ESP Code

```
/*
 * Smart Surveillance Turret - WiFi SMS Version
 * ESP32 DevKit V1
 * Supports: Twilio, CallMeBot (WhatsApp), and HTTP SMS APIs
 *
 * Hardware Connections:
 * - PIR Sensor OUT -> GPIO 13
 * - Motor Driver IN1 -> GPIO 25
 * - Motor Driver IN2 -> GPIO 26
 * - Motor Driver ENA -> GPIO 27 (PWM)
 */

#include <WiFi.h>
#include <HTTPClient.h>
#include <WiFiClientSecure.h>
#include <base64.h>
#include <time.h>


//NTP Time Server Configuration
#define NTP_SERVER "pool.ntp.org"
#define GMT_OFFSET_SEC 19800      // IST is UTC+5:30 (19800 seconds)
#define DAYLIGHT_OFFSET_SEC 0     // India doesn't use daylight saving

// ==================== WiFi Configuration ====================
#define WIFI_SSID "Redmi Note 14 5G"        // Replace with your WiFi name
#define WIFI_PASSWORD "12345678" // Replace with your WiFi password

// ==================== SMS Service Selection ====================
// Uncomment ONE of the following options:

// OPTION 1: Twilio (Reliable, paid service - $0.0079 per SMS)
#define USE_TWILIO
#define TWILIO_ACCOUNT_SID "AC8e3c20e93787d2d4a44beb855908d44f"
#define TWILIO_AUTH_TOKEN "db4f8173045eac368a6ccd5209762c03"
#define TWILIO_PHONE_NUMBER "+12272652168"  // Your Twilio number
#define RECIPIENT_PHONE "+918073197844"     // Recipient's number

// OPTION 2: CallMeBot WhatsApp (Free, uses WhatsApp)
 //#define USE_CALLMEBOT_WHATSAPP
 //#define CALLMEBOT_PHONE "+918073197844"  // Recipient's WhatsApp number (with
country code)
//#define CALLMEBOT_API_KEY "123123"       // Get from:
https://www.callmebot.com/blog/free-api-whatsapp-messages/

// OPTION 3: CallMeBot SMS (Free, limited)
// #define USE_CALLMEBOT_SMS
// #define CALLMEBOT_SMS_PHONE "+919876543210"
// #define CALLMEBOT_SMS_API_KEY "your_api_key"

// OPTION 4: Custom HTTP API (for Circuit Digest or other services)
// #define USE_CUSTOM_API
// #define CUSTOM_API_URL "http://api.example.com/send"
// #define CUSTOM_API_KEY "your_api_key"
// #define CUSTOM_PHONE "+919876543210"

// Pin Definitions
#define PIR_SENSOR_PIN 13
#define MOTOR_IN1 25
#define MOTOR_IN2 26
#define MOTOR_ENA 27

// PWM Settings
#define PWM_FREQ 5000
```

```
#define PWM_RESOLUTION
#define MOTOR_SPEED 200  // 0-255, adjust based on your motor

// Timing Settings
#define ROTATION_TIME_35_DEG 500  // milliseconds to rotate 35 degrees
(calibrate this)
#define STOP_DURATION 5000        // 5 seconds stop time
#define DEBOUNCE_TIME 2000        // 2 seconds debounce for PIR

// Position tracking
enum Position { LEFT, CENTER, RIGHT };
Position currentPosition = CENTER;

// Motion detection
bool motionDetected = false;
unsigned long lastMotionTime = 0;
bool alertSent = false;

// WiFi status
bool wifiConnected = false;

void setup() {
  Serial.begin(115200);
  Serial.println("\n\nSmart Surveillance Turret Starting...");

  // Configure PIR sensor
  pinMode(PIR_SENSOR_PIN, INPUT);

  // Configure motor pins
  pinMode(MOTOR_IN1, OUTPUT);
  pinMode(MOTOR_IN2, OUTPUT);
  pinMode(MOTOR_ENA, OUTPUT);

  // Setup PWM for motor speed control
  ledcAttach(MOTOR_ENA, PWM_FREQ, PWM_RESOLUTION);

  // Connect to WiFi
  connectWiFi();

  // Stop motor initially
  stopMotor();

  Serial.println("System Ready!");
  delay(2000);  // Allow PIR sensor to stabilize
}

void loop() {
  // Check WiFi connection
  if (WiFi.status() != WL_CONNECTED) {
    Serial.println("WiFi disconnected! Reconnecting...");
    connectWiFi();
  }

  // Check for motion
  checkMotion();

  // Perform scanning pattern
  scanArea();

  delay(100);
}

void connectWiFi() {
  Serial.print("Connecting to WiFi: ");
```

```
  Serial.println(WIFI_SSID);

  WiFi.mode(WIFI_STA);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);

  int attempts = 0;
  while (WiFi.status() != WL_CONNECTED && attempts < 20) {
    delay(500);
    Serial.print(".");
    attempts++;
  }

  if (WiFi.status() == WL_CONNECTED) {
    wifiConnected = true;
    Serial.println("\nWiFi Connected!");
    Serial.print("IP Address: ");
    Serial.println(WiFi.localIP());
    Serial.print("Signal Strength: ");
    Serial.print(WiFi.RSSI());
    Serial.println(" dBm");
  } else {
    wifiConnected = false;
    Serial.println("\nWiFi Connection Failed!");
  }
}

void scanArea() {
  // Move to LEFT position
  moveToLeft();
  currentPosition = LEFT;
  stopAndWait();

  // Move to CENTER position
  moveToCenter();
  currentPosition = CENTER;
  stopAndWait();

  // Move to RIGHT position
  moveToRight();
  currentPosition = RIGHT;
  stopAndWait();

  // Return to CENTER
  moveToCenter();
  currentPosition = CENTER;
  stopAndWait();
}

void moveToLeft() {
  Serial.println("Moving to LEFT position...");
  rotateMotor(true, ROTATION_TIME_35_DEG);
}

void moveToCenter() {
  Serial.println("Moving to CENTER position...");
  if (currentPosition == LEFT) {
    rotateMotor(false, ROTATION_TIME_35_DEG);
  } else if (currentPosition == RIGHT) {
    rotateMotor(true, ROTATION_TIME_35_DEG);
  }
}

void moveToRight() {
  Serial.println("Moving to RIGHT position...");
```

```
  rotateMotor(false, ROTATION_TIME_35_DEG);
}

void rotateMotor(bool clockwise, int duration) {
  ledcWrite(MOTOR_ENA, MOTOR_SPEED);

  if (clockwise) {
    digitalWrite(MOTOR_IN1, HIGH);
    digitalWrite(MOTOR_IN2, LOW);
  } else {
    digitalWrite(MOTOR_IN1, LOW);
    digitalWrite(MOTOR_IN2, HIGH);
  }

  delay(duration);
  stopMotor();
}

void stopMotor() {
  digitalWrite(MOTOR_IN1, LOW);
  digitalWrite(MOTOR_IN2, LOW);
  ledcWrite(MOTOR_ENA, 0);
}

void stopAndWait() {
  Serial.print("Stopped at position. Waiting ");
  Serial.print(STOP_DURATION / 1000);
  Serial.println(" seconds...");

  unsigned long startTime = millis();
  while (millis() - startTime < STOP_DURATION) {
    checkMotion();
    delay(100);
  }
}

void checkMotion() {
  int pirState = digitalRead(PIR_SENSOR_PIN);

  if (pirState == HIGH) {
    unsigned long currentTime = millis();

    if (currentTime - lastMotionTime > DEBOUNCE_TIME) {
      if (!alertSent) {
        Serial.println("âš ï¸  MOTION DETECTED!");
        sendAlert();
        alertSent = true;
        lastMotionTime = currentTime;
      }
    }
  } else {
    if (millis() - lastMotionTime > DEBOUNCE_TIME) {
      alertSent = false;
    }
  }
}

void sendAlert() {
  if (!wifiConnected) {
    Serial.println("Cannot send alert - WiFi not connected!");
    return;
  }

  // Get position text
```

```
  String positionText = "";
  switch (currentPosition) {
    case LEFT:
      positionText = "LEFT";
      break;
    case CENTER:
      positionText = "CENTER";
      break;
    case RIGHT:
      positionText = "RIGHT";
      break;
  }

  String message = "ALERT! Motion detected at " + positionText + " position.
Time: " + String(millis() / 1000) + "s";

  Serial.println("Sending alert: " + message);

  #ifdef USE_TWILIO
    sendTwilioSMS(message);
  #elif defined(USE_CALLMEBOT_WHATSAPP)
    sendCallMeBotWhatsApp(message);
  #elif defined(USE_CALLMEBOT_SMS)
    sendCallMeBotSMS(message);
  #elif defined(USE_CUSTOM_API)
    sendCustomAPI(message);
  #else
    Serial.println("ERROR: No SMS service configured!");
  #endif
}

// ==================== TWILIO SMS ====================
#ifdef USE_TWILIO
void sendTwilioSMS(String message) {
  WiFiClientSecure client;
  client.setInsecure(); // For testing only - use certificate in production

  HTTPClient https;

  String url = "https://api.twilio.com/2010-04-01/Accounts/" +
String(TWILIO_ACCOUNT_SID) + "/Messages.json";

  if (https.begin(client, url)) {
    // Create authorization header
    String auth = String(TWILIO_ACCOUNT_SID) + ":" + String(TWILIO_AUTH_TOKEN);
    String encodedAuth = base64::encode(auth);

    https.addHeader("Content-Type", "application/x-www-form-urlencoded");
    https.addHeader("Authorization", "Basic " + encodedAuth);

    // Create POST data
    String postData = "To=" + urlEncode(RECIPIENT_PHONE) +
                      "&From=" + urlEncode(TWILIO_PHONE_NUMBER) +
                      "&Body=" + urlEncode(message);

    int httpCode = https.POST(postData);

    if (httpCode > 0) {
      Serial.printf("Twilio Response Code: %d\n", httpCode);
      if (httpCode == HTTP_CODE_OK || httpCode == HTTP_CODE_CREATED) {
        String payload = https.getString();
        Serial.println("â œ SMS sent successfully!");
        Serial.println(payload);
      } else {
```

```
        String payload = https.getString();
        Serial.println("â˜— SMS failed!");
        Serial.println(payload);
      }
    } else {
      Serial.printf("â˜— Twilio Error: %s\n",
https.errorToString(httpCode).c_str());
    }

    https.end();
  } else {
    Serial.println("â˜— Unable to connect to Twilio");
  }
}
#endif

// ==================== CALLMEBOT WHATSAPP ====================
#ifdef USE_CALLMEBOT_WHATSAPP
void sendCallMeBotWhatsApp(String message) {
  HTTPClient http;

  String url = "https://api.callmebot.com/whatsapp.php?phone=" +
               urlEncode(CALLMEBOT_PHONE) +
               "&text=" + urlEncode(message) +
               "&apikey=" + String(CALLMEBOT_API_KEY);

  if (http.begin(url)) {
    int httpCode = http.GET();

    if (httpCode > 0) {
      Serial.printf("CallMeBot Response Code: %d\n", httpCode);
      if (httpCode == HTTP_CODE_OK) {
        Serial.println("â˜‘ WhatsApp message sent!");
      } else {
        Serial.println("â˜— WhatsApp message failed!");
      }
      Serial.println(http.getString());
    } else {
      Serial.printf("â˜— Error: %s\n", http.errorToString(httpCode).c_str());
    }

    http.end();
  }
}
#endif

// ==================== CALLMEBOT SMS ====================
#ifdef USE_CALLMEBOT_SMS
void sendCallMeBotSMS(String message) {
  HTTPClient http;

  String url = "https://api.callmebot.com/sms.php?phone=" +
               urlEncode(CALLMEBOT_SMS_PHONE) +
               "&text=" + urlEncode(message) +
               "&apikey=" + String(CALLMEBOT_SMS_API_KEY);

  if (http.begin(url)) {
    int httpCode = http.GET();

    if (httpCode > 0) {
      Serial.printf("CallMeBot SMS Response: %d\n", httpCode);
      if (httpCode == HTTP_CODE_OK) {
        Serial.println("â˜‘ SMS sent!");
      }
```

```
      Serial.println(http.getString());
    }

    http.end();
  }
}
#endif

// ==================== CUSTOM API (Circuit Digest, etc.) ====================
#ifdef USE_CUSTOM_API
void sendCustomAPI(String message) {
  HTTPClient http;

  // Example for Circuit Digest or generic REST API
  // Modify based on your API documentation

  if (http.begin(CUSTOM_API_URL)) {
    http.addHeader("Content-Type", "application/json");
    http.addHeader("Authorization", "Bearer " + String(CUSTOM_API_KEY));

    // Create JSON payload (modify based on API requirements)
    String jsonPayload = "{\"phone\":\"" + String(CUSTOM_PHONE) +
                         "\",\"message\":\"" + message + "\"}";

    int httpCode = http.POST(jsonPayload);

    if (httpCode > 0) {
      Serial.printf("API Response Code: %d\n", httpCode);
      Serial.println(http.getString());

      if (httpCode == HTTP_CODE_OK || httpCode == HTTP_CODE_CREATED) {
        Serial.println("â Alert sent via custom API!");
      }
    } else {
      Serial.printf("â Error: %s\n", http.errorToString(httpCode).c_str());
    }

    http.end();
  }
}
#endif

// ==================== URL ENCODING ====================
String urlEncode(String str) {
  String encoded = "";
  char c;
  char code0;
  char code1;

  for (int i = 0; i < str.length(); i++) {
    c = str.charAt(i);
    if (c == ' ') {
      encoded += '+';
    } else if (isalnum(c)) {
      encoded += c;
    } else {
      code1 = (c & 0xf) + '0';
      if ((c & 0xf) > 9) {
        code1 = (c & 0xf) - 10 + 'A';
      }
      c = (c >> 4) & 0xf;
      code0 = c + '0';
      if (c > 9) {
        code0 = c - 10 + 'A';
```

```
      }
      encoded += '%';
      encoded += code0;
      encoded += code1;
    }
  }
  return encoded;
}

/*
 * ==================== SETUP INSTRUCTIONS ====================
 *
 * 1. CHOOSE YOUR SMS SERVICE:
 *    Uncomment ONE option at the top of the code
 *
 * 2. TWILIO SETUP (Paid - Most Reliable):
 *    a. Sign up at: https://www.twilio.com/try-twilio
 *    b. Get $15 free credit
 *    c. Get your Account SID and Auth Token from Console
 *    d. Get a Twilio phone number
 *    e. Cost: ~$0.0079 per SMS
 *
 * 3. CALLMEBOT WHATSAPP SETUP (Free):
 *    a. Add CallMeBot to WhatsApp: +34 644 34 48 79
 *    b. Send: "I allow callmebot to send me messages"
 *    c. You'll receive your API key
 *    d. Free but limited to WhatsApp messages
 *    e. Guide: https://www.callmebot.com/blog/free-api-whatsapp-messages/
 *
 * 4. CIRCUIT DIGEST OR CUSTOM API:
 *    a. Check your API provider's documentation
 *    b. Modify the sendCustomAPI() function accordingly
 *    c. Update headers and JSON payload format
 *
 * 5. WIFI CREDENTIALS:
 *    Replace WIFI_SSID and WIFI_PASSWORD with your WiFi details
 *
 * ==================== WIRING (No GSM Module Needed!) ====================
 * - PIR Sensor OUT -> GPIO 13
 * - PIR Sensor VCC -> 5V
 * - PIR Sensor GND -> GND
 * - Motor Driver IN1 -> GPIO 25
 * - Motor Driver IN2 -> GPIO 26
 * - Motor Driver ENA -> GPIO 27
 * - Motor to L298N OUT1 and OUT2
 * - Power supply as per previous wiring guide
 *
 * NO SIM800L REQUIRED! Just WiFi connection needed.
 */
```