Project Report

Calculator Web Application

1. Project Overview

The goal of this project was to create a simple yet functional calculator web application that allows users to perform basic arithmetic operations. This project serves as a demonstration of fundamental web development skills, including HTML, CSS, and JavaScript.

2. Objectives

- To develop a user-friendly calculator interface.
- To implement basic arithmetic functions: addition, subtraction, multiplication, and division.
- To provide a toggle feature for light and dark themes.
- To enhance accessibility for users.
-

3. Technologies Used

- **HTML**: For structuring the calculator layout.
- **CSS**: For styling the calculator and making it visually appealing.
- **JavaScript**: For implementing the functionality of the calculator.
- **Local Storage**: To save user preferences for theme selection.

4. Features

4.1 Functional Features

- **Basic Operations**: Users can perform addition, subtraction, multiplication, and division.
- **Clear and Backspace Buttons**: Options to clear the display or delete the last entered character.
- **Equal Button**: Computes the result of the entered expression.

4.2 UI Features

- **Responsive Design**: The calculator is designed to be responsive, adapting to various screen sizes.

- **Theme Toggle**: Users can switch between light and dark themes for better visibility.
- **Accessibility Features**: Buttons include aria-labels for screen readers, ensuring the application is accessible to all users.

5. Implementation

5.1 Code Structure

- **HTML File (calculator.html)**: Contains the layout and elements of the calculator, including buttons and display areas.
- **CSS File (calculator.css)**: Styles the calculator, ensuring a modern and user-friendly design.
- **JavaScript File (calculator.js)**: Handles user interactions, calculations, and theme toggling functionality.

5.2 Key Code Snippets

Display Functionality

javascript

```
if (item.id === "equal") {
  try {
    display.innerText = eval(display.innerText);
  } catch {
    display.innerText = "Error!";
    setTimeout(() => (display.innerText = ""), 1000);
  }
}
```

Theme Toggle

javascript

```
themeToggleBtn.onclick = () => {
  const isDark = calculator.classList.toggle("dark");
  body.classList.toggle("active");
  themeToggleBtn.classList.toggle("active");
  localStorage.setItem('isDarkTheme', isDark);
};
```

6. Challenges and Solutions

6.1 Challenges

- **Handling User Input**: Ensuring the calculator handles invalid inputs gracefully without crashing.
- **Design Responsiveness**: Making sure the calculator looks good on both mobile and desktop devices.

6.2 Solutions

- Utilized try-catch blocks to manage errors during calculations.
- Employed CSS Flexbox and Grid for responsive layout design.

7. Conclusion

This calculator project not only showcases my ability to develop a functional web application but also demonstrates my understanding of core web technologies. The project meets the objectives set out at the beginning and serves as a solid foundation for future, more complex projects.

8. Future Enhancements

- **Advanced Functions**: Incorporate more complex mathematical operations (e.g., square roots, exponentiation).
- **History Log**: Keep a history of calculations performed during the session.
- **Mobile App Version**: Consider converting the project into a mobile application using frameworks like React Native.