

Tuomo Vitikainen

Hakukoneoptimointi Javascript-viitekehyksellä toteutetussa verkkosivustossa

Tietotekniikan pro gradu -tutkielma

10. helmikuuta 2022

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Tuomo Vitikainen

Yhteystiedot: `tuomo.m.j.vitikainen@student.jyu.fi`

Ohjaajat: Lakanen, Antti-Jussi ja Zhidkikh, Denis

Työn nimi: Hakukoneoptimointi Javascript-viitekehyksellä toteutetussa verkkosivustossa

Title in English: Todo

Työ: Pro gradu -tutkielma

Opintosuunta: Kaikki opintosuunnat

Sivumäärä: 34+2

Tiivistelmä: Tiivistelmä

Avainsanat: SEO, SPA, Wordpress

Abstract: Tiivistelmä englanniksi

Keywords: SEO, SPA, Wordpress

Termiluettelo

Termi1

Termi1.

Termi2

Termi2

Kuviot

Kuvio 1. Sivustojen renderöinti	10
Kuvio 2. DSMR-prosessimalli	12

Taulukot

Taulukko 1. Komentomuutokset gradu2:sta gradu3:een	29
--	----

Sisältö

1	JOHDANTO	1
2	HAKUKONEET JA HAKUKONEOPTIMOINTI	2
2.1	Hakukoneoptimointi	3
2.2	Hakukoneoptimoinnin keinot	4
2.3	Sisäinen hakukoneoptimointi	4
2.3.1	Avainsanojen optimointi	4
2.3.2	Sisältö	5
2.4	Ulkoinen hakukoneoptimointi	6
2.5	Tekninen hakukoneoptimointi	6
2.5.1	Robots.txt	6
2.5.2	Sitemap.xml	6
2.5.3	Sivustokartta	7
2.5.4	Sivun alkuperäinen osoite	7
2.5.5	Alt-attribuutti	7
2.5.6	Sisäinen linkitys	7
2.5.7	Sivuston nopeus	8
2.6	WordPress	8
2.5.1	Hakukoneoptimointi Wordpressissä	8
2.5.1	Pehmeät 404-sivut	8
2.6	SPA-verkkosivut	8
3	TUTKIMUS	11
3.1	Suunnittelutiede	11
3.2	Suunnittelutiede tutkimuksessa	13
3.2.1	Ongelman havainnollistaminen ja motivaatio	13
3.2.2	Tavoitteet ja vaatimukset	13
3.2.3	Mittausmenetelmät	14
3.3	Artefaktin tekninen toteutus	14
3.3.1	Sitemap ja robots -tiedostojen luominen	16
3.3.2	Sitemap ja Robots -tiedostojen luominen	17
3.3.2	Metatietojen tuominen sisällönhallintajärjestelmästä	18
3.3.3	Alkuperäisen sisällön osoittaminen	19
3.3.4	404-tapahtuman käsittely	20
3.4	Sivuston sisällöt	21
3.4.1	Sisältöjen syöttäminen	21
4	TULOKSET/ANALYYSI	23
5	YHTEENVETO	24
	LÄHTEET	25
	LIITTEET	29

A	Siirtyminen gradu2:sta gradu3:een	29
B	Harvemmin tarvittavat ominaisuudet	30

1 Johdanto

Internet sisältää miljoonia eri verkkosivuja ja suuria määriä tietoja, mistä johtuen hakukoneet pakollinen työkalu internetin käyttäjälle (Levene 2011). Hakukoneet ovat tärkeitä työkaluja internetin käyttäjien keskuudessa, sillä 93 prosenttia internetin liikenteestä tapahtuu hakukoneiden kautta (Egri ja Bayrak 2014). Hakukoneisiin liittyy olennaisesti myös verkkosivujen hakukoneoptimointi. Hakukoneoptimoinnilla tarkoitetaan toimia, joilla verkkosivun näkyvyys paranee hakukoneissa Vuonna 2018 tehdyssä tutkimuksessa todettiin, että hakukoneoptimointi auttaa lisäämään yrityksen markkinaosuutta sekä parantamaan markkinoijan brändipääomaa, jolla taas on vaikutusta esimerkiksi tuotetietoisuuteen ja ostopäätöksiin (Bhandari ja Bansal 2018).

SPA-toteutuksia on kuitenkin moitittu huonosta hakukonenäkyvyydestä, ja esimerkiksi suomalainen webkehitysyhteisö vierityspalkki.fi on huomionnut asian artikkelissaan. Siksi onkin tärkeää tutkia, kuinka SPA-verkkosivustoissa hakukoneoptimointi kuuluisi tehdä, jotta sivu sijoittuisi hyvin hakukoneissa. SPA-sivustojen toteutustavan erotessa perinteisestä se tuo mukanaan myös eroavaisuuksia hakukoneoptimointiin, sillä esimerkiksi dynaaminen sisältö ei ole aina suoraan näkyvissä hakuboteille (Hammer, Bratterud ja Fagernes 2013). Muita käsiteltäviä erikoistapauksia on esimerkiksi 404-virhesivun näyttäminen. Pehmeäksi 404 sivuksi kutsutaan sivua, joka kertoo että sisältöä ei löydy mutta palauttaa silti 200 OK -statuskoodin (Prieto, Álvarez ja Cacheda 2013). Tämä aiheuttaa ongelmia hakukoneoptimoinnin kanssa.

Olen työssäni havainnut, että SPA-toteutukset ovat yleistyneet myös verkkosivustoissa, joihin useimmiten päädytään hakukoneiden kautta.

2 Hakukoneet ja hakukoneoptimointi

Hakukoneella tarkoitetaan web-ohjelmaa, joka hakee käyttäjän hakusanojen perusteella verkkosivuja internetistä. Hakukoneilla on omat tietokantansa, joiden avulla ne palauttavat hakutuloksia. Internetin alkuaikoina ennen hakukoneita ylläpidettiin käsin listaa internetin palvelimista. Kuitenkin palvelinten yleistettyä ajantasaisen listan ylläpitäminen muuttui vaikeammaksi, joten tarve hakukoneille oli syntynyt. Ensimmäinen hakukone, nimeltään Archie, kehitettiin vuonna 1990 Alan Emtagen, Bill Heelan ja J. Peter Deutschin toimesta. Sen jälkeen hakukoneita on kehitetty kymmeniä. (Seymour, Frantsvog, Kumar ym. 2011) Googlen hakukone kehitettiin vuonna 1998, ja statcounter.com sivuston mukaan sillä on 86 prosentin markkinaosuus kaikista hakukoneista vuonna 2021. Toisena on hakukone nimeltään Bing 7.2 prosentin markkinaosuudella. (Statcounter 2021)

~~–Toimintaperiaate–~~

Hakukoneen toimintaperiaate perustuu internetissä olevien verkkosivustojen läpikäyntiin ja niistä löytyvän datan indeksointiin. Hakukone koostuu neljästä eri pääkomponentista, jotka ovat:

- Ryömijä (eng. crawler)
- Hakuindeksi (eng. search index)
- Kyselymoottori (eng. query engine)
- Käyttöliittymä

Käyttäjälle näkyvä osa hakukoneesta on käyttöliittymä. Käyttäjä voi syöttää hakukoneeseen yhden tai useampia hakusanoja. Kun kysely on tehty, hakutulokset ilmestyvät käyttöliittymään hakukoneen päättämässä järjestyksessä.

Ryömijällä (eng. crawler) tarkoitetaan järjestelmää, joka joukkolataa verkkosivustoja. Niillä on monia käyttötarkoituksia, mutta erityisen tärkeitä ne ovat hakukoneissa. Ryömijät toimivat automaattisesti niille annetun algoritmin perusteella, jonka perusajatuksena on ladata annettujen URL-osoitteiden sivut ja sivujen sisältämät hyperlinkit ja jatkaa hyperlinkkien osoittamille sivuille. (Olston ja Najork 2010)

Ryömijöiden lataamien verkkosivujen hakusanat tallennetaan, eli niistä luodaan tietokanta eli hakuindeksi. Hakuindeksia tarvitaan, jotta kyselyt voivat palauttaa relevantteja, parametrien mukaisia hakutuloksia. Hakuindeksi sisältää kaikki verkkosivuilta löytyvät sanat aakkosjärjestyksessä, ja tätä tiedostoa kutsutaan indeksitiedostoksi. Jokainen sana sisältää viitteen sivustoihin, joilta kyseinen sana löytyy. Tätä tiedostoa taas kutsutaan postituslistaksi. (eng. posting list). Hakuindeksi sisältää myös erillisen linkkitietokannan. Linkkitietokannan avulla voidaan hahmottaa webin rakennetta ja lisäksi päätellä sen kattavuutta. Linkkien avulla tehtävä analyysi voi vaikuttaa sivuston sijoitukseen hakukoneissa. (Levene 2011)

Hakumoottori sisältää algoritmit, jonka avulla verkkosivuja haetaan indekseistä. Niiden tarkat toimintaperiaatteet eivät ole yleisesti julkaistua tietoa, sillä tällöin hakukonesijoitusta voisi manipuloida paremmaksi optimoimalla sivusto erityisesti hakukoneen algoritmia varten (Levene 2011). Googlen käyttämä verkkosivujen arvostelualgoritmi on nimeltään PageRank, jonka on raportoitu sisältävän yli 100 muuttujaa. (Krrabaj, Baxhaku ja Sadrijaj 2017).. Esimerkiksi eräs keino on sivuston aihepiiriin liittymättömien sanojen viljely sisällössä tai metatiedoissa paremman hakukonesijoituksen toivossa. Useat hakukoneet kuitenkin tarkistavat tämän ja saattavat rankaista siitä poistamalla sivuston hakuindekseistä. Tällaista toimintaa kutsutaan Black Hat hakukoneoptimoinniksi. (Enache ym. 2014)

~~-Orgaaniset hakutulokset vs mainonta-~~

Hakukonenäkyvyydellä on tärkeä rooli yritysten markkinoinnissa. Katseentunnistusjärjestelmiä hyödyntäen on tutkimuksissa pystytty osoittamaan, että ihmiset mieluiten klikkaavat ensimmäisiä tuloksia (Lewandowski ym. 2018). On todettu, että noin 94 prosenttia ihmisistä katsovat vain ensimmäisen sivun hakutulokset läpi. 63 prosenttia ihmisistä taas katsoo vain 3 ensimmäistä hakutulosta. Ihmiset lähtökohtaisesti ennemmin muuttavat hakusanoja ja hakevat uudestaan, sen sijasta että etsisivät tuloksia toiselta sivulta. (Sharma ym. 2019)

2.1 Hakukoneoptimointi

Hakukoneoptimoinnilla tarkoitetaan verkkosivulle tehtyjä toimia, joilla voidaan parantaa sivun sijoitusta ja näkyvyyttä hakukoneissa. Sijoitukseen vaikuttavat monet eri tekijät, joista valtaosaan sivuston ylläpitäjä pystyy itse vaikuttamaan joko sivuston sisällön tai teknisen

toteutuksen kautta.

Kaikki sivustolle tehtävät muutokset eivät kuitenkaan ole hyväksytyjä, vaan hakukone saattaa rankaista vääristä toimista. Esimerkiksi Google saattaa poistaa selkeästi vilpillisiä tekniikoita käyttäneen sivuston hakemistostaan, ja lievemmistä rikkeistä antaa varoituksen. Tällaisia ei-suositulta keinoja kutsutaan black hat -tekniikoiksi. White hat tekniikat ovat sallittuja tekniikoita, jotka ovat kyseisen hakukoneen ohjeiston mukaisia. Black hat tekniikoista useimmiten käytetty on avainsanojen viljely, jossa sivustolle sijoitetaan suuri määrä hakusanoja paremman näkyvyyden toivossa. Hakusanat ovat sijoiteltu siten, että ne ovat vain hakukoneen nähtävissä, esimerkiksi jonkin muun elementin takana tai sijoiteltuna kuvien alt-tageihin. Hakukoneet kuitenkin nykyään erottavat sisällön jotka eivät selkeästi ole normaalin käyttäjän nähtävissä. (Malaga 2010) Toinen esimerkki tällaisesta tekniikasta on luoda ylimääräisiä sivustoja vain siinä käyttötarkoituksessa, että niistä lähtee linkkejä kohdesivustolle. (Sharma ym. 2019)

2.2 Hakukoneoptimoinnin keinot

Hakukoneoptimointi jaetaan usein kahteen eri osa-alueeseen, sivun sisäiseen ja sivun ulkoiseen hakukoneoptimointiin. Sivun sisäinen hakukoneoptimointi kattaa kaikki ne keinot, jotka sivuston ylläpitäjä itse pystyy tekemään hakukonenäkyvyytensä hyväksi. Tähän lukeutuu sivuston tekninen toteutus sekä kaikki sisältö, kuten tekstit, kuvat ja hakusanat. (Sharma ym. 2019). Sivuston tekninen toteutus ratkaisee myöskin sivuston nopeuden. Vuodesta 2010 lähtien nopeus on ollut arvostelutekijä Googlen hakukoneella tehdyissä työpöytähauissa, ja vuodesta 2018 lähtien myöskin mobiililaitteella tehdyissä hauissa (Osmani ja Grigorik 2019).

2.3 Sisäinen hakukoneoptimointi

2.3.1 Avainsanojen optimointi

Sisällönsuunnittelun alkuvaiheissa tulisi ottaa huomioon käytettävät avainsanat. Sivuston tulisi sisältää teemaan ja sivuston aihepiiriin sopivia avainsanoja, sillä käyttäjät käyttävät niitä

hakukoneissa verkkosivustoja hakiessaan. Onkin huomioitava, että avainsanat on liitettävä verkkosivuston muuhun sisältöön. Avainsanoja mietittäessä on hyvä kiinnittää huomiota et-
tä ne eivät olisi liian laajoja, eivätkä kuitenkaan liian vaikeita. (Hui ym. 2012) Krrabajin,
Baxhakun, Fesalin ja Dukagjin vuonna 2017 tekemässä tutkimuksessa saatiin hyviä tuloksia
hakukonenäkyvyyden suhteen käyttämällä todennäköisiä hakufraaseja, jotka samalla sopivat
sivuston aihepiiriin. (Krrabaj, Baxhaku ja Sadrijaj 2017).

Avainsanoja ei myöskään saa olla liikaa määrällisesti sivustolla, sillä se voidaan tulkita avain-
sanojen viljelyksi, jolloin sivuston sijoitus hakukoneissa voi jopa laskea. Avainsanatiheys
määritetään yleensä prosentteina suhteessa muuhun sisältöön. Hui ym. 2012 mukaan sopiva
tiheys olisi noin 3-8 prosenttia.

Avainsanojen sijoittelulle on myös syytä kiinnittää huomiota. Olennaisia paikkoja hakuko-
neoptimoinnin kannalta on muun muassa sivuston otsikko, metatiedot, HTML-otsikot ja
linkkitekstit. (Hui ym. 2012). Metatietoihin voi sivun lyhyen kuvauksen lisäksi asettaa tee-
maan sopivia hakusanoja (Davis 2006). HTML-merkintäkielessä on kuusi eri otsikkotyypp-
piä, joista H1-tagin on sivun pääotsikko, joka on merkitsevin ja joita kuuluu olla vain yksi
per sivu. H6-tagilla taas merkitään vähiten merkitsevä otsikko. Otsikkojen käyttö luo myös
sivun rakenteeseen hierarkiaa ja selkeän rakenteen. (Gupta ym. 2016) Tärkeä informa-
tio kannattaa aina kirjoittaa tekstinä kuvien sijasta, jolloin sisältö on myös hakukonebottien
luettavissa (Davis 2006).

Verkkosivuston domain ja sen pääte vaikuttavat myöskin hakukonenäkyvyyteen. Esimerkik-
si sana "video"domainin nimessä tuo etuja, kun käyttäjät hakevat verkkosivustoja liittyen
videoihin. Yleiset domain-päätteet kuten .org, .net ja .en voivat vaikuttaa hakukonenäkyvyy-
teen positiivisesti. (Hui ym. 2012)

2.3.2 Sisältö

Yleinen uskomus on, että sivuston sisältöjen päivittäminen auttaa hakukonesijoitukseen. On
kuitenkin todettu, että sivusto voi sijoittua korkealle hakutuloksissa vaikka sisältöä ei aktii-
visesti päivitetäisikään. Sisällön uudistuminen voi kuitenkin auttaa sivustoa saavuttamaan
paremman hakukonenäkyvyyden, mutta sijoituksen pitäminen ei välttämättä vaadi tuoretta

sisältöä. (Malaga 2010)

2.4 Ulkoinen hakukoneoptimointi

Hakukoneet ottavat huomioon sivustoon osoittavat linkit, jotka ovat olennainen osa hakukoneoptimointia kun tavoitellaan hyvää sijoitusta. Kuitenkaan pelkkä osoittavien linkkien määrä ei ole tärkein tekijä, vaan hakukoneista ainakin Googlen on todettu ottavan huomioon myös linkkien laatu. Google arvioi verkkosivustoja Page Rank -algoritmillla, ja arvostaa enemmän linkityksiä jotka tulevat verkkosivustoilta joilla on korkea Page Rank -pisteitys. (Malaga 2010). Ulkoisten linkitysten määrä yhdessä muodostavat linkkisuosion, ja suuresta suosiosta voidaan päätellä kyseessä olevan tärkeä sivu. Ulkoiset linkitykset esimerkiksi bannerien tai mainosten muodossa eivät ole algoritmien mielestä niin arvostettavia kuin tekstiviittaukset. (Patil Swati, Pawar ja Patil Ajay 2013)

2.5 Tekninen hakukoneoptimointi

Teknisillä optimointitekniikoilla tarkoitetaan toimia, jotka liittyvät verkkosivuston tekniseen toteutukseen. Laadukkaalla teknisellä toteutuksella varmistetaan muun muassa se, että hakukonebotit löytävät kaikki sivut jotka halutaan indeksoida.

2.5.1 Robots.txt

Robots.txt on palvelimelle asetettava tiedosto, joka asetetaan hakemiston juureen. Sen avulla voidaan kertoa hakukoneboteille, mitkä tiedostot niiden tulisi indeksoida ja mitkä ei. Esimerkiksi CSS-tyylitiedostoja, kuvakansioita tai käyttäjähallintasivuja ei usein haluta indeksoida, jotta ne eivät tulisi vastaan hakutuloksissa tavallisille käyttäjille. (Kumar 2013)

2.5.2 Sitemap.xml

Sitemap.xml on tiedosto, jonka avulla verkkosivuston ylläpitäjä voi informoida hakukoneille indeksoitavien sivujen osoitteet (Levene 2011). Googlen mukaan sivukartan luominen ja toimittaminen voi olla varsinkin silloin oleellista, kun sivusto on uusi tai kokoluokaltaan

poikkeuksellisen suuri, tai sisältää sisältöä joka on eristyksissä tai mihin on vähänlaisesti sisäisiä linkityksiä (Google 2021b).

2.5.3 Sivustokartta

Sivustokartalla tarkoitetaan XML-muotoista tiedostoa, joka sisältää sivuston sisäisen rakenteen, kuten sivujen väliset linkitykset. Sivustokartan voi itse luoda ja sen jälkeen toimittaa hakukoneelle, jotta sivuston indeksointi nopeutuu. (Kumar 2013)

2.5.4 Sivun alkuperäinen osoite

Verkkosivuston sivuihin voi osoittaa useita eri osotteita. Hakukoneryömiät käsittävät silloin sivun olevan sama sivu monistettuna. Jos hakukoneelle ei erikseen kerro sivun alkuperäistä osoitetta, se päättää indeksoitavan osoitteen itse, jolloin hakemistoon voi joutua väärä osoite. Alkuperäisen ja indeksoitavaksi haluttavan sivun voi merkitä erityisellä rel=canonical -elementillä. (Google 2021a; Ohye ja Kupke 2012)

2.5.5 Alt-attribuutti

Alt-attribuutilla tarkoitetaan kuvan tekstivastinetta, jonka tulisi olla lyhyt selostus kuvan sisällöstä. Hakukoneet eivät näe kuvia, mutta alt-attribuutit ovat niiden indeksoitavissa. Alt-attribuuttien käyttö on erityisen tärkeää varsinkin silloin, kun kuvaa käytetään linkkinä. Attribuutin käyttö on tärkeää myöskin sivuston saavutettavuuden kannalta, sillä sen avulla myös ruudunlukuohjelmia käyttävät ihmiset saavat tietoonsa kuvan sisältämän informaation. (Slatin 2001)

2.5.6 Sisäinen linkitys

Verkkosivuston omia sivuja tulisi myös linkittää toisiinsa. Tämän voi toteuttaa esimerkiksi artikkelin loppuun lisättävällä artikkelinostolla, jossa on listattu muita vastaavanlaisia artikkeleita sivustolla, jotka saattaisivat kiinnostaa lukijaa. Sisäisten linkitysten avulla ryömiät pääsevät indeksoimaan verkkosivuston eri sivuja, ja lisäksi se helpottaa myöskin sivuston käyttäjien liikkumista sivustolla. Jokaisella sivulla tulisi myöskin olla linkki, joka johtaa ta-

kaisin etusivulle. (Hui ym. 2012)

2.5.7 Sivuston nopeus

Sivuston nopea toiminta mahdollistaa sen, että käyttäjä viettää sivustolla mahdollisimman pitkiä aikoja. Lisäksi sivuston nopeus on Googlessa sivuston pisteytykseen vaikuttava tekijä (Egri ja Bayrak 2014; Osmani ja Grigorik 2019). Sivuston nopeuteen vaikuttaa suuri määrä tekijöitä, lähtien sivuston toteuttamistavasta. Verkkosivustoa voi optimoida nopeamaksi esimerkiksi käyttämällä vähän tilaa vieviä kuvaformaatteja, minifioimalla HTML-, JavaScript- ja tyylitiedostot ja poistamalla kaikki käyttämättä jäänyt lähdekoodi. (Egri ja Bayrak 2014).

2.6 Wordpress

2.5.1 Hakukoneoptimointi Wordpressissä

~~–Renderöinti-palvelimella vs-selaimella–~~

2.5.1 Pehmeät 404-sivut

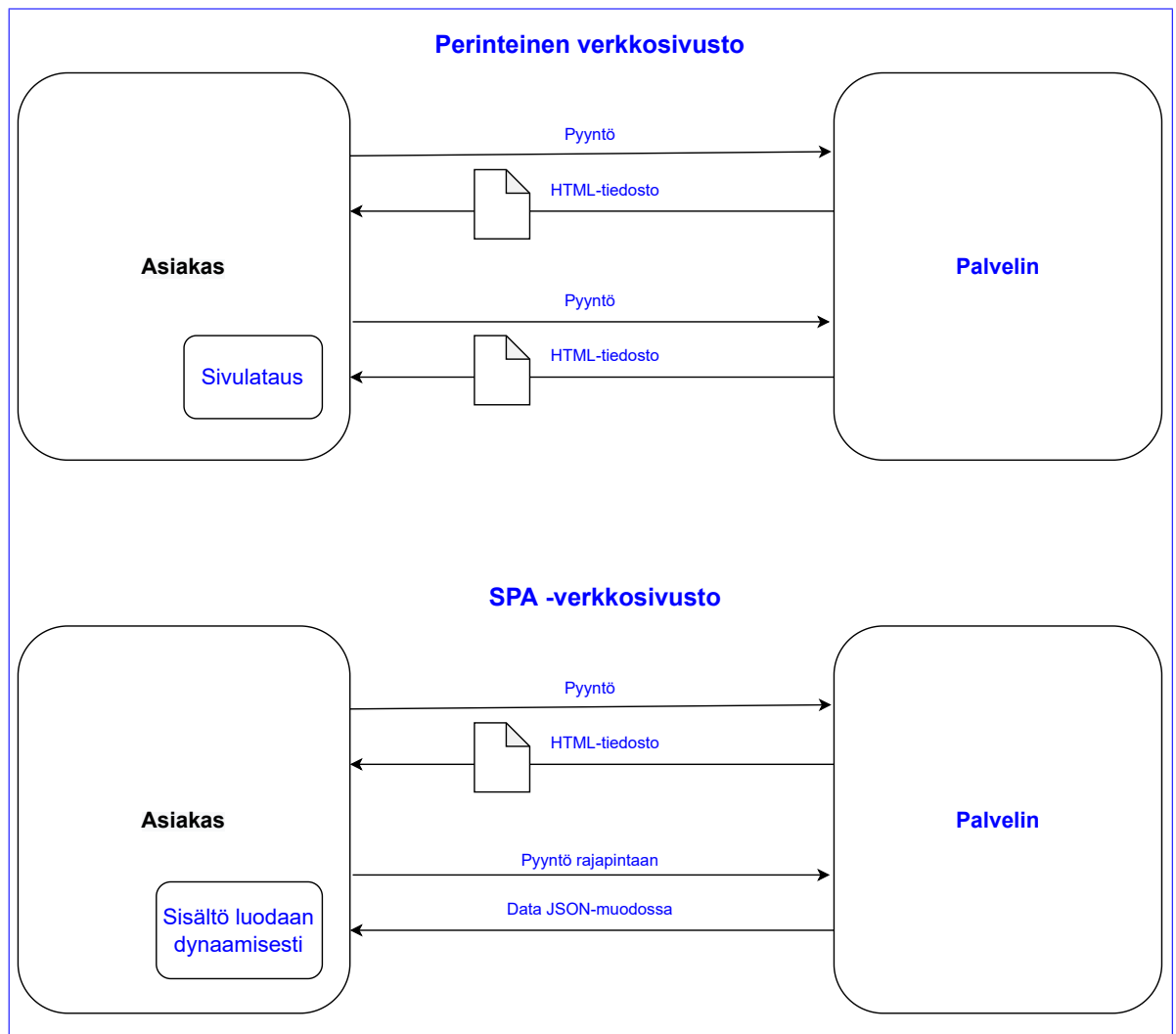
HTTP-statuskoodilla 404 ilmoitetaan, että haettua sivua ei löytynyt palvelimelta. Pehmeällä 404-virheellä tarkoitetaan tilannetta, jossa sivulla ei ole sisältöä tai se on esimerkiksi virheviesti joka ilmoittaa sivun puuttumisesta, mutta palvelin palauttaa tästä johtuvat erot huolimatta onnistuneen pyynnön HTTP-statuskoodin 200. (Meneses, Furuta ja Shipman 2012; Prieto, Álvarez ja Cacheda 2014) Tällöin sivuston käyttäjä saa tietoonsa että sisältöä ei löytynyt, mutta hakukoneryöimijät indeksoivat sivun virheellisestä statuskoodista johtuen ja käyttävän turhaan resursseja. 404-virhesivusta kannattaa tehdä selkeästi sivustoon kuuluvan sivu ja lisätä linkki, josta käyttäjä pääsee takaisin etusivulle. (Google 2022)

2.6 SPA-verkkosivut

Tämä osio käsittelee perinteisten verkkosivustojen ja niin kutsuttujen single page application -sivujen arkkitehtuuria sekä niiden eroja. Perinteisissä verkkosivustoissa käyttäjän halutessa

vaihtaa sivua, selain lähettää pyynnön palvelimelle. Jos haettu resurssi löytyy palvelimelta, palvelin palauttaa onnistuneen pyynnön HTTP-koodin 200 ja haetun verkkosivun. (Fiel-
ding, Gettys ym. 1998). Kun selain lopulta muodostaa lopullisen sivunäkymän, se päivittyy
käyttäjän selaimeen sivulatauksen myötä (Scott Jr 2015). Yksinkertaisimmillaan verkkosivusto
rakentuu HTML-dokumentista, CSS-tyylitiedostosta ja mahdollisesta JavaScript-tiedostosta,
jotka selain osaa kääntää verkkosivustoksi (Goodman 2002). Tällaisella yksinkertaisella
verkkosivustolla ryömijä osaa helposti käydä HTML-dokumentin läpi ja jatkaa eteenpäin
sieltä löytyvistä linkeistä.

Single page applicationilla tarkoitetaan tarkoitetaan verkkosivua, jossa näytettävä sisältö
muodostetaan dynaamisesti HTML-pohjaan. Ensimmäiset SPA-toteutukset pohjautuivat Java
ja Flash teknologioihin, mutta nykyään modernit kirjastot toimivat nykyään yksinomaan
JavaScriptillä (Mikowski ja Powell 2013). Käyttäjän vaihtaessa sivua uusi sisältö luodaan
dynaamisesti samaan noudettuun verkkosivun runkoon. Tällöin ei tapahtu myöskään sivulatauksia
ja samalla sovelluksen toiminta on nopeampaa. SPA-verkkosivuilla haetaankin yleensä sovellusmaista,
nopeaa käyttökokemusta. (Scott Jr 2015) Eri kirjastoissa on eroja, renderöidäänkö sisältö
verkkosivulle ensimmäisen kerran palvelimella vai käyttäjän selaimessa. Yleisesti tunnetuista
SPA-kirjastoista esimerkiksi Vue renderöi oletuksena näytettävän sisällön käyttäjän selaimella (Vuejs
2021). Tämä tarkoittaa sitä, että palvelimella on nähtävissä vain lähes tyhjä HTML-dokumentti,
jossa on osoitettuna tietyllä tunnisteella paikka johon dynaaminen sisältö luodaan. Tämä
aiheuttaa ongelmia hakukoneiden ryömijöiden läpikäydessä sivustoa, sillä palvelimella ei
ole nähtävissä mitään käyttäjän sisällönhallinnassa syöttämiä tietoja, joita indeksoida. Yksi
tapa ratkaista tämä ongelma on käyttää kirjastoa, joka tarjoaa mahdollisuuden renderöidä
verkkosivu palvelimella valmiiksi (Iskandar ym. 2020). Koska verkkosivu muodostetaan jo
palvelimella, ryömijät pystyvät käymään sen läpi kuten staattisen verkkosivun. Esimerkiksi
Vue-kirjastoon perustuva Nuxt tarjoaa mahdollisuuden renderöidä sivusto palvelimella (Vuejs
2021).



Kuvio 1. Sivustojen renderöinti

3 Tutkimus

Tässä luvussa käsitellään suunnittelutieteen prosessimallia sekä sen käyttöä tässä tutkimuksessa.

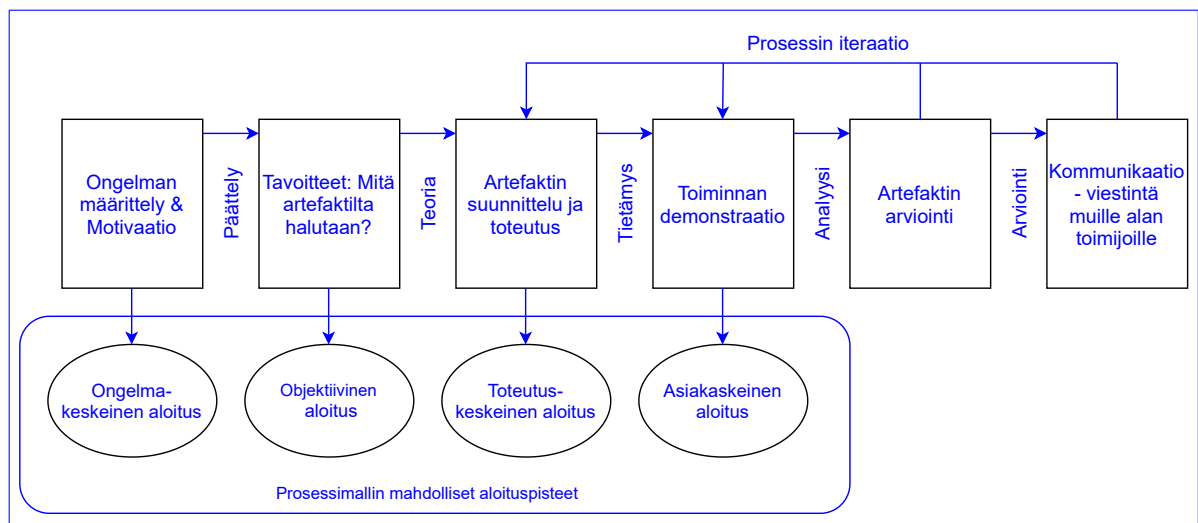
3.1 Suunnittelutiede

Suunnittelutiede määritellään Peffers ym. 2007 tutkimusartikkelissa seuraavalla tavalla: "Suunnittelutieteen avulla luodaan ja kehitetään artefakteja, joilla pyritään ratkaisemaan tunnistettuja ongelmia." (Peffers ym. 2007). Suunnittelutiedettä voidaan kutsua ongelmanratkaisuparadigmaksi. Artefaktiksi voidaan määritellä lähes mikä tahansa suunniteltu objekti, jonka avulla pyritään ymmärtämään tutkimusongelmaa ja tuottamaan siihen ratkaisuja. Artefaktin kehitys onkin prosessi, jossa hyödynnetään jo olemassa olevia teorioita sekä tutkimuksen aikana ilmennettyä uutta tietoa (Peffers ym. 2007).

Jotta ongelmia voidaan ratkaista, suunnitelmia arvioida ja tuloksia esittää, suunnittelutieteisiin kuuluu ennalta määritelty prosessi. Prosessin eri vaiheiden avulla suunnitteluongelmia voidaan lähestyä järjestelmällisesti tarkastellen tavoitteita ja asetettuja vaatimuksia, jotta voidaan päästä kohti parempia ratkaisuja. (Peffers ym. 2007).

Tässä tutkielmassa käytetään Peffers ym. 2007 esittämää suunnittelutieteen prosessimallia. Tätä mallia varten he tarkastelivat seitsemää aiemmin tehtyä suunnittelutiedettä käsittelevää tutkimusta, joiden pohjalta he koostivat oman prosessimallinsa. Prosessi on jaettu kuuteen eri aktiviteettiin, jotka on suunniteltu suoritettavan peräkkäisessä järjestyksessä. Todellisuudessa prosessimalli ei aktiviteettien peräkkäisyydestä huolimatta edellytä niiden kaikkien suorittamista, vaan mallia voi lähteä suorittamaan useammasta kohdasta. Peffers ym. 2007 ovat esittäneet mallissaan neljä eri aloitustuskohtaa, jotka sopivat eri kontekstissa oleville suunnittelutieteen hyödyntäjille.

Ensimmäinen aktiviteetti on sekä ongelman että ratkaisun arvon määrittely. Ongelman määrittely on tärkeää, sillä sen avulla kehitetään artefaktia ja sen ominaisuuksia vastaamaan ongelmaa mahdollisimman hyvin, jonka takia voi olla hyödyllistä jakaa ongelma pienempiin



Kuvio 2. DSMR-prosessimalli ~~÷PIIRRÄ KUVA~~ (Peffer ym. 2007)

osiin. Ratkaisun arvon määrittely taas motivoi tutkijaa etsimään ratkaisuja, toimien samalla johdantona tutkimuksen yleisölle, jotta he ymmärtävät tutkimuksen motiivit.

Toisena aktiviteettina on määrittellä ratkaisun tavoitteet. Tavoitteet tulee johtaa ongelman määrittelyn pohjalta ja konkreettisesta tutkimustilanteesta eli siitä, mitkä ovat käytettävissä olevat resurssit. Resursseihin kuuluu myös mahdolliset valmiit ratkaisut joita voidaan hyödyntää artefaktin kehittämisessä. Tavoitteet voivat olla laadullisia tai määrällisiä.

Kolmantena aktiviteettina on suunnitella ja kehittää artefakti. Tähän aktiviteettiin mennessä tutkijalla tulee olla riittävä teorian tieto hankittuna. Peffer ym. 2007 mukaan artefakti voi olla mikä tahansa suunniteltu objekti, jonka luomiseen on käytetty suunnitteluprosessia, ollen kuitenkin usein erilaisia konstruktioita, malleja, menetelmiä tai ilmentymiä.

Neljäntenä aktiviteettina on demonstroida artefaktin toimintaa. Artefakti vastaa silloin tutkimusongelman johonkin kohtaan, olematta kuitenkaan vielä täysin valmis.

Viidentenä aktiviteettina on artefaktin arviointi. Tämän kohdan tarkoituksena on tarkastella, kuinka hyvin se vastaa tutkimuksen alussa määriteltyyn ongelmaan. Arviointimenetelmä eroaa kuitenkin paljon riippuen artefaktin luonteesta. Arviointi voi olla esimerkiksi simulaatioita, asiakaspalautteita, määrällisiä suorituskykymittauksia tai vertailua toisessa toimenpiteessä määritellyihin tavoitteisiin. Saadun palautteen perusteella artefaktia voidaan jatkokehittää parem-

pien tulosten saamiseksi, tai jatkaa prosessimallin viimeiseen kohtaan eli viestintään.

Viimeinen aktiviteetti on viestintä. Tutkimuksesta usein tiedotetaan muille alan toimijoille, esimerkiksi julkaisemalla aiheesta tutkimusjulkaisu. Julkaisu voi pohjautua rakenteeltaan suunnittelutieteen prosessimalliin, lähtien ongelman havainnollistamisesta jatkuen artefaktin kehittämiseen, päättyen arviointiin ja johtopäätökseen. (Peffer ym. 2007).

3.2 Suunnittelutiede tutkimuksessa

Tässä luvussa esitellään tutkimusongelma, motivaatio tutkimuksen toteuttamiseen ja tutkimuksen tavoitteet. Tämän jälkeen toteutetaan artifakti, jonka avulla voidaan tarkastella tavoitteiden toteutumista. Lisäksi jotta päästään mittaamaan ja vertailemaan sivuston määrällisiä ominaisuuksia, luodaan yksinkertainen vertailukohde. Peffer ym. 2007 suunnittelutieteen prosessimallia ja siinä esiteltyjä aktiviteetteja.

3.2.1 Ongelman havainnollistaminen ja motivaatio

Tässä tutkimuksessa halutaan selvittää, kuinka hakukoneoptimoinnin tekninen toteutus tulisi tehdä SPA-verkkosivustossa. Vaikka hakukoneoptimointia on tutkittu tieteellisissä julkaisuissa paljon 2000-luvun alusta lähtien, tässä kontekstissa aihe ei juurikaan ole saanut huomiota, minkä takia aihetta on perusteltua tutkia.

3.2.2 Tavoitteet ja vaatimukset

Tässä alaluvussa käsitellään artefaktille asetettavat vaatimukset. Artefaktin tulee olemaan web-applikaatio, jota pystyy käyttämään verkkoselaimella. Verkkosivustolle pitää pystyä luomaan uusia sivuja sisällönhallintajärjestelmän kautta sekä lisäämään tehtyjä sivuja sivuvalikkoon. Tavoitteena on, että verkkosivusto täyttää toiminnoiltaan vähimmäisvaatimukset, jotta teknistä hakukoneoptimointia voidaan tehdä ja testata. Sen takia olen määrittänyt sivustolle tarkempia teknisiä vaatimuksia koskien sivuston rakennetta ja teknistä hakukoneoptimointia:

- Sivuston tulee olla ryömijöiden luettavissa ja indeksoitavissa:

- Sivuston täytyy omata robots.txt ja sitemap.xml -tiedostot.
- Linkit ovat hakukonerobottien luettavissa.
- Sivuston tulee noudattaa oikeaoppista HTML-dokumentin rakennetta:
 - Title-tagin
 - Heading-tagien oikeaoppinen käyttö
 - Kuvien alt-tekstit
- Sivuston ja yksittäisten sivujen metatiedot tulee olla päivitettävissä sisällönhallintajärjestelmästä.

3.2.3 Mittausmenetelmät

Verkkopalveluita voidaan testata lukuisilla eri tavoilla. Tässä tutkielmassa keskitymme hakukonenäkyvyyteen vaikuttaviin ominaisuuksiin, kuten sivuston rakenteeseen ja suorituskykyyn. Näiden ominaisuuksien mittaamiseksi on kehitetty useita eri sovelluksia, joista on valittu artefaktin testaukseen Googlen Lighthouse ... Tällaiset työkalut antavat tietoa verkkosivujen teknisestä toteutustavasta ja nopeudesta, mutta siltikin niiden antamiin tuloksiin kannattaa suhtautua suuntaa antavina. Kuitenkaan kaikki artefaktin vaatimukset eivät ole todettavissa testaustyökaluilla, vaan ne edellyttävät tekijän omia arvioita ja havaintoja. Tehtyä artefaktia arvioidaan suhteessa asetettuihin vaatimuksiin.

TODO: Testaustyökalujen valintaan vaihtoehtoja: netpeaksoftware.com/spider <https://www.seoptimizer.com/>

3.3 Artefaktin tekninen toteutus

Tutkimuksessa toteuttava artefakti tulee olemaan yksinkertainen verkkosivusto. Verkkosivuston käyttäjälle näkyvä osa eli käyttöliittymälogiikka tullaan toteuttamaan Nuxt.js -kirjastolla. Nuxt.js on Vue.js -kirjastoon pohjautuva kehys, jonka avulla voidaan toteuttaa dynaamisia web-sovelluksia ja verkkosivustoja. ~~Verrattuna perinteisiin verkkosivustoihin jotka koostuvat useista eri sivuista, luo dynaaminen verkkosivu hakukoneoptimoinnin toteuttamiselle omat haasteensa.~~ Nuxtia käytettäessä on mahdollista valita, suoritetaanko verkkosivun renderöinti palvelimella vai käyttäjän selaimessa. Tässä tutkimuksessa verkkosivuston renderöinti suoritetaan palvelimella.

Nykyaikainen verkkosivusto tulee olla helposti myös käyttäjien päivitettävissä. Sisällönhallintajärjestelmänä käytetään avoimeen lähdekoodiin pohjautuvaa Wordpressia, joka on kirjoitettu PHP-ohjelmointikielellä. Artefaktin rakentamisessa voidaan hyödyntää muitakin valmiita ratkaisuja, kuten Wordpressin valmiita lisäosia perustelluista syistä. Valittu sisällönhallintajärjestelmä vaikuttaa siihen, että millä tavalla tekniset ratkaisut käyttöliittymän yksityiskohtaisesti toteutetaan. Sisällönhallintajärjestelmästä riippumatta hakukoneoptimointiin vaikuttavat toimenpiteet ovat kuitenkin pääosin samat. Wordpress asettaa muutamia teknisiä vaatimuksia palvelinympäristölle. Wordpress suosittelee palvelimelta täyttävän seuraavat vaatimukset: PHP jonka versio on 7.4 tai uudempi, MySQL versio 5.7 tai MariaDB versio 10.2 tai uudempi.

Jotta sisällönhallintajärjestelmän kautta syötettyä dataa voidaan näyttää käyttöliittymässä, tulee tiedot hakea rajapinnan kautta. Wordpressissä on sisäänrakennettuna REST API -rajapinta, joka tarjoaa haetun datan JSON-muodossa. Vaikka REST API -rajapinnan käyttö olisi ollut ilmeinen valinta sen sisäänrakennettavuuden takia, päädyin käyttämään sen sijasta GraphQL-rajapintaa. Sen sijasta että käytettäisiin RESTIN päätepisteitä (eng. endpoint), GraphQL-kyselyt mahdollistavat tavan hakea tietoja yksityiskohtaisemmin. Wordpress ei oletuksena tarjoa kyseistä rajapintaa, vaan tuen tarjoaa ladattava lisäosa. Tässä työssä käytin WPGraphQL -nimistä lisäosaa, joka on ilmainen, avoimen lähdekoodin lisäosa. Kun kehitetään Wordpress-sivustoja, on yleensä järkevää tietyissä tapauksissa käyttää valmiita lisäosia sen sijasta, että tarvittavaa toiminnallisuutta lähtisi itse toteuttamaan ja kehittämään alusta asti. Muita olennaisia Wordpress-kehityksessä tarvittavia lisäosia on Advanced Custom Fields, joka mahdollistaa kustomoitavien lisäkenttien luomisen sisällönhallintajärjestelmään, jotta sisältöjä voidaan syöttää monipuolisemmin kuin Wordpressin omien kenttien avulla. Advanced Custom Fieldsiin taas on saatavilla GraphQL -lisäosa, jonka avulla kyselyillä voidaan hakea myös ACF-kenttiin syötettyjä tietoja.

~~Käytetyt teknologiat, kirjastot ja lisäosat listattuna: Sisällönhallintajärjestelmänä Wordpress MAMP-palvelinympäristö lokaaliin kehitykseen Wordpressin lisäosina WPGraphQL,~~

<u>nimi</u>
<u>Wordpress</u>
<u>Nuxt.js</u>
<u>WPGraphQL</u>
Advanced Custom Fields , WPGraphQL for Advanced Custom Fields , Yoast SEO Web-applikaatiokirjasto
<u>WPGraphQL for ACF</u>
<u>Yoast SEO</u>

Tämä kappale käsittelee verkkosivujen renderöintiä. Renderöinnillä tarkoitetaan tapaa, kuinka lähdekoodi käännetään toimivaksi, selaimessa käytettäväksi kokonaisuudeksi. Yksinkertaisimmillaan verkkosivusto rakentuu HTML-dokumentista, CSS-tyylitiedostosta ja mahdollisesta JavaScript-tiedostosta, jotka selain osaa kääntää verkkosivustoksi. Tällaisella yksinkertaisella verkkosivustolla ryömijä osaa helposti käydä HTML-dokumentin läpi ja jatkaa eteenpäin sieltä löytyvistä linkeistä. SPA-verkkosivustoissa taas vaihtoehtoja suorittaa renderöinti on useampia, ja tällä valinnalla on suora vaikutus siihen miten hakukoneoptimointi voidaan toteuttaa sivustolla. Useat yleisesti web-kirjastot kuten React ja Vue renderöivät näytettävän sisällön käyttäjän selaimella. Tämä tarkoittaa sitä, että palvelimella on vain nähtävissä lähes tyhjä HTML-dokumentti, jossa on osoitettuna tietyllä tunnisteella paikka johon dynaaminen sisältö luodaan. Tämä aiheuttaa ongelmia hakukoneiden ryömijöiden läpikäydessä sivustoa, sillä palvelimella ei ole nähtävissä mitään käyttäjän sisällönhallinnassa syöttämiä tietoja, joita indeksoida. SPA-verkkosivustojen tapauksessa tämä voidaan ratkaista kahdella eri tavalla. Jos käytetty kirjasto sisältää mahdollisuuden suorittaa renderöinti palvelimella (eng. server side rendering), sitä kannattaa hyödyntää hakukoneoptimoinnin helpottamiseksi. Tässä tutkimuksessa hyödynnän Nuxt.js-kirjaston SSR-ominaisuutta. On tosin otettava huomioon, että tämä luo erityisvaatimuksia palvelimelle, sillä JavaScriptin suorittaminen palvelimella vaatii node.js ympäristön.

3.3.1 [Sitemap ja robots -tiedostojen luominen](#)

Toinen vaihtoehto on generoida SPA-sivusta täysin staattinen HTML-verkkosivu. Tämä voi kuulostaa paluulta menneeseen aikaan, mutta tämän vaihtoehdon suosiolle on olemassa syytäkin. Staattinen HTML-verkkosivu on tietoturvallinen ja nopea, mutta silti sisältöjä pystyy hallitsemaan modernin sisällönhallintajärjestelmän kautta.

3.3.2 Sitemap ja Robots-tiedostojen luominen

Verkkosivuston hakukoneoptimointia varten on olemassa kaksi eri hyödyllistä tiedostoa. Robots.txt tiedoston avulla voidaan esimerkiksi kieltää ryömijöitä indeksoimasta tiettyjä sivuja, kuten esimerkiksi ylläpitäjille tarkoitettu kirjautumissivu. Sitemap.xml taas on tiedosto, joka sisältää listan verkkopalvelun eri sivujen osoitteista.

Nuxtilla näiden tiedostojen luominen onnistuu sen omien moduulien avustuksella. Ensimmäisenä asennetaan paketit komentorivityökalulla NPM-pakettienhallintaa käyttäen:

```
$ npm install @nuxtjs/sitemap @nuxtjs/robots
```

Kun paketit on asennettu onnistuneesti, ne otetaan käyttöön `nuxt.config.js` tiedostossa seuraavanlaisesti:

```
modules: [  
  '@nuxtjs/sitemap',  
  '@nuxtjs/robots',  
],
```

Nuxtin sitemap -moduuli osaa ottaa automaattisesti huomioon vain staattiset sivut, jotka on luotu sivuston pages -kansioon. Koska sovelluksessa kaikki sivut luodaan dynaamisesti sisällönhallintajärjestelmässä, joutuu sivukartan eteen tekemään manuaalista työtä. Jos sivuja on todella paljon tai ne muuttuvat usein, uskoisin olevan mahdollista tehdä erillinen funktio, joka osaa tehdä sivukartan automaattisesti ottaen huomioon myös dynaamiset sivut. Sivustolla tulee kuitenkin etusivun lisäksi olemaan vain kaksi eri sivua, joten ne on helppo lisätä manuaalisesti:

```
sitemap: {  
  routes: [  
    '/yhteydenotto',  
    '/referenssit',  
  ]  
},
```

Molemmat moduulit sisältävät lukuisan määrän eri asetuksia, jotka on nähtävillä niiden dokumentaatioissa. Esimerkiksi seuraavalla asetuksella voidaan kieltää Googlebottia indeksoimasta kaikkia sivuja tai tiedostoja, joiden osoitepolku alkaa `http://esimerkki.fi/users/`.

```
robots: [  
  {  
    UserAgent: 'Googlebot',  
    Disallow: () => '/users/'  
  }  
]
```

Kun halutut asetukset on asetettu, sovellus generoi automaattisesti molemmat tiedostot verkkosivun juureen.

3.3.2 Metatietojen tuominen sisällönhallintajärjestelmästä

Metatiedoilla tarkoitetaan tietoja, jotka eivät näy suoraan verkkosivulla, vaan hakukone sekä selain hyödyntää niitä. Esimerkiksi metatietojen otsikkoa käytetään selaimen välilehdissä informoidakseen käyttäjälle, mikä sivu on kyseessä. Wordpressin tapauksessa sivun metatiedot voisi luoda lisäkenttien avulla. ~~Päätin~~ Tässä tapauksessa päätettiin kuitenkin ~~tässä tapauksessa käyttää~~ hyödyntää Yoast SEO -lisäosaa, joka on suosituimpia hakukoneoptimointilajennuksia Wordpressiin, sekä yleisesti käytössä kun luodaan verkkopalveluita yrityksille. Yoast tukee ~~GraphQL-rajapintaa~~ GraphQL -rajapintaa, joka mahdollistaa metatietojen haun sovellukseen. Ensimmäisenä luodaan kysely:

```
const METADATA_QUERY = gql `  
query MyQuery {  
  page(id: "2", idType: DATABASE_ID) {  
    seo {  
      metatitle: title ,  
      metadesc: metaDesc ,  
    }  
  }  
}
```



```

    }
  }
  ‘;

```

Kyselyssä haetaan tunnisteella kaksi olevan sivun metaotsikko ja -kuvaus. Oletuksena sivuston metatiedot sijaitsevat `nuxt.config.js` -tiedostossa. Metatiedot voidaan asettaa `head`-objektin avulla jokaiselle verkkosivuston sivulle erikseen:

```

head() {
  return {
    title: this.page.seo.metatitle,
    meta: [
      {
        content: this.page.seo.metadesc,
      }
    ]
  }
}

```

Lista tehtävistä: Piirrä kuva prosessimallista Kuva-prosessista? Artefaktin vaatimukset Teoriaosuuden tarkastelu; synteesiä, lähteitä Palvelin-

3.3.3 Alkuperäisen sisällön osoittaminen

Verkkosivustoissa samaan sivuun voi osoittaa usea eri osoite. `Rel=canonical` -elementin avulla voidaan osoittaa alkuperäinen sivu, joka myös halutaan indeksoitavaksi hakukoneryömiäjöiden toimesta. Canonical-url tulee asettaa sivuston otsakkeeseen. Nuxtilla tämä voidaan tehdä `layouts/default.vue` -tiedostossa asettamalla linkki seuraavasti:

```

export default {
  head() {
    return {
      link: [
        {

```



```

~~~~~
<h1 v-if="error.statusCode === 404">Hups! Sivua ei valitettavasti
~~~~~

<h1 v-else>Virhe tapahtui: - {{ error.statusCode }} </h1>
~~~~~
<NuxtLink to="/">Palaa etusivulle </NuxtLink>
~~~~~
</div>
~~~~~
</template>
~~~~~
%DIF >
~~~~~
<script>
~~~~~
  export default {
~~~~~
    props: ['error'],
~~~~~
    layout: 'error'
~~~~~
  }
~~~~~
</script>
~~~~~

```

Virhekoodin ollessa 404 kerromme käyttäjälle että sivua ei löytynyt. Muun virheen sattuessa ilmoitamme virheestä ja näytämme HTTP-statuskoodin. Molemmissa tapauksissa käyttäjälle näytetään linkki, josta pääsee takaisin etusivulle.

3.4 Sivuston sisällöt

Tässä kappaleessa käsitellään verkkosivuston sisältöjä ja niiden syöttämistä, sekä tarkastellaan, tuoko SPA-menetelmällä luotu sivusto eroavaisuuksia prosessiin.

3.4.1 Sisältöjen syöttäminen

Wordpress-sisällönhallintajärjestelmää käytettäessä sisältöjen syöttäminen on yksinkertaista. Wordpressin sisältöeditori on nimeltään Gutenberg, joka mahdollistaa eri tyyppisten sisältöjen syöttämisen erilaisten lohkojen (eng. blocks) avulla. Sisältöeditorin käyttö on täysin samanlaista riippumatta siitä, onko kyseessä normaali Wordpress-sivusto vai rajapinnan kautta toteutettu SPA-sivusto. Sisältöeditori toimii niin kutsutulla WYSIWYG-periaatteella, eli oletuksena rakennetta tai muotoilukodeja ei näytetä muokkaajalle lainkaan, vaan sisältö on muokatessa

ulkonäötään lähellä lopputulosta. Kuitenkin kaikki muokkaajalle näkymätön metadata siirtyy GraphQL-rajapinnan kautta käyttöliittymälle, jonka ansiosta sivusta muodostuu rakenteellisesti oikeanlainen.

Yleisesti ottaen HTML-merkintämielessä on suositeltavaa käyttää vain yhtä H1 eli pääotsikkoa tarkoittavaa merkintää (Mozilla 2022). Artefaktin tapauksessa haluamme sen olevan aina ensimmäinen eli ylimmäinen elementti sisältöalueella. Lisäksi pääotsikon halutaan olevan sama kuin sisällönhallintajärjestelmässä syötetty sivun pakollinen otsikko. Koska sivun otsikko on rajapinnan kautta tullessa vain pelkkä teksti vailla mitään muotoilua, asetamme sen H1-tagien väliin. Muuttuja `post.content` taas sisältää kaiken datan mitä Gutenbergin WYSIWYG-editorissa on syötetty, mukaanlukien rakenteen rivinvaihtoineen ja muotoiluineen:

```
"content": "\n<p>Button blocks are not semantically <em>buttons</em>,"
```

Käyttöliittymän puolella sivun otsikko ja sisältö voidaan tulostaa `<article>` -elementtien sisään seuraavalla tavalla:

```
<h1 v-if="post">{{ post.title }}</h1>  
<article v-if="post" class="py-5" v-html="post.content"></article>
```

Wordpress-yhteisö tarjoaa osana teemantestausprosessia testidataa (Wordpress 2020). Testidata on saatavilla XML-muotoisessa tiedostossa, jonka pystyy tuomaan sisällönhallintajärjestelmään sen sisäänrakennetun työkalun avulla. Testidata sisältää yhteensä 21 sivua ja 51 artikkelia. Wordpress sisältää oletuksena paljon erilaisia ominaisuuksia, kuten mahdollisuuden kommentoida artikkelia tai käyttää erilaisia sivupohjia eri sivuilla. Data sisältää testitapauksia myös ominaisuuksiin, joita ei tässä tutkimuksessa käyttöliittymään toteuteta. Tämän takia testidataan tehdään karsintaa ja artefaktiin valitaan tähän tapaukseen sopivat sivut ja artikkelit.

4 Tulokset/Analyysi

5 Yhteenveto

Tutkielman viimeinen luku on Yhteenveto.

Lähteet

- Bhandari, Ravneet Singh, ja Ajay Bansal. 2018. "Impact of search engine optimization as a marketing tool". *Jindal Journal of Business Research* 7 (1): 23–36.
- Davis, Harold. 2006. *Search engine optimization*. "O'Reilly Media, Inc."
- Egri, Gokhan, ja Coskun Bayrak. 2014. "The role of search engine optimization on keeping the user on the site". *Procedia Computer Science* 36:335–342.
- Enache, Maria Cristina, ym. 2014. "Optimization Methods and Seo Tools". Teoksessa *International Conference «Risk in Contemporary Economy», Dunarea de Jos University of Galati, Galati*, 98–103.
- Fielding, R, Jim Gettys ym. 1998. "Hypertext Transfer Protocol-HTTP/1.1".
- Goodman, Danny. 2002. *Dynamic HTML: The definitive reference: A comprehensive resource for HTML, CSS, DOM & JavaScript*. "O'Reilly Media, Inc."
- Google. 2021a. *Consolidate duplicate URLs*. <https://developers.google.com/search/docs/advanced/crawling/consolidate-duplicate-urls>. [Viitattu: 2022-01-8].
- . 2021b. *Learn about sitemaps*. <https://developers.google.com/search/docs/advanced/sitemaps/overview>. Viitattu: 2021-01-8.
- . 2022. *Soft 404 errors*. <https://developers.google.com/search/docs/advanced/crawling/soft-404-errors>. Viitattu: 2022-02-3.
- Gupta, Swati, Nitin Rakesh, Abha Thakral ja Dev Kumar Chaudhary. 2016. "Search engine optimization: Success factors". Teoksessa *2016 Fourth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, 17–21. doi:10.1109/PDGC.2016.7913146.
- Hammer, Hugo Lewi, Alfred Bratterud ja Siri Fagernes. 2013. "Crawling JavaScript websites using WebKit-with application to analysis of hate speech in online discussions".

- Hui, Zhou, Qin Shigang, Liu Jinhua ja Chen Jianli. 2012. “Study on Website Search Engine Optimization”. Teoksessa *2012 International Conference on Computer Science and Service System*, 930–933. doi:10.1109/CSSS.2012.236.
- Iskandar, Taufan Fadhilah, Muharman Lubis, Tien Fabrianti Kusumasari ja Arif Ridho Lubis. 2020. “Comparison between client-side and server-side rendering in the web development”. Teoksessa *IOP Conference Series: Materials Science and Engineering*, 801:012136. 1. IOP Publishing.
- Krrabaj, Samedin, Fesal Baxhaku ja Dukagjin Sadrijaj. 2017. “Investigating search engine optimization techniques for effective ranking: A case study of an educational site”. Teoksessa *2017 6th Mediterranean Conference on Embedded Computing (MECO)*, 1–4. doi:10.1109/MECO.2017.7977137.
- Kumar, Arunjay. 2013. “Search engine optimization (SEO): technical analysis concepts”. *International Journal of Emerging Technology and Advanced Engineering* 3 (3): 123–128. <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.413.4659&rep=rep1&type=pdf>.
- Lewandowski, Dirk, Friederike Kerkmann, Sandra Rümmele ja Sebastian Sünkler. 2018. “An empirical investigation on search engine ad disclosure”. *Journal of the Association for Information Science and Technology* 69 (3): 420–437.
- Levene, Mark. 2011. *An introduction to search engines and web navigation*. John Wiley & Sons.
- Malaga, Ross A. 2010. “Search engine optimization—black and white hat approaches”. Teoksessa *Advances in Computers*, 78:1–39. Elsevier.
- Meneses, Luis, Richard Furuta ja Frank Shipman. 2012. “Identifying “Soft 404” error pages: analyzing the lexical signatures of documents in distributed collections”. Teoksessa *International Conference on Theory and Practice of Digital Libraries*, 197–208. Springer.
- Mikowski, Michael, ja Josh Powell. 2013. *Single page web applications: JavaScript end-to-end*. Simon / Schuster.

- Mozilla. 2022. *<h1>–<h6>: The HTML Section Heading elements*. https://developer.mozilla.org/en-US/docs/Web/HTML/Element/Heading_Elements. Viitattu: 2022-02-5.
- Ohye, Maile, ja Joachim Kupke. 2012. *The Canonical Link Relation*. RFC 6596, huhtikuu. doi:10.17487/RFC6596. <https://www.rfc-editor.org/info/rfc6596>.
- Olston, Christopher, ja Marc Najork. 2010. *Web crawling*. Now Publishers Inc.
- Osmani, Addy, ja Ilya Grigorik. 2019. *Using page speed in mobile search ranking*. [Verkko-sivu; Luettu 8.11.2021]. <https://developers.google.com/web/updates/2018/07/search-ads-speed>.
- Patil Swati, P, BV Pawar ja S Patil Ajay. 2013. “Search engine optimization: A study”. *Research Journal of Computer and Information Technology Sciences* 1 (1): 10–13. <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1070.1729&rep=rep1&type=pdf>.
- Peppers, Ken, Tuure Tuunanen, Marcus A. Rothenberger ja Samir Chatterjee. 2007. “A Design Science Research Methodology for Information Systems Research”. *Journal of Management Information Systems* 24 (3): 45–77. doi:10.2753/MIS0742-1222240302. eprint: <https://doi.org/10.2753/MIS0742-1222240302>. <https://doi.org/10.2753/MIS0742-1222240302>.
- Prieto, Víctor M., Manuel Álvarez ja Fidel Cacheda. 2013. “Analysis and detection of Soft-404 pages”. Teoksessa *Third International Conference on Innovative Computing Technology (INTECH 2013)*, 217–226. doi:10.1109/INTECH.2013.6653695.
- Prieto, Victor M, Manuel Álvarez ja Fidel Cacheda. 2014. “Soft-404 Pages, A Crawling Problem.” *J. Digit. Inf. Manag.* 12 (2): 73–92.
- Scott Jr, Emmit A. 2015. *SPA Design and Architecture: Understanding single-page web applications*. Simon / Schuster.
- Seymour, Tom, Dean Frantsvog, Satheesh Kumar ym. 2011. “History of search engines”. *International Journal of Management & Information Systems (IJMIS)* 15 (4): 47–58.

Sharma, Dushyant, Rishabh Shukla, Anil Kumar Giri ja Sumit Kumar. 2019. “A brief review on search engine optimization”. Teoksessa *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, 687–692. IEEE.

Slatin, John M. 2001. “The art of ALT: toward a more accessible Web”. *Computers and Composition* 18 (1): 73–81. ISSN: 8755-4615. doi:[https://doi.org/10.1016/S8755-4615\(00\)00049-9](https://doi.org/10.1016/S8755-4615(00)00049-9). <https://www.sciencedirect.com/science/article/pii/S8755461500000499>.

Statcounter. 2021. *Desktop Search Engine Market Share Worldwide*. <https://gs.statcounter.com/search-engine-market-share/desktop/worldwide>. Viitattu: 2020-11-30.

Wordpress. 2020. *Theme Unit Test*. https://codex.wordpress.org/Theme_Unit_Test. Viitattu: 2022-02-10.

Vuejs. 2021. *Vue.js Server-Side Rendering Guide*. <https://ssr.vuejs.org>. Viitattu: 2022-01-27.

Liitteet

A Siirtyminen gradu2:sta gradu3:een

Keskeneräisen tutkielman siirtäminen gradu2:sta gradu3:een ei ole kovin vaikeata. Aluksi on totta kai vaihdettava \documentclass-komennossa gradu2 gradu3:ksi. Komennon optioista suurin osa on poistettava, koska niitä ei enää tueta; ainoastaan merkistön ilmoittava optio jää jäljelle. Mahdollinen kandi-optio vaihdetaan optioksi bachelor.

Taulukossa 1 on lueteltu tarvittavat komentovaihdokset. Viiva tarkoittaa, ettei vastaavaa komentoa ole lainkaan. Huomaa erityisesti uudet komennot.

gradu2	gradu3
—	\maketitle
—	\supervisor
\acmccs	—
\aine	\subject
\copyrightowner	—
\fulltitle	—
\laitos	\department
\license	—
\linja	\studyline
\paikka	—
\setauthor	\author
\termlist	thetermlist-ympäristö
\tyyppi	\type
\yhteystiedot	\contactinformation
\yliopisto	\university
\ysa	—

Taulukko 1. Komentomuutokset gradu2:sta gradu3:een

Isoin työ voi aiheutua lähdeluettelon laatimistekniikan muuttumiseen sopeutumisesta.

B Harvemmin tarvittavat ominaisuudet

Aiemmin esiteltyjen lisäksi gradu3 tarjoaa seuraavat lisäominaisuudet:

- `\LaTeX 2 ϵ` :n vakio-optiot `draft` ja `final` toimivat.
- `\LaTeX 2 ϵ` :n vakio-optio `twoside` toimii myös. Tätä voi käyttää esimerkiksi gradun kansitusversion laatimiseen, mutta virallisen arvostelu- ja arkistokappaleen laatimiseen sitä ei suositella.
- Vaikka tutkielman suomenkielisyyttä ei tarvitse erikseen mainita, `finnish`-optio toimii.
- `\university`-komennolla voit ilmoittaa tutkielman kotiyliopistoksi jonkin muun kuin Jyväskylän yliopiston.
- `\department`-komennolla voit ilmoittaa tutkielman kotilaitokseksi jonkin muun kuin Informaatioteknologian tiedekunnan.
- `\subject`-komennolla voit ilmoittaa tutkielman oppiaineeksi jonkin muun kuin tietotekniikan. Huomaa, että oppiaine tulisi suomenkielisissä tutkielmissa kirjoittaa genetiivimuodossa ja isolla alkukirjaimella ("`Tietotekniikan`"), englanninkielisissä tutkielmissa in-preposition kanssa ("`in Information Technology`").
- `\type`-komennolla voit ilmoittaa tutkielman tyypin, jos se on jokin muu kuin `pro gradu` (oletus) tai kandidaatintutkielma (optiolla `bachelor`).
- `\setdate`-komennolla voit asettaa päivämäärän haluamaksesi. Anna komennolle kolme parametria – päivä, kuukausi ja vuosi – numeerisessa muodossa.
- Ympäristöllä `chapterquote` voit laittaa luvun alkuun mietelauseen. Sillä on yksi pakollinen parametri (lainauksen attribuutio).
- Komento `\graducsddate` sisältää käytössä olevan gradu3:n julkaisupäivämäärän ja `\graducsdversion` sen versionumeron.