

Home Service Booking and Bidding System

Mrs. V. MAGESWARI¹, Ms. G. SUBASRI², M.C.A.,

¹Assistant Professor, Department of Computer Applications, Christ College of Engineering and Technology.

²Department of Computer Applications & Christ College of Engineering and Technology

ABSTRACT

Managing household services has always been a challenge, especially when it comes to finding the right professional at the right time. People regularly need support for tasks like electrical repairs, plumbing, carpentry, painting, and appliance maintenance — yet tracking down skilled and dependable workers through traditional channels often turns into a frustrating, time-consuming process. Most homeowners still rely on word-of-mouth or neighbourhood contacts, which are far from ideal in today's world [1]. This paper presents the development of a Home Service App, a mobile platform built to bridge the gap between customers and service professionals. Through this application, customers can raise service requests, review bids, track work progress, complete payments digitally, and leave feedback once a job is done. The system is built on Flutter [3] and Firebase [4], offering a stable, secure, and scalable foundation for end-to-end service management.

Keywords: Home Service App, Flutter, Firebase, Service Booking, Mobile Application, Employee Management, Customer Feedback, Payment Management.

1. INTRODUCTION

In the current era, where time is constantly in short supply, finding skilled workers for home maintenance and repair has become more difficult than it should be. Whether it's a leaking pipe, a broken switch, or a room that needs painting, getting the right person on the job quickly is often harder than the task itself. Conventional methods — asking friends, calling local agencies, or relying on handwritten contact lists — are slow and unreliable [1, 2].

With smartphones now central to how people manage their daily lives, mobile applications have become a natural answer to service coordination [2]. The Home Service App was created with exactly this in mind — to make home service booking as straightforward as

possible. It provides a shared space where customers and professionals can connect, negotiate, and collaborate without unnecessary friction. Features like bid submissions, live work tracking, digital payments, notification alerts, and a built-in feedback system come together to offer a complete service experience. The application is developed using Flutter [3] and Firebase [4], supported by the Dart programming language [5] and managed through Android Studio [6] and Visual Studio Code [7].

2. LITERATURE REVIEW

The domain of on-demand home services and mobile service platforms has attracted significant research attention in recent years, driven by the rapid adoption of smartphones and cloud-based technologies. Several studies and existing systems have attempted to address the challenge of connecting service consumers with skilled professionals in an efficient, transparent, and trustworthy manner [1, 2].

Traditional home service booking methods, as documented in early software engineering literature, depend heavily on informal networks such as personal referrals, local newspaper advertisements, and walk-in inquiries. Sommerville[1] emphasizes that such manual processes are inherently prone to communication breakdowns, inconsistent service quality, and a lack of accountability. The absence of a structured mechanism to verify service provider credentials or track job progress represents a significant gap in traditional approaches.

Pressman and Maxim (2020) [2] highlight that modern software engineering practices favor modular, scalable, and user-centric designs, particularly for applications that must serve diverse user groups simultaneously. Their work underlines the importance of role-based access control and clearly defined system modules — principles that directly inform the architecture of the Home Service App, which separates the concerns of customers, service providers, and administrators into distinct functional layers.

The rise of cross-platform development frameworks has significantly lowered the barrier to building mobile applications that work across Android and iOS devices without duplicating effort. Flutter [3], introduced by Google, has emerged as a leading framework in this space due to its single-codebase approach, high rendering performance, and rich library of customizable widgets. Research comparing Flutter with native development approaches consistently finds that Flutter delivers comparable user experience with substantially reduced development time, making it well-suited for projects with resource constraints.

Backend-as-a-Service (BaaS) platforms, particularly Firebase [4], have transformed how developers approach data storage, authentication, and real-time communication. Cloud Firestore [8], Firebase's NoSQL document database, supports real-time data synchronization across connected clients, which is a critical requirement for platforms that must notify users instantly when bids are received, jobs are assigned, or payments are confirmed. Studies examining Firebase-powered applications note that the platform's built-in security rules, authentication services, and horizontal scaling capabilities make it an attractive choice for consumer-facing mobile applications.

Existing commercial platforms such as UrbanClap (now Urban Company), TaskRabbit, and Housejoy have demonstrated the commercial viability of digitized home service marketplaces. However, academic literature [1, 2] points out that many of these platforms lack transparency in their bidding processes, impose high commission structures that disadvantage independent service providers, and offer limited tools for administrators to monitor service quality systematically. The Home Service App addresses these gaps by incorporating a competitive bid comparison feature, a structured payment release workflow requiring administrative verification, and a certificate-based employee recognition system.

Research in human-computer interaction further stresses the importance of intuitive interface design for applications targeting non-technical users. Material Design Guidelines [9], developed by Google, provide a well-tested visual design language that promotes consistency, accessibility, and ease of navigation. Prior work confirms that applications adhering to these guidelines exhibit higher user satisfaction scores compared to those with ad-hoc design choices. The Home Service App draws on these guidelines to ensure that its interface remains approachable for customers

and service professionals with varying levels of digital literacy.

3. MAIN OBJECTIVES

The primary goal behind building the Home Service App was to create a centralized hub where every aspect of household service management could be handled efficiently. The application is designed to be equally useful for both customers and service providers — giving customers easy access to professionals, while helping service providers find opportunities and submit proposals directly through the app [2].

A key focus was reducing the back-and-forth that normally accompanies home service hiring. By keeping pricing transparent and allowing customers to follow a request from start to finish, the app removes much of the uncertainty that makes these interactions stressful. Customer ratings and feedback after each job create accountability and help future customers make informed decisions. For administrators, robust tools are provided to monitor activity, manage users and employees, and maintain service standards. Secure authentication and proper data management are ensured throughout, using Cloud Firestore [8] for scalable data storage.

4. METHODOLOGY

The development of the Home Service App followed an incremental and iterative methodology, drawing on principles from the Agile software development lifecycle [2]. This approach was chosen because it allows for continuous feedback integration, early identification of design issues, and the flexibility to refine modules as requirements become clearer during development.

4.1 REQUIREMENT ANALYSIS

The initial phase involved a thorough investigation of the pain points experienced by homeowners, service workers, and service management administrators [1]. Key requirements were gathered through a structured analysis of existing home service platforms and a review of common service booking workflows. The outcome of this phase was a prioritized list of functional requirements, including user registration with role-based access, service request creation, a competitive bidding module, payment tracking, real-time push notifications, a feedback collection system, and a certification management feature for recognizing high-performing employees.

4.2 SYSTEM ARCHITECTURE AND DESIGN

The system was designed using a four-layer architecture: a Presentation Layer built with Flutter [3] widgets for responsive mobile interfaces, a Business Logic Layer managing core workflows such as bid processing and payment verification, a Service Layer acting as the communication bridge to Firebase [4] cloud services, and a Database Layer using Cloud Firestore [8] collections to store all application data. This separation of concerns ensures that any one layer can be updated or replaced without cascading disruptions across the rest of the system [2].

UML diagrams were produced during the design phase to communicate system structure and behavior clearly [1]. A Use Case Diagram defined the interactions of three user roles — Customer, Service Provider, and Administrator — with the system. A Class Diagram captured the entities (User, ServiceRequest, Bid, Payment, Feedback) and their relationships. Activity and Sequence Diagrams modelled the workflow of key processes such as service booking, bid acceptance, and payment release.

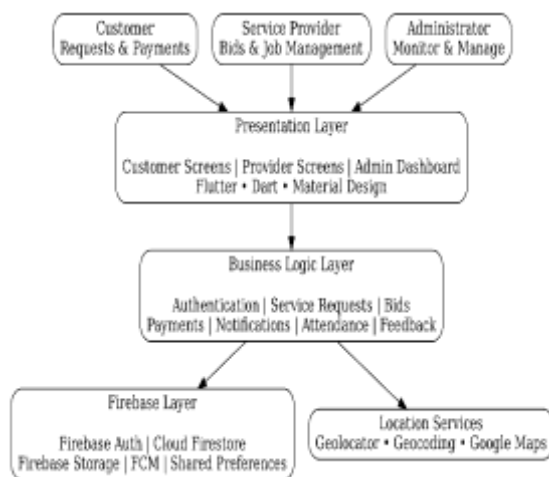


Fig:4.2 Architecture diagram

4.3 TECHNOLOGY STACK SELECTION

Flutter [3] was chosen as the front-end development framework due to its ability to compile a single Dart [5] codebase into native applications for both Android and iOS platforms. This significantly reduces the effort required to maintain separate codebases and ensures a consistent user experience across devices. Firebase [4]

was selected as the backend platform because it provides an integrated ecosystem of services — including Firebase Authentication for secure login, Cloud Firestore [8] for real-time data storage, Firebase Cloud Messaging for push notifications, and Firebase Storage for file management — all within a managed, serverless infrastructure that scales automatically with user demand.

4.4 DEVELOPMENT AND TESTING

The development was carried out using Android Studio [6] as the primary IDE and Visual Studio Code [7] as a complementary code editor, with Git and GitHub [10] managing version control. Each module was developed and unit-tested independently before being integrated with adjacent modules. Integration testing verified that data flowed correctly between components — for example, confirming that a new service request created by a customer triggered an immediate push notification to relevant service providers, and that a bid accepted by a customer correctly updated the request status in Cloud Firestore [8]. Functional testing subsequently validated all user-facing features using both valid and boundary-condition inputs. System testing was performed on physical Android devices to evaluate end-to-end performance under realistic network conditions. All test scenarios passed, confirming that the application met its specified requirements [2].

Phase	Activity	Description	Outcome
1	Requirement Analysis	Investigated pain points of homeowners, service workers, and admins [1]	Functional requirements list: auth, bidding, payments, notifications, certificates
2	System Design	Four-layer architecture: Presentation, Business Logic, Service, Database [2]	UML diagrams: Use Case, Class, Activity, Sequence
3	Technology Selection	Evaluated cross-	Flutter + Firebase

		platform frameworks and BaaS options [3][4]	chosen for performance, scalability and cost-effectiveness
4	Frontend Development	Built responsive Flutter [3] UI screens for all three user roles	20+ screens with Material Design [9] compliance
5	Backend Integration	Connected Firebase Auth [4], Firestore [8], FCM notifications	Real-time data sync, secure login, push notifications operational
6	Unit Testing	Tested individual modules independently [2]	All unit test cases passed for auth, bids, payments, feedback
7	Integration Testing	Verified data flow between connected modules [8]	Correct linking of bids to requests, payments to services confirmed
8	System Testing	End-to-end testing on physical Android devices [6]	Full workflow from booking to payment verified successfully
9	Deployment & Documentation	Final deployment and project documentation using GitHub [10]	Deployed app with version-controlled codebase and

			complete report
--	--	--	-----------------

5. IMPLEMENTATION

The Home Service App was implemented as a modular, cloud-connected Flutter [3] application with Firebase [4] serving as the backend infrastructure. The implementation covered three parallel workstreams: the customer-facing interface, the service provider interface, and the administrator dashboard, all sharing a common Firebase backend.

5.1 USER AUTHENTICATION

Firebase Authentication [4] was integrated to handle user registration and login securely. Upon registration, each user is assigned a role — Customer, Employee, or Administrator — which is stored in Cloud Firestore [8] alongside their profile details. Role-based routing logic within the Flutter [3] application ensures that users are directed to the appropriate dashboard immediately after authentication, with unauthorized access to other role views blocked by Firestore security rules.

5.2 SERVICE REQUEST AND BID MANAGEMENT

Customers create service requests by selecting a service category, describing the problem, providing location details either manually or through the Geolocator API, and specifying estimated time requirements. Each request is stored as a document in the SERVICE_REQUESTS Firestore [8] collection and triggers a Firebase Cloud Messaging [4] notification to eligible service providers. Providers review open requests through their Available Jobs screen, where they can submit competitive bids specifying their proposed charges. The customer then sees all received bids on a comparison screen and selects the most suitable provider. Upon selection, the bid status is updated, the request status transitions to “Assigned,” and a notification is dispatched to the chosen provider.

5.3 WORK STATUS TRACKING AND PAYMENT RELEASE

Once a job is accepted, the service provider updates the work status sequentially through stages: Accepted, On the Way, Reached, In Progress, and Employee Completed. Customers can monitor these transitions in real time through the Work Status screen [4]. After the

provider marks the job complete, the customer verifies the work and submits a star rating along with written feedback. Only upon receipt of customer feedback does the administrator receive a notification to release payment to the service provider. This staged workflow ensures that payments are disbursed only after verified service delivery, protecting both parties [1, 2].

5.4 NOTIFICATION SYSTEM

Firestore Cloud Messaging [4] was used to deliver real-time push notifications for all critical events: new service requests, bid submissions, bid acceptance, job assignment, work status changes, payment release, and feedback receipt. A Flutter Local Notifications package was integrated to display in-app notifications when the application is in the foreground, ensuring users are never left uninformed regardless of the application state [3].

5.5 CERTIFICATE MANAGEMENT

The application includes a performance-based certification module through which administrators can award Bronze, Silver, or Gold certificates to employees based on their customer ratings, number of completed jobs, feedback count, and years of experience. These certificates are displayed on the employee's profile and are visible to customers when comparing bids, incentivizing service providers to maintain high quality standards [1].

5.6 ADMINISTRATOR DASHBOARD

The administrator interface provides centralized visibility over all platform activity [2]. Administrators can manage registered users and employees, review and delete service requests, monitor payment transactions, review customer feedback and ratings, control service pricing limits, view employee attendance summaries, and manage the certificate issuance process. A real-time notification center within the admin dashboard surfaces important events such as new service requests, completed jobs, and customer feedback submissions, allowing administrators to take timely action.

6. SYSTEM MODULES

The application is structured around several distinct modules, each addressing a specific part of the service lifecycle [1, 2]:

- **Authentication Module:** Handles user registration, login, and role-based access, ensuring that customers, service providers, and

administrators each see only what is relevant to them.

- **Service Request Module:** Lets customers raise service requests with the necessary details, so providers know exactly what is needed before submitting a quote.
- **Bid Management Module:** Enables service providers to view open requests and submit competitive bids, giving customers the ability to compare options before making a decision.
- **Work Status Module:** Tracks each service from the moment it is accepted right through to completion, keeping customers informed at every stage.
- **Payment Management Module:** Manages the verification and release of payments, making transactions transparent and traceable for both parties.
- **Employee Management Module:** Stores and manages detailed information about service professionals, including their skills, qualifications, and performance history.
- **Attendance Module:** Records employee attendance and tracks how many service assignments each professional has completed over time.
- **Notification Module:** Delivers real-time alerts to users about new service requests, bid updates, payment confirmations, and other important events.
- **Feedback Module:** Allows customers to rate completed services and leave comments, creating an accountability loop that benefits the whole platform.
- **Administration Module:** Gives administrators a comprehensive dashboard to oversee users, employees, service activity, and platform health from a single interface.

7. ADVANTAGES OF THE PROPOSED SYSTEM

The Home Service App introduces a range of concrete improvements over traditional service booking approaches [1,2], benefiting customers, service providers, and platform administrators alike.

7.1 CENTRALIZED AND ACCESSIBLE SERVICE DISCOVERY

Rather than searching across multiple channels to find a suitable professional, customers can browse all available service categories in one place, submit a request in minutes, and begin receiving bids almost immediately [2]. This dramatically reduces the time and effort involved in initiating a service engagement.

7.2 TRANSPARENT AND COMPETITIVE PRICING

The bidding system enables customers to receive multiple quotations for the same job and compare them side by side before committing. Administrators can also set pricing ceilings for specific service categories, ensuring that neither customers nor providers are put in an unfair position. This level of price transparency is absent from traditional informal service hiring [1].

7.3 REAL-TIME PROGRESS VISIBILITY

Customers no longer need to call the service provider repeatedly to find out whether they are on their way or have completed the work. The step-by-step work status tracking gives customers live visibility into where the job stands, reducing anxiety and improving the overall service experience [4].

7.4 STRUCTURED AND VERIFIABLE PAYMENT PROCESS

payments are managed within the platform and released to the service provider only after the customer confirms that the work has been completed satisfactorily and the administrator approves the release [4, 8]. This eliminates the risk of disputes over whether a service was delivered and ensures that providers are compensated promptly once obligations are fulfilled.

7.5 ACCOUNTABILITY THROUGH RATINGS AND FEEDBACK

Every completed job generates a feedback record that is permanently associated with the service provider's profile. This creates a transparent track record that future customers can use when selecting a provider, and motivates professionals to consistently deliver quality work [1]. The accumulated ratings feed directly into the certification system, rewarding top performers with recognizable badges.

7.6 EMPLOYEE MOTIVATION THROUGH CERTIFICATION

The certificate management module goes beyond simple ratings by issuing formal recognition documents — Bronze, Silver, and Gold certificates — to employees who meet defined performance thresholds. These certificates are visible to customers at the bid comparison stage, giving higher-rated professionals a competitive advantage and encouraging a culture of continuous improvement across the workforce [1, 2].

7.7 REDUCED MANUAL EFFORT AND PAPERWORK

The entire service lifecycle — from booking to payment — is handled digitally within the application [2]. There is no need for paper-based job cards, handwritten receipts, or phone-based negotiations. This not only saves time for all parties involved but also creates a permanent, searchable audit trail for every transaction and service interaction.

7.8 MULTILINGUAL SUPPORT FOR BROADER REACH

The application supports multiple languages, making it accessible to users who may not be comfortable navigating an English-only interface. This is particularly relevant in the Indian context, where Tamil-speaking users in the Puducherry region benefit from a localized interface that lowers the barrier to adoption [9].

7.9 SCALABILITY AND FUTURE EXTENSIBILITY

Because the application is built on Firebase's [4] serverless cloud infrastructure and uses a modular Flutter [3] architecture, it can scale horizontally to accommodate a growing user base without requiring significant infrastructure changes. New service categories, additional user roles, or enhanced features such as GPS-based provider tracking, AI-driven service recommendations, and integrated payment gateways can be layered on top of the existing architecture as the platform evolves [2].

8. SCOPE

The scope of the Home Service App goes beyond what a simple booking tool would offer. It covers the complete lifecycle of a service engagement — from the initial request through to payment and feedback — while also managing employee records, attendance,

and notification flows. The modular design means that new features or service categories can be added in the future without requiring significant structural changes [2].

Being cloud-based through Firebase [4] and Cloud Firestore [8], the platform handles growing user loads as adoption increases. Future additions such as integrated payment gateways, GPS-based tracking, and AI-driven service recommendations are technically feasible within the existing framework and can be incorporated as the product evolves.



9. EFFECTS

The impact of the Home Service App is felt differently by each group involved. Customers gain faster, more reliable access to services with clear pricing they can compare upfront. Service providers benefit from improved visibility and a more consistent flow of work opportunities. Administrators have the tools they need to keep the platform running smoothly and maintain service quality across the board [1, 2].

At a broader level, the app eliminates the paper-based, informal communication that has traditionally surrounded home service arrangements. By moving everything into a structured digital workflow built on Material Design principles [9], it makes the entire process more efficient and easier to manage.

10. BENEFITS

The Home Service App delivers a range of practical benefits [1, 2]:

- Quick and straightforward access to a wide range of professional household services
- A faster, more structured process for submitting and managing service requests

- A transparent bidding system that allows customers to compare pricing before committing
- Real-time updates on the progress of active service assignments
- Improved communication through timely push notifications
- A secure and traceable payment and transaction management system
- Meaningful performance monitoring tools for service professionals
- A customer feedback and rating system that promotes service accountability
- Significant reduction in manual effort and reliance on paper-based processes
- A scalable platform architecture that supports future growth and new feature additions

11. DIFFICULTIES AND CHALLENGES FACED

Developing the Home Service App came with its share of technical and design challenges. Building a secure authentication system that could accommodate multiple user roles — each with their own permissions and views — required careful planning [4]. Keeping data synchronized in real time across customers, service professionals, and administrators added complexity, particularly when network conditions were inconsistent [8].

The bidding and payment modules demanded additional validation logic to ensure fairness and prevent errors during transactions. Getting the notification system, attendance tracking, and service status updates to work reliably across different scenarios required thorough testing [4]. Integrating Cloud Firestore [8] while maintaining stable performance across a variety of devices was another technical hurdle resolved through iterative development, consistent with best practices described by Pressman and Maxim [2].

12. CONCLUSION

The Home Service App successfully delivers on its promise of making household service management simpler and more organized. By bringing service booking, employee coordination, payment processing, and customer feedback into a single platform, the application reduces friction across the entire process

and improves how customers and service professionals interact with each other [1, 2].

This project demonstrates how mobile and cloud technologies — specifically Flutter [3], Firebase [4], and Dart [5] — can be applied to address everyday service management challenges in a practical way. The system meets its intended objectives and offers a working solution to the growing demand for on-demand, reliable home services. Several enhancements are planned for future versions, including integration of online payment gateways, GPS-based live tracking, AI-powered recommendations, in-app messaging, multi-language support, customer loyalty programs, subscription-based service plans, an advanced analytics dashboard, voice-assisted booking, and intelligent scheduling with route optimization [2]. Version control and collaboration throughout development were managed using GitHub [10]. Each of these additions will further strengthen the platform’s usefulness and scalability.



Fig:12.1 Work Status Page

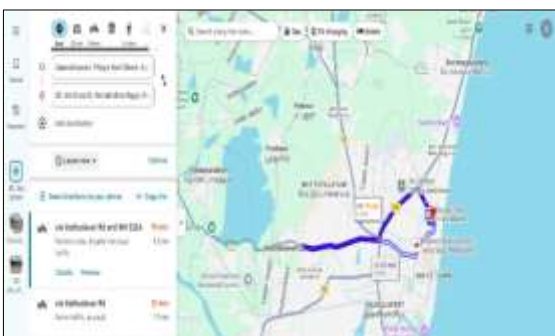


Fig:12.2 Location Page

Fig:12.3 Employee Management



REFERENCES

- [1] Ian Sommerville, Software Engineering, Pearson Education, 2016.
- [2] Roger S. Pressman and Bruce R. Maxim, Software Engineering: A Practitioner’s Approach, McGraw-Hill Education, 2020.
- [3] Flutter Official Documentation. Available at: <https://flutter.dev/docs>
- [4] Firebase Official Documentation. Available at: <https://firebase.google.com/docs>
- [5] Dart Programming Language Documentation. Available at: <https://dart.dev/guides>
- [6] Android Studio Documentation. Available at: <https://developer.android.com/studio>
- [7] Visual Studio Code Documentation. Available at: <https://code.visualstudio.com/docs>
- [8] Cloud Firestore Documentation. Available at: <https://firebase.google.com/docs/firestore>
- [9] Material Design Guidelines. Available at: <https://material.io/design>
- [10] GitHub Documentation. Available at: <https://docs.github.com>