Calculator

Objective: The objective of this project is to create a functional and user-friendly calculator using React. It allows users to perform basic mathematical operations like addition, subtraction, multiplication, and division, as well as advanced operations like percentage calculation and handling parentheses. The calculator is designed to be intuitive, responsive, and capable of error handling for invalid expressions.

About: This project is a simple React-based calculator application that mimics the functionality of a physical calculator. It utilizes React's state management to update and display calculations in real time. The app provides a clean and minimalistic interface where users can input numbers and operations through clickable buttons. Error handling ensures that invalid expressions result in a clear "Error" message, preventing crashes or unexpected behavior.

Features:

Basic Arithmetic Operations:

The calculator supports fundamental arithmetic operations such as:

Addition (+)

Subtraction (-)

Multiplication (*)

Division (/)

Percentage Calculation (%):

The % button allows the user to calculate percentages. For example, entering 50% will convert 50 to its decimal form (0.50) in a calculation.

Decimal Points (.):

Users can include decimal points in their numbers for performing calculations with floating-point precision.

Parentheses Handling (()):

Parentheses can be used to handle operations with complex expressions, ensuring the correct order of operations (PEMDAS/BODMAS rules).

Dynamic Input Display:

The current input and operations are displayed dynamically in a text input field, so users can see their entries in real time as they perform calculations.

Result Calculation (=):

When the = button is pressed, the calculator evaluates the expression and displays the result. The evaluation uses JavaScript's new Function() for parsing and calculating the entered expression.

Clear Functionality (AC):

The AC (All Clear) button allows users to reset the calculator, clearing all current input and results, setting the display back to 0.

Backspace (Back):

The Back button allows users to delete the last entered character, making it easier to correct mistakes without clearing the entire expression.

Error Handling:

If an invalid operation is entered (e.g., dividing by zero or incomplete expressions), the calculator displays "Error" to notify the user. It prevents the app from crashing due to invalid input.

Responsive Design:

The layout of the calculator is optimized for both desktop and mobile screens, ensuring the app works smoothly across different device sizes.

User-friendly Button Layout:

Buttons are well-spaced and large enough for easy tapping or clicking. This ensures good usability on touchscreen devices and for users of all ages.

Styled Input and Buttons:

The input field and buttons are styled using CSS for a clean, modern look. Buttons have a hover effect, making them more interactive, and the equal button has a distinctive blue color to highlight its importance.

Customizable CSS:

The project is designed with separate CSS, making it easy to customize the appearance (such as button colors, background, padding) according to the user’s preferences.

Fast and Lightweight:

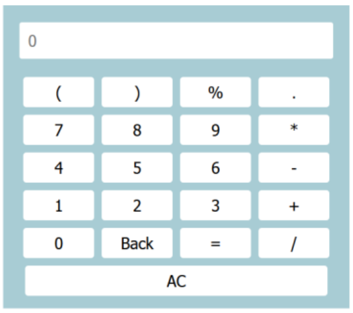
Built using React, this calculator is highly responsive, quick, and efficient in handling multiple operations without lag.

No External Dependencies:

The project only uses core React functionalities and no external libraries for the calculator logic, ensuring that it remains simple, lightweight, and easy to maintain.

These features make the calculator robust and capable of handling most simple to moderately complex calculations while maintaining a user-friendly interface.

Screenshot


0
(
)
%
.
7
8
9
*
4
5
6
-
1
2
3
+
0
Back
=
/
AC