AI TRIP CREATOR: CODING STANDARDS

THE ROLLING CAPSTONES

27 September 2024

Contents

1 Introduction

This document outlines the coding standards to be followed while developing a React application with a Firebase backend and MUI component library with CSS for styling. It also includes guidelines for handling API endpoints within the frontend codebase.

2 Folder Structure

- Organise files and folders logically based on their functionalities.
- Separate concerns by categorizing files into directories such as components, pages, utils, etc.
- Group related components or modules together within their respective directories.

3 Naming Conventions

3.1 React naming Conventions

- Use descriptive and meaningful names for React components.
- Components should be named using PascalCase convention.
- Components representing pages should be suffixed with "Page".
- Components representing reusable UI elements should be suffixed with "Component".

Note: An image example of React naming convention was originally included here.

4 CSS, Styling and Component Library

4.1 MUI Component library

4.1.1 Component Importing

- Import components directly from @mui/material to leverage tree-shaking.

Note: An image example of importing components from MUI component library was originally included here.

4.1.2 Component Customisation

- Use sx prop for inline styling whenever possible to ensure consistent theming.
- Use styled API from @mui/system for more complex component customization.

Note: Two image examples of component customisation were originally included here.

4.1.3 Theming

- Utilize the theme object for consistent styling.
- Define custom themes using createTheme and ThemeProvider.

Note: An image example of Theming from the MUI component library and React library was originally included here.

4.1.4 Code Readability

- Keep component usage clear and concise.
- Avoid deeply nested components.
- Break down large components into smaller, reusable components.

Note: An image example of code readability in the program was originally included here.

4.2 CSS Styling

4.2.1 General

- Use a consistent style format
- Avoid overly specific selectors.
- Use class selectors over ID selectors to promote reusability.
- Define a base font size and use relative units.
- Keep CSS files as small as possible.
- Remove unused styles.

Note: An image example of general styling in CSS was originally included here.

4.2.2 CSS Variables

- Use CSS variables for theme colours, fonts, and other repetitive values.

Note: An image example of CSS variables was originally included here.

4.2.3 Performance

- Minimise the use of complex selectors.
- Avoid using !important unless absolutely necessary.

5 Code Formatting

- Use consistent indentation (preferably 2 or 4 spaces) for improved readability. Follow JavaScript best practices for code formatting.
- Utilise ESLint for enforcing code consistency.
- Configure ESLint rules according to project requirements and coding standards.
- Regularly format and lint code to maintain code quality.

6 API Endpoint Handling

- Centralise API endpoint configurations and handling.
- Utilize asynchronous functions for making API requests.
- Abstract API logic into separate modules for reusability and maintainability.

Note: An image example of API endpoint handling was originally included here.

7 Version Control

- Utilise version control system Git for tracking changes and collaborating with team members.
- Follow branching strategies Git Flow for managing feature development and releases.
- Write meaningful commit messages that describe the purpose and scope of changes.
- Regularly push changes to remote repositories and pull updates from upstream branches.

8 Testing

- Implement unit tests, integration tests, and end-to-end tests to ensure code reliability and functionality.
- Use testing frameworks Jest for writing tests in React applications.
- Use Cypress for End-to-End Testing.
- Follow test-driven development (TDD) practices when applicable.
- Automate testing processes as much as possible to streamline development workflows.

Note: An image showing passed tests was originally included here.