

УДК 004.8

Oleksandr Berezhnyi¹, Nataliia Zhukova²

¹ PhD student NU “Zaporizhzhia Polytechnic”

² PhD (Philology), Associate Professor NU “Zaporizhzhia Polytechnic”

MAINTAINING CODE QUALITY AND ROBUSTNESS USING AI TOOLS

IT is the leading industry that integrates artificial intelligence (AI) into all workflows. AI contributes to the improvement and acceleration of management and development processes. In the long term, this reduces the company's expenses, increases efficiency, helps to cope with routine tasks, and provides a competitive advantage. Tools such as intelligent meeting recorders that help create and summarize meeting texts, AI-powered design platforms, generative AI coding platforms, and numerous other systems not only empower employees and speed up regular assignments but also make extremely cumbersome processes like optimization, code analysis and assessment feasible in everyday developers' activities.

Before AI tools broke into our lives, all code quality assessments were the responsibility of individual developers. In the workflow of delivering code, a person creates it, and another person or a group reviews it. All checks and assertions were personal and depended on the reviewers' skills, mood, and other human-related factors. This is not a very reliable approach.

Some generative AI platforms are Qodo – a generative AI coding platform [1] – and Codium-pr-agent, which integrates into version control systems and analyzes the code. They provide real-time feedback during the coding process, enabling the identification and resolution of issues immediately [2]. There is no doubt that these tools enhance the codebase after a short period of usage.

Codium-pr-agent is the tool that helps maintain code quality by checking code against established rules. In programming, there is a clean code concept, meaning that code is readable, understandable, and easily maintained by any developer. Based on this, Codium-pr-agent highlights malfunctions and misleading names and provides suggestions and explanations. It also detects security concerns and identifies the impact of each issue. As a pleasant bonus, it also estimates the effort required to review the code and gives developers recommendations about the focus areas.

Regarding Qodo, it is worth noting that it is integrated into the IDE and has access to the source code. It's not only a tool that completes or generates code; one of its most powerful functions is code analysis. It is very similar to Codium-pr-agent, but the main difference is the number of files it can analyze. This is essential when you need to inspect the whole project. Qodo detects parts that can cause critical errors and security vulnerabilities that are not obvious at first sight. AI can identify these elusive errors and suggest fixes by analyzing the code deeply and considering various scenarios [3]. This is crucial for projects that were established before generative AI was invented. Project architects and tech leaders don't have to spend hours switching from file to file and trying to understand legacy code logic, so there is no place for billion-dollar errors anymore.

Every codebase that aims to be stable and robust should be tested not only by quality assurance (QA) engineers but also by automated tests. This helps exclude the human factor and decreases efforts on QA. The earlier a defect is identified, the less money will be spent fixing it. According to NIST and IBM, the defect resolution cost can increase by 30 to 100 times if identified at the maintenance stage rather than earlier in development [4].

Such systems as Qodo can generate unit and integration tests. AI provides different positive and negative scenarios to cover the code. It is pretty standard for hidden bugs to be revealed during test runs. It's also possible to maintain a consistent style of tests by providing examples to Qodo. Nevertheless, there are concerns about fetching dependencies, especially if importing components from other namespaces

is necessary. Despite this, the generated tests are still very helpful in saving developers' time and covering all scenarios.

To summarize, AI tools are changing the game by making challenging analytic tasks an everyday routine. This will raise the quality standards of software products throughout the industry. Developers compete with each other to create stable and reliable products, and now it has become less labor-intensive. AI helps them to save time, and because of that, it can be dedicated to new features and investigation of ways to utilize innovative AI products.

REFERENCES

1. Generate confidence, not just code [Electronic resource]. – Access mode : <https://www.qodo.ai/> (accessed: 29.03.2025). – Title from the monitor.
2. Slynivuk O. AI Code Review: Can Technologies Really Ensure Code Quality? [Electronic resource] / O. Slynivuk. – Access mode : <https://eleks.com/research/ai-code-review/> (accessed: 2025.03.29). – Title from the monitor.
3. AI Code Review, 3 tools you should know [Electronic resource]. – Access mode : <https://swifterm.com/ai-code-review-3-tools-you-should-know/> (accessed: 2025.03.29). – Title from the monitor.
4. Tkachuk O. AI in Test Automation: From Costs to Benefits [Electronic resource] / O. Tkachuk. – Access mode : <https://blog.qatestlab.com/2024/11/27/test-automation-and-ai-benefits/> (accessed: 2025.03.29). – Title from the monitor.