



International Journal of Innovative Research in Science Engineering and Technology (IJIRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 9.935

Volume 15, Issue 10, October 2026

CodePilot AI: A Multi-Agent Autonomous Software Engineering Framework for Mobile-Based Repository Management

Prof. Swati Dronkar¹, Devanshu Mishra², Mayur Nandanwar³, Akshita Rai⁴, Yash Kuhikar⁵,
Pranay Ambule⁶

Department of Computer Science and Engineering, Priyadarshini College of Engineering, Nagpur, India

ABSTRACT: Modern software development often requires developers to access their computers for minor code modifications, bug fixes, and feature updates. This can limit productivity when developers are away from their workstation. CodePilot AI proposes an AI-powered multi-agent mobile software engineering platform that enables developers to interact with GitHub repositories directly from smartphones. Specialized AI agents collaboratively plan, generate, test, and deploy code changes through an automated development pipeline. This review examines the evolution of AI-assisted software engineering, autonomous coding systems, repository-aware AI, and multi-agent architectures for mobile software development. The reviewed systems include GitHub Copilot Workspace, OpenAI Codex, Google DeepMind AlphaEvolve, and Anthropic Claude Code. Their capabilities, findings, and identified research gaps are compared, with particular emphasis on mobile accessibility, repository-level operations, and end-to-end autonomy. The review identifies a research direction that combines AI-assisted coding, repository awareness, multi-agent coordination, automated validation/testing, and mobile-based interaction.

KEYWORDS: Artificial Intelligence, Multi-Agent Systems, Autonomous Software Engineering, Code Generation, GitHub, Repository Management, Mobile Software Engineering, Large Language Models.

I. INTRODUCTION

Software development traditionally requires developers to work with integrated development environments, terminals, version-control systems, testing frameworks, and deployment tools. These tools are highly effective, but they generally assume that developers have access to a desktop or laptop environment. The project presentation identifies this dependency as a limitation when developers need to perform relatively small code modifications, bug fixes, or feature updates while away from their primary workstation.

Another limitation arises from the manual nature of software development workflows. Even relatively simple modifications can require developers to move between multiple tools and environments. Code generation assistants can reduce the effort involved in writing code, but the project presentation notes that existing assistants generally do not perform the complete process of planning, testing, and deployment autonomously.

Recent developments in artificial intelligence have introduced systems capable of understanding source code, generating software, modifying existing repositories, and assisting with software engineering tasks. The literature survey in the project presentation includes systems such as GitHub Copilot Workspace [4], OpenAI Codex [2], Google DeepMind AlphaEvolve [3], and Anthropic Claude Code [1], demonstrating the progression from code-assistance tools toward more autonomous software engineering systems.

The central idea of CodePilot AI is to extend this development by introducing a mobile-first software engineering workflow. Rather than requiring a developer to access a complete desktop development environment, the proposed framework allows a developer to issue natural-language instructions through a mobile application. A coordinated group of AI agents can then interpret the requirement, work with the repository, modify source code, validate the changes, execute tests, and update the GitHub repository. This review therefore focuses on the evolution of AI-assisted software engineering, autonomous coding systems, repository-aware AI, and the potential of multi-agent architectures for mobile software development.

II. AI-ASSISTED SOFTWARE ENGINEERING

Artificial intelligence has increasingly been applied to various stages of software development. AI-based systems can support programmers by generating source code, explaining existing code, identifying errors, navigating software projects, and assisting with modifications.

The project presentation places particular emphasis on the transition from code assistance to autonomous software engineering. Instead of simply generating a code fragment in response to a prompt, newer systems can operate at a broader software-project level by understanding repositories and performing multiple development-related actions.

This progression is important because software development involves more than code generation. A complete development task may require understanding requirements, identifying relevant files, modifying multiple components, checking consistency, running tests, and updating the repository. An intelligent software engineering system therefore needs to operate across several related activities rather than treating code generation as an isolated operation.

III. EVOLUTION OF AUTONOMOUS CODING SYSTEMS

[1] Review of “Anthropic Claude Code”

Developer: Anthropic (2025)

The literature survey identifies Claude Code (2025) as an AI coding assistant capable of understanding software projects and performing autonomous programming. The presentation associates it with improvements in developer productivity through automation of repetitive coding operations and project navigation [1].

The PPT identifies desktop-oriented workflows and the absence of a dedicated mobile-first software engineering workflow as an important gap.

[2] Review of “OpenAI Codex”

Developer: OpenAI (2025)

The PPT identifies OpenAI Codex (2025) as an AI coding agent capable of generating, modifying, and explaining source code. The presentation associates it with strong performance in code generation, debugging, and repository-level software engineering tasks [2].

However, the project literature survey identifies the need for integration with external development environments and indicates that complete autonomous repository management is not provided within the described workflow. This distinction is important because the CodePilot concept aims at a more integrated pipeline extending from user instruction to repository update.

[3] Review of “Google DeepMind AlphaEvolve”

Developer: Google DeepMind (2025)

The PPT describes AlphaEvolve (2025) as an autonomous software optimization framework combining large-language-model reasoning with iterative evaluation. It demonstrated improvements in solving complex programming and optimization problems autonomously [3].

According to the presentation, its primary emphasis is algorithm optimization rather than collaborative software development workflows. This leaves a distinction between autonomous problem-solving and a general-purpose repository management system.

[4] Review of “GitHub Copilot Workspace”

Developer: Microsoft (2025)

The project literature survey identifies Microsoft’s GitHub Copilot Workspace (2025) as a repository-aware AI development system. According to the PPT, the system is capable of planning, implementing, and validating software changes directly from GitHub repositories. It enables natural-language-driven software development with integrated planning and code understanding [4].

The presentation identifies a limitation in that the workflow remains heavily associated with desktop-based software development and still requires developer supervision. This creates an opportunity for research into a more mobile-oriented and autonomous workflow.

IV. COMPARATIVE ANALYSIS

Based strictly on the project PPT, the reviewed systems can be summarized as shown in Table I.

TABLE I: Comparison of AI-Assisted and Autonomous Coding Systems

System	Main capability	Finding described in PPT	Gap identified in PPT
Claude Code [1]	Project understanding and autonomous programming	Automates repetitive coding and project navigation	Lack of dedicated mobile-first workflow
OpenAI Codex [2]	Code generation, modification and explanation	Strong code-generation, debugging and repository-level capabilities	Does not provide complete autonomous repository management
AlphaEvolve [3]	Autonomous software optimization	Iterative AI reasoning and evaluation for programming problems	Focuses on optimization rather than collaborative software development
GitHub Copilot Workspace [4]	Repository-aware development	Natural-language development with planning and code understanding	Desktop-oriented and dependent on developer supervision

These comparisons indicate that modern AI systems increasingly operate beyond simple code completion. Nevertheless, the project presentation identifies gaps relating to mobile accessibility, end-to-end autonomy, and integrated repository management.

Multi-Agent Approach to Software Engineering

A central concept of CodePilot AI is the use of a multi-agent AI architecture rather than relying on one general-purpose AI component.

In a multi-agent system, different AI components can be assigned different responsibilities. For an autonomous software engineering workflow, these responsibilities can correspond to tasks such as understanding requirements, planning changes, modifying source code, validating modifications, and executing tests.

The project presentation describes a coordinated team of AI agents that receives a natural-language instruction from the developer and performs the development workflow collaboratively. The intended sequence is to analyze the developer's request, work with the selected repository, make changes, validate those changes, execute tests, and push verified updates to GitHub.

This architecture can potentially separate responsibilities within the software-engineering workflow and provide a more structured process than a single-agent coding interaction.

Mobile-Based Software Engineering

The primary distinctive feature of CodePilot AI is its focus on mobile-based repository management.

The project presentation identifies mobile software engineering as a problem area in which developers have difficulty efficiently managing or modifying repositories from smartphones without relying on remote-desktop solutions.

The proposed approach uses an Android client containing:

- GitHub authentication
- Repository browsing
- A natural-language interface

This client communicates with a backend orchestration layer, which coordinates the AI processing. Such an approach is intended to provide developers with a simpler interface for initiating software-engineering tasks without requiring direct interaction with a full desktop development environment.

V. PROPOSED CODEPILOT AI ARCHITECTURE

According to the project presentation, CodePilot AI consists of three major layers.

A. Mobile Application

The first layer is the Android client. The PPT specifies Android using Java/Kotlin and identifies GitHub authentication, repository browsing, and a natural-language interface as its primary components. The mobile application serves as the developer-facing component of the system.

B. Backend Orchestrator

The second layer acts as the coordinator between the mobile application and AI engine. The architecture shown in the PPT includes:

- Django REST Framework
- MySQL
- Redis
- Celery
- Docker

This layer is intended to coordinate requests and background processing between the mobile client and the AI components.

C. Multi-Agent AI Engine

The third layer is the AI processing component, described in the PPT as the “brain” of the system. The presented technologies include:

- LangGraph
- LangChain
- Groq API
- Ollama
- GitHub API
- Git

Together, these components form the proposed technological foundation for coordinating AI-based repository operations.

VI. PROPOSED WORKFLOW

The intended workflow described by the project is shown in Fig. 1. The project presentation describes the developer as providing a natural-language instruction through the mobile application, after which a coordinated team of agents plans the task, modifies the repository, validates the code, executes tests, and pushes verified changes to GitHub.

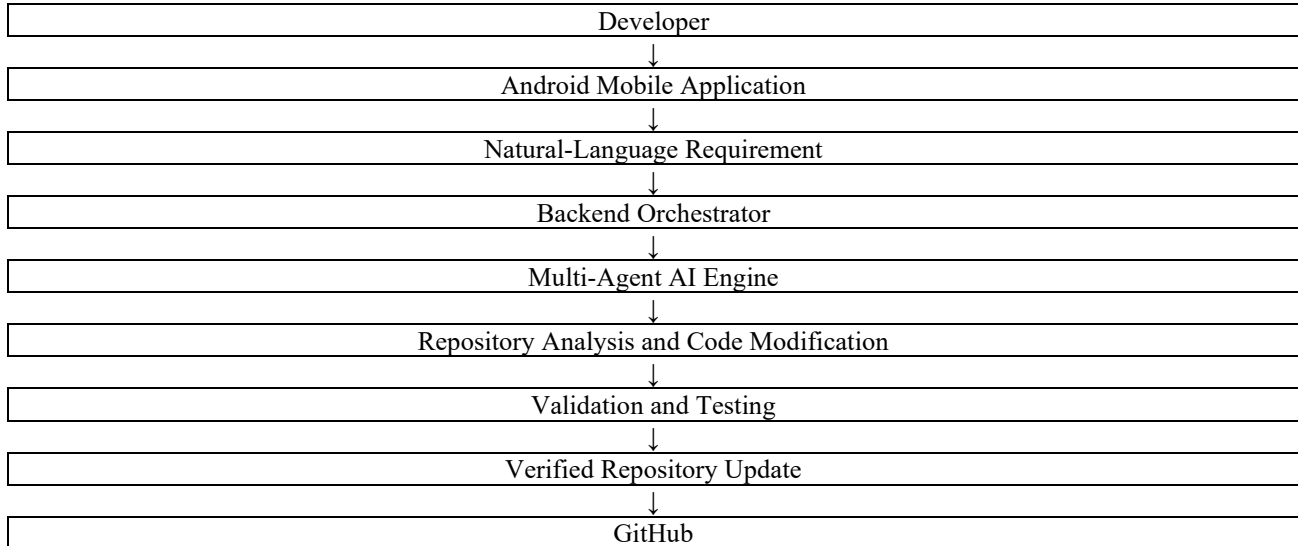


Fig. 1: Proposed CodePilot AI workflow.

VII. RESEARCH GAP

The literature presented in the project demonstrates that significant progress has been made in AI-assisted and autonomous software engineering. However, the reviewed systems reveal several areas that motivate the CodePilot concept.

First, the project presentation identifies a mobile accessibility gap. Existing solutions are primarily associated with desktop development environments, while a dedicated mobile-first software engineering workflow is not emphasized. Second, the reviewed systems do not all provide an integrated process covering the complete lifecycle from natural-language requirement to repository modification, validation, testing, and update.

Third, the project proposes combining these capabilities through a multi-agent architecture, allowing different AI components to participate in different stages of the software development process.

Therefore, the research direction represented by CodePilot AI is the integration of: AI-assisted coding + repository awareness + multi-agent coordination + automated validation/testing + mobile-based interaction.

VIII. DISCUSSION

The literature reviewed in the project presentation demonstrates a clear movement toward increasingly autonomous software engineering. Traditional coding assistants primarily assist individual programming activities, whereas newer systems are capable of understanding projects and performing larger development tasks.

The CodePilot concept builds on this progression by changing the interaction model. Instead of requiring developers to access a desktop environment and manually coordinate multiple tools, the proposed system places the interaction point on a mobile device.

The multi-agent architecture is particularly relevant because software engineering consists of multiple logically distinct activities. Planning, implementation, validation, and testing can be represented as separate stages within an orchestrated workflow. This provides a conceptual basis for constructing an autonomous software-engineering pipeline.

However, a review based on the current project proposal cannot establish actual improvements in productivity, accuracy, execution time, or software quality because the PPT does not provide experimental results from a completed

implementation. Therefore, these aspects should be evaluated in future project development rather than presented as established findings.

IX. FUTURE SCOPE

The project presentation proposes an autonomous development workflow centered around mobile repository management. Based on that scope, future development can focus on improving the coordination between AI agents, expanding repository-level operations, improving automated validation, and increasing the degree of autonomy available to developers.

The proposed architecture can also serve as a foundation for further development of mobile software engineering tools in which developers interact with repositories through natural-language commands rather than directly manipulating files and development tools.

X. CONCLUSION

AI-assisted software engineering is moving from simple code-generation assistance toward systems capable of understanding projects, modifying repositories, and performing broader development activities. The systems reviewed in the CodePilot AI project—Claude Code, OpenAI Codex, AlphaEvolve, and GitHub Copilot Workspace—illustrate different aspects of this transition.

The project presentation identifies mobile access, integrated autonomous workflows, and repository-level operations as areas where further development is possible. CodePilot AI proposes addressing these areas through a multi-agent architecture in which developers provide natural-language instructions through an Android application and AI components coordinate repository analysis, code modification, validation, testing, and GitHub updates.

The proposed architecture consists of a mobile application, backend orchestration layer, and multi-agent AI engine, providing a conceptual framework for mobile-based autonomous software engineering. As the actual implementation is still under development, further experimentation will be necessary to evaluate the practical effectiveness, reliability, autonomy, and usability of the proposed framework.

REFERENCES

1. Anthropic, Claude Code, 2025.
2. OpenAI, Codex, 2025.
3. Google DeepMind, AlphaEvolve, 2025.
4. Microsoft, GitHub Copilot Workspace, 2025.
5. FFmpeg Developers, “FFmpeg Multimedia Framework Documentation,” 2025.
6. PyTorch Foundation, “PyTorch Deep Learning Framework Documentation,” 2025.
7. LangChain/LangGraph, “Multi-Agent Orchestration Framework Documentation.”



INTERNATIONAL
STANDARD
SERIAL
NUMBER
INDIA



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

 9940 572 462  6381 907 438  ijirset@gmail.com



www.ijirset.com

Scan to save the contact details