

An-Najah National University		Faculty of Engineering and Information Technology Information Technology Department Computer Science Apprenticeship	
Data Mining		Spring 2024/2025	
Instructor: Dr. Baha Thabet			
HW2	Topics: LangChain Agent, Text embedding, LSH and Cosine similarity search		
	Due by May 26 , 2025 at 23:59		

Assignment: Building a LangChain Agent with Vector Search and Visualization

Objective

In this assignment, you will develop a **LangChain-based intelligent agent** that performs chat-based question answering and semantic search on a document dataset using Locality Sensitive Hashing (LSH) and Cosine Similarity. You will also visualize search results using t-SNE.

Dataset

You will receive a dataset named: **knowledge-base2.zip**

Extract it into a folder called knowledge-base2/ and place it in the root directory of your project.

Requirements

Agent Features

You must implement the following components:


1. Document loading and chunking
2. Embedding with Hugging Face transformer
3. Storing and retrieving embeddings using Chroma vector store
4. LSH-based similarity search with FAISS
5. Cosine similarity search with scikit-learn
6. Chat-based question answering using a LangChain conversational agent

Required Libraries

Ensure you install **Python 3.11 or 3.12** and the following Python packages:

`pip install langchain langchain-openai langchain-community langchain-chroma python-dotenv sentence-transformers faiss-cpu scikit-learn plotly.graph_objects and transformers`

Important Implementation Notes

 Use the embedding model:

HuggingFaceEmbeddings(model_name="sentence-transformers/all-MiniLM-L6-v2")

✅ Use LangChain with either ChatOpenAI or Ollama's local model:

ChatOllama(model="llama3.2:latest")

💡 If you do not have OpenAI API keys, use ChatOllama with llama3.2:latest model running locally.

✅ Perform approximate search with:

faiss.IndexLSH(d, d)

✅ Visualize search results using **t-SNE** via sklearn.manifold.TSNE and color code by similarity.

✅ **Comments (#) are mandatory** throughout your code to explain your logic.

Your Tasks

You must implement and report the following in your notebook:

1. Chat with LangChain Agent

- Ask two different questions using your LangChain chat agent.
- Print both questions and their answers clearly.

2. Search Using FAISS LSH

- Perform a similarity search using faiss.IndexLSH.
- Return the top 3 most similar chunks (text + metadata + distance).
- Visualize the result using t-SNE, coloring points by similarity to the query.

3. Search Using Cosine Similarity

- Perform cosine similarity using sklearn.metrics.pairwise.cosine_similarity.
- Return the top 3 most similar chunks (text + metadata + score).
- Visualize the result using t-SNE, coloring points by similarity.

You can use the notebook that used in the lecture as a starting point

RAG_LangChainExample.ipynb

Submission Instructions

1. Submit a Jupyter notebook (.ipynb) file with all code and visible outputs.
2. Create 3 screenshots showing the following:
 - LangChain chat outputs
 - FAISS search + t-SNE plot
 - Cosine similarity search + t-SNE plot
3. Place the notebook and screenshots into a .zip file named: **Your-Name.zip** and upload it to the moodle.

Evaluation Rubric (100%)

Criteria	Weight
-----	-----
Correct use of Chroma vector store	10%
Correct use of LangChain agent	10%
Correct implementation of FAISS LSH	20%
Correct implementation of cosine sim.	20%
Code comments and clarity	20%
Complete outputs & screenshots	20%

Good Luck!

If you run into issues with environment setup or embedding models, try to search in ChatGPT or contact me.