1 Apps iOS/Android Exercise : Currency!

What is Currency?

Currency app is designed to helps customers to hand different currency conversions.

Technology constraints

At the digital factory, we currently have a native Android/iOS application. We would like you to develop this application in Kotlin/Swift. We do *not* expect you to build the API – rather we would like you to use.

Note: Do not use any SDK, this task expected to use and integrate to the API provided

What are we looking to test?

- The overall software architecture of the application
- The structure and **quality of the code itself**
- The use of well-known patterns for Android/iOS development

What are we *not* looking to test?

- We don't expect you to be a designer. Something simple and intuitive is all we need

External Resources

- Fixer APIs

1 Apps iOS/Android Exercise : Currency!

The team at Currency!, have developed a backlog of the stories that they would like implemented

Sprint 1:

FF-1: Convert Currency

As a user I want to able to convert currencies from different bases.

ACCEPTANCE CRITERIA

- ❑ From/To dropdown lists showing all available currencies.
- ❑ Input field for amount (numbers only), by default always 1, and another input field to show the converted value.
- ❑ On amount change, converted value changed, And vice versa.
- ❑ Button to swap the values in FROM and TO, and accordingly converted data changed.

FF-2: Historical Data

As a user, I want to see the historical data for my FROM/TO selections in last 3 days (day by day)

ACCEPTANCE CRITERIA

- ❑ When I press details, I can see new screen having historical data for last 3 days (day by day) in the following look:
  - ❑ Historical data for last 3 days viewed in a list
  - ❑ Historical data for last 3 days viewed on chart **(Bonus)**

Sprint 2:

FF-3: Other Currencies

As a user, I want to see the rates converted to some other popular currencies.

ACCEPTANCE CRITERIA

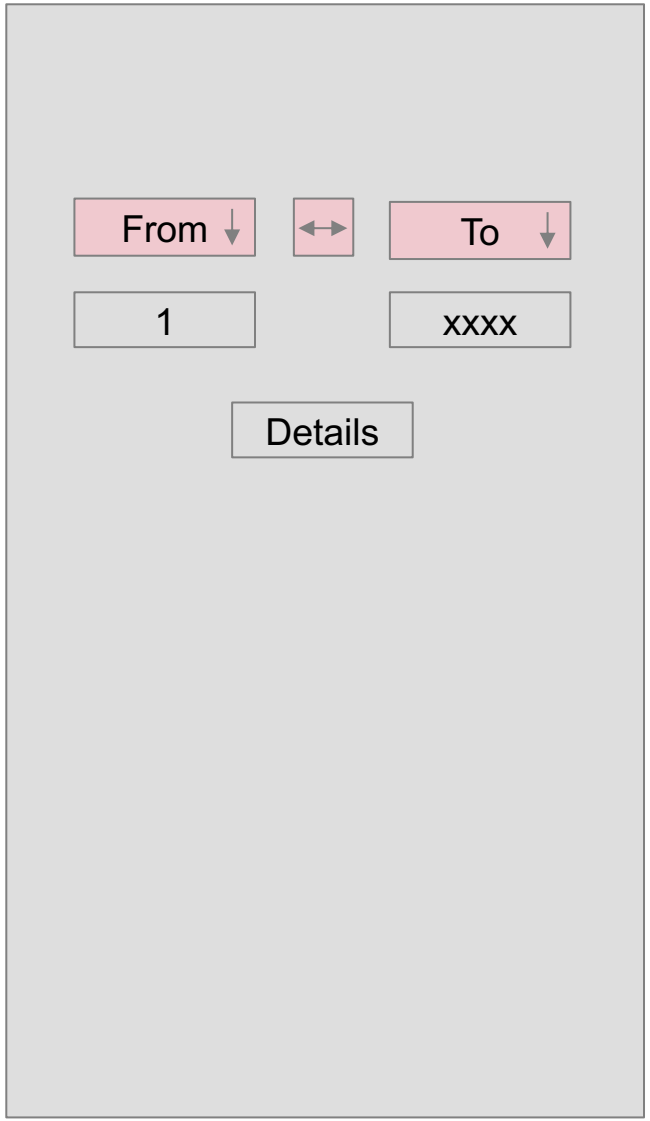
- ❑ When I press details, I can see new screen having conversion from the same base currency to 10 different popular currencies

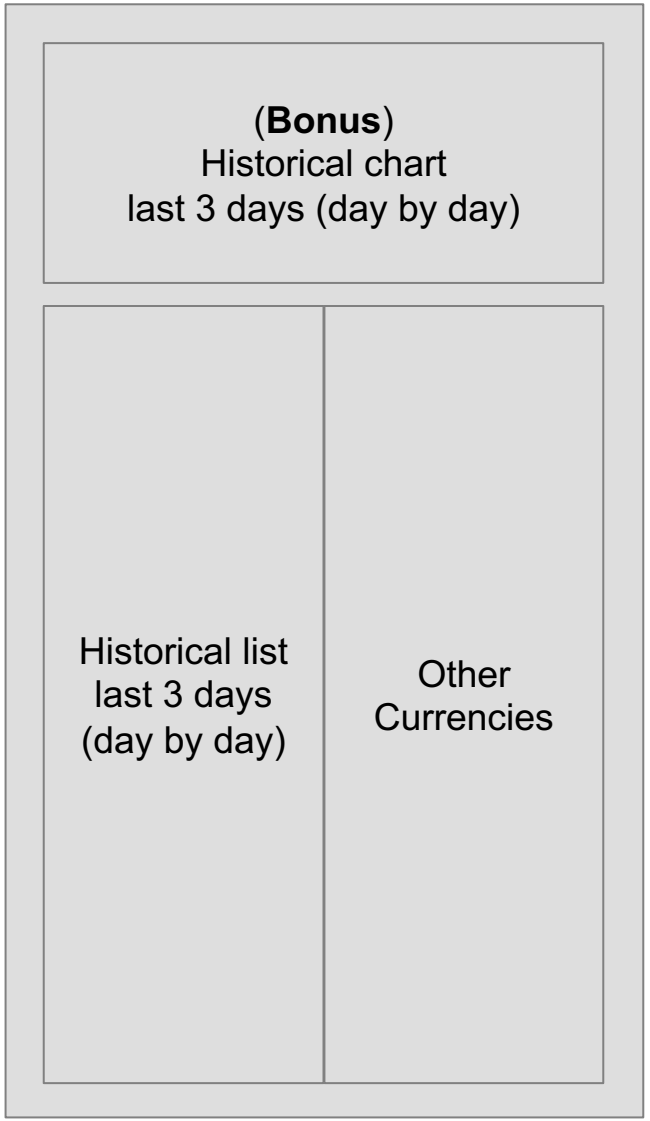
Note: Not all API endpoints are supported in the free subscription which intended to be used in this exercise, so you have to think on how to reach the business scope needed for each requested story using only the free endpoints.

TIP: focus on doing fewer stories but doing them right!

Things to care about

- For **Android**, we expect to use Jetpack MVVM, KTX, Data binding, Navigation component, Hilt, Coroutines, and Retrofit.
- For **iOS**, we expect to use MVVM, RXSwift.
- App should support portrait and landscape normally and without any issues in all app screens.
- App structure should follow clean code architecture with a good separation between app layers.
- We expect app to have a precise error handling, that handle all expected errors, and be able to differentiate between internet connection errors, and API errors.
- App screen layouts should be responsive and adaptive enough to work on different device screens, and **NO** tablet support is required.
- Make sure that your code is formatted and free of linting issues.
- Task should be delivered in a git repository (Github, Bitbucket, Gitlab, etc …), and we expect to see commits progress on the task, AVOID one single commit for all.
- Include unit tests for all applicable areas to be tested.


From
To
1
xxxx
Details


(**Bonus**)
Historical chart
last 3 days (day by day)
Historical list
last 3 days
(day by day)
Other
Currencies