

## EJERCICIO AJAX RESTAURANTE

### CONTEXTO

Se nos pide que realicemos una aplicación para la gestión de pedidos de un restaurante, donde el camarero irá con un teléfono cogiendo los pedidos en las mesas y enviará esos pedidos a la barra para bebidas y a la cocina para los platos a cocinar.

En la barra y la cocina habrá una cola de bebidas y platos a preparar cuando el barman o cocinero termine de preparar el producto, lo marcará como servido.

Para la gestión de esta aplicación utilizaremos el sistema de gestión de usuarios utilizado en la práctica USERPROFILE. Se tendrán que aplicar la misma lógica de registro, login, logout y areaPersonal. Y para continuar con el proyecto el usuario debe de iniciar sesión.

En la app se necesita la siguiente información: Mesas, bebidas y platos.

- **Mesas:**

Restaurante
Home
Cocina
Barra
Sala
Gestión
David Crespo Jorda

### Mesas

Nueva Mesa

Borrar	Modificar	Número	Comensales	Descripción
Borrar	Modificar	1	6	mesa de la esquina
Borrar	Modificar	2	4	mesa de la ventana
Borrar	Modificar	3	6	mesa de la segunda ventana
Borrar	Modificar	4	2	mesa de los enamorados de la esquina
Borrar	Modificar	5	8	mesa del centro derecha
Borrar	Modificar	6	10	mesa del fondo
Borrar	Modificar	7	10	Mesa 7 del centro
Borrar	Modificar	10	20	Mesa en privado

DWEC - Restaurante 2022 - David Crespo

Estructura de los datos :

```
const mesasSchema = mongoose.Schema({
  numero: {
    type: Number,
    required: true,
    min: 1,
    max: 100
  },
  comensales: {
    type: Number,
    required: true,
    min: 1,
    max: 50
  },
  descripcion: {
    type: String
  }
})
```

Se deben de validar los campos, mostrar alerta de creación y mostrar los errores que devuelve el servidor.

ENDPOINT: <https://restaurante.serverred.es/api/mesas>

<https://restaurante.serverred.es/api/comandas>

**NOTA:** Comprobar que no se pueda eliminar una mesa si existen comandas asociadas a la mesa.

- **Bebidas**

Restaurante
Home
Cocina
Barra
Sala
Gestión
David Crespo Jorda

## Bebidas

Nueva Bebida

Borrar	Modificar	Nombre	Precio
Borrar	Modificar	Caña	1.5
Borrar	Modificar	Cocacola	1.5
Borrar	Modificar	Agua Mineral	1
Borrar	Modificar	tanque	3
Borrar	Modificar	Fanta Naranja	1

DWEC - Restaurante 2022 - David Crespo

Estructura de los datos :

```
const bebidasSchema = mongoose.Schema({
  nombre: {
    type: String,
    required: true,
    min: 4,
    max: 60,
    trim: true
  },
  imagen: {
    type: String
  },
  precio: {
    type: Number,
    required: true,
    min: 0
  }
})
```

Se deben de validar los campos, mostrar alerta de creación y mostrar los errores que devuelve el servidor.

ENDPOINT: <https://restaurante.serverred.es/api/bebidas>

- **Platos**

Restaurante
Home
Cocina
Barra
Sala
Gestión
David Crespo Jorda

## Platos

Nuevo Plato

Borrar	Modificar	Nombre	Orden	Precio
Borrar	Modificar	Ensalada Mixta	Primero	4
Borrar	Modificar	Ensalada Especial	Primero	5
Borrar	Modificar	Macarrones a la boloñesa	Segundo	4
Borrar	Modificar	Fruta de temporada	Postre	2

DWEC - Restaurante 2022 - David Crespo

### Estructura de los datos :

```
const platosSchema = mongoose.Schema({
  nombre: {
    type: String,
    required: true,
    minlength: 4,
    maxlength: 60
  },
  imagen: {
    type: String
  },
  //orden : Primero, Segundo, Postre
  orden: {
    type: String,
    required: true,
  },
  precio: {
    type: Number,
    required: true,
    min: 0
  }
})
```

Se deben de validar los campos, mostrar alerta de creación y mostrar los errores que devuelve el servidor.

ENDPOINT : <https://restaurante.serverred.es/api/platos>

Unas vez creados los ficheros maestros de Mesas, bebidas y Platos. Vamos a crear una comanda nueva que

tendrá la siguiente vista.

- **Alta Comandas:**

Se debe de tener en cuenta las siguientes consideraciones:

- Se debe de validar el nombre, los comensales, pero las notas no son obligatorias.
- Las mesas se mostrará solo la numeración que se ha introducido, pero se guardará el id del registro de la mesa. Cuando se seleccione una mesa, esta cambiará de color a rojo con las clases “`mt-2 btn btn-danger p-3`”
- Cuando se seleccione una mesa se obtendrá el numero de comensales y las descripción asociada y se mostrará en la parte inferior.
- Igual en bebidas como en platos, los mostraremos con botones cada una de ellos, si se selecciona un botón este subirá a la lista de peticiones y se guardaran la información en un array que se mantendrá con los mismos datos que la tabla que se puesta en pantalla. Si se vuelve a seleccionar el mismo botón no se creará una nueva linea sino se aumentará la cantidad pedida en 1.
- Los platos se dividirán en el orden de: Primeros, Segundos y Postre.
- Una vez realizado toda la comanda se enviará al servidor mediante la siguiente estructura de datos.
- Hay que tener en cuenta que no es necesario rellenar el usuario, ya que por defecto el sistema ya lo reconoce, los estados deben de ser “Pendiente” y la fecha Entrada y la de Salida no se informarán en

esta petición.

- Para el correcto funcionamiento no se debe de refrescar la página en ningún caso, utilizar el DOM, capturar los errores que nos pueda dar el API y una vez grabada la comanda enviar el flujo a el listado de comanda.

```
const bebidasSchema = mongoose.Schema({
  nombre: {
    type: String,
    required: true,
    min: 4,
    max: 60,
    trim: true
  },
  cantidad: {
    type: Number,
    required: true,
    min: 0
  },
  precio: {
    type: Number,
    required: true,
    min: 0
  },
  // estado : Pendiente, Servido.
  estado: {
    type: String,
    required: true,
  }
})
```

```
const platosSchema = mongoose.Schema({
  nombre: {
    type: String,
    required: true,
    minlength: 4,
    maxlength: 60
  },
  cantidad: {
    type: Number,
    required: true,
    min: 0
  },
  precio: {
    type: Number,
    required: true,
    min: 0
  },
  // estado : Pendiente, Servido.
  estado: {
    type: String,
    required: true,
  },
})
```

```
const comandasSchema = mongoose.Schema({
  nombre: {
    type: String,
    required: true,
    minlength: 4,
    maxlength: 60,
    trim: true
  },
  comensales: {
    type: Number,
    required: true,
    min: 1
  },
  // estado : Pendiente, Servido, Cobrado.
  estado: {
    type: String,
    required: true,
  },
  fechaEntrada: {
    type: Date,
    default: Date.now
  },
  fechaSalida: {
    type: Date,
  },
  // Vinculación de comanda con mesa
  mesa: {
    type: mongoose.Schema.Types.ObjectId,
    required: true,
    ref: 'mesa'
  },
  // Vinculación de comanda con usuario
  user: {
    type: mongoose.Schema.Types.ObjectId,
    required: true,
    ref: 'user'
  },
  // bebidas
  bebidas: [bebidasSchema],
  // Comidas
  platos: [platosSchema],
  notas: {
    type: String
  }
})
```

Los ENDPOINTS serán :

- GET → ["https://restaurante.serverred.es/api/mesas"](https://restaurante.serverred.es/api/mesas)
- GET → ["https://restaurante.serverred.es/api/bebidas"](https://restaurante.serverred.es/api/bebidas)
- GET → ["https://restaurante.serverred.es/api/platos"](https://restaurante.serverred.es/api/platos)
- POST → <https://restaurante.serverred.es/api/comandas>

- **Listado de Comandas:**

en este apartado se mostrarán solo las comandas pendiente, una vez que el cliente pida la cuenta la comanda no estará en el listado.

En el listado el camarero solo podrá acceder para ver o cambiar la bebida o los platos pedidos y la forma de la página será la siguiente:

	Nombre	Mesa	Com.	Camarero	H Entrada
+Bebidas +Platos	Pepe	1	4	David Crespo Jorda	18:38
+Bebidas +Platos	Maria	4	2	David Crespo Jorda	23:27

DWECC - Restaurante 2022 - David Crespo

Los ENDPOINTS serán los siguientes:

- GET → <https://restaurante.serverred.es/api/camareros>
- GET → ["https://restaurante.serverred.es/api/mesas"](https://restaurante.serverred.es/api/mesas)
- GET → ["https://restaurante.serverred.es/api/comandas"](https://restaurante.serverred.es/api/comandas)