International Journal for Modern Trends in Science and Technology
Volume 12, Issue 07, pages 66-79.
ISSN: 2455-3778 online
Available online at: http://www.ijmtst.com/vol12issue06.html
DOI: https://doi.org/10.5281/zenodo.20739597

Smart Gate: IoT-Enabled RFID Based Vehicle Entry and Real Time Count Management System

Ch. Praveen, Kattula Vyshnavi Devi, Chappidi Rajeev, Kudipudi Siva Teja

Department of Information Technology , Swarnandhra College of Engineering & Technology, Seetharamapuram, Narsapur, Andhra Pradesh, INDIA.

To Cite this Article
Ch. Praveen, Kattula Vyshnavi Devi, Chappidi Rajeev & Kudipudi Siva Teja (2026). Smart Gate: IoT-Enabled RFID Based Vehicle Entry and Real Time Count Management System. International Journal for Modern Trends in Science and Technology, 12(07), 66-79. https://doi.org/10.5281/zenodo.20739597

Article Info
Received: 06 June 2026; Revised: 29 June 2026; Accepted: 01 July 2026.

KEYWORDS

Smart Gate System, Internet of Things (IoT), RFID, IR Sensors, ESP32, Thing Speak, Real-Time Monitoring, Vehicle Counting System

ABSTRACT

The rapid advancement of automation and Internet of Things (IoT) technologies has led to the development of intelligent systems for efficient access control and monitoring. This project presents a Smart Gate system that integrates RFID technology and IR sensors with IoT to automate vehicle entry and exit while maintaining real-time vehicle count management. The system utilizes an RFID module to identify vehicles equipped with RFID tags for secure entry authentication. In addition, IR sensors are employed to detect vehicle movement at both entry and exit points, ensuring accurate counting of vehicles within the premises. Upon successful detection, the gate is automatically controlled using a servo motor, allowing seamless vehicle access. A buzzer provides audible alerts, while an LCD display shows real-time system status and vehicle count.An ESP32 microcontroller acts as the core processing unit, enabling WiFi connectivity for transmitting data to the Thing Speak cloud platform. This allows remote monitoring and visualization of vehicle entry and exit data in real time. The system increases the count during entry and decreases it during exit, providing an accurate representation of the number of vehicles inside the area. The proposed system enhances security, reduces manual effort, and improves efficiency in vehicle access management. It is highly suitable for applications such as parking areas, residential complexes, toll gates, and institutional campuses. Furthermore, the integration of RFID, IR sensors, and IoT makes the system scalable and adaptable for future smart infrastructure developments.

1. INTRODUCTION

In recent years, the rapid advancement of automation technologies and the increasing adoption of the Internet of Things (IoT) have significantly transformed traditional systems into smart and intelligent solutions. One such area that has gained considerable attention is vehicle access control systems, where automation plays a crucial role in enhancing security, efficiency, and

real-time monitoring. Conventional gate management systems often rely on manual supervision, which can lead to inefficiencies, human errors, and security vulnerabilities. These limitations have created a growing demand for automated systems capable of managing vehicle entry with minimal human intervention.

The Smart Gate: IoT-Enabled RFID-Based Vehicle Entry and Real-Time Count Management System is designed to address these challenges by integrating RFID technology with IoT capabilities. This system provides a seamless and automated approach to vehicle identification, gate control, and real-time monitoring. By utilizing RFID tags, each vehicle can be uniquely identified, allowing only authorized access through the gate. The integration of IoT further enhances the system by enabling remote data monitoring and analysis through cloud platforms such as Thing Speak.

The proposed system employs an ESP32 microcontroller as the central processing unit, which coordinates the interaction between different hardware components, including the RFID module, servo motor, LCD display, and buzzer. When a vehicle equipped with an RFID tag approaches the gate, the RFID reader scans the tag and processes the information. Upon successful detection, the system automatically opens the gate using a servo motor and increments the vehicle count. Simultaneously, the updated data is transmitted to the cloud, enabling real-time tracking of vehicle entries.

In addition to automation, the system incorporates user-friendly features such as an LCD display to provide real-time status updates and a buzzer to indicate successful entry. These features improve the usability and effectiveness of the system, making it suitable for various applications such as parking lots, toll booths, residential complexes, and institutional premises. By reducing manual intervention, the system enhances operational efficiency while maintaining a high level of accuracy in vehicle counting.

Furthermore, the integration of IoT allows for continuous monitoring and data analysis, which can be used for decision-making and system optimization.

1.1 INTRODUCTION TO SMART GATE SYSTEMS

The ability to access real-time data smart gate systems represent a modern approach to access control by incorporating automation and intelligent technologies to manage entry and exit operations. Unlike traditional gate systems that require manual operation, smart gate systems utilize sensors, identification technologies, and microcontrollers to automate the process. These systems are designed to enhance security, reduce human effort, and improve overall system efficiency.

The system developed in this project is an example of a smart gate system that leverages RFID technology for vehicle identification. The use of a servo motor for gate control ensures smooth and automated operation, while additional components such as LCD displays and buzzers provide real-time feedback to users. This approach not only enhances security but also ensures a seamless and efficient entry process.

Fig 1.1 RFID Smart Parking System

1.2 OVERVIEW OF RFID TECHNOLOGY

Radio Frequency Identification (RFID) is a wireless communication technology used for automatic identification and data capture. It consists of two main components: an RFID tag and an RFID reader. The RFID tag contains a unique identification number, which is transmitted to the reader using radio frequency signals. The reader then processes this information and sends it to a microcontroller for further action.

RFID technology offers several advantages over traditional identification methods, such as barcodes or manual entry systems. It does not require direct line-of-sight, allowing for faster and more efficient scanning. Additionally, RFID systems are highly reliable and can operate in various environmental

conditions, making them suitable for real-world applications.

1.3 ROLE OF IOT SMART SYSTEMS

The Internet of Things (IoT) refers to the interconnection of physical devices through the internet, enabling them to collect, exchange, and analyze data. IoT plays a crucial role in transforming traditional systems into smart systems by providing real-time monitoring, remote access, and data-driven decision-making capabilities.

In the context of the smart gate system, IoT enables the transmission of vehicle entry data to a cloud platform, allowing users to monitor system activity remotely. The ESP32 microcontroller, with its built-in WiFi capabilities, facilitates seamless communication between the hardware components and the cloud. By using HTTP protocols, the system sends data such as vehicle count to the Thing Speak platform, where it can be visualized and analyzed.

The integration of IoT not only enhances system functionality but also provides scalability for future improvements. For instance, additional features such as mobile notifications, advanced analytics, and integration with other smart systems can be easily implemented. Thus, IoT serves as a key component in making the smart gate system more intelligent, efficient, and adaptable.

1.4 NEED FOR AUTOMATED VECHICLE ACCESS CONTROL

With the increasing number of vehicles and the growing demand for secure and efficient access control systems, traditional manual methods are no longer sufficient. Manual gate management systems are prone to errors, delays, and security risks, as they depend heavily on human intervention. These systems often fail to maintain accurate records of vehicle entries, leading to inefficiencies in management and monitoring.

Automated vehicle access control systems address these challenges by providing a reliable and efficient solution. By using technologies such as RFID and IoT, these systems can automatically identify vehicles, control gate operations, and maintain accurate records in real time. This not only reduces the workload on human operators but also enhances the overall security of the system.

1.5 OBJECTIVES OF THE PROJECT

The main objectives of this project are:

- To design and develop an RFID-based automated gate control system
- To implement secure vehicle identification using RFID technology
- To maintain real-time vehicle count using embedded systems
- To integrate IoT for remote monitoring using Thing Speak
- To enhance system efficiency by reducing manual intervention
- To provide a user-friendly interface using LCD display and buzzer alerts

1.6 SCOPE OF THE PROJECT

The scope of this project includes the design and implementation of an automated vehicle entry system using RFID and IoT technologies. The system is capable of identifying vehicles, controlling gate operations, and maintaining real-time records of vehicle entries. It can be deployed in various environments such as parking areas, residential complexes, toll gates, and institutional premises. In addition to its current functionality, the system can be further expanded to include advanced features such as multi-user authentication, mobile application integration, and real-time alerts. The scalability of the system allows it to adapt to different use cases and requirements, making it a versatile solution for modern access control systems.

1.7 LIMITATIONS OF PROJECT

The proposed RFID-based automated vehicle access control system has certain limitations: The system depends on RFID cards, so if a card is lost, damaged, or not detected properly, access may fail. The reading range of the RFID module is limited, requiring the vehicle to be very close to the reader. The system requires a stable internet connection for IoT functionality; without it, real-time monitoring and data updates to the cloud may not work. Security risks may arise if RFID cards are duplicated or unauthorized access occurs.

2. LITERATURE SURVEY

LITERATURE REVIEW

The rapid development of automation, embedded systems, and Internet of Things (IoT) technologies has significantly influenced modern access control and

monitoring systems. In recent years, researchers and engineers have explored various methods to improve vehicle entry management systems by integrating intelligent technologies such as RFID, wireless communication, and cloud computing. This chapter presents a review of existing research and systems related to RFID-based access control, IoT-enabled monitoring, and automated gate management systems.

The primary goal of this literature review is to analyze existing approaches, identify their strengths and limitations, and highlight the need for an improved system that combines automation with real-time data monitoring. The integration of RFID and IoT technologies has proven to be effective in enhancing system efficiency, accuracy, and security in vehicle access control applications.

2.1 RFID-BASED ACCESS CONTROL SYSTEM

RFID-based access control systems have been widely used in various applications such as parking management, toll collection, security systems, and inventory tracking. These systems rely on RFID tags and readers to automatically identify and authenticate users or vehicles. The use of radio frequency signals enables quick and contactless communication between the tag and the reader, making the process efficient and reliable.

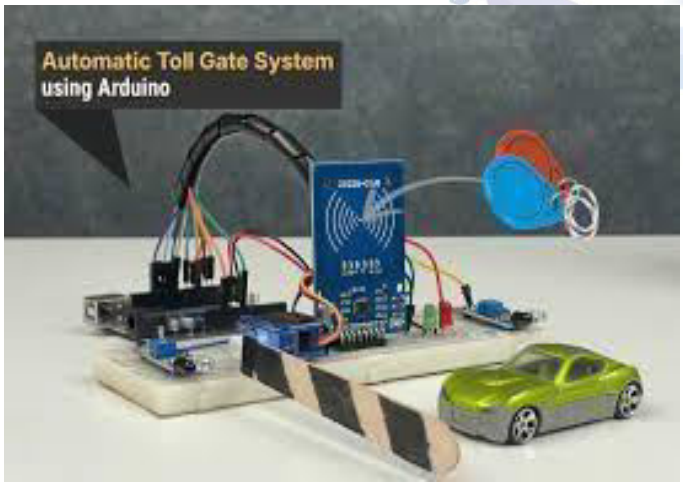

Fig 2.1 RFID Based Access Control System

Several studies have demonstrated the effectiveness of RFID technology in access control systems. Researchers have developed RFID-based parking systems where vehicles are automatically identified at entry points, eliminating the need for manual ticketing. These systems significantly reduce waiting time and improve traffic flow. Additionally, RFID-based toll collection systems have been implemented in highways to enable automatic fee deduction without stopping vehicles, thereby reducing congestion.

2.2 COMPARATIVE ANAYSIS TABLE

The introduction of IoT has revolutionized traditional systems by enabling real-time data collection, remote monitoring, and intelligent decision-making. IoT-based smart parking systems have gained significant attention in recent years, as they provide efficient solutions for managing vehicle parking and reducing congestion in urban areas.

In IoT-enabled parking systems, sensors and microcontrollers are used to detect vehicle presence and transmit data to cloud platforms. Users can access this information through web or mobile applications, allowing them to monitor parking availability in real time. These systems improve resource utilization and reduce the time spent searching for parking spaces.

Researchers have also explored the integration of IoT with access control systems to enhance functionality. By connecting devices such as RFID readers and microcontrollers to the internet, it becomes possible to store and analyze data remotely. Platforms like ThingSpeak and other cloud services are commonly used for data visualization and analysis.

2.3 AUTOMATED GATE CONTROL SYSTEMS

Automated gate control systems are designed to manage the opening and closing of gates without manual intervention. These systems are commonly used in residential areas, commercial buildings, and industrial facilities to enhance security and convenience. Various technologies, including sensors, remote controls, and identification systems, have been used to automate gate operations.

Early automated gate systems relied on simple mechanisms such as remote controls or keypads for operation. While these methods provided basic automation, they lacked advanced features such as user authentication and real-time monitoring. To overcome these limitations, modern systems incorporate technologies such as RFID, biometric authentication, and wireless communication.

Although automated gate systems improve convenience and security, many existing implementations do not include data tracking or remote

monitoring features. This limits their ability to provide insights into system usage and performance.

2.4 CLOUD -BASED DATA MONITORING USING THINKSPEACK

Cloud computing has emerged as a powerful tool for storing, processing, and analyzing data generated by IoT devices. Platforms such as Thing Speak provide an easy-to-use interface for collecting and visualizing data in real time. These platforms support various communication protocols, enabling seamless integration with microcontrollers and sensors.

Thing Speak is widely used in IoT applications due to its simplicity and ability to display data in graphical formats. It allows users to monitor parameters such as temperature, humidity, and system activity from anywhere using the internet. Researchers have utilized Thing Speak in projects involving environmental monitoring, healthcare systems, and smart agriculture.

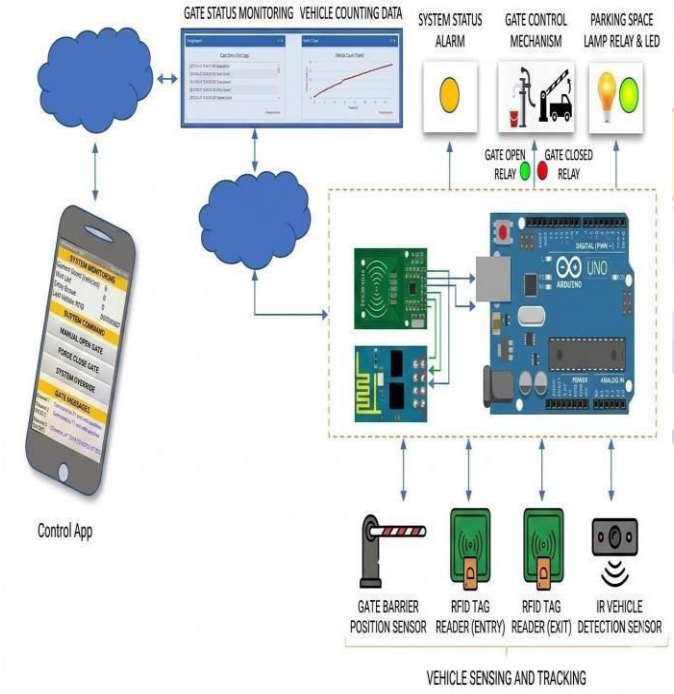

Fig 2.2 Working of Think Speak

In the context of vehicle access systems, cloud platforms can be used to store entry logs, monitor vehicle count, and analyze traffic patterns. This provides valuable insights that can be used for decision-making and system optimization. The ability to access data remotely enhances system usability and flexibility.

However, cloud-based systems require stable internet connectivity and proper configuration to ensure reliable operation. Data privacy and security are also important considerations, as sensitive information may be transmitted over the network.

2.5 LIMITATIONS OF EXISTING SYSTEMS

Although significant advancements have been made in RFID-based and IoT-enabled systems, several limitations still exist in current implementations. One of the major drawbacks is the lack of integration between identification systems and real-time monitoring platforms. Many systems operate independently without providing centralized data access.

Another limitation is the absence of accurate vehicle counting mechanisms in some access control systems. While RFID technology can identify vehicles, it may not always be used to maintain proper entry records. This results in incomplete data and reduces the effectiveness of the system.

Additionally, many existing systems are not user-friendly and lack proper feedback mechanisms. The absence of display units or alert systems can make it difficult for users to understand system status. Furthermore, some systems are not scalable and cannot be easily expanded to accommodate additional features or larger deployments.

Security is another critical concern, as unauthorized access and data breaches can compromise system integrity. Without proper authentication and encryption mechanisms, systems may be vulnerable to attacks.

3. PROPOSED METHODOLOGY

The proposed system integrates multiple sensing and control components to ensure reliable and intelligent gate operation. The RFID reader scans the unique identification number embedded in each RFID tag attached to authorized vehicles. When a valid tag is detected, the ESP32 verifies the authentication data and triggers the servo motor to open the gate smoothly. If an unauthorized tag is scanned, the system activates the buzzer alert and denies access, thereby enhancing security and preventing unauthorized vehicle entry.

To accurately differentiate between entry and exit operations, the system incorporates IR sensors placed strategically near the gate. These sensors detect vehicle movement and help determine whether a vehicle is entering or leaving the premises. Based on the sensor

input, the ESP32 updates the vehicle count accordingly. This dual-sensor mechanism minimizes counting errors and ensures precise occupancy management even during high traffic conditions.

An LCD display module is used to provide real-time information to users at the gate. It displays messages such as "Access Granted," "Access Denied," and the current vehicle count. This immediate visual feedback improves user interaction and transparency. The buzzer further enhances the user experience by providing audible alerts for successful authentication, unauthorized access attempts, or system errors.

The integration with the ThingSpeak cloud platform enables remote monitoring and data analytics. The ESP32 sends vehicle count data periodically through Wi-Fi, allowing administrators to track occupancy levels in real time from anywhere. The cloud dashboard can visualize data using graphs and charts, helping in trend analysis, peak hour identification, and efficient space management. This feature makes the system scalable and suitable for smart city applications.

The system is designed with energy efficiency and reliability in mind. The ESP32 consumes low power while providing high processing capability and built-in Wi-Fi functionality. The hardware components are compact and cost-effective, making the overall solution affordable for small- to medium-scale deployments. Additionally, the modular design allows easy maintenance, upgrades, and integration with additional security features such as cameras or biometric systems.

Overall, the IoT-enabled RFID-based smart gate system provides a modern, automated, and intelligent solution for vehicle access management. By combining authentication, real-time monitoring, cloud connectivity, and automated control, it enhances security, reduces human intervention, and ensures accurate occupancy tracking. This system represents a practical step toward digital transformation in access control and smart infrastructure management.

3.1 SYSTEM ARCHITECTURE

The system architecture of the RFID-based smart gate is structured into four major layers: input, processing, communication, and output. This layered architecture ensures clear separation of responsibilities among components and improves system reliability, scalability, and maintainability. Each layer performs a specific function while working in coordination with the others to enable smooth and automated gate operation.

The input layer serves as the data acquisition unit of the system. It consists primarily of the RFID reader and IR sensors. The RFID reader captures the unique identification number from the RFID tag attached to each authorized vehicle. Meanwhile, the IR sensors detect vehicle movement and determine whether a vehicle is entering or exiting the premises. Together, these components provide accurate and real-time input data to the system.

The RFID reader plays a critical role in secure access control. When a vehicle approaches the gate, the reader scans the RFID tag wirelessly using radio frequency signals. The captured tag ID is transmitted to the ESP32 microcontroller for validation. This contactless technology ensures quick response time, minimal wear and tear, and reduced human intervention.

The IR sensors complement the RFID system by enabling direction detection and vehicle counting. Positioned strategically at the entry and exit points, these sensors detect motion and signal the system when a vehicle passes through the gate. By analyzing the sequence of sensor triggers, the system determines whether to increment or decrement the vehicle count, ensuring precise occupancy tracking.

The processing layer forms the core intelligence of the architecture. It is centered around the ESP32 microcontroller, which acts as the central control unit. The ESP32 processes data received from the RFID reader and IR sensors, executes authentication algorithms, updates the vehicle count, and controls output devices accordingly. Its built-in Wi-Fi capability and high processing speed make it ideal for IoT-based applications.

Within the processing layer, logical decision-making takes place. The ESP32 checks whether the scanned RFID tag matches stored authorized IDs. If validation is successful, it triggers the servo motor to open the gate; otherwise, it denies access and activates the buzzer. Additionally, it ensures synchronization between entry/exit detection and count updates to avoid inconsistencies.

The communication layer enables remote connectivity and data sharing. Using the built-in Wi-Fi module of the ESP32, the system connects to the internet and sends vehicle count data to the ThingSpeak cloud platform.

Data transmission is performed using HTTP requests at regular intervals or upon specific events, ensuring that real-time information is available for monitoring.

The cloud integration enhances the system's capability by providing data storage, visualization, and analytics. Through the ThingSpeak dashboard, administrators can view live vehicle occupancy, historical trends, and peak traffic patterns. This remote monitoring feature makes the system suitable for smart parking management, institutional campuses, and commercial complexes.

The output layer translates processed decisions into physical actions and user feedback. It includes the servo motor, which controls the gate's opening and closing mechanism; the LCD display, which shows status messages and vehicle count; and the buzzer, which provides audible alerts. Together, these output components ensure clear communication with users and efficient gate automation, completing the end-to-end smart gate architecture.

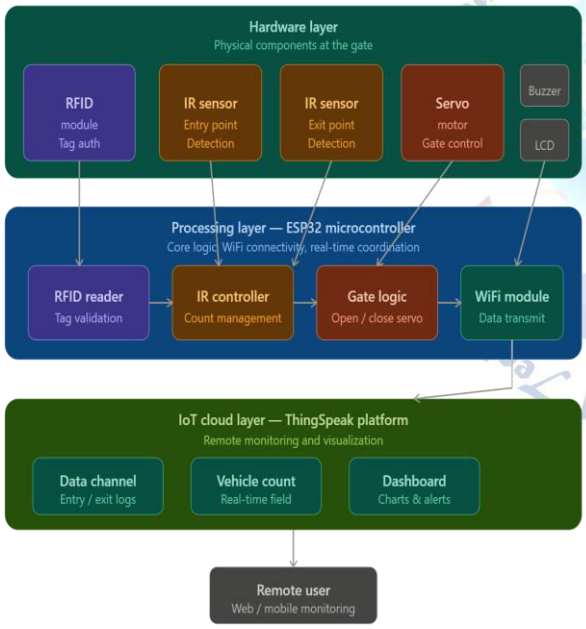

Fig 3.2 System Architecture

The **System Architecture** of the IoT-enabled RFID Smart Gate. It explains how the system is structured in multiple layers — starting from physical hardware components, moving to the ESP32 processing unit, and finally connecting to the ThingSpeak IoT cloud platform for remote monitoring.

The second diagram represents the **Use Case Diagram**, which focuses on how external actors such as the Vehicle/Driver and the System Administrator interact with the smart gate system and its various functionalities.

Together, these diagrams describe both the structural design and the functional behavior of the proposed system.

System Architecture

1. Hardware Layer

The **Hardware Layer** consists of all physical components that interact directly with the vehicle and the environment. These components include:

- **RFID Module** – Used to scan and read the RFID tag attached to authorized vehicles.
- **Two IR Sensors** – One sensor is placed at the entry point and the other at the exit point to detect vehicle movement and direction.
- **Servo Motor** – Controls the physical opening and closing of the gate.
- **Buzzer** – Generates audible alerts for successful access or unauthorized attempts.
- **LCD Display** – Shows real-time messages such as access status and vehicle count.

This layer is responsible for collecting input data and providing physical output actions.

2. Processing Layer (ESP32 Controller)

The **Processing Layer** is managed by the ESP32 microcontroller, which acts as the central control unit of the system.

The ESP32 performs the following operations:

- Receives input signals from the RFID reader and IR sensors
- Validates the RFID tag against authorized records
- Determines whether the vehicle is entering or exiting
- Increments or decrements the vehicle count
- Controls the servo motor to open or close the gate
- Activates the buzzer when required
- Updates the LCD display with status messages
- Prepares and formats data for cloud transmission

This layer ensures logical decision-making and coordination between all hardware components.

3. IoT Cloud Layer (ThingSpeak Platform)

The **IoT Cloud Layer** is implemented using the ThingSpeak platform.

After processing vehicle entry or exit data, the ESP32 sends updated vehicle count information to ThingSpeak via Wi-Fi using HTTP requests. The cloud platform stores and visualizes this data.

Through the ThingSpeak dashboard, the system administrator or authorized user can:

- Monitor real-time vehicle count
- View graphical data trends
- Analyze entry and exit patterns
- Access historical data records

This layer enables remote monitoring and enhances the smart functionality of the gate system.

Architecture Flow

Step 1: Vehicle Detection

The process begins when a vehicle approaches the gate. The IR sensor detects movement and signals the ESP32 that a vehicle is present near the entry or exit point.

Step 2: RFID Tag Reading

Once movement is detected, the RFID module scans the RFID tag attached to the vehicle. The tag ID is transmitted to the ESP32 for authentication.

Step 3: Authentication and Validation

The ESP32 checks whether the scanned tag matches the list of authorized vehicle IDs.

- If valid → Access is granted.
- If invalid → Access is denied and the buzzer is activated.

Step 4: Gate Operation

If authentication is successful, the ESP32 sends a control signal to the servo motor to open the gate.

After the vehicle passes through (confirmed by the IR sensor), the gate automatically closes to maintain security.

Step 5: Vehicle Count Update

Based on whether the vehicle is entering or exiting, the ESP32 updates the vehicle count:

- Entry → Count is incremented
- Exit → Count is decremented

The updated count is displayed on the LCD screen.

Step 6: Cloud Data Transmission

Finally, the ESP32 connects to Wi-Fi and sends the updated vehicle count and transaction data to the ThingSpeak cloud platform.

The data is stored and displayed in graphical form for remote access by the administrator.

3.2 USE CASE DIAGRAM

The use case diagram of the smart gate system illustrates how different actors interact with the automated gate infrastructure. The primary actors in this system include the Vehicle (with RFID tag), the System Administrator, and the Cloud Platform. Each actor plays a specific role in ensuring the proper functioning of the smart gate. The diagram visually represents these interactions and helps in understanding the overall system workflow in a structured manner.

The first major use case is **Scan RFID Tag**, which is initiated when a vehicle approaches the gate. The RFID reader captures the unique ID from the RFID tag attached to the vehicle. This process occurs automatically without requiring manual intervention, making the entry and exit process smooth and efficient. The scanned data is then forwarded to the ESP32 microcontroller for further processing.

The second important use case is **Authenticate Vehicle**. In this step, the ESP32 compares the scanned RFID tag ID with the stored database of authorized vehicle IDs. If the ID matches, authentication is successful; otherwise, access is denied. This authentication mechanism ensures that only registered and authorized vehicles can enter or exit the premises, thereby strengthening security.

Once authentication is successful, the **Open Gate** use case is triggered. The ESP32 sends a control signal to the servo motor, which opens the gate automatically. After a predefined delay or once the IR sensor detects that the vehicle has passed, the **Close Gate** use case is executed. This ensures proper gate control and prevents unauthorized vehicles from tailgating.

Another critical use case is **Update Vehicle Count**. Based on whether the vehicle is entering or exiting, the system increments or decrements the vehicle count. IR sensors help determine the direction of movement. This functionality ensures accurate tracking of the number of vehicles currently inside the premises and prevents overcrowding.

The system also includes the **Display Status Message** use case. The LCD screen provides real-time updates such as "Access Granted," "Access Denied," "Gate Opening," and the current vehicle count. This feature enhances user awareness and improves system transparency by clearly communicating the system status.

In cases of unauthorized access or system errors, the **Activate Alert** use case is executed. The buzzer generates an audible warning signal to alert security personnel or nearby users. This immediate response mechanism adds

an extra layer of protection and helps in quickly identifying suspicious activity.

The **Send Data to Cloud** use case ensures remote connectivity and monitoring. The ESP32 transmits updated vehicle count data and system status information to the ThingSpeak cloud platform via Wi-Fi. This enables real-time data logging, storage, and visualization, which can be accessed from anywhere through an internet-connected device.

Finally, the **Remote Monitoring by Admin** use case allows administrators or authorized users to observe live vehicle occupancy data through the cloud dashboard. They can analyze trends, check historical data, and make informed decisions regarding parking management and security control. Together, these use cases define the complete functional behaviour of the smart gate system and demonstrate how different components interact seamlessly within the smart gate environment.

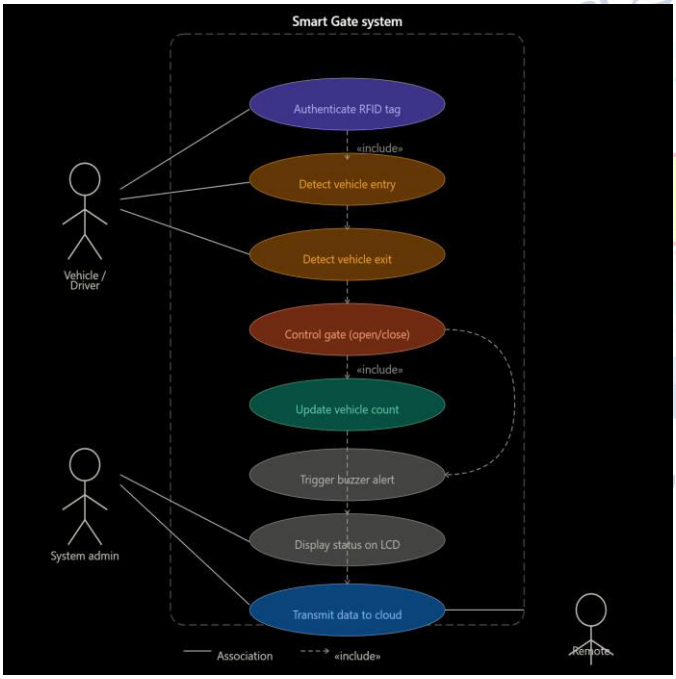

Fig 3.3 Use Case Diagram

The Use Case Diagram represents the functional behavior of the smart gate system by showing how external actors interact with the system. It focuses on *what the system does* rather than how it is internally structured. This diagram helps in understanding user interactions, system responses, and overall operational flow.

3.2.1 Actors in the System

1. Vehicle / Driver

The Vehicle (with RFID tag) or Driver is the primary actor. This actor interacts directly with the gate system during entry and exit. The interaction occurs automatically when the RFID tag is scanned and sensors detect movement.

2. System Administrator

The System Administrator is responsible for monitoring vehicle count and system performance through the cloud platform. The admin does not physically interact with the gate but monitors and analyzes data remotely.

Main Use Cases

1. Scan RFID Tag

This use case is initiated when a vehicle approaches the gate. The RFID reader scans the RFID tag attached to the vehicle. The system captures the tag ID and forwards it to the ESP32 for verification.

Actor involved: Vehicle/Driver

System response: Read tag and send for validation

2. Authenticate Vehicle

After scanning, the system checks whether the tag ID is authorized.

- If the tag is valid → Access is granted
- If the tag is invalid → Access is denied

This use case ensures secure access control.

Actor involved: System (internal processing)

Outcome: Valid or invalid access decision

3. Open Gate

If authentication is successful, the system triggers the servo motor to open the gate. This action allows the vehicle to enter or exit.

Actor involved: Vehicle/Driver

System response: Servo motor rotates to open gate

4. Close Gate

After the vehicle passes the gate (detected by IR sensor), the system automatically closes the gate to maintain security and prevent unauthorized entry.

Actor involved: System

System response: Gate closes automatically

5. Update Vehicle Count

This use case updates the total number of vehicles inside the premises.

- Entry → Increment count
- Exit → Decrement count

The IR sensors help determine the direction of movement.

Actor involved: System

System response: Accurate occupancy tracking

6. Display Status Message

The LCD display shows real-time information such as:

- "Access Granted"
- "Access Denied"
- Current vehicle count

This keeps the driver informed about the system status.

Actor involved: Vehicle/Driver

System response: Visual feedback

7. Activate Alert

If unauthorized access is detected, the buzzer generates an alert sound. This warns security personnel or nearby individuals.

Actor involved: System

System response: Audible alert

8. Send Data to Cloud

After processing entry or exit, the system sends updated vehicle count data to the ThingSpeak cloud platform using Wi-Fi.

Actor involved: System

System response: Data transmission via IoT

9. Monitor Data Remotely

The System Administrator logs into the ThingSpeak dashboard to view live vehicle count, historical data, and graphical reports.

Actor involved: System Administrator

System response: Real-time monitoring and visualization

Use Case Flow (Entry Scenario Example)

1. Vehicle approaches gate
2. IR sensor detects movement
3. RFID tag is scanned
4. System authenticates tag
5. Gate opens (if valid)
6. Vehicle passes through
7. Gate closes
8. Vehicle count updates
9. Data sent to cloud
10. Admin can monitor dashboard

3.3 CLASS DIAGRAM

The class diagram of the smart gate system provides a clear representation of the structural blueprint of the application. It identifies all the major classes, their attributes, methods, and the relationships between them. This object-oriented structure ensures modularity, reusability, and maintainability of the system. By organizing the project into well-defined classes, the system logic becomes easier to understand, implement, and extend in the future.

The **SmartGateSystem** class acts as the main coordinating class of the application. It contains attributes such as systemStatus and vehicleCount, which represent the operational state of the system and the current number of vehicles inside the premises. Its methods, including initializeSystem(), processEntry(), processExit(), updateCount(), and sendToCloud(), define the overall workflow of the smart gate. This class integrates all other components and ensures smooth interaction between them.

The **RFIDReader** class is responsible for handling tag scanning and validation. It includes attributes like tagID and readerStatus to store the scanned tag information and operational condition of the reader. Methods such as scanTag(), readTag(), and validateTag() manage the process of detecting and verifying vehicle identity. This class ensures secure authentication before granting access to vehicles.

The **IRSensor** class manages vehicle detection and movement tracking. With attributes like sensorStatus and sensorType, it identifies whether the sensor is active and whether it is placed at entry or exit points. The method detectVehicle() captures real-time motion data, which is crucial for determining vehicle direction and updating the count accurately.

The **ESP32Controller** class represents the core processing unit of the system. It contains attributes such as wifiStatus and processingState to track network connectivity and internal operations. Its methods — processInput(), controlGate(), connectWiFi(), and uploadData() — handle logical decision-making, device coordination, and IoT communication. This class acts as the brain of the entire smart gate system.

The **GateMotor** class controls the physical gate mechanism. Its attributes, including gateStatus and angle, represent whether the gate is open or closed and the rotation position of the servo motor. Methods like openGate() and closeGate() perform the mechanical operations required for vehicle access. This abstraction separates hardware control logic from the main system logic.

The **LCDDisplay** class is responsible for visual communication with users. It contains the attribute displayMessage to store the message currently shown on the screen. Methods such as showStatus() and showCount() ensure that users receive real-time information regarding authentication results and vehicle occupancy levels. This enhances usability and transparency.

The **Buzzer** class manages audible notifications within the system. The attribute buzzerStatus indicates whether the buzzer is active. Methods like beepEntry(), beepExit(), and alert() generate different sound signals for successful entry, exit confirmation, or unauthorized access attempts. This class improves system responsiveness and security awareness.

The **CloudService** class handles IoT integration and remote monitoring features. It includes attributes such as apiKey and serverURL to store authentication credentials and cloud endpoint details. Methods like sendData(), updateField(), and visualizeData() enable communication with the ThingSpeak platform. This class ensures that real-time vehicle count data is uploaded and available for remote analysis.

Finally, the **Vehicle** class represents the external entity interacting with the system. It includes attributes like vehicleID and rfidTag to uniquely identify each vehicle. Methods such as requestEntry() and requestExit() simulate the vehicle's interaction with the RFID reader and IR sensor. The relationships among these classes demonstrate aggregation and association, where the SmartGateSystem uses other component classes, and the ESP32Controller manages device coordination. Together, the class diagram reflects a well-structured object-oriented design that supports sensing, processing, actuation, and IoT communication in the smart gate system.

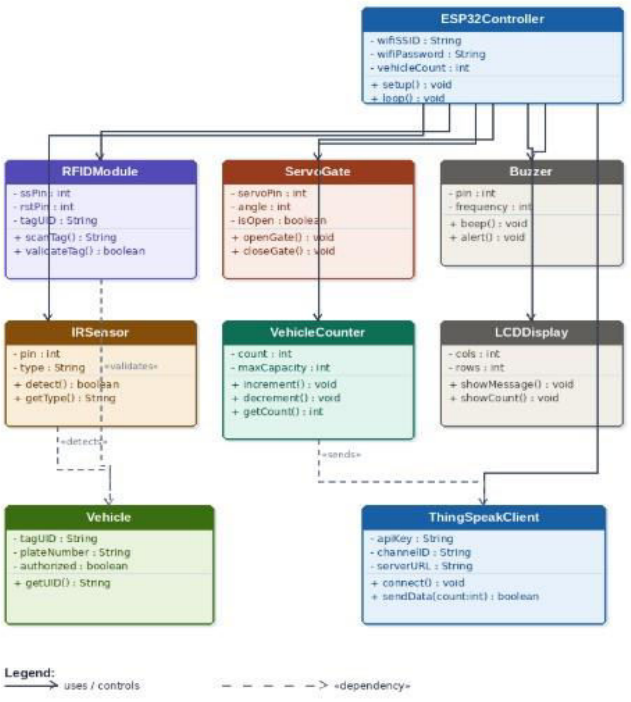

Fig 3.4 Class Diagram

The **Class Diagram** represents the structural design of the smart gate system. It shows the main classes, their attributes, methods, and the relationships between them. Unlike the use case diagram (which explains behavior), the class diagram explains the internal software structure and object-oriented organization of the system.

3.3.1 Main Classes in the System

The smart gate system consists of the following primary classes:

- SmartGateSystem
- RFIDReader
- IRSensor
- ESP32Controller
- GateMotor
- LCDDisplay
- Buzzer
- CloudService
- Vehicle

Each class has specific responsibilities and works together to achieve complete gate automation.

3.3.2 Core Control Class

3.3.2.1 SmartGateSystem Class

This is the main coordinating class of the system.

Attributes:

- systemStatus
- vehicleCount

Methods:

- initializeSystem()
- processEntry()
- processExit()
- updateCount()
- sendToCloud()

This class integrates all other classes and controls the overall workflow of the smart gate system.

3.3.3 Input-Related Classes

3.3.3.1 RFID Reader Class

Handles RFID tag scanning and validation.

Attributes:

- tagID
- readerStatus

Methods:

- scanTag()
- readTag()
- validateTag()

It communicates with the ESP32Controller to confirm whether a vehicle is authorized.

3.3.3.2 IRSensor Class

Responsible for detecting vehicle movement.

Attributes:

- sensorStatus
- sensorType (Entry / Exit)

Methods:

- detectVehicle()

It helps determine vehicle direction and supports accurate vehicle counting.

3.3.4 Processing Class

3.3.4.1 ESP32Controller Class

Represents the central processing unit.

Attributes:

- wifiStatus
- processingState

Methods:

- processInput()
- controlGate()
- connectWiFi()
- uploadData()

This class performs authentication logic, controls outputs, and manages cloud communication.

3.3.5 Output-Related Classes

3.3.5.1 GateMotor Class

Controls the physical gate mechanism.

Attributes:

- gateStatus
- angle

Methods:

- openGate()
- closeGate()

3.3.5.2 LCDDisplay Class

Displays system messages.

Attribute:

- displayMessage

Methods:

- showStatus()
- showCount()

3.3.5.3 Buzzer Class

Generates alert sounds.

Attribute:

- buzzerStatus

Methods:

- beepEntry()
- beepExit()
- alert()

3.3.6 IoT Communication Class

3.3.6.1 CloudService Class

Manages cloud data transmission.

Attributes:

- apiKey
- serverURL

Methods:

- sendData()
- updateField()
- visualizeData()

This class ensures communication between the ESP32 and the ThingSpeak cloud platform.

3.3.7 External Entity Class

3.3.7.1 Vehicle Class

Represents vehicles interacting with the system.

Attributes:

- vehicleID
- rfidTag

Methods:

- requestEntry()
- requestExit()

The vehicle interacts with the RFIDReader and IRSensor during entry and exit.

3.3.8 Relationships Between Classes

3.3.8.1 Association Relationships

- SmartGateSystem **uses** RFIDReader
- SmartGateSystem **uses** IRSensor
- SmartGateSystem **uses** ESP32Controller
- SmartGateSystem **uses** GateMotor
- SmartGateSystem **uses** LCDDisplay
- SmartGateSystem **uses** Buzzer
- SmartGateSystem **uses** CloudService

These associations show that the SmartGateSystem coordinates all components.

3.3.8.2 Control Relationship

The ESP32Controller manages:

- GateMotor operations
- LCDDisplay updates
- Buzzer alerts
- CloudService communication

This indicates strong control dependency.

3.3.**8.3 Interaction Relationship**

The Vehicle interacts with:

- RFIDReader (for authentication)
- IRSensor (for entry/exit detection)

This represents external interaction with the system.

3.3.9 Object-Oriented Design Benefits

The class diagram follows object-oriented principles such as:

- **Encapsulation** – Each class handles its own data and behavior
- **Modularity** – Components are separated for clarity
- **Reusability** – Individual classes can be reused in other IoT systems
- **Maintainability** – Easy to modify or upgrade components

The Class Diagram provides a clear structural representation of the smart gate system. It defines how sensing components, processing units, output devices, and cloud services are organized in an object-oriented manner.

While the system architecture shows hardware-layer communication and the use case diagram shows user interactions, the class diagram explains the internal software structure that enables automation, security, vehicle counting, and IoT-based monitoring in the RFID Smart Gate System.

RESULTS

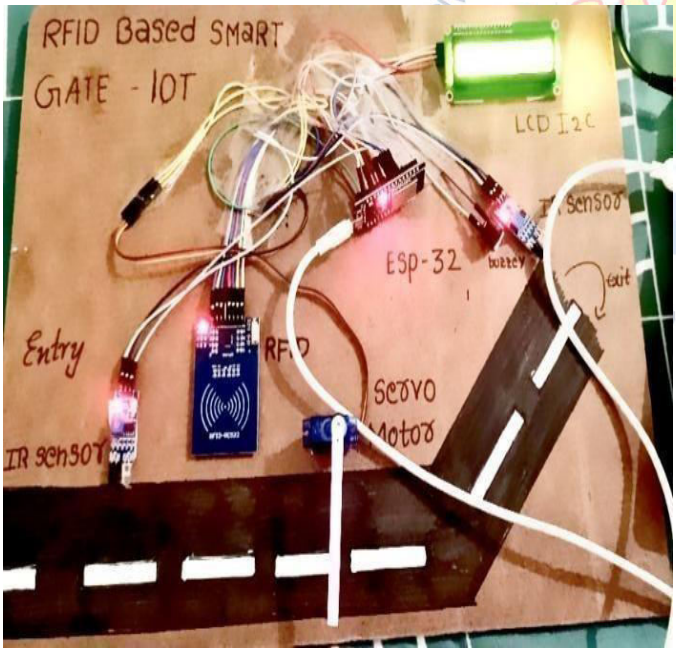

Fig 4.1 Working of Smart Gate System

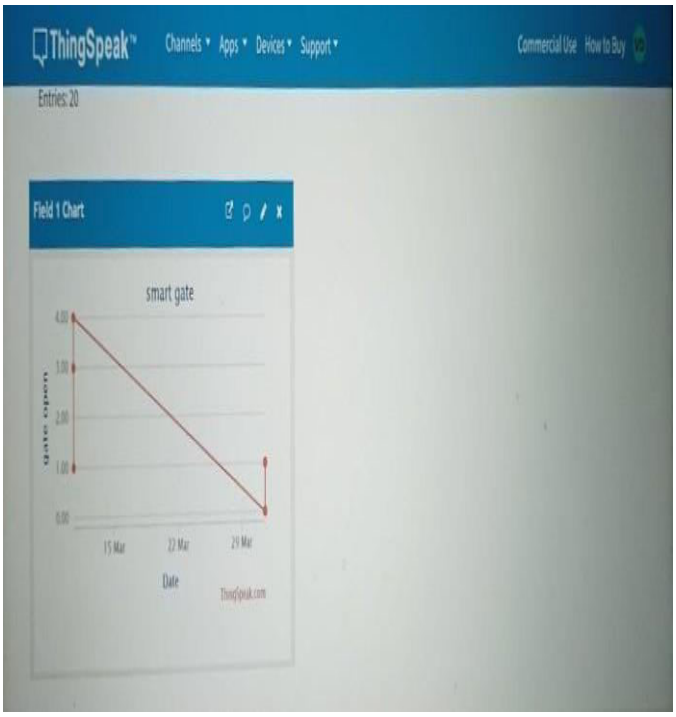

Fig 4.2 Visualization of Cloud

CONCLUSION

The primary objective of this project was to develop an efficient and automated system for managing vehicle entry using RFID technology and IoT integration. The system successfully achieves this objective by combining hardware components such as the ESP32 microcontroller, RFID module, servo motor, LCD display, and buzzer with software tools and cloud-based monitoring. Overall, the project provides a cost-effective, scalable, and user-friendly solution for automated vehicle entry management. It can be deployed in various real-world applications such as parking areas, toll gates, residential complexes, and institutional premises.

FUTURE ENHANCEMENT

Although the system performs effectively, there are several enhancements that can be implemented to improve its functionality and performance in the future.

1. Multi-User Authentication

The system can be upgraded to support multiple RFID tags with a database of authorized users, allowing selective access control.

2. Mobile Application Integration

A mobile application can be developed to monitor vehicle entries, receive notifications, and control the system remotely.

3. Camera Integration

A camera module can be added to capture vehicle images or implement license plate recognition for enhanced security.

4. Biometric Authentication

Additional authentication methods such as fingerprint or facial recognition can be integrated for higher security.

Conflict of interest statement

Authors declare that they do not have any conflict of interest.

REFERENCES

[1] RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification – K Finkenzeller, 3rd Edition, Wiley, 2010.

[2] Internet of Things: Architecture and Design Principles – R. Kamal, McGraw Hill Education, 2017.

[3] Embedded Systems Programming using C – Y. Kanetkar, BPB Publications, 2018.

[4] IEEE Research Papers – "RFID-Based Access Control Systems and IoT Applications."

[5] ResearchGate Publications – "IoT-Based Smart Parking and Vehicle Monitoring Systems."

[6] Espressif Systems – ESP32 Technical Reference Manual (2023) https://www.espressif.com□

[7] NXP Semiconductors – MFRC522 RFID Reader Datasheet (2022) https://www.nxp.com

[8] Arduino – Arduino IDE Documentation (2024) https://www.arduino.cc

[9] MathWorks – Thing Speak IoT Platform Documentation (2024) https://thingspeak.com

[10] GitHub – ESP32 RFID IoT Project Repositories https://github.com