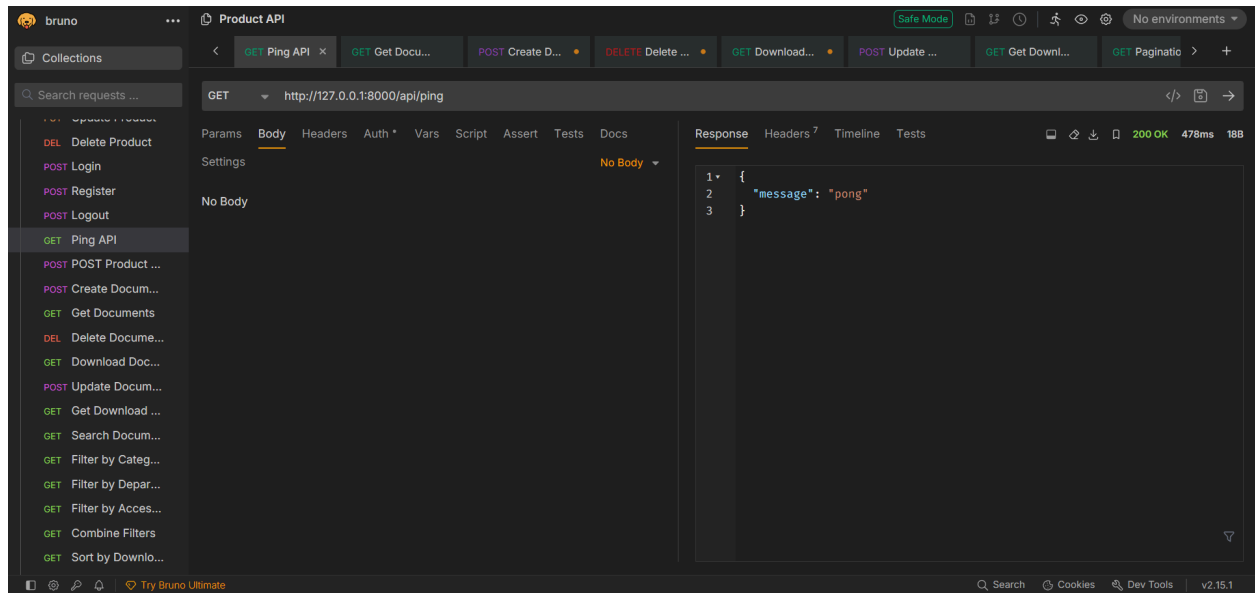


## PING API

Endpoint: GET /api/ping

Function: Check API server is running.



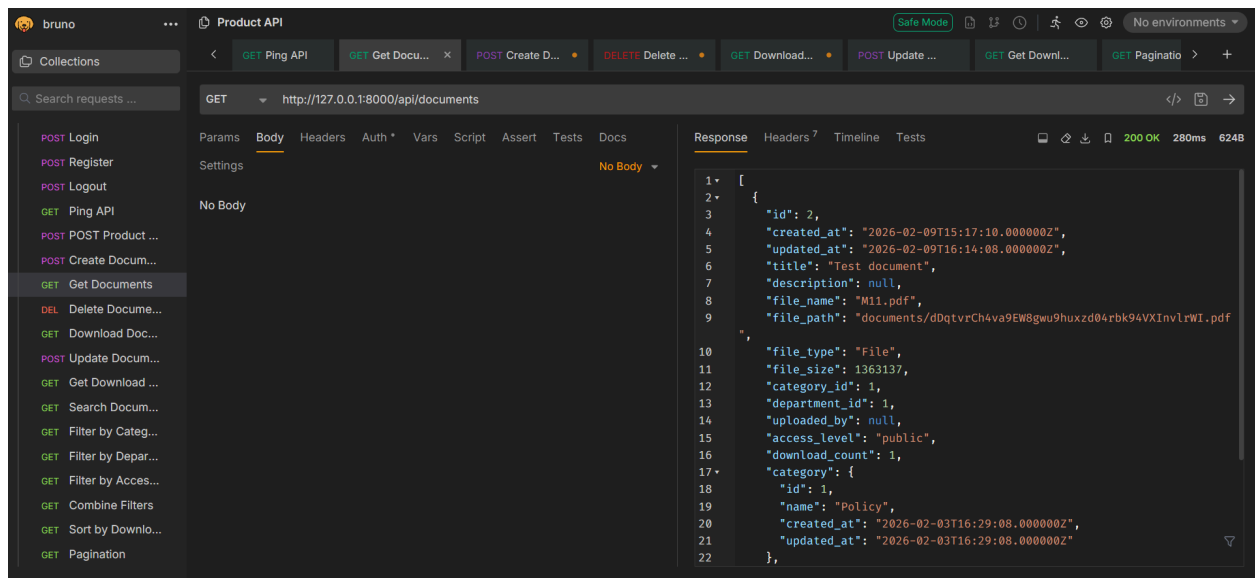
## Get Documents

Endpoint:

GET /api/documents

Function:

Retrieve list of documents.



## Create Document

Endpoint: POST /api/documents

Function: Upload new document.

The screenshot shows the Bruno API client interface. The left sidebar lists various API endpoints, with 'POST Create Document' selected. The main panel displays the details of the POST request to 'http://127.0.0.1:8000/api/documents'. The request body is a multipart form with the following data:

Key	Value	Content-Type
title	Test document	Text
description	test file	Text
file	M11.pdf	File
category_id	1	Text
department_id	1	Text
access_level	public	Text

The response panel shows a 201 Created status with the following JSON body:

```
{
  "title": "Test document",
  "description": null,
  "file_name": "M11.pdf",
  "file_path": "documents/ph4wGyJVAI0mLvccxMZsWsx7jH70ad1wnQKbZMXj.pdf",
  "file_type": "File",
  "file_size": 1363137,
  "category_id": "1",
  "department_id": "1",
  "uploaded_by": null,
  "access_level": "public",
  "download_count": 0,
  "updated_at": "2026-02-09T15:18:44.000000Z",
  "created_at": "2026-02-09T15:18:44.000000Z",
  "id": 3
}
```

## Update Document

Endpoint: POST /api/documents/{id}

Function: Update document information.

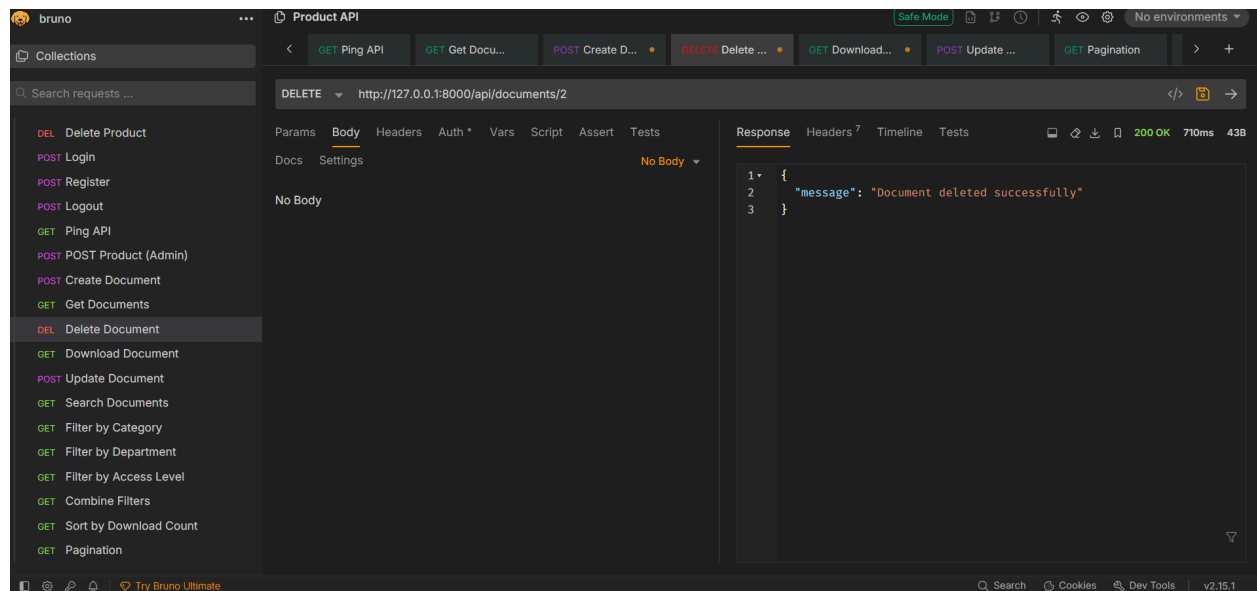
The screenshot shows the Bruno API client interface. The left sidebar lists various API endpoints, with 'GET Get Documents' selected. The main panel displays the details of the GET request to 'http://127.0.0.1:8000/api/documents'. The response panel shows a 200 OK status with the following JSON body:

```
[
  {
    "id": 2,
    "created_at": "2026-02-09T15:17:10.000000Z",
    "updated_at": "2026-02-09T16:14:08.000000Z",
    "title": "Test document",
    "description": null,
    "file_name": "M11.pdf",
    "file_path": "documents/dDqtvrCh4va9EW8gw9huxzd04rbk94VXInvlrWI.pdf",
    "file_type": "File",
    "file_size": 1363137,
    "category_id": 1,
    "department_id": 1,
    "uploaded_by": null,
    "access_level": "public",
    "download_count": 1,
    "category": {
      "id": 1,
      "name": "Policy",
      "created_at": "2026-02-03T16:29:08.000000Z",
      "updated_at": "2026-02-03T16:29:08.000000Z"
    }
  }
]
```

## Delete Document

Endpoint: DELETE /api/documents/{id}

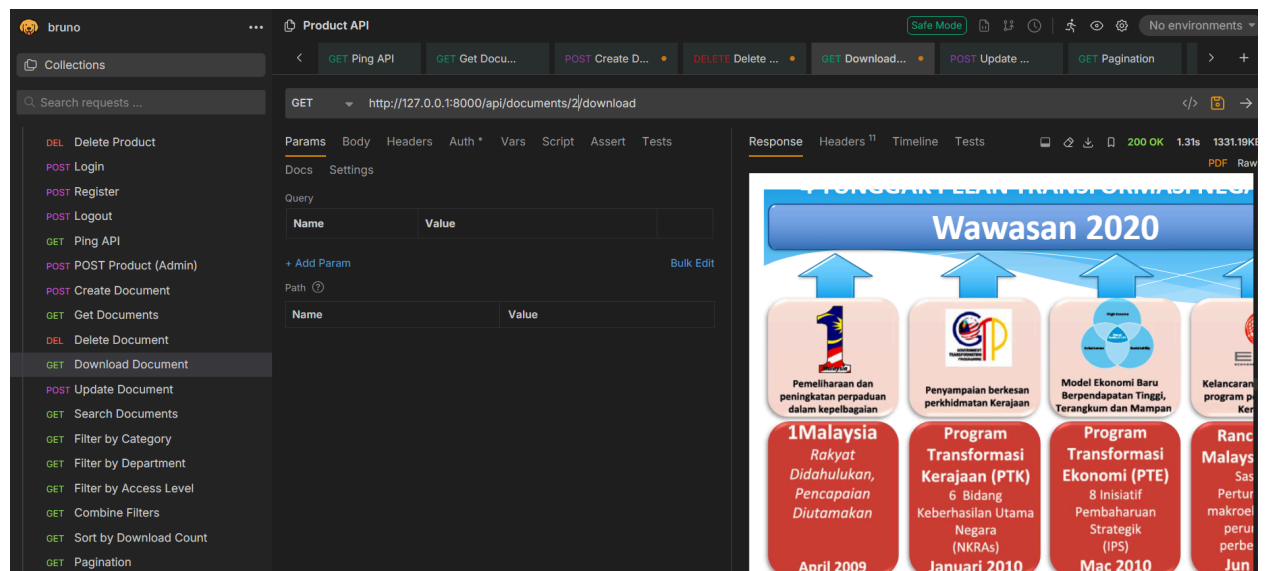
Function: Delete selected document.



## Download Document

Endpoint: GET /api/documents/{id}/download

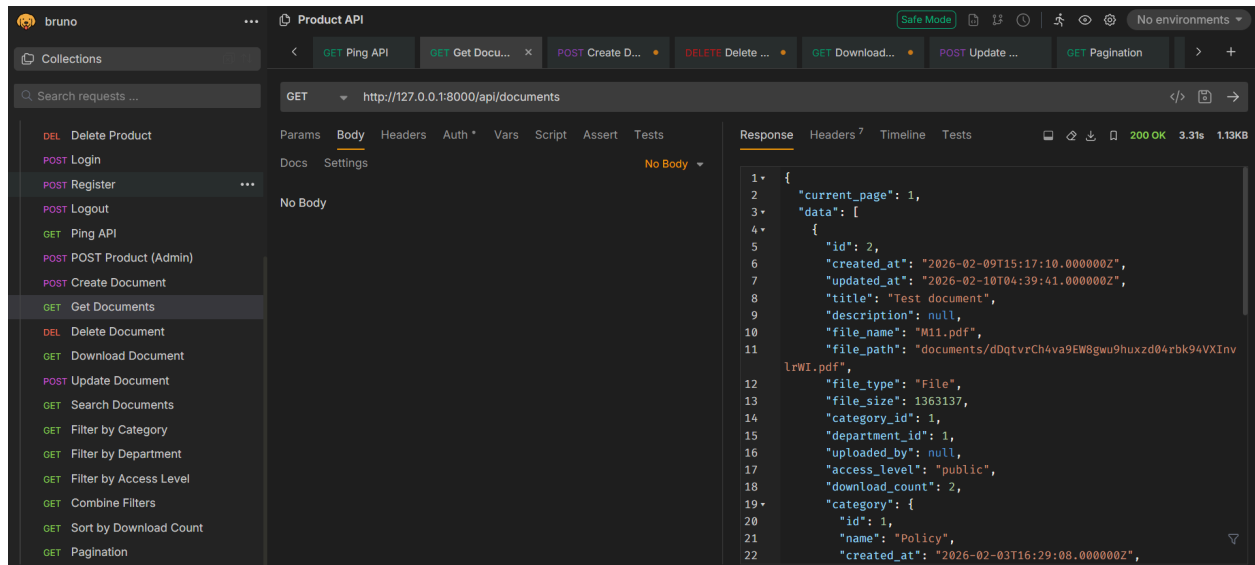
Function: Download document file.



## Download Count

Endpoint: GET /api/documents

Function: Verify download\_count increases after download.



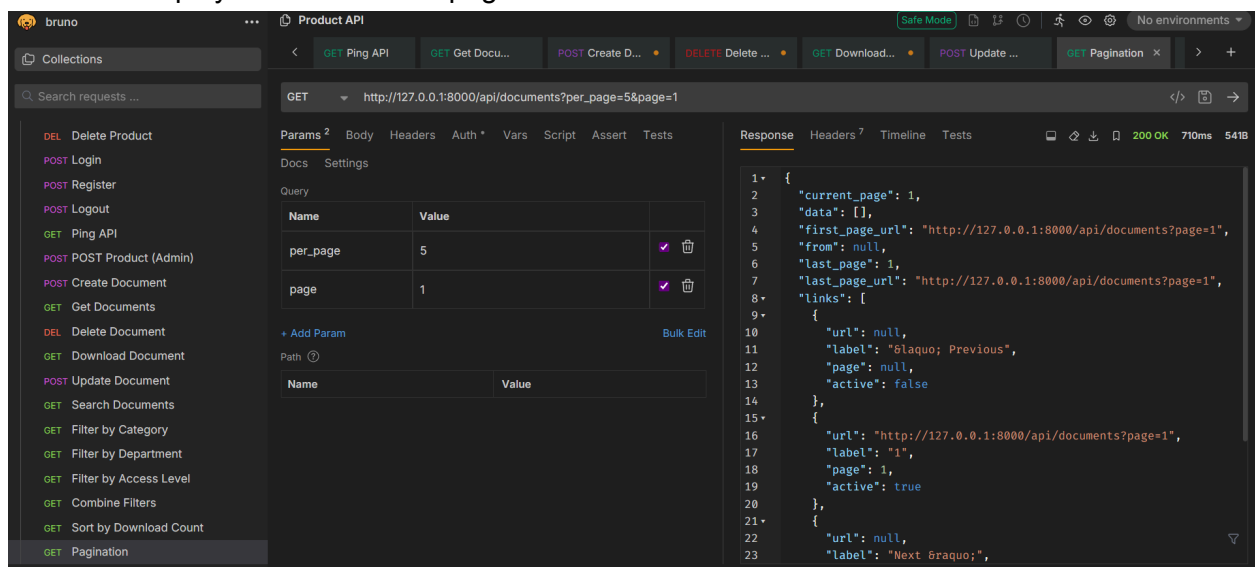
The screenshot shows the Bruno API client interface. The left sidebar lists various API endpoints, with 'GET Get Documents' selected. The main panel displays the details of a GET request to 'http://127.0.0.1:8000/api/documents'. The 'Body' tab is active, showing 'No Body'. The 'Response' tab shows a JSON response with the following structure:

```
1 {
2   "current_page": 1,
3   "data": [
4     {
5       "id": 2,
6       "created_at": "2026-02-09T15:17:10.000000Z",
7       "updated_at": "2026-02-10T04:39:41.000000Z",
8       "title": "Test document",
9       "description": null,
10      "file_name": "M11.pdf",
11      "file_path": "documents/dDqtvCh4va9EW8gwu9huxzd04rbk94VXInv
12      lrWI.pdf",
13      "file_type": "File",
14      "file_size": 1363137,
15      "category_id": 1,
16      "department_id": 1,
17      "uploaded_by": null,
18      "access_level": "public",
19      "download_count": 2,
20      "category": {
21        "id": 1,
22        "name": "Policy",
23        "created_at": "2026-02-03T16:29:08.000000Z",
```

## Pagination

Endpoint: GET /api/documents?per\_page=5&page=1

Function: Display documents with pagination.



The screenshot shows the Bruno API client interface. The left sidebar lists various API endpoints, with 'GET Pagination' selected. The main panel displays the details of a GET request to 'http://127.0.0.1:8000/api/documents?per\_page=5&page=1'. The 'Params' tab is active, showing the query parameters:

Name	Value
per_page	5
page	1

The 'Response' tab shows a JSON response with the following structure:

```
1 {
2   "current_page": 1,
3   "data": [],
4   "first_page_url": "http://127.0.0.1:8000/api/documents?page=1",
5   "from": null,
6   "last_page": 1,
7   "last_page_url": "http://127.0.0.1:8000/api/documents?page=1",
8   "links": [
9     {
10      "url": null,
11      "label": "< Previous",
12      "page": null,
13      "active": false
14    },
15    {
16      "url": "http://127.0.0.1:8000/api/documents?page=1",
17      "label": "1",
18      "page": 1,
19      "active": true
20    },
21    {
22      "url": null,
23      "label": "Next >",
```

## Backend

```
❖ PS C:\Users\User\final-demo\employee-doc-portal-api> php artisan serve
```

```
INFO Server running on [http://127.0.0.1:8000].
```

```
Press Ctrl+C to stop the server
```

## Frontend

```
> ▼ TERMINAL
⚠ PS C:\Users\User\final-demo\employee-doc-portal-api\frontend> npm run dev
VITE v7.3.1 ready in 641 ms
→ Local: http://localhost:5173/
→ Network: use --host to expose
→ press h + enter to show help
```

## Testing

```
> ▼ TERMINAL
⚠ Duration: 13.99s
❖ PS C:\Users\User\final-demo\employee-doc-portal-api> php artisan test

PASS Tests\Unit\ExampleTest
✓ that true is true 0.01s

PASS Tests\Feature\AuthApiTest
✓ user can register 0.69s
✓ user can login 0.04s
✓ user can logout 0.04s

PASS Tests\Feature\DocumentApiTest
✓ manager can upload document 0.07s
✓ employee cannot upload document 0.04s
✓ user can search documents 0.05s
✓ user can filter documents by category 0.04s
✓ admin can delete any document 0.04s

PASS Tests\Feature\ExampleTest
✓ the application returns a successful response 0.05s

Tests: 10 passed (27 assertions)
Duration: 1.26s

❖ PS C:\Users\User\final-demo\employee-doc-portal-api>
```