MQTT

بعد از اطمینان از اتصال به WiFi و سرور نوبت آن میرسد که اطلاعات سنسور رطوبت و سرعت فعلی فن را به سرور ارسال کنیم برای اینکار ما به topic نیاز داریم **topic ها مسیر های یکتایی بودن که به سنسور مورد نظر اشاره میکردن.**

اول تاپیک هارو تعریف میکنیم. این تاپیک ها بعدا در برنامه MQTT Dash هم استفاده خواهند شد.

const char\* mqttFanSpeedTopic = "AYO/fan";//تایپیک سرعت فن

const char\* mqttSoilMoistureTopic = "AYO/soil";// تاپیک سنسور رطوبت

## تابع publishMqttData

در کد برنامه این تابع برای ارسال اطلاعات سنسور رطوبت و سرعت فن به سرور MQTT استفاده میشود.

void publishMQTTData(int soil, int fanSpeed) {

    // بررسی اتصال به Wi-Fi

    if (WiFi.status() != WL\_CONNECTED) {

      Serial.println("MQTT: Wifi is not connected"); // چاپ پیام خطا اگر Wi-Fi وصل نباشد

      return; // خروج از تابع

    }

    // بررسی اتصال به سرور MQTT

    if (!client.connected()) {

         client.connect(mqttClientName); // تلاش برای اتصال به سرور MQTT

         Serial.println("MQTT: Attempting to connect to broker..."); // چاپ پیام در حال اتصال

         return; // خروج از تابع

    }

    // تبدیل مقادیر سرعت فن و رطوبت خاک به رشته

    char\* fanSpeedPtr = intToCharPtr(fanSpeed); // تبدیل سرعت فن به رشته

    char\* soilSpeedPtr = intToCharPtr(soil); // تبدیل رطوبت خاک به رشته

    // ارسال داده‌ها به سرور MQTT

    client.publish(mqttFanSpeedTopic, fanSpeedPtr); // ارسال سرعت فن

    client.publish(mqttSoilMoistureTopic, soilSpeedPtr); // ارسال رطوبت خاک

    // آزاد کردن حافظه اختصاص داده شده برای رشته‌ها

    free(fanSpeedPtr); // آزاد کردن حافظه سرعت فن

    free(soilSpeedPtr); // آزاد کردن حافظه رطوبت خاک

    // چاپ پیام موفقیت‌آمیز بودن ارسال داده‌ها

    Serial.println("MQTT: data published.");

}

**عملکرد:**

1. اتصال به WiFi رو چک میکنه و اگر وصل نبود از تابع بیرون میاد و به حلقه اصلی برنامه بر میگرده.
2. اتصال به سرور رو چک میکنه. اگر وصل نبود یکبار سعی میکنه وصل بشه. و بعد از تابع بیرون میاد.
3. حالا اگر WiFi وصل بود و سرور هم وصل بود. اطلاعات رو به سرور میفرسته. برای اینکار اول باید متغیر های soil و fanSpeed که از نوع int هستند. به string تبدیل شوند. چون تابع client.publish فقط string رو قبول میکنه. این کار تبدیل int به string با استفاده از تابع IntToCharPtr انجام میشود. و بعد از ارسال اطلاعات تابع free متغیر های string رو از حافظه آزاد میکنه.

## !!!: تابع loop تا اینجای کار

حالا تا اینجای کار تابع loop به این شیوه درآمده است.  
نخست مقدار سنسور رطوبت رو میخونه بر اساس اون سرعت فن رو تنظیم میکنه. و بعد مقدار سنسور رطوبت و سرعت فن رو توی LCD نمایش میده و بعد از اون اونهارو به سرور MQTT ارسال میکنه.

void loop() {

    float soilValue = readSoilMoistureValue();

    Serial.print("Sensor: Soil ");

    Serial.println(soilValue);

    setFanSpeedBasedOnSoilMoisture(soilValue);

    int actualFanSpeed = getActualFanSpeed(curFanSpeed);

    Serial.print("Sensor: Fan speed ");

    Serial.println(actualFanSpeed);

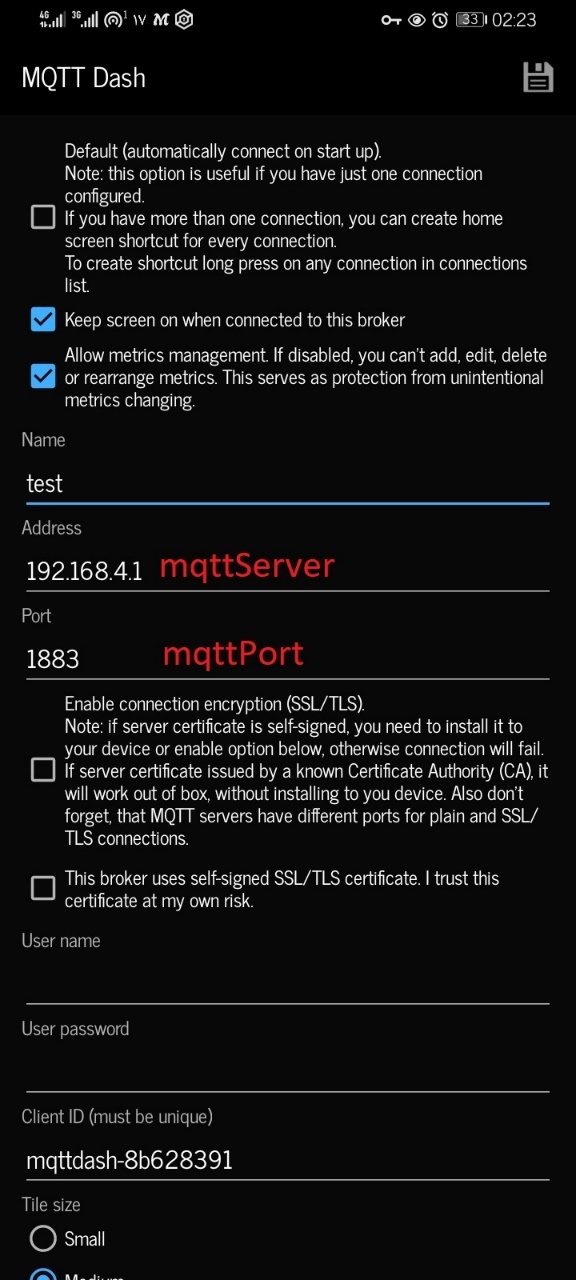
    refreshLCD(soilValue, actualFanSpeed);

    publishMQTTData(soilValue,actualFanSpeed);

    delay(500);

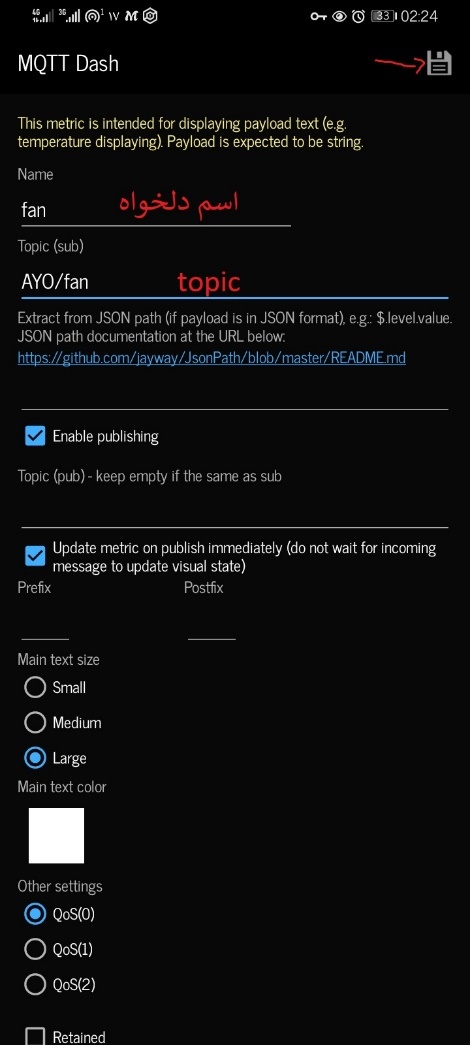
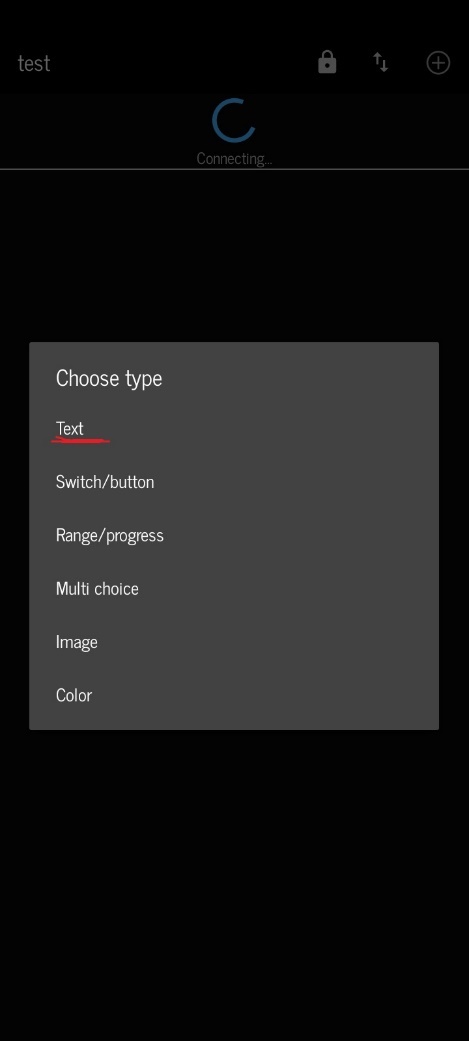
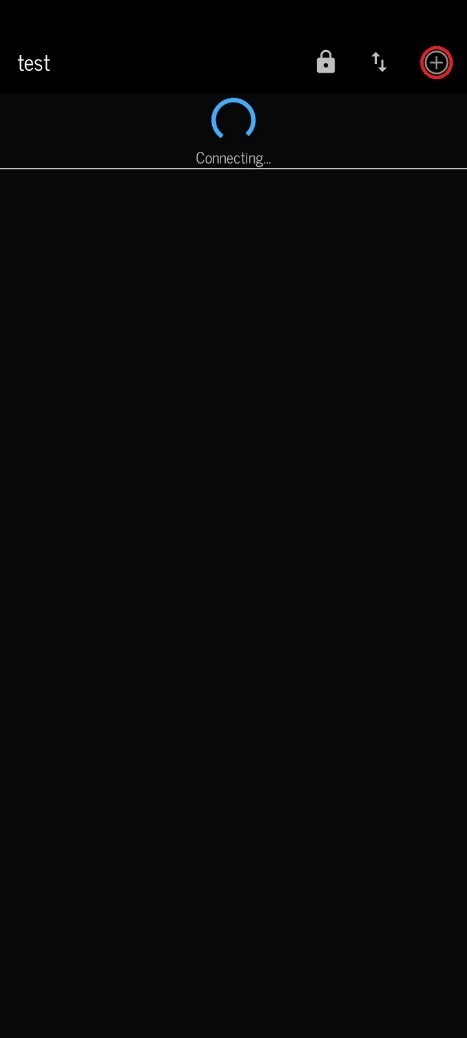
}

## راه اندازی برنامه MQTT Dash



اول ادرس سرور که توی کد مشخص کرده بودیم + پورت موردنظر رو وارد میکنیم.

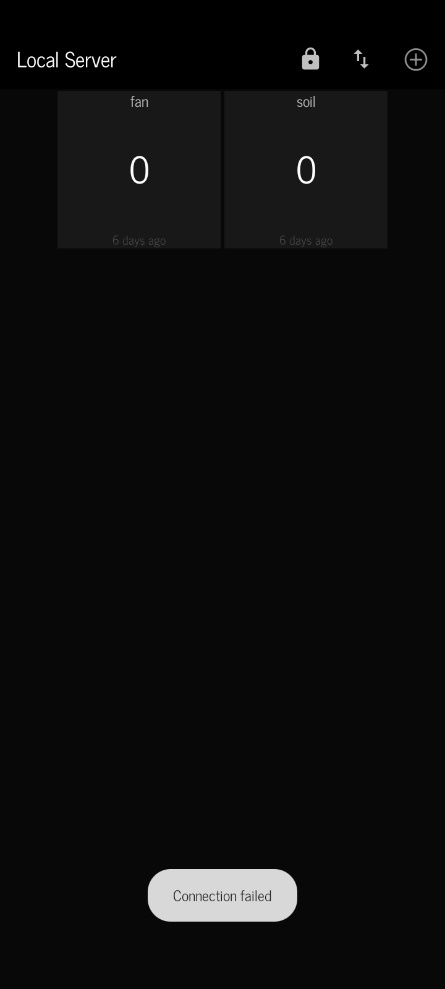
# 



بعد از اتصال به سرور باید یک topic از نوع Text بسازیم و از همون topic که در کد نوشتیم در اینجا هم بنویسیم به دو topic نیاز داریم یکی برای fan و یکی برای soil

const char\* mqttFanSpeedTopic = "AYO/fan";

const char\* mqttSoilMoistureTopic = "AYO/soil";



**با تشکر از توجه شما.**

استاد : **اصغر کریم پور گمش آبادی.**

گروه داشنجویان :

1. یاسین ابراهیم نژادیان
2. امید اعظامی
3. آرش حسین پور

موضوع: **پروژه شماره 1** کنترل سرعت فن بر اساس مقدار سنسور رطوبت و نمایش ان در LCD و سرور MQTT

تاریخ: **1403/10/17**

دانشگاه : **ملی مهارت تبریز**