Actividad práctica. RBCA (Role Based Access Control). Redirección

Descripción de la actividad

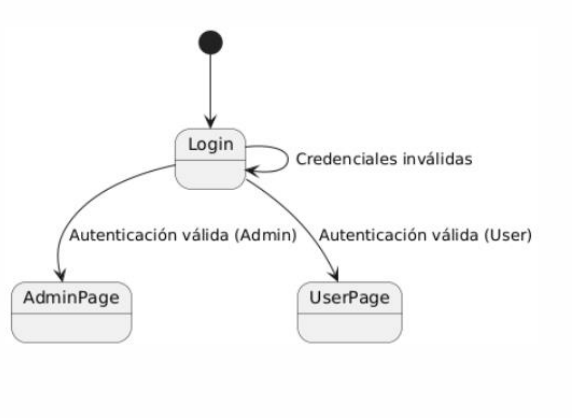
En esta página vamos a trabajar con la seguridad de acceso **basada en roles (RBCA) y la redirección entre módulos.** Para ello vamos a plantear una aplicación web con los siguientes módulos php

- index.html: Página de entrada a la aplicación
- login.php: Módulo de autenticación.
- admin.php: Página accesible solo para el administrador.
- user.php: Página para usuarios normales.

Inicialmente el usuario hará login. Si el usuario es administrador se le redirigirá hacia la página de administración (admin.php), sino, hacia el módulo user.php

Tendremos **un array asociativo simulando los usuarios registrados en el sistema**

Lo anterior puede representarse en un diagrama de transición de estados como el siguiente:



NOTA: Este diagrama muestra la transición entre páginas y lo tendrás que utilizar en tú proyecto futuro de la asignatura.

Role-Based Access Control (RBAC)

El **Role-Based Access Control (RBAC)**, o Control de Acceso Basado en Roles, es una **técnica de gestión de permisos** que organiza los privilegios de acceso de los usuarios en función de sus roles dentro de un sistema. En lugar de asignar permisos directamente a cada usuario, **los permisos se agrupan y se asignan a roles predefinidos**. Los usuarios se asignan a uno o más roles, y adquieren los

permisos correspondientes a esos roles. Esta técnica es ampliamente utilizada en aplicaciones web, ya que facilita la administración de seguridad y control de acceso, especialmente en sistemas con muchos usuarios.

Componentes principales de RBAC:

1. **Roles**: Representan un conjunto de permisos relacionados con una función o responsabilidad dentro de una organización o sistema. Ejemplos comunes son "Administrador", "Usuario", "Editor", "Moderador", etc.
2. **Permisos**: Son los derechos o privilegios asociados a las acciones que se pueden realizar en el sistema, como leer, escribir, eliminar o modificar datos. Un rol puede tener múltiples permisos.
3. **Usuarios**: Son las personas o entidades que interactúan con el sistema. A cada usuario se le asigna uno o más roles, lo que determina qué puede y no puede hacer dentro de la aplicación.
4. **Objetos de recursos**: Son los elementos del sistema a los que se quiere controlar el acceso, como archivos, bases de datos, páginas web o funciones.

Control de acceso en el código:

- El sistema verificará los roles de los usuarios en función de sus credenciales.
- **En cada página o recurso protegido, se realiza una verificación de permisos antes de permitir el acceso.**

NOTA :Muchos frameworks y plataformas web ya integran la funcionalidad de RBAC de manera eficiente. Más adelante veremos como se implementa esto en Symphony

Conceptos clave de RBAC en una SPA:

1. **Roles y permisos**: **El backend sigue definiendo los roles y permisos, pero en el frontend, se utilizan para mostrar/ocultar ciertas partes** de la interfaz de usuario (UI) o restringir el acceso a rutas específicas.
2. **Autenticación y autorización**: La autenticación (comprobación de identidad) y la autorización (permisos basados en roles) se realizan al inicio de la sesión. Los datos del usuario autenticado (incluyendo roles)

pueden ser almacenados en un token JWT (JSON Web Token es un estándar qué está dentro del documento RFC 7519). o en otro formato, y este token se usa en el frontend para aplicar RBAC.

Ejemplo de Token JWT

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6I
kpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.ikFGEvw-Du0f30vBaA742D_wqPA5BBHXgUY6
wwqab1w
```

A partir de ahí la aplicación cliente, con ese token, haría peticiones solicitando recursos, siempre con ese token JWT dentro de un encabezado, que sería Authorization: Bearer XXXXXXX, siendo Bearer el tipo de prefijo seguido de todo el contenido del token.

3. **Protección de rutas**: En las SPA, las rutas son gestionadas por el router del framework (Angular Router, React Router), y las rutas protegidas solo deben ser accesibles por usuarios con los roles adecuados.
4. **Validación del lado del servidor**: Aunque las rutas y componentes se pueden proteger en el frontend, siempre debe haber validación adicional en el backend para garantizar la seguridad total.

Redirección entre páginas

La **redirección entre páginas** en módulos PHP se realiza utilizando la función header(), **que permite enviar encabezados HTTP al navegador para redirigir al usuario a otra página.** Esta es una de las maneras más comunes de gestionar el flujo entre páginas en aplicaciones web.

Uso básico de header() para redirección:

Para redirigir a otra página, se utiliza la función header() seguida del encabezado Location, que especifica la URL de destino.

```
<?php

// Redirigir a la página de inicio

header('Location: inicio.php');

exit;  // Se usa exit para asegurarse de que el script termine después de la redirección

?>
```

Reglas de uso de header

- **No debe haber salida previa al header():**l encabezado HTTP solo se puede enviar si no ha habido ninguna salida antes, es decir, no debe haber ningún contenido HTML, espacios en blanco, o cualquier otra salida en el archivo PHP antes de la llamada a header().
- **exit o die después de la redirección:** Se recomienda utilizar exit o die justo después de la llamada a header()

Desarrollo de la actividad práctica

Módulo roles.php

Este módulo simplemente simula la base de datos de usuarios. Se cargará desde ortos módulos para su acceso

```
<?php
// Simulación de una base de datos con usuarios y roles
$usuarios = [
    "admin" => ["password" => "admin123", "rol" => "admin"],
    "usuario1" => ["password" => "user123", "rol" => "user"],
    "usuario2" => ["password" => "user456", "rol" => "user"]
];
?>
```

NOTA: Modifica este código para que las contraseñas no se guarden en claro

Módulo de Inicial (index.php)

Esta es la página de entrada a la aplicación. Simplemente presenta una interfaz con un formulario para hacer la solicitud de login al módulo logín.php

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Login</title>
</head>
<body>
    <h2>Iniciar sesión</h2>
    <form method="POST" action=¿----?>
        <label for="username">Usuario:</label>
        <input type="text" id="username" name="username" required><br>

        <label for="password">Contraseña:</label>
        <input type="password" id="password" name="password"
required><br>
```

```
        <button type="submit">Ingresar</button>
    </form>
</body>
</html>
```

NOTA: Modifica el código para que solicite el procesamiento de las credenciales al módulo de login.php

Módulo de Login (login.php)

Este módulo gestiona la autenticación de los usuarios y redirige según su rol. Fíjate que incluye el módulo php roles.php

```
<?php
session_start();
include 'roles.php';

//1.- Obtener credenciales
//2.- Verificación de las credenciales

//3.- Redirigir según el rol
//Si es administrador
            header('Location: admin.php');
//si es usuario normal
            header('Location: user.php');
//si no es ninguno de los dos
        echo "Usuario o contraseña incorrectos.";
?>
```

NOTA: Completa cada uno de los puntos del esquema de código

Módulo admin.php

Este es sencillamente el módulo admin.php que tiene que cargarse si el usuario que ha accedido es un administrador.

NOTA: Incluye este módulo tal cual en tú proyecto

```
<?php


<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Administrador</title>
</head>
<body>
    <h2>Bienvenido, Administrador</h2>
    <p>Esta es la página exclusiva para administradores.</p>
</body>
</html>
```

Página de Usuario Normal (user.php)

Esta página es accesible solo para los usuarios con rol de "user".

```
<?php


<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Usuario</title>
</head>
<body>
    <h2>Bienvenido</h2>
    <p>Esta es la página para usuarios normales.</p>


</body>
</html>
```

Implementa los módulos anteriores y contesta a las siguientes preguntas:

a) Implementa la práctica según las indicaciones que se proporcionan
b) Prueba la práctica comprendiendo su funcionamiento
c) Responde a las siguientes preguntas
    a. ¿ Que supone añadir un nuevo ROL?
    b. Puede tener un usuario con este sistema varios Roles. ¿ Qué implicaciones tiene en el diseño?
d) ¿Qué dimensión de la seguridad protege este esquema? : Autenticidad, Confidencialidad, Integridad, Disponibilidad, Trazabilidad
e) Añade el rol manager al esquema de seguridad y haz que algún usuario pueda acceder con este ROL. ¿ Qué partes has tenido que modificar?