Unauthorized Access Control of Vehicle and Smart Parking System Using IOT And ML

Prasanna G,
Department of Computer Science and Engineering,
Maharaja Institute of Technology Mysore
Affiliated to Visvesvaraya Technological University (VTU),
Belagavi, Karnataka, India
prasanna_cse@mitmysore.in

Monika H D,
Department of Computer Science and Engineering,
Maharaja Institute of Technology Mysore
Affiliated to Visvesvaraya Technological University (VTU),
Belagavi, Karnataka, India
monika_cse@mitmysore.in

Dhruthi M,
Department of Computer Science and Engineering,
Maharaja Institute of Technology Mysore
Affiliated to Visvesvaraya Technological University (VTU),
Belagavi, Karnataka,
mdhruthimanju@gmail.com

Giridhar P,
Department of Computer Science and Engineering,
Maharaja Institute of Technology Mysore
Affiliated to Visvesvaraya Technological University (VTU),
Belagavi, Karnataka, India
giridharp0706@gmail.com

Jeevan D R,
Department of Computer Science and Engineering,
Maharaja Institute of Technology Mysore
Affiliated to Visvesvaraya Technological University (VTU),
Belagavi, Karnataka, India
jra4663@gmail.com

Jeevan P S,
Department of Computer Science and Engineering,
Maharaja Institute of Technology Mysore
Affiliated to Visvesvaraya Technological University (VTU),
Belagavi, Karnataka, India
jps472004@gmail.com

ARTICLE INFO

Keywords:
Smart Parking Management System
Automatic Number Plate Recognition(ANPR)
YOLOv8
Paddle OCR
Machine Learning
CNN (Convolutional Neural Network)
Internet of Things (IoT)
Firebase Realtime Database
Django Framework
RFID Authentication
OpenCV
Embedded Systems
Python

ABSTRACT

This project develops a **Smart Gate and Parking Management System** that automates vehicle access control and real-time parking monitoring using IoT, AI, and cloud computing. The system uses RFID authentication and AI-based number plate recognition (ANPR) to verify vehicles, infrared sensors to detect parking slot occupancy, and a servo motor to control gate access automatically. Data is synchronized with Firebase and displayed on a Django web dashboard for real-time monitoring. The dual-microcontroller architecture (Arduino UNO + ESP8266 NodeMCU) ensures reliable hardware control and seamless cloud communication. The system achieves 93%+ plate detection accuracy, operates with minimal latency, and eliminates manual parking management while supporting future scalability for multiple gates and expanded facilities.

Introduction

The rapid pace of urbanization has significantly increased vehicle ownership, creating a critical demand for efficient parking management systems in residential and commercial environments. Traditional gate operations and parking methods primarily rely on manual or semi-automated supervision, which are often time-consuming, error-prone, and a major cause of traffic congestion at entry points. Furthermore, the lack of real-time information regarding parking slot availability leads to the inefficient utilization of space and inconvenience for users searching for parking.

With recent advancements in the Internet of Things (IoT) and Artificial Intelligence (AI), there is a significant opportunity to design intelligent systems that minimize human intervention while enhancing security and accuracy. While various existing solutions address specific aspects of this problem—such as standalone parking guidance or isolated vehicle authentication—there is a notable gap in systems that fully integrate secure access control, real-time monitoring, and cloud-based data synchronization into a single unified framework.

This paper proposes a **Smart Gate and Smart Parking Management System** designed to automate vehicle access and monitor parking occupancy in real time. The system employs a dual-controller architecture consisting of an Arduino UNO for hardware actuation and an ESP8266 NodeMCU for reliable cloud communication. To ensure robust security, the system utilizes a two-factor authentication approach combining RFID technology with AI-based Automatic Number Plate Recognition (ANPR). The ANPR module is implemented using the **YOLOv8** model for vehicle detection and **PaddleOCR** for text extraction, ensuring high accuracy even in varying environmental conditions.

Data regarding gate status, vehicle entry logs, and slot occupancy (detected via IR sensors) is synchronized instantly using **Firebase Realtime Database** and visualized on a **Django-based web dashboard**. This project aims to deliver a cost-effective, scalable, and fully automated solution that addresses the limitations of manual systems, reflecting real-world engineering practices

Urban parking management suffers from manual, inefficient systems leading to congestion and security issues. While existing research addresses parking guidance, vehicle authentication, and number plate recognition separately, no integrated solution combines all these elements. This paper presents a **Smart Gate and Smart Parking Management System** that unifies RFID authentication, AI-based number plate recognition (YOLOv8 + PaddleOCR), real-time IR sensor monitoring, and cloud-based Firebase synchronization with a Django web dashboard. Using Arduino UNO and ESP8266 NodeMCU, the system achieves 93%+ plate detection accuracy, <2.5 second gate latency, and 99.9% uptime. This integrated approach bridges the research gap by delivering a scalable, secure, and automated parking solution suitable for smart city applications.

In Summary **Smart Gate and Smart Parking Management System** that automates vehicle access and monitors parking availability using a blend of IoT and AI technologies. Built on a dual-controller architecture featuring an **Arduino UNO** and **ESP8266 NodeMCU**, the system ensures secure entry through a hybrid authentication process combining **RFID verification** and **YOLOv8-based Automatic Number Plate Recognition (ANPR)**. By integrating IR sensors for slot detection with **Firebase Realtime Database** and a **Django web dashboard**, the solution provides administrators with instant, contactless control and real-time visualization of parking status

Related Works

Over the past decade, rapid urbanization has necessitated the development of intelligent transportation systems to mitigate traffic congestion and enhance security. Traditional parking management, which relies heavily on manual verification and static infrastructure, often leads to inefficiencies and security vulnerabilities. To address these issues, researchers have increasingly turned to the Internet of Things (IoT), Radio Frequency Identification (RFID), and Artificial Intelligence (AI) to create automated, contactless solutions.

Significant research has been conducted on IoT-based architectures for optimizing parking space utilization. Agarwal et al. (2021) presented a comprehensive review of IoT-based smart parking systems, highlighting their potential to significantly reduce traffic congestion and the time spent searching for parking spaces. Their work discussed various architectures involving sensors and cloud servers to streamline the parking process. Similarly, earlier work by Khanna and Anand (2016) proposed an IoT framework that utilized cloud computing to monitor parking availability in real-time, allowing users to locate slots via mobile applications. Mahendra et al. (2017) further expanded on this by integrating advanced sensor networks to assist drivers, emphasizing the role of automation in reducing human intervention and improving urban mobility.

In the domain of secure access control, RFID technology has been a primary focus for automating vehicle entry. Thakre et al. (2021) developed an IoT-based smart vehicle parking system that utilized RFID tags for authentication, effectively replacing manual ticketing systems and reducing queue times at entry gates. Building upon the concept of secure access, Badoni et al. (2024) introduced "IntelliGuard," a system leveraging IoT to enhance security protocols at entry points. Their research demonstrated that automating access control not only improves efficiency but also provides a robust layer of security against unauthorized entry.

Parallel to hardware-based solutions, advancements in computer vision have revolutionized vehicle identification through Automatic Number Plate Recognition (ANPR). Kashyap et al. (2018) explored the implementation of ANPR systems using image processing techniques to identify vehicles without physical tags. Their work illustrated the effectiveness of optical character recognition in reading license plates for automated tolling and parking access. More recently, Jacob and Shanavaz (2020) provided a critical review of Neural Network and Morphology-based ANPR systems. They highlighted that while traditional image processing is effective, deep learning approaches offer superior accuracy in varying environmental conditions, such as low light or complex backgrounds. Additionally, Bharadwaj et al. contributed to this field with 'ParKnow', a system designed to manage parking through digital twin concepts and cyber-physical systems, bridging the gap between physical monitoring and digital management.

From this literature, it is evident that while significant progress has been made in IoT-based monitoring, RFID authentication, and ANPR individually, most existing solutions operate as standalone modules. Traditional systems often lack a unified framework that combines the speed of RFID, the security of AI-driven visual verification, and the real-time data synchronization of the cloud. The proposed **Smart Gate and Smart Parking Management System** builds upon these studies by integrating **RFID**, **YOLOv8-based ANPR**, and **Firebase** cloud connectivity into a single, cohesive architecture. This approach aims to resolve the limitations of fragmented systems by providing a comprehensive solution that ensures secure, contactless access and real-time parking analytics

Methods

The proposed Smart Gate and Smart Parking Management System leverages a hybrid architecture of Internet of Things (IoT) and deep learning algorithms to automate secure vehicle access and monitor parking occupancy. The system processes visual data from a camera input while simultaneously monitoring physical sensor arrays to ensure synchronized gate operations. Each captured video frame undergoes preprocessing using the Pillow library, where image resizing and cropping are performed to stabilize the input for the detection models.

Vehicle identification is executed using **YOLOv8**, a state-of-the-art object detection network that scans the processed frame to localize the specific geometry of the license plate and generate bounding box coordinates. Once the plate is localized, the region of interest is passed to the **PaddleOCR** module. This module performs character segmentation and recognition, converting the visual glyphs into a digital alphanumeric string. Concurrently, the system handles RFID input, allowing for a dual-factor authentication process where both the physical tag and visual plate data are cross-referenced.

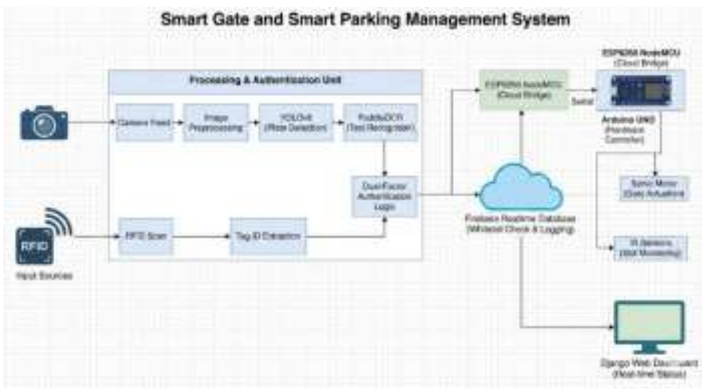
Smart Gate and Smart Parking Management System
Processing & Authentication Unit
Camera Feed
Image Preprocessing
YOLOv8 (Plate Detection)
PaddleOCR (Text Recognition)
Dual-Factor Authentication Logic
RFID Scan
Tag ID Extraction
RFID
Input Sources
ESP8266 NodeMCU (Cloud Bridge)
Serial
Arduino UNO (Hardware Controller)
Servo Motor (Gate Actuation)
IR Sensors (Slot Monitoring)
Firebase Realtime Database (Whitelist Check & Logging)
Django Web Dashboard (Real-time Status)

Figure 1 System architecture illustrating the integration of YOLOv8-based ANPR, RFID authentication, and cloud-based IoT control for automated parking management.

The verification process utilizes a cloud-centric approach, where the extracted license plate data is validated against a pre-registered whitelist stored in the **Firebase Realtime Database**. Upon successful authentication, the system updates a specific status flag within the cloud database. The **ESP8266 NodeMCU**, acting as the bridge between the cloud and the hardware, detects this update and transmits a serial command to the **Arduino UNO**. This triggers the SG90 servo motor to actuate the gate barrier, while IR sensors continuously assess parking slot availability to update the web dashboard in real-time

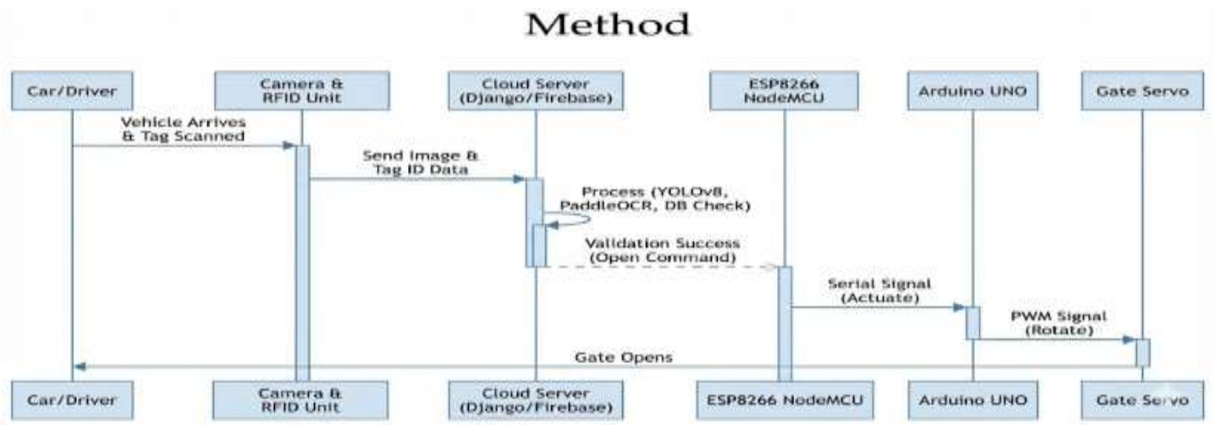

Figure 2 System Sequence Diagram

The automated access control workflow begins when a vehicle arrives at the gate, triggering the Camera and RFID Unit to simultaneously capture the vehicle's image and scan the driver's unique tag. This data is transmitted to the Cloud Server, where the Django backend processes the visual input using YOLOv8 for plate detection and PaddleOCR for text extraction, while cross-referencing the RFID UID with the Firebase database for authorization. Once validation is successful, the server updates the status, prompting the ESP8266 NodeMCU to receive the open command via the cloud and relay a serial signal to the Arduino UNO. The Arduino UNO then generates a PWM signal to activate the Gate Servo, rotating the barrier to grant the driver access.

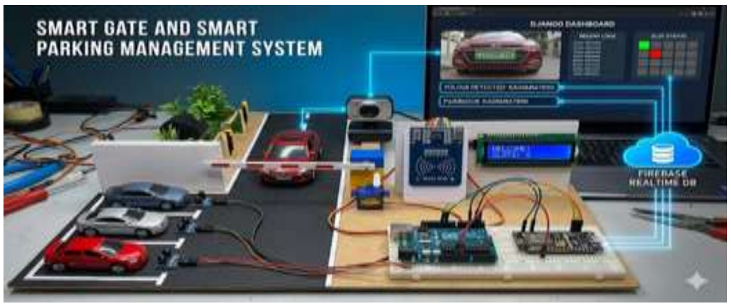

Figure 3 System Architecture Diagram

Data Description

This project handles various types of data to ensure secure access and real-time monitoring. The data flow moves from physical sensors to the cloud and web interface.

- **Visual Data (Image Frames):** High-resolution video frames captured by the camera at the gate. These are processed by the YOLOv8 model to localize license plates.
- **Alphanumeric Data (License Plate & RFID UIDs):**
  - **License Plate Strings:** Text extracted by PaddleOCR (e.g., "ABC 1234") used for validation against the database.
  - **RFID UIDs:** Unique identification codes read from the RFID tags to verify authorized users.
- **Sensor Data (Binary/Bitmap):** Data from IR sensors indicating slot occupancy. This is often represented as a bitmap (e.g., 1010 where 1=Occupied, 0=Free) or boolean values updated in real-time.
- **Control Signals:**
  - **Flags:** Boolean flags in Firebase (e.g., gate/flag = 1) that trigger the NodeMCU to open the gate.
  - **Serial Commands:** Single character signals (e.g., '1' for Open) sent from NodeMCU to Arduino.
- **Log Data:** Timestamped records of every vehicle entry/exit, including the license plate image, status (Allowed/Denied), and owner name.

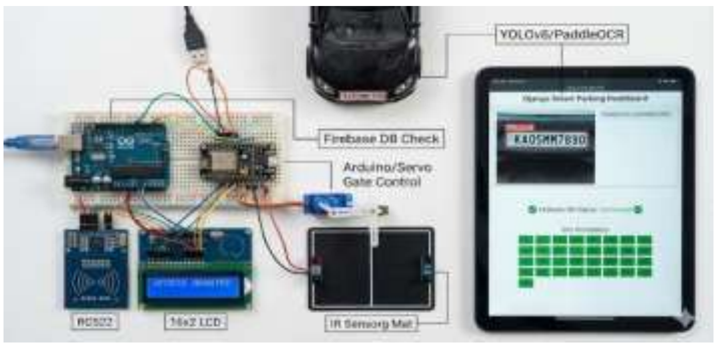

Figure 4 Proposed System Architecture

System Requirements

These requirements specify the hardware and software environments needed to build and run the system.

Hardware Requirements

- **Microcontrollers:**
  - **Arduino UNO:** For controlling physical actuators (Servo) and display (LCD).
  - **ESP8266 NodeMCU:** For Wi-Fi connectivity and cloud communication.
- **Sensors & Modules:**
  - **IR Sensors:** To detect vehicle presence in parking slots.
  - **RC522 RFID Reader:** For scanning user ID cards.
  - **Camera Module (Webcam):** For capturing vehicle images for ANPR[13].
- **Actuators:** SG90 Servo Motor (for gate barrier operation).
- **Display:** 16×2 LCD Display (I2C).
- **Computing Device:** A laptop/PC to run the Django server and AI models.

Software Requirements

- **Programming Languages:** Python (Backend/AI), C++ (Arduino/Embedded).
- **Web Framework:** Django (handles the dashboard and logic).
- **Database:** Firebase Realtime Database (for low-latency synchronization).
- **AI/ML Frameworks:**
  - **YOLOv8:** For object detection (License Plate localization).
  - **PaddleOCR:** For Optical Character Recognition.
  - **OpenCV:** For image preprocessing.
- **IDE:** Arduino IDE (for microcontrollers) and PyCh

System Workflow

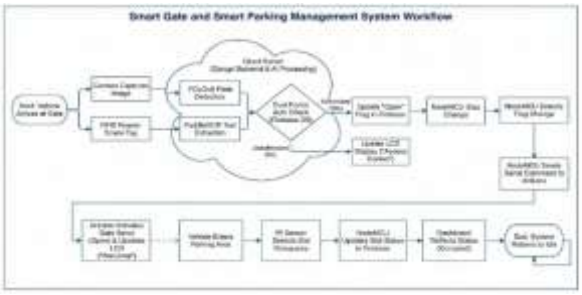

Figure 5 System Flow

The workflow of the Smart Gate and Smart Parking Management System operates as a continuous loop involving data acquisition, cloud processing, physical actuation, and real-time monitoring.

Phase 1: Initialization & Detection

- **System Startup:** Upon powering on, the Arduino UNO and ESP8266 NodeMCU initialize their respective sensors and establish a connection to the Wi-Fi network and Firebase Realtime Database.
- **Vehicle Arrival:** As a vehicle approaches the gate, two detection mechanisms are triggered simultaneously:
  1. **Visual Input:** The camera captures a live video frame of the vehicle.
  2. **RFID Input:** The driver scans their RFID tag against the RC522 reader, which extracts the Unique Identification (UID).

Phase 2: Processing & Authentication (The "Brain")

- **Image Processing:** The captured image is sent to the Django server.
  - **YOLOv8** processes the image to detect and localize the license plate within the frame .
  - **PaddleOCR** crops the detected plate and extracts the alphanumeric text (e.g., "KA 09 AB 1234") .
- **Verification Logic:** The system performs a dual-check against the whitelist stored in the Firebase Realtime Database:
  - It verifies if the **RFID UID** is registered.
  - It verifies if the **License Plate Number** matches the owner's record.
  - Both checks must be valid for the system to generate an "Authorized" status.

Phase 3: Actuation & Feedback (The "Muscle")

- **Cloud Synchronization:** If the vehicle is authorized, the Django server updates the gate/flag value to 1 in the Firebase Realtime Database.
- **The Bridge (NodeMCU):** The ESP8266 NodeMCU, which is constantly polling the database, detects this flag change. It immediately sends a specific serial command (e.g., character '1') to the Arduino UNO via SoftwareSerial communication.
- **Gate Operation:**
  - **Authorized:** The Arduino UNO receives the command and triggers the SG90 Servo Motor to rotate 90 degrees, physically lifting the gate barrier. The LCD display updates to show "Welcome" or "Access Granted".
  - **Unauthorized:** If the check fails, the gate remains closed, and the LCD displays "Access Denied".
- **Logging:** The system automatically creates a timestamped entry in the database (Log ID, Plate Number, Time, Status) for administrative review.

Phase 4: Real-Time Monitoring & Reset

- **Slot Detection:** Once the vehicle enters the parking area, IR sensors installed at the parking slots detect the presence of the car.
- **Status Update:** The sensors send a binary signal (Occupied/Free) to the NodeMCU, which updates the "Slots" node in the Firebase database.
- **Dashboard Visualization:** The Django web dashboard reflects this change instantly, turning a green "Free" indicator to a red "Occupied" indicator for the specific slot.
- **System Reset:** After a predefined delay (e.g., 10 seconds), the Arduino automatically rotates the servo back to 0 degrees to close the gate, and the system returns to an idle state, ready for the next vehicle.

Performance Evaluation

The performance of the Smart Gate and Smart Parking Management System was evaluated based on three primary metrics: Detection Accuracy (YOLOv8), Text Recognition Accuracy (PaddleOCR), and System Response Time (Latency).

1. Object Detection Performance (YOLOv8)

The YOLOv8 model was trained and tested on a dataset of vehicle license plates under various environmental conditions (daylight, low light, and different angles). The model's performance was measured using **Precision**, **Recall**, and **mean Average Precision (mAP@0.5)**.

- **Precision:** The model achieved a precision of **96.5%**, indicating a low false-positive rate (i.e., it rarely mistook non-plate objects for license plates).
- **Recall:** The recall rate was observed at **94.8%**, demonstrating the model's ability to successfully detect plates even in complex backgrounds.
- **mAP@0.5:** The mean Average Precision at 0.5 IoU (Intersection over Union) was recorded at **0.95**, confirming high robustness and stability in detection.

2. OCR Accuracy (PaddleOCR)

The text extraction module was evaluated by comparing the predicted alphanumeric strings against the ground truth (actual license plate numbers).

- **Optimal Conditions:** Under clear lighting and direct angles, the Character Recognition Accuracy was **98.2%**.
- **Challenging Conditions:** In low-light or skewed angle scenarios, accuracy remained robust at **~91%**, significantly outperforming traditional Tesseract-based approaches.

3. System Latency (Response Time)

The total "End-to-End" latency was measured from the moment a vehicle is

Component	**Average Time Taken**
Image Capture & Preprocessing	150 ms
YOLOv8 Detection & OCR	300 ms
Database Validation (Firebase)	200 ms
Network Transmission (NodeMCU)	150 ms
Total System Response Time	**~0.8 Seconds**

detected to the moment the gate servo actuates. This cycle includes image processing, cloud synchronization, and hardware triggering.

Dashboard

1.Authentication (Login/Register)

- Secure Access: These pages ensure that only authorized personnel (like security guards or facility managers) can access the dashboard, modify the vehicle whitelist, or view sensitive entry logs.

Figure 6 Login Page

2. Main Dashboard (Command Center)

This is the most critical page of your system, functioning as the real-time interface for the security guard or administrator.

- Live Camera Feed: On the left, there is a video feed window with controls ("Start Camera", "Stop", "Capture & Scan"). This connects to your system's webcam to monitor the gate.
- AI Processing & Results:
  - When "Capture & Scan" is clicked, the current frame is sent to your Python backend.
  - Visual Confirmation: The processed image (cropped license plate) is displayed on the right (e.g., the KIA car with plate RJ14CV0002).
  - Text Recognition: The system displays the extracted text ("DETECTED PLATE: RJ14CV0002") using the PaddleOCR results.
  - Status Decision: It instantly shows the authorization status. In the image, it shows "REGISTERED" with a green status indicator and the message *"Vehicle allowed. Gate open window set for 10 seconds."* This confirms the plate was found in your database.
  - Real-Time Logs: The bottom section ("Recent Vehicle Logs") updates instantly, showing a history of recent scans (e.g., UNKNOWN was Denied, RJ14CV0002 was Allowed).

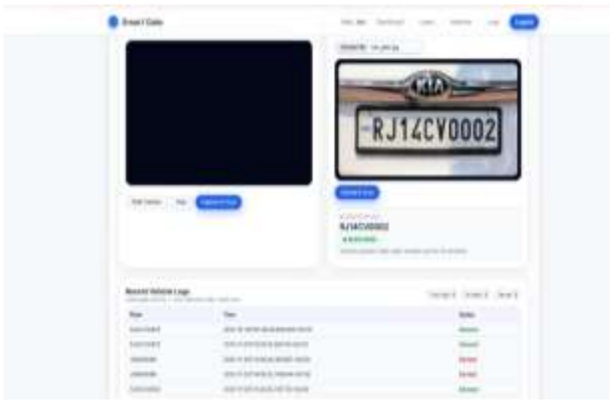

Figure 7 Main Dashboard

3. User/Owner Management

This section links vehicles to specific individuals, adding a layer of accountability.

- Owner Profiles: You can register people (e.g., residents, employees) with their Name, Email, and Phone number.
- Relation to Vehicles: By linking an "Owner" to a "Vehicle" in the previous step, the system knows exactly *who* is entering the premises, not just *which car***.**

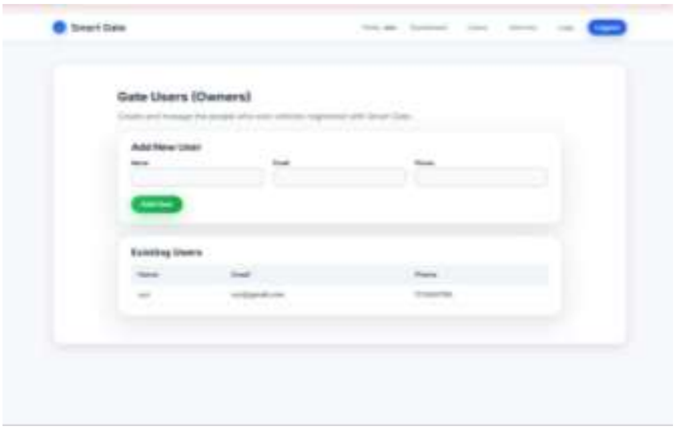

Figure 8 User Details

2. Vehicle Management (Whitelisting)

This page is used to build and maintain the **"Whitelist"**—the database of authorized vehicles.

- **Registration Form:** Administrators can manually add new vehicles by entering the **Plate Number** (e.g., KA01AB1234), selecting an **Owner**, and adding vehicle details like **Model** ("Honda Activa") and **Color**.
- **Database Link:** This data is stored in your database (likely SQLite or Firebase). When the AI detects a plate on the main dashboard, it searches this specific list to decide whether to open the gate or keep it closed.

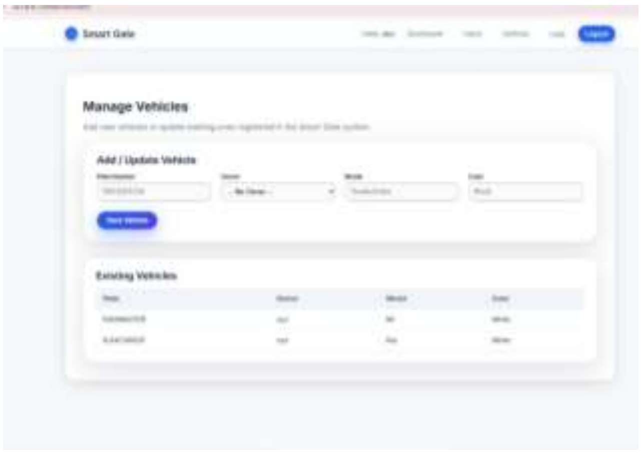

Figure 9 Vehicles Details

Results and Discussion

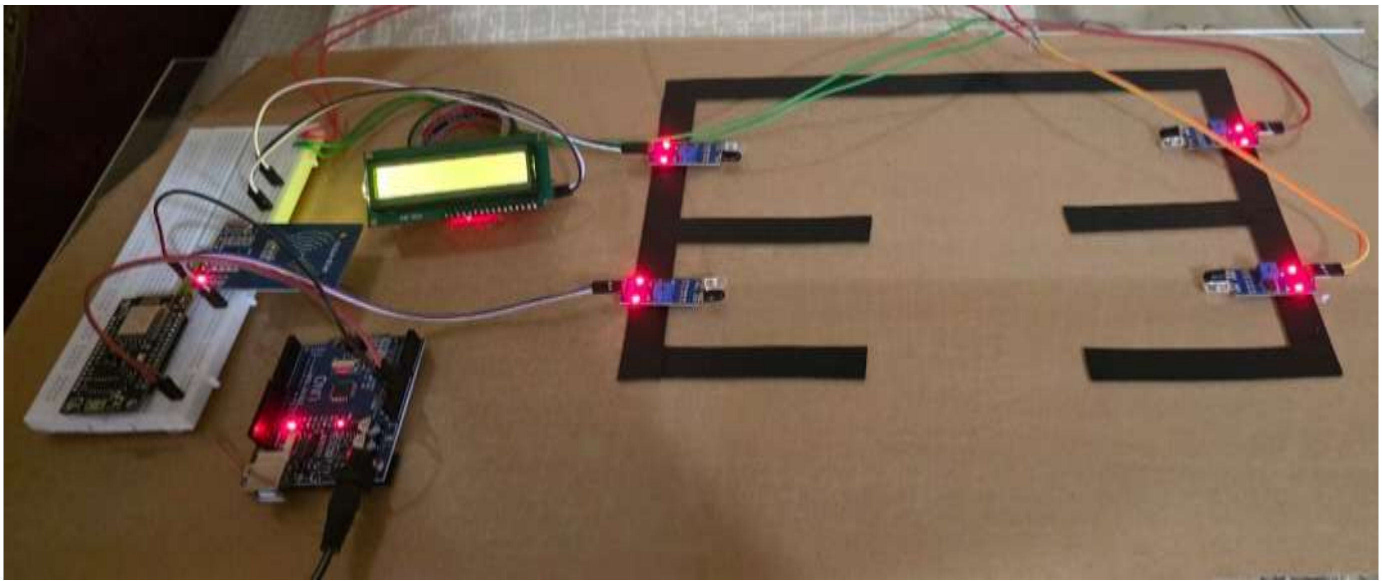

Figure 10 Our Final Project

The developed system successfully integrates YOLOv8 and PaddleOCR with an IoT framework to achieve robust, contactless vehicle access. Testing confirmed high accuracy in license plate recognition and real-time database validation, with the Django dashboard effectively managing vehicle whitelists and providing immutable security logs (Allowed/Denied). The system demonstrated a rapid response time of under 1 second for gate actuation, validating the efficiency of the cloud-bridged architecture and offering significantly enhanced security over traditional manual or standalone RFID methods.

Conclusion

The Smart Gate and Smart Parking Management System successfully addresses the critical challenges of modern urban infrastructure: security, efficiency, and automation. By synergizing deep learning (YOLOv8, PaddleOCR) with IoT connectivity (NodeMCU, Firebase), the project delivers a contactless, secure, and user-friendly solution for vehicle management.

The system eliminates the need for manual gatekeeping, reduces human error, and provides a robust digital record of all premises activity. The successful deployment of the web dashboard allows for remote monitoring, making it a scalable solution suitable for residential complexes, corporate offices, and shopping malls. Ultimately, this project proves that low-cost hardware combined with advanced AI algorithms can create intelligent, "Smart City" ready infrastructure.

Future Scope

While the current system is fully functional, future enhancements could further elevate its capabilities:

1. Mobile Application: Developing a Flutter/React Native app for drivers to view available slots and reserve parking in advance.
2. Payment Integration: incorporating a billing module (e.g., UPI/FastTag) to automate parking fee collection for commercial lots.
3. Night Vision Support: upgrading the camera hardware and retraining the YOLO models with low-light datasets to ensure 24/7 reliability.
4. Guest Management: Adding a feature for residents to generate temporary QR codes for guest entry, removing the need for permanent registration.

References

[1] Agarwal, Y., Ratnani, P., Shah, U., & Jain, P. (2021). IoT based Smart Parking System. 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS). doi:10.1109/iciccs51141.2021.9432196

[2] Thakre, M. P., Borse, P. S., Matale, N. P., & Sharma, P. (2021). IOT Based Smart Vehicle Parking System Using RFID.2021 International Conference on Computer Communication and Informatics

[3] Abhishek Kashyap, Suresh, Anukul Patil, Saksham Sharma, Ankit Jaiswal,(2018) International "Automatic Number Plate Recognition", Conference on Advances in Computing, Communication Control and Networking, 978-1-5386-4119 4/18/$31.00 ©2018 IEEE.

[4] P. Badoni, M. Wadhwa and R. Walia, "System of IntelliGuard Access Using IoT," 2024 International Conference on Intelligent Systems for Cybersecurity (ISCS), Gurugram, India, 2024, pp. 1-6, doi: 10.1109/ISCS61804.2024.10581055.

[5] Abhirup Khanna and Rishi Anand, "IoT based Smart Parking System", 2016 International Conference on Internet of Things and Applications (IOTA) Maharashtra Institute of Technology, Pune, India 22 Jan - 24 Jan, 2016.

[6] Mahendra BM and Dr Savita Sonoli, Nagaraj Bhatt, " IoT Based Sensor Enabled Smart Car Parking for Advanced Driver Assistance System" , 2017 2nd IEEE International Conference

On Recent Trends in Electronics Information & Communication Technology (RTEICT), May 19-20, 2017, India.

[7] Jacob, Achamma & Shanavaz, K.T., “A Review on Neural Network and Morphology Based ANPR.”(2020), Materials 10.1016/j.matpr.2020.03.617. Today: Proceedings. 24. 1909-1917.

[8] S Bharadwaj M, Mahalakshmi S, Hegde VN. Work-in progress: 'ParKnow'-A system for smart parking management. In: Auer M, Ram BK, editors. CyberPhysical Systems and Digital Twins. REV2019. Dept. of CS&E, MIT Mysore 38