**Frontend**

This document outlines the structure and functionality of the frontend for the calculator application. The frontend is built using React and consists of several components that work together to provide a user-friendly interface for performing calculations.

**Project Structure:**

The frontend code is organized into the following components:

* **Calculator.js:** The main component that handles the calculator logic.
* **Button.js:** A reusable component for creating calculator buttons.
* **Display.js:** Displays the current calculation input and result.
* **Modal.js:** A reusable component for displaying the calculation history.

**Component Breakdown**

**Calculator Component:**

* **State Management:**
* currentInput: Stores the currently displayed number or operation.
* previousInput: Stores the previous input for calculations.
* operation: Stores the selected arithmetic operation.
* history: Stores an array of previous calculations.
* showHistoryModal: Controls the visibility of the history modal.
* **Functionality:**
* Handles button clicks to update the calculator state based on the button value.
* Performs calculations based on the current input, previous input, and selected operation.
* Updates the calculation history.
* Fetches calculation history from the backend and displays it in the modal.
* **Dependencies:**
* useState and useEffect hooks from React for managing component state and side effects.
* axios for making API requests to the backend.

**Button Component:**

* **Props:**
* value: The value displayed on the button.
* onClick: A callback function to handle button clicks.
* **Functionality:**
* Renders a button element with the provided value and onClick handler.

**Display Component:**

* **Props:**
* value: The value to be displayed on the screen.
* **Functionality:**
* Renders the provided value as the display content.

**Modal Component:**

* **Props:**
* onClose: A callback function to close the modal.
* children: The content to be displayed inside the modal.
* **Functionality:**
* Renders a modal overlay with a close button and the provided content.

**Styling**

The frontend uses CSS to style the calculator and its components. The styles.css file contains the styles for the calculator, buttons, display, and modal.

**Interaction with Backend**

The calculator component fetches calculation history from the backend using the axios library. The backend endpoint is /history.

**Potential Improvements**

* **Error handling:** Implement error handling for cases like division by zero or invalid input.
* **Accessibility:** Improve accessibility by adding appropriate ARIA attributes and keyboard navigation support.
* **Performance optimization:** Optimize the component rendering and state updates for better performance.
* **Unit testing:** Write unit tests to ensure the correct behavior of components.