VDBMS

quyanh.nguyen@helsinki.fi

Table of contents

01 Vectors & Vectorisation

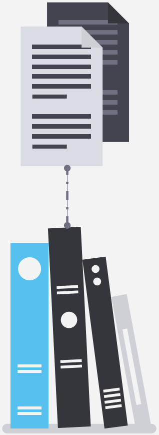
02 VDBMS Introduction

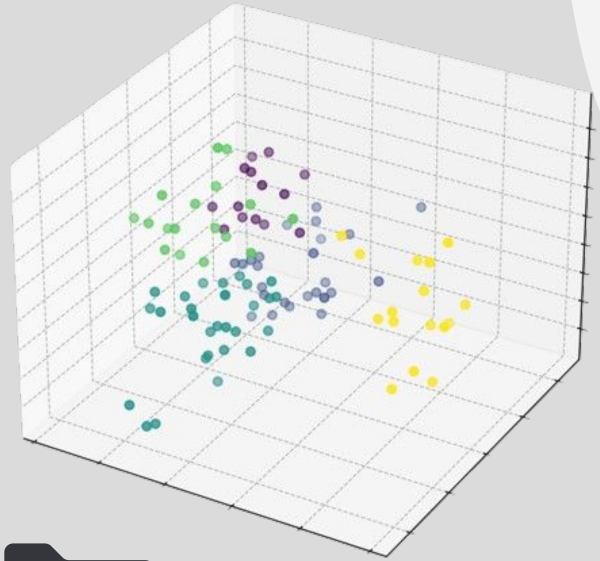
03 Similarity Search

04 Image Search Demo

05 Usage with LLMs

06 Naïve RAG Demo





01. Vectors & Vectorisation

What is a vector?

=> An **array of numbers**:

[0.03712432, 0.030104533, 0.079726376, 0.03848225]

- A vector that represents some data object is called a **vector embedding**.

- **Vectorisation:** The process of converting some data object, e.g. an image, a video, or a JSON document, into a vector embedding.

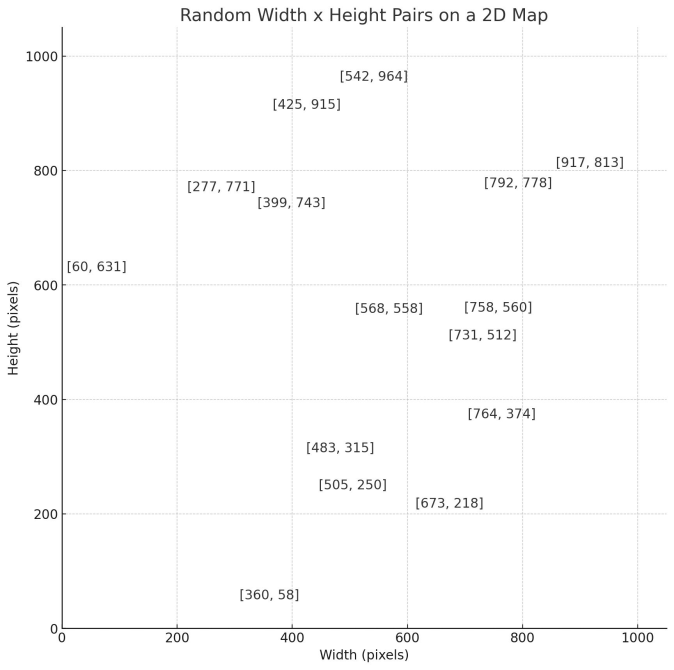
```
[[ 0.0618615, -0.534259 ,  0.545717 ],
 [-0.6649279, -0.590766 , -0.9606892],
 [-0.468922 ,  0.3946947, -0.8946314],
 [-0.2190834,  0.3915435,  0.5040905],
 [ 0.2973618,  0.9794774,  0.7686151],
 [ 0.8759577,  0.9395101,  0.7044962],
 [ 0.0140189, -0.1538083, -0.5995125],
 [ 0.0130325,  0.1164392, -0.2760475],
 [-0.8062036, -0.0230114,  0.6596168]]
```

Vectorisation

- In theory: We might want each value in a vector embedding to represent some meaningful attribute of the original object. For instance:

⇨ [1920, 1080]

If we were to plot these vectors:



$$d(a,b) = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}$$

Vectorisation

- In practice: We use **pre-trained models** specifically designed to create embeddings. Examples of pre-trained models include:

+ *text-embedding-3-small* and *text-embedding-3-large* by OpenAI for text e.g., raw text, code, json, etc.

+ *ResNet50* for images

+ *ImageBind* by Meta AI for images, video, audio, text, depth, thermal and inertial measurement units

These models are either:

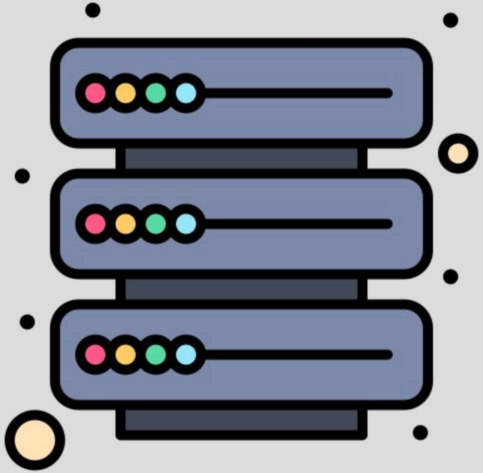
+ **Open-source** and **can be run locally** as part of a Vector DBMS;

+ **Proprietary** and **only accessible via API.**

02. VDBMS

- In a **Relational DBMS,** each **record** is a **row**, multiple rows make up a **table**, and multiple tables make up a **database**.
- In a **Vector DBMS**, each **record** is a **vector embedding**, multiple vector embeddings make up an **embedding space**.

=> A VDBMS is a DBMS for **managing embedding spaces**. It supports **storing** and **retrieving** vector embeddings, and each record can contain, in addition to the vector embedding, some **metadata** and the original data, also called a **payload**.



02. VDBMS

A VDBMS typically consists of:

- A **query processor,** which supports query types such as:

+ **k-nearest neighbor (k-NN)** queries

+ **Approximate nearest neighbor (ANN)** queries

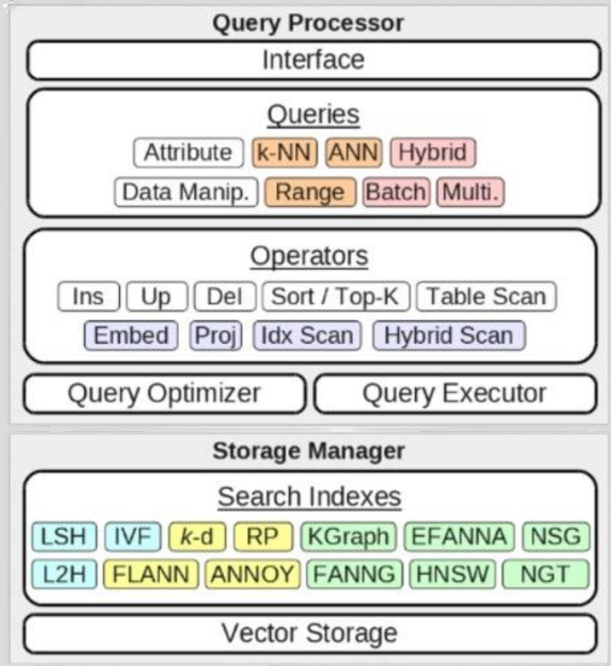
+ **Hybrid** queries

+ **Multi-vector** queries

+ **Aggregate** queries

- A **storage manager**, which **stores** and **indexes vectors** for faster retrieval.

- Optionally, **embedding models**.



Query Processor

Interface

Queries

Attribute k-NN ANN Hybrid

Data Manip. Range Batch Multi.

Operators

Ins Up Del Sort / Top-K Table Scan

Embed Proj Idx Scan Hybrid Scan

Query Optimizer

Query Executor

Storage Manager

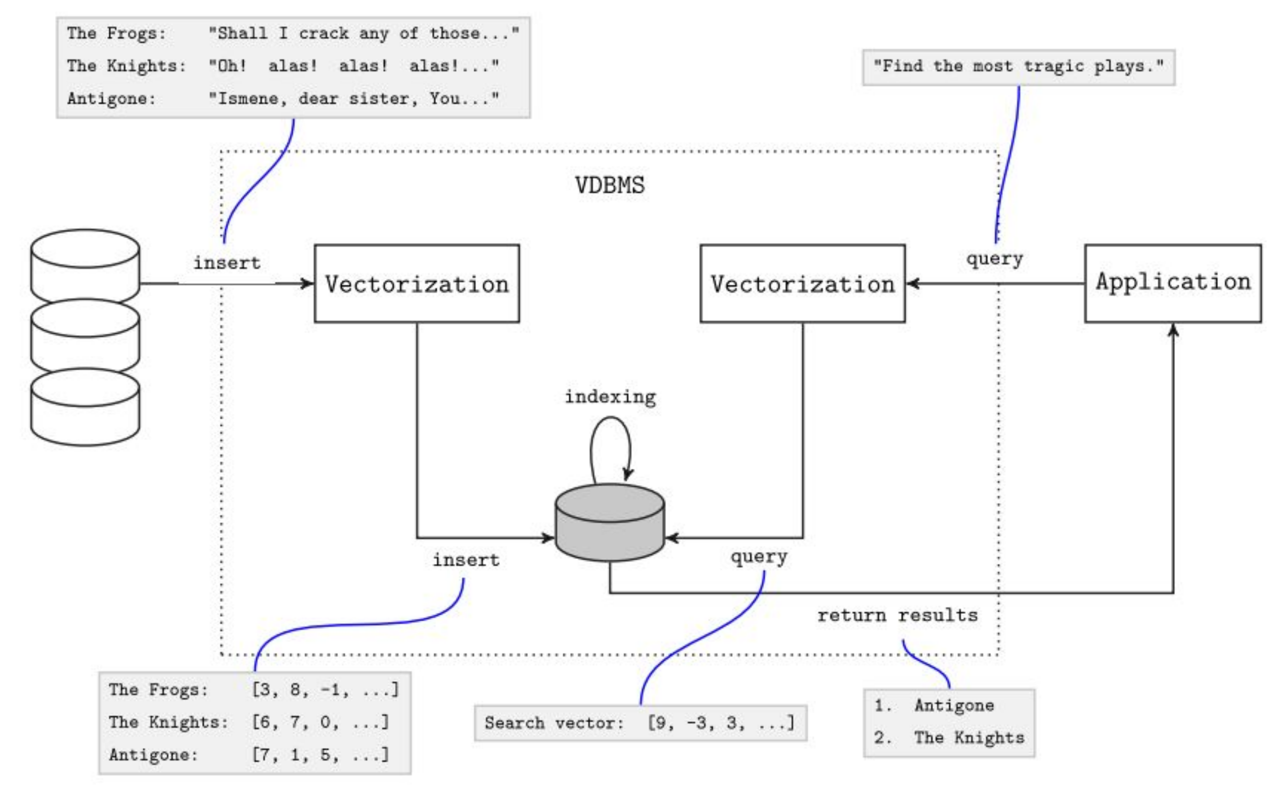
Search Indexes

LSH IVF *k*-d RP KGraph EFANNA NSG

L2H FLANN ANNOY FANNG HNSW NGT

Vector Storage

VDBMS Workflow



The Frogs:	"Shall I crack any of those..."
The Knights:	"Oh! alas! alas! alas!..."
Antigone:	"Ismene, dear sister, You..."

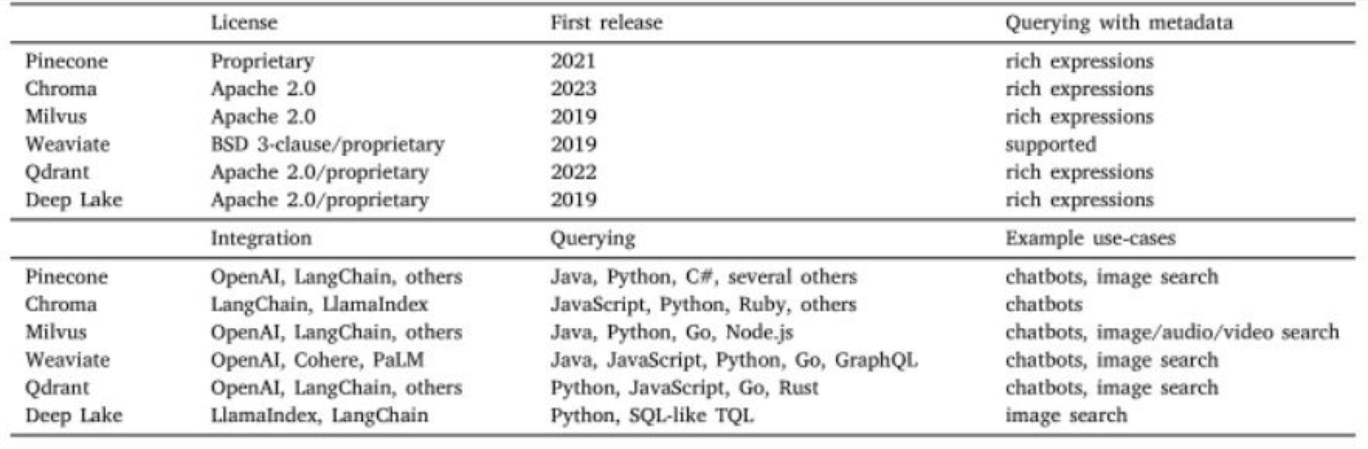
"Find the most tragic plays."

The Frogs:	[3, 8, -1, ...]
The Knights:	[6, 7, 0, ...]
Antigone:	[7, 1, 5, ...]

Search vector: [9, -3, 3, ...]

1. Antigone
2. The Knights

Popular VDBMSs



	License	First release	Querying with metadata
Pinecone	Proprietary	2021	rich expressions
Chroma	Apache 2.0	2023	rich expressions
Milvus	Apache 2.0	2019	rich expressions
Weaviate	BSD 3-clause/proprietary	2019	supported
Qdrant	Apache 2.0/proprietary	2022	rich expressions
Deep Lake	Apache 2.0/proprietary	2019	rich expressions

	Integration	Querying	Example use-cases
Pinecone	OpenAI, LangChain, others	Java, Python, C#, several others	chatbots, image search
Chroma	LangChain, LlamaIndex	JavaScript, Python, Ruby, others	chatbots
Milvus	OpenAI, LangChain, others	Java, Python, Go, Node.js	chatbots, image/audio/video search
Weaviate	OpenAI, Cohere, PaLM	Java, JavaScript, Python, Go, GraphQL	chatbots, image search
Qdrant	OpenAI, LangChain, others	Python, JavaScript, Go, Rust	chatbots, image search
Deep Lake	LlamaIndex, LangChain	Python, SQL-like TQL	image search

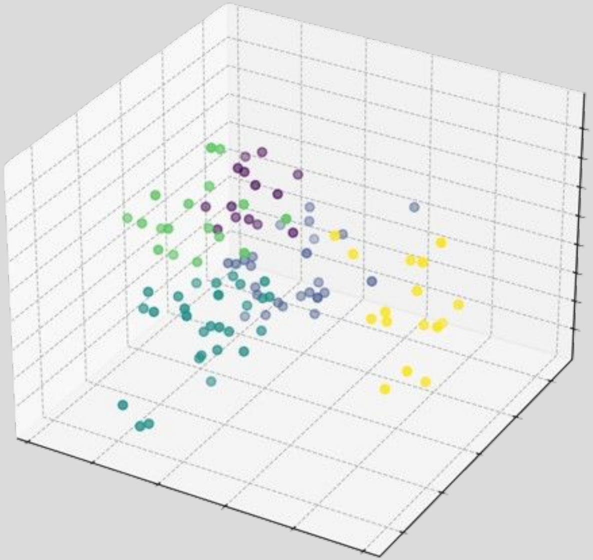
03. Similarity Search

We can perform similarity search on **anything that can be vectorised**:

- Documents
- Images and videos
- Audio

In practice, this means that VDBMSs can be used (and have been used) in:

- Storing and comparing molecular structures
- Recommendation systems
- Facial recognition
- Speech recognition

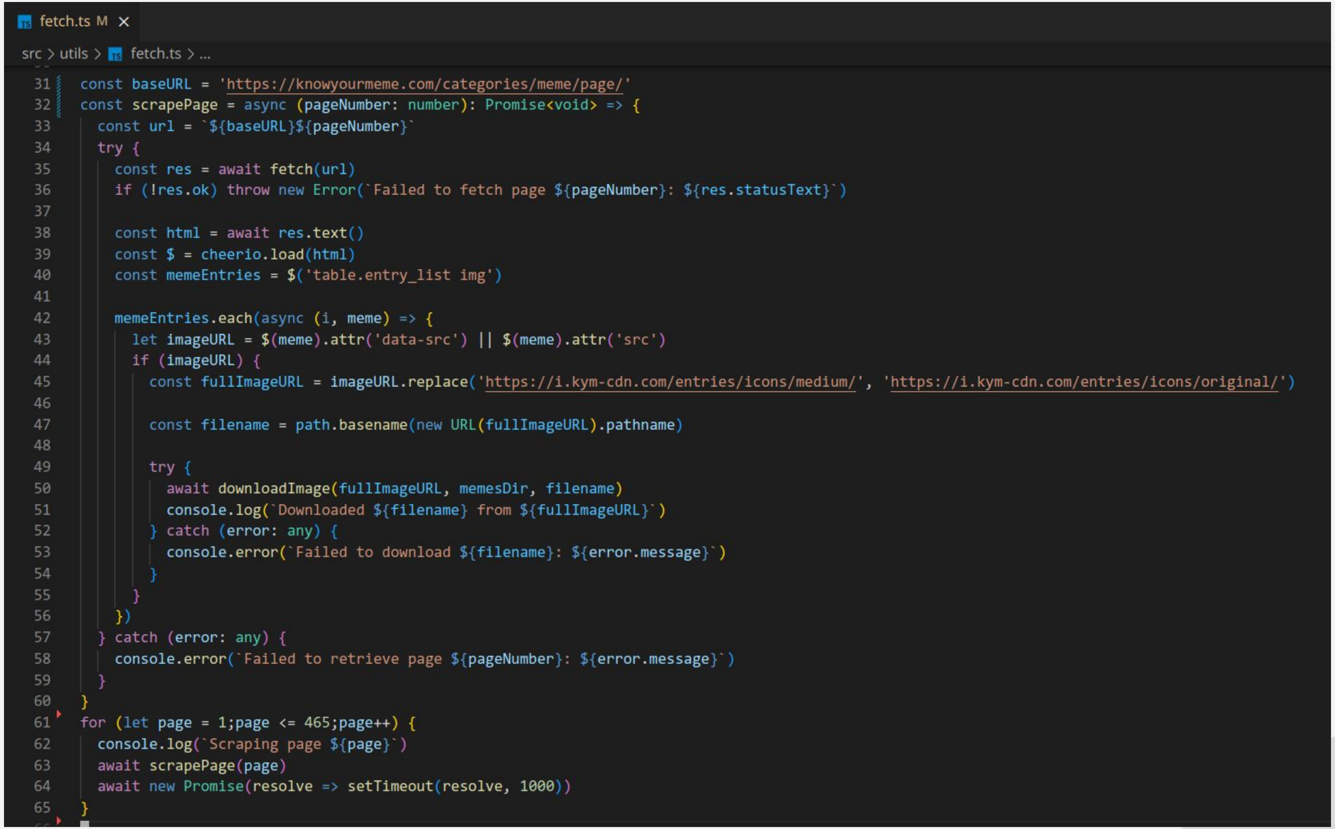


04. Image Search Demo

- **Idea**: a **meme search** app, where the user uploads an image and the app returns similar memes.
- **Implementation**: We shall use **Weaviate** to vectorise and store the memes and their embeddings. The embedding model, **sentence-transformers/clip-ViT-B-32-multilingual-v1**, is open-source and will be **run locally**. The memes will be **scraped** from **knowyourmeme.com**,
- **Repositories**:

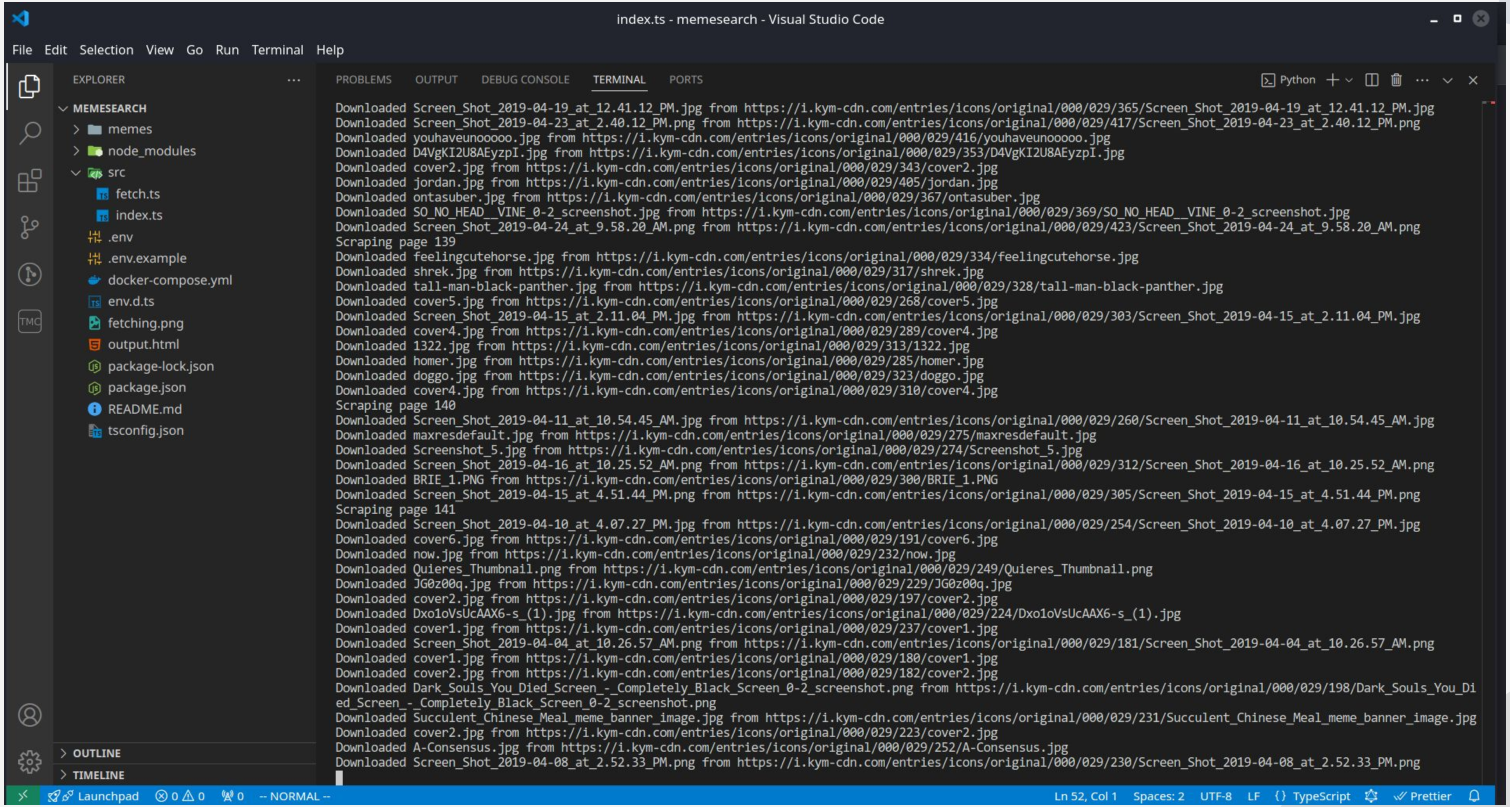
+ **Backend**: https://github.com/ElliotAtHelsinki/memesearch-server
+ **Frontend**: https://github.com/ElliotAtHelsinki/memesearch-client

Let's scrape the memes



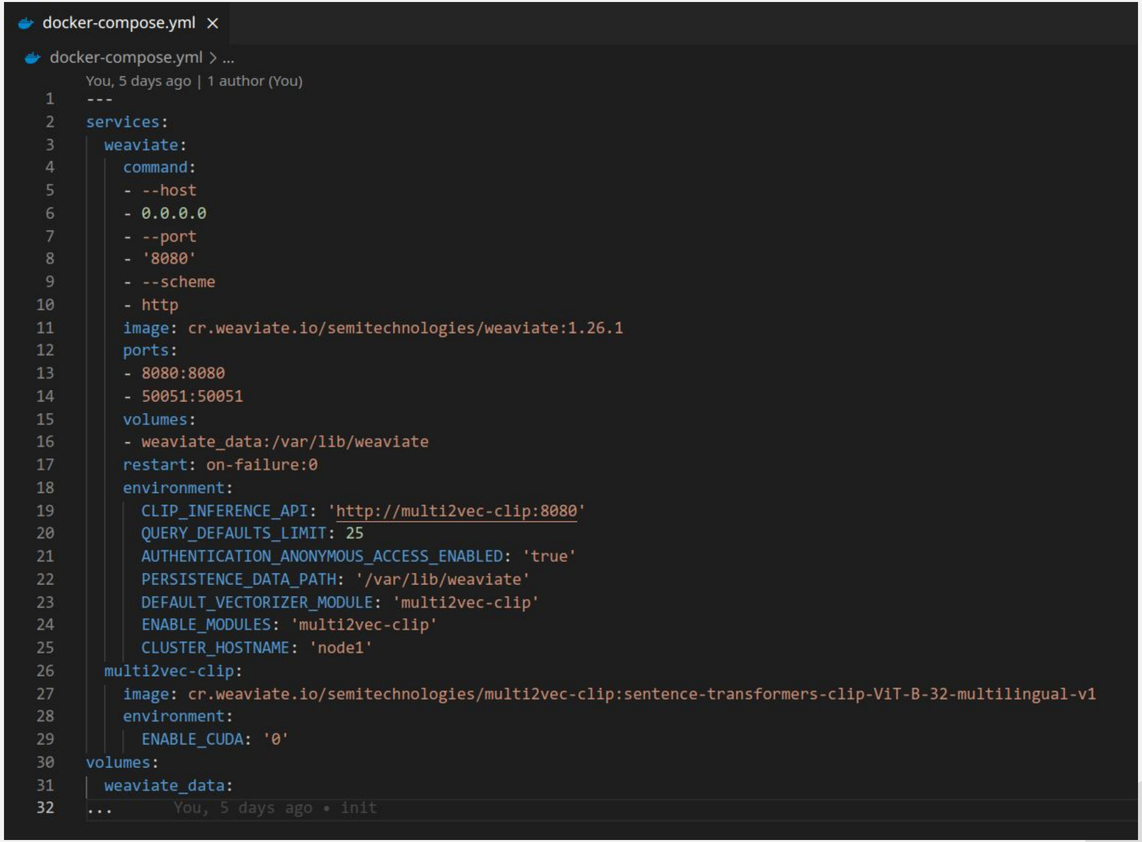
```typescript
const baseURL = 'https://knowyourmeme.com/categories/meme/page/'
const scrapePage = async (pageNumber: number): Promise<void> => {
  const url = `${baseURL}${pageNumber}`
  try {
    const res = await fetch(url)
    if (!res.ok) throw new Error(`Failed to fetch page ${pageNumber}: ${res.statusText}`)

    const html = await res.text()
    const $ = cheerio.load(html)
    const memeEntries = $('table.entry_list img')

    memeEntries.each(async (i, meme) => {
      let imageURL = $(meme).attr('data-src') || $(meme).attr('src')
      if (imageURL) {
        const fullImageURL = imageURL.replace('https://i.kym-cdn.com/entries/icons/medium/', 'https://i.kym-cdn.com/entries/icons/original/')

        const filename = path.basename(new URL(fullImageURL).pathname)

        try {
          await downloadImage(fullImageURL, memesDir, filename)
          console.log(`Downloaded ${filename} from ${fullImageURL}`)
        } catch (error: any) {
          console.error(`Failed to download ${filename}: ${error.message}`)
        }
      }
    })
  } catch (error: any) {
    console.error(`Failed to retrieve page ${pageNumber}: ${error.message}`)
  }
}
for (let page = 1;page <= 465;page++) {
  console.log(`Scraping page ${page}`)
  await scrapePage(page)
  await new Promise(resolve => setTimeout(resolve, 1000))
}
```

Let's scrape the memes



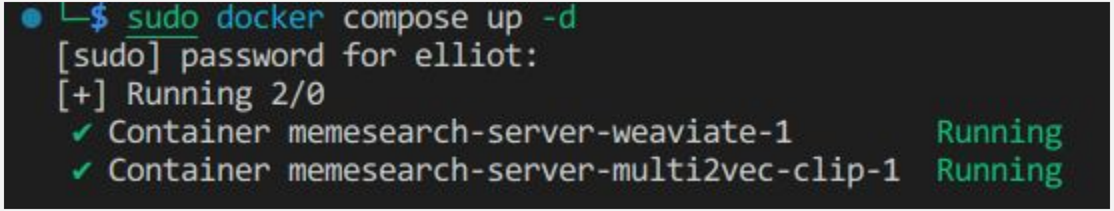
```
Downloaded Screen_Shot_2019-04-19_at_12.41.12_PM.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/365/Screen_Shot_2019-04-19_at_12.41.12_PM.jpg
Downloaded Screen_Shot_2019-04-23_at_2.40.12_PM.png from https://i.kym-cdn.com/entries/icons/original/000/029/417/Screen_Shot_2019-04-23_at_2.40.12_PM.png
Downloaded youhaveunooooo.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/416/youhaveunooooo.jpg
Downloaded D4VgKI2U8AEyzpI.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/353/D4VgKI2U8AEyzpI.jpg
Downloaded cover2.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/343/cover2.jpg
Downloaded jordan.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/405/jordan.jpg
Downloaded ontasuber.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/367/ontasuber.jpg
Downloaded SO_NO_HEAD__VINE_0-2_screenshot.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/369/SO_NO_HEAD__VINE_0-2_screenshot.jpg
Downloaded Screen_Shot_2019-04-24_at_9.58.20_AM.png from https://i.kym-cdn.com/entries/icons/original/000/029/423/Screen_Shot_2019-04-24_at_9.58.20_AM.png
Scraping page 139
Downloaded feelingcutehorse.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/334/feelingcutehorse.jpg
Downloaded shrek.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/317/shrek.jpg
Downloaded tall-man-black-panther.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/328/tall-man-black-panther.jpg
Downloaded cover5.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/268/cover5.jpg
Downloaded Screen_Shot_2019-04-15_at_2.11.04_PM.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/303/Screen_Shot_2019-04-15_at_2.11.04_PM.jpg
Downloaded cover4.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/289/cover4.jpg
Downloaded 1322.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/313/1322.jpg
Downloaded homer.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/285/homer.jpg
Downloaded doggo.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/323/doggo.jpg
Downloaded cover4.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/310/cover4.jpg
Scraping page 140
Downloaded Screen_Shot_2019-04-11_at_10.54.45_AM.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/260/Screen_Shot_2019-04-11_at_10.54.45_AM.jpg
Downloaded maxresdefault.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/275/maxresdefault.jpg
Downloaded Screenshot_5.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/274/Screenshot_5.jpg
Downloaded Screen_Shot_2019-04-16_at_10.25.52_AM.png from https://i.kym-cdn.com/entries/icons/original/000/029/312/Screen_Shot_2019-04-16_at_10.25.52_AM.png
Downloaded BRIE_1.PNG from https://i.kym-cdn.com/entries/icons/original/000/029/300/BRIE_1.PNG
Downloaded Screen_Shot_2019-04-15_at_4.51.44_PM.png from https://i.kym-cdn.com/entries/icons/original/000/029/305/Screen_Shot_2019-04-15_at_4.51.44_PM.png
Scraping page 141
Downloaded Screen_Shot_2019-04-10_at_4.07.27_PM.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/254/Screen_Shot_2019-04-10_at_4.07.27_PM.jpg
Downloaded cover6.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/191/cover6.jpg
Downloaded now.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/232/now.jpg
Downloaded Quieres_Thumbnail.png from https://i.kym-cdn.com/entries/icons/original/000/029/249/Quieres_Thumbnail.png
Downloaded JG0z00q.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/229/JG0z00q.jpg
Downloaded cover2.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/197/cover2.jpg
Downloaded Dxo1oVsUcAAX6-s_(1).jpg from https://i.kym-cdn.com/entries/icons/original/000/029/224/Dxo1oVsUcAAX6-s_(1).jpg
Downloaded cover1.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/237/cover1.jpg
Downloaded Screen_Shot_2019-04-04_at_10.26.57_AM.png from https://i.kym-cdn.com/entries/icons/original/000/029/181/Screen_Shot_2019-04-04_at_10.26.57_AM.png
Downloaded cover1.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/180/cover1.jpg
Downloaded cover2.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/182/cover2.jpg
Downloaded Dark_Souls_You_Died_Screen_-_Completely_Black_Screen_0-2_screenshot.png from https://i.kym-cdn.com/entries/icons/original/000/029/198/Dark_Souls_You_Di
ed_Screen_-_Completely_Black_Screen_0-2_screenshot.png
Downloaded Succulent_Chinese_Meal_meme_banner_image.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/231/Succulent_Chinese_Meal_meme_banner_image.jpg
Downloaded cover2.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/223/cover2.jpg
Downloaded A-Consensus.jpg from https://i.kym-cdn.com/entries/icons/original/000/029/252/A-Consensus.jpg
Downloaded Screen_Shot_2019-04-08_at_2.52.33_PM.png from https://i.kym-cdn.com/entries/icons/original/000/029/230/Screen_Shot_2019-04-08_at_2.52.33_PM.png
```

Setting up Weaviate



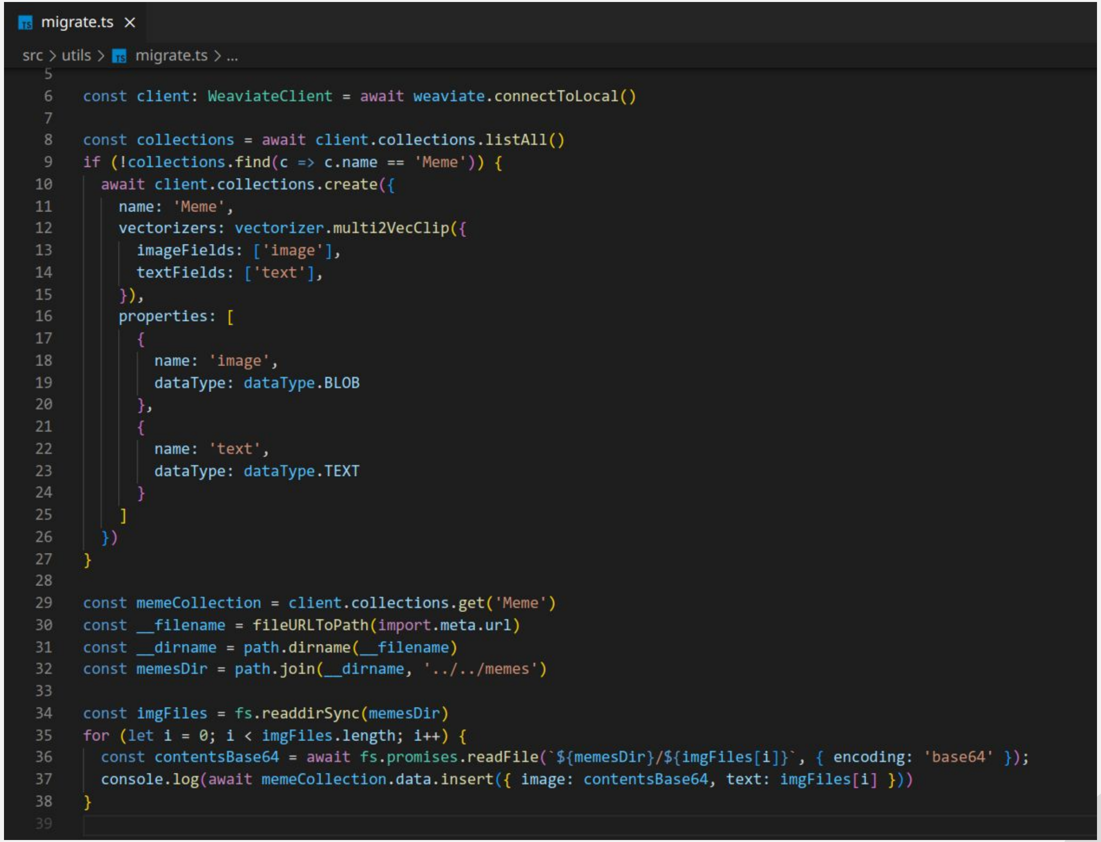
```yaml
---
services:
  weaviate:
    command:
    - --host
    - 0.0.0.0
    - --port
    - '8080'
    - --scheme
    - http
    image: cr.weaviate.io/semitechnologies/weaviate:1.26.1
    ports:
    - 8080:8080
    - 50051:50051
    volumes:
    - weaviate_data:/var/lib/weaviate
    restart: on-failure:0
    environment:
      CLIP_INFERENCE_API: 'http://multi2vec-clip:8080'
      QUERY_DEFAULTS_LIMIT: 25
      AUTHENTICATION_ANONYMOUS_ACCESS_ENABLED: 'true'
      PERSISTENCE_DATA_PATH: '/var/lib/weaviate'
      DEFAULT_VECTORIZER_MODULE: 'multi2vec-clip'
      ENABLE_MODULES: 'multi2vec-clip'
      CLUSTER_HOSTNAME: 'node1'
  multi2vec-clip:
    image: cr.weaviate.io/semitechnologies/multi2vec-clip:sentence-transformers-clip-ViT-B-32-multilingual-v1
    environment:
      ENABLE_CUDA: '0'
volumes:
  weaviate_data:
...
```

Setting up Weaviate



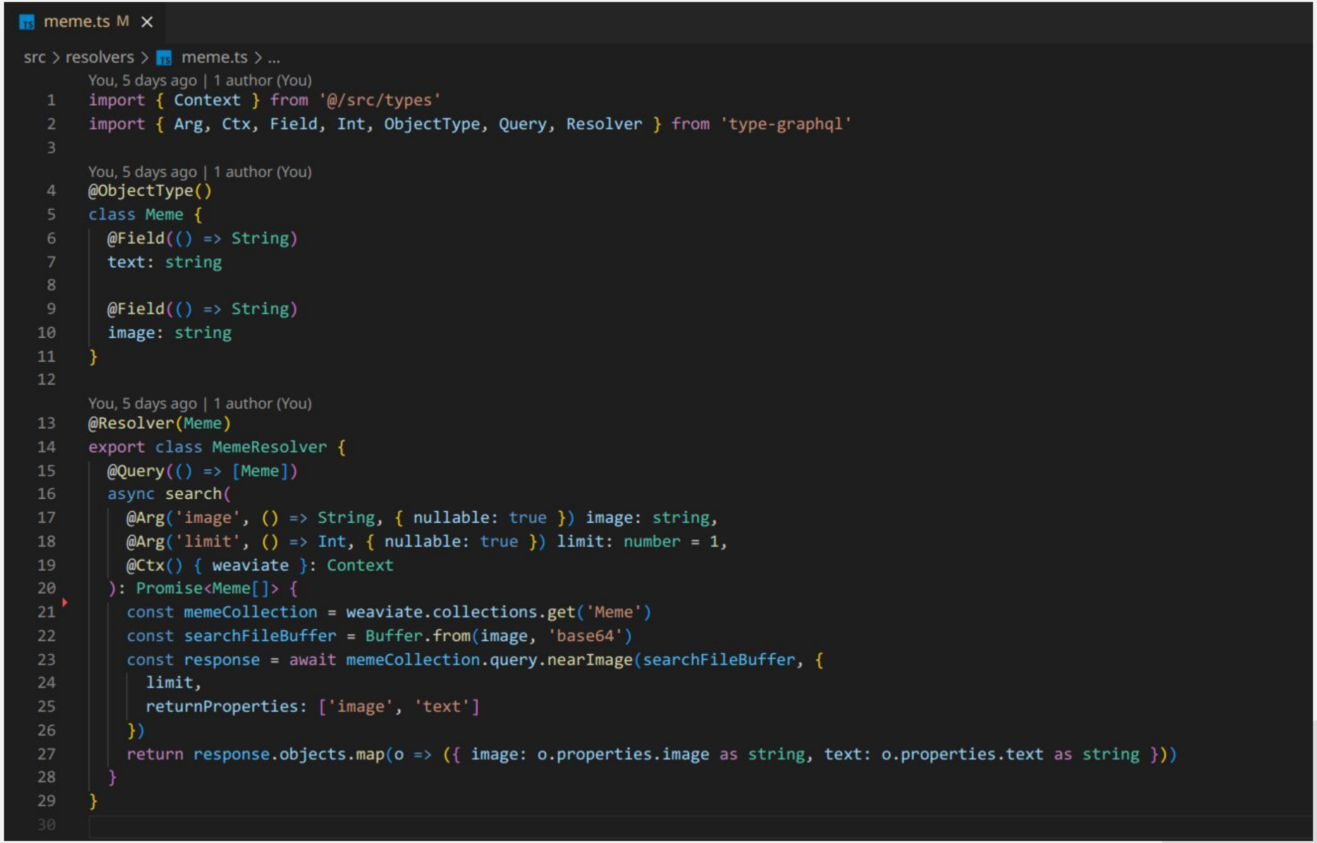
```
● └─$ sudo docker compose up -d
  [sudo] password for elliot:
  [+] Running 2/0
   ✔ Container memesearch-server-weaviate-1        Running
   ✔ Container memesearch-server-multi2vec-clip-1  Running
```

Vectorising the memes



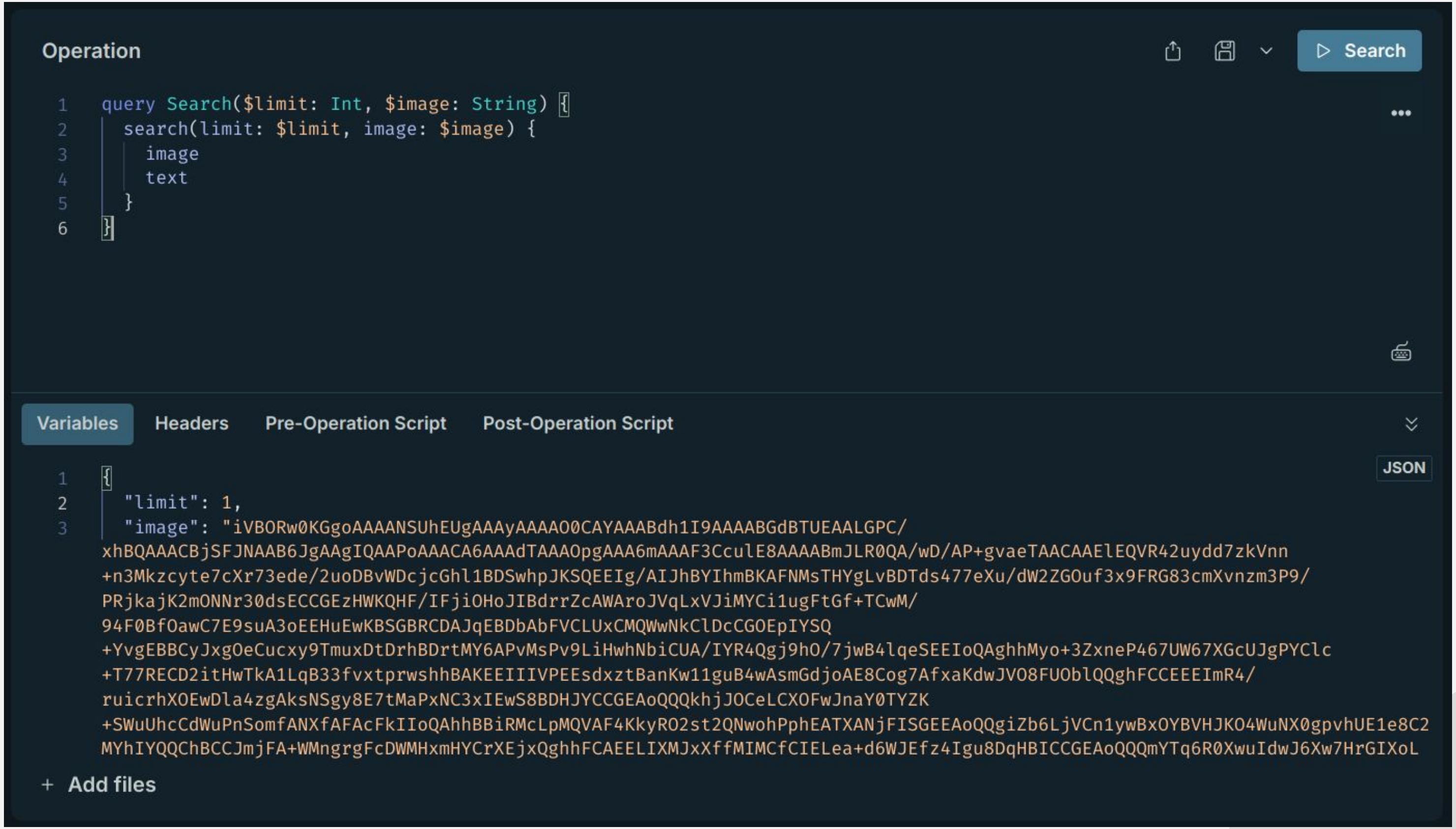
```typescript
const client: WeaviateClient = await weaviate.connectToLocal()

const collections = await client.collections.listAll()
if (!collections.find(c => c.name == 'Meme')) {
  await client.collections.create({
    name: 'Meme',
    vectorizers: vectorizer.multi2VecClip({
      imageFields: ['image'],
      textFields: ['text'],
    }),
    properties: [
      {
        name: 'image',
        dataType: dataType.BLOB
      },
      {
        name: 'text',
        dataType: dataType.TEXT
      }
    ]
  })
}

const memeCollection = client.collections.get('Meme')
const __filename = fileURLToPath(import.meta.url)
const __dirname = path.dirname(__filename)
const memesDir = path.join(__dirname, '../../memes')

const imgFiles = fs.readdirSync(memesDir)
for (let i = 0; i < imgFiles.length; i++) {
  const contentsBase64 = await fs.promises.readFile(`${memesDir}/${imgFiles[i]}`, { encoding: 'base64' });
  console.log(await memeCollection.data.insert({ image: contentsBase64, text: imgFiles[i] }))
}
```

Querying the memes


meme.ts M
src > resolvers > meme.ts > ...
You, 5 days ago | 1 author (You)

```ts
import { Context } from '@/src/types'
import { Arg, Ctx, Field, Int, ObjectType, Query, Resolver } from 'type-graphql'

@ObjectType()
class Meme {
  @Field(() => String)
  text: string

  @Field(() => String)
  image: string
}

@Resolver(Meme)
export class MemeResolver {
  @Query(() => [Meme])
  async search(
    @Arg('image', () => String, { nullable: true }) image: string,
    @Arg('limit', () => Int, { nullable: true }) limit: number = 1,
    @Ctx() { weaviate }: Context
  ): Promise<Meme[]> {
    const memeCollection = weaviate.collections.get('Meme')
    const searchFileBuffer = Buffer.from(image, 'base64')
    const response = await memeCollection.query.nearImage(searchFileBuffer, {
      limit,
      returnProperties: ['image', 'text']
    })
    return response.objects.map(o => ({ image: o.properties.image as string, text: o.properties.text as string }))
  }
}
```

Querying the memes



```graphql
query Search($limit: Int, $image: String) {
  search(limit: $limit, image: $image) {
    image
    text
  }
}
```

```json
{
  "limit": 1,
  "image": "iVBORw0KGgoAAAANSUhEUgAAAyAAAAO0CAYAAABdh1I9AAAABGdBTUEAALGPC/
xhBQAAACBjSFJNAAB6JgAAgIQAAPoAAACA6AAAdTAAAOpgAAA6mAAAF3CculE8AAAABmJLR0QA/wD/AP+gvaeTAACAAElEQVR42uydd7zkVnn
+n3Mkzcyte7cXr73ede/2uoDBvWDcjcGhl1BDSwhpJKSQEEIg/AIJhBYIhmBKAFNMsTHYgLvBDTds477eXu/dW2ZGOuf3x9FRG83cmXvnzm3P9/
PRjkajK2mONNr30dsECCGEzHWKQHF/IFjiOHoJIBdrrZcAWAroJVqLxVJiMYCi1ugFtGf+TCwM/
94F0BfOawC7E9suA3oEEHuEwKBSGBRCDAJqEBDbAbFVCLUxCMQWwNkClDcCGOEpIYSQ
+YvgEBBCyJxgOeCucxy9TmuxDtDrhBDrtMY6APvMsPv9LiHwhNbiCUA/7UgW67ghhhMyo+3ZxneP467UW67XGcUJgPYClc
+T77RECD2itHwTkA1LqB33fvxtprwshshBAKEEIIVPEEsdxztBanKwKw11gxdqFNk/Glu8KMjxKKMLK+VtDnaY0TYZK
+SWuUhcCdWuPnSomfANXfAFAcFkIIoQAhhhBBiRMcLpMQVAF4VF/HH4KkOZVcJ0WuNX0gpvhUE1E1e1e8C2
MYhIYQQChBCCJmjFA+WMngrgFcWD4WMngrgFcDWMHxmHYCrXEjxQghhFCAEELLIXMJxXffMIMCfCIELea+d6WJEfz4Igu8DqHBIBICCGEAoQQQmYTQ6R0XwuIdwJ6Xw7HrGIXoL"
```

Querying the memes

```
  {
    image: '/9j/4AAQSkZJRgABAQEAYABgAAD/4QAWRXhpZgAASUkqAAgAAAAAAAAAAAAD/2wBDAAgGBgcGBQgHBwcJCQgKDBQNDAsLDBkSEw8UHRofHh0aHBwgJC4nICIsIxwcKDcpLDAxNDQ0Hyc5PTgyPC4zNDL/
2wBDAQkJCQwLDBgNDRgyIRwhMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjL/wAARCADIAMgDASIAAhEBAxEB/8QAHwAAAQUBAQEBAQEAAAAAAAAAAAECAwQFBgcICQoL/8QA
tRAAAgEDAwIEAwUFBAQAAAF9AQIDAAQRBRIhMUEGE1FhByJxFDKBkaEII0KxwRVS0fAkM2JyggkKFhcYGRolJicoKSo0NTY3ODk6Q0RFRkdISUpTVFVWV1hZWmNkZWZnaGlqc3R1dnd4eXqDhIWGh4iJipKTlJWWl5iZ
mqKjpKWmp6ipqrKztLW2t7i5usLDxMXGx8jJytLT1NXW19jZ2uHi4+Tl5ufo6erx8vP09fb3+Pn6/8QAHwEAAwEBAQEBAQEBAQAAAAAAAAECAwQFBgcICQoL/8QAtREAAgECBAQDBAcFBAQAAQJ3AAECAxEEBSExBhJB
UQdhcRMiMoEIFEKRobHBCSMzUvAVYnLRChYkNOEl8RcYGRomJygpKjU2Nzg5OkNERUZHSElKU1RVVldYWVpjZGVmZ2hpanN0dXZ3eHl6goOEhYaHiImKkpOUlZaXmJmaoqOkpaanqKmqsrO0tba3uLm6wsPExcbHyMnK
0tPU1dbX2Nna4uPk5ebn6Onq8vP09fb3+Pn6/9oADAMBAAIRAxEAPwDxaiiivUMwooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAoort9D+GGsajZ/2lq8sOg6QMbrvUDszngbUO
CecdcZzwTScktxnEVu+DdT0nR/Fdlf65Y/btNi3+db+Uku/KMF+VyAcMQefStnX9R8E6fos+jeG9Pmv7mbaJNYvflYYYN+6THGeVJwDj1q/4Q8ZRXOr6DoVx4R8Kywyz29nJPJpoaZ1LKhYsWwXI5zjr2qW247Acv4y1
PSdY8V3t/odj9h02XZ5Nv5SRbMIob5UJAywJ49awq7P4r2Vpp3xL1e0sbWG1to/J2QwRhEXMKE4UcDkk/jXXeAtI8M3Hwc1vU/ENlEyW2oZa4SJfP2qIWEaOeRuPy8EffPTNHMoxTA8eqeys7jUb+2sbSPzLm5lWGJMg
bnYgAZPA5I617T4CvvDfxEvtT0C58HaRYILUzW81rEBKigqp3P1LZYHIx9K4n4beJv7M17SdK/sPRbv7TqcP+l3Vp5lxFuZF+R8/LjGRxwSTRzvXTVBY5PWdGv8Aw/qs+l6pB5F5Bt8yPerbdyhhypI6EHrVGvZfiV45
/sf4gapYf8It4YvvK8r/AEi+0/zZnzEh+ZtwzjOB7AVh/DPwtaeNvFGratqVnH/Z9mDcyWVohVWdyxWNFH8Aw2BnsBzSU/d5pBY82or3jw3AvifXJNG134WQ6VptyjCG6i014XgIUn55SBnOMAjHOBg5ry3SNIjsfihY
aLdCO5jg1qO0lDoCsoWYIcqc8HB496ane4WOZor2f4g6p4V8EeKby10rwvpV9qUwjkmF5AGt7UbFARIhgZIBcnjlx16Dlb6Tw/q/xqsTosFs+h3Go2aJElv5cTKfLDjyyBwTuyCOefWiM7q9gscFRXtHj7VPCfgjxZdW
2m+FtLv7+by5J1uoVNvarsUCOOMAAEgFifVh16DP+JfhrTLyfwbe+H9Oh0+TxFEAIEGxAzeXtyBwP9bg4Hakql7abhY8nor3XX9KXwTc2mjeH/hxHr0UcaPdahd6c9yZSc7lUhcKenPIGcY4rk/i74Z0/R7rRdX02wbT
oNWtTI9iyFDBIoUsCM/KcOoK9ip9aI1E2l3Cxxmk+GtX1yw1K+06086202Lzrt/MRfLTDHOGIJ4RumelZNe9/D7xf/avhLxrd/8ACO+H7P7DYCTybOy8uO4/dzHbKu47l+XGOOGPrXl+s+Of7Y0qew/4RbwxY+bt/wBI
sdP8qZMMD8rbjjOMH2JojOTbVgOUor0lV8A+Nxww8Jaw3Y/NZSH68bOB/sjn+Kua8TeBPEHhNt2o2Ra1P3LuA+ZC3T+IdOvfBqlNN2ejA5uiiiqEFFFFAD4pZIJkmhkaOWNgyOhwVI5BBHQ16JafFJdX09NJ8daWmt2K
nK3CHy7iI+oIwD2HbjqTXnFFKUU9xnba/wCDdHXRZ9f8L6/FqGmwbTPbXHyXUG5ggyuPmG49cD8etY3gn/kffDv/AGE7b/0atYVFCTtZsDuvjH/yVbWv+2H/AKIjrd0H/k23xR/2E0/9Ctq8ooqeT3UuwXPV/wBn3/kf
b7/sGSf+jYq898M3sGmeK9Hv7lttvbX0M0rYJwquCTgc9BWVRT5dW+4HtPxF+G/iDxT4ym1/QEttQ07UEiZZo7mMBAI0XJyRkHGcrniqHwn1Sz8MeIvEPhfWrwWcl7/oguY5SqpLGXXAcY2k7jhuOQO5FeS0VPI+XlbC
57Hc+CPi1FqBhh8R301mScX39sOseOcEgtuH4A9a4XQIbi3+K+mQXdyLq5j1yJJrhZC4lcTgMwY8tk5OT1zXLUU1F9QO6+Mf/JVta/7Yf+iI6wvBP/I++Hf+wnbf+jVrCoqlG0eUDuvjH/yVbWv+2H/oiOus8d6gNI0L
4TakULizto7gqOrbFtmx+leM0VPJt5Bc+gvGOm+J/F08XiL4f+JLqexuIl8yzt9SaEo44yFLBV4xkcHIPUmvLfHOjeKtIj03/hKdWe8mmEhigkvmuHt8bd27JIXOR0Jzt9q4+iiMHELnq3wbmtLvT/FvhyS7it73V7Hy
rXzTgMQkqnHqRvBx1wD6Gua1v4YeKfD2lXmp6pZwwWlqVBk+0I3mZcKCoBJ6kdQOK46ijlaldMD0lfCnhHwaPM8Yap/aeor/AMwjTHyFPpJJxjqDj5Tx/FWb4l+KGs65p/8AZNlHDpGihPLWytBgFP7rN1I9hgHPINcR
RTUOr1C4UUUVQgooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAK
KKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKA
CiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooq9baJq17YTX9ppl7PZwbvNuIrd2jj2jJ3MBgYBBOe1AF
GiiigAooooAKKKKACiiigAooooAKKKKACiiigAorr/Anw+vvHV1Klve2lrBCQJXkcNIMj+GMHJ+pwPfiujv9d8N/Di+l0/w7ocl7rdu22TUtXjI8tuuY4+McE8/L2+8Klz1stWOx5lc2txZzmC6g
lgmUAmOVCrAEAjg+oIP41veBfDf/AAlPiu1sJDts0zPdvnASFeWOe2eBn3rI1XVr7XNTm1LUrhri8nIMkrAAtgADgcdAB+Fd1/yJXwp/u6v4o/OOzX8x82fbIb2ok3a3VgYfi3V7vx142nl020muEP7ixtoIi7CFAcAK
BnoCxHbJ9K29J1Lx54W8FatoSeFbpdMu1me5nudPnDRq8YRiGyFACrnJHFcDZ3t3p12l3Y3U1rcx52TQSFHXIwcMORwSPxr174da3q2seAviF/amp3t95Wmfu/tVw8uzMU+cbicZwPyFTNcsfIEeNVpv4b12PUF099F1
Fb108xbdrVxIUzjcFxnGQRmsyve/jZ4z1DQNYttM0dzZXNzaLJc3sQxK8e9wkYbqoB3njnkc9cuUmmkuoHiGoaVqOkyrFqWn3VnIwyqXMLRkj2DAV2Xh/wCFOt6xpGt3l5bahps2nwebb282nvuvG2udiZIOcqBwD94c
evSeN9Rude+AvhjV9RcTXzX5iaYj5iB5y8+5CLn3FT/DPxFrd/4K8fXF5rGoXE1rpwe3kmuXdoW8uc5Qk5U5A5HoKhzly3QWPLLzwv4g060e7vtC1O1to8b5p7SREXJwMsRgckD8aqafpWo6tK0WnWF1eSKMslvC0hA9
woNW7zxR4g1G0e0vtd1O6tpMb4Z7uR0bByMqTg8gH8K9qXSbjRPhP4fs9B8R6VoF1qUS3d1dX12LeSfcgYqjBSeN6jIwQAOeealNx3A8K1DStR0mVYtSsLqzkYZVLmFoyR7BgKLTS9Qv7e5uLOwuriG1XfcSQws6wrgn
LkDCjAPJ9DXsHiq4huPg7PY694p0TWdbsblJLJ7O/E0jIWVfmzhmbDSZ46AE8jNVvgpc29n4a8dXV3bi5tobOOSWBukqBJyVP1AI/Gl7R8rdgseYL4c1x9POoJouotZBDIbkWrmMKBktuxjGO9V7TS9Qv7e5uLOwuriG
1XfcSQws6wrgnLkDCjAPJ9DXXx/FTXp5tabVZZbyDUdPmsktVmMUNuZOjqgBB2jI9SD96uo+Clzb2fhrx1dXduLm2hs45JYG6SoEnJU/UAj8ablKKbaA8xi8Na9PZC9i0TUpLQruE6WjlNvruxjFZiI0jqiKWdjhVUZJ
PoK9g+GnxA8Sa98UbSC/1BpLS9WVWtQMRRhY3dQi9BgqBnqR1JrT+Hmi2C/ETx3q8gt4zo9xMtt9oIWKHdJL85OPlAEeM9gTSdRxvdBY8gufDHiCytWurrQtTgt1GTLLaSKgHrkjFVdP0vUNWuGt9NsLq9mVd7R20LSM
FyBkhQTjJHPvXuvhK+1nTfEzXfiL4j+GNR0y4Vlubf8AtUP/AAnaY0ICrzjOMDGeOleNX17L4f8AFWq/8I7qU9tAlzNDDPZ3LLvhDnbh1PzKQFPXnANOM29AOj8QfCnW9H0jRLyzttQ1KbUIPNuLeHT33WbbUOx8EnOW
I5A+6ePTkbrQdYsp3hu9JvreWOH7Q6S27oyxZxvII4XPGema9Z+JniLW7DwV4BuLPWNQt5rrTi9xJDcujTN5cBy5ByxyTyfU15vp3jjxJpmuJrEerXM96sXkF7pzNui3bjGd2flz2og5ONwz1Fer6Unhb4p3n2GXSZt
E8ROpb7Tp0Re2kI6s6fw5J/llucVxPjDwfe+DNX/ALPvbi1nLAsjwShjtz/EvVT9R9CcVSmm7PcLHPUUUVQiW2uriyuEuLWeWCeM5SWJyrKfYjkV39r8S4dYtY9P8daRFrVsgwl3GBHdRDrwwxnoOM45Jrzuik4p7jO
10TwrpPiH4irpej3c1xoMeLiW5nXY6wqoZ93TBySmR6g1m+OvEn/AAlPiu6v4xts0xBaJjASFeFGO2eTj3rO03XtT0iy1C0sbnyYdQi8m5AjUl05+XJGQOT0IzWbSUdbsAr0H4feJdI0Pwl41sdRu/JudSsBDaJ5bt5j
+XMMZUEDl1646159RTlHmVmAV6D8YfEukeKfFtpfaLd/araOwSFn8t0w4kkJGGAPRh+defUUON2mB6Dq/iXSLr4I6D4ehu92q2t+000HluNqFpyDuI2n769D3+tJ8MfFmj6CNc0nXxKmm6zbeRJPECxjwHHIHOCHbkZw
ccdx5/RS5FZoLnY63ofgmy0q8n0rxhNqV4Cv2W1/s+SHjeM7nYYOFJPbkfhXTab4w8K+KfAdn4Y8YT3GnzadtFrewRlwVAwMgAnpwRjnAOc15RRScLrVhc7DV9H8D6foNy+n+KbrVtVyv2dEsXt4gNw3bt4JPG7oRzit
H4feJdI0Pwl41sdRu/JudSsBDaJ5bt5j+XMMZUEDl1646159RTcbqzYBXoPw+8S6RofhLxrY6jd+Tc6lYCG0Ty3bzH8uYYyoIHLr1x1rz6inKPMrMDq/hrrNh4f+IGl6pqk/kWcHm+ZJSZtu6J1HCgnqQOldH4b+Itj4
d+IviO8kja70LWLuYyFFIbYZGKOAcHoxyDg4PqMV5jRScE3qFz0a50D4XyagbiDxreRWRJJtP7OlaQdeBIRj06g1wWoLZpqV0unvK9kJnFu0o+do8naWx3xjNVqKajbqB6xc694N8a+BNC03W9Yn0TU9Hh+zxv8AZXuE
dQEUnCDkEIvUggg9R15qPTvAVl4iVZ9cvtR0mKz813gtmiee4D48oBh8qleck/jXGUVKhbZhc77U/iddRWL6T4TsIfD2lng/Z+Z5eMZeTrnGORzx1NcHJI8sjSSOzu5LMzHJJPUk02iqUUtgCiimIKKKKACiiigAooo
oAKKKKACiiigAooooAKKKKACiiigAooooAKKKKAP/9k=',
    text: 'kymdbstyleguide3.JPG'
  }
```

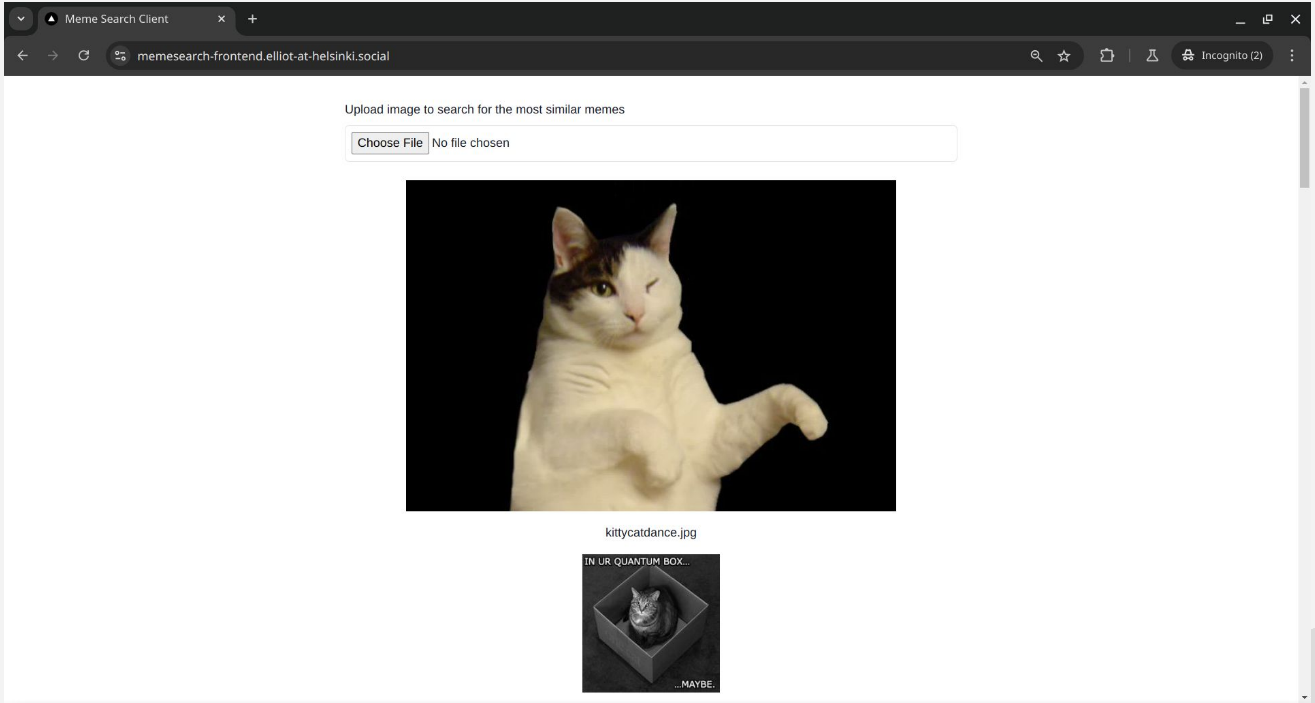
Moment of truth!

Visit **https://memesearch-frontend.elliot-at-helsinki.social/** and upload any image of your choice to test it out!

For the image:

Moment of truth!

We get back:



Upload image to search for the most similar memes

kittycatdance.jpg

05. Usage with LLMs

a. VecDB as a semantic cache:

- LLM-based chatbots often use external APIs.

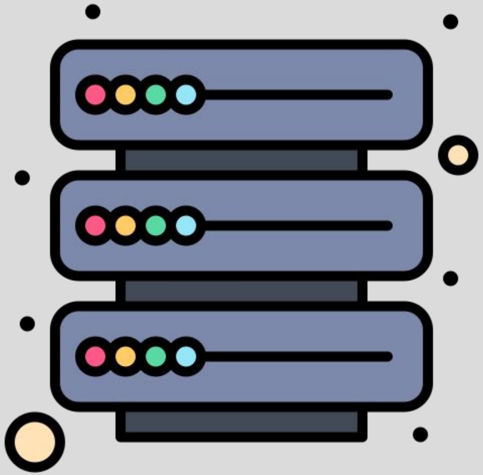
- Very often, customers might make the same query repeatedly, or many customers might ask the same question.

-> Invoking API calls every time is too costly.

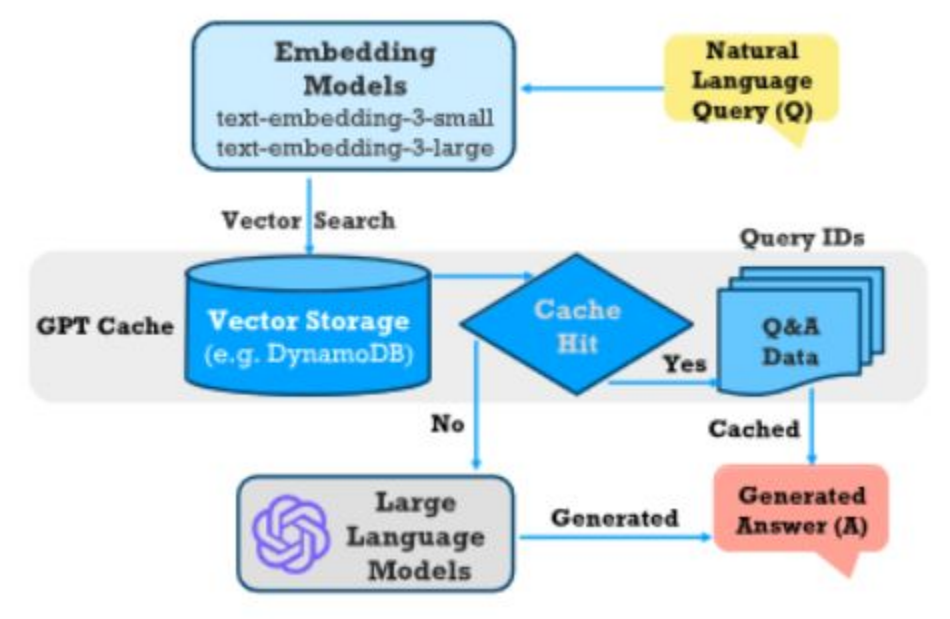
=> Solution: Semantic cache, which stores previous questions and answers.

=> Asked questions would not invoke new API calls.

=> Reduced response time.



VecDB as Semantic Cache



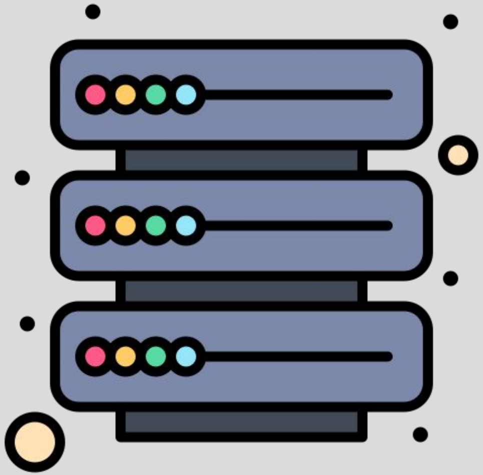
05. Usage with LLMs

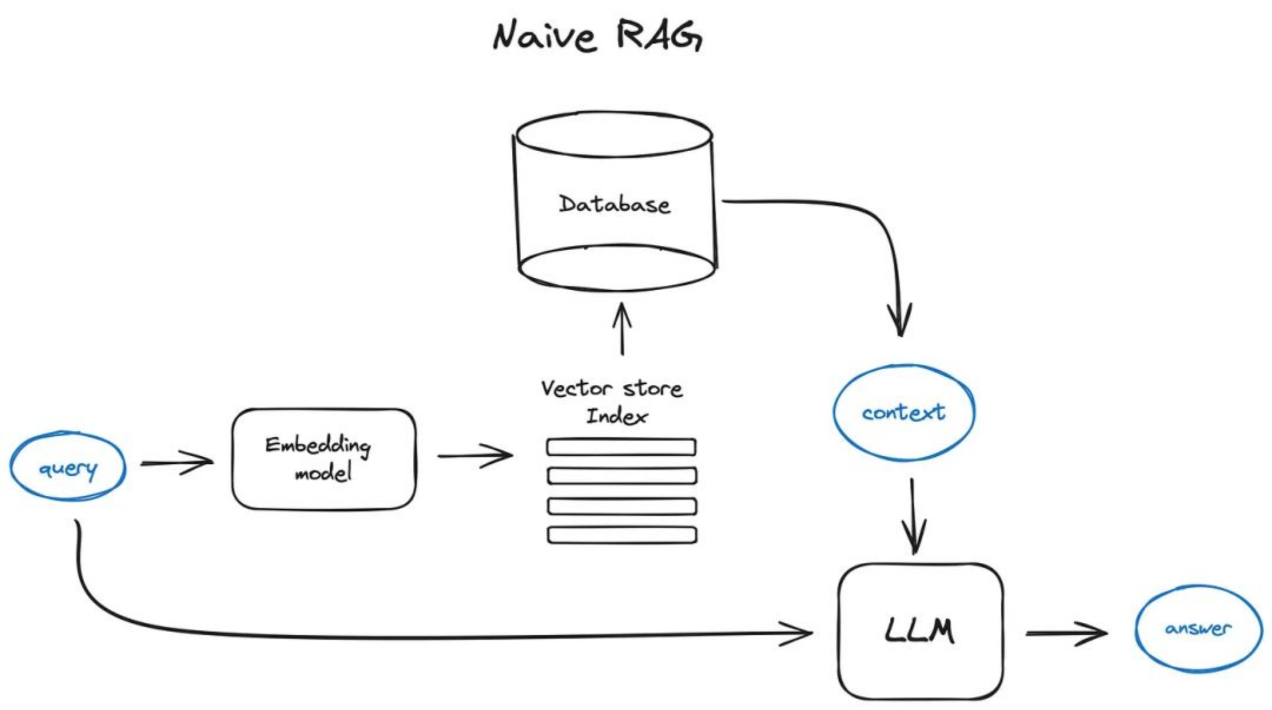
b. VecDB to power Retrieval-Augmented Generation (RAG):

Two major problems of LLMs:

- **Hallucination**
- New influx of information **in real-time**

=> Solution: VecDB as a dynamic external memory for LLMs.

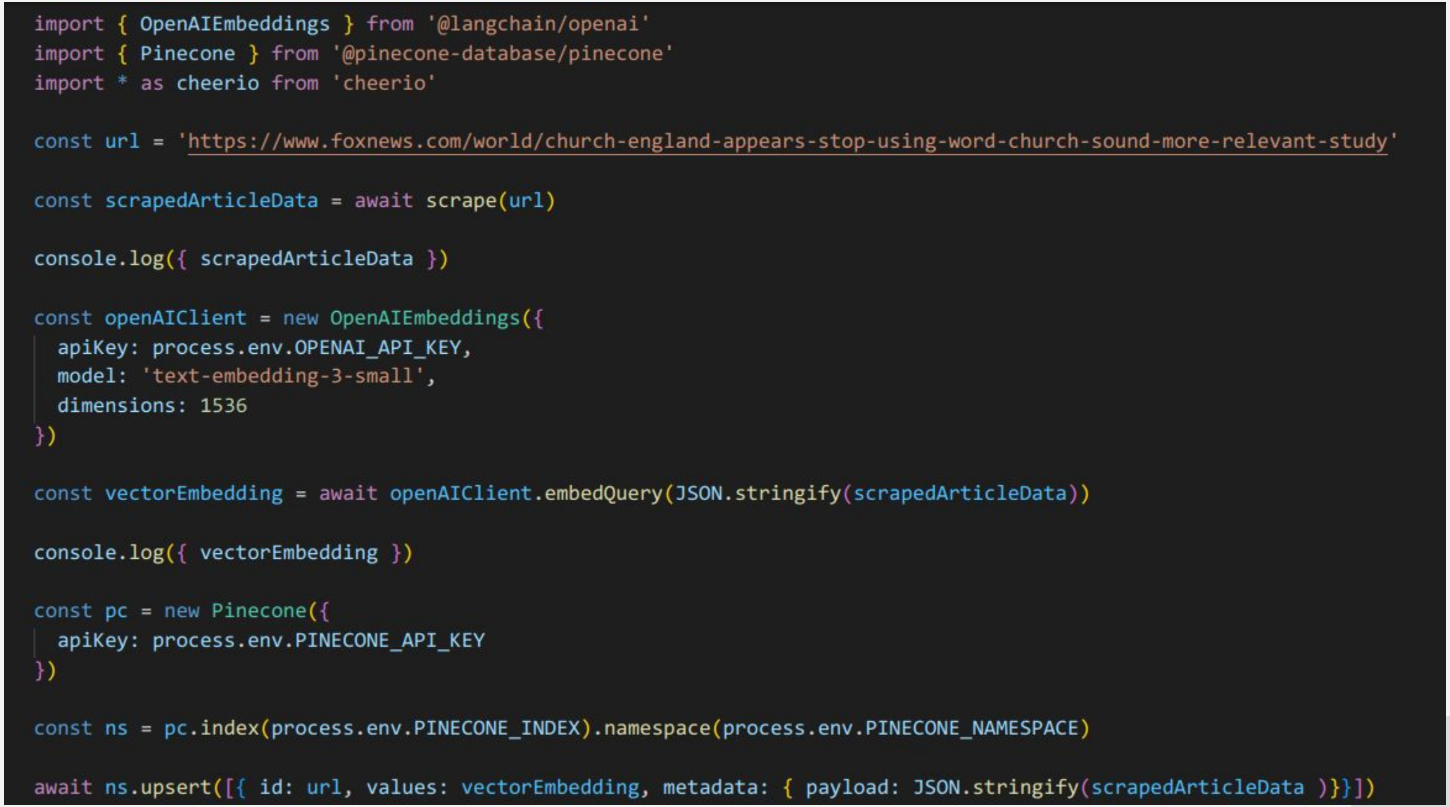



Naive RAG
Database
Vector store Index
query
Embedding model
context
LLM
answer

06. Naïve RAG Demo

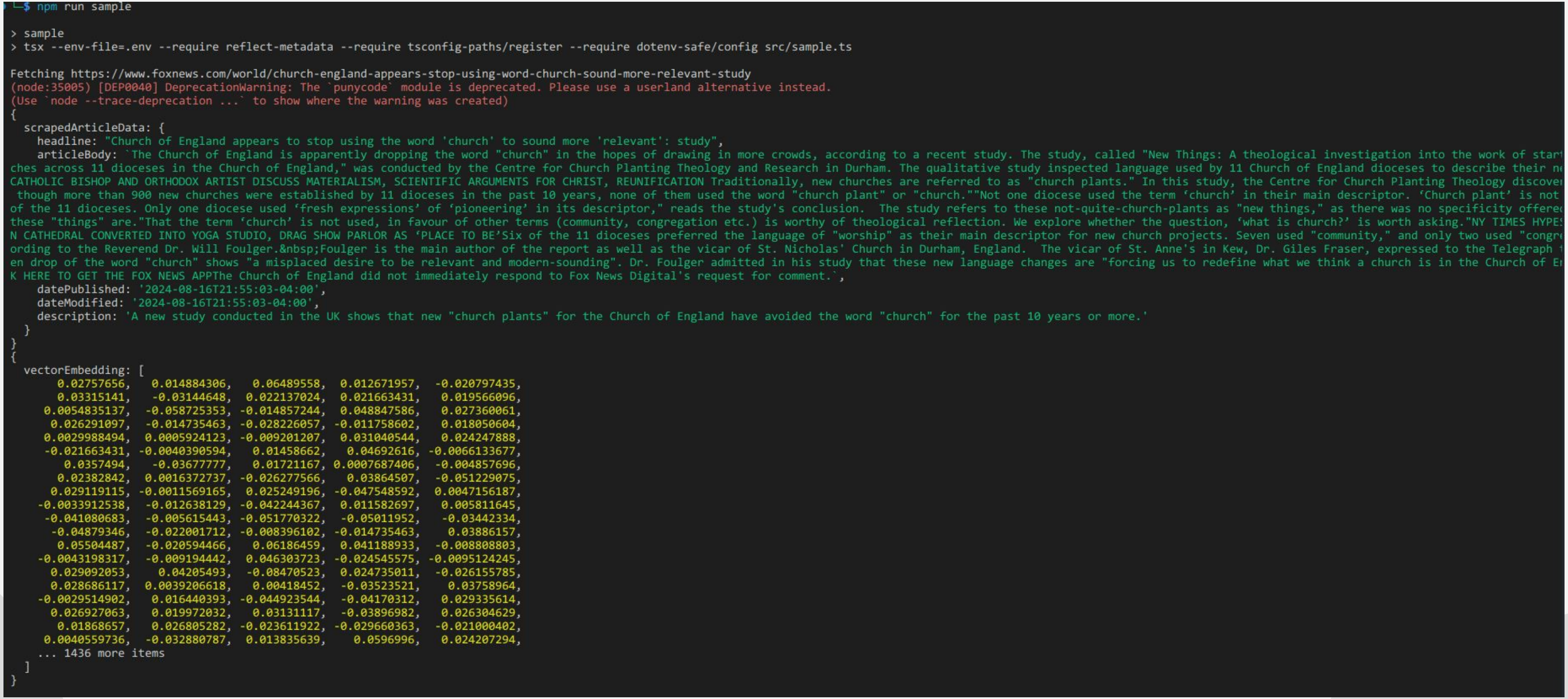
- **Idea**: FoxGPT, which is ChatGPT but equipped with real-time information updates from Fox News
- **Implementation**: We scrape Fox News articles, vectorise and store them in a **Pinecone** database. This is done daily to ensure that that our GPT has access to the latest articles and news updates. This time, we'll use OpenAI's **text-embedding-3-small** API to vectorise the news articles.
- **Repository**: https://github.com/elliotathelsinki/foxgpt-backend

Scrape, vectorise, and upsert



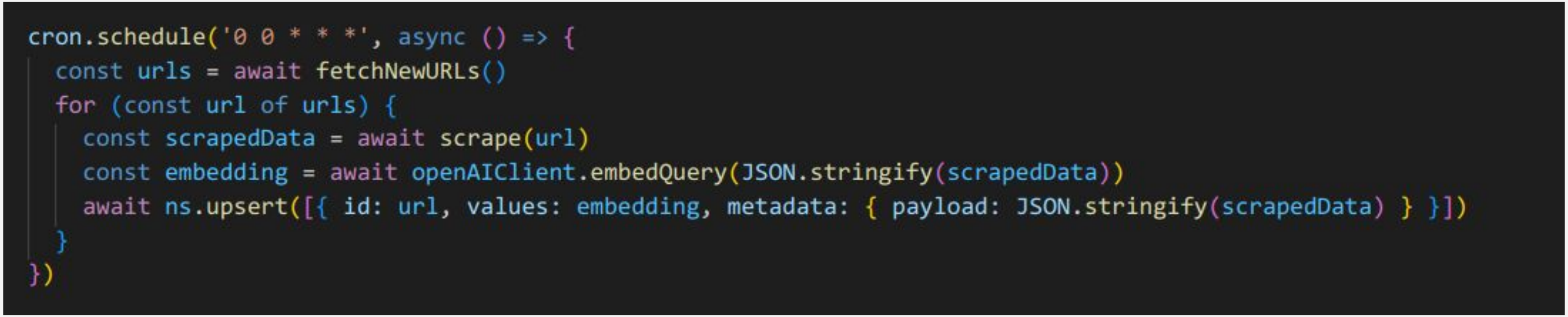
```
import { OpenAIEmbeddings } from '@langchain/openai'
import { Pinecone } from '@pinecone-database/pinecone'
import * as cheerio from 'cheerio'

const url = 'https://www.foxnews.com/world/church-england-appears-stop-using-word-church-sound-more-relevant-study'

const scrapedArticleData = await scrape(url)

console.log({ scrapedArticleData })

const openAIClient = new OpenAIEmbeddings({
  apiKey: process.env.OPENAI_API_KEY,
  model: 'text-embedding-3-small',
  dimensions: 1536
})

const vectorEmbedding = await openAIClient.embedQuery(JSON.stringify(scrapedArticleData))

console.log({ vectorEmbedding })

const pc = new Pinecone({
  apiKey: process.env.PINECONE_API_KEY
})

const ns = pc.index(process.env.PINECONE_INDEX).namespace(process.env.PINECONE_NAMESPACE)

await ns.upsert([{ id: url, values: vectorEmbedding, metadata: { payload: JSON.stringify(scrapedArticleData )}}])
```

Scrape, vectorise, and upsert



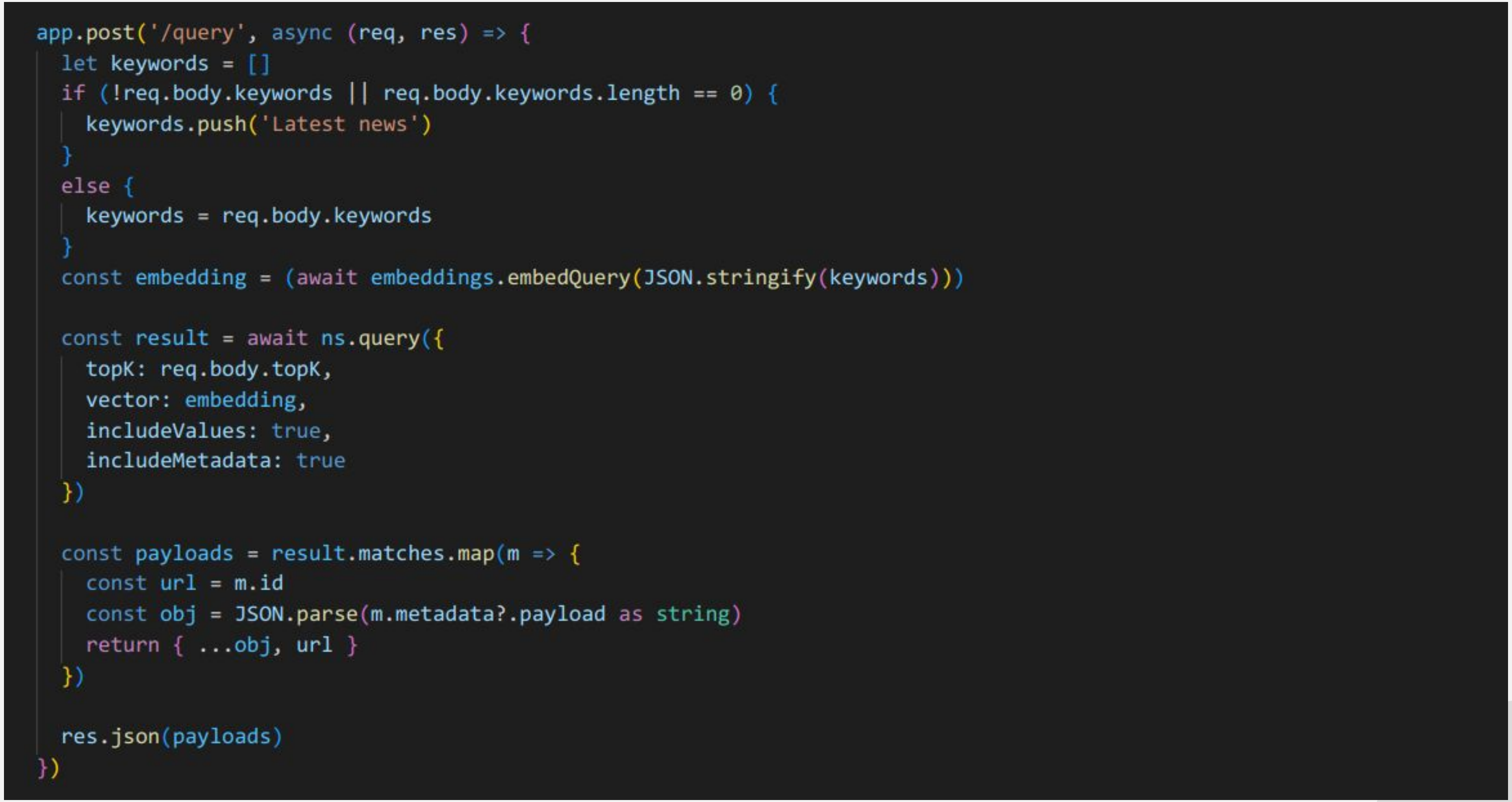
```
└─$ npm run sample

> sample
> tsx --env-file=.env --require reflect-metadata --require tsconfig-paths/register --require dotenv-safe/config src/sample.ts

Fetching https://www.foxnews.com/world/church-england-appears-stop-using-word-church-sound-more-relevant-study
(node:35005) [DEP0040] DeprecationWarning: The `punycode` module is deprecated. Please use a userland alternative instead.
(Use `node --trace-deprecation ...` to show where the warning was created)
{
  scrapedArticleData: {
    headline: "Church of England appears to stop using the word 'church' to sound more 'relevant': study",
    articleBody: `The Church of England is apparently dropping the word "church" in the hopes of drawing in more crowds, according to a recent study. The study, called "New Things: A theological investigation into the work of star
ches across 11 dioceses in the Church of England," was conducted by the Centre for Church Planting Theology and Research in Durham. The qualitative study inspected language used by 11 Church of England dioceses to describe their n
CATHOLIC BISHOP AND ORTHODOX ARTIST DISCUSS MATERIALISM, SCIENTIFIC ARGUMENTS FOR CHRIST, REUNIFICATION Traditionally, new churches are referred to as "church plants." In this study, the Centre for Church Planting Theology discove
 though more than 900 new churches were established by 11 dioceses in the past 10 years, none of them used the word "church plant" or "church.""Not one diocese used the term ‘church’ in their main descriptor. ‘Church plant’ is not
of the 11 dioceses. Only one diocese used ‘fresh expressions’ of ‘pioneering’ in its descriptor," reads the study's conclusion.  The study refers to these not-quite-church-plants as "new things," as there was no specificity offere
these "things" are."That the term ‘church’ is not used, in favour of other terms (community, congregation etc.) is worthy of theological reflection. We explore whether the question, ‘what is church?’ is worth asking."NY TIMES HYPE
N CATHEDRAL CONVERTED INTO YOGA STUDIO, DRAG SHOW PARLOR AS ‘PLACE TO BE’Six of the 11 dioceses preferred the language of "worship" as their main descriptor for new church projects. Seven used "community," and only two used "congre
ording to the Reverend Dr. Will Foulger.&nbsp;Foulger is the main author of the report as well as the vicar of St. Nicholas' Church in Durham, England.  The vicar of St. Anne's in Kew, Dr. Giles Fraser, expressed to the Telegraph
en drop of the word "church" shows "a misplaced desire to be relevant and modern-sounding". Dr. Foulger admitted in his study that these new language changes are "forcing us to redefine what we think a church is in the Church of E
K HERE TO GET THE FOX NEWS APPThe Church of England did not immediately respond to Fox News Digital's request for comment.`,
    datePublished: '2024-08-16T21:55:03-04:00',
    dateModified: '2024-08-16T21:55:03-04:00',
    description: 'A new study conducted in the UK shows that new "church plants" for the Church of England have avoided the word "church" for the past 10 years or more.'
  }
}
{
  vectorEmbedding: [
       0.02757656,   0.014884306,    0.06489558,   0.012671957,  -0.020797435,
       0.03315141,   -0.03144648,   0.022137024,   0.021663431,   0.019566096,
     0.0054835137,  -0.058725353,  -0.014857244,   0.048847586,   0.027360061,
      0.026291097,  -0.014735463,  -0.028226057,  -0.011758602,   0.018050604,
     0.0029988494,  0.0005924123,  -0.009201207,   0.031040544,   0.024247888,
     -0.021663431, -0.0040390594,    0.01458662,    0.04692616, -0.0066133677,
        0.0357494,   -0.03677777,    0.01721167,  0.0007687406,  -0.004857696,
       0.02382842,  0.0016372737,  -0.026277566,    0.03864507,  -0.051229075,
      0.029119115, -0.0011569165,   0.025249196,  -0.047548592,  0.0047156187,
    -0.0033912538,  -0.012638129,  -0.042244367,   0.011582697,   0.005811645,
     -0.041080683,  -0.005615443,  -0.051770322,   -0.05011952,   -0.03442334,
      -0.04879346,  -0.022001712,  -0.008396102,  -0.014735463,    0.03886157,
       0.05504487,  -0.020594466,    0.06186459,   0.041188933,  -0.008808803,
    -0.0043198317,  -0.009194442,   0.046303723,  -0.024545575, -0.0095124245,
      0.029092053,    0.04205493,   -0.08470523,   0.024735011,  -0.026155785,
      0.028686117,  0.0039206618,    0.00418452,   -0.03523521,    0.03758964,
    -0.0029514902,   0.016440393,  -0.044923544,   -0.04170312,   0.029335614,
      0.026927063,   0.019972032,    0.03131117,   -0.03896982,   0.026304629,
       0.01868657,   0.026805282,  -0.023611922,  -0.029660363,  -0.021000402,
     0.0040559736,  -0.032880787,   0.013835639,     0.0596996,   0.024207294,
    ... 1436 more items
  ]
}
```

Scrape, vectorise, and upsert



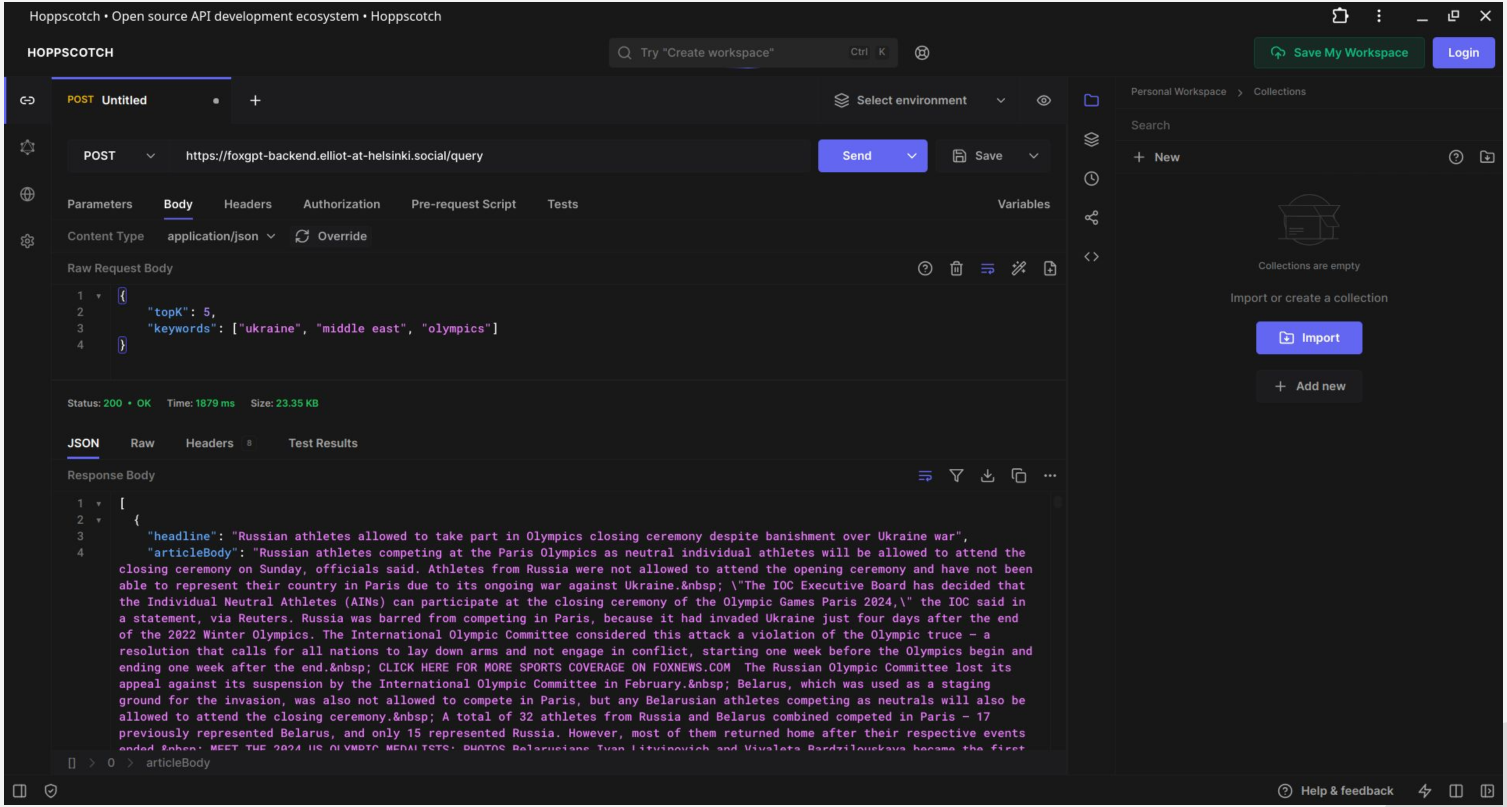
```
cron.schedule('0 0 * * *', async () => {
  const urls = await fetchNewURLs()
  for (const url of urls) {
    const scrapedData = await scrape(url)
    const embedding = await openAIClient.embedQuery(JSON.stringify(scrapedData))
    await ns.upsert([{ id: url, values: embedding, metadata: { payload: JSON.stringify(scrapedData) } }])
  }
})
```

Setting up a REST endpoint



```
app.post('/query', async (req, res) => {
  let keywords = []
  if (!req.body.keywords || req.body.keywords.length == 0) {
    keywords.push('Latest news')
  }
  else {
    keywords = req.body.keywords
  }
  const embedding = (await embeddings.embedQuery(JSON.stringify(keywords)))

  const result = await ns.query({
    topK: req.body.topK,
    vector: embedding,
    includeValues: true,
    includeMetadata: true
  })

  const payloads = result.matches.map(m => {
    const url = m.id
    const obj = JSON.parse(m.metadata?.payload as string)
    return { ...obj, url }
  })

  res.json(payloads)
})
```

Setting up a REST endpoint



POST https://foxgpt-backend.elliot-at-helsinki.social/query

Content Type: application/json

Override

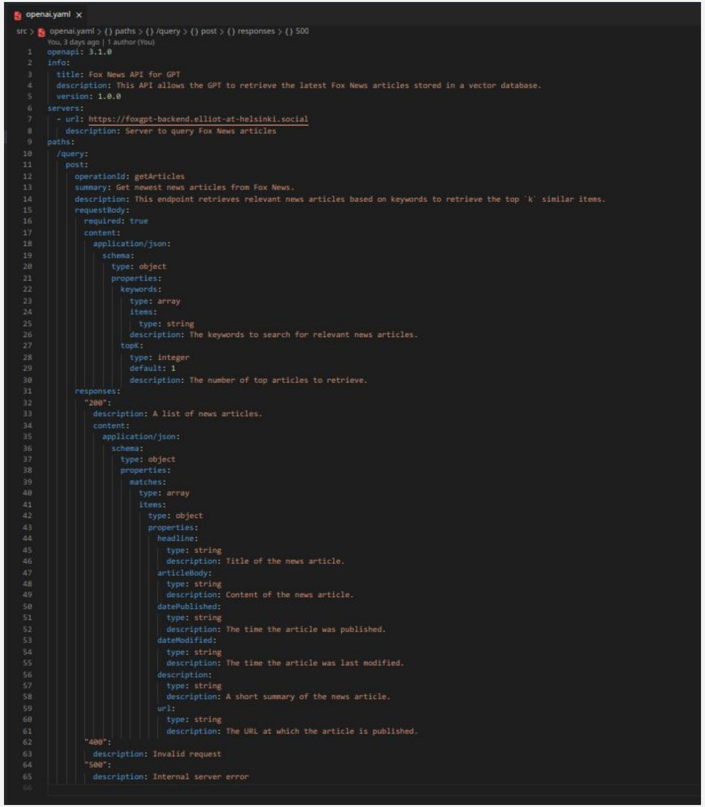
```
{
    "topK": 5,
    "keywords": ["ukraine", "middle east", "olympics"]
}
```

Status: 200 • OK Time: 1879 ms Size: 23.35 KB

```
[
  {
    "headline": "Russian athletes allowed to take part in Olympics closing ceremony despite banishment over Ukraine war",
    "articleBody": "Russian athletes competing at the Paris Olympics as neutral individual athletes will be allowed to attend the closing ceremony on Sunday, officials said. Athletes from Russia were not allowed to attend the opening ceremony and have not been able to represent their country in Paris due to its ongoing war against Ukraine.&nbsp; \"The IOC Executive Board has decided that the Individual Neutral Athletes (AINs) can participate at the closing ceremony of the Olympic Games Paris 2024,\" the IOC said in a statement, via Reuters. Russia was barred from competing in Paris, because it had invaded Ukraine just four days after the end of the 2022 Winter Olympics. The International Olympic Committee considered this attack a violation of the Olympic truce – a resolution that calls for all nations to lay down arms and not engage in conflict, starting one week before the Olympics begin and ending one week after the end.&nbsp; CLICK HERE FOR MORE SPORTS COVERAGE ON FOXNEWS.COM  The Russian Olympic Committee lost its appeal against its suspension by the International Olympic Committee in February.&nbsp; Belarus, which was used as a staging ground for the invasion, was also not allowed to compete in Paris, but any Belarusian athletes competing as neutrals will also be allowed to attend the closing ceremony.&nbsp; A total of 32 athletes from Russia and Belarus combined competed in Paris – 17 previously represented Belarus, and only 15 represented Russia. However, most of them returned home after their respective events ended.&nbsp; MEET THE 2024 US OLYMPIC MEDALISTS: PHOTOS Belarusians Ivan Litvinovich and Viyaleta Bardzilouskaya became the first
```

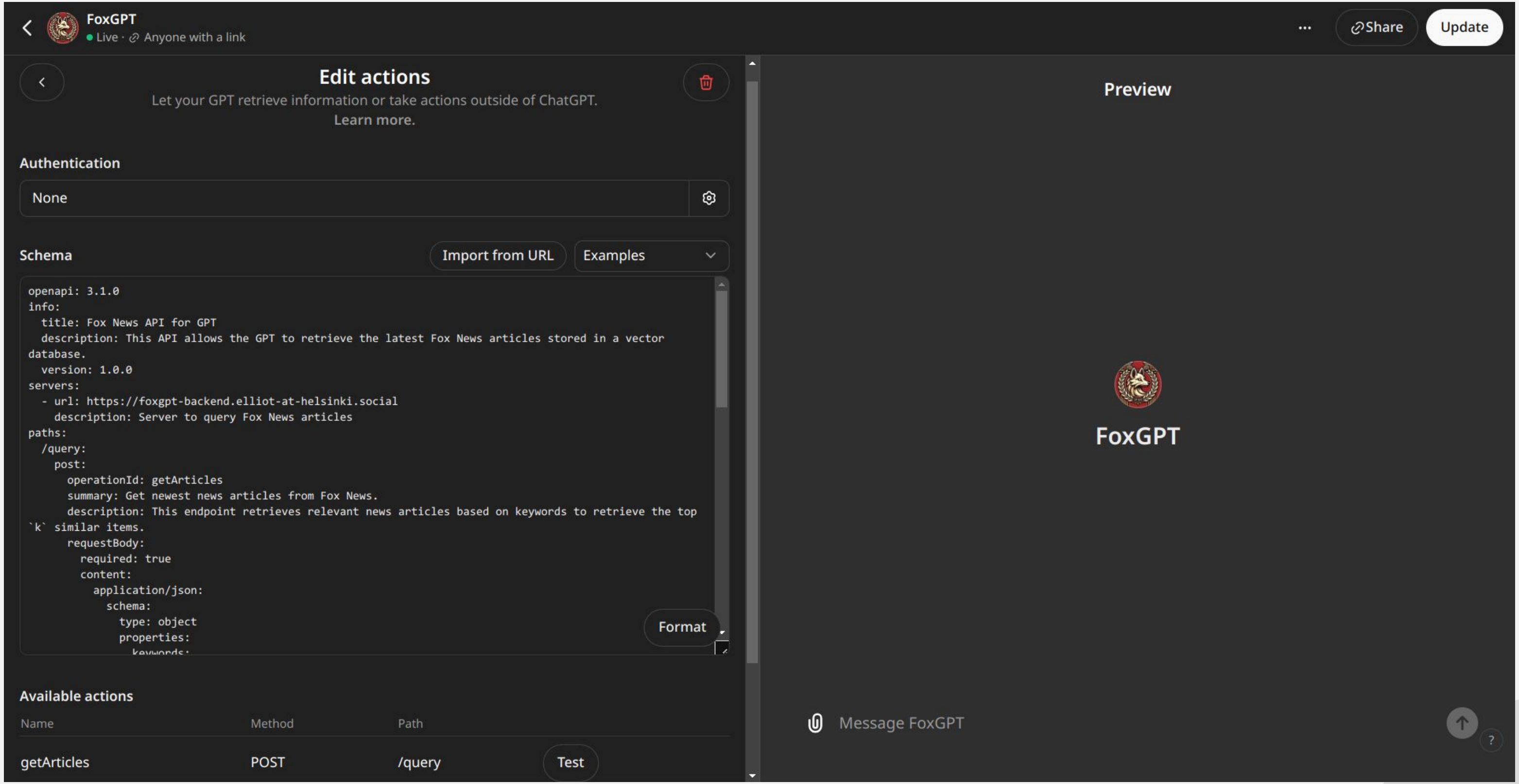
[] > 0 > articleBody

Defining a GPT Action


openai.yaml
src > openai.yaml > {} paths > {} /query > {} post > {} responses > {} 500
You, 3 days ago | 1 author (You)

```yaml
openapi: 3.1.0
info:
  title: Fox News API for GPT
  description: This API allows the GPT to retrieve the latest Fox News articles stored in a vector database.
  version: 1.0.0
servers:
  - url: https://foxgpt-backend.elliot-at-helsinki.social
    description: Server to query Fox News articles
paths:
  /query:
    post:
      operationId: getArticles
      summary: Get newest news articles from Fox News.
      description: This endpoint retrieves relevant news articles based on keywords to retrieve the top `k` similar items.
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              properties:
                keywords:
                  type: array
                  items:
                    type: string
                  description: The keywords to search for relevant news articles.
                topK:
                  type: integer
                  default: 1
                  description: The number of top articles to retrieve.
      responses:
        "200":
          description: A list of news articles.
          content:
            application/json:
              schema:
                type: object
                properties:
                  matches:
                    type: array
                    items:
                      type: object
                      properties:
                        headline:
                          type: string
                          description: Title of the news article.
                        articleBody:
                          type: string
                          description: Content of the news article.
                        datePublished:
                          type: string
                          description: The time the article was published.
                        dateModified:
                          type: string
                          description: The time the article was last modified.
                        description:
                          type: string
                          description: A short summary of the news article.
                        url:
                          type: string
                          description: The URL at which the article is published.
        "400":
          description: Invalid request
        "500":
          description: Internal server error
```

Defining a GPT Action



Edit actions

Let your GPT retrieve information or take actions outside of ChatGPT. Learn more.

Authentication

None

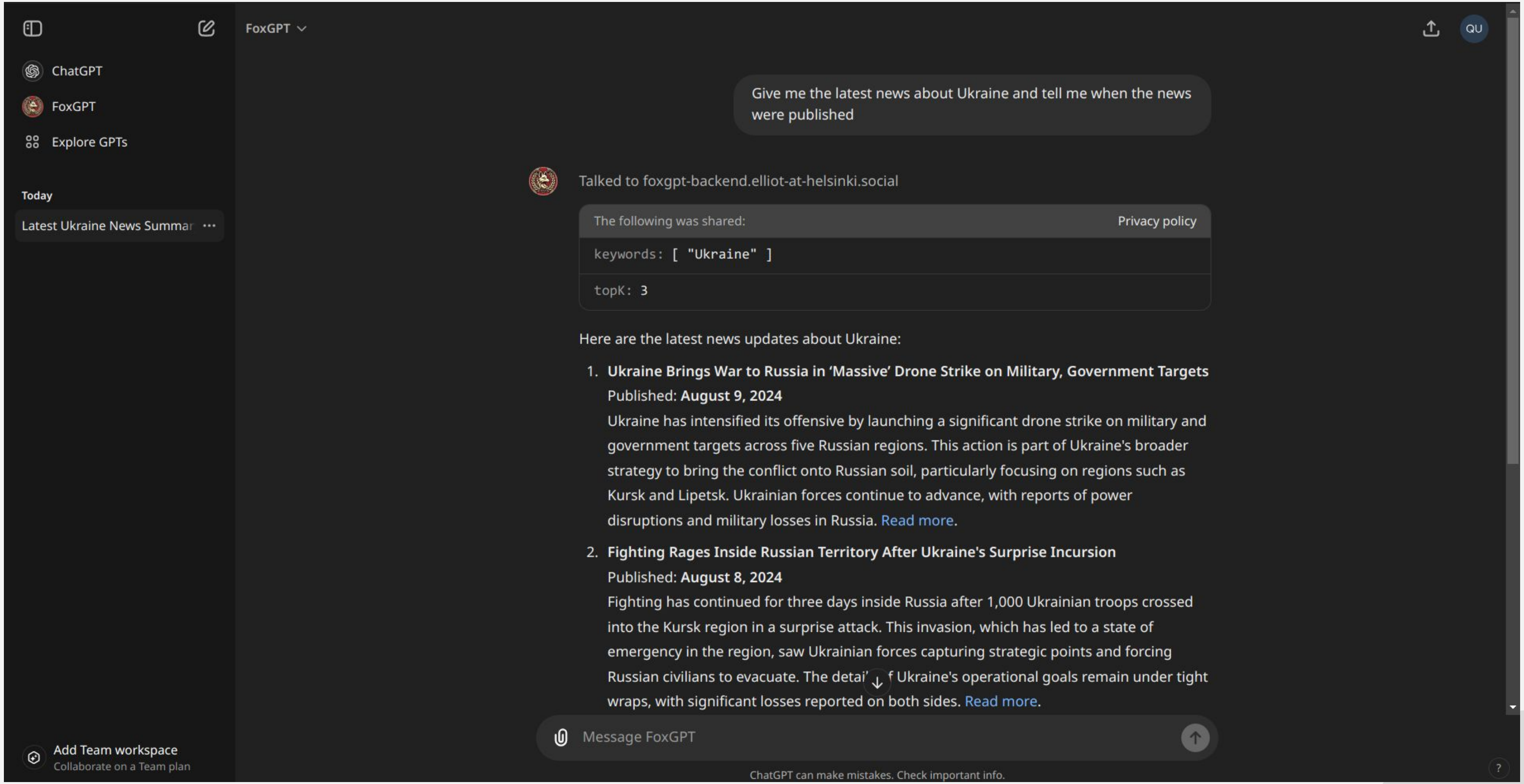
Schema

```
openapi: 3.1.0
info:
  title: Fox News API for GPT
  description: This API allows the GPT to retrieve the latest Fox News articles stored in a vector
database.
  version: 1.0.0
servers:
  - url: https://foxgpt-backend.elliot-at-helsinki.social
    description: Server to query Fox News articles
paths:
  /query:
    post:
      operationId: getArticles
      summary: Get newest news articles from Fox News.
      description: This endpoint retrieves relevant news articles based on keywords to retrieve the top
`k` similar items.
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              properties:
                keywords:
```

Available actions

Name	Method	Path	
getArticles	POST	/query	Test

Voilà!



Give me the latest news about Ukraine and tell me when the news were published

Talked to foxgpt-backend.elliot-at-helsinki.social

The following was shared: Privacy policy

```
keywords: [ "Ukraine" ]
topK: 3
```

Here are the latest news updates about Ukraine:

1. **Ukraine Brings War to Russia in 'Massive' Drone Strike on Military, Government Targets**
   Published: **August 9, 2024**
   Ukraine has intensified its offensive by launching a significant drone strike on military and government targets across five Russian regions. This action is part of Ukraine's broader strategy to bring the conflict onto Russian soil, particularly focusing on regions such as Kursk and Lipetsk. Ukrainian forces continue to advance, with reports of power disruptions and military losses in Russia. Read more.
2. **Fighting Rages Inside Russian Territory After Ukraine's Surprise Incursion**
   Published: **August 8, 2024**
   Fighting has continued for three days inside Russia after 1,000 Ukrainian troops crossed into the Kursk region in a surprise attack. This invasion, which has led to a state of emergency in the region, saw Ukrainian forces capturing strategic points and forcing Russian civilians to evacuate. The detai f Ukraine's operational goals remain under tight wraps, with significant losses reported on both sides. Read more.

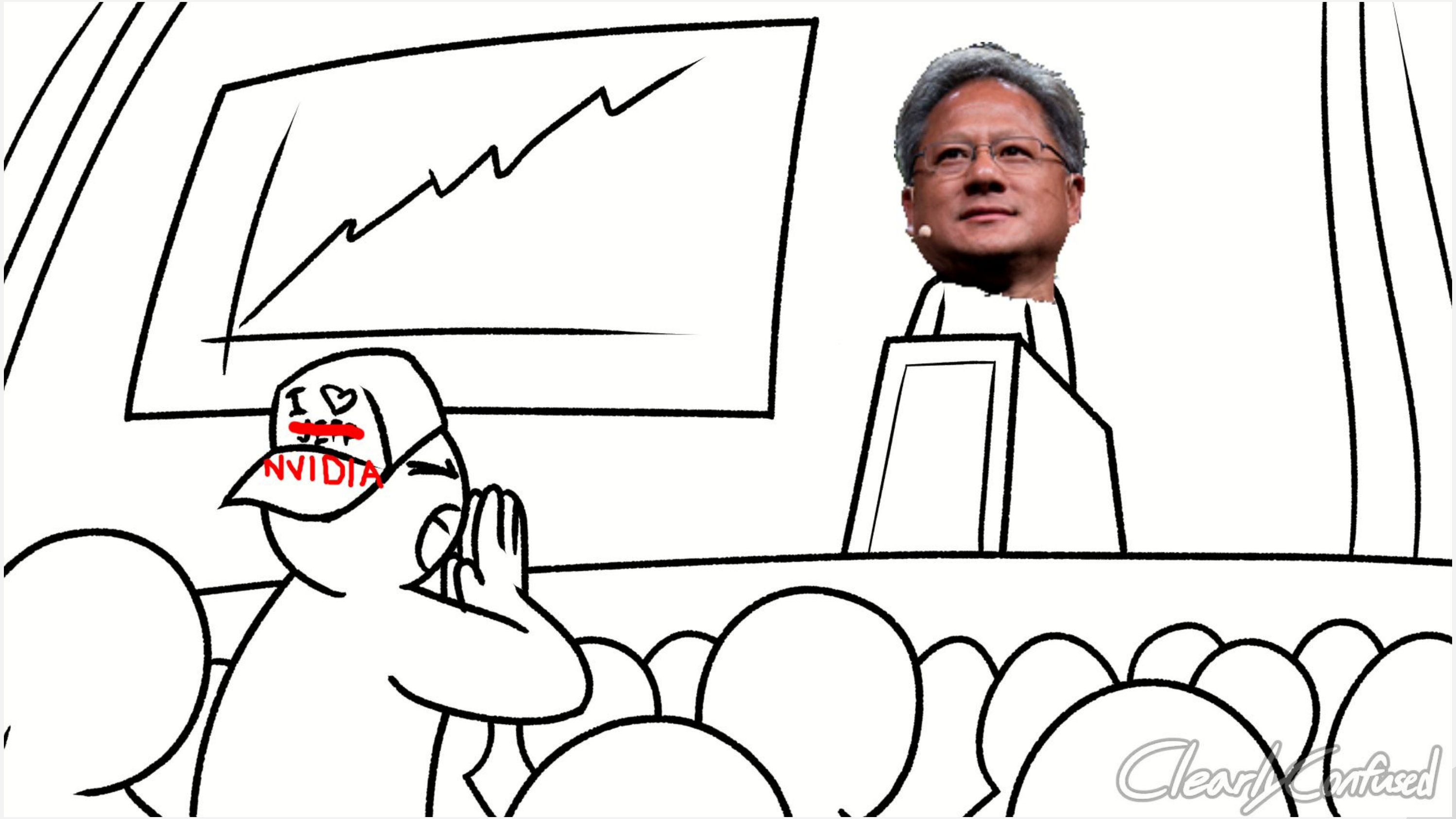
Message FoxGPT

ChatGPT can make mistakes. Check important info.

Voilà!

=> Visit **https://chatgpt.com/g/g-5pzMtoqjN-foxgpt** to test it yourself!

AI to the moon!

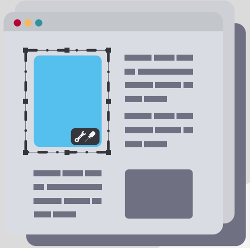


References

[1] Zhi Jing, Yongye Su, Yikun Han, Bo Yuan, Haiyun Xu, Chunjiang Liu, Kehai Chen, Min Zhang. 2024. When Large Language Models Meet Vector Databases: A Survey. Retrieved Aug 18, 2024 from https://arxiv.org/abs/2402.01763

[2] Toni Taipalus. 2024. Vector database management systems: Fundamental concepts, use-cases, and current challenges. Cognitive Systems Research 85, C, Article 101216, June 2024, 8 pages. https://doi.org/10.1016/j.cogsys.2024.101216

[3] James Jie Pan, Jianguo Wang, Guoliang Li. 2024. Vector Database Management Techniques and Systems. Retrieved Aug 18, 2024 from https://dl.acm.org/doi/pdf/10.1145/3626246.3654691



Thanks!

Do you have any questions?

quyanh.nguyen@helsinki.fi

CREDITS: This presentation template was created by **Slidesgo**, and includes icons by **Flaticon**, and infographics & images by **Freepik**

Please keep this slide for attribution