Comparative analysis of Open WebUI & AI Essentials Virtual Assistant

Objectives

After completing this lab, you will be able to:

1. Understand the differences between OpenWebUI and Virtual Assistant (HPE AI Essentials) in a RAG-based implementation.
2. Learn how to access and interact with both platforms.
3. Compare response quality, retrieval accuracy, and document retrieval capabilities.
4. Analyze the usability and effectiveness of both implementations for different AI-driven tasks.

Description

Retrieval-Augmented Generation (RAG) is an advanced AI technique that enhances generative models by integrating external document retrieval. This lab explores two implementations of RAG: OpenWebUI (using OpenWebUI pipelines) and HPE's AI Essentials Virtual Assistant. Participants will interact with these platforms, evaluate their responses, and analyze how each system retrieves and generates information. By the end of the lab, learners will understand the key differences in retrieval strategies, response accuracy, and document support across both implementations.

Key Concepts

1. **Retrieval-Augmented Generation (RAG) Pipeline**: A RAG pipeline is an AI architecture that combines retrieval and generation capabilities to produce contextually accurate answers. In this setup, the pipeline first retrieves relevant documents based on a query and then uses a language

model to generate a response based on the retrieved content. This approach improves accuracy by grounding responses in real information from the provided documents.

2. **HPE AI Essentials Virtual Assistant**: A proprietary AI-powered assistant integrated into HPE's Private Cloud AI, designed to enhance knowledge retrieval.
3. **OpenWebUI**: OpenWebUI is a web-based interface that serves as the front end for interacting with the RAG pipeline. It enables users to input queries, which are then sent to the backend for processing. By linking OpenWebUI with the pipeline, the lab enables an intuitive, interactive experience, where users can ask questions and receive responses directly on a web page.
4. **LangChain Library**: LangChain is a library for managing AI pipelines, enabling document processing and language model integration. In this lab, LangChain manages text processing, chunking of documents, and interaction with the language model, simplifying the process of creating and executing language models that rely on document retrieval.

Task 1: Accessing and Interacting with OpenWebUI

OpenWebUI provides an intuitive interface for interacting with Retrieval-Augmented Generation (RAG) pipelines. This task will guide you through accessing OpenWebUI, querying different models, and analyzing their responses. You will compare how different models within OpenWebUI handle the same query, helping you understand variations in retrieval accuracy and response quality.

A. Open the OpenWebUI Interface

Access OpenWebUI to ensures proper access before proceeding with further interactions.

1. Open a web browser and go to http://10.79.253.112:3000

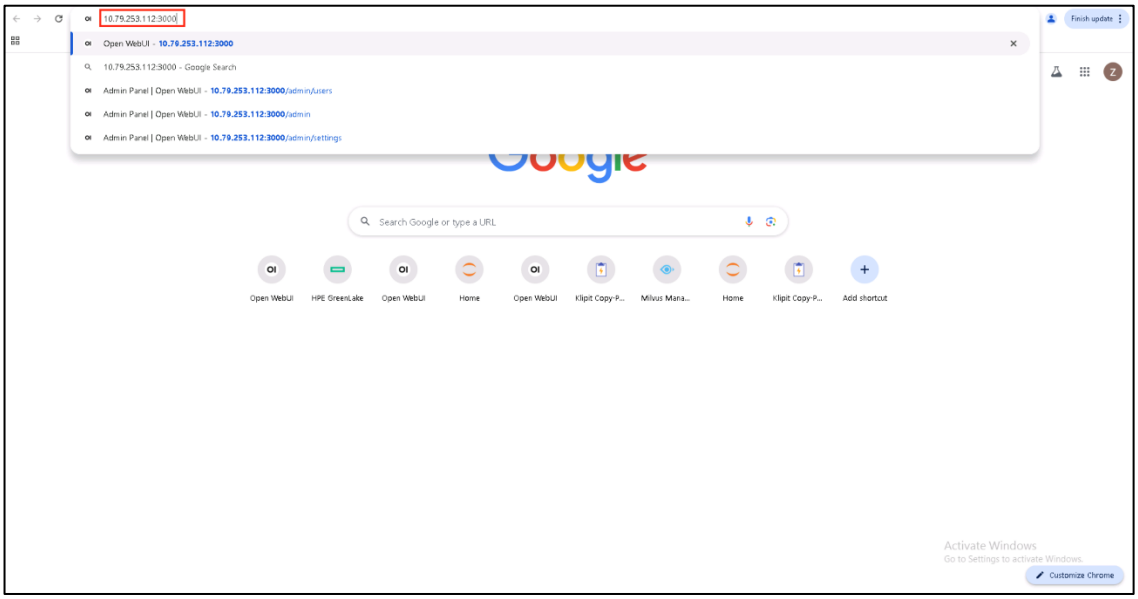


Figure 1: Accessing OpenWebUI by entering the URL in the browser.

2. Wait for the interface to load and verify you can access the chat interface.

Figure 2: OpenWebUI interface

B. Query Using Llama3.1:8B Model

Submit a query to the Llama3.1:8B model within OpenWebUI to observe how it processes and generates responses. Focus on evaluating its ability to provide accurate and contextually relevant answers.

1. Locate the model dropdown menu and select Llama3.1:8B from the dropdown.

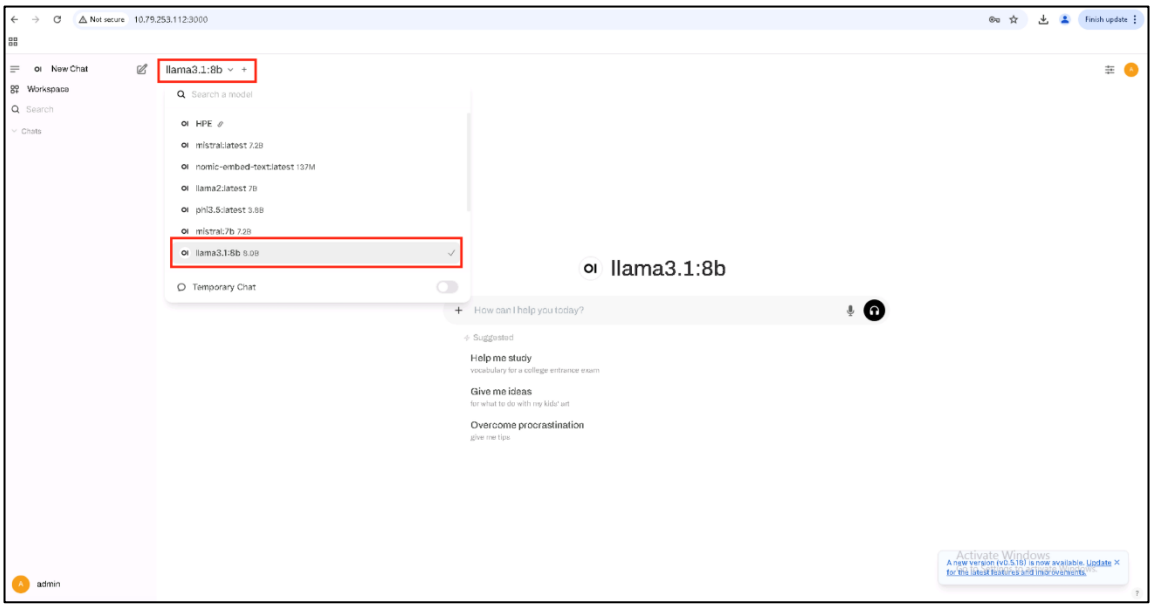


Figure 3: OpenWebUI interface with Llama3.1:8B model selected.

2. In the chat box, ask the following question: “which processor powers HPE Proliant Compute DL380a Gen12?”

3. Observe the response and note if it provides a generic answer.

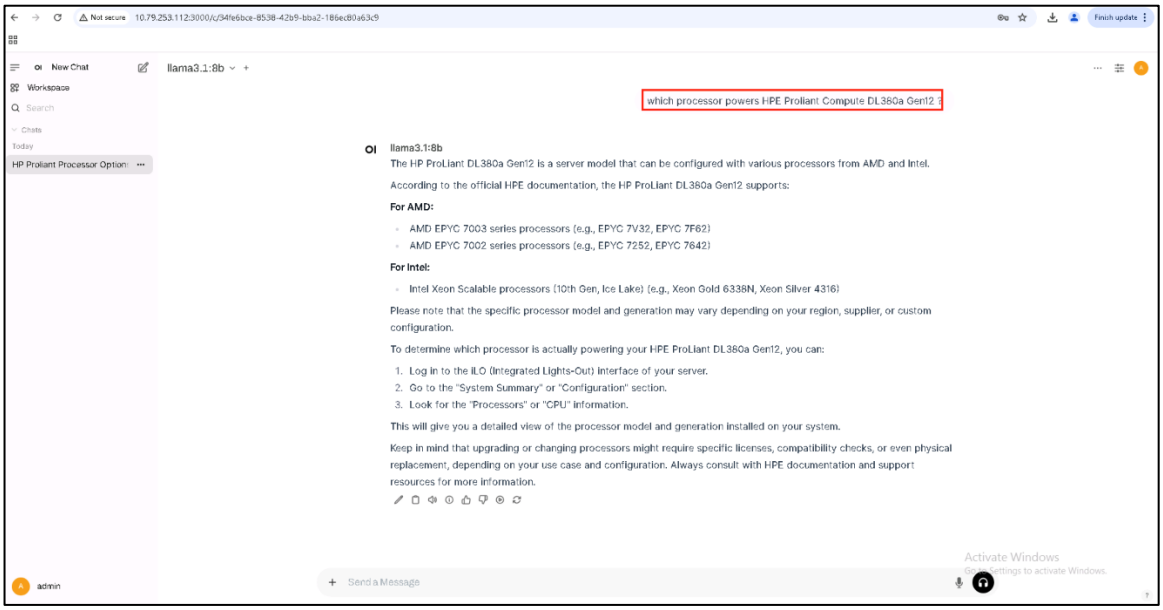


Figure 4: OpenWebUI displaying a response from the Llama3.1:8B model for a hardware-related query.

C. Query Using Custom HPE RAG Pipeline

1. Change the selected model to custom HPE RAG Pipeline from the dropdown menu.

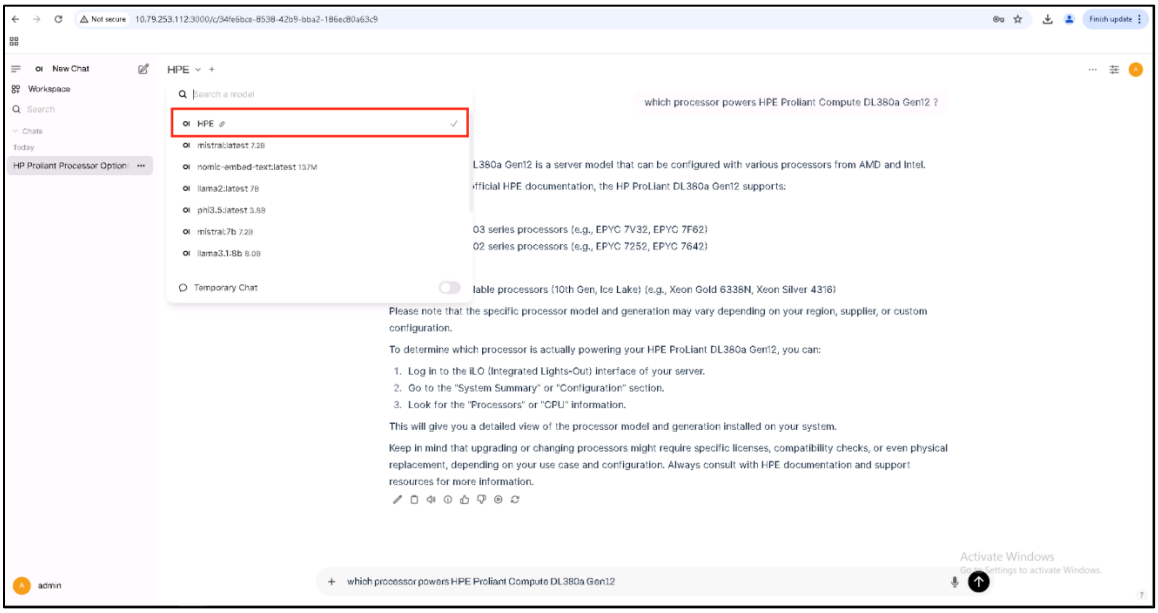


Figure 5: Switching the model in OpenWebUI to the HPE custom RAG Pipelinel.

2. Ask the same question: “which processor powers HPE Proliant Compute DL380a Gen12?”

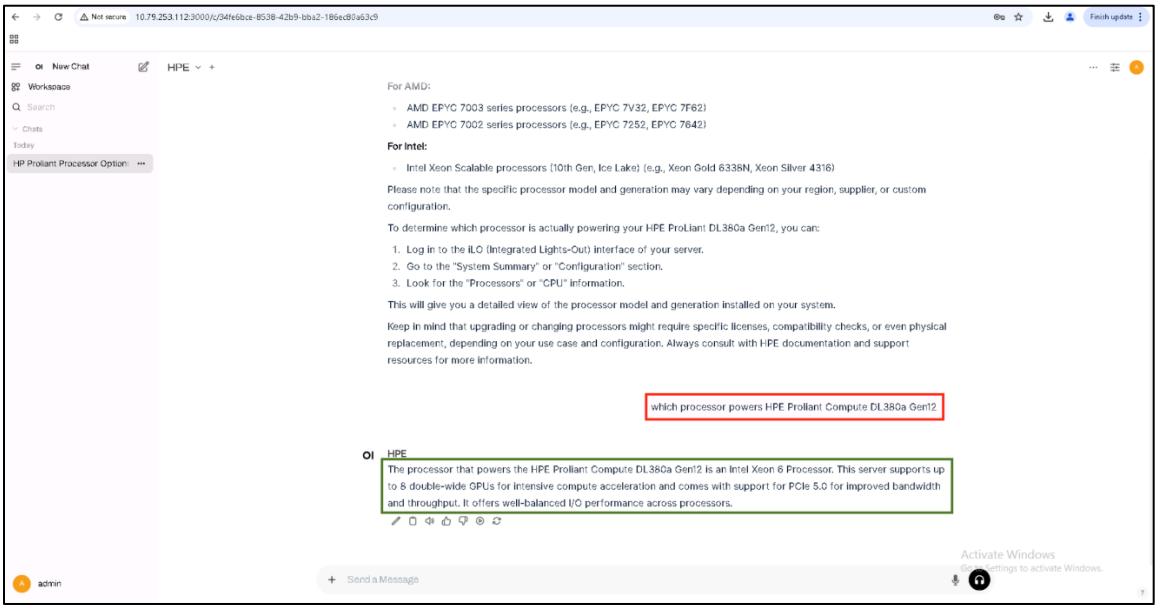


Figure 6: Response from the HPE RAG pipelinel providing a more precise and context-aware answer.

3. On Comparison, the response from the HPE RAG pipeline is more context-aware and directly answers the question, whereas the previous response from Llama3.1:8B was generic.

Task 2: Accessing and Using AI Essentials Virtual Assistant

The AI Essentials Virtual Assistant provides an enterprise-grade RAG-based interface for knowledge retrieval. This task will guide you through accessing the AI Essentials platform, launching the virtual assistant, and setting up an AI instance. By completing this task, you will gain familiarity with the assistant's interface and navigation options.

A. Launch AI Essentials Through HPE GreenLake

1. Start by accessing the HPE GreenLake dashboard. Under the Recent Services section, locate Private Cloud AI and click on the Launch button.

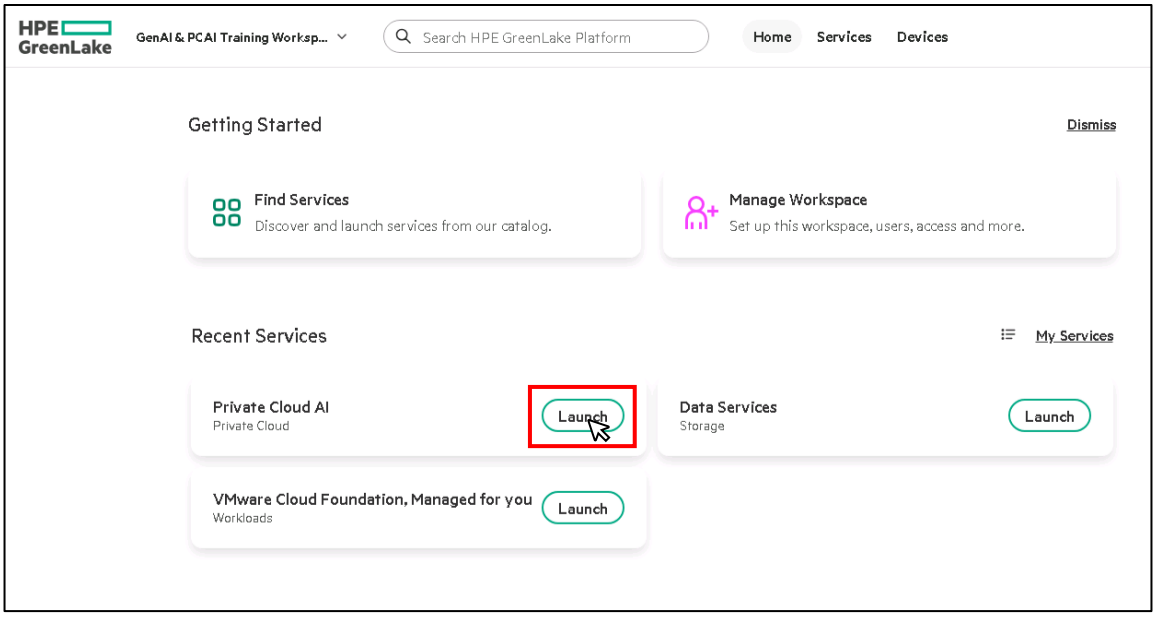


Figure 7: Accessing Private Cloud AI from the HPE GreenLake platform.

2. In the Private Cloud AI dashboard, select the Systems option to view the available nodes and resources.

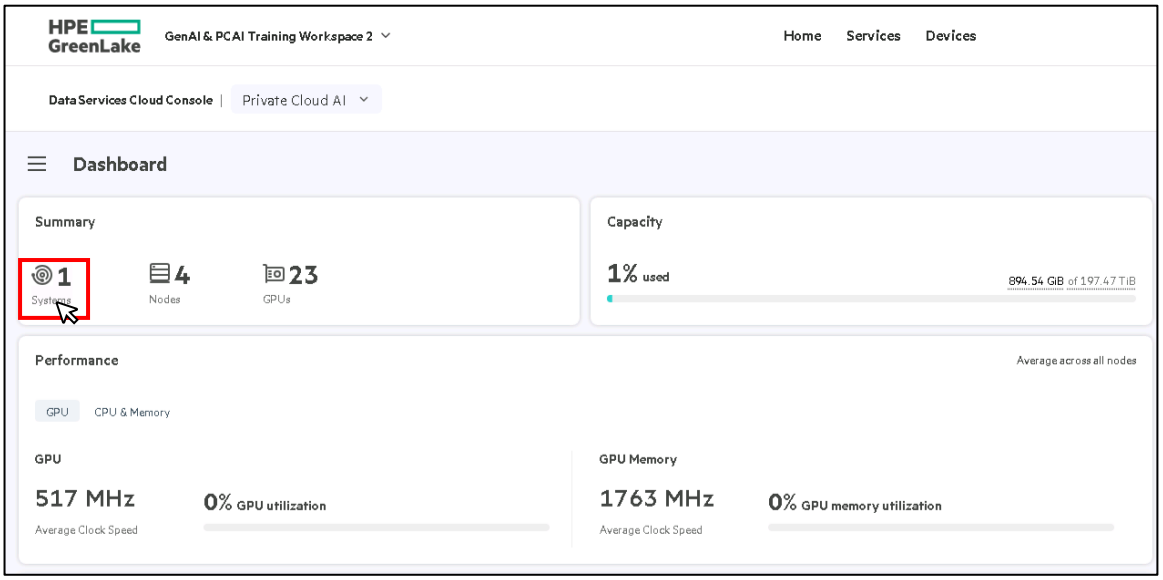


Figure 8: Private Cloud AI dashboard displaying system summary and GPU performance metrics.

3. In the Systems tab, identify the AI instance (e.g., PCAI2) and click on Launch to initiate the instance.

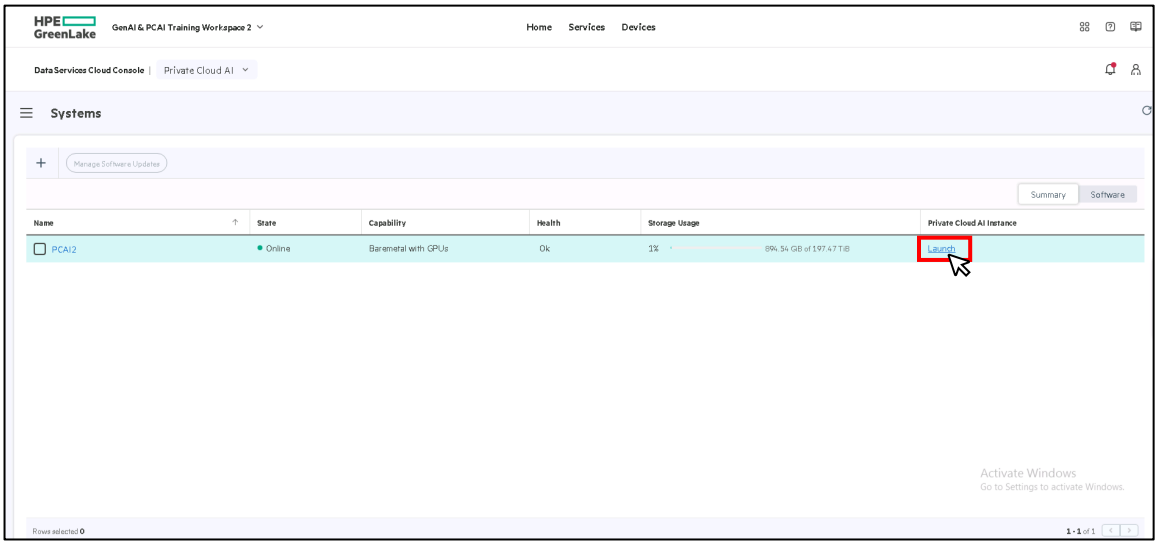


Figure 9: Private Cloud AI system list showing an active PCAI2 instance.

4. Click on the three-bar icon (hamburger menu) on the top-left corner to open the sidebar and view all available options.

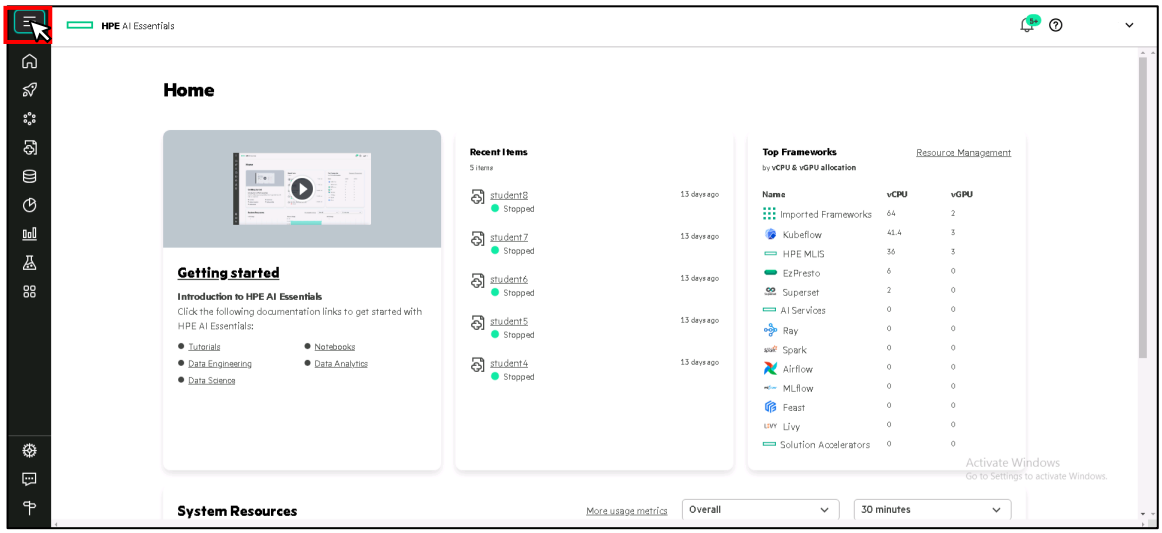


Figure 10: HPE AI Essentials home dashboard displaying system resources and recent activity.

B. Access Virtual Assistant

1. In the sidebar, navigate to the Deployed Application section to view the list of applications deployed.

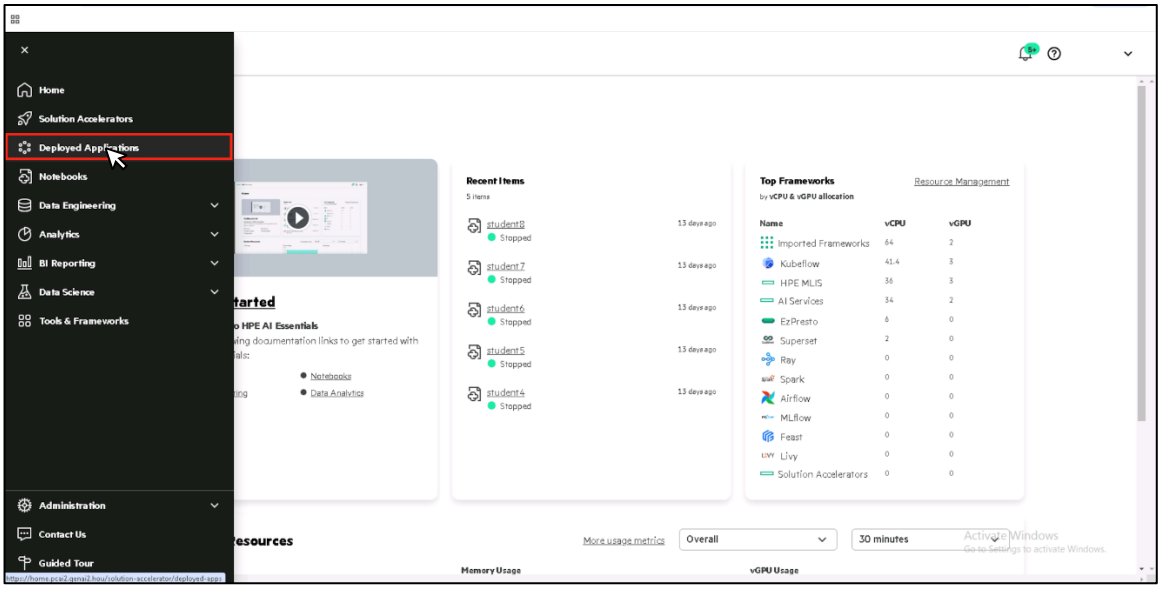


Figure 11: Navigating to Deployed Applications in HPE AI Essentials.

2. Find the Virtual Assistant tile and click Open.

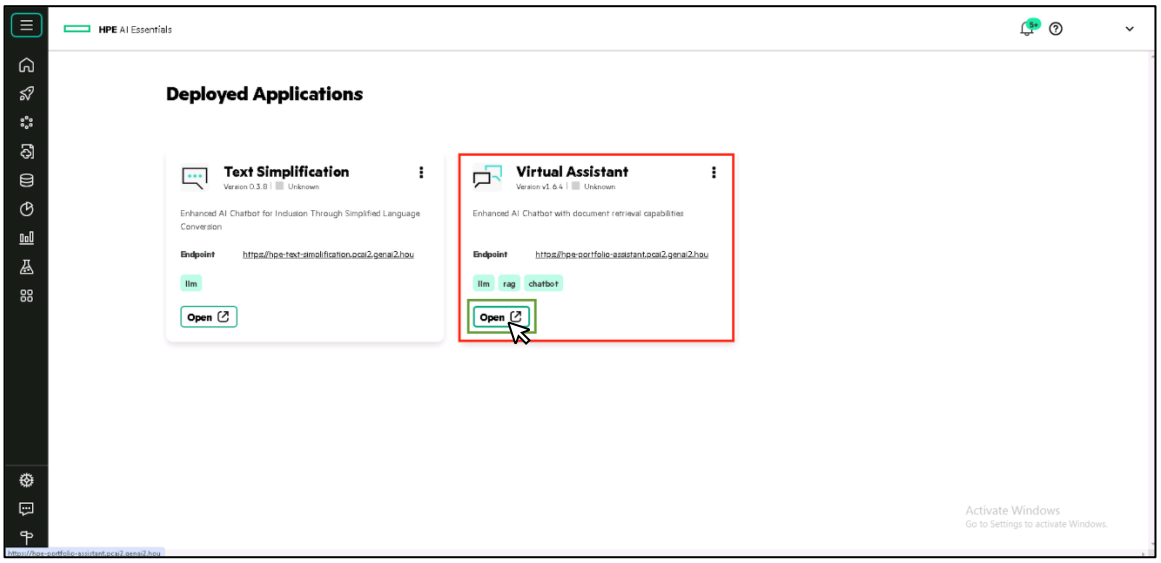


Figure 12: Launching the Virtual Assistant from Deployed Applications in HPE AI Essentials.

Task 3: Uploading a Document and Querying AI Essentials Virtual Assistant

AI Essentials Virtual Assistant allows users to upload documents to its private knowledge base for enhanced response accuracy. This task will guide you through the process of uploading a document, submitting queries, and analyzing how AI Essentials utilizes document retrieval for generating more precise responses.

A. **Upload a PDF to the Private Knowledge Base**

Upload a sample PDF document to AI Essentials' private knowledge base for AI-driven knowledge retrieval. The system will utilize the document's content to generate more precise responses.

1. In the Virtual Assistant interface, go to the Private Knowledge Base tab.

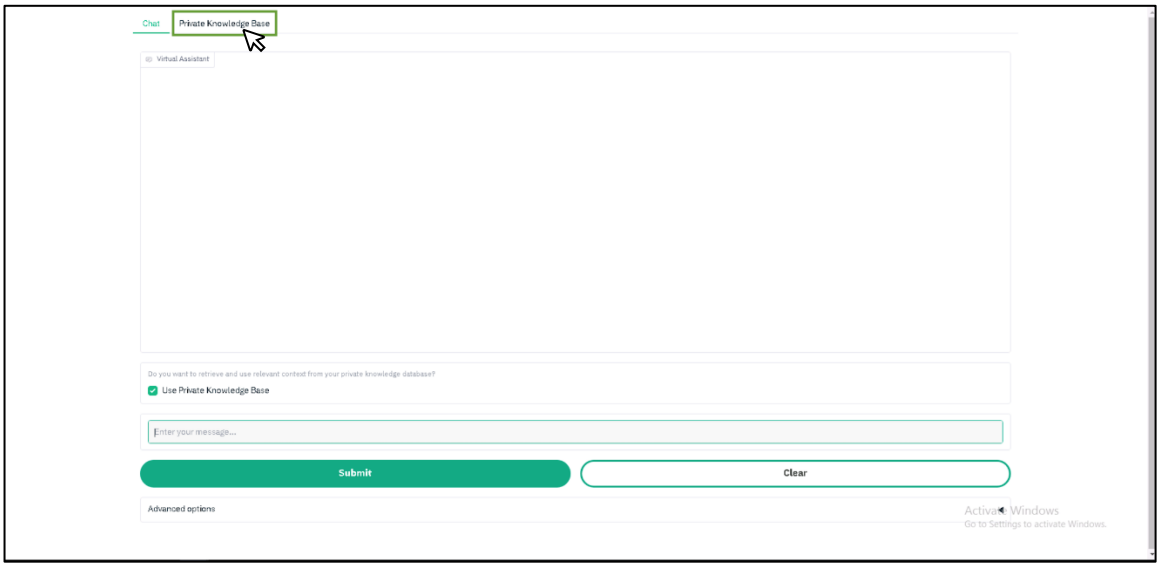


Figure 13: Virtual Assistant interface with Private Knowledge Base enabled for document retrieval.

2. Click on Drop file here.

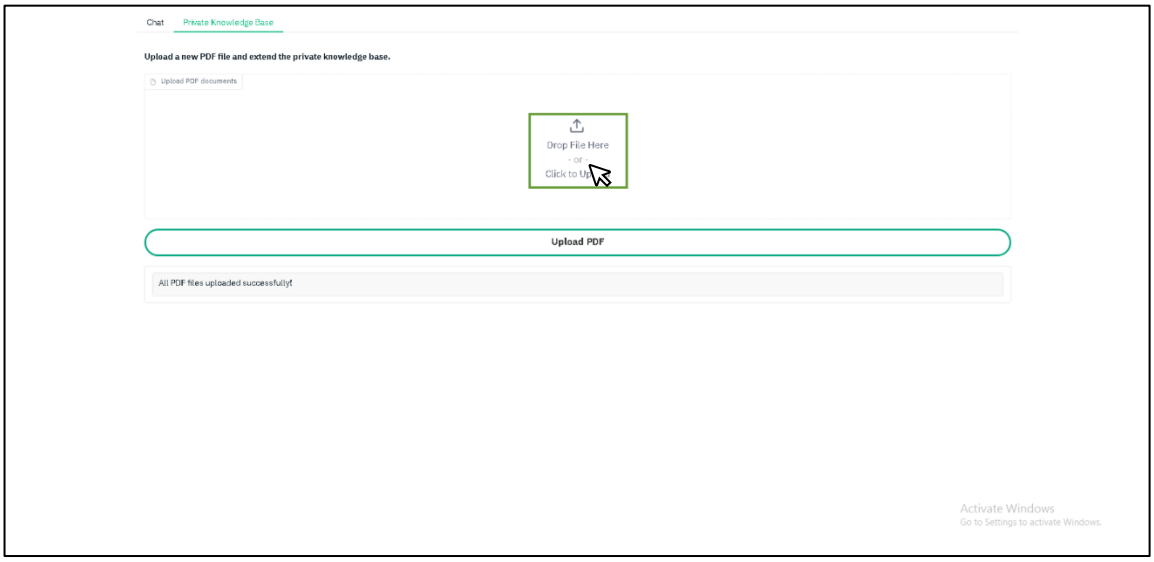


Figure 14: Uploading a PDF to the Private Knowledge Base in the Virtual Assistant.

3. Upload the same PDF you used in OpenWebUI.

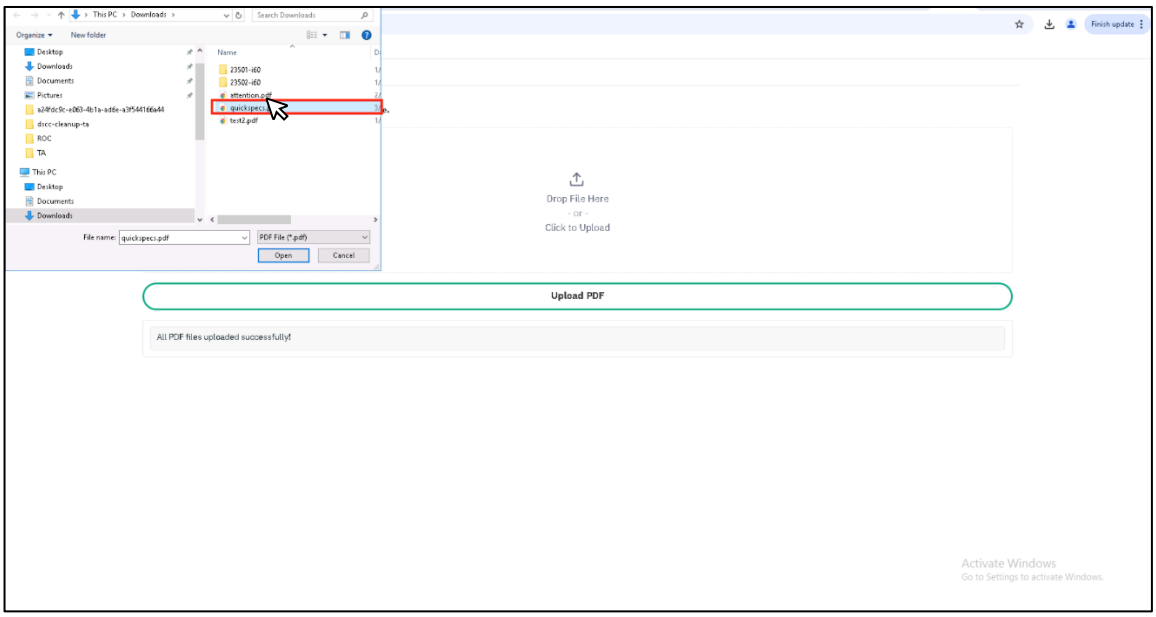


Figure 15: Selecting a PDF file for upload to the Private Knowledge Base in the Virtual Assistant.

B. Ask the Same Question in AI Essentials

Users will now submit the same query in AI Essentials Virtual Assistant that was used in OpenWebUI. This comparison will help evaluate how document retrieval affects response accuracy and content relevance.

1. Return to the Chat tab and ask the same question as before: “which processor powers HPE Proliant Compute DL380a Gen12?”.

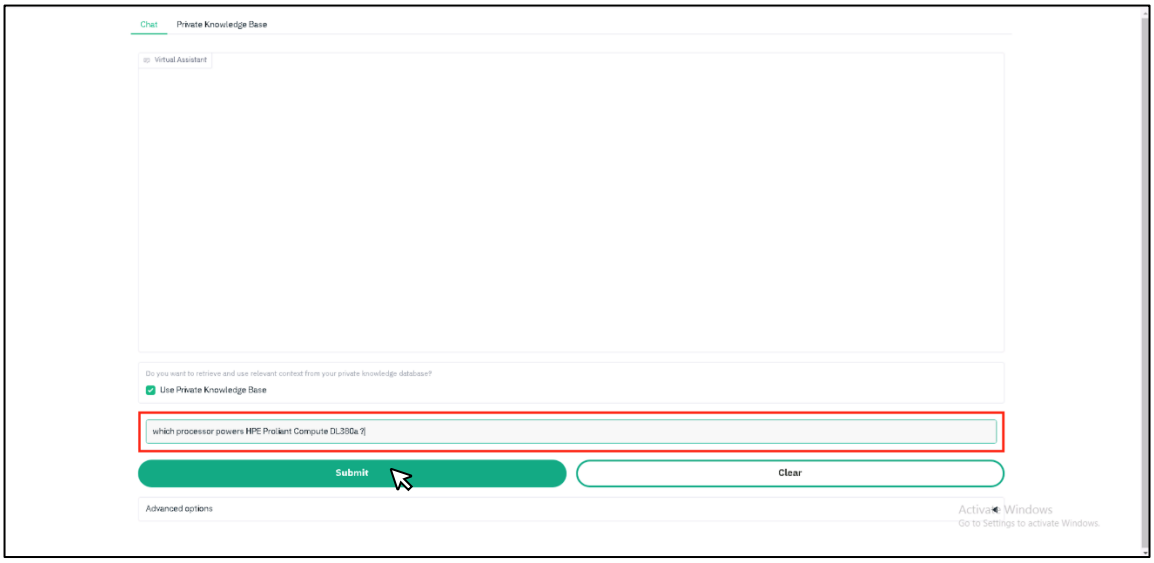


Figure 16: Querying the Virtual Assistant with document retrieval enabled.

2. On Comparison, the response is more context-aware and directly answers the question, making it more relevant and precise.

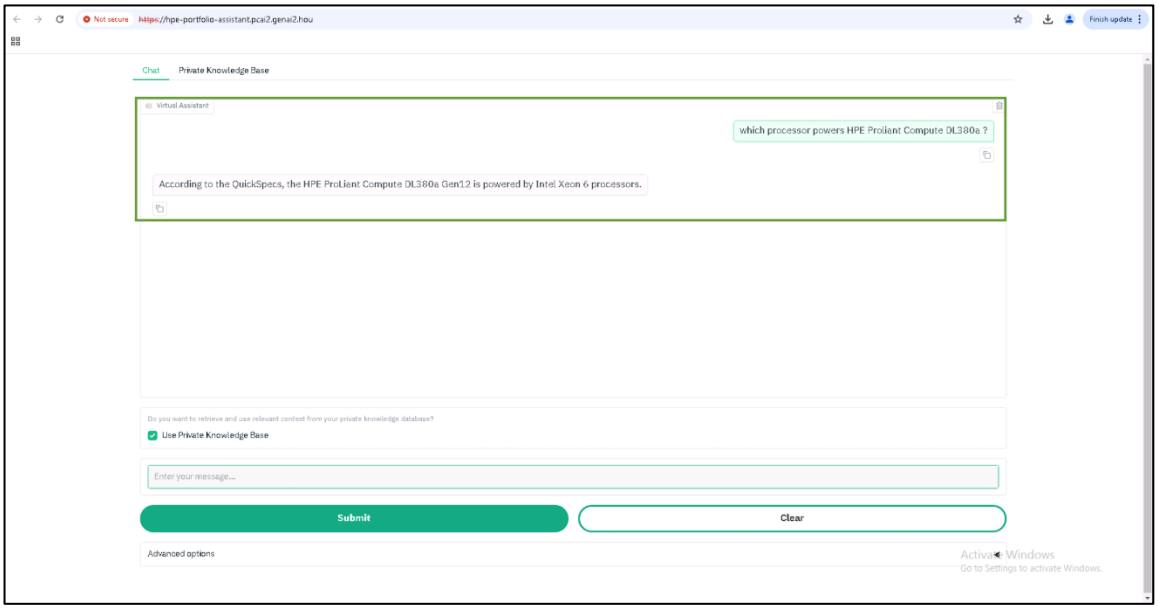
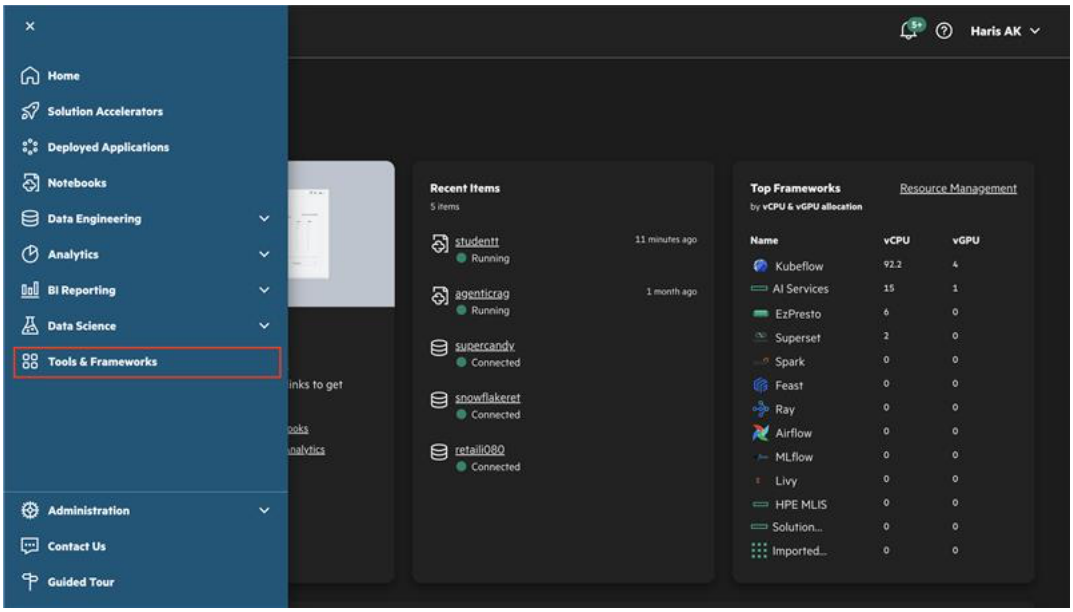


Figure 17: Virtual Assistant retrieving an accurate response using document-based knowledge.

Task 4: Uploading a Document and Querying AI Essentials Virtual Assistant

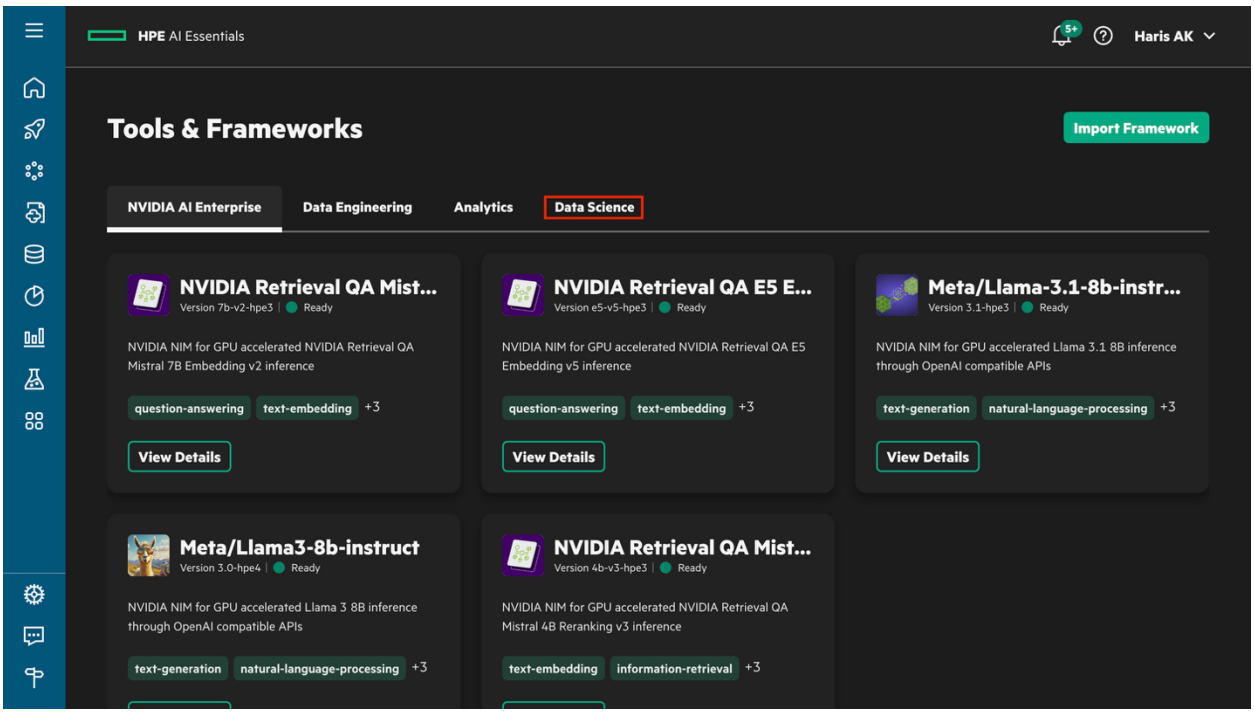
Step 1: Access "Tools & Frameworks"

From the left navigation menu, scroll down and click on Tools & Frameworks. This will open the main dashboard for various AI tools.



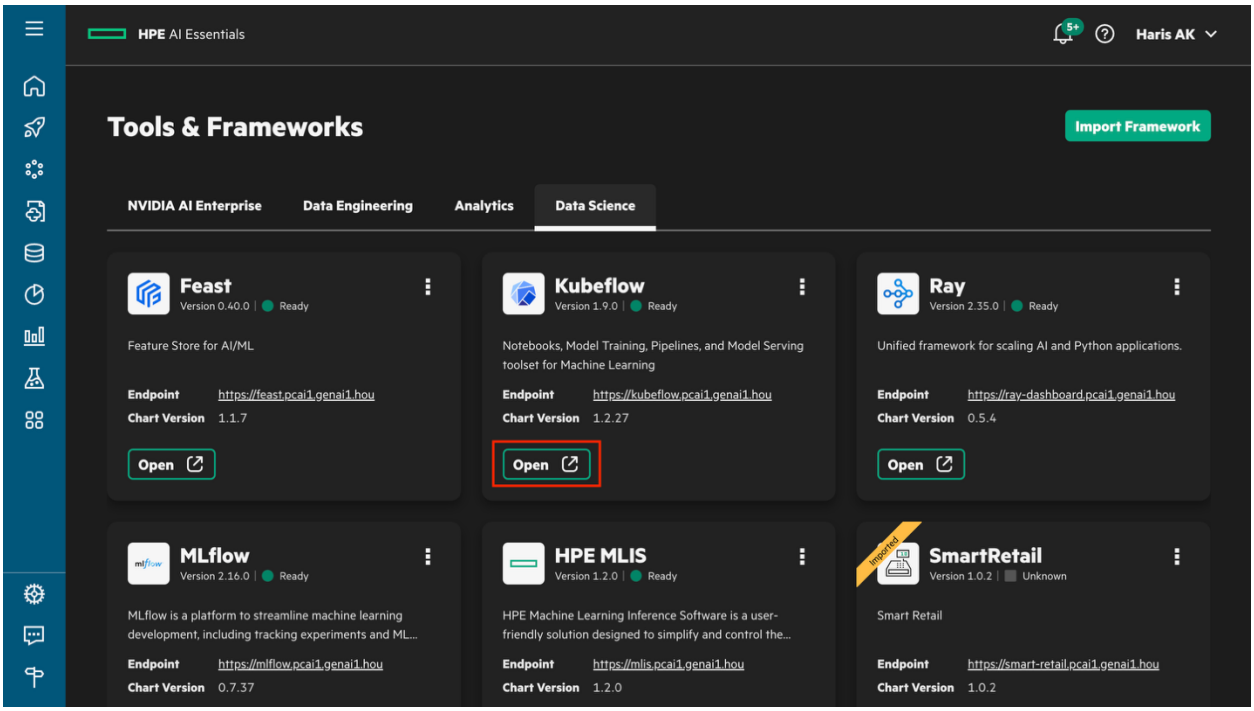
Step 2: Select "Data Science" Tab

Within Tools & Frameworks, click on the "Data Science" tab.



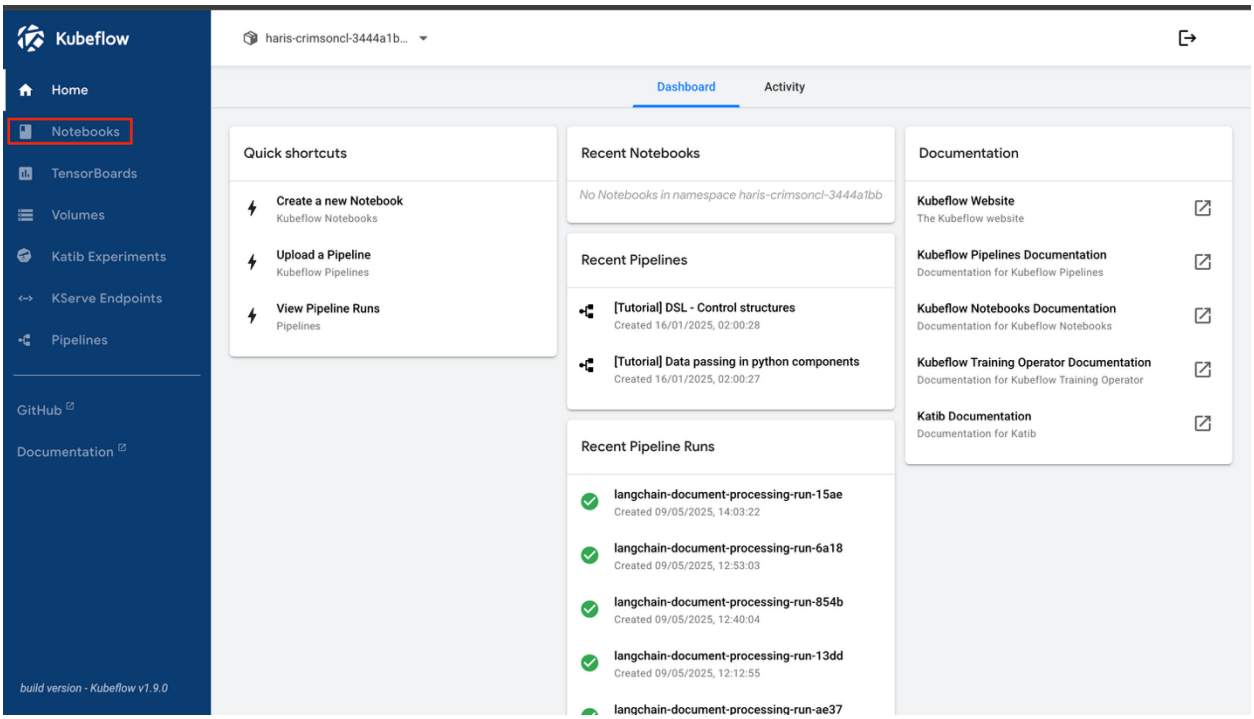
Step 3: Find Kubeflow and Click "Open"

Scroll through the list until you find Kubeflow. Click the "Open" button to launch the Kubeflow interface.



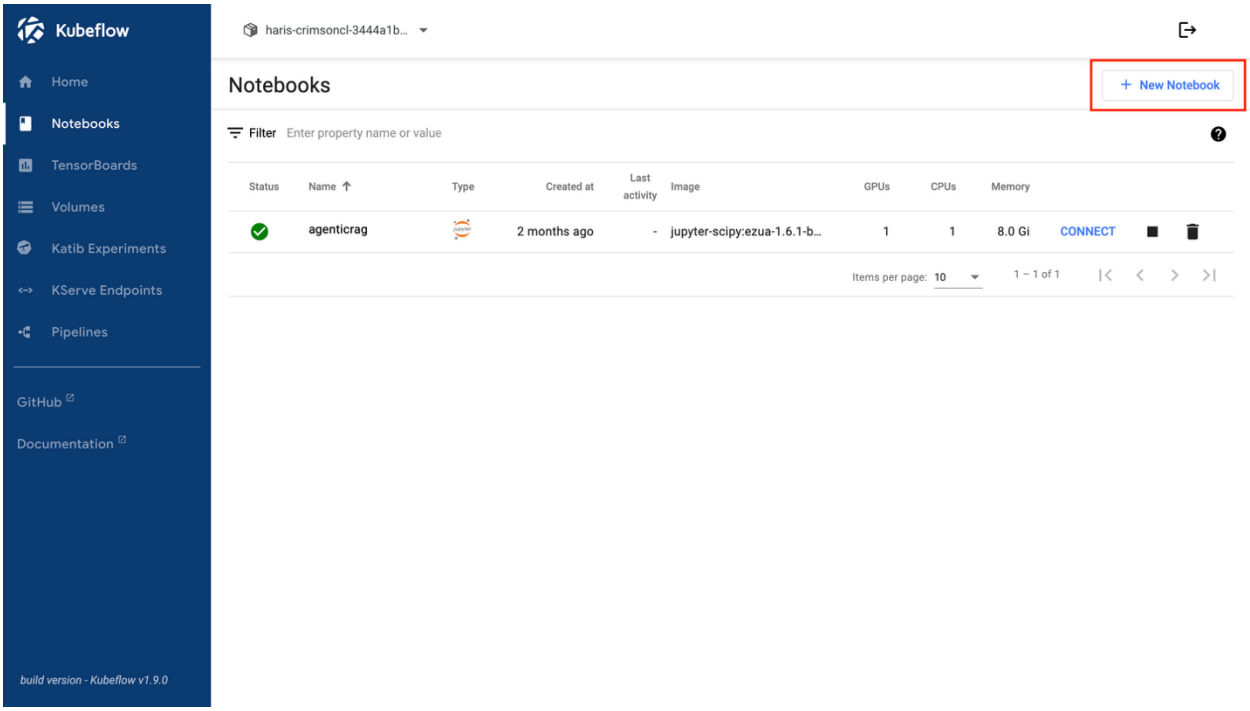
Step 4: Navigate to "Notebooks" Section

In the Kubeflow dashboard, click on Notebooks in the left sidebar. This section shows all existing notebooks and allows you to create new ones.



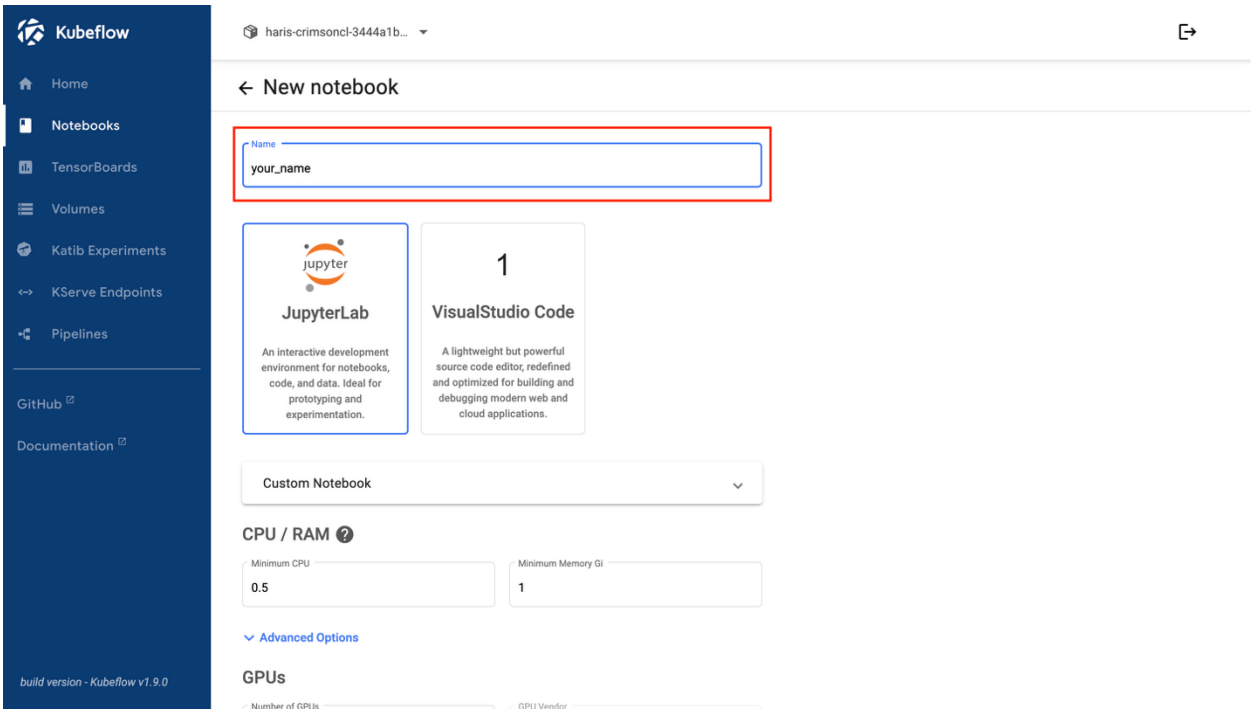
Step 5: Click "+ New Notebook"

On the Notebooks page, click the + New Notebook button in the top right corner.



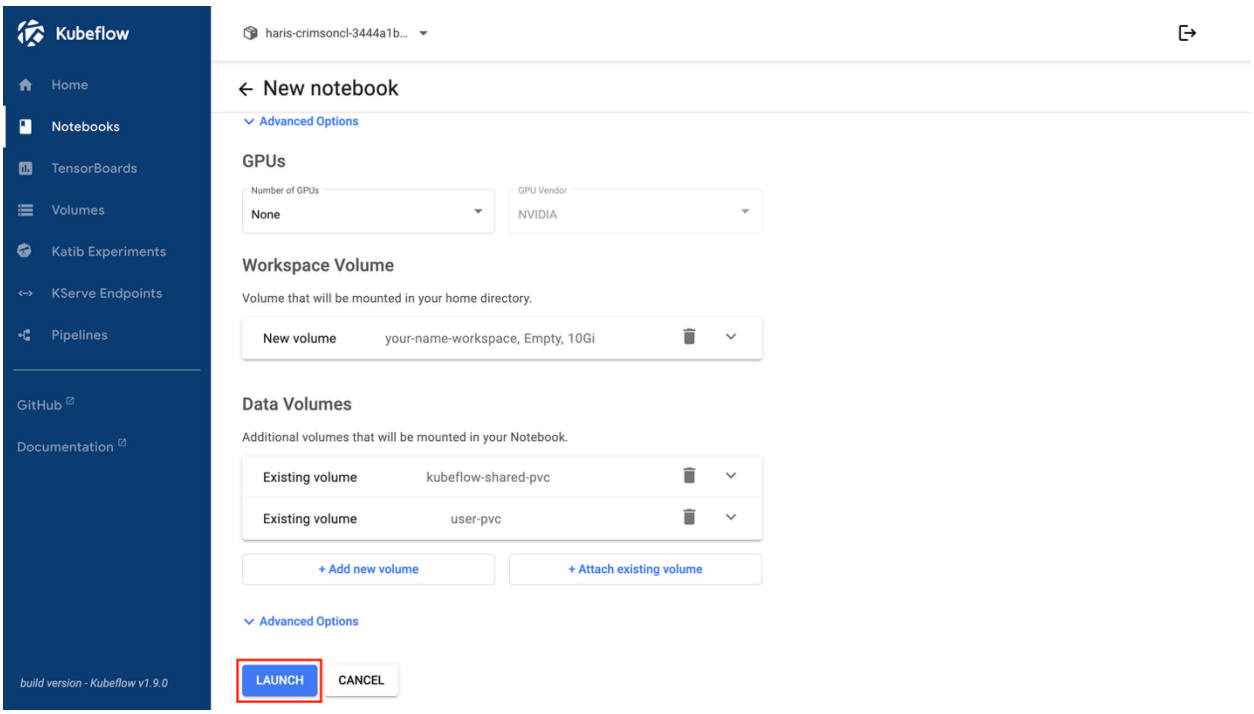
Step 6: Name Your Notebook

Enter a name for your new notebook (e.g., your_name). Select a development environment (e.g., JupyterLab or Visual Studio Code).



Step 7: Configure Resources

Set your CPU and memory requirements.

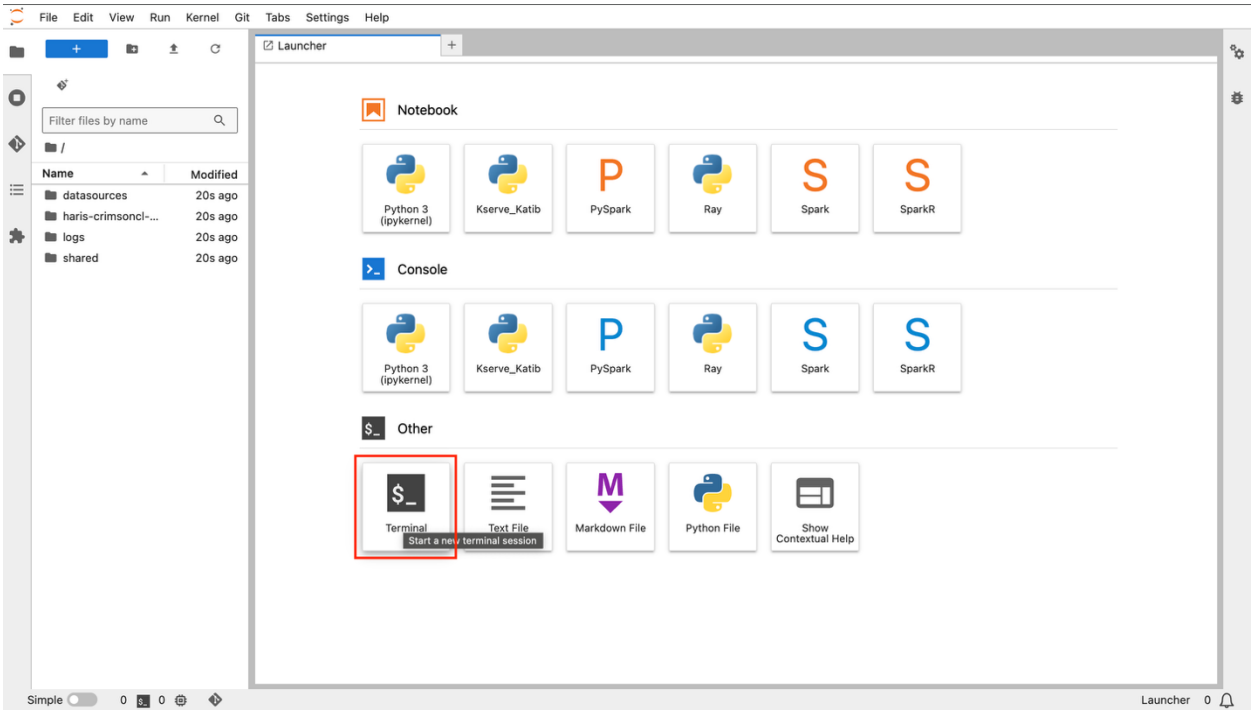


Step 8: Click "LAUNCH"

Scroll to the bottom and click LAUNCH to start the notebook environment. Kubeflow will create the pod and load your workspace.

Step 9: Open Terminal Inside JupyterLab

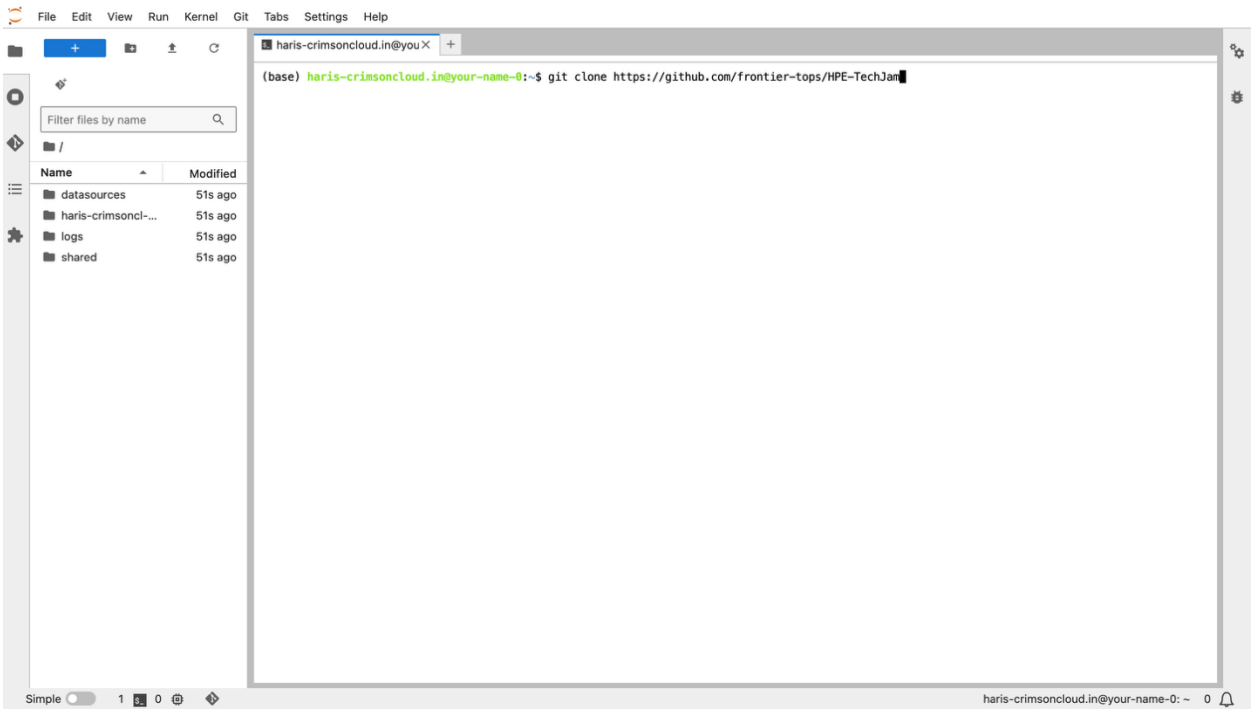
Once your notebook environment is up, launch a Terminal by clicking on the Terminal icon under the "Other" section in the launcher.



Step 10: Clone GitHub Repo & Install Requirements

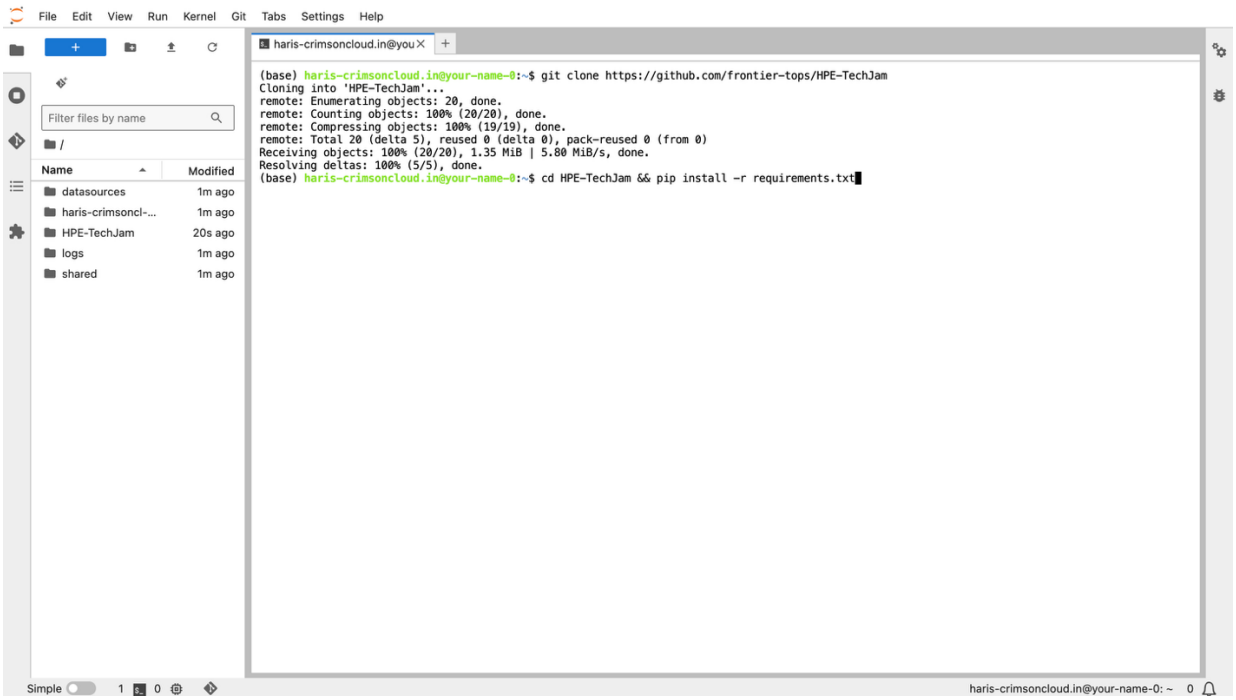
In the terminal, clone your GitHub repo:

git clone https://github.com/frontier-tops/HPE-TechJam

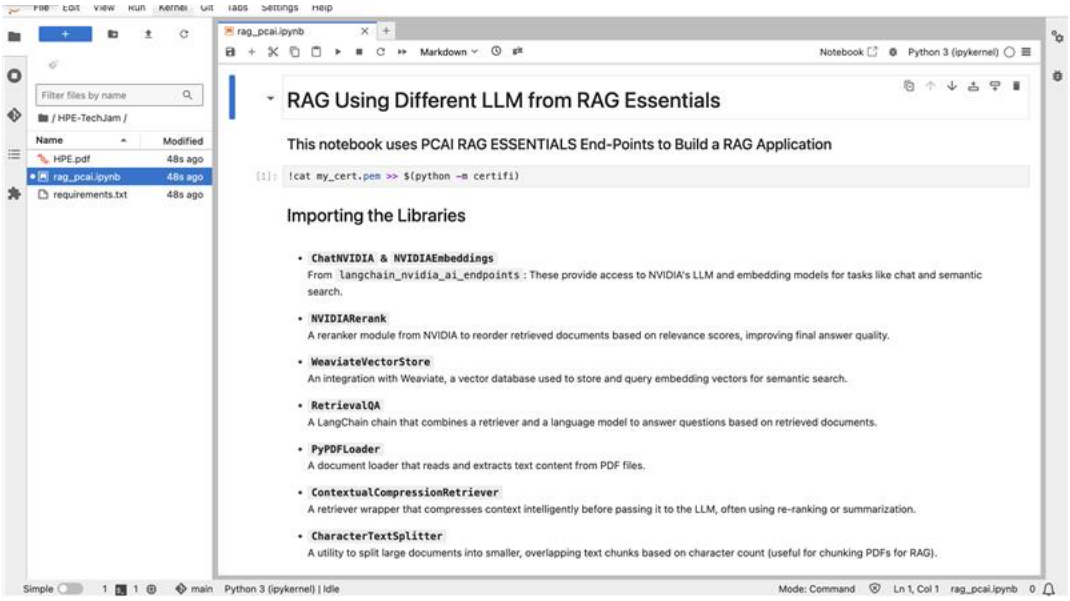


Then, move into the repo and install dependencies:

cd HPE-TechJam && pip install -r requirements.txt



Step 11: Open jupyter Notebook and execute



Task 5: Compare Responses from Both RAG Implementations

Evaluate and compare the responses generated by OpenWebUI and AI Essentials Virtual Assistant. Analyze key metrics such as accuracy, retrieval speed, and conciseness to determine the strengths and weaknesses of each system. Understanding these differences provides deeper insights into how different RAG implementations handle knowledge retrieval and response generation.

A. **Analyze Key Metrics**

Evaluate different parameters such as response accuracy, retrieval speed, and conciseness across both OpenWebUI and AI Essentials Virtual Assistant. Fill out a comparison table to document the findings.

Feature	**OpenWebUI (HPE RAG Pipeline)**	**AI Essentials Virtual Assistant**
Response Accuracy	Good	Very Good

Retrieval Speed	Good	Very Good
Conciseness	Good	Good
Document Support	Yes	Yes
Customizability	Very Good	Good
Interface Usability	Good	Very Good

Table 1: Comparative analysis of response accuracy, retrieval efficiency, and feature support across different RAG implementations.

Troubleshooting Guide

A. **Unable to Access the RAG Pipeline on Port 9099**

Possible Causes:

a. Pipeline containers may not be running, network access settings may be misconfigured, or firewall rules could be blocking the port.

Solution:

a. Confirm the container is running with docker ps. Restart it, if necessary, with docker start pipelines.

b. Ensure --add-host=host.docker.internal:host-gateway is included in the docker run command to enable container networking.

c. Check firewall rules and security settings to allow access on port 9099.

B. **Model Not Responding or Slow Performance**

Possible Causes:

a. Ollama container may not be running, GPU support may be misconfigured, or system resources may be overloaded.

Solution:

a. Verify the Ollama container is running on port 11434 by using docker ps. Start it if it's stopped with docker start ollama.

b. Ensure GPU support is correctly configured in Docker. Re-check Docker's access to the GPU if necessary.

c. Monitor system resource usage (CPU, memory, GPU) to ensure adequate capacity.

C. **Pipeline Errors When Processing Queries**

Possible Causes:

a. Missing library dependencies or issues within the hpe_pipe.py script.

Solution:

a. Check pipeline container logs with docker logs pipelines to identify specific errors.

b. Verify that required libraries (langchain, faiss-cpu, etc.) are installed in the container. Reinstall as needed.

c. Ensure hpe_pipe.py is correctly uploaded and saved in OpenWebUI under the pipeline settings.

Summary

The hands-on comparison highlights key differences in their retrieval-augmented generation capabilities. AI Essentials Virtual Assistant excels in response accuracy, retrieval speed, and interface usability, making it a strong choice for precise and efficient document-based queries. Both implementations offer document support, but OpenWebUI provides greater customizability as it provides custom pipeline integration, allowing more flexibility in tuning responses. While conciseness remains consistent across both systems, AI Essentials Virtual Assistant demonstrates better overall performance in handling knowledge-based queries with a more refined user experience.

Additional Resources

1. **OpenWebUI Documentation**: Detailed reference for setting up, configuring, and using OpenWebUI. **(Open WebUI Documentation)**
2. **HPE AI Essentials Documentation:** This document introduces HPE AI Essentials Software and summarizes key features in the general availability (GA) release. **(HPE AI Essentials Documentation)**