**PRUEBA UNO TALLER APLICACIONES DE INTERNET**

**PROFESOR: IVAN AYALA AYALA**

**FECHA : 2025**

**NOMBRES: Lissette Macias; Cristobal Barrios; Roger Rodriguez.**

**Interfaz de usuario gráfica, Aplicación, Tabla

El contenido generado por IA puede ser incorrecto.1 .- Crear código html/css/javascript para validación de un formulario de registro de clientes; nombre,run,edad; 40 Ptos, véase rubrica para código apartado Html/Javascript, elementos 1;2;6;7 y 9.**

**HTML**

<!DOCTYPE html> *<!-- Define que el documento está en formato HTML5 -->*

<html lang="es"> *<!-- Indica que el idioma principal del documento es español -->*

<head>

<meta charset="UTF-8"> *<!-- Establece la codificación de caracteres como UTF-8 -->*

<meta name="viewport" content="width=device-width, initial-scale=1.0"> *<!-- Hace que la página sea responsive en dispositivos móviles -->*

<link rel="stylesheet" href="./asset/styles/styles.css"> *<!-- Vincula la hoja de estilos CSS externa -->*

<title>pregunta 1</title> *<!-- Título que aparece en la pestaña del navegador -->*

</head>

<body>

*<!-- Sección principal del formulario -->*

<section id="form">

<h1>Formulario Datos Personales</h1> *<!-- Título principal del formulario -->*

<form> *<!-- Inicio del formulario -->*

*<!-- Campo para el nombre -->*

<div>

<label for="nombre">Nombre:</label> *<!-- Etiqueta asociada al input de nombre -->*

<input id="nombre" placeholder="Ingrese su nombre" type="text" maxlength="30" required> *<!-- Input de texto para el nombre, obligatorio, con máximo de 30 caracteres -->*

</div>

*<!-- Campo para el apellido -->*

<div>

<label for="apellido">Apellido:</label> *<!-- Etiqueta asociada al input de apellido -->*

<input id="apellido" placeholder="Ingrese su apellido" type="text" maxlength="30" required> *<!-- Input de texto para el apellido, obligatorio, con máximo de 30 caracteres -->*

</div>

*<!-- Campo para el RUT chileno -->*

<div>

<label for="rut">RUT:</label> *<!-- Etiqueta asociada al input de RUT -->*

<input

id="rut"

type="text"

placeholder="Ingrese su RUT"

maxlength="10"

required <!-- Campo obligatorio -->

oninput="verificarRut()" *<!-- Llama a la función JS verificarRut() cada vez que el usuario escribe -->*

pattern="[0-9]{7,8}[\-]?[0-9Kk]{1}" *<!-- Patrón de validación que acepta 7 u 8 dígitos, un guion opcional y un dígito verificador (número o K/k) -->*

>

<small id="mensajeRut" class="ayuda">(Ej: 12345678-9)</small> *<!-- Texto de ayuda para mostrar un ejemplo válido de RUT -->*

</div>

*<!-- Campo para la edad -->*

<div>

<label for="edad">Edad:</label> *<!-- Etiqueta asociada al input de edad -->*

<input

id="edad"

type="number"

placeholder="Ingrese su edad"

min="1" max="99"

required <!-- Campo obligatorio con límites de edad entre 1 y 99 -->

>

</div>

*<!-- Botón para guardar los datos -->*

<button id="btn-guardar" type="button">Guardar</button> *<!-- Botón de tipo "button" que no envía el formulario, ideal si se usará JS -->*

</form> *<!-- Fin del formulario -->*

</section>

*<!-- Scripts JavaScript -->*

<script src="./js/validarRut.js"></script> *<!-- Archivo externo para validar el RUT -->*

<script src="./js/saludar.js"></script> *<!-- Archivo externo para mostrar un saludo u otra lógica -->*

</body>

</html>

**./js/validarRut.js**

*// Función que valida si un RUT chileno es correcto*

function validarRut(*rutCompleto*) {

*// Quita puntos y guiones, y convierte todo a mayúsculas*

*rutCompleto* = *rutCompleto*.replace(/\./g, '').replace(/-/g, '').toUpperCase();

*// Si el largo es menor a 2, es inválido (mínimo 1 número + 1 dígito verificador)*

if (*rutCompleto*.length < 2) return false;

*// Separa el cuerpo del RUT (sin el dígito verificador)*

const rut = *rutCompleto*.slice(0, -1);

*// Separa el dígito verificador (último carácter)*

const dv = *rutCompleto*.slice(-1).toLowerCase();

let suma = 0;

let multiplo = 2;

*// Recorre el RUT desde el final hacia el inicio*

for (let i = rut.length - 1; i >= 0; i--) {

suma += parseInt(rut.charAt(i)) \* multiplo; *// Multiplica cada dígito por el multiplicador*

multiplo = multiplo === 7 ? 2 : multiplo + 1; *// Cicla el multiplicador entre 2 y 7*

}

*// Calcula el dígito verificador esperado usando el algoritmo del módulo 11*

const dvEsperado = 11 - (suma % 11);

let dvCalc = '';

if (dvEsperado === 11) dvCalc = '0';

else if (dvEsperado === 10) dvCalc = 'k';

else dvCalc = dvEsperado.toString();

*// Compara el dígito ingresado con el calculado y retorna true o false*

return dv === dvCalc;

}

*// Función que muestra mensaje visual sobre la validez del RUT mientras el usuario escribe*

function verificarRut() {

const rutInput = document.getElementById('rut'); *// Obtiene el input del RUT*

const mensaje = document.getElementById('mensajeRut'); *// Mensaje debajo del input*

const valor = rutInput.value.trim(); *// Valor del RUT sin espacios*

if (valor.length >= 7) { *// Solo valida si hay suficientes caracteres*

if (!validarRut(valor)) {

mensaje.textContent = 'RUT inválido'; *// Mensaje en rojo si es incorrecto*

mensaje.style.color = 'red';

} else {

mensaje.textContent = 'RUT válido'; *// Mensaje en verde si es correcto*

mensaje.style.color = 'green';

}

} else {

mensaje.textContent = '(Ej: 12345678-9)'; *// Mensaje neutro por defecto*

mensaje.style.color = '#888';

}

}

*// Restringe las teclas permitidas en el input del RUT (solo números, k/K y guion)*

document.getElementById('rut').addEventListener('keypress', function (*e*) {

const key = e.key;

const rutValido = /^[0-9kK-]$/;

if (!rutValido.test(key)) {

e.preventDefault(); *// Bloquea cualquier otro carácter*

}

});

**./js/saludar.js**

*// Escucha el clic del botón con id "btn-guardar"*

document.getElementById("btn-guardar").addEventListener("click", function () {

*// Obtiene y limpia (quita espacios) los valores de los campos del formulario*

const nombre = document.getElementById("nombre").value.trim();

const apellido = document.getElementById("apellido").value.trim();

const rut = document.getElementById("rut").value.trim();

const edad = document.getElementById("edad").value.trim();

*// Verifica si algún campo está vacío*

if (!nombre || !apellido || !rut || !edad) {

alert("Por favor, completa todos los campos antes de continuar.");

return; *// Detiene la ejecución si falta algún dato*

}

*// Verifica si el RUT es válido usando una función externa llamada validarRut()*

if (!validarRut(rut)) {

alert("El RUT ingresado no es válido.");

return; *// Detiene si el RUT es inválido*

}

*// Convierte la edad a número y valida que esté entre 1 y 99*

const edadNumerica = parseInt(edad);

if (isNaN(edadNumerica) || edadNumerica < 1 || edadNumerica > 99) {

alert("Por favor, ingresa una edad válida entre 1 y 99.");

return; *// Detiene si la edad no es válida*

}

*// Valida que el nombre tenga entre 4 y 20 caracteres*

if (nombre.length < 4 || nombre.length > 20) {

alert("El nombre debe tener entre 4 y 20 caracteres.");

return;

}

*// Valida que el apellido tenga entre 4 y 20 caracteres*

if (apellido.length < 4 || apellido.length > 20) {

alert("El apellido debe tener entre 4 y 20 caracteres.");

return;

}

*// Si todo está correcto, muestra un saludo con los datos ingresados*

alert(`¡Hola ${nombre} ${apellido}!\n\nTus datos son:\nRUT: ${rut}\nEdad: ${edad}`);

});

**./asset/styles/styles.css**

*/\* Estilos generales del body (fondo y centrado de contenido) \*/*

body {

background-color: #e0e1e3; */\* Color de fondo gris claro \*/*

display: flex; */\* Usa flexbox para centrar contenido \*/*

justify-content: center; */\* Centra horizontalmente \*/*

align-items: center; */\* Centra verticalmente \*/*

height: 100vh; */\* Altura completa de la ventana del navegador \*/*

margin: 0; */\* Elimina margen predeterminado del navegador \*/*

font-family: Arial, sans-serif; */\* Fuente general del sitio \*/*

}

*/\* Estilos para el contenedor del formulario \*/*

#form {

background-color: white; */\* Fondo blanco para el formulario \*/*

padding: 30px 40px; */\* Espaciado interno (vertical y horizontal) \*/*

border-radius: 10px; */\* Bordes redondeados \*/*

border: 1px solid #ccc; */\* Borde gris claro \*/*

box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1); */\* Sombra suave alrededor \*/*

width: 100%; */\* Ocupa todo el ancho disponible \*/*

max-width: 500px; */\* Ancho máximo para mantenerlo compacto \*/*

}

*/\* Estilo del título del formulario \*/*

h1 {

text-align: center; */\* Centra el texto \*/*

margin-bottom: 30px; */\* Espacio inferior \*/*

color: #333; */\* Color gris oscuro \*/*

}

*/\* Estilo del formulario y distribución vertical \*/*

form {

display: flex; */\* Flexbox para organizar campos \*/*

flex-direction: column; */\* Dirección vertical (de arriba hacia abajo) \*/*

gap: 20px; */\* Espacio entre los elementos del formulario \*/*

}

*/\* Estilo general de los inputs \*/*

input {

padding: 10px; */\* Espaciado interno \*/*

border-radius: 6px; */\* Bordes redondeados \*/*

border: 1px solid #bbb; */\* Borde gris claro \*/*

font-size: 14px; */\* Tamaño de texto \*/*

width: 100%; */\* Ocupa el ancho completo del contenedor \*/*

box-sizing: border-box; */\* Incluye padding y border en el ancho total \*/*

}

*/\* Estilo de las etiquetas (labels) \*/*

label {

margin-bottom: 5px; */\* Pequeño espacio debajo del label \*/*

font-weight: bold; */\* Texto en negrita \*/*

color: #444; */\* Gris medio \*/*

}

*/\* Estilo del texto de ayuda debajo del RUT \*/*

.ayuda {

font-size: 12px; */\* Letra más pequeña \*/*

color: #888; */\* Gris claro \*/*

margin-top: 4px; */\* Espacio superior respecto al input \*/*

display: block; */\* Hace que se muestre debajo (nueva línea) \*/*

}

*/\* Estilo del botón "Guardar" \*/*

#btn-guardar {

background-color: #0077cc; */\* Azul fuerte \*/*

color: white; */\* Texto blanco \*/*

padding: 12px; */\* Espaciado interno \*/*

font-size: 15px; */\* Tamaño del texto \*/*

border: none; */\* Sin borde \*/*

border-radius: 6px; */\* Bordes redondeados \*/*

cursor: pointer; */\* Cursor tipo "mano" al pasar encima \*/*

transition: background-color 0.3s ease; */\* Transición suave al cambiar color \*/*

}

*/\* Estilo del botón cuando el mouse pasa encima \*/*

#btn-guardar:hover {

background-color: #005fa3; */\* Azul más oscuro al hacer hover \*/*

}

**Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.2.- Registre localmente/base de Datos la información generada en pregunta uno; 40 Ptos, véase rubrica para código apartado Html/Javascript elementos 1;2;6;7 y 9.**

**HTML**

<!DOCTYPE html> *<!-- Indica que el documento está en HTML5 -->*

<html lang="es"> *<!-- Define el idioma principal del contenido como español -->*

<head>

<meta charset="UTF-8"> *<!-- Usa codificación UTF-8 para que se vean bien los caracteres especiales -->*

<meta name="viewport" content="width=device-width, initial-scale=1.0"> *<!-- Ajusta la vista para dispositivos móviles -->*

<link rel="stylesheet" href="./asset/styles/styles.css"> *<!-- Conecta la hoja de estilos CSS externa -->*

<title>pregunta 2</title> *<!-- Título que se mostrará en la pestaña del navegador -->*

</head>

<body> *<!-- Inicio del cuerpo visible de la página -->*

*<!-- Sección principal con el formulario -->*

<section id="form">

<h1>Formulario Datos Personales</h1> *<!-- Título del formulario -->*

<form> *<!-- Formulario para ingresar datos -->*

*<!-- Campo para el nombre -->*

<div>

<label for="nombre">Nombre:</label> *<!-- Etiqueta asociada al campo nombre -->*

<input id="nombre" placeholder="Ingrese su nombre" type="text" maxlength="30" required> *<!-- Campo de texto obligatorio, hasta 30 caracteres -->*

</div>

*<!-- Campo para el apellido -->*

<div>

<label for="apellido">Apellido:</label>

<input id="apellido" placeholder="Ingrese su apellido" type="text" maxlength="30" required>

</div>

*<!-- Campo para el RUT -->*

<div>

<label for="rut">RUT:</label>

<input

id="rut"

type="text"

placeholder="Ingrese su RUT"

maxlength="10"

required

oninput="verificarRut()" <!-- Llama a una función JS mientras se escribe el RUT -->

pattern="[0-9]{7,8}[\-]?[0-9Kk]{1}" *<!-- Patrón de validación para el RUT chileno -->*

>

<small id="mensajeRut" class="ayuda">(Ej: 12345678-9)</small> *<!-- Mensaje de ayuda debajo del campo -->*

</div>

*<!-- Campo para la edad -->*

<div>

<label for="edad">Edad:</label>

<input id="edad" type="number" placeholder="Ingrese su edad" min="1" max="99" required>

</div>

*<!-- Botón para guardar los datos -->*

<button id="btn-guardar" type="button">Guardar</button>

</form>

</section>

*<!-- Nueva sección vacía donde se mostrarán los datos ingresados -->*

<section id="cajadatos"></section>

*<!-- Scripts que validan el RUT y guardan los datos -->*

<script src="./js/validarRut.js"></script> *<!-- Script que valida el formato del RUT -->*

<script src="./js/guardarUsuarios.js"></script> *<!-- Script personalizado para guardar y mostrar los datos ingresados -->*

</body>

</html>

./js/validarRut.js

*// Función que valida si un RUT chileno es correcto según su dígito verificador*

function validarRut(*rutCompleto*) {

*// Quitamos puntos y guiones y pasamos todo a mayúscula*

*rutCompleto* = *rutCompleto*.replace(/\./g, '').replace(/-/g, '').toUpperCase();

*// Si tiene menos de 2 caracteres, no es válido (falta RUT o dígito verificador)*

if (*rutCompleto*.length < 2) return false;

*// Separar el RUT numérico del dígito verificador*

const rut = *rutCompleto*.slice(0, -1); *// todos los dígitos menos el último*

const dv = *rutCompleto*.slice(-1).toLowerCase(); *// último carácter convertido a minúscula*

*// Inicializamos la suma y el multiplicador (según regla del cálculo)*

let suma = 0;

let multiplo = 2;

*// Recorremos el RUT de derecha a izquierda*

for (let i = rut.length - 1; i >= 0; i--) {

suma += parseInt(rut.charAt(i)) \* multiplo;

multiplo = multiplo === 7 ? 2 : multiplo + 1; *// ciclo de 2 a 7*

}

*// Cálculo del dígito verificador esperado*

const dvEsperado = 11 - (suma % 11);

let dvCalc = '';

if (dvEsperado === 11) dvCalc = '0';

else if (dvEsperado === 10) dvCalc = 'k';

else dvCalc = dvEsperado.toString();

*// Retornamos true si el dígito ingresado coincide con el calculado*

return dv === dvCalc;

}

*// Función que se ejecuta cada vez que el usuario escribe en el campo RUT*

function verificarRut() {

const rutInput = document.getElementById('rut'); *// Input del RUT*

const mensaje = document.getElementById('mensajeRut'); *// Elemento donde mostramos mensaje*

const valor = rutInput.value.trim(); *// eliminamos espacios innecesarios*

*// Cuando el RUT tiene 7 o más caracteres, lo validamos*

if (valor.length >= 7) {

if (!validarRut(valor)) {

mensaje.textContent = 'RUT inválido';

mensaje.style.color = 'red';

} else {

mensaje.textContent = 'RUT válido';

mensaje.style.color = 'green';

}

} else {

*// Si es muy corto, mostramos ayuda básica*

mensaje.textContent = '(Ej: 12345678-9)';

mensaje.style.color = '#888';

}

}

**./js/guardarRut.js**

*// Selecciona el formulario en el HTML*

const form = document.querySelector('form');

*// Crea un elemento <tbody> que luego usaremos para llenar la tabla*

const tableBody = document.createElement('tbody');

*// Cuando la página cargue completamente, llamamos a la función que muestra los datos guardados*

window.addEventListener('DOMContentLoaded', () => {

mostrarDatos();

});

*// Prevenimos que el formulario recargue la página por defecto al presionar Enter*

form.addEventListener('submit', (*e*) => {

*e*.preventDefault();

});

*// Evento que se ejecuta cuando se hace clic en el botón "Guardar"*

document.getElementById("btn-guardar").addEventListener("click", function () {

*// Capturamos el contenido de cada input del formulario*

const nombre = document.getElementById("nombre").value.trim();

const apellido = document.getElementById("apellido").value.trim();

const rut = document.getElementById("rut").value.trim();

const edad = document.getElementById("edad").value.trim();

*// Validamos que todos los campos estén llenos*

if (!nombre || !apellido || !rut || !edad) {

alert("Por favor, completa todos los campos.");

return;

}

*// Validamos el RUT usando la función que ya definimos*

if (!validarRut(rut)) {

alert("El RUT ingresado no es válido.");

return;

}

*// Mostramos un mensaje personalizado de saludo si todo es válido*

alert(`¡Hola ${nombre} ${apellido}!\n\nTus datos se han guardado correctamente.`);

*// Obtenemos los datos actuales desde localStorage (si no hay nada, usamos un arreglo vacío)*

const data = JSON.parse(localStorage.getItem("usuarios")) || [];

*// Creamos un objeto nuevo con los datos ingresados*

const nuevoUsuario = {

nombre: nombre,

apellido: apellido,

rut: rut,

edad: edad

};

*// Agregamos el nuevo objeto al arreglo de usuarios*

data.push(nuevoUsuario);

*// Guardamos el arreglo actualizado en localStorage*

localStorage.setItem("usuarios", JSON.stringify(data));

*// Limpiamos el formulario*

form.reset();

*// Volvemos a mostrar los datos actualizados en pantalla*

mostrarDatos();

});

*// Esta función muestra todos los datos guardados en localStorage en una tabla*

function mostrarDatos() {

const datos = JSON.parse(localStorage.getItem("usuarios")) || [];

const contenedor = document.getElementById("cajadatos");

*// Creamos la estructura básica de la tabla con sus encabezados*

contenedor.innerHTML = `

<h2>Datos Almacenados</h2>

<table>

<thead>

<tr>

<th>Nombre</th>

<th>Apellido</th>

<th>RUT</th>

<th>Edad</th>

<th>Acciones</th>

</tr>

</thead>

</table>

`;

*// Creamos el cuerpo de la tabla donde irán los datos*

const table = contenedor.querySelector("table");

const tbody = document.createElement("tbody");

*// Recorremos cada usuario guardado y generamos una fila por cada uno*

datos.forEach((*usuario*, *index*) => {

const row = document.createElement("tr");

*// Insertamos los datos y botones de acción (Editar y Eliminar)*

row.innerHTML = `

<td>${usuario.nombre}</td>

<td>${usuario.apellido}</td>

<td>${usuario.rut}</td>

<td>${usuario.edad}</td>

<td>

<button class="btnEdit" onclick="editarUsuario(${index})">Editar</button>

<button class="btnDelete" onclick="eliminarUsuario(${index})">Eliminar</button>

</td>

`;

tbody.appendChild(row);

});

*// Agregamos el <tbody> completo a la tabla*

table.appendChild(tbody);

}

*// Esta función elimina un usuario del arreglo y actualiza localStorage*

function eliminarUsuario(*index*) {

const data = JSON.parse(localStorage.getItem("usuarios")) || [];

*// Quitamos el usuario correspondiente al índice recibido*

data.splice(index, 1);

*// Guardamos el nuevo arreglo sin ese usuario*

localStorage.setItem("usuarios", JSON.stringify(data));

*// Volvemos a mostrar la tabla actualizada*

mostrarDatos();

}

*// Esta función precarga los datos del usuario seleccionado para editarlos*

function editarUsuario(*index*) {

const data = JSON.parse(localStorage.getItem("usuarios")) || [];

const usuario = data[index];

*// Rellenamos los campos del formulario con los datos del usuario*

document.getElementById("nombre").value = usuario.nombre;

document.getElementById("apellido").value = usuario.apellido;

document.getElementById("rut").value = usuario.rut;

document.getElementById("edad").value = usuario.edad;

*// Eliminamos el usuario original para que al guardar se reemplace con los nuevos datos*

eliminarUsuario(index);

}

**./asset/styles/style.css**

*/\* Estilos generales para el cuerpo de la página \*/*

body {

background-color: #e0e1e3; */\* Color de fondo gris claro \*/*

display: flex; */\* Usa flexbox para organizar el contenido \*/*

flex-direction: column; */\* Organiza los elementos en columna (de arriba hacia abajo) \*/*

align-items: center; */\* Centra el contenido horizontalmente \*/*

height: auto; */\* Altura automática (se adapta al contenido) \*/*

margin: 0; */\* Elimina el margen por defecto del navegador \*/*

font-family: Arial, sans-serif; */\* Fuente usada en todo el sitio \*/*

padding: 40px 20px; */\* Espacio alrededor del contenido (arriba/abajo y a los lados) \*/*

}

*/\* Estilo del formulario \*/*

#form {

background-color: white; */\* Fondo blanco para diferenciarlo del fondo de la página \*/*

padding: 30px 40px; */\* Espaciado interno \*/*

border-radius: 10px; */\* Bordes redondeados \*/*

border: 1px solid #ccc; */\* Borde gris claro \*/*

box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1); */\* Sombra suave \*/*

width: 100%; */\* Ocupa todo el ancho disponible \*/*

max-width: 500px; */\* Ancho máximo para mantenerlo compacto \*/*

margin-bottom: 40px; */\* Espacio inferior respecto al contenido que sigue \*/*

}

*/\* Título del formulario \*/*

h1 {

text-align: center; */\* Centra el texto \*/*

margin-bottom: 30px; */\* Espacio debajo del título \*/*

color: #333; */\* Gris oscuro \*/*

}

*/\* Estilo general del formulario \*/*

form {

display: flex; */\* Flexbox para organizar los campos \*/*

flex-direction: column; */\* Orden vertical \*/*

gap: 20px; */\* Espacio entre los elementos \*/*

}

*/\* Estilo para los campos de entrada \*/*

input {

padding: 10px; */\* Espaciado interno \*/*

border-radius: 6px; */\* Bordes redondeados \*/*

border: 1px solid #bbb; */\* Borde gris claro \*/*

font-size: 14px; */\* Tamaño del texto \*/*

width: 100%; */\* Ancho completo del contenedor \*/*

box-sizing: border-box; */\* Incluye padding y borde dentro del ancho \*/*

}

*/\* Estilo para las etiquetas \*/*

label {

margin-bottom: 5px; */\* Espacio debajo de la etiqueta \*/*

font-weight: bold; */\* Texto en negrita \*/*

color: #444; */\* Gris medio \*/*

}

*/\* Estilo para el mensaje de ayuda bajo el RUT \*/*

.ayuda {

font-size: 12px; */\* Letra más pequeña \*/*

color: #888; */\* Gris claro \*/*

margin-top: 4px; */\* Espacio arriba del mensaje \*/*

display: block; */\* Ocupa línea completa debajo del input \*/*

}

*/\* Estilo para el botón "Guardar" \*/*

#btn-guardar {

background-color: #0077cc; */\* Azul fuerte \*/*

color: white; */\* Texto blanco \*/*

padding: 12px; */\* Espaciado interno \*/*

font-size: 15px; */\* Tamaño del texto \*/*

border: none; */\* Sin borde \*/*

border-radius: 6px; */\* Bordes redondeados \*/*

cursor: pointer; */\* Cambia el cursor al pasar el mouse \*/*

transition: background-color 0.3s ease; */\* Transición suave al cambiar el color \*/*

}

*/\* Estilo del botón al pasar el mouse encima \*/*

#btn-guardar:hover {

background-color: #005fa3; */\* Azul más oscuro \*/*

}

*/\* Botón para editar datos \*/*

.btnEdit {

background-color: #28a745; */\* Verde \*/*

color: white;

padding: 6px 10px; */\* Espaciado interior \*/*

font-size: 13px;

border: none;

border-radius: 6px;

cursor: pointer;

transition: background-color 0.3s ease;

}

.btnEdit:hover {

background-color: #218838; */\* Verde más oscuro al pasar el mouse \*/*

}

*/\* Botón para eliminar datos \*/*

.btnDelete {

background-color: #dc3545; */\* Rojo \*/*

color: white;

padding: 6px 10px;

font-size: 13px;

border: none;

border-radius: 6px;

cursor: pointer;

transition: background-color 0.3s ease;

}

.btnDelete:hover {

background-color: #c82333; */\* Rojo más oscuro \*/*

}

*/\* Contenedor donde se muestran los datos guardados \*/*

#cajadatos {

background-color: white;

width: 100%;

max-width: 700px; */\* Más ancho que el formulario para que entre la tabla \*/*

border-radius: 10px;

border: 1px solid #ccc;

box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);

padding: 20px;

margin-bottom: 40px;

}

*/\* Estilo de la tabla que aparece dentro de cajadatos \*/*

#cajadatos table {

width: 100%; */\* Ocupa todo el ancho \*/*

border-collapse: collapse; */\* Elimina espacios entre celdas \*/*

margin-top: 20px;

}

*/\* Estilo para las celdas de la tabla \*/*

#cajadatos th, #cajadatos td {

padding: 12px;

text-align: left;

border-bottom: 1px solid #ddd; */\* Línea divisoria gris claro \*/*

}

*/\* Fila del encabezado de la tabla \*/*

#cajadatos th {

background-color: #f8f9fa; */\* Gris muy claro \*/*

}

*/\* Estilo al pasar el mouse sobre una fila \*/*

#cajadatos tr:hover {

background-color: #f1f1f1; */\* Color de fondo más claro al hacer hover \*/*

}

**3.- Taller Uno , véase prototipos->los tres html css javascript ESTE TERCERO PUNTO CINCO; 44 Ptos, véase rubrica para código apartado Html/Javascript**

**Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.**

**HTML**

<!DOCTYPE html> *<!-- Indica que el documento está en formato HTML5 -->*

<html lang="es"> *<!-- Define que el idioma del contenido es español -->*

<head>

<meta charset="UTF-8"> *<!-- Usa la codificación UTF-8 para que los caracteres especiales se muestren bien -->*

<meta name="viewport" content="width=device-width, initial-scale=1.0"> *<!-- Permite que el sitio sea responsive en dispositivos móviles -->*

<title>Proyecto Extendido</title> *<!-- Título que se muestra en la pestaña del navegador -->*

<link rel="stylesheet" href="./asset/styles.css"> *<!-- Enlace a la hoja de estilos CSS -->*

</head>

<body> *<!-- Inicio del cuerpo de la página -->*

*<!-- Contenedor principal -->*

<main class="container">

<h1>¡Hola Mundo!</h1> *<!-- Título principal de la página -->*

<p id="text">Este es un párrafo de ejemplo.</p> *<!-- Párrafo con ID para cambiarle el color mediante JavaScript -->*

*<!-- lista con 3 elementos -->*

<ul id="list">

<li>Elemento 1</li>

<li>Elemento 2</li>

<li>Elemento 3</li>

</ul>

*<!-- Botón para cambiar el color de fondo de la página -->*

<button id="colorButton" aria-label="Cambiar color de fondo">

Cambiar Color de Fondo

</button>

*<!-- Botón para cambiar el color del texto -->*

<button id="textColorButton" aria-label="Cambiar Color del Texto">

Cambiar Color del Texto

</button>

*<!-- Botón para cambiar el color de todos los botones -->*

<button id="buttonColorButton" aria-label="Cambiar Color de los Botones">

Cambiar Color de los Botones

</button>

*<!-- Selector de color personalizado -->*

<input type="color" id="customColorInput" value="#ff0000">

*<!-- Botón para aplicar el color personalizado elegido -->*

<button id="applyCustomColor">Aplicar Color Personalizado</button>

</main>

*<!-- Enlace al archivo JavaScript donde se hacen los cambios de color -->*

<script src="./js/script.js"></script>

</body>

</html>

**./js/script.js**

*// Selecciona los elementos del HTML por su ID o etiqueta*

const colorButton = document.getElementById('colorButton'); *// Botón para cambiar color de fondo*

const textColorButton = document.getElementById('textColorButton'); *// Botón para cambiar color del texto*

const buttonColorButton = document.getElementById('buttonColorButton'); *// Botón para cambiar color de los otros botones*

const customColorInput = document.getElementById('customColorInput'); *// Selector de color personalizado*

const applyCustomColorButton = document.getElementById('applyCustomColor'); *// Botón para aplicar el color personalizado*

const textElement = document.getElementById('text'); *// Párrafo al que se le cambiará el color del texto*

const listItems = document.querySelectorAll('#list li'); *// Todos los elementos de la lista*

const buttons = document.querySelectorAll('button'); *// Todos los botones de la página*

*// Lista (array) de colores para cambiar el fondo aleatoriamente*

const backgroundColors = ['#FF5733', '#33FF57', '#3357FF', '#F333FF', '#33FFF5'];

*// Lista de colores para cambiar el texto aleatoriamente*

const textColors = ['#000000', '#FF0000', '#00FF00', '#0000FF', '#FF00FF'];

*// Lista de colores para cambiar los botones aleatoriamente*

const buttonColors = ['#FF5733', '#33FF57', '#3357FF', '#F333FF', '#33FFF5'];

*// Función que cambia el color de fondo de la página*

function changeBackgroundColor() {

const randomColor = backgroundColors[Math.floor(Math.random() \* backgroundColors.length)];

document.body.style.backgroundColor = randomColor; *// Aplica el color al fondo del body*

}

*// Función que cambia el color del texto del párrafo y de los ítems de la lista*

function changeTextColor() {

const randomColor = textColors[Math.floor(Math.random() \* textColors.length)];

textElement.style.color = randomColor; *// Cambia el color del texto principal*

listItems.forEach(*item* => {

*item*.style.color = randomColor; *// Cambia el color de cada ítem de la lista*

});

}

*// Función que cambia el color de fondo de todos los botones*

function changeButtonColors() {

buttons.forEach(*button* => {

const randomColor = buttonColors[Math.floor(Math.random() \* buttonColors.length)];

*button*.style.backgroundColor = randomColor; *// Aplica un color aleatorio a cada botón*

});

}

*// Función que aplica el color personalizado elegido por el usuario al fondo*

function applyCustomBackgroundColor() {

const customColor = customColorInput.value; *// Toma el valor del input de tipo color*

document.body.style.backgroundColor = customColor; *// Cambia el fondo con ese color*

}

*// Añade funciones a los botones para que se activen al hacer clic*

colorButton.addEventListener('click', changeBackgroundColor); *// Clic en "Cambiar Color de Fondo"*

textColorButton.addEventListener('click', changeTextColor); *// Clic en "Cambiar Color del Texto"*

buttonColorButton.addEventListener('click', changeButtonColors); *// Clic en "Cambiar Color de los Botones"*

applyCustomColorButton.addEventListener('click', applyCustomBackgroundColor); *// Clic en "Aplicar Color Personalizado"*

**./asset/styles/style.css**

*/\* Estilos generales del cuerpo (body) de la página \*/*

body {

font-family: Arial, sans-serif; */\* Tipo de letra general \*/*

display: flex; */\* Usamos Flexbox para centrar el contenido \*/*

justify-content: center; */\* Centra el contenido horizontalmente \*/*

align-items: center; */\* Centra el contenido verticalmente \*/*

height: 100vh; */\* Ocupa el 100% de la altura de la pantalla \*/*

margin: 0; */\* Elimina márgenes por defecto del navegador \*/*

background-color: #f0f0f0; */\* Color de fondo gris claro \*/*

transition: background-color 0.5s ease; */\* Efecto suave cuando cambia el color de fondo \*/*

}

*/\* Contenedor principal que envuelve todo el contenido \*/*

.container {

text-align: center; */\* Centra el texto dentro del contenedor \*/*

}

*/\* Estilo del título principal \*/*

h1 {

color: #333; */\* Color gris oscuro \*/*

transition: color 0.5s ease; */\* Transición suave al cambiar el color \*/*

}

*/\* Estilo del párrafo de ejemplo \*/*

p {

color: #555; */\* Gris más claro \*/*

font-size: 18px; */\* Tamaño del texto \*/*

transition: color 0.5s ease; */\* Transición suave al cambiar color \*/*

}

*/\* Estilos para la lista sin viñetas \*/*

ul {

list-style-type: none; */\* Quita los puntitos o numeración \*/*

padding: 0; */\* Elimina el espacio interior \*/*

}

*/\* Estilo para cada elemento de la lista \*/*

li {

background-color: #fff; */\* Fondo blanco \*/*

margin: 5px; */\* Espacio entre elementos \*/*

padding: 10px; */\* Espacio interno \*/*

border-radius: 5px; */\* Bordes redondeados \*/*

box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1); */\* Sombra ligera para dar relieve \*/*

transition: background-color 0.5s ease; */\* Efecto suave al cambiar color de fondo \*/*

}

*/\* Efecto al pasar el mouse por encima de los elementos de la lista \*/*

li:hover {

transform: scale(1.1); */\* Aumenta ligeramente el tamaño \*/*

background-color: #0dc030b8; */\* Cambia a un verde llamativo \*/*

transition: transform 0.3s ease, background-color 0.3s ease; */\* Efecto suave en tamaño y color \*/*

cursor: pointer; */\* Cambia el cursor a manito \*/*

}

*/\* Estilo general para todos los botones \*/*

button {

padding: 10px 20px; */\* Espaciado interno (vertical y horizontal) \*/*

font-size: 16px; */\* Tamaño del texto \*/*

color: #fff; */\* Color del texto (blanco) \*/*

background-color: #007BFF; */\* Fondo azul intenso \*/*

border: none; */\* Sin borde \*/*

border-radius: 5px; */\* Bordes redondeados \*/*

cursor: pointer; */\* Cursor tipo manito al pasar por encima \*/*

margin: 5px; */\* Espacio alrededor de cada botón \*/*

transition: background-color 0.3s ease; */\* Efecto suave al cambiar color de fondo \*/*

}

*/\* Estilo para los botones cuando el mouse pasa por encima \*/*

button:hover {

background-color: #0056b3; */\* Azul más oscuro al hacer hover \*/*

}

*/\* Estilo para el input de tipo color \*/*

input[type="color"] {

padding: 5px; */\* Espaciado interno \*/*

border-radius: 5px; */\* Bordes redondeados \*/*

border: 1px solid #ccc; */\* Borde gris claro \*/*

cursor: pointer; */\* Cursor tipo manito \*/*

}

**4.- Taller Dos , véase prototipos->nodeconbasededatos; 44 Ptos, véase rubrica para código apartado Html/Javascript**

**Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.**

**Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.**

**HTML**

**./public/index.html**

<!DOCTYPE html> *<!-- Indica que este archivo está escrito en HTML5 -->*

<html lang="es"> *<!-- Define que el idioma principal de la página es español -->*

<head>

<meta charset="UTF-8"> *<!-- Usa codificación UTF-8 para mostrar bien acentos y caracteres especiales -->*

<meta name="viewport" content="width=device-width, initial-scale=1.0"> *<!-- Hace que la página se vea bien en celulares -->*

<title>Pregunta 4</title> *<!-- Título que aparece en la pestaña del navegador -->*

<link rel="stylesheet" href="styles.css"> *<!-- Conecta la hoja de estilos CSS externa -->*

</head>

<body> *<!-- Comienza el contenido visible de la página -->*

*<!-- Contenedor principal -->*

<div class="container">

<h1>CRUD Simple</h1> *<!-- Título principal de la página -->*

*<!-- Formulario para ingresar un nuevo usuario -->*

<form id="userForm">

<label for="name">Nombre:</label> *<!-- Etiqueta para el campo de nombre -->*

<input type="text" id="name" placeholder="Ingresa el nombre"> *<!-- Campo de texto para el nombre -->*

<label for="run">RUN:</label> *<!-- Etiqueta para el campo de RUT -->*

<input type="text" id="run" placeholder="Ejemplo: 12345678-9"> *<!-- Campo de texto para el RUT -->*

<label for="age">Edad:</label> *<!-- Etiqueta para el campo de edad -->*

<input type="number" id="age" placeholder="Ingresa la edad"> *<!-- Campo numérico para la edad -->*

<button type="submit">Guardar</button> *<!-- Botón para enviar el formulario -->*

</form>

*<!-- Sección donde se mostrará la lista de usuarios registrados -->*

<div id="userList">

<h3>Lista de Usuarios</h3> *<!-- Subtítulo para la lista -->*

<ul id="userTableBody"></ul> *<!-- Lista vacía donde se cargarán los usuarios -->*

</div>

</div>

*<!-- Archivo JavaScript donde se maneja la lógica del formulario -->*

<script src="script.js"></script>

</body>

</html>

**./public/script.js**

<!DOCTYPE html> *<!-- Indica que este archivo está escrito en HTML5 -->*

<html lang="es"> *<!-- Define que el idioma principal de la página es español -->*

<head>

<meta charset="UTF-8"> *<!-- Usa codificación UTF-8 para mostrar bien acentos y caracteres especiales -->*

<meta name="viewport" content="width=device-width, initial-scale=1.0"> *<!-- Hace que la página se vea bien en celulares -->*

<title>Pregunta 4</title> *<!-- Título que aparece en la pestaña del navegador -->*

<link rel="stylesheet" href="styles.css"> *<!-- Conecta la hoja de estilos CSS externa -->*

</head>

<body> *<!-- Comienza el contenido visible de la página -->*

*<!-- Contenedor principal -->*

<div class="container">

<h1>CRUD Simple</h1> *<!-- Título principal de la página -->*

*<!-- Formulario para ingresar un nuevo usuario -->*

<form id="userForm">

<label for="name">Nombre:</label> *<!-- Etiqueta para el campo de nombre -->*

<input type="text" id="name" placeholder="Ingresa el nombre"> *<!-- Campo de texto para el nombre -->*

<label for="run">RUN:</label> *<!-- Etiqueta para el campo de RUT -->*

<input type="text" id="run" placeholder="Ejemplo: 12345678-9"> *<!-- Campo de texto para el RUT -->*

<label for="age">Edad:</label> *<!-- Etiqueta para el campo de edad -->*

<input type="number" id="age" placeholder="Ingresa la edad"> *<!-- Campo numérico para la edad -->*

<button type="submit">Guardar</button> *<!-- Botón para enviar el formulario -->*

</form>

*<!-- Sección donde se mostrará la lista de usuarios registrados -->*

<div id="userList">

<h3>Lista de Usuarios</h3> *<!-- Subtítulo para la lista -->*

<ul id="userTableBody"></ul> *<!-- Lista vacía donde se cargarán los usuarios -->*

</div>

</div>

*<!-- Archivo JavaScript donde se maneja la lógica del formulario -->*

<script src="script.js"></script>

</body>

</html>

**./public/styles.css**

*/\* Estilos generales del cuerpo de la página \*/*

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif; */\* Fuente moderna y legible \*/*

background-color: #f4f6f9; */\* Fondo gris claro \*/*

margin: 0; */\* Elimina márgenes por defecto \*/*

padding: 40px; */\* Espacio interior alrededor del contenido \*/*

display: flex; */\* Usa Flexbox para centrar \*/*

justify-content: center; */\* Centra el contenido horizontalmente \*/*

}

*/\* Contenedor principal del formulario y lista \*/*

.container {

background-color: #ffffff; */\* Fondo blanco \*/*

max-width: 600px; */\* Ancho máximo del contenedor \*/*

width: 100%; */\* Ocupa todo el ancho disponible \*/*

padding: 30px 40px; */\* Espaciado interno (vertical y horizontal) \*/*

border-radius: 12px; */\* Bordes redondeados \*/*

box-shadow: 0 10px 25px rgba(0, 0, 0, 0.1); */\* Sombra suave alrededor \*/*

}

*/\* Título principal \*/*

h1 {

text-align: center; */\* Centrado del texto \*/*

margin-bottom: 30px; */\* Espacio debajo del título \*/*

color: #2c3e50; */\* Color azul oscuro \*/*

}

*/\* Espacio inferior al formulario \*/*

form {

margin-bottom: 30px;

}

*/\* Estilo para cada etiqueta del formulario \*/*

label {

display: block; */\* Ocupa toda la línea \*/*

margin-top: 15px; */\* Espacio encima del label \*/*

margin-bottom: 5px; */\* Espacio debajo \*/*

font-weight: 600; */\* Negrita \*/*

color: #34495e; */\* Gris azulado \*/*

}

*/\* Estilo de los campos de entrada \*/*

input {

width: 100%; */\* Ocupa todo el ancho \*/*

padding: 10px 12px; */\* Espaciado interno \*/*

border-radius: 6px; */\* Bordes redondeados \*/*

border: 1px solid #ccc; */\* Borde gris claro \*/*

font-size: 15px;

transition: border-color 0.3s; */\* Efecto suave al cambiar el borde \*/*

box-sizing: border-box; */\* Incluye padding y borde en el ancho \*/*

}

*/\* Efecto al hacer clic en un campo de entrada \*/*

input:focus {

outline: none; */\* Quita el borde azul por defecto \*/*

border-color: #3498db; */\* Borde azul al enfocar \*/*

}

*/\* Estilo del botón principal (Guardar) \*/*

button {

margin-top: 20px; */\* Espacio encima \*/*

padding: 12px; */\* Espaciado interior \*/*

width: 100%; */\* Ancho completo \*/*

font-size: 16px;

color: white;

background-color: #3498db; */\* Azul principal \*/*

border: none;

border-radius: 6px;

cursor: pointer;

transition: background-color 0.3s; */\* Transición suave al hacer hover \*/*

}

*/\* Efecto al pasar el mouse sobre el botón \*/*

button:hover {

background-color: #2980b9; */\* Azul más oscuro \*/*

}

*/\* Contenedor de la lista de usuarios \*/*

#userList {

margin-top: 30px; */\* Espacio superior \*/*

}

*/\* Título de la lista \*/*

#userList h3 {

margin-bottom: 15px;

color: #2c3e50;

border-bottom: 1px solid #ddd; */\* Línea divisoria debajo del título \*/*

padding-bottom: 5px;

}

*/\* Lista sin viñetas \*/*

#userList ul {

list-style: none;

padding: 0;

margin: 0;

}

*/\* Estilo para cada usuario mostrado en la lista \*/*

#userList li {

background-color: #f9f9f9; */\* Fondo gris muy claro \*/*

padding: 15px;

border: 1px solid #e1e1e1; */\* Borde gris claro \*/*

border-radius: 6px;

margin-bottom: 10px;

display: flex; */\* Distribuye el contenido en línea \*/*

justify-content: space-between; */\* Espacio entre nombre y botones \*/*

align-items: center;

transition: box-shadow 0.3s ease; */\* Efecto suave al pasar el mouse \*/*

}

*/\* Efecto al pasar el mouse sobre un elemento de la lista \*/*

#userList li:hover {

box-shadow: 0 4px 12px rgba(0, 0, 0, 0.05); */\* Sombra suave al hacer hover \*/*

}

*/\* Botón de eliminar dentro de la clase .actions \*/*

.actions button {

margin-left: 10px; */\* Espacio entre botones si hay más de uno \*/*

padding: 8px 12px;

background-color: #e74c3c; */\* Rojo (eliminar) \*/*

font-size: 14px;

border: none;

border-radius: 4px;

color: white;

cursor: pointer;

transition: background-color 0.3s ease;

}

*/\* Efecto hover del botón de eliminar \*/*

.actions button:hover {

background-color: #c0392b; */\* Rojo más oscuro \*/*

}

**./scriptdatabasePostgresSQL.sql**

*-- Crea una base de datos llamada 'crud\_db'*

CREATE DATABASE crud\_db;

*-- Conecta a la base de datos recién creada*

\c crud\_db;

*-- Crea la tabla 'users' dentro de la base de datos*

CREATE TABLE users (

id SERIAL PRIMARY KEY, *-- Campo 'id' numérico autoincrementable, clave primaria*

name VARCHAR(100) NOT NULL, *-- Campo 'name' de tipo texto con un máximo de 100 caracteres, obligatorio*

run VARCHAR(20) UNIQUE NOT NULL, *-- Campo 'run' (como el RUT) que debe ser único y obligatorio*

age INT NOT NULL *-- Campo 'age' de tipo número entero, obligatorio*

);

**./server**

*// Importa el módulo express para crear el servidor web*

const express = require('express');

*// Importa el Pool desde 'pg' para conectarse a PostgreSQL*

const { Pool } = require('pg');

*// Crea la aplicación express*

const app = express();

*// Permite que el servidor reciba datos en formato JSON*

app.use(express.json());

*// Permite servir archivos estáticos desde la carpeta "public"*

app.use(express.static('public'));

*// Configuración de conexión a la base de datos PostgreSQL*

const db = new Pool({

user: 'postgres', *// Usuario de la base de datos*

host: 'localhost', *// Servidor (en este caso, local)*

database: 'crud\_db', *// Nombre de la base de datos*

password: 'root', *// Contraseña del usuario*

port: 5432, *// Puerto por defecto de PostgreSQL*

});

app.get('/api/users', async (*req*, *res*) => {

try {

const result = await db.query('SELECT \* FROM users'); *// Consulta todos los registros de la tabla 'users'*

*res*.json(result.rows); *// Envía los datos como respuesta en formato JSON*

} catch (err) {

*res*.status(500).send(err); *// Si ocurre un error, responde con código 500 y el error*

}

});

app.post('/api/users', async (*req*, *res*) => {

const { name, run, age } = *req*.body; *// Extrae los datos enviados por el cliente (nombre, rut y edad)*

try {

const result = await db.query(

'INSERT INTO users (name, run, age) VALUES ($1, $2, $3) RETURNING \*', *// Consulta SQL con parámetros*

[name, run, age] *// Valores que se insertan en la base de datos*

);

*res*.json(result.rows[0]); *// Devuelve el usuario insertado como respuesta*

} catch (err) {

*res*.status(500).send(err); *// Muestra error si falla*

}

});

app.delete('/api/users/:id', async (*req*, *res*) => {

const { id } = *req*.params; *// Obtiene el ID del usuario desde la URL*

try {

await db.query('DELETE FROM users WHERE id = $1', [id]); *// Elimina el usuario con ese ID*

*res*.json({ message: 'Usuario eliminado' }); *// Envía mensaje de confirmación*

} catch (err) {

*res*.status(500).send(err); *// Muestra error si falla*

}

});

const PORT = 3000; *// Puerto en el que se ejecutará el servidor*

app.listen(PORT, () => {

console.log(`Servidor corriendo en http://localhost:${PORT}`); *// Mensaje en consola cuando arranca el servidor*

});

**5.- Portafolios, 20 ptos,** puntaje véase rubrica de portafolios

Cristóbal barrios: <https://drive.google.com/drive/folders/1r-AeL-aGeQDbiPQaM5inEDi3mkfRDybL?usp=sharing>

Lisset Macias: <https://github.com/Lisfer23/PruebaUnoMasPortafolio>

Roger Rodriguez: <https://github.com/roger-rd/protafolio_y_ejercicios_clases>

**6.- Autoevaluación, 16 ptos,** puntaje véase rubrica de Autoevaluación

Gráfico

El contenido generado por IA puede ser incorrecto.Roger Rodriguez

Lissette Macías

Tabla

El contenido generado por IA puede ser incorrecto.

Cristóbal Barrios

**Tabla

El contenido generado por IA puede ser incorrecto.**

**7.- Coevaluación, 16 ptos,** puntaje véase rubrica de coevaluación

Roger Rodriguez evalúa a Lisset macias

Imagen que contiene Texto

El contenido generado por IA puede ser incorrecto.

Lisset Macias Evalúa a Cristobal Barrios

Texto

El contenido generado por IA puede ser incorrecto.

Cristóbal Barrios Evalúa a Roger Rodriguez

Imagen que contiene Texto

El contenido generado por IA puede ser incorrecto.