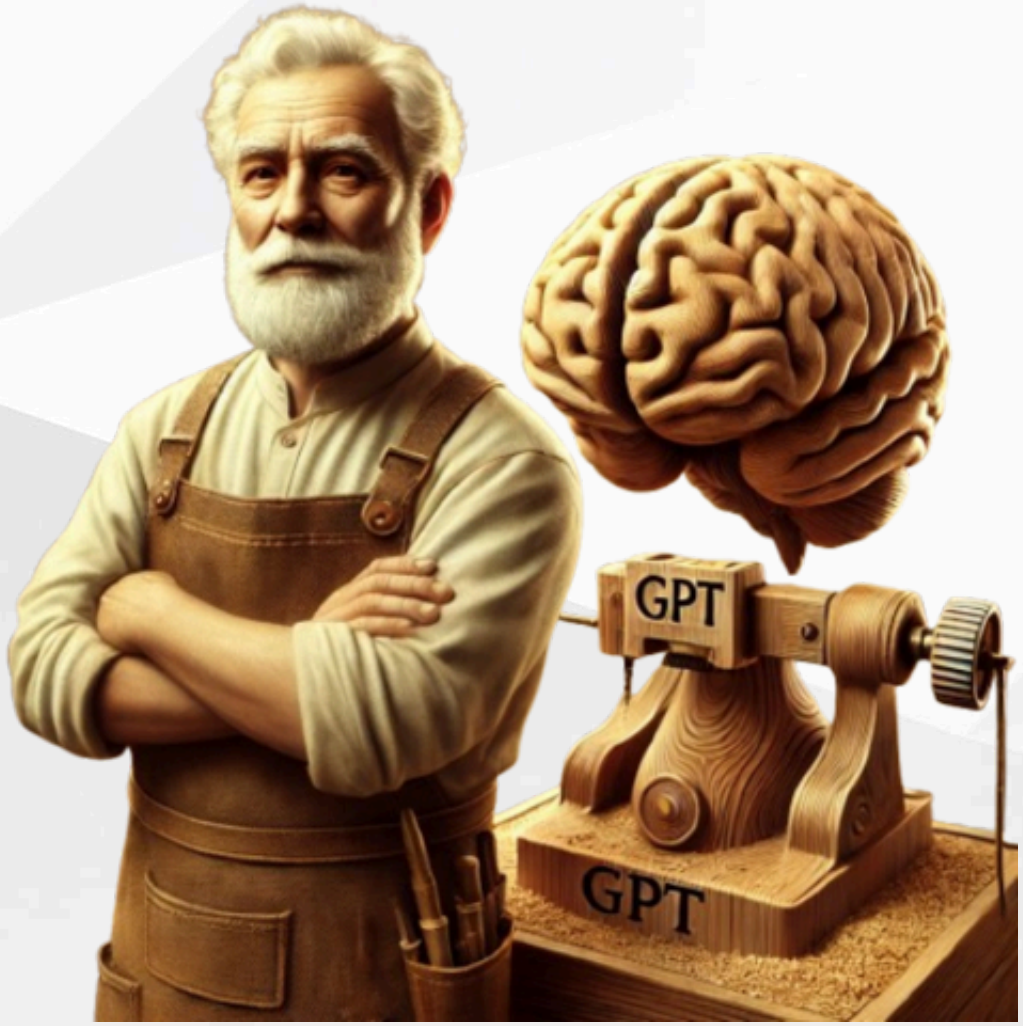
Developing Open LLM applications with

Apache OpenServerless

Lesson 6

Vector Database



Vector Database

- Vector Database
- Embedding and Search
- Importing PDF

Vector Database

Milvus: concepts

- Milvus is a No-SQL database optimized for **vector searches**
  - finding **similarities** in a dataset looking at their **numeric representation**
  - text is transformed in a numeric representatino using an **embedding model** then stored
- Multiple **databases**, each database has multiple **collections**
- Each collection has a **schema** and **indexes**

Access Milvus

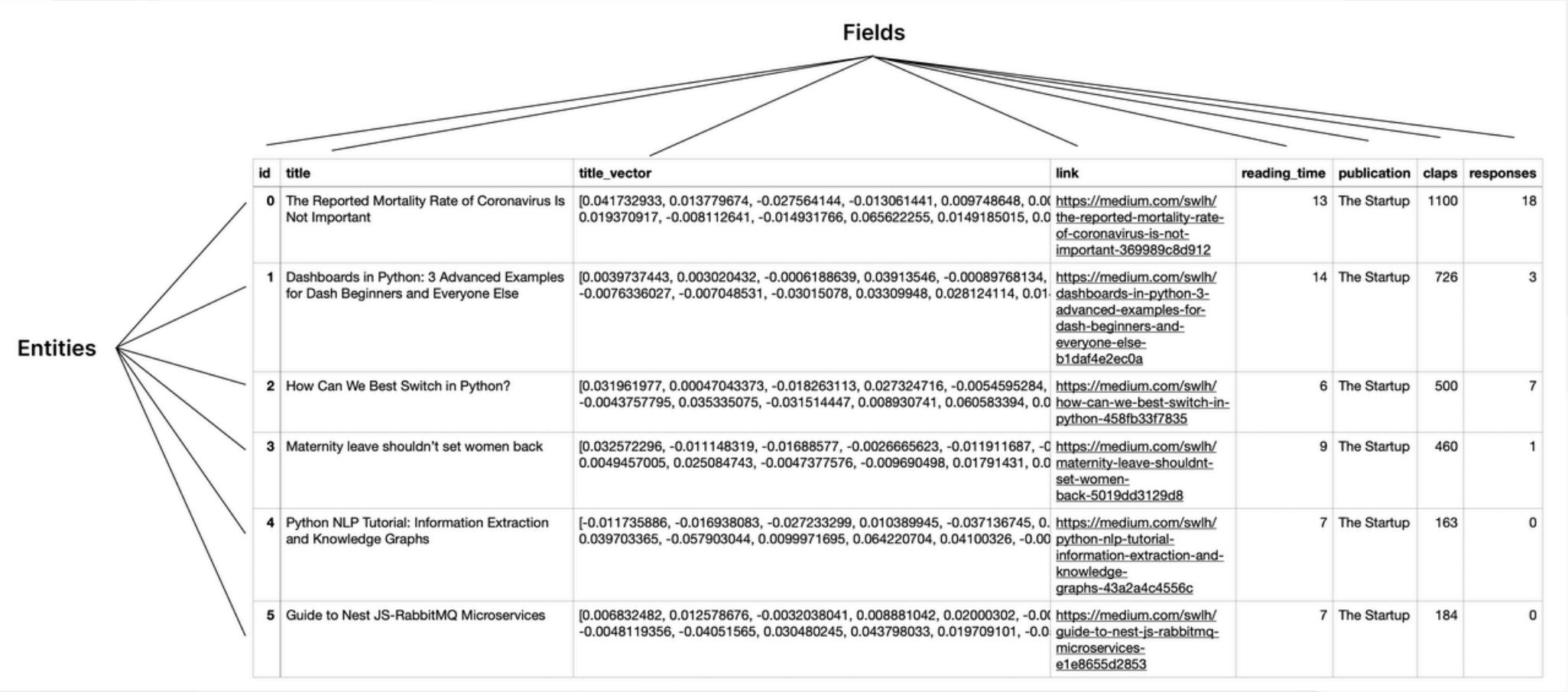
```
import os
from pymilvus import MilvusClient

uri = f"http://{os.getenv("MILVUS_HOST")}"
token = os.getenv("MILVUS_TOKEN")
db_name =  os.getenv("MILVUS_DB_NAME")
client = MilvusClient(uri=uri, token=token, db_name=db_name)
```

- Using Milvus

```
client.list_collections()
client.drop_collection("test")
```

Collections



id	title	title_vector	link	reading_time	publication	claps	responses
0	The Reported Mortality Rate of Coronavirus Is Not Important	[0.041732933, 0.013779674, -0.027564144, -0.013061441, 0.009748648, 0.0(0.019370917, -0.008112641, -0.014931766, 0.065622255, 0.0149185015, 0.0	https://medium.com/swlh/the-reported-mortality-rate-of-coronavirus-is-not-important-369989c8d912	13	The Startup	1100	18
1	Dashboards in Python: 3 Advanced Examples for Dash Beginners and Everyone Else	[0.0039737443, 0.003020432, -0.0006188639, 0.03913546, -0.00089768134, -0.0076336027, -0.007048531, -0.03015078, 0.03309948, 0.028124114, 0.01	https://medium.com/swlh/dashboards-in-python-3-advanced-examples-for-dash-beginners-and-everyone-else-b1daf4e2ec0a	14	The Startup	726	3
2	How Can We Best Switch in Python?	[0.031961977, 0.00047043373, -0.018263113, 0.027324716, -0.0054595284, -0.0043757795, 0.035335075, -0.031514447, 0.008930741, 0.060583394, 0.0	https://medium.com/swlh/how-can-we-best-switch-in-python-458fb33f7835	6	The Startup	500	7
3	Maternity leave shouldn't set women back	[0.032572296, -0.011148319, -0.01688577, -0.0026665623, -0.011911687, -(0.0049457005, 0.025084743, -0.0047377576, -0.009690498, 0.01791431, 0.0	https://medium.com/swlh/maternity-leave-shouldnt-set-women-back-5019dd3129d8	9	The Startup	460	1
4	Python NLP Tutorial: Information Extraction and Knowledge Graphs	[-0.011735886, -0.016938083, -0.027233299, 0.010389945, -0.037136745, 0. 0.039703365, -0.057903044, 0.0099971695, 0.064220704, 0.04100326, -0.00	https://medium.com/swlh/python-nlp-tutorial-information-extraction-and-knowledge-graphs-43a2a4c4556c	7	The Startup	163	0
5	Guide to Nest JS-RabbitMQ Microservices	[0.006832482, 0.012578676, -0.0032038041, 0.008881042, 0.02000302, -0.0(-0.0048119356, -0.04051565, 0.030480245, 0.043798033, 0.019709101, -0.0	https://medium.com/swlh/guide-to-nest-js-rabbitmq-microservices-e1e8655d2853	7	The Startup	184	0

Create Schema

- Parameters

```
from pymilvus import DataType
COLLECTION = "test"
DIMENSION=1024
```

- Define Schema

```
schema = client.create_schema()
schema.add_field(field_name="id", datatype=DataType.INT64, is_primary=True, auto_id=True)
schema.add_field(field_name="text", datatype=DataType.VARCHAR, max_length=DIMENSION)
schema.add_field(field_name="embeddings", datatype=DataType.FLOAT_VECTOR, dim=DIMENSION)
```

Create Index and Collection

- Define Index

```
index_params = client.prepare_index_params()
index_params.add_index("embeddings", index_type="AUTOINDEX", metric_type="IP")
```

- Create Collection with Index and Schema

```
client.create_collection(
     collection_name=COLLECTION,
     schema=schema, index_params=index_params)
```

Insert

```
text = "Hello World"
vec = [float(i) for i in range(0,DIMENSION)]  # TO BE REPLACED with embedding
client.insert(COLLECTION, {"text":text, "embeddings": vec})
```

Retrieve

```
qit = client.query_iterator(collection_name=COLLECTION, batchSize=2, output_fields=["text"])
res = qit.next()
print(res[0].get("text"))
```

```
Hello World
```

Embedding

Embedding

- Use an embedding model

```
import sys, requests as req
MODEL="mxbai-embed-large:latest"
DIMENSION=1024
```

- Invoke the embeeding API

```
inp = "Hello World"
url = f"https://{os.getenv("AUTH")}@{os.getenv("OLLAMA_HOST")}/api/embeddings"
msg = { "model": MODEL, "prompt": inp, "stream": False }
res = req.post(url, json=msg).json()
out = res.get('embedding', [])
```

VectorDB with embedding

- Using the `VectorDB` class

```
!code packages/vdb/load/vdb.py
import sys ; sys.path.append("packages/vdb/load")
import vdb
db = vdb.VectorDB({})
```

- Insert text with embedding

```
db.insert("Hello World")
db.insert("This is a test")
db.insert("This is another test")
db.insert("Testing")
```

Vector Search

- Prepare

```
text = "Test"
vec = db.embed(text)
```

- Execute the actual search

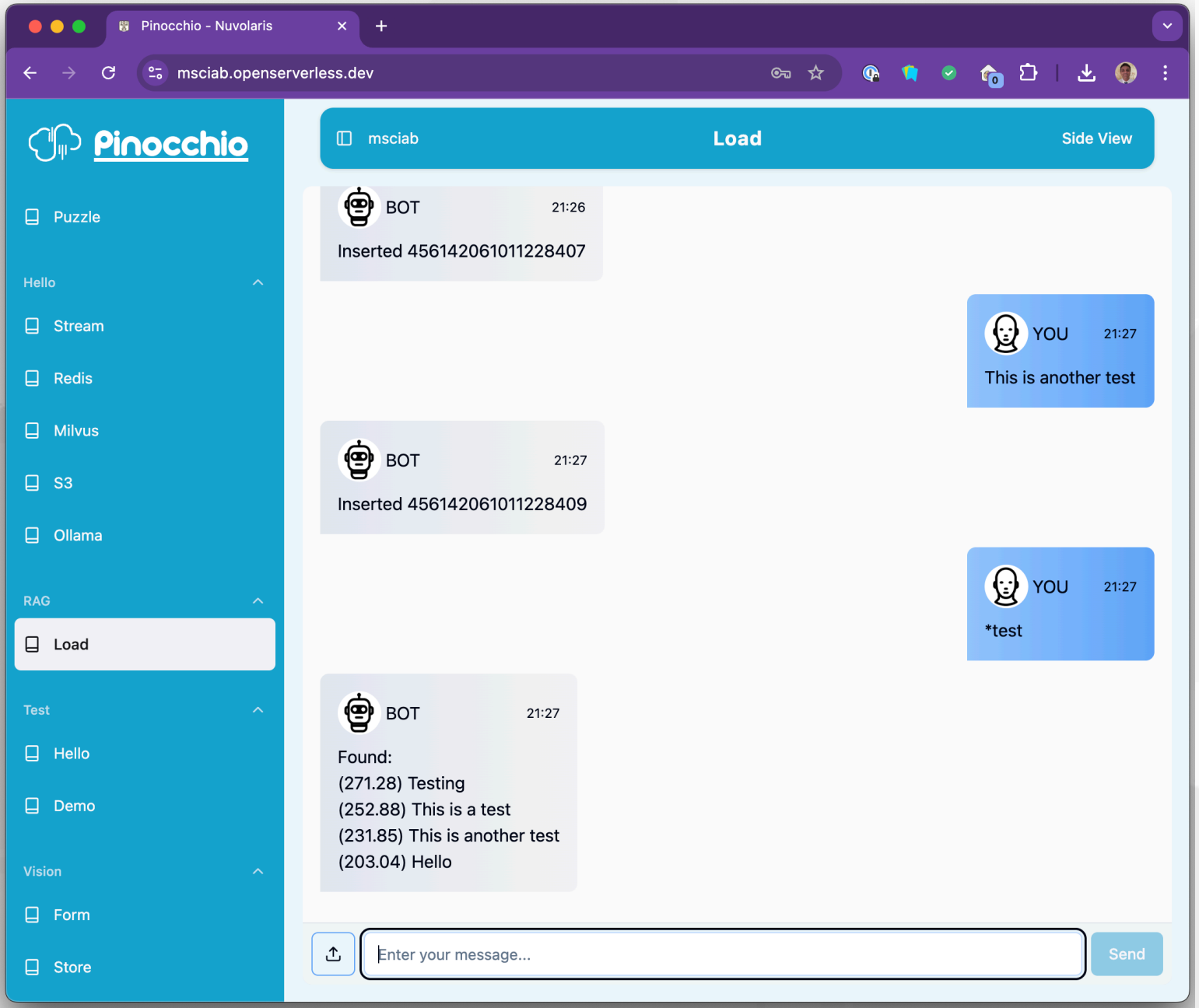
```
cur = client.search(collection_name=COLLECTION, # collection
    search_params={"metric_type": "IP"},        # how to measure distance
    anns_field="embeddings",                    # where to search
    data=[vec],                                 # what to search
    output_fields=["text"]                      # field to return
)
```

Vector Search results

```
for item in cur[0]:
     dist = item.get('distance', 0)
     text = item.get("entity", {}).get("text", "")
     print(dist, text)
```

```
271.28466796875 Testing
252.88241577148438 This is a test
231.84872436523438 This is another test
181.67494201660156 Hello World
```

Note output is ordered by **distance**



`vdb/load` action

Using command line

```
!ops invoke vdb/load input="openserverless cli import"
!ops invoke vdb/load input="*openserverless" | jq .body.output
```

Looking at code

```
!code packages/vdb/load/vdb.py
!code packages/vdb/load/load.py
```

PDF Import

Extract content from PDF files

Pages from a PDF

```
import pymupdf
doc = pymupdf.open("lessons/bitcoin.pdf")  # load a file
text = doc[0].get_text()                   # extract text from page
```

Split in sentences

```
import nltk.data
nltk.download('punkt')                     # load tokenizer model
from nltk.tokenize import sent_tokenize
sentences = sent_tokenize(text)            # extract sentences from text
enum = enumerate(sentences, 1)             # enumerate
sent = next(enum)                          # extract one sentence
```

ops ai loader

```
!ops ai loader
```

```
ai loader [--action=<action>] <file>...
```

```
!ops ai loader lessons/bitcoin.pdf
```

```
>>> converting lessons/bitcoin.pdf
saved lessons/bitcoin.pdf.txt
```

Excercise: import from the web

- Add to the `vdb/load` the ability to import content from http pages providing an `https://` url

Hints:

- process `input` starting with `https://`
- use `requests` to read the url content
- use `BeautifulSoap (bs4)` to process the content and extract text
- Tokenize text with regular expressions: https://stackoverflow.com/questions/75253187/tokenizing-the-text-without-the-use-of-libraries