# Design and Implementation of a Secure Mobile-Cloud Platform for On-Demand Household Services and Real-Time Provider Tracking

Pratap N. Shinde
Electronics &Telecommunication Engineering
TSSM's Bhivarabai Sawant College of Engineering & Research, Pune, India

Jayashree P. Shinde
School of Electrical & Communication Sciences
JSPM University, Pune, India

Rushabh Pawar
Department of E&TC Engg.
TSSM's Bhivarabai Sawant C.O.E & Research Pune, India

Bhagesh Jambagi
Department of E&TC Engg.
TSSM's Bhivarabai Sawant C.O.E & Research Pune, India

Madhuri Gurav
Department of E&TC Engg.
TSSM's Bhivarabai Sawant C.O.E & Research Pune, India

## ABSTRACT
This paper presents FreeServe, a full-stack web application developed to provide on-demand household services by efficiently connecting customers with skilled service professionals in their vicinity. The platform facilitates convenient access to a variety of services, including electrical, plumbing, gardening, and carpentry, through a unified digital interface. This proposed real time application enables users to browse available services, book appointments, track the real-time location of service professionals, and submit feedback on executed work. For service professionals, the platform offers tools to manage appointments and track earnings. Advanced technologies including block chain are incorporated for secure storage of customer ratings, reviews, and work history, while machine learning powers an intelligent chatbot for real-time user assistance. The overarching goal is to improve the skilled-workforce connection ecosystem, making it efficient, secure, and engaging for all stakeholders. The observed performance metrics such as application launch time, average booking confirmation time, data retrieval time, provider profile loading time and notification delivery time are optimum.

## Keywords
Household Services, Full-stack Web Application, Real-time Location Tracking, Service Bookings, Chatbot, Blockchain, Flutter

## 1. INTRODUCTION
The rapid growth of digital technology has significantly transformed the way people access day-to-day services. Modern users increasingly prefer online platforms that provide quick, reliable, and convenient solutions for household needs such as plumbing, electrical work, carpentry, cleaning, and appliance repair. To address this demand, this research paper presents the design and development of FreeServe, a user-friendly full-stack web and mobile application that connects customers with skilled service professionals through a centralized secure digital platform.

A web application enabling users to book handyman services using a technology stack comprising MongoDB, Express, Node.js, React, Redux, Axios, JWT, and Node mailer. The research presented in [1] explores features such as a user-friendly interface, location-based searching and booking, pre-defined service costs, verification of service providers, a notification and feedback system, a chatbot for user queries, and secure authentication. The service provider features [2] including payment gateways and in-app chat or call options are explored using Android Studio IDE, Java, XML, Firebase Realtime Database, Firebase Authentication, and Google Maps API. An Android application has been discussed for household services including plumbing, electrical, mechanical, pest control, home painting, and machine repairing, aimed particularly at urban areas [3][4]. The system designed using modern technologies including the Flutter framework for cross-platform application development is presented in [5], while Firebase for real-time database management and secure authentication in [6]. Machine Learning techniques for intelligent chatbot-based customer support is discussed in detail [8][9]. The application simplifies the traditional process of finding service providers by allowing users to browse nearby services, schedule appointments, and manage bookings directly through an interactive interface. This approach reduces the time, effort, and communication barriers commonly faced in conventional service-booking methods. Automated chatbot assistance helps users quickly resolve queries and obtain support without extensive human intervention [10]. Real-time location tracking is integrated using GPS, Geo location APIs, WebSockets, LeafletJS, and routing technologies [7][11][12], enabling customers to monitor worker locations during active service requests. The platform also improves operational efficiency for service professionals by providing features such as appointment management, booking history, earnings tracking, and availability control. In this paper the proposed system contributes toward the modernization of the household service industry by improving service accessibility, reducing manual effort, and creating broader employment opportunities through digital connectivity.

The proposed work presents a web application which bridges skilled working professionals with customers, offering a streamlined registration process, location-based service browsing, and one-click appointment booking. The application demonstrated thought this work is committed to creating an efficient service booking platform that allows users to browse services, select service providers, and schedule appointments easily and conveniently. By combining efficient service

booking, real-time communication, location tracking, and intelligent user assistance, the FreeServe platform aims to provide a secure, accessible, and cost-effective solution for both customers and service providers. The enhanced communication and service management between users and service providers using Firebase-based real-time data handling and secure booking management is advantageous over the past research.

## 2. SYSTEM METHODOLOGY

In the proposed system, there are three key entities. First one is customers who seek to book a service, second is service professionals, who accept appointments from customers, and third is admins, who oversee and control the system. The functionalities provided to each of these entities are discussed in detail. Customers can log in with their credentials and browse available services in their locality. They can book a service appointment by selecting the desired service category and track the worker's real-time location after the booking is confirmed. Customers can also view their complete booking history and leave feedback for completed work. Ratings and reviews submitted by customers are securely stored on the block chain, making them tamper proof. Customers can additionally interact with the chatbot to raise complaints related to active bookings or to obtain relevant information. Service professionals can activate or deactivate their profile based on their availability. After receiving an appointment request, they can view the customer's location and decide whether to accept the service. Work records and completed booking data are securely stored on the Blockchain. Workers can view their work history along with ratings and reviews received from customers and can also track their earnings from completed work. Admins have the authority to register new workers, view feedback and complaints submitted by customers, and remove workers from the system if necessary. Admins can also analyze worker performance based on customer feedback to improve service availability and quality.

Overall, the proposed system empowers customers with efficient service selection, seamless appointment booking, real-time location tracking, genuine feedback mechanisms, and interactive AI-powered assistance. Service professionals benefit from efficient appointment handling, customer location visibility, and earnings tracking.

## 3. DESIGN AND DEVELOMENT

The proposed system is designed using a modern and efficient technology stack to ensure smooth functionality, secure data handling, and better user interaction. Firebase is implemented as the backend service to manage user authentication, cloud storage, and real-time synchronization of data. It supports secure login and registration features while enabling fast communication between customers and service professionals. The backend infrastructure also helps in maintaining reliable application performance and efficient handling of booking-related information. The application interface is developed using the Flutter framework, which allows cross-platform development for both Android and iOS devices through a single codebase. Flutter provides a flexible and responsive user interface that improves usability and enhances the overall experience of the application. Firebase Firestore and real-time database are integrated with the application to securely store user records, booking details, service information, and other essential data while supporting real-time updates and efficient data access. For proper execution of the system, a smartphone or computer with a minimum of 4 GB RAM and a stable internet connection is required. Android devices or emulators are used during testing and deployment of the application. The combination of Flutter and Firebase creates a scalable and reliable environment for implementing the proposed household service management system. The system is designed using Flutter or React Native for building a highly interactive, dynamic, and responsive user interface. Machine Learning and object detection features are incorporated to provide automated solutions. Firebase is used with the Flutter database to store, utilize, and process user information efficiently. The User Module is designed to provide customers with a simple and convenient way to access home services through the application. Users can either create an account or continue as guest users to explore the platform. By specifying their location, they can view nearby available services and workers. The module allows users to compare workers based on pricing details, ratings, and availability before actual selection. Customers can schedule appointments by choosing a suitable date and time according to their convenience. In addition, users can benefit from promotional offers, discounts, and vouchers provided within the application. The module also supports real-time worker tracking, enabling customers to monitor the service provider’s status and arrival. To ensure flexibility and ease of transaction, the system provides both online and offline payment options for completing bookings securely.

The Admin Module is responsible for managing and monitoring the overall functioning of the platform. Administrators can add or remove service categories depending on customer requirements and business needs. They also have the authority to manage workers by adding new service providers or removing inactive ones from the platform. The module maintains records of successful payments to ensure proper financial management and transparency. Additionally, administrators can track worker history, performance, and service quality to maintain customer satisfaction. By analyzing customer behavior and service demand patterns, the admin can improve worker allocation and availability, thereby enhancing the efficiency and reliability of the system. The service provider module enables workers or service providers to efficiently manage their profiles and services on the platform. Providers can create and update their professional profiles, including skills, experience, and contact information. They can also add, modify, or update the services they offer along with pricing details. The module allows service providers to accept or reject booking requests based on their availability and preferences. Furthermore, providers can manage their schedules and update their availability status in real time to avoid booking conflicts. The system also gives access to customer details and booking history for better service management. In addition, service providers can track completed services and maintain records of their earnings, helping them monitor their performance and income effectively. The flowchart depicted in Fig. 1 illustrates the end-to-end booking process within the FreeServe application:

The use case diagram shown in Fig. 2 illustrates the interactions between the two primary entities first Admin and second user and the core system functions. Both entities interact with the create account module and the chatbot. The user can select a service, book a worker in an available time slot, and submit booking confirmation and feedback.

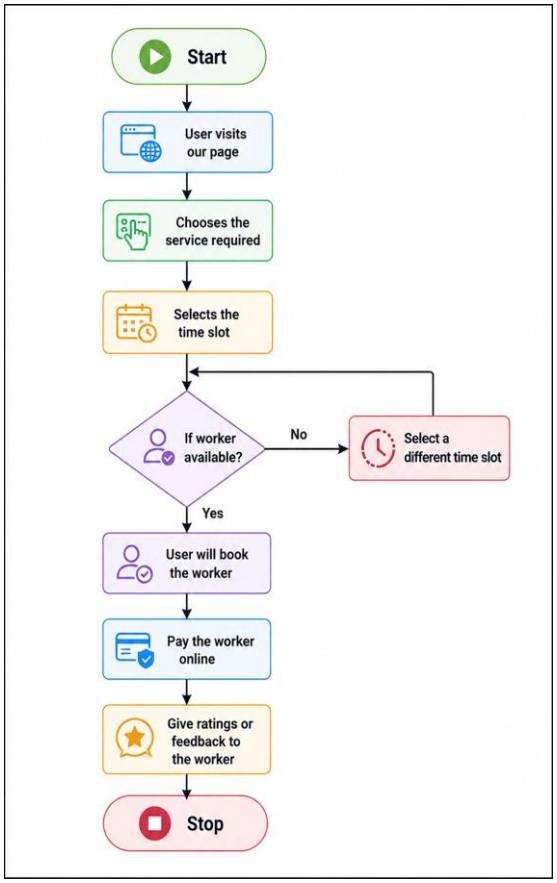


**Fig. 1: Flowchart of the FreeServe booking process**

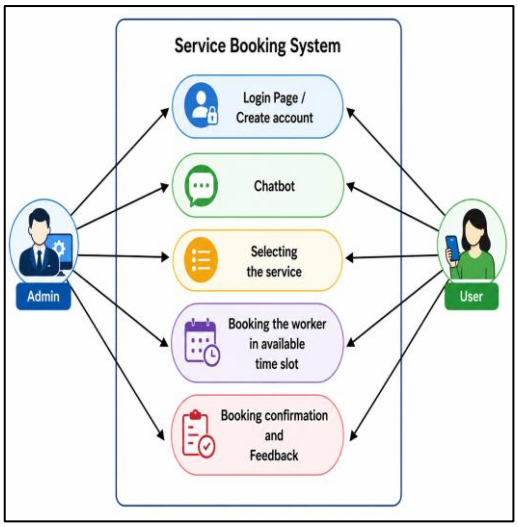


**Fig. 2: Use case diagram**

# 4. SYSTEM IMPLEMENTATION

## 4.1 Authentication Module

To ensure secure access to the platform, a robust authentication system has been implemented. Users whether customers, service professionals, or admins are required to authenticate themselves before accessing their respective dashboards. Users can either log in with existing credentials or register a new account. During the login process, users must provide their email address or phone number along with a password. For registration, additional information is required including name, email, phone number, and a desired password. Service professionals are additionally prompted to specify their default working location, service category, and the city where they primarily serve. Upon submission, all credentials are sent as a payload to the backend via an Axios POST request. In the Node.js backend, passwords are hashed before storage during registration. During login, the entered password is hashed and compared with the stored entry. If a match is found, a JWT bearer token is generated containing the user's email, role, issued time, and expiry time. This token is returned to the frontend and stored in the browser's local storage for subsequent authenticated requests. The registration profile screen of the application is depicted in Fig. 3.

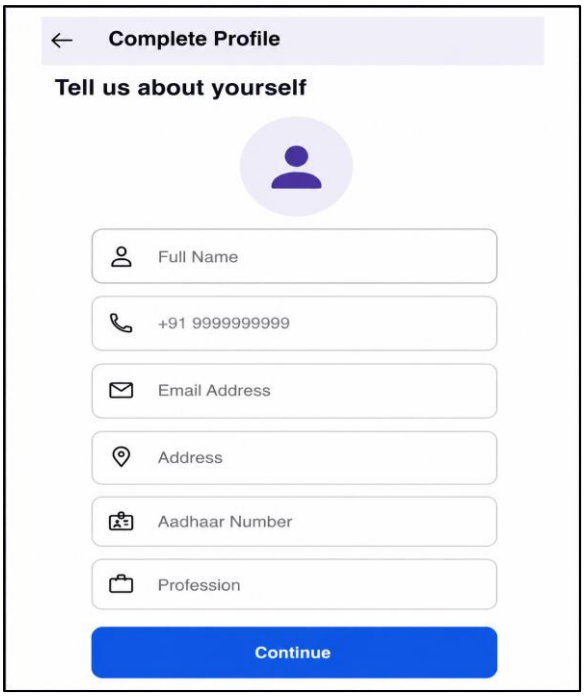


**Fig. 3: Registration profile screen**

## 4.2 Customer Module

### *4.2.1 Customer Dashboard*

After logging in and setting up their location, customers are presented with the main dashboard showing all available service categories. The listed categories are commercial shifting, office shifting, car & vehicles, cleaning, electrician, plumber, ac repair, and carpentry. Featured new services along with starting prices are also highlighted on the dashboard which is more advantageous from customer point of view. The Categories wise customer dashboard is shown in Fig. 4

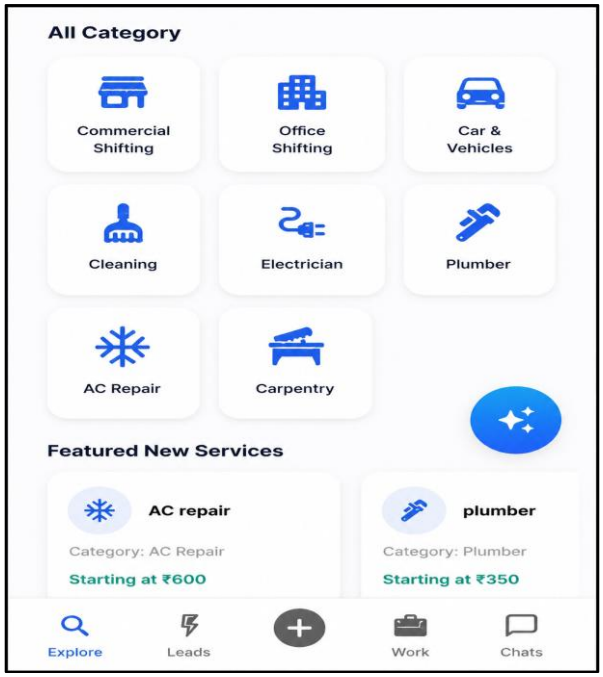


**Fig. 4: Categories wise customer dashboard**

### *4.2.2 Setting the Customer's Location*

After logging in, customers must first set their location. The browser's geolocation API is used to display a map of the customer's area. The customer places a pin at the exact location, and the corresponding latitude and longitude are retrieved and stored. The customer then enters their detailed address including house number, street, area, and pin code, which is sent to the backend via an Axios POST request.

### *4.2.3 Browsing Available Services*

Customers can browse all services available in their city along with the number of active service professionals available under each category. This data is fetched from the backend via an Axios GET request using the city name as a query parameter and the JWT token as the authorization header. The backend runs a MongoDB find query filtered by city and returns the results.

### *4.2.4 Booking a Service*

Customers book a desired service by clicking the "Book" button as shown in Fig.5. The booking screen allows the customer to select a package (Basic, Standard, or Premium), choose a date and time slot, enter the service address, and add any additional notes. An Axios POST request is made to the backend which uses the Haversine formula to match the customer with the nearest active professional. A 30-minute cooling period is enforced between successive bookings for a single professional to prevent overbooking.

### *4.2.5 Viewing Booking History*

Customers can view all past and active bookings categorized as active, inactive, or rejected/cancelled. The booking history is retrieved via a GET request to the backend. An aggregate MongoDB query matches booking IDs stored in the customer's document with the corresponding service professional's service history.

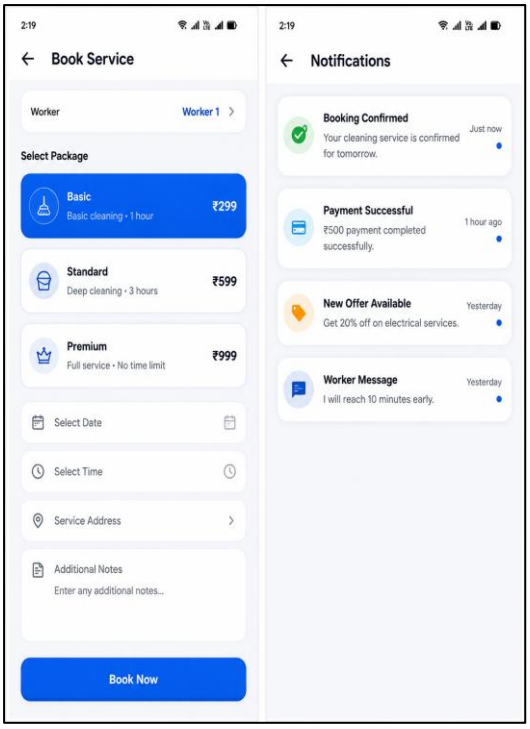


**Fig. 5 : Book service screen and notifications**

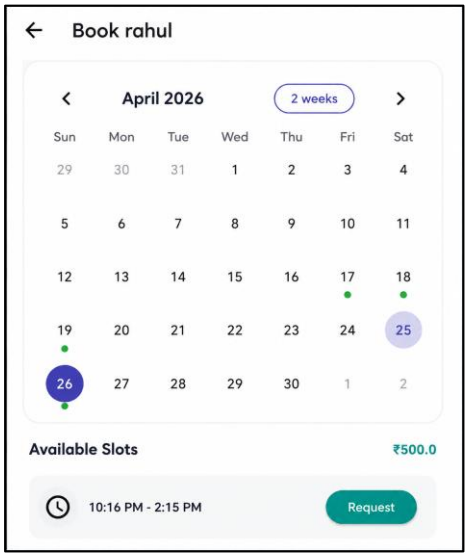


**Fig. 6: Customer active service – calendar & slot selection**

Active bookings appear at the top, with rejected bookings at the bottom. The customer active service details are reflected in Fig.6.

### *4.2.6 Managing Active Services*

The booking history section allows customers to manage active bookings. Details such as the assigned professional's name, phone number, email, and scheduled date and time are displayed. Once the service professional accepts the request, customers can track their real-time location. Customers may cancel an active service at any time. Upon service completion, a unique 6-digit OTP is generated and displayed to the customer, which must be shared with the service professional to officially mark the service as completed.

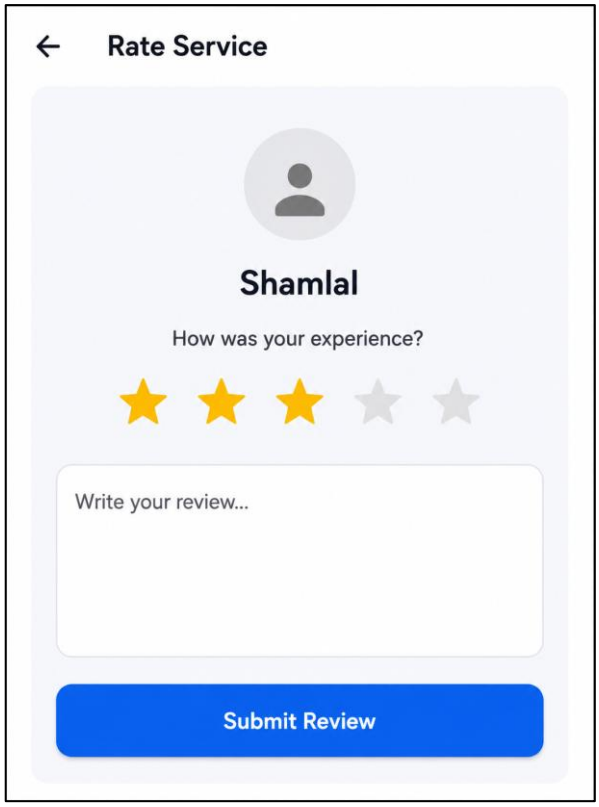


**Fig. 7: Rate service screen**

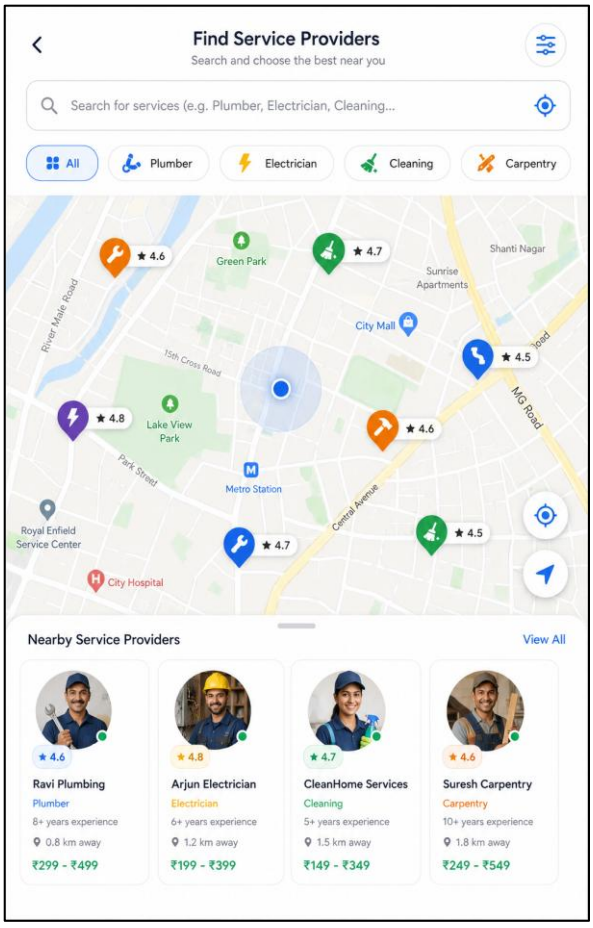


**Fig. 8: Real-Time location tracking screen**

### *4.2.7 Rating the Service*

Once the service is closed, it moves to the inactive section of the booking history. Customers can then submit a star rating, a written review, and the amount paid as shown in Fig 7. This data is sent to the backend via the update service route, which locates the booking using the booking ID and the service professional's email, and updates the corresponding database entries.

### *4.2.8 Real-Time Location Tracking Module*

When a service professional accepts a booking, real-time location tracking is enabled for both parties. The Leaflet open-source JavaScript map library is used to render maps, while the OSRM (Open Source Routing Machine) [13] C++ library calculates the shortest path and displays turn-by-turn navigation directions.

When the service professional activates the navigator, the customer's location is fetched from the database and marked as the destination on the map as depicted in Fig.8. A WebSocket connection is initiated, and a socket room is created using the service professional's email as the room identifier. The service professional's GPS coordinates are fetched every 15 seconds via the browser's Geolocation API and broadcast to all connected clients in the socket room. The customer's interface updates in real time to reflect the professional's current position along with their movement trail.

## 5. PERFORMANCE AND ANALYSIS

### 5.1 Experimental Setup

The FreeServe real time application was tested after completing the implementation of all major modules, including customer registration, service provider registration, booking management, and the admin panel. The application was developed using Flutter with Firebase services for authentication, database management, and cloud storage. Google Maps was integrated to provide location-based services and route visualization.

Testing was carried out on Android smartphones using both Wi-Fi and mobile internet connections. The objective of the testing was to verify that the application worked correctly under normal usage conditions and that all modules communicated properly through Firebase. The performance metrics is tabulated in Table 1.

**Table 1. Performance results of the application**

| Sr. No. | Performance Metric | Observed Result |
|---|---|---|
| 1 | Application Launch Time | 1.8 s |
| 2 | OTP Authentication Success Rate | 100% |
| 3 | Average Booking Confirmation Time | 2.3 s |
| 4 | Firestore Data Retrieval Time | 0.42 s |
| 5 | Provider Profile Loading Time | 0.55 s |
| 6 | Notification Delivery Time | < 2 s |
| 7 | Overall Booking Success Rate | 98% |

### 5.2 Scenario-Based Evaluation and Analysis

Instead of evaluating individual screens, the application was tested by performing complete user workflows. In user registration scenario a new customer created an account using mobile number authentication. After OTP verification, the user completed the profile and was successfully redirected to the

home page. The registration details were stored correctly in Firebase. In booking service, the customer selected a service provider, chose a preferred date and time, and submitted a booking request. The request appeared immediately in the provider's dashboard, confirming that the booking information was synchronized successfully through Cloud Firestore. In provider response scenario the service provider reviewed the booking request and accepted it. After acceptance, the booking status changed automatically on the customer's side without requiring the application to be restarted or refreshed manually. The resolution of booking is at priority and is reflected in booking history by updating booking and completed job status. This assures the booking records were maintained correctly within the database system.

The existing household service systems generally provide basic user registration, service booking, and database management, but often have limitations in security, real-time tracking, scalability, and cross-platform support. In contrast, the developed FreeServe system provides secure user registration and OTP-based authentication through Firebase, streamlined service booking with provider and schedule selection, and real-time service-provider tracking using Google Maps. It uses Cloud Firestore for scalable and real-time data synchronization. It is developed using Flutter, enabling deployment on both Android and iOS from a single codebase. Thus, the presented FreeServe real time digital system offers a more secure, intelligent, scalable, real-time, and cross-platform solution compared with existing systems.

## 10. CONCLUSION

The proposed work successfully demonstrates FreeServe, a user-centric Home Service Mobile Application that connects customers with multiple service providers through a unified digital platform. The application enables users to discover suitable services, book appointments, select convenient time slots, and manage service requests, while service providers can efficiently manage bookings, availability, and customer interactions. The integration of real-time data handling and service-provider coordination improves operational efficiency and reduces the response time for service request execution. Furthermore, the use of Flutter supports responsive cross-platform application development, while Firebase provides secure authentication, scalable backend services, and real-time data synchronization. Overall, FreeServe demonstrates the potential of integrating mobile and cloud technologies to deliver a secure, scalable, accessible, and efficient solution for on-demand household service management.

## 11. REFERENCES

[1] G. Baskaran, K. Saundariya, D. Prabakaran, and R. Senthilkumaran, “A Web Application Based Administration Panel For Handyman Services”, IEEE Delhi Section Conference, DELCON 2022, 585-589.

[2] H. S. B. Shyamala, K. Rao, P. Bhandarkar, P. P. Vetekar, and G. Laxmi, “An Android Application for Home Services”, International Journal of Engineering Research and Technology (IJERT),2020, Vol.07, No.05, 92-101.

[3] A. Adekar, A. Dalvi, P. Gharat, and P. Ratanghayra, "Household Veritas - A Platform That Provides Household Services," International Journal of Engineering Research and Technology (IJERT),2023, Vol.10, No.03, 748-753.

[4] M. Naik and S. Patil, "Online Home Service Provider System Using Android Application," International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), 2019, Vol.5, no.2, 17-23.

[5] Google Developers, "Flutter Documentation," Google, 2024, *https://flutter.dev/docs*

[6] Google Firebase, "Firebase Realtime Database and Authentication Documentation," 2024, *https://firebase.google.com/docs*

[7] A. Raj and V. Kumar, "Real-Time Location Tracking System Using GPS and WebSocket," International Journal of Advanced Research in Computer Science, 2021, Vol. 10, no.3, 112-118,

[8] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," International Conference on Learning Representations, 2013, 1-12.

[9] A. Vaswani et al., "Attention Is All You Need," 31st Conference on Neural Information Processing System (NISP 2017), 1-17.

[10] M. B. Hoy, Alexa, Siri, Cortana, and More, “An Introduction to Voice Assistants," Medical Reference Services Quarterly, 2018, Vol. 37, no. 1, 81-88.

[11] "Haversine Formula for Calculate distance, bearing and more between latitude / longitude points," Movable TypeScripts. *https://www.movable-type.co.uk/scripts/latlong.html*

[12] Leaflet JS Documentation, "Leaflet - Open Source JavaScript Library for Interactive Maps," 2024. *https://leafletjs.com.*

[13] OSRM Project, "Open Source Routing Machine Documentation,"2024. *https://project-osrm.org*