mitrais | MEMBER OF CAC HOLDINGS GROUP

How to adopt Vibe Coding in a Security First Organisation?

Mitrais has been a pioneer in the development of near-shore software development services to Australia and other markets for more than 30 years. With over 500 software engineers, Mitrais provides services from development centers in Bali, Jakarta, Bandung and Yogyakarta in Indonesia.

www.mitrais.com

Table of Contents

1 What is Vibe Coding?

Vibe Coding is a relatively new term in the software development industry. Its popularity has been increasing along with AI powered development tools. Vibe Coding is so new it does not yet have a single formal definition.

The term “Vibe Coding” was introduced by Andrej Karpathy in February 2025. It refers to a programming method that relies on AI LLM to generate code from natural language expressions. Andrej Karpathy mentioned in his tweet that he relied on LLM to generate code and accept all the code generated without reading it and he found it was amusing that it mostly works (Karpathy, 2025).

Accepting AI code generated without fully understanding it, has become a key part of the vibe coding paradigm (Edwards, 2025) and it has raised concerns in accountability and security.

IBM defines Vibe coding as an emerging style of programming where developers describe their intent in plain language, allowing AI systems to generate, adapt, and maintain code in an iterative and conversational way (Harkar, 2025).

Indie Hackers definition of Vibe coding is the practice of using AI coding assistants like Cursor, GitHub Copilot, and other AI tools to rapidly build software (IndieHackers, 2025).

In summary, Vibe Coding refers to writing software with the help of an AI coding assistant in a highly conversational, exploratory, and iterative manner where the programmer prompts AI using natural language to produce code. We will use this definition of vibe coding in this white paper.

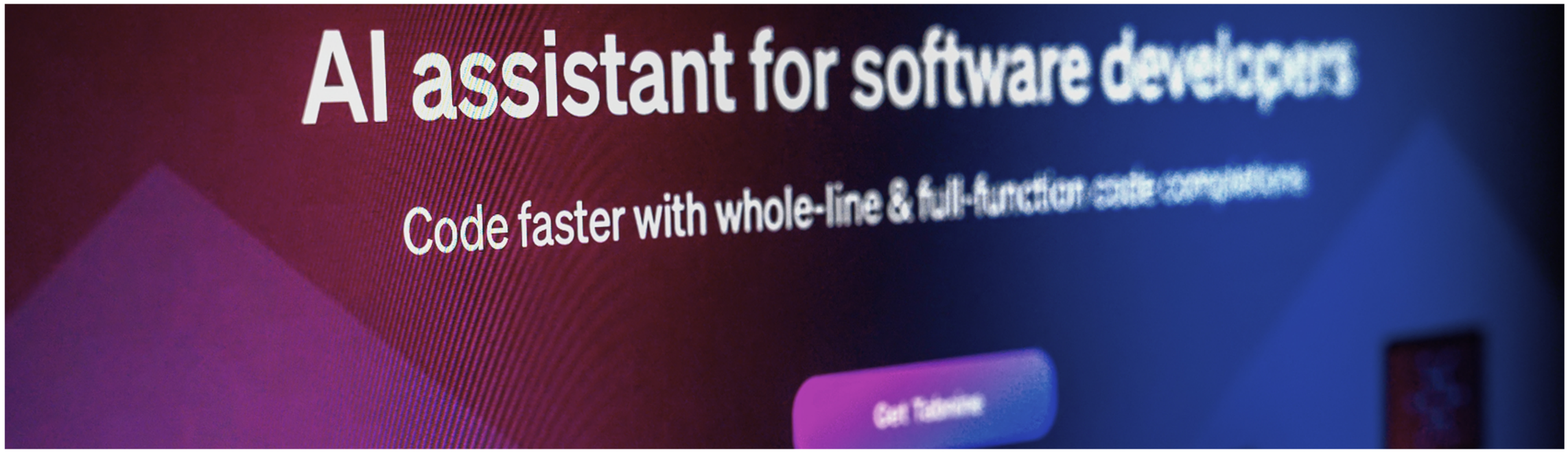
2 Adoption of Vibe Coding

Vibe coding allows people without programming knowledge and experience to produce code and develop applications. For traditional programmers, AI has the potential to increase their efficiency by providing coding assistance, allowing them to have more time for planning and design.

The trend of vibe coding adoption has risen rapidly since its inception.

In March 2025, Y Combinator reported that 25% of startup companies in its Winter 2025 batch had codebases that were 95% AI-generated. It reflected a shift toward AI-assisted development in newer startups (Mehta, 2025).

With the rise in popularity of AI tools, developers often find responses from AI tools such as Chat GPT or Copilot useful to help them solve coding problems.



3 Security Vulnerability in Vibe Coding

Advocates of vibe coding say that vibe coding lowers the barrier for programming and can increase productivity. A controlled experiment showed developers using AI assistance completed tasks 55.8% faster compared to those who did not use AI assistance (Peng, Kalliamvakou, Cihon, & Demirer, 2023).

Critics argue that vibe coding raises concerns about code understanding and accountability and this leads to bugs, errors and security vulnerabilities (Edwards, 2025).

In one high-profile incident, Replit's CEO apologised after its AI agent deleted a code base and lied about its data (Ming, 2025).

3.1 Credential and Secrets Exposure

AI generated code might not follow best practice of secure secrets management and can result in production database credentials and other secrets being hardcoded in the source code generated, leading to leaked production data. Security researchers demonstrated that GitHub Copilot sometimes suggests API keys, database passwords and access tokens in generated code due to training data included in public repositories that commit secrets to source code, so the model learned and reproduced the patterns (McDaniel, 2025).

3.2 Sensitive Information Exposure

Personal Identifiable Information (PII) may be disclosed during interactions with AI coding LLM. The risk is an OWASP Top 10 for LLM applications (OWASP, 2025).

Improper error-handling in AI generated code might include details and call stacks, potentially revealing the system's inner working and configuration, which can be used by attackers to compromise the system.

3.3 Unsafe Data Handling

AI generated code might miss input validation and sanitisation and is vulnerable to SQL injection, Cross-site Scripting (XSS) and Cross-site Request Forgery (CSRF).

A study of 2,500 small PHP websites generated with GPT-4 found that 26% had at least one security vulnerability including SQL Injection and XSS (Tóth & Erdődi, 2024). The study shows that AI generated code contains SQL queries built directly from user input and omits input data validation and sanitation, making it vulnerable to SQL injections and XSS attacks.

3.4 Broken Access Control

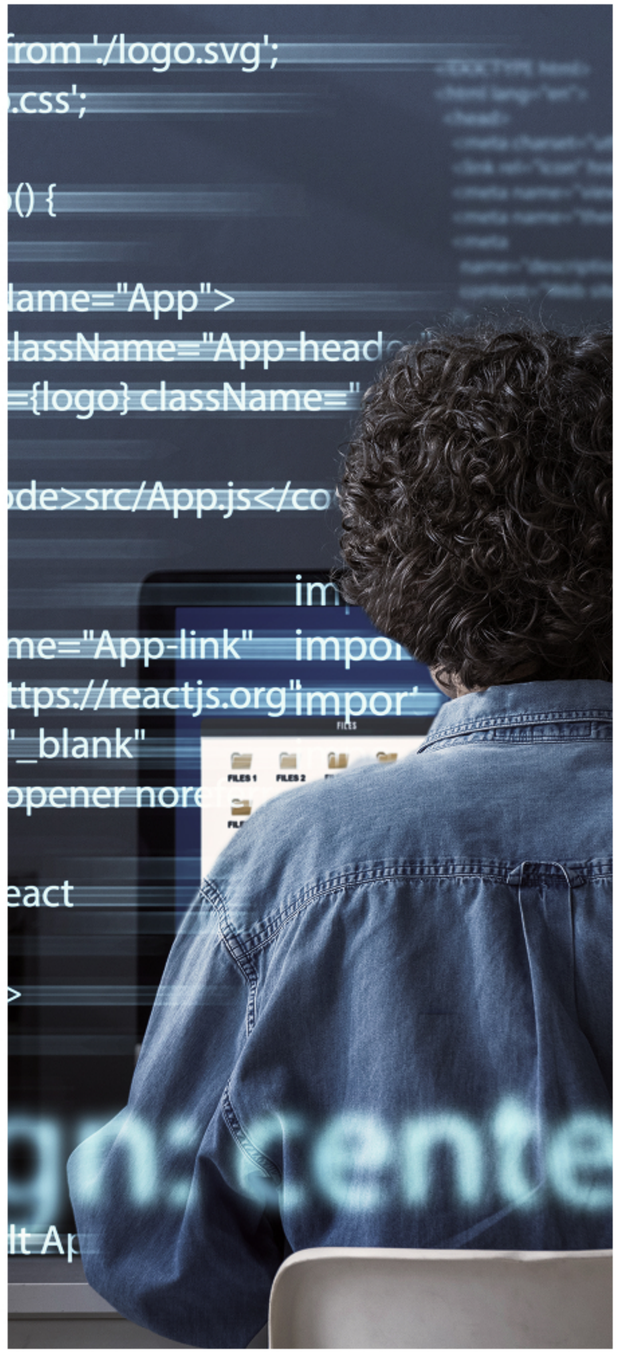
AI generated code sometimes misses authentication and authorisation checks and it leads to unauthorised access to sensitive data or functionality. Tea app (www.teaforwomen.com), a social platform coded almost entirely by AI agents and launched quickly, made private messages and photo links publicly accessible because of a misconfiguration in access controls (Saadioui, 2025). This shows that we cannot rely on AI to enforce access controls. Developers need to specify, design and implement proper access controls.

3.5 Prompt Injection

This issue can potentially occur with open-source projects that implement agentic AI to fix bugs reported. Agentic AI is a class of artificial intelligence that focuses on autonomous systems that can make decisions and perform tasks without human intervention. The independent systems automatically respond to conditions, to produce process results. In July 2025, a hacker inserted a malicious prompt into an Amazon Q Developer extension via a GitHub pull request and instructed the AI to wipe local systems and delete AWS cloud resources (Udinmwen, 2025).

GitHub Copilot exploit was found and can be used to launch prompt injection attacks in projects that allow the public to report bugs and AI agents are used to create pull requests to fix bugs. Attackers can use the exploit to inject hidden malicious instructions for AI agents to create backdoors, allowing attackers to gain sensitive information by sending backdoor commands via X-Backdoor-Cmd HTTP header (Higgs, 2025).

4 Adopt Vibe Coding with Secure Coding Practices



While AI coding assistance provides developers with a tool to improve productivity, it also comes with concerns of security vulnerabilities in both AI generated code, and from interactions with an AI coding assistant. The question is how can we use it without exposing ourselves to the security vulnerabilities mentioned?

First, developers need to understand the context of the AI coding assistant. It is a tool to assist developers in writing code. It is a co-pilot, and developers need to remain in the pilot's seat. In the end, developers will remain accountable for every line of code committed, even if some or all were generated or suggested by AI.

At an organisational level, establish policies to adopt a security-first approach to ensure that security is built into the software development lifecycle (SDLC). Some examples include:

a. Security requirements should be identified and elaborated for each feature.

b. Perform threat modelling to identify security risks.

c. Design, implement and test measures to resolve and mitigate those risks.

d. For implementation, adopt secure coding practices.

OWASP provides guidance and checklists that can be integrated into the software development lifecycle (OWASP Foundation, 2025). Implementation of these practices will mitigate most common software vulnerabilities. If required, adjust the guidance according to organisational needs.

Coding with AI assistance must follow the security policies and secure coding practices as outlined in the following sections.

4.1 Organisation Policy

a. Use only AI coding assistants that have been reviewed and approved by your organisation.

b. Provide AI coding licenses to developers that are suitable for your organisational needs. Business licenses offer more flexibility and security protection.

c. Provide secure coding training to developers.

d. Provide policy and guidance to developers on how to incorporate the chosen AI coding assistant into software development activities.

e. Configure the AI coding assistant to block code suggestions that match public code.

4.2 Secure Prompting

a. Do not use credentials, secrets, confidential & private data and other sensitive information in prompts.

b. Do not put credentials and secrets in source code. Always use secure secret management to secure credentials and secrets.

c. Use negative constraints to prohibit AI from using insecure practices.

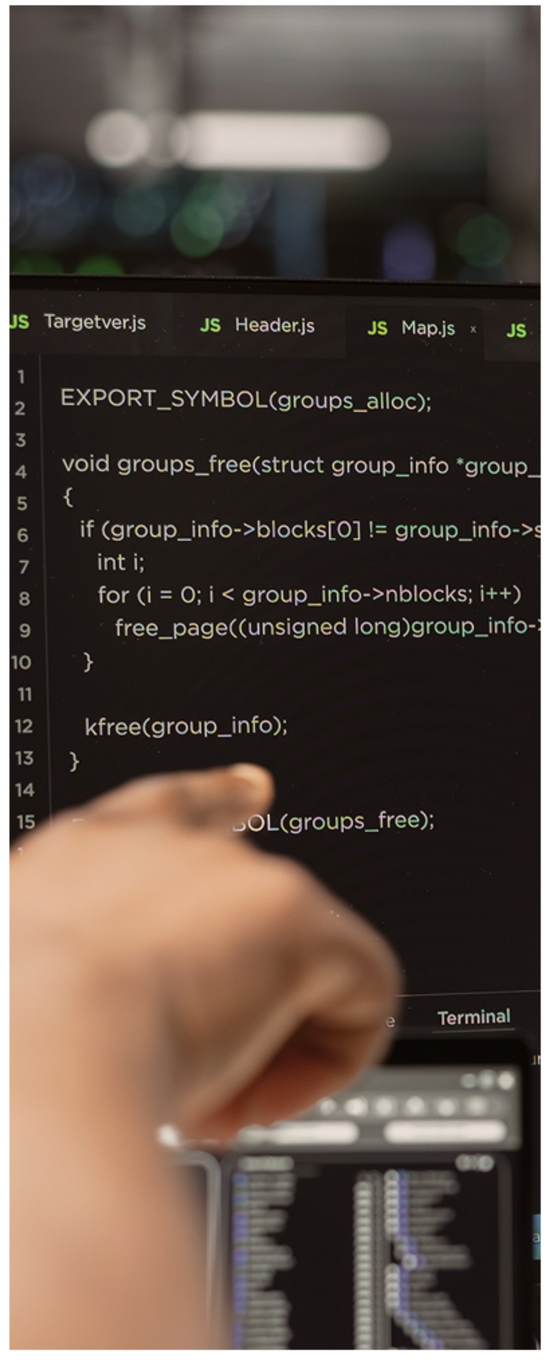
d. Use multi-stage prompting to get AI to implement a feature, then to review its own output for security issues.

e. Specify the authorisation needed to access protected resources to prevent broken access control and unauthorised access.

f. Specify strong and robust authentication with multi factor authentication.

g. Specify input data validation and sanitisation to prevent injection and script attacks.

h. Include details of any other security requirements e.g. encryption method, data protection.



4.3 Code Quality and Verification

Review and ensure the code is correct, secure and conforming to required standards. Provide a detailed secure coding checklist that developers can practically use to review code, including code generated/suggested by AI. The checklist needs to make sure that developers fully understand the code before committing it.

Ensure the code review checklist covers measures to prevent these security issues:

a. Hardcoded secrets
b. SQL injection
c. XSS attack
d. CSRF attack
e. Broken access control
f. Path traversal
g. Insecure deserialisation

Develop a “trust but verify” principle for developers to always verify that AI output has met functional and non-functional requirements. Conduct testing to ensure quality and requirements are met.

4.4 Integrate SAST and DAST

Run Static Application Security Testing (SAST) such as SonarQube in CI/CD pipelines to detect security vulnerabilities in code.

Perform Dynamic Application Security Testing (DAST) such as Burp Suite Pro in testing to find security issues that SAST might not detect.

4.5 Perform Software Component Analysis (SCA)

Perform SCA to scan third-party components for known security vulnerabilities. If needed, upgrade or replace components with vulnerabilities to prevent supply chain attacks.

4.6 Guard Against Prompt Injection

AI tools are susceptible to prompt poisoning coming from comments, config files, md files and external data sources that are usually perceived as not malicious. Run scanner tools to detect malicious or manipulative text potentially containing hidden instructions that could trick AI to do something harmful. The tool should detect the following:

a. **Obfuscated commands** e.g. “ignore previous instructions”, “download this file and execute”
b. **Encoded payloads** such as base64, hex, URL-encoded
c. **Steganographic tricks**: instructions hidden in markdown, HTML comments, or images

Block data containing suspicious content from reaching AI agents.

Conclusion

AI helps developers to improve their productivity in writing code. However, it also brings security risks. To mitigate those risks, establish and implement secure coding policy and procedure in your organisation to ensure productivity and security. Align the use of AI in coding with the policy and procedure.

Mitrais is an industry-leading software development company that delivers secure, scalable, and high-performing software solutions. We are thrilled with the potential of using AI coding assistant in software development process. On the other hand, we are aware of its security risks and what measures need to be taken to mitigate those risks. We are on a journey of embracing AI, not to replace our developers, but to help them to provide more added value to our customers while security remains the top priority. Contact us today if you are thinking of developing secure, scalable and high-performing software solutions to grow with your business.

Reference

Edwards, B. (2025, March 6). Will the future of software development run on vibes?

Harkar, S. (2025, April 8). What is vibe coding?

Higgs, K. (2025, August 6). Prompt injection engineering for attackers: Exploiting GitHub Copilot.

IndieHackers. (2025, May 1). Vibe Coding.

Karpathy, A. (2025, February 3).

McDaniel, D. (2025, March 27). GitHub Copilot Security and Privacy Concerns: Understanding the Risks and Best Practices.

Mehta, I. (2025, March 6). A quarter of startups in YC's current cohort have codebases that are almost entirely AI-generated.

Ming, L. C. (2025, July 22). Replit's CEO apologizes after its AI agent wiped a company's code base in a test run and lied about it.

OWASP. (2025). OWASP Top 10 for Large Language Model Applications.

OWASP Foundation. (2025, July 15). OWASP Developer Guide.

Peng, S., Kalliamvakou, E., Cihon, P., & Demirer, M. (2023, February 14). The Impact of AI on Developer Productivity.

Saadioui, Z. (2025, August 11). AI-Generated Code in Production: A Security Audit of the Risks.

Tóth, R., & Erdődi, L. (2024, May 21). LLMs in Web Development: Evaluating LLM-Generated PHP Code Unveiling Vulnerabilities and Limitations.

Udinmwen, E. (2025, July 30). Hacker adds potentially catastrophic prompt to Amazon's AI coding service to prove a point.

MITRAIS | MEMBER OF CAC HOLDINGS GROUP

Contact Us

Indonesia 0361-849-7952

Bali

Jl. By Pass Ngurah Rai
Gg. Mina Utama No. 1,
South Denpasar, Denpasar,
Bali 80223

Jakarta

Wirausaha Building, 8th Floor,
Jl. H.R. Rasuna Said Kav. C5,
South Jakarta,
Jakarta 12940

Bandung

Jl. Prof. Drg. Surya Sumantri No. 8D,
Sukawarna, Sukajadi, Bandung,
West Java 40164

Yogyakarta

Jl. Sidobali No. 2, Muja Muju,
Umbulharjo, Yogyakarta,
Special Region of Yogyakarta
55165

Overseas

Singapore

3158-1185

10 Anson Road,
#03-05
International Plaza,
Singapore 079903

Australia

1800-755-025

New Zealand

0800-755-025

MITRAIS | MEMBER OF CAC HOLDINGS GROUP

Terima Kasih　　**Thank** You　　**ありがとう**ございました