Optum

Instructions For Capstone

United Health Group

AI DOJO

Purpose of the Document

This document is your guide to successfully complete the capstone project. This capstone project is a graded assignment and is the final step towards your GenAI certification.

Kindly read the document carefully before you start your work on the capstone. Additionally, you can download the document and keep it for reference.

If you have any additional questions along the way, feel free to drop us an email at the following email IDs:

1. UAIS Queries
2. General Content/Capstone related queries for AI Dojo

Table of Contents

The Capstone: Clinical Intelligence System

Project Outcomes

In this assignment, you will apply key skills gained during the Generative AI Track, including:

- Apply Generative AI techniques to solve complex real-world challenges
- Build domain-specific AI applications for targeted business problems
- Implement and optimize Retrieval-Augmented Generation (RAG) architectures for applications where precision is critical

Project Overview

The Clinical Intelligence System assignment challenges you to build a medical question-answering platform using advanced RAG techniques.

This assignment marks the culmination of your applied generative AI training. You will focus on solving a critical problem: providing accurate and accessible medical information in a domain where precision and reliability are essential.

The completed system will demonstrate how generative AI can turn raw medical knowledge into actionable clinical insights in real-world healthcare settings.

The Challenge

Modern healthcare struggles with managing the rapid growth of medical information. Medical literature now doubles approximately every 73 days, making it difficult for healthcare professionals and patients to access reliable, up-to-date information.

Traditional keyword-based search tools are often inadequate for handling complex clinical queries that require contextual understanding.

The Assignment

Your goal is to create an intelligent system that interprets clinical questions, retrieves relevant information from trusted medical sources, and generates factually accurate, context-aware responses.

The intelligent system must be able to:

- Understand natural language medical queries with domain-specific meaning
- Retrieve and assess relevant content from verified medical knowledge bases
- Generate clear, accurate, and well-referenced answers based on the retrieved information

Additional Resources

The Project Submission Guide Video provides and overview of the capstone video, walks you through the process for submitting your capstone, and highlights common mistakes learners make.

The Top Submission Issues document outlines common mistakes learners make when submitting their capstone and provides guidance about how to avoid these mistakes.

If you would like to connect with peers who are also working through this capstone to ask questions or seek help with troubleshooting, you can join the Capstone 1 Teams channel.

Solution Requirements & Specifications

Building Your RAG System

Create a notebook named “**code.ipynb”** where you will build your RAG system. Your Jupyter Notebook (code.ipynb) must include the complete implementation of your Clinical Intelligence System using a Retrieval-Augmented Generation (RAG) architecture. This notebook will serve as the primary artifact for evaluation.

.

Components	Description
Dataset Loading and Preprocessing	• Load the dataset from *capstone1_rag_dataset.csv*. • Chunking is not necessary, as each document is appropriately sized.
Embedding and Vector Store Creation	• Use OpenAI’s text-embedding-3-small model to generate document embeddings. • Store the documents and embeddings in a ChromaDB vector database for retrieval.
Retrieval Strategy Exploration	• Build and experiment with your RAG workflow by exploring different retrieval strategies **(at least 2)** and prompt engineering techniques. • You must implement and explore at least two (or more) of the following retrieval approaches: ○ Semantic search ○ Semantic search with threshold filtering ○ Hybrid search (e.g., keyword + semantic) ○ Reranking based on relevance scores • **Note:** Do not remove any of the strategies explored. All approaches should be retained in the notebook for full transparency and proper evaluation.

Generation Pipeline Integration	• Connect the selected retriever with the generation component. (Here you can choose the best retriever strategy based on the previous exploration) • Use instruction-style prompt engineering that incorporates retrieved context accurately.
	• Generate answers using OpenAI's gpt-4.1-mini model.*
Validation Using the Evaluation Dataset	• See the following section: Evaluating Your RAG System
Testing and Generating Final Responses Using the Test Dataset	• See the following section: Testing Your RAG System

*If you started your capstone project using gpt-4o-mini you can continue using gpt-4o-mini through December 2025. After that time, the model will be deprecated.

Note:

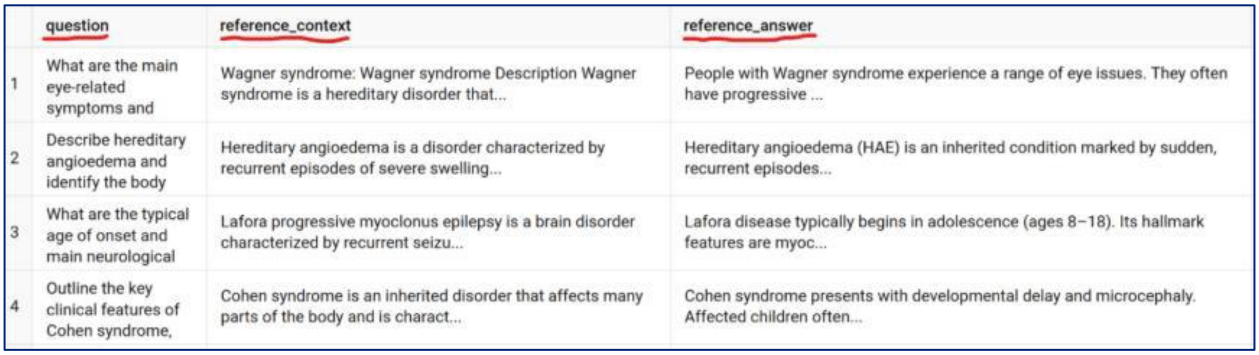
- Large Language Models (LLMs) can occasionally produce hallucinated responses. In this task, the 100 documents provided should be treated as verified and factual context sources. Your RAG system must rely exclusively on these documents stored in the vector database to answer questions. Do not use the LLM's internal training data, as the sources used during model training are unknown and cannot be verified.
- If a question is out of context or only partially answerable using the available documents, your system should handle it appropriately by relying only on the provided context and clearly stating any limitations in the response.
- Focus on generating accurate, context-aware responses and evaluating the effectiveness of your system.

This notebook should reflect your entire experimentation and development process for the capstone project. It should include:

- All explored retrieval strategies (e.g., semantic search, thresholding, hybrid, reranking).
- Clear identification and explanation of the best-performing workflow based on your choice.
- Proper code structure with clear comments explaining key steps of your best RAG workflow. Include detailed descriptions of each step in the workflow.
- Clearly structured using appropriate descriptive markdown headings and summaries to explain each section.

Evaluating Your RAG System

Use the evaluation dataset in the file *capstone1_rag_validation.csv*, which contains 10 medical questions along with reference context and reference answers.



	question	reference_context	reference_answer
1	What are the main eye-related symptoms and	Wagner syndrome: Wagner syndrome Description Wagner syndrome is a hereditary disorder that...	People with Wagner syndrome experience a range of eye issues. They often have progressive ...
2	Describe hereditary angioedema and identify the body	Hereditary angioedema is a disorder characterized by recurrent episodes of severe swelling...	Hereditary angioedema (HAE) is an inherited condition marked by sudden, recurrent episodes...
3	What are the typical age of onset and main neurological	Lafora progressive myoclonus epilepsy is a brain disorder characterized by recurrent seizu...	Lafora disease typically begins in adolescence (ages 8–18). Its hallmark features are myoc...
4	Outline the key clinical features of Cohen syndrome,	Cohen syndrome is an inherited disorder that affects many parts of the body and is charact...	Cohen syndrome presents with developmental delay and microcephaly. Affected children often...

Run these 10 questions through your RAG system to assess its performance.

Evaluate your system using the following standard metrics covered in the RAG course:

Metrics	Description

Retriever Performance Metrics:	**Precision and Recall:** Use custom defined metrics like precision@k or recall@k OR metrics from DeepEval to assess retrieval precision and recall.
Response Performance Metrics	☒ **Response Relevancy:** Assess the relevancy of generated responses with DeepEval. ☒ **Hallucination Check:** Assess if the generated responses suffer from any contradictions or fabricated information with DeepEval.

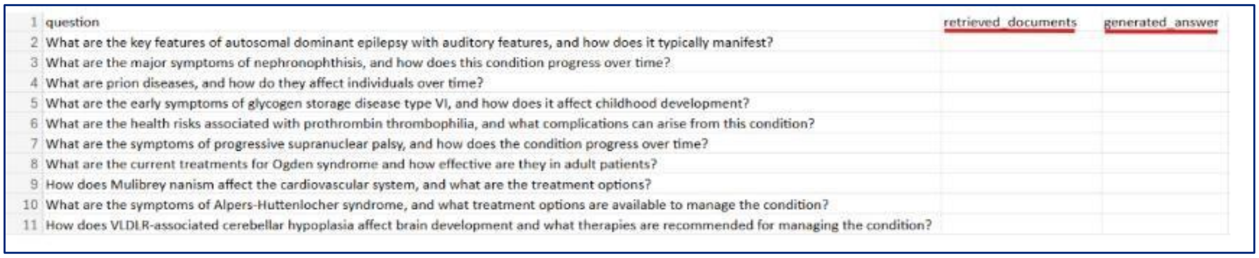
Refer to the Project Submission Guide Video mentioned earlier for hands-on examples.

These evaluations will help you understand how well your retriever and generator components are working. Use the insights to refine your retrieval strategy and prompt engineering before proceeding to the test dataset.

- Display the results in a DataFrame format in the notebook itself.
- These evaluations should help you refine your retrieval and prompt engineering strategies before running the test dataset.
- You do NOT have to save or submit validation results as a CSV file

Testing Your RAG System

Use the test dataset provided in the file *capstone1_rag_test_questions.csv*, which contains 10 unseen medical questions as shown below:

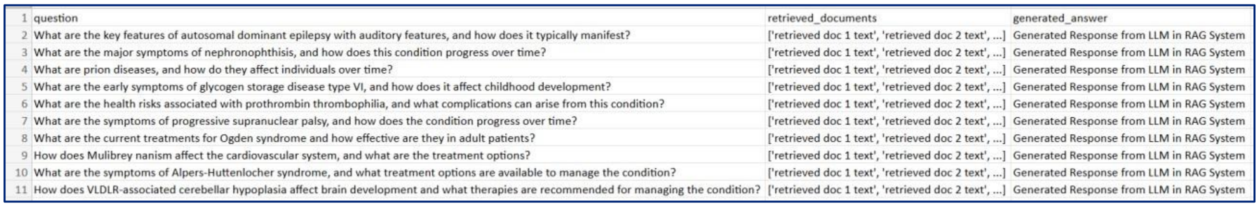


1	question	retrieved_documents	generated_answer
2	What are the key features of autosomal dominant epilepsy with auditory features, and how does it typically manifest?		
3	What are the major symptoms of nephronophthisis, and how does this condition progress over time?		
4	What are prion diseases, and how do they affect individuals over time?		
5	What are the early symptoms of glycogen storage disease type VI, and how does it affect childhood development?		
6	What are the health risks associated with prothrombin thrombophilia, and what complications can arise from this condition?		
7	What are the symptoms of progressive supranuclear palsy, and how does the condition progress over time?		
8	What are the current treatments for Ogden syndrome and how effective are they in adult patients?		
9	How does Mulibrey nanism affect the cardiovascular system, and what are the treatment options?		
10	What are the symptoms of Alpers-Huttenlocher syndrome, and what treatment options are available to manage the condition?		
11	How does VLDLR-associated cerebellar hypoplasia affect brain development and what therapies are recommended for managing the condition?		

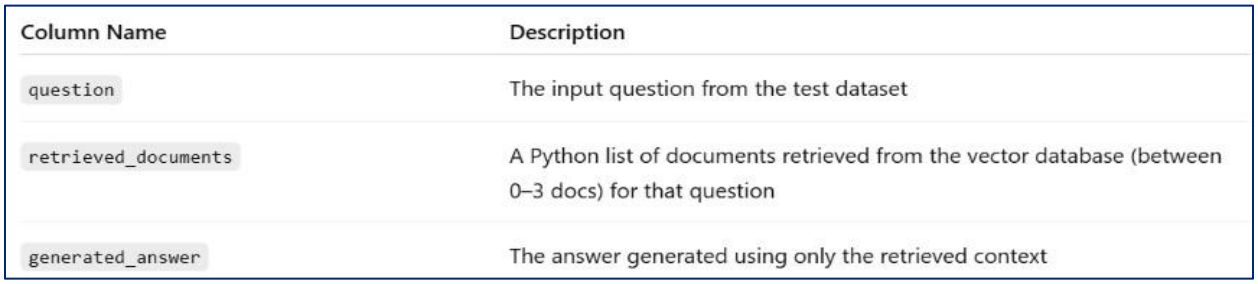
1. Run these questions through your RAG system.
2. For each question, retrieve the Top-K (K ≤ 3) relevant context documents and generate a response.
3. Record the results in the following format:
   - **retrieved_documents:** A python list containing the Top-K (K can be between 0-3 documents based on your retrieval strategy) context documents retrieved for each question.
     - Retrieved documents should be put in as a python list of strings (one string for each document)
     - If using LangChain Document objects, do not put them directly, extract their page_content and put the strings into a python list
     - (Do not combine all retrieved documents and upload it as a single context document in this column)
     - You may default to K = 3, however, you are encouraged to experiment with techniques like similarity thresholding, hybrid search, or reranking.
     - Your retrieval strategy may return fewer than 3 documents if that improves relevance.
     - You may use fewer than three documents if your retrieval strategy (e.g., reranking, hybrid search or thresholding) deems fewer to be relevant. If no relevant documents are found, return an empty list (i.e., []) and ensure the generated answer reflects this limitation.
   - **generated_answer:** The final response generated by your system for the given question.
     - Ensure the answer is grounded in the retrieved documents
     - If no relevant documents were retrieved, the generated answer should reflect that limitation clearly (e.g., "The question cannot be answered using the available documents.")
4. Store the outputs in a CSV file named submission.csv. Make sure the results in this file reflect your best-performing retrieval and generation pipeline. This file will be used for final evaluation and must follow the format shown in the snapshot above. It must include the following columns:

□ **question -** The input question from the test dataset □ **retrieved_documents**

□ **generated_answer**



1	question	retrieved_documents	generated_answer
2	What are the key features of autosomal dominant epilepsy with auditory features, and how does it typically manifest?	['retrieved doc 1 text', 'retrieved doc 2 text', ...]	Generated Response from LLM in RAG System
3	What are the major symptoms of nephronophthisis, and how does this condition progress over time?	['retrieved doc 1 text', 'retrieved doc 2 text', ...]	Generated Response from LLM in RAG System
4	What are prion diseases, and how do they affect individuals over time?	['retrieved doc 1 text', 'retrieved doc 2 text', ...]	Generated Response from LLM in RAG System
5	What are the early symptoms of glycogen storage disease type VI, and how does it affect childhood development?	['retrieved doc 1 text', 'retrieved doc 2 text', ...]	Generated Response from LLM in RAG System
6	What are the health risks associated with prothrombin thrombophilia, and what complications can arise from this condition?	['retrieved doc 1 text', 'retrieved doc 2 text', ...]	Generated Response from LLM in RAG System
7	What are the symptoms of progressive supranuclear palsy, and how does the condition progress over time?	['retrieved doc 1 text', 'retrieved doc 2 text', ...]	Generated Response from LLM in RAG System
8	What are the current treatments for Ogden syndrome and how effective are they in adult patients?	['retrieved doc 1 text', 'retrieved doc 2 text', ...]	Generated Response from LLM in RAG System
9	How does Mulibrey nanism affect the cardiovascular system, and what are the treatment options?	['retrieved doc 1 text', 'retrieved doc 2 text', ...]	Generated Response from LLM in RAG System
10	What are the symptoms of Alpers-Huttenlocher syndrome, and what treatment options are available to manage the condition?	['retrieved doc 1 text', 'retrieved doc 2 text', ...]	Generated Response from LLM in RAG System
11	How does VLDLR-associated cerebellar hypoplasia affect brain development and what therapies are recommended for managing the condition?	['retrieved doc 1 text', 'retrieved doc 2 text', ...]	Generated Response from LLM in RAG System



Column Name	Description
`question`	The input question from the test dataset
`retrieved_documents`	A Python list of documents retrieved from the vector database (between 0–3 docs) for that question
`generated_answer`	The answer generated using only the retrieved context

Grading and Scoring

The submission will be evaluated based on the following aspects:

- Performance metrics measuring retrieval and response quality in your RAG system
  - □ Ability to handle questions that are unanswerable or only partially answerable
  - □ Ensuring that the 100 context documents are treated as the sole source of truth and used exclusively to answer questions (the system should not use internal LLM knowledge)
- Code Quality
- Completeness of the RAG workflow, including all explored strategies and the final selected approach in the notebook

Once you submit your capstone, it will be evaluated thoroughly, and your evaluation report will take approximately 14 business days to complete. The grades would be passed to your LMS, and you would receive a detailed report regarding your submission. If you do not receive your grade after 4 weeks of your submission, you can send us an email or drop in a comment in the AIML community.

Grading Rubrics

The following 3-point Rubrics (with a maximum scoring opportunity of 12), would be used for scoring your submission:

Component	**Definition**	**Scoring Guidelines**	**Needs Improvement (1 point)**	**Satisfactory (2 points)**	**Excellent (3 points)**
RAG Performance – Response Quality	Assesses the quality of the responses generated by the RAG system for all 10 questions in the test dataset.	Response quality will be evaluated based on two key performance metrics: Response Relevance and Hallucination Score.	Most responses are irrelevant to the user's queries and human reference answers and contain significant hallucinations.	The majority (>50% but <80%) of responses are relevant to the user queries and human reference answers, with minimal to no hallucinations.	Almost all (>=80%) responses are relevant to the user queries and human reference answers and are free from hallucinations.

RAG Performance – Retrieval Quality	Evaluates the effectiveness of the retrieval component in identifying the most relevant context chunks for answering a given query, based on the topK retrieved chunks (k<=3) for all 10 questions in the test dataset. Remember the number of chunks should be <= 3 where you can filter out irrelevant chunks if needed using various strategies like similarity with threshold, reranking etc.	Retrieval quality will be assessed using two standard metrics: Precision@k and Recall@k for k<=3.	Low Precision@k and Recall@k scores. Most of the retrieved chunks are either irrelevant or fail to capture the key information required to answer the query accurately.	Moderate Precision@k and Recall@k scores. At least half of the queries (>50% but <80%) retrieve relevant chunks that partially or fully support the correct answer.	High Precision@k and Recall@k scores. Almost all queries (>=80%) retrieve highly relevant chunks that comprehensively support the correct answer.
Solution Code Quality	Evaluates the overall quality of the code implementing the solution notebook, focusing on readability, modularity, correctness, and documentation.		Code is difficult to read or understand, lacks proper structure, and contains minimal or no documentation and comments. Error handling is inadequate or absent.	Code is mostly modular, with some level of documentation and comments. Structure and readability are acceptable, though there may be minor issues.	Code is clean, well-structured, and modular. It includes comprehensive documentation, comments, clear variable and function naming, and adheres to best practices for readability,

					modularity, and error handling.

RAG Workflow Design & Implementation	Evaluates the overall architecture and implementation of the RAG pipeline - from data ingestion and indexing to retrieval, generation, and response evaluation.		RAG workflow is incomplete or contains critical issues. Key components such as indexing, semantic retrieval, or integration between retriever and generator are either missing or incorrectly implemented.	A basic RAG pipeline is correctly implemented with end-to-end functionality. The learner has incorporated semantic search in the retrieval process but has not explored advanced strategies such as hybrid search (combining keyword and semantic), reranking.	The RAG workflow is thoughtfully architected and fully functional. The learner has explored at least two or more retrieval strategies (e.g., hybrid search, reranking, filterbased retrieval) and incorporated fallback mechanisms - such as noanswer prompts to reduce hallucinations and handle outof-context queries. Prompt engineering has been applied to enhance generation quality.

Grading Outcomes

Total Score	Grade	Pass/Fail	Description
11–12 points	Excellent	Pass	Outstanding performance across all areas. Demonstrates a strong understanding of RAG concepts and implementation.

8–10 points	Satisfactory	Pass	Good overall performance with a solid grasp of key concepts, though there is room for improvement in some areas.
4–7 points	Needs Improvement	Fail	Basic or inconsistent performance. Key areas of the RAG system require further development and refinement.

Submission

Your final submission will include the following:

1. **code.ipynb** - Final Jupyter Notebook
2. **submission.csv** - Test Dataset Responses

We recommend you review the submission checklist on the next page before submitting your files.

Resubmission

Your first submission of the Generative AI capstone is free, and you get one free resubmission if you don't pass the first time. If you fail the capstone twice, your third (and every subsequent) submission will incur a $100 charge to your department's GL. This is to encourage you to do your very best work the first time you submit!

Submission Checklist

Sl No.	Description	Status
1	The Jupyter notebook is saved as **code.ipynb**	☐ Checked

2	The Jupyter notebook has all the major sections including: • Dataset loading and pre-processing • Vector store creation • Retrieval strategy exploration • Generation pipeline integration • Validating RAG pipeline on evaluation dataset • Predictions on test dataset.	☐ Checked
3	The best performing workflow of your choice is clearly identified and explained in your Jupyter Notebook	☐ Checked
4	You have explored at least two or more retrieval strategies in your Jupyter notebook	☐ Checked
5	The notebook has clear code comments explaining the key steps of the RAG workflow.	☐ Checked
6	The notebook has a clear structure using appropriate descriptive markdown heading and text descriptions to explain each section.	☐ Checked
7	The notebook has been run against the test dataset “capstone1_rag_test_questions.csv.”	☐ Checked
8	The responses on the test questions generated, are saved in the correct format in a CSV file, as **submission.csv**.	☐ Checked
9	The responses on all 10 test questions have been stored in the following three columns (per question) in **submission.csv**: • **question** – The original question • **retrieved_documents** - the TopK retrieved documents for that question based on your retrieval strategy have been put in the correct format (a	☐ Checked

	python list of strings – one string for each document) • **generated_answer** – RAG response for the question	
10	The **submission.csv file** is correctly formatted with the correct column names, as mentioned above.	☐ Checked