International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 12 Issue: 03 | Mar 2025 www.irjet.net p-ISSN: 2395-0072

Ramification Of AI-powered Vibe Coding in Agile Development

Ms. Mansi M.Rajapurkar[1], Ms. Pranjali D. Chaudhari[2]

[1,2]Masters Of Science (Information Technology) From University of Mumbai, Maharashtra, India
[2]Bachelors of Education (Maths/Science)

Abstract - *In today's fast-moving tech world, the pressure to develop and deliver quickly has never been greater. Businesses and users expect fast feature rollouts, smooth updates, and flexible solutions that adapt to changing needs. AI-powered tools are making this possible, helping developers speed up coding, simplify workflows, and automate tedious tasks while maintaining high-quality standards. Vibe coding is all about improving the developer experience in Agile development environments, and it has a distinguishable impact on specifically the coding phase of the software development lifecycle. In this paper, we look at both the positive and negative impacts vibe coding has, especially when combined with AI-powered tools. From a positive perspective, vibe coding makes development faster and more efficient by automating repetitive tasks, suggesting code intelligently, and helping to quickly identify bugs and review code, leading to faster iteration, better code quality, and smooth collaboration among team members. Automating these routine tasks also boosts developer satisfaction, helping to reduce burnout and create a more enjoyable, productive work environment.*

Additionally AI tools can be trained to follow organization's security standards and policies to ensure regulations and data privacy as where as security requirements stays intact. On the contrary, resistance to adopting AI, the risk of too much dependency on automation, and the potential to limit creative problem-solving can have adverse implications. This paper highlights the importance for Agile developer teams to utilize AI tools for quicker development, while still relying on their own expertise to ensure tasks are completed efficiently, creatively, and flexibly, all while maintaining strict adherence to development procedures and standards for performance and security.

Key Words: **Vibe Coding, Agile Development, AI-Powered Tools, Code Automation, Developer Experience, Code Quality, Security Standards, Continuous Improvement, Collaboration, Flexibility**

1. INTRODUCTION

1.1 Agile Development Methodology

Agile software development is a broad term that encompasses various approaches to software development, all of which emphasize core values and principles such as prioritizing individuals and interactions over processes and tools, delivering working software over focusing on extensive documentation, fostering customer collaboration over strict contract negotiations, and adapting to change rather than rigidly following a plan.

The Agile methodology is a project management and software development approach that emphasizes flexibility, collaboration, and customer-centricity. Agile is used by major companies today like Facebook, Google, Amazon, etc for the development process. It follows the iterative as well as incremental approach that takes an approach that focuses on making progress step by step, ensuring that a functional product is delivered quickly while continuously improving along the way.

Agile's emphasis on iterative cycles, frequent reviews, and flexibility makes it a powerful methodology for adapting to changing requirements and delivering software that meets customer needs quickly and efficiently.

1.2 Vibe Coding

Vibe coding anchors on experience developers have while coding, beyond just the technical tasks. It involves the environment surrounding a developer's work, including their emotional state, interactions with fellow members, and experience of the development process. The goal is to revamp this experience, making it effortless and more productive. It creates an environment where developers can be creative and collaborate effectively in a stress free manner.

In Agile development, vibe coding is a natural fit. Agile emphasizes flexibility, quick iterations, and teamwork to adapt to change. Developers often face pressure to deliver results fast, making it essential for them to stay motivated and focused. Vibe coding reduces stress, fosters an engaging atmosphere and promotes developer well-being supporting the rapid pace and collaborative nature of Agile (Beck & Fowler, 2001; Highsmith, 2002).

Vibe coding integrates both technical skills and developer experience, taking into consideration team dynamics, task complexity, and tools availability. It focuses on handling technical challenges as well as mental and emotional

aspects of coding, which ultimately improves productivity and job satisfaction (Källberg & Runeson, 2012).

AI has a key role in vibe coding. AI tools automate routine tasks like code generation, bug detection, and testing, allowing developers to focus on creative aspects of development. These tools improve efficiency while maintaining high standards for code quality and security (McMillan & Swaminathan, 2019; Fry & Johnson, 2021).

However, relying too heavily on AI can dampen creativity and disrupt team collaboration. The key is finding a balance, using AI to enhance the developer experience without replacing human creativity (Holtz & Seitz, 2020).

Vibe coding enhances the developer experience by creating a positive and productive environment, fostering creativity and collaboration. By integrating AI tools with Agile methodologies, it streamlines workflows while supporting developers' mental well-being and efficiency.

1.3 AI Powered Coding

AI-powered tools used in the coding phase of Agile development refer to software applications that utilize artificial intelligence to automate tasks and enhance the efficiency during software development lifecycle. These tools integrate machine learning, natural language processing, and other AI techniques to optimize coding tasks such as code completion, bug detection, and real-time feedback. The goal of these tools is to boost the development process, improvise code quality, and curtail human error, while allowing developers to focus on higher-level design and problem-solving (McMillan & Swaminathan, 2019).

AI tools can significantly improve speed and flexibility. Agile emphasizes collaboration, adaptability, and delivering working software in small, frequent iterations. AI-powered tools contribute by automating repetitive tasks, which streamlines workflows and accelerates the development cycle. For example, these tools can identify and fix bugs faster, suggest code optimizations, or even generate code based on developer input, thus improving overall efficiency (Dingsøyr et al., 2012).

The integration of vibe coding with AI-powered tools further enhances the developer's experience. By reducing the cognitive load and eliminating mundane tasks, AI tools help foster a more enjoyable and productive environment. Developers experience less stress, leading to higher morale, creativity, and collaboration—key elements of Agile methodologies (Fry & Johnson, 2021).

The concept of vibe coding ensures that AI tools not only serve technical purposes but also contribute to a positive and collaborative team culture (Holtz & Seitz, 2020).

Popular free and open-source AI-powered tools that are widely used by developers in the coding phase include Eclim, SonarLint, Prettier, GitHub Copilot.

2. LITERATURE REVIEW

2.1. Agile development methodologies

This methodology has revolutionized software development by emphasizing adaptability, collaboration, and iterative progress. Unlike traditional waterfall models, Agile promotes a flexible, customer-centric approach where requirements evolve through continuous feedback. This methodology enhances efficiency, reduces risks, and ensures faster delivery of high-quality software. Agile is widely adopted in industries that demand rapid development cycles and responsiveness to changing requirements.

2.1.1 Phases of Agile Development

Agile development is all about flexibility and continuous improvement. Instead of following a rigid sequence, teams revisit different phases throughout the process to refine and enhance the product. It all starts with **planning**, where teams define project goals, outline requirements, and map out a roadmap. Then comes the **design phase**, where wireframes, system architecture, and UI/ UX elements take shape to guide development. In the **development phase**, coding happens in short sprints, ensuring steady progress with regular updates. The **testing phase** follows, where teams identify and fix bugs through unit, integration, and user testing. After that, the **review phase** allows teams to gather feedback, assess completed work, and make necessary refinements. Finally, the **launch phase** brings the product to users, with ongoing monitoring to ensure smooth performance.

Since Agile is iterative, these steps aren't just one-time tasks—they're revisited as needed to adapt to changes and continuously improve the final product.

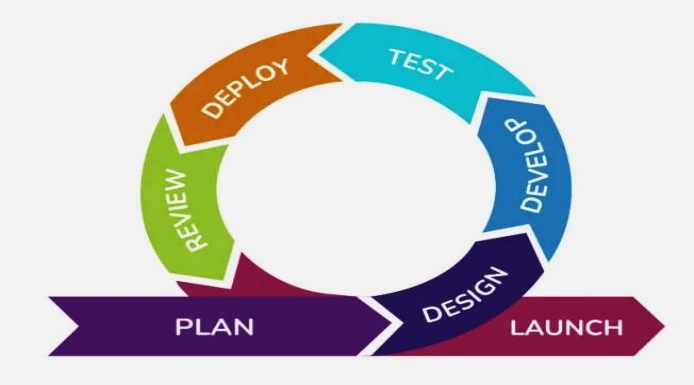

Fig -1: Phases of Agile Model

2.1.2 Popular Agile Development Frameworks

Several frameworks have emerged under the Agile umbrella, each catering to different team structures and project complexities.

Scrum: It is one of the most popular frameworks, focusing on time-boxed iterations called sprints, where teams deliver functional increments of software. It involves roles such as the Product Owner, Scrum Master, and Development Team, ensuring transparency and collaboration through ceremonies like daily stand-ups, sprint reviews, and retrospectives.

Kanban is another Agile framework that visualizes work progress using a board, helping teams manage flow and limit work-in-progress. Unlike Scrum, Kanban does not enforce fixed iterations but encourages continuous delivery. It is particularly effective for teams handling ongoing maintenance and support tasks, as it emphasizes efficiency, reducing bottlenecks, and optimizing workflow.

Lean Agile is derived from Lean manufacturing principles and focuses on eliminating waste, maximizing value delivery, and continuously improving processes. It encourages just-in-time development, minimizing inefficiencies, and optimizing resource utilization. Lean principles emphasize delivering only what is necessary, reducing delays, and fostering a culture of continuous learning and process refinement.

Each of these Agile frameworks follows the core Agile principles of adaptability, collaboration, and iterative delivery but is tailored to different team structures and business needs. The choice of framework depends on factors like project complexity, team size, and organizational goals, making Agile a versatile and widely applicable methodology in modern software development.

2.1.3 Principles of Agile

The highest priority is to satisfy the customer through the early and continuous delivery of valuable software.

- Adapts to stakeholders changing requirements, even late in development .
- Agile development tackles changes to strengthen the customer's competitive advantage.
- Deliver functional software consistently, aiming for shorter time frames whenever feasible.
- Developers and stakeholders should collaborate daily to ensure alignment with customer requirements.
- Empower motivated individuals by providing the right environment and support, trusting them to deliver successful results.
- Prioritize face-to-face communication as the most efficient and effective way.
- Measure progress primarily by the functionality of the working software.
- Agile processes support sustainable development, enabling stakeholders to maintain a steady and consistent pace without major fluctuations.
- Consistently prioritizing technical excellence and strong design enhances agility.
- Simplicity, achieved by minimizing unnecessary work, is essential.
- Self-organizing teams create the best architectures, requirements, and designs.
- The team regularly evaluates its effectiveness and adapts its approach for continuous improvement.

Implementing the twelve principles of Agile will help build an Agile organization that can help with being more flexible to adapt to emerging changes in the process. It will also help in reducing the waste in your system to make the final solution more cost-efficient. It also focuses on delivering the product in a shorter duration of time with accuracy.

2.1.4 Importance of Agile principles

Adaptability and Flexibility: Agile encourages teams to embrace change, allowing them to adjust to new requirements as the project progresses. This makes it easier to respond to shifting customer needs and changing market conditions, keeping the team on track and relevant.

Customer Satisfaction: At the heart of Agile is delivering valuable software quickly and consistently. By focusing on continuous delivery, teams can make sure the product aligns with what the customer really wants, providing real value every step of the way.

Continuous Improvement: Agile fosters a culture where teams regularly reflect on their work and find ways to do better. This constant process of learning and tweaking leads to more efficient workflows and better outcomes over time.

Efficiency and Collaboration: Agile puts a strong emphasis on collaboration, self-organizing teams, and open communication. These principles help boost team productivity and spark innovation, making it easier to solve problems together and work toward common goals.

Quality and Sustainability: Agile encourages practices that focus on long-term quality. By prioritizing technical excellence and solid design, teams are able to deliver software that isn't just functional today, but can also be maintained and built upon in the future.

2.1.5 Importance in Software Development

There are several compelling reasons to choose Agile methodologies for software development such as:

- **Flexibility and Adaptability**: Agile allows teams to adapt to changing requirements, enabling them to respond quickly to shifting priorities, market demands, and customer feedback, ensuring the software remains relevant.
- **Customer Satisfaction**: Agile emphasizes customer collaboration, delivering incremental value to meet evolving needs and expectations, which leads to higher satisfaction.
- **Faster Time to Market**: Agile promotes iterative development, enabling faster releases in smaller, usable increments, offering immediate value to users and stakeholders.
- **Continuous Improvement**: Agile fosters a culture of ongoing learning through retrospectives and feedback loops, leading to higher productivity and improved processes over time.
- **Enhanced Collaboration**: Agile encourages cross-functional, self-organizing teams, fostering collaboration and shared responsibility for the project's success.
- **Transparency and Visibility**: Agile practices like daily stand-ups and sprint reviews promote transparency, ensuring stakeholders are aware of project status, issues, and priorities.
- **Risk Mitigation**: By involving customers early and regularly, Agile reduces the risk of building software that doesn't meet expectations, allowing for continuous validation and feedback.
- **Increased Product Quality**: Agile's iterative approach includes continuous testing and quality checks, leading to early issue identification and higher software quality.
- **Empowered Development Teams**: Agile gives teams ownership and decision-making power, fostering creativity, innovation, and job satisfaction, which leads to better outcomes and higher retention.
- **Measurable Progress**: Agile uses metrics like velocity and burn-down charts to track progress, providing insights for data-driven decisions and project success monitoring.

Adopting Agile methodologies for software development brings a host of benefits, including flexibility, improved customer satisfaction, quicker time to market, and a culture of continuous improvement. It encourages better collaboration, transparency, and helps mitigate risks along the way. Agile also leads to higher product quality, empowers development teams, and provides measurable progress throughout the process. These advantages make Agile a highly appealing and effective choice for today's software development projects.

2.2 Vibe Coding and AI Tools

Vibe coding enhances the developer experience by prioritizing not only technical tasks but also emotional well-being, collaboration, and overall workflow efficiency. It aligns seamlessly with Agile principles, fostering a stress-free and engaging environment that improves productivity and innovation. With the increasing complexity of software development, maintaining an optimal work environment has become essential for sustaining motivation and efficiency in Agile teams (Beck & Fowler, 2001; Källberg & Runeson, 2012).

The integration of AI-powered coding tools has further refined vibe coding by automating mundane tasks such as syntax correction, bug detection, and test case generation. These tools leverage machine learning and natural language processing to assist developers in writing high-quality code efficiently (McMillan & Swaminathan, 2019). AI tools not only boost productivity but also enhance code

consistency and security, enabling teams to focus on creative problem-solving (Fry & Johnson, 2021).

Both free and open-source, as well as paid AI tools, play a crucial role in modern development environments. Open-source tools offer flexibility and customization, fostering community-driven improvements, while paid tools provide enterprise-grade features such as cloud integrations and real-time collaboration. The choice between these tools depends on project requirements, scalability, and organizational needs (Holtz & Seitz, 2020).

Despite their advantages, AI tools must be integrated thoughtfully into the development process. Over-reliance on AI can lead to reduced human engagement and hinder collaborative problem-solving. Agile methodologies emphasize adaptability and teamwork, and AI should be leveraged to support rather than replace human decision-making (Dingsøyr et al., 2012). When balanced effectively, AI-powered coding tools enhance vibe coding, creating a development culture that is efficient, collaborative, and conducive to innovation.

3. TOOLS

3.1 Eclim

It integrates Eclipse's powerful features like code completion and validation with your preferred editor, primarily Vim and other editors like Emacs and Sublime Text. It works by running an Eclipse instance that communicates with the editor, either through a headless setup or by running the Eclim server within Eclipse's graphic user interface.

3.2 SonarQube

It's a tool that analyzes code for bugs, vulnerabilities and automates quality checking. In reference to vibe coding, it enhances the developer experience by giving real-time feedback and maintaining consistent code quality. It integrates with Eclipse, IntelliJ IDEA, and Visual Studio IDE and apart from this can be used as a standalone tool which proves to be beneficial in Agile development.

3.3 Prettier

It is a code formatter helpful in maintaining code by automatically formatting code according to a predefined set of rules. It improves the developer experience by reducing the load of manual styling, allowing developers to focus on complex tasks. Prettier is not tied to a specific IDE and can be integrated into various editors and IDEs, such as Visual Studio Code, Sublime Text and Atom. It can also work as a standalone tool in the development workflow.

3.4 Codacy

It automates code reviews by analyzing code quality through static analysis, identifying issues such as bugs, security vulnerabilities, and code style violations. By providing instant feedback on code quality, it fosters a more streamlined and efficient workflow, enhancing the overall developer experience and reducing stress during the coding process. Codacy can be integrated with GitHub, Bitbucket and GitLab. It works as a cloud-based tool operating across different platforms.

3.5 GitHub Copilot

AI-powered code completion tool assisting developers by suggesting code lines based on the prompts generated by the user. It fosters a more effective and less stressful development environment helpful in promoting productivity and innovation. It can be used as an extension for Visual Studio Code, JetBrains and Neovim like IDEs making it a versatile tool.

4. CASE STUDIES

4.1 Microsoft's AI & Vibe Coding for Redefining Software Innovation

The blog post by Microsoft highlights how AI is transforming businesses by enhancing productivity. It showcases customer stories showcasing the adoption of generative AI to solve problems, particularly in software development. AI is becoming a core part of the transformation plan, and organizations are including AI into software development. Companies struggle with time constraints and the need to innovate quickly especially within Agile development . Developers face difficulties in delivering high-quality code quickly, dealing with the complexity of tasks.

Generative AI, such as Microsoft's AI tools, offers a solution by automating coding tasks, bug fixes, and generating code suggestions. This AI tool assists developers in writing code faster and more effectively and reducing manual effort. Businesses like **PepsiCo**, **TVSnext**, and **HCLTech** have adopted AI to help speed up their software development cycles and optimize resources. The case study is relevant to both paid and free open-source AI tools for code generation in Agile development. Vibe coding in agile methodologies to automate code generation aligns with the need for faster iterations and streamlined workflows promoting continuous delivery.

4.2 GitHub's Survey on AI Adoption in Software Development (2024)

This case study stems out of GitHub's developer survey, which looks at the effect of AI-powered coding tools upon software development. The study examines how AI influences several key aspects of Agile methodology. This particularly includes code security, customer requirements, efficiency, learning, code quality, organizational adoption, as well as overall usage. The clear findings offer understanding into the ways that AI is reshaping development workflows within diverse regions especially within the coding phase.

4.2.1 Enhancing Code Quality with AI

This survey highlights the positive impact of vibe coding AI tools on code quality. In the U.S., an impressive 90% of developers say AI significantly enhances the quality of their code. India follows closely with 81%, while Brazil and Germany report 61% and 60%, respectively. Hardly few developers feel that AI has less to no impact on code quality.

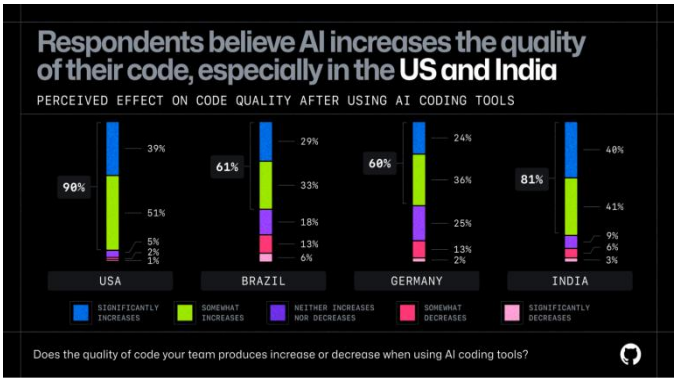

Fig -2: Survey of Code Quality with AI

In Agile development, maintaining high code quality is crucial for long-term success. AI-driven tools align with Agile principles by helping teams reduce technical debt through smart code suggestions, automated refactoring, and predictive issue detection. By keeping code cleaner and more maintainable, these tools support sustainable software development and smoother project workflows.

4.2.2 Time Optimization and Task Efficiency

The developers are making the most of the time saved by using AI coding tools. Instead of just speeding up tasks, many are reinvesting that time into areas that drive long-term success. In the U.S. and Germany, 47% of developers say they dedicate their extra time to designing systems and customer solutions, while 46% focus more on working closely with their teams. Meanwhile, in Brazil and India, 44% are using AI-enabled efficiencies to dive deeper into researching emerging technologies. This shift aligns perfectly with Agile methodologies, where teamwork and continuous learning are at the heart of development. By automating repetitive coding tasks, AI frees developers to focus on higher-level problem-solving, strategic decision-making, and refining user stories with greater precision. With AI reducing the manual workload, Agile teams can iterate faster, innovate more effectively, and collaborate more seamlessly.

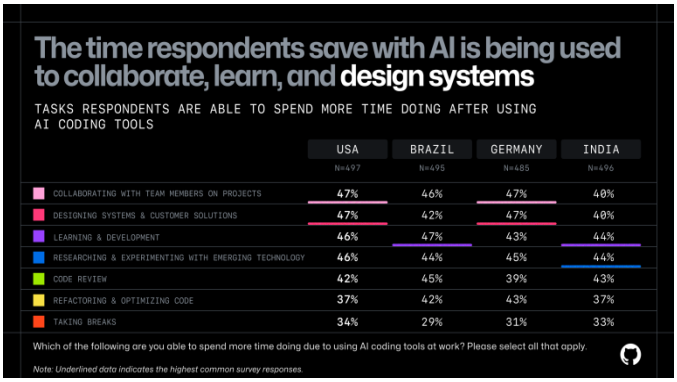

Fig -3: Survey on Time Optimization and Task Efficiency using AI

4.2.3 Organizational Adoption of AI Coding Tools

The way companies embrace AI coding tools varies significantly across different regions. In the U.S. and Brazil, 38% of organizations actively encourage their use, while India sees a slightly higher adoption rate at 40%. In contrast, Germany lags behind, with only 30% of companies promoting AI tools, as more organizations take a neutral stance on adoption.

For Agile teams, strong organizational support for AI can make a real difference. When companies actively encourage AI usage, teams have the freedom to experiment with automation, streamline workflows, and boost code quality. This leads to faster development cycles and greater efficiency.

On the other hand, companies that remain hesitant or neutral may experience slower adoption rates, missing out on the full potential of AI-driven productivity.

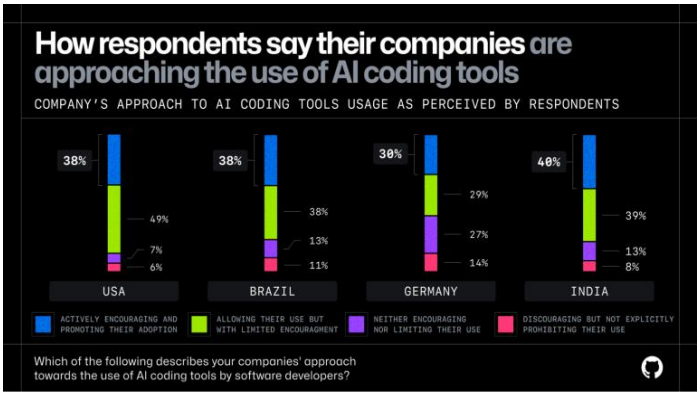

Fig -4: Survey on organization adopting AI coding tools

4.2.4 Adopting New Programming Languages and Understanding Codebases

Developers see AI coding tools as a powerful ally when it comes to learning new programming languages and navigating complex codebases. In the U.S., 71% of developers say AI makes it easier to pick up new languages, with India close behind at 69%. Brazil and Germany also report strong benefits, with 60% and 61% of developers, respectively, finding AI helpful in this area.

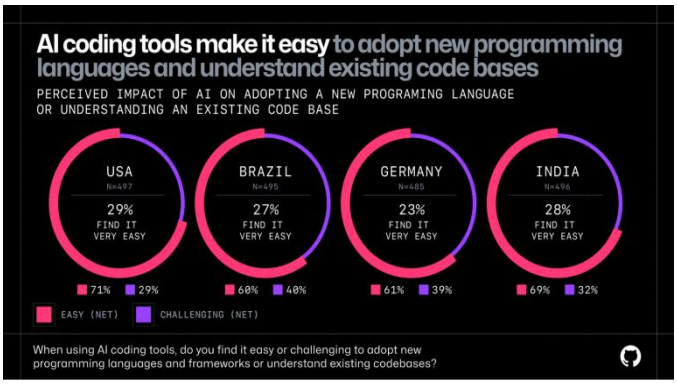

Fig -5: Survey on adopting new languages & existing code bases using AI coding tools

This advantage is especially valuable for Agile teams that frequently work across multiple technology stacks and shift between different frameworks. AI-driven code assistance helps developers quickly grasp unfamiliar languages and understand existing codebases, reducing onboarding time and improving knowledge sharing. As a result, teams can accelerate development cycles, collaborate more effectively, and maintain a steady pace of innovation.

4.2.5 AI and Code Security

Developers are increasingly confident that AI coding tools play a positive role in improving code security. In the U.S., 51% of respondents believe AI will "somewhat" enhance security, while 38% expect a significant improvement. Similar trends emerge in Brazil, Germany, and India, where most developers see AI as a valuable asset for secure coding. Across all regions, fewer than 2% believe AI will have little to no impact—or could even harm security. This aligns well with Agile development's focus on continuous improvement and proactive risk management. By automating security checks, detecting potential threats, and analyzing vulnerabilities in real time, AI enables development teams to identify and fix issues earlier in the software lifecycle. This not only reduces security risks but also strengthens the overall stability and reliability of Agile-driven projects.

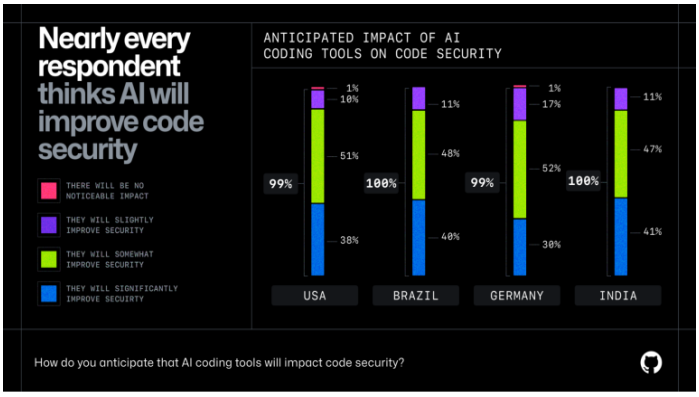

Fig -6: Survey of AI coding tool's impact on security

4.2.6 AI's role in meeting Customer Requirements

Developers see AI coding tools as a valuable asset in delivering better customer-focused solutions. In the U.S., 73% of respondents say AI significantly improves their ability to meet customer needs, followed by 68% in Brazil and 66% in India. Germany reports a slightly lower, yet still strong, confidence level at 61%. For Agile teams, AI plays a key role in adapting to evolving customer demands. By automating repetitive coding tasks and providing intelligent optimization suggestions, AI frees up developers to concentrate on refining user-centric features. This enhances responsiveness and agility in development workflows. Additionally, AI-driven insights can help teams prioritize backlog items based on predictive analytics, ensuring that product development stays aligned with customer expectations.

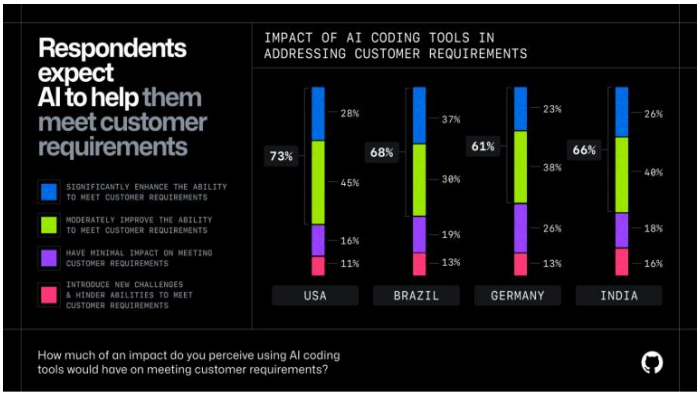

Fig -7: Survey on AI's impact in meeting customer requirements

4.2.7 The Growing Popularity of AI Coding Tools

AI coding tools are rapidly becoming a staple in software development, with over 80% of developers in the U.S., Brazil, and India using them at work. Germany, while slightly lower, still shows a strong adoption rate at 70%.

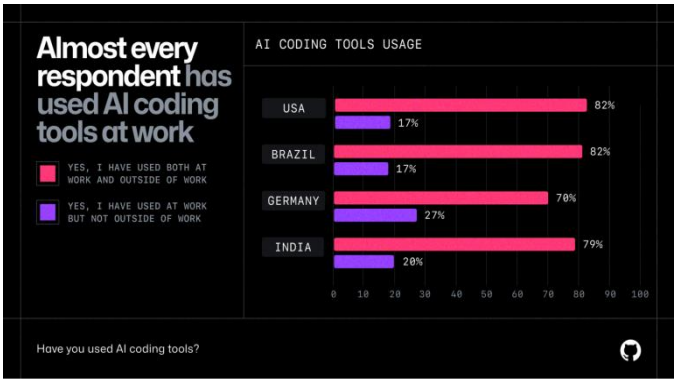

Fig -8: Survey on use of AI coding tools

What's even more interesting is that many developers across all regions use AI coding tools not just professionally, but also in personal projects—highlighting their growing impact beyond the workplace. This widespread use signals a shift in how software is being built. AI coding tools are no longer just optional add-ons; they're becoming essential for improving efficiency, reducing development time, and fostering collaboration. For Agile teams, integrating AI into their workflows aligns perfectly with Agile's core principles—helping teams stay adaptable, continuously improve, and build better software, faster.

AI coding tools are transforming Agile development by enhancing security, improving efficiency, and boosting code quality. Organizations that embrace AI empower teams to innovate faster and deliver higher-quality software. AI also simplifies onboarding, supports learning new languages, and improves code comprehension which is a key to Agile's flexible, cross-functional approach. By automating repetitive tasks, AI frees developers to focus on collaboration, problem-solving, and strategic decisions.

Ultimately, AI is reshaping how developers work, learn, and contribute. Companies that integrate AI into Agile workflows will see greater efficiency, happier developers, and stronger software outcomes.

5. FINDINGS

AI-Powered Vibe Coding is Transforming Agile Development. AI-driven coding tools are reshaping Agile development, especially in the coding phase, by speeding up workflows, improving code quality, and fostering collaboration. Tools like GitHub Copilot, SonarQube, Codacy, and Prettier provide real-time coding assistance, reducing repetitive tasks and allowing developers to focus on higher-level problem-solving.

One of the biggest benefits is better code quality. AI-powered tools help create cleaner, more structured, and optimized code through automated refactoring, predictive issue detection, and instant feedback. According to a GitHub survey, most developers recognize AI's role in reducing technical debt and maintaining high software standards.

Another key shift is how developers use their time. Instead of just coding faster, AI frees them up to focus on system design, customer-driven solutions, and researching new technologies. This aligns perfectly with Agile's focus on continuous learning and iterative improvement, showing that AI isn't replacing developers—it's enhancing their strategic thinking and problem-solving abilities.

AI adoption varies by region, with countries like India and the U.S. leading the way, while Germany takes a more cautious approach. A company's stance on AI also matters—organizations that actively encourage AI integration see faster development cycles, higher productivity, and better software quality.

AI tools are also making a big impact on onboarding and cross-team collaboration. Developers working with different technology stacks find AI support invaluable in understanding codebases and picking up new programming languages. In Agile teams, where flexibility and knowledge-sharing are key, this adaptability helps drive innovation.

Security remains a top priority, and AI coding tools play a critical role in identifying vulnerabilities, automating security checks, and ensuring compliance with industry standards. However, developers still need to review and

validate AI-generated code to maintain best practices and prevent potential risks.

Ultimately, while AI-powered Vibe Coding boosts efficiency and productivity, it doesn't replace human expertise. The best Agile teams find a balance between AI automation and developer intuition, creativity, and oversight. Companies that embrace this synergy will not only deliver software faster but also reduce burnout and build stronger, more maintainable solutions.

6. CONCLUSIONS

To conclude, integrating AI tools for generating lines or blocks of code, specifically through Vibe coding, has significantly improved the coding phase of the Agile development model. Streamlining the process by automating repetitive tasks allows developers to focus more on creative and problem-solving aspects. This speeds up development and also aligns with Agile's goal of delivering functional software rapidly and iteratively.

Tools used in vibe coding enhances collaboration, which is a key component of Agile. In situations where developers may be unavailable due to geographical boundaries or so, AI can help others understand their code, providing valuable insights. This ensures the team remains cohesive and productive in the interim absence of a member. The ability to easily understand and build on each other's work directly supports Agile's emphasis on teamwork and adaptability.

Another major benefit is AI assistance in writing effective prompts to generate cleaner, error-free code. With mindful prompt engineering, developers can reduce issues at both the syntax and logic levels. However, using AI tools effectively requires a solid understanding of their foundational functionalities. Developers need to know how to craft precise prompts and avoid common pitfalls to ensure that AI-generated code is both secure and adheres to coding best practices. This expertise is much needed to avoid potential coding errors and security vulnerabilities as products developed under such scenarios might at a point process customers' confidential data in future.

Despite the immense benefits AI tools provide, the role of the developer cannot be replaced. Developers must still attentively review, test, and refine AI-generated code. Agile principles, such as continuous improvement and adaptability, are best supported by human oversight. Developers are accountable for ensuring that the code is functional, secure and aligns with the project's objectives avoiding scope creep.

Vibe coding, when used alongside skilled developers, AI tools in the Agile coding phase can be a revolutionizer. Vibe coding speeds up development, improvises collaboration and minimises errors. However, the expertise and judgment of developers are still crucial for creating a final product aligning to stakeholders requirements and standards.

REFERENCES

[1] https://en.wikipedia.org/wiki/Agile_software_development

[2] https://www.techtarget.com/searchsoftwarequality/definition/agile-software-development

[3] Beck, K., & Fowler, M. (2001). *Agile Software Development: The Agile Manifesto*. Agile Alliance, *Planning Extreme Programming*. Addison-Wesley.

[4] Källberg, M., & Runeson, P. (2012). The emotional experience of software development: A grounded theory study. *Empirical Software Engineering*, 17(2), 163-190. Developer experience and productivity in Agile software projects. *Empirical Software Engineering, 17(5)*, 786-810.

[5] Holtz, A., & Seitz, R. (2020). *AI-Assisted Software Development: A Survey of Tools and Methods*. Springer. The role of AI in developer experience: Balancing automation and creativity. *International Journal of Software Research, 28(3)*, 198-215. *The Role of Artificial Intelligence in Agile Development: Challenges and Opportunities*. Springer.

[6] McMillan, R., & Swaminathan, N. (2019). *AI in Software Development: Current Trends and Future Impact*. ACM Computing Surveys, 51(3), 1-25. Machine learning applications in software engineering: A systematic review. *IEEE Transactions on Software Engineering, 45(3)*, 245-261.

[7] Dingsøyr, T., Nerur, S., Balijepally, V., & Sammut, S. (2012). A decade of Agile methodologies: Towards explaining Agile software development. *Journal of Systems and Software*, 85(6), 1213-1221.

[8] Fry, J., & Johnson, J. (2021). *Optimizing Developer Efficiency Using AI Tools in Agile Environments*. Software Engineering Journal, 29(4), 440-453. AI in Agile development: Enhancing efficiency and collaboration. *Software Engineering Journal, 36(4)*, 255-269.

[9] https://blogs.microsoft.com/blog/2025/03/10/https-blogs-microsoft-com-blog-2024-11-12-how-real-world-businesses-are-transforming-with-ai/

[10] https://github.blog/news-insights/research/survey-ai-wave-grows/#key-survey-findings

[11] https://www.agilealliance.org/agile101/12-principles-behind-the-agile-manifesto/

[12] https://businessmap.io/agile/project-management/principles

[13] https://premieragile.com/types-of-agile-frameworks/

[14] https://fibery.io/blog/product-management/agile-frameworks/

[15] https://www.fegno.com/agile-methodology-for-software-development/

BIOGRAPHIES

Ms. Mansi M. Rajapurkar
M. Sc Information Technology

Ms. Pranjali D. Chaudhari
M. Sc Information Technology
B.Ed (Maths/Science)