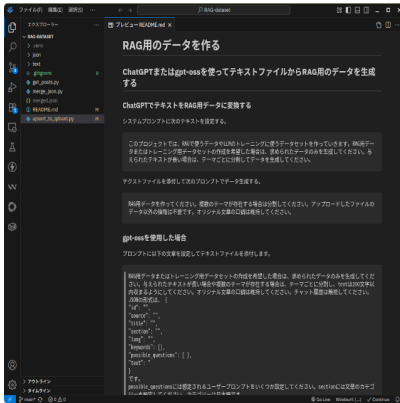


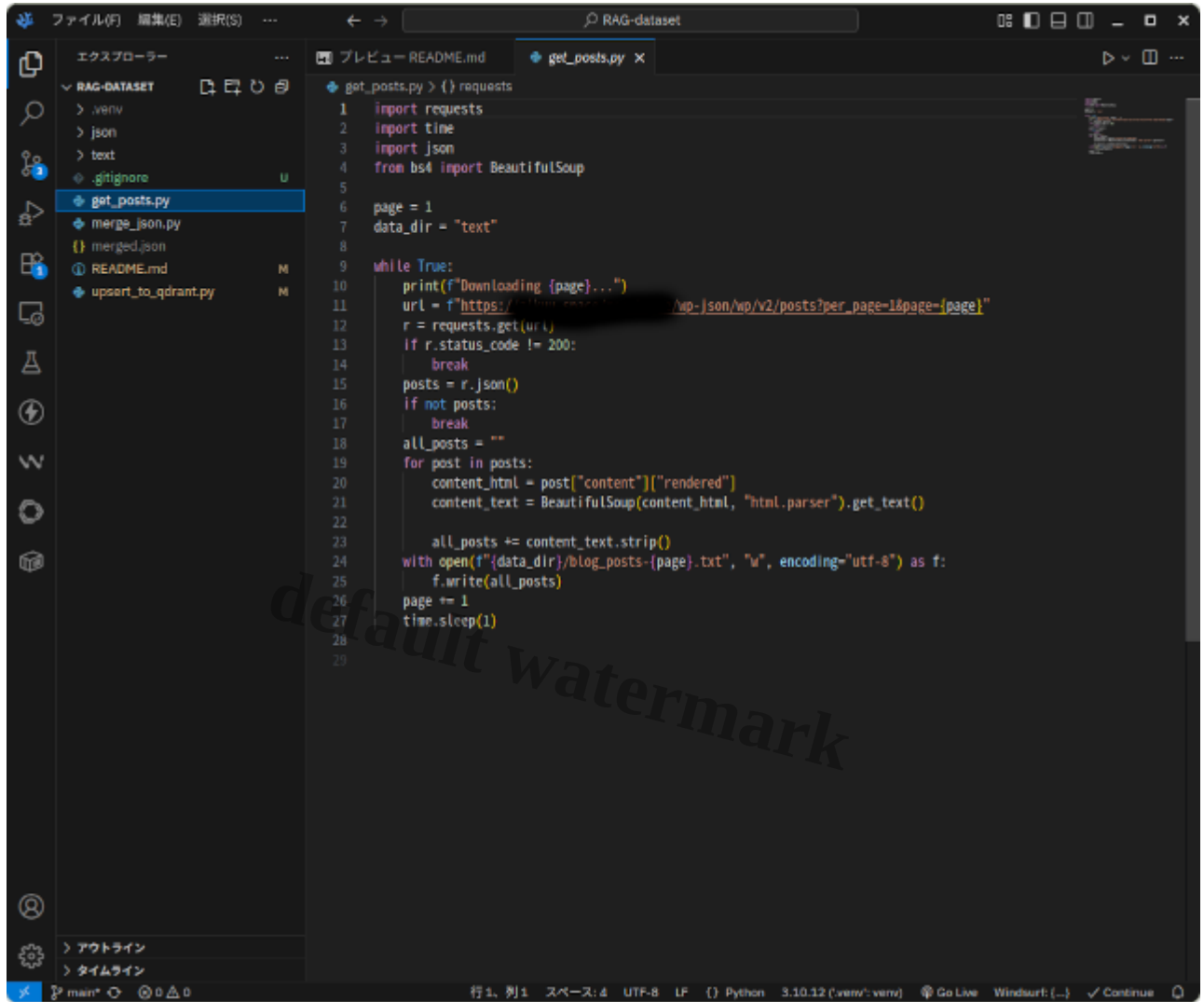


RAGç?ˆã??ã?¼ã?¿ã??ä½?æ?•ã?•ã?|Qdrantã•«ç?»é?²ã?•ã??

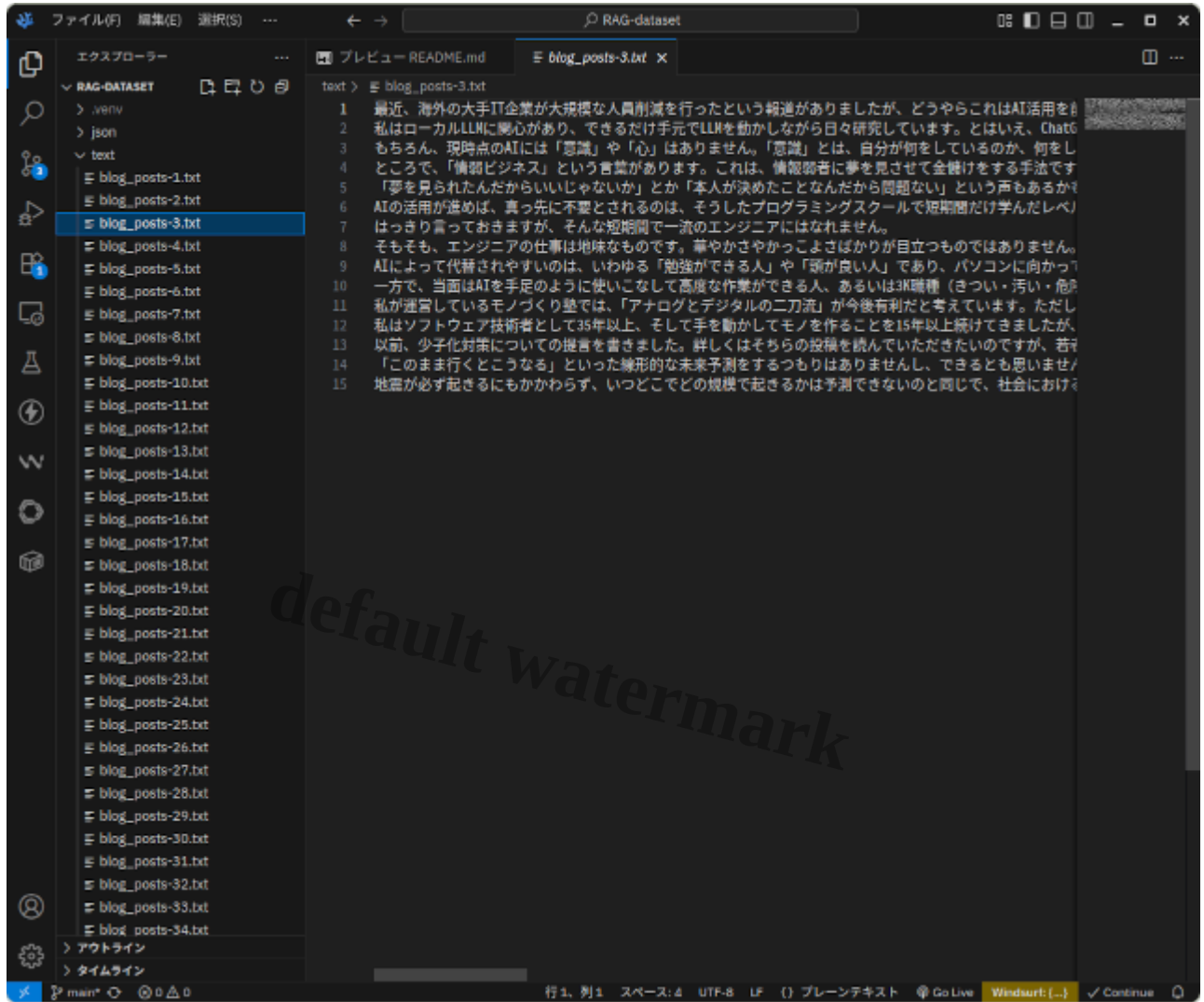
Description

à»?æ?¥â•̄-â?â•̄®â??â?â?•̄â•̄®è°?â°?â??â??â?|â?³â?â?¼â?°â?â•̄-â?â?•̄GPT5â??gpt-ossâ•̄Sâ½?æ?•̄â•̄
 ?â?•̄RAGç?•̄â??Qdrant â??â?•̄â?¿â?¼â??â?¼â?¿â??â?¼â?¹â«ç?»é?â?â??â??â?-
 â?°â?©â? â??æ?•̄,â•̄â•̄¾â?°â?â?â??â??â?â?°â?©â? â•̄-â?â?â?â?|Pythonâ•̄Sè°?è¿°â??â?•̄
 â??â??â??æ?£â?°â?°â?µâ?³â??â?«â?³â?¼â??â?°â?â?â?â?â?â?è¿°â?â?°â??â?â?®â•̄Sâ?é?£â?°â?â??â?-
 â?°â?©â? â•̄-â?â?â?â?•̄æ?•̄â?°â?¾â?°â??

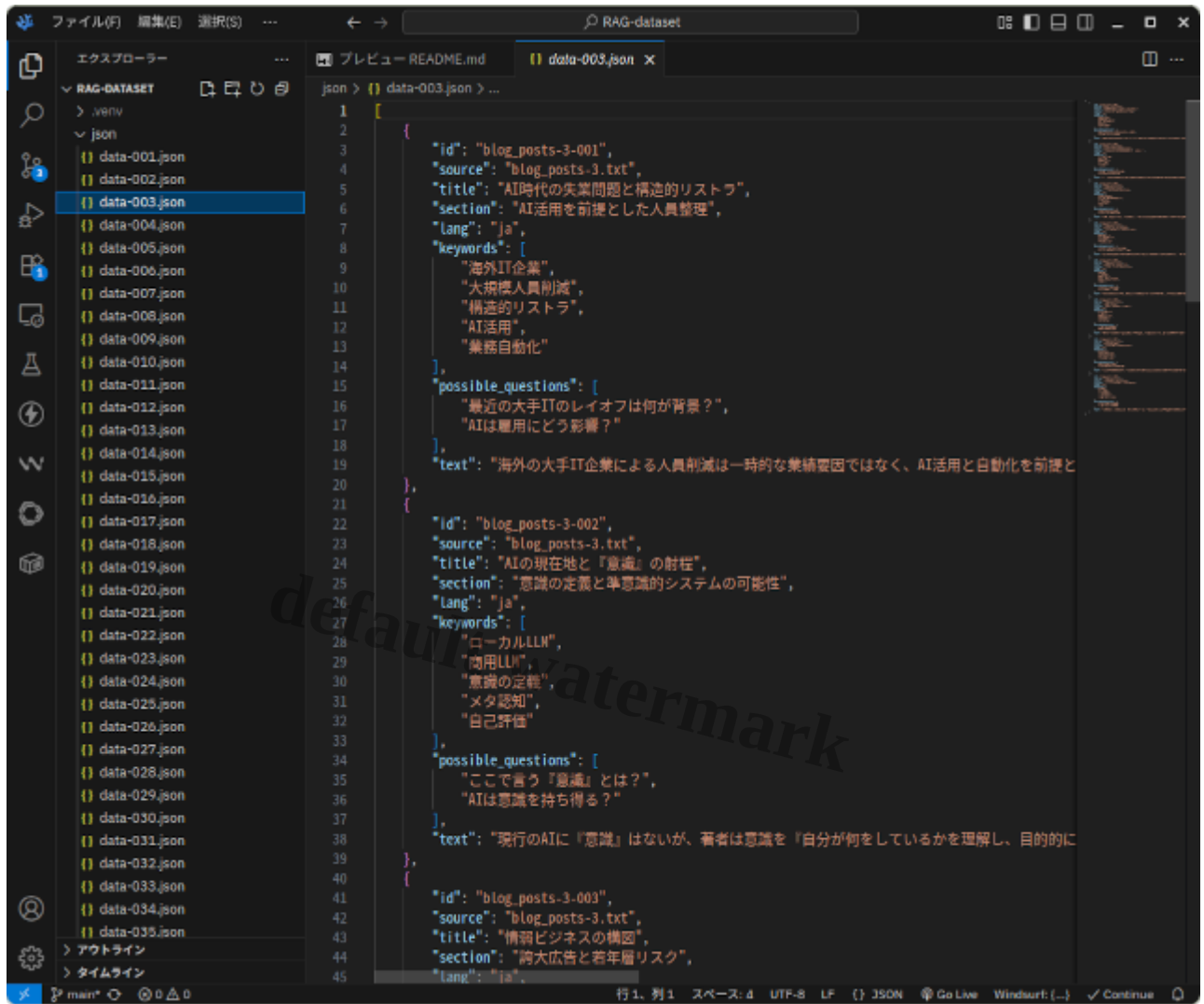
ã??ã?ã°ã•?ã??è"?ä°?ã•®æ?-æ??ã??ã??ã?|ã?³ã?ã?¼ã??ã•?ã??ã??ã?ã°ã?©ã? â•¯ç°|â?ã•ã??ã•
®ã•§ã•?ã??



ã·?ã·@ã??ã?ã?°ã?©ã? ã??ã½¿ã·£ã·|ã??ã?ã?°ã·?ã??ã??ã?|ã?³ã?ã?¼ã??è·?ã°?ã·@æ?-æ??ã·-ã?ã·
®ã??ã·?ã·ª½¿ã·§ã·?ã??



ã•ã??ã??GPT5ã??gpt-ossã•@ã??ã??ã??ã??ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?
jã??ã??ã?»¥å?•ã•@æ??ç`¿ã•sã??è§ã??ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?
@å•¥å?ã•sGPT5ã•å?£ã??ã?
•@æ?¹ã?ã?-ã?¹ã?ã?³ã?¹ã?é??ã?ã?ã?½¿ã?ã?¿ã?°ã?è?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?



ã?ã?ã?ã?ã½¢ã«æ§?é? å??ã?ã??ã?ã?ã?ã?ã?ã?ã½?æ¥ã?æ¥½ã«ã?
ã??ã¾ã?ã??

ã?ã?ã¾ã§ã®ã½?æ¥ã¯æ?¬æ¥ãã??ã°èª??ã??ã?ã?ã?ã?æ?ã£ã?ã?ã¾ã?ã?ã?ã?ã?
¾æ??ã?ã§ã¯ã?æ?ã?ã??ã?ã??ã?¾ã?ã??ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?ã?
ã?
ã?
ã?

ã?ã®æ??ã?
ã??ã??ã?
ã??ã??ã?
ã??ã?
ã?
ã?
ã?

ã?æ?ã?

The image shows a VS Code editor window with a Python script named `merge_json.py` open. The script is designed to merge multiple JSON files into a single JSON file. It uses `argparse` for command-line arguments and `pathlib` for file handling. The script includes comments in Japanese explaining its functionality and usage.

The left sidebar shows the project structure with files like `.venv`, `json`, `text`, `.gitignore`, `get_posts.py`, `merge_json.py`, `merged.json`, `README.md`, and `upsert_to_qdrant.py`.

The bottom status bar shows the file is named `main.py` and is located in a directory named `Kazuo Tsubaki`.

```
#!/usr/bin/env python3
# merge_json.py
"""
指定ディレクトリ内の *.json を読み込み、1つの JSON ファイルに結合します。
- 各ファイルのトップレベルが list: そのまま結合 (extend)
- それ以外 (dict/str/num 等) : 結果配列に1要素として追加 (append)

使い方:
python merge_json.py --dir ./data --out merged.json
python merge_json.py --dir ./data --out merged.json --recursive --indent 2
"""

from __future__ import annotations
import argparse
import json
import sys
from pathlib import Path
from json import JSONDecodeError
import uuid

def parse_args() -> argparse.Namespace:
    p = argparse.ArgumentParser(
        description="ディレクトリ内のJSONファイルを結合して1つのJSONにします。"
    )
    p.add_argument("--dir", "-d", type=str, default=".", help="入力ディレクトリ (デフォルト: 現在)")
    p.add_argument("--out", "-o", type=str, default="merged.json", help="出力ファイルパス (デフォルト: *.json)")
    p.add_argument("--pattern", type=str, default="*.json", help="検索パターン (デフォルト: *.json)")
    p.add_argument("--recursive", "-r", action="store_true", help="サブディレクトリも含める")
    p.add_argument("--indent", type=int, default=None, help="整形インデント幅, 未指定なら最小化")
    p.add_argument("--encoding", type=str, default="utf-8", help="入出力の文字エンコーディング (5)")
    return p.parse_args()

def iter_json_files(root: Path, pattern: str, recursive: bool):
    if recursive:
        yield from sorted(root.rglob(pattern))
    else:
        yield from sorted(root.glob(pattern))

def main():
    args = parse_args()
    root = Path(args.dir)
    if not root.exists() or not root.is_dir():
```

i¼?æ?-å??ã·?ã??ã?;ã?õã?«ã·-ã?ã·?ã·®ã??ã?ã°ã?©ã? ã·S Qdrantã·«ä¿·å?ã·?ã·¾ã·?ã??ã·?ã·
®ã??ã?ã°ã?©ã? ã??æ?,ã·ç·®ã·«i¼?æ???é??ã»ã·©ã·?ã·?ã??ã·¾ã·?ã·?ã??

The screenshot shows a VS Code editor with a Python script named `upsert_to_qdrant.py`. The script is designed to interact with a Qdrant vector database. It includes the following code:

```

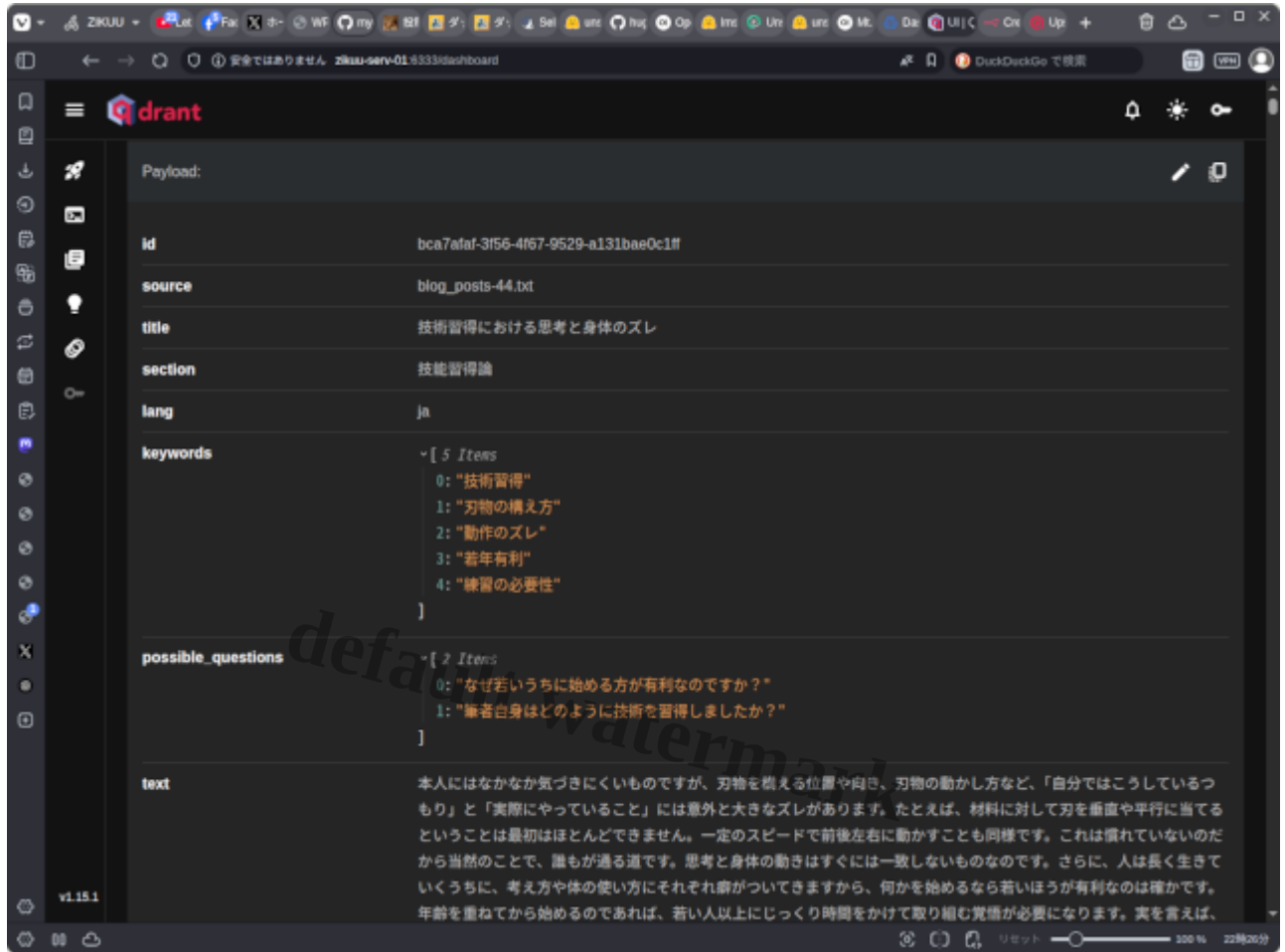
1  import requests
2  from qdrant_client import QdrantClient
3  from qdrant_client.models import VectorParams, Distance, PointStruct
4  from uuid import uuid4
5  from pathlib import Path
6  from typing import Any, Dict, List
7  import json
8
9  collection_name = "demo"
10 json_path = ".merged.json"
11 ollama_host = "http://localhost:11434"
12 embeddings_model = "mxbai-embed-large"
13 client = QdrantClient(url="http://localhost:6333")
14
15 if not client.collection_exists(collection_name=collection_name):
16     client.create_collection(
17         collection_name=collection_name,
18         vectors_config=VectorParams(size=1024, distance=Distance.COSINE))
19
20 def load_merged_json() -> List[Dict[str, Any]]:
21     """JSON ファイルを読み込み、トップレベルのリストとして返す"""
22     path = Path(json_path)
23     with path.open("r", encoding="utf-8") as f:
24         data = json.load(f)
25     if not isinstance(data, list):
26         raise ValueError("JSON のトップレベルはリストである必要があります。")
27     return data
28
29 def query_relevant_data(prompt: str):
30     response = requests.post(
31         f"{ollama_host}/api/embed",
32         json={"model": embeddings_model, "input": prompt} )
33     data = response.json()
34     embeddings = data["embeddings"][0]
35
36     adjusted_prompt = f"Represent this sentence for searching relevant passages: {prompt}"
37     response = requests.post(
38         f"{ollama_host}/api/embed",
39         json={"model": embeddings_model, "input": adjusted_prompt} )
40     data = response.json()

```

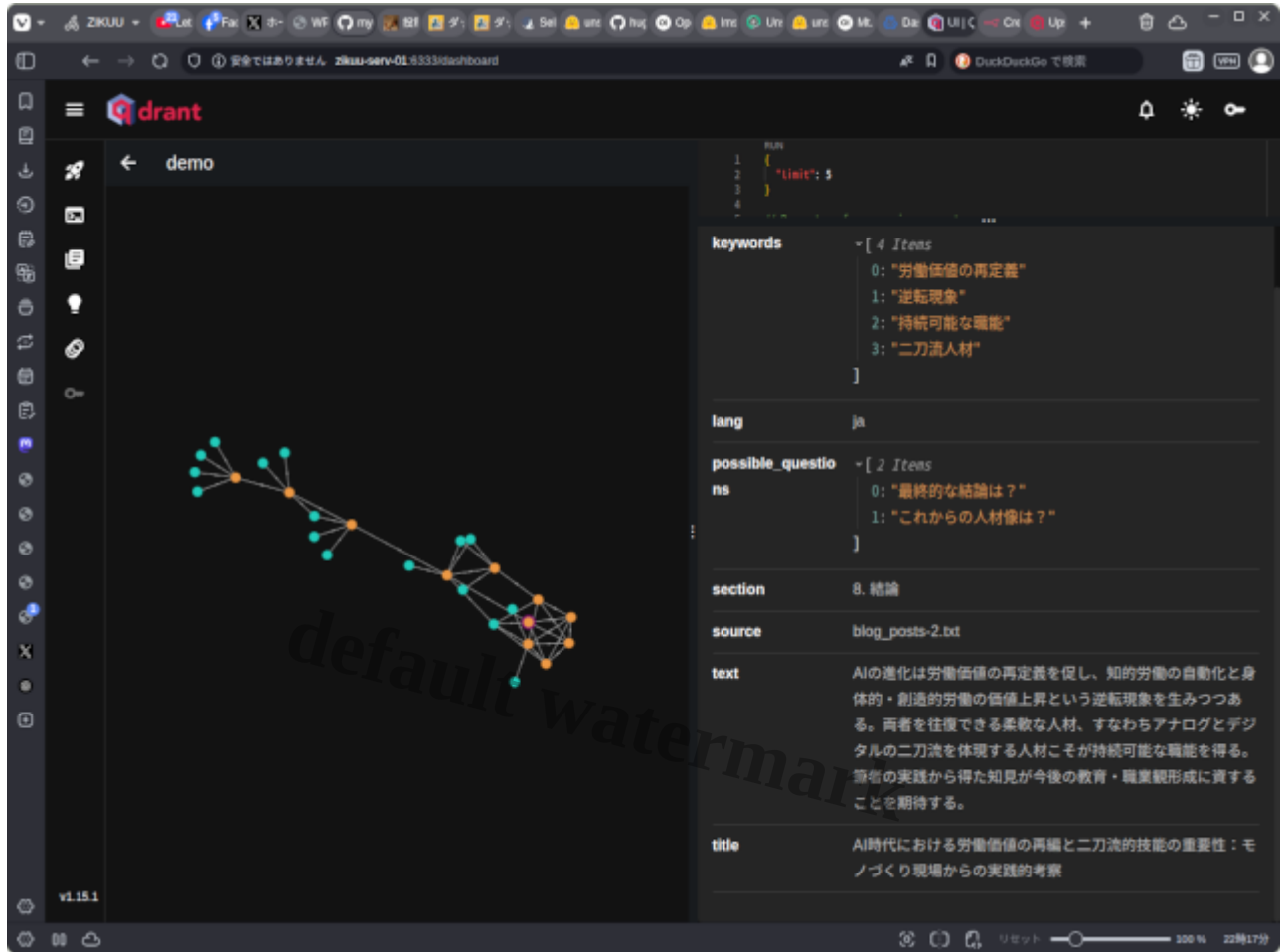
The interface also shows a file explorer on the left with a project named `RAG-DATASET` containing files like `.env`, `json`, `text`, `.gitignore`, `get_posts.py`, `merge_json.py`, `merged.json`, `README.md`, and `upsert_to_qdrant.py`. The bottom status bar indicates the current file is `main.py` in the `Python` environment, with a file size of 10.1 KB and a UTF-8 encoding.

à»?â??ã·-â??ã?è¼¼ã·¿ã?£ã??ã?«ã«Ollamaã·®mxbai-embed-largeã??ã½¿ã·?ã·¼ã·?ã·?ã??GPUé·
 ?æ·è¼¼?ã·®é·ã·?ã?µã?¼ã·?ã?¼ã·Sã??ã·?ã!ã·?ã??Ollamaã??ã?©ç?·ã·?ã·?ã·®ã·Sã·?350à»¶ã·
 ®ã??ã?¼ã·¿ã??Qdrantã«çç?»é²ã·?ã??ã·®ã«5ã??ã»ã·©ã·?ã·?ã??ã·¼ã·?ã·?ã??GPUæ·è¼¼?æ©?ã·
 ¢ã??2000à»¶ã·Sã??1ã??ç?·ã!¶ã·Sçµ?ã·?ã??ã?¶ç?ã·ã·æ·?ã·?ã??ã·¼ã·?ã??

ã·?ã·jã??ã·?Qdrantã·«ç?»é?²ã·?ã·?ã·?ã·?¼ã·¿ã·šã·?ã·??



ã??ã?ã?¿ã?¼ã??ã?¼ã?¿ã?¿ã?¼ã?¹ã-ã??ã?¼ã?¿ã®æ?·ã?³ã??ã??ã?-ã??ã?«ã?²ã·ã·?ã·à¿·ã?ã·
 ?ã·?ã??ã?¼ã?¿ã??ã?¼ã?¹ã·Sã·?ã??ã?¼ã?¿ã·?ã??ã?¼ã?¿ã®è¿ã·?ã??æ²?ç´£ã·Sã·ã??ã??ã·?ã·
 «ã·ã·£ã·jã·?ã·¾ã·?ã??Qdrantã«ã·-ã??ã?¼ã?¿ã®ä½·ç½®é?£ã¿ã??ã?°ã?©ã??èj´ç²°ã·
 ?ã??æ©?è?½ã·?ã·?ã??ã·¾ã·?ã??



ä»?å¾?ã•®ä½?æ¥ã•ã•?ã•|ã•ã•

1. JSONă•«æ?ªă•?æ•?ă•®æ®?ă?ªă•®ă??ă?¼ă?¿ă•®ă•?æ•?ă?ª?è?ª•?
2. ă•?ă!ă?ª ă?ª?ª?¼ă?¿ă??Qdrantă•«ç?»é?ªă•?
3. RAGă?ªè©|ă•?ă?ª?ª?ª?ª°ă?©ă? ă??æ? ,ă•?ă•!ă?¹æ??ă?ªç£èª•ă•?ă??
4. ă•?é?ª?ă•?ă?ª??ă•ă?ª??ă?¼ă?¿ă?ª?è!ç?ª?ă•?ă!ă?ª•¼ă?ª?¼ă?ª?ª?ç!ª?ª?è¿ă?ă•?

ã•ã•?ã•?æ??ã•?ã•§ã•?ã•??

[illegible]

ã??ã?ã?°ã?©ã? ã•ã??ã?¼ã?¿ã•ã?¼ç??ã??ã?¹ã?¿ã??ã??ã?ã?£ã?¬ã?»ã?¹ã§ã•ã?GitHubã?ªã?•
ã?ã??ã?ªã?«¿½¿ã?•ã?¿ã?±æ??ã?•ã?¿¼ã?®æ??æ?ã?«ã?•ã?¿ã?¾ã?•ã??

Category

1. æ?¥ã? ã•®æ'»å??

Tags

1. AI
2. LLM

-
- 3. Python
 - 4. Qdrant
 - 5. RAG
 - 6. ä,?é??å??å,?
 - 7. åð§è|•æ¨jè¨?èª?ã?¢ã??ã?«
 - 8. å±±±æ¢¨ç??

Date Created
2025å¹´8æ??16æ?¥

Author
kazuo-tsubaki

default watermark