Saavutettavuusauditoinnin kehittäminen

Aleksi Taivalantti

2020 Laurea

Laurea-ammattikorkeakoulu

Saavutettavuusauditoinnin kehittäminen

Aleksi Taivalantti

Tietojenkäsittely

Opinnäytetyö

Marraskuu 2020

Laurea-ammattikorkeakoulu **Tiivistelmä**
Tietojenkäsittely
It-tradenomi(AMK)

Aleksi Taivalantti
Saavutettavuusauditoinnin kehittäminen
Vuosi 2020 Sivumäärä 39

Tämän opinnäytetyön tavoitteena oli kehittää yksinkertainen ja tarkka prosessi verkkopalvelujen saavutettavuusauditointeja varten. Saavutettavuusauditoinnilla tarkistetaan, mikäli verkkopalvelu on Suomessa voimaan astuneen digitaalisten palvelujen tarjoamisen lain mukainen. Prosessin tarkoituksena oli myös yhdenmukaistaa toimeksiantajayrityksen sisällä tehtävät auditoinnit, minimoida auditoinneissa sattuvien inhimillisten virheiden määrä sekä toimia koulutus ja perehdytys materiaalina. Työn toimeksiantajana toimi suunnittelu- ja ohjelmistotoimisto Agenda Helsinki.

Saavutettavuusauditointiprosessi kehitettiin prosessin kehittämisen menetelmällä. Prosessin kehittämistä varten tutustuttiin verkkopalvelujen tekniseen toteutukseen, sekä dokumentoitiin asiakasprojektina tehty saavutettavuusauditointi jarjestohautomo.fi-sivustolle. Saavutettavuusauditoinnin tulokset validoitiin asiantuntijalla, jonka jälkeen auditoinnista syntynyt dokumentaatio analysoitiin. Tästä analyysistä muodostui prosessin kuvaus ja prosessikaavio, joiden mukaisesti saavutettavuusauditointiprosessi kehitettiin.

Pääasiallisena tietoperustana käytettiin World Wide Web Konsortion luomaa WCAG 2.1 –saavutettavuusohjeistusta, johon Euroopan Unionin laatima saavutettavuusdirektiivi pohjautuu. Viitekehys keskittyi erityisesti verkkopalvelujen saavutettavuuden tekniseen puoleen verkkokehittäjän näkökulmasta.

Opinnäytetyön tuloksena syntyi helposti seurattava prosessi, jonka avulla voidaan tarkistaa, ovatko verkkopalvelut saavutettavia lain puitteissa.

Asiasanat: saavutettavuus, auditointi, verkkopalvelu, wcag2.1, prosessi

Laurea University of Applied Sciences
Business and Information Technology
Bachelor's Degree

Aleksi Taivalantti

Developing a Process for Accessibility Audits

Year 2020 Pages 39

Abstract

The goal of this Bachelor's thesis was to develop a simple and an accurate process for accessibility audits on websites. An accessibility audit is used to review if websites are adequate by the standards of the Finnish law of Act on the Provision of Digital Services. The idea of the process is also to standardize the audits inside the client company of this thesis, minimize the human errors and to be used as educational and induction material. The client company is digital design agency Agenda Helsinki.

The process for accessibility audits was developed with a method of process development. For the development process the technical aspects of websites were studied thoroughly. After the required knowledge was collected an accessibility audit was performed and documented as a customer project on a website jarjestohautomo.fi. The results of the accessibility audit were reviewed by an UX-expert followed by a deep analysis of the documentation. This analysis was then used to develop a description of the process and a process diagram on which basis the final accessibility auditing process was created.

The main source of information was the accessibility guidelines WCAG 2.1 created by the Word Wide Web Consortium. The Finnish law of Act on the Provision of digital services enforces European Accessibility Directive which is directly based on WCAG 2.1-guidelines. The frame of reference concentrated especially on the technical aspects of websites from a web developers point of view.

The result of this thesis was an easy-to-follow process which can be used to perform an audit on any website to review if they meet the accessibility requirements enforced by the Finnish law.

Keywords: accessibility, auditing, websites, wcag2.1, process

Sisällys

1 Johdanto

Vuonna 2019 Suomessa voimaan astunut laki digitaalisten palveluiden tarjoamisesta velvoitti julkisen sektorin toimijat varmistamaan, että heidän tarjoamat digitaaliset palvelut ovat saavutettavissa kaikille. Laki asetti saavutettavuuden toteutumiselle aikarajan, jonka mukaan uusien, 23.9.2018 jälkeen julkaistujen verkkopalveluiden, tulee täyttää lain vaatimukset 28.9.2018 mennessä. Tätä vanhempien verkkopalveluiden siirtymäaika oli pidempi ja niiden tuli olla saavutettavia 23.9.2020 mennessä. Jälkimmäisen aikarajan vuoksi julkisen sektorin toimijat ovat lähteneet tänä syksynä sankoin joukoin liikkeelle korjaamaan verkkopalveluidensa saavutettavuutta. (306/2019; Selovuo 2019, 17-19).

Voimaan astunut laki on pakottanut myös verkkokehittäjiä ja -suunnittelijoita tutustumaan verkkosaavutettavuuteen sekä muokkaamaan omia työtapoja saavutettavuuden näkökulmasta. Useat ovat ottaneet saavutettavuusauditointien tekemisen omaan repertuaariinsa.

Tämän opinnäytetyön tavoitteena on kehittää helposti seurattava saavutettavuusauditointiprosessi, jota seuraamalla verkkokehittäjät ja –suunnittelijat voivat tehdä saavutettavuusauditointeja asiakasprojekteissa. Prosessin tarkoituksena on yhdenmukaistaa työnantajani, Agenda Helsingin, tekemät saavutettavuusauditoinnit sekä vähentää inhimillisten virheiden määrää ja tehdä auditoinneista samaan aikaan kattavia ja tehokkaampia.

Prosessi luodaan tutustumalla verkkosaavutettavuuden tekniseen puoleen, dokumentoimalla asiakasprojektina tehty saavutettavuusauditointi, tarkistuttamalla saavutettavuusauditoinnin tulokset asiantuntijalla ja analysoimalla tehdyn auditoinnin työnkulkua jonka pohjalta laaditaan oman työpaikkani sisällä käytettävä saavutettavuusauditointiprosessi. Kehittämismenetelmänä käytetään prosessien kehittämisen menetelmää.

Kehitystyön tietoperusta keskittyy verkkopalvelujen tekniseen puoleen, ja sen vaikutukseen digitaalisten palvelujen saavutettavuudessa. Koko opinnäytetyö on kirjoitettu kehittäjän näkökulmasta.

2 Työn lähtökohdat

Agenda Helsinki Oy on noin kymmenen työntekijän suunnittelu- ja ohjelmistoyhtiö. Yrityksen ydinosaamista on digitaalisten verkkopalveluiden ja sovellusten kehittäminen, sekä käytettävyys- ja saavutettavuustestaukset. (Agenda Helsinki 2020).

Suoritin työharjoitteluni Agenda Helsingillä 2019-2020, jonka jälkeen yritys palkkasi minut töihin verkkokehittäjäksi.

Tämän vuoden kesän ja syksyn aikana saavutettavuusauditointipyyntöjen määrä nousi huomattavasti. Jotta pystyimme vastaamaan kysynnän määrään, ja ylläpitämään työn laatua, oli tarpeen luoda helposti seurattava prosessi, jonka avulla voi tehdä laadukkaita verkkopalveluiden saavutettavuustestauksia ja minimoida inhimillisten virheiden määrä.

Olin jo aiemmin tehnyt saavutettavuusauditointeja, joten nyt oli tärkeää dokumentoida ja analysoida oman tutkimuksen ja kokemuksen kautta muovautuneen auditointiprosessin eri vaiheet, ja luoda tämän analyysin perusteella yksinkertainen, mutta kattava lista auditoinnin eri vaiheista.

2.1 Kehittämistavoite

Opinnäytetyön kehittämistavoitteena oli kehittää Agenda Helsingille saavutettavuusauditointiprosessi. Prosessin tuli olla johdonmukainen, helposti seurattava, laadukas ja tarkka. Prosessia seuraamalla työntekijät voivat suorittaa saavutettavuusauditointeja mihin tahansa verkkopalveluihin.

2.2 Aihealueen rajaus

Koska opinnäytetyö on kirjoitettu verkkokehittäjän näkökulmasta, aihealueesta rajataan pois sisällöntuotanto, joka on yksi verkkosaavutettavuuden tärkeistä osa-alueista. Sisällöntuotanto rajataan pois myös siksi, että saavutettavuusauditoinneissa harvoin pyydetään ottamaan kantaa itse sisältöön, ja uskon vahvasti, että sisällön arvioimisen pitäisi tehdä siihen erikoistunut asiantuntija.

Auditoinnissa keskitytään kokonaan ns. pöytäkonenäkymään, eikä opinnäytetyöhön dokumentoida ollenkaan asiakasprojektissa tehtyä mobiiliauditointia. Myös verkkosivuilta löytyvien liitetiedostojen auditointi rajataan pois.

2.3 Keskeiset käsitteet

Auditointi

Järjestelmällinen ja dokumentoitu prosessi, jolla arvioidaan annettua auditointinäyttöä mahdollisimman objektiivisesti. Tarkoituksena on määrittää, onko sovitut auditointikriteerit täytetty. (Itä-Suomen yliopisto 2020).

Html

Verkkodokumenttien eli verkkosivujen merkkauskieli. Html on lyhenne sanoista Hyper Text Markup Language. Html koostuu semanttisista, ja ei-semanttisista elementeistä, joita käytetään verkkosivujen rakennuspalikoina. (W3schools 2020).

Käyttäjä

Tässä opinnäytetyössä käyttäjällä viitataan lähtökohtaisesti henkilöön, joka käyttää verkkopalveluita.

Prosessi

Toimenpiteiden sarja, jolla pyritään ennalta määritettyyn lopputulokseen. (Jyväskylän yliopisto 2020).

Saavutettavuus

Saavutettavan palvelun tai tuotteen esittämistä siten, että kaikki voivat käyttää sitä. Haittaavat ominaisuudet eivät saa estää sisällön havainnoimista tai lukemista eikä palvelun käyttämistä. (Selovuo 2019, 13).

Sarkainjärjestys

Järjestys, missä kohdennus osuu eri html-elementteihin verkkosivulla näppäimistön sarkainpainiketta (tab) painamalla. (Korpela 2011, 167).

Semantiikka

Sanan oikeaa tulkintaa ja sen oikeaoppista käyttöä lauseessa. Tässä opinnäytetyössä semantiikasta puhutaan Html-merkkauskielen yhteydessä, jolloin se tarkoittaa html-elementin tarkoituksen oikea tulkinta, ja sen oikeaoppinen käyttö lähdekoodissa. (Penland 2020).

3 Saavutettavuus

Saavutettavuus on digitaalisen informaation esittämistä kaikille, riippumatta henkilöiden toiminnallisista haasteista. Verkkosaavutettavuus takaa kaikille yhtäläiset oikeudet palveluihin, sisältöihin ja verkossa tapahtuvaan toimintaan. (Selovuo 2019, 13).

Saavutettavuus koskee jollain tapaa noin 20% väestöstä. Tähän kuuluu ihmiset, joilla on haasteita kuulo-, tai näköaistien kanssa, motorisia ongelmia tai kognitiivisia vaikeuksia. Näistä suurin ryhmä on kognitiivisia vaikeuksia kokevat henkilöt, joihin lasketaan myös muistisairaat. (Selovuo 2019, 15).

Saavutettavia palveluja tarvitsee siis valtava määrä ihmisiä, kun miettii että Internetin käyttö on 13-kertaistunut viimeisen kahdenkymmenen vuoden aikana, ja tällä hetkellä Internetiä käyttää noin 4,8 miljardia ihmistä (Internet World Stats 2020). Pelkästään EU:ssa oli vuonna 2016 noin 80 miljoonaa asukasta, jotka tarvitsivat saavutettavia palveluita. Ikärakenteen vuoksi tämä määrä tulee kasvamaan lähivuosina radikaalisti (Euroopan parlamentti 2016).

Internetin suosion kasvun myötä myös yksityisen ja julkisen sektorin palvelut ovat alkaneet siirtymään Internetiin enenevissä määrin. Palveluiden digitalisoituminen tuo merkittäviä säästöjä palveluntarjoajille, eikä asiakkaiden tarvitse enää lähteä kotisohvalta pois niitä käyttääkseen (Selovuo 2019, 5). Palvelut ovat siis lähempänä meitä kuin koskaan, mutta vain jos ne on toteutettu tavalla, joka mahdollistaa niiden käytön kaikille toimintarajoitteista huolimatta.

Palveluiden digitalisoitumisen vuoksi on äärimmäisen tärkeää, että ne ovat saavutettavissa kaikille, eikä kukaan jää tämän digitalisoituvan yhteiskunnan ulkopuolelle. (Selovuo 2019, 5).

3.1 WCAG 2.1

WCAG on kansainvälisen Word Wide Web –konsortion (W3C) kehittämä ohjeistus saavutettavalle verkkosisällölle. Ohjeistuksen tehtävänä on mahdollistaa Internetin käyttö mahdollisimman suurelle osalle ihmisistä, myös heille, joilla on jonkin tyyppinen vamma tai rajoite. Näitä vammoja ja rajoitteita ovat esimerkiksi näkö- ja kuulorajoitteet, kognitiiviset ja fyysiset rajoitteet sekä oppimisvaikeudet ja neurologiset sairaudet. (W3C 2018).

Ensimmäinen versio ohjeistuksesta julkaistiin 2000-luvun vaihteessa. Ohjetta on sen jälkeen päivitetty ahkerasti teknologian ja tiedon kehittyessä, ja sen viimeisin versio (WCAG 2.1) julkaistiin 5.6.2018. Seuraava päivitysversio WCAG 3.0 julkaistaan näillä näkymin vuonna 2021. (W3C 2020).

Ohjeistus koostuu taulukossa 1 esitetyistä neljästä periaatteesta: havaittavuus, hallittavuus, ymmärrettävyys ja toimintavarmuus. Näiden neljän osan alle on jaoteltu ohjeet, jotka määrittävät tarkemmin yleiset tavoitteet ja puitteet. Tasoja ohjeistuksessa on kolme: A-, AA- ja AAA-taso. Näistä tasoista AAA on vaativin toteuttaa mutta se on saavutettavuudeltaan kattavin. Ohjeistuksessa otetaan kantaa etenkin digitaalisten palveluiden tekniseen toteutukseen. (W3C 2018; Saavutettavasti 2020).

Periaate	Selitys
Havaittavuus	Kaikki sisältö on käyttäjän havaittavissa
Hallittavuus	Käyttöliittymä ja sen navigointi on helppoa eri tekniikoilla

Ymmärrettävyys	Sisältö on selkeää ja toiminnallisuudet ovat helposti ymmärrettävissä
Toimintavarmuus	Sisällön on oltava käytettävissä eri laitteilla, selaimilla ja avustavilla teknologioilla

Taulukko 1: WCAG 2.1 –ohjeistuksen neljä periaatetta

3.2 Saavutettavuusdirektiivi

Vuonna 2016 Euroopan unionissa laadittiin direktiivi takaamaan digitaalisten palveluiden tasaarvo. Direktiivin tarkoituksena on taata julkisen sektorin verkkopalveluiden käytettävyys kaikille, käyttäjän vammasta tai toimintaesteestä huolimatta. (2016/2102).

Direktiivin standardi (EN 301 549 luvut 9-11) on lähes identtinen WCAG:n 2.0 AA-tason saavutettavuusohjeistuksesta, jonka uudempaa versiota (2.1) käytetään yleisesti mittapuuna saavutettavuustestauksissa. Tämä Euroopassa laadittu nk. ”saavutettavuusdirektiivi” astui voimaan Suomessa 1.4.2019 säädetyn digitaalisten palveluiden tarjoamisen lain nojalla. (306/2019; Selovuo 2019, 5, 17-20).

4 Semanttinen Html

Html on merkkauskieli, jolla luodaan hypertekstidokumentteja, eli verkkosivuja, joita on mahdollista lukea Internet-selaimella. Yksinkertaistettuna verkkosivut ovat todella kehittyneitä asiadokumentteja, kuten tämä opinnäytetyö, jotka sisältävät linkkejä toisiin asiadokumentteihin. Kuten koulujen dokumenttipohjissa, myös verkkosivuilla on vakiintuneita käytäntöjä, joita seuraamalla verkkosivut ovat helppolukuisia, ja ne toimivat odotetulla tavalla. (Hostinger 2019).

Kun puhutaan semanttisesta Html:stä, sillä viitataan oikeaoppiseen, ja vakiintuneeseen tapaan käyttää Html-merkkauskieltä. Html koostuu kahdenlaisista elementeistä, semanttisista ja ei-semanttisista. Semanttiset elementit välittävät tietoa itsestään kuten esimerkiksi elementin merkityksen ja tarkoituksen. Semanttinen Html on siis näiden semanttisten elementtien tarkoituksen ymmärtämistä, ja niiden oikeanlaista käyttöä. (Html 2020).

4.1 Html vaikuttaa ruudunlukijaan

Näkövammaisille Internetin käyttäjille yksi tärkeimpiä työkaluja on ruudunlukijaohjelmat. Ne mahdollistavat verkkopalvelujen käytön ilman visuaalista vahvistusta, sillä ruudunlukijaohjelmat kääntävät digitaalisen informaation puheeksi. Ruudunlukijaohjelmia käytetään yleisesti tietokoneen näppäimistöllä, ja ne sisältävät useita verkkopalvelun selaamista nopeuttavia ominaisuuksia, kuten linkki- ja otsikkotasojen- listaukset. (Webaim 2020).

Yleisimpiä ruudunlukijoita ovat Applen VoiceOver (jota käytetään tässä opinnäytetyössä), Windowsin Narrator, Jaws ja NVDA. Käytetyimmät mobiiliruudunlukijat ovat Androidin TalkBack ja Mac OSX:n VoiceOver. (Webaim 2019).

Ruudunlukijat on kehitetty havaitsemaan verkkosivuja oikeaoppisen html-kielen mukaan. Ruudunlukijat analysoivat verkkosivujen rakenteen tunnistamalla niille opetetut html-elementit. Elementtien tunnistamisen avulla käyttäjät, jotka selaavat verkkosivuja apuohjelmilla, pystyvät etsimään verkkosivuilta informaatiota nopeammin ja helpommin. (Webaim 2017).

Tämän vuoksi on erityisen tärkeää, että verkkopalvelut kehitetään vakiintuneita ja oikeaoppisia käytänteitä seuraten. Näiden vakiintuneiden käytänteiden noudattaminen tekee verkkopalvelun käyttämisestä ennalta-arvattavaa, joka taas parantaa verkkopalvelun saavutettavuutta ja käytettävyyttä. (Webaim 2017).

4.2 Html-merkkauskielen saavutettavuuteen vaikuttavat käytänteet

Html on rakenteellinen merkkauskieli josta saa parhaan ja ymmärrettävimmän lopputuloksen aikaiseksi seuraamalla sen kielioppia tarkkaan. Useimmilla html-elementeillä on semanttinen tarkoitus, joka tarkoittaa sitä, että elementti välittää eteenpäin informaatiota siitä mitä elementin sisällä oleva sisältö koskee, tai mikä sisällön tarkoitus on. Hyvänä esimerkkinä tästä on H1-otsikkotaso. Se kertoo, että H1-elementin sisällä oleva teksti on sivun tärkein otsikko. (Html 2020).

Html-elementit vaikuttavat kahteen asiaan, sisällön ulkoasuun, ja siihen miten selaimet tunnistavat elementit. Oikeaoppinen html vaikuttaa siis siihen, miten ihmiset näkevät sisällön, ja miten selaimet tunnistavat sen. (Html 2020).

Semanttinen html on myös tärkeää hakukoneoptimoinnin kannalta, sillä hakukoneet antavat suuremman painoarvon avainsanoille, jotka ovat otsikkoelementtien tai linkkien sisälle, kuin esimerkiksi otsikoksi tai linkiksi muotoilulle ei-semanttisille div- ja span-elementeille. (MDN 2020).

4.2.1 Rakenne-elementit

WCAG 2.1-saavutettavuusohjeistuksen kriteeri 1.4.1 vaatii, että verkkopalvelussa olevan informaation rakenne ja suhteet on mahdollista selvittää apuohjelmilla (W3C 2018). Tämän kriteerin voi täyttää esimerkiksi merkitsemällä verkkopalvelun eri osat Html:n rakenne-elementeillä taulukon 2 osoittamalla tavalla, joka tekee verkkopalvelusta helpomman navigoida ruudunlukijalla. Rakenne-elementeillä kuvataan dokumentin, eli verkkosivun, sisäistä rakennetta (Korpela 2011, 86). Näitä rakenteita ovat esimerkiksi:

- Ylätunniste <header>
- Navigaatio <nav>
- Pääsisältö <main>
- Alatunniste <footer>
- Sivusisältö <aside>

<table>
<tr><td colspan="3"><header></td></tr>
<tr><td rowspan="2"><nav></td><td rowspan="2"><main></td><td rowspan="2"><aside></td></tr>
<tr></tr>
<tr><td colspan="3"><footer></td></tr>
</table>

Taulukko 2: Yksinkertainen sivupohja rakenne-elementeillä

Myös verkkosivun hakutoiminto kannattaa merkitä erikseen, mutta sitä varten ei ole vielä olemassa omaa rakenne-elementtiä. Hyvä käytäntö on lisätä hakulomakkeeseen Aria role -attribuutti ”role=search”. Lisäämällä Aria roolin hakulomakkeeseen, ruudunlukija tunnistaa elementin hakutoimintona, ja käyttäjällä on mahdollista hypätä hakuun ruudunlukijan toimintolistauksen kautta. (W3C 2020).

4.2.2 Otsikkotasot

Html-merkkauskielessä on yhteensä kuusi eri otsikkotasoa, joilla ilmaistaan otsikoiden jäsennystasoja (Korpela 2011, 100). Yhdellä sivulla saa olla vain yksi H1-tason otsikko, joka ilmaisee sivun tärkeimmän asian. H1-tason jälkeen tärkein on H2 jne. Jotta verkkopalvelun semanttinen rakenne säilyy, otsikkotasoja ei saa hypätä yli.

Otsikkotasojen oikeinkäyttäminen on tärkeää, sillä ruudunlukija tunnistaa eri otsikkotasot ja kerää ne yhteen verkkosivun sisällysluetteloksi. Tämän sisällysluettelon avulla käyttäjät voivat selata verkkosivun sisältöä vaivattomasti ja nopeasti.

4.2.3 Linkit

Linkit ovat verkkosivujen tärkeimpiä elementtejä ja niiden avulla ihmiset ympäri maailman siirtyvät verkkosivulta toiselle. Tämän fundamentaalisen toiminnon vuoksi linkkien saavutettavuuden tulisi olla prioriteettilistalla hyvin korkealla. (Webaim 2020).

Linkkien oikein käyttäminen on onneksi helppoa, eikä se vaadi muuta kuin Html:n ankkurilinkki-elementin käyttämistä, ja linkin selkokielistä nimeämistä. Usein linkkien saavutettavuushaasteet liittyvät juurikin niiden nimeämiseen. Esimerkiksi yleisesti käytetty ”Lue lisää”-linkki voi aiheuttaa haasteita ruudunlukijan käyttäjälle, mikäli linkin ympäröivistä elementeistä ei pysty helposti päättelemään linkin kontekstia. Linkin nimen tulisikin ilmaista linkin kohdetta mahdollisimman tarkasti. (Papunet 2020).

Verkkopalvelujen visuaalisille käyttäjille linkkien tarkat kuvaukset saattavat tuntua vieraalta. Linkkien pitkät nimet voivat jopa huonontaa verkkosivujen käyttökokemusta ja sen visuaalista ilmettä. Näissä tilanteissa voi olla perusteltua nimetä linkki esimerkiksi ”Lue lisää” tai ”Paina tästä”. Mikäli linkit nimetään edellä mainittuun tapaan, on suositeltavaa lisätä linkkiin sen toimintaa kuvaava Aria-label –attribuutti. Kun ruudunlukija havaitsee elementissä Aria-label –attribuutin, ruudunlukija sivuuttaa itse linkin nimen, ja lukee vain Aria-attribuutin.

4.2.4 Painikkeet

Html-merkkauskielessä on useita eri tapoja tehdä painikkeita. Näistä semanttisin on käyttää Html:n omaa button-elementtiä, joka on esitelty kuviossa 1. Button-elementtiin on sisäänrakennettu monia hyviä ominaisuuksia jotka parantavat saavutettavuutta (W3C 2020). Näitä ovat muun muassa:

- Oletustyylit jotka helpottavat erottamaan painikkeen muusta sisällöstä
- Ruudunlukija tunnistaa elementin painikkeena
- Elementti on klikattava ja siihen voi siirtää kohdennuksen näppäimistöllä

Nämä ominaisuudet tekevät painikkeesta semanttisen. Painikkeen voi rakentaa myös ei-semanttisista div- ja span-elementeistä, mutta se vaatii enemmän työtä kuten kuviossa 2 on esitetty, ja toiminnallisuus pitää lisätä itse esimerkiksi JavaScript-ohjelmointikieleltä. (W3Schools 2020; MDN 2020).

```
<button>Lähetä</button>
```

Kuvio 1: Yksinkertainen Html-merkkauskielen painike-elementti

```
<div tabindex="0" role="button" style="box-shadow: 0px 0px
2px 0px #000000; padding: .1rem .3rem; border-radius:
3px">Lähetä</div>
```

Kuvio 2: Ei-semanttinen kokoomaelementti muokattuna samoihin ominaisuuksiin yksinkertainen painike-elementti

5 WAI-ARIA

Aria-attribuutit ovat niin olennainen osa verkkosivujen saavutettavuutta, että siitä on syytä kirjoittaa oma kappaleensa. WAI-ARIA tulee sanoista Web Accessibility Initiative - Accessible Rich Internet Application suite. (W3C 2020).

Aria-attribuutit on kehitetty ruudunlukijoita varten, ja niiden tarkoitus on parantaa web-sisällön saavutettavuutta. Tämän teknologian avulla käyttäjät voivat saada ylimääräistä tietoa sivujen ja eri komponenttien tiloista, jota visuaaliset käyttäjät eivät näe. Aria-attribuutit ovat siis tehty nimenomaan niille käyttäjille, jotka selaavat Internetiä pelkästään näppäimistöllä. (Selovuo 2019, 95-96).

On kuitenkin mainittava, että WAI-ARIA ei ole ensisijainen ratkaisu verkkopalvelun saavutettavuuteen. Mikäli verkkopalvelu on rakennettu oikeaoppisella Html-merkkauskielellä, WAI-ARIA:lle on harvoin tarvetta. (MDN 2020).

Aria-attribuutteja on kehitetty valtava määrä eri käyttötarkoituksiin (W3C 2020). Verkkopalveluja kehittäessä olen kuitenkin huomannut, että näistä sadoista eri attribuuteista on vain kolme, joita jatkuvasti käytän. Nämä kolme attribuuttia on esitelty seuraavissa alaluvuissa.

5.1 Aria-label

Aria-labelin avulla voidaan määrittää mitä ruudunlukija lukee attribuutin omaavan elementin kohdalla. Kuten kuviossa 3 on esitetty, verkkopalvelujen visuaaliset käyttäjät näkevät painikkeen tekstinä ”Lue lisää”, mutta ruudunlukija kertoo käyttäjälle ensin elementin tarkoituksen eli ”linkki”, ja sen jälkeen aria-labelin arvon ”10 syytä nimetä linkki oikein”. Aria-label siis yliajaa elementille annetun arvon (esimerkiksi ”Lue lisää”). (W3C 2020).

```
<a href="#" aria-label="10 syytä nimetä linkki oikein">Lue lisää</a>
```

Kuvio 3: Linkki jolle on annettu arvo Aria-label -attribuutilla

5.2 Aria-expanded

Attribuutin avulla voidaan ilmoittaa ruudunlukijalle dynaamisten sisältöelementtien tila. Yksi tällaisista sisältöelementeista on navigaatiovalikko, joka avautuu ja sulkeutuu. Lisäämällä Aria-expanded –attribuutti sisältöelementin avaa/sulje-painikkeeseen, kuten kuviossa 4 on esitetty, ruudunlukija voi kertoa käyttäjälle onko elementti avattuna vai suljettuna. Kun painiketta klikkaa ja valikko avautuu, myös aria-expanded –attribuutin arvo vaihtuu ”false” ja ”true” välillä. (ADG 2018).

```
<button aria-expanded="true" id="menu" href="#"
onclick="toggle('navbar', 'menu')">Valikkopainike</button>
```

Kuvio 4: Valikkopainikkeen Aria-expanded -attribuutti on "true" kun valikko on avattu

5.3 Aria-role

Role-attribuutilla voidaan määrittää elementille uusi semanttinen merkitys. Esimerkiksi kokoomaelementistä (div- tai span-elementti) voidaan tehdä artikkelin, painikkeen tai hakulomakkeen. Role-attribuutti toimii siis samaan tapaan kuin Html:n rakenne-elementti. (Webaim 2020).

Hyvä esimerkki role-attribuutin käyttämisestä on kuviossa 5 esitetty hakulomake. Antamalla hakulomakkeelle semanttisen roolin ”search”, ruudunlukija tunnistaa lomakkeen hakulomakkeena ja käyttäjä voi hypätä siihen ruudunlukijan pikanäppäimen kautta. (W3C 2020).

```
<form role="search">
  <label for="search-form">Hae sivulta:
</label>
  <input type="text" id="search-form">
  <input type="submit" value="Hae">
</form>
```

Kuvio 5: Ruudunlukija tunnistaa lomakkeen hakulomakkeeksi role-attribuutin avulla

6 Tutkimus- ja kehittämismenetelmät

Tutkimusmenetelmänä käytettiin auditointia. Tutkimustieto kerättiin auditoimalla luvussa 7 esitetty verkkosivu. Auditoinnin tarkoituksena oli tuoda näkyväksi erilaiset toiminnot ja prosessit sekä vakiintuneet käytänteet. Auditointi on tehokas tapa tuoda esille henkilökunnan hiljaista tietoa. (Haapakorpi 2011, 41-42).

Opinnäytetyössä kehitetyssä saavutettavuusauditointiprosessissa kehittämismenetelmänä käytettiin prosessin kehittämisen menetelmää. Prosessin kehittämisellä tähdätään yleensä yrityksen toiminnan parantamiseen ja tehostamiseen. (JHS 2012).

Prosessikuvaukset auttavat hallitsemaan ja jäsentämään kokonaisuuksia, sekä auttaa löytämään toiminnasta tehostamistarpeita. Prosessikuvauksia voi myös käyttää esimerkiksi uusien työntekijöiden perehdyttämiseen ja kouluttamiseen. Kuvausten avulla on myös mahdollista kerätä hiljaista tietoa. (JHS 2012).

Yleensä prosessin kehittäminen aloitetaan, kun eteen tulee ongelma, johon halutaan löytää ratkaisu. Kehittämisen laajuus voi vaihdella pienen osa-alueen parantamisesta uusien menetelmien käyttöönottoon. (JHS 2012).

Prosessin kehittäminen aloitetaan yleensä kehittämistarpeen havaitsemisesta, josta siirrytään prosessin kuvaamiseen. Prosessin kuvaamisen eri vaiheet ovat: kuvattavan prosessin valitseminen, käyttötarkoituksen ja kuvauksen tarkkuuden päättäminen, kuvaustapojen valitseminen, prosessin kuvaaminen ja kokonaisuuteen sovittaminen. (JHS 2012).

Prosessin kuvaus koostuu yleensä kolmesta eri asiasta:

1. Perustiedoista, joista käy ilmi prosessille tärkeät asiat.
2. Sanallisesta kuvauksesta, joka sisältää muun muassa prosessin eri vaiheet, toiminnot, toimijat ja tehtävät.
3. Kaavio, joka on prosessin graafinen kuvaus.

Prosessin kuvaamisen tarkkuus on myös hyvä määrittää. Näitä tarkkuustasoja ovat muun muassa prosessikartta, toimintamalli, prosessin kulku ja työnkulku. Näistä tarkin kuvaus on työnkulku, jota tullaan käyttämään tässä opinnäytetyössä. (JHS 2012).

Reliabiliteetilla ja validiteetilla määritetään, onko tutkimusmenetelmät ja tuloksista johdetut päätelmät päteviä ja oikeutettuja (Hiltunen 2009. Reliabiliteetilla mitataan mittaus- tai tutkimusmenetelmän ja tutkimustulosten luotettavuutta (Hiltunen 2009). Validiteetilla taas ilmaistaan sitä, miten mittaus- tai tutkimusmenetelmä mittaa tutkittavaa asiaa, eli sitä, mitä on

tarkoitus mitata (Hiltunen 2009, Hirsijärven 2002, 213 mukaan). Tulosten validiteetilla tarkoitetaan tutkimuksen pätevyyttä ja oikeellisuutta (Hiltunen 2009, Nummenmaan 1997, 203 mukaan).

7 Case: järjestöhautomo.fi saavutettavuusauditointi

Case-tutkimuksen tarkoituksena on kerätä tutkimustietoa kehitettävää auditointiprosessia varten. Tutkimustieto kerätään suorittamalla saavutettavuusauditointi jarjestohautomo.fi-sivuston etusivulle. Etusivu on valittu siksi, että siitä löytyy monipuolisesti erilaisia verkkokomponentteja, ja täten saavutettavuusauditoinnista tulee mahdollisimman monipuolinen.

Saavutettavuusauditoinnin jokainen vaihe dokumentoidaan tarkasti. Jokainen sivulta löydetty saavutettavuusvirhe kirjataan ylös ja kuvankaappauksia käytetään havainnoimaan paitsi virheitä, myös auditointiohjelmien käyttöä.

Tämä dokumentaatio tullaan myöhemmin analysoimaan kehitettävän saavutettavuusauditointiprosessin perustaksi.

7.1 Taustaa

Suomen Pakolaisapu pyysi Agenda Helsinkiä suorittamaan saavutettavuusauditoinnin kolmelle sivustolle. Nämä sivustot olivat pakolaisapu.fi, jarjestohautomo.fi ja empathy.fi.

Projektin laajuuteen sisältyi jokaisen sivuston ensimmäinen auditointi (kaikki sivustot kokonaisuudessaan), saavutettavuuskorjausten jälkeinen toinen auditointi, sekä saavutettavuusseloste.

Tähän opinnäytetyöhön dokumentoitava esimerkki koskee sivuston jarjestohautomo.fi etusivun ensimmäistä auditointia.

7.2 Dokumentointi

Saavutettavuusongelmien dokumentoimiseen käytän Google Driven Sheets-työkalua. Sheets on helppokäyttöinen taulukko-ohjelma joka muistuttaa Microsoft Exceliä. Sheets valikoitui käyttööni koska se on ilmainen pilvipalvelu ja samaa taulukkoa voi muokata useampi ihminen samaan aikaan. Jälkimmäinen syy mahdollistaa useamman työntekijän osallistumisen samaan projektiin, kunhan dokumenttiin on selvästi merkitty mitkä osiot ovat jo tehty, ja mitkä ovat kenelläkin työn alla.

Taulukko on helppo ja järjestelmällinen tapa kirjata ylös saavutettavuusongelmia ja seurata omaa edistymistä. Siitä on myös helppo siirtää informaatiota myöhemmin itse saavutettavuusraporttiin.

Kuviossa 6 esitetty taulukkomallin pohja on otettu työpaikallani käytetystä verkkosivujen loppuaudioinnin virhetaulukosta, johon verkkosuunnittelijat merkitsevät kehitettävänä olevien verkkosivujen kosmeettisia virheitä (esim marginaalit, välistykset, fonttivirheet yms.). Taulukkomalli on jokaiselle työntekijälle tuttu, joten sen käyttöönotto sujuu vaivatta. Taulukkoon on tehty muutamia lisäyksiä saavutettavuusauditointeja varten.

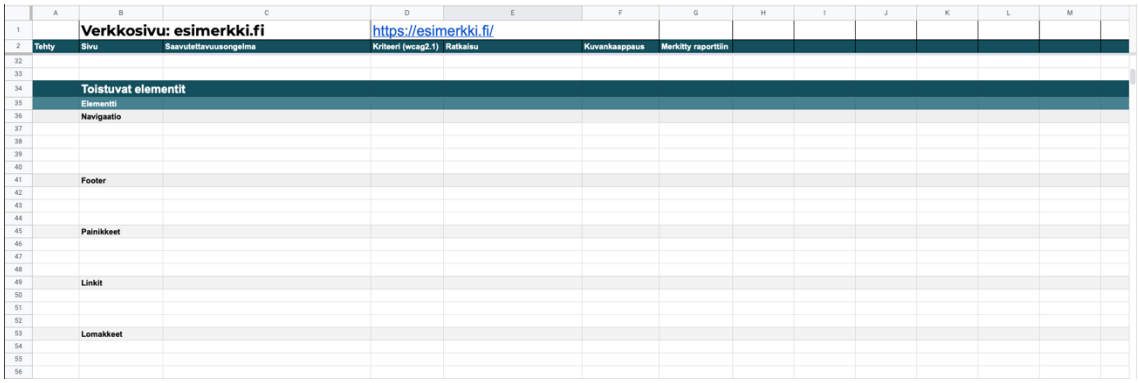
Taulukko on jaettu seitsemään sarakkeeseen:

- Tehty (merkitään kun sivu/komponentti on auditoitu)
- Sivu/toistuva elementti (merkitään millä sivulla tai missä elementissä ongelma on)
- Saavutettavuusongelma
- Kriteeri (merkitään WCAG 2.1-kriteeri)
- Ratkaisu
- Kuvankaappaus (otetaan kuvankaappaus ongelmasta, mikäli tarpeellista)
- Merkitty raporttiin (merkitään kun kohta on siirretty lopulliseen saavutettavuusraporttiin)

	A	B	C	D	E	F	G
1		**Verkkosivu: esimerkki.fi**		https://esimerkki.fi/			
2	**Tehty**	**Sivu**	**Saavutettavuusongelma**	**Kriteeri (wcag2.1)**	**Ratkaisu**	**Kuvankaappaus**	**Merkitty raporttiin**
3		**Etusivu**					
4							
5							
6							
7							
8							
9							

Kuvio 6: Taulukosta löytyy otsikot tehty, sivu, saavutettavuusongelma, kriteeri, kuvankaappaus ja merkitty raporttiin

Teen taulukkoon myös oman kohdan toistuvia elementtejä varten, kuten on esitetty kuviossa 6. Suuri osa saavutettavuusongelmista liittyy toistuviin elementteihin kuten painikkeisiin, linkkeihin ja lomakkeisiin, joita ei yleensä kehitetä sivukohtaisesti, vaan uusiokäytetään läpi sivuston. Niitä ei ole syytä raportoida joka auditoitavan sivun kohdalla uudestaan, vaan yksi kerta riittää.



	Verkkosivu: esimerkki.fi		https://esimerkki.fi/			
Tehty	Sivu	Saavutettavuusongelma	Kriteeri (wcag2.1)	Ratkaisu	Kuvankaappaus	Merkitty raporttiin
	Toistuvat elementit					
	Elementti					
	Navigaatio					
	Footer					
	Painikkeet					
	Linkit					
	Lomakkeet					

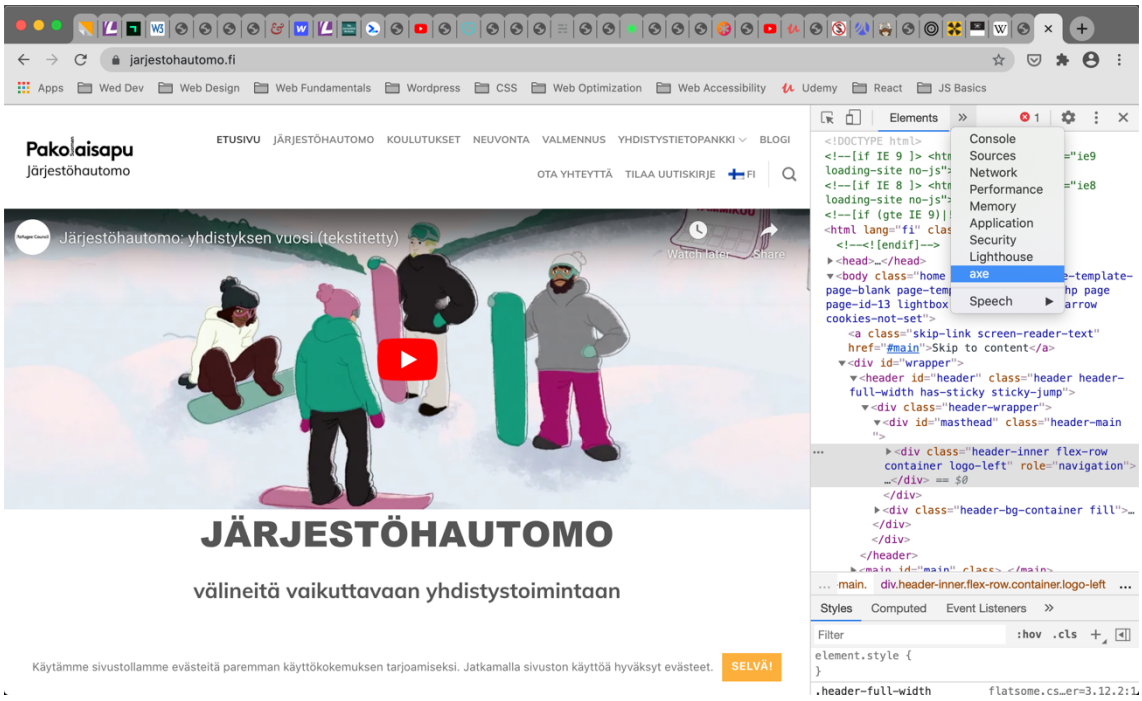
Kuvio 7: Taulukkoon raportoidaan myös toistuvat elementit

Luon samaan dokumenttiin toisen välilehden mobiiliauditointia varten. Kopion uudelle välilehdelle jo aiemmin tehdyn taulukon.

7.3 Ohjelmisto

Automaattiseen tarkistukseen käytän Deque Axe –ohjelmistoa. Olen käyttänyt auditointeihin myös WebAimin WAVE-ohjelmaa ja Google Chrome-selaimen lighthouse-työkalua. Dequen Axe on ollut käyttämistäni saavutettavuusohjelmista tarkin ja kattavin, ja Axe kertoo jokaista saavutettavuusongelmaa koskevan WCAG-saavutettavuuskriteerin ongelman yhteydessä.

Axe-ohjelma on helppo asentaa suoraan Google Chrome –selaimeen, jonka jälkeen se löytyy Chromen kehittäjän työkaluista. Kuviossa 8 näytetään Axe-ohjelman sijainti Google Chrome-selaimessa.



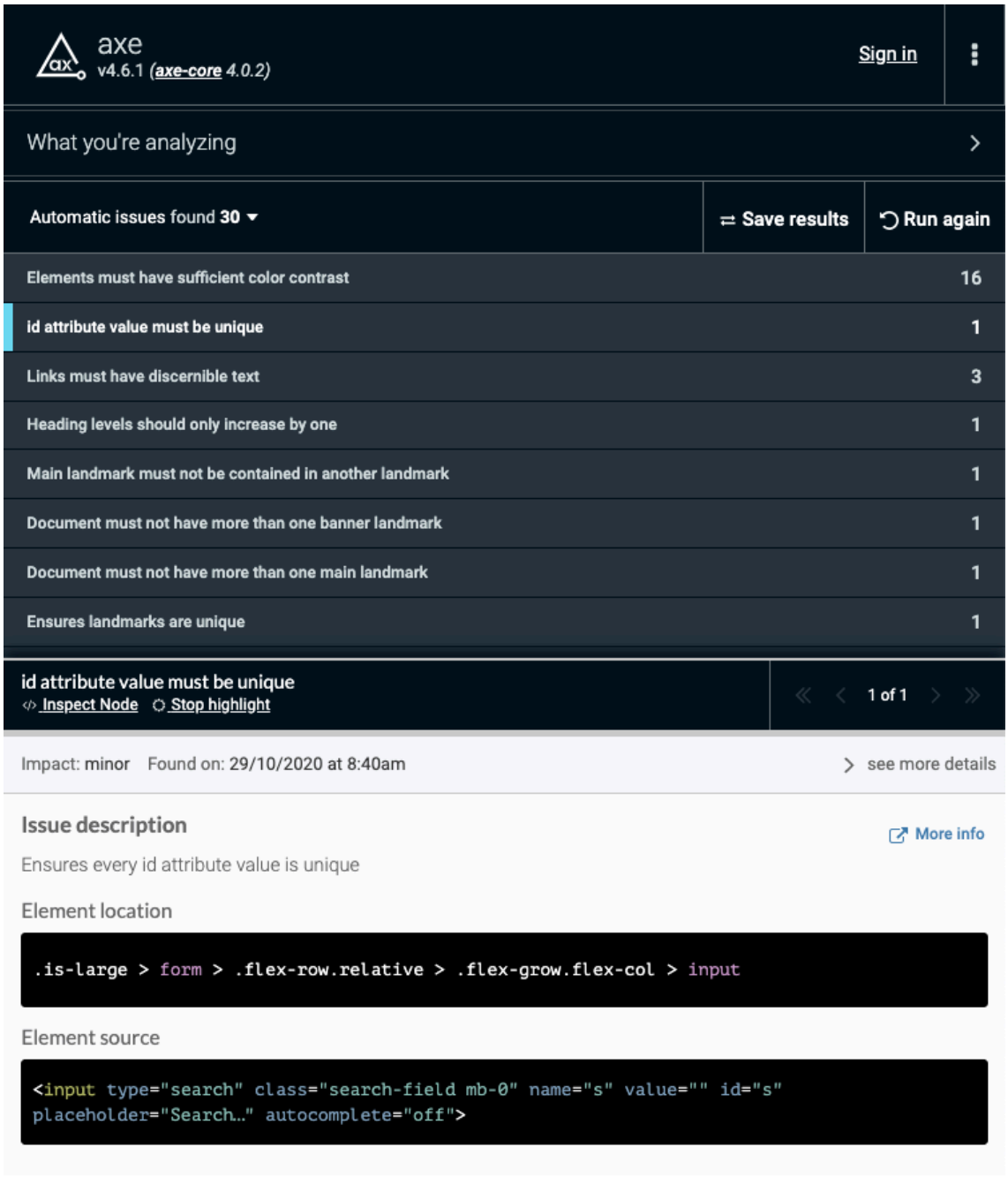
Kuvio 8: Asennuksen jälkeen Axe-ohjelma löytyy Chromen kehittäjätyökaluista

Axe analysoi verkkopalvelusta aina yhden sivun kerrallaan, eli auditointi tulee tehdä aina jokaiselle sovitulle sivulle erikseen. On olemassa myös maksullisia auditointiohjelmia, jotka skannaavat koko sivuston läpi kerralla, ja muodostavat auditoinnista automaattisen raportin. Vaikka koko sivuston auditointi kuulostaa teoriassa hyvältä, on siinä muutamia ongelmia. Sivun automaattisen auditoinnin yhteydessä pitää jokainen ongelmakohta tarkistaa manuaalisesti itse ja todentaa, että ongelma on validi. Joistain ongelmista tulee myös ottaa kuvankaappaukset, jotka liitetään lopulliseen saavutettavuusraporttiin ongelman havainnollistamiseksi. Tämä tarkoittaa sitä, että sivuston auditointi pitää joka tapauksessa tarkistaa itse, eikä koko sivuston automaattinen auditointi tarjoa täten oikoreittiä nopeampaan auditointiin.

7.4 Saavutettavuusongelmien auditointi

Kun Axe-ohjelma on otettu käyttöön Chrome-selaimen kehittäjän työkaluista, käyttäjän tarvitsee vain klikata ”Analyze”-painiketta ja ohjelma tekee automaattisen saavutettavuusauditoinnin sille sivulle, joka on sillä hetkellä selainikkunassa aktiivisena.

Kun auditointi on valmis, Axe näyttää kuviossa 9 esitetyn listauksen sivulta löytyneistä saavutettavuusongelmista.



Kuvio 9: Axe listaa saavutettavuusongelmat ja antaa ongelmakuvaukset

7.4.1 Kontrasti

Aloitan listan ensimmäisestä kohdasta, joka kertoo, että etusivulla on 16 elementtiä jotka eivät läpäise saavutettavuusohjeistuksen vaatimusta riittävästä kontrastierosta (1.4.3 Kontrasti). Käyn jokaisen kohdan läpi ja otan kuvankaappaukset Axen osoittamista elementeistä (mikäli elementit on tehty samasta toistuvasta komponentista, otan vain yhdestä kuvankaappauksen).

Ennen tarkistetun elementin kirjaamista taulukkoon, määritän, onko kyseessä toistuva elementti vai sivukohtainen elementti. Kaikki Axen osoittamat 16 elementtiä, joissa on riittämätön kontrasti, ovat toistuvia elementtejä. Kirjaan raporttiin merkinnän toistuvien komponenttien kohdalle.

7.4.2 Toistuvat id-attribuutit

Seuraava Axen havaitsema ongelma liittyy id-attribuutteihin. Ohjelma havaitsi kahdella eri elementillä saman id-attribuutin, mikä on semanttisesti väärin sekä saavutettavuusohjeistuksen kriteerin 4.1.1 vastainen.

Hetken lähdekoodia tarkkailtuani huomaan, että Dom-rakenteessa on kaksi hakulomaketta samalla id-attribuutilla. Yksi pöytäkone-versiolle ja yksi mobiili-versiolle. Kirjaan ongelman taulukkoon ja kirjoitan ylös myös ratkaisun, jonka voin myöhemmin kopioida asiakkaalle lähetettävään saavutettavuusraporttiin. Otan myös lähdekoodista kuvankaappauksen, jonka voin liittää raporttiin havainnollistavaksi avuksi. Kuviossa 10 on esitetty esimerkki raporttiin liitettävästä kuvankaappauksesta, josta näkyy myös saman id-attribuutin omaavat hakulomakkeet.

```
        ▼<form method="get" class="searchform" action="https://
        jarjestohautomo.fi/" role="search">
          ▼<div class="flex-row relative">
            ▼<div class="flex-col flex-grow">
                <input type="search" class="search-field mb-0" name="s" value
                id="s" placeholder="Search…" autocomplete="off">
              </div>
            ▶<div class="flex-col">…</div>
              </div>
            ▼<div class="live-search-results text-left z-top">
              <div class="autocomplete-suggestions" style="position: absolute;
              display: none; max-height: 300px; z-index: 9999;"></div>
              </div>
            </form>
          </div>
        </div>
      </div>
      <div class="mfp-preloader">Loading...</div>
    </div>
  ▶<button title="Close (Esc)" type="button" class="mfp-close">…</button>
  </div>
  <a class="skip-link screen-reader-text" href="#main">Skip to content</a>
▶<div id="wrapper">…</div>
▼<div id="main-menu" class="mobile-sidebar no-scrollbar mfp-hide">
  ▼<div class="sidebar-menu no-scrollbar ">
    ▼<ul class="nav nav-sidebar nav-vertical nav-uppercase">
      ▼<li class="header-search-form search-form html relative has-icon">
        ▼<div class="header-search-form-wrapper">
          ▼<div class="searchform-wrapper ux-search-box relative form-flat is-
          normal">
            ▼<form method="get" class="searchform" action="https://
            jarjestohautomo.fi/" role="search">
              ▼<div class="flex-row relative">
                ▼<div class="flex-col flex-grow">
                    <input type="search" class="search-field mb-0" name="s" value
                    id="s" placeholder="Search…" autocomplete="off">
                  </div>
                ▶<div class="flex-col">…</div>
                  </div>
                ▶<div class="live-search-results text-left z-top">…</div>
                </form>
```

Kuvio 10: Lähdekoodista löytyi kaksi hakulomaketta samalla id-attribuutilla

7.4.3 Linkkien nimeäminen

Seuraava listalla oleva virhe on ”Linkeillä on oltava havaittava nimi” (2.4.4 Linkin tarkoitus, 4.1.2 Nimi, rooli, arvo). Axen mukaan virheitä on kolme kappaletta ja ne sijaitsevat sivuston alasivutunnisteessa eli footerissa. Lähdekoodista huomaan, että alasivutunnisteessa on kolme sosiaalisen median linkkiä, joista jokaisessa on sama ongelma. Linkkien tarkoitus on selitetty

tekstin sijaan ikoneilla, eikä linkeille ole annettu niiden tarkoitusta kuvaavia Aria-label -attribuutteja, kuten kuviosta 11 huomaa.

Otan kuvankaappauksen lähdekoodista, ja alasivutunnisteen linkeistä ja merkitsen ne raporttiin. Kirjaan ylös myös ratkaisuehdotuksen.

```
▼<a href="https://fi-fi.facebook.com/jarjestohautomo" target=
"_blank" data-label="Facebook" rel="noopener noreferrer nofollow"
class="icon button circle is-outline facebook tooltip
tooltipstered">
  ▼<i class="icon-facebook">
     ::before
   </i>
 </a>
```

Kuvio 11: Some-linkistä puuttuu kuvaava teksti

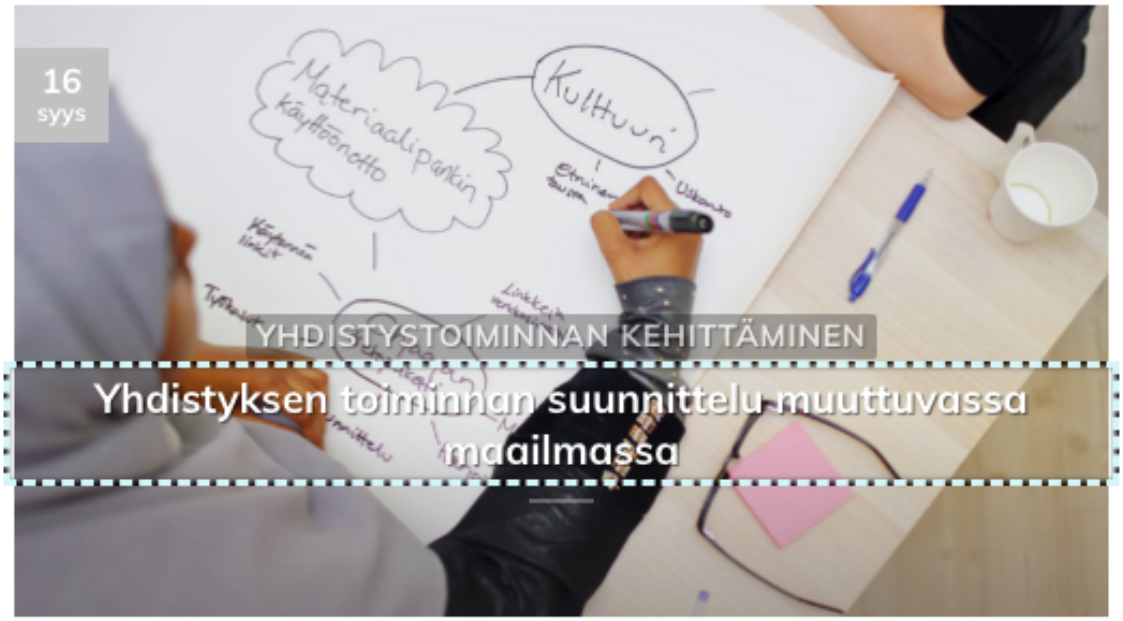
7.4.4 Otsikkotasot

Seuraava virhe liittyy otsikkotasoihin ja Html-merkkauskielen semantiikkaan. Otsikkotasot (H1-H6) saavat laskea aina yhdellä. Eli H1-tason jälkeen ei saa laskea tasoon H3, vaan välissä on käytettävä H2-tasoa.

Axe-ohjelmassa on kuvion 12 osoittama käytännöllinen ominaisuus, joka korostaa virheen sijainnin auditoitavalla verkkosivulla.

Kuvio 12: Highlight-painikkeella voi korostaa virheen sijainnin

Klikkaan kohtaa ”Highlight” ja Axe luo kuviossa 13 näkyvän tarkennusrajan saavutettavuusongelman ympärille.



Kuvio 13: Axe luo tarkennusrajat saavutettavuusongelman ympärille

Tarkistan vielä lähdekoodista otsikkotasojen kehittymisen ja huomaan, että etusivun otsikkotasot menevät seuraavasti: H1, H2 ja H5. Merkitsen tämän ylös raporttiin ja teen ehdotuksen siitä, kuinka otsikkotasot tulisi korjata.

7.4.5 Sisäkkäiset rakenne-elementit

Seuraava ongelma liittyy rakenne-elementteihin. Axe ilmoittaa, että sivustolla on kaksi rakenne-elementtiä sisäkkäin. Tämä on semanttisesti väärin ja voi vaikeuttaa sivuston käyttöä ruudunlukijalla. Katson ongelman kuvauksesta, että mikä rakenne-elementti on kyseessä.

Huomaan, että ei-semanttiselle div-elementille (div- ja span-elementeillä ei ole semanttista merkitystä) on annettu rakenne-rooli ”role”-attribuutilla. Katson elementin sijainnin lähdekoodista ja huomaan, että Axen osoittama elementti sijaitsee kuviossa 14 osoitetulla tavalla main-elementin sisällä. Otan ruudunkaappauksen lähdekoodista ja merkitsen saavutettavuusongelman raporttiin. Ratkaisuksi esitän ”role”-attribuutin poistamista ei-semanttisesta div-elementistä, sillä sitä ei tarvitse ulomman elementin ollessa saman roolin omaava main-elementti.

```
▼<main id="main" class>
  ▼<div id="content" role="main" class="content-area">
```

Kuvio 14: Kaksi rakenne-elementtiä sisäkkäin

7.4.6 Elementtien lopputarkastus

Nyt kun Axen ongelmalista on käyty läpi, tarkistan vielä sivuston täysin manuaalisesti kokeilemalla sivun eri toimintoja ja katsomalla sivun lähdekoodia. Huomaan, että sivulla olevien linkkien toimintaa esitetään vain värien avulla. Tämä ei täytä saavutettavuuskriteeria 1.4.1. Merkitsen ongelman raporttiin ja ehdotan ratkaisuksi aktiivisten ja kohdistettujen linkkien alleviivausta. Kuviossa 15 esitetään saavutettavuuskriteerin 1.4.1 mukainen ongelma, jossa aktiivinen linkki esitetään hieman eri värillä.

Kuvio 15: Linkkien toiminta osoitetaan vain värin avulla

7.5 Verkkosivulla liikkuminen

Tärkeä osa verkkopalvelujen saavutettavuutta on se, että palvelua voi käyttää ilman kohdistinlaitetta (hiiri). Verkkopalvelun tulee olla helposti hallittavissa myös pelkällä näppäimistöllä tai apuohjelmalla kuten ruudunlukija.

Axe-ohjelmassa on ominaisuus, joka listaa kaikki sivulta löytyvät klikattavat elementit sekä niiden sarkainjärjestyksen. En aio kuitenkaan käyttää Axea tämän osan testaukseen, vaan aion kokeilla sivulla liikkumista sekä sen toimintojen käyttämistä ihan vain näppäimistöllä sekä ruudunlukijalla.

Tärkeitä kohtia, joihin tulee kiinnittää huomiota, on sarkainjärjestyksen loogisuus, kuvien vastinetekstit, linkkien nimeämiset sekä se, että pystyykö kaikkiin klikattaviin elementteihin kohdistaan näppäimistöllä ja ruudunlukijalla.

7.5.1 Näppäimistö

Aloitan testaamisen näppäimistöllä. On olemassa selaimiin asennettavia lisäohjelmia, jotka automaattisesti tarkistavat ja merkkaavat kaikki kohdistettavat elementit, jonka jälkeen auditoija voi käydä sarkainjärjestyksen läpi silmämääräisesti. Olen aiemmin käyttänyt näitä ohjelmia (Web Insights, Axe yms.) mutta olen tullut siihen lopputulokseen, että paras tapa tutkia sarkainjärjestystä ja kohdistuksen toimivuutta, on käydä sivusto läpi näppäimistöllä. Tällöin saa oikean tuntuman siihen, miten sivusto toimii näppäimistöllä.

Aloitan testauksen päivittämällä sivun ja sen jälkeen klikkaamalla sarkainta. Sarkain kohdistuu ensimmäisenä hyppylinkkiin (hyppylinkillä voi siirtyä suoraan sivun sisältöön ohittaen navi-

gaation), joka on hyvien käytäntöjen mukaista. Ohitan hyppylinkin ja siirryn sarkaimella navigaatioon, sillä haluan testata, miten navigaatiossa oleva alasvetovalikko toimii. Huomaan kuitenkin, että sivustolta puuttuu kokonaan aktiivisen elementin kohdennin, mikä tekee sivuston käytöstä pelkällä näppäimistöllä melkein mahdotonta.

Siirryn seuraavaksi navigaatiossa olevan alasvetovalikon luokse. Alasvetovalikon tulisi aueta, kun kohdennus siirtyy alasivujen otsikoihin, mutta näin ei tapahdu, vaan kohdennus käy jokaisen alasivun navigaatiolinkin läpi mutta visuaalisesti sivustolla ei tapahdu mitään.

Loppusivu toimii kuten pitääkin, ja sarkainjärjestys on looginen. Teen jälleen merkinnät raporttiin ja kirjoitan ratkaisuehdotukset.

7.5.2 Ruudunlukija

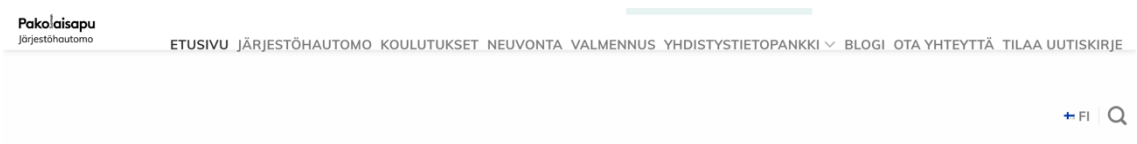
Seuraavaksi testaan sivun toimivuuden ruudunlukijalla. Ohjelmana käytän Mac Os VoiceOveria. Sivu toimii ruudunlukijalla hyvin, ja ainoat puutteet, jotka löysin, on jo merkattu aiemmin raporttiin (mm. puuttuvat alt-tekstit).

Tarkistan myös ruudunlukijan tuottaman sisältölistan sivusta, ja sekin on kunnossa yhtä maamerkki-virhettä lukuun ottamatta, joka on myös raportoitu jo aiemmin.

7.6 Tekstin suurennos

Testaan sivun toimivuuden suurennetulla tekstillä käyttäen Mozilla Firefox-selaimella. Firefox-selaimella teksti on helppo suurentaa muutamalla klikkauksella. Suurennan tekstiä 200 prosenttia ja tarkistan, että kaikki sivun elementit pysyvät niille määrätyissä paikoissa, ja että teksti on helposti luettavissa.

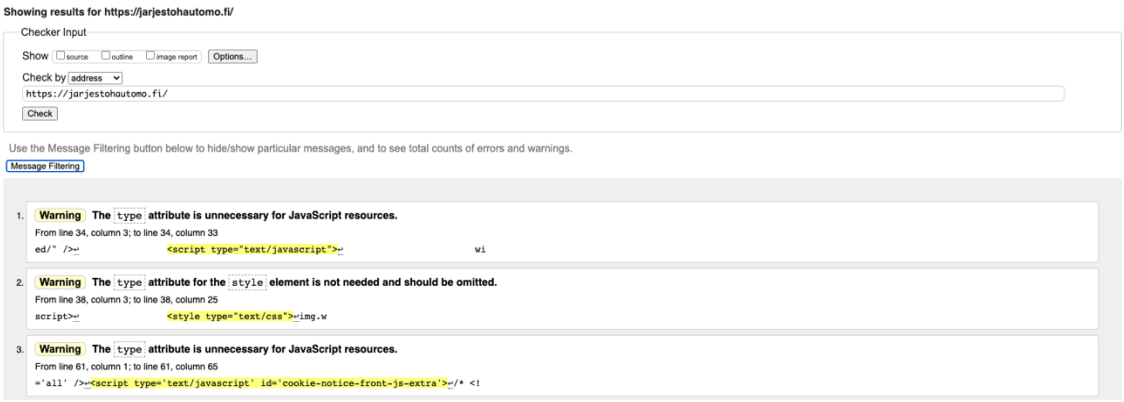
Sivusto toimii suurennetulla tekstillä hyvin, ja ainoat elementit jotka siirtyvät oman alkuperäisen paikan ulkopuolelle, ovat kuviossa 16 esitetty kielivalinta ja hakutoiminto. Ne kuitenkin siirtyvät loogisesti toiselle riville, ja pysyvät täysin käyttökelpoisina. En merkitse tätä raporttiin, sillä se ei vaikuta sivun käyttöön mitenkään, ja korjaus olisi lähinnä kosmeettinen (ja myös aikaa vievä).



Kuvio 16: Tekstiä suurentaessa haku- ja kielivalinta-painikkeet siirtyvät toiselle riville

7.7 Lähdekoodin validointi

Viimeisenä jäljellä on lähdekoodin validointi. Käytän W3C:n kehittämää Html -validaattoria. Kuten kuviossa 17 näkyy, validaattori löytää etusivun lähdekoodista 33 virhettä. Käyn läpi jokaisen virheen, ja yksikään virheistä ei ole sellainen, joka vaikuttaisi sivun toimintaan. Korjattavia virheitä ovat esimerkiksi ylimääräiset aloitus- ja lopetustagit, joita löytää harmittavan usein auditoitavilta sivuilta.



Kuvio 17: Html-validaattori näyttää useita pieniä virheitä etusivun lähdekoodista

8 Auditoinnin validointiprosessi oikeellisuuden takaamiseksi

Ennen auditointiprosessin kehittämistä, halusin varmistaa, ettei tekemässäni auditoinnissa ollut puutteita. Tuleva prosessi tehdään suoraan tekemäni saavutettavuusauditoinnin perusteella, eli mikäli auditointi olisi virheellinen tai puutteellinen, tulisi tämä virhe tai puute toistumaan myös kehitettävässä prosessissa luoden valuvian. Tällä olisi vaikutus kaikkiin kehitetyllä prosessilla tehtyihin saavutettavuusauditointeihin.

Tarkastutin tekemäni saavutettavuusauditoinnin käytettävyysasiantuntijalla. Hän teki saavutettavuusauditoinnin samalle sivulle (jarjestohautomo.fi etusivu), jonka jälkeen vertailimme saavutettavuusauditointien tuloksia. Tuloksista ei löytynyt huomattavia eroja, ja ainoat löytyneet erot liittyivät ehdotettuihin ratkaisuihin.

Tämän validoinnin perusteella voi siirtyä kehittämään saavutettavuusauditointiprosessia.

9 Saavutettavuusauditoinnin kehittäminen

Tämän luvun tarkoituksena on kuvata auditointiprosessin kehittymisen eri vaiheita, sekä esittää lopullinen prosessi joka tulee työpaikkani käyttöön saavutettavuusauditointeja varten.

Auditointiprosessin kehittäminen sai alkunsa tarpeesta yhdenmukaistaa tämän opinnäytetyön toimeksiantajayrityksen työntekijöiden tekemät auditoinnit, sekä perehdyttää että kouluttaa enemmän työntekijöitä saavutettavuusauditointeihin. Työntekijöiden koulutus on tärkeää etenkin nyt, kun saavutettavuusauditointipyyntöjä tulee Suomessa voimaan astuneen digitaalisten palvelujen tarjoamisen lain asettaman aikarajan vuoksi enenevissä määrin.

Kehitettävällä auditointiprosessilla halutaan myös nopeuttaa auditointiprojektien kokonaisaikaa, vähentää inhimillisten virheiden määrää sekä muodostaa selkeä rakenne auditointien tekemiseen.

9.1 Oman auditointiprosessin kehittyminen

Tutustuin ensimmäisen kerran saavutettavuuteen kunnolla Laurean palvelumuotoilukurssilla, jossa tutkin yhdessä opiskelijaryhmäni kanssa Laurean verkkosivujen saavutettavuutta. Tutkimme verkkosivujen toimintaa ja teimme käyttäjätestauksia näkövammaisten palvelu- ja toimintakeskus Iiriksessä. Käyttäjätestauksien yhteydessä tutustuimme myös Annanpuran toimintaan, joka on yksi Suomen johtavista saavutettavuusasiantuntijayrityksistä.

Kurssilla tehty yhteistyö Iiriksen ja Annanpuran kanssa antoi loistavan pohjatiedon ja -ymmärryksen saavutettavuudesta. Kaikilla tapaamillamme käyttäjätestaajilla ja asiantuntijoilla oli jonkin asteinen näkövamma, joten pääsimme oppimaan aiheesta suoraan heiltä, joihin verkkopalvelujen saavutettavuus vaikuttaa.

Tarkempi tekninen osaaminen on syntynyt oman tutkimuksen myötä. Olen verkkokehittäjänä melkein päivittäin tekemisissä verkkosivujen saavutettavuuden kanssa, sillä kaikki tekemäni asiakasprojektit tehdään WCAG 2.1 AA-tason mukaisiksi. Jokainen tekninen ratkaisu tulee siis tehdä huolella ja saavutettavuus mielessä. Tämän vuoksi on ollut tärkeää löytää hyviä tiedonlähteitä saavutettavaan verkkokehittämiseen.

Internetistä löytyy valtava määrä tietoa saavutettavuudesta, mutta parhaat löytämäni lähteet ovat olleet W3.org (WCAG-saavutettavuusohjeistuksen kehittäjä), ADG (Sveitsiläisen säätiön rahoittama verkko-oppimisalusta joka on suunnattu kehittäjille) ja Inclusive Components (e-kirja saavutettavista verkkokomponenteista). Kaikki edellä mainitut lähteet on tehty yhteistyössä ihmisten kanssa, joilla on jonkin asteinen vamma tai toimintarajoite.

Itse prosessi, jolla teen saavutettavuusauditointeja, on muodostunut opiskelemalla saavutettavuuteen liittyvää tutkimusta, sekä takaisinmallintamalla asiantuntijoiden tekemiä saavutettavuusauditointeja. Näihin saavutettavuusauditointeihin olen päässyt tutustumaan työni kautta, sillä teen ajoittain asiakasprojekteina saavutettavuuskorjauksia eri verkkopalveluihin. Ennen saavutettavuuskorjausprojektin aloittamista, on aina tarpeellista tutustua kyseisen

verkkopalvelun saavutettavuusraporttiin. Laaditut raportit ovat yleensä hyvin informatiivisia ja niissä käydään usein läpi myös itse auditointiprosessia.

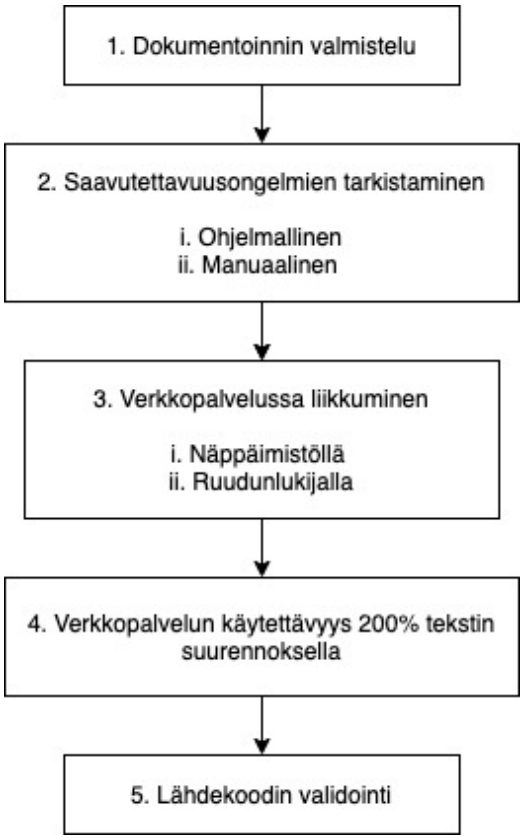
Tämän teoriapohjan kautta olen pystynyt kehittämään toimivan prosessin saavutettavuusauditointeja varten. Teen myös tiivistä yhteistyötä kollegani kanssa, joka on käytettävyysasiantuntija.

9.2 Case-tutkimuksen analysointi

Jarjestohautomo.fi-sivuston saavutettavuusauditoinnin (luku 7) dokumentaatiosta huomaan, että auditointi jakautui karkeasti viiteen osaan. Nämä osat olivat dokumentoinnin valmistelu, saavutettavuusongelmien havaitseminen, verkkosivulla liikkuminen, verkkosivun käyttäminen 200% tekstin suurennoksella ja lähdekoodin validointi.

Näistä osista saavutettavuusongelmien havaitsemisessa ja verkkosivuilla liikkumisessa oli kaksi vaihetta. Saavutettavuusongelmien havaitseminen tehtiin ensin ohjelmallisesti ja manuaalisesti, jonka jälkeen sivu auditoitiin vielä kerran täysin manuaalisesti. Verkkosivuilla liikkuminen testattiin ensin näppäimistöllä ja sen jälkeen ruudunlukijalla.

Dokumentaation avulla pystyin kehittämään kuviossa 18 esitetyn prosessikaavion, joka kuvaa auditointiprosessin rakennetta.



Kuvio 18: Kehitetyn saavutettavuusauditointiprosessin prosessikaavio

9.3 Saavutettavuusauditointiprosessi

Prosessin kehittämisen menetelmällä muodostui viisivaiheinen prosessi, joka kattaa saavutettavuuden tärkeimmät aspektit sisällöntuotantoa lukuun ottamatta. Prosessi ei ole staattinen, vaan sitä on tarkoitus kehittää jatkuvasti omien huomioiden sekä kollegoilta saadun palautteen perusteella.

Seuraavissa alaluvuissa on esitelty tarkemmat ohjeet kuviossa 18 esitetyn saavutettavuusauditointiprosessin eri vaiheisiin. Ohjeet ovat osa Agenda Helsingille kehitettyä auditointiprosessia.

9.3.1 Dokumentointi

Google Sheets-ohjelmalla luodaan taulukko, ja alla olevat sarakkeet lisätään taulukkoon:

- Tehty
- Sivu/Elementti
- Saavutettavuusongelma
- Kriteeri (WCAG 2.1)

- Ratkaisu
- Kuvankaappaus
- Merkitty raporttiin

Jokaiselle auditoitavalle sivulle ja elementille tehdään oma rivistö ja se merkitään selvästi. Tämä on tärkeää, jotta on helppo tunnistaa mitä sivua tai elementtiä saavutettavuusongelma koskee.

Mobiiliauditointia varten luodaan identtinen sivu.

9.3.2 Sivun auditointi Axe-saavutettavuustyökalulla

Jokainen sovittu sivu auditoidaan Axe-ohjelmalla. Axen osoittamat saavutettavuusongelmat tarkistetaan ja varmennetaan että ongelma on validi. Mikäli ongelma on validi, se merkitään auditointitaulukkoon. Taulukkoon merkitään myös, että mikä saavutettavuusohjeistuksen kriteeri on kyseessä (esimerkiksi 1.4.3 Kontrasti).

Mikäli saavutettavuusongelmaa on vaikea selittää, tai visuaalisesta esimerkistä voisi olla asiakkaalle hyötyä, taulukkoon voi tallentaa kuvankaappauksen ongelmasta.

Kun ongelma on tunnistettu ja dokumentoitu, on hyvä kirjoittaa selkokielinen ratkaisu saavutettavuusongelmaan. Mikäli ratkaisusta on epäselvyyksiä, ratkaisun voi tarkistaa osoitteesta: https://www.w3.org/TR/WCAG21. Sivustolta löytyy selitys jokaisesta saavutettavuuskriteeristä ratkaisuineen.

Muita hyviä lähteitä on:

- https://developer.mozilla.org/en-US/docs/Web/Accessibility
- https://papunet.net/ (suomenkielinen)
- https://www.accessibility-developer-guide.com/

Kun Axen ongelmalista on käyty läpi, tulee seuraavat kohdat käydä vielä läpi manuaalisesti, sillä Axe ei välttämättä huomaa näitä ongelmia:

- Sivun antamien virheviestien tulee olla selkeitä ja erottua selvästi
- Ennalta nauhoitetuissa videoissa on oltava tekstitys
- Sivujen rakenne tulee olla looginen
- Pelkällä värillä ei saa antaa merkitystä sisällölle, valinnoille, virheille yms.
- Äänitiedosto eivät saa toistua automaattisesti
- Teksti ei saa olla kuvatiedostona
- Mikäli aikarajoitettu toiminnallisuus, käyttäjällä on oltava mahdollisuus pysäyttää laskuri tai muuttaa aikarajan pituutta

- Liikkuvan sisällön tulee olla pysäytettävissä
- Sivulla oltava kuvaava otsikko (<title>)
- Aktiivisen elementin korostus oltava selkeä
- Sivuston pääkielestä poikkeavassa sisällössä oltava oikea lang-attribuutti
- Navigaation on oltava samanlainen läpi sivuston
- Sisältö ei saa vilkkua yli kolme kertaa

9.3.3 Sivuston käyttö näppäimistöllä

Sivuston käytettävyys tarkistetaan sarkainpainikkeella. Sivu kannattaa päivittää ennen testauksen aloittamista, jotta saadaan selville, että mikä elementti on sarkainjärjestyksessä ensimmäinen.

Tarkistettavat asiat:

- Hyppylinkki (Linkki jonka avulla käyttäjä voi siirtyä suoraan sisältöön ohittaen navigaation)
- Sarkainjärjestyksen loogisuus (sarkainpainikkeella liikkuessa kohdistettavien elementtien järjestys)
- Jokaiseen klikattavaan kohtaan on saatava kohdennus
- Avautuvissa elementeissä (esimerkiksi haitarilinkki) oltava Aria-expanded –attribuutti

9.3.4 Ruudunlukijaohjelma

Sivut käydään läpi ruudunlukijalla samaan tapaan kuin käyttäen sarkainpainiketta eli tutkitaan, onko ruudunlukijalla intuitiivista selata ympäri sivua, ja pystyykö sivun toiminnallisia elementtejä käyttämään myös ruudunlukijalla. Mahdolliset ongelmat kirjataan raporttiin. Monet ongelmista saattavat olla samoja kuin näppäimistöllä havaitut, joten on hyvä välttää kirjaamasta samaa ongelmaa kahdesti.

Sivut tulisi tarkistaa vähintään seuraavilla ruudunlukijoilla:

- Apple VoiceOver
- Windows Narrator

9.3.5 Tekstisuurennos

Sivuston käytettävyys ja luettavuus tarkistetaan 200% tekstin suurennoksella. Tämän voi tehdä esimerkiksi Firefox-selaimella (view > zoom > text only) komennolla cmd ja +.

Tarkistettavat asiat:

- Tekstien tulee pysyä niille tarkoitetuissa alueissa
- Kirjaimet eivät saa mennä päällekkäin
- Tekstien tulee olla luettavissa
- Sivupohja ei saa muuttua liikaa
- Sivuston käytettävyys ei saa kärsiä

9.3.6 Lähdekoodin validointi

Sivun lähdekoodi validoidaan osoitteessa: https://validator.w3.org.

Tarkistettavat asiat:

- Sivustolla ei saa olla ylimääräisiä aloitus- ja lopetus-tageja
- Sivuston lähdekoodin semantiikassa ei saa olla vakavia virheitä

10 Yhteenveto

Tämän opinnäytetyön teoriaosuuden aikana selvitettiin verkkopalveluiden toteutuksen teknistä puolta ja sitä, miten se vaikuttaa verkkosaavutettavuuteen. Tutkimuksen aikana selvisi, että saavutettavien verkkopalveluiden rakentaminen ei vaadi kehittäjältä saavutettavuuden erityisosaamista, vaan ennen kaikkea Html-merkkauskielen oikeaoppista, eli semanttista käyttämistä (muutamia poikkeuksia lukuun ottamatta).

Saavutettavuuden teoriaan tutustuminen syvensi osaamistani verkkokehittäjänä ja auttoi tekemään analyyttisempiä päätöksiä opinnäytetyössä tehdyn saavutettavuusauditoinnin yhteydessä.

Opinnäytetyön toiminnallisessa osassa kehitettiin prosessi verkkopalvelujen saavutettavuusauditointeja varten. Prosessista muodostui viisivaiheinen työnkulkua kuvaava helposti seurattava auditointiprosessi, joka kertoo lyhyesti ja tarkasti mitä auditoijan tulee tehdä kunkin vaiheen kohdalla. Prosessi kattaa saavutettavuuden tärkeimmät kriteerit pois lukien sisällöntuotantoon liittyvät kriteerit.

Opinnäytetyössä kehitettyä saavutettavuusauditointiprosessia on jo ehditty käyttää kahden asiakasprojektin auditointiin. Näiden projektien lopputulokset on arvioitu minun sekä käytettävyysasiantuntijan toimesta, ja tulokset ovat olleet lupaavia, eikä puutteita ole tähän mennessä havaittu. Tämä antaa vahvan viitteen siitä, että opinnäytetyön tuloksen validiteetti ja reliabiliteetti toteutuivat.

Prosessia on kuitenkin tarkoitus kehittää jatkuvasti eteenpäin helpottaakseen saavutettavuusauditointien suorittamista, sekä vastaamaan alati muuttuvan teknologian vaatimuksiin.

Lähteet

Painetut

Selovuo, K. 2019. Saavutettavuusopas. 1. painos. Helsinki: Corellia Helsinki.

Korpela, J.2011. HTML5 – Uudet ominaisuudet. 1. painos. Espoo: WSOYpro.

Sähköiset

Penland, J. 2020. What on earth is semantic markup? Viitattu 4.11.2020. https://html.com/semantic-markup/

W3schools 2020. What is HTML? Viitattu 4.11.2020. https://www.w3schools.com/whatis/whatis_html.asp

Internet World Stats 2020. Viitattu 8.10.2020. https://www.internetworldstats.com/emarketing.htm

European Parliament 2016. Online public services to be made more accessible for the disabled and elderly. Viitattu 16.11.2020. https://www.europarl.europa.eu/news/en/press-room/20161020IPR47872/online-public-services-to-be-made-more-accessible-for-the-disabled-and-elderly?quizBaseUrl=https%3A%2F%2Fquizweb.europarl.europa.eu

W3C 2018. Verkkosisällön saavutettavuusohjeet (WCAG) 2.1. Viitattu 17.10.2020. https://www.w3.org/Translations/WCAG21-fi/

Saavutettavasti 2020. Viitattu 17.10.2020. https://www.saavutettavasti.fi/verkkosisaltojen-saavutettavuus/wcag/

2016/2102. Directive (EU) 2016/2102 of the European Parliament and of the Council of 26 October 2016 on the accessibility of the websites and mobile applications of public sector bodies. Viitattu 20.10.2020. https://eur-lex.europa.eu/eli/dir/2016/2102/oj

306/2019. Laki digitaalisten palvelujen tarjoamisesta. Viitattu 20.10.2020. https://www.finlex.fi/fi/laki/alkup/2019/20190306

Webaim 2019. Screen reader user survey #8 results. Viitattu 16.10.2020. https://webaim.org/projects/screenreadersurvey8/

Webaim 2017. Designing for screen reader compatibility. Viitattu 16.10.2020. https://webaim.org/techniques/screenreader/

MDN 2020. HTML: A good basis for accessibility. Viitattu 16.10.2020. https://developer.mozilla.org/en-US/docs/Learn/Accessibility/HTML

W3Schools 2020. HTML Accessibility. Viitattu 23.10.2020. https://www.w3schools.com/html/html_accessibility.asp

MDN 2020. HTML: A good basis for accessibility. Viitattu 23.11.2020. https://developer.mozilla.org/en-US/docs/Learn/Accessibility/HTML

Webaim 2020. Links and Hypertext. Viitattu 16.11.2020. https://webaim.org/techniques/hypertext/

Papunet 2020. Nimeä linkit ja otsikot ymmärrettävästi. Viitattu 21.10.2020. https://papunet.net/saavutettavuus/nimea-linkit-ja-otsikot-ymmarrettavasti

Jyväskylän yliopisto 2020. Mitä prosessit ovat? Viitattu 28.11.2020. https://www.jyu.fi/laatua/ohjaus/prosessien-mallintaminen/mitaprosessitovat

Html 2020. What on Earth is semantic markup? Viitattu 29.11.2020. https://html.com/semantic-markup/

W3C 2018. Onnistumiskriteeri 1.3.1 Informaatio ja suhteet. Viitattu 29.11.2020. https://www.w3.org/Translations/WCAG21-fi/ - info-and-relationships

MDN 2020. ARIA. Viitattu 30.11.2020. https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA

Itä-Suomen yliopisto 2020. Auditoinnin määritelmä. Viitattu 30.11.2020. http://www3.uef.fi/web/guest/auditointi

JHS 2012. JHS 152 Prosessin kuvaaminen. Viitattu 30.11.2020. http://docs.jhs-suositukset.fi/jhs-suositukset/JHS152/JHS152.html

Agenda Helsinki 2020. Yritysesittely. Viitattu 2.12.2020. https://agendahelsinki.fi/yritys/

W3C 2020. WCAG 3.0 Introduction. Viitattu 2.12.2020. https://www.w3.org/WAI/standards-guidelines/wcag/wcag3-intro/

W3C 2020. Search landmark. Viitattu 3.12.2020. https://www.w3.org/TR/wai-aria-practices-1.1/examples/landmarks/search.html

Webaim 2020. Introduction to ARIA. Viitattu 3.12.2020. https://webaim.org/techniques/aria/

W3C 2020. ARIA14: Using aria-label to provide an invisible label where a visible label cannot be used. Viitattu 3.12.2020 https://www.w3.org/TR/WCAG20-TECHS/ARIA14.html

Webaim 2020. Designing for Screen Reader Compatibility. Viitattu 3.12.2020. https://webaim.org/techniques/screenreader/

ADG 2018. Marking elements expandable using aria-expanded. Viitattu 3.12.2020. https://www.accessibility-developer-guide.com/examples/sensible-aria-usage/expanded/

W3C 2020. WAI-ARIA Overview. Viitattu 3.12.2020. https://www.w3.org/WAI/standards-guidelines/aria/

W3C 2020. Accessible Rich Internet Applications (WAI-ARIA) 1.1 6. Supported States and Properties. Viitattu 3.12.2020. https://www.w3.org/WAI/PF/aria-1.1/states_and_properties

W3C 2020. Button Examples. Viitattu 3.12.2020. https://www.w3.org/TR/wai-aria-practices-1.1/examples/button/button.html

Hostinger 2019. What is HTML? The Basics of Hypertext Markup Language Explained. Viitattu 3.12.2020. https://www.hostinger.com/tutorials/what-is-html

Haapakorpi 2011. Auditointiprosessi ja sen vaikutukset yliopistossa. Viitattu 4.12.2020. https://karvi.fi/app/uploads/2014/09/KKA_0711.pdf

Hiltunen 2009. Validiteetti ja reliabiliteetti. Viitattu 4.12.2020. http://www.mit.jyu.fi/OPE/kurssit/Graduryhma/PDFt/validius_ja_reliabiliteetti.pdf

Kuviot

Taulukot