Tampereen ammattikorkeakoulu

Responsiivinen verkkosovellus

Minna Hannula

OPINNÄYTETYÖ
Toukokuu 2020

Tieto- ja viestintätekniikka
Ohjelmistotekniikka

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tieto- ja viestintätekniikka
Ohjelmistotekniikka

HANNULA, MINNA:
Responsiivinen verkkosovellus

Opinnäytetyö 39 sivua
Toukokuu 2020

Opinnäytetyön aiheena on responsiivisen verkkosovelluksen toteuttaminen. Työssä suunniteltiin ja toteutettiin yksisivuinen verkkosovellus, jonka käyttöliittymä mukautuu erikokoisille laitteille. Tavoitteiden toteutuminen todettiin testaamalla käyttöliittymä erikokoisilla laitteilla sekä mittaamalla verkkosovelluksen suoritusteho.

Työssä suunniteltiin verkkosovellus määrätyn sisällön perusteella. Suunnitelmassa luotiin käyttöliittymästä malli, joka on mahdollisimman helppokäyttöinen. Verkkosovellus toteutettiin tehdyn suunnitelman perusteella. Toteutuksessa noudatettiin suunnitelmaa ja pyrittiin toteuttamaan responsiivisuus niin, että verkkosovelluksen suoritusteho pysyy madollisimman tehokkaana. Toteutuksen jälkeen verkkosovelluksen käyttöliittymä testattiin simuloimalla käyttöliittymää erikokoisilla laitteilla. Suoritustehon testaamisessa käytettiin verkkosovelluksen suoritustehon profilointia ja auditointia. Verkkosovelluksen testit suoritettiin käyttämällä selaimen kehittäjän työkalua.

Testituloksista ilmenee, että toteutettu responsiivinen verkkosovellus toimii suunnitellulla tavalla ja täyttää tavoitteet. Käyttöliittymätestien tuloksista nähtiin, että verkkosovelluksen käyttöliittymä mukautuu pienille, keskisuurille ja suurille näytöille. Suoritustehon testeissä puolestaan havaittiin, että responsiivinen toteuttaminen ei heikennä verkkosovelluksen suoritustehoa mobiililaitteilla. Tietokoneilla puolestaan responsiivinen toteutus aiheuttaa pientä heikentymistä suoritustehossa.

Verkkosovelluksen jatkokehityksessä voitaisiin ottaa huomioon suoritustehon parantaminen. Verkkosovelluksen suoritustehoa on mahdollista parantaa tietokoneella korjaamalla testituloksista löydetyt suoritustehoa heikentävät ominaisuudet. Suoritustehoa on mahdollista parantaa myös pienien ja keskisuurien näyttöjen käyttöliittymissä.

Asiasanat: responsiivisuus, verkkosovellus, CSS, HTML, Angular

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
ICT Engineering
Software Engineering

HANNULA, MINNA:
Responsive Web Site

Bachelor's thesis 39 pages
May 2020

The purpose of this thesis was to create a responsive web design. The goal of the work was to design and develop a single page web application with user interface that adapts to different screen sizes. A user interface was simulated on different devices and the performance was tested with browser developer tools.

In the work, a web application was designed based on specific content. A user interface was designed and developed to improve the performance of the web application. The user interface was tested by simulating the web application on different screen sizes. The performance was also tested with browser developer tools using performance profiler and auditions.

Test results showed that the user interface of the web application worked as planned. The user interface adapts to small, medium and large screens. The performance tests did not reveal any decrease in the performance on the mobile devices. On desktops, however, responsivity causes a slight decrease in performance.

In further development the performance of the web application could be improved. The performance on desktops could be improved by fixing the failed test results. The performance could also be improved on mobile devices.

Key words: responsivity, web site, CSS, HTML, Angular

SISÄLLYS

LYHENTEET JA TERMIT

CSS	Cascading Style Sheets, tyyliohje
HTML	HyperText Markup Language, kuvauskieli
JS	JavaScript, dynaaminen komentosarjakieli

1 JOHDANTO

Tässä opinnäytetyössä tutkitaan responsiivisen verkkosovelluksen toteuttamista. Opinnäytetyössä suunnitellaan yksisivuinen verkkosovellus, jonka käyttöliittymä ja toiminnot mukautuvat erikokoisille näytöille niin, että käyttökokemus on aina mahdollisimman hyvä. Verkkosovellus toteutetaan tehdyn suunnitelman mukaisesti Angular-kehysympäristöä käyttämällä. Verkkosovelluksen repsonsiivisuus toteutetaan CSS-tyyliohjeilla ja -luokilla.

Mobiililaitteiden kehityksen ja yleistymisen myötä internetin käyttö ja käyttökokemus on muuttunut. Verkkosovelluksen tulee nykyään olla käytettävyydeltään ja ulkoasultaan kaikille laitteille soveltuva, jotta verkkosovelluksella on käyttäjiä. Suunnittelijat ja kehittäjät kohtaavat kuitenkin haasteita responsiivisen verkkosovelluksen toteuttamisessa. On olemassa monia tapoja toteuttaa responsiivinen verkkosovellus, mutta kaikki tavat pohjautuvat kuitenkin CSS-tyyliohjeisiin ja -luokkiin. Tässä opinnäytetyössä käsitellään, miten responsiivisuus tulee ottaa huomioon verkkosovelluksen toteuttamisessa.

Verkkosovelluksen responsiivisuus on ollut jo pitkään yksi osa-alue verkkosovellusten suunnittelussa ja toteutuksessa. Responsiivisuudesta on paljon kirjallisuutta, artikkeleita sekä lopputöitä. Teknologia kehittyy jatkuvasti, ja myös verkkosovellusten responsiivisuus ja sen toteuttaminen muuttuvat. Responsiivisuuden ydin ei ole vuosikymmenien aikana muuttunut, mutta vuosi sitten käytetty toteutustapa voi olla jo vanha, jonka vuoksi on tärkeää, että verkkosovellusten kehittäjät päivittävät jatkuvasti osaamistaan.

Työn tavoitteena on toteuttaa toimiva yksisivuinen verkkosovellus, jota voi käyttää erikokoisilla laitteilla. Toteutettavassa verkkosovelluksessa tulee olla yleisesti käytössä olevia elementtejä sekä komponentteja. Lisäksi verkkosovelluksen toteutuksessa tulee ottaa huomioon responsiivisen kehityksen modernit toteutustavat. Tavoitteena on tehdä suoritusteholtaan mahdollisimman tehokas ja käyttäjäystävällinen verkkosovellus.

2 RESPONSIIVISUUS

2.1 Mitä responsiivisella verkkosovelluksella tarkoitetaan?

Kaksi vuosikymmentä sitten verkkosovellusten suunnittelu ja kehittäminen oli suoraviivaista ja yksinkertaista, internetin käyttö ei ollut yleistä ja käytössä olevien laitteiden näytön koot olivat lähes standardit. Laitekehityksen myötä verkkosovellusten haasteeksi tulivat verkkosovellusten sovittaminen erikokoisille näytöille. Alkuun ongelma ratkaistiin luomalla erikokoisten näyttöjen verkkosovellukset erillisille verkko-osoitteille. Tämä tapa oli kuitenkin työläs, ja riskinä oli, että osa verkkosovellusten sisällöstä jäi puuttumaan joltain verkkosovelluksen toteutukselta. Tämän vuoksi kehitettiin responsiivinen tapa toteuttaa verkkosovellus, jolloin verkkosovellus mukautuu erikokoisille näytöille käyttäen vain yhtä verkko-osoitetta. (Carver 2015, 4-5.)

Mobiililaitteiden käyttö on kasvanut viime vuosien aikana paljon, ja ennusteiden mukaan käyttö tulee kasvamaan vielä entisestään. Älypuhelimien käyttäjiä ennustetaan olevan maailmanlaajuisesti vuonna 2020 3,5 miljardia (Gu, 2019). Jatkuvasti kasvavan käytön vuoksi yhä useampaa verkkosovellusta käytetään erikokoisilla laitteilla. Responsiivisen verkkosovelluksen merkitys kasvaa, sillä jokaiselle käyttäjälle, näytön koosta riippumatta, tulisi tarjota sama sisältö ja käyttökokemus verkkosovellusta käytettäessä.

Parantuneen käyttökokemuksen lisäksi responsiivisen verkkosovelluksen hyötyjä ovat mm. taloudelliset säästöt sekä hakukoneoptimointi. Kustannustehokkuus tulee esiin responsiivisen verkkosovelluksen suunnittelussa, kehityksessä ja ylläpidossa. Verkkosovelluksesta ei tarvitse tehdä useita suunnitelmia tai tyyliohjeita, ja verkkosovelluksen kehityksen päätyttyä, ylläpidettävänä on vain yksi verkkosovellus yhdellä verkko-osoitteella. Responsiiviseen verkkosovellukseen on myös helppo liittää sosiaalisen median linkkejä ja hyödyntää mobiililaitteeseen asennettuja sovelluksia. Lisäksi esimerkiksi Google suosittelee priorisoimaan responsiivisuutta, sillä hakukoneen on helpompi käsitellä ja optimoida verkkosovelluksia, joilla on sama sisältö ja HTML-koodi. (Jia 2017.)

Responsiivisen verkkosovelluksen haasteina esiin nousevat mm. selaintuen puute, pitkittynyt latausaika ja pidempi kehitysaika. Selaintuen puute on lähes historiaa, mutta edelleen saattaa olla käytössä vanhoja selaimia, jotka eivät tue responsiivisia verkkosovelluksia. Pitkittynyt latausaika johtuu siitä, että verkkosovellus lataa käyttöliittymälle tarpeetonta sisältöä. Esimerkiksi vain tietyllä käyttöliittymällä näkyvät kuvat heikentävät responsiivisen verkkosovelluksen suoritustehoa ja aiheuttavat haasteita kehitystyössä. Tämän ja muiden haasteiden vuoksi responsiivisen verkkosovelluksen kehitys kestää kauemmin kuin yhdelle näytön koolle määritetyn verkkosovelluksen kehitys. (Jia 2017.)

2.2 Mobile first -periaate

Mobile first -periaatteella tarkoitetaan, että ensimmäisenä suunnitellaan ja toteutetaan verkkosovelluksen mobiililaitteen käyttöliittymä. Periaate pohjautuu mobiililaitteiden käytön kasvuun, verkkosovelluksen sisällön tarkkaan suunnitteluun ja käyttökokemuksen laajentamiseen mobiililaitteilla. (Kadlec 2013, 160-161.) Mobile first -periaate on vakiinnuttanut paikkansa verkkosovellusten suunnittelussa. Aikaisemmin periaatetta voitiin pitää vaihtoehtoisena tapana suunnitella ja toteuttaa verkkosovellus, mutta nykyään se on jo vakiintunut lähtökohta verkkosovellusten suunnittelussa.

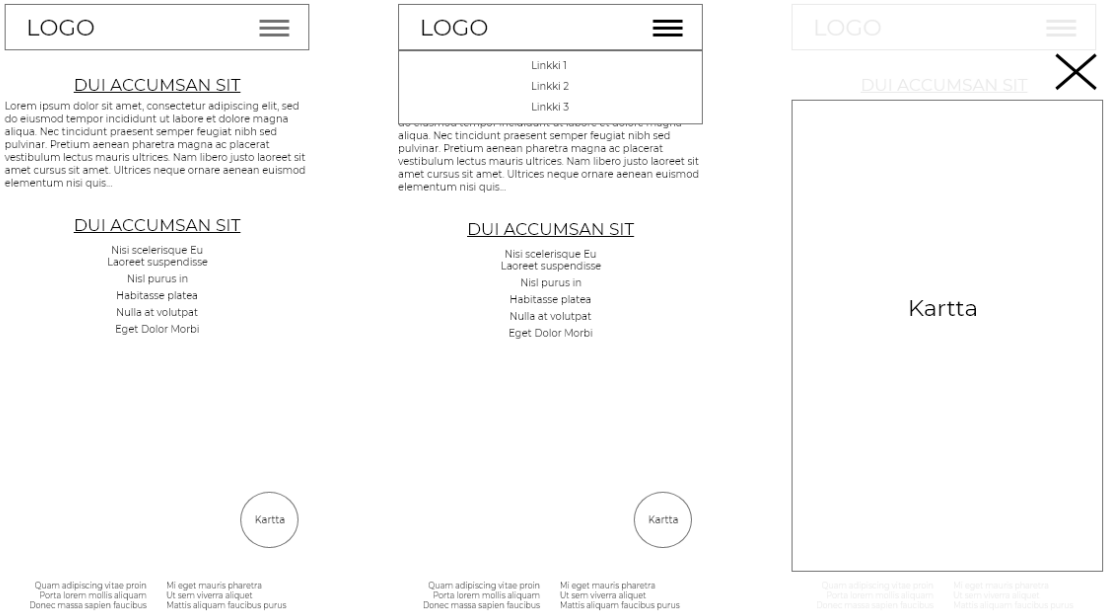
Mobile first -periaate auttaa suunnittelijaa priorisoimaan sisällön tärkeyden ja miettimään, miten mobiililaitteita kannattaa verkkosovelluksessa hyödyntää. Pienille näytöille on hankalaa saada paljon tietoa näkyviin niin, että käyttöliittymä on käyttäjäystävällinen ja selkeä. Kun sisältöä joudutaan miettimään ja priorisoimaan pienille näytöille, käyttöliittymään ei lisätä turhaa sisältöä suuremmille näytöille, jolloin verkkosovellus pysyy siistinä ja selkeänä. Lisäksi mobile first -periaatteen avulla voidaan hyödyntää mobiililaitteen erilaisia ominaisuuksia käyttökokemuksen lisäämiseksi. Esimerkiksi älypuhelimen antureita voidaan käyttää sijainnin paikantamiseen, mobiililaitteeseen ladattuja sovelluksia voidaan hyödyntää verkkosovelluksen sisällön jakamiseen ja verkkosovelluksesta voidaan tehdä interaktiivisempi kosketusnäytön ominaisuuksia hyödyntämällä. (Kadlec 2013, 162-164.)

3 RESPONSIIVISEN VERKKOSOVELLUKSEN TOTEUTTAMINEN

3.1 Verkkosovelluksen suunnittelu

Työssä tehdään verkkosovellus yritykselle. Verkkosovelluksen tulee sisältää yrityksen logo ja kuvaus, tiedot palveluista sekä kartta, jossa näytetään yrityksen sijainti. Sisältövaatimusten perusteella on aloitettu responsiivisen verkkosovelluksen suunnitelman toteutus.

Ensimmäisenä verkkosovelluksen suunnittelussa priorisoidaan verkkosovelluksen sisältö ja suunnitellaan, miten elementit sijoitellaan verkkosovellukseen. Verkkosovelluksen sisältö ja toiminnot vaihtelevat näytön koon mukaan. Kuvassa 1 esitetään pienien näyttöjen käyttöliittymän suunnitelma.



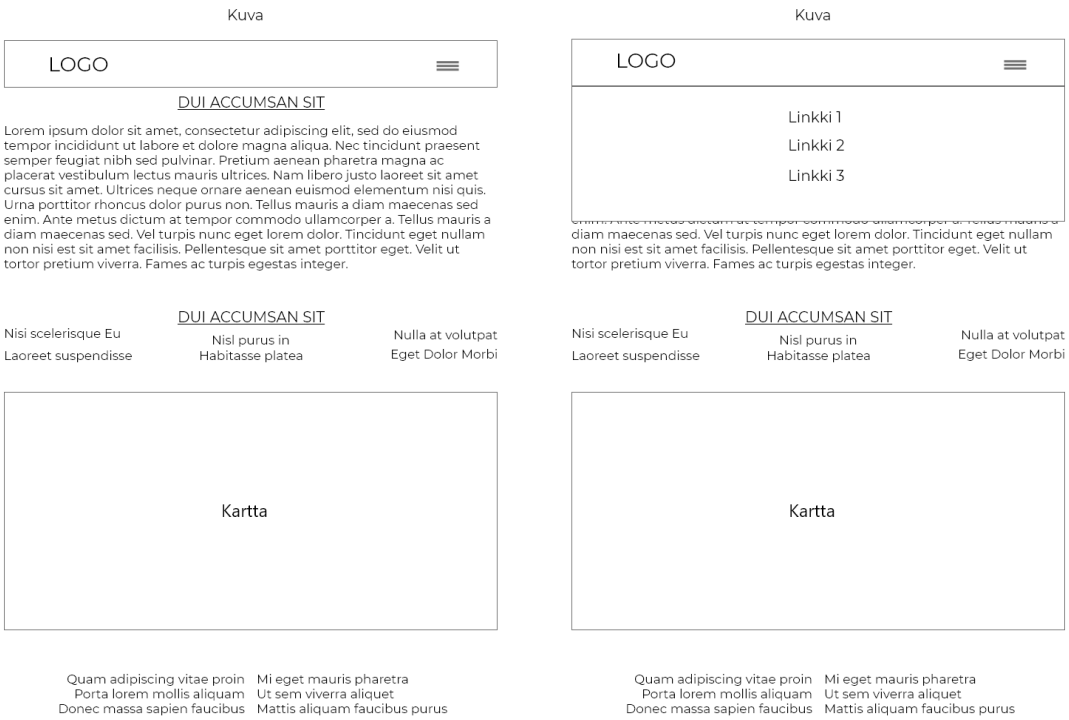
KUVA 1. Verkkosovelluksen käyttöliittymä pienillä näytöillä

Pienillä näytöillä verkkosovelluksessa on navigaatiopalkki, pudotusvalikko, tekstisisältö, luettelo, kartta sekä alatunniste. Elementit sijoittuvat näyttöalueelle päällekkäin. Verkkosovelluksessa on painikkeita, joilla voidaan avata ja sulkea verkkosovelluksen elementtejä.

Käytettävyyden parantamiseksi, pienillä näytöillä navigaatiolinkit ovat sijoitettu pudotusvalikkoon, jolloin navigaatiopalkki on siisti ja verkkosovelluksen navigaatioon saadaan lisättyä useampia navigaatiolinkkejä. Pudotusvalikko voidaan avata ja sulkea navigaatiopalkissa olevalla painikkeella. Pudotusvalikko peittää osan verkkosovelluksen sisällöstä, jolloin sisältö pysyy paikallaan pudotusvalikon auetessa.

Verkkosovelluksen kartta on sijoitettu erilliseen elementtiin, joka avataan sille tarkoitetulla painikkeella. Sisältö on näyttöalueen suuruinen, jolloin käyttäjän on helppo lukea karttaa. Kartan sisältävä elementti voidaan sulkea elementissä olevalla painikkeella.

Verkkosovellukselle toteutetaan oma käyttöliittymä keskisuurille näytöille. Keskisuurilla näytöillä elementtien sijoittelu, verkkosovelluksen sisältö ja toiminnot poikkeavat pienien näyttöjen käyttöliittymästä. Kuvassa 2 esitetään verkkosovelluksen käyttöliittymän suunnitelma keskisuurille näytöille.



KUVA 2. Verkkosovelluksen käyttöliittymä keskisuurilla näytöillä

Keskisuurilla näytöillä verkkosovelluksessa on ylätunniste, navigaatiopalkki, pudotusvalikko, tekstisisältö, luettelo, kartta sekä alatunniste. Elementit sijoittuvat

näyttöalueelle päällekkäin. Verkkosovelluksen käyttöliittymään sijoitetaan painike navigaatiopalkkiin.

Keskisuurilla näytöillä navigaatiolinkit sijoitetaan pudotusvalikkoon käytettävyyden parantamiseksi. Pudotusvalikon toiminta vastaa pienien näyttöjen käyttöliittymän pudotusvalikon toimintaa. Käytettävyyden parantamiseksi myös keskisuurilla näytöillä pudotusvalikko peittää osan sisällöstä.

Pienien näyttöjen toiminnoista poiketen keskisuurilla näytöillä kartta on esillä käyttöliittymässä koko ajan. Kartta sijoitetaan luettelon alapuolelle, ja sen on yhtä leveä kuin verkkosovelluksen muu sisältö. Kartasta pyritään tekemään mahdollisimman suuri, jolloin se on helposti luettava.

Viimeisenä verkkosovellukselle suunnitellaan suurien näyttöjen käyttöliittymä. Suurien näyttöjen käyttöliittymä ja sisältö poikkeavat pienille ja keskisuurille näytöille suunnitelluista käyttöliittymistä. Kuvassa 3 esitetään verkkosovelluksen käyttöliittymä suurilla näytöillä.

KUVA 3. Verkkosovelluksen käyttöliittymä suurilla näytöillä

Suurilla näytöillä verkkosovelluksessa on ylätunniste, navigaatiopalkki, tekstisisältö, luettelo, kartta sekä alatunniste. Elementit sijoitetaan näkymään

allekkain ja rinnakkain. Suurilla näytöillä navigaatiopalkki sisältää logon ja navigaatiolinkit.
Suurilla näytöillä ei ole tarvetta sijoittaa navigaatiolinkkejä pudotusvalikkoon, sillä navigaatiopalkki on lähes yhtä leveä kuin näyttö. Näin ollen navigaatiopalkkiin saadaan sijoitettua useampia navigaatiolinkkejä. Tämä myös parantaa käytettävyyttä suurilla näytöillä, sillä käyttäjä pystyy helposti ja nopeasti navigoimaan verkkosovelluksessa.

Myös suurilla näytöillä verkkosovelluksen kartta on koko ajan käyttöliittymässä näkyvissä. Kartta on sijoitettu luettelon viereen. Tällä pyritään tiivistämään verkkosovelluksen sisällön kokoa niin, että sisältö mahtuisi kokonaisuudessaan näyttöalueelle. Näin käyttäjän on helpompi löytää etsimänsä tieto verkkosovelluksesta, jolloin verkkosovelluksen käytettävyys paranee.

Verkkosovellus toteutetaan yksisivuisena, jolloin navigaatiolinkit ohjaavat käyttäjän yrityksen sosiaalisen median kanaviin. Vaihtoehtoisesti navigaatiolinkit voisivat ohjata käyttäjän verkkosovelluksen eri elementteihin. Verkkosovelluksen sisältö ei vaadi useamman sivun verkkosovelluksen toteutusta, jonka vuoksi useamman sivun verkkosovelluksesta koituisi vain ylimääräistä työtä ja kustannuksia. Lisäksi yksisivuisella verkkosovelluksella on tehokkaampaa tarjota käyttäjälle etsittävä tieto.

3.2 Verkkosovelluksen kehitys

Verkkosovelluksen toteutuksen seuraava vaihe on kehitys. Kehityksessä verkkosovellus toteutetaan Angular-kehysympäristöllä ja CSS-tyyliohjeella vastaamaan tehtyä suunnitelmaa. Verkkosovelluksen kehityksessä jatketaan mobile first -periaatetta, eli ensin luodaan verkkosovellus pienille näytöille, jonka jälkeen se muokataan vastaamaan keskisuurien ja suurien näyttöjen suunnitelmaa.

3.2.1 Verkkosovelluksen media

Verkkosovelluksessa ylätunnisteen kuva on toteutettu suunnitelman mukaisesti, pienien näyttöjen käyttöliittymässä ei ole ylätunnistetta tai kuvaa. Suurempien näyttöjen käyttöliittymissä ylätunniste ja sen sisältämä kuva on sijoitettu näytön yläreunaan navigaatiopalkin yläpuolelle. Ylätunnisteen kehityksessä tulee huomioida verkkosovelluksen suoritusteho erikokoisilla laitteilla. Pienien näyttöjen käyttöliittymässä ei ole ylätunnistetta, jolloin kuvaa ei tarvitse ladata verkkosovellukseen. Keskisuurien näyttöjen käyttöliittymään puolestaan voidaan ladata pienempi kuva kuin suurien näyttöjen käyttöliittymään.

Verkkosovelluksen ylätunniste on toteutettu käyttämällä picture-elementtiä, joka sisältää kolme elementtiä, joissa määritetään erikokoisille laitteille ladattava kuva. Picture-elementin sisällä on kaksi source-elementtiä, joissa määritetään ladattava kuva sekä luodaan ehto, milloin kuva ladataan. Lisäksi picture-elementissä on img-elementti, jossa määritetään ladattava kuva, mikäli kummankaan source-elementin ehto ei täyty. Kuvassa 4 esitetään ylätunnisteen kuvan lataus.

```
<!-- Picture elementti, joka toimii ylätunnisteena -->
<picture class="picture">
    <!-- Source elementti, määritettän ladattava kuva kun media attribuutin ehto täyttyy -->
    <source media="(min-width: 825px) and (min-height: 570px)" srcset="assets/images/Kuva_suuri.png 1x">
    <source media="(min-width: 570px) and (max-width: 825px) and (min-height: 570px)" srcset="assets/images/Kuva_keskisuuri.png 1x">
    <!--Img elementti, määritetään ladattava kuva mikäli source elementtien ehdot eivät täyty-->
    <img class="image" src="//:0" alt="Header image">
</picture>
<!-- Kutsutaan navigation komponenttia -->
<app-navigation></app-navigation>
```

KUVA 4. Ylätunnisteen kuvan lataus

Kuvasta 4 nähdään, että ylätunnisteen kuvaksi on valittu kaksi eri kuvaa, Kuva_suuri.png ja Kuva_keskisuuri.png. Ensimmäisessä source-elementissä määritetään media-attribuutilla, että Kuva_suuri.png ladataan vain, jos näytön leveys on vähintään 825 px ja korkeus on vähintään 570 px. Toisessa source-elementissä määritetään, että Kuva_keskisuuri.png ladataan vain, jos näytön leveys on vähintään 570 px ja korkeintaan 825 px, ja näytön korkeus on vähintään 570 px. Mikäli kumpikaan ehto ei täyty, ladataan img-elementin src-attribuutissa määritetty kuva. Attribuutissa on kuitenkin ladattavan kuvan polku asetettu tyhjäksi, jolloin mitään kuvaa ei ladata.

Source-elementeille on luotu useampi ehto, joiden kaikkien tulee täyttyä, jotta kuva ladataan. Useammalla ehdolla voidaan varmistaa, että ylätunnisteen kuvaa ei ladata pienien näyttöjen käyttöliittymään näytön suunnasta riippumatta ja, että suurille näytöille ei ladata ylätunnisteeseen kahta kuva. Mikäli ehtona olisi määritetty vain näytön leveys, kuva ladattaisiin myös pienille näytöille, kun näyttö olisi käännettynä vaakatasoon. Lisäksi suurille näytöille olisi ladattu kaksi kuvaa, mikäli Kuva_keskisuuri.png lataamisen ehtona ei olisi määritetty näytön maksimileveyttä.

Elementtien avulla voidaan määrittää, mikä kuva ladataan, mutta käyttöliittymän responsiivisuuden takaamiseksi ylätunnisteeseen on lisätty kaksi tyyliluokkaa. Header-tyyliluokassa määritetään picture-elementin näkyvyys erikokoisilla näytöillä. Image-luokassa puolestaan määritetään kuvan leveys. Tyyliluokat esitetään kuvassa 5.

```
/* Picture elementti */
.picture {
    /* Elementti piilotetaan käyttöliittymästä */
    display: none;
}
/* Näytön leveyden ja korkeuden ollessa vähintään 570px */
@media screen and (min-width: 570px) and (min-height: 570px) {
    .picture {
        /* Elementti muutetaan block elementiksi, joka näkyy käyttöliittymässä  */
        display: block;
        /* Elementin leveys */
        width: 100%;
    }
    /* Img elementti */
    .image {
        /* Kuvan leveys */
        width: 100%;
    }
}
```

KUVA 5. Ylätunnisteen ja kuvan tyyliluokat

Tyyliluokassa ei enää tarvitse huomioida kuvan lataamista, mutta tyyliluokassa kiinnitetään huomiota ylätunnisteen näkyvyyteen ja kuvan leveyteen. Elementti tulee piilottaa pienillä näytöillä, sillä img-elementissä on hakukoneoptimoinnin parantamiseksi määritetty alt-attribuutilla kuvan vaihtoehtoinen teksti. Vaikka pienien näyttöjen käyttöliittymään ei ladata kuvaa, alt-attribuutin arvo tulostuu käyttöliittymään. Elementin piilottamisen avulla myös vaihtoehtoinen teksti piilotetaan käyttöliittymästä.

Elementin piilottaminen toteutetaan tyyliohjeen avulla display-toiminnolla. Elementti pysyy piilotettuna, kunnes display-toiminnolle asetetaan toinen arvo. Toiminnon arvo muutetaan käyttämällä media query -sääntöä.

Media query -sääntö tai mediakysely on CSS-tyyliohjeen sääntö, jolla voidaan selvittää käytettävän laitteen ominaisuuksia. Kysely palauttaa arvon true, mikäli käytettävä laite toteuttaa kyselyn lausekkeen. Vain jos kyselyn palauttama arvo on true, toteutetaan kyselyn sisältämät tyyliluokat, muutoin tyyliluokat jäävät toteuttamatta. Mediakysely rakentuu valinnaisista mediatyypeistä ja mediaominaisuuksien lausekkeista. Kyselyjä voidaan yhdistää loogisten operaattorien avulla, jolloin kyselystä saadaan tarkempia ja ne voidaan kohdentaa tietyille laitteille. (MDN 2008.)

Ylätunnisteen kuvan näkyvyydessä käytetyssä mediakyselyssä on kolme lauseketta. Ensimmäisessä lausekkeessa on määritetty mediatyyppi, toisessa ja kolmannessa lausekkeessa on määritetty mediaominaisuus. Mediakysely palauttaa arvon true, mikäli käytetty mediatyyppi on kuvaruutu sekä kuvaruudun leveys ja korkeus ovat vähintään 570 px. Mikäli palautettava arvo on true, picture-elementin näkyvyys muutetaan niin, että elementti on näkyvillä. Muulloin elementti piilotetaan.

Lisäksi mediakyselyn toteutuessa määritetään ylätunnisteelle ja kuvalle leveys. Molempien elementtien leveys on määritetty joustavaksi ja koko näytön levyiseksi. Elementin joustavuus luodaan asettamalle width-toiminnon arvoksi prosenttiluku. Tällöin elementin leveys suhteutetaan laitteen näytön leveyteen. Siispä arvon ollessa 100 %, elementti on yhtä leveä kuin näyttö.

3.2.2 Verkkosovelluksen navigaatio ja logo

Verkkosovelluksessa toteutetaan kaksi erilaista navigaatiota, pienillä ja keskisuurilla näytöillä navigaatiolinkit sijoitetaan pudotusvalikkoon ja suurilla näytöillä navigaatiolinkit ovat navigaatiopalkissa. Lisäksi navigaatiopalkin sijainti näytöllä riippuu näytön koosta. Pienillä näytöillä navigaatiopalkki pysyy aina

näytön yläreunassa, kun taas keskisuurilla ja suurilla näytöillä navigaatiopalkki on ylätunnisteen kuvan alla, mikäli ylätunnisteen kuva näkyy kuvaruudulla. Muutoin navigaatiopalkki on näytön yläreunassa.

Pienillä ja keskisuurilla näytöillä pudotusvalikko aukaistaan navigaatiopalkin painikkeella. Painike ja pudotusvalikko tulevat näkyviin vain pienillä ja keskisuurilla näytöillä, jonka vuoksi komponenttien näkyvyydessä on käytetty CSS-tyyliohjeen mediakyselyä. Kuvassa 6 esitetään painikkeen ja pudotusvalikon tyyliluokka navigaatiopalkissa.

```
/* Navigaatiopalkin painike */
.navigationBtn {
    /* Marginaali */
    margin-right: 5%;
    /* Tekstin sijainti */
    text-align: right;
}
/* Pudotusvalikko */
.dropdown {
    /* Pudotusvalikon sijainti */
    position: fixed;
    left: 0;
    /* Marginaali */
    margin-top: 20px;
    /* Leveys */
    width: 100%;
    /* Elementin järjestys pinossa */
    z-index: 1;
    /* Tekstin sijainti */
    text-align: center;
    /* Tyyli */
    background-color: #e1e1e1;
}
/* Näytön leveyden ollessa vähintään 820px */
@media screen and (min-width: 820px) {
    .navigationBtn, .dropdown{
        /* Piilotetaan painike ja pudotusvalikko */
        display: none;
    }
}
```

KUVA 6. Navigaatiopalkin painikkeen ja pudotusvalikon tyyliluokat

Painikkeen ja pudotusvalikon tyyliluokissa käytetään määrättyä ja joustavaa ulkoasua. Painikkeen oikean reunan marginaali on toteutettu joustavasti antamalla ominaisuudelle arvoksi prosenttiluku. Tällöin marginaali mukautuu jokaisen laitteen näytön koon perusteella. Pudotusvalikossa on puolestaan käytetty määrättyä ulkoasua. Pudotusvalikon yläreunan marginaali määritetään pikselisuureella, jolloin marginaali on jokaisella näytöllä koosta riippumatta 20 px. Pudotusvalikon leveys on puolestaan joustava ja aina näytön leveyden

suuruinen. Mediakyselyllä on määritetty, että näytön leveyden ollessa vähintään 820 px, painike ja pudotusvalikko piilotetaan käyttämällä display-toimintoa ja asettamalla toiminnon arvoksi none.

Navigaatiolinkit mukautuvat myös responsiivisesti näytön kokoon. Pudotusvalikossa linkit ovat listattuna päällekkäin, kun taas navigaatiopalkissa linkit on listattu riviin. Navigaatiolinkkien responsiivisuus ja muotoilu on toteutettu kahdessa komponentissa. Navigation-komponentissa, jossa luodaan verkkosovelluksen navigaatio, määritetään navigaatiolinkkien näkyvyys navigaatiopalkissa. Navigation-links-komponentissa, jossa luodaan navigaatiolinkit, määritetään linkkien sijoittaminen ja tyyli. Kuvassa 7 esitetään tyyliluokka, jossa määritetään navigaatiolinkkien näkyvyys navigaatiopalkissa. Kuvassa 8 puolestaan esitetään tyyliluokat, jossa määritetään linkkien sijoittaminen ja tyyli.

```
/* Navigaatiolinkit */
.navigationLink {
    /* Piilotetaan linkit navigaatiopalkista */
    display: none;
}
/* Näytön leveyden ollessa vähintään 820px */
@media screen and (min-width: 820px) {
    .navigationLink{
        /* Elementti muutetaan block elementiksi, joka näkyy käyttöliittymässä */
        display: block;
        /* Marginaali */
        margin: auto 0;
    }
}
```

KUVA 7. Navigaatiolinkkien näkyvyys navigaatiopalkissa

```
/* Lista linkeistä */
.linkList {
    /* Elementin täyte*/
    padding: 0;
    /* Tyyli */
    list-style: none;
}
/* Linkki */
.link {
    /* Elementin täyte */
    padding: 2%;
}
/* Linkin teksti */
.link a {
    /* Tyyli */
    text-decoration: none;
    color: ■#000000;
}
/* Näytön leveyden ollessa vähintään 820px */
@media screen and (min-width: 820px){
    .linkList {
        /* Tekstin sijainti */
        text-align: right;
    }
    .link {
        /* Linkit näytetään rivissä */
        display: inline;
        /* Marginaali */
        margin-right: 2%;
    }
    .link a {
        /* Elementin horisontaalinen ja vertikaalinen täyte */
        padding: 1% 2%;
        /* Tyyli */
        border-bottom: 1.5px solid ■#000000;
    }
}
```

KUVA 8. Navigaatiolinkkien sijoittaminen ja tyyli

Navigaatiolinkkien näkyvyys määritetään display-toiminnon avulla. Pienillä ja keskisuurilla näytöillä navigaatiolinkit piilotetaan näkyvistä, mutta laitteen näytön leveyden ollessa vähintään 820 px, muutetaan elementti block-tyyppiseksi elementiksi, joka on käyttöliittymässä esillä. Navigaatiolinkkien näkyvyyden yhteydessä määritetään elementin vertikaalinen ja horisontaalinen marginaali.

Navigaatiolinkkien sijoittamisessa käytetään määrättyä ja joustavaa ulkoasua. Navigaatiolinkkien listan sisäinen täyte on poistettu asettamalla padding-toiminnolle määrätty arvo 0. Yksittäisen navigaatiolinkin marginaali ja sisäinen täyte on kuitenkin määritetty joustavaksi antamalla toiminnoille arvoksi prosenttiluvut. Arvoja sekä linkkien sijoittelua listassa muutetaan mediakyselyn avulla. Yli 820 px levyisillä näytöillä lisätään linkille marginaali sekä täyte, ja

muutetaan listan esitystapa display-toiminnolla. Lisäksi linkkien tyyliin lisätään suunnitelman mukaisesti alareunan rajaus border-bottom-toiminnolla.

Navigaatiopalkkiin on sijoitettu bittikarttakuvana logo. Jotta logo on graafisesti miellyttävä ja asettuu navigaatiopalkkiin näytön koosta riippumatta, on logon koko määritetty tyyliluokan avulla. Kuvassa 9 esitetään logon tyyliluokka.

```
/* Logo */
.logo {
    /* Marginaali */
    margin-left: 1%;
    /* Logon leveys */
    width: 120px;
}
/* Näytön leveyden ollessa vähintään 450px */
@media screen and (min-width: 450px) {
    .logo{
        /* Logon leveys */
        width: 130px;
    }
}
```

KUVA 9. Logon tyyliluokka

Logoon on käytetty määrättyä ulkoasua, logon koko on määrätty pikselisuureella ja kuva kasvaa vain mediakyselyllä, kun näytön leveys on vähintään 450 px. Tässä työssä logo on haluttu pitää myös suurilla näytöillä pienenä, mutta haluttaessa logoa olisi voitu vielä suurentaa. Logon leveys on kannattavinta toteuttaa määrätyllä suureella, jolloin logo on parhaiten hallittavissa. Mikäli logon leveys olisi joustava, mediakyselyitä tulisi tehdä enemmän, jotta logo ei suurene tai pienene liikaa. Logon marginaali on toteutettu joustavasti, jolloin sen sijainti mukautuu näytön kokoon.

Navigaatiopalkki on tässä työssä toteutettu kahdella elementillä. Ensimmäisellä elementillä määritetään navigaatiopalkin sijainti näytöllä. Toisella elementillä määritetään navigaatiopalkin tyylittely sekä koko ja luodaan navigaatiopalkin sisällä oleville elementeille paikat. Kuvassa 10 esitetään tyyliluokat, jotka luovat navigaatiopalkin kokonaisuuden.

```
/* Navigaatiopalkin paikka */
.navigationContainer {
    /* Navigaatiopalkin sijainti */
    position: fixed;
    top: 0;
    left: 0;
    /* Navigaatiopalkin leveys ja korkeus */
    width: 100%;
    height: 90px;
    /* Elementin järjestys pinossa */
    z-index: 1;
}
/* Navigaatiopalkki */
.navigationBar {
    /* Navigaatiopalkin sisällön esitystapa */
    display: grid;
    grid-template-columns: 35% 65%;
    align-items: center;
    /* Navigaatiopalkin marginaali ja  keskitys */
    margin-top:5px;
    margin-left: 50%;
    transform: translateX(-50%);
    /* Navigaatiopalkin leveys ja korkeus */
    width: 90%;
    height: 80px;
    /* Elementin järjestys pinossa */
    z-index: 1;
    /* Tyyli */
    background-color: #e8e8e8;
}
```

```
/* Näytön leveyden ollessa vähintään 570px */
@media screen and (min-width:570px) {
    .navigationContainer{
        /* Muutetaan navigaatiopalkin sijaintia */
        position: sticky;
    }
}
/* Näytön leveyden ollessa vähintään 820px */
@media screen and (min-width: 820px) {
    .navigationBar{
        /* Muutetaan navigaatiopalkin sisällön sijaintia */
        grid-template-columns: 40% 60%;
    }
}
```

KUVA 10. Navigaatiopalkin tyyliluokat

Navigaatiopalkin paikan määrittämisessä on käytetty joustavaa ja määrättyä ulkoasua. Sijainti on pienillä näytöillä asetettu määrätyksi käyttämällä position-toiminnossa arvoa fixed. Navigaatiopalkin paikka saadaan pysymään näytön yläreunassa, kun top-toiminnon arvoksi asetetaan 0. Left-toiminnolla puolestaan elementti pysyy näytön vasemmassa reunassa. Navigaatiopalkin leveys on määritetty laitteen näytön levyiseksi joustavalla ulkoasulla ja korkeus on asetettu määrätyksi ulkoasuksi asettamalla korkeuden arvoksi 90 px.

Navigaatiopalkin sijainti saadaan keskisuurilla ja suurilla näytöillä muutettua suunnitelman mukaisesti ylätunnisteen alle asettamalla position-toiminnon arvoksi sticky. Toiminnon arvon muuttamisella saadaan navigaatiopalkki sijoitettua ylätunnisteen alle ylätunnisteen näkyessä kuvaruudulla. Muulloin navigaatiopalkki on näytön yläreunassa. Navigaatiopalkin sijaintia muutetaan, kun näytön leveys on vähintään 570 px.

Navigaatiopalkki asetetaan verkkosovelluksessa sisällön päälle toiminnolla z-index. Verkkosovelluksen elementtien voidaan ajatella olevan pinossa, jolloin z-index-toiminnon arvolla voidaan määrittää pinon päällimmäinen elementti. Z-index-toiminnon oletusarvo on 0, joten muuttamalla toiminnon arvoksi 1, saadaan elementtiä nostettua pinossa ylöspäin.

Navigaatiopalkissa sisällölle luodaan paikat joustavalla ruudukolla. Ruudukko luodaan antamalla display-toiminnolle arvoksi grid. Ruudukolle määritetään kaksi kolumnia, joiden leveys riippuu näytön koosta. Pienillä ja keskisuurilla näytöillä vasemmanpuoleisen kolumnin leveys on 35 % ja oikeanpuoleisen kolumnin leveys on 65 %. Suurilla näytöillä kolumnien kokoa muutetaan mediakyselyn avulla. Kolumnien yhteenlaskettu leveys on 100 %, eli ruudukko on yhtä leveä kuin elementti.

Navigaatiopalkin sisältö asetetaan joustavasti näytön keskelle marginaalin ja transform-toiminnon avulla. Vasemman reunan marginaali asetetaan 50 % päähän näytön reunasta ja transform-toiminnolla sijoitetaan elementti horisontaalisesti uudelleen -50 %. Näin elementti saadaan keskitettyä näytölle koosta riippumatta.

3.2.3 Verkkosovelluksen sisältö

Verkkosovelluksen sisältö koostuu kolmesta komponentista: content-text, content-list ja content-map. Sisällön komponentit on sijoitettu verkkosovelluksessa home-komponenttiin, jossa luodaan elementit, johon sisältökomponentit sijoitetaan. Home-komponentissa luodaan myös sisällön otsikot. Kuvassa 11 esitetään home-komponentin tyyliluokat.

```
.homePage {
    /* Vertikaalinen ja horisontaalinen marginaali*/
    margin: 100px auto;
    /* Leveys ja korkeus */
    width: 80%;
    min-height: 100%;
}

.textContent, .listContent, .mapContent, .contentContainer {
    /* Vertikaalinen ja horisontaalinen marginaali*/
    margin: 5% 1%;
}

.headlinePrimary, .headlineSecondary {
    /* Leveys */
    width: 100%;
    /* Tekstin sijainti */
    text-align: center;
    /* Fonttikoko */
    font-size: 110%;
    /* Tyyli */
    font-weight: normal;
    text-transform: uppercase;
    text-decoration: underline;
}
/* Näytön leveyden ollessa vähintään 570px */
@media only screen and (min-width: 570px){
    .homePage {
        /* Vertikaalinen ja horisontaalinen marginaali*/
        margin: 5% auto;
    }
    .textContent, .contentContainer {
        /* Vertikaalinen ja horisontaalinen marginaali*/
        margin: auto 1%;
    }
    .listContent, .mapContent {
        /* Vertikaalinen ja horisontaalinen marginaali*/
        margin: 10% 1%;
    }
}
/* Näytön leveyden ollessa vähintään 825px */
@media only screen and (min-width: 825px){
    .contentContainer {
        /* Sisällön sijoittelu */
        display: grid;
        grid-template-columns: 30% 70%;
        /* Marginaali */
        margin-top: 7%;
    }
    .textContent, .contentContainer {
        /* Vertikaalinen ja horisontaalinen marginaali*/
        margin: 1% 1%;
    }
    .headlineSecondary {
        /* Tekstin sijainti */
        text-align: left;
    }
}
```

KUVA 11. Verkkosovelluksen sisällön sijoittaminen

Pienillä ja keskisuurilla näytöillä sisältö on aseteltu päällekkäin, jolloin se on helpommin luettavaa. Suurilla näytöillä sisällön ensimmäinen osa, tässä tapauksessa tekstisisältö, on yksittäin ja kaksi muuta sisältökohdetta on sijoitettu ruudukkoon (luettelo ja kartta). Myös toisen otsikon sijainti muuttuu suurilla näytöillä, otsikko siirtyy elementin keskeltä oikeaan reunaan. Sisällössä käytetään joustavaa ulkoasua, jolloin se mukautuu näytön kokoon.

Verkkosovelluksen tekstisisältö on toteutettu content-text-komponentissa. Komponentti sisältää vain tekstin ja komponentti sijoitetaan home-komponentissa elementtiin, jonka tyyliluokka on textContent. Kuvassa 12 esitetään content-text-komponentin tyyliluokka.

```
/* Teksti */
.textContent {
    /* Marginaali */
    margin: auto;
    /* Leveys */
    width: 100%;
}
```

KUVA 12. Tekstisisällön tyyliluokka

Tekstisisällön tyyliluokassa määritetään vain elementin leveys sekä marginaali. Elementti on joustava ja sen leveys asetetaan yhtä suureksi kuin home-komponentin textContent-tyyliluokan elementti, johon tekstisisältö sijoitetaan. Elementin marginaalin arvoksi on asetettu auto, jolloin elementti sijoittuu horisontaalisesti ympäröivän elementin sisälle.

Verkkosovellukseen on tehty luettelo, jonka kolumnien määrä muuttuu näytönkoon mukaan. Pienillä ja suurilla näytöillä luettelossa on yksi kolumni, mutta keskisuurilla näytöillä kolumneja on kolme. Lisäksi suurilla näytöillä luettelon teksti asettuu elementin vasempaan reunaan. Kuvassa 13 esitetään luettelon tyyliluokka.

```
/* Lista sisältö */
.list {
    /* Marginaali */
    margin: 0;
    /* Elementin täyte */
    padding: 0;
    /* Tekstin sijainti */
    text-align: center;
    /* Tyyli */
    list-style: none;
}
/* Näytön leveyden ollessa vähintään 570px */
@media only screen and (min-width: 570px) {
    .list {
        /* Listan kolumien marginaali ja kolumnien lukumäärä */
        columns: auto 3;
    }
}
/* Näytön leveyden ollessa vähintään 825 px */
@media only screen and (min-width: 825px) {
    .list {
        /* Listan kolumien marginaali ja kolumnien lukumäärä */
        columns: auto 1;
        /* Tekstin sijainti */
        text-align: left;
    }
}
```

KUVA 13. Luettelon tyyliluokka

Luettelon kolumnien määrää voidaan muokata columns-toiminnon avulla. Columns-toiminnossa määritetään ensin kolumnin leveys ja sen jälkeen kolumnien lukumäärä. Luettelon kolumnien leveydeksi on asetettu arvo auto. Tällöin kolumnit täyttävät elementin ja ovat keskenään samankokoisia. Luettelon marginaalin ja täytteen arvo on määrätty nollaksi, jolloin luettelolla ei ole marginaalia tai täytettä.

Sisällön viimeisenä komponenttina on kartta. Pienillä näytöillä kartta avataan painikkeella, joka on sijoitettu näyttöalueen alareunaan. Keskisuurilla ja suurilla näytöillä kartta on käyttöliittymässä koko ajan esillä home-komponentissa. Kartan avaavan painikkeen tyyliluokka esitetään kuvassa 14.

```
/* Kartan avaava painike */
.button {
    /* Sijainti */
    position: fixed;
    bottom: 3%;
    right: 3%;
    /* Elementin täyte */
    padding: 4%;
     /* Elementin järjestys pinossa */
     z-index: 2;
    /* Tekstin sijainti */
    text-align: center;
    /* Tyyli */
    background-color: #e8e8e8;
    border: none;
    border-radius: 50%;
    color: #000000;
    text-decoration: none;
}
/* Näytön leveyden ollessa vähintään 570px*/
@media only screen and (min-width: 570px) {
    .button {
        /* Painike piilotetaan käyttöliittymästä */
        display: none;
    }
}
```

KUVA 14. Kartan avaava painike

Painikkeen sijainti on määrätty, mutta sen etäisyys alareunaan ja näytön oikeaan reunaan mukautuu näytön koon mukaan. Painikkeen koko on myös riippuvainen näytön koosta, sillä elementin täyte on joustava. Z-index-toiminnon avulla painike nostetaan muun sisällön päälle asettamalle toiminnon arvoksi 2. Näytön leveyden ollessa vähintään 570 px, painike piilotetaan käyttöliittymän näkymästä display-toiminnolla.

Pienillä näytöillä painikkeesta avautuu näytön kokoinen elementti, johon on sijoitettu kartta ja elementin sulkeva painike. Keskisuurilla ja suurilla näytöillä samaa elementtiä käytetään kartan sijainnin määrittämiseen. Kuvassa 15 esitetään kartan sisältävän elementin ja kartan tyyliluokat.

```
.mapContainer {
    /*Sijainti ja keskitys */
    position: fixed;
    left: 0;
    top: 0;
    /* Leveys ja korkeus */
    width: 100%;
    height: 100vh;
    /* Elementin järjestys pinossa */
    z-index: 3;
    /* Tyyli */
    background-color: rgba(255, 255, 255, 0.9);
}
/* Kartta */
.map {
    /* Sijainti ja keskitys */
    left: 50%;
    top: 50%;
    transform: translate(-50%, -50%);
    /* Leveys ja korkeus */
    width: 90%;
    height: 80%;
}
```

```
/* Näytön leveyden ollessa vähintään 570px */
@media only screen and (min-width: 570px){
    .mapContainer {
        /*Sijainti ja keskitys */
        position: relative;
        left: auto;
        top: auto;
        /* Leveys ja korkeus */
        width: 100%;
        height: auto;
        /* Elementin järjestys pinossa */
        z-index: 0;
        /* Tyyli */
        background-color: transparent;
    }
    .map{
        /* Sijainti ja keskityksen poisto */
        position: relative;
        left: auto;
        top: auto;
        transform: translate(0,0);
        /* Leveys ja korkeus */
        width: 100%;
        height: 400px;
    }
}
```

KUVA 15. Kartan responsiivisuus

Pienillä näytöillä kartan sijainnin määrittävä elementti asetetaan näyttöalueen kokoiseksi. Sijainti saadaan määritettyä näyttöalueen kokoiseksi käyttämällä height-toiminnossa suuretta vh. 1 vh vastaa näyttöalueen sadasosaa, jolloin 100 vh:ta on yhtä suuri kuin näyttöalue. Elementin leveys on joustava ja määritetty yhtä suureksi kuin näytön leveys. Z-index-toiminnon arvolla 3 elementti asetetaan muiden elementtien päälle.

Keskisuurilla ja suurilla näytöillä kartan sijainnin määräävän elementin position-, top- ja left-toiminnot asetetaan oletusarvoisiksi mediakyselyä käyttämällä. Näin sijainnin määräävä elementti asettuu home-komponentissa sille osoitettuun paikkaan. Elementin korkeudeksi asetetaan arvo auto, jolloin korkeus mukautuu

automaattisesti elementin sisällön mukaan. Lisäksi elementin z-indexin arvoksi asetetaan 0, jolloin se on samalla tasolla muun sisällön kanssa.

Pienillä näytöillä kartta keskitetään sijainnin määräävään elementtiin. Kartta on hieman pienempi kuin sijainnin määräävä elementti, jotta kartan yläpuolelle saadaan asetettua suunnitelman mukaisesti kartan sulkeva painike. Keskisuurilla ja suurilla näytöillä kartan sijainniksi asetetaan oletusarvo ja keskitykset poistetaan mediakyselyä käyttämällä. Kartan leveys asetetaan sijainnin määräävän elementin suuruiseksi ja kartan korkeudeksi määritetään 400 px.

Kartan sulkeva painike on käytössä vain pienillä näytöillä. Painikkeen tyyliluokassa määritetään painikkeen näkyvyys ja sijainti kartan sijainnin määräävässä elementissä. Kuvassa 16 esitetään kartan sulkevan painikkeen tyyliluokka.

```
/* Kartan sulkeva painike */
.closeBtn {
    /* Sijainti */
    position: absolute;
    top: 5%;
    right: 5%;
}
/* Näytön leveyden ollessa vähintään 570px */
@media only screen and (min-width: 570px){
    .closeBtn{
        /* Painike piilotetaan käyttöliittymästä */
        display: none;
    }
}
```

KUVA 16. Kartan sulkevan painikkeen tyyliluokka

Kartan sulkeva painike asetetaan pienillä näytöillä kartan yläpuolelle. Kartan korkeuden ollessa 80 % voidaan painikkeen sijainti asettaa joustavasti 5 %:n päähän yläreunasta. Sulkevaa painiketta ei tarvita keskisuurilla ja suurilla näytöillä, jonka vuoksi mediakyselyä käyttämällä painike piilotetaan suurien näyttöjen käyttöliittymistä.

3.2.4 Verkkosovelluksen alatunniste

Verkkosovellukseen on toteutettu myös alatunniste, joka sisältää tässä työssä tekstiä. Alatunnisteeseen voidaan laittaa esimerkiksi yrityksen yhteystietoja tai sosiaalisen median linkkejä. Alatunniste on sijoitettu näytön alareunaan näytön koosta riippumatta. Kuvassa 17 esitetään alatunnisteen tyyliluokka.

```
/* Alatunniste */
.footer {
    /* Alatunnisteen sijainti */
    position: absolute;
    /* Keskittäminen */
    left: 50%;
    transform: translateX(-50%);
    /* Elementin sisällön sijoittaminen */
    display: grid;
    grid-template-columns: auto auto;
    /* Marginaali */
    margin: 0;
    /* Leveys */
    width: 90%;
    /* Testin sijainti */
    text-align: center;
    /* Fonttikoko */
    font-size: 80%
}
/* Alatunnisteen vasen kolumni */
.columnLeft {
    /* Marginaali */
    margin-right: 3%;
    /* Tekstin sijainti */
    text-align: right;
    /* Fonttikoko*/
    font-size: x-small;
    /* Tyyli */
    white-space: pre;
}
/* Alatunnisteen oikea kolumni */
.columnRight {
    /* Marginaali */
    margin-right: 3%;
    /* Tekstin sijainti */
    text-align: left;
    /* Fonttikoko*/
    font-size: x-small;
    /* Tyyli */
    white-space: pre;
}
/* Näytön leveyden ollessa vähintään 825px ja korkeuden ollessa vähintään 1085px */
@media screen and (min-width: 825px) and (min-height:1085px) {
    .footer {
        /* Alatunnisteen sijainti */
        bottom: 0;
    }
    .columnLeft {
        /* Marginaali */
        margin-right: 2%;
        /* Fonttikoko */
        font-size: small;
    }
    .columnRight{
        /* Marginaali */
        margin-left: 2%;
        /* Fonttikoko */
        font-size: small;
    }
}
```

KUVA 17. Alatunniste

Alatunniste sijoitetaan keskitetysti käyttöliittymän alareunaan. Mikäli verkkosovelluksen sisältö menee näyttöalueen yli, alatunniste sijoitetaan sisällön perään. Muussa tapauksessa alatunniste sijoitetaan näyttöalueen alareunaan. Sijainti on toteutettu asettamalla alatunnisteen position-toiminnon arvoksi abosolute ja lisäämällä sijaintiin toiminnon bottom arvoksi 0 kun näytön leveys on suurempi kuin 825 px ja korkeus on suurempi kuin 1085 px.

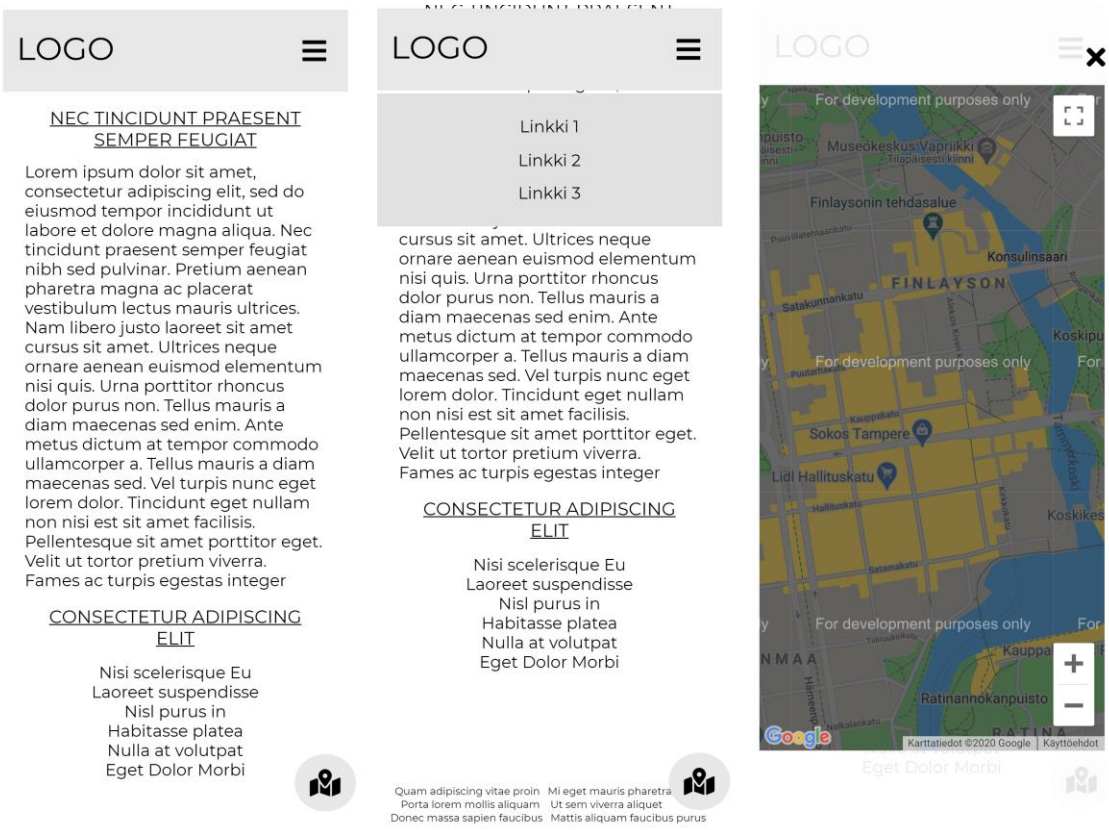
Alatunnisteeseen luodaan ruudukko, johon alatunnisteen sisältö sijoitetaan. Ruudukko sisältää kaksi kolumnia, joiden leveys on määritetty joustavaksi arvolla auto. Arvolla auto luodaan kaksi yhtä suurta kolumnia, joiden yhteenlaskettu leveys on yhtä suuri kuin elementin leveys. Ruudukon sisältö on myös joustava,

sisällön fonttikoko ja marginaali muuttuu näytön koon mukaan. Fonttikokoa muutettaessa varmistetaan, että teksti on luettavaa erikokoisilla laitteilla.

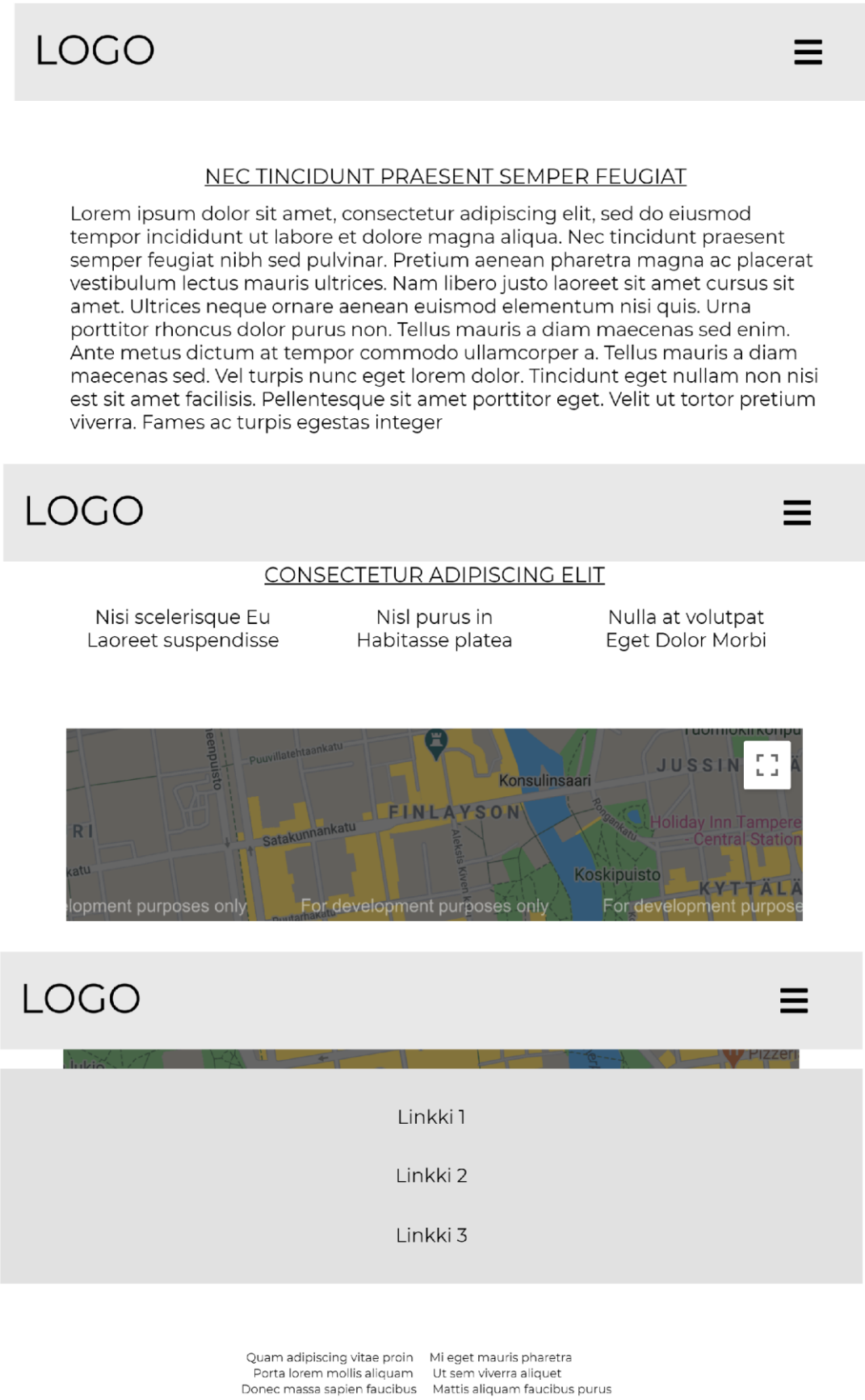
4 RESPONSIIVISEN VERKKOSOVELLUKSEN TESTAUS

Verkkosovelluksen responsiivisuutta voidaan testata erilaisilla kolmannen osapuolen kehysympäristöillä tai selaimen kehittäjän työkalulla. Tässä työssä toteutetun verkkosovelluksen testaamisessa on käytetty apuna Chrome-selaimen kehittäjän työkalua. Kehittäjän työkalun avulla voidaan testata verkkosovelluksen käyttöliittymä simuloimalla verkkosovelluksen toiminta erikokoisilla laitteilla. Lisäksi kehittäjän työkalun avulla voidaan suorittaa suoritustehon mittaus sekä auditointi käyttämällä Lighthouse ohjelmaa.

Kehittäjän työkalulla voidaan simuloida verkkosovelluksen käyttöliittymää useilla eri mobiililaitteilla. Simulointi voidaan toteuttaa myös responsiivisesti, jolloin näytön kokoa voi muuttaa joustavasti. Verkkosovelluksen pienien näyttöjen käyttöliittymä testattiin simuloimalla kahta näytön suuntaa, pysty- ja vaakasuuntaa. Kuvassa 18 esitetään käyttöliittymän pystysuunnan simuloinnin kuvat ja kuvassa 19 esitetään käyttöliittymän vaakasuunnan simuloinnin kuvat.



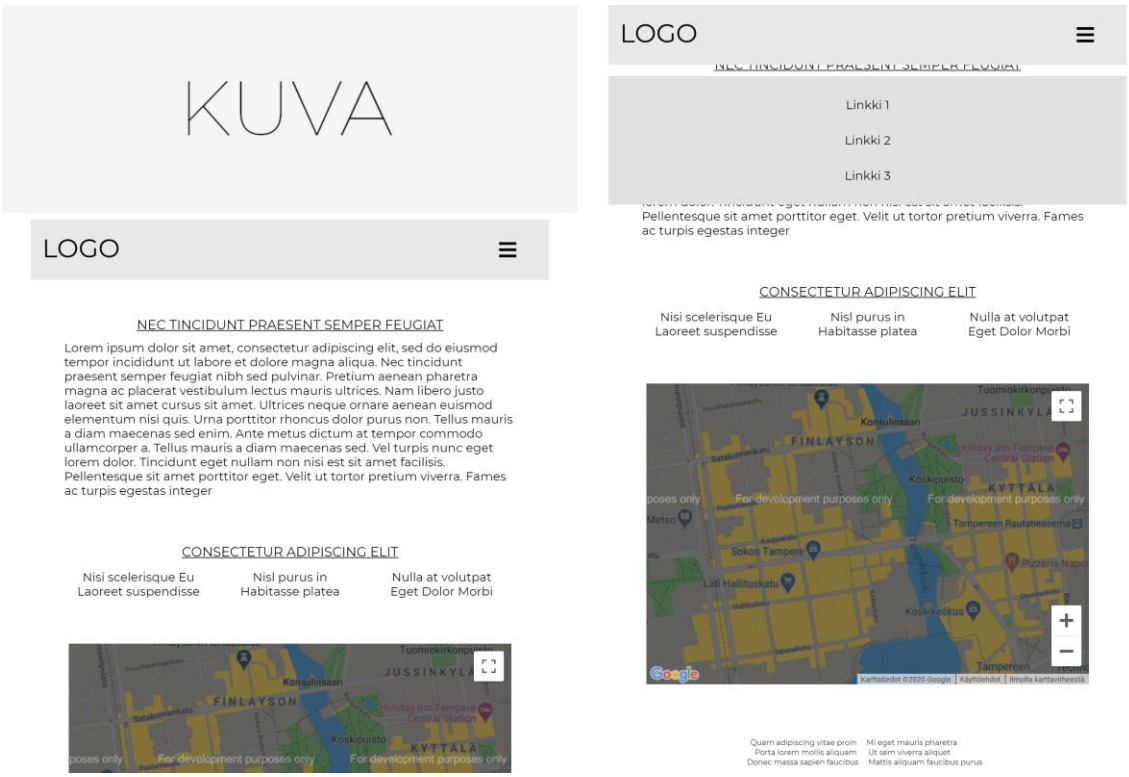
KUVA 18. Verkkosovelluksen käyttöliittymä pienillä näytöillä, pystysuunta



KUVA 19. Verkkosovelluksen käyttöliittymä pienillä näytöillä, vaakasuunta

Pienien näyttöjen käyttöliittymän simuloinnista havaitaan, että tehty käyttöliittymäsuunnitelma toteutuu verkkosovelluksessa. Näyttösuunnasta riippumatta pienillä näytöillä ei ole ylätunnistetta. Navigaatiopalkissa on painike, joka avaa pudotusvalikon, jossa on navigaatiolinkit. Pystysuuntaisessa näyttösuunnassa verkkosovelluksen kartta avataan ja suljetaan painikkeella. Lisäksi käyttöliittymän elementit ovat suunnitellulla paikalla.

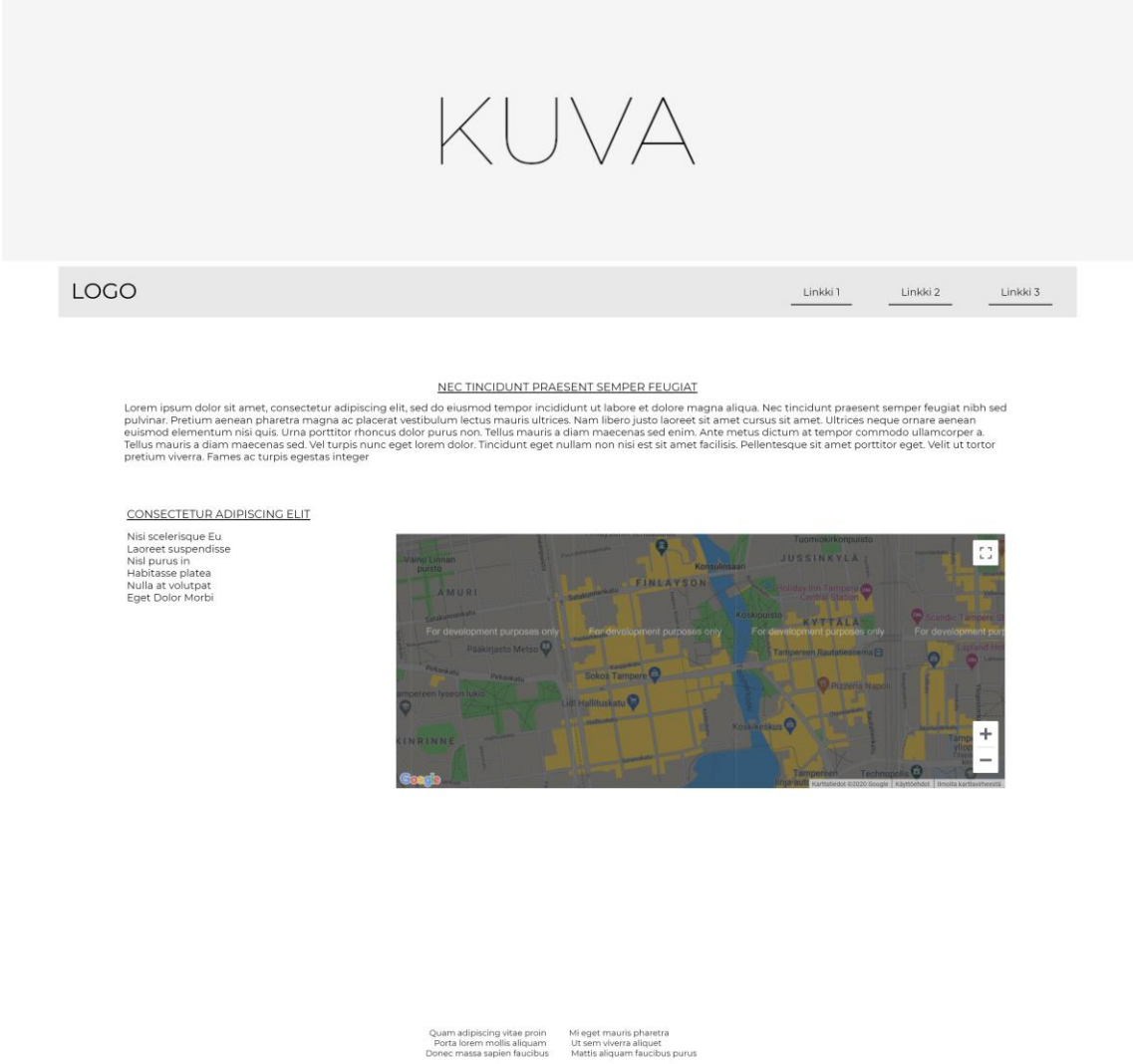
Kehittäjän työkalulla simuloitiin myös keskisuurien näyttöjen käyttöliittymä. Simulointi toteutettiin vain yhdellä näytön suunnalla, sillä keskisuurilla näytöillä on sama käyttöliittymä pysty- ja vaakasuunnassa. Kuvassa 20 esitetään kuvat keskisuurien näyttöjen käyttöliittymästä.



KUVA 20. Verkkosovelluksen käyttöliittymä keskisuurilla näytöillä

Keskisuurien näyttöjen käyttöliittymän simuloinnista havaitaan, että käyttöliittymä toimii edelleen suunnitellulla tavalla. Pienien näyttöjen käyttöliittymästä poiketen, keskisuurien näyttöjen käyttöliittymässä on ylätunniste, joka sisältää kuvan. Lisäksi kartta on käyttöliittymässä näkyvissä automaattisesti. Simuloinnista havaitaan myös, että elementit sijoittuvat halutulla tavalla käyttöliittymälle.

Viimeisenä verkkosovelluksesta testataan suurien näyttöjen käyttöliittymä. Suurien näyttöjen käyttöliittymä on simuloitu kannettavalla tietokoneella. Kuvassa 21 esitetään kuva suurien näyttöjen käyttöliittymästä.

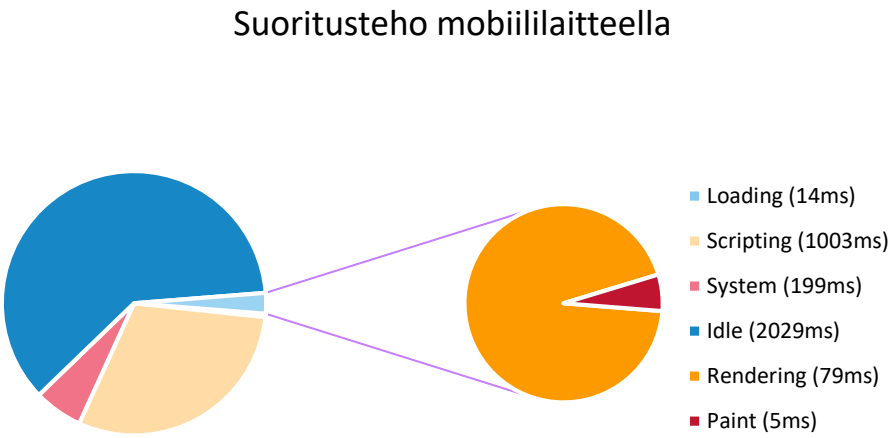


KUVA 21. Verkkosovelluksen käyttöliittymä suurilla näytöillä

Suurien näyttöjen käyttöliittymätestistä havaitaan, että myös suurilla näytöillä käyttöliittymä vastaa suunnitelmaa. Pienempien näyttöjen käyttöliittymistä poiketen, suurien näyttöjen käyttöliittymässä navigaatiolinkit on sijoitettu navigaatiopalkkiin. Elementit sijoittuvat edelleen halutulla tavalla käyttöliittymään, luettelo ja kartta ovat samalla tasolla sekä alatunniste pysyy näytön alareunassa.

Verkkosovelluksen suoritusteho testattiin kehittäjän työkalun suoritustehon profilointityökalulla. Työkalun avulla voidaan testata verkkosovelluksen

lataukseen kuluva aika. Työkalun asetuksia muuttamalla on mahdollista simuloida toiminta mobiililaitteella hidastamalla nettiyhteyden sekä prosessorin nopeutta. Työkalulla testattiin verkkosovelluksen toiminta mobiililaitteella ja tietokoneella. Suoritusteho profiloitiin viisi kertaa. Kuviossa 1 esitetään profilointien tuloksien keskiarvo suoritustehoa simuloitaessa mobiililaitteella.



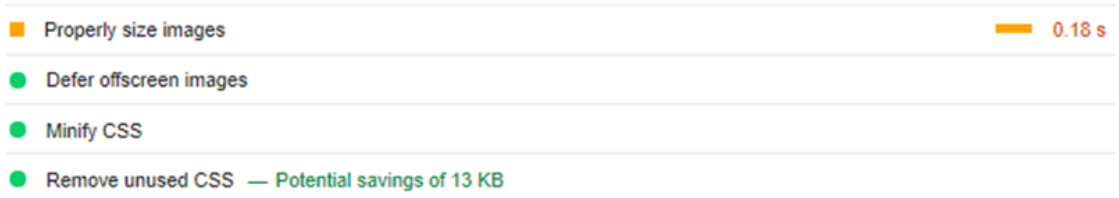
KUVIO 1. Mobiililaitteen suoritustehon tulokset

Profiloinnissa mitataan verkkosovelluksen lataukseen, verkkosovelluksen koodin kääntämiseen, käyttöliittymän tulostamiseen ja järjestelmän toimintoihin kuluva aika. Responsiivisuudella vaikutetaan verkkosovelluksen HTML-, CSS- ja JS-tiedostojen kääntämiseen ja käyttöliittymän tulostumiseen kuluvaan aikaan. Suoritustehon profiloinnissa kiinnitetäänkin huomiota rendering- ja paint-prosessien arvoihin.

Suoritustehon profiloinnista mobiililaitteella havaitaan, että responsiivisuuteen vaikuttavat prosessit ovat lyhytkestoisia. Mobiililaitteella verkkosovelluksen latausaika on testien mukaan 3,3 s. Prosessit, joihin responsiivisuus vaikuttaa kestivät mobiililaitteella yhteensä 0,08 s. Tuloksesta voidaan siis todeta, että responsiivinen toteutus ei ole heikentänyt verkkosovelluksen suoritustehoa mobiililaitteella.

Kehittäjän työkalussa on mahdollista suorittaa myös auditointi verkkosovelluksen toiminnasta. Auditoinnissa tehdään useita testejä, joissa testataan

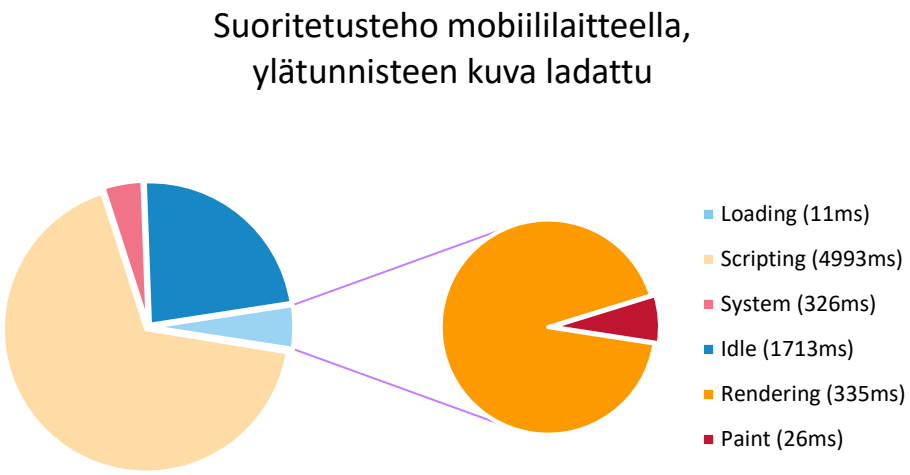
verkkosovelluksen suoritustehoa. Auditoinnissa selvitetään mm. tiedostojen koko ja käyttämättömän ohjelmakoodin määrä. Auditoinnissa mitattuja arvoja verrataan ennalta määrättyihin arvoihin ja niiden perusteella tulostetaan raportti, jossa kerrotaan, suorittiko verkkosovellus testin hyväksytysti vai hylätysti, ja miten suoritustehoa voitaisiin parantaa. Kuvassa 22 esitetään verkkosovelluksen auditoinnin tulos mobiililaitteella simuloituna.



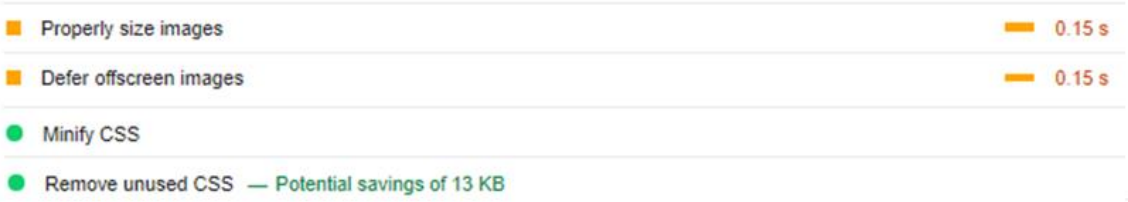
KUVA 22. Auditointi mobiililaitteella

Auditoinnin tuloksista havaitaan, että mobiililaitteella logon kuvan koko heikentää suoritustehoa. Kohdassa Properly sized images on kerrottu oransilla värillä, että testi meni läpi, mutta pienentämällä kuvaa suoritusteho voisi parantua 0,18 s. Muut auditoinnin testit, joihin responsiivinen toteuttaminen vaikuttaa, ovat menneet läpi. Nämä testit näytetään auditoinnin raportissa vihreällä värillä. Auditoinnissa voidaan huomata ylätunnisteen kuvan lataamatta jättämisen parantaneen verkkosovelluksen suoritustehoa. Kohta Defer offscreen images on suoritettu auditoinnissa hyväksytysti, jolloin voidaan todeta, että ylätunniste ei heikennä mobiililaitteella verkkosovelluksen suoritustehoa.

Ylätunnisteen kuvan latauksen vaikutus testattiin vielä muokkaamalla ylätunnisteen koodia niin, että kuva ladattiin myös pienien näyttöjen käyttöliittymään ja piilotettiin näkyvistä. Suoritusteho profiloitiin jälleen viisi kertaa. Kuviossa 2 esitetään profilointien tuloksien keskiarvo ja kuvassa 23 auditoinnin tulos.



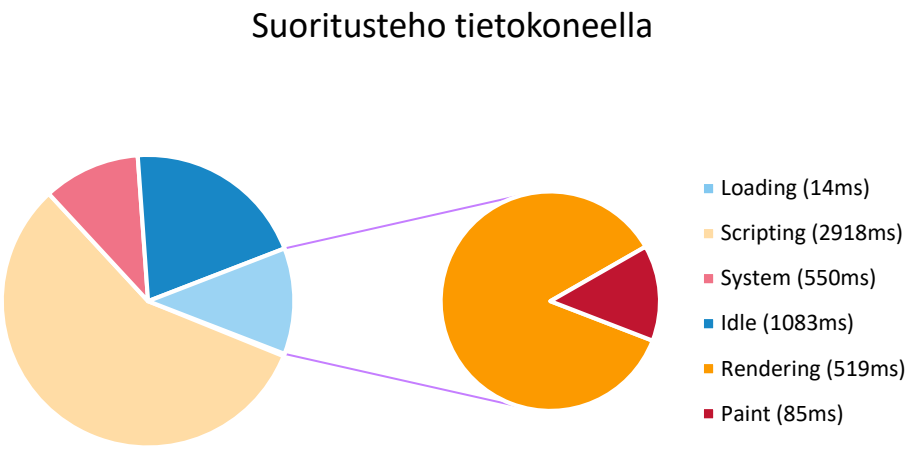
KUVIO 2. Mobiililaitteen suoritustehon tulokset, ylätunnisteen kuva ladataan



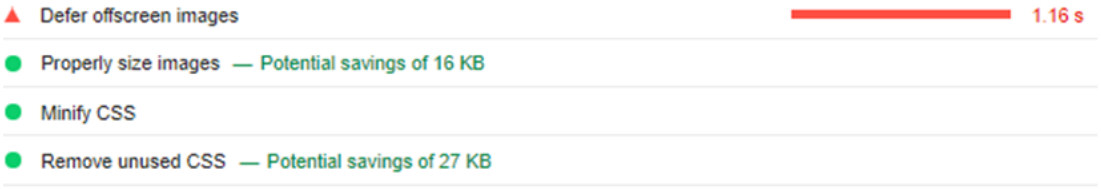
KUVA 23. Auditointi mobiililaitteella, ylätunnisteen kuva ladataan

Suoritustehon tuloksista havaitaan, että pelkkä kuvan piilottaminen heikentää verkkosovelluksen suoritustehoa profiloinnin mittauksen perusteella 0,28 s. Myös auditoinnin raportista havaitaan, että kuvan piilottaminen käyttöliittymästä heikentää suoritustehoa 0,15 s. Lisäksi mobiililaitteella ladattavien tiedostojen koko kasvaa ja tämä myös heikentää suoritustehoa.

Suoritustehon profilointi ja auditointi suoritettiin myös tietokoneella simuloituna. Suoritustehon profiloinnissa, kehittäjän työkalujen asetuksista, nettiyhteyden ja prosessorin nopeutta parannettiin, jolloin simulointi vastaa enemmän oikeaa tilannetta. Suoritusteho profiloitiin jälleen viisi kertaa. Kuviossa 3 esitetään profilointien tuloksien keskiarvo ja kuvassa 24 esitetään auditoinnin tulokset.



KUVIO 3. Tietokoneen suoritustehon tulokset



KUVA 24. Auditointi tietokoneella

Profiloinnin mittaustuloksista havaitaan, että verkkosovelluksen lataukseen kuluva aika kasvaa hieman ja lataus kestää nyt 5,1 s. Responsiivisen toteutuksen vaikutukset suoritustehossa ovat kuitenkin pienet, sillä rendering- ja paint-prosessien yhteenlaskettu aika on 0,6 s. Profiloinnin tuloksesta voidaan todeta, että responsiivinen toteutus ei vaikuta merkittävästi verkkosovelluksen suorituskykyyn.

Auditoinnin raportin perusteella responsiivisuus vaikuttaa suoritustehoon, ja responsiivista toteutusta parantamalla suoritustehoa olisi mahdollista parantaa 1,6 s:lla. Auditoinnissa verkkosovellus ei ole suorittanut Defer offscreen images-testiä hyväksytysti. Raportista käy ilmi, että auditoinnissa tulosta on heikentänyt sisällössä oleva karttanäkymä.

5 POHDINTA

Työssä onnistuttiin toteuttamaan responsiivinen verkkosovellus etukäteen laaditun sisällön pohjalta. Verkkosovelluksen käyttöliittymä mukautuu erikokoisille laitteille ja jokaisen laitekoon käyttöliittymä sisältää saman sisällön. Responsiivinen toteuttaminen ei myöskään heikentänyt verkkosovelluksen suoritustehoa merkittävästi, ja näin ollen käyttökokemus saadaan säilytettyä hyvänä jokaisella laitteella.

Työ aloitettiin responsiivisen verkkosovelluksen suunnittelusta. Suunnittelussa huomioitiin onnistuneesti sisällön priorisointi sekä käyttöliittymän käytettävyys. Sisällön priorisoinnissa tekstisisältö, joka sisältää yrityksen kuvauksen, ja luettelo, jossa kerrotaan palveluista, nousivat tärkeimmiksi. Tämän vuoksi pienien näyttöjen käyttöliittymässä kuvaus ja luettelo ovat aina näkyvissä. Kartta ja navigaatiolinkit ovat puolestaan piilotettuja ja aukeavat painikkeella, sillä ne sisältävät toissijaista sisältöä.

Responsiivisen verkkosovelluksen toteutuksen tavoitteina oli toteuttaa suunnitelmaa vastaava verkkosovellus erikokoisille laitteille heikentämättä suoritustehoa. Verkkosovellukselle suoritettujen testien perusteella verkkosovelluksen toteuttamisessa onnistuttiin tavoitteiden mukaisesti. Käyttöliittymästä nähdään, että elementit ja toiminnot ovat käyttöliittymällä suunnitellulla paikalla, ja suoritustehon profilointi ja auditointi kertovat, että responsiivinen toteuttaminen ei aiheuta suoritustehon merkittävää heikentymistä.

Suoritustehon kannalta tärkeimmäksi toiminnoksi verkkosovelluksen toteutuksessa on käyttöliittymän ylätunniste erikokoisilla näytöillä. Testituloksista tuli ilmi, että kuvan lataamatta jättäminen on tärkeää, jotta suoritusteho ei heikkene. Responsiivisen verkkosovelluksen toteuttaminen olisi ollut yksinkertaisempaa, mikäli ylätunnisteeseen olisi ladattu vain yksi kuva, jonka koko muuttuu ja joka tarvittaessa piilotetaan käyttöliittymästä. Nykyään verkkosovelluksia kehitettäessä on kuitenkin tärkeää ottaa huomioon verkkosovelluksen suorituskyky, sillä käyttäjä saattaa poistua verkkosovelluksesta, mikäli se ei lataudu riittävän nopeasti.

Verkkosovelluksen suoritustehoon liittyy monia asioita, mutta responsiivisuuden näkökulmasta verkkosovelluksen tehokkuutta olisi vielä voitu parantaa luomalla verkkosovellukselle useampia kuvia ylätunnisteeseen sekä logoon tai käyttämällä vektorigrafiikka bittikarttakuvien sijaan. Lisäksi verkkosovelluksen tiedostomäärää olisi voitu pienentää luomalla jokaiselle käyttöliittymälle oma tyyliohje.

Verkkosovelluksen toteutuksessa luotiin käyttöliittymän elementit komponentteina. Jokaisella komponentilla on oma tyylitiedosto, jossa määritetään komponentin responsiivisuus ja tyyli eri käyttöliittymillä. Mikäli verkkosovelluksen toteutuksessa jokaiselle käyttöliittymälle olisi toteutettu oma tyyliohje, olisi verkkosovelluksessa vain kolme tyyliohjetta. Lisäksi verkkosovelluksen index.html-tiedostossa olisi voitu ladata tyyliohje näytön koon mukaan. Näin verkkosovelluksen suoritustehoa olisi voitu parantaa.

Kaiken kaikkiaan toteutettu verkkosovellus täyttää työssä määritetyt tavoitteet ja verkkosovelluksen suoritusteho ei tällaisenaan ole liian suuri. Mobiililaitteella verkkosovelluksen latausajaksi tuli noin 3 s, joka on kohtuullinen latausaika. Tietokoneella latausaika jäi hieman suureksi, joten verkkosovellusta voisi vielä kehittää niin, että suoritustehoa saataisiin parannettua suurilla näytöillä.

LÄHTEET

Carver, M. 2015. The responsive web. Shelter Island: Manning Publications Co.

Gu, T. 2019. Newzoo's Global Mobile Market Report: Insights into the World's 3.2 Billion Smartphone Users, the Devices They Use & the Mobile Games They Play. Artikkeli. Julkaistu 17.9.2019. Luettu 16.3.2020. https://newzoo.com/insights/articles/newzoos-global-mobile-market-report-insights-into-the-worlds-3-2-billion-smartphone-users-the-devices-they-use-the-mobile-games-they-play/

Jia, G. 2017. What Are The Pros and Cons of Responsive Web Design. Artikkeli. Julkaistu 1.6.2017. Luettu 16.3.2020. https://www.mockplus.com/blog/post/pros-and-cons-of-responsive-web-design

Kadlec, T. 2013. Implementing responsive design. Berkeley: New Rides.

MDN. 2008. Using media queries. Opas dokumentti. Julkaistu 8.10.2008. Päivitetty 4.5.2020. Luettu 5.5.2020. https://developer.mozilla.org/en-US/docs/Web/CSS/Media_Queries/Using_media_queries