Departamento de Computación
Programación Integrativa de Componentes

Nombre: Carlos Pogo
NRC: 16496

Tarea 1.2: Aplicaciones web js con POO

Tema:

Este proyecto es una aplicación web para la gestión de reservas y mesas en un restaurante. Utiliza HTML, CSS, JavaScript y Bootstrap para crear una interfaz de usuario interactiva y atractiva.

Estructura:

Tarea1.2
- css
  - styles.css
- html
  - mesas.html
  - reserva.html
- img
-js
  - mesas.js
  - reservas.js
index.html

Código:

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <title>Gestión de Reservas - Restaurante</title>
    <link
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"
rel="stylesheet">
    <link rel="stylesheet" href="css/styles.css">
</head>
<body>
    <div class="container">
        <h1 class="mt-5">Gestión de Reservas - Restaurante</h1>
        <nav class="nav nav-pills nav-justified mt-3">
            <a class="nav-item nav-link active"
href="htmls/reservas.html">Reservas</a>    
            <a class="nav-item nav-link active" href="htmls/mesas.html">Mesas</a>
        </nav>
    </div>
    <div class="container">
        <img src="img/restaurante.jpg" alt="" width="760px">
    </div>
    <script src="js/script.js"></script>
</body>
</html>
```

styles.css

```
body {
    padding-top: 20px;
    background-color: #99bbdd;
}

.container {
    max-width: 800px;
    margin: 0 auto;
    background-color: #ffffff;
    border-radius: 10px;
    box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1);
    padding: 20px;
    margin-top: 50px;
}

#formulario-reserva, #formulario-mesa {
    background: #f9f9f9;
    padding: 20px;
    border-radius: 5px;
    box-shadow: 0 0 10px rgba(0,0,0,0.1);
}
```

```css
.table {
    margin-top: 20px;
}



h1 {
    color: #343a40;
    text-align: center;
}
.nav-pills .nav-link {
    color: #007bff;
    font-weight: bold;
}
.nav-pills .nav-link.active {
    background-color: #007bff;
    color: #ffffff;
}
```

mesas.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Mesas</title>
    <link
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body>
    <div class="container">
        <h2 class="mt-5">Mesas</h2>
        <a href="../index.html"><button class="btn btn-warning my-
3">Regresar</button></a>
        <button class="btn btn-primary my-3" id="addMesaBtn">Crear Mesa</button>
        <table class="table">
            <thead>
                <tr>
                    <th>ID</th>
                    <th>Número</th>
                    <th>Capacidad</th>
                    <th>Ubicación</th>
```

```
                    <th>Acciones</th>
                </tr>
            </thead>
            <tbody id="mesas-list">
                <!-- Lista de mesas -->
            </tbody>
        </table>
    </div>
    <div id="formulario-mesa" class="container mt-5" style="display:none;">
        <h3 id="form-titulo-mesa">Crear Mesa</h3>
        <form id="form-mesa">
            <input type="hidden" id="mesa-id">
            <div class="form-group">
                <label for="mesa-numero">Número</label>
                <input type="number" class="form-control" id="mesa-numero"
required>
            </div>
            <div class="form-group">
                <label for="mesa-capacidad">Capacidad</label>
                <input type="number" class="form-control" id="mesa-capacidad"
required>
            </div>
            <div class="form-group">
                <label for="mesa-ubicacion">Ubicación</label>
                <input type="text" class="form-control" id="mesa-ubicacion"
required>
            </div>
            <button type="submit" class="btn btn-success">Guardar</button>
            <button type="button" class="btn btn-secondary"
id="cancelarMesaBtn">Cancelar</button>
        </form>
    </div>
    <script src="../js/mesas.js"></script>
</body>
</html>
```

reservas.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Reservas</title>
```

```
    <link
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body>
    <div class="container">
        <h2 class="mt-5">Reservas</h2>
        <a href="../index.html"><button class="btn btn-warning my-
3">Regresar</button></a>
        <button class="btn btn-primary my-3" id="addReservaBtn">Crear
Reserva</button>
        <table class="table">
            <thead>
                <tr>
                    <th>ID</th>
                    <th>Fecha</th>
                    <th>Hora</th>
                    <th>Número de Personas</th>
                    <th>Estado</th>
                    <th>Acciones</th>
                </tr>
            </thead>
            <tbody id="reservas-list">
                <!-- Lista de reservas -->
            </tbody>
        </table>
    </div>
    <div id="formulario-reserva" class="container mt-5" style="display:none;">
        <h3 id="form-titulo-reserva">Crear Reserva</h3>
        <form id="form-reserva">
            <input type="hidden" id="reserva-id">
            <div class="form-group">
                <label for="reserva-fecha">Fecha</label>
                <input type="date" class="form-control" id="reserva-fecha"
required>
            </div>
            <div class="form-group">
                <label for="reserva-hora">Hora</label>
                <input type="time" class="form-control" id="reserva-hora"
required>
            </div>
            <div class="form-group">
                <label for="reserva-numeroPersonas">Número de Personas</label>
                <input type="number" class="form-control" id="reserva-
numeroPersonas" required>
```

```
            </div>
            <div class="form-group">
                <label for="reserva-estado">Estado</label>
                <select class="form-control" id="reserva-estado" required>
                    <option value="pendiente">Pendiente</option>
                    <option value="confirmada">Confirmada</option>
                    <option value="cancelada">Cancelada</option>
                </select>
            </div>
            <button type="submit" class="btn btn-success">Guardar</button>
            <button type="button" class="btn btn-secondary"
id="cancelarReservaBtn">Cancelar</button>
        </form>
    </div>
    <script src="../js/reservas.js"></script>
</body>
</html>
```

mesas.js

```
class Mesa {
    // Constructor para crear una nueva mesa
    constructor(id, numero, capacidad, ubicacion) {
        this.id = id;
        this.numero = numero;
        this.capacidad = capacidad;
        this.ubicacion = ubicacion;
    }
}

class MesaManager {
    // Constructor para inicializar el administrador de mesas
    constructor() {
        // Cargar mesas desde el localStorage o inicializar una lista vacía
        this.mesas = JSON.parse(localStorage.getItem('mesas')) || [];

        // Obtener referencias a los elementos del DOM
        this.mesasList = document.getElementById('mesas-list');
        this.formMesa = document.getElementById('form-mesa');
        this.addMesaBtn = document.getElementById('addMesaBtn');
        this.cancelarMesaBtn = document.getElementById('cancelarMesaBtn');
        this.formularioMesa = document.getElementById('formulario-mesa');

        // Asignar eventos a los botones
```

```
        this.addMesaBtn.addEventListener('click', () => this.showForm());
        this.cancelarMesaBtn.addEventListener('click', () => this.hideForm());
        this.formMesa.addEventListener('submit', (event) => this.addMesa(event));

        // Renderizar las mesas al iniciar
        this.renderMesas();
    }

    // Método para renderizar la lista de mesas en la tabla
    renderMesas() {
        this.mesasList.innerHTML = ''; // Limpiar la lista actual
        this.mesas.forEach((mesa, index) => {
            const row = document.createElement('tr'); // Crear una nueva fila
para cada mesa
            row.innerHTML = `
                <td>${mesa.id}</td>
                <td>${mesa.numero}</td>
                <td>${mesa.capacidad}</td>
                <td>${mesa.ubicacion}</td>
                <td>
                    <button class="btn btn-warning"
onclick="mesaManager.editMesa(${index})">Editar</button>
                    <button class="btn btn-danger"
onclick="mesaManager.deleteMesa(${index})">Eliminar</button>
                </td>
            `;
            this.mesasList.appendChild(row); // Agregar la fila a la tabla
        });
    }

    // Método para mostrar el formulario de agregar/editar mesa
    showForm() {
        this.formMesa.reset(); // Resetear el formulario
        this.formularioMesa.style.display = 'block'; // Mostrar el formulario
    }

    // Método para ocultar el formulario de agregar/editar mesa
    hideForm() {
        this.formMesa.reset(); // Resetear el formulario
        this.formularioMesa.style.display = 'none'; // Ocultar el formulario
    }

    // Método para agregar o editar una mesa
    addMesa(event) {
```

```
        event.preventDefault(); // Prevenir el comportamiento por defecto del
formulario
        const id = document.getElementById('mesa-id').value;
        const numero = document.getElementById('mesa-numero').value;
        const capacidad = document.getElementById('mesa-capacidad').value;
        const ubicacion = document.getElementById('mesa-ubicacion').value;

        if (id) {
            // Si hay un ID, estamos editando una mesa existente
            const index = this.mesas.findIndex(m => m.id == id);
            this.mesas[index] = new Mesa(id, numero, capacidad, ubicacion);
        } else {
            // Si no hay ID, estamos agregando una nueva mesa
            const newId = this.mesas.length > 0 ? this.mesas[this.mesas.length -
1].id + 1 : 1;
            this.mesas.push(new Mesa(newId, numero, capacidad, ubicacion));
        }

        // Guardar las mesas en el localStorage
        localStorage.setItem('mesas', JSON.stringify(this.mesas));
        // Renderizar las mesas actualizadas
        this.renderMesas();
        // Ocultar el formulario
        this.hideForm();
    }

    // Método para editar una mesa existente
    editMesa(index) {
        const mesa = this.mesas[index];
        document.getElementById('mesa-id').value = mesa.id;
        document.getElementById('mesa-numero').value = mesa.numero;
        document.getElementById('mesa-capacidad').value = mesa.capacidad;
        document.getElementById('mesa-ubicacion').value = mesa.ubicacion;
        this.formularioMesa.style.display = 'block'; // Mostrar el formulario
    }

    // Método para eliminar una mesa
    deleteMesa(index) {
        this.mesas.splice(index, 1); // Eliminar la mesa del array
        localStorage.setItem('mesas', JSON.stringify(this.mesas)); // Actualizar
el localStorage
        this.renderMesas(); // Renderizar las mesas actualizadas
    }
}
```

```
// Inicializar el administrador de mesas
const mesaManager = new MesaManager();
```

reservas.js

```
class Reserva {
    // Constructor para crear una nueva reserva
    constructor(id, fecha, hora, numeroPersonas, estado) {
        this.id = id;
        this.fecha = fecha;
        this.hora = hora;
        this.numeroPersonas = numeroPersonas;
        this.estado = estado;
    }
}

class ReservaManager {
    // Constructor para inicializar el administrador de reservas
    constructor() {
        // Cargar reservas desde el localStorage o inicializar una lista vacía
        this.reservas = JSON.parse(localStorage.getItem('reservas')) || [];

        // Obtener referencias a los elementos del DOM
        this.reservasList = document.getElementById('reservas-list');
        this.formReserva = document.getElementById('form-reserva');
        this.addReservaBtn = document.getElementById('addReservaBtn');
        this.cancelarReservaBtn = document.getElementById('cancelarReservaBtn');
        this.formularioReserva = document.getElementById('formulario-reserva');

        // Asignar eventos a los botones
        this.addReservaBtn.addEventListener('click', () => this.showForm());
        this.cancelarReservaBtn.addEventListener('click', () => this.hideForm());
        this.formReserva.addEventListener('submit', (event) =>
this.addReserva(event));

        // Renderizar las reservas al iniciar
        this.renderReservas();
    }

    // Método para renderizar la lista de reservas en la tabla
    renderReservas() {
        this.reservasList.innerHTML = ''; // Limpiar la lista actual
        this.reservas.forEach((reserva, index) => {
```

```
            const row = document.createElement('tr'); // Crear una nueva fila
para cada reserva
            row.innerHTML = `
                <td>${reserva.id}</td>
                <td>${reserva.fecha}</td>
                <td>${reserva.hora}</td>
                <td>${reserva.numeroPersonas}</td>
                <td>${reserva.estado}</td>
                <td>
                    <button class="btn btn-warning"
onclick="reservaManager.editReserva(${index})">Editar</button>
                    <button class="btn btn-danger"
onclick="reservaManager.deleteReserva(${index})">Eliminar</button>
                </td>
            `;
            this.reservasList.appendChild(row); // Agregar la fila a la tabla
        });
    }

    // Método para mostrar el formulario de agregar/editar reserva
    showForm() {
        this.formReserva.reset(); // Resetear el formulario
        this.formularioReserva.style.display = 'block'; // Mostrar el formulario
    }

    // Método para ocultar el formulario de agregar/editar reserva
    hideForm() {
        this.formReserva.reset(); // Resetear el formulario
        this.formularioReserva.style.display = 'none'; // Ocultar el formulario
    }

    // Método para agregar o editar una reserva
    addReserva(event) {
        event.preventDefault(); // Prevenir el comportamiento por defecto del
formulario
        const id = document.getElementById('reserva-id').value;
        const fecha = document.getElementById('reserva-fecha').value;
        const hora = document.getElementById('reserva-hora').value;
        const numeroPersonas = document.getElementById('reserva-
numeroPersonas').value;
        const estado = document.getElementById('reserva-estado').value;

        if (id) {
            // Si hay un ID, estamos editando una reserva existente
            const index = this.reservas.findIndex(r => r.id == id);
```

```
            this.reservas[index] = new Reserva(id, fecha, hora, numeroPersonas,
estado);
        } else {
            // Si no hay ID, estamos agregando una nueva reserva
            const newId = this.reservas.length > 0 ?
this.reservas[this.reservas.length - 1].id + 1 : 1;
            this.reservas.push(new Reserva(newId, fecha, hora, numeroPersonas,
estado));
        }

        // Guardar las reservas en el localStorage
        localStorage.setItem('reservas', JSON.stringify(this.reservas));
        // Renderizar las reservas actualizadas
        this.renderReservas();
        // Ocultar el formulario
        this.hideForm();
    }

    // Método para editar una reserva existente
    editReserva(index) {
        const reserva = this.reservas[index];
        document.getElementById('reserva-id').value = reserva.id;
        document.getElementById('reserva-fecha').value = reserva.fecha;
        document.getElementById('reserva-hora').value = reserva.hora;
        document.getElementById('reserva-numeroPersonas').value =
reserva.numeroPersonas;
        document.getElementById('reserva-estado').value = reserva.estado;
        this.formularioReserva.style.display = 'block'; // Mostrar el formulario
    }

    // Método para eliminar una reserva
    deleteReserva(index) {
        this.reservas.splice(index, 1); // Eliminar la reserva del array
        localStorage.setItem('reservas', JSON.stringify(this.reservas)); //
Actualizar el localStorage
        this.renderReservas(); // Renderizar las reservas actualizadas
    }
}

// Inicializar el administrador de reservas
const reservaManager = new ReservaManager();
```

Link video:

https://uespe-my.sharepoint.com/:v:/g/personal/cepogo_espe_edu_ec/EXzlaD4fQVxPvRogLGgLww0BGFOSl0OBskGlQSTJ5tj5YQ?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJPbmVEcml2ZUZvckJ1c2luZXNzIiwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IldlYiIsInJlZmVycmFsTW9kZSI6InZpZXciLCJyZWZlcnJhbFZpZXciOiJNeUZpbGVzTGlua0NvcHkifX0&e=M1lkF1

Hosting

https://14365-cepogo.github.io/Tarea1.2_Integrativa_PogoCarlos/