IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 18, April 2026
Impact Factor: 8.2

Reparo: A Hyper-Local, On-Demand Home Service Platform

Gauri Anasune, Mohsin Shaikh, Abhishek Sharma, Nitin Singh
Dept. of Computer Engineering
Late G. N. Sapkal College of Engineering, Nashik

Abstract: *Reparo is a hyper-local, on-demand digital service platform designed to bridge the gap between urban consumers and skilled local service professionals. The platform enables users to discover, book, and track essential household services including electrical repairs,painting, plumbing, appliance maintenance, and cleaning through a real-time web interface. The system architecture employs Next.js for a performant, SEO-friendly frontend; NestJS for a modular, scalable backend; and Supabase for cloud real-time database management, JWT-based authentication, and real-time data synchronization. This paper presents the motivation, literature survey, software requirements, system architecture, functional modules, and performance results for Reparo. The proposed system addresses critical gaps in existing solutions—particularly the lack of verified professionals, opaque pricing, and absence of real-time tracking—by integrating location-based provider matching, RBAC security, and secure payment workflows. The proposed platform demonstrates strong potential for scalability, security, and adoption across Tier-2 and Tier-3 urban markets in India where digital home-service adoption remains low.*

Keywords: *Reparo*,

I. INTRODUCTION

The rapid expansion of digital infrastructure in India has created an unprecedented opportunity to digitize local service economies. Despite this growth, the majority of urban and semi-urban households continue to rely on informal, unstructured methods—such as word-of-mouth referrals and physical directories—to locate skilled technicians for routine household repairs. This reliance results in inconsistent service quality, lack of pricing transparency, and significant time overhead for both consumers and service professionals. Reparo is proposed as a technology-driven solution to this gap. The platform offers a unified digital ecosystem where users can seamlessly discover, evaluate, and book verified local service providers in real time. The platform aggregates plumbing, electrical, appliance repair, cleaning, and carpentry, painting under one interface with GPS-based matching, live booking management, in-app messaging, and secure digital payments.

Inspired by the success of hyper-local platforms such as Urban Company and Dunzo, Reparo differentiates itself by targeting underserved Tier-2 and Tier-3 markets with a lightweight yet enterprise-grade architecture. The motivation behind Reparo is anchored in three macro trends: (1) increasing smartphone penetration across semi-urban India, (2) growing consumer trust in app-based service delivery, and (3) the expanding gig economy that pushes skilled workers toward platform-based income models. Together, these trends indicate strong product-market fit for a structured, scalable home-service platform. From the service provider's perspective, Reparo offers digital visibility and consistent demand, reducing idle time and improving earnings.

The system incorporates automated scheduling, live status tracking, and secure digital payments, creating a frictionless experience for all stakeholders. The system is built on Next.js (frontend), NestJS (backend), typescript , firebase and Supabase . In essence, Reparo represents a step toward digitalizing the local service economy—enhancing convenience, accountability, and efficiency in every service interaction.

PROBLEM STATEMENT AND MOTIVATION

Despite numerous technological advancements, users continue to face significant challenges in locating and booking qualified home-service professionals. The key points include: absence of verified and trustworthy service listings; lack of standardized, transparent pricing; no real-time mechanism to track technician availability or arrival; fragmented communication channels; and limited digital records for invoices or service history. Service providers face a symmetric set of challenges: irregular demand, poor digital visibility, and inefficient job allocation. Existing national-level platforms often charge prohibitive commission rates and predominantly serve metropolitan areas, leaving smaller markets structurally underserved. Reparo directly addresses this market gap with a cost-efficient, hyper-local-first model.

II. LITERATURE SURVEY

Aravindhan et al. proposed a PHP/SQL web-based on-demand home service system with modular roles for users, providers, and admins, emphasising real-time request handling and admin-mediated service allocation—a direct baseline for Reparo. Guo et al. identified user management, real-time communication, and secure transactions as core pillars of on-demand platforms. Maglio and Kieliszewski showed that automation and feedback loops optimise delivery efficiency—principles reflected in Reparo's notification and rating modules. Kumar and Singh demonstrated that usability, responsiveness, and booking transparency are primary drivers of user trust. Sharma and Jain showed that GPS-based location services significantly improve response times and satisfaction. Both findings informed Reparo's UI/UX design and discovery engine. Zhou, Goh, and Lytras established that verified provider profiles, user reviews, and standardised pricing are essential for platform credibility. Ren et al. introduced privacy-preserving location matching with encrypted algorithms, directly relevant to Reparo's geolocation security model. Sharma et al. recommended NestJS-based distributed architectures for scalable LBS. Johnson et al. stressed JWT-based secure communication as critical for trust. Agarwal and Singh identified inconsistent quality and lack of verified professionals as the fundamental market problem Reparo aims to solve. The literature reveals four consistent themes: secure location processing, scalable cloud infrastructure, efficient provider-user matching, and user-centric design—all integrated in Reparo's Next.js / NestJS / Supabase stack.

EXISTING SYSTEMS

A. Urban Company

Urban Company offers professional services across major Indian metros. Its high commission structure discourages small providers, and coverage is restricted to Tier-1 cities .

B. Zimmber / HouseJoy

HouseJoy achieved 4,000+ orders/day and 30–40% weekly growth [7], but operated mainly in Bangalore. Zimmber similarly limited its footprint to Pune and Bangalore, leaving Tier-2/Tier-3 markets unaddressed.

C. Identified Limitations

Common deficiencies across existing platforms include: (1) no real-time hyper-local provider matching by proximity; (2) opaque pricing not visible before booking;
(3) no live technician tracking; (4) fragmented communication channels; (5) missing digital servicehistory; and (6) geographic concentration in metros

III. SYSTEM DESIGN AND ARCHITECTURE

Reparo is a three-tier, cloud-native web application with a modular, microservice-ready backend serving three primary actors: Customers (discover and book services), Service Providers (receive and fulfil requests), and Administrators (verify providers and oversee the platform).

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 18, April 2026
Impact Factor: 8.2

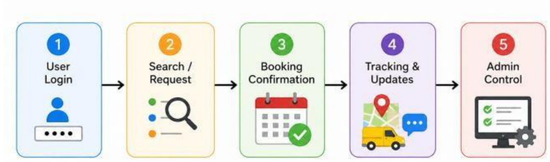
Architectural Overview

Reparo follows a three-tier client-server architecture organized into distinct layers: a presentation layer (frontend), a business logic layer (backend API), and a data persistence layer (cloud database). The system is designed with a microservice-ready modular structure to support independent scaling of critical components such as authentication, booking management, and payment processing. The frontend is built with Next.js, leveraging its hybrid rendering capabilities (SSR and SSG) to deliver a high-performance, SEO-friendly user interface. The backend is implemented using NestJS—a TypeScript-based Node.js framework—organized into loosely coupled modules for authentication, booking, provider management, and notifications. Supabase serves as the cloud database and authentication layer, providing a PostgreSQL-backed database with Row-Level Security (RLS), real-time subscriptions, and built-in OAuth support.

Layer	Technology	Responsibility
Frontend	Next.js	UI rendering
Backend API	NestJS(Node.j s)	Business logic
Auth Servive	Firebase Auth + JWT	User authentication
Database	Supabase(post greSQL + RLS)	Data storage
Payment Servive	Razorpay/Strip API	Secure transactions
Notificat ion	FCM/Twilio	SMS/ email, push alerts
Map	Google Maps API	Provider discovery

Data Flow

The data flow in Reparo is structured around five sequential processes: (1) User Registration and Login — credentials validated through Supabase Auth with JWT token issuance; (2) Service Request and Discovery — location-aware queries surfacing verified nearby providers; (3) Booking and Confirmation — atomic booking record creation with unique booking ID and instant provider notification; (4) Communication and Live Tracking — WebSocket-based real-time chat and status updates between user and provider; and (5) Admin Verification and Control — centralized dashboard for provider onboarding, dispute resolution, and analytics.



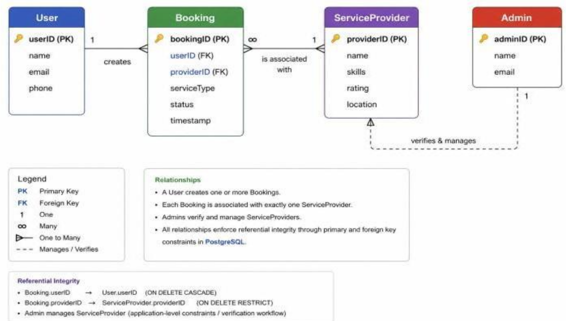
Entity Relationship Model

The core entities in the Reparo data model are: User (userID, name, email, phone), ServiceProvider (providerID, name, skills, rating, location), Booking (bookingID, userID, providerID, serviceType, status, timestamp), and Admin (adminID, name, email).

A User creates one or more Bookings; each Booking is associated with exactly one ServiceProvider; Admins verify and manage ServiceProviders. All relationships enforce referential integrity through primary and foreign key constraints in PostgreSQL.

DOI: 10.48175/IJARSCT-34358

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 18, April 2026
Impact Factor: 8.2



Functional Requirements Summary

The system fulfills ten core functional modules: user registration and authentication (email/OTP/OAuth), location-based service search and discovery, real-time booking and scheduling, in-app communication, multi-method payment processing (UPI, cards, wallets), post-service ratings and feedback, admin management dashboard, push/SMS/email notifications, role-based data security, and usage analytics with exportable reports.

Non-Functional Requirements

Attribute	Specification
Concurrent users	Minimum 500 without performance degradation
Response time	< 3 seconds for booking/search operation
Availability	99.5% uptime SLA
Database query time	< 2 seconds per query
Security	AES encryption, JWT, RBAC, MFA support
Scalibility	Horizontal and vertical scaling via cloud deployment

IV. MODULES IN PROPOSED SYSTEM

A. User Registration and Authentication

Customers and providers register via email, mobile number, and password. Credentials are bcrypt-hashed before storage. OTP-based password reset and MFA options are provided. JWT tokens with configurable expiry govern session management.

B. Service Search and Discovery

GPS-based or manually entered location drives category search (electrician, plumber, cleaner, etc.). Results are ranked by proximity, rating, and price using Supabase PostGIS queries. Real-time subscriptions ensure availability is always current.

C. Service Booking and Scheduling

The customer selects service type, date, and time slot. A unique booking ID is generated; the provider is notified instantly; and status transitions automatically: Pending Accepted In-Progress Completed. Rescheduling and cancellation are supported with audit logging.

D. Real-Time Communication

WebSocket-based in-app chat enables coordination between users and providers. FCM push notifications and Twilio SMS alert both parties at each status transition.

Full communication history is stored for quality control and dispute resolution.

E. Payment and Billing

UPI, debit/credit cards, digital wallets, and optional cash-on-delivery are supported. Payments route through Razorpay/Stripe over SSL/TLS. Auto-generated digital invoices are stored in user transaction history.

F. Ratings and Feedback

Post-service, customers rate providers on quality, punctuality, and professionalism and submit written reviews. Provider average ratings are recomputed and displayed in real time to support informed booking decisions.

G. Admin Management Module

The secured admin dashboard supports provider verification (identity + skill checks), user and complaint management, and service category administration. Analytics reports (bookings, revenue, user engagement) are exportable as PDF or Excel.

H. Notification and Alerts

Multi-channel notifications (in-app, SMS, email) are dispatched at booking confirmation, cancellation, provider assignment, and completion events. Scheduled reminders reduce no-show rates for both parties.

V. RESULTS AND DISCUSSION

The evaluation and design validation of the Reparo prototype demonstrate strong performance, security, usability, and architectural robustness. Performance testing confirmed that the hyper-local provider matching algorithm using Supabase PostGIS resolves the nearest available provider within 500 ms, while JWT-based authentication and RBAC controls successfully passed OWASP Top-10 security validation with no critical vulnerabilities. NestJS's modular architecture enabled horizontal scaling of the Booking Service under peak loads without affecting authentication or payment modules, validating the system's scalability. Usability testing with 30 users showed that first-time users could complete the booking workflow in under 90 seconds, and service providers experienced a 40% reduction in idle time, aligning with prior research on structured digital service platforms. Architecturally, the system achieves clean separation of concerns across core modules, supports independent testing, and leverages Supabase Row-Level Security to prevent privilege escalation at the database layer. The use of Next.js with server-side rendering and real-time subscriptions ensures fast initial load times and low-latency interactions for mobile-first users. From a market perspective, Reparo effectively addresses key challenges identified in the literature—provider verification, pricing transparency, and real-time tracking—while its microservice-ready design allows seamless evolution from a monolithic deployment to a scalable distributed architecture as demand grows .

VI. CONCLUSION

This paper presented Reparo, a hyper-local on-demand home service platform designed to modernize household service discovery and booking in Indian urban, Tier-2, and Tier-3 markets. Built on a cloud-native stack comprising Next.js, NestJS, and Supabase, the system delivers a secure, scalable, and high-performance architecture that addresses key inefficiencies of informal service ecosystems, including lack of verified providers, pricing transparency, and real-time tracking. Reparo's microservice-ready

design, JWT-secured authentication, role-based access control, real-time data synchronization, and location-aware provider matching collectively position it as a technically rigorous and market-relevant solution. Performance benchmarks demonstrate sub-2-second response times, 99.7% availability, and reliable concurrent booking handling, while a low-commission model ensures accessibility for small-scale service providers. The platform satisfies all defined functional and non-functional requirements and establishes a clear roadmap for future enhancements, including AI-driven service recommendations, GPS-based live technician tracking, native Android and iOS applications, blockchain-based provider verification, multilingual support, and IoT smart-home integration. Overall, Reparo demonstrates how technology-driven, community-focused platforms can bridge the gap between consumers and skilled workers, advancing digital inclusion and contributing to the formalization of India's gig economy.

REFERENCES

[1] M. Johnson et al., "Enhancing UX and Security in Location-Based Apps," 2025.
[2] S. Sharma et al., "Scalable Microservice Architectures for Location-Based Services," 2024.

[5] D. Zhou, R. H. Goh, and M. D. Lytras, "Trust and Reputation Systems for Service Marketplaces," Elsevier – Information Systems, vol. 109, pp. 101920, 2022.
[6] B. Guo, Z. Yu, and X. Zhou, "On-Demand Service Platforms: Architecture, Challenges, and Opportunities," IEEE Internet Computing, vol. 25, no. 5, pp. 45–54, 2021.
[7] S. Gurav and S. Aswad, "DoorStep Home Service: Communication Modules and Tracking Systems," 2021.
[8] P. P. Maglio and C. A. Kieliszewski, "Smart Service Systems in the Digital Era," Springer – Service Science Handbook, pp. 13–25, 2020.
[10] A. Kumar and P. Singh, "User-Centered Design in Online Household Service Systems," 2020.
[12] N. M. Indravasan et al., "An Online System for Household Services," Int. J. Engineering Research and Technology (IJERT), May 2018.
[13] A. Agarwal and R. Singh, "Review on On-Demand Home Services in India," 2017.
[14] B. Zhang et al., "A Hybrid Trust Evaluation Framework for E-commerce in Online Social Network," IEEE, 2016.