

Technical Project Report - Android Module

AWAV

Subject: Universidade de Aveiro - Computação Móvel

Date: 7/04/2025

Students: 114588: Henrique Teixeira
114614: Martim Santos

Project abstract: An app that facilitates festivals and overall events, by encapsulating all money transactions with NFC, ticket purchasing and historical data, and improves the users experience by having a proximity-based chat for sharing messaging and photos of what's happening in the event and by providing smart spending insights, allowing the user to track and analyse expenses

Report contents:

[1 Application concept](#)

[Overview](#)

[Essential journey map](#)

[2 Implemented solution](#)

[Architecture overview \(technical design\)](#)

[Implemented interactions](#)

[Project Limitations](#)

[New features & changes after the project presentation](#)

[3 Conclusions and supporting resources](#)

[Lessons learned](#)

[Work distribution within the team](#)

[Reference materials](#)

[Project resources](#)

1 Application concept

Overview

→ what is this app for? Who are the typical users? How would they benefit from using the app in their lives?

This app provides 'A Way of Amplifying Vibes (AWAV)' for festival goers, organizers and sellers. It intends to simply and enhance festival experiences for all parties involved.

There are 3 interfaces on the app, the participant, sales worker and organizer.

The **participant** has 5 main screens:

- **Home:** A simple screen where the user, if is yet to acquire a ticket only has the option to buy one. If he has a ticket, this is where he will have access to this wallet and methods of payment
- **Profile:** This is the only other screen the user has access to if he is yet to acquire a ticket. Here, he has the option to edit his name and profile picture, as well as visualizing what tickets he has, his spends and logout.
- **Stands:** The interface where the user can see the menus and order items from the stands present at the event
- **Feed:** The social part of the app is located here. This acts as a way of letting the participants interact with each other, letting them upload texts and images in real time about what is going on in the event.
- **Schedule:** Where the participants can see the schedule of when and where everything is going to happen in the event.

The **Stand Worker** has 3 main screens:

- **Manage Stand:** The interface used to add and manage the products the stand has
- **Sales analytics:** A way for workers to see a number of useful statistics about how the stand sales are going.
- **Profile:** The same as referred above

The **Organizer** has 4 main screens:

- **Manage Events:** Where the organizer can manage the event schedule, the stands and the event details.
- **Create event:** Where new events are created
- **Users:** The way of giving accounts the Stand Worker role
- **Profile:** The same as referred above

Essential journey map

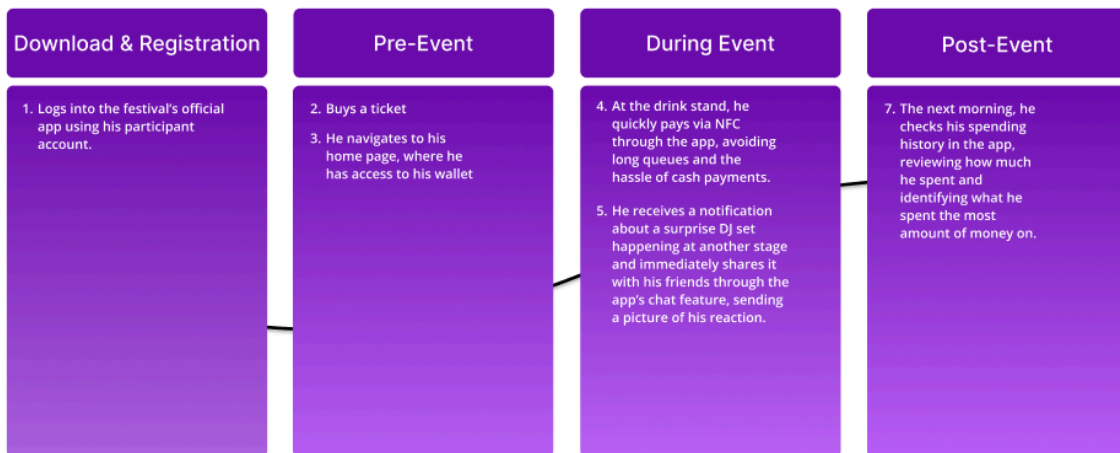
Participant Journey Map



Tiago Pedrosa

Goals

- Make seamless cashless payments.
- Get event updates and promotions.
- Stay connected with attendees.
- Track expenses throughout the event.



Opportunities

- Implement personalized recommendations based on spending and event interests
- Enhance social features, such as event-specific chat groups or live reactions.
- Provide real-time spending alerts to help users manage their budget better.
- Offer in-app pre-orders for food and drinks to further reduce waiting times.

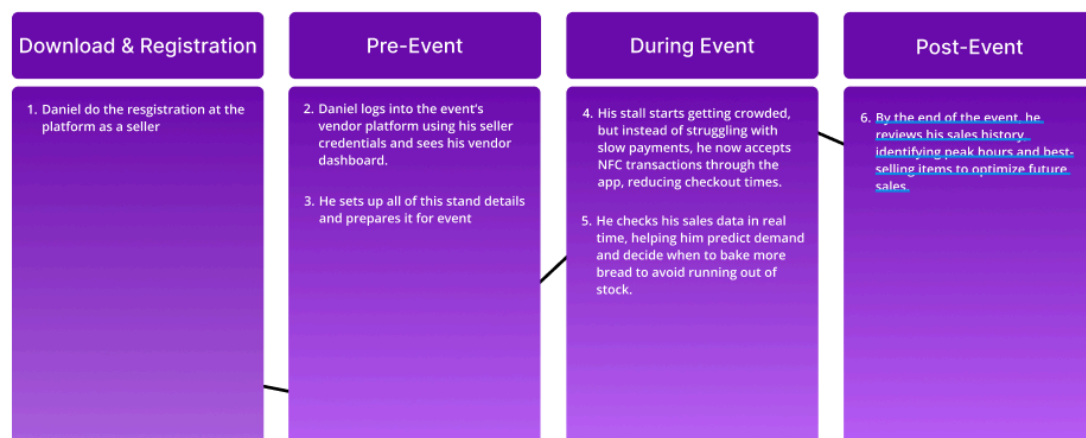
Sales Worker Journey Map



Daniel Ferreira

Goals

- Increase sales by enabling cashless transactions.
- Monitor best-selling products and adjust stock accordingly.
- Improve customer experience with quick and easy payments.
- Reduce time spent by clients at queues.



Opportunities

- Connect the vendor dashboard with logistics or suppliers to optimize restocking
- AI to recommend the best baking or restocking times based on sales trends.
- Introducing automatic alerts when stock levels are low.
- Implementing automated discount systems for loyal customers or dynamic promotions.

Sales Worker Journey Map



2 Implemented solution

Architecture overview (technical design)

The AWAV application follows the MVVM (Model-View-ViewModel) architecture pattern as recommended by Android Architecture Guidelines. The app is organized into distinct layers, each with specific responsibilities:

1. UI Layer (View): Jetpack Compose screens and components
2. ViewModel Layer: Manages UI state and business logic
3. Repository Layer: Provides data access and abstracts data sources
4. Data Layer: Room database and Firebase integrations

Key Components

UI Layer

- Uses Jetpack Compose for declarative UI
- Implements navigation through a dedicated navigation component
- UI components are organized by feature in the screens directory

ViewModel Layer

- ViewModels (such as TicketViewModel, EventDetailsViewModel) manage UI state using StateFlow
- Implements Android's ViewModel architecture component
- Provides factory patterns for dependency injection

Repository Layer

- Abstracts data sources through repository interfaces and implementations
- Examples: FirebaseFeedRepository, EventsRepository, AuthRepository
- Handles synchronization between local and remote data sources
- Manages offline/online transitions

Data Layer

- Local Storage: Room database (AppDatabase) for offline persistence
- Remote Storage: Firebase Firestore for cloud storage
- Model Classes: Data entities for both Room and Firebase

Data Models & Persistence Strategy

Firebase Integration

- Firebase Firestore for remote data storage
- Firebase Authentication for user management
- Firebase Storage for image storage
- Firebase Cloud Messaging for push notifications
- Firebase Functions for serverless operations

Data Synchronization

- Two-way synchronization between Room and Firebase
- Offline-first approach with local caching
- Real-time updates using Firebase listeners
- Background synchronization for data integrity

Advanced Design Strategies

Background Processing

- Coroutines for asynchronous operations
- Firebase Functions for server-side processing
- Background synchronization of data

Security Features

- Firebase Authentication for secure login
- Firebase App Check for preventing unauthorized API usage
- Data validation at multiple layers

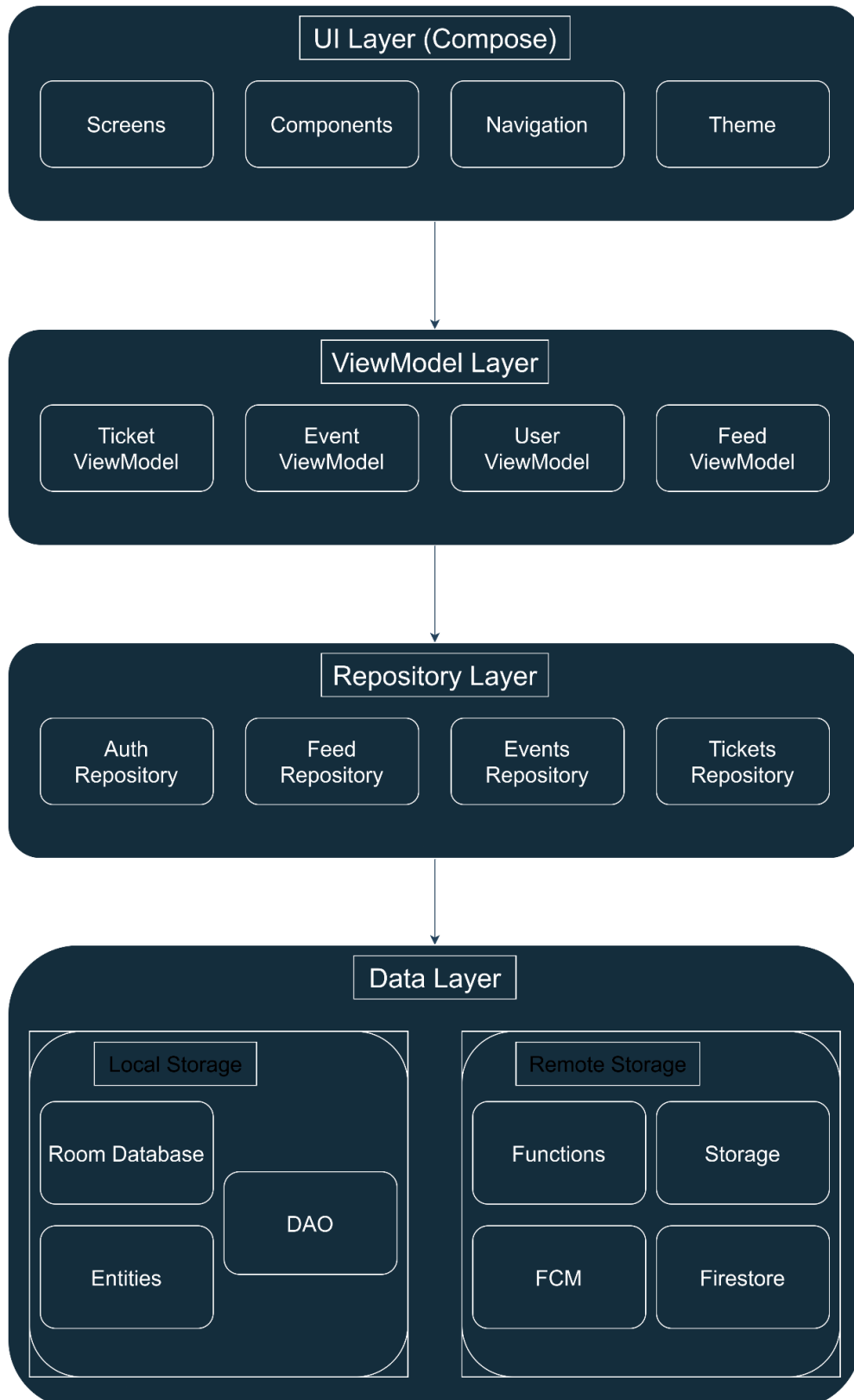
Offline Capability

- Room database for offline storage
- Synchronization strategies when coming back online

- Resilient error handling for network issues

Push Notifications

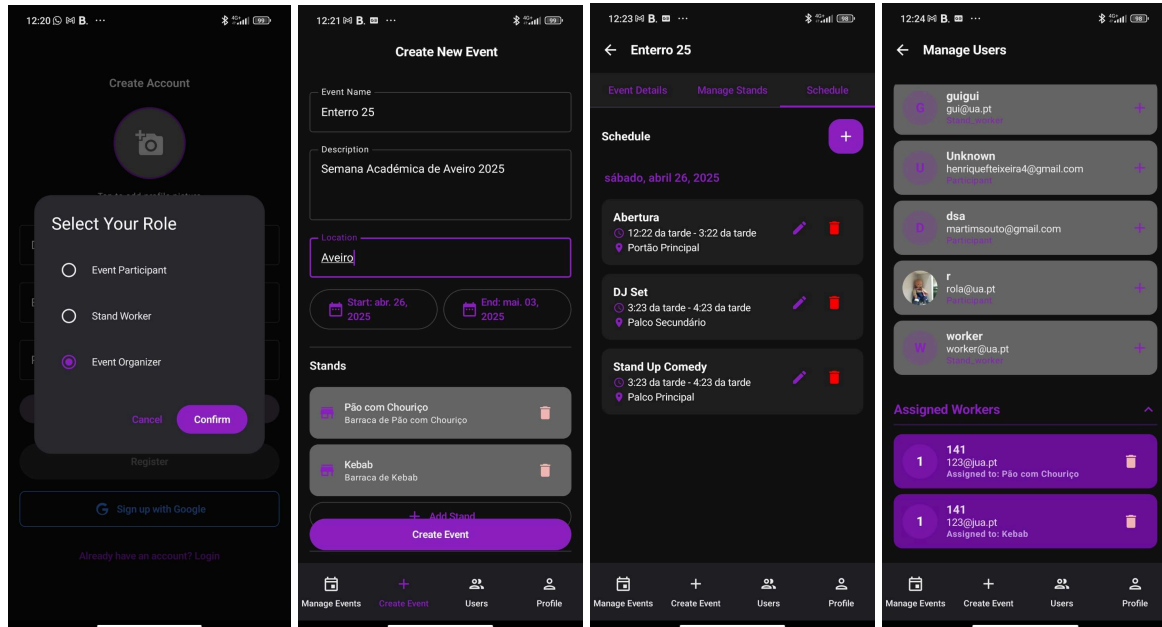
- Firebase Cloud Messaging (FCM) for push notifications
- Custom `AWAVFirebaseMessagingService` for handling notifications
- Token management for user-specific notifications
- Background message handling and notification display



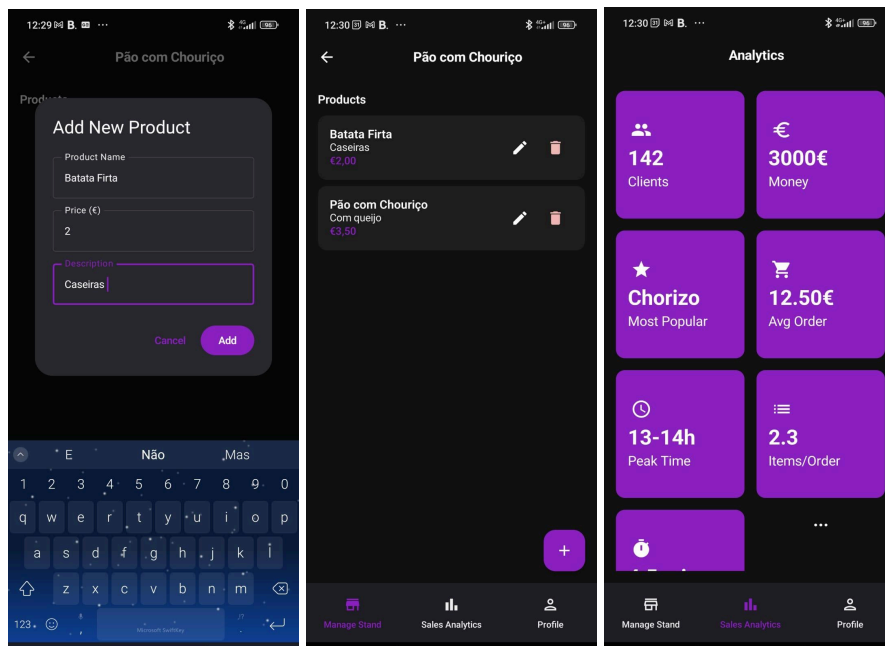
Implemented interactions

Here we are presenting the flows that fulfill the user stories presented in the Journey Maps

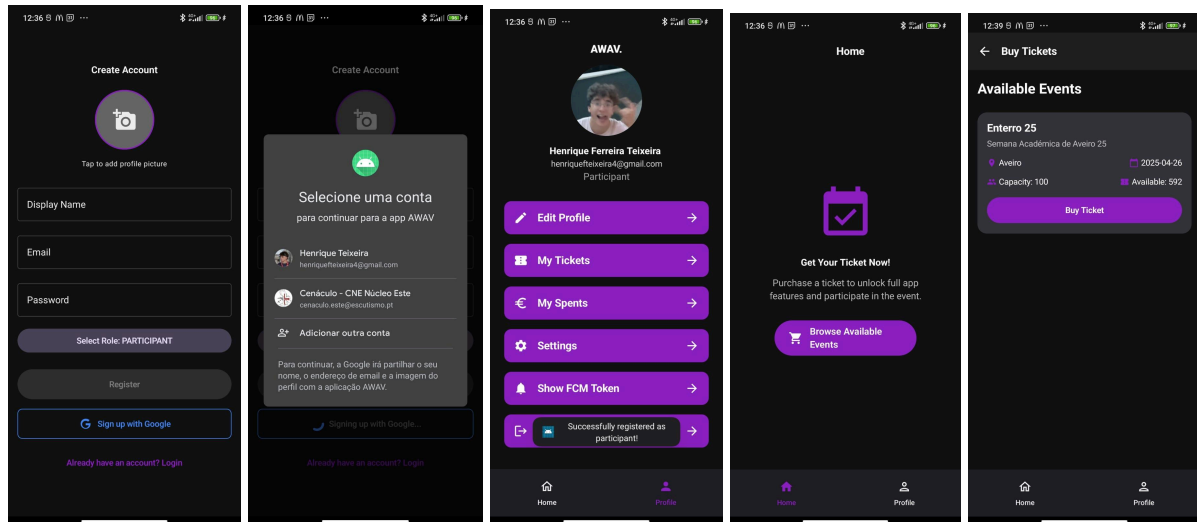
- **Organizer:** Create an Event and fill its requirements



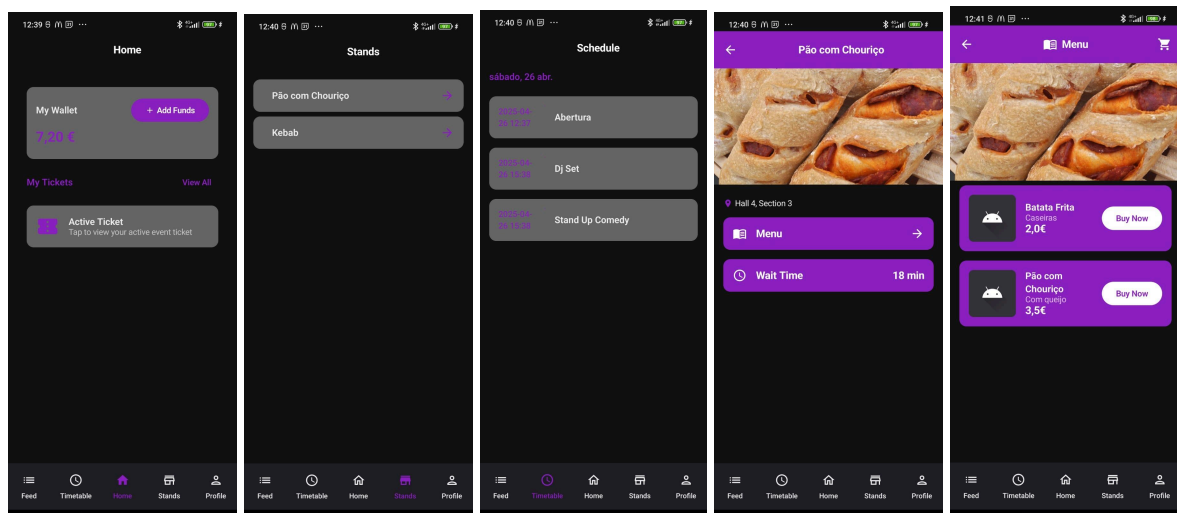
- **Store Worker:** Add items to Stands and see its analytics



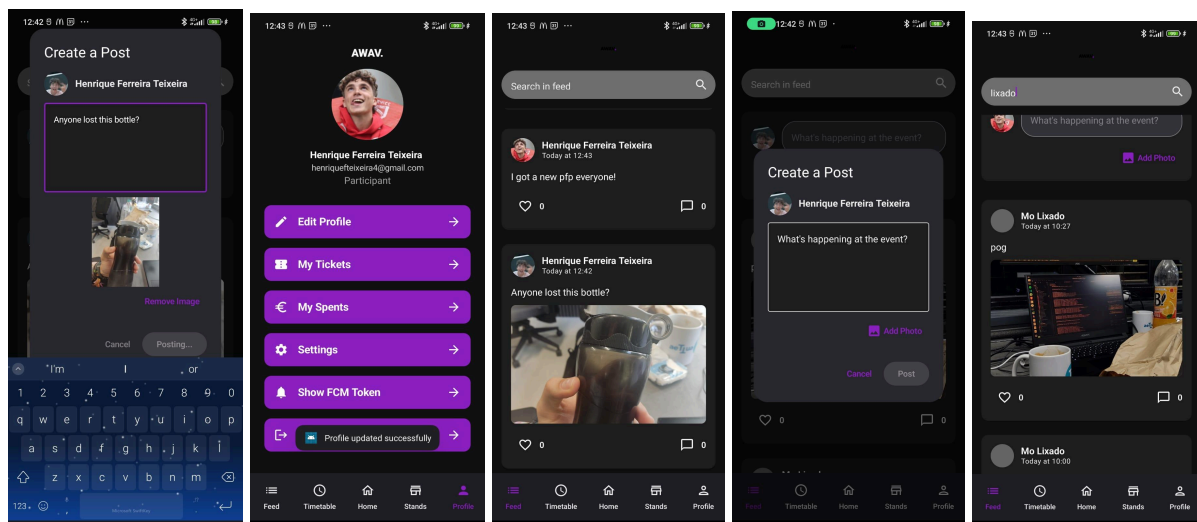
Participant: Buy Ticket



Participant: See schedule & Order Food



Participant: Interact with feed (post and filtering) & change profile



Project Limitations

NFC and QR code payments weren't implemented, we underestimated its complexity. The participants and store worker analytics also were not implemented.

New features & changes after the project presentation

-

3 Conclusions and supporting resources

Lessons learned

The only android cellphone we had available was an android 12, having the app compatible with the last android version to run on the emulator and on android 12 was quite challenging, We also were not aware of the amount of UI we would have to develop. Having so many entities coherent with each other also proved to be a challenge.


Work distribution within the team

Taking into consideration the overall development of the project, the contribution of each team member was distributed as follows: Henrique Teixeira did 50% of the work, and Martim Santos contributed with 50%.

Reference materials

<https://developer.android.com/codelabs/basic-android-kotlin-compose-persisting-data-room?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-6-pathway-2%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-persisting-data-room#5>
<https://firebase.google.com/docs/android/setup>
<https://firebase.google.com/docs/firestore/quickstart>
<https://firebase.google.com/docs/firestore/data-model>
<https://firebase.google.com/docs/auth/android/start>
<https://firebase.google.com/docs/auth/android/google-signin>

Project resources

Resource:	Available from:
Code repository:	https://github.com/martimsantos31/ICM-AWAV-P1
Ready-to-deploy APK:	https://github.com/martimsantos31/ICM-AWAV-P1/tree/main/APK
App Store page:	-
Demo video:	 demo_114614_114588