MobileUpLLC/trainee-test-android

Тестовое задание для стажировки "Нашкодим - Android"

Привет! Наша стажировка "Нашкодим - Android" вот-вот начнется. Кандидатов, наверняка, будет немало, а тех, кто проводит стажировку - не так и много, поэтому в выборе мы ограничены. Это тестовое задание - важный этап для прохождения на стажировку. Мы верим, что ты с ним справишься 😉

Что необходимо сделать

Идея приложения довольно проста: это список криптовалют и отдельный экран с чуть более подробной информацией о каждой монете. Ниже ты увидишь подробное описание функциональности приложения. Приложение несложное, но, как и везде, есть свои "детали" 🙂

Все необходимые ссылки:

- Figma
- CoinGecko API

Стек для реализации

- Kotlin
- Single Activity
- Jetpack Compose / XML
- Retrofit
- Kotlinx Serialization / Gson / Moshi / etc.
- Coroutines / RxJava

Процесс реализации

1. Для начала, ознакомься полностью с этим текстом, дизайном, API. Так у тебя сложится понимание, что должно получиться в результате. Если будут вопросы на этом этапе - задавай их через уже знакомого тебе HR'а 🙂
2. Создай репозиторий на GitHub'е, сделай его открытым, чтобы потом выслать нам ссылку на него.
3. Сделай декомпозицию. Разбей тестовое задание на подзадачи (то есть, декомпозируй).
4. Получившиеся задачи оцени по времени реализации. Результат этого и предыдущего пункта запиши в README.md в своем репозитории.
5. На этом шаге уже можно переходить к разработке, следуя своему плану.
6. Когда завершишь - сообщи об этом все тому же HR'у и скинь ссылку на свой репозиторий.

На что мы обратим внимание

Для нас важен вопрос "Как?". Мы будем обращать внимание на то, как ты организуешь весь код, на какие компоненты разобъешь, слои, и так далее. Читаемый и понятный код - это то, что мы тут очень любим. А также смотрим на:

- Умение разбивать задачи и планировать временные затраты
- Реализованный функционал из требований
- Соответствие дизайну
- Стиль и оформление кода
- Наличие ошибок (крашей, неправильного поведения)
- Читаемость и осмысленность истории git-коммитов

Функциональные требования

Экран со списком криптовалют (Экраны в Figma 1.*.*)

Сверху - классический Toolbar со статичным заголовком. В нем же ниже - "чипсы" (Chips), которые используются как переключатель

валюты для запроса списка. Валюта (usd, eur) будет передаваться в запрос на сервер в качестве query параметра. Это будет определять то, в эквиваленте какой валюты мы получим цену каждого токена из списка. На экране три состояния:

- Загрузка данных
- Список криптовалют
- Ошибка при загрузке

API метод для получения списка: **/coins/markets**. Метод поддерживает пагинацию. Будет достаточно запросить первые 20-30 элементов (постраничную загрузку реализовывать не нужно).

При нажатии на элемент списка осуществляется переход на экран детальной информации о криптовалюте.

Экран с детальной информацией о криптовалюте (Экраны в Figma 2.*.*)

Сверху все тот же простой Toolbar с кнопкой навигации назад. В качестве заголовка - название криптовалюты, которую смотрим. Тут так же три состояния:

- Загрузка данных
- Информация о криптовалюте
- Ошибка при загрузке

Информация состоит из изображения криптовалюты, описания и перечисления категорий. Запрос на сервер: **/coins/{id} (пример: *coins/bitcoin*)** Перечисление категорий - это поле **categories** из получаемого объекта.

Дополнительное задание (Экран в Figma 1.1.3)

Это задание необязательно к выполнению, но будет плюсом, если ты его сделаешь. Задача - добавить Pull to Refresh в экран списка криптовалют. Все😄

Успехов, друг!