02/20/2025

API UTILISATION

1. DIGITAL COMMONWEALTH API

The Digital Commonwealth API provides access to metadata from the Boston Public Library's vast collection of digitized archives, including books, manuscripts, photographs, and historical documents.

A. Usage

- The API is queried to fetch metadata, such as document titles, descriptions, authors, and publication dates.
- The retrieved metadata is then embedded into a vector database for efficient retrieval.

B. Data Retrieved

- Item titles, descriptions, and subjects
- Collection information
- URLs for digital access

C. Example API call

```
import requests
url = "https://www.digitalcommonwealth.org/collections/metadata.json"
response = requests.get(url)
print(response.json())
```

2. PINECONE VECTOR DATABASE API

Pinecone is a vector database service used to store and efficiently search high-dimensional embeddings generated from the Digital Commonwealth metadata. This enables fast similarity searches for relevant resources.

A. Usage

- Metadata from the Digital Commonwealth API is transformed into vector embeddings.

- These embeddings are stored in a Pinecone index, enabling similarity-based searches.
- When a user queries the system, it retrieves the most relevant documents based on vector similarity.

B. Data Retrieved

- High-dimensional vector representations of metadata
- Similarity-ranked search results

C. Example API call

```python
from pinecone import Pinecone, ServerlessSpec
pc = Pinecone(api_key="your_api_key")
index_name = "bpl-index"
index = pc.Index(index_name)
query_result = index.query(vector=[0.1, 0.2, 0.3], top_k=3, include_metadata=True)
print(query_result)
```

3. OPENAI API

The OpenAI API is used to process natural language queries and generate contextual responses based on the retrieved documents.

A. Usage

- When a user submits a query, the system retrieves relevant metadata from Pinecone.
- The metadata is passed as context to OpenAI's language model to generate a coherent and informative response.

B. Data Retrieved

- AI-generated responses based on retrieved metadata
- Contextual insights extracted from historical documents

C. Example API call

```python
import openai
openai.api_key = "your_api_key"
response = openai.ChatCompletion.create(
    model="gpt-4",
    messages=[{"role": "user", "content": "Tell me about Boston Public Library's history"}]
)
print(response["choices"][0]["message"]["content"])
```