Assignment Name: Clinical Intelligence System

Outcomes

In this assignment, you will **apply the following skills** developed during the completion of the Generative AI Track:

- Various Generative AI techniques to solve complex real-world problems
- Domain-specific AI applications to solve specific business problems
- Implement & optimize RAG architectures for precision-critical applications

Overview of the Assignment

The "Clinical Intelligence System" assignment requires you to build a sophisticated medical question-answering platform leveraging state-of-the-art Retrieval-Augmented Generation (RAG) techniques.

This assignment represents the culmination of applied generative AI learning, requiring learners to address the critical challenge of accessible, accurate medical information retrieval in a domain where precision and reliability are paramount.

For this assignment, you will develop an intelligent system capable of interpreting clinical queries, retrieving relevant information from trusted medical knowledge bases, and generating contextually appropriate, factually accurate responses.

The final working system will bridge the gap between raw medical data and actionable clinical insights, demonstrating practical applications of generative AI in healthcare settings.

The Healthcare Information Challenge Assignment

Modern healthcare faces a significant information management crisis. The volume of medical literature doubles approximately every 73 days, making it increasingly difficult for healthcare professionals and patients to access relevant information efficiently. Traditional search methods are inadequate for complex clinical queries that require nuanced understanding and contextual interpretation.

For this assignment you are tasked with developing an intelligent system that can:

- Understand natural language medical queries with domain-specific context
- Access and evaluate relevant information from trusted medical sources
- Generate concise, contextually accurate responses with appropriate references

Solution Requirements & Specifications

The Core System Architecture

The Clinical Intelligence System must implement a complete RAG architecture with the following key components.

Sl No.	Key Component	Details of the component/s
1	**Document Processing Engine**	**Loading Mechanism**: Proper document loading methods to load the data from the dataset file (chunking is not necessary given the document sizes)
2	**Retrieval Engine**	● **Semantic Representation**: Standard OpenAI's `text-embedding-3-small` embedder for embedding documents ● **Vector Database**: Storing documents and embeddings in a vector database for fast retrieval using **ChromaDB** ● **Retrieval Strategies:** Implementing simple semantic search retrieval and then exploring more advanced strategies, like semantic search with thresholding, hybrid search, reranking
3	**Response Generator Engine**	● **Context-Integrated Prompting**: Proper instruction prompts to instruct LLMs to do contextual RAG based response generation ● **LLM**: Standard OpenAI's `gpt-4o-mini` for generating responses ● **Source Attribution**: Show retrieved information sources which were used to generate responses ● **Uncertainty Handling**: Clear communication: Let the user know if a question cannot be answered due to missing context etc., rather than giving incorrect or made-up responses.

Technical Stack Requirements

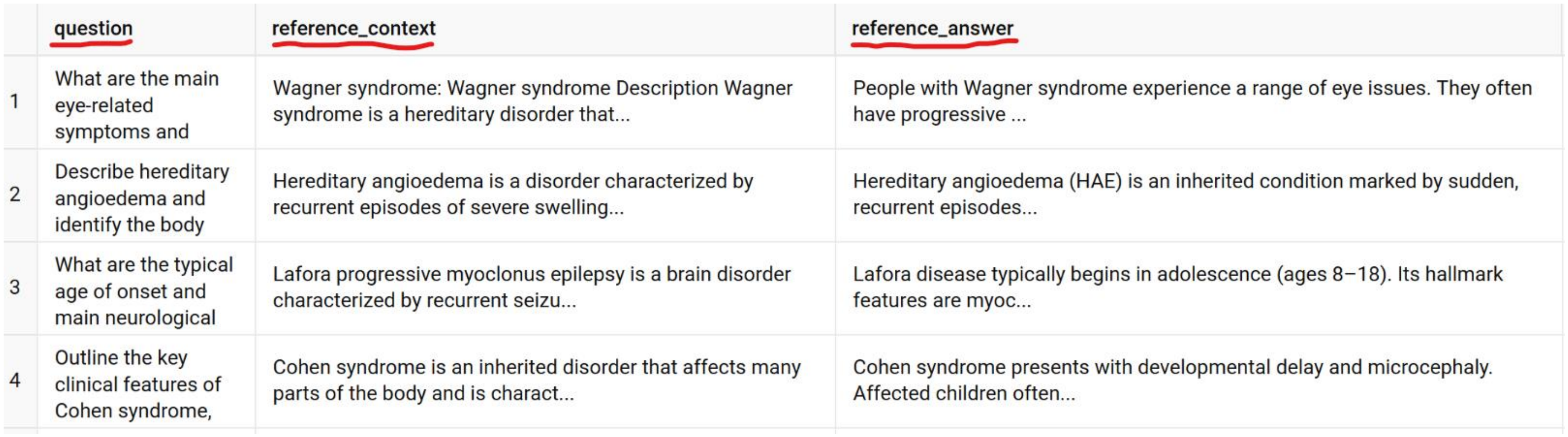
We recommend you leverage the following tools and models to ensure consistency in responses:

Sl. No	Component	Description
1	**Large Language Models (LLM)**	OpenAI LLM API (GPT-4o-mini)
2	**Vector Databases**	Modern vector database storage (Chroma DB)
3	**Embedding Models**	The preferred embedding model is OpenAI text-embedding-3-small
4	**Development Framework**	Orchestration frameworks such as LangChain
5	**Evaluation Framework**	Frameworks like DeepEval or your own custom LLM-as-a-Judge prompting

Data Set Details

- **For building your RAG System,** use the following file "**capstone1_rag_dataset.csv**".
  - Index the documents in this dataset to build your RAG System
  - The `context` column contains the documents to be indexed
  - Given the size of these documents, chunking is not necessary
  - Build and experiment with your RAG workflow with various retrieval strategies, prompt engineering for generating an accurate response and evaluating your system
- **For evaluating your RAG System**, use the **RAG Evaluation Dataset** consisting of **10 questions along with reference context and reference answers** in the file **"capstone1_rag_validation.csv**". These questions can

be used to evaluate whether your RAG System is working as expected using standard evaluation methods covered in the RAG course. The dataset looks like the following snapshot:

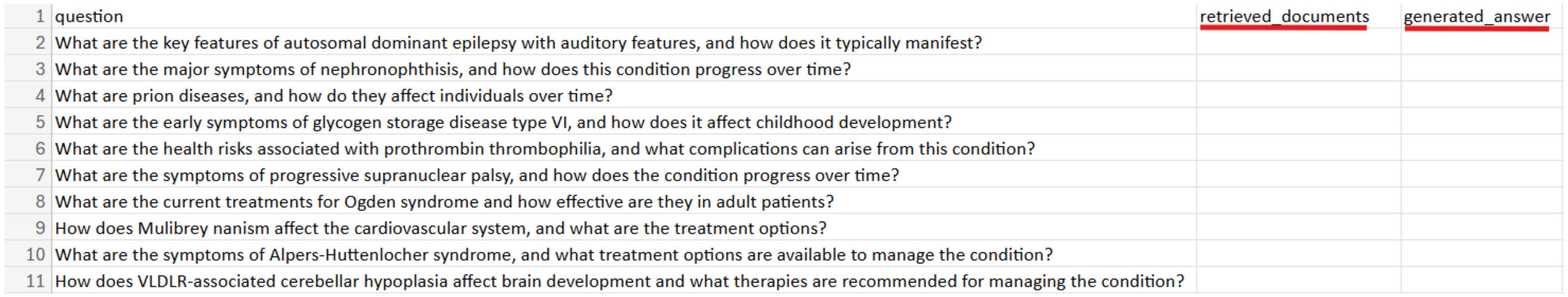


	question	reference_context	reference_answer
1	What are the main eye-related symptoms and	Wagner syndrome: Wagner syndrome Description Wagner syndrome is a hereditary disorder that...	People with Wagner syndrome experience a range of eye issues. They often have progressive ...
2	Describe hereditary angioedema 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...

Use the file and

- Run these 10 questions through your RAG System and find out the performance of your RAG System
- Evaluate the performance of your RAG application using:
    - Retriever performance metrics: precision@k, recall@k by comparing your topK retrieved context documents with the **`reference_context`**. Remember K can be <= 3.
    - Response performance metrics: response relevancy, hallucination check by comparing your generated answer with the **`reference_answer`**
- These metrics will help you evaluate how the retriever and response generator are performing and improve it as necessary (exploring different retrieval strategies and prompt engineering) before running the test questions (in the next point) on your system.

- **For testing your RAG System on unseen data**, use the **RAG Test Dataset** consisting of 10 questions. This file is "**capstone1_rag_test_questions.csv**". Run these questions through your RAG System and document the retrieved TopK (K<=3) documents and generated responses to evaluate the performance of your system before you submit. These outputs must be stored in the relevant columns (**`retrieved_documents`** and **`generated_answer`**) as depicted in the following dataset snapshot.



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?		

Submission Instructions:

What would you Submit?

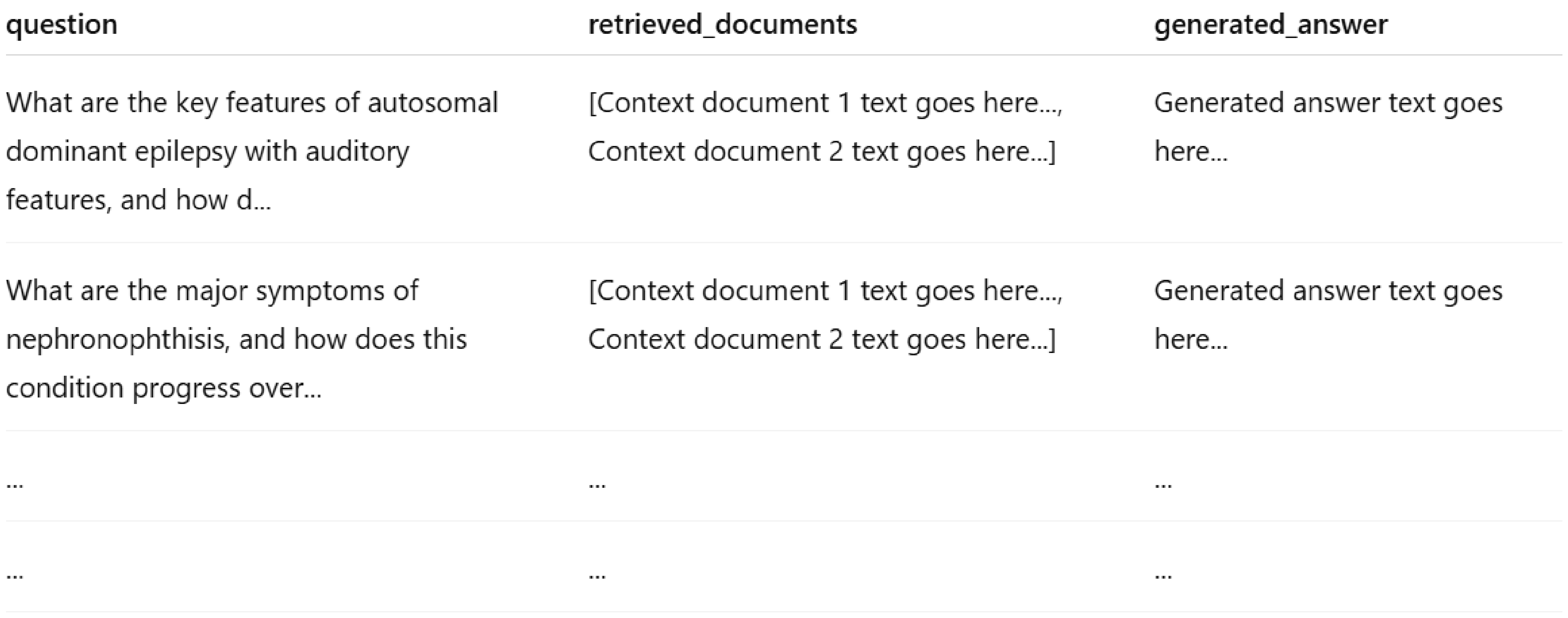
Your final submission should include the following

Sl No	Submission Item	Description
1	Final Jupyter Notebook (**`code.ipynb`**)	The notebook (**`code.ipynb`**) should contain the complete RAG system implementation workflow. This should include all explored strategies and clearly highlight the best-performing workflow at the end with code, comments and markdown headings and descriptions. You can also add in your evaluation results on the validation dataset
2	Test Dataset Responses (**`submission.csv`**)	● Based on the **RAG Test Dataset** (**capstone1_rag_test_questions.csv**), run each question on your RAG System and submit the following: ○ CSV file (**`submission.csv`**) containing the following: ■ Question (Column: **`question`**) ■ TopK(k<=3) retrieved documents for each question (by default you can use k=3, but you can experiment with various approaches like similarity thresholding or reranking and make the topK documents more relevant and that could have <3 context document also) (Column: **`retrieved_documents`**) ■ Generated response for each question (Column: **`generated_answer`**)

Submission Guidelines:

- The notebook file (`code.ipynb`) should contain the following
  - Detailed Markdown headings and descriptions along with proper code comments for code cells
  - Code should consist of the following:
    - Data loading and processing using the 100 documents in “**capstone1_rag_dataset.csv**”
    - Creating Vector Database and Retriever
    - Exploring a variety of retrieval strategies (semantic search, semantic search with threshold, hybrid, reranking) - **At least 2 or more should be explored.**
    - Connecting Generation Pipeline to the Retriever
    - Validation being done on the 10 questions of the validation dataset, **“capstone1_rag_validation.csv**” (results can just be printed or shown as a dataframe in notebook itself)
    - Code which should return topK (k<=3) retrieved documents and generated response for each input question from the test data, “**capstone1_rag_test_questions.csv**” (check the next point for more details on the submission format for the results)
- The test dataset responses file (`submission.csv`) should look like the following snapshot:

- Column 1 should be **`question`** which has each of the 10 questions (1 per row)
- Column 2 should be **`retrieved_documents`** where you can store the topK retrieved context documents as a list in the column **(the list can have anywhere between 0-3 documents per question based on your retrieval strategy; 0 means an empty list and could be possible for out of context questions)**
- Column 3 should be **`generated_answer`** where you can store the generated response



question	retrieved_documents	generated_answer
What are the key features of autosomal dominant epilepsy with auditory features, and how d...	[Context document 1 text goes here..., Context document 2 text goes here...]	Generated answer text goes here...
What are the major symptoms of nephronophthisis, and how does this condition progress over...	[Context document 1 text goes here..., Context document 2 text goes here...]	Generated answer text goes here...
...	...	...
...	...	...

- **Note:** Large Language Models (LLMs) are prone to hallucinations. In this task, you must treat the 100 documents provided as verified, factual context documents. Your RAG system should rely solely on these documents stored in your vector database to answer questions. Do **not** use the internal trained data of the LLM to answer questions, as the sources used during model training are unknown and unverifiable.

- If a question is out of context or only partially answerable based on the available documents, your system should be able to handle such questions by following the above guidelines and clearly acknowledging its limitations whenever necessary.

How Would Your Submission Be Evaluated

The submission would be evaluated by looking at several aspects, including:

- Performance metrics measuring RAG retrieval quality and response quality
  - Certain questions may not be answerable or might be partially answerable – make sure your RAG System can handle it
  - Consider the 100 context documents as the absolute truth and your RAG System should only use this information to answer the questions (do not let it use its internal trained knowledge as it may have been trained on public data which cannot be trusted)
- Code Quality
- RAG workflow used in the code (keep all your explored strategies along with the best one in the notebook)

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)	Final Score
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.	

	Definition	**Scoring Guidelines**	**Needs Improvement (1 point)**	**Satisfactory (2 points)**	**Excellent (3 points)**	**Final Score**
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.	
	Definition	**Scoring Guidelines**	**Needs Improvement (1 point)**	**Satisfactory (2 points)**	**Excellent (3 points)**	**Final Score**
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	Code is clean, well-structured, and modular. It includes comprehensive documentation, comments, clear variable and function	

				may be minor issues.	naming, and adheres to best practices for readability, modularity, and error handling.	
	Definition	**Scoring Guidelines**	**Needs Improvement (1 point)**	**Satisfactory (2 points)**	**Excellent (3 points)**	**Final Score**
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, filter-based retrieval) and incorporated fallback mechanisms - such as no-answer prompts to reduce hallucinations and handle out-of-context queries. Prompt	

					engineering has been applied to enhance generation quality.	

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.

Note to Learners

About Evaluation

Once you submit your capstone, it will be evaluated thoroughly, and your evaluation report will be available in MyLearning after 14 business days. You will receive a detailed report regarding your submission in your UAIS workspace. If you do not receive your grade after 21 business days of your submission, you can send us an email or drop in a comment in the AIML community.

About Resubmission

Once you submit your Capstone, you will be unable to resubmit until you have received your feedback. If you fail, you can resubmit your Capstone one more time without additional charges. After that, you will be assigned a new capstone project, and your department will be charged $100 per attempt for any additional resubmissions. Manager approval will be required.

--------------------------------end of document---