Phase 2 Evaluation Report

Project Topic: Beyond Autocomplete: An Empirical Investigation into AI-Assisted Programming—Productivity Gains vs. Cognitive Skill Decay

1. Executive Summary

This report evaluates the performance of the Phase 2 RAG Pipeline, which was designed to transition from the prompt engineering experiments in Phase 1 to a production-grade retrieval system. The system ingested a curated corpus of 15 academic papers (focusing on AI Productivity, Security, and Skill Decay) and was evaluated against a set of 20 diverse research queries.

Key results indicate that the Zero-Inference and Strict Citation constraints defined in Phase 1 were successfully operationalized in code. The implementation of a Cross-Encoder Reranker (Enhancement) significantly improved retrieval precision for specific statistical claims. The designed system achieved a 100% Citation Compliance Rate (no hallucinations observed across 20 runs).

2. System Architecture & Ingestion Strategy

To ensure "Research-Grade" output, the ingestion pipeline was engineered to preserve strict metadata provenance.

- **Corpus:** 15 PDF documents (including Peng2023, Perry2023, Prather2023).
- **Chunking Strategy:** RecursiveCharacterTextSplitter (Size: 1000, Overlap: 100). This size was selected to capture full distinct arguments while maintaining retrieval granularity.
- **Metadata Injection:** A critical lesson from Phase 1 was that models confuse sources if context is vague. In Phase 2, every text chunk was programmatically tagged with SourceID, Year, and Title before embedding.
- **Vector Store:** ChromaDB using text-embedding-3-small.

3. Evaluation Methodology

The system was stress-tested using a 20-query evaluation set designed to mimic real-world research workflows. The queries were categorized into four logic types:

- **Direct Retrieval (5):** Extraction of specific metrics (e.g., "How much faster was the treatment group in Peng2023?").
- **Critical Analysis (5):** Handling negative results or risks (e.g., "Security vulnerabilities in Pearce2022").
- **Cross-Source Synthesis (8):** Combining conflicting papers (e.g., Speed vs. Security trade-offs).

- **Edge Cases (2):** Testing "Trust Behavior" and refusal to hallucinate (e.g., "Impact on cooking recipes").

4. Quantitative Metrics

The outputs were scored based on the Phase 2 Rubric (1–4 Scale):

- Groundedness / Faithfulness (Score: 4.0): The model strictly adhered to the "Zero-Inference" rule. When evidence was missing, such as in edge cases Q19 and Q20, it explicitly stated "I cannot find evidence" rather than hallucinating an answer.
- Citation Correctness (Score: 4.0): Every major claim in the evaluation logs is supported by an inline (SourceID) tag. No fabricated citations were observed across all the 20 test runs.
- Retrieval Precision (Score: 3.5): The Reranker successfully surfaced relevant chunks for 18 out of 20 queries. However, a minor friction point was observed where the system struggled with industry-specific jargon that was not explicitly present in the academic texts.

5. Analysis of Failure Modes & Successes

Three representative cases from the evaluation_results.json log highlight the system's capabilities and boundaries.

Success Case: Cross-Source Synthesis

- **Query (Q16):** "Compare the findings of Peng2023 and Perry2023 regarding the trade-off between speed and security."
- **Result:** The system correctly retrieved chunks from Peng2023 (citing 55.8% speedup) and Perry2023 (citing insecure code generation).
- **Analysis:** This validates the **Enhancement (Reranking)**. A standard similarity search might have prioritized generic "AI impact" sections. The Reranker correctly identified the high-value Results sections from two distinct PDFs, allowing the LLM to construct a "On one hand... on the other hand" narrative.

Failure Mode 1: The "Vocabulary Mismatch" Gap

- **Query (Q18):** "What does the corpus say about the impact of AI on 'Code Churn'?"
- **Result:** "I cannot find evidence in the provided corpus to answer this."
- **Analysis:** While the corpus discusses code modification and refactoring, the specific n-gram "Code Churn" does not appear frequently in the ingested chunks.
- **Verdict: Correct Refusal.** In Phase 1, Prompt A might have hallucinated a definition. In

Phase 2, the system correctly identified a retrieval gap. This is a "safe failure."

Failure Mode 2: The Corpus Boundary

- **Query (Q20):** "What is the impact of AI on C++ specifically versus Python?"
- **Result:** "I cannot find evidence... the context provided focuses on... productivity and security."
- **Analysis:** The corpus is heavily skewed towards Python (Weber2024) and JavaScript (Peng2023). There is no comparative study of C++ in the manifest.
- **Verdict:** Faithfulness Verified. The system refused to invent a comparison that did not exist, proving the "Trust Behavior" requirement is met.

Failure Mode 3: The "Context Fragmentation" Gap (Recall Limit)

- **Query (Q10):** "What specific security vulnerabilities (CWEs) were most common in Pearce2022's analysis?"
- **Result:** The system correctly identified high-level categories like "SQL Injection" and "Cross-Site Scripting" but failed to provide the exhaustive list of all 89 tested scenarios.
- **Analysis:** This highlights a limitation of the Fixed-Size Chunking Strategy (1000 chars). The original paper's "Table 1" (listing all CWEs) spans multiple pages. The chunker split this table into disjoint pieces. Since the retrieval is limited to k=4 chunks, the system retrieved the first chunk of the table (containing the most common CWEs) but "orphaned" the subsequent chunks containing the long-tail vulnerabilities.
- **Verdict:** Recall Limitation. While the answer was accurate (High Precision), it was incomplete (Low Recall) due to the architectural decision to prioritize top-k chunks over full-document parsing. This suggests Phase 3 may need a "Table-Aware" chunking strategy.

6. Impact of Enhancements (Reranking)

To satisfy the "Measurable Improvement" requirement, I integrated a Cross-Encoder Reranker (ms-marco-MiniLM-L-6-v2).

- **Mechanism:** The system retrieves the top 6 semantic matches, then the Cross-Encoder re-scores them based on deep relevance, passing only the top 4 to the LLM.
- **Measurable Benefit:** For Query Q6 ("What percentage of code... contained vulnerabilities?"), the semantic search initially ranked the Introduction section of Pearce2022 highest (which contained keywords but no data). The Reranker successfully promoted the Results section (containing the "40%" statistic) to the #1 slot, ensuring the answer was numeric and precise.

7. Phase 3 Design Choices

Based on these findings, the following requirements are set for the Phase 3 Product:

1. **Artifact Generation:** The system handles synthesis (Q16) well but outputs it as text. Phase 3 will try to convert this part into a structured Evidence Table (Claim | Source | Confidence).
2. **Gap Analysis Included:** When the system refuses an answer (like Q18), the response should explicitly highlight why (e.g., "Keywords not found in corpus") to help the user refine their search.