**Currency Converter Report**

**Overview**

The Currency Converter is a web application that allows users to convert amounts between different currencies using real-time exchange rates. The application is built with HTML, CSS, Bootstrap 5, and JavaScript, ensuring a responsive and user-friendly interface.

**Project Structure**

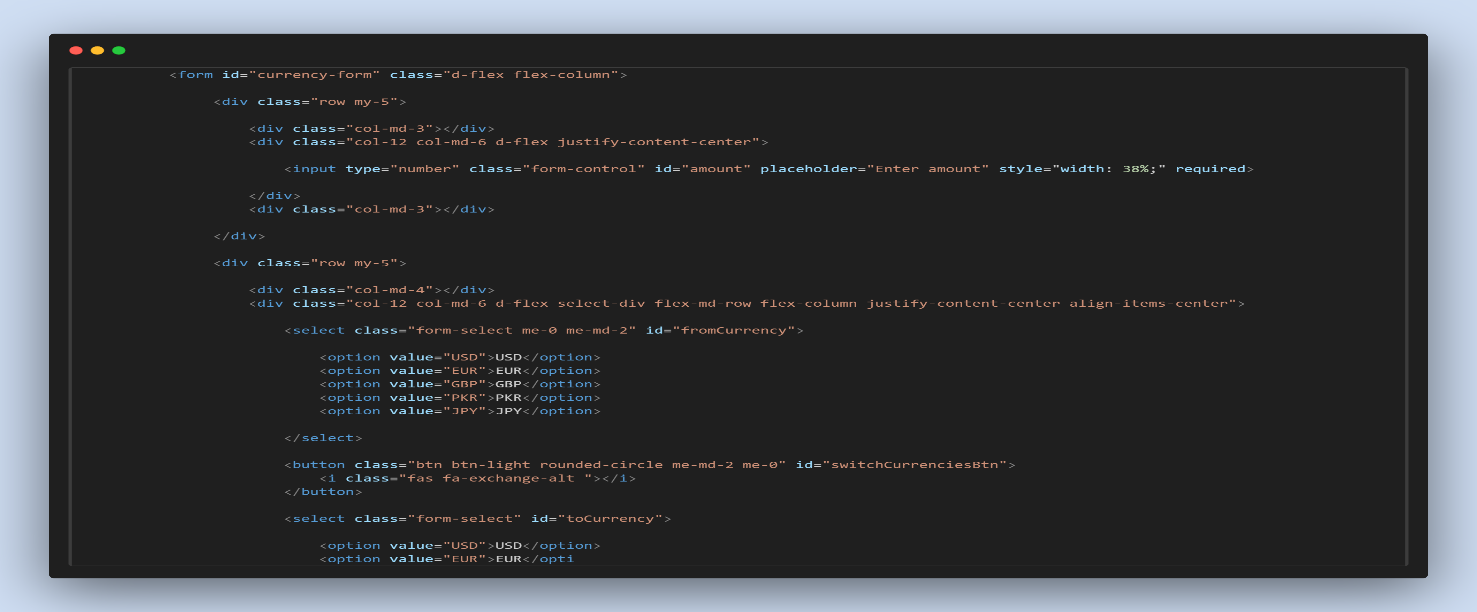
The project consists of three main files:

1. **index.html**: The main HTML file containing the structure of the application.
2. **style.css**: The CSS file for styling the application.
3. **script.js**: The JavaScript file responsible for the functionality of the converter.

**index.html**

This file includes the layout of the currency converter, using Bootstrap for styling and responsiveness. Key elements include:

1. A form for user input (amount, currencies).
2. Exchange rate display.
3. Convert button.





**script.js**

This JavaScript file manages the exchange rates and handles user interactions, such as:

* Updating the exchange rate based on selected currencies.
* Converting currency when the button is clicked.
* Switching currencies.

**Features**

* **Responsive Design**: The application adapts to different screen sizes using Bootstrap classes.
* **Dynamic Exchange Rates**: The application updates exchange rates based on user selection, ensuring accurate conversions.
* **User-Friendly Interface**: Clear input fields, buttons, and labels make it easy for users to navigate.

**Conclusion**

* The Currency Converter project successfully implements a functional and visually appealing currency conversion tool. It leverages modern web technologies to create an efficient user experience.



**Student Registration Form**

**Introduction**

This project implements a student registration form that validates user input through JavaScript. The form collects essential details such as name, address, gender, state, city, pincode, course, and email ID. It utilizes Bootstrap 5 for responsive design and styling.

**Technologies Used**

* HTML
* CSS
* Bootstrap 5
* JavaScript

**Features**

1. **Responsive Design:** The form adapts to various screen sizes using Bootstrap's grid system.
2. **Form Validation:**
   * Uses regex to validate names, pincode, email, and required fields.
   * Displays appropriate feedback messages for invalid inputs.
3. **Dynamic City Selection:** The city dropdown updates based on the selected state, providing relevant city options.

**Code Structure**

The project consists of three primary files:

* index.html: Contains the HTML structure of the form.
* style.css: Defines the custom styles for the form.
* script.js: Implements the validation logic and dynamic city selection.

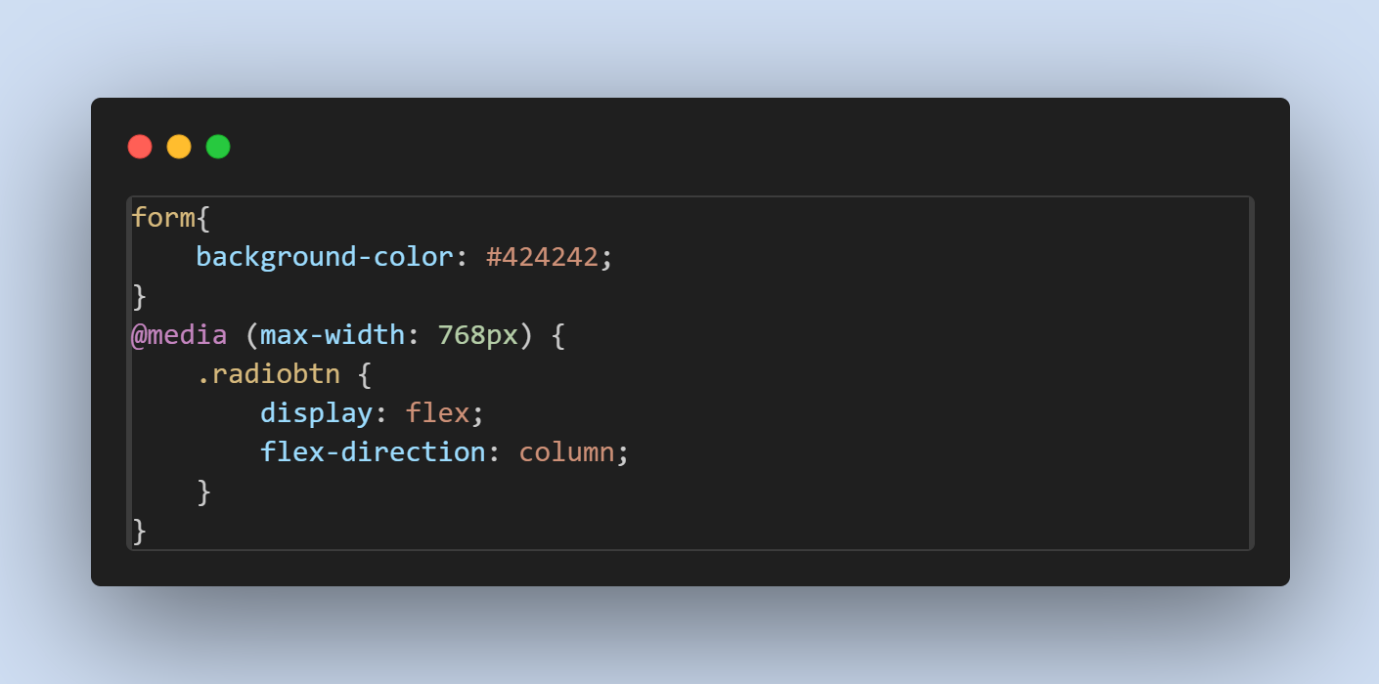
**HTML (index.html)**

The HTML structure includes a form with various input fields. The form employs Bootstrap classes for styling and layout.

****

**CSS (style.css)**

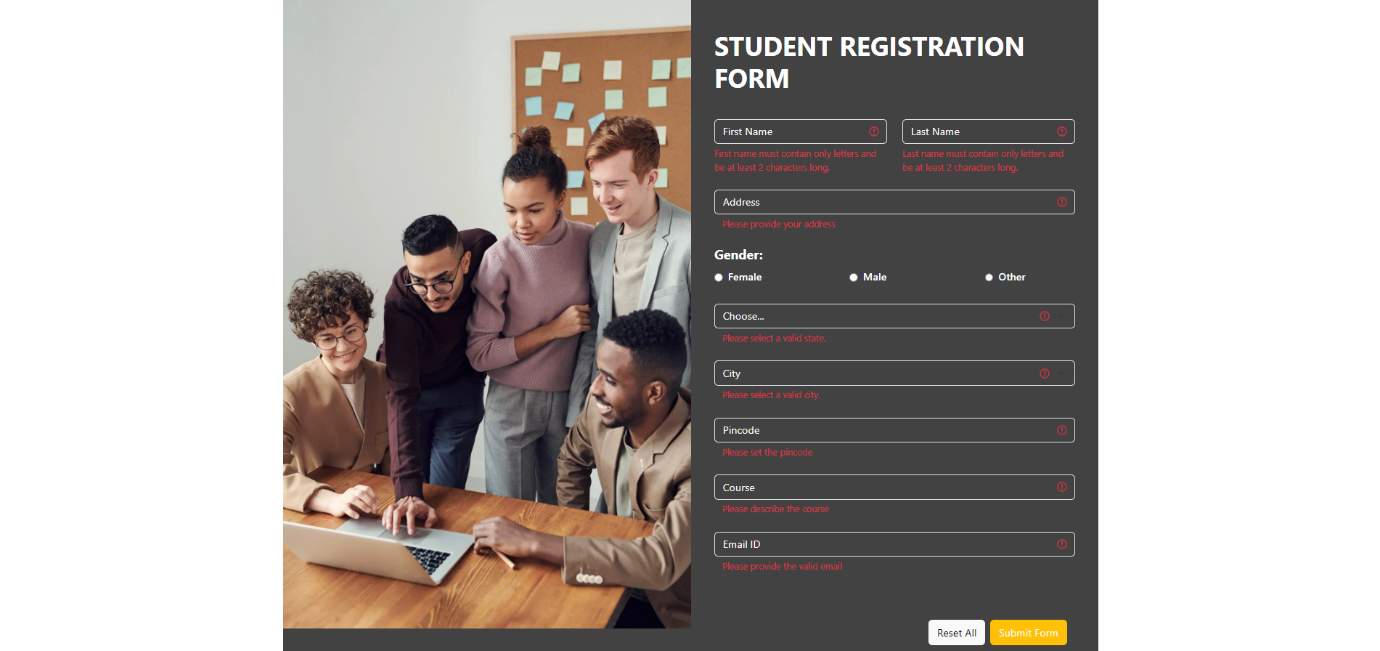
Custom styles are applied to enhance the form's appearance, such as background color and placeholder text color.

****

**JavaScript (script.js)**

The JavaScript code handles form validation and dynamic behavior for the city selection based on the state chosen by the user.





**Conclusion**

This student registration form project effectively demonstrates the integration of HTML, CSS, and JavaScript to create a user-friendly and responsive application. The validation mechanisms enhance user experience by providing instant feedback, ensuring data integrity upon submission.

**Calculator Project Report**

**Introduction**

This report outlines the design and functionality of a simple calculator web application built using HTML, CSS (Bootstrap), and JavaScript. The application supports basic arithmetic operations, including addition, subtraction, multiplication, and division, as well as advanced operations such as square root, square, and reciprocal.

**Features**

* **Basic Operations:** Users can perform addition, subtraction, multiplication, and division.
* **Advanced Operations:** Includes functionalities for square (x²), square root (√x), and reciprocal (1/x).
* **Display System:** A dual-display feature shows both the current input and the ongoing operation.
* **Clear Functions:** Options to clear the current entry (CE) and clear all inputs (C).
* **User Interface:** Utilizes Bootstrap for responsive design and improved aesthetics.

**HTML Structure**

The HTML document provides a structured layout for the calculator. The main components include:

* **Header:** Displays the title and icons for navigation.
* **Display Section:** Shows the current number and preview of the operation.
* **Button Groups:** Organized in sections for easy access to numbers and operations.

**JavaScript Functionality**

The JavaScript file contains the core functionality for the calculator. Key functions include:

* **Display Update:** The updateDisplay() function updates the displayed numbers and operations.
* **Number Input:** The appendNumber() function appends numbers to the current input.
* **Operator Handling:** Functions like addOperator() and calculate() manage the arithmetic operations.
* **Clear Functions:** Implemented with clearEntry() and clearAll(), allowing users to reset inputs easily.
* **Advanced Operations:** The operate() function handles special operations like square root and reciprocal.

**CSS Styling**

Custom styling is applied through the style.css file, enhancing the visual appearance of buttons and displays while utilizing Bootstrap for layout and responsiveness.

**Code Snippets**

Below are key snippets from the project:

**HTML Example**

<section class="constant">

<p id="preview" class="text-end text-white display-6 fw-normal"></p>

<p id="display" class="text-end text-white display-1 fw-bold">0</p>

</section>

**JavaScript Example**

function calculate() {

if (!previousInput) return; // Skip if no previous input

const prev = parseFloat(previousInput);

const current = parseFloat(currentInput) || prev; // Use the current input or fallback to previous input

switch (operator) {

case '+':

result = prev + current;

break;

case '-':

result = prev - current;

break;

case 'X':

result = prev \* current;

break;

case '÷':

result = prev / current;

break;

default:

return;

}

currentInput = result.toString(); // Display the result as the new current input

previousInput = ''; // Clear previous input after calculation

operator = ''; // Clear the operator

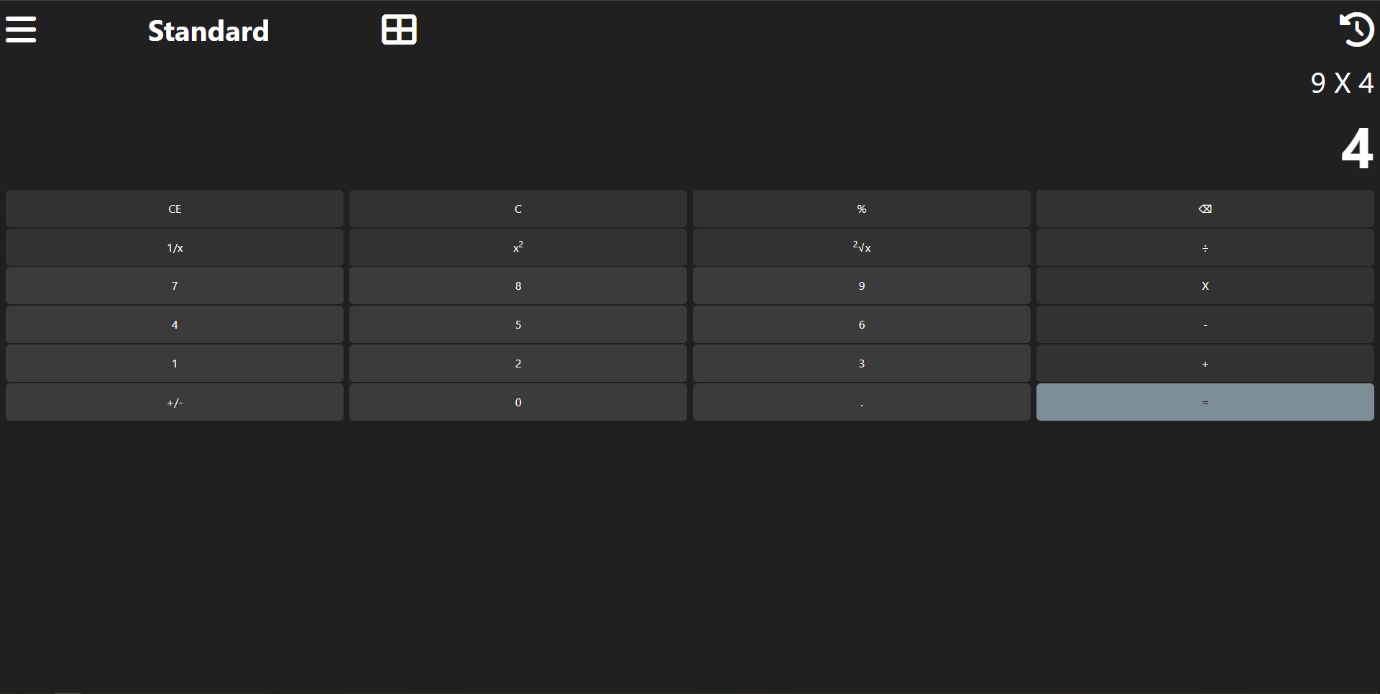
isCalculated = true; // Mark that the calculation is complete

updateDisplay();

}

**Conclusion**

The calculator project effectively combines essential mathematical operations with a user-friendly interface. It serves as a practical tool for users needing quick calculations while also demonstrating the application of web development skills in HTML, CSS, and JavaScript.



**Google Drive**

https://drive.google.com/drive/folders/1tK0O25hK0\_1WUQWrU1xrOWUMHFj5pmbb