Technical Task: Product Browser App (Kotlin Multiplatform)

Overview

As part of Revest's mobile engineering team, you may be expected to translate business requirements into scalable, maintainable code across platforms. This task will help us evaluate your understanding of clean architecture, multiplatform development with Kotlin, and your ability to build a simple, functional mobile experience for both Android and iOS.

You are required to build a Kotlin Multiplatform Mobile (KMM) application using Kotlin Compose Multiplatform, targeting both Android and iOS.

The app should consume product data from a public API: https://dummyjson.com/docs/products

Business Requirements

Revest is exploring a cross-platform mobile product catalog prototype for internal use. The app should allow users to:

1. View a list of products showing name, price, and thumbnail.
2. Tap a product to view detailed information including title, description, brand, price, andrating.
3. Search products by keyword (integrate API-based search).
4. [Optional] Filter products by category.

Technical Requirements

Architecture and Code Structure:

- Use Clean Architecture: separate data, domain, and presentation layers.
- Use Kotlin Compose Multiplatform for building shared UI components across Android and iOS.
- Use Ktor Client for API requests.
- Use kotlinx.serialization for JSON parsing.
- Use StateFlow for managing UI state in ViewModels.

- Dependency Injection is optional (manual is acceptable).

Technical Task: Product Browser App (Kotlin Multiplatform)

Functionality Scope:

- Android and iOS support via Kotlin Multiplatform.
- Product list screen.
- Product detail screen.
- Search functionality integrated with the API.
- At least one business use case (e.g., GetProducts, SearchProducts).
- At least one unit test for a use case or repository.

Deliverables

1. A GitHub repository containing the full source code.
2. A README file that includes:
    - Summary of the business requirements.
    - Project architecture overview.
    - Instructions on how to build and run the app on Android and iOS.
    - Any trade-offs or assumptions made during development.

Evaluation Criteria

Architecture: Clean separation of concerns with well-structured modules

API Integration: Proper use of Ktor client and handling of network data

UI Design: Functional and consistent UI using Compose across Android and iOS

Code Quality: Readable, maintainable, testable Kotlin code

Platform Support: Working build for both Android and iOS targets

Business Understanding: Accurate translation of feature requirements into functional app logic

Bonus Points: Filtering by category, local caching, iOS previews using Compose Multiplatform