BO·TT·EGA
IT minds

Training program:

AI-Assisted Domain-driven Design and business models archetypes

Info:

Name:	**AI-Assisted Domain-driven Design and business models archetypes**
Code:	**ddd-ddd**
Category:	Domain Driven Design and Event Storming
Target audience:	architects developers team_lead tech_lead po
Duration:	3 days
Format:	60% workshop/40% lecture

Modern AI-assisted development clearly shows that simple "vibe coding" does not work for projects whose complexity extends beyond prototypes and toy applications. Generative AI can produce code extremely quickly, but it only works reliably inside an environment that is understandable and constrained — built from well-defined modules, clear domain boundaries, and a sound architecture.

This is why certain engineering skills are now essential for using AI as a real productivity multiplier:

- Modularization, enabling delegation to agents in small, unambiguous contexts.
- Verification and critical evaluation of AI output based on engineering best practices.
- Designing autonomous models, APIs, and test layers that define stable boundaries for AI.
- Knowledge of business domain archetypes, allowing you to assemble systems like building blocks and prototype new solutions extremely quickly, with heavy help from generative AI.

AI (for now) does not think modularly, does not design architecture, and does not preserve systemic coherence. Its power emerges only when a human intentionally reduces context and defines the constraints within which an agent can operate. The engineer's role is to plan structure, define boundaries, specify contracts — and only then delegate implementation. In this model, AI becomes a massive productivity accelerator.

Throughout the course we use tools such as Claude Code to create precise system-level instructions and control AI through specifications rather than one-off prompts. Participants learn to build their own agent-driven workflows for module implementation, testing, domain analysis, and code review.

On the technical side, code generation is grounded in well-known tactical patterns and local layered architectures, ports and adapters, separation of concerns, and modularization techniques — creating an ideal environment for controlled use of AI. On the business side, participants work with proven domain archetypes such as Product, Pricing, Accounting/Billing, Party, and graph-based models. These archetypes create a shared "business language" through which AI can effectively collaborate with engineers and enable prototyping of real systems in hours, not months.

The training focuses on the practical design of systems that are both AI-friendly and agent-friendly. It teaches modern engineering practices that allow AI to be used effectively, safely, and deliberately across

large parts of the software development lifecycle — from domain exploration, through modular architecture and autonomous model implementation, to testing, verification, and rapid prototyping through business archetypes and agent workflows.

This combination of technical patterns, business archetypes, spec-driven development, and advanced Claude agents transforms AI from a curiosity into a production-ready engineering tool with real power.

It's all about the content.

- AI will finally behave predictably, because you will learn how to design modules, layers, and ports /adapters in a way that eliminates hallucinations and chaos in generated code.
- You will prototype systems in hours, using domain archetypes (Product, Pricing, Billing, Party, Graph-based models) that AI understands well and can implement instantly.
- You will build your own agent workflows, enabling Claude to implement modules, write tests, analyze domains, and perform code reviews based on a CLAUDE.md specification — instead of relying on ad-hoc prompts.
- You will control AI instead of guessing what it will do, by learning how to critically evaluate its output, detect architectural and domain flaws, and correct them effectively.

BOTTEGA IT minds

Training program

1. AI Assistance

- 1.1. What AI-assistance actually is
- 1.2. Human - AI collaboration model

2. Why Vibe Coding Doesn’t Work

- 2.1. Why vibe coding fails beyond prototype-level complexity
- 2.2. How system complexity breaks AI’s ability to maintain coherence
- 2.3. Context reduction as a fundamental principle
- 2.4. Practical exercise

3. Modularization for Safe AI Collaboration

- 3.1. Why modularization is essential in the AI era
- 3.2. The architect’s role: decisions AI cannot make
- 3.3. How to identify module and domain boundaries
  - 3.3.1. Single Source of Truth (SSOT)
  - 3.3.2. Pivoting domain events
- 3.4. Practical exercise

4. Implementing a Module with AI

- 4.1. First steps with Claude
- 4.2. CLAUDE.md specification
- 4.3. Layered architecture + Ports and Adapters with AI
- 4.4. Common pitfalls when implementing with AI
  - 4.4.1. Context reduction
  - 4.4.2. Planning phase as the key success factor
- 4.5. Typical AI-generated patterns:

BOTTEGA IT minds

- 4.5.1. Aggregates
- 4.5.2. Application Services
- 4.5.3. Value Objects
- 4.6. Debugging with AI
- 4.7. Practical exercise

5. Testing Strategies with AI

- 5.1. Where and how to use test-generating agents
- 5.2. Detecting untested scenarios and risk areas
- 5.3. Avoiding explosion of AI-generated tests
- 5.4. What counts as "observable behavior" for AI
- 5.5. Verifying AI-generated tests
- 5.6. TDD with AI
- 5.7. Practical exercise

6. Domain Exploration and Business Archetypes

- 6.1. Domain archetypes — the key to prototyping systems in hours
- 6.2. Archetypes covered in detail:
  - 6.2.1. Product
  - 6.2.2. Pricing
  - 6.2.3. Accounting/Billing
  - 6.2.4. Graph-based models
  - 6.2.5. Party
- 6.3. Expanding the domain — thinking like a product architect
- 6.4. Practical exercise: working with ready-made models in code

7. Advanced Agent Setup and Spec-Driven Development

- 7.1. Agents for domain/archetype analysis

BOTTEGA IT minds