Manual Técnico

Jessica Carolina Osorio Hernández (ID 100112053 / josori10@ibero.edu.co)
Duván Arley Hernández Niño (ID 100110002 / dherna84@ibero.edu.co)

Corporación Universitaria Iberoamericana
Ingeniería de Software
Ingeniera: José Castro

Villeta Cundinamarca
Diciembre 2023

IBEROAMERICANA
CORPORACIÓN UNIVERSITARIA

Introducción

Este manual técnico proporciona una visión detallada de la implementación y la estructura de código de la APP-WEB Lenguaje de Señas desarrollada. Incluye detalles sobre las tecnologías utilizadas, la arquitectura general y las funcionalidades clave de la aplicación.

Objetivos

- Describir la estructura del código fuente.

- Explicar las funcionalidades principales y cómo están implementadas.

- Detallar la conexión a la base de datos y el manejo de datos.

- Proporcionar información detallada para que otros desarrolladores comprendan y modifiquen el código si es necesario.

Tecnologías Utilizadas

- HTML

- CSS

- jQuery

- PHP

- MySQL

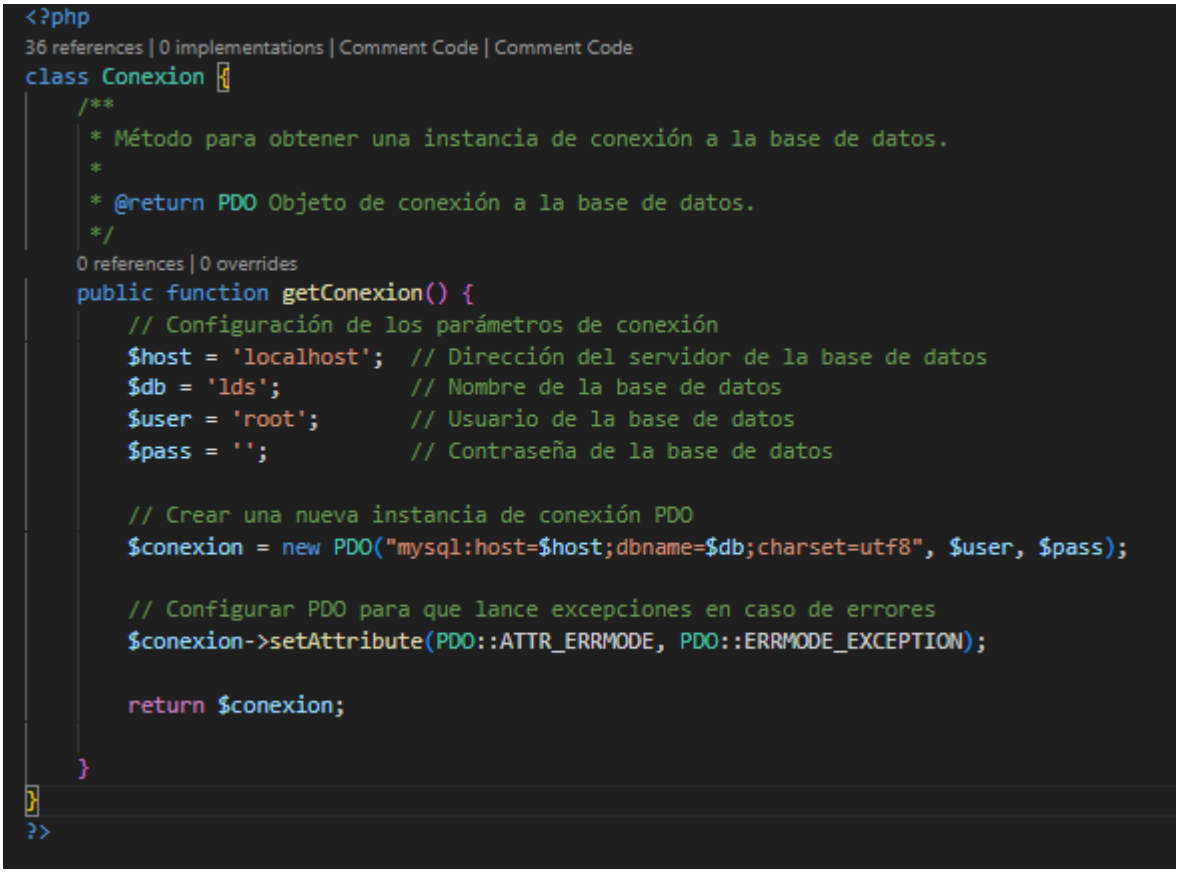
IBEROAMERICANA
CORPORACIÓN UNIVERSITARIA

Estructura del Código Fuente:

Conexión DB

PHP (conexión.php)

En el archivo PHP conexion.php, se encuentra la clase conexion, que se encarga de establecer una conexión segura a la base de datos MySQL utilizando PDO (PHP Data Objects). Aquí está una explicación detallada de cada parte de la clase:



```php
<?php
36 references | 0 implementations | Comment Code | Comment Code
class Conexion {
    /**
     * Método para obtener una instancia de conexión a la base de datos.
     *
     * @return PDO Objeto de conexión a la base de datos.
     */
    0 references | 0 overrides
    public function getConexion() {
        // Configuración de los parámetros de conexión
        $host = 'localhost';  // Dirección del servidor de la base de datos
        $db = 'lds';          // Nombre de la base de datos
        $user = 'root';       // Usuario de la base de datos
        $pass = '';           // Contraseña de la base de datos

        // Crear una nueva instancia de conexión PDO
        $conexion = new PDO("mysql:host=$host;dbname=$db;charset=utf8", $user, $pass);

        // Configurar PDO para que lance excepciones en caso de errores
        $conexion->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

        return $conexion;

    }
}
?>
```

IBEROAMERICANA
CORPORACIÓN UNIVERSITARIA

Registro de Usuarios

HTML (index.html)

En el archivo HTML index.html, se define la estructura de la página de registro. Este formulario simple recopila información esencial del usuario, como nombre, correo electrónico y contraseña. Además, se vinculan los archivos CSS y jQuery necesarios para el diseño y la funcionalidad del formulario.

```
<form id="registerForm">
  <div class="text-center mb-3">
    <p>Sign up with:</p>
  </div>

  <!-- Name input -->
  <div data-mdb-input-init class="form-outline mb-4">
    <input type="text" id="registerName" class="form-control" name="name" required />
    <label class="form-label" for="registerName">Name</label>
  </div>

  <!-- Username input -->
  <div data-mdb-input-init class="form-outline mb-4">
    <input type="text" id="registerUsername" class="form-control" name="username" required />
    <label class="form-label" for="registerUsername">Username</label>
  </div>

  <!-- Email input -->
  <div data-mdb-input-init class="form-outline mb-4">
    <input type="email" id="registerEmail" class="form-control" name="mail" required />
    <label class="form-label" for="registerEmail">Email</label>
  </div>

  <!-- Password input -->
  <div data-mdb-input-init class="form-outline mb-4">
    <input type="password" id="registerPassword" class="form-control" name="pass" required/>
    <label class="form-label" for="registerPassword">Password</label>
  </div>

  <!-- Repeat Password input -->
  <div data-mdb-input-init class="form-outline mb-4" id="cntpr">
    <input
      type="password"
      id="registerRepeatPassword"
      class="form-control" required
    />
    <label class="form-label" for="registerRepeatPassword"
      >Repeat password</label
    >
```

IBEROAMERICANA
CORPORACIÓN UNIVERSITARIA

jQuery (login.js)

En el archivo jQuery login.js, se utiliza la biblioteca jQuery para manejar la interactividad del formulario. La función principal se activa cuando el formulario se envía y evita que la página se recargue. Recopila los datos del formulario y los envía al servidor utilizando una solicitud AJAX, lo que permite la actualización de la página sin necesidad de recarga completa.

```
$("#Register").click(function() {
    // Realizar validaciones
    if (validateForm()) {
      // Si las validaciones son exitosas, enviar el formulario por Ajax
      $.ajax({
        url: "controller/querys/create/user.php",
        method: "POST",
        data: $("#registerForm").serialize(),
        success: function(response) {
          // Manejar la respuesta del servidor aquí
          location.reload();
        },
        error: function(error) {
          console.error(error);
        }
      });
    }else{

    }
  });

function validateForm() {
    var isValid = true;
    $(".form-outline").each(function() {
      var input = $(this).find(".form-control");
      if (input.val() === "") {
        isValid = false;
        // Agrega el estilo de error o muestra un mensaje de error
        $(this).addClass("has-error");
      } else {
        // Si el campo es válido, elimina cualquier estilo de error
        $(this).removeClass("has-error");
      }
    });
    return isValid;
}
```

PHP (user.php)

En el script PHP user.php, se manejan las solicitudes POST del formulario de registro. Los datos del formulario se reciben, y se realiza una inserción en la base de datos MySQL. La contraseña se almacena de forma segura utilizando el algoritmo de hash. Se incluye un ejemplo básico de conexión a la base de datos y ejecución de una consulta INSERT.

```php
<?php

require_once('../../../model/conexion/conexion.php');
require_once('../../../model/querys/create/registro.php');
if (session_status() == PHP_SESSION_NONE) {
        session_start();
}

if ($_SERVER["REQUEST_METHOD"] == "POST") {

        if( isset($_POST['name']) && isset($_POST['username']) && isset($_POST['mail']) && isset($_POST['pass'])){

                if( !empty($_POST['name']) && !empty($_POST['username']) && !empty($_POST['mail']) && !empty($_POST['pass'])){

                        $name = trim($_POST['name']);
                        $username = trim($_POST['username']);
                        $mail = trim($_POST['mail']);
                        $pass = md5(trim($_POST['pass']));

                        $model = new Create();

                        $consulta = $model->inserUser($name, $username, $mail, $pass);

                        echo $consulta;

                }else{
                        echo "no se informaron los campos solicitados";
                }

        }else{
                echo "Envio incorrecto de información";
        }

}
```

```php
    public function inserUser($name, $username, $mail, $pass){

        $resultado=null;
        $model=new conexion();
        $consulta=$model->get_conexion();
        $lRet=False;

        $sql="INSERT INTO usuarios (name,CodigoUsuario,Email,Password)
        VALUES (:name,:CodigoUsuaio,:Email,:Password)";
        $result=$consulta->prepare($sql);
        $result->bindParam(':name',$name);
        $result->bindParam(':CodigoUsuaio',$username);
        $result->bindParam(':Email',$mail);
        $result->bindParam(':Password',$pass);

        if($result){

            $result->execute();

            $lRet = TRUE;

        }

        return $lRet;
    }
```

IBEROAMERICANA
CORPORACIÓN UNIVERSITARIA

Login de Usuarios

HTML (index.html)

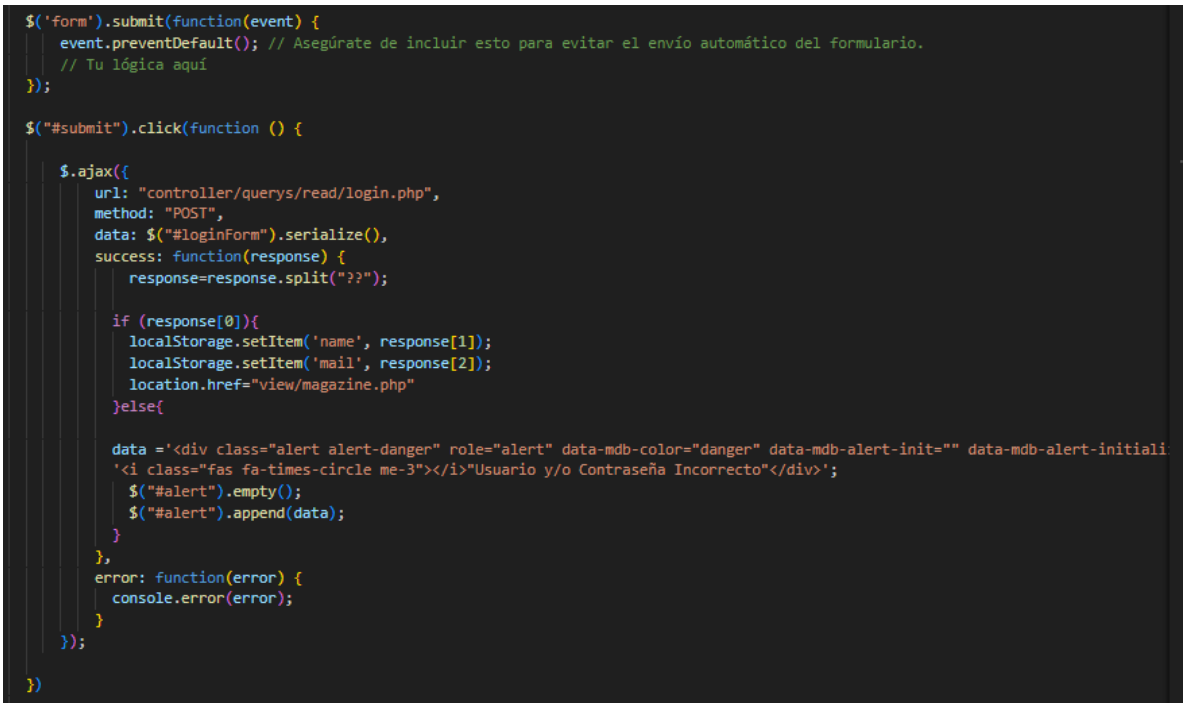
En el archivo HTML index.html, se define la estructura de la página de login. Este formulario simple recopila información esencial del usuario, usuario y contraseña. Además, se vinculan los archivos jQuery necesarios para el diseño y la funcionalidad del formulario.

```
<form id="loginForm">
  <div class="text-center mb-3">
    <p>Sign in with:</p>
  </div>

  <!-- Email input -->
  <div data-mdb-input-init class="form-outline mb-4">
    <input type="text" id="username" name="username" class="form-control" />
    <label class="form-label" for="username">Email o username</label>
  </div>

  <!-- Password input -->
  <div data-mdb-input-init class="form-outline mb-4">
    <input type="password" id="loginPassword" name="password" class="form-control" />
    <label class="form-label" for="loginPassword">Password</label>
  </div>

  <!-- Submit button -->
  <button type="submit" class="btn btn-primary btn-block mb-4" id="submit">
    Sign in
  </button>
  <div id="alert">

  </div>


  <!-- Register buttons -->
</form>
```

IBEROAMERICANA
CORPORACIÓN UNIVERSITARIA

jQuery (login.js)

En el archivo jQuery login.js, se utiliza la biblioteca jQuery para manejar la interactividad del formulario. La función principal se activa cuando el formulario se envía y evita que la página se recargue. Recopila los datos del formulario y los envía al servidor utilizando una solicitud AJAX, lo que permite la actualización de la página sin necesidad de recarga completa.



```
$('form').submit(function(event) {
    event.preventDefault(); // Asegúrate de incluir esto para evitar el envío automático del formulario.
    // Tu lógica aquí
});

$("#submit").click(function () {

    $.ajax({
        url: "controller/querys/read/login.php",
        method: "POST",
        data: $("#loginForm").serialize(),
        success: function(response) {
            response=response.split("??");

          if (response[0]){
            localStorage.setItem('name', response[1]);
            localStorage.setItem('mail', response[2]);
            location.href="view/magazine.php"
          }else{

          data ='<div class="alert alert-danger" role="alert" data-mdb-color="danger" data-mdb-alert-init="" data-mdb-alert-initiali
          '<i class="fas fa-times-circle me-3"></i>"Usuario y/o Contraseña Incorrecto"</div>';
            $("#alert").empty();
            $("#alert").append(data);
          }
        },
        error: function(error) {
          console.error(error);
        }
    });

})
```

PHP (user.php)

En el script PHP user.php, se manejan las solicitudes POST del formulario de login. Los datos del formulario se reciben, y se realiza una consulta en la base de datos MySQL. La contraseña se almacena de forma segura utilizando el algoritmo de hash. Se incluye un ejemplo básico de conexión a la base de datos y ejecución de una consulta SELECT.

```php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Verificar si se han enviado los datos de inicio de sesión
    if (isset($_POST['username']) && isset($_POST['password'])) {
        // Verificar si los campos no están vacíos
        if (!empty($_POST['username']) && !empty($_POST['password'])) {
            // Limpiar y obtener los valores del formulario
            $username = trim($_POST['username']);
            $password = md5(trim($_POST['password']));

            // Crear una instancia del modelo
            $model = new queriesGet();

            // Obtener información del usuario desde la base de datos
            $consulta = $model->showUser($username);

            if (!empty($consulta)) {
                foreach ($consulta as $row) {
                    // Verificar la contraseña usando password_verify en lugar de md5
                    if (password_verify($password, $row['Password'])) {
                        // Almacenar información en variables
                        $cName = $row['name'];
                        $cMail = $row['Email'];
                        $_SESSION['user'] = $row['CodigoUsuario'];

                        $lRet = true; // Credenciales válidas
                    }
                }
            }
        }
    }
}
```

```php
class queriesGet {
    1 reference | 0 overrides
    public function showUser($user) {
        $resultado = array();
        $modelo = new conexion();
        $conexion = $modelo->get_conexion();

        $sql = "SELECT * FROM usuarios WHERE CodigoUsuario = :user OR Email = :mail";

        $result = $conexion->prepare($sql);
        $result->bindParam(":user", $user);
        $result->bindParam(":mail", $user);

        $result->execute();

        while ($f = $result->fetch(PDO::FETCH_ASSOC)) {
            $resultado[] = $f;
        }

        return $resultado;
    }
}
```

Visuaización de contenido

HTML (magazine.php)

En el archivo HTML-PHP magazine.php, se define la estructura de la página contenido

```
    <!-- Navbar -->
    <nav class="navbar navbar-expand-lg navbar-light bg-white fixed-top">
      <div class="container-fluid">
        <!-- Navbar brand -->
        <a class="navbar-brand" id="user">
          <img src="../img/notices/user.png" height="40" alt="" loading="lazy"
               style="margin-top: -3px;"  />
        </a>
        <i class="fas fa-hands fa-2x logoutButton" ></i>

        <i class="fas fa-circle-plus fa-2x new" ></i>

          </ul>
        </div>
      </div>
    </nav>
    <!-- Navbar -->

  </header>
  <!--Main Navigation-->
  <br>
  <!--Main layout-->
  <main class="my-5">
    <div class="container">

      <!--Section: Content-->
      <section>
        <div class="row gx-lg-5" id="contenedorPaginas">


        </div>
      </section>
      <!--Section: Content-->


    </div>
  </main>
  <!--Main layout-->
```

IBEROAMERICANA
CORPORACIÓN UNIVERSITARIA

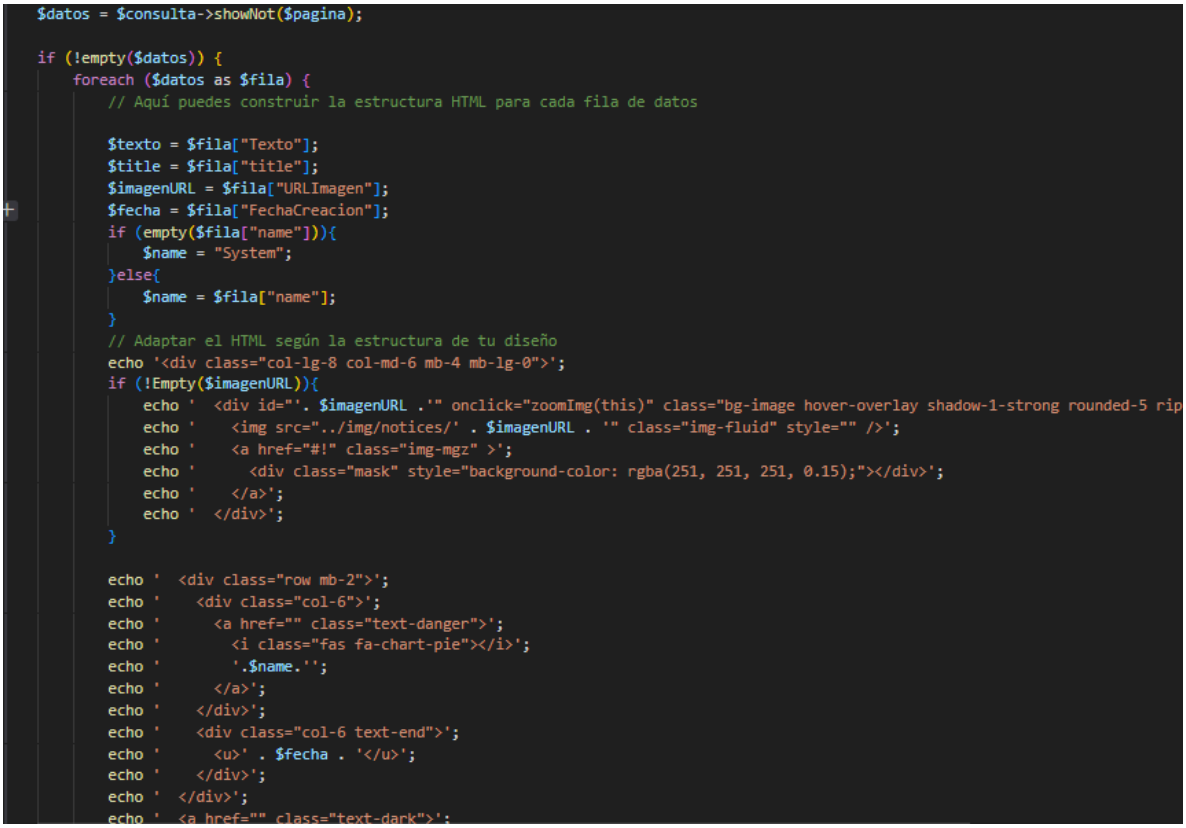
jQuery (magazine.js)

En el archivo jQuery magazine.js, se utiliza la biblioteca jQuery para manejar la recopilación de los datos de los datos extraídos del servidor utilizando una solicitud AJAX, lo que permite la actualización de la página sin necesidad de recarga completa.

```
let paginaActual = 1;

agregarPagina(paginaActual)

// Función para cargar más páginas al hacer scroll
Comment Code
function cargarMasPaginas() {
  const contenedor = $("#contenedorPaginas");
  const alturaContenedor = contenedor.height();
  const scrollTop = $(window).scrollTop();
  const alturaVentana = $(window).height();

  // Carga más contenido si el scroll está cerca del final del contenedor
  if (scrollTop + alturaVentana > alturaContenedor - 100) {
    paginaActual++;
    agregarPagina(paginaActual);
  }
}

// Función para agregar una página al contenedor
Comment Code
function agregarPagina(numeroPagina) {

  $.ajax({
    url: "../controller/querys/read/load.php", // Reemplaza con la URL de tu script PHP para obtener datos
    method: "GET",
    data: { pagina: numeroPagina },
    success: function (data) {
      // Insertar datos en el contenedor de páginas
      $("#contenedorPaginas").append(data);
    },
    error: function (error) {
      console.error("Error al obtener datos:", error);
    }
  });

}
// Agregar un listener para el evento de scroll
window.addEventListener("scroll", cargarMasPaginas);
```

PHP (load.php)

En el script PHP load.php, se manejan las solicitudes de información del contenido organizándolo en la estructura solicitada por el HTML



```php
$datos = $consulta->showNot($pagina);

if (!empty($datos)) {
    foreach ($datos as $fila) {
        // Aquí puedes construir la estructura HTML para cada fila de datos

        $texto = $fila["Texto"];
        $title = $fila["title"];
        $imagenURL = $fila["URLImagen"];
        $fecha = $fila["FechaCreacion"];
        if (empty($fila["name"])){
            $name = "System";
        }else{
            $name = $fila["name"];
        }
        // Adaptar el HTML según la estructura de tu diseño
        echo '<div class="col-lg-8 col-md-6 mb-4 mb-lg-0">';
        if (!Empty($imagenURL)){
            echo '  <div id="'. $imagenURL .'" onclick="zoomImg(this)" class="bg-image hover-overlay shadow-1-strong rounded-5 rip
            echo '    <img src="../img/notices/' . $imagenURL . '" class="img-fluid" style="" />';
            echo '    <a href="#!" class="img-mgz" >';
            echo '      <div class="mask" style="background-color: rgba(251, 251, 251, 0.15);"></div>';
            echo '    </a>';
            echo '  </div>';
        }

        echo '  <div class="row mb-2">';
        echo '    <div class="col-6">';
        echo '      <a href="" class="text-danger">';
        echo '        <i class="fas fa-chart-pie"></i>';
        echo '        '.$name.'';
        echo '      </a>';
        echo '    </div>';
        echo '    <div class="col-6 text-end">';
        echo '      <u>' . $fecha . '</u>';
        echo '    </div>';
        echo '  </div>';
        echo '  <a href="" class="text-dark">';
```

```php
class queriesNot {
    1 reference | 0 overrides
    public function showNot($pagina, $porPagina = 10) {
        $resultado = null;
        $modelo = new conexion();
        $conexion = $modelo->get_conexion();

        $inicio = ($pagina - 1) * $porPagina;

        $sql = "SELECT imagenes.*, usuarios.name
                FROM imagenes
                LEFT JOIN usuarios ON imagenes.UsuarioCreador = usuarios.CodigoUsuario
                ORDER BY imagenes.FechaCreacion DESC
                LIMIT $inicio, $porPagina";

        $result = $conexion->prepare($sql);
        $result->execute();

        while ($f = $result->fetch()) {
            $resultado[] = $f;
        }

        return $resultado;
    }
}
```

IBEROAMERICANA
CORPORACIÓN UNIVERSITARIA

Creación de contenido

HTML (newNot.php)

En el archivo HTML-PHP newNot.php, Este formulario simple recopila información esencial Titulo, Descripción y adjuntos JPG. Además, se vinculan los archivos jQuery necesarios para el diseño y la funcionalidad del formulario.

```
<section class="g-start-2" style="grid-row: 2">
  <div class="row gx-lg-5" style="justify-content:center; margin-top: 25%;">

    <form  style="width: 26rem;" id="formNot" enctype="multipart/form-data">

      <div data-mdb-input-init class="form-outline mb-4">
        <input type="text" class="form-control" id="title"  name="title"></textarea>
        <label class="form-label" for="title">Titulo</label>
      </div>
      <!-- Message input -->
      <div data-mdb-input-init class="form-outline mb-4">
        <textarea class="form-control" id="txt" rows="4" name="txt"></textarea>
        <label class="form-label" for="txt">Message</label>
      </div>

      <label class="form-label" for="customFile">Default file input example</label>
      <input type="file" class="form-control" id="customFile" name="imagen" accept=".jpg"/>


      <!-- Submit button -->
      <button data-mdb-ripple-init type="submit" class="btn btn-primary btn-block mb-4" style="margin-top:10%;">Send</button>
    </form>

  </div>
</section>
```

IBEROAMERICANA
CORPORACIÓN UNIVERSITARIA

jQuery (addNot.js)

En el archivo jQuery addNot.js, se utiliza la biblioteca jQuery para manejar la recopilación de los datos de los datos extraídos del servidor utilizando una solicitud AJAX, lo que permite la actualización de la base de datos.

```
$("#formNot").submit(function (e) {
  e.preventDefault();

  var formData = new FormData(this);

  if (
    !formData.get("txt").trim() == "" ||
    !formData.get("imagen").name.trim() == ""
  ) {
    $.ajax({
      url: "../../controller/querys/create/notice.php",
      type: "POST",
      dataType: "html",
      data: formData,
      cache: false,
      contentType: false,
      processData: false,
      success: function (response) {
        if (response) {
          let timerInterval;
          Swal.fire({
            title: "Actualización",
            html: "Realizada con exito",
            timer: 4000,
            timerProgressBar: true,
            didOpen: () => {
              Swal.showLoading();
              const timer = Swal.getPopup().querySelector("b");
            },
            willClose: () => {
              clearInterval(timerInterval);
            },
          }).then((result) => {
            /* Read more about handling dismissals below */
            if (result.dismiss === Swal.DismissReason.timer) {
              location.href = "../magazine.php";
            }
          });
        }
```

IBEROAMERICANA
CORPORACIÓN UNIVERSITARIA

PHP (notice.php)

En el script PHP notice.php, se manejan las solicitudes POST del formulario de registro. Los datos del formulario se reciben, y se realiza una inserción en la base de datos MySQL. Se incluye un ejemplo básico de conexión a la base de datos y ejecución de una consulta INSERT.

```php
$model = new Create();

$url = "";
$title = $_POST['title'];
$txt = $_POST['txt'];
$user = $_SESSION['user'];

// Verificar si se cargó correctamente la imagen
if (isset($_FILES["imagen"]) && $_FILES["imagen"]["error"] == 0) {
        $carpeta_destino = "../../../img/notices/";
        // Obtener el nombre original de la imagen
        $nombre_original = $_FILES["imagen"]["name"];
        // Generar un nombre aleatorio de 10 dígitos
        $nombre_aleatorio = mt_rand(1000000000, 9999999999);
        // Crear el nuevo nombre de la imagen
        $nuevo_nombre = $nombre_aleatorio;
        $nuevo_nombre = pathinfo($nuevo_nombre, PATHINFO_FILENAME) . '.jpg';
        // Ruta completa donde se guardará la imagen
        $ruta_completa = $carpeta_destino . $nuevo_nombre;
        // Subir la imagen al servidor
        move_uploaded_file($_FILES["imagen"]["tmp_name"], $ruta_completa);
        // Ejemplo de cambiar tamaño con GD
        list($ancho, $alto) = getimagesize($ruta_completa);
        $nuevo_ancho = 2000; // Nuevo ancho en píxeles
        $nuevo_alto = 2000; // Nuevo alto en píxeles

        $imagen_redimensionada = imagecreatetruecolor($nuevo_ancho, $nuevo_alto);
        $imagen_original = imagecreatefromjpeg($ruta_completa);

        imagecopyresized($imagen_redimensionada, $imagen_original, 0, 0, 0, 0, $nuevo_ancho, $nuevo_alto, $ancho, $alto);
        // Guardar la imagen redimensionada
        imagejpeg($imagen_redimensionada, $carpeta_destino . $nuevo_nombre);
        // Liberar memoria
        imagedestroy($imagen_original);
        imagedestroy($imagen_redimensionada);
        $url = $nuevo_nombre;
}

$consulta = $model->inserMagazine($url, $title, $txt, $user);
```

IBEROAMERICANA
CORPORACIÓN UNIVERSITARIA

```php
class Create{
    1 reference | 0 overrides
    public function inserMagazine($url,$title,$txt,$user){

        $resultado=null;
        $model=new conexion();
        $consulta=$model->get_conexion();
        $lRet=False;

        $sql="INSERT INTO imagenes (URLImagen,title,Texto,UsuarioCreador)
        VALUES (:url,:title,:txt,:user)";
        $result=$consulta->prepare($sql);
        $result->bindParam(':url',$url);
        $result->bindParam(':title',$title);
        $result->bindParam(':txt',$txt);
        $result->bindParam(':user',$user);

        if($result){

            $result->execute();

            $lRet = TRUE;

        }

        return $lRet;
    }
```