```
#include <LiquidCrystal.h>
#include <Servo.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);  // LCD pin configuration
Servo myServo;  // Servo object

const int yesButton = 7;  // "Yes" button
const int noButton = 8;   // "No" button
const int servoPin = 9;   // Servo motor pin
const int trigPin = 10;   // Ultrasonic sensor trigger
const int echoPin = 6;    // Ultrasonic sensor echo

const int redPin = A0;    // Red pin of RGB LED
const int greenPin = A1;  // Green pin of RGB LED
const int bluePin = A2;   // Blue pin of RGB LED

int lastYesState = HIGH;
int lastNoState = HIGH;
int questionIndex = 0;
int servoAngle = 90;  // Start servo in middle
bool systemActive = false;  // Tracks if person is detected
bool ledOverride = false;  // Prevent flickering when LED is actively showing
red/green

unsigned long lastDetectionTime = 0;  // Tracks last presence time
unsigned long lastSensorCheck = 0;    // Tracks last sensor check time
unsigned long lastLEDChange = 0;      // Tracks last LED flicker time
const int sensorInterval = 500;       // Ultrasonic sensor checks every 500ms
const int resetTime = 15000;          // Reset system after 15 seconds without person
const int ledFlickerInterval = 1000;  // Change LED color every 1 second (flickering
effect)

const char* questions[] = {
   "Do you want free cash?",
   "Do you really need the money?",
   "Are you donating the money?",
   "Is saving just delaying consumption?",
   "Is your credit score above 300?",
   "Does paying bills on time make you financially responsible?",
   "Are you going to be financially responsible with this money?",
   "Are you about to buy something dumb with this money?",
   "If you put your money in a blender, is it liquid assets?",
```

```
    "If time is money, is an ATM a time machine?"
};

// Responses array for randomization
const char* responses[] = {
    "hm...",
    "thought so...",
    "interesting...",
    "really?",
    "let's see...",
    "are you sure?",
    "okay then...",
    "we'll see about that...",
    "hmmm, intriguing..."
};

void setup() {
    pinMode(yesButton, INPUT_PULLUP);
    pinMode(noButton, INPUT_PULLUP);
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);

    pinMode(redPin, OUTPUT);
    pinMode(greenPin, OUTPUT);
    pinMode(bluePin, OUTPUT);

    lcd.begin(16, 2);
    myServo.attach(servoPin);
    myServo.write(servoAngle);  // Start at 90°

    lcd.print("Waiting for you...");
}

void loop() {
    unsigned long currentMillis = millis();  // Current time

    // Only check ultrasonic sensor every 500ms (instead of every loop)
    if (currentMillis - lastSensorCheck >= sensorInterval) {
        lastSensorCheck = currentMillis;
        bool personDetected = detectPerson();

        if (personDetected) {
```

```
            lastDetectionTime = currentMillis;  // Update last seen time

            if (!systemActive) {
                systemActive = true;  // Activate system once
                questionIndex = 0;  // Reset questions
                servoAngle = 90;  // Reset servo to middle
                myServo.write(servoAngle);
                lcd.clear();
                displayQuestion(questions[questionIndex]);
            }
        }
    }

    // Allow button presses while system is active
    if (systemActive) {
        checkButtons();
    }

    // Handle LED flickering when idle
    if (!ledOverride && millis() - lastLEDChange >= ledFlickerInterval) {
        lastLEDChange = millis();
        flickerLED();  // Random flickering effect
    }

    // Reset system if person is gone for more than 15 seconds
    if (systemActive && (millis() - lastDetectionTime > resetTime)) {
        systemActive = false;
        lcd.clear();
        lcd.print("Waiting for you...");
    }
}

// Function to detect a person using ultrasonic sensor (non-blocking)
bool detectPerson() {
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    long duration = pulseIn(echoPin, HIGH, 30000);  // Timeout to avoid getting stuck
    int distance = duration * 0.034 / 2;  // Convert to cm
```

```
    return (distance > 0 && distance <= 70);  // Detect if within 70 cm
}

// Function to check button presses
void checkButtons() {
    int yesState = digitalRead(yesButton);
    int noState = digitalRead(noButton);

    if (yesState == LOW && lastYesState == HIGH) {
        moveServo(-25);  // Rotate -25° for "Yes"
        displayRandomResponse();  // Display a random response
        changeLEDColor("green");  // Turn RGB LED green for "Yes"
        delay(2000);  // Allow response to be visible for 2 seconds
        nextQuestion();
    }

    if (noState == LOW && lastNoState == HIGH) {
        moveServo(25);  // Rotate +25° for "No"
        displayRandomResponse();  // Display a random response
        changeLEDColor("red");  // Turn RGB LED red for "No"
        delay(2000);  // Allow response to be visible for 2 seconds
        nextQuestion();
    }

    lastYesState = yesState;
    lastNoState = noState;
}

// Function to move servo
void moveServo(int angleChange) {
    servoAngle += angleChange;

    if (servoAngle > 180) servoAngle = 180;
    if (servoAngle < 0) servoAngle = 0;

    myServo.write(servoAngle);
    delay(300);
}

// Function to cycle through questions
void nextQuestion() {
```

```
    questionIndex = (questionIndex + 1) % 10;
    displayQuestion(questions[questionIndex]);
}

// Function to display and scroll long text
void displayQuestion(const char* text) {
    lcd.clear();
    int len = strlen(text);

    if (len <= 16) {
        lcd.print(text);  // Print directly if it fits
    } else {
        for (int i = 0; i <= len - 16; i++) {
            lcd.clear();
            lcd.setCursor(0, 0);
            lcd.print(text + i);
            delay(300);
        }
    }
}

// Function to display a random response
void displayRandomResponse() {
    lcd.clear();  // Clear screen before showing response
    int randomIndex = random(0, 9);  // Random index between 0 and 8
    lcd.print(responses[randomIndex]);  // Display random response
}

// Function to change RGB LED color instantly
void changeLEDColor(String color) {
    ledOverride = true;  // Stop flickering temporarily
    if (color == "green") {
        analogWrite(redPin, 0);
        analogWrite(greenPin, 255);
        analogWrite(bluePin, 0);
    } else if (color == "red") {
        analogWrite(redPin, 255);
        analogWrite(greenPin, 0);
        analogWrite(bluePin, 0);
    }
    delay(500);
    ledOverride = false;  // Resume flickering after a short delay
```

```
}

// Function to make RGB LED flicker between random colors
void flickerLED() {
   int r = random(50, 200);
   int g = random(50, 200);
   int b = random(50, 200);

   analogWrite(redPin, r);
   analogWrite(greenPin, g);
   analogWrite(bluePin, b);
}
```