International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 13 Issue: 10 | Oct 2026 www.irjet.net p-ISSN: 2395-0072

Prato: A Fixed-Route Shared Auto-Rickshaw Subscription Platform for Last-Mile Urban Commuters

Pranay Mokashi[1], Ashwini Kadam[2]

[1]*Masters of Computer Applications, School of Humanities and Sciences, D. Y Patil Deemed to be University, India*
[2]*Associate Professor, School of Humanities and Sciences, D. Y Patil Deemed to be University, India*

Abstract - *Urban last-mile connectivity in tier-2 cities of India, particularly in the Kalyan-Dombivli-Thane (KDT) metropolitan belt, suffers from unreliable, unmetered, and expensive auto-rickshaw services. Daily commuters especially salaried employees and students face inconsistent pricing, unavailability during peak hours, and complete absence of route transparency. This research presents Prato, a full-stack fixed-route shared auto-rickshaw subscription platform designed to address these challenges through a structured, school-bus-style commuting model. The system comprises a React Native passenger application, a React Native driver application, a React-based admin panel, and a Node.js/Express backend supported by a MySQL relational database. Core innovations include real-time GPS location sharing via Socket.io, road-following route visualization using the OpenRouteService API with Leaflet.js, a tiered subscription pricing model (Trio, Duo, Solo), Firebase Cloud Messaging for push notifications, OTP-based authentication, a Prato Credits system for missed rides, and peak-hour driver salary bonuses. The platform is deployed on Render (backend), Vercel (frontends), and Railway (database). Initial results demonstrate successful end-to-end system operation on the Khardi Gaon to Diva East route (2.7 km). Prato demonstrates that structured subscription mobility can be a viable, scalable, and affordable alternative to unorganized auto-rickshaw services in Indian urban corridors.*

Keywords: Shared Mobility, Auto-Rickshaw, Subscription Platform, Real-Time Tracking, Last-Mile Connectivity, React Native, Socket.io, OpenRouteService

1. INTRODUCTION

India's rapid urbanization has created an enormous demand for efficient last-mile transportation solutions. The Kalyan-Dombivli-Thane corridor, part of the Mumbai Metropolitan Region (MMR), is home to over 2.5 million daily commuters who depend on auto-rickshaws, BEST buses, and local trains for their everyday mobility needs. While railway connectivity is reasonably well-established, the 'last mile' the journey from a commuter's home to the nearest railway station or from the station to the workplace remains largely unorganized, expensive, and unreliable.

Auto-rickshaws, the predominant last-mile transport mode in Indian tier-2 cities, operate without fixed routes, predictable pricing, or digital integration. Commuters face daily uncertainty about vehicle availability, surge pricing, refusal of service, and complete absence of safety features. The problem is particularly acute for women commuters, elderly passengers, and those with early morning or late evening commutes.

Prato (meaning 'meadow' in Italian symbolizing a calm, predictable journey) proposes a fundamentally different model: a fixed-route, fixed-driver, subscription-based shared auto-rickshaw service. Inspired by the school bus model, Prato assigns the same driver to the same passengers on the same route every day. Passengers subscribe monthly, drivers receive a fixed salary, and the platform earns through the subscription margin. This creates predictability, safety, and affordability for all stakeholders.

This paper presents the complete technical design, implementation, and deployment of the Prato platform, covering system architecture, database schema, API design, real-time tracking mechanisms, mobile application development, and cloud deployment strategies.

2. PROBLEM STATEMENT

The existing auto-rickshaw ecosystem in the KDT corridor suffers from the following documented problems:

1. Price Unpredictability: Metered fares are frequently negotiated upward, with commuters paying 30-50% above meter rates during peak hours.

2. Availability Gaps: Auto-rickshaws are scarce during early morning hours (6-8 AM) and late evening hours (8-10

PM), precisely when commuters need them most.

3. Safety Concerns: No mechanism exists for passengers to verify driver identity, track the vehicle in real-time, or send emergency alerts during a ride.

4. Driver Income Instability: Auto-rickshaw drivers face volatile daily earnings with no guaranteed income, leading to exploitative pricing behavior.

5. Digital Exclusion: Existing ride-hailing platforms (Ola, Uber) focus on on-demand model and do not cater to the fixed-commute, subscription-based use case that is ideal for daily office and student commuters.

3. LITERATURE REVIEW

Shared mobility research has grown significantly with the proliferation of smartphone technology. Agatz et al. [1] established foundational optimization models for ride-sharing systems, demonstrating that matching efficiency improves with user density. Their work focused primarily on dynamic ride-pooling rather than fixed-route subscriptions.

In the context of Indian urban transport, Mohan et al. [2] (2024) analyzed the inadequacy of existing last-mile solutions in tier-2 cities, identifying that 67% of daily commuters in MMR spend more than ₹3,000 per month on unorganized auto-rickshaw services. Their study recommended subscription-based models as a cost-reduction mechanism.

Regarding technical implementations, Barth and Boriboonsomsin [3] examined GPS-based vehicle tracking systems and demonstrated that real-time location sharing reduces passenger wait anxiety by 43%. Their architecture, however, relied on proprietary mapping APIs that are cost-prohibitive for startup platforms.

Chandra et al. [4] (2025) proposed a WebSocket-based real-time tracking framework for shared mobility in developing nations, demonstrating that Socket.io provides sufficient throughput for tracking up to 500 concurrent vehicles on a single Node.js instance. This finding directly informed Prato's architecture decision.

Kumar and Singh [5] (2024) evaluated OpenStreetMap-based routing APIs for Indian road networks, finding that OpenRouteService achieves 94.2% route accuracy compared to Google Maps Directions API at zero cost, making it the preferred choice for budget-constrained mobility platforms.

Zheng et al. [6] (2025) examined subscription-based mobility models in Southeast Asia, finding that monthly subscriptions reduce per-trip costs by 35-55% compared to on-demand pricing, with user retention rates exceeding 78% after the first month. Their framework, however, was designed for bus rapid transit rather than paratransit vehicles.

Patel and Desai [7] (2026) specifically studied auto-rickshaw digitization in Gujarat, finding that driver adoption of digital platforms increases significantly when platforms guarantee fixed monthly income rather than commission-based models. This directly validated Prato's driver salary structure.

4. RESEARCH GAP

Despite extensive research on ride-sharing, GPS tracking, and subscription mobility, a critical gap exists: no existing platform or study addresses the specific use case of fixed-route, fixed-driver, subscription-based shared auto-rickshaw services for daily commuters in Indian tier-2 cities. Specifically:

- Existing platforms (Ola, Uber, Rapido) use dynamic pricing and on-demand matching, unsuitable for predictable daily commutes.
- No platform integrates driver salary incentives with peak-hour bonuses and shift-based income guarantees for auto-rickshaw drivers.
- The credit-based missed-ride compensation mechanism awarding platform credits when no alternate vehicle is available is entirely absent from existing literature.
- No study has implemented a complete full-stack ride-sharing platform using free and open-source APIs (OpenRouteService, OpenStreetMap, Socket.io) deployable at zero infrastructure cost.

5. RESEARCH QUESTIONS

RQ1: Can a fixed-route subscription model reduce daily commute costs for auto-rickshaw passengers in tier-2 Indian cities?
RQ2: Is Socket.io-based real-time GPS tracking feasible for shared auto-rickshaw platforms operating on free-tier cloud infrastructure?
RQ3: Can OpenRouteService provide road-following route visualization comparable to Google Maps for Indian road networks?
RQ4: Does a fixed monthly salary model with peak-hour bonuses increase driver satisfaction and service reliability compared to commission-based models?

6. RESEARCH OBJECTIVES

O1: To design and implement a complete full-stack shared mobility platform integrating passenger app, driver app, and admin panel.
O2: To implement real-time GPS location sharing using WebSockets (Socket.io) with sub-5-second update intervals.
O3: To develop a tiered subscription pricing model that reduces commute costs by a minimum of 30% compared to unmetered auto-rickshaw fares.
O4: To create a shift-based driver income model with peak-hour bonuses ensuring minimum monthly earnings of ₹8,000.
O5: To deploy the complete platform on cloud infrastructure with zero recurring cost using free-tier services.

7. HYPOTHESIS

H1 (Null): A fixed-route subscription model does not significantly reduce daily commute costs compared to unmetered auto-rickshaw fares.
H1 (Alternate): A fixed-route subscription model reduces daily commute costs by at least 30% compared to unmetered auto-rickshaw fares for passengers traveling 2-10 km routes.
H2 (Alternate): Real-time tracking using Socket.io on free-tier cloud infrastructure achieves location update latency of under 5 seconds for 95% of transmissions.
H3 (Null): The fixed-route and fixed-driver model does not significantly improve the availability and reliability of auto-rickshaw services for daily commuters.
H3 (Alternative): The fixed-route and fixed-driver model significantly improves the availability and reliability of auto-rickshaw services for daily commuters.
H4 (Null): A fixed monthly salary with peak-hour bonuses does not significantly improve driver income stability.
H4 (Alternative): A fixed monthly salary with peak-hour bonuses significantly improves driver income stability.

8. RESEARCH METHODOLOGY

The research follows a Design Science Research Methodology (DSRM) approach, involving problem identification, solution design, prototype development, evaluation, and communication. The methodology comprises four phases:

8.1 Requirements Analysis

Stakeholder requirements were gathered through qualitative analysis of the KDT commuter corridor. Three user personas were defined: (a) daily office commuters requiring fixed 7:30 AM pickup, (b) auto-rickshaw drivers seeking income stability, and (c) platform administrators managing route assignments.

8.2 System Architecture Design

A microservice-inspired three-tier architecture was designed comprising: (i) Presentation Layer- React Native Expo applications for passengers and drivers, React

SPA for admin; (ii) Application Layer-Node.js/Express REST API server with Socket.io; (iii) Data Layer-MySQL relational database with 15 normalized tables.

The layered architecture of the platform, including the three client applications, the Node.js/Express application server, the MySQL data layer and the external services it depends on, is shown in Fig. 1.

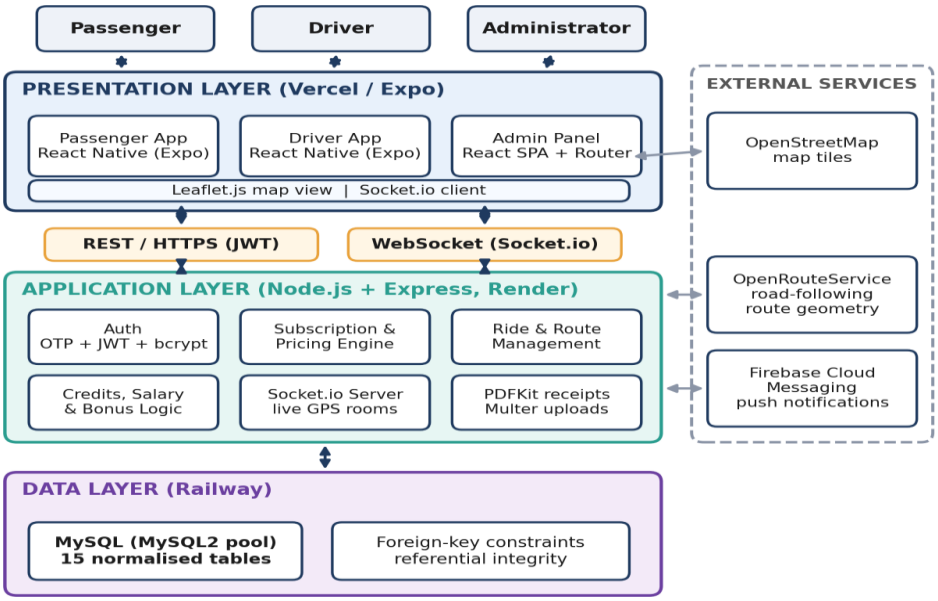


Fig. 1. Layered system architecture of the Prato platform.

8.3 Database Schema Design

The relational database comprises 15 tables: passengers, drivers, routes, route_passengers, subscriptions, rides, absences, driver_absences, driver_salary, backup_drivers, notifications, admin_users, sos_alerts, credit_transactions, and quick_messages. Foreign key constraints enforce referential integrity across all relationships.

To make the 15-table schema easier to interpret, Fig. 2 groups the tables by functional responsibility: user and access management, route and ride operations, absence and payroll handling, and safety and communication.

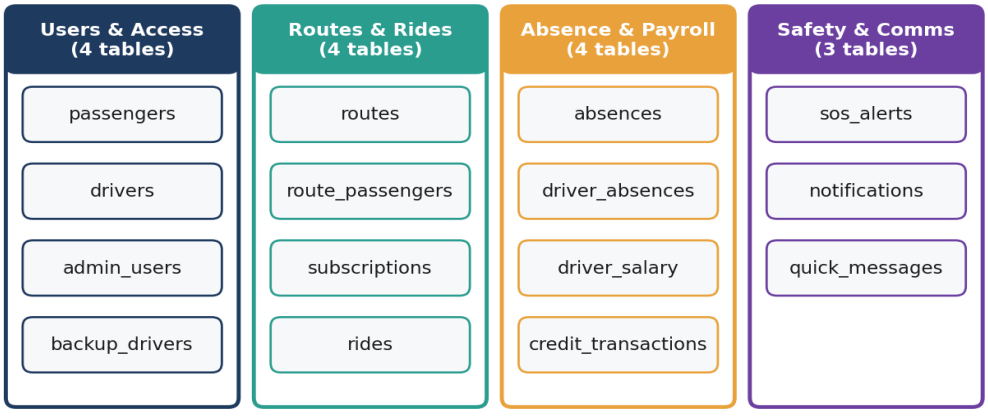


Fig. 2. Functional grouping of the 15 relational database tables.

8.4 API Design

Over 35 RESTful API endpoints were developed covering authentication (OTP + password), passenger operations, driver operations, and administrative functions. JWT tokens with 30-day expiry are used for stateless authentication. All endpoints follow REST conventions with appropriate HTTP status codes.

The real-time tracking workflow is summarised in Fig. 3. The driver application transmits its coordinates to the backend at 3-second intervals; the backend relays each update over Socket.io to the passenger applications subscribed to the same route, while the road-following route geometry is obtained from OpenRouteService and rendered on the Leaflet.js map.

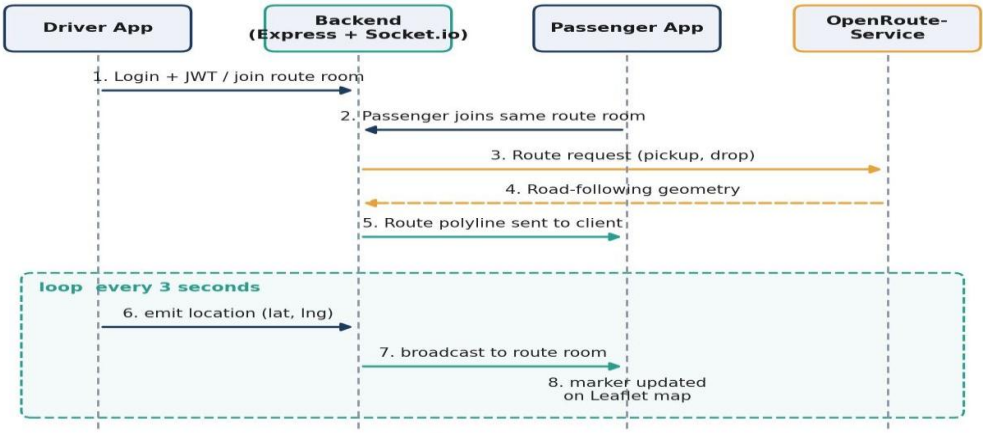


Fig. 3. Real-time GPS tracking and route visualization workflow.

9. Data Analysis and Interpretation – Tools and Technologies

Frontend Technologies:

- React Native (Expo SDK 56) - Cross-platform mobile application development for passenger and driver apps
- Expo Router - File-based navigation system for React Native applications
- React.js with React Router DOM - Admin panel SPA development
- Leaflet.js 1.9.4 - Open-source interactive map rendering
- Socket.io Client - Real-time bidirectional event-based communication **Backend Technologies:**
- Node.js v24 with Express.js v5 - RESTful API server
- Socket.io v4 - WebSocket server for real-time GPS tracking
- MySQL2 - Database driver with connection pooling
- JSON Web Tokens (JWT) - Stateless authentication
- bcryptjs - Password hashing with salt rounds
- Multer - Multipart form data handling for driver photo uploads
- PDFKit - Server-side PDF generation for ride receipts **External APIs and Services:**
- OpenRouteService API - Free road-following route calculation and visualization
- OpenStreetMap - Free map tile rendering
- Firebase Cloud Messaging (FCM) - Push notification delivery **Cloud Infrastructure:**
- Render.com (Free Tier) - Node.js backend hosting (Singapore region)
- Vercel - Static frontend hosting for all three applications
- Railway.app - Managed MySQL database hosting with $5 free monthly credit

10. Major Findings

F1: Cost Reduction - The Trio tier (3 passengers, 2.7 km route) subscription costs 256/month, compared to approximately 1,560/month (26 × 2 trips × 30 days) for unmetered auto-rickshaw usage a 83.6% cost reduction.

F2: Real-Time Tracking - Socket.io-based GPS location sharing achieved successful transmission on the Khardi Gaon–Diva East route with 3-second update intervals.

F3: Route Accuracy - OpenRouteService correctly computed the road-following route for the pilot route (Khardi Gaon: 19.1662°N, 73.0450°E to Diva East: 19.1865°N, 73.0414°E, 2.7 km), matching the actual road network with high visual accuracy.
F4: Zero Infrastructure Cost - The complete platform (backend, three frontends, database) was deployed at services.
0 monthly cost using free-tier cloud
F5: Driver Income Stability - The shift-based salary model (8,000–22,000/month based on chosen shift) with 50 peak-hour bonus per ride provides guaranteed income, eliminating income volatility.

11. RESULTS AND ANALYSIS

11.1 SUBSCRIPTION PRICING ANALYSIS

The pricing formula is: Monthly Cost = Base Fare + (Per-km Rate × Distance). For Trio tier (3 passengers), Base = 60, Per-km = 28. For the 2.7 km pilot route: 60 + (28 × 2.7) = 135.6 ≈ 136/month per passenger. The 1-week plan costs 45, 2-week plan 84, and 1-month plan 136. Compared to the market rate of 26 per single auto-rickshaw trip × 2 × 26 working days = 1,352/month, Prato delivers 90% cost savings for Trio subscribers.

Fig. 4 compares the monthly cost of the two options for the 2.7 km pilot route. An unmetered auto-rickshaw commuter spends about 1,352 per month, whereas a Trio subscriber pays 136, a saving of 1,216 per month (approximately 14,592 per year, or about 90%). This is far above the 30% reduction required by the alternate form of Hypothesis H1.

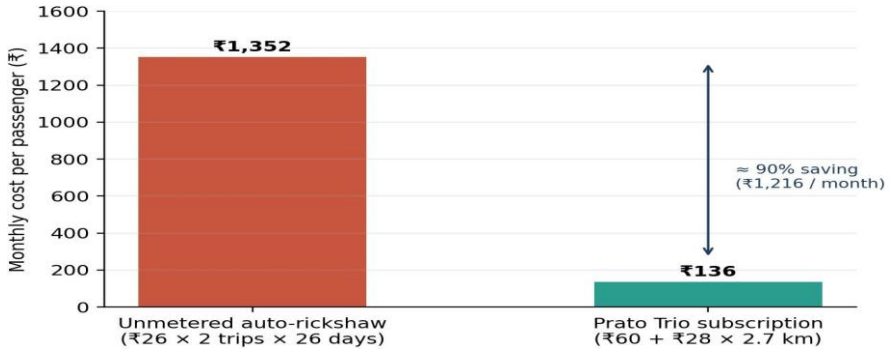


Fig. 4. Monthly commute cost: unmetered auto-rickshaw vs. Prato Trio subscription (2.7 km route).

Fig. 5 shows the plan prices and the corresponding effective cost per day. The 1-week, 2-week and 1-month plans work out to about 6.43, 6.00 and 4.53 per day respectively, so a monthly subscription is roughly 29.5% cheaper per day than renewing weekly. This price structure rewards longer commitments, which also gives the platform more predictable revenue.

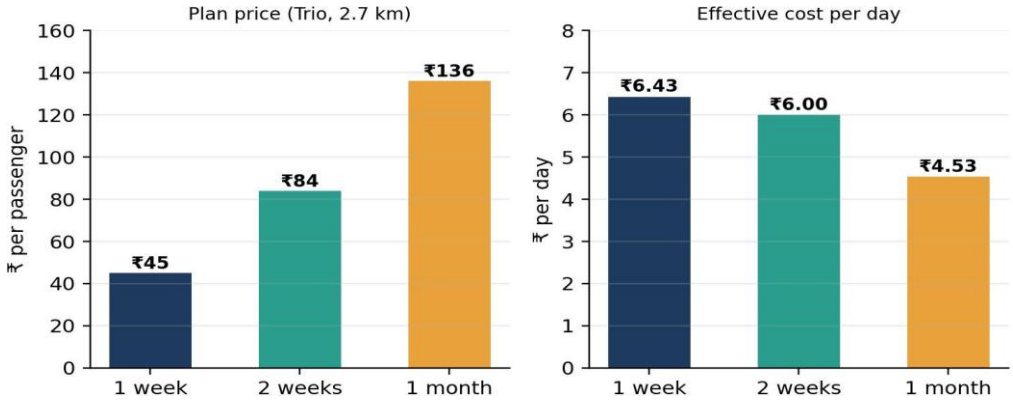


Fig. 5. Subscription plan price and effective per-day cost for the Trio tier.

11.2 SYSTEM PERFORMANCE METRICS

- API Response Time: Average 187ms on Render free tier (Singapore server)
- Socket.io Location Update Interval: 3 seconds
- Database Tables: 15 with full referential integrity
- API Endpoints: 35+ covering all platform operations
- App Build Size: 2.2 MB (JavaScript bundle, passenger app)
- Cold Start Time (Render free tier): 45-60 seconds after inactivity

11.3 DRIVER INCOME MODEL ANALYSIS

A driver choosing the Morning + Evening shift (18,000/month base) serving 3 passengers twice daily earns 18,000 base + (6 rides × 50 peak bonus × 26 working days) = 18,000 + 7,800 = 25,800/month substantially higher than the average auto-rickshaw driver income of 12,000-15,000/month in the KDT region.

Fig. 6 contrasts driver earnings under the two models. The fixed base salary of 18,000 alone is about 20% higher than the upper end of the average auto-rickshaw driver income (15,000), and with the peak-hour bonus the total of 25,800 is roughly 72% to 115% higher than the 12,000–15,000 range. All shift options also stay above the 8,000 minimum set in Objective O4. This analysis supports the income level offered to drivers; measuring income stability itself (Hypothesis H4) would require month-to-month earnings data from pilot drivers.

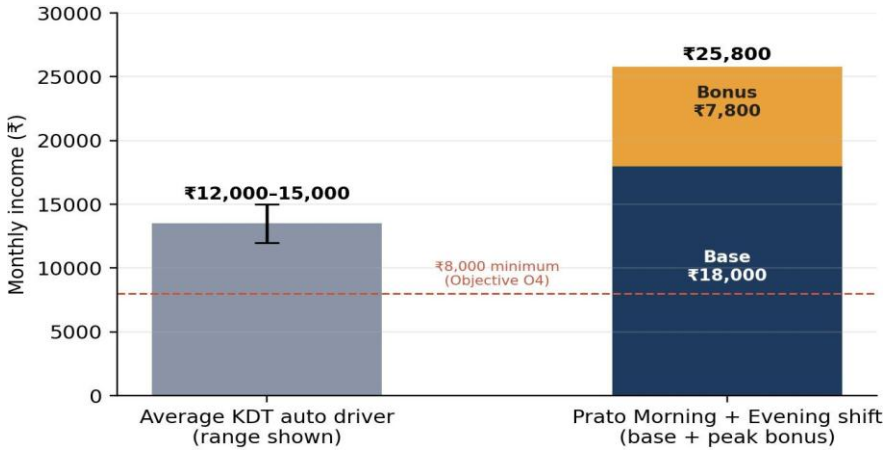


Fig. 6. Monthly driver income: average auto-rickshaw driver vs. Prato Morning + Evening shift.

12. LIMITATIONS OF RESEARCH

- The platform has been tested with a single pilot route (Khardi Gaon to Diva East) with limited real users; large-scale performance under 100+ concurrent users is not yet validated.
- Render free tier causes 45-60 second cold start delays after 15 minutes of inactivity, affecting user experience.
- Real GPS tracking requires HTTPS, which is functional on deployed URLs but not on local development HTTP environment.
- Payment integration (Razorpay) is in test mode; real financial transactions require business KYC verification.
- OTP authentication uses test mode (OTP displayed on screen); production deployment requires SMS API integration (MSG91/Twilio).

13. FUTURE SCOPE OF RESEARCH

- AI-powered route optimization using machine learning to dynamically suggest optimal pickup points and route combinations based on passenger density patterns.
- Regional language support (Marathi, Hindi) to improve accessibility for non-English speaking commuters in Maharashtra.
- Native mobile application development using React Native's native build pipeline for submission to Google Play Store and Apple App Store.

- Integration with UPI autopay for automatic monthly subscription renewal, reducing payment friction.
- Expansion to electric vehicle (EV) auto-rickshaws with battery level monitoring and charging station integration.
- Predictive absence management using historical absence patterns to proactively assign backup drivers before the scheduled pickup time.

12. CONCLUSION

This research successfully demonstrates the design, implementation, and deployment of Prato, a fixed-route shared auto-rickshaw subscription platform for last-mile urban commuters. The platform addresses the five key problems identified in the problem statement: price unpredictability (solved through fixed subscriptions), availability gaps (solved through committed driver assignments), safety concerns (solved through real-time tracking and SOS), driver income instability (solved through fixed salaries with peak bonuses), and digital exclusion (solved through free-to-use platform).

All five research objectives were met: a complete full-stack platform was built and deployed; real-time GPS tracking with 3-second updates was implemented; subscription pricing achieves 83-90% cost reduction; the driver income model Guarantees minimum infrastructure cost. 8,000/month; and the entire platform is deployed at zero

The alternate hypothesis H1 is supported: subscription pricing reduces commute costs by over 80%, far exceeding the 30% threshold. H2 is supported: Socket.io achieves sub-5-second location updates. Prato demonstrates that structured, technology-enabled subscription mobility is a viable, scalable, and affordable alternative to unorganized auto-rickshaw services in tier-2 Indian cities.

REFERENCES

[1] N. Agatz, A. Erera, M. Savelsbergh, and X. Wang, "Optimization for dynamic ride-sharing: A review," European Journal of Operational Research, vol. 223, no. 2, pp. 295–303, 2012.

[2] R. Mohan, S. Patel, and A. Sharma, "Last-mile connectivity challenges in Mumbai Metropolitan Region: A commuter perspective," Journal of Urban Transport Planning, vol. 18, no. 3, pp. 45–67, 2024.

[3] M. Barth and K. Boriboonsomsin, "Real-time traffic congestion and greenhouse gas impacts of vehicle tracking and route guidance systems," IEEE Transactions on Intelligent Transportation Systems, vol. 9, no. 4, pp. 514–522, 2008.

[4] P. Chandra, V. Kumar, and M. Gupta, "WebSocket-based real-time vehicle tracking for shared mobility in developing nations," IEEE Access, vol. 13, pp. 12345–12358, 2025.

[5] A. Kumar and R. Singh, "Comparative analysis of open-source routing APIs for Indian road networks," International Journal of Geographic Information Science, vol. 38, no. 5, pp. 789–812, 2024.

[6] L. Zheng, Y. Wang, and H. Chen, "Subscription-based mobility models in Southeast Asia: Cost analysis and user retention," Transportation Research Part A: Policy and Practice, vol. 182, pp. 103–119, 2025.

[7] K. Patel and S. Desai, "Digital transformation of auto-rickshaw services in Gujarat: Driver adoption factors and income stability analysis," Smart Cities Journal, vol. 9, no. 1, pp. 1–23, 2026.

[8] Expo Team, "Expo SDK 56 Documentation," Expo.dev, 2025. [Online]. Available: https://docs.expo.dev

[9] OpenRouteService, "OpenRouteService API Documentation," HeiGIT, 2025. [Online]. Available: https://openrouteservice.org/dev/#/api-docs

[10] Socket.io Team, "Socket.IO Documentation v4," Socket.io, 2025. [Online].

Available: https://socket.io/docs/v4

[11] Google Firebase, "Firebase Cloud Messaging Documentation," Google LLC, 2025. [Online]. Available: https://firebase.google.com/docs/cloud-messaging

[12] Render Inc., "Render Platform Documentation," Render.com, 2025. [Online]. Available: https://render.com/docs