УНАПРЕЂЕЊЕ КВАЛИТЕТА WEB АПЛИКАЦИЈА ПРИМЕНОМ CORE WEB VITALS

Ненад Којић[1] Василије Василијевић [2] Милена Весић[3] Наталија Вугделија [4] Лука Лукић [5]

Резиме: Примарни циљ програмера web апликација је задовољство корисника али и позиција њихове апликације у поступку рангирања од стране web претраживача. Ове две ствари су могле да се реализују потпуно одвојено: стилизовање садржаја у складу са тренутним трендовима и писање позадинског кода и примена техника који повећавају SEO квалитет апликације. Параметри за рангирање су техничке карактеристике сајта. Неке од тих техничких карактеристика су сада директно повезане са квалитетом корисничког искуства нпр. брзина учитавања странице и респонзивност исте. Google тим, који је задужен за развој Chrome прегледача је објавио „Core Web Vitals“ као скуп метрика којима се мери брзина, респонзивност и визуелни квалитет апликација који директно утичу на корисничко искуство а индиректно на квалитет рангирања. У овом раду представиће се неки од најбитнијих елемената „Core Web Vitals“ као и начини њихове имплементације у тестној web апликацији у циљу повећања перформанси и квалитета корисничког искуства.

Кључне речи: мерење корисничког искуства, унапређење квалитета web апликација, SEO, Web Vitals

QUALITY IMPROVEMENT OF WEB APPLICATIONS BY APPLICATION OF CORE WEB VITALS

Abstract: The primary goal of web application developers is user satisfaction, but also the position of their application in the ranking process by the web search engine. These two things could be realized completely separately: stylizing the content in accordance with current trends and setting the background code and applying techniques that increase the SEO quality of the application. Ranking parameters are the technical characteristics of the site. Some of these technical characteristics are now directly related to the quality of the user experience, e.g. page loading speed and responsiveness. The Google team in charge of Chrome browser development has released “Core Web Vitals” as a set of metrics that measure the speed, responsiveness and visual quality of applications that directly affect the user experience and indirectly the ranking quality. This paper will present some of the most important elements of “Core Web Vitals” as well as ways to implement them in a test web application in order to increase performance and quality of user experience.

Key words: measuring of user experience, improving the quality of web applications, SEO, Web Vitales

1. УВОД

Често коришћене класификације друштва се дефинишу и у односу на техничка достигнућа човечанства. Након проналаска парне машине, и *прве индустријске револуције*, данас се налазимо у *четвртој индустријској револуцији* која је дефинисана као информатичко друштво [1]. Примарна карактеристика овог друштва је употреба технологије у свакодневним активностима. Додатно, уводи се и класификација деце који су од свог рођења у контакту са технологијом и информационим друштвом. Тако данас имамо пост миленијалце тј. генерацију Z [2]. Генерација Z, као будући примарни

[1] Професор, Академија техничко-уметничких струковних студија Београд одсек Висока школа за информационе и комуникационе технологије, Здравка Челара 16, Београд, nenad.kojic@ict.edu.rs

[2] Студент, Академија техничко-уметничких струковних студија Београд одсек Висока школа за информационе и комуникационе технологије, Здравка Челара 16, Београд, vasilije.vasilijevic.11.17@ict.edu.rs

[3] Сарадник у настави, Академија техничко-уметничких струковних студија Београд одсек Висока школа за информационе и комуникационе технологије, Здравка Челара 16, Београд, milena.vesic@ict.edu.rs

[4] Предавач, Академија техничко-уметничких струковних студија Београд одсек Висока школа за информационе и комуникационе технологије, Здравка Челара 16, Београд, natalija.vugdelija@ict.edu.rs

[5] Сарадник у настави, Академија техничко-уметничких струковних студија Београд одсек Висока школа за информационе и комуникационе технологије, Здравка Челара 16, Београд, luka.lukic@ict.edu.rs

конзумент информационих технологија, има карактеристику да се добро сналазе са технологијом, друштвеним мрежама и интеракцији са апликацијама. Како су web апликације постале изузетно доминантне постоји стална потреба да се исте перманентно усавршавају и прилагођавају тржишту које је све софистицираније. Генерација Z има високе критеријуме, висока очекивања од апликација и врло брзо се задовољавају текућим понудама. Све брже се тражи нешто ново, иновативно, другачије и производња web апликација се третира као и било који други производ на „масовном тржишту" [3].

Информационо друштво самим тим поставља врло строге критеријуме по којима вреднује различите web апликације и ови критеријуми су до скоро били индивидуално класификовани [4], [5]. Са друге стране, велики светски web претраживачи, желећи да својим корисницима, који претражују Интернет, понуде што квалитетнији резултат претраге имају своје критеријуме за рангирање web сајтова. Како је циљ сваког web програмера да његов производ буде што боље оцењен и што боље рангиран, они су приморани да се руководе критеријумима за рангирање. „Борба" у којој програмери теже да задовоље што више критеријума, док их web претраживачи стално модификују, је стална и неминовно проузрокује да се сајтови стално мењају, прилагођавају и унапређују [4], [5].

Из угла крајњих корисника, квалитет задовољства се мери субјективном оценом квалитета који зовемо корисничко искуство [6]. Иако већина корисника не зна да класификује и квантификује критеријуме које анализира, свако од нас зна да не жели апликације које се дуго учитавају, имају велике слике, немају све слике учитане, имају премало или превише информација, нису интерактиве, нису прилагођене за различите величине екрана и сл. [7]. Иако је до сада ово било само индивидуално корисничко искуство, са једне стране, а са друге смо имали критеријуме за рангирање који су дефинисани конкретним web претраживачем, ово сада постаје један јединствен термин.

Компанија Google прави велики број промена у својим критеријумима за вредновање квалитета web сајтова. Ове промене су врло честе и захтевају од програмера да стално унапређују своје кодове да би и даље било високо вредновани у поступку претраге. Google је развио нови алат који је интегрисан у Chrome прегледач и којим се врше јасна мерења појединих карактеристика корисничког искуства. Тако се досадашњи субјективни осећај брзине учитавања странице, интерактивности сајта, стабилности у раду и сл. у потпуности квантификују и постоје јасни критеријуми за њихово мерење и унапређење. Google је најавио да ће нови додатни критеријум за рангирање бити Core Web Vitals [8] и да ће постати и критеријум за појављивање на Google Top Stories. Када се ово очекује је неизвесно, али по најавама је то 2021. година јер је због Корона вируса програмерима дато додатно време да своје апликације прилагоде новим критеријумима. Иако у овом тренутку ови критеријуми нису у званичном скупу правила, Google анализе показују да странице које испуњавају ове нове критеријуме имају 24% мању вероватноћу да ће корисници да напусте те web локације. Употребом Core Web Vitals може се врло прецизно и једнозначно мерити квалитет корисничког искуства. Core Web Vitals за сада има три базна елемента на којима се заснива и који се квантификују [8]: Largest contentful paint (LCP), First input delay (FID) и Cumulative layout shift (CLS). Поред ових критеријума, постоје и додатни којима се може мерити и анализирати квалитет корисничког искуства.

У овом раду ће се представити појам и намена Core Web Vitals, анализирати његови параметри за квалитет корисничког искуства, објаснити појединачни елементи и на конкретном примеру показати како се могу имплементирати и мерити. Овај рад

организован је кроз четири поглавља: Увод, у коме је дата сврха увођења и намена Core Web Vitals, након чега је у другом поглављу објашњен сваки од кључних елемената Core Web Vitals и допунских елемената којима се утиче на квалитет корисничког искуства. У трећем поглављу су дати алати за мерење описаних елемената и резултати примене на примеру реалне web апликације. На крају је дат закључак и даље смернице у будућем раду, праћене коришћеном литературом.

2. CORE WEB VITALS

2.1. LCP (Largest Contentful Paint)

LCP представља време потребно да се учита највећи (главни) садржај странице [8]. Задовољавајући опсег у коме би требало да се прикаже највећи садржај странице је од 0 до 2,5 секунде од почетка учитавања странице. Пре увођења LCP програмери су на различите начине покушавали да измере ово време. Најчешће помоћу JavaScript догађаја „load" и „DOMContentLoaded", али они заправо нису добри јер не морају одговарати ономе што корисник заправо види на екрану. LCP је тренутак када прегледач почне формирати DOM и корисник почиње да види први знак учитавања.

Добар LCP резултат подразумева да се у првих 2,5 секунде од почетка учитавања странице види главни садржај странице. Шта тачно подразумева главни део или највећи садржај који носи корисничку информацију:

- <img> елементи
- <image> елементи унутар <svg> елемента
- <video> елементи
- елемент са позадинском сликом учитаном преко CSS-а
- елементи на нивоу блока који садрже текстуалне нодове

Величина елемента која се пријављује за највећи садржај је она величина која се налази у оквиру viewport-а (видљивог дела екрана). Уколико се елемент не налази у оквиру viewport-а ти делови се не рачунају у величину елемента. Што се тиче слика, за оне које су веће од димензија које су приказане на екрану биће узета у обзир само она величина која је у видљивом делу екрана. FCP је највећи садржајни елемент до тренутка када се не појави LCP.

Шта све утиче на LCP и зашто постоје разлике у мерењима код различитих сајтова? Ово је повезано са пуно фактора од којих су најбитнији: спор одзив сервера, JavaScript и CSS који блокирају рендеровање странице, време учитавања ресурса и рендеровање на страни клијента. С обзиром на те објективне околности, програмер може да примени неке од наредних решења за унапређење квалитета овог параметра: усмеравање корисника на најближи CDN (ово је изузетно добро решење којим се захтеви корисника усмеравају на њима најближе сервере за преузимање садржаја), кеширање (посебно у случају статичких делова сајта), читање HTML страница из кеша, инталација Service Workers-а.

Код приказивања критичних делова садржаја потребно је кодом јасно дефинисати да су ти делови они које приоритетно треба преузимати, као у примеру:

<link rel="preconnect" href="https://nekiURL.com">

Додатно, у ситуацији када је потребно брже обрађивање и памћење ранијег DNS lookup треба користити dns-prefetch као у примеру

<link rel="dns-prefetch" href="https:// nekiURL.com">

Препоруке су увек усмерене на редукцију CSS и JavaScript кода, а посебно да се раздвоје они који су потребни да би се страница учитала, па тек након тога да се учита остатак. За потребе редукције времена приказа за CSS је најбоље користити style таг у head секцији и на тај начин смањити LCP. Поред тога, минификацијом фајлова обавезно уклонити коментаре и непотребне размаке. Треба бити врло обазрив код JavaScript bundle алата попут WebPack-a који креирају један фајл што може проузроковати дуго учитавање странице. Тај проблем се може решити разбијањем кода на мање делове. То се обично ради LazyLoading методом тако да се раздвоји JS на мање делове најчешће по броју страница које желимо да имамо.

Слике су увек један од главних елемената web странице јер носе велику количину информација, и изазивају врло битан визуелни доживљај код корисника. Слике, са друге стране, имају доста већу тежину него текст, и самим тим лоше утичу на LCP. Зато се препоручује да се слике оптимизују и да буду што мање по тежини и да буду у новијим форматима (JPEG 2000 или WebP). Додатно, врло је пожељно урадити и CDN за коришћене слике.

Честа појава која утиче на корисничко искуство је промена фонта, која се примени на web страници неколико тренутака након учитавања фонта и то се визуелно увек примети. Ово се дешава због кашњења приликом учитавања фонта. Да би се ово спречило, појединим ресурсима треба дати приоритет приликом преузимања. Ово се може постићи додавањем атрибута rel="preload" тагу линк. Тако ће се преузимања фонта започети што је пре могуће и описани проблеми у корисничком исксутву се потпуно или већим делом редукују.

Ово се значајно може унапредити користећи gzip компресију на серверу да би се смањила величина фајла који се шаље као response од стране сервера.

Посебно интересантна је могућност избора типа садржаја у односу на квалитет интернет конекције тј. мреже која се користи за преузимање садржаја. Тако се корисницима са бољим квалитетом сигнала може приказати жељени видео фајл, док се корисницима са споријим протоком сигнала треба приказати слика уместо предвиђеног видео садржаја. С обзиром на то да апликација треба да има респонсиван, прилагодљив дизајн, за различите резолуције уређаја на којима се гледа, пожељно је приказати различите димензије и квалитет слика за различите уређаје и резолуције.

Употребом видео фајлова или слика, неминовно је да се ти фајлови морају прво добити од стране web сервера да би се приказали. Ово време се свим наведеним техникама може редуковати али на крају ипак постоји неко време потребно за учитавања. Повећању корисничког искуства у многоме доприноси употреба loading spinner-a којим се кориснику даје до знања да се садржај учитава и да треба да сачека. У супортном корисник може мислити да је то очекивани садржај и напустити страницу незадовољан јер није добио оно што је очекивао, пре него што се жељени садржај учита.

2.2. FID (First Input Delay)

FID означава кашњење, то јест то је време које је потребно страници да одговори на корисникову акцију од тренутка када корисник ступи у интеракцију са страницом [8]. Фаза у којој browser почне да приказује пикселе на екрану кориснику није много битна, колико то да корисник може одмах ступити у интеракцију са страницом.

FID је време потребно да цео систем реализује оно што је замишљено: од тренутка када корисник иницира неку акцију, на пример када кликне на дугме, до тренутка када је browser у стању да му на ту акцију одговори. Најчешће долази до кашњења јер је у browseru main thread заузет обрадом пристиглих мрежних захтева, па није у стању да одговори на корисников захтев.

За добро корисничко искуство web странице FID треба да буде мањи од 100 милисекунди. Одлагања FID-а се обично дешавају између FCP-а и TTI (Time To Interaction). Најчешћи елементи преко којих корисник комуницира са страницом су: елементи input тј. textarea, елемент select и елемент a.

Имајућу све ово у виду, могу се применити неке од следећих методологија којима се FID може унапредити:

- Треба оптимизовати third party кодове, нпр. за оглашавање.
- Треба смањити извршење JS кода разбијањем на мање делове.

Добра пракса је имати и оптимистичан приступ у преузимању ресурса. Да би имали ефекат брзе странице требало би пратити метрике које показују куда корисници најчешће иду приликом посете прве странице. Праћењем тих статистика можемо урадити оптимистично (унапред) преузимање ресурса да би побољшали корисничко искуство. Препорука је да се не претерује и не буде превише оптимистичан, већ треба поступати на основу јасних статистика.

2.3. CLS (Cumulative Layout Shift)

Учитавање елемената DOM-а се најчешће дешава асинхроно [8]. Ово за последицу има мање или веће померање, најчешће скоковито, појединих делова странице, након добијања response који стиже након претходног, а свој садржај треба да прикаже хронолошки пре претходно добијеног.

Да бисмо одредили нестабилност изгледа првенствено гледамо колико визуелног подручја је погођено нестабилношћу. Тај први концепт називамо Impact Region. Узмимо за пример елемент који је учитан и има своју позицију на страни, али се одједном тај елемент помера на доле када је реклама (Ad) учитана изнад тог елемента. Прво идентификујемо све DOM елементе који су померени из једног фрејма у други. Реклама (Ad) у овом случају није померени елемент због тога што није постојала у претходном фрејму. Након што идентификујемо све елементе којима је визуелна позиција промењена правимо унију претходног и садашњег стања позиције елемата и добијамо Impact Region. Величина тог региона у односу на viewport назива се Impact Fraction.

Други концепт је Move Distance. Поставља се питање колико далеко је померени елемент од претходног стања. Након што измеримо максимални померај елемента између два стања и упоредимо га са viewport-ом добијамо другу фракцију која се назива Distance Fraction.

Сада када имамо две фракције желимо да направимо укупан резултат тако што помножимо Impact Fraction и Distance Fraction и добијемо Layout Shift резултат за појединачан фрејм. Ако те резултате сумирамо од почетка учитавања странице до тренутка када корисник напусти страницу на крају добијамо Cumulative Layout Shift резултат.

Следећи проблем са којим се сусрећемо су анимације на страници. CSS нам дозвољава да анимирамо било које својство које желимо, али ако је то својство повезано са layout-ом као што је позиционирање, browser ће за сваки анимациони фрејм поново покренути (rerunning) свој layout engine и сваки фрејм ће узроковати Layout Shift. То може бити мали Layout Shift за сваки фрејм али Cumulative Layout Shift резултат може бити огроман, у зависности од тога колико је корисник провео времена на тој страници. Решење је да за позиционирање анимације користимо tranform својство да не би угрозили CLS резултат. Разлика између transform и position својства је у томе што transform не утиче на окружујући layout и такође browser може једноставно да покреће анимације на одвојеном thread-у од thread-а који покреће JavaScript.

Идеално, страница не би требала да има уопште Layout Shift, али то није могуће за већину апликација, тако да је прилично добар резултат CLS-а мањи од 0.1. Да би се ово достигло може се урадити следеће:

- Дефинисати димензије слика помоћу size атрибута
- Дефинисати fallback фонт у CSS-у да би се уколико је конекција спора учитао тај фонт уместо предвиђеног (користити font-style matching tool)
- Користити position: fixed за notice обавештења
- За велике промо Ad рекламе користити резервисан простор
- Обезбедити да потребан CSS буде учитан пре садржаја

3. АЛАТИ ЗА МЕРЕЊЕ CORE WEB VITALS И РЕЗУЛТАТИ РАДА

Гуглови популарни алати за web девелопере сада подржавају и мерење Core Web Vitals, пружајући помоћ за детекцију и решавање проблема корисничког искуства [8]. У ту групу алата спадају:

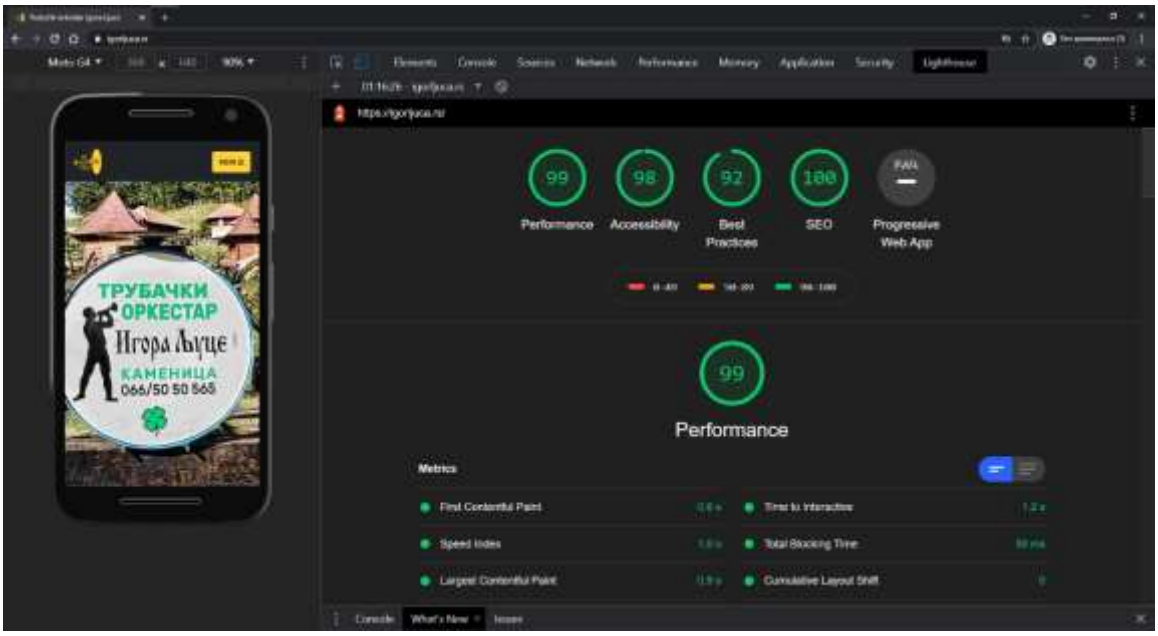
- Lighthouse (Lighthouse је аутоматизована алатка за ревизију web страница која помаже програмерима да дијагностикују проблеме и идентификују могућности за побољшање корисничког искуства својих web страница)
- Page Speed Insights (PageSpeed Insights(PSI) извештава о лабораторијским и теренским перформансама странице и на мобилним и на десктоп уређајима. PageSpeed Insights помаже у идентификовању могућности по страници да побољшају доживљај странице. У PSI-у можете јасно да видите да ли ваша страница испуњава прагове за добро искуство у свим Core Web Vitals вредностима на врху извештаја)
- Chrome DevTools (Chrome DevTools Performance panel има нови одељак Experience који може помоћи у откривању неочекиване промене распореда (CLS). Ово је корисно за проналажење и решавање проблема са визуелном нестабилношћу на вашој страници који доприносе померању кумулативног распореда)
- Search Console (Search Console пружа сјајан преглед група страница којима је потребна пажња)
- Chrome UX Report API (Извештај Chrome UX је јавни скуп података о стварном корисничком искуству на милионима web локација)
- Web Vitals JavaScript библиотека (Свака од метрика може се измерити помоћу JavaScript-а користећи стандардни web API.

Мерење помоћу кода се подудара са начином мерења у до сада поменутим алатима.

Ако желите да започнете процес оптимизације корисничког искуства, предлог је испратити следећи ток посла:

- Користити Search Console, нови Core Web Vitals извештај да би идентификовали групе страница којима је потребна оптимизација
- Када су странице идентификоване, користити Page Speed Insights (који покрећу Lighthouse и Chrome UX Report) да би дијагностиковали проблеме страница. Page Speed Insights (PSI) је доступан преко Search Console или директним уносом URL-а преко PSI web сајта
- Уколико је потребан custom dashboard користити Chrome UX Dashboard
- Уколико су потребна упутства за даљу оптимизацију користити web.dev's measure tool да би добили сет инструкција за оптимизацију, користећи PSI податке
- Користити Lighthouse CI за pull захтеве да би били сигурни да не постоји регресија када се промена извршава на продукцији

Ово је пример оптимизације на најјефтинијем shared хостингу, такође без коришћења HTTP2 протокола. Иако сервер није најбољи, могуће је остварити резултате пратећи до сада наведене кораке у оптимизацији.



Слика 1 – Lighthouse резултат

Слике: Оптимизација слика је омогућила смањење тежине слика. Коришћене су JPEG 2000 progressive и WebP формат. Са њима се постигла велика уштеда у тежини слика. Слике које су битне за учитавање постављене су на CDN. Препорука је да и остале слике буду на CDN-у али у овом примеру остале слике се налазе на хостингу. У зависности од уређаја, коришћени су media упити за слике различитих димензија применом `srcset` атрибута елемента img као `srcset="image-1000.jpg 1000w, image-2000.jpg 2000w"`. Додатно су уклоњени meta подаци са слика, који су иницијално били са фотоапарата, а који су имали велики габарит. За одлагање слика које се налазе ван viewport-а коришћена је lazy loading images техника. Помоћу lazy sizes реализовано је одлагање слика ван екрана. Урађена је минификација CSS-а и JS-а. Некоришћени CSS уклоњен је помоћу алата purgecss. Одрађена је компресија на серверу. Поред gzip може се користити и brotli. За ресурсе који су били потребни да би се сајт што пре приказао додати су атрибуте atribut rel="preconnect", rel="preload" или rel="dns-prefetch".

Потребно је урадити критичан пут рендера. Издвојен је критичан CSS помоћу алата који се зове PentHouse. Најбоље је критични CSS издвојити у style тагу у head секцији. Тако добијамо побољшан FCP. Урађено је кеширање ресурса који се не мењају често на серверу. Ресурси који блокирају рендер могу се у зависности од потребе одложити са defer да не би блокирали main thread.

Такође је потребно додати атрибут async у тагу script уколико желимо да се преузимање скрипте врши асинхроно.

4. ЗАКЉУЧАК

У раду је дат преглед Core Web Vitals и његова три најважнија елемента који представљају квантитативан начин мерења корисничког искуства у Chrome прегледачу. Указано је на њихову намену, сврху и начин употребе и показано је како на конкретном примеру треба предузети описане активности да би се добио релативно добар резултат корисничког искуства. Даљи рад ће бити усмерен на унапређењу критеријума и алата за повећање корисничког искуства и квалитета web страница.

5. ЛИТЕРАТУРА

[1] Hartwell, R. M. (2017). The industrial revolution and economic growth (Vol. 4). Taylor & Francis.

[2] Turner, A. (2015). Generation Z: Technology and social interest. The journal of individual Psychology, 71(2), 103-113.

[3] Lies, J. (2019). Marketing Intelligence and Big Data: Digital Marketing Techniques on their Way to Becoming Social Engineering Techniques in Marketing. International Journal of Interactive Multimedia & Artificial Intelligence, 5(5).

[4] Akram, U., Hui, P., Khan, M. K., Tanveer, Y., Mehmood, K., & Ahmad, W. (2018). How website quality affects online impulse buying. Asia Pacific Journal of Marketing and Logistics.

[5] Chen, X., Huang, Q., & Davison, R. M. (2017). The role of website quality and social capital in building buyers' loyalty. International Journal of Information Management, 37(1), 1563-1574.

[6] Kiruthika, J., Khaddaj, S., Greenhill, D., & Francik, J. (2016, August). User Experience design in web applications. In 2016 IEEE Intl Conference on Computational Science and Engineering (CSE) and IEEE Intl Conference on Embedded and Ubiquitous Computing (EUC) and 15th Intl Symposium on Distributed Computing and Applications for Business Engineering (DCABES) (pp. 642-646). IEEE.

[7] Kang, J. S., & Lee, Y. J. (2018). User experience of responsive web on multi-device environment. Journal of Digital Convergence, 16(11), 465-470.

[8] https://web.dev/vitals/