Assignment 3: Build a Personal Learning Portal

Start Assignment

- Due Oct 12 by 11:59pm
- Points 100
- Submitting a text entry box, a website url, a media recording, or a file upload

Assignment 3 – Build a Personal Learning Portal (PLP)

From Learning Questions to Deep Search, RAG, and Reflecti Evaluation

1. Assignment Overview

In this capstone assignment, you will build a **Personal Learning Portal (PLP)** — an interactive, L enabled system designed to help a user (yourself or a target learner) understand a complex doma deeply.

Your PLP should guide users through a learning journey:

- **Ask meaningful questions**
- **Collect and analyze diverse sources** (text, video, audio)
- **Use retrieval-augmented generation (RAG)** to generate answers
- **Evaluate learning outcomes** using quantitative and qualitative methods

You may optionally extend your PLP using **reasoning agents**, **multi-agent planning**, or **knowled graphs**.

2. Learning Objectives

By the end of this assignment, you will be able to:

- Translate open-ended topics into structured learning questions and objectives
- Curate domain-specific learning resources via search and deep search tools
- Build a retrieval-augmented LLM system for question-answering and synthesis
- Evaluate factuality and completeness using frameworks like RAGAs and ARES
- Reflect on and document design decisions, learning outcomes, and limitations

3. Assignment Workflow

Step 1: Define Topic and Learning Goals

- Choose a domain (e.g., Quantum Computing, AI in Climate Risk)

- Formulate 5–7 *learning questions* (What, Why, How, When, Where, Who)
- Translate them into 3–5 structured **learning objectives** (using Bloom's verbs)

Step 2: Draw Inspiration from Learning Platforms

- Review 2–3 Personal Learning Portals (Degreed, Canvas, EducateMe, Valamis)
- Identify and adapt 2–3 features (e.g., module views, progress tracking, user feedback)

Step 3: Web and Deep Search for Resources

- Start with naive search (Google, Perplexity, Bing, Scholar)
- Optionally use deep search frameworks like:
  - **ManuSearch** [Huang et al., 2025]
  - **Open Deep Search (ODS)** [Alzubi et al., 2025]
  - **R-Search** [Zhao et al., 2025]

Collect:

- 10–15 quality sources: academic papers, reports, blogs, videos, podcasts
- For each source, document: title, URL, type, and relevance

Step 4: Build Your RAG System

- Use LangChain, LlamaIndex
- Process documents: chunk, embed, store
- Build a query-response system that:
  - Retrieves relevant chunks
  - Generates a response
  - Displays citations or source metadata

4. **System Enhancements (Optional – Bonus of 10%)**

You may choose to **extend your PLP system** with one of the following enhancements:

Option A: Add Reasoning Capabilities

- Use Chain-of-Thought (CoT), Self-Ask, or ReAct-style reasoning.
- Evaluate reasoning quality using **llm-reasoners (https://github.com/maitrix-org/llm-reasoner**

Option B: Add Agentic Workflow (Optional)

- Implement a multi-tool or multi-agent system (e.g., planner → search → summarizer).
- Evaluate with **RagaAI-Catalyst (https://github.com/raga-ai-hub/RagaAI-Catalyst)** or with benchmarks like **Deep Research Bench**.

Option C: Use a Knowledge Graph (Optional)

- Use Neo4j or another tool to extract and visualize concepts/entities from your corpus.
- Connect the graph to an LLM to assist in guided exploration.

These enhancements are optional and meant to challenge those with extra interest or prior experience.

5. Evaluate What Was Learned

Use at least one evaluation approach to assess your system's effectiveness:

- **RAGAs** or **ARES**: Measure factuality, groundedness, context recall, and relevance.
- **Deep Research Bench**: Run your agent/system against one or more scenarios (if applicable).
- **User Reflection/Feedback**: Ask your system to reflect on what it learned; log outputs and ass alignment with goals.

Document 3–5 sample queries, show the system's answers, and explain whether they helped *you a simulated user* learn the topic better.

6. Deliverables

Component	Format	Required
System code	.ipynb, .py, or repo	✅
Learning corpus	.csv, .json, or list of URLs with metadata	✅
PLP Interface	Notebook, Streamlit, or Gradio app	✅
Evaluation log + output samples	.csv, .md, or screenshots	✅
Final report	2–3 pages .md or .pdf	✅
GitHub repository link	Submit on Canvas	✅

7. Evaluation Rubric (Highlights)

Category	Points	Description
1. Learning Goals & Alignment	20 pts	Clear learning objectives; thoughtful learning questions; well-scoped inquiry. Mapped to source collection and system outpu
2. Search Strategy & Source Curation	15 pts	Use of naive and deep search (optional); documentation of hov sources were selected. Diversity and credibility of the learning corpus.

Category	Points	Description
3. RAG Pipeline Implementation	20 pts	Working RAG pipeline using appropriate tools (e.g., LangChain LlamaIndex). Includes chunking, embedding, citation display. Code is modular and well-documented.
4. Evaluation & Evidence	15 pts	Use of tools like RAGAs, ARES, or Deep Research Bench. Cle evaluation methodology and interpretation of outputs. Qualitati + quantitative insights.
5. PLP Interface & Learning Experience	20 pts	Interactive, usable system design. Incorporates features inspire by PLPs (e.g., feedback, modular views, iteration history). Prioritizes learning experience.
6. Final Report & Reflection	10 pts	Coherent summary of system goals, methods, results, and reflections. Includes system diagram and thoughtful analysis of what worked/didn't.

Bonus: Optional Enhancements (+10 pts max)

Category	Points	Criteria
Advanced System Enhancements	+10 pts	For students who go beyond the base project by integrating:

- Reasoning agents (ReAct, CoT, Self-Ask)
- Multi-agent workflows (e.g., planner → searcher → summarizer)
- Knowledge graphs (Neo4j, RDF)

Grading:

- Excellent (9–10 pts): well-justified, well-executed
- Partial (5–8 pts): implemented but not fully integrated
- Minimal (1–4 pts): attempted but weakly scoped |

8. **Helpful Resources**

Building LLM Interfaces

- [Building the Simplest LLM with Jupyter Notebook: A Student's Guide (https://coda.io/@peter-sigurdson/building-the-simplest-llm-with-jupyter-notebook-a-students guide)](https://coda.io/@peter-sigurdson/building-the-simplest-llm-with-jupyter-notebook-a-students guide)
- [From Notebook to Web App in 10 Minutes with Streamlit (Medium) (https://medium.com/@bhagyarana80/from-notebook-to-web-app-in-10-minutes-with-streamlit-](https://medium.com/@bhagyarana80/from-notebook-to-web-app-in-10-minutes-with-streamlit-)

8a80aa0dab0a)

- Streamlit Official — "Create an App" Tutorial (https://docs.streamlit.io/get-started/tutorials/create-an-app?utm_source=chatgpt.com)
- Deploying Machine Learning Models with Python & Streamlit (365 Data Science) (https://365datascience.com/tutorials/machine-learning-tutorials/how-to-deploy-machine-learning-models-with-python-and-streamlit/?utm_source=chatgpt.com)
- FreeCodeCamp — "How to Build Your AI Demos with Gradio" (https://www.freecodecamp.org/news/how-to-build-your-ai-demos-with-gradio/?utm_source=chatgpt.com)
- DataCamp — "Building User Interfaces for AI Applications with Gradio" (https://www.datacamp.com/tutorial/gradio-python-tutorial?utm_source=chatgpt.com)
- PyImageSearch — "Introduction to Gradio for Building Interactive Applications" (2025) (https://pyimagesearch.com/2025/02/03/introduction-to-gradio-for-building-interactive-application utm_source=chatgpt.com)

LangChain & LlamaIndex

- LangChain Official Tutorials (https://python.langchain.com/docs/tutorials/?utm_source=chatgpt.com)
- LangChain Tutorial: A Guide to Building LLM-Powered Applications (Elastic blog) (https://www.elastic.co/blog/langchain-tutorial?utm_source=chatgpt.com)
- LlamaIndex Starter Tutorial (Using OpenAI) (https://docs.llamaindex.ai/en/stable/getting_started/starter_example/?utm_source=chatgpt.c

9. **References (MLA Format with URLs)**

1. Huang, Lisheng, et al. "ManuSearch: Democratizing Deep Search in Large Language Models. *arXiv*, 2025. https://arxiv.org/abs/2505.18105 (https://arxiv.org/abs/2505.18105?utm_source=chatgpt.com)
2. Alzubi, Salaheddin, et al. "Open Deep Search: Democratizing Search with Open-Source Reasoning Agents." *arXiv*, 2025. https://arxiv.org/abs/2503.20201 (https://arxiv.org/abs/2503.20201)
3. Zhao, Qingfei, et al. "R-Search: Empowering LLM Reasoning with Search via Multi-Reward Reinforcement Learning." *arXiv*, 2025. https://arxiv.org/abs/2506.04185 (https://arxiv.org/abs/2506.04185?utm_source=chatgpt.com)
4. Chen, Mingyang, et al. "ReSearch: Learning to Reason with Search for LLMs via Reinforceme Learning." *arXiv*, 2025. https://arxiv.org/abs/2503.19470 (https://arxiv.org/abs/2503.19470?utm_source=chatgpt.com)
5. Bosse, Nikos I., et al. "Deep Research Bench: Evaluating AI Web Research Agents." *arXiv*, 20 https://arxiv.org/abs/2506.06287 (https://arxiv.org/abs/2506.06287?utm_source=chatgpt.com
6. Zhang, Wenlin, et al. "Deep Research: A Survey of Autonomous Research Agents." *arXiv*, 202 https://arxiv.org/abs/2508.12752 (https://arxiv.org/abs/2508.12752)
7. Center for Teaching. *Bloom's Taxonomy: Overview*. Vanderbilt University, https://cft.vanderbilt.edu/wp-content/uploads/sites/59/Blooms-Taxonomy.pdf (https://cft.vanderbilt.edu/wp-content/uploads/sites/59/Blooms-Taxonomy.pdf). Accessed 7 Sept 2025.