Duodeka Todo App

This repository contains a TODO list application developed as part of a technical assessment for DUODEKA. The application is built using React, leveraging Context and Hooks for state management, and utilizes Material-UI for the user interface.

Features

- Add, edit, check, and delete TODO items.
- All items are stored in a global state (using Context API).
- Uses Material-UI components for a sleek user interface.
- Mobile-friendly design with a responsive layout.
- Multiple pages with a two-column layout, featuring a menu and content area.

Technical Stack

- **React**: Frontend library for building user interfaces.
- **Context API and Hooks**: For managing global state without Redux.
- **Material-UI**: React components that implement Google's Material Design.
- **Lodash/ES6 Methods**: For efficient data manipulation.
- **Docker**: For containerizing and distributing the application.

Running the App via Docker

You can easily run this app using Docker. Follow these steps:

1. **Pull Docker image:**

   ```
   docker pull eternalmay33/todo-list:latest
   ```

2. **Run the Docker Container**:

   ```
   docker run -p 3301:80 eternalmay33/todo-list:latest
   ```

   This will start the app on port 3301 of your local machine.

3. **Access the app**

   Open your browser and go to **http://localhost:3301**.

Repository Link

You can find the source code for the project on GitHub:

https://github.com/33may/duodeka-todo.git

Approach and Learnings

In this project, I approached the task by focusing on creating a simple yet effective global state management solution using Context and Hooks. This allowed me to avoid the overhead of Redux for a prototyping scenario while still providing a structured way to manage state.

Key Learnings:

- **Global State Management**: Learned to efficiently handle global state without external libraries like Zustand or Redux.
- **Material-UI Integration**: Continued working and exploring implemention of a user interface using Material-UI components and mobile responsive design with tailwind.
- **Dockerization**: Practiced how to containerize a React application using Docker, making it easy to build, share, and deploy.

Contact

For any queries or further information, feel free to contact me at

508874@student.funty.nl