***Flutter Entrance Test***

***Nexlesoft Cooperation, Jul 2023***

# Overview

You need to implement the following three simple screens in this examination using real APIs integration.

|  |  |  |
| --- | --- | --- |
| **Sign Up** | **Sign In** | **Blank dashboard** |
| Graphical user interface, text, application  Description automatically generated | Graphical user interface, application  Description automatically generated |  |

When the user opens the app, the Sign-In screen appears. Users can switch to the Sign-up screen. Once users log into the app successfully or sign up successfully, the blank dashboard screen shows.

On the dashboard screen, the users can open the left menu and do logout.

# General project requirement

* Technologies
  + Use getx to manage route, state, navigation, and dependency injection
* Design
  + Follow the design from Figma to implement this test: <https://www.figma.com/file/U2iq3TMMraabKGYwHbGZZk/?node-id=246%3A1334>
  + Please make sure you follow the design as close as possible.
* API integration
  + You can use the real APIs in our development server to implement the required pages. Refer to the **API Reference** section
* Others
  + It’s a plus if the code implementation is covered by some unit tests
* Test submission
  + The implementation is uploaded to GitHub, and you send back to Nexle the repo link to review.
  + Also, upload screenshots of the two screens you did to Github.

# UI Requirements

## Signup screen

* First name, last name, email, and password are required fields.
  + First name rule
    - The first name is required.
  + Last name rule
    - The last name is required.
  + Email rule:
    - The email is required.
    - The email is not valid.
  + Password rule:
    - The password is required.
    - The password must be between 6-18 characters.
    - The password must contain at least one digit, one special character, and one letter.

You will see in the design file how to show validation errors on the Sign-up and Sign-In screens.

* If required fields are not entered, validation error messages are shown, and the **Sign-up** button is disabled.
* If there’s no validation error, when the Signup button is clicked, the app calls signup API
* If signup is successful, the app then calls the signin API to get the login user info, and the access token, and refresh token. If the signin API call is successful, the app does the following:
  + Gets the accessToken and refreshToken from the signin API, saves to app storage. The accessTonen and refreshToken will be the Logout process.
  + Navigate to the Dashboard screen that is defined next.

## Sign-In screen

* Email and password are required fields.
  + Email rule:
    - The email is required
    - The email is not valid
  + Password rule:
    - The password is required
* If required fields are not entered, validation error messages are shown, and the **Login** button is disabled.
* If there’s no validation error, when the Login button is clicked, the app calls login API to log in and saves the token to the browser’s storage. Next time when users launch the web app.
  + If there’s a saved token, the app automatically logs the user in.
  + If there’s no saved token, the app asks the users to log in again.
* If login is successful, the app navigates to the dashboard screen.

## Dashboard screen

* This screen is an authenticated screen. It can be seen in either of the following cases
  + When login succeeds
  + When signup succeeds
  + When users launch the app after successfully logged-in/signed-up previously
* This screen has the default avatar in the top right corner.
* Users can open left menu from hamburger icon and do Logout from there.
* When Logout is clicked, the app calls Logout API.
* If the Logout API is successful, the app does the following
  + Clear the token saved in the browser’s storage
  + Navigate the users back to the Login page

## Sidemenu

* This sidemenu appears when the user taps to the hamburger menu and logout. Check [design section](#_UI_Requirements) for more information

# API Reference

## Login API

* URL
  + <http://streaming.nexlesoft.com:3001/auth/signin>
* Method: POST
* Request payload:
  + email (string)
  + password (string)

E.g.

**{**

**"email": "test1@gmail.com",**

**"password": "12345678"**

**}**

* Response
  + user
    - id (string)
    - email (string)
    - firstName (string)
    - lastName (string)
  + accessToken (string)
  + refreshToken (string)

E.g.

**{**

**"user": {**

**"id": 2,**

**"createdAt": "2023-06-08T04:56:01.051Z",**

**"updatedAt": "2023-06-08T04:56:01.051Z",**

**"email": "test1@gmail.com",**

**"firstName": "Tester",**

**"lastName": "Mr",**

**"role": "USER"**

**},**

**"accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOjIsImVtYWlsIjoidGVzdDFAZ21haWwuY29tIiwiaWF0IjoxNjg2MjAwMTk3LCJleHAiOjE2ODYyOTAxOTd9.a-TQ0MNNwNTczUqcxsDB9Zi748NtrCEgAtgPL98QsEE",**

**"refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOjIsImVtYWlsIjoidGVzdDFAZ21haWwuY29tIiwiaWF0IjoxNjg2MjAwMTk3LCJleHAiOjE2ODY0NTkzOTd9.BsfOo9lqMXIpBE-7b1mAsxo1az5efpaqK0BWKpYQXoc"**

**}**

## Signup API

* URL
  + <http://streaming.nexlesoft.com:3001/auth/signup>
* Method: POST
* Request Payload
  + firstName (string)
  + lastName (string)
  + email (string)
  + password (string)

E.g.

**{**

**"email": "test2@gmail.com",**

**"password": "12345678",**

**"firstName": "Tester",**

**"lastName": "Mr"**

**}**

* Response
  + id (string)
  + email (string)
  + firstName (string)
  + lastName (string)

E.g.

**{**

**"id": 2,**

**"createdAt": "2023-06-08T04:56:01.051Z",**

**"updatedAt": "2023-06-08T04:56:01.051Z",**

**"email": "test1@gmail.com",**

**"firstName": "Tester",**

**"lastName": "Mr",**

**"role": "USER"**

**}**

## Logout API

* URL
  + <http://streaming.nexlesoft.com:3001/auth/signout>
* Method: POST
* Request Payload
  + refreshToken (string)
* Request header
  + Authorization: {accessToken}
* Response (empty body)
  + Status: 204 for success

***Note****:*

* *accessToken is the one you got from the Login API response*
* *the refreshToken is the one you got from the login API response.*