UNIWERSYTET WSB MERITO OPOLE

wcześniej Wyższa Szkoła Bankowa

Inżynieria Oprogramowania dla prędkości produktu

STUDIA PODYPLOMOWE

Program

9	192	12	12
Liczba miesięcy nauki	Liczba godzin zajęć	Liczba zjazdów	Liczba modułów

Inżynieria Produktowa & Analityka (32h)

- **Mentalność inżynierii produktowej i AAARR**
  Framework AAARR i rola inżyniera w cyklu product discovery-delivery. Mapowanie blokad technicznych w dostarczaniu wartości. Metryki AAARR jako input dla AI do analizy wartości biznesowej — AI analizuje dane o zachowaniach użytkowników i pomaga priorytetyzować co budować dalej.

- **Analityka produktowa i śledzenie zdarzeń**
  Projektowanie schematu zdarzeń, wzorce instrumentacji, implementacja śledzenia w kluczowych przepływach aplikacji. Budowa dashboardu metryk biznesowych. Łączenie metryk technicznych z wynikami biznesowymi.

Trunk-Based Development & Feature Flags (32h)

- **Migracja do Trunk-Based Development**
  Migracja z GitFlow na TBD. Analiza workflow — identyfikacja długich pętli feedbacku. Ciągła integracja, short-lived branches, commit bezpośrednio do maina. TBD daje natychmiastowy feedback przy pracy z AI: commit → test → wynik w minutach, nie dniach.

- **Feature flags i pipeline CI/CD**
  Implementacja feature flags (kill switch, gradual rollout, A/B test), techniki dark launch i branch by abstraction. Budowa pipeline CI/CD z równoległymi testami i quality gates. Feature flags jako safety net — kod pisany z AI trafia na produkcję za flagą, kill switch wyłącza zmianę natychmiast.

Automatyczne Testowanie dla Prędkości (32h)

- **Strategia testowania i testy jednostkowe**
  Piramida testów, koszty i trade-offy każdego poziomu. Dodawanie pokrycia testowego do nietestowanego kodu legacy. TDD w praktyce (Red-Green-Refactor). Testy automatyczne to natychmiastowa weryfikacja kodu generowanego z AI — im lepsze pokrycie, tym pewniej i szybciej pracujesz z AI.

- **Testy integracyjne, E2E i BDD**
  Testy integracyjne dla API, testy E2E dla krytycznych scenariuszy. BDD jako most między wymaganiem a kodem — specyfikacja Given-When-Then jest jednocześnie wymaganiem biznesowym, testem akceptacyjnym i instrukcją dla narzędzi AI.

Obserwowalność & Doskonałość Produkcyjna (32h)

- **Instrumentacja — logi, metryki, tracing**
  Trzy filary obserwowalności. Zamiana print statements na strukturalne logowanie. Metryki biznesowe i techniczne. Distributed tracing między komponentami. Obserwowalność daje pewność że zmiany działają poprawnie na produkcji — bez niej szybsze tempo pracy z AI to więcej niepewności.

- **Alerty, obsługa incydentów, symulacja**
  Projektowanie alertów (sygnał vs szum). Tworzenie runbooków. Symulacja incydentu produkcyjnego na żywo — wykrycie, diagnoza, rozwiązanie, post-mortem bez szukania winnych. Kompletny cykl bezpieczeństwa przy częstych wdrożeniach.

Modelowanie i Architektura Systemów (48h)

- **Strategiczny Domain-Driven Design**

Event Storming jako technika odkrywania domen i granic w systemie. Identyfikacja domen, subdomen i bounded contexts w aplikacji legacy. Wzorce mapowania kontekstów, praktyczne modelowanie domeny. Czyste granice kontekstów = lepszy kontekst dla narzędzi AI — developer pracujący z AI w dobrze zdefiniowanym bounded context dostaje znacznie trafniejsze wyniki.

- **Wzorce taktyczne DDD**

Transaction Script vs Domain Model — kiedy każdy jest odpowiedni. Agregaty, Encje, Value Objects. Domain Events i wzorzec Repository. Praktyczna refaktoryzacja kodu legacy używając wzorców taktycznych. Czysty model domeny = wspólny język między developerem a narzędziami AI.

- **Wzorce architektoniczne i ewolucja**

Monolit → Modularny monolit → Mikroserwisy: kiedy i dlaczego każdy ma sens. Framework decyzyjny. Architektura event-driven, szyny komunikatów, wzorce asynchroniczne, eventual consistency. Modularna architektura = mniejszy blast radius per zmianę — developer z AI pracujący w izolowanym module nie ryzykuje zepsucia reszty systemu.

AI-Assisted Development w Praktyce & Synteza (16h)

- **Wdrożenia, AI-assisted capstone i synteza**

Migracje baz danych bez przestoju (expand-contract), metryki DORA i kultura DevOps. Kulminacja kursu — studenci budują kompletną funkcję w przekształconym systemie z pełnym wsparciem AI: specyfikacja BDD → implementacja z AI → testy weryfikują → CI/CD deployuje za feature flag → obserwowalność monitoruje. Kontrast z dniem 1, gdzie AI niewiele pomagało. Synteza: mapa transformacji systemu i refleksja jak każda praktyka z weekendów 1-11 zwiększyła efektywność pracy z AI.