

**UNIVERSIDAD MARIANO GÁLVEZ DE
GUATEMALA**
**FACULTAD DE INGENIRÍA EN SISTEMAS DE
INFORMACION.**

CURSO: PROGRAMACION II.

SECCION; A

**INGENIERO ELMER JOSUE ESTABAN
CHILEL**

TAREA NO. 1

ANDERSON JOSUE RAMOS BAUTISTA
JOCOTENAGO, 27 DE JULIO DE 2025

POST CLIENTES:

The screenshot shows the PostgreSQL client interface with the 'clientes' table selected. The table has columns: id, nombre, correo, telefono, created_at, and updated_at. A SQL query is entered in the editor, and the 'Tables' panel shows the 'clientes' table with 6 records.

id	nombre	correo	telefono	created_at	updated_at
1	Carlos	carlos@carlos.com	900.0	2023-01-01 00:00:00	2023-01-01 00:00:00
2	Luis	luis@luis.com	900.0	2023-01-01 00:00:00	2023-01-01 00:00:00
3	Carlos	carlos@carlos.com	900.0	2023-01-01 00:00:00	2023-01-01 00:00:00
4	Carlos	carlos@carlos.com	900.0	2023-01-01 00:00:00	2023-01-01 00:00:00
5	Carlos	carlos@carlos.com	900.0	2023-01-01 00:00:00	2023-01-01 00:00:00
6	Carlos	carlos@carlos.com	900.0	2023-01-01 00:00:00	2023-01-01 00:00:00

```

INSERT INTO "clientes" (
  "id", "nombre", "correo", "telefono", "created_at",
  "updated_at"
) VALUES (DEFAULT, 'CARLOS', 'CARLOS@CARLOS.COM', 900.0,
  CURRENT_TIMESTAMP, CURRENT_TIMESTAMP);
  
```

POST: PRODUCTOS

The screenshot shows the PostgreSQL client interface with the 'productos' table selected. The table has columns: id, nombre, precio, stock, created_at, and updated_at. A SQL query is entered in the editor, and the 'Tables' panel shows the 'productos' table with 6 records.

id	nombre	precio	stock	created_at	updated_at
1	Carota	1200.0	10	2023-01-01 00:00:00	2023-01-01 00:00:00
2	Carota	1200.0	10	2023-01-01 00:00:00	2023-01-01 00:00:00
3	Carota	1200.0	10	2023-01-01 00:00:00	2023-01-01 00:00:00
4	Carota	1200.0	10	2023-01-01 00:00:00	2023-01-01 00:00:00
5	Carota	1200.0	10	2023-01-01 00:00:00	2023-01-01 00:00:00
6	Carota	1200.0	10	2023-01-01 00:00:00	2023-01-01 00:00:00

```

INSERT INTO "productos" (
  "id", "nombre", "precio", "stock", "created_at", "updated_at"
) VALUES (DEFAULT, 'CAROTA', 1200.0, 10, CURRENT_TIMESTAMP,
  CURRENT_TIMESTAMP);
  
```

POST: PEDIDOS.

The screenshot shows a REST client interface with a POST request to `http://localhost:8080/api/pedidos/create`. The request body is a JSON object: `{ "id_cliente": 2, "fecha": "2023-07-24", "total": 1256.75 }`. The response is a 201 status code with a location header. Below the request, the SQL query is visible: `INSERT INTO "pedidos" ("id_pedido", "id_cliente", "fecha", "total", "created_at", "updated_at") VALUES (DEFAULT, $1, $2, $3, $4, $5) RETURNING "id_pedido", "id_cliente", "fecha", "total", "created_at", "updated_at";`. On the right, the database schema is shown with the `pedidos` table selected. The table has columns: `id_pedido` (integer), `id_cliente` (integer), `fecha` (date), and `total` (numeric). The table contains two records.

id_pedido	id_cliente	fecha	total
1	2	2023-07-23	126.75
2	2	2023-07-24	1256.75

POST:DETALLES.

The screenshot shows a REST client interface with a POST request to `http://localhost:8080/api/ordenes/create`. The request body is a JSON object: `{ "id_pedido": 1, "id_producto": 2, "cantidad": 3, "subtotal": 1200.56 }`. The response is a 201 status code with a location header. Below the request, the SQL query is visible: `INSERT INTO "ordenes" ("id_detalle", "id_pedido", "id_producto", "cantidad", "subtotal", "created_at", "updated_at") VALUES (DEFAULT, $1, $2, $3, $4, $5, $6) RETURNING "id_detalle", "id_pedido", "id_producto", "cantidad", "subtotal", "created_at", "updated_at";`. On the right, the database schema is shown with the `ordenes` table selected. The table has columns: `id_detalle` (integer), `id_pedido` (integer), `id_producto` (integer), `cantidad` (integer), and `subtotal` (numeric). The table contains two records.

id_detalle	id_pedido	id_producto	cantidad	subtotal
1	1	2	3	1200.56
2	1	2	3	1200.56

GET:CLIENTES

The image shows two side-by-side screenshots. The left screenshot is from VS Code, displaying a REST client request to `http://localhost:8080/api/clientes/` with a `JSON` response. The right screenshot is from Postman, showing the same endpoint and a table of client data.

VS Code REST Client Response:

```
{
  "id": 1,
  "nombre": "Carlos",
  "correo": "carlosramos@gmail.com",
  "telefono": "45678901",
  "createdAt": "2025-07-23T20:40:44.876Z",
  "updatedAt": "2025-07-23T20:40:44.876Z"
}
```

Postman Response Table:

id	nombre	correo	telefono
1	Carlos	carlosramos@gmail.com	45678901
2	Luisa	luisa@gmail.com	987654321
3	Esteban	esteban@gmail.com	123456789
4	Felipe	felipe@gmail.com	987654321
5	Rita	rita@gmail.com	123456789
6	Luis	luis@gmail.com	987654321
7	Marta	marta@gmail.com	123456789
8	Juan	juan@gmail.com	987654321

GTE: DETALLES.

The image shows two side-by-side screenshots. The left screenshot is from VS Code, displaying a REST client request to `http://localhost:8080/api/clientes/1` with a `JSON` response. The right screenshot is from Postman, showing the same endpoint and a table of client details.

VS Code REST Client Response:

```
{
  "id_detalle": 1,
  "id_producto": 1,
  "id_producto": 1,
  "cantidad": 1,
  "subtotal": "25.00",
  "createdAt": "2025-07-23T21:08:16.151Z",
  "updatedAt": "2025-07-23T21:08:16.151Z"
}
```

Postman Response Table:

id_detalle	id_producto	id_producto	id_producto
1	1	1	1
2	1	1	1

GTE:PEDIDOS.

Default workspace | http://localhost:8080/api/pedidos/1

New HTTP Request

Automatic | Headers | Body | Script | Settings

Body data | raw | json | text | binary

url: http://localhost:8080/api/pedidos/1

Method: GET

Status: 200 OK Time: 11

JSON

```
{
  "id_pedido": 1,
  "id_cliente": 1,
  "fecha": "2025-07-27",
  "total": "1200.0",
  "createdAt": "2025-07-27T22:00:11.633Z",
  "updatedAt": "2025-07-27T22:00:11.633Z"
}
```

Executing (default): SELECT id_pedido, id_cliente, fecha, total, createdAt, updatedAt FROM pedidos AS pedido;

Tables

production

Tables

id_pedido	id_cliente	fecha	total	createdAt	updatedAt
1	1	2025-07-27	1200.0	2025-07-27T22:00:11.633Z	2025-07-27T22:00:11.633Z

GET: PRODUCTOS.

Default workspace | http://localhost:8080/api/productos/1

New HTTP Request

Automatic | Headers | Body | Script | Settings

Body data | raw | json | text | binary

url: http://localhost:8080/api/productos/1

Method: GET

Status: 200 OK Time: 11

JSON

```
{
  "id": 1,
  "nombre": "Leche",
  "precio": 1200.0,
  "stock": 10,
  "createdAt": "2025-07-27T22:00:11.633Z",
  "updatedAt": "2025-07-27T22:00:11.633Z"
}
```

Executing (default): SELECT id, nombre, precio, stock, createdAt, updatedAt FROM productos AS producto;

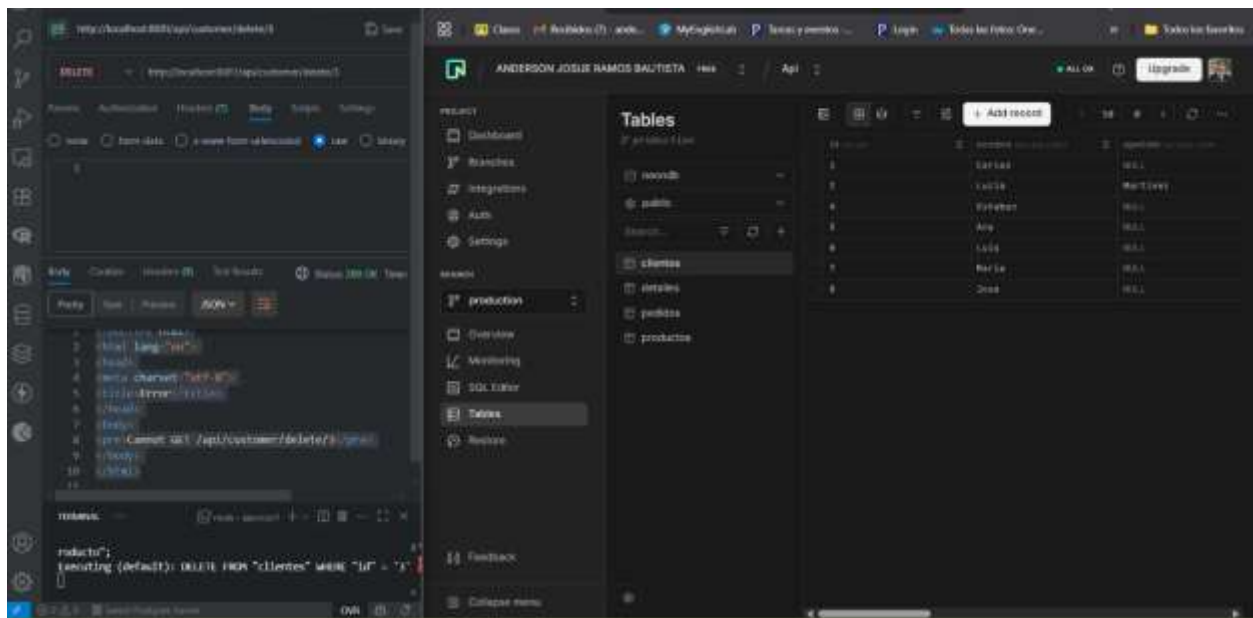
Tables

production

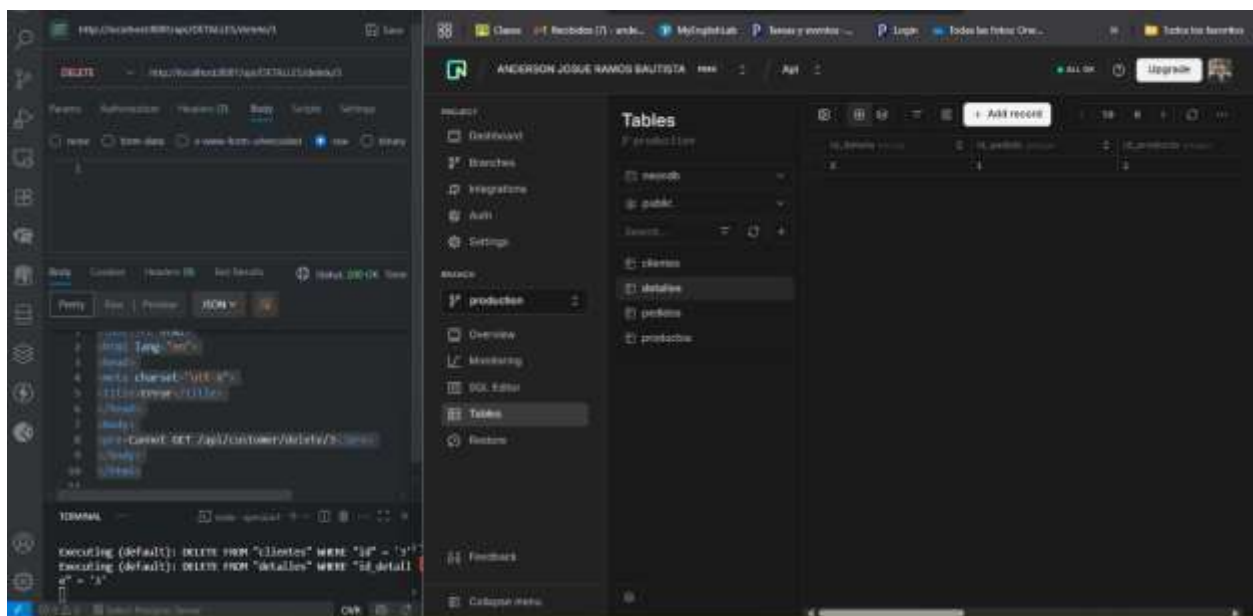
Tables

id	nombre	precio	stock	createdAt	updatedAt
1	Leche	1200.0	10	2025-07-27T22:00:11.633Z	2025-07-27T22:00:11.633Z

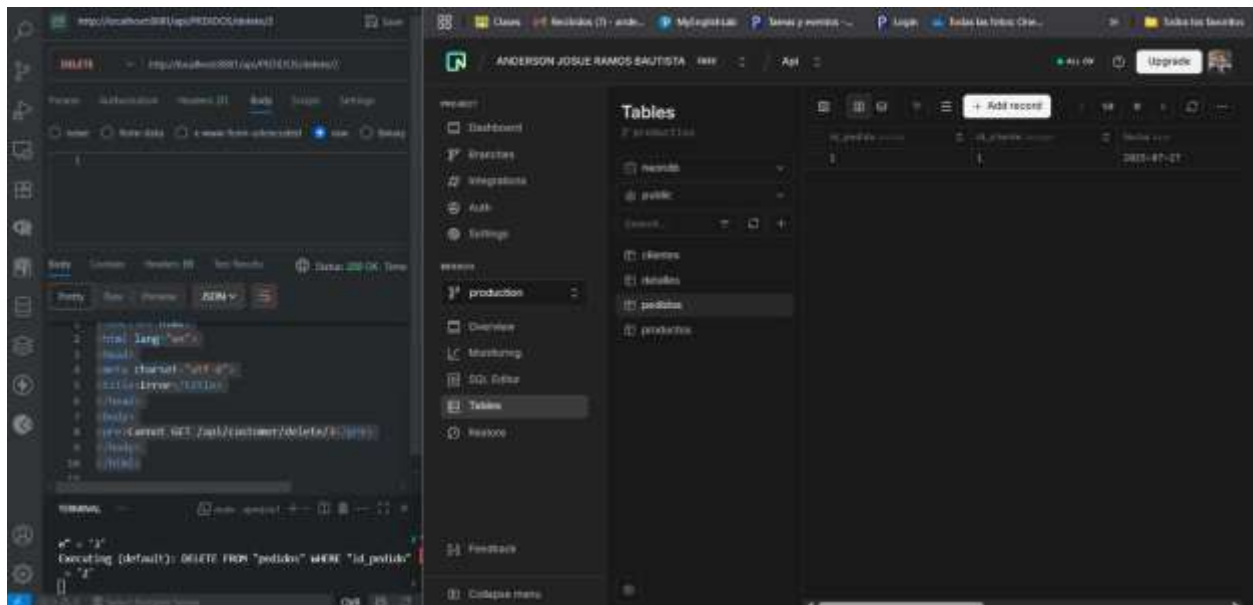
DELET: CLIENTE.



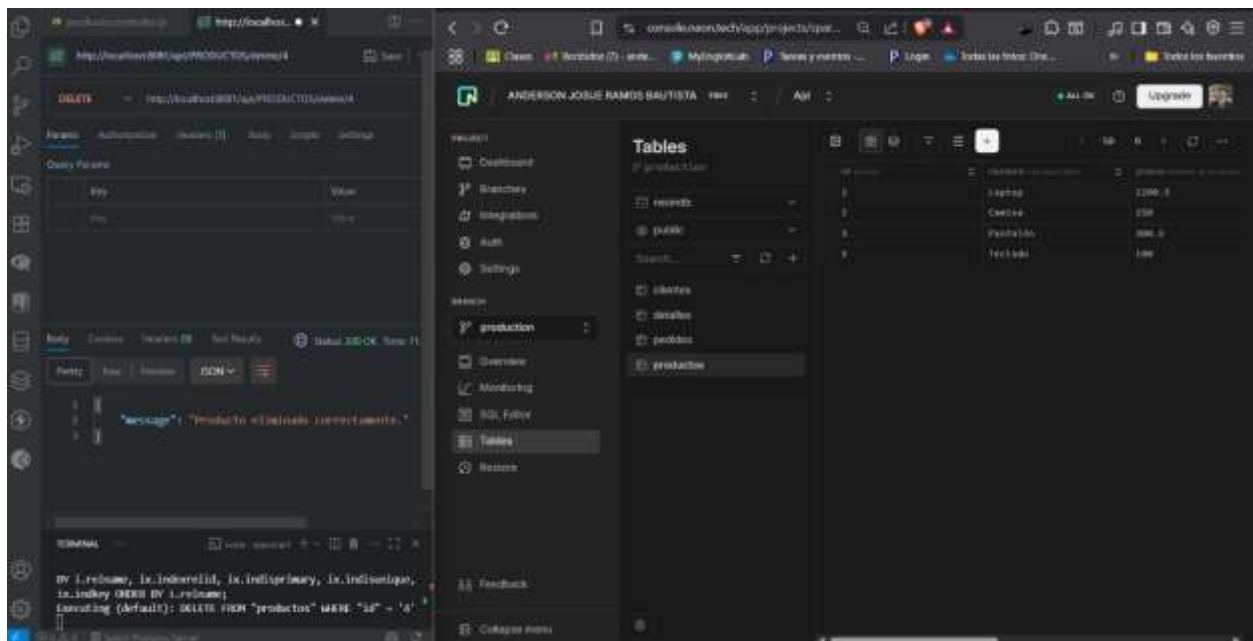
DELTE:DETALLES.



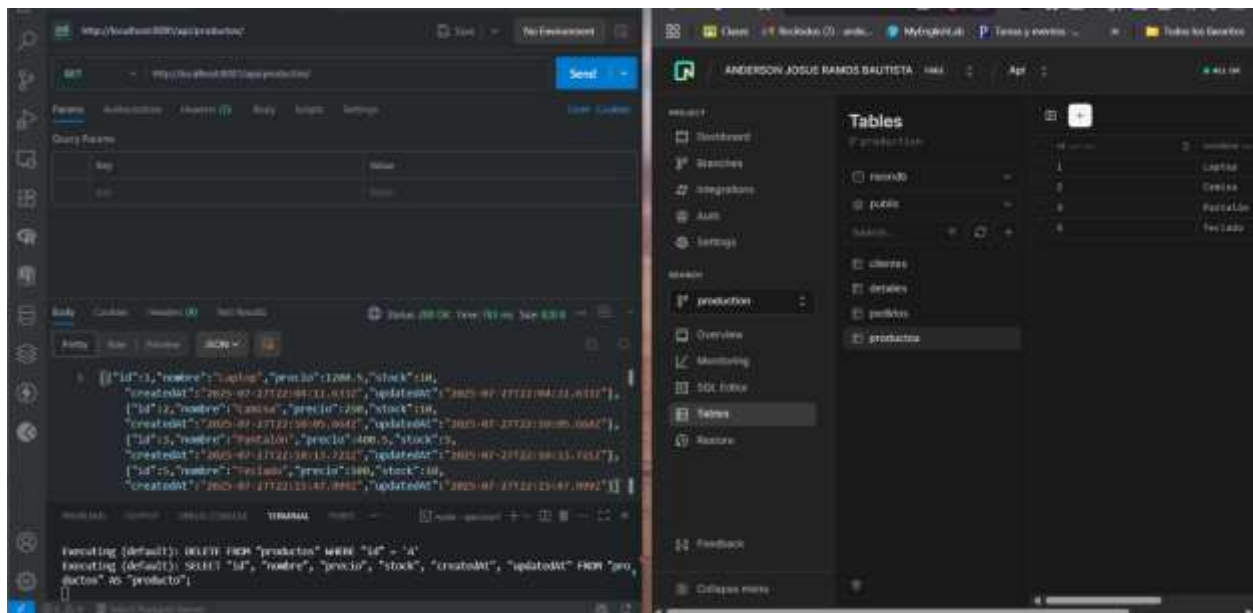
DELET: PEDIDOS.



DELTE: PRODUCTOS.



GETALL: PRODUCTOS



URL: <http://localhost:3000/api/productos/>

Query Results

key	value
id	value

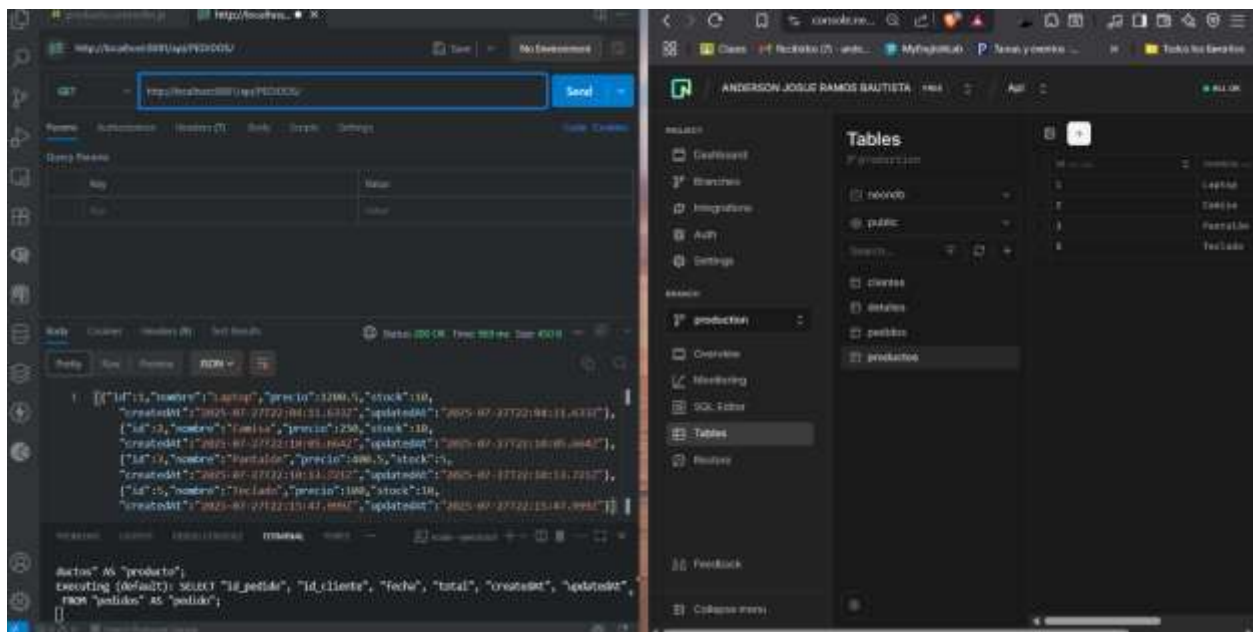
Body: `JSON` `Text Results`

Executing (default): DELETE FROM "productos" WHERE "id" = 'A';
 Executing (default): SELECT "id", "nombre", "precio", "stock", "createdAt", "updatedAt" FROM "productos" AS "producto";

Results:

```
[{"id":1,"nombre":"zapato","precio":1200.5,"stock":10,"createdAt":"2025-07-27T22:04:11.631Z","updatedAt":"2025-07-27T22:04:11.631Z"}, {"id":2,"nombre":"camisa","precio":1250,"stock":10,"createdAt":"2025-07-27T22:04:05.064Z","updatedAt":"2025-07-27T22:04:05.064Z"}, {"id":3,"nombre":"pantalón","precio":400.5,"stock":5,"createdAt":"2025-07-27T22:04:13.721Z","updatedAt":"2025-07-27T22:04:13.721Z"}, {"id":4,"nombre":"vestido","precio":100,"stock":10,"createdAt":"2025-07-27T22:15:47.099Z","updatedAt":"2025-07-27T22:15:47.099Z"}]
```

GETALL: PEDIDOS.



URL: <http://localhost:3000/api/pedidos/>

Query Results

key	value
id	value

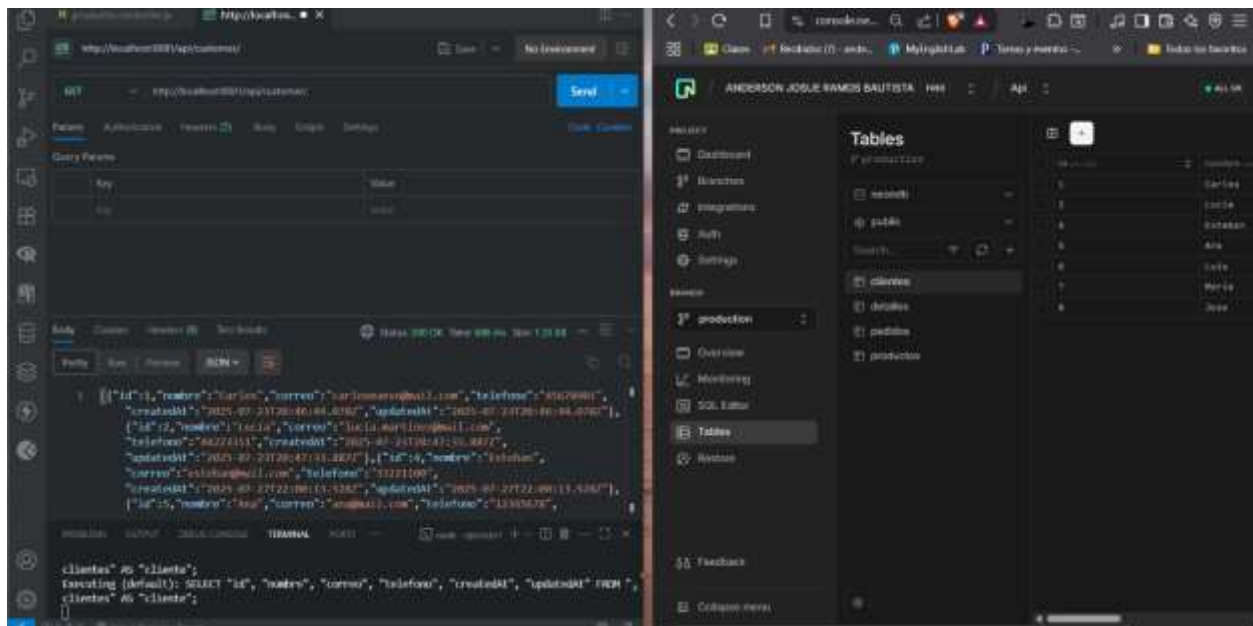
Body: `JSON` `Text Results`

Executing (default): DELETE FROM "pedidos" WHERE "id" = 'A';
 Executing (default): SELECT "id_pedido", "id_cliente", "fecha", "total", "createdAt", "updatedAt" FROM "pedidos" AS "pedido";

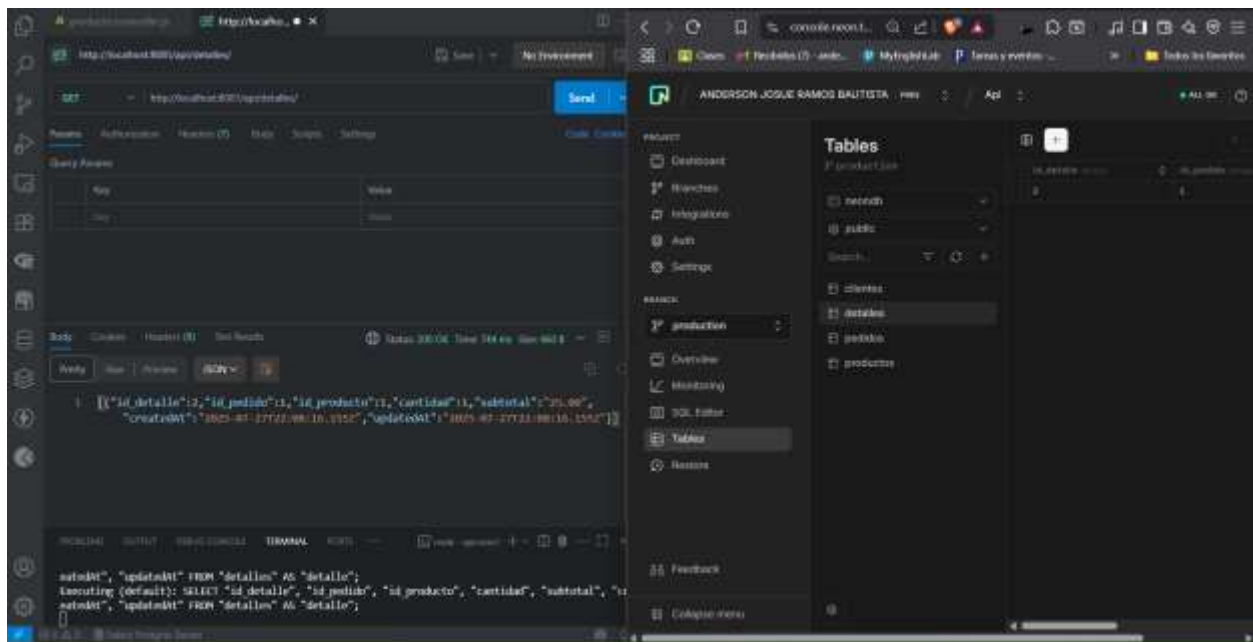
Results:

```
[{"id_pedido":1,"id_cliente":1,"fecha":"2025-07-27T22:04:11.631Z","total":1200.5,"createdAt":"2025-07-27T22:04:11.631Z","updatedAt":"2025-07-27T22:04:11.631Z"}, {"id_pedido":2,"id_cliente":2,"fecha":"2025-07-27T22:04:05.064Z","total":1250,"createdAt":"2025-07-27T22:04:05.064Z","updatedAt":"2025-07-27T22:04:05.064Z"}, {"id_pedido":3,"id_cliente":3,"fecha":"2025-07-27T22:04:13.721Z","total":400.5,"createdAt":"2025-07-27T22:04:13.721Z","updatedAt":"2025-07-27T22:04:13.721Z"}, {"id_pedido":4,"id_cliente":4,"fecha":"2025-07-27T22:15:47.099Z","total":100,"createdAt":"2025-07-27T22:15:47.099Z","updatedAt":"2025-07-27T22:15:47.099Z"}]
```

GETALL:CLIENTES.



GETALL: DETALLES.



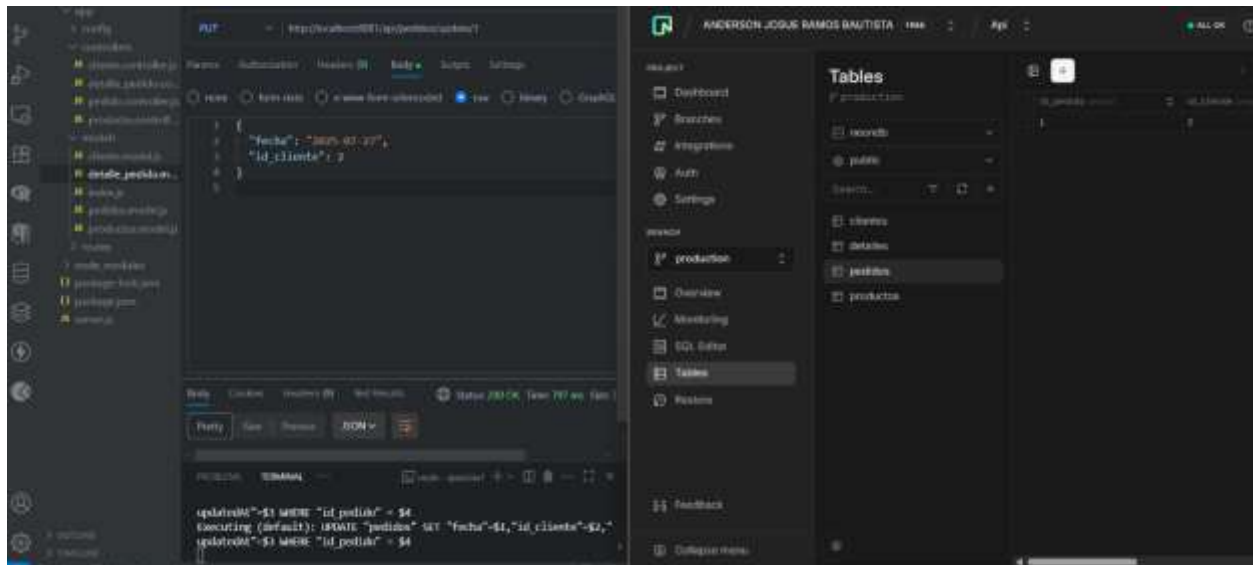
PUT: DETALLES.

The image shows a development environment with VS Code on the left and Postman on the right. In VS Code, a file named `detalle_pedido.js` contains a JSON object representing an order detail: `{ "id_pedido": 1, "id_producto": 1, "cantidad": 1, "subtotal": 150.00 }`. The Postman interface shows a PUT request to `http://localhost:8080/api/v1/orders/updates/1` with the same JSON body. The response status is 200 OK, and the body contains a success message: `{ "message": "Detalle de pedido actualizado correctamente." }`. The terminal at the bottom shows the command: `Executing (default): UPDATE "detalle" SET "id_pedido"=$1,"id_producto"=$2,"cantidad"=$3,"subtotal"=$4, "updatedAt"=$5 WHERE "id_detalle" = $6`.

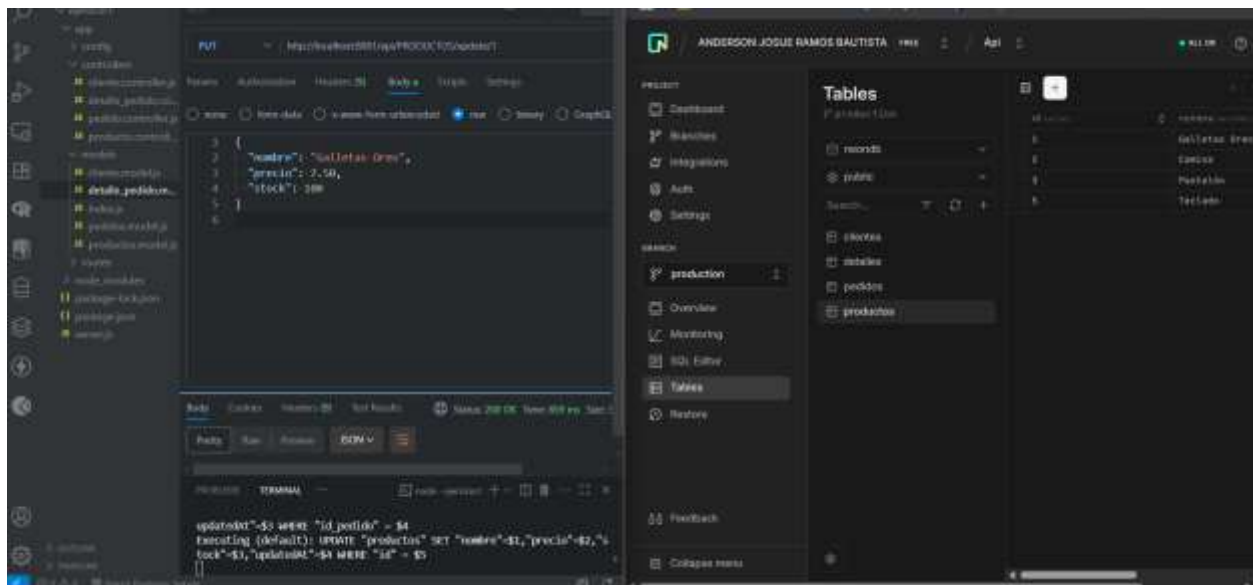
PUT: CLIENTE.

The image shows a development environment with VS Code on the left and Postman on the right. In VS Code, a file named `detalle_cliente.js` contains a JSON object representing a client: `{ "nombre": "Andres Jose Ramos", "correo": "andresramos@gmail.com", "telefono": "123456789", "status": true }`. The Postman interface shows a PUT request to `http://localhost:8080/api/v1/customers/updates/1` with the same JSON body. The response status is 200 OK, and the body contains a success message: `{ "message": "Cliente was updated successfully." }`. The terminal at the bottom shows the command: `Executing (default): UPDATE "clientes" SET "nombre"=$1,"correo"=$2,"telefono"=$3,"updatedAt"=$4 WHERE "id" = $5`.

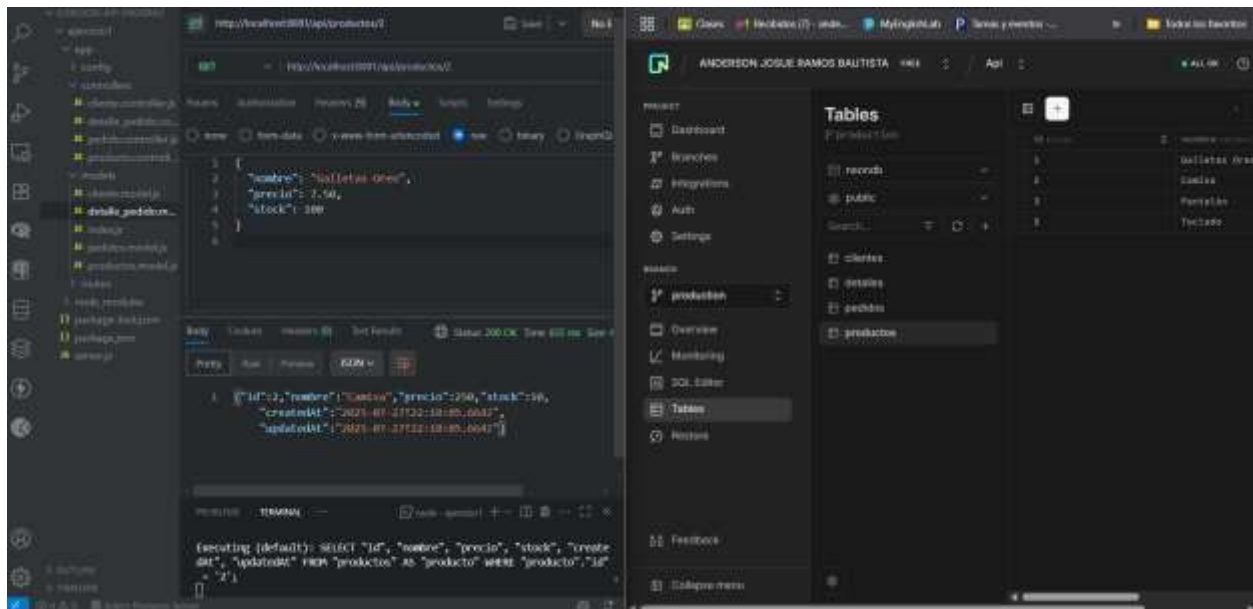
PUT: PEDIDOS



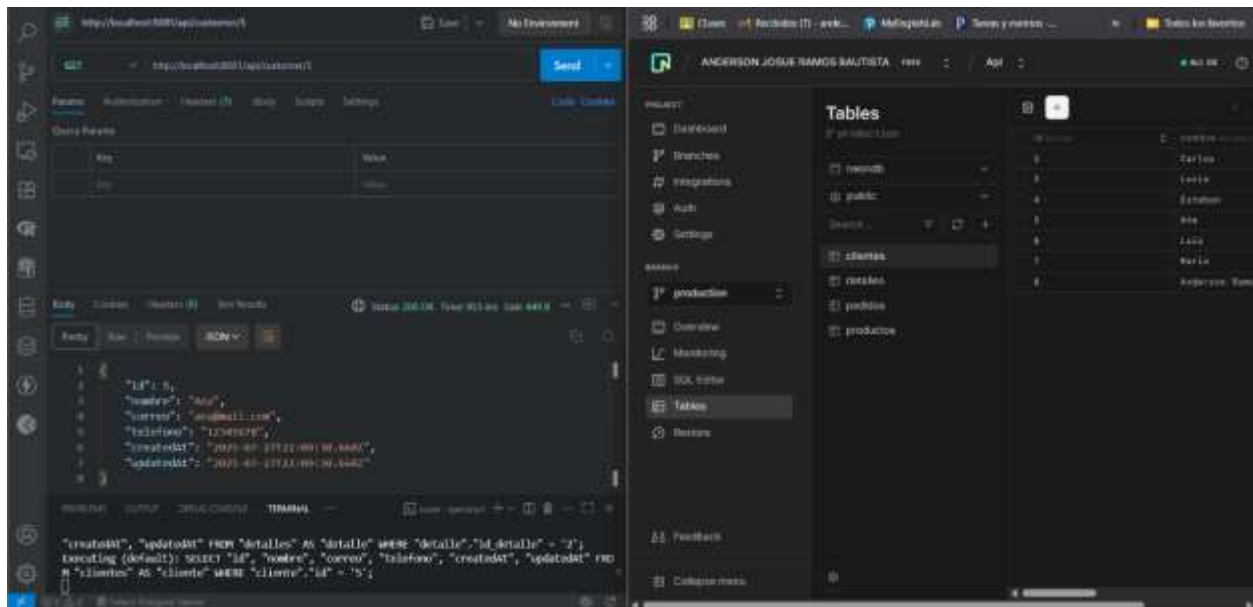
PUT: PRODUCTOS.



GETBYID: PRODUCTOS.



Getbyid: Clientes.



Getbyid: detalles:

GET <http://localhost:8080/api/detalles/1>

Query Params

Key	Value
id	1

Body

```
{
  "id_detalle": 1,
  "id_producto": 1,
  "id_cliente": 1,
  "cantidad": 1,
  "subtotal": "156.00",
  "created_at": "2025-07-27T22:06:14.191Z",
  "updated_at": "2025-07-27T22:06:24.242Z"
}
```

Status: 200 OK, Time: 67 ms, Size: 459 B

Executing (default): SELECT "id_detalle", "id_producto", "id_cliente", "cantidad", "subtotal", "created_at", "updated_at" FROM "detalles" AS "detalle" WHERE "detalle"."id_detalle" = '1';

Terminal

```
SELECT "id_detalle", "id_producto", "id_cliente", "cantidad", "subtotal", "created_at", "updated_at" FROM "detalles" AS "detalle" WHERE "detalle"."id_detalle" = '1';
```

Getbyid: pedidos.

GET <http://localhost:8080/api/pedidos/1>

Query Params

Key	Value
id	1

Body

```
{
  "id_pedido": 1,
  "id_cliente": 2,
  "fecha": "2025-07-27",
  "total": "156.00",
  "created_at": "2025-07-27T22:05:34.721Z",
  "updated_at": "2025-07-27T22:23:42.798Z"
}
```

Status: 200 OK, Time: 55 ms, Size: 448 B

Executing (default): SELECT "id_pedido", "id_cliente", "fecha", "total", "created_at", "updated_at" FROM "pedidos" AS "pedido" WHERE "pedido"."id_pedido" = '1';

Terminal

```
SELECT "id_pedido", "id_cliente", "fecha", "total", "created_at", "updated_at" FROM "pedidos" AS "pedido" WHERE "pedido"."id_pedido" = '1';
```