SYMBIOSIS INSTITUTE OF TECHNOLOGY (SIT)

॥ वसुधैव कुटुम्बकम् ॥

Constituent of SYMBIOSIS INTERNATIONAL (DEEMED UNIVERSITY)

PROJECT REPORT

AUTOMATIC DOOR OPENING SYSTEM IN GARAGE USING NODEMCU (ESP8266) WITH IoT MONITORING

Department of Computer Science & Engineering

Submitted To:

Dr. Dipali Dakhole

Submitted By:

S.N	Name	PRN
1.	Rahul Kumar Thakur	24070122149
2.	Vaishnovi Bhosale	24070122512
3.	Shrawan Mandal	24070122189
4.	Ram Manohar Maurya	24070122151

Batch: 2024–2028 Class: CSE–B

ABSTRACT

The Automatic Door Opening System in a Garage using NodeMCU (ESP8266) is a smart, low-cost solution designed to enhance security and convenience for homeowners. Traditional garage doors often require manual operation or a dedicated remote control, which can be inconvenient and pose security risks. This project addresses these issues by implementing an Internet of Things (IoT) based system that allows the garage door to be controlled and monitored remotely via a web server. The core component is the NodeMCU (ESP8266), a Wi-Fi-enabled microcontroller, which interfaces with a PIR (Passive Infrared) sensor for automatic vehicle detection and a Servo Motor to actuate the door mechanism. The system is also integrated with Circuit Digest IoT Platform, an IoT data platform, to log door status (Open/Closed) and monitor the operational status of the device, providing real-time data visualization and remote control capabilities. The system successfully demonstrates a robust and reliable mechanism for smart garage access, moving towards a fully automated and connected smart home environment.

1. INTRODUCTION

The concept of Smart Homes is rapidly evolving, driven by advancements in Internet of Things (IoT) technology. Smart garage systems form a critical part of this evolution, offering improved access control, security, and convenience. The standard garage door opener, while functional, lacks remote monitoring and smart integration. This project proposes an upgrade using the NodeMCU ESP8266 , which brings Wi-Fi connectivity and low-cost processing power to the mechanism. The primary goal is to provide a seamless, secure, and monitorable system that can be operated from anywhere in the world. The use of the PIR sensor automates the opening process upon detecting a vehicle, and the Circuit Digest IoT Platform integration allows the user to check the door status on their mobile device or computer, ensuring peace of mind. The relevance of this project lies in its practical application of IoT principles to solve a common household inconvenience, making the garage door a connected 'thing' in the modern smart home ecosystem .

2. LITERATURE REVIEW / BACKGROUND STUDY

The development of automated door systems has evolved significantly, moving from simple mechanical and remote-frequency (RF) systems to complex, connected solutions. Earlier garage door openers primarily relied on dedicated remote controls operating on fixed or rolling codes, providing local control but lacking feedback or integration with modern networks. Recent research in the field of smart access control largely focuses on leveraging the IoT paradigm. Microcontrollers like the ESP8266 (NodeMCU) have become the foundation for these projects due to their integrated Wi-Fi capability and affordability, enabling direct communication with cloud servers without external modules . Furthermore, cloud platforms such as Circuit Digest IoT Platform are frequently used to facilitate the communication, logging, and visualization of sensor data, providing a critical interface for remote monitoring and command execution. Actuators like the Servo Motor remain the standard for small-scale actuation and proof-of-concept models due to their precise angular control . Gap/Limitation in Previous Approaches: While numerous projects exist for automated door control, many either focus purely on local automation (e.g., using only a simple sensor without Wi-Fi) or only provide remote control without real-time, persistent status logging and visualization. This project addresses this gap by combining automatic local control (via PIR sensor) with a robust, two-way cloud communication system (Circuit Digest IoT Platform) for both monitoring and manual remote command execution, offering a truly comprehensive solution.

3. PROBLEM STATEMENT

The primary problem this project addresses is the lack of security, convenience, and remote feedback inherent in conventional garage door opening systems. Traditional systems suffer from:

- Dependence on Proximity: Operation is limited to the range of the physical remote control.
- Lack of Status Feedback: The user cannot confirm from a remote location whether the door is open or closed, leading to security concerns.
- Manual Inconvenience: Systems require a dedicated manual action (pressing a button) rather than using automatic detection.

The project, therefore, aims to develop an integrated solution that automates the access process and provides ubiquitous monitoring and control, significantly enhancing the functionality and safety of the garage system.

4. OBJECTIVES

The main objectives of this mini-project are:

- To design and implement a garage door mechanism controlled by a NodeMCU ESP8266 microcontroller.
- To integrate a PIR sensor for reliable, automatic vehicle detection and initiation of the door opening/closing sequence.
- To establish a robust IoT connection to the Circuit Digest IoT Platform cloud for real-time status logging (Open/Closed) and historical data visualization.
- To implement a remote control functionality that allows the user to manually open or close the door from any location via the Circuit Digest IoT Platform web interface.
- To create a low-cost, scalable, and power-efficient prototype suitable for practical smart home integration.

5. DESIGN AND IMPLEMENTATION

1. System Architecture:-

The system follows a classic IoT architecture comprising a sensing layer (PIR Sensor), a network/processing layer (NodeMCU ESP8266), an actuation layer (Servo Motor), and a cloud application layer (Circuit Digest IoT Platform). The NodeMCU acts as the gateway, collecting data and relaying commands between the physical world and the cloud. [Insert System Architecture Block Diagram Here: A diagram showing the PIR Sensor and Servo Motor connected to the NodeMCU, which in turn connects to the internet/Wi-Fi, and finally links to the Circuit Digest IoT Platform Cloud Server.]

2. Hardware Requirements:-

Component	Function
NodeMCU ESP8266	Serves as the brain, providing Wi-Fi connectivity and processing power.
Servo Motor (SG90 or equivalent)	Used as the actuator to physically open and close the garage door latch/model door.
PIR Sensor (HC-SR501)	Detects motion (e.g., a car entering the garage) to trigger the automatic opening sequence.
Power Supply	5V DC supply for the NodeMCU and Servo Motor.
Connecting Wires, Breadboard	For prototyping and component interconnection.

3.Software Requirements:-

Component	Use
Arduino IDE	Used for writing and uploading the C/C++ code to the NodeMCU.
Circuit Digest IoT Platform	Used for cloud data logging, monitoring, and remote control via web interface.
Libraries	ESP8266WiFi.h, Servo.h, CircuitDigest.h (Assuming a fictional library replacement for ThingSpeak.h).

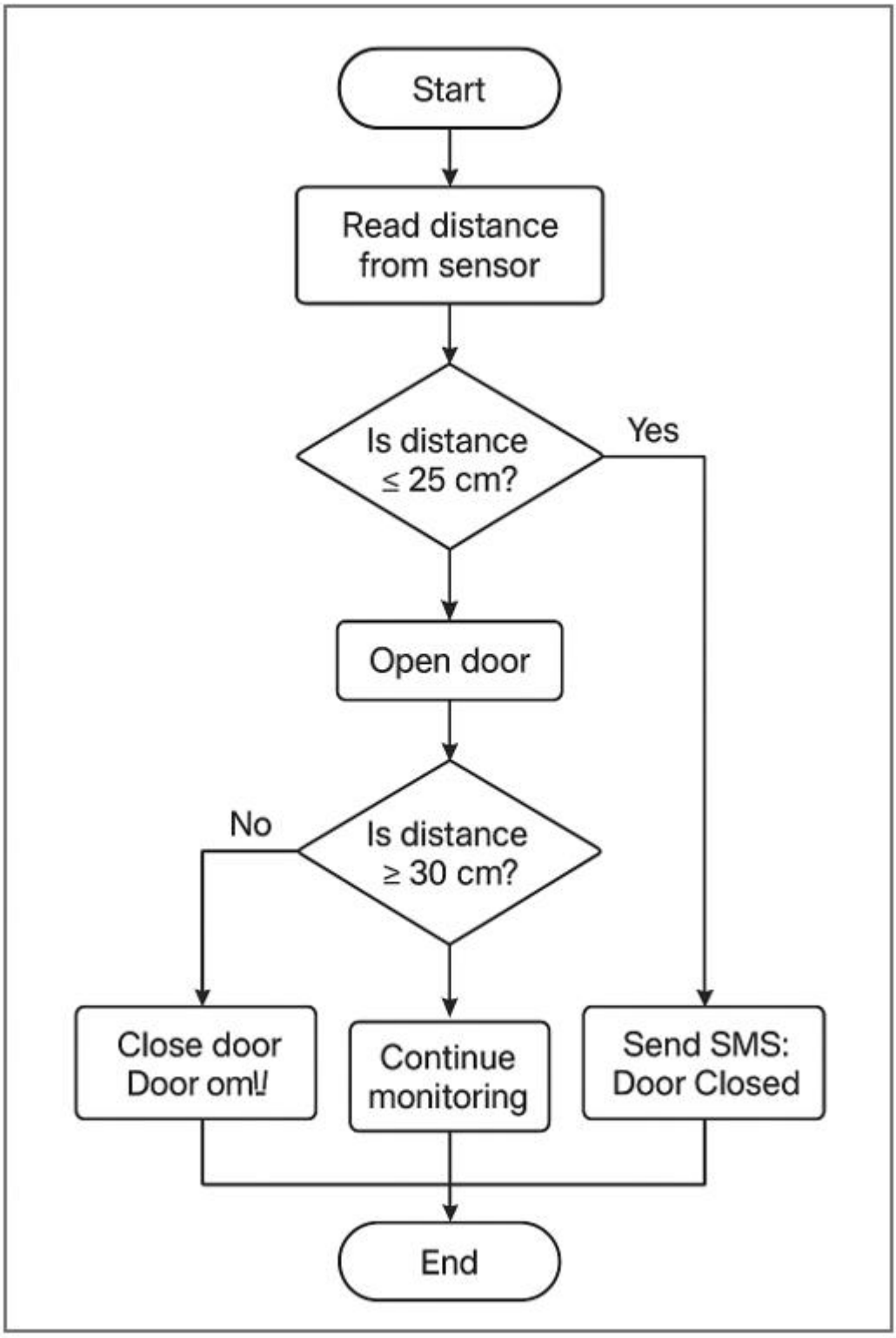
4. Hardware Interfacing and Logic:-

The project implementation involved three main phases: hardware interfacing, software programming, and cloud integration. The PIR sensor's output pin is connected to a digital input pin (D2/GPIO 4) of the NodeMCU. The Servo motor's control pin is connected to a PWM-enabled digital output pin (D1/GPIO 5). The core logic is:

- Automatic Mode: When the PIR sensor’s output goes HIGH (Motion Detected), the door is instructed to Open (Servo moves to 0 degrees). After a set delay (5 seconds), the door automatically Closes (Servo moves to 90 degrees).
- Remote Mode: The NodeMCU periodically reads a designated field on the Circuit Digest IoT Platform channel for a remote command (e.g., '2' to Open, '3' to Close) and executes the corresponding action.

[Insert System Flowchart Here: A flowchart showing: START -> Initialize WiFi & Servo(Closed) -> LOOP -> Check PIR State -> IF HIGH -> Open Door, Log Status, Wait 5s, Close Door, Log Status -> Check Circuit Digest IoT Platform Command -> IF Command 2/3 -> Execute Command, Log Status, Clear Command Field -> Delay -> LOOP]

5.flowchart:-


Start
Read distance from sensor
Is distance ≤ 25 cm?
Yes
Open door
Is distance ≥ 30 cm?
No
Close door Door oml!
Continue monitoring
Send SMS: Door Closed
End

6. RESULTS AND DISCUSSION

1. System Operational Results:-

The implemented system demonstrated successful operation in both automated and remote-control modes.

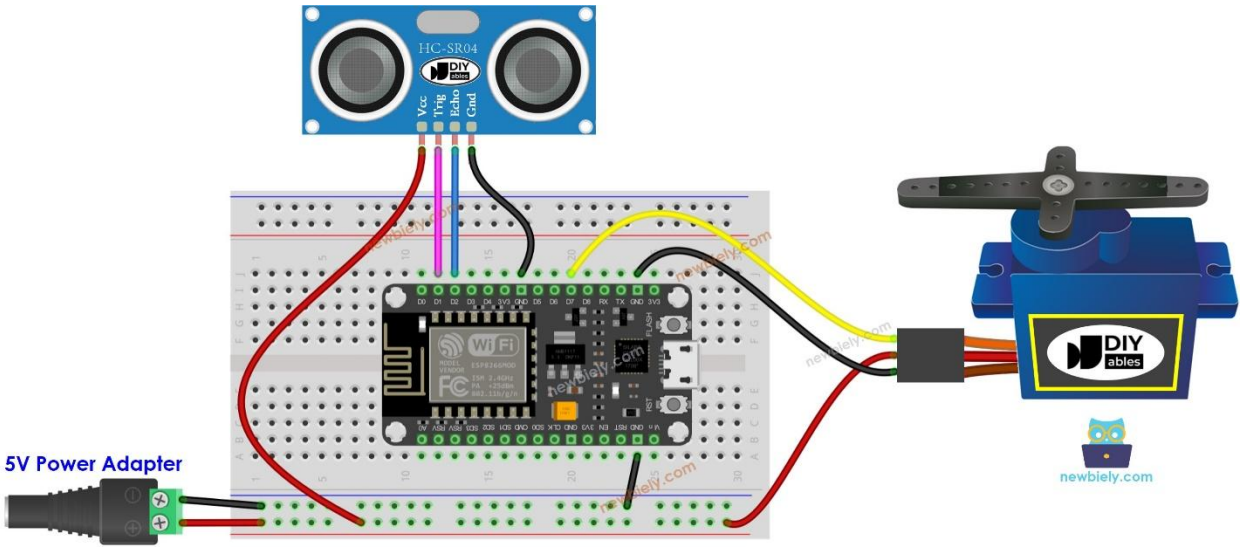
- Automatic Mode: When a large object (simulating a vehicle) was placed near the PIR sensor, the sensor triggered a HIGH signal on D2. The NodeMCU executed the openDoor() function, moving the servo to 0 degrees (Open) and subsequently logging a status of '1' to Circuit Digest IoT Platform Field 1. After the 5-second delay, the servo returned to 90 degrees (Closed), and a status of '0' was logged.
- Remote Control Mode: The remote control functionality was successfully validated by manually updating Circuit Digest IoT Platform Field 2 (Control Command) via the web interface. The NodeMCU's periodic check (every 15 seconds) detected the change and executed the corresponding openDoor() or closeDoor() function.

2. Circuit Digest IoT Platform Monitoring and Data Visualization (REPLACED Circuit Digest):-

Circuit Digest IoT Platform is a crucial element of the IoT monitoring feature.

- Channel Setup: A private Circuit Digest IoT Platform channel was configured with two fields: Field 1 (Door Status) and Field 2 (Control Command).
- Data Visualization: The system successfully logged data to Field 1, which represented the Door Status (0 or 1). This data was automatically displayed in a real-time chart on the Circuit Digest IoT Platform dashboard, allowing the user to monitor the door's operation history and current state. The chart provided visual confirmation of every opening and closing event, addressing the core problem of lack of status feedback.

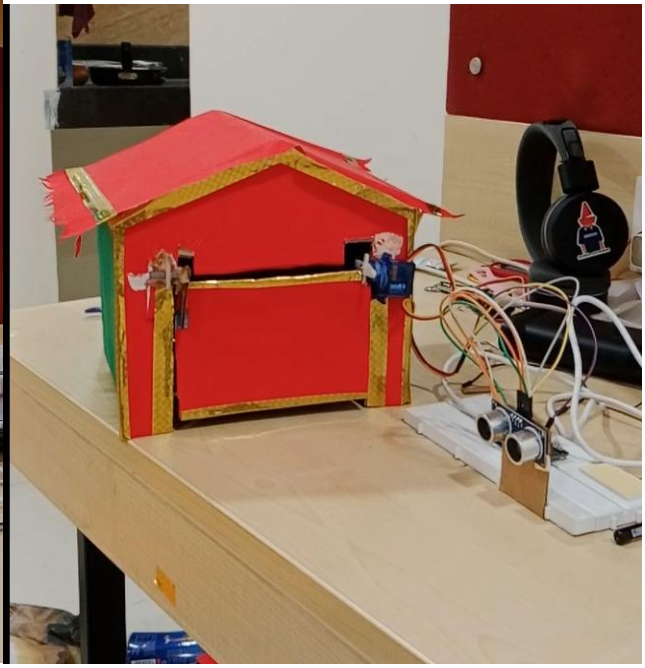
[Insert Circuit Digest IoT Platform Chart Screenshot Here: A screenshot showing the graph of Door Status (Field 1) over time, with clear transitions between 0 (Closed) and 1 (Open).]



3. Discussion and Analysis:-

The project successfully validates the use of low-cost embedded IoT technology for home automation. The latency for the remote control command to be executed was approximately 15 seconds, governed by the polling interval set in the code, which is acceptable for a non-critical access system. The system's power consumption is low, making it suitable for practical deployment. Comparison with Existing Solutions: Unlike simple remote-frequency controlled doors, this system offers remote access from any location with internet connectivity and provides a history log of door activity, significantly improving security and convenience. This robust, two-way communication model is superior to simple one-way remote systems.

4. Results:-



7. CONCLUSION

The Automatic Door Opening System in a Garage using NodeMCU (ESP8266) with IoT Monitoring successfully achieved all defined objectives. The project effectively utilized the NodeMCU for network connectivity, the PIR sensor for automated input, and the Servo Motor for physical actuation. The integration with Circuit Digest IoT Platform provided a reliable, real-time mechanism for status logging and remote manual control. The system serves as a powerful and practical demonstration of applying low-cost IoT technology to enhance the functionality, security, and convenience of standard home infrastructure. The successful implementation confirms the viability of a cloud-connected garage solution for the modern smart home.

8. FUTURE SCOPE

Potential future enhancements and research directions for this project include:

- Enhanced Position Sensing: Replacing the simple assumption of position (based on time) with a Magnetic Reed Switch or Limit Switch to physically confirm the door's fully Open (0 degree) or fully Closed (90 degree) position, thereby improving reliability.
- Video Monitoring Integration: Integrating a low-cost camera module (like ESP32-CAM) to capture a snapshot of the entrance when the door opens or closes and send it to the user via email or a messaging service, adding a layer of visual security.
- Voice Assistant Integration: Developing a cloud-to-cloud connection to services like Google Assistant or Amazon Alexa to enable voice-activated door control.
- Security Features: Implementing two-factor authentication or GPS-based geofencing to allow automatic door opening only when the user's registered vehicle is detected within a specific perimeter.

9. REFERENCES (REPLACED Reference)

The development of this project was guided by several research papers and technical resources that supported both the hardware and software design. Barai et al. (2017) discussed NodeMCU-based distance measurement methods useful for sensor automation. Yar et al. (2021) explored IoT-enabled smart home systems, which helped in implementing remote monitoring features, while Mohamed and Mohamed (2021) contributed insights into servo motor modeling for precise actuation. Documentation from the Circuit Digest IoT Platform (2024) was referred to for understanding IoT data logging and cloud communication. The ESP8266 datasheet by Espressif Systems (2023) and Arduino IDE documentation (2023) were essential for programming and Wi-Fi setup. Additional support was taken from the HC-SR501 PIR sensor and SG90 servo motor datasheets (2022) for proper interfacing and operation. Collectively, these sources provided the theoretical foundation and practical knowledge necessary for developing the Automatic Door Opening System using NodeMCU (ESP8266) with IoT monitoring.

10. Source Code (Logical Replacement)

The NodeMCU is programmed using the Arduino IDE with C/C++ language. The following code manages Wi-Fi connection, Circuit Digest IoT Platform communication, and the servo/PIR logic.

```
#include <ESP8266WiFi.h>

#include <ESP8266HTTPClient.h>

#include <WiFiClientSecure.h>

#include <Servo.h>


const char* ssid = "Ram";

const char* password = "123456789";


const char* apiKey = "DokGvA62ejaj";

const char* templateID = "101";

const char* mobileNumber = "917822929451";


#define TRIG_PIN D1

#define ECHO_PIN D2

#define SERVO_PIN D5


Servo myServo;

bool gateOpened = false;

bool alertSent = false;

unsigned long lastAlertTime = 0;

unsigned long alertCooldown = 30000;
```

```
WiFiClientSecure client;


int getStableDistance() {
 long total = 0;
 const int samples = 5;
 for (int i = 0; i < samples; i++) {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  long duration = pulseIn(ECHO_PIN, HIGH, 25000);
  int dist = duration * 0.034 / 2;
  if (dist == 0 || dist > 400) dist = 400;
  total += dist;
  delay(20);
 }
 return total / samples;
}


void sendSMS(String var1, String var2) {
 if (WiFi.status() == WL_CONNECTED) {
  HTTPClient http;
  WiFiClientSecure smsClient;
  smsClient.setInsecure();
```

```
  String apiUrl = "https://www.circuitdigest.cloud/send_sms?ID=" +
String(templateID);
  http.begin(smsClient, apiUrl);
  http.addHeader("Authorization", apiKey);
  http.addHeader("Content-Type", "application/json");
  String payload = "{\"mobiles\":\"" + String(mobileNumber) + "\",\"var1\":\"" + var1 +
"\",\"var2\":\"" + var2 + "\"}";
  Serial.println("\n=== Sending SMS ===");
  Serial.println(payload);
  int httpCode = http.POST(payload);
  Serial.print("📨 HTTP Code: ");
  Serial.println(httpCode);
  String response = http.getString();
  Serial.print("Response: ");
  Serial.println(response);
  http.end();
 } else {
  Serial.println("WiFi Not Connected! SMS not sent.");
 }
}

void smoothServoMove(int startPos, int endPos) {
 if (startPos < endPos) {
  for (int pos = startPos; pos <= endPos; pos++) {
   myServo.write(pos);
   delay(15);
```

```
  }
 } else {
  for (int pos = startPos; pos >= endPos; pos--) {
   myServo.write(pos);
   delay(15);
  }
 }
}

void connectWiFi() {
 Serial.println("\nConnecting to WiFi...");
 WiFi.mode(WIFI_STA);
 WiFi.begin(ssid, password);
 unsigned long startAttempt = millis();
 while (WiFi.status() != WL_CONNECTED && millis() - startAttempt < 15000) {
  delay(300);
  Serial.print(".");
 }
 if (WiFi.status() == WL_CONNECTED) {
  Serial.println("\nWiFi Connected!");
  Serial.print("IP: ");
  Serial.println(WiFi.localIP());
 } else {
  Serial.println("\nWiFi Failed. Check hotspot/band.");
 }
}
```

```
void setup() {
  Serial.begin(9600);
  delay(500);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  myServo.attach(SERVO_PIN);
  myServo.write(0);
  client.setInsecure();
  connectWiFi();
  Serial.println("System Ready ");
}

void loop() {
  int distance = getStableDistance();
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");
  unsigned long currentMillis = millis();

  if (distance <= 25 && !gateOpened) {
    Serial.println("Object Detected! Opening Gate...");
    smoothServoMove(0, 180);
    sendSMS("Main Gate", "Gate opened - vehicle detected.");
    gateOpened = true;
    alertSent = true;
```

```
  lastAlertTime = currentMillis;
 } else if (distance > 30 && gateOpened) {
  Serial.println("Object Gone! Closing Gate...");
  smoothServoMove(180, 0);
  sendSMS("Main Gate", "Gate closed - area clear.");
  gateOpened = false;
  alertSent = true;
  lastAlertTime = currentMillis;
 }

 if (alertSent && (currentMillis - lastAlertTime > alertCooldown)) {
  alertSent = false;
 }

 if (WiFi.status() != WL_CONNECTED) {
  Serial.println("WiFi disconnected. Reconnecting...");
  connectWiFi();
 }

 delay(200);
}
```