Version: DEV

HTTP API

A complete reference for RAGFlow's RESTful API. Before proceeding, please ensure you have your RAGFlow API key ready for authentication.

OpenAI-Compatible API

Create chat completion

POST `/api/v1/chats_openai/{chat_id}/chat/completions`

Creates a model response for a given chat conversation.

This API follows the same request and response format as OpenAI's API. It allows you to interact with the model in a manner similar to how you would with OpenAI's API.

Request

- Method: POST
- URL: `/api/v1/chats_openai/{chat_id}/chat/completions`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"model"`: `string`
  - `"messages"`: `object list`
  - `"stream"`: `boolean`

Request example

```
curl --request POST \
     --url http://{address}/api/v1/chats_openai/{chat_id}/chat/completions \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '{
        "model": "model",
        "messages": [{"role": "user", "content": "Say this is a test!"}],
        "stream": true
      }'
```

Request Parameters

- `model` (*Body parameter*) `string`, *Required* The model used to generate the response. The server will parse this automatically, so you can set it to any value for now.
- `messages` (*Body parameter*) `list[object]`, *Required* A list of historical chat messages used to generate the response. This must contain at least one message with the `user` role.
- `stream` (*Body parameter*) `boolean` Whether to receive the response as a stream. Set this to `false` explicitly if you prefer to receive the entire response in one go instead of as a stream.

Response

Stream:

```
{
    "id": "chatcmpl-3a9c3572f29311efa69751e139332ced",
    "choices": [
        {
            "delta": {
                "content": "This is a test. If you have any specific questions or
need information, feel",
                "role": "assistant",
                "function_call": null,
                "tool_calls": null
            },
            "finish_reason": null,
            "index": 0,
            "logprobs": null
        }
    ],
    "created": 1740543996,
```

```
    "model": "model",
    "object": "chat.completion.chunk",
    "system_fingerprint": "",
    "usage": null
}
// omit duplicated information
{"choices":[{"delta":{"content":" free to ask, and I will do my best to provide an
answer based on","role":"assistant"}}]}
{"choices":[{"delta":{"content":" the knowledge I have. If your question is unrelated
to the provided knowledge base,","role":"assistant"}}]}
{"choices":[{"delta":{"content":" I will let you know.","role":"assistant"}}]}
// the last chunk
{
    "id": "chatcmpl-3a9c3572f29311efa69751e139332ced",
    "choices": [
        {
            "delta": {
                "content": null,
                "role": "assistant",
                "function_call": null,
                "tool_calls": null
            },
            "finish_reason": "stop",
            "index": 0,
            "logprobs": null
        }
    ],
    "created": 1740543996,
    "model": "model",
    "object": "chat.completion.chunk",
    "system_fingerprint": "",
    "usage": {
        "prompt_tokens": 18,
        "completion_tokens": 225,
        "total_tokens": 243
    }
}
```

Non-stream:

```
{
    "choices":[
        {
            "finish_reason":"stop",
            "index":0,
            "logprobs":null,
            "message":{
                "content":"This is a test. If you have any specific questions or need
information, feel free to ask, and I will do my best to provide an answer based on
```

```
the knowledge I have. If your question is unrelated to the provided knowledge base, I
will let you know.",
                "role":"assistant"
            }
        }
    ],
    "created":1740543499,
    "id":"chatcmpl-3a9c3572f29311efa69751e139332ced",
    "model":"model",
    "object":"chat.completion",
    "usage":{
        "completion_tokens":246,
        "completion_tokens_details":{
            "accepted_prediction_tokens":246,
            "reasoning_tokens":18,
            "rejected_prediction_tokens":0
        },
        "prompt_tokens":18,
        "total_tokens":264
    }
}
```

Failure:

```
{
  "code": 102,
  "message": "The last content of this conversation is not from user."
}
```

DATASET MANAGEMENT

Create dataset

POST `/api/v1/datasets`

Creates a dataset.

Request

- Method: POST
- URL: `/api/v1/datasets`

- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"name"`: `string`
  - `"avatar"`: `string`
  - `"description"`: `string`
  - `"embedding_model"`: `string`
  - `"permission"`: `string`
  - `"chunk_method"`: `string`
  - `"parser_config"`: `object`

Request example

```
curl --request POST \
     --url http://{address}/api/v1/datasets \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '{
      "name": "test_1"
      }'
```

Request parameters

- `"name"`: (*Body parameter*), `string`, *Required*
  The unique name of the dataset to create. It must adhere to the following requirements:
  - Permitted characters include:
    - English letters (a-z, A-Z)
    - Digits (0-9)
    - "_" (underscore)
  - Must begin with an English letter or underscore.
  - Maximum 65,535 characters.
  - Case-insensitive.

- `"avatar"`: (*Body parameter*), `string`
  Base64 encoding of the avatar.

- `"description"`: (*Body parameter*), `string`
  A brief description of the dataset to create.

- `"embedding_model"`: (*Body parameter*), `string`
  The name of the embedding model to use. For example: `"BAAI/bge-zh-v1.5"`

- `"permission"`: (*Body parameter*), `string`
  Specifies who can access the dataset to create. Available options:

  - `"me"`: (Default) Only you can manage the dataset.
  - `"team"`: All team members can manage the dataset.

- `"chunk_method"`: (*Body parameter*), `enum<string>`
  The chunking method of the dataset to create. Available options:

  - `"naive"`: General (default)
  - `"manual"`: Manual
  - `"qa"`: Q&A
  - `"table"`: Table
  - `"paper"`: Paper
  - `"book"`: Book
  - `"laws"`: Laws
  - `"presentation"`: Presentation
  - `"picture"`: Picture
  - `"one"`: One
  - `"knowledge_graph"`: Knowledge Graph
    Ensure your LLM is properly configured on the **Settings** page before selecting this. Please also note that Knowledge Graph consumes a large number of Tokens!
  - `"email"`: Email

- `"parser_config"`: (*Body parameter*), `object`
  The configuration settings for the dataset parser. The attributes in this JSON object vary with the selected `"chunk_method"`:

- If `"chunk_method"` is `"naive"`, the `"parser_config"` object contains the following attributes:
  - `"chunk_token_count"`: Defaults to `128`.
  - `"layout_recognize"`: Defaults to `true`.
  - `"html4excel"`: Indicates whether to convert Excel documents into HTML format. Defaults to `false`.
  - `"delimiter"`: Defaults to `"\n!?。；！？"`.
  - `"task_page_size"`: Defaults to `12`. For PDF only.
  - `"raptor"`: Raptor-specific settings. Defaults to: `{"use_raptor": false}`.
- If `"chunk_method"` is `"qa"`, `"manuel"`, `"paper"`, `"book"`, `"laws"`, or `"presentation"`, the `"parser_config"` object contains the following attribute:
  - `"raptor"`: Raptor-specific settings. Defaults to: `{"use_raptor": false}`.
- If `"chunk_method"` is `"table"`, `"picture"`, `"one"`, or `"email"`, `"parser_config"` is an empty JSON object.
- If `"chunk_method"` is `"knowledge_graph"`, the `"parser_config"` object contains the following attributes:
  - `"chunk_token_count"`: Defaults to `128`.
  - `"delimiter"`: Defaults to `"\n!?。；！？"`.
  - `"entity_types"`: Defaults to `["organization","person","location","event","time"]`

Response

Success:


```
{
    "code": 0,
    "data": {
        "avatar": null,
        "chunk_count": 0,
        "chunk_method": "naive",
        "create_date": "Thu, 24 Oct 2024 09:14:07 GMT",
        "create_time": 1729761247434,
        "created_by": "69736c5e723611efb51b0242ac120007",
        "description": null,
        "document_count": 0,
        "embedding_model": "BAAI/bge-large-zh-v1.5",
        "id": "527fa74891e811ef9c650242ac120006",
        "language": "English",
```

```
        "name": "test_1",
        "parser_config": {
            "chunk_token_num": 128,
            "delimiter": "\\n!?;。；！？",
            "html4excel": false,
            "layout_recognize": true,
            "raptor": {
                "user_raptor": false
            }
        },
        "permission": "me",
        "similarity_threshold": 0.2,
        "status": "1",
        "tenant_id": "69736c5e723611efb51b0242ac120007",
        "token_num": 0,
        "update_date": "Thu, 24 Oct 2024 09:14:07 GMT",
        "update_time": 1729761247434,
        "vector_similarity_weight": 0.3
    }
}
```


Failure:


```
{
    "code": 102,
    "message": "Duplicated knowledgebase name in creating dataset."
}
```


Delete datasets

DELETE `/api/v1/datasets`

Deletes datasets by ID.

Request

- Method: DELETE
- URL: `/api/v1/datasets`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`

- Body:
  - `"ids"`: `list[string]`

Request example

```
curl --request DELETE \
     --url http://{address}/api/v1/datasets \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '{
     "ids": ["test_1", "test_2"]
     }'
```

Request parameters

- `"ids"`: (*Body parameter*), `list[string]`
  The IDs of the datasets to delete. If it is not specified, all datasets will be deleted.

Response

Success:

```
{
    "code": 0
}
```

Failure:

```
{
    "code": 102,
    "message": "You don't own the dataset."
}
```

Update dataset

PUT `/api/v1/datasets/{dataset_id}`

Updates configurations for a specified dataset.

Request

- Method: PUT
- URL: `/api/v1/datasets/{dataset_id}`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"name"`: `string`
  - `"embedding_model"`: `string`
  - `"chunk_method"`: `enum<string>`

Request example

```
curl --request PUT \
     --url http://{address}/api/v1/datasets/{dataset_id} \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '
     {
          "name": "updated_dataset"
     }'
```

Request parameters

- `dataset_id`: (*Path parameter*)
  The ID of the dataset to update.
- `"name"`: (*Body parameter*), `string`
  The revised name of the dataset.
- `"embedding_model"`: (*Body parameter*), `string`
  The updated embedding model name.
  - Ensure that `"chunk_count"` is `0` before updating `"embedding_model"`.
- `"chunk_method"`: (*Body parameter*), `enum<string>`
  The chunking method for the dataset. Available options:
  - `"naive"`: General
  - `"manual"`: Manual
  - `"qa"`: Q&A

- `"table"`: Table
- `"paper"`: Paper
- `"book"`: Book
- `"laws"`: Laws
- `"presentation"`: Presentation
- `"picture"`: Picture
- `"one"`:One
- `"email"`: Email
- `"knowledge_graph"`: Knowledge Graph
  Ensure your LLM is properly configured on the **Settings** page before selecting this. Please also note that Knowledge Graph consumes a large number of Tokens!

Response

Success:

```
{
    "code": 0
}
```

Failure:

```
{
    "code": 102,
    "message": "Can't change tenant_id."
}
```

List datasets

GET `/api/v1/datasets?page={page}&page_size={page_size}&orderby={orderby}&desc={desc}&name={dataset_name}&id={dataset_id}`

Lists datasets.

Request

- Method: GET
- URL: `/api/v1/datasets?page={page}&page_size={page_size}&orderby={orderby}&desc={desc}&name={dataset_name}&id={dataset_id}`
- Headers:
  - `'Authorization: Bearer <YOUR_API_KEY>'`

Request example

```
curl --request GET \
     --url http://{address}/api/v1/datasets?page={page}&page_size={page_size}
&orderby={orderby}&desc={desc}&name={dataset_name}&id={dataset_id} \
     --header 'Authorization: Bearer <YOUR_API_KEY>'
```

Request parameters

- `page`: (*Filter parameter*)
  Specifies the page on which the datasets will be displayed. Defaults to `1`.
- `page_size`: (*Filter parameter*)
  The number of datasets on each page. Defaults to `30`.
- `orderby`: (*Filter parameter*)
  The field by which datasets should be sorted. Available options:
  - `create_time` (default)
  - `update_time`
- `desc`: (*Filter parameter*)
  Indicates whether the retrieved datasets should be sorted in descending order. Defaults to `true`.
- `name`: (*Filter parameter*)
  The name of the dataset to retrieve.
- `id`: (*Filter parameter*)
  The ID of the dataset to retrieve.

Response

Success:

```
{
    "code": 0,
```

```
    "data": [
        {
            "avatar": "",
            "chunk_count": 59,
            "create_date": "Sat, 14 Sep 2024 01:12:37 GMT",
            "create_time": 1726276357324,
            "created_by": "69736c5e723611efb51b0242ac120007",
            "description": null,
            "document_count": 1,
            "embedding_model": "BAAI/bge-large-zh-v1.5",
            "id": "6e211ee0723611efa10a0242ac120007",
            "language": "English",
            "name": "mysql",
            "chunk_method": "knowledge_graph",
            "parser_config": {
                "chunk_token_num": 8192,
                "delimiter": "\\n!?;。；！？",
                "entity_types": [
                    "organization",
                    "person",
                    "location",
                    "event",
                    "time"
                ]
            },
            "permission": "me",
            "similarity_threshold": 0.2,
            "status": "1",
            "tenant_id": "69736c5e723611efb51b0242ac120007",
            "token_num": 12744,
            "update_date": "Thu, 10 Oct 2024 04:07:23 GMT",
            "update_time": 1728533243536,
            "vector_similarity_weight": 0.3
        }
    ]
}
```


Failure:


```
{
    "code": 102,
    "message": "The dataset doesn't exist"
}
```

FILE MANAGEMENT WITHIN DATASET

Upload documents

POST `/api/v1/datasets/{dataset_id}/documents`

Uploads documents to a specified dataset.

Request

- Method: POST
- URL: `/api/v1/datasets/{dataset_id}/documents`
- Headers:
  - `'Content-Type: multipart/form-data'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Form:
  - `'file=@{FILE_PATH}'`

Request example

```
curl --request POST \
     --url http://{address}/api/v1/datasets/{dataset_id}/documents \
     --header 'Content-Type: multipart/form-data' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --form 'file=@./test1.txt' \
     --form 'file=@./test2.pdf'
```

Request parameters

- `dataset_id`: (*Path parameter*)
  The ID of the dataset to which the documents will be uploaded.
- `'file'`: (*Body parameter*)
  A document to upload.

Response

Success:

```
{
    "code": 0,
    "data": [
        {
            "chunk_method": "naive",
            "created_by": "69736c5e723611efb51b0242ac120007",
            "dataset_id": "527fa74891e811ef9c650242ac120006",
            "id": "b330ec2e91ec11efbc510242ac120004",
            "location": "1.txt",
            "name": "1.txt",
            "parser_config": {
                "chunk_token_num": 128,
                "delimiter": "\\n!?;。；！？",
                "html4excel": false,
                "layout_recognize": true,
                "raptor": {
                    "user_raptor": false
                }
            },
            "run": "UNSTART",
            "size": 17966,
            "thumbnail": "",
            "type": "doc"
        }
    ]
}
```


Failure:


```
{
    "code": 101,
    "message": "No file part!"
}
```


Update document

PUT `/api/v1/datasets/{dataset_id}/documents/{document_id}`

Updates configurations for a specified document.

Request

- Method: PUT

- URL: `/api/v1/datasets/{dataset_id}/documents/{document_id}`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"name"`:`string`
  - `"meta_fields"`:`object`
  - `"chunk_method"`:`string`
  - `"parser_config"`:`object`

Request example

```
curl --request PUT \
     --url http://{address}/api/v1/datasets/{dataset_id}/info/{document_id} \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --header 'Content-Type: application/json' \
     --data '
     {
          "name": "manual.txt",
          "chunk_method": "manual",
          "parser_config": {"chunk_token_count": 128}
     }'
```

Request parameters

- `dataset_id`: (*Path parameter*)
  The ID of the associated dataset.
- `document_id`: (*Path parameter*)
  The ID of the document to update.
- `"name"`: (*Body parameter*), `string`
- `"meta_fields"`: (*Body parameter*), `dict[str, Any]` The meta fields of the document.
- `"chunk_method"`: (*Body parameter*), `string`
  The parsing method to apply to the document:
  - `"naive"`: General
  - `"manual`: Manual
  - `"qa"`: Q&A

  - `"table"`: Table
  - `"paper"`: Paper
  - `"book"`: Book
  - `"laws"`: Laws
  - `"presentation"`: Presentation
  - `"picture"`: Picture
  - `"one"`: One
  - `"email"`: Email
- `"parser_config"`: (*Body parameter*), `object`
  The configuration settings for the dataset parser. The attributes in this JSON object vary with the selected `"chunk_method"`:
  - If `"chunk_method"` is `"naive"`, the `"parser_config"` object contains the following attributes:
    - `"chunk_token_count"`: Defaults to `128`.
    - `"layout_recognize"`: Defaults to `true`.
    - `"html4excel"`: Indicates whether to convert Excel documents into HTML format. Defaults to `false`.
    - `"delimiter"`: Defaults to `"\n!?。；！？"`.
    - `"task_page_size"`: Defaults to `12`. For PDF only.
    - `"raptor"`: Raptor-specific settings. Defaults to: `{"use_raptor": false}`.
  - If `"chunk_method"` is `"qa"`, `"manuel"`, `"paper"`, `"book"`, `"laws"`, or `"presentation"`, the `"parser_config"` object contains the following attribute:
    - `"raptor"`: Raptor-specific settings. Defaults to: `{"use_raptor": false}`.
  - If `"chunk_method"` is `"table"`, `"picture"`, `"one"`, or `"email"`, `"parser_config"` is an empty JSON object.
  - If `"chunk_method"` is `"knowledge_graph"`, the `"parser_config"` object contains the following attributes:
    - `"chunk_token_count"`: Defaults to `128`.
    - `"delimiter"`: Defaults to `"\n!?。；！？"`.
    - `"entity_types"`: Defaults to `["organization","person","location","event","time"]`

Response

Success:

```
{
    "code": 0
}
```

Failure:

```
{
    "code": 102,
    "message": "The dataset does not have the document."
}
```

Download document

GET `/api/v1/datasets/{dataset_id}/documents/{document_id}`

Downloads a document from a specified dataset.

Request

- Method: GET
- URL: `/api/v1/datasets/{dataset_id}/documents/{document_id}`
- Headers:
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Output:
  - `'{PATH_TO_THE_FILE}'`

Request example

```
curl --request GET \
     --url http://{address}/api/v1/datasets/{dataset_id}/documents/{document_id} \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --output ./ragflow.txt
```

Request parameters

- `dataset_id`: (*Path parameter*)
  The associated dataset ID.
- `documents_id`: (*Path parameter*)
  The ID of the document to download.

Response

Success:

```
This is a test to verify the file download feature.
```

Failure:

```
{
    "code": 102,
    "message": "You do not own the dataset 7898da028a0511efbf750242ac1220005."
}
```

List documents

GET `/api/v1/datasets/{dataset_id}/documents?page={page}&page_size={page_size}&orderby={orderby}&desc={desc}&keywords={keywords}&id={document_id}&name={document_name}`

Lists documents in a specified dataset.

Request

- Method: GET
- URL: `/api/v1/datasets/{dataset_id}/documents?page={page}&page_size={page_size}&orderby={orderby}&desc={desc}&keywords={keywords}&id={document_id}&name={document_name}`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`

Request example

```
curl --request GET \
     --url http://{address}/api/v1/datasets/{dataset_id}/documents?page={page}
&page_size={page_size}&orderby={orderby}&desc={desc}&keywords={keywords}
&id={document_id}&name={document_name} \
     --header 'Authorization: Bearer <YOUR_API_KEY>'
```

Request parameters

- `dataset_id`: (*Path parameter*)
  The associated dataset ID.
- `keywords`: (*Filter parameter*), `string`
  The keywords used to match document titles.
- `page`: (*Filter parameter*), `integer` Specifies the page on which the documents will be displayed. Defaults to `1`.
- `page_size`: (*Filter parameter*), `integer`
  The maximum number of documents on each page. Defaults to `30`.
- `orderby`: (*Filter parameter*), `string`
  The field by which documents should be sorted. Available options:
  - `create_time` (default)
  - `update_time`
- `desc`: (*Filter parameter*), `boolean`
  Indicates whether the retrieved documents should be sorted in descending order. Defaults to `true`.
- `id`: (*Filter parameter*), `string`
  The ID of the document to retrieve.

Response

Success:

```
{
    "code": 0,
    "data": {
        "docs": [
            {
                "chunk_count": 0,
                "create_date": "Mon, 14 Oct 2024 09:11:01 GMT",
                "create_time": 1728897061948,
```

```
                "created_by": "69736c5e723611efb51b0242ac120007",
                "id": "3bcfbf8a8a0c11ef8aba0242ac120006",
                "knowledgebase_id": "7898da028a0511efbf750242ac120005",
                "location": "Test_2.txt",
                "name": "Test_2.txt",
                "parser_config": {
                    "chunk_token_count": 128,
                    "delimiter": "\n!?。；！？",
                    "layout_recognize": true,
                    "task_page_size": 12
                },
                "chunk_method": "naive",
                "process_begin_at": null,
                "process_duation": 0.0,
                "progress": 0.0,
                "progress_msg": "",
                "run": "0",
                "size": 7,
                "source_type": "local",
                "status": "1",
                "thumbnail": null,
                "token_count": 0,
                "type": "doc",
                "update_date": "Mon, 14 Oct 2024 09:11:01 GMT",
                "update_time": 1728897061948
            }
        ],
        "total": 1
    }
}
```


Failure:


```
{
    "code": 102,
    "message": "You don't own the dataset 7898da028a0511efbf750242ac1220005. "
}
```


Delete documents

DELETE `/api/v1/datasets/{dataset_id}/documents`

Deletes documents by ID.

Request

- Method: DELETE
- URL: `/api/v1/datasets/{dataset_id}/documents`
- Headers:
  - `'Content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"ids"`: `list[string]`

Request example

```
curl --request DELETE \
     --url http://{address}/api/v1/datasets/{dataset_id}/documents \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '
     {
          "ids": ["id_1","id_2"]
     }'
```

Request parameters

- `dataset_id`: (*Path parameter*)
  The associated dataset ID.
- `"ids"`: (*Body parameter*), `list[string]`
  The IDs of the documents to delete. If it is not specified, all documents in the specified dataset will be deleted.

Response

Success:

```
{
    "code": 0
}.
```

Failure:

```
{
    "code": 102,
    "message": "You do not own the dataset 7898da028a0511efbf750242ac1220005."
}
```

Parses documents in a specified dataset.

Request

- Method: POST
- URL: `/api/v1/datasets/{dataset_id}/chunks`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"document_ids"`: `list[string]`

Request example

```
curl --request POST \
     --url http://{address}/api/v1/datasets/{dataset_id}/chunks \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '
     {
          "document_ids":
["97a5f1c2759811efaa500242ac120004","97ad64b6759811ef9fc30242ac120004"]
     }'
```

Request parameters

- `dataset_id`: (*Path parameter*)
  The dataset ID.
- `"document_ids"`: (*Body parameter*), `list[string]`, *Required*
  The IDs of the documents to parse.

Response

Success:

```
{
    "code": 0
}
```

Failure:

```
{
    "code": 102,
    "message": "`document_ids` is required"
}
```

Stop parsing documents

DELETE `/api/v1/datasets/{dataset_id}/chunks`

Stops parsing specified documents.

Request

- Method: DELETE
- URL: `/api/v1/datasets/{dataset_id}/chunks`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"document_ids"`: `list[string]`

Request example

```
curl --request DELETE \
     --url http://{address}/api/v1/datasets/{dataset_id}/chunks \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '
     {
```

```
         "document_ids":
["97a5f1c2759811efaa500242ac120004","97ad64b6759811ef9fc30242ac120004"]
     }'
```

Request parameters

- `dataset_id`: (*Path parameter*)
  The associated dataset ID.
- `"document_ids"`: (*Body parameter*), `list[string]`, *Required*
  The IDs of the documents for which the parsing should be stopped.

Response

Success:

```
{
    "code": 0
}
```

Failure:

```
{
    "code": 102,
    "message": "`document_ids` is required"
}
```

CHUNK MANAGEMENT WITHIN DATASET

Add chunk

POST `/api/v1/datasets/{dataset_id}/documents/{document_id}/chunks`

Adds a chunk to a specified document in a specified dataset.

Request

- Method: POST
- URL: `/api/v1/datasets/{dataset_id}/documents/{document_id}/chunks`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"content"`: `string`
  - `"important_keywords"`: `list[string]`

Request example

```
curl --request POST \
     --url http://{address}/api/v1/datasets/{dataset_id}/documents/{document_id}/
chunks \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '
     {
          "content": "<CHUNK_CONTENT_HERE>"
     }'
```

Request parameters

- `dataset_id`: (*Path parameter*)
  The associated dataset ID.
- `document_ids`: (*Path parameter*)
  The associated document ID.
- `"content"`: (*Body parameter*), `string`, *Required*
  The text content of the chunk.
- `"important_keywords`(*Body parameter*), `list[string]`
  The key terms or phrases to tag with the chunk.
- `"questions"`(*Body parameter*), `list[string]` If there is a given question, the embedded chunks will be based on them

Response

Success:

```
{
    "code": 0,
```

```
{
    "code": 102,
    "message": "`content` is required"
}
```

```
            "dataset_id": "72f3be1ebdf411efb7250242ac120006",
            "document_id": "61d68474be0111ef98dd0242ac120006",
            "id": "12ccdc56e59837e5",
            "important_keywords": [],
            "questions": []
        }
    }
}
```

Lists chunks in a specified document.

Request

- Method: GET
- URL: `/api/v1/datasets/{dataset_id}/documents/{document_id}/chunks?keywords={keywords}&page={page}&page_size={page_size}&id={chunk_id}`
- Headers:
  - `'Authorization: Bearer <YOUR_API_KEY>'`

Request example

```
curl --request GET \
     --url http://{address}/api/v1/datasets/{dataset_id}/documents/{document_id}/
chunks?keywords={keywords}&page={page}&page_size={page_size}&id={chunk_id} \
     --header 'Authorization: Bearer <YOUR_API_KEY>'
```

Request parameters

- `dataset_id`: (*Path parameter*)
  The associated dataset ID.
- `document_ids`: (*Path parameter*)
  The associated document ID.

- `keywords` (*Filter parameter*), `string`
  The keywords used to match chunk content.
- `page` (*Filter parameter*), `integer`
  Specifies the page on which the chunks will be displayed. Defaults to `1`.
- `page_size` (*Filter parameter*), `integer`
  The maximum number of chunks on each page. Defaults to `1024`.
- `id` (*Filter parameter*), `string`
  The ID of the chunk to retrieve.

Response

Success:


```
{
    "code": 0,
    "data": {
        "chunks": [
            {
                "available_int": 1,
                "content": "This is a test content.",
                "docnm_kwd": "1.txt",
                "document_id": "b330ec2e91ec11efbc510242ac120004",
                "id": "b48c170e90f70af998485c1065490726",
                "image_id": "",
                "important_keywords": "",
                "positions": [
                    ""
                ]
            }
        ],
        "doc": {
            "chunk_count": 1,
            "chunk_method": "naive",
            "create_date": "Thu, 24 Oct 2024 09:45:27 GMT",
            "create_time": 1729763127646,
            "created_by": "69736c5e723611efb51b0242ac120007",
            "dataset_id": "527fa74891e811ef9c650242ac120006",
            "id": "b330ec2e91ec11efbc510242ac120004",
            "location": "1.txt",
            "name": "1.txt",
            "parser_config": {
                "chunk_token_num": 128,
                "delimiter": "\\n!?;。；！？",
                "html4excel": false,
                "layout_recognize": true,
```

```
                "raptor": {
                    "user_raptor": false
                }
            },
            "process_begin_at": "Thu, 24 Oct 2024 09:56:44 GMT",
            "process_duation": 0.54213,
            "progress": 0.0,
            "progress_msg": "Task dispatched...",
            "run": "2",
            "size": 17966,
            "source_type": "local",
            "status": "1",
            "thumbnail": "",
            "token_count": 8,
            "type": "doc",
            "update_date": "Thu, 24 Oct 2024 11:03:15 GMT",
            "update_time": 1729767795721
        },
        "total": 1
    }
}
```


Failure:


```
{
    "code": 102,
    "message": "You don't own the document 5c5999ec7be811ef9cab0242ac12000e5."
}
```


Delete chunks

DELETE `/api/v1/datasets/{dataset_id}/documents/{document_id}/chunks`

Deletes chunks by ID.

Request

- Method: DELETE
- URL: `/api/v1/datasets/{dataset_id}/documents/{document_id}/chunks`
- Headers:
  - `'content-Type: application/json'`

    - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
    - `"chunk_ids"`: `list[string]`

Request example

```
curl --request DELETE \
     --url http://{address}/api/v1/datasets/{dataset_id}/documents/{document_id}/
chunks \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '
     {
          "chunk_ids": ["test_1", "test_2"]
     }'
```

Request parameters

- `dataset_id`: (*Path parameter*)
  The associated dataset ID.
- `document_ids`: (*Path parameter*)
  The associated document ID.
- `"chunk_ids"`: (*Body parameter*), `list[string]`
  The IDs of the chunks to delete. If it is not specified, all chunks of the specified document will be deleted.

Response

Success:

```
{
    "code": 0
}
```

Failure:

Update chunk

```
{
    "code": 102,
    "message": "`chunk_ids` is required"
}
```

Request

- Method: PUT
- URL: `/api/v1/datasets/{dataset_id}/documents/{document_id}/chunks/{chunk_id}`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"content"`: `string`
  - `"important_keywords"`: `list[string]`
  - `"available"`: `boolean`

Request example

```
curl --request PUT \
     --url http://{address}/api/v1/datasets/{dataset_id}/documents/{document_id}/
chunks/{chunk_id} \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '
     {
          "content": "ragflow123",
          "important_keywords": []
     }'
```

Request parameters

- `dataset_id`: (*Path parameter*)
  The associated dataset ID.
- `document_ids`: (*Path parameter*)
  The associated document ID.
- `chunk_id`: (*Path parameter*)
  The ID of the chunk to update.
- `"content"`: (*Body parameter*), `string`

The text content of the chunk.

- `"important_keywords"`: (*Body parameter*), `list[string]`
  A list of key terms or phrases to tag with the chunk.
- `"available"`: (*Body parameter*) `boolean`
  The chunk's availability status in the dataset. Value options:
  - `true`: Available (default)
  - `false`: Unavailable

Response

Success:

```
{
    "code": 0
}
```

Failure:

```
{
    "code": 102,
    "message": "Can't find this chunk 29a2d9987e16ba331fb4d7d30d99b71d2"
}
```

Retrieve chunks

POST `/api/v1/retrieval`

Retrieves chunks from specified datasets.

Request

- Method: POST
- URL: `/api/v1/retrieval`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`

- Body:
  - `"question"`: `string`
  - `"dataset_ids"`: `list[string]`
  - `"document_ids"`: `list[string]`
  - `"page"`: `integer`
  - `"page_size"`: `integer`
  - `"similarity_threshold"`: `float`
  - `"vector_similarity_weight"`: `float`
  - `"top_k"`: `integer`
  - `"rerank_id"`: `string`
  - `"keyword"`: `boolean`
  - `"highlight"`: `boolean`

Request example

```
curl --request POST \
     --url http://{address}/api/v1/retrieval \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '
     {
          "question": "What is advantage of ragflow?",
          "dataset_ids": ["b2a62730759d11ef987d0242ac120004"],
          "document_ids": ["77df9ef4759a11ef8bdd0242ac120004"]
     }'
```

Request parameter

- `"question"`: (*Body parameter*), `string`, *Required*
  The user query or query keywords.
- `"dataset_ids"`: (*Body parameter*) `list[string]`
  The IDs of the datasets to search. If you do not set this argument, ensure that you set `"document_ids"`.
- `"document_ids"`: (*Body parameter*), `list[string]`
  The IDs of the documents to search. Ensure that all selected documents use the same embedding model. Otherwise, an error will occur. If you do not set this argument, ensure that you set `"dataset_ids"`.

- `"page"`: (*Body parameter*), `integer`
  Specifies the page on which the chunks will be displayed. Defaults to `1`.
- `"page_size"`: (*Body parameter*)
  The maximum number of chunks on each page. Defaults to `30`.
- `"similarity_threshold"`: (*Body parameter*)
  The minimum similarity score. Defaults to `0.2`.
- `"vector_similarity_weight"`: (*Body parameter*), `float`
  The weight of vector cosine similarity. Defaults to `0.3`. If x represents the weight of vector cosine similarity, then (1 - x) is the term similarity weight.
- `"top_k"`: (*Body parameter*), `integer`
  The number of chunks engaged in vector cosine computation. Defaults to `1024`.
- `"rerank_id"`: (*Body parameter*), `integer`
  The ID of the rerank model.
- `"keyword"`: (*Body parameter*), `boolean`
  Indicates whether to enable keyword-based matching:
  - `true`: Enable keyword-based matching.
  - `false`: Disable keyword-based matching (default).
- `"highlight"`: (*Body parameter*), `boolean`
  Specifies whether to enable highlighting of matched terms in the results:
  - `true`: Enable highlighting of matched terms.
  - `false`: Disable highlighting of matched terms (default).

Response

Success:


```
{
    "code": 0,
    "data": {
        "chunks": [
            {
                "content": "ragflow content",
                "content_ltks": "ragflow content",
                "document_id": "5c5999ec7be811ef9cab0242ac120005",
                "document_keyword": "1.txt",
                "highlight": "<em>ragflow</em> content",
                "id": "d78435d142bd5cf6704da62c778795c5",
                "image_id": "",
```

```
                "important_keywords": [
                    ""
                ],
                "kb_id": "c7ee74067a2c11efb21c0242ac120006",
                "positions": [
                    ""
                ],
                "similarity": 0.9669436601210759,
                "term_similarity": 1.0,
                "vector_similarity": 0.8898122004035864
            }
        ],
        "doc_aggs": [
            {
                "count": 1,
                "doc_id": "5c5999ec7be811ef9cab0242ac120005",
                "doc_name": "1.txt"
            }
        ],
        "total": 1
    }
}
```


Failure:


```
{
    "code": 102,
    "message": "`datasets` is required."
}
```


CHAT ASSISTANT MANAGEMENT

Create chat assistant

POST `/api/v1/chats`

Creates a chat assistant.

Request

- Method: POST
- URL: `/api/v1/chats`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"name"`: `string`
  - `"avatar"`: `string`
  - `"dataset_ids"`: `list[string]`
  - `"llm"`: `object`
  - `"prompt"`: `object`

Request example

```
curl --request POST \
     --url http://{address}/api/v1/chats \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '{
    "dataset_ids": ["0b2cbc8c877f11ef89070242ac120005"],
    "name":"new_chat_1"
}'
```

Request parameters

- `"name"`: (*Body parameter*), `string`, *Required*
  The name of the chat assistant.
- `"avatar"`: (*Body parameter*), `string`
  Base64 encoding of the avatar.
- `"dataset_ids"`: (*Body parameter*), `list[string]`
  The IDs of the associated datasets.
- `"llm"`: (*Body parameter*), `object`
  The LLM settings for the chat assistant to create. If it is not explicitly set, a JSON object with the following values will be generated as the default. An `llm` JSON object contains the following attributes:
  - `"model_name"`, `string`

The chat model name. If not set, the user's default chat model will be used.

- `"temperature"`: `float`
  Controls the randomness of the model's predictions. A lower temperature results in more conservative responses, while a higher temperature yields more creative and diverse responses. Defaults to `0.1`.
- `"top_p"`: `float`
  Also known as “nucleus sampling”, this parameter sets a threshold to select a smaller set of words to sample from. It focuses on the most likely words, cutting off the less probable ones. Defaults to `0.3`
- `"presence_penalty"`: `float`
  This discourages the model from repeating the same information by penalizing words that have already appeared in the conversation. Defaults to `0.2`.
- `"frequency penalty"`: `float`
  Similar to the presence penalty, this reduces the model’s tendency to repeat the same words frequently. Defaults to `0.7`.
- `"max_token"`: `integer`
  The maximum length of the model's output, measured in the number of tokens (words or pieces of words). Defaults to `512`. If disabled, you lift the maximum token limit, allowing the model to determine the number of tokens in its responses.

- `"prompt"`: (*Body parameter*), `object`
  Instructions for the LLM to follow. If it is not explicitly set, a JSON object with the following values will be generated as the default. A `prompt` JSON object contains the following attributes:
  - `"similarity_threshold"`: `float` RAGFlow employs either a combination of weighted keyword similarity and weighted vector cosine similarity, or a combination of weighted keyword similarity and weighted reranking score during retrieval. This argument sets the threshold for similarities between the user query and chunks. If a similarity score falls below this threshold, the corresponding chunk will be excluded from the results. The default value is `0.2`.
  - `"keywords_similarity_weight"`: `float` This argument sets the weight of keyword similarity in the hybrid similarity score with vector cosine similarity or reranking model similarity. By adjusting this weight, you can control the influence of keyword similarity in relation to other similarity measures. The default value is `0.7`.
  - `"top_n"`: `int` This argument specifies the number of top chunks with similarity

scores above the `similarity_threshold` that are fed to the LLM. The LLM will *only* access these 'top N' chunks. The default value is `8`.

- `"variables"`: `object[]` This argument lists the variables to use in the 'System' field of **Chat Configurations**. Note that:
  - `"knowledge"` is a reserved variable, which represents the retrieved chunks.
  - All the variables in 'System' should be curly bracketed.
  - The default value is `[{"key": "knowledge", "optional": true}]`.
- `"rerank_model"`: `string` If it is not specified, vector cosine similarity will be used; otherwise, reranking score will be used.
- `top_k`: `int` Refers to the process of reordering or selecting the top-k items from a list or set based on a specific ranking criterion. Default to 1024.
- `"empty_response"`: `string` If nothing is retrieved in the dataset for the user's question, this will be used as the response. To allow the LLM to improvise when nothing is found, leave this blank.
- `"opener"`: `string` The opening greeting for the user. Defaults to `"Hi! I am your assistant, can I help you?"`.
- `"show_quote"`: `boolean` Indicates whether the source of text should be displayed. Defaults to `true`.
- `"prompt"`: `string` The prompt content.

Response

Success:


```
{
    "code": 0,
    "data": {
        "avatar": "",
        "create_date": "Thu, 24 Oct 2024 11:18:29 GMT",
        "create_time": 1729768709023,
        "dataset_ids": [
            "527fa74891e811ef9c650242ac120006"
        ],
        "description": "A helpful Assistant",
        "do_refer": "1",
        "id": "b1f2f15691f911ef81180242ac120003",
        "language": "English",
        "llm": {
            "frequency_penalty": 0.7,
```

```
            "max_tokens": 512,
            "model_name": "qwen-plus@Tongyi-Qianwen",
            "presence_penalty": 0.4,
            "temperature": 0.1,
            "top_p": 0.3
        },
        "name": "12234",
        "prompt": {
            "empty_response": "Sorry! No relevant content was found in the knowledge
base!",
            "keywords_similarity_weight": 0.3,
            "opener": "Hi! I'm your assistant, what can I do for you?",
            "prompt": "You are an intelligent assistant. Please summarize the content
of the knowledge base to answer the question. Please list the data in the knowledge
base and answer in detail. When all knowledge base content is irrelevant to the
question, your answer must include the sentence \"The answer you are looking for is
not found in the knowledge base!\" Answers need to consider chat history.\n ",
            "rerank_model": "",
            "similarity_threshold": 0.2,
            "top_n": 6,
            "variables": [
                {
                    "key": "knowledge",
                    "optional": false
                }
            ]
        },
        "prompt_type": "simple",
        "status": "1",
        "tenant_id": "69736c5e723611efb51b0242ac120007",
        "top_k": 1024,
        "update_date": "Thu, 24 Oct 2024 11:18:29 GMT",
        "update_time": 1729768709023
    }
}
```


Failure:


```
{
    "code": 102,
    "message": "Duplicated chat name in creating dataset."
}
```


Update chat assistant

PUT `/api/v1/chats/{chat_id}`

Updates configurations for a specified chat assistant.

Request

- Method: PUT
- URL: `/api/v1/chats/{chat_id}`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"name"`: `string`
  - `"avatar"`: `string`
  - `"dataset_ids"`: `list[string]`
  - `"llm"`: `object`
  - `"prompt"`: `object`

Request example

```
curl --request PUT \
     --url http://{address}/api/v1/chats/{chat_id} \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '
     {
          "name":"Test"
     }'
```

Parameters

- `chat_id`: (*Path parameter*)
  The ID of the chat assistant to update.
- `"name"`: (*Body parameter*), `string`, *Required*
  The revised name of the chat assistant.
- `"avatar"`: (*Body parameter*), `string`
  Base64 encoding of the avatar.

- `"dataset_ids"`: (*Body parameter*), `list[string]`
  The IDs of the associated datasets.
- `"llm"`: (*Body parameter*), `object`
  The LLM settings for the chat assistant to create. If it is not explicitly set, a dictionary with the following values will be generated as the default. An `llm` object contains the following attributes:
  - `"model_name"`, `string`
    The chat model name. If not set, the user's default chat model will be used.
  - `"temperature"`: `float`
    Controls the randomness of the model's predictions. A lower temperature results in more conservative responses, while a higher temperature yields more creative and diverse responses. Defaults to `0.1`.
  - `"top_p"`: `float`
    Also known as “nucleus sampling”, this parameter sets a threshold to select a smaller set of words to sample from. It focuses on the most likely words, cutting off the less probable ones. Defaults to `0.3`
  - `"presence_penalty"`: `float`
    This discourages the model from repeating the same information by penalizing words that have already appeared in the conversation. Defaults to `0.2`.
  - `"frequency penalty"`: `float`
    Similar to the presence penalty, this reduces the model’s tendency to repeat the same words frequently. Defaults to `0.7`.
  - `"max_token"`: `integer`
    The maximum length of the model's output, measured in the number of tokens (words or pieces of words). Defaults to `512`. If disabled, you lift the maximum token limit, allowing the model to determine the number of tokens in its responses.
- `"prompt"`: (*Body parameter*), `object`
  Instructions for the LLM to follow. A `prompt` object contains the following attributes:
  - `"similarity_threshold"`: `float` RAGFlow employs either a combination of weighted keyword similarity and weighted vector cosine similarity, or a combination of weighted keyword similarity and weighted rerank score during retrieval. This argument sets the threshold for similarities between the user query and chunks. If a similarity score falls below this threshold, the corresponding chunk will be excluded from the results. The default value is `0.2`.
  - `"keywords_similarity_weight"`: `float` This argument sets the weight of keyword

similarity in the hybrid similarity score with vector cosine similarity or reranking model similarity. By adjusting this weight, you can control the influence of keyword similarity in relation to other similarity measures. The default value is `0.7`.

- `"top_n"`: `int` This argument specifies the number of top chunks with similarity scores above the `similarity_threshold` that are fed to the LLM. The LLM will *only* access these 'top N' chunks. The default value is `8`.
- `"variables"`: `object[]` This argument lists the variables to use in the 'System' field of **Chat Configurations**. Note that:
  - `"knowledge"` is a reserved variable, which represents the retrieved chunks.
  - All the variables in 'System' should be curly bracketed.
  - The default value is `[{"key": "knowledge", "optional": true}]`
- `"rerank_model"`: `string` If it is not specified, vector cosine similarity will be used; otherwise, reranking score will be used.
- `"empty_response"`: `string` If nothing is retrieved in the dataset for the user's question, this will be used as the response. To allow the LLM to improvise when nothing is found, leave this blank.
- `"opener"`: `string` The opening greeting for the user. Defaults to `"Hi! I am your assistant, can I help you?"`.
- `"show_quote`: `boolean` Indicates whether the source of text should be displayed. Defaults to `true`.
- `"prompt"`: `string` The prompt content.

Response

Success:

```
{
    "code": 0
}
```

Failure:

```
{
    "code": 102,
    "message": "Duplicated chat name in updating dataset."
```

```
}
```

Delete chat assistants

DELETE `/api/v1/chats`

Deletes chat assistants by ID.

Request

- Method: DELETE
- URL: `/api/v1/chats`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"ids"`: `list[string]`

Request example

```
curl --request DELETE \
     --url http://{address}/api/v1/chats \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '
     {
          "ids": ["test_1", "test_2"]
     }'
```

Request parameters

- `"ids"`: (*Body parameter*), `list[string]`
  The IDs of the chat assistants to delete. If it is not specified, all chat assistants in the system will be deleted.

Response

Success:

```
{
    "code": 0
}
```

Failure:

```
{
    "code": 102,
    "message": "ids are required"
}
```

List chat assistants

GET `/api/v1/chats?page={page}&page_size={page_size}&orderby={orderby}&desc={desc}&name={chat_name}&id={chat_id}`

Lists chat assistants.

Request

- Method: GET
- URL: `/api/v1/chats?page={page}&page_size={page_size}&orderby={orderby}&desc={desc}&name={dataset_name}&id={dataset_id}`
- Headers:
  - `'Authorization: Bearer <YOUR_API_KEY>'`

Request example

```
curl --request GET \
     --url http://{address}/api/v1/chats?page={page}&page_size={page_size}
&orderby={orderby}&desc={desc}&name={dataset_name}&id={dataset_id} \
     --header 'Authorization: Bearer <YOUR_API_KEY>'
```

Request parameters

- `page`: (*Filter parameter*), `integer`
  Specifies the page on which the chat assistants will be displayed. Defaults to `1`.

- `page_size`: (*Filter parameter*), `integer`
  The number of chat assistants on each page. Defaults to `30`.
- `orderby`: (*Filter parameter*), `string`
  The attribute by which the results are sorted. Available options:
  - `create_time` (default)
  - `update_time`
- `desc`: (*Filter parameter*), `boolean`
  Indicates whether the retrieved chat assistants should be sorted in descending order. Defaults to `true`.
- `id`: (*Filter parameter*), `string`
  The ID of the chat assistant to retrieve.
- `name`: (*Filter parameter*), `string`
  The name of the chat assistant to retrieve.

Response

Success:


```
{
    "code": 0,
    "data": [
        {
            "avatar": "",
            "create_date": "Fri, 18 Oct 2024 06:20:06 GMT",
            "create_time": 1729232406637,
            "description": "A helpful Assistant",
            "do_refer": "1",
            "id": "04d0d8e28d1911efa3630242ac120006",
            "dataset_ids": ["527fa74891e811ef9c650242ac120006"],
            "language": "English",
            "llm": {
                "frequency_penalty": 0.7,
                "max_tokens": 512,
                "model_name": "qwen-plus@Tongyi-Qianwen",
                "presence_penalty": 0.4,
                "temperature": 0.1,
                "top_p": 0.3
            },
            "name": "13243",
            "prompt": {
                "empty_response": "Sorry! No relevant content was found in the
knowledge base!",
                "keywords_similarity_weight": 0.3,
```

```
                "opener": "Hi! I'm your assistant, what can I do for you?",
                "prompt": "You are an intelligent assistant. Please summarize the
content of the knowledge base to answer the question. Please list the data in the
knowledge base and answer in detail. When all knowledge base content is irrelevant to
the question, your answer must include the sentence \"The answer you are looking for
is not found in the knowledge base!\" Answers need to consider chat history.\n",
                "rerank_model": "",
                "similarity_threshold": 0.2,
                "top_n": 6,
                "variables": [
                    {
                        "key": "knowledge",
                        "optional": false
                    }
                ]
            },
            "prompt_type": "simple",
            "status": "1",
            "tenant_id": "69736c5e723611efb51b0242ac120007",
            "top_k": 1024,
            "update_date": "Fri, 18 Oct 2024 06:20:06 GMT",
            "update_time": 1729232406638
        }
    ]
}
```


Failure:


```
{
    "code": 102,
    "message": "The chat doesn't exist"
}
```


SESSION MANAGEMENT

Create session with chat assistant

POST `/api/v1/chats/{chat_id}/sessions`

Creates a session with a chat assistant.

Request

- Method: POST
- URL: `/api/v1/chats/{chat_id}/sessions`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"name"`: `string`
  - `"user_id"`: `string` (optional)

Request example

```
curl --request POST \
     --url http://{address}/api/v1/chats/{chat_id}/sessions \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '
     {
          "name": "new session"
     }'
```

Request parameters

- `chat_id`: (*Path parameter*)
  The ID of the associated chat assistant.
- `"name"`: (*Body parameter*), `string`
  The name of the chat session to create.
- `"user_id"`: (*Body parameter*), `string`
  Optional user-defined ID.

Response

Success:

```
{
    "code": 0,
    "data": {
        "chat_id": "2ca4b22e878011ef88fe0242ac120005",
```

```
        "create_date": "Fri, 11 Oct 2024 08:46:14 GMT",
        "create_time": 1728636374571,
        "id": "4606b4ec87ad11efbc4f0242ac120006",
        "messages": [
            {
                "content": "Hi! I am your assistant, can I help you?",
                "role": "assistant"
            }
        ],
        "name": "new session",
        "update_date": "Fri, 11 Oct 2024 08:46:14 GMT",
        "update_time": 1728636374571
    }
}
```

Failure:

```
{
    "code": 102,
    "message": "Name cannot be empty."
}
```

Update chat assistant's session

PUT `/api/v1/chats/{chat_id}/sessions/{session_id}`

Updates a session of a specified chat assistant.

Request

- Method: PUT
- URL: `/api/v1/chats/{chat_id}/sessions/{session_id}`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"name"`: `string`
  - `"user_id"`: `string` (optional)

Request example

```
curl --request PUT \
     --url http://{address}/api/v1/chats/{chat_id}/sessions/{session_id} \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '
     {
          "name": "<REVISED_SESSION_NAME_HERE>"
     }'
```

Request Parameter

- `chat_id`: (*Path parameter*)
  The ID of the associated chat assistant.
- `session_id`: (*Path parameter*)
  The ID of the session to update.
- `"name"`: (*Body Parameter*), `string`
  The revised name of the session.
- `"user_id"`: (*Body parameter*), `string`
  Optional user-defined ID.

Response

Success:

```
{
    "code": 0
}
```

Failure:

```
{
    "code": 102,
    "message": "Name cannot be empty."
}
```

List chat assistant's sessions

GET `/api/v1/chats/{chat_id}/sessions?page={page}&page_size={page_size}&orderby={orderby}&desc={desc}&name={session_name}&id={session_id}`

Lists sessions associated with a specified chat assistant.

Request

- Method: GET
- URL: `/api/v1/chats/{chat_id}/sessions?page={page}&page_size={page_size}&orderby={orderby}&desc={desc}&name={session_name}&id={session_id}&user_id={user_id}`
- Headers:
  - `'Authorization: Bearer <YOUR_API_KEY>'`

Request example

```
curl --request GET \
     --url http://{address}/api/v1/chats/{chat_id}/sessions?page={page}
&page_size={page_size}&orderby={orderby}&desc={desc}&name={session_name}
&id={session_id} \
     --header 'Authorization: Bearer <YOUR_API_KEY>'
```

Request Parameters

- `chat_id`: (*Path parameter*)
  The ID of the associated chat assistant.
- `page`: (*Filter parameter*), `integer`
  Specifies the page on which the sessions will be displayed. Defaults to `1`.
- `page_size`: (*Filter parameter*), `integer`
  The number of sessions on each page. Defaults to `30`.
- `orderby`: (*Filter parameter*), `string`
  The field by which sessions should be sorted. Available options:
  - `create_time` (default)
  - `update_time`
- `desc`: (*Filter parameter*), `boolean`
  Indicates whether the retrieved sessions should be sorted in descending order.

Defaults to `true`.

- `name`: (*Filter parameter*) `string`
  The name of the chat session to retrieve.
- `id`: (*Filter parameter*), `string`
  The ID of the chat session to retrieve.
- `user_id`: (*Filter parameter*), `string`
  The optional user-defined ID passed in when creating session.

Response

Success:

```
{
    "code": 0,
    "data": [
        {
            "chat": "2ca4b22e878011ef88fe0242ac120005",
            "create_date": "Fri, 11 Oct 2024 08:46:43 GMT",
            "create_time": 1728636403974,
            "id": "578d541e87ad11ef96b90242ac120006",
            "messages": [
                {
                    "content": "Hi! I am your assistant, can I help you?",
                    "role": "assistant"
                }
            ],
            "name": "new session",
            "update_date": "Fri, 11 Oct 2024 08:46:43 GMT",
            "update_time": 1728636403974
        }
    ]
}
```

Failure:

```
{
    "code": 102,
    "message": "The session doesn't exist"
}
```

Delete chat assistant's sessions

DELETE `/api/v1/chats/{chat_id}/sessions`

Deletes sessions of a chat assistant by ID.

Request

- Method: DELETE
- URL: `/api/v1/chats/{chat_id}/sessions`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"ids"`: `list[string]`

Request example

```
curl --request DELETE \
     --url http://{address}/api/v1/chats/{chat_id}/sessions \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '
     {
          "ids": ["test_1", "test_2"]
     }'
```

Request Parameters

- `chat_id`: (*Path parameter*)
  The ID of the associated chat assistant.
- `"ids"`: (*Body Parameter*), `list[string]`
  The IDs of the sessions to delete. If it is not specified, all sessions associated with the specified chat assistant will be deleted.

Response

Success:

```
{
    "code": 0
```

```
{
    "code": 102,
    "message": "The chat doesn't own the session"
}
```

Converse with chat assistant

POST `/api/v1/chats/{chat_id}/completions`

Asks a specified chat assistant a question to start an AI-powered conversation.

> **NOTE**
>
> - In streaming mode, not all responses include a reference, as this depends on the system's judgement.
> - In streaming mode, the last message is an empty message:
>
> ```
> data:
> {
>   "code": 0,
>   "data": true
> }
> ```

Request

- Method: POST
- URL: `/api/v1/chats/{chat_id}/completions`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"question"`: `string`

- `"stream"`: `boolean`
- `"session_id"`: `string` (optional)
- `"user_id`: `string` (optional)

Request example

```
curl --request POST \
     --url http://{address}/api/v1/chats/{chat_id}/completions \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data-binary '
     {
     }'
```

```
curl --request POST \
     --url http://{address}/api/v1/chats/{chat_id}/completions \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data-binary '
     {
          "question": "Who are you",
          "stream": true,
          "session_id":"9fa7691cb85c11ef9c5f0242ac120005"
     }'
```

Request Parameters

- `chat_id`: (*Path parameter*)
  The ID of the associated chat assistant.
- `"question"`: (*Body Parameter*), `string`, *Required*
  The question to start an AI-powered conversation.
- `"stream"`: (*Body Parameter*), `boolean`
  Indicates whether to output responses in a streaming way:
  - `true`: Enable streaming (default).
  - `false`: Disable streaming.
- `"session_id"`: (*Body Parameter*)
  The ID of session. If it is not provided, a new session will be generated.
- `"user_id"`: (*Body parameter*), `string`
  The optional user-defined ID. Valid *only* when no `session_id` is provided.

Response

Success without `session_id`:

```
data:{
    "code": 0,
    "message": "",
    "data": {
        "answer": "Hi! I'm your assistant, what can I do for you?",
        "reference": {},
        "audio_binary": null,
        "id": null,
        "session_id": "b01eed84b85611efa0e90242ac120005"
    }
}
data:{
    "code": 0,
    "message": "",
    "data": true
}
```

Success with `session_id`:

```
data:{
    "code": 0,
    "data": {
        "answer": "I am an intelligent assistant designed to help answer questions by
summarizing content from a",
        "reference": {},
        "audio_binary": null,
        "id": "a84c5dd4-97b4-4624-8c3b-974012c8000d",
        "session_id": "82b0ab2a9c1911ef9d870242ac120006"
    }
}
data:{
    "code": 0,
    "data": {
        "answer": "I am an intelligent assistant designed to help answer questions by
summarizing content from a knowledge base. My responses are based on the information
available in the knowledge base and",
        "reference": {},
        "audio_binary": null,
        "id": "a84c5dd4-97b4-4624-8c3b-974012c8000d",
        "session_id": "82b0ab2a9c1911ef9d870242ac120006"
    }
}
data:{
```

```
    "code": 0,
    "data": {
        "answer": "I am an intelligent assistant designed to help answer questions by
summarizing content from a knowledge base. My responses are based on the information
available in the knowledge base and any relevant chat history.",
        "reference": {},
        "audio_binary": null,
        "id": "a84c5dd4-97b4-4624-8c3b-974012c8000d",
        "session_id": "82b0ab2a9c1911ef9d870242ac120006"
    }
}
data:{
    "code": 0,
    "data": {
        "answer": "I am an intelligent assistant designed to help answer questions by
summarizing content from a knowledge base ##0$$. My responses are based on the
information available in the knowledge base and any relevant chat history.",
        "reference": {
            "total": 1,
            "chunks": [
                {
                    "id": "faf26c791128f2d5e821f822671063bd",
                    "content": "xxxxxxxx",
                    "document_id": "dd58f58e888511ef89c90242ac120006",
                    "document_name": "1.txt",
                    "dataset_id": "8e83e57a884611ef9d760242ac120006",
                    "image_id": "",
                    "similarity": 0.7,
                    "vector_similarity": 0.0,
                    "term_similarity": 1.0,
                    "positions": [
                        ""
                    ]
                }
            ],
            "doc_aggs": [
                {
                    "doc_name": "1.txt",
                    "doc_id": "dd58f58e888511ef89c90242ac120006",
                    "count": 1
                }
            ]
        },
        "prompt": "xxxxxxxxxxx",
        "id": "a84c5dd4-97b4-4624-8c3b-974012c8000d",
        "session_id": "82b0ab2a9c1911ef9d870242ac120006"
    }
}
data:{
    "code": 0,
    "data": true
```

```
}
```

Failure:

```
{
    "code": 102,
    "message": "Please input your question."
}
```

Create session with agent

POST `/api/v1/agents/{agent_id}/sessions`

Creates a session with an agent.

Request

- Method: POST
- URL: `/api/v1/agents/{agent_id}/sessions?user_id={user_id}`
- Headers:
  - `'content-Type: application/json' or 'multipart/form-data'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - the required parameters: `str`
  - other parameters: The parameters specified in the **Begin** component.

Request example

If the **Begin** component in your agent does not take required parameters:

```
curl --request POST \
     --url http://{address}/api/v1/agents/{agent_id}/sessions \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '{
     }'
```

If the **Begin** component in your agent takes required parameters:

```
curl --request POST \
     --url http://{address}/api/v1/agents/{agent_id}/sessions \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data '{
            "lang":"Japanese",
            "file":"Who are you"
     }'
```

If the **Begin** component in your agent takes required file parameters:

```
curl --request POST \
     --url http://{address}/api/v1/agents/{agent_id}/sessions?user_id={user_id} \
     --header 'Content-Type: multipart/form-data' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --form '<FILE_KEY>=@./test1.png'
```

Request parameters

- `agent_id`: (*Path parameter*)
  The ID of the associated agent.
- `user_id`: (*Filter parameter*) The optional user-defined ID for parsing docs (especially images) when creating a session while uploading files.

Response

Success:

```
{
    "code": 0,
    "data": {
        "agent_id": "b4a39922b76611efaa1a0242ac120006",
        "dsl": {
            "answer": [],
            "components": {
                "Answer:GreenReadersDrum": {
                    "downstream": [],
                    "obj": {
                        "component_name": "Answer",
                        "inputs": [],
                        "output": null,
```

```
                "params": {}
            },
            "upstream": []
        },
        "begin": {
            "downstream": [],
            "obj": {
                "component_name": "Begin",
                "inputs": [],
                "output": {},
                "params": {}
            },
            "upstream": []
        }
    },
    "embed_id": "",
    "graph": {
        "edges": [],
        "nodes": [
            {
                "data": {
                    "label": "Begin",
                    "name": "begin"
                },
                "dragging": false,
                "height": 44,
                "id": "begin",
                "position": {
                    "x": 53.25688640427177,
                    "y": 198.37155679786412
                },
                "positionAbsolute": {
                    "x": 53.25688640427177,
                    "y": 198.37155679786412
                },
                "selected": false,
                "sourcePosition": "left",
                "targetPosition": "right",
                "type": "beginNode",
                "width": 200
            },
            {
                "data": {
                    "form": {},
                    "label": "Answer",
                    "name": "dialog_0"
                },
                "dragging": false,
                "height": 44,
                "id": "Answer:GreenReadersDrum",
                "position": {
```

```
                            "x": 360.43473114516974,
                            "y": 207.29298425089348
                        },
                        "positionAbsolute": {
                            "x": 360.43473114516974,
                            "y": 207.29298425089348
                        },
                        "selected": false,
                        "sourcePosition": "right",
                        "targetPosition": "left",
                        "type": "logicNode",
                        "width": 200
                    }
                ]
            },
            "history": [],
            "messages": [],
            "path": [
                [
                    "begin"
                ],
                []
            ],
            "reference": []
        },
        "id": "2581031eb7a311efb5200242ac120005",
        "message": [
            {
                "content": "Hi! I'm your smart assistant. What can I do for you?",
                "role": "assistant"
            }
        ],
        "source": "agent",
        "user_id": "69736c5e723611efb51b0242ac120007"
    }
}
```


Failure:


```
{
    "code": 102,
    "message": "Agent not found."
}
```


Converse with agent

POST `/api/v1/agents/{agent_id}/completions`

Asks a specified agent a question to start an AI-powered conversation.

> **NOTE**
>
> - In streaming mode, not all responses include a reference, as this depends on the system's judgement.
> - In streaming mode, the last message is an empty message:
>
> ```
> data:
> {
>   "code": 0,
>   "data": true
> }
> ```

Request

- Method: POST
- URL: `/api/v1/agents/{agent_id}/completions`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"question"`: `string`
  - `"stream"`: `boolean`
  - `"session_id"`: `string`
  - `"user_id"`: `string` (optional)
  - other parameters: `string`

Request example

If the **Begin** component does not take parameters, the following code will create a session.

```
curl --request POST \
     --url http://{address}/api/v1/agents/{agent_id}/completions \
     --header 'Content-Type: application/json' \
```

```
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data-binary '
     {
     }'
```

If the **Begin** component takes parameters, the following code will create a session.

```
curl --request POST \
     --url http://{address}/api/v1/agents/{agent_id}/completions \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data-binary '
     {
          "lang":"English",
          "file":"How is the weather tomorrow?"
     }'
```

The following code will execute the completion process

```
curl --request POST \
     --url http://{address}/api/v1/agents/{agent_id}/completions \
     --header 'Content-Type: application/json' \
     --header 'Authorization: Bearer <YOUR_API_KEY>' \
     --data-binary '
     {
          "question": "Hello",
          "stream": true,
          "session_id": "cb2f385cb86211efa36e0242ac120005"
     }'
```

Request Parameters

- `agent_id`: (*Path parameter*), `string`
  The ID of the associated agent.
- `"question"`: (*Body Parameter*), `string`, *Required*
  The question to start an AI-powered conversation.
- `"stream"`: (*Body Parameter*), `boolean`
  Indicates whether to output responses in a streaming way:
  - `true`: Enable streaming (default).
  - `false`: Disable streaming.
- `"session_id"`: (*Body Parameter*)

The ID of the session. If it is not provided, a new session will be generated.

- `"user_id"`: (*Body parameter*), `string`
  The optional user-defined ID. Valid *only* when no `session_id` is provided.
- Other parameters: (*Body Parameter*)
  Parameters specified in the **Begin** component.

Response

success without `session_id` provided and with no parameters specified in the **Begin** component:

```
data:{
    "code": 0,
    "message": "",
    "data": {
        "answer": "Hi! I'm your smart assistant. What can I do for you?",
        "reference": {},
        "id": "31e6091d-88d4-441b-ac65-eae1c055be7b",
        "session_id": "2987ad3eb85f11efb2a70242ac120005"
    }
}
data:{
    "code": 0,
    "message": "",
    "data": true
}
```

Success without `session_id` provided and with parameters specified in the **Begin** component:

```
data:{
    "code": 0,
    "message": "",
    "data": {
        "session_id": "eacb36a0bdff11ef97120242ac120006",
        "answer": "",
        "reference": [],
        "param": [
            {
                "key": "lang",
                "name": "Target Language",
                "optional": false,
                "type": "line",
                "value": "English"
```

```
            },
            {
                "key": "file",
                "name": "Files",
                "optional": false,
                "type": "file",
                "value": "How is the weather tomorrow?"
            },
            {
                "key": "hhyt",
                "name": "hhty",
                "optional": true,
                "type": "line"
            }
        ]
    }
}
data:
```

Success with parameters specified in the **Begin** component:

```
data:{
    "code": 0,
    "message": "",
    "data": {
        "answer": "How",
        "reference": {},
        "id": "0379ac4c-b26b-4a44-8b77-99cebf313fdf",
        "session_id": "4399c7d0b86311efac5b0242ac120005"
    }
}
data:{
    "code": 0,
    "message": "",
    "data": {
        "answer": "How is",
        "reference": {},
        "id": "0379ac4c-b26b-4a44-8b77-99cebf313fdf",
        "session_id": "4399c7d0b86311efac5b0242ac120005"
    }
}
data:{
    "code": 0,
    "message": "",
    "data": {
        "answer": "How is the",
        "reference": {},
        "id": "0379ac4c-b26b-4a44-8b77-99cebf313fdf",
        "session_id": "4399c7d0b86311efac5b0242ac120005"
```

```
    }
}
data:{
    "code": 0,
    "message": "",
    "data": {
        "answer": "How is the weather",
        "reference": {},
        "id": "0379ac4c-b26b-4a44-8b77-99cebf313fdf",
        "session_id": "4399c7d0b86311efac5b0242ac120005"
    }
}
data:{
    "code": 0,
    "message": "",
    "data": {
        "answer": "How is the weather tomorrow",
        "reference": {},
        "id": "0379ac4c-b26b-4a44-8b77-99cebf313fdf",
        "session_id": "4399c7d0b86311efac5b0242ac120005"
    }
}
data:{
    "code": 0,
    "message": "",
    "data": {
        "answer": "How is the weather tomorrow?",
        "reference": {},
        "id": "0379ac4c-b26b-4a44-8b77-99cebf313fdf",
        "session_id": "4399c7d0b86311efac5b0242ac120005"
    }
}
data:{
    "code": 0,
    "message": "",
    "data": {
        "answer": "How is the weather tomorrow?",
        "reference": {},
        "id": "0379ac4c-b26b-4a44-8b77-99cebf313fdf",
        "session_id": "4399c7d0b86311efac5b0242ac120005"
    }
}
data:{
    "code": 0,
    "message": "",
    "data": true
}
```

Failure:

```
{
    "code": 102,
    "message": "`question` is required."
}
```

`&orderby={orderby}&desc={desc}&id={session_id}&user_id={user_id}&dsl={dsl}`

Lists sessions associated with a specified agent.

Request

- Method: GET
- URL: `/api/v1/agents/{agent_id}/sessions?page={page}&page_size={page_size}&orderby={orderby}&desc={desc}&id={session_id}`
- Headers:
  - `'Authorization: Bearer <YOUR_API_KEY>'`

Request example

```
curl --request GET \
     --url http://{address}/api/v1/agents/{agent_id}/sessions?page={page}
&page_size={page_size}&orderby={orderby}&desc={desc}&id={session_id}
&user_id={user_id} \
     --header 'Authorization: Bearer <YOUR_API_KEY>'
```

Request Parameters

- `agent_id`: (*Path parameter*)
  The ID of the associated agent.
- `page`: (*Filter parameter*), `integer`
  Specifies the page on which the sessions will be displayed. Defaults to `1`.
- `page_size`: (*Filter parameter*), `integer`
  The number of sessions on each page. Defaults to `30`.
- `orderby`: (*Filter parameter*), `string`
  The field by which sessions should be sorted. Available options:
  - `create_time` (default)
  - `update_time`

- `desc`: (*Filter parameter*), `boolean`
  Indicates whether the retrieved sessions should be sorted in descending order.
  Defaults to `true`.
- `id`: (*Filter parameter*), `string`
  The ID of the agent session to retrieve.
- `user_id`: (*Filter parameter*), `string`
  The optional user-defined ID passed in when creating session.
- `dsl`: (*Filter parameter*), `boolean`
  Indicates whether to include the dsl field of the sessions in the response. Defaults to `true`.

Response

Success:


```
{
    "code": 0,
    "data": [{
        "agent_id": "e9e2b9c2b2f911ef801d0242ac120006",
        "dsl": {
            "answer": [],
            "components": {
                "Answer:OrangeTermsBurn": {
                    "downstream": [],
                    "obj": {
                        "component_name": "Answer",
                        "params": {}
                    },
                    "upstream": []
                },
                "Generate:SocialYearsRemain": {
                    "downstream": [],
                    "obj": {
                        "component_name": "Generate",
                        "params": {
                            "cite": true,
                            "frequency_penalty": 0.7,
                            "llm_id": "gpt-4o___OpenAI-API@OpenAI-API-Compatible",
                            "max_tokens": 256,
                            "message_history_window_size": 12,
                            "parameters": [],
                            "presence_penalty": 0.4,
                            "prompt": "Please summarize the following paragraph. Pay
attention to the numbers and do not make things up. The paragraph is as follows:
```

```
\n{input}\nThis is what you need to summarize.",
                            "temperature": 0.1,
                            "top_p": 0.3
                        }
                    },
                    "upstream": []
                },
                "begin": {
                    "downstream": [],
                    "obj": {
                        "component_name": "Begin",
                        "params": {}
                    },
                    "upstream": []
                }
            },
            "graph": {
                "edges": [],
                "nodes": [
                    {
                        "data": {
                            "label": "Begin",
                            "name": "begin"
                        },
                        "height": 44,
                        "id": "begin",
                        "position": {
                            "x": 50,
                            "y": 200
                        },
                        "sourcePosition": "left",
                        "targetPosition": "right",
                        "type": "beginNode",
                        "width": 200
                    },
                    {
                        "data": {
                            "form": {
                                "cite": true,
                                "frequencyPenaltyEnabled": true,
                                "frequency_penalty": 0.7,
                                "llm_id": "gpt-4o___OpenAI-API@OpenAI-API-
Compatible",
                                "maxTokensEnabled": true,
                                "max_tokens": 256,
                                "message_history_window_size": 12,
                                "parameters": [],
                                "presencePenaltyEnabled": true,
                                "presence_penalty": 0.4,
                                "prompt": "Please summarize the following paragraph.
Pay attention to the numbers and do not make things up. The paragraph is as follows:
```

```
\n{input}\nThis is what you need to summarize.",
                                "temperature": 0.1,
                                "temperatureEnabled": true,
                                "topPEnabled": true,
                                "top_p": 0.3
                            },
                            "label": "Generate",
                            "name": "Generate Answer_0"
                        },
                        "dragging": false,
                        "height": 105,
                        "id": "Generate:SocialYearsRemain",
                        "position": {
                            "x": 561.3457829707513,
                            "y": 178.7211182312641
                        },
                        "positionAbsolute": {
                            "x": 561.3457829707513,
                            "y": 178.7211182312641
                        },
                        "selected": true,
                        "sourcePosition": "right",
                        "targetPosition": "left",
                        "type": "generateNode",
                        "width": 200
                    },
                    {
                        "data": {
                            "form": {},
                            "label": "Answer",
                            "name": "Dialogue_0"
                        },
                        "height": 44,
                        "id": "Answer:OrangeTermsBurn",
                        "position": {
                            "x": 317.2368194777658,
                            "y": 218.30635555445093
                        },
                        "sourcePosition": "right",
                        "targetPosition": "left",
                        "type": "logicNode",
                        "width": 200
                    }
                ]
            },
            "history": [],
            "messages": [],
            "path": [],
            "reference": []
        },
        "id": "792dde22b2fa11ef97550242ac120006",
```

```
        "message": [
            {
                "content": "Hi! I'm your smart assistant. What can I do for you?",
                "role": "assistant"
            }
        ],
        "source": "agent",
        "user_id": ""
    }]
}
```

Failure:

```
{
    "code": 102,
    "message": "You don't own the agent ccd2f856b12311ef94ca0242ac1200052."
}
```

Delete agent's sessions

DELETE `/api/v1/agents/{agent_id}/sessions`

Deletes sessions of a agent by ID.

Request

- Method: DELETE
- URL: `/api/v1/agents/{agent_id}/sessions`
- Headers:
  - `'content-Type: application/json'`
  - `'Authorization: Bearer <YOUR_API_KEY>'`
- Body:
  - `"ids"`: `list[string]`

Request example

```
curl --request DELETE \
     --url http://{address}/api/v1/agents/{agent_id}/sessions \
     --header 'Content-Type: application/json' \
```

```
--header 'Authorization: Bearer <YOUR_API_KEY>' \
--data '
{
     "ids": ["test_1", "test_2"]
}'
```

Request Parameters

- `agent_id`: (*Path parameter*)
  The ID of the associated agent.
- `"ids"`: (*Body Parameter*), `list[string]`
  The IDs of the sessions to delete. If it is not specified, all sessions associated with the specified agent will be deleted.

Response

Success:

```
{
    "code": 0
}
```

Failure:

```
{
    "code": 102,
    "message": "The agent doesn't own the session cbd31e52f73911ef93b232903b842af6"
}
```

AGENT MANAGEMENT

List agents

GET `/api/v1/agents?page={page}&page_size={page_size}&orderby={orderby}&desc={desc}&name={agent_name}&id={agent_id}`

Lists agents.

Request

- Method: GET
- URL: `/api/v1/agents?page={page}&page_size={page_size}&orderby={orderby}&desc={desc}&name={agent_name}&id={agent_id}`
- Headers:
  - `'Authorization: Bearer <YOUR_API_KEY>'`

Request example

```
curl --request GET \
     --url http://{address}/api/v1/agents?page={page}&page_size={page_size}
&orderby={orderby}&desc={desc}&name={agent_name}&id={agent_id} \
     --header 'Authorization: Bearer <YOUR_API_KEY>'
```

Request parameters

- `page`: (*Filter parameter*), `integer`
  Specifies the page on which the agents will be displayed. Defaults to `1`.
- `page_size`: (*Filter parameter*), `integer`
  The number of agents on each page. Defaults to `30`.
- `orderby`: (*Filter parameter*), `string`
  The attribute by which the results are sorted. Available options:
  - `create_time` (default)
  - `update_time`
- `desc`: (*Filter parameter*), `boolean`
  Indicates whether the retrieved agents should be sorted in descending order. Defaults to `true`.
- `id`: (*Filter parameter*), `string`
  The ID of the agent to retrieve.
- `name`: (*Filter parameter*), `string`
  The name of the agent to retrieve.

Response

Success:


```
{
    "code": 0,
    "data": [
        {
            "avatar": null,
            "canvas_type": null,
            "create_date": "Thu, 05 Dec 2024 19:10:36 GMT",
            "create_time": 1733397036424,
            "description": null,
            "dsl": {
                "answer": [],
                "components": {
                    "begin": {
                        "downstream": [],
                        "obj": {
                            "component_name": "Begin",
                            "params": {}
                        },
                        "upstream": []
                    }
                },
                "graph": {
                    "edges": [],
                    "nodes": [
                        {
                            "data": {
                                "label": "Begin",
                                "name": "begin"
                            },
                            "height": 44,
                            "id": "begin",
                            "position": {
                                "x": 50,
                                "y": 200
                            },
                            "sourcePosition": "left",
                            "targetPosition": "right",
                            "type": "beginNode",
                            "width": 200
                        }
                    ]
                },
                "history": [],
                "messages": [],
                "path": [],
                "reference": []
            },
            "id": "8d9ca0e2b2f911ef9ca20242ac120006",
```

```
            "title": "123465",
            "update_date": "Thu, 05 Dec 2024 19:10:56 GMT",
            "update_time": 1733397056801,
            "user_id": "69736c5e723611efb51b0242ac120007"
        }
    ]
}
```


Failure:


```
{
    "code": 102,
    "message": "The agent doesn't exist."
}
```


Edit this page