***ReactJS Entrance Test***

***Nexlesoft Cooperation, May 2022***

# Overview

You need to implement these pages in this examination using real APIs integration.

* Login page
* Sign up page
* Dashboard page

When the user opens the app, the Sign-up screen will appear. After the user signs up successfully, the app will forward to the Dashboard page.

If the user already logged into the app before, the next time when the app opens the app again, if the token is still valid, the app will open the Dashboard screen.

# General project requirement

* Technologies
  + Use [react-react-app](https://www.npmjs.com/package/react-scripts) as the initial code structure. Refer to <https://create-react-app.dev>
  + Use [reactstrap](https://www.npmjs.com/package/reactstrap) to build UI components. Refer to <https://reactstrap.github.io> for more details
  + Use [bootstrap 4/bootstrap 5](https://www.npmjs.com/package/bootstrap) for CSS adjustment if needed
  + Use [redux-thunk](https://www.npmjs.com/package/redux-thunk) for state management
  + Use react hook where possible
  + Use [react-router-dom](https://www.npmjs.com/package/react-router-dom) for navigation.
* Design
  + Follow the design from Figma for pages implemented in this exercise: <https://www.figma.com/file/U2iq3TMMraabKGYwHbGZZk/Untitled?node-id=0%3A1>.
  + 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 **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 three screens you did to Github

# UI Requirements

## Signup page

* 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 in Sign up and Sign In page.

* If required fields are not entered, validation error messages are shown, and **Signup** button is disabled.
* If there’s no validation error, when the Signup button is clicked, the app calls signup API to create a new account. Refer to **API Reference** below for the details.
* If signup is successful, the app
  + Gets the token from the signup API to save to the browser’s storage
  + Navigate to a dashboard page that is empty. The requirement of the dashboard is mentioned below in another section.
* Besides, there’s a **Signin instead** button that can be used to navigate between Sign up and Sign in page.

## Login page

* 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 **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 page

* This page is an authenticated page. This page 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 page has the default avatar in the top right corner. When users click on this avatar, a context menu is shown which has an option for Logout.
* 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

# API Reference

## Login API

* URL
  + <http://streaming.nexlesoft.com:4000/api/auth/signin>
* Method: POST
* Request payload:
  + email (string)
  + password (string)

E.g.

**{**

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

**"password": "12345678"**

**}**

* Response
  + id (int)
  + firstName (string)
  + lastName (string)
  + displayName (string)
  + token (string)
  + refreshToken (string)

E.g.

**{**

**"\_id": "5f64ac99e52d546e14a868b3",**

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

**"admin": false,**

**"firstName": "Trung",**

**"lastName": "Huynh",**

**"createdAt": "2020-09-18T12:48:25.081Z",**

**"updatedAt": "2020-09-18T12:48:25.086Z",**

**"\_\_v": 0,**

**"displayName": "Huynh Trung",**

**"token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOiI1ZjY0YWM5OWU1MmQ1NDZlMTRhODY4YjMiLCJpYXQiOjE2NTIxODkwODQsImV4cCI6MTY1MjI3NTQ4NH0.OPwHPF0d0hjnLQaLtfde8tUj8Vo7VJAuAiXF-N5dDGw",**

**"refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOiI1ZjY0YWM5OWU1MmQ1NDZlMTRhODY4YjMiLCJpYXQiOjE2NTIxODkwODQsImV4cCI6MTY1NDc4MTA4NH0.vQJqt9i5fzj2sUqXDUqKjWcFu53wejdpf5ds-8VG8LE"**

**}**

## Signup API

* URL
  + <http://streaming.nexlesoft.com:4000/api/auth/signup>
* Method: POST
* Request Payload
  + firstName (string)
  + lastName (string)
  + email (string)
  + password (string)

E.g.

**{**

**"firstName": "Trung",**

**"lastName": "Huynh",**

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

**"password": "12345678"**

**}**

* Response
  + id (int)
  + firstName (string)
  + lastName (string)
  + displayName (string)
  + token (string)
  + refreshToken (string)

E.g.

**{**

**"\_id": "627a67b3dfcfa579ffcc525e",**

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

**"admin": false,**

**"firstName": "Trung",**

**"lastName": "Huynh",**

**"createdAt": "2022-05-10T13:25:07.366Z",**

**"updatedAt": "2022-05-10T13:25:07.369Z",**

**"\_\_v": 0,**

**"displayName": "Huynh Trung",**

**"token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOiI2MjdhNjdiM2RmY2ZhNTc5ZmZjYzUyNWUiLCJpYXQiOjE2NTIxODkxMDcsImV4cCI6MTY1MjI3NTUwN30.M1xmyufSX4mPPasyW4X84gfSS5LSxo5Id7p\_nFXVuzw",**

**"refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOiI2MjdhNjdiM2RmY2ZhNTc5ZmZjYzUyNWUiLCJpYXQiOjE2NTIxODkxMDcsImV4cCI6MTY1NDc4MTEwN30.dXNHuRZ54Kt3m2ivDg4NqrJMpsvGencE0gmacRCuO3M"**

**}**

## Logout API

* URL
  + <http://streaming.nexlesoft.com:4000/api/auth/logout>
* Method: POST
* Request header
  + Authorization: {token}
* Response
  + {Empty response}

# Notes:

All the APIs are CORS enabled and only allow these web URLs in the whitelist:

* <http://localhost:3000>
* <http://localhost:3001>
* <http://localhost:4000>
* <http://localhost:4001>

So, make sure your web app is running on one of those URLs in order to avoid the CORS issue.