AI Assisted Development – tworzenie oprogramowania z pomocą AI (kod: AI-Assisted-dev)

Zapytaj o szczegóły
tel. 22 63 64 164
akademia@alx.pl

Najbliższe terminy
2026-12-17 (Zdalnie)
2026-12-17 (Warszawa)

Opis i cel szkolenia

Szkolenie **AI Assisted Development** prezentuje, jak świadomie wykorzystywać narzędzia AI w procesie tworzenia oprogramowania. Od analizy wymagań i pracy ze specyfikacją, przez generowanie i refaktoryzację kodu, aż po testy, code review, dokumentację i bezpieczeństwo.

Celem naszego szkolenia jest solidna i uporządkowana dawka wiedzy, praktyk, które pozwalają zwiększyć produktywność zespołu. Ograniczając przy tym ryzyka związane z halucynacjami modeli, niską jakością kodu, czy wyciekiem danych.

Podczas zajęć uczestnicy poznają m.in. sposoby dobierania modeli i narzędzi do konkretnych zadań programistycznych, pracę z agentami AI w środowiskach developerskich, zarządzanie kontekstem projektu oraz przygotowywanie dokumentacji „LLM-ready". Ważną częścią szkolenia są dobre praktyki pracy z kodem legacy, automatyzacja pętli: specyfikacja ☐ implementacja ☐ testy ☐ poprawki, a także wykorzystanie linterów, testów i skanerów typów jako technicznych zabezpieczeń przed błędami generowanymi przez AI. Szkolenie ma charakter praktyczny.

Czas trwania

2 dni, 9:00 - 17:00

Program

1. Ekosystem AI Assisted Development i dobór narzędzi
   - — Czym jest AI Assisted Development i jak zmienia codzienną pracę zespołów developerskich?
   - — Przegląd głównych klas narzędzi: modele konwersacyjne, agenci kodujący, rozszerzenia IDE, narzędzia code review i automatyzacji.
   - — Model-to-Task Mapping: jak dobierać model lub narzędzie do rodzaju zadania.
   - — Które narzędzia lepiej sprawdzają się przy refaktoryzacji, analizie logiki biznesowej, pisaniu testów, dokumentacji i pracy z kodem legacy?
   - — Różnice między pracą z modelem konwersacyjnym, asystentem w IDE i agentem wykonującym zadania w repozytorium.
   - — Praktyczne ograniczenia modeli: halucynacje, błędne założenia, problemy z kontekstem, nadmierna pewność odpowiedzi.
2. Pattern Recognition: analiza kodu i architektury z pomocą AI
   - — Wykorzystanie AI do rozpoznawania wzorców i antywzorców w kodzie.
   - — Identyfikacja code smells i miejsc wymagających refaktoryzacji.
   - — Analiza zależności w większych systemach i repozytoriach.
   - — Odtwarzanie logiki biznesowej na podstawie istniejącego kodu.
   - — Tworzenie mapy modułów, komponentów i zależności na potrzeby dalszych prac.
   - — Wykorzystanie AI do przygotowania technicznego opisu fragmentu systemu.
3. Zarządzanie kontekstem i dokumentacja LLM-ready
   - — Context Window Management: jak dostarczać modelowi właściwy kontekst bez zalewania go zbędnymi informacjami.

- RAG, long context i praca na plikach projektowych — kiedy które podejście ma sens?
- Jak pisać README, pliki .md i dokumentację techniczną, aby były użyteczne dla zespołu i dla modeli AI.
- Tworzenie dokumentacji domenowej: pojęcia, reguły biznesowe, wyjątki, ograniczenia.
- Definiowanie standardów projektu: styl kodu, architektura, konwencje nazewnicze, zasady testowania.
- Wymuszanie na AI stosowania wzorców projektowych i standardów zespołu.
- Przykładowe pliki kontekstowe dla agentów i środowisk developerskich.

4. Praca z agentami AI i środowiskami developerskimi
   - Jak efektywnie współpracować z agentem AI w IDE i repozytorium.
   - Tryb planowania: analiza problemu przed rozpoczęciem implementacji.
   - Dzielenie zadania na małe, kontrolowalne kroki.
   - Tworzenie własnych promptów, instrukcji projektowych i reużywalnych workflowów.
   - Automatyzacja powtarzalnych zadań developerskich.
   - Praca z plikami kontekstowymi, np. regułami projektu, instrukcjami dla agenta i opisem architektury.
   - Kontrola zmian generowanych przez AI: diff, commit, review, rollback.
5. Specification-Driven Development z wykorzystaniem AI
   - Praca od specyfikacji do implementacji: dlaczego AI wymaga precyzyjnego opisu zadania.
   - Tworzenie specyfikacji funkcjonalnej i technicznej dla modelu.
   - Przekładanie wymagań biznesowych na zadania developerskie.
   - Generowanie szkicu rozwiązania, planu implementacji i listy ryzyk.
   - Weryfikacja założeń przed wygenerowaniem kodu.
   - Przykładowy workflow: wymaganie biznesowe ☐ specyfikacja ☐ implementacja ☐ testy ☐ code review.
   - Jak zadawać pytania modelowi, aby nie „dopowiadał” nieistniejących wymagań.
6. Jakość kodu, testy i minimalizowanie halucynacji
   - Typowe błędy w kodzie generowanym przez AI.
   - Jak wymuszać analizę problemu i uzasadnienie decyzji technicznych przed implementacją.
   - Pętla automatycznej poprawy: AI generuje kod ☐ testy wykrywają błąd ☐ AI proponuje poprawkę.
   - Integracja AI z testami jednostkowymi, integracyjnymi i regresyjnymi.
   - Generowanie brakujących testów dla istniejącego kodu.
   - Wykorzystanie linterów, formatterów, skanerów typów i narzędzi CI jako bariery bezpieczeństwa.
   - Rola człowieka w ocenie jakości: co można automatyzować, a czego nie warto oddawać modelowi.
7. AI w pracy z kodem legacy
   - Bezpieczne wprowadzanie zmian w systemach słabo udokumentowanych.
   - Analiza fragmentów starego kodu i odtwarzanie ich intencji.
   - Generowanie dokumentacji technicznej dla istniejących modułów.
   - Tworzenie testów zabezpieczających przed refaktoryzacją.
   - Planowanie migracji do nowszych wersji bibliotek, frameworków lub standardów języka.
   - Identyfikacja miejsc wysokiego ryzyka przed rozpoczęciem zmian.
   - Jak nie dopuścić do sytuacji, w której AI „upraszcza” kod, niszcząc ukrytą logikę biznesową.

Zapytaj o szczegóły
tel. 22 63 64 164
akademia@alx.pl

Najbliższe terminy
2026-12-17 (Zdalnie)
2026-12-17 (Warszawa)

8. Workflows & Best Practices: dzień z życia dewelopera pracującego z AI
   - Scenariusze pracy z AI w typowym cyklu developerskim.
   - Implementacja nowej funkcjonalności z pomocą AI.
   - Refaktoryzacja i poprawa czytelności kodu.
   - Przygotowanie code review z wykorzystaniem AI.
   - Debugowanie błędów z pomocą modelu.
   - Tworzenie dokumentacji zmian i opisów technicznych.
   - Świadomy użytkownik AI vs. bezrefleksyjne kopiowanie kodu.
   - Jak budować zespołowe standardy korzystania z AI w codziennej pracy.
9. Bezpieczeństwo w AI Assisted Development
   - Security-first development w projektach wspieranych przez AI.
   - Wykrywanie podatności w kodzie generowanym przez model i przez człowieka.
   - Przykłady typowych problemów: SQL Injection, XSS, błędy autoryzacji, niewłaściwa walidacja danych.
   - Prompt injection w aplikacjach korzystających z modeli językowych.
   - Ryzyka związane z dynamicznym budowaniem zapytań do modeli wewnątrz aplikacji.
   - Ochrona danych osobowych, danych firmowych, kluczy API i sekretów projektowych.
   - Zasady bezpiecznego korzystania z publicznych i firmowych narzędzi AI.
10. Odpowiedzialność, governance i standardy organizacyjne
    - Nowa rola dewelopera: odpowiedzialność za produkt w świecie, w którym część kodu generuje AI.
    - Code review po nowemu: jak zmienia się kontrola jakości kodu.
    - Kiedy można używać AI, a kiedy powinno to być zabronione lub ograniczone?
    - Wewnętrzne zasady pracy z AI w zespole developerskim.
    - Ryzyka licencyjne i organizacyjne związane z kodem generowanym przez AI.
    - Dokumentowanie użycia AI w procesie wytwarzania oprogramowania.

Domyślnie (dla grup ogólnodostępnych) szkolenie realizujemy z wykorzystaniem modeli openai, czasem dodatkowo prezentując pracę z alternatywami (np. lokalnymi oss)

Szkolenie realizujemy zarówno w trybie ogólnodostępnym (każdy może się zapisać - w terminach opublikowanych na stronie), jak również na zamówienie dla grup z firm oraz instytucji. Czy to w formie interaktywnych warsztatów, czy prelekcji, również konferencyjnych (dla większych wydarzeń).

Przeznaczenie i wymagania

Szkolenie jest przeznaczone dla osób technicznych oraz osób decyzyjnych zaangażowanych w proces tworzenia oprogramowania: programistów, tech leadów, architektów, engineering managerów, liderów zespołów IT oraz osób odpowiedzialnych za standardy technologiczne w organizacji.

Od uczestników wymagana jest podstawowa znajomość procesu tworzenia oprogramowania i doświadczenie praktyczne w pracy z kodem, projektami IT lub zespołami developerskimi. Szkolenie nie jest kursem programowania od zera — zakładamy, że uczestnicy rozumieją pojęcia takie jak repozytorium, testy, code review, API, framework, czy środowisko IDE. Nie jest natomiast wymagana wcześniejsza, a w szczególności zaawansowana, znajomość narzędzi AI.

W celu możliwości aktywnej pracy na zajęciach, będziemy oczekiwali od uczestników posiadania aktywnego abonamentu (np. github copilot, chatgpt,

Zapytaj o szczegóły
tel. 22 63 64 164
akademia@alx.pl

Najbliższe terminy
2026-12-17 (Zdalnie)
2026-12-17 (Warszawa)

claude). Więcej informacji podamy przed zajęciami (będzie potrzebna karta płatnicza, należy spodziewać się kosztów na poziomie ok. 20-30 dolarów; dla grup, w razie potrzeby możemy zorganizować licencje i konta uczestnikom na czas szkolenia).

Certyfikaty

Uczestnicy szkolenia otrzymują imienne certyfikaty sygnowane przez ALX.

Lokalizacje

— Warszawa – ul. Jasna 14/16A
— Zdalnie – zajęcia realizowane poprzez platformę Zoom
— Kraków – ul. św. Filipa 23
— Warsaw (English) – Jasna 14/16A
— Online (English) – your home, office or wherever you want
— na życzenie dowolne miejsce w Polsce, lub UE (zajęcia prowadzone w języku angielskim)

Cena szkolenia

1790 PLN netto (VAT 23%)

W cenę szkoleń organizowanych w naszej siedzibie wliczone są:

— autorskie materiały szkoleniowe,
— indywidualne stanowisko komputerowe do pracy podczas zajęć,
— certyfikaty ukończenia szkolenia,
— drobny poczęstunek oraz ciepłe i zimne napoje,
— możliwość jednorazowego kontaktu z instruktorem (instruktorami) po szkoleniu i zadawania pytań dotyczących materiału szkolenia.

Cena szkolenia nie zawiera obiadów. Można je dokupić w cenie 35 zł netto za obiad.

Zapytaj o szczegóły
tel. 22 63 64 164
akademia@alx.pl

Najbliższe terminy
2026-12-17 (Zdalnie)
2026-12-17 (Warszawa)