BO·TT·EGA
IT minds

Training program:

Software Architecture For Frontend Developers

Info:

Name:	**Software Architecture For Frontend Developers**
Code:	**architecture-fe**
Category:	JS and Front-end
Target audience:	developers architects team_lead
Duration:	3-4 days
Format:	33% workshop, 33% praca w grupie, 33% ćwiczenia

The training is intended for experienced front-end developers who want and/or need to "go beyond" their framework and programming language and explore models, styles, patterns and principles of software architecture. The topics discussed are either strictly front-end related, or universal, or knowledge of which will significantly improve communication between front-end developers and back-end developers and architects.

The training program is a general framework - we precede specific training with a pre-training analysis.

It's all about the content.

- Architectural thinking (high-level) instead of code and framework thinking (low-level)
- Emphasis on understanding the essence of architectural patterns and styles
- Searching for the pros and cons of each solution, understanding trade-offs
- Matching the class of solution to the class of problem
- Emphasis on understanding the business context and its impact on the architecture

BOTTEGA IT minds

Training program

1. Architecture Introduction

- 1.1. Architectural drivers
- 1.2. Goals of Architecture
- 1.3. Architect as a role

2. Documenting Software Architecture

- 2.1. ADR (Architecture Decision Records)
- 2.2. RFC (Request For Comments)
- 2.3. C4 Model
- 2.4. Event Storming / Context Mapping

3. Architecture Styles and Patterns

- 3.1. Modular Monolith / Modulith
  - 3.1.1. Non Modular Monolith
- 3.2. Microservices
  - 3.2.1. Microservices-related patterns
  - 3.2.2. Distributed Monolith
  - 3.2.3. MicroFrontends
- 3.3. Micro Kernel
- 3.4. Hexagonal
- 3.5. CRUD
- 3.6. CQRS
- 3.7. Event Sourcing

4. Modularization

- 4.1. Types of Coupling

BOTTEGA IT minds

- 4.2. Cohesion
- 4.3. Encapsulation
- 4.4. Anti-Corruption Layer
- 4.5. Canonical Data Model
- 4.6. God Classes
- 4.7. Anemic Domain Model
- 4.8. Dependency Inversion
- 4.9. Law of Demeter
- 4.10. Inversion of Control
- 4.11. Hollywood Principle

5. Capabilities

- 5.1. Reusability
- 5.2. Scalability
- 5.3. Resilience
- 5.4. Availability
- 5.5. Fault Tolerance
- 5.6. Testability
- 5.7. Performance
- 5.8. Consistency

6. Integration (optional module)

- 6.1. Command, Event, Query
- 6.2. REST
- 6.3. CQRS vs API Composition
- 6.4. Messaging
- 6.5. Contracts

BOTTEGA IT minds

BOTTEGA IT minds