

Evaluación

Requerimiento (incluye validaciones, regresar código de error y que funcione correctamente).	Valor	Realizado (Si/No)
1. GET /products - Filtro por parámetros de búsqueda.	15	
2. POST /products/cart	15	
3. GET /products/:id	10	
4. Middleware de autenticar	10	
5. POST /admin/products	10	
6. PUT /admin/products/:id	15	
7. DELETE /admin/products:id	10	
8. GET /, /home, /shopping_cart	10	
9. Código estructurado y modularizado - Archivo de router global - Archivo de router para productos - Archivo de router para endpoints de administrador de productos	10	
10. Manejo de archivos	10	

Conclusiones

La práctica se trató de implementar una API RESTful para un sistema e-commerce, usando rutas, middlewares, validaciones, persistencia en archivos y demás, la modulación de controladores y rutas fue lo que me facilitó y ayudo a finalmente lograr mantener la estabilidad del código. Esta práctica me hizo ver la importancia del manejo correcto de estados HTTP, autenticaciones y actualización de datos en aplicaciones web.

Algunos de los retos fueron el manejo de errores (de manera correcta claro) y para mí el mayor encontrar la mejor estructura para ordenar el proyecto para poder mezclar el front-end y back-end para que todo funcionara en base a los requerimientos. Finalmente, gracias a esta práctica consigo ver un flujo de trabajo lo más similar al de proyectos profesionales en el desarrollo de APIs modernas lo cual agradezco ya que me gusta ver las cosas que se llega a ser capaz en comparación a algo cercano a lo profesional.

Repositorio: <https://github.com/Angel751658/DASW.git>

Evidencias

Inicio del servidor

```
C:\Users\Hp\OneDrive\Pro\html\github\DASW\Practica 3>nodemon server.js
[nodemon] 3.1.9
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node server.js`
Server running on http://localhost:3000
█
```

Listado de productos GET /products

http://localhost:3000/products

GET http://localhost:3000/products

Params Authorization Headers (6) Body Scripts Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

Body Cookies Headers (7) Test Results

200 OK • 135 ms • 235 B

JSON Preview Visualize

```
1 []
```

Creación de productos POST /admin/products

http://localhost:3000/admin/products

POST http://localhost:3000/admin/products

Params Authorization Headers (10) Body Scripts Settings Cookies

Headers 8 hidden

Key	Value	Description	Bulk Edit	Presets
<input checked="" type="checkbox"/> x-auth	admin			
<input checked="" type="checkbox"/> Content-Type	application/json			

Body Cookies Headers (7) Test Results

201 Created • 36 ms • 286 B

JSON Preview Visualize

```
1 {
2   "message": "Producto creado: Producto Test 1"
3 }
```

Validación de productos creados

The image displays three sequential screenshots of a REST client interface, showing the results of GET requests to the endpoint `http://localhost:3000/products`. Each screenshot shows a `200 OK` status, indicating successful retrieval of data.

First Screenshot: The response body is a JSON array containing one product object. The status bar shows `200 OK`, `108 ms`, and `944 B`.

```
[{"uuid": "a91ead39-052b-4e15-98c9-4707442d1847", "imageUrl": "https://via.placeholder.com/150", "title": "Producto Test 1", "description": "Descripción de prueba del producto 1", "unit": "pieza", "category": "Test", "pricePerUnit": 100, "stock": 10}]
```

Second Screenshot: The response body is a JSON array containing two product objects. The status bar shows `200 OK`, `34 ms`, and `3.68 KB`.

```
[{"uuid": "49156122-6155-4309-8168-a4be85e800db", "imageUrl": "https://m.media-amazon.com/images/I/712YYjIJ9sL._AC_UF894,1000_QL80_.jpg", "title": "Absolute Carnage", "description": "Comic Absolute Carnage.", "unit": "pieza", "category": "Marvel", "pricePerUnit": 300, "stock": 10}, {"uuid": "dde651f6-43a1-430a-932a-ea8f6061fbc0", "imageUrl": "https://static1.srcdn.com/wordpress/wp-content/uploads/2023/05/crisis-on-infinite-earths.jpg?q=49", "title": "Crisis on Infinite Earths", "description": "Comic Crisis on Infinite Earths de DC.", "unit": "pieza", "category": "DC", "pricePerUnit": 400, "stock": 10}]
```

Third Screenshot: The response body is a JSON array containing two product objects. The status bar shows `200 OK`, `34 ms`, and `3.68 KB`.

```
[{"uuid": "fcf25c36-8876-419b-9081-0377b4d8ed8b", "imageUrl": "https://static1.srcdn.com/wordpress/wp-content/uploads/sharedimages/2024/07/batman-under-the-red-hood.jpg", "title": "Batman: Under the Red Hood", "description": "Comic Batman: Under the Red Hood.", "unit": "pieza", "category": "DC", "pricePerUnit": 300, "stock": 10}, {"uuid": "9f6b34e1-deba-4b44-a303-4bbb776863e0", "imageUrl": "https://gonkbok.com/_next/image?url=https%3A%2F%2F6fmx7z4dqmnl.cloudfront.net%2Fspider-man-krav", "title": "Spider-man: Kraven's Last Hunt", "description": "Comic Kraven's Last Hunt.", "unit": "pieza", "category": "Marvel", "pricePerUnit": 315, "stock": 10}]
```

Filtro de productos GET /products?query=Marvel:, /products?query=DC:Batman

The image shows two screenshots of a REST client interface, likely Postman, demonstrating API calls to a local server at `http://localhost:3000`.

Top Screenshot: GET /products?query=Marvel:

- Method:** GET
- URL:** `http://localhost:3000/products?query=Marvel:`
- Status:** 200 OK
- Response Body (JSON):**

```
[
  {
    "uuid": "49156122-6155-4309-8168-a4be85e800db",
    "imageUrl": "https://m.media-amazon.com/images/I/712YYjIJ9sL._AC_UF894,1000_QL80_.jpg",
    "title": "Absolute Carnage",
    "description": "Comic Absolute Carnage.",
    "unit": "pieza",
    "category": "Marvel",
    "pricePerUnit": 300,
    "stock": 10
  },
  {
    "uuid": "d09457cd-4cb2-4fde-9d71-e38b613cf6bf",
    "imageUrl": "https://gonkbonk.com/_next/image?url=https%3A%2F%2Fd6fmx7z4dqm1.cloudfront.net%2Finfinity-gauntl",
    "title": "The Infinity Gauntlet",
    "description": "Comic The Infinity Gauntlet.",
    "unit": "pieza",
    "category": "Marvel",
    "pricePerUnit": 450,
    "stock": 10
  }
]
```

Bottom Screenshot: GET /products?query=DC:Batman

- Method:** GET
- URL:** `http://localhost:3000/products?query=DC:Batman`
- Status:** 200 OK
- Response Body (JSON):**

```
[
  {
    "uuid": "365a2950-bda0-4735-a416-a47988e0992e",
    "imageUrl": "https://images-na.ssl-images-amazon.com/images/S/compressed.photo.goodreads.com/books/1346331835i",
    "title": "Batman: The Killing Joke",
    "description": "Comic Batman: The Killing Joke.",
    "unit": "pieza",
    "category": "DC",
    "pricePerUnit": 350,
    "stock": 10
  },
  {
    "uuid": "fcf25c36-8876-419b-9081-0377b4d8ed8b",
    "imageUrl": "https://static1.srcdn.com/wordpress/wp-content/uploads/sharedimages/2024/07/batman-under-the-red-",
    "title": "Batman: Under the Red Hood",
    "description": "Comic Batman: Under the Red Hood.",
    "unit": "pieza",
    "category": "DC",
    "pricePerUnit": 300,
    "stock": 10
  }
]
```

Buscar producto específico GET /products/:id

The screenshot shows a REST client interface with a GET request to `http://localhost:3000/products/53914de5-f836-4368-b355-6a7fd7f08cc9`. The response is a 200 OK status with a JSON body containing product details.

```
{
  "uuid": "53914de5-f836-4368-b355-6a7fd7f08cc9",
  "imageUrl": "https://cdn.marvel.com/u/prod/marvel/i/mg/e/f0/511307b2f1200/clean.jpg",
  "title": "Civil War",
  "description": "Conflicto entre héroes de Marvel.",
  "unit": "pieza",
  "category": "Marvel",
  "pricePerUnit": 400,
  "stock": 10
}
```

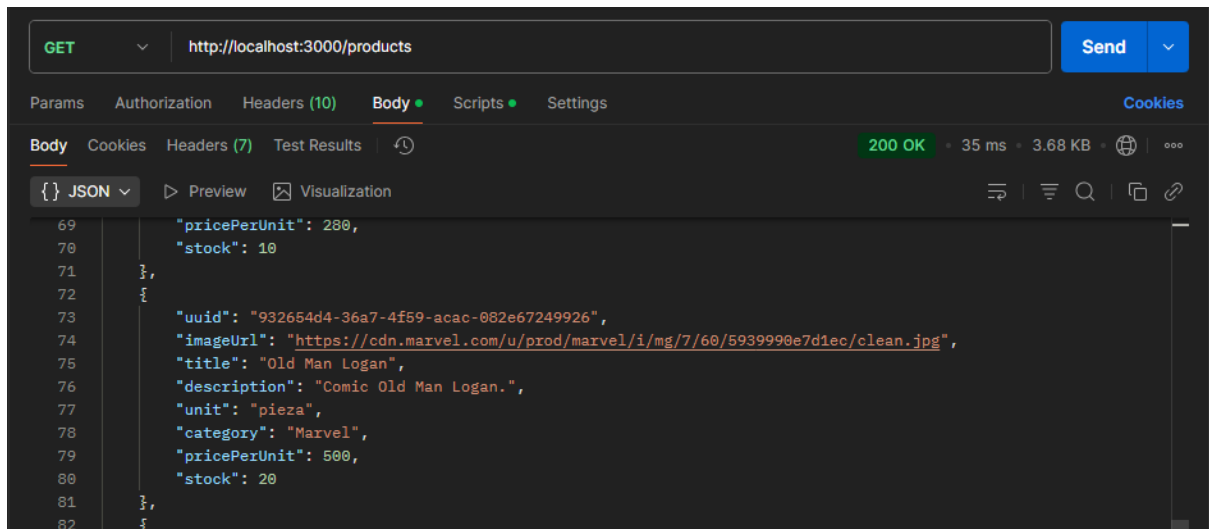
The second screenshot shows a GET request to `http://localhost:3000/products/5` resulting in a 404 Not Found status. The response body contains an error message: `"error": "Producto no encontrado"`.

Actualizar producto PUT /admin/products/:id

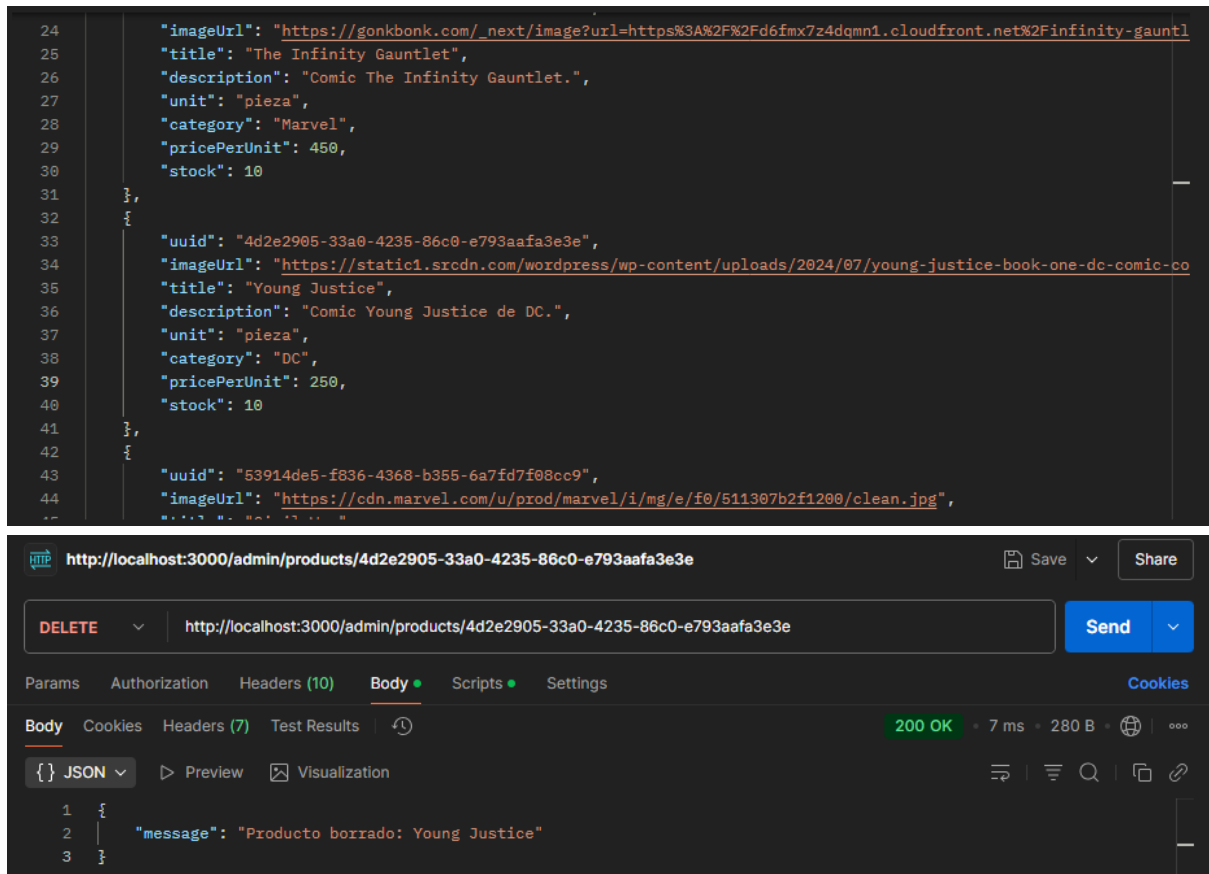
The screenshot shows a REST client interface with a PUT request to `http://localhost:3000/admin/products/932654d4-36a7-4f59-acac-082e67249926`. The request body is a JSON object with updated product information.

```
{
  "pricePerUnit": 500,
  "stock": 20
}
```

The response is a 200 OK status with a JSON body containing a success message: `"message": "Producto actualizado: Old Man Logan"`.



Borrar producto DELETE /admin/products/:id



```

22     {
23       "uuid": "d09457cd-4cb2-4fde-9d71-e38b613cf6bf",
24       "imageUrl": "https://gonkbonk.com/_next/image?url=https%3A%2F%2F6fmx7z4dqm1.cloudfront.net%2Finfinity-gauntl
25       "title": "The Infinity Gauntlet",
26       "description": "Comic The Infinity Gauntlet.",
27       "unit": "pieza",
28       "category": "Marvel",
29       "pricePerUnit": 450,
30       "stock": 10
31     },
32   ],
33   {
34     "uuid": "53914de5-f836-4368-b355-6a7fd7f08cc9",
35     "imageUrl": "https://cdn.marvel.com/u/prod/marvel/i/mg/e/f0/511307b2f1200/clean.jpg",
36     "title": "Civil War",
37     "description": "Conflicto entre héroes de Marvel.",
38     "unit": "pieza",
39     "category": "Marvel",
40     "pricePerUnit": 400,
41     "stock": 10
42   }
43 ]

```

Operaciones de carrito `POST /products/cart`

HTTP `http://localhost:3000/products/cart` Save Share

POST `http://localhost:3000/products/cart` Send

Params Authorization Headers (10) Body Scripts Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON Beautify

```

1 [
2   { "productUuid": "49156122-6155-4309-8168-a4be85e800db", "amount": 2 },
3   { "productUuid": "365a2950-bda0-4735-a416-a47988e0992e", "amount": 1 },
4   { "productUuid": "9f6b34e1-deba-4b44-a303-4bbb776863e0", "amount": 1 },
5   { "productUuid": "fcf25c36-8876-419b-9081-0377b4d8ed8b", "amount": 3 },
6   { "productUuid": "932654d4-36a7-4f59-acac-082e67249926", "amount": 1 }
7 ]
8

```

HTTP `http://localhost:3000/products/cart` Save Share

POST `http://localhost:3000/products/cart` Send

Params Authorization Headers (10) Body Scripts Settings Cookies

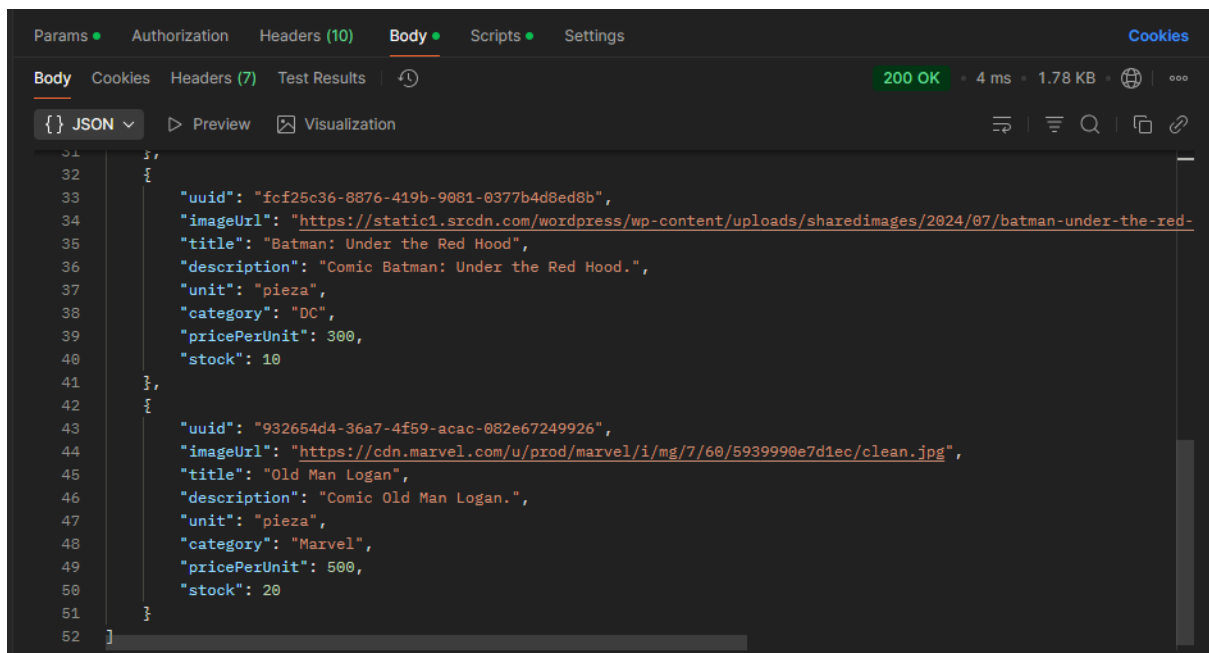
Body Cookies Headers (7) Test Results 200 OK • 4 ms • 1.78 KB

☒ JSON Preview Visualization

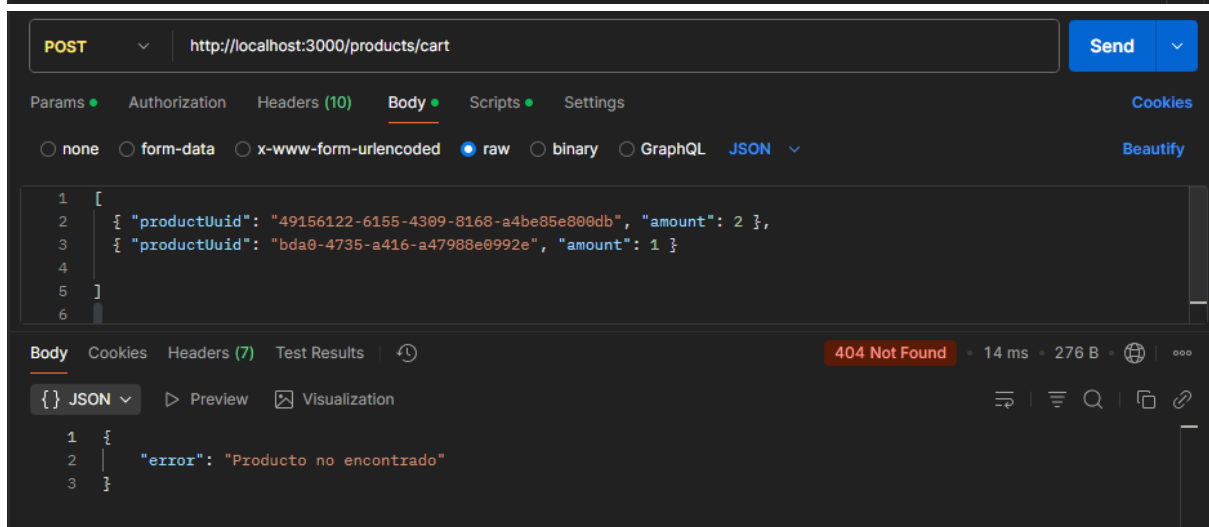
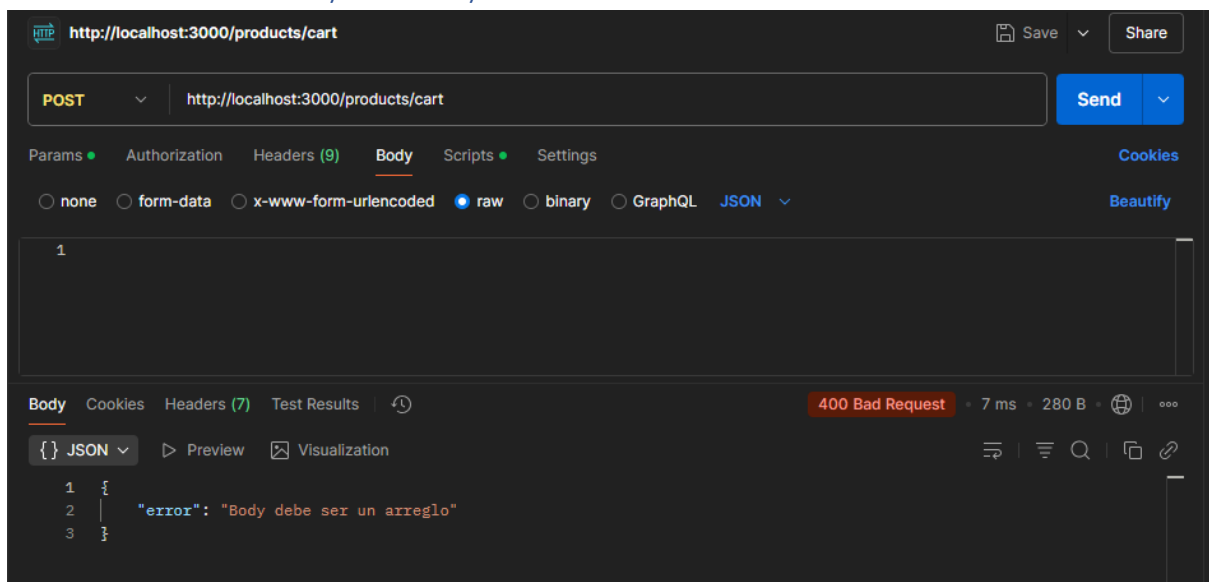
```

1 [
2   {
3     "uuid": "49156122-6155-4309-8168-a4be85e800db",
4     "imageUrl": "https://m.media-amazon.com/images/I/712YYjIJ9sL._AC_UF894,1000_QL80_.jpg",
5     "title": "Absolute Carnage",
6     "description": "Comic Absolute Carnage.",
7     "unit": "pieza",
8     "category": "Marvel",
9     "pricePerUnit": 300,
10    "stock": 10
11  },
12  {
13    "uuid": "365a2950-bda0-4735-a416-a47988e0992e",
14    "imageUrl": "https://images-na.ssl-images-amazon.com/images/S/compressed.photo.goodreads.com/books/1346331835i
15    "title": "Batman: The Killing Joke",
16    "description": "Comic Batman: The Killing Joke.",
17    "unit": "pieza",
18    "category": "DC",
19    "pricePerUnit": 350,
20    "stock": 10
21  }
22 ]

```

Error 400 al enviar body inválido y 404 al usar UUID inexistente.



Middleware de autenticación

http://localhost:3000/admin/products/365a2950-bda0-4735-a416-a47988e0992e

PUT

http://localhost:3000/admin/products/365a2950-bda0-4735-a416-a47988e0992e

Send

Params

Authorization

Headers (10)

Body

Scripts

Settings

Cookies

Headers

8 hidden

	Key	Value	Description	***	Bulk Edit	Presets
<input type="checkbox"/>	x-auth	admin				
<input checked="" type="checkbox"/>	Content-Type	application/json				
	Key	Value	Description			

Body

Cookies

Headers (7)

Test Results

403 Forbidden

11 ms

321 B

JSON

Preview

Visualization

1 {

2 "error": "Acceso no autorizado, no se cuenta con privilegios de administrador"

3 }

Páginas estáticas

Página de inicio

localhost:3000

Navbar

Home

About Us

Products

Search products...

Search

Login

Página de inicio

Página de Productos

localhost:3000/shopping_cart

Navbar

Home

About Us

Products

Search products...

Search

Login

Absolute Carnage

Quantity: 1

Price: 300.00 MXN

Spider-man: Kraven's Last Hunt

Total purchase:

Absolute Carnage: 1 x 300.00 MXN

Kraven's Last Hunt: 1 x 315.00 MXN

Under the reed hood: 1 x 300.00 MXN

The Killing Joke: 1 x 350.00 MXN

Shipping cost: 70.00 MXN

Total: 1335.00 MXN

Pay

Cancel