Here, as of my understanding and prior experiences with some exploration, I elaborate the problem to solve in this assessment, what I tried in the code and what can be the future enhancements briefly for your reference. Since you didn't expect the full perfect solution and most of these things are under improvement, I tried for some extent to solve the problem within this time constraint. I can further improve and implement the enhancements with your guidance later.

Problem

This assessment explores enhancing a Graph-based Retrieval-Augmented Generation (RAG) system using an example document, "SAP HANA on VMware vSphere best practices". The desired enhancements are:

1. Building a graph community by pulling all the linked documents including protected documents (eg: SAP) and handling unavailable document.
2. Including nodes for diagrams and images in the graph.
3. Performing chain of thought when answering queries.
4. Let to answer directly and correctly for some sort of questions like parameter settings and best practices.
5. Let to work the similar concepts queries.

Initial setup

First, I cloned the repo to my local machine and try to run the code for an understanding with *gemma 2B* model with *ollama*. Then I tried with *deepseek-r1* with *ollama* and created a free instance in *neo4j Aura* also. Since I don't have enough GPU, I couldn't run the repository. So, I bought *RTX 5000, 24GB VRAM* in *Runpod* for some days and setup the vscode in my local machine via ssh and clone the repository and created virtual environment within the persistent volume in the pod.

I installed all requirements and run the repo successfully. Due to the resource constraint, to reduce the time consumption to create the knowledge graph triplets and indexes and split the pdf to small pages and test with some different queries to understand the above limitations.

Tried things with results

1. **Building a graph community by pulling all the linked documents including protected documents and handling unavailable document.**
    - Link Extraction: Uses PyMuPDF (fitz) to parse all URI links from each page.
    - Classification: SAP Notes (sap.com + note in the link),PDFs (.pdf ending),Other links.

- SAP SSO Detection: Recognizes if a link leads to an SAP login page (by checking HTML patterns).
- Retry Mechanism: Retries failed downloads with exponential backoff.
- Metadata Generation: Stores metadata including page number, download status, and category.
- Download Directory: Saves files to ./linked_docs
  The code is in the **pull_links.ipynb**

2. **Including nodes for diagrams and images in the graph.**
   - Tried Blip image captioning model (base variant) for getting the caption for the images extracted from the pdf. (Have the idea of giving that caption for graph database with relevant node) – not a good performance (can validate in **diagram_image.ipynb** file in repo) in the SAP HANA pdf diagrams but good performance in some other pdfs with images.
   - Tried with llava-hf model for understanding the model and generate the captions – better performance
   - Tried with google's Pix2Struct base model – better performance

3. **Performing chain of thought when answering queries and answering directly for some sort of queries with the understanding of similar concepts queries.**
   - Added prompting for CoT to the LLM before passing the retrieved graph nodes.
     - Eg: "Please think step-by-step and explain your reasoning clearly. "
       "Break the explanation into logical parts before giving your final answer."
   - Tried hybrid retrieving means graph + text retrieval
     - I used *PropertyGraphIndex* for creating the knowledge graph with *neo4j* and LLM *bge-m3* for creating vector embedding with text to semantic understanding – this works better

   Codes are available in **cot_hybrid_retriever.py**

   Sample queries,

What are the things, limit the maximum number of vCPUs and vRAM available for a VM?

```
So, putting it all together: The main factors that limit vCPUs and vRAM are:

1. **CPU Type**: Different Intel CPUs have varying maximums.
2. **Hypervisor Version**: VMware versions set specific limits.
3. **Workload Type**: OLTP or OLAP affects resource allocation.
4. **Infrastructure Design**: Use of VCF features like vSAN can reduce available resources.
```

which component are involved in setting SAP HANA and VMWare according to best practice

?

```
So, the main components involved would include:

1. **Hardware**: Specifically 2-socket Sapphire Rapids servers supporting SNC-2.
2. **Virtualization Platform**: VMware vSphere (ESXi 7/8) with appropriate updates.
3. **Storage Solution**: Likely using vSAN or other scalable storage solutions mentioned.
4. **Networking**: VLANs for traffic segmentation, as noted before.
5. **Memory and CPU Configuration**: Proper allocation considering NUMA alignment and SNC settings.
6. **VM Setup**: Configuring VMs with correct vCPUs, memory, and sockets as per best practices.
7. **Prerequisites**: Ensuring compatibility between hardware, software versions, and SAP requirements.
8. **Deployment Strategy**: Using VCF with separate domains for SAP HANA to optimize performance.
```

I didn’t change anything in main.py. I tried to implement every possible issue mentioned in the assessment separately in the repository.