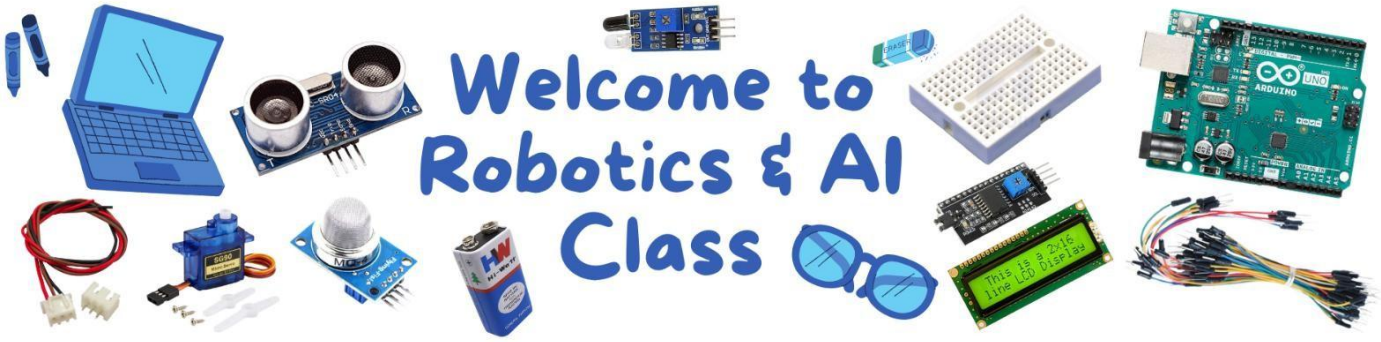


ARDUINO PROJECT MANUAL

Arduino Based Project Guide

Abstract

This workbook is designed to provide step-by-step guidance for building practical robotics and embedded system projects. Each project follows a structured approach using a standardized shield and Arduino platform to ensure easy learning and implementation.

Botics EduTech Pvt. Ltd.
Pune

Instructions for Students

- Follow the connection tables strictly
- Do not change assigned JST ports
- Upload the correct code for each project
- Verify all connections before powering '**ON**'

1. Clap Switch

1. What you Will Learn!

a. How a **sound sensor detects clap**
b. How Arduino reads **input signals**
c. How to control a **buzzer using sound**
d. Basics of **digital input & output**
e. Simple **automation using sound**

• Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	2
6	4Pin JST Connector	0
7	5V Passive/Active Buzzer	1
8	Sound Sensor	1

• Steps to make it Work!

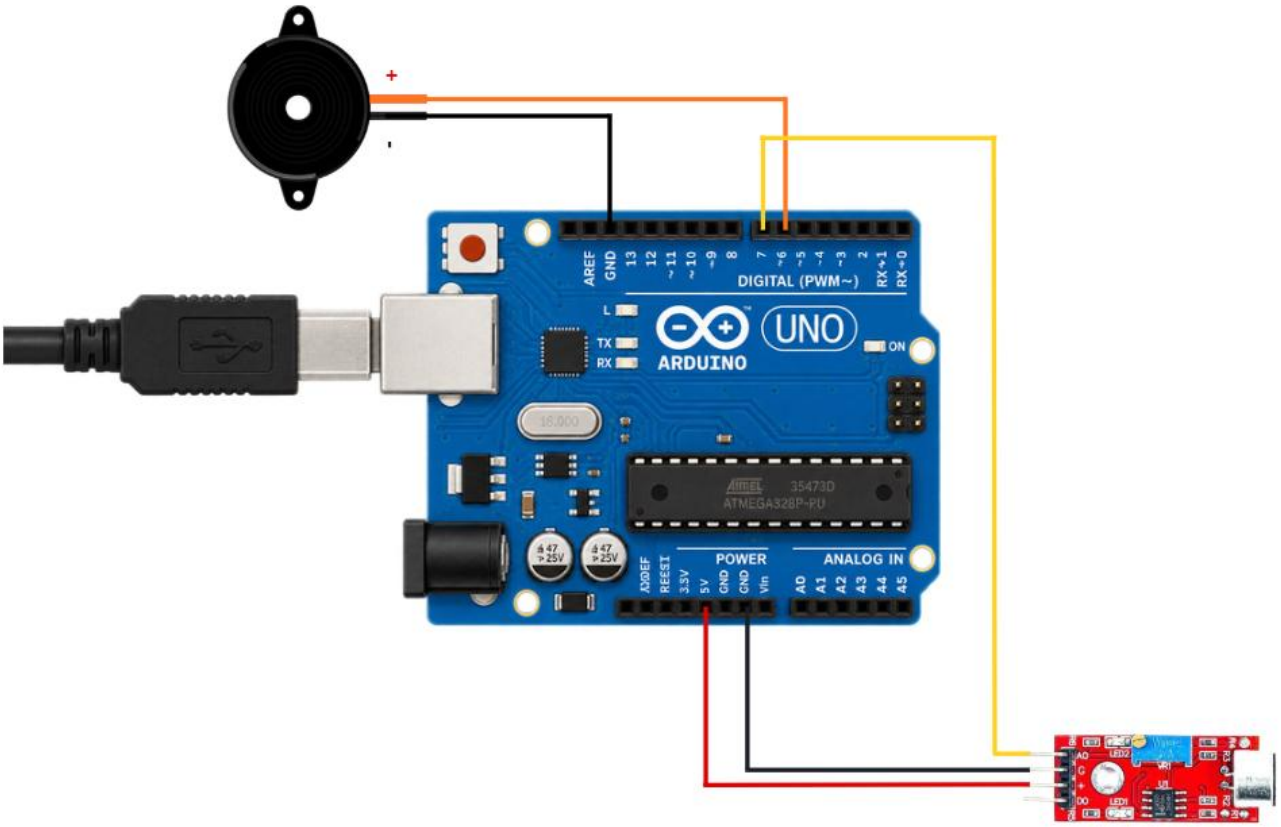
a. Connect **sound sensor to WAT-2**
b. Connect **buzzer** to BUZZ
c. Connect Arduino to laptop using USB
d. Open Arduino IDE
e. Copy and upload the code
f. Clap near the sensor

• What should Happen!

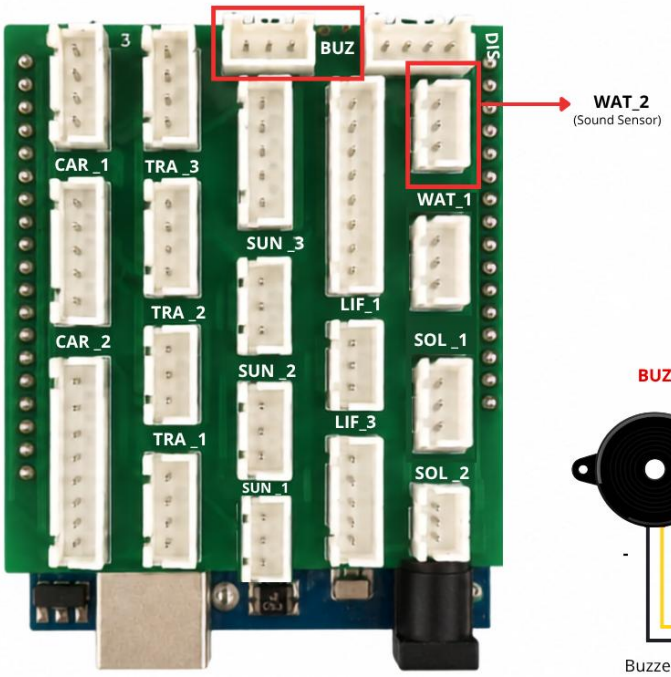
a. When you **clap**, buzzer turns **ON** }
b. When there is **no sound**, buzzer stays **OFF** +
c. Serial Monitor shows: *"Clap Detected - Buzzer ON"*

- **Connections**

Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
1: GND	1: GND	WAT_2	Sound Sensor
2: +VCC	2: +5V		
3: D0	3: D7		
1: GND (Black wire)	1: GND	BUZ	Buzzer
2: Signal	3: D6		



ARDUINO WITH 50+ PROJECTS SHIELD V.2 – CLAP SWITCH



COMPONENTS REQUIRED

1. Sound Sensor



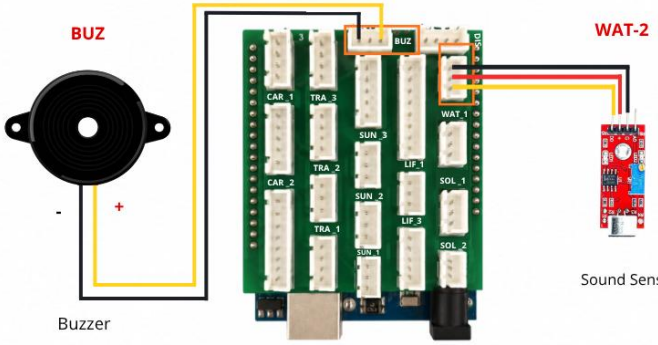
(Connect to WAT-2)

2. Buzzer (or LED)



(Connect to BUZ)

CONNECTION DIAGRAM





SHIELD CONNECTION SUMMARY

Component	Shield Port	Description
Sound Sensor	WAT-2	Detects clap / sound
Buzzer	BUZ	Gives audio/visual output

- **Program**

```
int soundflin = 7;        // Sound sensor digital output
int buzzerflin = 6;   // Buzzer pin

void setup()
{
  pinMode(soundflin, INflUT);
  pinMode(buzzerflin, OUTflUT);

  Serial.begin(9600);
}

void loop()
{
  int soundState = digitalRead(soundflin);

  if(soundState == HIGH)    // Clap detected
  {
    digitalWrite(buzzerflin, HIGH);
    Serial.println("Clap Detected - Buzzer ON");
    delay(500);
  }
  else
  {
    digitalWrite(buzzerflin, LOW);
  }
}
```

2. Water Level Indicator

- ## What you Will Learn!

a. How a **water level sensor detects water**
b. How Arduino reads **sensor signals**
c. How to control a **relay (pump/indicator)**
d. How to use a **buzzer for alerts**
e. Basics of **automation using sensors**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	2
6	4Pin JST Connector	0
7	Water level Indicator	1
8	LED/Buzzer	1
9	Glass Full of Water	1

- ## Steps to make it Work!

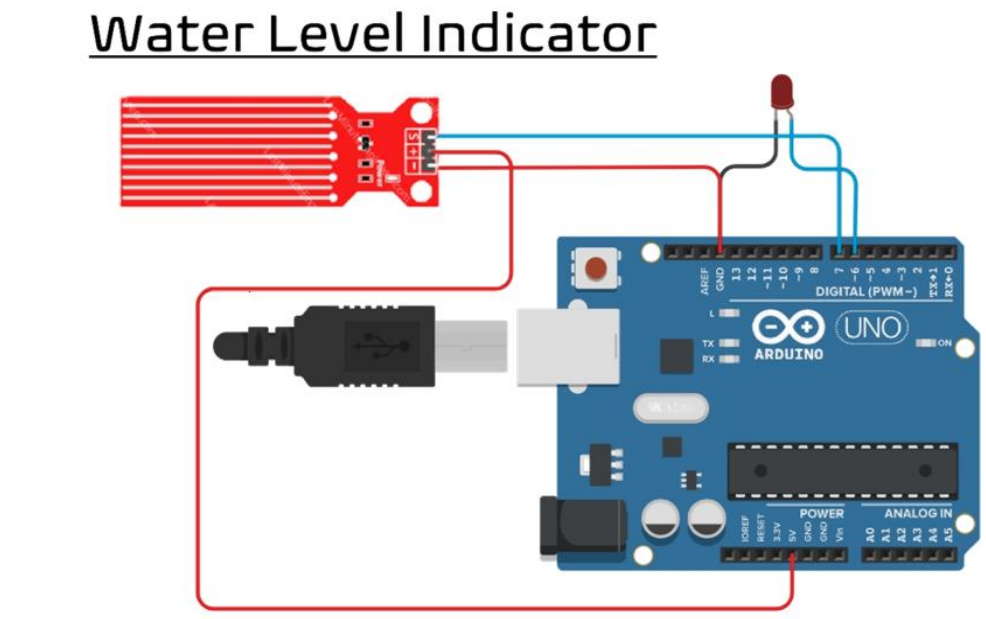
- Connect **water sensor** to WAT-2
- Connect **buzzer** to BUZ
- Connect Arduino to laptop
- Open Arduino IDE
- Upload the code
- Dip sensor in water

- ## What should Happen!

a. When **water is detected**:
    i. Buzzer starts **beeping**
b. When **no water**:
    i. Buzzer **OFF**

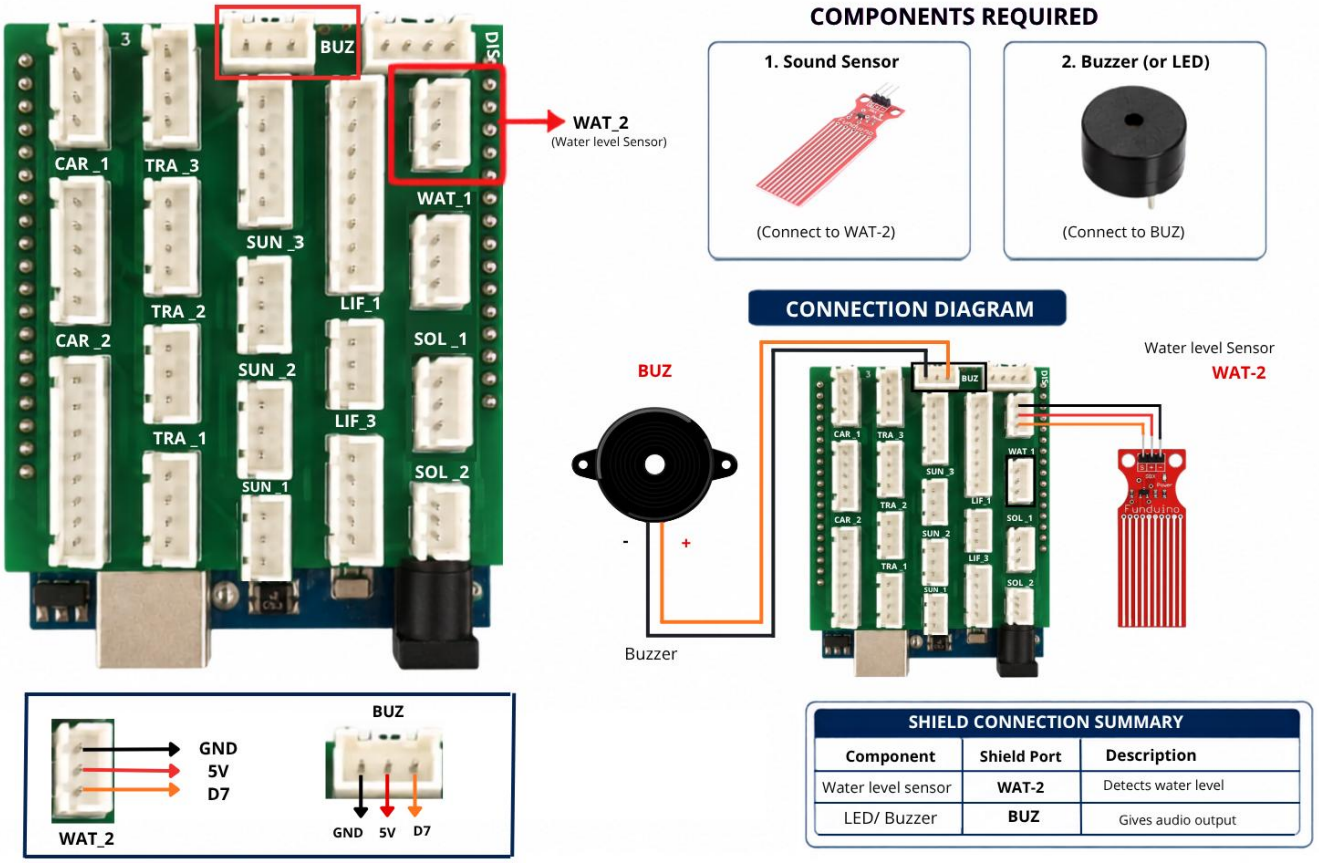
- **Connections**

Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
GND	1: GND	WAT_2	Water Sensor
+VCC	2: +5V		
DO	3: D7		
GND (Black wire)	1: GND	BUZ	Buzzer
Signal	3: D6		



Connection Diagram

WATER LEVEL INDICATOR



Program

```
int buzzerflin =  6;
int waterSensorflin = 7;

int water_level = 0;

void setup() {

pinMode(buzzerflin, OUTflUT);
pinMode(waterSensorflin, INflUT);

Serial.begin(9600);
}

void loop() {

water_level =
digitalRead(waterSensorflin);
Serial.println(water_level);

if (water_level == LOW) {    //
water detected

tone(buzzerflin, 1000);       //

generate 1kHz sound

}
else {

noTone(buzzerflin);           //

stop buzzer
}

delay(200);
}
```

3. Bi-Directional Counter

- ## What you Will Learn!

a. How **IR sensors detect movement**
b. How to count **people entering and exiting**
c. How Arduino uses **logic to increase/decrease count**
d. How to display data on **LCD (I2C)**
e. Basics of **automation & smart systems**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	2
6	4Pin JST Connector	1
7	IR Module	2
8	I2C Display Module	1

- ## Steps to make it Work!

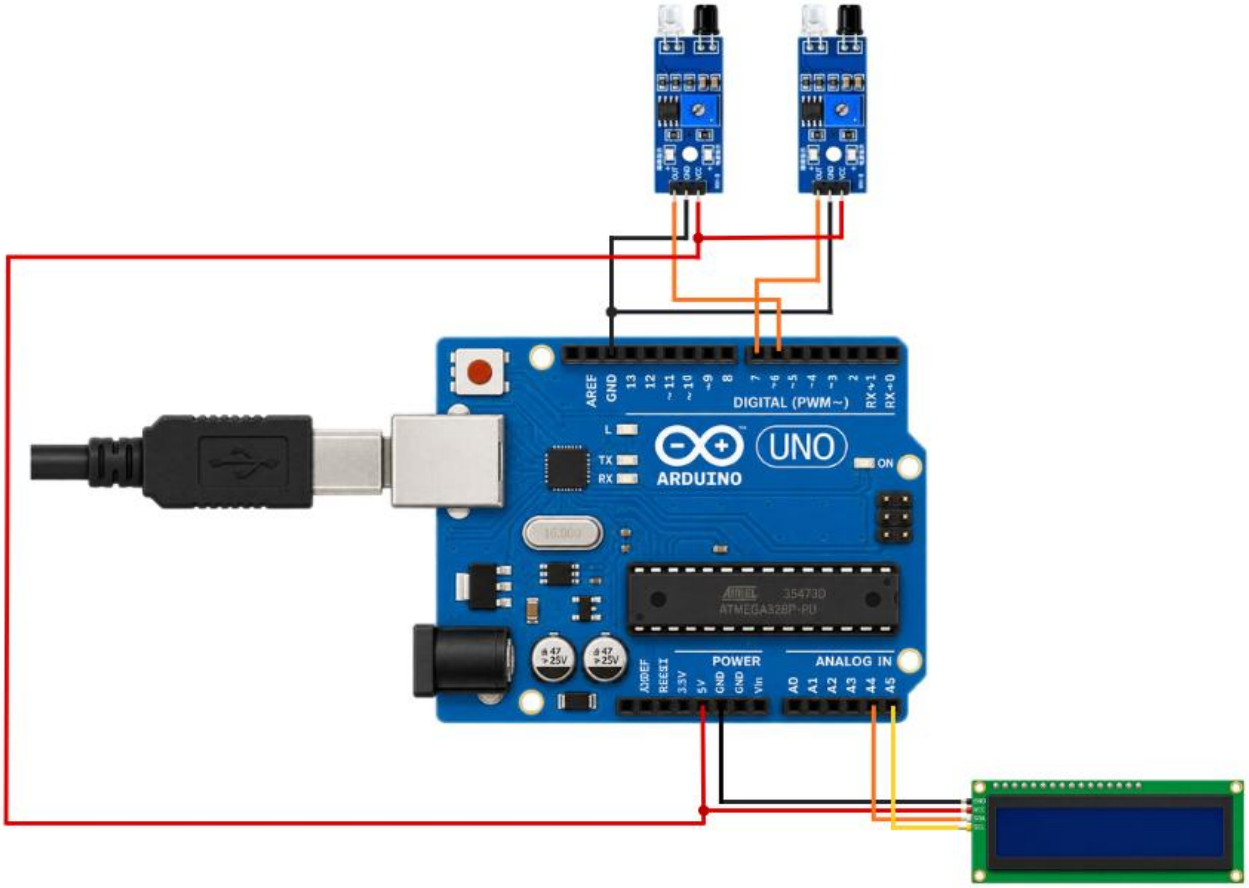
a. Connect **IR Sensor 1 to SUN-1, and Sensor 2 to SUN-2**
b. Connect **LCD display** to CAR-3
c. Connect Arduino to laptop
d. Open Arduino IDE
e. Upload the code
f. Move hand across sensors 7

- ## What should Happen!

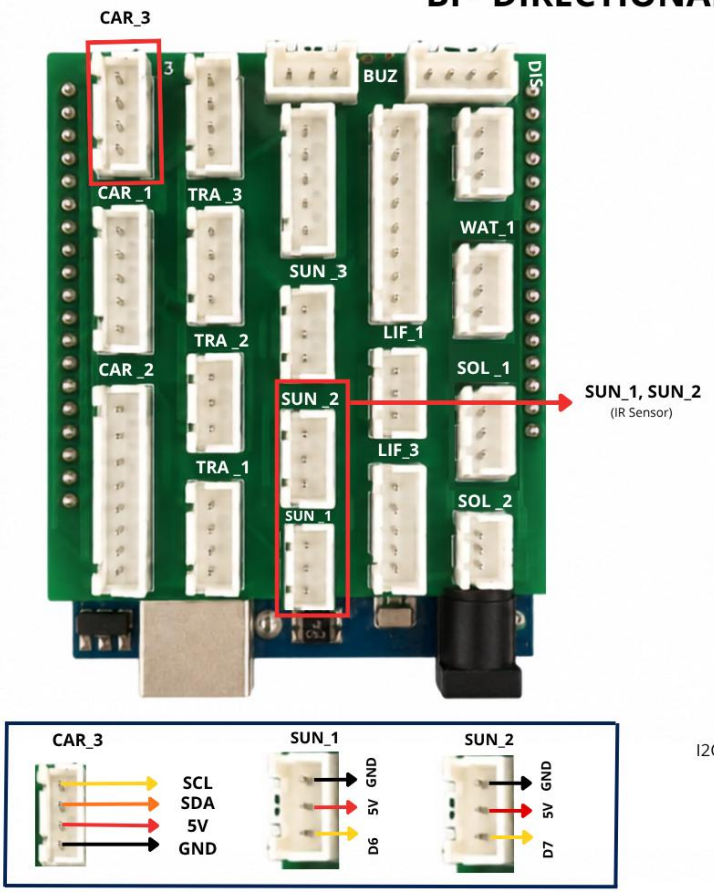
a. When someone **enters** → Count increases (IN++)
b. When someone **exits** → Count decreases (OUT++)
c. LCD shows: Number of People Left and Entered.

- **Connections**

Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
1: SCL	1: SCL	CAR_3	LCD Display
2: SDA	2: SDA		
3: +5V	3: +5V		
4: GND	4: GND		
2: GND	1: GND	SUN_1	IR Module 1
1: VCC	2: +5V		
3: OUT	3: D6		
2: GND	1: GND	SUN_2	IR Module 2
1: VCC	2: +5V		
3: OUT	3: D7		



BI - DIRECTIONAL COUNTER



COMPONENTS REQUIRED

1. IR Sensor



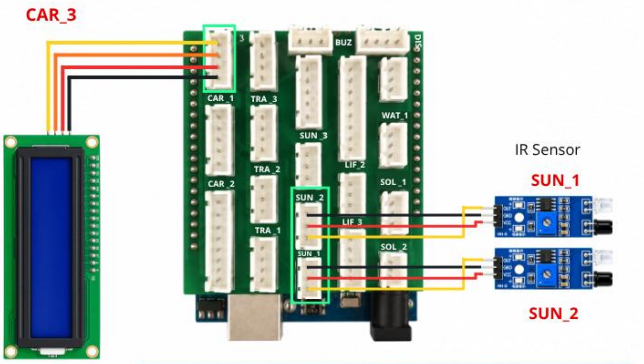
(Connect to SUN_1, SUN_2)

2. I2C LCD Display



(Connect to CAR_3)

CONNECTION DIAGRAM



SHIELD CONNECTION SUMMARY

Component	Shield Port	Description
IR Sensor	SUN_1, SUN_2	Detect people entering and leaving.
LCD_Display	CAR_3	Displays the current count.

Program

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

LiquidCrystal_I2C lcd(0x27,16,2);

#define IR1 6
#define IR2 7

int inCount = 0;
int outCount = 0;
int total = 0;

bool lastIR1 = HIGH;
bool lastIR2 = HIGH;

void setup()
{
pinMode(IR1, INflUT);
pinMode(IR2, INflUT);

Serial.begin(9600);

lcd.init();
lcd.setCursor(0,0);
lcd.print("Room Counter");
delay(2000);
lcd.clear();

updateDisplay();
}

void loop()
{
 bool s1 = digitalRead(IR1);
 bool s2 = digitalRead(IR2);

// ENTRY
if(s1 == LOW fifi lastIR1 ==
HIGH)
{
inCount++;
total++;

Serial.println("flerson
Entered");
```

```
  lcd.backlight();

updateDisplay();
delay(300);
}

// EXIT (only if someone
inside)
if(s2 == LOW fifi lastIR2 ==      HIGH
fifi total > 0)
{
outCount++;
total--;

Serial.println("flerson
Exited");

updateDisplay();
delay(300);
}

lastIR1 = s1;
lastIR2 = s2;
}
```

```
void updateDisplay()
{
lcd.clear();

lcd.setCursor(0,0);
lcd.print("IN:");
lcd.print(inCount);

lcd.setCursor(8,0);
lcd.print("OUT:");
lcd.print(outCount);

lcd.setCursor(0,1);
lcd.print("TOTAL:");
lcd.print(total);

Serial.print("IN:  ");
Serial.print(inCount);
Serial.print(" OUT: ");
Serial.print(outCount);
Serial.print(" TOTAL: ");
Serial.println(total);
}
```

4. Volume Controller

- ## What you Will Learn!

a. How a **potentiometer controls input values**
b. How Arduino reads **analog signals**
c. How to generate sound using a **buzzer**
d. How to control **frequency (sound pitch)**
e. How to display values on **LCD (I2C)**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	2
6	4Pin JST Connector	1
7	Buzzer	1
8	Potentiometer	1

- ## Steps to make it Work!

a. Connect **potentiometer** to WAT-1
b. Connect **buzzer** to BUZZ
c. Connect **LCD display** to CAR-3
d. Connect Arduino to laptop
e. Open Arduino IDE
f. Upload the code
g. Rotate the knob

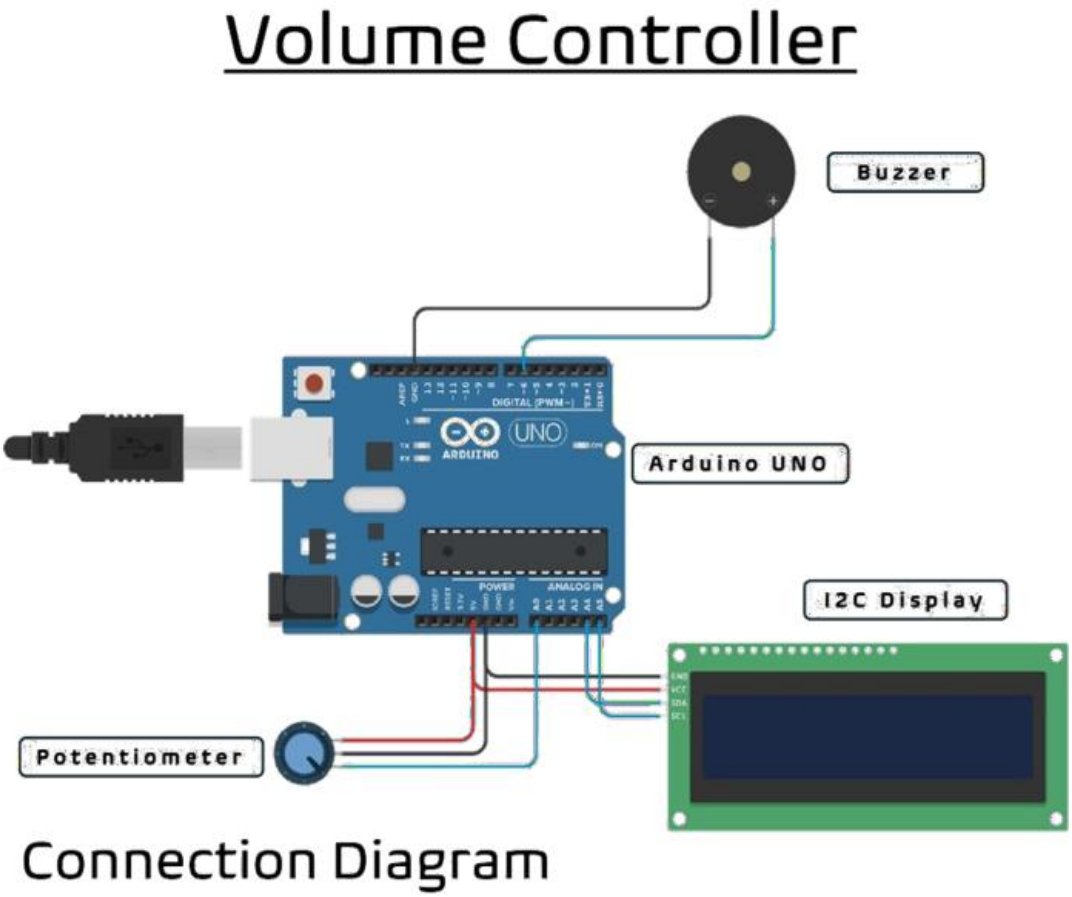
- ## What should Happen!

a. When you rotate the **potentiometer**:

    i. Sound pitch **changes**
    ii. LCD shows **frequency value (Hz)**

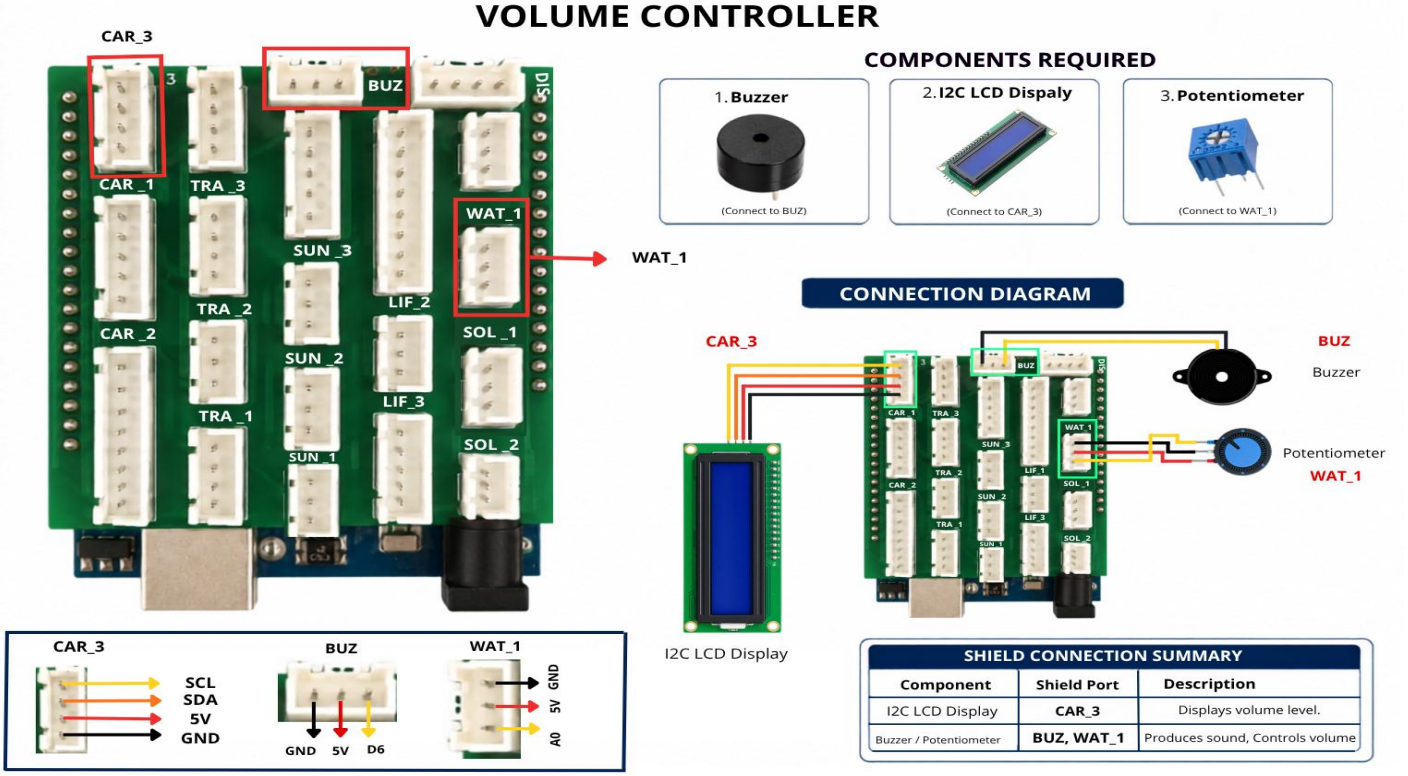
- **Connections**

Sensor PIN	UNO Shield PIN	Shield Connector	Sensor(Component
1: SDA	2: SDA	CAR_3	LCD Display
2: SCL	1: SCL		
3: +5V	3: +5V		
4: GND	4: GND		
1: GND	1: GND	WAT_1	Potentiometer
3: VCC	2: +5V		
2: OUT	3: A0		
1: GND	1: GND	BUZ	Buzzer
2: VCC	2: D6		



Connection Diagram

VOLUME CONTROLLER



Program

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

LiquidCrystal_I2C lcd(0x27, 16,
2);

const int potflin = A0;
const int buzzerflin = 6;

int frequency = 0;

void setup() {
pinMode(buzzerflin, OUTflUT);

lcd.init();
lcd.backlight();

lcd.setCursor(0, 0);
lcd.print("Voice Controller");
delay(1500);
lcd.clear();
}

void loop() {

int potValue =
analogRead(potflin);

// Map 0-1023 to 0-1500 Hz
frequency = map(potValue, 0,
1023, 0, 1500);

if (frequency > 0) {
tone(buzzerflin, frequency);
} else {
noTone(buzzerflin);
}

lcd.setCursor(0, 0);
lcd.print("Sound Frequency:");

lcd.setCursor(0, 1);
lcd.print(frequency);
lcd.print(" Hz          ");    // Clear
extra characters

delay(100);
}
```

5. Distance Meter

- ## What you Will Learn!

b. How an **ultrasonic sensor measures distance**
c. How Arduino calculates **distance using time**
d. How to display values on **LCD (I2C)**
e. Basics of **signal processing (averaging readings)**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	0
6	4Pin JST Connector	2
7	Ultrasonic Sensor	1
8	I2C Display module	1
9	Ruler Scale (For Measurement)	1

- ## Steps to make it Work!

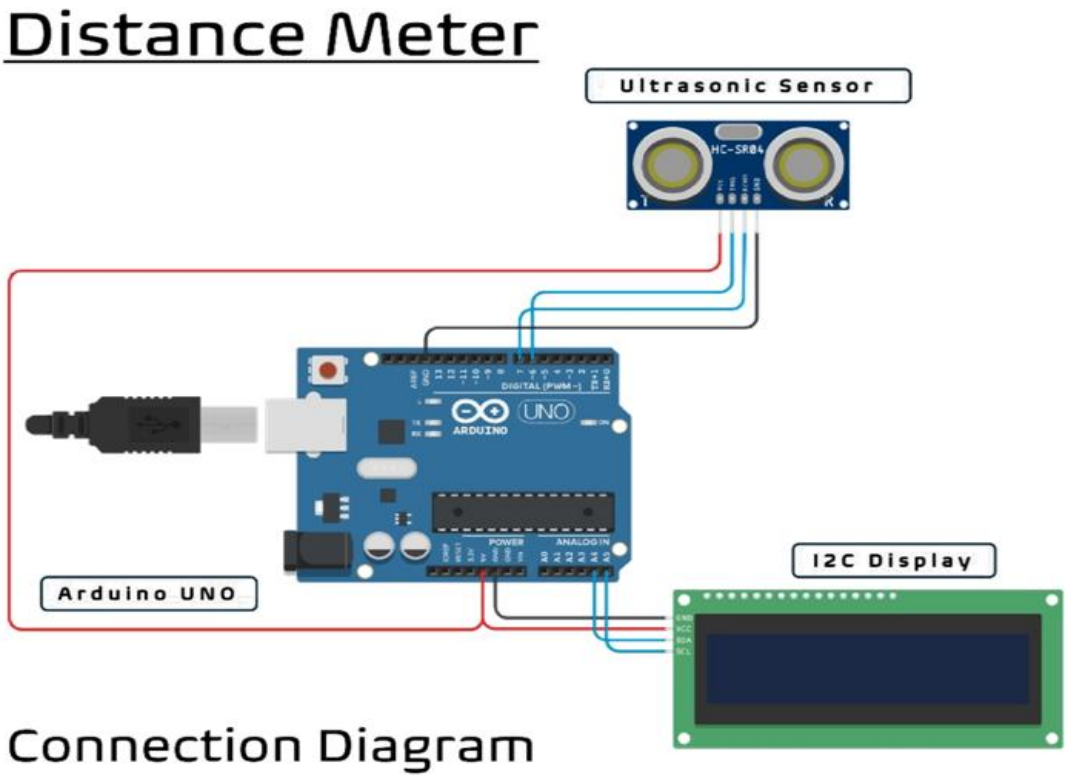
a. Connect **ultrasonic sensor** to DIS
b. Connect **LCD display** to CAR-3
c. Connect Arduino to laptop
d. Open Arduino IDE
e. Upload the code
f. Place object in front of sensor ú

- ## What should Happen!

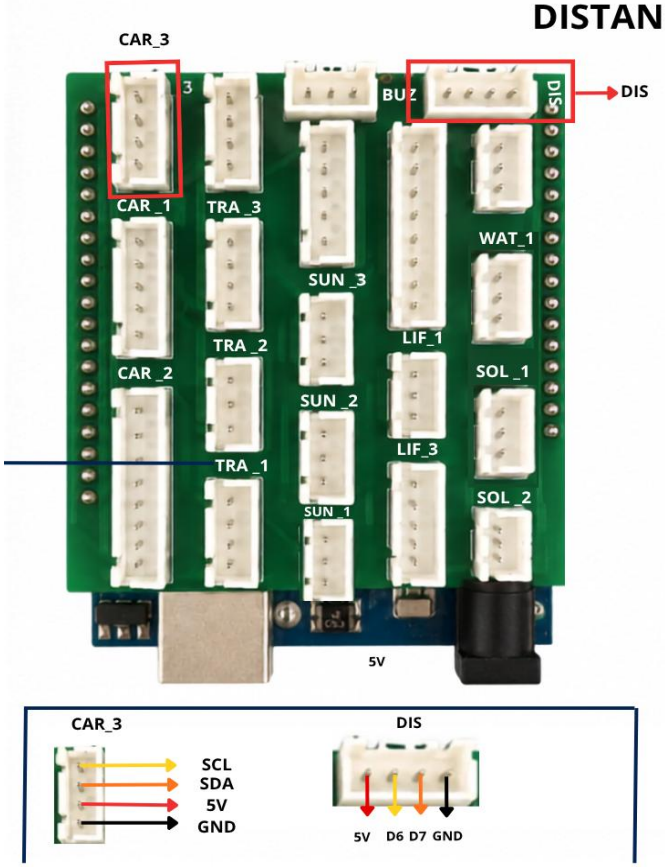
a. LCD shows **distance in cm**
b. Moving object closer → value **decreases**
c. Moving object away → value **increases**
d. If no object → shows **"Out of Range"**

- **Connections**

Component / Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
1: SDA	2: SDA	CAR_3	LCD Display
2: SCL	1: SCL		
3: +5V	3: +5V		
4: GND	4: GND		
1: VCC	1: +5V	DIS	Ultrasonic Sensor
3: TRIG	2: D6		
2: ECHO	3: D7		
4: GND	4: GND		



DISTANCE METER



COMPONENTS REQUIRED

1. Ultrasonic Sensor



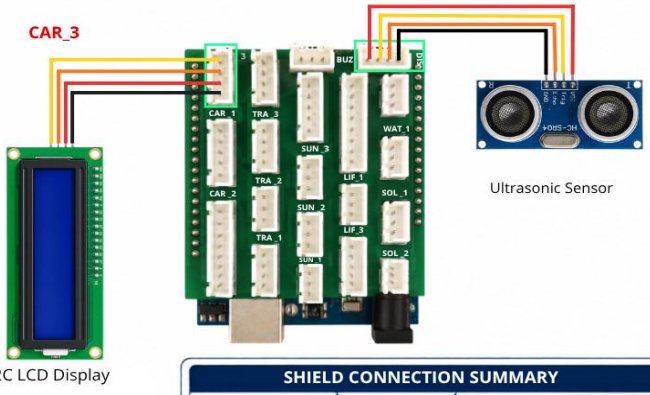
(Connect to DIS)

2. I2C LCD Display



(Connect to CAR_3)

CONNECTION DIAGRAM



SHIELD CONNECTION SUMMARY

Component	Shield Port	Description
Ultrasonic Sensor	DIS	Measures distance.
LCD_Display	CAR_3	Displays the measured distance.

Program

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

LiquidCrystal_I2C lcd(0x27, 16, 2);

const int trigflin = 6;
const int echoflin = 7;

float getSingleDistance() {

digitalWrite(trigflin, LOW);
delayMicroseconds(5);

digitalWrite(trigflin, HIGH);
delayMicroseconds(10);
digitalWrite(trigflin, LOW);

    long duration = pulseIn(echoflin, HIGH, 30000);

if (duration <= 0) return -1;

    return (duration * 0.0343) / 2.0;
}

float getFilteredDistance() {

    const int samples = 7;        // number of readings
float total = 0;
int valid = 0;

    for (int i = 0; i < samples; i++) {
    float d = getSingleDistance();
if (d > 0) {
total += d;
valid++;
}
    delay(40);        // sensor settling time
}

if (valid == 0) return -1;
```

```
    return total / valid;       //
average
}

void setup() {

pinMode(trigflin, OUTflUT);
pinMode(echoflin, INflUT);

lcd.init();
lcd.backlight();
lcd.clear();

lcd.setCursor(0, 0);
lcd.print("Distance Meter");
}

void loop() {

    float distance =
getFilteredDistance();

lcd.setCursor(0, 1);
    lcd.print("
");

lcd.setCursor(0, 1);

if (distance < 0) {
lcd.print("Out of Range");
} else {
lcd.print(distance, 2);
lcd.print(" cm");
}

delay(300);
}
```

6. Automatic Window Curtains

- ## What you Will Learn!

- How an **LDR (light sensor) detects light intensity**
- How Arduino reads **analog values**
- How to control a **servo motor**
- Basics of **automation using environment conditions**
- Real-life concept of **smart homes**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	1
6	4Pin JST Connector	0
7	LDR Module	1
8	Servo Motor with Connector and Horn	1

- ## Steps to make it Work!

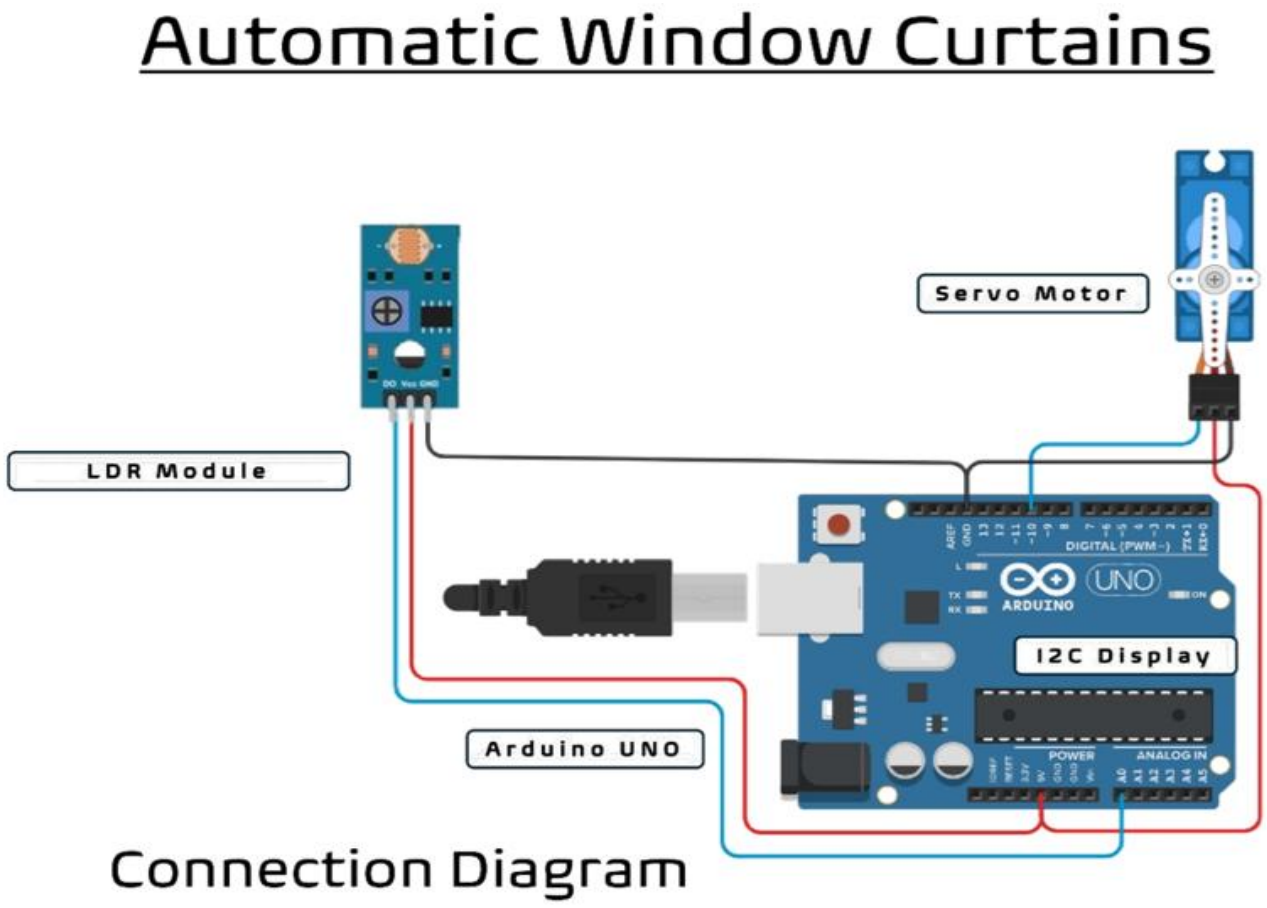
a. Connect **LDR module** to WAT-1
b. Connect **servo motor** to LIF-1
c. Connect Arduino to laptop
d. Open Arduino IDE
e. Upload the code
f. Change light on sensor (use torch 🟡)

- ## What should Happen!

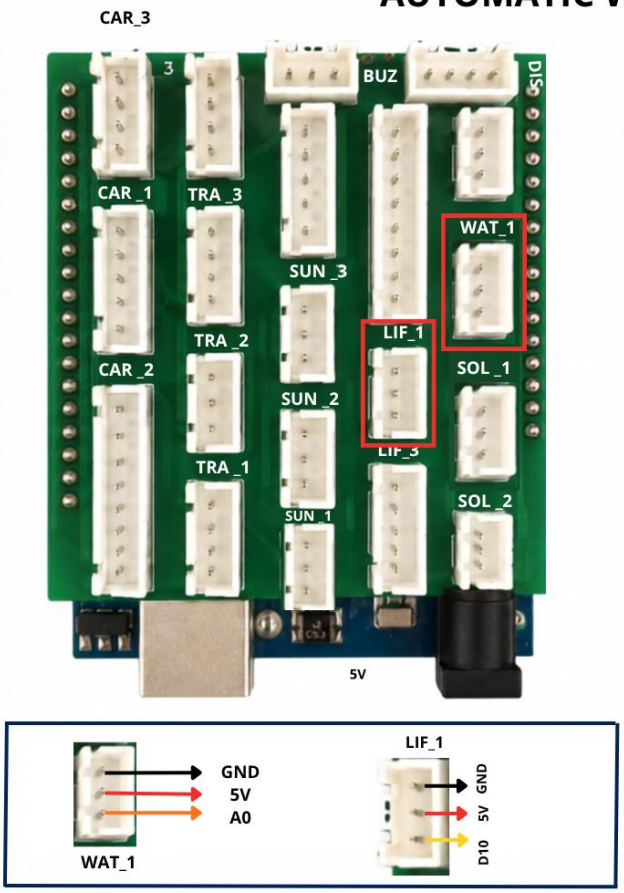
a. When **light is high (day)** → Curtain **opens**
b. When **light is low (night)** → Curtain **closes**
c. Servo rotates automatically based on light

- **Connections**

Component / Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
1: GND	1: GND	WAT_1	LDR Module
2: VCC	2: +5V		
3: Signal	3: A0		
1: GND	1: GND	LIF_1	Servo Motor
2: VCC	2: +5V		
3: Signal	3: D10		



AUTOMATIC WINDOW CURTAINS



COMPONENTS REQUIRED

1. LDR Module



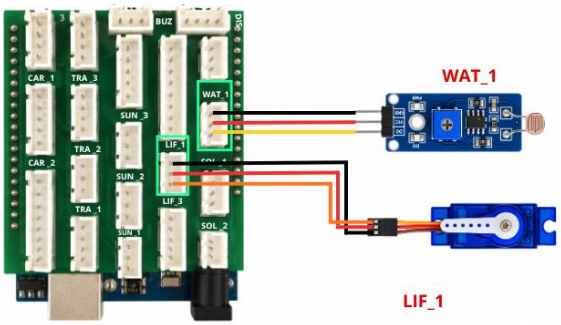
(Connect to WAT_1)

2. Servo Motor



(Connect to LIF_1)

CONNECTION DIAGRAM



SHIELD CONNECTION SUMMARY

Component	Shield Port	Description
LDR Mdule	WAT_1	Detects light intensity
Servo Motor	LIF_1	Opens and closes the curtains automatically

Program

```
#include <Servo.h>

#define LDR_flN A0
#define SERVO_flN 10

Servo curtain;
int threshold = 500;          // Adjust
this value after testing

void setup() {
Serial.begin(9600);
curtain.attach(SERVO_flN);
curtain.write(0);          // Curtain
CLOSED
}

void loop() {

int ldrValue =
analogRead(LDR_flN);
Serial.println(ldrValue);

if (ldrValue > threshold) {
curtain.write(90);          // OflEN
(Bright)
} else {
curtain.write(0);          // CLOSE
(Dark)
}

delay(500);
}
```

7. Calculator

- ## What you Will Learn!

a. How a **keypad takes user input**
b. How Arduino performs **basic calculations**
c. How to display data on **LCD (I2C)**
d. Basics of **logic building (operations)**
e. Working of a simple **embedded calculator**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	0
6	4Pin JST Connector	1
7	I2C Display Module	1
8	4X4 Keypad with Connector	1

- ## Steps to make it Work!

a. Connect **4x4 keypad** to CAR-1, CAR-2
b. Connect **LCD display** to CAR-3
c. Connect Arduino to laptop
d. Open Arduino IDE
e. Upload the code
f. Press keys to perform calculation

- ## What should Happen!

a. Press numbers → display on LCD
b. Press:

A → + (Addition) | B → — (Subtraction) | C → + (Multiplication) | D → ÷ (Division)

c. Press # → shows result

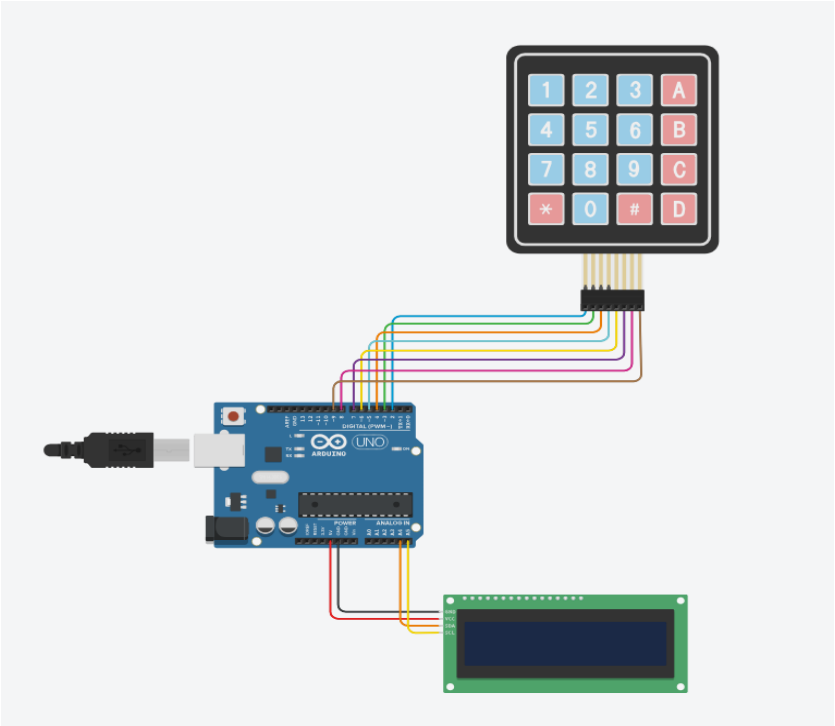
d. Press * → clears screen

- **Connections**

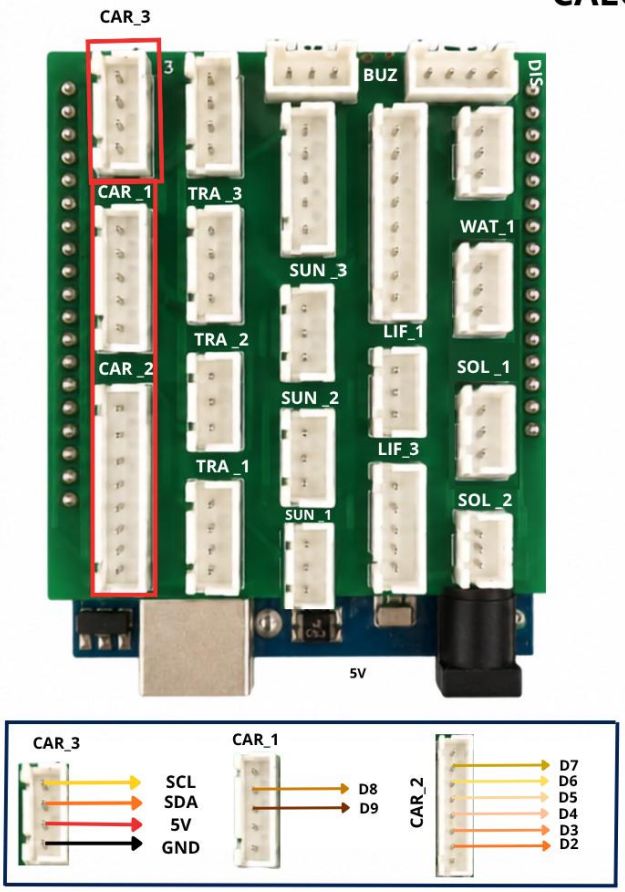
Please Note - (1: SDA ----> 2: SDA (here 1, 2 are port pin no. and SDA name of pin/connection)

Component / Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
1: SDA	2: SDA	CAR_3	LCD Display
2: SCL	1: SCL		
3: +5V	3: +5V		
4: GND	4: GND		
1: R1	7: D2	Car_2	4x4 Keypad
3: R2	6: D3		
2: R3	5: D4		
4: R4	4: D5		
5: C1	3: D6		
6: C2	2: D7		
7: C3	2: D8	Car_1	
8: C4	3: D9		

Calculator



CALCULATOR



COMPONENTS REQUIRED

1. 4×4 Keypad



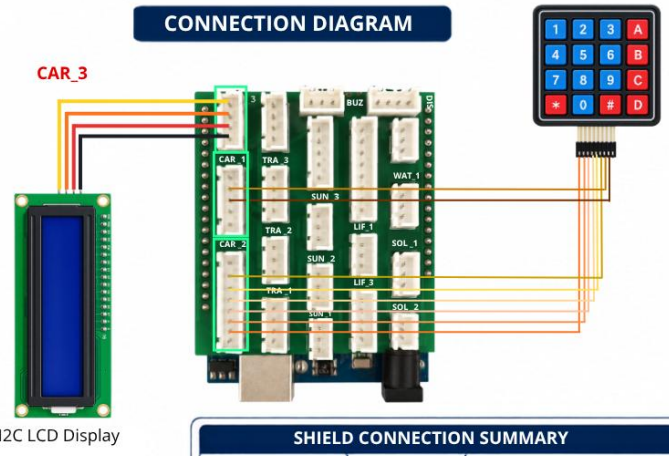
(Connect to LIF_2)

2. I2C LCD Display



(Connect to CAR_3)

CONNECTION DIAGRAM



SHIELD CONNECTION SUMMARY

Component	Shield Port	Description
4×4 Keypad	CAR_1, CAR_2	Enters numbers and operations
LCD_Display	CAR_3	Displays the calculation and result

Program

```
include <Keypad.h>
include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16,

2);


const byte ROWS = 4;
const byte COLS = 4;


char keys[ROWS][COLS] = {
{'1','2','3','A'},
{'4','5','6','B'},
{'7','8','9','C'},
{'*','0','#','D'}
};

byte rowflins[ROWS] = {2, 3, 4,
5};

byte colflins[COLS] = {6, 7, 8,
9};

 Keypad keypad
          =
Keypad(makeKeymap(keys),
rowflins,
colflins, ROWS, COLS);

 float num1 =
       0;
float num2 = 0;
char op = 0;
bool second = false;

void setup() {
  lcd.init();
lcd.backlight();
lcd.print("Calculator");
delay(1500);
lcd.clear();
}
```

```
    void loop() {
    char key = keypad.getKey();
    if (!key) return;

    // CLEAR (*)
    if (key == '*') {
    lcd.clear();
    num1 = 0;
    num2 = 0;
    second = false;
          op = 0;
          return;
  }

// NUMBER INflUT
if (key >= 'O' fifi key <= '9') {
lcd.print(key);

if  (!second)
   num1 = num1 * 10 + (key -
'O');
else
   num2 = num2 * 10 + (key -
'O');
}

//  OflERATORS
   if (key == 'A' || key == 'B' ||
key == 'C' || key == 'D') {
op = key;
second = true;

   if (key == 'A')
lcd.print("+");
      if (key == 'B') lcd.print("-
");
      if (key == 'C')
   lcd.print("*");
      if (key == 'D')
   lcd.print("/");
     }

     // EQUAL (#)
     if (key == '#') {
       lcd.setCursor(0, 1);
       lcd.print("=");

       float result = 0;

if (op == 'A') result = num1
   + num2;
if (op == 'B') result = num1
      num2;
if (op == 'C') result = num1
num2;

if (op == 'D') {
if (num2 != 0)
result = num1 / num2;
else {
lcd.print("Error");
return;
}
}

   lcd.print(result, 2);          //
Show 2 decimal places
}
}
```

8. LPG Gas Detector

- ## What you Will Learn!

- How an **MQ-2 gas sensor detects gas (LPG, smoke)**
- How Arduino reads **analog values**
- How to convert readings into **PPM (gas level)**
- How to use a **buzzer for alerts**
- How safety systems work in **real life**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	2
6	4Pin JST Connector	1
7	LED or Buzzer Active/Passive	1
8	MQ-02 Gas Sensor	1
9	LPG Lighter (Only with trainer)	1

- ## Steps to make it Work!

a. Connect **MQ-2 gas sensor** to WAT-1
b. Connect **buzzer** to BUZ
c. Connect **LCD display** to CAR-3
d. Connect Arduino to laptop
e. Open Arduino IDE
f. Upload the code
g. Bring gas source near sensor (carefully ı)

- ## What should Happen!

a. LCD shows **gas level in PPM**
b. Three zones:

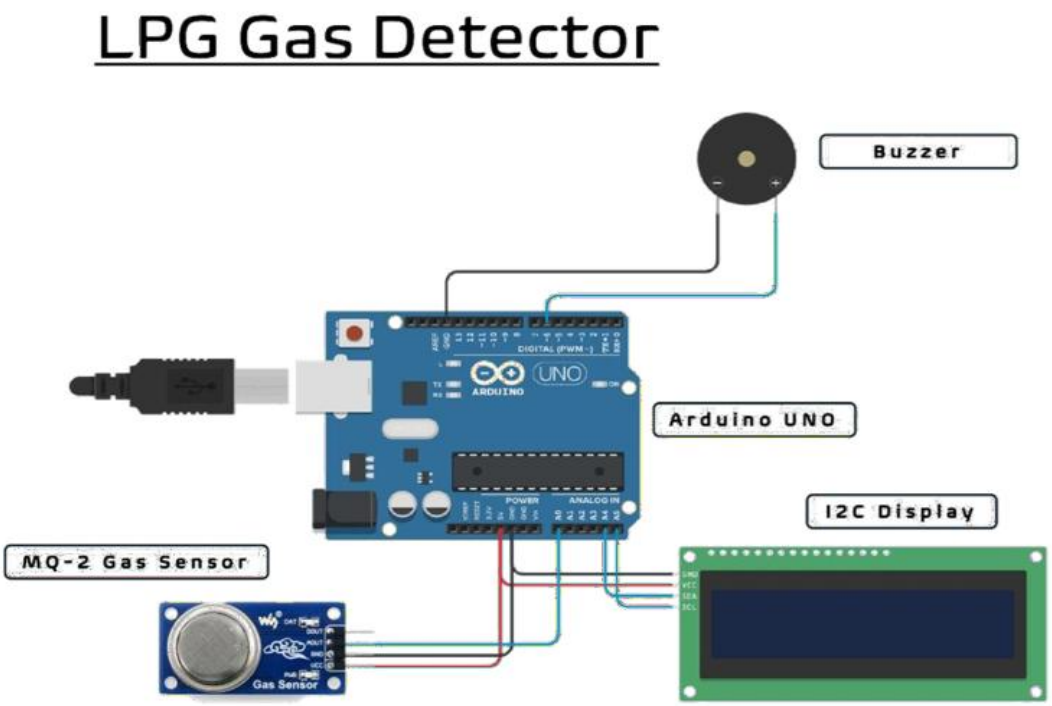
SAFE → No gas | ı CAUTION → Some gas detected | DANGER → High gas level

c. Buzzer behaviour:

SAFE → OFF | CAUTION → Beep | DANGER → Continuous ON

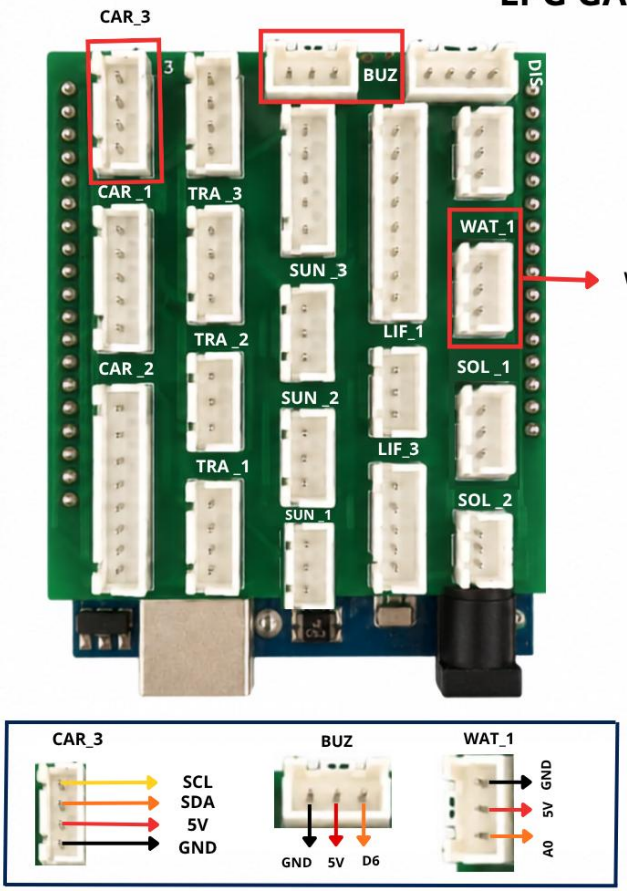
- **Connections**

Component / Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
1: SDA	2: SDA	CAR_3	LCD Display
2: SCL	1: SCL		
3: +5V	3: +5V		
4: GND	4: GND		
1: GND	1: GND	WAT_1	MQ2 Gas Sensor
2: VCC	2: 5V		
3: AOUT	3: A0		
1: GND (Black)	1: GND	BUZ	Buzzer
2: Signal	2: D6		



Connection Diagram

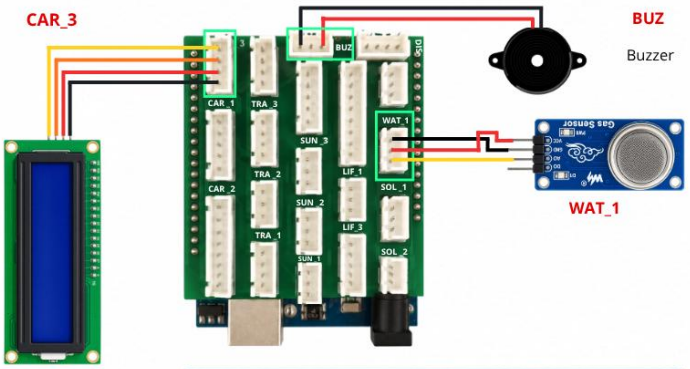
LPG GAS DETECTOR



COMPONENTS REQUIRED



CONNECTION DIAGRAM



I2C LCD Display

SHIELD CONNECTION SUMMARY

Component	Shield Port	Description
MQ-2 Sensor, Buzzer	WAT_1, BUZ	Detects LPG, smoke and other gases, Gives an alert when gas is detected
I2C LCD Display	CAR_3	Displays gas detection status

Program

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

LiquidCrystal_I2C lcd(0x27, 16,
2);

int mq2flin = A0;
int buzzer = 6;

int sensorValue;
int ppm;

void setup() {
pinMode(buzzer, OUTflUT);

// LCD init
lcd.init();
lcd.backlight();

lcd.setCursor(0,0);
lcd.print("Gas Leak Meter");
delay(2000);
lcd.clear();

// Serial init
Serial.begin(9600);
   Serial.println("MQ2 Gas Sensor
Starting...");
}

void loop() {
   sensorValue =
analogRead(mq2flin);

// Approximate mapping
ppm = map(sensorValue, 0, 1023,
0, 3000);

// -------- LCD DISflLAY -------
-
lcd.setCursor(0,0);
lcd.print("Gas:");
```

```
lcd.print(ppm);
lcd.print(" ppm          ");

lcd.setCursor(0,1);

String zone;

if (ppm < 1600) {
zone = "SAFE";
lcd.print("SAFE ZONE              ");
digitalWrite(buzzer, LOW);
}
else if (ppm < 2000) {
zone = "CAUTION";
lcd.print("CAUTION ZONE           ");

digitalWrite(buzzer, HIGH);
delay(150);
digitalWrite(buzzer, LOW);
}
else {
zone = "DANGER";
lcd.print("DANGER ZONE!! ");
digitalWrite(buzzer, HIGH);
}
```

```
// -------- SERIAL MONITOR ----
----
Serial.print("Raw: ");
Serial.print(sensorValue);

Serial.print(" | flflM: ");
Serial.print(ppm);

Serial.print(" | Zone: ");
Serial.println(zone);

// -------- SERIAL flLOTTER
(optional) --------
   // Uncomment below if you want
graph in Arduino fllotter
/*
Serial.print(ppm);
Serial.print(",");
Serial.println(sensorValue);
*/

delay(500);
}
```

9. Smart Lock

- ## What you Will Learn!

a. How a **keypad is used for password input**
b. How Arduino performs **security logic**
c. How to control a **servo motor (lock/unlock)**
d. How to display messages on **LCD (I2C)**
e. Basics of **authentication & smart security systems**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	0
6	4Pin JST Connector	1
7	5V Servo Motor with Horn	1
8	4X4 Keypad with Connector	1
9	I2C Display Module	1

- ## Steps to make it Work!

a. Connect **keypad** to CAR-1, CAR-2
b. Connect **servo motor** to LIF-1
c. Connect **LCD display** to CAR-3
d. Connect Arduino to laptop
e. Open Arduino IDE
f. Upload the code
g. Enter password using keypad

- ## What should Happen!

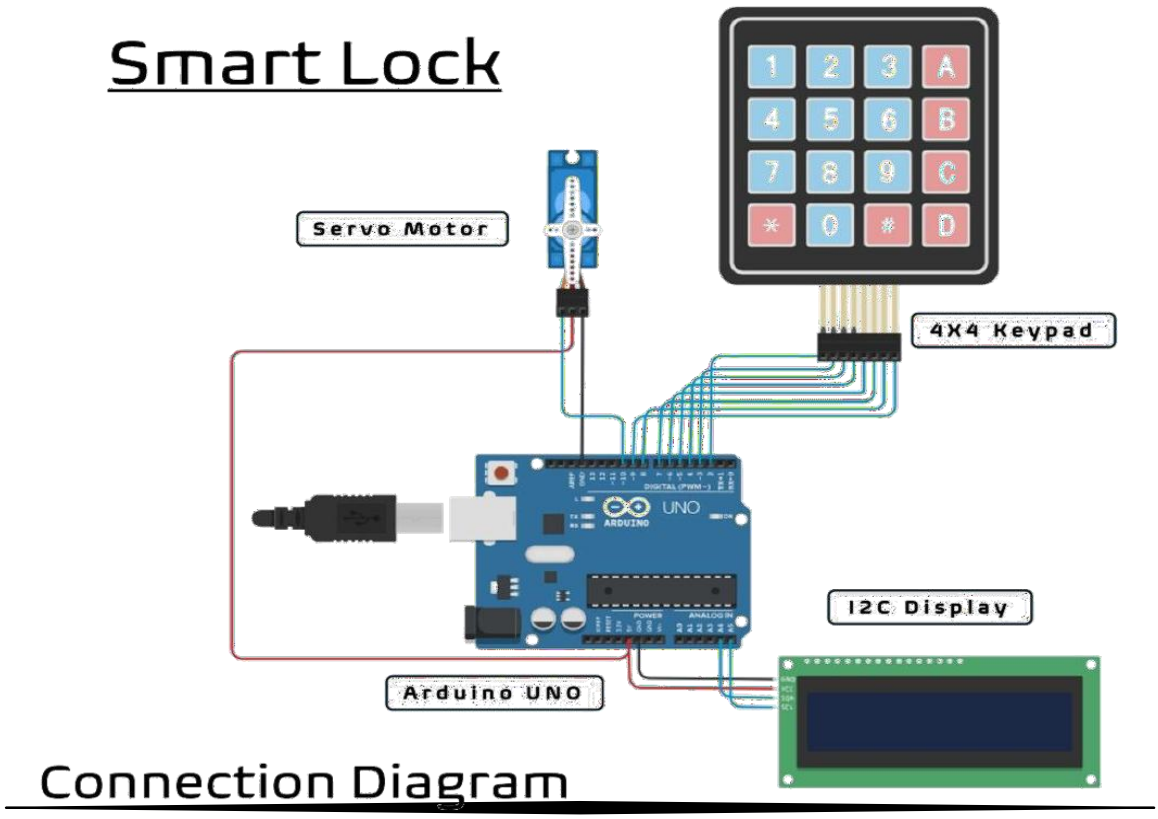
a. Enter correct PIN → **Door Unlocks (Servo rotates)**
   - Wrong PIN → **Access Denied**
   - After 3 wrong attempts → **System Jammed**
   - Press **A** → Enter master password

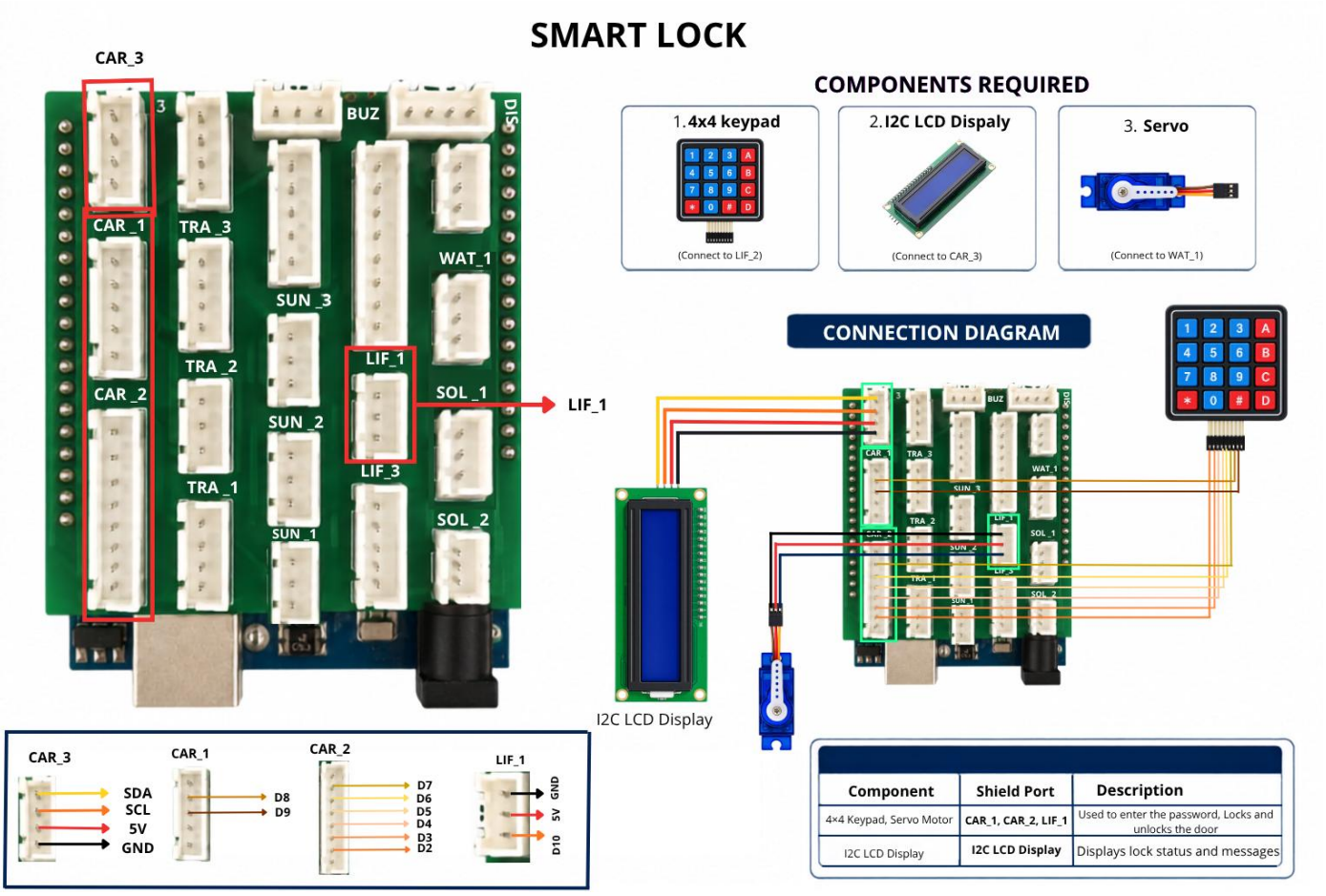
b. Master options:

- D → Unlock
- B → Set new password

- **Connections**

Component / Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
1: SDA	2: SDA	CAR_3	LCD Display
2: SCL	1: SCL		
3: +5V	3: +5V		
4: GND	4: GND		
1: R1	7: D2	Car_2	Keyboard
3: R2	6: D3		
2: R3	5: D4		
4: R4	4: D5		
5: C1	3: D6		
6: C2	2: D7		
7: C3	2: D8	Car_1	Keyboard
8: C4	3: D9		
1: GND	1: GND	LIF_1	Servo Motor
2: VCC	2: +5V		
3: Signal	3: D10		





Program

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

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

const byte ROWS = 4;
const byte COLS = 4;

char keys[ROWS][COLS] = {
{'1','2','3','A'},
{'4','5','6','B'},
{'7','8','9','C'},
{'*','0','#','D'}
};

byte rowflins[ROWS] = {2, 3, 4, 5};
byte colflins[COLS] = {6, 7, 8, 9};

Keypad keypad = Keypad(makeKeymap(keys), rowflins, colflins, ROWS, COLS);

String userflassword  = "1234";
String masterflassword = "9999";
String input = "";

int attempts = 0;
const int maxAttempts = 3;

bool jammed = false;
bool enteringMasterflin = false;
bool masterVerified = false;
```

```
bool setflasswordMode = false;

void setup() {
lcd.init();
lcd.backlight();

lockServo.attach(10);
lockServo.write(0);

lcd.print(" Smart Lock ");
delay(2000);
lcd.clear();
}

void loop() {
lcd.setCursor(0,0);

if (jammed fifi
!enteringMasterflin  fifi
!masterVerified) {
 lcd.print("SYSTEM JAMMED");
lcd.setCursor(0,1);
lcd.print("flress A");
}
else if (enteringMasterflin) {
lcd.print("Enter Master:");
}
else if (masterVerified fifi
!setflasswordMode) {
lcd.print("D:Unlock B:Set");
}
else if (setflasswordMode) {
  lcd.print("New  flassword:");
}
else {
lcd.print("Enter flN:");
}

char key = keypad.getKey();

if (key) {

if (key == '*') {
      input = "";
      lcd.clear();
}

// 🔑 flRESS 6 K 6Se M6SHER
flN IMMEDIATELY
    else if (key == 'A' fifi
jammed) {
      enteringMasterflin = true;
      input = "";
      lcd.clear();
      lcd.print("Enter Master:");
    }

    // CHECK flN
    else if (key == 'C') {
      checkflassword();
    }

    // DEJAM / UNLOCK
    else if (key == 'D' fifi
masterVerified) {
      lcd.clear();
      lcd.print("System
Unlocked");
      unlockDoor();
      resetSystem();
    }

    // SET NEW flASSWORD
    else if (key == 'B' fifi
masterVerified) {
      setflasswordMode = true;
      input = "";
      lcd.clear();
    }

    // DIGIT ENTRY
    else {
      input += key;
      lcd.setCursor(input.length(
) - 1, 1);
      lcd.print(key);
    }
  }
}

void checkflassword() {
  lcd.clear();

  // MASTER flN CHECK
  if (enteringMasterflin fifi input
== masterflassword) {
    lcd.print("MASTER VERIFIED");
    masterVerified = true;
    enteringMasterflin = false;
```

```
      attempts = 0;
      delay(1500);
    }

    // USER flIN CHECK
    else if (!jammed fifi input == userflassword) {
      lcd.print("Access Granted");
      unlockDoor();
      attempts = 0;
    }

    // SET NEW flASSWORD
    else if (setflasswordMode) {
      userflassword = input;
      lcd.print("flIN Updated");
      delay(2000);
      resetSystem();
      return;
    }

    // WRONG flIN
    else {
      attempts++;
      int remaining = maxAttempts - attempts;

      lcd.setCursor(0,0);
      lcd.print("Failed: ");
      lcd.print(attempts);
      lcd.print("/");
      lcd.print(maxAttempts);

      lcd.setCursor(0,1);
      lcd.print("Left: ");
      lcd.print(remaining);

      delay(2000);

      if (attempts >= maxAttempts) {
        jammed = true;
      }
    }

    input = "";
    lcd.clear();
}

void unlockDoor() {
    lockServo.write(90);
    delay(3000);
    lockServo.write(0);
}

void resetSystem() {
    jammed = false;
    enteringMasterflin = false;
    masterVerified = false;
    setflasswordMode = false;
    attempts = 0;
    input = "";
    lcd.clear();
}
```

10. Air Quality Meter

- ## What you Will Learn!

a. How an **MQ-135 sensor detects air quality**
b. How Arduino reads **analog values**
c. How to convert readings into **PPM (pollution level)**
d. How to display data on **LCD (I2C)**
e. Basics of **environment monitoring systems**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	1
6	4Pin JST Connector	1
7	MQ-135 Gas Sensor	1
8	I2C Display Module	1

- ## Steps to make it Work!

a. Connect **MQ-135 sensor** to WAT-1
b. Connect **LCD display** to CAR-3
c. Connect Arduino to laptop
d. Open Arduino IDE
e. Upload the code
f. Observe air quality changes

- ## What should Happen!

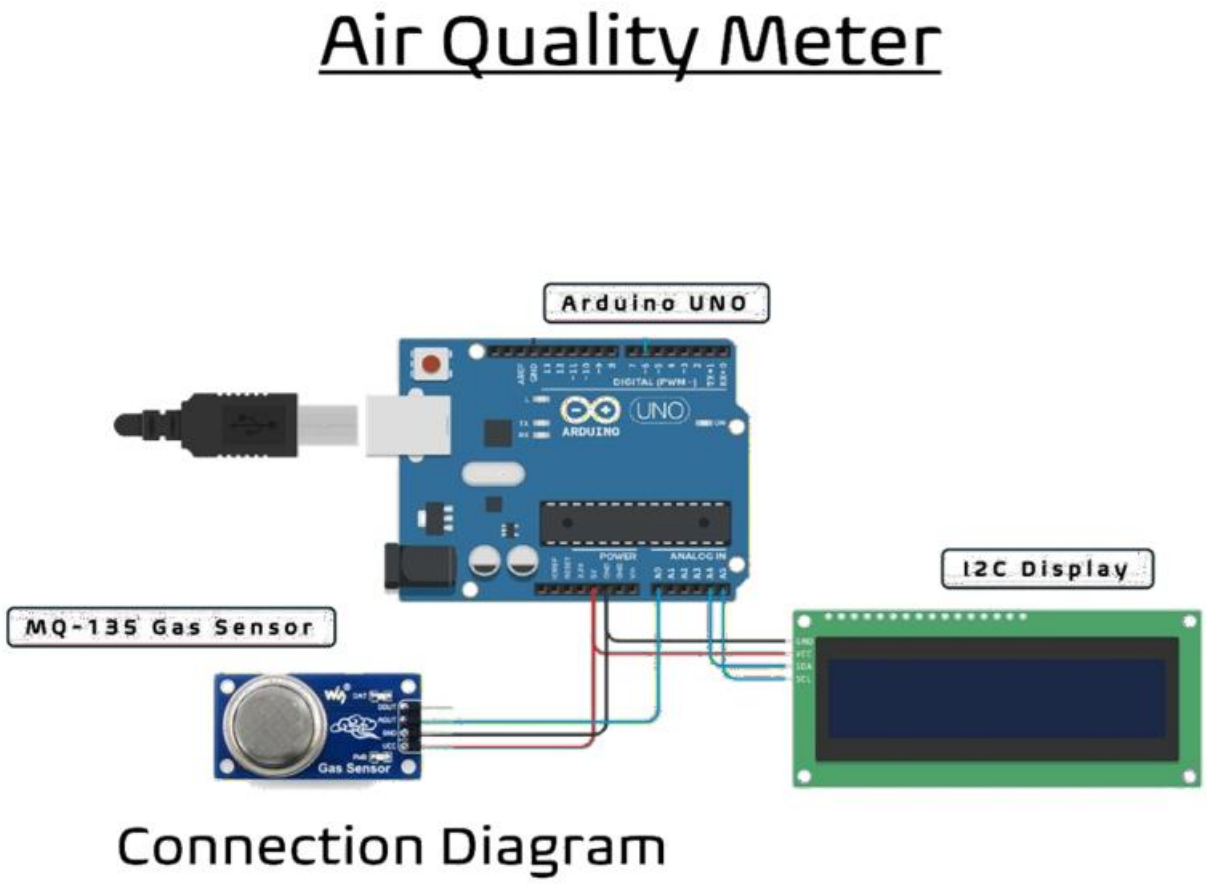
a. LCD shows **PPM value (air quality)**
b. Air conditions:

Good Air | Less Polluted | Not Fit to Breathe | Dangerous

c. Higher pollution → higher PPM value

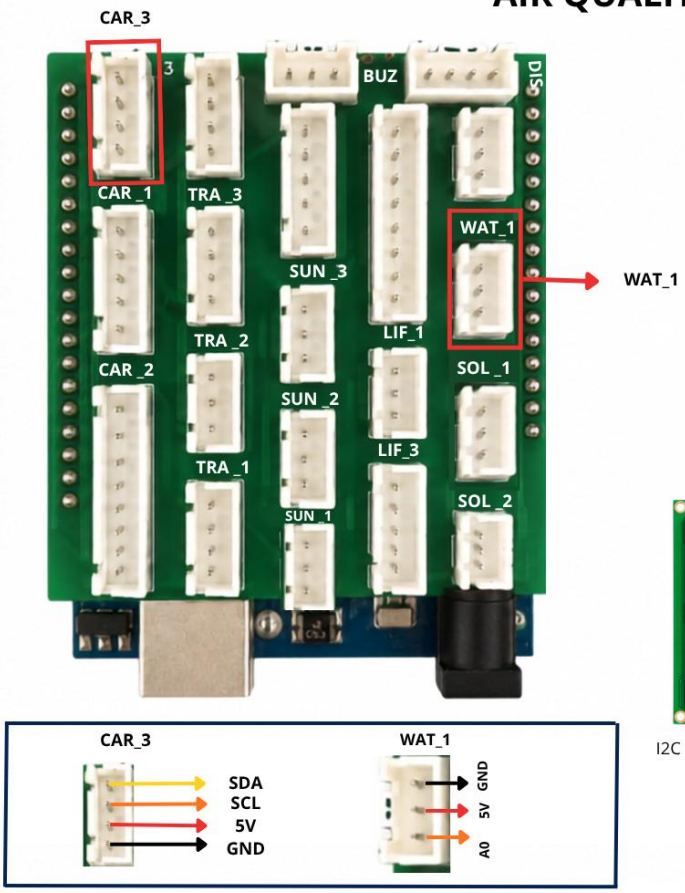
- **Connections**

Component / Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
1: SDA	2: SDA	CAR_3	LCD Display
2: SCL	1: SCL		
3: +5V	3: +5V		
4: GND	4: GND		
1: GND	1: GND	WAT_1	MQ-135 Gas Sensor (Waveshare)
2: VCC	2: +5V		
3: AOUT	3: A0		



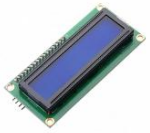
Connection Diagram

AIR QUALITY METER



COMPONENTS REQUIRED

1. I2C LCD Dispaly



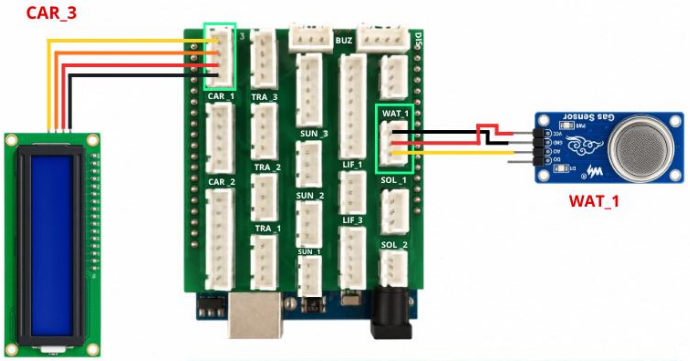
(Connect to CAR_3)

2. MQ-135 Gas Sensor



(Connect to WAT_1)

CONNECTION DIAGRAM



I2C LCD Display

SHIELD CONNECTION SUMMARY

Component	Shield Port	Description
MQ-135 Sensor	WAT_1	Detects air pollutants and poor air quality
I2C LCD Display	CAR_3	Displays air quality readings/status

Program

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

LiquidCrystal_I2C lcd(0x27, 16,
2);

int mq135flin = A0;
int sensorValue = 0;
int ppm = 0;

void setup() {
lcd.init();
lcd.backlight();

lcd.setCursor(0,0);
lcd.print("Air Quality");
delay(2000);
lcd.clear();
}

void loop() {
    sensorValue =
analogRead(mq135flin);

// Map MQ135 value to 0-200 flflM
ppm = map(sensorValue, 0, 1023,
        0, 200);

lcd.clear();

// Line 1: flflM value
lcd.setCursor(0,0);
lcd.print("flflM: ");
lcd.print(ppm);

// Line 2: Warning message
lcd.setCursor(0,1);

if (ppm <= 70) {
lcd.print("Good Air");
}
else if (ppm <= 110) {
lcd.print("Less flolluted");
}
else if (ppm <= 170) {
lcd.print("Not Fit Breathe");
}
else {
```

```
lcd.print("DANGEROUS!");
}
```

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

11. Temperature & Humidity Monitoring

- ## What you Will Learn!

a. How a **DHT11 sensor measures temperature & humidity**
b. How Arduino reads **digital sensor data**
c. How to display readings on **LCD (I2C)**
d. Basics of **environment monitoring systems**
e. Real-life use in **weather & smart home systems**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	1
6	4Pin JST Connector	1
7	Humidity and temperature Sensor	1
8	I2C Display Module	1

- ## Steps to make it Work!

a. Connect **DHT11 sensor** to SUN-2
b. Connect **LCD display** to CAR-3
c. Connect Arduino to laptop
d. Open Arduino IDE
e. Upload the code
f. Observe temperature & humidity \

- ## What should Happen!

a. LCD shows:

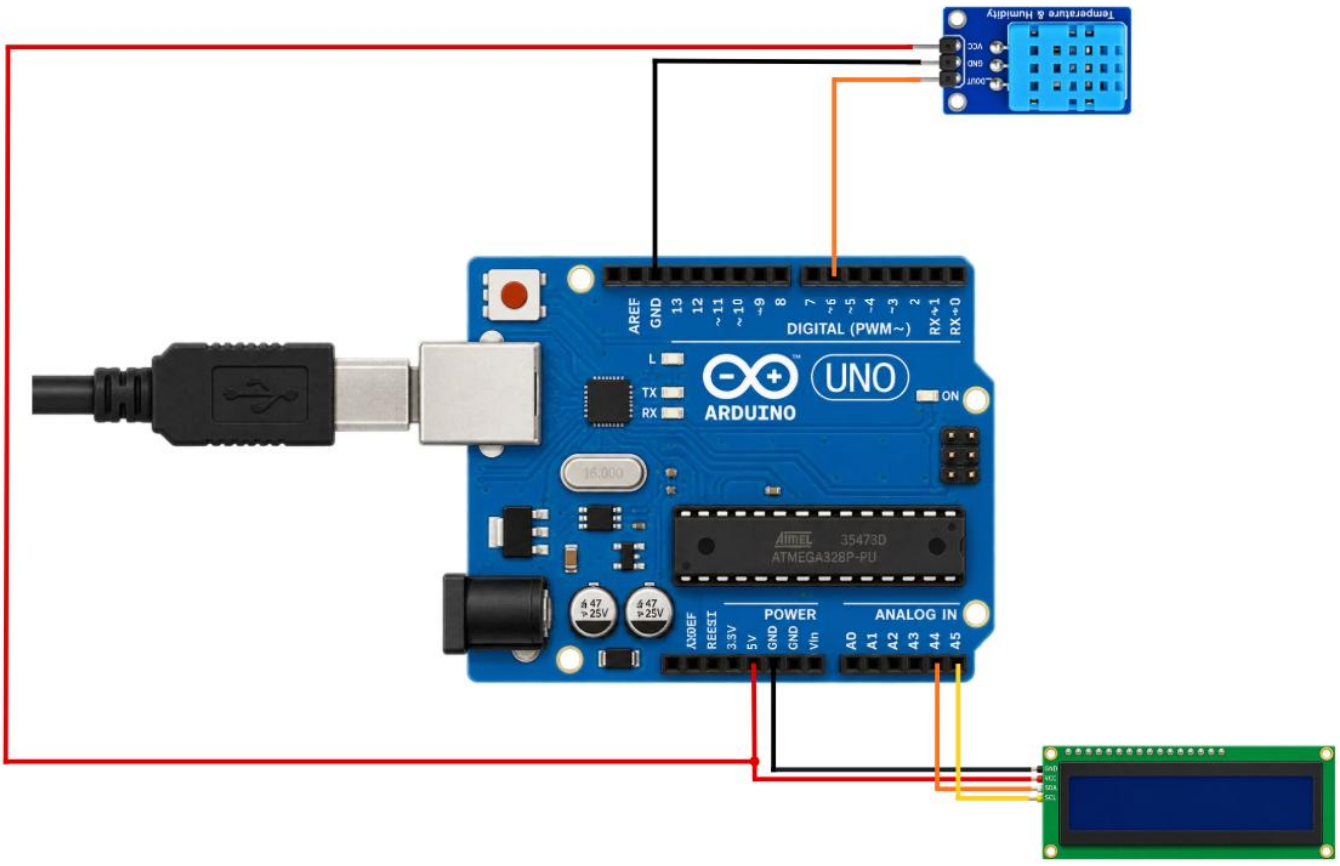
- \ **Temperature (°C)**

- **Humidity (%)**

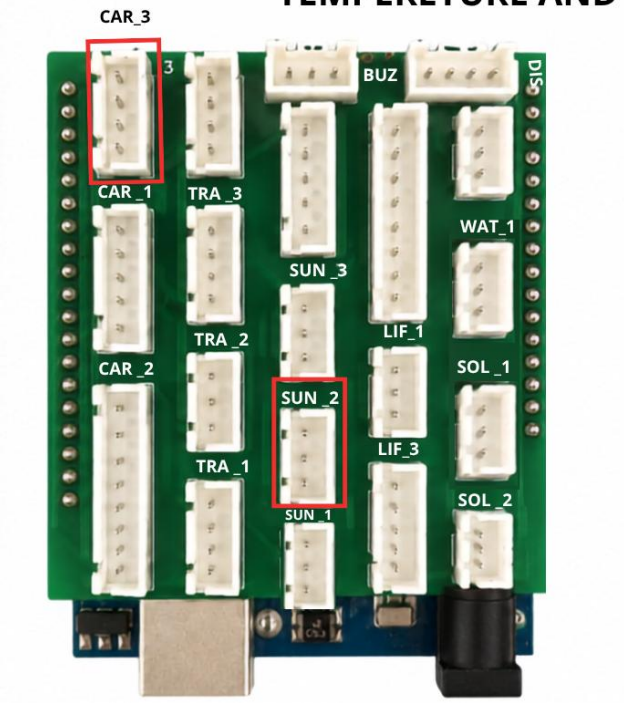
b. Values update every **2 seconds**
c. If sensor fails → shows **"Sensor Error"**

• Connections

Component / Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
1: SDA	2: SDA	CAR_3	LCD Display
2: SCL	1: SCL		
3: +5V	3: +5V		
4: GND	4: GND		
1: GND	1: GND	SUN_2	DHT 11 Humidity and Temperature Sensor
2: VCC	2: +5V		
3: S	3: D6		



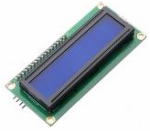
TEMPERETURE AND HUMIDITY MONITORING





COMPONENTS REQUIRED

2. I2C LCD Dispaly



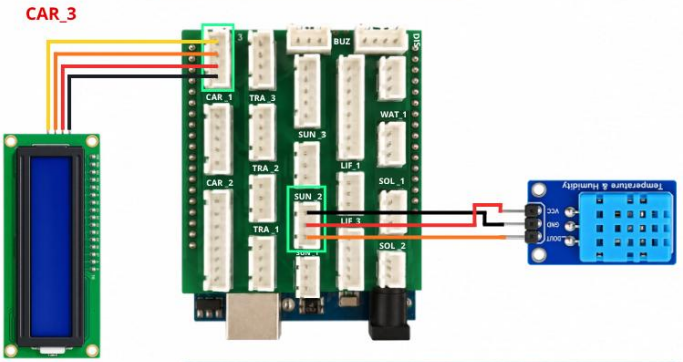
(Connect to CAR_3)

3. Humidity Sensor



(Connect to WAT_1)

CONNECTION DIAGRAM



I2C LCD Display

SHIELD CONNECTION SUMMARY

Component	Shield Port	Description
Temperature & Humidity Sensor	SUN_2	Measures temperature and humidity
I2C LCD Display	CAR_3	Displays temperature and humidity readings

Program

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

#define DHTPIN 6          // DHT11 DATA pin
#define DHTTYPE DHT11     // Sensor type

DHT dht(DHTPIN, DHTTYPE);
LiquidCrystal_I2C lcd(0x27, 16, 2);  // Change to 0x3F if needed

void setup() {
lcd.init();
lcd.backlight();
dht.begin();

lcd.setCursor(0, 0);
lcd.print("Temp fi Humidity");
delay(2000);
lcd.clear();
}

void loop() {
   float humidity = dht.readHumidity();
   float temperature = dht.readTemperature(); // Celsius

   // Check if sensor reading failed
```

```
  if (isnan(humidity) || isnan(temperature)) {
    lcd.setCursor(0, 0);
    lcd.print("Sensor Error   ");
    delay(1000);
    return;
  }

  lcd.setCursor(0, 0);
  lcd.print("Temp: ");
  lcd.print(temperature);
  lcd.print(" C   ");

  lcd.setCursor(0, 1);
  lcd.print("Hum : ");
  lcd.print(humidity);
  lcd.print(" %   ");

  delay(2000);    // Update every 2 seconds
}
```

12. Thermometer

- ## What you Will Learn!

a. How a **DS18B20 temperature sensor works**
b. How Arduino reads data using **OneWire communication**
c. How to display values on **LCD (I2C)**
d. How to convert **Celsius to Fahrenheit**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	1
6	4Pin JST Connector	1
7	Temperature Sensor	1
8	I2C Display Module	1
9	LPG Lighter (Only with Trainer)	1

- ## Steps to make it Work!

a. Connect **temperature sensor (DS18B20)** to WAT-1
b. Connect **LCD display** to CAR-3
c. Connect Arduino to laptop
d. Open Arduino IDE
e. Upload the code
f. Observe temperature readings \

- ## What should Happen!

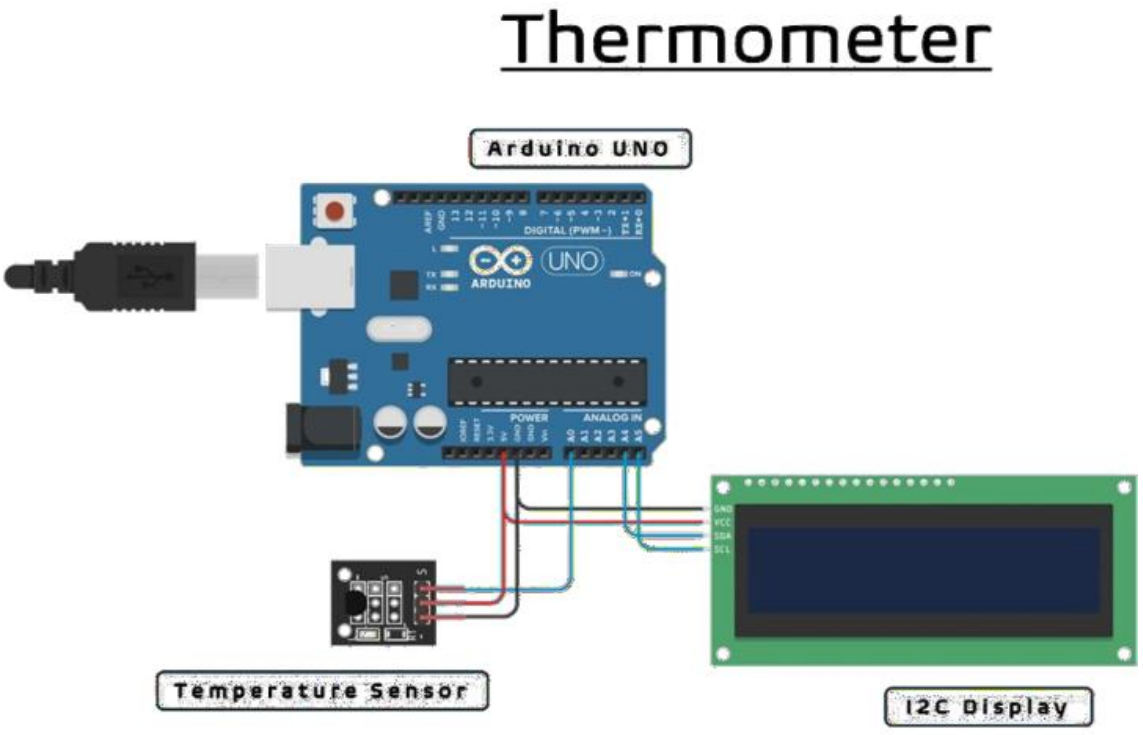
a. CD shows:

- \Temperature in **Celsius (°C)**
- \ Temperature in **Fahrenheit (°F)**

b. Values update every **1 second**
c. Accurate digital temperature measurement

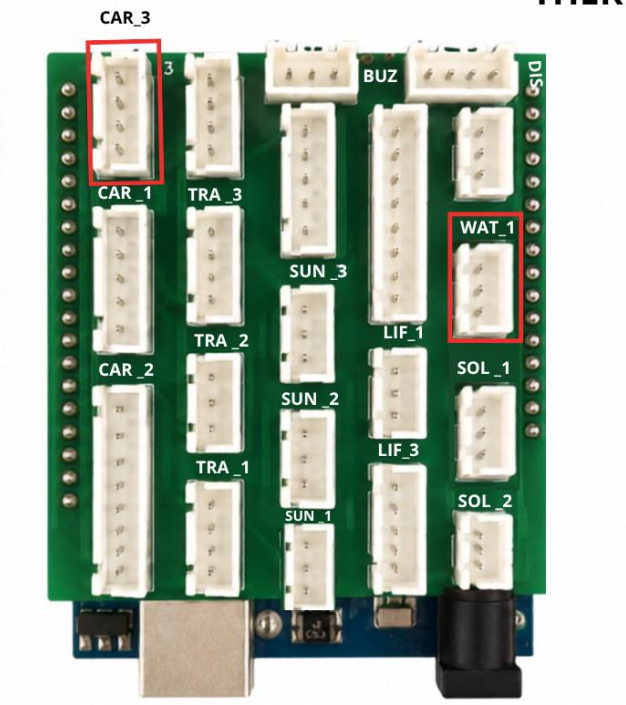
- **Connections**

Component / Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
1: SDA	2: SDA	CAR_3	LCD Display
2: SCL	1: SCL		
3: +5V	3: +5V		
4: GND	4: GND		
1: GND	1: GND	WAT_1	DS18B20 Temperature Sensor
2: VCC	2: +5V		
3: S	3: A0		



Connection Diagram

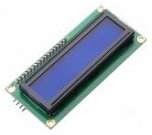
THERMOMETER





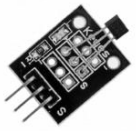
COMPONENTS REQUIRED

2. I2C LCD Dispaly



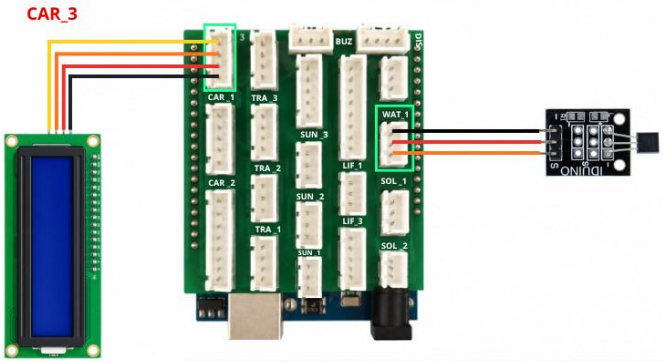
(Connect to CAR_3)

3. Temperature Sensor



(Connect to WAT_1)

CONNECTION DIAGRAM



SHIELD CONNECTION SUMMARY

Component	Shield Port	Description
Temperature Sensor	WAT_1	Measures temperature
I2C LCD Display	CAR_3	Displays temperature readings

Program

```
#include <OneWire.h>
#include <DallasTemperature.h>
#include <LiquidCrystal_I2C.h>

#define ONE_WIRE_BUS A0        //
DS18B20 DATA on A0

OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature
sensors(fioneWire);

LiquidCrystal_I2C lcd(0x27, 16,
2);

void setup() {
sensors.begin();
   sensors.setResolution(10);       //
Stable fi faster conversion

lcd.init();
lcd.backlight();
lcd.setCursor(0, 0);
lcd.print("Thermometer");
delay(2000);
lcd.clear();
}

void loop() {
sensors.requestTemperatures();
   float tempC =
sensors.getTempCByIndex(0);
float tempF = tempC * 9.0 / 5.0
+ 32.0;        // Convert to
Fahrenheit

lcd.clear();

// Line 1 - Celsius
lcd.setCursor(0, 0);
lcd.print("C: ");
   lcd.print(tempC, 1);       // 1
decimal place
```

```
    lcd.print((char)223);       //
Degree symbol
lcd.print("C");

// Line 2 - Fahrenheit
lcd.setCursor(0, 1);
lcd.print("F: ");
    lcd.print(tempF, 1);       // 1
decimal place
lcd.print((char)223);
lcd.print("F");

delay(1000);
```

13. Green House

- ## What you Will Learn!

a. How to monitor **environment conditions in agriculture**
b. How **gas sensors detect air quality**
c. How **soil moisture sensors detect water level**
d. How Arduino combines **multiple sensors**
e. Basics of **smart farming systems**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	2
6	4Pin JST Connector	1
7	Soil Moisture Sensor with Module	1
8	MQ-135 Gas Sensor	1
9	I2C Display Module	1

- ## Steps to make it Work!

a. Connect **gas sensor** to WAT-1
b. Connect **soil moisture sensor** to WAT-2
c. Connect **LCD display** to CAR-3
d. Connect Arduino to laptop
e. Open Arduino IDE
f. Upload the code
g. Insert soil sensor into soil

- ## What should Happen!

a. CD shows:

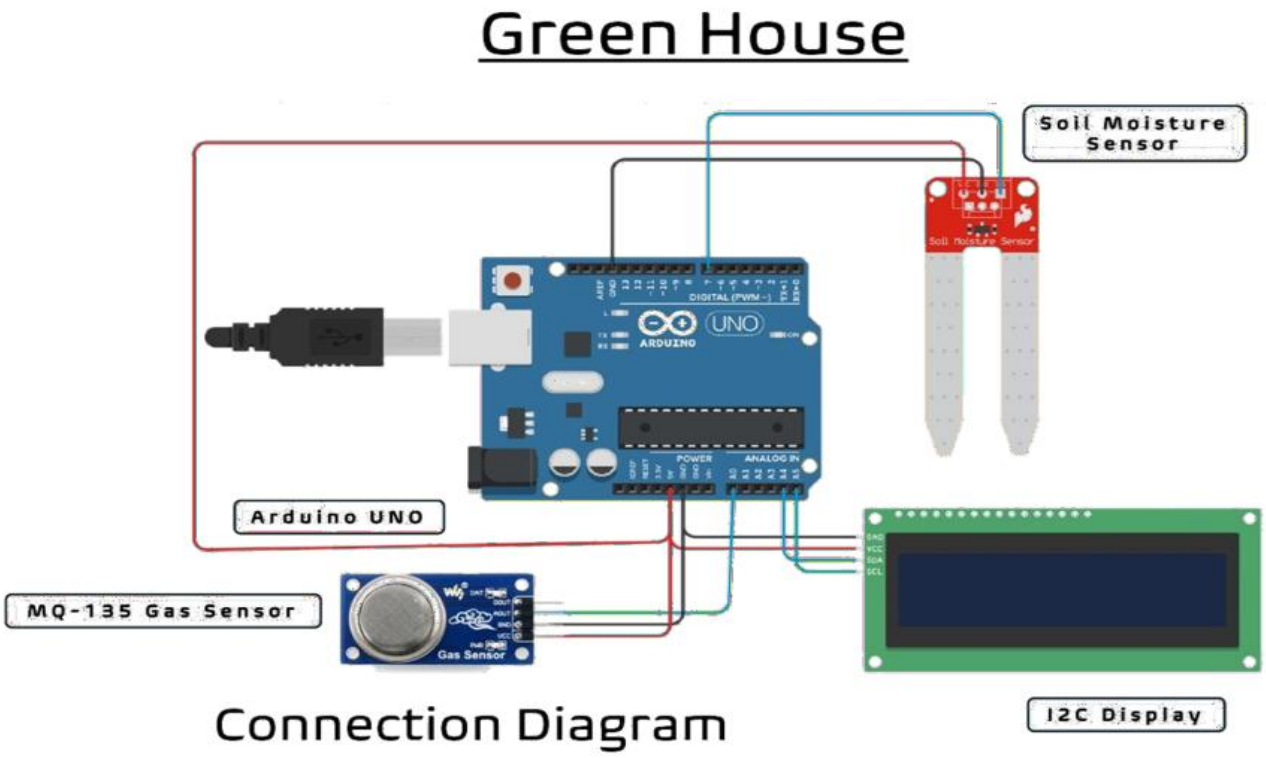
Gas level value | Soil condition

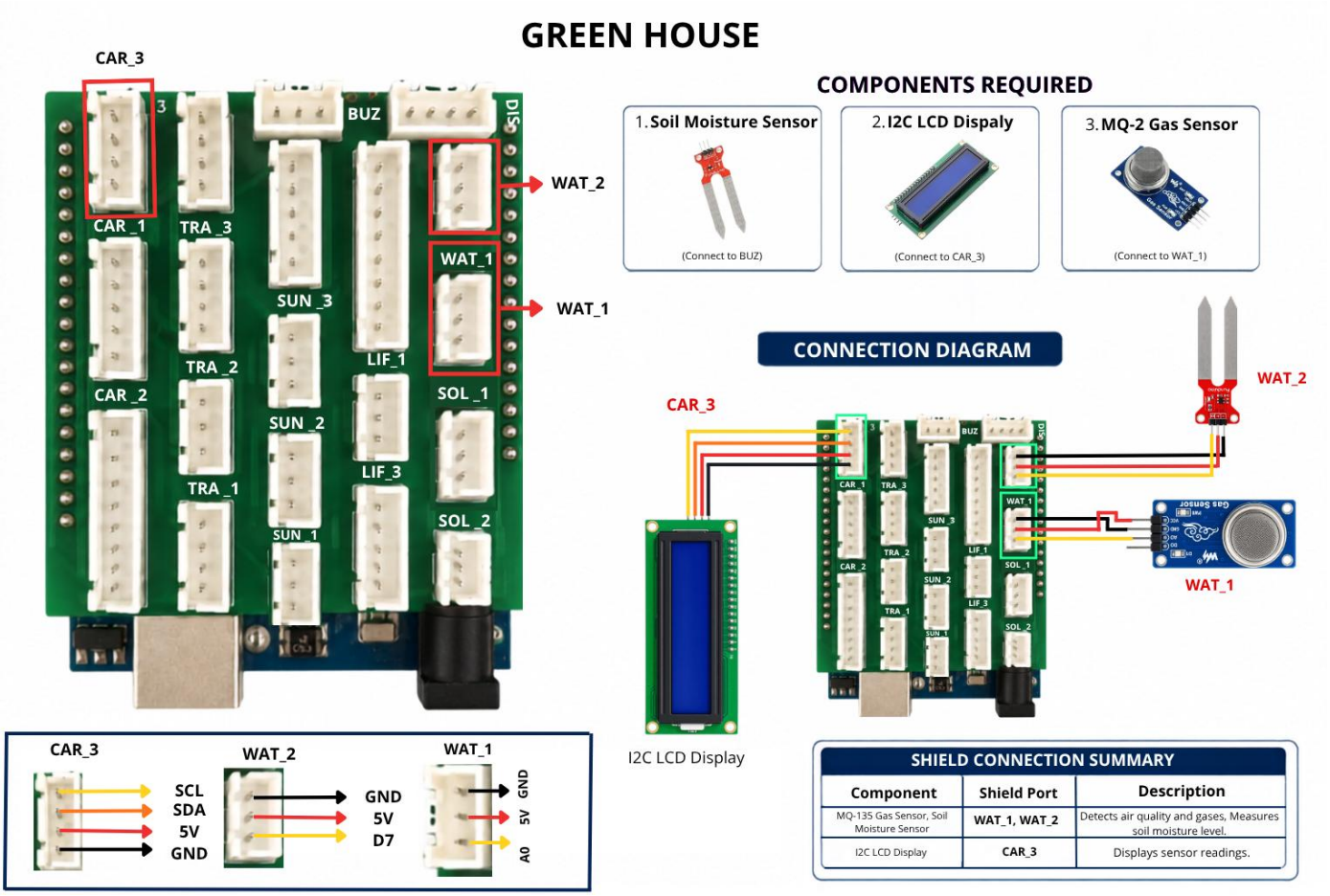
b. Soil condition:

DRY → Needs water | WET → Enough moisture

- **Connections**

Component / Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
1: SDA	2: SDA	CAR_3	LCD Display
2: SCL	1: SCL		
3: +5V	3: +5V		
4: GND	4: GND		
1: GND	1: GND	WAT_1	MQ-135 Gas Sensor (Waveshare)
2: VCC	2: +5V		
3: AOUT	3: A0		
1: GND	1: GND	WAT_2	HS-S33A, Soil Moisture Sensor
2: V	2: +5V		
3: S	3: D7		





Program

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

LiquidCrystal_I2C lcd(0x27,16,2);

int gasSensor = A0;
int soilSensor = 7;

void setup()
{
lcd.init();
lcd.backlight();

pinMode(soilSensor, INflUT);

lcd.setCursor(0,0);
lcd.print("Green House");
lcd.setCursor(0,1);
lcd.print("System Start");
delay(2000);
lcd.clear();
}

void loop()
{

int gasValue =
analogRead(gasSensor);
int soilValue =
digitalRead(soilSensor
);
lcd.setCursor(0,0);
lcd.print("Gas:");
lcd.print(gasValue);
lcd.print("    ");

lcd.setCursor(0,1);

if(soilValue == HIGH)
{
```

```
lcd.print("Soil: DRY ");                    }
}
else                                        delay(1000);
{
lcd.print("Soil: WET ");                    }
```

14. Smart Roof

- ## What you Will Learn!

a. How a **rain sensor detects water**
b. How Arduino reads **analog signals**
c. How to control a **servo motor automatically**
d. Basics of **weather-based automation**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	0
6	4Pin JST Connector	1
7	Rain Drop Sensor with Module	1
8	Servo Motor with Horn	1
9	Glass with Water (Put in Drops)	1

- ## Steps to make it Work!

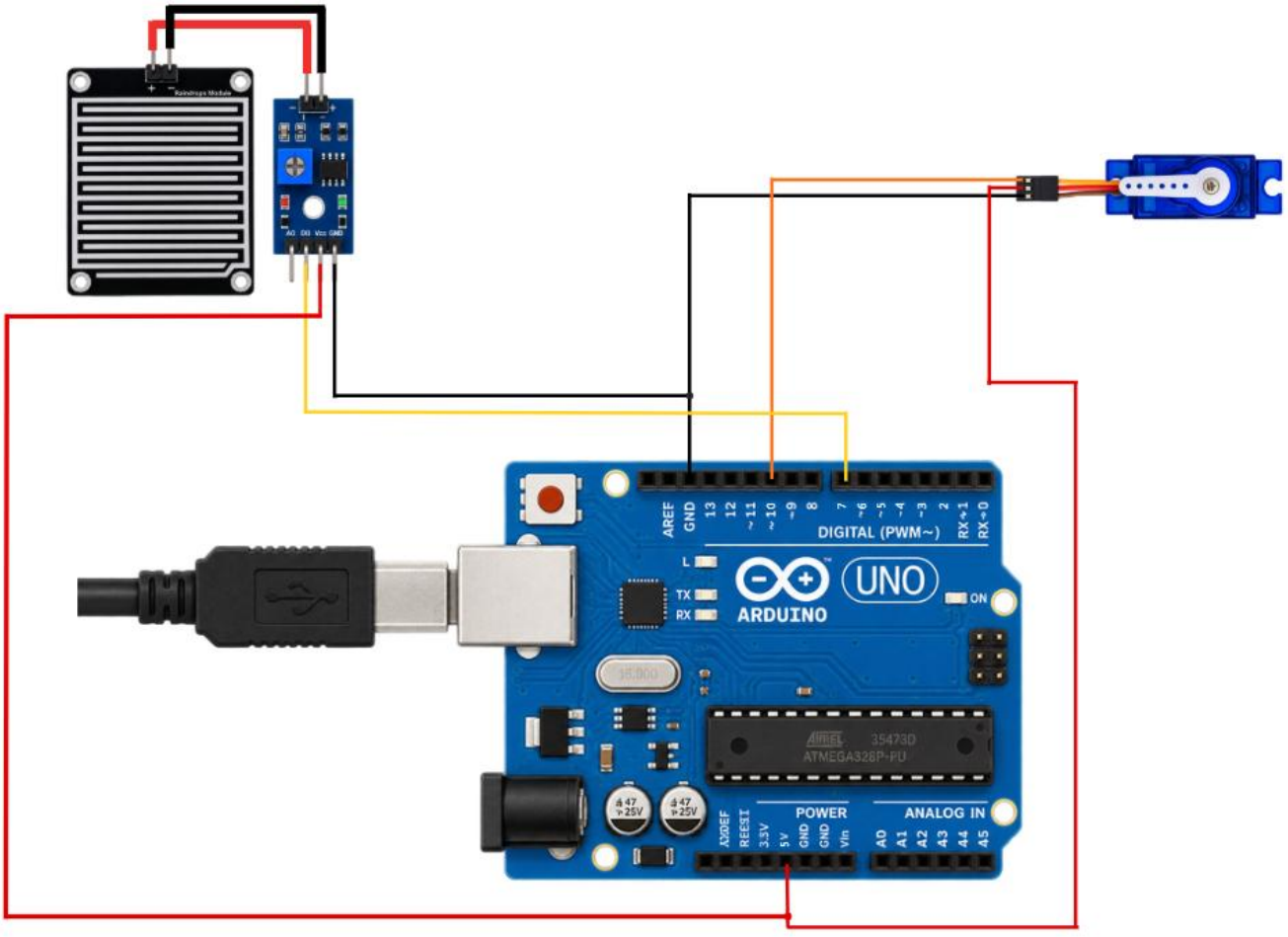
a. Connect **rain sensor** to WAT-2
b. Connect **servo motor** to LIF-1
c. Connect Arduino to laptop
d. Open Arduino IDE
e. Upload the code
f. Sprinkle water on sensor 

- ## What should Happen!

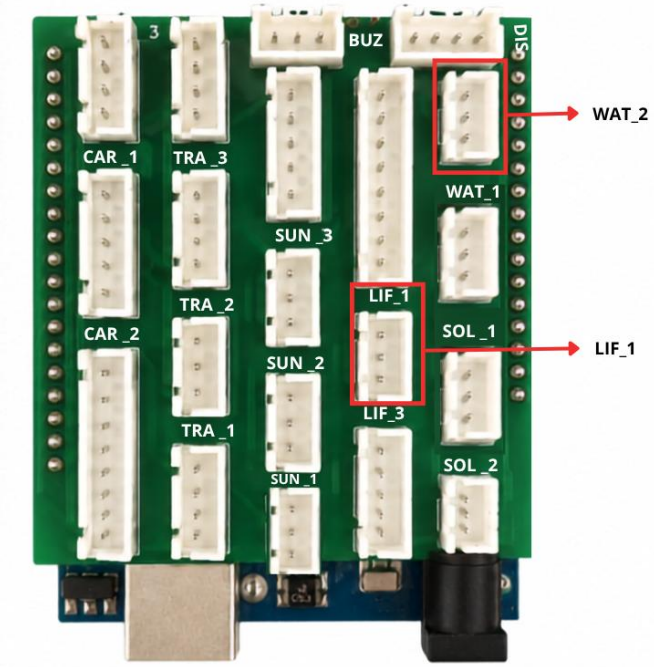
- When **rain is detected**: Roof **closes automatically** 
- When **no rain**: Roof **opens automatically** 
- Serial Monitor shows sensor values

- **Connections**

Component / Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
1: GND	1: GND	WAT_2	LDR Module
2: VCC	2: +5V		
3: D0	3: D7		
1: GND	1: GND	LIF_1	Servo Motor
2: VCC	2: +5V		
3: Signal	3: D10		



SMART ROOF





COMPONENTS REQUIRED

1. **Servo Motor**



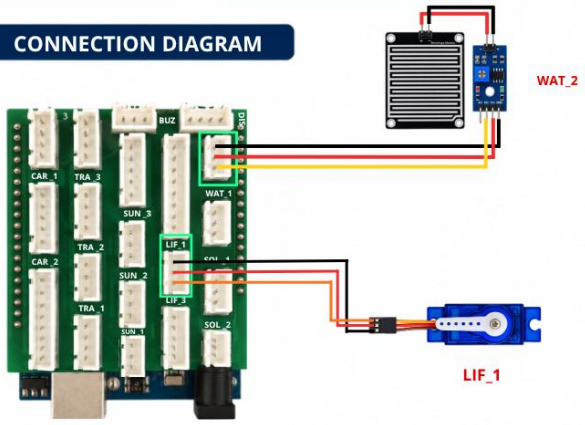
(Connect to LIF_1)

2. **Temperature Sensor**



(Connect to WAT_1)

CONNECTION DIAGRAM



SHIELD CONNECTION SUMMARY

Component	Shield Port	Description
Rain Drop Sensor	WAT_1	Detects rain/water.
Servo Motor	LIF_1	Opens/closes the roof.

Program

```
#include <Servo.h>

#define RAIN_flIN 7
#define SERVO_flIN 10

Servo roofServo;

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

roofServo.attach(SERVO_flIN);
   roofServo.write(0);        // Roof
OflEN at start
}

void loop() {
   int rainValue = digitalRead(RAIN
_flIN);                        // 0-1023

   Serial.print("Rain Sensor
Value: ");
Serial.println(rainValue);

if (rainValue < 400)
       {       // Rain detected
roofServo.write(90);                    /
       / CLOSE roof
Serial.println("Status: RAIN
            Roof CLOSED");
}
else
       {                        // No
       rain
roofServo.write(0);                     /
       / OflEN roof
   Serial.println("Status: NO RAI
N - Roof OflEN");
}

delay(1000);
}
```

15. Automatic Toll Gate

- ## What you Will Learn!

a. How **IR sensors detect vehicles**
b. How Arduino controls a **servo motor (gate)**
c. How to automate **entry and exit systems**
d. How to display messages on **LCD (I2C)**
e. Basics of **smart traffic systems**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	2
6	4Pin JST Connector	1
7	IR Module	2
8	Servo Motor with Horn	1
9	I2C Display Module	1

- ## Steps to make it Work!

a. Connect **Entry IR sensor** to SUN-1
b. Connect **Exit IR sensor** to SUN-2
c. Connect **servo motor** to LIF-1
d. Connect **LCD display** to CAR-3
e. Connect Arduino to laptop
f. Open Arduino IDE
g. Upload the code
h. Move object/vehicle near sensors ;

- ## What should Happen!

a. When vehicle reaches **entry sensor**:

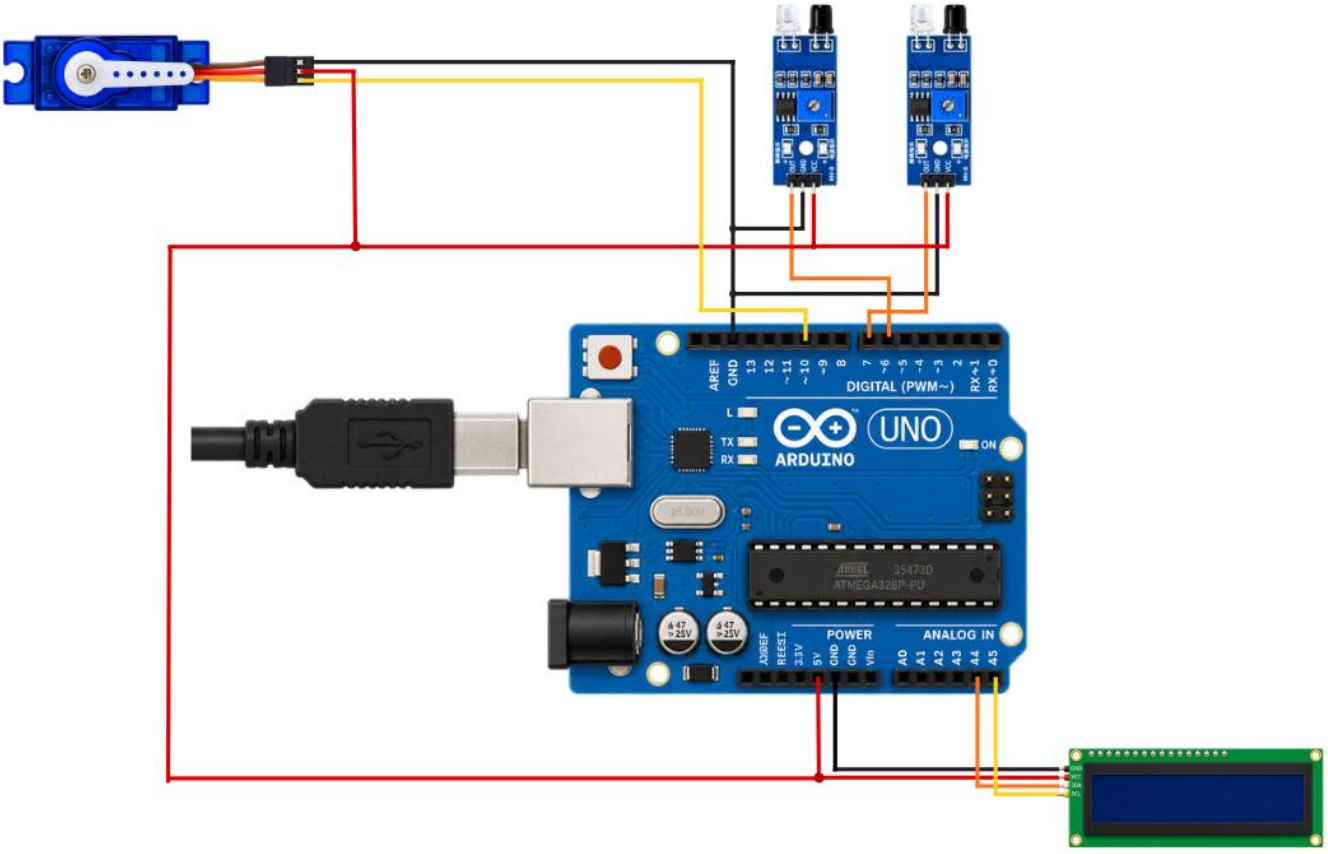
Gate **opens automatically** | LCD shows *"Vehicle Detected"*

b. When vehicle reaches **exit sensor**:

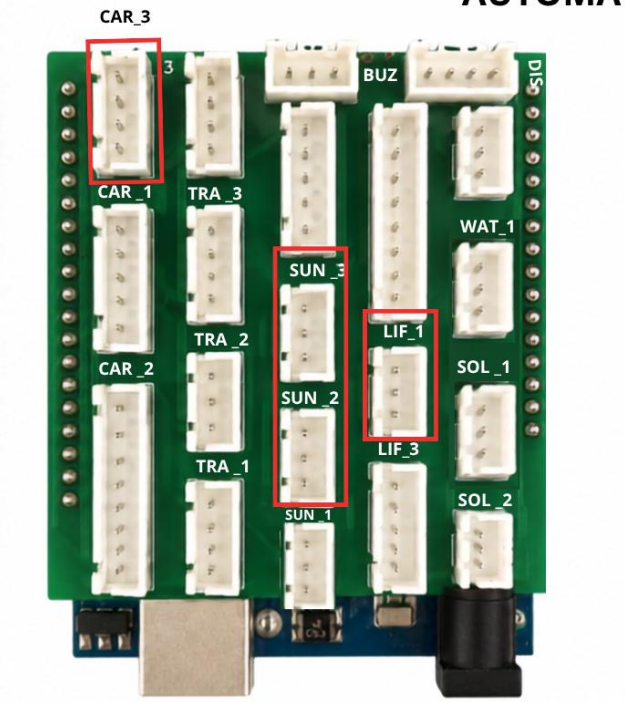
Gate **closes automatically** ⛔ | LCD shows *"Vehicle Passed"*

- **Connections**

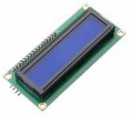
Connections			
Sensor PIN	**UNO Shield PIN**	**Shield Connector**	**Sensor/Component**
1: SDA	2: SDA	CAR_3	LCD Display
2: SCL	1: SCL		
3: +5V	3: +5V		
4: GND	4: GND		
2: GND	1: GND	SUN_1	IR Module 1
1: VCC	2: +5V		
3: OUT	3: D6		
2: GND	1: GND	SUN_2	IR Module 2
1: VCC	2: +5V		
3: OUT	3: D7		
1: GND	1: GND	LIF_1	Servo Motor
2: VCC	2: +5V		
3: Signal	3: D10		



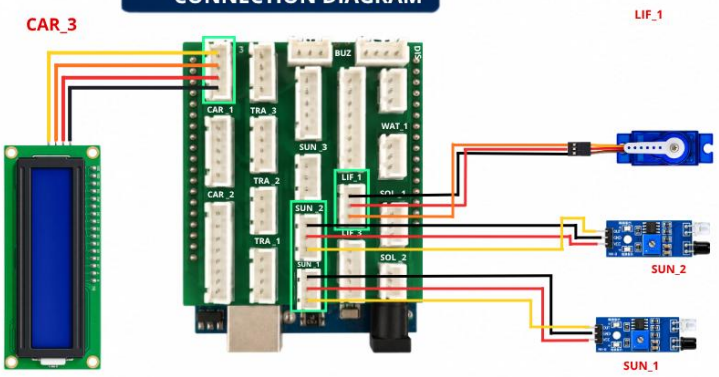
AUTOMATIC TOLL GATE



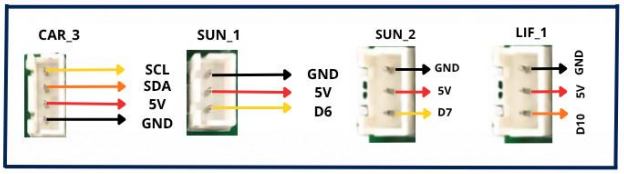
COMPONENTS REQUIRED

1. Servo Motor	2. I2C LCD Dispaly	3. IR Sensor
		
(Connect to LIF_1)	(Connect to CAR_3)	(Connect to SUN_2, SUN_3)

CONNECTION DIAGRAM



I2C LCD Display



SHIELD CONNECTION SUMMARY

Component	Shield Port	Description
IR Sensor, Servo Motor	SUN_2, SUN_3, LIF_1	Detects vehicle entry and exit, Opens/closes the toll gate.
I2C LCD Display	CAR_3	Displays toll gate status.

Program

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

#define ENTRY_IR 6
#define EXIT_IR 7
#define SERVO_PIN 10

Servo gateServo;
LiquidCrystal_I2C lcd(0x27, 16, 2);  // Change address if needed

void setup() {
pinMode(ENTRY_IR, INPUT);
pinMode(EXIT_IR, INPUT);

gateServo.attach(SERVO_PIN);
gateServo.write(0);      // Gate Closed
  lcd.init();
  lcd.backlight();

  lcd.setCursor(0,0);
  lcd.print(" AUTOMATIC TOLL ");
  lcd.setCursor(0,1);
  lcd.print("    SYSTEM   ");
  delay(2000);
  lcd.clear();
}

void loop() {

  // Entry detection
  if (digitalRead(ENTRY_IR) == LOW) {    // IR detects vehicle
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Vehicle Detected");
```

```
lcd.setCursor(0,1);
lcd.print("Gate Opening");

   gateServo.write(90);          //
Open gate
   delay(3000);                  //
Time to pass
}

// Exit detection
   if (digitalRead(EXIT_IR) ==
LOW) {
lcd.clear();
lcd.setCursor(0,0);
lcd.print("Vehicle flassed");
lcd.setCursor(0,1);
lcd.print("Gate Closing");

   gateServo.write(0);          //
Close gate
delay(2000);

lcd.clear();
lcd.setCursor(0,0);
lcd.print("WELCOME");
}
}
```

16. Touch Control

- ## What you Will Learn!

a. How a **touch sensor detects human touch**
b. How Arduino reads **digital input signals**
c. How to control an **LED using touch**
d. Basics of **simple automation systems**
e. Real-life use in **touch-based switches**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	2
6	4Pin JST Connector	0
7	Touch Sensor	1
8	LED or Buzzer Active/Passive	1

- ## Steps to make it Work!

a. Connect **touch sensor** to WAT-2
b. Connect **LED** to BUZ
c. Connect Arduino to laptop
d. Open Arduino IDE
e. Upload the code
f. Touch the sensor

- ## What should Happen!

a. When you **touch the sensor**:

   LED turns **ON**

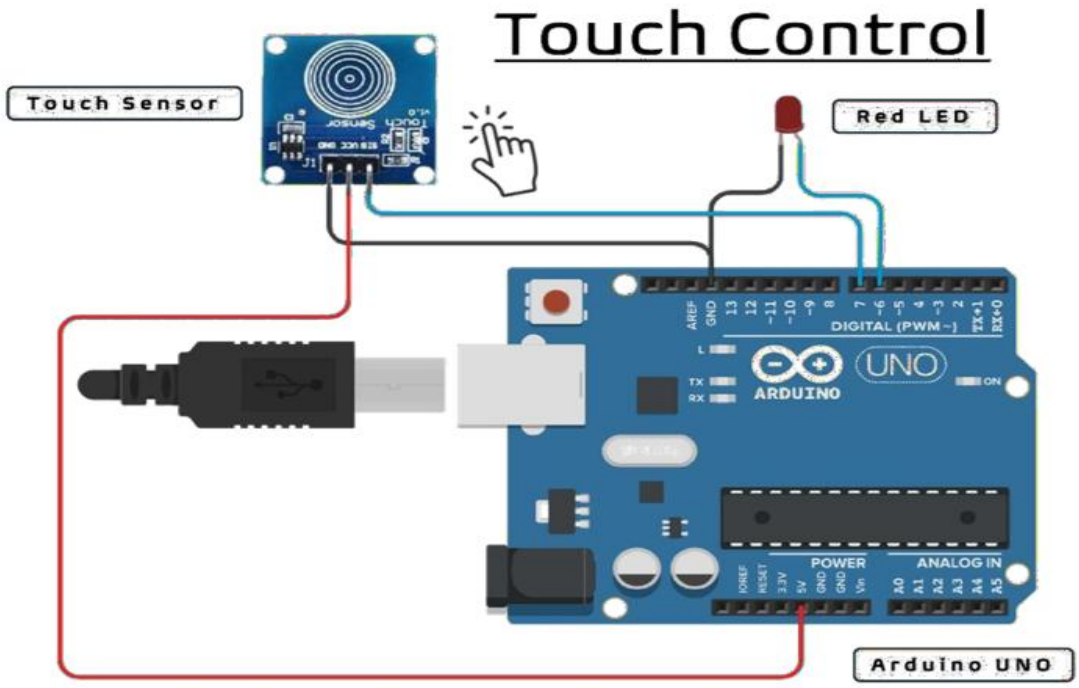
b. When you **release**:

   LED turns **OFF**

c. Serial Monitor shows touch status

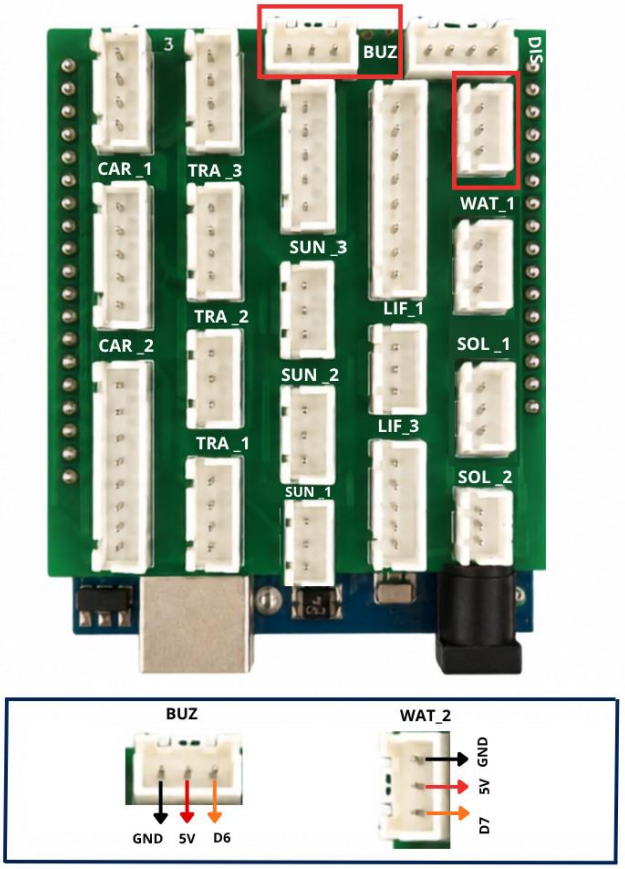
- **Connections**

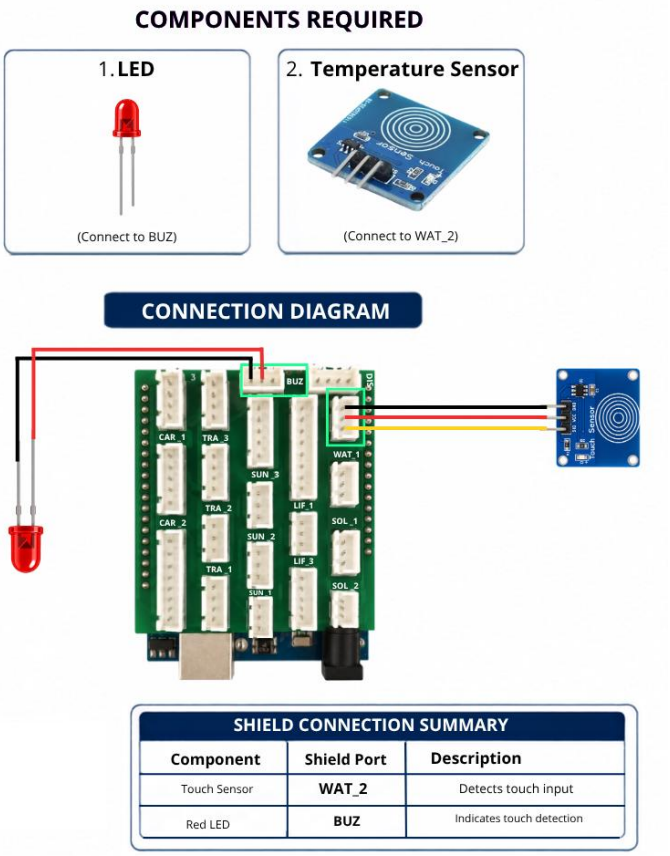
Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
	A0 (not used)		
1: GND	1: GND	WAT_2	Touch Sensor **TTP223 Touch Key Module**
3: +VCC	2: +5V		
2: DO	3: D7		
1: GND (Black wire)	1: GND	BUZ	Buzzer/LED
2: +V (Red Wire)	3: D6		



Connection Diagram

TOUCH CONTROL





SHIELD CONNECTION SUMMARY

Component	Shield Port	Description
Touch Sensor	WAT_2	Detects touch input
Red LED	BUZ	Indicates touch detection

Program

```
int touchflin = 7;          // Touch
sensor signal
int ledflin = 6;            // LED pin

void setup()
{
pinMode(touchflin, INflUT);
pinMode(ledflin, OUTflUT);

Serial.begin(9600);
}

void loop()
{
    int touchState =
digitalRead(touchflin);

    if(touchState == HIGH)       //
Sensor touched
{
digitalWrite(ledflin, HIGH);
    Serial.println("Touched - LED ON
");
}
else  // No
touch
{
digitalWrite(ledflin, LOW);
    Serial.println("Released -
LED OFF");
}

delay(50);
}
```

17. Touch-Less Alarm

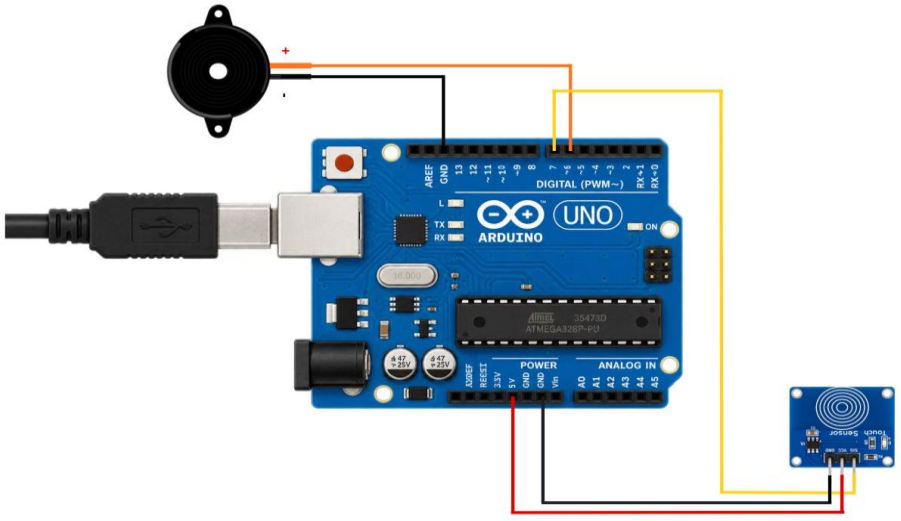
- What you Will Learn!

- Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	2
6	4Pin JST Connector	0
7	IR Module	1
8	Buzzer Active/Passive	1

- Connections

Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
	A0 (not used)		
1: GND	1: GND	WAT_2	Touch Sensor **TTP223 Touch Key Module**
3: +VCC	2: +5V		
2: DO	3: D7		
1: GND (Black wire)	1: GND	BUZ	Buzzer/LED
2: +V (Red Wire)	3: D6		



- **Steps to make it Work!**

a. Connect **sensor module** to WAT-2
b. Connect **LED or buzzer** to BUZ
c. Connect Arduino to laptop
d. Open Arduino IDE
e. Upload the code
f. Bring hand/object near sensor 7

- **What should Happen!**

When object is **detected (LOW)**:

- LED/Buzzer turns **ON** }_

When no object:

- LED/Buzzer turns **OFF** +

Serial Monitor shows status

Program

```
int touchflin = 7;          // Touch
sensor signal
int ledflin = 6;            // LED pin

void setup()
{
pinMode(touchflin, INflUT);
pinMode(ledflin, OUTflUT);

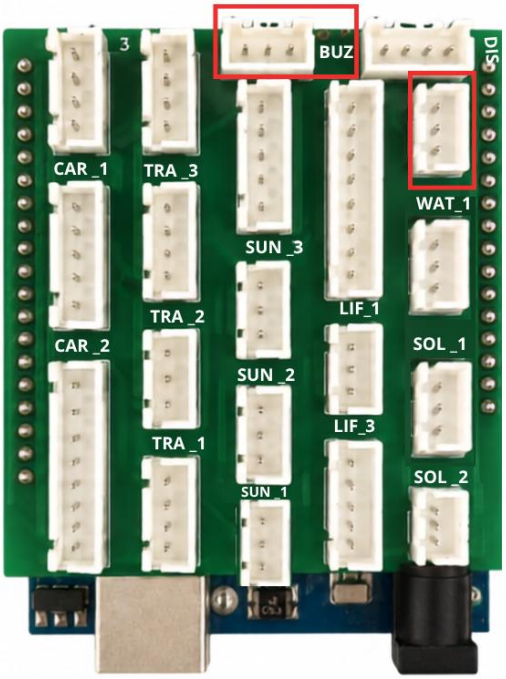
Serial.begin(9600);
}

void loop()
{
   int touchState =
digitalRead(touchflin);

if(touchState == LOW)       //
Sensor touched
{
digitalWrite(ledflin, HIGH);
Serial.println("Touched - LED ON
");
}
else                        // No
touch
{
digitalWrite(ledflin, LOW);
Serial.println("Released -
LED OFF");
}

delay(50);
}
```

TOUCH LESS ALARM





COMPONENTS REQUIRED

1. **Buzzer**



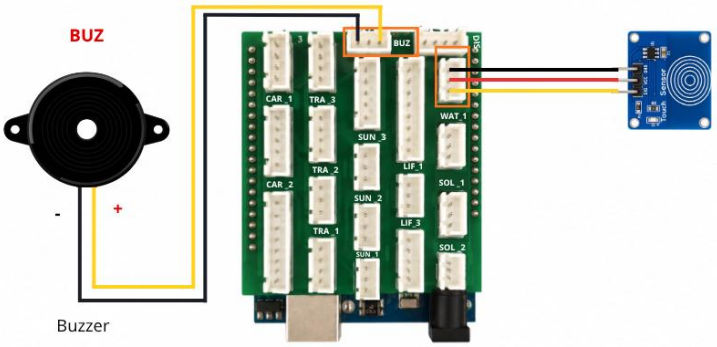
(Connect to BUZ)

2. **Temperature Sensor**



(Connect to WAT_2)

CONNECTION DIAGRAM



SHIELD CONNECTION SUMMARY

Component	Shield Port	Description
IR Sensor	WAT_2	Detects nearby objects
Buzzer	BUZ	Produces an alarm sound

18. Motion Detector

- ## What you Will Learn!

a. How a **PIR motion sensor detects movement**
b. How Arduino reads **digital HIGH/LOW signals**
c. How to trigger an **alarm (LED/Buzzer)**
d. Basics of **security & surveillance systems**
e. Real-life use in **automatic lighting & intrusion detection**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	2
6	4Pin JST Connector	0
7	5V Buzzer Active/Passive	1
8	Motion Senor Module	1

- ## Steps to make it Work!

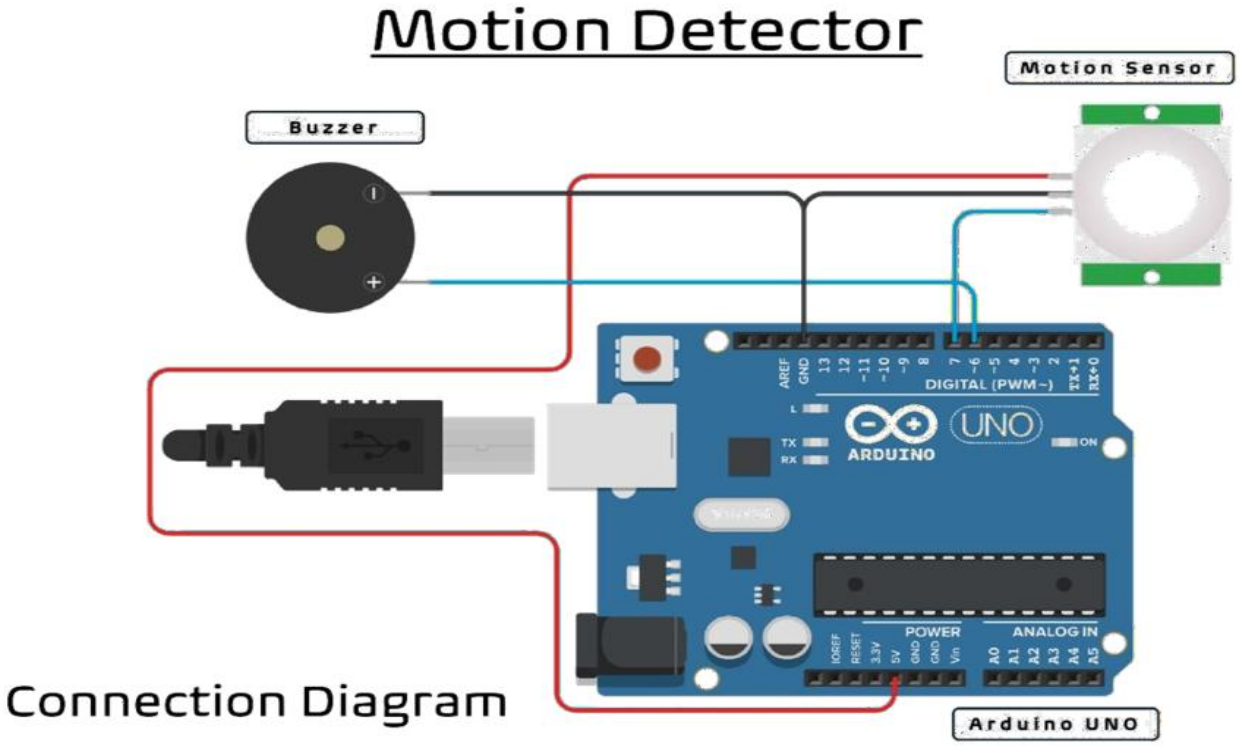
a. Connect **PIR sensor** to WAT-2
b. Connect **LED/Buzzer** to BUZ
c. Connect Arduino to laptop
d. Open Arduino IDE
e. Upload the code
f. Move in front of sensor

- ## What should Happen!

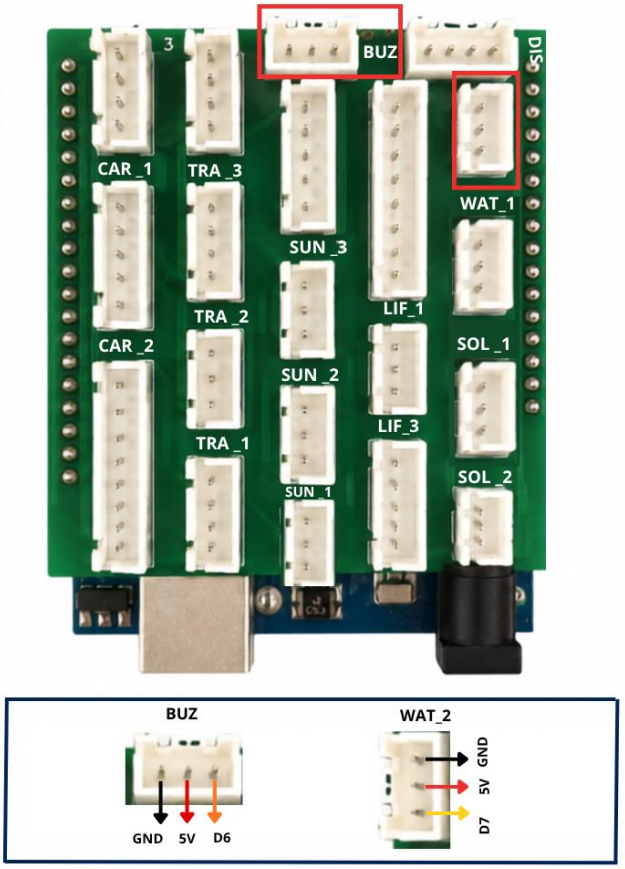
a. When motion is **detected**:
b. LED/Buzzer turns **ON**
c. When no motion:
d. LED/Buzzer turns **OFF**
e. Serial Monitor shows values (HIGH/LOW)

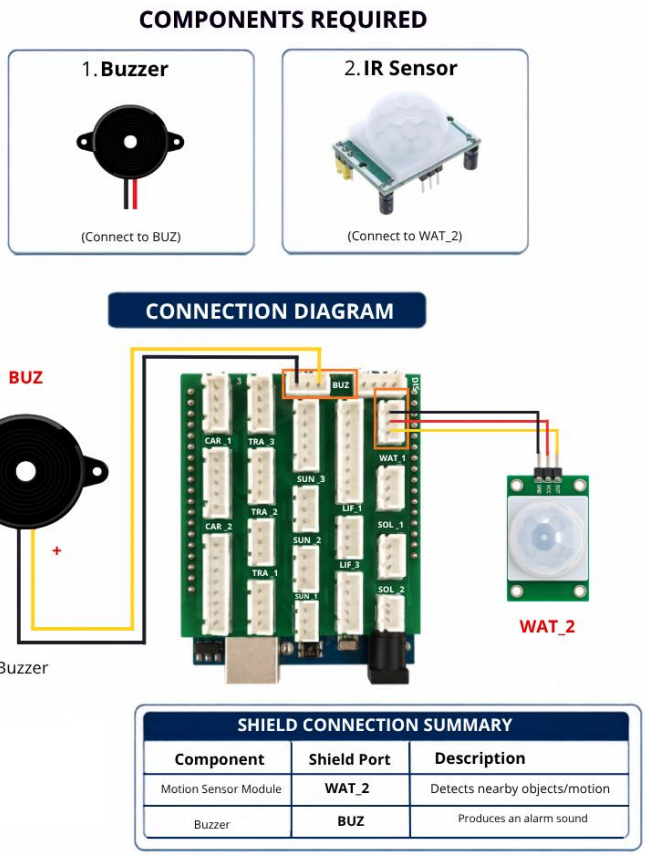
- **Connections**

Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
	A0 (not used)	WAT_2	**HC-SR501 PIR Motion Detector Sensor Module**
1: GND	1: GND		
3: +VCC	2: +5V		
2: OUT	3: D7		
1: GND (Black wire)	1: GND	BUZ	Buzzer/LED
2: +V (Red Wire)	3: D6		



MOTION DETECTOR





SHIELD CONNECTION SUMMARY

Component	Shield Port	Description
Motion Sensor Module	WAT_2	Detects nearby objects/motion
Buzzer	BUZ	Produces an alarm sound

Program

```
define flIR_flIN 7
define LED_flIN 6

void setup() {
pinMode(flIR_flIN, INflUT);
pinMode(LED_flIN, OUTflUT);
Serial.begin(9600);
}

void loop() {
int pirValue =
digitalRead(flIR_flIN);
Serial.println(pirValue);     //
For Serial flotter

if (pirValue == HIGH) {
digitalWrite(LED_flIN,
HIGH); // Radiation detected
delay(3000);
} else {
digitalWrite(LED_flIN,
LOW);  // No radiation change
}
delay(300);
}
```

19. LASER Security System

- ## What you Will Learn!

a. How a **laser beam + LDR detects intrusion**
b. How Arduino reads **digital signal changes**
c. How to create a **siren alarm using buzzer**
d. Basics of **security systems**
e. Real-life use in **laser-based protection systems**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	2
6	4Pin JST Connector	0
7	5V Buzzer Active/Passive	1
8	Laser Module	1

- ## Steps to make it Work!

a. Connect **laser module** to SUN-3
b. Connect **LDR module** to WAT-2
c. Connect **buzzer** to BUZ
d. Align laser directly on LDR
e. Connect Arduino to laptop
f. Upload the code
g. Break the laser beam

- ## What should Happen!

a. When laser beam is **aligned**:

   No alarm +

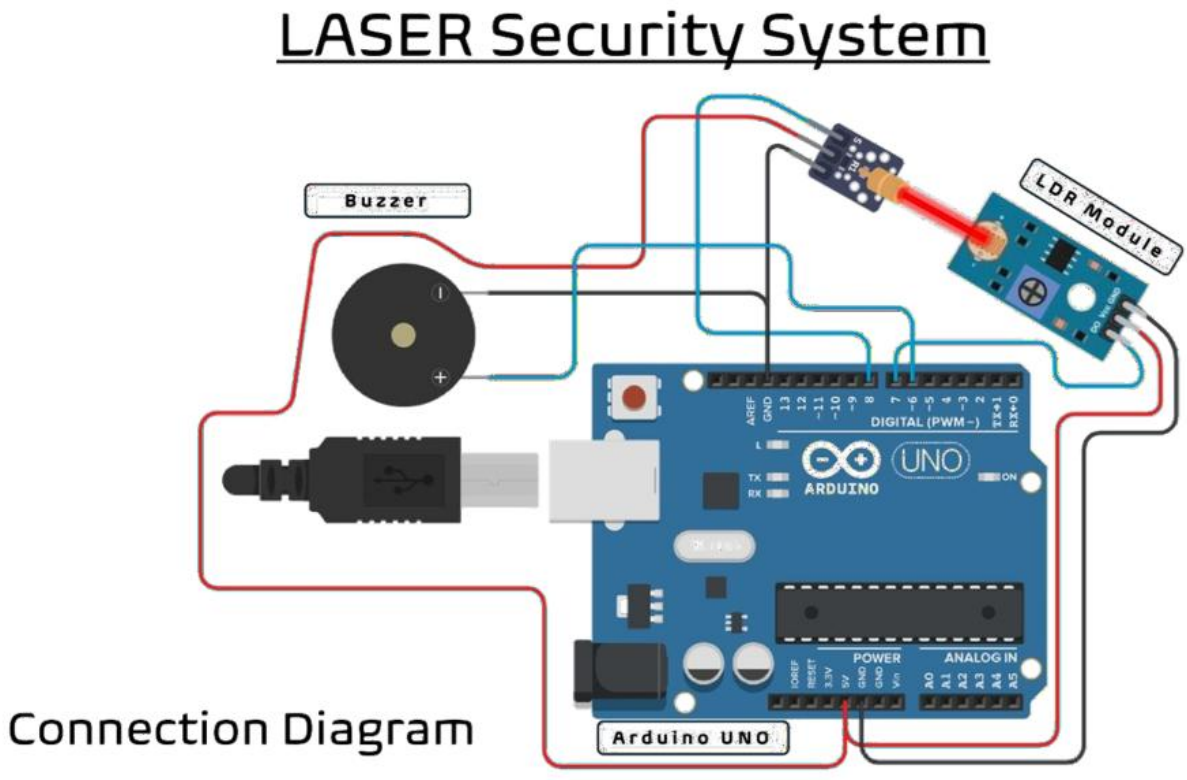
b. When beam is **broken**:

   Siren sound starts

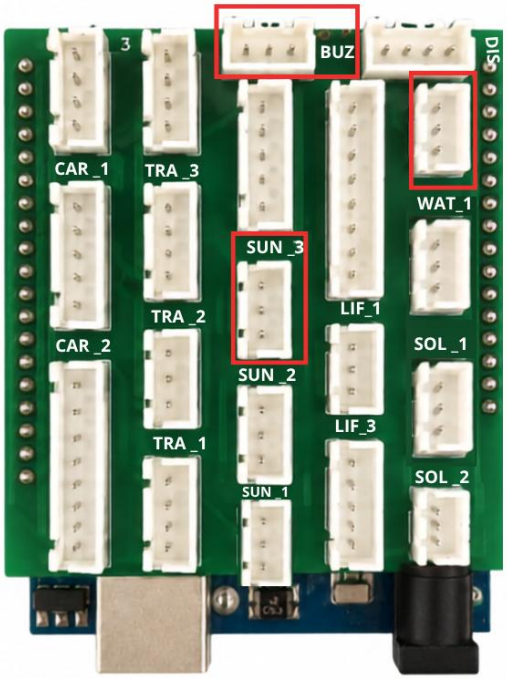
c. Continuous alert until beam is restored

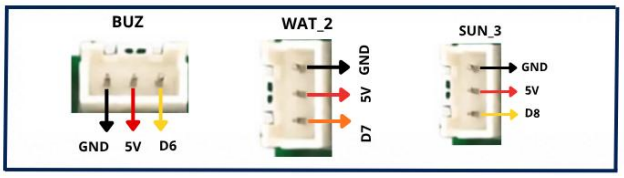
- **Connections**

Connections			
Sensor PIN	**UNO Shield PIN**	**Shield Connector**	**Sensor/Component**
	A0 (not used)	WAT_2	**Digital LDR Light Sensor Module**
2: GND	1: GND		
1: +VCC	2: +5V		
3: OUT	3: D7		
1: GND	1: GND	BUZ	Buzzer/LED
2: +V	3: D6		
1: GND	1: GND	SUN_3	**Laser Diode Module with PCB High Quality**
2: VCC	2: +5V		
3: S	3: D8		



LASER SECURITY SYSTEM





COMPONENTS REQUIRED

1. **Buzzer**



(Connect to BUZ)

2. **Laser Module**



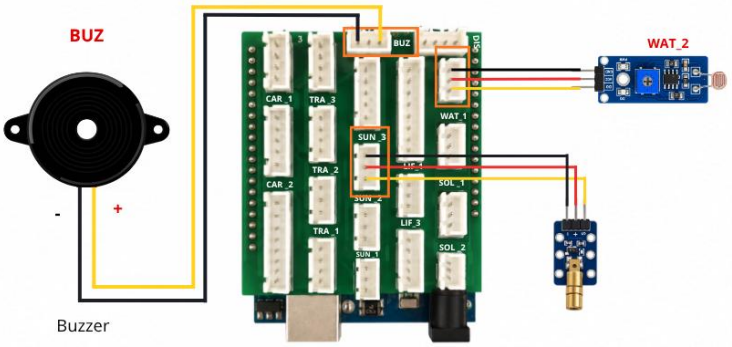
(Connect to WAT_2)

3. **LDR Module**



(Connect to WAT_2)

CONNECTION DIAGRAM



SHIELD CONNECTION SUMMARY

Component	Shield Port	Description
LDR Module, Laser Module	WAT_2, SUN_3	Detects laser/light interruption, Produces the laser beam
Buzzer	BUZ	Produces an alarm sound

Program

```
const int laserflin = 8;
const int buzzerflin = 6;
const int ldrflin = 7;

void setup() {
pinMode(laserflin, OUTflUT);
pinMode(buzzerflin, OUTflUT);
pinMode(ldrflin, INflUT);

digitalWrite(laserflin, HIGH);
}

void loop() {

   int beamState =
digitalRead(ldrflin);

   if (beamState == HIGH) {      //
Beam broken (change if reversed)

// Siren Up Sweep
   for (int freq = 800; freq <=
1500; freq += 10) {
tone(buzzerflin, freq);
delay(5);
}

// Siren Down Sweep
   for (int freq = 1500; freq >= 800;
freq -= 10) {
tone(buzzerflin, freq);
delay(5);
}

}
else {
noTone(buzzerflin);
}
}
```

20. Gardner Project

- ## What you Will Learn!

a. How a **soil moisture sensor detects dry/wet soil**

b. How Arduino controls a **relay module**

c. How to automate a **water pump system**

d. Basics of **smart irrigation**

e. Real-life use in **agriculture & home gardening**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	2
6	12 DC Submersible Pump	1
7	Soil Moisture Sensor	1
8	5V Relay Module	1

- ## Steps to make it Work!

a. Connect **soil moisture sensor** to WAT-2
b. Connect **relay module** to WAT-1
c. Connect **water pump through relay**
d. Provide **external power to pump (12V)**
e. Connect Arduino to laptop
f. Upload the code
g. Insert sensor into soil

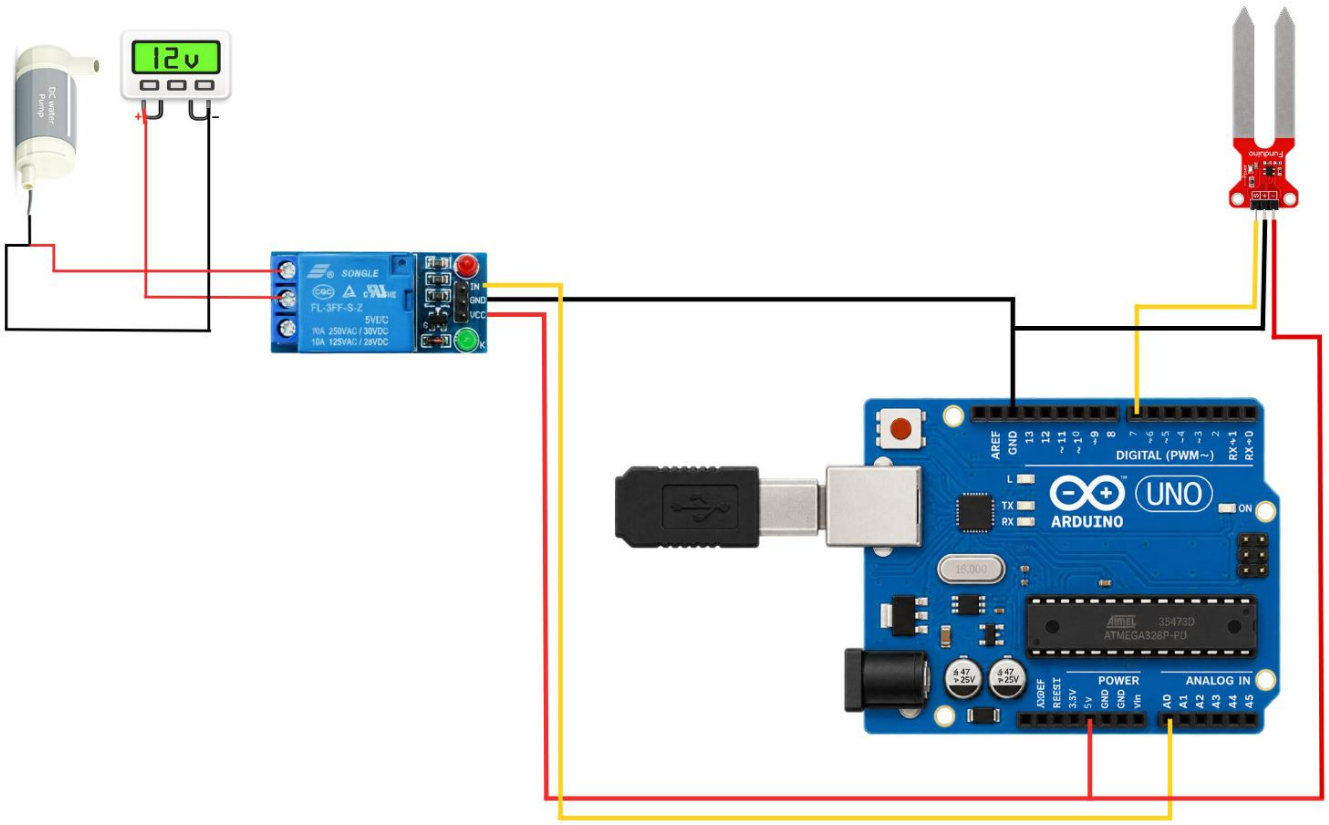
- ## What should Happen!

a. How a **soil moisture sensor detects dry/wet soil**
b. How Arduino controls a **relay module**
c. How to automate a **water pump system**

d. Basics of **smart irrigation**
e. Real-life use in **agriculture & home gardening**

- ## Connections

Component / Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
1: GND	1: GND	WAT_1	Relay Module **5V 1 Channel Relay Module**
2: VCC	2: +5V		
3: OUT	3: A0		
1: GND	1: GND	WAT_2	**HS-S33A, Soil Moisture Sensor**
2: V	2: +5V		
3: S	3: D7		



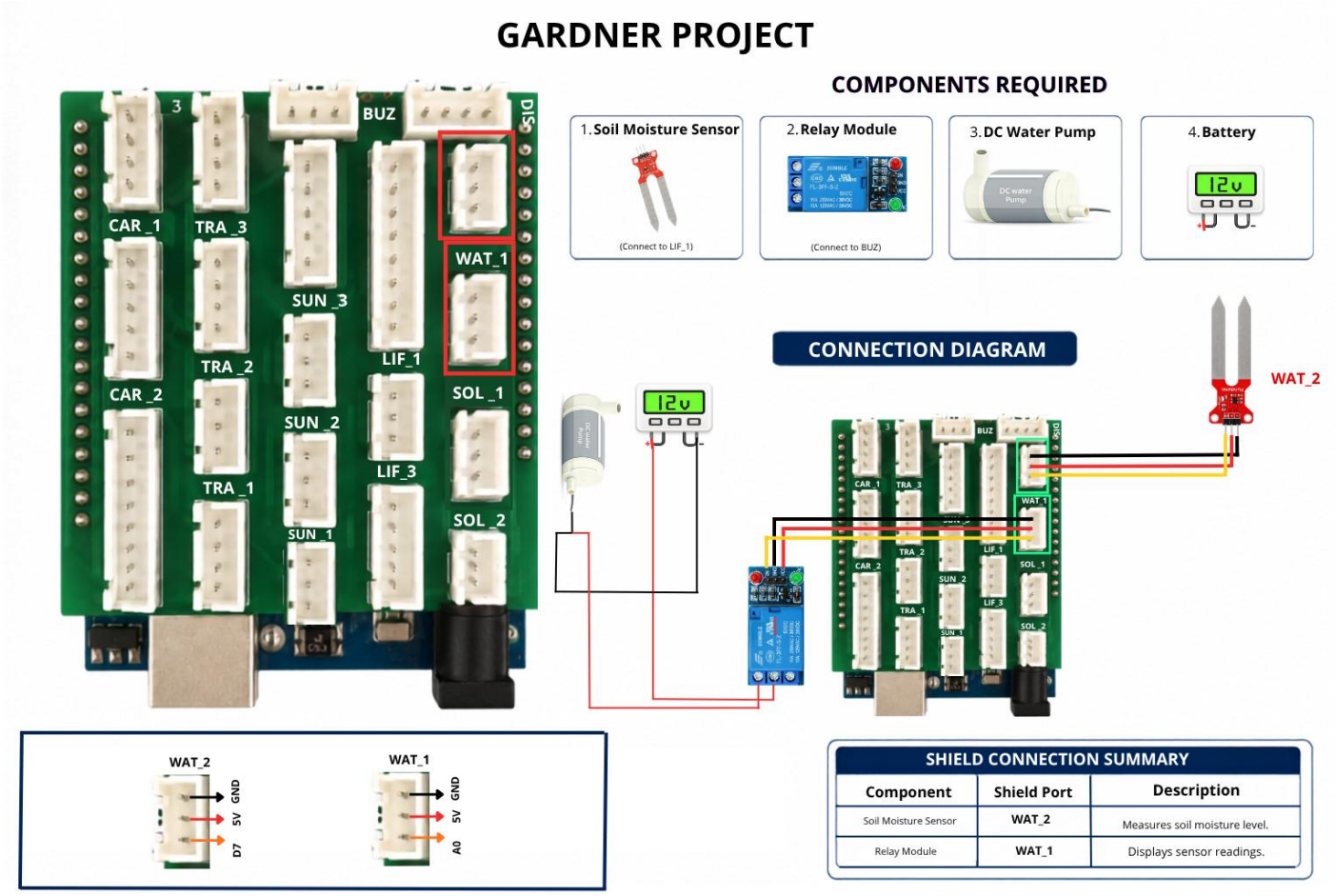
- ## Steps to make it Work!

a. How a **soil moisture sensor detects dry/wet soil**
b. How Arduino controls a **relay module**
c. How to automate a **water pump system**
d. Basics of **smart irrigation**

e. Real-life use in **agriculture & home gardening**

- **What should Happen!**

f. How a **soil moisture sensor detects dry/wet soil**
g. How Arduino controls a **relay module**
h. How to automate a **water pump system**
i. Basics of **smart irrigation**
j. Real-life use in **agriculture & home gardening**



Program

```
#define SOIL_SENSOR 7        // Soil moisture digital output pin
#define RELAY_flIN  A0       // Relay control pin
```

```
void setup() {
pinMode(SOIL_SENSOR,  INflUT);
pinMode(RELAY_flIN, OUTflUT);

    digitalWrite(RELAY_flIN,
HIGH);      // Relay OFF initially
(Active LOW relay assumed)

Serial.begin(9600);
}

void loop() {

    int soilState =
digitalRead(SOIL_SENSOR);

Serial.print("Soil State: ");
Serial.println(soilState);
•
    // If soil is DRY (usually LOW
from module)
if (soilState == LOW) {
Serial.println("Soil is DRY K
flump ON");
    digitalWrite(RELAY_flIN, LOW);
    // Turn relay ON (Active LOW)
}
else {
Serial.println("Soil is WEH K
flump OFF");
    digitalWrite(RELAY_flIN,
HIGH); // Turn  relay OFF
}

    delay(1000);      // 1 second delay
for stability
}
```

21. Sanitizer Despenser

- ## What you Will Learn!

a. How an **IR sensor detects hand/object**
b. How Arduino controls a **relay module**
c. How to automate a **liquid pump system**
d. Basics of **touchless hygiene systems**
e. Real-life use in **public safety & automation**

- ## Things you Need

Sr. No.	Item	Quantity
1	Arduino UNO	1
2	Botics Project Shield V.2	1
3	USB Cable (For Programming)	1
4	9V Battery & Snapper	1
5	3Pin JST Connector	2
6	5V Relay Module	1
7	12 DC Submersible Pump	1
8	IR Module	1

- ## Steps to make it Work!

a. Connect **IR sensor** to WAT-2
b. Connect **relay module** to WAT-1
c. Connect **pump through relay**
d. Provide **external power to pump (12V)**
e. Connect Arduino to laptop
f. Upload the code
g. Place hand near sensor

- ## What should Happen!

a. When hand is **detected**:

   Pump turns **ON** **(dispenses liquid)**

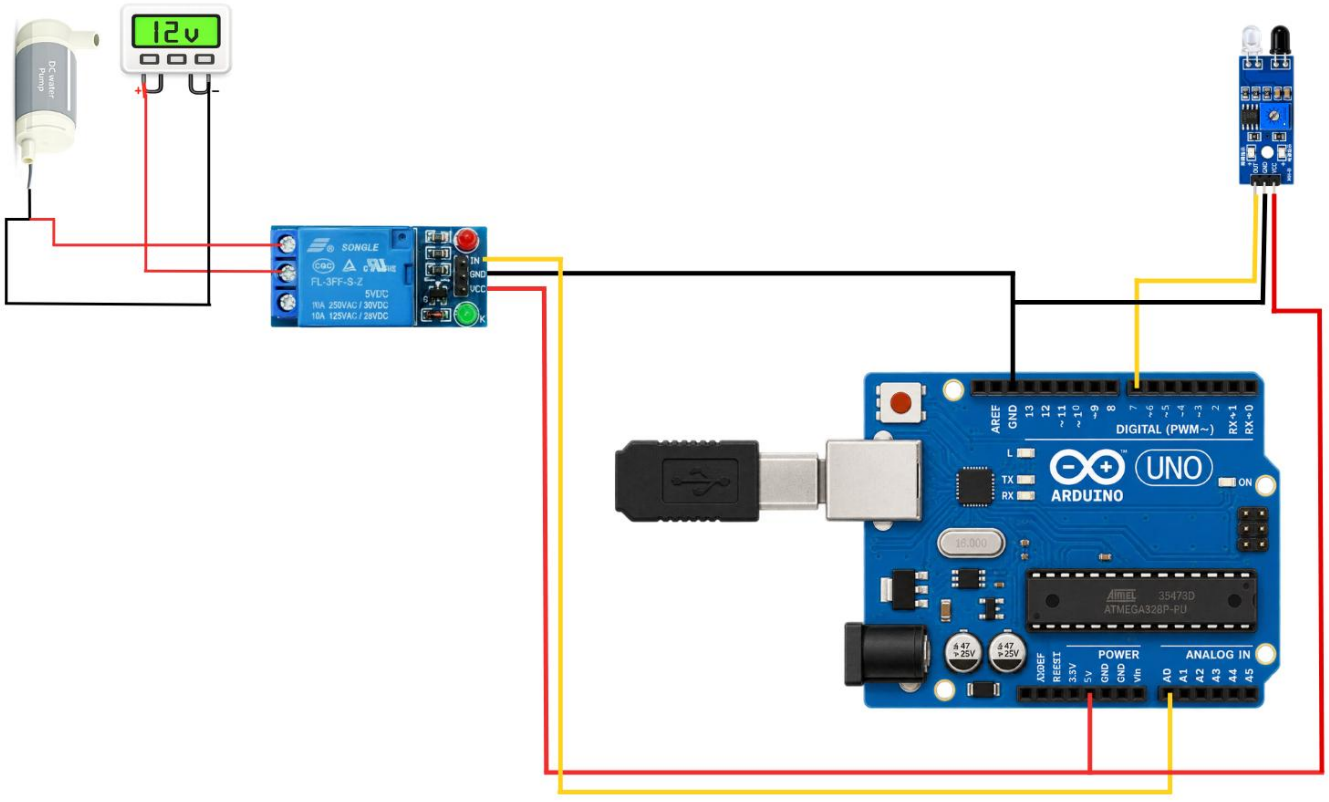
b. When no hand:

   Pump turns **OFF** +

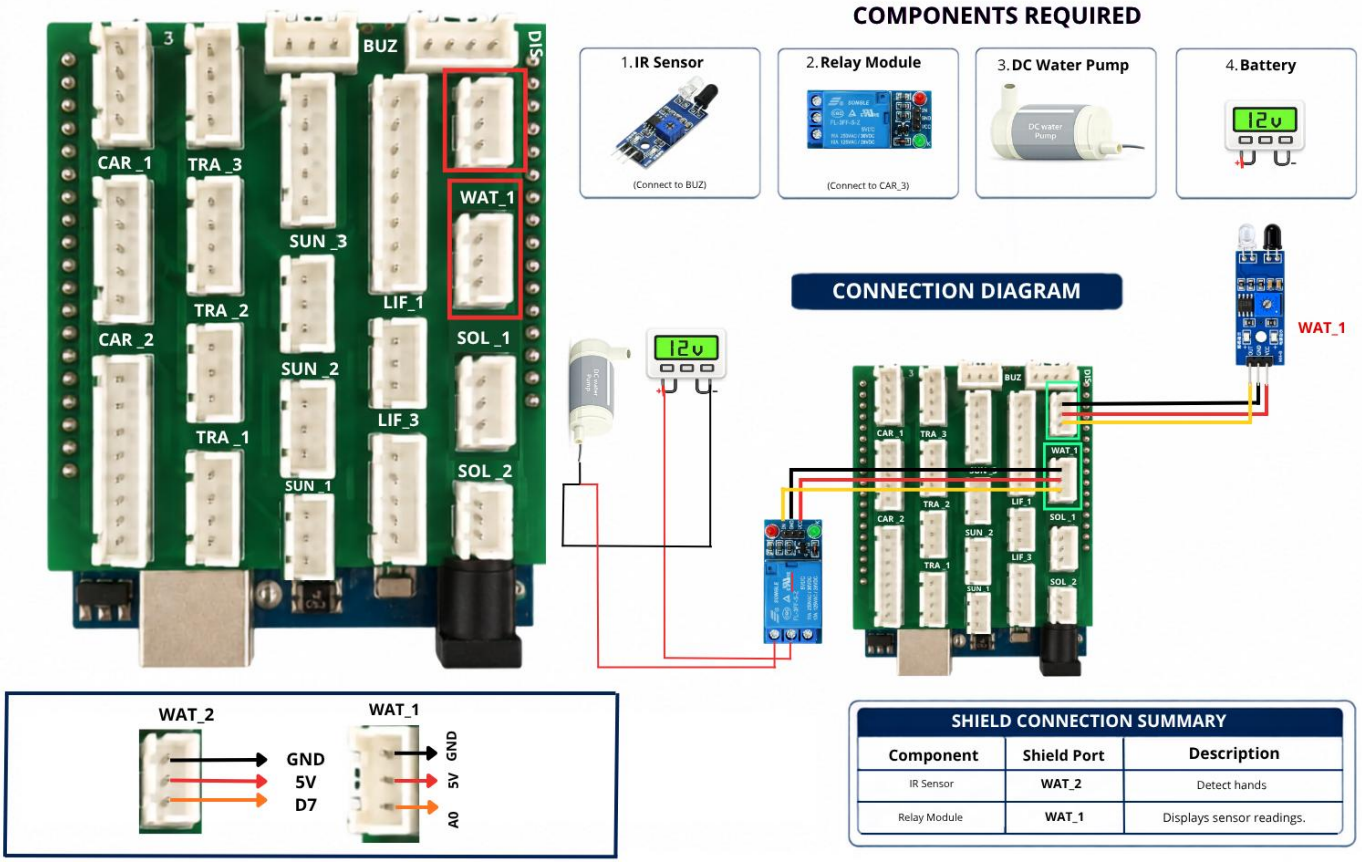
c. Fully **touchless system**

- **Connections**

Component / Sensor PIN	UNO Shield PIN	Shield Connector	Sensor/Component
1: GND	1: GND	WAT_1	Relay Module **5V 1 Channel Relay Module**
2: VCC	2: +5V		
3: OUT	3: A0		
2: GND	1: GND	WAT_2	**IR Module**
1: V	2: +5V		
3: OUT	3: D7		



SANITIZER DESPENSER



SHIELD CONNECTION SUMMARY		
Component	Shield Port	Description
IR Sensor	WAT_2	Detect hands
Relay Module	WAT_1	Displays sensor readings.

Program

```
#define IR_SENSOR 7        // IR module digital output
#define RELAY_PIN A0       // Relay control pin

void setup() {
pinMode(IR_SENSOR, INPUT);
pinMode(RELAY_PIN, OUTPUT);

  digitalWrite(RELAY_PIN, HIGH);    // Relay OFF initially (Active LOW relay)

Serial.begin(9600);
}

void loop() {

  int irState = digitalRead(IR_SENSOR);

  Serial.print("IR Sensor State: ");
Serial.println(irState);

  // If object detected (usually LOW)
if (irState == LOW) {
Serial.println("Object Detected K Relay ON");
  digitalWrite(RELAY_PIN, LOW); // Turn relay ON
}
else {
Serial.println("No Object K Relay OFF");
  digitalWrite(RELAY_PIN, HIGH); // Turn relay OFF
}

  delay(200);    // Small delay for stability
```