Code Analysis and Software Quality Training

Duration: **2 Day**

BİLGİNÇ
IT Academy
info@bilginc.com
bilginc.com

Overview

Code Analysis and Software Quality Training is a two-day online program designed to help software development teams create applications that are secure, sustainable, measurable, and easier to maintain. The course brings together Clean Code principles, static and dynamic code analysis, code metrics, code review practices, and secure software development methods within a unified approach to software quality.

Throughout the program, participants will learn how to recognize quality issues and structural weaknesses in source code, evaluate findings from code analysis tools, and incorporate quality controls into their development workflows. Key topics include identifying code smells, managing technical debt, measuring code quality, and determining areas that require improvement.

The training introduces practical methods for establishing consistent coding standards across teams, implementing systematic quality controls, and continuously monitoring software quality. Participants will gain the knowledge needed to apply these practices to existing projects and support more reliable and maintainable software development processes.

Prerequisites

Basic programming skills and a general understanding of software development processes are recommended. Advanced knowledge of a specific programming language or code analysis tool is not necessary to participate in this training.

Who Should Attend

- Software developers
- Software architects and technical team leaders
- Test engineers and quality assurance professionals
- DevOps and DevSecOps teams
- Managers overseeing software quality processes
- Professionals aiming to strengthen code review practices and software quality standards

Code Analysis and Software Quality Training

Duration: **2 Day**

BİLGİNÇ IT Academy
info@bilginc.com
bilginc.com

What You Will Learn

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

- Understand the relationship between software quality, maintainability, and technical debt.
- Apply Clean Code principles to develop readable and maintainable code.
- Identify code smells and structural quality issues.
- Use static and dynamic code analysis techniques.
- Evaluate code metrics to identify quality improvement opportunities.
- Apply secure coding principles.
- Understand Code Review and White-Box Testing practices.
- Integrate automated quality checks and Quality Gates into CI/CD pipelines.
- Interpret code analysis reports to support continuous software quality improvement.

Outline

1. Software Quality from an Organizational Perspective

- Understanding how software quality affects projects and organizations
- Principles of sustainable software development
- Recognizing quality risks during the early stages of development
- Organizational coding standards and software quality policies
- Examining the connections between software quality, cost, and technical debt

2. Clean Code and Maintainable Software Development

- Fundamental Clean Code principles
- Techniques for writing readable and maintainable source code
- Minimizing code duplication
- Applying meaningful naming conventions and appropriate code structures
- Recognizing code smells
- Evaluating when refactoring is necessary

3. Static Code Analysis

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

- Understanding static code analysis
- Detecting defects and quality issues without executing the application
- Examining code smells, structural weaknesses, and potential security vulnerabilities
- Configuring analysis rules and quality thresholds
- Reading and evaluating static analysis reports
- Incorporating static analysis into development workflows

4. Dynamic Code Analysis and Runtime Evaluation

- Comparing static and dynamic analysis approaches
- Observing application behavior during execution
- Evaluating performance, resource consumption, and memory-related problems
- Detecting runtime errors and potential risks
- Using analysis findings to guide software quality improvements

5. Code Metrics and Software Quality Standards

- Measuring code complexity and dependencies
- Evaluating readability and maintainability indicators
- Assessing testability and reusability
- Measuring and prioritizing technical debt
- Establishing consistent organizational coding standards
- Setting measurable objectives for software quality

6. Secure Code Development

- Fundamental principles of secure software development
- Applying security practices at the source code level
- Identifying common security mistakes and vulnerabilities
- Incorporating security controls into the software development lifecycle
- Coordinating software quality and security checks

7. Code Review and White-Box Testing

- Understanding the objectives and scope of code reviews

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

- Applying effective code review techniques
- Developing structured code review checklists
- Exploring White-Box Testing approaches
- Managing review feedback and corrective actions
- Encouraging a quality-focused culture within development teams

8. Code Quality in CI/CD Workflows

- The relationship between continuous integration and software quality
- Integrating automated quality checks into CI/CD pipelines
- Implementing Quality Gates
- Monitoring and reporting software quality indicators
- Establishing early warning mechanisms
- Improving organizational software quality maturity