Project Proposal: DSCI.644.02 — Software Engineering for Data Science

1. Project Overview

This semester, you will apply rigorous Software Engineering principles to Data Science and AI workflows. Your team must choose **one** of the following three tracks. Regardless of the choice, the focus is on *engineering quality*—reproducibility, testing, modularity, and documentation—not just model accuracy.

Option 1: Randomly Assigned Industry/Research Project

- **Context:** Your team has been paired with a specific problem statement or stakeholder (e.g., from a research lab or industry partner).
- **Action:** Refer to **MyCourses** > **Project Assignments** to view your specific topic.
- **Goal:** Apply the SE lifecycle (Requirements $\to$ Prototype $\to$ Optimization $\to$ Delivery) to your assigned problem.

Option 2: The "Meaning-Typed Programming" (MTP) Challenge

- **Focus:** Engineering robust AI applications using the **byLLM** framework.
- **Context:** Traditional LLM integration relies on fragile string prompting. You will investigate **Meaning-Typed Programming**, where you use semantic types (native code) to control AI behavior.
- **Goal:** Build a data-intensive application (e.g., an automated data cleaning pipeline or a complex information extraction tool) using the `by <model>` operator. Evaluate if this method improves code maintainability and reliability compared to standard prompt engineering.
- **Resources:** byLLM (Jaseci) & *MTP: Meaning-Typed Language Abstraction (2025)*.

Option 3: The AI-Agent Track (MSR 2026 Mining Challenge)

- **Focus:** Mining Software Repositories (MSR) to analyze the quality of AI-generated code.
- **Context:** The **AIDev Dataset** contains "Agentic-PRs"—pull requests created by AI agents.
- **Goal:** Treat code as data. Build an analysis pipeline to investigate the security, maintainability, or correctness of code written by AI agents compared to humans.
- **Resources:** MSR 2026 Mining Challenge.

---

2. Potential Research Questions (RQs)

Select or adapt 1-2 questions. (For Option 1, derive similar questions based on your specific assignment).

For Option 2 (MTP / byLLM)

- **RQ1 (Data Quality):** Can the `by` operator be used to automatically detect and repair "dirty data" (e.g., inconsistent date formats) more reliably than regex-based scripts?
- **RQ2 (Engineering Overhead):** How does the development time and lines of code (LOC) for an MTP-based scraper compare to a traditional `BeautifulSoup` + `OpenAI API` approach?
- **RQ3 (Determinism):** How often does the `by` operator return the wrong data type (schema violation) when processing large, noisy datasets compared to manual prompting?

For Option 3 (AI-Agent / MSR)

- **RQ1 (Code Smells):** Do AI agents introduce specific types of "Code Smells" (e.g., Long Method, Duplicate Code) more frequently than human developers in Data Science notebooks?
- **RQ2 (Dependency Management):** How often do Agentic-PRs fail due to hallucinated or incorrect library dependencies (e.g., `pip install non_existent_package`)?
- **RQ3 (Security):** Can we detect hardcoded credentials or PII leaks in AI-generated pull requests using static analysis?

---

3. Project Phases & Deliverables

Phase 1: Definition and Open Investigation

- **Goal:** Define the problem scope and validate the feasibility of your approach.
- **Activities:**
  - **Literature/Context Review:** Read the MTP paper (Opt 2), the MSR challenge docs (Opt 3), or your assigned brief (Opt 1).
  - **Problem Definition:** Clearly state what you are building or analyzing.
  - **Sanity Check:** Propose a high-level architecture. (e.g., "We will ingest the AIDev dataset into MongoDB and use PyDriller to extract metrics.")
- **Deliverable (20%):** 1-2 page report outlining the problem, dataset strategy, and Research Questions.

Phase 2: Initial Solution (The Baseline)

- **Goal:** Build an end-to-end "Skeleton" pipeline to identify engineering bottlenecks.
- **Activities:**
  - **Data Ingestion:** Write scripts to load and clean your initial data.
  - **Baseline Implementation:**
    - *Opt 1:* A basic script that attempts to solve the assigned problem.
    - *Opt 2:* A "Manual Prompting" version of your app to serve as a baseline.
    - *Opt 3:* A simple analysis script (e.g., counting PRs) to prove you can parse the data.

    - **Reflection:** Identify "Why is this hard?" (e.g., "The data is too large for memory" or "The LLM is inconsistent").

- **Deliverable (20%):** 3-5 page report detailing the initial design, preliminary results, and technical challenges.

Phase 3: Optimized Solution (Refinement)

- **Goal:** Apply SE patterns to improve performance, reliability, and validity.
- **Activities:**
    - **Optimization:** Address Phase 2 limitations.
        - *Opt 1:* Modularize code, add logging, and improve algorithmic efficiency.
        - *Opt 2:* Implement the full **byLLM** abstraction with refined Type definitions.
        - *Opt 3:* Add advanced metrics (e.g., Cyclomatic Complexity) and statistical significance tests.
    - **Validation:** Run your solution on the full dataset.
- **Deliverable (20%):** 8-10 page report presenting the improved architecture, quantitative results, and a discussion of trade-offs.

Phase 4: Final Report and Package

- **Goal:** Documentation, Validation Analysis, and Packaging.
- **Activities:**
    - **Confusion Matrix / Error Analysis:**
        - If building a tool (Opt 1 & 2): When does it fail? (False Positives/Negatives).
        - If analyzing data (Opt 3): What are the threats to validity?
    - **Documentation:** `README.md`, environment setup (`requirements.txt` / Dockerfile), and inline comments.
    - **Packaging:** Organize code into a proper structure (e.g., `src/`, `tests/`, `data/`).
- **Deliverable (40% Total):**
    - **Presentation (20%):** Live demo and defense of your results.
    - **Final Report (20%):** 10+ page report + GitHub Repository link.

---

4. Grade Distribution Summary

Phase	Focus	Deliverable	Weight
I	**Problem Definition**	1-2 Page Report	**20%**
II	**Baseline & Initial Prototype**	3-5 Page Report	**20%**

Phase	Focus	Deliverable	Weight
III	**Optimized Solution**	8-10 Page Report	**20%**
Pre-IV	**Presentation**	Live Demo/Presentation	**20%**
IV	**Final Package**	10+ Page Report + Code	**20%**