**HelperConnect: Real-Time Assistance for Customers in Need**

**Table of Contents**

[Introduction 3](#_Toc188627736)

[App Overview 3](#_Toc188627737)

[Technical Architecture 4](#_Toc188627738)

[User Authentication 6](#_Toc188627739)

[Location Tracking and Google Maps Integration 7](#_Toc188627740)

[Data Management 7](#_Toc188627741)

[Networking 8](#_Toc188627742)

[Core Android Concepts 9](#_Toc188627743)

[Activities and Intents 9](#_Toc188627744)

[ViewModels and LiveData 9](#_Toc188627745)

[RecyclerView 10](#_Toc188627746)

[Permissions 10](#_Toc188627747)

[Multithreading 10](#_Toc188627748)

[User Interface and User Experience 11](#_Toc188627749)

[UI Design 11](#_Toc188627750)

[User Experience 11](#_Toc188627751)

[Feedback and Error Handling 12](#_Toc188627752)

[Testing and Debugging 12](#_Toc188627753)

[Challenges and Solutions 12](#_Toc188627754)

[Conclusion 13](#_Toc188627755)

[References 15](#_Toc188627756)

# Introduction

Help Assistance App is an app targeted at providing customers with help they request with the assistance of the helpers in real-time. The app solves the problem of how to quickly obtain trustworthy help in the moments when you have an urgent need in a handyman or a delivery person. Developed with the use of Android Studio, Java, Firebase for real-time data managing, and Google Maps API in order to help the customers and the helpers to interact comfortably. Specifically in the contemporary world where everything is fast moving, this application proves to be most convenient in aiding the helpless get assistance and the helper get jobs within the shortest time.

# App Overview

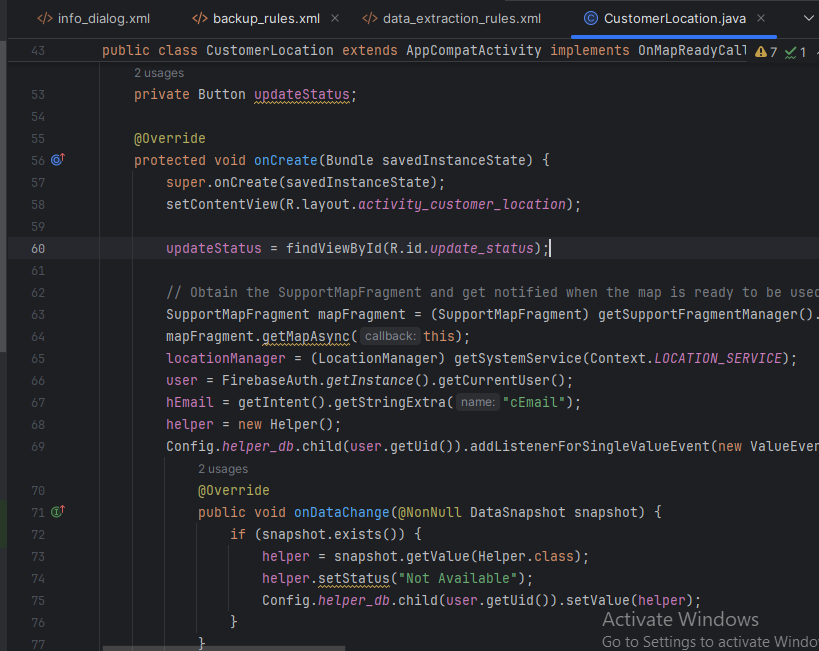
Help Assistance App is a tool that helps to narrow the communication gap between the customer and the helper, therefore, matching them. Customers can use the application to call for assistance on specific chores like delivery, repairs, cleaning among other services; helpers can accept requests depending on the location and time of order (Boppiniti, 2021). There are two primary user roles in the app: customers who require help, and helpers who offer their services.

The app features several screens for different user interactions:

**Login/Signup Screen**: Users can register or log in using their email and password. They can select their role (customer or helper) and proceed accordingly.

**Customer Map Screen**: Customers can view their current location and find nearby helpers. They can also track helpers' real-time locations and estimated arrival times.

**Helper Dashboard**: Helpers can manage their availability, view customer requests, and navigate to the customer’s location.



**Figure 1: CustomerLocation.java Code**

(Source: Self-Created)

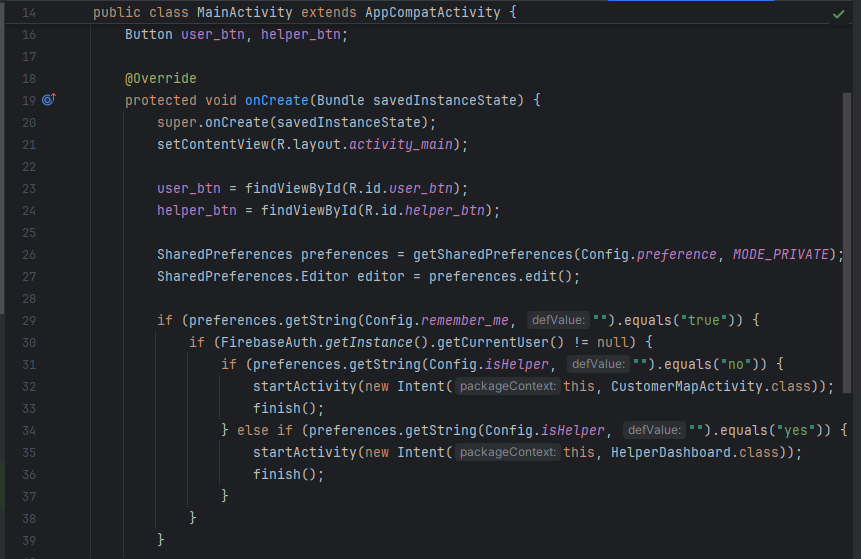
**Helper Location and Customer Location**: These screens show real-time locations using Google Maps API.

# Technical Architecture

The Help Assistance App is structured into various packages to ensure clear organization and maintainability. The primary components include:

**Activities**: These are individual screens within the app, such as the login, signup, customer map, helper dashboard, and location screens.

**Models**: This includes data models for Customer and Helper, which store attributes like email, name, contact information, and location data (latitude and longitude).



**Figure 2: MainActivity.java Code**

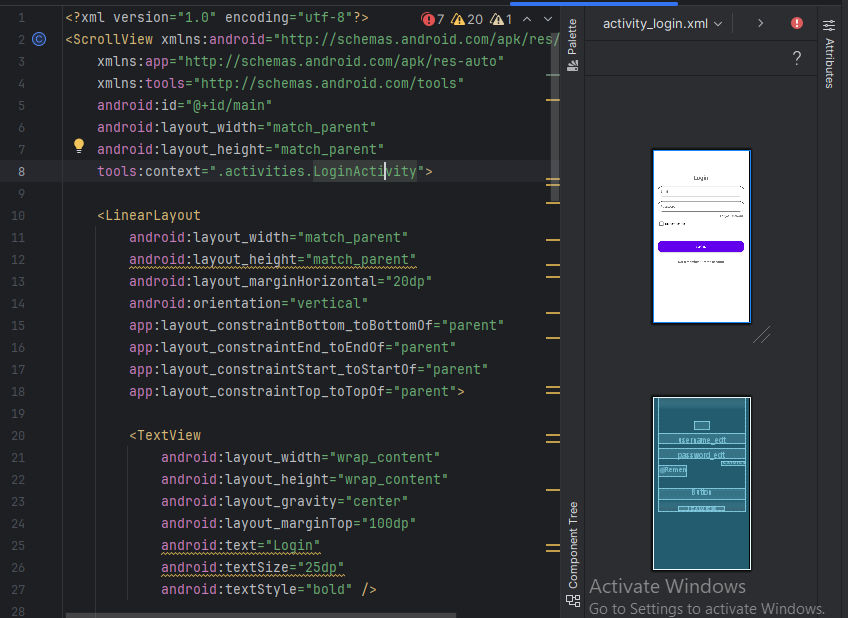
(Source: Self-Created)

**Helpers**: These are utility classes and configurations that support app functions, such as constants and utility functions for Firebase and Google Maps integration.

Key components of the app’s architecture are:

**Customer and Helper Data Models**: These models represent the data structures for each type of user. They include essential attributes such as the user’s name, email, contact number, and current location. The Helper model also includes additional fields for specialty and status (e.g., available or unavailable).

**Firebase Database Integration**: Firebase is used to store all user-related data, including customer and helper information. The data is stored in Firebase Realtime Database, and the app listens for changes to ensure real-time updates.



**Figure 3: activity\_login.xml Code**

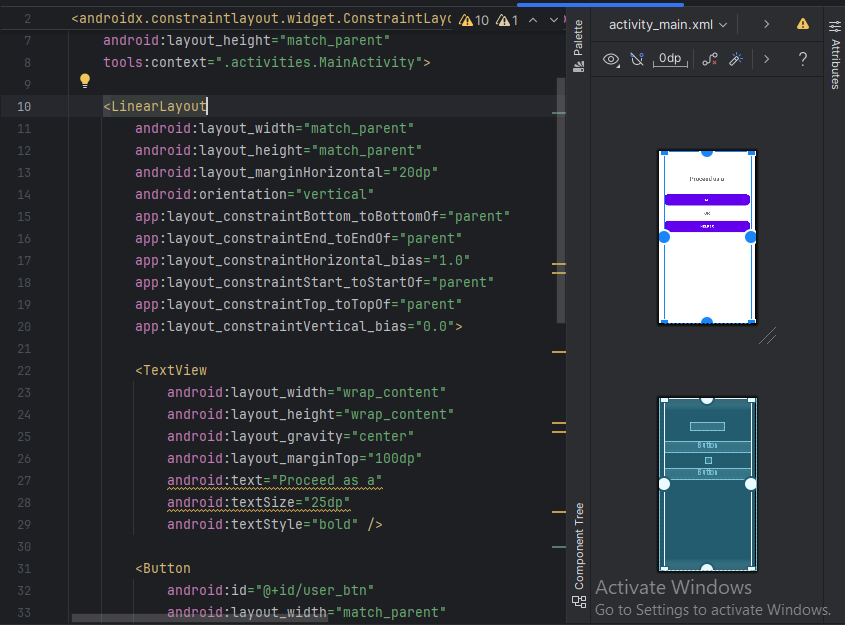
(Source: Self-Created)

## User Authentication

Firebase Authentication manages the sign-up and login process. The app supports email/password authentication, allowing users to create an account or log in securely. Upon successful authentication, the app checks the user's role (customer or helper) and redirects them accordingly to the relevant screen.

## Location Tracking and Google Maps Integration

Google Maps API is integrated into the app to track the location of customers and helpers in real-time (Kim, Choi, Kim, & Lee, 2021). The app uses LocationManager to fetch the user's location and updates it continuously. Markers are displayed on the map to show the users' positions, and estimated arrival times are calculated based on the current location.

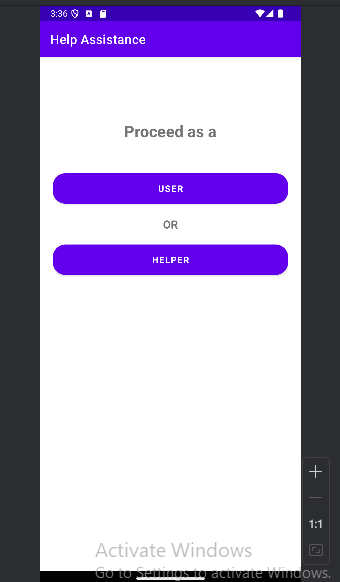


**Figure 4: activity\_main.xml Code**

(Source: Self-Created)

## Data Management

Data for both customers and helpers is stored in Firebase Realtime Database. Customer and helper data is categorized into separate nodes for quick access. Whenever a user updates their location or status (e.g., a helper marks themselves as available), Firebase listeners ensure these changes are immediately reflected across the app for all users.



**Figure 5: Home Screen Page**

(Source: Self-Created)

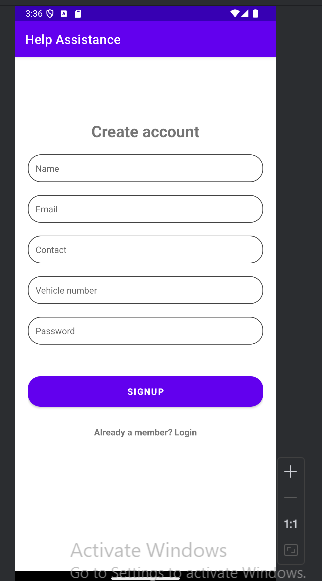
## Networking

Networking in the app is primarily handled by Firebase, ensuring seamless communication between customers and helpers. The app fetches real-time updates regarding users’ locations, statuses, and other relevant information. When a user updates their status (e.g., a helper becomes available), Firebase pushes these updates to other users, ensuring everyone has the most current information.

# Core Android Concepts

## Activities and Intents

In the Help Assistance App, Activities are used to manage the different screens of the app. For example, the LoginActivity handles the user authentication, while the MapActivity displays the customer’s map and helper’s location in real-time (Chougale, Yadav, Gaikwad, & Vidyapeeth, 2021). Each screen in the app is represented by an individual activity, ensuring that users have a clear navigation path. Intents are used to pass data between activities.



**Figure 6: User Signup Page**

(Source: Self-Created)

## ViewModels and LiveData

ViewModels and LiveData are utilized to manage UI-related data in a lifecycle-conscious manner. For instance, the LocationViewModel tracks the user’s current location, which is updated in real-time. The LiveData object holds this location data and automatically updates the UI (such as the map view or status) without worrying about lifecycle issues (e.g., configuration changes).

## RecyclerView

RecyclerView is employed to display dynamic lists in the app, such as a list of available helpers or customer requests. When a customer requests assistance, helpers who are marked as available are shown in a scrollable list on the HelperDashboardActivity. This allows for an efficient display of multiple data items with support for pagination and dynamic updates.

## Permissions

The app requires specific permissions such as location access and internet connectivity to function effectively. Location permissions are requested at runtime using the Permissions API. If the app needs to get the user’s location, a prompt shows up, asking permission (Shelke, Patil, Pinjari, & Budaragade, 2024). Much like that, you also need internet access permissions for network calls that can be things like getting data from Firebase or Google Maps.

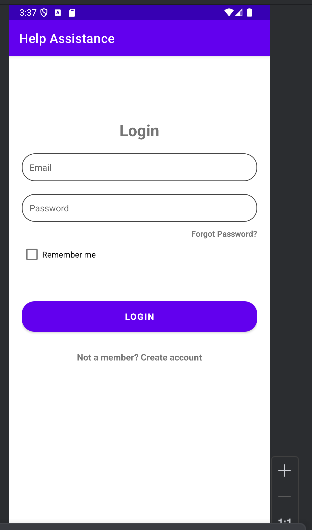
## Multithreading

Handling long running tasks visually with in the app requires multithreading. Kotlin Coroutines are used for tasks like retrieving location update or another network call to Firebase. This means Coroutines enable an asynchronous operations that ensures good UI interactivity while tasks like fetching the location or syncing data run persistently in the background.

# User Interface and User Experience

## UI Design

The Help Assistance App follows Material Design principles with clean simple and intuitive interface. Screen design using flexible and responsive layout provided to various devices with ConstraintLayout to match UI widget size (Nurdin, Pettalongi, & Mangasing, 2021). The login and signup screens have large, easily tappable buttons, whereas on the map screen we use interactive components such as MapView with clear user locations showing with the relevant map marks.



**Figure 7: Helper Login Page**

(Source: Self-Created)

## User Experience

The app provides a smooth and intuitive user experience by ensuring a seamless flow between screens. The navigation is straightforward, with clear transitions between the login/signup screens and the main map interface. Once logged in, users are directed to the map screen, where they can immediately view available helpers or make a request for assistance.

## Feedback and Error Handling

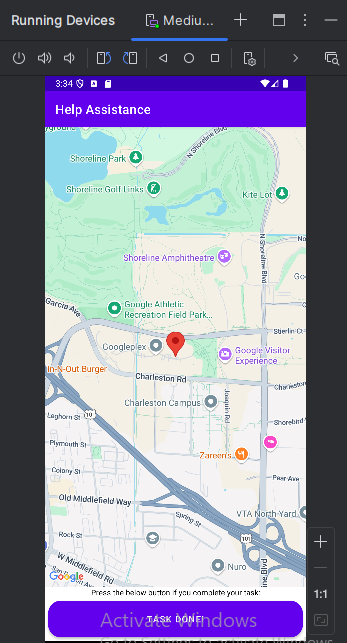
The app incorporates error handling and feedback mechanisms to ensure users are informed of their actions. For instance, if a user enters incorrect login credentials, a Toast message will appear with an error message (Eryonucu & Papadimitratos, 2022). Input validation is implemented during signup and login processes, where invalid data (e.g., missing email or password) is flagged, and the user is prompted to correct it.

# Testing and Debugging

The development team implemented unit tests, UI tests, and instrumented tests to ensure the app’s functionality. Unit tests were used to validate the logic behind user authentication and data processing, ensuring that the sign-up and login flows worked correctly. UI tests focused on testing the user interface elements like buttons and form submissions. and the app. The team used Logcat for debugging, to review real time logs to understand the reasons behind app runtime behavior issues which can also be termed as crashes of the app. To track user interactions and catch errors in the cloud database we also used Firebase Analytics.

# Challenges and Solutions

During development of the app there were some challenges. But the biggest hindrance was Firebase: I struggled to integrate it for real time syncing because you really had to put thought into the structure of the database and how the app communicates with Firebase (Bossard, 2023). The second challenge is to handle location permissions; we have to track users’ live locations, so we need to provide dynamic runtime permissions.



**Figure 8: User Location Map Page**

(Source: Self-Created)

To overcome these challenges, the team made extensive use of Firebase documentation and tutorials to correctly configure the database and ensure data integrity.

# Conclusion

Finally, the app resolves the real world problem of connecting customers to helpers when it is bad. It enables practical and sufficient real time location tracking and immediate assistance requests for users. App’s easy to use and smooth interaction help it be a handy instrument for folks that require quick help. The future enhancements include the addition of in app chat between customers and helpers, adding artificial intelligence based recommendations to match helpers with tasks and adding support for different regions with advanced geolocation functionality to track their locations more precisely.

# References

Boppiniti, S. T. (2021). Real-time data analytics with ai: Leveraging stream processing for dynamic decision support. *International Journal of Management Education for Sustainable Development*, *4*(4). Available At: <https://ijsdcs.com/index.php/IJMESD/article/download/589/227> [Accessed On: 21.01.25]

Bossard, A. (2023). On Modern User Interface Development and the Case of Android Applications. *J. Softw.*, *18*(4), 209-217. Available At: <https://www.jsoftware.us/vol18/JSW-V18N4-491.pdf> [Accessed On: 22.01.25]

Chougale, P., Yadav, V., Gaikwad, A., & Vidyapeeth, B. (2021). Firebase-overview and usage. *International Research Journal of Modernization in Engineering Technology and Science*, *3*(12), 1178-1183.Available At: <https://www.researchgate.net/profile/Anil-Gaikwad-12/publication/362539877_FIREBASE_-OVERVIEW_AND_USAGE/links/62efc738505511283e9a5318/FIREBASE-OVERVIEW-AND-USAGE.pdf> [Accessed On: 22.01.25]

Eryonucu, C., & Papadimitratos, P. (2022, May). Sybil-based attacks on google maps or how to forge the image of city life. In *Proceedings of the 15th ACM Conference on Security and Privacy in Wireless and Mobile Networks* (pp. 73-84). Available At: <https://dl.acm.org/doi/pdf/10.1145/3507657.3528538> [Accessed On: 22.01.25]

Kim, H. J., Choi, M. H., Kim, M. H., & Lee, S. (2021). Development of an ethernet-based heuristic time-sensitive networking scheduling algorithm for real-time in-vehicle data transmission. *Electronics*, *10*(2), 157. Available At: <https://www.mdpi.com/2079-9292/10/2/157/pdf> [Accessed On: 22.01.25]

Nurdin, N., Pettalongi, S. S., & Mangasing, M. (2021). Implementation of Geographic Information System Base On Google Maps API to Determine Bidikmisi Scholarship Recipient Distribution in Central Sulawesi Indonesia. Journal of Humanities and Social Sciences Studies, 3(12), 38-53. Available At: <https://www.al-kindipublisher.com/index.php/jhsss/article/download/2575/2264> [Accessed On: 22.01.25]

Shelke, A., Patil, K., Pinjari, S., & Budaragade, A. P. (2024). Connecting Communities: An Android Social Networking Application with Firebase & Java. Available At: <https://www.researchgate.net/profile/Avinash-Budaragade/publication/381190663_Connecting_Communities_An_Android_Social_Networking_Application_with_Firebase_Java/links/666128e4b769e769191acb72/Connecting-Communities-An-Android-Social-Networking-Application-with-Firebase-Java.pdf> [Accessed On: 22.01.25]