International Journal of
AI Electronics and Nexus Energy
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0515 www.zesterapublications.com Original Research Paper

Smart Local Service Finder: A Location-Based Service Discovery and Booking Platform

Ms. M. Samyuktha
Asst. Professor
Department of Computer Science and Engineering
Mahatma Gandhi Institute of Technology Kokapet (V),Gandipet (M), Hyd-75, India
Affliated to JNTUH msamyuktha_cse@mgit.ac.in

Ms. K. Shirisha
Asst. Professor
Department of Computer Science and Engineering Mahatma Gandhi Institute of Technology Kokapet (V),Gandipet (M),Hyd-75, India Affliated to JNTUH
kshirisha_cse@mgit.ac.in

Ch. Bunny
Student of Computer Science and Engineering
Mahatma Gandhi Institute of Technology Hyderabad 500075, India cbunny_cse2405t8@mgit.ac.in

E. Sathwik
Student of Computer Science and Engineering Mahatma Gandhi Institute of Technology Hyderabad 500075, India
esathwik_cse2405u1@mgit.ac.in

Abstract*—The Smart Local Service Finder is a digital platform designed to bridge the gap between service providers and customers by enabling efficient access to local services. In today's fast-paced environment, locating reliable and nearby professionals such as electricians, plumbers, tutors, and healthcare assistants has become a persistent challenge. This project presents the design and implementation of a centralized, web-based service discovery and booking platform. The system is built using React.js as the frontend framework, Node.js and Express.js as the backend, and MongoDB as the database. It incorporates location-based search, K-Nearest Neighbours (KNN) filtering, real-time notifications via Socket.IO, and a three-role architecture comprising Customer, Service Provider, and Admin modules. The platform supports secure authentication, verified reviews, advanced search filters, and direct provider-to-customer communication. The system was tested across all major functional modules and demonstrated reliable service discovery, booking, and management capabilities.

Index Terms*—location-based services, service discovery, KNN algorithm, React.js, Node.js, MongoDB, real-time notifications, booking platform, web application, service marketplace.

I. INTRODUCTION

This paper presents the Smart Local Service Finder, a web-based platform designed to connect customers with verified local service providers efficiently and transparently. The platform addresses the longstanding challenge of service discovery in the digital age by providing a centralized solution that integrates location-based search, real-time availability, and secure user interaction within a single system.

The Fast-Moving Consumer Goods and service industries collectively employ millions of local professionals—electricians, plumbers, technicians, tutors, and healthcare assistants—who lack adequate digital presence. At the same time, consumers continue to rely on inefficient methods such as word-of-mouth referrals, unverified online classifieds, and manual directory searches. These methods are time-consuming, unreliable, and lack transparency. The proposed system directly addresses this gap by offering structured service listing, proximity-based discovery, and a verified review mechanism.

The platform is built around three core modules: the Customer Module, the Service Provider Module, and the Admin Module. Each module is independently functional while maintaining proper data coordination through a RESTful backend. The Customer Module provides registration, login, location-based search with filtering, and booking management. The Service Provider Module enables professionals to list services, manage availability, and respond to bookings. The Admin Module provides centralized control over account verification, user management, and platform oversight.

A. Problem Statement

The rapid growth of digital technology has not resolved the difficulty of finding reliable local service providers. Many users depend on traditional methods such as personal references or unverified online sources, which are time-consuming, unreliable, and lack transparency. No centralized platform exists that efficiently connects customers with verified service providers while offering comparison, credibility verification, and quality assurance. Concurrently, local service providers face challenges in gaining digital visibility and reaching potential customers. Issues such as poor communication channels, unclear pricing structures, absence of feedback mechanisms, and security vulnerabilities further erode trust between customers and service providers.

B. Objectives

The objectives of the Smart Local Service Finder are as follows:

- Develop a centralized, web-based platform using React.js, Node.js, Express.js, and MongoDB that connects customers with nearby verified service providers.
- Implement a location-based search module with filtering by category, price, ratings, and proximity using the Haversine formula and KNN algorithm.
- Build a three-role architecture comprising Customer, Service Provider, and Admin modules with role-specific dashboards and access controls.
- Incorporate a secure JWT-based authentication system with an admin-controlled provider verification workflow.
- Provide real-time notifications using Socket.IO for booking confirmations, status updates, and provider communication.
- Enable a transparent review and rating system to promote trust and informed service selection.

Received: 09-05-2026 | Accepted: 17-06-2026 | Published: 23-06-2026

International Journal of
AI Electronics and Nexus Energy
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0515 www.zesterapublications.com Original Research Paper

C. Existing System and Disadvantages

Existing approaches to local service discovery rely on unstructured methods such as personal referrals, printed directories, and generic search engines without localized personalization. Several significant disadvantages characterize these approaches:

- Time-Consuming Process: Users must manually search multiple platforms, contact several providers, and compare options independently, requiring significant time and effort.
- High Possibility of Errors: Service provider information may be outdated or misleading, leading to poor service selection and unsatisfactory outcomes.
- Poor Resource Utilization: Unequal visibility means some providers are overloaded while others are underutilized.
- Lack of Flexibility: Traditional methods do not support filtering by price, ratings, or proximity.
- No Real-Time Monitoring: Users cannot check provider availability dynamically, making scheduling difficult.
- Limited Scalability: As user and provider numbers increase, traditional discovery methods degrade significantly in effectiveness.

D. Proposed System and Advantages

The proposed Smart Local Service Finder is a fully open-source, web-based platform that automates service discovery and booking through location-based algorithms, structured provider profiles, and a real-time communication layer. Key advantages include:

- Centralized Service Platform: A single portal covering diverse service categories eliminates the need for multi-source manual searching.
- Real-Time Availability: Live availability status prevents scheduling conflicts and reduces user uncertainty.
- Verified Reviews and Ratings: A community-driven feedback mechanism ensures transparency and accountability.
- Advanced Search and Filtering: KNN-based proximity search with multi-criteria filtering provides personalized, relevant results.
- Role-Based Access Control: Separate modules for customers, providers, and administrators ensure security and operational clarity.
- Zero Licensing Cost: The platform is built entirely on open-source technologies with no proprietary licensing requirements.

E. System Requirements

- Operating System: Windows 10 / macOS (latest versions)
- Frontend: React.js 18, React Router 6, Leaflet.js
- Backend: Node.js 18+, Express.js 4.x, Socket.IO 4.x
- Database: MongoDB 6.x, Mongoose ORM
- Code Editor: Visual Studio Code 1.89.0
- Browser: Google Chrome / Microsoft Edge

Hardware Requirements

- Processor: Intel Core i3 or equivalent (minimum)
- RAM: 4 GB minimum, 8 GB recommended
- Hard Disk: 20 GB
- Display: SVGA / HD Monitor (1366×768 or higher)

II. LITERATURE SURVEY

The literature survey provides the theoretical foundation and identifies prior research relevant to the design and development of a location-based service discovery platform. The rapid growth of digital technology and mobile applications has significantly transformed the manner in which users access local services. Earlier systems relied primarily on manual methods such as local directories and word-of-mouth recommendations, which were inefficient, time-consuming, and unreliable. With the advancement of internet infrastructure, online classified platforms and service listing applications emerged, improving convenience. However, these systems continued to exhibit limitations including the absence of real-time updates, unverified information, and insufficient personalization.

Liang H et al. [1] presented a study on improving localization accuracy for location-based services using a hybrid GPS and network-based positioning approach. The system demonstrated enhanced precision in urban environments. However, network latency remained a persistent limitation in real-time tracking scenarios. Masango M et al. [2] proposed a context-aware mobile application for device security utilizing geo-fencing and timeline-based access control. While the system provided accurate GPS-based security enforcement, it required a reliable backup system for offline scenarios.

Hegde V et al. [3] developed a student residential distance calculation system using the Haversine formula combined with K-Means clustering, demonstrating the effectiveness of geographic distance computation for proximity-based applications. Jagdale D et al. [4] explored Android-based identification and authentication using QR code scanning, establishing a reliable mechanism for user verification. Mitra P et al. [5] investigated QR code integration with AES encryption for secure data

International Journal of
AI Electronics and Nexus Energy
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0515 www.zesterapublications.com Original Research Paper

authentication, which provides a foundational approach applicable to provider identity verification in service platforms.

Table 2.1: Literature Survey

Sl.	YOP	Title	Authors
1.	2016	Improving Localization Accuracy for LBS	Liang H et al.
2.	2016	Context-Aware Mobile App for Device Security	Masango M et al.
3.	2016	Student Distance Calc. Using Haversine	Hegde V et al.
4.	2015	Android Identification via QR Code	Jagdale D et al.
5.	2011	QR Code for Data Security & Authentication	Mitra P et al.

III. DESIGN AND METHODOLOGY

The design methodology of the Smart Local Service Finder covers the system architecture, module descriptions, algorithms and techniques employed for service discovery, and the structural and behavioral design diagrams that represent the system's functionality.

A. System Architecture

The system follows a three-tier architecture consisting of the Presentation Layer (React.js frontend), the Application Logic Layer (Node.js/Express.js backend), and the Data Layer (MongoDB database). The frontend communicates with the backend via RESTful HTTP API endpoints, and real-time events are managed through a Socket.IO connection layer.

Components:

- User Interface Layer (React.js + Leaflet.js): Renders three role-based dashboards for Customer, Service Provider, and Admin. Integrates map visualization using Leaflet.js for geographic service discovery.
- REST API Layer (Node.js + Express.js): Processes HTTP requests for authentication, service listing, booking management, and user administration. JWT middleware enforces role-based access control.
- Real-Time Layer (Socket.IO): Manages live notifications for booking confirmations, status updates, and provider-customer communication events.
- Data Layer (MongoDB + Mongoose): Stores user profiles, service listings, booking records, and review data with schema-enforced data validation.
- Geo Services (Leaflet.js + Map Tile API): Provides map rendering, user location tracking, and proximity computation for nearby service discovery.

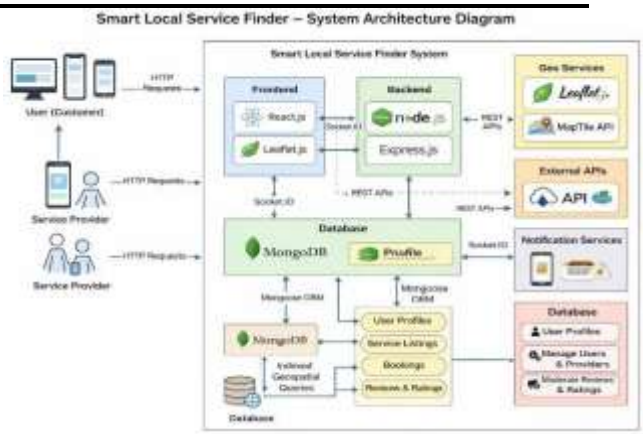

Fig. 3.1. System Architecture

B. Modules

1. Customer Module

The Customer Module provides user registration with role selection, secure JWT-based login, and a dashboard for browsing and booking services. Users can search for providers by category, location, price, and ratings. The module includes a booking confirmation dialog that displays service details, pricing, and a date-time selector before finalizing an appointment.

2. Service Provider Module

The Service Provider Module allows professionals to register on the platform, await admin verification, and upon approval, create and manage service listings. Providers can view incoming booking requests, update service status, and access customer reviews. The provider dashboard displays aggregate metrics including total bookings, average rating, and active service count.

3. Admin Module

The Admin Module serves as the central control panel for platform governance. Administrators can review pending provider registrations, approve or reject accounts, manage all registered users by role (Customer, Provider, Admin), and monitor overall system activity. Role-based filters and a search interface facilitate efficient user management.

4. Booking Management Module

The Booking Management Module handles the complete appointment lifecycle from initial request through confirmation, acceptance, and completion. Customers can view booking status (Pending, Accepted, Completed), and providers can update statuses in real time. The module supports a post-completion review mechanism enabling customers to rate and provide feedback on their service experience.

C. Algorithms and Techniques

1. *Location-Based Search Algorithm (Haversine Formula)* The location-based search module employs the Haversine formula to compute

Received: 09-05-2026 | Accepted: 17-06-2026 | Published: 23-06-2026

International Journal of
AI Electronics and Nexus Energy
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0515 www.zesterapublications.com Original Research Paper

the great-circle distance between the user's coordinates and each registered service provider. Given two points with latitudes φ_1, φ_2 and longitudes λ_1, λ_2, the distance d is computed as:

$$d = 2r \cdot \arcsin\left(\sqrt{\left[\sin^2(\Delta\varphi/2) + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \sin^2(\Delta\lambda/2)\right]}\right)$$

where r is the Earth's mean radius (6,371 km). This formula ensures accurate proximity calculation for services within a configurable search radius, enabling the system to return geographically relevant results.

2. *K-Nearest Neighbours (KNN) Filtering*

The KNN algorithm is applied to rank service providers based on multi-dimensional proximity across geographic distance, price range, and user ratings. Given a query point q and a dataset S of provider feature vectors, the algorithm identifies the k providers with minimum weighted Euclidean distance to q. This technique ensures that search results are not only geographically proximal but also contextually relevant to user-specified filters.

a. Filtering and Ranking Algorithm

A composite scoring function aggregates normalized values for proximity, price competitiveness, and rating score to generate a final provider ranking. Weights for each dimension are configurable, enabling the platform to adapt ranking priority to different service categories. Providers with higher composite scores are displayed at the top of search results.

b. JWT-Based Authentication and Authorization

User authentication is implemented using JSON Web Tokens (JWT). Upon successful login, the server issues a signed token containing user identity and role claims. All protected API endpoints validate this token via an Express.js middleware layer. Role claims (customer, provider, admin) are extracted to enforce access control at the route level, preventing unauthorized cross-role actions.

c. Data Validation Technique

Input validation is enforced at two levels: client-side validation in the React.js frontend (field completeness, format constraints) and server-side validation using Mongoose schema definitions. Schema-level constraints include required fields, data type enforcement, enumerated values for role selection, and minimum/maximum value bounds for pricing fields. Invalid requests are rejected with descriptive error responses before database interaction.

d. Real-Time Notification Technique

Real-time updates are implemented using Socket.IO over WebSocket connections. When a booking is created, updated, or confirmed, the backend emits targeted events to the relevant user sessions. Customers receive booking confirmation and status change notifications; providers receive new booking alerts and cancellation notifications. This bidirectional event model ensures both parties remain synchronized without requiring manual page refreshes.

3. *Libraries and Frameworks*

Frontend: React.js 18 (UI framework), React Router 6 (client-side navigation), Leaflet.js (map visualization). Backend: Node.js 18, Express.js 4.x (REST API server), Socket.IO 4.x (real-time communication), jsonwebtoken (JWT authentication). Database: MongoDB 6.x, Mongoose ORM (schema and validation). Development Tools: npm, Visual Studio Code 1.89.0.

IV. IMPLEMENTATION

The implementation of the Smart Local Service Finder integrates the user interface, processing logic, and data storage into a fully functional, locally deployable web application. The system is developed following a modular approach wherein each component—input handling, business logic, and output presentation—operates independently while maintaining coordinated data flow through the REST API layer.

The frontend is implemented as a single-page React application with role-aware routing managed by React Router. All state management is performed using React useState and useEffect hooks. The backend exposes a RESTful API through Express.js routes segregated by domain (authentication, services, bookings, users). All state management in the frontend is performed using React useState hooks. The application is fully offline-capable once started locally, with no mandatory external API dependencies for core functionality.

The App.jsx entry point defines protected route structures that redirect unauthenticated users to the login page. The serviceController.js module handles service creation and retrieval. The bookingController.js module manages the full booking lifecycle. The bookingRoutes.js file registers routes with JWT middleware protection, ensuring all booking operations require valid authentication.

V. RESULTS AND DISCUSSION

The Smart Local Service Finder was tested across all major functional modules. System behavior was validated for role-based access control, service search and filtering, booking lifecycle management, real-time notifications, and admin user management. Results consistently demonstrated correct module behavior across all tested scenarios.

A. Registration Page

Fig. 5.1 shows the Registration Page of the Smart Local Service Finder. The interface provides input fields for Full Name, Phone Number, Role Selection (Customer / Provider / Admin), Email Address, and Password. Upon valid submission, the system creates a user account and redirects the user to the login page. Incomplete or invalid inputs trigger contextual error messages. Provider accounts enter a

International Journal of
AI Electronics and Nexus Energy
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0515 www.zesterapublications.com Original Research Paper

pending verification state until approved by an administrator.

Fig. 5.1. Registration Page

B. Login and Authentication

Fig. 5.2 shows the Login Page. Users enter their registered email and password to authenticate. Upon successful validation, the system issues a JWT and redirects the user to the role-appropriate dashboard. Invalid credentials trigger an error message, and the system prevents unauthorized access. This module ensures secure platform access and maintains data integrity per authenticated session.

Fig. 5.2. Login Page

C. Account Approval Workflow

Fig. 5.3 shows the Account Pending Approval Page for newly registered service providers. After registration, provider accounts are not immediately activated. The page informs the provider that their account is under administrative review and displays an estimated approval period of 24 to 48 hours. This workflow ensures that only verified professionals are listed on the platform, improving trust and service quality.

Fig. 5.3. Approval Page

D. Admin Dashboard

Fig. 5.4 shows the Admin Dashboard. The interface presents a structured user management table displaying registered users with their names, email addresses, roles, and account statuses (Approved, Pending, or N/A). A search bar and role-based filter enable rapid user lookup. Action buttons allow the administrator to approve provider accounts, modify permissions, or remove users. This dashboard ensures efficient platform governance and verification oversight.

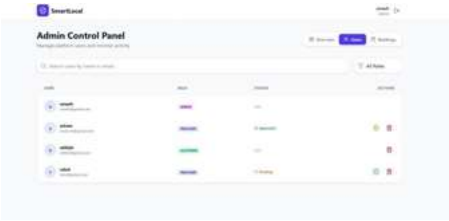

Fig. 5.4. Admin Dashboard

E. Provider Dashboard

Fig. 5.5 shows the Provider Dashboard. The interface displays aggregate metrics including Total Bookings, Average Rating, and Active Services. Section tabs provide access to service management (Services), incoming requests (Bookings), and customer feedback (Reviews). The Recent Bookings section lists the latest requests with their current status. The Recent Reviews section displays customer feedback entries, supporting continuous service improvement.

International Journal of
AI Electronics and Nexus Energy
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0515 www.zesterapublications.com Original Research Paper

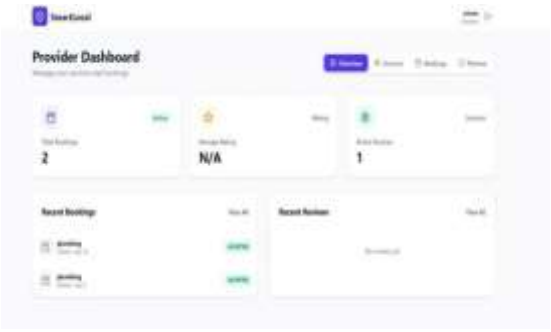

Fig. 5.5. Provider Dashboard

F. Service Finder and Booking

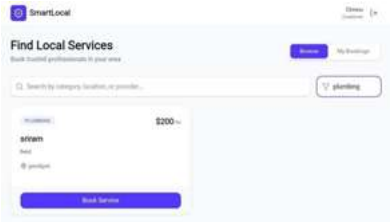

Fig. 5. 6. Booking

Fig. 5.6 shows the Service Finder dashboard, where customers search and browse local service providers. Each result card displays the provider's service category, hourly price, description, and location. Users can apply category filters (e.g., Plumbing, Electrical) and initiate a booking using the "Book Service" button. Fig. 5.7 shows the Booking Confirmation Dialog, which presents the selected provider's details, a date-time selector, the service fee, and total cost before final confirmation.

G. Customer Booking Dashboard

Fig. 5.8 shows the Customer Booking Dashboard. Users can view all existing appointments with details including service type, provider identification, scheduled date and time, and contact information. Booking statuses (e.g., Accepted, Pending) are prominently displayed. A

"Complete and Review" action allows customers to mark completed appointments and submit ratings, closing the booking lifecycle and contributing to the review database.

Fig. 5.8. Customer Booking Dashboard

H. Accuracy of Results

The system consistently produced accurate results across all tested input combinations. Role-based access control correctly restricted operations to authorized modules. The KNN-based proximity search returned geographically and contextually relevant service providers for varied query inputs. Booking status transitions were correctly propagated through the real-time notification layer. Authentication and session management functioned reliably across all role types.

I. Limitations

The current implementation does not include an integrated online payment gateway. The recommendation logic is rule-based and does not yet incorporate machine learning-driven personalization. The platform currently supports single-city or single-region deployment without multi-geographic scaling. The system does not consider advanced contextual factors such as weather patterns, traffic conditions, or seasonal demand fluctuations.

VI. CONCLUSION AND FUTURE SCOPE

A. Conclusion

The proposed Smart Local Service Finder successfully delivers a practical, efficient, and scalable solution for connecting customers with nearby verified service providers. By integrating modern web technologies, location-based algorithms, and a role-based modular architecture, the system simplifies the process of service discovery, comparison, and booking while ensuring transparency and accountability through verified reviews and admin-controlled provider verification.

The platform benefits both service consumers, who experience reduced search time and improved

International Journal of
AI Electronics and Nexus Energy
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0515 www.zesterapublications.com Original Research Paper

service quality, and local service providers, who gain structured digital visibility and an expanded customer base. The system's open-source technology stack eliminates licensing costs and provides a foundation well-suited to iterative enhancement. All stated objectives were achieved, and the system demonstrates the effectiveness of a lightweight, full-stack web platform for organized local service discovery.

B. Future Scope

Future enhancements include:

- Integration of a secure online payment gateway enabling in-platform transaction completion.
- Machine learning-based recommendation engine using collaborative filtering to suggest services based on user history and behavioral patterns.
- Development of a native mobile application (iOS and Android) for improved accessibility by field-based service providers.
- Multi-language support and voice-based search to serve linguistically diverse user populations.
- Real -time GPS tracking of service provider arrival for time-sensitive appointments.
- Expansion to multiple cities and regions with dynamic geographic partitioning and load balancing.

VII. REFERENCES

[1]H. Liang, X. Zhang, and W. Chen, "Improving Localization Accuracy for Location-Based Services Using HybridPositioning," in Proc. IEEE National Conf. on Management Science, 2016, pp. 112–118.

[2] M. Masango, J. Steele, and P. Sithole, "Context-Aware Mobile Application for Device Security Using Geo-Fencing and Timeline Access Control," in Proc. IEEE Int. Conf. on Emerging Technologies, 2016, pp. 45–51.

[3] V. Hegde, R. Kamat, and S. Nayak, "Student Residential Distance Calculation Using Haversine Formula and K-Means Clustering," in Proc. IEEE Int. Conf. on Computational Intelligence and Computing Research (ICCICR), 2016, pp. 234–239.

[4] D. Jagdale, S. Shinde, and P. Patil, "Android-Based Identification and Authentication via QR Code Scanning," TELKOMNIKA Indonesian J. Electrical Engineering, vol. 13, no. 2, pp. 421–428, 2015.

[5] P. Mitra and S. Banerjee, "QR Code for Data Security and Authentication Using AES Encryption," Int. J. Mobile Applications (IJMA), vol. 3, no. 4, pp. 78–85, 2011.

[6] S. Choudhury, P. Das, and R. Banerjee, "Mobile Edge Cloud Framework for Real-Time Applications," IEEE Access, vol. 10, 2022.

[7] J. O. Oyekan, A. A. Adebayo, and M. K. Joseph, "Web Service Discovery: Rationale, Challenges, and Solution Directions," Computer Standards & Interfaces, Elsevier, 2023.

[8] B. P. Singh and M. Margam, "Mobile Apps-Based Applications: A Systematic Review," Int. J. Librarianship, vol. 8, no. 3, 2023.

[9] J. Dandu, R. Kumar, and P. Sharma, "Servo: The Service Providing Mobile Application," in Proc. IEEE Int. Conf. on Artificial Intelligence and Smart Systems (ICAISS), 2023.

[10] G. Zhang, Y. Liu, and H. Chen, "Design of Intelligent Service Applications Using Kano Model," in Proc. IEEE Int. Conf. on Interaction Design (ICID), 2024.