ATLANT Test Assignment (RU)

Android developer

Copyright © Tensigma LTD

Updated: 15-Nov-2019

Задание:

Реализуйте Android-приложение на языке программирования Kotlin, выполняющее 2 основные функции:

1. Взаимодействие с удаленным REST сервером для авторизации пользователя
2. Взаимодействие с удаленным WebSocket сервером для получения realtime данных

При запуске приложения необходимо показывать экран логина, состоящий из двух элементов EditText для Email и Password пользователя, одного элемента Button для отправки запроса. По нажатию на кнопку происходит процесс авторизации.

После успешной авторизации необходимо:

- Сохранить токен локально;
- Поддерживать актуальную сессию, путем обновления токена (время можно извлечь из токена, стандарт JWT);
- Экран логина заменить на экран (или открыть новый), отображающий краткую информацию о пользователе в любом формате, а также кнопку Logout, при нажатии на которую должен происходить логаут с возвратом на экран логина. Кроме того, на данном экране должны отображаться realtime данные, получаемые по протоколу websocket (см. ниже);
- В случае если пользователь авторизован и вышел из приложения не сделав логаут, при новом запуске приложения необходимо начать со второго экрана, отображающего информацию о пользователе.

API:

Логин:

POST https://api.dev.karta.com/accounts/auth

{"email":"hello@karta.com","password":"12345678"}

Обновление токена:

POST https://api.dev.karta.com/accounts/sessions/refresh

HEADER: Authorization Token

Логаут:

POST https://api.dev.karta.com/accounts/sessions/end

HEADER: Authorization Token

{"session_id":"bcd4f3f6-a302-45ec-a17c-69f09c72eb0d"}

P.S. session_id - извлекается из токена (JWT)

Информация о профиле:

GET https://api.dev.karta.com/accounts/current

HEADER: Authorization Token

В случае успешной авторизации пользователя необходимо осуществить подписку на новые неподтвержденные биткоин-транзакции с помощью Blockchain WebSocket API: https://www.blockchain.com/api/api_websocket

На экране с информацией об авторизованном пользователе должен присутствовать постоянно обновляющийся список транзакций, общая сумма отображенных транзакций, а также три кнопки:

«Запуск» - подписывается на сообщения о новых транзакциях;
«Остановка» - отписывается от сообщений о новых транзакциях;
«Сброс» - очищает список транзакций и сбрасывает сумму.

Пример использования:

1. Пользователь нажимает на кнопку «Запуск». По каждому сообщению в список добавляется транзакция, обновляется сумма транзакций.
2. Пользователь нажимает на кнопку «Сброс». Список сообщений сбрасывается, сумма обнуляется, но при следующем сообщении снова пополняется список и обновляется сумма.
3. Пользователь нажимает на кнопку «Остановка». Сообщения перестают приходить, но список и сумма остаются в том же состоянии, что и после последнего полученного сообщения.

Приветствуется применение подхода Clean Architecture. Особое внимание будет уделено стилистике и расширяемости кода.