# JavaScript Developer II (kod: K-JS-2)

## Opis i cel kursu

JavaScript nie bez powodu jest jednym z najpopularniejszych języków programowania na świecie. Jest w zasadzie niezbędny do tworzenia nowoczesnych aplikacji internetowych, ale może być również wykorzystywany w wielu innych typach aplikacji. Nasz kurs jest przeznaczony dla osób, które chcą rozwinąć swoją znajomość JavaScript poza podstawowe zastosowania i nauczyć się tworzyć przy jego wykorzystaniu bardziej rozbudowane aplikacje. Poznajemy TypeScript, React z TypeScriptem, projektowanie i architekturę większej aplikacji SPA, zarządzanie stanem, testy i dokumentowanie komponentów, wdrożenie aplikacji, Next.js oraz dodawanie funkcji AI do aplikacji. Kurs jest drugą częścią wariantu XL kursu JavaScript i React (Frontend Developer), ale można go kupić również osobno.

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

**Najbliższe terminy**
2026-12-10 (Zdalnie)
2026-12-10 (Warszawa)
2026-12-19 (Zdalnie)
2026-12-19 (Warszawa)
2027-03-13 (Zdalnie)
2027-03-13 (Warszawa)
2027-04-01 (Zdalnie)
2027-04-01 (Warszawa)

## Program

1. **Rynek, aktualne oczekiwania i rola AI we frontendzie**
   - Obecne oczekiwania wobec front-end developera
   - Co jest dziś standardem: JavaScript, TypeScript, React, Git, praca z API, testy, podstawy deployu
   - Co się zdeaktualizowało lub zeszło na dalszy plan
   - Co jest aktualnie na topie: TypeScript, React, Next.js, Vite, komponentowe UI, testowanie, Storybook, AI-assisted development
   - Rola AI w pracy developera
   - Ograniczenia AI: halucynacje, bezpieczeństwo, błędy w kodzie, konieczność review
   - Omówienie projektu końcowego kursu
   - Ćwiczenie I: analiza przykładowej oferty pracy frontendowej i mapowanie wymagań na tematy kursu.
2. **Node.js, npm, edytor z AI i workflow projektowy**
   - Node.js jako środowisko narzędziowe dla frontendu
   - Menedżer npm, package.json, skrypty projektowe
   - Instalacja zależności
   - Vite jako środowisko projektowe
   - Zmienne środowiskowe i pliki .env
   - Konfiguracja edytora z asystentem AI (np. Cursor) do pracy z projektem
   - Podstawowa struktura projektu frontendowego
   - Konfiguracja ESLint i Prettier
   - Ćwiczenie II: utworzenie projektu React/Vite, konfiguracja edytora z AI i podstawowego workflow developerskiego.
3. **Git, GitHub i praca projektowa**
   - Instalacja i konfiguracja Git
   - Repozytorium lokalne i zdalne
   - Polecenia commit, branch, merge, pull, push
   - Praca na branchach
   - Pull request - idea i uproszczony workflow
   - Rozwiązywanie prostych konfliktów przy łączeniu gałęzi
   - README projektu
   - Dobre praktyki commitowania
   - Ćwiczenie III: założenie repozytorium, pierwszy branch, prosty pull request i rozwiązanie konfliktu.
4. **Zaawansowany JavaScript**
   - Obsługa błędów: try/catch, throw, własne błędy
   - Callbacki i callback hell

- Promises
- Składnia async/await
- Promise.all, Promise.allSettled
- Event loop - koncepcyjnie i praktycznie
- Mikro- i makro-zadania
- Moduły ES: import/export
- Organizacja kodu w modułach
- Praktyczne wzorce pracy z kodem asynchronicznym
- Ćwiczenie IV: refaktoryzacja kodu z callbacków do Promise i async/await.

5. **TypeScript w aplikacjach frontendowych**
   - Po co TypeScript w większych aplikacjach frontendowych
   - Różnice między JavaScript i TypeScript
   - Typy podstawowe
   - Typowanie zmiennych
   - Typowanie funkcji: parametry i wartości zwracane
   - Typowanie tablic i obiektów
   - Typy: type vs interface
   - Union types
   - Literal types
   - Optional properties
   - Type narrowing - zawężanie typów
   - Type guards
   - Generics - podstawy
   - Utility types: Pick, Omit, Partial, Record
   - Typowanie danych z API
   - Typowanie propsów komponentów React
   - Dobre praktyki pracy z TypeScriptem w projekcie frontendowym
   - Ćwiczenie V: przygotowanie typów dla modeli danych aplikacji oraz typowanie funkcji i komponentów.
6. **Komunikacja client-server i bezpieczeństwo**
   - REST API w praktyce
   - Metody HTTP, statusy, nagłówki
   - JSON i struktury odpowiedzi API
   - Obsługa błędów API
   - Uwierzytelnianie i autoryzacja - koncepcyjnie
   - Tokeny, sesje, JWT - podstawy
   - Przechowywanie tokenów - ryzyka i dobre praktyki
   - CORS - koncepcyjnie
   - Mock backend / fake API
   - Krótkie omówienie GraphQL i WebSockets: kiedy się ich używa, bez głębokiego wdrożenia
   - Ćwiczenie VI: połączenie aplikacji z mock API i przygotowanie prostego przepływu logowania/autoryzacji.
7. **React z TypeScriptem - komponenty, state i hooks**
   - Myślenie komponentowe
   - JSX i renderowanie UI
   - Props i typowanie propsów
   - Renderowanie warunkowe
   - Renderowanie list
   - Obsługa zdarzeń
   - State komponentu
   - Aktualizacja obiektów i tablic w stanie
   - Formularze kontrolowane
   - Lifting state up

**Zapytaj o szczegóły**

tel. 22 63 64 164
akademia@alx.pl

**Najbliższe terminy**

2026-12-10 (Zdalnie)
2026-12-10 (Warszawa)
2026-12-19 (Zdalnie)
2026-12-19 (Warszawa)
2027-03-13 (Zdalnie)
2027-03-13 (Warszawa)
2027-04-01 (Zdalnie)
2027-04-01 (Warszawa)

- Kompozycja komponentów
- Context API - kiedy wystarczy, a kiedy lepszy jest Redux Toolkit
- Hook useEffect i synchronizacja z zewnętrznymi systemami
- Hook useRef
- Custom hooks
- Organizacja komponentów w większej aplikacji
- Ćwiczenie VII: zbudowanie bazowych komponentów aplikacji w React + TypeScript.

8. **Analiza wymagań, Atomic Design i dekompozycja UI**
   - Definiowanie wymagań aplikacji
   - User stories
   - Analiza widoków i przepływów użytkownika
   - Podstawy pracy z makietą
   - Przejście od makiety do komponentów
   - Wprowadzenie do Atomic Design
   - Atomy, molekuły, organizmy, template'y/widoki
   - Dekompozycja komponentowa na podstawie mockupów
   - Uwzględnianie accessibility i WCAG na etapie projektowania UI
   - Przygotowanie backlogu / listy zadań na bazie user stories
   - Ćwiczenie VIII: analiza wymagań, praca z makietą i dekompozycja UI zgodnie z Atomic Design.
9. **Architektura aplikacji SPA**
   - Projektowanie struktury folderów
   - Podział aplikacji na warstwy: UI, logika, API, routing, stan
   - Routing aplikacji
   - Organizacja komponentów, widoków i modułów
   - Separacja logiki biznesowej od warstwy prezentacyjnej
   - Organizacja komunikacji z API
   - Obsługa stanów aplikacji: loading, error, success
   - Przygotowanie aplikacji pod dalszą rozbudowę
   - Planowanie implementacji większej funkcjonalności
   - Ćwiczenie IX: zaprojektowanie architektury aplikacji SPA i przygotowanie struktury projektu.
10. **Stan aplikacji, Redux Toolkit i integracja z API**
    - Local state vs global state
    - Kiedy globalny store ma sens
    - Redux Toolkit - idea i podstawy
    - Store
    - Slices
    - Actions i reducers
    - Typowanie Redux Toolkit w TypeScript
    - Asynchroniczna komunikacja z API
    - Loading/error/success state
    - Autoryzowane zapytania do API
    - Organizacja logiki aplikacji
    - Ćwiczenie X: dodanie globalnego stanu i integracji z API do aplikacji SPA.
11. **AI w zaawansowanej aplikacji frontendowej**
    - Jak działają LLM API: klucze, endpointy, format żądań
    - Bezpieczne korzystanie z kluczy API
    - Backend/proxy/serverless jako pośrednik między frontendem a LLM API
    - Łączenie z API wybranego dostawcy modeli, np. OpenAI
    - Wysyłanie wiadomości i odbieranie odpowiedzi
    - Przekazywanie kontekstu aplikacji do modelu
    - Streaming odpowiedzi: ReadableStream, efekt pisania w UI

**Zapytaj o szczegóły**

tel. 22 63 64 164
akademia@alx.pl

**Najbliższe terminy**

2026-12-10 (Zdalnie)
2026-12-10 (Warszawa)
2026-12-19 (Zdalnie)
2026-12-19 (Warszawa)
2027-03-13 (Zdalnie)
2027-03-13 (Warszawa)
2027-04-01 (Zdalnie)
2027-04-01 (Warszawa)

- Przechowywanie historii konwersacji w JS - tablica messages
- Obsługa stanów: loading, error, success
- Dodanie funkcji AI do budowanej aplikacji SPA
- Ćwiczenie XI: dodanie funkcji AI do aplikacji, np. generatora opisu, podsumowania, sugestii lub asystenta kontekstowego.

12. **Storybook i dokumentowanie UI**
    - Po co dokumentować komponenty wizualnie
    - Wprowadzenie do Storybooka
    - Stories dla komponentów
    - Warianty komponentów
    - Dokumentowanie propsów i stanów komponentów
    - Komponenty w izolacji
    - Dokumentowanie komponentów powstałych dla funkcji AI
    - Storybook jako narzędzie komunikacji w zespole
    - Ćwiczenie XII: przygotowanie Storybooka dla wybranych komponentów aplikacji.
13. **Testowanie aplikacji frontendowej**
    - Po co testować frontend
    - Testy jednostkowe logiki JS/TS
    - Testy komponentów React
    - React Testing Library - podstawy
    - Mockowanie API
    - Testowanie loading/error/success state
    - Testowanie wybranej logiki funkcji AI
    - Testy E2E - koncepcyjnie
    - Playwright/Cypress - krótkie omówienie
    - Co testować, a czego nie testować
    - Ćwiczenie XIII: napisanie podstawowych testów dla logiki, komponentu i wybranego przepływu użytkownika.
14. **Wdrożenie aplikacji na Netlify**
    - Build aplikacji
    - Zmienne środowiskowe na produkcji
    - Deploy aplikacji SPA na Netlify
    - Konfiguracja projektu po stronie Netlify
    - Obsługa routingu SPA po wdrożeniu
    - Typowe problemy po deployu
    - Sprawdzenie działania aplikacji produkcyjnej
    - Ćwiczenie XIV: wdrożenie aplikacji na Netlify.
15. **Next.js - SSR, hydratacja i prosty projekt**
    - Czym Next.js różni się od klasycznej aplikacji React SPA
    - Routing w Next.js
    - Client-side rendering vs server-side rendering
    - SSR
    - SSG - krótkie omówienie
    - Hydratacja - o co chodzi i dlaczego ma znaczenie
    - Server Components - koncepcyjnie
    - Client Components
    - Fetchowanie danych w Next.js
    - Layouty
    - SEO w aplikacjach renderowanych po stronie serwera
    - Metadane, struktura treści i wpływ renderowania na indeksowanie
    - Jamstack i headless CMS - krótkie omówienie
    - Prosty projekt w Next.js
    - Kiedy wybrać klasyczne SPA, a kiedy Next.js

**Zapytaj o szczegóły**

tel. 22 63 64 164
akademia@alx.pl

**Najbliższe terminy**

2026-12-10 (Zdalnie)
2026-12-10 (Warszawa)
2026-12-19 (Zdalnie)
2026-12-19 (Warszawa)
2027-03-13 (Zdalnie)
2027-03-13 (Warszawa)
2027-04-01 (Zdalnie)
2027-04-01 (Warszawa)

— Ćwiczenie XV: stworzenie prostego projektu Next.js z routingiem i pobieraniem danych.

## Przeznaczenie i wymagania

Dla osób, które znają podstawy JavaScriptu i chcą pracować z nowoczesnym frontendem: absolwentów naszego kursu podstawowego, osób, które pisały już skrypty na stronach WWW, programistów innych języków dołączających do projektów frontendowych oraz testerów automatyzujących.

Oczekujemy znajomości takich pojęć jak zmienna i funkcja oraz umiejętności pisania prostych programów z wykorzystaniem pętli i struktur danych (np. tablic). Kurs jest naturalnym dalszym krokiem w nauce od podstaw, np. po naszym kursie JavaScript i React (Frontend Developer), lub po napisaniu kilku samodzielnych programów.

Na tym kursie nie uczymy się podstaw programowania jako takiego. Skupiamy się na głębszym poznawaniu JavaScriptu, TypeScriptu i Reacta.

## Certyfikaty

Uczestnicy szkolenia otrzymują imienne certyfikaty sygnowane przez ALX.

**Zapytaj o szczegóły**

tel. 22 63 64 164
akademia@alx.pl

**Najbliższe terminy**

2026-12-10 (Zdalnie)
2026-12-10 (Warszawa)
2026-12-19 (Zdalnie)
2026-12-19 (Warszawa)
2027-03-13 (Zdalnie)
2027-03-13 (Warszawa)
2027-04-01 (Zdalnie)
2027-04-01 (Warszawa)