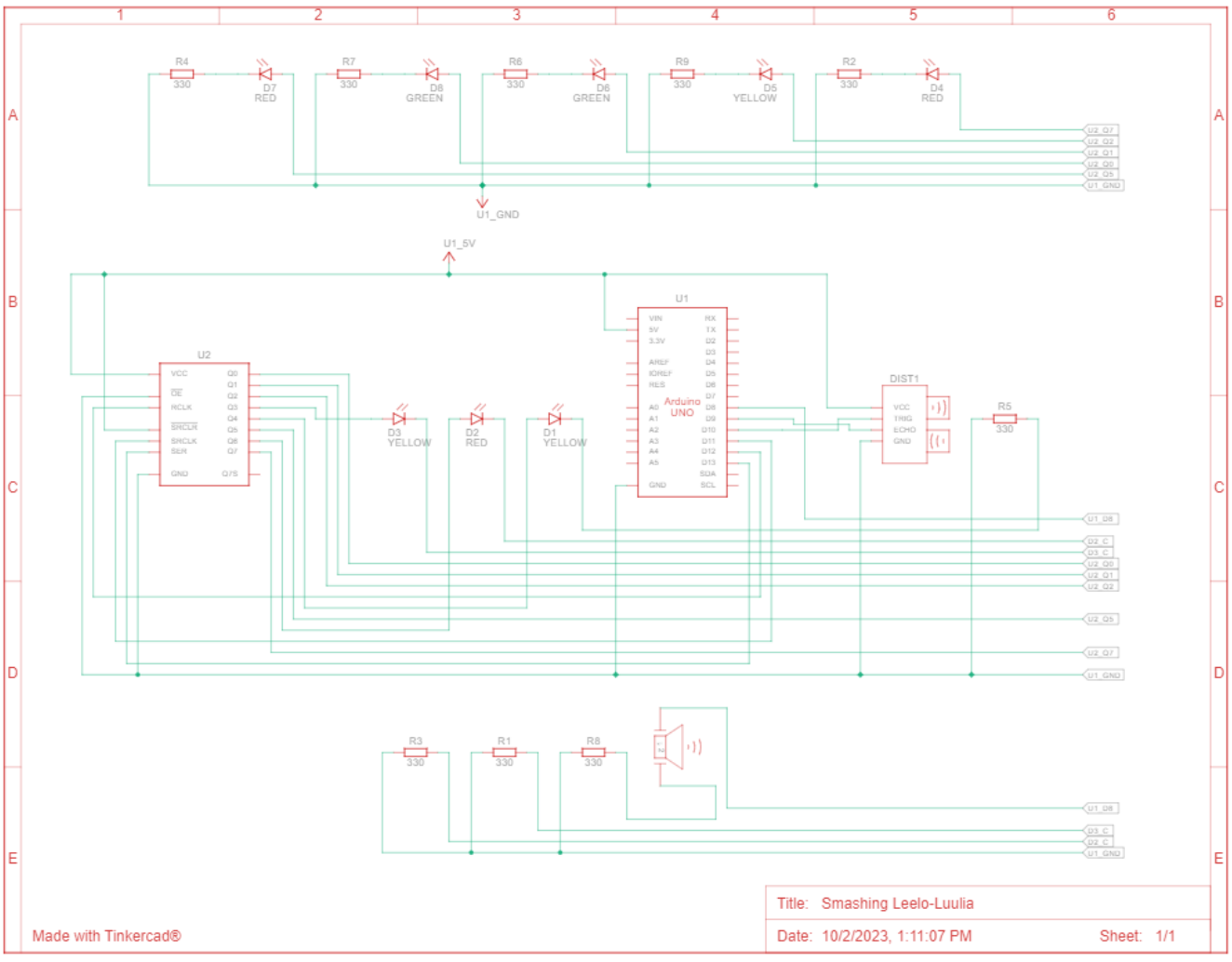
- **Parts used :**

  - 1x Breadboard
  - Jumper Wires

Name	Quantity	Component
U1	1	Arduino Uno R3
D1 D3 D5	3	Yellow LED
D2 D4 D7	3	Red LED
D6 D8	2	Green LED
R1 R2 R3 R4 R5 R6 R7 R9 R8	9	330 Ω Resistor
U2	1	8-Bit Shift Register
DIST1	1	Ultrasonic Distance Sensor
PIEZO1	1	Piezo

Project Description :

This project utilizes an Arduino board to create a distance-monitoring system using an ultrasonic sensor (HC-SR04). The system measures distances in real-time, displaying the results on a row of LEDs and emitting audible alerts through a buzzer when certain distance thresholds are crossed. This project is both educational and practical, as it can be used to create a simple distance measurement and alert system for various applications.



Algorithm and Program source code:

A) Algorithm :

1. **Initialize Pins:**
    - Assign pin numbers for the ultrasonic sensor's trigger and echo pins, LED display pins, and buzzer pin.
    - Set the initial state and mode of these pins (output or input) using the pinMode() function.

2. **Main Loop:**
    - Enter the main loop (loop()) which runs continuously.\

3. **Ultrasonic Sensor Measurement:**
    - Send a short pulse to the ultrasonic sensor's trigger pin to initiate a measurement.
    - Measure the time taken for the pulse to bounce back and receive it on the echo pin.
    - Convert the duration into distance in centimeters using the formula: distance = (duration / 2) / 29.1.

4. **Check Distance Range:**
    - Check if the measured distance is within a valid range (between 0 and 150 cm). If not, print "Out of range" to the serial monitor.
    - If the distance is within the valid range, print the distance in centimeters to the serial monitor.

5. **Map Distance to LED Display Range:**
    - Map the measured distance to a range suitable for the LED display using the map() function. The distance range is mapped to LED index range (0 to 7) to determine which LED to light up based on the distance.

6. **Buzzer Alerts:**
    - Depending on the mapped distance value (LOOK_UP), trigger different buzzer alerts.
        - If LOOK_UP is between 3 and 4, play a short beep with a frequency of 200 Hz for 200 milliseconds.
        - If LOOK_UP is between 5 and 6, play a longer beep with a frequency of 5000 Hz for 200 milliseconds.
        - If LOOK_UP is 7, turn off the buzzer (no tone).

7. **Update LED Display:**
    - Shift out the corresponding LED pattern (from the values_leds array) to the shift register using the shiftOut() function. This pattern represents the LED indicators for the current distance range.
    - Update the LED display by latching the shift register output to the LEDs.

8. **Delay and Repeat:**
    - Introduce a delay of 200 milliseconds to control the refresh rate of the distance measurement, LED display, and buzzer alerts.
    - Repeat the loop to continuously monitor the distance and update the display and buzzer alerts accordingly.

B) Program Source Code :

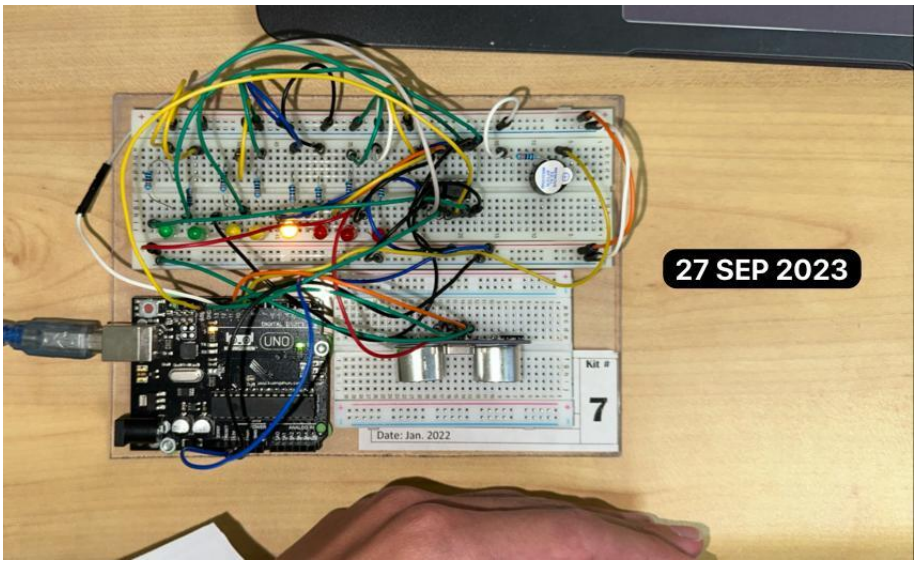
```
int buzzerPin = 8;
int trigPin = 10;
int echoPin = 9;
int clockPin = 11;
int latchPin = 12;
int dataPin = 13;
byte values_leds[9] =
{
  B00000001,
  B00000010,
  B00000100,
  B00001000,
  B00010000,
  B00100000,
  B01000000,
  B10000000,
  B00000000,
};
int LOOK_UP = 0;
int duration;
int distance;

void setup()
{
  Serial.begin (9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(clockPin, OUTPUT);
  pinMode(latchPin, OUTPUT);
  pinMode(dataPin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);
}

void loop()
{
  digitalWrite(latchPin, LOW);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(1000);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = (duration / 2) / 29.1;

  if (distance >= 150 || distance <= 0)
  {
    Serial.println("Out of range");
  }
  else
```

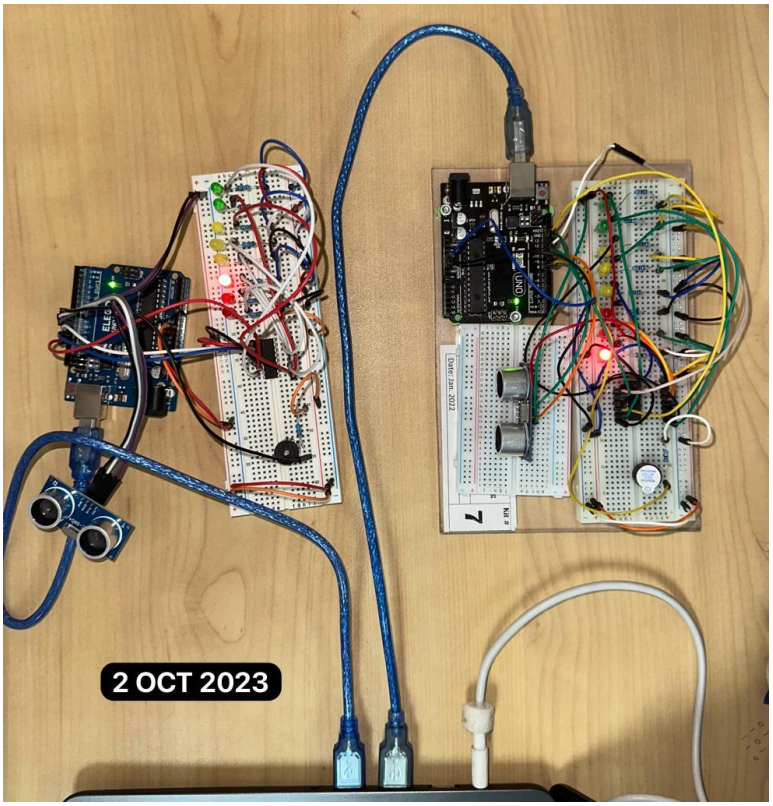
```
  {
    Serial.print(distance);
    Serial.println("cm");
  }

  LOOK_UP = map(distance,  0, 300, 7, 0);

  if (LOOK_UP <= 0) {
    LOOK_UP = 0;
  }
  else if (LOOK_UP >= 3 && LOOK_UP <= 4)
  {
    tone(buzzerPin, 200000, 200);
    delay(100);
    noTone(buzzerPin);
  }
  else if (LOOK_UP >= 5 && LOOK_UP <= 6)
  {
    tone(buzzerPin, 5000, 200);
    delay(200);
    noTone(buzzerPin);
  }
  else if (LOOK_UP == 7)
  {
    noTone(buzzerPin);
  }
  shiftOut(dataPin, clockPin, MSBFIRST, values_leds[LOOK_UP]);
  digitalWrite(latchPin, HIGH);
  delay(200);
}
```

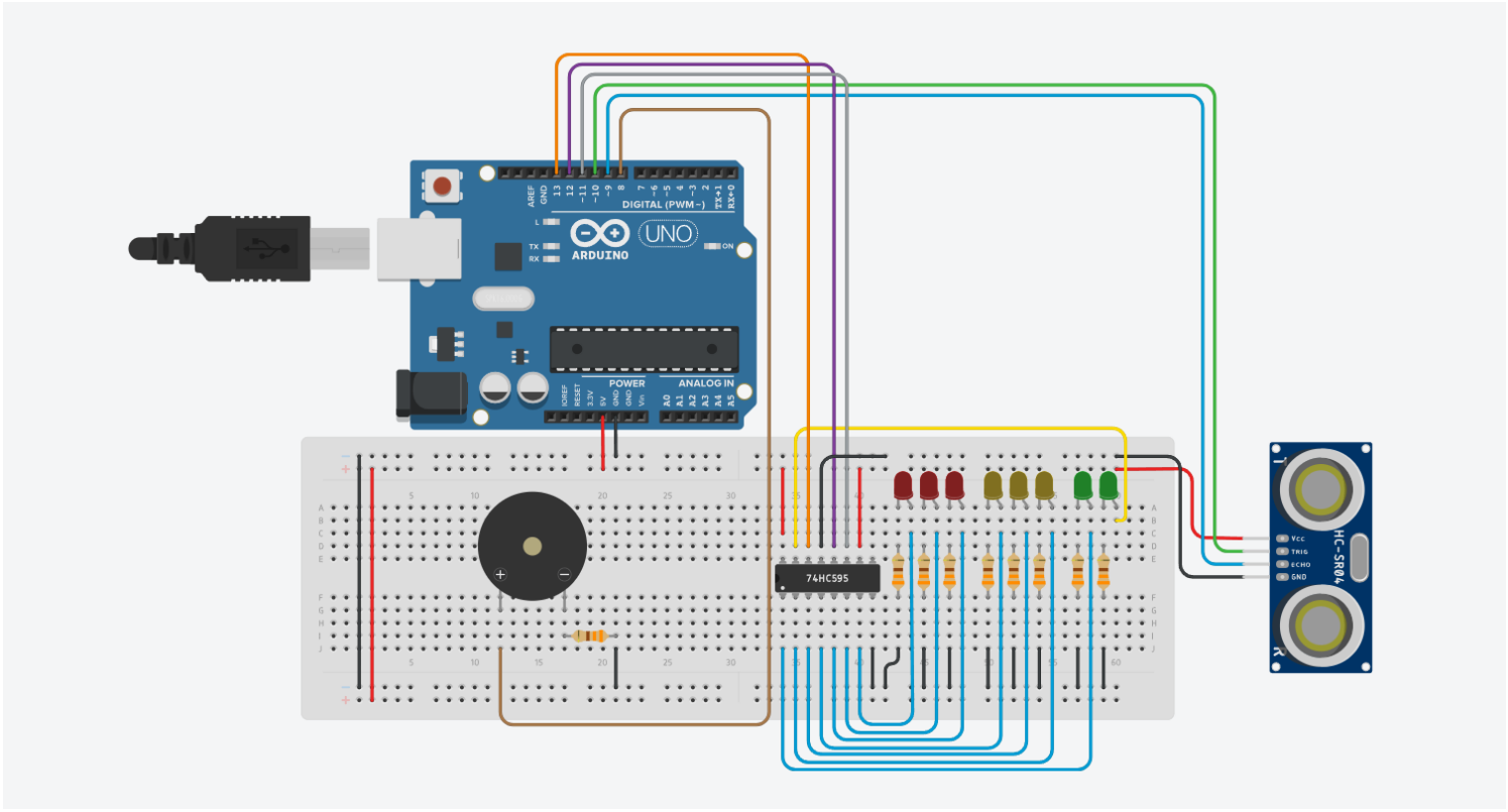
Program description :

1. **Initialize the Arduino pins for various components:**
    - **'buzzerPin': Initialize the pin for the buzzer.**
    - **'trigPin' and' echoPin': Initialize pins for the ultrasonic sensor (HC-SR04).**
    - **'clockPin', 'latchPin', and 'dataPin': Initialize pins for controlling the LED display.**
    - **'values_leds[]': Define an array of binary values representing LED patterns.**
2. **Set up the Arduino in the' setup()' function:**
    - **Initialize the serial communication at a baud rate of 9600 for debugging.**
    - **Configure pins as input or output:**
        - **'trigPin': Set as OUTPUT for sending trigger signals to the ultrasonic sensor.**
        - **'echoPin': Set as INPUT for receiving echo signals from the ultrasonic sensor.**
        - **'clockPin',' latchPin', and 'dataPin': Set as OUTPUT for controlling the LED display.**
        - **'buzzerPin': Set as OUTPUT for controlling the buzzer.**
3. **Enter the main loop in the 'loop()' function:**
    - **Lower the latch pin to prepare for LED data transmission.**
    - **Send a trigger signal to the ultrasonic sensor by setting 'trigPin' to HIGH and then LOW after a short delay.**
    - **Measure the duration it takes for the echo signal to return from the ultrasonic sensor and calculate the distance in centimetres.**
    - **Check if the distance is within a valid range (between 0 and 150 cm).**
    - **If the distance is valid:**
        - **Print the distance in centimetres to the serial monitor.**
        - **Map the distance value to an index in the 'values_leds[]' array.**
        - **Depending on the mapped index (LOOK_UP), control the buzzer and update the LED display:**
            - **If LOOK_UP is 0 or below 3, turn off the buzzer.**
            - **If LOOK_UP is between 3 and 4, emit a short beep and delay.**
            - **If LOOK_UP is between 5 and 6, emit a longer beep and delay.**
            - **If LOOK_UP is 7, turn off the buzzer.**
            - **Update the LED display based on LOOK_UP.**
    - **Raise the latch pin to display the LED data.**
    - **Introduce a delay to control the update rate of the system.**
4. **The loop continues to run indefinitely, constantly measuring the distance, updating the LED display, and emitting buzzer alerts based on the current distance.**

Implementation In the Laboratory :






DIGITAL (PWM~)
UNO
ARDUINO
POWER
ANALOG IN
74HC595
HC-SR04
VCC
TRIG
ECHO
GND