Manual de Desarrollador: Sistema de Control Automático

Equipo de Desarrollo: Departamento de Electrónica

November 29, 2024

Contents

1 Introducción

Este manual está dirigido a los desarrolladores del sistema de **Control Automático** utilizado en el laboratorio de electrónica del Departamento de Electrónica de la UDLAP. En este documento se describen los componentes clave del código HTML del sistema y su funcionamiento.

2 Descripción del Código HTML

A continuación se describe cada sección del código HTML proporcionado.

2.1 Estructura Básica

El archivo comienza con la declaración del tipo de documento y el elemento `html`, que define la versión del documento como HTML5.

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

La línea `lang="en"` indica que el contenido está en inglés.

2.2 Sección Head

La sección `head` contiene metadatos y enlaces a recursos externos como hojas de estilo y scripts. Esta sección incluye:

- `meta charset="UTF-8"`: Define la codificación de caracteres.
- `meta name="viewport" content="width=device-width, initial-scale=1.0"`: Hace que la página sea responsive.
- `title`: Especifica el título de la página que aparece en la pestaña del navegador.
- Enlaces a hojas de estilo: Se incluyen tres archivos de estilo CSS: `control-automatico.css`, `styles.css` y el de `Bootstrap`.

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>preview-almacen</title>
    <link rel="stylesheet" href="css/control-automatico.css">
    <link rel="stylesheet" href="css/styles.css">
    <link href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" rel="styleshee
</head>
```

2.3 Cuerpo de la Página

El cuerpo de la página (`body`) contiene el contenido visible y se divide en varias secciones:

2.3.1 Navegación

La barra de navegación `nav` contiene la imagen del logotipo de la UDLAP y los encabezados del sistema, que muestran el nombre del departamento y el título del sistema.

```
<nav id="main-nav">
    <div class="logo-container">
        <img src="resources/images/logo-udlap.png" alt="" class="img-fluid" id="logo-udlap">
    </div>
```

```
    <div class="header-container" id="header-container">
        <header id="nav-departamento">
            Departamento de Electrónica
        </header>
        <header id="nav-titulo">
            -Control automático del laboratorio de electrónica-
        </header>
    </div>
</nav>
```

2.3.2 Contenedores de Botones

En el cuerpo de la página, se encuentran los botones de interacción que permiten al usuario acceder a diferentes funcionalidades del sistema. Están organizados en un contenedor de filas (`row`) y columnas (`col`) usando el sistema de grillas de Bootstrap.

Cada botón tiene una imagen (`img`) y un `tooltip` que proporciona una descripción cuando el usuario pasa el mouse sobre el botón.

```
<div class="container-buttons">
    <div class="row">
        <div class="col-3 column-left">
            <div class="button-ca" id="button-upper-left">
                <a href="http://localhost/sistema_almacen/php/forms/form-check_id.php">
                    <img src="resources/images/icon-adduser.png" alt="">
                    <div class="tooltip-left">
                        <h3>Agregar Usuario</h3>
                    </div>
                </a>
            </div>
        </div>
    </div>
</div>
```

2.3.3 Botones de Control

Cada botón en la sección tiene una acción asociada. Estos botones permiten al usuario realizar tareas como agregar un usuario, consultar información, cambiar de semestre, generar reportes de inventarios y más.

Los botones están estructurados de manera similar:

- `Agregar Usuario`: Redirige al formulario para agregar un nuevo usuario.
- `Consultas`: Redirige al formulario de consultas.
- `Cambio de semestre`: Cambia el semestre actual.
- `Reporte de inventarios`: Muestra un reporte detallado del inventario.

2.4 Imagen Central

En la sección central, se presenta el escudo de la UDLAP, alineado en el centro de la página.

```
<div class="col-6 text-center">
    <img src="resources/images/escudo-udlap.png" alt="UDLAP Logo" class="img-fluid" id="escudo-udlap">
</div>
```

2.5 Final del Cuerpo

La página termina con una serie de botones de la columna derecha, cada uno con una imagen representativa y su respectivo tooltip. En el código proporcionado, los botones son:

- `Control automático`: Redirige al formulario de control automático.
- `Estadísticas`: Accede a las estadísticas del sistema.
- `Mantenimiento`: Redirige al área de mantenimiento del sistema.

```
<div class="col-3 column-right">
    <div class="button-ca" id="button-upper-right">
        <a href="html/form-automatic_control.html">
            <img src="resources/images/icon-user.png" alt="">
            <div class="tooltip-right">
                <h3>Control automático</h3>
            </div>
        </a>
    </div>
</div>
```

3 Módulo 2: Base de Datos

El módulo 2 del sistema está basado en diversas tablas que contienen la información esencial del laboratorio. A continuación, se describen las tablas principales utilizadas para gestionar los datos.

3.1 Tabla `chips`

La tabla `chips` contiene información sobre los chips utilizados en el laboratorio. Sus columnas son las siguientes:

- `ART_NO` (double): Número de artículo del chip.
- `POSICIONX` (varchar(512)): Posición física del chip en el laboratorio.
- `ETIQUETA` (varchar(512)): Etiqueta que identifica al chip.
- `CONECTOR` (varchar(512)): Tipo de conector del chip.
- `DESCRIP1` (varchar(512)): Descripción adicional del chip.
- `DESCRIP2` (varchar(512)): Descripción adicional del chip.
- `MINIMO` (double): Cantidad mínima en existencia del chip.
- `EXISTENCIA` (double): Cantidad actual de chips en existencia.
- `PEDIDOS` (double): Cantidad de chips pedidos.
- `CONECTOR_2` (varchar(512)): Otro tipo de conector asociado al chip.
- `PEDIDO` (varchar(512)): Información adicional sobre el pedido.
- `PRECIO` (double): Precio del chip.
- `FECHA_ADQ` (varchar(512)): Fecha de adquisición del chip.
- `PROVEEDOR` (varchar(512)): Proveedor del chip.
- `CHKX` (varchar(512)): Estado de verificación del chip.

- `CONT_1` (double): Contador relacionado con el chip.
- `CONT_2` (double): Otro contador relacionado con el chip.
- `NO_PROVEE` (varchar(512)): Información del proveedor que no provee el chip.
- `STOCK` (varchar(512)): Estado del stock del chip.

3.2 Tabla `conexion`

La tabla `conexion` almacena las conexiones de los componentes electrónicos en el laboratorio, con las mismas columnas que la tabla `chips`, proporcionando detalles sobre cada conexión, como el artículo conectado, la posición, el conector utilizado, y más.

- Las columnas son idénticas a las de la tabla `chips`, con la diferencia de que esta tabla se enfoca en las conexiones de los dispositivos.

3.3 Tabla `empleado`

La tabla `empleado` muestra información sobre los empleados del laboratorio. Sus columnas son las siguientes:

- `NUMERO` (int): Número de empleado.
- `NOMBRE` (varchar(512)): Nombre del empleado.
- `PUESTO` (varchar(512)): Puesto o cargo del empleado en el laboratorio.
- `UNIDADES` (double): Cantidad de unidades gestionadas por el empleado.
- `CLASE` (varchar(512)): Clase o tipo de trabajo que realiza el empleado.

3.4 Tabla `equipo`

La tabla `equipo` contiene los registros de los equipos electrónicos del laboratorio, con las siguientes columnas:

- `aparato` (varchar(255)): Nombre del aparato o equipo.
- `marca` (varchar(255)): Marca del equipo.
- `modelo` (varchar(255)): Modelo del equipo.
- `numero_ser` (varchar(255)): Número de serie del equipo.
- `fecha` (varchar(255)): Fecha de adquisición o registro del equipo.
- `encargado` (varchar(255)): Persona encargada del equipo.
- `posicion` (varchar(255)): Ubicación física del equipo.
- `inv_v04` (varchar(255)): Información adicional sobre el inventario del equipo.
- `fecha_inv` (varchar(255)): Fecha de inventario del equipo.
- `status` (varchar(255)): Estado del equipo (activo, inactivo, etc.).
- `chk` (varchar(255)): Estado de verificación del equipo.
- `inventario` (varchar(255)): Número de inventario del equipo.
- `nueva_pos` (varchar(255)): Nueva ubicación del equipo.
- `pos_temp` (varchar(255)): Posición temporal del equipo.

- `rep_mtto` (float): Registro de mantenimiento del equipo.
- `invo02` (varchar(255)): Información adicional sobre el inventario.
- `pos_may19` (varchar(255)): Posición de inventario de mayo de 2019.
- `no_af` (varchar(255)): Número de equipo asignado.
- `inv_p11` (varchar(255)): Información adicional del inventario.
- `posp18` (varchar(255)): Posición física en el inventario.
- `responinv` (varchar(255)): Persona responsable del inventario.
- `hora_inv` (float): Hora del inventario.
- `grupo` (varchar(255)): Grupo al que pertenece el equipo.
- `es_af` (varchar(255)): Estado de Afiliación.

3.5 Tabla `prim14a`

La tabla `prim14a` registra los préstamos realizados en el laboratorio, con las siguientes columnas:

- `NUMERO` (double): Número de préstamo.
- `NOMBRE` (varchar(512)): Nombre de la persona que realizó el préstamo.
- `NOMPAR` (varchar(512)): Nombre del equipo o componente prestado.
- `TIPMOV` (varchar(512)): Tipo de movimiento del préstamo (entrada o salida).
- `FECHA` (varchar(512)): Fecha del préstamo.
- `ENCARGADO` (double): Número del encargado del préstamo.
- `HORA` (double): Hora del préstamo.
- `CANTOMULTA` (double): Monto de multa en caso de retraso.
- `REAL_VAL` (double): Valor real del préstamo.
- `DEUDOR` (double): Persona que debe el equipo prestado.

3.6 Tabla `usuarios`

La tabla `usuarios` contiene los registros de todos los usuarios del laboratorio, con las siguientes columnas:

- `numero` (int): Número de usuario.
- `nombre` (varchar(255)): Nombre del usuario.
- `status` (varchar(10)): Estado del usuario (activo o inactivo).
- `permiso` (varchar(10)): Nivel de permiso del usuario.
- `nivel` (varchar(10)): Nivel de acceso del usuario.
- `faltas` (float): Número de faltas registradas.
- `fecha_falt` (date): Fecha de la última falta.
- `descripcio` (varchar(255)): Descripción adicional del usuario.

- `deudac_usd` (float): Deuda en dólares estadounidenses.
- `e_mail` (varchar(255)): Correo electrónico del usuario.
- `telefono` (varchar(20)): Número de teléfono del usuario.
- `deudax_usd` (float): Deuda adicional en dólares estadounidenses.
- `CLAVE` (varchar(255)): Clave o contraseña del usuario.

4 Carpeta `Forms`

La carpeta `Forms` contiene todos los archivos necesarios para la implementación y el manejo de formularios en el sistema. Estos formularios están escritos en PHP y permiten la interacción con el usuario a través de diversas interfaces, tales como el registro de nuevos usuarios, la actualización de datos, la gestión de solicitudes, entre otros.

Cada formulario está diseñado para capturar, validar y procesar información proveniente del usuario de manera eficiente. Además, todos los formularios cuentan con validación de datos, asegurando que la información ingresada cumpla con los requisitos establecidos, y están interconectados con las tablas de la base de datos para almacenar, actualizar o consultar los registros de manera segura.

La subcarpeta `credentials` contiene archivos relacionados con la gestión de las credenciales de acceso, tanto para los usuarios como para los administradores, garantizando la seguridad en el acceso al sistema.

4.1 Formulario de Verificación de ID de Usuario (check_id.php)

El formulario `check_id.php` es una página web diseñada para que los usuarios ingresen su ID y lo verifiquen en el sistema. Este formulario es el primero en el proceso de registro y modificación de usuarios, permitiendo validar la existencia del ID antes de proceder con la modificación de la información del usuario.

4.1.1 Relación con la Página Principal (index.html)

Este formulario es accedido desde el *index.html* del sistema, donde el usuario interactúa con un botón para iniciar el proceso de verificación del ID. El formulario `check_id.php` es invocado mediante un botón en el *index.html* que permite al usuario acceder a la verificación de su ID. El botón que activa esta acción está vinculado con el enlace del formulario y al hacer clic en él, el usuario es redirigido a esta página donde puede ingresar su ID para comprobar si está registrado en el sistema.

4.1.2 Estructura del Código HTML

El código HTML de la página está compuesto por las siguientes secciones clave:

- **Enlace a Hojas de Estilo:**
  - `styles.css`: Estilo general de la página.
  - `styles-adduser.css`: Estilo adicional para la página de registro y modificación de usuarios.
  - `styles-check_id.css`: Estilo específico para la página de verificación de ID.
  - `Bootstrap CSS`: Se utiliza para el diseño responsive y la visualización de los elementos de la página.
- **Script:** Se enlaza el archivo JavaScript `script-adduser.js`, que probablemente maneje la interacción de la página (como validaciones adicionales, si las hay).
- **Barra de Navegación:** La barra de navegación contiene el logotipo de la universidad y los encabezados que describen la función de la página. Se muestra el nombre del *Departamento de Electrónica* y el título *Registro y modificación de usuarios.*
- **Formulario:** El formulario es el elemento principal de la página, donde el usuario debe ingresar su ID:

  - **Campo de Entrada:** Un campo de texto con el id `user-id` y nombre `user-id`. Este campo está diseñado para aceptar solo números (`pattern="[0-9]+"`), lo que indica que el ID debe ser numérico.
  - **Botón de Envío:** Un botón `Verificar`, el cual, al ser presionado, enviará el formulario a la URL `bkend-check_id.php` usando el método `POST`.

- **Redirección:** En la parte superior de la página se encuentra un botón `regresar` que permite al usuario regresar a la página principal (index.html).

4.1.3 Descripción de la Funcionalidad

La funcionalidad principal de este formulario es verificar la validez del ID de usuario ingresado. Al ingresar un ID y presionar el botón `Verificar`, el sistema enviará una solicitud `POST` al script `bkend-check_id.php`. Este script se encargará de validar si el ID es correcto y existe en la base de datos del sistema, proporcionando retroalimentación al usuario sobre la validez de su ID.

4.1.4 Explicación del Código HTML

A continuación, se presenta una breve descripción de las secciones clave del código HTML:

- **Formulario (`form`):** El formulario está diseñado para enviar los datos al script PHP de backend `bkend-check_id.php`. Este archivo manejará la lógica de verificación en el servidor.
- **Campo de Entrada (`input`):** El campo de entrada `user-id` tiene un patrón que asegura que solo se ingresen números, lo que es una validación básica en el lado del cliente.
- **Botón de Envío (`button`):** El botón de tipo `submit` es utilizado para enviar el formulario al servidor para su verificación.
- **Barra de Navegación:** Se incluye un enlace con el logotipo y encabezados, facilitando la navegación dentro del sistema.

4.1.5 Recomendaciones

Es importante asegurarse de que el archivo `bkend-check_id.php` esté correctamente implementado para manejar la lógica de validación del ID en la base de datos y brindar la respuesta adecuada al usuario.

4.2 Formulario: `form-consults-check`$_{admin_id.php}$

Este archivo HTML proporciona un formulario para verificar el ID y la clave de un administrador en un sistema. El formulario solicita al usuario que ingrese el ID y la contraseña de un administrador y, al enviarlo, realiza una consulta para verificar su validez.

4.2.1 Estructura y Funcionalidad

4.2.2 Formulario de Verificación de ID

El archivo contiene un formulario HTML con dos campos principales:

- `admin-id`: Un campo de texto donde se debe ingresar el ID de un administrador. Este campo es obligatorio y solo permite números como entrada, gracias al atributo `pattern="[0-9]+"`.
- `admin-password`: Un campo de texto donde se debe ingresar la clave del administrador. Este campo también es obligatorio y solo permite números como entrada, con el mismo patrón que el campo anterior.

Los datos del formulario se envían al servidor mediante una solicitud POST al archivo `bkend-consults-check_admin_id.php`.

4.2.3 Interfaz de Usuario

Barra de Navegación La barra de navegación incluye el logo de la universidad (*UDLAP*) y el nombre del departamento (*Departamento de Electrónica*). También se muestra el título de la página (*-Registro y modificación de usuarios-*), proporcionando contexto al usuario sobre el propósito de la página.

Contenedor Principal El contenedor principal incluye:

- Un enlace para regresar a la página principal (`index.html`).
- Un formulario que contiene los campos para ingresar el ID y la clave del administrador. Este formulario está contenido dentro de un cuadro blanco con la clase `box-white` para mejorar la visibilidad y presentación.

Botón de Envío El formulario incluye un botón de tipo `submit` con la clase `btn btn-primary btn-lg`, que envía los datos del formulario al servidor para su verificación.

4.2.4 Estilos y Diseño

La página utiliza varios recursos de estilos para su presentación:

- `styles.css`: Hoja de estilo principal para la página.
- `styles-adduser.css`: Hoja de estilo específica para la funcionalidad de agregar usuarios.
- `styles-check_id.css`: Hoja de estilo personalizada para esta página de verificación de ID de administrador.
- `bootstrap.min.css`: Framework CSS utilizado para el diseño responsivo y los componentes visuales.

Estas hojas de estilo contribuyen a la estructura visual y la accesibilidad de la página.

4.2.5 Flujo de Navegación

El usuario puede interactuar con la página de la siguiente manera:

1. Ingresar a la página `form-check_id.php`.
2. Ingresar el ID y la clave del administrador en los campos correspondientes.
3. Presionar el botón `Verificar` para enviar los datos al servidor.
4. El formulario se envía a `bkend-consults-check_admin_id.php`, donde los datos se validan y verifican.

El flujo es sencillo y permite a los usuarios verificar los datos del administrador con facilidad.

4.2.6 Scripts y Funciones

El archivo incluye un enlace a un archivo JavaScript:

- `script-adduser.js`: Archivo JavaScript que se carga para realizar cualquier acción adicional en el formulario, aunque no se utiliza explícitamente en este archivo HTML.

Sin embargo, el script no parece tener un impacto directo en esta página en particular, ya que no hay funciones o eventos relacionados con JavaScript en el formulario.

4.2.7 Mensajes de Estado

La página no incluye mensajes de estado visibles en la interfaz de usuario. Los mensajes de éxito o error se esperan en la página de respuesta luego de la verificación del ID y la clave del administrador.

4.3 Formulario: `select_type.php`

Este archivo HTML corresponde a un formulario que permite al usuario seleccionar el tipo de consulta dentro del laboratorio de electrónica. El formulario ofrece tres opciones: Equipos, Conexiones y Chips. Dependiendo de la opción seleccionada, el usuario será redirigido a diferentes formularios para realizar consultas específicas.

4.3.1 Estructura y Funcionalidad

4.3.2 Enlace a Hojas de Estilo

El formulario utiliza varias hojas de estilo para proporcionar un diseño atractivo y responsivo:

- `control-automatico.css`: Estilo específico para la interfaz de control automático.
- `styles.css`: Estilo general que se aplica a todo el sitio web.
- `styles-select_type.css`: Estilo personalizado para este formulario de selección de tipo de consulta.
- Bootstrap: Framework de diseño que permite un diseño responsivo, optimizando la visualización en diferentes dispositivos.

4.3.3 Barra de Navegación

El formulario incluye una barra de navegación en la parte superior, que contiene:

- El logo de la universidad (`logo-udlap.png`), que sirve como un enlace a la página principal del sitio.
- El nombre del departamento: *Departamento de Electrónica*.
- Un título que describe la funcionalidad de la página: *Selector de consultas del laboratorio de electrónica*.

La barra de navegación proporciona acceso rápido a la página principal y ofrece contexto sobre el propósito de la página.

4.3.4 Contenedor Principal

El formulario se encuentra dentro de un contenedor principal, que incluye:

- Un encabezado que da la bienvenida al usuario e indica que debe seleccionar una opción.
- Tres botones de selección, cada uno representando una consulta diferente: *Equipos*, *Conexiones* y *Chips*.

Cada botón está asociado con una imagen y un título que lo representa visualmente, y redirige a los respectivos formularios de consulta:

- El botón `Equipos` redirige a `form-consult_equipos.php`.
- El botón `Conexiones` redirige a `form-consult_conexiones.php`.
- El botón `Chips` redirige a `form-consult_chips.php`.

Estos botones proporcionan una forma rápida y visual de navegar a los diferentes tipos de consultas disponibles.

4.3.5 Footer

El formulario incluye un pie de página centrado con el texto:

> *© 2024 Departamento de Electrónica - UDLAP*

Este pie de página proporciona información de copyright y refuerza la marca institucional de la universidad.

4.3.6 Flujo de Navegación

El usuario puede interactuar con el formulario de la siguiente manera:

1. Ingresar a la página `select_type.php` desde el menú de opciones.
2. Seleccionar una de las opciones de consulta (Equipos, Conexiones o Chips).
3. Ser redirigido al formulario correspondiente donde podrá realizar la consulta detallada.

Este diseño simplifica el proceso de navegación y mejora la experiencia del usuario al permitirle elegir entre varias opciones de consulta de manera clara y accesible.

4.4 Formulario: `form-consult_equipos.php`

Este archivo PHP es utilizado para mostrar una lista de equipos dentro del laboratorio de electrónica. Permite a los usuarios ver, editar y eliminar equipos a través de una interfaz web. La página muestra una tabla con los detalles de cada equipo y tiene funcionalidades de búsqueda y paginación.

4.4.1 Estructura y Funcionalidad

4.4.2 Conexiones y Consultas

El archivo incluye módulos para gestionar la conexión a la base de datos y realizar consultas relacionadas con los equipos:

- `conn.php`: Módulo encargado de establecer la conexión con la base de datos.
- `bkend-consult_equipos.php`: Módulo que contiene funciones para obtener todos los equipos y realizar consultas específicas sobre ellos.

4.4.3 Procesamiento de Solicitudes

El archivo verifica si se ha solicitado editar o eliminar un equipo. Si es el caso, llama a las funciones correspondientes para procesar estas acciones. Si no se obtienen equipos, la variable `$equipos` se inicializa como un array vacío para evitar errores.

4.4.4 Mostrar Equipos

La página obtiene todos los equipos disponibles desde la base de datos y los muestra en una tabla. Si no existen equipos, se muestra un mensaje indicando que no hay equipos disponibles.

4.4.5 Edición de Equipos

Si un equipo es seleccionado para ser editado, sus datos son mostrados de manera editable. Al presionar el botón de "Editar", los campos se vuelven editables, y el botón cambia a "Guardar". Cuando se guarda, los datos son enviados a un script PHP (`mod-edit_equipo.php`) mediante AJAX.

4.4.6 Eliminación de Equipos

Cada equipo tiene un botón de "Eliminar". Al presionar este botón, se solicita confirmación al usuario y, si el usuario acepta, el equipo es eliminado de la base de datos a través de AJAX. La fila correspondiente se elimina de la tabla.

4.4.7 Interfaz de Usuario

Barra de Navegación La barra de navegación incluye el logo de la universidad, el nombre del departamento (*Departamento de Electrónica*) y el título de la página (-*Consulta y búsqueda de equipos*-). Esto proporciona contexto al usuario sobre el propósito de la página.

Contenedor Principal El contenedor principal incluye un campo de búsqueda que permite al usuario filtrar los resultados de la tabla de equipos. Además, la tabla muestra los siguientes campos de cada equipo:

- Aparato
- Marca
- Modelo
- Número de serie
- Encargado
- Posición
- Estado
- Fecha de inventario

Cada fila incluye botones de "Editar" y "Eliminar" para realizar las acciones correspondientes.

Paginación El sistema también incluye controles de paginación, permitiendo a los usuarios navegar entre las diferentes páginas de resultados.

4.4.8 Estilos y Diseño

La página utiliza Bootstrap para el diseño responsivo y otros estilos personalizados:

- `bootstrap.min.css`: Framework CSS para el diseño responsivo y componentes visuales.
- `styles-consults.css`: Hoja de estilo personalizada para este formulario, que da formato a los elementos de la página.

4.4.9 Flujo de Navegación

El usuario puede interactuar con la página de la siguiente manera:

1. Ingresar a la página `form-consult_equipos.php`.
2. Buscar equipos mediante el campo de búsqueda.
3. Seleccionar un equipo para editar o eliminar.
4. Si se selecciona editar, los campos del equipo se vuelven editables. Luego, al hacer clic en "Guardar", los cambios se guardan mediante AJAX.
5. Si se selecciona eliminar, el equipo se elimina de la base de datos tras una confirmación.

El flujo de trabajo es intuitivo y permite a los usuarios gestionar fácilmente los equipos del laboratorio.

4.4.10 Scripts y Funciones

El archivo incluye varias funciones JavaScript:

- `filterTable()`: Filtra los resultados de la tabla de equipos según el texto ingresado en el campo de búsqueda.
- Funciones para editar y guardar los cambios de un equipo mediante AJAX.
- Función para eliminar un equipo mediante AJAX.
- Funciones de paginación para navegar entre las páginas de resultados.

Estas funciones mejoran la experiencia del usuario, permitiendo una interacción fluida con la página sin necesidad de recargarla.

4.4.11 Mensajes de Estado

La página muestra mensajes de éxito o error en una barra de notificación al realizar acciones de edición o eliminación. Estos mensajes proporcionan retroalimentación inmediata al usuario sobre las acciones realizadas.

4.5 Formulario: `form-consult_conexiones.php`

Este archivo PHP es utilizado para mostrar una lista de conexiones dentro de un sistema. Permite a los usuarios ver, editar y eliminar conexiones a través de una interfaz web. La página muestra una tabla con los detalles de cada conexión y tiene funcionalidades de búsqueda y paginación.

4.5.1 Estructura y Funcionalidad

4.5.2 Conexiones y Consultas

El archivo incluye módulos para gestionar la conexión a la base de datos y realizar consultas relacionadas con las conexiones:

- `conn.php`: Módulo encargado de establecer la conexión con la base de datos.
- `mod-edit_conexiones.php`: Módulo que contiene funciones para actualizar una conexión.
- `mod-delete_conexiones.php`: Módulo que contiene funciones para eliminar una conexión.

4.5.3 Procesamiento de Solicitudes

El archivo obtiene todas las conexiones disponibles desde la base de datos y las muestra en una tabla. Si no existen conexiones, se muestra un mensaje indicando que no hay conexiones disponibles.

4.5.4 Mostrar Conexiones

La página obtiene todas las conexiones disponibles desde la base de datos y las muestra en una tabla. Si no existen conexiones, se muestra un mensaje indicando que no hay conexiones disponibles.

4.5.5 Edición de Conexiones

Cada conexión tiene un botón "Editar" que permite hacer los campos de la fila editables. Al presionar el botón, los campos se vuelven editables y el botón cambia a "Guardar". Cuando se guarda, los datos son enviados a un script PHP (`mod-edit_conexiones.php`) mediante AJAX.

4.5.6 Eliminación de Conexiones

Cada conexión tiene un botón "Eliminar". Al presionar este botón, se solicita confirmación al usuario y, si el usuario acepta, la conexión es eliminada de la base de datos a través de AJAX. La fila correspondiente se elimina de la tabla.

4.5.7 Interfaz de Usuario

Barra de Navegación La barra de navegación incluye el logo de la universidad, el nombre del departamento (*Departamento de Electrónica*) y el título de la página (-*Consulta y búsqueda de conexiones*-). Esto proporciona contexto al usuario sobre el propósito de la página.

Contenedor Principal El contenedor principal incluye un campo de búsqueda que permite al usuario filtrar los resultados de la tabla de conexiones. Además, la tabla muestra los siguientes campos de cada conexión:

- ART_NO
- POSICIONX
- ETIQUETA
- CONECTOR
- DESCRIP1
- DESCRIP2
- EXISTENCIA

Cada fila incluye botones de "Editar" y "Eliminar" para realizar las acciones correspondientes.

Paginación El sistema también incluye controles de paginación, permitiendo a los usuarios navegar entre las diferentes páginas de resultados.

4.5.8 Estilos y Diseño

La página utiliza Bootstrap para el diseño responsivo y otros estilos personalizados:

- `bootstrap.min.css`: Framework CSS para el diseño responsivo y componentes visuales.
- `styles-consults.css`: Hoja de estilo personalizada para este formulario, que da formato a los elementos de la página.

4.5.9 Flujo de Navegación

El usuario puede interactuar con la página de la siguiente manera:

1. Ingresar a la página `form-consult_conexiones.php`.
2. Buscar conexiones mediante el campo de búsqueda.
3. Seleccionar una conexión para editar o eliminar.
4. Si se selecciona editar, los campos de la conexión se vuelven editables. Luego, al hacer clic en "Guardar", los cambios se guardan mediante AJAX.
5. Si se selecciona eliminar, la conexión se elimina de la base de datos tras una confirmación.

El flujo de trabajo es intuitivo y permite a los usuarios gestionar fácilmente las conexiones.

4.5.10 Scripts y Funciones

El archivo incluye varias funciones JavaScript:

- `filterTable()`: Filtra los resultados de la tabla de conexiones según el texto ingresado en el campo de búsqueda.
- Funciones para editar y guardar los cambios de una conexión mediante AJAX.
- Función para eliminar una conexión mediante AJAX.
- Funciones de paginación para navegar entre las páginas de resultados.

Estas funciones mejoran la experiencia del usuario, permitiendo una interacción fluida con la página sin necesidad de recargarla.

4.5.11 Mensajes de Estado

La página muestra mensajes de éxito o error en una barra de notificación al realizar acciones de edición o eliminación. Estos mensajes proporcionan retroalimentación inmediata al usuario sobre las acciones realizadas.

4.6 Formulario: `form-adduser.php`

Este archivo HTML es utilizado para registrar un nuevo usuario dentro de un sistema. A través de un formulario interactivo, el usuario puede ingresar su ID, nombre completo, nivel de permisos, faltas, estado, email, teléfono, y un nuevo PIN, los cuales son enviados a un archivo PHP para su procesamiento.

4.6.1 Estructura y Funcionalidad

4.6.2 Formulario de Registro

El archivo contiene un formulario estructurado con varios campos de entrada, donde el usuario puede proporcionar la información necesaria para su registro:

- `ID`: Campo de texto para ingresar el ID de estudiante (solo números).
- `Nombre Completo`: Campo de texto para ingresar el nombre completo del usuario.
- `Nivel`: Campo de texto para ingresar el nivel de permisos del usuario.
- `Faltas`: Campo de texto para ingresar el número de faltas del usuario.
- `Estado`: Un campo de selección (`Activo/Inactivo`) para determinar el estado del usuario.
- `Email`: Campo de texto para ingresar la dirección de correo electrónico.
- `Teléfono`: Campo de texto para ingresar el número telefónico.
- `Nuevo PIN`: Campo de contraseña para establecer un nuevo PIN numérico.
- `Confirmar PIN`: Campo de contraseña para confirmar el nuevo PIN.

4.6.3 Procesamiento de Solicitudes

El formulario envía los datos al archivo PHP `bkend-add_user.php` mediante el método `POST`. Este archivo se encarga de procesar la información, almacenarla en la base de datos y realizar las validaciones correspondientes.

4.6.4 Validaciones

Los campos de entrada están validados a nivel de cliente utilizando atributos como `pattern`, para asegurarse de que solo se ingresen valores válidos, como números en los campos de ID, faltas, teléfono y PIN.

4.6.5 Alerta de Registro

Al cargar la página, se ejecuta un script JavaScript que muestra una alerta informando al usuario que si el ID proporcionado no se encuentra en la base de datos, el sistema procederá a registrarlo como un nuevo usuario.

4.6.6 Interfaz de Usuario

Barra de Navegación La barra de navegación está compuesta por:

- El logo de la universidad *UDLAP*.
- El nombre del departamento (*Departamento de Electrónica*).
- El título de la página (*-Registro de usuarios-*).

Estos elementos proporcionan contexto al usuario sobre el propósito de la página.

Contenedor Principal El contenedor principal incluye el formulario de registro de usuario y un botón para regresar a la página principal. El formulario tiene los siguientes elementos:

- Un campo de texto para ingresar el ID del estudiante.
- Campos adicionales para ingresar el nombre completo, nivel de permisos, faltas, estado, email, teléfono, nuevo PIN y confirmar PIN.
- Un botón de envío para enviar los datos del formulario.

Estilos y Diseño El formulario hace uso de las siguientes hojas de estilo:

- `styles.css`: Hoja de estilo general para el diseño de la página.
- `styles-adduser.css`: Hoja de estilo personalizada para el formulario de registro de usuario.
- `bootstrap.min.css`: Framework CSS para el diseño responsivo y componentes visuales.

El diseño es responsivo gracias a la integración de Bootstrap, lo que permite que el formulario se adapte a diferentes tamaños de pantalla.

4.6.7 Flujo de Navegación

El flujo de navegación es el siguiente:

1. El usuario accede a la página `form-consult_adduser.php`.
2. El usuario ingresa la información requerida en los campos del formulario.
3. Al presionar el botón "Enviar", los datos del formulario se envían al archivo `bkend-add_user.php` para su procesamiento.
4. Si el ID no existe en la base de datos, el sistema registra al usuario como nuevo.

4.6.8 Mensajes de Estado

El sistema muestra una alerta al cargar la página informando al usuario sobre el registro de un nuevo usuario si el ID no se encuentra en la base de datos.

4.6.9 Scripts y Funciones

El archivo incluye el siguiente script JavaScript:

- `script-adduser.js`: Este script contiene la función que se ejecuta al cargar la página, mostrando una alerta sobre el registro de un nuevo usuario si el ID no se encuentra en la base de datos.

4.6.10 Ejemplo de Código

A continuación se muestra un fragmento de código correspondiente al formulario de registro de usuario:

```
<form id="user-form" action="../modules/bkend-add_user.php" method="POST">
    <div class="form-group">
        <label for="user-id">ID</label>
        <input type="text" class="form-control" id="user-id" name="user-id" pattern="[0-9]+" required pl
    </div>
    ...
    <button type="submit" class="btn btn-primary btn-lg" id="button-enviar">Enviar</button>
</form>
```

4.7 Formulario: `form-edit_user.php`

Este archivo PHP permite la edición de los datos de un usuario en el sistema. Los usuarios pueden modificar sus detalles como nombre, nivel, faltas, estado, correo electrónico, teléfono, y PIN. Los cambios se envían al servidor a través de un formulario HTML y se procesan mediante un script PHP.

4.7.1 Estructura y Funcionalidad

4.7.2 Formulario de Edición de Usuario

El archivo contiene un formulario con varios campos que permiten editar la información de un usuario. Los campos incluyen:

- `user-id`: Un campo de solo lectura que muestra el ID del usuario.
- `user-name`: Un campo para ingresar el nombre completo del usuario.
- `user-level`: Un campo para ingresar el nivel del usuario.
- `user-absences`: Un campo para ingresar la cantidad de faltas del usuario.
- `user-status`: Un campo desplegable para seleccionar el estado (activo/inactivo).
- `user-email`: Un campo para ingresar el correo electrónico del usuario.
- `user-phone`: Un campo para ingresar el número de teléfono del usuario.
- `user-password` y `user-password-confirm`: Campos para ingresar y confirmar el nuevo PIN.

4.7.3 Procesamiento de Solicitudes

Cuando el formulario se envía, los datos son procesados por el archivo PHP `bkend-edit_user.php`. Los valores de los campos son validados y actualizados en la base de datos. Si el usuario desea cambiar el PIN, se verifican ambos campos de la contraseña antes de realizar la actualización.

4.7.4 Precarga de Datos

El formulario se llena automáticamente con los datos del usuario a partir de un parámetro en la URL (`user_id`). Los datos se obtienen mediante una solicitud `fetch` que solicita un script PHP (`mod-fetch-user-data.php`) que devuelve los datos del usuario en formato JSON. Los valores nulos se reemplazan por un valor por defecto de "0" para evitar que los campos se queden vacíos.

4.7.5 Interfaz de Usuario

Barra de Navegación La barra de navegación contiene el logo de la universidad, el nombre del departamento (*Departamento de Electrónica*), y el título de la página (*-Editar Usuario-*), proporcionando un contexto visual para el usuario.

Contenedor Principal El contenedor principal incluye un formulario con campos de entrada para los datos del usuario. El formulario está dividido en dos secciones: una sección con los datos básicos del usuario (ID, nombre, nivel, faltas) y una sección adicional con los datos más específicos (estado, correo electrónico, teléfono, y PIN). Los usuarios pueden modificar estos campos según sea necesario.

Botón de Regreso Se incluye un botón de regreso (`icon-return.png`) que permite al usuario volver a la página principal. Esto ayuda a mejorar la navegación en la interfaz.

4.7.6 Estilos y Diseño

La página usa el framework Bootstrap para el diseño responsivo, junto con estilos personalizados:

- `bootstrap.min.css`: Framework CSS para crear un diseño limpio y responsivo.
- `styles-edituser.css`: Hoja de estilo personalizada que da formato a los elementos específicos de esta página de edición.

4.7.7 Flujo de Navegación

El usuario puede interactuar con la página de la siguiente manera:

1. Ingresar a la página `form-edit_user.php`.
2. Los datos del usuario se cargan automáticamente en el formulario.
3. El usuario puede modificar los campos de la forma y hacer clic en el botón "Actualizar" para enviar los cambios al servidor.
4. Los cambios se procesan y actualizan en la base de datos.

El flujo es simple y permite a los usuarios actualizar sus datos de manera eficiente.

4.7.8 Scripts y Funciones

El archivo incluye varias funciones JavaScript:

- `focus()`: Un evento que selecciona automáticamente el contenido de los campos de texto, email y contraseña cuando el usuario hace clic en ellos.
- Función `fetch()`: Se utiliza para obtener los datos del usuario desde el servidor a través del script `mod-fetch-user-data.php` y rellenar el formulario.
- Validación de los campos del formulario para asegurar que el PIN coincida entre los campos de "Nuevo PIN" y "Confirmar PIN".

Estas funciones mejoran la experiencia de usuario al hacer que la página sea más interactiva y dinámica.

4.7.9 Mensajes de Estado

El formulario no incluye mensajes explícitos de estado en la interfaz de usuario, pero se esperan notificaciones de éxito o error en la página que redirige al usuario una vez que se envía el formulario, según si la actualización fue exitosa o no.

5 Carpeta: `Modules`

La carpeta `Modules` contiene todos los archivos necesarios para hacer funcionar el backend del sistema. Esta carpeta incluye los módulos que gestionan las operaciones de servidor como la conexión a la base de datos, la manipulación de datos, la autenticación de usuarios, y las consultas a la base de datos, entre otros. Cada módulo está diseñado para cumplir con una función específica dentro del sistema y se organiza de manera que facilite su mantenimiento y expansión.

5.1 Estructura General

Dentro de `Modules`, los archivos se agrupan según su propósito. Algunos de los módulos incluyen:

- `bkend-consults-check_admin_id.php`: Módulo encargado de verificar la validez de los ID y contraseñas de los administradores.
- `conn.php`: Módulo que establece la conexión a la base de datos.
- `mod-edit_users.php`: Módulo para gestionar la edición de usuarios.
- `mod-delete_users.php`: Módulo que contiene la lógica para eliminar usuarios de la base de datos.
- `mod-insert_user.php`: Módulo para insertar nuevos usuarios en la base de datos.

5.2 Funcionalidad Principal

Los módulos en esta carpeta permiten al sistema realizar las siguientes tareas:

- Autenticación y validación de usuarios (por ejemplo, verificar ID de administradores).
- Manipulación de registros de usuarios y administradores.
- Interacción con la base de datos para almacenar y recuperar información.
- Implementación de lógica de negocio, como validación de datos y control de acceso.

La organización modular permite un fácil mantenimiento y actualización del sistema, facilitando la integración de nuevas funcionalidades o el ajuste de las existentes.

5.3 Módulo: `conn.php`

Este archivo PHP es responsable de establecer una conexión con la base de datos utilizando PDO (PHP Data Objects). Se conecta a un servidor MySQL con los parámetros proporcionados y maneja posibles errores de conexión.

5.3.1 Estructura y Funcionalidad

5.3.2 Conexión a la Base de Datos

El archivo establece una conexión con la base de datos `almacen` en el servidor `localhost` usando las credenciales de usuario y contraseña especificadas:

- `server`: El nombre del servidor de base de datos, en este caso `localhost`.
- `user`: El nombre de usuario para autenticar la conexión con la base de datos.
- `password`: La contraseña asociada al usuario para establecer la conexión.
- `database`: El nombre de la base de datos a la que se conecta (`almacen`).

La conexión se establece mediante el objeto `PDO`, que utiliza la cadena de conexión `mysql:host=`*server*; *dbname* =`database;ch`
para especificar el servidor y la base de datos.

5.3.3 Manejo de Excepciones

El código utiliza un bloque `try-catch` para manejar excepciones durante el proceso de conexión:

- `try`: Intenta establecer la conexión a la base de datos utilizando PDO.
- `catch`: Si ocurre un error (como fallos de autenticación o problemas con el servidor), se captura la excepción y se muestra un mensaje de error personalizado con `die()`.

El atributo PDO::ATTR$_E$RRMODEseestableceenPDO::ERRMODE$_E$XCEPTIONparaquecualquiererrordelaconexióngenereu

El archivo contiene una línea comentada de echo para verificar la conexión, pero esta línea no debe estar activa en producción.

5.3.5 Interfaz de Usuario

Este archivo no tiene una interfaz de usuario visible, ya que está destinado a establecer una conexión con la base de datos en el backend. Sin embargo, la conexión es fundamental para el funcionamiento de otros módulos del sistema.

5.3.6 Flujo de Navegación

El flujo de este archivo es automático, y se ejecuta al ser incluido en otros scripts que requieren acceso a la base de datos:

1. El archivo se incluye en otros módulos que necesiten realizar consultas a la base de datos.
2. La conexión a la base de datos se establece en segundo plano, sin interacción directa con el usuario.
3. Si la conexión es exitosa, el script continúa su ejecución. Si ocurre un error, el proceso se detiene y se muestra un mensaje de error.

5.3.7 Estilos y Diseño

Este archivo no tiene estilos ni diseño, ya que su único propósito es la conexión a la base de datos. No se generan elementos visuales ni se utilizan hojas de estilo.

5.3.8 Scripts y Funciones

Este archivo no contiene JavaScript ni funciones adicionales. Su única función es establecer la conexión con la base de datos y manejar posibles errores de conexión.

5.3.9 Mensajes de Estado

En caso de éxito, no se muestra ningún mensaje explícito de estado al usuario. Si ocurre un error, se muestra un mensaje de error mediante die() que indica el fallo en la conexión a la base de datos.

5.4 Módulo: `reader.php`

Este archivo PHP es responsable de obtener y mostrar todos los registros de usuarios almacenados en la base de datos del sistema. El módulo realiza una consulta SQL para obtener los datos y luego los presenta en formato tabular.

5.4.1 Estructura y Funcionalidad

5.4.2 Conexión a la Base de Datos

El archivo incluye el archivo conn.php para establecer la conexión a la base de datos. La conexión se realiza mediante PDO, lo que permite un manejo eficiente de las consultas SQL y proporciona seguridad contra inyecciones SQL.

5.4.3 Consulta SQL

El archivo ejecuta una consulta SQL para seleccionar todos los registros de la tabla usuarios de la base de datos. La consulta se prepara y ejecuta utilizando el método prepare() y execute() de PDO, lo que garantiza la ejecución segura de la consulta.

5.4.4 Obtención de Datos

Una vez ejecutada la consulta, los datos obtenidos se almacenan en la variable $user_info, queesunarrayasoci$

Los datos se presentan en formato tabular utilizando echo:

- Se imprime una fila de encabezado con los nombres de las columnas.
- Se imprime una línea de separación (un guion de 50 caracteres).
- Luego, se recorren todas las filas de datos y se imprime cada valor de las columnas.

5.4.6 Interfaz de Usuario

El archivo no tiene una interfaz de usuario en sí, ya que el propósito principal de este módulo es mostrar los resultados en formato de texto plano. Los datos se muestran directamente en el navegador como una tabla con los valores de las columnas.

5.4.7 Estilos y Diseño

Este archivo no hace uso de estilos CSS ya que su objetivo principal es la visualización de los datos de la base de datos en formato tabular, sin requerir una presentación visual elaborada.

5.4.8 Flujo de Navegación

El flujo de trabajo en este archivo es el siguiente:

1. Se incluye el archivo conn.php para establecer la conexión a la base de datos.
2. Se ejecuta la consulta SQL para obtener todos los registros de la tabla usuarios.
3. Los datos se muestran en la página en formato tabular.

5.4.9 Mensajes de Estado

Este archivo no presenta mensajes de estado ni errores de forma explícita. Si la conexión a la base de datos falla, se generará un error en el archivo conn.php que interrumpirá la ejecución del código.

5.5 Módulo: check_id_admin.php

Este archivo PHP maneja la verificación del ID y la clave de un administrador a través de un formulario enviado por el método POST. El archivo se encarga de validar la existencia del ID y la clave en las tablas correspondientes de la base de datos y, si todo es correcto, redirige al usuario al formulario de selección de tipo.

5.5.1 Estructura y Funcionalidad

5.5.2 Conexión a la Base de Datos

El archivo incluye el archivo conn.php al principio para establecer la conexión con la base de datos utilizando PDO. Este archivo es responsable de manejar la conexión a la base de datos y gestionar las excepciones en caso de errores.

5.5.3 Procesamiento de Datos del Formulario

Cuando se envía el formulario mediante el método POST, el archivo recoge y sanitiza los datos ingresados por el usuario (admin-id y admin-password). Luego, valida que los campos no estén vacíos. Si algún campo está vacío, el proceso se detiene y se muestra un mensaje de error.

5.5.4 Consultas a la Base de Datos

El archivo realiza dos consultas SQL para verificar la existencia del ID de administrador y la clave proporcionada:

- sql_check_empleado: Consulta la tabla empleado para verificar si el NUMERO (ID del administrador) existe.
- sql_check_usuario: Consulta la tabla usuarios para verificar si el NUMERO y la CLAVE coinciden.

5.5.5 Redirección del Usuario

Si ambas validaciones son correctas, el archivo redirige al usuario a la página form-select_type.php pasando el user_id como parámetro en la URL. Si alguno de los campos está vacío o las consultas no encuentran coincidencias, se muestra un mensaje de error.

5.5.6 Manejo de Errores

El archivo utiliza un bloque try-catch para capturar y manejar errores de conexión a la base de datos o de ejecución de las consultas SQL. Si ocurre un error, se detiene la ejecución y se muestra un mensaje detallado con el error.

5.5.7 Flujo de Navegación

El flujo de navegación es el siguiente:

1. El usuario ingresa al formulario form-check_id.php y envía su ID y clave.
2. El archivo bkend-consults-check$_{a}dmin_{i}d.phpprocesalosdatos.SielIDylaclavesoncorrectos, elsistemaredirigealus$
3. Si alguno de los datos es incorrecto, se muestra un mensaje de error y no se permite la redirección.

5.5.8 Estilos y Diseño

Este archivo no incluye estilos o diseño propios, ya que su función es puramente de procesamiento de datos. Los estilos se gestionan en otros archivos relacionados con la interfaz de usuario.

5.5.9 Scripts y Funciones

El archivo no contiene scripts JavaScript, ya que el procesamiento y la validación de los datos se realiza completamente en el servidor mediante PHP. Sin embargo, se hace uso de las funciones PHP de sanitización y validación de entrada.

5.5.10 Mensajes de Estado

En caso de que los datos proporcionados sean incorrectos o falten, el archivo muestra un mensaje de error correspondiente y detiene la ejecución del proceso.

5.6 Módulo: bkend-consult_equipos.php

Este archivo PHP maneja las operaciones de lectura, actualización y eliminación de equipos en una base de datos. Las funciones permiten obtener todos los equipos, buscar un equipo por su número de serie, actualizar sus datos o eliminar un equipo específico. Además, el archivo también gestiona las solicitudes de edición y eliminación mediante formularios enviados por el usuario.

5.6.1 Estructura y Funcionalidad

5.6.2 Conexión a la Base de Datos

El archivo incluye el archivo conn.php, que se utiliza para establecer la conexión con la base de datos utilizando PDO.

5.6.3 Funciones de Gestión de Equipos

El módulo define varias funciones para gestionar los equipos almacenados en la base de datos:

- getEquipments(*connection*) : *Recuperatodoslosequiposdelatabla*equipo.getEquipmentBySerial(*connection*,numero_s *Recuperaunequipoespecíficodelatabla*equipo*utilizandosunúmerodeserie*.
- updateEquipment(*connection*,numero_ser, ...): Actualiza los detalles de un equipo en la base de datos, incluyendo el aparato, marca, modelo, encargado, posición, estado y fecha de inventario.
- deleteEquipment(*connection*,numero_ser): Elimina un equipo de la base de datos usando su número de serie.

Cada una de estas funciones realiza una consulta SQL preparada utilizando PDO para protegerse contra inyecciones SQL. También se verifica si la ejecución de las consultas fue exitosa; si no lo fue, se muestra un mensaje de error.

5.6.4 Procesamiento de Solicitudes de Edición o Eliminación

La función processRequest(*connection*)*seencargadeprocesarlassolicitudesPOSTenviadasdesdeelformulario* :

Si se envía una solicitud de edición (con la clave edit), se llama a la función updateEquipment() para actualizar los datos del equipo.

Si se envía una solicitud de eliminación (con la clave delete), se llama a la función deleteEquipme para eliminar el equipo.

Una vez procesada la solicitud, el sistema redirige al usuario a la página search_equipment.php con un parámetro de éxito correspondiente (success=edit o success=delete).

5.6.5 Flujo de Navegación

El flujo de navegación es el siguiente:

1. El usuario envía una solicitud POST para editar o eliminar un equipo.
2. La función processRequest() valida y procesa la solicitud.
3. Si la acción es exitosa, el usuario es redirigido a la página search_equipment.php.

5.6.6 Manejo de Errores

Cada consulta SQL dentro de las funciones está envuelta en una validación para comprobar si la ejecución fue exitosa. Si ocurre algún error durante la ejecución de la consulta, se muestra un mensaje de error detallado con la descripción del fallo.

5.6.7 Scripts y Funciones

Este módulo no incluye scripts JavaScript, ya que todas las operaciones se realizan en el servidor mediante PHP.

5.6.8 Mensajes de Estado

Cuando el formulario de edición o eliminación se procesa con éxito, el usuario es redirigido a `search_equipment.php` con un mensaje de éxito (`success=edit` o `success=delete`). En caso de error, se muestra un mensaje de error en el servidor.

5.7 Módulo: `mod-edit_equipment.php`

Este archivo PHP permite la edición de los detalles de un equipo existente en la base de datos. A través de un formulario, el usuario puede actualizar información relacionada con un equipo, como el aparato, la marca, el modelo, el encargado, la posición y el estado del equipo, utilizando su número de serie como identificador.

5.7.1 Estructura y Funcionalidad

5.7.2 Conexión a la Base de Datos

El archivo incluye el archivo `conn.php`, que gestiona la conexión a la base de datos utilizando PDO.

5.7.3 Actualización de Datos

El archivo maneja una solicitud POST, donde se recogen los siguientes datos del formulario enviado:

- `id`: El número de serie del equipo, utilizado como identificador único.
- `aparato`: El nombre o tipo de aparato que se está editando.
- `marca`: La marca del equipo.
- `modelo`: El modelo del equipo.
- `encargado`: El responsable del equipo.
- `posicion`: La ubicación o posición del equipo.
- `status`: El estado actual del equipo (por ejemplo, activo, en reparación, etc.).

Estos datos se utilizan en una consulta SQL `UPDATE` para actualizar el registro correspondiente en la base de datos, en la tabla `equipo`.

5.7.4 Ejecutando la Consulta

La consulta SQL está preparada utilizando PDO, y los parámetros del formulario se vinculan utilizando `bindParam` para evitar problemas de inyección SQL. Si la ejecución de la consulta es exitosa, se devuelve la respuesta `success`. Si ocurre un error, se devuelve `error`.

5.7.5 Flujo de Navegación

El flujo de navegación es el siguiente:

1. El usuario envía una solicitud POST para actualizar los datos de un equipo.
2. El archivo `mod-edit_equipment.php` procesa la solicitud, ejecutando la consulta SQL de actualización.
3. Si la operación es exitosa, se devuelve el mensaje `success`. Si hay un error en la ejecución, se devuelve el mensaje `error`.

5.7.6 Manejo de Errores

Si ocurre un error al ejecutar la consulta de actualización, el sistema devuelve el mensaje `error`. No se detalla el tipo de error, pero esto puede ser útil para la depuración.

5.7.7 Mensajes de Estado

El archivo devuelve mensajes de estado simples como respuesta:

- `success`: Indica que la actualización fue exitosa.
- `error`: Indica que hubo un problema al ejecutar la actualización.

Estos mensajes son útiles para mostrar información al usuario o para manejar el flujo en el frontend.

5.8 Módulo: `mod-delete_equipos.php`

Este archivo PHP se encarga de eliminar un equipo específico de la base de datos en función de su número de serie. Recibe una solicitud `POST` que contiene el ID del equipo, y, tras verificar su existencia, ejecuta una consulta SQL `DELETE` para eliminar el registro correspondiente de la tabla `equipo`.

5.8.1 Estructura y Funcionalidad

5.8.2 Conexión a la Base de Datos

El archivo incluye el archivo `conn.php`, que gestiona la conexión a la base de datos mediante PDO.

5.8.3 Eliminación de un Equipo

El archivo recibe el ID del equipo a eliminar a través de una solicitud `POST`. El parámetro `id` representa el número de serie del equipo que se va a eliminar.

5.8.4 Consulta SQL de Eliminación

La consulta SQL preparada es:

```
DELETE FROM equipo WHERE numero_ser = :id
```

Esta consulta elimina el equipo que tiene el número de serie proporcionado en el parámetro `id`. El parámetro es vinculado a la consulta utilizando `bindParam` para asegurar la seguridad de la consulta y evitar inyecciones SQL.

5.8.5 Ejecutando la Consulta

Si la consulta se ejecuta correctamente, el sistema devuelve el mensaje `success`. Si la ejecución falla por cualquier motivo, se devuelve un mensaje de error. En caso de que se detecte una excepción durante la ejecución de la consulta, se captura y muestra un mensaje de error con la descripción del problema.

5.8.6 Flujo de Navegación

El flujo de navegación se produce de la siguiente manera:

1. El usuario envía una solicitud `POST` para eliminar un equipo especificando el número de serie del equipo a través del parámetro `id`.
2. El archivo `mod-delete_equipos.php` valida la existencia del `id` y ejecuta la consulta SQL `DELETE` correspondiente.
3. Si la eliminación es exitosa, se devuelve el mensaje `success`. Si hay un error, se devuelve un mensaje de error con la causa.

5.8.7 Manejo de Errores

Si no se proporciona el `id` o si ocurre algún problema durante la ejecución de la consulta, se maneja mediante los siguientes mensajes:

- `Error al eliminar el equipo.`: Este mensaje se muestra si la consulta no se ejecuta correctamente.
- `Error: [mensaje de error del sistema].`: Si ocurre un error durante la ejecución de la consulta, se captura la excepción y se muestra el mensaje de error específico.
- `ID del equipo no proporcionado.`: Este mensaje se muestra si no se ha enviado el `id` en la solicitud.

5.9 Módulo: `bkend-consult_conexiones.php`

Este archivo PHP proporciona funciones para gestionar las conexiones de equipos almacenadas en una base de datos. Permite obtener datos de la tabla `conexion`, actualizar conexiones existentes, y eliminar conexiones específicas mediante el uso de consultas SQL preparadas y controladas por PDO.

5.9.1 Conexión a la Base de Datos

El archivo incluye el archivo `conn.php`, que gestiona la conexión a la base de datos mediante PDO.

5.9.2 Funciones Definidas

5.9.3 1. Obtener Todas las Conexiones

```
function getConexiones($connection) {
    $sql = "SELECT * FROM conexion";
    $stmt = $connection->prepare($sql);

    // Verificar errores en la ejecución de la consulta
    if (!$stmt->execute()) {
        echo "Error ejecutando la consulta: " . $stmt->errorInfo()[2];
        return false; // Devolver false en caso de error
    }

    return $stmt->fetchAll(PDO::FETCH_ASSOC);
}
```

Esta función obtiene todos los registros de la tabla `conexion`. Si la ejecución de la consulta falla, se muestra un mensaje de error.

5.9.4 2. Obtener una Conexión Específica por ART_NO

```
function getConexionByArtNo($connection, $art_no) {
    $sql = "SELECT * FROM conexion WHERE ART_NO = :art_no";
    $stmt = $connection->prepare($sql);

    // Verificar errores en la ejecución de la consulta
    if (!$stmt->execute([':art_no' => $art_no])) {
        echo "Error ejecutando la consulta: " . $stmt->errorInfo()[2];
        return false;
    }

    return $stmt->fetch(PDO::FETCH_ASSOC);
}
```

Esta función obtiene una conexión específica de la tabla `conexion` mediante el parámetro `ART_NO`.

5.9.5 3. Actualizar una Conexión

```
function updateConexion($connection, $art_no, $posicionx, $etiqueta, $conector, $descripcion1, $descripc
    $sql = "UPDATE conexion
            SET POSICIONX = :posicionx, ETIQUETA = :etiqueta, CONECTOR = :conector,
                DESCRIP1 = :descripcion1, DESCRIP2 = :descripcion2, EXISTENCIA = :existencia
            WHERE ART_NO = :art_no";
    $stmt = $connection->prepare($sql);

    // Verificar errores en la ejecución de la consulta
    if (!$stmt->execute([
        ':art_no' => $art_no,
        ':posicionx' => $posicionx,
        ':etiqueta' => $etiqueta,
        ':conector' => $conector,
        ':descripcion1' => $descripcion1,
        ':descripcion2' => $descripcion2,
        ':existencia' => $existencia
    ])) {
        echo "Error actualizando la conexión: " . $stmt->errorInfo()[2];
    }
}
```

Esta función actualiza los datos de una conexión específica en la tabla `conexion`. Recibe los parámetros necesarios para la actualización de los campos correspondientes.

5.9.6 4. Eliminar una Conexión

```
function deleteConexion($connection, $art_no) {
    $sql = "DELETE FROM conexion WHERE ART_NO = :art_no";
    $stmt = $connection->prepare($sql);

    // Verificar errores en la ejecución de la consulta
    if (!$stmt->execute([':art_no' => $art_no])) {
        echo "Error eliminando la conexión: " . $stmt->errorInfo()[2];
    }
}
```

Esta función elimina una conexión de la base de datos utilizando el `ART_NO` de la conexión a eliminar.

5.9.7 5. Procesar Solicitudes de Edición o Eliminación

```
function processRequest($connection) {
    if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        // Verificar si se ha solicitado una edición
        if (isset($_POST['art_no'], $_POST['posicionx'], $_POST['etiqueta'], $_POST['conector'], $_POST[
            $art_no = $_POST['art_no'];
            $posicionx = $_POST['posicionx'];
            $etiqueta = $_POST['etiqueta'];
            $conector = $_POST['conector'];
            $descripcion1 = $_POST['descripcion1'];
            $descripcion2 = $_POST['descripcion2'];
            $existencia = $_POST['existencia'];

            updateConexion($connection, $art_no, $posicionx, $etiqueta, $conector, $descripcion1, $descr
            echo "success"; // Responder con éxito al frontend
        }
        // Verificar si se ha solicitado una eliminación
        elseif (isset($_POST['id'])) {
            $art_no = $_POST['id'];
            deleteConexion($connection, $art_no);
            echo "success"; // Responder con éxito al frontend
        }
    }
}
```

La función `processRequest` maneja tanto las solicitudes de actualización como de eliminación de conexiones. Si se envía una solicitud `POST` con los parámetros adecuados, se realiza la actualización o eliminación correspondiente y se responde con el mensaje `success`.

5.9.8 Flujo de Navegación

El flujo de ejecución es el siguiente:

1. Se recibe una solicitud `POST` desde el frontend con los parámetros necesarios para la operación (actualización o eliminación).
2. Si se trata de una solicitud de edición, se actualiza la conexión con los datos proporcionados.
3. Si se trata de una solicitud de eliminación, se elimina la conexión correspondiente.
4. Se responde con `success` al frontend.

5.9.9 Manejo de Errores

Si ocurre algún error en la ejecución de las consultas, se devuelve un mensaje de error que incluye la descripción proporcionada por PDO en `errorInfo()[2]`. Además, si no se reciben los parámetros esperados, no se realiza ninguna acción y no se devuelve ningún resultado.

5.10 Formulario: `bkend-edit_conexiones.php`

Este archivo PHP es responsable de gestionar la actualización de las conexiones en la base de datos. Permite a los usuarios modificar los detalles de una conexión existente en el sistema mediante una solicitud POST. Los datos enviados son validados, sanitizados y luego procesados para actualizar la base de datos.

5.10.1 Estructura y Funcionalidad

5.10.2 Conexiones y Consultas

El archivo incluye un bloque de código para gestionar la actualización de una conexión en la base de datos. Este bloque realiza lo siguiente:

- Incluye el archivo de conexión a la base de datos (`conn.php`), necesario para interactuar con la base de datos.
- Verifica que la solicitud HTTP sea del tipo POST.
- Recoge y sanitiza los datos enviados por el cliente a través de la solicitud POST.
- Realiza una validación para asegurar que todos los datos necesarios estén presentes.
- Si los datos son válidos, ejecuta una consulta SQL para actualizar la conexión en la base de datos.
- Si se realiza la actualización correctamente, se devuelve un mensaje de éxito; de lo contrario, se devuelve un mensaje de error.

5.10.3 Procesamiento de Solicitudes

El archivo verifica si la solicitud HTTP es del tipo POST, lo cual indica que el usuario está intentando enviar nuevos datos para actualizar una conexión. Si la solicitud no es POST, se muestra un mensaje de error indicando que el método no está permitido.

5.10.4 Mostrar Conexiones

El archivo no gestiona la visualización de las conexiones, ya que está destinado únicamente a la actualización de datos en la base de datos. El resultado de la actualización es indicado mediante una respuesta que puede ser `success` o un mensaje de error.

5.10.5 Edición de Conexiones

Este formulario permite la edición de una conexión a través de los siguientes pasos:

1. El usuario envía los datos de una conexión mediante una solicitud POST.
2. Los datos enviados son validados y sanitizados.
3. Si los datos son válidos, se ejecuta una consulta SQL para actualizar los registros en la base de datos.
4. Si la actualización es exitosa, se muestra el mensaje `success`; si no, se devuelve un error indicando que no se encontraron cambios.

5.10.6 Eliminación de Conexiones

El archivo no incluye funcionalidades de eliminación de conexiones. El proceso está centrado únicamente en la actualización de registros existentes.

5.10.7 Interfaz de Usuario

Este archivo no define directamente la interfaz de usuario, ya que su propósito es gestionar la lógica del lado del servidor para la actualización de una conexión. La interfaz de usuario debe ser gestionada en otro archivo PHP o en el lado del cliente mediante JavaScript.

Barra de Navegación No se incluye una barra de navegación en este archivo.

Contenedor Principal El archivo no gestiona el contenedor principal, ya que se limita a procesar los datos enviados por el cliente.

Paginación Este archivo no incluye funciones de paginación, ya que su único propósito es procesar y actualizar los datos enviados mediante POST.

5.10.8 Estilos y Diseño

No se incluyen estilos ni diseño visual, ya que el archivo está enfocado exclusivamente en la parte lógica del servidor. La parte visual debe ser gestionada por otros archivos.

5.10.9 Flujo de Navegación

El usuario puede interactuar con la página de la siguiente manera:

1. Enviar los datos de una conexión mediante una solicitud POST.
2. Si los datos son válidos, se actualiza la conexión en la base de datos.
3. Recibir una respuesta de éxito (`success`) o error.

El flujo de trabajo está orientado únicamente al procesamiento de solicitudes de actualización de datos.

5.10.10 Scripts y Funciones

Este archivo no incluye funciones JavaScript, ya que está centrado en la lógica del servidor para procesar solicitudes POST. Las funciones de interfaz de usuario deben ser gestionadas en otros archivos.

5.10.11 Mensajes de Estado

El archivo muestra mensajes de estado en forma de texto para indicar el éxito o el error al procesar la actualización de la conexión.

5.11 Formulario: `bkend-delete_conexiones.php`

Este archivo PHP es responsable de gestionar la eliminación de una conexión de la base de datos. Permite a los usuarios eliminar una conexión específica mediante un parámetro `ART_NO`, que es recibido a través de una solicitud POST.

5.11.1 Estructura y Funcionalidad

5.11.2 Conexiones y Consultas

El archivo incluye una función llamada `deleteConexion()` que realiza los siguientes pasos:

- Incluye el archivo de conexión a la base de datos (`conn.php`).
- Define la función `deleteConexion()` que recibe la conexión a la base de datos y el identificador `ART_NO`.
- Prepara y ejecuta una consulta SQL para eliminar la conexión especificada mediante el `ART_NO`.
- Si la consulta se ejecuta correctamente, la función devuelve `true`; si no, devuelve `false` y muestra un mensaje de error.

5.11.3 Procesamiento de Solicitudes

El archivo verifica si el parámetro `ART_NO` ha sido enviado mediante una solicitud POST. Si el parámetro está presente, intenta eliminar la conexión correspondiente. Si la eliminación es exitosa, se muestra un mensaje de éxito (`success`); si ocurre un error, se muestra un mensaje de error.

5.11.4 Mostrar Conexiones

Este archivo no se encarga de mostrar las conexiones, sino únicamente de procesar la eliminación de una conexión específica según el `ART_NO` enviado.

5.11.5 Edición de Conexiones

Este archivo no incluye funcionalidades de edición, ya que su único propósito es eliminar registros de conexiones en la base de datos.

5.11.6 Eliminación de Conexiones

Este formulario permite la eliminación de una conexión específica mediante los siguientes pasos:

1. El usuario envía el `ART_NO` de la conexión que desea eliminar mediante una solicitud POST.
2. La función `deleteConexion()` ejecuta una consulta SQL para eliminar la conexión correspondiente de la base de datos.
3. Si la eliminación es exitosa, se muestra el mensaje `success`; si no, se muestra un mensaje de error indicando el fallo en la eliminación.

5.11.7 Interfaz de Usuario

Este archivo no gestiona directamente la interfaz de usuario, ya que está orientado a la lógica de la eliminación en el servidor. La interfaz debe ser proporcionada por otros archivos PHP o en el lado del cliente mediante JavaScript.

Barra de Navegación No se incluye una barra de navegación en este archivo.

Contenedor Principal El archivo no gestiona el contenedor principal, ya que su tarea es procesar los datos enviados por el cliente para eliminar una conexión de la base de datos.

Paginación Este archivo no incluye funciones de paginación, ya que su único propósito es eliminar conexiones.

5.11.8 Estilos y Diseño

No se incluyen estilos ni diseño visual, ya que el archivo está enfocado únicamente en la parte lógica del servidor. La parte visual debe ser gestionada por otros archivos.

5.11.9 Flujo de Navegación

El usuario puede interactuar con la página de la siguiente manera:

1. Enviar el parámetro `ART_NO` mediante una solicitud POST.
2. Si el parámetro está presente, se intenta eliminar la conexión correspondiente.
3. Recibir una respuesta de éxito (`success`) o error.

El flujo de trabajo está orientado únicamente al procesamiento de solicitudes para eliminar una conexión.

5.11.10 Scripts y Funciones

Este archivo no incluye funciones JavaScript, ya que está centrado en la lógica del servidor para procesar solicitudes POST. Las funciones de interfaz de usuario deben ser gestionadas en otros archivos.

5.11.11 Mensajes de Estado

El archivo muestra mensajes de estado en forma de texto para indicar el éxito o el error al procesar la eliminación de la conexión.

5.12 Formulario: `backend-check_user_id.php`

Este archivo PHP se utiliza para verificar la existencia de un usuario en la base de datos mediante un identificador único `user-id` (en este caso, `NUMERO`). Dependiendo de si el usuario existe o no, redirige al formulario de edición de usuario o al formulario para agregar un nuevo usuario.

5.12.1 Estructura y Funcionalidad

5.12.2 Conexiones y Consultas

El archivo realiza los siguientes pasos:

- Se incluye el archivo de conexión a la base de datos (`conn.php`).
- Se recibe el parámetro `user-id` desde una solicitud POST, el cual se sanitiza para prevenir inyecciones de código.
- Se prepara una consulta SQL para buscar el `user-id` en la tabla `usuarios`.
- Si el `user-id` existe en la base de datos, el archivo redirige al usuario al formulario de edición de usuario (`form-edit_user.php`).
- Si el `user-id` no existe, el archivo redirige al usuario al formulario de agregar usuario (`form-add_user.php`).

5.12.3 Procesamiento de Solicitudes

El archivo procesa la solicitud POST enviada por el usuario. Primero, se sanitiza el `user-id` para evitar cualquier tipo de ataque de inyección SQL. Luego, se ejecuta una consulta SQL para verificar si el `user-id` existe en la base de datos. Dependiendo del resultado, se realiza una redirección a uno de los formularios correspondientes.

5.12.4 Mostrar Conexiones

Este archivo no muestra ninguna conexión o información detallada sobre las conexiones. Su única función es verificar si un `user-id` existe en la base de datos y redirigir al formulario adecuado.

5.12.5 Edición de Conexiones

El archivo redirige a un formulario de edición de usuario en caso de que el `user-id` exista en la base de datos. Esto permite a los usuarios editar su información en el sistema.

5.12.6 Eliminación de Conexiones

El archivo no incluye funcionalidades de eliminación de conexiones o usuarios. Su propósito es solo verificar la existencia del usuario y redirigir a los formularios correspondientes.

5.12.7 Interfaz de Usuario

Este archivo no gestiona la interfaz de usuario directamente. Solo se encarga de verificar si un `user-id` existe y de redirigir al formulario adecuado.

Barra de Navegación Este archivo no incluye una barra de navegación, ya que está orientado a la validación de la existencia de un `user-id` y la posterior redirección.

Contenedor Principal Este archivo no gestiona un contenedor principal ni ningún tipo de contenido visual. Solo realiza el procesamiento de la solicitud y redirige según el resultado de la consulta.

Paginación Este archivo no incluye controles de paginación, ya que no gestiona la visualización de registros de usuarios, sino solo la validación de la existencia de un `user-id`.

5.12.8 Estilos y Diseño

No se aplican estilos ni diseño visual en este archivo. El archivo es puramente funcional, diseñado para procesar el `user-id` y realizar redirecciones según la existencia del mismo.

5.12.9 Flujo de Navegación

El usuario interactúa con este archivo de la siguiente manera:

1. El usuario envía un `user-id` mediante una solicitud POST.
2. El archivo valida si el `user-id` existe en la base de datos.
3. Si el `user-id` existe, el archivo redirige al usuario al formulario de edición de usuario (`form-edit_user.php`).
4. Si el `user-id` no existe, el archivo redirige al usuario al formulario para añadir un nuevo usuario (`form-add_user.php`).

5.12.10 Scripts y Funciones

El archivo incluye una sencilla funcionalidad de validación y redirección:

- `filter_input()`: Sanitize y validar el `user-id` enviado por el cliente para prevenir inyecciones de código.
- Consulta SQL para verificar la existencia del `user-id` en la tabla `usuarios`.
- `header()`: Redirección al formulario adecuado dependiendo de si el `user-id` existe en la base de datos.

Estas funciones permiten una interacción fluida y segura, asegurando que el flujo de trabajo del usuario se gestione correctamente.

5.13 Formulario: `backend-add_user.php`

Este archivo PHP se encarga de procesar un formulario enviado por POST para agregar un nuevo usuario a la base de datos. Los datos del usuario se reciben del formulario, se validan y se insertan en la base de datos de forma segura.

5.13.1 Estructura y Funcionalidad

5.13.2 Conexiones y Consultas

Este archivo realiza lo siguiente:

- Se incluye el archivo de conexión a la base de datos (`conn.php`).
- Se recoge la información enviada por el formulario a través de `_POST`.
- Se realizan validaciones sobre los datos del formulario (como verificar que las contraseñas coincidan y que el PIN sea numérico).
- Se utiliza `password_hash()` para encriptar la contraseña antes de almacenarla en la base de datos.
- Se prepara una consulta SQL para insertar el nuevo usuario en la tabla `usuarios`.
- Si la inserción es exitosa, se redirige al usuario a una página de éxito (`success.php`).

5.13.3 Procesamiento de Solicitudes

El archivo procesa una solicitud POST enviada por un formulario. Recibe los datos de usuario (`user-id`, `user-name`, `user-level`, etc.) y realiza lo siguiente:

- Verifica que las contraseñas coincidan. Si no es así, se detiene el proceso y muestra un mensaje de error.
- Verifica que el PIN (`user-password`) sea un valor numérico.
- Utiliza `password_hash()` para encriptar la contraseña antes de almacenarla en la base de datos.

- Ejecuta la consulta SQL para insertar el nuevo usuario en la base de datos.
- Si la consulta es exitosa, redirige al usuario a `success.php`.
- Si la consulta falla, se captura el error mediante `try-catch` y se muestra un mensaje con la descripción del error.

5.13.4 Mostrar Conexiones

Este archivo no tiene la función de mostrar las conexiones. Su único objetivo es procesar los datos del formulario y agregar un nuevo usuario a la base de datos.

5.13.5 Edición de Conexiones

Este archivo no realiza ninguna edición de conexiones ni modificaciones en los datos de los usuarios existentes. Su función se limita a insertar nuevos usuarios.

5.13.6 Eliminación de Conexiones

El archivo no incluye ninguna funcionalidad de eliminación de usuarios o conexiones.

5.13.7 Interfaz de Usuario

Este archivo no gestiona la interfaz de usuario directamente. Se enfoca en recibir y procesar los datos enviados desde un formulario y realizar la inserción en la base de datos.

Barra de Navegación El archivo no maneja una barra de navegación, ya que su único propósito es procesar los datos del formulario de alta de usuarios.

Contenedor Principal No tiene un contenedor principal visual, ya que está destinado a ser un archivo PHP puramente funcional.

Paginación No se incluye funcionalidad de paginación, ya que el archivo solo gestiona la inserción de un único usuario en la base de datos.

5.13.8 Estilos y Diseño

Este archivo no tiene relación con el diseño visual ni con estilos CSS. Su función es procesar los datos recibidos desde un formulario y manejar la inserción del nuevo usuario en la base de datos de manera segura.

5.13.9 Flujo de Navegación

El flujo de navegación es el siguiente:

1. El usuario envía un formulario con los datos del nuevo usuario.
2. El archivo valida que las contraseñas coincidan y que el PIN sea numérico.
3. Si la validación es exitosa, se inserta el nuevo usuario en la base de datos.
4. Si la inserción es exitosa, el archivo redirige al usuario a `success.php`.
5. Si ocurre un error, se captura y muestra el mensaje de error.

5.13.10 Scripts y Funciones

Este archivo utiliza las siguientes funciones:

- `filter_input()`: Para sanitizar los datos recibidos (aunque no se utiliza directamente en este archivo).
- `password_hash()`: Para encriptar la contraseña antes de almacenarla en la base de datos.
- `PDO::prepare()`: Para preparar la consulta SQL de inserción.

- `PDO::execute()`: Para ejecutar la consulta SQL con los datos proporcionados.
- `header()`: Redirige a la página de éxito si la inserción es exitosa.
- `die()`: Muestra un mensaje de error si las contraseñas no coinciden o si la solicitud no es válida.

Estas funciones aseguran que el proceso de registro sea seguro y eficiente. Además, el uso de `password_hash()` garantiza que las contraseñas se almacenen de manera segura, protegiendo la información sensible de los usuarios.

5.14 Formulario: `backend-edit_user.php`

Este archivo PHP procesa un formulario de edición de usuario y permite actualizar la información del usuario en la base de datos. Dependiendo de si se cambia la contraseña, la actualización puede incluir el nuevo PIN cifrado.

5.14.1 Estructura y Funcionalidad

5.14.2 Conexiones y Consultas

Este archivo realiza lo siguiente:

- Se incluye el archivo de conexión a la base de datos (`conn.php`).
- Se recogen los datos enviados por el formulario de edición mediante `_POST`.
- Se validan las contraseñas, asegurándose de que coincidan antes de realizar la actualización.
- Si se ha cambiado la contraseña, se valida que el nuevo PIN sea numérico y se cifra con `password_hash()` antes de almacenarlo.
- Si no se cambia la contraseña, la consulta de actualización no incluye el campo de contraseña.
- Se prepara y ejecuta una consulta SQL para actualizar los datos del usuario en la base de datos.
- Si la actualización es exitosa, el usuario es redirigido a una página de éxito (`success.php`).

5.14.3 Procesamiento de Solicitudes

El archivo procesa una solicitud POST enviada desde un formulario de edición de usuario. Recibe los datos del formulario (`user-id`, `user-name`, `user-level`, etc.) y realiza lo siguiente:

- Recoge y sanitiza los valores del formulario con `filter_input()` para prevenir posibles vulnerabilidades de inyección de código.
- Si se cambia la contraseña, se verifica que las contraseñas coincidan y que el nuevo PIN sea numérico.
- Si la contraseña se cambia, se utiliza `password_hash()` para encriptar la nueva contraseña antes de almacenarla en la base de datos.
- Si no se cambia la contraseña, el archivo actualiza los otros campos del usuario sin tocar el campo de la contraseña.
- Se prepara la consulta SQL para actualizar los datos del usuario, incluyendo o no la contraseña dependiendo de si se cambió.
- Ejecuta la consulta de actualización con los valores del formulario. Si la actualización es exitosa, se redirige al usuario a `success.php`.
- Si ocurre un error durante la actualización, se captura el error con `try-catch` y se muestra el mensaje de error correspondiente.

5.14.4 Mostrar Conexiones

Este archivo no tiene la función de mostrar las conexiones. Se enfoca en actualizar los datos de un usuario específico, según su ID.

5.14.5 Edición de Conexiones

El archivo es responsable de la edición de los datos de un usuario. Dependiendo de si se modifica o no la contraseña, la actualización puede implicar el cambio de varios campos en la tabla `usuarios`, como el nombre, nivel, faltas, estado, correo electrónico y teléfono. Si se proporciona un nuevo PIN, este se encripta antes de ser almacenado.

5.14.6 Eliminación de Conexiones

Este archivo no realiza ninguna eliminación de datos. Solo permite la actualización de los datos del usuario.

5.14.7 Interfaz de Usuario

El archivo no gestiona directamente la interfaz de usuario. Se enfoca en procesar los datos de un formulario de edición de usuario.

Barra de Navegación Este archivo no incluye una barra de navegación. Su único propósito es procesar los datos de edición de un usuario.

Contenedor Principal El archivo no tiene un contenedor principal visual, ya que su propósito es puramente funcional.

Paginación No se incluye funcionalidad de paginación, ya que el archivo solo actualiza los datos de un usuario específico y no interactúa con una lista de usuarios.

5.14.8 Estilos y Diseño

El archivo no tiene relación con el diseño visual ni con estilos CSS. Su función es procesar los datos del formulario de edición y manejar la actualización de los datos del usuario en la base de datos de manera segura.

5.14.9 Flujo de Navegación

El flujo de navegación es el siguiente:

1. El usuario envía un formulario con los datos actualizados.
2. El archivo valida que las contraseñas coincidan y que el PIN sea numérico.
3. Si la contraseña se cambia, se cifra el PIN antes de almacenarlo. Si no se cambia, solo se actualizan los otros campos.
4. El archivo ejecuta la consulta de actualización de los datos del usuario.
5. Si la actualización es exitosa, el archivo redirige al usuario a `success.php`.
6. Si ocurre un error, se captura y muestra el mensaje de error.

5.14.10 Scripts y Funciones

Este archivo utiliza las siguientes funciones:

- `filter_input()`: Para sanitizar los datos recibidos y prevenir inyecciones de código.
- `password_hash()`: Para encriptar la contraseña antes de almacenarla en la base de datos.
- `PDO::prepare()`: Para preparar la consulta SQL de actualización.

- `PDO::execute()`: Para ejecutar la consulta SQL con los valores proporcionados.
- `header()`: Redirige a la página de éxito si la actualización es exitosa.
- `die()`: Muestra un mensaje de error si las contraseñas no coinciden o si la solicitud no es válida.

Estas funciones aseguran que el proceso de actualización sea seguro, verificando las contraseñas y cifrando el PIN antes de almacenarlo, manteniendo la seguridad de la información del usuario.

5.15 Formulario: `mod_fetch-user-data.php`

Este archivo PHP procesa una solicitud para obtener los datos de un usuario específico a partir de su ID, que se pasa a través de la URL. Los datos se devuelven en formato JSON.

5.15.1 Estructura y Funcionalidad

5.15.2 Conexiones y Consultas

El archivo realiza lo siguiente:

- Se incluye el archivo de conexión a la base de datos (`conn.php`).
- Se obtiene el ID del usuario desde la URL utilizando `filter_input()` y se sanitiza correctamente para evitar inyecciones de código.
- Si el ID del usuario es válido, se prepara una consulta SQL para recuperar los datos del usuario de la tabla `usuarios`.
- Si el usuario existe en la base de datos, los datos se devuelven en formato JSON.
- Si el usuario no existe, se devuelve un mensaje de error en formato JSON.
- Si ocurre cualquier excepción durante la ejecución del código, se captura y se devuelve un mensaje de error en formato JSON.

5.15.3 Procesamiento de Solicitudes

El archivo procesa una solicitud GET enviada con un parámetro `user_id`. El flujo de procesamiento es el siguiente:

- El ID de usuario se obtiene de la URL utilizando `filter_input()` para garantizar que sea seguro.
- Si se proporciona un ID válido, el archivo ejecuta una consulta SQL para recuperar los datos del usuario de la base de datos.
- Si el usuario existe en la base de datos, los datos se retornan en formato JSON con `json_encode()`.
- Si el usuario no se encuentra en la base de datos, se devuelve un mensaje de error en formato JSON.
- Si ocurre un error durante el proceso, se captura la excepción y se devuelve el mensaje de error correspondiente en formato JSON.

5.15.4 Mostrar Conexiones

Este archivo no maneja la visualización directa de datos. Su propósito es recuperar y devolver los datos de un usuario específico en formato JSON.

5.15.5 Edición de Conexiones

Este archivo no realiza ediciones en los datos del usuario. Solo se encarga de obtener los datos de un usuario específico basado en el ID proporcionado.

5.15.6 Eliminación de Conexiones

Este archivo no incluye ninguna funcionalidad para eliminar conexiones. Su única función es obtener los datos de un usuario.

5.15.7 Interfaz de Usuario

Este archivo no maneja directamente la interfaz de usuario, ya que su objetivo es procesar y devolver los datos en formato JSON. La interfaz debe ser gestionada en otro lugar.

Barra de Navegación Este archivo no tiene una barra de navegación, ya que está destinado solo a la función de backend para recuperar los datos de un usuario.

Contenedor Principal El archivo no presenta una estructura visual. Su única salida es un JSON con los datos del usuario o un mensaje de error.

Paginación No se incluye paginación, ya que el archivo solo maneja la recuperación de un único usuario según su ID.

5.15.8 Estilos y Diseño

Este archivo no se ocupa del diseño o los estilos visuales. Se enfoca en el procesamiento de datos en el backend y la entrega de información en formato JSON.

5.15.9 Flujo de Navegación

El flujo de navegación es el siguiente:

1. El cliente realiza una solicitud GET con el parámetro `user_id` en la URL.
2. El archivo sanitiza el ID recibido y lo usa para ejecutar una consulta SQL.
3. Si el usuario existe, los datos se devuelven en formato JSON.
4. Si no existe, se devuelve un mensaje de error en formato JSON.
5. Si ocurre algún error en el proceso, se captura y devuelve un mensaje de error en formato JSON.

5.15.10 Scripts y Funciones

Este archivo utiliza las siguientes funciones:

- `filter_input()`: Para sanitizar el ID del usuario recibido a través de la URL.
- `PDO::prepare()`: Para preparar la consulta SQL.
- `PDO::execute()`: Para ejecutar la consulta SQL.
- `json_encode()`: Para devolver los resultados en formato JSON.
- `header()`: Para especificar el tipo de contenido como JSON en la respuesta.

Estas funciones permiten procesar de manera eficiente la solicitud GET, recuperar los datos del usuario desde la base de datos y devolverlos en formato JSON. En caso de error, se capturan excepciones y se proporcionan mensajes de error detallados.

5.16 Página de éxito: `bkend-success.php`

Esta página web es un mensaje de confirmación que se muestra después de realizar cambios exitosos en el sistema. Informa al usuario que los cambios han sido realizados con éxito y que será redirigido a la página principal.

5.16.1 Estructura y Funcionalidad

5.16.2 Estructura HTML

La página está estructurada de la siguiente manera:

- Se establece el tipo de documento HTML y el idioma en inglés (`lang="en"`).
- En el `head`, se define la codificación de caracteres `UTF-8` y la configuración de la vista para dispositivos móviles.
- Se incluye un `meta` para redirigir automáticamente a la página principal después de 3 segundos. Si la redirección no ocurre, el usuario puede hacer clic en un enlace para ir manualmente.
- Se incluyen los archivos de estilos CSS de Bootstrap para darle estilo a la página y una hoja de estilos personalizada (`styles-success.css`) para la página de éxito.

5.16.3 Interfaz de Usuario

La página presenta un mensaje claro para informar al usuario sobre el éxito de la operación:

- Un encabezado con el texto `"¡Cambios realizados exitosamente!"`.
- Un mensaje adicional que informa que el usuario será redirigido en 3 segundos a la página principal.
- Un enlace que permite al usuario redirigirse manualmente a la página principal si la redirección automática falla.

5.16.4 Estilos y Diseño

El diseño de la página está basado en Bootstrap, lo que le da un diseño limpio y sencillo. Además, se utiliza una hoja de estilos personalizada `styles-success.css` para aplicar el estilo específico de la página de éxito.

5.16.5 Redirección Automática

El comportamiento principal de la página es redirigir al usuario a la página principal (`index.html`) después de 3 segundos. Esto se logra mediante el siguiente `meta`:

```
<meta http-equiv="refresh" content="3;url=../../index.html">
```

Si la redirección automática no funciona, el usuario puede hacer clic en el enlace `"haz clic aquí"` para ir a la página principal.

5.16.6 Flujo de Navegación

El flujo de navegación es el siguiente:

1. El usuario realiza cambios en el sistema (por ejemplo, crea, edita o elimina un usuario).
2. Después de que la operación se completa correctamente, el sistema redirige al usuario a esta página de confirmación.
3. El usuario ve el mensaje de éxito y la redirección automática en 3 segundos.
4. Si la redirección no se lleva a cabo automáticamente, el usuario puede hacer clic en el enlace para ir a la página principal manualmente.

5.16.7 Interactividad

La página tiene un comportamiento muy sencillo. Solo presenta un mensaje y realiza la redirección automáticamente, por lo que no tiene una interactividad compleja.

5.16.8 Scripts y Funciones

La página no incluye scripts adicionales, ya que el comportamiento esperado se logra a través de HTML y la redirección con el `meta` tag.

5.16.9 Estilos Personalizados

El archivo `styles-success.css` probablemente contiene reglas CSS para ajustar el estilo visual de la página. Esta hoja de estilos es usada para mejorar la apariencia del mensaje y asegurarse de que se vea bien en todos los dispositivos.