© JUN 2026 | IRE Journals | Volume 9 Issue 12 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I12-1718756

Software Engineering in the Age of Artificial Intelligence

ABUBAKAR BELLO BADA[1], SIRAJO ABDULLAHI BAKURA, PHD[2], FAROUK MOUSA ALIYU[3], ABDULSALAM IBRAHIM MAGAWATA[4]

[1, 2, 3, 4]*Computer Science Department, Federal University Birnin-Kebbi, Kebbi State – Nigeria.*

Abstract- ***Artificial Intelligence (AI) has emerged as a transformative force across multiple sectors, including software engineering, where its integration represents a significant paradigm shift in how software systems are conceived, developed, managed, and maintained. The adoption of AI technologies is reshaping industry practices, redefining developer roles, and altering required skill sets. This paper examines the multifaceted impact of AI on software engineering, with particular emphasis on improvements in development efficiency, software quality, project management, and the democratization of software development. Using a conceptual and analytical approach grounded in existing literature and practical observations, the study explores how AI-driven technologies, such as Generative AI, Large Language Models, and intelligent automation, are transforming key stages of the software development lifecycle. The analysis reveals that AI-enhanced tools significantly improve productivity, reduce time-to-market, and enable innovative development practices, including automated code generation, intelligent testing and debugging, adaptive systems, and broader participation by non-technical stakeholders. However, the findings also highlight critical challenges associated with AI adoption, including over-reliance on automated systems, ethical and legal concerns, security vulnerabilities, and the potential erosion of core software engineering skills. The paper concludes that AI is not replacing software engineers but augmenting their capabilities, necessitating a balanced and responsible integration that combines human oversight, ethical governance, and continuous skill development to fully realize the benefits of AI-driven software engineering.***

Keywords: Software Engineering, Artificial Intelligence, GenAI, LLM

I. INTRODUCTION

Software engineering, the backbone of modern technological development and a discipline that is focused on the systematic design, development, and maintenance of software systems, has undergone significant transformations over the decades. Most recently, it is undergoing transformations due to advancement in Artificial Intelligence [1]. Artificial Intelligence, which is focused on creating machines that can exhibit intelligence, has had significant influence in different sectors e.g., healthcare, finance, and entertainment. In healthcare, AI is now been used for diagnostic purposes, personalized treatment, and drug discovery. In financial sector, AI is used to detect fraud and asses risk among other uses [2]. The integration of Artificial Intelligence (AI) into software engineering represents a pivotal shift and has transformed software engineering field [3]. The coming together of AI and Software engineering has given rise to a collaborative relationship where intelligent systems augment and accelerate traditional development processes, pushing the industry into an era of unprecedented innovation. With the advancement of AI algorithms such as Deep Learning, the role of AI in software engineering has gone beyond code completion suggestion to automating routine tasks e.g., software testing, requirement extraction, and maintenance, and proposing solutions to complex programming challenges [3], [4]. The impact of AI on Software Engineering is driven by some AI innovations e.g. Generative AI (GenAI), Large Language Models (LLM), and Reinforcement learning models. These technologies represent an important shift in how software is conceived, developed, delivered, and maintained. Generative AI, a subset of AI capable of creating new content, has emerged as a powerful tool in code generation and creative problem solving. This technology has the potential to automate significant portions of the software development process, from writing regular code to suggesting complex algorithms. LLM like ChatGPT have capabilities in understanding and generating human-like text, including programming languages. These models can understand context, generate relevant code snippets, and even explain complex programming concepts. The ability to optimize decision-making processes by Reinforcement learning models is helping to create more adaptive and efficient project

© JUN 2026 | IRE Journals | Volume 9 Issue 12 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I12-1718756

management methodologies in software engineering [2], [5], [6]. This paper examines the current state of software engineering in the AI era, detailing the impact of AI on software engineering, challenges faced by software engineering in an AI era and strategies to overcome them, and future prospects of software engineering.

II. THE IMPACT OF AI ON SOFTWARE ENGINEERING

The integration of AI technologies has profoundly affected various aspects of software development, revolutionized traditional processes and introduced new paradigms. From automating code generation to enabling predictive maintenance, AI-powered tools are enhancing productivity, reducing development time, and improving software quality [7]. As AI continues to evolve, it not only accelerates traditional software engineering practices but also enables entirely new paradigms, such as adaptive systems that learn and evolve post-deployment. This intersection of AI and software engineering is not merely a trend but a pivotal shift, heralding unprecedented levels of innovation and efficiency in the software development lifecycle [8], [9], [10]. By integrating the capabilities of AI tools, software development is not only streamlined, but also redefined in terms its quality and efficiency. It has enhanced the quality of software by ensuring higher quality of code through identifying potential issues early. Some of the impact of AI on Software Engineering include:

2.1. Enhanced Code Generation

AI systems are transforming software development by automating coding process. They are capable of translating instructions from human language into executable codes. Large Language Models (LLM) such as ChatGPT and GenAI tools such as INCODER, CODEGEN, AlphaCode, CanvaCode, etc., can generate codes base on natural language input using sophisticated algorithms [11], [12]. Software development process is enhanced by the use of AI tools for code generation thereby reducing the effort and time required for coding and increasing productivity. AI algorithms can suggest solutions, generate code snippets, entire functions, and even complete software modules based on natural language descriptions, design specifications, or existing codebases [1], [13]. This can be done using AI-powered tools such as GitHub Copilot and ChatGPT. AI-powered models, such as those based on natural language processing (NLP), can translate plain-language requirements into functional code, thereby bridging the gap between technical and non-technical stakeholders [11]. This minimizes development time and allows engineers to focus on higher-order problem solving. It has also enabled faster and more efficient creation of high-quality code, for instance, context-aware code suggestions can reduce syntax errors, especially in large codebases. Additionally, automated refactoring and optimization can result in more efficient code that is easier to maintain and scale.

2.2. Improved Testing and Debugging

AI brought improved testing and debugging of software applications by automating complex tasks and reducing testing time. AI tools use algorithms to generate test cases, execute tests, and analyze the results; this significantly reduces testing time and efforts [14]. Furthermore, AI tools can predict areas that have the potential for failure by identifying patterns and analyzing data [15]. AI tools that use techniques like Natural Language Processing (NLP) can be used to create test cases directly from requirements written in natural language [11], [16]. This gives the non-technical stakeholders the ability to contribute to the testing of software.

AI has contributed greatly in improving software testing and debugging through its ability to analyze large amount of test data in order to understand the pattern and identify or predict potential defects. By its nature, AI can help in understanding complex software requirements and converting them into test cases [17]. This is especially useful in agile development environments where requirements frequently change, and maintaining updated test cases manually can be challenging. Some modern software have a lot of lines of codes and analyzing and fixing bugs by humans may be difficult in this kind of situations, this is where AI plays an important role because of its ability to manage complicated software applications [18]. AI learns from its own testing and debugging mistakes, which makes it better in terms of accuracy and efficiency [19]. This helps in enhancing the quality,

speed, and efficacy of software testing and debugging. AI Tools like DeepCode, Diffblue, and SonarQube use machine learning to detect issues early in the development lifecycle, reducing time-to-market and maintenance costs.

2.3. Intelligent Project Management

Software project management is a complex and multifaceted task that is sometimes associated with problems like human error, lack of proper resource allocation, and delay in meeting timelines. Artificial Intelligence enhances different aspects of software project management leading to achievement of better resource allocation, effective risk identification and mitigation, and accurate timeline prediction by the software team [20]. AI tools optimize software project management through predicting resource requirements, proactive risk identification and mitigation, and allocating tasks based on team strengths and past performance [21]. For instance, AI tools like Asana, ClickUp, Resource Guru, and Float can recommend the optimal allocation of resources based on project requirements, team skills, and past project performance, thereby reducing resource wastage and improving overall productivity. AI tools can utilize Machine Learning algorithms to analyze previous project data and current condition of the project to provide a better and more accurate forecast of project timelines [22]. Tools like Notion, Basecamp, and Jira can be used for this and their functionalities lead to anticipation of potential project delays and adjustment of plans accordingly.

2.4. Adaptive and Self-Healing Systems

Adaptive and self-healing systems are systems that are capable of detecting performance anomalies, identifying underlying issues, and automatically making corrections to improve performance without requiring human intervention [23]. These systems monitor their output and external environment continuously, and when a degradation in performance is detected, corrective actions that may or may not include model adjustment and retraining are initiated [24]. AI facilitates the development of these adaptive and self-healing systems that can monitor their own performance and resolve issues autonomously. For example, AI-driven observability tools like Dynatrace use anomaly detection to enhance system reliability while Mabl automatically identifies and adapts to changes in web applications during testing thereby minimizing test maintenance. Anomaly detection systems on the other hand monitor data streams to identify unusual patterns and initiate corrective actions like what happens in some AI-powered network security systems that detect and respond to cyber threats [25].

2.5. Democratization of Software Development

Artificial Intelligence has brought democracy to software engineering by democratizing software development. Democratization of software development is the process of making software development accessible to everybody including the non-technical persons [5]. AI is making software development more accessible to wider populace by lowering the entry barrier for non-programmers. This is done by no-code and low-code platforms like Bubble and OutSystems that allow people with less technical expertise to create applications and contribute to software development process [26]. Another phenomenon that allows democratization of software development is the integration of generative AI (GenAI) into software development, which gives rise to “vibe coding”. This “vibe coding” democratizes software development by allowing users to express functionality in natural language while the AI tool generates the code [27], [28], [29]. This enables individuals with no programming skills to develop a full software application.

Table I: Summary of AI impact on Software Engineering

Area of Impact	AI Contribution	Tools/Examples
Code Generation	Automates coding, suggests functions, reduces syntax errors	GitHub Copilot, ChatGPT, AlphaCode
Testing & Debugging	Automated test cases, defect prediction, pattern recognition	DeepCode, SonarQube, Diffblue

Project Management	Resource allocation, timeline prediction, risk mitigation	Asana, Jira, Notion
Adaptive Systems	Self-healing, anomaly detection, automated corrections	Dynatrace, Mabl
Democratization	Low-code/no-code platforms, "vibe coding"	Bubble, OutSystems

III. CHALLENGES OF SOFTWARE ENGINEERING IN THE AI ERA

The integration of AI in software engineering, from requirement gathering to coding represents an important shift in how software is created. This can increase productivity of software engineers and improve software quality and innovation. However, this also brings new challenges that need to be addressed by the industry. The challenges include:

3.1. Bias in AI Systems

AI models are not always objective; they are trained on datasets that are curated and fed to them by engineers (humans). Since humans select, clean, and label these datasets, the models are susceptible to human-imposed biases, which can manifest in software outputs. As such, bias models can propagate or even amplify discrimination within software systems [30], [31]. For example, large language models might favor certain coding conventions over others, limiting diversity in solutions. The sources of such bias can be broadly classified into three categories: Data bias, such as gender dominance in variable naming or pronoun usage, and preferential treatment of certain languages. Algorithmic bias, for instance, optimization processes that inadvertently prioritize less common tasks or skew toward certain problem domains, and Human bias, including confirmation bias and other cognitive distortions introduced, often unconsciously, by developers during dataset preparation, model tuning, or evaluation. Recognizing and mitigating these intertwined sources of bias is essential for ensuring fairness, transparency, and inclusivity in AI-assisted software development.

3.2. Over-Reliance on AI

Over-reliance on AI-generated software engineering tasks is a great concern. Software engineers can be tempted to trust every output given by AI tools without proper review or good understanding of the underlying logic; this can lead to vulnerabilities [13]. Such over-reliance also risks skill erosion among developers. If AI tools automate too many fundamental software engineering tasks, human developers might lose proficiency in core skills, leading to them becoming less capable of independent problem-solving and other complex software engineering tasks.

3.3. Security Risks

AI-generated code may introduce vulnerabilities that are not immediately apparent, this vulnerabilities may expose the system to attacks in which inputs are manipulated [1]. These attacks have been gaining popularity especially in GenAI. These AI-generated codes also pose privacy risks; this is because the model producing them train on sensitive datasets, potentially exposing personal or proprietary information. The codes generated may include restrictive licenses conflicting with business interests [32].

AI tools can help in streamlining deployment by generating configuration files and infrastructure-as-code (IaC) templates, improving speed, consistency, and reducing manual errors. However, risks include insecure defaults, exposure of sensitive data, weak access controls, and misalignment with security policies [33].

3.4. Ethical and Legal Concerns

The evolution of AI from theoretical concept to practical tool that is capable of automating software development tasks like testing is very good but this power must be used responsibly to ensure ethical and legal compliance in its usage to create software.

Automation of some software tasks by AI may lead to loss of those jobs by humans, furthermore,

autonomous AI software tend to take decision-making authority away from humans [17], [34]. AI software that are unethical can violate people's privacy and/or discriminate against vulnerable communities [35]. There is also the potential of open sources license violations, copyright infringement (because generated code may replicate protected material, and that can lead to legal consequences for the organizations) [36]. Developers and industry managers must take into account all these ethical and legal challenges and make sure AI systems strike a balance between autonomy on one hand and respect for people's right, law, and job security on the other.

IV. STRATEGIES TO OVERCOME CHALLENGES

Just like any new technology, AI has its challenges in software engineering, and depending on the challenge, there are many ways to overcome it. In order for software engineers to keep up with the rapid development of AI, organizations and individuals must commit to ongoing learning. Both academic institutions and industries must equip developers with skills needed for AI-enhanced development [37]. Learning should be integrated into daily workflows of software engineers and the learning should focus on AI-related skills such as prompt, critical-thinking, and AI tool integration [9].

Regulatory body needs to be established that will govern the use of AI in software engineering. Its regulatory function should cover but not limited to data privacy, model training, and decision-making process; it should also establish strong ethical guidelines and best practices. The body should develop tools that will audit AI systems to ensure compliance with regulatory standards and implement frameworks for responsible AI use that will address bias, accountability, and transparency [19], [30], [38]. Software engineers need to participate in ethical standards and certification for AI practices that conform to the established guidelines; this will help in avoiding ethical violation by AI tools in software development [39]. Developing models in which AI complements human expertise and creativity rather than replacing them in their work should be encouraged by the regulatory bodies.

AI-generated codes can have vulnerabilities, therefore, all AI-generated codes should be subjected to rigorous review process, all inputs should be validated to avoid injection attack, and AI should be used for vulnerability detection. There should be continuous monitoring to detect anomalies and a process for rapidly patching vulnerabilities should be established [13].

To address the job displacement threat of AI in software engineering, a focus must be made on reskilling and upskilling software developers and promotion of human-AI collaboration [40], [41]. Software developers must participate in training programs that will equip them with skills needed to adapt to the AI era.

V. FUTURE PROSPECTS OF AI IN SOFTWARE ENGINEERING

The future of AI in software engineering is a promising one; moving from a mere concept to a real partner in the development process. This partnership is transformative, with AI reshaping development, testing, deployment, and maintenance. It will be characterized by increased automation, enhanced efficiency, and a shift in focus for developers towards more creative and strategic tasks [42]. AI is not expected to replace software engineers in software engineering but rather augment their capabilities and change their roles and skill sets.

With AI, the role of software developer in software engineering will shift from a pure coder to a higher-level architect and strategic problem-solver. Since AI will be handling some planning and most of the implementation details, human engineers will focus more on creative innovation and critically guiding and validating AI-generated solutions [43].

The future of software engineering is marching towards an AI-driven development life cycle (AI-DLC) in which AI manages almost everything from requirement gathering to deployment.

VI. CONCLUSION

The integration of AI in software engineering, just like in other sectors, represents a paradigm shift, providing exciting opportunities and significant challenges at the same time. It has great potential for revolutionizing how software is conceived and created; leading to more sophisticated and innovative software.

To achieve the immense potential of AI in software engineering, a careful and deliberate approach is necessary. It lies in striking a balance between using the capabilities of AI and maintaining a good human oversight, critical-thinking, and human creativity. AI has come to stay and the future of software engineering in the age of AI is not one where AI replaces humans, but rather, it shifts the skill requirements of the sector. AI in software engineering creates a kind of symbiotic relationship between software developers and AI tools where they complement each other's strength leading to each achieving a potential that is impossible to achieve in isolation. This symbiotic relationship means that both software developers and the industry have a significant role to play in the future of software engineering in the age of AI. Software developers need to adapt to new ways of working by acquiring new technical skills e.g. AI integration skills, prompt engineering, AI output validation, Ethical AI implementation, and have a shift in mindset towards collaborating with AI systems. Industry on the other hand should not look at AI as a cost-cutting tool but rather an enhancement tool that will lead to a highly augmented and innovative workforce. Industries need to focus on re-skilling initiatives of their workforce and redefining their roles according to the new technical skills.

As the field of AI in software engineering continues to grow, it becomes imperative for organizations, policy-makers, and developers to design a future for the industry where AI enhances human potentials in software development rather than replacing them.

Declaration of generative AI and AI-assisted technologies in the manuscript preparation process
During the preparation of this work the author(s) used ChatGPT in order to polish the written article and improve its readability. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the published article.

REFERENCES

[1] K. Garg, "Impact of Artificial Intelligence on software development: Challenges and Opportunities," *International Journal of Software & Hardware Research in Engineering (IJSHRE)*, vol. 11, pp. 1–3, 2023, doi: 10.26821/IJSHRE.11.8.2023.110801.

[2] H. Sheikh, C. Prins, and E. Schrijvers, "Artificial Intelligence: Definition and Background," pp. 15–41, 2023, doi: 10.1007/978-3-031-21448-6_2.

[3] K. Bhandari, K. Kumar, and A. L. Sangal, "Artificial Intelligence in Software Engineering: Perspectives and Challenges," *ICSCCC 2023 - 3rd International Conference on Secure Cyber Computing and Communications*, pp. 133–137, 2023, doi: 10.1109/ICSCCC58608.2023.10176436.

[4] F. A. Batarseh, R. Mohod, A. Kumar, and J. Bui, "The application of artificial intelligence in software engineering: a review challenging conventional wisdom," *Data Democracy: At the Nexus of Artificial Intelligence, Software Development, and Knowledge Engineering*, pp. 179–232, Jan. 2020, doi: 10.1016/B978-0-12-818366-3.00010-1.

[5] Akhilesh Gadde, "Democratizing Software Engineering through Generative AI and Vibe Coding: The Evolution of No-Code Development," *Journal of Computer Science and Technology Studies*, vol. 7, no. 4, pp. 556–572, May 2025, doi: 10.32996/JCSTS.2025.7.4.66.

[6] st Xiaoxue Ren, nd Xinyuan Ye, and th Xiaohu Yang, "From Misuse to Mastery: Enhancing Code Generation with Knowledge-Driven AI Chaining 4 th Zhenchang Xing".

[7] S. Roobini, M. Kavitha, H. Deenadayalan, and A. Muthusamy, "AI-Powered Tools to Enhance the Stages of Software Development," *https://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.401*

8/979-8-3693-9356-7.ch017, pp. 435–478, Jan. 2025, doi: 10.4018/979-8-3693-9356-7.CH017.

[8] A. Khan and A. A. A. Jilani, "From Algorithms to Intelligence: The Historical Perspective of AI in Software Development," *https://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/979-8-3373-0370-3.ch001*, no. 3, pp. 1–56, Jan. 2025, doi: 10.4018/979-8-3373-0370-3.CH001.

[9] L. Bailey and F. J. Looft, "The Impact of AI on Software Development," 2024. [Online]. Available: https://www.wpi.edu/project-based-learning/project-based-education/major-qualifying-project

[10] M. Alenezi and M. Akour, "AI-Driven Innovations in Software Engineering: A Review of Current Practices and Future Directions," *Applied Sciences 2025, Vol. 15, Page 1344*, vol. 15, no. 3, p. 1344, Jan. 2025, doi: 10.3390/APP15031344.

[11] J. K. Nirmal and M. Sreejith, "Enhancing Natural Language to Code Generation in the SantaCoder Model through In-Context Learning," Jun. 17, 2024. doi: 10.20944/preprints202406.1105.v1.

[12] X. Ren, X. Ye, D. Zhao, Z. Xing, and X. Yang, "From Misuse to Mastery: Enhancing Code Generation with Knowledge-Driven AI Chaining," Sep. 2023, [Online]. Available: http://arxiv.org/abs/2309.15606

[13] H. W. Marar, "Advancements in software engineering using AI," *Computer Software and Media Applications*, vol. 6, no. 1, p. 3906, Feb. 2024, doi: 10.24294/csma.v6i1.3906.

[14] A. Ramadan, H. Yasin, and B. Pektaş, "The Role of Artificial Intelligence and Machine Learning in Software Testing," 2023.

[15] Z. Zong and Y. Guan, "AI-Driven Intelligent Data Analytics and Predictive Analysis in Industry 4.0: Transforming Knowledge, Innovation, and Efficiency," *Journal of the Knowledge Economy*, vol. 16, no. 1, pp. 864–903, May 2024, doi: 10.1007/S13132-024-02001-Z/METRICS.

[16] T. Bazzan *et al.*, "Analysing the Role of Generative AI in Software Engineering-Results from an MLR," 2024.

[17] A. Gu *et al.*, "Challenges and Paths Towards AI for Software Engineering," Mar. 2025, Accessed: Jun. 18, 2025. [Online]. Available: https://arxiv.org/abs/2503.22625v1

[18] G. Umang, "Exploring the Use of Artificial Intelligence for Software Testing and Debugging." Accessed: Aug. 01, 2025. [Online]. Available: https://www.researchgate.net/publication/377816057_Exploring_the_Use_of_Artificial_Intelligence_for_Software_Testing_and_Debugging

[19] B. C. Dolores and M. Serrano, "The Integration and Impact of Artificial Intelligence in Software Engineering," *International Journal of Advanced Research in Science, Communication and Technology (IJARSCT) International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal*, vol. 3, no. 2, 2023, [Online]. Available: https://www.researchgate.net/publication/383455349

[20] S. Georgiev, Y. Polychronakis, S. Sapountzis, and N. Polychronakis, "The role of artificial intelligence in project management: a supply chain perspective," *Supply Chain Forum*, 2024, doi: 10.1080/16258312.2024.2384823.

[21] R. K. Reddy, "The Role of Artificial Intelligence In Project Management For Software Engineering," *IJSART*, vol. 10, no. 7, Jul. 2024, [Online]. Available: www.ijsart.com

[22] A. M. Felicetti, A. Cimino, A. Mazzoleni, and S. Ammirato, "Artificial intelligence and project management: An empirical investigation on the appropriation of generative Chatbots by project managers," *Journal of Innovation and Knowledge*, vol. 9, no. 3, Jul. 2024, doi: 10.1016/j.jik.2024.100545.

[23] E. al. Nand Kumar, "Self-Healing Networks AI-Based Approaches for Fault Detection and Recovery," *Power System Technology*, vol. 47, no. 4, pp. 371–386, Dec. 2023, doi: 10.52783/pst.206.

[24] A. Rajuroy and M. Emmanuel, "Adaptive Self-Healing AI Models for Enhancing SELFI Performance with Real-Time Retraining Detection," Feb. 2025. [Online]. Available:

https://www.researchgate.net/publication/389073861

[25] P. Rauba, N. Seedat, K. Kacprzyk, and M. Van Der Schaar, "Self-Healing Machine Learning: A Framework for Autonomous Adaptation in Real-World Environments," 2023.

[26] M. Ilesanmi, "Introduction to No-Code AI Platforms and Their Role in AI Democratization." Accessed: Jun. 17, 2025. [Online]. Available: https://www.researchgate.net/publication/392501733_Introduction_to_No-Code_AI_Platforms_and_Their_Role_in_AI_Democratization

[27] M. Stoops, P. A. A. Calderón, and Ó. M. Peña Bañuelos, "The Democratization of Software Development and Design Thinking," *Innovative Design Thinking Approaches in Software Engineering*, pp. 111–140, Jun. 2025, doi: 10.4018/979-8-3693-9531-8.CH005.

[28] S. H. Maes, "The Gotchas of AI Coding and Vibe Coding. It's All About Support And Maintenance," 2025, doi: 10.5281/zenodo.15343349.

[29] P. P. Ray, "A Review on Vibe Coding: Fundamentals, State-of-the-art, Challenges and Future Directions," *Authorea Preprints*, May 2025, doi: 10.36227/TECHRXIV.174681482.27435614/V1.

[30] K. Peter, "The Use of AI in Software Engineering_ A Synthetic Knowledge Synthesis of the Recent Research Literature," 2024.

[31] N. Upadhyaya, "The Role of Artificial Intelligence in Software Development: A Literature Review," 2023, doi: 10.13140/RG.2.2.12291.92965.

[32] B. William and B. Williams, "REGULATORY APPROACHES TO AI-GENERATED CONTENT AND PRIVACY PROTECTION," 2025, Accessed: Aug. 13, 2025. [Online]. Available: https://www.researchgate.net/publication/393412916

[33] S. Jayakumar, "Security Risks using AI in Software Development: Mitigation Strategies & Best Practices." Accessed: Aug. 13, 2025. [Online]. Available: https://www.opsmx.com/blog/security-risks-of-ai-in-software-development-what-you-need-to-know/

[34] I. D. S. Venkata, R. Varaprasad, Y. Rama Mohan, T. Aditya Sai Srinivas, and Y. Sravanthi, "A Review on New Challenges in AI and Software Engineering," *International Journal of Advanced Research in Science, Communication and Technology*, pp. 34–42, Oct. 2022, doi: 10.48175/IJARSCT-7137.

[35] A. Mohammad and B. Chirchir, "Challenges of Integrating Artificial Intelligence in Software Project Planning: A Systematic Literature Review," *Digital*, vol. 4, no. 3, pp. 555–571, Jun. 2024, doi: 10.3390/DIGITAL4030028.

[36] B. Li, C. Liu, L. Fan, S. Chen, Z. Zhang, and Z. Liu, "Open Source, Hidden Costs: A Systematic Literature Review on OSS License Management," *IEEE Transactions on Software Engineering*, 2025, doi: 10.1109/TSE.2025.3586411.

[37] G. Tomasz, "The impact of AI on software development: opportunities and challenges," 2024. Accessed: Aug. 04, 2025. [Online]. Available: https://www.future-processing.com/blog/the-impact-of-ai-on-software-development-opportunities-and-challenges/

[38] P. Jain, "Interaction between Software Engineering and Artificial Intelligence-A Review," 2023.

[39] SummaVerse, "Addressing Job Displacement Due to AI: Solution and Strategy." Accessed: Jul. 09, 2025. [Online]. Available: https://summaverse.com/blog/addressing-job-displacement-due-to-ai-solution-and-strategy

[40] A.-O. Tonye, "AI and Workforce Displacement: Strategies for Mitigation in Nigeria." Accessed: Jul. 09, 2025. [Online]. Available: https://www.researchgate.net/publication/384079821_Title_AI_and_Workforce_Displacement_Strategies_for_Mitigation_in_Nigeria

[41] F. Robert, "The Impact of AI on Job Roles, Workforce, and Employment: What You Need to

Know." Accessed: Jul. 09, 2025. [Online]. Available: https://www.innopharmaeducation.com/blog/the-impact-of-ai-on-job-roles-workforce-and-employment-what-you-need-to-know

[42] Bryan, "Will AI Replace Software Engineers? Exploring the Future of Software Development." Accessed: Aug. 04, 2025. [Online]. Available: https://onlinecs.baylor.edu/news/will-ai-replace-SWEs

[43] J. M. Britney and R. R. Adam, "The Impact of AI on the Software Engineering Job Market: Threats and Opportunities." Accessed: Aug. 04, 2025. [Online]. Available: https://www.researchgate.net/publication/390209369_The_Impact_of_AI_on_the_Software_Engineering_Job_Market_Threats_and_Opportunities