УДК 004.8
Valerii Kozlov[1], Nataliia Zhukova[2]
[1] PhD student NU "Zaporizhzhia Polytechnic"
[2] PhD (Philology), Associate Professor NU "Zaporizhzhia Polytechnic"

USING AI MODELS FOR CODE GENERATION

Artificial intelligence (AI) has become an integral part of human life. It has significantly transformed various industries, including hardware and software development. One of the most notable advancements is using AI models for code generation, which helps developers write, optimize, and debug code faster and more efficiently. Tools based on AI models such as ChatGPT, Claude AI, and Copilot have gained popularity and widespread adoption, offering real-time code suggestions, code completion, and even complete method or script generation. These tools enhance developers' productivity by reducing manual and repetitive coding tasks and assisting with complex and composite logic.

ChatGPT, developed by OpenAI, is a versatile code-generation assistant capable of generating code snippets and explaining basic programming concepts. It can be used interactively to answer coding-related requests and suggest solutions for various programming tasks. Claude AI, developed by Anthropic, is a more efficient tool for code generation. It functions as an AI assistant, providing code generation and explanations while emphasizing safety and user control. GitHub Copilot, developed by GitHub, OpenAI, and Microsoft, is designed to integrate with integrated development environments (IDE) such as Visual Studio and Visual Studio

Code. It extends IDE functionality by offering AI-powered file context analysis, suggesting entire lines or blocks of code. This makes the tool especially useful for writing boilerplate code and implementing standard methods and algorithms.

These AI-powered tools have significantly changed software development, allowing programmers to work more efficiently while focusing on high-level problem-solving. AI models help reduce errors and improve code quality by offering syntax corrections, best practices, and debugging assistance. AI-powered code generation also provides junior developers with a good educational tool by explaining complex programming patterns and concepts and suggesting optimized solutions [1]. These tools help developers focus on innovation rather than spending time on routine coding or repetitive manual work.

Despite all the advantages, code generation using tools based on AI models presents several challenges and ethical issues.

The first issue is code quality and reliability. Code generated using AI-powered tools is not always optimized [2] and can sometimes contain logical errors [3]. Developers must carefully review automatically generated code to ensure its correctness and efficiency.

The second significant concern is intellectual property and licensing issues. Most AI models are trained on open-source code, raising questions about copyright compliance. This leads to concerns about proper code attribution and ownership.

The third issue is the over-reliance on AI tools, which may lead to a decline in problem-solving skills among developers. Developers may become too dependent on these tools' suggestions rather than developing their skills and solutions.

However, the main concern is security breaches. Security risks are another challenge, as AI models may introduce vulnerabilities in code, potentially exposing applications to cyber threats [4]. As AI increasingly influences software development, it is essential to find a balance between autogenerated code and the need for human oversight to ensure that the software remains high-quality and secure.

AI-powered tools like ChatGPT, Claude AI, and GitHub Copilot are transforming software development by enhancing productivity, automating repetitive tasks, and assisting in learning. However, challenges related to code reliability, intellectual property, and security breaches need careful management. To maximize the benefits of tools based on artificial intelligence models, developers must use them as supplementary resources, ensuring a balance between AI capabilities and human expertise for efficient, high-quality software development.

REFERENCES

1. Zhang B. Practices and Challenges of Using GitHub Copilot: An Empirical Study [Electronic resource] / B. Zhang, P. Liang, X. Zhou, A. Ahmad, M. Waseem. – Access mode: https://arxiv.org/abs/2303.08733/ (accessed: 15.03.2025). – Title from the monitor.

2. Wermelinger M. Using GitHub Copilot to Solve Simple Programming Problems [Electronic resource] / M. Wermelinger. – Access mode: https://dl.acm.org/doi/10.1145/3545945.3569830/ (accessed: 2025.03.15). – Title from the monitor.

3. Dakhel A. M. GitHub Copilot AI pair programmer: Asset or Liability? [Electronic resource] / A. M. Dakhel, V. Majdinasab, A. Nikanjam, F. Khomh, M. Desmarais, Z. Jiang. – Access mode: https://arxiv.org/abs/2206.15331/ (accessed: 2025.03.15). – Title from the monitor.

4. Khoury R. How Secure is Code Generated by ChatGPT? [Electronic resource] / R. Khoury, A. R. Avila, J. Brunelle, B. M. Camara. – Access mode: https://arxiv.org/abs/2304.09655/ (accessed: 2025.03.15). – Title from the monitor.