

Documentación del CRUD de Alojamientos: Gestión de Usuarios y Administrador en PHP y MYSQL

Descripción general

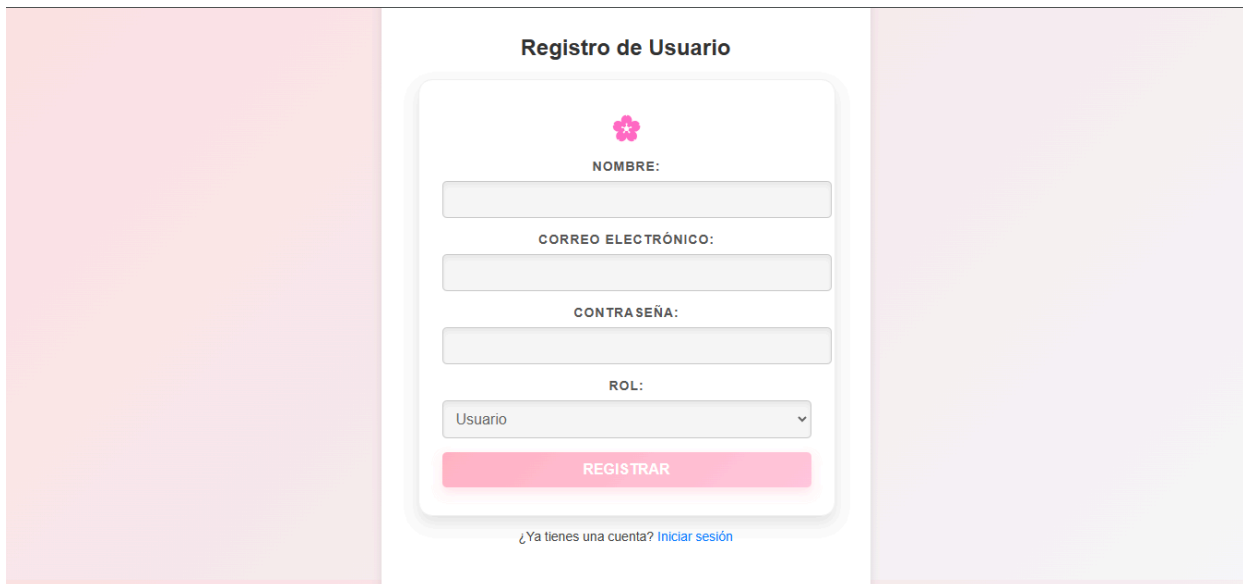
Este proyecto consiste en una aplicación web que permite la gestión de alojamientos a usuarios a través de operaciones CRUD en la Creación de usuarios y la Eliminación de estos utilizando como una base de datos MySQL con el servicio de XAMPP y como back-end de desarrollo con PHP para la gestión de usuarios y alojamientos, así también para manejar la autenticación e interacción con la base de datos.

Desarrollo


Los usuarios pueden registrarse mediante un formulario que recoge sus datos, como nombre, correo electrónico, contraseña y rol (usuario o administrativo), y una vez registrados, pueden iniciar sesión para acceder a su cuenta. Al ingresar, se les redirige a una página donde pueden seleccionar alojamientos disponibles que se encuentran precargados desde la base de datos, los administradores tienen permisos especiales, ya que pueden eliminar alojamientos de la base de datos, pero no pueden añadir los alojamientos de los usuarios. Además, los administradores tienen acceso para visualizar todos los usuarios registrados y eliminarlos si es necesario, lo que les otorga control total sobre la gestión de la aplicación. En la parte de la estructuración de los archivos, el proyecto se organiza en varias páginas PHP, como index.php para el registro de usuarios, registro.php para validar y almacenar los datos en la base de datos, validar_sesion.php para la autenticación de usuarios en el inicio de sesión, y cuenta_usuario.php, donde los administradores pueden ver y gestionar los alojamientos asociados a la cuenta de los usuarios. La base de datos, llamada mi_aplicacion, contiene las tablas usuarios (para almacenar los datos de los usuarios), alojamientos (para la información de los alojamientos) y usuario_alojamientos (para relacionar a los usuarios con los alojamientos seleccionados). El proyecto también implementa prácticas de seguridad como el uso de password_hash()

y `password_verify()` para asegurar las contraseñas de los usuarios, y la autenticación se gestiona mediante sesiones PHP. La aplicación está pensada para ejecutarse en un servidor PHP con acceso a MySQL, proporcionando una interfaz sencilla y amigable que permite a los usuarios realizar las operaciones necesarias sin complicaciones. El repositorio en GitHub alberga todo el código fuente del proyecto, lo que permite su revisión, modificación y ampliación según sea necesario.

Diseño del proyecto



Registro de Usuario



NOMBRE:

CORREO ELECTRÓNICO:

CONTRASEÑA:

ROL:

Usuario

REGISTRAR

[¿Ya tienes una cuenta? Iniciar sesión](#)

Bienvenido, Michael!

Rol: administrativo

Administración de Usuarios

ID	Nombre	Correo Electrónico	Rol	Acciones
26	Michael Dx	admin12@example.com	usuario	<button>Eliminar</button>

[Regresar al registro](#)



Inicio de Sesión

CORREO ELECTRÓNICO:

CONTRASEÑA:

INICIAR SESIÓN

¿No tienes una cuenta? [Regístrate aquí](#)

Bienvenido, 123!

Rol: usuario

No tienes acceso a esta sección.

[Regresar al registro](#)

Estructuración de la base de datos en MYSQL

The screenshot displays the MySQL Workbench interface. On the left, the 'Schemas' pane shows a tree view with 'mi_aplicacion' expanded, revealing tables: 'alojamiento', 'alojamientos', 'usuarios', 'usuarios_alojamientos', and 'usuario_alojamientos'. The main area shows the 'Structure' tab for the 'mi_aplicacion' database. It lists 5 tables with their respective actions (Browse, Structure, Search, Insert, Empty, Drop) and details like engine (InnoDB), collation (utf8mb4_general_ci), size, and overhead. A 'Sum' row indicates the total size of the tables is 160.0 KiB. Below the table list, there is a 'Create new table' dialog box with fields for 'Table name' and 'Number of columns' (set to 4), and a 'Create' button. The bottom of the interface shows a 'Console' tab.

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> alojamiento	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> alojamientos	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> usuarios	★ Browse Structure Search Insert Empty Drop	4	InnoDB	utf8mb4_general_ci	32.0 KiB	-
<input type="checkbox"/> usuarios_alojamientos	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	48.0 KiB	-
<input type="checkbox"/> usuario_alojamientos	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	48.0 KiB	-
5 tables	Sum	4	InnoDB	utf8mb4_general_ci	160.0 KiB	0 B

Usuarios

The screenshot shows a MySQL database management interface. On the left, a tree view displays the database structure, including schemas like 'information_schema', 'mi_aplicacion', and 'mysql'. The main area shows a SQL query: `SELECT * FROM `usuarios``. Below the query, there are options for 'Show all', 'Number of rows' (set to 25), 'Filter rows' (Search this table), and 'Sort by key' (None). The query results are displayed in a table with columns: id, nombre, correo, contrasena, and rol. The table contains three rows of data. Below the table, there are options for 'Check all', 'With selected', 'Edit', 'Copy', 'Delete', and 'Export'. At the bottom, there are options for 'Query results operations' including 'Print', 'Copy to clipboard', 'Export', 'Display chart', and 'Create view'.

	id	nombre	correo	contrasena	rol
<input type="checkbox"/>	14	Admin	admin@example.com	adminpassword	administrativo
<input type="checkbox"/>	26	Michael Dx	admin12@example.com	\$2y\$10\$z9rvx1EtleXb1/V0ff94neTOYNFinAcMRXqqvFyeS4t...	usuario
<input type="checkbox"/>	34	123	admin122@example.com	\$2y\$10\$OtTe8R5a6RGCGJCCerbkN UkDkmgPGe4/7fx0ZjWYwP...	usuario

Ejemplos de implementación de `password_hash()` y `password_verify()` para una mayor seguridad en el uso de los datos dentro del proyecto

```
if ($_SERVER['REQUEST_METHOD'] === 'POST') {  
    $nombre = htmlspecialchars($_POST['nombre']);  
    $correo = htmlspecialchars($_POST['correo']);  
    $contrasena = password_hash($_POST['contrasena'], PASSWORD_BCRYPT);  
    $rol = htmlspecialchars($_POST['rol']);  
}
```

```
<?php
include 'db.php';

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $correo = $_POST['correo'];
    $contrasena = $_POST['contrasena'];

    // Verificar usuario y contraseña
    $sql = "SELECT * FROM usuarios WHERE correo = ?";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("s", $correo);
    $stmt->execute();
    $result = $stmt->get_result();
    $usuario = $result->fetch_assoc();

    if ($usuario && password_verify($contrasena, $usuario['contrasena'])) {
        session_start();
        $_SESSION['usuario_id'] = $usuario['id'];
        $_SESSION['nombre'] = $usuario['nombre'];
        header("Location: cuenta_usuario.php");
    } else {
        echo "Credenciales incorrectas.";
    }

    $stmt->close();
    $conn->close();
}
?>
```