Flutter Project Exercise

— — — — — — — — —

Before You Proceed Further, Read the Instructions below :

Use **Material Components** in the design, and use 'MaterialApp'(only required once), 'Scaffold' along with 'AppBar' in all the screens.
The design should look decent and should be 'Responsive'.(don't focus majorly on the design part, as long as it looks 'decent', it's okay).

Libraries to use : **BLoC** for state management, **GoRouter** for navigation and **graphql_flutter** for implementing GraphQL APIs.(you can use any libraries of your choice for other tasks).

NOTE : Kindly refrain from using 'Query' *(or any other 'UI' related widgets provided by 'graphql_flutter' library)* in Flutter UIs and limit the utilization exclusively to using only the Bloc framework.

Some APIs will give **error responses**(which are intentional), and they **need to be handled properly** in the application. Use 'Card' widget in all of the screens to show the data in the UI.(also for list items)

Instructions to **implement tests** are provided at the end of this document. Kindly ensure the inclusion of tests in the project as it is imperative for our assessment process. ***Please make sure that all tests run flawlessly and yield successful results prior to project submission.***

Will be **assessed** on **standard/best practices** followed, **code quality**, and **organization/separation** of code, and whether the above **mentioned libraries** are used or not. Along with that, this will also be tested by running the app on different sizes of 'Android' and 'iOS' emulators to check responsiveness of layouts in the screens.

Instructions :

1. Download and install Node.js from here.
2. Download and install Postman from here.

3. You will be given a ‘zip’ file of node js project and postman collection.
4. Open node js project and run commands ‘npm install’ to install the libraries and run ‘npm start’ to start the server. You can open ‘http://localhost:4000’ which will show a ‘sandbox’ where you can run the queries from there instead of a postman.
5. Use postman collection to get graphql APIs which are to be implemented in the project. It has the url, and the queries for each of the APIs.

Overview

The app should contain a ‘Login screen’ and ‘Main screen’(with bottom navigation bar).

Login Screen

Create the login screen which takes username and password. The Password should be toggleable(meaning should be able to show/hide). After login, the user should be taken to the ‘Main screen’.

Note : As the API has not been furnished, you can develop the login screen without any backend integration.

Main Screen

This ‘Main screen’ will have a ‘bottom navigation bar’ with three bottom navigation items ‘Home’, ‘Accounts’ and ‘Services’. And the respective screen should be shown when clicking on bottom navigation items.

Note: Use appropriate ‘assets’(like png or svg) as icons(by downloading from any open source) to the bottom navigation item, and do not use inbuilt ‘Icons’ class for the icons.

Home Screen

The Home screen should contain the logged in user’s username(whichever entered in the login page) on top.

Use ‘Home’ graphql API provided in the postman collection for querying API and showing it in the UI.

Use the ‘Card’ widget to show the details.

Accounts Screen

The Accounts screen should contain the list of accounts.(Use ‘Card’ widget for showing the list item of the account).

On clicking of any particular ‘account’ should take the user to the transactions screen.

In the transactions screen, show the ‘accountNumber’ and ‘balance’ on top(use ‘Card’ widget), and below it, should have two tabs ‘Transactions’ and ‘Details’.(use ‘TabBar’ widget)

When clicked on the ‘Transactions’ tab, it should show the list of transactions.

When clicked on the ‘Details’ tab should show all the account details(in ‘Card’ widget).

Use ‘Accounts’ and ‘Transactions’ graphql API provided in the postman collection for querying API and showing it in the UI.

Services Screen

The Services screen will have a list of services which are ‘Loans’, ‘Statements’ and ‘Contacts’.

Loans Screen

Clicking on ‘Loans’ should not open a new screen. Instead, it should show a ‘SnackBar’ with some error message of your choice.

Statements Screen

Clicking on ‘Statements’ should open a new screen. It should have a list of dropdowns with ‘years’ (from 2023 to 2019) and the list of statements should be filtered(handle this from app side) based on the selected year.(use ‘Statements’ graphQL API from postman collection).

Clicking on any ‘statement’ should take the user to a new screen where a dummy pdf file (of your choice) should be displayed.(it can be the same pdf for all statements, you can use any pdf flutter library to achieve this).

Contacts Screen

Clicking on ‘Contacts’ should take the user to a new screen to show the list. The ‘Contacts’ graphql API (from postman collection) was intentionally made to fail. So this needs to be handled in the contacts

screen and show the message properly that is received from the graphql API response.(use ‘Contacts’ graphQL API from postman collection)

Implement Tests(**Important**)

- Please write a ‘Unit test’ to verify the functionality of the ‘list of accounts' retrieval .
- Please write tests to any of the two 'blocs' that you will be implementing in your project.
- Please write UI tests for 'home' and 'services' screens.