

UI | Qdrant

localhost:6333/dashboard#/collections/vector_crud

vector_crud

Welcome Console Collections Tutorial Datasets Access Tokens

Points Info Cluster Search Quality Snapshots Visualize Graph

Point 1045496c-c2b2-446f-ba1d-e0087714a3a8

Payload

```
{ "text": "Best practices for securing APIs in production" }
```

Vectors:

Default vector Length: 768

Copy Open Graph Find Similar

Point 23f63d11-1100-44e2-adf7-3064dbf3793f

Payload

```
{ "text": "Exploring biking trails in the Alps." }
```

Odrant v1.6.3

The screenshot shows the Qdrant UI interface. On the left is a sidebar with navigation links: Welcome, Console, Collections (which is selected), Tutorial, Datasets, and Access Tokens. Below the sidebar are tabs: Points, Info, Cluster, Search Quality, Snapshots, Visualize, and Graph. The main area is titled 'vector_crud'. It displays two vector documents. The first document has the ID '1045496c-c2b2-446f-ba1d-e0087714a3a8' and its payload is: { "text": "Best practices for securing APIs in production" }. The second document has the ID '23f63d11-1100-44e2-adf7-3064dbf3793f' and its payload is: { "text": "Exploring biking trails in the Alps." }. Both documents have a 'Payload' section with copy and edit icons, and a 'Vectors' section showing a default vector of length 768 with copy, open graph, and find similar buttons.

UI | Qdrant VectorDB using Ollama and Qdrant

127.0.0.1:8000/docs#/default/store_docs_vector_doc_post

VectorDB using Ollama and Qdrant 0.1.0 OAS 3.1

/openapi.json

default

GET / Welcome

POST /vector-doc Store Docs

Parameters

No parameters

Request body required

application/json

Edit Value | Schema

```
{  
  "docs": "RAG fetches relevant documents from an external knowledge base such as APIs, databases etc"  
}
```

UI | Qdrant

VectorDB using Ollama and Qdrant

127.0.0.1:8000/docs#/default/store_docs_vector_doc_post

Responses

Curl

```
curl -X 'POST' \
  'http://127.0.0.1:8000/vector-doc' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "docs": "RAG fetches relevant documents from an external knowledge base such as apis, databases etc"
}'
```

Request URL

```
http://127.0.0.1:8000/vector-doc
```

Server response

Code Details

200 Response body

```
{ "status": "Document stored successfully" }
```

Download

Response headers

```
content-length: 41
content-type: application/json
date: Wed, 07 Jan 2026 14:51:42 GMT
server: uvicorn
```

Responses

Code	Description	Links
200	Successful Response	No links

Media type

application/json

UI | Qdrant X VectorDB using Ollama and Qdrant +

127.0.0.1:8000/docs#/default/search_query_vector_query_post

Curl

```
curl -X 'POST' \
  'http://127.0.0.1:8000/vector-query?limit=3' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "query": "Explain RAG"
}'
```

Request URL

```
http://127.0.0.1:8000/vector-query?limit=3
```

Server response

Code Details

200 Response body

```
{
  "query": "Explain RAG",
  "results": [
    {
      "id": "949356e4-e590-4871-84d4-fbb51c6c7a47",
      "score": 0.77124895,
      "payload": {
        "text": "RAG is Retrieval Augmented Generation"
      }
    },
    {
      "id": "e1df0865-f980-4351-99a6-ddaff8925c70",
      "score": 0.75198858,
      "payload": {
        "text": "How to connect Qdrant Cloud from Google Colab."
      }
    },
    {
      "id": "b5affecbb-6afe-4467-8dd3-21b76c8113c4",
      "score": 0.67480876,
      "payload": {
        "text": "A guide to building RAG systems with Qdrant."
      }
    }
  ]
}
```

Download