Wprowadzenie do Aplikacji Internetowych (Webowych)

dr inż. Grzegorz Rogus
rogus@agh.edu.pl

Tematyka wykładów

Od zera do webDevelopera

Plan wykładów

1. **Wprowadzenie do aplikacji webowych** – omówienie stosu technologicznego

2-4 Prezentacja standardów

2. HTML5 + wstęp do CSS

3. CSS + RWD – Zasady projektowania aplikacji webowych

4. Nowoczesny JS
   ECMS7 (2016), TypeScript

Plan wykładów cd

5-6. Frontend:

7 . Backend:

Autentykacja
Web Socket

Wprowadzenie do Aplikacje Webowych

Koncepcja

Architektura

Technologie

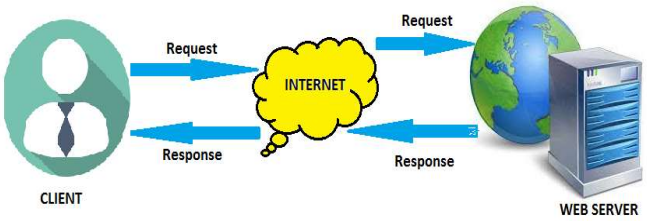
Czym są Aplikacje Webowe

Perspektywa użytkownika końcowego

Aplikacja webowa (web application) to aplikacja uruchamiana w przeglądarce, która przez dostarczony interfejs ma dostarczać użytkownikowi jakąś konkretną usługę. Takie aplikacje w znakomitej większości komunikują się z głównym serwerem, by móc serwować użytkownikowi treści i reagować na jego akcje.

Stosowany Model architektoniczny

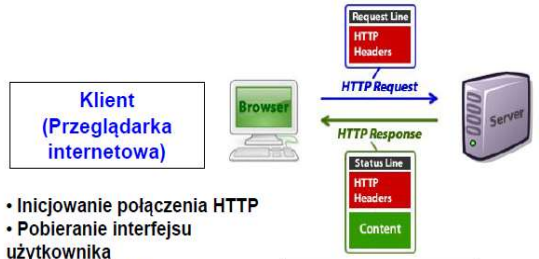
Rzeczywista architektura



W E B TECHNOLOGY

Client - Server Architecture

Podstawowa architektura: klient-serwer



Klient (Przeglądarka internetowa)

- Inicjowanie połączenia HTTP
- Pobieranie interfejsu użytkownika
- Prezentacja interfejsu użytkownika
- Interakcja z użytkownikiem
- Buforowanie odpowiedzi
- Kryptografia

Protokół HTTP

- Oparty na TCP
- Komendy tekstowe
- Transmisja 8-bitowa
- Bezstanowy, bezsesyjny

Serwer aplikacji oraz danych

- Obsługa żądań HTTP
- Rejestracja żądań
- Uwierzytelnianie i kontrola dostępu
- Kryptografia
- Wybór wersji językowej wysyłanych plików

Klient HTTP

Przeglądarka – http client

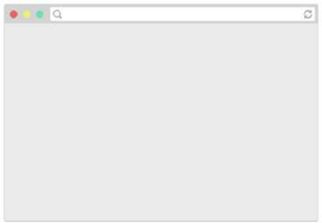
Przeglądarka to aplikacja której zadaniem jest wyświetlenie zawartości stron internetowych lub w przypadku aplikacji webowej warstwy prezentacyjnej

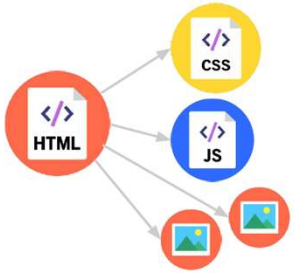
Odpowiada za:

- Wysyłanie żądań pobrania dokumentów
- Wizualizacje pobranych dokumentów
- Obsługę interakcji z użytkownikiem końcowym

Strony internetowe

Strony/widok są napisane w języku znaczników zwanym HTML,więc przeglądarki wyświetlają stronę internetową, czytając i interpretując jej kod HTML.



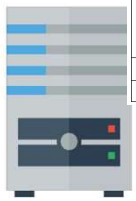


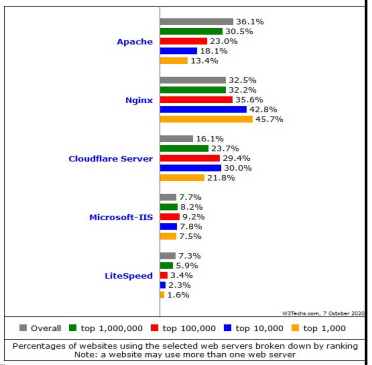
Plik HTML może zawierać linki do innych zasobów, takich jak obrazy, filmy, a także pliki **JavaScript** i **CSS** (arkusz stylów), które przeglądarka również ładuje

Serwer HTTP

Web Server

Serwer WWW to program działający na komputerze, który dostarcza strony internetowe w odpowiedzi na żądania. Przechowuje lub generuje zwróconą stronę internetową.





Serwer HTTP

Serwer HTTP - serwer WWW - program nieprzerwanie pracujący, obsługujący repozytorium dokumentów (np. HTML), które udostępnia sieciowym klientom HTTP.

Do zadań serwera HTTP należy:

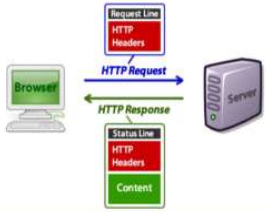
- obsługa żądań HTTP i ich rejestracja w plikach dziennika (log files),
- uwierzytelnianie i kontrola dostępu użytkowników końcowych za pomocą nazwy i hasła,
- kryptograficzne szyfrowanie komunikacji sieciowej z klientem http,
- automatyczny wybór odpowiedniej wersji językowej dokumentu.

Protokół Http

Hyper Text Transfer Protocol (HTTP/1.1)
[Protokół przesyłania dokumentów Hiper-Tekstowych]

- **Oparty na TCP,**
- **Komendy tekstowe,**
- **Transmisja 8-bitowa,**
- **Bezstanowy i bezsesyjny.**

RFC 2616
(czerwiec 1999)
RFC (Request for Comments)



Bezstanowość" *(ang. stateless)*, co oznacza że nigdzie nie istnieje zapis stanu poprzednio wykonanych operacji, a kolejne transakcje są wykonywane niezależnie.

Najważniejsze metody protokołu HTTP:

- **HEAD** - wysyła żądanie przesłania nagłówka zawierającego meta-dane (informację), bez przesyłania samego zasobu.
- **GET** - wysyła żądanie pobrania konkretnego zasobu URI (np. strony internetowej napisanej w języku HTML).
- **POST** - wysyła żądanie do serwera akceptacji zasobu dołączonego do żądania.

pozostałe: **PUT, DELETE, TRACE, OPTIONS, CONNECT.**

Zakres kodów	Znaczenie
100 - 199	Informacyjne.
200 - 299	Żądanie (od klienta) powiodło się
300 - 399	Żądanie klienta zostało przekazane, wymagane są dalsze działania.
400 - 499	Żądanie klienta nie powiodło się.
500 - 599	Błąd serwera.

Metody w protokole Http

HTTP Methods (Verbs)

- GET – odczyt danych (w formularzach metoda wysyłki danych do URL – dane zapisane w adresie)
- HEAD – odczyt danych na temat adresu URL
- PUT – zapis do zasobu pod URL
- POST – wysłanie danych pod adresURL oraz otrzymanie informacji zwrotnej
- DELETE - Delete a URL

GET oraz POST (formularze) są najczęściej uzywane.

REST APIs używa GET, PUT, POST, and DELETE

Nagłówek Http

Jak mówi klient (przeglądarka)?

http request

Linia startowa	Metoda sciezka http/wersja	
Nagłówek	Nazwa1: wartość1 Nazwa2: wartość2	
Ciało zapytania	dane, które wysyłamy do serwera (opcjonalnie)	

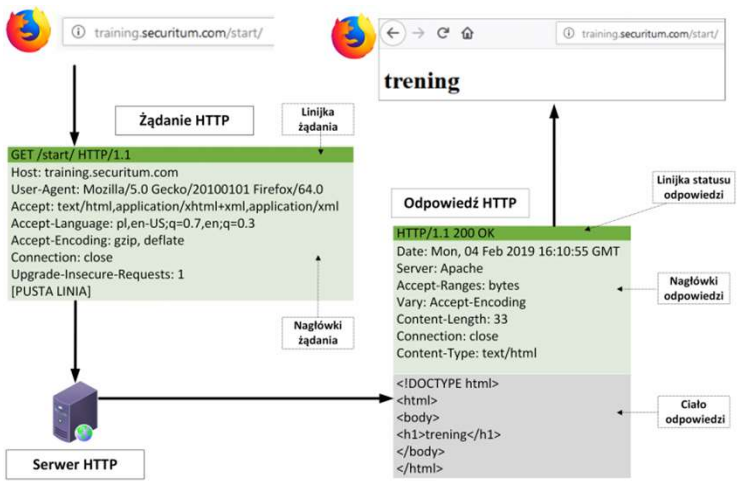
Jak mówi serwer?

http response

http/wersja	kod statusu
Nazwa1: wartość1 Nazwa 2: wartość2	
Zawartość pliku (opcjonalnie)	

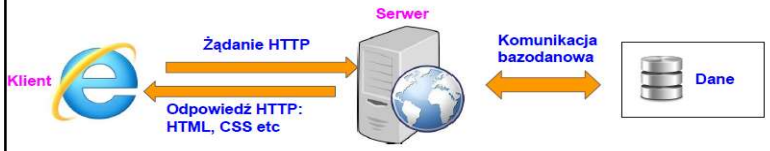
Siła drzemie... w nagłówkach
w standardzie HTTP 1.1 otrzymujemy:
- 31 nagłówków zapytania
- 34 nagłówków odpowiedzi

Nagłówek http - przykład



Podstawowa architektura: klient-serwer

Klasyczny model komunikacji synchronicznej HTTP:



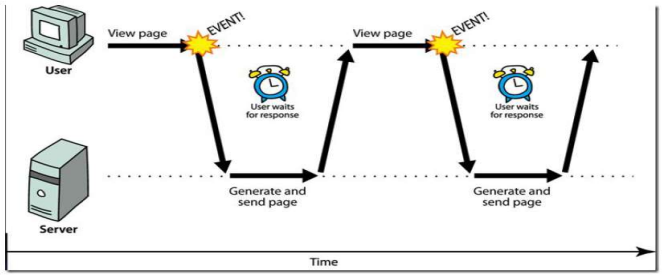
Jest to model synchronicznej komunikacji HTTP, gdzie klient wysyła żądanie, serwer je odbiera następnie przetwarza (np. komunikuje się z bazą danych) i na końcu generuje odpowiedź.

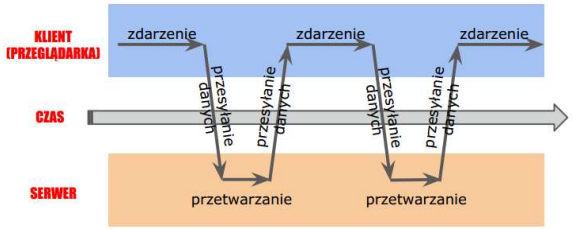
W sytuacji takiej klient musi czekać z kolejnym żądaniem do momentu kiedy nie dostanie odpowiedzi od serwera.

W modelu synchronicznym mamy bardzo mały poziom aktywności oraz interaktywności strony, strona musi być przeładowana (pobrana) po każdym żądaniu klienta, jeżeli strony są złożone to proces ten jest długi.

Podstawowa architektura: klient-serwer

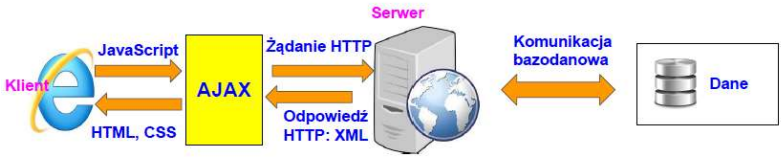
Klasyczny model komunikacji synchronicznej HTTP:





Podstawowa architektura: klient-serwer

Model komunikacji asynchronicznej HTTP:

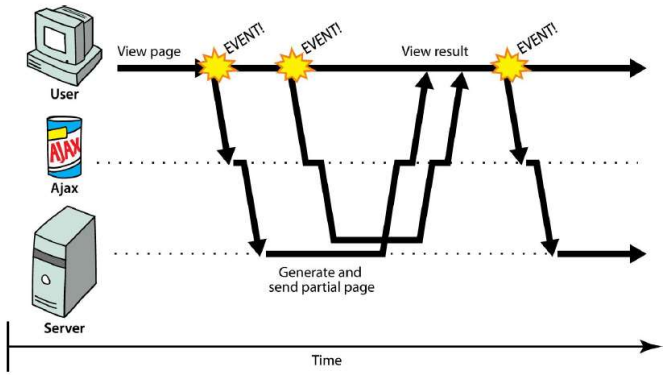


Jest to model asynchronicznej komunikacji HTTP, gdzie klient (przeglądarka) nie czeka na przyjście odpowiedzi na żądanie serwera, a wykonuje dalsze żądania. Pośredniczy w tym wbudowany w przeglądarkę mechanizm silnika AJAX.

W takim modelu nie ma konieczności przeładowania strony przy każdej operacji klienta, wystarczy, że zostaną odczytane brakujące dane, a dzięki odpowiednim narzędziom zmodyfikowana zostanie zawartość strony.

Podstawowa architektura: klient-serwer

Model komunikacji asynchronicznej HTTP:



Typy aplikacji webowych

4 główne typy apliakcji :

1. O statycznej treści (to jest strona internetowa a nie web application!!!!**)**

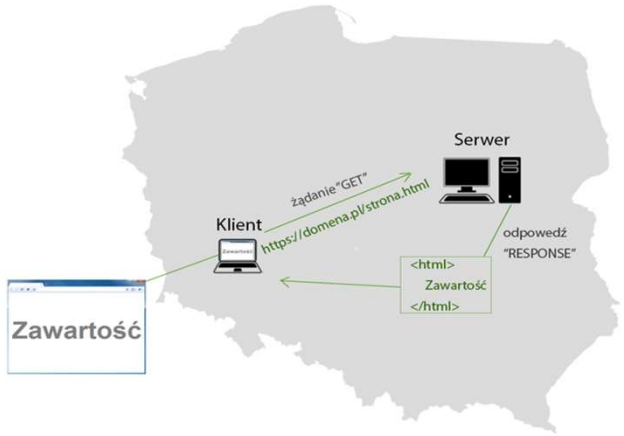
2. Server-side rendering:
Serwer wysyła nową stronę HTML dla każdej unikalnej ścieżki (może to strony internetowe, ale często jest to aplikacja internetowa)

3. Single-page application

4. Aplikacje progresywne

Od strony WWW do aplikacji internetowej

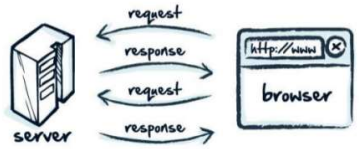
Jak działają strony internetowe?



Na początku była strona statyczna

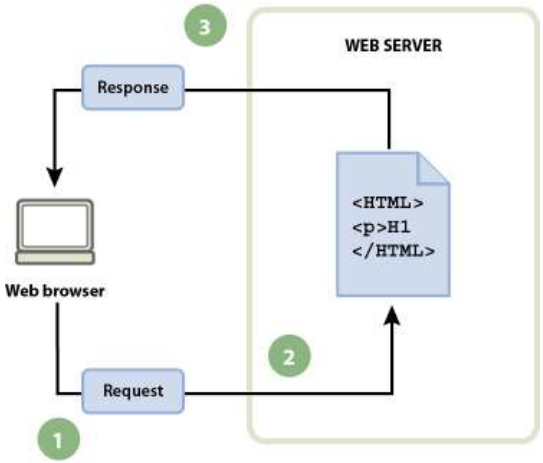
Tradycyjna strona internetowa

- Korzysta z HTML5, CSS3, JavaScript
- Bazuje na klasycznej architekturze klient-serwer
- Zawiera wiele stron



Strona WWW to dokument HTML udostępniany w Internecie lub lokalnej sieci komputerowej przez tzw. serwer WWW.

Proces dostarczenia statycznej strony - podsumowanie



Strony statyczne

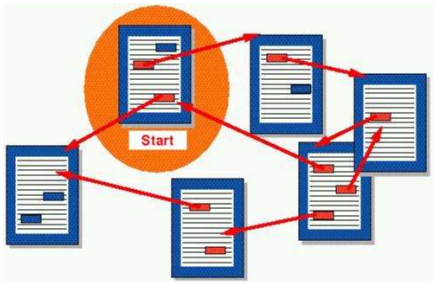
HTML + CSS produces

Opisuje kontekst i strukturę strony

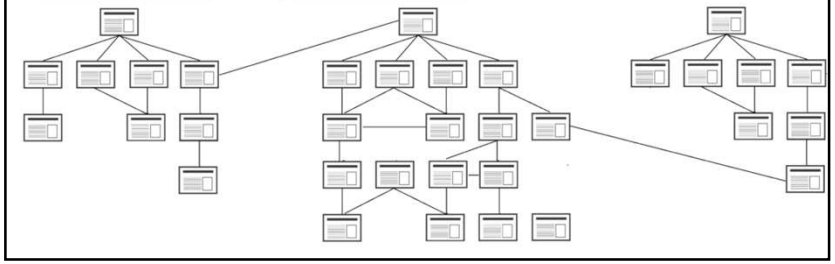
Definiuje wygląd strony

Strona internetowa, która wygląda zawsze tak samo

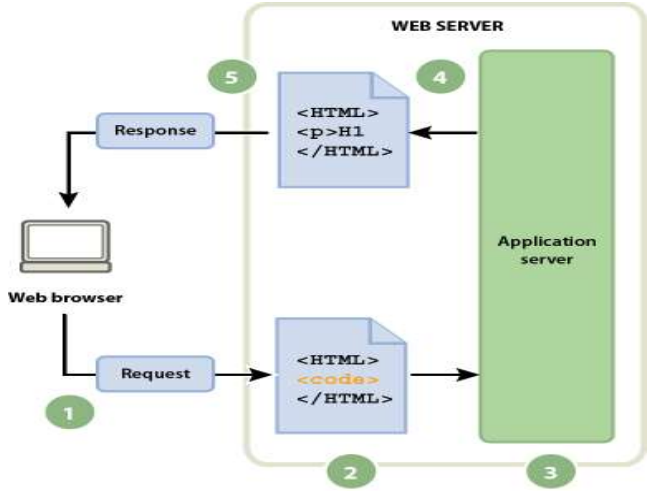
Strona internetowa WWW



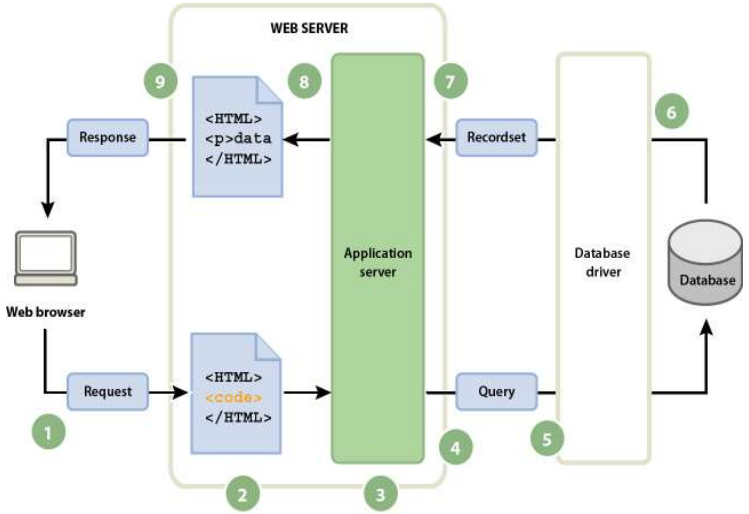
Zbiór stron powiązanych linkami (hiperłączami)



Proces dostarczenia strony dynamicznej – w praktyce

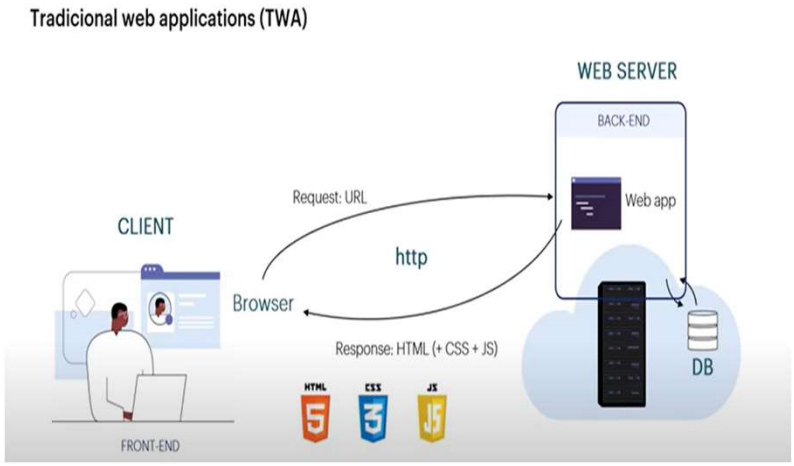


Generacja strony dynamicznej wersja szczegółowa



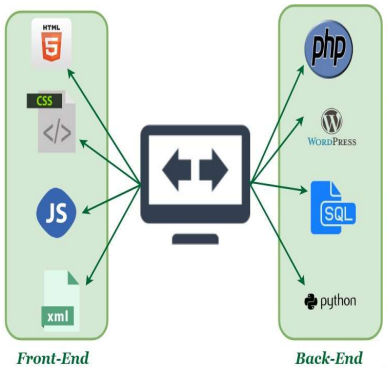
29

Tradycyjna aplikacja webowa



30

Technologie Web

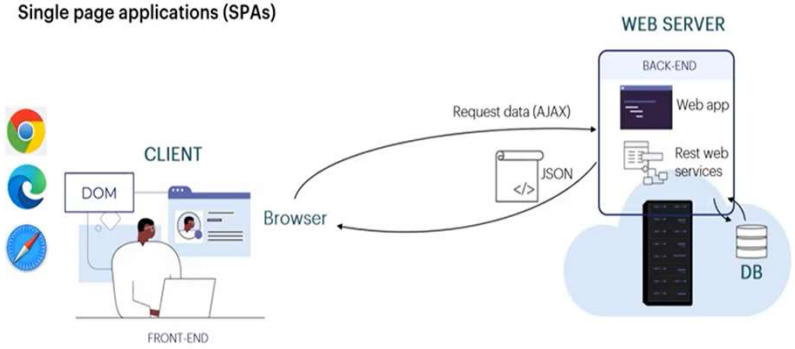


FRONT-END - pojęciowo odnosi się w technologiach internetowych do kodu wykonywanego po stronie użytkownika. W ogólności do tej kategorii można zaliczyć HTML, CSS oraz JavaScript.

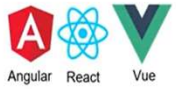
BACK-END - pojęciowo odnosi się w technologiach internetowych do kodu wykonywanego po stronie serwera. W ogólności do tej kategorii można zaliczyć PHP, Perl, CGI, Ruby, Java, C#, itp.

Nowoczesne aplikacje webowe

Single page applications (SPAs)

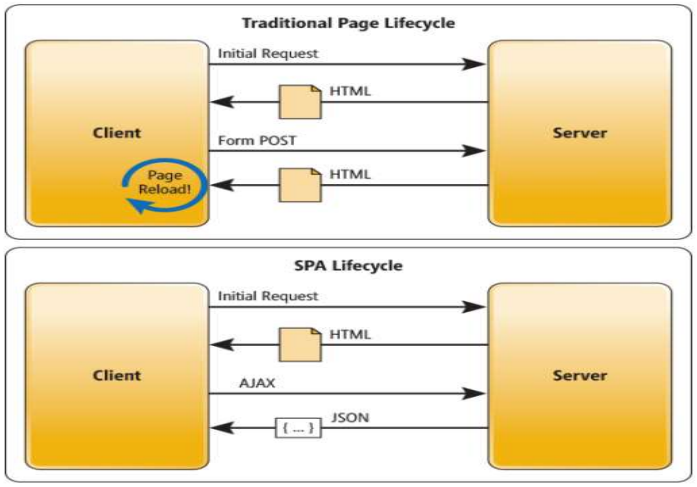


SINGLE PAGE APP (SPA)



Aplikacja (SPA) to aplikacja internetowa, która wchodzi w interakcję z użytkownikiem, dynamicznie przepisując bieżącą stronę internetową nowymi danymi z serwera internetowego, zamiast domyślnej metody wczytywania przez przeglądarkę całych nowych stron.

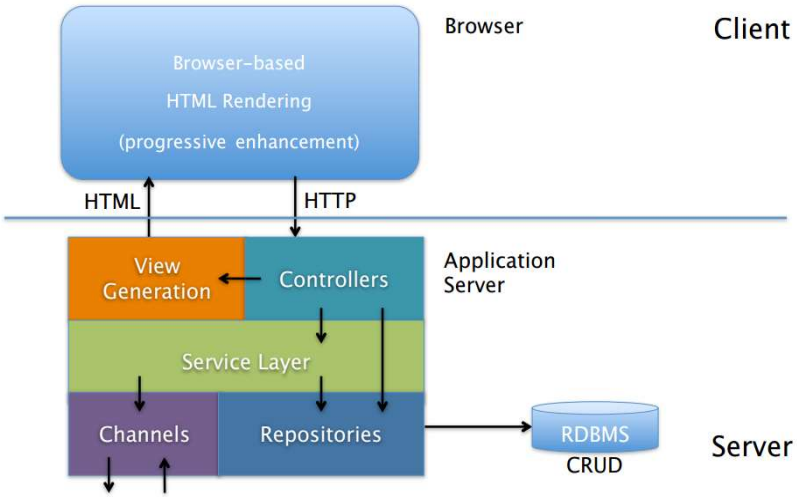
SPA in action



Single Page Applications (SPA)

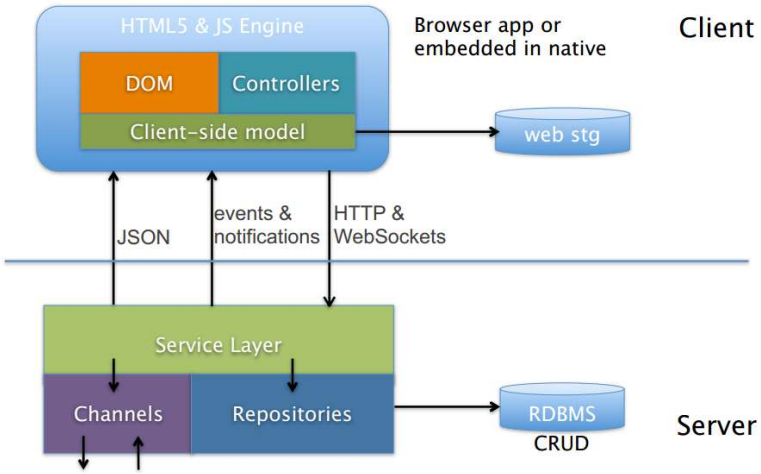
- Strona startowa aplikacji jest jedyną stroną pobieraną w całości z serwera (potem interakcje Ajax, WebSocket)
- Strona aplikacji nie przeładowuje się w czasie pracy z nią
- Nie następuje nawigacja do innych stron
- Zmienia się stan (i wygląd) strony w przeglądarce
- User Experience (UX) podobny do aplikacji desktopowych
- Technologie: HTML5, CSS, JavaScript, Ajax, WebSocket
- Frameworki: Angular, ReactJS, Vue, …

Od server-side app w kierunku inteligentnych klientów i serwisów



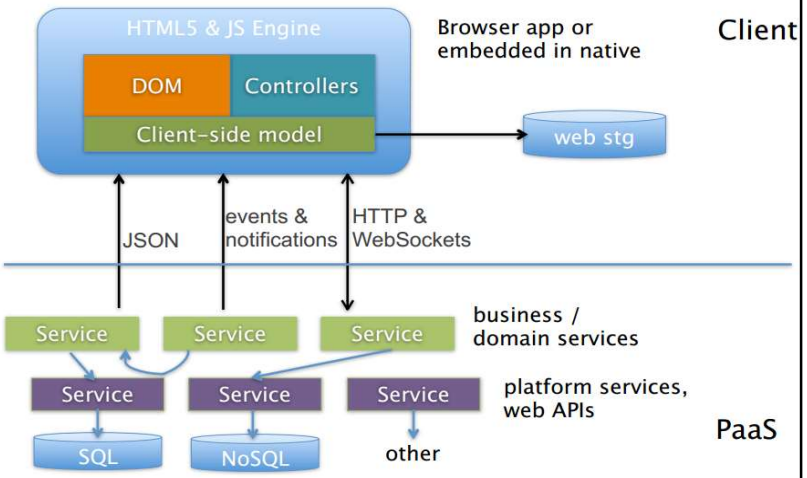
35

Od server-side app w kierunku inteligentnych klientów i serwisów

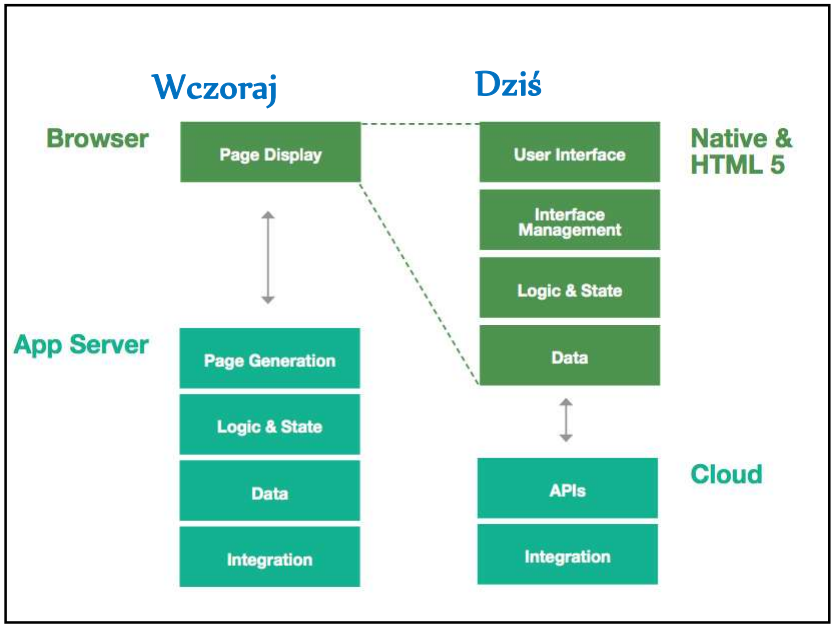


36

Od server-side app w kierunku inteligentnych klientów i serwisów



37



38

Zmiany w tworzeniu aplikacji webowych

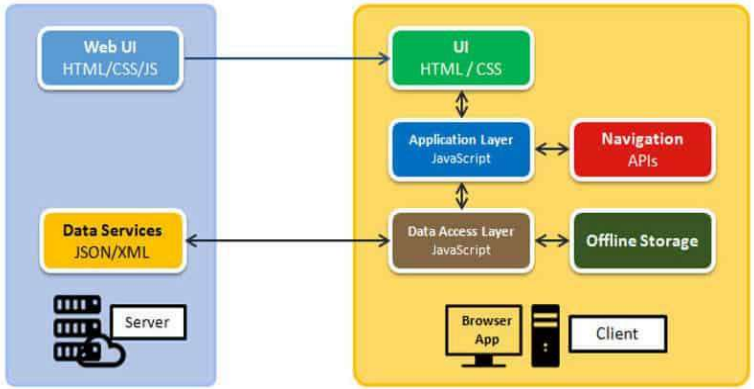
Warunki:

- nowoczesne przeglądarki,
- rozwój języka JavaScript,
- większy nacisk położony na wygodę użycia.

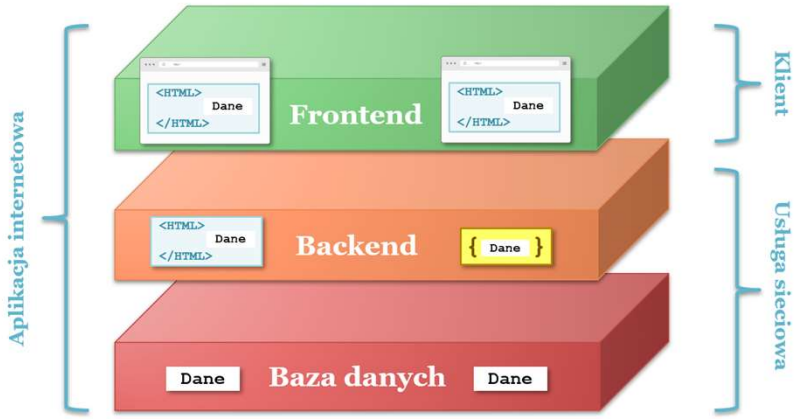
Zmiany w aplikacjach internetowych:

- z punktu widzenia użytkownika działają jak aplikacje desktopowe,
- szybki i dużo bardziej interaktywny interfejs,
- potrafią działać nawet offline,
- działają na wielu platformach (RWD).

Architektura SPA



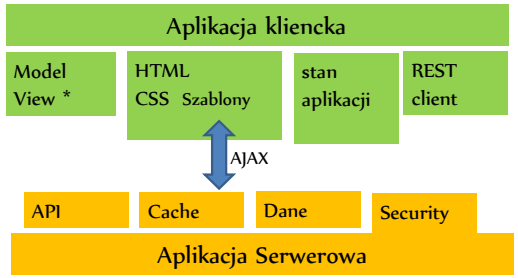
Architektura aplikacji internetowej



41

Aplikacja SPA (Single Page Aplication)

- wysoka wydajność
- user experience
- współpraca frontend – backend
- możliwość zmiany backendu niezależnie od frontendu (i vice versa)
- ten sam backend dla wielu aplikacji
- SEO
- Kompatybilność miedzy przeglądarkami

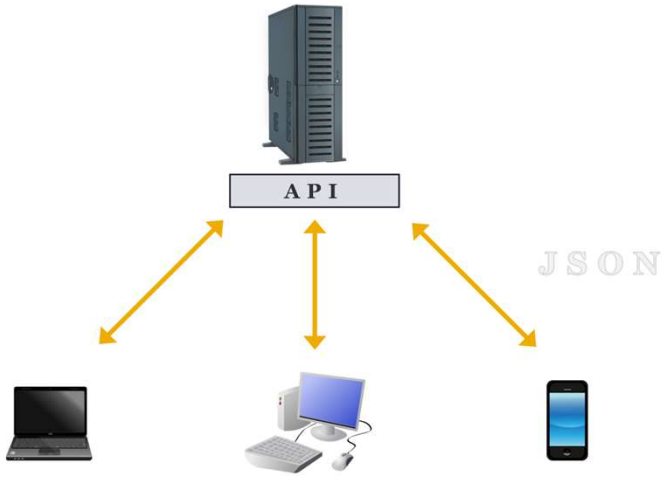


42

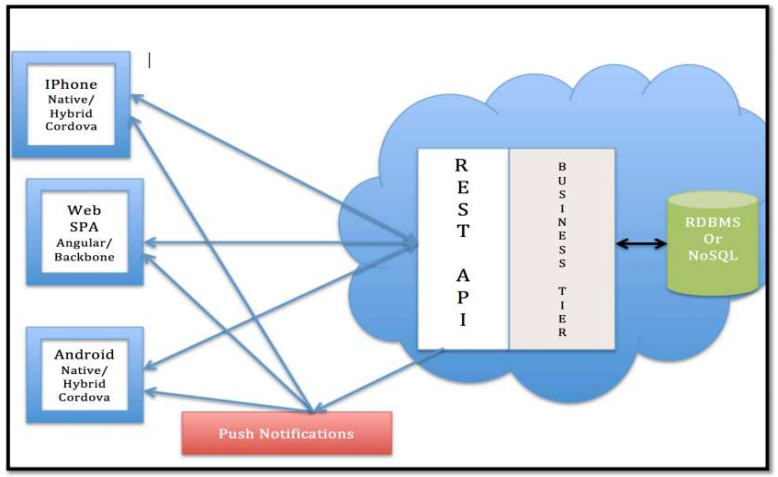
Nowoczesna architektura aplikacji Webowych - podsumowanie

- jeden backend wspólny dla wielu frontendów/klientów
  - Aplikacja HTML-owa
  - Natywna aplikacja mobilna
  - Inna aplikacja korzystająca z API
- backend nie generuje żadnego HTML-a (tylko przy inicjalizacji)
- odpowiada jedynie na żądania HTTP
- na wyjściu generuje XML lub JSON
- frontend HTML-owy napisany w JavaScript
- komunikacja z backendem z wykorzystaniem AJAX
- komunikacja fronten – backend z wykorzystaniem RestAPI (coraz częściej GraphQL)

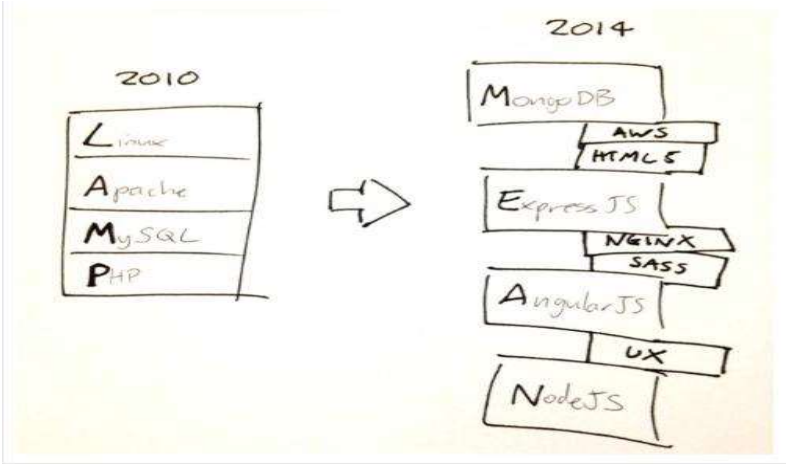
Nowoczesna architektura Web



Nowoczesna architektura Web



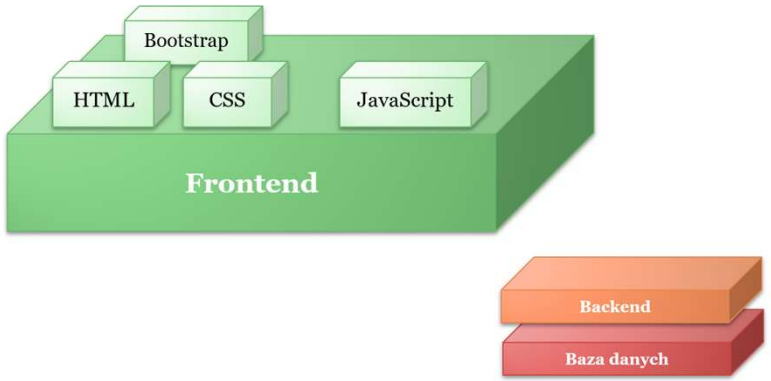
Stos technologiczny – kiedyś i dziś



TECHNOLOGIE FRONT-END

47

Technologie frontendowe

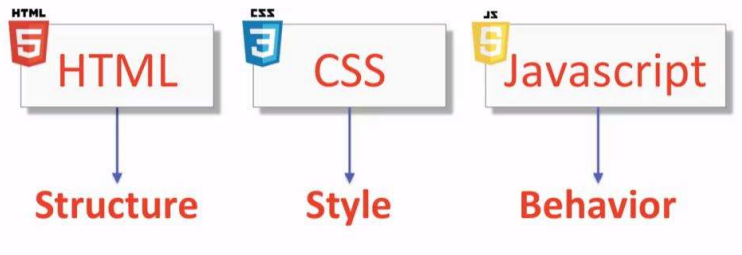


48

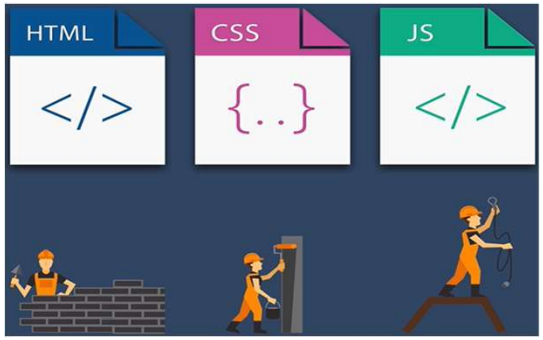
Podstawy FRONT-END –czyli MUST HAVE

Podstawowe umiejętności:

- **HTML 5**;
- **Kaskadowe Arkusze Styli CSS**;
- **Język JavaScript**;



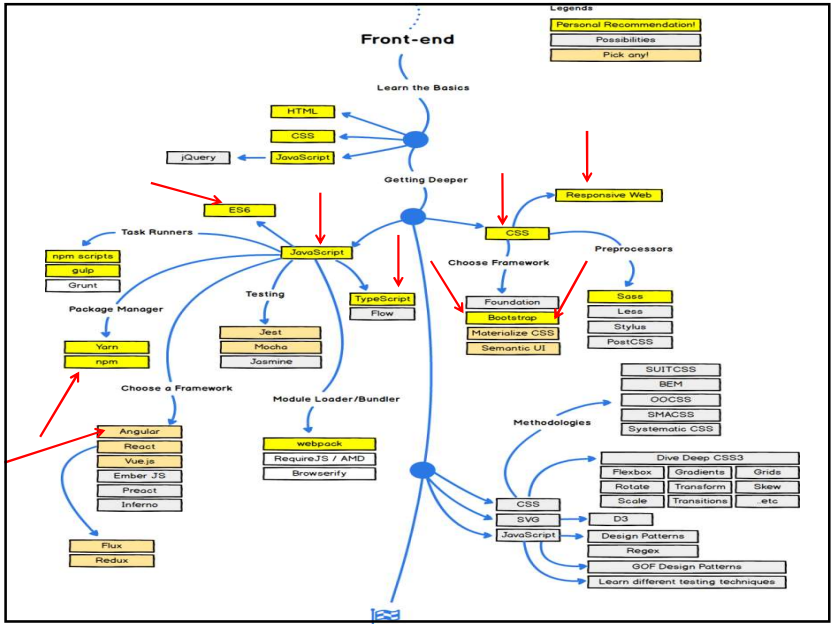
Brygada budowlana



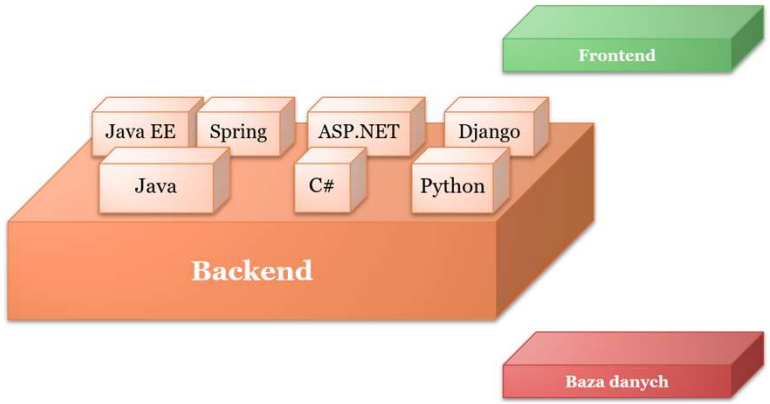
TECHNOLOGIE FRONT-END'OWE

Obejmują:

- Język znaczników HTML
- Kaskadowe arkusze styli CSS
  - Preprocesowy CSS (Less, Saas, itp.)
  - Biblioteki (Bootstrap, Materialize, Material Design)
  - RWD (Responsive Web Design): media queries
- JavaScript
  - MV* frameworki: Angular, ReactJS, VueJS ….
  - Testy jednostkowe: Karma, Jasmine, Jest
  - Biblioteki: Vanilla JS, Babel JS
- Narzędzia
  - Npn, yard (managery pakietów)
  - Gulp, Grunt (automatyzacja)
  - WebPackage (budowa pakietów)

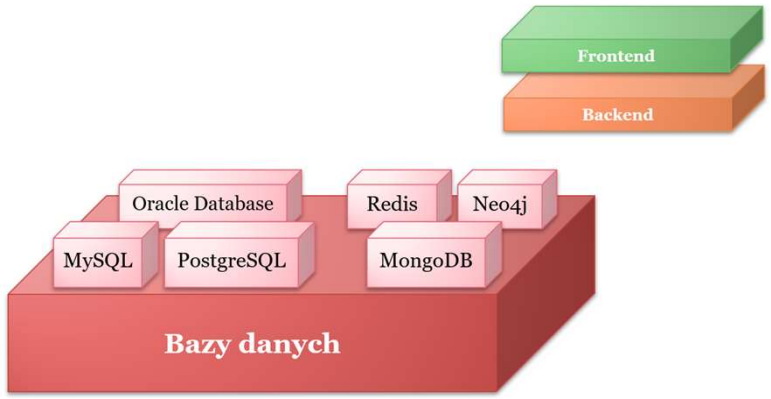


Technologie backendowe



53

Bazy danych



54