Proyecto: Sistema Web de Gestión de Reservas de Salones para Eventos

Índice

1. Introducción

1.1. Objetivo del Proyecto

Desarrollar un sistema web para la gestión de reservas de salones de eventos, permitiendo a los usuarios consultar, reservar y presupuestar servicios, y a los administradores gestionar la información de salones, servicios, presupuestos y usuarios.

1.2. Descripción General

La aplicación permite a los usuarios autenticarse mediante la API pública de DummyJSON, visualizar salones disponibles, seleccionar servicios adicionales y generar presupuestos. El panel de administración está protegido y permite la gestión completa de las entidades del sistema. El sistema también genera comprobantes PDF con QR y logo.

2. Requerimientos Funcionales

2.1. Inicio de Sesión con DummyJSON

Se implementó un formulario de login que utiliza la API pública https://dummyjson.com/auth/login. Los usuarios pueden autenticarse con credenciales válidas obtenidas de https://dummyjson.com/users.

2.2. Almacenamiento del accessToken

El accessToken recibido tras el login exitoso se guarda en sessionStorage, permitiendo identificar al usuario durante su sesión y proteger rutas del sistema.

2.3. Protección de Funcionalidades de Administración

El acceso al panel y sus funcionalidades está restringido a usuarios autenticados con rol de administrador. Se verifica la existencia del token y el rol antes de permitir el acceso.

2.4. Listado de Usuarios DummyJSON en el Panel

Se incluyó una sección en el panel que muestra usuarios registrados en DummyJSON, mostrando nombre, email, teléfono y ciudad.

2.5. ABM de Entidades

El sistema permite crear, modificar y eliminar salones, servicios y presupuestos desde el panel de administración.

3. Desarrollo e Implementación

3.1. Estructura de Archivos

Archivos HTML

- index.html: Página principal. Muestra banner, cards de salones, mapa, login modal, detalles y reservas.

- cpanel.html: Panel de administración. ABM(altas, bajas, modificaciones) de salones, servicios, usuarios y mensajes.

- contacto.html: Formulario de contacto y mapa.

- institucional.html: Información institucional.

- login.html: Formulario de inicio de sesión.

Archivos CSS

- style.css: Estilo general. Personalización de Bootstrap, responsividad, paleta de colores, etc.

Archivos JavaScript

Inicialización de datos

- ini_salones.js, ini_servicios.js, ini_presupuestos.js

Gestión de salones

- render_salones.js, crud_salones.js

Gestión de servicios

- servicios.js

Gestión de reservas y comprobantes

- reserva.js

Gestión de usuarios y autenticación

- login.js, logout.js, auth.js, proteger_ruta.js, usuarios.js

Gestión de mensajes de contacto

- contacto.js, mensajes_contacto.js

3.2. Implementación del Login

Al autenticarse, el sistema guarda el token y el rol. El usuario “michaelw” es considerado administrador y accede al panel.

3.3. Protección de Rutas y Roles

Se creó un script que protege rutas validando si el usuario tiene token y el rol correspondiente antes de acceder a páginas protegidas.

3.4. Gestión de Salones

Desde el panel se puede realizar ABM de salones, incluyendo nombre, descripción, dirección, capacidad, precio, estado e imagen.

3.5. Gestión de Servicios

El administrador puede gestionar servicios adicionales, cargarlos, editarlos y eliminarlos desde el panel.

3.6. Gestión de Presupuestos

Los usuarios pueden seleccionar salón y servicios, y generar un comprobante PDF con resumen, logo y QR.

3.7. Listado de Usuarios DummyJSON

En el panel se visualiza una tabla con usuarios de DummyJSON. Se muestra información no sensible: nombre, correo, teléfono y ciudad.

4. Conclusiones

El sistema desarrollado cumple con los objetivos planteados: autenticación segura, protección de rutas, gestión completa de entidades, generación de comprobantes con QR y visualización de usuarios externos. Su estructura modular facilita futuras mejoras y mantenimiento.

5. Anexos

- Fragmentos de código relevantes (protección de rutas, listado de usuarios, etc.).

Ejemplo de protección de ruta:

```
export function protegerRuta(rolRequerido = null) {
  if (!sessionStorage.getItem('accessToken')) {
    window.location.href = "login.html";
    return;
  }
  if (rolRequerido && sessionStorage.getItem('rol') !== rolRequerido) {
    alert("No tienes permisos para acceder a esta sección.");
    window.location.href = "index.html";
  }
}
```

Ejemplo de listado de usuarios DummyJSON:

```
document.addEventListener('DOMContentLoaded', async () => {
  const res = await fetch('https://dummyjson.com/users');
  const data = await res.json();
  const tbody = document.querySelector('#tablaUsuarios tbody');
  tbody.innerHTML = data.users.map(u => `
    <tr>
      <td>${u.firstName} ${u.lastName}</td>
      <td>${u.email}</td>
      <td>${u.phone}</td>
      <td>${u.address.city}</td>
    </tr>
  `).join('');
});
```

Ejemplo de sección HTML para usuarios:

```html
<section class="my-5" id="seccion-usuarios">
  <h2 class="mb-4 text-center">Usuarios registrados</h2>
  <table class="table table-bordered" id="tablaUsuarios">
    <thead>
      <tr>
        <th>Nombre</th>
        <th>Email</th>
        <th>Teléfono</th>
        <th>Ciudad</th>
      </tr>
    </thead>
    <tbody>
      <!-- Acá se insertan los usuarios -->
    </tbody>
  </table>
```