Ensuring Code Quality and Security in AI assisted Software Engineering Training

Duration: **1 Day**

BİLGİNÇ IT Academy
info@bilginc.com
bilginc.com

Ensuring Code Quality and Security in AI-Assisted Software Engineering is a practical course for software professionals who want to take advantage of AI-assisted development while retaining responsibility for code quality, application security, and compliance.

Generative AI tools can accelerate software delivery, but accepting AI-generated code without appropriate validation can introduce logical defects, security vulnerabilities, unnecessary complexity, and overlooked edge cases. This course focuses on helping developers critically assess AI outputs and reinforce them with established software engineering practices.

Participants examine common AI-generated coding problems, including hallucinated APIs, incorrect assumptions, incomplete test scenarios, and insecure implementation patterns. Testing, static analysis, automated quality checks, and secure coding techniques are applied to realistic development scenarios. The course also uses OWASP Top 10 risks to explore how AI-generated code can be assessed and improved from a security perspective.

Beyond technical controls, the programme addresses code provenance, intellectual property, licensing, data protection, and organisational AI policies. Learners therefore develop an approach to AI-assisted software engineering that balances productivity with quality, security, governance, and responsible use.

Prerequisites

Participants should have:

- Experience writing code in at least one programming language,
- Familiarity with software development practices such as version control and testing,
- A basic understanding of application security concepts,
- Awareness of AI-assisted development tools such as GitHub Copilot or similar solutions.

Who Should Attend

This course is designed for:

- Software developers and engineers using or introducing AI-assisted coding tools,
- Technical leads responsible for code quality and security standards,
- DevOps and platform engineers integrating AI and automation into development workflows,
- Technical teams within organisations adopting AI-assisted software development as part of a wider transformation programme.

Ensuring Code Quality and Security in AI assisted Software Engineering Training

Duration: **1 Day**

BILGINÇ IT Academy
info@bilginc.com
bilginc.com

What You Will Learn

By the end of this course, participants will be able to:

- Use AI-assisted development tools while retaining accountability for code quality and security,
- Identify logical defects, insecure patterns, and other common risks within AI-generated code,
- Integrate testing strategies, static analysis, and automated quality checks into AI-assisted workflows,
- Detect and remediate vulnerabilities associated with OWASP Top 10 risks,
- Refactor AI-generated code to improve maintainability, performance, and robustness,
- Determine when AI-generated outputs are sufficiently reliable and when additional validation is necessary,
- Contribute to organisational governance frameworks for responsible AI-assisted development.

Training Outline

Kick-Off and the AI Development Landscape

- Review AI-assisted development tools and their capabilities
- Demonstrate model comparison tools and prompting approaches
- Examine productivity benefits against quality and security trade-offs
- Discuss current uses of AI within development workflows
- **Lab:** Evaluate the quality of AI-generated code
- Assess AI outputs using a structured checklist
- Identify correctness, maintainability, and security concerns
- Classify and prioritise findings according to risk

Challenge Exercise: Task Management API

- Generate a task manager incorporating dependencies and scheduling logic
- Detect circular dependencies and resource conflicts
- Evaluate the implementation against business requirements
- Apply structured code review techniques

Understanding AI Code Quality Pitfalls

- Explore common failure patterns in AI-generated code
- Identify hallucinated APIs and incorrect assumptions
- Recognise hidden complexity and over-engineering
- Find missing edge cases and inconsistent logic
- Maintain coding standards across human and AI contributions
- Integrate linters, formatters, and automated code review tools
- Discuss pull request and review practices

Lab: Testing and Refactoring AI-Generated Code

- Generate and execute AI-created unit tests
- Identify gaps in test coverage
- Validate edge cases such as invalid inputs and concurrency
- Refactor for clarity, modularity, and maintainability

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

- Improve logging and error handling
- Apply performance improvements

Challenge Exercise: E-Commerce Pricing Engine

- Build pricing logic covering discounts, taxes, and promotions
- Identify calculation problems and overlooked edge cases
- Strengthen test coverage and ensure deterministic outcomes
- Apply static analysis and quality gates

Security in AI-Assisted Development

- Introduce OWASP Top 10 risks in the context of AI-generated code
- Examine common vulnerabilities in authentication, data handling, and APIs
- Explore security scanning and dependency analysis tools
- Align secure coding practices with AI-assisted workflows

Lab: Identify and Remediate Vulnerabilities

Analyse an AI-generated user management system and investigate:

- Broken access control
- Weak cryptographic practices
- Injection vulnerabilities
- Authentication weaknesses
- Differences between manual reviews and automated security tool findings
- Parameterised queries
- Strong password hashing
- Input validation and sanitisation
- Secure token handling

Security Testing and Validation

- Create test cases that simulate attacks
- Perform basic penetration testing scenarios
- Validate remediation against defined security requirements

Advanced Exercise: Multi-Factor Authentication

- Extend the application with secure authentication mechanisms
- Address relevant edge cases
- Consider timing attack risks
- Balance security requirements with usability

Governance, Intellectual Property, and Compliance

- Code provenance and AI-generated content considerations
- Licensing and intellectual property risks
- Data protection requirements
- Organisational policies governing AI usage
- Establish responsible AI development practices

Group Exercise: Responsible AI Coding Policy

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

- Define organisational standards for using AI development tools
- Establish tool selection and approval requirements
- Address code attribution and intellectual property protection
- Define quality and security gates
- Establish developer training and competency expectations
- Define incident response and audit processes

Industry Scenario Workshops

- Financial services
- Healthcare
- Government
- Retail
- Identify sector-specific regulatory and compliance requirements
- Balance productivity benefits against risk management requirements

Policy Presentation and Synthesis

- Present and review team policies
- Consolidate identified best practices
- Define an implementation roadmap and success metrics

Wrap-Up and Key Takeaways

- Reinforce critical assessment of AI-generated code
- Connect course learning with real-world development practices
- Define next steps for adopting AI within software development workflows