**Ultrasonic Range Measuring System**

**📘 Project Title:**

**Smart Distance Monitoring System Using ESP32 and Blynk**

**📌 Objective:**

To measure the distance of nearby objects using an ultrasonic sensor and provide real-time feedback through:

* **16x2 LCD display**
* **Red and Green LEDs**
* **Buzzer alert**
* **Blynk IoT platform (Web & Mobile Dashboard)**

**📦 Components Used**

| **Component Name** | **Pin No** | **Destination Component** | **Pin No** | **Special Remark** |
| --- | --- | --- | --- | --- |
| HC-SR04 Trigger Pin | D5 | ESP32 GPIO | D5 | Digital Output |
| HC-SR04 Echo Pin | D18 | ESP32 GPIO | D18 | Digital Input |
| Red LED (+) | D23 | ESP32 GPIO | D23 | Glows if object too far |
| Green LED (+) | D19 | ESP32 GPIO | D19 | Glows if object is close |
| Buzzer (+) | D4 | ESP32 GPIO | D4 | Beeps if object too far |
| LCD SDA | D21 | ESP32 GPIO | D21 | I2C Data Line |
| LCD SCL | D22 | ESP32 GPIO | D22 | I2C Clock Line |
| VCC (All components) | 5V | ESP32 5V | 5V | Power Supply |
| GND (All components) | GND | ESP32 GND | GND | Common Ground |

**📦 Required Libraries**

Install these in **Arduino IDE > Library Manager** (Ctrl + Shift + I or Tools > Manage Libraries):

| **Library Name** | **Description** |
| --- | --- |
| **Blynk** (v1.1.0 or newer) | For Blynk IoT cloud communication |
| **LiquidCrystal\_I2C** | For I2C LCD display |
| **Wire** (built-in) | Enables I2C communication |

**🔌 Circuit Diagram**

**(**[**URL:-https://app.cirkitdesigner.com/project/95bfa2e8-4d62-4933-92b1-40fa42d364e3**](URL:-https://app.cirkitdesigner.com/project/95bfa2e8-4d62-4933-92b1-40fa42d364e3)**)**

* Connect **HC-SR04**:
  + VCC → 5V
  + GND → GND
  + Trig → D5
  + Echo → D18
* Connect **16x2 I2C LCD**:
  + SDA → D21
  + SCL → D22
  + VCC → 5V
  + GND → GND
* Connect **Red LED** → D23 (with resistor ~220Ω)
* Connect **Green LED** → D19 (with resistor ~220Ω)
* Connect **Buzzer** → D4

**🧠 Working Principle**

1. **HC-SR04 sensor** sends ultrasonic pulses to detect the distance of objects.
2. **ESP32** processes this distance:
   * If the object is within a set range (e.g., < 20 cm):
     + Green LED glows.
     + LCD shows “WITHIN RANGE”.
   * If the object is farther:
     + Red LED and buzzer **blink/beep alternately**.
     + LCD alternates between showing distance and warning.
3. Distance is also sent to **Blynk IoT dashboard** via **Virtual Pin V0**.

**🧾 Code with Explanation**

**✅ Complete Arduino Code**

#define BLYNK\_TEMPLATE\_ID "TMPL3PMEeaoKT"

#define BLYNK\_TEMPLATE\_NAME "Ultrasonic Range Sensor"

#define BLYNK\_AUTH\_TOKEN "ji40lK4BvuFEtISK9Bzp8Ca-1wA4dySz"

#include <WiFi.h>

#include <BlynkSimpleEsp32.h>

#include <LiquidCrystal\_I2C.h>

// Pin Mapping

#define TRIG\_PIN 5

#define ECHO\_PIN 18

#define RED\_LED 23

#define GREEN\_LED 19

#define BUZZER 4

LiquidCrystal\_I2C lcd(0x27, 16, 2); // LCD I2C address

// WiFi credentials

char ssid[] = "Student";

char pass[] = "Learn@123";

// Blynk Timer for regular updates

BlynkTimer timer;

bool blinkState = false;

int rangeThreshold = 20; // in cm

bool isWithinRange = false;

// Function: Measures distance using HC-SR04

long getDistanceCM() {

digitalWrite(TRIG\_PIN, LOW);

delayMicroseconds(2);

digitalWrite(TRIG\_PIN, HIGH);

delayMicroseconds(10);

digitalWrite(TRIG\_PIN, LOW);

long duration = pulseIn(ECHO\_PIN, HIGH);

long distance = duration \* 0.034 / 2;

return distance;

}

// Function: Main logic for distance checking and outputs

void measureAndDisplay() {

long distance = getDistanceCM();

isWithinRange = distance < rangeThreshold;

// Print to Serial Monitor

Serial.print("Distance: ");

Serial.print(distance);

Serial.println(" cm");

// Send to Blynk App

Blynk.virtualWrite(V0, distance);

// If within range (near)

if (isWithinRange) {

digitalWrite(GREEN\_LED, HIGH);

digitalWrite(RED\_LED, LOW);

digitalWrite(BUZZER, LOW);

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Distance: ");

lcd.print(distance);

lcd.print("cm");

lcd.setCursor(0, 1);

lcd.print("WITHIN RANGE");

}

// If out of range (far)

else {

blinkState = !blinkState;

digitalWrite(GREEN\_LED, LOW);

digitalWrite(RED\_LED, blinkState);

digitalWrite(BUZZER, blinkState);

lcd.clear();

if (blinkState) {

lcd.setCursor(0, 0);

lcd.print("Distance: ");

lcd.print(distance);

lcd.print("cm");

lcd.setCursor(0, 1);

lcd.print("OUT OF RANGE");

} else {

lcd.setCursor(0, 0);

lcd.print("WARNING:");

lcd.setCursor(0, 1);

lcd.print("TOO FAR!");

}

}

}

void setup() {

Serial.begin(115200);

pinMode(TRIG\_PIN, OUTPUT);

pinMode(ECHO\_PIN, INPUT);

pinMode(RED\_LED, OUTPUT);

pinMode(GREEN\_LED, OUTPUT);

pinMode(BUZZER, OUTPUT);

lcd.init();

lcd.backlight();

Blynk.begin(BLYNK\_AUTH\_TOKEN, ssid, pass);

// Call measurement every second

timer.setInterval(1000L, measureAndDisplay);

}

void loop() {

Blynk.run();

timer.run();

}

**🌐 Blynk Setup Instructions (Online Version)**

1. Go to [https://blynk.cloud](https://blynk.cloud/) → Log in or Register.
2. Create New Template:
   * Name: Ultrasonic Range Sensor
   * Hardware: ESP32
   * Connection: WiFi
3. Add Datastream:
   * Type: Virtual Pin
   * Pin: V0
   * Data Type: Integer
4. Add Web Dashboard Widget:
   * Use **Label** or **Gauge**
   * Link to **V0**
5. Copy Template ID, Name, and Auth Token → Paste in Code

**🖥️ Serial Monitor Output Example**

Distance: 12 cm

→ WITHIN RANGE

Distance: 28 cm

→ OUT OF RANGE → Beeping + Red LED

**✅ Features**

* Real-time distance feedback
* Visual and sound alert for danger zone
* Blynk integration (remote monitoring)
* Clear LCD display status