Self-Assessment Report

Ollama Chat Application

LLMs and Multi-Agent Systems Course

Group Code:	OmerAndYogever
Students:	Omer Caplan, ID: 208753665 Yogev Cuperman, ID: 207540550
Project Name:	Ollama Chat Application
Repository:	https://github.com/OmerCaplan/AgentsOrchestration_assignment_1
Assessment Date:	November 10, 2025
Self-Grade:	79/100

Category-by-Category Assessment

Category	Weight	My Score	Final Score
Project Documentation	20%	16/20	16
README & Code Documentation	15%	14/15	14
Project Structure & Code Quality	15%	13/15	13
Configuration & Security	10%	9.5/10	9.5
Testing & QA	15%	12.5/15	12.5
Research & Analysis	15%	0/15	0
UI/UX & Extensibility	10%	8.5/10	8.5
TOTAL	**100%**		**79**

Grade Level: 70-79 (■■■ - Good)

With a self-grade of 79, we expect **reasonable and balanced** scrutiny. The evaluators will check adherence to main criteria but will allow room for minor imperfections, as this is a first assignment in the course.

Justification for Self-Assessment

Strengths

The project demonstrates strong software engineering practices with excellent documentation across multiple dimensions. The formal PRD provides clear project requirements and objectives, while the comprehensive README offers detailed installation, configuration, and troubleshooting guidance. The README documentation is particularly strong with step-by-step instructions, multiple platform support, comprehensive troubleshooting section, and clear usage examples, earning it 14/15 points. Code quality is excellent with proper documentation through docstrings and comments, descriptive naming conventions, and clean, maintainable structure with good modular organization earning 13/15 points.

Security and configuration management are exemplary with proper use of .env files, no hardcoded credentials, and clear documentation of all configuration parameters. The testing suite is comprehensive, covering 16 different test scenarios with a 100% success rate.

The project includes three distinct UI implementations - Basic Tkinter, Modern CustomTkinter, and Web Interface. This demonstrates versatility and user-centric design, though the UI/UX score reflects that while multiple implementations exist, there may be room for additional polish in areas such as explicit extensibility mechanisms, accessibility features, or more advanced UI/UX patterns.

The documentation of the AI interaction process adds educational value and transparency to the development methodology, showing the iterative development approach and actual prompts used.

Weaknesses

The most significant deficiency is the complete absence of research and analysis components, worth 15% of the grade. The project does not include experiments comparing different models, parameter sensitivity analysis, performance benchmarking, Jupyter notebooks with data analysis, visualizations of model behavior, or academic references. This represents the primary factor limiting the grade.

Additional gaps include the lack of formal architecture diagrams such as C4 models or UML diagrams, absence of formal test coverage reports, and potential areas for improvement in UI/UX such as explicit plugin systems, advanced accessibility features, or interactive dashboards. The project structure, while good and modular, may not strictly follow all recommended folder organization patterns.

Investment

Significant effort was invested in creating three complete UI implementations, writing comprehensive documentation including a formal PRD, developing a robust test suite with 16 test cases, and properly documenting the AI-assisted development process. The work includes Windows-specific instructions, troubleshooting guides, and multiple installation pathways. This represents substantial work beyond a minimal implementation and demonstrates commitment to delivering a polished, user-ready application.

Innovation

Providing three UI options demonstrates creative thinking and understanding that different users have different preferences and use cases. The inclusion of a modern CustomTkinter interface shows awareness of contemporary UI trends, while the web-based option demonstrates versatility in implementation approaches. The comprehensive documentation of the AI interaction process provides educational value beyond the technical implementation.

Learning

The project demonstrates learning in multiple areas: GUI development with different frameworks, API integration with local LLM services, environment management using UV package manager, security best practices, comprehensive testing methodologies, and professional documentation standards. The progression from basic Tkinter to modern CustomTkinter to web-based interfaces shows skill development throughout the project.

Academic Integrity Declaration

We hereby declare that:

• Our self-assessment is honest and truthful
• We checked our work against all criteria before determining the grade
• We are aware that a high self-grade will lead to more rigorous review
• We accept that the final grade may differ from our self-assessment
• This work is the product of our work (of the team) and we are responsible for all software

Student 1: Omer Caplan (ID: 208753665)

Signature: Omer **Date:** November 10, 2025

Student 2: Yogev Cuperman (ID: 207540550)

Signature: Yogev **Date:** November 10, 2025

Assignment 1: LLM Agent Orchestration

Grade Report

Student ID:	38966
Team:	OmerAndYogever
Repository:	https://github.com/OmerCaplan/AgentsOrchestration_assignment_1
Assessment Date:	December 01, 2025

FINAL GRADE	**44%**
STATUS	**NEEDS IMPROVEMENT**

Assessment Feedback

This submission requires significant improvement. You need to focus on fundamental software engineering practices including planning, documentation, testing, and version control.

Strengths

• Good documentation (comprehensive README, 100% docstring coverage)

• Multiple UI implementations (3 different interfaces)

• Clean PDR document

Required Improvements

• CRITICAL: Add .gitignore file (security requirement)

• CRITICAL: Add research analysis section (parameter investigation required - 0/10 points lost)

• CRITICAL: Improve git workflow (only 2 commits - version control requirement)

• Add comprehensive unit tests (more coverage needed)

• Implement cost tracking and analysis

• Add prompt documentation

• Add CI/CD pipeline

• Improve code modularity (files too large)

Please review the requirements and resubmit. Please review course materials and requirements carefully before your next submission.

Assessed: December 01, 2025
This grade reflects your overall software engineering practices