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 3, November 2026
Impact Factor: 8.2

Customer-Centric Local Service Marketplace

Gawali Smita Arun[1], Jagtap Aniket Bhaskar[2], Bhonde Rohan Raju[3], Bhagwat Pragati Dnyaneshwar[4], Prof. A. B. Jadhav[5], Prof. P. R. Bhor[6], Dr. H. B. Jadhav[7]

[1,2,3,4]Students Department, of Computer Engineering
[5,6]Assistant Professor, Department of Computer Engineering
[7] HOD, Department of Computer Engineering
Adsul's Technical Campus, Chas, Ahilyanagar, Maharashtra, India

Abstract: *The increasing reliance on mobile applications for everyday tasks has highlighted the need for efficient discovery of local service providers. Users often face difficulties in locating reliable electricians, plumbers, carpenters, and pest-control services through fragmented sources of information. This paper presents Customer-Centric Local Service Marketplace, a native Android application developed to simplify access to sample local service provider information. The application is implemented in Java with XML-based user interfaces and uses a locally bundled JSON asset (providers.json) for data storage. Provider records are loaded at runtime by a dedicated data-management component, converted into Java objects, and held in memory using ArrayList collections. Users can browse predefined service categories, search providers by name, filter results by city through text matching, view detailed provider information, initiate phone calls via the ACTION_DIAL intent, and open map applications through geo URIs constructed from address and city fields. The system does not employ remote databases, backend servers, or real-time synchronization. The implemented features demonstrate a practical, offline-capable approach to organizing and presenting local service information within a mobile interface. The work contributes a clear technical description of a JSON-driven Android architecture suitable for educational and prototype contexts.*

Keywords: Android Application, Java, JSON, Local Service Discovery, Mobile Application, Service Marketplace

I. INTRODUCTION

Mobile applications have become an integral part of daily life, supporting communication, commerce, navigation, and access to local services. The proliferation of smartphones has increased expectations that relevant information should be discoverable quickly and presented in an organized manner. In the domain of local services such as electrical work, plumbing, carpentry, and pest control, users frequently encounter fragmented information distributed across informal directories, social media groups, or general search engines. This fragmentation can lead to inefficient search processes and difficulty in comparing available providers.

The motivation for the present work arises from the practical need to provide a simple, centralized mobile interface for discovering sample local service provider details. Many existing commercial platforms rely on complex backend infrastructures, user authentication, real-time availability updates, and payment systems. While such features are valuable at scale, they introduce development and operational complexity that may not be necessary for educational prototypes or limited-scope deployments. An offline-capable application that organizes provider information by category and supports basic search and filtering can still deliver tangible usability benefits.

The research problem addressed by this project is the lack of a lightweight, self-contained Android application that presents local service provider information in a structured and searchable form without depending on external servers or databases. The primary objectives of the work are: (1) to design and implement a native Android application using Java and XML that allows category-based browsing of service providers; (2) to load and process provider data from a local JSON asset; (3) to support provider-name search and city-based text filtering; (4) to enable interaction with device

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 3, November 2026
Impact Factor: 8.2

phone and map applications through standard Android intents; and (5) to document the architecture and implementation accurately for academic evaluation.

The scope of the system is limited to the presentation and interaction with sample provider data bundled within the application package. The application does not support provider registration, user accounts, live location tracking, appointment booking, online payments, or automatic data updates. Its main contribution is a clearly described, fully local JSON-based architecture that demonstrates practical techniques for data loading, in-memory filtering, and external application integration on the Android platform.

Method

System Overview

Customer-Centric Local Service Marketplace is a native Android application whose package name is com.example.localservicefinder. It is developed in Android Studio using the Java programming language for application logic and XML for user interface layouts. Upon launch, the application loads a collection of sample service provider records from a JSON file stored in the application assets. Users interact with a home screen that displays service categories and a search field. Selecting a category or submitting a search navigates the user to a provider listing screen where further filtering by name and city is possible. Selecting an individual provider opens a detail screen that displays contact information, address, rating, and description, and provides buttons to open the device dialer or a map application.

Figure 1: Overall System Architecture

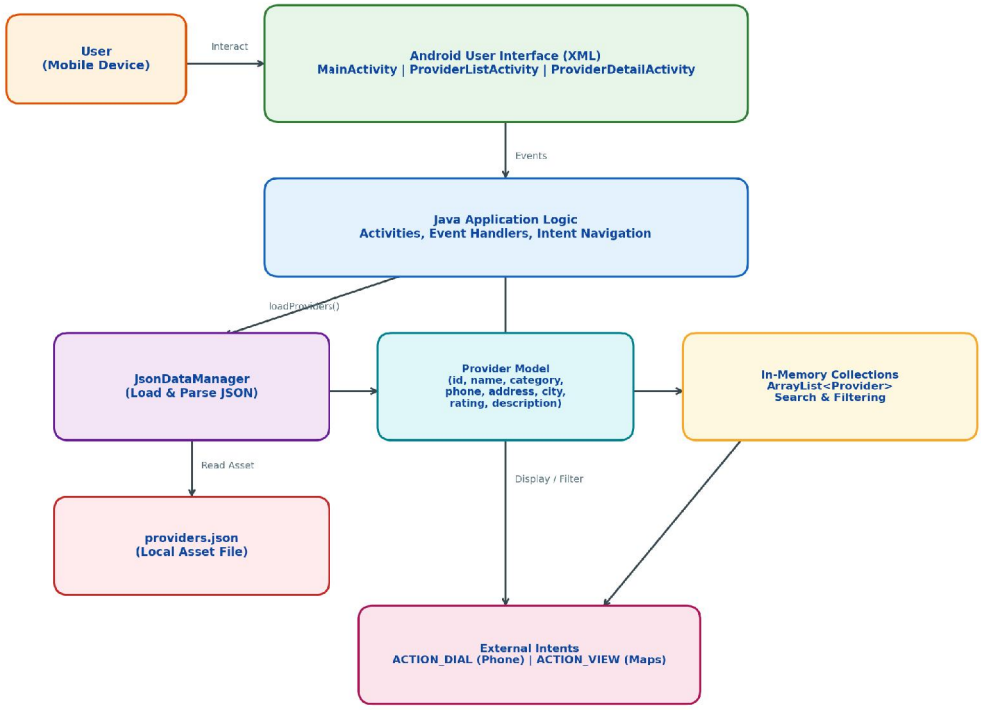


Figure 1. Overall System Architecture of the Customer-Centric Local Service Marketplace.

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 3, November 2026
Impact Factor: 8.2

System Architecture

The architecture consists of four principal layers. The presentation layer comprises three activities: MainActivity (home screen), ProviderListActivity (listing and filtering), and ProviderDetailActivity (detailed view and external intents). The domain layer is represented by the Provider model class, which encapsulates the attributes of a service provider. The data layer is implemented by JsonDataManager, which reads the providers.json asset, parses the JSON structure using the org.json package, and returns an ArrayList of Provider objects. Finally, the integration layer uses Android Intent mechanisms to launch the system phone dialer (ACTION_DIAL) and map applications (ACTION_VIEW with a geo URI). All provider processing after the initial load occurs entirely in memory; no persistent local database or remote service is used.

Technology Stack

The application is written in Java and targets the Android platform. User interfaces are defined in XML layout resources (activity_main.xml, activity_provider_list.xml, and activity_provider_detail.xml) and styled with drawable resources such as bg_rounded_white.xml and bg_primary_button.xml. Data interchange relies on the JSON format, which is well supported by the Android org.json library. Navigation between screens is performed with explicit Intents that carry category selections, search queries, or individual provider attribute values as extras. External integration is achieved through the standard ACTION_DIAL and ACTION_VIEW intent actions, allowing the application to remain free of custom telephony or mapping code.

Data Management

Provider information is stored in the asset file providers.json. The file contains a root object with a "providers" array. Each array element is a JSON object possessing the fields id (integer), name, category, phone, address, city, rating (numeric), and description. At runtime, JsonDataManager obtains an InputStream from the asset manager, reads the entire content into a string via a BufferedReader, and constructs a JSONObject. The providers array is retrieved and each element is mapped to a new Provider instance. The resulting ArrayList is returned to the calling activity. Because the data set is small and static, it is reloaded independently by MainActivity and ProviderListActivity rather than being shared through a global singleton or ViewModel. This design keeps the implementation simple and avoids lifecycle-related state management complexity.

Figure 3: JSON Data Processing Workflow

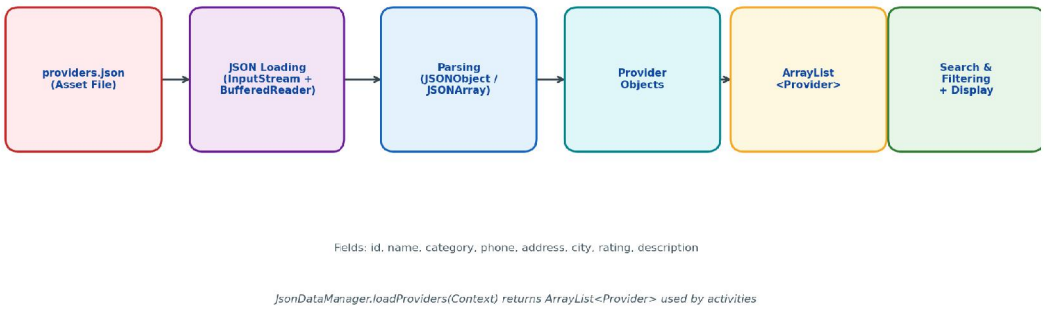


Figure 3. JSON Data Processing Workflow from asset file to filtered display.

Search and Filtering Method

ProviderListActivity maintains two ArrayList collections: allProviders (the complete loaded set) and filteredProviders (the current result set). Filtering is performed by the private method filterProviders(). The method reads the current text

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 3, November 2026
Impact Factor: 8.2

of the name search EditText and the city search EditText, converts both to lower case, and iterates over allProviders. A provider is retained when three conditions hold simultaneously: (1) the selected category is "ALL" or exactly matches the provider's category (case-insensitive); (2) the provider name contains the name query as a substring; and (3) the provider city contains the city query as a substring. Empty query strings act as wildcards. After the filtered list is constructed, the activity updates a count TextView and regenerates the visual cards inside a LinearLayout container. TextWatchers attached to both EditTexts invoke filterProviders() on every character change, producing immediate interactive feedback.

Provider Interaction

When a provider card is tapped, ProviderListActivity starts ProviderDetailActivity and passes the provider's attributes as Intent extras. The detail activity populates TextViews with name, category, phone, address, city, rating, and description. The "Call Provider" button constructs an Intent with action ACTION_DIAL and a tel: URI containing the encoded phone number; the system then opens the dialer with the number pre-filled, leaving the actual call initiation to the user. The "View Map" button constructs a geo:0,0?q= URI whose query parameter is the concatenation of address and city, and launches it with ACTION_VIEW. Any installed map application that handles geo URIs may respond. No GPS coordinates are stored or obtained at runtime; the map application is responsible for geocoding the textual address.

Application Workflow

The complete user journey proceeds as follows. The user launches the application and is presented with the home screen showing the total number of loaded providers, a search field, and four category buttons (Electrician, Plumber, Carpenter, Pest Control). Entering a search query and confirming it, or tapping a category button, starts ProviderListActivity with the appropriate CATEGORY and optional SEARCH_QUERY extras. On the listing screen the user may further refine results by typing in the name or city filter fields. Tapping a provider card opens the detail screen. From the detail screen the user may return to the list, open the dialer, or open a map application. All data remains local; network connectivity is not required for any of these operations.

Figure 2: Application Workflow

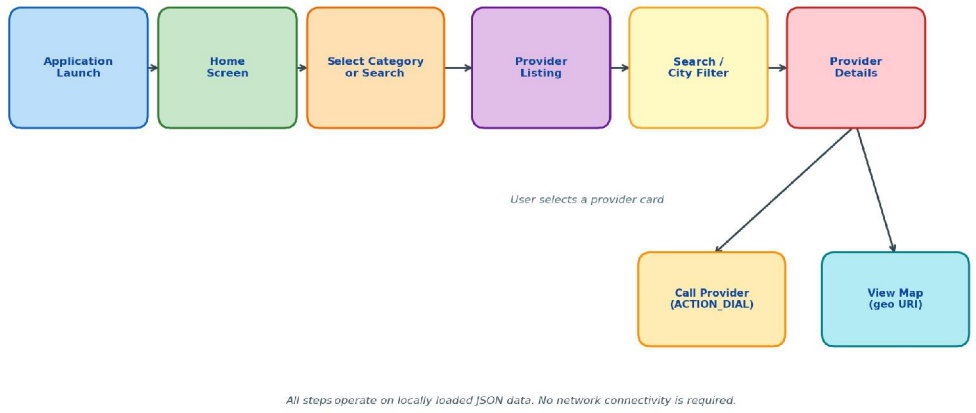


Figure 2. Application Workflow from launch to provider contact or map view.

II. RESULTS AND DISCUSSION

The implemented application successfully realizes the intended functionality using only local resources. The home screen presents a clean blue-and-white interface with clearly labeled category buttons and a prominent search field.

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 3, November 2026 Impact Factor: 8.2

Selecting any category navigates to a list of matching providers rendered as rounded white cards containing name, category icon, address, city, and rating. Real-time filtering by name and city updates the displayed cards and the result count without perceptible delay for the small sample data set.

The provider detail screen displays all stored attributes in a readable vertical layout and offers two action buttons. The dialer integration correctly opens the system phone application with the selected number, and the map integration correctly launches a geo URI that map applications can resolve. These behaviors confirm that the Intent-based external integration operates as designed.

Because the application relies exclusively on a static JSON asset, its behavior is deterministic and independent of network conditions. This characteristic is advantageous for demonstration and offline use. At the same time, it constitutes a clear limitation: the set of providers cannot be updated without rebuilding and redistributing the application package. The current sample data set contains five providers, all located in Pune, which is sufficient to exercise every code path but is not representative of a production service directory.

No formal usability study, performance benchmark, or multi-device test campaign was conducted as part of this academic project. Consequently, claims regarding user satisfaction, search latency under large data sets, or battery impact cannot be supported by measured evidence. The results reported here are limited to functional verification of the implemented features against the stated requirements.

III. CONCLUSION

This paper has presented the design and implementation of Customer-Centric Local Service Marketplace, a native Android application that organizes sample local service provider information for convenient mobile access. The application addresses the practical difficulty of discovering service providers by offering category browsing, name search, city filtering, detailed information display, and seamless integration with the device dialer and map applications. All data are stored in a local JSON asset and processed entirely in memory using Java collections, resulting in a self-contained system that requires no backend infrastructure. The implemented features fulfill the stated project objectives while remaining transparent about the absence of remote synchronization, user accounts, or real-time updates. The work provides a concrete example of a JSON-driven Android architecture that can serve as a foundation for further educational or prototype development.

IV. FUTURE SCOPE

Several enhancements can be considered for future iterations of the system. Integration of a backend server and a relational or NoSQL database would permit dynamic provider registration and real-time information updates. User authentication would enable personalized features such as favorites and history. GPS-based nearby search could replace or complement the current textual city filter. Additional capabilities such as online appointment booking, customer reviews, payment integration, provider dashboards, and push notifications would move the application closer to a production-grade marketplace. Cloud synchronization would allow the same data to be accessed from multiple devices. Each of these features lies outside the current implementation and would require substantial additional design and development effort.

ACKNOWLEDGMENTS

The authors express their sincere gratitude to the project guides A. B. Jadhav, P. R. Bhor, and Dr. H. B. Jadhav for their continuous guidance, constructive feedback, and support throughout the development of this project. Their expertise and encouragement were instrumental in shaping both the technical implementation and the academic presentation of this work.

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 3, November 2026
Impact Factor: 8.2

Author Contributions

[Author contribution statement to be confirmed by the project team. A possible formulation is: All authors contributed to the conception, design, implementation, and documentation of the application. Specific module-level responsibilities may be detailed here after internal confirmation.]

Conflicts of Interest

[The authors declare that they have no conflicts of interest related to this work. This statement is subject to final verification by the authors prior to submission.]

REFERENCES

1. Android Developers, "Data and file storage overview," Android Developers Documentation. [Online]. Available: https://developer.android.com/training/data-storage. [Accessed: Nov. 12, 2026].
2. Android Developers, "App architecture: Data layer," Android Developers Documentation. [Online]. Available: https://developer.android.com/topic/architecture/data-layer. [Accessed: Nov. 12, 2026].
3. Android Developers, "Intents and Intent Filters," Android Developers Documentation. [Online]. Available: https://developer.android.com/guide/components/intents-filters. [Accessed: Nov. 12, 2026].
4. Android Developers, "JSONObject," Android API Reference. [Online]. Available: https://developer.android.com/reference/org/json/JSONObject. [Accessed: Nov. 12, 2026].
5. M. S. Kamal Bashah, N. Kryvinska, and D. van Thanh, "Novel service discovery techniques for open mobile environments and their business applications," in Exploring Services Science, M. Snene, Ed. Berlin, Germany: Springer, 2012, pp. 146-157.
6. Roumani and J. G. D. da Costa, "Mobile services discovery and selection in the publish/subscribe paradigm," in Proc. ACM Int. Workshop Data Eng. Wireless Mobile Access, 2004.
7. S. Garba, R. Mohamad, and N. A. Saadon, "Web service discovery approaches for dynamic mobile environment: A systematic literature review," Int. J. E-Services Mobile Appl., vol. 13, no. 4, pp. 16-38, 2021.
8. K. Kim and J. Lee, "Identifying and structuring service functions of mobile applications in Google's Android Market," Inf. Syst. e-Bus. Manage., vol. 16, pp. 383-406, 2018.
9. A R. Mawlood-Yunis, "Parsing remote XML and JSON files, using HTTPUrlConnection, XmlPullParser, and AsyncTask," in Android for Java Programmers. Cham, Switzerland: Springer, 2022, pp. 433-474.
10. N. Z. B. Zulkifli and S. B. M. Noah, "Data exchange model for classified advertisement using Android," in Proc. IEEE Student Conf. Research and Development (SCOReD), 2015, pp. 1-5.