Documentación del Proyecto

App Sabor Urbano

Parcial 1

Tecnicatura Superior en Desarrollo de Software – IFTS 29

Materia: Desarrollo de Sistemas Web (Back End)

Comsion B.

Grupo: N° 4 (FastRoute)

Integrantes:

- Nicolini, Guido
- Vazquez, Ian
- Caeiro, Felicitas

1. Descripción del Proyecto

La aplicación desarrollada es una API REST para el restaurante **“Sabor Urbano”**, un bodegón porteño con más de 40 años de historia.
El sistema permite gestionar la información de **clientes**, **productos** y **pedidos**, utilizando Node.js y Express. Los datos actualmente se encuentran almacenados en archivos JSON, para la siguiente entrega se incorporara el uso de Mongo.

Se implementaron controladores, modelos, servicios y rutas para cada entidad, respetando una arquitectura organizada en capas.

Además, se incorporó **Pug como motor de plantillas** para renderizar vistas dinámicas, lo que permite mostrar datos de clientes, productos y pedidos en el navegador.

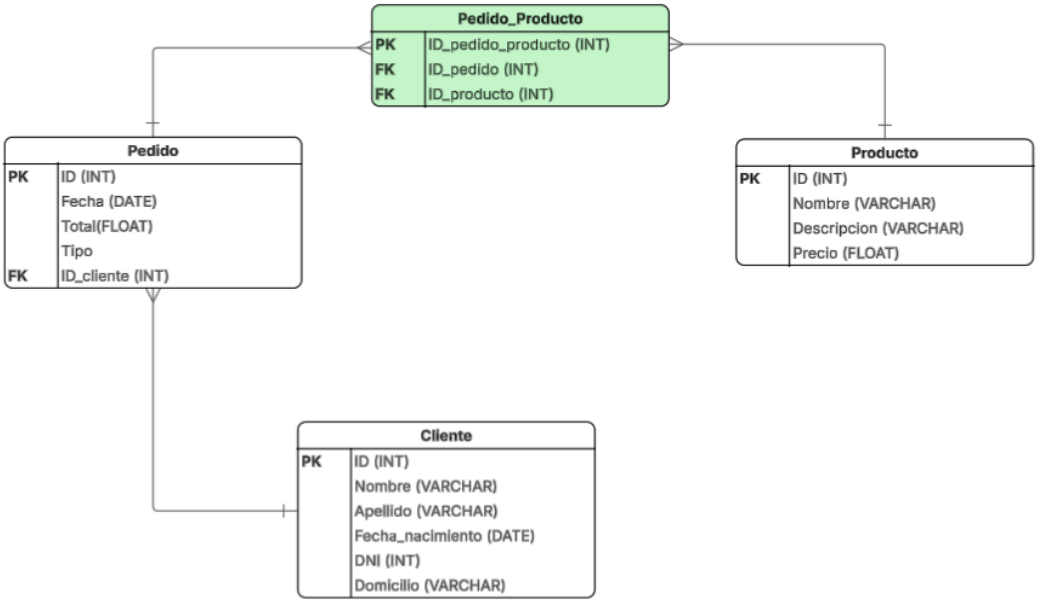
Repositorio en github: https://github.com/felicaeiro/IFTS29-Grupo4-SaborUrbano

2. Estructura del Proyecto

```
sabor-urbano-app/
│
├── config/
│   ├── db.js
│   └── server.js
│
├── controllers/
│   ├── clienteController.js
│   ├── pedidoController.js
│   └── productoController.js
│
├── models/
│   ├── clienteModel.js
│   ├── pedidoModel.js
│   └── productoModel.js
│
├── routes/
│   ├── clienteRoutes.js
│   ├── pedidoRoutes.js
│   └── productoRoutes.js
│
```

```
├── services/
│   ├── clienteService.js
│   ├── pedidoService.js
│   └── productoService.js
│
├── views/
│   ├── index.pug
│   ├── cliente.pug
│   ├── pedido.pug
│   └── producto.pug
│
├── public/
│   ├── css/
│   ├── js/
│   └── data-base/
│
├── middleware/
│   └── authMiddleware.js
│
├── tests/
│
├── .env
├── .gitignore
├── package.json
└── README.md
```

3. Entidades y Relaciones



4. Funcionamiento de la Aplicación

1. **Inicio del servidor**: se ejecuta con `npm run dev`, levantando Express en el puerto configurado.

2. **Manejo de datos**: todos los datos se leen y escriben en los archivos JSON ubicados en `public/data-base`.

3. **Rutas**:
   - `/clientes` → CRUD de clientes.
   - `/pedidos` → CRUD de pedidos.
   - `/productos` → CRUD de productos.

4. **Servicios**: encapsulan la lógica de acceso y persistencia de datos.

5. **Controladores**: gestionan la lógica de negocio y responden a las solicitudes HTTP.

6. **POO**: cada entidad tiene su **modelo** con propiedades definidas (`Cliente`, `Pedido`, `Producto`).

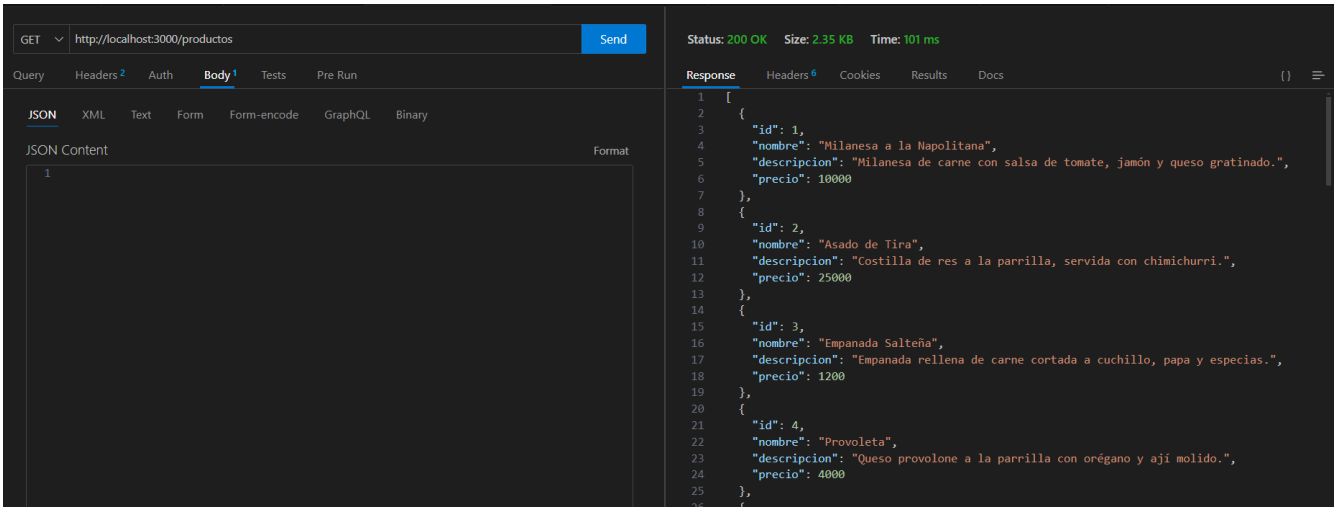
7. **Pug:** motor de plantillas que renderiza vistas dinámicas:

    - `index.pug`: página de inicio del sistema.
    - `producto.pug`: catálogo de productos.
    - `detalleProducto.pug:` vista detallada del producto seleccionado.
    - `pedido.pug`: visualización de pedidos.
    - `editarPedido.pug:` formulario para editar un pedido existente.
    - `nuevoPedido.pug:` formulario para crear un pedido nuevo.

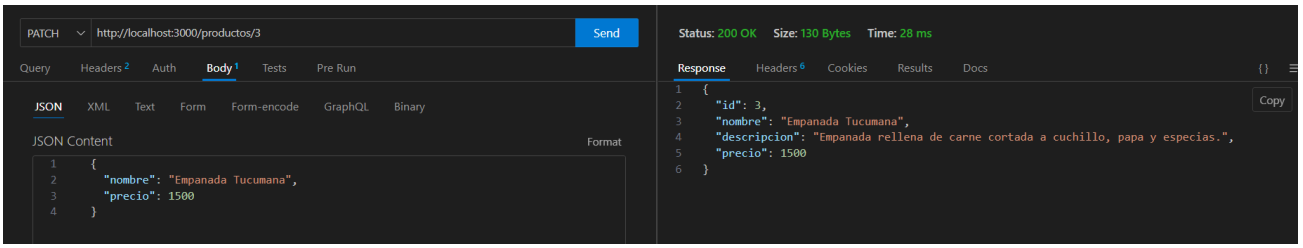
5. Pruebas

Se realizaron pruebas de los endpoints con **ThunderClient/Postman**:

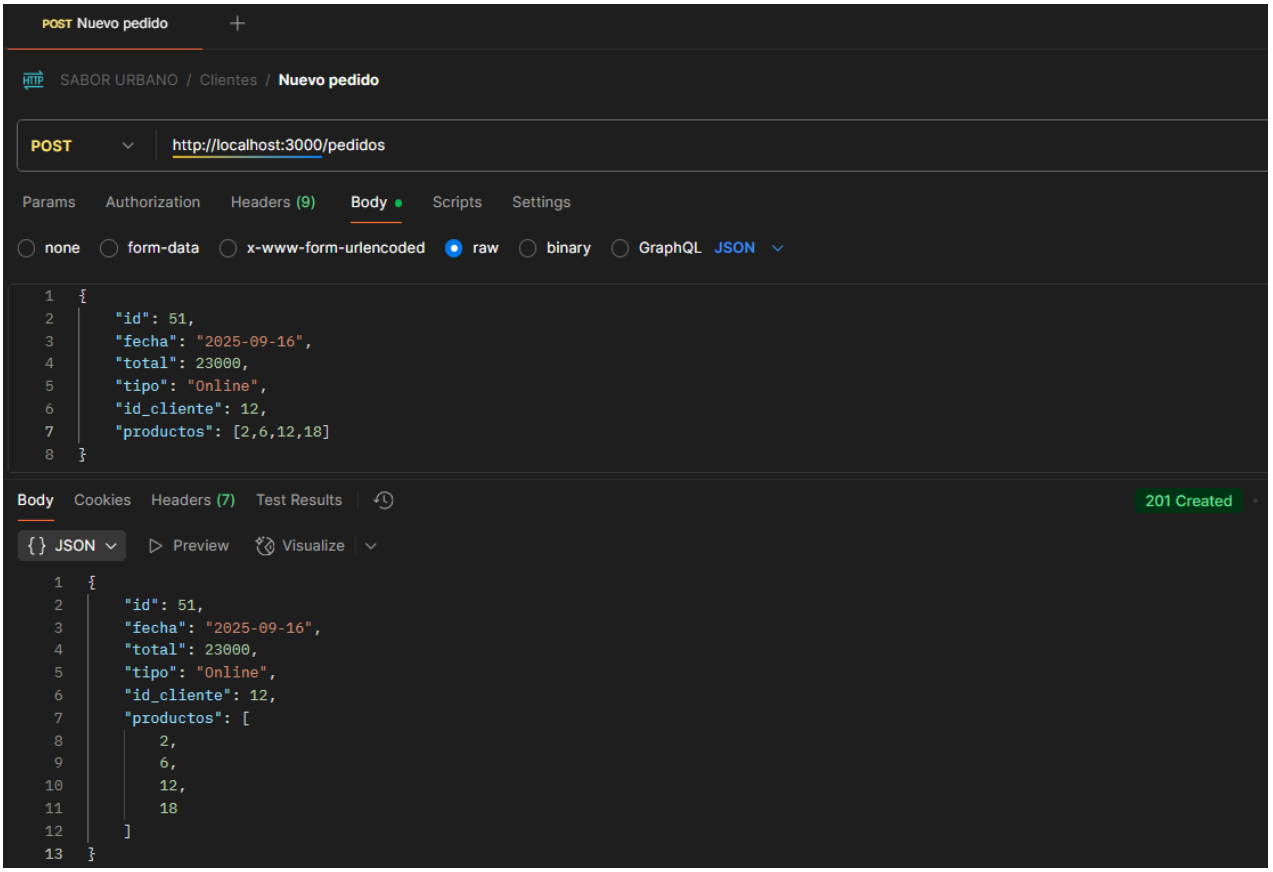
- **GET /productos** devuelve la lista de todos los productos.



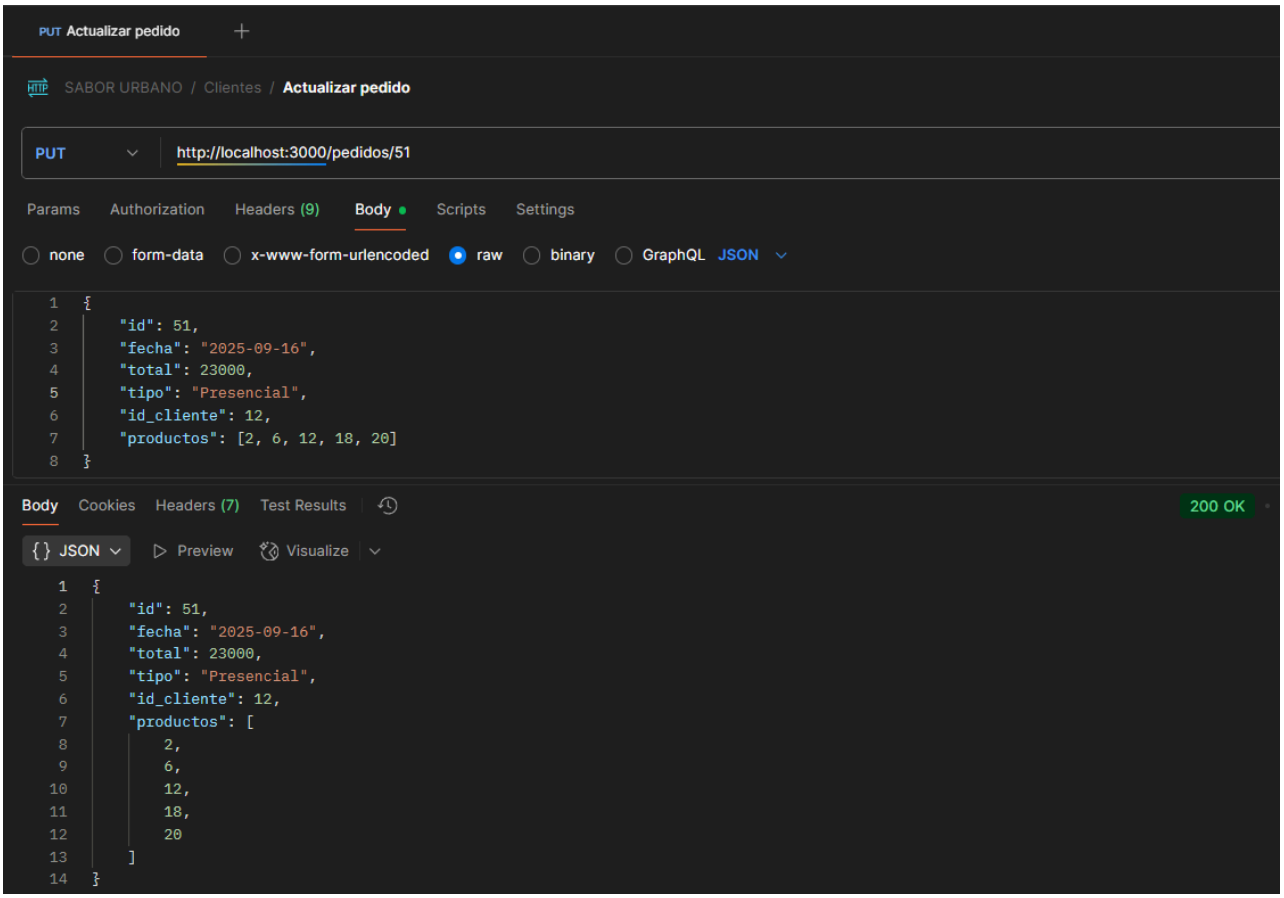
- **PATCH /productos/:id** actualiza solo los campos enviados por body del ítem correspondiente al id.



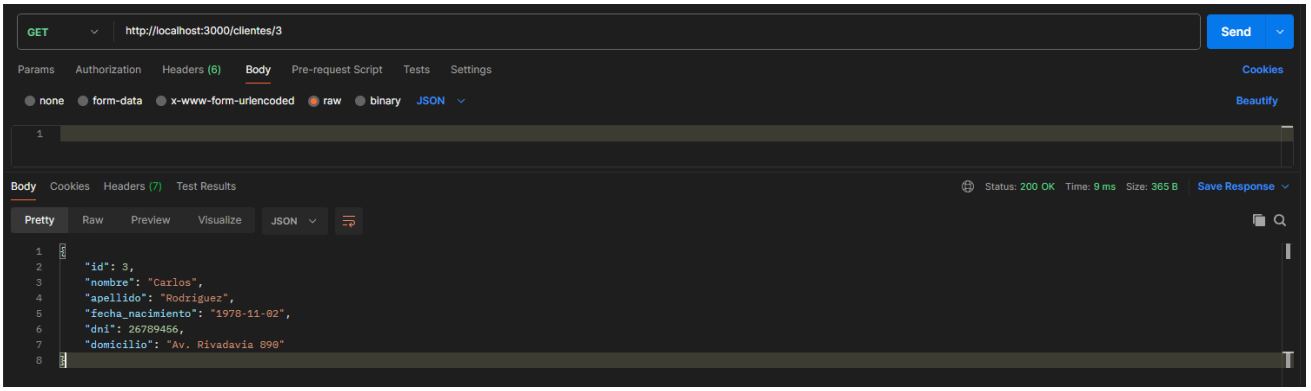
- **POST /pedidos** crea un nuevo pedido asociado a un cliente.

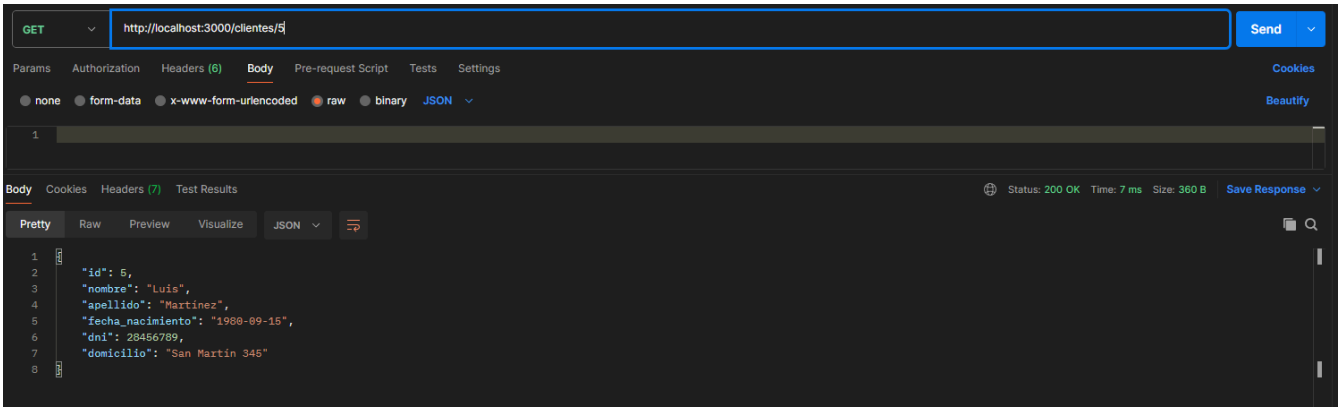


- **PUT /pedidos/:id** actualiza un nuevo pedido asociado a un cliente.

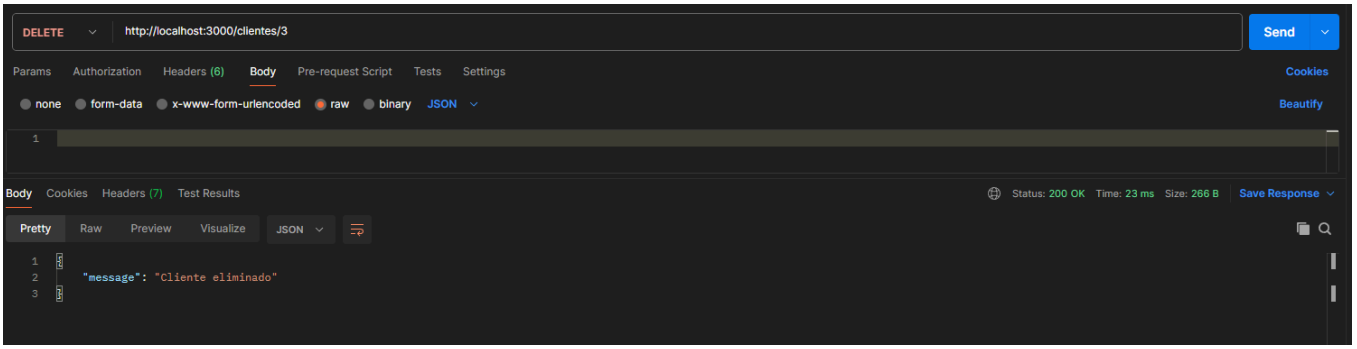


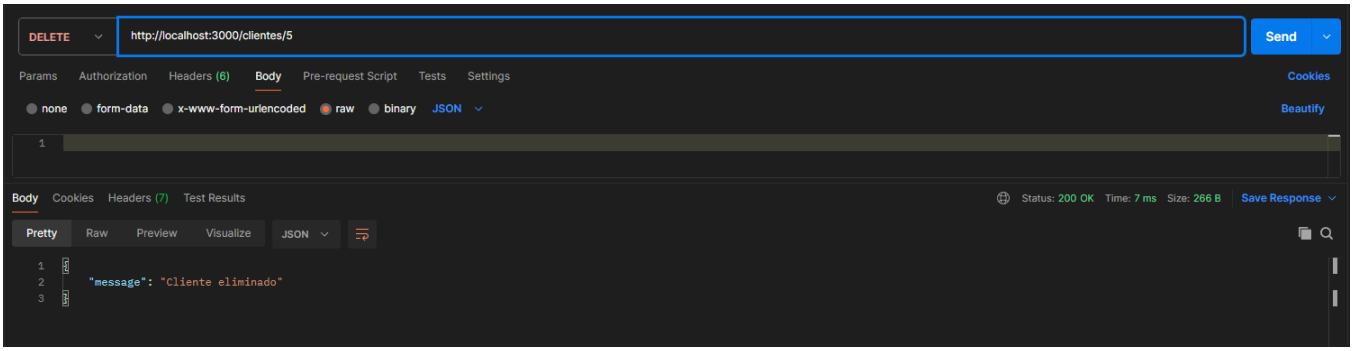
- **GET/clientes/:id** obtiene un cliente por el ID





- **DELETE /clientes/:id** elimina un cliente por el ID.





6. Roles de los Integrantes

Todos los integrantes del grupo trabajamos de manera colaborativa en todo el proyecto, ya que queríamos aprender y comprender cada etapa del proceso de desarrollo.
Esto incluyó:

- Configuración del servidor y estructura de carpetas.
- Desarrollo de modelos y servicios.
- Desarrollo de controladores y rutas.
- Integración de vistas con Pug.
- Pruebas de los endpoints con ThunderClient o Postman.
- Documentación y presentación del proyecto.

Cada miembro aportó en todas las áreas, compartiendo tareas y conocimientos para asegurar que todos entendieran el funcionamiento completo de la aplicación.

8. Bibliografía y Recursos

- Documentación oficial de Node.js
- Documentación oficial de Express
- Documentación oficial de Pug
- Documentación oficial de Mongo
- Apuntes de la materia **Desarrollo de Sistemas Web (Back End) – IFTS 29**
- Videos de apoyo: YouTube *BroCode - MongoDB tutorial for beginners*.