

Academlo Meals Evidences

Users

Create user evidence

The screenshot displays a REST client interface with a sidebar on the left and a main workspace on the right. The sidebar lists two API collections: 'Academlo_Meals' and 'Motorcycle_Api'. Under 'Academlo_Meals', the 'users' folder is expanded, showing several endpoints. The 'POST {{Host}}/users/signup' endpoint is selected. The main workspace shows the details of this request. The method is 'POST' and the URL is '{{Host}}/users/signup'. The 'Body' tab is active, showing a JSON payload with the following structure:

```
1 {
2   ... "name": "Usuario de Prueba",
3   ... "email": "usuarioTest@correo.com",
4   ... "password": "1234567890",
5   ... "role": "admin"
6 }
```

Below the request details, the response is shown. The status is '201 Created', the time is '1469 ms', and the size is '495 B'. The response body is displayed in the 'Pretty' tab, showing a JSON object with a 'token' and a 'user' object:

```
1 {"token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MCwiaWF0IjoxNzA0MjI5Mjg4LjE3MDQyMzY0ODh9.pq73RPMNxtzpr8E358W9SzQ1TxuHC7rTntGyMUjmy88",
  "user": {"id": 8, "name": "Usuario de Prueba", "email": "usuarioTest@correo.com", "role": "admin"}}
```

User login evidence

The screenshot displays a REST client interface with a sidebar on the left and a main workspace on the right. The sidebar lists API collections: 'Academlo_Meals' (expanded) and 'Motorcycle_Api'. Under 'Academlo_Meals', the 'users' folder is expanded, showing endpoints like 'POST {{Host}}/users/signup', 'POST {{Host}}/users/login' (selected), 'PATCH {{Host}}/users/1', 'DEL {{Host}}/users/1', 'GET {{Host}}/users/orders', and 'GET {{Host}}/users/orders/1'. Other folders include 'restaurants', 'meals', 'reviews', and 'orders'. The main workspace shows the details of the selected 'POST {{Host}}/users/login' endpoint. The 'Body' tab is active, displaying a JSON payload:

```
{  "email": "usuarioTest@correo.com",  "password": "1234567890"}
```

. The 'Response' section is empty, featuring a cartoon astronaut and the text 'Click Send to get a response'.

Academlo_Meals / users / `{{Host}}/users/login`

POST `{{Host}}/users/login` Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "email": "usuarioTest@correo.com",
3   "password": "1234567890"
4 }
```

Response

Click Send to get a response

Update user evidence

The screenshot displays a REST client interface with a sidebar on the left containing a project tree. The tree includes folders for 'Academlo_Meals' and 'Motorcycle_Api'. Under 'Academlo_Meals', there are subfolders for 'users', 'restaurants', 'meals', 'reviews', and 'orders'. The 'users' folder is expanded, showing a list of endpoints: 'POST {{Host}}/users/signup', 'POST {{Host}}/users/login', 'PATCH {{Host}}/users/1', 'DEL {{Host}}/users/1', 'GET {{Host}}/users/orders', and 'GET {{Host}}/users/orders/1'. The 'PATCH {{Host}}/users/1' endpoint is selected.

The main panel shows the details of the selected endpoint. The URL is 'PATCH {{Host}}/users/8'. The 'Body' tab is active, displaying a JSON payload:

```
1 {
2   ... "name": "El Prueba API's",
3   ... "email": "usuarioTesteador@correo.com"
4 }
```

Below the body, the 'Status' is '200 OK', 'Time' is '569 ms', and 'Size' is '571 B'. The 'Body' tab is also active, showing a 'Pretty' view of the response JSON:

```
1 {
2   "message": "method patch - update: from users",
3   "updateUser": {
4     "id": 8,
5     "name": "El Prueba API's",
6     "email": "usuarioTesteador@correo.com",
7     "password": "$2b$14$S4tK26XuZM0/oqunNfBay.CkPVxZi14hajHFYFznZPlHMRJqw/Eka",
8     "status": "available",
9     "role": "admin",
10    "createdAt": "2024-01-02T21:01:27.137Z",
11    "updatedAt": "2024-01-02T21:04:47.140Z"
12  }
13 }
```

Update user protection evidence

The screenshot shows a REST client interface with a sidebar on the left listing API endpoints under 'Academlo_Meals' and 'Motorcycle_Api'. The main panel displays a PATCH request to `{{Host}}/users/3` with a JSON body. The response shows a 401 Unauthorized status with an error message.

Request:

- Method: PATCH
- URL: `{{Host}}/users/3`
- Body (JSON):

```
1 {
2   ... "name": "El Prueba API's",
3   ... "email": "usuarioTesteador@correo.com"
4 }
```

Response:

- Status: 401 Unauthorized
- Time: 610 ms
- Size: 973 B
- Body (JSON):

```
1 {
2   "status": "error",
3   "message": "You do not own this account",
4   "stack": "Error: You do not own this account\n    at protectAccountOwner (C:\\Academlo\\Node.js\\API-Academlo-Meals\\src\\modules\\users\\users.\n      middleware.js:23:19)\n    at Layer.handle [as handle_request] (C:\\Academlo\\Node.\n      js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\layer.js:95:5)\n    at next (C:\\Academlo\\Node.\n      js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\route.js:144:13)\n    at C:\\Academlo\\Node.\n      js\\API-Academlo-Meals\\src\\modules\\users\\users.middleware.js:15:5\n    at process.processTicksAndRejections (node:internal/process/\n      task_queues:95:5)",
5   "err": {\n6     "statusCode": 401,\n7     "status": "error",\n8     "isOperational": true\n9   }\n10 }
```

Delete user evidence

The screenshot displays a REST client interface with a dark theme. On the left sidebar, the 'Academlo_Meals' collection is expanded, showing a tree structure with folders for 'users', 'restaurants', 'meals', 'reviews', and 'orders'. The 'users' folder is selected, and the 'DELETE {{Host}}/users/1' endpoint is highlighted.

The main panel shows the configuration for the selected endpoint: `DELETE {{Host}}/users/8`. The 'Authorization' tab is active, showing a 'Bearer Token' type. A warning message states: 'Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about variables'. The 'Token' field contains the variable `{{Token}}`.

Below the configuration, the 'Body' tab is selected, showing a status of '204 No Content', a time of '602 ms', and a size of '189 B'. The response is displayed in 'Pretty' format, showing a single line of content.

Delete user protection evidence

The screenshot shows a REST client interface with a sidebar on the left containing a file tree. The tree has folders for 'Academlo_Meals' and 'Motorcycle_Api'. Under 'Academlo_Meals', there is a 'users' folder with several endpoints: POST {{Host}}/users/signup, POST {{Host}}/users/login, PATCH {{Host}}/users/1, DEL {{Host}}/users/1, GET {{Host}}/users/orders, and GET {{Host}}/users/orders/1. Below this is a 'restaurants' folder, then 'meals' and 'reviews' folders, and finally an 'orders' folder with endpoints POST {{Host}}/meals, GET {{Host}}/meals/me, PATCH {{Host}}/meals/8, and DEL {{Host}}/meals/8. The 'Motorcycle_Api' folder is also visible.

The main panel shows a DELETE request to the URL `{{Host}}/users/3`. The 'Authorization' tab is selected, showing a 'Bearer Token' type and a token field containing `{{Token}}`. A warning message states: 'Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about [variables](#)'.

The 'Body' tab is selected, showing a JSON response in 'Pretty' format. The response is a 401 Unauthorized error with the following structure:

```
1 {
2   "status": "error",
3   "message": "You do not own this account",
4   "stack": "Error: You do not own this account\n    at protectAccountOwner (C:\\Academlo\\Node.js\\API-Academlo-Meals\\src\\modules\\users\\users.\n      middleware.js:23:19)\n    at Layer.handle [as handle_request] (C:\\Academlo\\Node.\n      js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\layer.js:95:5)\n    at next (C:\\Academlo\\Node.\n      js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\route.js:144:13)\n    at C:\\Academlo\\Node.\n      js\\API-Academlo-Meals\\src\\modules\\users\\users.middleware.js:15:5\n    at process.processTicksAndRejections (node:internal/process/\n      task_queues:95:5)",
5   "err": {
6     "statusCode": 401,
7     "status": "error",
8     "isOperational": true
9   }
10 }
```

The status bar at the bottom indicates 'Status: 401 Unauthorized', 'Time: 532 ms', and 'Size: 973 B'. There are also buttons for 'Save as example' and a search icon.

Deleted user evidence

Items	Queries	History								
Search for item...										
► Functions										
▲ Tables										
errors										
meals										
orders										
pg_stat_statements										
restaurants										
reviews										
users										
			id	name	email	password	status	role	createdAt	updatedAt
			1	User A	user@email.com	1234567...	available	admin	2024-01-...	2024-01-...
			2	User B	user2@email.com	1234567...	available	admin	2024-01-...	2024-01-...
			3	User C	user3@email.com	1234567...	available	admin	2024-01-...	2024-01-...
			4	User D	user4@email.com	1234567...	available	norm...	2024-01-...	2024-01-...
			5	User F	user5@email.com	1234567...	available	norm...	2024-01-...	2024-01-...
			6	Usuario Dev	usuarioDev@correo.com	\$2b\$14\$...	available	admin	2024-01-...	2024-01-...
			7	Usuario Non Dev	usuarioND@correo.com	\$2b\$14\$...	available	norm...	2024-01-...	2024-01-...
			8	El Prueba API's	usuarioTesteador@correo...	\$2b\$14\$...	disabled	admin	2024-01-...	2024-01-...

Find all user orders evidence

The screenshot shows a REST client interface with a sidebar on the left listing API endpoints. The main panel displays a GET request to `{{Host}}/users/orders` with a status of 200 OK. The response body is shown in JSON format, containing two order objects.

API Endpoints (Left Sidebar):

- Academlo_Meals
 - users
 - POST `{{Host}}/users/signup`
 - POST `{{Host}}/users/login`
 - PATCH `{{Host}}/users/1`
 - DEL `{{Host}}/users/1`
 - GET `{{Host}}/users/orders`
 - GET `{{Host}}/users/orders/1`
 - restaurants
 - meals
 - reviews
 - orders
 - POST `{{Host}}/meals`
 - GET `{{Host}}/meals/me`
 - PATCH `{{Host}}/meals/8`
 - DEL `{{Host}}/meals/8`
- Motorcycle_Api

Request Details (Main Panel):

- Method: GET
- URL: `{{Host}}/users/orders`
- Status: 200 OK
- Time: 532 ms
- Size: 813 B

Response Body (JSON):

```
{
  "id": 8,
  "mealId": 4,
  "userId": 8,
  "totalPrice": 30,
  "quantity": 2,
  "status": "completed",
  "createdAt": "2024-01-02T02:55:56.915Z",
  "updatedAt": "2024-01-02T02:59:02.271Z",
  "meal_id": 4,
  "user_id": 8,
  "meal": {
    "name": "HotDogos",
    "restaurant": {
      "name": "El Changarro"
    }
  }
},
{
  "id": 7,
  "mealId": 2,
  "userId": 8,
  "totalPrice": 100,
  "quantity": 4,
  "status": "cancelled",
  "createdAt": "2024-01-02T02:45:21.375Z",
  "updatedAt": "2024-01-02T02:59:42.855Z",
  "meal_id": 2,
  "user_id": 8,
```

Find one user order evidence

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

- Left Sidebar:** A tree view of API endpoints under the 'Academlo_Meals' folder. The 'orders' folder is expanded, and the endpoint `GET {{Host}}/users/orders` is selected.
- Top Bar:** The URL bar shows `Academlo_Meals / users / {{Host}}/users/orders`. The method is set to `GET` and the path is `{{Host}}/users/orders/7`. A 'Send' button is visible.
- Authorization Tab:** The 'Authorization' tab is active, showing 'Bearer Token' as the type. A warning message states: 'Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about variables'. The token field contains `{{Token}}`.
- Response Body:** The 'Body' tab is active, showing the response in JSON format. The status is `200 OK`, time is `546 ms`, and size is `564 B`. The response data is:

```
1 {
2   "message": "method get - findOne: from users",
3   "data": {
4     "id": 7,
5     "mealId": 2,
6     "userId": 8,
7     "totalPrice": 100,
8     "quantity": 4,
9     "status": "cancelled",
10    "createdAt": "2024-01-02T02:45:21.375Z",
11    "updatedAt": "2024-01-02T02:59:42.855Z",
12    "meal_id": 2,
13    "user_id": 8,
14    "meal": {
15      "name": "Chilakillers",
16      "restaurant": {
17        "name": "Crustáceo Cascarudo"
18      }
19    }
20  }
21 }
```

Find one user order error evidence

The screenshot shows a REST client interface with a sidebar on the left listing API endpoints. The main panel displays a GET request to `{{Host}}/users/orders/2` with a Bearer Token authorization header. The response body shows a 404 Not Found status with an error message: "The order with id: 2 does not exist on this account".

Endpoint List (Left Sidebar):

- Academlo_Meals
 - users
 - POST `{{Host}}/users/signup`
 - POST `{{Host}}/users/login`
 - PATCH `{{Host}}/users/1`
 - DEL `{{Host}}/users/1`
 - GET `{{Host}}/users/orders`
 - GET `{{Host}}/users/orders/1`
 - restaurants
 - meals
 - reviews
 - orders
 - POST `{{Host}}/meals`
 - GET `{{Host}}/meals/me`
 - PATCH `{{Host}}/meals/8`
 - DEL `{{Host}}/meals/8`
- Motorcycle_Api

Request Details (Main Panel):

- Method: GET
- URL: `{{Host}}/users/orders/2`
- Authorization: Bearer Token
- Token: `{{Token}}`

Response Details (Main Panel):

- Status: 404 Not Found
- Time: 178 ms
- Size: 653 B
- Body (Pretty):

```
1 {
2   "status": "error",
3   "message": "The order with id: 2 does not exist on this account",
4   "stack": "Error: The order with id: 2 does not exist on this account\n    at C:\\Academlo\\Node.js\\API-Academlo-Meals\\src\\modules\\users\\users.controller.js:99:21\n    at process.processTicksAndRejections (node:internal/process/task_queues:95:5)",
5   "err": {
6     "statusCode": 404,
7     "status": "error",
8     "isOperational": true
9   }
10 }
```

JWT users protection evidence

The screenshot shows a REST client interface with a sidebar on the left containing a file tree for 'Academlo_Meals' and 'Motorcycle_Api'. The main panel displays a GET request to 'Academlo_Meals / users / {{Host}}/users/orders/2'. The 'Authorization' tab is active, showing a 'Bearer Token' type and a 'Token' field with the value '{{Token}}'. A warning message states: 'Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about [variables](#)'. The 'Body' tab shows the response in JSON format, which is a 401 Unauthorized error. The response body is as follows:

```
1 {
2   "status": "error",
3   "message": "You are not logged in!. Please login to get access",
4   "stack": "Error: You are not logged in!. Please login to get access\n    at C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\src\\common\\middlewares\\middlewares.js:16:21\n    at C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\src\\common\\errors\\catchAsync.js:3:9\n    at Layer.handle [as handle_request] (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\layer.js:95:5)\n    at trim_prefix (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\index.js:328:13)\n    at C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\index.js:286:9\n    at Function.process_params (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\index.js:346:12)\n    at next (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\index.js:280:10)\n    at Function.handle (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\index.js:175:3)\n    at router (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\index.js:47:12)\n    at Layer.handle [as handle_request] (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\layer.js:95:5)",
5   "err": {
6     "statusCode": 401,
7     "status": "error",
8     "isOperational": true
9   }
10 }
```

Encrypted password evidence

users										orders			
id	name	email	password	status	role	createdAt	updatedAt						
1	User A	user@email.com	1234567890	available	admin	2024-01-...	2024-01-...						
2	User B	user2@email.com	1234567890	available	admin	2024-01-...	2024-01-...						
3	User C	user3@email.com	1234567890	available	admin	2024-01-...	2024-01-...						
4	User D	user4@email.com	1234567890	available	norm...	2024-01-...	2024-01-...						
5	User F	user5@email.com	1234567890	available	norm...	2024-01-...	2024-01-...						
6	Usuario Dev	usuarioDev@correo.com	\$2b\$14\$sAoCaiKfmZKqtLLrh7Y.ne96yj9vafyem/uExmCP8mSodIIx7xuF.	available	admin	2024-01-...	2024-01-...						
7	Usuario Non Dev	usuarioND@correo.com	\$2b\$14\$YKZ1aF3nK1joqK9S06CtruZjXYPHK/1BEo1Hxy.VP40d4S.cKsYTK	available	norm...	2024-01-...	2024-01-...						
8	El Prueba API's	usuarioTesteador@correo...	\$2b\$14\$S4tK26XuZMQ/oqunNfBay.CkPVxZII14hajHFYFznZP1HMRJqw/Eka	available	admin	2024-01-...	2024-01-...						
9	Usuario X	usuario@correo.com	\$2b\$14\$PYpfCj.RaMyCjRVteS.92un9IvBMQ2eSdKyIHJXnC/xrYwUsG8I.q	available	norm...	2024-01-...	2024-01-...						

Restaurants

Create restaurant evidence

The screenshot shows a REST client interface with a sidebar on the left containing a tree view of API endpoints. The main panel displays a POST request to `{{Host}}/restaurants` with a JSON body. The response is shown in the bottom panel, indicating a successful creation with a 201 status code.

Left Sidebar (API Explorer):

- Academlo_Meals
 - users
 - restaurants
 - POST {{Host}}/restaurants**
 - GET {{Host}}/restaurants
 - GET {{Host}}/restaurants/1
 - PATCH {{Host}}/restaurants/1
 - DEL {{Host}}/restaurants/1
 - meals
 - reviews
 - orders
- Motorcycle_Api

Main Panel (Request Setup):

- URL: `Academlo_Meals / restaurants / {{Host}}/restaurants`
- Method: **POST**
- Body Type: **JSON**
- Body Content:

```
1 {
2   ... "name": "Restaurant A",
3   ... "address": "ABC #123 col.XYZ",
4   ... "rating": 3
5 }
```

Bottom Panel (Response):

- Status: **201 Created**
- Time: **523 ms**
- Size: **384 B**
- Body Content (Pretty):

```
1 {
2   "message": "method post - create: from restaurants",
3   "restaurant": {
4     "name": "Restaurant A",
5     "address": "ABC #123 col.XYZ",
6     "rating": 3
7   }
8 }
```


Find all restaurants evidence

The screenshot shows a REST client interface with a sidebar on the left containing a project tree. The main panel displays a GET request to `{{Host}}/restaurants` with a status of 200 OK. The response body is shown in JSON format, containing an array of restaurant objects. The first object in the array is for a restaurant named 'Cocina de Mama Pancha' with an ID of 1. The second object is partially visible, showing an ID of 2 and a name 'Crustáceo Cascarudo'.

Academlo_Meals / restaurants / `{{Host}}/restaurants`

GET `{{Host}}/restaurants` Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (8) Test Results Status: 200 OK Time: 463 ms Size: 2.39 KB Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "method get - findAll: from restaurants",
3   "data": [
4     {
5       "id": 1,
6       "name": "Cocina de Mama Pancha",
7       "address": "1234 ABC col.XYZ",
8       "rating": 4,
9       "status": "active",
10      "createdAt": "2024-01-01T00:00:00.000Z",
11      "updatedAt": "2024-01-01T00:00:00.000Z",
12      "reviews": [
13        {
14          "id": 1,
15          "userId": 2,
16          "comment": "Uy que rico ñam ñam ☺",
17          "restaurantId": 1,
18          "rating": 4,
19          "status": "active",
20          "createdAt": "2024-01-01T00:00:00.000Z",
21          "updatedAt": "2024-01-01T00:00:00.000Z",
22          "restaurant_id": 1,
23          "user_id": 2
24        }
25      ]
26    },
27    {
28      "id": 2,
29      "name": "Crustáceo Cascarudo",
```

Find one restaurant evidence

The screenshot shows a REST client interface with a sidebar on the left containing a project tree. The main area displays a GET request to `{{Host}}/restaurants/2`. The response body is shown in JSON format, indicating a successful GET operation and returning details for a restaurant with ID 2.

Project Tree (Left Sidebar):

- Academlo_Meals
 - users
 - POST `{{Host}}/users/signup`
 - POST `{{Host}}/users/login`
 - PATCH `{{Host}}/users/1`
 - DEL `{{Host}}/users/1`
 - GET `{{Host}}/users/orders`
 - GET `{{Host}}/users/orders/1`
 - restaurants
 - POST `{{Host}}/restaurants`
 - GET `{{Host}}/restaurants`
 - GET `{{Host}}/restaurants/1`
 - PATCH `{{Host}}/restaurants/1`
 - DEL `{{Host}}/restaurants/1`
 - meals
 - reviews
 - orders
- Motorcycle_Api

Request Details:

- Method: GET
- URL: `{{Host}}/restaurants/2`
- Params: Query Params table with 2 columns (Key, Value, Description)

Response Details:

- Status: 200 OK
- Time: 466 ms
- Size: 703 B
- Body (Pretty):

```
1 {
2   "message": "method get - findOne: from restaurants",
3   "data": {
4     "id": 2,
5     "name": "Crustáceo Cascarudo",
6     "address": "1234 ABC col.XYZ",
7     "rating": 5,
8     "status": "active",
9     "createdAt": "2024-01-01T00:00:00.000Z",
10    "updatedAt": "2024-01-01T00:00:00.000Z",
11    "reviews": [
12      {
13        "id": 2,
14        "userId": 1,
15        "comment": "Uy que askooooo 🤔",
16        "restaurantId": 2,
17        "rating": 1,
18        "status": "active",
19        "createdAt": "2024-01-01T00:00:00.000Z",
```

Update restaurant evidence

The screenshot displays a REST client interface with a sidebar on the left and a main workspace on the right. The sidebar lists the API structure under 'Academlo_Meals', including endpoints for 'users' and 'restaurants'. The 'restaurants' folder is expanded, showing a 'PATCH {{Host}}/restaurants/1' endpoint selected. The main workspace shows the details for this endpoint, with the URL set to 'PATCH {{Host}}/restaurants/6'. The 'Body' tab is active, showing a JSON payload for updating a restaurant. Below the request, the response is displayed in the 'Body' tab, showing a successful status of 200 OK with a message and updated restaurant details.

Request:

```
1 {
2   ... "name": "Restaurant 1",
3   ... "address": "ABC #123 col.XYZ",
4   ... "rating": 4
5 }
```

Response:

```
1 {
2   "message": "method patch - update: from restaurants",
3   "updatedRestaurant": {
4     "id": 6,
5     "name": "Restaurant 1",
6     "address": "ABC #123 col.XYZ",
7     "rating": 3,
8     "status": "active",
9     "createdAt": "2024-01-02T21:25:37.455Z",
10    "updatedAt": "2024-01-02T21:28:32.003Z",
11    "reviews": []
12  }
13 }
```

Delete restaurant evidence

The screenshot displays a REST client interface with a sidebar on the left and a main workspace on the right. The sidebar, titled 'Academlo_Meals', contains a tree view of API endpoints. The 'restaurants' folder is expanded, showing a list of endpoints including a DELETE endpoint for 'restaurants/1'. The main workspace shows a DELETE request configuration for the endpoint 'Academlo_Meals / restaurants / {{Host}}/restaurants/6'. The request method is 'DELETE' and the URL is '({{Host}})/restaurants/6'. The 'Body' tab is selected, showing a message 'This request does not have a body'. The response section at the bottom shows a status of '204 No Content', a time of '614 ms', and a size of '189 B'. The response body is empty.

Academlo_Meals / restaurants / `{{Host}}/restaurants/1`

DELETE `{{Host}}/restaurants/6` Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

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

This request does not have a body

Body Cookies Headers (6) Test Results 🌐 Status: 204 No Content Time: 614 ms Size: 189 B 📄 Save as example ⋮

Pretty Raw Preview Visualize Text ⌵ 🔄

1

Deleted restaurant evidence

Items

Queries

History

Search for item...

⌵

Functions

Tables

errors

meals

orders

pg_stat_statements

restaurants

reviews

users

id	name	address	rating	status		createdAt	updatedAt	
1	Cocina de Mama Pancha	1234 ABC col.XYZ	4	active	⌵	2024-01-01 00:00:00+00	2024-01-01 00:00:00+00	
2	Crustáceo Cascarudo	1234 ABC col.XYZ	5	active	⌵	2024-01-01 00:00:00+00	2024-01-01 00:00:00+00	
3	Doña Pelos Foods	1234 ABC col.XYZ	3	active	⌵	2024-01-01 00:00:00+00	2024-01-01 00:00:00+00	
4	El Chagarro	1234 ABC col.XYZ	2	active	⌵	2024-01-01 00:00:00+00	2024-01-01 00:00:00+00	
5	Tacos don Mencho	1234 ABC col.XYZ	1	active	⌵	2024-01-01 00:00:00+00	2024-01-01 00:00:00+00	
6	Restaurant 1	ABC #123 col.XYZ	3	inactive	⌵	2024-01-02 21:25:37.455+00	2024-01-02 21:30:09.391+00	

JWT restaurant protection evidence

The screenshot shows a REST client interface with a sidebar on the left listing API endpoints under 'Academlo_Meals'. The main panel displays a 'DELETE' request to 'Academlo_Meals / restaurants / {{Host}}/restaurants/1'. The 'Body' tab is selected, showing 'This request does not have a body'. The response tab shows a '401 Unauthorized' status with a JSON error message.

Request:

- Method: DELETE
- URL: `{{Host}}/restaurants/1`
- Body: This request does not have a body

Response:

Status: 401 Unauthorized Time: 4 ms Size: 1.53 KB

```
{
  "status": "error",
  "message": "You are not logged in!. Please login to get access",
  "stack": "Error: You are not logged in!. Please login to get access\n    at C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\src\\common\\middlewares\\middlewares.js:16:21\n    at C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\src\\common\\errors\\catchAsync.js:3:9\n    at Layer.handle [as handle_request] (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\layer.js:95:5)\n    at next (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\route.js:144:13)\n    at next (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\route.js:140:7)\n    at next (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\route.js:140:7)\n    at next (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\route.js:140:7)\n    at next (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\route.js:140:7)\n    at next (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\route.js:140:7)\n    at next (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\route.js:140:7)",
  "err": {
    "statusCode": 401,
    "status": "error",
    "isOperational": true
  }
}
```

Restaurant admin restriction evidence

The screenshot displays a REST client interface with a sidebar on the left containing a project tree. The main area shows a POST request to `{{Host}}/restaurants` with a JSON body. The response is a 403 Forbidden status with a detailed error message and stack trace.

Request:

```
POST {{Host}}/restaurants
```

```
{
  "name": "Restaurant Testing",
  "address": "ABC #123 col.XYZ",
  "rating": 5
}
```

Response:

```
{
  "status": "error",
  "message": "You do not have permission to perform this action",
  "stack": "Error: You do not have permission to perform this action\n    at C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\src\\\\common\\\\middlewares\\\\middlewares.js:51:17\n    at Layer.handle [as handle_request] (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\layer.js:95:5)\n    at Route.dispatch (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\route.js:144:13)\n    at Layer.handle [as handle_request] (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\layer.js:95:5)\n    at Function.process_params (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\index.js:284:15)\n    at next (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\index.js:346:12)\n    at C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\index.js:280:10)\n    at process.processTicksAndRejections (node:internal/process/task_queues:95:5)",
  "err": {
    "statusCode": 403,
    "status": "error",
    "isOperational": true
  }
}
```

Status: 403 Forbidden **Time:** 450 ms **Size:** 1.55 KB

Restaurant rating validator evidence

The screenshot displays a REST client interface with a sidebar on the left showing a project structure under 'Academlo_Meals'. The main area shows a POST request to the endpoint `{{Host}}/restaurants`. The request body is a JSON object:

```
1 {
2   ... "name": "Restaurant Testing",
3   ... "address": "ABC-#123-col.XYZ",
4   "rating": 7
5 }
```

The response status is **422 Unprocessable Entity** with a time of 407 ms and a size of 445 B. The response body, shown in JSON format, contains an error message:

```
1 {
2   "status": "error",
3   "message": [
4     {
5       "code": "too_big",
6       "maximum": 5,
7       "type": "number",
8       "inclusive": true,
9       "exact": false,
10      "message": "Number must be less than or equal to 5",
11      "path": [
12        "rating"
13      ]
14    }
15  ]
16 }
```


Reviews

Create review evidence

The screenshot displays a REST client interface with a sidebar on the left containing a tree view of API endpoints. The main panel shows a POST request to `{{Host}}/restaurants/reviews/6` with a JSON body. The response is shown in the bottom panel, indicating a successful creation of a review.

API Endpoints (Left Sidebar):

- Academlo_Meals
 - users
 - POST `{{Host}}/users/signup`
 - POST `{{Host}}/users/login`
 - PATCH `{{Host}}/users/1`
 - DEL `{{Host}}/users/1`
 - GET `{{Host}}/users/orders`
 - GET `{{Host}}/users/orders/1`
 - restaurants
 - POST `{{Host}}/restaurants`
 - GET `{{Host}}/restaurants`
 - GET `{{Host}}/restaurants/1`
 - PATCH `{{Host}}/restaurants/1`
 - DEL `{{Host}}/restaurants/1`
 - meals
 - reviews
 - POST `{{Host}}/restaurants/review/1`
 - PATCH `{{Host}}/restaurants/review/...`
 - DEL `{{Host}}/restaurants/review/...`
 - orders
 - Motorcycle_Api

Request Details (Main Panel):

- Method: POST
- URL: `{{Host}}/restaurants/reviews/6`
- Body (JSON):

```
{  1: {  2:   "comment": "Ñam ñam que rica comida",  3:   "rating": 5  4: }
```

Response Details (Bottom Panel):

- Status: 201 Created
- Time: 201 ms
- Size: 492 B
- Body (JSON):

```
{  1: {  2:   "message": "method post - create: from reviews",  3:   "data": {  4:     "status": "active",  5:     "id": 6,  6:     "comment": "Ñam ñam que rica comida",  7:     "rating": 5,  8:     "restaurantId": 6,  9:     "userId": 6, 10:    "updatedAt": "2024-01-02T21:40:55.687Z", 11:    "createdAt": "2024-01-02T21:40:55.687Z" 12:   } 13: }
```

Update review evidence

The screenshot displays a REST client interface with a sidebar on the left and a main workspace on the right. The sidebar lists the API structure under 'Academlo_Meals', including endpoints for 'users', 'restaurants', 'meals', 'reviews', 'orders', and 'Motorcycle_Api'. The 'reviews' folder is expanded, showing a list of review endpoints. The main workspace is configured for a PATCH request to the endpoint `{{Host}}/restaurants/review/6/6`. The request body is a JSON object:

```
{  "comment": "wakala kiaskooooo",  "rating": 1}
```

. The response is displayed in the bottom panel, showing a JSON object with a success message and updated review details:

```
{  "message": "method patch - update: from reviews",  "updateRestaurant": {    "id": 6,    "userId": 6,    "comment": "wakala kiaskooooo",    "restaurantId": 6,    "rating": 1,    "status": "active",    "createdAt": "2024-01-02T21:40:55.687Z",    "updatedAt": "2024-01-02T21:43:02.004Z",    "restaurant_id": 6,    "user_id": 6  }}}
```

Academlo_Meals / reviews / `{{Host}}/restaurants/review/2/2` Save Send

PATCH `{{Host}}/restaurants/reviews/6/6`

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies Beautify

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

```
1 {
2   "comment": "wakala kiaskooooo",
3   "rating": 1
4 }
```

Body Cookies Headers (8) Test Results Status: 200 OK Time: 653 ms Size: 523 B Save as example

Pretty Raw Preview Visualize **JSON**

```
1 {
2   "message": "method patch - update: from reviews",
3   "updateRestaurant": {
4     "id": 6,
5     "userId": 6,
6     "comment": "wakala kiaskooooo",
7     "restaurantId": 6,
8     "rating": 1,
9     "status": "active",
10    "createdAt": "2024-01-02T21:40:55.687Z",
11    "updatedAt": "2024-01-02T21:43:02.004Z",
12    "restaurant_id": 6,
13    "user_id": 6
14  }
15 }
```

Update review protection evidence

The screenshot displays a REST client interface with a sidebar on the left containing a project tree. The tree includes folders for 'Academlo_Meals', 'users', 'restaurants', 'meals', 'reviews', 'orders', and 'Motorcycle_Api'. The 'reviews' folder is expanded, showing a list of endpoints. The main panel shows a PATCH request to `{{Host}}/restaurants/review/5/5`. The request body is a JSON object: `{ "comment": "wakala kiaskoooo", "rating": 1 }`. The response status is `401 Unauthorized` with a message: `"You do not own the account of this review"`. The response body is a JSON object: `{ "status": "error", "message": "You do not own the account of this review", "stack": "Error: You do not own the account of this review\n at C:\\Academlo\\Node.js\\API-Academlo-Meals\\src\\modules\\restaurants\\restaurants.controller.js:120:21\n at process.processTicksAndRejections (node:internal/process/task_queues:95:5)", "err": { "statusCode": 401, "status": "error", "isOperational": true } }`.

Academlo_Meals / reviews / `{{Host}}/restaurants/review/2/2`

PATCH `{{Host}}/restaurants/reviews/5/5` Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   ... "comment": "wakala kiaskoooo",
3   ... "rating": 1
4 }
```

Body Cookies Headers (8) Test Results Status: 401 Unauthorized Time: 530 ms Size: 649 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "status": "error",
3   "message": "You do not own the account of this review",
4   "stack": "Error: You do not own the account of this review\n at C:\\Academlo\\Node.\njs\\API-Academlo-Meals\\src\\modules\\restaurants\\restaurants.controller.js:120:21\n at process.processTicksAndRejections (node:internal/\nprocess/task_queues:95:5)",
5   "err": {
6     "statusCode": 401,
7     "status": "error",
8     "isOperational": true
9   }
10 }
```

Delete review evidence

The screenshot shows a REST client interface with a sidebar on the left containing a file tree. The tree has folders for 'Academlo_Meals', 'meals', 'reviews', and 'orders'. Under 'Academlo_Meals', there are sub-folders 'users' and 'restaurants'. The 'users' folder contains endpoints for signup, login, and orders. The 'restaurants' folder contains endpoints for listing, getting a single restaurant, and deleting a restaurant. The 'reviews' folder is expanded, showing endpoints for creating, updating, and deleting a review. The 'DELETE {{Host}}/restaurants/review/...' endpoint is selected.

The main panel displays the details of the selected endpoint: `DELETE {{Host}}/restaurants/reviews/6/6`. The 'Params' tab is active, showing a table with one row: 'Key' and 'Value'. The 'Body' tab is also visible, showing the response status: '204 No Content', 'Time: 609 ms', and 'Size: 189 B'. The response body is empty.

Academlo_Meals

- users
 - POST `{{Host}}/users/signup`
 - POST `{{Host}}/users/login`
 - PATCH `{{Host}}/users/1`
 - DEL `{{Host}}/users/1`
 - GET `{{Host}}/users/orders`
 - GET `{{Host}}/users/orders/1`
- restaurants
 - POST `{{Host}}/restaurants`
 - GET `{{Host}}/restaurants`
 - GET `{{Host}}/restaurants/1`
 - PATCH `{{Host}}/restaurants/1`
 - DEL `{{Host}}/restaurants/1`
- meals
- reviews
 - POST `{{Host}}/restaurants/review/1`
 - PATCH `{{Host}}/restaurants/review/...`
 - DEL `{{Host}}/restaurants/review/...`
- orders

Motorcycle_Api

Academlo_Meals / reviews / `{{Host}}/restaurants/review/2/2`

DELETE `{{Host}}/restaurants/reviews/6/6` Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body Cookies Headers (6) Test Results Status: 204 No Content Time: 609 ms Size: 189 B Save as example

Pretty Raw Preview Visualize Text

1

Delete review protection evidence

The screenshot shows a REST client interface with a sidebar on the left containing a tree view of API endpoints. The main panel displays a DELETE request to `{{Host}}/restaurants/reviews/5/5`. The response is a 401 Unauthorized status with a JSON body indicating an error.

Endpoint: `DELETE {{Host}}/restaurants/reviews/5/5`

Params: Query Params table with columns: Key, Value, Description.

Key	Value	Description
Key	Value	Description

Body: Status: 401 Unauthorized, Time: 525 ms, Size: 649 B. The response body is shown in JSON format:

```
1 {
2   "status": "error",
3   "message": "You do not own the account of this review",
4   "stack": "Error: You do not own the account of this review\n    at C:\\Academlo\\Node.js\\API-Academlo-Meals\\src\\modules\\restaurants\\restaurants.controller.js:138:21\n    at process.processTicksAndRejections (node:internal/process/task_queues:95:5)",
5   "err": {
6     "statusCode": 401,
7     "status": "error",
8     "isOperational": true
9   }
10 }
```

Deleted review evidence

Items

Queries

History

Search for item...

Functions

Tables

errors

meals

orders

pg_stat_statements

restaurants

reviews

users

restaurants				users					
id	user_id	comment	restaurant_id	rating	status	createdAt	updatedAt		
1	2 →	Uy que rico ñam ñam 😊	1 →	4	active	2024-01-...	2024-01-...		
2	1 →	Uy que askoooo 🤔	2 →	1	active	2024-01-...	2024-01-...		
3	3 →	Ewk wakala 🤔	4 →	2	active	2024-01-...	2024-01-...		
4	4 →	Delicious 😊	3 →	5	active	2024-01-...	2024-01-...		
5	5 →	Hay mejores hay peores 😐	5 →	3	active	2024-01-...	2024-01-...		
6	6 →	wakala kiaskooooo	6 →	1	deleted	2024-01-...	2024-01-...		

Meals

Create meal evidence

The screenshot shows a REST client interface with a sidebar on the left containing a tree view of API endpoints. The main area displays a POST request to `{{Host}}/meals/6` with a JSON body. Below the request, the response is shown in a 'Pretty' JSON format, indicating a successful creation with a 201 status.

Left Sidebar (API Explorer):

- Academlo_Meals
 - users
 - restaurants
 - meals
 - POST `{{Host}}/meals/5`
 - GET `{{Host}}/meals`
 - GET `{{Host}}/meals/2`
 - PATCH `{{Host}}/meals/2`
 - DEL `{{Host}}/meals/2`
 - reviews
 - orders
- Motorcycle_Api

Main Request Area:

- Method: POST
- URL: `{{Host}}/meals/6`
- Body Type: JSON
- Body (Raw):

```
1 {
2   "name": "Tacos de ojos de pez",
3   "price": 10
4 }
```

Response Area:

- Status: 201 Created
- Time: 544 ms
- Size: 472 B
- Body (Pretty):

```
1 {
2   "messages": "method post - create: from meals",
3   "data": {
4     "status": "active",
5     "id": 8,
6     "name": "Tacos de ojos de pez",
7     "price": 10,
8     "restaurantId": 6,
9     "updatedAt": "2024-01-02T21:48:41.324Z",
10    "createdAt": "2024-01-02T21:48:41.324Z"
11  }
12 }
```

Find all meals evidence

The screenshot shows a REST client interface with a sidebar on the left containing a project tree. The main area displays a GET request to `{{Host}}/meals` with a status of 200 OK, a time of 448 ms, and a size of 2.75 KB. The response body is shown in JSON format, containing a message and an array of two meal objects.

Project Tree:

- Academlo_Meals
 - users
 - restaurants
 - meals
 - POST `{{Host}}/meals/5`
 - GET `{{Host}}/meals`
 - GET `{{Host}}/meals/2`
 - PATCH `{{Host}}/meals/2`
 - DEL `{{Host}}/meals/2`
 - reviews
 - orders
- Motorcycle_Api

Request Details:

- Method: GET
- URL: `{{Host}}/meals`
- Params: (empty)
- Authorization: (empty)
- Headers: (7)
- Body: (empty)
- Pre-request Script: (empty)
- Tests: (empty)
- Settings: (empty)

Response Details:

- Status: 200 OK
- Time: 448 ms
- Size: 2.75 KB
- Save as example

Response Body (JSON):

```
1 {
2   "message": "method get - findAll: from meals",
3   "data": [
4     {
5       "id": 1,
6       "name": "Tacoshando",
7       "price": 28,
8       "restaurantId": 1,
9       "status": "active",
10      "createdAt": "2024-01-01T00:00:00.000Z",
11      "updatedAt": "2024-01-01T00:00:00.000Z",
12      "restaurant_id": 1,
13      "restaurant": {
14        "id": 1,
15        "name": "Cocina de Mama Pancha",
16        "address": "1234 ABC col.XYZ",
17        "rating": 4,
18        "status": "active",
19        "createdAt": "2024-01-01T00:00:00.000Z",
20        "updatedAt": "2024-01-01T00:00:00.000Z"
21      }
22    },
23    {
24      "id": 3,
25      "name": "Gorditas Calientes mmmm",
26      "price": 38,
27      "restaurantId": 3,
28      "status": "active",
29      "createdAt": "2024-01-01T00:00:00.000Z",
```

Find one meal evidence

The screenshot shows a REST client interface with a sidebar on the left containing a project tree. The tree includes folders for 'Academlo_Meals' and 'Motorcycle_Api'. Under 'Academlo_Meals', there are sub-folders for 'users', 'restaurants', and 'meals'. The 'meals' folder is expanded, showing a list of requests: 'POST {{Host}}/meals/5', 'GET {{Host}}/meals', 'GET {{Host}}/meals/2' (which is selected), 'PATCH {{Host}}/meals/2', and 'DEL {{Host}}/meals/2'. Other folders like 'reviews' and 'orders' are also visible under 'Academlo_Meals'.

The main panel displays the details of the selected 'GET {{Host}}/meals/2' request. The method is 'GET' and the URL is '{{Host}}/meals/2'. The 'Send' button is visible. Below the request details, there are tabs for 'Params', 'Authorization', 'Headers (7)', 'Body', 'Pre-request Script', 'Tests', and 'Settings'. The 'Body' tab is active, showing the response in 'Pretty' format. The response is a JSON object with a status of '200 OK', a time of '456 ms', and a size of '665 B'. The JSON content is as follows:

```
1 {
2   "message": "method get - findOne: from meals",
3   "data": {
4     "id": 2,
5     "name": "Chilakillers",
6     "price": 25,
7     "restaurantId": 2,
8     "status": "active",
9     "createdAt": "2024-01-01T00:00:00.000Z",
10    "updatedAt": "2024-01-02T00:37:01.447Z",
11    "restaurant_id": 2,
12    "restaurant": {
13      "id": 2,
14      "name": "Crustáceo Cascarudo",
15      "address": "1234 ABC col.XYZ",
16      "rating": 5,
17      "status": "active",
18      "createdAt": "2024-01-01T00:00:00.000Z",
19      "updatedAt": "2024-01-01T00:00:00.000Z"
20    }
21  }
22 }
```

Update meal evidence

The screenshot displays a REST client interface with a sidebar on the left and a main workspace on the right.

Sidebar:

- Academlo_Meals
 - users
 - restaurants
 - meals
 - POST {{Host}}/meals/5
 - GET {{Host}}/meals
 - GET {{Host}}/meals/2
 - PATCH {{Host}}/meals/2**
 - DEL {{Host}}/meals/2
 - reviews
 - orders
- Motorcycle_Api

Main Workspace:

- URL: `Academlo_Meals / meals / {{Host}}/meals/2`
- Method: **PATCH**
- Body (JSON):

```
{  1  {  2    "name": "Chilakillers",  3    "price": 30  4  }
```
- Response (JSON):

```
1  {  2    "message": "method patch - update: from meals",  3    "data": {  4      "id": 2,  5      "name": "Chilakillers",  6      "price": 30,  7      "restaurantId": 2,  8      "status": "active",  9      "createdAt": "2024-01-01T00:00:00.000Z", 10     "updatedAt": "2024-01-02T21:55:45.408Z", 11     "restaurant_id": 2, 12     "restaurant": { 13       "id": 2, 14       "name": "Crustáceo Cascarudo", 15       "address": "1234 ABC col.XYZ", 16       "rating": 5, 17       "status": "active", 18       "createdAt": "2024-01-01T00:00:00.000Z", 19       "updatedAt": "2024-01-01T00:00:00.000Z" 20     } 21   } 22 }
```
- Status: 200 OK, Time: 605 ms, Size: 666 B

Delete meal evidence

The screenshot shows a REST client interface with a sidebar on the left containing a tree view of API collections. The main panel displays a DELETE request configuration for the endpoint `{{Host}}/meals/2`. The request is configured with a `DELETE` method and a `Send` button. Below the request configuration, there are tabs for `Params`, `Authorization`, `Headers (7)`, `Body`, `Pre-request Script`, `Tests`, and `Settings`. The `Params` tab is active, showing a table for Query Params. The response section at the bottom shows a status of `204 No Content`, a time of `614 ms`, and a size of `189 B`. The response body is empty, and the `Body` tab is active.

Academlo_Meals / meals / `{{Host}}/meals/2`

DELETE `{{Host}}/meals/2` Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body Cookies Headers (6) Test Results

Status: 204 No Content Time: 614 ms Size: 189 B Save as example

Pretty Raw Preview Visualize Text

1

Deleted meal evidence

restaurants								
id	name	price	restaurant_id	status		createdAt	updatedAt	
1	Tacoshando	28	1 →	active	✓	2024-01-...	2024-01-...	
2	Chilakillers	30	2 →	inactive	✓	2024-01-...	2024-01-...	
3	Gorditas Calientes mmmmm	38	3 →	active	✓	2024-01-...	2024-01-...	
4	HotDogos	15	4 →	active	✓	2024-01-...	2024-01-...	
5	Higado en su jugo	25	5 →	active	✓	2024-01-...	2024-01-...	
7	Pitalu	3	5 →	active	✓	2024-01-...	2024-01-...	
8	Tacos de ojos de pez	10	6 →	active	✓	2024-01-...	2024-01-...	

JWT meals protection evidence

The screenshot displays a REST client interface with a sidebar on the left showing a project structure: Academlo_Meals (containing users, restaurants, and meals) and Motorcycle_Api. The main panel shows a POST request to `{{Host}}/meals/5` with a JSON body: `{ "name": "Pitalu", "price": 3 }`. The request was sent, but the response is a 401 Unauthorized status. The response body, shown in Pretty JSON format, contains an error message and a detailed stack trace.

Request:

- Method: POST
- URL: `{{Host}}/meals/5`
- Body (JSON):

```
{  "name": "Pitalu",  "price": 3}
```

Response:

- Status: 401 Unauthorized
- Time: 6 ms
- Size: 1.58 KB
- Body (Pretty JSON):

```
{  "status": "error",  "message": "You are not logged in!. Please login to get access",  "stack": "Error: You are not logged in!. Please login to get access\n    at C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\src\\\\common\\\\middlewares\\\\middlewares.js:16:21\n    at C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\src\\\\common\\\\errors\\\\catchAsync.js:3:9\n    at Layer.handle [as handle_request] (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\layer.js:95:5)\n    at trim_prefix (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\index.js:328:13)\n    at C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\index.js:286:9\n    at Function.process_params (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\index.js:346:12)\n    at next (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\index.js:280:10)\n    at Function.handle (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\index.js:175:3)\n    at router (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\index.js:47:12)\n    at Layer.handle [as handle_request] (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\layer.js:95:5)",  "err": {    "statusCode": 401,    "status": "error",    "isOperational": true  }}
```

Meals admin restriction evidence

The screenshot displays a REST client interface with a sidebar on the left showing a project structure: Academlo_Meals (containing users, restaurants, and meals) and Motorcycle_Api. The main panel shows a POST request to `{{Host}}/meals/5` with a JSON body: `{ "name": "Pitalu", "price": 3 }`. The response status is 403 Forbidden, with a time of 437 ms and a size of 1.59 KB. The response body, shown in Pretty JSON format, contains an error message and a stack trace.

Request:

- Method: POST
- URL: `{{Host}}/meals/5`
- Body (JSON):

```
{  "name": "Pitalu",  "price": 3}
```

Response:

- Status: 403 Forbidden
- Time: 437 ms
- Size: 1.59 KB
- Body (Pretty JSON):

```
{  "status": "error",  "message": "You do not have permission to perform this action",  "stack": "Error: You do not have permission to perform this action\n    at C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\src\\\\common\\\\middlewares\\\\middlewares.js:51:17\n    at Layer.handle [as handle_request] (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\layer.js:95:5)\n    at Route.dispatch (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\route.js:144:13)\n    at Layer.handle [as handle_request] (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\layer.js:95:5)\n    at C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\index.js:284:15\n    at param (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\index.js:365:14)\n    at param (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\index.js:376:14)\n    at Function.process_params (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\index.js:421:3)\n    at next (C:\\\\Academlo\\\\Node.\n    js\\\\API-Academlo-Meals\\\\node_modules\\\\express\\\\lib\\\\router\\\\index.js:280:10)",  "err": {    "statusCode": 403,    "status": "error",    "isOperational": true  }}
```


Orders

Create order evidence

The screenshot displays a REST client interface with a sidebar on the left containing a tree view of API endpoints. The main panel shows a POST request to `{{Host}}/orders` with a JSON body. Below the request, the response is shown in a 'Pretty' JSON format.

API Endpoints (Left Sidebar):

- Academlo_Meals
 - users
 - restaurants
 - meals
 - POST `{{Host}}/meals/5`
 - GET `{{Host}}/meals`
 - GET `{{Host}}/meals/2`
 - PATCH `{{Host}}/meals/2`
 - DEL `{{Host}}/meals/2`
 - reviews
 - POST `{{Host}}/restaurants/review/1`
 - PATCH `{{Host}}/restaurants/review/...`
 - DEL `{{Host}}/restaurants/review/...`
 - orders
 - POST `{{Host}}/orders` (Selected)
 - GET `{{Host}}/orders/me`
 - PATCH `{{Host}}/orders/8`
 - DEL `{{Host}}/orders/8`
 - Motorcycle_Api

Request Details (Main Panel):

 - Method: POST
 - URL: `{{Host}}/orders`
 - Body (JSON):

```
1 {
2   "quantity": 4,
3   "mealId": 2
4 }
```

Response Details (Main Panel):

 - Status: 201 Created
 - Time: 618 ms
 - Size: 467 B
 - Body (Pretty JSON):

```
1 {
2   "messages": "method post - create: from orders",
3   "data": {
4     "status": "active",
5     "id": 9,
6     "quantity": 4,
7     "mealId": 2,
8     "userId": 7,
9     "totalPrice": 120,
10    "updatedAt": "2024-01-02T22:06:37.879Z",
11    "createdAt": "2024-01-02T22:06:37.879Z"
12  }
13 }
```

Create order meal validation evidence

The screenshot displays a REST client interface with a sidebar on the left listing API endpoints. The main panel shows a POST request to `{{Host}}/orders` with a JSON body: `{ "quantity": 2, "mealId": 6 }`. The response status is `404 Not Found` with a message: `"Meal with id: 6 not found"`. The error details in the response body indicate the meal ID 6 does not exist.

Request:

- Method: POST
- URL: `{{Host}}/orders`
- Body (JSON):

```
1 {
2   "quantity": 2,
3   "mealId": 6
4 }
```

Response:

- Status: 404 Not Found
- Time: 511 ms
- Size: 603 B
- Body (JSON):

```
1 {
2   "status": "error",
3   "message": "Meal with id: 6 not found",
4   "stack": "Error: Meal with id: 6 not found\n    at C:\\Academlo\\Node.js\\API-Academlo-Meals\\src\\modules\\orders\\orders.middleware.js:25:21\n    at process.processTicksAndRejections (node:internal/process/task_queues:95:5)",
5   "err": {
6     "statusCode": 404,
7     "status": "error",
8     "isOperational": true
9   }
10 }
```

Find all user orders evidence

The screenshot shows a REST client interface with a sidebar on the left listing API endpoints. The main panel displays a GET request to `{{Host}}/orders/me` with a status of 200 OK. The response body is shown in JSON format, containing two order items.

Endpoint: `Academlo_Meals / orders / {{Host}}/orders/me`

Method: GET

Status: 200 OK, Time: 533 ms, Size: 821 B

Response Body (JSON):

```
1 {
2   "messages": "method get - findAll: from orders",
3   "data": [
4     {
5       "id": 9,
6       "mealId": 2,
7       "userId": 7,
8       "totalPrice": 120,
9       "quantity": 4,
10      "status": "active",
11      "createdAt": "2024-01-02T22:06:37.879Z",
12      "updatedAt": "2024-01-02T22:06:37.879Z",
13      "meal_id": 2,
14      "user_id": 7,
15      "meal": {
16        "name": "Chilakillers",
17        "restaurant": {
18          "name": "Crustáceo Cascarudo"
19        }
20      }
21    },
22    {
23      "id": 6,
24      "mealId": 2,
25      "userId": 7,
26      "totalPrice": 75,
27      "quantity": 3,
28      "status": "active",
29      "createdAt": "2024-01-02T02:36:38.882Z",
```

Update order evidence

The screenshot shows a REST client interface with a sidebar on the left and a main panel on the right. The sidebar lists the API structure under 'Academlo_Meals', including endpoints for users, restaurants, meals, reviews, and orders. The 'orders' endpoint is expanded, showing a PATCH request to `{{Host}}/orders/8`. The main panel displays the details of this PATCH request, including the URL, method, and the JSON body.

URL: `Academlo_Meals / orders / {{Host}}/orders/8`

Method: PATCH

Query Params:

Key	Value	Description
Key	Value	Description

Body:

```
{
  "messages": "method patch - update: from orders",
  "data": {
    "id": 9,
    "mealId": 2,
    "userId": 7,
    "totalPrice": 120,
    "quantity": 4,
    "status": "completed",
    "createdAt": "2024-01-02T22:06:37.879Z",
    "updatedAt": "2024-01-02T22:08:52.231Z",
    "meal_id": 2,
    "user_id": 7,
    "meal": {
      "name": "Chilakillers",
      "restaurant": {
        "name": "Crustáceo Cascarudo"
      }
    }
  }
}
```

Status: 200 OK **Time:** 695 ms **Size:** 567 B

Update order protection evidence

The screenshot shows a REST client interface with a sidebar on the left containing a project tree. The tree includes folders for 'Academlo_Meals' and 'Motorcycle_Api'. Under 'Academlo_Meals', there are folders for 'users', 'restaurants', and 'meals'. The 'meals' folder is expanded, showing several endpoints: 'POST {{Host}}/meals/5', 'GET {{Host}}/meals', 'GET {{Host}}/meals/2', 'PATCH {{Host}}/meals/2', and 'DEL {{Host}}/meals/2'. The 'orders' folder is also expanded, showing 'POST {{Host}}/orders', 'GET {{Host}}/orders/me', 'PATCH {{Host}}/orders/8', and 'DEL {{Host}}/orders/8'. The 'PATCH {{Host}}/orders/8' endpoint is selected.

The main panel shows the details of the selected endpoint. The URL is 'PATCH {{Host}}/orders/8'. The 'Params' tab is active, showing a table with two columns: 'Key' and 'Value'. The table is empty.

The 'Body' tab is active, showing the response body in JSON format. The response is a 401 Unauthorized error. The status is '401 Unauthorized', the time is '610 ms', and the size is '636 B'. The response body is:

```
{
  "status": "error",
  "message": "You do not own the account of this order",
  "stack": "Error: You do not own the account of this order\n    at C:\\\\Academlo\\\\Node.js\\\\API-Academlo-Meals\\\\src\\\\modules\\\\orders\\\\orders.controller.js:53:21\n    at process.processTicksAndRejections (node:internal/process/task_queues:95:5)",
  "err": {
    "statusCode": 401,
    "status": "error",
    "isOperational": true
  }
}
```

Delete order evidence

The screenshot shows a REST client interface with a sidebar on the left containing a tree view of API collections. The main panel displays a DELETE request configuration for the endpoint `{{Host}}/orders/9`. The request is configured with the method `DELETE` and the URL `{{Host}}/orders/9`. The response status is `204 No Content`, with a time of `695 ms` and a size of `189 B`. The response body is empty.

Academlo_Meals

- users
- restaurants
- meals
 - POST `{{Host}}/meals/5`
 - GET `{{Host}}/meals`
 - GET `{{Host}}/meals/2`
 - PATCH `{{Host}}/meals/2`
 - DEL `{{Host}}/meals/2`
- reviews
 - POST `{{Host}}/restaurants/review/1`
 - PATCH `{{Host}}/restaurants/review/...`
 - DEL `{{Host}}/restaurants/review/...`
- orders
 - POST `{{Host}}/orders`
 - GET `{{Host}}/orders/me`
 - PATCH `{{Host}}/orders/8`
 - DEL `{{Host}}/orders/8`
- Motorcycle_Api

DELETE `{{Host}}/orders/9` **Send**

Params **Authorization** **Headers (7)** **Body** **Pre-request Script** **Tests** **Settings** **Cookies**

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body **Cookies** **Headers (6)** **Test Results** **Status: 204 No Content** **Time: 695 ms** **Size: 189 B** **Save as example**

Pretty **Raw** **Preview** **Visualize** **Text** **Copy** **Search**

1

Delete order protection evidence

The screenshot shows a REST client interface with a sidebar on the left containing a file tree. The main area displays a DELETE request to `{{Host}}/orders/8`. The response is a 401 Unauthorized status with a JSON body containing an error message and stack trace.

Request:

- Method: DELETE
- URL: `{{Host}}/orders/8`

Response:

- Status: 401 Unauthorized
- Time: 615 ms
- Size: 636 B

Body (JSON):

```
1 {
2   "status": "error",
3   "message": "You do not own the account of this order",
4   "stack": "Error: You do not own the account of this order\n    at C:\\\\Academlo\\\\Node.js\\\\API-Academlo-Meals\\\\src\\\\modules\\\\orders\\\\orders.\n              controller.js:71:21\n    at process.processTicksAndRejections (node:internal/process/task_queues:95:5)",
5   "err": {
6     "statusCode": 401,
7     "status": "error",
8     "isOperational": true
9   }
10 }
```


Deleted order evidence

Items			meals					users		
Search for item...			id	meal_id	user_id	totalPrice	quantity	status	createdAt	updatedAt
Functions			1	2 →	3 →	15	1	active	2024-01-...	2024-01-01 00:00:00+00
Tables			2	4 →	5 →	45	3	active	2024-01-...	2024-01-01 00:00:00+00
errors			3	1 →	1 →	55	2	active	2024-01-...	2024-01-01 00:00:00+00
meals			4	3 →	2 →	115	3	active	2024-01-...	2024-01-01 00:00:00+00
orders			5	5 →	4 →	50	2	active	2024-01-...	2024-01-02 03:00:25.986+...
pg_stat_statements			6	2 →	7 →	75	3	completed	2024-01-...	2024-01-02 22:14:07.489+...
restaurants			7	2 →	8 →	100	4	active	2024-01-...	2024-01-02 02:59:42.855+...
reviews			8	4 →	8 →	30	2	active	2024-01-...	2024-01-02 02:59:02.271+...
users			9	2 →	7 →	120	4	cancelled	2024-01-...	2024-01-02 22:18:13.793+...

JWT order protection evidence

The screenshot displays a REST client interface with a sidebar on the left listing API endpoints under 'Academlo_Meals' and 'Motorcycle_Api'. The main panel shows a POST request to `{{Host}}/orders` with a JSON body: `{ "quantity": 2, "mealId": 4 }`. The response is a 401 Unauthorized status with a detailed error message and stack trace.

Request Details:

- Method: POST
- URL: `{{Host}}/orders`
- Body (JSON):

```
{  1  {  2    "quantity": 2,  3    "mealId": 4  4  }
```

Response Details:

- Status: 401 Unauthorized
- Time: 14 ms
- Size: 1.58 KB
- Body (Pretty):

```
1  {  2    "status": "error",  3    "message": "You are not logged in!. Please login to get access",  4    "stack": "Error: You are not logged in!. Please login to get access\n    at C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\src\\common\\middlewares\\middlewares.js:16:21\n    at C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\src\\common\\errors\\catchAsync.js:3:9\n    at Layer.handle [as handle_request] (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\layer.js:95:5)\n    at trim_prefix (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\index.js:328:13)\n    at C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\index.js:286:9\n    at Function.process_params (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\index.js:346:12)\n    at Function.handle (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\index.js:280:10)\n    at router (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\index.js:175:3)\n    at Layer.handle [as handle_request] (C:\\Academlo\\Node.\n    js\\API-Academlo-Meals\\node_modules\\express\\lib\\router\\layer.js:95:5)",  5    "err": {  6      "statusCode": 401,  7      "status": "error",  8      "isOperational": true  9    }  }
```

Order status validator evidence

The screenshot displays a REST client interface with a sidebar on the left showing a project structure. The main area is configured for a PATCH request to the endpoint `{{Host}}/orders/10`. The 'Query Params' section is empty. The 'Body' tab is active, showing a JSON response in 'Pretty' format. The response indicates a 404 Not Found status with an error message: 'Order with id: 10 has already been cancelled'. The error stack trace points to a file in the project's source code.

Request Configuration:

- Method: PATCH
- URL: `{{Host}}/orders/10`
- Query Params: Empty table

Key	Value	Description
Key	Value	Description

Response Details:

- Status: 404 Not Found
- Time: 286 ms
- Size: 641 B

JSON Response (Pretty):

```
1 {
2   "status": "error",
3   "message": "Order with id: 10 has already been cancelled",
4   "stack": "Error: Order with id: 10 has already been cancelled\n    at C:\\Academlo\\Node.js\\API-Academlo-Meals\\src\\modules\\orders\\orders.\n      middleware.js:15:21\n    at process.processTicksAndRejections (node:internal/process/task_queues:95:5)",
5   "err": {
6     "statusCode": 404,
7     "status": "error",
8     "isOperational": true
9   }
10 }
```