BOTTEGA IT minds

Training program:

Software Craftsmanship with Generative AI - Hands-On training for TypeScript or Java+Spring Boot Developers or React

Info:

Name:	**Software Craftsmanship with Generative AI - Hands-On training for TypeScript or Java+Spring Boot Developers or React**
Code:	**ai-craft**
Category:	AI
Target audience:	architects developers
Duration:	3 days
Format:	20% lecture / 80% workshop

Artificial Intelligence is no longer just a tool for code generation; it is becoming a true partner in a programmer's daily work.

During these 3-day intensive workshops, you will learn how to effectively use Generative AI in the spirit of Software Craftsmanship.

Variants of the training:

- focus on backend programming in Java and Spring Boot
- focus on backend programming in TypeScript
- focus on frontend programming in React + TypeScript

Through a series of practical exercises, live coding sessions, and real-world use cases, you will:

- You will learn how to collaborate with AI to create clean, maintainable, and well-designed code.
- You will apply the Test Driven Development (TDD) approach and the Red-Green-Refactor cycle with AI support.
- You will discover how AI can support test creation, refactoring, and design decision-making.
- You will understand how to maximize productivity while maintaining full control over the codebase.
- You will learn about common pitfalls of working with AI and how to avoid them.

You will also explore architectural patterns and best practices that not only support building scalable and maintainable systems but also significantly facilitate collaboration with Generative AI:

- **Domain-Driven Design (DDD)** – allows organizing the domain model in a way that is understandable for AI, making interactions more contextual and accurate.
- **Ports and Adapters (Hexagonal Architecture)** – supports a clean separation of responsibilities, enabling AI to operate safely on isolated components without the risk of coupling.
- **Modularization** – ensures clear boundaries and responsibilities within the codebase, allowing AI to generate and refactor code in well-defined contexts.
- **Microservices Architecture** – enables distributed development and precise AI support at the level of individual services, supporting their independent evolution and deployment.

BO·TT·EGA
IT minds

It's all about the content.

- Domain-Driven Design (DDD): helps structure the domain model in a way that AI can understand and reason about, making interactions with AI more context-aware and meaningful
- Ports and Adapters (Hexagonal Architecture): promotes a clean separation of concerns, enabling AI to safely assist with isolated components without risking tight coupling
- Modularization: provides clear boundaries and responsibilities in the codebase, allowing AI to generate or refactor code within well-defined contexts
- Microservices Architecture: enables distributed development and targeted AI assistance on a per-service basis, supporting independent evolution and deployment

BOTTEGA IT minds

Training program

1. Fundamentals of Collaboration with Generative AI in a Developer's Daily Work

- 1.1. Introduction to Generative AI in the Context of Software Craftsmanship
  - 1.1.1. What is Generative AI and how it changes the way developers work
  - 1.1.2. Overview of AI tools for Java/Spring developers
  - 1.1.3. Ethics and responsibility when using AI in coding
- 1.2. Basics of Effective Collaboration with AI
  - 1.2.1. Writing effective prompts and communicating with AI
  - 1.2.2. The role of context, intent, and structure in working with AI
  - 1.2.3. Common mistakes and misunderstandings
- 1.3. Practical Exercises: AI as a Partner in Daily Work
  - 1.3.1. Creating business logic code with the help of AI
  - 1.3.2. Code correction and improvement with AI support
  - 1.3.3. Verifying results and the quality of generated code
  - 1.3.4. Implementing technical and integration elements

2. Quality Assurance – Testing and Architecture

- 2.1. Test-Driven Development (TDD) with AI Involvement
  - 2.1.1. Principles and values of TDD
  - 2.1.2. Red-Green-Refactor with AI assistance
  - 2.1.3. Generating unit and integration tests
  - 2.1.4. Generated tests vs. system correctness
- 2.2. Practical Exercises: AI as an Assistant in Maintaining Quality
  - 2.2.1. Refactoring unreadable code with AI
  - 2.2.2. Extending functionality using tests

BOTTEGA IT minds

2.2.3. Maintainable tests that protect against regression

3. Architectural Patterns and AI in Complex Systems

3.1. Domain-Driven Design and Contextual Understanding by AI

3.1.1. Introduction to DDD and its importance in working with AI

3.1.2. Building domain models with AI readability in mind

3.1.3. Co-creating aggregates, value objects, and domain services with AI

3.2. Hexagonal Architecture, Modularity, and Microservices

3.2.1. Benefits of Ports and Adapters in the context of AI

3.2.2. Creating and refactoring modules in large systems

3.2.3. Scaling solutions with AI support at the microservices level

3.3. Practical Applications and Implementation Strategy

3.3.1. Case study: extending an existing application with AI

3.3.2. Case study: building new applications with AI

3.3.3. Setting boundaries of responsibility between human and AI.

3.3.4. Plan for implementing AI into the production team.