Wunder Mobility

Hello fellow Code Enthusiast!

As part of our interview process we would like you to complete a programming test. The purpose of this assignment is to develop a small app. Please read the description, then create an app to solve the problem at hand. Thanks in advance for your time and effort!

Technical requirements

Please use the following technologies:

Android SDK, Kotlin, RXJava (RXKotlin), minSdk Version 21, any third party library can be used

Important! We are currently refactoring our architecture in reactive fashion, so please make sure to use RxJava or Coroutine to fetch data from network.

Description

As part of the test assignment we would like to ask you to build an app which

tackles the everyday problems we face at Wunderfleet. Please create a git repository for this task, before you start working. Keep commits clean and commit messages short.

The app is a basic combination from displaying data, using the map sdk and consumit REST API's.

The app should have the following 2 views:

View 1: Google Map view

- Access a JSON feed of cars:

  https://s3.eu-central-1.amazonaws.com/wunderfleet-recruiting-dev/cars.json

- Display all car objects from this JSON feed as a pin on the map. The pin should appear on the given geolocation per car
- Also show the user's updated live location
- After tapping on a pin(car) on the map, all other pins should disappear and the car name should be displayed on top of the map pin. Tapping again on the "selected" car will open View 2.

View 2: Car detail view - should show detail information about the clicked car

- Expects the id of the clicked car pin as an attribute
- Fetch detail car data from another JSON feed:

  https://s3.eu-central-1.amazonaws.com/wunderfleet-recruiting-dev/cars/carId (For example getting the detail information for carId = 1: https://s3.eu-central-1.amazonaws.com/wunderfleet-recruiting-dev/cars/1)

- Show the vehicleType image, included in the fetched car object, and display all fetched attributes in a table
- Add an additional button below this table to “quick-rent” this car. To start a quick rental, the user needs to click this button.
- On button click, call another REST API endpoint and provide the carId as a POST parameter to start a (simulated) rental on this car. This endpoint is protected by an Authorization. So don’t forget to add the authorization header, mentioned below.

  URL:
  https://4i96gtjfia.execute-api.eu-central-1.amazonaws.com/default/wunderfleet-recruiting-mobile-dev-quick-rental

Example Body of the POST request:

```
{
    "carId":16
}
```

Header:
Authorization: Bearer df7c313b47b7ef87c64c0f5f5cebd6086bbb0fa

Example of return value:


```
{
  "reservationId": 11,
  "carId": 16,
  "cost": 0,
  "drivenDistance": 0, "licencePlate": "B-BA3048",
  "startAddress": "Uhlandstraße 142",
  "userId": 12, "isParkModeEnabled": false, "damageDescription": "", "fuelCardPin":
  "1234",
  "endTime": 1404778918,
  "startTime": 1404778018
}
```

Don’t forget to notify the user about the success or unsuccessful “rental” call with a toast or something similar.

Additional Information:

When you are ready and done, please create a remote repository in Github, push your project into this repository and share it with us.

Please add a README.txt in your project and answer the following questions in this file:

1. Describe possible performance optimizations for your Code.
2. Which things could be done better, than you’ve done it?

Keep the UI minimal but straightforward, and do not worry about the design, we do not have any restrictions here, native UI elements will suffice! If you feel the need to add some cool features to our small app, feel free to do so.

Also, it would be great if you structure your code using the MVP/MVI/MVVM pattern, or any other pattern that you find suitable (and let us know why).

Feel free to contact us with any questions regarding the assignment or the submission process. We would be happy to help as best we can!

Good luck!