GITHUB ISSUES

ELASTICSEARCH - OPEN AI

In [23]:
```
%pip install --upgrade pip
```

```
Requirement already satisfied: pip in /Library/Frameworks/Python.framework/V
ersions/3.11/lib/python3.11/site-packages (24.3.1)
Note: you may need to restart the kernel to use updated packages.
```

In [24]:
```
# Install the required packages
%pip install openai
%pip install --upgrade typing-extensions
```

```
Requirement already satisfied: openai in /Library/Frameworks/Python.framewor
k/Versions/3.11/lib/python3.11/site-packages (1.55.0)
Requirement already satisfied: anyio<5,>=3.5.0 in /Library/Frameworks/Pytho
n.framework/Versions/3.11/lib/python3.11/site-packages (from openai) (4.6.2.
post1)
Requirement already satisfied: distro<2,>=1.7.0 in /Library/Frameworks/Pytho
n.framework/Versions/3.11/lib/python3.11/site-packages (from openai) (1.9.0)
Requirement already satisfied: httpx<1,>=0.23.0 in /Library/Frameworks/Pytho
n.framework/Versions/3.11/lib/python3.11/site-packages (from openai) (0.25.
0)
Requirement already satisfied: jiter<1,>=0.4.0 in /Library/Frameworks/Pytho
n.framework/Versions/3.11/lib/python3.11/site-packages (from openai) (0.7.1)
Requirement already satisfied: pydantic<3,>=1.9.0 in /Library/Frameworks/Pyt
hon.framework/Versions/3.11/lib/python3.11/site-packages (from openai) (2.1
0.0)
Requirement already satisfied: sniffio in /Library/Frameworks/Python.framewo
rk/Versions/3.11/lib/python3.11/site-packages (from openai) (1.3.1)
Requirement already satisfied: tqdm>4 in /Library/Frameworks/Python.framewor
k/Versions/3.11/lib/python3.11/site-packages (from openai) (4.67.0)
Requirement already satisfied: typing-extensions<5,>=4.11 in /Library/Framew
orks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from opena
i) (4.12.2)
Requirement already satisfied: idna>=2.8 in /Library/Frameworks/Python.frame
work/Versions/3.11/lib/python3.11/site-packages (from anyio<5,>=3.5.0->opena
i) (2.10)
Requirement already satisfied: certifi in /Library/Frameworks/Python.framewo
rk/Versions/3.11/lib/python3.11/site-packages (from httpx<1,>=0.23.0->opena
i) (2024.8.30)
Requirement already satisfied: httpcore<0.19.0,>=0.18.0 in /Library/Framewor
ks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from httpx<
1,>=0.23.0->openai) (0.18.0)
Requirement already satisfied: annotated-types>=0.6.0 in /Library/Framework
s/Python.framework/Versions/3.11/lib/python3.11/site-packages (from pydantic
<3,>=1.9.0->openai) (0.7.0)
Requirement already satisfied: pydantic-core==2.27.0 in /Library/Frameworks/
Python.framework/Versions/3.11/lib/python3.11/site-packages (from pydantic<
3,>=1.9.0->openai) (2.27.0)
Requirement already satisfied: h11<0.15,>=0.13 in /Library/Frameworks/Pytho
n.framework/Versions/3.11/lib/python3.11/site-packages (from httpcore<0.19.
0,>=0.18.0->httpx<1,>=0.23.0->openai) (0.14.0)
Note: you may need to restart the kernel to use updated packages.
Requirement already satisfied: typing-extensions in /Library/Frameworks/Pyth
on.framework/Versions/3.11/lib/python3.11/site-packages (4.12.2)
Note: you may need to restart the kernel to use updated packages.
```

In [ ]:

In [25]:
```
#Install elastic search
!pip install elasticsearch
```

```
Requirement already satisfied: elasticsearch in /Library/Frameworks/Python.f
ramework/Versions/3.11/lib/python3.11/site-packages (8.16.0)
Requirement already satisfied: elastic-transport<9,>=8.15.1 in /Library/Fram
eworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from ela
sticsearch) (8.15.1)
Requirement already satisfied: urllib3<3,>=1.26.2 in /Library/Frameworks/Pyt
hon.framework/Versions/3.11/lib/python3.11/site-packages (from elastic-trans
port<9,>=8.15.1->elasticsearch) (2.2.3)
Requirement already satisfied: certifi in /Library/Frameworks/Python.framewo
rk/Versions/3.11/lib/python3.11/site-packages (from elastic-transport<9,>=8.
15.1->elasticsearch) (2024.8.30)
```

In [ ]:

In [26]:
```
# Import the required packages
import requests
import datetime as dt
from datetime import datetime
from pprint import pprint
import pandas as pd
```

In [ ]:

In [27]:
```
# Declare the headers
headers = {
    "Accept": "application/vnd.github+json",
    "access_token": "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",
    "Git_Username":"PFA24SCM25S"
}
```

In [28]:
```
# Declare the owner and the repository
owners = ['langchain-ai','langchain-ai','microsoft','openai', 'elastic',  'm
repos = ['langchain','langgraph','autogen','openai-cookbook', 'elasticsearch
```

In [29]:
```
page = 1
per_page = 10
from_date = (dt.date.today() - dt.timedelta(days=60)).isoformat() #days=60 b

# from datetime import date
# from dateutil.relativedelta import relativedelta

# from_date = (date.today() - relativedelta(months=2)).isoformat()
# print(from_date)
```

In [ ]:

In [30]:
```
# Method that returns the base url
def fetch_url(owner, repo):
```

```
    return f"https://"+headers["Git_Username"]+":"+headers["access_token"]+f
```

In [ ]:

In [31]:
```
# Fetching the Issues from the GitHub repository
issues=[]
for owner in owners:
    for repo in repos:
        if (owner=='langchain-ai' and repo=='langchain') or (owner=='langcha
            flag = True
            url = fetch_url(owner, repo)
            while flag:
                response = requests.get(f"{url}/issues", headers=headers,par
                for obj in response.json():
                    if datetime.strptime(from_date, "%Y-%m-%d") <= datetime.
                        issueObject = {
                            "_type": "issue",
                            "_repo": repo,
                            "_issueNumber": str(obj['number']),
                            "_title": str(obj['title']),
                            "_createdAt": str(obj['created_at']),
                            "_closedAt": str(obj['closed_at']) if str(obj['c
                            # Few Issues might still be open, we add "2024-1
                            "_state": str(obj['state']),
                            "_body": str(obj['body'])[:5000]
                            # Here we are considering only the first 5000 ch
                            # there is a limit on the the text tokens that w
                            # Please refer https://platform.openai.com/docs/
                            # Please refer to https://github.com/openai/oper
                        }
                        issues.append(issueObject)
                    else:
                        flag = False
                        break

                if not response.ok or len(response.json()) == 0:
                    break

                page+=1
```

In [32]:
```
#Sample Issue
pprint(issues[0])
```

```
{'_body': '### Checked other resources\n'
          '\n'
          '- [X] I added a very descriptive title to this issue.\n'
          '- [X] I searched the LangChain documentation with the integrated
'
          'search.\n'
```

```
            '- [X] I used the GitHub search to find a similar question and '
            "didn't find it.\n"
            '- [X] I am sure that this is a bug in LangChain rather than my '
            'code.\n'
            '- [X] The bug is not resolved by updating to the latest stable '
            'version of LangChain (or the specific integration package).\n'
            '\n'
            '### Example Code\n'
            '\n'
            '```\r\n'
            '#----------------\r\n'
            '# HuggingFace embedding  (no issue)\r\n'
            'from langchain_huggingface import HuggingFaceEmbeddings\r\n'
            'embeddings = '
            'HuggingFaceEmbeddings(model="sentence-transformers/all-mpnet-base
-v2")\r\n'
            '\r\n'
            '\r\n'
            '#----------------\r\n'
            '# create langchain-chroma persistent client with collection name
'
            "'example_collection;  (no issue)\r\n"
            'from langchain_chroma import Chroma\r\n'
            '\r\n'
            'vector_store = Chroma(\r\n'
            '    collection_name="example_collection",   # collection is "tabl
e" '
            'in vectore store \r\n'
            '    embedding_function=hf,    # hf is huggingface embeddings '
            'derived  from the previous step \r\n'
            '    persist_directory="./vectorstore/chroma_langchain_db",  # Whe
re '
            'to save data locally, remove if not necessary\r\n'
            ')\r\n'
            '\r\n'
            '\r\n'
            '#----------------\r\n'
            '# add at least one document into  vector collection (no issue)\r\
n'
            'from uuid import uuid4\r\n'
            'from langchain_core.documents import Document\r\n'
            '\r\n'
            'document_1 = Document(\r\n'
            '    page_content="I had chocolate chip pancakes and scrambled egg
s '
            'for breakfast this morning.",\r\n'
            '    metadata={"source": "tweet"},\r\n'
            '    id=1,\r\n'
            ')\r\n'
            '\r\n'
```

```
          'documents = [\r\n'
          '    document_1,\r\n'
          ']\r\n'
          '\r\n'
          'uuids = [str(uuid4()) for _ in range(len(documents))]\r\n'
          'vector_store.add_documents(documents=documents, ids=uuids)\r\n'
          '\r\n'
          '\r\n'
          '\r\n'
          '#----------------  ERROR ENCOUNTERED when running get_by_ids \r\
n'
          '# attempt to run get_by_Ids yields NotImplementedError\r\n'
          "vector_store.get_by_ids(['6314982d-455f-47cc-bf97-6e5324f6af62'])
\r\n"
          '\r\n'
          '```\n'
          '\n'
          '### Error Message and Stack Trace (if applicable)\n'
          '\n'
          '{\r\n'
          '\t"name": "NotImplementedError",\r\n'
          '\t"message": "Chroma does not yet support get_by_ids.",\r\n'
          '\t"stack": '
          '"---------------------------------------------------------------
----------\r\n'
          'NotImplementedError                       Traceback (most recent
'
          'call last)\r\n'
          'Cell In[87], line 3\r\n'
          '      1 # testing get the first two document ids\r\n'
          "      2 # ids = ['db1e5f74-f18d-4765-a193-d30eaed7552f', "
          "'12861b34-df54-4e40-8e1e-ae9ea901d378']\r\n"
          '----> 3 '
          "vector_store.get_by_ids(['6314982d-455f-47cc-bf97-6e5324f6af62'])
\r\n"
          '      5 # get_by_ids() functionality is not avaiable until '
          'v0.2.11\r\n'
          '\r\n'
          'File '
          '~/Documents/0_-_Python_Projects/05_Gen_AI/venv_3_11/lib/python3.1
1/site-packages/langchain_core/vectorstores/base.py:164, '
          'in VectorStore.get_by_ids(self, ids)\r\n'
          '    140 \\"\\"\\"Get documents by their IDs.\r\n'
          '    141 \r\n'
          '    142 The returned documents are expected to have the ID field
'
          'set to the ID of the\r\n'
          '   (...)\r\n'
          '    161 .. versionadded:: 0.2.11\r\n'
          '    162 \\"\\"\\"\r\n'
```

```
            '    163 msg = f\\"{self.__class__.__name__} does not yet support
'
            'get_by_ids.\\"\r\n'
            '--> 164 raise NotImplementedError(msg)\r\n'
            '\r\n'
            'NotImplementedError: Chroma does not yet support get_by_ids."\r\
n'
            '}\n'
            '\n'
            '### Description\n'
            '\n'
            'I am just trying to run the vector_store method `get_by_ids`  - i
t '
            'is listed as one of the available methods in '
            '[here](https://python.langchain.com/api_reference/chroma/vectorst
ores/langchain_chroma.vectorstores.Chroma.html)\r\n'
            '\n'
            '\n'
            '### System Info\n'
            '\n'
            '$ python -m langchain_core.sys_info\r\n'
            '\r\n'
            'System Information\r\n'
            '------------------\r\n'
            '> OS:  Darwin\r\n'
            '> OS Version:  Darwin Kernel Version 23.6.0: Mon Jul 29 21:13:00
'
            'PDT 2024; root:xnu-10063.141.2~1/RELEASE_X86_64\r\n'
            '> Python Version:  3.11.10 (main, Nov 19 2024, 15:24:32) [Clang '
            '12.0.0 (clang-1200.0.32.29)]\r\n'
            '\r\n'
            'Package Information\r\n'
            '-------------------\r\n'
            '> langchain_core: 0.3.19\r\n'
            '> langchain: 0.3.7\r\n'
            '> langchain_community: 0.3.4\r\n'
            '> langsmith: 0.1.143\r\n'
            '> langchain_chroma: 0.1.4\r\n'
            '> langchain_experimental: 0.3.3\r\n'
            '> langchain_groq: 0.2.1\r\n'
            '> langchain_huggingface: 0.1.2\r\n'
            '> langchain_text_splitters: 0.3.2\r\n'
            '\r\n'
            'Optional packages not installed\r\n'
            '-------------------------------\r\n'
            '> langgraph\r\n'
            '> langserve\r\n'
            '\r\n'
            'Other Dependencies\r\n'
            '------------------\r\n'
```

```
'> aiohttp: 3.11.6\r\n'
'> async-timeout: Installed. No version info available.\r\n'
'> chromadb: 0.5.20\r\n'
'> dataclasses-json: 0.6.7\r\n'
'> fastapi: 0.115.5\r\n'
'> groq: 0.12.0\r\n'
'> httpx: 0.27.2\r\n'
'> httpx-sse: 0.4.0\r\n'
'> huggingface-hub: 0.26.2\r\n'
'> jsonpatch: 1.33\r\n'
'> numpy: 1.26.4\r\n'
'> orjson: 3.10.11\r\n'
'> packaging: 24.2\r\n'
'> pydantic: 2.9.2\r\n'
'> pydantic-settings: 2.6.1\r\n'
'> PyYAML: 6.0.2\r\n'
'> requests: 2.32.3\r\n'
'> requests-toolbelt: 1.0.0\r\n'
'> sentence-transformers: 3.3.1\r\n'
'> SQLAlchemy: 2.0.36\r\n'
'> tenacity: 9.0.0\r\n'
'> tokenizers: 0.20.3\r\n'
'> transformers: 4.46.3\r\n'
'> typing-extensions: 4.12.2',
'_closedAt': '2024-12-31T00:36:30Z',
'_createdAt': '2024-11-22T01:13:50Z',
'_issueNumber': '28276',
'_repo': 'langchain',
'_state': 'open',
'_title': 'langchain-chroma== 0.1.4   method get_by_ids is listed in '
          'documentation BUT I am getting NotImplementedError',
'_type': 'issue'}
```

In [33]:
```
#Number of Issues in the given timeframe
pprint(len(issues))
```

```
3879
```

In [34]:
```
# Convert the list of Issues to a DataFrame
df_Issues = pd.DataFrame(issues)
```

In [35]:
```
# Replacing all NaN values with None in columns as elasticsearch does not re
df_Issues.fillna("None", inplace=True)
```

In [36]:
```
# Function to create embeddings from OpenAI API
def embed(texts):
    # Make a request to OpenAI API to get embeddings
    embeddings = client.embeddings.create(
        input=texts,
        model='text-embedding-ada-002'
```

```
    )
    # Extract embeddings from the API response
    return [result.embedding for result in embeddings.data]
```

In [ ]:

In [37]:
```
## Embedding creation using openAI of GitHub Issues.

from openai import OpenAI
from tqdm import tqdm
import time

# Initialize OpenAI client with API key
client = OpenAI(api_key="sk-proj-b_EAft00QZDKM8-lKhXuEhXlc5GLUSIYw7T7kZ9WD36

Issue_embeddings = []

# Batch size for processing data
batch_size = 500

# Initialize data structure for storing text
data = [
    [], # Titles
]
count=0;
# Embed and insert in batches
for i in tqdm(range(0, len(df_Issues))):
    title = str(df_Issues.iloc[i]['_title']).replace("\n", "") or ''
    body = str(df_Issues.iloc[i]['_body']).replace("\n", "") or ''

    # Merge 'repository name','title' and 'body' of the GitHub Issue
    combined_text = f"Repository:{owner}/{repo} Issue Title:{title} Issue Bo
    data[0].append(combined_text)
    if len(data[0]) % batch_size == 0:
        print("Embedding batch...")

        embeddings_batch = embed(data[0])
        Issue_embeddings.extend(embeddings_batch)
        data = [[]]
        print("Waiting for 1 minute before the next batch...")

        time.sleep(50)

# Embed the remaining data if any
if len(data[0]) != 0:
    embeddings_rem = embed(data[0])
    print(len(embeddings_rem))
    Issue_embeddings.extend(embeddings_rem)
```

```
  0%|                                                  | 0/3879 [00:00<?, ?i
t/s]
Embedding batch...
Waiting for 1 minute before the next batch...
 13%|█▎                                    | 500/3879 [01:05<07:20,  7.68i
t/s]
Embedding batch...
Waiting for 1 minute before the next batch...
 26%|██▌                                   | 1000/3879 [02:06<06:03,  7.91i
t/s]
Embedding batch...
Waiting for 1 minute before the next batch...
 39%|███▊                                  | 1500/3879 [03:08<04:56,  8.02i
t/s]
Embedding batch...
Waiting for 1 minute before the next batch...
 52%|█████                                 | 2000/3879 [05:28<05:49,  5.38i
t/s]
Embedding batch...
Waiting for 1 minute before the next batch...
 64%|██████▎                               | 2500/3879 [07:01<04:16,  5.37i
t/s]
Embedding batch...
Waiting for 1 minute before the next batch...
 77%|███████▌                              | 3000/3879 [08:02<02:24,  6.09i
t/s]
Embedding batch...
Waiting for 1 minute before the next batch...
100%|██████████████████████████████████████| 3879/3879 [09:02<00:00,  7.15i
t/s]
379
```

In [38]:
```
# Adding Generated embeddings to GitHub_Issue_vector column in the datafram

df_Issues["GitHub_Issue_vector"] = Issue_embeddings
```

In [39]:
```
# Check if the new Column is created
df_Issues.tail()
```

Out[39]:

	_type	_repo	_issueNumber	_title	_createdAt	
3874	issue	elasticsearch	113347	Update JDK version in CONTRIBUTING.md	2024-09-22T16:38:04Z	26
3875	issue	elasticsearch	113346	deps(updatecli): bump all policies	2024-09-22T06:22:37Z	23
3876	issue	elasticsearch	113345	[CI] KibanaUserRoleIntegTests testSearchAndMSe...	2024-09-22T05:36:19Z	25
3877	issue	elasticsearch	113344	[CI] RollupIndexerStateTests testMultipleJobTr...	2024-09-22T05:16:59Z	13
3878	issue	elasticsearch	113343	[CI] DocsClientYamlTestSuiteIT test {yaml=refe...	2024-09-22T04:44:47Z	05

In [40]:
```
# Configure Elasticsearch connection
from elasticsearch import Elasticsearch,helpers
es = Elasticsearch(['http://localhost:9200'])
es.ping()   #connection testing
```

Out[40]: True

In [41]:
```
# Define functions to generate actions for Indexing
def generate_actions(issues):
    for issue in issues:
        yield {
            "_op_type": "index",  # Action type: 'index' for inserting
            "_index": index_name,  # Index name
            "_source": issue  # The document
        }
```

In [42]:
```
# Indexing the documents
index_name = 'pdgithub_issues'
success, failed = helpers.bulk(es, generate_actions(issues), index=index_nam

# Print the results
print(f"Successfully indexed {success} documents.")
print(f"Failed to index {failed} documents.")
```

```
Successfully indexed 3879 documents.
Failed to index [] documents.
```

In [43]:
```
#Index Mapping for githubissues
```

```
index_mapping= {
    "properties": {
      "GitHub_Issue_vector": {
          "type": "dense_vector",
          "dims": 1536,
          "index": "true",
          "similarity": "cosine"
      },
     "_type": {"type": "text"},
     "_repo":{"type":"text"},
     "_issueNumber": {"type": "long"},
     "_title": {"type": "text"},
     "_createdAt": {"type": "date"},
     "_closedAt": {"type": "text"},
     "_state": {"type": "text"},
     "_body": {"type": "text"}
   }
}

if es.indices.exists(index="pdgithub_issues"):
    es.indices.delete(index="pdgithub_issues")

es.indices.create(index="pdgithub_issues", body={"mappings": index_mapping})
```

Out[43]:
```
ObjectApiResponse({'acknowledged': True, 'shards_acknowledged': True, 'inde
x': 'pdgithub_issues'})
```

In [ ]:

In [44]:
```
# Bulk indexing for githubissues

def dataframe_to_bulk_actions(df_Issues):
    for index, row in df_Issues.iterrows():
        yield {
            "_index": 'pdgithub_issues',
            "_source": {
                "_type": row['_type'],
                "_repo":row['_repo'],
                "_issueNumber": row['_issueNumber'],
                "_title": row['_title'],
                "_createdAt": row['_createdAt'],
                "_closedAt": row['_closedAt'],
                "_state": row['_state'],
                "_body": row['_body'],
                "GitHub_Issue_vector": row['GitHub_Issue_vector']
            }
        }

start = 0
```

```
end = len(df_Issues)
batch_size = 500

for batch_start in range(start, end, batch_size):
    batch_end = min(batch_start + batch_size, end)
    batch_dataframe = df_Issues.iloc[batch_start:batch_end]
    actions = list(dataframe_to_bulk_actions(df_Issues.iloc[start:end]))

success, failed = helpers.bulk(es, actions)
print(f"Inserted {success} records into Elasticsearch. Failed records: {fail
```

```
Inserted 3879 records into Elasticsearch. Failed records: []
```

In [ ]: