

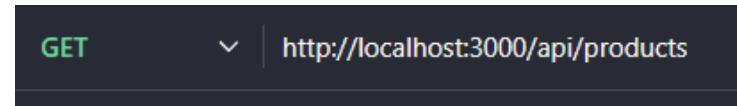
Asignatura	Datos del estudiante	Fecha
Sistemas Web	Nombres: Carlos Sebastian Apellidos: Pilamunga Quiroga	05/12/2025
Semestre: Cuarto	Cédula: 1751889278	
Tutor: Ing. Liliana Nogales		

CRUD COMPLETO. (getall, getbyid, delete, update, create)

Primero le mandamos a correr el proyecto

```
C:\Users\Carlos\Desktop\web_inventario>cd client
C:\Users\Carlos\Desktop\web_inventario\client>node app.js
[dotenv@017.2.3] injecting env (5) from .env -- tip: 📁 encrypt with Dotenvx: https://dotenvx.com
[dotenv@017.2.3] injecting env (6) from .env -- tip: ⚙️ specify custom .env file path with { path: '/custom/path/.env' }
Servidor en http://localhost:3000
Conectado a la base de datos PostgreSQL
```

- **Getall**



Aquí me muestra todos los datos que tengo en este caso solo me muestra dos datos

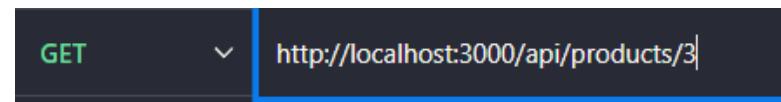
“<http://localhost:3000/api/products>” seleccionamos el método GET

The screenshot shows the Postman interface with a request to "http://localhost:3000/api/products" using the "GET" method. The "Params" tab is selected. The response body is shown in "Pretty" format:

```

1 [
2   {
3     "id": 2,
4     "nombre": "Juego de sábanas Queen",
5     "descripcion": "Juego de sábanas 180 hilos, color azul.",
6     "precio": "89.50",
7     "stock": 15,
8     "categoria": null,
9     "activo": true,
10    "fecha_creacion": "2025-12-03T04:00:51.827Z"
11  },
12  {
13    "id": 3,
14    "nombre": "Polera Overdsize Negra",
15    "descripcion": "Polera de algodón modelo oversize unisex",
16    "precio": "49.99",
17    "stock": 45,
18    "categoria": null,
19    "activo": true,
20    "fecha_creacion": "2025-12-03T04:16:49.848Z"
21  }
22 ]
```

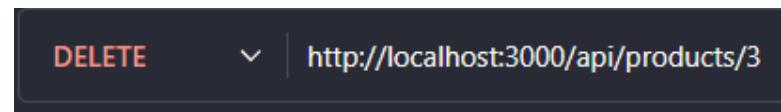
## • Getbyid



Aquí solamente pedí que me muestre el producto con ID 3 “http://localhost:3000/api/products/3”

A screenshot of a browser or API testing tool. The top bar shows a GET method and the URL http://localhost:3000/api/products/3. The main area displays the response body in JSON format, showing a single product object with ID 3.

## • Delete



En el método delete vamos a eliminar el dato con ID 3 una vez eliminado al momento de buscar ese dato por su ID 3 no va a aparecer

A screenshot of a browser or API testing tool. The top bar shows a DELETE method and the URL http://localhost:3000/api/products/3. The main area displays the response body in JSON format, showing an error message indicating the product was not found.

Y si mandamos otro método GET podemos verificar que no existe el dato

```

1  [
2   {
3     "id": 2,
4     "nombre": "Juego de sebas Queen",
5     "descripcion": "Juego de sábanas 180 hilos, color azul.",
6     "precio": "89.50",
7     "stock": 15,
8     "categoria": null,
9     "activo": true,
10    "fecha_creacion": "2025-12-03T04:00:51.827Z"
11  }
12 ]

```

- **Update**

Aquí vamos a actualizar solamente el nombre en vamos a cambiar de “Juego de webQueen”

” a “Sistemas Web“ debemos seleccionar el método PUT y ponemos

“<http://localhost:3000/api/products/2>” y le damos SEND

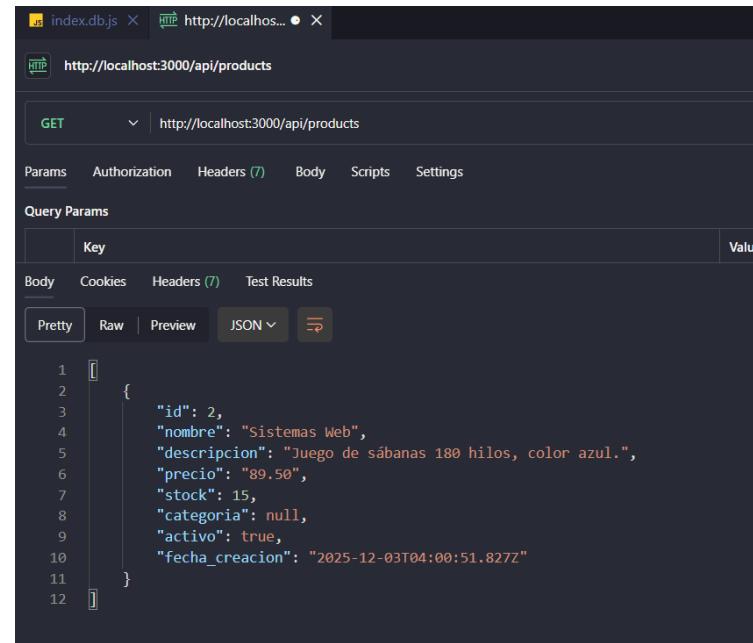
```

1  [
2   {
3     "id": 2,
4     "nombre": "Sistemas Web",
5     "descripcion": "Juego de sábanas 180 hilos, color azul.",
6     "precio": "89.50",
7     "stock": 15,
8     "categoria": null,
9     "activo": true,
10    "fecha_creacion": "2025-12-03T04:00:51.827Z"
11  }
12 ]

```

Status: 200 OK Time: 144 ms Size: 433 B

Ahora para ver si se actualizo hacemos un GET nuevamente y se verán los cambios

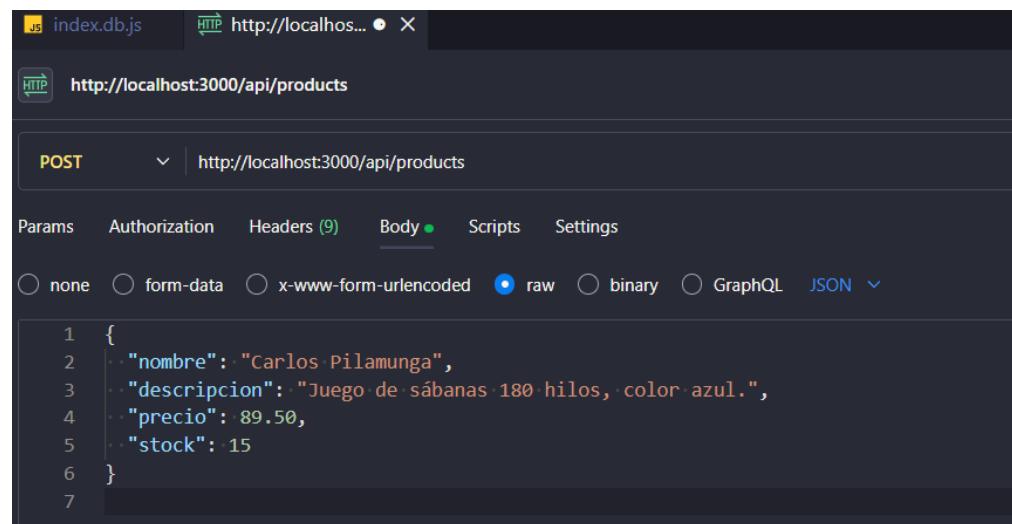


```
1 [ 2   { 3     "id": 2, 4     "nombre": "Sistemas Web", 5     "descripcion": "Juego de sábanas 180 hilos, color azul.", 6     "precio": "89.50", 7     "stock": 15, 8     "categoria": null, 9     "activo": true, 10    "fecha_creacion": "2025-12-03T04:00:51.827Z" 11  } 12 ]
```

- **Create**

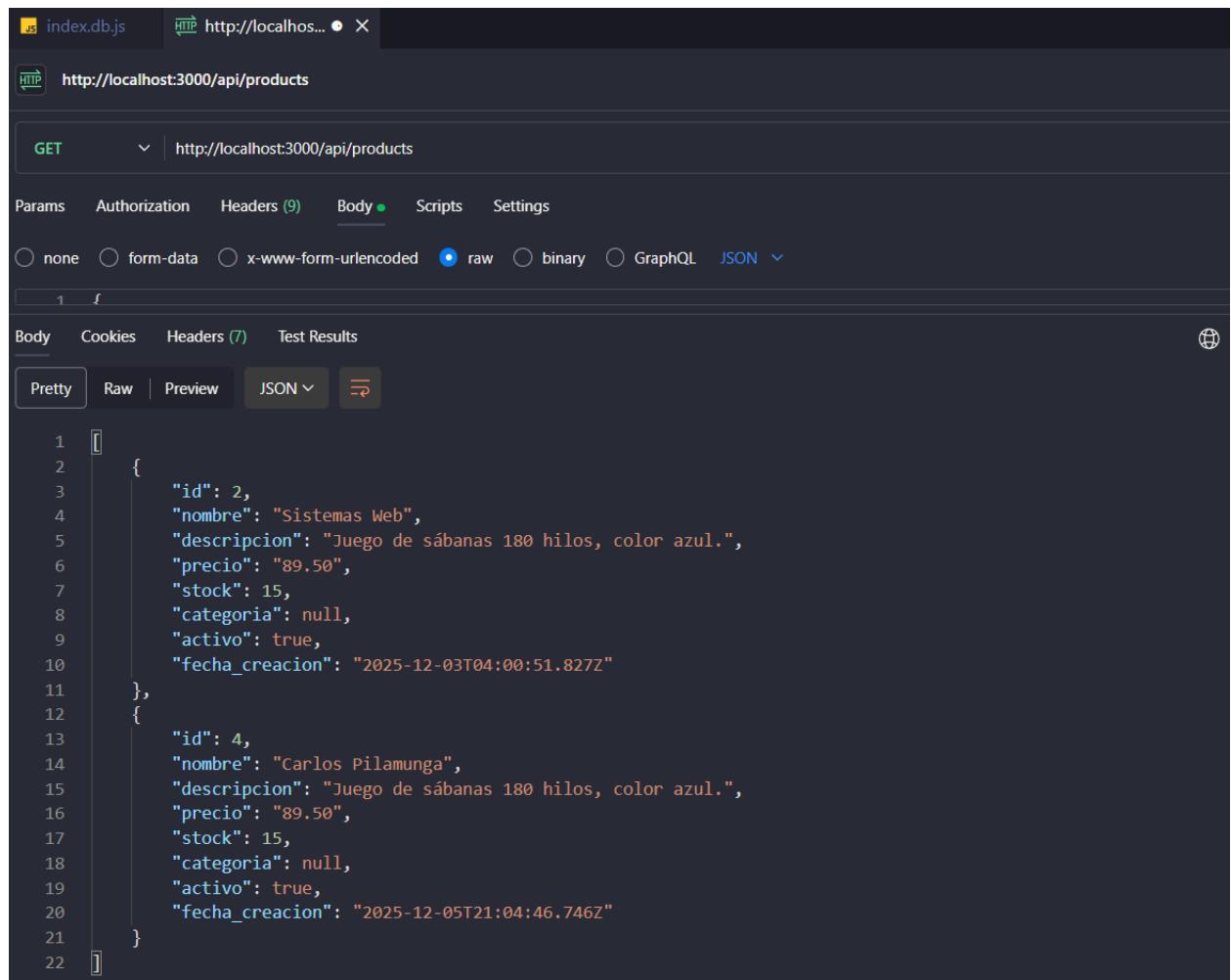
Ahora vamos a crear datos seleccionamos el método POST con esta ruta url como en todas “<http://localhost:3000/api/products>” para insertar damos clic BODY luego en RAW y tiene que estar en JSON

Dato que se creara



```
1 { 2   "nombre": "Carlos Pilamunga", 3   "descripcion": "Juego de sábanas 180 hilos, color azul.", 4   "precio": 89.50, 5   "stock": 15 6 }
```

Después hacemos nuevamente un método GET para verificar si se insertó el dato



The screenshot shows a Postman interface with the following details:

- Request Type: GET
- URL: <http://localhost:3000/api/products>
- Method: GET
- Headers: Authorization, Headers (9), Body (green dot), Scripts, Settings
- Body Type: raw (selected)
- Body Content:

```
1 [  
2 {  
3   "id": 2,  
4   "nombre": "Sistemas Web",  
5   "descripcion": "Juego de sábanas 180 hilos, color azul.",  
6   "precio": "89.50",  
7   "stock": 15,  
8   "categoria": null,  
9   "activo": true,  
10  "fecha_creacion": "2025-12-03T04:00:51.827Z"  
11 },  
12 {  
13   "id": 4,  
14   "nombre": "Carlos Pilamunga",  
15   "descripcion": "Juego de sábanas 180 hilos, color azul.",  
16   "precio": "89.50",  
17   "stock": 15,  
18   "categoria": null,  
19   "activo": true,  
20   "fecha_creacion": "2025-12-05T21:04:46.746Z"  
21 }  
22 ]
```
- Body Options: Pretty, Raw, Preview, JSON (selected)
- Test Results: Not visible in the screenshot

The JSON response shows two products inserted into the database.