BO·TT·EGA
IT minds

Program szkolenia:

Tworzenie Modularnych Monolitów

Informacje:

Nazwa:	**Tworzenie Modularnych Monolitów**
Kod:	**NET-Monoliths**
Kategoria:	.NET
Odbiorcy:	developerzy, architekci
Czas trwania:	3 dni
Forma:	50% wykłady / 50% warsztaty

Zakres

Szkolenie przeznaczone dla projektantów i architektów, którzy chcą konstruować duże monolityczne aplikacje w nowoczesnym stylu tak aby ich utrzymanie i rozwój dawały radość a nie frustracje.

Przy małych i średnich projektach zastosowanie architektury mikroserwisowej nie jest opłacalne a pokładane nadzieje szybko sa rozwiewane dużymi kosztami rozwoju i utrzymania oprogramowania.

Kompleksowe podejście wytwarzania rozwiązań opartych o modularne monolity obejmuje: niezwykle istotny etap modelowania, pozwalający zaplanować architekturę na wysokim poziomie i wybrać odpowiednie style architektoniczne do proponowanych rozwiązań. Następnie uczestnicy implementują zaplanowane rozwiązanie napotykając na różne pułapki i przeszkody zaplanowane przez trenera odzwierciedlające problemy i wyzwania które pojawiają się codziennej pracy.
Forma

Przyrostowe stosowanie wiedzy w praktyce
Szkolenie prowadzone jest w formie, która łączy następujące po sobie na przemian wykłady, warsztaty i dyskusje.

Podczas wykładów trener prezentuje kolejne rozdziały wiedzy merytorycznej, uzupełniając je o komentarze na podstawie własnych doświadczeń.

Podczas warsztatów tworzymy rozwiązanie które z każdym zadaniem jest uzupełniane o nowe funkcjonalności ilustrujące zagadnienia teoretyczne poznane podczas poprzedzającego je wykładu.

Podczas dyskusji uczestnicy mają dostęp do wiedzy eksperckiej trenera oraz mają możliwość zweryfikowania swoich rozwiązań z wypracowanymi przez innych uczestników szkolenia.

BOTTEGA IT minds

Szczegółowy program:

1. Modelowanie strategiczne

- 1.1. Modelowanie granic
  - 1.1.1. Bounded context separacja i integracja
  - 1.1.2. Domain distillation techniki wyłaniania
  - 1.1.3. Context mapping mapowanie modeli
- 1.2. Zrozumienie problemu Event Storming
- 1.3. Dokumentowanie C4

2. Strukturyzowanie aplikacji

- 2.1. Enkapsulacja wspomagana kontenerami Ioc
- 2.2. Jak grupować kod w moduły
  - 2.2.1. Package by Layers
  - 2.2.2. Package by Feature
  - 2.2.3. Ports And Adapters
  - 2.2.4. Package by Component

3. Integracja modułów

- 3.1. Interfejsy komunikacyjne
  - 3.1.1. Command
  - 3.1.2. Query
  - 3.1.3. Event
- 3.2. Rodzaje operacji
  - 3.2.1. Asynchronous
  - 3.2.2. Synchronous
- 3.3. Wzorce wspomagające realizacje

BOTTEGA IT minds

- 3.3.1. Mediator pattern
- 3.3.2. Command-handler pattern

4. Zależności

- 4.1. Zależności cykliczne jak ich unikać
- 4.2. Zależności tranzytywne jak sobie z nimi radzić (repack)
- 4.3. Zarządzanie zależnościami za pomocą analiza statyczna

5. Moduły

- 5.1. Api gateway techniki integracji z światem zewnętrznym
  - 5.1.1. gRPC
  - 5.1.2. JsonRpc
  - 5.1.3. REST
- 5.2. Modelowanie skomplikowanej logiki domenowej
  - 5.2.1. Port and adapters
  - 5.2.2. Domain Driven Design Building blocks
- 5.3. CRUD szybkie prototypownie
  - 5.3.1. Podejście "Database First"
  - 5.3.2. Podejście "Code first"
  - 5.3.3. Code generation
- 5.4. Modelowanie skomplikowanych odczytów
  - 5.4.1. CQRS Command Query Responsibility Separation
  - 5.4.2. Techniki łączenia danych
    - 5.4.2.1. database side
    - 5.4.2.2. client side
- 5.5. Modelowanie długotrwałych procesów biznesowych
  - 5.5.1. Event sourcing long running

BOTTEGA IT minds

- 5.5.2. Orchestration
- 5.5.3. Choreography
- 5.6. Moduły narzędziowe (utility)
- 5.7. Moduły interfejsu użytkownika
- 5.7.1. composition ui
- 5.7.1.1. client side
- 5.7.1.2. server side

6. Modularność a przetrzymywanie danych

- 6.1. techniki separacji danych
- 6.2. zarządzanie strukturą danych
- 6.2.1. wersjonowanie
- 6.2.2. aktualizacja struktury
- 6.3. Integracja z narzędziami CI/CD

7. Modularność a elementy infrastrukturalne

- 7.1. Programowanie zorientowane na aspekty
- 7.2. Wzorce projektowe wspomagające
- 7.2.1. Dekorator
- 7.2.2. Proxy

8. Podejście do testów

- 8.1. integration tests how to write it
- 8.2. Complex Write
- 8.2.1. normal testing piramid (many unittest, few end to end)
- 8.2.2. BDD applys here
- 8.3. CRUD square (average unit tests, avr end to end)
- 8.4. Complex Read

BOTTEGA IT minds

8.4.1. reverse piramid (many end to end, few unit tests)