

## Manual Técnico del Proyecto de E-commerce

### 1. Conexión y Configuración de la Base de Datos

#### Base de Datos: MySQL

El sistema utiliza MySQL como base de datos para almacenar la información de productos, clientes, carritos, y órdenes de pago.

#### Configuración:

- **Nombre de la BD:** ecommerce\_db
- **Tablas Principales:**
  - productos: Almacena información de productos.
  - clientes: Almacena la información de los clientes registrados.
  - carritos: Guarda los carritos de compra de los usuarios (cliente o usuarios provisionales).
  - ordenes\_pago: Detalles de las órdenes de compra.

#### Conexión:

La conexión a la base de datos se realiza desde el archivo server.js usando Node.js con la librería mysql2 y con MySQL como Base de Datos.

```
1 // Configurar la conexión a la base de datos
2 const db = mysql.createConnection({
3   host: 'localhost',
4   port: 3308, // Cambia este puerto si es necesario
5   user: 'root',
6   password: 'root',
7   database: 'db_ecommerce'
8 });
9
10 // Conectar a la base de datos
11 db.connect(err => {
12   if (err) {
13     console.error('Error connecting to the database:', er
14 r);    return;
15   }
16   console.log('Connected to the database.');
```

## Migración y Creación de Tablas:

Las tablas se pueden crear usando las sentencias SQL definidas en el proyecto, asegurándose de que las relaciones estén correctamente configuradas.

```
1 CREATE TABLE clientes (  
2     id INT AUTO_INCREMENT PRIMARY KEY,  
3     nombre VARCHAR(255) NOT NULL,  
4     apellido VARCHAR(255) NOT NULL,  
5     email VARCHAR(255) UNIQUE NOT NULL,  
6     password VARCHAR(255) NOT NULL  
7 );  
8  
9  
10 -- Tabla de carritos (un único carrito por cliente)  
11 CREATE TABLE carritos (  
12     id INT AUTO_INCREMENT PRIMARY KEY,  
13     numero_orden VARCHAR(50) NOT NULL UNIQUE,  
14     cliente_id INT NOT NULL,  
15     cantidad_productos INT NOT NULL,  
16     monto_total DECIMAL(10, 2) NOT NULL,  
17     fecha_creacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
18     FOREIGN KEY (cliente_id) REFERENCES clientes(id) ON DELETE CASCADE  
19 );
```

## 2. Configuración de las API's

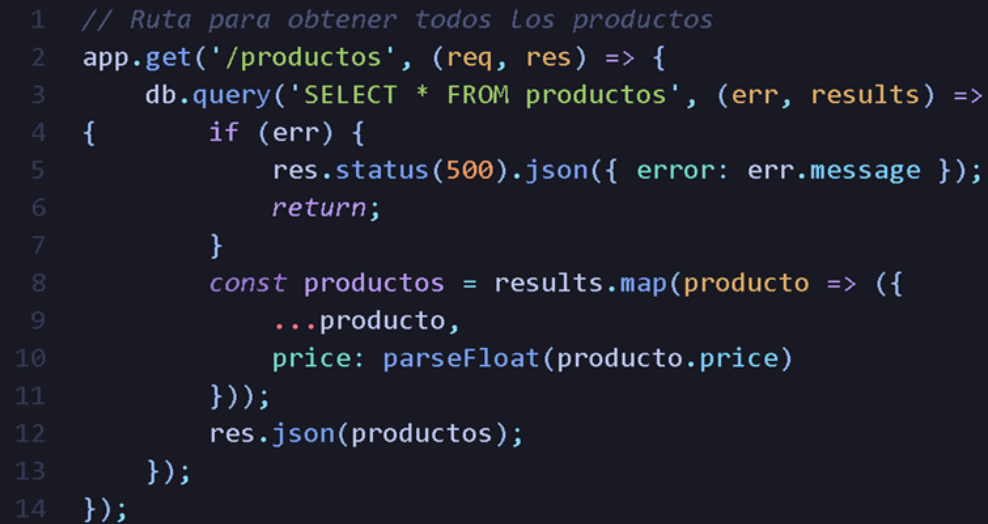
### API REST:

El backend del proyecto utiliza una API REST para interactuar con la base de datos. Se emplea **Express.js** como framework para crear las rutas.

### Rutas Principales:

- **Productos:**
  - **GET /api/productos:** Obtener todos los productos.
  - **GET /api/productos/:id:** Obtener un producto por su ID.
- **Carritos:**
  - **POST /api/carrito:** Crear un nuevo carrito de compra.
  - **PUT /api/carrito/:id:** Actualizar un carrito con nuevos productos.
- **Orden de Compra:**
  - **POST /api/orden:** Crear una nueva orden de pago.

### Ejemplo de configuración:



```
1 // Ruta para obtener todos los productos
2 app.get('/productos', (req, res) => {
3     db.query('SELECT * FROM productos', (err, results) =>
4     {
5         if (err) {
6             res.status(500).json({ error: err.message });
7             return;
8         }
9         const productos = results.map(producto => ({
10             ...producto,
11             price: parseFloat(producto.price)
12         }));
13         res.json(productos);
14     });
15 });
```

## 3. Arquitectura del Software

### Frontend:

- **Tecnología:** React.js
- **Componentes Clave:**
  - ProductGrid.js: Para mostrar todos los productos.
  - Carrito.js: Manejo del carrito de compras.
  - Confirmacion.js: Resumen final de la compra y generación de ticket PDF.

### Backend:

- **Tecnología:** Node.js y Express.js
- **Base de datos:** MySQL
- **API REST:** Se utiliza para la comunicación entre el frontend y backend.

### Almacenamiento Temporal:

El sistema utiliza localStorage en el navegador para almacenar temporalmente los datos del carrito cuando no hay conexión con la base de datos.

### Generación de Tickets:

Usa la librería **jsPDF** en el frontend para generar tickets en formato PDF con la información del pedido.

#### 4. Características Mínimas del Sistema

- **Servidor:**
  - **CPU:** Mínimo 2 núcleos
  - **Memoria RAM:** Mínimo 4 GB
  - **Almacenamiento:** 20 GB disponibles
  - **Node.js:** Versión 14+
  - **MySQL:** Versión 5.7+
- **Cliente:**
  - **Navegador Web:** Soporte para ECMAScript 6 (Chrome, Firefox o Safari)
  - **Resolución Mínima:** 1280x720

#### 5. Instalación del Proyecto

##### Requisitos Previos:

1. Tener instalado Node.js y npm.
2. Tener MySQL instalado y configurado.

##### Pasos de Instalación:

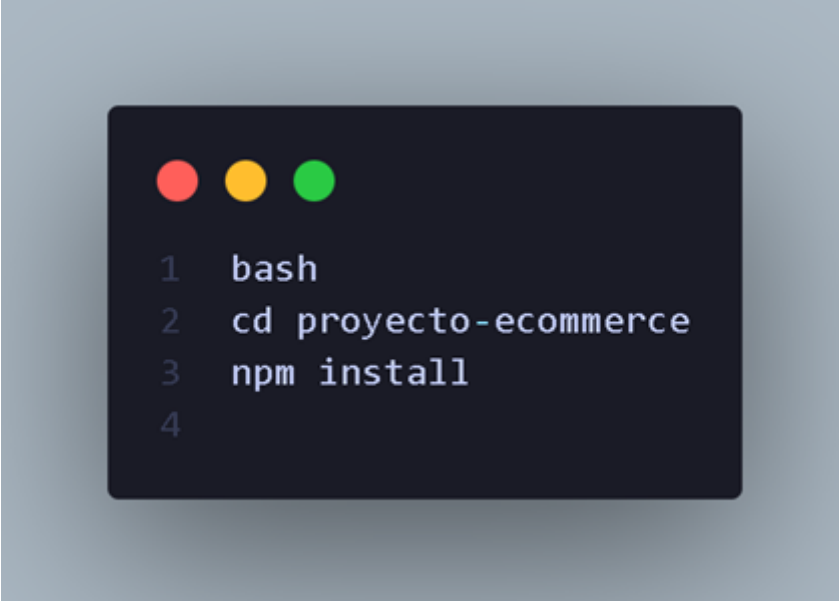
Todos los próximos pasos se realizan sobre consola

1. Clonar el repositorio:

A screenshot of a terminal window with a dark background and light-colored text. The terminal has three colored window control buttons (red, yellow, green) in the top-left corner. The text inside the terminal shows a list of commands with line numbers: 1 bash, 2 git clone https://github.com/BrunoDiazMartinez/E-commerce.git, and 3. The text is in a monospaced font.

```
1 bash
2 git clone https://github.com/BrunoDiazMartinez/E-commerce.git
3
```

2. Instalar dependencias:



```
1 bash
2 cd proyecto-ecommerce
3 npm install
4
```

3. Iniciar el servidor (*Abrir una consola específica que apunte la carpeta*):



```
1 node .\src\service\server.js
```

4. Iniciar el sistema (*Abrir una consola específica que apunte la carpeta*):



```
1 npm start
```