Report/Documentation

Introduction

The Android application is designed to showcase a Compose-based user interface for an online shopping platform. The app includes screens for product details, order history, and a common product item display. This report provides a comprehensive overview of the implementation, design choices, and reflections on the development process.

Implementations

1. Architecture:

The application was made for following a Model-View-ViewModel architecture, leveraging the ViewModel and Compose for the UI components.

The Room database is used for local data storage, handling entities such as Product, ShoppingCartProduct, and Order.

And the Retrofit is initialized for the network communication with the API.

2. User Interface Components:

I´ve used a bunch of various and different layout components such as for example Box, Column, Row, and LazyColumn, which are employed for structuring the screens of the application.

The AsyncImage component is utilized for asynchronous displaying and loading the images of the specific products.

3. Navigation:

The navigation between the different screens is possible with the usage of the NavController from the Android Navigation component.

Icons, iconbuttons and buttons is used for a better user interaction, enhancing the user experience in total.

4. State Management:

State management is handled using MutableStateFlow in the ViewModel.

Loading states are monitored, ensuring a smooth user experience during data fetching. This makes the user to instantly start to wait for something, instead of being clueless when nothing happens.

5. Database:

Room database operations are performed asynchronously using coroutines in the ProductRepository. The repository is the string that connects the data sources of the application and the ViewModel.

Reflections

1. Decisions and Improvements:

If creating, or when, a similar project comes up again, more attention could be given to error handling. Especially during the network requests, to provide a better user experience in case of failures.

A more modularized approach could be adopted to further enhance code maintainability and readability. Also, simplicity is a key word to stay organized in a project like this and is time-consuming to fix at the end of the project. So that´s something that can improve.

2. Time Allocation:

A lot of time was spent on design and implementations of the UI components to ensure a user-friendly interface which is visually appealing, and easy to use and understand.

Debugging takes a lot of time, so therefor, error handling is an important ingredient in a project like this. A lot of time went to find out about error codes and stuff like that.

3. Satisfaction:

The thing about android programming, is that when one succeed with a piece of code, it gets, at least me, an amazing feeling of satisfaction. Because it´s not always it shows you directly what the issue with the code is, so therefor when one finds out, satisfaction levels are through the roof.

4. Challenges during the project:

The most challenging was probably how to manage the different screens and components, which interluded with each other. If the structure and code wasn´t clean and tidy, one could easily get confused.

Also, incorporating proper error handling and feedback mechanisms was challenging, since I didn´t realize that it was important until quite late in the project.

Final Thoughts

The development and project of this application has been a valuable learning experience. I will take this knowledge, experience, and curiosity further into my future career as a developer. Android is a huge market, and this has given me a great foundation to keep working on.