

**Escuela:**

Centro de Estudios Tecnológicos Industriales y Servicios no.084

**Actividad 1:**

Formulario

**Maestro:**

Gabriel Ignacio China Cortez

**Integrantes:**

Lepe González Axel Yeray

**Grado/grupo:**

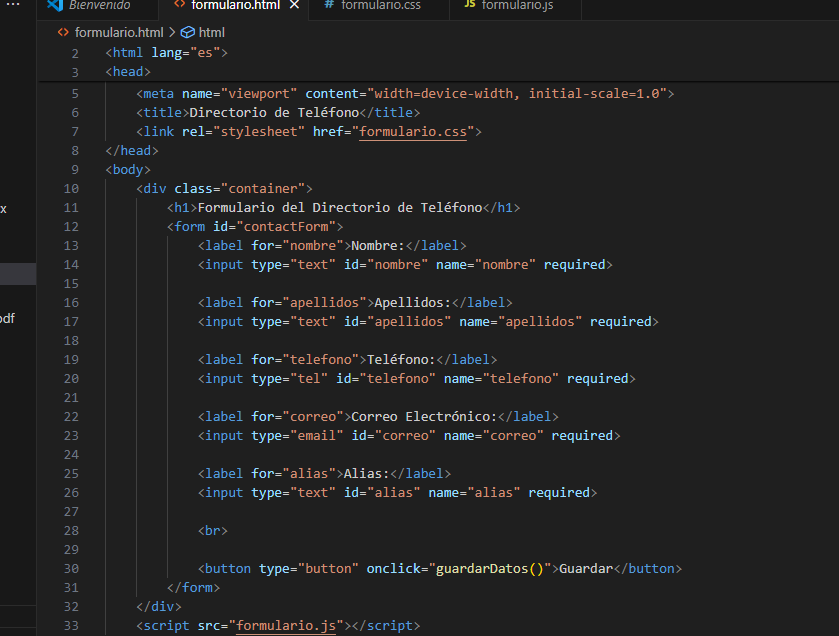
4°E

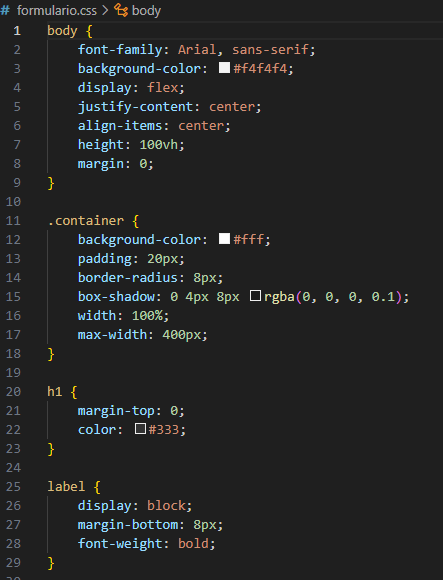
**Fecha:**

Viernes 30 de agosto del 2024

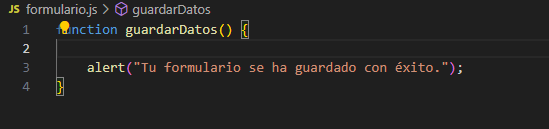
**Capturas**

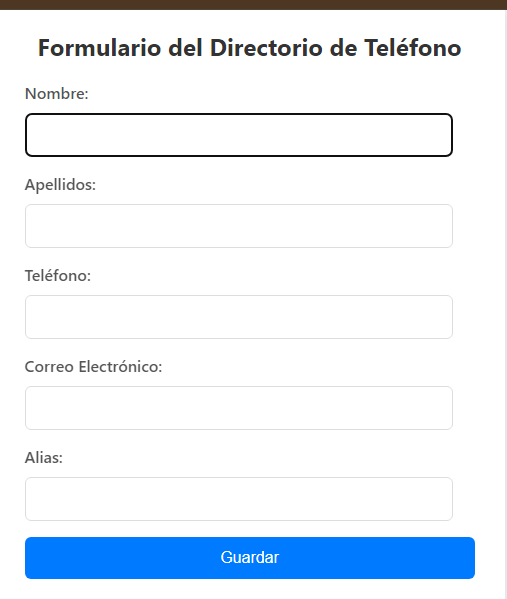
**HTML**

****

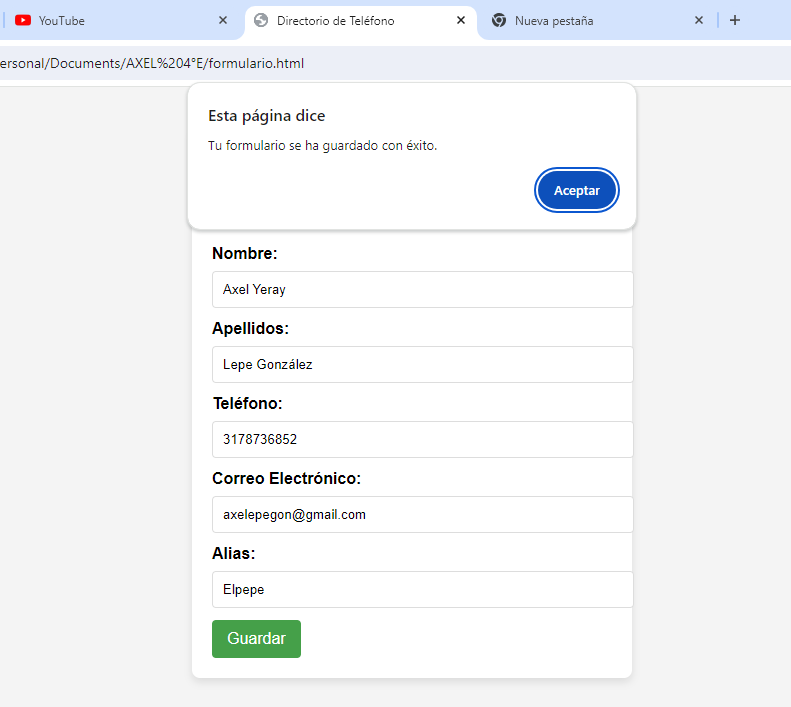
**CSS**

**JS**

****

**Capturas del programa**

****

****

**5 buenas practicas de PHP**

1. **Uso de Prepared Statements para SQL**: Siempre utiliza prepared statements con PDO o MySQLi para prevenir ataques de inyección SQL y mejorar la seguridad de tu aplicación.

**php**

$stmt = $pdo->prepare('SELECT \* FROM users WHERE email = :email');

$stmt->execute(['email' => $email]);

$user = $stmt->fetch();

1. **Validación y Sanitización de Datos**: Valida y sanitiza todos los datos de entrada del usuario para evitar problemas de seguridad y errores.

**php**

$email = filter\_var($\_POST['email'], FILTER\_SANITIZE\_EMAIL);

if (!filter\_var($email, FILTER\_VALIDATE\_EMAIL)) {

// Manejar error de validación

}

1. **Manejo de Errores y Excepciones**: Usa bloques try-catch para manejar excepciones y errores de manera controlada y evitar la exposición de información sensible.

**php**

try {

$pdo = new PDO($dsn, $user, $password);

} catch (PDOException $e) {

echo 'Error: ' . $e->getMessage();

}

1. **Uso de require y include para Modularización**: Organiza tu código en archivos separados y usa require o include para modularizarlo y mantener la claridad.

**php**

// En main.php

include 'header.php';

1. **Gestión de Sesiones y Cookies**: Usa session\_start() para manejar sesiones y asegúrate de configurar correctamente las cookies para mejorar la seguridad.

php

Copiar código

session\_start();

$\_SESSION['user\_id'] = $userId;

### **5 buenas practicas de HTML**

1. **Uso Correcto de Etiquetas Semánticas**: Utiliza etiquetas semánticas como <header>, <footer>, <article>, y <section> para mejorar la accesibilidad y SEO.

**html**

<header>

<h1>Bienvenido a mi sitio web</h1>

</header>

1. **Incluir Metaetiquetas**: Incluye metaetiquetas en el <head> para definir la codificación de caracteres, la vista de la página en dispositivos móviles y otras configuraciones.

**html**

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

1. **Uso de Atributos alt en Imágenes**: Siempre proporciona atributos alt en las imágenes para mejorar la accesibilidad.

**html**

<img src="logo.png" alt="Logotipo de la empresa">

1. **Creación de Formularios Accesibles**: Asegúrate de usar etiquetas <label> asociadas a campos de formulario para mejorar la accesibilidad.

**Html**

<label for="name">Nombre:</label>

<input type="text" id="name" name="name">

1. **Estructuración Adecuada con <div> y <span>**: Usa <div> para secciones de bloque y <span> para elementos en línea, y evita el uso excesivo de estas etiquetas.

**html**

<div class="container">

<span class="highlight">Texto destacado</span>

</div>

### **5 buenas practicas de JavaScript**

1. **Declaración de Variables con let y const**: Usa let y const en lugar de var para una mejor gestión del alcance de las variables.

javascript

const PI = 3.14;

let radius = 5;

1. **Evitar el Uso de eval()**: No uses eval() ya que puede introducir vulnerabilidades y errores en tu código.

javascript

// Evitar

let result = eval("2 + 2");

1. **Uso de try-catch para Manejo de Errores**: Usa bloques try-catch para manejar errores en tu código y evitar fallos inesperados.

javascript

try {

// Código que puede fallar

} catch (error) {

console.error('Error:', error);

}

1. **Funciones Anónimas y Arrow Functions**: Usa funciones anónimas o arrow functions para una sintaxis más concisa y clara.

javascript

const add = (a, b) => a + b;

1. **Manipulación del DOM de Forma Eficiente**: Evita manipular el DOM repetidamente; en su lugar, realiza las actualizaciones en un solo bloque.

javascript

const list = document.getElementById('list');

let html = '';

items.forEach(item => {

html += `<li>${item}</li>`;

});

list.innerHTML = html;

### **5 buenas practicas de CSS**

1. **Uso de Flexbox y Grid para Layouts**: Utiliza Flexbox y Grid para crear layouts flexibles y responsivos de manera más fácil.

**css**

.container {

display: flex;

justify-content: center;

align-items: center;

}

1. **Uso de Variables CSS**: Utiliza variables CSS para mantener la consistencia en los valores y facilitar cambios en el tema.

css

:root {

--primary-color: #3498db;

}

.button {

background-color: var(--primary-color);

}

1. **Uso de rem y em para Tamaños**: Usa unidades relativas como rem y em para tamaños de fuente y otros elementos para que sean más escalables y accesibles.

**css**

body {

font-size: 16px;

}

h1 {

font-size: 2rem;

}

1. **Estilos Responsivos con Media Queries**: Usa media queries para crear estilos que se adapten a diferentes tamaños de pantalla.

**css**

@media (max-width: 768px) {

.container {

flex-direction: column;

}

}

1. **Optimización de Imágenes con background-image**: Utiliza background-image en lugar de <img> para imágenes decorativas y controla su tamaño y posición con CSS.

**css**

.hero {

background-image: url('hero.jpg');

background-size: cover;

background-position: center;

}