Test task "Mobile Android Application Developer"

Programming language requirements

The application must be implemented using one of the three proposed languages: Java, Kotlin, Flutter. It is allowed to use third-party libraries and any publicly available APIs.

Functional requirements

The following functions need to be implemented:

1. Authorization in the Client account service (Peanut)

The user should have the ability to log in and save the authorization without the need to enter the login and password again. The authorization should be implemented through the following method:

POST method `IsAccountCredentialsCorrect` from the REST service:
`https://peanut.ifxdb.com/docs/clientcabinet/index.html`

The method returns an access token with a limited lifespan in response.

The application should handle the situation of token expiration correctly (when methods from the following points do not work correctly).

Test account data:

```
Login: 2088888
Password: ral11lod
```

The user should have the ability to log out of the account and log in again.

2. Fetching user profile personal data from the Peanut service

Use the methods `GetAccountInformation` and `GetLastFourNumbersPhone` to get data and display them. The documentation is provided in Swagger, and in the body of POST requests, the user's login and the token obtained from the authentication method of the Peanut service in step 1 should be used.

The user should have the ability to view basic information about their account.

3. Fetching a list of user trades from the Peanut service

Use the `GetOpenTrades` method from the Peanut service, which provides access to the user's list of trades. The documentation is provided in Swagger, and in the body of POST requests, the user's login and the token obtained from the authentication method of the Peanut service in step 1 should be used.

The UI should be implemented in a way that allows the user to conveniently obtain the resulting list of trades (for example, in the form of a scrollable set of "cards"). The list should respond correctly to user scrolling, as well as refreshing the list (implementation can be done using a Refresh button or a swipe from top to bottom).

4. Calculation and display of the user's profit amount

It is necessary to use the data from point 3 to calculate and display the total user profit amount (the "profit" field from the method's response). The value should respond to the list updates.

5. List of promotional campaigns (Additional, optional task)

It is necessary to devise and implement the display of company promotional campaigns. The data materials can be obtained through the `GetCCPromo` method from the SOAP service:
`https://api-forexcopy.contentdatapro.com/Services/CabinetMicroService.svc`

This service does not require authentication, and an example request and response are provided in the WSDL. The `lang` parameter can be passed as `en` (for English localization).

The results should be presented in the form of "cards" with the ability to navigate through the provided hyperlink. Instead of the domain «`forex-images.instaforex.com`» you can use «`forex-images.ifxdb.com`».

Interface requirements

The application's structure, UI design, and layout are at the discretion of the applicant. When making decisions, user convenience should be taken into account. Even with the minimal set of test functions described above, the application should have a production-ready appearance.

The application should handle screen rotations, transitions between neighboring applications, and work with various screen resolutions correctly. It is also crucial to handle possible exceptions when there is a loss of internet connection and inform the user in a friendly manner.

Format for submitting results

You need to provide the compiled version of the application in *.apk format and its source code. The source code can be provided either in the form of an archive or hosted in a public repository (github, gitlab, or similar resource).

Before starting the test task, you should inform us of your estimated completion time. After completing the task, please indicate the resulting time. Do not be afraid of this criterion, even if the results differ significantly. Making a reasonable estimate of the time required for the task is a useful skill. It's not always possible to do it objectively, but it's worth trying.