Table of Contents

Introduction

I began by testing the connection to the database:

```
# Standard library imports
import os

# Database imports
import psycopg2


db_info = os.getenv("DB_INFO")
if not db_info:
    raise ValueError("Database not found. Please set the DB_INFO environment
variable.")

conn = psycopg2.connect(db_info)
cur = conn.cursor()
#-------------------------------------------------------------------------------
cur.execute("CREATE TABLE test (id serial PRIMARY KEY, num integer, data
varchar);")
cur.execute("INSERT INTO test (num, data) VALUES (%s, %s)", (100, "abc'def"))
cur.fetchone()
conn.commit()

cur.close()
conn.close()
```

As a result, I got a new table with a record in it. Later, I deleted that table since it was just for testing.

After that, I decided to work on document processing function. The goal is to take files that are currently in `../../kb_new/`, get their metadata and extract text and then upload them to the database that I created.

I had to change the metadata that is passed through after processing the document and I got rid of chunking for now for simplicity:

```
metadata = DocumentMetadata(file_name=file_name,
                            file_size=file_size,
                            file_type=file_name.split('.')[-1],
                            page_count=0,  # Placeholder for page count
                            word_count=word_count,
                            char_count=char_count,
                            keywords=[],  # Placeholder for keywords
                            source="system_upload"
```

```
                                abstract="", # Placeholder for abstract
                                ).model_dump()
```

A lot of it is placeholders just to make prototyping faster. Had to do some bug fixing related to names and class' attributes, but I got the desired output:

```
[{'file_name': 'DoT.pdf', 'file_size': 79797, 'file_type': 'pdf', 'page_count': 0,
'word_count': 1018, 'char_count': 3580, 'keywords': [], 'source': 'system_upload',
'abstract': ''}]```

Now I can try to put all this data to the database, along with uploading plain
text to a dedicated table.

```py
metadata, content = load_documents_from_directory(NEW_DOCUMENTS_DIR)
def upload_documents_to_db(metadata, content):
    """
    Upload document metadata and content to the PostgreSQL database.

    Args:
        metadata: Metadata of the document.
        content: Content of the document.

    Returns:
        None
    """
    conn = db_connection()
    cur = create_cursor(conn)

    try:
        # Insert metadata into the database - single document case
        meta = metadata[0]  # Get the single metadata object
        cur.execute("""
            INSERT INTO filemetadata (file_name, file_size, file_type, page_count,
word_count, char_count, keywords, source, abstract)
            VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s)
        """, (meta['file_name'], meta['file_size'], meta['file_type'],
              meta['page_count'], meta['word_count'],
              meta['char_count'], meta['keywords'],
              meta['source'], meta['abstract']))

        # Add this before the INSERT statement
        print(f"Content type: {type(content)}")
        print(f"Content length: {len(content) if hasattr(content, '__len__') else
'N/A'}")
        if isinstance(content, list):
            print(f"Content is a list with {len(content)} items")
            doc = content[0]  # Extract first item if it's a list
        else:
            doc = content
        #print(doc)
```
```

```
        # Insert content into the database - single document case
        cur.execute("""
            INSERT INTO largeobject (plain_text)
            VALUES (%s)
        """, (doc,))

        conn.commit()
    except Exception as e:
        print(f"Error uploading document to the database: {e}")
        conn.rollback()
    finally:
        cur.close()
        conn.close()

upload_documents_to_db(metadata, content)
```

My editor auto-completed a good portion of this function, but I still had to do a bunch of tweakments such as changing the data type of the passed values and I spent a long time trying to fix a bug that was not really a bug.

The "bug" was that the long text is not displayed fully but is condensed like this:

bug-page

That's 30 minutes out of my life gone, but I learned something. 😃