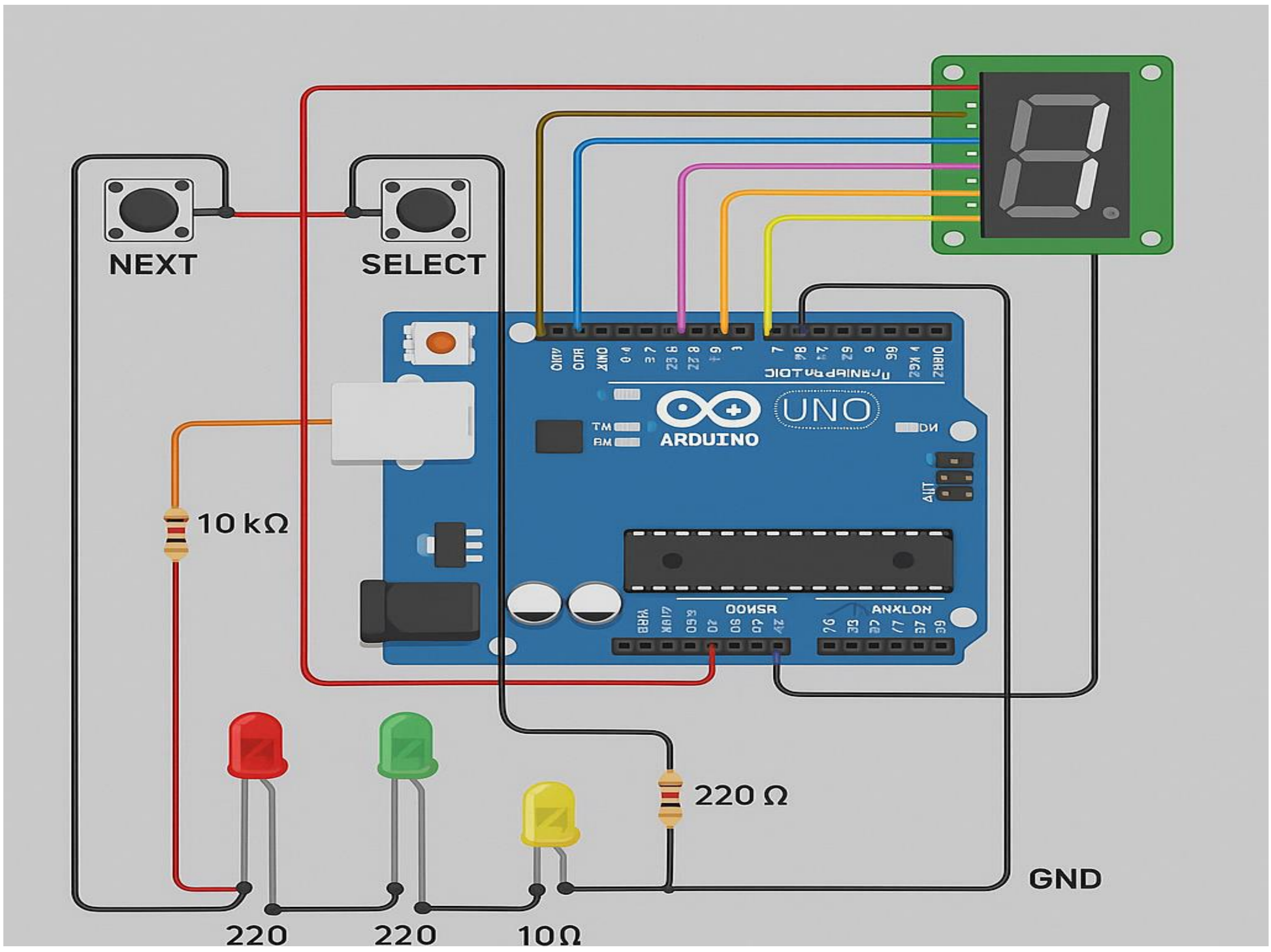
Rotary Encoder (By using Push-Button)

Components Required:

1. Ardiuno/Nano
2. Seven-segment display/LCD
3. Push Buttons (2 to 3)
4. 3 LEDs (Different colours)
5. 10k resistors (total 2)
6. 220 resistors (total 3)

Diagram:



Code :

```
const int segA = 5;

const int segB = 6;

const int segC = 7;
const int segD = 8;
const int segE = 9;
const int segF = A0;
const int segG = A1;

// Pin definitions for individual LEDs
const int redLED = 10;
const int yellowLED = 11;
const int greenLED = 12;

// Array to store the segment patterns for digits 0-9
const byte digitPatterns[10][7] = {
  {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
};

// Variables for non-blocking timing
unsigned long previousMillisLED1 = 0;
unsigned long previousMillisLED2 = 0;
unsigned long previousMillisLED3 = 0;
unsigned long previousMillisTimer = 0;

// Blink intervals (milliseconds)
const long intervalLED1 = 500;   // Red LED blinks every 500ms
const long intervalLED2 = 1000;  // Yellow LED blinks every 1000ms
const long intervalLED3 = 1500;  // Green LED blinks every 1500ms
const long intervalTimer = 1000; // Update timer every second

// State variables
int seconds = 0;
bool led1State = LOW;
bool led2State = LOW;
bool led3State = LOW;
bool timerRunning = true;

void setup() {
  // Set all segment pins as outputs
  pinMode(segA, OUTPUT);
  pinMode(segB, OUTPUT);
  pinMode(segC, OUTPUT);
  pinMode(segD, OUTPUT);
  pinMode(segE, OUTPUT);
  pinMode(segF, OUTPUT);
  pinMode(segG, OUTPUT);

  // Set LED pins as outputs
  pinMode(redLED, OUTPUT);
  pinMode(yellowLED, OUTPUT);
  pinMode(greenLED, OUTPUT);

  // Initialize the display
  displayDigit(0);

  Serial.begin(9600);
  Serial.println("Seven-segment display timer with blinking LEDs - direct pin control");
  Serial.println("Enter 't' to toggle timer");
  Serial.println("Enter 'r' to reset timer");
}

void loop() {
```

```
// Current time
unsigned long currentMillis = millis();

// Handle Serial input
handleSerialInput();

// Handle timer updating
if (timerRunning && (currentMillis - previousMillisTimer >= intervalTimer)) {
  previousMillisTimer = currentMillis;
  seconds = (seconds + 1) % 10;  // 0-9 counter
  displayDigit(seconds);
  Serial.print("Timer: ");
  Serial.println(seconds);
}

// Handle RED LED blinking (independent of display)
if (currentMillis - previousMillisLED1 >= intervalLED1) {
  previousMillisLED1 = currentMillis;
  led1State = !led1State;
  digitalWrite(redLED, led1State);
}

// Handle YELLOW LED blinking (independent of display)
if (currentMillis - previousMillisLED2 >= intervalLED2) {
  previousMillisLED2 = currentMillis;
  led2State = !led2State;
  digitalWrite(yellowLED, led2State);
}

// Handle GREEN LED blinking (independent of display)
if (currentMillis - previousMillisLED3 >= intervalLED3) {
  previousMillisLED3 = currentMillis;
  led3State = !led3State;
  digitalWrite(greenLED, led3State);
  }
}

// Function to display a digit on the seven-segment display
void displayDigit(int digit) {
  if (digit >= 0 && digit <= 9) {
    digitalWrite(segA, digitPatterns[digit][0]);
    digitalWrite(segB, digitPatterns[digit][1]);
    digitalWrite(segC, digitPatterns[digit][2]);
    digitalWrite(segD, digitPatterns[digit][3]);
    digitalWrite(segE, digitPatterns[digit][4]);
    digitalWrite(segF, digitPatterns[digit][5]);
    digitalWrite(segG, digitPatterns[digit][6]);
  }
}

void handleSerialInput() {
  if (Serial.available() > 0) {
    char input = Serial.read();

    if (input >= '0' && input <= '9') {
      seconds = input - '0';  // Convert ASCII to integer
      displayDigit(seconds);
      Serial.print("Set timer to: ");
      Serial.println(seconds);
    }
    else if (input == 't' || input == 'T') {
      timerRunning = !timerRunning;
      Serial.print("Timer: ");
      Serial.println(timerRunning ? "RUNNING" : "PAUSED");
    }
    else if (input == 'r' || input == 'R') {
      seconds = 0;
      displayDigit(seconds);
      Serial.println("Timer reset to 0");
    }
  }
}
```