LLMs & Multi-Agent Orchestration - Exercise 5

Context Windows, RAG, and Scientific Experimentation

Selected Option: Option 1

Group Name:	eldad_ron_bar_yacobi
Team Member 1:	Eldad Ron (ID: 207021916)
Team Member 2:	Bar Yacobi (ID: 315471367)
Submission Date:	2025-12-10
Self-Grade:	100/100

GitHub Repository:
https://github.com/er1009/LLMs-And-Multi-Agent-Orchestration-Course/tree/main/ex5

Self-Assessment & Justification

We have rigorously self-assessed our submission for Exercise 5 (Context Windows Lab) and assigned a grade of **100/100**. This grade reflects a submission that not only meets but exceeds the requirements for a scientific, empirical exploration of LLM context limitations.

Key Strengths & Achievements:

1. **Comprehensive Experimental Suite:** We successfully implemented all four required experiments:

Needle in a Haystack: Demonstrated the "Lost in the Middle" phenomenon with statistical significance.
Context Size Impact: Quantified the latency/accuracy trade-off with growing context windows.
RAG vs. Full Context: Empirically proved RAG's superiority (80% vs 60% accuracy) and efficiency.
Context Strategies: Implemented and compared SELECT, COMPRESS, and WRITE strategies for long-horizon tasks.

2. **Production-Grade Engineering:**

Modular Architecture: Clean separation of concerns using Abstract Base Classes (Strategy Pattern) for experiments.
Robust Tooling: A resilient OllamaClient with backoff retries and a modern PersistentClient integration for ChromaDB.
Configuration Management: Centralized settings using Pydantic-style dataclasses and environment variables.

3. **Scientific Rigor & Analysis:**

All experiments run multiple trials (N=10) to ensure statistical validity.
Automated calculation of means, standard deviations, and confidence intervals.
Generation of publication-quality visualizations (matplotlib/seaborn) saved directly to the results directory.

4. **Documentation & UX:**

Extensive **PRD** and **Architecture** documents provided.
A polished CLI interface with progress bars (tqdm), clear logging, and color-coded summaries.
One-click setup and execution via setup.sh and main.py.

Our submission represents a complete, robust, and scientifically rigorous lab environment for studying LLM context behaviors.

Rubric Compliance Checklist

Category	Status	Notes
Project Documentation	20/20	Comprehensive PRD, Architecture, and experimental goals.
README & Code Docs	15/15	Clear setup, CLI usage, and type-hinted code.
Structure & Quality	15/15	SOLID principles, modular design, clean abstractions.
Config & Security	10/10	Env vars for models/secrets, secure ChromaDB setup.
Testing & QA	15/15	Unit tests present, robust error handling implemented.
Research & Analysis	15/15	Statistical aggregation, dual-axis plotting, N-trials.
UI/UX & Extensibility	10/10	Professional CLI, progress tracking, extensible design.
Total	100/100	Full Compliance

Special Notes

Ollama Requirement: This project requires a local instance of Ollama running. The setup script setup.sh attempts to install it automatically if missing. **Model Selection:** The experiments default to llama2 but can be configured to use mistral or tinyllama via the CLI or .env file. **First Run:** The first execution of Experiment 3 (RAG) may be slower due to the initial download of the embedding model (all-MiniLM-L6-v2). **Visualizations:** All generated graphs are saved to the results/ directory. Python's matplotlib is used backend-agnostically to ensure compatibility across OS environments.

Special Documents

The following additional documentation is included in the repository: **docs/PRD.md:** Product Requirement Document detailing the experimental design, success metrics, and functional requirements. **docs/ARCHITECTURE.md:** Technical architecture overview, including class diagrams, data flow, and key design decisions (ADRs). **PROJECT_SUMMARY.md:** A high-level executive summary of the entire project status and generated artifacts.

Comments

(This space is reserved for grader comments)