COURSE SYLLABUS

AI610 • LLM Systems & Evaluation

School of Computational Arts & Sciences • Graduate • 6 ECTS

Overview

Build production-grade LLM applications beyond prompting: retrieval and grounding, safety and policy checks, tool use, and systematic evaluation harnesses. Students implement test suites for quality, hallucination risk, and regression, then iterate on architecture with measured evidence.

LOGISTICS

Credits: 6 ECTS
Level: Graduate
School: School of Computational Arts & Sciences
Prerequisites: None listed
Tags: llm, evaluation, responsible-ai
Meeting time: Weekly systems lecture + evaluation practicum
Instruction mode: Build-and-measure: every feature ships with an evaluation harness

LEARNING OUTCOMES

You will be able to:
- Build a retrieval-augmented generation (RAG) pipeline with measurable quality gates
- Design safety checks (policy filters, refusal handling) and test them
- Run offline and online evaluations and interpret trade-offs

- Build a retrieval-augmented generation (RAG) pipeline with measurable quality gates
- Design safety checks (policy filters, refusal handling) and test them
- Run offline and online evaluations and interpret trade-offs

ASSESSMENT

Components
- Coursework: 60%
- Final project: 40%

Grades prioritize traceable evaluation. Each milestone includes (1) a hypothesis, (2) an evaluation dataset or rubric, (3) results, and (4) a decision. ’Nice demos’ without measurement do not score well.

WEEKLY PLAN

Schedule
Week 1: Week 1
  - LLM system anatomy: prompting, retrieval, tools, and caching
Week 2: Week 2
  - Evaluation sets: gold answers, rubrics, and human review protocols
Week 3: Week 3
  - RAG basics: chunking, embedding, ranking, and failure analysis
Week 4: Week 4
  - Safety: policy filters, prompt injection, and data leakage

Extended outline

- LLM system anatomy: prompting, retrieval, tools, and caching
- Evaluation sets: gold answers, rubrics, and human review protocols
- RAG basics: chunking, embedding, ranking, and failure analysis
- Safety: policy filters, prompt injection, and data leakage
- Cost/performance: latency budgets, caching, and batch evaluation
- Final: ship an evaluated LLM feature with a report and limitations

POLICIES & RESOURCES

- Security: never paste secrets into prompts; assume prompts may be logged.
- Privacy: do not use personal data in evaluation sets unless explicitly approved.
- Reproducibility: record model versions, prompts, and evaluation code for reruns.

Suggested resources

- Evaluation worksheet: rubric + examples of good failure notes
- RAG debugging checklist: retrieval failures vs generation failures
- Safety test suite starter: injection attempts and refusal checks