Omkar Navnath Shingote , 2026, 14:6
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Generative AI in Software Engineering

Omkar Navnath Shingote[1] Suyash Sagar Tajane[2] Shivam Maruti Zaware[3]
Assistant Professor D.S.Bhor[4]

[1,2,3]Undergraduate Student,Electronics & Telecommunication Engineering, Jaihind College Of Engineering Kuran, Maharashtra,India,
[4]Assistant Professor,Electronics & Telecommunication Engineering, Jaihind College Of Engineering Kuran,

Abstract-. Generative Artificial Intelligence (Generative AI) is transforming software engineering by automating tasks such as code generation, debugging, testing, and documentation. It helps developers improve productivity, reduce development time, and create efficient software applications. AI-powered tools assist in various stages of the Software Development Life Cycle (SDLC), making software development faster and more effective. However, challenges such as code accuracy, security, and data privacy require human supervision. This paper explores the applications, benefits, and challenges of Generative AI in software engineering.

Keywords: Generative AI, Software Engineering, Code Generation, Automated Testing, Debugging, Software Development Life Cycle (SDLC).

I. INTRODUCTION

Generative Artificial Intelligence (Generative AI) is an emerging technology that is transforming the field of software engineering. It uses advanced machine learning models, particularly Large Language Models (LLMs), to generate code, analyze software requirements, detect errors, and assist developers in various programming tasks. The integration of Generative AI into software engineering has introduced new opportunities to improve productivity, efficiency, and software quality.

Software engineering involves several stages, including requirement analysis, system design, coding, testing, deployment, and maintenance. Traditionally, these activities require considerable time, effort, and technical expertise. Generative AI helps simplify these processes by automating repetitive tasks and providing intelligent assistance to developers. AI-powered tools such as GitHub Copilot and ChatGPT can generate code snippets, suggest programming solutions, explain complex algorithms, and assist in debugging. These capabilities allow developers to focus more on problem-solving, system architecture, and innovation. One of the major applications of Generative AI is automated code generation, which enables developers to convert natural language instructions into programming code. It can also support software testing by generating test cases, identifying potential bugs, and suggesting improvements. Furthermore, Generative AI can assist in documentation, code optimization, and software maintenance, making the development process more efficient. Despite its advantages, Generative AI presents several challenges. AI-generated code may contain errors, security vulnerabilities, or inefficient logic. Issues related to data privacy, intellectual property, and the reliability of generated outputs also require attention. Therefore, human supervision and proper validation are essential to ensure the development of secure and dependable software systems. Generative AI is not intended to completely replace software engineers but to support their capabilities through intelligent automation. The collaboration between human expertise and AI-driven tools can improve software development practices and reduce development time. This paper focuses on the role of Generative AI in software engineering, its major applications, benefits, limitations, and future possibilities. It highlights how Generative AI can contribute to the development of intelligent, reliable, and efficient software systems.

II. WORKING

Generative Artificial Intelligence (Generative AI) works by using advanced machine learning models

© 2026 1.Omkar Navnath Shingote . his is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly credited.

to understand user instructions, generate programming code, and assist in software development activities. It is trained on large datasets containing programming languages, software documentation, and existing code examples. This training enables AI models to recognize programming patterns and generate relevant solutions based on user requirements.

The working process of Generative AI in software engineering consists of the following stages:

- **Input and Requirement Analysis:** The developer provides instructions in natural language, describing the required software functionality. Generative AI analyzes the input to understand the objectives, programming language, and expected output.
- **Data Processing and Model Understanding:** The AI model processes the input using Natural Language Processing (NLP) techniques. It identifies important information, understands programming concepts, and determines suitable solutions based on its learned patterns.
- **Code Generation:** After understanding the requirements, the AI model generates appropriate programming code. It can create functions, algorithms, application components, and complete code segments using languages such as Python, Java, C++, and JavaScript.
- **Code Testing and Debugging:** The generated code is tested to identify errors and verify its functionality. AI-powered tools can suggest corrections, identify potential bugs, and recommend improvements. However, developers must execute and validate the code to ensure its correctness.
- **Code Optimization and Documentation:** Generative AI can improve code readability, suggest efficient algorithms, and generate technical documentation. It also assists in explaining complex programming concepts and maintaining existing software applications.
- **Human Review and Deployment:** Software engineers review the generated output for accuracy, security, and performance. After successful testing and validation, the code can be integrated into the software system and deployed.

Generative AI follows an iterative process in which developers can provide additional instructions to refine the generated output. This interaction helps improve the quality of the code and ensures that the final software meets the required specifications.
Thus, Generative AI acts as an intelligent assistant throughout the software development life cycle. It reduces repetitive work, accelerates development, and supports efficient software engineering practices while maintaining the importance of human expertise and supervision.

Advantages:

- **Increased Productivity:** Automates repetitive programming tasks and helps developers complete projects faster.
- **Code Generation:** Generates code from natural language instructions, reducing manual coding efforts.
- **Error Detection:** Helps identify programming errors and suggests possible debugging solutions.
- **Automated Testing:** Generates test cases and assists in identifying software defects.
- Reduced Development Time: Accelerates software development by simplifying coding, testing, and documentation.
- **Improved Learning:** Helps beginners understand programming concepts and learn new languages.
- **Code Optimization:** Suggests improvements in code readability, performance, and maintainability.

Disadvantages:

- **Code Inaccuracy:** AI-generated code may contain logical errors or incorrect outputs.
- **Security Risks:** Generated code may introduce security vulnerabilities if not properly reviewed.
- **Data Privacy Issues:** Sharing sensitive source code with AI tools may create privacy and confidentiality risks.
- **Limited Creativity:** AI may struggle with complex problems requiring innovative solutions and human judgment.

- **Dependency on AI:** Excessive reliance on AI tools may reduce developers' independent problem-solving skills.
- **Maintenance Challenges:** AI-generated code may be difficult to understand, modify, or maintain.
- **Ethical and Legal Concerns:** Questions regarding copyright, intellectual property, and ownership of generated code may arise.

III. CONCLUSION

Generative AI is transforming software engineering by improving productivity, automating repetitive tasks, and simplifying software development. It assists developers in code generation, debugging, testing, and documentation, thereby reducing development time and effort. However, challenges such as code accuracy, security risks, and data privacy must be carefully addressed. Human supervision remains essential to ensure software quality and reliability. In conclusion, Generative AI is a powerful tool that supports software engineers and enables the development of efficient, intelligent, and reliable software systems.

REFERENCES

1. Q. Zhang et al., "A survey on large language models for software engineering," Science China Information Sciences, vol. 69, Art. no. 141102, 2026.
2. S. Weng et al., "Data preparation and quality for code-centric generative software engineering tasks: A systematic literature review," Frontiers of Computer Science, vol. 20, Art. no. 2009203, 2026.
3. A. Nguyen-Duc et al., "Generative artificial intelligence for software engineering—A research agenda," Software: Practice and Experience, vol. 55, no. 11, pp. 1806–1843, 2025.
4. J. Wang, Y. Huang, C. Chen, Z. Liu, S. Wang, and Q. Wang, "Software testing with large language models: Survey, landscape, and vision," IEEE Transactions on Software Engineering, vol. 50, no. 4, pp. 911–936, Apr. 2024.
5. "Generative AI in the software development lifecycle," Computer, vol. 58, no. 7, pp. 17–19, Jul. 2025.
6. X. Hou et al., "Large language models for software engineering: A systematic literature review," ACM Transactions on Software Engineering and Methodology, vol. 33, no. 8, Art. no. 220, 2024.