AUTOPARK

A NEXT-GEN PARKING SOLUTION

IoT Based Smart Parking System

Abstract

Urban parking congestion, largely driven by rising vehicle density, causes over 30% of city traffic and results in significant economic losses due to fuel waste and delays. Existing systems relying on manual management or RFID face issues like inefficiency, human error, and security risks. To tackle this, we developed **AUTOPARK**—an IoT-powered smart parking system designed for seamless vehicle authentication, real-time slot tracking, and contactless payments. **AUTOPARK** enhances security, automates operations, and improves space utilization, making it ideal for scalable deployment in smart cities. This project not only delivered technical improvements but also strengthened our skills in teamwork, problem-solving, and project execution.

Contents

Problem statement

Urban areas face severe parking inefficiencies, causing traffic congestion and revenue losses.

Manual systems are prone to human errors, unauthorized parking, and fraud.

RFID-based systems, while more secure, suffer from high costs, cloning risks, and maintenance issues.

Introduction

To address the mentioned problems, we present AUTOPARK, an IoT-driven smart parking solution designed for automation, affordability, and scalability. It uses ESP8266 microcontrollers, IR sensors, servo motors, and QR code-based authentication to fully automate entry, exit, slot monitoring, and billing.

Users scan a unique QR code for access, with real-time slot availability shown on OLED displays and a cloud dashboard. Exit is similarly automated, with fees calculated based on parking duration. This reduces manual intervention, unauthorized access, and operational costs.

Comparing with existing systems

Cost Analysis: RFID vs. IoT

- RFID: $50–$70 per vehicle; high maintenance (tag replacement, recalibration)
- IoT (ESP8266 + IR sensors): ~30% lower cost; minimal maintenance

Efficiency Metrics

- Response Time:
  - IoT: 45% faster entry/exit
- Automation: Higher in IoT-based systems
- Security: RFID more secure unless IoT uses enhanced encryption

Scalability

- RFID: Costly, infrastructure-heavy for dense areas
- IoT: Easily scalable; modular expansion
- Real-world examples: Madrid, Tokyo — 60% better space utilization with AI

Hardware Layer (1/2)

IR Sensor Module

- Detects vehicle presence (range: 2–30 cm)
- Operates at 3.3V–5V with <1 ms response time
- Real-time parking slot occupancy detection

Servo Motor Module

- Controls entry/exit gates
- Torque: 2.5–10 kg/cm | Voltage: 4.8V–6V
- Rotational range: 0–180° for barrier movement
- Triggered after QR code verification

Parking Slot Detection System

- 1 ESP8266 + 4 IR sensors + OLED display
- Detects and shows real-time slot availability
- Pin configuration mapped to GPIO pins (refer to table)

OLED Display Module

- 0.96” or 1.3” screen, 128x64 resolution
- Supports I2C/SPI interfaces
- Displays real-time parking data and animations

Connectors, Adapter & LEDs

- Connectors: Insulated copper wires for stable signals
- DC Adapter: 5V regulated power for system components
- LEDs: Green = Available, Red = Occupied slots

Gate Control System Pin Configuration

- ESP8266 connected to servo motor for gate control
- Dedicated GPIO pins for servo and IR input
- Ensures responsive gate operation

Hardware Layer (2/2)

AUTOPARK OLED Animation

- Displays animation of approaching car
- Branding text: “Next-Gen Parking System”
- Shows live status of parking slots

Circuit Diagram & Dual ESP8266 Setup

- ESP8266 (1): Handles display + slot detection
- ESP8266 (2): Manages gate control + web communication

ESP8266 Microcontroller Module

- Acts as central control unit
- Wi-Fi-enabled 32-bit RISC processor (80–160 MHz)
- 64 KB SRAM, 4 MB Flash Memory
- Supports Wi-Fi 802.11 b/g/n with WPA/WPA2
- Ultra-low power consumption; ideal for IoT

Communication layer

01 Enables seamless data exchange between hardware, cloud, and users

02 Ensures secure and real-time system responsiveness

Wireless Communication

- Uses Wi-Fi (802.11 b/g/n) via ESP8266 module
- Supports HTTP requests (web communication)
- Uses MQTT protocol for low-latency messaging
- Ensures data security with TLS/SSL encryption
- Provides real-time, efficient data transfer

Cloud Integration

- Supports Firebase / AWS for cloud database storage
- Real-time sync of parking slot data to users
- Uses JSON format for efficient data handling
- Optimized protocols to reduce bandwidth and latency
- Ensures fast, secure, and scalable communication

Application layer

01 Manages user interface, data processing, and analytics

02 Users register via web portal → Unique QR code generated

03 Admin scans QR code to authorize entry/exit

04 Servo motor triggered on successful verification

05 Entry/exit timestamps recorded for billing

06 IR sensors monitor slots in real time

07 OLED display & cloud dashboard show live slot status

08 Auto billing system calculates charges based on duration

09 Supports digital payments: Credit Card, UPI, Wallets

10 Web app shows: slot availability, transaction history, cost estimate

11 User Site: autoparkltd.netlify.app

12 Admin Site: autoparkadmin.netlify.app

Methodology

Step 1: Requirements Gathering
- Urban parking needs analysis
- Defined scope: real-time monitoring, automation, reduced congestion

Step 2: Hardware Design & Selection
- ESP8266 for Wi-Fi and low power use
- IR sensors for vehicle detection
- OLED display for slot status
- Servo motors for automated gate control
- LEDs and power modules for reliability

Step 3: Software & Cloud Integration
- Web app for user/admin interface (HTML, CSS, JS)
- Firebase/AWS for real-time data sync and storage
- Secure API-based communication

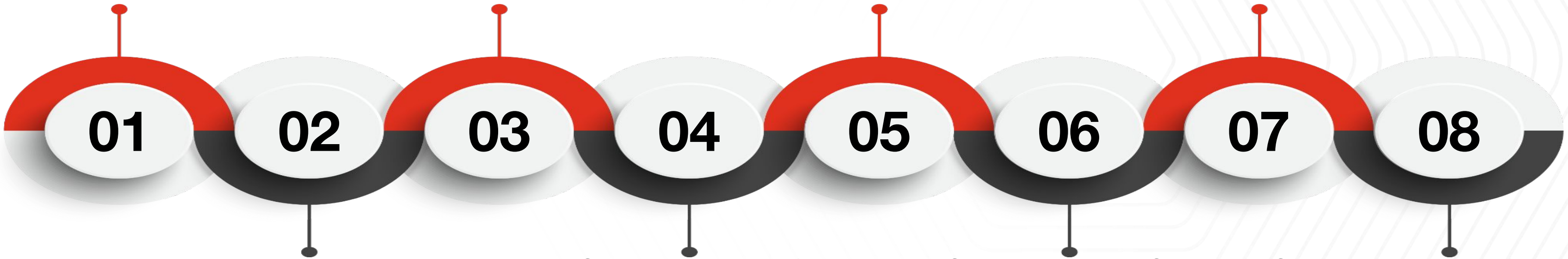
Step 4: Integration & Testing
- Hardware-software sync (IR, OLED, QR, servo)
- Functional + integration testing for performance

Step 5: QR Code Authentication
- Unique, encrypted QR for each user
- HTTPS-secured admin scanning and access control

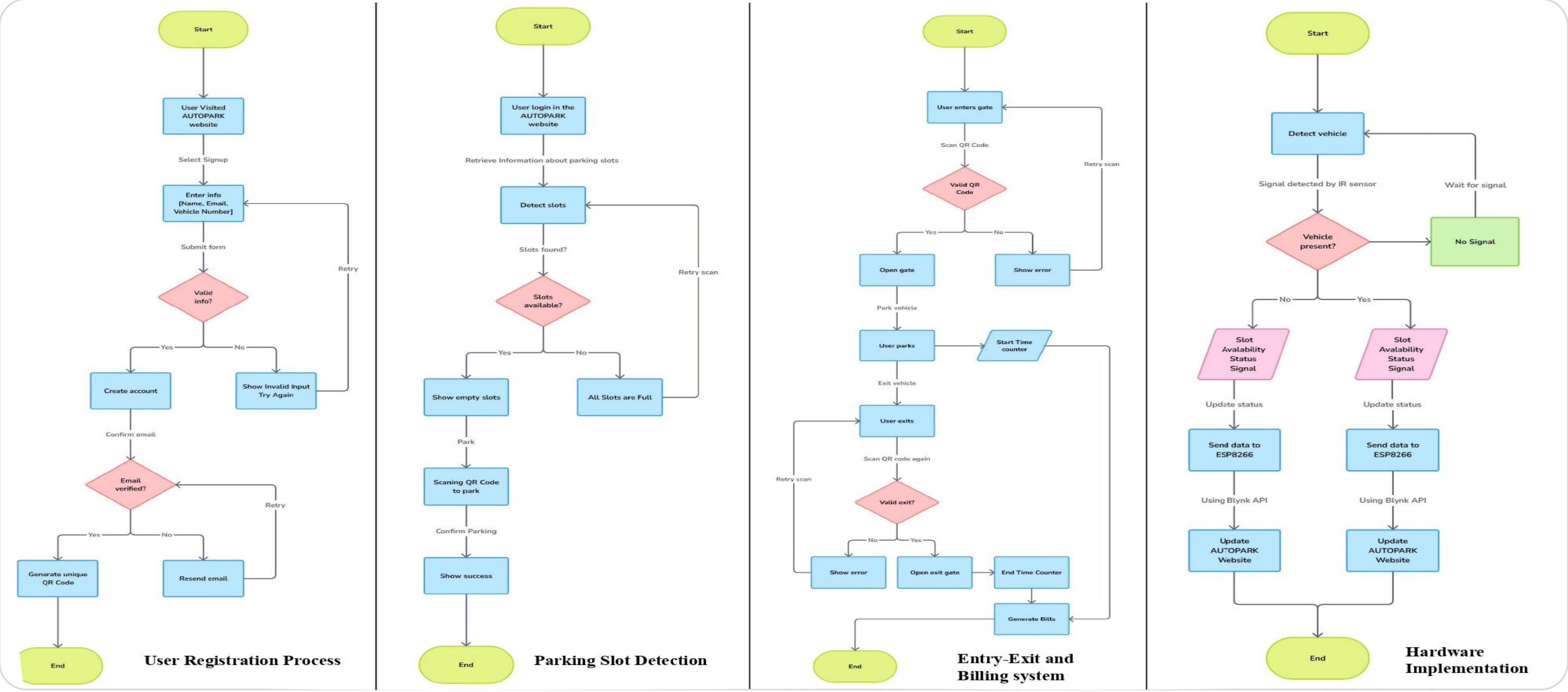
Step 6: Real-Time Slot Detection
- IR sensors → ESP8266 → Cloud → Web + OLED update

Step 7: Billing & Payment System
- Entry/exit time tracking
- Auto bill generation with UPI, card, wallet payments

Step 8: Final Testing & Deployment
- Real-world deployment + continuous monitoring
- User feedback loop for refinement and optimization



Workflow


Start
User Visited AUTOPARK website
Select Signup
Enter info [Name, Email, Vehicle Number]
Submit form
Valid info?
Yes
No
Create account
Show Invalid Input Try Again
Retry
Confirm email
Email verified?
Yes
No
Generate unique QR Code
Resend email
Retry
End
User Registration Process
Start
User login in the AUTOPARK website
Retrieve Information about parking slots
Detect slots
Slots found?
Slots available?
Yes
No
Show empty slots
All Slots are Full
Retry scan
Park
Scaning QR Code to park
Confirm Parking
Show success
End
Parking Slot Detection
Start
User enters gate
Scan QR Code
Valid QR Code
Yes
No
Open gate
Show error
Retry scan
Park vehicle
User parks
Start Time counter
Exit vehicle
User exits
Scan QR code again
Retry scan
Valid exit?
No
Yes
Show error
Open exit gate
End Time Counter
Generate Bills
End
Entry-Exit and Billing system
Start
Detect vehicle
Signal detected by IR sensor
Wait for signal
Vehicle present?
No Signal
No
Yes
Slot Avalability Status Signal
Slot Avalability Status Signal
Update status
Update status
Send data to ESP8266
Send data to ESP8266
Using Blynk API
Using Blynk API
Update AUTOPARK Website
Update AUTOPARK Website
End
Hardware Implementation

Algorithms (1/2)

Algorithm of User Registration Process

Start
User visits AUTOPARK website
Select "Signup"
Enter information: Name, Email, Vehicle Number
Submit form
Check if information is valid:
If No: Show "Invalid Input", Retry
If Yes: Proceed
Create account
Confirm email
Check if email is verified:
If No: Resend email, Retry
If Yes: Proceed
Generate unique QR Code
End

Algorithm of Parking Slot Detection

Start
User logs in to AUTOPARK website
Retrieve parking slot information
Detect slots
Check if slots are found:
If No: Retry slot detection
If Yes: Proceed
Check if slots are available:
If No: Show "All slots are full", End
If Yes: Proceed
Show empty slots
User parks and scans QR Code
Confirm parking
Show success
End

Algorithms (2/2)

Algorithm of Entry-Exit and Billing System

Start
User enters gate
Scan QR Code
Check if QR Code is valid:
If No: Show error, Retry
If Yes: Open gate
User parks vehicle
Start time counter
User exits vehicle
User exits parking
Scan QR Code again
Check if QR Code is valid for exit:
If No: Show error, Retry
If Yes: Open exit gate
End time counter
Generate bill
End

Algorithm of Hardware Implementation

Start
Detect vehicle using IR sensor
Wait for IR signal
Check if vehicle is present:
If No: Update status: Slot available
If Yes: Update status: Slot occupied
Send slot status data to ESP8266
Use Blynk API to update AUTOPARK website
End

Implementation

Hardware Integration

- Components: ESP8266, IR sensors, OLED, servo motors, 5V DC power supply
- Arduino IDE used for microcontroller programming
- Servo motors control gates based on QR code authentication

Network & Cloud Connectivity

- ESP8266 enables Wi-Fi data transfer
- Firebase used for real-time database and synchronization
- Secure API endpoints manage communication between system and cloud

Web Application Development

- Built using HTML, CSS, JS, Node.js, Express.js
- Functions: user registration, QR generation, slot monitoring, online payment

Security Measures

- QR codes encrypted
- HTTPS enforced for all data transfers
- Anomaly detection alerts for suspicious activities

System Testing

- Functional, load, and network security tests conducted
- Performance evaluated for multi-user scenarios

Results

System Performance

- Real-time vehicle detection with <1 sec response time
- Smooth gate automation using QR authentication
- OLED displays show live parking status

Cloud & Admin Efficiency

- Remote access to parking records and transaction logs
- Centralized dashboard for monitoring and reporting

Security and Stability

- Encrypted QR codes prevent unauthorized access
- HTTPS and anomaly detection enhance data protection

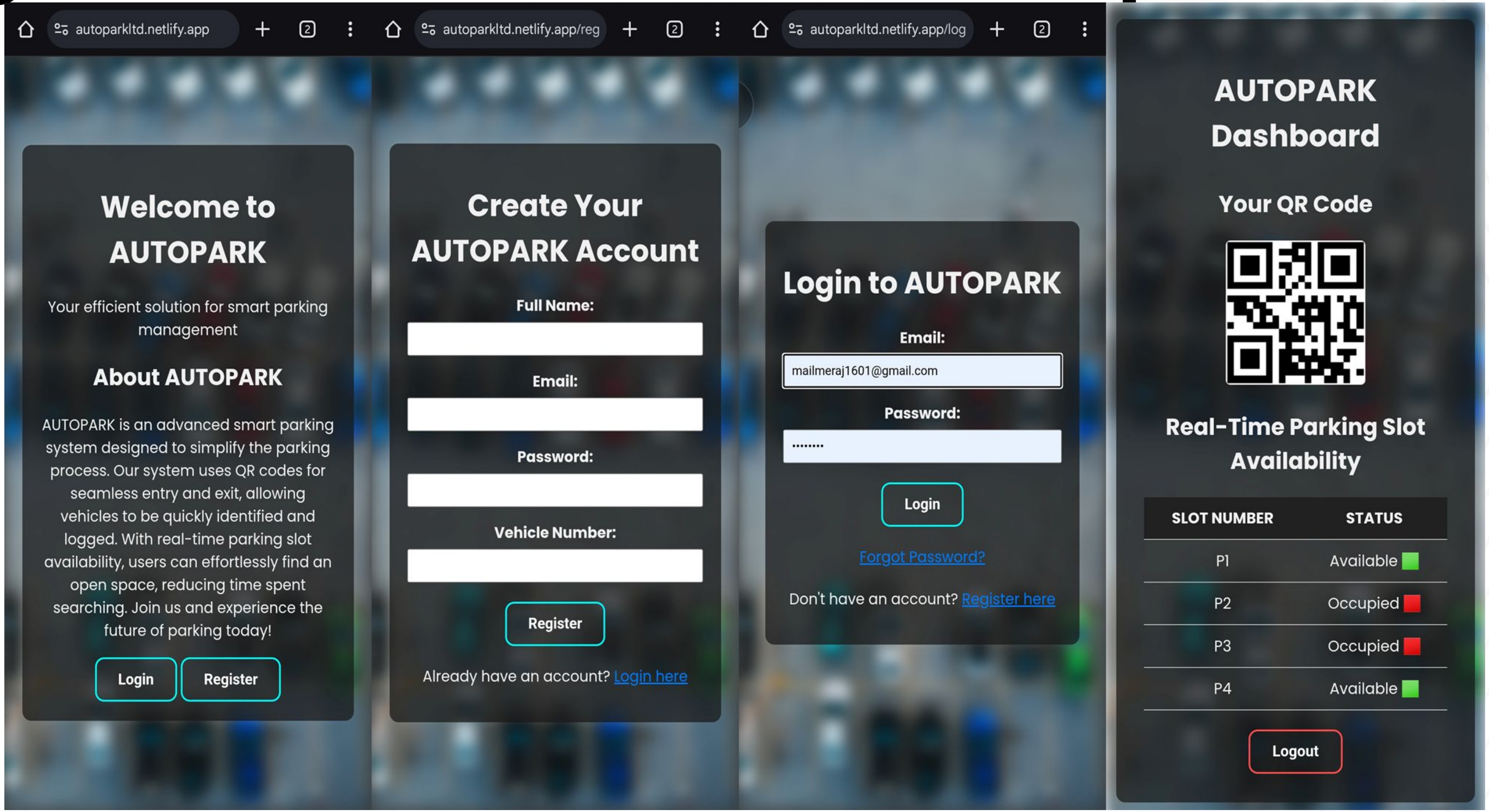
User Experience

- Easy registration, real-time slot updates, secure digital payments
- Positive feedback for system speed and reliability

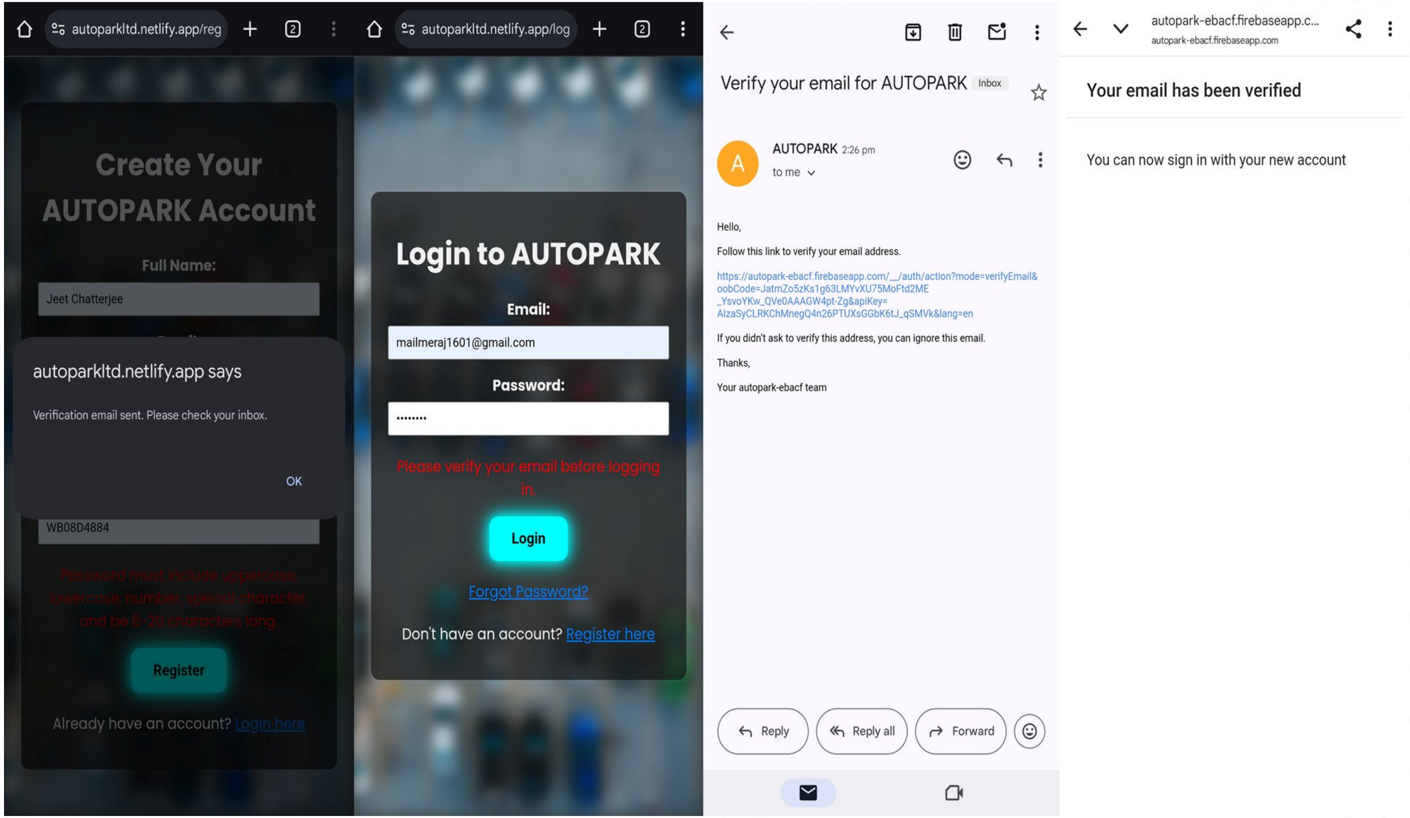
Improvement Areas

- Occasional sync delays due to network issues
- Future updates: edge computing & AI-based predictive analytics

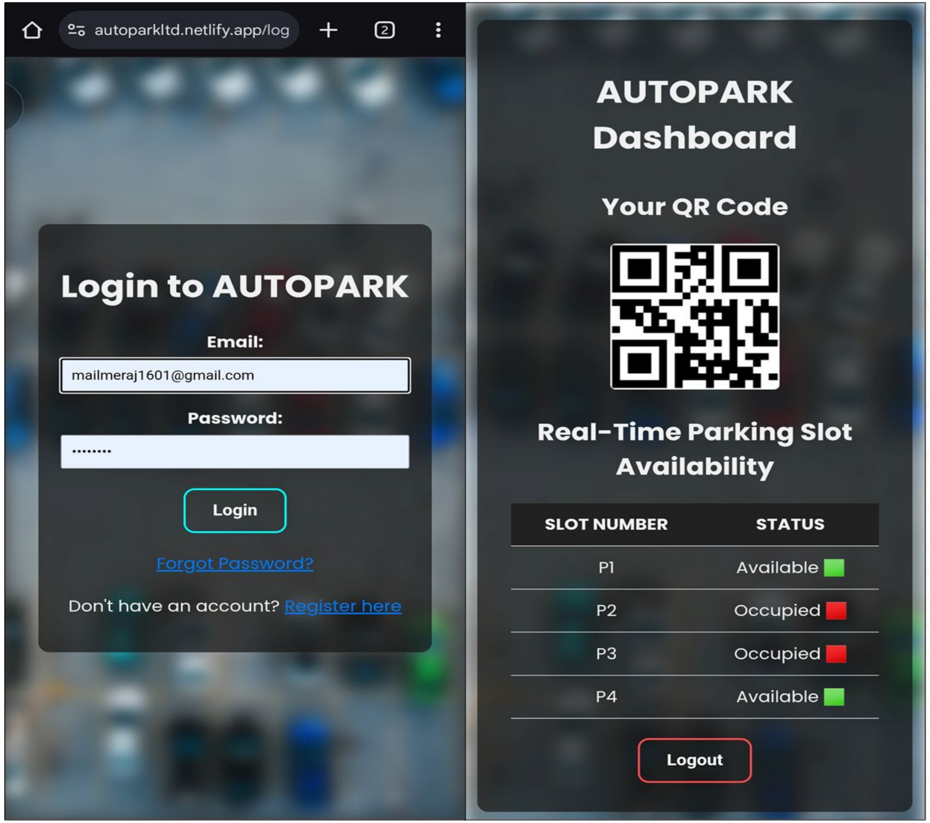
Registration Process of Autopark



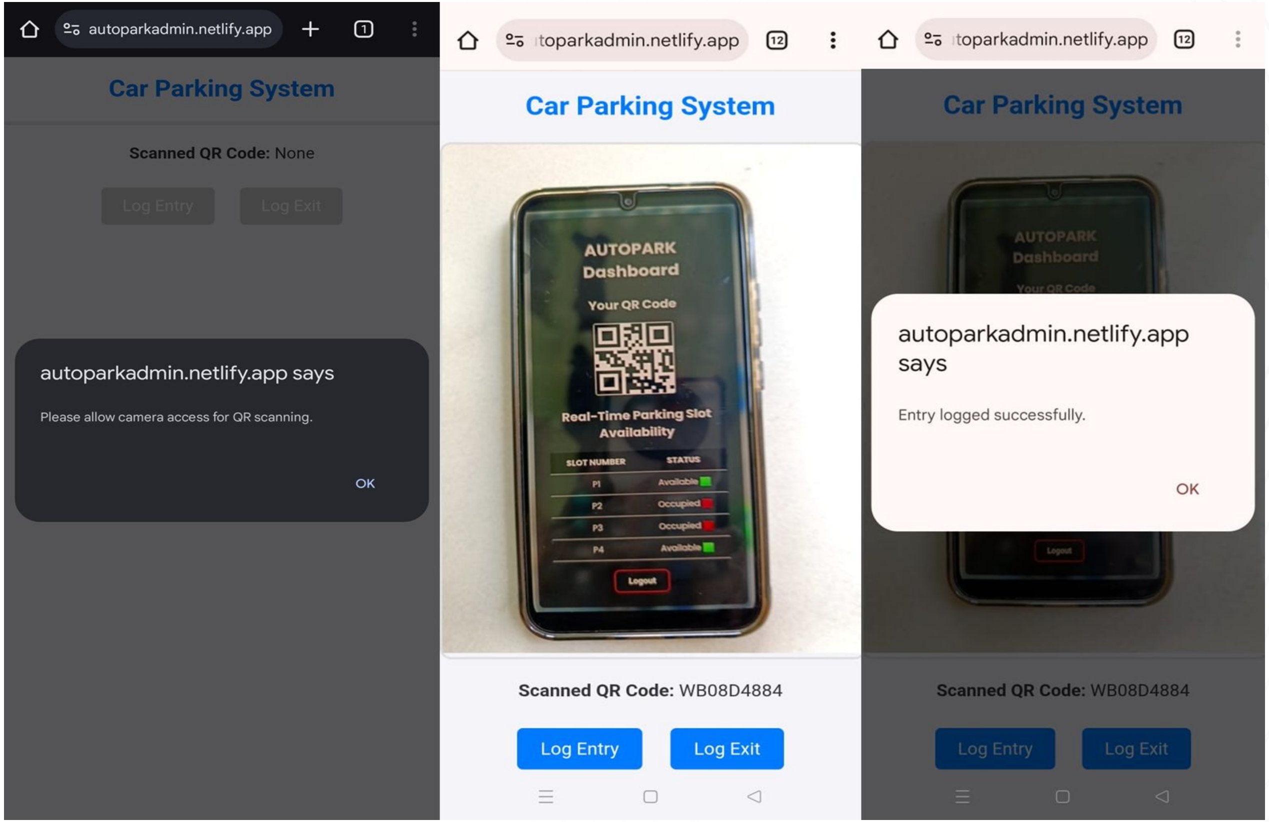
Email Verification Process of Autopark



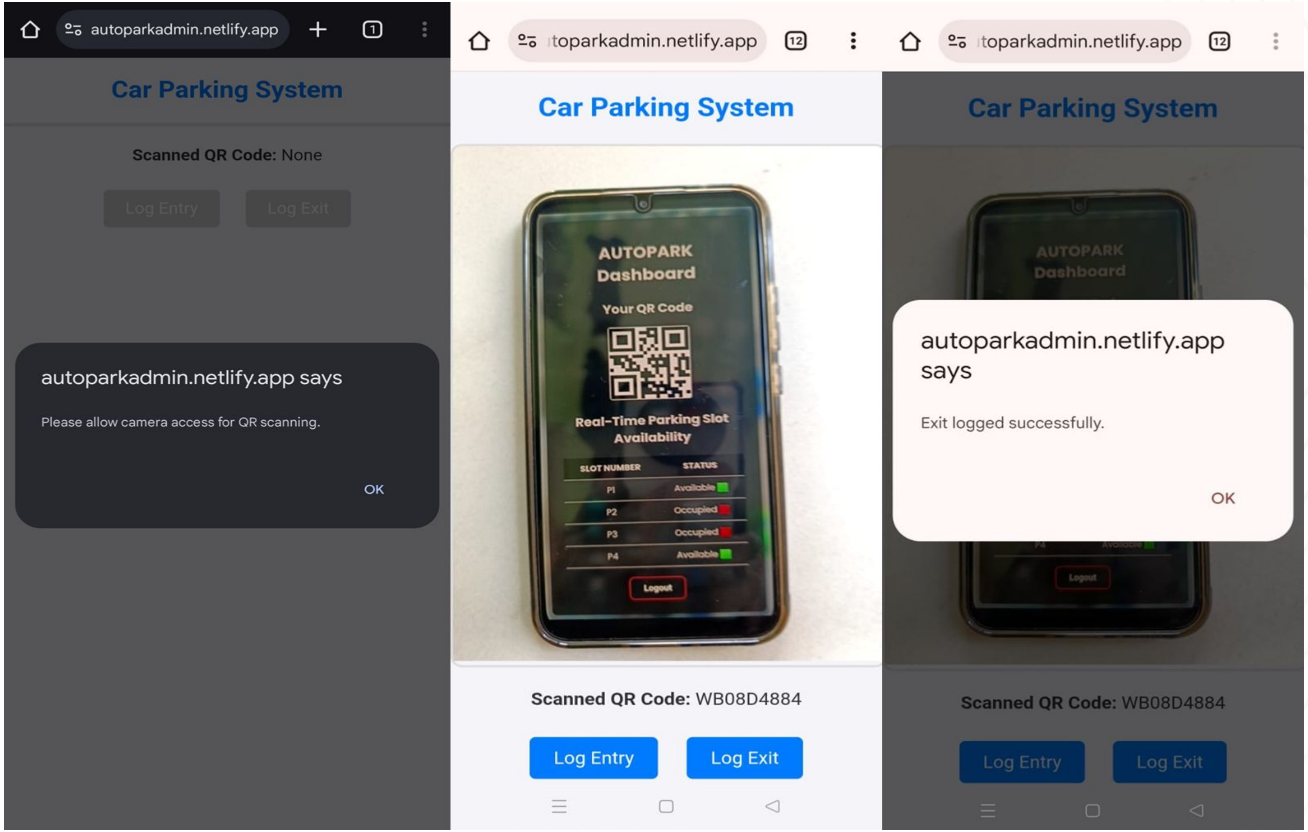
Login to Autopark



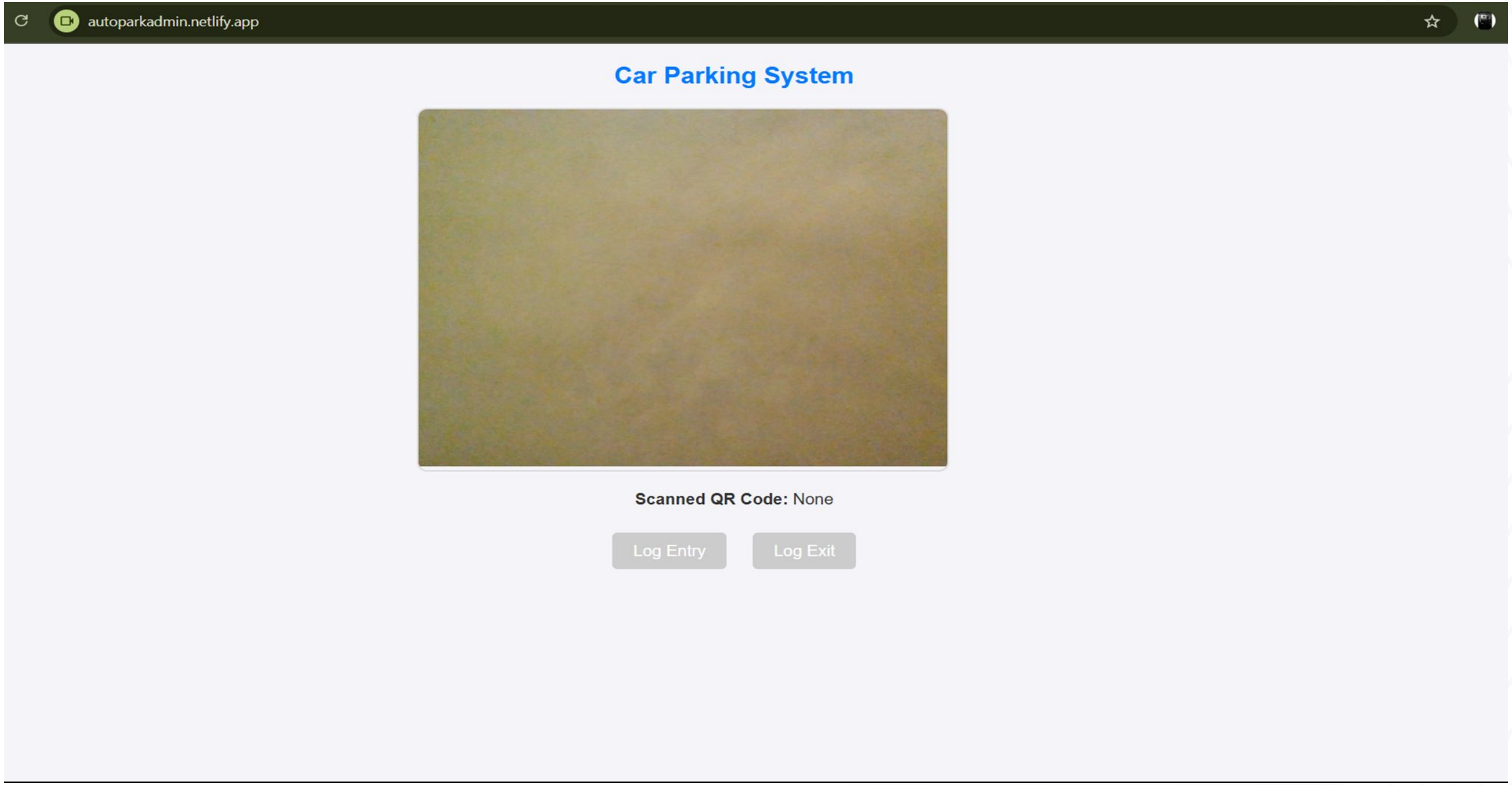
Autopark Log Entry Page



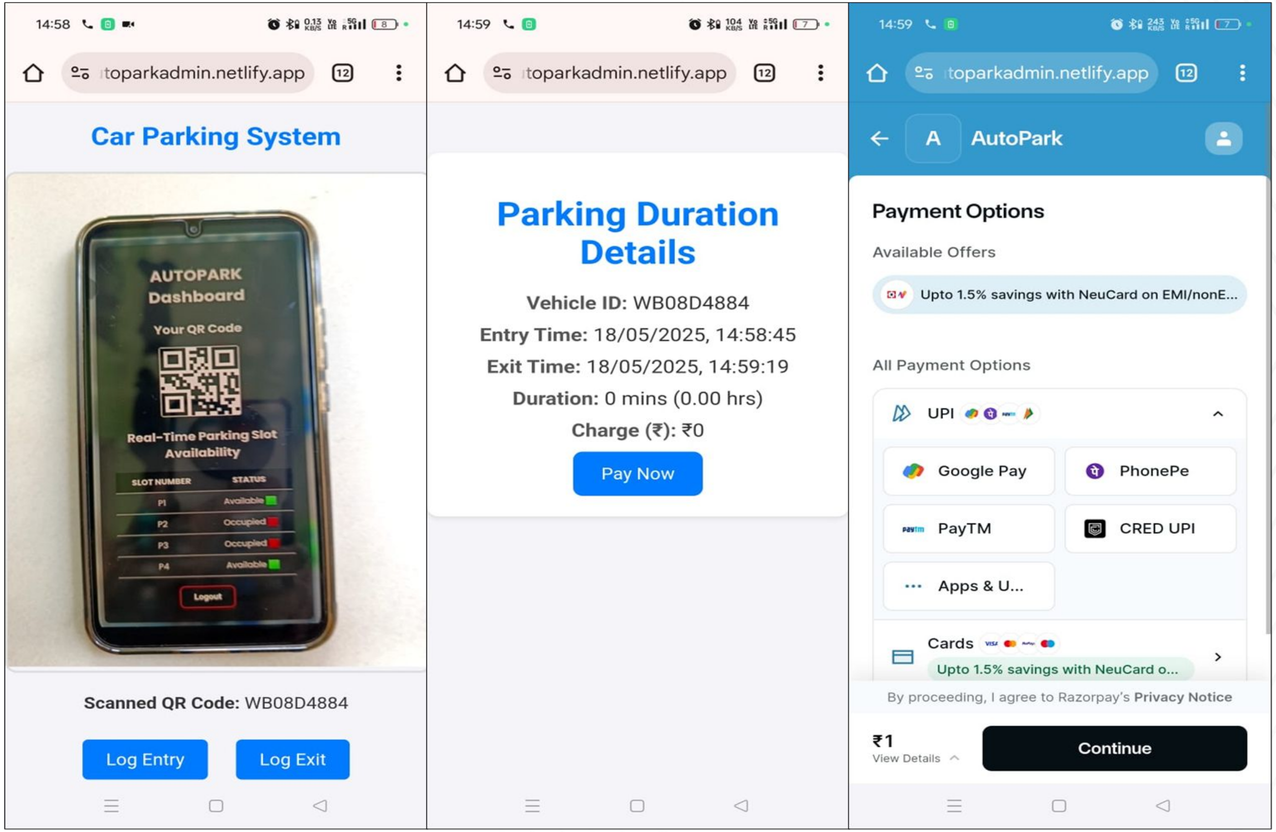
Autopark Log Exit Page



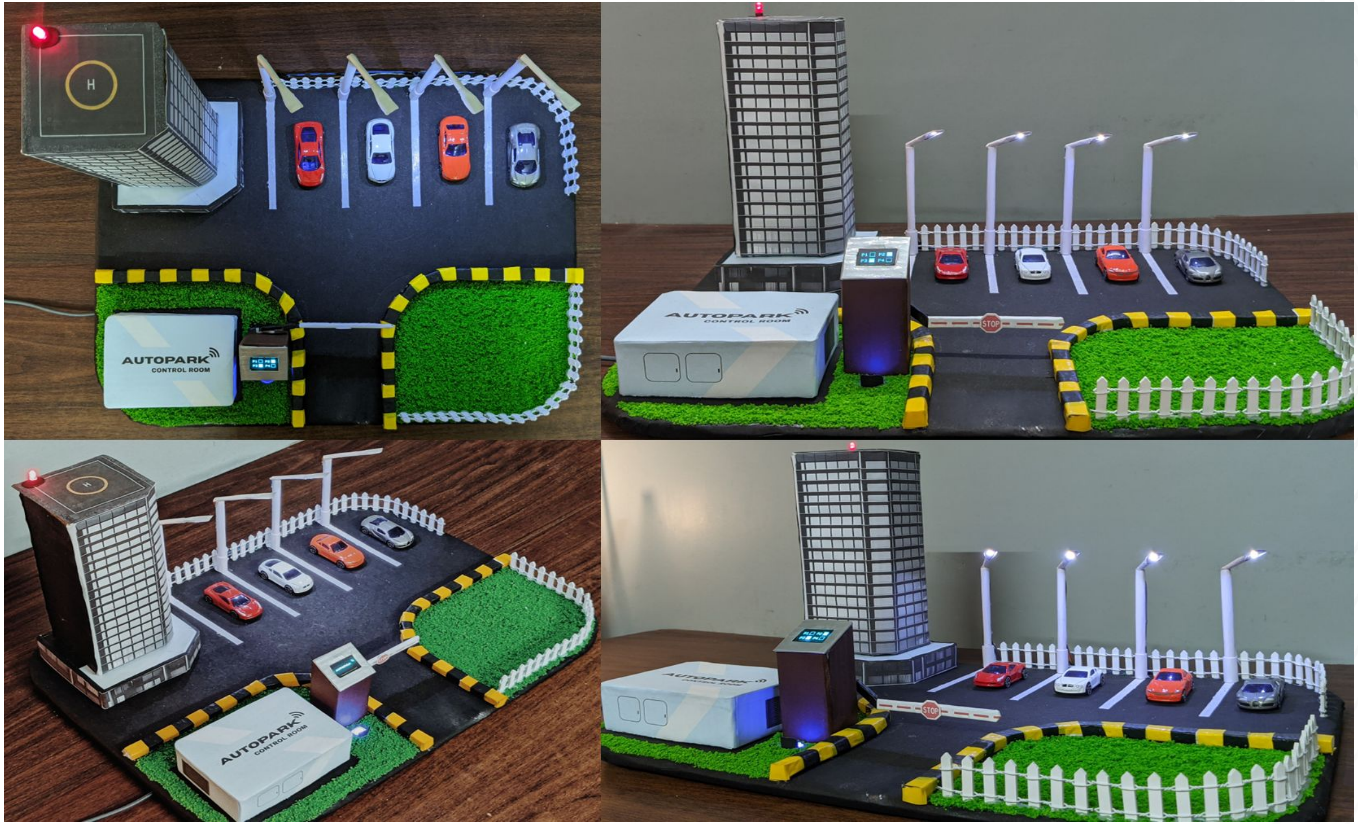
Autopark Admin Interface



Autopark Payment Gateway



Prototype Illustration



Challenges and limitations

01 Infrastructure & Deployment

Challenge: Retrofitting outdated facilities; high initial cost
Implementation: Modular hardware design & phased deployment

02 Network Reliability & Latency

Challenge: Unstable Wi-Fi, latency issues
Implementation: Use of mesh networks & hybrid communication models

03 Sensor Accuracy & Environment

Challenge: Weather interference, dust accumulation
Implementation: Regular maintenance & adaptive calibration

04 Security & Privacy

Challenge: QR code forgery, data breaches
Implementation: End-to-end encryption & multi-factor authentication

05 Cost & Maintenance

Challenge: High setup cost for small entities
Implementation: Open-source platforms & predictive maintenance tools

06 User Adoption

Challenge: Resistance from low digital literacy regions
Implementation: Intuitive UI/UX & awareness campaigns

Future Vision

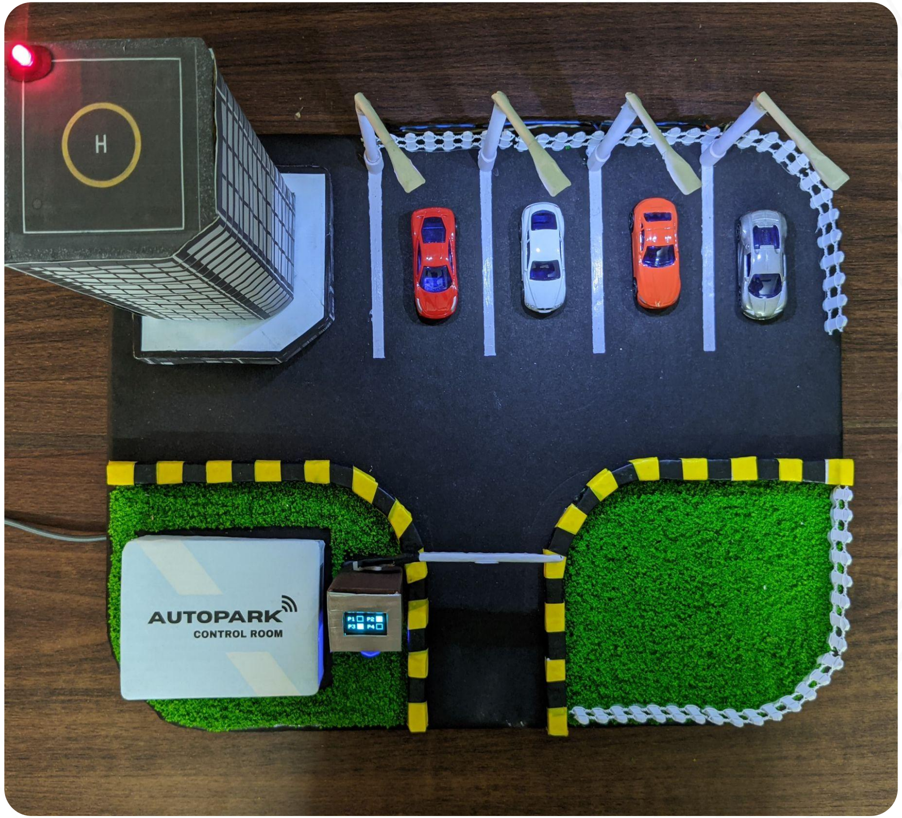
AI-driven slot reservation, blockchain security, voice navigation

Future Scope

- Integration with Payment Systems
- Mobile App Development
- Advanced sensors and machine learning
- Real time parking slot reservation
- Integration with smart city platforms

Conclusion

- **Revolutionizing Parking with IoT:**
  - Combines ESP8266 and Blynk app for real-time parking management.
  - Enhances convenience, efficiency, and user accessibility.
- **Scalable and Sustainable:**
  - Adaptable for residential, commercial, and public parking needs.
  - Supports smart city goals by reducing fuel consumption and emissions.
- **Future-Ready Innovation:**
  - Opportunities to integrate AI for predictive parking and optimization.
  - Enhanced data security measures to protect user information.
- **Impact:**
  - AUTOPARK is a step toward sustainable, efficient, and intelligent urban mobility.



References

- Javaheri, Atusa, et al. "Evaluating Impact of Smart Parking Systems on Parking Violations." IEEE Access (2024). https://doi.org/10.1109/ACCESS.2024.3503513
- [24] Jenila, C., and K. Harshan. "IoT-based Smart Parking System: Hardware-Centric Approach for Addressing Urban Parking Challenges." 2024 2nd International Conference on Networking, Embedded and Wireless Systems (ICNEWS). IEEE, 2024. https://doi.org/10.1109/ICNEWS60873.2024.10731004
- [25] Piruthiviraj, P., et al. "IoT based Vehicle Monitoring System for Safety Standards in Smart Cities." 2024 4th International Conference on Pervasive Computing and Social Networking (ICPCSN). IEEE, 2024. https://doi.org/10.1109/ICPCSN62568.2024.00110
- [27] Araújo, Anderson, et al. "Reliability analysis of an IoT-based smart parking application for smart cities." 2017 IEEE International Conference on Big Data (Big Data). IEEE, 2017. https://doi.org/10.1109/BigData.2017.8258426
- [39] Alam, Md Rokibul, et al. "A Survey on IoT Driven Smart Parking Management System: Approaches, Limitations and Future Research Agenda." IEEE Access 11 (2023): 119523-119543. 10.1109/ACCESS.2023.3327306

Thank you

Submitted by:

JENIFA KHATUN	(ROLL NO: 18700321048)
JEET CHATTERJEE	(ROLL NO: 18700321084)
ARCHISHMAN DUTTA	(ROLL NO: 18700321049)
ADDRI KIRAN DEY	(ROLL NO: 18700321044)
ABHIPSA CHAKRABORTY	(ROLL NO: 18700321060)

Under the supervision of

PROF. (Dr.) DIPSIKHA DAS