**Primer Proyecto Práctico.**

**1ª Evaluación:**

**Construcción de una Calculadora Interactiva Básica**

**ÍNDICE.**

[**Descripción General:** 3](#_Toc180088270)

[**Especificación de Requisitos.** 3](#_Toc180088271)

[1. Requisitos Funcionales: 3](#_Toc180088272)

[2. Requisitos de Interfaz de Usuario: 3](#_Toc180088273)

[3. Requisitos de Sistema: 4](#_Toc180088274)

[4. Requisitos de Seguridad: 4](#_Toc180088275)

[**Análisis.** 4](#_Toc180088276)

[Diagrama de Casos de Uso: 4](#_Toc180088277)

[**Diseño.** 5](#_Toc180088278)

[Diseño de la Funcionalidad: 5](#_Toc180088279)

[**Entrega Final:** 6](#_Toc180088280)

# **Descripción General:**

El objetivo de este proyecto es que desarrolles una calculadora interactiva básica utilizando JavaScript, siguiendo las buenas prácticas de programación en entornos cliente. Este proyecto cubre tanto el desarrollo técnico como el análisis y diseño de la solución.

Este proyecto evalúa los criterios de evaluación de los R.A.1, R.A.2, R.A.3 y R.A.4. Cada criterio de evaluación se puntúa de 0,00 a 10,00.

Tienes que entregar el proyecto en formato comprimido \*.zip/\*.rar y subirlo a la tarea del Moddle.

# **Especificación de Requisitos.**

## 1. Requisitos Funcionales:

1. **Operaciones básicas**: La calculadora debe permitir realizar las siguientes operaciones matemáticas:
   * Suma
   * Resta
   * Multiplicación
   * División
   * Raíz cuadrada de un número.
2. **Control de errores**:
   * La calculadora debe manejar errores comunes, como la división por cero, mostrando un mensaje adecuado.
3. **Historial de operaciones**: La calculadora debe almacenar los resultados de las operaciones realizadas en un array y permitir que el usuario vea este historial.
4. **Repetición de operaciones**: El usuario debe poder repetir operaciones sin necesidad de recargar la página.

## 2. Requisitos de Interfaz de Usuario:

1. **Interfaz sencilla**: La calculadora no requiere interfaz gráfica avanzada. Los resultados y las operaciones pueden mostrarse utilizando la consola del navegador.
2. **Input básico**: Los números se introducirán en el código como parámetros o mediante prompt, y los resultados se visualizarán en la consola del navegador.
3. **Menú interactivo**: Los usuarios deben poder seleccionar la operación que desean realizar (suma, resta, etc.) a través de un menú textual.

## 3. Requisitos de Sistema:

1. **Entorno de ejecución**: El proyecto debe ser compatible con cualquier navegador moderno (Chrome, Firefox, Edge, Safari), utilizando solo HTML y JavaScript.
2. **Independencia de servidor**: Todo el código se ejecuta en el cliente, sin necesidad de conexión a un servidor.

## 4. Requisitos de Seguridad:

1. **Validación básica de inputs**: Aunque el input se proporciona en la consola o a través de prompt, es importante asegurarse de que se ingresen números válidos y no cadenas de texto.
2. **Manejo de excepciones**: Debe haber un manejo adecuado de excepciones, especialmente en operaciones como la división por cero o inputs no numéricos.

# **Análisis.**

## Diagrama de Casos de Uso:

**Actores:**

1. **Usuario**: Persona que interactúa con la calculadora introduciendo los números y eligiendo las operaciones a realizar.

**Casos de Uso:**

1. **Realizar operación**: El usuario ingresa dos números y selecciona una operación (suma, resta, multiplicación, división, raíz cuadrada).
2. **Manejo de errores**: El sistema valida que los números sean correctos y que no haya intentos de división por cero.
3. **Historial de operaciones**: El sistema guarda cada operación en un array para que el usuario pueda consultar los resultados previos.
4. **Mostrar historial**: El usuario puede ver el historial de todas las operaciones realizadas durante la sesión.

**Diagrama (explicado):**

* **Caso de uso principal**: "Realizar operación" con el actor **Usuario**.
* **Extensiones**: Manejo de errores como la división por cero o ingreso de datos inválidos.
* **Incluye**: Guardar el resultado de cada operación en el historial.

Puedes dibujar un diagrama de casos de uso sencillo utilizando herramientas como Lucidchart, draw.io, o simplemente con un gráfico a mano.

# **Diseño.**

## Diseño de la Funcionalidad:

**1. Flujo de trabajo de la calculadora:**

* El usuario ingresa dos números.
* Elige la operación a realizar.
* El sistema realiza la operación.
* Si es válida, se muestra el resultado en la consola y se guarda en el historial.
* Si hay un error (por ejemplo, división por cero), se muestra un mensaje de error y no se guarda en el historial.

**2. Estructura del código:**

* **Funciones principales**:
  + calculadora(): Función principal que solicita los números y realiza la operación.
  + operacion(num1, num2, tipo): Realiza la operación según el tipo seleccionado (suma, resta, multiplicación, etc.).
  + mostrarHistorial(): Muestra en consola todos los resultados almacenados en el array historial.
  + validarInput(): Valida que los inputs sean números y que no haya errores como división por cero.

**3. Estructura de Datos:**

* **Array** historial[]: Almacena los resultados de las operaciones.
* **Variables**:
  + num1 y num2: Almacenan los números ingresados por el usuario.
  + resultado: Almacena el resultado de la operación actual.

**4. Diseño Modular:**

* Se recomienda que el código esté dividido en funciones bien definidas:
  + Una función para cada operación (suma, resta, etc.).
  + Una función para manejar errores.
  + Una función que gestiona el historial de operaciones.

# **Entrega Final:**

El proyecto completo deberá alojarse en un repositorio de GitHub del estudiante. Para la entrega de esta tarea en Moddle, será necesario proporcionar el enlace a dicho repositorio.

El repositorio deberá incluir, como mínimo, lo siguiente:

1. **Código funcional:** Los estudiantes deben entregar una aplicación web que contenga al menos los siguientes archivos: .html, .css, y .js.
2. **Diagrama de Casos de Uso:** Se debe incluir un diagrama que ilustre los principales casos de uso de la aplicación.
3. **Informe breve:** Un documento conciso en el que se expliquen las decisiones de diseño tomadas y los desafíos encontrados durante la implementación del proyecto.
4. **Archivo README.md:** Un archivo README en el que se detalle la funcionalidad de la aplicación.