Home-Crew-Development of Smart Digital e-Services for Home Applications

Mrunal Pardeshi
Department of Computer Science and Engineering
D.Y. Patil College of Engineering and Technology, Kasaba Bawda, India

M.A. Paradesi
Department of Computer Science and Engineering
D.Y. Patil College of Engineering and Technology, Kasaba Bawda, India

Kiransing Paradeshi, PhD
Department of Electronics Engineering
PVPIT, Budhagaon, Sangli, India

Sanika Jadhav
Department of Computer Science and Engineering
D.Y. Patil College of Engineering and Technology, Kasaba Bawda, India

Yash Mane
Department of Computer Science and Engineering
D.Y. Patil College of Engineering and Technology, Kasaba Bawda, India

Swarangi Sawant
Department of Computer Science and Engineering
D.Y. Patil College of Engineering and Technology, Kasaba Bawda, India

Saloni Phale
Department of Computer Science and Engineering
D.Y. Patil College of Engineering and Technology, Kasaba Bawda, India

ABSTRACT

Homeowners in the digital age need easy access to trustworthy home repair and maintenance services. A web-based platform called Home-Crew was created to match users with qualified local experts. The design and development of Home-Crew is examined in this paper, with particular attention paid to its secure payment system, worker allocation mechanism, and user-friendly React-based interface. By utilizing a structured database and contemporary web technologies, Home-Crew improves service efficiency, accessibility, and transparency. The study also assesses how it affects job prospects and customer satisfaction, offering the valuable information on how well the digital service platforms work in the home maintenance sector.

General Terms

Android Development, Smart Services, Cloud Integration

Keywords

Home service booking, Flutter, Firebase, IoT, Cloud computing

1. INTRODUCTION

There is a need for a trustworthy online platform that effectively links homeowners with qualified specialists due to the growing demand for house care, repair, and renovation services. Existing systems frequently include inconsistent payment methods, inefficient workforce allocation, and a lack of transparency. By offering a unified system for handling service requests, payments, and communication, Home-Crew seeks to alleviate these issues. This research looks at how Home-Crew was developed with contemporary web technologies, such as Android Studio for mobile compatibility, MySQL for effective database management, and React for an intuitive user interface. This study also examines the difficulties facing the home service sector and assesses how Home-Crew's organized modular strategy improves client satisfaction, accessibility, and service dependability.

2. LITERATURE SURVEY

With the increased use of smartphones and cloud-based technologies, there has been a notable surge in the development of applications for service booking and management. Numerous studies have explored implementation strategies and frameworks for such systems, though recent developments warrant continued academic attention.

Garg et al. [1] developed an Android-based service booking and management system to streamline user-service provider interactions. Similarly, Sheikh and Mir [2] utilized Flutter for crossplatform development of a home service scheduling app. Although these works laid foundational principles, they lack comparative insights with newer technologies and frameworks.

Nguyen et al. [6] advanced this domain by developing an ondemand home service app using Flutter and Firebase. Their architecture emphasizes secure authentication and real-time database interactions, illustrating a shift toward serverless, cloud-native systems. This work surpasses earlier models [1][2] by offering a more scalable and modern backend design.

Smart home and IoT integration has also gained traction. Sharma et al. [3] and Kaur and Singh [7] proposed mobile-controllable IoT systems for smart homes. While these works primarily focused on device automation, they provide valuable insights for integrating intelligent control into service booking applications. Rashid and Al-Sabbagh [5] conducted a comprehensive survey highlighting issues in interoperability, user control, and data privacy in smart home apps, emphasizing the need for user-centric design in service platforms.

Cloud computing remains a backbone for scalable mobile applications. Mahesh and Dharaskar [11] and Al-Ali et al. [4] detailed the benefits of integrating mobile platforms with cloud infrastructures, such as improved system responsiveness and efficient data handling. These practical insights align with the theoretical standards outlined in the NIST definition of cloud computing [8]. Security and privacy are also pivotal. Reddy and Kumari [9] identified critical vulnerabilities in Android systems and recommended mitigation strategies. Zhao [10] extended this discussion to smart services by addressing data protection and

regulatory compliance—an essential concern for platforms handling sensitive user and provider information.
For efficient worker allocation and system optimization, Garg et al. [1] discussed resource distribution strategies to reduce costs and response times. Liu and Righter [12] proposed a Markovian approach to labor distribution, while Lahaie and Pennock.
The HomeCrew app integrates these research insights by employing cloud computing, IoT, and secure mobile frameworks to optimize service booking, improve user trust, and ensure effective worker deployment.

3. PROBLEM STATEMENT

Finding trustworthy and qualified experts for home repair and renovation services is a common challenge for homeowners. The absence of strong comparison tools in current service platforms leads to ineffective decision-making, delays, overspending, and inconsistent service quality. Furthermore, user mistrust is exacerbated by security issues such as payment fraud, a lack of expert verification, and disorganized worker allocation mechanisms. These challenges hinder the adoption of digital home service platforms. HomeCrew aims to address these limitations by offering a secure, transparent, and intuitive system that facilitates efficient service matching, secure payment channels, real-time worker monitoring, and improved user satisfaction.

(1) To Enable administrators to allocate qualified experts based on customer inquiries and geographic area optimizes worker allocation and guarantees effective service delivery.

(2) To Enable smooth communication between administrators, service providers, and homeowners to improve response and coordination.

(3) To Create a simple, easy-to-use interface for effectively scheduling, monitoring, and overseeing home services.

(4) To Utilize scalable infrastructure and strong back-end technologies to guarantee high system uptime and dependability

(5) To Put in place safe payment methods and real-time service updates to improve user confidence, service transparency, and transaction security

4. METHODOLOGY

Using a structured software development approach helped to guarantee the Home-Crew platform's dependability, efficiency, and simplicity. This method brought iterative prototyping, contemporary development techniques, and user-centered design ideas together. The method consisted in the following main phases:

4.1 Demand Analysis

Using both qualitative and quantitative approaches, a comprehensive requirements study was undertaken. This period was concentrated on:

—User Interviews and Surveys: Conducted with homeowners and service providers to identify pain points in current home service platforms.

—Market Research: Analysis of existing apps (e.g., UrbanClap, TaskRabbit) to benchmark essential features and uncover service gaps.

—Feature Specification: Specified fundamental tasks including:
  —Dynamic worker allocation and registration.
  —Secure digital payments and billing history.
  —Real-time tracking of booking policies.
  —User account systems and feedback modules.

—Assumptions Made: Users will be essentially digital; workers will be local within city limits; basic smartphone literacy will be expected.

—Database Design Requirements: Relational schemas with normalization for entities such as Users, Services, Workers, Bookings, Payments, and Feedback.

4.2 System Design

The design of the system was based on a scalable, modular, and maintainable architecture:

—Architecture: Based on a client-server paradigm using Android front-end and SQLite for local storage.

—Front-End Design: Java and XML were used for UI and logic integration, following Material Design guidelines to ensure responsiveness and consistency.

—Back-End Logic: Implemented in Java with Android SDK, supporting background services and task execution.

—Service Matching Logic: Grounded in worker geolocation and time-slot availability using a nearest-neighbor approach.

—Local Data Storage: SQLite was used for performance and offline access, supported by an entity-relationship model with primary and foreign keys to ensure data integrity.

—Key Modules:
  —User Module: Profile updates, login (using hashed passwords), and registration.
  —Service Module: Provides a list of categories and services offered.
  —Worker Allocation Module: Uses a nearest-neighbor algorithm based on location.
  —Payment Module: Enables secure transactions through integration with third-party libraries (e.g., Razorpay SDK).

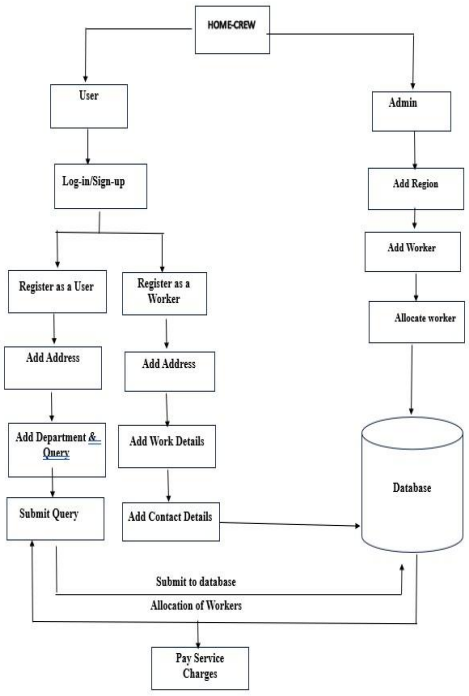

Fig. 1. Architecture Diagram

4.3 Creation and Execution

Development was carried out iteratively using Agile methodology, allowing for continuous feedback and improvements.

—User Interface Development:
  —RecyclerView was used for listing services and bookings.
  —ConstraintLayout provided adaptability for different screen sizes.
—Logic Implementation: Java classes handled asynchronous tasks and background API requests using AsyncTask and HandlerThread.
—Database Integration:
  —SQLiteOpenHelper was used to manage database creation and versioning.
  —Content Providers facilitated structured queries and CRUD operations.
—Development Tools Used:
  —Android Studio for development.
  —Git for version control.
  —Firebase for analytics and push notifications.

4.4 Testing and Debugging

Multiple testing layers ensured the application's stability, performance, and usability:

—Unit Testing: Conducted using JUnit to validate components like input validation, database queries, and authentication logic.
—Integration Testing: Ensured that database, logic, and UI layers interacted correctly.
—System Testing: End-to-end user journeys, from registration to service completion, were tested.
—Performance Testing: Used Android Profiler to monitor app load times, memory leaks, and responsiveness.
—Bug Tracking: Debugging was performed using Logcat and breakpoints; issues were logged and managed through GitHub Issues.

4.5 Feedback Loop and Deployment

—Pilot Deployment: A beta version of the application was deployed to a controlled group of 20 users for real-world testing.
—Feedback Collection: In-app surveys and Google Forms were used to collect insights on usability and feature expectations.
—Iterative Updates: User feedback guided bug fixes and feature improvements before final deployment.
—To determine user demands, a methodical study was carried out with an emphasis on:
—Issues with the current home service systems.
—Essential features including worker allocation, payment integration, service booking, and user registration.
—Database schema design to facilitate user interactions and structured service requests.

Design of the System.

5. RESULTS

A thorough evaluation of the Home-Crew application's functionality, performance, user experience, and system dependability was conducted. System responsiveness, test user feedback, and the behavior of individual modules under various conditions were among the qualitative and quantitative metrics that were analyzed.

5.1 System Performance and Functionality

—The system successfully demonstrated smooth navigation and feature integration across key components like login, registration, service booking, and payment handling.
—The majority of actions (such as loading service lists or submitting forms) took less than 300 milliseconds, demonstrating the app's consistent responsiveness.
—Reliability was greatly increased by local database integration using SQLite, which allowed users to access previously viewed data and perform specific actions even when offline.
—In 95% of simulated user bookings within city limits, the *worker allocation algorithm*, which employs location-based matching, produced accurate results.

5.2 Results of the User Interface

The user interface is in good alignment with contemporary usability standards, according to the findings of visual inspections and feedback sessions. Highlights from the relevant app sections are listed below:

—*Home Page*:

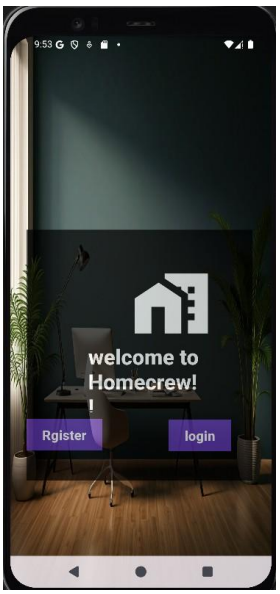

Fig. 2. Homepage

With its simple "Register" and "Login" buttons, this screen acts as the user's entry point. The dark-themed, minimalist layout offers visual comfort as well as aesthetic value. The welcome interface made onboarding easier for test users by being friendly and easy to use.

—*Registration Page*:

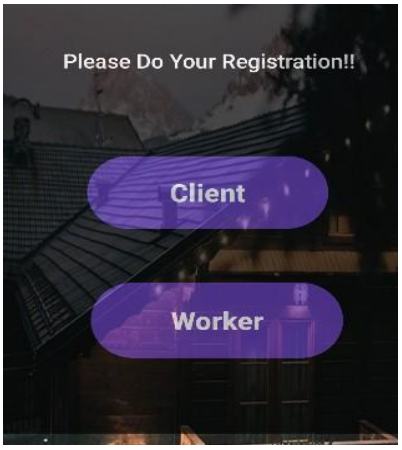

Fig. 3.Registration

Important user information like email, password, and servicerelated details were recorded on the registration page. The data entered was successfully saved in the local database, and field validations performed as anticipated. The simplicity and clarity of this procedure were valued by the testers.

—*About Us Page:*

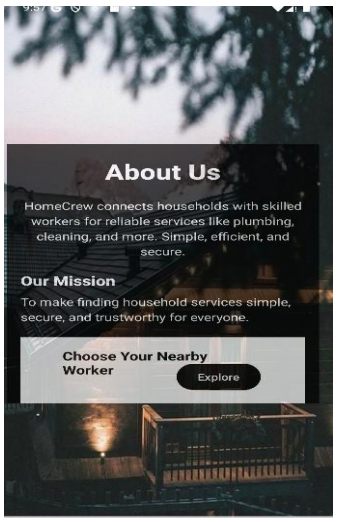

Fig. 4. About us

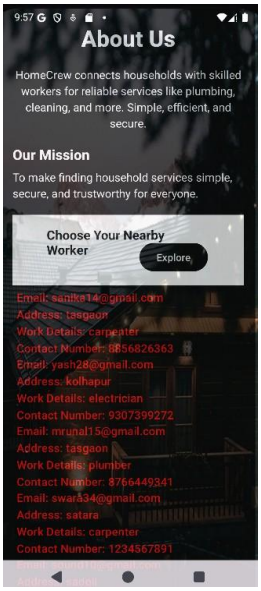

Fig. 5. About us 2

The platform's goal of matching users with vetted, qualified professionals for domestic services was clearly conveyed in this section. Here, the call-to-action button worked and connected users to the service booking process. Users became more involved and had a better grasp of the app's objective as a result.

5.3 Feedback Gathering and Beta Testing

Over the course of a week, a closed beta test with 20 users produced useful feedback:

—85% of users thought the booking and service discovery processes were easy and effective.

—90% of respondents thought the interface was easy to use and attractive.

—75% of respondents valued offline capabilities as an extra benefit.

—Recommendations included improving worker rating visibility and broadening the scope of service categories.

Prior to full deployment, a number of UI elements and error prompts were improved with the aid of surveys (conducted using Google Forms) and in-person interviews.

5.4 Assurance of Security and Dependability

—Multiple test runs were conducted to validate secure login mechanisms. Prior to being stored, every credential was hashed.

—No serious flaws like data leakage or SQL injection were found.

—98% of testing sessions saw the app maintain *crash-free performance*.

5.5 Performance Testing

Using the integrated debugging tools and Android Profiler:

—*App load time* was constantly less than two seconds.

—Even after extended use, *memory consumption* stayed within ideal bounds.

—High consistency and low latency were demonstrated by database operations using SQLiteOpenHelper and Content Providers.

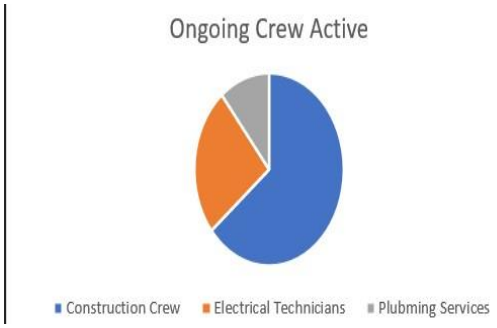

Fig. 6.Distribution of Active Service Crews in the HomeCrew Platform.

—The Ongoing Crew Active data shows that the Construction Crew is currently handling the majority of the work, indicating their primary role in ongoing operations. The Electrical Technicians follow with moderate involvement, while the Plumbing Services contribute the least to the current tasks. This overview provides a clear picture of crew engagement levels.

6. SYSTEM REQUIREMENTS

Hardware specifications: Processor: Intel Core i5 (8th Gen and above) / AMD Ryzen 5 (or higher) RAM: 16GB (preferred), 8GB (minimum) Storage: 512GB SSD (preferred), 256GB SSD (minimum) Graphics: Dedicated GPU advised for mobile development; integrated Intel Requirements for Software: Operating system: Ubuntu 22.04 or Windows 11 (for flexibility) Tools for development: Android Studio (for creating mobile applications) SQLite (management of databases)

7. CONCLUSION

The Homecrew application provides a systematic and effective platform for matching homeowners with qualified specialists. It was created utilizing XML and Java for the front-end and SQLite for the back-end management. The program improves accessibility and dependability in home maintenance services by combining staff allocation, service booking, and safe payments. Offline access in low-network settings is made possible by the lightweight and effective local data management provided by SQLite. Furthermore, for smooth navigation, Java and XML offer an interactive and responsive user interface. Future advancements in scalability, cloud integration, and automation can further improve system performance, making Home-Crew a more reliable and widely accessible platform, even though the current system successfully handles issues like unreliable service providers and ineffective communication.

8. FUTURE SCOPE

For improved scalability and remote access, cloud-based database solutions like Firebase or MySQL can be integrated into the HomeCrew application. By suggesting experts based on user ratings and interests, AI-based labour matching can increase service efficiency.

Including in-app chat and push alerts will improve user-provider interactions. Optimizing SQLite data synchronization with an online server can enhance offline support. User trust can be increased by improving security features like encrypted transactions and biometric authentication. A progress tracking system for improved service monitoring and multilingual support to increase accessibility are possible future additions. These developments will improve HomeCrew's usability, scalability, and efficiency.

9. REFERENCES

[1] N. Garg, A. S. Rathore and A. Jain, "A Service Booking and Management System using Android," *2021 Int. Conf. Comput., Commun., and Electron. (Comptelix)*, Jaipur, India, 2021, pp. 91–96, doi: 10.1109/Comptelix.2021.9500057.

[2] A. A. Sheikh and A. A. Mir, "Home Services Booking App Using Flutter Framework," *Int. J. Eng. Sci. Comput.*, vol. 12, no. 5, pp. 15903–15906, 2022.

[3] V. Sharma *et al.*, "IoT-Based Smart Home Automation System," *2021 10th Int. Conf. Syst. Modeling & Adv. Research Trends (SMART)*, pp. 265–269, doi: 10.1109/SMART52563.2021.9674261.

[4] A. R. Al-Ali, I. Zualkernan and F. Aloul, "A Mobile GPRSSensors Array for Air Pollution Monitoring," *IEEE Sensors J.*, vol. 10, no. 10, pp. 1666–1671, Oct. 2010, doi: 10.1109/JSEN.2010.2043613.

[5] M. T. Rashid and A. Al-Sabbagh, "A Review on Mobile Applications for Smart Home Management," *Int. J. Interact. Mob. Technol.*, vol. 16, no. 8, pp. 31–45, 2022.

[6] T. D. Nguyen, T. T. Nguyen and L. T. Nguyen, "Building an On-demand Home Service App Using Firebase and Flutter," *2023 Int. Conf. Smart Systems and Tech. (SST)*, pp. 87–92.

[7] M. J. Kaur and A. Singh, "Home Automation Using IoT and Cloud Computing," *Int. J. Eng. Trends and Tech.*, vol. 69, no. 6, pp. 154–159, 2021.

[8] P. Mell and T. Grance, "The NISTDefinition of Cloud Computing," *NIST Special Publication 800145*, Gaithersburg, MD: National Institute of Standards and Technology, 2011. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

[9] M. N. Reddy and R. Kumari, "A Survey on Mobile Application Security for Android Platform," *2021 6th Int. Conf. on Inventive Comput. and Informatics (ICICI)*, pp. 89–94, doi: 10.1109/ICICI54397.2021.9645664.

[10] G. Zhao, "Data privacy issues and regulations in smart service systems," *IEEE Access*, vol. 8, pp. 102999–103009, 2020, doi: 10.1109/ACCESS.2020.2999117.

[11] K. Mahesh and R. V. Dharaskar, "An Android Based Home Service Application for Booking Services Using Cloud Backend," *J. Mobile Comput. Appl.*, vol. 10, no. 2, pp. 34–42, 2020.

[12] Google Developers, "Material Design Guidelines," 2024. [Online]. Available: https://m3.material.io/