

Design and implementation of a single-gate parking barrier prototype using ESP32-CAM based on the Internet of Things

Devandra Alifka iswahyudi^{1*}, Abdullah Hanif², Karyo Budi Utomo³

^{1,2,3}Department of Information Technology, Politeknik Negeri Samarinda, Indonesia
devandrx25@gmail.com^{1*}, abdullahhanif@polnes.ac.id², karyobudiutomo@polnes.ac.id³

Abstract

The rapid proliferation of motorized vehicles within educational institutions necessitates robust parking management infrastructures to mitigate the unauthorized utilization of reserved spaces. This study proposes the design and implementation of an automated, Internet of Things (IoT)-based parking barrier prototype, employing an ESP32-CAM microcontroller as the core computing unit. The primary objective is to restrict facility access exclusively to authorized personnel by automating license plate identification and cross-referencing this telemetry with a cloud database. Unlike conventional edge-heavy systems, this architecture offloads Optical Character Recognition (OCR) processing to a cloud API to reduce thermal throttling and hardware latency on the ESP32-CAM. The methodology integrates an HY-SRF05 ultrasonic sensor for vehicle proximity detection, the ESP32-CAM for optical image acquisition, and a micro servo motor for barrier actuation. Upon detecting a vehicle within a 50 cm threshold, the system autonomously captures the license plate, processes the image via a cloud-based License Plate Recognition (LPR) API, and verifies the extracted string against a Firebase Realtime Database. Empirical testing demonstrated high system reliability; the ultrasonic sensor exhibited a spatial error margin of less than 2 cm, and the integrated IoT communication architecture achieved an operational latency of merely 2 to 3 seconds.

Keywords: Computer vision; ESP32-CAM; Internet of Things; License plate recognition; Smart campus.

1. Introduction

The proliferation of motorized vehicles in urban centers has significantly increased the demand for efficient parking management, particularly within densely populated educational institutions. In Indonesia, the national motorcycle population exceeded 126 million units by 2022, creating a direct macroscopic impact on regional infrastructures. At the Samarinda State Polytechnic (Polnes), specifically within the Information Technology Department, this phenomenon has manifested as a critical infrastructural bottleneck. Parking lots explicitly reserved for academic faculty and departmental staff are frequently occupied by unauthorized vehicles. Existing parking management paradigms, which heavily rely on manual human surveillance and static visual deterrents, have proven inadequate in mitigating this unauthorized ingress, actively diminishing workplace productivity as personnel expend excessive time securing alternative parking [1], [2], [3].

A review of contemporary literature elucidates various approaches to IoT-based parking management, though several technological limitations persist [4]. Previous research successfully developed an IoT smart parking prototype utilizing NodeMCU ESP8266 architecture [5]. However, the system remained critically dependent on physical Radio Frequency Identification (RFID) tokens for user authentication, presenting scalability and token-loss vulnerabilities. Alternatively, other studies proposed automated tokenless systems leveraging Raspberry Pi microcontrollers and progressive web application technology. While effective, heavy edge-computing approaches utilizing single-board computers often demonstrate significant vulnerability to thermal throttling and require higher power consumption, making them less ideal for scalable, low-cost deployments across multiple institutional gates.

Advancements in computer vision have facilitated the transition toward automated license plate recognition in smart parking applications, enabling robust vehicle entry control at single-gate parking barriers [6], [7]. Integrating machine learning and IoT telemetry enables sophisticated smart parking systems capable of autonomous vehicle identification. Furthermore, decentralized parking logic through fog computing and cloud synchronization, such as the Firebase Realtime Database, has proven highly effective. Building on these foundational insights, this research proposes a novel and practical architectural deployment of the low-cost ESP32-CAM microcontroller specifically tailored for single-gate parking barrier control.

To address the limitations of prior systems that execute heavy computational processes locally, this proposed architecture offloads the Optical Character Recognition (OCR) workload to a cloud-based LPR API [8]. The extracted telemetry is subsequently verified against a Firebase Realtime Database. This study posits that integrating lightweight edge-acquisition hardware with robust cloud processing yields

a highly responsive and cost-effective access control system without the thermal limitations of local edge AI. The objective of this study is to engineer a functional prototype capable of automating authentication based strictly on standard Indonesian license plate topography, providing real-time data synchronization to a web dashboard, and establishing a scalable technological foundation supporting the Smart Campus vision.

2. Methodology

The research methodology adopts an iterative prototyping approach, progressing through distinct developmental phases: requirements analysis, system architecture design, physical implementation, rigorous empirical verification, and evaluation.

2.1. System architecture and hardware topology

The proposed automated parking system operates across three interdependent layers: the sensory input layer, the central control and processing layer, and the mechanical output layer. As illustrated in Figure 1, the hardware topology integrates these components into a unified circuit.

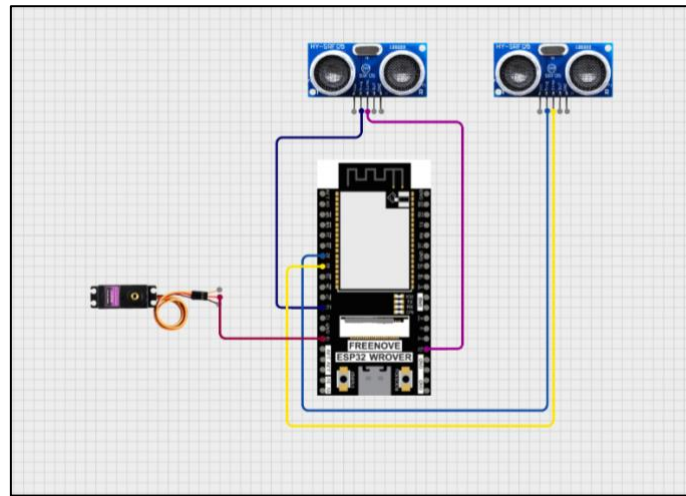


Fig. 1: Hardware topology and circuit design of the parking barrier system.

The sensory input layer is governed by the HY-SRF05 ultrasonic sensor, which operates by emitting 40,000 Hz high-frequency sound waves. Ultrasonic sensors are often used for vehicle detection due to their cost-effectiveness, reliability, and suitability for outdoor conditions [9]. The sensor calculates the time-of-flight of the acoustic echo reflecting off an approaching vehicle to determine precise spatial proximity. This component acts as the primary hardware trigger, ensuring the system remains in a low-power standby state until a vehicle breaches a predefined spatial threshold.

The central control layer is anchored by the ESP32-CAM (WROVER) microcontroller. Selected for its integrated Wi-Fi connectivity and cost-efficiency, it functions as the primary logic controller and the optical acquisition module via its onboard OV2640 camera. Power stability is critical. Sudden current draws during mechanical actuation frequently cause edge devices to reset. To counteract this, the hardware topology incorporates an LM2596 step-down DC-DC buck converter. Operating at a switching frequency of 150 kHz, the LM2596 efficiently regulates a primary 12V 10A power supply down to a stable 5V output, preventing voltage sags. The mechanical output layer utilizes an MG90S micro servo motor as the primary rotary actuator to manipulate the physical barrier mechanism.

2.2. Software logic and cloud integration

The microcontroller's operational logic was programmed utilizing the Arduino IDE ecosystem, leveraging specialized libraries including WiFi.h for network protocols, esp_camera.h for optical buffer management, and FirebaseESP32.h for database interactions. The algorithmic workflow is depicted in Figure 2.

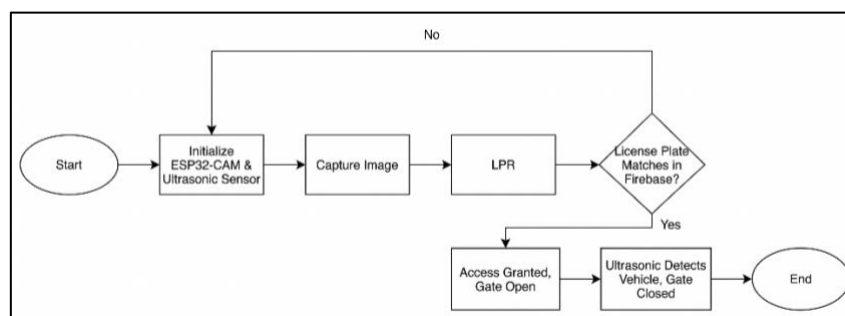


Fig. 2: System algorithmic flowchart and IoT data integration.

The algorithmic flow initiates upon the ultrasonic sensor detecting an object within a 50 cm perimeter. The ESP32-CAM captures a digital image of the vehicle's frontal topography and transmits this visual buffer to a third-party LPR API via an HTTP POST request formatted

as multipart/form-data. The external API executes LPR algorithms to extract alphanumeric characters and returns the parsed string to the microcontroller. By leveraging computer vision and OCR libraries, LPR-based parking systems can automate vehicle identification and entry control, removing the need for manual checks [6].

Upon receiving the string, the ESP32-CAM initiates a synchronous query to the Firebase Realtime Database, scanning the */PlatTerdaftar* (Registered Plates) node. If the parsed string matches an authorized entry, the microcontroller transmits a Pulse Width Modulation (PWM) signal to the servo motor, actuating a 90-degree rotation to open the barrier. Concurrently, the system records the access event, exact timestamp, and verification status into the */Riwayat* (History) node. Recent IoT parking platforms integrate edge devices with Firebase services to log vehicle entry and maintain real-time access history [10].

2.3. Web dashboard interface

To facilitate remote administrative oversight, a highly responsive and modern web dashboard was engineered utilizing a stack of HTML, CSS, JavaScript, and the Tailwind CSS framework. The HTML provides the semantic skeletal structure of the interface, organizing the administrative controls and the real-time access history tables. JavaScript is utilized to manage dynamic client-side logic, establishing direct WebSocket connections to the Firebase Realtime Database. This specific configuration enables instantaneous, bidirectional telemetry synchronization without requiring manual browser refreshes. Through this dashboard, administrators can dynamically add or revoke license plate credentials and trigger a manual hardware override by writing a boolean flag to the */Kontrol/BukaPaksa* node, which the ESP32-CAM continuously polls.

Furthermore, the Tailwind CSS framework was implemented to handle the interface's aesthetic and responsive design. As a utility-first CSS framework, Tailwind allows for rapid UI prototyping directly within the HTML markup, significantly reducing the CSS bundle size and ensuring optimal loading speeds across devices. To achieve a professional, elegant, and modern visual hierarchy, Tailwind's utility classes were utilized to implement a Glassmorphism aesthetic. By combining transparent background classes (e.g., *bg-white/10*), backdrop blur filters (*backdrop-blur-md*), and subtle border accents, the dashboard mimics a sleek 'Back Glass' effect. This clean aesthetic ensures that critical data streams, such as real-time vehicle entry logs, remain visually prioritized and highly legible for operators, effectively addressing the need for an intuitive administrative overview.

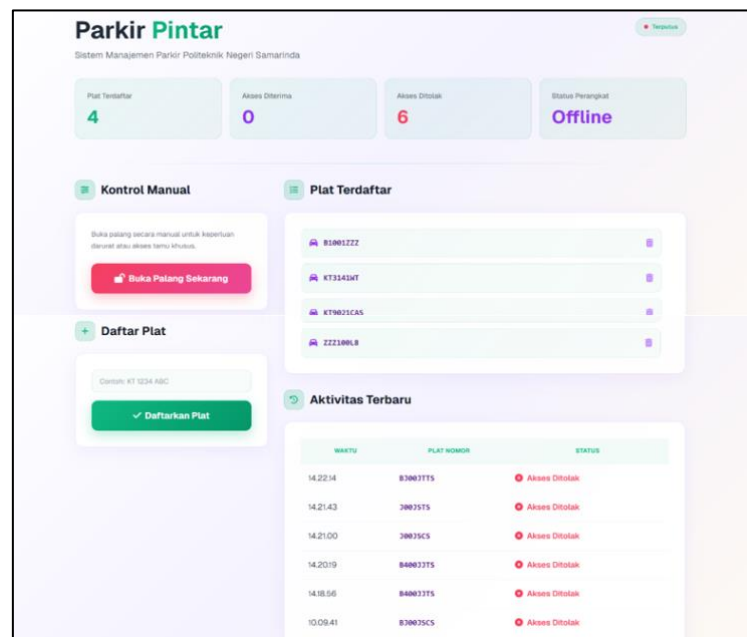


Fig .3: The responsive web dashboard interface for real-time monitoring and manual hardware override.

3. Results and Discussion

To validate the engineering objectives, the assembled functional prototype was subjected to rigorous empirical testing, segmented into hardware subsystem verification, Computer Vision accuracy, and holistic IoT integration analysis.

3.1. Hardware subsystem verification

The spatial accuracy of the HY-SRF05 ultrasonic sensor is paramount, as premature or delayed triggering compromises the camera's ability to capture a sharply focused image. The sensor was evaluated by placing standard physical obstacles at exact measured distances.

Table 1: Empirical evaluation of ultrasonic sensor spatial accuracy

Actual Distance (cm)	Sensor Computation (cm)	Absolute Deviation (cm)	Verification Status
10	11	1	Accurate
30	29	1	Accurate
50	52	2	Accurate
100	99	1	Accurate
150	148	2	Accurate

As detailed in Table 1, the ultrasonic sensor exhibited a maximum absolute deviation of merely 2 cm. This precision guarantees that the ESP32-CAM optical trigger executes precisely when the vehicle occupies the optimal focal plane. The MG90S servo motor demonstrated smooth actuation, confirming the efficacy of the LM2596 buck converter in delivering adequate peak current.

3.2. License Plate Recognition (LPR) Accuracy Evaluation

To rigorously evaluate the reliability of the offloaded Cloud API approach, the system's LPR accuracy was tested under varying environmental lighting conditions (Morning, Afternoon, Night/Low-light).

Table 2: LPR API Performance Metrics across different lighting conditions

Environmental Condition	Total Attempts	Correct Identifications (True Positive)	False Negatives / Misreads	Accuracy (%)	Precision (%)	Recall (%)
Morning (Optimal Light)	18	17	1	94.4	100	94.4
Afternoon (Direct Glare)	23	21	2	91.3	100	91.3
Night (Low-light)	4	3	1	75	100	75

Based on the experimental results presented in Table 2, the proposed cloud-based License Plate Recognition (LPR) system demonstrated reliable recognition performance under varying environmental lighting conditions. Under optimal morning illumination, the system achieved the highest recognition accuracy of 94.4%, successfully identifying 17 out of 18 license plates. During the afternoon, performance decreased slightly to 91.3% due to direct sunlight and glare, which occasionally reduced image clarity and affected character segmentation. The most significant performance degradation was observed under low-light nighttime conditions, where the system achieved 75.0% accuracy, correctly recognizing only three of four license plates. Despite the reduction in accuracy and recall under challenging lighting conditions, the system consistently maintained 100% precision, indicating that all recognized license plates were correctly classified without producing false positive identifications. These findings demonstrate that ambient illumination primarily influences the detection capability of the cloud-based LPR system, while recognition reliability remains high once a license plate is successfully detected. Therefore, integrating supplementary illumination or dedicated lighting is expected to further improve recognition performance, particularly during nighttime operation.

3.3. IoT integration and latency evaluation

Operational latency defined as the total temporal delay from initial ultrasonic detection to the final mechanical servo actuation is a critical metric for traffic flow efficiency.

Table 3: Assessment of end-to-end IoT system integration and synchronization

Timestamp	Processed License Plate	Firebase Verification Status	Hardware Barrier Response	Dashboard Synchronization
09:21	B1001ZZZ	Access Granted	Actuated (Opened)	Log updated synchronously
09:23	KT5544AA	Access Denied	Static (Closed)	Denial notification displayed
09:30	B1001ZZZ	Access Granted	Actuated (Opened)	Log updated synchronously
09:40	N/A (Simulation)	Offline	Static (Closed)	System marked Offline
09:45	KT8845BG	Access Granted	Actuated (Opened)	Log updated synchronously

Observational analysis indicated that the end-to-end processing latency consistently averaged between 2 and 3 seconds. The web dashboard exhibited near-instantaneous synchronization, reflecting database mutations in under two seconds.

3.4. Discussion and Implications

The architectural decision to offload OCR processing to a dedicated cloud API successfully circumvented the thermal throttling and latency issues typically associated with deploying heavy machine learning models natively on edge devices like the ESP32-CAM. Achieving a 2 to 3-second holistic latency proves that this hybrid approach is highly efficient for real-world academic parking scenarios.

However, the reliance on an external Cloud LPR API introduces distinct scientific and operational limitations. As indicated by the LPR evaluation, the accuracy degrades significantly in low-light environments because the ESP32-CAM relies entirely on ambient illumination. To mitigate this without abandoning the cloud approach, future iterations must integrate synchronized high-intensity LED flashes.

Furthermore, while the Cloud API reduces local computation, it creates a strict dependency on continuous internet connectivity. A comparative analysis suggests that transitioning from a Cloud API to a decentralized edge-based Local Area Network (LAN) deployment utilizing models like YOLOv5 deployed on a dedicated campus local server could provide higher data privacy, zero internet dependency, and potentially faster inference times.

From a software scalability perspective, the current integration of a NoSQL cloud infrastructure (Firebase) provided an exceptionally stable telemetry backbone. Nevertheless, for institution-wide deployment involving multiple gates and thousands of users, the current vanilla HTML/JS dashboard should be refactored utilizing modern full-stack frameworks (e.g., Next.js or Laravel) to support complex relational queries, user roles, and advanced data analytics.

4. Conclusion

This research successfully engineered and validated an automated parking barrier prototype utilizing an ESP32-CAM microcontroller integrated within a cloud-offloaded IoT architecture. By transitioning from conventional physical authentication to automated license plate recognition, the system effectively mitigates unauthorized vehicular ingress. Empirical testing confirmed the high precision of the ultrasonic detection mechanism (spatial deviation < 2 cm) and demonstrated an exceptionally efficient end-to-end processing latency of 2 to 3 seconds. While the offloaded Cloud API approach successfully prevented edge-device thermal throttling, it highlighted vulnerabilities in low-light optical acquisition. Future research should prioritize integrating supplementary automated illumination and exploring the deployment of lightweight deep learning models (such as YOLO) on localized campus edge servers to eliminate external internet dependencies and maximize systemic resilience.

References

- [1] M. A. Ala'anzy, A. Abilakim, R. Zhanuzak, and L. Li, 'Real time smart parking system based on IoT and fog computing evaluated through a practical case study', *Sci. Rep.*, vol. 15, no. 1, p. 33483, Sep. 2025, doi: 10.1038/s41598-025-15507-6.
- [2] J. Yang, J. He, and X. Wang, 'Design of Intelligent Parking System Based on Internet of Things and Cloud Platform', *Int. J. Grid High Perform. Comput.*, vol. 15, no. 2, pp. 1–18, Jan. 2023, doi: 10.4018/IJGHPC.316836.
- [3] V. Knights, O. Petrovska, and M. Prchkovska, 'Enhancing Smart Parking Management through Machine Learning and AI Integration in IoT Environments', in *Navigating the Internet of Things in the 22nd Century - Concepts, Applications, and Innovations*, M. Usman Tariq, Ed., IntechOpen, 2024. doi: 10.5772/intechopen.1006490.
- [4] I. Yasmin, R. Rubab, M. Afzal, and J. Rusool, 'AI-Driven Parking Management: ANPR-Based Entry & Biometric Gate Control', *Int. J. Innov. Sci. Technol.*, pp. 1437–1452, Jul. 2025, doi: 10.33411/ijist/20257314371452.
- [5] G. Pradhan, M. R. Prusty, V. S. Negi, and S. Chinara, 'Advanced IoT-integrated parking systems with automated license plate recognition and payment management', *Sci. Rep.*, vol. 15, no. 1, p. 2388, Jan. 2025, doi: 10.1038/s41598-025-86441-w.
- [6] M. Li and L. Zhang, 'Deep Learning-based License Plate Recognition in IoT Smart Parking Systems using YOLOv6 Algorithm', *Int. J. Adv. Comput. Sci. Appl.*, vol. 14, no. 12, 2023, doi: 10.14569/IJACSA.2023.0141223.
- [7] M. M. Abdellatif, N. H. Elshabasy, A. E. Elashmawy, and M. AbdelRaheem, 'A low cost IoT-based Arabic license plate recognition model for smart parking systems', *Ain Shams Eng. J.*, vol. 14, no. 6, p. 102178, Jun. 2023, doi: 10.1016/j.asej.2023.102178.
- [8] Y.-C. Yu, 'Smart Parking System Based on Edge-Cloud-Dew Computing Architecture', *Electronics*, vol. 12, no. 13, p. 2801, Jun. 2023, doi: 10.3390/electronics12132801.
- [9] C. N S and V. N D, 'IoT-Enabled Parking Space Detection Using Ultrasonic Sensors', *World J. Adv. Res. Rev.*, vol. 7, no. 3, pp. 390–399, Sep. 2020, doi: 10.30574/wjarr.2020.7.3.0301.
- [10] M. Ebrahimishadman and A. Souri, 'Edge-Optimized Automatic Number Plate Recognition for IoT-Based Smart Parking Using YOLOv8', *Comput. Mater. Contin.*, vol. 0, no. 0, pp. 1–10, 2026, doi: 10.32604/cmc.2026.084246.