Phase 1 Framing Brief

Personal Research Portal (PRP)

Research Domain

The domain of this project is the impact of AI-assisted tools (such as large language models and coding assistants) on software engineering practice. In particular, this research focuses on how these tools influence developer productivity, code quality, and skill development, with an emphasis on early-career and junior software engineers.

As AI tools become increasingly integrated into development workflows, they are changing how developers write, debug, and reason about code. While many claims suggest significant productivity gains, there is also growing concern about over-reliance, reduced understanding, and long-term effects on developer skills. This domain is well-suited for a research-focused system because it contains a mix of empirical studies, industry reports, and ongoing debates with incomplete or conflicting evidence.

Main Research Question

How do AI tools affect software engineer productivity, code quality, and skill development, particularly for early-career developers?

This question is intentionally framed to capture both positive and negative impacts while remaining grounded in empirical evidence rather than anecdotal claims. It also aligns well with research tasks that require citation-backed answers and explicit handling of uncertainty.

Sub-Questions

To break down the main research question into retrievable and evaluable components, this project explores the following sub-questions:

1. What productivity gains or losses are empirically measured when software engineers use AI-assisted tools?
2. How does AI assistance affect code quality, correctness, and maintainability?
3. How do the impacts of AI tools differ between junior or early-career developers and senior developers?
4. What risks or negative effects (such as over-reliance or skill degradation) are identified in the literature?
5. Where do studies disagree, or where is evidence inconclusive or limited?

6. What long-term impacts on developer skill formation are suggested but not yet empirically validated?

These sub-questions will later inform retrieval queries and synthesis tasks in subsequent phases of the project.

Scope and Boundaries

This project focuses on evidence related to software engineering tasks such as coding, debugging, testing, and code comprehension. Priority is given to empirical studies, controlled experiments, and reputable industry or academic reports that explicitly evaluate the effects of AI tools on developers.

The scope intentionally excludes:

- Purely opinion-based articles or blog posts without supporting data
- General discussions of AI ethics not tied to software engineering practice
- AI applications outside the context of programming or developer workflows

By narrowing the scope, the project aims to maintain a clear evidentiary standard and ensure that claims can be evaluated against concrete sources.

Purpose and Motivation

The purpose of this research is to better understand how AI tools are reshaping the work of software engineers, especially those early in their careers who may be most influenced by these technologies. As AI-assisted development becomes more common in both educational and professional settings, it is important to distinguish between claims that are well-supported by evidence and those that are speculative or overstated.

This framing also directly supports the goals of the Personal Research Portal by emphasizing traceability, citation correctness, and transparency around uncertainty. The insights gained in Phase 1 will inform the design of a research-grade retrieval and grounding system in Phase 2, and ultimately a usable research portal in Phase 3.

Phase 1 Analysis Memo

Patterns, Failure Modes, and Phase 2 Design Choices

Overview

Phase 1 evaluated how two large language models respond to core research tasks related to AI-assisted software engineering when provided with academic and industry sources. Specifically, we examined model behavior across two tasks (Paper Triage and Claim-Evidence Extraction) using two prompt variants (baseline vs. structured) and two models. Across the 16 runs, clear patterns emerged regarding grounding, overgeneralization, citation behavior, and the impact of prompt structure.

This memo summarizes those patterns, identifies recurring failure modes, and outlines how these findings inform the design of a research-grade RAG system in Phase 2.

Observed Patterns

1. Prompt Structure Significantly Improves Groundedness

Across both tasks and both models, **Prompt B (structured, with explicit guardrails)** consistently outperformed Prompt A on groundedness and citation correctness.

For Paper Triage, Prompt A outputs often produced fluent summaries but blurred distinctions between:

- Measured outcomes vs. inferred implications
- Short-term experimental results vs. long-term productivity or skill claims

Prompt B, by contrast, encouraged models to:

- Separate findings from limitations more clearly
- Avoid speculative language when evidence was limited
- Explicitly acknowledge uncertainty when appropriate

This pattern was especially visible in papers like *How AI Impacts Skill Formation*, where Prompt A frequently overstated conclusions about long-term skill degradation, while Prompt B remained closer to the authors' cautious framing.

2. Models Tend to Overgeneralize Without Explicit Constraints

A recurring failure mode across models was overgeneralization beyond the study population or task. For example:

- Findings from novice programming education studies were sometimes framed as applying to “software engineers” broadly.
- Results from short, controlled tasks (e.g., HTTP server implementation) were interpreted as representative of real-world, long-term developer productivity.

These failures occurred even when the relevant limitations were present in the provided text. This suggests that models prioritize high-level narratives unless explicitly constrained to respect scope.

3. Citation Behavior Is Weak Without Enforcement

In Claim-Evidence Extraction tasks, models frequently produced reasonable claims but failed to:

- Tie those claims to specific textual evidence
- Distinguish between direct evidence and interpretation

Prompt A often resulted in claims that were directionally correct but insufficiently grounded. Prompt B reduced this issue by explicitly requiring citation-style references and discouraging unsupported claims, though minor citation vagueness remained a challenge.

This pattern reinforces that citation correctness is not a default model behavior, even when source material is available.

4. Differences Between Models Were Secondary to Prompt Effects

While some differences between models were observed (such as verbosity or stylistic confidence) the largest performance gains came from prompt design rather than model choice.

Both models exhibited:

- Hallucination under weak constraints
- Improved discipline under structured prompts

This suggests that, for research-oriented tasks, prompt engineering and system design may matter more than marginal model improvements, especially in early-stage research tools

Key Failure Modes Identified

Across the evaluation runs, the following failure modes were most common:

1. **Overgeneralization**
   Claims extended beyond the study’s population, task, or time horizon.
2. **Speculative Interpretation**
   Models inferred long-term impacts (e.g., skill degradation) without explicit evidence.

3. **Missing or Vague Citations**
   Claims were presented without clear links to supporting text.
4. **Blurring Evidence and Opinion**
   Interpretive language was sometimes presented as empirical fact.

These failures were more frequent under Prompt A and less frequent (but not eliminated) under Prompt B.

Implications for Phase 2 Design

The findings from Phase 1 directly inform several design decisions for Phase 2:

1. Structured Prompts as a Default

Phase 2 will rely on explicit, structured prompts that:

- Separate claims, evidence, and interpretation
- Require citation identifiers tied to retrieved chunks
- Encourage models to state when evidence is missing or inconclusive

2. Strong Emphasis on Citation Traceability

Given persistent citation weaknesses, Phase 2 will prioritize:

- Chunk-level retrieval identifiers
- Enforced citation formats
- Logging that makes every claim traceable to a specific source segment

3. Explicit Handling of Scope and Uncertainty

The system should:

- Surface study populations and task constraints in retrieved evidence
- Discourage synthesis that collapses heterogeneous findings
- Explicitly flag areas where the corpus does not support a claim

4. Evaluation Beyond Fluency

Phase 2 evaluation will focus less on surface-level coherence and more on:

- Groundedness
- Evidence coverage
- Faithfulness to source limitations

Conclusion

Phase 1 demonstrates that large language models can assist with research-oriented tasks in software engineering, but only when supported by strong prompt structure and explicit grounding requirements. Without these safeguards, models tend to overgeneralize, speculate, and weaken citation fidelity.

These insights justify the need for a retrieval-augmented, citation-aware system in Phase 2 and provide concrete guidance for its design. By addressing the observed failure modes, the Personal Research Portal can move from fluent summarization toward trustworthy, research-grade synthesis.