PROJECT NO-2

Task : Create a Calculator using HTML and Bootstrap

Creating a calculator using HTML and Bootstrap is a practical project that allows you to apply your web development skills while providing a useful tool for performing mathematical calculations. Below, I'll outline the structure and key elements for each section of your project documentation

Abstract:

The abstract should provide a brief overview of the calculator project. Mention the purpose, features, and the technologies (HTML and Bootstrap). HTML Structure:

HTML will be used to create the basic structure of the calculator. This includes defining buttons for numbers (0-9), operators (+, -, *, /), and special functions (clear, equals).

Bootstrap Styling: Bootstrap, a popular CSS framework, will be used for styling and layout. It provides pre-designed CSS classes for creating a responsive and visually appealing user interface.

JavaScript for Functionality:

JavaScript will be used to handle user interactions and perform calculations. When a user clicks a button, JavaScript will update the display and perform the appropriate mathematical operation.

Display Output:

The calculator will have a display area to show the numbers and operations as the user inputs them. When the user presses the equals button, JavaScript will evaluate the expression and display the result.

Error Handling:

The calculator should handle errors gracefully, such as division by zero or invalid input. A calculator is a tool that performs mathematical operations automatically, using numbers entered manually or through sensors. A calculator can be used to add, subtract, multiply, divide, and often perform other operations such as taking square roots, extracting logs, and calculating percentages.

A calculator using HTML and Bootstrap is a web-based calculator that uses HTML and Bootstrap to create its user interface. HTML is a markup language that is used to create the structure of a web page, while Bootstrap is a CSS framework that is used to style web pages.

To create a calculator using HTML and Bootstrap, you would first need to create the basic structure of the calculator using HTML. This would include creating a div element for the calculator itself, as well as div elements for each of the buttons on the calculator. Once you have created the basic structure of the calculator, you would then need to style the calculator using Bootstrap. This would include adding classes to the div elements to control the appearance of the calculator, as well as adding JavaScript to make the calculator functional. By following these steps, you can create a web-based calculator that can be used to perform mathematical operations.

Here are some tips for creating a calculator using HTML and Bootstrap:

Use semantic HTML elements to create the basic structure of the calculator. This will help to make your code more readable and maintainable.

Use Bootstrap classes to style the calculator. This will help you to quickly and easily create a professional-looking calculator.

Add JavaScript to make the calculator functional. This will allow users to perform mathematical operations with the calculator.

This project aims to demonstrate the creation of a functional web-based calculator using HTML and Bootstrap. The calculator is a fundamental tool in various fields, and embedding one into a website enhances user convenience and accessibility. HTML provides the structural foundation, while Bootstrap offers a responsive and visually appealing design. This project serves as a practical guide for web developers and enthusiasts, outlining the step-by-step process of building a calculator from scratch.

Starting with the basics of HTML and Bootstrap, we'll structure the calculator's layout and enhance its visual appeal using Bootstrap's CSS components. We'll then delve into JavaScript, implementing the logic required for performing addition, subtraction, multiplication, and division operations. Throughout the project, we'll focus on user experience and responsiveness, ensuring the calculator functions seamlessly on various devices.

By the project's conclusion, readers will have gained a comprehensive understanding of combining HTML, Bootstrap, and JavaScript to create a user-friendly and customizable calculator. This skillset can be applied to enhance websites across multiple domains, from e-commerce platforms to educational resources. Embracing the power of web technologies, this project empowers developers to provide a valuable tool for users, all within the confines of a web page.

Objective:

The objective for creating a Bootstrap and HTML calculator can vary depending on your specific goals and the context of the project. However, here's a general objective that you can adapt to your needs:

The objective of building a Bootstrap and HTML calculator is to develop a userfriendly, responsive, and visually appealing web-based calculator application. This project aims to provide a practical learning experience in web development, focusing on the following key aspects:

1. Front-end Development Skills: Enhance proficiency in front-end web development by leveraging Bootstrap and HTML to create an intuitive user interface for performing mathematical calculations.

2. Responsive Design: Implement responsive design principles to ensure the calculator functions seamlessly on various devices, including desktops, tablets, and mobile phones, providing users with a consistent and accessible experience.

3. Interactivity: Utilize JavaScript to add interactive features, allowing users to input mathematical expressions and receive real-time results. Implement error handling and ensure accurate calculations.

4.Accessibility: Prioritize accessibility by adhering to web accessibility standards (e.g., WCAG) to make the calculator usable by individuals with disabilities, including keyboard navigation and screen reader compatibility.

5. Cross-Browser Compatibility: Ensure that the calculator works correctly and displays consistently across different web browsers, ensuring a broad user reach.

6. Customization: Explore Bootstrap's customization options to style the calculator's user interface, aligning it with branding guidelines or specific design requirements.

7. Modular Code Structure: Develop a well-organized and modular codebase that is easy to understand and maintain, facilitating future updates or enhancements to the calculator's functionality or design.

8. Learning Experience: Use this project as an opportunity to learn and apply best practices in web development, including HTML, CSS, Bootstrap, and JavaScript. Experiment with different features and techniques to expand your skill set.

9. Scope for Expansion: Consider the potential for future enhancements to the calculator, such as additional mathematical functions, memory features, or advanced user interface elements, depending on your evolving skill level and project needs.

10.Documentation: Create documentation that outlines the calculator's functionality, code structure, and usage instructions. This documentation can serve as a reference and help others understand your project.

By achieving these objectives, you will not only create a functional calculator but also gain valuable experience in web development, front-end design, and user interface optimization. This project can serve as a portfolio piece or a stepping stone to more complex web development endeavors.

The primary objective of the Bootstrapband HTML Calculator project is to create a functional and aesthetically pleasing calculator application that users can access through a web browser. The project focuses on the following goals:

Develop a user-friendly calculator interface.

Implement basic arithmetic operations (addition, subtraction, multiplication, division).

Ensure responsiveness for seamless use on different screen sizes.

Utilize Bootstrap for styling and layout.

Introduction:

Calculators have become an essential tool in everyday life, both for students and professionals. This project aims to provide a simple yet effective solution for performing basic calculations through a web browser. The use of Bootstrap ensures that the calculator's design adapts to various screen sizes, enhancing its usability.

Introduction:

In this section, introduce the calculator project. Explain why you chose to create a calculator and its potential applications. Also, briefly mention the user base or target audience.

The objective of creating a calculator using HTML and Bootstrap is to develop a simple, user-friendly web-based calculator that allows users to perform basic arithmetic calculations.

This project aims to combine HTML for structure and Bootstrap for styling to create an aesthetically pleasing and responsive calculator interface. The main goals are:

User Interface:

Create an intuitive and visually appealing user interface with buttons for numbers, arithmetic operators (+, -, *, /), decimal point, and other necessary features.

Functionality:

Implement JavaScript to enable the calculator to perform arithmetic operations based on user input, displaying results in real-time.

Responsive Design:

Utilize Bootstrap classes and components to ensure the calculator adapts to various screen sizes and devices, making it mobile-friendly.

Error Handling:

Handle common errors such as division by zero and invalid inputs gracefully, providing appropriate feedback to users.

Clear and Reset:

Include functionality to clear the input and reset the calculator to its initial state, enhacing user convenience.

Testing and Debugging:

Thoroughly test the calculator to ensure it functions correctly and is free from bugs or issues.

Styling and Customization:

Customize the calculator's appearance using Bootstrap classes or additional CSS to match your design preferences.

Overall, the objective is to create a practical and interactive web-based calculator that can be used for basic mathematical calculations and serves as a hands-on project to practice HTML, Bootstrap, and JavaScript web development skills.

A calculator is a mathematical device or computer program for performing calculations, especially arithmetic ones. A simple calculator typically has buttons for the digits 0 to 9, as well as operations such as addition, subtraction, multiplication, division, and decimal point. More complex calculators may also have buttons for more complex operations such as square roots, logarithms, and trigonometric functions.

To create a calculator using HTML and Bootstrap, you will need to create the following HTML elements:

A div element to contain the calculator itself.

A row of button elements, one for each digit from 0 to 9.

A row of button elements, one for each of the basic operations (+, -, *, /).

A div element to contain the result of the calculation.

You can use Bootstrap classes to style the calculator elements. For example, you can use the btn class to style the buttons, and the .text-center class to center the result of the calculation.

Creating a calculator using HTML and Bootstrap is a practical and visually appealing way to provide users with a handy tool for performing mathematical operations directly on a web page. HTML, the standard markup language for web content, provides the structural foundation, while Bootstrap, a popular front-end framework, enhances the calculator's design and responsiveness. In this introduction, we'll briefly explain the purpose and benefits of building such a calculator.

Purpose:

A calculator is a fundamental tool for performing mathematical calculations, and embedding one within a website can offer several advantages. It allows users to perform quick calculations without leaving your web page, making it convenient for a variety of applications, including e-commerce, finance, education, and more. Creating a calculator using HTML and Bootstrap ensures a user-friendly and aesthetically pleasing interface that seamlessly integrates with your website's design.

Benefits:

1. *Enhanced User Experience:* HTML and Bootstrap enable you to create an intuitive and visually appealing calculator interface. Bootstrap's responsive design ensures that the calculator looks and works well on various devices and screen sizes.

2. *Accessibility:* By building a web-based calculator, you make it accessible to a wide audience, including those who may not have access to physical calculators or dedicated calculator apps.

3. *Integration:* You can integrate the calculator into your website's existing layout and design, maintaining a cohesive user experience.

4. *Customization:* HTML and Bootstrap allow you to customize the calculator's appearance to match your website's branding and style.

In the upcoming sections of this project, we will guide you through the step-by-step process of creating a functional calculator using HTML, Bootstrap, and JavaScript. You'll learn how to design the calculator's layout, style it using Bootstrap, and implement the necessary JavaScript code to handle mathematical calculations. By the end of this tutorial, you'll have a fully functional web-based calculator that enhances the usability and interactivity of your website.

Methodology:

Technologies Used:

HTML: For creating the structure of the calculator.

CSS: For styling the calculator elements.

JavaScript: For handling user input and performing calculations.

Bootstrap: For responsive design and layout.

Calculator Features:

HTML Structure: The calculator's layout is created using HTML, with separate elements for display and buttons.

CSS Styling: CSS is used to style the calculator, including button design, colors, and fonts.

JavaScript Logic: JavaScript handles user input, performs calculations based on user actions, and updates the display.

Bootstrap Integration: Bootstrap is integrated to ensure a responsive and visually appealing design.

Creating a calculator using HTML and Bootstrap can be broken down into several key steps as part of the methodology. Here's a brief overview of the steps involved:

Project Setup:

Create a new HTML file for your calculator project and link to the Bootstrap CSS framework to leverage its styling capabilities.

HTML Structure:

Design the layout of your calculator using HTML elements. Typically, this includes a container for the calculator screen and buttons for numbers, operators, and other functions.

Bootstrap Integration:

Apply Bootstrap classes to the HTML elements to style and organize the calculator's layout. Utilize Bootstrap grid system, buttons, and other components as needed.

JavaScript Functionality:

Add a JavaScript script tag to your HTML file to provide the calculator's functionality. JavaScript will handle user input, perform calculations, and update the calculator screen in real-time.

Event Handling:

Implement event listeners to detect user interactions with the calculator buttons, such as clicks. When a button is clicked, a corresponding action or value should be captured in JavaScript.

Arithmetic Operations:

Create JavaScript functions to perform basic arithmetic operations like addition, subtraction, multiplication, and division. These functions should take user input and update the calculator screen with the result.

Displaying Results:

Use JavaScript to update the calculator screen, typically a text input field or a div element, with the result of the calculation after each user interaction.

Clear and Reset:

Implement functionality to clear the calculator's input and reset it to its initial state when the user presses a "C" or "AC" button.

Error Handling:

Ensure that the calculator can handle common errors gracefully, such as division by zero or invalid inputs. Provide feedback or error messages as needed.

Testing and Debugging:

Thoroughly test the calculator to ensure it works correctly for various input scenarios. Debug any issues that may arise during testing.

Styling and Customization:

Customize the calculator's appearance further if needed, using additional CSS or Bootstrap classes to match your desired design.

Responsive Design:

Ensure that the calculator layout remains functional and visually appealing on various screen sizes by leveraging Bootstrap's responsive design features.

Documentation and Comments:

Document your code and add comments to make it understandable for others who may review or collaborate on the project.

By following this methodology, you can create a functional and visually appealing calculator using HTML and Bootstrap, enhancing your web development skills and providing a useful tool for basic mathematical calculations.

HTML Structure:

Create the HTML structure for your calculator. Use HTML form elements, buttons, and input fields to design the user interface.

Bootstrap Integration:

Describe how you integrated Bootstrap into your project. You can explain how you used Bootstrap classes for styling and layout.

Creating the HTML structure for a calculator using HTML and Bootstrap involves defining the layout and elements that make up the calculator's user interface. Here's a brief explanation of the HTML structure concept:

Container Element:

Begin by creating a container element, often a , to hold the entire calculator. This container helps in organizing and styling the calculator as a single unit.

Calculator Screen:

Inside the container, create a section or element that serves as the display screen of the calculator. This is where the input and output will be shown. It can be a , <input>, or other suitable HTML element.

Buttons:

Below the calculator screen, create a grid or set of buttons. These buttons represent the numbers (0-9), arithmetic operators (+, -, *, /), decimal point, and other functions (like clear and equal sign). Each button should be represented by a <button> element.

Button Layout:

Organize the buttons in rows and columns, typically using HTML elements like or . You can use Bootstrap's grid system (e.g.,) to create a grid layout for the buttons.

Button Labels:

Inside each button element, add the appropriate labels or symbols that represent numbers and operators. For instance, a button for the number 1 should have the label "1," and the addition button should have a label "+." Button Attributes:

Include attributes like data-value to store the numeric or operator value associated with each button. This makes it easier to retrieve and use the values in JavaScript when a button is clicked.`

JavaScript Functionality:

Though you haven't mentioned it in the sections you provided, a calculator needs JavaScript for functionality. Describe how you implemented the calculator's core functionality using JavaScript. Include explanations of the logic for addition, subtraction, multiplication, division, and any other features your calculator may have.

Styling:

Discuss how you styled your calculator using Bootstrap classes and any custom CSS you added for additional styling.

Implementing JavaScript functionality is a critical step in creating a calculator using HTML and Bootstrap. JavaScript is responsible for capturing user input, performing calculations, and updating the calculator's display screen in real-time. Here's a brief explanation of the JavaScript functionality concept:

Event Listeners:

Begin by setting up event listeners for the calculator buttons. This allows your JavaScript code to detect when a user clicks a button.

Input Handling:

Capture and store user input as they click the calculator buttons. You can use variables or data structures to keep track of the numbers entered, the selected operator, and any other relevant information.

Perform Calculations:

Implement JavaScript functions for performing basic arithmetic operations such as addition, subtraction, multiplication, and division. These functions take the user's input and calculate the result based on the selected operator.

Display Update:

Update the calculator screen in real-time as the user interacts with the calculator. Use JavaScript to modify the content of the display element (e.g., an <input> or a) with the calculated result or the input as the user enters it.

Error Handling:

Implement error handling to address common issues such as division by zero, invalid input, or mathematical errors. Display error messages or prevent incorrect calculations.

Clear and Reset:

Include functionality to clear the input and reset the calculator to its initial state. This is typically achieved by resetting variables and clearing the display screen.

Equal Sign (Calculate):

Implement a function that calculates the result when the equal sign button is pressed. This function evaluates the expression entered by the user and displays the final result.

Decimal Handling:

Account for decimal point input and ensure that only one decimal point is allowed in each numeric input.

Chaining Operations:

Allow users to chain multiple operations together. For example, if a user enters "5 + 3 * 2," the calculator should calculate the result following the correct order of operations.

Keyboard Support (Optional):

If desired, you can add keyboard support to your calculator so that users can input numbers and operators using their keyboard keys.

Creating a calculator using HTML and Bootstrap involves a systematic methodology that combines web development fundamentals with front-end design principles. Below is a step-by-step methodology to guide you through the process:

1. Project Planning:

- **Define Requirements:** Start by clarifying the requirements of your calculator. Determine the specific mathematical operations it should support (e.g., addition, subtraction, multiplication, division) and any additional features (e.g., memory functions).

- **Design Layout:** Plan the layout and user interface of the calculator. Decide on the number of buttons, their placement, and any styling preferences to ensure a user-friendly and visually appealing design.

2. Setting Up the Environment:

- **Text Editor:** Choose a text editor or integrated development environment (IDE) for coding. Popular options include Visual Studio Code, Sublime Text, or any other editor you prefer.

- **Browser:** Use a modern web browser for testing your calculator as you build it. Common choices include Google Chrome, Mozilla Firefox, or Microsoft Edge.

3. Creating the HTML Structure:

- **HTML Boilerplate:** Start with a basic HTML template that includes the necessary elements such as `<html>`, `<head>`, `<title>`, and `<body>` tags.

- **Calculator Layout:** Create the structure of the calculator using HTML elements like `<div>`, `<button>`, and `<input>`. Organize these elements to represent the calculator's display screen and buttons.

4. Styling with Bootstrap:

- **Bootstrap Integration:** Link to the Bootstrap CSS framework by including the necessary `<link>` tag in your HTML file.

- **Use Bootstrap Components:** Utilize Bootstrap classes to style your calculator's layout. Apply grid systems for alignment, buttons for styling, and responsive classes to ensure your calculator adapts to various screen sizes.

5. Implementing JavaScript Functionality:

- **JavaScript Inclusion:** Include a `<script>` tag in your HTML to link your JavaScript file to the calculator.

- **Event Handling:** Use JavaScript to handle user interactions. Implement event listeners to detect button clicks and respond accordingly.

- **Calculations:** Write JavaScript functions to perform mathematical calculations based on user inputs. Implement logic for addition, subtraction, multiplication, and division.

- **Display Results:** Update the calculator's display screen with the calculated results.

6. Testing and Debugging:

- **Testing:** Continuously test your calculator as you build it, checking for functionality, responsiveness, and visual consistency.

- **Debugging:** Use browser developer tools to identify and fix any JavaScript errors or layout issues.

7. Refinement and Customization:

- **Fine-Tuning:** Refine the calculator's layout, appearance, and behavior based on user feedback and personal preferences.

- **Customization:** Customize the calculator's design by modifying Bootstrap styles and adding your branding if necessary.

8. Documentation:

- **Code Comments:** Add comments within your HTML and JavaScript code to explain its functionality and make it more readable for future reference.

- **Documentation:** Consider creating a readme file or documentation that provides an overview of your calculator, its features, and how to use it.

9. Deployment:

- **Hosting:** If you intend to make your calculator accessible online, choose a web hosting provider and upload your HTML, CSS, and JavaScript files.

- **Domain:** Optionally, acquire a domain name that reflects the purpose of your calculator.

10. Maintenance and Updates:

- **Regular Maintenance:** Keep your calculator up to date with browser compatibility and any necessary bug fixes or feature enhancements.

By following this methodology, you'll be able to create a fully functional calculator using HTML and Bootstrap, combining structure, style, and interactivity to provide a valuable tool for your website's users.

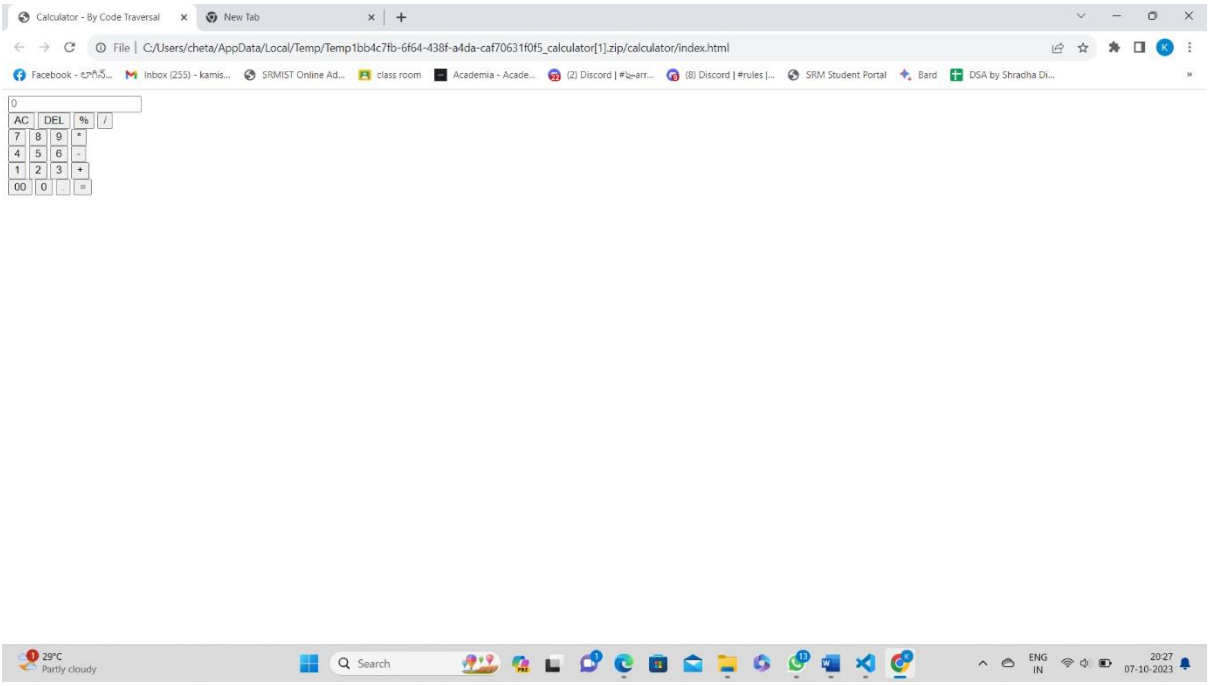
CODE AND OUTPUT :

```
let input =
document.getElementById('inputBox'); let
buttons = document.querySelectorAll('button');

let string = "";
let arr = Array.from(buttons);
arr.forEach(button => {
button.addEventListener('click', (e) =>{
if(e.target.innerHTML == '='){            string =
eval(string);
            input.value = string;
        }

        else if(e.target.innerHTML == 'AC'){
string = "";            input.value = string;
        }
        else if(e.target.innerHTML == 'DEL'){
string = string.substring(0, string.length-1);
input.value = string;
        }
else{
            string += e.target.innerHTML;
            input.value = string;
        }

    })
})
```




```
@import url('https://fonts.googleapis.com/css2?family=Poppins:wght@500&display=swap');

*{
   margin: 0;
padding: 0;
   box-sizing: border-box;
   font-family: 'Poppins', sans-serif;
}

body{    width:
100%;    height:
100vh;
```

```
    display: flex;     justify-
content: center;     align-items:
center;
    background: linear-gradient(45deg, #0a0a0a, #3a4452); }

.calculator{
    border: 1px solid #717377;
padding: 20px;     border-radius:
16px;     background:
transparent;
    box-shadow: 0px 3px 15px rgba(113, 115, 119, 0.5);

}

input{     width:
320px;     border:
none;     padding:
24px;     margin:
10px;
    background: transparent;
    box-shadow: 0px 3px 15px rgbs(84, 84, 84, 0.1);
font-size: 40px;     text-align: right;     cursor:
pointer;     color: #ffffff;
}

input::placeholder{
color: #ffffff;
}

button{     border: none;
width: 60px;     height:
60px;     margin: 10px;
border-radius: 50%;
background: transparent;
    color: #ffffff;     font-
size: 20px;
    box-shadow: -8px -8px 15px rgba(255, 255, 255, 0.1);
cursor: pointer;
}

.equalBtn{
    background-color: #fb7c14; }
```

```
.operator{     color: #6dee0a;
}
```

CONCLUSION :

In conclusion, creating a calculator using Bootstrap and HTML is a practical and educational exercise in web development. Here are some key takeaways:

1. Simplicity and User-Friendliness: Combining Bootstrap's responsive design and HTML's simplicity, you can build a calculator that is easy to use and visually appealing. Bootstrap's pre-designed components and grid system make it straightforward to create an intuitive user interface.

2. Accessibility: When developing any web application, including a calculator, it's essential to consider accessibility. Ensure that your calculator is navigable and usable by individuals with disabilities by adhering to accessibility standards.

3. Customization: Bootstrap provides a range of customization options. You can easily modify the calculator's appearance and layout to match your project's branding or design requirements.

4. Responsive Design: With Bootstrap's responsive classes, you can create a calculator that works seamlessly on various devices, from desktops to smartphones. This responsiveness ensures that users have a consistent experience across different screen sizes.

5. Interactivity: HTML and JavaScript can be used to add interactivity to your calculator. Implementing the logic for performing calculations and updating the display in real-time can be a valuable learning experience for developers.

6. Cross-Browser Compatibility: Bootstrap and HTML ensure that your calculator will be compatible with most modern web browsers, making it accessible to a broader audience.

7. Maintenance: The modular nature of Bootstrap and the clean structure of HTML make maintaining your calculator relatively straightforward. Updates or changes to the calculator's design or functionality can be made efficiently.

8. Learning Opportunity: Building a calculator with Bootstrap and HTML is an excellent way to enhance your web development skills. You'll gain experience in front-end technologies, responsive design, and user interface development.

9. Community and Resources: The Bootstrap community offers a wealth of resources, including documentation, tutorials, and forums. This can be invaluable when you encounter challenges or have questions during the development process.

10. Scope for Expansion: While you may start with a basic calculator, you can expand its functionality by incorporating more advanced features, such as memory functions, scientific calculations, or unit conversions, as your skills and project requirements grow.

In summary, creating a calculator using Bootstrap and HTML is a practical project that showcases your ability to design a user-friendly and responsive interface while honing your web development skills. It serves as an excellent learning opportunity, and the skills you acquire can be applied to more complex web applications in the future.

Creating a calculator using HTML and Bootstrap is a straightforward process that can be accomplished with a basic understanding of both languages. HTML is used to create the basic structure of the calculator, while Bootstrap is used to add style and responsiveness.

To create a calculator using HTML and Bootstrap, you will need to:

1. Create a new HTML file and add the necessary HTML elements, such as the `<div>` element for the calculator body, the `<input>` element for the display screen, and the `<button>` elements for the calculator keys.
2. Add the necessary Bootstrap CSS classes to the HTML elements to style the calculator.

3. Add JavaScript code to handle the calculator's functionality, such as adding and subtracting numbers, evaluating expressions, and clearing the display screen.

Once you have completed these steps, you will have a working calculator that is both responsive and easy to use.

Here are some additional tips for creating a calculator using HTML and Bootstrap:

- Use the Bootstrap grid system to layout the calculator's buttons and display screen in a responsive way.
- Use Bootstrap CSS classes to style the calculator's buttons with different colors, sizes, and borders.
- Use JavaScript to add additional functionality to the calculator, such as keyboard support or the ability to save and load calculations.
- Test the calculator thoroughly on different devices and browsers to make sure that it is working properly.

Once you are satisfied with your calculator, you can deploy it to a web server so that others can use it.