Reporte de Vulnerabilidad

Archivo: index.php

Código Analizado:

```
<!doctype html>
<html lang="en">

<head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.5.2/css/all.min.css">


    <title>Hello, world!</title>
</head>

<body>
    <script>
        function eliminar(){
            var respuesta  = confirm("Estas seguro que deseas eliminar");
            return respuesta;
        }
    </script>
    <div class="container-fluid-row ">

        <div class="row px-4">
            <h1>Prueba Tecnica</h1>

            <div class="col-4">
                <div class="card">
                    <div class="card-body">
                        <form action="" method="post">
                            <!-- Mostrar Error     -->
                            <?php
                            include "models/conexion.php";
                            include "controller/registro.php";

                            ?>
                            <h3>Registro de Persona</h3>
                            <div class="mb-3">
                                <label for="nombre" class="form-label">Nombre</label>
                                <input type="text" class="form-control" id="nombre"
name="nombre">

                            </div>
                            <div class="mb-3">
                                <label for="apellido" class="form-label">Apellido</label>
                                <input type="text" class="form-control" id="apellido"
name="apellido">
                            </div>
                            <div class="mb-3">
                                <label for="dni" class="form-label">Cedula</label>
                                <input type="text" class="form-control" id="dni" name="dni">
                            </div>
                            <div class="mb-3">
                                <label for="fecha" class="form-label">Fecha Nacimiento</label>
                                <input type="date" class="form-control" id="fecha"
name="fecha">
                            </div>
                            <div class="mb-3">
                                <label for="email" class="form-label">Email</label>
                                <input type="email" class="form-control" id="email"
name="email">
                            </div>
                            <button type="submit" name="btnregistrar" class="btn btn-success"
value="ok">Registrar</button>
```

```
                        </form>
                    </div>
                </div>
            </div>
            <div class="col-8">
                <?php include "controller/eliminar.php"; ?>
                <table class="table table-success table-striped">
                    <thead>
                        <tr>
                            <th scope="col">#</th>
                            <th scope="col">Nombre</th>
                            <th scope="col">Apellido</th>
                            <th scope="col">Cedula</th>
                            <th scope="col">Fecha Nacimiento</th>
                            <th scope="col">Correo</th>
                            <th scope="col">Acciones</th>
                        </tr>
                    </thead>
                    <tbody>
                        <?php
                        include("models/conexion.php");
                        $registrosPorPagina = 5;
                        $pagina = isset($_GET['pagina']) ? (int)$_GET['pagina'] : 1;
                        if ($pagina < 1) $pagina = 1;

                        $offset = ($pagina - 1) * $registrosPorPagina;

                        // Total de registros y páginas
                        $resultadoTotal = $conexion->query("SELECT COUNT(*) AS total FROM
personas");
                        $totalRegistros = $resultadoTotal->fetch_object()->total;
                        $totalPaginas = ceil($totalRegistros / $registrosPorPagina);

                        // Consulta paginada
                        $sql = $conexion->query("SELECT * FROM personas LIMIT
$registrosPorPagina OFFSET $offset");
                        while ($a = $sql->fetch_object()) { ?>
                            <tr>
                                <td><?= $a->id ?></td>
                                <td><?= htmlspecialchars($a->nombre) ?></td>
                                <td><?= htmlspecialchars($a->apellido) ?></td>
                                <td><?= htmlspecialchars($a->cedula) ?></td>
                                <td><?= htmlspecialchars($a->fecha_nacimiento) ?></td>
                                <td><?= htmlspecialchars($a->correo) ?></td>
                                <td>
                                    <a class="btn btn-warning" href="edit.php?id=<?= $a->id
?>"><i class="fa-solid fa-pencil"></i></a>

                                    <a onclick="return eliminar()" class="btn btn-danger"
href="index.php?id=<?= $a->id ?>"><i class="fa-solid fa-trash"></i></a>
                                </td>
                            </tr>
                        <?php } ?>
                    </tbody>
                </table>
                <!-- Paginación -->
                <nav>
                    <ul class="pagination">
                        <?php if ($pagina > 1): ?>
                            <li class="page-item"><a class="page-link" href="?pagina=<?=
$pagina - 1 ?>">Anterior</a></li>
                        <?php endif; ?>

                        <?php for ($i = 1; $i <= $totalPaginas; $i++): ?>
                            <li class="page-item <?= $i == $pagina ? 'active' : '' ?>">
                                <a class="page-link" href="?pagina=<?= $i ?>"><?= $i ?></a>
                            </li>
                        <?php endfor; ?>

                        <?php if ($pagina < $totalPaginas): ?>
                            <li class="page-item"><a class="page-link" href="?pagina=<?=
$pagina + 1 ?>">Siguiente</a></li>
                        <?php endif; ?>
                    </ul>
                </nav>
            </div>
        </div>
    </div>

    <!-- Optional JavaScript; choose one of the two! -->
```

```
    <!-- Option 1: Bootstrap Bundle with Popper -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>

    <!-- Option 2: Separate Popper and Bootstrap JS -->
    <!--
    <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.2/dist/umd/popper.min.js"
integrity="sha384-IQsoLXl5PILFhosVNubq5LC7Qb9DXgDA9i+tQ8Zj3iwWAwPtgFTxbJ8NT4GN1R8p"
crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.min.js"
integrity="sha384-cVKIPhGWiC2Al4u+LWgxfKTRIcfu0JTxR+EQDz/bgldoEyl4H0zUF0QKbrJ0EcQF"
crossorigin="anonymous"></script>
    -->
</body>

</html>
```

Análisis: ```html
Vulnerabilidades de Seguridad

Inyección SQL

Línea aproximada: Dentro de `controller/eliminar.php` (no proporcionado) y posiblemente en `controller/registro.php` (no proporcionado) si no se sanitizan las entradas antes de usarlas en las consultas SQL. La vulnerabilidad reside en la concatenación directa de datos proporcionados por el usuario en la consulta SQL. En la línea: `...`, el parametro id, si bien es un entero, si el script `eliminar.php` usa este parametro sin validarlo y escaparlo en una consulta SQL, puede ser vulnerable.

Mitigación: Utilizar consultas preparadas (prepared statements) con parámetros en PDO o MySQLi para evitar la inyección SQL. Nunca concatenar directamente datos proporcionados por el usuario en las consultas. Validar y sanitizar todas las entradas del usuario. Asegurarse de que el script `eliminar.php` sanitiza el parametro ID antes de usarlo.

Calidad del Código

Acoplamiento Alto

El código presenta un acoplamiento alto entre las vistas (HTML) y los controladores (PHP). Se incluyen directamente archivos de controlador dentro de las vistas (`include "controller/registro.php";` y `include "controller/eliminar.php";`). Esto dificulta la reutilización y el mantenimiento del código.

Mejora: Implementar un patrón de diseño MVC (Modelo-Vista-Controlador) más estricto. Las vistas solo deben encargarse de la presentación, los controladores deben manejar la lógica de la aplicación y los modelos deben interactuar con la base de datos.

Falta de Manejo de Errores

El código no incluye un manejo robusto de errores. Por ejemplo, no se verifica si las consultas a la base de datos se ejecutan correctamente.

Mejora: Implementar un manejo de errores adecuado utilizando bloques `try-catch` para capturar excepciones y registrar o mostrar mensajes de error apropiados. Verificar siempre el resultado de las consultas a la base de datos.

Posible Duplicación de Código

Se incluyen multiples veces la conexión a la base de datos (`models/conexion.php`).

Mejora: Crear una función o clase para la conexión a la base de datos y llamarla desde donde sea necesario. Usar `require_once` o `include_once` para asegurar que el archivo se incluya solo una vez.

Legibilidad

Si bien el código es relativamente legible, podría mejorarse la organización y la documentación.

Mejora: Agregar comentarios para explicar la lógica de las diferentes secciones del código. Utilizar nombres de variables y funciones descriptivos.

Solución Propuesta

1. **Sanitización de Entradas:** Implementar funciones de validación y sanitización para todas las entradas del usuario (nombre, apellido, cédula, fecha, email) antes de insertar o actualizar datos en la base de datos. Utilizar `htmlspecialchars()` para mostrar datos en la vista y evitar ataques XSS (ya se hace, pero verificar consistencia).
2. **Consultas Preparadas:** Utilizar consultas preparadas (prepared statements) con parámetros en PDO o MySQLi para prevenir la inyección SQL.
3. **Patrón MVC:** Refactorizar el código para seguir un patrón de diseño MVC más estricto, separando la lógica de la aplicación en modelos, vistas y controladores.
4. **Manejo de Errores:** Implementar un manejo de errores adecuado utilizando bloques `try-catch` para capturar excepciones y registrar o mostrar mensajes de error apropiados.
5. **Validación en el Servidor:** Siempre realizar validación de datos en el servidor, incluso si ya se realiza en el cliente (JavaScript). La validación del lado del cliente puede ser omitida o manipulada.
6. **Control de Acceso:** Implementar un sistema de autenticación y autorización para proteger el acceso a las funcionalidades de la aplicación. (Fuera del alcance del snippet, pero importante en una app real).
7. **Paginación Segura:** Validar que el parámetro `pagina` sea un entero positivo antes de usarlo en la consulta SQL para la paginación.
8. **Protección CSRF:** Implementar protección contra ataques CSRF (Cross-Site Request Forgery), especialmente en los formularios de registro y eliminación. Esto se puede lograr mediante el uso de tokens CSRF.
9. **Uso de Frameworks:** Considerar el uso de un framework PHP (como Laravel, Symfony o CodeIgniter) para facilitar el desarrollo y mejorar la seguridad y la calidad del código. Estos frameworks ya implementan muchas de las medidas de seguridad necesarias.
10. **Auditoria de Seguridad:** Realizar una auditoría de seguridad exhaustiva del código para identificar y corregir posibles vulnerabilidades.

```
```