PROGRAM SZKOLENIA APLIKACJE WEBOWE

Szkolenie WebSockets

Zbudujesz aplikację czasu rzeczywistego na WebSockets, odporną na zerwane połączenia.

CZAS TRWANIA	POZIOM	FORMA
2 dni	**średniozaawansowany**	**warsztat dla zespołu, zdalnie lub stacjonarnie**

Czat, powiadomienia i panele na żywo wymagają danych bez odświeżania strony. Kłopoty zaczynają się później: połączenie zrywa się w pociągu, wiadomości giną, a druga instancja serwera nie wie o klientach pierwszej.

Na szkoleniu budujesz taką aplikację najpierw na natywnym API, potem z Socket.IO, a ten sam kanał powiadomień piszesz też w SSE, żeby porównać koszt obu rozwiązań. Przez cały czas podglądasz ruch sieciowy, więc wiesz, jak wygląda komunikacja między klientami a serwerem. Powiadomienia na żywo jako część projektu API znajdziesz na szkoleniu API: REST i GraphQL.

PROGRAM

2 dni

- Komunikacja w czasie rzeczywistym
  - Polling, long polling, Server-Sent Events, WebSockets
  - Kiedy WebSockets, a kiedy wystarczy SSE
  - Ćwiczenie: kanał powiadomień w Server-Sent Events
  - WebTransport: co daje i kiedy warto go rozważyć
- Protokół WebSocket
  - Handshake i upgrade połączenia HTTP
  - Ramki, ping/pong, zamykanie połączenia
  - Szyfrowane połączenie `wss://`
- Natywne API
  - Obiekt `WebSocket` w przeglądarce
  - Wbudowany klient `WebSocket` w Node.js (stabilny od 22.4)
  - Serwer w Node.js z biblioteką `ws`
  - Format wiadomości: JSON i dane binarne
- Podgląd ruchu i testy
  - Zakładka Network i podgląd wiadomości w DevTools
  - Testowanie serwera z linii poleceń (websocat, wscat)
  - Testy automatyczne: mock połączenia w Playwright (`routeWebSocket`)

- Socket.IO
  - Zdarzenia, potwierdzenia (acknowledgements)
  - Pokoje i przestrzenie nazw
  - Rozgłaszanie do wybranych klientów (broadcast)
- Niezawodność
  - Ponowne łączenie i backoff
  - Odtwarzanie stanu: ręcznie i przez connection state recovery w Socket.IO
  - Heartbeat: ping/pong serwera i wiadomości aplikacyjne klienta
  - Wykrywanie martwych połączeń
  - Backpressure i `bufferedAmount`
- Skalowanie
  - Pokaz: wiele instancji serwera z adapterem Redis Streams, który w przeciwieństwie do klasycznego adaptera Redis wspiera connection state recovery
  - Load balancer i sticky sessions (kiedy są potrzebne)
- Bezpieczeństwo
  - Uwierzytelnianie połączenia (token, cookie)
  - Weryfikacja nagłówka `Origin`
  - Walidacja wiadomości i limity

DLA KOGO?

- Dla programistów, którzy budują czat, powiadomienia lub panel na żywo.
- Dla zespołów, które mają problem z gubionymi połączeniami.

CO MUSISZ UMIEĆ?

- Programować w JavaScript, w tym `Promise` i `async` / `await`.
- Uruchomić prosty serwer w Node.js.

CZEGO SIĘ NAUCZYSZ?

- Dobierzesz technologię (SSE, WebSocket, WebTransport) do problemu.
- Zbudujesz klienta na natywnym API i serwer w Node.js z biblioteką `ws`.
- Podejrzysz i zdebugujesz wiadomości w DevTools i z linii poleceń.
- Napiszesz test, który symuluje zerwane połączenie.
- Zorganizujesz komunikację w pokoje i wyślesz wiadomość do wybranej grupy.
- Obsłużysz zerwane połączenie bez utraty wiadomości i z czytelnym stanem w interfejsie.
- Zobaczysz, jak uruchomić aplikację na kilku instancjach serwera z Redisem.
- Zabezpieczysz połączenie przed nieautoryzowanym dostępem.

TRENER

Piotr Kowalski

W branży IT od 2008 roku. Od 2014 jest mentorem dla wielu developerów. Jako trener spędził na sali szkoleniowej kilka tysięcy godzin, a na LinkedIn ma 52 rekomendacje. Każde jego szkolenie to dawka solidnej wiedzy oraz humoru.

Więcej: piecioshka.pl/piotr-kowalski

ZORGANIZUJMY TO SZKOLENIE W TWOIM ZESPOLE

Program dopasuję do Waszego projektu i poziomu grupy. Napisz, z czym pracujecie, a odpowiem z propozycją terminu i zakresu.

KONTAKT **piecioshka.pl/kontakt**
E-MAIL **piecioshka+blog [małpka] gmail.com**
SZKOLENIE **piecioshka.pl/szkolenia/websockets/**