

UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE

Estudiante: Richard Alban

Materia: Aplicaciones Distribuidas

NRC: 2546

Gestión de Productos y Categorías

Ver Productos

The screenshot displays a REST client interface with a dark theme. At the top, a request bar shows the method 'GET', a dropdown arrow, the URL 'http://localhost:8001/producto', and a blue 'Send' button. Below this, a tabbed interface includes 'Query' (selected), 'Headers 2', 'Auth', 'Body', 'Tests', and 'Pre Run'. The 'Query' tab shows 'Query Parameters' with a table containing one row: 'parameter' and 'value'. To the right, the 'Response' tab is active, showing the status '200 OK', size '66 Bytes', and time '56 ms'. The response body is a JSON array with one object:

```
[
  {
    "id": 1,
    "nombre": "producto1",
    "descripcion": "asd",
    "precio": 10.00
  }
]
```


POST

http://localhost:8001/producto

Send

Query

Headers 2

Auth

Body 1

Tests

Pre Run

JSON

XML

Text

Form

Form-encode

GraphQL

Binary

JSON Content

Format

```
1  {
2    "nombre": "producto1",
3    "descripcion": "asd",
4    "precio": 10.00
5  }
```

Status: 201 Created

Size: 64 Bytes

Time: 41 ms

Response

Headers 5

Cookies

Results

Docs

{}

≡

```
1  {
2    "id": 2,
3    "nombre": "producto1",
4    "descripcion": "asd",
5    "precio": 10.00
6  }
```

Actualizar Producto

PUT

▼

http://localhost:8001/producto/1

Send

Query

Headers²

Auth

Body¹

Tests

Pre Run

JSON

XML

Text

Form

Form-encode

GraphQL

Binary

JSON Content

Format

1

{

2

"id": 1,

3

"nombre": "producto actualizado",

4

"descripcion": "descripcion actualizada",

5

"precio": 15.00

6

}

7

Status: 200 OK

Size: 95 Bytes

Time: 151 ms

Response

Headers⁵

Cookies

Results

Docs

{}

≡

1

{

2

"id": 1,

3

"nombre": "producto actualizado",

4

"descripcion": "descripcion actualizada",

5

"precio": 15.00

6

}

Eliminar Producto

DELETE

▼

http://localhost:8001/producto/1

Send

Query

Headers²

Auth

Body¹

Tests

Pre Run

JSON

XML

Text

Form

Form-encode

GraphQL

Binary

JSON Content

Format

1

{

2

"id": 1,

3

"nombre": "producto actualizado",

4

"descripcion": "descripcion actualizada",

5

"precio": 15.00

6

}

7

Status: 204 No Content

Size: 0 Bytes

Time: 573 ms

Response

Headers³

Cookies

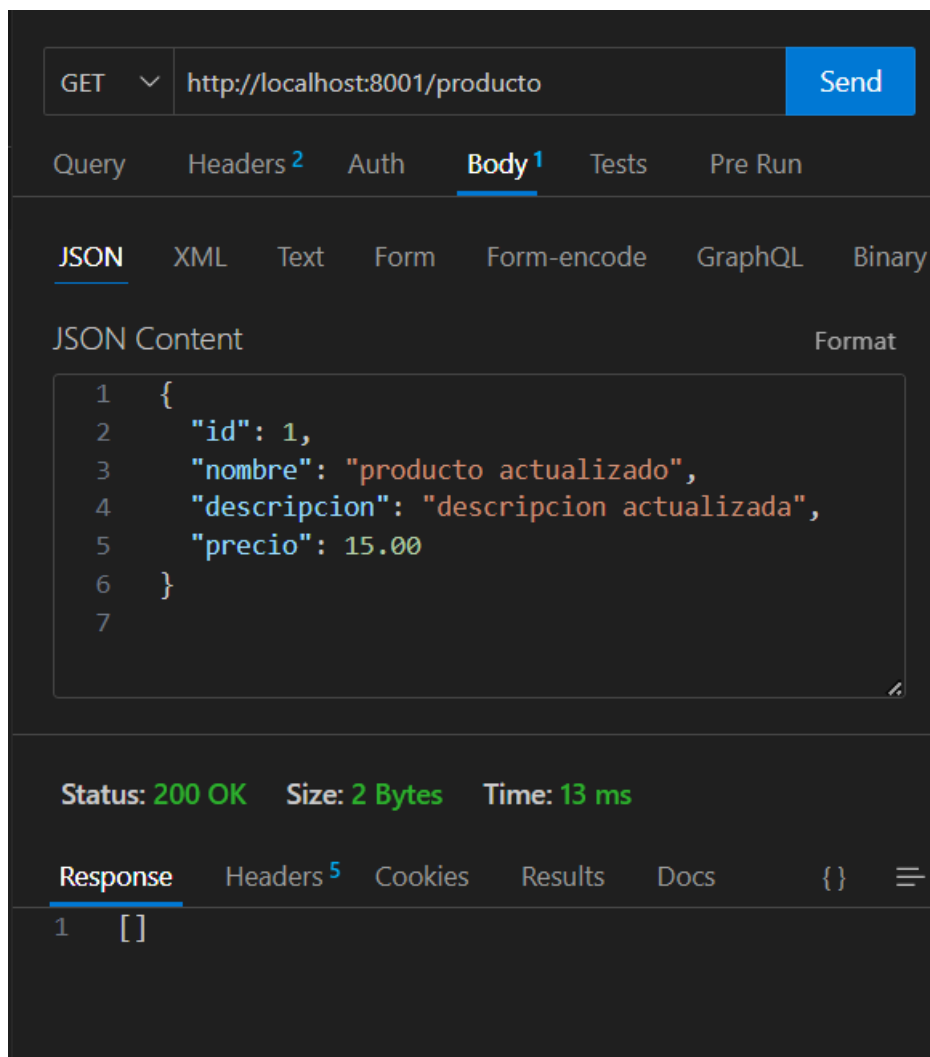
Results

Docs

{}

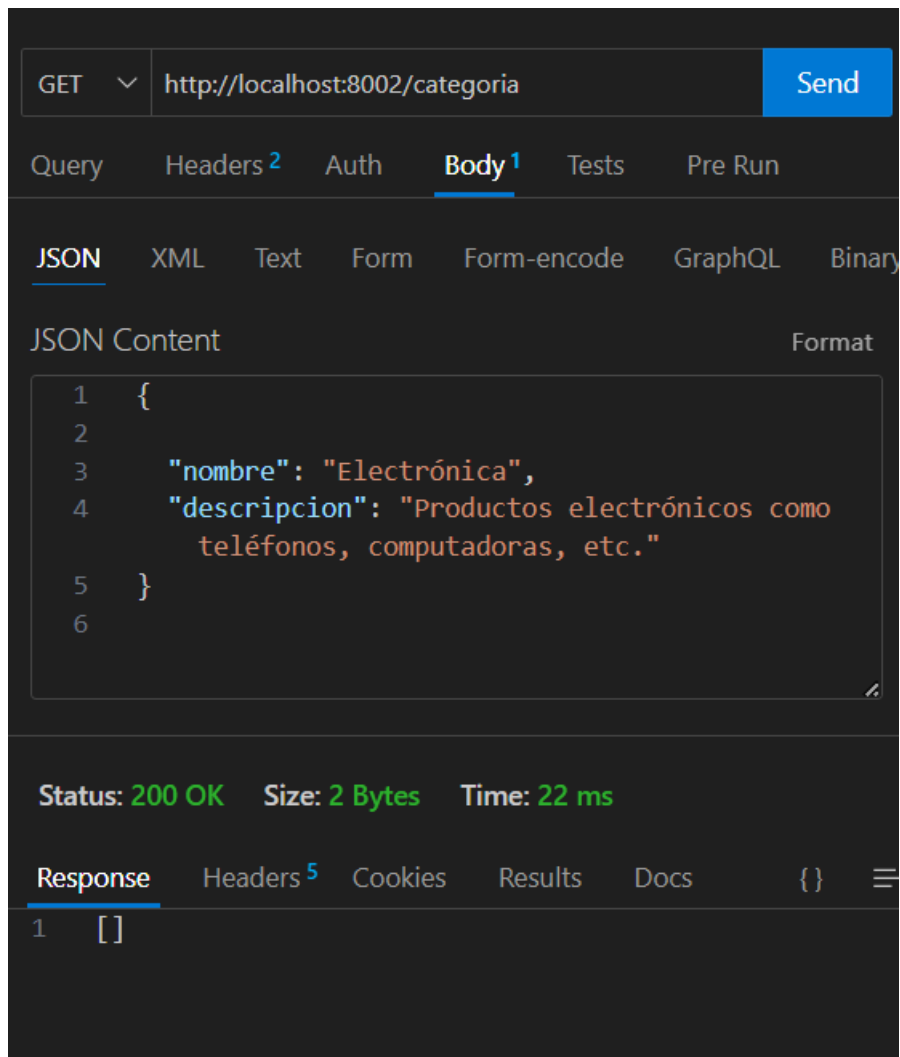
≡

1



Categoria

Ver Categoria



Crear Categoría

POST

http://localhost:8002/categoria

Send

Query

Headers²

Auth

Body¹

Tests

Pre Run

JSON

XML

Text

Form

Form-encode

GraphQL

Binary

JSON Content

Format

1

{

2

3

"nombre": "Electrónica",

4

"descripcion": "Productos electrónicos como

teléfonos, computadoras, etc."

5

}

6

Status: 201 Created

Size: 98 Bytes

Time: 214 ms

Response

Headers⁵

Cookies

Results

Docs

{}

≡

1

{

2

"id": 1,

3

"nombre": "Electrónica",

4

"created_at": "2025-01-27T15:55:12.532092",

5

"categoriaProductos": []

6

}

GET

http://localhost:8002/categoria

Send

Query

Headers 2

Auth

Body 1

Tests

Pre Run

JSON

XML

Text

Form

Form-encode

GraphQL

Binary

JSON Content

Format

```
1  {
2
3    "nombre": "Electrónica",
4    "descripcion": "Productos electrónicos como
5                  teléfonos, computadoras, etc."
6  }
```

Status: 200 OK

Size: 100 Bytes

Time: 31 ms

Response

Headers 5

Cookies

Results

Docs

{}

≡

```
1  [
2    {
3      "id": 1,
4      "nombre": "Electrónica",
5      "created_at": "2025-01-27T15:55:12.532092",
6      "categoriaProductos": []
7    }
8  ]
```

Editar Categoría

PUT

http://localhost:8002/categoria/1

Send

Query

Headers²

Auth

Body¹

Tests

Pre Run

JSON

XML

Text

Form

Form-encode

GraphQL

Binary

JSON Content

Format

1

{

2

3

"nombre": "ElectrónicaPrueba",

4

"descripcion": "Productos electrónicos como

teléfonos, computadoras, etc."

5

}

6

Status: 200 OK

Size: 104 Bytes

Time: 56 ms

Response

Headers⁵

Cookies

Results

Docs

{}

≡

1

{

2

"id": 1,

3

"nombre": "ElectrónicaPrueba",

4

"created_at": "2025-01-27T15:55:12.532092",

5

"categoriaProductos": []

6

}

GET

http://localhost:8002/categoria/1

Send

Query

Headers 2

Auth

Body 1

Tests

Pre Run

JSON

XML

Text

Form

Form-encode

GraphQL

Binary

JSON Content

Format

```
1  {
2
3    "nombre": "ElectrónicaPrueba",
4    "descripcion": "Productos electrónicos como
5                  teléfonos, computadoras, etc."
6  }
```

Status: 200 OK

Size: 104 Bytes

Time: 14 ms

Response

Headers 5

Cookies

Results

Docs

{}

≡

```
1  {
2    "id": 1,
3    "nombre": "ElectrónicaPrueba",
4    "created_at": "2025-01-27T15:55:12.532092",
5    "categoriaProductos": []
6  }
```

Eliminar Categoría

DELETE

▼

http://localhost:8002/categoria/1

Send

Query

Headers²

Auth

Body¹

Tests

Pre Run

JSON

XML

Text

Form

Form-encode

GraphQL

Binary

JSON Content

Format

1

{

2

3

"nombre": "ElectrónicaPrueba",

4

"descripcion": "Productos electrónicos como

teléfonos, computadoras, etc."

5

}

6

Status: 204 No Content

Size: 0 Bytes

Time: 603 ms

Response

Headers³

Cookies

Results

Docs

{}

≡

1

GET

http://localhost:8002/categoria

Send

Query

Headers²

Auth

Body¹

Tests

Pre Run

JSON

XML

Text

Form

Form-encode

GraphQL

Binary

JSON Content

Format

1

{

2

3

4

5

6

"nombre": "Electrónica",

"descripcion": "Productos electrónicos como

teléfonos, computadoras, etc."

}

Status: 200 OK

Size: 2 Bytes

Time: 22 ms

Response

Headers⁵

Cookies

Results

Docs

{ }

≡

1

[]

Agregar productos a categoría:

The screenshot shows a REST client interface with a dark theme. At the top, a POST request is configured to `http://localhost:8002/categoria/2/productos/2`. The 'Body' tab is selected, showing a JSON payload. Below the body, the status is `201 Created`, size is `53 Bytes`, and time is `41 ms`. The 'Response' tab is selected at the bottom, showing the JSON response.

POST `http://localhost:8002/categoria/2/productos/2` Send

Query Headers² Auth **Body¹** Tests Pre Run

JSON XML Text Form Form-encode GraphQL Binary

JSON Content Format

```
1 {
2
3   "nombre": "Electrónica",
4   "descripcion": "Productos electrónicos como
5     teléfonos, computadoras, etc."
6 }
```

Status: **201 Created** Size: **53 Bytes** Time: **41 ms**

Response Headers⁵ Cookies Results Docs {} ≡

```
1 {
2   "id": 1,
3   "productoId": 2,
4   "enrollmentDate": "2025-01-27"
5 }
```

Obtener todos los productos de una categoría:

The screenshot shows a REST client interface with the following components:

- Request Bar:** Method: GET, URL: `http://localhost:8002/categoria/2/productos`, and a blue **Send** button.
- Request Tabs:** Query, Headers ², Auth, **Body ¹**, Tests, Pre Run.
- Body Tab:** JSON (selected), XML, Text, Form, Form-encode, GraphQL, Binary.
- JSON Content:** A code editor showing the following JSON object:

```
1  {  
2  
3    "nombre": "Electrónica",  
4    "descripcion": "Productos electrónicos como  
                    teléfonos, computadoras, etc."  
5  }  
6
```
- Status Bar:** Status: 200 OK, Size: 3 Bytes, Time: 23 ms.
- Response Tabs:** **Response** (selected), Headers ⁵, Cookies, Results, Docs, {}, ≡.
- Response Content:** A code editor showing the following JSON array:

```
1  [  
2    2  
3  ]
```

Eliminar Producto de una Categoría:

DELETE

▼

http://localhost:8002/categoria/2/productos/2

Send

Query

Headers²

Auth

Body¹

Tests

Pre Run

JSON

XML

Text

Form

Form-encode

GraphQL

Binary

JSON Content

Format

1 {

2

3 "nombre": "Electrónica",

4 "descripcion": "Productos electrónicos como

5 teléfonos, computadoras, etc."

6 }

Status: 204 No Content

Size: 0 Bytes

Time: 28 ms

Response

Headers³

Cookies

Results

Docs

{}

≡

1

GET

http://localhost:8002/categoria/2/productos

Send

Query

Headers²

Auth

Body¹

Tests

Pre Run

JSON

XML

Text

Form

Form-encode

GraphQL

Binary

JSON Content

Format

1 {

2

3 "nombre": "Electrónica",

4 "descripcion": "Productos electrónicos como

5 teléfonos, computadoras, etc."

6 }

Status: 200 OK

Size: 2 Bytes

Time: 13 ms

Response

Headers⁵

Cookies

Results

Docs

{}

≡

1 []









Administration - Dashboard

SQL File 6*




producto

categorias

categoria_producto





Limit to 1000 rows






1 •

SELECT * FROM sisdb_examen.categoria_producto;

Result Grid

 Filter Rows:

Edit: 

Export/Import

	id	enrollment_date	producto_id	categoria_id
*	NULL	NULL	NULL	NULL

Result Grid

Form Editor

Field Types

categoria_producto3 x

Apply

Revert

Output

Action Output

Administration - Dashboard SQL File 6* producto categorias categoria_pro

Limit to 1000 rows

```
1 • SELECT * FROM sisdb_examen.categoria_producto;
```

Result Grid

	id	enrollment_date	producto_id	categoria_id
▶	1	2025-01-27	2	2
*	NULL	NULL	NULL	NULL

Result Grid

Form Editor

Field Types

categoria_producto2 x Apply Revert

Enlace al git:

<https://github.com/RichardAlban/ExamenDistribuidas.git>