Report of Web and Mobile Assignment 3

Aykhan Ismayilzada

The React-based Flash Card App is an application designed for web platforms. This report provides explanation of the app's architecture, React components, state management using hooks, routing, API integration, and overall functionality.

**1. App Component:**

The App component serves as the entry point for the Flash Card App. It utilizes the react-router-dom library to set up routes for different pages. Within the <Routes> component, the app defines routes for the Home, Flash Cards, and Contact pages. The App component encapsulates the entire application within a Router, facilitating navigation between different views.

**2. Home Component:**

The Home component functions as the landing page of the application. It provides users with an introduction to the Flash Card App, outlining its purpose and functionalities. Furthermore, it showcases projects created during the course, each represented by the ProjectItem component. This approach enhances reusability, as the ProjectItem component encapsulates the logic for rendering individual project items, promoting a modular and maintainable codebase.

**3. ProjectItem Component:**

The ProjectItem component is a presentation component responsible for rendering individual project items. It receives props such as title, description, and link to display project information. By encapsulating the rendering logic within a separate component, the codebase benefits from improved readability, maintainability, and reusability.

**4. ContactPage Component:**

The ContactPage component is the entry point for users to submit messages through a contact form. It incorporates the ContactForm component, which handles the capture of user inputs for the subject, email, and content of the message. Upon submission, the form triggers an API call to create a new message, facilitating seamless communication between users and the app.

**5. ContactForm Component:**

The ContactForm component, a functional component, employs React hooks to manage the state of form inputs. The form includes fields for the subject, email, and content, with built-in error handling for submission failures. The submission of the form triggers an API call to create a new message, providing a streamlined communication channel and exemplifying React's capacity for managing form state efficiently.

**6. FlashCards Component:**

The FlashCards component acts as a container for routing within the Flash Card feature. It defines routes for the Flash Card List (FlashCardList) and Flash Card Item (FlashCardItem). Leveraging the react-router-dom library, the component facilitates navigation between different views within the Flash Card section.

**7. FlashCardList Component:**

The FlashCardList component is important in managing the display and interaction with flash cards. It fetches flash card data from a local API using the api module and provides users with the ability to search, filter by status, and sort cards based on different criteria. The component efficiently manages the state of flash cards and incorporates a modal (react-modal) for creating and editing flash cards.

**8. FlashCardItem Component:**

The FlashCardItem component, is responsible for rendering individual flash cards. Utilizing React hooks, it manages the state of card flipping, enabling users to toggle between the front and back views of a flash card. The component displays crucial information, including the front and back content, last modification date, and card status, enhancing the user's learning experience.

**9. FlashCardForm Component:**

The FlashCardForm component is for rendering the form used for creating or editing flash cards. It manages form inputs for the front, back, and status fields, updating their state with React hooks. The component includes a selection dropdown for the status field and triggers API calls for creating or updating flash cards upon form submission, showcasing the seamless integration of form management and API interaction.

**10. API Module (api.js):**

The api module plays a pivotal role in handling communication with the local API server. It offers functions for fetching, creating, updating, and deleting flash cards and messages. By abstracting away the complexities of HTTP requests, the api module ensures that the app interacts seamlessly with the server, showcasing the power of modularization in maintaining a clean and organized codebase.

**Conclusion:**

In conclusion, the React-based Flash Card App uses the efficiency and flexibility of the React library in web application development. Each component within the app serves a distinct purpose, contributing to the overall functionality and user experience. The use of React hooks for state management, the integration of routing for navigation, and the modularization of components facilitate coding a project.