Qdrant

Qdrant MCP

Semantic Code Lookup for AI Assistants

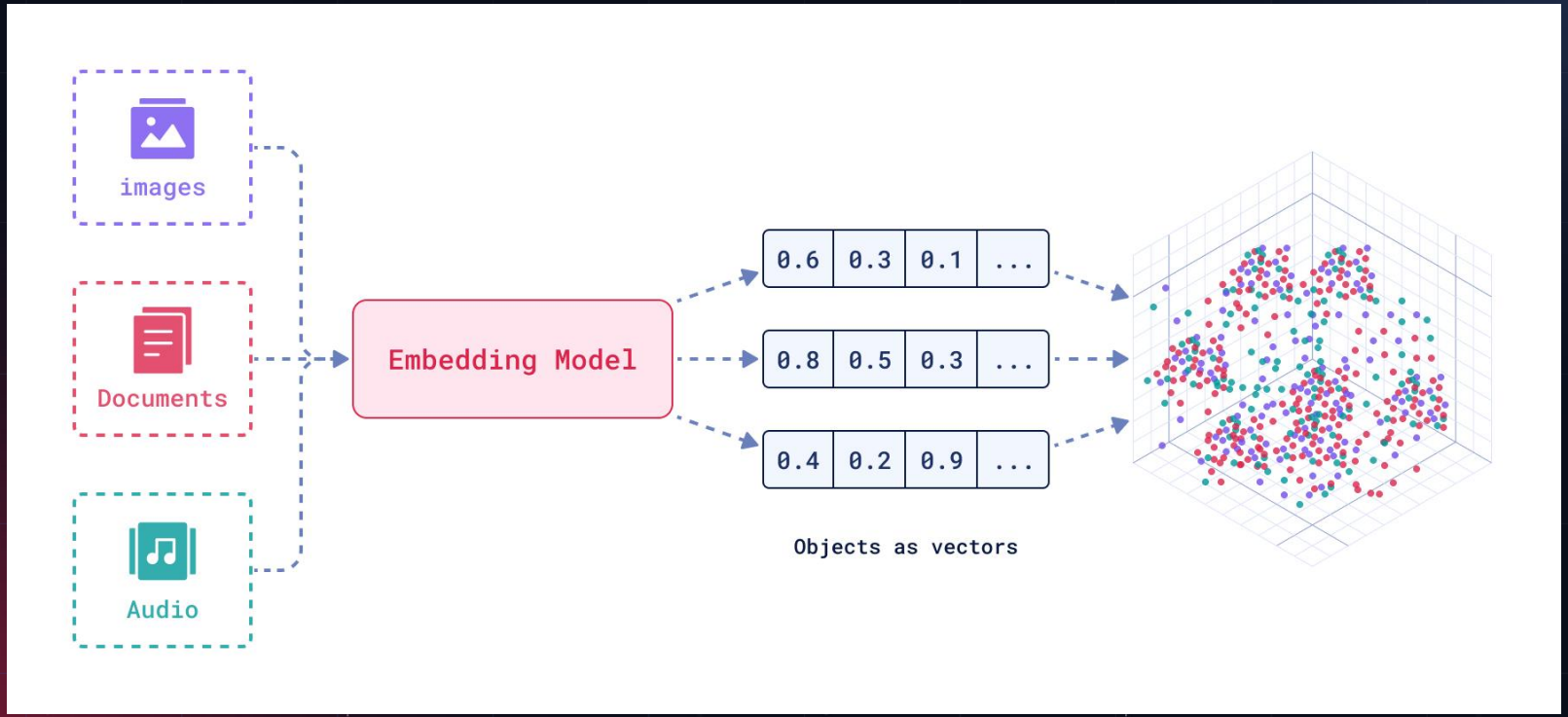
Evgeniya Sukhodolskaya,
Developer Advocate,
Qdrant

Qdrant & Semantic Lookup

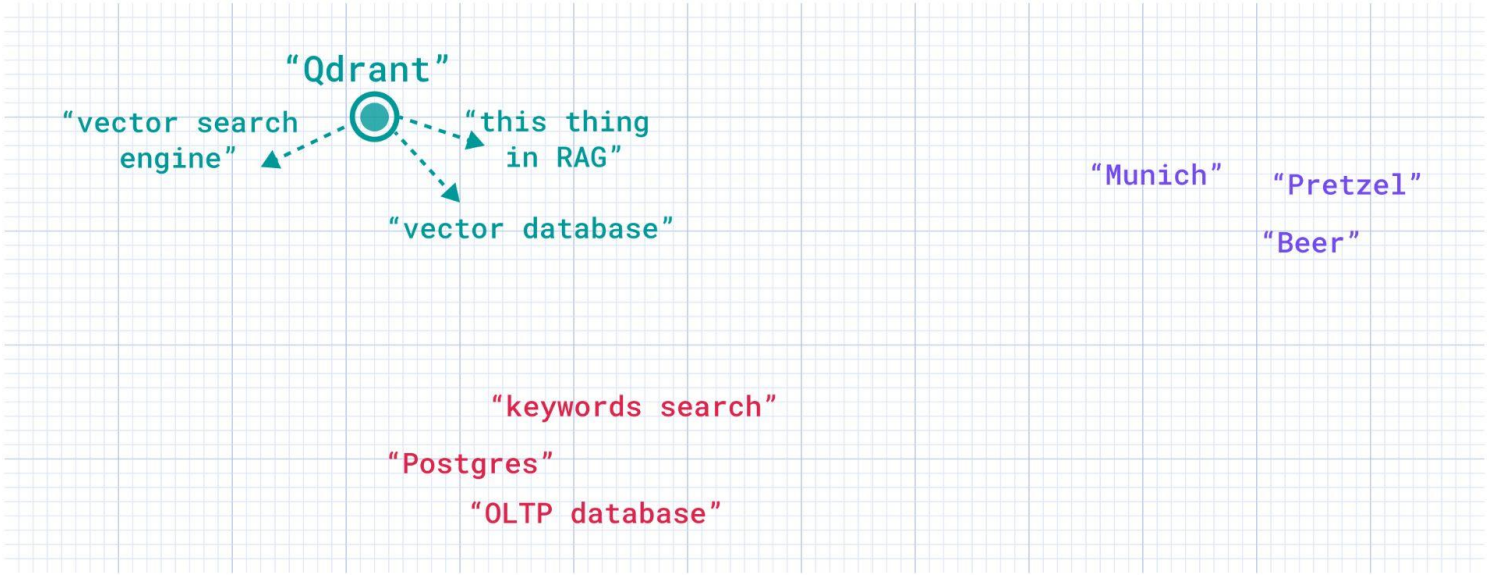
Qdrant is an open-source Vector Search Engine

or you might have also heard the term “*vector database*”, “*semantic similarity search engine*”, “*this thing in (Agentic) RAG*”

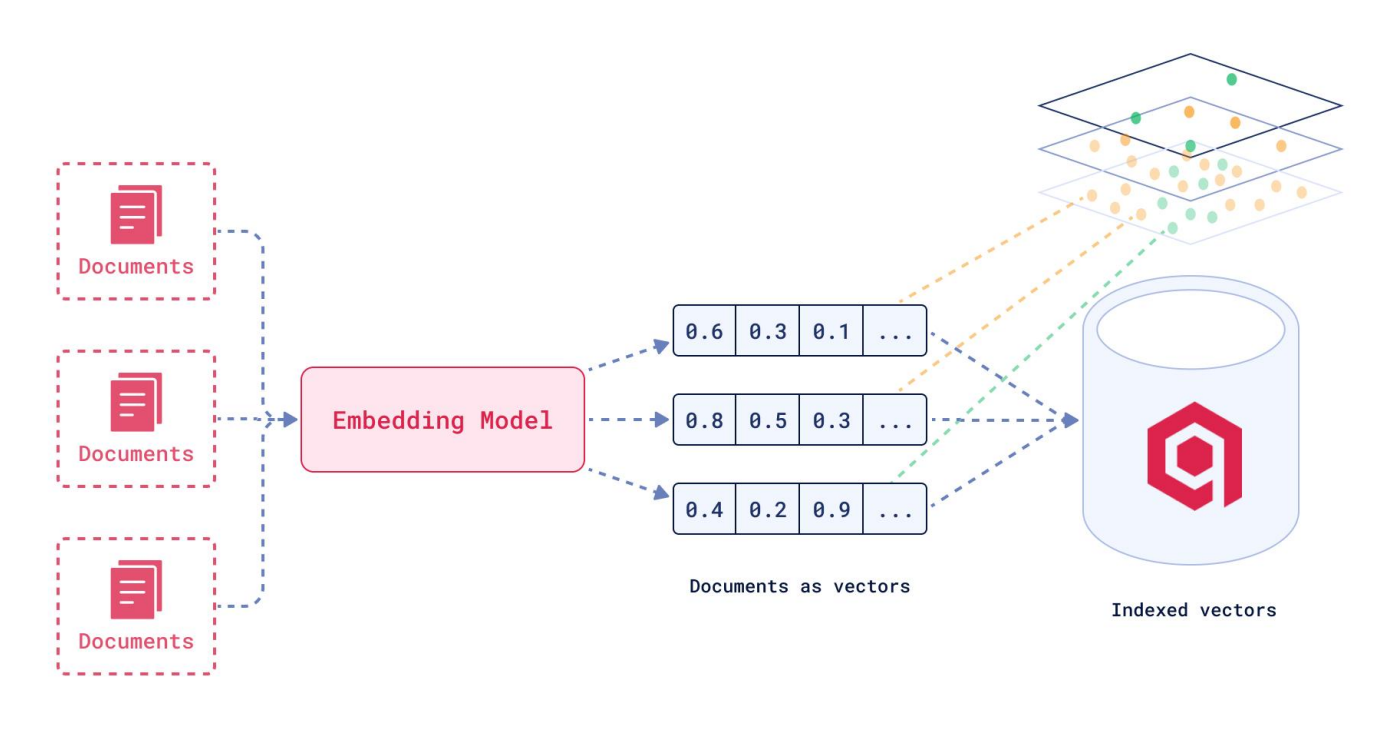
Qdrant is a Vector Search Engine



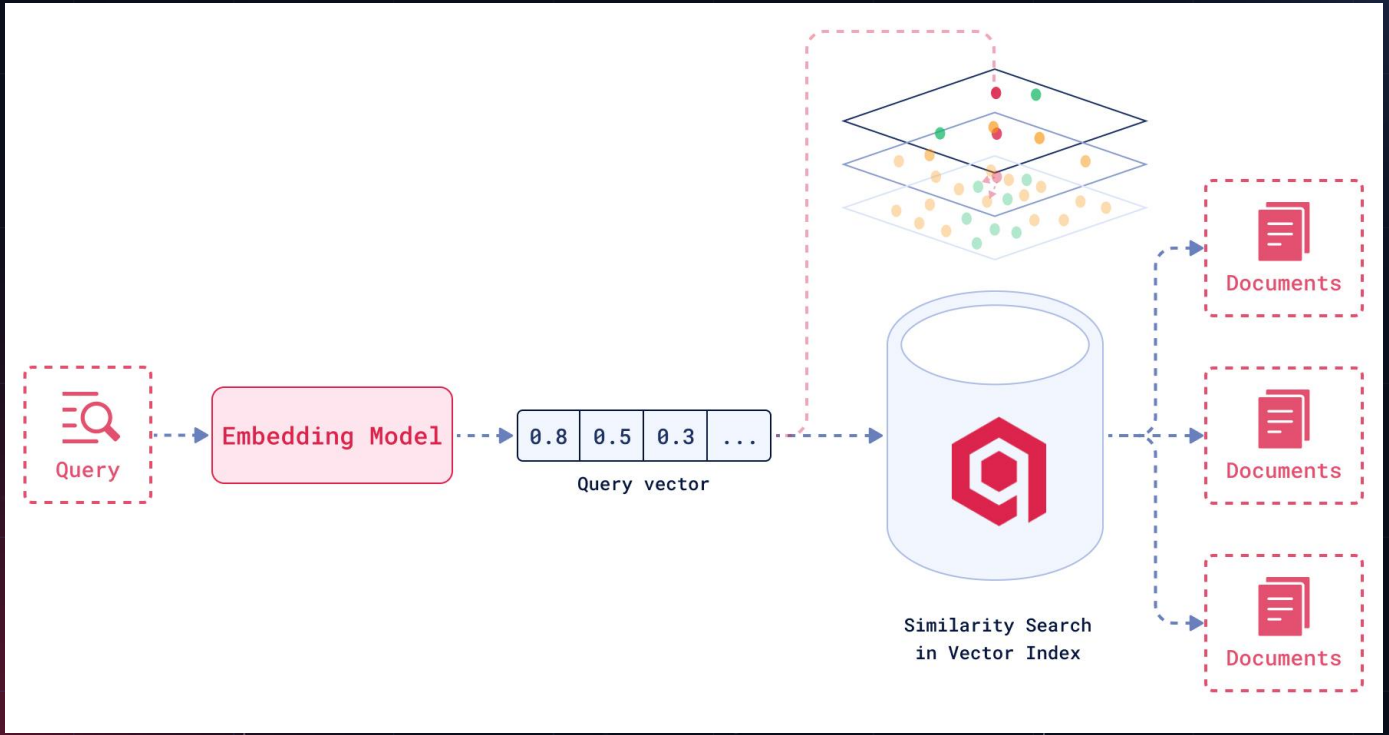
Qdrant is a Vector Search Engine



Qdrant is a Vector Search Engine: Store



Qdrant is a Vector Search Engine: Find



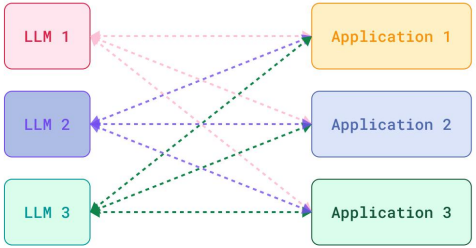
Sources for Ones Interested

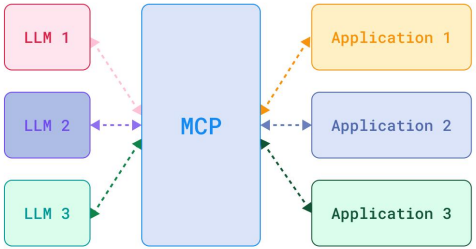
Article *“An Introduction to Vector Databases”* –
an overview of all relevant concepts;

Article *“Built for Vector Search”* –
why vector search needs a dedicated solution.

Qdrant MCP Server

Model Context Protocol (MCP)





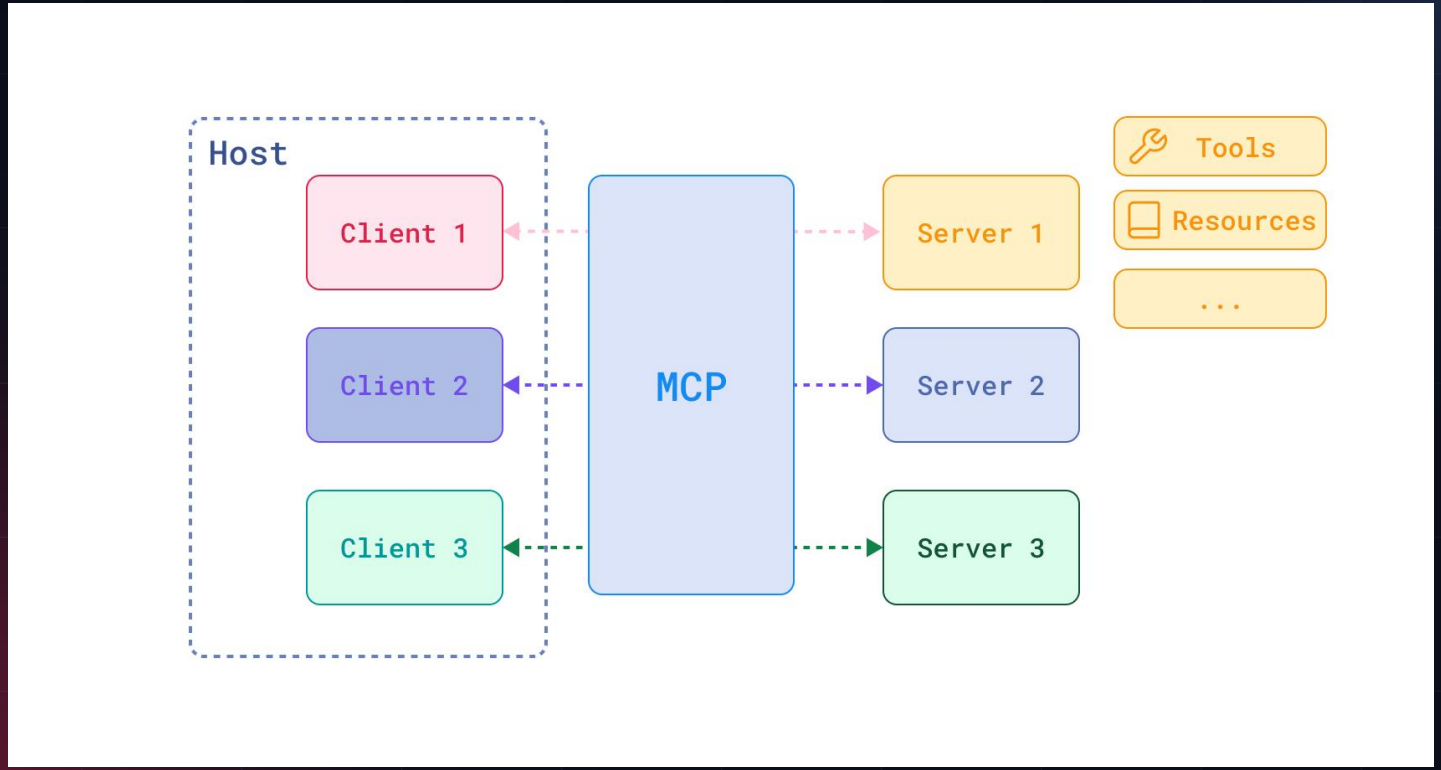
"MCP is an open protocol that standardizes how applications provide context to LLMs.

Think of MCP like a USB-C port for AI applications."

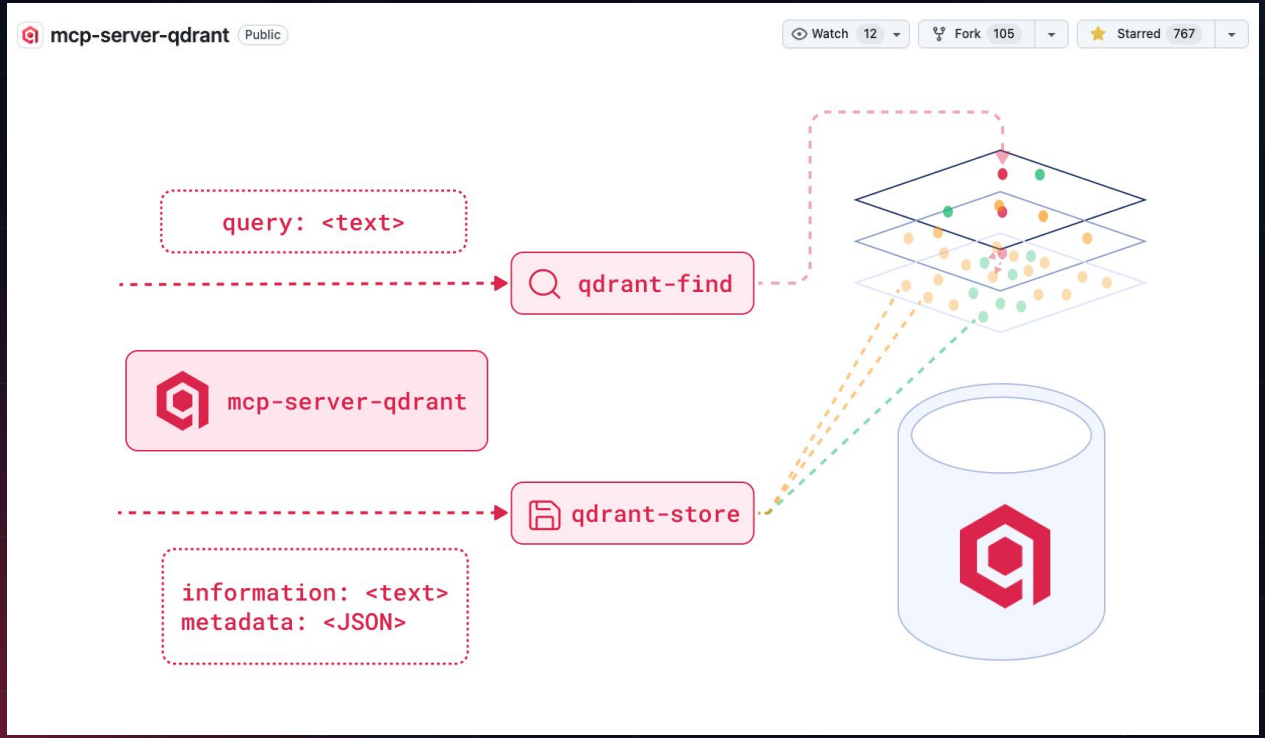
modelcontextprotocol.io

HuggingFace MCP course

Model Context Protocol (MCP)



Qdrant MCP Server



mcp-server-qdrant

Possible Applications

- Inline Retrieval-Augmented Generation (RAG)
  <originally used as a note-taker>;
- Automating your codebase documentation;
- Personalizing your code assistant with
  best practices, templates,
  and specifics of your codebase;
- (today's talk) Providing API reference
  and code documentation for agents.

How it all Started

👤 **User:**

- *Implement hybrid search in Qdrant*

🤖 **Coding Assistant:**

- Uses deprecated method “search” instead of “query_points”;
- Doesn’t know Qdrant, since 1.10.0, supports combining searches with “prefetch” & fusion;
- Doesn’t know Qdrant can calculate Inverse Document Frequency (IDF) on the server side;
- Doesn’t know about local & cloud inference...

=> People come & say: “*Qdrant doesn’t have hybrid search functionality*” 🤦

How it all Started

So, AI coding assistant fails to implement hybrid search with Qdrant (and it's even one of the widely known applications).

Why?
LLMs are retrained, but it's mostly **impossible to keep them** constantly **updated** & **in sync** with specific library versions.

And Qdrant has:

- **Several clients** (Java, Python, GO, C#, Rust, Typescript);
- **Several versions** (1.15 comes out this week, btw), each with new features.

LLM-friendly API reference?
Yes, we have **/llm-full.txt** at our API Reference page, but it's big (**~53K tokens**), so models will still have troubles to generate desired code when fed such a large context.

Idea: mcp-server-qdrant as API Reference for Agents

What do We Want

We need, for all package functions, to **store**:

- Well-curated snippet;
- Accurate description;
- Programming language tag;
- Version tag;
- Package name tag (*who said we'll stop at Qdrant if it works?*).

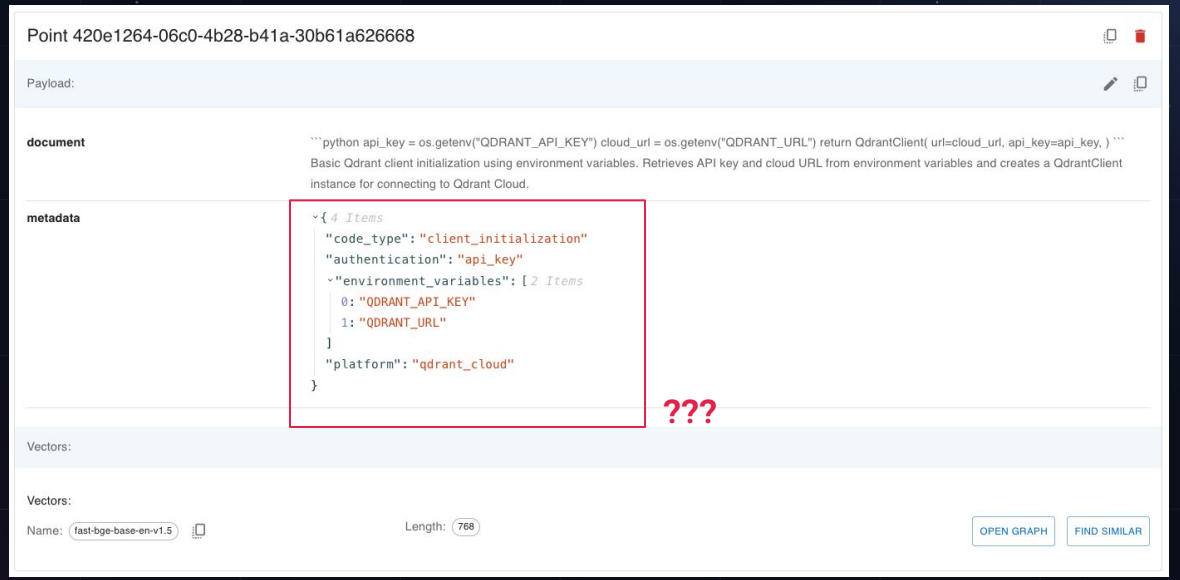
We need, for every user request, to **find:**

- **Semantic match** by description of a code snippet,
  filtered by fitting version, package name, and programming language

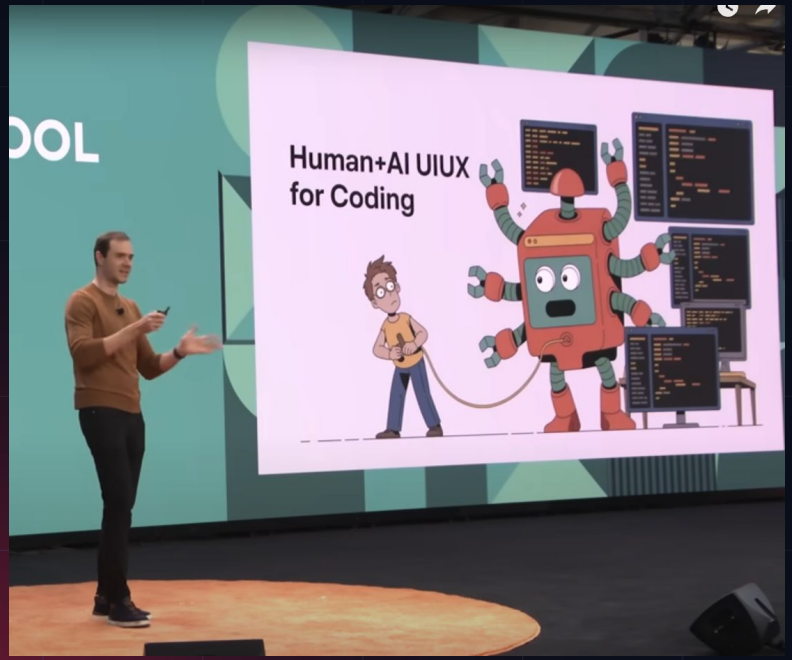
At the Same Time AI Assistant

qdrant-store:
sometimes stores weird metadata

qdrant-find:
troubles with consistent & correctly-formed filters



Keep Agents on a Leash



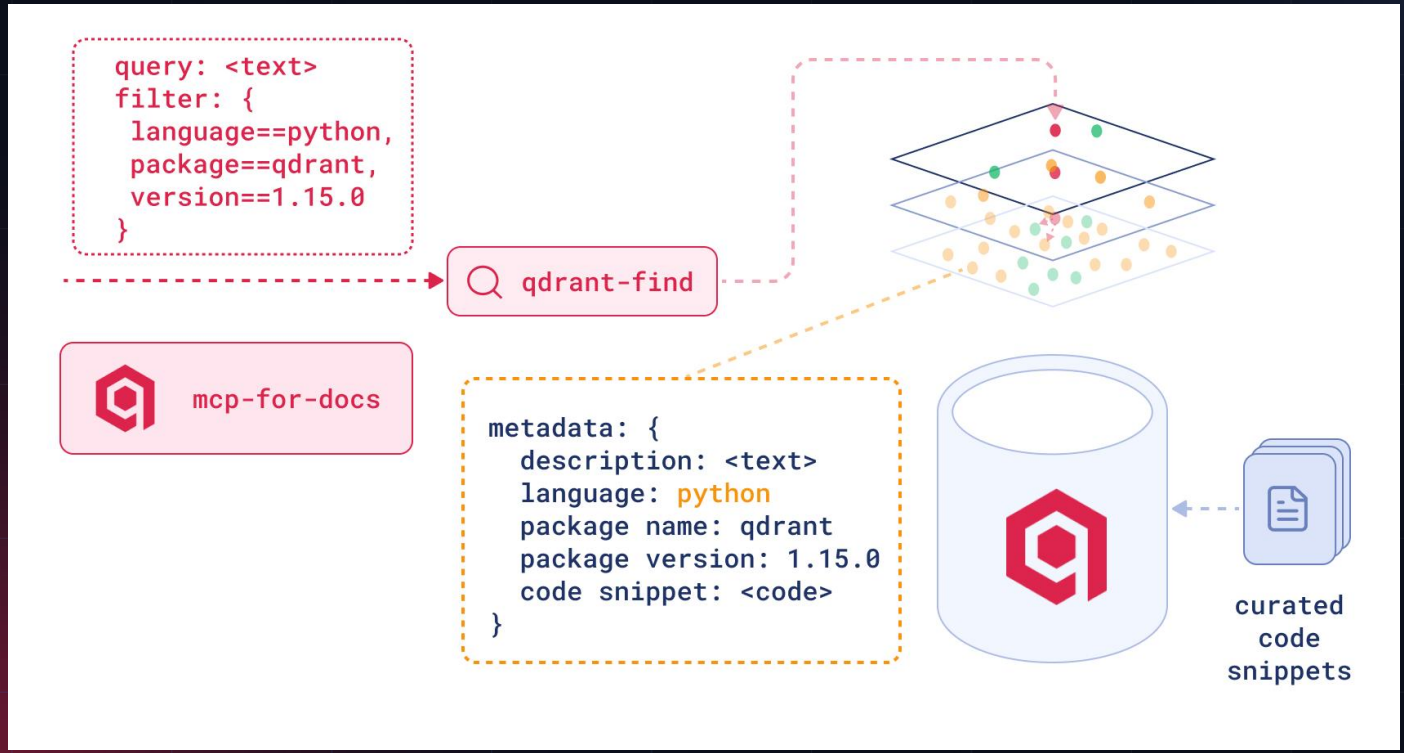
We need to help our AI coding assistants by avoiding situations with **too many degrees of freedom**.

So, we updated our **mcp-server-qdrant** to use it as a **customizable base class**:

- **Customizable filters** restrictions & LLM-friendly format of metadata filtering
- Customizable switching `qdrant-store` tool on/off (`**read_only` mode**)

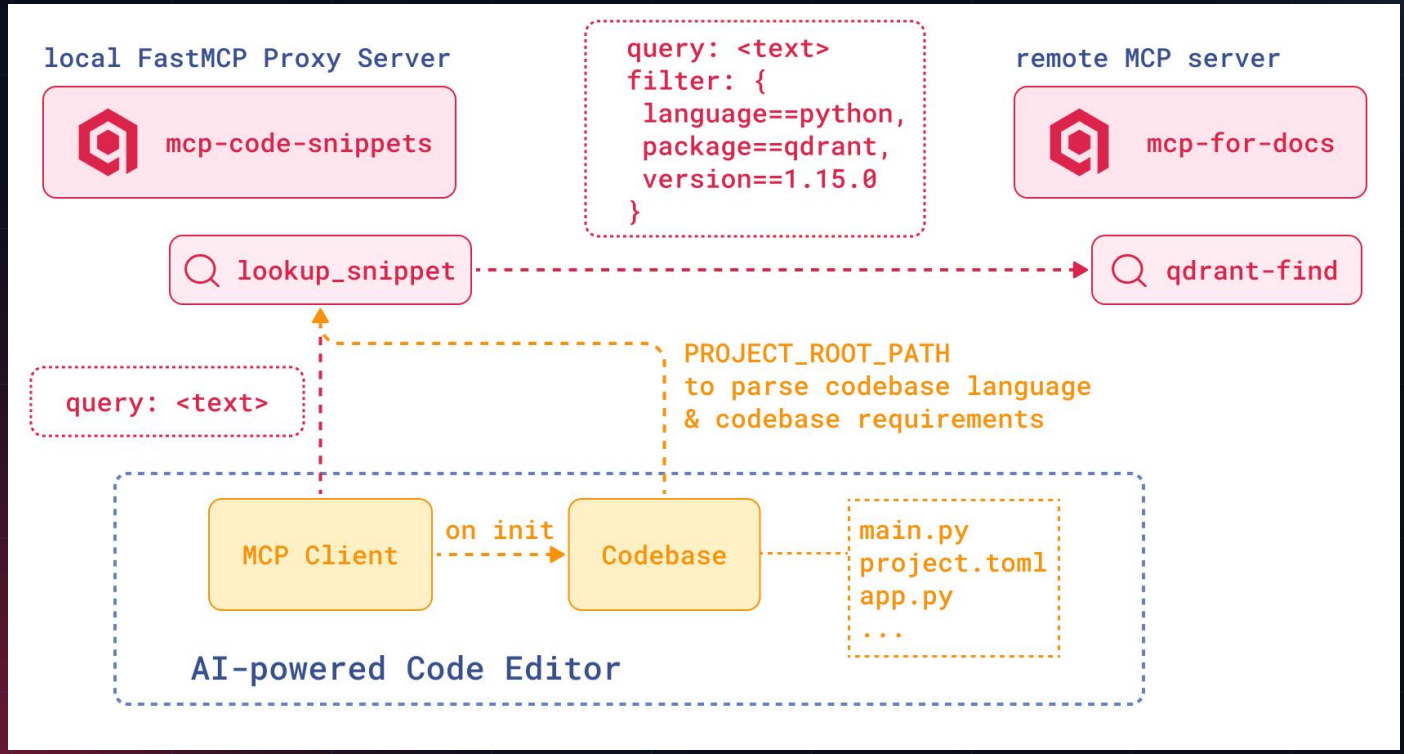
And built on top our
own custom MCP server, suitable solely as an API reference for AI coding assistants.

#1 Remote MCP server: mcp-for-docs



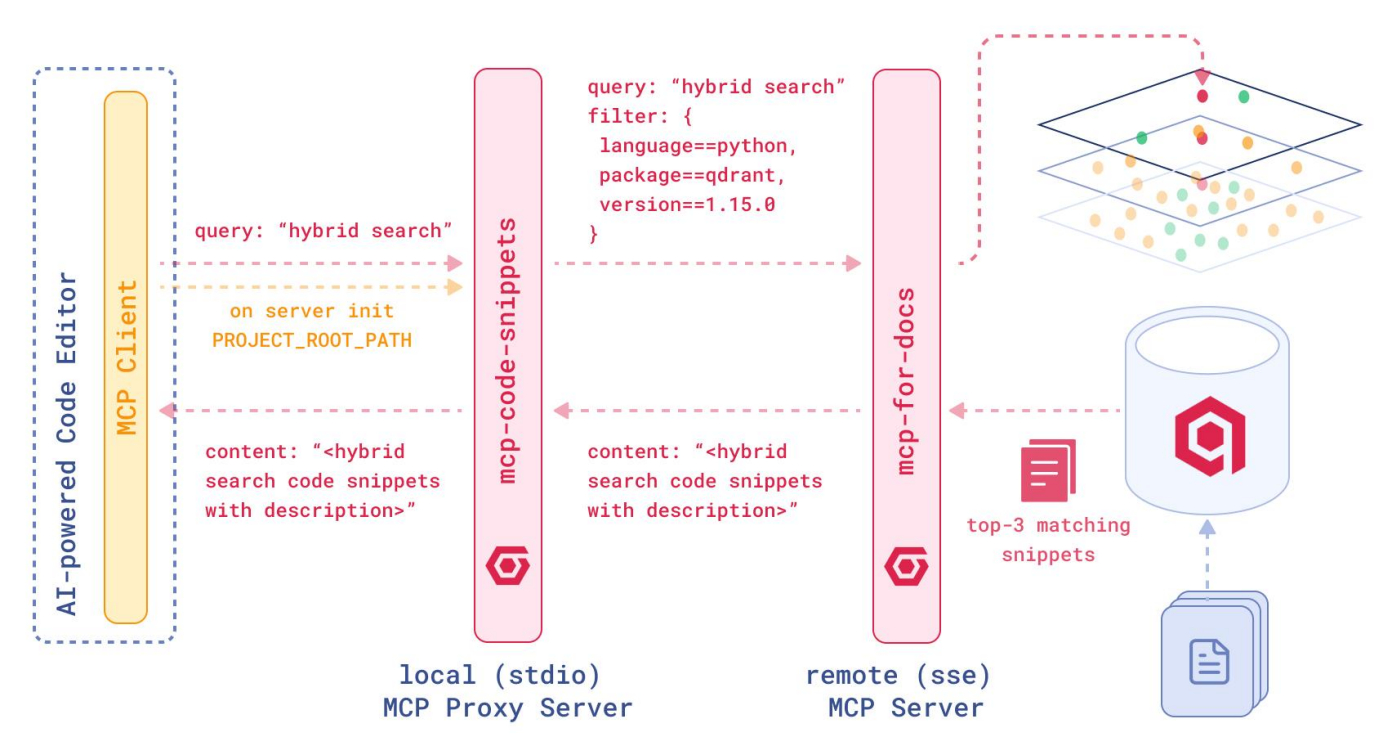
mcp-for-docs

#2 Local MCP Proxy: mcp-for-docs



mcp-code-snippets

How Does it Work



How to Install

{} mcp.json U ●

.vscode > {} mcp.json > ...

```
{
    "servers": {
        ✓Running | Stop | Restart | Debug | 1 tools | More...
        "mcpCodeSnippets": {
            "command": "uvx",
            "args": ["mcp-code-snippets"],
            "env": {
                "MCP_PROXY_CONFIG": "/Users/evgeniya_sukhodolskaya/Desktop/Qdrant/mcp-test/.vscode/mcp_proxy.json",
                "PROJECT_ROOT_PATH": "/Users/evgeniya_sukhodolskaya/Desktop/Qdrant/mcp-test"
            }
        }
    }
}
```

{} mcp.json U ● {} mcp_proxy.json U ●

.vscode > {} mcp_proxy.json > ...

```
{
    "mcpServers": {
        "mcp-server-qdrant": {
            "url": "https://mcp.qdrant.tech/mcp/",
            "transport": "streamable-http"
        }
    }
}
```

What’s Now & What’s Next

What’s Now:

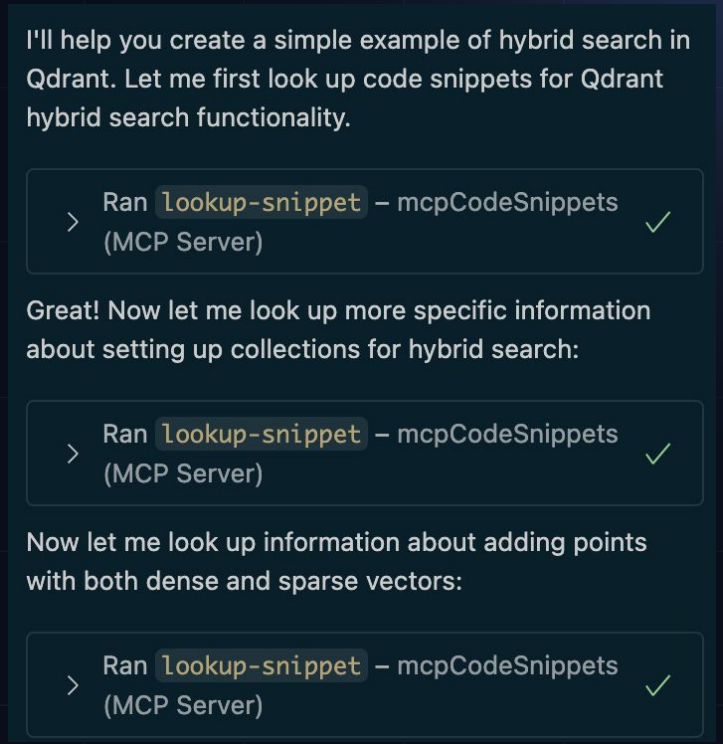
`**mcp-code-snippets**` can be installed and works with Qdrant code snippets of the latest version.
So if you use it, hybrid search case will work–>

Plans:

- Further development, publishing & maintenance;
- Package versioning support;
- Support for other packages;
- Further development of the base `**mcp-server-qdrant**` server.

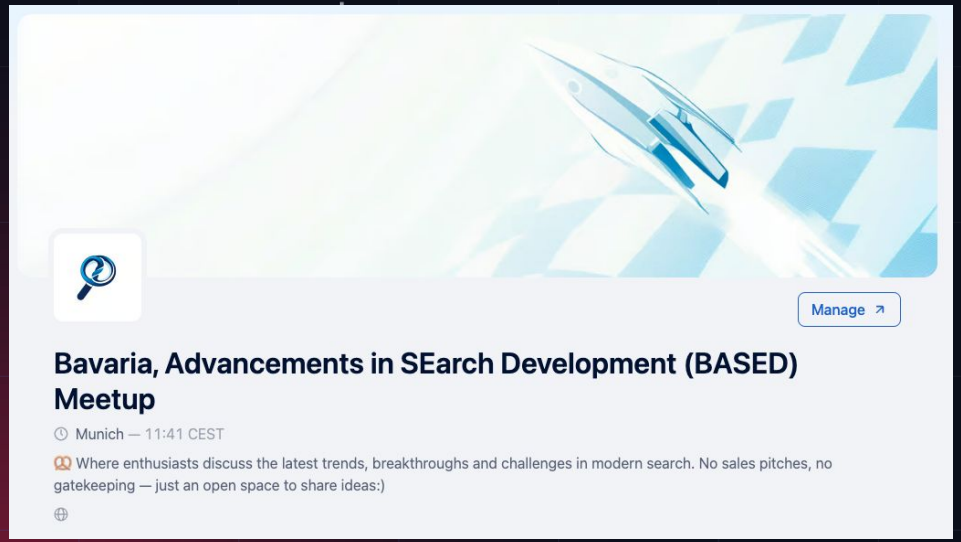
Goal:

Turn `**mcp-code-snippets**` into a **universal open-source API reference for AI coding assistants**.



Thank You!
Let’s Keep the Tech Yapping Going:)

My Munich Search Meetup



lu.ma/based_meetup

My LinkedIn -> Evgeniya Sukhodolskaya