Multi Agent Code Review and Debugging Network

Himanshu Jhawar
Department of Computer Science
Columbia University
New York, USA
hj2713@columbia.edu

Anouksha Rajesh
Department of Data Science
Columbia University
New York, USA
ar4829@columbia.edu

Yeshitha Bhuvanesh
Department of Computer Science
Columbia University
New York, USA
yb2649@columbia.edu

***Abstract*—We propose an end to end system, the *Multi Agent Code Review and Debugging Network*, that uses three collaborative LLM based agents to automate software quality assurance. The *Code Reviewer* agent flags logic errors, style violations, and security issues; the *Bug Finder* agent simulates execution to detect potential defects; and the *Refactor* agent suggests safe, maintainable code improvements. The system integrates with GitHub to review real world repositories and output actionable feedback to developers. We will implement and evaluate the system on open source code within a 5 week development cycle, demonstrating alignment with course objectives.**

***Index Terms*—LLM Agents, Code Review, Debugging, Generative AI, Software Engineering, GitHub Integration**

I. Introduction

Code review is a critical yet time consuming step in modern software development aimed at catching bugs and enforcing coding standards [1]. Recent advances in large language models (LLMs) have introduced new AI driven tools that assist developers by automatically analyzing pull requests to flag logic errors, security flaws, and style violations [1], [2]. However, the full potential of LLMs in automated review remains underexplored. The industry is now shifting toward *agentic AI workflows*, where specialized agents plan and act semi autonomously [5], [6].

For instance, GitHub's Copilot vision includes autonomous agents that proactively manage tasks such as bug fixing and code reviews [6]. Inspired by these developments, we propose a network of three cooperative LLM agents *Reviewer*, *Bug Finder*, and *Refactor* to automate code analysis and improvement through a unified pipeline.

II. Related Work

Automated code review and multi agent AI are active research areas. Industry tools such as Graphite integrate AI reviewers into GitHub workflows to analyze pull requests, detect logical errors, security vulnerabilities, and maintainability problems [1].

In academia, Lu et al. (2023) introduce *LLaMA Reviewer*, fine tuning a LLaMA model for review tasks and achieving comparable performance to specialized code review models [3]. Rasheed et al. (2024) propose an LLM based pipeline that identifies bugs, code smells, and suggests optimizations [4]. In the multi agent domain, Pan et al. (2025) present *AgentMesh*, which assigns specialized roles such as Planner, Coder, Debugger, and Reviewer to cooperating LLMs. Despite these advancements, no system yet automates the end to end review, debug, and refactor workflow as our proposed system does.

III. Objectives and Scope

The project aims to evaluate whether a team of specialized LLM agents can outperform a single model in software quality assurance tasks.

A. Research Questions

Can distinct agents focused individually on reviewing, bug detection, and refactoring deliver more comprehensive and accurate insights than a unified model? We will compare issue detection rates and developer evaluations between our multi agent system and baseline tools.

B. Problems Addressed

We target common code quality issues such as logical bugs (e.g., off by one errors), security vulnerabilities (e.g., injection risks), and code smells that reduce maintainability [1]. Additionally, the Refactor agent aims to generate safe code improvements that preserve functionality.

C. Expected Outcomes

Deliverables include:

- A functional prototype integrated with GitHub.
- Multi agent source code and configuration files.
- Evaluation report and case studies on open source repositories.

D. Course Alignment

This project aligns with the "LLM Based Gen AI" course goals by applying generative AI to practical engineering problems involving multi step reasoning, agent orchestration, and prompt design.

IV. Proposed System

The proposed system orchestrates three LLM based agents:

- **Code Reviewer Agent:** Takes code diffs or pull requests as input and flags issues such as incorrect logic, poor naming, or insecure code patterns.
- **Bug Finder Agent:** Simulates code execution or analyzes logic paths to detect runtime errors and edge case bugs.
- **Refactor Agent:** Suggests maintainable code improvements (e.g., variable renaming, function extraction) while preserving functionality.

An orchestrator coordinates these agents sequentially, consolidating their outputs into a unified review report. Each agent's feedback appears as comments on GitHub pull requests.

V. Feasibility, Timeline, and Resources

A. Timeline (5 Weeks)

- **Week 1:** System design, prompt templates, and GitHub integration setup.
- **Week 2:** Implement and test the Code Reviewer agent.
- **Week 3:** Develop the Bug Finder agent and integrate outputs.
- **Week 4:** Implement the Refactor agent with safety verification.
- **Week 5:** Integrate all agents, test on open source repositories, and prepare evaluation.

B. Resources

The system requires:

- Access to an LLM (e.g., GPT 4, Claude, or Llama 2).
- GitHub API tokens and a local environment for orchestration.
- Minimal compute (API calls and lightweight scripts).

C. Risks and Mitigation

- **LLM Hallucination:** Use prompt constraints and self verification.
- **Scope Creep:** Limit focus to Python code reviews.
- **API Limits:** Optimize batching and caching.
- **Integration Complexity:** Modularize agents and follow established webhook workflows.

VI. Innovation and Impact

This project's novelty lies in orchestrating multiple LLM agents each with a specialized role into a collaborative system for end to end code review. The design embodies the agentic AI vision championed by GitHub and AWS, demonstrating how coordinated AI entities can handle complex engineering tasks efficiently [5], [6].

Empirical studies suggest that LLM augmented reviews improve developer efficiency and confidence, particularly for complex codebases [7]. Our system could evolve into an AI powered continuous integration (CI) tool, enhancing code quality and developer productivity at scale.

VII. Conclusion

This proposal outlines the *Multi Agent Code Review and Debugging Network*, detailing its design, objectives, and feasibility. Rooted in emerging research and industry innovation, the project aspires to demonstrate how coordinated LLM agents can meaningfully elevate code quality assurance bridging the gap between theoretical AI capabilities and practical software engineering impact.

Acknowledgment

We thank the course faculty for guidance and the open source developer community for providing repositories for evaluation.

References

[1] Graphite.dev. "AI Code Review Integrations." Available: *https://graphite.dev*

[2] Lu et al., "LLaMA Reviewer: Fine tuned LLM for Automated Code Review," arXiv, 2023.

[3] Rasheed et al., "LLM Based Bug Detection and Refactoring Pipelines," arXiv, 2024.

[4] Pan et al., "AgentMesh: Multi Agent Collaboration for Software Tasks," arXiv, 2025.

[5] AWS Blog. "Building Specialized Agents for Complex Workflows." Available: *https://aws.amazon.com*

[6] GitHub Blog. "The Next Generation of Copilot Agents." Available: *https://github.blog*

[7] Dalsteinsson et al., "AI Augmented Code Review Improves Developer Confidence," arXiv, 2025.