Przykładowy raport z audytu **wydajności frontendu** strony WWW

STUDIO SIDEKICKS X Bigger Picture X WEBDEV INSIDER

2022 Studio Sidekicks Sp. z o.o.

Uwaga!

W dniach **27 - 30. czerwca 2022** odbywa się nabór do kursu Zoptymalizowany Frontend.

Ostatnia szansa przed dłuższą wakacyjną przerwą.

Niniejszy dokument został przygotowany w celach podglądowych na bazie rzeczywistego raportu. Linki i fragmenty kodu zostały usunięte.

1. Wprowadzenie

Firma XYZ zwróciła się na początku stycznia 2022 r. do Web Dev Insider z prośbą o wykonanie audytu wydajności frontendu jednej ze stron produktowej, znajdującej się pod adresem …… Audyt został przeprowadzony w dniach 10.01 - 24.01.2022.

Wydajność frontendu określa jakość strony, a ta z kolei jest składową:
- szybkości ładowania na różnych typach urządzeń (np. smartfon, tablet, laptop/komputer stacjonarny) z uwzględnieniem różnej jakości połączenia internetowego, skupiając się przede wszystkim na średniej szybkości internetu mobilnego tj. 3G Fast (1.6 Mbps/768 Kbps 150ms RTT) oraz tzw. średnim urządzeniu (Moto G4);
- odczuć związanych z ładowaniem i wygodą korzystania ze strony (tzw. postrzegalna wydajność, poziom UX);
- spełnienia wymagań względem SEO oraz jej bezbarierowego użytkowania wśród każdej grupy internautów (Accessibility).

Wiele badań wskazuje na to, że wydajność stron WWW ma bezpośrednie przełożenie na UX, zadowolenie użytkowników, a tym samym rezultaty finansowe z punktu widzenia biznesowego. Szybka strona dostarcza pozytywnych wrażeń internautom, pomagając zrealizować ich cele w sposób zdecydowany i konkretny - szybciej otrzymują to, czego oczekują. Są również bardziej zaangażowani treścią znajdującą się na stronach, a także bardziej skłonni do podejmowania oczekiwanych - z punktu widzenia biznesu - decyzji, np. zakup produktu, podjęcie kontaktu bezpośredniego, wypełnienie formularza. Szybka, zoptymalizowana strona buduje pozytywny obraz firmy, jaka stoi za daną stroną, a tym samym przyczynia się do obdarzenia jej zaufaniem. Warto wspomnieć o wątku ekologicznym - rezultatem zoptymalizowanej strony jest mniejsze zużycie zasobów serwerowych, a także niższy transfer danych.

Wydajność strony WWW określa jej jakość, a ta z kolei od 2021 stała się również jednym z kluczowych parametrów wpływających na pozycję strony w wynikach wyszukiwania Google.

Metryki, które są szczególnie istotne z punktu widzenia określenia poziomu wydajności strony:
- TTFB
- Start Render
- First Contentful Paint
- LCP
- TTI
- TBT
- FID
- CLS
- liczba ładowanych zasobów
- liczba zasobów blokujących renderowanie strony

2022 Studio Sidekicks Sp. z o.o.

2. Kluczowe rekomendacje, dzięki którym strona WWW jest wydajna

Poniższa lista prezentuje **najważniejsze składowe** strony, dzięki którym jest wydajna i przyjazna użytkownikom:

- niski TTFB (odpowiedź z backendu/serwera; TCP/TLS handshake);
- krótki czas, potrzebny, by wyrenderować w przestrzeni Above the Fold najpierw jakikolwiek element layoutu (Start Render, następnie First Contentful Paint), a potem jak najszybciej najważniejsze strategiczne elementy strony - np. logo, zdjęcie promocyjne, nagłówek, tekst (Largest Contentful Paint); cel to jak najszybsze załadowanie elementów na górze strony, by pomimo dalszego ładowania innych zasobów na stronie, móc zainteresować użytkownika pierwszymi treściami w jak najkrótszym czasie;
- jak najmniejszy rozmiar pobieranych plików, a zwłaszcza tych, które są odpowiedzialne za wizualne wyrenderowanie widocznych dla oka elementów (np. HTML, obrazków, plików CSS/JS);
- Zachowana mała waga plików, zwłaszcza tych odpowiedzialnych za przestrzeń Above the Fold, z uwagi na „zasadę 14kb";
- jak najmniejsza liczba ładowanych zasobów;
- odpowiednia kolejność ładowanych zasobów z uwagi na ich rodzaj i znaczenie w procesie ładowania i renderowania strony;
- ładowanie tylko tylu zasobów na stronie, ile jest potrzebnych w danym momencie wizycie dla użytkownika (tzw. Lazy loading);
- zadbanie o zarezerwowanie przestrzeni dla elementów, które doładowują się na żądanie/życzenie - w miarę, kiedy powinny być widoczne na ekranie dla użytkownika;
- ograniczenie lub umiejętne ładowanie zasobów zewnętrznych (tzw. 3rd parties - potrzebne np. do analityki ruchu na stronie, widget czatu itd.);
- poprawnie napisany kod animacji, styli CSS, HTML oraz Javascript, mający przełożenie na czas renderowania layoutu, płynności korzystania ze strony (tj. interakcji), a także zużycia baterii urządzenia;
- zadbanie o dostosowywanie się wyglądu strony do rodzaju i rozmiaru urządzenia, na którym jest wyświetlana;
- Kod HTML musi być poprawny strukturalnie i semantycznie;
- Wyniki metryk Google Web Vitals powinny **mniej więcej** mieścić się w wyznaczonych granicach według specyfikacji Google poniżej:

2022 Studio Sidekicks Sp. z o.o.

3. Cel audytu

Celem testowania wydajności frontendu strony ……. jest przedstawienie rezultatów badań, ich analizy, a także rekomendacji technicznych, dzięki którym możliwe będzie udoskonalenie jakości wspomnianej strony. Głównym priorytetem jest późniejsza możliwość wdrożenia poprawek wydajnościowych w oparciu o niniejszy dokument, skupiając się przede wszystkim na poprawie szybkości ładowania wspomnianej strony. Dodatkowym „efektem ubocznym" przeprowadzonych optymalizacji powinny być udoskonalone czasy/pomiary metryk Core Web Vitals oraz potencjalna poprawa pozycji strony w wynikach wyszukiwania Google.

Warto podkreślić, że poprawa jakości strony WWW powinna w głównej mierze przełożyć się na **udoskonalenie rzeczywistej, postrzegalnej wydajności strony.**

4. Metodologia badania

Badanie wydajności frontendu oraz stworzenie rekomendacji technicznych zostało przeprowadzone w oparciu o kilka etapów:

- automatyczne testowanie szybkości ładowania w okresie 02.01 - 10.01.2022 przy pomocy narzędzia SpeedCurve (testy syntetyczne);
- wywiad wśród grupy 10 osób, prosząc o opinie na temat postrzegania szybkości ładowania strony;
- manualne testowanie szybkości ładowania oraz renderowania przy użyciu narzędzi WebPageTest, Lighthouse oraz DevTools + zauważenie problemów;
- nanoszenie odpowiednich zmian w kodzie w środowisku testowym (przy wykorzystaniu narzędzia DevTools - Local Overrides, dzięki któremu możliwe było testowanie strony produkcyjnej z uwzględnieniem zmian w kodzie w czasie rzeczywistym);
- opracowanie finalnych rekomendacji technicznych wraz z dowodem na ich działanie.

2022 Studio Sidekicks Sp. z o.o.

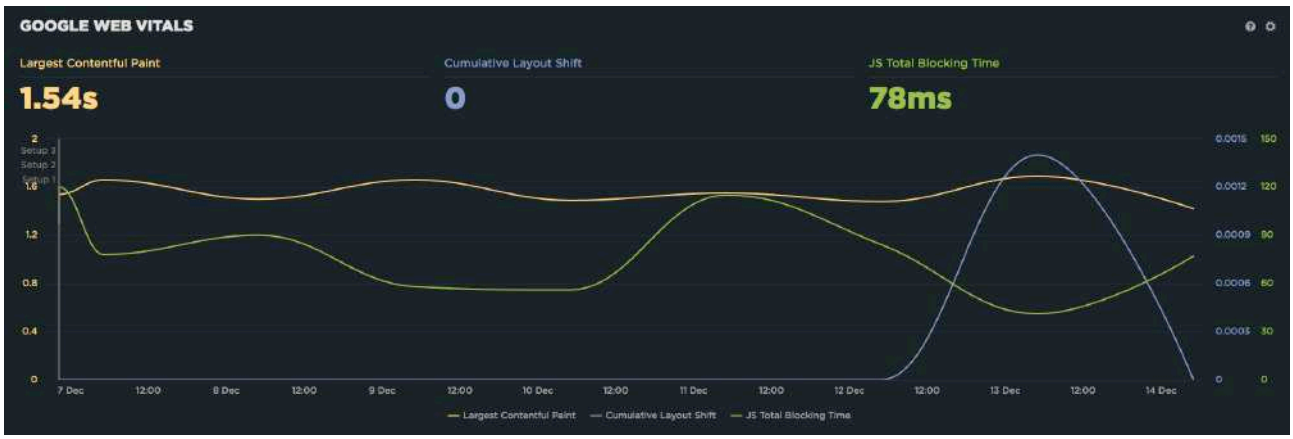
5. Automatyczne testowanie strony

Poniższe wykresy pochodzą z narzędzia SpeedCurve i przedstawiają wyniki pomiarów wydajności w oparciu o najważniejsze metryki. Dane pochodzą z okresu 02.01 - 10.01.2022. Testy były oparte na urządzeniu iPhone X, na połączeniu 4G, z Niemiec. Celem zautomatyzowanego badania jest jedynie stwierdzenie, czy strona prezentuje stabilny poziom wydajności, stałe czasy ładowania zasobów i czy nie jest możliwe zaobserwowanie pewnych anomalii w wyznaczonym okresie badania.

1. Czasy renderowania

Element **Last Painted Hero** to okienko z informacją o plikach Cookies. Podczas badania strony okazało się, że wspomniany element jest ładowany za pośrednictwem zewnętrznego zasobu managera tagów (z zewnętrznego serwera) i najprawdopodobniej dłuższe czasy serwerowe tego dostawcy spowodowały nieznaczne odchylenia od normy na poniższym wykresie, w niektórych dniach.

2. Google Web Vitals

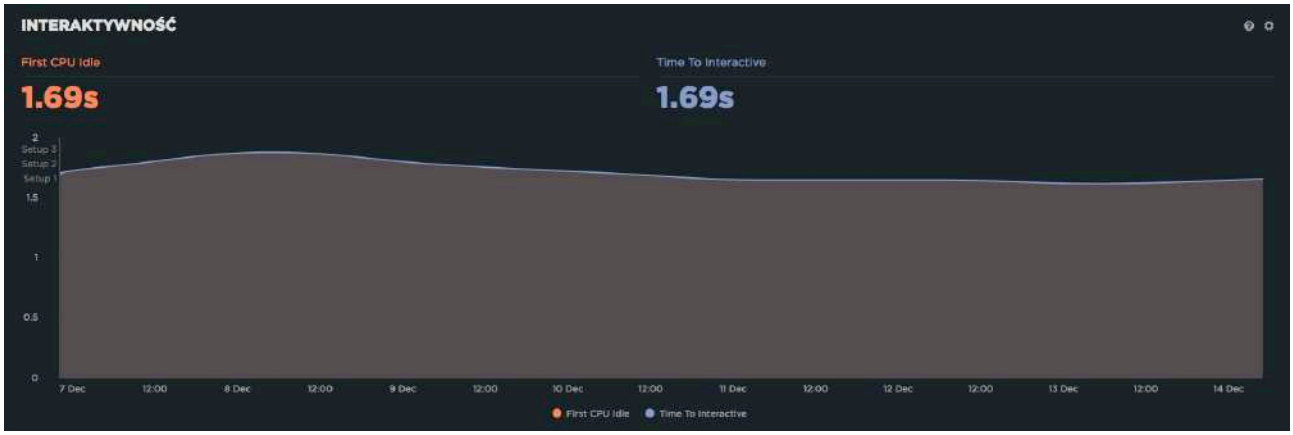


2022 Studio Sidekicks Sp. z o.o.

3. Wyniki Lighthouse

4. Czas ładowania strony

5. Interaktywność



2022 Studio Sidekicks Sp. z o.o.

6. CPU Time

7. Blokujące zasoby

8. Liczba ładowanych zasobów na stronie

2022 Studio Sidekicks Sp. z o.o.

5.1. Automatyczne testowanie strony - wnioski

Na podstawie zgromadzonych danych można wysnuć wniosek, że **strona prezentuje stabilny i w miarę jednakowy poziom wydajności**. Nieznaczne odchylenia od średnich wyników strony (w niektórych dniach w wyznaczonym okresie czasu) są nadal w granicach normy i nie powinny dawać powodów do niepokoju. Najprawdopodobniej są one wynikiem gorszych czasów zwracania odpowiedzi z zewnętrznych serwerów, do których strona produktowa się łączy (tag manager).

Na uwagę zasługuje fakt, iż w trakcie testowania strony, nastąpiła zmiana w kodzie strony (na przełomie 11 i 12 stycznia 2022), dzięki której zmniejszyła się liczba ładowanych obrazków, a także liczba blokujących renderowanie arkuszy styli CSS (z 3 do 2). **Jest to zmiana pozytywna.**

2022 Studio Sidekicks Sp. z o.o.

6. Opinie rzeczywistych użytkowników strony

W trakcie trwania audytu, przepytanych zostało 10 wybranych osób z zespołu Studio Sidekicks (były to również osoby, które na co dzień nie pracują na stanowiskach programistycznych), celem stworzenia obrazu postrzegalnej wydajności badanej strony. Testy były przeprowadzane przede wszystkich na urządzeniach mobilnych z mobilnym internetem o szybkości 3G lub LTE, a także komputerach/laptopach o stałym połączeniu internetowym.

Zdecydowana większość określiła poziom wydajności strony na urządzeniach mobilnych jako **umiarkowany**. Pozostali ankietowani wydali opinię jako „**dobry**”.

Na gorszym połączeniu internetowym denerwował dość długi czasu oczekiwania na wyrenderowanie jakiegokolwiek elementu na stronie w przestrzeni Above the Fold.
Część użytkowników jednogłośnie stwierdziło, że strona ładuje się niestabilnie - przez krótki czas widoczny jest inny element na stronie, zasłaniający główny banner, a gdy oczom ukazuje się górne zdjęcie, strona zostaje „przykryta” warstwą z informacjami na temat plików cookies, uniemożliwiając interakcję na stronie.

Osoby z większym poczuciem estetyki, zauważyli, że przycisk MENU jest niepoprawnie wycentrowany względem logo, a po jego kliknięciu elementy nawigacyjne zasłaniają ów przycisk, by je zamknąć.

Część użytkowników zwróciła uwagę na długo ładujące się zdjęcie w hero bannerze (w przestrzeni Above the Fold).

Użytkownicy laptopów/komputerów stacjonarnych w większości nie mieli zastrzeżeń co do ładowania i użytkowania strony. Pojawiły się sporadyczne głosy na temat poczucia ładowania strony od nowa wraz z każdą wizytą.

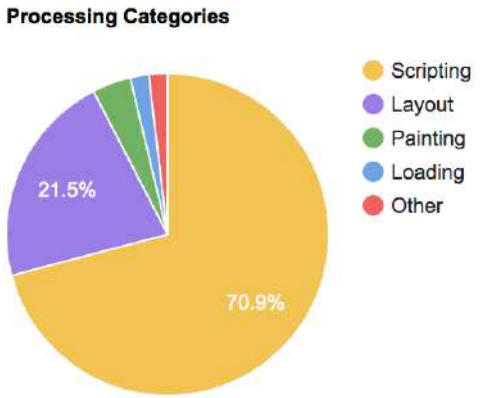
2022 Studio Sidekicks Sp. z o.o.

7. Testowanie i wyniki oryginalnej strony

7.1 Urządzenie mobilne, mobilny internet

Poniższe wyniki testów syntetycznych pochodzą z narzędzia WebPageTest, z następującymi parametrami:
- **lokacja:** Frankfurt, Niemcy
- **przeglądarka:** Chrome, na emulowanym urządzeniu Motorola G4
- **połączenie internetowe:** 3G Fast (1.6Mbps/768 Kbps 150ms RTT)
- **URL strony: ……**

					Web Vitals			Document Complete			Fully Loaded			
	First Byte	Start Render	First Contentful Paint	Speed Index	Largest Contentful Paint	Cumulative Layout Shift	Total Blocking Time	Time	Requests	Bytes In	Time	Requests	Bytes In	Cost
First View (Run 1)	1.414s	2.486s	2.420s	4.294s	4.773s	0.002	≥ 1.188s	7.875s	49	417 KB	8.097s	50	418 KB	$----
Repeat View (Run 2)	0.894s	2.000s	1.984s	3.463s	3.753s	0.003	≥ 1.221s	6.463s	38	312 KB	6.690s	39	313 KB	



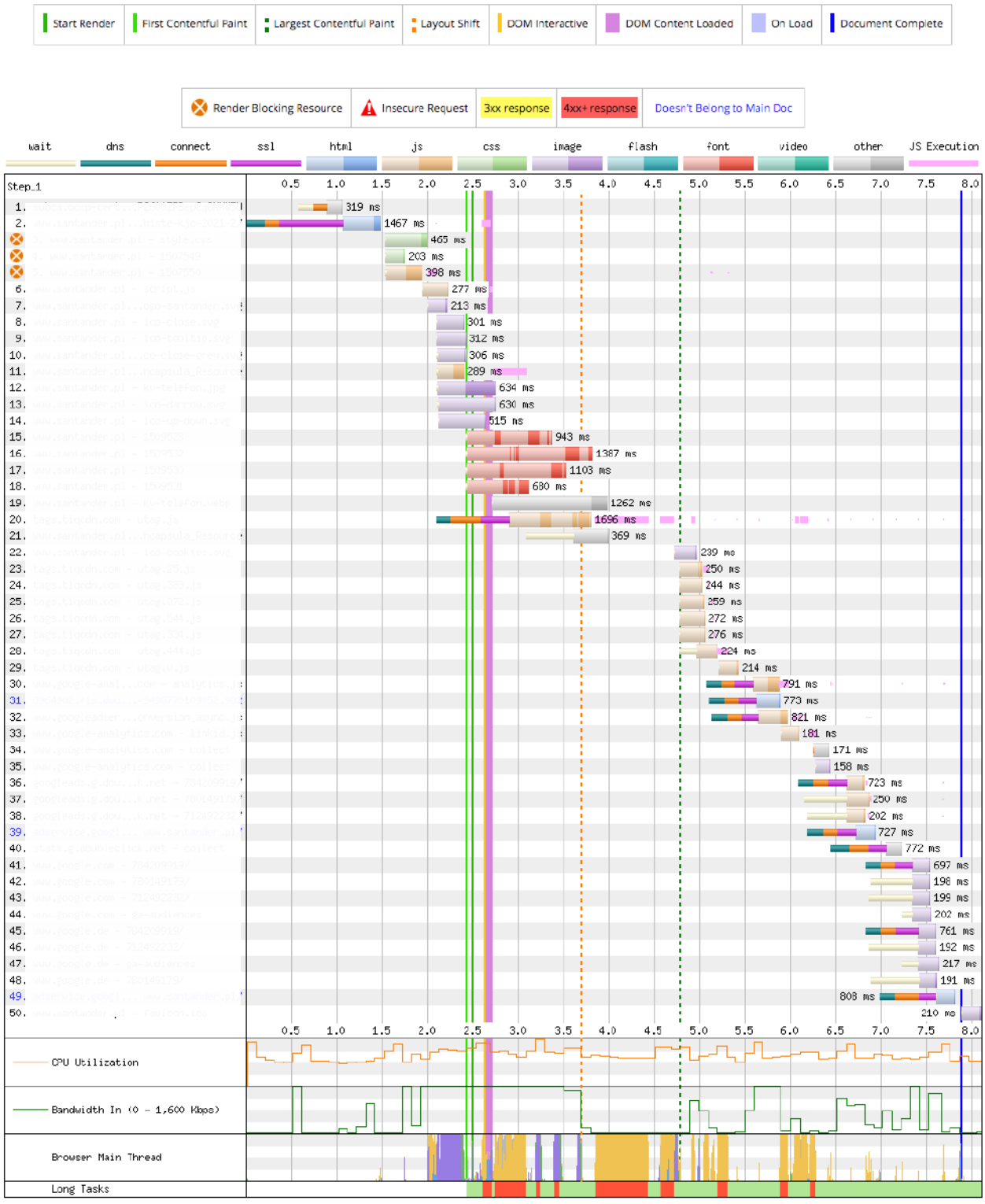
Pełny raport: link do raportu WebPageTest…

Wideo przedstawiające ładowanie strony: Link do nagrania wideo WebPageTest…

Warto uświadomić sobie jak wygląda proces ładowania i tym samym renderowania strony dla użytkownika końcowego na testowanym urządzeniu i w założonym środowisku (lokacja, szybkość internetu), w oparciu o powyższe nagranie wideo.

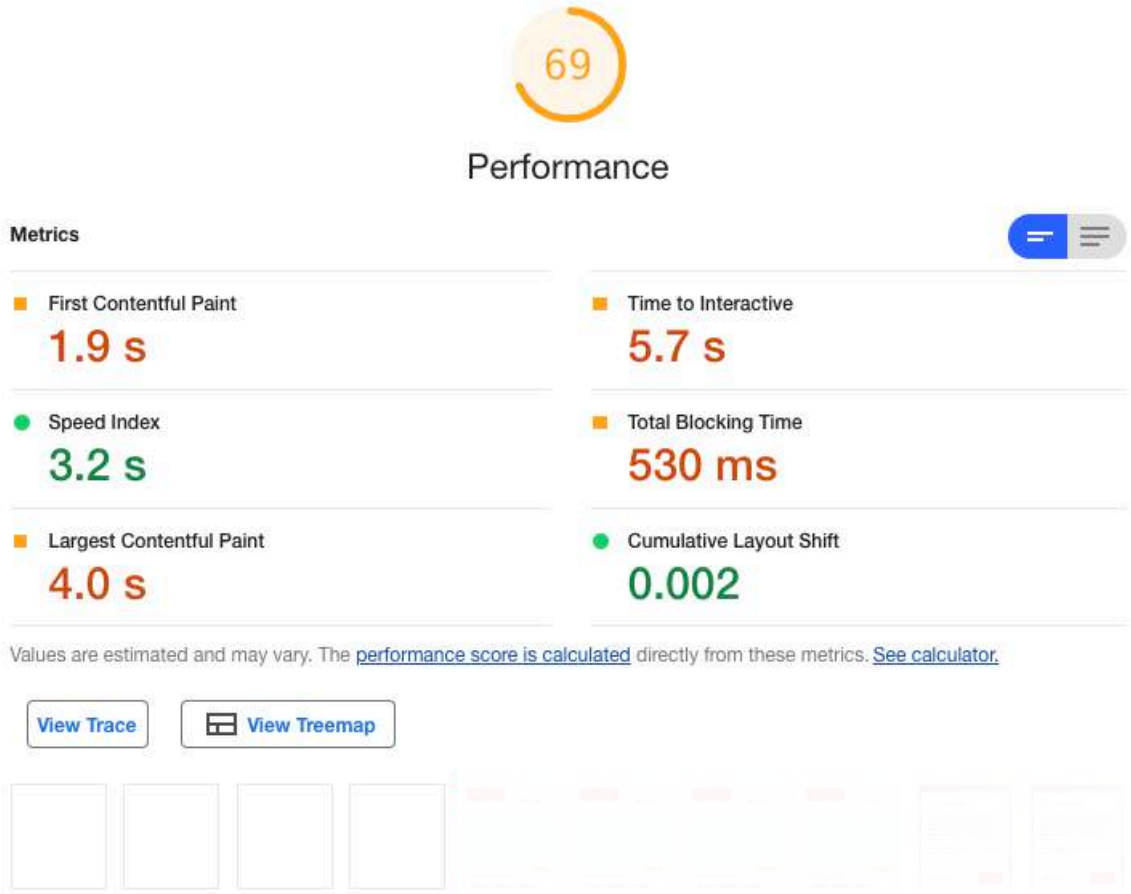
2022 Studio Sidekicks Sp. z o.o.

Waterfall zasobów



2022 Studio Sidekicks Sp. z o.o.

Poniższe wyniki testu pochodzą z narzędzia Lighthouse, uruchamianego w przeglądarce Chrome, emulując urządzenie mobilne i mobilny internet. Warto zauważyć, że walidacja certyfikatu SSL z serwerem OSCP nie występuje w tym teście z powodu jego zapamiętania w pamięci podręcznej cache, stąd też wyniki testu są znacznie lepsze względem testów na WebPageTest lub Pagespeed Insights.



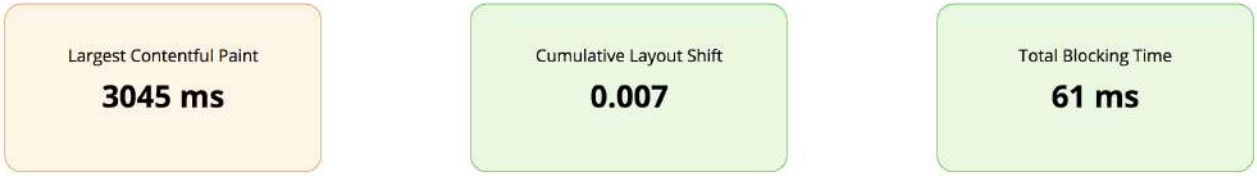
2022 Studio Sidekicks Sp. z o.o.

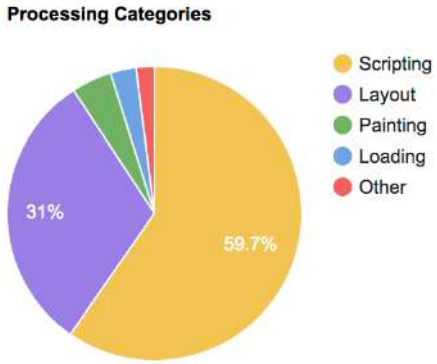
7.2 Desktop, połączenie 4G

Poniższe wyniki testów syntetycznych pochodzą z narzędzia WebPageTest, z następującymi parametrami:

- **lokacja:** Frankfurt, Niemcy
- **przeglądarka:** Chrome, desktop, rozdzielczość 1366x768px
- **połączenie internetowe:** 4G (9 Mbps, 170ms RTT)
- **URL strony:** ……

					Web Vitals			Document Complete			Fully Loaded			
	First Byte	Start Render	First Contentful Paint	Speed Index	Largest Contentful Paint	Cumulative Layout Shift	Total Blocking Time	Time	Requests	Bytes In	Time	Requests	Bytes In	Cost
First View (Run 1)	1.493s	2.100s	2.119s	3.017s	3.045s	0.007	≥ 0.061s	3.111s	21	501 KB	5.815s	51	562 KB	$$---
Repeat View (Run 2)	0.987s	1.700s	1.758s	2.172s	2.517s	0.007	≥ 0.067s	4.061s	38	457 KB	4.061s	38	457 KB	





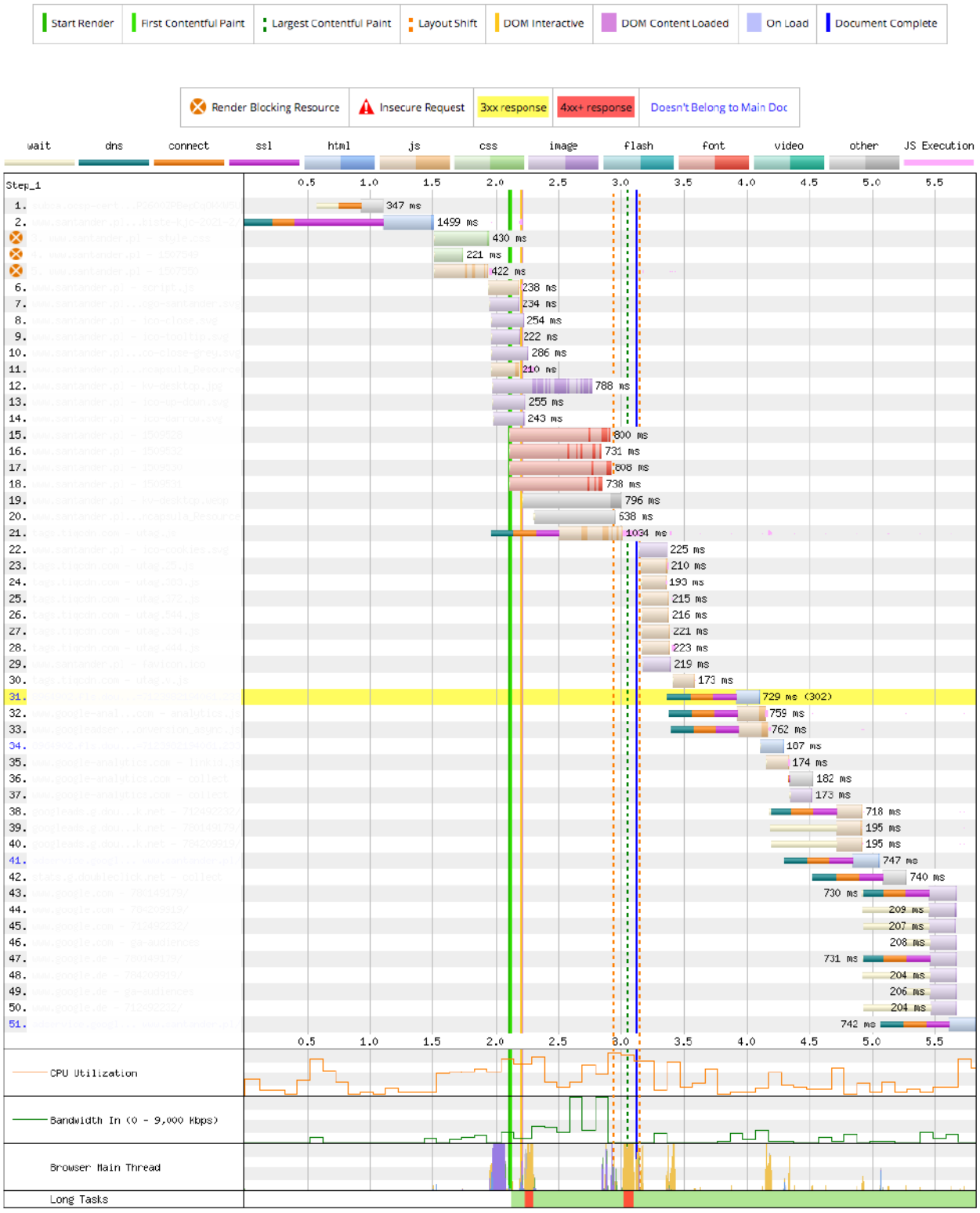
Pełny raport: Link do raportu WebPageTest…

Wideo przedstawiające ładowanie strony: Link do nagrania wideo WebPageTest…

Warto uświadomić sobie jak wygląda proces ładowania i tym samym renderowania strony dla użytkownika końcowego na testowanym urządzeniu i w założonym środowisku (lokacja, szybkość internetu), w oparciu o powyższe nagranie wideo.

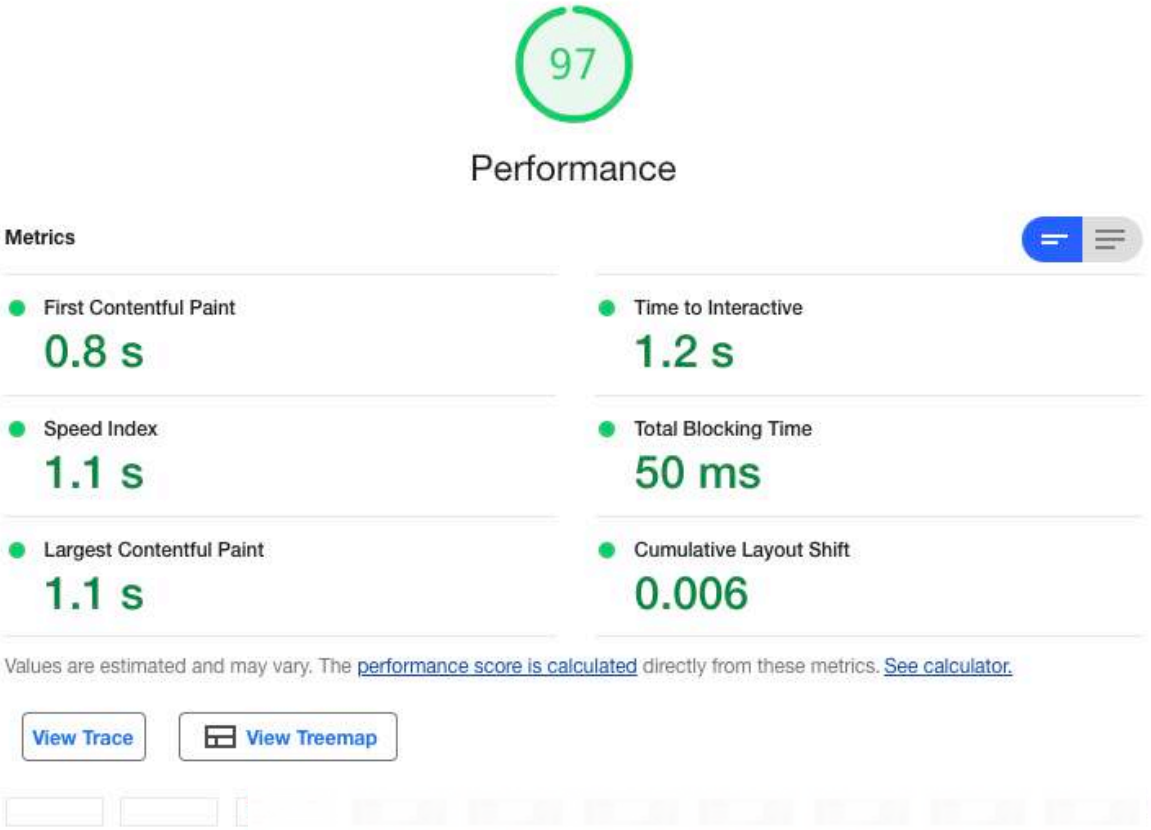
2022 Studio Sidekicks Sp. z o.o.

Waterfall zasobów



2022 Studio Sidekicks Sp. z o.o.

Poniższe wyniki testów pochodzą z narzędzia Lighthouse, uruchamianego w przeglądarce Chrome, emulując urządzenie desktop i stałe połączenie internetowe (kablowe). Warto zauważyć, że walidacja certyfikatu SSL z serwerem OSCP nie występuje w tym teście z powodu jego zapamiętania w pamięci podręcznej cache, stąd też wyniki testu są znacznie lepsze względem testów na WebPageTest lub Pagespeed Insights.



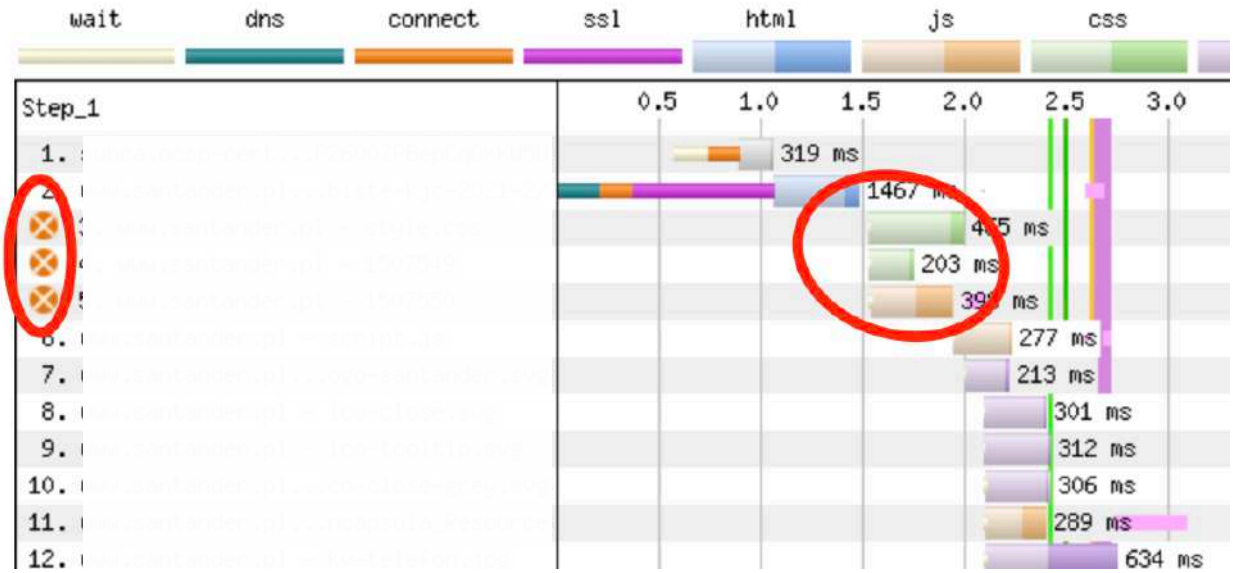
2022 Studio Sidekicks Sp. z o.o.

7.3 Wnioski z testowania oryginalnej strony na urządzeniu mobilnym oraz desktopie

Najważniejszym centrum informacji na temat problemów wydajnościowych jest **waterfall zasobów** (widoczny w punkcie 7.1 oraz 7.2) i to na nim oparto większość poniższych wniosków. Z uwagi na umiarkowanie dobrą wydajność strony na desktopie, poniższe wnioski zostały stworzone zwłaszcza w oparciu o testy strony na urządzeniu mobilnym i mobilnym internecie. Rekomendacje techniczne zawarte w dalszej części dokumentu pomogą jednak usprawnić wydajność strony nie tylko na smartfonach i wolniejszym połączeniu, ale również na desktopach.

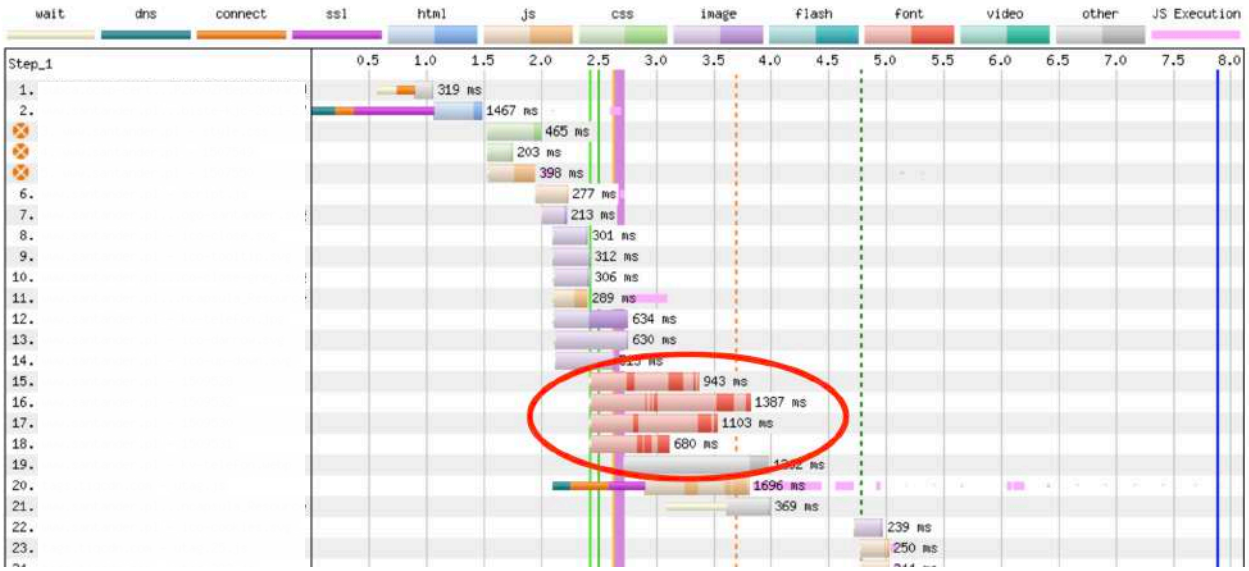
1. Zauważono długi czas oczekiwania na wyrenderowanie pierwszego elementu na stronie (średnio ok. 2.4 - 2.5s; patrz - metryka **Start Render**). Powodem zaistniałej sytuacji jest przede wszystkim długi czas negocjacji TLS / walidacji certyfikatu względem zewnętrznej usługi OSCP. Jest to domena certyfikatów typu EV (Extended Validation), gdzie ów proces zajmuje od jednej do kilku sekund, w zależności od dystansu sieciowego między użytkownikiem a najbliższym punktem OSCP. Więcej informacji na temat tego problemu można znaleźć w niniejszym artykule. Wymieniony w tym punkcie problem (będący jednocześnie popularną cechą serwisów o zwiększonej potrzebie zaspokojenia potrzeb bezpieczeństwa) odciska piętno na każdy kolejny element na stronie (na jego czas ładowania i renderowania). Na starcie traci się średnio 1-1.5s względem załadowania chociażby elementu **Largest Contentful Paint**.

2. Analiza waterfall'a wykazała ładowanie 3 blokujących renderowanie strony zasobów w postaci plików dwóch arkuszy CSS i jednego pliku JS. Proces ładowania i renderowania strony jest zablokowany do momentu ich pełnego załadowania. Jest to również kolejny powód długiego czasu oczekiwania na wyrenderowanie jakiekolwiek elementu na stronie.



2022 Studio Sidekicks Sp. z o.o.

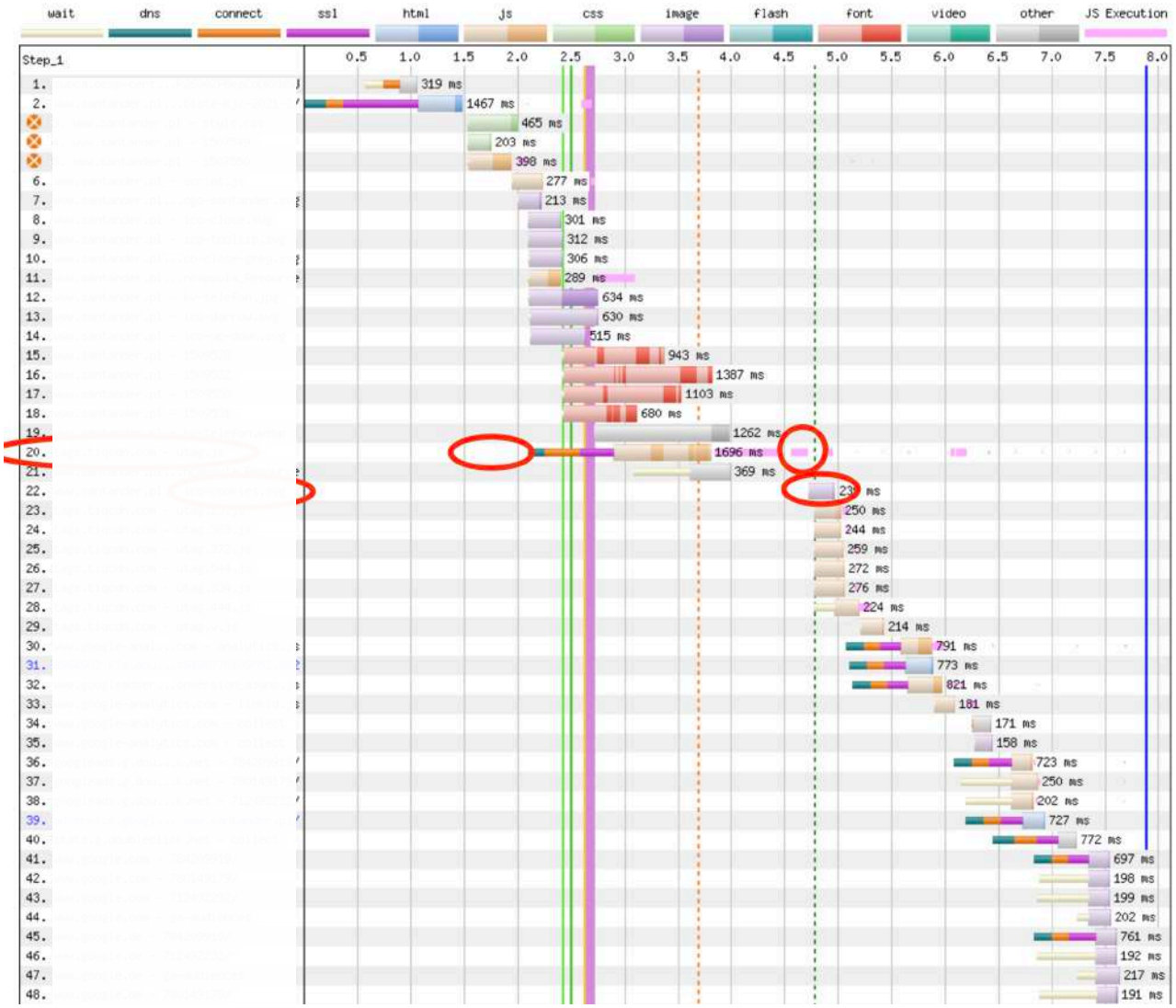
3. Fonty ładują się zbyt późno względem innych zasobów, powodując zwłaszcza na słabszym połączeniu internetowym, niekorzystny efekt wyświetlania tekstu w sposób nieostylowany, rażąco różniący się od docelowo ładowanego kroju czcionki. Na pochwałę zasługuję jednak fakt, że tekst pojawia się tak szybko, jak to możliwe (zastosowanie *font-display: swap*).



Powodem zaistniałej sytuacji (oprócz punktu 2. niniejszej listy problemów wydajnościowych) jest również fakt, że definicja @font-face fontów znajduje się w zewnętrznym pliku CSS. Przeglądarka najpierw musi pobrać HTML, następnie wszystkie pliki CSS, a później dopiero, z uwagi na inne zasoby, załadować pliki fontów, czyniąc ich ładowanie na osi czasu zbyt późnym.

2022 Studio Sidekicks Sp. z o.o.

4. Skrypt zewnętrzny utag.js ładuje się zbyt późno, ładując późno tym samym szereg kolejnych skryptów analitycznych (Google Analytics, Google Ads, tagi JS).

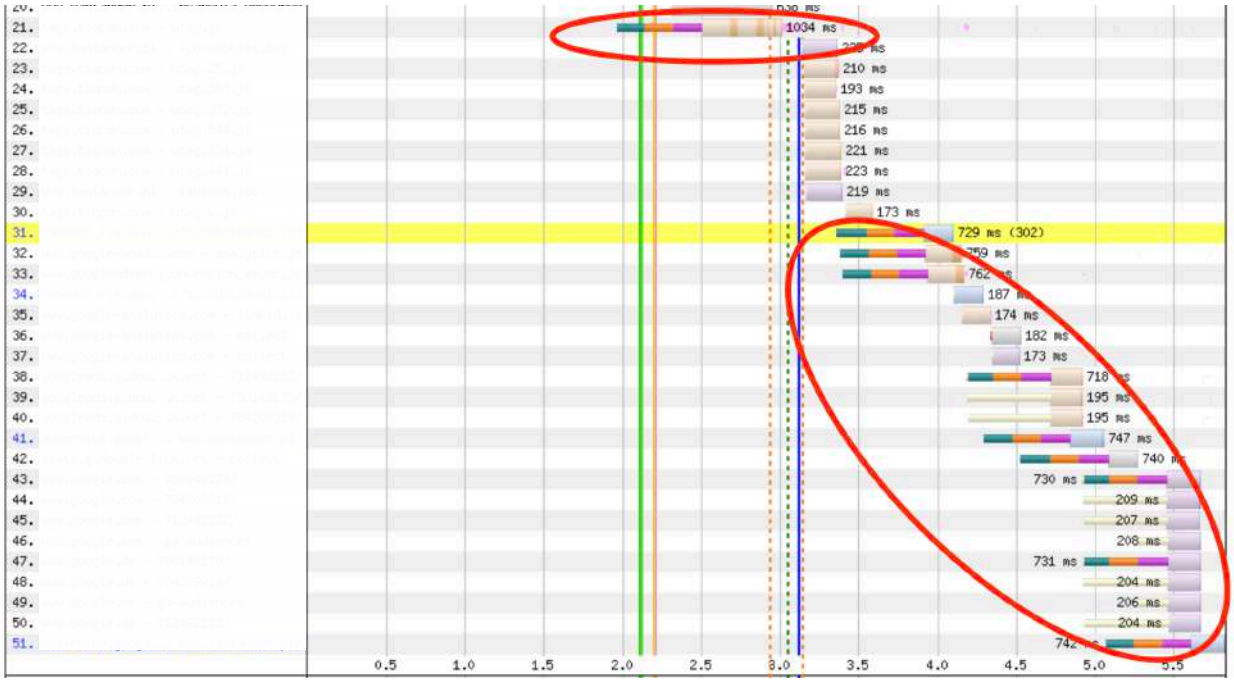


Na podstawie waterfall'a można wywnioskować, iż okienko z informacją o plikach cookies jest również ładowane przez tag manager, dając użytkownikowi wyjątkowo niekorzystany rezultat w postaci załadowania (i uniemożliwienia interakcji na stronie) okienka przykrywającego 100% strony, które dodatkowo jest uzależnione od szybkości zwracania danych z zewnętrznego serwera. Wspomnienie okienko o plikach cookies staje się również nowym (po zdjęciu w hero bannerze) elementem **Largest Contentful Paint** w procesie renderowania strony, którego czas ładowania uzależniony jest od zewnętrznego dostawcy (tag manager). Efektem tego stanu rzeczy jest długi czas potrzebny do wyświetlenia LCP, sięgający średnio 4.7s podczas pierwszej wizyty na stronie.

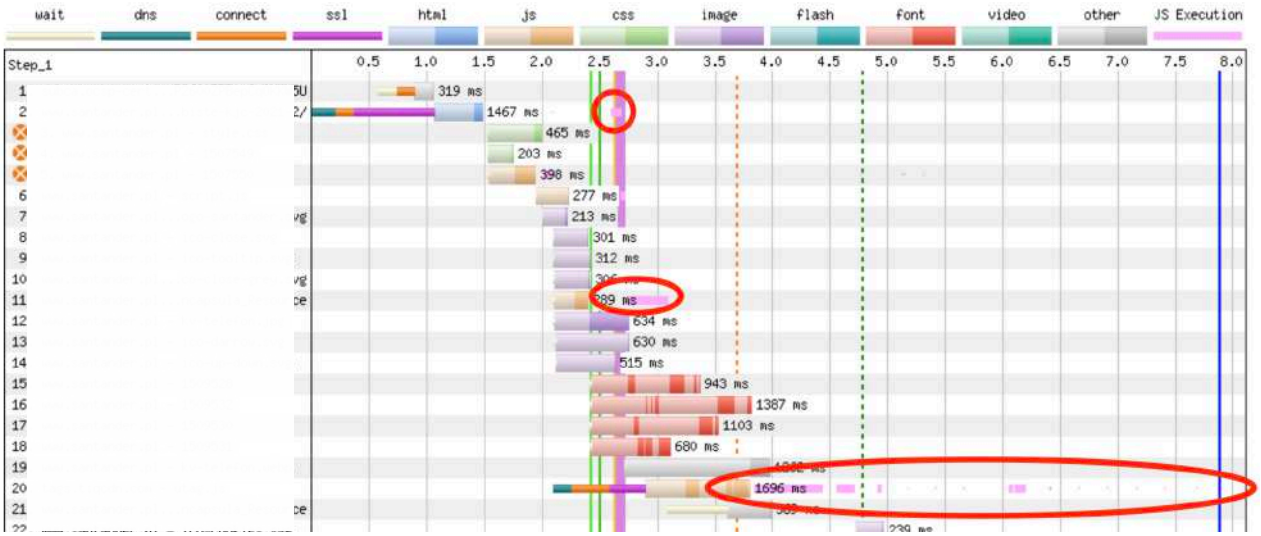
Kolejnym wnioskiem płynącym z tej sytuacji jest fakt, że tag manager ładuje kolejne pliki JS, które modyfikują strukturę DOM (okienko o plikach cookies), jak i te, które nie modyfikują (np. Google Analytics). Kombinacja obu rodzajów skryptów uruchamianych z poziomu pliku początkowego …….js uniemożliwia rozsądne przeorganizowanie kolejności ładowanych zasobów, by nie zaburzać procesu tworzenia drzewa renderowania.

2022 Studio Sidekicks Sp. z o.o.

5. Skrypt tag managera ładuje całą sekwencję kolejnych skryptów JS (serwowanych z kolejnych zewnętrznych serwerów, wydłużając ich ładowanie przez DNS lookup + TCP/TLS handshake), znacząco obciążając przeglądarkę użytkownika w postaci długich zadań „zleconych” przeglądarce (tzw. Long Tasks). Owe skrypty, wraz z problemami natury frontendowej samego kodu testowanej strony, powodują zbyt dużą sumę nadwyżek Long Tasks (>50ms) w przeglądarce, czyniąc wartość Total Blocking Time ok. 1.2s.

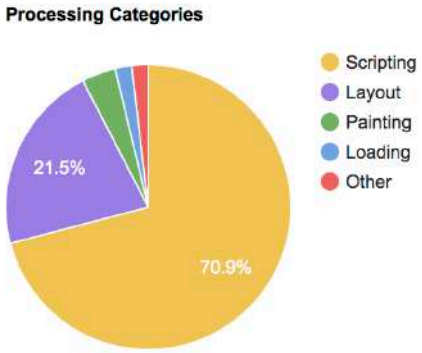


Warto zwrócić uwagę na czas i częstotliwość wykonywania kodu JS (co również stanowi problem długiego czasu **Total Blocking Time**).

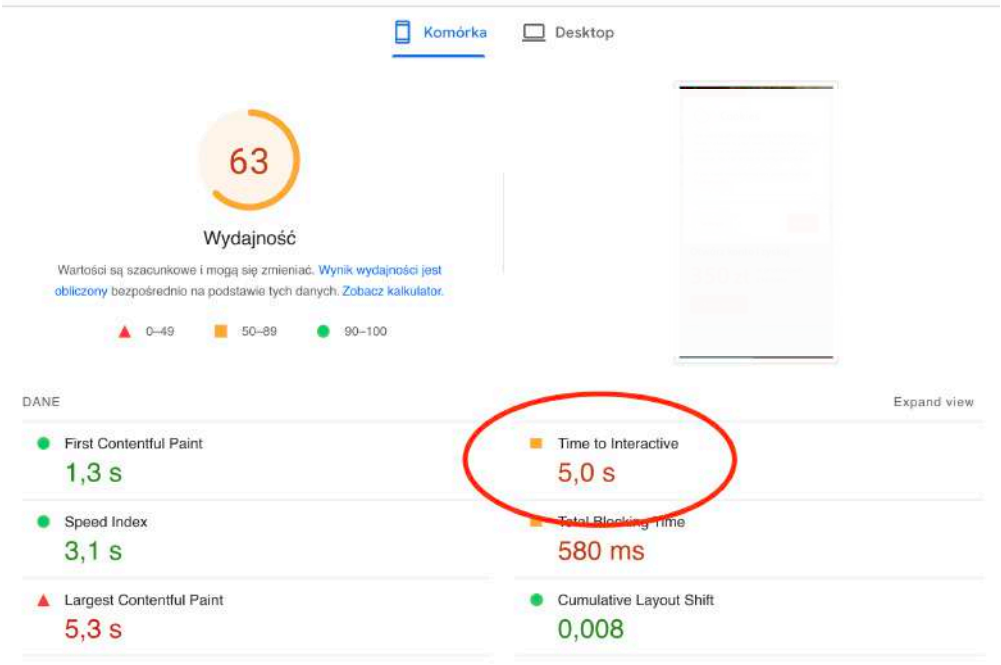


2022 Studio Sidekicks Sp. z o.o.

6. Nie można nie odnotować faktu, że większość problemów wydajnościowych na stronie pochodzi od zbyt dużej ilości kodu JS, którym główny wątek przeglądarki jest znacząco obciążony przez większość czasu. Powoduje to również negatywne reperkusje względem czasu do interaktywności na stronie (metryka **Time To Interactive**).



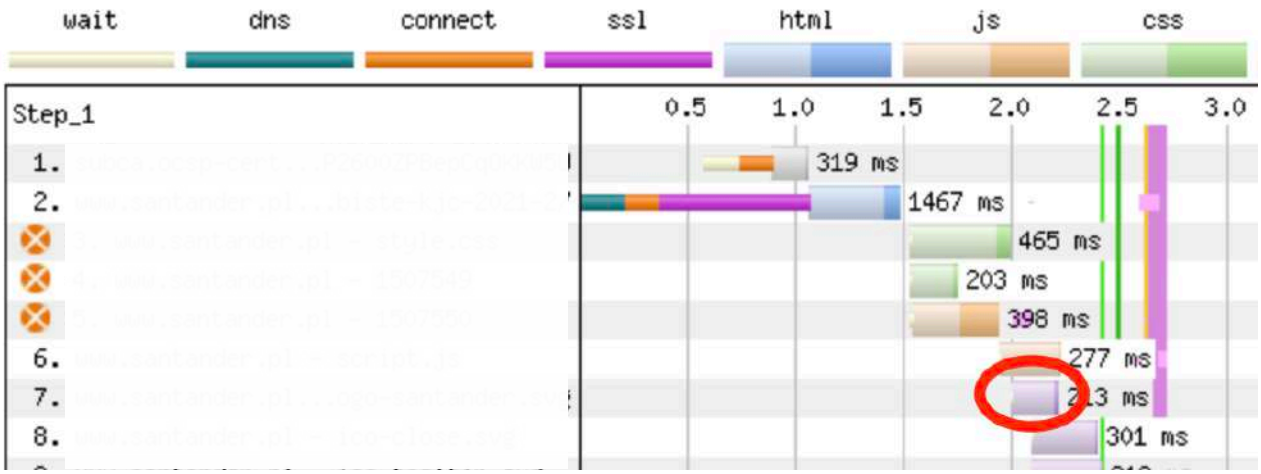
Category	Time (ms)
Scripting	1,820
Layout	552
Painting	99
Loading	48
Other	48



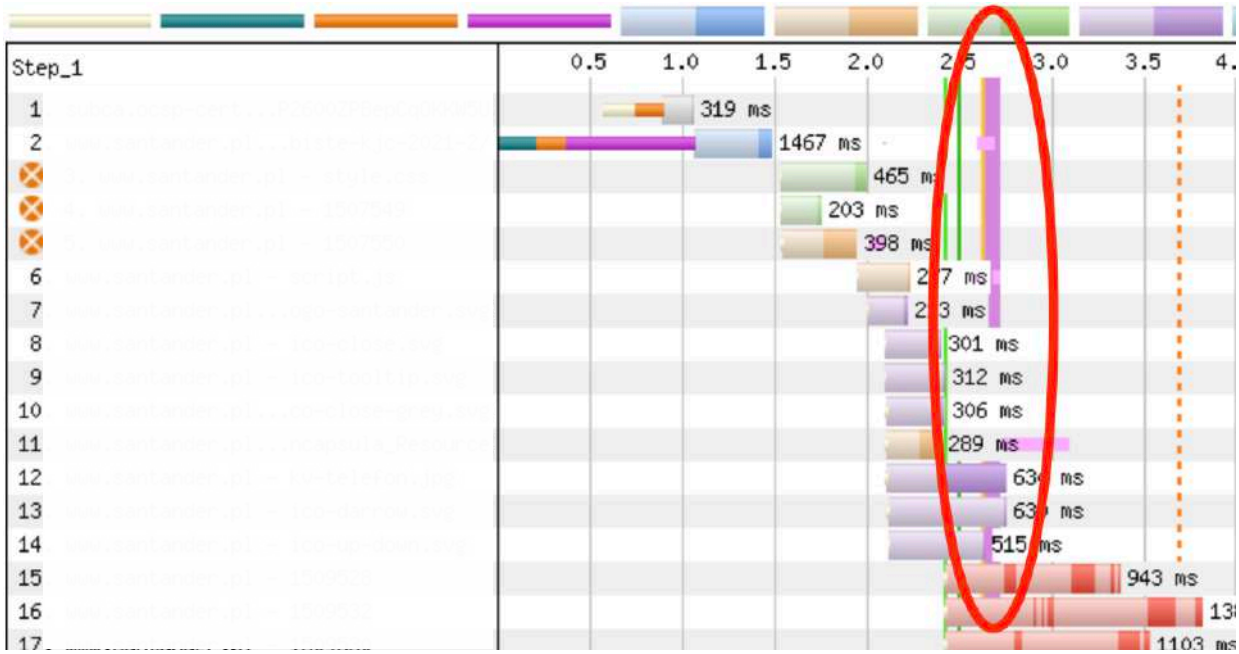
Powyższe wyniki pochodzą z testu przeprowadzonego za pomocą Pagespeed Insights.

2022 Studio Sidekicks Sp. z o.o.

7. Logo strony (będące ważnym elementem w przypadku dużej marki) ładuje się zbyt późno (jest to rezultat blokujących styli CSS i pliku JS, ładowanych przed logo).

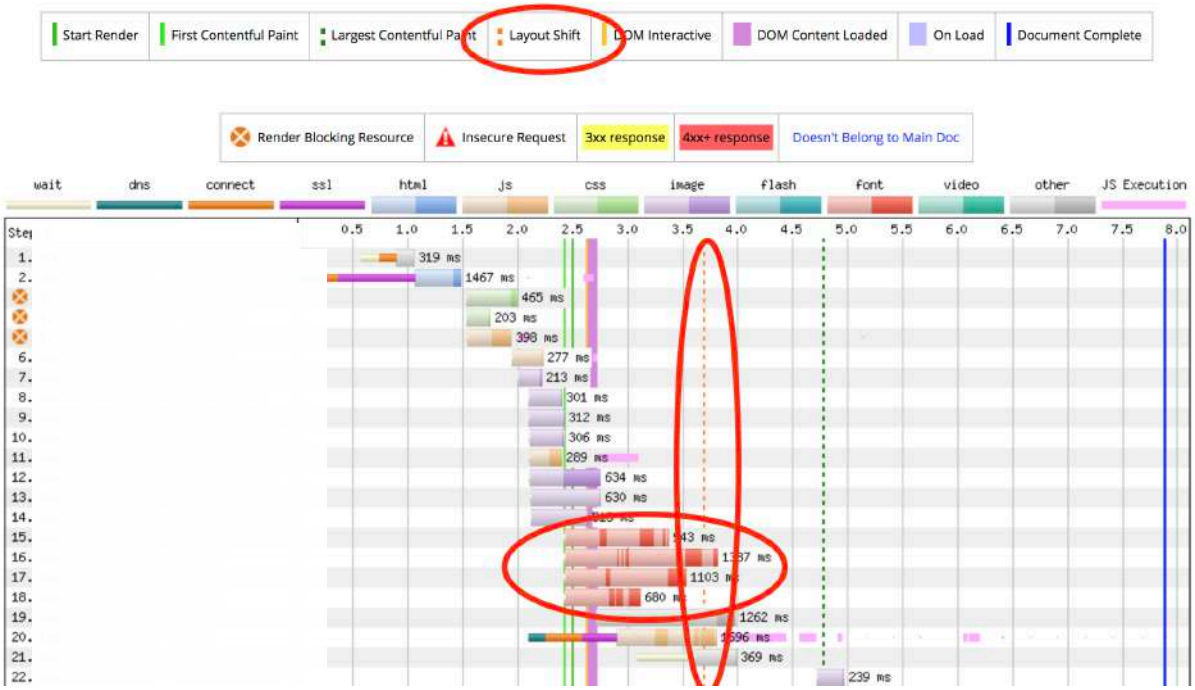


8. Zbyt duże obciążenie eventu DOM Content Loaded (na waterfall'u widoczny jest szeroki fioletowy pasek).

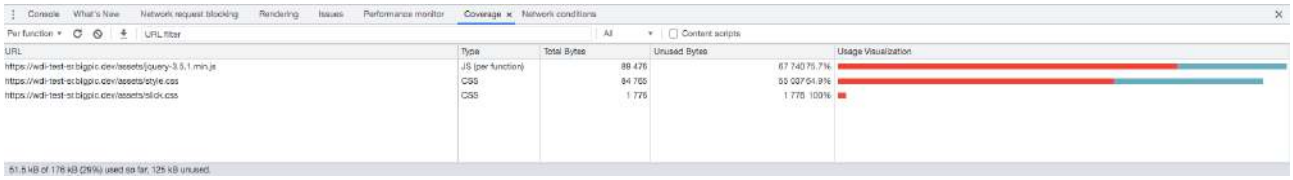


2022 Studio Sidekicks Sp. z o.o.

9. Na stronie zauważono nieznaczną niestabilność tekstu na stronie, z powodu późno ładowanych fontów, czego efektem jest lekkie przesuwanie się treści w momencie doładowywania kolejnych fontów. Nie powoduje to jednak powodów do niepokoju względem **Cumulative Layout Shift**.



10. Spora ilość kodu, która jest ładowana, a nie jest realnie wykorzystywana na stronie. Dotyczy to styli CSS, ale przede wszystkim JS. Zaobserwowano np. arkusz CSS, którego kod w 100% nie jest powiązany z kodem HTML, wobec czego jest ładowany bezużytecznie.



11. Kod HTML, CSS i JS nie jest zminifikowany. Wydłuża to czas potrzebny do załadowania tych zasobów, tak bardzo potrzebny w przypadku HTML i CSS, by rozpocząć renderowanie elementów w przestrzeni Above the Fold w krótkim czasie.

12. Pliki JS nie są złączone, a także logika kodu JS znajduje się zarówno w dokumencie HTML jak i zewnętrznych plikach.

13. Zastosowanie przestarzałych i sporych rozmiarowo bibliotek i rozwiązań (np. jQuery). Ładowana jest spora ilość kodu (biblioteka), którego w zdecydowanej większości nie wykorzystuje się na stronie.

14. Niektóre obrazki nie posiadają atrybutu **width** i **height**.

15. Nieprawidłowa kolejność ładowanych zasobów JS w sekcji <head> jak i <body>.

2022 Studio Sidekicks Sp. z o.o.

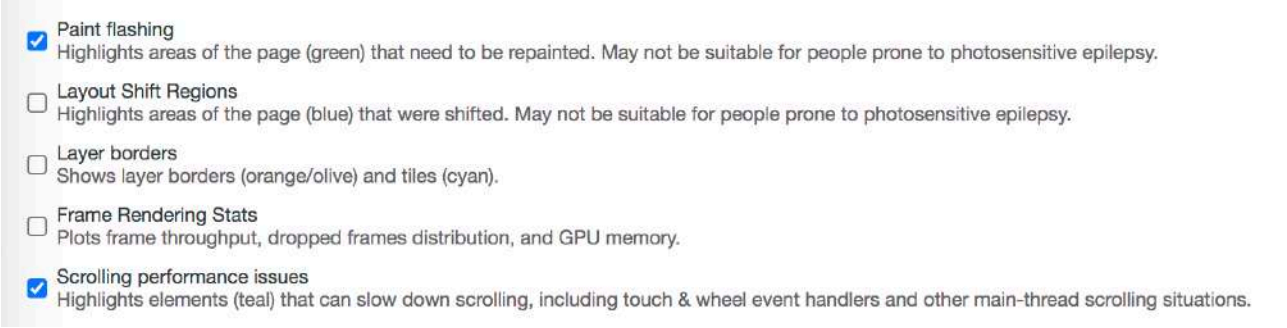
16. Kod JS powoduje zmiany w strukturze DOM, iterując po wielu elementach struktury (np. p, li, h2, h1). Wydłuża to czas wykonywania Javascriptu i wprowadza niepotrzebne zmiany w strukturze DOM, przyczyniając się do większego obciążenia głównego wątku przeglądarki.

17. Wielokrotnie wywoływanie document.addEventListener.

2022 Studio Sidekicks Sp. z o.o.

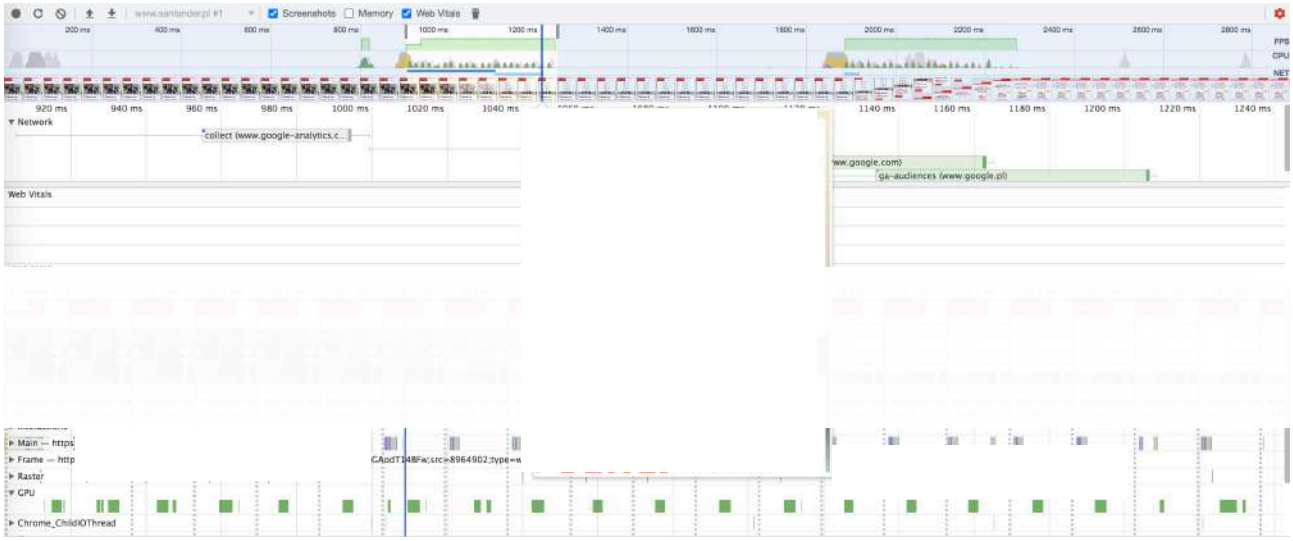
7.4 Testowanie renderowania strony i wnioski

Pierwszy test wydajności renderowania oparto o narzędzie **DevTools - Rendering**, korzystając z opcji Paint Flashing oraz Scrolling performance issues, **po załadowaniu strony**.



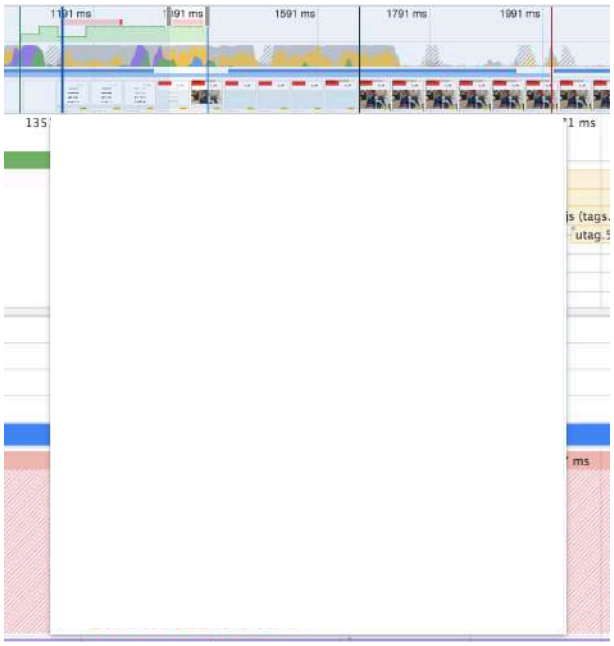
Badanie podczas interakcji takich jak scrollowanie, klikanie odnośników lub wyzwalanie automatycznej animacji w sekcji „Tu masz zawsze za 0 zł” **nie wykazały nieprawidłowości**. Strona jest wolna od efektu tzw. jank (gdy podczas scrollowania strona się trzęsie i daje nieprzyjemny efekt zacinania się przeglądarki podczas korzystania z niej).

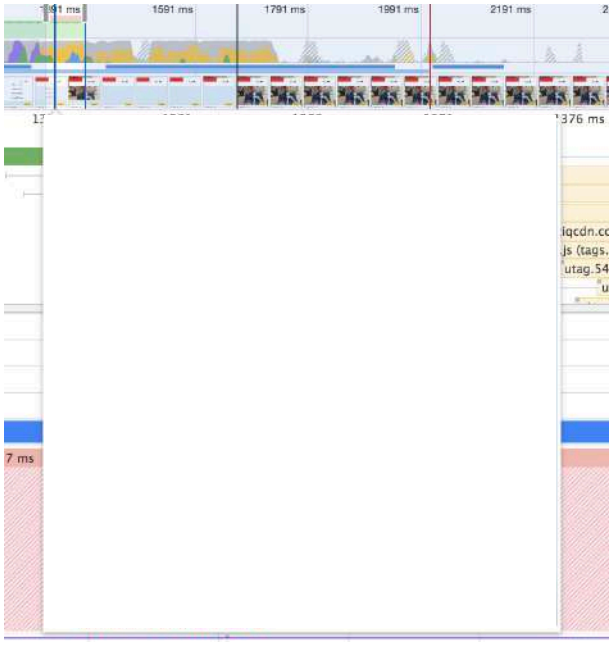
Test renderowania strony przy pomocy zakładki **Performance** w narzędziu DevTools podczas wyzwolenia otwarcia/zamknięcia nawigacji na urządzeniach mobilnych wykazał uruchamianie kosztownej fazy **Layout**, a następnie **Paint** i **Composite** (przez właściwość CSS - **top**), jednak z uwagi na nawigację osadzoną na warstwie o niewielkich rozmiarach i małą liczbę wewnętrznych węzłów DOM, nie powodowało to problemów z „wypadaniem” klatek podczas renderowania animacji.

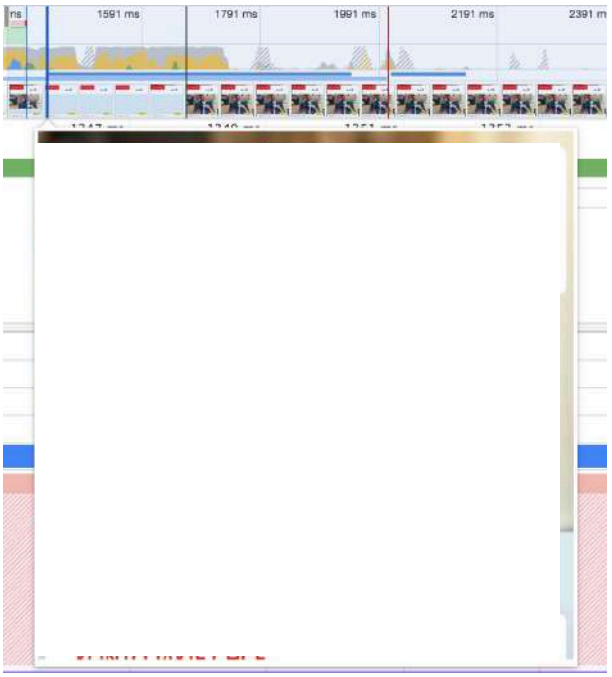


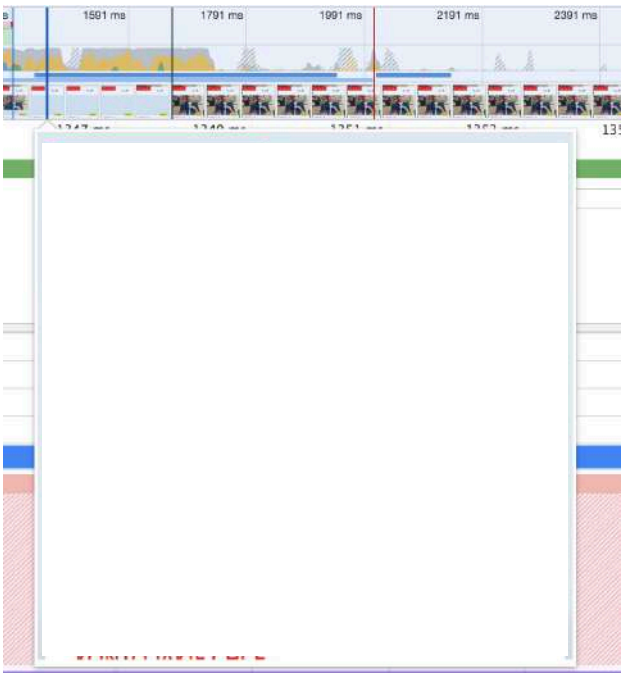
2022 Studio Sidekicks Sp. z o.o.

Podczas testowania renderowania **w trakcie ładowania strony** (od chwili wyzwolenia ładowania strony poprzez wpisanie adresu w przeglądarce aż do momentu jej pełnego załadowania), wykryto nieprawidłowości związane z nawigacją na urządzeniach mobilnych. Nawigacja jest otwarta od momentu wyrenderowania pierwszych elementów w przestrzeni Above the Fold, a następnie automatycznie się zamyka po ok. 0.5s. (w zależności od szybkości ładowania zasobów).







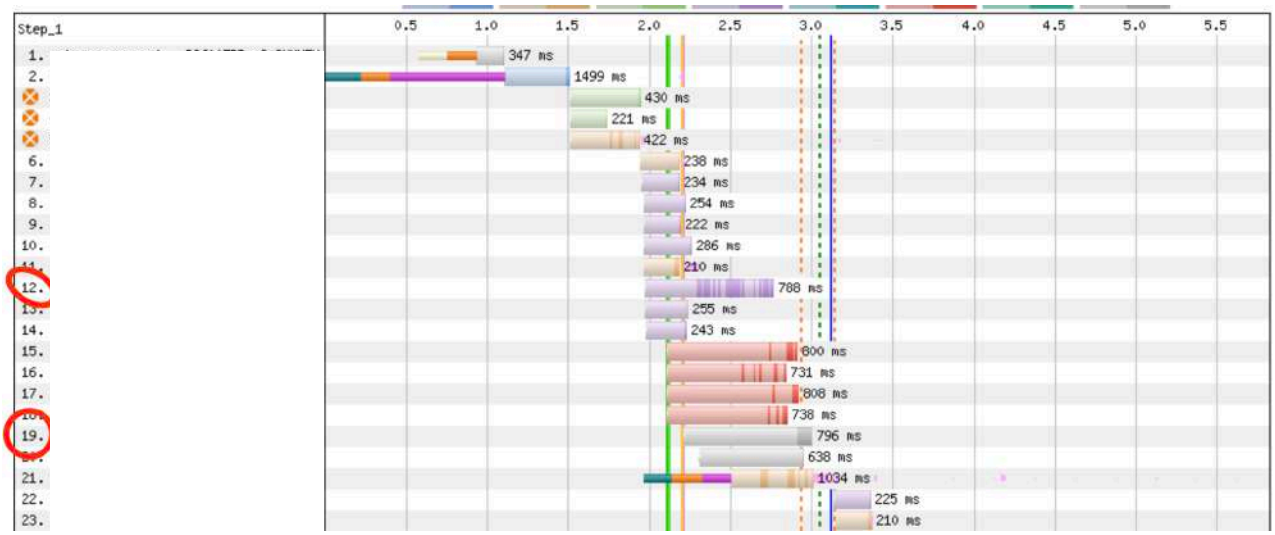


Zaistniałą sytuację warto prześledzić na nagraniu pod adresem **link do nagrania WebPageTest**. W 2.5s widoczna jest warstwa z otwartą nawigacją, która następnie w 2.6s zamyka się. Im gorsza jakość połączenia internetowego i gorsze parametry smartfona, tym problem jest bardziej i dłużej zauważalny.

2022 Studio Sidekicks Sp. z o.o.

Warto zwrócić uwagę na obrazek w przestrzeni Above the Fold, który - podobnie jak nawigacja - najpierw jest widoczny (ładowany zasób: **link do zasobu .jpg**), a następnie znika, by został podmieniony na inny (**link do zasobu .webp**).

W rezultacie, użytkownik na urządzeniu mobilnym ładuje ten sam zasób, w dwóch formatach jednocześnie (jpg oraz webp), dodatkowo zmuszając przeglądarkę do dwukrotnego renderowania tego samego elementu. Poniżej waterfall z zaznaczonymi zasobami, opisanymi powyżej.



Wspomniany obrazek stanowi strategicznie bardzo ważny element w przestrzeni Above the Fold i zgodnie z zasadą wymienioną w punkcie 2. niniejszego dokumentu, powinno się dążyć jego załadowania i wyrenderowania w jak najkrótszym czasie, nawet jeśli nie został on zakwalifikowany jako Largest Contentful Paint. W testowanej stronie zasób ten jest ładowany zbyt późno, a ponadto w dwóch formatach jednocześnie, by finalnie wyrenderować tylko ten drugi. Daje to również negatywne wrażenie podczas renderowania, gdy obrazek ładuje się, by później zniknąć i załadować ponownie po kolejnych sekundach lub ułamkach sekund (w zależności od jakości połączenia internetowego).

2022 Studio Sidekicks Sp. z o.o.

8. Rekomendacje techniczne

Rekomendacje techniczne opracowano na bazie wniosków z punktów 7.1 - 7.4.

8.1 Usunięcie nieużywanego kodu

Arkusz **link do arkusza styli** (plik .css) jest niepotrzebnie ładowany - analiza kodu nie wykazała powiązania klas CSS zawartych w tym pliku z żadnym elementem struktury HTML. Dzięki jego usunięciu, zmniejszy się liczba żądań oraz liczba zasobów blokujących renderowanie strony.

Przed:

Po:

8.2 Wstrzyknięcie kodu CSS bezpośrednio do dokumentu HTML

Strona boryka się z problemem **późnego renderowania**. Powodem tej sytuacji jest wydłużony czas zwracania odpowiedzi z backendu (walidacja SSL / łączenie z serwerem OSCP), ale także blokujące zasoby w sekcji <head> (arkusze styli CSS oraz skrypty JS [źródło]). Dobrym pomysłem

2022 Studio Sidekicks Sp. z o.o.

jest wstrzyknięcie kodu CSS bezpośrednio do kodu HTML, aby po jego pobraniu, przeglądarka nie musiała wykonywać kolejnego żądania do zewnętrznego pliku CSS, a po prostu móc tworzyć drzewo renderowania od razu po otrzymaniu kodu HTML wraz z kodem CSS w postaci pierwszego - i tym samym jednego - żądania. Opisana technika często nosi miano wstrzykiwania „krytycznego kodu CSS", gdzie w sekcji <head> dokumentu HTML wstrzykuje się style CSS odpowiadające za elementy krytyczne, widoczne od razu, w przestrzeni Above the Fold, natomiast z racji iż testowana strona jest prostą stroną statyczną (niezawierającą zbyt dużej ilości CSS), proces ten można uprościć i wstrzyknąć cały arkusz.

Kolejnym aspektem, dla którego warto podjąć ten krok, są fonty. Definicja @font-face znajduje się obecnie w zewnętrznym arkuszu, przez co wydłuża się czas oczekiwania na załadowanie plików fontów. Przeglądarka musi najpierw pobrać kod HTML z serwera, następnie wykonać żądanie do zewnętrznego pliku CSS i dopiero po jego pobraniu rozpozna i rozpocznie pobieranie pliki fontów. Wstrzykując kod CSS do HTML omijamy ten krok - przeglądarka pobierze HTML wraz z kodem CSS i od razu rozpozna, że musi załadować pliki fontów.

Warto zwrócić uwagę, iż kod CSS wstrzykiwany do sekcji <head> powinien być **zminifikowany**. Rekomendowane jest upewnienie się, że ścieżki do plików graficznych zawartych w arkuszu są poprawne po procesie wstrzyknięcia styli do HTML.

Opisana technika wstrzykiwania styli CSS do kodu HTML powinna być zautomatyzowana przez developerów (przykład).

Przed:

Po:

 2022 Studio Sidekicks Sp. z o.o.

8.3 Lokalny kod JS

W sekcji <head> dokumentu HTML ładowany jest skrypt JS (biblioteka jQuery), który blokuje renderowanie strony. Rekomenduje się **złączenie (konkatenację) całego lokalnego kodu JS** strony (link do pliku .js + inny plik .js + kod osadzony w dokumencie HTML) **do jednego pliku i osadzenie go pod koniec znacznika zamykającego </body>**. Pomoże to przyspieszyć proces renderowania strony, a także zmniejszyć liczbę żądań podczas ładowania strony.

Oprócz konkatenacji wspomnianego wyżej kodu JS, rekomenduje się jego zminifikowanie. Pozwoli to zmniejszyć rozmiar przesyłanych danych z serwera do przeglądarki.

Przykład osadzenia złączonych skryptów w postaci jednego, zminifikowanego pliku, pod koniec dokumentu HTML:

```
        </div>
    </footer>

    <script type="text/javascript" src="./assets/script.js"></script>
</body>

</html>
```


2022 Studio Sidekicks Sp. z o.o.

8.4 Ładowanie zewnętrznego skryptu tag managera - zmiana kolejności

Skrypt tag managera, ładowany obecnie asynchronicznie (słusznie) powinien być przesunięty na sam początek sekcji <head>. Pozwoli to na jego szybsze ładowanie względem innych zasobów i wcześniejsze załadowanie kolejnych plików JS (w tym zewnętrznych).

Przed:

Po:

2022 Studio Sidekicks Sp. z o.o.

8.5 Zmiana techniki ładowania zdjęcia w górnym bannerze

Zaobserwowano, iż późne ładowanie zdjęcia na górze strony spowodowane jest:

a) osadzaniem zdjęcia przez CSS

b) manipulacją formatem ładowanego zdjęcia przez JS

c) brakiem priorytetyzacji ładowania

Ładowanie plików graficznych przy pomocy CSS (background-image) jest statystycznie zawsze wolniejsze od ładowania przez znacznik <img> (źródło). Zaleca się ładowanie zdjęć zawsze poprzez HTMLowy tag <img>, który oprócz usprawnienia czasu ładowania, pozwoli również na dodanie tekstu alternatywnego. Dzięki temu obrazek będzie indeksowany przez wyszukiwarki (np. Google Grafika), a czytniki ekranowe będą zdolne rozpoznać zdjęcie i odczytać ów tekst alternatywny osobie niepełnosprawnej.

Osadzenie obrazka w dokumencie HTML umożliwi również łatwiejsze ładowanie różnych rozmiarów obrazka ze względu na rozdzielczość ekranu użytkownika, a także ładowanie odpowiedniego formatu zdjęcia, obsługiwanego przez przeglądarki.

Zaobserwowano, iż lokalny kod JS na testowanej stronie, manipuluje formatem zdjęcia w zależności od rozpoznanej przeglądarki. Zauważone problemy długiego ładowania zdjęcia, a także wspomniany w punkcie 7.4 problem jego ładowania, późniejszego zniknięcia i wyrenderowania na nowo, są powodem manipulacji tłem bannera strony poprzez niniejszy JS:

```
if(/^((?!chrome|an

   if (window.matc
      document.que                                                    .webp)';
   } else if(windo
      document.que                                                    webp)';
   } else {
      document.que                                                    .webp)';
   }
}
```

Źródło: link do pliku .js

Rekomenduje się osadzenie obrazka w górnym bannerze w następujący sposób:

```
<section class="banner overflow-hidden" id="banner" role="img" aria-label="                    ">
    <div class="banner__img">
        <div>
            <picture>
                <source src=                                    .webp" media="(max-width: 767px)">
                <source src=                                    .webp" media="(min-width: 768px) and (max-width: 1023px">

                <img
                    width=
                    heigh
                    src="                                       g"
                    alt="
            </picture>
        </div>
    </div>

    <div class="container h-100">
```

Ważne: należy uzupełnić powyższy przykład dodatkowymi znacznikami <source> z formatem jpg/ png w razie potrzeby wsparcia starszych przeglądarek (przykład).

2022 Studio Sidekicks Sp. z o.o.

Powyższe rozwiązanie **wymaga napisania dodatkowego kodu CSS**, dzięki któremu możliwe będzie wypozycjonowanie zdjęcia w taki sam sposób, jaki przewiduje obecny design testowanej strony.

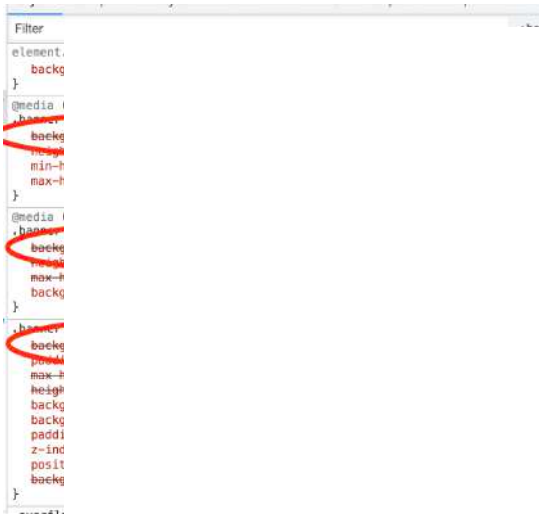
2022 Studio Sidekicks Sp. z o.o.

```css
.banner {position: relative;}
.banner__img {
    position: absolute;
    top: 0;
    left: 0;
    right: 0;
    bottom: 0;
}

.banner__img div {
     display: flex;
     justify-content: center;
     position: relative;
     width: 100%;
     height: 100%;
}

.banner__img {
      width: 100%;
      height: 100%;
}

.banner__img picture {
      position: absolute;
}

.banner__img img {
    display: block;
    height: 100%;
    width: auto;
}
```

Przykład kodu CSS

Ważne: *zaprezentowany kod należy użyć z rozwagą. Powinno się przetestować zachowanie bannera strony na różnych przeglądarkach i rozdzielczościach i w razie potrzeby dostosować style.*

Oprócz powyższego kodu CSS, należy usunąć z arkusza reguły CSS, które osadzają obrazek poprzez background-image. Przykład obok.



2022 Studio Sidekicks Sp. z o.o.

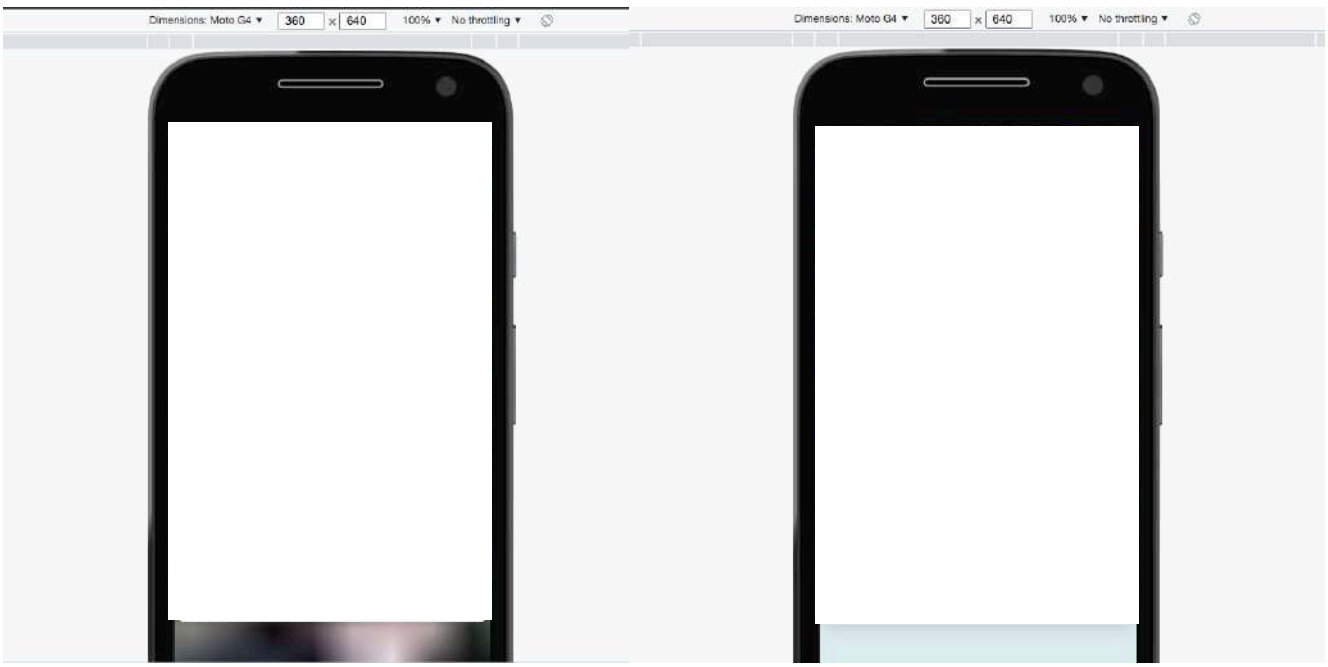
Ponadto, celem wyższej priorytetyzacji ładowania zasobu odpowiedzialnego za górny banner, należy skorzystać ze znacznika <link> preload, dzięki czemu przeglądarka będzie mogła szybciej załadować ów zasób, w zależności od danej rozdzielczości ekranu.

Ważne: należy dostosować adresy URL zdjęć oraz wartość atrybutów media, zgodnie z designem strony i breakpointami, dzięki którym na danej rozdzielności ładowane jest stosowne zdjęcie.

Kolejną rekomendacją jest zastosowanie **Low Quality Image Placeholder** (źródło). Nie jest to konieczność (by to zastosować), natomiast by zaprezentować użytkownikowi obrazek natychmiastowo (chociaż w słabej jakości), można pokusić się o wygenerowanie obrazka (np. o szerokości 50px) w formie Data URI (base64) i osadzić go jako background-image dla warstwy . Przykład kodu:

2022 Studio Sidekicks Sp. z o.o.

Dzięki zastosowanej technice, obrazek w trakcie ładowania wyświetli się niemalże od razu w słabej jakości (nie wymaga dodatkowego żądania, gdyż jego kod jest w dokumencie HTML), by być zamienionym odpowiednim obrazkiem chwilę później.



Proces pojawienia się właściwego obrazka można uczynić kreatywnym, np. z użyciem prostej animacji CSS.

8.6 Optymalizacja kodu JS

Po przeanalizowaniu kodu osadzonego bezpośrednio w kodzie HTML jak i w pliku **link do pliku .js**, należałoby przede wszystkim wyeliminować niepotrzebną iterację po elementach HTML i podmienianie niektórych wartości (chociażby tekstowych) „w locie”. Przykładem opisywanej sytuacji jest poniższy kod:

```
document.addEventListener("DOMContentLoaded", function () {
    $("p, li, h2, h1").each(function () {
        va
        te                                                  nbsp;");
        te


        );
        $(
    });
});
```

Modyfikacja elementów DOM wydłuża czas do interaktywności na stronie i znacząco obciąża główny wątek przeglądarki. Rekomenduje się usunięcie powyższego fragmentu kodu i zastosowanie zmian bezpośrednio w kodzie HTML (np. fraza *xxx* zamieniona na *yyy*).

2022 Studio Sidekicks Sp. z o.o.

Zaobserwowane nadmierne dodawanie nasłuchiwania zdarzeń. Przykład:

```
document.addEventListener(
    "DOI
    pas:
);
docume:
    "DOI
    pas:
);
docume:                                                    {
    $("1
```

Rekomenduje się wykonanie powyższego kodu JS bez dodawania nasłuchiwania na event DOMContentLoaded, z racji umiejscowienia całego (skonkatentowansego z biblioteką JS i zewnętrznym plikiem script.js) pliku pod koniec znacznika </body>. Przykład:

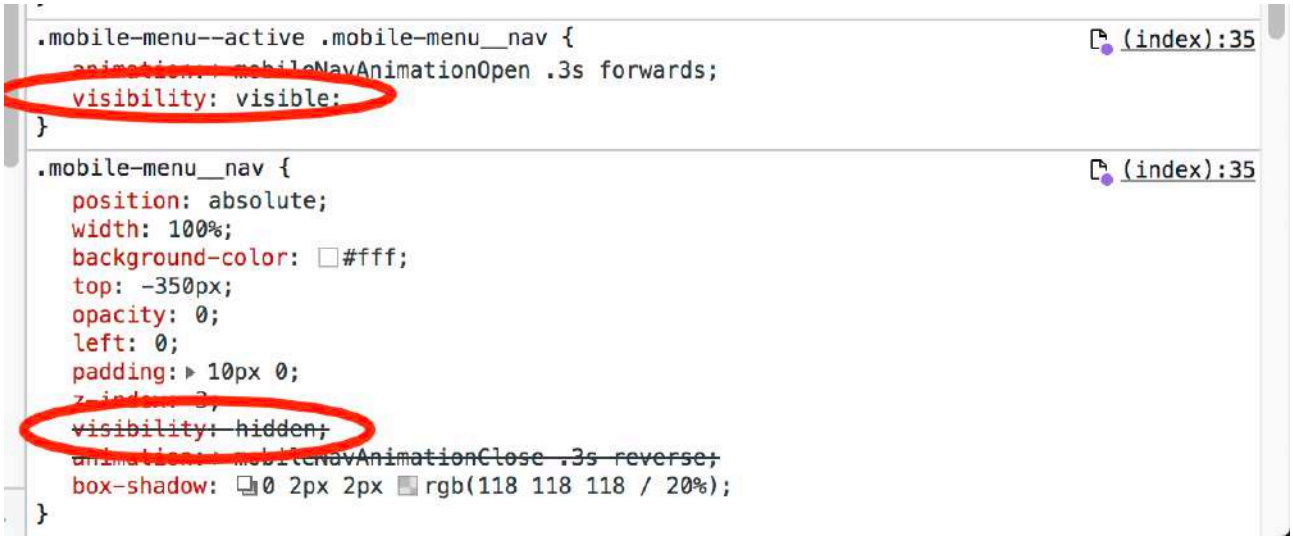
```
(function() {
                                                                                    ";
})();
```

Należy prześledzić kod JS i dokonać stosownych zmian, celem ograniczenia nasłuchiwania zdarzeń w dokumencie.

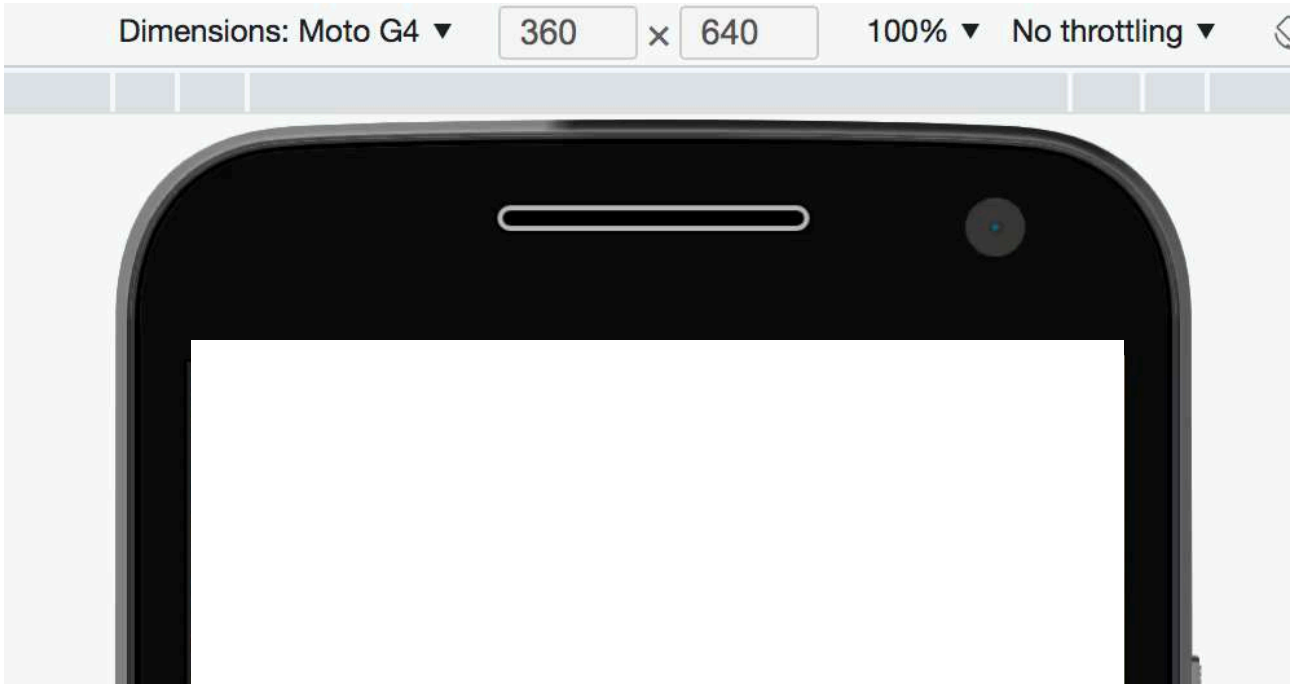
2022 Studio Sidekicks Sp. z o.o.

8.7 Problem z górną nawigacją na urządzeniach mobilnych

Uruchamianie animacji CSS o nazwie *mobileNavAnimationClose* w klasie *.mobile-menu__nav* jest powodem pojawienia się tego elementu podczas początkowej fazy renderowania strony, by po chwilę została zamknięta/zwinięta. Rekomenduje się zastosowanie innego rozwiązania otwierania/zamykania nawigacji, aby początkowy stan elementu nie wyzwalał animacji lub po prostu uczynił tę warstwę niewidoczną. Rozwiązaniem może być zastosowanie właściwości *visibility: hidden*, która tuż po kliknięciu przycisku Menu, zostałaby zmieniona na wartość *initial* lub *visible*. Należy pamiętać o dostosowaniu animacji do wskazanej poprawki.



Wedle części opinii użytkowników testowanej strony, można by pokusić się o dostosowanie wyglądu belki nawigacyjnej na urządzeniach mobilnych i postarać się wyśrodkować przycisk wyzwalający warstwę nawigacyjną Menu, a także dostosować style CSS, by otwarta nawigacja nie nachodziła na ów przycisk.



2022 Studio Sidekicks Sp. z o.o.

8.8 Dalsza eliminacja żądań

Na stronie występuję kilka plików/ikonek SVG, które mogłyby być osadzone bezpośrednio w

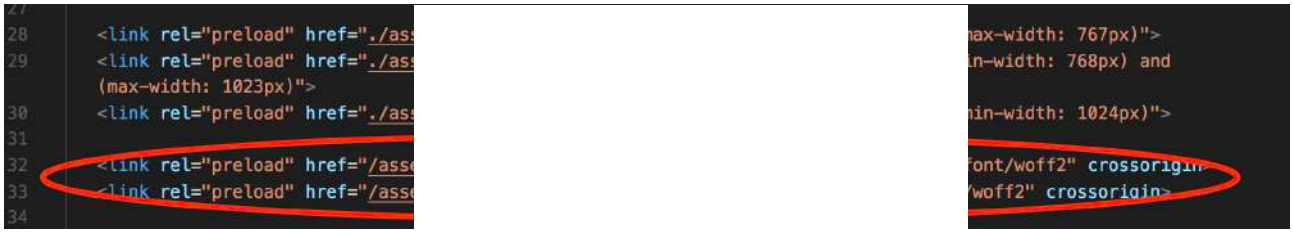
dokumencie HTML. Przykładem takiego elementu jest logo. Pozwoli to na jego szybsze renderowanie.

2022 Studio Sidekicks Sp. z o.o.

8.9 Priorytetyzacja ładowanych fontów

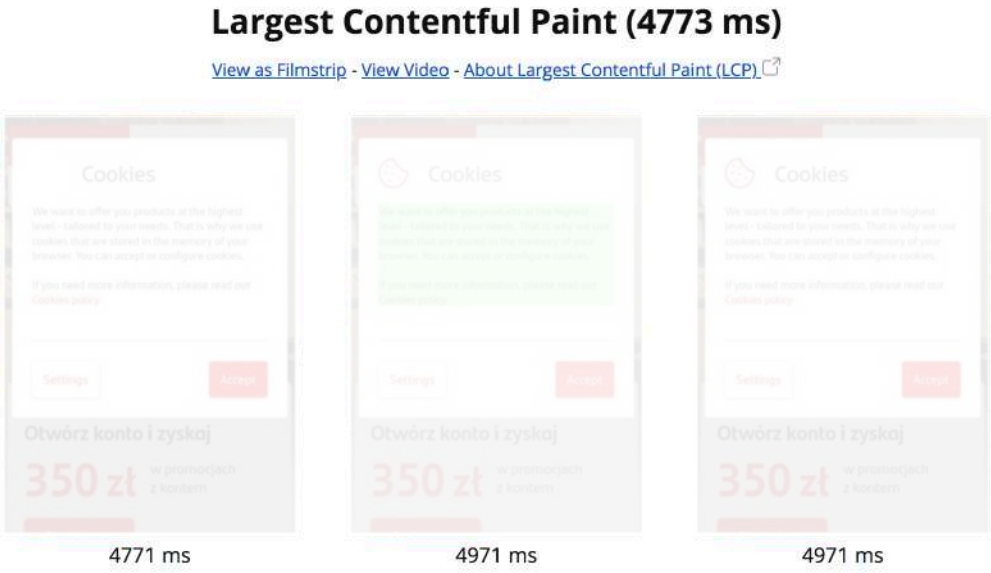
Dobrym pomysłem na skrócenie czasu, kiedy treści na stronie prezentowane są w formie nieostylowanej/korzystając z domyślnych, systemowych krojów czcionek, jest zastosowanie <link> preload, podobnie jak w przypadku zdjęć. Warto zwrócić uwagę, aby wskazać przeglądarce tylko najważniejsze fonty (idealnie tylko te, które są wykorzystywane do najważniejszych/największych elementów w przestrzeni Above the Fold).

Przykład zastosowania:



8.10 Largest Contentful Paint (LCP) zależny od skryptu zewnętrznego

Podczas każdego testu wydajności, z wykorzystaniem każdego narzędzia, elementem zakwalifikowanym jako Largest Contentful Paint jest okienko z informacją o plikach cookies. Dotyczy to wizyty na stronie, kiedy użytkownik nie potwierdził jeszcze polityki ciasteczek, czyniąc tę informacją ukrytą przy każdej następnej wizycie. W momencie potwierdzenia polityki, elementem LCP jest tekst w górnym bannerze.



Analiza strony i jej kodu dowiodła, że za wyrenederowanie tego elementu odpowiada kod HTML/JS załadowany przez zewnętrzny skrypt tag managera.

2022 Studio Sidekicks Sp. z o.o.

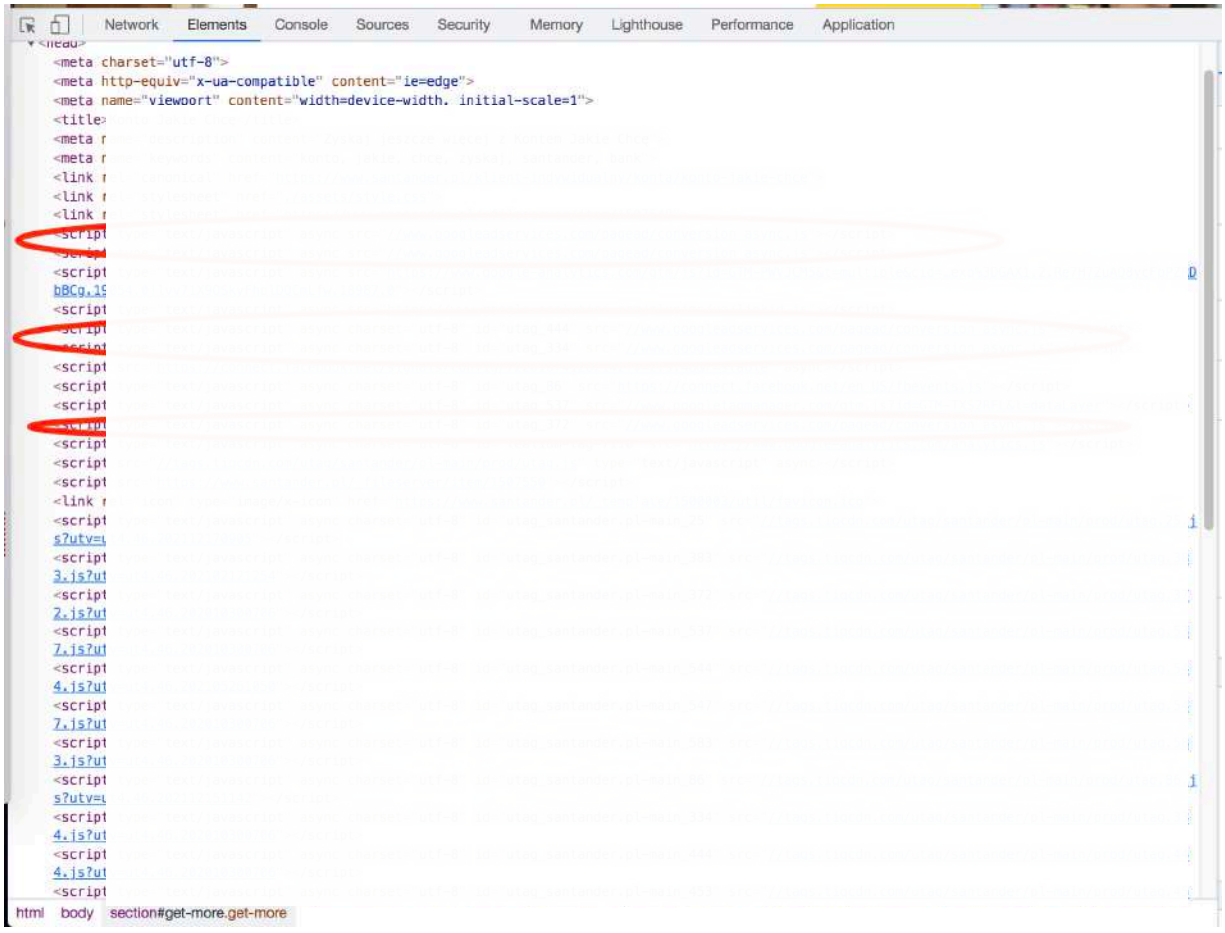
Im dłuższy będzie czas ładowania kodu z tego zewnętrznego źródła (by do którego się połączyć wymagana jest faza TCP/TLS handshake [źródło]), tym gorszy będzie czas wyrenderowania tego największego elementu w przestrzeni Above the Fold. Wpływa to niezwykle niekorzystnie na postrzeganie i użyteczność strony, im wolniejszy jest smartfon lub połączenie internetowe.

Rekomenduje się, by kod odpowiedzialny za wyświetlanie tego okienka, przenieść bezpośrednio do kodu HTML strony i obsłużyć jego otwieranie/zamykanie **lokalnie załadowanym plikiem JS** (idealnie asynchronicznie), poza skrypt tag managera. Dzięki temu pojawienie się tego elementu nie będzie zależne od szybkości załadowania pliku JS z zewnętrznego serwera, a będzie obsłużone z lokalnego poziomu strony, ładowanego z tego samego serwera.

Im szybciej będzie ładowany i wyrenderowany wspomniany w tym punkcie element, tym lepsze czasy będzie prezentowała metryka Largest Contentful Paint. Bez wdrożenia opisanej rekomendacji nie jest to możliwe.

8.11 Analiza skryptów ładowanych przez utag.js

Zaobserwowano, iż manager tagów ładuje niektóre zewnętrzne zasoby wielokrotnie. Przykład dokumentu HTML tuż po załadowaniu wszystkich plików JS poprzez Tealium.



Rekomenduje się analizę tagów i kodu JS. Być może będzie możliwe ograniczenie ładowania niektórych plików/kodu, przyspieszając całkowity czas ładowania strony i uzyskanie krótszych czasów do pełnej interaktywności.

Autorski kod JS, który ładowany jest za pośrednictwem utag.js, powinien być przeanalizowany pod kątem zastosowanej technologii. Wysoce rekomenduje się np. eliminację biblioteki jQuery i przepisanie kodu z użyciem „czystego” Javascriptu. Pozwoliłoby to na uzyskanie wydajniejszego kodu, a także znacząco mniejszego rozmiaru kodu JS.

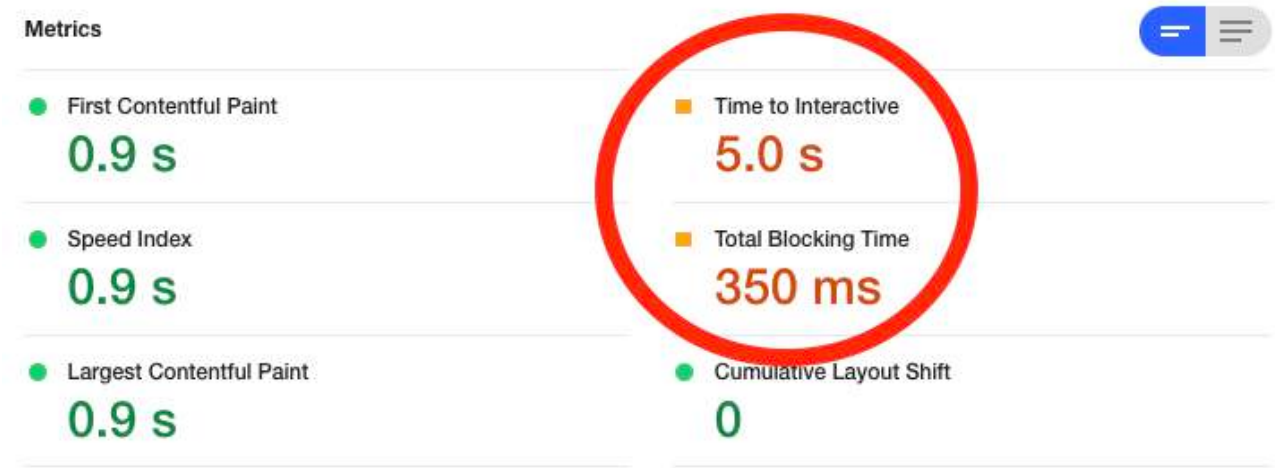
2022 Studio Sidekicks Sp. z o.o.

Celem uświadomienia sobie wpływu kodu ładowanego przez tag manager na stronę (jej interaktywność, responsywność; metryki, które odpowiadają za ich poziom to np. Total Blocking Time oraz TTI), dokonano szybkiego testu, polegającego na załadowaniu niemalże pustego dokumentu HTML, bez żadnych dodatkowych plików, a jedynie ze skryptem tag managera.

Przykład:

```
<html>
    <head>
        <script type="text/javascript">
            (function (a, b, c, d) {
                a = "//tags.tiqcdn.com/utag/santander/pl-main/prod/utag.js";
                b = document;
                c = "script";
                d = b.createElement(c);
                d.src = a;
                d.type = "text/java" + c;
                d.defer = true;
                a = b.getElementsByTagName(c)[0];
                a.parentNode.insertBefore(d, a);
            })();
        </script>
    </head>
    <body>
        Pusta strona, bez żadnych plików.
    </body>
</html>
```

Rezultat badania wydajności (narzędzie Lighthouse):



2022 Studio Sidekicks Sp. z o.o.

8.12 Wolny certyfikat SSL (Extended Validation)

Rekomendacją, która pozwoliłaby na przyspieszenie strony i uniknięcie dodatkowego żądania do serwera OSCP, jest zmiana typu certyfikatu (np. z metodą walidacji Domain Validation). Sugeruje się przeczytanie i analizę tego artykułu, bardzo szczegółowo przedstawiającego owe zagadnienie, a następnie przedyskutowanie niniejszej rekomendacji z zespołem IT/zespołem bezpieczeństwa.

Nowoczesne przeglądarki nie prezentują już informacji przy adresie URL strony na temat firmy, która wykupiła zainstalowany certyfikat (zielony pasek). Zniknął tym samym argument, że certyfikat SSL z Extended Validation buduje wizerunek najwyższego poziomu bezpieczeństwa. Certyfikat z walidacją domenową jest równie bezpieczny, a przy okazji przyniesie wiele benefitów wydajnościowych na stronie.

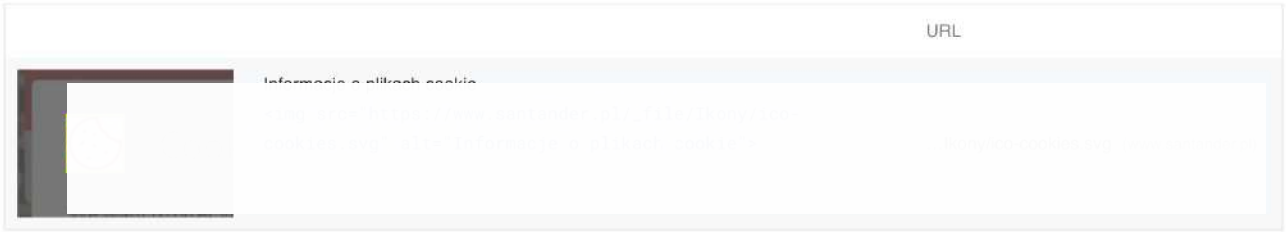
8.13 Minifikacja dokumentu HTML

Oprócz zminifikowania plików CSS i JS, wysoce rekomenduje się zastosowanie tej samej techniki dla kodu HTML. Pomoże to w uzyskaniu szybszego pobierania tego najważniejszego zasobu (od którego zależą wszystkie inne żądania na stronie) i krótszego czasu renderowania strony.

8.14 Brak ustalonej szerokości i wysokości ikonki w okienku o plikach cookies

Narzędzie Lighthouse wskazuje na brak atrybutów width/height dla obrazka-ikonki znajdującej się w okienku o polityce cookies. Rekomenduje się dodanie brakujących rozmiarów, celem uniknięcia potencjalnych przesunięć układu strony w trakcie ładowania omawianego zasobu (metryka CLS).

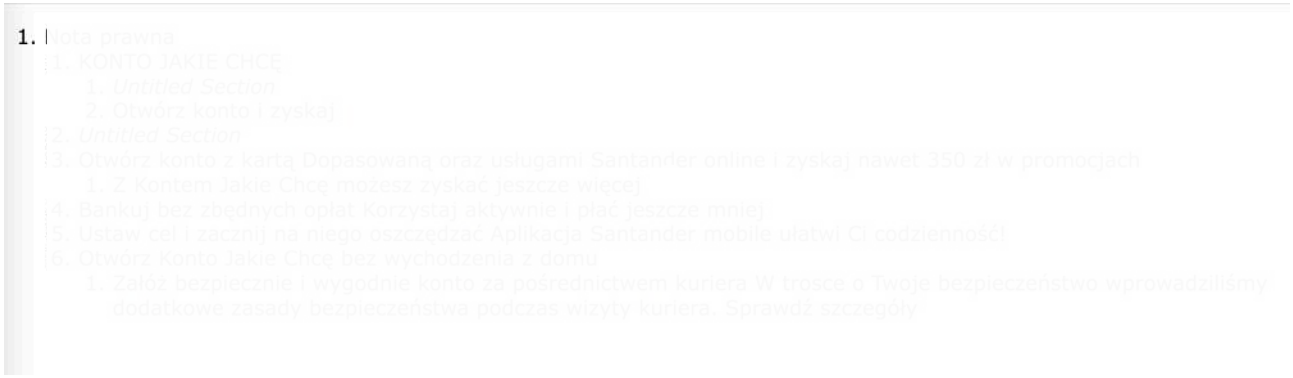
Dobrym pomysłem, by zaoszczędzić kolejne żądanie, byłoby osadzenie tego obrazka (SVG) bezpośrednio w kodzie HTML (<svg>).



2022 Studio Sidekicks Sp. z o.o.

8.15 Niepoprawna struktura nagłówków na stronie

Korzystając z narzędzia HTML5 Validator / Outliner, przydatnego do ustalenia struktury nagłówków na stronie, zaobserwowano, że na testowanej stronie nagłówki zostały dobrane nieprawidłowo, czyniąc problemy dla czytników ekranowych oraz SEO.



Rekomenduje się analizę treści na stronie pod kątem ważności nagłówków oraz treści i dopasowanie określonego nagłówka <h1>, będącego tytułem strony, a następnie nagłówków podrzędnych, związanych z kolejnymi sekcjami.

Na stronie zauważono kilka problemów w kodzie HTML, czyniąc go niepoprawnym. Zaleca się wygenerowanie raportu na stronie https://validator.w3.org/nu/, a następnie poprawę problemów według rekomendacji.

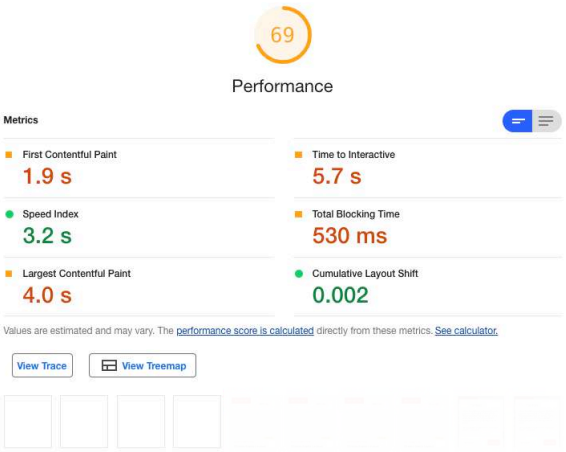
2022 Studio Sidekicks Sp. z o.o.

Oczekiwane rezultaty po zastosowaniu rekomendacji technicznych

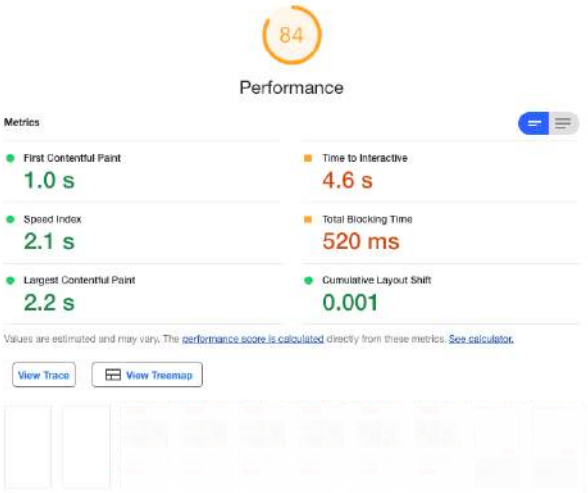
Z uwagi na dużą zależność skryptów zewnętrznych na stronie, które odpowiadają za jeden z najważniejszych czynników postrzegalnej wydajności - Largest Contentful Paint, a także interaktywność na stronie - ocena finalnych, spodziewanych rezultatów po zastosowaniu wszystkich poprawek wydajnościowych jest trudna. Im więcej rekomendowanych zmian nastąpi, tym lepszego rezultatu należy oczekiwać.

W przypadku braku zastosowania jakichkolwiek rekomendowanych poprawek dotyczących skryptów zewnętrznych tag managera i poleganiu jedynie na pozostałych sugestiach wydajnościowych, estymacje wskazują blisko 50% poprawę metryki First Contentful Paint, ok. 30% lepszy rezultat Speed Index, prawie 50% krótszy czas LCP oraz 20% krótki czas do pełnej interaktywności na stronie.

Przed:

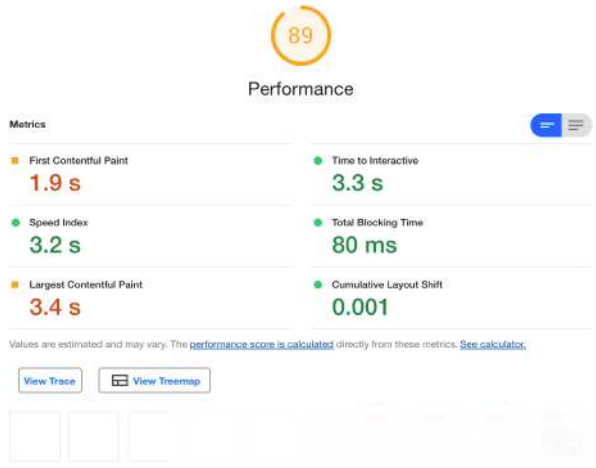


Po:

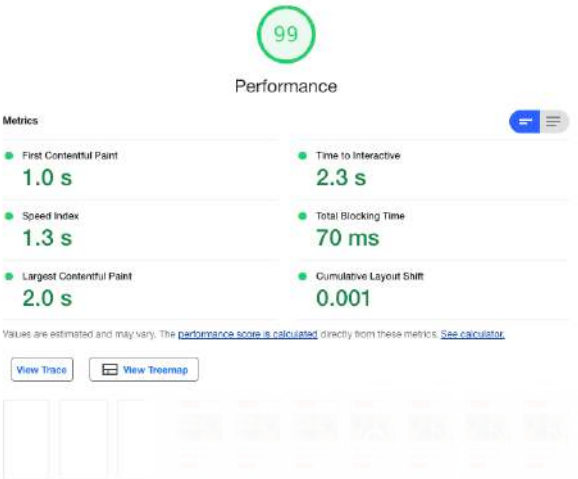


Gdyby testy wydajnościowe przed/po oprzeć o stronę **bez skryptów zewnętrznych**, rezultat końcowy byłby następujący:

Przed:



Po:



2022 Studio Sidekicks Sp. z o.o.

Jeżeli chciał(a)byś tworzyć raport taki, jak ten, wraz z rekomendacjami technicznymi…

…w dniach **27 - 30. czerwca 2022** odbywa się nabór do kursu Zoptymalizowany Frontend.

Pokażę Ci narzędzia i nauczę z nich korzystać. Przedstawię Ci techniki optymalizacyjne i jak optymalizować frontend według sprawdzonego procesu.

Wiedza i praktyka nabyta w tym kursie pozwoli Ci udoskonalić audyty SEO, które tworzysz.

Służę pomocą.

Zapraszam do kontaktu, w razie pytań odnośnie raportu lub kursu Zoptymalizowany Frontend.

Bartłomiej Miś

Web Dev Insider

Email:

kontakt@webdevinsider.pl

2022 Studio Sidekicks Sp. z o.o.