



# SMART Learning IoT Kit

[www.quadstore.in](http://www.quadstore.in)





# Table of Contents

---

## Introduction

1. Introduction to UNO R4 Board
2. Warranty – Register Now!
3. Understanding Uno R4 board
4. Installing Arduino IDE
5. Connecting the UNO R4 Board
6. Uploading Your First Program (Blinking internal LED Test)

## Projects

1. External LED Blink
2. Traffic Lights with 3 LEDs
3. LED control using Push Button
4. Passive Buzzer Sound
5. Active Buzzer Beep
6. RGB LED Color Mixing
7. Potentiometer-controlled LED Brightness (PWM)
8. LDR Night Lamp
9. Infrared Sensor (IR) Detector
10. Servo Motor Sweep
11. Ultrasonic Distance Finder
12. DHT11 Weather Monitor
13. Smart Gate
14. I2C LCD Display
15. Digital Locker
16. 1-Digit 7 Segment Display
17. Weather Station
18. Led Direction Bar
19. Relay
20. Bluetooth
21. 2WD Car Chassis Assembly
22. Line Following Car
23. Obstacle Avoidance Car
24. Light Following Car
25. Bluetooth Controlled Car
26. Wifi Blynk
27. ESP32 Blinking Led
28. Webserver LED Toggle using ESP32

# Introduction to Uno R4 boards (Must Read)

## About the UNO R4 Lite/Minima Board

The **UNO R4** board is an upgraded version of the previous UNO R4. There are **2 versions** of the Uno R4 board available which is **Minima** and **Wifi**.

## What is difference between UNO R4 Lite & Uno R4 minima?

The board included in Quad Store kit is a compatible Uno R4 Lite which is exact same as the Minima board. Our boards has the exact same processor, larger memory, and uses a USB Type-C connector and expandable I/O ports.

Despite improvements, it is still easy to use for beginners.

So, in short, **Quad Store's Uno R4 Lite = Arduino Uno R4 Minima**.

## What is difference between compatible and original board?

- **Arduino boards are open-source**, so anyone can legally manufacture their own version that works exactly like the original.
- A **compatible board** is a board built to work exactly like the original Arduino, following the same pin layout and functionality.
- Compatible boards (like Quad Store's) offer the same functionality but with upgraded PCB quality and durability.
- They often include **extra expansion** options such as additional I/O pads or improved connectors.
- Compatible boards usually provide better value, with longer warranties, rust-proof coating, and stronger local support

## Why Quad Store Uno R4 Lite board is better than original Arduino Uno R4 Minima board?

- **Superior PCB** construction for improved signal integrity and durability.
- Additional plated-through holes and **expansion pads** for extra analogue/digital/IO connections — ideal for complex DIY builds.
- Protective **rust-proof coating** (conformal finish) to safeguard against humidity and corrosion (especially useful in Indian environments).
- Standard 3-month **warranty** included; register the product on our website and get an additional 3 months – giving you a full **6-month** peace-of-mind.
- **Enhanced customer support**: we provide quicker responses, dedicated documentation, and local spare-parts availability — beyond the generic support offered by many brands

# Warranty: (Register Now!)

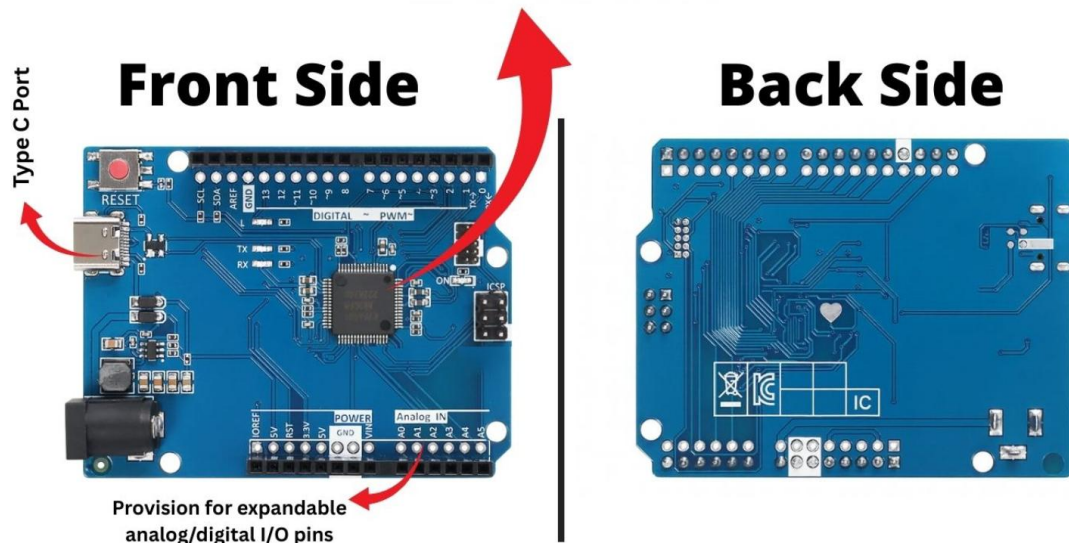
To **claim standard warranty** for Uno R4 Lite boards, **registration** on the Quad Store website is required. We provide a standard **3-month** warranty, and by registering your product online, you receive an additional 3 months—giving you a full **6 months** of peace of mind.

Visit below link and register within **7 days** of purchase of product.

<https://quadstore.in/warranty/>

## Quad Store R4-Lite board

**Comes with Renesas RA4M1 series  
microcontroller**



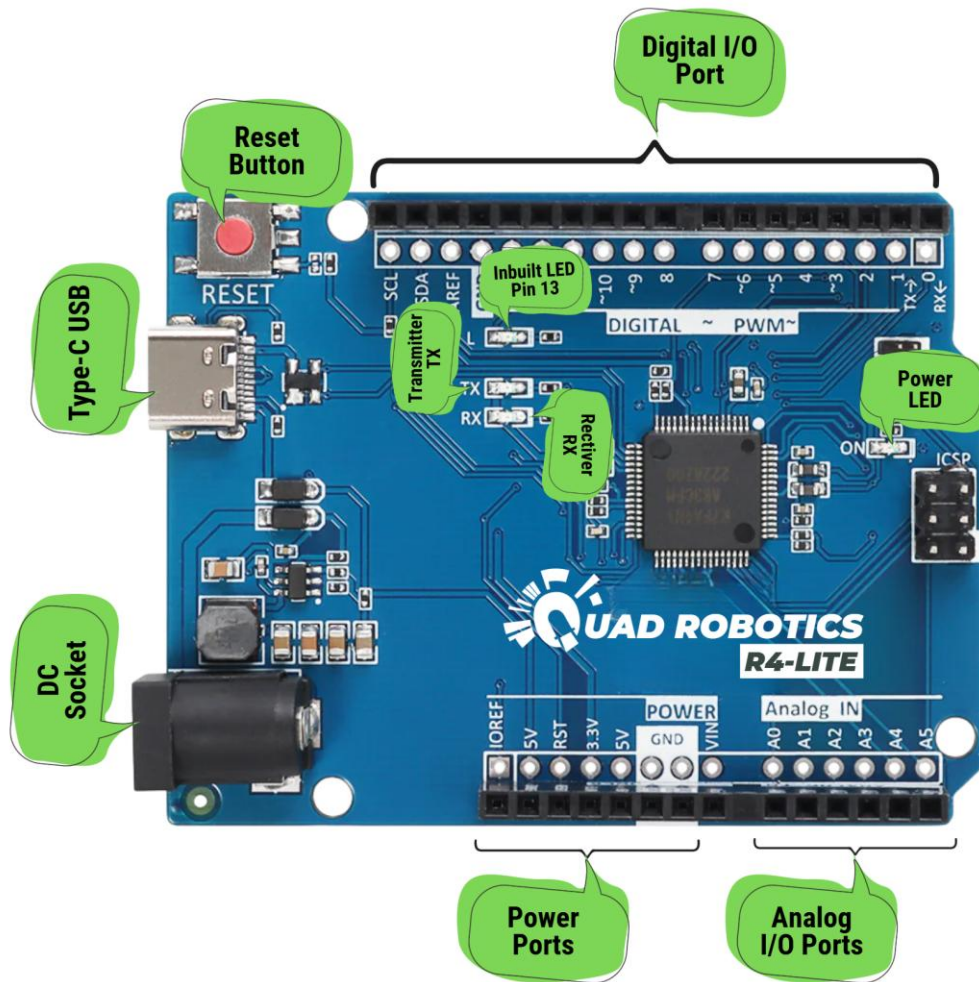
## Our boards are fully compatible with Arduino IDE

- Model: Renesas RA4M1 (RA4M1F44LBA)
- Core: Arm® Cortex®-M4, 32-bit
- Clock speed: 48 MHz
- Flash memory: 256 KB
- SRAM: 32 KB
- Operating voltage: 5V



# Understanding Uno R4 Board:

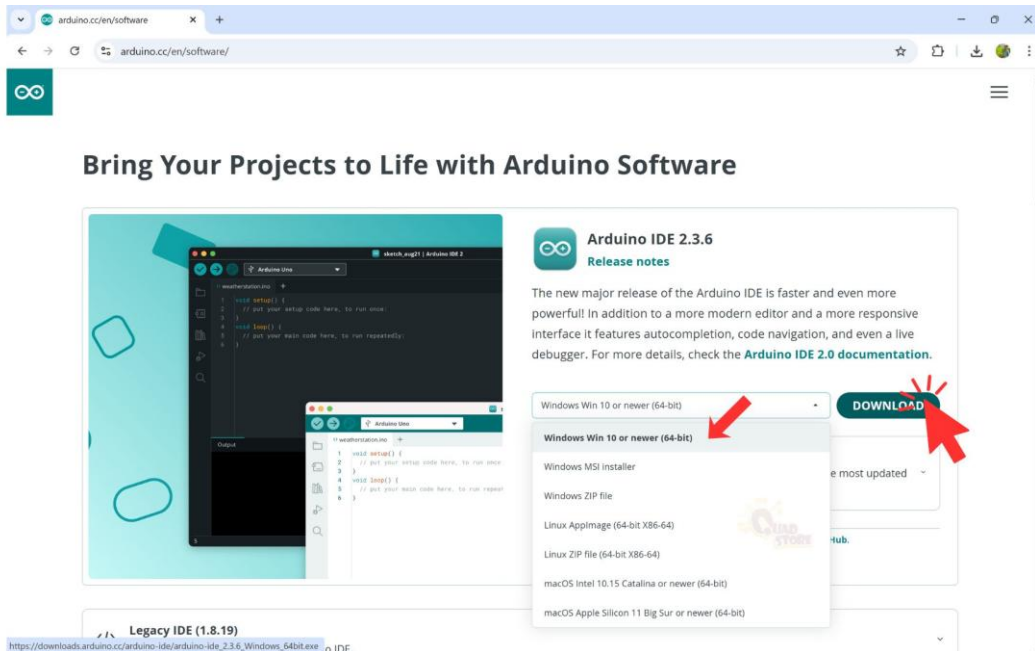
The reference board used in this book is **Quad Store's UNO R4 Lite board**, an Arduino-compatible Uno R4 minima model. A diagram of the Quad Store UNO board is shown below. The board included in your kit **may or may not** carry the **Quad Store / Quad Robotics logo**, as it depends on availability. However, all boards function identically.



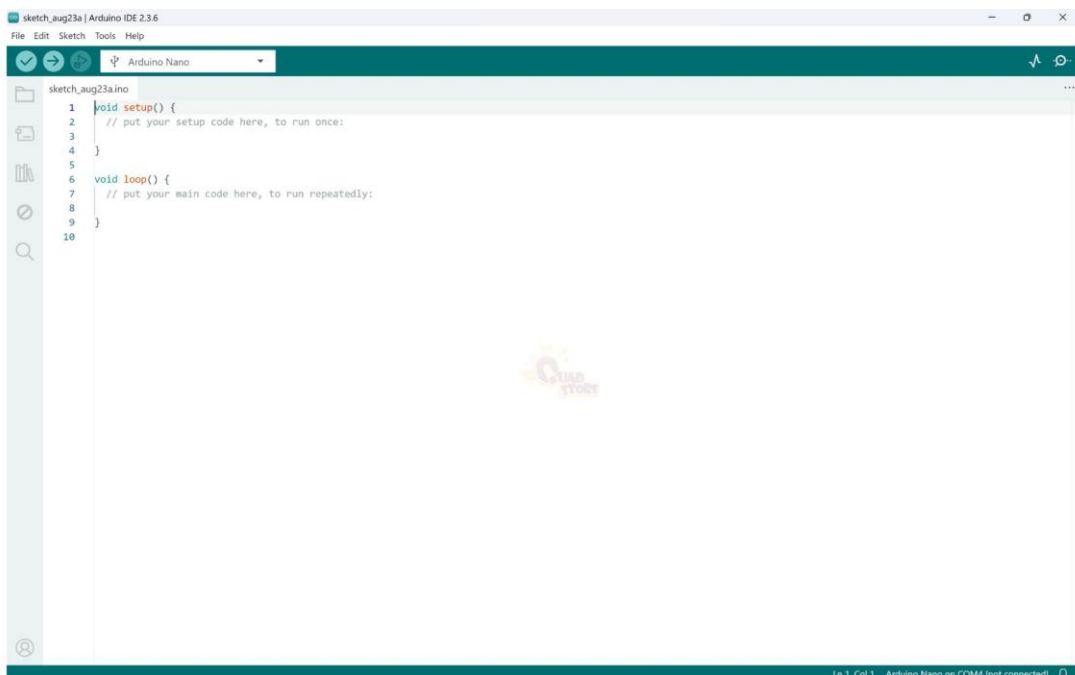
- **Type-C USB:** Used to power the board and upload programs from your computer.
- **Reset Button:** Restarts the board and reloads the running program.
- **DC Socket:** Allows powering the board using an external 7–12V adapter.
- **Digital I/O Port:** Pins used for digital input/output operations, including PWM.
- **Inbuilt LED Pin 13:** Built-in test LED connected to digital pin 13 for quick debugging.
- **Transmitter (TX):** Sends serial data from the board to other devices.
- **Receiver (RX):** Receives serial data into the board from other devices.
- **Power LED:** Indicates that the board is powered ON.
- **Power Ports:** Provide various voltage outputs (5V, 3.3V, GND, VIN) for sensors and modules.
- **Analog I/O Ports:** Pins A0–A5 used for reading analog sensor values.

# Installing Arduino IDE

1. Visit the official Arduino website: <https://www.arduino.cc/en/software>
2. Download and install the Arduino IDE for Windows, Mac, or Linux.

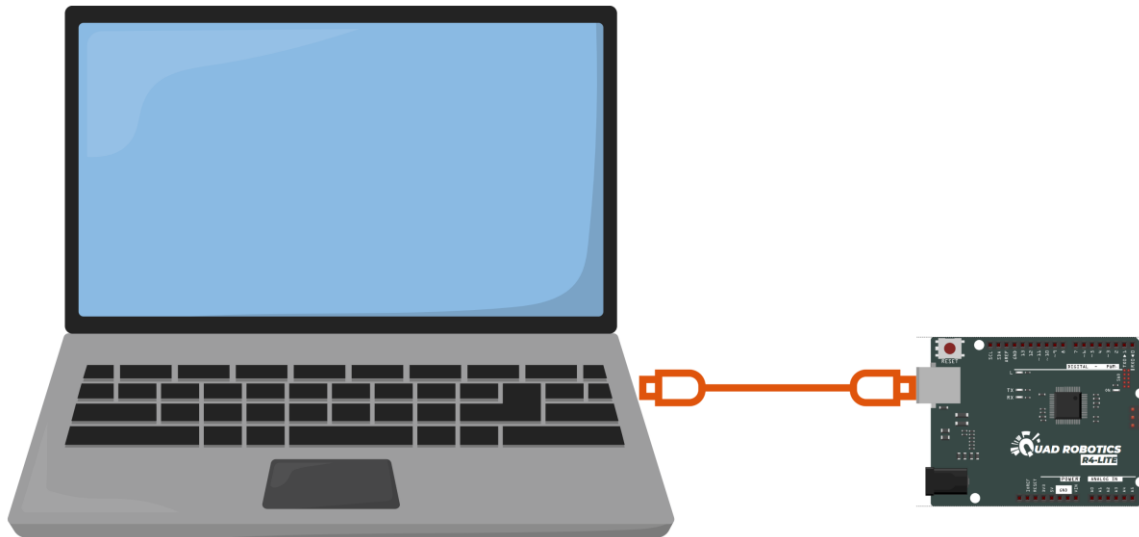


3. Open the IDE by clicking on the desktop icon after installation. IDE should open as shown.

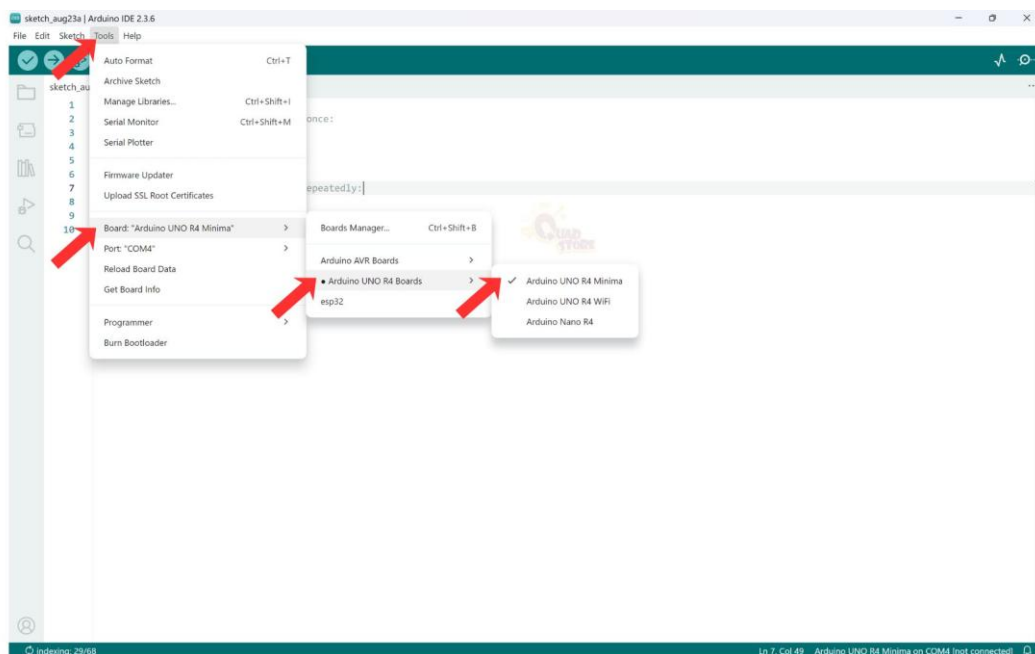


# Connecting the UNO R4 Board

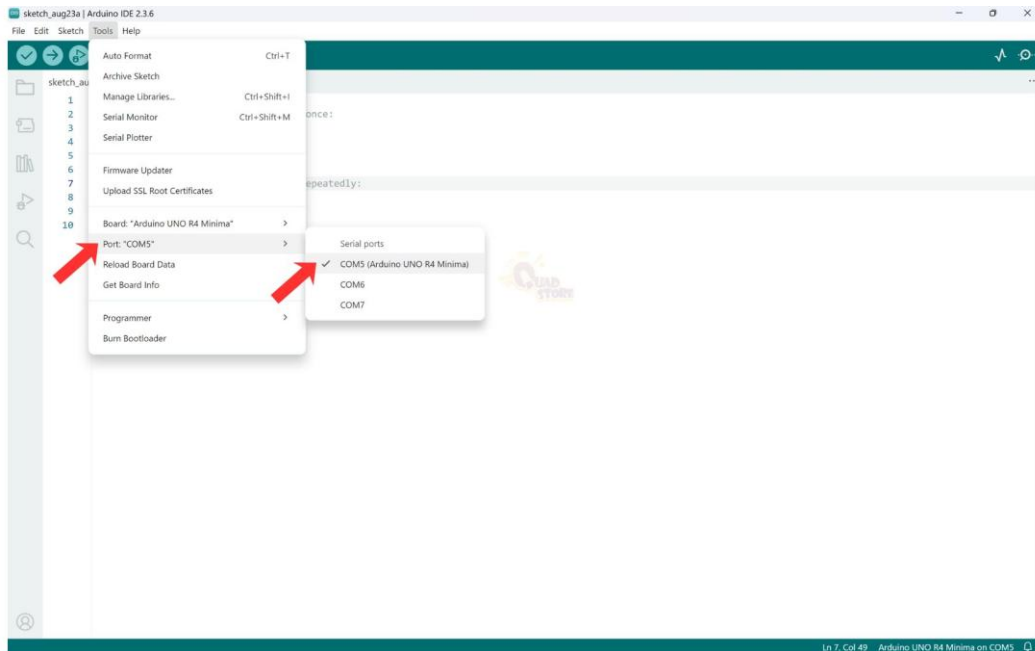
1. Use the Type-C to USB-A cable provided in the kit.
2. Plug the Type-C end into the UNO R4 and the USB-A end into your computer. If your computer or laptop has only a Type-C port, please purchase a Type-C to Type-A USB adapter and use it, or you may use a Type-C to Type-C USB cable instead.



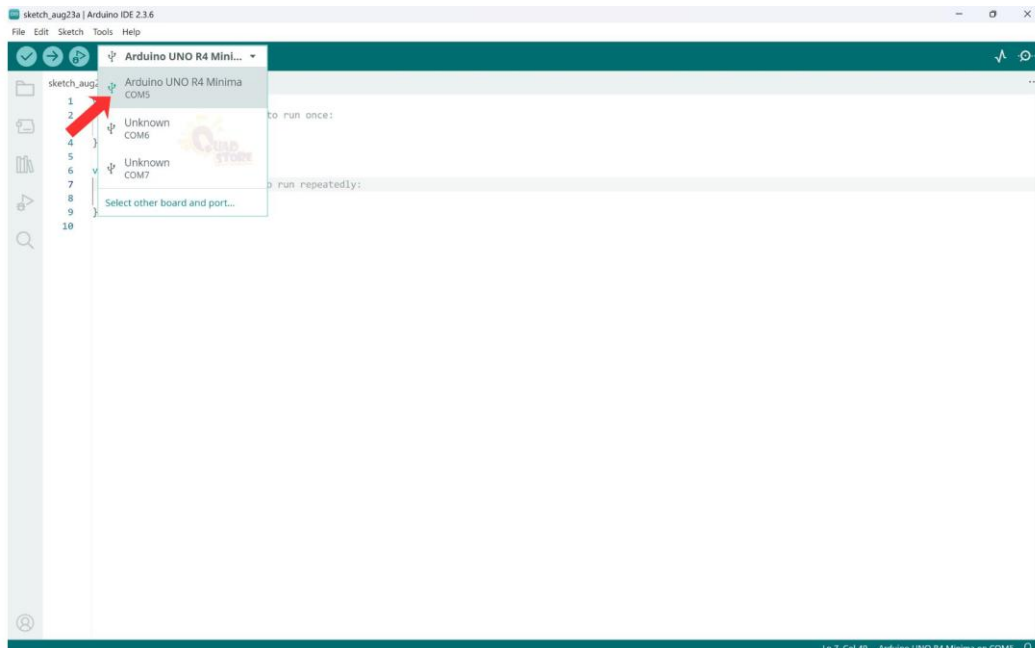
3. The power LED on the board will turn ON.
4. Go to **Tools** → **Board** → **Arduino Uno R4 Board** → Select “**Arduino Uno R4 Minima**” or “**Arduino Uno R4 Wifi**” based on your board. Here you need to select “**Arduino Uno R4 Minima**”.



5. Make sure the correct port is selected by going to **Tools → Port → COM** (Arduino Uno R4 boards) based on your COM port. **NOTE:** COM port number will change based on your computer/laptop.



6. Now you need to select the board. Click the drop-down menu under “Select Board” option. Select the corresponding board. Ensure the correct board is getting reflected in the IDE screen.





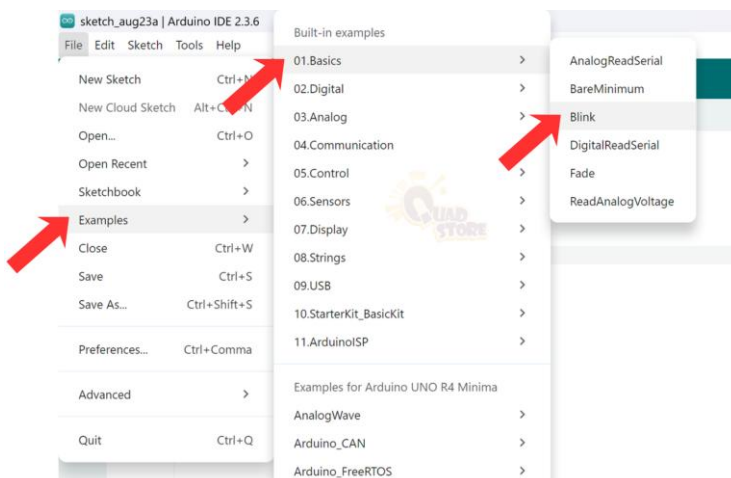
## Uploading Your First Program (Blink Test)

This program makes the built-in LED blink and ensures the board and IDE are working correctly.

```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // Turn LED on
  delay(1000);                     // Wait 1 second
  digitalWrite(LED_BUILTIN, LOW);  // Turn LED off
  delay(1000);                     // Wait 1 second
}
```

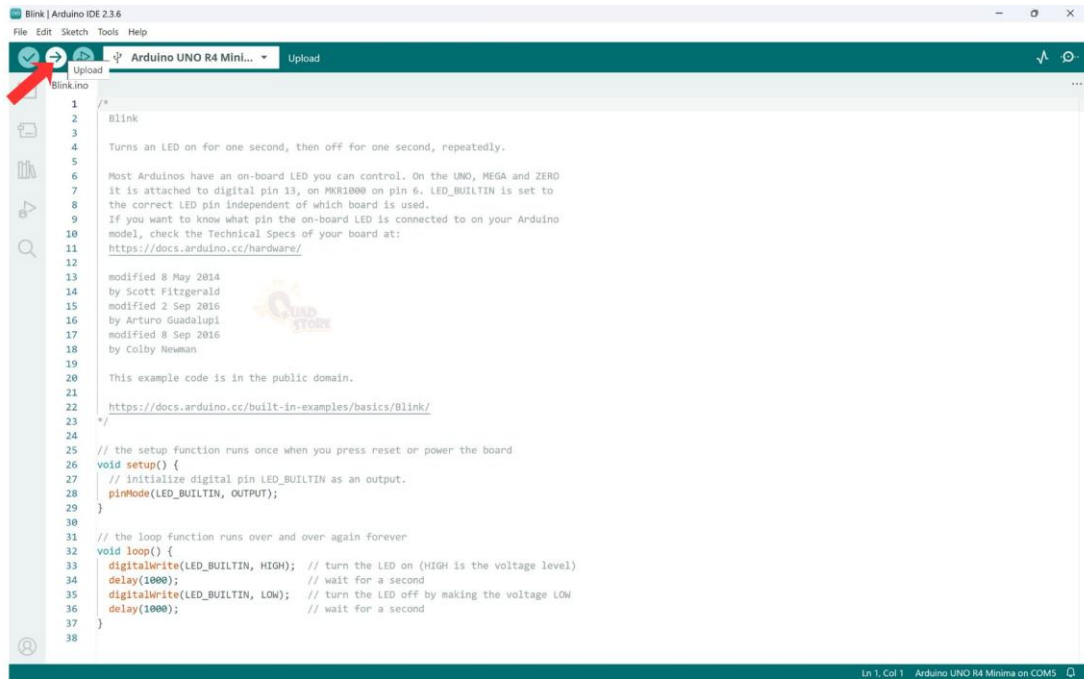
### Steps to Upload Code

1. Either **copy & paste the above code** into the Arduino IDE **(OR)** open **File → Examples → 0.1 Basics → Blink** program.

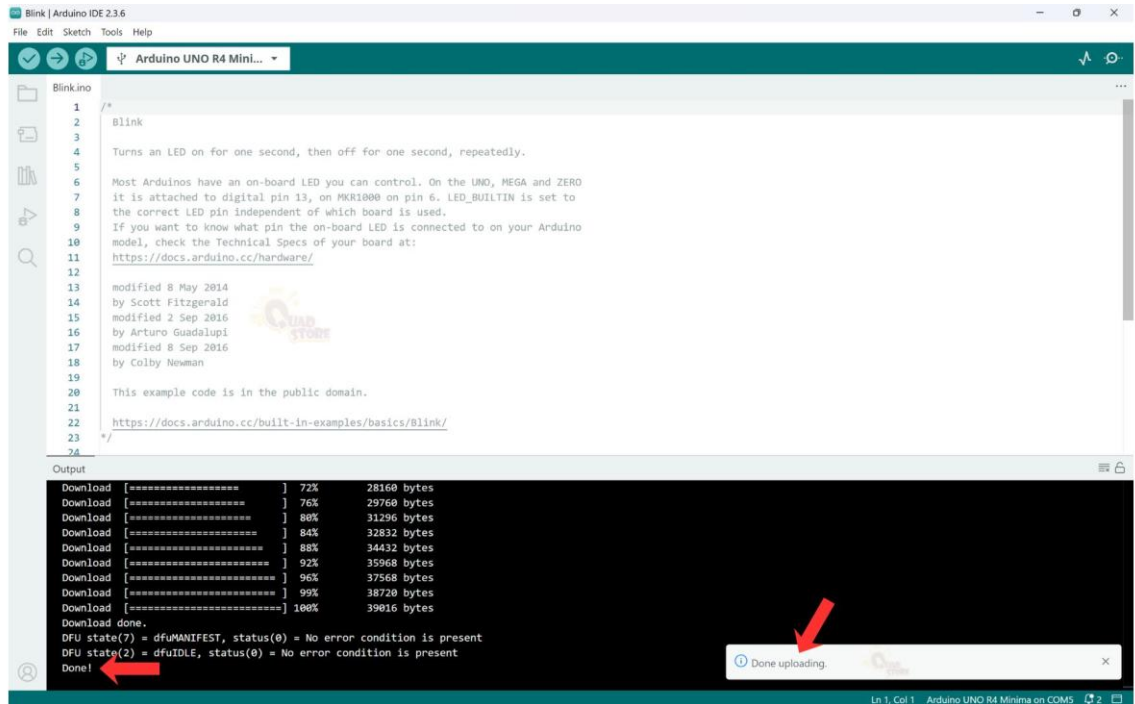




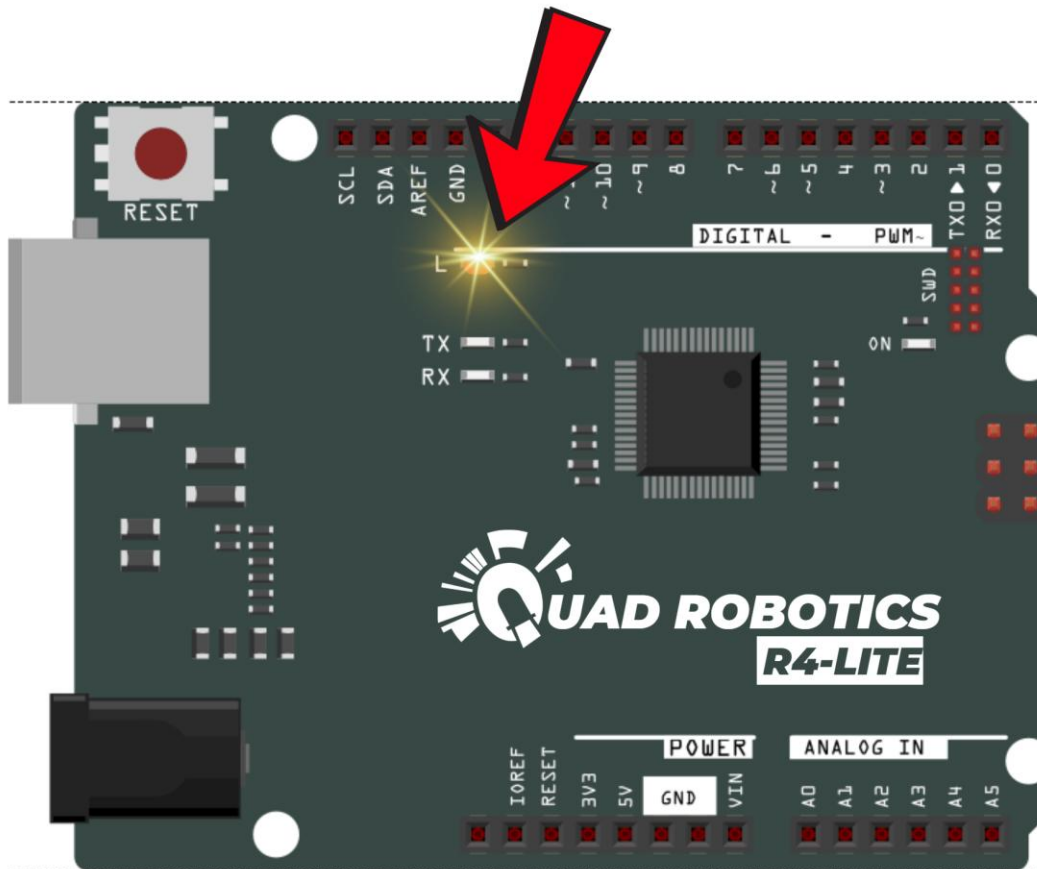
2. Click on the **Upload button** (right arrow icon).



3. Wait for the code to **compile and upload**. You should see a message “Done!” at the bottom of the screen. The board will start running the program automatically.



**Output:** You should see the built-in LED under PIN 13 starts to blink at a regular interval.



# Project 1: Blinking External LED

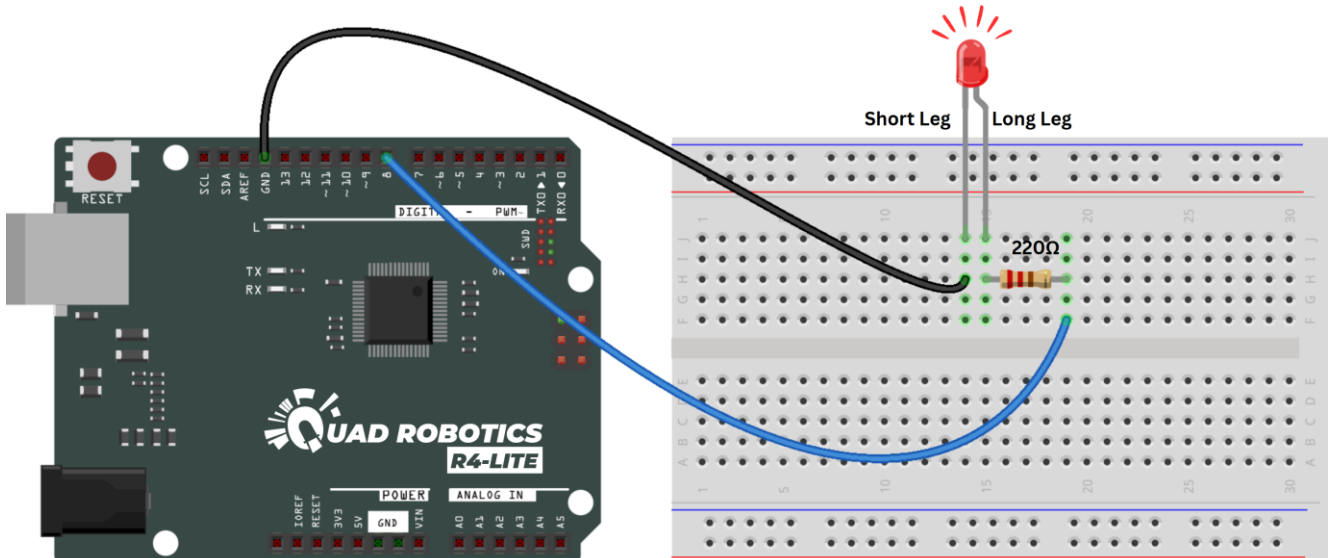
🌟 Objective: Blink an external LED connected on a breadboard.

## 📦 Components Required

- ✓ UNO R4 board
- ✓ Breadboard
- ✓ 1 × LED (any color)
- ✓ 1 × 220Ω Resistor
- ✓ Male to Male Jumper wires
- ✓ Type C to A USB cable

## 💡 Circuit Diagram

Component	Connection Method	Uno R4
LED (Anode, Long Leg + )	Through <b>220Ω</b> resistor	Pin <b>8</b>
LED (Cathode, Short Leg -)	Direct Connection	GND



## 💻 Code

```
int led = 8;

void setup() {
  pinMode(led, OUTPUT);
}

void loop() {
  digitalWrite(led, HIGH);
  delay(1000);
  digitalWrite(led, LOW);
  delay(1000);
}
```

### Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **copy and paste the above code** into the Arduino IDE OR open the file **1.LEDBlink.ino** from the provided **CODE folder**.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as Arduino Uno R4 minima from drop down list.
6. Click on the **Upload button** (right arrow icon).
7. Wait for the code to **compile and upload**. The board will start running the program automatically

### Output

The program turns the LED ON (HIGH) and OFF (LOW) every second.

### DIY Extension

Try different colors of LEDs.

Add more LEDs on pins 9 and 10 for patterns like traffic lights.

# Project 2: Traffic Lights with 3 LEDs

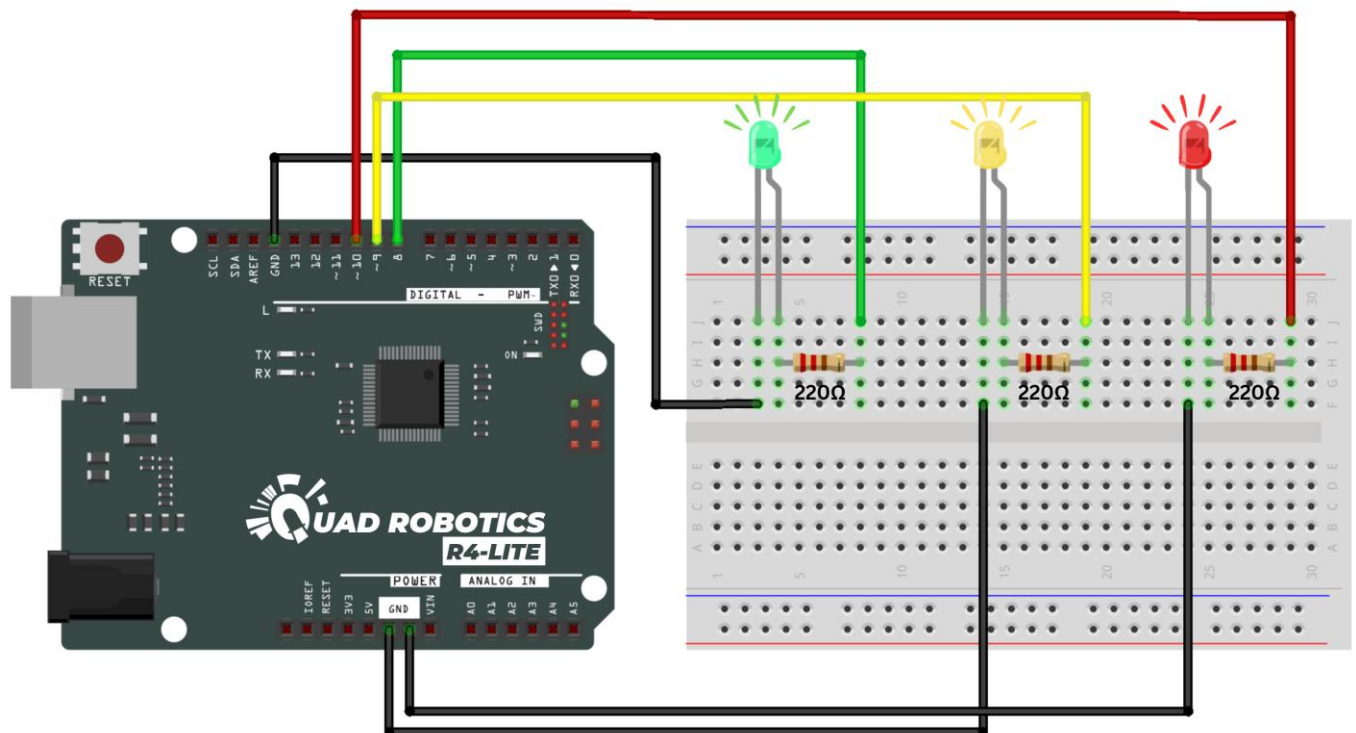
🌟 Objective: Simulate a simple traffic light system using 3 LEDs.

## 📦 Components Required

- ✓ UNO R4 board
- ✓ Breadboard
- ✓ 3 × LEDs (Red, Yellow, Green)
- ✓ 3 × 220Ω Resistors
- ✓ Male to Male Jumper wires
- ✓ Type C to A USB cable

## 💡 Circuit Diagram

Component	Connection Method	Uno R4
Green LED (Anode, Long Leg +)	Through <b>220Ω resistor</b>	Pin 8
Green LED (Cathode, Short Leg -)	Direct Connection	<b>GND</b>
Yellow LED (Anode, Long Leg +)	Through <b>220Ω resistor</b>	Pin 9
Yellow LED (Cathode, Short Leg -)	Direct Connection	<b>GND</b>
Red LED (Anode, Long Leg +)	Through <b>220Ω resistor</b>	Pin 10
Red LED (Cathode, Short Leg -)	Direct Connection	<b>GND</b>





## Code

```
int red = 10;
int yellow = 9;
int green = 8;

void setup() {
  pinMode(red, OUTPUT);
  pinMode(yellow, OUTPUT);
  pinMode(green, OUTPUT);
}
void loop() {
  digitalWrite(red, HIGH);
  delay(3000);
  digitalWrite(red, LOW);

  digitalWrite(yellow, HIGH);
  delay(1000);
  digitalWrite(yellow, LOW);

  digitalWrite(green, HIGH);
  delay(3000);
  digitalWrite(green, LOW);
}
```

## Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **copy and paste above code** into the Arduino IDE OR open the file **2.TrafficLights.ino** from the **CODE folder**.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as Arduino Uno R4 minima from drop down list.
6. Click on the **Upload button** (right arrow icon).
7. Wait for the code to **compile and upload**. The board will start running the program automatically.

## Output

Red stays ON for 3 seconds, Yellow for 1 second, and Green for 3 seconds.

## DIY Extension

Add a pedestrian button to change the lights when pressed.

Use a buzzer for sound alerts.

# Project 3: LED control using Push Button

🌟 Objective: To control an **LED** using a **push button** connected to an **UNO R4**.

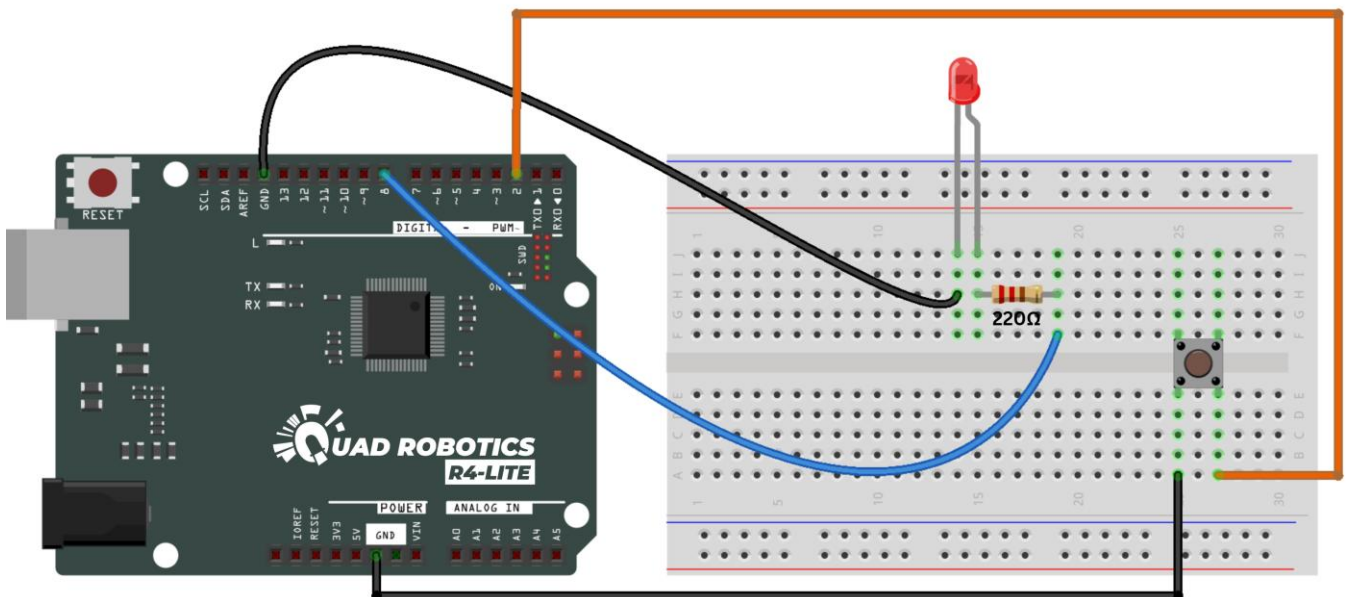
- When button is pressed → LED turns ON
- When button is released → LED turns OFF

## 🛒 Components Required

- ✓ UNO R4 board
- ✓ Breadboard
- ✓ 1 × LED (Any color of your choice)
- ✓ 1 × 220Ω Resistors
- ✓ 1 × Push Button Switch
- ✓ Male to Male Jumper wires

## 💡 Circuit Diagram

Component	Connection Method	Uno R4
LED (Anode, Long Leg +)	Through <b>220Ω resistor</b>	Pin 6
LED (Cathode, Short Leg -)	Direct Connection	<b>GND</b>
Push Button	One side	Pin 2
Push Button	Other side	<b>GND</b>



## Code

```
const int buttonPin = 2;
const int ledPin = 8;

void setup() {
  pinMode(buttonPin, INPUT_PULLUP); // internal pull-up
  pinMode(ledPin, OUTPUT);
}

void loop() {
  int buttonState = digitalRead(buttonPin);

  if (buttonState == LOW) {
    // button pressed
    digitalWrite(ledPin, HIGH);
  } else {
    digitalWrite(ledPin, LOW);
  }
}
```

### Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **copy and paste above code** into the Arduino IDE OR open the file **3.PushButtonLED.ino** from the provided **CODE folder**.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as Arduino Uno R4 minima from drop down list.
6. Click on the **Upload button** (right arrow icon).
7. Wait for the code to **compile and upload**. The board will start running the program automatically.

## Output

- LED stays OFF initially
- Press the button → LED turns ON
- Release the button → LED turns OFF.

## DIY Extension

### Toggle LED mode

Press button once → LED ON

Press again → LED OFF

# Project 4: Passive Buzzer Sound

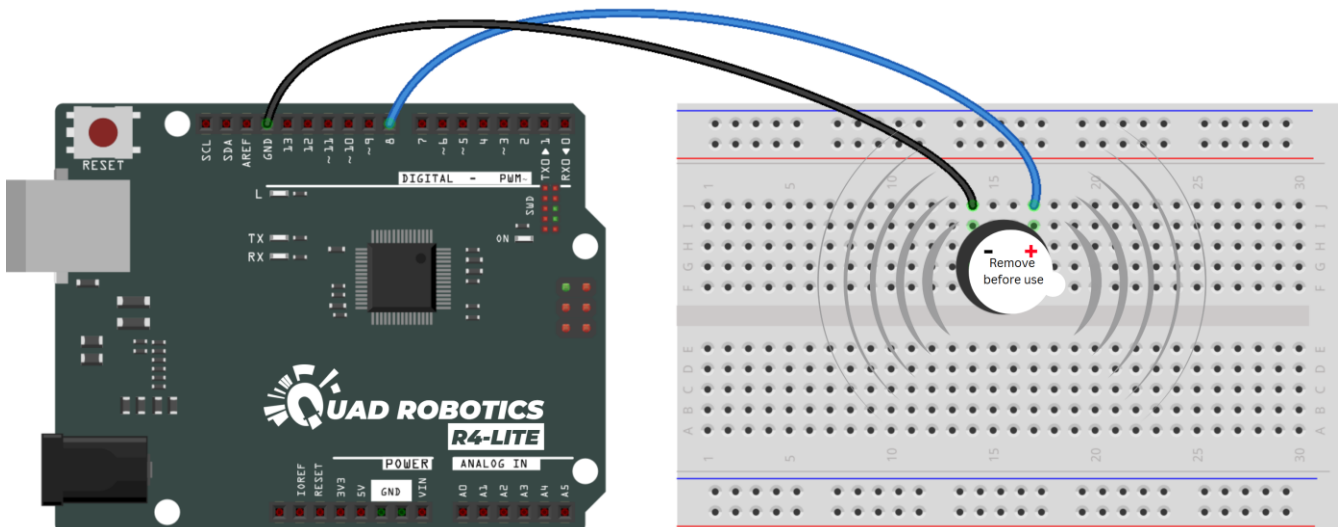
🌟 Objective: Generate sound using a passive buzzer with tone function.

## 🧰 Components Required

- ✓ UNO R4 board
- ✓ Passive Buzzer (remove the top sticker on the buzzer before use)
- ✓ Male to Male Jumper wires
- ✓ Breadboard

## 🔌 Circuit Diagram

Component	Connection Method	Uno R4
Buzzer (Positive +)	Direct Connection	Pin 8
Buzzer (Negative -)	Direct Connection	GND



## 💻 Code

```
int buzzer = 8;

void setup() {
}

void loop() {
  tone(buzzer, 1000); // 1kHz tone
  delay(1000);
  noTone(buzzer);
  delay(1000);
}
```

### Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **copy and paste above code** into the Arduino IDE OR open the file **4.PassiveBuzzer.ino** from the provided **CODE folder**.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as Arduino Uno R4 minima from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

### Explanation & Output

The passive buzzer produces sound when driven with different frequencies.

Here we generate a 1kHz tone for 1 second ON and 1 second OFF.

### DIY Extension

Change frequency for different musical notes.

Try making a melody.



# Project 5: Active Buzzer Beep

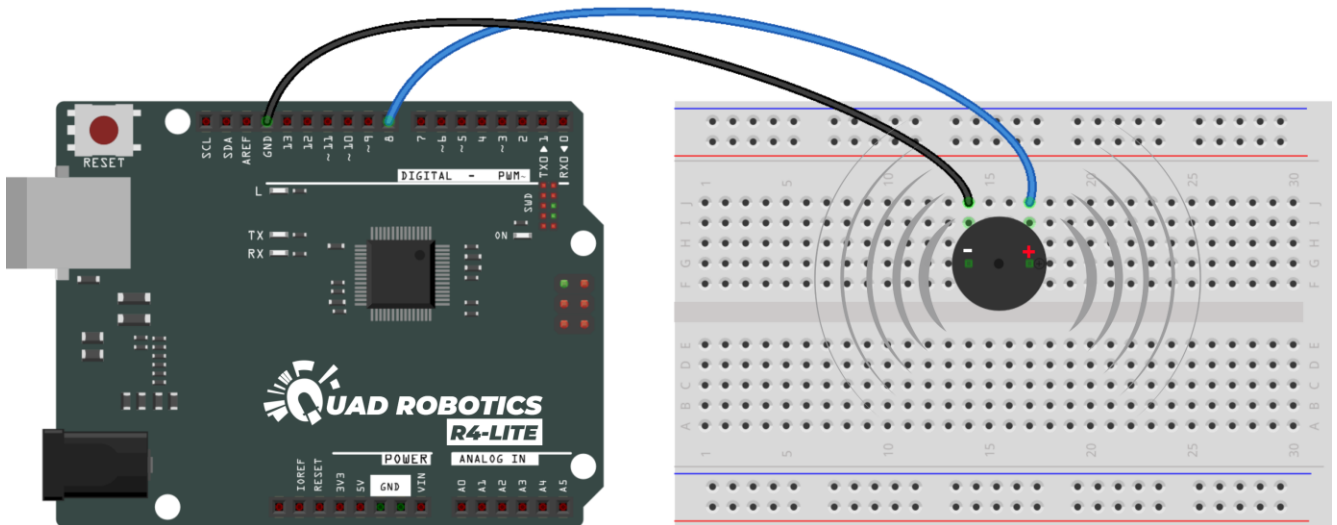
🌟 Objective: Turn an active buzzer ON and OFF.

## 📦 Components Required

- ✓ UNO R4 board
- ✓ Active Buzzer
- ✓ Male to Mae Jumper wires
- ✓ Breadboard

## 🔌 Circuit Diagram

Component	Connection Method	Uno R4
Buzzer (Positive +)	Direct Connection	Pin 8
Buzzer (Negative -)	Direct Connection	GND



## 💻 Code

```
int buzzer = 8;

void setup() {
  pinMode(buzzer, OUTPUT);
}

void loop() {
  digitalWrite(buzzer, HIGH);
  delay(1000);
  digitalWrite(buzzer, LOW);
  delay(1000);
}
```

### Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **5.ActiveBuzzer.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

### Explanation & Output

An active buzzer produces a fixed beep when powered.

We simply switch it ON and OFF every second.

### DIY Extension

Use buzzer for alarms in sensor projects.

# Project 6: RGB LED Color Mixing

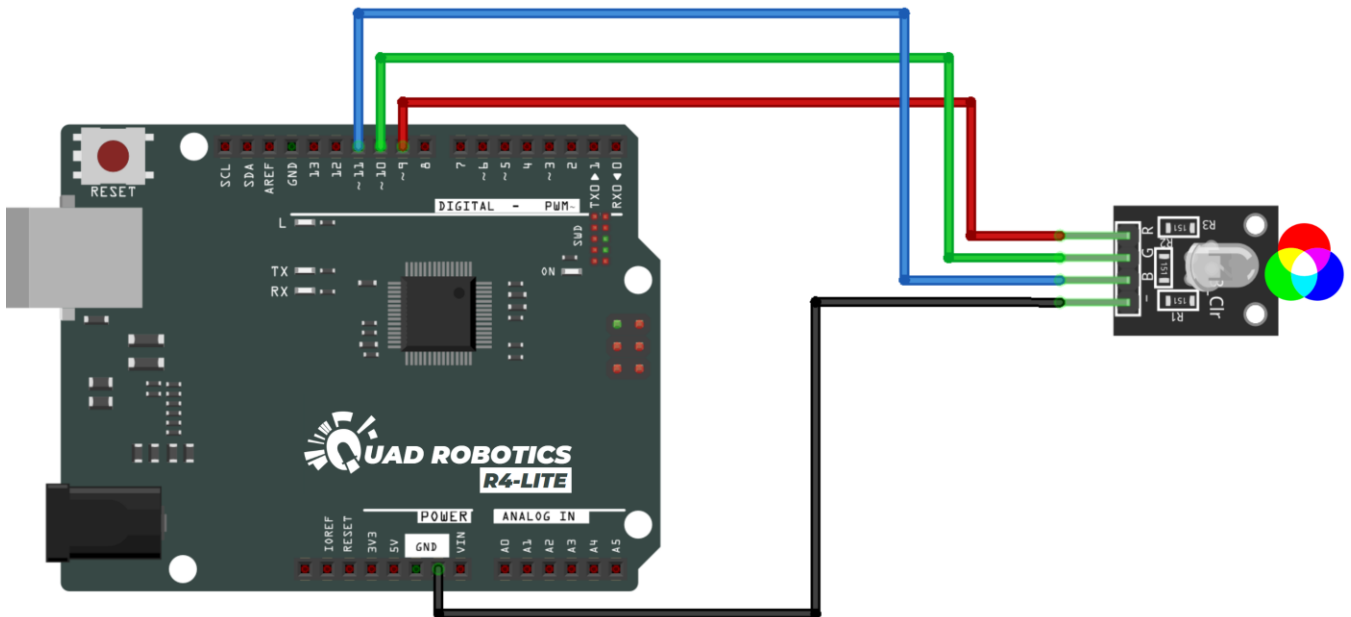
🌟 Objective: Control an RGB LED to show different colors.

## 📦 Components Required

- ✓ UNO R4 board
- ✓ RGB LED Module
- ✓ Male to Female Jumper wires. A breadboard is not required when using these wires, as you can directly connect the RGB module to the female end.

## 🔌 Circuit Diagram

Component	Connection Method	Uno R4
RGB Module – R (Red)	Direct Connection	Pin 9
RGB Module – G (Green)	Direct Connection	Pin 10
RGB Module – B (Blue)	Direct Connection	Pin 11
RGB Module – GND	Direct Connection	GND



## Code

```
int red = 9;
int green = 10;
int blue = 11;

void setup() {
  pinMode(red, OUTPUT);
  pinMode(green, OUTPUT);
  pinMode(blue, OUTPUT);
}

void loop() {
  digitalWrite(red, HIGH);
  digitalWrite(green, LOW);
  digitalWrite(blue, LOW);
  delay(1000);

  digitalWrite(red, LOW);
  digitalWrite(green, HIGH);
  delay(1000);

  digitalWrite(green, LOW);
  digitalWrite(blue, HIGH);
  delay(1000);
}
```

## Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **6.RGBLed.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

## Explanation & Output

An RGB LED combines red, green, and blue light to produce colors. We switch between red, green, and blue every second.

## DIY Extension

Mix multiple colors by turning ON two pins together.  
Use analogWrite for smooth fading.

# Project 7: Potentiometer-controlled LED Brightness

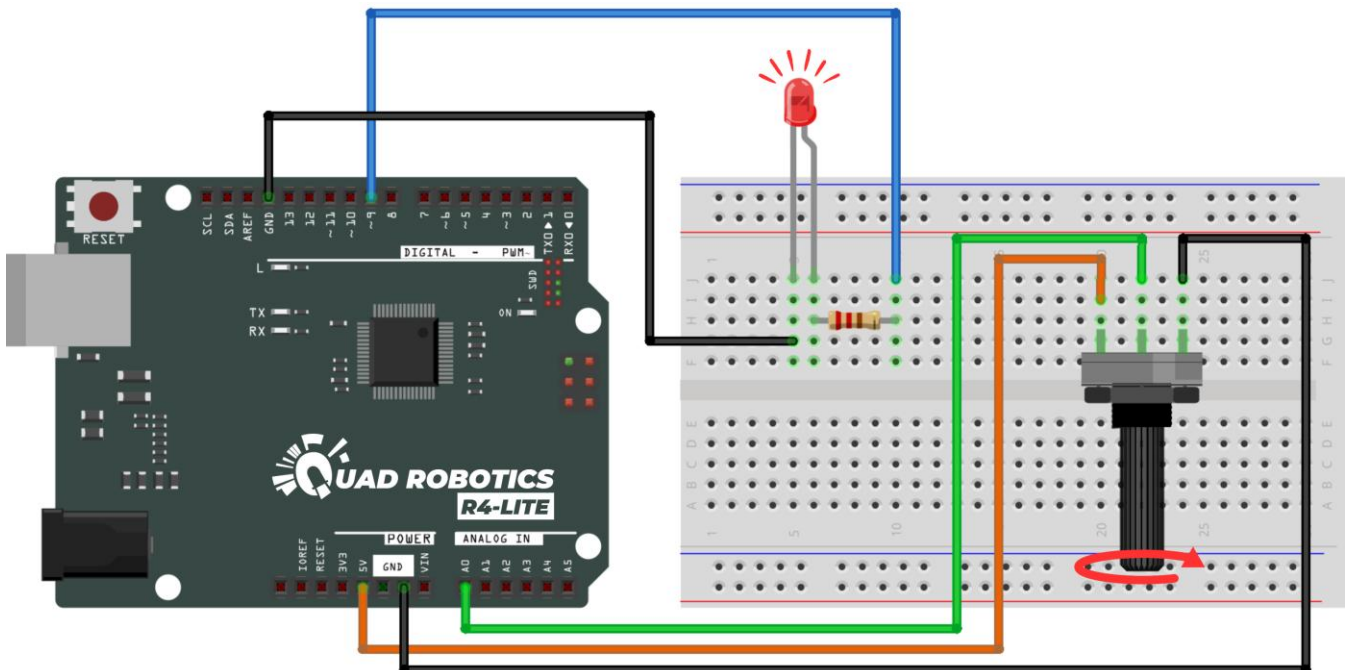
🌟 Objective: Use a potentiometer to adjust LED brightness (PWM).

## 🧰 Components Required

- ✓ UNO R4 board
- ✓ 10K Potentiometer
- ✓ LED
- ✓ 220Ω Resistor
- ✓ Breadboard
- ✓ Male to Male Jumper wires

## 💡 Circuit Diagram

Component	Connection Method	Uno R4
Potentiometer (Middle Pin / Wiper)	Direct Connection	A0
Potentiometer (Left Pin)	Direct Connection	5V
Potentiometer (Right Pin)	Direct Connection	GND
LED (Anode, Long Leg +)	Through <b>220Ω resistor</b>	Pin 9
LED (Cathode, Short Leg -)	Direct Connection	GND





## Code

```
int potPin = A0;
int led = 9;
int val = 0;

void setup() {
  pinMode(led, OUTPUT);
}

void loop() {
  val = analogRead(potPin);
  val = map(val, 0, 1023, 0, 255);
  analogWrite(led, val);
}
```

## Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **7.PotLED.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

## Explanation & Output

Potentiometer provides analog input from 0–1023.

We map it to 0–255 to control LED brightness with PWM.

Output: Turn the potentiometer knob from left to right to see the LED brightness change from low to high.

## DIY Extension

Try controlling buzzer pitch with potentiometer.

# Project 8: Photoresistor/ Light Dependent Resistor (LDR) or Automatic Night Lamp

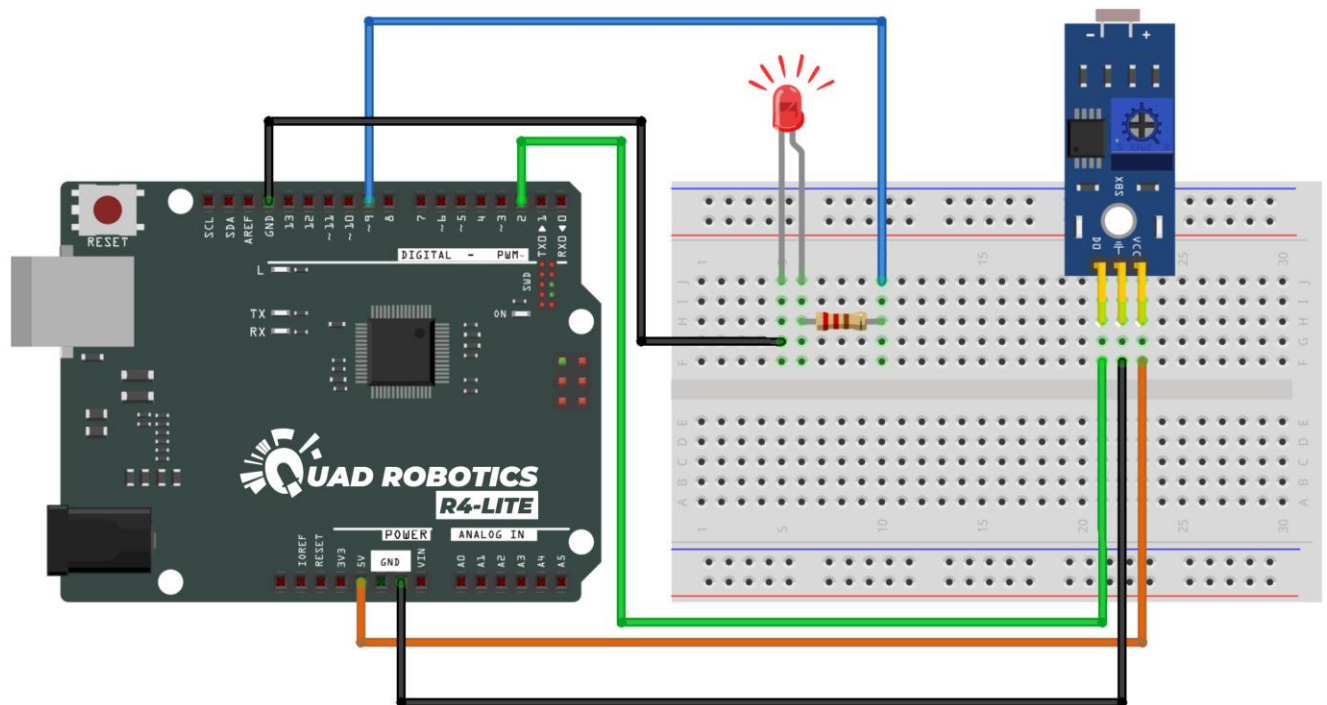
✦ Objective: Turn ON an LED in darkness using LDR sensor.

## 🧰 Components Required

- ✓ UNO R4 board
- ✓ Photoresistor / LDR
- ✓ LED
- ✓ Breadboard
- ✓ Jumper wires

## 💡 Circuit Diagram

Component	Connection Method	Uno R4
LED (Anode, Long Leg + )	Through <b>220Ω</b> resistor	Pin <b>9</b>
LED (Cathode, Short Leg -)	Direct Connection	GND
LDR (VCC)	Direct Connection	5v
LDR (GND)	Direct Connection	GND
LDR (SIGNAL /D0)	Direct Connection	Pin <b>2</b>



## Code

```
int ldr = 2;
int led = 9;

void setup() {
    pinMode(led, OUTPUT);
    pinMode(ldr, INPUT);
}

void loop() {
    int val = digitalRead(ldr);
    if(val == 500){
        digitalWrite(led, HIGH);
    } else {
        digitalWrite(led, LOW);
    }
}
```

## Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **8.LDRLamp.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

## Explanation & Output

The LDR detects light level.

LED turns ON when it's dark (low value) or close the LDR with your finger to turn ON the LED.

## DIY Extension

Adjust threshold value for sensitivity.

Use for automatic room lights.

# Project 9: Infrared Detector using TCRT5000

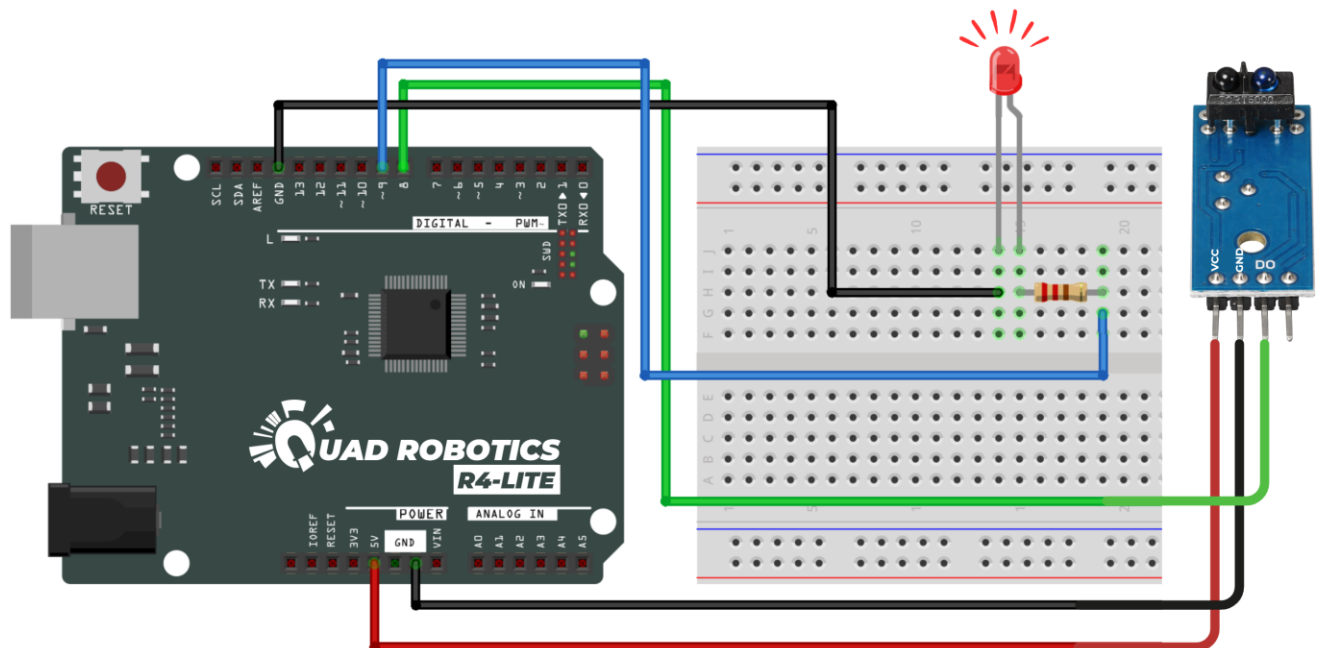
🌟 Objective: Detect objects using TCRT5000 IR sensor.

## 📦 Components Required

- ✓ UNO R4 board
- ✓ IR Sensor – TCRT5000 Module
- ✓ LED or Buzzer
- ✓ Breadboard
- ✓ Jumper wires

## 💡 Circuit Diagram

Component	Connection Method	Uno R4
IR Sensor - VCC	Direct Connection	<b>5V</b>
IR Sensor - GND	Direct Connection	<b>GND</b>
IR Sensor - DO /Signal	Direct Connection	<b>Pin 8</b>
LED (Anode, Long Leg + )	Through <b>220Ω</b> resistor	<b>Pin 9</b>
LED (Cathode, Short Leg -)	Direct Connection	<b>GND</b>



## Code

```
int ir = 8;
int led = 9;

void setup() {
  pinMode(ir, INPUT);
  pinMode(led, OUTPUT);
}

void loop() {
  if(digitalRead(ir) == LOW){
    digitalWrite(led, HIGH);
  } else {
    digitalWrite(led, LOW);
  }
}
```

## Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **copy and paste the above code** into the Arduino IDE OR open the file **9.IRDetector.ino** from the provided **CODE folder**.
6. Click on the **Upload button** (right arrow icon).
7. Wait for the code to **compile and upload**. The board will start running the program automatically.

## Explanation & Output

IR sensor detects reflected infrared light from objects.

When obstacle is detected, it turns on the LED.

## DIY Extension

Use in line-following robots.

# Project 10: Servo Motor Sweep

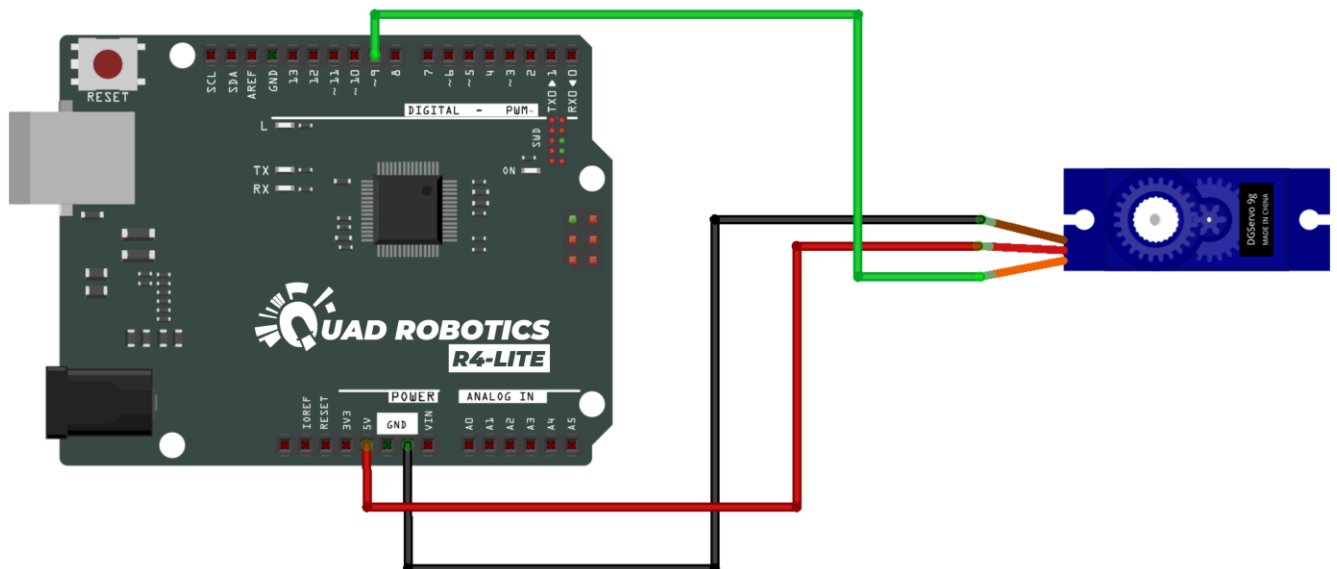
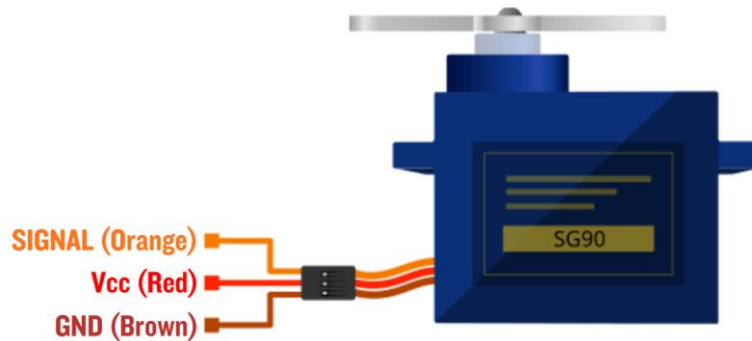
🌟 Objective: Move SG90 servo back and forth.

## 📦 Components Required

- ✓ UNO R4 board
- ✓ SG90 Servo
- ✓ Jumper wires. Use male to male jumper wires and insert into servo motor socket pins.

## 🔌 Circuit Diagram

Component	Connection Method	Uno R4
Servo Signal (Orange wire)	Direct Connection	Pin 9
Servo VCC (Red wire)	Direct Connection	5V
Servo GND (Brown wire)	Direct Connection	GND



## Code

```
#include <Servo.h>

Servo myservo;

void setup() {
  myservo.attach(9);
}

void loop() {
  for(int pos=0; pos<=180; pos++){
    myservo.write(pos);
    delay(15);
  }
  for(int pos=180; pos>=0; pos--){
    myservo.write(pos);
    delay(15);
  }
}
```

### Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **10.ServoSweep.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

## Explanation & Output

Servo is controlled by PWM signals.

We sweep from 0° to 180° and back.

## DIY Extension

Use for robot arms.

# Project 11: Ultrasonic Distance Finder

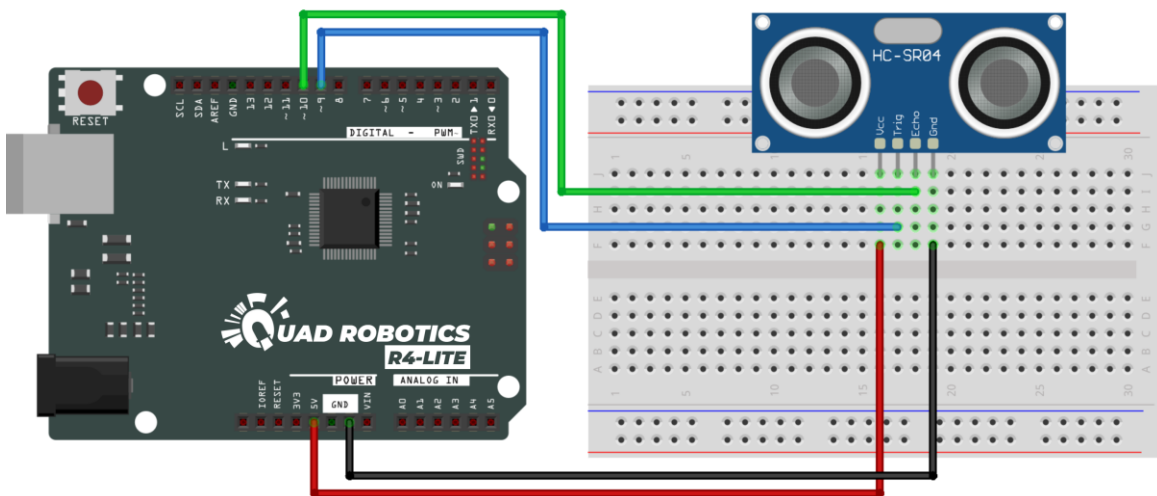
🌟 Objective: Measure distance using HC-SR04 ultrasonic sensor.

## 🧰 Components Required

- ✓ UNO R4 board
- ✓ Ultrasonic Sensor
- ✓ Breadboard
- ✓ Jumper wires

## 💡 Circuit Diagram

Component	Connection Method	Uno R4
Ultrasonic Sensor – Trig	Direct Connection	Pin 9
Ultrasonic Sensor – Echo	Direct Connection	Pin 10
Ultrasonic Sensor – VCC	Direct Connection	5V
Ultrasonic Sensor – GND	Direct Connection	GND



## 💻 Code

```
int trig = 9;
int echo = 10;
long duration;
int distance;

void setup() {
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  Serial.begin(9600);
}

void loop() {
  digitalWrite(trig, LOW);
  delayMicroseconds(2);
```



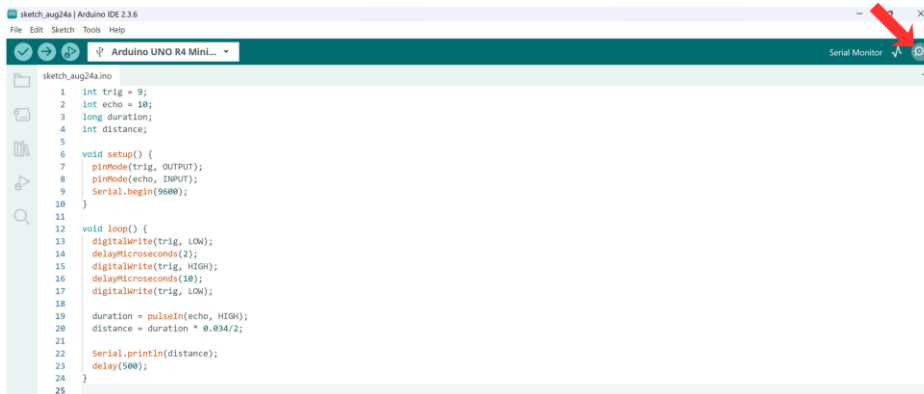
```
digitalWrite(trig, HIGH);
delayMicroseconds(10);
digitalWrite(trig, LOW);

duration = pulseIn(echo, HIGH);
distance = duration * 0.034/2;

Serial.println(distance);
delay(500);
}
```

## Steps to Upload Code

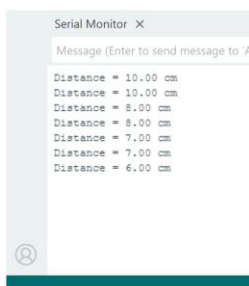
1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **11.Ultrasonics.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.



## Explanation & Output

Sensor sends ultrasonic pulse and measures echo time.

Distance is calculated using speed of sound and it is displayed in the serial monitor.



**✂️ DIY Extension:** Make obstacle avoidance robots.

# Project 12: DHT11 Weather Monitor

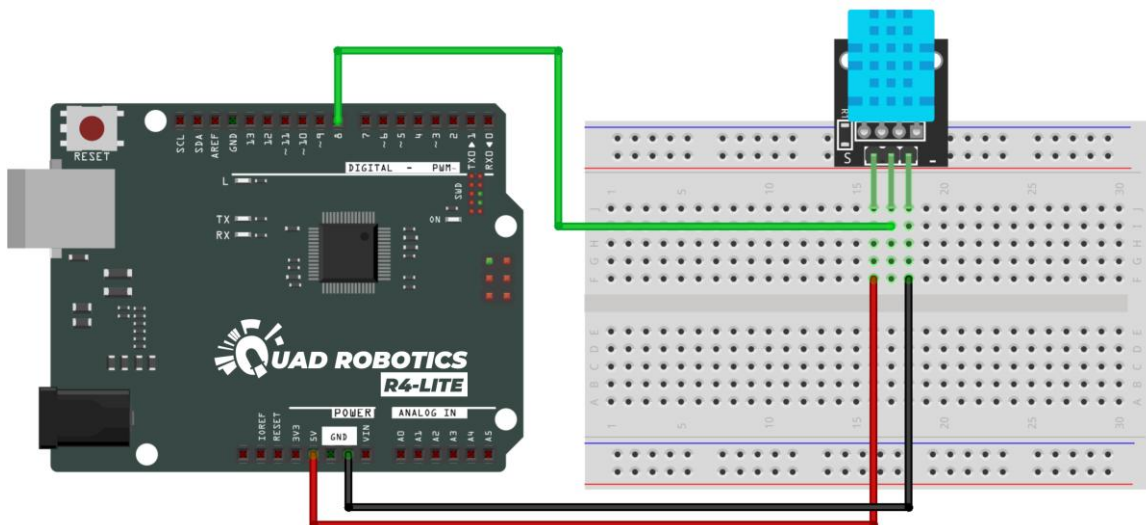
🌟 Objective: Read temperature and humidity using DHT11 sensor.

## 📦 Components Required

- ✓ UNO R4 board
- ✓ DHT11 Module
- ✓ Breadboard
- ✓ Jumper wires

## 🔌 Circuit Diagram

Component	Connection Method	Uno R4
DHT11 – VCC (+)	Direct Connection	5V
DHT11 – Data (S)	Direct Connection	Pin 8
DHT11 – GND (-)	Direct Connection	GND



## 💻 Code

```
#include <DHT.h>
#define DHTPIN 8
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  dht.begin();
}

void loop() {
  float h = dht.readHumidity();
  float t = dht.readTemperature();

  Serial.print("Humidity: ");
```

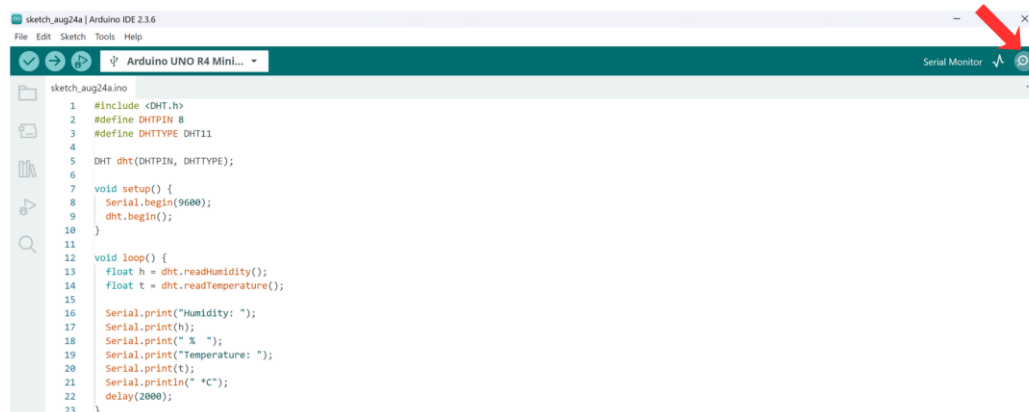
```

Serial.print(h);
Serial.print(" % ");
Serial.print("Temperature: ");
Serial.print(t);
Serial.println(" *C");
delay(2000);
}

```

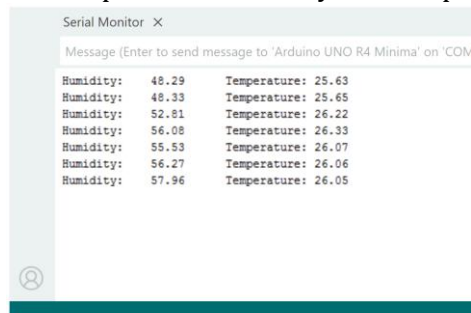
## Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **12.DHT11Monitor.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.



## Explanation & Output

DHT11 provides humidity and temperature readings. Values are printed on Serial Monitor.



**✂️ DIY Extension:** Make your own weather station.

# Project -13: Automated Gate/Smart Dustbin

✦ Objective: To make an automated gate for a miniature model using ultrasonic distance indicator so that when an object comes within a threshold the system. You can use the same connection setup for Smart Dustbin.

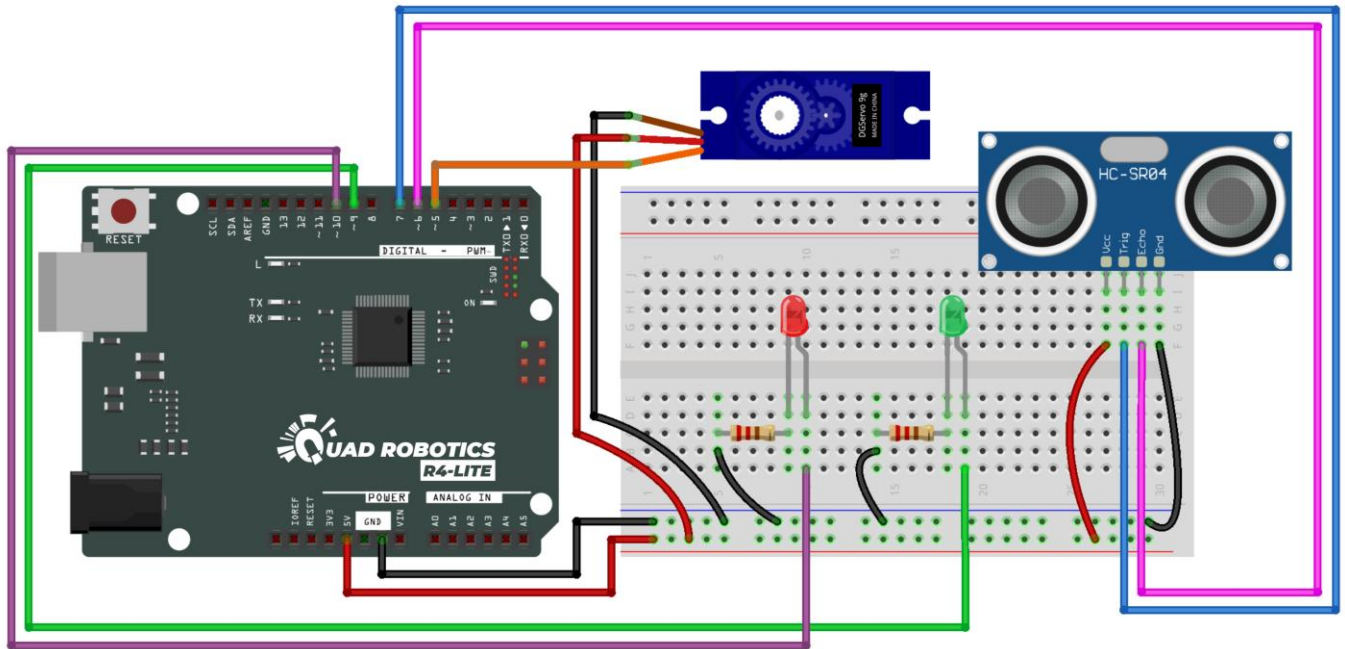
- turns Red LED ON,
- moves a servo to *open* the gate (e.g., rotate to opening angle), and when the object is away:
- turns Green LED ON,
- moves the servo to *close* the gate.

## Components Required

- ✓ UNO R4
- ✓ HC-SR04 ultrasonic sensor
- ✓ 1 × Red LED + 220Ω resistor
- ✓ 1 × Green LED + 220Ω resistor
- ✓ 1 × SG90 Servo Motor
- ✓ Jumper wires
- ✓ Breadboard
- ✓ USB cable for Arduino

## Circuit Diagram

Component - Ultrasonic Sensor	Connection Method	Uno R4
Ultrasonic Sensor – VCC	Direct Connection	5V
Ultrasonic Sensor – GND	Direct Connection	GND
Ultrasonic Sensor – TRIG	Direct Connection	Pin D7
Ultrasonic Sensor – ECHO	Direct Connection	Pin D6
Component - LED	Connection Method	Uno R4
Green LED (Anode, Long Leg +)	Through <b>220Ω resistor</b>	Pin D9
Green LED (Cathode, Short Leg –)	Direct Connection	GND
Red LED (Anode, Long Leg +)	Through <b>220Ω resistor</b>	Pin D10
Red LED (Cathode, Short Leg –)	Direct Connection	GND
Component - Servo Motor	Connection Method	Uno R4
Servo Signal (Orange wire)	Direct Connection	Pin D5
Servo VCC (Red wire)	External <b>5V supply</b> ( <i>recommended</i> )	5V
Servo GND (Brown wire)	Common Ground	GND



### Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **13.SmartGate.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

### Code

```
// UNO R4 - Ultrasonic + LEDs + Servo Gate
#include <Servo.h>

// Pins
const uint8_t TRIG_PIN = 7;
const uint8_t ECHO_PIN = 6;
const uint8_t GREEN_PIN = 9;
const uint8_t RED_PIN = 10;
const uint8_t SERVO_PIN = 5;

// Servo settings
Servo gateServo;
const int CLOSED_ANGLE = 0;    // servo angle when gate closed
```

```

const int OPEN_ANGLE    = 90;    // servo angle when gate open
const unsigned int SERVO_STEP_DELAY = 12; // ms between small steps for smooth
motion

// Ultrasonic settings
const unsigned long SENSOR_TIMEOUT = 30000UL; // microseconds
const float CM_PER_US = 0.0343 / 2.0; // speed of sound factor (cm/us /2)
const float THRESHOLD_CM = 15.0; // trigger distance in cm

// State tracking
bool gateOpen = false;
float lastDistance = -1;

void setup() {
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  pinMode(GREEN_PIN, OUTPUT);
  pinMode(RED_PIN, OUTPUT);

  digitalWrite(TRIG_PIN, LOW);
  digitalWrite(GREEN_PIN, LOW);
  digitalWrite(RED_PIN, LOW);

  Serial.begin(115200);
  delay(100);
  Serial.println("Ultrasonic + Servo Gate starting...");

  gateServo.attach(SERVO_PIN);
  // initialize gate closed
  setServoAngleSmooth(CLOSED_ANGLE);
  gateOpen = false;
  showGreen();
}

float readDistanceCM() {
  // Trigger 10us pulse
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  unsigned long duration = pulseIn(ECHO_PIN, HIGH, SENSOR_TIMEOUT);
  if (duration == 0) return -1.0; // timeout/no echo
  float dist = duration * CM_PER_US;
  return dist;
}

void showGreen() {
  digitalWrite(GREEN_PIN, HIGH);
  digitalWrite(RED_PIN, LOW);
}

void showRed() {
  digitalWrite(GREEN_PIN, LOW);
  digitalWrite(RED_PIN, HIGH);
}

void setServoAngleSmooth(int targetAngle) {
  int current = gateServo.read(); // get current angle

```

```

// If servo returns 255 (unknown), set it to target immediately
if (current == 255) {
    gateServo.write(targetAngle);
    delay(300);
    return;
}
if (current < targetAngle) {
    for (int a = current; a <= targetAngle; a++) {
        gateServo.write(a);
        delay(SERVO_STEP_DELAY);
    }
} else if (current > targetAngle) {
    for (int a = current; a >= targetAngle; a--) {
        gateServo.write(a);
        delay(SERVO_STEP_DELAY);
    }
}
// small settle delay
delay(80);
}

void openGate() {
    if (!gateOpen) {
        Serial.println("Opening gate...");
        setServoAngleSmooth(OPEN_ANGLE);
        gateOpen = true;
    }
}

void closeGate() {
    if (gateOpen) {
        Serial.println("Closing gate...");
        setServoAngleSmooth(CLOSED_ANGLE);
        gateOpen = false;
    }
}

void loop() {
    float dist = readDistanceCM();

    if (dist < 0) {
        Serial.println("No echo");
        // optional: indicate unknown by blinking both LEDs briefly
        digitalWrite(GREEN_PIN, LOW);
        digitalWrite(RED_PIN, LOW);
    } else {
        Serial.print("Distance: ");
        Serial.print(dist, 1);
        Serial.println(" cm");
        lastDistance = dist;

        if (dist <= THRESHOLD_CM) {
            // object near -> red + open gate
            showRed();
            openGate();
        } else {
            // clear -> green + close gate
            showGreen();
            closeGate();
        }
    }
}

```

```
}  
  
delay(120); // measurement pacing  
}
```

## Output

After uploading the code:

- On power-up: gate closed, Green LED ON.
- When an object moves within the threshold (e.g., car toy or hand): Red LED ON, servo rotates to open gate (smooth motion).
- When object moves away: Green LED ON, servo rotates back to closed position.
- Serial monitor prints distances and actions.

**NOTE:** The same wiring setup and code can be used for Smart Dustbin project as well. The concept is the same.

## DIY Extension

### ◆ Add more emotions

- **Manual override:** add a push button to manually open/close the gate.
- **Safety:** add a limit switch to detect fully open/closed positions.
- **Add buzzer:** beep when gate is opening or when object detected.



# Project 14: I2C LCD Display Basics

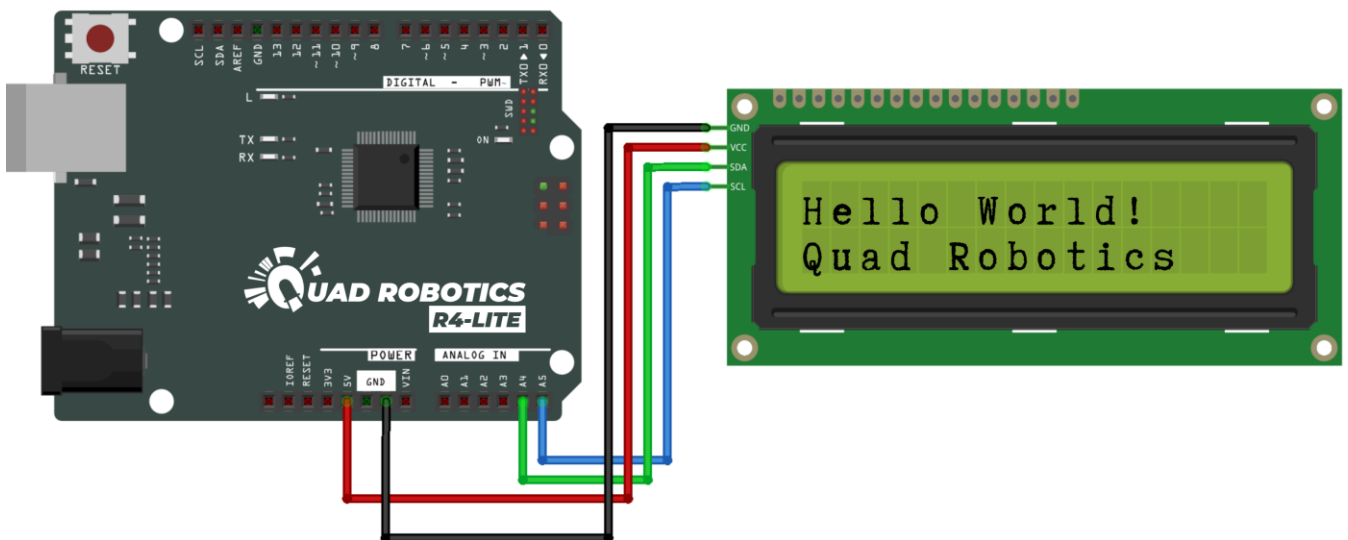
🌟 Objective: Display text on I2C LCD display.

## 📦 Components Required

- ✓ UNO R4
- ✓ 0.96 inch OLED Display (I2C)
- ✓ Male to Female Jumper wires

## 💡 Circuit Diagram

Component	Connection Method	Uno R4
I2C LCD – VCC	Direct Connection	5V
I2C LCD – GND	Direct Connection	GND
I2C LCD – SDA	Direct Connection	A4
I2C LCD – SCL	Direct Connection	A5



## ✅ Required Library

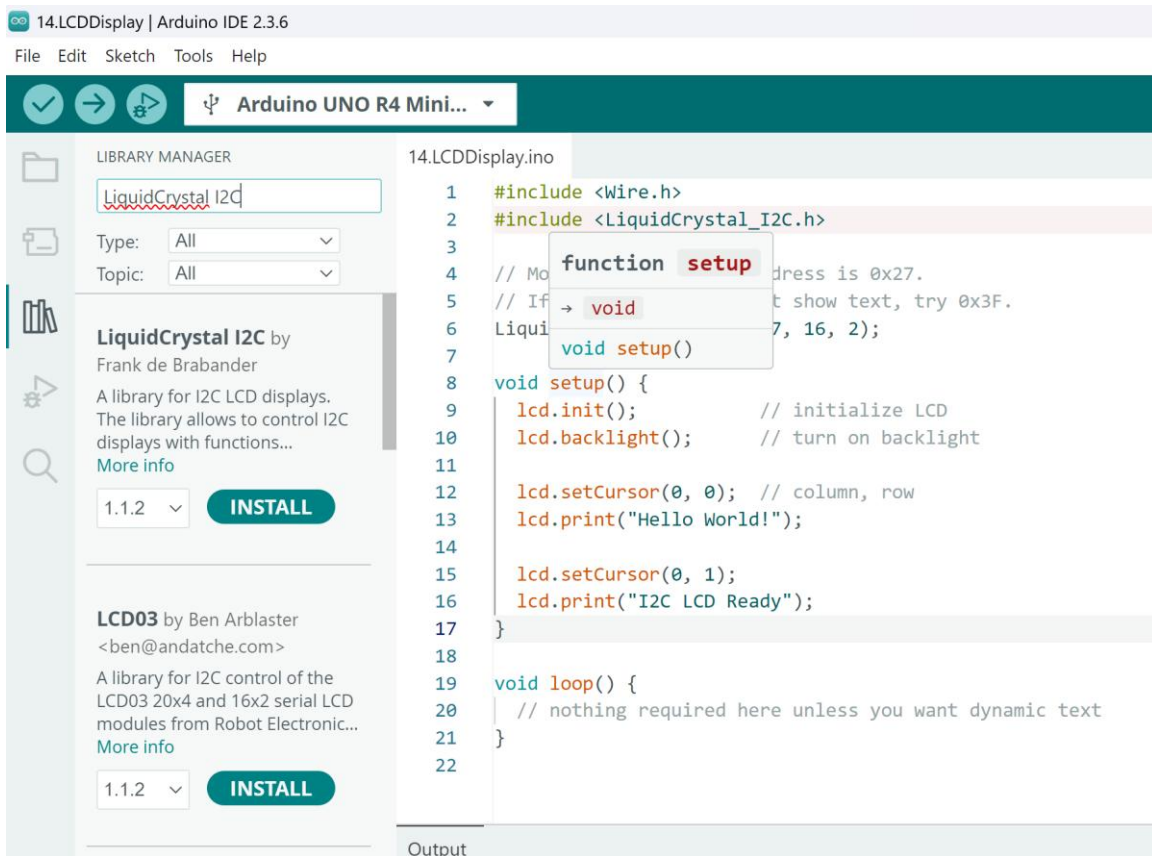
Install this library in Arduino IDE:

**LiquidCrystal\_I2C**

(Author: Frank de Brabander)

### Steps:

Sketch → Include Library → Manage Libraries → search "**LiquidCrystal I2C**" → Install



### Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **14.LEDDispaly.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

## Code

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

// Most common I2C LCD address is 0x27.
// If your display doesn't show text, try 0x3F.
LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup() {
  lcd.init();           // initialize LCD
  lcd.backlight();      // turn on backlight

  lcd.setCursor(0, 0);  // column, row
  lcd.print("Hello World!");

  lcd.setCursor(0, 1);
  lcd.print("Quad Robotics");
}

void loop() {
  // nothing required here unless you want dynamic text
}
```

## Explanation & Output

The LED uses I2C communication (SDA, SCL).

Output: We display simple text “Hello World” on the first line of screen and “Quad Robotics” text on the second line.

## DIY Extension

Show sensor values on LED.

Make a mini weather station display.

# Project -15: Digital Locker

🌟 Objective: Build a digital password-protected smart safe where:

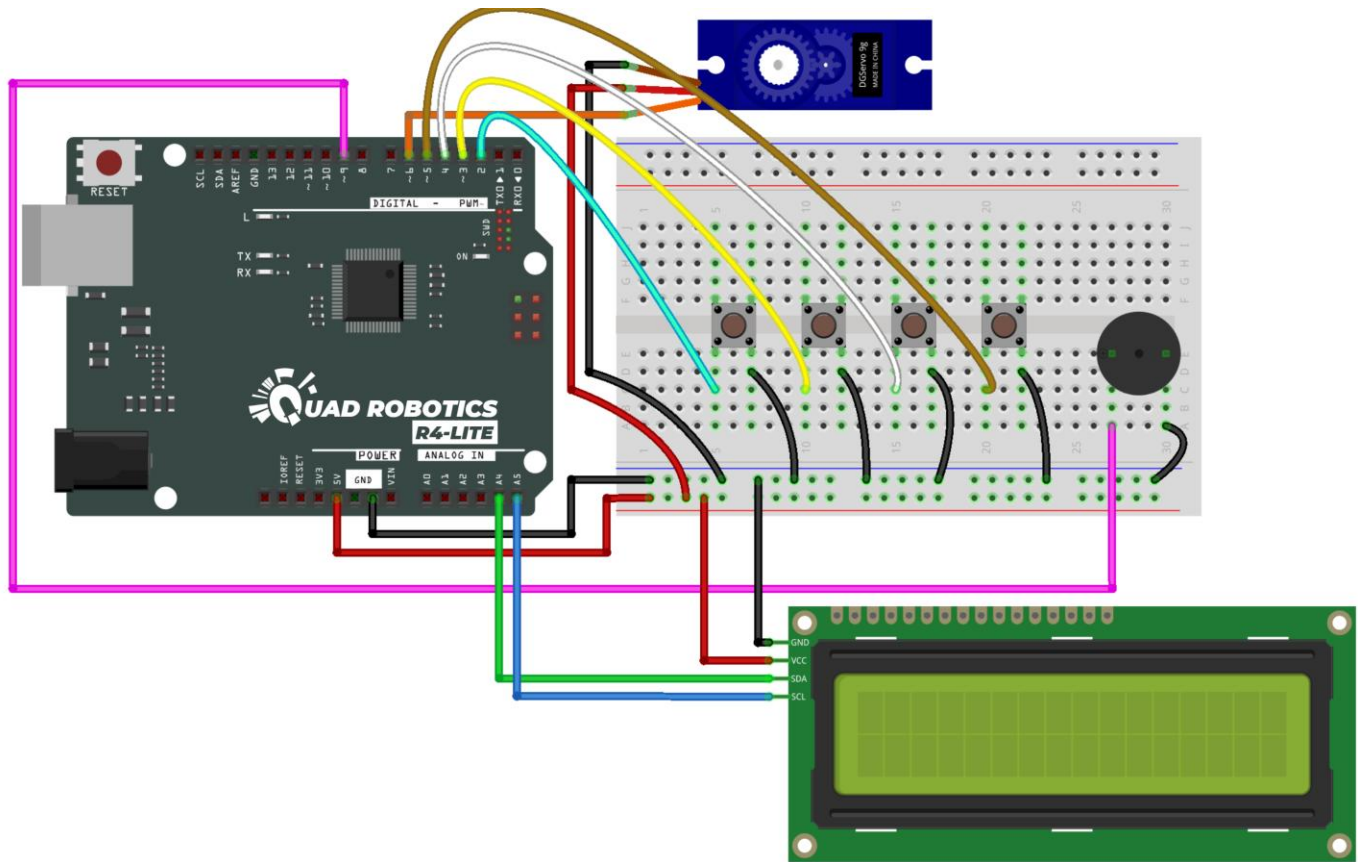
- User enters a 4-digit password using push buttons
- LCD guides the user
- Correct password → Servo unlocks the box
- Wrong password → buzzer alert + message on LCD

## Components Required

- ✓ UNO R4
- ✓ 4 × Push buttons
- ✓ I2C 1602 LCD
- ✓ Servo motor (SG90)
- ✓ Active buzzer
- ✓ Jumper wires
- ✓ Breadboard

## Circuit Diagram

Component - Push Buttons	Connection Method	Uno R4
Button <b>B1</b>	Direct Connection	Pin <b>D2</b>
Button <b>B1</b> (Other Leg)	Direct Connection	<b>GND</b>
Button <b>B2</b>	Direct Connection	Pin <b>D3</b>
Button <b>B2</b> (Other Leg)	Direct Connection	<b>GND</b>
Button <b>B3</b>	Direct Connection	Pin <b>D4</b>
Button <b>B3</b> (Other Leg)	Direct Connection	<b>GND</b>
Button <b>B4</b>	Direct Connection	Pin <b>D5</b>
Button <b>B4</b> (Other Leg)	Direct Connection	<b>GND</b>
Component - LCD	Connection Method	Uno R4
I2C LCD – VCC	Direct Connection	<b>5V</b>
I2C LCD – GND	Direct Connection	<b>GND</b>
I2C LCD – SDA	Direct Connection	<b>A4</b>
I2C LCD – SCL	Direct Connection	<b>A5</b>
Component - Servo Motor	Connection Method	Uno R4
Servo Signal (Orange wire)	Direct Connection	Pin <b>D6</b>
Servo VCC (Red wire)	Direct Connection	<b>5V</b>
Servo GND (Brown wire)	Common Ground	<b>GND</b>
Component - Buzzer	Connection Method	Uno R4
Buzzer Signal (+)	Direct Connection	Pin <b>D9</b>
Buzzer GND (–)	Direct Connection	<b>GND</b>



### 📖 Steps to Upload Code

1. Connect your **UNO R4** board to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **15.DigitalLocker.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

## Code

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <Servo.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);
Servo lockServo;

int buttons[4] = {2, 3, 4, 5};
int password[4] = {1, 2, 3, 4};
int entered[4];
int indexPos = 0;

const int buzzer = 9;
const int servoPin = 6;

void setup() {
  // LCD
  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("SMART SAFE BOX");
  lcd.setCursor(0, 1);
  lcd.print("ENTER CODE...");
  delay(1500);
  lcd.clear();

  // Buttons
  for (int i = 0; i < 4; i++)
    pinMode(buttons[i], INPUT_PULLUP);

  // Servo
  lockServo.attach(servoPin);
  lockServo.write(0); // locked

  // Buzzer
  pinMode(buzzer, OUTPUT);
}

bool checkPassword() {
  for (int i = 0; i < 4; i++) {
    if (entered[i] != password[i]) return false;
  }
  return true;
}

void loop() {
  lcd.setCursor(0, 0);
  lcd.print("ENTER CODE:  ");
  lcd.setCursor(0, 1);
  lcd.print("Keys: ");
  lcd.print(indexPos);
  lcd.print("    ");

  for (int i = 0; i < 4; i++) {
    if (digitalRead(buttons[i]) == LOW) {
      delay(200);
      entered[indexPos] = i + 1;
    }
  }
}
```

```

    indexPos++;

    if (indexPos == 4) {
        lcd.clear();
        if (checkPassword()) {
            lcd.setCursor(0, 0);
            lcd.print("CORRECT CODE!");
            lcd.setCursor(0, 1);
            lcd.print("SAFE UNLOCKED");
            lockServo.write(90); // unlock
        } else {
            lcd.setCursor(0, 0);
            lcd.print("WRONG CODE");
            lcd.setCursor(0, 1);
            lcd.print("ACCESS DENIED");
            tone(buzzer, 600, 400);
            lockServo.write(0); // ensure locked
        }

        delay(2000);
        lcd.clear();
        indexPos = 0;
    }
}
}
}
}

```

## Output

After uploading the code:

- 4 push buttons simulate numeric keypad digits as 1, 2, 3, 4.
- User enters 4 digits → stored in entered[].
- Program compares entered digits with password {1,2,3,4}.
- LCD displays real-time prompts and status.
- Correct password → servo rotates to open the safe (90° ).
- Wrong password → error message + buzzer alert.

## DIY Extension

### ◆ Add more emotions

- Add "change password" option
- Add IR remote password input
- Add attempt counter → lockout after 3 wrong tries
- Add a vibration sensor → tamper detection
- Add MAX7219 matrix for fancy animations
- Add buzzer beep for each button press.

# Project 16: 1-Digit 7 Segment Display

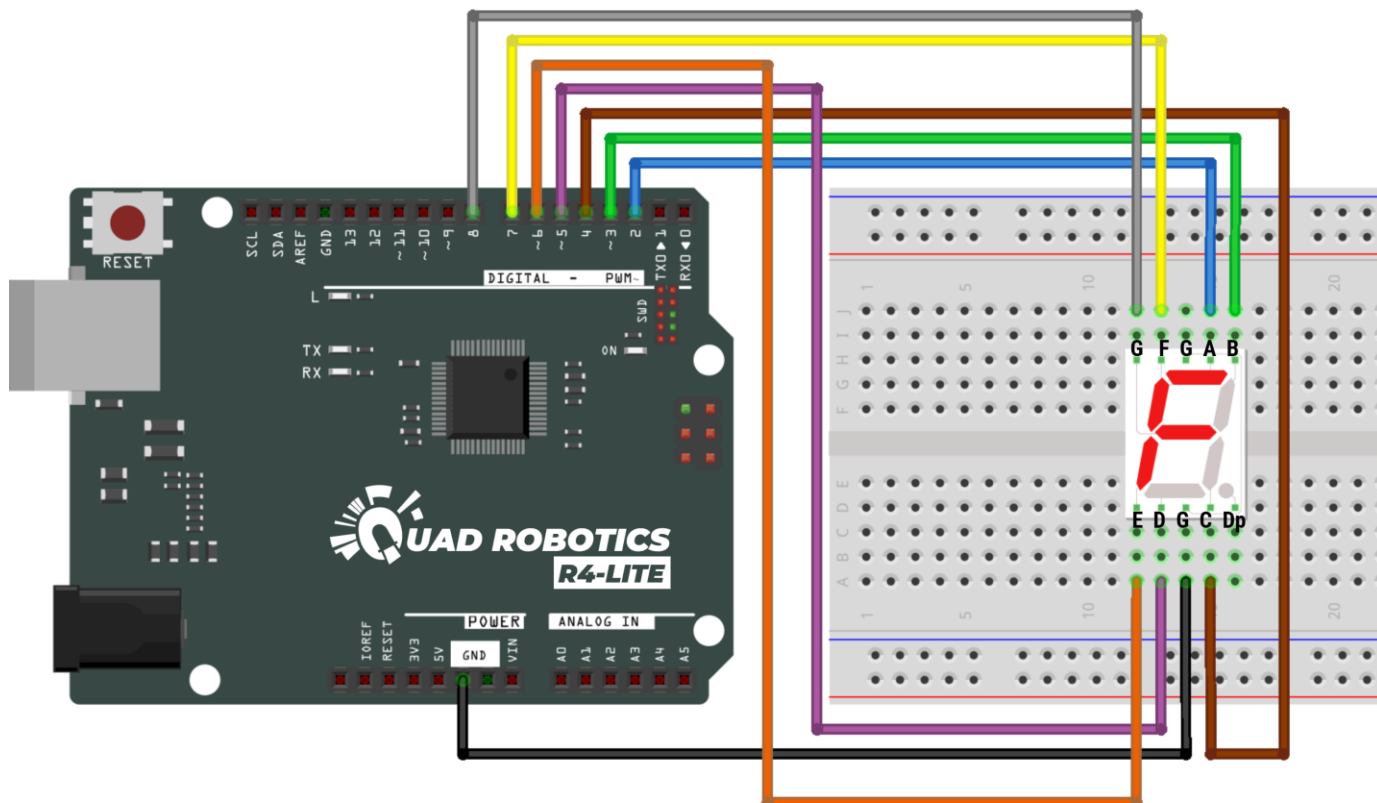
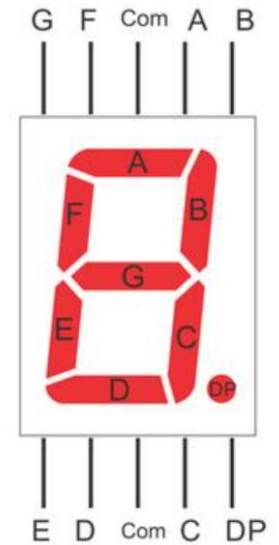
🌟 Objective: To connect a **1-digit 7-segment display** to an **Arduino UNO R4** and display numbers from 0 to 9 by controlling each segment using digital output pins.

## 🧰 Components Required

- ✓ UNO R4 board
- ✓ 1-digit 7-segment display (Common Cathode)
- ✓ 8 × 220Ω resistors (optional for safety)
- ✓ Jumper wires
- ✓ Breadboard

## 💡 Circuit Diagram

Component	Connection Method	Uno R4
Segment A	Direct Connection /Optional <b>220Ω resistor</b>	Pin <b>D2</b>
Segment B	Direct Connection /Optional <b>220Ω resistor</b>	Pin <b>D3</b>
Segment C	Direct Connection /Optional <b>220Ω resistor</b>	Pin <b>D4</b>
Segment D	Direct Connection /Optional <b>220Ω resistor</b>	Pin <b>D5</b>
Segment E	Direct Connection /Optional <b>220Ω resistor</b>	Pin <b>D6</b>
Segment F	Direct Connection /Optional <b>220Ω resistor</b>	Pin <b>D7</b>
Segment G	Direct Connection /Optional <b>220Ω resistor</b>	Pin <b>D8</b>
Common Pin (COM)	Direct Connection	<b>GND</b>
Decimal Point (DP)	<i>Not connected</i>	—





## Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **16.7Segment1Digit.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

## Code

```
// Pin connections for each segment
int a = 2;
int b = 3;
int c = 4;
int d = 5;
int e = 6;
int f = 7;
int g = 8;
int dp = 9;

// Array of segments for easy looping
int segments[] = {a, b, c, d, e, f, g};

void setup() {
  // Set all segment pins as outputs
  for (int i = 0; i < 7; i++) {
    pinMode(segments[i], OUTPUT);
  }
  pinMode(dp, OUTPUT);
}

void displayDigit(int digit) {
  // Segment patterns for digits 0-9 (common cathode)
  const byte numbers[10][7] = {
    // a b c d e f g
    {1,1,1,1,1,1,0}, // 0
    {0,1,1,0,0,0,0}, // 1
    {1,1,0,1,1,0,1}, // 2
    {1,1,1,1,0,0,1}, // 3
    {0,1,1,0,0,1,1}, // 4
    {1,0,1,1,0,1,1}, // 5
    {1,0,1,1,1,1,1}, // 6
    {1,1,1,0,0,0,0}, // 7
    {1,1,1,1,1,1,1}, // 8
  }
```

```

    {1,1,1,1,0,1,1}    // 9
};

// Write segment values
for (int i = 0; i < 7; i++) {
    digitalWrite(segments[i], numbers[digit][i]);
}
}

void loop() {
    for (int i = 0; i < 10; i++) {
        displayDigit(i);
        delay(1000);    // show each number for 1 second
    }
}

```

## 🔍 Explanation & Output

After uploading the code:

- The 7-segment display will show:  
0 → 1 → 2 → 3 → 4 → 5 → 6 → 7 → 8 → 9
- Each number appears for 1 second before switching.
- All segments glow clearly using individual resistors.

## ✂️ DIY Extension

**Display a custom pattern:** Show letters like A, b, C, d using custom segment mappings.

# Project -17: Smart Weather Station

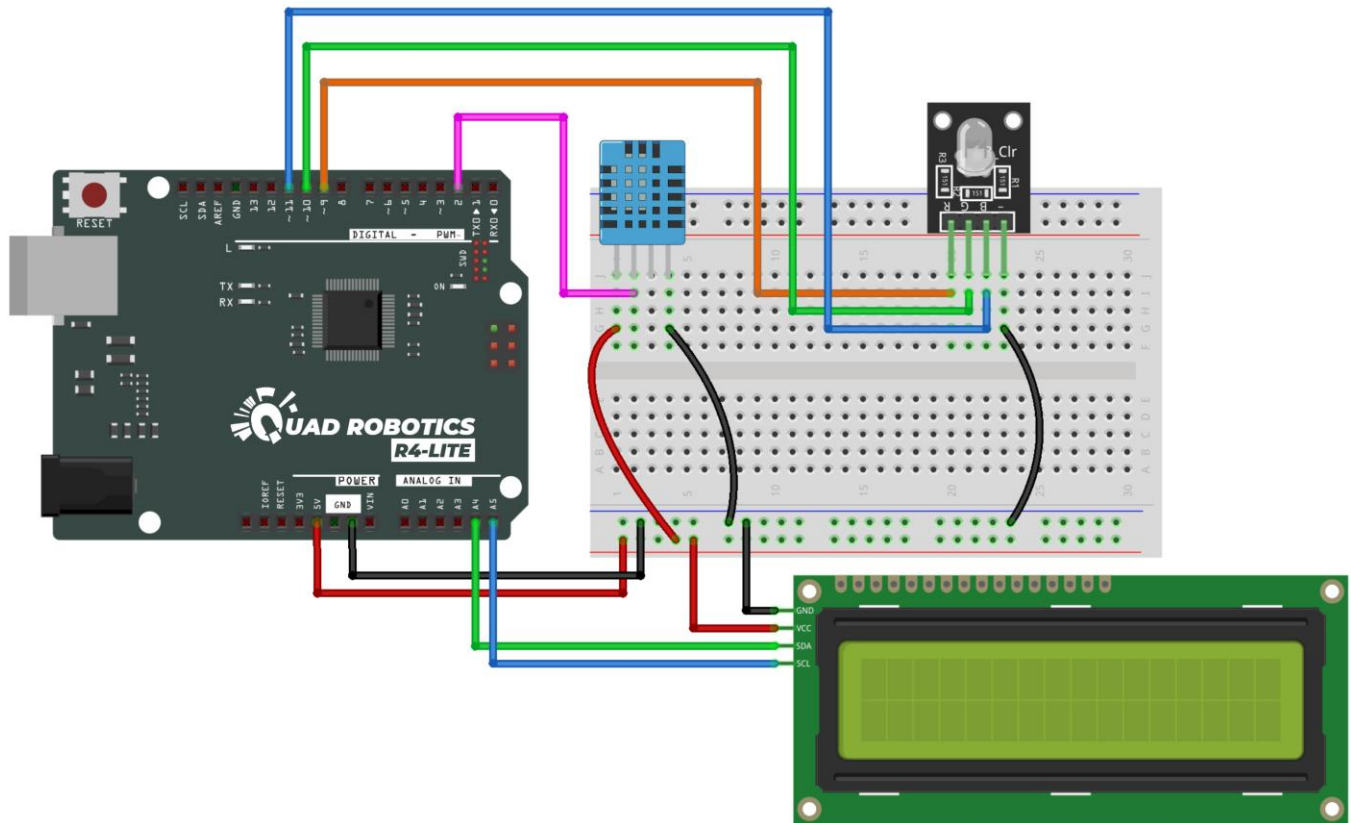
✦ Objective: Build a mini weather station that shows temperature & humidity on the LCD and uses RGB LED to show climate condition.

## Components Required

- ✓ UNO R4
- ✓ DHT11 sensor
- ✓ I2C 1602 LCD
- ✓ RGB LED
- ✓ Jumper wires

## Circuit Diagram

Component - DHT11 Temperature Sensor	Connection Method	Uno R4
DHT11 – Signal	Direct Connection	Pin <b>D2</b>
DHT11 – VCC	Direct Connection	<b>5V</b>
DHT11 – GND	Direct Connection	<b>GND</b>
Component - LCD Display	Connection Method	Uno R4
I2C LCD – VCC	Direct Connection	<b>5V</b>
I2C LCD – GND	Direct Connection	<b>GND</b>
I2C LCD – SDA	Direct Connection	<b>A4</b>
I2C LCD – SCL	Direct Connection	<b>A5</b>
Component - RGB Led	Connection Method	Uno R4
RGB LED – Red (R)	Direct Connection	Pin <b>D9</b>
RGB LED – Green (G)	Direct Connection	Pin <b>D10</b>
RGB LED – Blue (B)	Direct Connection	Pin <b>D11</b>
RGB LED – Common Cathode (-)	Direct Connection	<b>GND</b>



### Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **17.WeatherStation.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

### Code

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <DHT.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);

#define DHTPIN 2
#define DHTTYPE DHT11
```

```

DHT dht(DHTPIN, DHTTYPE);

int R = 9, G = 10, B = 11;

void setColor(int r, int g, int b) {
  analogWrite(R, r);
  analogWrite(G, g);
  analogWrite(B, b);
}

void setup() {
  lcd.init();
  lcd.backlight();
  dht.begin();
  pinMode(R, OUTPUT);
  pinMode(G, OUTPUT);
  pinMode(B, OUTPUT);
}

void loop() {
  float t = dht.readTemperature();
  float h = dht.readHumidity();

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Temp: "); lcd.print(t); lcd.print("C");
  lcd.setCursor(0, 1);
  lcd.print("Hum : "); lcd.print(h); lcd.print("%");

  if (t < 20) setColor(0, 0, 255);      // cold - blue
  else if (t > 30) setColor(255, 0, 0); // hot - red
  else setColor(0, 255, 0);            // ideal - green

  delay(1500);
}

```

## Output

After uploading the code:

- LCD updates every 1.5 seconds.
- RGB LED shows climate condition.

## DIY Extension

### ◆ Add more emotions

- Add buzzer alarm for extreme heat
- Add MAX7219 scrolling temperature
- Add humidity-controlled fan

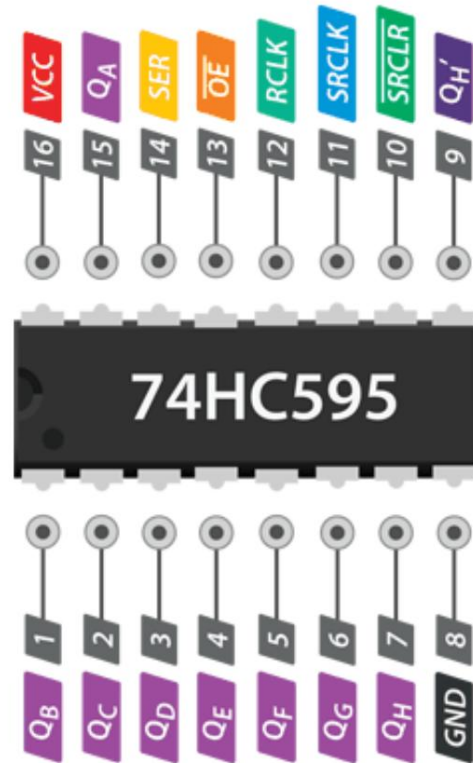
# Project -18: LED Direction Bar

🌟 Objective: Display a smooth bar-graph animation (LEDs light progressively from 1→7 then back 7→1) using a single 74HC595 shift register driven by an UNO R4. Good demo for shift-register concepts and efficient use of IO pins.

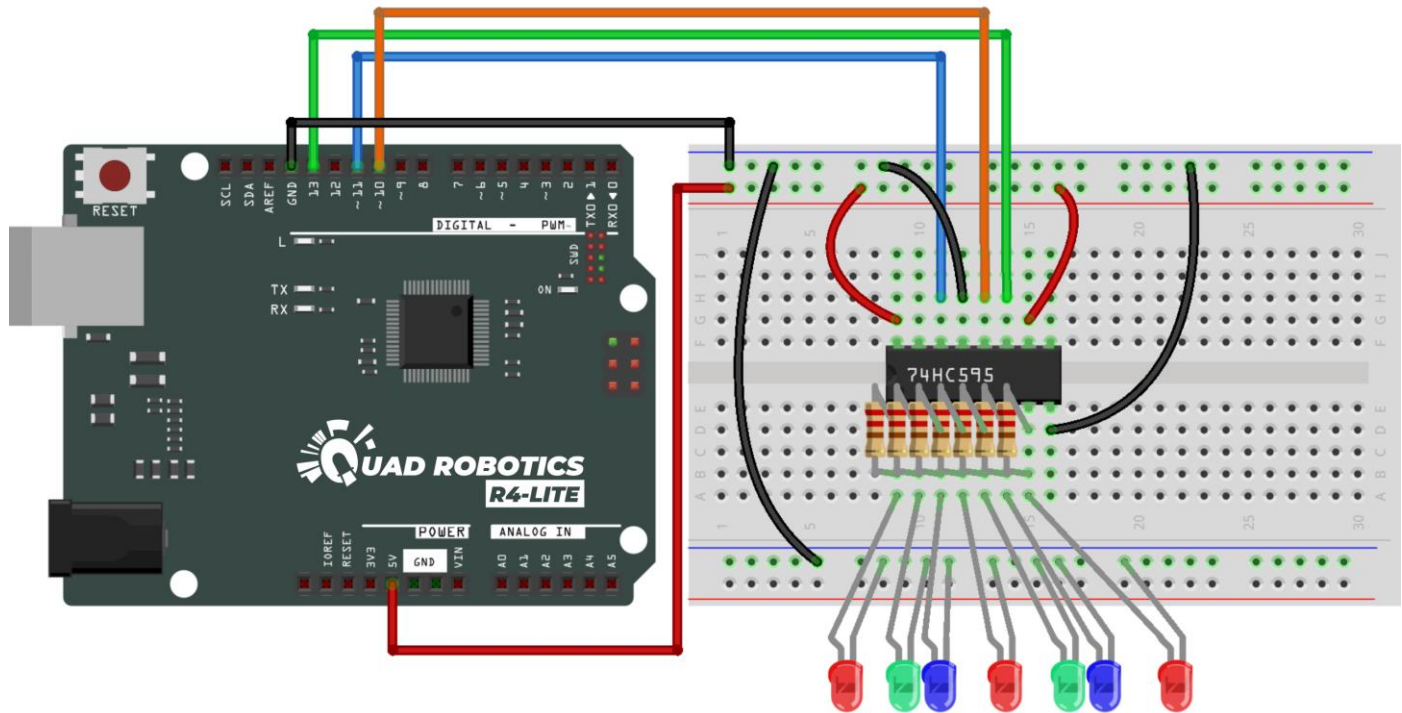
## 📦 Components Required

- ✓ UNO R4 board
- ✓ 1 × 74HC595 (8-bit shift register)
- ✓ 7 × LEDs (any color)
- ✓ 7 × 220 Ω resistors (one per LED)
- ✓ Breadboard & jumper wires

## 💡 Circuit Diagram



Component - 74HC595 Pin	Connection Method	Uno R4
74HC595 VCC (Pin 16)	Direct connection (+5V)	5V
74HC595 GND (Pin 8)	Direct connection (Ground)	GND
SER / DS (Pin 14)	Serial data input	D11
SRCLK / SH_CP (Pin 11)	Shift clock	D13
RCLK / ST_CP (Pin 12)	Latch clock	D10
SRCLR / MR (Pin 10)	Tie HIGH to disable reset	5V
OE (Pin 13)	Tie LOW to enable outputs	GND
QH' (Pin 9)	Cascade output (optional)	Not connected
Component - Led's	Connection Method	74HC595 Pin
LED 1 (Anode +)	Through 220Ω resistor	QB (Pin 1)
LED 2 (Anode +)	Through 220Ω resistor	QC (Pin 2)
LED 3 (Anode +)	Through 220Ω resistor	QD (Pin 3)
LED 4 (Anode +)	Through 220Ω resistor	QE (Pin 4)
LED 5 (Anode +)	Through 220Ω resistor	QF (Pin 5)
LED 6 (Anode +)	Through 220Ω resistor	QG (Pin 6)
LED 7 (Anode +)	Through 220Ω resistor	QH (Pin 7)
All LEDs (Cathode -)	Direct Connection	GND (Pin 8)



### Steps to Upload Code

1. Connect your **UNO R4** board to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **18.LedDirection.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

## Code

```
// Bar-graph mode using 74HC595
// Arduino pins
const int dataPin = 11; // DS (74HC595 pin 14)
const int clockPin = 13; // SH_CP (74HC595 pin 11)
const int latchPin = 10; // ST_CP (74HC595 pin 12)

// Optional: push button to toggle behavior (not required)
const int buttonPin = 2; // use INPUT_PULLUP if you wire a button

const int STEP_DELAY = 180; // ms between steps

// Write one byte to the 74HC595 (LSB -> Q0)
void shiftWrite(byte value) {
    digitalWrite(latchPin, LOW);
    // Use LSBFIRST so bit0 -> Q0 (LED1), bit7 -> Q7 (LED8)
    shiftOut(dataPin, clockPin, LSBFIRST, value);
    digitalWrite(latchPin, HIGH);
}

void setup() {
    pinMode(dataPin, OUTPUT);
    pinMode(clockPin, OUTPUT);
    pinMode(latchPin, OUTPUT);

    // Optional button
    pinMode(buttonPin, INPUT_PULLUP);

    // Clear outputs
    shiftWrite(0x00);
}

void loop() {
    // Grow from 0 to 8 LEDs
    for (int i = 1; i <= 8; i++) {
        byte pattern = 0;
        for (int b = 0; b < i; b++) {
            pattern |= (1 << b); // set bit b (LSB = LED1)
        }
        shiftWrite(pattern);
        delay(STEP_DELAY);
        // If button pressed, you could break/modify behavior here (optional)
    }

    // Pause at full
    delay(300);

    // Shrink from 8 down to 0
    for (int i = 8; i >= 0; i--) {
        byte pattern = 0;
        for (int b = 0; b < i; b++) {
            pattern |= (1 << b);
        }
        shiftWrite(pattern);
        delay(STEP_DELAY);
    }

    // Optional pause
```



```
delay(300);  
}
```

## Output

After uploading the code:

- LEDs light progressively from LED1 → LED7 one by one until all are ON.
- After a brief pause the LEDs turn off one by one (shrinking).
- The animation repeats continuously.
- Step speed is controlled by STEP\_DELAY (increase to slow down, decrease to speed up).

## DIY Extension

### Add more emotions

- **Mode button:** Press to switch between Bar Graph, Running Dot, and Knight Rider styles.
- **Ultrasonic input:** Use HC-SR04 distance reading to display the proximity as bar height (mapping distance to 0..8).

# Project 19: Relay Module

✦ Control a relay module using an Arduino so that the relay turns **ON for 2 seconds and OFF for 2 seconds repeatedly**.

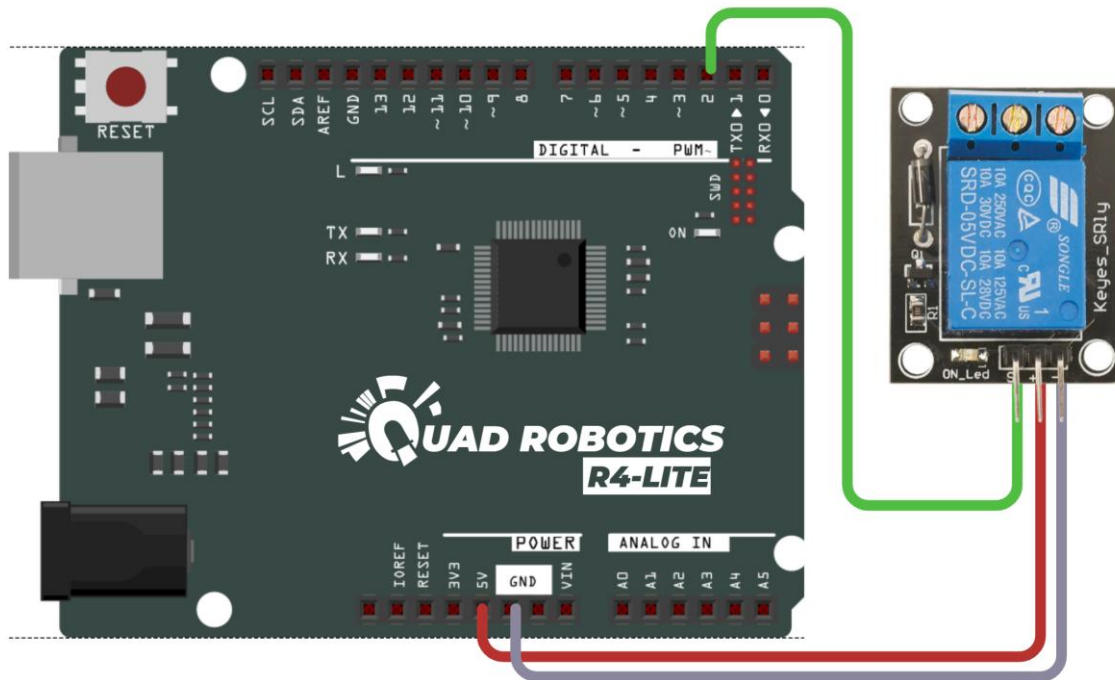
This can be used to switch AC/DC loads like bulbs, motors, or pumps at timed intervals.

## 📦 Components Required

- ✓ UNO R4 board
- ✓ Relay Module
- ✓ Jumper wires

## 💡 Circuit Diagram

Component	Connection Method	Uno R4
Relay - VCC	Direct Connection	<b>5V</b>
Relay - GND	Direct Connection	<b>GND</b>
Relay - S/Signal	Direct Connection	<b>Pin 2</b>



## Code

```
int relayPin = 2;    // Relay connected to digital pin 2

void setup() {
  pinMode(relayPin, OUTPUT);
}

void loop() {
  digitalWrite(relayPin, HIGH); // Relay ON
  delay(2000);                  // Wait 2 seconds

  digitalWrite(relayPin, LOW);  // Relay OFF
  delay(2000);                  // Wait 2 seconds
}
```

## Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **19.Relay.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

## Explanation & Output

- Relay clicks every 2 seconds
- ON LED on relay glows → OFF → ON repeatedly
- Connected load switches ON/OFF in the same interval

# Project 20: Bluetooth Control - Android App

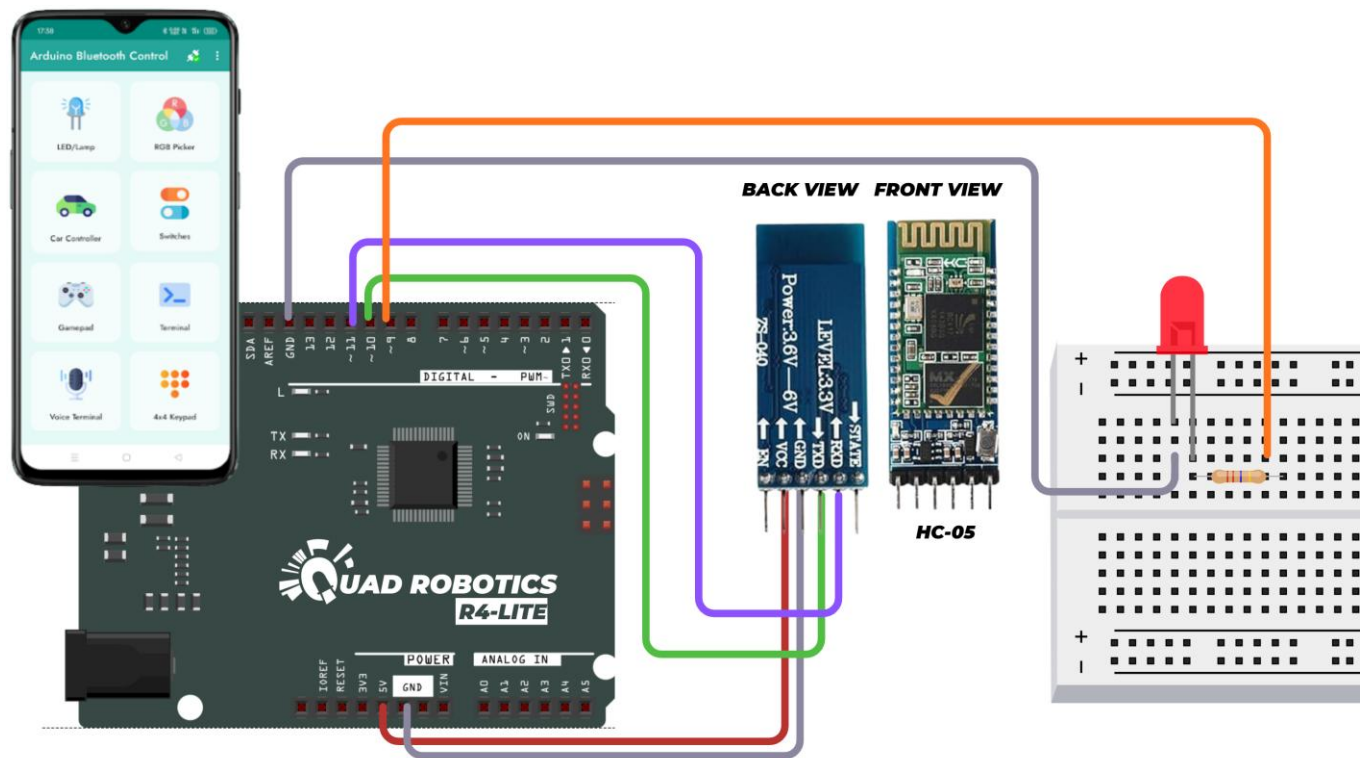
✦ Control an LED connected to **pin 9** using the Android app you shared (Bluetooth HC-05/06 Arduino Control). This **works only** in **Android Phones**. **iOS is not supported**.

## 🧰 Components Required

- ✓ UNO R4 board
- ✓ Bluetooth Module
- ✓ Led
- ✓ 220-ohm resistor
- ✓ Jumper wires
- ✓ Android phone to install app

## 💡 Circuit Diagram

Component	Connection Method	Uno R4
Reley - VCC	Direct Connection	<b>5V</b>
Reley - GND	Direct Connection	<b>GND</b>
Reley - S /Signal	Direct Connection	<b>Pin 2</b>



## Code

```
#include <SoftwareSerial.h>

SoftwareSerial BT(10, 11); // RX, TX

int ledPin = 9;
char data;

void setup() {
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, LOW);

  Serial.begin(9600);
  BT.begin(9600);    // HC-05 default baud rate
}

void loop() {
  if (BT.available()) {
    data = BT.read();

    if (data == '1') {
      digitalWrite(ledPin, HIGH); // LED ON
    }

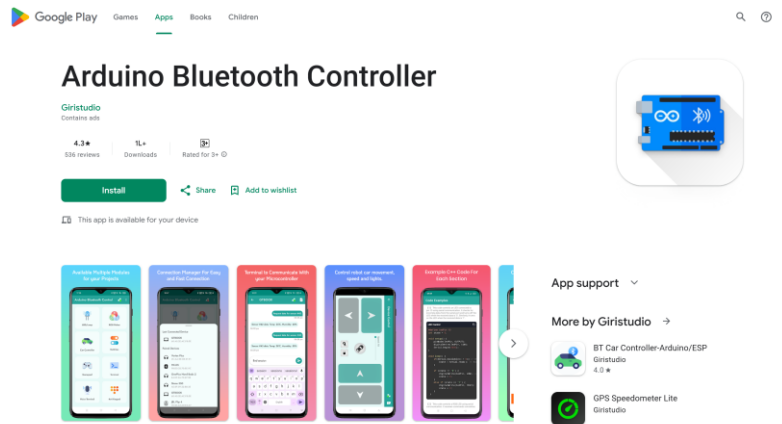
    if (data == '0') {
      digitalWrite(ledPin, LOW);   // LED OFF
    }
  }
}
```

## Steps to Upload Code

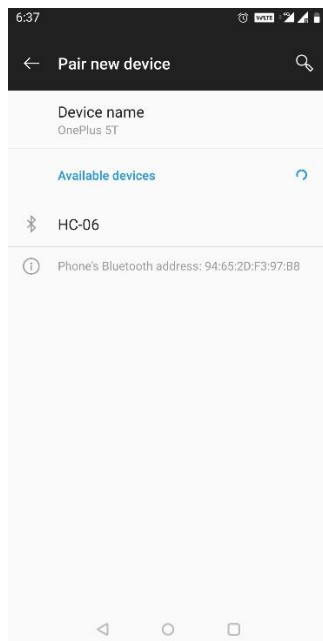
1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **20.Bluetooth.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

## Install the Android App

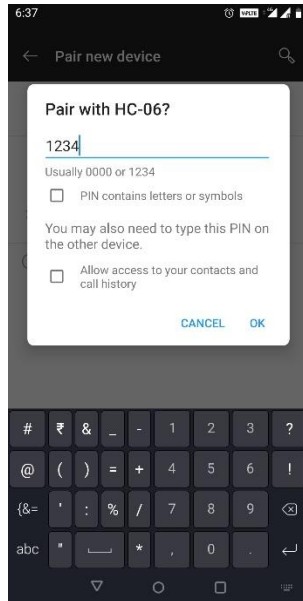
- Go to “Play Store” in you Android Phone
- Search for “Arduino Bluetooth Controller”
- Install the App “Arduino Bluetooth Controller” by GiriStudio



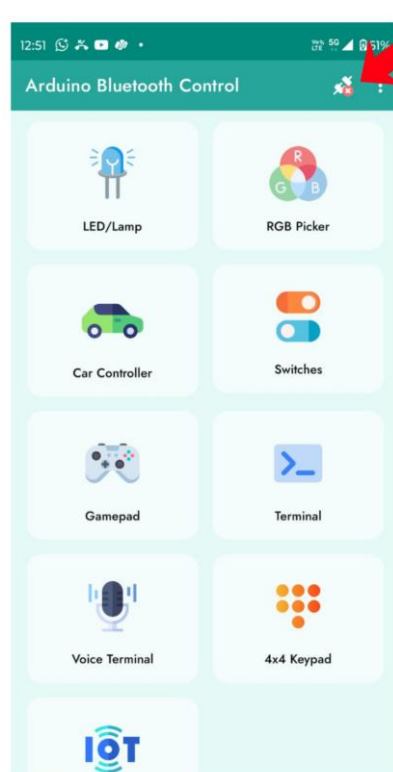
- Open Setting in your Android Phone. Go to Bluetooth and Click Pair Option to pair with HC-05/06 module
  - Bluetooth name: HC-05



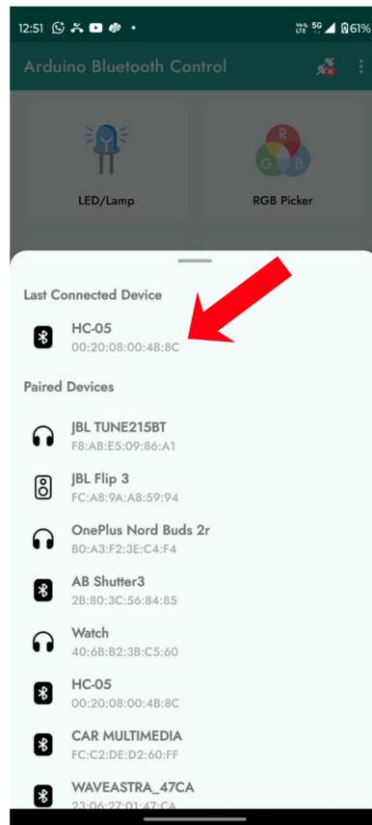
- Enter Password: 1234 or 0000



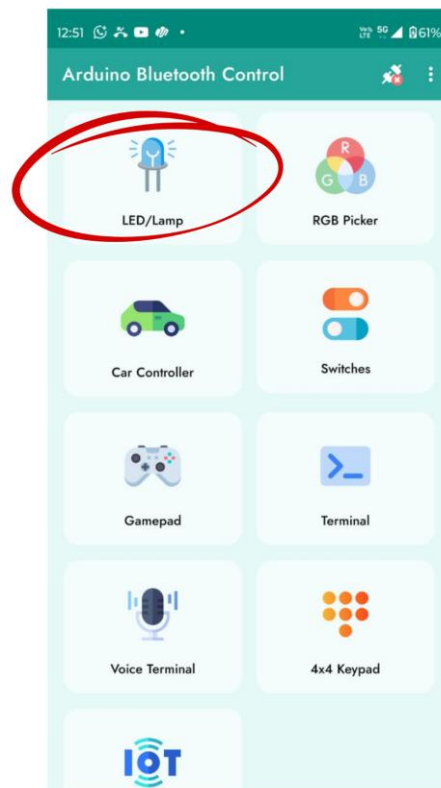
- Now open the “Arduino Bluetooth App. Click on the “Plug” icon on the top right.



- Click on HC-05 to get connected

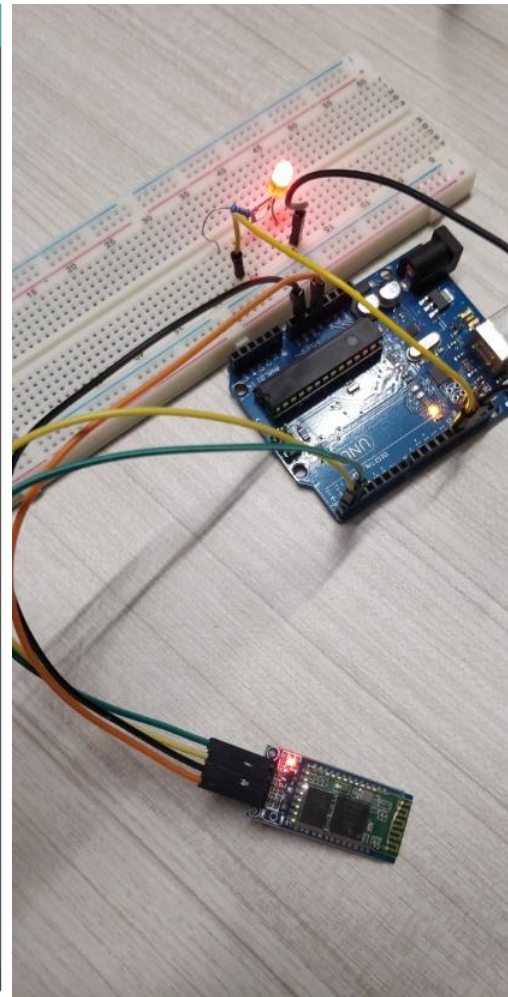


- Now click on LED/Lamp option





- Click on On=Off button to turn the LED on and off using you Bluetooth app.

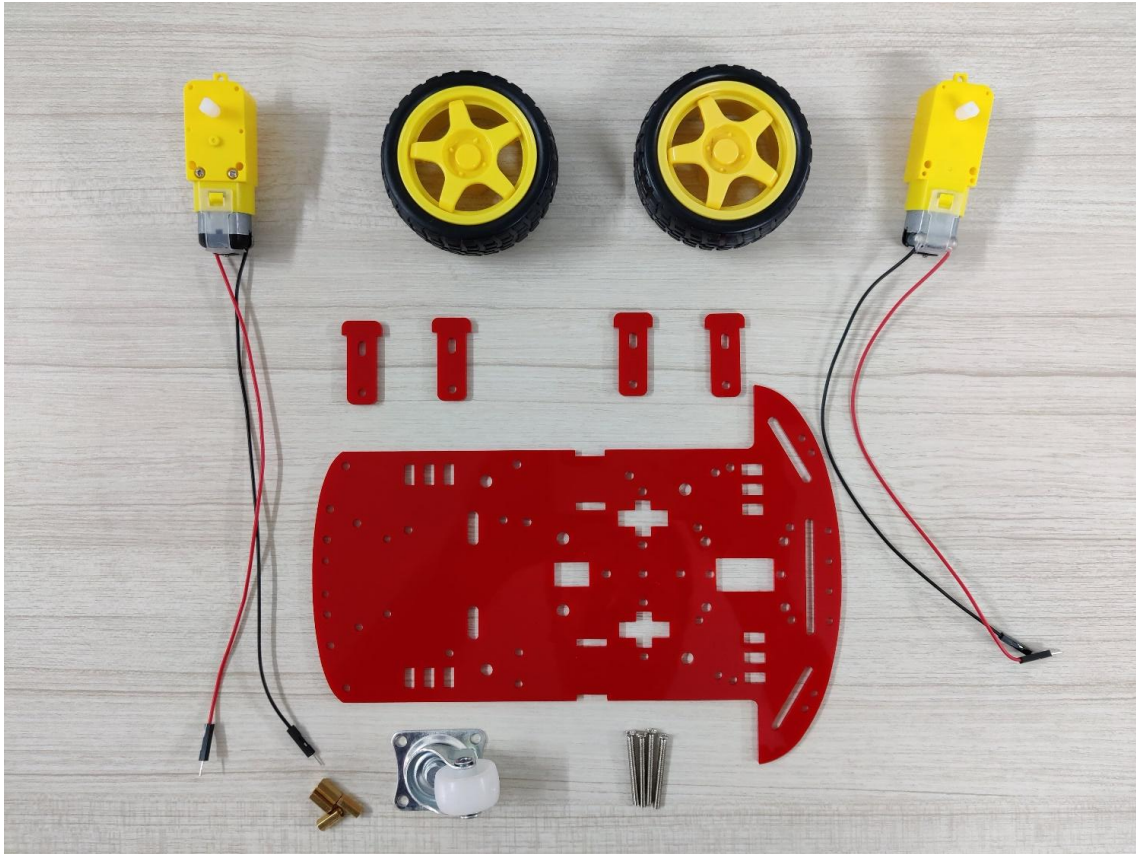


### Explanation & Output

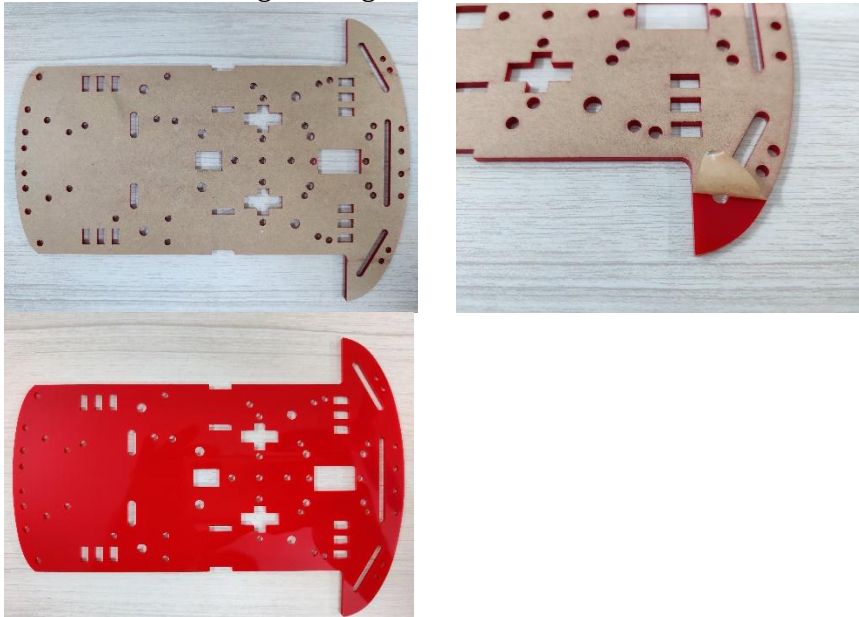
The Arduino receives Bluetooth commands from the mobile app and checks the character sent. If it receives '1' the LED on pin 9 turns ON, and if it receives '0' the LED turns OFF.

# Project 21: 2wd Car Chassis Assembly

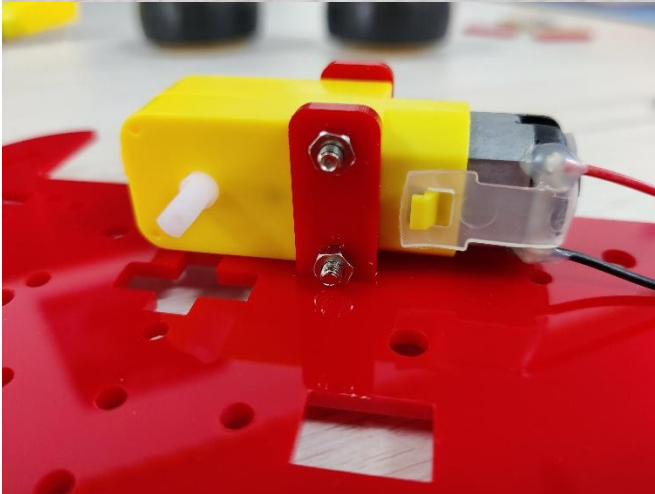
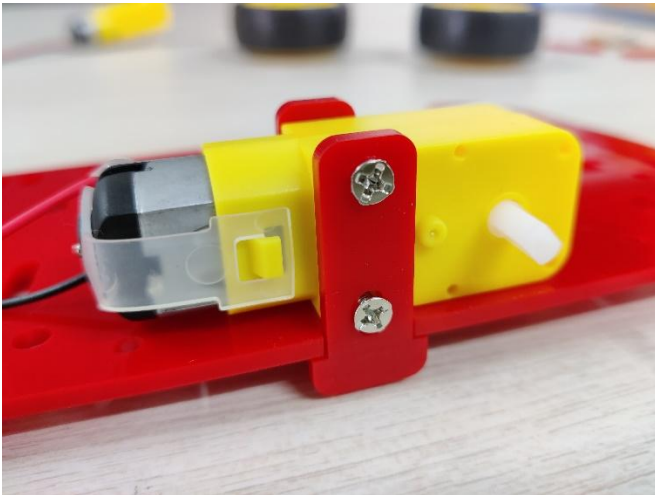
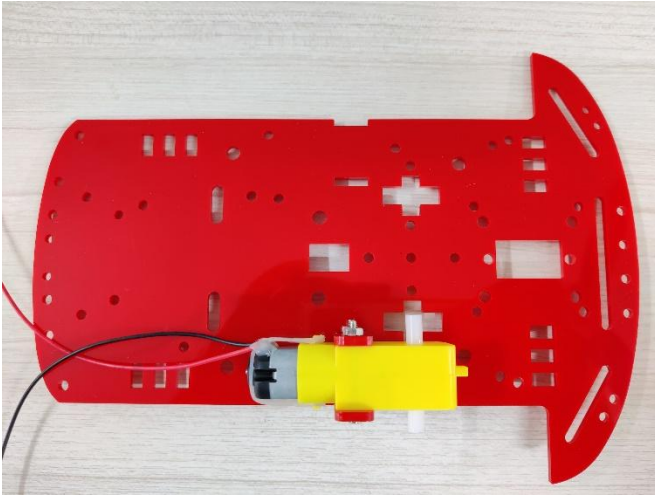
The basis for all the robot car is assembling the chassis and we will show you step by step how to assemble the car.



Step-1: Peel the outer sticker from the car chassis and the motor holders to make the chassis reveal its original bright color.

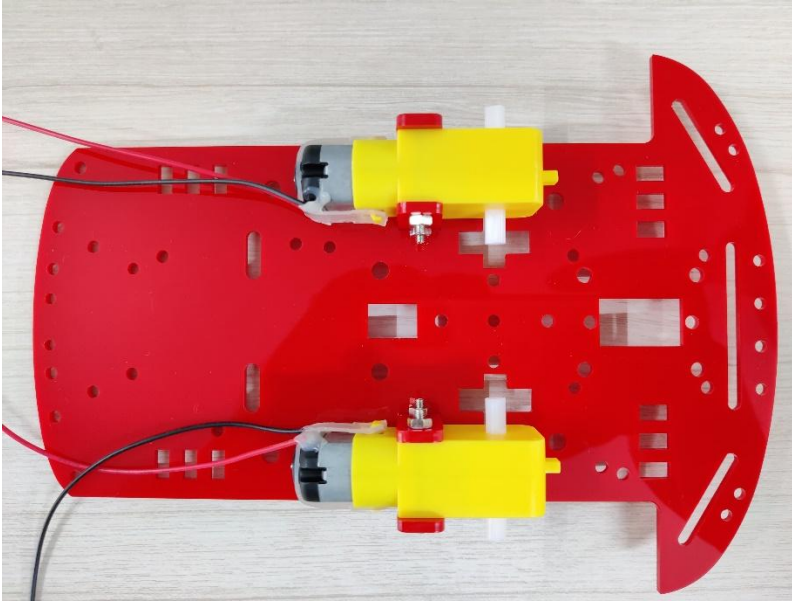


Step-2: Insert the motor holder into its respective slot and install the gear motor using the screws and nuts provided.

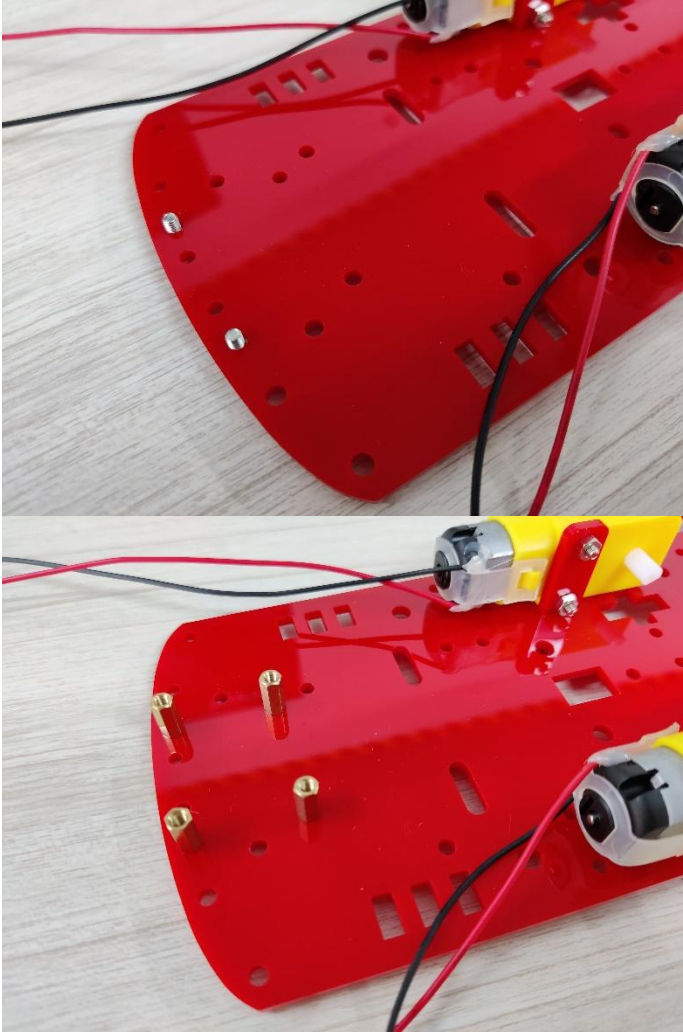




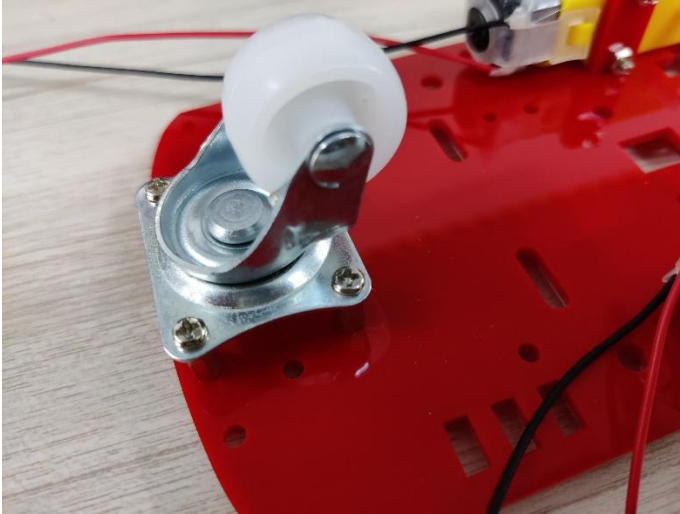
Step-3: Repeat the steps and install the gear motor in other side of the chassis as well.



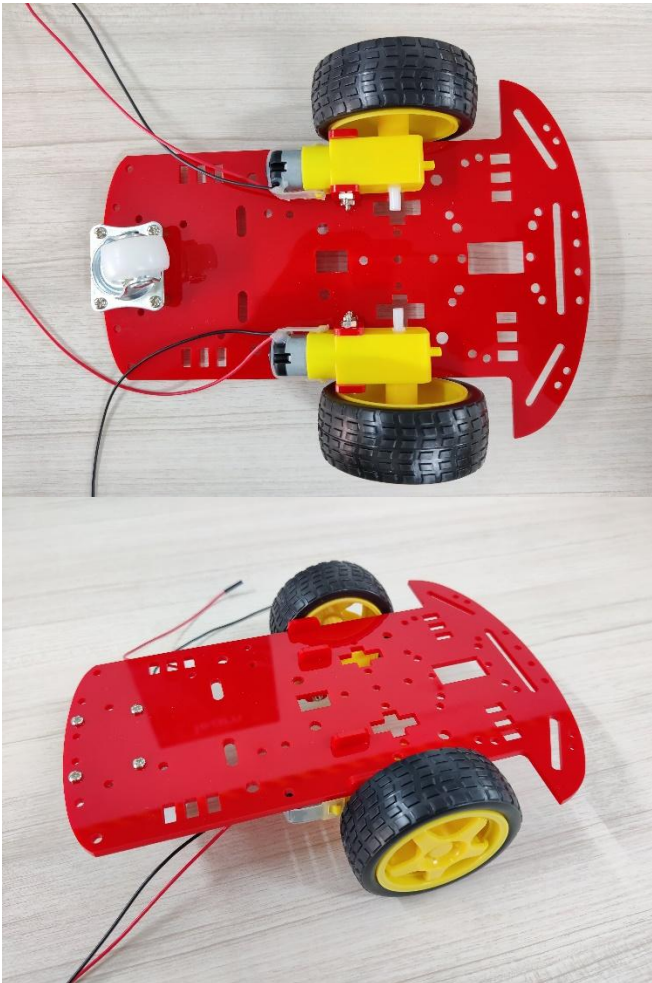
Step-4: Use the small screws and install all 4 Hex mount at the back end of the car chassis.



Step-5: Install the castor wheel on top of the hex mount using the screws.



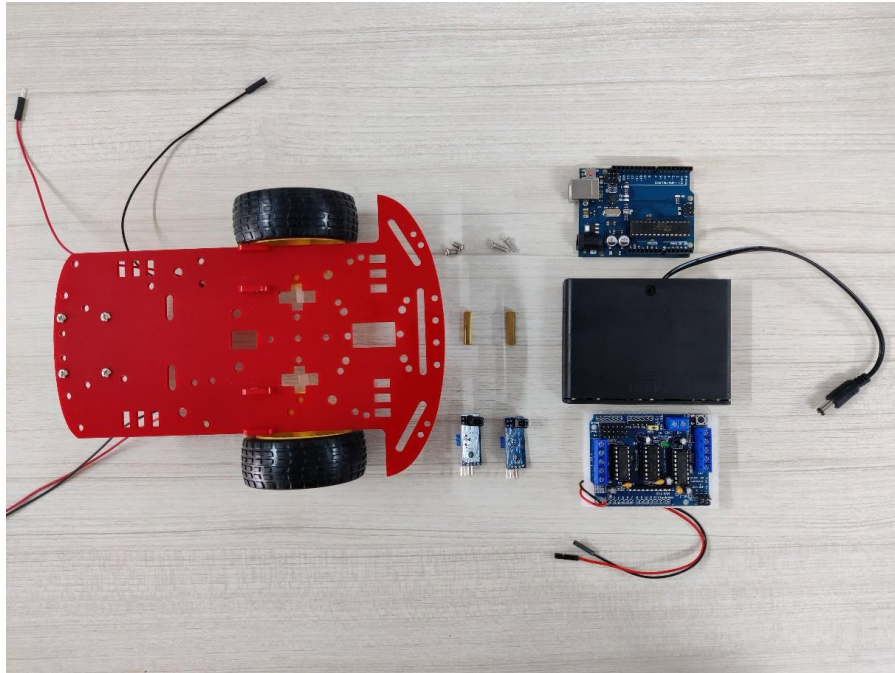
Step-6: Insert the wheels into the gear motor. Please align the wheel according the slot of the gear motor to insert properly. This completes the assembly instructions of 2WD car chassis.



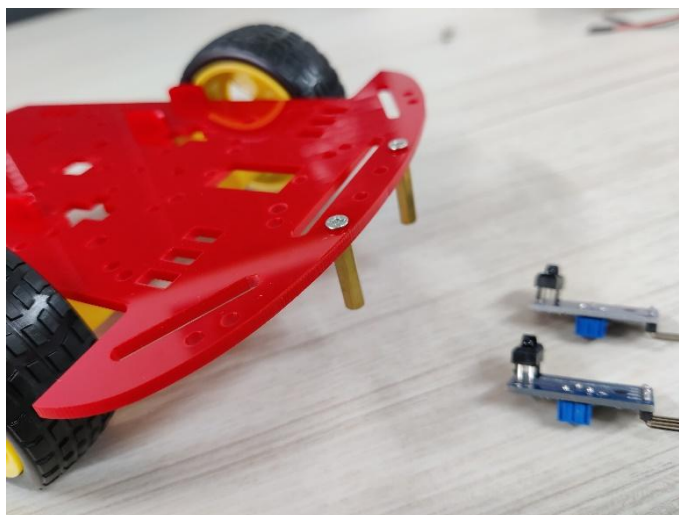
# Project 22: Line Following Car

In this project we will build a line following car which will follow the black line in a white background.

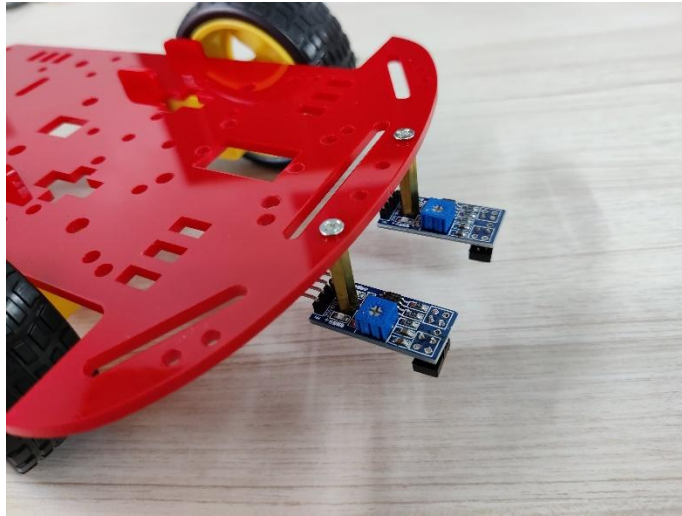
Step-1: Take an assembled 2WD Car chassis. Just in case you are not aware how to assemble the 2WD car chassis please refer to the “2WD Card Chassis Assembly Instruction” section for details.



Step-2: Install 2 hex connectors and screw the line sensors to them as shown in the picture.

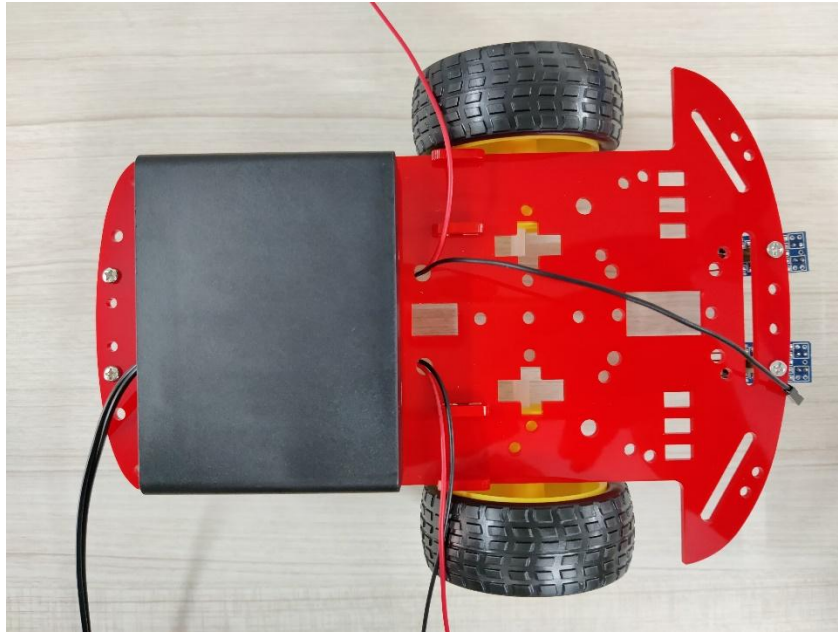




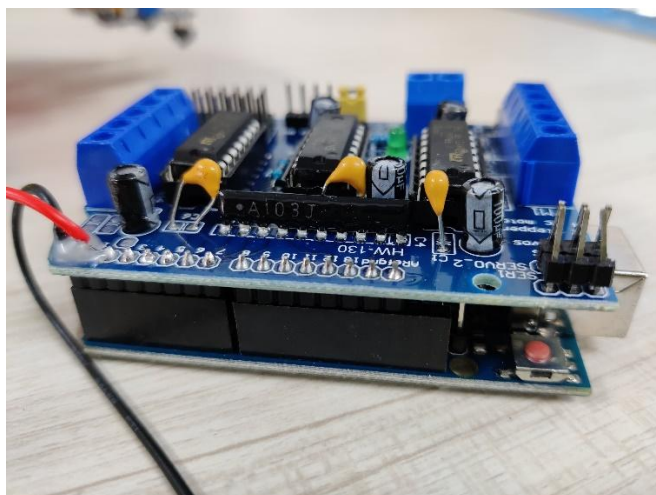
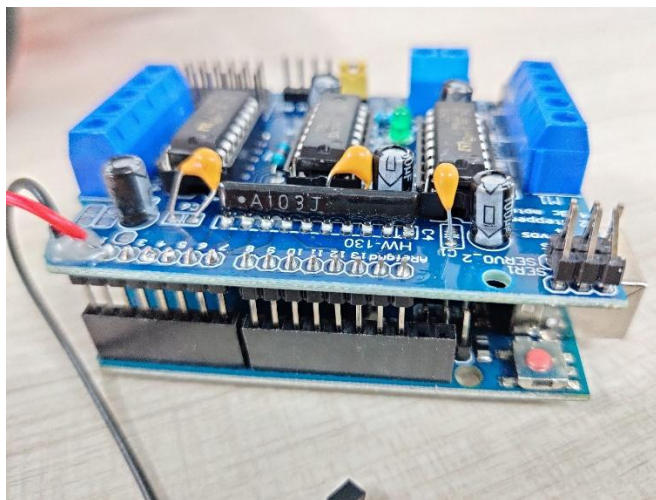


Step-3: Insert 6 x AA batteries in battery holder and place it in the back side of the car chassis using double side stickers.

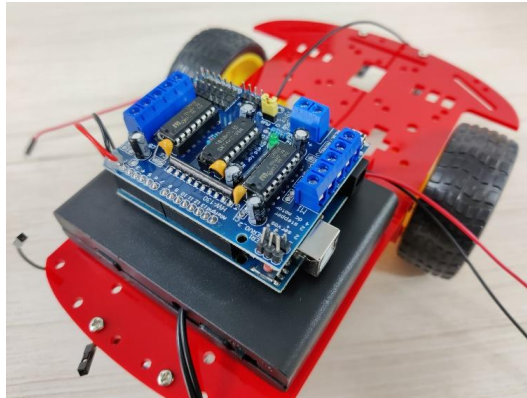
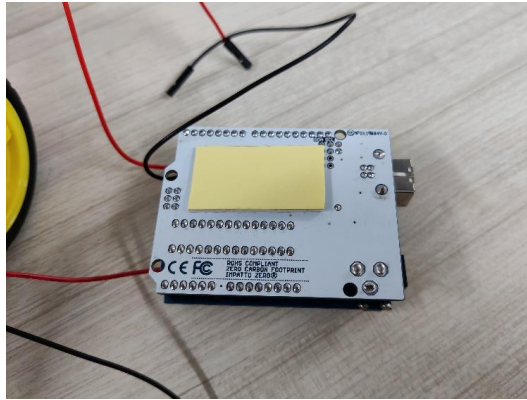




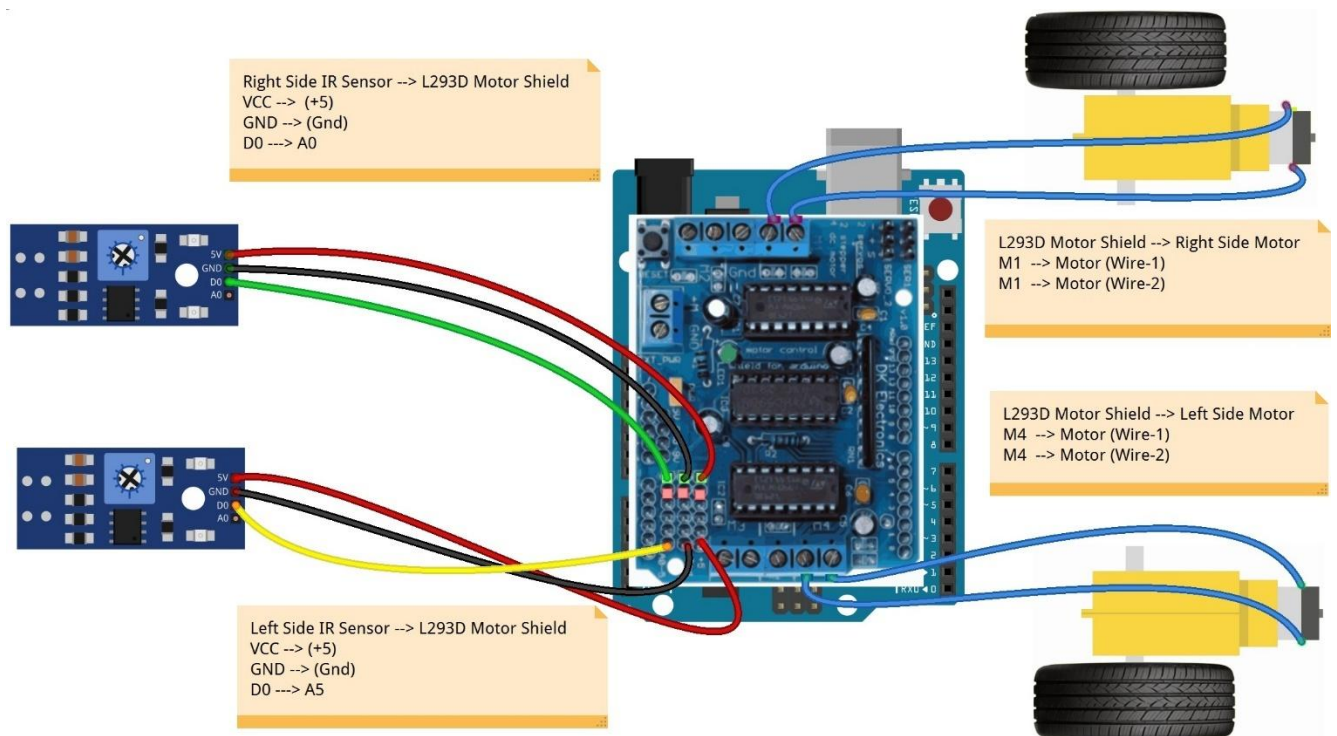
Step-4: Insert the L293D motor driver shield on top of the Uno R4 board and use a double side tape and fix it on top of the battery holder.







## Step-5: Circuit Connection

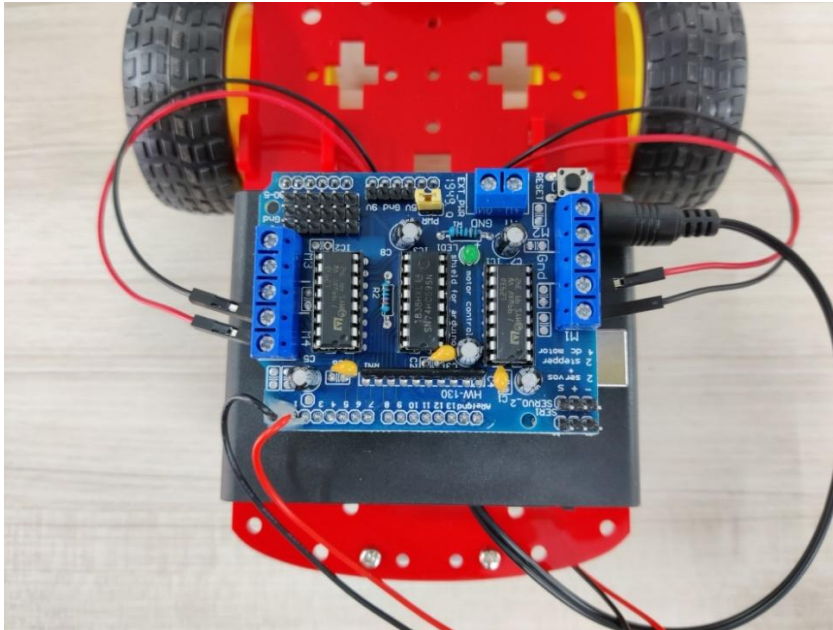


*Motors to L293D driver shield*

Left Side Motor	L293 Shield mounted on Uno R4
Wire-1 (Red)	M4
Wire-2 (Black)	M4

Right Side Motor	L293 Shield mounted on Uno R4
Wire-1 (Red)	M1
Wire-2 (Black)	M1

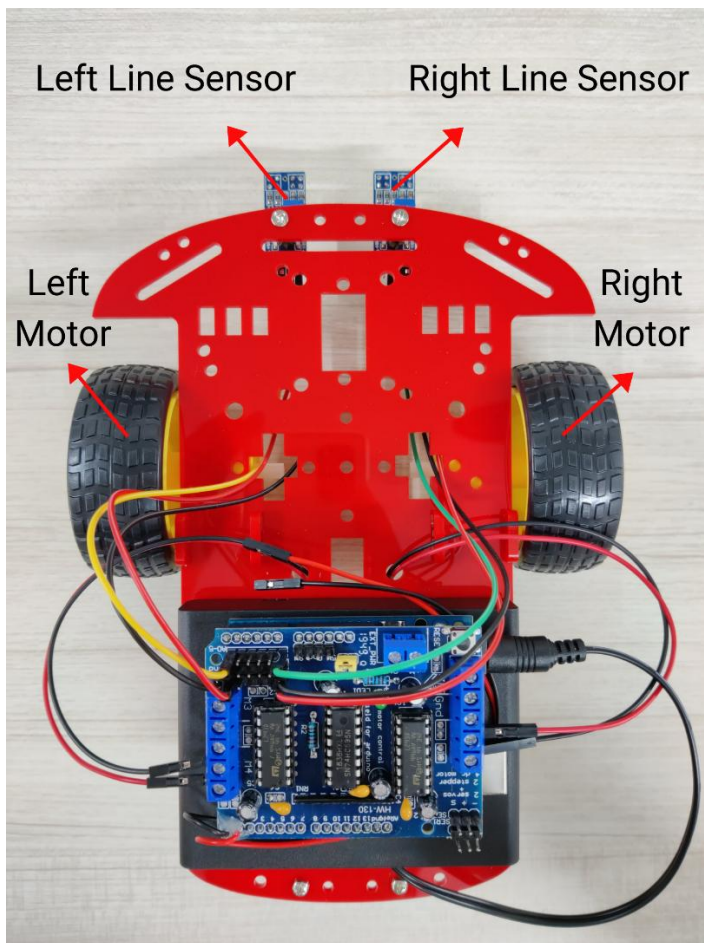
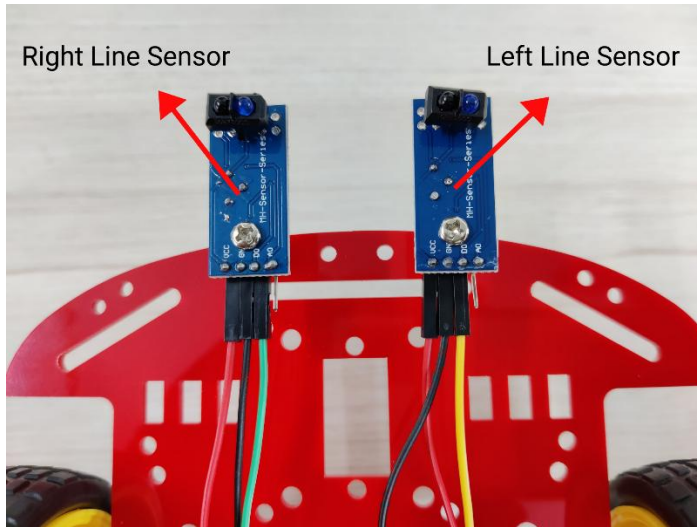
**Note:** kindly note that there is no polarity in gear motors. Hence the red and black wires provided with the motor is just for reference purpose. They can be interchanged if the motor is required to rotate in different directions.



Line Sensor to L293D driver shield

Left Side Line Sensor	L293 Shield mounted on Uno R4
VCC	+5
GND	Gnd
D0	A5

Right Side Line Sensor	L293 Shield mounted on Uno R4
VCC	+5
GND	Gnd
D0	A0



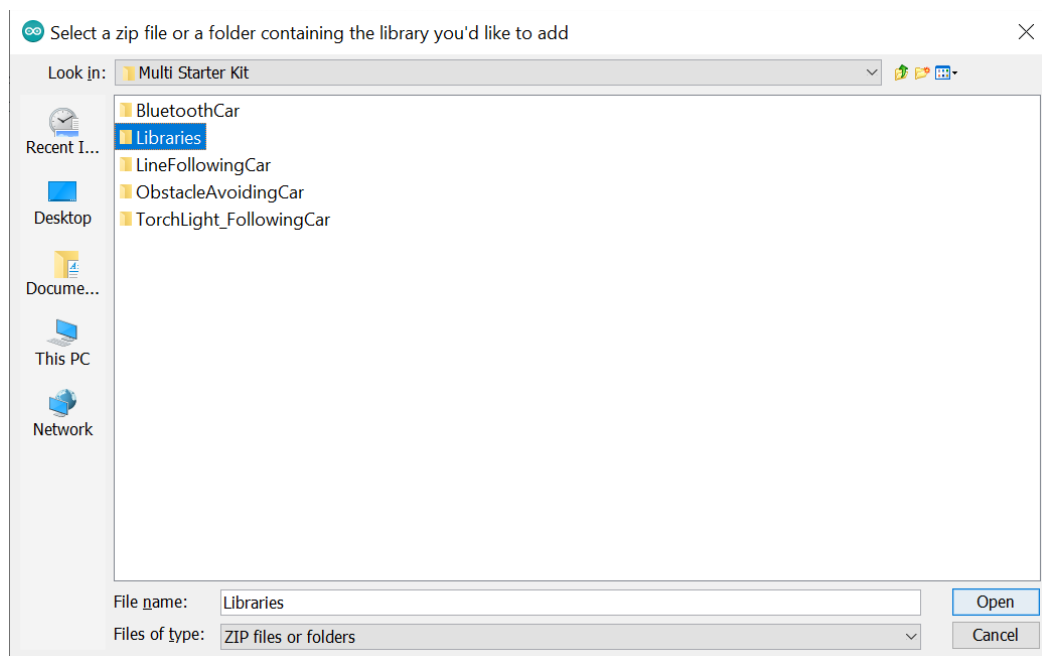
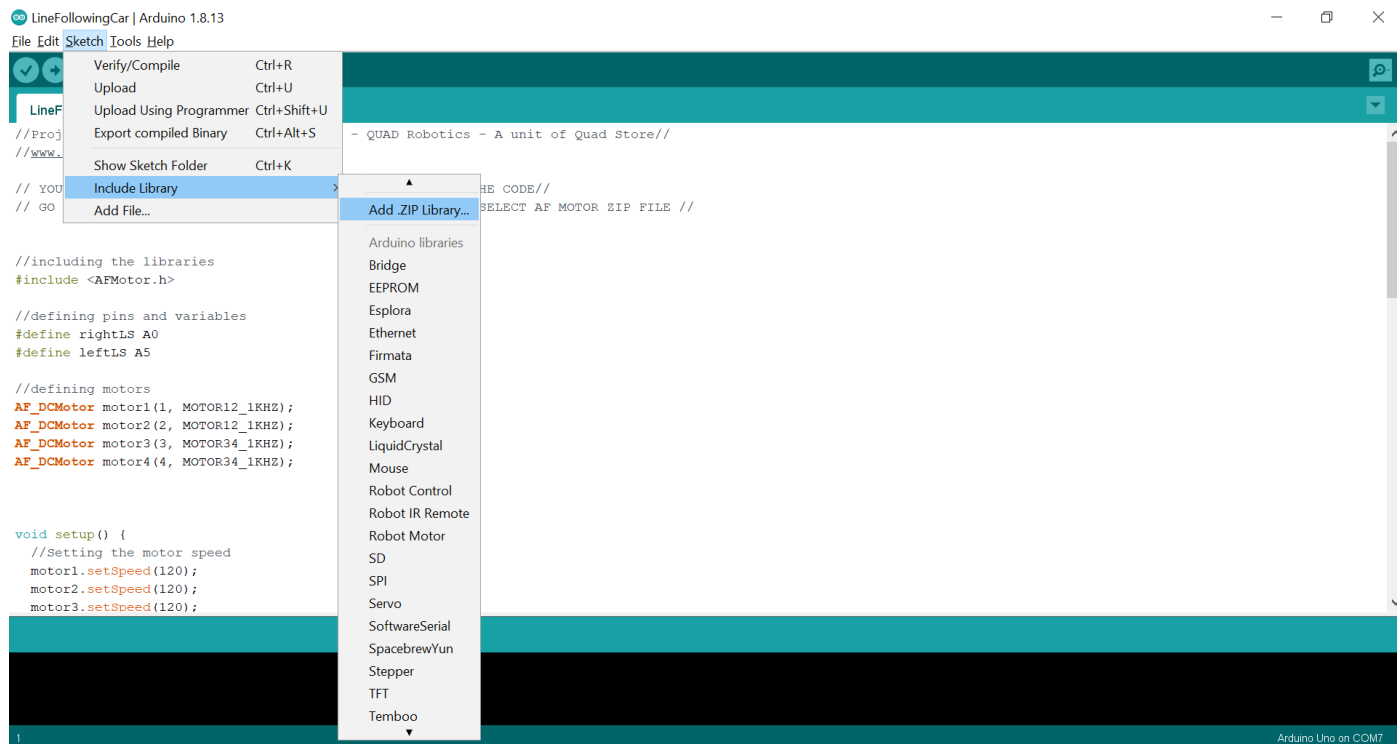
### Upload Code to Uno R4 board:

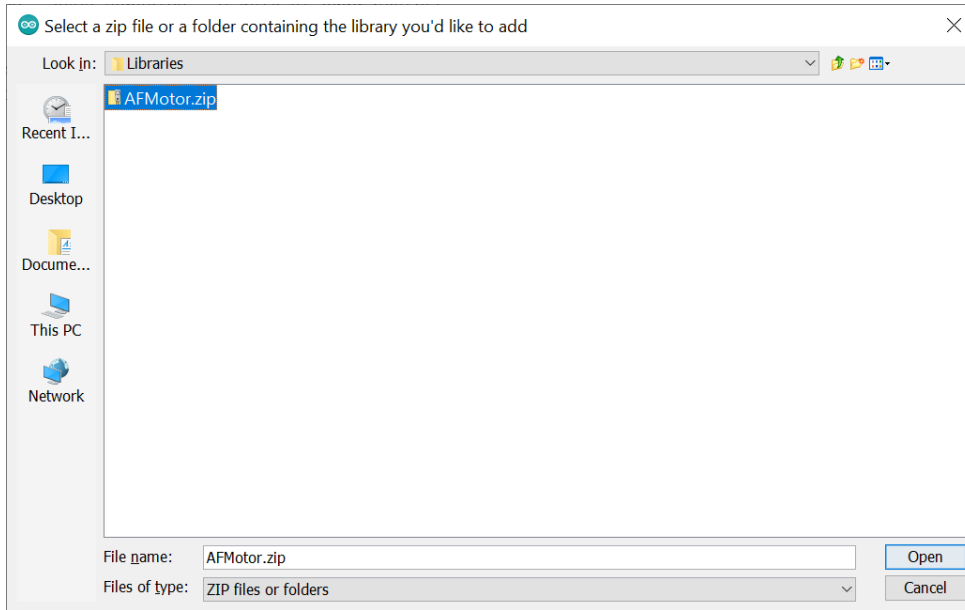
Use the provided USB cable and connect the Uno R4 board to your laptop/desktop computer.

### Installing Libraries:

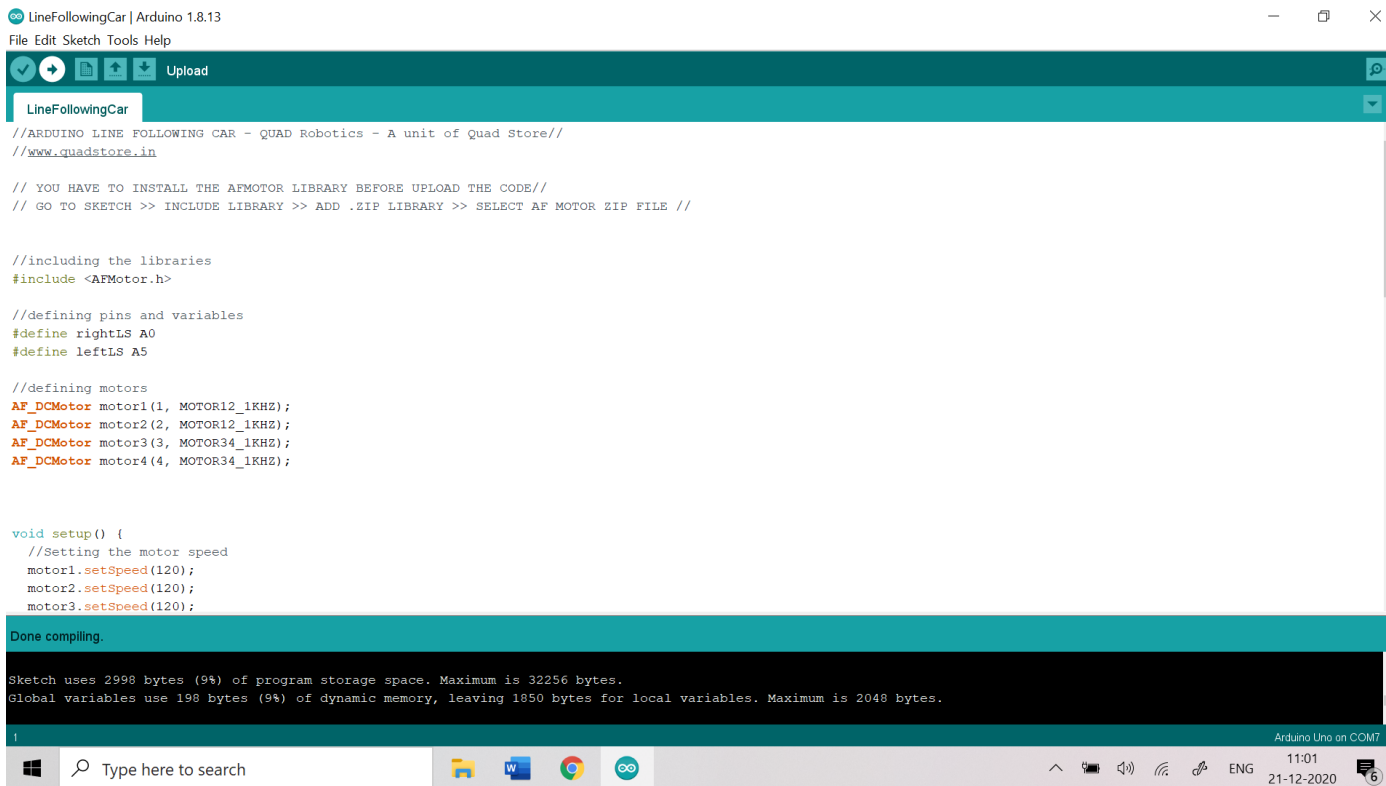
Open the **LineFollowingCar.ino** code using the Arduino IDE.

Before uploading the code to the Uno R4 board you need to install the “**AFMotor.zip**” library. Follow the below steps to install the library.





Compile and Upload the code to Uno R4 board.

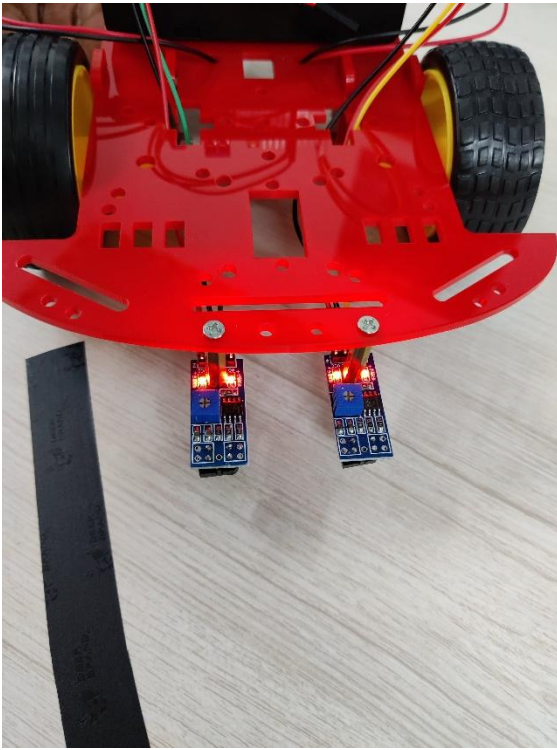


Step-6: Insert the DC power jack from battery holder into Uno R4 dc jack and Power ON the Uno R4 board. Now you should make sure the line sensors are working as expected.

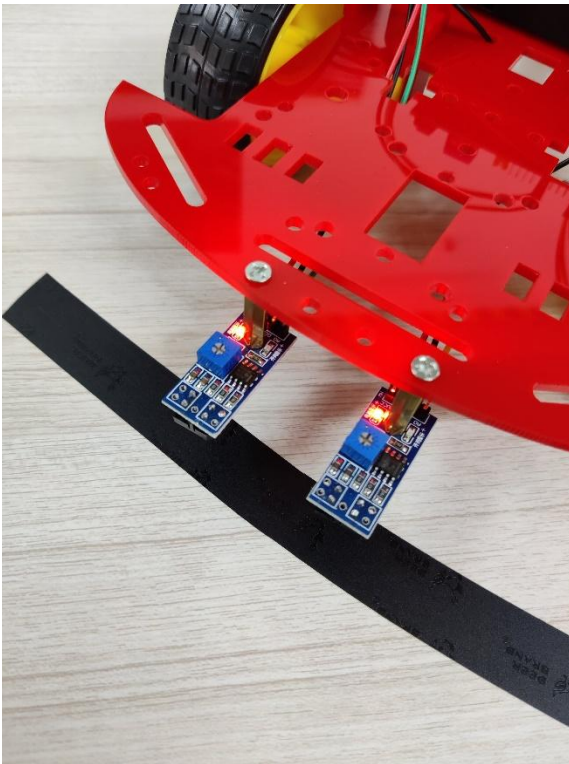
**Condition-1:** When line sensor is in white surface then you should notice the below points.

- You should see both LED's in each line sensor should be turned ON.
- Both wheels of the car should rotate in forward direction. If any of the wheels or both wheels rotation in reverse direction, then please interchange the red and black wire connection which goes to the L293D motor driver shield.





**Condition-2:** Black Surface: When line sensors is in black surface you see only one Led in each light sensor should be turned ON.



Note: If the above steps are not working as expected then please adjust the potentiometer in the line sensor using the screwdriver to its desired level to make it work.

Demonstration when one of the line sensor is in black surface and when the other one is in white surface. Make sure your line sensors also work in the same manner.



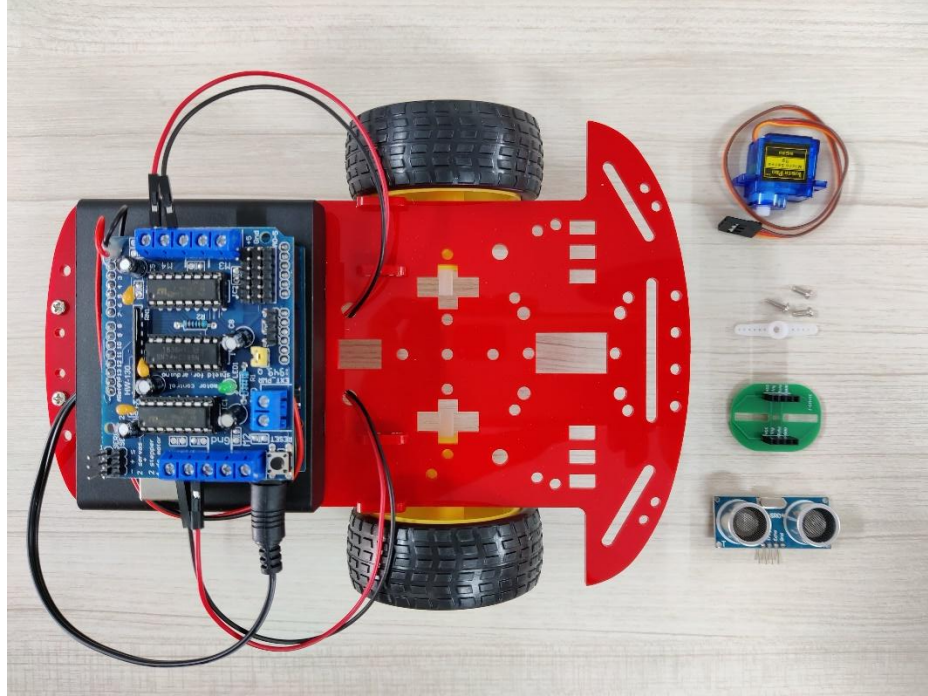
Step-7: Make a circuit in oval shape or round shape using black insulation tape. Kindly note the surface should be white with black line in it. If you are using reflective surfaces like floor, marble, granite, tiles it might not work well. So please use a chart paper or matt white surface to get the best results.



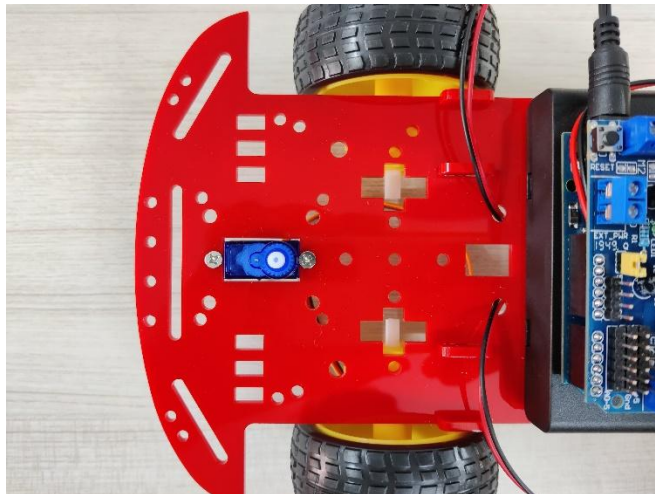


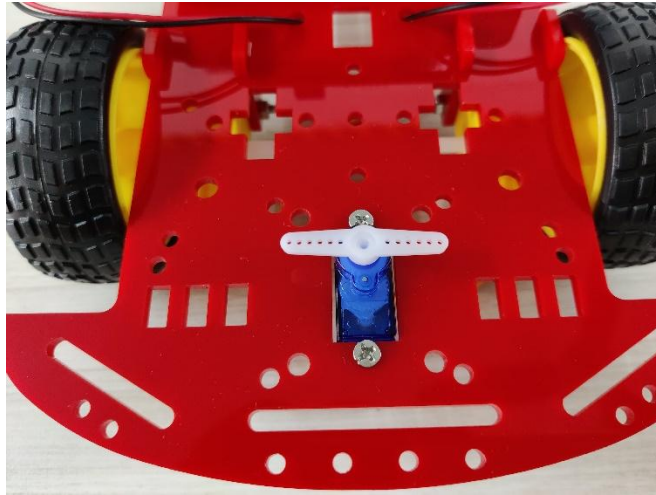
# Project 23: Obstacle Avoidance Car

Step-1: Take an assembled 2WD Car chassis. Just in case you are not aware how to assemble the 2WD car chassis please refer to the “2WD Card Chassis Assembly Instruction” section for details.

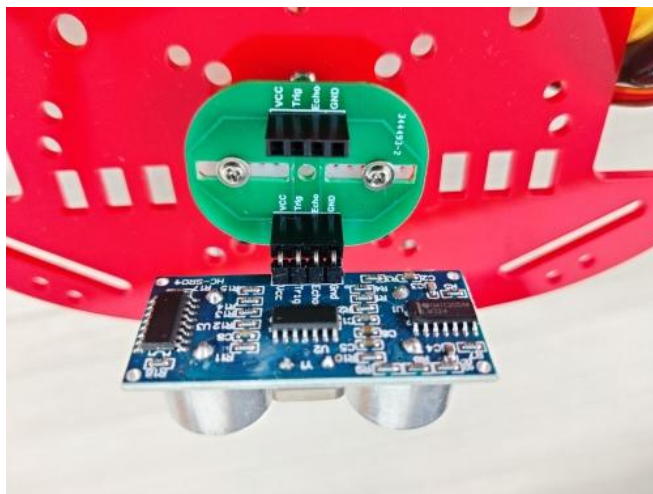
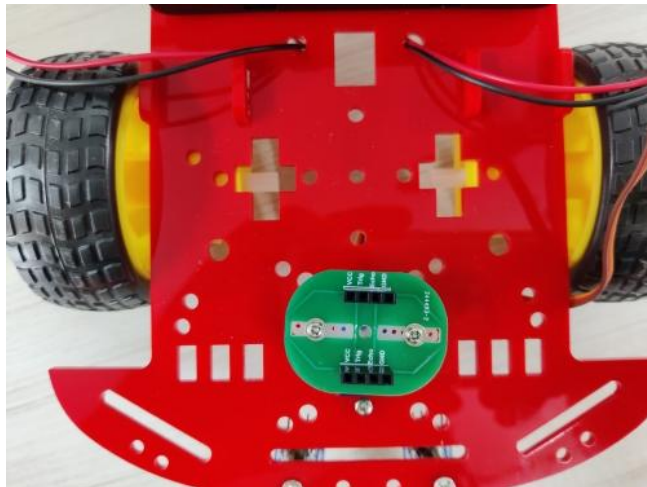


Step-2: Insert Servo motor from bottom to the servo slot and screw them. Insert servo arm to the servo.

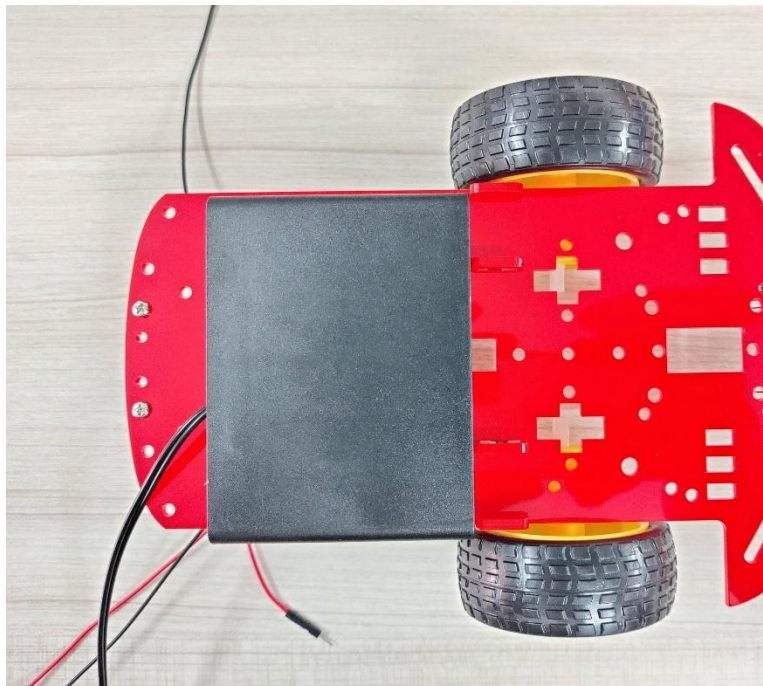
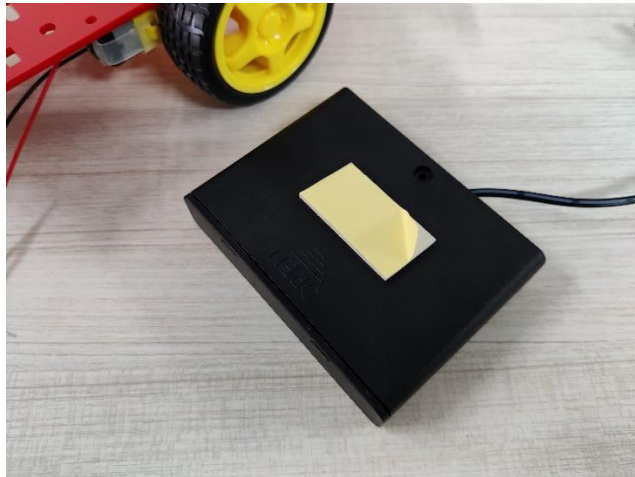




Step-3: Place the Ultrasonic sensor holder on the servo arm and screw them. Kindly note the direction in which the ultrasonic sensor holder is mounted, because it has to match with the pin terminals mentioned in the ultrasonic sensor. Insert the ultrasonics sensor into the holder.

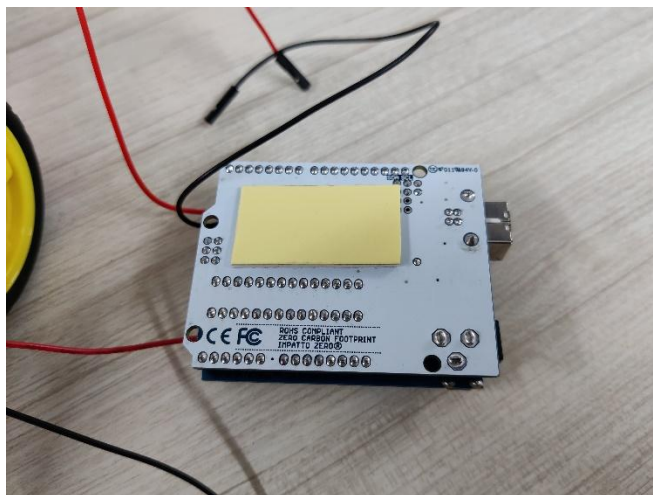
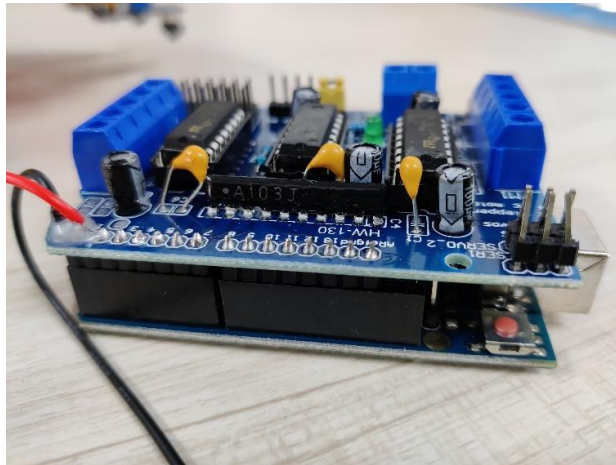
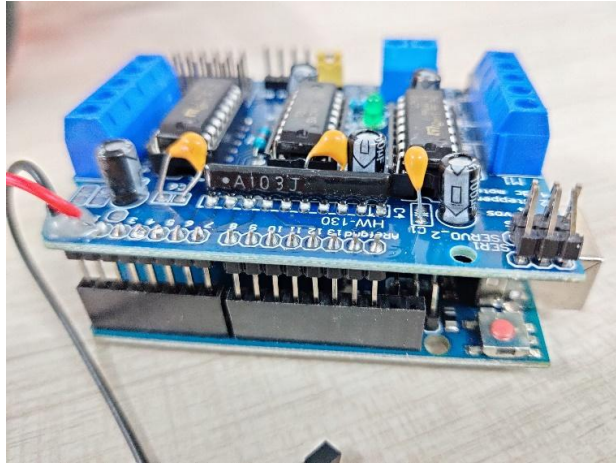


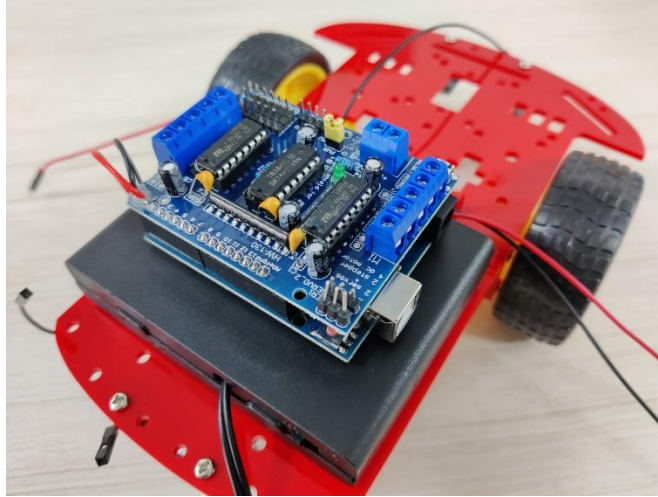
Step-4: Insert 6 x AA batteries in battery holder and place it in the back side of the car chassis using double side stickers.



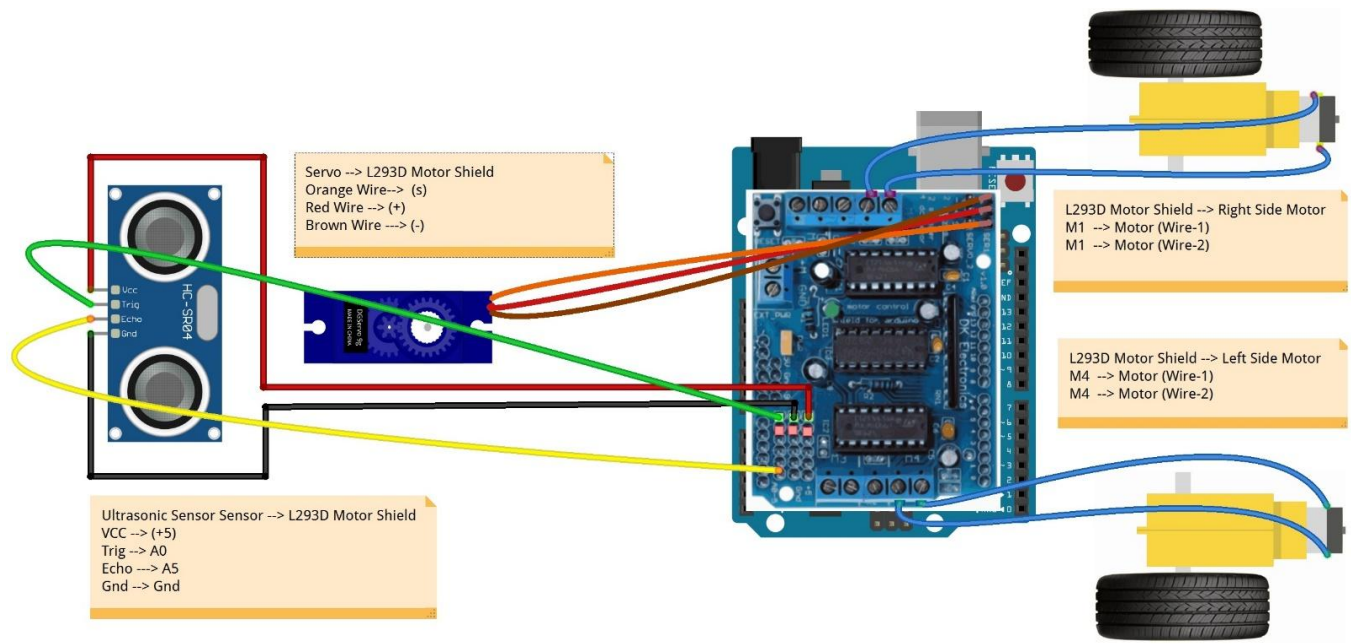


Step-5: Insert the L293D motor driver shield on top of the Uno R4 board and use a double side tape and fix it on top of the battery holder.





## Step-6: Circuit Connection

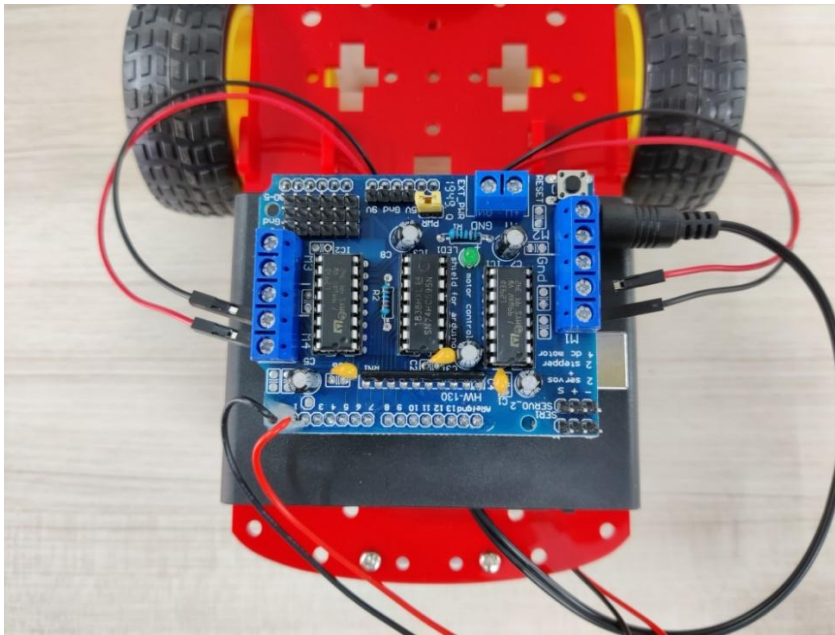


## Motors to L293D driver shield

Left Side Motor	L293 Shield mounted on Uno R4
Wire-1 (Red)	M4
Wire-2 (Black)	M4

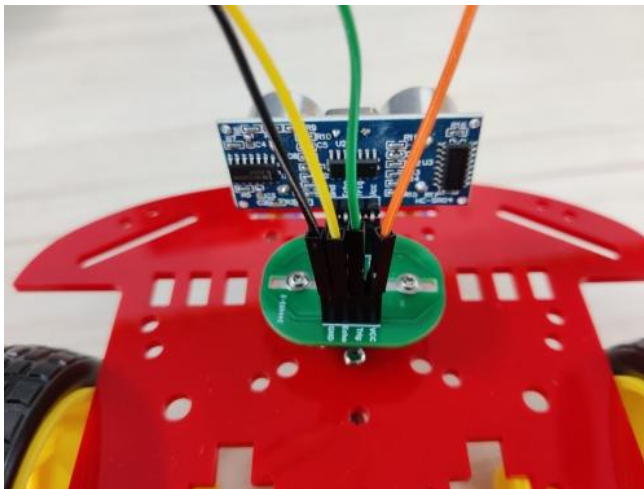
Right Side Motor	L293 Shield mounted on Uno R4
Wire-1 (Red)	M1
Wire-2 (Black)	M1

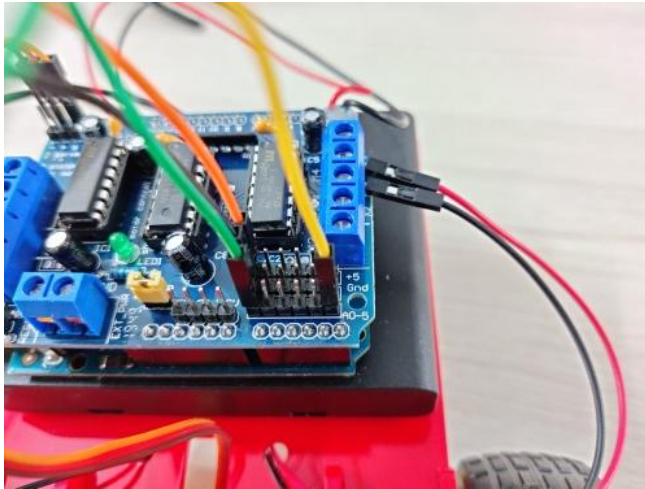
**Note:** kindly note that there is no polarity in gear motors. Hence the red and black wires provided with the motor is just for reference purpose. They can be interchanged if the motor is required to rotate in different directions.



Ultrasonic Sensor to L293D driver shield:

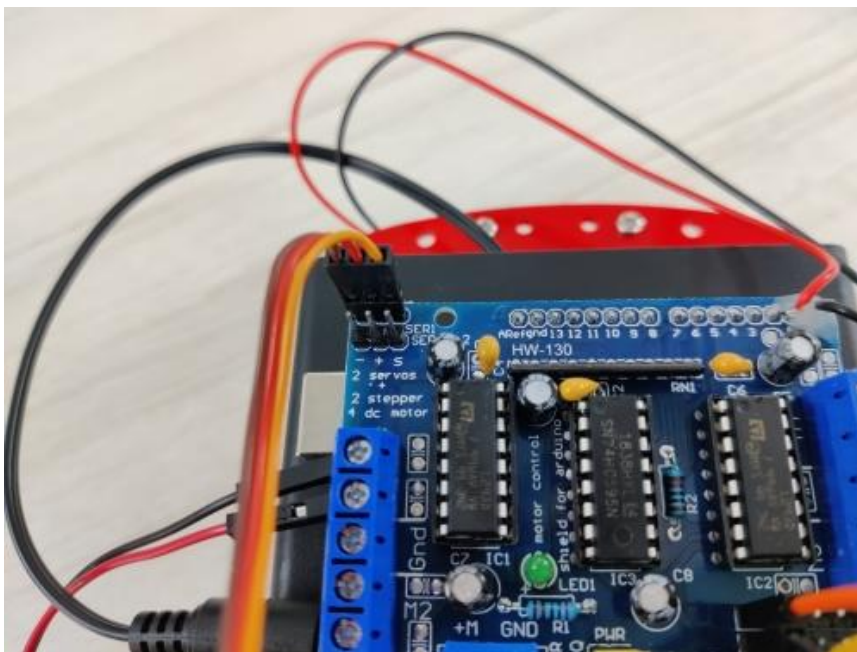
Ultrasonic Sensor	L293 Shield mounted on Uno R4
Gnd	Gnd
Echo	A5
Trig	A0
Vcc	+5



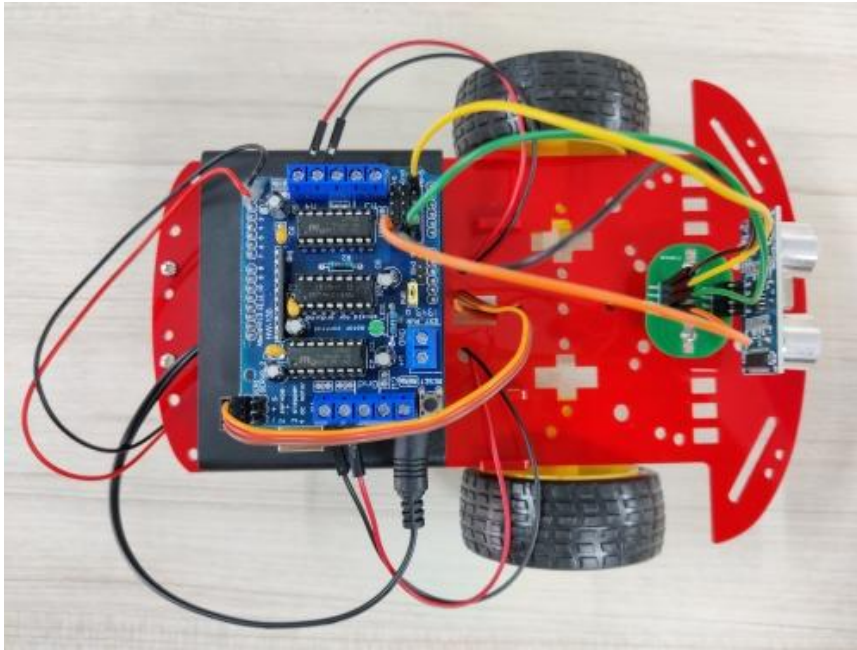


Servo Motor to L293D driver shield:

Servo Motor	L293 Shield mounted on Uno R4
Brown Color Wire	-
Red Color Wire	+
Orange color Wire	s







### Upload Code to Uno R4 board:

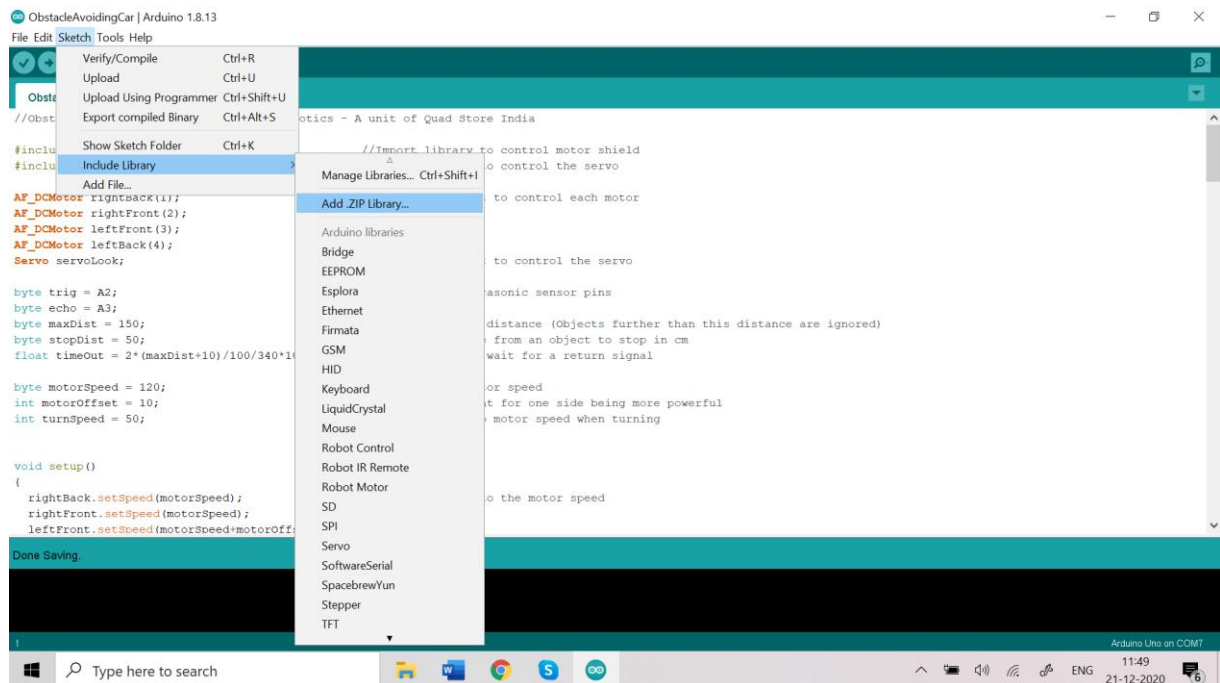
Use the provided USB cable and connect the Uno R4 board to your laptop/desktop computer.

### Installing Libraries:

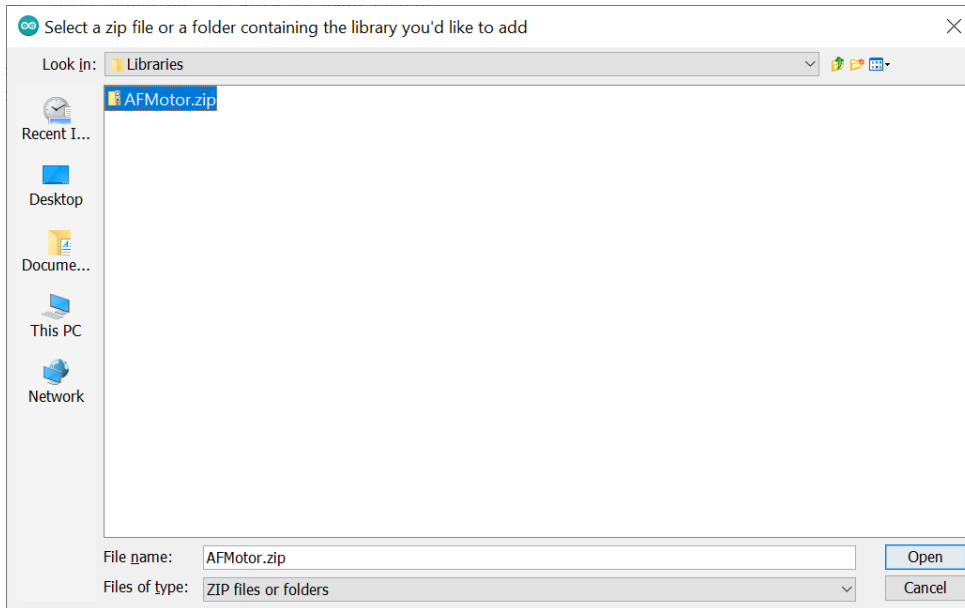
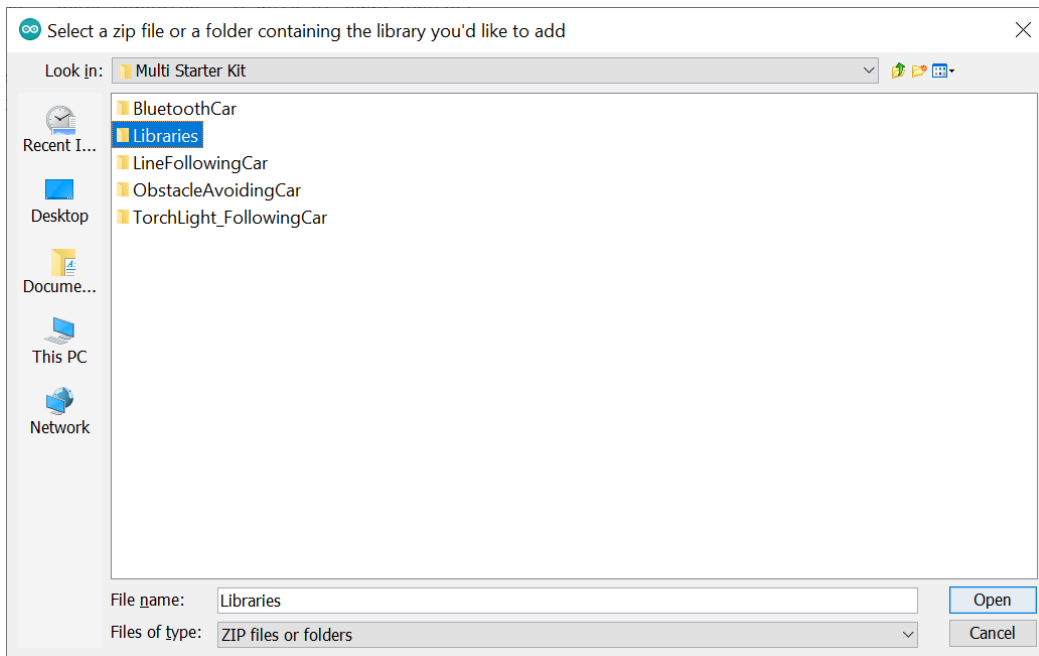
Open the **ObstacleAvoidingCar.ino** code using the Arduino IDE.

Before uploading the code to the Uno R4 board you need to install the “**AFMotor.zip**” library.

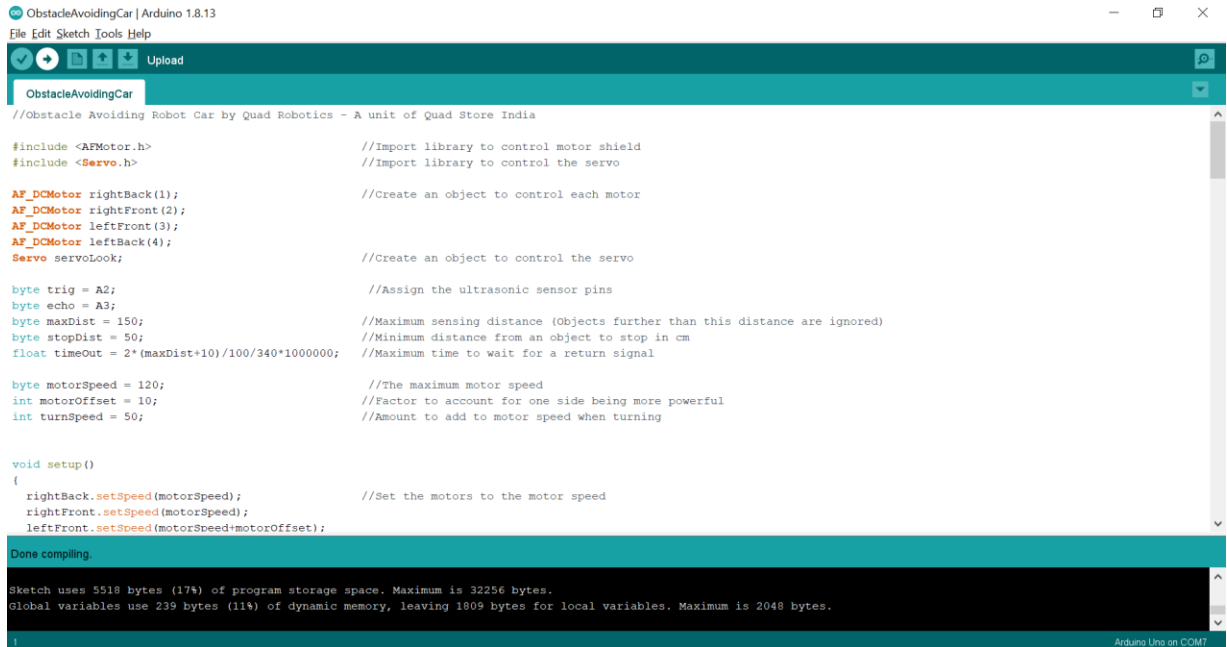
**NOTE:** If you had already installed this AFMotor.zip library during the previous projects then you can ignore this step. Else follow the below steps to install the library.







Compile and upload the code to Uno R4 board.



```
ObstacleAvoidingCar | Arduino 1.8.13
File Edit Sketch Tools Help
Upload
ObstacleAvoidingCar
//Obstacle Avoiding Robot Car by Quad Robotics - A unit of Quad Store India

#include <AFMotor.h> //Import library to control motor shield
#include <Servo.h> //Import library to control the servo

AF_DCMotor rightBack(1); //Create an object to control each motor
AF_DCMotor rightFront(2);
AF_DCMotor leftFront(3);
AF_DCMotor leftBack(4);
Servo servoLook; //Create an object to control the servo

byte trig = A2; //Assign the ultrasonic sensor pins
byte echo = A3;
byte maxDist = 150; //Maximum sensing distance (Objects further than this distance are ignored)
byte stopDist = 50; //Minimum distance from an object to stop in cm
float timeOut = 2*(maxDist+10)/100/340*1000000; //Maximum time to wait for a return signal

byte motorSpeed = 120; //The maximum motor speed
int motorOffset = 10; //Factor to account for one side being more powerful
int turnSpeed = 50; //Amount to add to motor speed when turning

void setup()
{
  rightBack.setSpeed(motorSpeed); //Set the motors to the motor speed
  rightFront.setSpeed(motorSpeed);
  leftFront.setSpeed(motorSpeed+motorOffset);
}

Done compiling.
Sketch uses 5518 bytes (17%) of program storage space. Maximum is 32256 bytes.
Global variables use 239 bytes (11%) of dynamic memory, leaving 1809 bytes for local variables. Maximum is 2048 bytes.
Arduino Uno on COM7
```

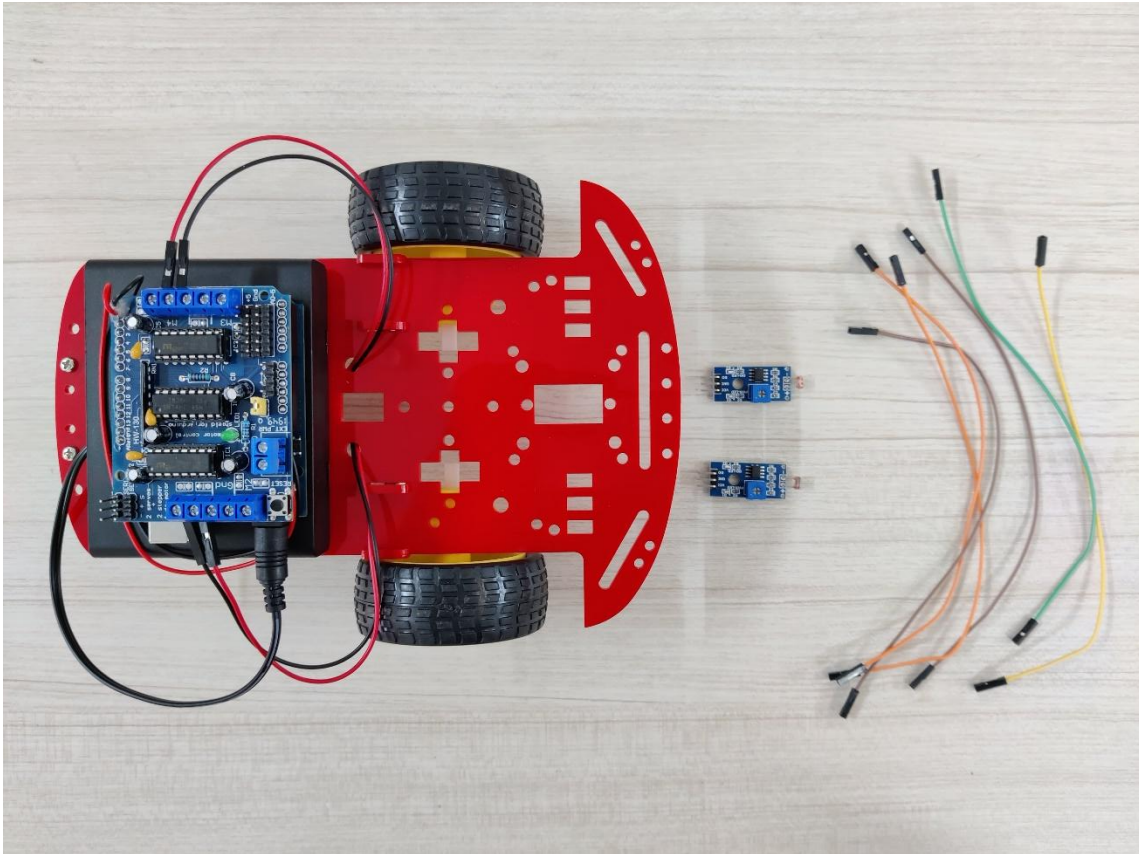
Step-7: Insert the DC power jack from battery holder into Uno R4 dc jack and Power ON the Uno R4 board.

Result: You will see the car avoiding the obstacles and going towards the longest path available where there is no obstacle in front of it.

# Project 24: Light Following Car

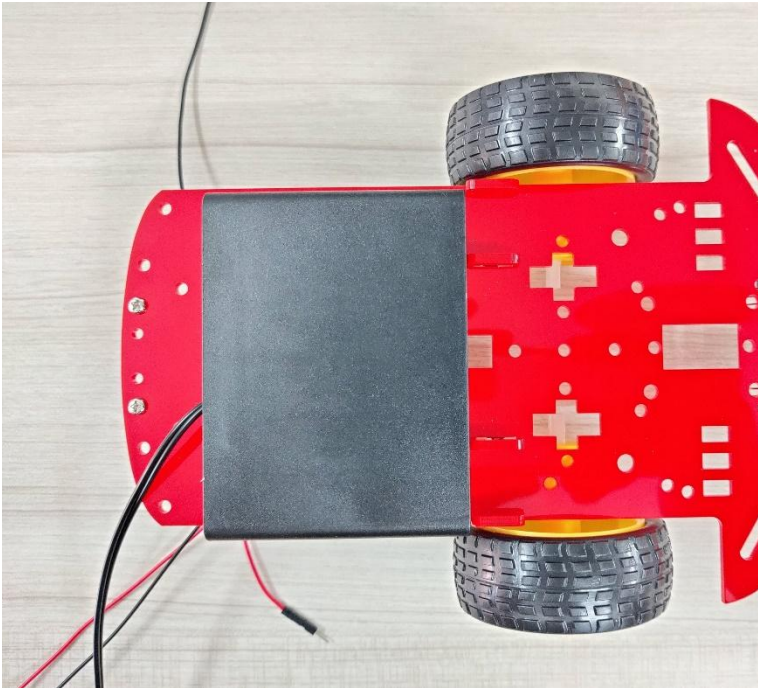
In this project we will build a light following car which will follow the light in the dark.

Step-1: Take an assembled 2WD Car chassis. Just in case you are not aware how to assemble the 2WD car chassis please refer to the “2WD Card Chassis Assembly Instruction” section for details.

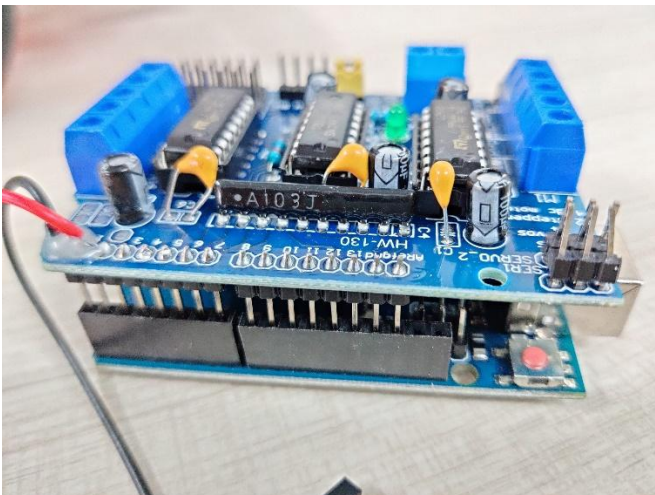


Step-2: Insert 6 x AA batteries in battery holder and place it in the back side of the car chassis using double side stickers.

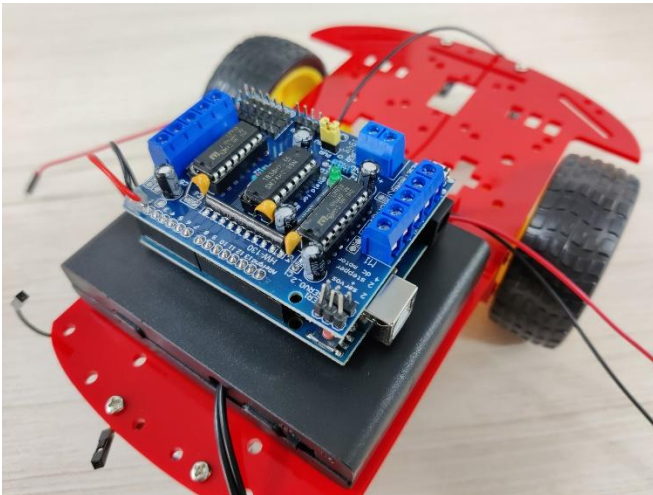
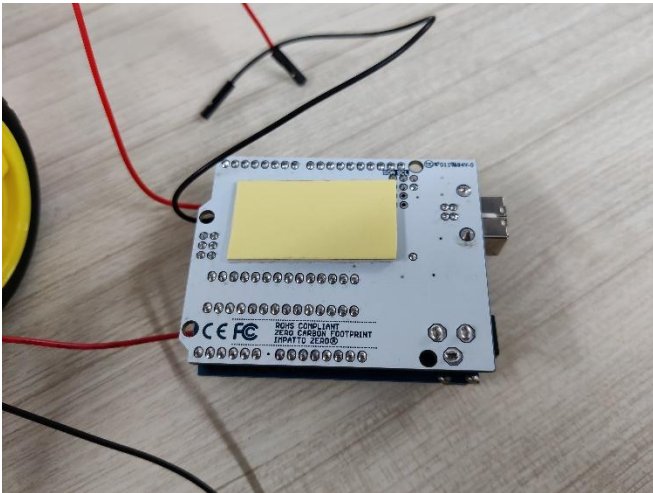
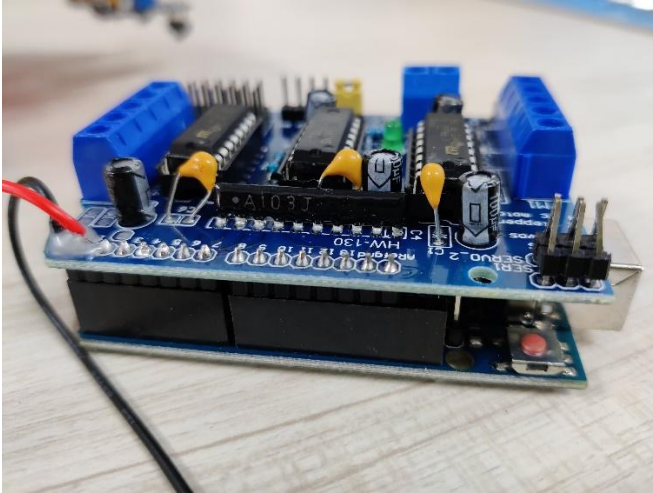




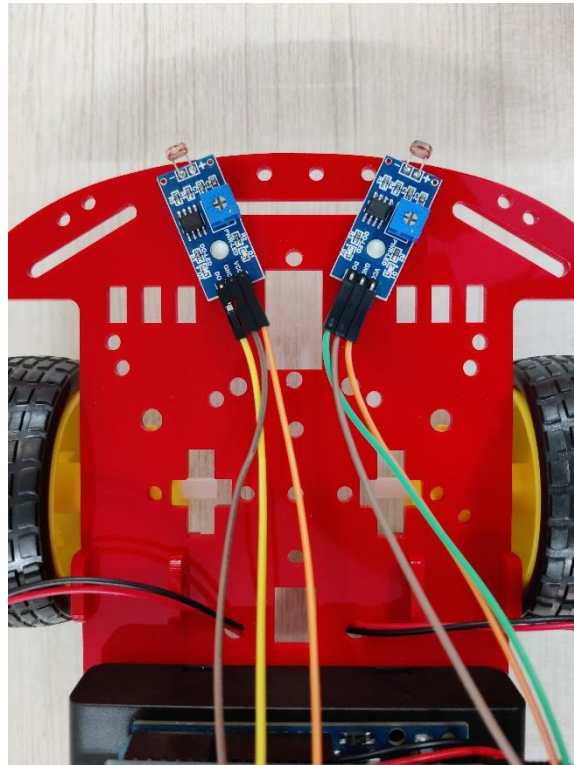
Step-3: Insert the L293D motor driver shield on top of the Uno R4 board and use a double side tape and fix it on top of the battery holder.



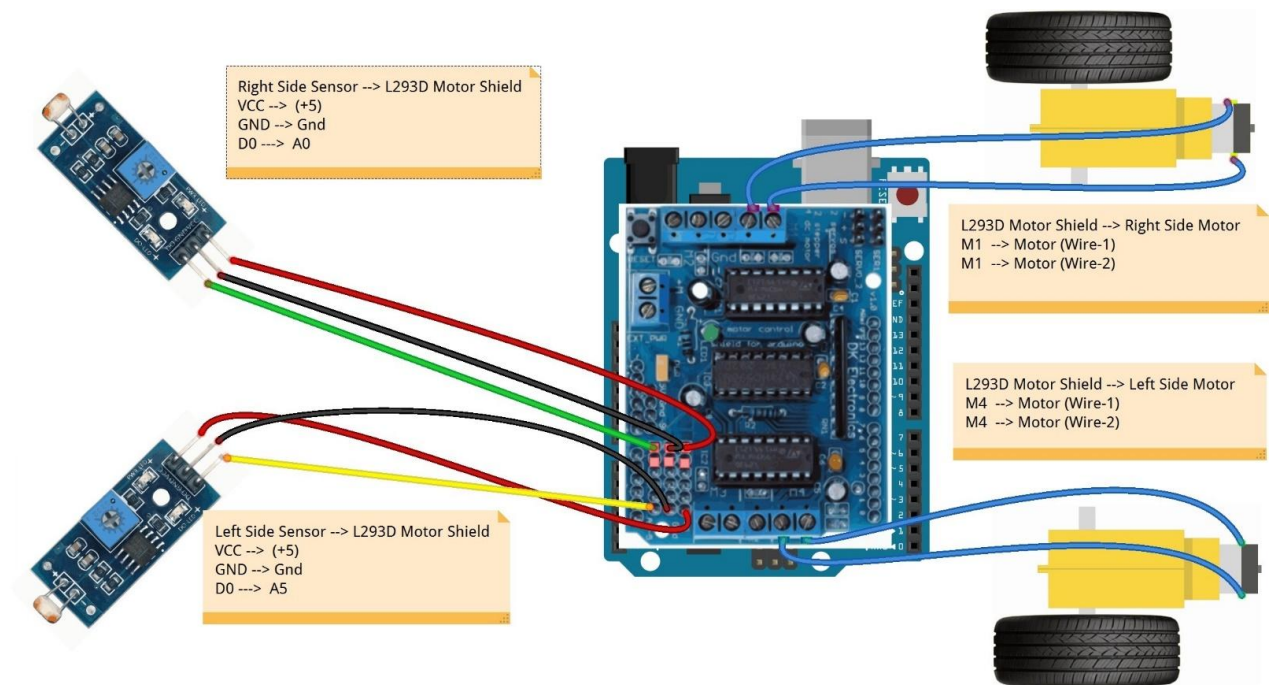




Step-4: Use double side tape and fix the Photoresistor in the 45 degree angle as shown in the picture below.



## Step-6: Circuit Connection

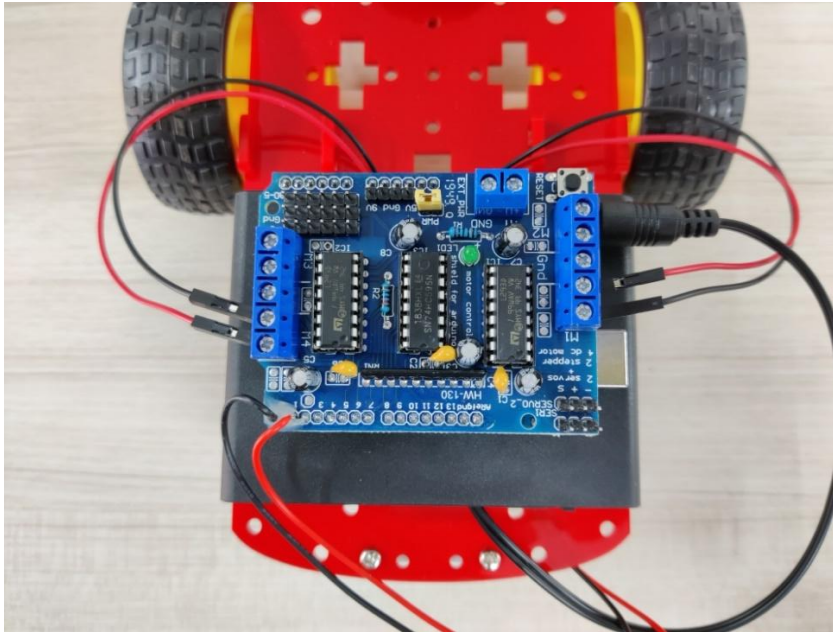


## Motors to L293D driver shield

Left Side Motor	L293 Shield mounted on Uno R4
Wire-1 (Red)	M4
Wire-2 (Black)	M4

Right Side Motor	L293 Shield mounted on Uno R4
Wire-1 (Red)	M1
Wire-2 (Black)	M1

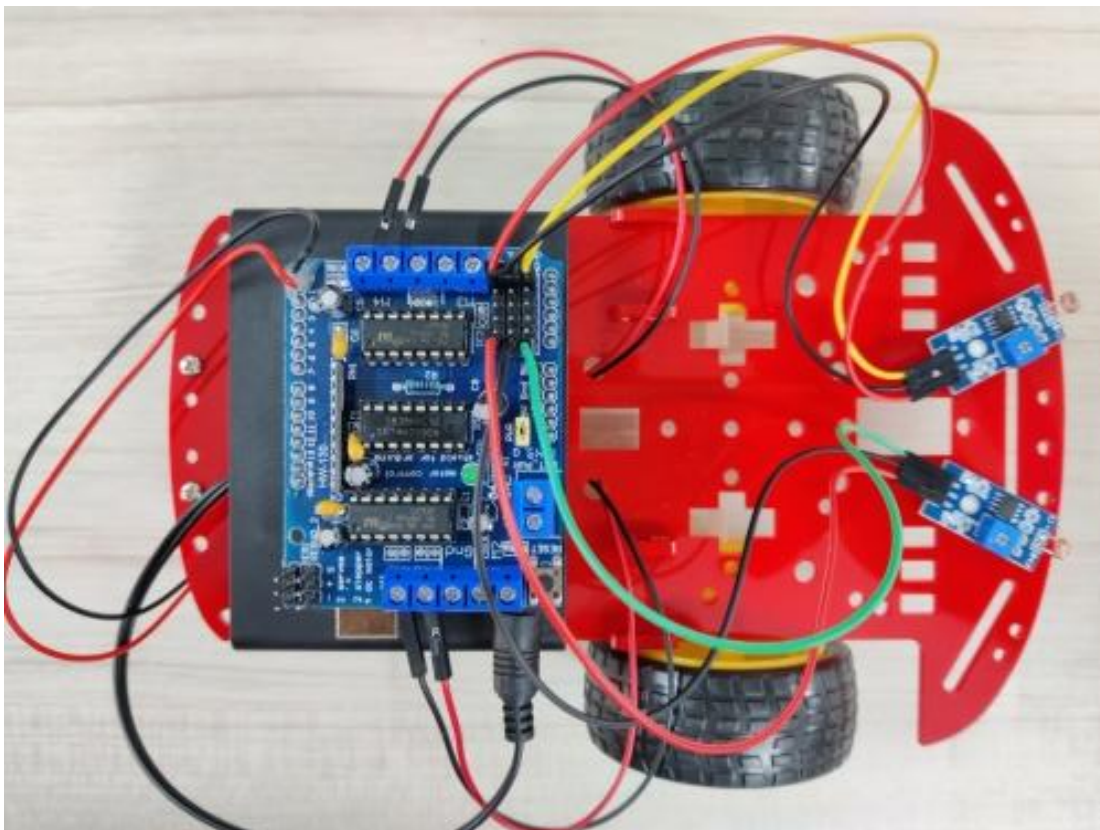
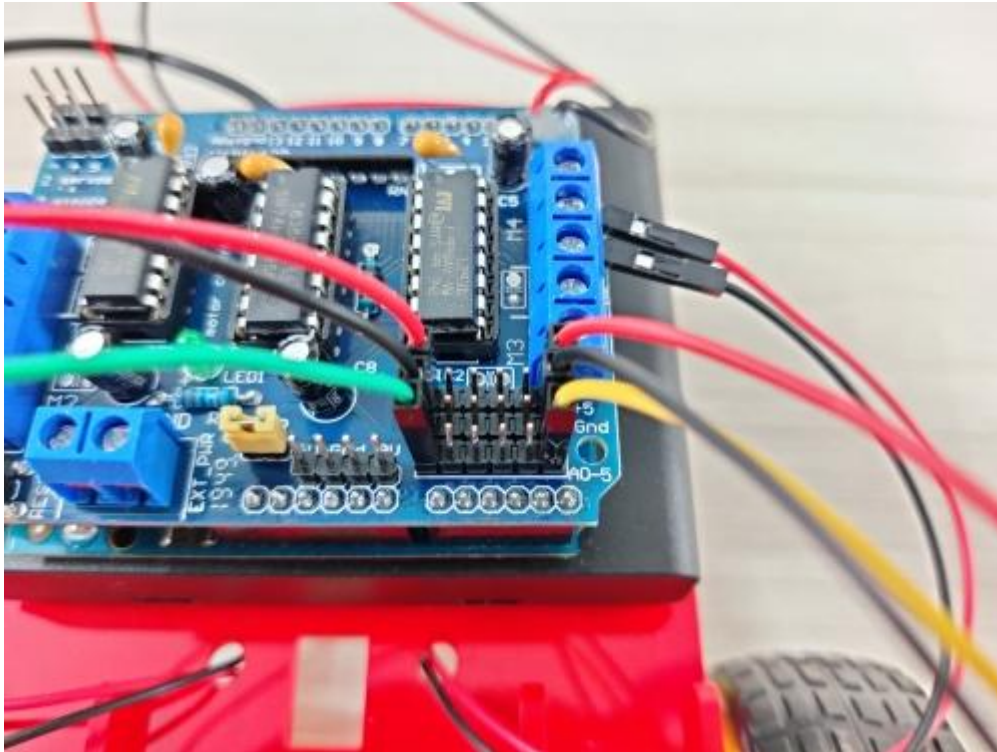
**Note:** kindly note that there is no polarity in gear motors. Hence the red and black wires provided with the motor is just for reference purpose. They can be interchanged if the motor is required to rotate in different directions.



Left Side Photoresistor	L293 Shield mounted on Uno R4
VCC	+5
GND	Gnd
D0	A5

Right Side Photoresistor	L293 Shield mounted on Uno R4
VCC	+5
GND	Gnd
D0	A0





### **Upload Code to Uno R4 board:**

Use the provided USB cable and connect the Uno R4 board to your laptop/desktop computer.

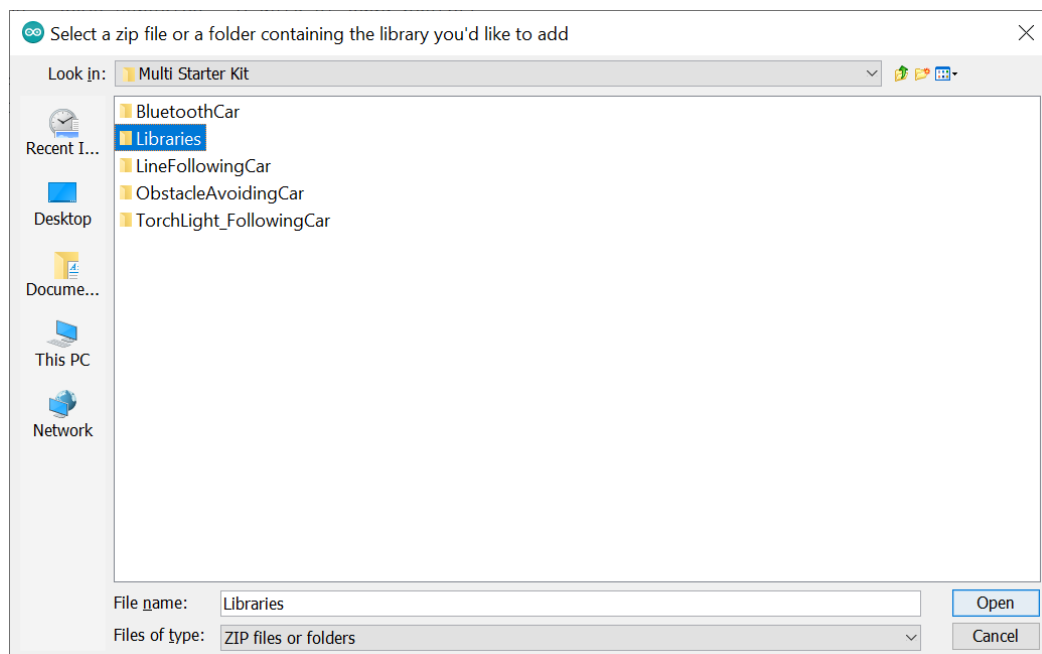
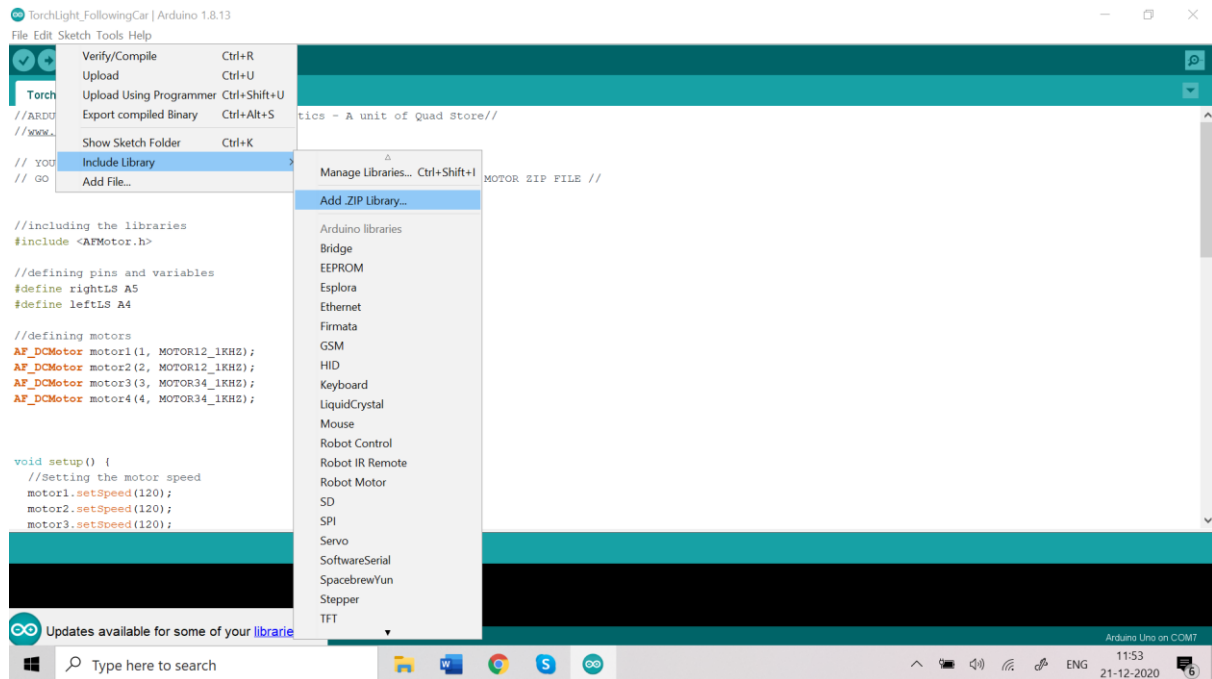


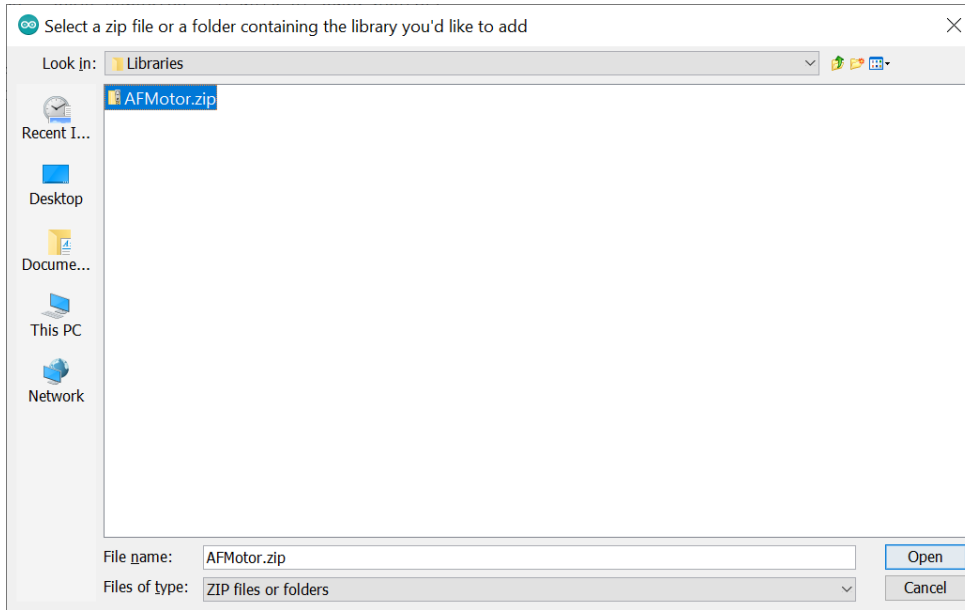
## Installing Libraries:

Open the **TorchLight\_FollowingCar.ino** code using the Arduino IDE.

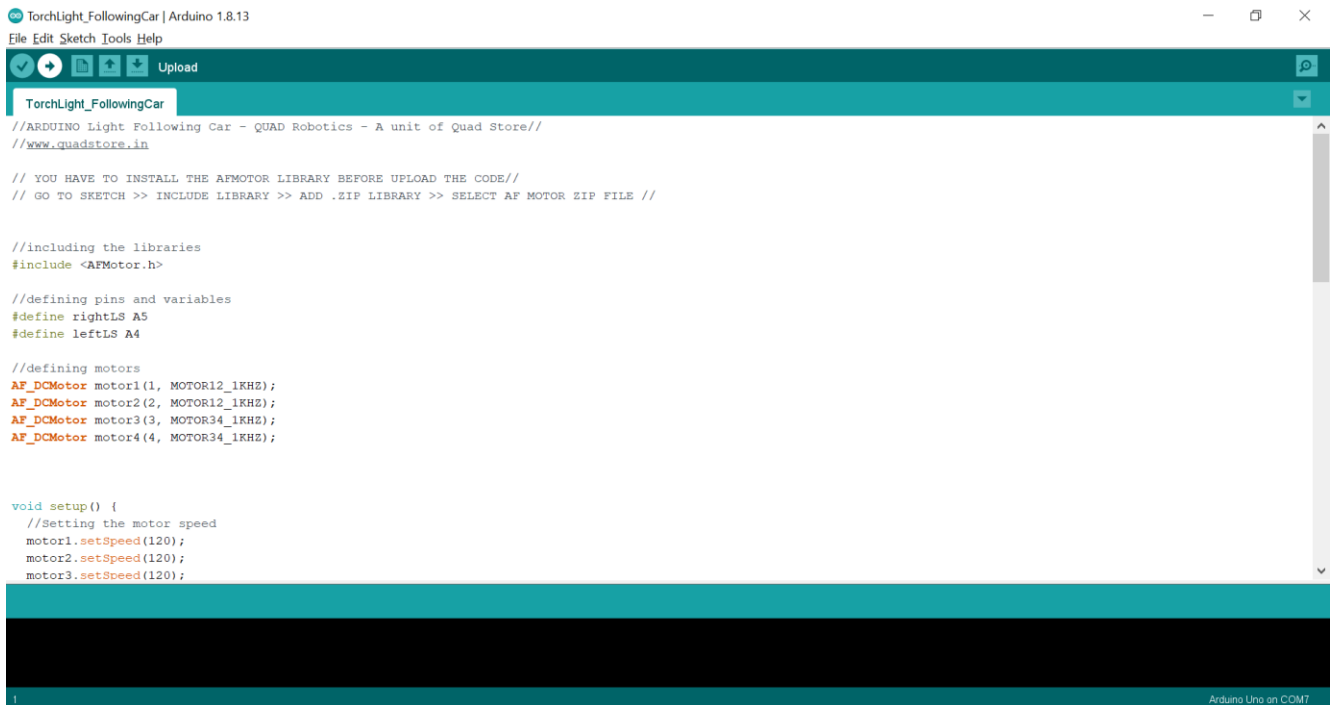
Before uploading the code to the Uno R4 board you need to install the “**AFMotor.zip**” library.

**NOTE:** If you had already installed this AFMotor.zip library during the previous projects then you can ignore this step and directly compile and upload the code to UNO R4 board. Else follow the below steps to install the library.





Compile and Upload the code to Uno R4 board.



Step-8: Insert the DC power jack from battery holder into Uno R4 dc jack and Power ON the Uno R4 board.

Result: In a dark room, show the light in front of the cars photoresistor sensors and see the robot car following the light.

# Project 25: Bluetooth Controlled Car

Introduction: In this project we will demonstrate how to assemble a Bluetooth controlled car using Android app. (Works only with Android mobile phones). Does not support iOS.

Parts Required:

- 2WD Car Chassis Assembled
- Uno R4
- L293D motor driver shield
- HC05 Bluetooth module
- Arduino Bluetooth RC Car app – Download from PlayStore.
- **IMPORTANT NOTE:** Kindly upload the code for Bluetooth car to UNO R4 board before installation. If you want to upload the code after installation, then you need to remove the power supply of the Bluetooth module before uploading. If you try to upload the code with Bluetooth module power supply, it will throw error message.

Step-1: Upload the Code to Uno R4 board

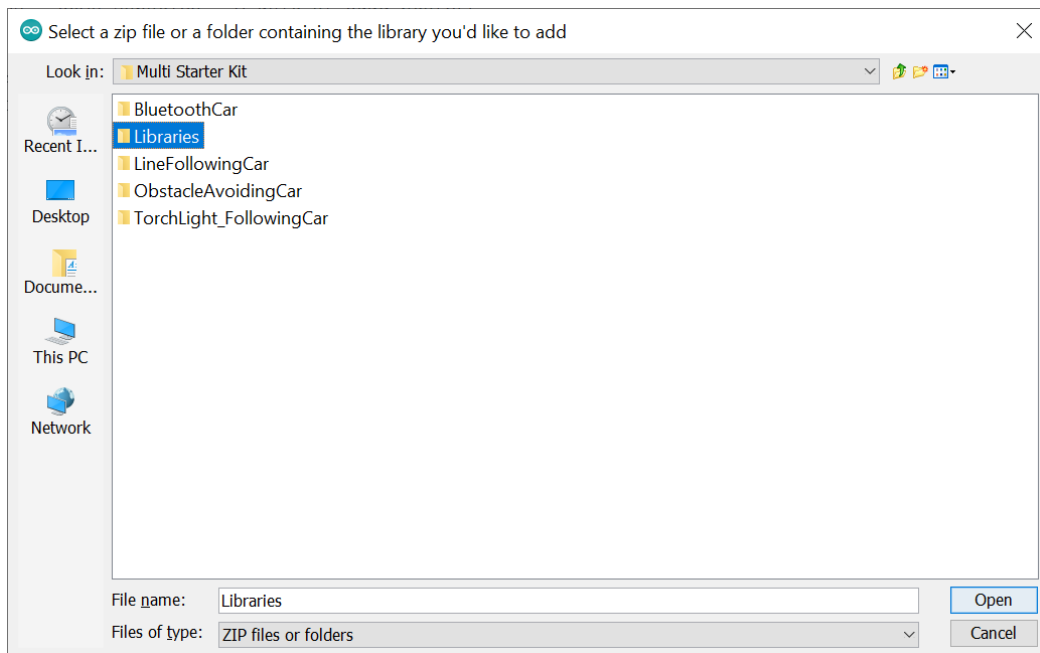
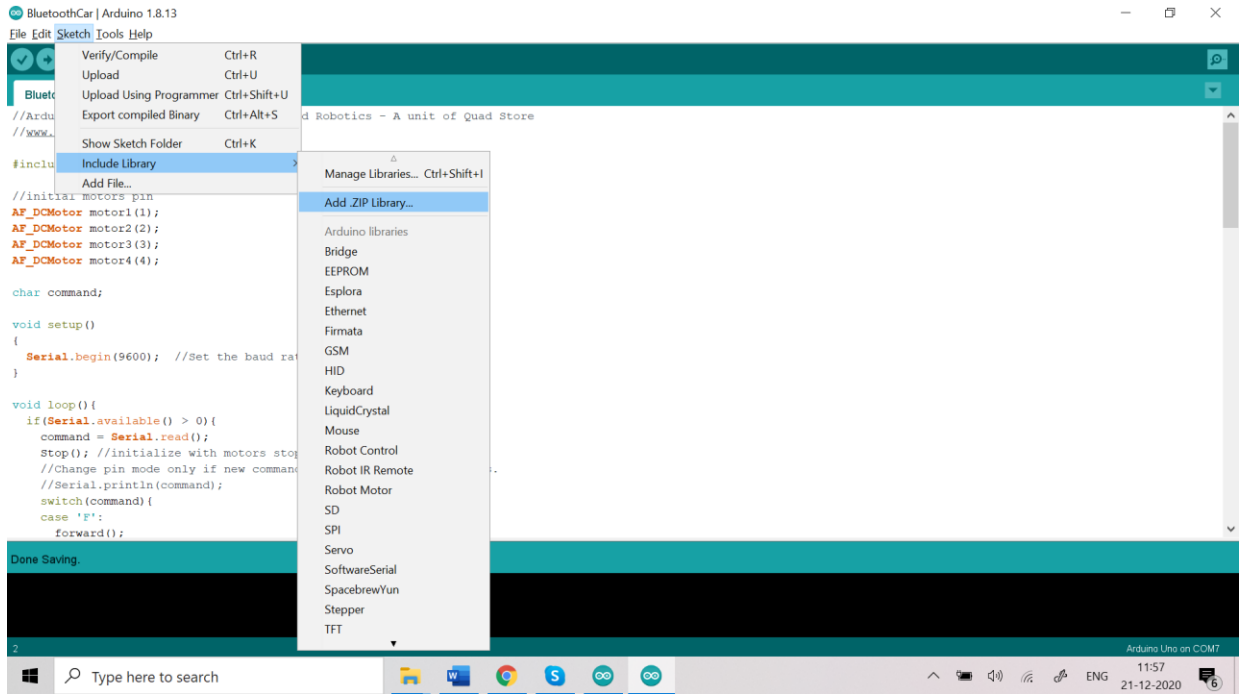
Use the provided USB cable and connect the Uno R4 board to your laptop/desktop computer.

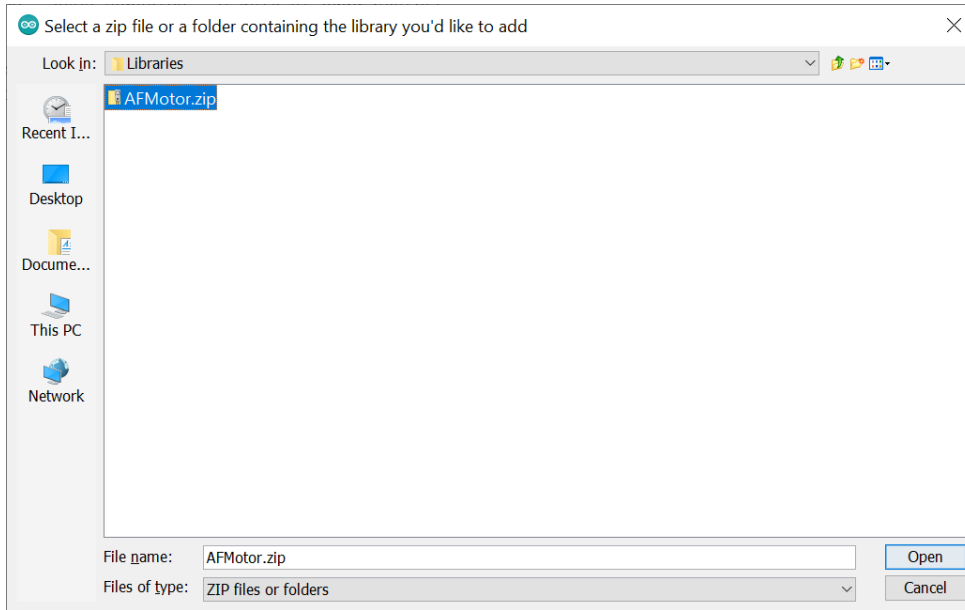
## Installing Libraries:

Open the **BluetoothCar.ino** code using the Arduino IDE.

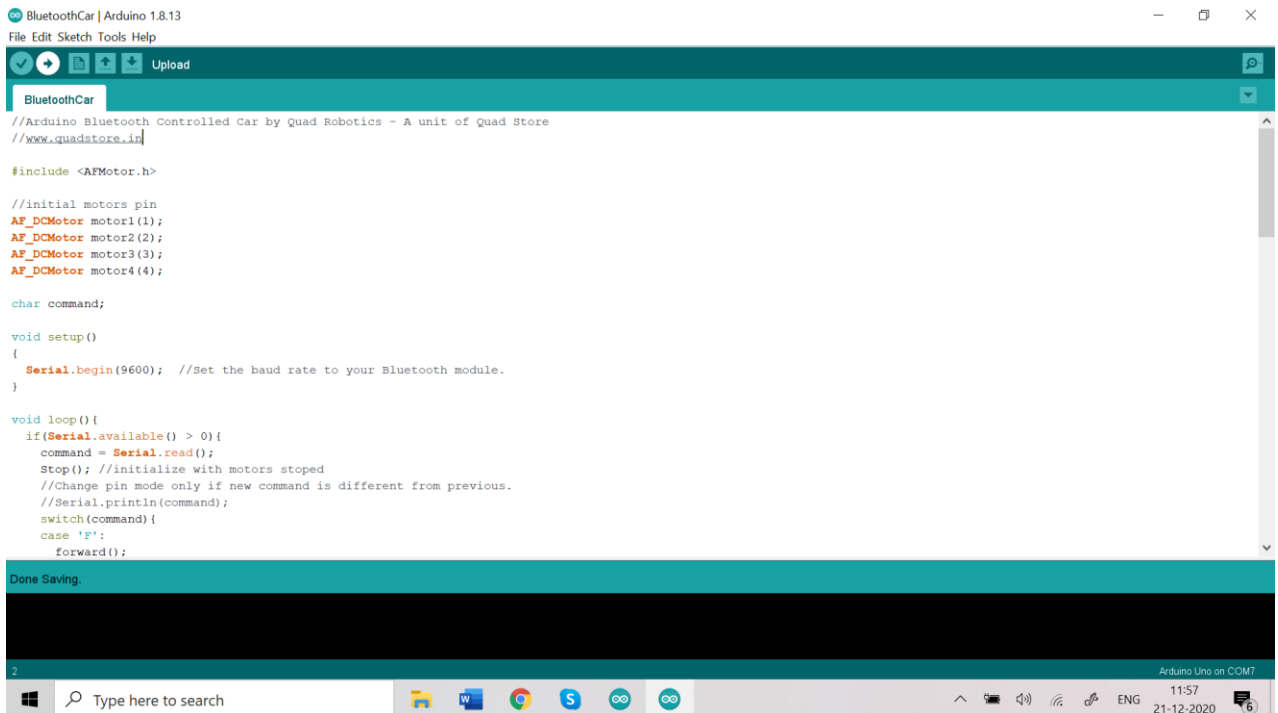
Before uploading the code to the Uno R4 board you need to install the “**AFMotor.zip**” library.

**NOTE:** If you had already installed this AFMotor.zip library during the previous projects then you can ignore this step and directly compile and upload the code to UNO R4 board. Else follow the below steps to install the library.

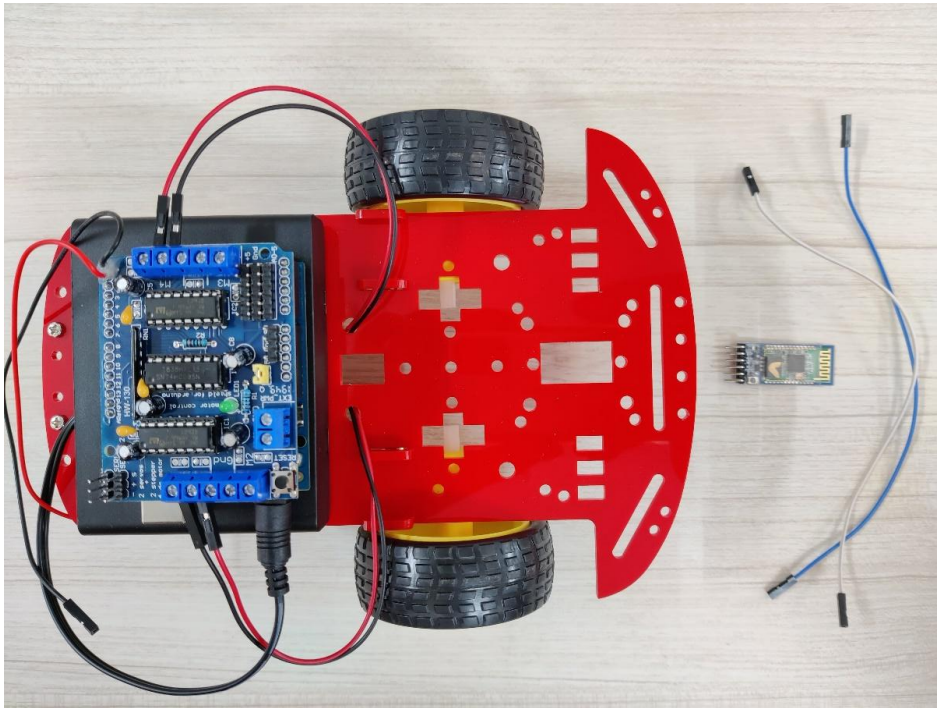




Compile and Upload the code to Uno R4 board.



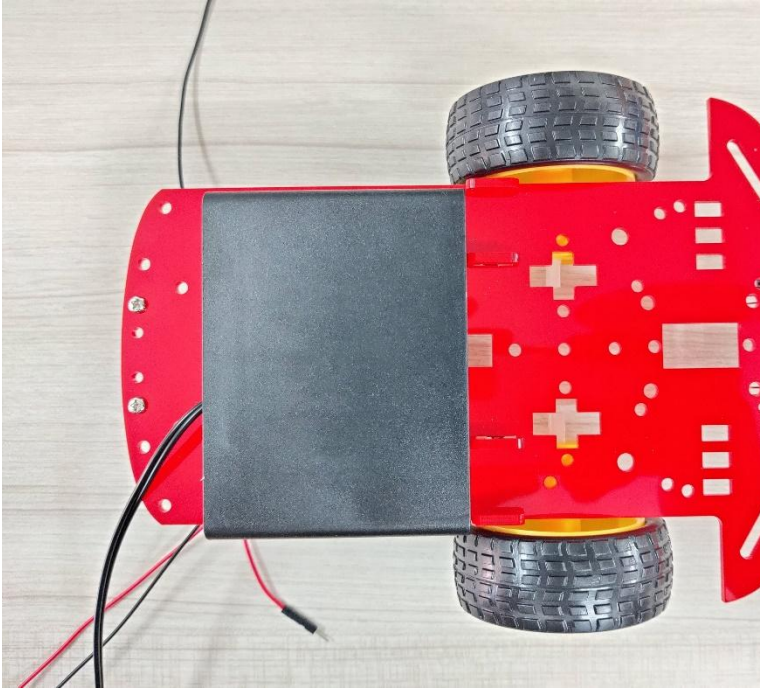
Step-2: Take an assembled 2WD Car chassis. Just in case you are not aware how to assemble the 2WD car chassis please refer to the “2WD Card Chassis Assembly Instruction” section for details.



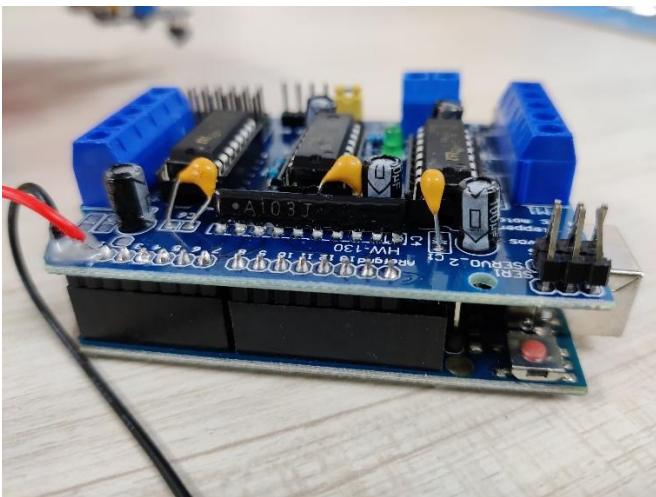
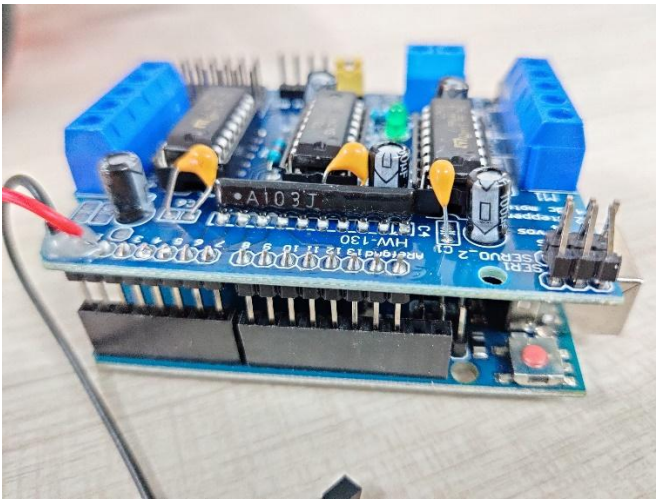
Step-3: Insert 6 x AA batteries in battery holder and place it in the back side of the car chassis using double side stickers.



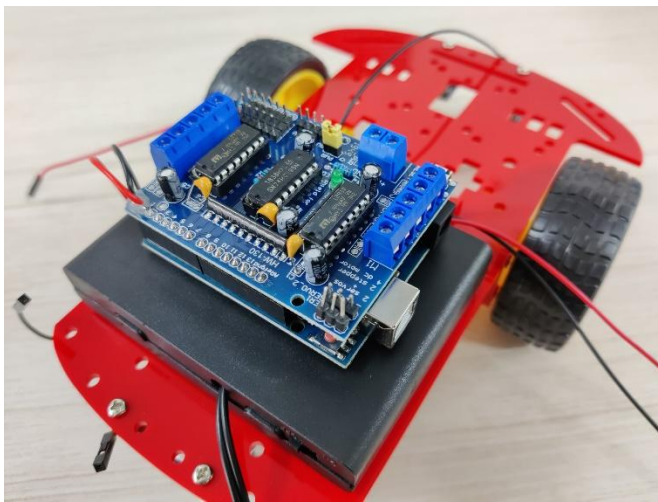
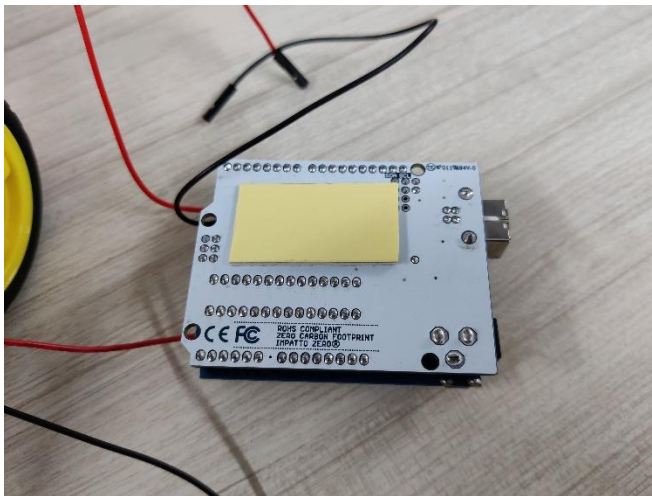




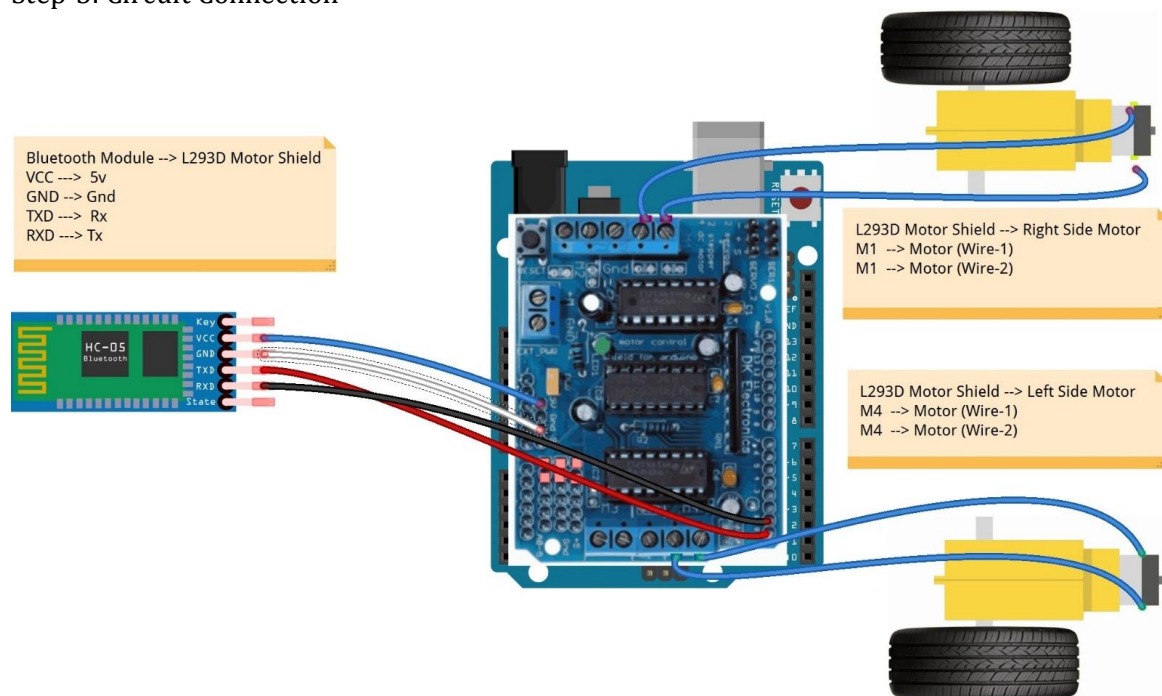
Step-4: Insert the L293D motor driver shield on top of the Uno R4 board and use a double side tape and fix it on top of the battery holder.







## Step-5: Circuit Connection

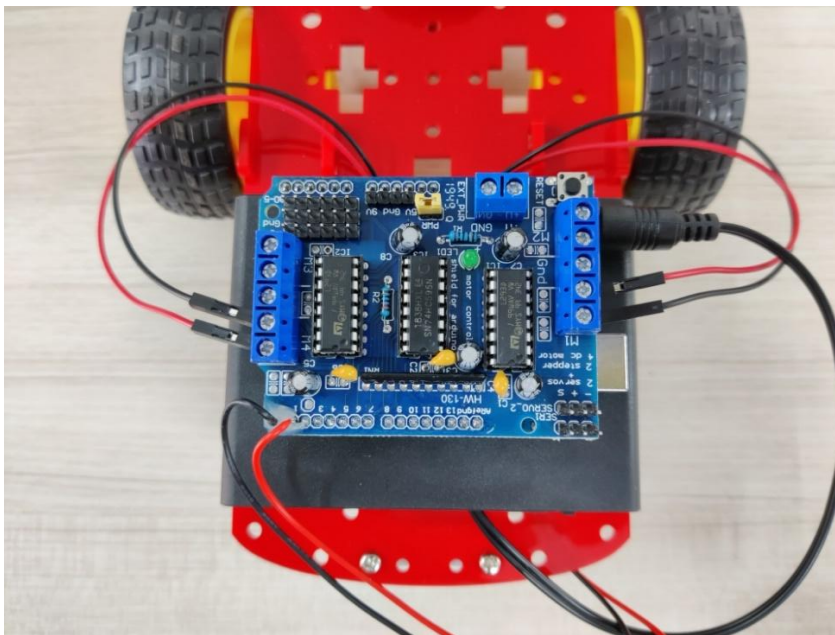


## Motors to L293D driver shield

Left Side Motor	L293 Shield mounted on Uno R4
Wire-1 (Red)	M4
Wire-2 (Black)	M4

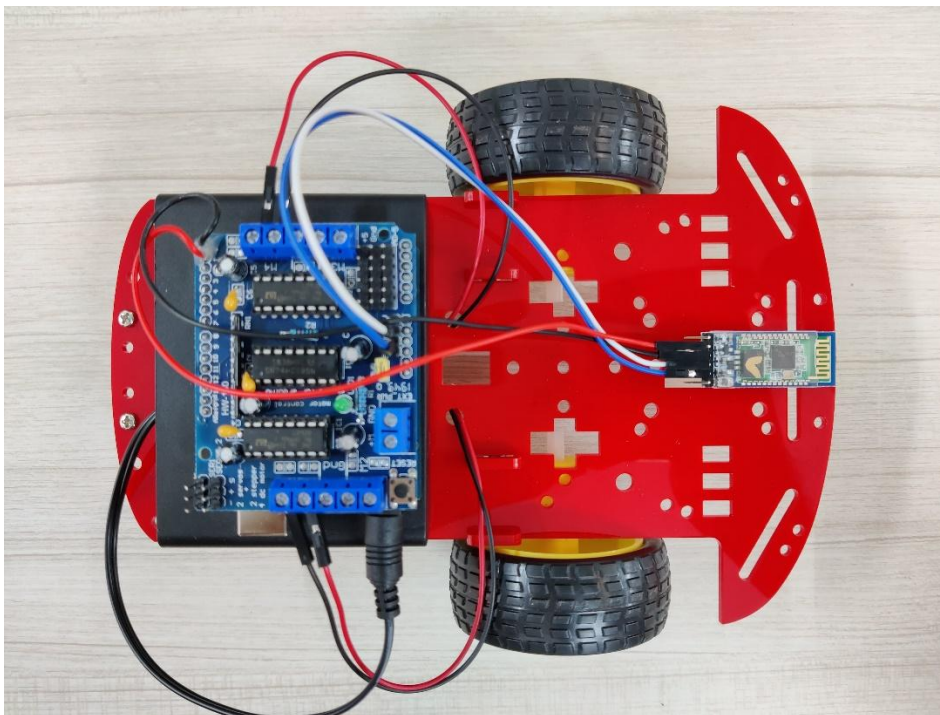
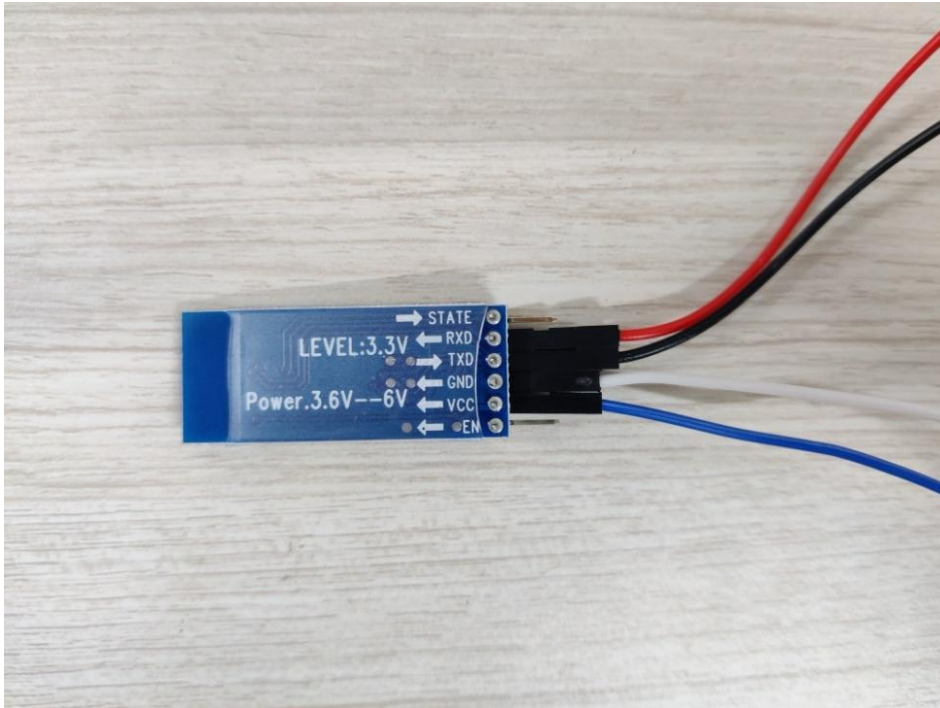
Right Side Motor	L293 Shield mounted on Uno R4
Wire-1 (Red)	M1
Wire-2 (Black)	M1

**Note:** kindly note that there is no polarity in gear motors. Hence the red and black wires provided with the motor is just for reference purpose. They can be interchanged if the motor is required to rotate in different directions.

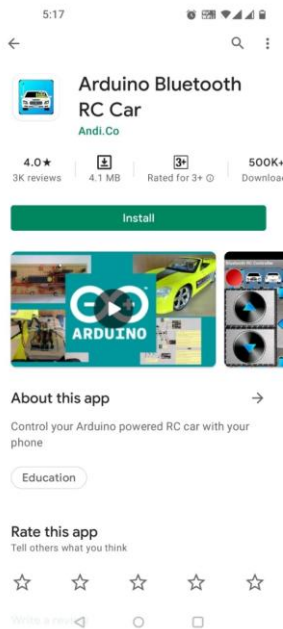


## Bluetooth Module to L293D motor driver shield:

Bluetooth Module	L293 Shield mounted on Uno R4
RX	Tx (Pin 1)
TX	Rx (Pin 0)
GND	Gnd
VCC	5v



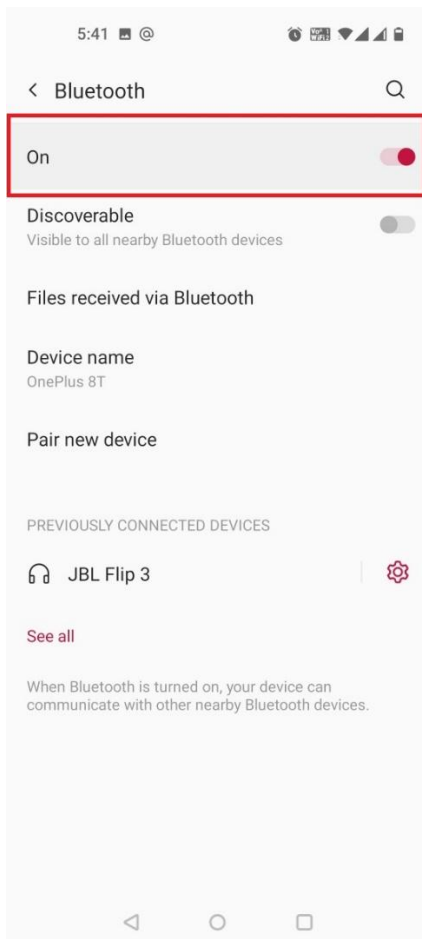
Installing the Bluetooth RC controller app from PlayStore in your Android mobile phone:  
Go to playstore and search for “Arduino Bluetooth RC Car” and install it.



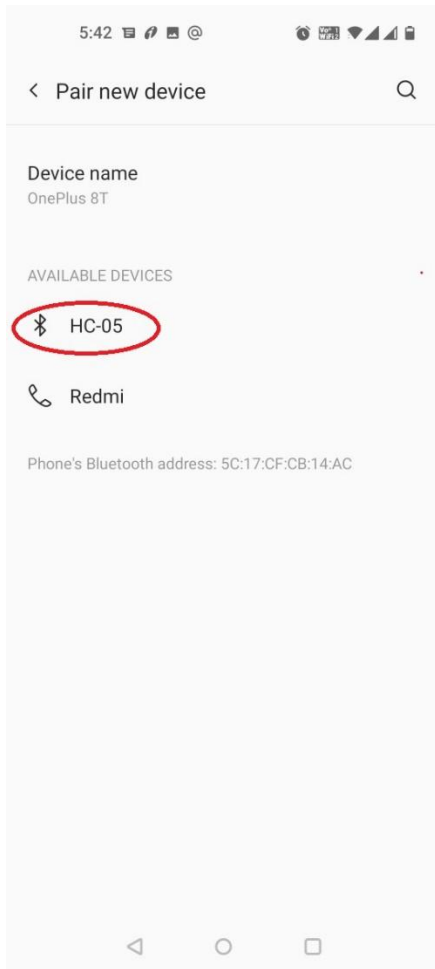
Power on the Car and you would see the RED led in the Bluetooth module blinking. This means the Bluetooth module is waiting for pairing with the app.

In your mobile phone

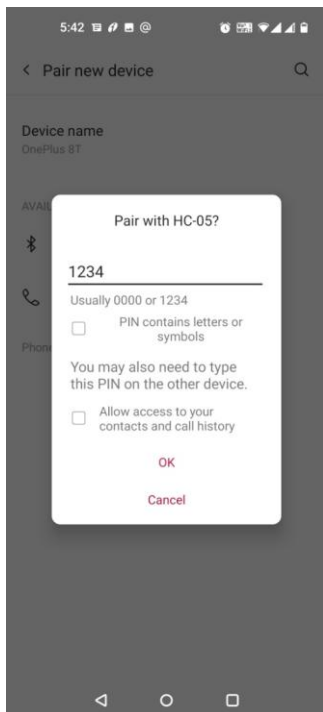
Go to Settings → Bluetooth → Turn on Bluetooth Option



Click “Pair new device” and from the Available devices select “HC-05”



When prompted for password please enter 1234 and click Ok.

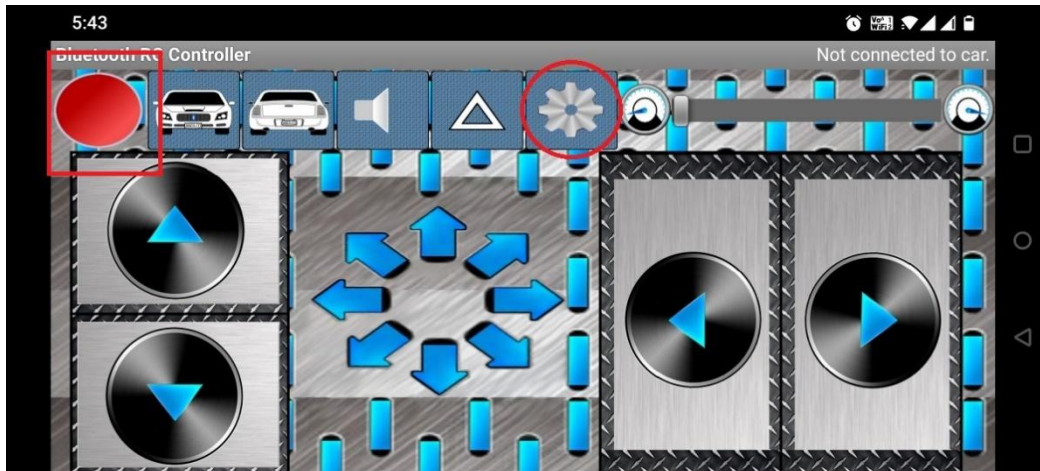




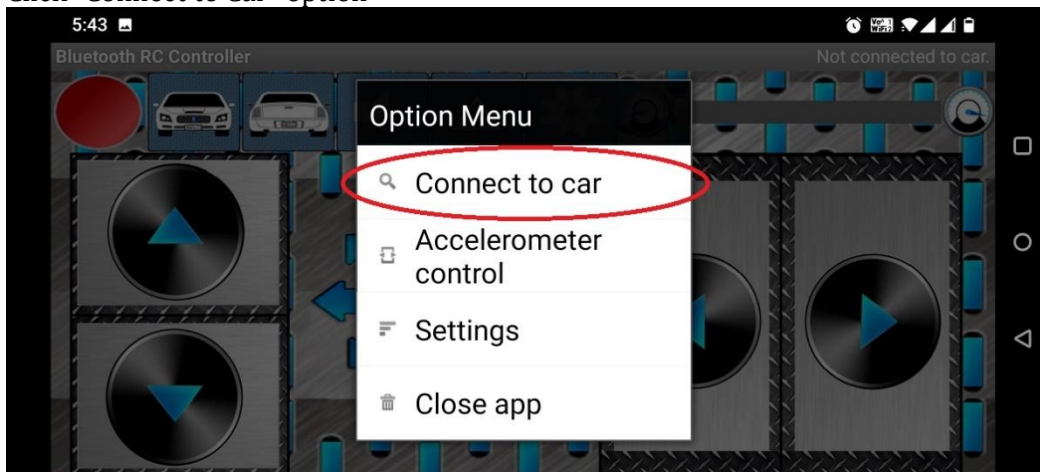
Kindly note still your Bluetooth module is NOT yet paired with the app.

Now open the “Arduino Bluetooth RC Car” app. You will see a RED circle indicator blinking on the left corner of the app which indicates that bluetooth module is not yet paired with the app.

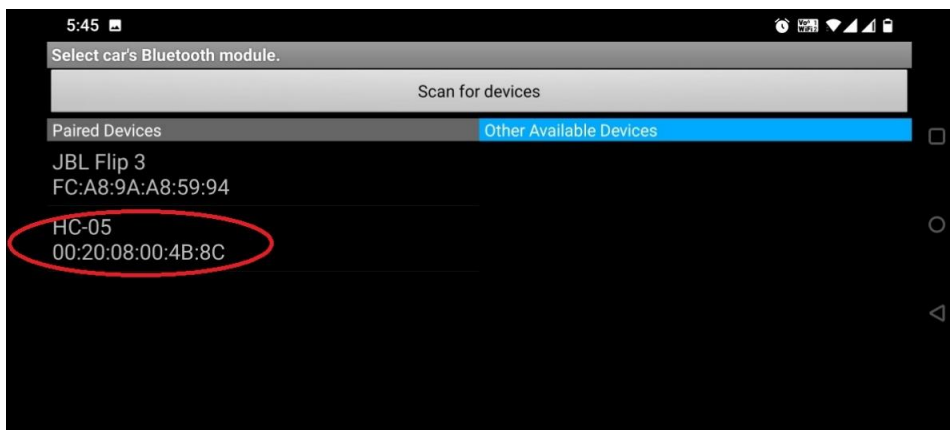
Now Click on the setting wheel icon.



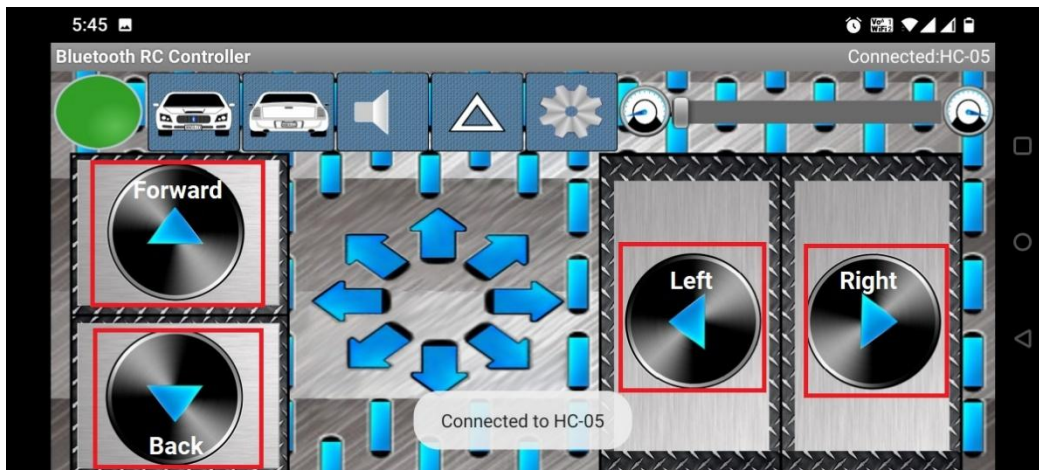
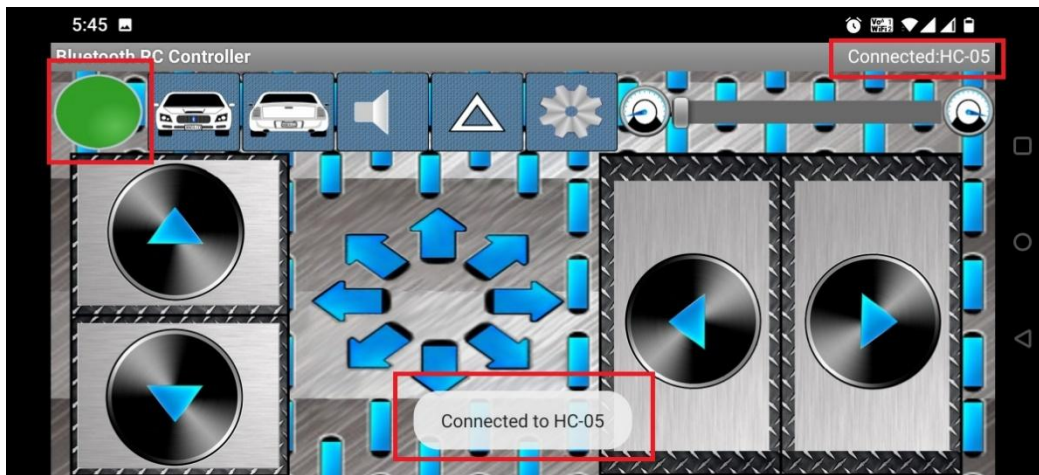
Click “Connect to Car” option



In the list of Bluetooth devices listed select “HC-05” option.



Now you should see the red flashing led turning to SOLD Green and also a text which reads Connected to HC-05 being displayed.



**Result:** Now that the app is connected to your Bluetooth car use the Forward, Backward, Left and Right direction button to make the car move in desired direction.



# Project 26: Wifi ESP8266-01 Control with Blynk App

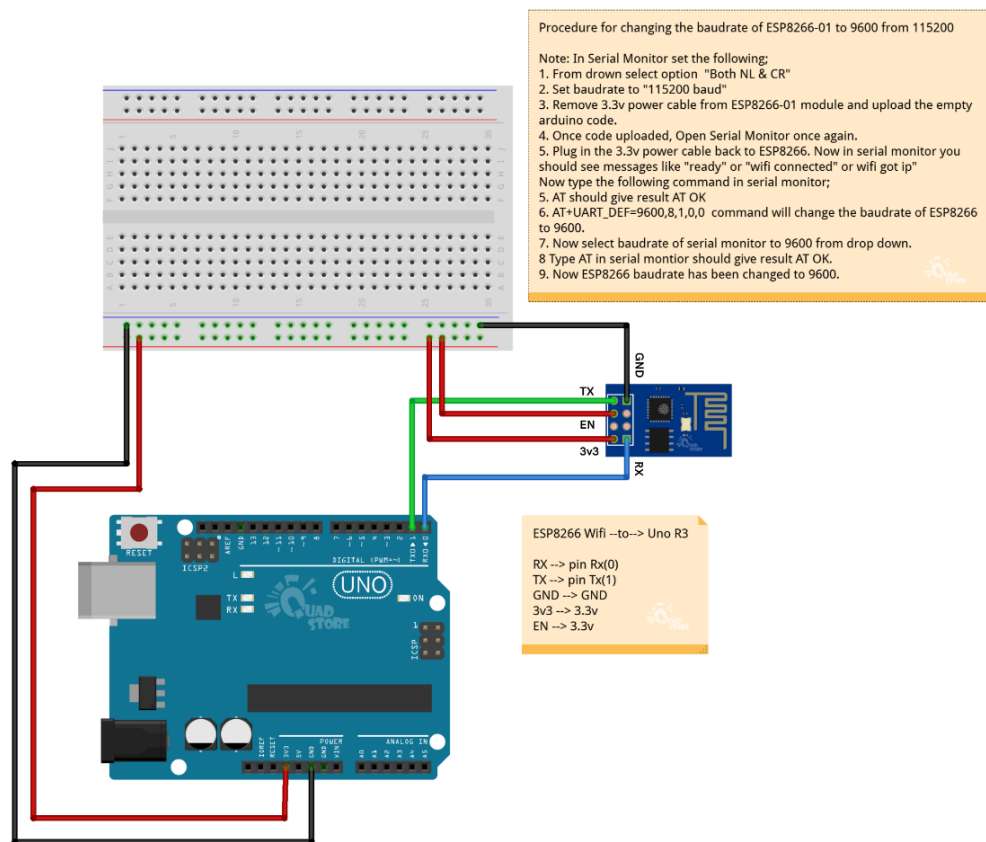
**Introduction:** Control LED using wifi and blynk app from your mobile.

**Parts Required:**

- Uno R4 board, USB Cable
- Breadboard
- Few jumper wires
- ESP8266-01 module
- LED
- Resistor – 220ohm
- Blynk app need to be installed

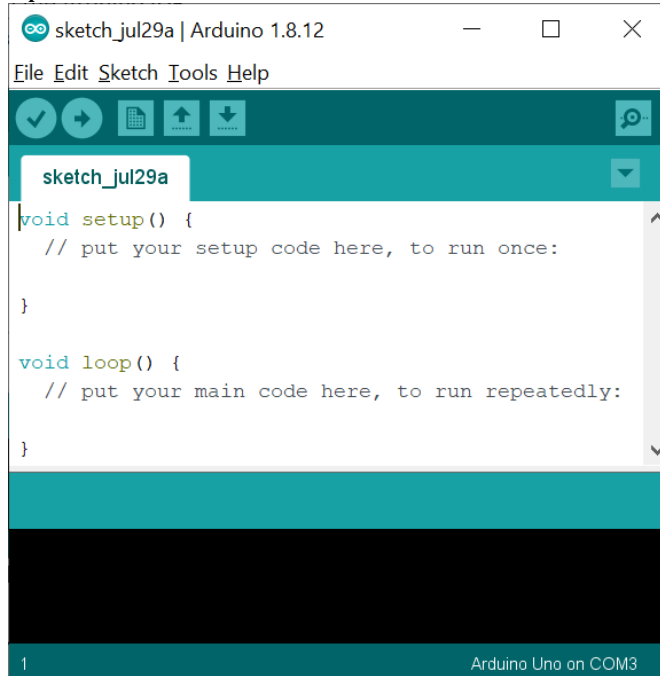
**Note:** You need to change the circuit connection twice for setting up the ESP8266-01 wifi module.

**First circuit connection for changing the baudrate of ESP8266-01 module:**  
Provide the connection as per the below circuit.

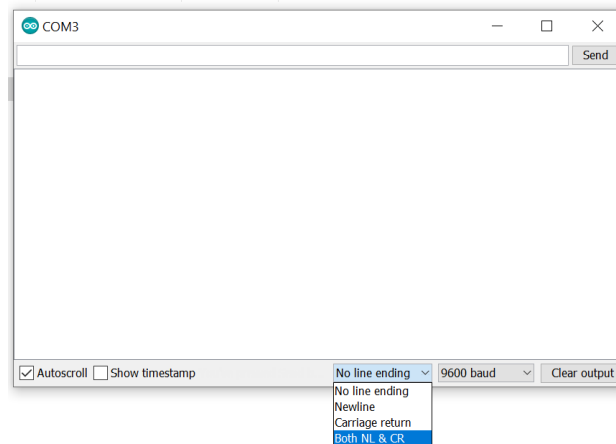


**Action-1:** Procedure for changing the baudrate of ESP8266-01 to 9600 from 115200

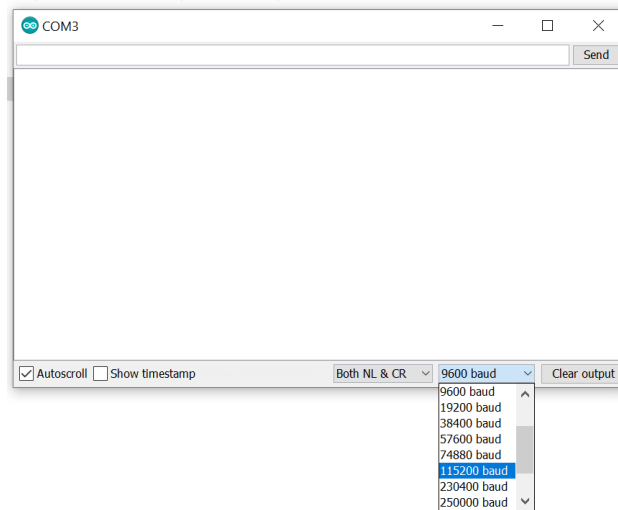
1. Open New Sketch from the Arduino IDE.



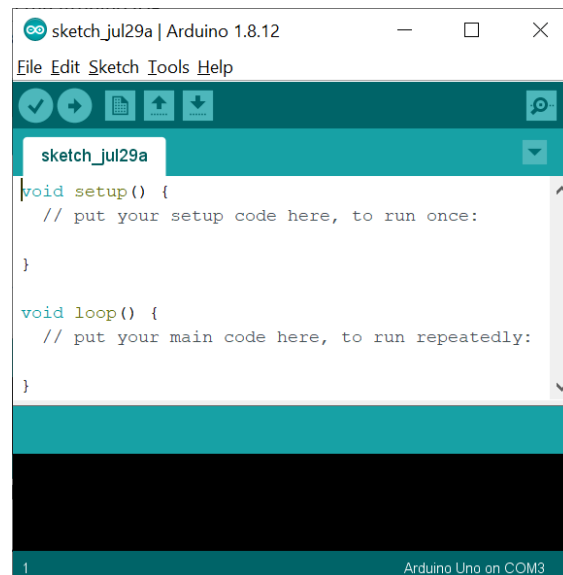
2. Open Serial Monitor and change the following options
3. From dropdown select option "Both NL & CR"



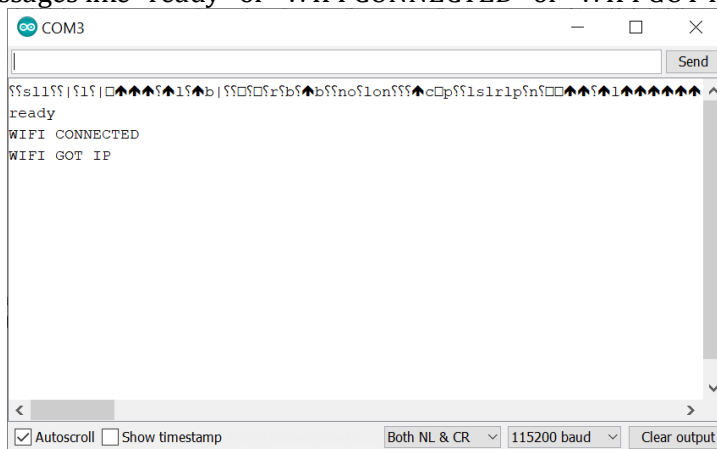
4. Set baudrate to "115200 baud"



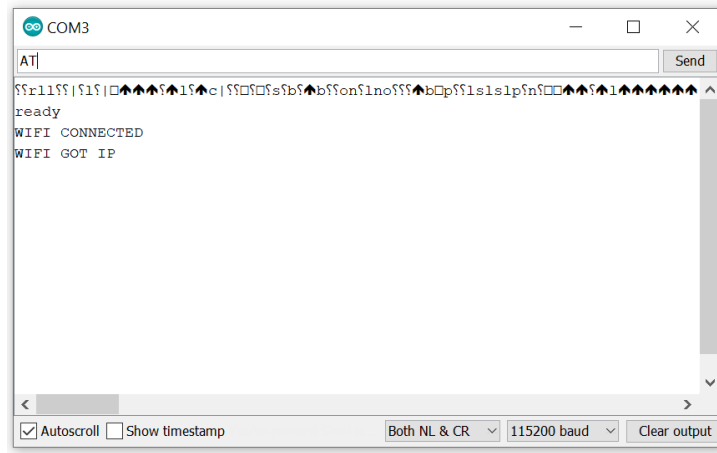
5. Remove **3.3v power cable** from ESP8266-01 module.
6. Upload the New empty Sketch/Code to the Uno R4 board.



7. Once code uploaded, Open Serial Monitor once again.
8. Plug in the **3.3v power cable** back to ESP8266. Now in serial monitor you should see messages like "ready" or "WIFI CONNECTED" or "WIFI GOT IP"



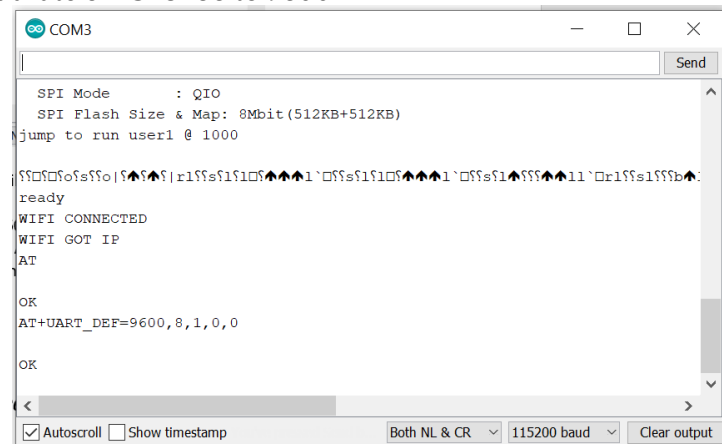
9. Now type the command **AT** in serial monitor and click Send button.



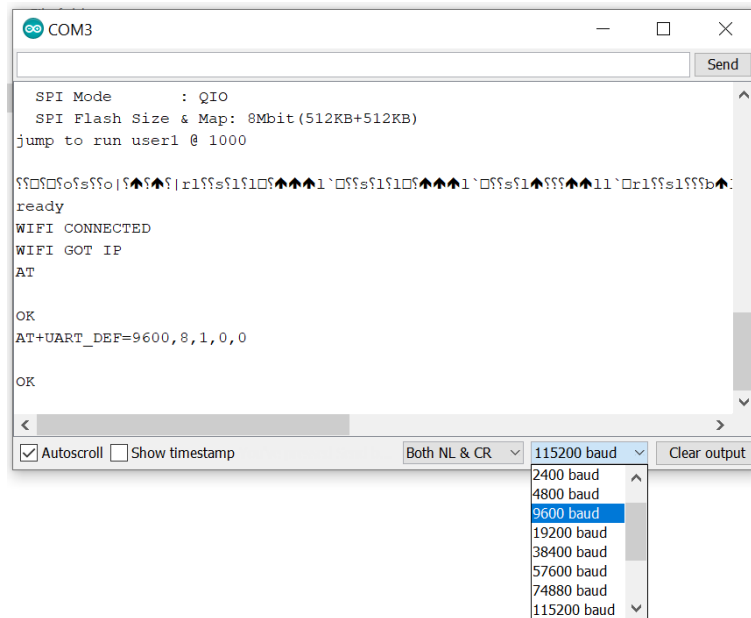
10. AT should give result AT OK



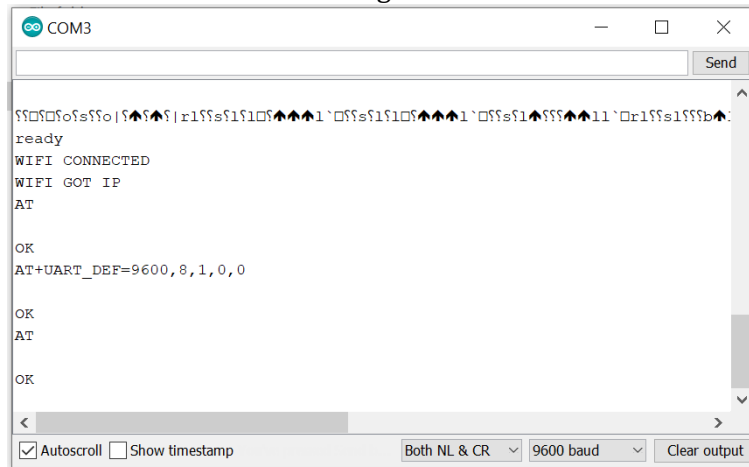
11. Now type command **AT+UART\_DEF=9600,8,1,0,0** command will change the baudrate of ESP8266 to 9600.



12. Now select baudrate of serial monitor to 9600 from drop down.



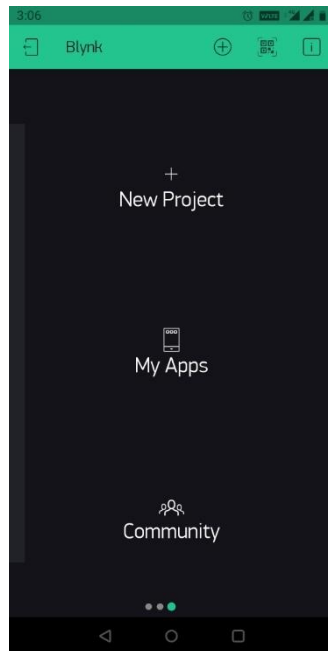
13. Type AT in serial monitor should give result AT OK.



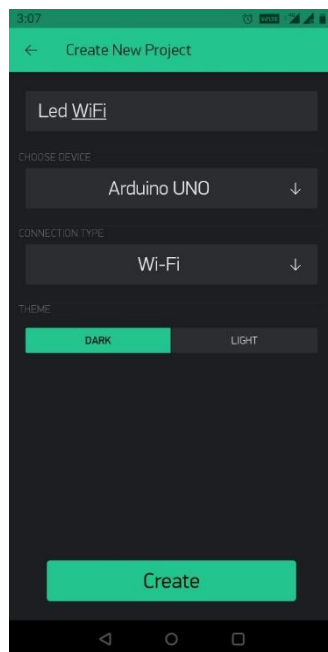
14. Success! Now the baudrate of ESP8266-01 has been changed to 9600.

### Installation and setup of Blynk App:

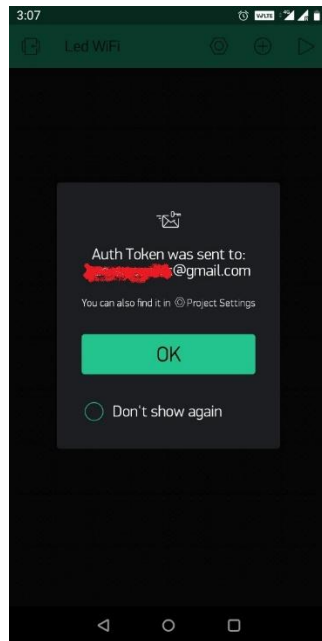
1. Download and install Blynk app from Google Playstore or Appstore
2. Open Blynk App and create "New Project"



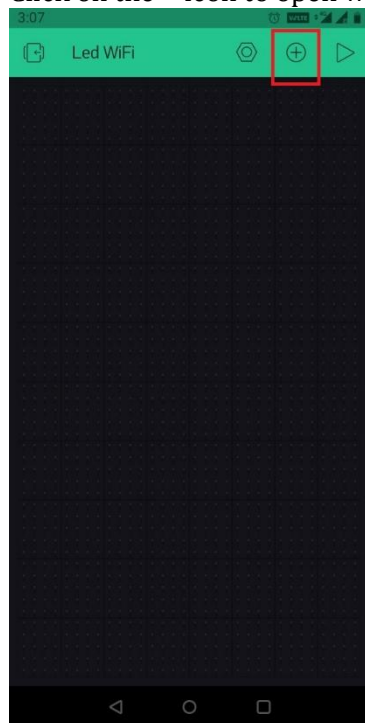
3. Enter the "Project Name" as Led Wifi.
4. Select the "Board Type" as Arduino Uno and click Create



5. Auth token will be generated and sent to your email address.

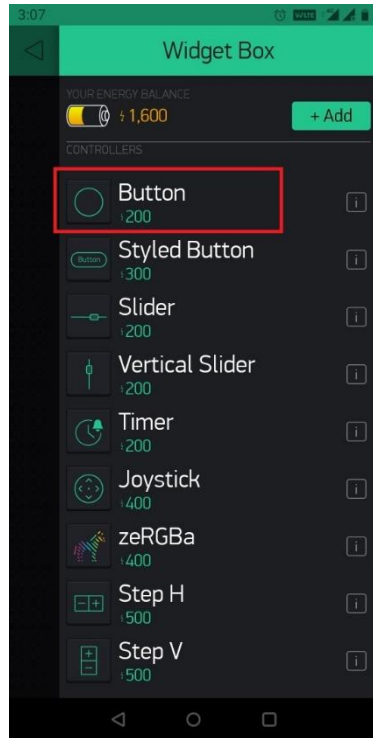


6. Click on the + icon to open widget box

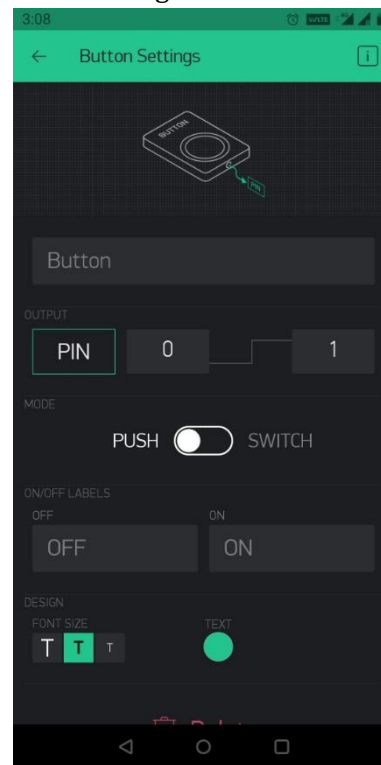
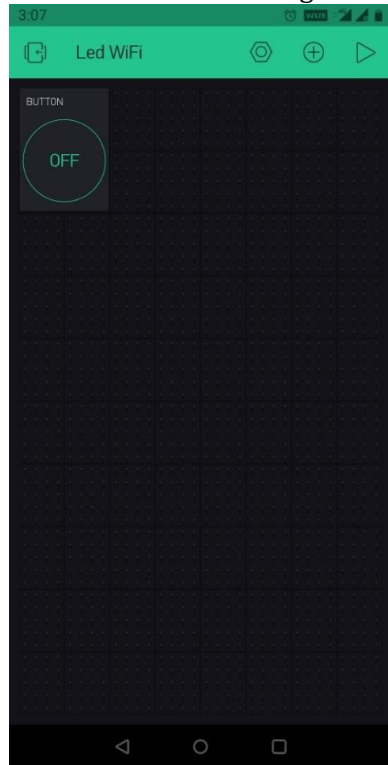


7. Add a new widget "Button"

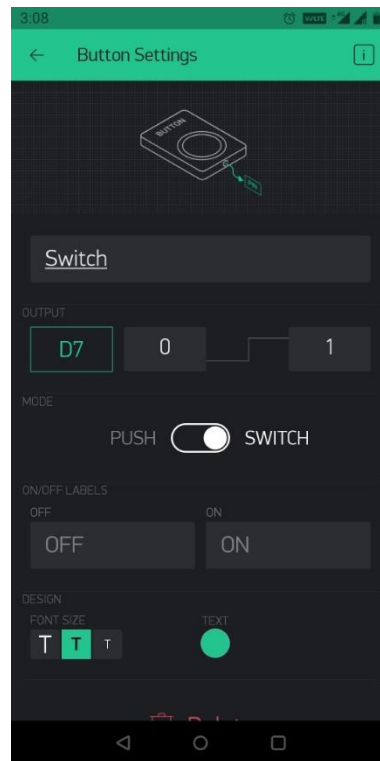
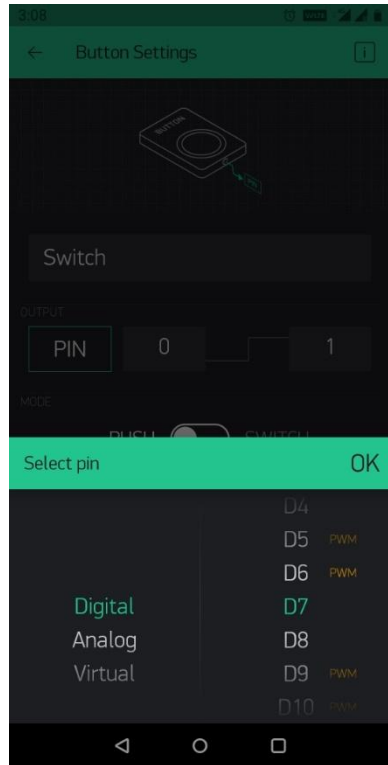




8. Click on the button widget will take to its setting

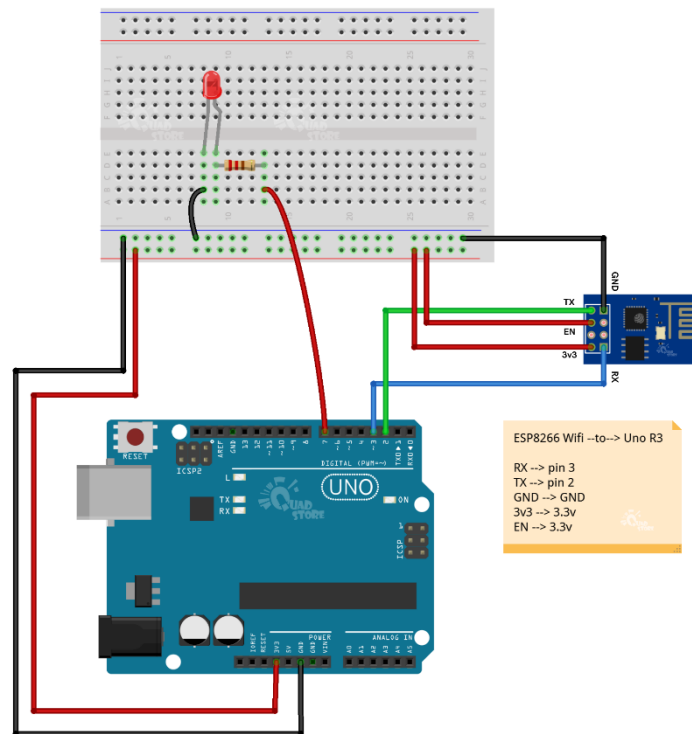


9. Change the PIN to "Digital - pin 7", rename the widget to "Switch" and change the PUSH option to SWITCH and go back.



10. Now all the configuration is completed at Blynk app.

***Second circuit connection for using the ESP8266-01 wifi module with your phones Blynk app: Provide the connection as per the below circuit.***



## Update your CODE:

1. **Library:** Install Blynk.zip and BlynkESP8266\_Lib.zip libraries. Otherwise your code will give error message.
2. Open the code named **"wifi\_blynk.ino"**
3. Make the following changes in your CODE to reflect the correct **"Auth Token"**, **"Network Name"** and **"Network Password"**



```
WIFI_Blynk $
Value Display widget attached to Virtual Pin V5
*****

/* Comment this out to disable prints and save space */
#define BLYNK_PRINT Serial

#include <ESP8266_Lib.h>
#include <BlynkSimpleShieldEsp8266.h>

// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "Your Auth Token"; // Replace here with the Auth token you have received in your email.

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "Your Network Name"; // Enter your Wifi Network Name
char pass[] = "Your Network Password"; //Enter your Wifi Network Password

// Hardware Serial on Mega, Leonardo, Micro...
//define EspSerial Serial1

// or Software Serial on Uno, Nano...
#include <SoftwareSerial.h>
SoftwareSerial EspSerial(2, 3); // TX, RX

// Your ESP8266 baud rate:
Done Saving
49 Arduino Uno on COM3
```

**Note:** You would have received the Auth Token in your email. Kindly copy and paste it in the code.

4. Next enter your Wifi network **"User Name"** and **"Password"**.



```
WIFI_Blynk
Value Display widget attached to Virtual Pin V5
*****

/* Comment this out to disable prints and save space */
#define BLYNK_PRINT Serial

#include <ESP8266_Lib.h>
#include <BlynkSimpleShieldEsp8266.h>

// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "zzv2cKTYV5S8f0we8y-eLen5bucbyGZ"; // Replace here with the Auth token you have received in your email.

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "Your Network Name"; // Enter your Wifi Network Name
char pass[] = "Your Network Password"; //Enter your Wifi Network Password

// Hardware Serial on Mega, Leonardo, Micro...
//define EspSerial Serial1

// or Software Serial on Uno, Nano...
#include <SoftwareSerial.h>
SoftwareSerial EspSerial(2, 3); // TX, RX

// Your ESP8266 baud rate:
Done Saving
53 Arduino Uno on COM3
```

5. Once you have entered the Auth token, User Name and Passowrd you code should look something like this;



```
Wifi_Blynk | Arduino 1.8.12
File Edit Sketch Tools Help

Wifi_Blynk $
Value Display widget attached to Virtual Pin V5
*****

/* Comment this out to disable prints and save space */
#define BLYNK_PRINT Serial

#include <ESP8266_Lib.h>
#include <BlynkSimpleShieldEsp8266.h>

// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "zjfdjklldkfdsljkfs-euir889efjjdlkG2"; // Replace here with the Auth token you have received in your email.

// Your Wifi credentials.
// Set password to "" for open networks.
char ssid[] = "quadstoreuse"; // Enter your Wifi Network Name
char pass[] = "Northypass"; //Enter your Wifi Network Password

// Hardware Serial on Mega, Leonardo, Micro...
//define EspSerial Serial1

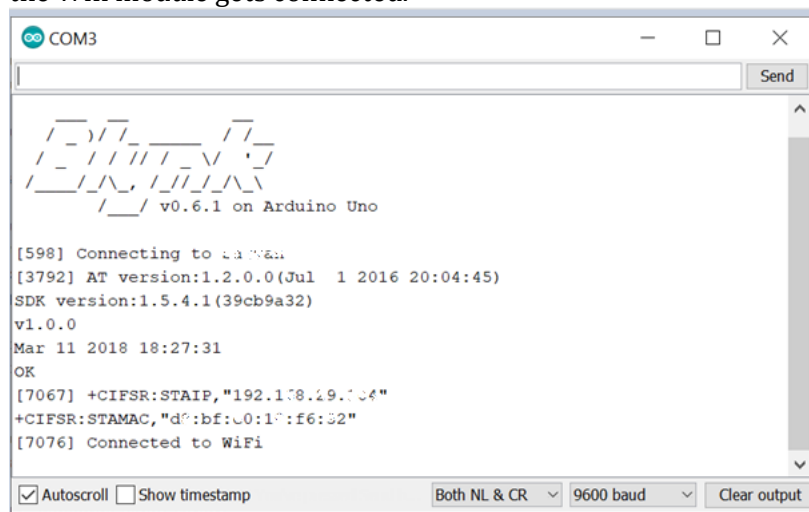
// or Software Serial on Uno, Nano...
#include <SoftwareSerial.h>
SoftwareSerial EspSerial(2, 3); // TX, RX

// Your ESP8266 baud rate:

Done Saving.

53 Arduino Uno on COM3
```

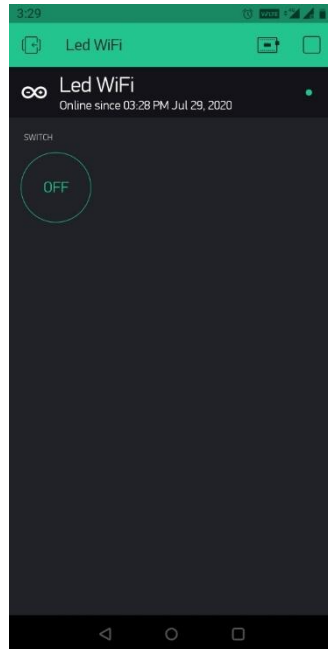
6. Now perform the below steps to get connected with the wifi module:
7. Remove 3.3v power cable from ESP8266-01 module and then upload the updated code.
8. Plug in the 3.3v power cable back to ESP8266-01 module.
9. Open “Serial Monitor” and check if the ESP8266-01 module is getting connected to your Wifi network successfully.
10. Ensure the baudrate is at 9600 then you should see like the screen below where the Wifi module gets connected.



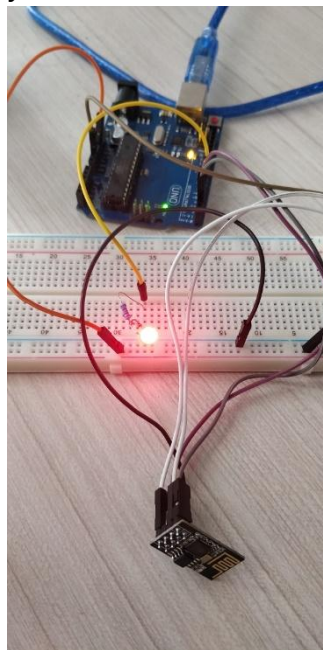
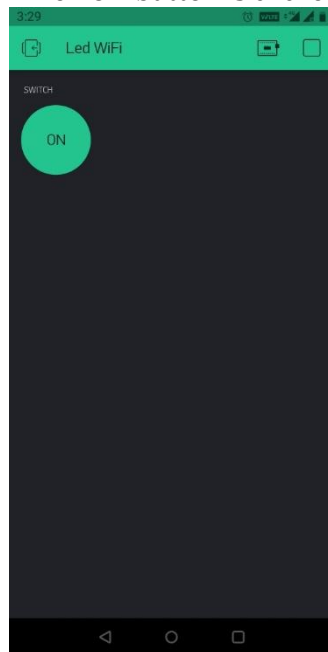
```
COM3
[598] Connecting to STAIP
[3792] AT version:1.2.0.0(Jul 1 2016 20:04:45)
SDK version:1.5.4.1(39cb9a32)
v1.0.0
Mar 11 2018 18:27:31
OK
[7067] +CIFSR:STAIP,"192.168.49.104"
+CIFSR:STAMAC,"d8:bf:00:18:f6:32"
[7076] Connected to WiFi

Autoscroll Show timestamp Both NL & CR 9600 baud Clear output
```

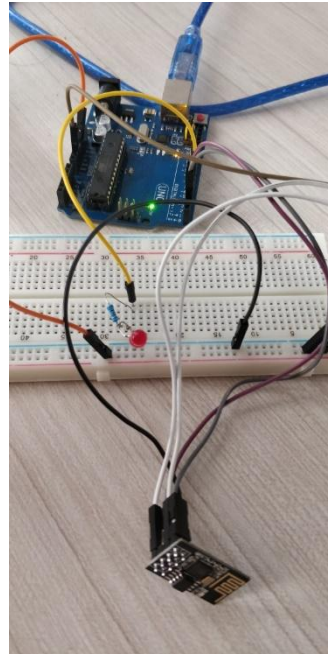
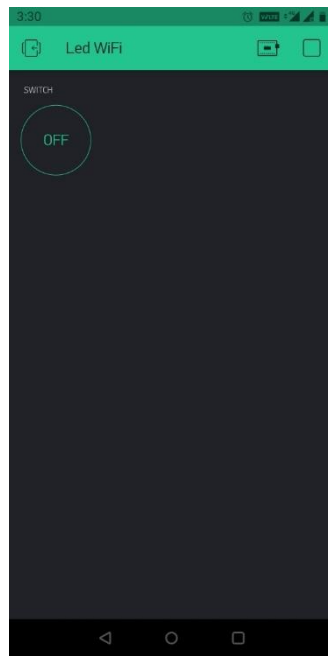
11. Open Blynk app and you should see the project getting connected to your Uno R4 board through wifi automatically.



12. **Output:** Click ON and OFF button through Blynk app to control the LED.
13. When ON button is clicked you would see the LED turn ON through Wifi.



Similarly, when OFF button is clicked the LED turns OFF through Wifi.





# Project 27: ESP32 blinking internal led


## Initial Setup: How to Use ESP32 Board with Arduino IDE

### ✅ Step 1: Install the Arduino IDE

1. Go to the official Arduino website:

 <https://www.arduino.cc/en/software>

2. Download the IDE for your operating system (Windows, Mac, or Linux).
3. Install it by following the on-screen instructions.

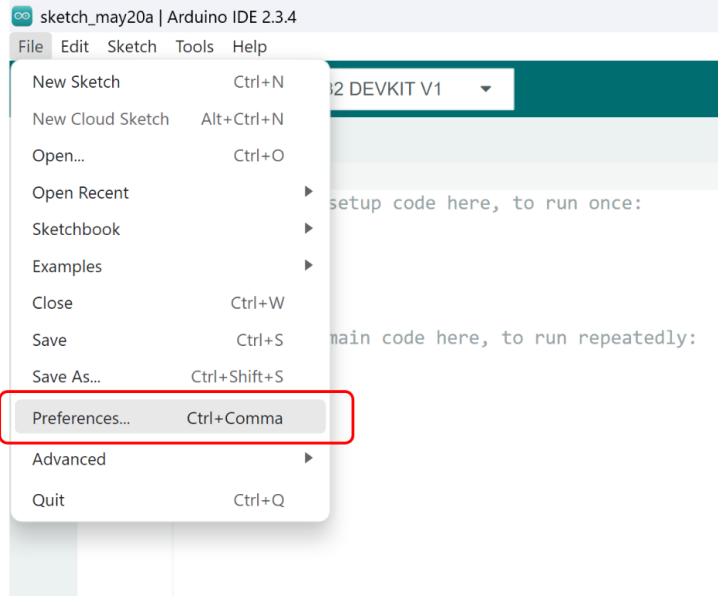
 Recommended: Use Arduino IDE version 1.8.x or higher for best ESP32 compatibility.

### ✅ Step 2: Install ESP32 Board Package

The Arduino IDE does not support ESP32 out of the box. You need to install the ESP32 board definitions.

#### ♦ 2.1 Open Arduino IDE

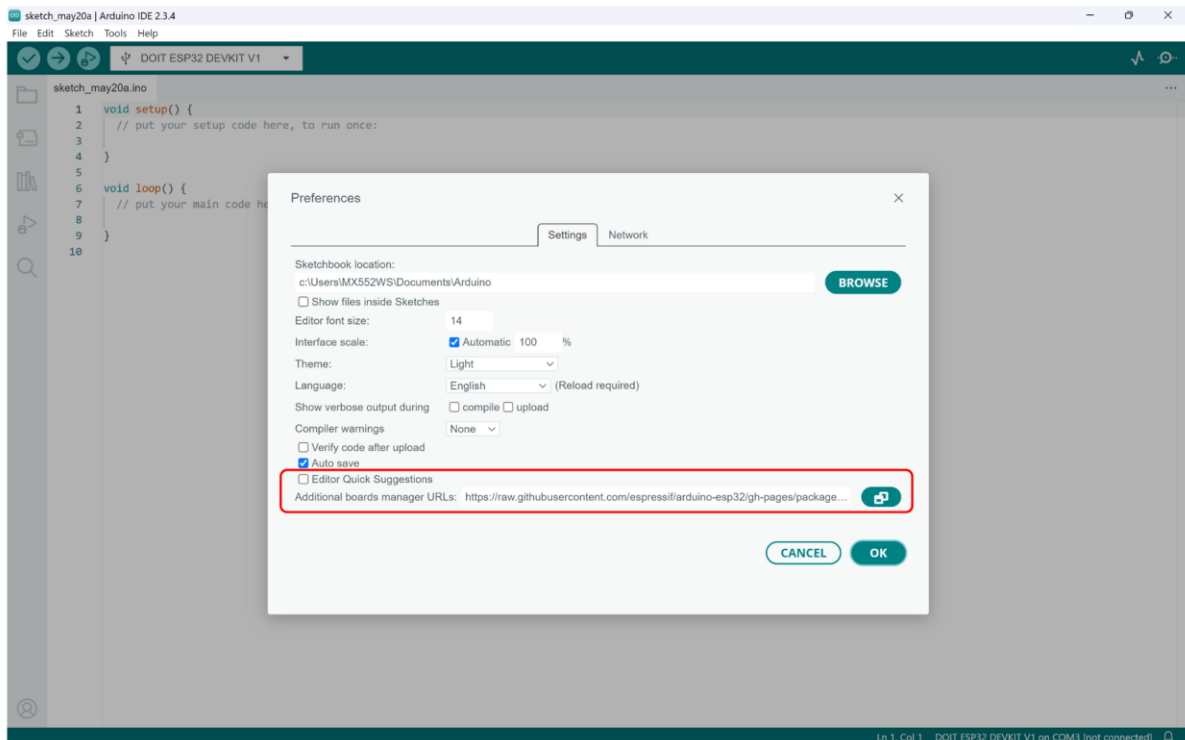
Go to **File > Preferences**



#### ♦ 2.2 Add Board Manager URL

In the "Additional Board Manager URLs" field, paste the following link:

[https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package\\_esp32\\_index.json](https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json)

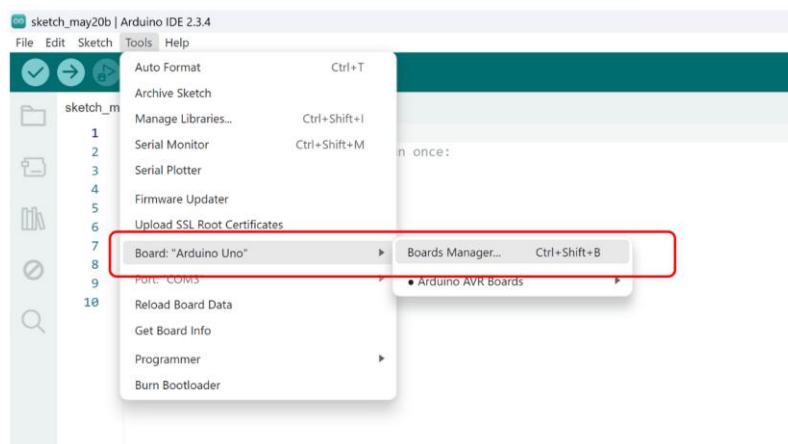


⚠ If you already have a URL there, separate this one with a comma.

Click OK.

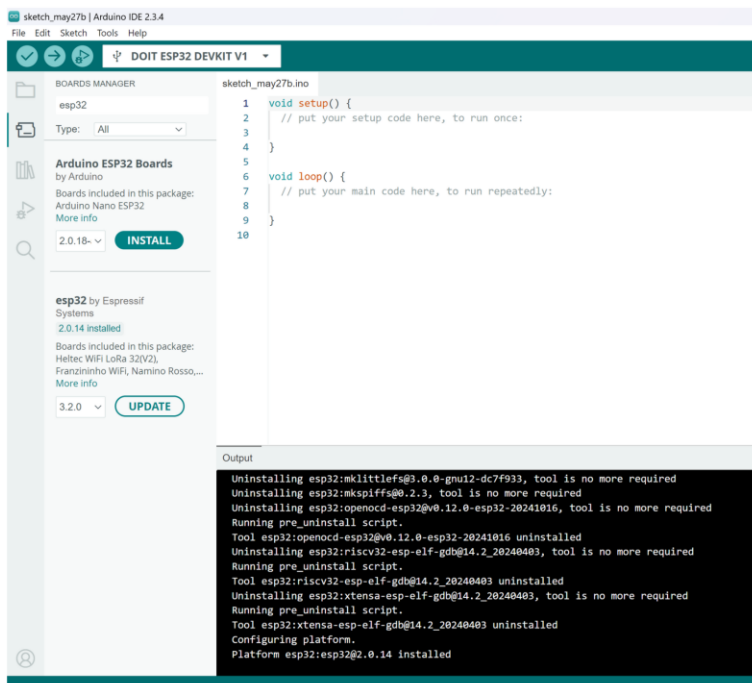
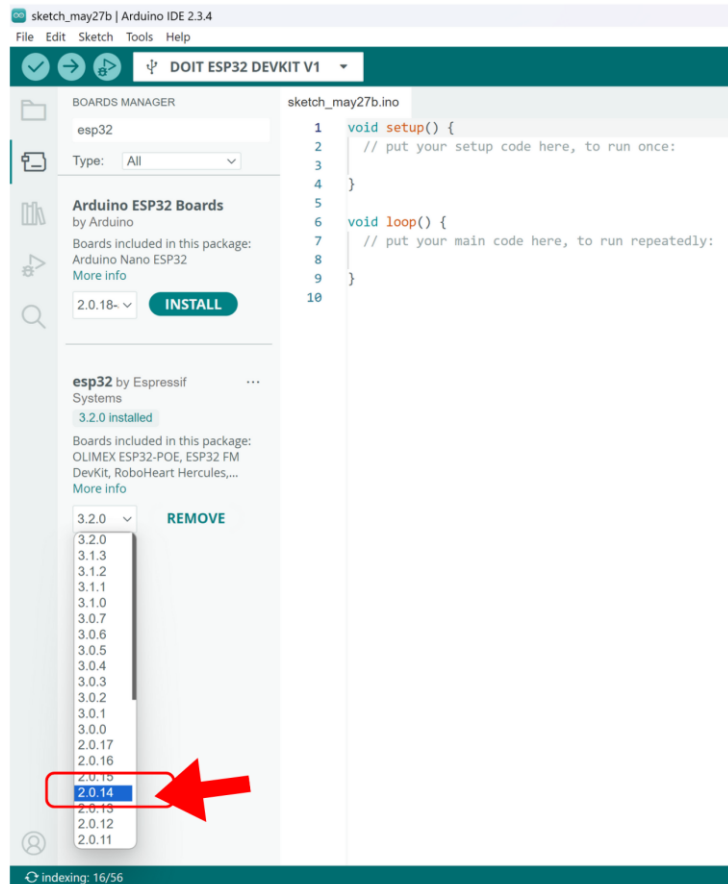
### ✅ Step 3: Install the ESP32 Boards

1. Go to **Tools > Board > Boards Manager**.



2. In the search bar, type **ESP32**. Locate **"ESP32 by Espressif Systems"**. From the dropdown select **version 2.0.14** and click **install**.

**NOTE:** If you install the latest version some of the codes might not be compatible and you will receive error during compilation of the code.



3. Wait for the installation to complete.

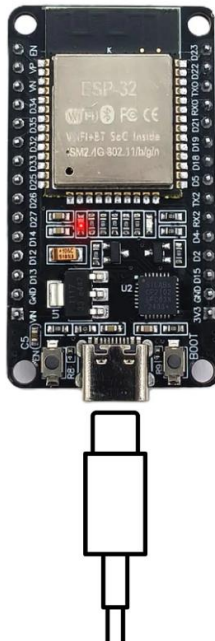
```

Output
Uninstalling esp32:mklittlefs@3.0.0-gnu12-dc7f933, tool is no more required
Uninstalling esp32:mkspliffs@0.2.3, tool is no more required
Uninstalling esp32:openocd-esp32@v0.12.0-esp32-20241016, tool is no more required
Running pre_uninstall script.
Tool esp32:openocd-esp32@v0.12.0-esp32-20241016 uninstalled
Uninstalling esp32:riscv32-esp-elf-gdb@14.2_20240403, tool is no more required
Running pre_uninstall script.
Tool esp32:riscv32-esp-elf-gdb@14.2_20240403 uninstalled
Uninstalling esp32:xtensa-esp-elf-gdb@14.2_20240403, tool is no more required
Running pre_uninstall script.
Tool esp32:xtensa-esp-elf-gdb@14.2_20240403 uninstalled
Configuring platform.
Platform esp32:esp32@2.0.14 installed

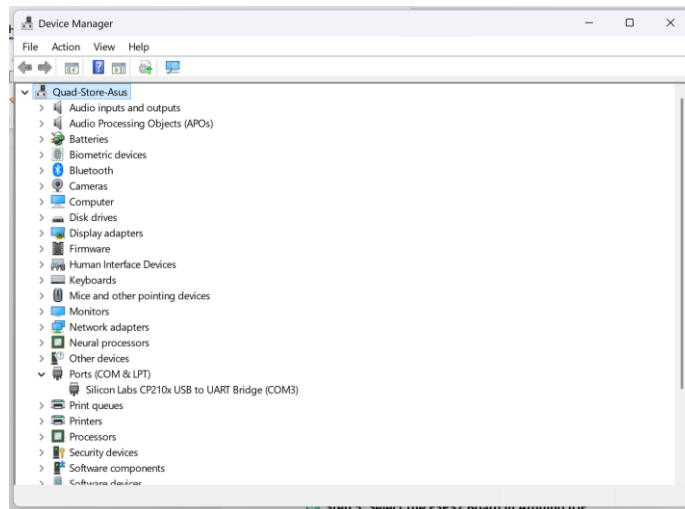
```

#### ✓ Step 4: Connect the ESP32 Board to Your PC

- Use the USB cable to connect your ESP32 board to your laptop or computer.



- Open Device Manager (Windows) or System Information (Mac) to find the COM port assigned to your board.



**NOTE:** If the COM port does not appear in the Device Manager or Arduino IDE, you may need to install the **CP210x USB to UART Bridge VCP drivers** to enable communication.

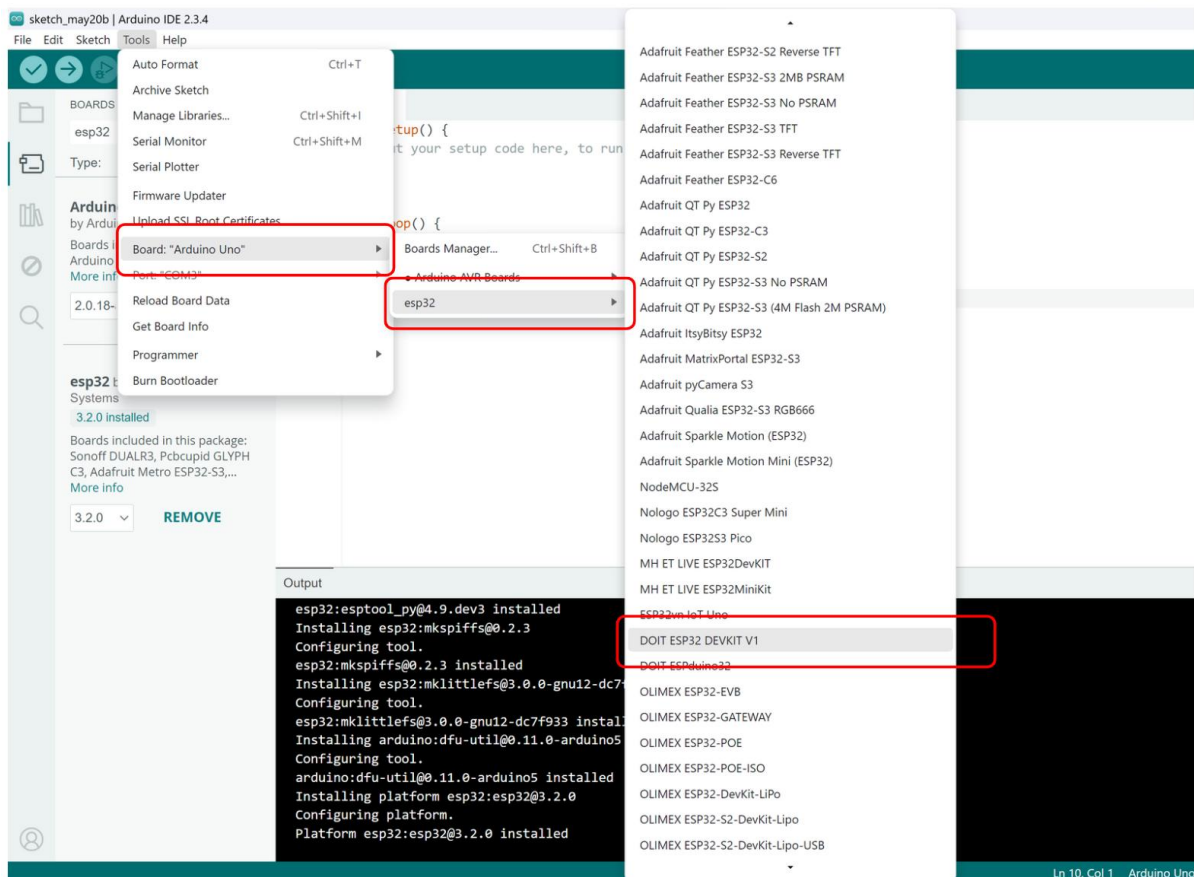
You can download and install the driver from the below link

<https://www.silabs.com/developer-tools/usb-to-uart-bridge-vcp-drivers?tab=downloads>

### ✅ Step 5: Select the ESP32 Board in Arduino IDE

1. Go to **Tools > Board**.
2. Scroll down and choose your board — for example:

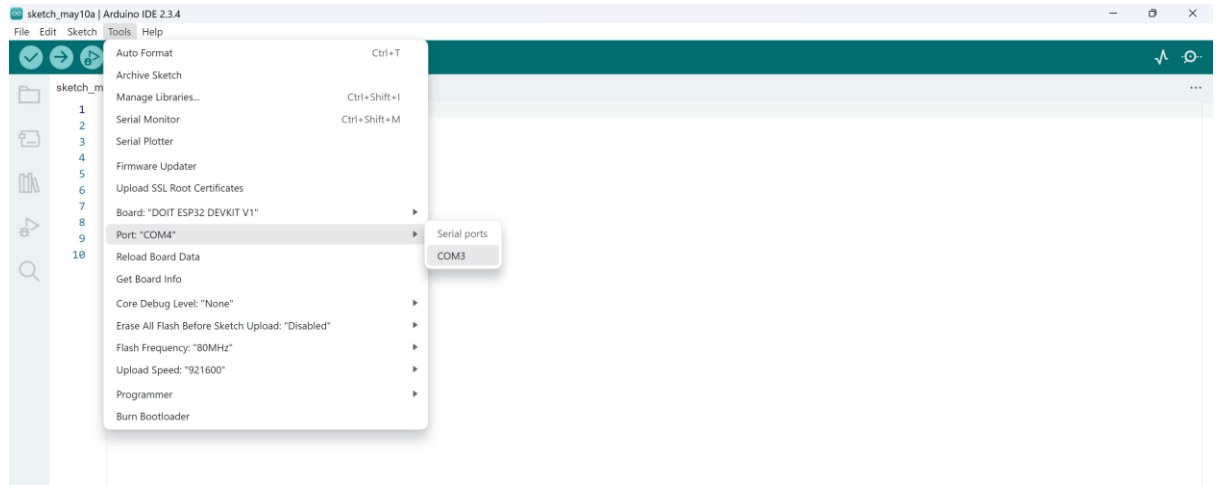
#### ✅ DOIT ESP32 DEVKIT V1



### ✅ Step 6: Select the Correct Port

1. Go to **Tools > Port**.

2. Choose the port corresponding to your ESP32 board (e.g., COM3, COM4, etc.).



💡 **NOTE:** If the COM port does not appear in the Device Manager or Arduino IDE, you may need to install the **CP210x USB to UART Bridge VCP drivers** to enable communication.

You can download and install the driver from the below link as per your operating system.

<https://www.silabs.com/developer-tools/usb-to-uart-bridge-vcp-drivers?tab=downloads>

### ✅ Step 7: Upload a Sample Program (Blink LED)

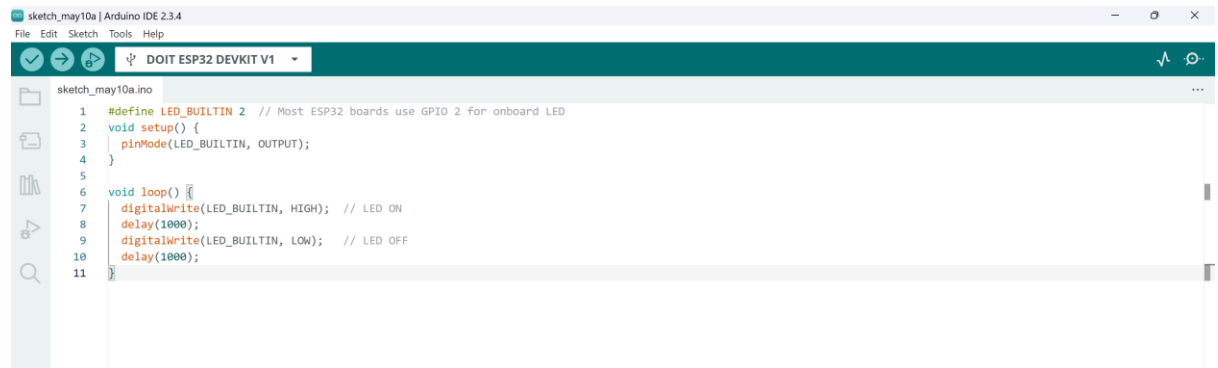
Copy this code:

```
#define LED_BUILTIN 2 // Most ESP32 boards use GPIO 2 for onboard LED
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // LED ON
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW); // LED OFF
  delay(1000);
}
```

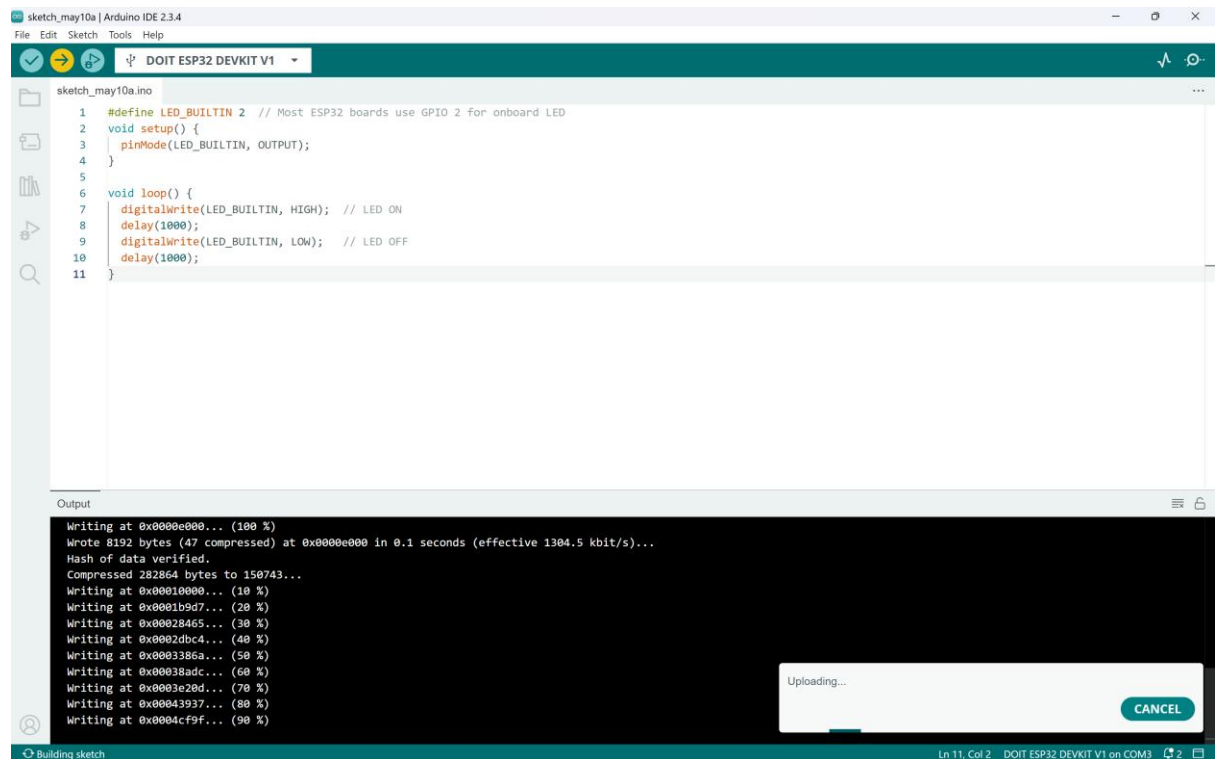


1. Paste the code in the Arduino IDE.



```
sketch_may10a | Arduino IDE 2.3.4
File Edit Sketch Tools Help
DOIT ESP32 DEVKIT V1
sketch_may10a.ino
1 #define LED_BUILTIN 2 // Most ESP32 boards use GPIO 2 for onboard LED
2 void setup() {
3   pinMode(LED_BUILTIN, OUTPUT);
4 }
5
6 void loop() {
7   digitalWrite(LED_BUILTIN, HIGH); // LED ON
8   delay(1000);
9   digitalWrite(LED_BUILTIN, LOW); // LED OFF
10  delay(1000);
11 }
```

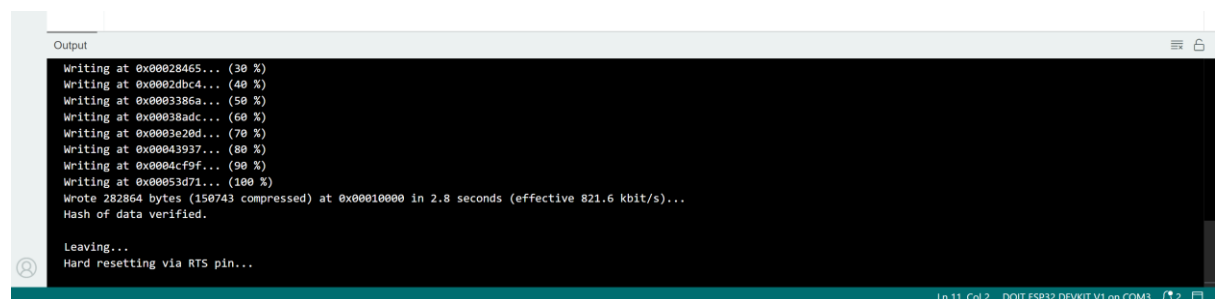
2. Click the Upload (→) button. Code should be uploaded successfully.



```
sketch_may10a | Arduino IDE 2.3.4
File Edit Sketch Tools Help
DOIT ESP32 DEVKIT V1
sketch_may10a.ino
1 #define LED_BUILTIN 2 // Most ESP32 boards use GPIO 2 for onboard LED
2 void setup() {
3   pinMode(LED_BUILTIN, OUTPUT);
4 }
5
6 void loop() {
7   digitalWrite(LED_BUILTIN, HIGH); // LED ON
8   delay(1000);
9   digitalWrite(LED_BUILTIN, LOW); // LED OFF
10  delay(1000);
11 }

Output
Writing at 0x0000e000... (100 %)
Wrote 8192 bytes (47 compressed) at 0x0000e000 in 0.1 seconds (effective 1304.5 kbit/s)...
Hash of data verified.
Compressed 282864 bytes to 150743...
Writing at 0x00010000... (10 %)
Writing at 0x0001b3d7... (20 %)
Writing at 0x00028465... (30 %)
Writing at 0x0002dbc4... (40 %)
Writing at 0x0003386a... (50 %)
Writing at 0x00038adc... (60 %)
Writing at 0x0003e20d... (70 %)
Writing at 0x00043937... (80 %)
Writing at 0x0004cf9f... (90 %)
Uploading...
CANCEL

Building sketch Ln 11, Col 2 DOIT ESP32 DEVKIT V1 on COM3
```



```
Output
Writing at 0x00028465... (30 %)
Writing at 0x0002dbc4... (40 %)
Writing at 0x0003386a... (50 %)
Writing at 0x00038adc... (60 %)
Writing at 0x0003e20d... (70 %)
Writing at 0x00043937... (80 %)
Writing at 0x0004cf9f... (90 %)
Writing at 0x00053d71... (100 %)
Wrote 282864 bytes (150743 compressed) at 0x00010000 in 2.8 seconds (effective 821.6 kbit/s)...
Hash of data verified.
Leaving...
Hard resetting via RTS pin...

Ln 11, Col 2 DOIT ESP32 DEVKIT V1 on COM3
```

3. Or open the code “27.ESPBlinkLed.ino” directly from the CODE folder and click upload button.

### If You See "Connecting..." Error:

- Hold the BOOT button on the ESP32 until it starts uploading.
- Release the button when uploading begins. Code should upload successfully.

### Step 8: View Output

1. Once uploaded, the onboard LED should start blinking (1 second ON, 1 second OFF).



### Common Issues & Fixes

Problem	Solution
"Failed to connect" error	Hold BOOT button while uploading
No COM port shown	Try another USB cable or driver (CP2102/CH340)
Board not blinking	Try GPIO 2, or check if your board uses GPIO 5
Upload stuck on "Connecting..."	Press and hold BOOT, or tap it once quickly

# Project 28: Web Server-Based Inbuilt LED Toggle Using ESP32 – IOT

This project turns the **ESP32** into a **Wi-Fi web server**, accessible from any smartphone or PC on the same network. The web page shows a **toggle switch** that turns the ESP32's **inbuilt LED ON or OFF** in real-time using **AJAX**, ensuring **fast response and no page reloads**.

## Components Required:

- ✓ ESP32 Dev Board × 1
- ✓ USB Cable × 1
- ✓ Wi-Fi Network (You need to know your **WiFi Network Username & Password**)
- ✓ Smartphone / PC × 1

 No external components are needed — uses the **inbuilt LED (GPIO 2)**.

## Wi-Fi Setup:

**IMPORTANT:** Before uploading the code, replace:

```
const char* ssid = "YOUR_SSID";
```

```
const char* password = "YOUR_PASSWORD";
```

with your actual **Wi-Fi credentials**. "YOUR\_SSID" should be replaced with your Wifi Username and "YOUR\_PASSWORD" should be replaced with your Wifi password.

**NOTE: both your SSID and Password should be within Double-Quotes “ ”.**

## ESP32 Arduino Code:

```
#include <WiFi.h>

const char* ssid = " YOUR_SSID";          // Replace with your WiFi Username
const char* password = " YOUR_PASSWORD"; // Replace with your WiFi password

WiFiServer server(80);
bool ledState = false;

void setup() {
  Serial.begin(115200);
  pinMode(LED_BUILTIN, OUTPUT);
  digitalWrite(LED_BUILTIN, LOW);

  WiFi.begin(ssid, password);
  Serial.print("Connecting to WiFi");
```

```

while (WiFi.status() != WL_CONNECTED) {
    delay(300);
    Serial.print(".");
}

Serial.println("\nConnected. IP:");
Serial.println(WiFi.localIP());

server.begin();
}

void loop() {
    WiFiClient client = server.available();
    if (!client) return;

    String request = "";
    while (client.connected()) {
        if (client.available()) {
            char c = client.read();
            request += c;

            if (c == '\n') {
                // Handle toggle requests
                if (request.indexOf("GET /toggle?state=1") != -1) {
                    ledState = true;
                    digitalWrite(LED_BUILTIN, HIGH);
                }
                if (request.indexOf("GET /toggle?state=0") != -1) {
                    ledState = false;
                    digitalWrite(LED_BUILTIN, LOW);
                }

                // Send HTML page
                if (request.indexOf("GET /toggle") == -1) {
                    client.println("HTTP/1.1 200 OK");
                    client.println("Content-Type: text/html");
                    client.println("Connection: close");
                    client.println();
                    client.println("<!DOCTYPE html><html><head><meta name='viewport'");
                    client.println("content='width=device-width, initial-scale=1'>");
                    client.println("<title>ESP32 Toggle LED</title>");
                    client.println("<style>");
                    client.println("body{font-family:sans-serif;text-align:center;padding-top:40px;}");
                    client.println("h1{margin-bottom:5px;} p{margin-top:0;color:gray;}");
                    client.println(".switch{position:relative;display:inline-block;width:60px;height:34px;}");
                    client.println(".switch input{display:none;}");
                    client.println(".slider{position:absolute;cursor:pointer;top:0;left:0;right:0;bottom:0;background-color:#ccc;transition:.4s;}");
                    client.println(".slider:before{position:absolute;content:'';height:26px;width:26px;left:4px;bottom:4px;background-color:white;transition:.4s;}");
                    client.println("input:checked + .slider{background-color:#2196F3;}");
                    client.println("input:checked + .slider:before{transform:translateX(26px);}");
                    client.println(".slider.round{border-radius:34px;}");
                    client.println(".slider.round:before{border-radius:50%;}");
                }
            }
        }
    }
}

```

```

        client.println("</style>");
        client.println("</head><body>");

        client.println("<h1>QUAD ROBOTICS</h1>");
        client.println("<p>A unit of Quad Store</p>");
        client.println("<h2>Smart Web Controlled LED</h2>");

        client.println("<div style='display:inline-block;'>");
        client.print("<label class='switch'><input type='checkbox' "
onchange='toggleLED(this)' ">");
        if (ledState) client.print("checked");
        client.println("><span class='slider round'></span></label>");
        client.println("<div style='display:flex; justify-content:space-
between; margin-top:5px;'>");
        client.println("<span style='width:60px; text-
align:left;'>OFF</span>");
        client.println("<span style='width:60px; text-
align:right;'>ON</span>");
        client.println("</div></div>");

        client.println("<script>function toggleLED(element){");
        client.println("var xhr=new XMLHttpRequest();");
        client.println("xhr.open('GET','/toggle?state='+ (element.checked?1:0
),true);");
        client.println("xhr.send();");
        client.println("}</script>");
        client.println("</body></html>");
    } else {
        client.println("HTTP/1.1 200 OK");
        client.println("Content-Type: text/plain");
        client.println("Connection: close");
        client.println();
        client.println("OK");
    }
    break;
}
}
}
}

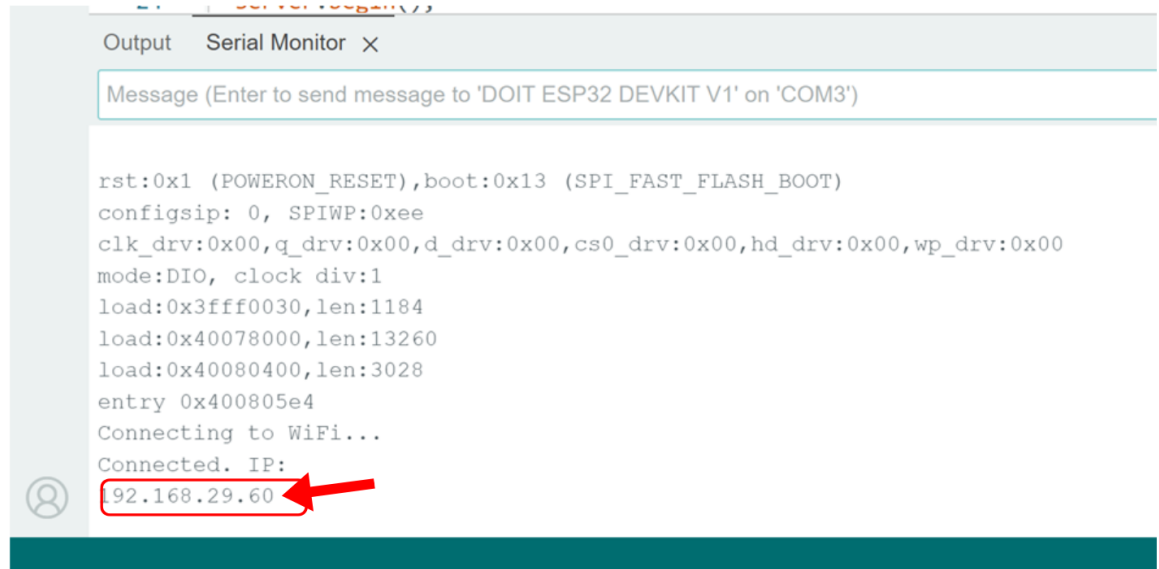
delay(1);
client.stop();
}

```

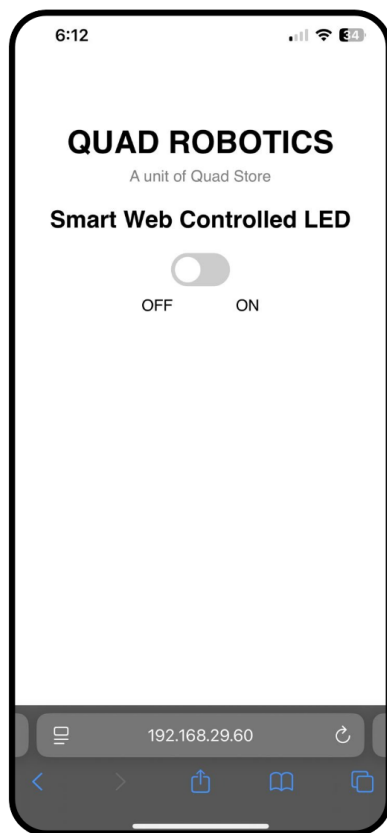
#### Steps to Run the Project:

1. Open Arduino IDE and connect your ESP32.
2. Paste the code into the IDE (**or**) open the program **28.Webserver\_LedToggle.ino** directly from CODE folder.
3. Set your Wi-Fi credentials in the CODE.
4. Select:
  - **Board:** DOIT ESP32 DEVKIT V1
  - **Port:** COMx (as shown)

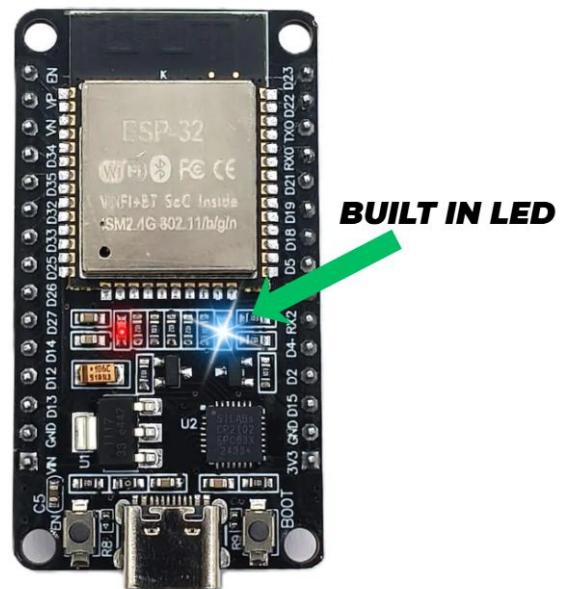
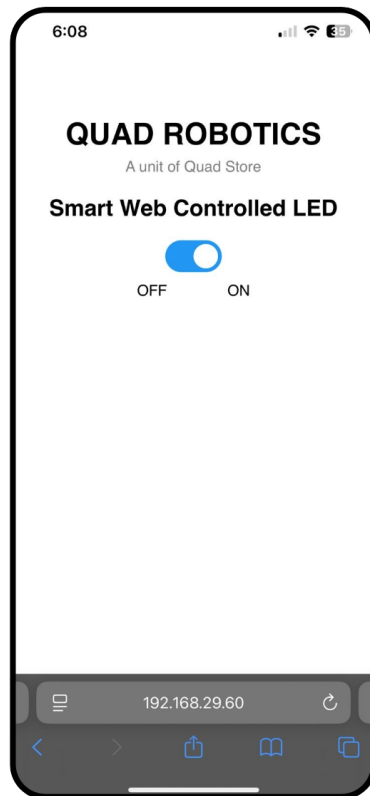
5. Click **Upload**.
6. Open **Serial Monitor** at 115200 baud and note the **IP address**. **If it does not show the IP address then press the RESET Button in the ESP32 once.**



7. Enter the IP address in a **mobile/desktop browser** on the **same Wi-Fi**. **NOTE: Your mobile and ESP32 should be connected to same WIFI network.**



8. Toggle the switch to control the **inbuilt LED** instantly.



✓ Expected Output:

Toggle Switch	LED Status
Switched ON	LED turns ON
Switched OFF	LED turns OFF