2CODERS

iOS Technical Assignment

The test will require building different screens based on “The Movie Db” service.

Reference: https://developers.themoviedb.org

1) Main Screen - Trending Movies:
    a) Must show a list of movies within a collection view within a custom layout.
    b) Must be paginated through infinite scrolling.
    c) The entity must show a title, an image*, and a short description.
2) Details Screen: Once you pick a movie, the application must open a new screen with the details of the selected movie.
    a) The screen must show extended content apart from the standard information, such as ratings, votes, authors - feel free to add as much information as you want. (Scrollable)
3) Search screen: Search content. The app must be able to search for different movies and series, selecting which one in advance. The search feature must be throttleable.

**all images should be fetched based on the view size*

Bonus features:

- Ability to have a favorite list (locally), but also indication on the entity itself
- Offline mode (storage of choice)
- Pre-cached via concurrent/queue download of content for immediate access
- Low-res images before accessing the final version
- Image caching mechanism
- Build a .xcframework core within the model to be consumed for tvOS, iOS, iPadOS & macOS

Technical details:

- ➢ Usage of more advanced architectures like MVVM , TCA or some similar.
- ➢ Free to use SwiftUI but must show some knowledge of UIKit or vice-versa.
- ➢ Use SPM/Cocoapods for any dependency that might be required apart from networking, that should be native
- ➢ Design, style, and animations as you prefer
- ➢ Test code coverage.
- ➢ Integrate SwiftLint within XCode (introduce some rules)
- ➢ The interface must work on iPad and iOS in landscape and portrait mode. Feel free to adapt the design to look better based on the screen size.
- ➢ Environment (i.e. Prod/Dev)t selection via scheme within its configuration
- ➢ The assignment must be delivered via Git repository, containing commit history
- ➢ English might drive the development of the exercise, files, comments, classes, etc.

Goals:

- ❖ Prove mastery of advanced programming swift skills and concepts on different Apple platforms.
  - ➢ Good knowledge of advanced architectures.
  - ➢ Protocol-oriented programming.
  - ➢ Object-oriented programming.
  - ➢ Classes, objects and management of access control levels.
  - ➢ Generics
  - ➢ Working with Rest API & JSON data
  - ➢ Native networking management in Swift
  - ➢ Different request management: Queue and concurrency.
- ❖ Xcode environments, schemes & applications capabilities:
  - ➢ Shared schemes
  - ➢ Environment variables from the Xcode env when running a target in debug.

- Knowledge and experience with writing code to support cross-platform (iOS/iPadOS/tvOS/visionOS), as well as handling older OS version (deprecating APIs)

- Experience with advanced Interface-builder concepts, including:
  - Strategies for proper sourcing & instantiation of interface builder .xib & .storyboard.
  - Variable auto layout traits for universal applications (iOS & iPadOS) that support all device classes & device orientations
- Clean code practices:
  - Avoidance of overly complex code and common pitfalls such as cyclomatic dependencies, repeated code, large function signatures, very long lines of code, etc.
- Advanced experience using Xcode's XCTest Suite, including:
  - 80+% code coverage in XCTests for the framework target
  - Writing XCTests with expectations to succeed/fail
  - Tests that cover asynchronous code.
  - Recording & running UITests