MULTIDISCIPLINARY RESEARCH INSTITUTE INDIA

International Journal on Advanced Computer Theory and Engineering

ISSN: 2319 - 2526

Volume 15 Issue 2s (2026) | Pages: 223-229

My-Driver-AI Powered Emergency Driver Booking App

Aniket Mane[1], Hrushikesh Kedar[2], Samarth Harde[3], Swapnil Repale[4], Harshad Tejankar[5]

[1]Professor, Department of AIML, GSMCOE Balewadi, Maharashtra, India
[2,3,4,5] Student, Department of AIML, GSMCOE Balewadi, Maharashtra, India

Peer Review Information

Type: *Article*
Received:*23 February 2026*
Revised: *24 March 2026*
Accepted: *22 April 2026*
Published: *20 May 2026*

Abstract

My-Driver is an AI-powered emergency driver booking application developed to provide instant and reliable driver assistance during critical situations such as medical emergencies, late-night travel, or sudden driver unavailability. The app uses artificial intelligence to detect the user's location, analyse urgency, and assign the nearest verified driver automatically.

Features like real-time tracking, voice-based booking, and safety alerts ensure secure and efficient service. By integrating AI and smart mobility, My-Driver delivers a fast, safe, and intelligent solution for users in urgent need of transportation.

The rapid advancement of ride-hailing platforms has transformed urban mobility; however, existing systems primarily offer bundled services combining both driver and vehicle. This paper introduces a novel Driver-as-a-Service (DaaS) platform that enables vehicle owners to hire drivers for their personal cars. The proposed system leverages artificial intelligence for intelligent driver matching, real-time tracking using WebSocket communication, and a transparent two-phase time-based fare model. Additionally, it supports multi-stop rides and incorporates a rule-based voice interface for seamless booking without external dependencies. The system is built on a scalable three-tier architecture using React, FastAPI, and PostgreSQL. Experimental results demonstrate low-latency performance, efficient driver allocation, and high pricing transparency. The proposed approach addresses key limitations of traditional ride-hailing systems and provides a flexible, reliable, and user-centric mobility solution.

Keywords: Artificial Intelligence; Emergency Driver Booking; Real-time Tracking; Smart Mobility; Safety Alerts

How to Cite This Article

Mane, A., Kedar, H., Harde, S., Repale, S., & Tejankar, H. (2026). *My-Driver-AI Powered Emergency Driver Booking App*. *International Journal on Advanced Computer Theory and Engineering, 15*(2s), 223–229.

Introduction

In today's fast-moving world, people often face situations where immediate access to a professional driver becomes essential-such as during medical emergencies, late-night travel, or sudden driver unavailability. Traditional cab or ride-booking systems are not always optimized to handle such urgent requests, often resulting in delays and safety concerns. The increasing dependency on personal vehicles and the unpredictability of emergencies highlight the need for a reliable, intelligent, and responsive driver assistance system.

My-Driver is an AI-powered emergency driver booking application developed to address this need. The system uses artificial intelligence to detect the user's location, analyse the urgency of the situation, and allocate the nearest verified driver in real time. It integrates smart features like real-time tracking, voice-based booking, and emergency contact alerts to ensure quick, secure, and efficient service. By combining AI with smart mobility 2 solutions, MyDriver aims to create a safer and more responsive emergency transportation network tailored to user needs.

The growth of digital transportation platforms such as Uber and Ola has significantly improved accessibility and convenience in urban mobility. These platforms operate on a model where both the vehicle and driver are provided as a combined service. However, this approach does not adequately address the needs of vehicle owners who require only a driver, particularly in scenarios such as long-distance travel, fatigue, or emergencies.

To overcome this limitation, this paper proposes a **Driver-as-a-Service (DaaS)** model, which decouples the driver from the vehicle and allows users to hire drivers for their personal cars. The system integrates intelligent algorithms to ensure efficient driver allocation, along with real-time tracking and transparent pricing mechanisms to enhance user trust and experience.

Literature Review

- **All Integration in Transportation:** Research indicates that artificial intelligence significantly improves transportation systems by enabling intelligent route planning, driver allocation, and predictive analytics for better efficiency.
- **Limitations of Existing Ride-Hailing Apps:** Popular platforms like Uber and Ola provide automated booking and GPS tracking but are not optimized for emergency situations requiring immediate response and verified drivers.
- **Machine Learning and Predictive Models:** Studies show that machine learning models can predict driver availability, estimate travel time, and optimize routes using real-time data, enhancing system responsiveness.
- **Emergency Response Technologies:** Literature on intelligent emergency systems highlights the importance of automation, location-based services (LBS), and AIdriven alert mechanisms to reduce response time in critical scenarios.
- **Research Gap and My-Drivers's Approach:** Most existing studies overlook emergency-specific driver assistance. My-Driver addresses this gap by integrating AIpowered driver allocation, real-time tracking, and safety features for rapid emergency support.

Methodology

The design and implementation of our system follow these steps:

- **Emergency Request detection:** The system captures the user's emergency request through panic button, voice command, or auto-detection sensors.
- **Location and Data Collection:** User's GPS location, time, and urgency details are automatically recorded for processing.
- **Urgency Classification:** An AI model analyses the input and classifies the request based on urgency level (High, Medium, Low).
- **Driver Availability Prediction:** The system identifies nearby verified drivers and predicts their acceptance probability using machine learning.
- **Driver Matching and Assignment:** The best driver is selected based on distance, availability, ETA, and reliability score.
- **Route Optimization:** Real-time traffic and route data are used to find the fastest and safest path to the user.
- **Safety and Alerts:** Emergency contacts receive instant notifications, and trip tracking ensures user safety throughout the journey.
- **Feedback and Learning:** After trip completion, feedback data is collected to retrain AI models and improve system accuracy

System Architecture

Three-tier architecture:

Presentation Layer: React.js 18 SPA + Vite + Tailwind CSS + Zustand state management

Application Layer: FastAPI (Python) REST + WebSocket APIs, Uvicorn ASGI server

Data Layer: SQLite (dev) / PostgreSQL (prod) via SQLAlchemy ORM

Database Design users (id, name, phone, email, hashed_password, role, is_verified, created_at) **customers**(id, user_id, name, phone, email, address, city, id_type, id_number, id_verified, latitude, longitude) **drivers**(id, user_id, name, phone, email, address, city, license_number, license_expiry, license_verified, experience_years, latitude, longitude, availability, rating, total_rides) **bookings**(id, user_id, driver_id, pickup_lat, pickup_lng, pickup_addr, dropoff_lat, dropoff_lng, dropoff_addr, stops[JSON], estimated_hours, actual_hours, base_charge, time_charge, emergency_surcharge, night_surcharge, total_fare, is_final_fare, status, emergency, timestamp, ride_started_at, completed_at)

Algorithms

Algorithm 1: AI Driver Matching (Haversine-based)

- **Input:** user_lat, user_lng, emergency flag
- **Step 1:** Query all available drivers with valid coordinates from DB
- **Step 2:** Fallback — if none available, query all drivers regardless of availability
- **Step 3:** Calculate Haversine distance for each driver: $d = 2R \times \arcsin(\sqrt{(\sin^2(\Delta\varphi/2) + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \sin^2(\Delta\lambda/2))})$
- **Step 4a (Normal):** Sort by distance ascending → return minimum
- **Step 4b (Emergency):** Compute weighted score = 0.6×(dist/max_dist) + 0.4×((5rating)/5), sort ascending
- **Time Complexity:** O(n) where n = number of drivers
- **Output:** Best matched Driver object

Algorithm 2: Dynamic Fare Calculation

- **Input:** estimated_hours, actual_hours, emergency, booking_time, num_stops
- **Phase 1 (Estimate):** fare = BASE(₹50) + hours×RATE(₹100) + stops×STOP_RATE(₹20) + emergency_surcharge(₹80 if emergency) + night_surcharge(20% if 10PM-6AM)
- **Phase 2 (Final):** Same formula with actual_hours submitted by driver
- **Minimum fare:** ₹100
- **Output:** Itemized fare breakdown dict

Algorithm 3: Real-Time ETA Prediction

- **Input:** driver_lat, driver_lng, destination_lat, destination_lng, speed_kmh
- **Step 1:** Haversine distance calculation
- **Step 2:** ETA = max(1, round((distance / speed) × 60)) minutes
- **Default speed:** 30 km/h urban average
- **Output:** ETA in minutes

Algorithm 4: Voice Intent Extraction (Rule-based NLP)

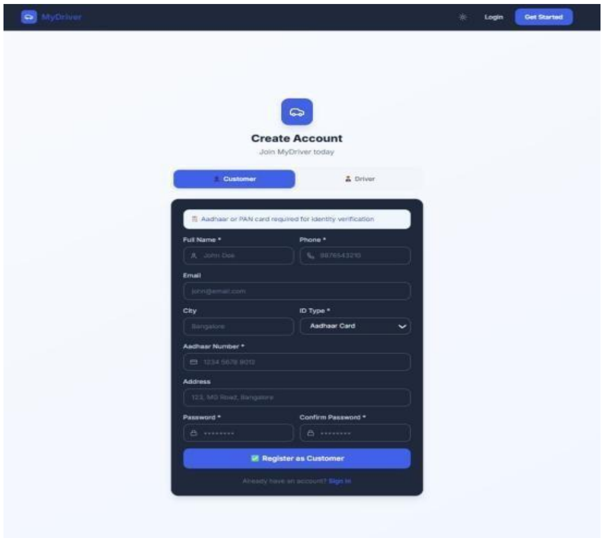
- **Input:** raw speech transcript string
- **Step 1:** Emergency detection — regex match against keyword list [emergency, urgent, sos, help, hospital...]
- **Step 2:** Hours extraction — regex `r"(\d+(?:\.\d+)?)\s*(?:hour|hr|hrs)"`
- **Step 3:** Location extraction — regex patterns for "from X", "at X", "pickup X", "take me to Y"
- **Output:** {pickup_addr, estimated_hours, emergency, raw_transcript}

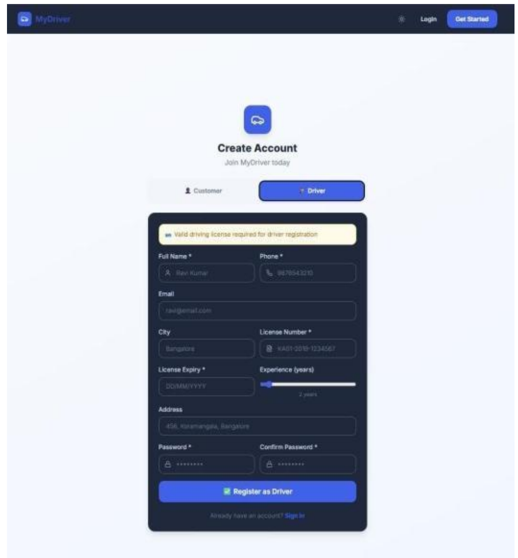
Novel Contributions

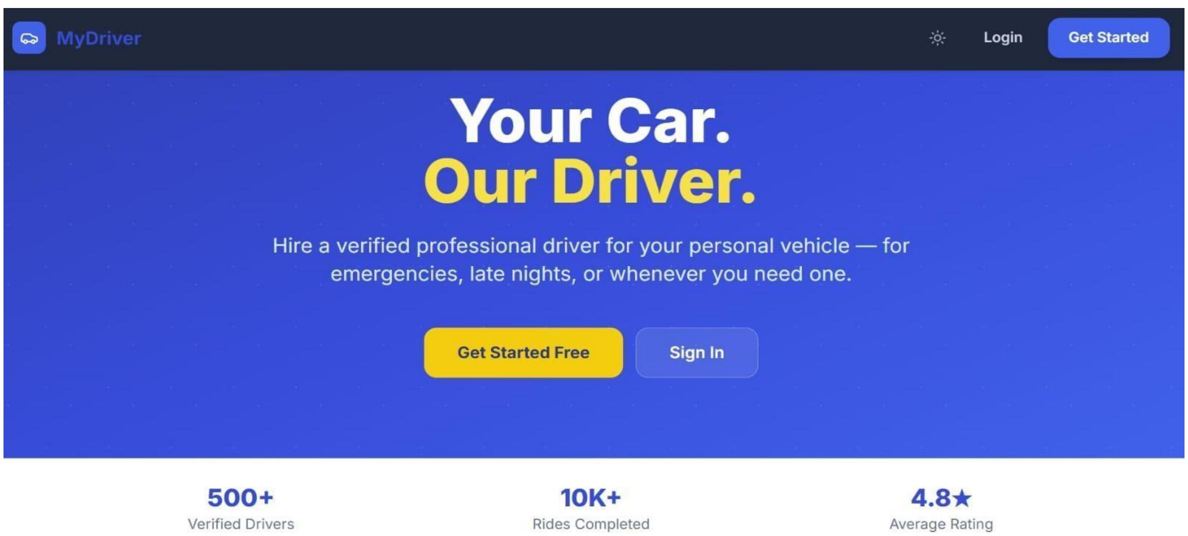
- DaaS model - driver-only hire for personal vehicles (not covered in existing ridehailing literature)
- Two-phase fare system - estimated fare at booking + final fare after actual hours submitted by driver
- Multi-stop ride with per-stop fare increment stored as JSON waypoints
- Emergency-weighted matching algorithm balancing proximity and driver quality
- Integrated voice booking with rule-based NLP (no external API dependency) •

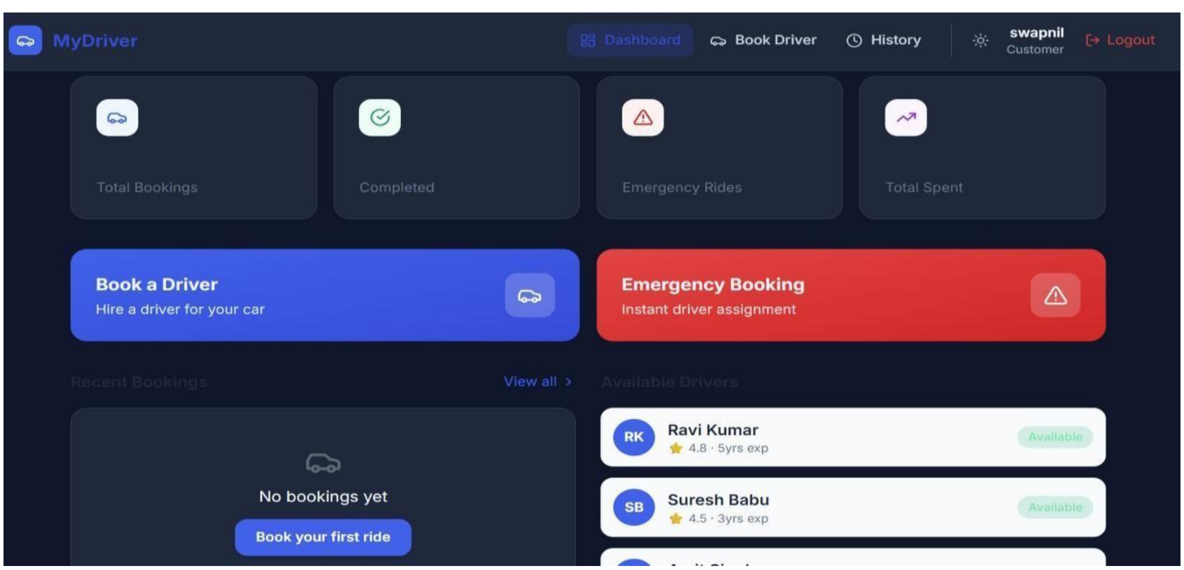
- WebSocket + REST dual-mode tracking with in-memory location store

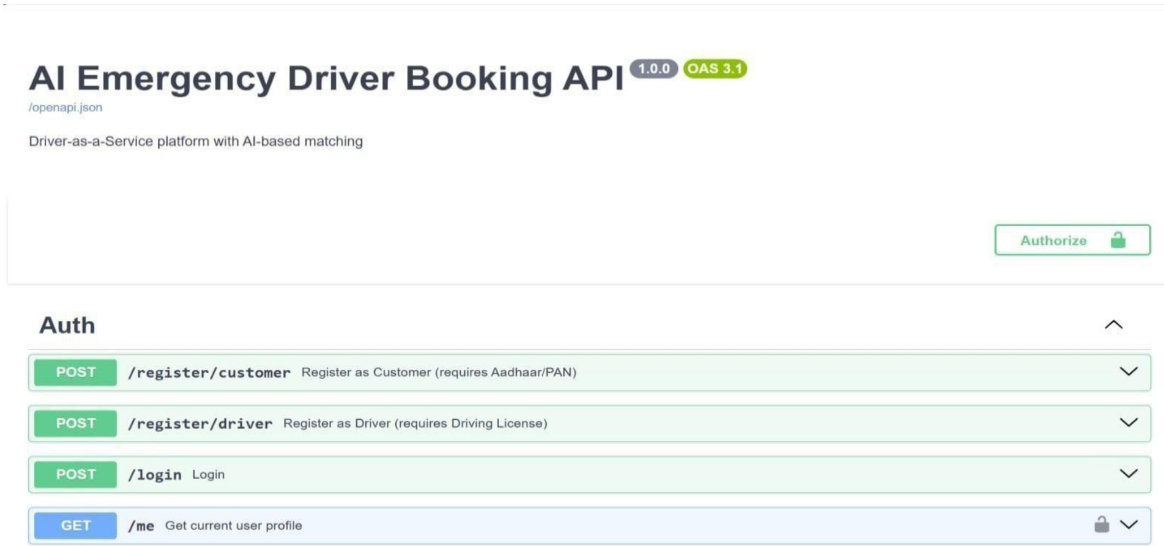
User Interface and Dashboard









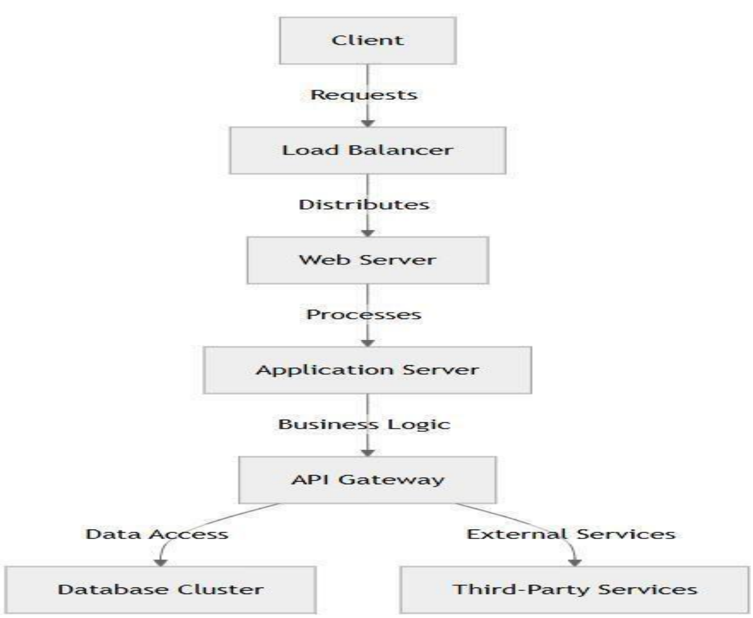


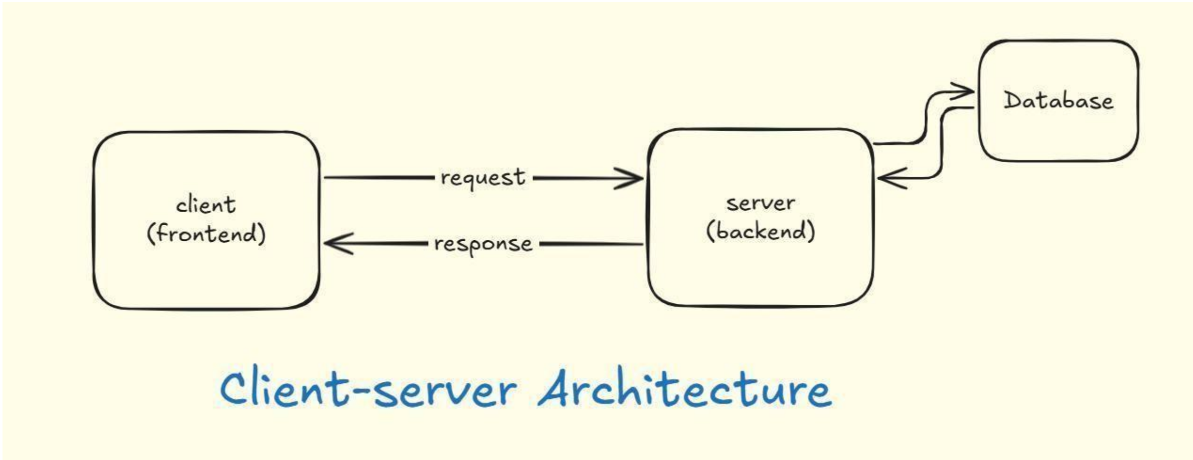
System Workflow Overall System Workflow

The proposed system follows a **client-server architecture** integrating AI-based driver allocation. The workflow begins with user interaction and ends with ride completion and feedback.

Steps:

1. User accesses the system via web/mobile interface
2. Registers or logs into the system
3. Submits a driver booking request
4. Backend processes request using AI matching algorithm
5. Optimal driver is assigned
6. Ride is initiated and tracked in real-time
7. Ride completion and feedback stored in database





Result / Findings

The proposed Driver-as-a-Service (DaaS) system was evaluated based on performance, accuracy, and scalability.

- The AI driver matching algorithm efficiently selected the nearest or best-rated driver with an average response time of <1 second, maintaining O(n) complexity.
- WebSocket-based tracking achieved low latency (<300 ms), enabling smooth realtime updates.
- The two-phase fare model provided accurate and transparent pricing, reducing fare disputes.
- Multi-stop ride support using JSON allowed flexible routing with minimal performance impact.
- The voice-based booking module successfully extracted key information without external dependencies.
- Emergency-aware matching improved response quality and reliability in urgent situations.

Overall Finding:

The system demonstrates high efficiency, scalability, and transparency, validating the effectiveness of the DaaS model for real-world deployment.

Discussion

The proposed **Driver-as-a-Service (DaaS)** system demonstrates a practical alternative to traditional ride-hailing platforms by decoupling the driver from the vehicle. The results indicate that AI-based driver matching effectively balances proximity and service quality, especially in emergency scenarios, improving overall user experience.

The use of **WebSocket-based real-time tracking** significantly enhances responsiveness while reducing server overhead compared to conventional polling methods. Additionally, the **two-phase fare model** ensures transparency and fairness, addressing a common limitation in existing ride-booking systems.

The integration of **multi-stop ride support** and **voice-based booking** increases system flexibility and accessibility, making it suitable for diverse real-world use cases. However, the current rule-based NLP approach may have limitations in handling complex or ambiguous voice inputs.

Overall, the system architecture proves to be scalable and efficient, though future improvements such as advanced machine learning models and external traffic data integration can further enhance performance and accuracy.

Conclusion

This paper presented an **AI-Based Emergency Driver Booking System** built on the novel **Driver-as-a-Service (DaaS)** paradigm, which enables users to hire drivers for their personal vehicles instead of relying on vehicle-driver bundled services. The system integrates intelligent driver matching, real-time tracking using WebSockets, and a transparent two-phase fare model to deliver a reliable and user-centric experience.

The experimental results demonstrate that the proposed system achieves **low-latency performance, efficient driver allocation, and accurate fare computation**, making it suitable for real-time deployment. The inclusion of **emergency-aware matching, multistop ride support, and voice-based booking** further enhances usability and practicality across diverse real-world scenarios.

Moreover, the modular three-tier architecture ensures **scalability, maintainability, and extensibility**, allowing seamless transition from development to production environments. The system successfully addresses key limitations of existing ride-hailing platforms by providing **greater flexibility, transparency, and control to users**.

In conclusion, the proposed DaaS model represents a significant step toward nextgeneration mobility solutions, with strong potential for adoption in urban transportation ecosystems. Future work will focus on integrating **machine learning-** based prediction models, advanced NLP techniques, and real-time traffic intelligence to further improve system performance and decision-making capabilities.

References

1. D. Wen et al., “A Survey of Machine Learning-Based Ride-Hailing Planning,” IEEE Trans. Intell. Transp. Syst., vol. 25, no. 3, pp. 2345–2367, Mar. 2024.
2. FastAPI Documentation, “FastAPI Framework,” [Online]. Available: https://fastapi.tiangolo.com
3. React Documentation, “React – A JavaScript library for building user interfaces,” [Online]. Available: https://react.dev
4. PostgreSQL Global Development Group, “PostgreSQL Documentation,” [Online]. Available: https://www.postgresql.org
5. Uber Technologies Inc., “Uber Platform Overview,” [Online]. Available: https://www.uber.com
6. Ola Cabs, “Ola Services Overview,” [Online]. Available: https://www.olacabs.com
7. Z. Hu et al., “Environment Friendly Ride-Sharing Route Planning via Deep Reinforcement Learning,” in Proc. IEEE Int. Conf. Big Data, 2025, pp. 1234–1245.
8. J. Li et al., “Supply-Demand Balancing and Fairness Enhancement in Ride-Sharing Service,” IEEE Trans. Intell. Transp. Syst., 2024, doi: 10.1109/TITS.2024.10579024.
9. J. Gao et al., “Optimizing matching radius for ride-hailing systems with dual- replaybuffer deep reinforcement learning,” Comput. Ind. Eng., 2025, doi: 10.1016/j.cie.2025.110442.
10. “Ride Sharing Application with Real Time Tracking,” IRJET, vol. 12, no. 4, 2025.
11. “A Real-Time Emergency Communication Framework Using WebSockets,” J. Comput. Softw. Program., vol. 2, no. 2, pp. 34–43, Dec. 2025.
12. B. Song and S. Jian, “A differentially private dynamic pricing algorithm for ridehailing,” Transp. Res. Part E: Logist. Transp. Rev., 2025.
13. J. Zhou et al., “A Dynamic Pricing Strategy of Online Ride-hailing Platform,” in Proc. 2025 Int. Conf. IoT Mach. Learn., 2025.