onyx

Backend APIs › Ingestion API

Backend APIs

Ingestion API

Ingest Arbitrary Documents

Typical uses for the Ingestion API

A set of backend APIs are provided to take in and process arbitrary document data sent directly to the backend API server. This is generally used for:

- Creating arbitrary documents that may not exist in any actual sources but contain useful information.
- Programmatically passing in documents to Onyx. This is sometimes simpler than creating a connector.
- Editing specific documents in Onyx when the Onyx admin either does not want to or does not have permission to update the original document in the source.
- Supplementing existing connector functionalities. For example, passing in README file contents and attributing it to the GitHub or GitLab source type.

Example Document Ingestion

This example creates a new Document in Onyx of the “Web” type. This document will now show up in Onyx’s search flows like any other webpage pulled in by a Web connector.

onyx

```
{
  "document": {
    "id": "ingestion_document_1",
    "sections": [
      {
        "text": "This is the contents of the document that will be processed and saved into the vector+keyword document index. ",
        "link": "https://docs.danswer.dev/introduction#what-is-danswer"
      },
      {
        "text": "You can include multiple content sections each with their own link or combine them. ",
        "link": "https://docs.danswer.dev/introduction#main-features"
      }
    ],
    "source": "web",
    "semantic_identifier": "Danswer Ingestion Example",
    "metadata": {
      "tag": "informational",
      "topics": ["danswer", "api"]
    },
    "doc_updated_at": "2024-04-25T08:20:00Z"
  },
  "cc_pair_id": 1
}
```

Body Cookies Headers (4) Test Results Status: 200 OK Time: 1089 ms Size: 186 B Save as example

Pretty Raw Preview Visualize JSON

```
{
    "document_id": "ingestion_document_1",
    "already_existed": true
}
```

onyx

```
  "document": {
    "id": "ingestion_document_1",
    "sections": [
      {
        "text": "This is the contents of the document that will be processed and saved int
        "link": "https://docs.onyx.app/introduction#what-is-onyx"
      },
      {
        "text": "You can include multiple content sections each with their own link or com
        "link": "https://docs.onyx.app/introduction#main-features"
      }
    ],
    "source": "web",
    "semantic_identifier": "Onyx Ingestion Example",
    "metadata": {
        "tag": "informational",
        "topics": ["onyx", "api"]
    },
    "doc_updated_at": "2024-04-25T08:20:00Z"
  },
  "cc_pair_id": 1
}'
```

> Note: The Bearer auth token is generated on server startup in Onyx MIT. There is better API Key support as part of Onyx EE.

See below for a breakdown of the different fields provided:

`id` : this is the unique ID of the document, if a document of this ID exists it will be updated/replaced. If not provided, a document ID is generated from the semantic_identifier field instead and returned in the response.

`sections` : list of sections each containing textual content and an optional link. The document chunking tries to avoid splitting sections internally and favors splitting at section borders. Also the link of the document at query time is the link of the best matched section.

`source` : Source type, full list can be checked by searching for DocumentSource **here**

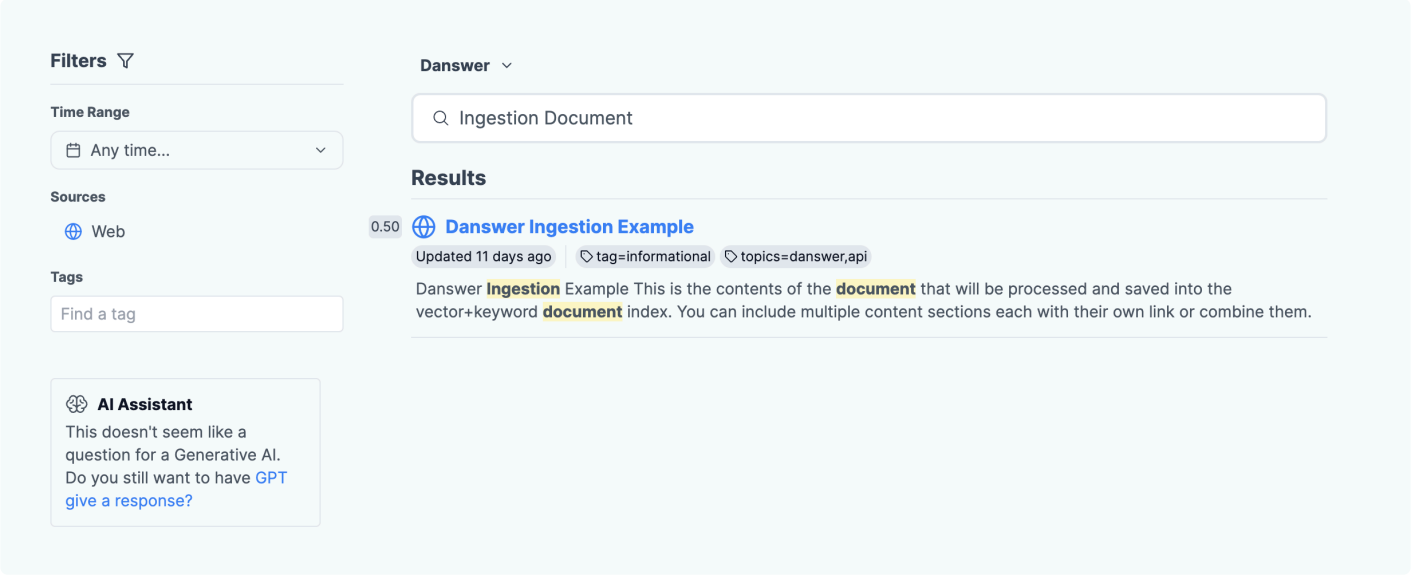
`semantic_identifier` : This is the “Title” of the document as shown in the UI (see image below)

onyx

›

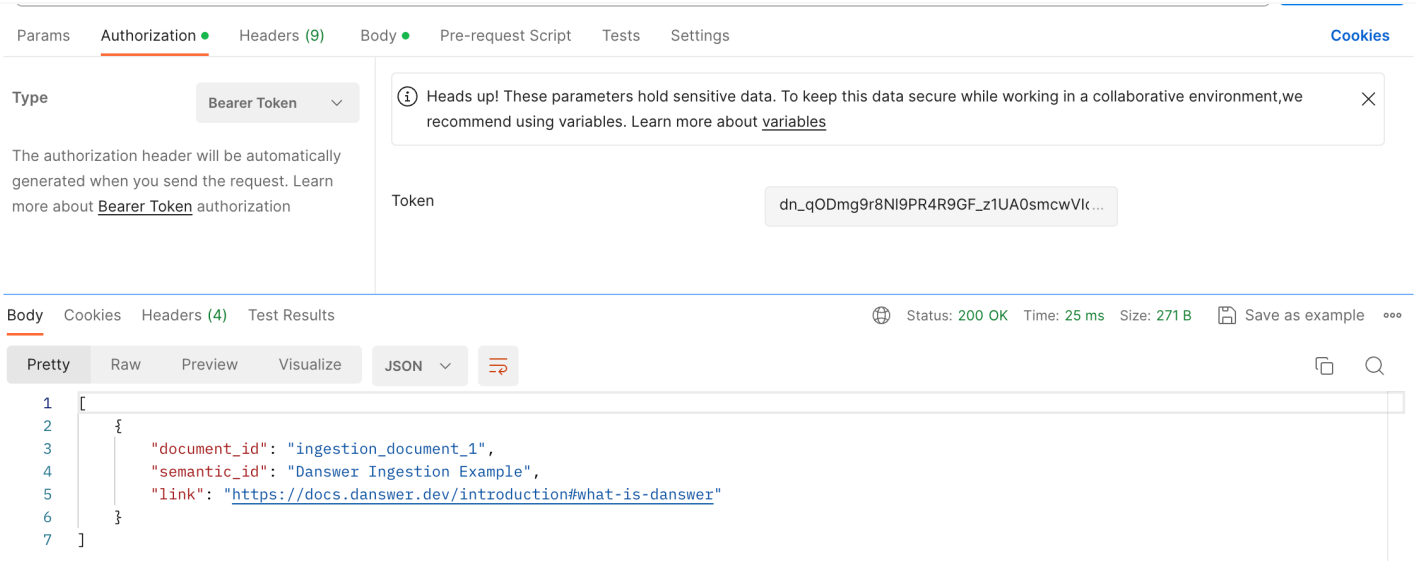
`cc_pair_id` : This is the “Connector” ID seen on the Connector Status pages. For example, if running locally, it might be `http://localhost:3000/admin/connector/2` . This allows attaching the ingestion doc to existing connectors so they can be assigned to groups or deleted together with the connector. If not provided or set to `1` explicitly, it is considered part of the default catch-all connector.

For even more details, the code for the relevant object is found **here**, called “DocumentBase”



Checking Ingestion Documents

onyx



Custom Tools

Answer with Quote