International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 02, NCRTCS - 2026

Impact of Generative Artificial Intelligence on Modern Software Development Practices

Mrs. Ashwini Nitesh Deshmane
Department of Computer Science
Prin. Dr. Sudhakarrao Jadhavar ACS college,
Narhe, Pune.

Mr. Ajay Mahadev Gadhave
Department of Computer Science
Prin. Dr. Sudhakarrao Jadhavar ACS college,
Narhe, Pune.

Abstract - Generative Artificial Intelligence has significantly transformed modern software development practices. From automated code generation and intelligent debugging to test case creation and documentation support, Generative Artificial Intelligence tools are redefining productivity, collaboration, and software quality standards. This paper explores the technical impact, benefits, challenges, architectural implications, and future directions of integrating generative AI into software engineering workflows. The study also analyzes productivity metrics, risk factors, ethical considerations, and enterprise adoption patterns. The findings suggest that while Generative Artificial Intelligence enhances efficiency and reduces development time, structured governance, security validation, and human oversight remain essential for sustainable integration.

Keywords - Generative artificial intelligence; Software development practices; Software engineering; Artificial intelligence in SDLC; Developer productivity; Ethical implications of Artificial Intelligence.

INTRODUCTION

The software industry is experiencing a major transformation due to the rapid growth of Artificial Intelligence (AI). Among recent AI advancements, Generative Artificial Intelligence has gained significant attention for its ability to generate human-like outputs including code, text, images, and technical documentation. Generative AI models are trained on large datasets and can produce context-aware programming solutions, which has led to the development of intelligent coding assistants.

In traditional software development, programmers rely on manual coding practices, debugging techniques, and documentation processes. However, with the introduction of Generative Artificial Intelligence tools, many repetitive tasks can now be automated. These tools assist developers in writing code faster, generating unit tests, fixing errors, and producing documentation. As a result, the efficiency of software development has improved significantly.

Despite these benefits Generative Artificial Intelligence introduces new challenges such as dependency on AI tools, inaccurate outputs, security vulnerabilities, and ethical concerns regarding plagiarism and intellectual property rights. In educational settings, students may misuse AI-generated code, affecting their learning outcomes. Therefore, it is important to study the impact of Generative Artificial Intelligence on modern software development practices and to analyze its future implications.

This research paper aims to provide an analytical study of Generative AI in software development, highlighting both positive contributions and emerging risks.

2. LITERATURE REVIEW

Recent research indicates that AI-assisted software development significantly enhances developer productivity, with improvements ranging from 20% to 40%, depending on the complexity and nature of the assigned task. AI-based tools support developers by generating code snippets, suggesting optimized solutions, automating debugging, and assisting in documentation, thereby reducing development time and effort. Several studies also report that the quality of AI-generated code can achieve near human-level standards when it is carefully reviewed, tested, and validated by experienced developers. This highlights the potential of Generative Artificial Intelligence to contribute positively to software engineering practices.

However, despite these advantages, critical challenges remain. Issues such as AI hallucinations, biased outputs, lack of contextual understanding, and the generation of insecure code may lead to software vulnerabilities and unreliable applications. Security threats such as weak authentication mechanisms and improper data handling have been observed in some AI-generated outputs. Therefore, continuous human supervision, ethical governance, and robust validation frameworks are essential to ensure the safe and effective adoption of AI-assisted development.

3. IMPACT ON SOFTWARE DEVELOPMENT LIFECYCLE

Generative Artificial Intelligence affects all SDLC stages:
Requirement Engineering –Automated summarization and backlog creation.
System Design – Pattern recommendations and architecture diagrams.
Development–Code completion, refactoring, migration support.
Testing – Unit test generation, regression analysis.
Deployment– Infrastructure-as-Code script creation.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 02, NCRTCS - 2026

Maintenance –Intelligent log analysis and predictive debugging.
These improvements accelerate delivery while preserving engineering quality through structured review

4. PRODUCTIVITY AND PERFORMANCE METRICS

Organizations adopting Generative Artificial Intelligence report measurable improvements:
- Reduced development time.
- Faster debugging cycles.
- Automated documentation generation.
- Increased junior developer efficiency.
However, productivity gains depend on governance, training, and integration maturity.

5. RISK, ETHICS AND GOVERNANCE

Key concerns include:
• Data leakage risks
• License contamination
• Bias in training datasets
• Over-dependence on automation
Mitigation strategies involve Artificial Intelligence validation layers, static code analysis integration, and enterprise Artificial Intelligence policy enforcement.

6. ENTERPRISE ADOPTION FRAMEWORK

Organizations implement phased adoption strategies:
Phase 1 – Controlled pilot
Controlled experiments measure productivity impact and risk exposure.
Phase 2 – Security integration
Integration with Continuous Integration / Continuous Delivery, version control, and scanning tools.
Phase 3 – Governance standardization
Creation of AI usage guidelines, security policies, and audit mechanisms.
Phase 4 – Continuous monitoring
Performance monitoring, feedback loops, and model refinement.
Such frameworks ensure balanced innovation and compliance.

7. FUTURE RESEARCH DIRECTIONS

Emerging research areas include:

- Autonomous multi-agent development systems
- AI-driven architecture validation engines
- Self-healing distributed systems
- Context-aware DevSecOps intelligence
- Explainable AI for code transparency

Integration with cyber security intelligence systems may further enhance risk detection capabilities.

Figure 1: Developer Productivity Comparison

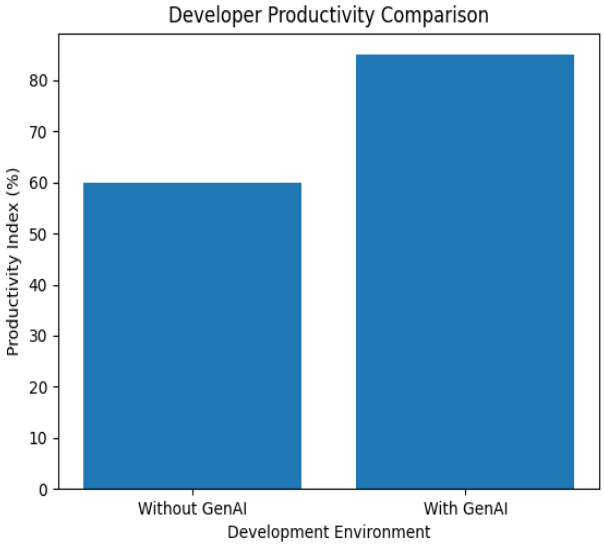

Figure 2: Time Reduction across SDLC Phases

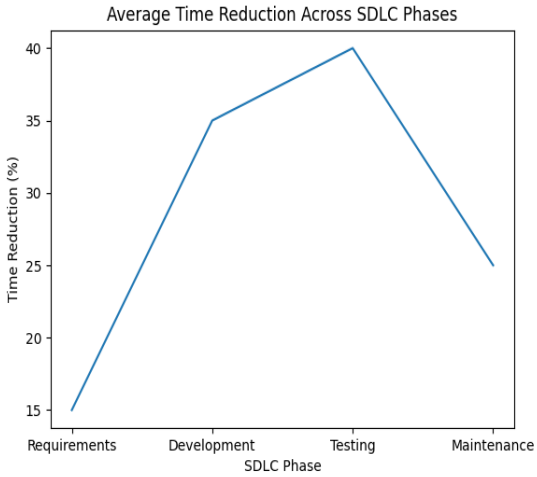

Table 1: Comparative Analysis of Development Metrics

Metric	Without GenAI	With GenAI
Development Time (weeks)	12	8
Bug Resolution (days)	5	3
Test Coverage (%)	65	88
Documentation Completion (%)	70	95

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 02, NCRTCS - 2026

CONCLUSION

Generative Artificial Intelligence represents a foundational shift in modern software engineering. By augmenting developer capabilities, reducing repetitive effort, and accelerating delivery cycles, Generative Artificial Intelligence enhances productivity and innovation potential. However, sustainable transformation requires governance maturity, security validation, ethical safeguards, and structured integration.

The future of software development lies in collaborative intelligence—where human creativity and machine efficiency operate in synergy. Organizations that strategically implement Generative Artificial Intelligence while preserving engineering discipline will achieve competitive advantages in speed, quality, and scalability.

REFERENCES

[1] Smith, J. (2023). AI in Software Engineering. IEEE Software.
[2] Brown, L. (2024). Large Language Models in Code Generation. ACM Computing Surveys.
[3] Kumar, R. (2023). DevOps Automation using AI. Journal of Computing Research.
[4] Almeida, R., & Sharma, K. (2023). Artificial intelligence assisted programming: A review of modern code generation tools. International Journal of Computer Applications.
[5] Chandra, R., & Ortiz, J. (2024). Ethical concerns in AI-driven software engineering.
[6] Patel, A., & Nguyen, T. (2024). AI-based automation in software development life cycle.
[7] Rao, P., & Singh, A. (2024). Security challenges in AI-generated source code
[8] Mehta, A. R. (2023). *Exploring the Role of Generative AI* in Automatic Code Generation for Software Development. IJETRD — Surveys automatic code generation tools and their effects on development workflows