Faysal Aziz

AI-Augmented Development

Mastering GitHub Copilot & Building an AI-Powered Dev Team - 2 days

FA-0082 · 2 days

Agentic & Generative AI · Software Development

Introduction

Software development is undergoing a structural shift. Tools like GitHub Copilot are no longer passive assistants; they are evolving into autonomous collaborators capable of writing, reviewing, and even shipping code. Modern teams are beginning to treat AI not as a feature, but as a distributed layer of capability embedded across the entire SDLC.

This course is designed to help developers move beyond autocomplete and into orchestration—learning how to work with AI agents as if they were members of the development team. Participants will explore how GitHub Copilot has evolved into a full-spectrum development partner, capable of handling tasks from coding to pull request generation and terminal workflows, and how to design workflows where multiple AI systems collaborate alongside humans.

The instructor brings over 30 years of industry experience and will focus entirely on real-world, production-grade practices rather than academic abstractions.

Learning outcomes

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

- Understand the evolution of AI-assisted development into agent-based workflows
- Effectively use GitHub Copilot across IDEs, terminals, and repositories
- Leverage Copilot's coding agent for autonomous task execution
- Design prompt strategies for reliable and secure code generation
- Integrate AI into the full software development lifecycle
- Orchestrate multiple AI tools as a cohesive "AI dev team"
- Establish governance, security, and quality controls for AI-generated code

- Measure productivity and ROI from AI adoption in development teams

---

Prerequisites

- Intermediate proficiency in at least one programming language
- Familiarity with Git, GitHub workflows, and pull requests
- Understanding of software development lifecycle (SDLC)
- Basic exposure to APIs and development tools (IDE, CLI)
- No prior AI experience required, but helpful

Training outline

1. Foundations of AI-Augmented Development

1. Evolution from code completion to autonomous agents
2. The shift from prompt-response to execution-based AI workflows
3. Overview of AI in modern developer ecosystems
4. Capabilities and limitations of large language models in coding
5. Developer productivity trends and AI adoption in 2026

2. Deep Dive into GitHub Copilot

1. Architecture and working principles of Copilot

   Copilot across environments

   1. IDE integrations (VS Code, JetBrains, Visual Studio)
   2. Terminal and CLI usage
   3. GitHub-native workflows

   Core capabilities

   1. Code generation and completion
   2. Code explanation and refactoring
   3. Test generation and debugging
   4. Documentation and summarization
4. Context awareness and multi-file reasoning
5. Model selection and optimization strategies
6. Understanding Copilot’s strengths vs weaknesses

3. Copilot Coding Agent and Autonomous Workflows

1. Introduction to coding agents
2. Assigning GitHub issues to Copilot agents
3. Autonomous pull request generation
4. Self-review and iterative refinement
5. Built-in security scanning and validation
6. CLI handoff and cross-environment execution
7. Custom agents and extensibility
8. Multi-agent collaboration within GitHub

4. Prompt Engineering for Developers

1. Structuring prompts for deterministic outputs
2. Context injection techniques
3. Iterative prompting and refinement loops

   Prompt patterns for

   1. Code generation
   2. Refactoring
   3. Testing
   4. Debugging
5. Reducing hallucinations and improving accuracy
6. Prompting for security and compliance

5. AI Across the Software Development Lifecycle

1. Planning and requirements generation
2. Architecture design with AI assistance
3. Coding and implementation workflows
4. Automated testing and QA
5. Code review and pull request automation
6. Documentation and knowledge sharing
7. Deployment and DevOps integration
8. Observability and monitoring with AI insights

6. Using AI as a Development Team

1. Concept of “AI teammates” vs tools

   Role-based AI usage

   1. AI as junior developer
   2. AI as reviewer
   3. AI as DevOps engineer
   4. AI as technical writer
3. Orchestrating multiple AI agents (Copilot, Claude, Codex)
4. Task delegation strategies
5. Parallelizing development with AI agents
6. Human-in-the-loop design patterns
7. Communication patterns between humans and AI

7. Advanced Team Workflows with AI

1. AI-driven backlog grooming
2. Automated issue resolution pipelines
3. AI-assisted sprint planning
4. Continuous integration with AI-generated code
5. AI in code review pipelines
6. Managing large codebases with AI context awareness
7. Integrating AI into enterprise development environments

8. Governance, Security, and Quality Control

1. Risks of AI-generated code
2. Common security vulnerabilities in generated code
3. Code validation and static analysis integration
4. Establishing review policies for AI contributions
5. Auditability and traceability of AI-generated changes
6. Managing intellectual property and licensing concerns
7. Organizational policies for AI usage

9. Performance Measurement and Optimization

1. Measuring developer productivity gains

2. Metrics for AI effectiveness
3. Developer experience and cognitive load
4. Cost vs value analysis of AI tools
5. Continuous improvement strategies
6. Feedback loops for AI system tuning

10. Future of AI in Software Development

1. Agentic development platforms
2. AI-native development environments
3. Emerging trends in multi-agent systems
4. The role of developers in an AI-first world
5. Strategic positioning for developers and organizations

© 2026 Faysal Aziz · Kuala Lumpur