

Ewolucja technologii frontendowych

Czyli czym są aplikacje “mikrofrontendowe” i dlaczego mogą być dobrą alternatywą dla tradycyjnego monolitu?

Tomasz Jankiewicz 19365

Wprowadzenie

Tworzenie aplikacji internetowych przez ostatnie trzydzieści lat przeszło długą drogę. Pomimo tego, że podstawowe elementy budujące front aplikacji (HTML, CSS oraz JS) zmieniły się niewiele, to rewolucję przeżyły narzędzia pomocnicze, które ułatwiają ich obsługę i integrację. Te narzędzia znacząco wpłynęły na kulturę pracy, umożliwiając szybsze budowanie większych i bardziej skomplikowanych projektów. Do takich narzędzi należą popularne frameworki i biblioteki, takie jak Angular, Vue czy React, wraz z ich różnymi wariantami. Dzięki nim komercyjne projekty mogą w krótkim czasie stać się dużymi, złożonymi systemami, które są wielofunkcyjne i zależne od wielu zewnętrznych elementów.

W tradycyjnym podejściu do pracy nad takim projektem, zespoły zajmujące się rozwojem i utrzymaniem projektu pracują nad wspólnym repozytorium kodu oraz w jednej określonej technologii. To sprawdzone podejście, ale niesie ze sobą wiele problemów technicznych i logistycznych. Między innymi problem z wdrażaniem zmian, co wymaga często zatrzymania całej aplikacji. Jednym ze sposobów rozwiązania problemu monolitycznych projektów jest podejście modułowe, a jednym z przykładów takiego podejścia są aplikacje mikrofrontendowe.

Czym są mikrofrontendy?

Mikrofrontendy (ang. Microfrontends) wywodzą się z idei mikrouslug stosowanych w technologiach backendowych. Dziedziczą z niej nazwę łatwo kojarzącą się przez deweloperów, oraz podejście do tworzenia aplikacji składających się z wielu niezależnych modułów, które są odpowiedzialne za wykonanie pojedynczych zadań. Podział problemu na mniejsze i łatwiejsze do zarządzania elementy jest znaczącą praktyką programistyczną, a aplikacje mikrofrontendowe podnoszą tę praktykę na wyższy poziom.

W tradycyjnym podejściu do projektów, mającym jedno duże repozytorium, zespoły deweloperskie pracują nad różnymi fragmentami tego projektu. Choć taka struktura może pozwolić na pracę wielu zespołów, aplikacja i jej fragmenty są silnie ze sobą powiązane i zależne. Zespół "A" musi pracować w tej samej technologii co zespół "B", a oba zespoły muszą stale utrzymywać synchronizację swojego kodu. Dodatkowo, oba zespoły mogą chcieć wprowadzić swoje zmiany do wersji produkcyjnej w różnym czasie, co może prowadzić do blokowania pracy innych zespołów lub wydłużać czas, w którym aplikacja jest niedostępna dla klientów.

Przytoczone sytuacje stanowią skrajne przykłady, ale mogą wystąpić w przypadku pracy nad dużym i skomplikowanym projektem. Dlatego warto rozważyć alternatywne metody pracy.

Aplikacje mikrofrontendowe są właśnie taką alternatywą, podejściem do pracy nad dużymi projektami. Mikrofrontendy są osobnymi aplikacjami, posiadającymi niezależne repozytoria i indywidualne podejście do rozwiązywania problemów, w tym wybór technologii, z jakiej zespół się najlepiej czuje. Takie podejście zapewnia ogromną niezależność podczas rozwoju i wdrażania mikroaplikacji jako małych, niezależnych elementów większego projektu.

Można odnieść wrażenie, że mikrofrontendy wymagają nowego sposobu pisania aplikacji, ale to jest mylne przekonanie. Zasada tworzenia poszczególnych elementów pozostaje taka sama, choć istnieją kilka istotnych wskazówek, które pomogą stworzyć aplikację, która łatwo połączy się z innymi elementami w większą całość, podobnie jak układanka.

Jakie wymagania musi spełnić aplikacja, aby stała się microforntendem, lub była na nich oparta?

Aplikacja mająca pełnić funkcję "zewnętrznego komponentu" w swojej budowie nie będzie się różniła od typowego projektu budowanego w wybranym przez siebie frameworku. Jednak istnieje kilka kluczowych zasad, których należy przestrzegać.

- Skupienie na jednym zadaniu lub grupie bliskich sobie funkcji: Aplikacja-komponent powinna mieć jasno określone zadanie i skoncentrować się na jego realizacji.
- Osobne repozytorium i linia rozwoju: Komponent powinien być przechowywany w oddzielnym repozytorium z własną linią rozwoju oraz serwisem hostingowym. Powinien być również uruchomiony w swoim natywnym środowisku.
- Przekazywanie danych: Komponent powinien umożliwiać otrzymywanie danych od aplikacji nadrzędnej oraz przekazywanie ich w dół i w górę hierarchii komponentów.
- Uniwersalność: Komponent powinien być zaprojektowany w taki sposób, aby mógł być wykorzystywany przez więcej niż jedną aplikację nadrzędną, jeśli zachodzi taka potrzeba.
- Komunikacja wewnętrzna: Komponent powinien zapewniać formy komunikacji zarówno z wewnętrznymi funkcjami sterującymi jego działaniem, jak i informującymi o jego stanie.

Projekt nadrzędny, który korzysta z mikrofrontendów, pełni rolę zarządcy, a do jego zadań należy:

- Dynamiczne i asynchroniczne ładowanie komponentów.
- Sterowanie komunikacją między komponentami.
- Dystrybucja danych i poleceń do komponentów.
- Warstwa kompatybilności dla specyficznych zachowań komponentów dostarczanych przez mikrofrontendy.
- Zarządzanie zasobami, aby uniknąć nadmiernego ładowania danych związanych z wykorzystaniem wielu współdzielonych bibliotek i środowisk pomiędzy komponentami.

- Zapewnienie routingu i nawigacji.
- Organizacja nazw komponentów.
- Rozwiązywanie konfliktów.
- Komponowanie layoutu zbudowanego z różnych komponentów.

Stopień stosowania się do tych zasad zależy od stopnia wykorzystania i ilości zewnętrznych komponentów w projekcie.

Jak stworzyć lub korzystać z mikrofrontendu?

Obecnie tworzenie i wykorzystanie mikrofrontendów jest możliwe dzięki różnym narzędziom, które są odpowiedzialne za przygotowanie i dostarczenie projektu jako komponentu do innej aplikacji, a także umożliwiają łatwe ich wykorzystanie. Istnieje wiele narzędzi i bibliotek, które wspierają tworzenie mikrofrontendów, takich jak SystemJS, Parcel, Rollup czy natywne moduły JavaScript.

Jednak obecnie najbardziej popularnym narzędziem jest najprostszy Iframe oraz plugin Modular Federation, dostępny w bibliotece Webpack 5. Webpack umożliwia automatyczne dzielenie kodu źródłowego na modułowe fragmenty, co pozwala na szybsze ładowanie elementów aplikacji przez przeglądarkę. Konfiguracja narzędzia polega na ustawieniu odpowiednich parametrów w pliku konfiguracyjnym aplikacji. Dzięki temu narzędzie to jest łatwe w zastosowaniu i zachęca do eksperymentowania.

Dzięki wykorzystaniu takich narzędzi, tworzenie i wykorzystywanie mikrofrontendów staje się bardziej efektywne i elastyczne, umożliwiając modułową budowę aplikacji i integrację z innymi systemami.

Schemat budowy aplikacji mikrofrontedowej wraz z prostym przykładem

Założmy, że projektem jest aplikację e-commerce składającą się z trzech głównych komponentów: strony głównej, listy produktów i koszyka zakupowego. Przy implementacji podejścia mikrofrontendowego te elementy będą rozwijane niezależnie od siebie.

Strona główna (Home Microfrontend):

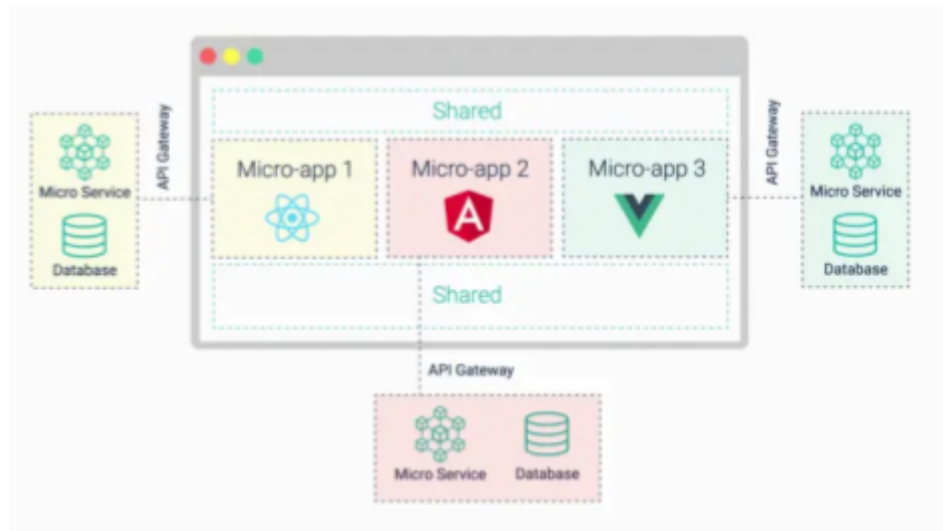
- Odpowiada za prezentację witryny, banerów promocyjnych i informacji ogólnych.
- Może mieć swoje style CSS, pliki JavaScript i zależności.
- Udostępnia interfejs API, który umożliwia innym mikrofrontendom komunikację z nią.

Lista produktów (Product List Microfrontend):

- Odpowiada za wyświetlanie listy produktów, filtrowanie i sortowanie.
- Może mieć swoją logikę biznesową i interakcje z użytkownikiem.
- Udostępnia interfejs API, który umożliwia innym mikrofrontendom komunikację z nią, na przykład pobranie szczegółów produktu.

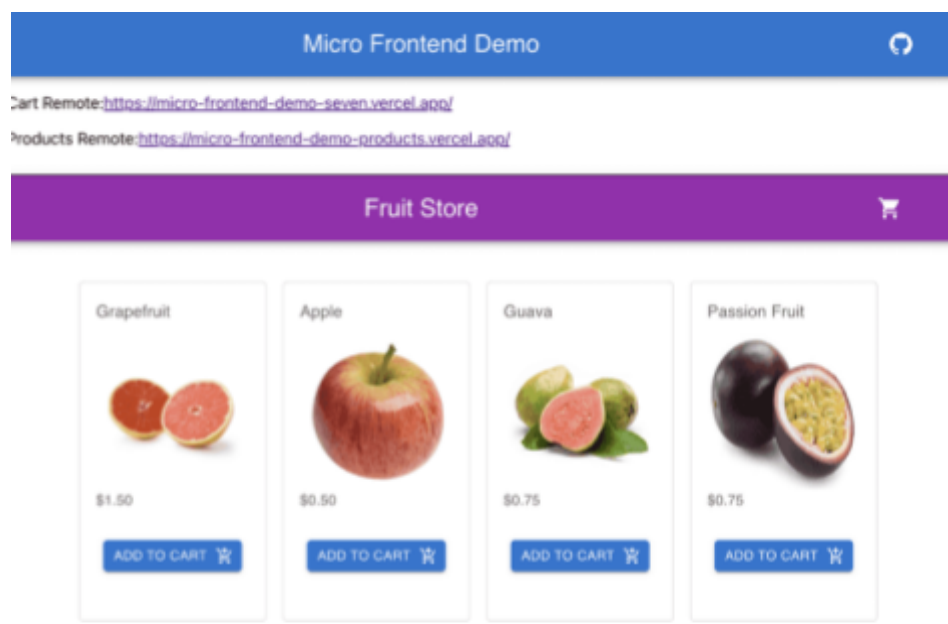
Koszyk zakupowy (Cart Microfrontend):

- Odpowiada za zarządzanie koszykiem zakupowym, dodawanie i usuwanie produktów.
- Może mieć swoje funkcje walidacji i obliczeń.
- Udostępnia interfejs API, który umożliwia innym mikrofrontendom komunikację z nią, na przykład zaktualizowanie liczby produktów w koszyku.



Wizualizacja układu aplikacji zbudowanej na mikrofrontendach.
źródło: CUELOGIC - <https://www.cuelogic.com/blog/micro-frontends-part1>

Każdy z tych elementów **może być zmieniany w czasie rzeczywistym** bez wpływu na pracę całości aplikacji. Zastosowane przy tym technologie są od siebie niezależne, a istotna jest wyłącznie zasada pozostawiania łatwych furtek komunikacyjnych pomiędzy poszczególnymi elementami.



Przykład aplikacji e-commerce zbudowanej przez Oskari Rautiainen
<https://oskari.io/blog/zustand-module-federation/>

Założenia korzystania z micro frontendów oraz zalety takiego podejścia

Podejście mikrofrontendowe stanowi alternatywę dla tradycyjnego sposobu zarządzania bazą kodową projektu i wprowadza modularność, co znacznie ułatwia rozbudowę większych aplikacji.

Kiedy aplikacja może być łatwo podzielona na mniejsze, łatwiejsze do konserwacji i rozbudowy komponenty, mikrofrontendy poprawiają jakość pracy dedykowanego zespołu programistów. Zapewnia to niezależność od innych zespołów lub głównego cyklu wprowadzania zmian do głównej aplikacji. Ponieważ mikrofrontendy są przechowywane w osobnych repozytoriach i mogą być wykorzystywane przez więcej niż jedną aplikację jednocześnie, zespół deweloperski nie jest ograniczony do stosowania tej samej technologii czy narzędzi co projekt główny. Mogą korzystać z bibliotek i narzędzi, które nie są obecne w projekcie głównym, co znacznie upraszcza listę elementów wymaganych w projekcie głównym.

Podobieństwo do menedżerów pakietów, takich jak NPM, polega na dystrybucji narzędzi programistycznych, ale mikrofrontendy są ładowane dynamicznie (w czasie rzeczywistym), co ułatwia ich dostarczanie do klienta. Wszelkie zmiany wprowadzane w linii produkcyjnej mikrofrontendu są natychmiast dostępne dla wszystkich aplikacji korzystających z tych usług.

Wprowadzenie nowej, nie doświadczonej ekipy deweloperów do projektu może być łatwe, szczególnie w przypadku częściowo mikrofrontendowego podejścia. Deweloper, nie musi przejmować się strukturą plików projektu aplikacji głównej i nie ma konieczności długotrwałego wdrażania. Skupia się jedynie na swoim zadaniu, a implementacja nowej funkcjonalności odbywała się automatycznie, nie wymagając od niego szczegółowej wiedzy na temat kluczowych fragmentów aplikacji głównej.

Takie środowisko pracy tworzy interesującą sytuację, w której osoby zewnętrzne lub firmy mogą pracować nad projektem o wysokim stopniu poufności, nie mając bezpośredniego dostępu do kodu źródłowego. To zapewnia większe bezpieczeństwo i kontrolę nad projektem, jednocześnie umożliwiając zewnętrznym zespołom skoncentrowanie się na realizacji swoich zadań bez konieczności wnikania w całość aplikacji.

Dodatkową zaletą jest wydajność, wynikająca z modularnego podejścia, które wymusza ładowanie elementów interfejsu tylko wtedy, gdy są one potrzebne. To może przyspieszyć ładowanie strony, ponieważ nie trzeba jednorazowo pobierać całej aplikacji.

W rezultacie podejście mikrofrontendowe przynosi wiele korzyści, takich jak łatwiejsze zarządzanie, niezależność od innych zespołów, elastyczność w wyborze technologii i narzędzi, dynamiczne ładowanie komponentów, oraz poprawa wydajności aplikacji.

Należy jednak pamiętać że nie jest to podejście idealne, ponieważ wymagają nieco bardziej skomplikowanej konfiguracji niż tradycyjne projekty monolityczne. Co może przysporzyć

problemy nie doświadczonemu zespołowi nadzorującemu projekt. Dlatego wykorzystanie tej architektury najbardziej przydaje się przy dużych projektach w których poziom skomplikowania na starcie jest już dość wysoki.