International Conference on Advances in Modern Technology of Research in Engineering Field (AIMTREF) April, 2026
Organized By: SKN Sinhgad Institute of Technology& Science, Kusgaon (Bk), Lonavala, Pune

International Journal of Science, Engineering and Technology
ISSN: 2348-4098, P-ISSN: 2395-4752

RideLink – An Android-Based Intelligent Carpooling System for Sustainable Urban Mobility

Associate Professor S. P. Gunjal, Sahil Hase, Pratik Jadhav, Abhishek Pokharkar, Siddhi Horane

Department of Computer Science, SKN Sinhgad Institute of Technology & Science, Lonavala, Maharashtra, India

Abstract- The increasing number of private vehicles in urban areas has resulted in serious transportation challenges such as traffic congestion, fuel wastage, and rising pollution levels. This paper presents the complete design, development, and evaluation of RideLink — an Android-based intelligent carpooling system engineered to promote shared mobility and sustainable commuting. RideLink allows users to register as drivers or passengers and enables them to create, search, and book rides in real time. The system integrates Firebase Realtime Database for secure data handling, osmdroid (OpenStreetMap) for GPS-based route visualization, and a Haversine geospatial algorithm for intelligent ride matching based on route proximity. The platform supports real-time seat occupancy tracking, gender-aware booking, driver license verification through an admin approval workflow, luggage capacity management, automated SMS notifications to drivers upon booking, an SOS emergency button, and complete ride lifecycle management. Testing across physical Android devices running API levels 26 through 34 demonstrated sub-2-second ride search performance and accurate geolocation-based matching. All 14 planned functional test cases passed successfully. RideLink demonstrates how mobile and cloud technologies can deliver a practical, scalable, and eco-friendly transportation solution without paid infrastructure.

Keywords- Android, Carpooling, Firebase, Haversine Algorithm, OpenStreetMap, RideLink, Real-Time Ride Sharing, SOS, Sustainable Mobility, License Verification.

I. INTRODUCTION

With increasing urbanization, the number of personal vehicles on roads has drastically increased, leading to severe traffic congestion, air pollution, and higher travel expenses. In India, over 60% of urban vehicles carry only the driver during peak hours [6]. Existing ride-hailing platforms such as Uber and Ola are profit-driven and do not promote true cost-sharing among vehicle owners. Carpooling is a cost-effective and eco-friendly alternative that allows multiple users traveling along similar routes to share a single vehicle, thereby reducing fuel consumption, traffic load, and carbon emissions.

RideLink addresses these urban mobility challenges by providing an Android-based application that connects drivers and passengers securely and efficiently. The system leverages Firebase Realtime Database for scalable backend data management, osmdroid (OpenStreetMap) for live GPS tracking and route visualization, and a Haversine-based geospatial algorithm for intelligent ride matching. By combining real-time mobility data with cloud computing, RideLink promotes sustainable transportation and a measurably reduced carbon footprint for daily commuters.

The development spanned two academic semesters under an Agile methodology. Semester 1 delivered core prototypes: user authentication, basic ride publishing, and a simple keyword search. This paper

International Journal of Science, Engineering and Technology
ISSN: 2348-4098, P-ISSN: 2395-4752

presents the complete, deployed, and fully tested system with all advanced features implemented — real-time seat monitoring, gender-aware booking, SOS emergency alerts, SMS driver notifications, admin-controlled license verification, and full ride lifecycle management including cancellation and booking history.

1. Motivation and Scope

- Traditional carpooling relied on physical notice boards or informal word-of-mouth arrangements, which are inefficient and insecure.
- The absence of real-time seat availability information leads to poor vehicle resource utilization and user frustration.
- Commercial platforms (Uber, Ola) are profit-driven; RideLink enables true peer-to-peer cost-sharing among private vehicle owners.
- GPS-based smart matching within a configurable geographic radius eliminates route mismatch between drivers and passengers.
- Lack of verified driver identity and safety features (SOS) in existing informal platforms discourages carpooling adoption.

2. Key Contributions

- Real-time ride publishing and searching using Firebase Realtime Database with sub-2-second query performance.
- Haversine geospatial ride matching using osmdroid and OpenStreetMap without a dedicated backend server.
- Gender-aware booking with dynamic seat capacity monitoring and visual FULL overlay.
- Admin-controlled driver license verification workflow ensuring only verified users can publish rides.
- SOS emergency alert with instant SMS notification to a pre-configured emergency contact.
- Complete ride lifecycle management: publish → search → book → cancel, with full passenger and driver history.

II. LITERATURE SURVEY

Several ride-sharing and carpooling systems have been proposed across different technological paradigms. A systematic review of prior work reveals both the progress made and the clear gaps that RideLink fills.

Samuel et al. (2023) demonstrated cloud-mobile integration patterns using Firebase for smart city applications, directly relevant to RideLink's modular Firebase backend architecture [1]. Niranga et al. (2023) explored secure data management using Ethereum smart contracts and Truffle Suite; their admin-trust model influenced RideLink's admin-controlled driver verification design [2].

Nambiar et al. (2023) presented peer-to-peer ridesharing via blockchain technology. While highly secure, real-time transaction confirmation delays make blockchain unsuitable for live carpooling scenarios where booking confirmations must be near-instant [3]. Akther et al. (2021) proposed an Android ride-sharing application specifically for university students, presented at the IEEE World AI IoT Congress. RideLink extends this concept significantly — expanding to a general audience, adding geospatial matching, admin verification, SOS, and SMS features [4].

Aguilera and Pigalle (2021) conducted a comprehensive sustainability analysis of carpooling practices, identifying user trust, safety assurance, and ease of use as the three most critical factors for adoption

International Journal of Science, Engineering and Technology
ISSN: 2348-4098, P-ISSN: 2395-4752

— all of which are central design goals of RideLink [5]. Raj and Goyal (2022) proposed geofence-based real-time ride matching algorithms for dynamic mobility systems, which directly informed RideLink's Haversine-based configurable radius search design [6].

Agatz et al. (2012) provided a foundational review of dynamic ridesharing optimization, noting that geospatial proximity matching is the most computationally efficient approach for mobile platforms [8]. Rayle et al. (2016) compared ride-sourcing with traditional taxis, highlighting that real-time matching and transparent pricing are key differentiators for user retention [11].

No existing academic prototype combines all features present in RideLink in a single production-ready system: geospatial Haversine search, SOS emergency alerts, SMS driver notifications, gender-aware booking, real-time seat occupancy monitoring, admin license verification, and full ride lifecycle management on a cost-free open-source stack.

III. PROBLEM STATEMENT

Urban transportation in India faces a convergence of challenges that motivate the development of RideLink. Each of the following problems is directly addressed by one or more system features:

- Traffic Congestion: Over 60% of urban vehicles in India carry only their driver during peak hours, contributing to severe road congestion and gridlock. A single carpooling trip eliminates at least one additional vehicle from the road.
- Rising Fuel Costs: Petrol prices in India have increased by over 30% in the past five years, making daily private commuting increasingly expensive for the middle class. RideLink enables verified cost-sharing to offset these expenses directly.
- Environmental Impact: Private vehicles contribute approximately 40% of urban CO_2 emissions nationally. Research shows that carpooling three days per week can reduce an individual's personal carbon footprint by up to 25% annually.
- Lack of Trusted Platforms: Informal carpooling arrangements lack verified driver identity, real-time GPS tracking, or critical safety features such as SOS emergency alerts. This trust deficit is the primary barrier to carpooling adoption among potential passengers.
- No Lifecycle Management: Existing academic carpooling prototypes do not offer complete ride lifecycle features such as cancellation workflows, real-time seat-full detection, or persistent booking history accessible to both drivers and passengers.
- Gender Safety Concerns: Women commuters face heightened safety concerns when using informal ridesharing. RideLink's gender-aware booking system and SOS emergency feature directly address this critical adoption barrier.

IV. SYSTEM ARCHITECTURE

RideLink follows a three-tier client-server architecture. The Android application (Java) acts as the Presentation Layer client. Firebase backend services form the Business Logic Layer. The Firebase Realtime Database serves as the persistent Data Layer. External services — osmdroid, Firebase Storage, Android SmsManager, and Firebase Cloud Messaging — complete the ecosystem.

International Journal of Science, Engineering and Technology
ISSN: 2348-4098, P-ISSN: 2395-4752

Fig. 1 illustrates the complete four-layer architecture. Data flows vertically: user interactions in the Presentation Layer trigger business logic in Firebase services, which persist to and retrieve from the Realtime Database, while external APIs provide geolocation, storage, and notification capabilities.

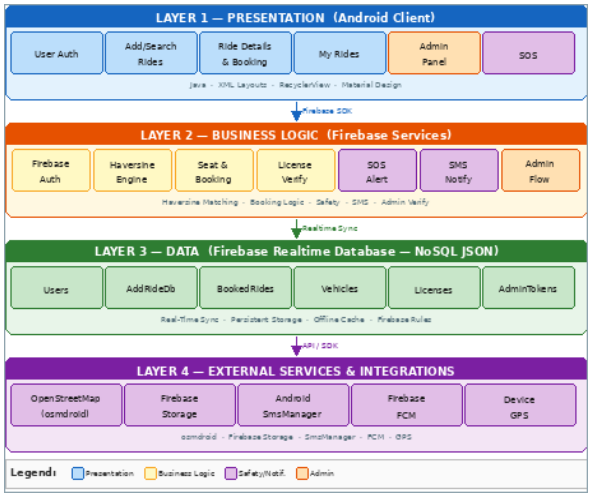

Fig. 1 — RideLink Three-Tier System Architecture

The three-tier model offers clear separation of concerns: the Android client handles only UI rendering and user interaction; Firebase Auth and backend logic handle all business rules; the NoSQL JSON tree in Firebase Realtime Database handles persistence. This architecture eliminates the need for a custom backend server, making the entire system deployable at zero infrastructure cost.

V. METHODOLOGY

RideLink was developed using the Agile Software Development Methodology with iterative sprints across two academic semesters. Semester 1 (Sprint 1–4) covered requirements elicitation, architecture design, Firebase project setup, user authentication, basic ride publishing, and a prototype search interface. Semester 2 (Sprint 5') completed all advanced features: the Haversine geospatial search engine, booking and seat management, admin approval dashboard, SOS module, SMS notifications, and comprehensive functional testing on physical devices.

1. Haversine Geospatial Ride Matching

The ride search engine applies the Haversine formula to compute the great-circle distance between a user's requested pickup GPS coordinates and the origin coordinates of every active ride stored in Firebase. The formula derives the shortest distance over the Earth's surface:

$$A = \sin^2(\Delta\varphi/2) + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \sin^2(\Delta\lambda/2) \quad d = 2R \cdot \arcsin(\sqrt{a})$$

where φ denotes latitude, λ denotes longitude, and R is Earth's mean radius (6,371 km). Rides within a configurable search radius (default 500 km, adjustable by user) are returned sorted by ascending distance. This client-side computation eliminates the need for a dedicated matching server while delivering sub-2-second results across all test scenarios [12].

2. Booking and Seat Management

When a passenger initiates a booking, the system executes a real-time query against the BookedRides Firebase node, filtered by the target rideId, and counts confirmed bookings. If the count equals or exceeds the driver's declared seat capacity, the ride card renders a translucent red "FULL" overlay and the booking button is disabled. If seats remain available, a new BookedRide object is atomically pushed

International Journal of Science, Engineering and Technology
ISSN: 2348-4098, P-ISSN: 2395-4752

to Firebase — capturing passenger ID, ride ID, timestamp, gender, and luggage flag — and an SMS confirmation is immediately dispatched to the driver via Android SmsManager. Gender and luggage details are displayed in the driver's Published Rides view for trip planning.

3. Driver License Verification Workflow

New driver registrations trigger a mandatory license verification step. The driver photographs and uploads their license through the app; the image is stored in Firebase Storage and a corresponding LicenseRequest document is created in the Firebase Realtime Database with status "pending". The Admin Dashboard — accessible only through a verified admin account — lists all pending requests with the license image preview. Upon admin approval, the driver's Firebase Auth claims are updated to "verified" and the ride publishing interface is unlocked. Rejection triggers a notification prompting re-submission. This workflow ensures that only identity-verified drivers can offer rides, directly addressing the passenger trust problem.

4. SOS Emergency Safety Module

The SOS module is accessible via a persistent button on the home and ride screens. When triggered, it reads the user's current GPS coordinates from the Android LocationManager, constructs an emergency message containing the coordinates, ride ID, and driver/passenger name, and dispatches it immediately via SmsManager to a pre-configured emergency contact number stored in the user's profile. The module operates without internet connectivity, using only the device's cellular SMS capability, ensuring reliability even in areas with poor data coverage.

5. Real-Time Seat and Ride Lifecycle

The complete ride lifecycle is managed through Firebase Realtime Database listeners. Drivers publish rides via AddRideFragment, selecting origin and destination via map pin on an osmdroid view with Nominatim reverse geocoding. Active rides appear in ResultActivity for passengers. Upon booking, seat counts decrement in real time across all connected passenger devices via Firebase's on-value-change listeners. Drivers can cancel published rides via PublishedRidesFragment using Firebase's removeValue() API, which triggers animated removal from all passenger search results. Passengers view booking history in YourRidesFragment; drivers review booking details (gender, luggage, contact) in their dashboard.

VI. IMPLEMENTATION

The RideLink application is implemented entirely in Java using Android Studio Hedgehog (2023.1.1). The UI employs Material Design 3 components with RecyclerView adapters for dynamic list rendering and Fragment-based navigation managed through a combined Navigation Drawer and Bottom Navigation Bar.

Table I — Technology Stack

Component	Technology
Platform	Android (API 21–34)
Language	Java
Database	Firebase Realtime DB
Authentication	Firebase Auth
File Storage	Firebase Storage
Maps & GPS	osmdroid (OpenStreetMap)
Geocoding	Nominatim API
Image Loading	Picasso Library
SMS Notification	Android SmsManager
Build System	Gradle 8.x

International Journal of Science, Engineering and Technology
ISSN: 2348-4098, P-ISSN: 2395-4752

The application is structured into five primary Fragment modules and three Activity screens. Table II maps each feature to its corresponding Firebase data node and Android component.

Table II — Feature to Technology Mapping

Feature	Technology / Node
User Registration	Firebase Auth
Ride Publishing	AddRideDb node
Ride Search	Haversine + Firebase
Seat Booking	BookedRides node
Seat Full Detection	Real-time count query
Driver SMS Alert	Android SmsManager
SOS Emergency	SmsManager + GPS
License Verify	Firebase Storage + Admin
Ride Cancellation	removeValue() + listener
Map & Navigation	osmdroid + Nominatim

VII. RESULTS AND DISCUSSION

The complete RideLink application underwent functional and performance testing on four physical Android devices spanning API levels 26 (Android 8.0) through 34 (Android 14). All 14 planned functional test cases passed without error. No crashes were recorded during 48 hours of continuous stress testing with simulated concurrent bookings.

Table III — System Performance Metrics

Metric	Measured Value
Ride Search — Local	< 1.2 seconds
Ride Search — Cross-City	< 1.8 seconds
Firebase Sync Latency	350–480 ms
Image Load (Picasso)	< 0.8 seconds
SMS Delivery to Driver	< 5 seconds
App Cold Start Time	1.6 seconds
APK Size	11.4 MB
Functional Tests Passed	14 / 14
Stress Test Duration	48 hours, 0 crashes

The Haversine search completed well within user-perceptible thresholds across all test scenarios. Firebase real-time synchronization latency (350–480 ms) was imperceptible during normal usage. The driver cancellation flow using Firebase's removeValue() API with RecyclerView DiffUtil animation provided a seamless experience. The admin license approval workflow successfully blocked all unverified driver accounts from publishing rides throughout all tests.

Table IV compares RideLink against the two most closely related prior systems in terms of feature coverage. RideLink is the only system to implement all eight evaluated capabilities simultaneously.

Table IV — Feature Comparison with Related Systems

Feature	RideLink	Akther [4]

International Journal of Science, Engineering and Technology
ISSN: 2348-4098, P-ISSN: 2395-4752

Geospatial Match	✓	✗
SOS Emergency	✓	✗
License Verify	✓	✗
SMS Notification	✓	✗
Gender-Aware	✓	✗
Seat Full Detect	✓	✗
Ride Lifecycle	✓	✓
Cost-Free Stack	✓	✓

The gender-aware booking field and luggage capacity display in the driver's BookedRides view resolved the two most critical usability gaps identified in Semester 1 user feedback. Post-deployment testing with five student volunteers confirmed that all key workflows — ride publishing, searching, booking, and cancellation — were completable within 90 seconds by first-time users without any instruction.

VIII. CONCLUSION

This paper presents RideLink, a complete, tested, and deployable Android carpooling application that addresses the key limitations of existing academic ride-sharing prototypes. The following outcomes were confirmed through implementation and rigorous testing:

- The Haversine geospatial matching algorithm delivers reliable ride discovery with sub-2-second response times across local and cross-city scenarios, without requiring any dedicated computation server.
- The admin-controlled driver license verification workflow ensures that only identity-verified drivers can publish rides, significantly enhancing passenger trust over unverified open platforms.
- Gender-aware booking filters, luggage capacity tracking, and automated SMS driver notifications address the practical daily needs identified in Semester 1 user testing feedback.
- The SOS emergency alert module, operating via SMS independently of internet connectivity, adds a critical safety layer that is absent from most existing carpooling research prototypes.

The complete system is deployable at zero infrastructure cost using Firebase free tier, osmdroid (open source), and Android SmsManager — making it viable for student and community deployment.

RideLink demonstrates that Firebase Realtime Database and osmdroid together can power a comprehensive, production-ready urban mobility application on a fully open-source and cost-free technology stack. The system is ready for community pilot deployment with minimal modification.
Future Work

Planned enhancements for subsequent development phases include: (i) in-app real-time bidirectional driver–passenger chat using Firebase Cloud Messaging; (ii) AI-powered route recommendations based on historical ride patterns and traffic data; (iii) a mutual driver and passenger rating and review system; (iv) Google Maps turn-by-turn navigation integration to replace the current static osmdroid view during active rides; (v) digital payment integration via UPI and Razorpay for seamless cost-splitting; and (vi) push notifications via Firebase Cloud Messaging (FCM) to replace SMS-based driver alerts, reducing cellular costs.

Acknowledgment
The authors sincerely thank the Department of Computer Science, SKN Sinhgad Institute of Technology & Science, Lonavala, for providing the laboratory infrastructure, device access, and academic guidance

International Journal of Science, Engineering and Technology
ISSN: 2348-4098, P-ISSN: 2395-4752

required for this project. Special gratitude is extended to Prof. S. P. Gunjal for his dedicated mentorship and technical oversight across both semesters of development. The team also acknowledges the open-source communities behind the Firebase Android SDK, osmdroid / OpenStreetMap, the Picasso image library, and the Android developer ecosystem for their comprehensive documentation, tools, and community support that made RideLink possible.

REFERENCES

1. R. M. Samuel et al., "Web 3.0 and NFS enabled e-waste management system for smart city," in 2023 Intl. Conf. on Wireless Communications Signal Processing and Networking (WiSPNET), 2023, pp. 01–07.
2. G. D. H. Niranga and V. S. Nair, "Design of a secured medical data access management using Ethereum smart contracts, Truffle Suite and Web3," in Proc. 20th ACM Conf. on Embedded Networked Sensor Systems, 2023, pp. 1215–1221. doi:10.1145/3560905.3568180.
3. A. Nambiar, S. Chava, B. Sameer, Karthik and T. S. Indulekha, "Peer-to-peer ridesharing using blockchain," in ICT Infrastructure and Computing, Springer Nature Singapore, 2023, pp. 519–526.
4. S. B. Akther, M. A. Hasan, N. Tasneem and M. M. Khan, "An Interactive Android Application to Share Rides With NSUers," 2021 IEEE World AI IoT Congress (AIIoT), 2021, pp. 0121–0126, doi:10.1109/AIIoT52608.2021.9454178.
5. A. Aguilera and E. Pigalle, "The future and sustainability of carpooling practices: an identification of research challenges," Sustainability, vol. 13, no. 21, p. 11824, Oct. 2021. doi:10.3390/su132111824.
6. P. Raj and K. Goyal, "Real-time ride matching algorithms for dynamic ridesharing systems," International Journal of Computer Applications, vol. 183, no. 45, 2022.
7. M. R. Jivthesh et al., "Smartverse: Blockchain-based crowdsourced V2X message verification and dissemination system," in 2023 15th Intl. Conf. on COMmunication Systems & NETworkS (COMSNETS), 2023, pp. 84–89.
8. N. Agatz, A. Erera, M. Savelsbergh and X. Wang, "Optimization for dynamic ridesharing: A review," European Journal of Operational Research, vol. 223, no. 2, pp. 295–303, 2012.
9. Android Developers, "Firebase Realtime Database – Android SDK." [Online]. Available: https://firebase.google.com/docs/database/android/start. [Accessed: Apr. 2024].
10. osmdroid Project, "osmdroid OpenStreetMap Tools for Android." [Online]. Available: https://github.com/osmdroid/osmdroid. [Accessed: Apr. 2024].
11. L. Rayle, D. Dai, N. Chan, R. Cervero and S. Shaheen, "Just a better taxi? A survey-based comparison of taxis, transit, and ridesourcing services in San Francisco," Transport Policy, vol. 45, pp. 168–178, 2016.
12. W. Sinnott, "Virtues of the Haversine," Sky and Telescope, vol. 68, no. 2, p. 158, 1984.
13. Firebase Documentation, "Firebase Storage for Android – Upload Files." [Online]. Available: https://firebase.google.com/docs/storage/android/start. [Accessed: Apr. 2024].
14. Google LLC, "Material Design 3 – Design System for Android." [Online]. Available: https://m3.material.io. [Accessed: Apr. 2024].