

## Spis treści

Streszczenie .....	6
Abstract .....	6
Wprowadzenie .....	7
Cel i zakres pracy .....	8
1. Badane technologie .....	9
HTTP 1.0 .....	9
HTTP 1.1 .....	10
HTTP 2.0 .....	12
SPDY .....	13
Wydajność części front-endu i back-endu .....	13
Wydajność sieci bezprzewodowych .....	16
Mobilne witryny Internetowe .....	19
2. Opis narzędzia Snail Project .....	23
Agile .....	23
Pierwsza iteracja Snail Project .....	23
HTML .....	24
CSS .....	24
PHP .....	24
JavaScript .....	24
jQuery .....	24
Ajax .....	24
Twitter Bootstrap .....	24
Git .....	24
PageSpeed Insight .....	25
GTMetrix .....	26
Modernizr .....	26
Analiza struktury szablonu HTML .....	27
3. Dlaczego warto wybrać narzędzie Snail Project .....	29
4. Narzędzie Snail Project w porównaniu z innymi .....	30
Test porównujący narzędzia .....	31
PageSpeed Insights .....	31
GTmetrix .....	32
Load Impact .....	32

WebPagetest .....	32
WebSiteOptimization .....	33
Neustar .....	33
Pingdom .....	33
Snail Project .....	33
Podsumowanie wykonanych testów.....	33
Wnioski .....	35
<b>5. Badania 10 najpopularniejszych stron Internetowych w Polsce .....</b>	<b>36</b>
Google.pl .....	36
Facebook.com .....	37
Allegro.pl.....	39
Google.com .....	42
Youtube.com .....	43
Onet.pl i WP.pl.....	45
Gazeta.pl.....	48
Wikipedia.pl.....	50
Olx.pl.....	52
Wnioski .....	55
<b>6. Reguły wydajności .....</b>	<b>57</b>
Unikanie zbędnych przekierowań .....	57
Włączenie kompresji danych.....	58
Wykorzystanie pamięci podręcznej przeglądarki.....	58
Skrócenie czasu odpowiedzi serwera .....	59
Minimalizowanie plików CSS .....	60
Minimalizowanie plików HTML .....	60
Minimalizowanie plików JavaScript.....	60
Eliminowanie blokujących skryptów .....	60
Optymalizacja obrazów .....	61
Priorytetyzacja widocznej treści.....	61
Optymalizacja fontów .....	62
WebP .....	63
CSS Sprites.....	65
CDN .....	65
Strona WWW czy aplikacja ? .....	67
Doświadczenia użytkownika w płyn na wydajność .....	68

Wydajność przeglądarek internetowych .....	69
7. Badanie czy poprawa błędów ma przełożenie na lepszą wydajność serwisu .....	72
8. Czy warto trzymać się wytycznych World Wide Web Consortium ? .....	77
9. Czy wprowadzenie SPDY oraz HTTP 2.0 przyniesie zakładany wzrost wydajności ?... ...	78
10. Podsumowanie .....	80
Rezultat pracy.....	81
Propozycje kontynuacji i wizja rozwoju .....	81
Literatura .....	83

## **Streszczenie**

Praca ta prezentuje kondycję najpopularniejszych serwisów webowych w Polsce. Zostały one przebadane pod kątem optymalizacji grafiki, minimalizacji plików, które są pobierane przez użytkowników, użycie kompresji danych itp. Wpływ na jakość współczesnych serwisów internetowych mają nie tylko coraz większe wymagania użytkowników, lecz także ewolucja całej sieci Internet, oraz sieci komórkowe czy rozwój protokołów sieciowych (HTTP oraz SPDY).

Oprócz tekstu niniejszej pracy powstał projekt oraz implementacja serwisu do badań i oceny jakości istniejących w sieci systemów webowych o nazwie "Snail Project". Następnie przy jego pomocy przeprowadzono badania istniejących w Internecie serwisów WWW oraz oceniono ich jakość biorąc pod uwagę różne kryteria.

Dodatkowo zostały wykonane dwie strony internetowe, na przykładzie, których dodatkowo analizowane są wyniki przeprowadzonych badań, wraz z ingerencją w kod źródłowy obu stron.

## **Abstract**

Thesis presents condition of most popular web services in Poland. They were checked for graphic optimisation, files minimisation which are download by user, data compression usage etc. Influence on quality of nowaday web services, not only have increasing users needs, but also evolution of whole Internet and cell phones network whether network protocols (HTTP or SDPY).

Besides of this thesis was created project and implementation of web service named "Snail project" for testing existing web pages and rate their quality. Next step was research of existing on the Internet web pages and rate their quality by adopted for different criteria.

Additionally were made two websites on which example additionally was showed research results with interference into both web pages source code.

## **Wprowadzenie**

Tempo rozwoju Internetu rośnie wraz z ilością jego użytkowników, a jego obecny stan już dawno przerósł oczekiwania swoich twórców. Celem "pierwszego Internetu", którego przodkiem były pierwsze węzły sieci ARPANET, była pomoc nad koordynowaniem badań naukowych dla wojska. Był to eksperiment, którego zadaniem było zbadanie możliwości budowania sieci komputerowych bez specjalnej jednostki centralnej. Taka sieć miałyby funkcjonować bez problemów nawet w przypadku uszkodzenia części jej struktury. Wszystkie rozwiązania, które istniały przed wprowadzeniem tego eksperimentu w życie, opierały się na zarządzaniu całą siecią przez jeden główny komputer, którego awaria pozbawiała możliwości pracy wszystkich jej użytkowników. Współczesna postać Internetu to szereg stron Internetowych, aplikacji webowych, kilometry światłowodów i miliardy użytkowników na całym świecie.

Prócz popularyzacji szerokopasmowego dostępu do Internetu wraz z nim dorasta inny sposób łączenia się z siecią – xG, czyli sieci komórkowe. Każda z generacji 2G, 3G czy 4G (oraz przyszłe) dotyczą urządzeń mobilnych, z którymi obecni użytkownicy sieci się nie rozstają. Jest to szybki dostęp do informacji zawartych w aplikacjach webowych wprost z urządzenia mieszczącego się w kieszeni spodni.

Smartfon, tablet, laptop czy komputer stacjonarny, niezależnie od sposobu dostępu do sieci łączy jeden problem obecnego Internetu – wydajność aplikacji webowych. W erze Internetu LTE czy coraz popularniejszego wykorzystywania światłowodów, nawet najmniejsza strona internetowa nie będzie działać w sposób szybki i bezproblemowy, jeśli się nie zadba o jej kod, ustawienia odpowiednich parametrów, wartości i konfiguracji serwera, na którym będzie ona uruchomiona.

Profesjonalni programiści lub amatorzy tworzenia aplikacji webowych muszą wykonać szereg czynności optymalizacyjnych, aby ich serwis działał szybko i wydajnie. Gdyby pozostawić istnienie takiej strony internetowej samej sobie, użytkownicy mogliby się poczuć niekomfortowo, gdyby czas oczekiwania na załadowanie się tak cennych informacji wynosił kilka minut, lub też zawartość strony zawierała błędy np. z wyświetlaniem obrazów. Ponieważ wydajność aplikacji jest najbardziej podstawowym czynnikiem, jaki wpływa na jej sukces, należy o nią zadbać jeszcze przed tworzeniem nowego rozwiązania lub przy pomocy odpowiednich narzędzi przetestować i zoptymalizować bieżące.

Duże firmy takie jak Adobe, Apple, Google, Microsoft, Yahoo! i wiele innych wspiera rozwój sieci oraz badania nad nowymi technologiami np. HTML 5 czy CSS 3 mającymi na celu poprawę stabilności i szybkości działania rozwiązań, czy narzędzi internetowych. Jednak przodownikiem w tych badaniach jest World Wide Web Consortium - (w skrócie W3C), czyli organizacja, zajmująca się ustanawianiem standardów pisania i przesyłu stron WWW [1].

## **Cel i zakres pracy**

Główym celem tej pracy jest projekt oraz implementacja serwisu do badań i oceny jakości istniejących w sieci systemów webowych. Następnie przy pomocy wykonanego narzędzia wykonanie testów istniejących w Internecie serwisów WWW oraz ich ocena pod kątem wydajności.

Dodatkowym elementem pracy jest wykonanie dwóch osobnych stron internetowych, które również zostaną poddane analizie, a na ich przykładzie zostanie zaprezentowany sens i cel optymalizowania własnych serwisów webowych.

Projekt powinien przede wszystkim testować jakość pobierania stron internetowych. Oceniane będą czynniki, które są związane z konstrukcją strony i realny wpływ na czas pobierania strony np. kompresja danych, optymalizacja obrazów, skryptów itd. implementacja powinna być ogólnodostępna tak, aby można było wykonać test dowolnego serwisu webowego.

## 1. Badane technologie

HTTP jest obecnie najpopularniejszym i najbardziej rozpowszechnionym protokołem służącym do komunikacji klientów z serwerami. Jego początki nie zakładały tak wszechstronnego użycia jak ma to miejsce obecnie. Niemal wszystkie aplikacje i urządzenia korzystające z sieci Internet korzystają z tego rozwiązania.

Pierwszą wersję protokołu HTTP stworzył Tim Berners-Lee, i zakładała ona jak najprostszą komunikację. Został zaprojektowany do przesyłania dokumentów hipertekstowych i oparty na protokołach TCP/IP i oznaczony numerem wersji 0.9. Funkcjonalności, jakie zaimplementowano w protokole przedstawiają się następująco:

- Żądanie klienta to prosty ciąg znaków ASCII
- Żądanie klienta zakończone było znakiem nowego wiersza (znak CRLF)
- Odpowiedź z serwera była prostym strumieniem tekstowym
- Odpowiedź była sformatowana w języku HyperText Markup Language (HTML)
- Po zakończeniu przesyłania dokumentu połączenie było zamykane

W popularnych serwerach takich jak Apache [2] czy Nginx [3] wciąż dostępna jest obsługa protokołu HTML 0.9. Jego działanie jest traktowane raczej jako ciekawostka i wykorzystanie do niekonwencjonalnych zadań.

```
$> telnet website.org 80  
  
Connected to xxx.xxx.xxx.xxx  
  
GET /rfc/rfc1945.txt HTTP/1.0  
  
(plain-text response)  
(connection closed)
```

Powyższe żądanie składa się z metody GET oraz ścieżki do żądanego dokumentu, natomiast odpowiedź jest zwykłym dokumentem HTML. Najprostsze połączenie jakie można stworzyć.

### HTTP 1.0

Największy rozwój specyfikacji protokołu HTML nastąpił w latach 1991 – 1995. W tym czasie powstawało takie oprogramowanie jak przeglądarki Internetowe, narzędzia do komunikacji. To był także czas rozwoju infrastruktury całego Internetu. Pożądanie nowych funkcjonalności rozwijającego się Internetu wymagało czegoś więcej niż HTTP 0.9, który nie zadowalał programistów tworzących aplikacje inne niż klient-serwer = żądanie-odpowiedź.

W Maju 1996 roku zbiór „dobrych praktyk i wzorców” został opublikowany przez „HTTP Working Group” (HTTP-WG) pod nazwą RFC 1945 i była to pierwsza specyfikacja protokołu HTTP 1.0. Specyfikacja ta była jednak jedynie informacyjna, a sam protokół do chwili obecnej nie doczekał się oficjalnej, formalnej dokumentacji i nie jest także oficjalnym standardem, który obowiązuje w internecie.

```
$> telnet website.org 80
```

Connected to xxx.xxx.xxx.xxx

GET /rfc/rfc1945.txt HTTP/1.0 ①

User-Agent: CERN-LineMode/2.15 libwww/2.17b3

Accept: \*/\*

HTTP/1.0 200 OK ②

Content-Type: text/plain

Content-Length: 137582

Expires: Thu, 01 Dec 1997 16:00:00 GMT

Last-Modified: Wed, 1 May 1996 12:45:26 GMT

Server: Apache 0.84

(plain-text response)

(connection closed)

① Numer wersji protokołu HTTP oraz następujący po nim nagłówek

② Status odpowiedzi oraz następujący po nim nagłówek

- Przykładowe klasy odpowiedzi Status-Code [4]:  
"200" ; OK  
"201" ; Created  
"202" ; Accepted  
"204" ; No Content  
"302" ; Found  
"403" ; Forbidden  
"404" ; Not Found

W przeciwieństwie do poprzedniej wersji nastąpiło kilka zmian:

- Żądanie może się składać z kilku pól nagłówka oddzielonych znakami nowego wiersza
- Obiekt z odpowiedzią jest poprzedzony wierszem ze statusem odpowiedzi
- Obiekt zawierający odpowiedź zawiera własny zestaw pól nagłówka oddzielonych znakami nowego wiersza
- Po każdym żądaniu połączenie między klientem a serwerem jest zamykane

Nagłówki żądań oraz odpowiedzi są zapisane w kodzie ASCII, lecz obiekt odpowiedzi może być każdego innego typu np. plikiem tekstowym, plikiem HTML, obrazem itp. Specyfikacja RFC opisuje nie tylko negocjację typu mediów ale także kodowanie treści, autoryzację, autoryzację serwera proxy, obsługę zestawów znaków, formaty daty, typ komunikatów wieloczesciowych, zapis do pamięci podręcznej itd.

## HTTP 1.1

Praca nad dokumentowaniem protokołu w wersji 1.0 trwała około pięciu lat (do 1999 roku). Równolegle trwały prace nad przekształceniem technologii w nowszą wersję oznaczoną jako HTTP 1.1 i to ona jako pierwsza została oficjalnie opublikowana w styczniu 1997 roku w specyfikacji RFC 2068. Dwa lata po tej publikacji zostało wprowadzonych kilka usprawnień w

wydajności lecz kod protokołu pozostał taki sam, jednak nowsza wersja specyfikacji to RFC 2616.

Podobnie jak to miało miejsce podczas przechodzenia z wersji 0.9 na nowszą, również teraz znaleziono wiele problemów i wprowadzono usprawnienia w stosunku do wersji 1.0. Przykładowe optymalizacje to między innymi kolejkowanie żądań, transmisja danych w blokach, żądania przesyłania zakresu bajtów, kodowanie przesyłanych danych czy podtrzymywanie połączenia.

Poniżej znajduje się przykładowa sesja HTTP 1.1:

```
$> telnet website.org 80
Connected to xxx.xxx.xxx.xxx

GET /index.html HTTP/1.1 ①
Host: website.org
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_7_4)... (snip)
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,sdch
Accept-Language: en-US,en;q=0.8
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.3
Cookie: __qca=P0-800083390... (snip)

HTTP/1.1 200 OK ②
Server: nginx/1.0.11
Connection: keep-alive
Content-Type: text/html; charset=utf-8
Via: HTTP/1.1 GWA
Date: Wed, 25 Jul 2012 20:23:35 GMT
Expires: Wed, 25 Jul 2012 20:23:35 GMT
Cache-Control: max-age=0, no-cache
Transfer-Encoding: chunked

100 ③
<!doctype html>
(snip)

100
(snip)

0 ④

GET /favicon.ico HTTP/1.1 ⑤
Host: www.website.org
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_7_4)... (snip)
Accept: */
Referer: http://website.org/
Connection: close ⑥
Accept-Encoding: gzip,deflate,sdch
Accept-Language: en-US,en;q=0.8
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.3
```

Cookie: \_\_qca=P0-800083390... (snip)

HTTP/1.1 200 OK ⑦  
Server: nginx/1.0.11  
Content-Type: image/x-icon  
Content-Length: 3638  
Connection: close  
Last-Modified: Thu, 19 Jul 2012 17:51:44 GMT  
Cache-Control: max-age=315360000  
Accept-Ranges: bytes  
Via: HTTP/1.1 GWA  
Date: Sat, 21 Jul 2012 21:35:22 GMT  
Expires: Thu, 31 Dec 2037 23:55:55 GMT  
Etag: W/PSA-GAu26oXbDi

(icon data)  
(connection closed)

- ①- Żądanie pliku HTML wraz z metadanymi opisującymi kodowanie, zestaw znaków i ciasteczką.
- ②- Blok z odpowiedzią na pierwsze żądanie HTML.
- ③- Liczba bajtów w bloku danych wyrażona jako liczba szesnastkowa w kodzie ASCII (256 bajtów)
- ④- Koniec strumienia bloków danych z odpowiedzią.
- ⑤- Żądanie pliku ikony przesyłany tym samym połączeniem TCP
- ⑥- Informacja do serwera, że połączenia nie będzie ponownie wykorzystane.
- ⑦- Odpowiedź z plikiem ikony i zamknięcie połączenia

Pierwszą różnicą jest żądanie dwóch obiektów przez TCP, co wcześniej było niemożliwe poprzez zamykane połączenia po zakończeniu żądania. Podtrzymywanie i ponowne wykorzystanie tego samego połączenia znacznie umożliwia znacznie szybsze dostarczenie mu odpowiedzi. W celu zamknięcia połączenia wykorzystany jest nagłówek (Connection: Close).

W ulepszonej wersji protokołu HTTP 1.0 też została wprowadzona funkcja podtrzymywania połączenia poprzez użycie nagłówka (Connection: Keep-Alive). Nagłówek ten jednak nie jest wymagany, jeśli serwer aplikacji webowej korzysta z nowszej wersji protokołu.

## HTTP 2.0

Coraz powszechniejsze i bardziej zaawansowane aplikacje webowe, już nawet nie statyczne czy interaktywne strony WWW, wymagają także większej wydajności. W odpowiedzi na to z pozoru banalne zapotrzebowanie powstaje protokół HTTP w wersji 2.0. Zakłada on zmniejszenie opóźnień, zwiększenie wydajności transmisji danych oraz uzyskanie większej przepływności. W tym momencie nie planuje się zmieniać takich cech protokołu jak nagłówki HTTP oraz ich wartości i funkcje.

Wszystkie aplikacje webowe, które już działają w Internecie i zostały uruchomione na serwerach w momencie pełnego wdrożenia nowego protokołu sieciowego będą z niego korzystały bez konieczności wprowadzania zmian w kodzie aplikacji. Ważne jest, aby serwer oraz klient obsługiwały to połączenie. Zmiany w działaniu aplikacji powinny być jednak zauważalne, jeśli prócz serwera, aplikacja jest dobrze skonfigurowana i wolna od błędów, jakie może wskazać np. „Snail Project” przygotowany specjalnie w celu wykonania testu wydajnościowego aplikacji. Warto także pamiętać, że tworząc najbardziej nowoczesną aplikację trzeba zadbać o kompatybilność wsteczną protokołu. Serwer Apache (jeśli programista nie skonfiguruje połączenia inaczej) zawsze wykorzystuje najwyższą możliwą wersję protokołu HTTP.

## SPDY

SPDY jest bardziej zbiorem rozwiązań od firmy Google, które mają przyspieszyć ładowanie się stron Internetowych poprzez różne modyfikacje używanego obecnie standardu HTTP 1.1, niż odrębną wersją protokołu sieciowego. Można SPDY uznać bardziej za rozszerzenie niż nowy standard. Nowe idee firmy z Mountain View to między innymi kompresja nagłówków (lepsza niż do tej pory), priorytetyzacja pakietów oraz multipleksowanie. Ostatnie zagadnienie odnosi się do propozycji, aby można było pobierać wiele plików w jednym zapytaniu. W chwili obecnej każda przeglądarka Internetowa wykorzystuje nowe połączenie dla każdego elementu pobieranej strony WWW, który jest oddzielnym plikiem. Osobno pobierane są pliki grafiki, pliki Flash, skrypty JavaScript, a także kaskadowe arkusze stylów, które nie zostały osadzone w kodzie strony. Google proponuje także ulepszenie obecnego protokołu TCP, który pozostaje niezmieniony od 1981 roku. Kolejne proponowane zmiany to zmniejszenie czasu, po którym przesyłany pakiet zostaje uznany za utracony z 3 sekund do 1 sekundy, zwiększenie okna przeciążenia (congestion window) z 3 do 10 pakietów oraz wdrożenie protokołu TCP Fast Open. HTTP Working Group [5], ma na celu wdrożenie tego usprawnienia do protokołu HTTP 2.0. Następnie twórcy serwerów WWW i przeglądarki będą mogli je zaimplementować. Według założeń przyspieszy to znaczco ładowanie się stron internetowych.

## Wydajność części front-endu i back-endu

Front-end to część strony czy aplikacji webowej, którą użytkownik może zobaczyć lub „doświadczyć”. Ponieważ ludzi uważa się za wzrokowców, doświadczenia użytkownika są bardzo ważne, aby przyciągnąć uwagę odbiorcy.

Na „front-end” składa się między innymi hipertekstowy język znaczników – HTML, kaskadowe arkusze stylów w plikach z rozszerzeniem CSS, mające na celu „ozdobienie” wcześniej wspomnianych znaczników czy zawartych w nich informacji. Współczesnym użytkownikom często serwuje się także odrobinę „magii” w postaci skryptów języka JavaScript czy biblioteki jQuery. Wszelkie pojawiające się i znikające elementy, ruszające się przyciski czy wyskakujące okienka, można stworzyć pisząc skrypty, które później są dołączane do plików .html.

Prócz dostarczania „user experience” zadaniem front-endu jest także komunikacja z back-endem aplikacji przekazywanie jej interakcji użytkownika lub wyświetlanie mu informacji od serwera. Back-end zazwyczaj jest dzielony na trzy części: serwer, aplikację (działającą na serwerze) oraz bazę danych. Każda z nich jest w pewien sposób uniwersalna np. gdyby aplikację (a raczej logikę aplikacji) przenieść z jednego serwera na inny oraz podpisać do bazy danych z innymi (ale spójnymi) danymi, aplikacja też by działała. Żeby zaprogramować

aplikację działającą po tzw. stronie serwera najczęściej korzysta się z takich języków programowania jak: PHP, Java, Ruby czy Python.

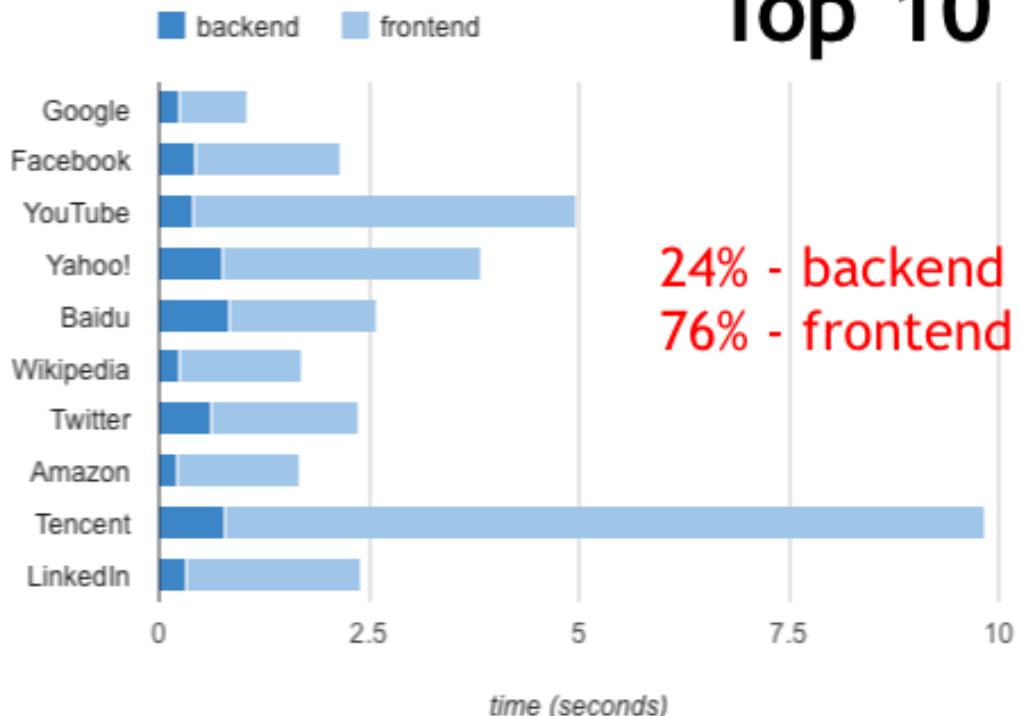
Wybór konkretnej technologii to bardzo ważny element tworzenia aplikacji webowej, jednak dyskusja na temat, który z nich jest najszybszy, najwydajniejszy, czy którego kod jest najłatwiejszy w utrzymaniu może pochłonąć kilka lat. Przede wszystkim należy dobrą odpowiednio język do tworzonej aplikacji. Dla wszystkich jednak można korzystać ze wspólnych koncepcji back-endowych, wymienionych między innymi w opisany projekcie „Snail Project”. Sugestie zawarte w wynikach stanowią uniwersalne rozwiązania, które każdy programista może zastosować w swoim projekcie, aby jego aplikacja działała bardziej wydajnie.

Działania części określanej jako front-end oraz back-end mimo różnych zadań są ze sobą połączone. Przykładowo osoba chcącą zarezerwować bilet lotniczy, po wpisaniu odpowiedniego adresu w swojej przeglądarce Internetowej, w pierwszej kolejności doświadczy kontaktu z wyglądem strony. Wszystkie informacje, które można od razu przeczytać na pierwszej stronie zazwyczaj nie wymagają od razu połączenia z bazą danych. Dopiero w momencie sprawdzania aktualnej informacji i ilości miejsc dla danego lotu, następuje komunikacja z serwerem, wyszukanie odpowiednich informacji w bazie danych, ewentualna filtracja ich przez kontrolery i zwrócenie ich do części wizualnej, gdzie użytkownik sam dokona wyboru czy dokonać kolejnej akcji np. rezerwacji biletu. Ciężko by było stworzyć taką aplikację, gdyby nie wykorzystać bazy danych znajdującej się po stronie serwera, czy nie używać arkusza stylów lub skryptu pobierającego wartość maksymalnej ceny biletu podanej przez użytkownika.

Steve Souders na swoim blogu [6] opublikował testy dotyczące czasu, jaki jest potrzebny do załadowania się strony internetowej pod względem podziału na front-end oraz back-end. Wyniki te dowodzą, że front-end to nawet 92% czasu, jaki jest potrzebny na gotowość aplikacji.

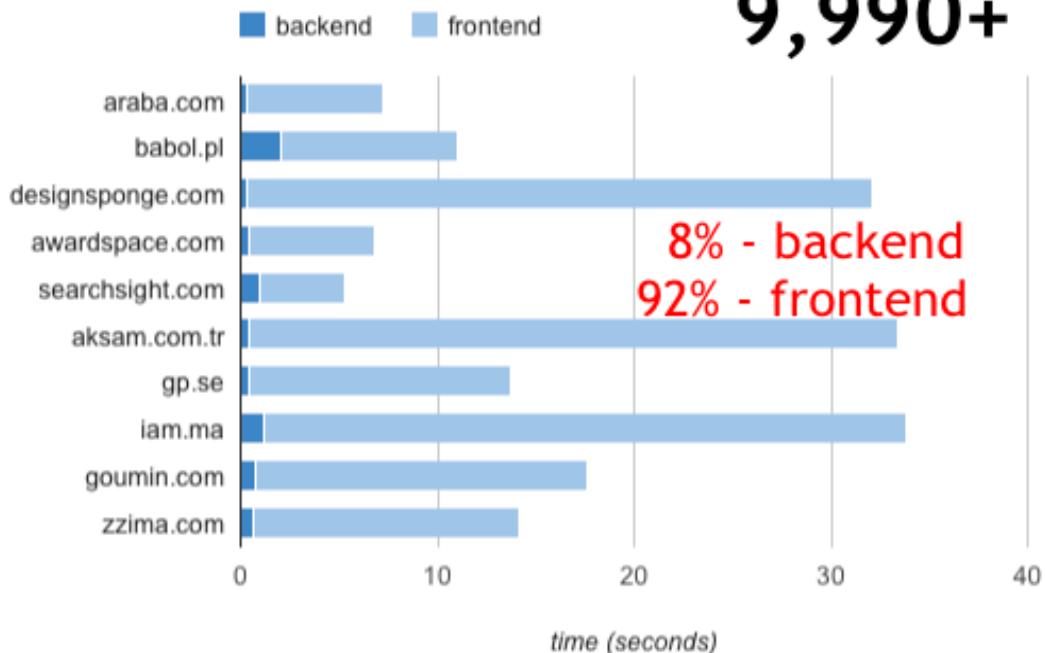
Poniżej przytoczę wyniki, jakie uzyskał Steven Souders. Sprawdzian został przeprowadzony dla serwisów znajdujących się w rankingu TOP 10 najczęściej odwiedzanych stron internetowych na świecie.

# Top 10



Rysunek 1.1 Top 10 najpopularniejszych stron WWW na świecie  
Źródło: <http://www.stevesouders.com/blog/2012/02/10/the-performance-golden-rule/>

9,990+



Rysunek 1.2 Ostatnie 10 stron WWW rankingu top 10000  
Źródło: <http://www.stevesouders.com/blog/2012/02/10/the-performance-golden-rule/>

Opublikowane rezultaty są jednoznaczne, back-end to część pracy programisty, który w znacznie mniejszym stopniu wpływa na wydajność działania aplikacji webowej, anieżeli część „front-endowa”. Kiedy nie zastosuje się rad, które sugeruje wykonany i opisany „Snail project”, nawet najwydajniejszy na świecie serwer nie poradzi sobie z odpowiedziami na „niepotrzebne” żądania, czy przesyłaniem do klienta ciężkich plików graficznych lub nieozptymalizowanych skryptów blokujących ładowanie się pozostałej części strony.

### **Wydajność sieci bezprzewodowych**

Początek sieci bezprzewodowych sięga lat 70 XX wieku, kiedy to powstała pierwsza sieć o nazwie AlochaNET. Była to pierwsza publiczna prezentacja sieci bezprzewodowej opracowanej w 1971 roku na Uniwersytecie Hawajskim. Sposób działania był bardzo prosty. Nadawca zaczynał transmisję informacji zawartych w ramkach, a następnie oczekiwali potwierdzenia odebrania ramki od odbiorcy. Po tak prostej komunikacji cała transmisja zaczynała się od nowa. To rozwiązanie miało bardzo dużo wad, gdyż sposób działania przypominał komunikację „walkie talkie” - komunikacja mogła następować z jednej strony jednocześnie. To jeden z największych problemów tego rozwiązania. Eliminował je z wykorzystania go do zaawansowanych rozwiązań technologicznych. Rewolucję okazało się opracowanie przez Roberta Metcalfe i Davida Boggs'a Etherntu opartego na kablu koncentrycznym. Ostateczna specyfikacja komunikacji przez Ethernet została opracowana przez takie firmy jak Xerox, Intel i Digital Equipment i opublikowana w 1980 roku. W 1985 roku dokument został zaakceptowany przez "Institute of Electrical and Electronics Engineers", ustanawiając normę IEEE 802.3. Od tego momentu, dzięki szerokiemu zastosowaniu w przemyśle, rachunkowości, administracji, sieć Ethernet zyskała na popularności. Pierwsze karty bezprzewodowe zostały wyprodukowane w latach 90, jednak ich cena oraz dostępność w sklepach blokowała rozwój technologii sieciowo-radiowych. Organizacja Institute of Electrical and Electronics Engineers w 1997 roku organizacja stworzyła standard sieciowy na częstotliwości radiowej 2,4 GHz, uzyskał on oznaczenie 802.11. Standard posiadał bardzo niską przepustowość około 2 Mb/s. Dopiero rozwój tego rozwiązania oznaczony 802.11b umożliwiał korzystanie z prędkości do 11Mb/s, a zastosowanie częstotliwości 2,4 GHz umożliwiło działanie sieci radiowej bez zakłóceń przez urządzenie wykorzystujące mikrofale. Kolejnym krokiem było stworzenie standardu 802.11a, który działał przy wykorzystaniu częstotliwości 5GHz i przepustowości 54Mb/s. Drugi standard sieci nie zyskał poparcia przez brak kompatybilności ze standardem 802.11b oraz mniejszy zasięg efektywny, a także wysoką ceną urządzeń obsługujących ten standard. Jedynym atutem tego standardu było to, że posiadał osiem nie pokrywających się kanałów dla częstotliwości fal radiowych. Następnie w roku 2002 na targach Comdex zaprezentowano nowe urządzenie pracujące w standardzie IEEE 802.11g. Standard bardzo szybko został zaakceptowany przez organizacje IEEE i to spowodowało iż znalazły się u większości producentów.

Tabela 1.1 Porównanie rozszerzeń standardu 802.11

Protokół 802.11	Rok publikacji	Częstotliwość (GHz)	Szerokość pasma (MHz)	Prędkość transmisji danych na strumień (Mb/s)
g	1999	2,4	20	1, 2, 5,5, 11
b	2003	2,4	20	6, 9, 12, 18, 24, 36, 48, 54

n	2009	2,4	20	7.2, 14.4, 21.7, 28.9, 43.3, 57.8, 65, 72.2
n	2009	5	40	15, 30, 45, 60, 90, 120, 135, 150
ac	2014	5	20,40,50,160	~866,7

Obecnie najbardziej rozpowszechnionymi standardami są „b” oraz „g” korzystające z szerokości pasma 2,4 GHz. Jeśli chce się przyspieszyć działanie sieci, najlepiej wykorzystać standardy 802.11 „n” oraz wdrażany „ac”. Najnowsze trendy podwajają szerokość pasma z 20 MHz do 40 MHz na każdy kanał. Wykorzystują modulację wyższego rzędu oraz zwielokrotniają moduły radiowe tak, aby transmitować wiele strumieni jednocześnie.

Sieci bezprzewodowe przeżywają w ostatnich latach niekończącą się ewolucję. Powszechny dostęp do Internetu staje się wręcz standardem dla ludzkości. Dostęp do poczty elektronicznej, obsługa połączeń telefonicznych, przeglądanie stron www, z każdego miejsca na ziemi niezależnie od położenia, to oczekiwania współczesnego społeczeństwa.

Siecią bezprzewodową jest każda sieć, która nie łączy urządzeń wykorzystywanych do komunikacji za pomocą kabla. Sieć LTE jest w stanie transmitować dane z danego serwera do urządzenia mobilnego użytkownika tak, aby mógł on obejrzeć daną stronę Internetową bez użycia kabla typu skrętka. Dane aplikacji internetowych mogą być przesyłane nie tylko przy użyciu sieci mobilnych jak 4G. Systemy webowe korzystające z płatności mogą używać sieci NFC. Natomiast Bluetooth może zostać użyty do komunikacji P2P czy Wi-Fi, lecz nie do przesyłania danych na drodze serwer-klient, a na przykład strumieniowania danych audio czy wideo.

Mechanizmy dostarczania danych za pomocą komunikacji radiowej bardzo się różnią od architektury „kablowej” jednak doświadczenia użytkownika powinny być identyczne. Zarówno jeśli chodzi o wydajność jak i oczekiwane rezultaty takiego połączenia.

Obecnie najpopularniejszymi sieciami bezprzewodowymi są technologie takie jak LTE, HSPA, Wi-Fi, WiMax, Bluetooth czy standardy 3G. Żeby zrozumieć podstawowe zasady działania i założenia takich sieci opisane zostały te, z których najczęściej korzystają użytkownicy aplikacji internetowych. Ponieważ programiści optymalizując swoje aplikacje internetowe pod względem wydajności i sposobu działania sieci bezprzewodowych mogą jedynie zyskać w oczach użytkowników, nie należy pomijać tych informacji.

Matematyczny model Twierdzenia Shannona-Hartleya [7], pomaga w określeniu pojemności kanału niezależnie od zastosowanej technologii.

$$C = BW * \log_2 \left( 1 + \frac{S}{N} \right)$$

(Wzór 2.1 Matematyczny model Twierdzenia Shannona-Hartleya)

C - oznacza pojemność kanału w bitach na sekundę

BW – dostępna szerokość pasma w hercach

S – sygnał w watach

N – szum w watach

Powyższa bardzo uproszczona formuła przedstawia najważniejsze aspekty połączenia bezprzewodowego. Prócz wykonania testów wydajnościowych np. za pomocą projektu Snail opisanego we wcześniejszej części pracy, oraz optymalizacji części front-endowej czy back-endowej należy także mieć świadomość jak działa większość bezprzewodowych sieci bez znaczenia jaki jej rodzaj, wersja, czy akronim został akurat zastosowany. Dostępna szerokość pasma, siła sygnału czy odległość między nadajnikiem i odbiornikiem też w istotny sposób wpływają na prędkość przesyłania danych do np. urządzeń mobilnych.

Nie wszystkie zakresy częstotliwości są tak samo wydajne. Im niższa częstotliwość nadawania sygnału tym większy zasięg i pokrycie, lecz wymaga to budowy większej infrastruktury np. długich anten. Odwrotnie jest w przypadku użycia sygnału o wysokiej częstotliwości, można wtedy przesyłać większą ilość danych ale zasięg jest dużo mniejszy. To znaczy pokrycie jest mniejsze oraz wymaga większej ilości nadajników tak, aby klient łączył się z najsilniejszym z nich.

Korzystanie z niektórych zakresów częstotliwości jest bardziej wydajne od innych. Na przykład aplikacje, które wykorzystują tylko nadawanie (np. gdyby użyto infrastruktury i aplikacji wysyłającej tylko powiadomienia push), będą działać najlepiej na niskich częstotliwościach. Natomiast aplikacje wykorzystujące komunikację dwukierunkową, będą działały dużo wydajniej kiedy zastosuje się mniejsze komórki, zapewniające wyższą przepustowość i jednocześnie mniejszą konkurencję.

Siła sygnału jest, po szerokości pasma, drugim najważniejszym czynnikiem, który może stanowić pewne ograniczenia w całej sieci bezprzewodowej. Jak wspomniano wcześniej siła sygnału to stosunek sygnału (mierzony w watach) do szumu (również mierzony w watach).

$$\left(\frac{S}{N}\right)$$

(Wzór 2.2 Siła sygnału)

Sygnal musi być tym silniejszy im większe zakłócenia występują na jego drodze. Zakłócenia mogą być emitowane przez różne urządzenia emitujące fale radiowe o częstotliwości zbliżonej do 2,5 GHz np. kuchenka mikrofalowa. Najczęściej są to jednak inne urządzenia Wi-Fi działające na tym samym kanale lub częstotliwości i szerokości wykorzystywanej pasma, takie jak punkty dostępu na przystankach autobusowych, czy routery sąsiadów.

Czynniki wpływające na wydajność sieci bezprzewodowych to nie różnica odległości między odbiorcą i nadawcą. Ważny jest poziom szumów w danych lokalizacjach. Moc przetwarzania oraz schemat modulacji sygnału. Siła nadawanego sygnału oraz moc jego odbierania. Zakłócenia generowane przez użytkowników w tych samych sieciach czy lokalizacjach oraz użytkowników znajdujących się w podobnej lokalizacji lecz innych sieciach.

W świecie idealnym każda sieć bezprzewodowa byłaby nadawana w innej szerokości pasma i na innej częstotliwości tak, aby sieci nie zakłócały się wzajemnie. Jednak jest to niemożliwe do zastosowania, szczególnie w dużych miastach i aglomeracjach takich jak Wrocław, gdzie bezprzewodowa komunikacja jest bogato wykorzystywana. Żeby aplikacja webowa była wydajna i działała w „każdych warunkach”, należy mieć na uwadze wykorzystanie i sposób działania oraz wydajność właśnie takich sieci.

## Mobilne witryny Internetowe

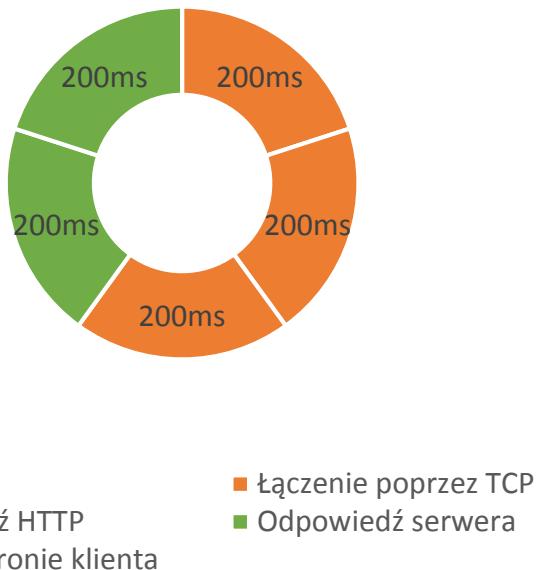
Obsługa stron internetowych z punktu widzenia telefonów komórkowych niewiele się różni od obsługi tych samych witryn za pomocą komputera stacjonarnego czy laptopa. Mobilna przeglądarka Internetowa po wywołaniu adresu odpowiedniej strony, szuka serwera o określonym adresie IP, wysyła mu żądanie wyświetlenia zawartości strony Internetowej i czeka na odpowiedź generując użytkownikowi telefonu gotową stronę www. Wszystko wydaje się być proste a sposób działania znany od kilkudziesięciu lat. Dlaczego w takim w razie większość programistów dba o to, aby ich strony Internetowe były przystosowane do obsługi na urządzeniach takich jak smartfony czy tablety ? Największy problem to przepustowość oraz opóźnienia sieci takich jak 2G, 3G, czy 4G, z których korzystają urządzenia mobilne.

Według dokumentacji Google Insights [8], badania przeprowadzone przez Google określają, że uwaga skupiona przez osobę obsługującą np. tablet na ładującej się stronie Internetowej nie przekracza jednej sekundy. Rozkojarzenie spowodowane zbyt długim oczekiwaniem na pojawienie się zawartości strony może doprowadzić do porzucenia strony przez użytkownika na rzecz innej, która wyrenderuje się szybciej.

Oczywiście obecne rozwiązania technologiczne nie pozwalają na pełne wyświetlenie się strony w określonym przez Google czasie. Należy zadbać jednak, aby część widoczna na ekranie pojawiła się możliwe jak najszybciej tak, aby skupić uwagę odbiorcy i umożliwić mu korzystanie ze strony, a pozostałe elementy systematycznie przesyłać do urządzenia.

Kryterium renderowania widocznej części witryny w czasie krótszym niż jedna sekunda to pewne wyzwanie nie tylko dla developerów, ale także dla mobilnych sieci, ponieważ niektóre problemy nie występują w innych sieciach kablowych czy światłowodowych. Obecnie najpopularniejszą siecią na świecie jak i w Polsce jest tzw. 3G bazująca na standardzie UMTS. Jej następcą czyli 4G jest dopiero w pełni wdrażana i rozpowszechniana na świecie, przez producentów oraz dostawców telefonii komórkowej. Dotyczy to nie tylko budowanej infrastruktury, ale także najnowszej wersji telefonów obsługujących daną wersję standardu sieci telekomunikacyjnej.

## Renderowanie witryny w czasie jednej sekundy



Rysunek 1.3 Etapy i czas renderowania witryny przez urządzenie mobilne  
 Źródło: Opracowanie własne

Wykres 1.2 przedstawia typową sekwencję komunikacji między przeglądarką a serwerem. 600 milisekund zostało wykorzystane na wyszukiwanie DNS i identyfikację nazwy hosta (np. pwr.edu.pl) powiązanej z adresem IP, wymianę danych w związkach z uzgodnieniami TCP oraz przesłanie żądania HTTP. W pozostałych 400ms serwer musi dać odpowiedź klientowi, aplikacja kliencka musi wykonać kod a przeglądarka wyrenderować treść i pokazać ją użytkownikowi urządzenia mobilnego.

Aby zmieścić się w ciągu jednej sekundy, serwer powinien zwrócić początkową część HTML-a w czasie 200ms lub nawet krótszym, aby mieć cenny zapas czasu na przesłanie danych.

Należy się przyjrzeć przekierowaniom na swojej stronie tak, aby ich liczba była jak najmniejsza. Ponieważ każde dodatkowe przekierowanie HTTP to dodatkowy czas na wymianę danych lub wyszukanie serwera DNS, co przekłada się na większe opóźnienia w sieciach 3G i wyższych.

Protokół TCP szacuje wydajność połączenia według „Strategii powolnego startu”. Według założeń ma zapobiegać spowolnieniom w protokole transmisji danych. „Slow-start” (ang. powolny start) zazwyczaj jest używany równolegle z innymi algorytmami, które zapobiegają wysłaniu do sieci większej ilości danych niż jest ona w stanie odebrać.

Działanie algorytmu polega na tym, że okno TCP początkowo mieści jeden maksymalny segment. Okno to jest zwiększane o jeden segment wraz z każdym odebranym pakietem ACK, to znaczy, że okno podwaja się co RTT. Powolny start kończy działanie, kiedy nastąpi utrata pakietu lub okno osiągnie maksymalną szerokość.

Na podstawie powyższego opisu działania algorytmu powolnego startu TCP, należy wywnioskować, że każde nowe połączenie klient-serwer nie korzysta od razu z pełnej przepustowości łączna. Biorąc pod uwagę fakt, że w ramach nowego połączenia serwer jest w

stanie wysłać maksymalnie 10 pakietów czyli ok. 14 KB, a następnie, żeby zwiększyć transfer musi poczekać na odpowiedź klienta, należy ograniczyć liczbę sesji wymiany danych. Można tego dokonać poprzez zminimalizowanie rozmiaru pliku z widoczną częścią strony, aby ten nie przekraczał wspomnianych 14 KB. Zabieg ten pozwoli na to żeby przeglądarka w ciągu jednej sesji wymiany danych była w stanie odrysować szablon strony.

Tak jak we wcześniej opisanych testach „pełnych stron www”, również w wersjach mobilnych należy unikać kodów JavaScript umieszczonych w osobnych plikach, blokujących zasoby strony w części widocznej na ekranie użytkownika. Jeśli przeglądarka renderując widok strony napotka w kodzie link do pliku ze skryptem HTML, zacznie go interpretować. Dodatkowo, jeśli będzie to skrypt zewnętrzny lub niesynchroniczny, program zatrzyma wczytywanie kolejnych elementów strony, dopóki nie przetworzy obecnych zasobów. Zarówno kod CSS jak i JavaScript, jeśli są niezbędne do działania widocznej dla odbiorcy części strony, powinny zostać umieszczone bezpośrednio w kodzie HTML. Rozwiążaniem pośrednim może być umieszczenie odpowiednich skryptów czy arkuszy stylów na końcu kodu HTML. W ten sposób użytkownik zacznie analizować widoczny dla niego tekst, czy szablon strony, a dodatkowe wizualne efekty zostaną załadowane w odpowiedniej kolejności. Często w kodzie używana jest biblioteka jQuery, która wzbogaca statyczne strony o animacje i dodatkowe funkcjonalności. Bez najmniejszych obaw jej opcje można wywoływać już po załadowaniu się strony tak, aby nie blokowała wczytywania się pozostałych elementów. Wpłynie to pozytywnie na skrócenie czasu renderowania witryny.

Interpretacja kodu JavaScript jak i CSS powinna zająć ostatnie 200ms dopełniające pełną sekundę, lecz jest to dość problematyczne do określenia ze względu na różnorodność urządzeń mobilnych. Tablety czy telefony różnią się od siebie nie tylko pamięcią, mocą procesora, ilością rdzeni i szybkością działania aplikacji jak i samego mobilnego systemu operacyjnego. W związku z tą różnorodnością, łatwiej jest zadbać o kod strony, jej poziom skomplikowania oraz całkowitą spójność i kompresję punktów składowych tak, aby jej szybkość i poprawność działania pozytywnie zaskakiwała odbiorców wszystkich możliwych urządzeń wyświetlających konkretną witrynę internetową.

Sieć 4G to następca systemu trzeciej generacji. Założenia nowszego standardu są następujące:

- Komutacja pakietów na protokole IP
- Rzadsze występowanie przestojów i błędów transferu
- Istotne przyspieszenie możliwości transferowych
- Szybszy czas reakcji w porównaniu do sieci 3G
- Uproszczenie struktury sieci szkieletowej

W rezultacie dostęp do strony jest szybszy i mniej wadliwy. Nie należy jednak w pełni ufać nowemu standardowi i zaniedbać optymalizację własnego serwisu www. Zdecydowana większość ludzi na świecie nie posiada najnowszych urządzeń obsługujących sieci LTE i wciąż korzystają z technologii przesyłania danych 3G.

Nad najnowszą technologią przesyłu danych prócz takich potęg jak Nokia, Huawei [9], Samsung, Ericsson [10] pracuje także Unia Europejska [11].

Korzyści wynikające z założeń sieci 5G mogą być przełomowe. Nowy system ma charakteryzować się odpornością na spadki wydajności związane z liczbą jednocześnie

korzystających z sieci użytkowników. Prędkość przesyłania danych ma osiągać nawet 100 Gbps (obecnie LTE osiąga przepustowość do 300 Mbps).

Wspomniani twórcy najnowszej technologii szacują pojawienie się najnowszej edycji standardu telefonii komórkowej dopiero w roku 2025.

Bardziej efektywne wykorzystanie TCP można uzyskać poprzez użycie protokołu HTTP 2.0 czy rozszerzenia HTTP 1.1 – SPDY. Nowy standard protokołu przesyłania dokumentów hipertekstowych nie jest jeszcze w pełni ukończony, jednak można z nich korzystać w wersjach testowych modułów w serwerów takich jak Nginx czy Apache. Pozwoli to między innymi na kompresję nagłówków, używanie priorytetów czy użycie mechanizmu Server Push, który poprzez eliminację opóźnień sieci zwiększa wydajność połączenia.

## **2. Opis narzędzia Snail Project**

### **Agile**

W trakcie tworzenia Projektu o nazwie Snail wykorzystane zostało „programowanie zwinne” - (ang. Agile software development). Ponieważ założeniem było jak najszybsze otrzymywanie wyników poprzez wykonywanie skryptów, „layout „był tworzony „w trakcie” programowania innych zadań. Programowanie iteracyjno-przyrostowe, powstałe jako alternatywa do tradycyjnych metod typu waterfall, świetnie się sprawdziło podczas implementowania różnego rodzaju skryptów np. Insight. Jednym z założzeń rozwiązań Agile jest praca w zespołach, które mogą niezależnie od siebie tworzyć pewne części danego projektu. W tym przypadku zespół składał się z jednej osoby oraz „odbiorcy”. Współpraca owocowała wieloma pomysłami oraz zmianami w trakcie tworzenia projektu, podejście typu Waterfall mogłoby się zakończyć niepowodzeniem projektu, lub brakiem możliwości wprowadzania jakichkolwiek zmian po fazie projektowej.

Agile bardzo często znajduje zastosowanie w małych zespołach programistycznych, w których jest dobra komunikacja, tak by nie tworzyć rozbudowanej dokumentacji kodu. Kolejne etapy wytwarzania oprogramowania zamknięte są w iteracjach, w których za każdym razem przeprowadza się testowanie wytworzzonego kodu, zebranie wymagań, planowanie rozwiązań itd. Metoda nastawiona jest na szybkie wytwarzanie oprogramowania wysokiej jakości oraz regularna adaptacja do zmieniających się wymagań.

### **Pierwsza iteracja Snail Project**

Zanim zaimplementowano skrypty badające wydajność serwisów webowych, stworzony został *layout* Projektu. Po wpisaniu odpowiedniego adresu URL w przeglądarce wyświetli się górny pasek szerokości okna przeglądarki, na którym znajduje się nazwa projektu oraz adres IP użytkownika, który odwiedził stronę. Pozostałą część okna zajmuje obraz ślimaka, stanowiący symbol projektu, a na jego środku widnieje miejsce, w którym użytkownik powinien wprowadzić adres URL w postaci „[www.pwr.edu.pl](http://www.pwr.edu.pl)” bez poprzedzającego adres prefiksu <http://> oraz przycisk „Test”, który po wcisnięciu uruchomi wszystkie skrypty dokonujące odpowiednich analiz, parsując wyniki i wyświetli je w określonych odpowiednimi stylami sposob, aby były czytelne i zrozumiałe dla odbiorcy. Podczas oczekiwania na wykonanie się skryptów testujących do momentu wyświetlenia wyników widoczny jest „loader” w postaci obracających się kropek sugerujący, że właśnie trwa wykonywanie testu i należy poczekać na jego wynik. Rozmiar obrazu ze ślimakiem jest przystosowany do szerokości i wysokości okna przeglądarki a wyniki prezentowane są na białym półprzezroczystym tle, które jest nałożone na estetyczny gradient, tło całej witryny.

Do nadania Projektowi ostatecznego kształtu wykorzystane zostały takie technologie i frameworki, jak PHP, JavaScript, Ajax, Bootstrap, Mustache.js, jQuery oraz oczywiście HTML oraz CSS. Wszystko służyło odpowiedniemu ułożeniu i pozycjonowaniu elementów, oraz odpowiedniemu parsowaniu wyników zwracanych przez skrypty.

Cała aplikacja webowa jest anglojęzyczna, ponieważ czyni to ją bardziej uniwersalną i dostępną również dla programistów z zagranicy, którzy chcą dokonywać testów wydajności i optymalizacji stron Internetowych.

## **HTML**

HTML (ang. HyperText Markup Language) – hipertekstowy język znaczników, pozwala stworzyć szablon dokumentu, który zostanie zinterpretowany i wyświetlony w przeglądarce Internetowej [12]. W jego składni wykorzystywane są pary znaczników umieszczone w nawiasach ostrokątnych, np. `<title>`, `</title>` lub `<h1>`, `</h1>`. Między znacznikami `<script>` i `</script>` można osadzać ciąg instrukcji języków skryptowych np. JavaScript.

Uważa się, że język HTML nie należy do języków programowania z uwagi na to, że w jego składni nie przewidziano wyrażeń obliczeniowych, warunkowych, iteracyjnych jak również tworzenie zmiennych.

Najważniejszą cechą języka HTML, jest niezależność od systemu operacyjnego i wykorzystywanego sprzętu komputerowego co w dużej mierze przyczyniło się do jego popularności jak i rozwoju całej sieci Internet.

## **CSS**

CSS (ang. Cascading Style Sheets) – kaskadowe arkusze stylów. CSS jest językiem służącym do opisu elementów znaczników HTML oraz formatowaniu wyświetlania jego elementów w przeglądarce Internetowej [10]. Język ten został opracowany przez organizację W3C.

## **PHP**

PHP jest obiektowym językiem programowania, który został zaprojektowany, aby dynamicznie generować strony Internetowe oraz budować aplikacje webowe w czasie rzeczywistym.

## **JavaScript**

JavaScript, w skrócie JS – jest obiektowym, skryptowym językiem programowania, stworzonym przez firmę Netscape. Dzięki instrukcjom warunkowym, obiektom i pozostałym właściwościom języka, można reagować na zdarzenia zachodzące na stronie Internetowej (np. po kliknięciu w dany przycisk zmienić całe tło strony Internetowej), właśnie dzięki temu język JavaScript jest szeroko wykorzystywany przy tworzeniu stron, aplikacji i systemów webowych [10].

## **jQuery**

jQuery jest biblioteką JavaScript, która w prosty sposób umożliwia manipulację drzewem DOM. Jej największą zaletą jest to, że do działania nie wymaga manipulacji kodem HTML.

## **Ajax**

AJAX (ang. Asynchronous JavaScript and XML) - asynchroniczny JavaScript i XML, technologia pozwalająca przesyłać dane pomiędzy użytkownikiem a serwerem bez potrzeby przeładowywania całej strony WWW, co wiąże się z dłuższym czasem oczekiwania na wynik jakiejś akcji [10].

## **Twitter Bootstrap**

Twitter Bootstrap jest frameworkiem CSS. Jest on rozwijany przez zespół programistów Twittera. Framework implementuje zestaw klas, stylów i rozwiązań, które ułatwiają tworzenie interfejsu graficznego stron oraz aplikacji Internetowych.

## **Git**

Git to rozproszony system kontroli wersji. Stworzył go Linus Torvalds jako narzędzie wspomagające rozwój jądra Linux. Jest coraz popularniejszym narzędziem wśród zespołów

pracujących nad jednym wspólnym rozwiązaniem. Umożliwia wersjonowanie tworzonego oprogramowania i powrót do jego wcześniejszych kopii.

### PageSpeed Insight

Kolejną iteracją, zgodnie z założeniami programowania zwinnego, lecz pierwszym zaimplementowanym skryptem w projekcie został Page Speed Insights, który mierzy wydajność stron na komputerach. Skrypt w projekcie pobiera adres strony Internetowej podanej przez użytkownika, parsuje go, a następnie przesyła do serwera Google, gdzie następuje analiza strony. Po wykonaniu testu wyniki są zwracane do skryptu zaimplementowanego w Snail. Są to wyniki w postaci obiektów, następnie są one parsowane i wyświetlane użytkownikowi w sposób czytelny i zrozumiały w postaci tabeli.

Rezultat działania PageSpeed mieści się w zakresie od 0 do 100 punktów. Im wyższy wynik, tym lepszy. Wynik na poziomie co najmniej 85 punktów oznacza, że strona działa dobrze. Jednak na potrzeby Projektu, wynik jest wyświetlany w postaci procentowej. Po krótkim badaniu na grupie około 30 studentów, określono, że procent zoptymalizowania serwisu webowego jest dla odbiorców bardziej znaczącym wynikiem niż punkty. Google ostrzega, że narzędzie PageSpeed Insights jest nieustannie usprawniane, więc wynik może się zmieniać, gdy zostaną dodane nowe reguły oceny serwisów.

Insights sprawdza, jak można poprawić wydajność strony w następujących aspektach:

- czas wczytywania części strony widocznej na ekranie: czas, który upływa od momentu wysłania żądania nowej strony do momentu wyrenderowania części strony widocznej na ekranie przez przeglądarkę.
- czas pełnego wczytania strony: czas, który upływa od momentu, gdy użytkownik wysyła żądanie nowej strony do momentu, gdy strona zostanie w pełni wyrenderowana przez przeglądarkę.

Ze względu na dużą zmienność wydajności połączenia sieciowego PageSpeed Insights uwzględnia wyłącznie aspekty wydajności renderowania strony niezależne od sieci: konfigurację serwera, strukturę kodu HTML strony, a także korzystanie z zasobów zewnętrznych, takich jak obrazy, pliki JavaScript i CSS. Zastosowanie się do sugestii powinno poprawić względną wydajność strony, jednak wydajność bezwzględna wciąż pozostanie zależna od połączenia sieciowego użytkownika.

Skrypt zwraca wynik działania zawsze tych samych wartości:

- Unikanie zbędnych przekierowań
- Włączenie kompresji danych
- Wykorzystanie pamięci podręcznej przeglądarki
- Skrócenie czasu odpowiedzi serwera
- Minimalizowanie plików CSS
- Minimalizowanie plików HTML
- Minimalizowanie plików JavaScript
- Eliminowanie blokujących skryptów
- Optymalizacja obrazów
- Priorytetyzacja widocznej treści

Insights w Snail Project określa stopień wydajności strony w procentach. Rezultat „This site is optimized in 70%” należy czytać jako „Strona jest zoptymalizowana w 70 procentach”. Im wyższy wynik osiąga testowana aplikacja, tym mniej „kroków” należy wykonać, aby zoptymalizować jej wydajność i jakość działania. PageSpeed dokonuje analizy stron Internetowych za pomocą reguł ustalonych przez Google. Każda reguła opiera się na ogólnych zasadach działania strony Internetowej, takie jak buforowanie zasobów, przekazywanie danych i wielkości pobierania danych, odpowiednią minimalizację plików ze stylami, skryptami itp.

Każda reguła generuje zmiennoprzecinkową wagę, priorytet, z jakim należy uwzględnić pierwszeństwo wprowadzania zmian, aby osiągnąć lepszą wydajność. Na przykład, jeśli optymalizacja obrazów na testowanej stronie Internetowej pomoże zmniejszyć ich rozmiar o 1 MB, to ta reguła otrzyma większą przykładowo o 5,5845 wagę niż zminimalizowanie rozmiaru plików CSS, które np. zmniejszą rozmiar przesyłanych danych o 300 KB. Ważne jest jednak, aby zastosować wszystkie sugestie, które przy ponownym wykonaniu mogą dać lepszy rezultat punktowy. Innym przykładem może być sugestia włączenia kompresji danych w ustawieniach serwera, która będzie dwa razy ważniejsza od optymalizacji samych obrazów na stronie. Tabela wyników jest ustalona względem najważniejszych cech, jakie należy wprowadzić (według Google), lecz w drugiej kolejności należy zwracać uwagę na odpowiednie wagi, które wyświetlają się po wykonaniu testu.

## GTMetrix

Zgodnie z zasadą programowania zwanego, kolejną iteracją w projekcie Snail było zaimplementowanie skryptu, który niezależnie od miejsca przebywania użytkownika (w tym przypadku, osoby, która dokonuje testu) wykona analizę strony pod względem długości ładowania się elementów, czasu odpowiedzi, dokona pomiaru rozmiaru strony oraz utworzy wykres typu *waterfall*, z wynikami, aby były one zrozumiałe i czytelne dla odbiorcy.

GTmetrix jest projektem rozwijanym przez wyspecjalizowanych programistów w firmie „Gossamer Threads” w Vancouver. Firma posiada ponad 19 lat doświadczenia w pracy z technologiami webowymi. Serwis ten nie posiada swojego własnego systemu oceny strony, korzysta z PageSpeed Insight oraz narzędzia Yslow do prezentowania wyników użytkownikowi. Serwis ten udostępnia jednak API do skryptu, który spełnia założenia analizy strony, oraz umożliwia implementację na własnej stronie Internetowej, co sprawiło, że został on wykorzystany w Snail Project. Aby móc skorzystać z tego rozwiązania należało dokonać rejestracji na stronie [www.gtmetrix.com](http://www.gtmetrix.com) oraz wygenerować indywidualny „API Key” dla developera.

Skrypt pobiera adres z paska na głównej stronie Projektu, a następnie przesyła go na własny serwer i dokonuje weryfikacji, przed nieuprawnionym użyciem. Następnie żądanie serwera WWW jest przesyłane przez obiekt XMLHttpRequest do skryptu na serwerze Projektu Snail. JavaScript przetwarza odpowiednio ten obiekt prezentując zwrócony wynik w formie graficznej w odpowiednim miejscu na stronie. Cały test jest wykonywany przez „user-agent: Mozilla/5.0 (X11; Linux i686 on x86\_64; rv:25.0) Gecko/20100101 Firefox/25.0”.

## Modernizr

Korzystanie z nowych technologii Internetowych może wydawać się świetną zabawą, dopóki nie muszą być wspierane przez przeglądarki, które często nie obsługują najnowszych trendów, celowo bądź, tylko do czasu kolejnej aktualizacji przeglądarki. Modernizr [13] to biblioteka, która ułatwia pisanie warunkowego kodu JavaScript i CSS do obsługi użycia w każdej sytuacji.

Modernizr dokonuje sprawdzenia obsługi nowych trendów (np. obsługi cieni, responsywności, HTML Audio/Video) przez przeglądarkę, a następnie tworzy obiekt JavaScript z wynikami, i dodaje klasy do elementu HTML.

Biblioteka może testować obsługę, funkcjonalności HTML5, CSS3 dla przeglądarki gdzie została uruchomiona, dlatego wynik jaki jest zwracany na stronie Snail Project dotyczy przeglądarki przez, którą użytkownik wszedł na stronę projektu. Dzięki tej wiedzy, autor serwisu jest w stanie sprawdzić, jak wykorzystane technologie są interpretowane przez różne przeglądarki internetowe (przeglądarka musi być fizycznie obecna na urządzeniu użytkownika, i to za jej pomocą należy wykonać badanie).

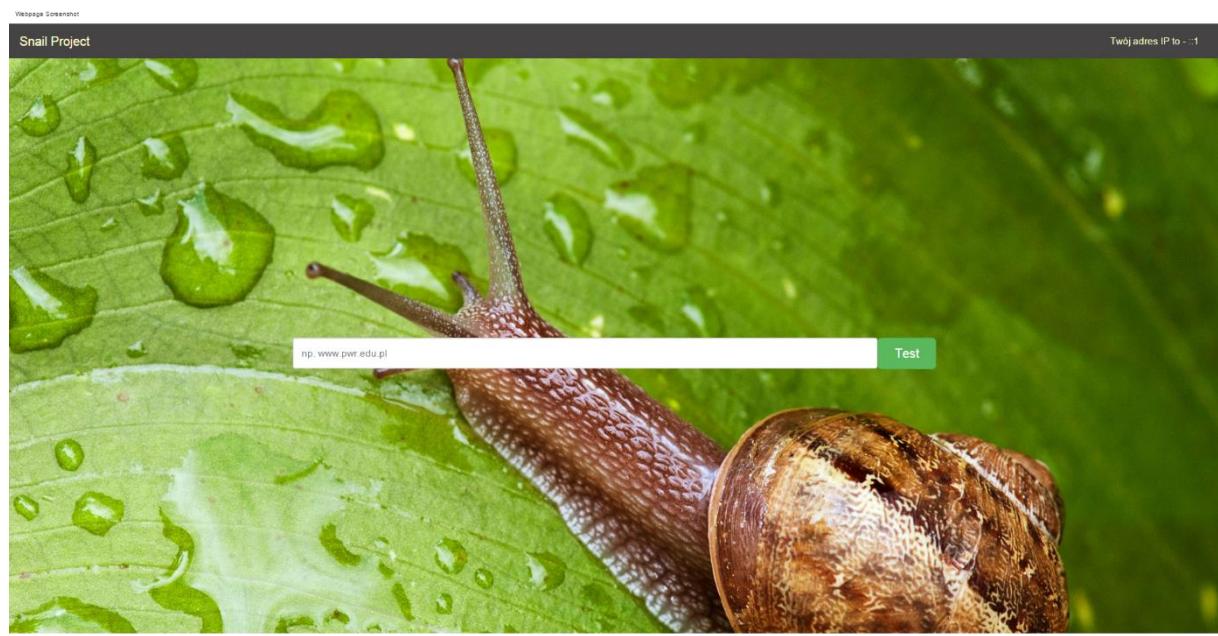
### Analiza struktury szablonu HTML

Zgodnie z duchem Agile, po konsultacjach z Profesorem Leszkiem Borzemskim ustalono następną iterację powstawania projektu o nazwie Snail jaką jest sprawdzenie poprawności kodu HTML testowanej strony z wykorzystaniem Validatora konsorcjum W3C.

W3C to skrót od World Wide Web Consortium, które zajmuje się ustanawianiem standardów sieci WWW takich jak skrypty, style, znaczniki HTML itp.

Validator, jest dostępny na stronie Internetowej konsorcjum jednak możliwe jest także wykorzystanie go w projektach bez implementowania całego systemu testującego. Działa to na zasadzie zadania zapytania AJAX serwera konsorcjum, a następnie odpowiednie sparsowanie wyniku, który jest zwracany w formacie JSON. Po pobraniu adresu testowanej strony z odpowiedniego miejsca na stronie głównej projektu Snail, wysyłane jest odpowiednie żądanie a po jego przetworzeniu wynik jest wyświetlany w odpowiednim miejscu na stronie.

Wynikiem jest lista błędów (o ile takie wystąpią), z krótkim komentarzem do problemu jaki nastąpił, oraz linia i kolumna w kodzie testowanej strony, w której dany błąd się znajduje. Poniższy obrazek przedstawia główną stronę, którą jako pierwszą zobaczy użytkownik korzystający z tego narzędzia.



Rysunek 2.1 Strona główna narzędzia Snail Project  
 Źródło: Opracowanie własne



Rysunek 2.2 Schemat działania narzędzia Snail po wpisaniu adresu URL badanej strony  
 Źródło: Opracowanie własne

Rysunek 2.2 [41] przedstawia uproszczony schemat działania „Snail” po wpisaniu odpowiedniego adresu strony internetowej, która ma zostać przebadana.

### **3. Dlaczego warto wybrać narzędzie Snail Project**

Narzędzie zostało wykonane z myślą o pomocy w optymalizacji stron Internetowych. Po wykonaniu testów prezentowane wyniki sugerują, jakie zmiany należy wprowadzić, aby testowana aplikacja webowa działała szybciej i bardziej wydajnie. Narzędzie podobnie jak WebPageSpeed oraz GTmetrix wykorzystuje silnik testujący Insight stworzony przez firmę Google. Snail Project generuje także wykres typu Waterfall po to, by można było się dowiedzieć, które z obiektów na stronie działają nieprawidłowo. Widać na nim także ilość zapytań do serwera, zwracany jest rozmiar pobieranej strony, a po kliknięciu odpowiedniego przycisku można zobaczyć wszystkie nagłówki odpowiedzi. Dodatkową informacją jaką jest w stanie pokazać opisywany projekt jest informacja o tym, jakie możliwości HTML i CSS wykorzystuje przeglądarka Internetowa, z której użytkownik wszedł na stronę Projektu. Snail jako jedyny potrafi wykorzystać validator konsorcjum W3C, aby przetestować stronę pod kątem błędów w kodzie HTML, XHTML, SMIL, MathML, itp. Naturalnym argumentem za tym, by wybrać właśnie to narzędzie do pracy nad wydajnością swojego serwisu webowego, jest przyjazny użytkownikowi wygląd, który jest czytelny, przejrzysty i sprawia, że czytanie oraz analiza wyników jest dla użytkownika bezproblemowa.

Snail Project skupia w sobie siłę najpopularniejszych narzędzi internetowych i udostępnia je z poziomu odwiedzonej strony Internetowej. Nie wymaga instalacji, dodatkowej implementacji kodu ani instalowania żadnych programów czy frameworków. Narzędzie to zostało zaprogramowane tak, aby każdy z testów mógł być wykonywany niezależnie od innych. Dla porównania serwis WebPagetest, działa w taki sposób, że dopiero po wykonaniu wszystkich testów i pod warunkiem zwrócenia wyników zostaną one przedstawione użytkownikowi. W przypadku gdyby wszystkie serwery Google przestały działać, narzędzie Snail w dalszym ciągu będzie w stanie przedstawić pozostałe wyniki, dzięki, którym można pracować nad poprawą wydajności swojego serwisu.

Snail powinien przede wszystkim służyć twórcom stron internetowych. To właśnie dla nich, i dla ułatwienia sprawnego zarządzania połączono skrypty różnych firm w jedno uniwersalne narzędzie. Dobre i sprawdzone rozwiązania firm takich jak Google, którzy udostępniają API do swoich skryptów bardzo dobrze się sprawdzają jako podstawa budowania nowego narzędzia. Autorska część projektu to między innymi implementacja wykorzystywanych skryptów w kodzie strony, oraz obsługa zdarzeń i odpowiednio odbieranie obiektów z wynikami analiz. Budowa szkieletu całego narzędzia tak aby wyróżniało się spośród innych, dostępnych na rynku tego typu narzędzi funkcjonalnością. Przemyślana konstrukcja poszczególnych funkcji, aby najlepiej sprawdzały się w dostarczaniu programiście danych do analizy. Odpowiednia prezentacja wyników użytkownikowi w sposób czytelny i jasny do zrozumienia i powtórzenia badania. Umożliwienie kontynuacji prac nad projektem, do tego celu wybrano język PHP a nie np. Java, która jest komplikowana i uruchamiana na maszynie wirtualnej, przez co trudniej „na bieżąco” wprowadzać zmiany w projekcie, ponieważ jest to mniej efektywne. Ostatnim etapem było wdrożenie projektu na zewnętrzny serwer plików, tak aby był on powszechnie dostępny w internecie.

Jak wspomniano wcześniej sukcesem tego narzędzia może być wszechstronność (inne narzędzia nie są w stanie jednocześnie badać wydajności serwisu, poprawności kodu strony oraz testować użyteczności przeglądarki użytkownika).

#### **4. Narzędzie Snail Project w porównaniu z innymi**

Aby móc ocenić wydajność serwisu webowego nie wystarczy być świadomym, jakie techniczne zmiany można wprowadzić w serwisie Internetowym. Nie można do oceny każdego projektu podejść w dokładnie taki sam sposób. Na początku próby określenia wydajności właściciel serwisu webowego, lub programista mający za zadania dokonać optymalizacji powinien postawić sobie kilka znaczących pytań: Co właściwie rozumiem poprzez wydajność serwisu ? Co dokładnie chciałbym osiągnąć optymalizując serwis ? Gdzie najwięcej czasu traci użytkownik oczekując na interakcję ze stroną www ? Najważniejsze pytanie powinno brzmieć: Jaki jest cel poprawy wydajności ?

Dla niektórych osób najważniejszym osiągnięciem i czynnikiem optymalizacyjnym jest skrócenie czasu ładowania się strony Internetowej. Inni przywiążają największą wagę do poprawy „jakości” kodu tak, aby był semantyczny.

„Semantyka - dział semiotyki zajmujący się badaniem związków, jakie zachodzą między wyrażeniami języka a przedmiotami, do których się one odnoszą” – Słownik Języka Polskiego

Powyższy cytat odnosi się raczej do języka polskiego aczkolwiek kod pisany przez programistów również jest określany poprawnym czy też semantycznym jeśli jego odniesienia, wywołania, funkcje, klasy, metody są napisane prawidłowo, zwracając oczekiwane wyniki i są interpretowane bezbłędnie przez przeglądarki Internetowe czy serwery.

Prócz czasu, który jest dominującą skalą oceny wydajności stron www przez użytkowników sieci, można poddać optymalizacji wydajności praktycznie każdą część aplikacji. Dlatego warto sobie odpowiedzieć na zadane wcześniej pytanie „Jaki jest cel poprawy wydajności ?”. Celem może być zwiększena wydajność serwera, przesyłania danych, wykorzystanie nowszych technologii sieciowych, np. zmiana protokołów HTTP, przykładowo zmiana serwera z Apache na Nginx, poprawna validacja strony, SEO itp.

„Gdzie najwięcej czasu traci użytkownik oczekując na interakcję ze stroną www ?” Odpowiedzi może być tak wiele jak celów, które sobie stawia osoba chcącą zoptymalizować daną aplikację. Czas odpowiedzi z serwera może być bardzo długi. Przesyłanie dokumentu HTML może stanowić ponad 20% czasu udzielania odpowiedzi. Skrypty na stronie blokują wczytywanie się pozostałych elementów, zbyt długie wczytywanie się obrazów itp.

Jak widać wyznaczenie celów działań optymalizacyjnych nie jest jednoznaczne dla każdej osoby, jest wręcz subiektywnym zdaniem, które prowadzi do określenia pewnych działań, które mają osiągnąć zamierzony rezultat. Być może nie jest to łatwa w interpretacji definicja, jednak wszelkie wątpliwości zostaną rozwiane poniżej.

W odpowiedzi na sformułowane wyżej zapytania powstała aplikacja webowa o nazwie „Snail Project – ang. Projekt Slimak” Jest to aplikacja webowa, która testuje czy w innych aplikacjach, stronach Internetowych wdrożone zostały najważniejsze zasady optymalizacyjne. Wynik jaki otrzyma użytkownik na pewno pomoże w dokonaniu wyboru celów optymalizacji, skonkretyzuje do czego przyłożyć największą wagę podczas czynności polepszających działanie innego serwisu oraz oceni aktualny stan danego serwisu czy aplikacji.

## Test porównujący narzędzia

W sieci Internet, która jest pełna różnych ciekawych projektów, istnieją rozwiązania podobne do Projektu Slimak. Różnią się od siebie szybkością działania, niektóre wykonują w ekspresowym tempie testy a inne dłuższy czas. Nie wszystkie serwisy zwracają ten sam wynik, a co gorsza różnice między nimi są znaczne i bardzo często niezgodne z prawdą.

Aby lepiej poznać różnice między serwisami, które wykonują pomiary wydajnościowe to właśnie one zostały poddane testowi. Serwisy badały różne czynniki dlatego specjalnie przygotowano test tak żeby w miarę możliwości każdy z nich miał szansę przejść go pozytywnie.

Zadaniem każdego serwisu było:

- Pozytywne wykonanie testu
- Zwrócenie wyniku, który zawiera najbliższy prawdziwemu rozmiarowi/wagi strony
- Podanie ilości zapytań do strony

Na potrzeby tego eksperymentu przygotowano specjalną stronę Internetową, która będzie służyła jako materiał do badań i późniejszej oceny. Strona HTML zawiera w sobie elementy drzewa DOM, których atrybutami są 4 obrazy o wymiarach odpowiednio: 2074x1383 piksele 1954x1301 pikseli, 1954x1301 pikseli, 1815x1031 pikseli o łącznej wadze 1,71 megabajta. Sam plik o rozszerzeniu HTML zajmuje powierzchnię o wadze 28 kilobajtów. Szablon HTML wczytuje również bibliotekę Bootstrap oraz wszystkie jej style o łącznej wadze 169 kilobajtów.

Plik ze skryptami JavaScript dla frameworka Bootstrap 32 kilobajty. 3 foldery, w celu uporządkowania plików tworzących stronę. Przewijany slider ze zdjęciami o łącznej wadze 1.12 megabajta. Dodatkowo na stronie zostały celowo wygenerowane błędy, aby sprawdzić jak wpływają one na wyniki testów. Błędy to odwołanie do pliku favicon.ico, który nie istnieje. Odwołanie w kodzie HTML do pliku .CSS, który także nie istnieje.

Rozmiar całej strony testowej na dysku twardym to **3,23 megabajta**. (Rozmiar został podany przez explorator systemu Microsoft Windows 8)

Strona została następnie opublikowana w Internecie tak, aby można było przeprowadzić eksperiment w łatwy sposób posługując się jedynie docelowym adresem URL.

### PageSpeed Insights

Opis: Jest to narzędzie do testowania wydajności stron Internetowych, którego autorem jest firma Google [14]. Test jest przeprowadzany i oceniany zarówno pod kątem telefonów komórkowych jak i komputerów stacjonarnych (włączając w to laptopy). Strona została też oceniona w skali 0-100 jednak bez określenia konkretnej jednostki jaką można by było się posługiwać.

Wyniki są ciekawe, dobrze opisane lecz to wszystko co pokazuje narzędzie Google.

Wynik: Udało się wykonać test, jednak zwracane wyniki sprowadzają się jedynie do sugerowania użytkownikowi jakie poprawki może on zastosować w swoim projekcie, aby był wydajniejszy.

#### [GTmetrix](#)

Opis: GTmetrix [15] jest projektem rozwijanym przez wyspecjalizowanych programistów w firmie „Gossamer Threads” w Vancouver. Firma posiada ponad 19 lat doświadczenia w pracy z technologiami webowymi. Serwis ten nie posiada swojego własnego systemu oceny strony, korzysta z PageSpeed Insight oraz narzędzia Yslow do prezentowania wyników użytkownikowi. Rezultaty z obu dostępne są w osobnych zakładkach i trzeba się między nimi przełączać, aby dostrzec różnice w ocenie. Trzecią zakładką jest linia czasu, czyli wykres czasu ładowania się kolejnych elementów w zależności od zapytania do serwera o konkretny element. Testy przeprowadzane są przez serwer w Vancouver w Kanadzie oraz przez agenta Internetowego Mozilla.

Wynik: Test został wykonany pozytywnie. Waga strony według serwisu to 2.90 megabajta, a ilość zapytań do serwera to dziesięć.

#### [Load Impact](#)

Opis: Serwis zajmujący się odpłatną optymalizacją stron i serwisów Internetowych. Bez rejestracji i bez dodatkowych płatności można jednak wykonać test wydajności strony www[16]. Test jest wykonywany przez serwery w różnych miejscach na całym świecie. Rezultat zawiera ilość zapytań do serwera, średni czas ładowania się strony (dodatkowy wykres przedstawia czas ładowania w zależności od serwera). Na stronie znajdują się także wykresy kołowe przedstawiające, jaką część czasu ładowania się przedstawiają poszczególne grupy elementów na stronie np. skrypty, obrazki, HTML oraz style CSS.

Wynik: Test został wykonany pozytywnie. Wynik nie pokazuje jaki jest ciężar strony. Ilość zapytań do serwera to dziesięć.

#### [WebPagetest](#)

Opis: WebPagetest [17] jest projektem open source, który jest przede wszystkim tworzony i wspierany przez Google w ramach poprawy szybkości obiektów w Internecie. WebPagetest jest narzędziem, który został pierwotnie opracowany przez AOL do stosowania wewnętrz firmy. Platforma jest w stanie aktywnego rozwoju na GitHub a okresowo bywa pakowany do plików ZIP i udostępniany na oficjalnej stronie dla użytkowników. Wersja online jest prowadzona przez Fundację WPO. Firmy wspierają projekt poprzez udostępnianie hostingu, pomoc w ulepszaniu narzędzi testujących itp. Każdy programista może zwrócić się z prośbą do serwisu o udostępnienie specjalnego klucza dla programistów „API key”, za pomocą którego można wykorzystać WebPagetest do implementowania we własnej aplikacji webowej. Wyniki są bardzo wnikliwe, zawierają dużo wykresów i analiz. Jedynym mankamentem jest User eXperience. Zakładki, służące do przechodzenia pomiędzy wynikami i sam sposób przedstawienia rezultatów nie jest estetyczny i nie przyciąga wzroku. Przez to dużo ważnych szczegółów można pominąć.

Wynik. Test został wykonany pozytywnie. Waga strony 2.70 megabajta. Liczba zapytań do serwera wynosi dziesięć.

## [WebSiteOptimization](#)

Opis: Narzędzie [18] do analizowania stron www, którego celem jest sugerowania użytkownikowi co mógłby poprawić, aby jego strona wczytywała się szybciej. Ciekawą informacją jest podanie zależności pomiędzy różną prędkością łącza Internetowego a czasem wczytywania się strony. Ponieważ eksperyment zakładał podstawowe testy, nie sprawdzono poprawności innych wyników niż zakładane.

Wynik: Serwis uważa, że test został wykonany prawidłowo, jednak wyniki całkowicie odbiegają od zakładanych. Według serwisu waga pobranych elementów wynosi 1,27 megabajta. Liczba zapytań do serwera 7.

## [Neustar](#)

Opis: Serwis [19] oferuje test przy pomocy narzędzia, które wykonuje test wydajnościowy innego serwisu Internetowego a następnie przedstawia wyniki w ładny graficznie sposób. Test jest wykonywany także przez serwery w różnych miastach np. Dublin, Singapur, Washington oraz San Francisco. Rezultaty, jakie zostały wygenerowane najbardziej odbiegają od zakładanych.

Wynik: Test został wykonany. Bez względu na położenie serwera test wygenerował te same wyniki dla wagi strony – 60,3 kilobajty ! Pokazane zostały również tylko obiekty strony, na które serwer odpowiedział statusem 200 i było ich 7.

## [Pingdom](#)

Opis: Pingdom [20] jest narzędziem, które może być wykorzystywane przez użytkowników w celu ustalenia czasu ładowania strony Internetowej poprzez generowanie licznych raportów, takich jak rozmiar strony, pamięci podręcznej przeglądarki, klas wydajności i wiele innych. Pozwala także na śledzenie historii wydajności i przeprowadzenia testów z różnych miejsc. Narzędzie to cechuje się czytelnością, przejrzystością oraz szybkością działania.

Wynik: Test został wykonany pozytywnie. Waga strony to 2.90 megabajta. Liczba zapytań do serwera to dziesięć.

## [Snail Project](#)

Opis Projektu, jego zasada działania i części składowe zostały przedstawione w poprzednich rozdziałach pracy.

Wynik: Test został wykonany w całości. Waga całej strony Internetowej wynosi 3,02 megabajta. Liczba połączeń z serwerem wynosi dziesięć.

## [Podsumowanie wykonanych testów](#)

Ocena wykonania testu w całości była oceniana w sposób poniekąd binarny, to znaczy albo test się wykonał i pokazał wynik albo nie można było wykonać testów i żaden wynik ani informacja na temat testowanej strony nie został zaprezentowany. Wszystkie narzędzia zwracały mniej lub bardziej wiarygodne wyniki dlatego każdemu należy się plus za poprawną implementację skryptów testujących.

Jakość skryptów pozostawia jednak wiele do życzenia. Najbliższym prawdy okazało się narzędzie „Snail Project”, które działało na niezależnym serwerze. Większość opisanych

wcześniej narzędzi korzystało z silnika testującego firmy Google, jednak na wyniki mogło mieć wpływ wiele różnych czynników.

Niektóre z narzędzi np. Neustar pokazywały tylko te zapytania, które otrzymały pozytywną odpowiedź z serwera, mimo wszystko nie jest to prawidłowe prezentowanie wyników, ponieważ użytkownik nie wie, wtedy jakich dokonać zmian w kodzie, aby wszystkie zapytania posiadały status odpowiedzi 200.

## **Wnioski**

Różne firmy tworzące narzędzia do wykonywania testów i oceny aplikacji webowych, na swój sposób wykorzystują już istniejące skrypty bądź swoje autorskie rozwiązania, jednak wszystkie pokazują rozbieżności w wynikach działa tych narzędzi. Celem eksperimentu nie było sprawdzanie poprawności zaimplementowanych skryptów czy też sposobu działania, lecz szybkie zweryfikowanie czy projekt, który powstał jako część pracy magisterskiej może być z powodzeniem wykorzystywany przez programistów oraz wszystkich ludzi chcących przetestować swoje strony Internetowe, aplikacje webowe czy po prostu testować strony już istniejące w sieci celem wyciągnięcia wniosków czy sugerowania ich autorom zmian jakie można wprowadzić. Rezultaty jednoznacznie wskazały, że Snail Project nie odstaje od reszty łatwo dostępnych w internecie narzędzi, na dodatek w tym prostym eksperymencie wypadł najlepiej. Pozostałe badania opisane w dalszej części pracy zostały wykonane z wykorzystaniem właśnie tego narzędzia.

## 5. Badania 10 najpopularniejszych stron Internetowych w Polsce

Poniżej przedstawiono analizę dziesięciu najpopularniejszych stron Internetowych w Polsce według rankingu Alexa [21] na dzień 1 stycznia 2015 roku.

Alexa Internet jest amerykańską spółką kontrolowaną przez grupę Amazon. Prowadzi ona witrynę o nazwie Alexa, która dostarcza informacje na temat generowanego ruchu do innych stron Internetowych. Spółka została założona w 1996 roku, natomiast Amazon przejął ją w roku 1999. Wyszukiwarka Alexa zaopatrzona jest w robota Internetowego, który indeksuje ponad 4,5 miliarda stron. Posiada także katalog stron WWW.

Badania przeprowadzono przy pomocy rozwiązania o nazwie „Snail Project”. Dokładny opis działania Projektu oraz jego skryptów został rozwinięty w rozdziale pt. „Snail Project”.

Analiza najpopularniejszych stron w Polsce ma na celu sprawdzenie czy najnowsze dostępne technologie są implementowane we współczesnych serwisach webowych.

**Google.pl** - Wyszukiwarka stron WWW, grafiki, grup dyskusyjnych, katalog tworzony na bazie ODP i wzbogacany przy użyciu własnej technologii Google. Polska domena wyszukiwarki Google.com

Strona główna Google.pl zawiera jeden obrazek, który jest zmieniany w zależności od okazji (Dzień Kobiet, Dzień Dziecka, Święta Narodowe itd.), standardowo jest to logo firmy Google. Prócz tego na stronie znajduje się pasek wyszukiwania. Jest to bardzo mała i dobrze zoptymalizowana strona, ponieważ służy ona jedynie do wpisania odpowiedniej frazy wyszukiwania. Poniższy obrazek przedstawia stronę główną wyszukiwarki Google dla Polski.



Rysunek 5.1 Strona główna google.pl  
Źródło: Opracowanie własne

Biorąc pod uwagę fakt, że firma z Mountain View jest liderem w kwestii wyszukiwarek Internetowych oraz tworzonych przez siebie narzędzi, zarówno programistycznych jak i np.

przeglądark Internetowych, to nawet główna strona wyszukiwarki nie osiągnęła maksymalnej liczby punktów w teście. 97% jak na tak potężną korporację to mimo wszystko zaskoczenie. Wynik działania skryptu PageSpeed Insights nie zwrócił długiej listy rzeczy, jakie należy poprawić, aby strona wczytywała się szybciej, jedyne sugestie to zmniejszenie rozmiaru pliku z kodem JavaScript oraz nadanie priorytetów widocznej zawartości strony. Google używa własnego serwera webowego GWS – Google Web Server [22], a czas odpowiedzi na żądanie przesłania strony wyniósł ~4,5 skeundy. Liczba zapytań do serwera wyniosła 12. Można zmniejszyć liczbę zapytań poprzez zastosowanie CSS Sprites oraz połączenie skryptów w jeden plik lub zagnieżdżenie ich w kodzie strony jeśli nie są zbyt duże. Validator kodu HTML znalazł 32 błędy między innymi „Line 10, Column 6071: Duplicate ID locale”, większość dotyczy złego wykorzystania stylów CSS w kodzie HTML. Jednak tak duża korporacja powinna tworzyć kod zgodny z założeniami World Wide Web Consortium i nie pozwalać sobie na wytykanie takich błędów jak „Legacy encoding iso-8859-2 used. Documents should use UTF-8.”.

**Facebook.com** – Jest to portal społecznościowy umożliwiający kontakt ludziom z całego świata, wymianę zdjęciami, muzyką oraz filmami wideo.

Poniższy obraz przedstawia widok strony głównej portalu Facebook.



Rysunek 5.2 Strona główna serwisu Facebook.com  
 Źródło: Opracowanie własne

Wynikiem analizy serwisu Internetowego Facebook jest osiągnięcie 89 procent zoptymalizowania strony głównej, na którą trafia użytkownik podczas pierwszego renderowania tej strony. Całkowity czas pobierania dokumentu HTML wyniósł około 3 sekund, co jest wynikiem lepszym niż w przypadku witryny Google.pl, mimo większej liczby elementów na stronie. Dwadzieścia siedem zapytań do serwera jak dla stron, której zadaniem jest umożliwienie użytkownikowi rejestrację w serwisie lub zalogowanie się nie jest

najlepszym osiągnięciem. Liczbę zapytań do serwera można zmniejszyć o siedem, łącząc obrazki i tworząc CSS Sprites, przez co nie trzeba wykonywać osobnego zapytania do każdej grafiki. Waga całej odebranej strony, która zostanie wyrenderowana poprzez agenta webowego wynosi ~1,16 megabajta, co jest dosyć dobrym wynikiem. Można go jednak poprawić stosując się do reguły minimalizacji rozmiaru plików *.js*. Wynik działania skryptu PageSpeed zwrócił wyniki, z których można wyczytać następującą regułę ważności poprawy elementów na stronie „10 razy ważniejsze i większe korzyści przyniesie wyeliminowanie blokowania skryptów niż zmniejszenie wagi plików ze skryptami”.

Importance	Type
0	Avoid landing page redirects
0	Enable compression
0	Leverage browser caching
0	Reduce server response time
0	Minify CSS
0	Minify HTML
0.2705999999999999	Minify JavaScript
10	Eliminate render-blocking JavaScript and CSS in above-the-fold content
0	Optimize images
0	Prioritize visible content

Rysunek 5.3 Błędy serwisu Facebook.com  
 Źródło: Opracowanie własne

Analiza kodu strony pod kątem błędów, które prócz zwiększenia szybkości interpretowania, mogą wpływać na wydajność działania, zwróciła 40 błędów. Najczęściej powtarzającymi się błędami jest błędne użycie stylów CSS lub niewykorzystywanie wszystkich wczytanych arkuszy stylów.

Facebook posiada ponad miliard użytkowników na całym świecie, którzy wymieniają się zdjęciami, publikują posty i rozmawiają na zintegrowanym czacie. Całkowity ostateczny wynik jak na tak wielki serwis jest dość dobrym osiągnięciem, jednak błędy jakie zostały wytknięte poprzez narzędzie Snail można w stosunkowo łatwy sposób poprawić i przyniosą

wymierne korzyści zarówno dla użytkowników (szybszym czasem działania i wydajnością) jak i właścicieli (większa liczba zadowolonych użytkowników).

**Allegro.pl** – Największy i najpopularniejszy Polski serwis aukcyjny, którego popularność powinna motywować do utrzymywania swojej strony optymalnej i wydajnej.

The screenshot shows the Allegro.pl homepage with the following key features:

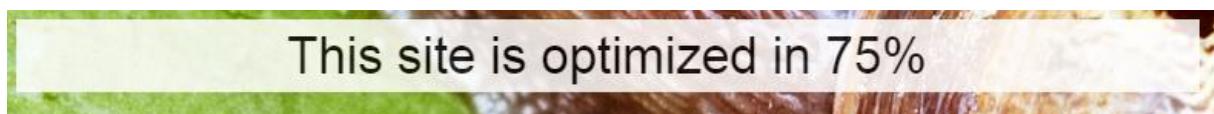
- Header:** Includes a search bar ("czego szukasz?"), a dropdown menu ("wszystkie działy"), a magnifying glass icon, a shopping cart icon ("koszyk jest pusty"), and navigation links for "wystaw przedmiot", "moje allegro", "zakłóż konto", and "zaloguj".
- Main Banner:** Promotes a mobile app offer: "Tylko w Aplikacji Mobilnej" (Only in the mobile app), "Kupuj taniej o 10 zł" (Buy cheaper by 10 zł), and "Za zakupy za min. 50 zł zapłacisz 10 zł mniej!" (For purchases over min. 50 zł, pay 10 zł less!). It also links to "Szczegóły na kupony.allegro.pl".
- Section: wyjątkowe okazje dla Ciebie** (Special offers for you): Shows a grid of products with discounts:
  - Wkładki do Gillette mach 3, 12 sztuk Ostrza Nożyki: -74% (99,00 zł → 25,00 zł)
  - Latarka LED Zoom CREE XM-L T6 + akcesoria: -46% (129,00 zł → 70,00 zł)
  - Łóżko dla dzieci Auto Car 160x80 materac, natka: -22% (450,00 zł → 349,00 zł)
  - Zegarek z ukrytą kamerą Full HD 1080P + 8GB: -38% (369,00 zł → 190,00 zł)
  - Dwustronna pościel 200x220 + przesłonka: -46% (169,00 zł → 59,00 zł)
  - Orbiti trening: (459,00 zł)
- Offer 1:** GEOX shoes with a 60% discount (RABAT -60% "wysyka gratis").
  - Image: Two black and red high-top sneakers.
  - Text: "GEOX"
  - Text: "SPRAWDZ OFERTĘ" (Check offer)
  - Text: "Przy zamówieniu poniżej 200 zł"
- Offer 2:** SONY SRS-BTV5 speaker.
  - Image: A yellow Sony SRS-BTV5 speaker.
  - Text: "Bezprzewodowy głośnik SONY SRS-BTV5"
  - Text: "249 zł" (Original price) and "99 zł" (Offer price)
  - Text: "Sony"
  - Text: "Technologia Bluetooth", "Tryb głośnomówiący", "Technologia NFC"
- Section: ostatnio przeglądane** (Recently viewed): Shows a placeholder message: "Jak to możliwe, że tu jest pusto?" (How is it possible that this is empty?) and "Poszperaj w ofertach Allegro".
- Charity Section:** Encourages support for the "Instytutu Matki Polki w Łodzi" (Institute of the Polish Mother in Łódź). It features four teddy bears and the text: "Wspieraj Klinikę Rehabilitacji Instytutu Matki Polki w Łodzi".
- Footer:** Includes social media links for PAYBACK, Facebook, Twitter, YouTube, and Allegro TV, as well as a link to the "charityteam allegro".
- Page Bottom:** Shows the URL "http://allegro.pl/" and a note about cookie usage.

Rysunek 5.4 Testowana strona serwisu Allegro.pl  
Źródło: Opracowanie własne

Powyższy obraz jest odzwierciedleniem strony głównej Allegro. Niestety bardzo często zmieniający się *content* strony powoduje, że okresowe wykonywanie analizy przy pomocy narzędzia Snail Project może przynosić różne wyniki. Wynika to przede wszystkim ze zmieniającej się liczby elementów na stronie takich jak grafika, opisy, promocje itp.

Analiza wykonana w celu porównania serwisu razem z 9 pozostałymi najpopularniejszymi serwisami webowymi w Polsce, została dokonana i opisana w ciągu jednego dnia tak, aby uniknąć rozbieżności wyników, które mogłyby wystąpić podczas zmiany zawartości obiektów na witrynie.

Wskaźnik zoptymalizowania narzędzia Snail, pokazał 75% zoptymalizowania pierwszej strony serwisu Allegro.



Rysunek 5.5 Ocena zoptymalizowania testowanego serwisu przez narzędzie Snail  
Źródło: Opracowanie własne

Niestety wynik ten jest bardzo przeciętny, szczególnie, że oprócz oceny ogólnej Snail sugeruje poprawić wiele elementów wpływających na szybkość i wydajność sprawdzanej strony.

Importance	Type
0	Avoid landing page redirects
1.7341	Enable compression
3.9907357804232806	Leverage browser caching
0	Reduce server response time
0.2101	Minify CSS
0.2957	Minify HTML
0.4638000000000016	Minify JavaScript
10	Eliminate render-blocking JavaScript and CSS in above-the-fold content
16.2896	Optimize images
0	Prioritize visible content

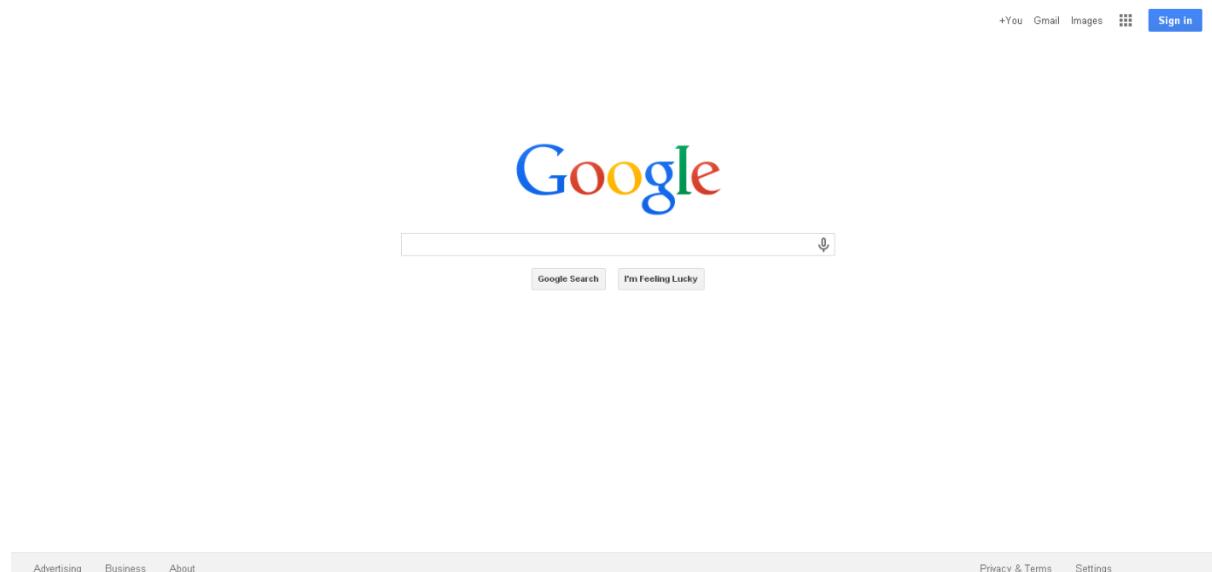
Rysunek 5.6 Błędy serwisu Allegro.pl  
 Źródło: Opracowanie własne

Jak widać na powyższej liście reguł, które należy poprawić najważniejszym elementem jest optymalizacja obrazów na stronie głównej. Jest to ponad szesnaście razy ważniejsze niż odblokowanie kompresji przesyłanych plików w ustawieniach serwera nginx, z którego Allegro korzysta (jeśli serwer korzysta z kompresji danych, jego ustawienia mogą nie być optymalne, lub nie cała treść jest ulega kompresji). Następnym co do ważności elementem wymagającym poprawy jest wyeliminowanie blokujących renderowanie strony skryptów JavaScript oraz kaskadowych arkuszy stylów. Pozostałe cechy, jakie należy poprawić wpłyną na poprawienie ogólnej oceny optymalizacji testowanej strony. Dodatkowo gdyby programiści pracujący nad kodem serwisu zastosowali wymienione reguły do wszystkich podstron serwisu, mogliby to znaczco wpływać na szybkość jego działania, co przekłada się na zadowolenie użytkowników i być może większe zyski. 78 zapytań do serwera to spory wynik, który także wpływa znaczco na szybkość pobierania całej strony. Rezultat ten można poprawić, tak jak w przypadku serwisu Facebook, łącząc grafiki poprzez stworzenie CSS Sprites, a mniej rozbudowane skrypty umieszczając w ciele dokumentu HTML lub łącząc je w mniejszej ilości plików.

Korzystanie z narzędzia Snail dostarcza nie tylko informacji takich jak liczba elementów drzewa DOM danej witryny, liczba zapytań i odpowiedzi serwera, ale umożliwia też

przeglądanie nagłówków jakimi wymieniają się przeglądarka Internetowa i serwer testowanej witryny. Można dzięki nim odczytać takie informacje jak wykorzystywany serwer, używaną wersję protokołu HTTP, wykorzystywaną metodę kompresji czy odczytać pobrany plik w języku znaczników HTML.

**Google.com** - Najpopularniejsza wyszukiwarka stron WWW oraz treści w Internecie.



Rysunek 5.7 Strona główna serwisu Google.com  
Źródło: Opracowanie własne

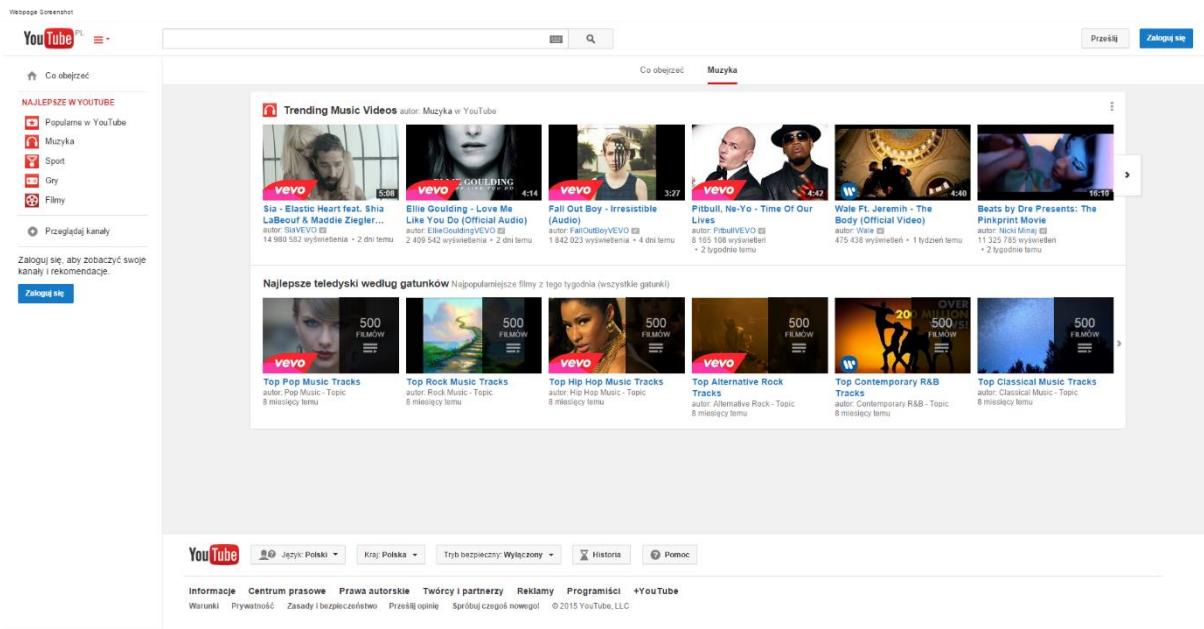
Domena Google.com została oceniona na 90% zoptymalizowania (natomiast wcześniejszy wynik dla Google.pl wskazywał na 97% optymalizacji). Czas ładowania jest również dłuższy w porównaniu z polską domeną i wynosi 6,2 sekundy.

Różnica wynikająca z wyników po przeprowadzeniu analizy Google.pl oraz Google.com wynika przede wszystkim z przekierowania jakie jest wykorzystywane. Wpisując w pasku adresu przeglądarki adres [www.google.com](http://www.google.com) przed załadowaniem widocznej części strony działa skrypt sprawdzający położenie geograficzne użytkownika a następnie przekierowuje do wyszukiwarki w odpowiednim języku. Właśnie dlatego domena z końcówką .com posiada więcej przekierowań i ma słabszy wynik niż bezpośrednie korzystanie z wyszukiwarki w języku polskim z domeną w Polsce.

Liczba błędów jest również zbliżona do polskiej wersji strony z wyszukiwarką Google bo jest to 30 błędów. Także i tym razem pierwszym wymienionym problemem jest kodowanie strony, które posiada następujący opis „Using windows-1252 instead of the declared encoding iso-8859-1.”

Wynik działania Snail w pierwszej kolejności sugeruje zmniejszenie liczby przekierowań w celu szybszego pobierania się wywoływanej strony Internetowej. Narzędzie to proponuje także priorytetyzację widocznej części strony. Dzieje się tak, dlatego że w pierwszej kolejności po wywołaniu adresu i pobieraniu zawartości strony, nic się nie wyświetla użytkownikowi, tylko działa skrypt lokalizujący użytkownika i następuje przekierowanie na odpowiednią wersję językową wyszukiwarki od Google.

**Youtube.com** – Serwis Internetowy umożliwiający publikowanie własnych filmów oraz prostą obróbkę.



Rysunek 5.8 Widok testowanej strony serwisu youtube.com

Źródło: Opracowanie własne

Powyższy obraz przedstawia stronę Youtube.com, której najważniejszą częścią jest sugerowany materiał, który użytkownik może oglądać bez potrzeby rejestracji w serwisie. Liczba elementów, oraz ich waga zmienia się i może być różna dla każdego użytkownika jak również zależy od daty wykonania analizy.

Poniżej znajduje się grafika, na której widać wynik działania skryptu zaimplementowanego w Snail Project, którego zadaniem jest analiza danego serwisu webowego oraz wyświetlenie wyniku wraz z wagą z jaką należy poprawić konkretne elementy oraz ich opisem.

Importance	Type
0	Avoid landing page redirects
0	Enable compression
16.906250000000007	Leverage browser caching
0	Reduce server response time
0	Minify CSS
0.3122000000000003	Minify HTML
0.3721000000000004	Minify JavaScript
10	Eliminate render-blocking JavaScript and CSS in above-the-fold content
6.1899	Optimize images
0	Prioritize visible content

Rysunek 5.9 Błędy serwisu youtube.com  
 Źródło: Opracowanie własne

Na grafice widać, jak ważne jest odpowiednie wykorzystanie pamięci podręcznej przeglądarki, umożliwia to późniejsze odtwarzanie elementów z pamięci agenta Internetowego bez potrzeby ponownego pobierania danego elementu.

Drugim najważniejszym elementem wpływającym na szybkość pobierania strony Youtube.com jest wyeliminowanie blokujących renderowanie widoku strony skryptów JavaScript oraz stylów CSS (Charakterystyka wyżej wymienionych reguł i ich znaczenie zostało opisane w kolejnym rozdziale niniejszej pracy).

Następną rzeczą wymagającą uwagi programisty jest optymalizacja grafik na stronie. Optymalizacja ma na celu zarówno odpowiednie dopasowanie rozmiaru obrazów oraz zmniejszenie jakości (konwersja stratna) lub zmianę formatu celem zmniejszenia wagi pliku.

Aby pobrać całą stronę z serwera wykonanych zostało 49 zapytań do serwera. Naturalnie ich liczbę można zmniejszyć, tak jak w przypadku serwisu Facebook czy Allegro, można zastosować CSS Sprites oraz redukcję ilości skryptów poprzez zastosowanie wcześniej opisanych wskazówek.

Pozostałe elementy wymagające interwencji są także ważne jednak przyniosą mniej odczuwalną poprawę szybkości w porównaniu z wykonaniem optymalizacji wcześniej opisanych kroków.

Prócz wymienionych reguł wydajności również poprawa 1001 błędów w kodzie strony przyniesie wydajniejszą pracę serwisu. Większość z nich jest generowana poprzez działanie skryptów generujących kod HTML, który aktualnie nie jest używany. Gdyby zoptymalizować działanie tych skryptów, mniejsza ilość kodu byłaby interpretowana przez przeglądarkę i serwis mógłby działać wydajniej.

**Onet.pl i WP.pl** - To dwa konkurencyjne serwisy informacyjne, finansowe i rozrywkowe, zawierają katalogi stron Internetowych, pocztę e-mail i strony WWW, czat, serwisy społecznościowe, gry online, platformę e-commerce, fora i blogi. Wiadomości z kraju i ze świata, ogłoszenia, pasaż handlowy, wyszukiwarki i katalog stron.

Poniżej znajdują się obrazki przedstawiające wygląd testowanych serwisów.

The screenshot shows the homepage of Onet.pl. At the top, there's a navigation bar with the Onet logo, search fields, and various links like Sympatia, Allegro, Zumi, Gry online, VOD, Sympatia+, Dysk, and Poczta. Below the header, there's a large image of a group of people in riot gear. A main headline reads: "Dwa szturmy francuskiej policji, zamachowcy zostali zabici, są ofiary wśród zakładników". Below the main article, there are three smaller news cards: one about Magdalena Ogórek being a candidate for president, another about President of Krakow, and a third about a film. To the right, there's a weather forecast for Warsaw (4°C, 10°C) and a section for "WIADOMOŚCI" with various news items. At the bottom, there's another news card about Józef Oleksy's death, followed by sections for "SPORT" and "BIZNES".

This screenshot shows a different layout of the Onet.pl homepage. It features a prominent "MÓJ ONET" button at the top left. Below it, there are several news cards: one about Józef Oleksy's death, another about Wałęsa, and one about Kaczyński. There are also sections for "POLECAMY" (recommended) and "POLECAMY" (recommended). The overall design is more minimalist compared to the first screenshot.

Rysunek 5.10 Widok testowanej strony serwisu Onet.pl

Źródło: Opracowanie własne

**PILNE** Terror we Francji. Szturmy policji na pozycje napastników. Bracia Kouachi nie żyją. Ofiary wśród zakładników

Kopacz: wydarzenia we Francji nie zagrażają Polakom w kraju

Wenderlich dla WP: Oleksy nigdy nie mówił o śmierci

Ostatnia wola Józefa Oleksego

Nad Polskę nadciąga orkan Feliks. Ostrzeżenie dla 11 województw

Odkryto największe podziemne miasto na świecie

Aktorka w ogniu krytyki

Eksperci dla WP: uroda nie zapewni jej wygranej

Nigdzie się nie ukryjesz. Gen. Polko dla WP: jesteśmy śledzeni na każdym kroku

Kolejny hit 2015? W końcu mamy zwieństwo!

Ten Polak był największym siłaczem świata

Pendolino śmiertelnie potrąciło 26-latkę

Fatalne wiadomości. To koniec sportów zimowych, jakie znamy?

Pochwalił się "zabawką" za ponad 6 mln złotych

Zaskakujące działanie gumy do żucia. Komu może zaszkodzić?

Kasia Zielińska urodziła

Ile jest masła w maśle? Nie daj się nabrac

Rysunek 5.11 Widok testowanej strony serwisu Wirtualna Polska  
 Źródło: Opracowanie własne

Prócz zbliżonego wyglądu i samych funkcjonalności serwisów, technicznie serwisy różni praktycznie wszystko. Wynik analizy obu stron jako lepiej zoptymalizowany wskazuje firmę Onet.pl, która osiągnęła 81% zoptymalizowania, a Wirtualna Polska 74%. Samo zoptymalizowanie serwisów nie wygląda najgorzej jednak obie firmy powinny zmniejszyć ilość zapytań do serwera. W przypadku Onetu są to 153 zapytania natomiast WP to aż 253

zapytania. Wirtualna Polska posiada również wiele błędów w kodzie swojej witryny (263 błędy), które również wpływają na wydajność serwisu. Gdyby sporządzić pojedynek na poziom optymalizacji obu Serwisów to Onet miałby czterokrotnie lepszy wynik jeśli chodzi o czas pobierania strony. Około 121 sekund dla onetu i 468 sekund dla WP. Obie strony są bardzo obszerne jednak na przykładzie Onetu (pomijając błędy jakie wskazało narzędzie Snail), można udowodnić, że da się zoptymalizować lepiej działanie swojego serwisu i Wirtualna Polska powinna wziąć dobry przykład z konkurenta.

Poniżej znajduje się także porównanie błędów obu Serwisów

Importance	Type
0	Avoid landing page redirects
0.3109	Enable compression
1.0863053902116402	Leverage browser caching
0	Reduce server response time
0.35340000000000005	Minify CSS
0.21330000000000002	Minify HTML
0.26920000000000005	Minify JavaScript
12	Eliminate render-blocking JavaScript and CSS in above-the-fold content
4.414000000000001	Optimize images
0	Prioritize visible content

Rysunek 5.12 Błędy serwisu onet.pl  
 Źródło: Opracowanie własne

Importance	Type
0	Avoid landing page redirects
0.3343	Enable compression
3.485966435185185	Leverage browser caching
0	Reduce server response time
0.5287000000000001	Minify CSS
0.2860999999999997	Minify HTML
1.2086999999999999	Minify JavaScript
44	Eliminate render-blocking JavaScript and CSS in above-the-fold content
21.5241	Optimize images
4	Prioritize visible content

Rysunek 5.13 Błędy serwisu wp.pl  
 Źródło: Opracowanie własne

Gdyby oba serwisy zastosowały się do sugerowanych reguł, a ich wyniki optymalizacji oscylowały wokół 90% można by mówić o bardzo wydajnych serwisach webowych. Niestety jednak powyższe wyniki obrazują, że programiści powinni reagować na to by ich serwisy działały optymalnie, przez co były lepiej odbierane przez użytkowników, którzy częściej odwiedzają takie strony, gdzie czas potrzebny od wykonania pierwszego żądania do pobrania i wyrenderowania witryny nie przekracza kilku sekund.

**Gazeta.pl** – To serwis tożsamy z dwoma powyższymi. Również zawiera najnowsze informacje, wyszukiwarkę, katalog stron WWW, pozwala na założenie konta e-mail i korzystanie forum. Posiada także serwisy tematyczne i lokalne w 20 miastach w Polsce. Jest to Internetowy odpowiednik Gazety Wyborczej, co wpływa na jego popularność, jednak wydajność nie jest jego mocną stroną.



Rysunek 5.14 Strona główna serwisu gazeta.pl

Źródło: Opracowanie własne

Wynik analizy tego portalu Internetowego jest bardzo słaby. Sama optymalizacja została oceniona na zaledwie 56%. Czas pobierania strony nie jest długi ze względu na mniejszą liczbę obrazków niż w serwisach takich jak Onet czy Wirtualna Polska jednak kompresja używana przez serwer Apache, którego używa portal jest źle skonfigurowany przez co trafił na pierwsze miejsce najważniejszych do poprawy wydajności cech. Poniższy obraz przedstawiający wynik działania narzędzia Snail sugeruje poprawę ósmiu na dziesięć testowanych reguł co jest bardzo złym wynikiem.

Importance	Type
0	Avoid landing page redirects
29.308	Enable compression
10.32631861772487	Leverage browser caching
0	Reduce server response time
0	Minify CSS
2.871800000000001	Minify HTML
1.144099999999997	Minify JavaScript
22	Eliminate render-blocking JavaScript and CSS in above-the-fold content
16.07179999999996	Optimize images
2	Prioritize visible content

Rysunek 5.15 Błędy serwisu gazeta.pl  
 Źródło: Opracowanie własne

Poza odblokowaniem i/lub poprawą konfiguracji kompresji pobieranych danych należy poprawić optymalizację obrazów na stronie, dokonać zmiany ustawień cachowania danych, dokonać minimalizacji plików HTML, JavaScript. Poprawienie wszystkich problemów z wydajnością może znaczco wpływać na szybkość pobierania strony Gazeta.pl. Należy także wspomnieć o błędach w kodzie źródłowym, których liczba to ponad 1001 (skrypt nie przekracza tej liczby błędów). Podsumowując wynik tego testu, gdyby nie popularność Gazety Wyborczej w Polsce, to sam serwis na pewno nie cieszyłby się taką popularnością, bo nie można powiedzieć, że przyciąga użytkowników szybkością działania oraz wydajnością. Wykorzystując narzędzie Snail każdy czytelnik może jednak się przyjrzeć bliżej błędem opóźniającym czas pobierania i wyświetlania jakie posiada serwis Internetowy Gazety Wyborczej.

**Wikipedia.pl** – Polskojęzyczna wersja Internetowej encyklopedii, którą tworzą sami użytkownicy. Jak widać na poniższym obrazku, strona Internetowa Wikipedii nie posiada zbyt dużo grafiki, a sama strona główna jest raczej skierowana do użytkownika, który powinien wybrać odpowiedni dla siebie język przeglądania witryny.



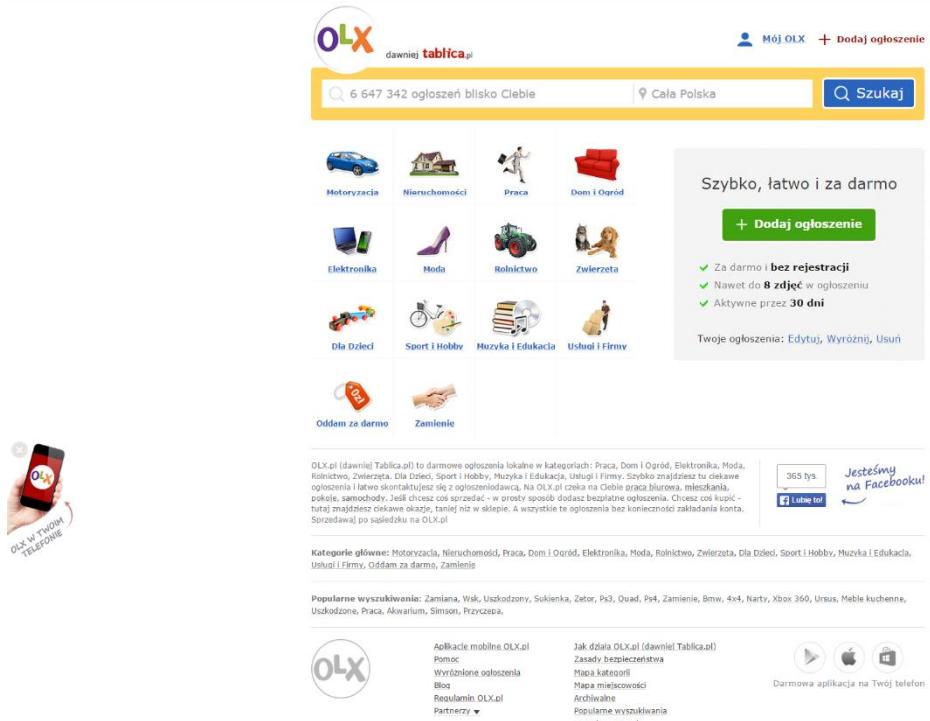
Rysunek 5.16 Widok strony głównej portalu wikipedia.org  
Źródło: Opracowanie własne

Niestety nawet tak nieskomplikowana pod względem funkcjonalności encyklopedia Internetowa nie jest całkowicie wolna od błędów powodujących opóźnienie pobierania tej strony. Największą wagę należy skupić na poprawienie ustawień związanych z zarządzaniem pamięcią w przeglądarce użytkownika. To znaczy zadbać o odpowiednio częste sprawdzanie czy są dostępne nowe elementy na stronie, a pozostałe odpowiednio odtwarzać z bufora agenta Internetowego użytkownika. Następnie (jak w większości opisanych wcześniej stronach), należy wyeliminować blokujące renderowanie strony skrypty JavaScript oraz kaskadowe arkusze stylów. Pozostałe elementy wymagające uwagi to optymalizacja obrazów, oraz minimalizacja rozmiaru pliku HTML dla witryny.

Importance	Type
0	Avoid landing page redirects
0	Enable compression
8.999503968253968	Leverage browser caching
0	Reduce server response time
0	Minify CSS
0.0857	Minify HTML
0	Minify JavaScript
6	Eliminate render-blocking JavaScript and CSS in above-the-fold content
0.2894000000000005	Optimize images
0	Prioritize visible content

Rysunek 5.17 Błędy portalu wikipedia.org  
 Źródło: Opracowanie własne

**Olx.pl** – To serwis oferujący zamieszczanie darmowych ogłoszeń lokalnych w różnych kategoriach. Jak informuje wyżej testowany serwis Wikiperia.org - „OLX jest firmą Internetową posiadającą biura w Nowym Jorku, Buenos Aires, Poznaniu, Moskwie i Pekinie. Serwis ogłoszeniowy OLX pozwala na darmowe zamieszczanie ogłoszeń w wielu kategoriach, takich jak praca, nieruchomości, auto-moto, usługi, korepetycje itp.”



http://olx.pl

Rysunek 5.18 Strona główna portalu olx.pl

Źródło: Opracowanie własne

Jak widać strona główna serwisu nie jest tak rozbudowana jak to miało miejsce np. dla serwisu Onet.pl. Do jej pobrania zostało wysłanych 39 żądań, całkowity czas pobierania wyniósł około dziesięciu sekund co jest dość sporym wynikiem. Całkowity rozmiar witryny to w przybliżeniu 700kb zużytego transferu danych w przypadku pobierania strony poprzez sieć komórkową. Liczbę zapytań do serwera można zmniejszyć o dziewięć poprzez zastosowanie CSS Sprites.

Importance	Type
0	Avoid landing page redirects
0	Enable compression
3.4802372685185192	Leverage browser caching
0	Reduce server response time
0.191	Minify CSS
0.1317	Minify HTML
0.3211	Minify JavaScript
10	Eliminate render-blocking JavaScript and CSS in above-the-fold content
1.8979000000000001	Optimize images
0	Prioritize visible content

Rysunek 5.19 Błędy serwisu olx.pl  
 Źródło: Opracowanie własne

Obrazek przedstawiający, z jaką wagą należy poprawić wydajność poszczególnych elementów serwisu sugeruje zacząć od wyeliminowania blokujących renderowanie strony skryptów JavaScript oraz stylów CSS. Następnymi, co do ważności elementami są zmiana atrybutów wykorzystania pamięci przeglądarki poprzez poszczególne obiekty na stronie oraz optymalizacja grafiki. Zmniejszenie czasu pobierania witryny zostanie osiągnięte także, gdy zmniejszeniu ulegną pliki z rozszerzeniem .css, .html oraz .js.

Narzędzie Snail wskazało również 73 błędy w kodzie źródłowym strony. Większość z nich dotyczy nieprawidłowego użycia kaskadowych arkuszy stylów, lecz są też takie, które dotyczą dwukrotnego wykorzystania atrybutu *ID* dla znaczników HTML, oraz nieodpowiedniego wykorzystania znaczników określających sposób cytowania tekstu na stronie.

Serwis z ogłoszeniami jest na tyle popularny, że jego użytkownikom z pewnością spodobałaby się poprawa wydajności ich ulubionego serwisu, który zamyka zestawienie dziesięciu najpopularniejszych witryn Internetowych w Polsce. Optymalizacja poprzez zmniejszenie liczby opisanych problemów, przekłada się nie tylko na szybkość pobierania strony, ale także wyświetlanie poszczególnych obiektów, oraz niezawodność i wydajność serwisu.

## Wnioski

Jak wynika z przeprowadzonych analiz, sześć na dziesięć czołowych serwisów jest zoptymalizowane w ponad 80%, choć można tą ocenę narzędzia Snail jeszcze poprawić stosując się do wcześniej opisanych reguł. W chwili przeprowadzania badań architektura stron nie jest zoptymalizowana pod kątem nowych technologii takich jak SPDY, HTTP 2.0, ani bardzo dobrze zoptymalizowana (używająca wymienionych reguł wydajności) czy nawet pozbawiona błędów w kodzie samych plików *.html*. Same protokoły sieciowe nie są na tyle „inteligentne”, aby samemu radzić sobie z problemami związanymi z budowaniem serwisów webowych. To programiści powinni przede wszystkim dbać o to, aby ich dzieła nie były ociężałe i uciążliwe dla użytkowników. Dbając o zasady tworzenia wydajnych systemów już na poziomie implementacji można uniknąć wielu kłopotów wydajnościowych w przyszłości. Jednym z takich przykładów są pliki JavaScript. Dopóki nie zostaną one wczytane, przeglądarka blokuje pobieranie innych danych. Dlatego należy mieć na uwadze jak ważne jest odpowiednie umiejscowienie skryptów w kodzie strony. Strony Internetowe są coraz „pojemniejsze”, użytkownicy wymagają, aby ich doświadczenia były jak najlepsze. Wszechobecność skryptów, które symulują ruch obiektów na stronie, odtwarzają i zarządzają animacjami, dynamicznie zmieniają obiekty na stronach, mnogość stylów CSS i coraz bardziej rozbudowane pliki znaczników HTML określające szablon strony i podział na sekcje powoduje, że użytkownik musi poczekać kilka sekund zanim zobaczy upragnioną treść na witrynie. Obecnie nie wystarczy już „tylko” włączenie kompresji dla tych danych, należy zadbać o wiele elementów mających wpływ na szybkość pobierania witryny i wydajność jej działania. Powiedzenie „Duży może więcej” w przypadku przeprowadzonych testów również znajduje zastosowanie. Domeny takie jak Google, Facebook czy Wikipedia są w stanie przesyłać swoje strony do użytkownika w czasie około 4.5 sekund. Są to domeny ogólnosławowe, z których na co dzień korzysta kilka milionów użytkowników. Niestety krajowe „produkty”, z których też korzystają użytkownicy na całym świecie, jednak nie w tak dużej skali, działają zwyczajnie wolniej. Oczywiście serwisy wybrane do klasyfikacji przez serwis Alexa pochodzą z różnych kategorii, lecz to ich popularność jest wyznaczniem miejsca w rankingu. Dlatego kategoria „popularne” jest wspólna dla wybranych portali Internetowych.

Tabela 5.1 Podsumowanie wyników analizy według narzędzia Snail

Miejsce w rankingu	Alexa	Domena	Wielkość strony [KB]	Czas odpowiedzi [s]	Serwer WWW	Metoda kompresji	Liczba żądań HTTP	Liczba błędów w kodzie HTML	Ocena ogólna Snail Project
1	Google.pl	335,921	4,552	gws	gzip	12		32	97%
2	Facebook.com	1168,407	3,038	b.d.	gzip	27		40	89%
3	Allegro.pl	1439,641	101,353	nginx	gzip	78		75	75%
4	Google.com	335,443	6,293	gws	gzip	11		30	90%
5	Youtube.com	656,053	53,391	gwiseguy/2.0	gzip	49		1001	74%
6	Onet.pl	1338,245	121,339	nginx	gzip	153		45	81%
7	Wp.pl	4235,298	468,283	aris	gzip	253		263	76%
8	Gazeta.pl	2949,219	196,772	apache	b.d.	165		1001	59%
9	Wikipedia.org	96,060	3,207	apache	gzip	19		20	85%
10	Olx.pl	692,946	19,830	b.d.	gzip	39		73	85%

Powyższa tabela przedstawia podsumowanie wyników zebranych podczas przeprowadzonego testu. Można w niej znaleźć zarówno informacje o liczbie błędów w języku znaczników HTML, liczbie zapytań do serwera zadanych w celu pobrania witryny, nazwie używanego serwera, wielkości pobranej, czasie odpowiedzi na wszystkie zapytania, używanej metodzie kompresji i najważniejsze, ocenę narzędzie Snail.

## 6. Reguły wydajności

### Unikanie zbędnych przekierowań

Przekierowania URL są stosowane, aby user-agent został przeniesiony w inne miejsce niż nastąpiło poprzednie wywołanie strony. Najczęściej stosowana technika przy skracaniu adresów Internetowych oraz przenoszeniu serwisów webowych pod inny adres.

Przekierowanie za pomocą metatagów „Refresh”:

```
<html><head>
  <meta http-equiv="Refresh" content="0; url=http://www.pwr.edu.pl/" />
</head><body>
  <p>Nowy adres <a href="http://www.wiz.pwr.edu.pl/">link</a>!</p>
</body></html>
```

Liczba 0 w atrybutie „content” określa liczbę sekund w czasie których ma być wyświetlana strona.

Drugim sposobem jest przekierowanie z wykorzystaniem JavaScript:

```
<script language="JavaScript" type="text/javascript">
location.href="pwr.php";
</script>
```

Trzeci sposób to użycie ramek:

```
<frameset rows="100%">
  <frame src="http://www.pwr.edu.pl/">
</frameset>
<noframes>
  <body>Please follow <a href="http://www.wiz.pwr.edu.pl/">link</a>!</body>
</noframes>
```

Dzięki temu rozwiązaniu strona spod nowego adresu wyświetli się, ale w pasku przeglądarki adres się nie zmieni.

Przekierowanie w PHP:

```
<?php
  header('HTTP/1.1 301 Moved Permanently');
  header('Location: http://www.wiz.pwr.edu.pl/');
  header('Connection: close');
  exit();
?>
```

Powyższy kod w języku PHP wygeneruje nagłówek:

```
HTTP/1.1 301 Moved Permanently
Location: http://www.wiz.pwr.edu.pl/
Content-Type: text/html
Content-Length: 174
```

Inne kody stanu 3xx protokołu HTTP jakie można wykorzystać podczas przekierowania:

- 300 – Wiele możliwości, przeważnie wykorzystywany przy stronach wielojęzycznych
- 301 – stałe przekierowanie
- 302 – przekierowanie tymczasowe (znaleziony)
- 303 – zobacz inne
- 307 – przekierowanie tymczasowe

Z punktu widzenia SEO przekierowanie 301 jest ważne, ponieważ przekierowuje ono siłę czy też wagę linków ze starej na nową stronę. Dzięki czemu pozycja w Google nie spada po przeniesieniu strony pod nowy adres.

Ostatnim sposobem jest przekierowanie za pomocą .htaccess:

Poniższą linię kodu należy umieścić w pliku .htaccess:

Redirect 301 / http://www.example.com/

### **Włączenie kompresji danych**

Korzystanie z kompresji przesyłanych danych pomiędzy klientem a serwerem ma na celu bardziej efektywne wykorzystanie dostępnej przepustowości łącza. Włączenie kompresji HTTP zapewnia krótszy czas transmisji danych pomiędzy przeglądarkami Internetowymi a serwerami, na których dane się znajdują. Kompresowane mogą być dane zarówno statyczne jak i te generowane dynamicznie przez aplikacje [23].

Nagłówek Accept-Encoding wskazuje na to, że przeglądarka WWW obsługuje kompresję, natomiast nagłówek Content-Encoding jest stosowany przez serwer (jeśli obsługuje kompresję), który określa skompresowaną odpowiedź.

Najpopularniejszą metodą kompresji danych jest użycie formatu *gzip*. Jest to otwarty format, czyli taki, który nie został objęty żadnymi patentami oraz prawem ograniczającym jego wykorzystywanie. Mniej popularnym i zarazem mniej efektywnym formatem kompresji danych jest *deflate*. Opisane poniżej wyniki testów dziesięciu najpopularniejszych serwisów webowych w Polsce pokazały, że najczęściej stosowaną metodą kompresji przesyłanych danych jest użycie *gzip*.

Tabela 6.1 Właściwości metody kompresji

Opcja kompresji	Typ danych	Podstawowa konfiguracja
Format kompresowanych plików	Statyczny	.txt, .htm, and .html
	Dynamiczny	.exe, .dll, and .asp
Format kompresji	Statyczny	gzip oraz deflate
	Dynamiczny	gzip oraz deflate
Poziom kompresji	Statyczny	10
	Dynamiczny	0

### **Wykorzystanie pamięci podręcznej przeglądarki**

Zapisywanie zasobów statycznych w pamięci podręcznej przeglądarki pomaga oszczędzić czas użytkownikowi, który wielokrotnie odwiedza daną witrynę. Nagłówki zapisywania w pamięci podręcznej powinny dotyczyć wszystkich zasobów statycznych, które można zapisać, a nie

tylko ich niewielkich podzbiorów na przykład obrazów. Zasoby, które można zapisać w pamięci podręcznej to pliki JS i CSS, pliki obrazów oraz inne pliki zawierające obiekty binarne, a w tym, multimedia, pliki PDF itp. Sam HTML nie jest statyczny i nie powinien być domyślnie uwzględniany w zapisywaniu w pamięci podręcznej, ale można zmniejszyć jego rozmiar używając narzędzi do minimalizowania rozmiaru plików .html.

Należy włączyć funkcję zapisywania materiałów z serwera w pamięci podręcznej przeglądarki. Czas przechowywania zasobów statycznych w pamięci podręcznej powinien wynosić co najmniej tydzień. Zasoby zewnętrzne, takie jak reklamy czy widżety, powinny być przechowywane w pamięci podręcznej co najmniej jeden dzień.

Dla wszystkich zasobów zapisywanych w pamięci podręcznej Google zaleca następujące ustawienia:

„Ustaw wartość Expires na co najmniej tydzień, a najlepiej na rok w przód. (Preferujemy ustawienie Expires zamiast Cache-Control: max-age, ponieważ jest częściej obsługiwane). Nie ustawiaj wartości dłuższej niż rok w przód, ponieważ będzie to naruszeniem wytycznych RFC.

Jeżeli wiesz dokładnie, kiedy dane zasoby się zmienią, możesz ustawić krótszy okres ważności. Jeżeli jednak wiesz tylko, że „wkrótce mogą się zmienić”, ale bez konkretów, ustaw długi okres ważności i użyj odcisku cyfrowego w URL-u (technika opisana poniżej).”

Nagłówki *Expires* i *Cache-Control: max-age*, określają okres, w którym przeglądarka może używać zasobów zapisanych w pamięci podręcznej bez sprawdzania, czy na serwerze WWW nie znajduje się ich nowsza wersja. Gdy zostaną ustawione, po pobraniu zasobu przeglądarka nie będzie wysyłała żądań GET do upływu terminu ich ważności, osiągnięcia przez nie maksymalnego wieku bądź do czasu opróżnienia pamięci podręcznej przez użytkownika.

Nagłówki *Last-Modified* i *ETag*, określają sposób decydowania przez przeglądarkę, czy pliki są takie same, na potrzeby zapisania w pamięci podręcznej. W przypadku nagłówka *Last-Modified* takim wyznacznikiem jest data. W przypadku nagłówka *ETag* wskazówką może być dowolna wartość jednoznacznie identyfikująca zasób (standardowo stosowane są wersje pliku i identyfikatory numeryczne zawartości). *Last-Modified* to „słaby” nagłówek, ponieważ przeglądarka określa, czy pobrać dany element z pamięci podręcznej czy nie w oparciu o mechanizmy heurystyczne. Te nagłówki pozwalają przeglądarce skutecznie aktualizować zasoby zapisane w pamięci podręcznej poprzez wysyłanie warunkowych żądań GET, gdy użytkownik wyraźnie ponownie załaduje stronę. Warunkowe żądania GET nie zwracają pełnej odpowiedzi, jeżeli zasób na serwerze się nie zmienił, i z tego względu charakteryzują się krótszym czasem oczekiwania niż pełne żądanie GET.

Ważne jest, by określić jeden z nagłówków *Expires* i *Cache-Control max-age*, oraz jeden z *Last-Modified* i *ETag*, w odniesieniu do wszystkich zasobów, które można zapisać w pamięci podręcznej. Nie ma potrzeby określać zarówno nagłówka *Expires*, jak i *Cache-Control: max-age*, albo i *Last-Modified*, i *ETag*.

### **Skrócenie czasu odpowiedzi serwera**

„Czas odpowiedzi serwera to czas wczytywania kodu HTML niezbędnego do rozpoczęcia renderowania strony z serwera. Mogą wystąpić różnice pomiędzy poszczególnymi cyklami, ale nie powinny one być zbyt duże. W rzeczywistości duża zmienność czasu odpowiedzi serwera może sugerować problem z wydajnością.

Czas odpowiedzi serwera powinien wynosić mniej niż 200 ms. Spowolnienie może być spowodowane przez wiele czynników: mało wydajne algorytmy aplikacji lub zapytania do bazy danych, powolny routing, architekturę, biblioteki, nadmierne wykorzystanie zasobów obliczeniowych procesora bądź pamięci. Poprawiając czas odpowiedzi serwera, należy wziąć wszystkie te czynniki pod uwagę. Po wykonaniu testów oraz interpretacji wyników należy wprowadzać stopniowo różne rozwiązania mające na celu poprawę wydajności. Wykorzystanie różnych popularnych rozwiązań takich jak CMS, MVP, MVC itp. Skutkuje również opisaniem problemów z wydajnością nawet w specyfikacji technicznej danego rozwiązania.” [8]

### **Minimalizowanie plików CSS**

Minimalizowanie rozmiaru plików polega na usuwaniu zbędnych bajtów, na przykład dodatkowych odstępów, komentarzy, znaków łamania wiersza i wcięć. Kompaktowy kod CSS, HTML i JavaScript pomaga skrócić czas pobierania tych plików, ich analizowania oraz wykonywania. Dodatkowo w przypadku CSS i JavaScriptu możliwe jest dalsze zmniejszanie rozmiaru pliku poprzez zmianę nazw zmiennych, stosowanie metodyki DRY (Ang. Don't Reapeat Yourself) [24].

W celu zmniejszenia rozmiaru kodu CSS można użyć narzędzi takich jak YUI Compressor i cssmin.js, które są polecane przez Google. Istnieje także sporo narzędzi programistycznych, które już w trakcie tworzenia kodu HTML, stylów CSS czy skryptów JavaScript dbają i automatyczne modyfikowanie kodu to jak najmniejszego rozmiaru pliku. Są to takie narzędzia jak Koala czy Codekit.

### **Minimalizowanie plików HTML**

Aby zmniejszyć rozmiar kodu HTML można wykorzystać rozszerzenie PageSpeed Insights dla przeglądarki Chrome, aby wygenerować kompaktową wersję kodu HTML. Należy uruchomić analizę HTML danej strony oraz wybrać odpowiednią regułę zmniejszania kodu.

### **Minimalizowanie plików JavaScript**

Narzędziami sugerowanymi przez Google do zmniejszenia ich rozmiaru celem szybszego przesyłania poprzez sieć są np. Closure Compiler, JSMin lub YUI Compressor. Tak jak wspomniano przy opisie minimalizowania rozmiaru plików kaskadowych arkuszy stylów, można użyć także narzędzi typu Koala czy Codekit, które „w locie” dbają o odpowiednie formatowanie kodu i zmniejszają jego rozmiar.

### **Eliminowanie blokujących skryptów**

Zanim przeglądarka wyrenderuje widok strony, w pierwszej kolejności musi dokonać jej analizy. Jeżeli w jej trakcie napotka blokujący zewnętrzny skrypt, musi go zinterpretować zanim dokona dalszej analizy kodu samej strony. Każdy taki skrypt oznacza dodatkowe połączenia z serwerem w celu pobrania kodu JavaScript i przesłanie wyniku jego zawartości do klienta WWW, co znacznie pogarsza User eXperience.

Krótkie skrypty JavaScript wymagane do wyrenderowania części strony widocznej na ekranie powinny zostać umieszczone pomiędzy znacznikami HTML, natomiast pobranie kodu JavaScript niezbędnego do realizacji dodatkowych funkcji takich jak ruchome elementy, pokazy slajdów, itp. powinno odbywać się dopiero po dostarczeniu treści.

Krótkie skrypty zewnętrzne możesz włączyć bezpośrednio do dokumentu HTML. Wbudowanie niewielkich plików pozwoli przeglądarkom kontynuować renderowanie strony.

Aby zapobiec blokowaniu wczytywania stron przez kod JavaScript, przy wczytywaniu zasobów zaleca się użycie w kodzie HTML atrybutu *async*.

Na przykład:

```
<script async src="my.js">
```

Rozwiążanie to jednak nie sprawdza się jeżeli w kodzie JavaScript używana jest komenda `document.write`. Należy wtedy przepisać kod w taki sposób, aby nie wykorzystywać `document.write` do wpisywania tekstu do dokumentu HTML. Przy asynchronicznym wczytywaniu skryptów, należy zwrócić uwagę, by aplikacja uwzględniała wczytywanie ich w odpowiedniej kolejności, jeżeli jest to kod JavaScript zależący od siebie nawzajem.

Wykorzystanie różnych bibliotek JavaScript, na przykład jQuery, umożliwia wzbogacenie strony o dodatkowe funkcje takiej jak animacje, przejścia, zmianę stylów CSS i inne efekty. Większość takich skryptów można zastosować już po wyrenderowaniu części widocznej na ekranie dla użytkownika. Ponieważ na doświadczenia użytkownika, oraz czas ładowania się strony ma to duży wpływ, warto stosować tą radę w większości projektów aplikacji webowych.

## Optymalizacja obrazów

Stosowanie jak najmniejszych rozmiarów obrazów, ma na celu skrócenie czasu oczekiwania na załadowanie przez użytkownika. Właściwe sformatowanie i skompresowanie obrazów może zaoszczędzić wiele bajtów danych. Skracą to czas wczytywania strony przy wolnym połączeniu, i jest szczególnie cenione przez użytkowników „mobilnych”, ze względu na transfer danych i opóźnienia sieci [25].

„Warto przeprowadzić zarówno podstawową, jak i zaawansowaną optymalizację wszystkich obrazów. Podstawowa optymalizacja to wycinanie zbędnych obszarów, maksymalna akceptowalna redukcja głębi kolorów, usuwanie komentarzy do obrazów i korzystanie z odpowiedniego formatu plików” [8]. Te czynności można wykonać przy pomocy dowolnego programu do edycji obrazów, takiego jak GIMP [25] czy Adobe Photoshop [26].

W sieci dostępnych jest wiele narzędzi umożliwiających bezstratną (tzn. bez szkody dla jakości obrazów) kompresję plików JPEG i PNG. Są to zarówno narzędzia darmowe (jpegtran, optipng, czy Gimp) oraz komercyjne (Adobe Photoshop – potężne narzędzie służące zarówno do edycji jak i tworzenia grafiki, nie tylko na potrzeby Internetu)[8]

- Pliki PNG są niemal zawsze lepsze od plików GIF, jednak niektóre starsze wersje przeglądarek mogą tylko częściowo obsługiwać pliki PNG charakteryzujące się obsługą kanału alfa.
- Pliki GIF stają się coraz mniej popularne w sieci, ponieważ bezcelowe jest ich wykorzystywanie do grafik przedstawiających statyczne obrazy, ze względu na niską jakość oraz większy rozmiar w porównaniu z formatem PNG.
- Formatu JPG najlepiej używać w przypadku wszystkich plików, które nie wymagają przezroczystości czyli przedstawiające zdjęcia, portrety, krajobrazy.

## Priorytetyzacja widocznej treści

W momencie, gdy wymagana ilość danych przekracza początkowy rozmiar okna przeciążenia, konieczna jest dodatkowa wymiana informacji między serwerem a przeglądarką użytkownika, co znaczco wpływa na czas odpowiedzi. W przypadku sieci o długim czasie oczekiwania, na

przykład sieci komórkowych, takich jak 3G, 4G, może to powodować poważne opóźnienia wczytywania stron.

Krok jaki należy wykonać, aby strony wczytywały się szybciej, to ograniczenie rozmiaru plików zawierających znaczniki HTML, obrazy, style CSS czy kod JavaScript, które są wymagane do wyrenderowania widocznej dla użytkowników części strony.

Już na etapie projektowania serwisu webowego można zastosować pewne reguły, które będą miały wpływ na jego wydajność:

- Należy zaplanować strukturę strony w taki sposób, aby jej najważniejsza część, widoczna na ekranie, wczytywała się w pierwszej kolejności. Jeżeli HTML wczytuje zewnętrzne skrypty np. jQuery przed główną zawartością, najważniejsza jest zmiana kolejności w taki sposób, żeby główna zawartość wczytywała się jako pierwsza.
- Należy zmniejszyć ilość danych wykorzystywanych przez zasoby serwisu.
- Po przeprojektowaniu witryny pod kątem poprawnego wyświetlania na różnych urządzeniach i wczytywania najważniejszej zawartości na początku trzeba zmniejszyć ilość danych niezbędnych do wyrenderowania stron, posługując się następującymi metodami:
  - Należy stosować style CSS zamiast obrazów tam, gdzie jest to możliwe np. jako tło dla serwisu, czy przycisków itp.
  - Należy włączyć kompresję danych używając np. gzip
  - Wczytywanie skryptów należy umieścić na końcu kodu HTML
  - Należy zoptymalizować fonty używane w całym serwisie webowym

## Optymalizacja fontów

Aby nie narazić serwisu webowego na opóźnienia związane ze zbyt długim wczytywaniem fontów [27], należy odpowiednio je zoptymalizować. Dzięki temu doświadczenia użytkowników będą lepsze i pozwolą odczuć pozytywne skojarzenia z daną stroną. Odpowiednie użycie czcionek pozwala zwiększyć wygodę użytkowników: rozszerzyć rozpoznawalność marki, polepszyć czytelność, funkcjonalność i sprawność wyszukiwania, zapewniając przy tym możliwość zmiany skali i pracy przy wielu różnych rozdzielczościach i formatach ekranu, od wielocalowych monitorów po wyświetlacze urządzeń przenośnych.

Przede wszystkim nie należy używać zbyt wielu krojów pisma na jednej czy kilku stronach serwisu. Pozwoli to na skrócenie czasu oczekiwania użytkowników na wyświetlenie danej treści.

„Należy używać podzbiorów fontów. W przypadku wielu czcionek można wydzielić podzbiory, rozdzielić je na wiele zakresów Unicode i dostarczyć tylko glify konkretnie wymagane na danej stronie – pozwala to ograniczyć rozmiar pliku i zwiększyć prędkość pobierania zasobów. Przy określaniu podzbiorów należy mieć na uwadze optymalizację czcionek pod kątem ponownego użycia. Jeżeli na kilku podstronach serwisu wykorzystywane są różne znaki alfabetu łacińskiego czy też cyrylicy, dobrze dobrana czcionka będzie wszystkie te znaki obsługiwać. Czcionki należy przesyłać w formatach dostosowanych do każdej z przeglądarek: każda czcionka powinna być dostarczana w formatach WOFF2, WOFF, EOT i TTF. Ważne jest, aby mieć pewność, że do formatów EOT i TTF stosowana będzie kompresja GZIP, ponieważ formaty te nie są domyślnie kompresowane, co może skutkować wydłużonym czasem oczekiwania na wyrenderowanie strony.

Jedną z najważniejszych czynności podczas optymalizacji fontów jest określenie zasad ponownej walidacji i optymalnego buforowania, ponieważ czcionki to zasoby statyczne. Są rzadko aktualizowane, dlatego warto zmienić ustawienia serwera tak, aby zapewniał długi okres ważności max-age i wysyłał tokeny walidacji, co umożliwia efektywne ponowne używanie czcionek na różnych stronach w witrynie.

@font-face jest regułą CSS umożliwiającą określenie lokalizacji konkretnego zasobu czcionki, jej stylu i kodów Unicode, dla których powinien obowiązywać. Połączenie takich deklaracji reguł @font-face można wykorzystać do utworzenia rodziny czcionek, dzięki której przeglądarka może określić, które czcionki trzeba pobrać i zastosować na bieżącej stronie.

W każdej deklaracji reguły @font-face określa się nazwę rodziny czcionek, dzięki której może ona pełnić rolę logicznej grupy wielu deklaracji, właściwości czcionki, takie jak styl, grubość, rozcięgnięcie i deskryptory źródłowy zawierający uporządkowaną według ważności listę lokalizacji tego zasobu czcionki” [8].

```
@font-face {  
    font-family: 'PWR Font';  
    font-style: normal;  
    font-weight: 450;  
    src: local('PWR Font'),  
        url('/fonts/pwr.woff2') format('woff2'),  
        url('/fonts/pwr.woff') format('woff'),  
        url('/fonts/pwr.ttf') format('ttf'),  
        url('/fonts/pwr.eot') format('eot');  
}
```

## WebP

Firma Google wydała własny kodek obrazów, który miał być pewnego rodzaju rewolucją. Webp, to nowy format obrazów rastrowych, zarówno animowanych jak i tych statycznych. Według producenta rozmiar plików w formacie .png zostanie zmniejszony aż o 26%, plików z rozszerzeniem.jpeg w przedziale od 25 do 34%, a pozostałych formatów obrazu około 22%. Należy także dodać, że kompresja obrazków jest stratna. Bez obaw można natomiast obrazki bez tła czyli z tak zwanym kanałem alfa, bowiem nie zostanie ono zastąpione białym tłem tak jak ma to miejsce podczas konwersji plików z formatu .gif na .jpeg.

Pierwszą przeglądarką, która wprowadziła obsługę powyższego formatu obrazków była oczywiście Google Chrome, od wersji 9.0.569, która jest tworzona przez tego samego producenta. Kolejną przeglądarką wspierającą opisywany kodek była Opera 11.10. Został on wykorzystany w funkcji „Opera Turbo”. Sposób działania polegał na tym, że serwer „w locie” kompresował obrazki i przesyłał je do klienta, a ostatecznie do odbiorcy końcowego.

Nie wszyscy Twórcy popularnych przeglądarek Internetowych zaadoptowali pomysł przedsiębiorstwa z Mountain View. Jak można przeczytać na forum webhosting.pl [28], Twórcy Firefox oraz Internet Explorer nie wsparli obsługi nowego formatu plików w swoich programach. Jako argumenty takiego zachowania wymienili wady nowo powstałego rozwiązania:

- (jeszcze we wczesnych wersjach) brak kanału alfa
- brak wsparcia profili ICC

- brak obsługi metadanych EXIF (np. brak informacji o parametrach z aparatu użytkownika)

20 maja 2011 roku Jeff Muizelaar (jeden z pracowników Mozilli) stwierdził, że nowy format jest „wypieczony” tylko w połowie”.

Wpis na forum pochodzi z roku 2011, aby się przekonać czy po dość długim okresie adaptacji .webp, wykonano prostu test użyteczności kodeka.

W pierwszej kolejności pobrano odpowiednie kodeki [29][30], i rozpakowano je na dysku twardym. Następnie w celach testowych wybrano logo Politechniki Wrocławskiej [31], oraz przygotowano szablon HTML zawierający tylko linie kodu wczytujące dwa obrazki. Pierwszy- oryginalne logo w formacie .gif, a drugi- zakodowany w formacie .webp. Po zapoznaniu się z dokumentacją kodeków na stronie producenta [32], użyto skryptu o nazwie „gif2webp”, aby dokonać konwersji typu pliku przy użyciu komendy:

„*gif2webp -o poziom-en.webp poziom-en.gif*”

Następnie pliki (oryginalny i ten powstały w wyniku działania skryptu) zostały przeniesione do odpowiedniego folderu, aby mogły zostać wczytane przez stronę HTML.

Pierwsze co rzuciło się w oczy to waga plików:

- Oryginalny ma rozmiar 603 x 100 pixeli, wagę 6,58 KB oraz 8 bitową głębię kolorów.
- Informacje o pliku w nowym formacie goole'a, rzeczywiście nie podają właściwie żadnych informacji o rozmiarze czy paletie barw. Jedyna informacja jaką można uzyskać to rozmiar pliku. Wynosi on 4,41 KB

Test możliwości otwarcia strony Internetowej z prawidłowo wczytanym logo Politechniki Wrocławskiej potwierdził jedynie słowa sceptyków sprzed lat. Jedynie przeglądarki Chrome w wersji (39.0.2171) oraz Opera (ver. 26) były w stanie pokazać oczekiwany efekt. Po wywołaniu strony w tych programach ukazały się dwa identyczne pod względem wymiarów, oraz w subiektywnej ocenie tej samej jakości pliki graficzne. Firefox w wersji 34.0.5, Internet Explorer 11 oraz Safari 5.1.7 nie przeszły testu prawidłowo. Miejsce, w którym powinno znajdować się wczytane logo zostało zastąpione pustym miejscem lub ikoną symbolizującą błędnie wczytany obraz.

Format .webp powstał z zamysłem przyspieszenia ładowania się obrazków, poprzez ich kompresję. Kodek jak i format nie zdobyły jednak dużej popularności i uznania w oczach programistów. Google zaprezentowało swój kodek 30 września 2010 roku. Po ponad czterech latach jego obecności jedyne szerokie zastosowanie, jakie udało się znaleźć to obrazki w wirtualnym sklepie GooglePlay, na urządzeniach mobilnych działających pod systemem Android, jest to niezauważalna dla odbiorcy zmiana, raczej wymuszona przez Google.

## CSS Sprites

Za pomocą odpowiednich stylów CSS można oszczędzić pasmo serwera oraz czas ładowania się strony, oraz zapobiec nieprzyjemnym efektom „doładowywania” się brakujących grafik. Użycie techniki CSS Sprites jest bardzo proste a jednocześnie rzadko wykorzystywane przez twórców stron Internetowych. W przypadku, gdy programista używa około 30 obrazków takich jak logo witryny, ikonki, przyciski czy tła, są one przypisywane do elementów kodu HTML za pomocą kaskadowego arkusza stylów, jako ich tła. W momencie, gdy użytkownik odwiedza daną witrynę, przeglądarka pobiera kod HTML oraz pliki ze stylami CSS, a także wysyła żądania do serwera, w których prosi o przesłanie wszystkich dodatkowych elementów, jakie zadeklarowane zostały w kodzie. W tym przypadku będą to obrazy. Przy 30 obrazkach, serwer musi odpowiedzieć aż na 30 żądań, by strona wyglądała poprawnie, natomiast po użyciu opisywanej techniki będzie to jedno zapytanie do serwera i jedna odpowiedź. W tym czasie zostaną pobrane wszystkie obrazy w jednym pliku, który dodatkowo przy pomocy np. biblioteki jQuery można wyświetlić dopiero po załadowaniu się całego dokumentu HTML.

## CDN

Content Delivery Network [33] to sieć wielu komputerów używanych w celu zwiększenia wydajności dostarczania serwisów Internetowych do odbiorców oraz w celu zmniejszenia obciążenia serwera dostawcy treści. W sieci CDN zawiera się wiele serwerów „dostawców” rozmieszczonych w wielu miejscach na całym świecie. Ich zadaniem jest pobieranie bądź aktualizacja swoich danych na podstawie głównego serwera webowego tak, aby każdy serwer w danej sieci zawierał te same informacje. W momencie odwiedzenia przez użytkownika danego serwisu, widzi on kopię danych dostarczoną przez najbliższy serwer, co skraca czas odpowiedzi na zapytanie.

Content Delivery Network działa na podobnej zasadzie jak proxy między użytkownikiem a danym serwerem. W zależności od hostingu na, którym programista zamierza umieścić swoją aplikację można zaoszczędzić ponad 80% transferu. Dzieje się tak dlatego, że niezależnie od kraju w którym znajduje się docelowy serwer z aplikacją, przy użyciu CDN użytkownik skorzysta z najszybszego i najbliższego dla jego lokalizacji serwera usługi CDN.

Content Delivery Network to bardzo szybko rozwijająca się technologia. Początkowo zakładała tylko dostarczanie statycznych treści dla użytkowników. Jej rozwój okazał się tak dynamiczny, że obecnie jest wykorzystywana również do rozpraszania transmisji strumieniowych oraz coraz częściej aplikacji webowych. Istnieje wiele firm czy operatorów świadczących usługi z wykorzystaniem powyższego rozwiązania. Wybór odpowiedniego dostawcy usługi pozwala znaczco zmniejszyć koszty utrzymania własnej infrastruktury sieci komputerów.

Fundamentalną korzyścią korzystania z CDN jest zmniejszenie obciążenia serwera głównego z aplikacją Internetową. Użytkownicy otrzymują dane z serwera, który CDN określi jako najbliższy do komputera użytkownika. Drugą korzyścią jest oczywiście wspomniany wcześniej, zdecydowanie krótszy czas odpowiedzi na zapytania do serwera, co skutkuje szybszym dostarczeniem zawartości do komputera użytkownika, przez co witryna czy serwis jest atrakcyjniejszy dla odbiorcy.

Prócz czasu dostępu nie ma nic ważniejszego niż sama dostępność serwisu. Najczęściej usługa CDN chroni aplikację przed atakami typu DoS oraz DDoS. Ten drugi to rodzaj ataku na serwer

lub system komputerowy, w którym uniemożliwia się działanie takiego systemu poprzez zapełnienie jego wszystkich wolnych zasobów. Taki atak jest zazwyczaj przeprowadzany przez wiele komputerów tzw. „zombie”. Takie komputery same łączą się z daną usługą sieciową, bez obecności użytkownika. Bezpieczeństwo aplikacji powinno być zapewnione jeszcze przed jej implementacją. Jaka jest użyteczność, nawet najbardziej wydajnej strony www, którą w każdej chwili może usunąć każdy bardziej doświadczony użytkownik sieci lub haker ?

Duże firmy, które posiadają własne sieci serwerów na całym świecie udostępniają usługę CDN. Jest to zdecydowanie tańsze rozwiązywanie jeśli chodzi o cenę wykorzystania takiego rozwiązania w porównaniu do kosztów samodzielnego instalowania serwerów w różnych miejscach świata (czy tylko Polski). Zewnętrzne firmy oferują także dodatkowe usługi oferujące niezawodność na przykład poprzez kopię danych lub redundancję i odtwarzanie ich w przypadku problemów z danym serwerem. Pozostawiając konfigurację serwera, który będzie trzymaj kopię danej strony, często też można korzystać z najnowszych technologii nawet, jeśli nie została ona wdrożona na serwerze „głównym”. Dla przykładu firma CloudFlare dba o to, aby świadczone przez nich usługi korzystały z protokołu SPDY, obsługi priorytetów dla zapytań, czy optymalizacji dla sesji. W chwili obecnej nie są to jeszcze standardowe rozwiązania stosowane na szeroką skalę, raczej są dopiero wdrażane w nowszych projektach aplikacji webowych. Firma ta zapewnia też ważną ze względów wydajnościowych funkcję nazwaną przez nich „Always Online”. Zostanie aktywowana w momencie gdy „główny serwer”, na którym docelowo została wdrożona aplikacja webowa przestanie działać. Kopia zostanie odtworzona z pamięci cache serwerów tak, aby była zawsze dostępna dla użytkowników.

Prócz użytkowników poszukujących informacji, Internet przemierząją roboty sieciowe, takie jak indeksujący witryny Googlebot. Szybszy czas odpowiedzi jest przez niego postrzegany jako zaleta, co ma wpływ na późniejsze umieszczenie witryny w hierarchii rezultatów wyszukiwanych przez Google.

Programista lub właściciel aplikacji webowej może sam wybrać, czy chce skorzystać z usług darmowych czy komercyjnych usług CDN.

Przykłady komercyjnych dostawców usługi:

- Akamai Technologies [34]
- Amazon CloudFront
- Mirror Image
- CacheFly
- CDNetworks
- ChinaCache

Wśród darmowych dostawców CDN są takie firmy jak:

- BootstrapCDN [35]
- CloudFlare [36]
- Coral Content Distribution Network
- Incapsula (darmowa wersja w reklamami Incapsula)
- Google - PageSpeed Service

## **Strona WWW czy aplikacja ?**

Współczesne strony Internetowe są coraz bardziej skomplikowane a do ich tworzenia używane są coraz to nowocześniejsze narzędzia. Aby otrzymać jak najlepszą wydajność aplikacji webowych należy dokonać optymalizacji na wielu płaszczyznach. Każda z nich ma swoje własne wymagania systemowe lecz każdą można optymalizować. Protokoły transportowe, komunikacyjne, bazy danych, „UX - User eXperience”, czy po prostu front-end i back-end aplikacji. Nie ma jednej „złotej zasady”, którą można stosować z „zamkniętymi oczami” do każdej aplikacji czy strony Internetowej. Najważniejsze jest określenie celu jaki zamierza się osiągnąć podczas optymalizacji oraz skąd się bierze problem z wydajnością. „Snail Project” w pewnym stopniu pomaga wytknąć błędy, lub problemy z daną aplikacją.

Najczęstsze problemy w komunikacji na linii klient – serwer to ograniczenia protokołu TCP, które w dużym stopniu wpływają na działanie protokołu http, ograniczenia oraz funkcjonalność protokołu HTTP, opóźnienia pasma oraz jego wydajność w komunikacji WWW, ograniczenia oraz optymalizacja przeglądarki. Największe jednak znaczenie mają aktualne trendy w tworzeniu stron. Przecież to użytkownik wymaga, aby strona była w odpowiedni sposób sformatowana, a także swoją funkcjonalnością, wydajnością oraz interakcją zachęcała do niejednokrotnego skorzystania z jej zasobów.

Obecne wymagania użytkowników co do interfejsów Internetowych aplikacji nie zostaną zaspokojone poprzez prosty szablon HTML z kolorowymi hiperlinkami. Aktualnie przeciętny użytkownik Internetu nie jest w stanie prawidłowo ocenić czy aktualnie znajduje się „na stronie www” czy też już korzysta z „aplikacji webowej”. Dla programisty jest to istotna różnica, szczególnie kiedy chce wprowadzić poprawki wydajnościowe w danym projekcie.

Przodkiem strony Internetowej jest zwykły dokument hipertekstowy. Stanowił on podstawę działalności protokołu HTTP 0.9. Składał się z odpowiednio sformatowanego tekstu przy użyciu znaczników oraz zawartych w nim odnośników. Nawiązanie sesji przy użyciu tego protokołu zakładało tylko wysłanie dokumentu z hipertekstem do odbiorcy.

Strona Internetowa to poniekąd rozszerzenie dokumentu hipertekstowego. Zmiana natomiast polega na tym, że grupa robocza, która pracowała nad protokołem HTTP rozszerzyły możliwości hipertekstu poprzez obsługę mediów. Multimedialne treści takie jak obrazy, dźwięk czy wideo na stronach Internetowych stały się atrakcyjne dla odwiedzających. Bogate treści były jednak statyczne co doprowadziło do ewoluwowania statycznych stron w aplikacje WWW. Zasobne treści były również powodem dla, którego sam protokół HTTP przeszedł niemałą przemianę. Wielokrotne połączenia, szybsze przesyłanie dokumentów, skrócenie czasów ładowania się strony, caching, oraz metadane HTTP, podtrzymywanie połączenia to najważniejsze z cech nowszych wersji protokołu transportowego.

Obecne strony Internetowe, w których zaimplementowana interakcja z użytkownikiem jest już podstawową cechą noszą raczej nazwę aplikacji webowych. Różne skrypty mające na celu zwiększenie atrakcyjności, animacje obrazów, nietypowe wyświetlanie informacji pobranych z baz danych czyni współczesny Internet pełnym mniej lub bardziej zaawansowanych projektów stron www.

W związku z ciągłym rozwojem dokumentów hipertekstowych, czas ładowania się strony nie jest jedynym wyznacznikiem wydajności. Ważne są także inne cechy jak ilość jednocześnie ładowających się fragmentów, sekwencje wgrywanych elementów, kolejność wykonania

skryptów, stabilność i szybkość serwera, bazy danych, połączenia sieciowego, pierwszej reakcji użytkownika itp.

## **Doświadczenia użytkownika w płyn na wydajność**

„User experience, UX (ang. doświadczenie użytkownika) – całość wrażeń, jakich doświadcza użytkownik podczas korzystania z produktu interaktywnego. Pojęcie to używane jest najczęściej w odniesieniu do oprogramowania, serwisów Internetowych lub urządzeń elektronicznych.” [37].

Tak naprawdę bardzo ciężko jest określić kiedy aplikacja jest szybka i wydajna, ponieważ w zależności od odczuć danej osoby, określanie tych wartości może być różne, dla każdej osoby. Jest to ocena na tyle subiektywna, że nie jest nawet oceniana z punktu widzenia wydajności. Ocena nie może podlegać dobór tonacji kolorów, rozmieszczenie elementów na stronie Internetowej czy animacje jakie zostały a niej zaimplementowane.

Oczywiście inaczej to wygląda z punktu widzenia programisty, który koduje wszystkie elementy. To on musi zadbać o to, aby kod czy skrypty mają dostarczyć użytkownikowi „wrażeń”, były odpowiednio dobrze walidowane oraz interpretowane przez klientów stron www.

Steve Souders jest pracownikiem firmy Google. Zajmuje się wydajnością stron WWW i inicjatywami open source. Jest twórcą YSlow - rozszerzenia dla dodatku Firebug, służącego do analizy wydajności i optymalizacji stron Internetowych. Opisał jak można niezależnie od używanego urządzenia, telefon komórkowy, laptop, komputer stacjonarny czy tablet, określić (nie jest to oficjalna metoda) zależność czasu opóźnienia od wrażeń użytkownika.

*Tabela 6.2 Wrażenie użytkownika a opóźnienie*

<b>Opóźnienie [ms]</b>	<b>Wrażenia użytkownika</b>
0-100	Natychmiastowa reakcja użytkownika
100 – 300	Odczucie niewielkiego opóźnienia
300 – 1000	Komputer pracuje normalnie
Powyżej 1000	Mögliwe rozproszenie uwagi
Powyżej 10 000	Przerwanie czynności przez użytkownika

Jak wynika z tabeli, należy jak dostarczyć treść użytkownikowi w czasie poniżej jednej sekundy, aby użytkownik nie rozproszył swojej uwagi. Należy mieć na uwadze także opóźnienia wynikające z połączenia poprzez TCP. Dobrze przygotowana aplikacja jest w stanie, w czasie poniżej jednej sekundy wyświetlić jakąś treść, lub taki element strony, aby skupić uwagę użytkownika, podczas gdy pozostałe elementy zostaną wczytane w następnej kolejności, lecz jak najkrótszym czasie.

Doświadczenia użytkownika to temat, który porusza różne dziedziny nauk takie jak psychologia, filozofia, geometria, czy sztuka. Kolejność wymienionych dziedzin nie ma tutaj znaczenia, jednak każda z nich może mieć wpływ podczas projektowania aplikacji na jej wygląd. Na przykład geometria i sztuka, na rozkład elementów strony i wykorzystane barwy. Wszystkie koncepcje aplikacji Internetowych można przetestować z technicznego punktu widzenia wykorzystując do tego celu „Snail Project”. Zastosowanie się do wskazówek i wyników testu pomoże nawet najbardziej atrakcyjną dla użytkownika treść przekształcić w

wydajną i dobrze zoptymalizowaną, nie obciążającą sieci treść przesyłaną do klienta poprzez protokoły TCP/IP czy HTTP.

## **Wydajność przeglądarek internetowych**

W trakcie projektowania aplikacji internetowych nie trzeba się martwić o poszczególne gniazda TCP czy UDP, ponieważ zarządza nimi przeglądarka internetowa. Dodatkowo stos sieciowy zarządza ograniczeniem liczby połączeń, formatowaniem żądań, izolowaniem poszczególnych aplikacji między sobą oraz obsługę serwera proxy . Ale najważniejsze jest, aby mimo wszystko nie zapominać o tym, że wydajność protokołów TCP, http czy sieci mobilnych w dużej mierze wpływa na jakość działania tworzonej aplikacji.

Nowoczesne przeglądarki Internetowe są tworzone z myślą o szybkim i bezbłędnym dostarczaniu treści poprzez Internet. Zawierają w sobie ogromną liczbę warstw optymalizacji obsługi stron www. Popularne programy do odwiedzania stron w Internecie zawierają moduły zarządzające procesami, dbające o bezpieczeństwo w sieci, optymalizujące pamięć podręczną, renderowanie grafiki, maszyny wirtualne Javy, obsługę audio-video i wiele innych. To prawie systemy operacyjne dla sieci.

Ogólna wydajność przeglądarki i dbałość o jej najbardziej aktualną wersję ma wielki wpływ na ogólną wydajność i szybkość uruchamiania się aplikacji Internetowych. Dlatego w projekcie, który przygotowano (narzędzie narzędzie Snail) można sprawdzić nie tylko obsługę aktualnych rozwiązań technologicznych takich jak HTML 5, CSS 3. Dzięki zaimplementowanej w projekcie bibliotece „modernizr” [13], można określić poprawność takich opcji jak caching strony, obsługę video dla HTML 5 etc. Nawet najlepsza optymalizacja serwisu nie będzie miała wpływu na wydajność jeśli użytkownik końcowy w dalszym ciągu posługuje się przeglądarką Internet Explorer 6 zaimplementowanej w Windows XP. W przeciwieństwie do IE, Google Chrome „uczy się” [38] działać coraz szybciej w trakcie jego obsługi przez użytkownika. Przeglądarka „zapamiętuje” kolejność przeglądania stron, najczęstsze wyszukiwania, sposób zadawania zapytań do wyszukiwarki. Wszystko ma na celu optymalizację działania poprzez uprzedzanie działań użytkownika tak, aby kolejne „zadania” wykonywały się szybciej, niż miałoby to nastąpić bez tej optymalizacji. Chrome wstępnie rozpoznaje nazwy za pomocą DNS, wcześniej nawiązuje połączenie TCP, aby otrzymać wstępny wygląd strony.

Stos obsług sieci we współczesnych przeglądarkach jest nie tylko menedżerem gniazd komunikacji, to wręcz platforma do ich obsługi, która posiada własne kryteria optymalizacji, interfejsy API, wtyczki, dodatki i usługi. Każdy producent tych aplikacji uważa swoje parametry ustawień za najbardziej wydajne, to właśnie dlatego do wyboru jest pokaźna gama takich programów do korzystania z Internetu jak, Chrome, Opera, Internet Explorer, Safari, Firefox czy Chromium.

Cykl życia pojedynczych gniazd komunikacyjnych nie jest zarządzany przez aplikacje Internetowe uruchomione przez przeglądarkę. Dzięki temu krytyczne mechanizmy optymalizacji wydajności są zautomatyzowane, np. ponowne wykorzystanie gniazd, nadawanie priorytetów żądaniom, negocjacje protokołów, ograniczenie liczby powiązań, późne wiązania itd. Założeniem przeglądarek jest oddzielenie zarządzania cyklem życia żądań od zarządzania gniazdami komunikacyjnymi.

Gniazda są grupowane w pule według źródeł, każda z nich określa własne zasady bezpieczeństwa i wymagania dotyczące liczby połączeń. Żądanie, które zostanie zawieszone,

jest ustawiane w kolejne i zostanie mu nadany priorytet. Następnie każde żądanie zostanie powiązane z gniazdem w puli. Gniazdo może zostać użyte ponownie przez wiele żądań, jeśli sam serwer nie zakończy połączenia. Umieszczanie gniazd w puli i ponowne użycie połączenia TCP odbywa się automatycznie, dzięki czemu zyskuje wydajność aplikacji. Przeglądarka może optymalizować pulę i zamykać nieużywane gniazda. Obsługiwać żądania w kolejce według priorytetów. Aktywnie otwierać gniazda i uprzedzać żądania. Minimalizować opóźnienie i zwiększać prędkość transmisji poprzez ponownie wykorzystanie gniazd komunikacyjnych.

Usługi takie jak zarządzanie gniazdami komunikacyjnymi, połączonymi, pamięcią podręczną, bezpieczeństwo, obsługa żądań i odpowiedzi są realizowane przez odpowiednie typy połączeń. HTTP, WebRTC, XMLHttpRequest, czy Server-Sent Events wykorzystują odpowiednio wszystkie bądź tylko niektóre z wymienionych usług. W Internecie jest tyle różnych stron czy aplikacji ilu jest ich twórców, dlatego też nie ma jednego najlepszego protokołu czy interfejsu API w pełni wydajnie działającego dla wszystkich typów połączeń czy usług. W zależności od potrzeb aplikacji może ona korzystać z jednej lub kilku różnych protokołów transportowych, opóźnień komunikatów, w różny sposób wykorzystywać pamięć podręczną przeglądarki. Poniższa tabela przedstawia porównanie funkcjonalności protokołów XHR [39], SSE oraz WebSocket. Wybór odpowiedniego może stanowić „drugie życie” dla wolno działających i niewydajnych aplikacji, które są źle odbierane przez użytkowników.

*Tabela 6.3 Wysokopoziomowe funkcjonalności protokołów*

	XMLHttpRequest	Server-Sent Events	WebSocket
<b>Strumienianie żądań</b>	tak	nie	tak
<b>Strumienianie odpowiedzi</b>	ograniczone	tak	tak
<b>Mechanizm ramkowania</b>	tak	nie (base64)	ramkowanie binarne
<b>Binarna transmisja danych</b>	tak	tak	tak
<b>Kompresja</b>	tak	tak	ograniczona
<b>Aplikacyjny protokół transportowy</b>	HTTP	HTTP	WebSocket
<b>Sieciowy protokół transportowy</b>	TCP	TCP	TCP

Przeglądarki Internetowe po wywołaniu adresu jakieś strony, sprawdzają czy zasób nie jest przechowywany w ich pamięci podręcznej, jeżeli tak to zwracana jest zawartość z kopii zapisanej lokalnie. W przeciwnym przypadku, wysyłane jest żądanie do sieci, a po otrzymaniu odpowiedzi jest ona umieszczana w cache-u przeglądarki tak, aby była dostępna w przypadku kolejnego żądania i dostęp do niej niemal natychmiastowy. Przeglądarki Internetowe automatycznie wykonują pewne czynności:

- Przetwarzają zlecenia dostępu do każdego zasobu
- Zarządzają wielkością pamięci podręcznej i usuwają jej zasoby

- Nadają ważność nieaktualnym zasobom

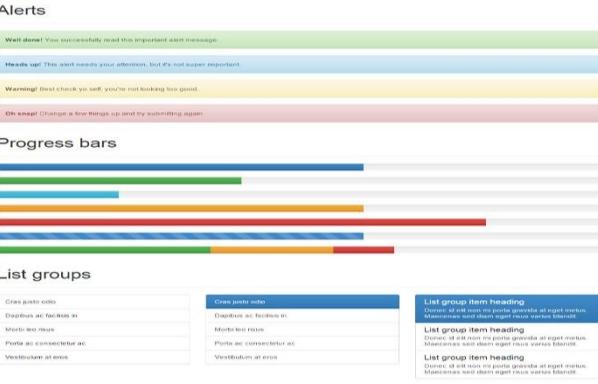
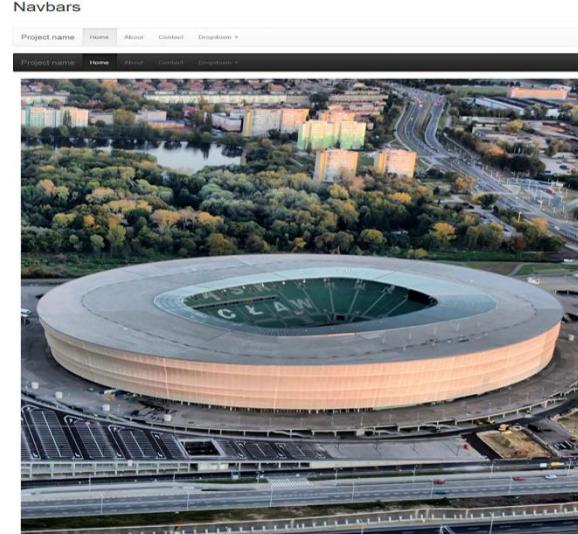
Aby przeglądarka po stronie klienta mogła prawidłowo zarządzać pamięcią, należy zadbać o to aby serwer zwracał prawidłowe zlecenia dostępu do pamięci podręcznej. Odpowiednie nagłówki takie jak ETag, Last-Modified oraz Cache-Control odpowiednio przygotowane pomagają wydajnie obsługiwać stronę www.

## **7. Badanie czy poprawa błędów ma przełożenie na lepszą wydajność serwisu**

Korzystanie z narzędzia Snail, które jest unikalnym rozwiązaniem wśród tego typu rozwiązań może dostarczyć programiście wystarczająco wiele danych, aby ten dokonał odpowiednich zmian w swoim serwisie webowym. Wiele projektów powstaje na zasadzie „Jeśli się wyświetla, to jest dobrze, po co przenosić te skrypty na koniec ?”. Bez zrozumienia tego, jak kod strony interpretuje przeglądarka internetowa, jak działają poszczególne skrypty, jak należy rozmieścić elementy w kodzie. W celu udowodnienia sensu budowania „świadomych” projektów internetowych wykonano dwie strony internetowe. Są to specjalnie przygotowane, na potrzeby analizy, dwie identycznie wyglądające strony WWW. Różnią się jednak pod względem kodu. Jedna z nich zawiera nieozptymalizowaną grafikę, a druga przeciwnie. Jedna zawiera błędy kodu HTML, a druga jest ich pozbawiona. Celowo oba szablony mają ten sam wygląd, aby udowodnić, że zmiany jakich można dokonać, celem zwiększenia wydajności i szybkości pobierania strony, nie wpływają na ich ostateczny wygląd.

Obie strony nie zawierają istotnej treści, ich zadaniem jest uzasadnienie sensu optymalizacji witryn internetowych, dlatego ich wygląd składa się z następujących części składowych:

- Wczytanie same frameworka Bootstrap *ver. 3.3*
- Zastosowanie wszystkich podstawowych stylów tego narzędzia na stronie w celu użycia plików CSS, które muszą zostać wczytane
- Wczytanie dużych plików obrazów, na których przedstawione są miejsca publiczne we Wrocławiu, jak Stadion Miejski, Rynek, czy jeden z budynków Politechniki Wrocławskiej. Natomiast w wersji zoptymalizowanej jest to użycie techniki „Sprites CSS” do wyświetlania zoptymalizowanych (przy użyciu narzędzia Adobe Photoshop) tych samych grafik.
- Mini pokaz slajdów, który wykorzystuje skrypt jQuery, oraz pliki JavaScript frameworka Bootstrap.
- Nieozptymalizowana strona zawiera także błędy w pliku *index.html*, które narzędzie Snail Project jest w stanie wykryć.



*Rysunek 7.1 Wygląd stron WWW wykonanych specjalnie do analizy błędów  
Źródło: Opracowanie własne*

Powyższy obrazek przedstawia wygląd przygotowanej specjalnie do przedstawienia analizy strony internetowej, która ze względu na swoją długość, została przedstawiona na obrazku w dwóch kolumnach.

„Projekt Slimak” po wykonaniu analizy obu wersji stron internetowych, zwrócił następujące wyniki.

Importance	Type
0	Avoid landing page redirects
0	Enable compression
4.5	Leverage browser caching
0	Reduce server response time
0.1357	Minify CSS
0.075	Minify HTML
0.2973	Minify JavaScript
10	Eliminate render-blocking JavaScript and CSS in above-the-fold content
1369.6194000000003	Optimize images
0	Prioritize visible content

Rysunek 7.2 Rezultat strony bez optymalizacji  
 Źródło: Opracowanie własne

Importance	Type
0	Avoid landing page redirects
0	Enable compression
4	Leverage browser caching
0	Reduce server response time
0	Minify CSS
0	Minify HTML
0	Minify JavaScript
6	Eliminate render-blocking JavaScript and CSS in above-the-fold content
0.4486000000000005	Optimize images
0	Prioritize visible content

Rysunek 7.3 Rezultat strony zoptymalizowanej  
 Źródło: Opracowanie własne

Jak widać na obrazkach z rezultatami, największą wagę należy zwrócić na optymalizację obrazków na stronie w wersji bez optymalizacji, ponieważ jest to główna przyczyna długiego wczytywania się strony. Na rysunku 7.3 również widać sugestię poprawy obrazków, ponieważ

zostały one zoptymalizowane przed połączeniem w jeden (zgodnie z zasadą CSS Sprites), a ten ostatecznie nie został poddany kolejnej optymalizacji.

Rozmiary plików CSS, HTML, oraz JavaScript zostały zmniejszone przy użyciu narzędzia „CodeKit”. Jego działanie polega na tworzeniu zminimalizowanej wersji plików, pozbawionej wszelkiego rodzaju formatowania. Takie kompresowanie polega na usunięciu zbędnych spacji, wcięć, komentarzy czy znaków nowej linii. Pełna wersja pliku Bootstrap ze stylami to rozmiar 136 KB natomiast zminimalizowany rozmiar to już 108 KB. Nie da się jednoznacznie stwierdzić, czy różnica jest duża czy mała. W przypadku pobierania tego elementu przez sieć 3G, różnica może być bardziej odczuwalna niż w przypadku LTE, natomiast kiedy tworzone są dużo większe serwisy, zawierające kilka tysięcy linii kodu, to po prostu warto zadbać o to, aby użytkownik (bez względu na rodzaj połączenia z Internetem) nie musiał pobierać zbędnych kilobajtów, skoro wynik działania jest taki sam.

Na stronie określonej jako „lżejsza” wczytywanie skryptów w kodzie HTML, następuje dopiero przed zamknięciem znacznika </body>, prawie na samym końcu pliku. Jak wspomniano we wcześniejszych rozdziałach zabieg ten ma na celu wyeliminowanie, bądź skrócenie czasu blokowania renderowania pozostałych elementów witryny.

**Found 5 problem(s) with the test site**

Line 30, Column 3:  
**Saw < when expecting an attribute name. Probable cause: Missing > immediately before.**

Line 30, Column 7:  
**End tag had attributes.**

Line 30, Column 13:  
**Stray end tag div.**

Line 32, Column 24:  
**An body start tag seen but an element of the same type was already open.**

Line 32, Column 24:  
**Cannot recover after last error. Any further errors will be ignored.**

Rysunek 7.4 Wynik walidacji kodu dla niezoptymalizowanej wersji strony testowej  
 Źródło: Opracowanie własne

Celowo zrobiono kilka błędów w kodzie „gorszej” witryny, które zostały prawidłowo wychwycone przez sprawdzający poprawność walidacji skrypt. Nie przedstawiono obrazka z wynikiem dla drugiej strony ponieważ liczba błędów wynosi tam po prostu zero. Odpowiednie znaczniki zostały domknięte, i wyeliminowano pozostałe błędy.

**16 requests, Total request 5506 bytes, Total response 17514637 bytes 48065 msec**

Rysunek 7.5 Rezultaty pobierania dla pełnej wersji strony testowej  
 Źródło: Opracowanie własne

**9 requests, Total request 3249 bytes, Total response 3043086 bytes 9754 msec**

Rysunek 7.6 Rezultaty testowania dla zoptymalizowanej strony testowej  
 Źródło: Opracowanie własne

Różnica wyników przedstawiona na dwóch powyższych grafikach wynika oczywiście ze względów na zastosowanie się do sugerowanych przez narzędzie Snail poprawek. Liczba zapytań zmniejszyła się ponieważ usunięto wczytywanie elementów strony, które celowo nie

istniały. Zmniejszono liczbę obrazków, ponieważ cztery zostały połączone w jeden. Ewentualne dodatkowe style CSS zostały umieszczone w pliku *bootstrap.min.css*, a nie osobno.

Jakość optymalizacji została oceniona przez Snail w stosunku 87% do zera, oczywiście dla zoptymalizowanej strony. Celem analizy nie było osiągnięcie 100% i pokazanie wydajnościowego kontrastu, tylko to, jaki wpływ mają działania optymalizacyjne na rezultat działania narzędzia oraz szybkość pobierania samej witryny.

Na rysunkach 7.5 oraz 7.6 widać czas jaki upłynął od wysłania pierwszego żądania do pobrania całej witryny przez skrypt testujący. Dla pełnej wersji strony było to aż 48 sekund!. W porównaniu do 9,7 sekundy drugiej strony to prawie pięciokrotna różnica.

## **8. Czy warto trzymać się wytycznych World Wide Web Consortium ?**

Jak wynika z badań przeprowadzonych przy użyciu narzędzia Snail, które jako jedyne wraz z testem wydajności serwisu potrafi dokonać analizy kody HTML pod kątem jego poprawności z wytycznymi W3C, niestety wśród najpopularniejszych Polskich serwisów webowych nikt nie dba o semantyczność kodu. Liczba błędów jakie zostały znalezione w serwisach waha się pomiędzy 20 (serwis Wikipedia.org) do ponad 1001 kiedy skrypt przestaje dalej analizować błędy (serwis Gazeta.pl). Są to jednak bardzo duże korporacje, czy zatem można stwierdzić, że „duży może więcej” ? Nie można jednoznacznie odpowiedzieć na to pytanie. Jest to problem ocierający się o problem *logiki rozmytej* [42]. Nie można jednoznacznie stwierdzić, że robienie błędów w kodzie jest bardzo złe w skutkach dla sieci Internet. Przykładowo badania przeprowadzone na specjalnie przygotowanej do testów strony internetowej zawierającej błędy (około 30), potwierdziły, że ich poprawa prawie, nie ma wpływu na szybkość pobierania się strony internetowej. Dlaczego prawie ? Część błędów zawierała niedomknięte znaczniki, bądź komentarze w deklaracjach klas. Usunięcie ich spowodowało zmniejszenie wagi pliku HTML, co przekłada się na realny wpływ wydajności pobierania strony. Ale Konsorcjum ściśle określa, jak „prawidłowo” używać HTML-a. Błądem jest brak wartości atrybutu „alt” w znaczniku <img>, co teoretycznie nie ma znaczenia podczas prawidłowego działania strony internetowej. Ma on sens dopiero gdy zawiedzie wczytywanie obrazka, lub dla obsługi programów „czytających” zawartość stron, które są ułatwieniem dla osób niedowidzących. Lecz dla osób niedosłyszących niesie jedynie informację o nazwie obrazka, który nie został prawidłowo pobrany. Jednak dla programistów, którym zależy na zwiększeniu wydajności serwisu, taka opcja nie musi wcale być potrzebna. Dlaczego ? Takie serwisy jak na przykład Youtube.com czy Spotify.com, nie powstały z myślą o osobach z wcześniej wspomnianymi schorzeniami. Czy w takim razie takie serwisy mogą sobie pozwolić na błędy w kodzie ? „Standardy W3C” to zbiór ogólnych wskazówek dla różnych producentów programów do korzystania z Internetu. Programów takich jak roboty sieciowe, przeglądarki internetowe, czy właśnie oprogramowanie służące łatwiejszej obsłudze sieci przez osoby z różnymi schorzeniami. Istnieje też druga strona medalu, jaką są webmasterzy. To nie prawda, że nie muszą się stosować do wskazówek. Sami producenci narzędzi webowych, którzy są zrzeszeni z W3C czasami nie przykładają zbyt bacznej uwagi co do standardów. Zapewne dlatego też każda przeglądarka WWW posiada własne style i za jej pomocą Internet wygląda „inaczej” niż podczas korzystania z konkurencyjnego rozwiązania.

Strona internetowa, która jest jednym wielkim obrazkiem, jest niezgodna ze standardami Organizacji. Czy jest ona źle wykonana ? – Technicznie.... Wcale nie. Być może, będzie to rozwiązanie czytelne dla odbiorców. Natomiast strona internetowa, której tło jest czerwone, czcionka rozmiaru 500 pikseli, powodująca uciążliwe przewijanie tekstu dodatkowo zlewającą się z tłem może być całkowicie zgodna z zasadami W3C. Wszystkie informacje specyfikacji należy traktować jako kodeks etyczny, który może być cennym zbiorem informacji dla webmasterów. Tak jak w każdej dziedzinie, życia należy kierować się rozsądkiem ale i dobrymi manierami. To samo dotyczy tworzenia strony internetowej. Warto stosować reguły publikowane przez Konsorcjum, tworząc prywatne strony WWW jak i wielkie serwisy, ponieważ każdy kod serwisu powinien być pisany w taki sposób, aby był w stanie zapewnić prawidłowe wyświetlanie w jak największej liczbie przeglądarek internetowych.

## **9. Czy wprowadzenie SPDY oraz HTTP 2.0 przyniesie zakładany wzrost wydajności ?**

Rosnąca waga stron internetowych spowodowana rozwojem sieci, języków programowania i wymaganiami samych użytkowników, powoduje „zapychanie się” Internetu. Więcej danych do przesłania to dłuższy czas oczekiwania. Rozwój sieci jednak nie stoi w miejscu.

Jak wspomniano w poprzednich rozdziałach, SPDY nie jest traktowany jako nowy standard sieciowy. Jest bardziej zbiorem rozwiązań firmy Google na problemy wydajności Internetu. Na początku roku 2015 wersją jaką można było ”testować” był SPDY w wersji 4. Przeglądarka internetowa Chrome 40.0.2 nie obsługuje tej technologii domyślnie. Trzeba ją ręcznie uruchomić korzystając z „*chrome://flags/*”. Rozwój HTTP 2.0 oraz SPDY nie jest traktowany jako różne technologie. Docelowo oba rozwiązania mają tworzyć jeden protokół sieciowy o nazwie HTTP 2.0.

TCP stosuje algorytm nazywany „Slow Start”. Działa on na takiej zasadzie, że w trakcie połączenia, prędkość przesyłania danych wzrasta, jednak każda strata pakietu powoduje ponowne zmniejszenie prędkości połączenia i jego stopniowe przyspieszanie. Taki algorytm został zaimplementowany w obawie przed przepełnieniem Internetu.

Protokół HTTP 2.0 implementujący SPDY, nie rozwiązuje powyższego problemu TCP. Jedynie proponuje transmisję danych przez protokół SSL/TLS, warstwy aplikacji (w celu ochrony prywatności), co jest związane z dodatkowymi czasami RTT.

Server Push to nowe rozwiązanie opisywanych protokołów, które zakłada, że serwer może „przewidywać” jakie dane będą kolejno pobierane przez klienta i podsypać je bez potrzeby nawiązywania dodatkowych żądań. Jeśli to rozwiązanie się sprawdzi, może być szczególnie cenne przez użytkowników, których sieci cechują się dużymi opóźnieniami czyli wysokim RTT.

Nowe rozwiązanie zakłada wiele innych zmian, jak np. jeszcze lepiej przemyślana kompresja nagłówków, wysyłanie wielu żądań bez oczekiwania na odpowiedź poprzednich (multipleksowanie połączeń HTTP podczas jednego połączenia TCP) itp.

Guy Podjarny jest pracownikiem firmy Akamai, który sprawdził jak obecnie wdrożone w niektórych przeglądarkach i serwerach (do serwerów Apache czy Nginx, można użyć specjalnych modułów, umożliwiających korzystanie ze SPDY oraz HTTP w wersji 2), wpływa na szybkość przesyłania danych. Oto rezultat tych badań:

Network Speed (Down/Up Kbps, Latency ms)	SPDY vs HTTPS	SPDY vs HTTP
Cable (5,000/1,000, 28)	SPDY 6.7% faster	SPDY 4.3% slower
DSL (1,500/384,50)	SPDY 4.4% faster	SPDY 0.7% slower
Low-Latency Mobile (780/330,50)	SPDY 3% faster	SPDY 3.4% slower
High-Latency Mobile (780/330,200)	SPDY 3.7% faster	SPDY 4.8% slower

Rysunek 9.1 Wpływ użycia SPDY na szybkość pobierania stron WWW  
 Źródło: <https://twitter.com/guypod>

Podjarny, przeprowadził badania pod koniec 2013 roku i opublikował je na swoim profilu [43] za pośrednictwem serwisu Twitter. Z wyników można wywnioskować, że (wtedy jeszcze wersja SPDY 3) działała wolniej podczas połączeń nie wykorzystujących szyfrowanego połączenia. Jeśli zgodnie z założeniami cały następca HTTP 1.1 będzie oparty o SPDY, to w pewnych warunkach może wcale nie przynieść zakładanego wzrostu wydajności sieci. Testy należałyby powtórzyć, lecz dopiero w chwili gdy twórcy udostępnią „wersje alfa lub beta” swojego dzieła, i wtedy może to być dobry czas na krytykę i ewentualne zmiany podejścia do problemu wydajności w Internecie.

Mimo wielu lat pracy nad rozwojem protokołów sieciowych, opisywane zagadnienia, wciąż mają jedynie dobre założenia. Wprowadzenie specjalnych modułów, czy dodatków umożliwiających korzystanie z testowych wersji tych technologii, nie pozwala w pełni odczuć racjonalnych zmian. Do wszystkiego można dojść małymi krokami. Dlatego przed wdrożeniem w pełni funkcjonalnego protokołu sieci „nowej generacji” jakim prawdopodobnie będzie HTTP 2.0 może upływać jeszcze wiele miesięcy badań i testów.

## **10.Podsumowanie**

Sieć staje się coraz bardziej obszerna i pojemna. Rozrasta się na tyle, że zarazem coraz trudniej znaleźć dobre informacje. Czasami trzeba wręcz przejrzeć wiele stron internetowych by trafić na tą, gdzie informacje będą miały jakąś wartość. Gdyby wszystkie strony w Internecie (1,175,383,770 dnia 1 stycznia 2015 roku) [40], zostały odpowiednio przygotowane pod względem szybkości pobierania to czas wyszukiwania potrzebnych informacji mógłby zostać znaczająco skrócony. Mogłoby się to przełożyć na krótszy czas pracy lub przeciwnie, większy zysk spowodowany wydajnością pracowników.

Użytkownicy Internetu są coraz bardziej wymagający. Doświadczenia użytkownika odgrywają znaczącą rolę przy tworzeniu serwisów webowych. Dziś amatorzy sieci wymagają by strony, które odwiedzają były atrakcyjne wizualnie, przyciągały wzrok oraz zawierały różne multimedia. Nikogo już nie interesują „białe” strony, zapełnione treścią.... Łączenie obrazu, dźwięku oraz tekstu w jednym miejscu jest bardzo atrakcyjną formą przekazywania danych użytkownikowi. Sposób ten już dawno przegonił radio i być może telewizję.

Za ewolucją stron i doświadczeniami użytkowników idą także problemy. Najpoważniejszym jest oczywiście wydajność takich rozwiązań. Przekolorowane i obładowane grafiką miejsca w sieci, do których dostępu wymaga jednocześnie kilka milionów odbiorców, nie zawsze spełniają dobrze swoją rolę.

Aby Internet nie stał się jarmakiem, gdzie można znaleźć zarówno „perełki” jak i mnóstwo niepotrzebnych nikomu przedmiotów, należy dbać o proces doskonalenia i ulepszania istniejących obecnie serwisów webowych. Dzięki narzędziom takim jak Snail Project, które zawiera najważniejsze opcje potrzebne do przetestowania swojej strony WWW, można wyciągnąć wnioski jak i postawić sobie cele poprawy portalu. Natomiast podczas implementacji nowych rozwiązań, należy pamiętać o złotych regułach wydajności, które po zaprogramowaniu, również można poddać analizie wykorzystując ponownie Snail, który oceni jakość powstałego serwisu.

## **Rezultat pracy**

Wynikiem prac nad badaniem jakości samych stron internetowych jak i szybkości ich pobierania jest narzędzie o nazwie Snail Project oraz niniejsza praca magisterska, w której zostały opisane wyniki badania stanu obecnych serwisów webowych, analiza niektórych z nich, oraz wnioski wyciągnięte z dokonanych badań.

Narzędzie Snail spełniło nie tylko początkowo zakładane wymagania (testowanie szybkości pobierania stron internetowych i generowanie wykresu pobieranych elementów drzewa DOM). „Slimak” sprawdza także skanowany dokument pod kątem błędów w kodzie. Wystawia także ocenę stronie, która jest stopniem zoptymalizowania testowanej witryny. Dodatkowo sprawdza najważniejsze założenia, które według badań firmy Google wpływają najbardziej na szybkość działania i pobierania strony internetowej. Projekt potrafi także sprawdzić aktualnie używaną przez użytkownika przeglądarkę internetową pod kątem obsługi funkcji takich technologii jak HTML 5 oraz CSS 3.

## **Propozycje kontynuacji i wizja rozwoju**

Kolejne lata ewolucji Internetu zapewne w końcu przyniosą całkowite wdrożenie protokołów sieciowych (nie tylko HTTP 2.0 ale i kolejnych), być może większą popularyzację rozszerzenia sieciowego jakim jest SPDY. To jednak nie wszystko, jako pierwsza powinna się rozstrzygnąć walka o dominację pomiędzy technologiami HTML 5 oraz Flash. Według obserwacji sposobu wykonywania stron internetowych, aplikacji etc. nadjejdzie kres Flasha. Najnowsze obecnie wydanie hipertekstowego języka znaczników oferuje potężne możliwości tworzenia nowoczesnych szablonów, które bez dodatkowych wtyczek obsługują dźwięki czy wideo. Szybkość działania i prostota tej technologii jest niewątpliwie potężnym orężem w tej walce.

Samo narzędzie, które powstało równolegle do tekstu pracy magisterskiej, mogłoby implementować kilka dodatkowych rozwiązań. Najciekawszym byłaby możliwość badania mobilnych stron WWW.

Dominacja urządzeń mobilnych powinna też zmotywować następców narzędzia Snail do tworzenia mobilnych rozwiązań do testów innych serwisów webowych. Mogłoby to przynieść ciekawe efekty skracając czas działania niektórych obecnie wykorzystywanych skryptów (kod kompilowany jest wciąż dużo szybszym rozwiązaniem niż kod interpretowany).

Implementacja piątego skryptu badającego pozycję danej strony internetowej w wyszukiwarce może przynieść ciekawe rezultaty badań. Taki skrypt powinien badać „łączność semantyczną” [42]. To znaczy, że należy analizować treść zamieszczoną na danej stronie WWW pod kątem różnych fraz, które łączą się w skojarzenia, i są one znaczące dla wyników wyszukiwania. Dodatkowa implementacja, naturalnie powinna także analizować, czy wprowadzanie zmian sugerowanych przez poprzednie skrypty ma wpływ na pozycję w wyszukiwarce, ewentualnie budowa wykresu jak następowała zmiana pozycji w czasie np. w ciągu roku.

Obecnie nazwa „strony internetowe” ewoluowała w „serwisy internetowe”, następnie „portale” „aplikacje webowe”, a dzisiaj już nawet granice między nimi wszystkimi się zacierają. Nie ma jednego kryterium określającego czy dany obiekt jest stroną, czy aplikacją zamieszczoną w Internecie. Poprzez mnogość wykorzystywanych rozwiązań do tworzenia portali takich jak np. Facebook użytkownik jest ograniczony do rejestracji w danej usłudze a następnie może z niej korzystać wręcz wprost z kieszeni.

Za kilka, kilkanaście lat możliwe, że już odejdzie się od pojęcia „strona WWW”. Właściwie wszystko będzie można zastąpić natywnymi aplikacjami. Wymagania dzisiejszych użytkowników sieci są na tyle skomplikowane, że prawie każdy nowy „startup” korzysta z jakiejś bazy danych, kilku skryptów, i innych wcześniej wymienianych technologii. A gdyby tak aplikacje (prócz urządzeń mobilnych) były uruchamiane z serwerów na przykład firmy Google ? Skompilowanie aplikacji *Gmail*, (która obecnie i tak korzysta z bazy danych własnej firmy) i z dedykowanego serwera udostępnianie miliardom użytkowników tylko specjalnego ekranu umożliwiającego korzystanie z aplikacji ? Niesamowita prędkość działania nie wymagająca HTML, CSS, JavaScript i tym podobnych.

Wszystkie rozwiązania przyszłości i wizje działania szybkich serwisów webowych bez oczekiwania na pobranie, niezależnie od jakości wykorzystywanego połączenia z Internetem są ograniczone jedynie ludzką wyobraźnią.

## Literatura

---

- [1] <http://validator.w3.org/>
- [2] <http://httpd.apache.org/>
- [3] <http://nginx.com/>
- [4] <http://www.w3.org/Protocols/HTTP/HTRESP.html>
- [5] <http://en.wikipedia.org/wiki/HTTP/2>
- [6] <http://www.stevesouders.com/blog/2012/02/10/the-performance-golden-rule/>
- [7] [http://pl.wikipedia.org/wiki/Twierdzenie\\_Shannona-Hartleya](http://pl.wikipedia.org/wiki/Twierdzenie_Shannona-Hartleya)
- [8] <https://developers.google.com/speed/docs/insights/mobile?hl=pl>
- [9] [http://www.huawei.eu/files/publications/pdf/huawei\\_5g\\_white\\_paper\\_en\\_20140129.pdf](http://www.huawei.eu/files/publications/pdf/huawei_5g_white_paper_en_20140129.pdf)
- [10] <http://www.ericsson.com/res/docs/whitepapers/wp-5g.pdf>
- [11] <http://www.komputerswiat.pl/nowosci/internet/2014/25/ue-i-korea-poludniowa-beda-wspolnie-rozwijac-internet-5g.aspx>
- [12] Vossen G., Hageman S., *Serwis WEB 2.0*, Gliwice 2010, Helion
- [13] <http://modernizr.com/>
- [14] <http://developers.google.com/speed/pagespeed/insights>
- [15] <http://gtmetrix.com>
- [16] <http://loadimpact.com>
- [17] <http://www.webpagetest.org/>
- [18] <http://analyze.websiteoptimization.com/>
- [19] <http://www.neustar.biz/resources/tools/free-website-performance-test>
- [20] <http://tools.pingdom.com/fpt/>
- [21] <http://www.alexa.com/topsites/countries/PL>
- [22] [http://en.wikipedia.org/wiki/Google\\_platform](http://en.wikipedia.org/wiki/Google_platform)
- [23] Souders S., *Wydajne witryny internetowe. Przyspieszanie działania serwisów WWW*, Gliwice 2008, Helion
- [24] Sharkie C., Fisher A., *Responsywne strony WWW*, Gliwice 2013, Helion
- [25] <http://www.gimp.org/>
- [26] <http://www.adobe.com/pl/products/photoshop.html>
- [27] [http://www.w3.org/standards/techs/webfonts#w3c\\_all](http://www.w3.org/standards/techs/webfonts#w3c_all)
- [28] <http://webhosting.pl/Mozilla.odrzucila.obslugę.formatu.WebP>

- 
- [29] <https://developers.google.com/speed/webp/>
  - [30] <https://developers.google.com/speed/webp/download>
  - [31] <http://www.logotyp.pwr.edu.pl/Default.aspx?page=ZnakPWr>
  - [32] <https://developers.google.com/speed/webp/docs/using>
  - [33] Souders S., *Jeszcze wydajne witryny internetowe. Przyspieszanie działania serwisów WWW*, Gliwice 2010, Helion
  - [34] <http://www.akamai.com/html/solutions/network-operator-solutions.html>
  - [35] <http://www.bootstrapcdncdn.com/>
  - [36] <https://www.cloudflare.com/features-cdn>
  - [37] [http://pl.wikipedia.org/wiki/User\\_experience](http://pl.wikipedia.org/wiki/User_experience)
  - [38] <https://www.igvita.com/posa/high-performance-networking-in-google-chrome/>
  - [39] <https://developer.mozilla.org/pl/docs/XMLHttpRequest>
  - [40] <http://www.internetlivestats.com/total-number-of-websites/>
  - [41] <http://www.easel.ly/>
  - [42] Enge E., Spencer S., Stricchiola J., Fishkin R., *Sztuka SEO Optymalizacja witryn internetowych*, Gliwice 2013, Helion
  - [43] <https://twitter.com/guypod>