Technical Design - just a calculator

Motivation

The purpose of this project is to serve as an evaluative tool during the Software Engineer hiring process at Vial. The project entails the creation of a functional calculator application, showcasing the applicant's skills and expertise in software development.

Technologies

Front-end

The front-end is developed using React, utilizing the create-react-app template with TypeScript integration. This approach allows for efficient development and maintenance. The project leverages various packages to facilitate abstraction and asset management.

Back-end

The back-end of the application is built using Nest.js, a Node.js-based framework. A significant feature of the back-end is the implementation of an authentication middleware. For this project, a basic authentication mechanism is employed, utilizing a predefined combination of usernames and passwords.

We have two endpoints:

Method	Endpoint	Parameters	What this do?

POST	/auth/login	Request body: JSON, with username and password
		{
		"username":"***",
		"password":"***"
		}
			If the right username and password combination is sent, returns an access token. If not, returns 401 status.

GET	/profile	Header:
		{
		"Authentication":"Bearer ***",
		}
			If the right token is sent, returns the user id and name.

Authentication and Authorization

The back-end serves as the authentication mechanism. Utilizing the Nest.js Passport libraries, the application validates user credentials to generate a Bearer authentication token. The token is used to validate the user's authentication status.

In this specific project, authentication is employed solely to verify the accuracy of the provided username and password combination. Upon authentication success, the user's authentication status is displayed via a banner.

User Interface and Functionality

The front-end's architecture adheres to a simple yet coherent structure that aligns with the project's scope. The 'src' folder houses essential application files such as 'App.tsx' and 'index.tsx', along with 'pages' and 'components' subdirectories. The 'components' folder includes five key files: 'Keypad', 'Login', 'NumberKey', 'OperatorKey', and 'Output'. The 'pages' folder contains the 'Calculator' page. Notably, the project's timeline was impacted by unforeseen circumstances during the initial two days, which limited development progress.

Front-end Packages and Dependencies

The front-end implementation incorporates several packages and dependencies to enhance functionality and aesthetics. These include:

- **Material UI:** Utilized for easy integration of Material Design components.
- **Axios:** Employed for simplified API data retrieval.
- **Styled Components:** Chosen for clear and maintainable component styling.
- **Prettier:** Integrated as a development dependency to ensure code consistency.

Calculator Functionality

The calculator's capabilities encompass fundamental arithmetic operations (addition, subtraction, multiplication, and division), support for parentheses, exponentiation, memory for calculations, and a login button positioned at the top of the viewport, with a banner that shows the authentication status.

Login Modal

Upon clicking the login button, users are presented with a login modal featuring input fields for usernames and passwords, along with a login button. Successful authentication prompts a message indicating successful login, with the login button banner changing to a green color. Conversely, entering incorrect credentials triggers an appropriate error message.