Opinnäytetyö (AMK)

Tieto- ja viestintätekniikka

2019

Tanja Imberg

ESITTELYMALLI KÄYTTÄJÄTIEDON JA SAAVUTETTAVUUDEN KEHITTÄMISEEN

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tieto- ja viestintätekniikka

2019 | 39 sivua

Tanja Imberg

ESITTELYMALLI KÄYTTÄJÄTIEDON JA SAAVUTETTAVUUDEN KEHITTÄMISEEN

Opinnäytetyön tarkoituksena oli suunnitella ja toteuttaa Kuntien Tieran Parasta Palvelua, PSOP-järjestelmän palveluntuottajien vertailusta esittelymalli, jota voidaan hyödyntää keräämällä palautetta ja kehitysehdotuksia.

Tämä esittelymalli rakennettiin aiemmin luodun graafisen esittelymallin pohjalta. Pääasioina ovat käytettävyyden ja saavutettavuuden parantaminen. Nykyisessä PSOP-järjestelmän selauksessa on ilmennyt joitain saavutettavuutta rajoittavia haasteita, kuten näytönlukulaitteiden haasteet. Hyvällä suunnittelulla voidaan parantaa PSOPin saavutettavuutta, jolloin näkörajoitteisetkin pystyvät vaivattomammin käyttämään järjestelmää. Saavutettavuudessa on huomioitu typografia ja värimaailmat sekä kontrasti. Nämä kolme ovat erityisen tärkeitä eri tavoin rajoittuneille käyttäjille. Esimerkiksi kun tekstissä on oikeanlainen fontti, selkokielinen sisältö sekä hyvä taustan ja tekstin kontrasti, auttavat nämä jo huomattavasti esimerkiksi lukihäiriöisiä henkilöitä saamaan teksteistä selvää. Myös heikkonäköiset, kuten iäkkäämmät ihmiset, hyötyvät käyttöliittymän hyvästä kontrastista. Värimaailmassa taas on tärkeä huomioida, että värit eivät saa yksinään ohjata käyttäjää, vaan niissä täytyy olla myös kuvaava teksti. Käytettävyydessä on huomioitu erityisesti järjestelmän responsiivisuus. Responsiivisuuden kehittämisen tueksi on myös perehdytty Tilastokeskuksen mobiili- ja tablettilaitteiden internetin käytön tutkimukseen. Responsiivinen sivusto asettuu näyttökoon mukaan erikokoisille näytöille sopiviksi, loogisiksi ja helppokäyttöisiksi. Esittelymallin responsiivisuus on toteutettu Bootstrapin avulla. Esittelymallissa järjestelmän käytettävyyttä parannettiin myös uudella hakutoiminnolla, joka hakee palvelut palvelusetelin numerolla, sekä uudenlaisella hakutuloslistauksella ja lisätietosivulla.

Opinnäytetyön toteutuksessa on käytetty useita teknologioita. Esittelymallin Node.js -sovellus, joka on yhdistetty MongoDB -tietokantaan. Tietokantayhteyden ja Node.js -pohjaisen sovelluksen vuoksi, esittelymallilla pystyy hakemaan palveluntuottajia tietokannasta. Palveluntuottajien tiedot listautuvat hakutuloksiin. Esittelymallin käytettävyyden ja jotkin saavutettavuuden tavoitteet onnistuivat hyvin, joskin jatkokehitettävääkin vielä on.

ASIASANAT:

käytettävyys, saavutettavuus, MongoDB, Node.js

BACHELOR´S | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Information and communications technology

2019 | 39 pages

Tanja Imberg

MOCKUP TO DEVELOP USER INFORMATION AND ACCESSIBILITY

The purpose of this thesis was to design and implement a mockup for Kuntien Tiera's Parasta Palvelua PSOP -system's service provider comparison which could be utilized by collecting feedback and suggestions for improvement.

This mockup is based on an earlier graphic mockup. Its main roles are to improve usability ja accessibility. There have been some challenges that limit accessibility, for example, the challenges of using screen readers. With good planning, it is possible improve the accessibility of the PSOP system and then visually impaired users might use this system more easily. Accessibility takes into account typography and color schemes, as well as contrast. These three considerations are especially important for users that are in different ways impaired. For example, the right font with plain text content, and with a good contrast between font and the background, helps users with dyslexia to read more easily. Also, visually impaired people, such as the elderly, also benefit from good contrast in the user interface. In the colour scheme, however, it is important to note that colours must not guide the user by itself but must also contain descriptive text. Usability has been taken into account, especially the system's responsivity. The studies of the Internet use on mobile and tablet devices by Statistics Finland has been oriented to support the development of responsiveness. Responsive design detects the visitor's screen size and orientation and changes the layout accordingly. The responsivity of this mockup has been implemented with Bootstrap. The usability of the mockup has also been improved with a new search function of the service voucher ticket number. There is also a new kind of a search results model and a service provider page.

Several technologies have been used in the thesis. Mockup is a Node.js application that has been connected to MongoDB database. Because of database connectivity and the Node.js-based application, the user can search service providers from the database and the results will be listed in the search results. The mockup's usability and some accessibility goals succeeded well although there is still room for improvement.

KEYWORDS:

Usability, Accessibility, MongoDB, Node.js

SISÄLTÖ

KUVAT

KÄYTETYT LYHENTEET TAI SANASTO

¶

API	Application programming interface, ohjelmointirajapinta
BSON	JSON-asiakirjat binaarikoodatussa muodossa
CSS 3	Cascading Style Sheets. Tyylitiedosto, jonka säännöillä tehdään verkkosivuston ulkoasu
ejs	Embedded JavaScript templates
Etätietovarasto	Tietovarasto, joka ei sijaitse omalla koneella, mutta siihen voidaan ottaa yhteys (engl. remote repository)
Framework	Vapaasti käännettynä kehityskehys. Kehitysohjelmisto, jossa on mm. koodikirjastoja ja työkaluja, joilla helpotetaan ohjelmiston kehitys.
HTML 5	Hypertext Markup Language. Avoimesti standardoitu kuvauskieli, joka tunnetaan erityisesti kielenä, jolla internetsivut on ohjelmoitu.
ISO	International Organisation for Standardization
JS	JavaScript-kieli

Kommitti	Tietovarastoon tallennettu versio tai muutosjoukko (engl. commit)
Kommitoida	Tarkoittaa kommitin tallentamista tietovarastoon (engl. to commit)
Lokaali	Paikallinen, globaalin vastakohta
Muutosjoukko	Kokoelma tiedostoihin tehtyjä muutoksia (engl. changeset)
Natiivisovellus	Natiivisovellus tehdään sekä Androidille että iOSille omalla koodilla, eli samaan sovellukseen tarvitaan kaksi koodia, jotta sovellus toimii Android ja iOS -puhelimissa
PaaS	A cloud platform as a service
PSOP	Palveluseteli- ja ostopalvelujärjestelmä (Parasta Palvelua)
Sass	Syntactically Awesome Style Sheets
Tietovarasto	Tallennuspaikka projektin tiedostoille ja niiden historialle (engl. repository)
Työntää muutoksia	Tarkoittaa paikallisen tietovaraston kommittien lähettämistä etätietovarastoon (engl. to push)
UI/UX	User Interface / User Experience

VCS	Version control systems, versionhallintajärjestelmä
Vetää muutoksia	Tarkoittaa kommittien hakemista etätietovarastosta, monissa ohjelmissa myös niiden yhdistämistä paikalliseen haaraan (engl. to pull)
WCAG	Web Content Accessibility Guidelines, Verkkosisällön saavutettavuusohjeet
Widget	Nopeakäyttöinen minisovellus, pienoisohjelma

1 JOHDANTO

Opinnäytetyö sisältää kaksi kokonaisuutta: ohjelmoidun esittelymalli, sekä raportin. Esittelymallilla esitetään, miten järjestelmä voisi toimia ja miltä se voisi näyttää, sekä raportin, joka kertoo työn etenemisestä, haasteista ja toteutuksesta. Opinnäytetyön sisältö perustuu ammatilliseen tietoon, kattavaan lähdemateriaaleihin perehtymiseen sekä aikaisempiin palvelun käytettävyystutkimuksiin, asiakaspalauteisiin ja kehitysehdotuksiin. Työssä kerrotaan myös, mitä valinnanvapaus ja asiakaslähtöisyys ovat. Jotta lukeminen on jouhevaa, käydään myös läpi opinnäytetyön taustaa siitä, mitä teknologioita on käytetty, sekä keskeisimpiä käytettävyyden ja saavutettavuuden asioita.

Toimeksiantajana toimii Kuntien Tiera, joka on vuonna 2010 perustettu valtakunnallinen ja kuntaorganisaatioiden omistama inhouse-yhtiö. Yritys tarjoaa ICT-palveluita kuntien, kaupunkien ja kuntayhtymien sujuvan arjen ja toiminnan kehittämisen tueksi. Palvelukehitys perustuu omistajaohjaukseen sekä valtakunnalliseen yhteistyöhön kuntien kanssa. Tieralla on viisi toimipistettä, joista suurin on Turussa. [1.] Toimeksiantajayrityksen palvelun tärkeimpiin kiinnekohtiin kuuluu asiakaslähtöisyys, mikä tarkoittaa, että palvelun toiminnan lähtökohtana ovat asiakkaiden tarpeet sekä tavoitteet.

Tieran SOTE-palvelut on kehittänyt yhteistyössä kuntien kanssa Parasta Palvelua -verkkopalvelun ja sitä tukevan palveluseteli- ja ostopalvelujärjestelmän PSOP:n. PSOP-verkkopalvelussa kuntalainen, asiakas, pystyy tarkastelemaan hänelle myönnettyjä palveluseteleitä sekä palveluita. Järjestelmässä olevat palvelut voivat olla eri sosiaali- ja terveyshuollon palveluista aina varhaiskasvatuksen palveluihin. Käyttäjä voi käyttää järjestelmää joko järjestelmään kirjautuneena tai kirjautumatta. Palveluntuottajien vertailussa voi vertailla palveluntuottajien palveluiden hintoja, laatua ja muita tietoja. Parasta Palvelua tekee mahdolliseksi asiakassetelien ja henkilökohtaisen budjetin käytön sekä toteuttaa valinnanvapautta. [2,3.] Parasta Palvelua -järjestelmä on perustettu valinnanvapauden tueksi. Valinnanvapaudella tarkoitetaan, että yksilöllä on oikeus valita hoitopaikka ja hoitava terveydenhuollon ammattilainen itse. [4.]

Toimeksiantona oli kehittää toimeksiantajalle PSOP:sta ohjelmoitu esittelymalli, jolla havainnollistettaisiin asiakkaille palvelusetelin numerolla hakua sekä saavutettavuuden ja käytettävyyden parannuksia. Toimeksiannon käytännön osuuden tavoite oli luoda responsiivinen esittelymalli Palveluseteli- ja ostopalvelujärjestelmän palveluntuottajan vertailusta. Tärkeä ilmenevä ominaisuus opinnäytetyössä oli käyttöliittymä, jonka avulla

käyttäjiä pystyttäisiin ohjaamaan niin sanotulla ohjaavalla mallilla. Ohjaavalla mallilla käyttäjää ohjataan käyttöliittymän avulla järjestelmän käytössä. Tällöin sivustolle ei tarvitse runsaita ohjetekstejä. Myös palvelusetelin numerolla haku -toiminto oli tärkeä toiminnallisuus, joka helpottaa palveluntuottajien hakua. Palvelusetelin numerolla haettaessa ohjelma osaa hakea oikeat tiedot annetun palvelusetelin numeron tietojen perusteella. Vaihtoehtoinen haku -toiminto olisi kunnalla haku -toiminto. Tällöin käyttäjä voisi hakea palveluita kunnan perusteella esimerkiksi Turussa olevia palveluita. Hakutulossivuja eli palveluntuottajien vertailusivuja, tuli olla vain yksi. Palveluntuottajien vertailuun toivotaan monia muitakin käytettävyyteen liittyviä muutoksia. Myös palvelun saavutettavuutta oli tarkoitus parantaa. Saavutettavuusdirektiivi on tullut voimaan 22.12.2016, mikä tarkoittaa, että kaikkien sähköisten palveluiden saavutettavuutta vaaditaan parantaa ja tehdä verkkosivustot ja palvelut kaikille saavutettavaksi. Verkkosivustojen, jotka on julkaistu ennen 23.9.2018, pitää olla saavutettavuusvaatimusten mukaisia 23.9.2020 mennessä. [5,6.] Esittelymallista oli tarkoitus tehdä mahdollisimman toimiva näytönlukulaitteella sekä pistekirjoitusnäytöllä.

Esittelymallin pohjana käytettiin aiemmin luotua graafista esittelymallia, joka on tehty Adobe XD -ohjelmalla. Graafista esittelymallia esiteltiin palvelunmuotoilun työpajassa asiakkaille viime keväänä 2019, ja siitä saatujen kehitysehdotuksien ja kommenttien avulla alettiin viemään esittelymallin kehitystä eteenpäin. Ohjelmoitu esittelymalli on hyvä pohja, jota toimeksiantajayrityksessä voidaan jatkokehittää. Sitä tullaan hyödyntämään toimittajan apuna. Toimittaja toteuttaa verkkosivuston lopullisen toteutuksen. Käytännön työtä esiteltiin asiakaspäivillä 10.10.2019, jolloin kerättiin palautetta, ja käytiin läpi mitä asioita on tärkeä alkaa jatkokehittämään.

2 KÄYTETTÄVYYS JA SAAVUTETTAVUUS

2.1 Käytettävyys

Käytettävyys (engl. usability) on tuotteen tai palvelun ominaisuus, joka kertoo, kuinka optimaalinen kyseinen tuote tai palvelu on. Käytettävyys on melko laaja käsite, jonka määrittelyyn on luotu joitakin eri menetelmiä. [7.]

Käytettävyyden kehittämiseen voi käyttää ISO 9241–11 -standardin määritelmää ja Jacob Nielsenin käytettävyysmenetelmä.

ISO 9241-11 -standardi määrittelee käytettävyyden seuraavalla tavalla:

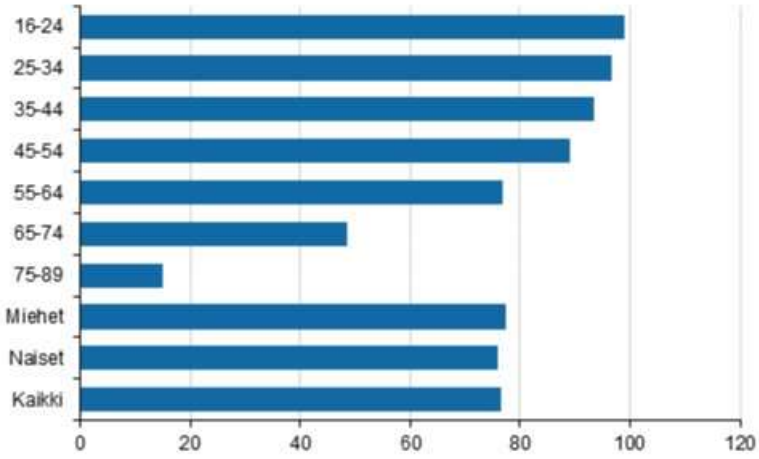
> ”Se vaikuttavuus, tehokkuus ja tyytyväisyys, jolla tietyt määritellyt käyttäjät saavuttavat määritellyt tavoitteet tietyssä ympäristössä.” [8.]

Tietotekniikassa käytettävyys on usein käytetty termi ja siihen sisältyy useimmiten useita eri asioita käyttöliittymän lisäksi. Näitä ovat muun muassa esteettömyys, saavutettavuus ja käyttäjäkokemus, verkkosivut, laitteiden sekä ohjainten muotoilu. Käytettävyyden toteuttamisessa tietojärjestelmän kehittäjältä edellytetään laajaa perehtyneisyyttä, ymmärrystä ja osaamista toteutettavasta ohjelmistosta. Tietotekniikka ei ole ainoa käytettävyyden kehityksessä ja suunnittelussa käytettävä tieteenala, vaan siinä yhdistetään useampaa erilaista tieteenalaa, kuten psykologiaa, kognitiivista psykologiaa, tietotekniikkaa, sekä kognitiotiedettä ja teollista muotoilua. [7.] Tässä opinnäytetyössä käytettävyydellä tarkoitetaan sitä, kuinka helppoa esittelymallia ja tulevaa palvelua olisi käyttää ja kuinka helposti käyttäjä löytää tarvitsemansa tiedot. Myös saavutettavuus ja helppokäyttöisyys ovat oleellisia.

2.1.1 Internetin käyttö mobiililaitteilla

Kolmella neljästä 16–89-vuotiaasta on kosketusnäytöllinen ja 3G- tai 4G-internetyhteydellä varustettu puhelin. Vuodesta 2013 vain 55%:lla oli kosketusnäytöllinen älypuhelin. Älypuhelimien käyttö yleistyy noin 5 prosenttiyksikköä vuodessa. Vuosikymmenen alussa, kun ensimmäiset älypuhelimet tulivat markkinoille, älypuhelimien käyttäjien määrän kasvu oli vielä tätäkin nopeampaa. Alle 55-vuotiaista 94%:lla on nykyään älypuhelin,

joista vain vanhimmissa ikäryhmissä älypuhelimien käyttö ei ole niin yleistä (Kuvio 1). [9.]

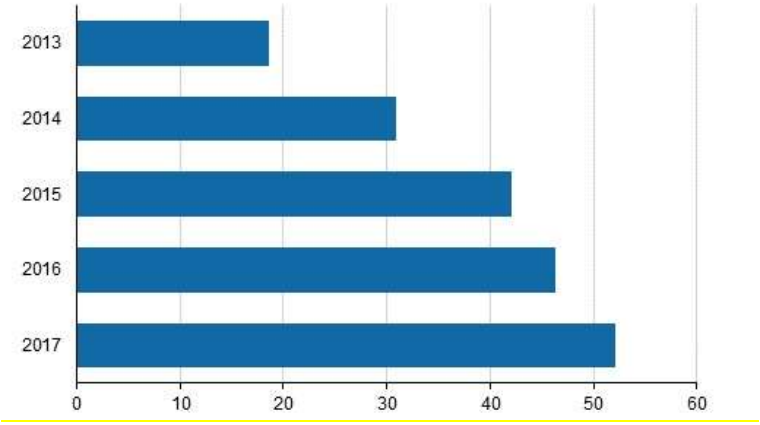


Kuvio 1 Älypuhelin omassa käytössä 2017, %-osuus väestöstä [9]

Älypuhelimien helppo käytettävyys ja monipuolisten sovellusten lisääntyminen näkyy kasvaneena internetin käyttönä. Lähes joka kymmenes käyttäjä ei käytä älypuhelimella internettiä. Haltijoista n. 75 % käytti älypuhelintaan internetyhteyksiin viikoittain. [9.]

2.1.2 Internetin käyttö tablettitietokoneella

Vuonna 2017 noin 45 % 16–89-vuotiaasta suomalaisista oli käyttänyt viimeisten 3 kuukauden aikana tablettitietokoneella internetiä. Tablettitietokoneiden käyttö kotitalouksissa on kasvanut jatkuvasti vuodesta 2013 (Kuvio 2). [9.]



Kuvio 2 Kotitaloudet, joissa on käytössä tablettitietokone 2013-2017, %-osuus väestöstä [9]

Vuonna 2017 Tilastokeskuksen tutkimuksessa tutkittiin joitain internetin käyttötarkoituksia matkapuhelimella ja tablettitietokoneella. Tutkimukseen osallistuneilta kysyttiin mihin käyttötarkoituksiin he käyttävät laitteitaan: sähköpostin, e-lehtien ja e-kirjojen lukeminen, podcast, yhteisöpalveluihin osallistuminen, pelaaminen, kuvien/videoiden selailu ja katselu tai musiikin kuuntelu, sekä verkko-ostaminen. Tutkimuksen viiteaikana oli kolme kuukautta. [9.]

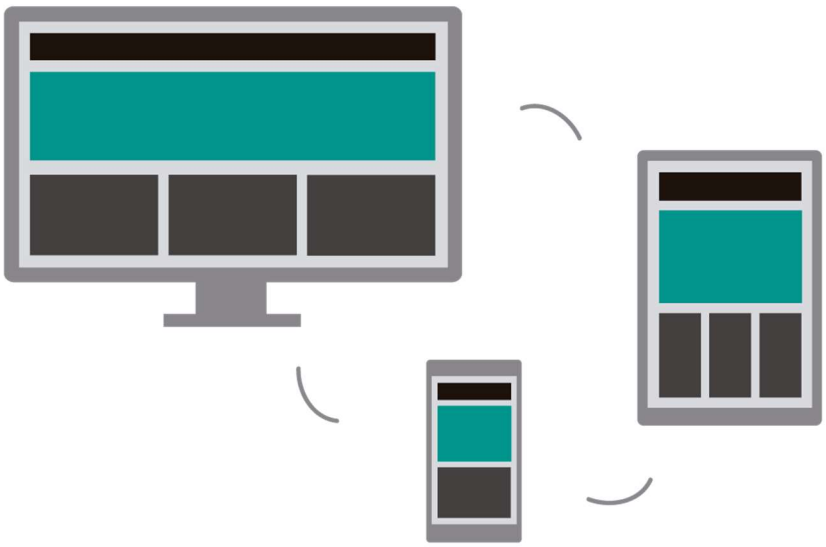
Kun vertaa tablettitietokoneen käytön osalta vuoden 2015 ja 2017 tuloksia, eivät ne juuri eroa toisistaan. Tabletteja käytetään korkeintaan hieman vähemmän edellä mainittuihin käyttötarkoituksiin, kun taas matkapuhelimen käyttö eri käyttötarkoituksiin on sen sijaan lisääntynyt. Vuonna 2017 matkapuhelimien käyttö oli noin kaksi kertaa yleisempää kuin vastaava tablettitietokoneen käyttö. [9.]

2.1.3 Responsiivisuus

Kuten aikaisemman luvussa näimme (Kuvio 1), kosketusnäytöllisten mobiililaitteiden käyttö on kasvanut runsaasti viime vuosien aikana. Tästä voimme päätellä, että monet käyttävät palveluita myös älypuhelimen kautta, jolloin sivuston tulee olla helppokäyttöinen ja siisti. Verkkosivustoja luodessa on hyvä myös huomioida, että myös tablettitietokoneiden käyttäjien määrä on lisääntynyt runsaasti (Kuvio 2).

Koska mobiililaitteiden kautta tapahtuvan Internetin käytön arvioidaan lisääntyvän huomattavasti [10], olisi hyvä huomioida käytön lisääntyvyys viestintää, sivustoja ja erilaisia

palveluita suunniteltaessa. Monessa tapauksessa on hyödyllistä, että sivustoja voi katsella mobiilipäätelaitteella. Aiemmin monista sivustoista on tehty vastaavat erilliset mobiilisivustot tai natiivisovellus eri mobiilialustoille. Tällöin joudutaan ylläpitämään useampaa sivustoa tai sovellusta samaan aikaan, joka on aikaa vievää ja vaatii eri alustojen asiantuntijuutta. Responsiivisesti toteutettu sivusto mahdollistaa sen, että verkkosivu skaalautuu eli mukautuu eri päätelaitteilla katsottaessa sopivaan kokoon. Tällöin käyttäjälle näytetään hänen päätelaitteelleen optimoitua internetsivustoa, eli jos käyttäjä käyttää tablettitietokonetta, niin verkkosivusto optimoituu tablettitietokoneen näyttöön sopivan kokoiseksi. Myös sivuston asettelu asettuu käyttäjän laitteelle sopivaksi, jolloin sisältöä on helppo lukea. (Kuva 1). [11.]



Kuva 1. Sivuston asettelun responsiivisuus eri laitteilla [12]

Responsiivinen sivusto toimii nopeammin ja parantaa käyttäjä kokemusta. Kun kehitetään mobiilisivut ensin, painotus keskittyy tehokkuuteen. Responsiiviset sivut on myös tulevaisuutta katsoen pitkäkestoisempia. Jokaisen sivun tulee aikanaan olla optimoituja uudelle teknologialle, mutta kun rakentaa responsiivisesti varmistetaan, että optimoinnista ei tule mukanaan täyttä sivuston uudelleen suunnittelemista. [13.]

2.2 Saavutettavuus

Saavutettavuus tarkoittaa esteettömyyttä digitaalisessa maailmassa: sitä, että erilaisten ihmisten on helppoa käyttää verkkopalveluja ja niissä olevia sisältöjä. Verkkoympäristössä kohde on helposti lähestyttävissä kaikille ihmisille. [14.] Hyvällä saavutettavuudella parannetaan ihmisten yhdenvertaisuutta digitaalisessa yhteiskunnassa [15].

Saavutettavuutta voi myös ajatella näkökulmana, joka kiinnittää huomiota erilaisiin käyttäjiin ja heidän moniin erilaisiin tilanteisiinsa, tarpeisiinsa ja mahdollisiin rajoiteisiin tai haasteisiin. Saavutettavuus onkin hyvin keskeinen osa asiakaslähtöisyyttä. [15.] Esimerkiksi verkkopankkisivusto on saavutettava silloin, kun kaikki käyttäjät voivat vaivattomasti, ilman muiden apua, asioida verkkopankissa [14].

Saavutettavan verkkopalvelun suunnittelussa ja toteutuksessa tulisi huomioida

- tekninen toteutus
- selkeys ja ymmärrettävyys
- helppokäyttöisyys.

Teknisesti hyvin toteutettu verkkopalvelu on lähdekoodiltaan selkeää ja loogista. Siinä noudatetaan HTML-standardeja sekä WCAG-ohjeistusta. Palvelu toimii hyvin erilaisilla päätelaitteilla ja avustavilla teknologioilla, kuten puheenohjauksella ja ruudunlukuohjelmalla. Helppokäyttöinen verkkopalvelu on helppo hahmottaa ja navigaatio on selkeä, ja etsitty sivu, toiminto tai sisältö löytyy ilman vaivaa. Verkkopalvelun pitäisi olla yksinkertainen, ei monitasoinen. Myös sivujen nimeäminen vaikuttaa merkittävästi navigaation helppokäyttöisyyteen, joten sivut tulisi nimetä kuvaaviksi ja selkeiksi. Myös pääsisällön tulisi erottua selkeästi muista elementeistä. [15.]

Verkkopalvelun sisällön saavutettavuudella tarkoitetaan kognitiivista saavutettavuutta. Tämä tarkoittaa sitä, että verkkosivuston sisältöä on helppo ymmärtää, omaksua ja käyttää. Saavutettavuutta voidaan parantaa mm. selkokielisellä, selkeällä yleiskieltä. Myös sisällöin hyvällä ja selkeällä kokonaisuuksiin jaksottamisella voidaan parantaa saavutettavuutta. [14.]

2.2.1 Typografia

Typografian saavutettavuusnäkökulmasta digitaalinen typografia on helposti muokattavissa, kopioitavissa ja siirrettävissä. Sitä voi lukea näytönlukulaitteilla, tai pistekirjoitusnäytön avulla. Digitaalisen tekstin etuja on myös se, että tekstiä voi vapaasti suurentaa ja pienentää, taikka muuttaa värejä ja kontrasteja. Digitaalisen typografian käytössä on joitakin ongelmia, kuten resoluutio, fontit ja näyttölaitteen fonttiasetukset. Jos tekstin esittää kuvana, niin jotkin näistä ongelmista poistuvat. Kun tekstisisältö näytetään kuvana, se aiheuttaa omat ongelmansa ja koska digitaalinen kuva ei ole tekstiä se poistaa digitaalisuuden edun. Kuvien muokkaaminen ja suurentaminenkaan ei ole niin yksinkertaista ja helppoa, kuin tekstin kanssa on. Jotta sivusto olisi mahdollisimman saavutettava, tulisi kuvien käyttöä tekstin sijaan ehdottomasti välttää. [16.]

Päätteetön (sans-serif) fontti on suoraviivainen ja tasainen. Kirjaimissa ei ole koukeroita eikä paksunnettuja kohtia. Päätteetön fontti on usein kirjainten välistykseltään avara ja näytöllä tarkka ja käyttäjälle helppolukuinen [16]. Päätteettömät fontit ovat lukijaystävällisempiä, ja esimerkiksi lukihäiriöisten henkilöiden on helpompi saada selvää päätteettömästä tekstistä. Tästä syystä esittelymallin fontit ovat Arialin päätteettömiä fontteja. Myös Verdana tai Trebuchet ovat hyviä sans-serif leipätekstin fontteja. Huomiota pitää kiinnittää fontin kirjaimien ja numeroiden väleihin ja fontti koko varsinkin dokumenteissa olisi hyvä olla 11 tai 12 pt tai fontin mukaan suurempi. Jos tekstissä käytetään värejä, olisi hyvä huomioida tekstin ja taustan välinen kontrasti. Kontrastin tulisi olla riittävän suuri, eli esimerkiksi painike vaalealla taustavärillä ja tummalla tekstillä. Jos käyttää värejä toisin päin, eli tummalla pohjalla ja vaalealla fontilla, tulee olla hyvä huolehtia, että fonttikoko on riittävän suuruista. [17.]

Jotta verkkosivustot olisi säädöksien mukaisesti saavutettavat, tulisi sivustolla olla fontin koon muokkausominaisuus, sekä kontrastin säätö. Esittelymallissa nämä ominaisuudet eivät ole toiminnallisia.

2.2.2 Värimaailma

Väri on tärkeä työkalu WWW-sisällön suunnittelussa. Väreillä voidaan tehostaa sivuston esteettistä vetovoimaa, käytettävyyttä ja saavutettavuutta.

WCAG 2.0 -ohjeistuksessa on joitain kriteerejä tummuuksista ja värienkäytöstä. Muun muassa värit eivät saa olla ainoa tapa antaa käyttäjille heidän tarvitsemansa tieto, koska joillakin käyttäjillä on vaikeuksia havaita värejä. [18] Ihmiset, joilla on osittainen näkökyky, omaavat usein rajoitetun värinäön tai heikentyneen näkökyvyn. Ihmiset, jotka käyttävät vain tekstiä tai heille sovitettuja näyttöjä ja selaimia, tarvitsevat sivuston, jonka informaation saanti ei perustu vain väreihin. Esimerkiksi kun käyttäjä täyttää lomaketta ja lomakkeen täytössä epäonnistuu jokin asia, pitäisi epäonnistumisesta tulla ilmoitus myös tekstinä tai symbolina eikä pelkästään värillä. Eli väriin perustuvilla toiminnallisuuksilla ja ominaisuuksilla tulisi olla myös vaihtoehto, johon ei tarvita värien havaitsemista. WCAG-kriteeristössä ei kielletä värikästä ulkoasua, mutta on tärkeää huomioida käyttäjien rajoitteet. Suomessa värisokeiden osuus on jopa 6–7 prosenttia väestöstä. [18,19.]

Jotta heikkonäköiset saisivat tekstistä sujuvasti selvää, tulisi taustan ja tekstin välillä olla hyvä ja selkeä tummuusero eli kontrasti. Kontrastista on muutenkin hyötyä esimerkiksi kirkkaassa auringonpaisteessa. Tekstin ja taustan välisen kontrastisuhteen pitää olla vähintään 4,5:1, jolloin kontrasti täyttää kriteeritason AA. Mustan ja valkoisen värin kontrastiero on 21:1. Pienempi kontrasti riittää, jos teksti on suurta kokoa, esimerkiksi vähintään 18 pt tai lihavoituna 14 pt. On tärkeää kiinnittää huomiota myös siihen, että kontrasti säilyy, kun linkin tekstin tai taustan väri muuttuu, silloin kun hiiri viedään sen päälle. [18.]

Kontrastin voi tarkistaa saavutettavuuden tarkistukseen luoduilla työkaluilla, kuten Webaimin Wave-työkalulla, joka näyttää kerralla sivuston kaikki tekniset saavutettavuusongelmat.

Opinnäytetyössä on käytetty toimeksiantajan omaa Parasta Palvelua graafista ohjetta, jossa on fontit ja värit valmiiksi määriteltyinä. Olisi ollut myös mahdollisuus käyttää värejä melko luovasti ja uusia värejäkin. Nykyisen graafisen ohjeen värimaailman koettiin hyväksi pienillä muokkauksilla. Esimerkiksi tummanharmaan sävystä on tehty hieman tummempi. Painikkeissa, linkeissä ja esimerkiksi kentissä on taustaansa nähden hyvät kontrastit ja niissä on kuvaava teksti.

3 TEKNOLOGIAT JA KOODIKIELET

Tämän esittelymallin toteutuksessa on käytetty useita erilaisia teknologioita. Teknologiat ovat melko ketteriä ja toimivat hyvin yhdessä. Visual Studio Code -tekstinkäsittelyohjelmalla pystyy kirjoittamaan kaikkia toteutuksessa käytettyjä ohjelmointikieliä. Codessa on myös lisäosa, jonka avulla tekstinkäsittelyohjelman kautta pystyy kommitoimaan käsiteltävän ohjelmistokoodin suoraan SourceTreen (versionhallinnan työkalun) kautta BitBucketiin. VS Codessa on myös oma terminaalinsa, jonka kautta pystyy tekemään komentoja esimerkiksi käynnistääkseen palvelimen. Tässä luvussa kerrotaan ohjelmointikielistä ja muista käytetyistä teknologioista.

3.1 HTML 5 -merkkauskieli

HTML on HyperText Markup Language on merkkauskieli, jota käytetään verkkosivujen luomiseen. HTML on kaikkein yleisimmin käytetty ohjelmointikieli verkkosivujen kehittämisessä. Se rakentuu perättäisistä, sekä sisäkkäisistä elementeistä [20]. Elementteinä toimivat tagit, jotka ovat hakasulkeiden sisällä. Tagien väliin laitetaan haluttu sisältö. Esimerkiksi <h3> tagi merkkaa otsikkoa, jonka teksti saadaan näkyviin kirjoittamalla otsikon teksti tagien <h3> ja </h3> väliin seuraavasti: <h3>Otsikko</h3>.

HTML kieltä voidaan kirjoittaa esimerkiksi tekstieditorissa .html -tiedostomuodossa. Tagit kirjoitetaan puumallin mukaisesti, sillä selaimet lukevat koodia järjestyksessä ylhäältä alaspäin. HTML 5 on HTML-merkintäkielen uusin versio.

3.2 CSS 3 -tyyli

Cascading Style Sheets (CSS) on yksinkertainen keino lisätä ulkoasullisia asioita, kuten fontteja, värejä, sijoittelua tai välityksiä nettisivuihin. CSS:lla voi esimerkiksi tyylitellä HTML elementtien värityksiä (Kuva 2).

Kuva 2 Otsikon väri on asetettu vihreäksi

Yllä olevassa esimerkissä on css-kielellä määritetty elementin <h3> väri vihreäksi, joka näkyy selaimessa vihreänä otsikkona. Selain lisää CSS -tyylimäärittelyn koodissa määritettyihin elementteihin näyttääkseen ne selaimessa. Tyyli-määrittely sisältää ominaisuudet ja niiden arvot, jotka määrittävät verkkosivun ulkoasun. CSS on yksi ydin teknologioista verkkosivun kehityksessä HTML:n ja JavaScriptin lisäksi. [21.]

3.3 JavaScript-kieli

JavaScript on ohjelmointikieli, jota voi käyttää lisätäkseen verkkosivustolle erilaisia käyttäytymisiä. Sitä voidaan käyttää validoimaan käyttämääsi dataa lomake muotoon, tehdä ”raahaa ja pudota” (engl. drag and drop) -toimintoja, animoida sivun elementtejä, kuten menuja, painikkeita, ja paljon muita asioita. JavaScript toimii etsimällä HTML elementtejä ja tekemällä tälle elementille jotain. Se toimii aivan kuten CSS, mutta tapa millä se toimii, on melko erilainen. [20.]

3.4 Bootstrap -framework

Työn responsiivisuuden saavuttamisen apuna on käytetty Bootstrapia, joka on erittäin hyvä työkalu niin sanotulle ”Mobile-first” -suunnittelulle. Responsiivinen verkkosivu olisi tärkeä ominaisuus PSOP-järjestelmälle, koska kuten luvussa 2.1.1 ja 2.2.2 kerrottiin lyhyesti tilastokeskuksen tutkimuksista mobiili- ja tablettilaitteiden internetin käytöstä, on mobiililaitteilla internetin käyttö kasvanut viimevuosina hyvin paljon. Koska käyttäjien internetin käyttö mobiililaitteilla on kasvanut, voisimme myös ajatella, että yhä useampi haluaisi pystyä käyttämään palvelua mobiililla kin. Tämän takia olisi suotavaa, että järjestelmä olisi responsiivinen niin mobiilille, kuin tabletillekin.

Bootstrap on kaikkein suosituin ilmainen avoimen lähdekoodin työkalu responsiivisen sivuston kehittämiseen HTML, CSS ja JavaScript koodilla [22]. Se sisältää valmiita CSS-

tyylimäärittelyjä sekä lisäksi myös paljon JavaScript- ja JQuery-funktioita [23]. Bootstrap on kehitysympäristö, jonka avulla voidaan luoda responsiivisia sivustoja. Bootstrapilla pystyy nopeasti luomaan prototyyppejä tai rakentamaan kokonaisen sovelluksen. [24.] Bootstrap on responsiivinen ja ”mobile-first” suunnittelua noudattava ja keskittyy yksinkertaistamaan informatiivisen internetsivuston kehittämistä. Bootstrap toimii myös kaikilla selaimilla. [22,25.]

Esittelymallissa on käytetty Bootstrappia sen helppokäyttöisyyden, ja yksinkertaisen, mutta modernin ulkoasullisen ilmeen takia, minkä myötä on mahdollista kirjoitta hyvin vähän CSS-koodia. Bootstrapissa on myös valmiita apuluokkia, joiden avulla välttyy suurelta CSS-koodin kirjoittamiselta. Bootstrap tarjoaa hyötyä myös elementtien asetteluun ja muotoiluun. Opinnäytetyössä ei perehdytä syvemmin Bootstrapin toimintoihin, koska tässä tapauksessa ei rakenneta kokonaista sivustoa. Työssä on hyödynnetty paljon Bootstrapin sivujen kautta löytyviä selkeitä ja kattavia esimerkillisiä ohjeita. Koska esittelymalli on melko pieni, on Bootstrap hyvä ratkaisu, mutta jos työ olisi isompi, voisi työmäärä olla melko raskas Bootstrapin koodi-intensiivisyyden vuoksi.

3.5 Atlassian työkalut

Esittelymallin kehityksen dokumentaatioon on käytetty Atlassianin Confluencea, versionhallintajärjestelmänä käytettiin BitBucketia sekä versionhallinnan käyttöliittymänä SourceTree -ohjelmaa. Työn ja tehtävien seurantaan on käytetty Jiraa ja Trelloa. Näiden avulla kaikki tarvittava data oli keskitettynä ja tarvittaessa helposti saatavilla.

Versionhallintajärjestelmä eli VCS on apuohjelma, joka seuraa ja hallinnoi tiedostoihin ja koodiin tehtyjä muutoksia. Tiedostojärjestelmätasolla se seuraa tiedostoissa ja hakemistoissa tapahtuvia muutos-, poisto- ja lisäystoimintoja. VCS tekee saman lähdekooditiedostojen tekstirivien kanssa. Suosittuja versionhallintajärjestelmiä ovat mm. Git, Mercurial, SVN ja Perforce. Tässä työssä on käytetty VCS Git:iä. [26.]

Jokaisella tiimin jäsenellä on omalla tietokoneellaan lokaali tietovarasto (engl. repository). Tietovarastossa on mahdollisesti etätietovarastosta (engl. remote repository) saadut tiedostot. Kun tiimin jäsen muokkaa ohjelmistokoodia omalla tietokoneellaan hän tallentaa muutokset omalle tietokoneelleen tietovarastoonsa. SourceTreen avulla jäsen pystyy helposti kommitoimaan (engl. to commit) tietovarantonsa etätietovarantoon, josta

muut jäsenet saavat ohjelmistokoodin muutokset ja muutosjoukot (engl. changeset) itselleen. Versionhallintajärjestelmää on hyvä käyttää ohjelmistoprojektissa, jossa useampi tiimin jäsen tekee muutoksia ohjelmistokoodiin yhtäaikaisesti. Tämä on siksi, että VCS seuraa kehittäjien välisiä ristiriitoja ja auttaa niissä. Kehittäjien on helppo seurata mitä on tehty missäkin vaiheessa VCSn luoman jäljitysketjun vuoksi. Myös mahdolliset bugit ja vikakoodit löytyvät helpommin. VSC säilyttää muutoshistorian ja lähdekoodin tilan koko projektin ajan. Tämän takia on mahdollista kumota tai palauttaa lähdekoodin tila viimeksi tunnettuun tilaan. Esimerkiksi jos sovelluksessa on virhe, voidaan se palauttaa aiemmin toimivaksi todettuun tilaan. [26.]

SourceTree yksinkertaistaa vuorovaikutusta Git-arkistojen kanssa. Sillä pystyy hallinnoimaan ja visualisoimaan arkistoja [27], jolloin ei tarvitse kirjoittaa komentoja komentokehoteikkunaan. SourceTreen kautta pystyy lähettämään koneella tehdyt muutokset Git-arkistoon. Tässä työssä on käytetty BitBuckettia, joten tiedostot tallentuvat sinne.

3.6 Palvelin ja tietokanta

Esittelymalli saatiin tehtyä sovellukseksi palvelimen ja MongoDB -tietokanta yhteyden avulla. Tällöin työstä saatiin toiminnallinen ja kykenevä hakemaan dataa tietokannasta. Tarvittavat tiedot talletetaan tietokantaan, ja Nodella tehtyyn sovellukseen luodaan koodi, jolla noudetaan tietoja tietokannasta sovelluksen käyttöön.

3.6.1 Node.js -palvelin

Node.js on ilmainen avoimen lähteen palvelinympäristö, joka toimii useilla eri alustoilla (Windows, Mac OS X, Linux, Unix jne.) ja käyttää JavaScriptiä palvelimella [28]. Se on hyvin tehokas JavaScript -pohjainen alusta, jota käytetään paljon käyttäjältä syötettä vaativien internet sovellusten kehittämiseen. Node.js on järjestelmäriippumaton runtime-ympäristö palvelinpuolen ja verkkoratkaisujen kehittämiseen, jossa applikaatiot kirjoitetaan JavaScriptillä. Sillä on myös rikas kirjasto erilaisia JavaScript -moduuleita, jotka yksinkertaistavat applikaation kehitystä merkittävissä määrin. [28,29.]

Node.js omaa muutamia erittäin tärkeitä ja hyviä ominaisuuksia. Ensinnäkin kaikki Node.jsn kirjaston APIt ovat asynkronisia, eli ei-estäviä. Tämä tarkoittaa, että Node.jsllä

tehty serveri ei odota APIn palauttavan dataa. Serveri siirtyy seuraavaan APIin sen jälkeen, kun on kutsunut aikaisempaa APIa ja Events of Node.js auttaa serveriä saamaan aiemman API kutsun vastauksen. Koska Node.js on rakennettu Google Chromen V8 JavaScript moottorilla, niin Node.jsn kirjasto on hyvin nopea toteuttamaan koodin ajoa. Node.js on suunniteltu skaalattavien nettiapplikaatioiden rakentamiseen. Se on samankaltainen muotoilultaan, ja on ottanut vaikutteita järjestelmiltä kuten Rubyn Event Machine tai Pythonin Twisted. Esittelymallissa käytettiin Nodea sen monimuotoisuuden ja laajan yhteensopivuuden kanssa. Node.js ja MongoDB ovat sopivat toisilleen mainiosti. [29,30.]

3.6.2 MongoDB -tietokanta

Tietokanta on fyysinen säilytystila erilaisille kokoelmille. Sinne kerätään dataa, joita sittemmin voi hakea tietokannasta järjestelmään ja järjestelmästä tallettaa tietokantaan. Yksittäisellä MongoDB-palvelimella on tyypillisesti useampia tietokantoja. [31.]

MongoDB on järjestelmäriippumaton ja skeematon NoSQL tietokanta, johon säilötään JSON-tyyppistä dokumentteja (Kuva 3). Tästä syystä MongoDB on hyvin joustava ja skaalautuva hajautettu tietokanta. Kokoelmat ovat MongoDB dokumenttien ryhmiä, joiden dokumentit voivat sisältää erilaisia kenttiä. Tyypillisesti kaikilla kokoelman dokumenteilla on sama tai toisiinsa liittyvä tarkoitus. [31.]

```
{
   _id: ObjectId(7df78ad8902c)
   title: 'MongoDB Overview',
   description: 'MongoDB is no sql database',
   by: 'tutorials point',
   url: 'http://www.tutorialspoint.com',
   tags: ['mongodb', 'database', 'NoSQL'],
   likes: 100,
   comments: [
      {
         user:'user1',
         message: 'My first comment',
         dateCreated: new Date(2011,1,20,2,15),
         like: 0
      },
      {
         user:'user2',
         message: 'My second comments',
         dateCreated: new Date(2011,1,25,7,45),
         like: 5
      }
   ]
}
```

Kuva 3 Esimerkki MongoDB-dokumentista [31]

Dokumenteilla on oma dynaaminen skeema, mikä tarkoittaa sitä, että samassa kokoelmassa olevat dokumentit eivät tarvitse samoja kenttiä tai rakenteita. Tällöin kokoelman dokumenttien yleiset kentät voivat siis sisältää erityyppisiä tietoja, kun taas SQL -tietokannoissa on eri taulukot erityyppisille tiedoille, jotka linkitetään toisiinsa (Kuva 4). Tämä on yksi NoSQL -tietokannan merkittävistä eduista, koska se nopeuttaa sovelluskehitystä ja vähentää käyttöönoton monimutkaisuuta. [31,32.]

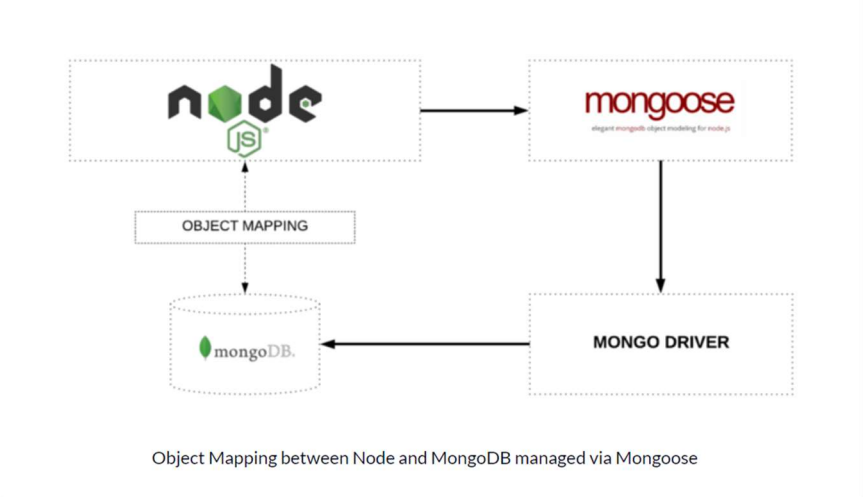
Kuva 4 Esimerkki SQL datan tallettamismuodosta [32].

Monesti dokumenteilla on sisäkkäisiä dokumentteja, joihin viitataan niiden omalla ObjectId-tunnuksella (_id-kenttä). MongoDB käyttää Objectid-tunnuksia kunkin asiakirjan _id-kentän oletusarvona. Tämä Objectid-tunnus luodaan automaattisesti monimutkaisista merkkiyhdistelmistä, minkä tahansa asiakirjan luomisen aikana. Monimutkaisten yhdistelmien vuoksi jokainen _id-kenttä on yksilöllinen. ObjectId on 12-tavuinen BSON-tyyppi, joka rakentuu neljästä eri asiaa edustavasta merkkijonosta. Neljä ensimmäistä tavua kuvastaa aikaa. Kolme seuraavaa tavua kuvaavat laitteen tunnistusta, seuraavat kaksi tavua sisältää prosessin id:n ja viimeiset kolme tavua ovat satunnaisia lukuja. [33.]

Koska esittelymalli on suppea ja tarvitsee vain vähän ja pientä dataa, on MongoDB sopiva valinta esittelymallin tietokannaksi. Työssä käytetään MongoDBtä, koska sitä on helppo käyttää ja se pystyy välttämään tiukkoja skeemoja. Tällöin saman kokoelman sisään voi tallettaa monenlaisia dokumentteja. MongoDB sopii myös hyvin yhteen Node.js:n kanssa, koska MongoDBlla on hyvä wrapper-paketti Mongoose. [31.]

Mongoose on Object Data Modeling (ODM) kirjasto MongoDBlle ja Node.jslle. Käyttäjä voi mongoosella hallinnoida datojen välisiä suhteita, määrittää tietokantaan talletettavan datan rakenteen ja suorittaa MongoDBn omia operaatioita (Kuva 5). Eli Mongoosella

määritellään skeema, joka kuvaa MongoDB-kokoelmaa ja määrittelee tämän kokoelmassa olevien asiakirjojen muodon. Mongoosella hallitaan JavaScriptin kautta. [32,34.]



Kuva 5 Kartoitus Noden ja MongoDBn välillä, Mongoosella hallinnoituna [32.]

4 TOTEUTUS

Esittelymallin ulkoasun suunnittelu käynnistyi sujuvasti, koska Parasta Palvelua palveluseteli- ja ostopalvelujärjestelmästä on tehty jo aiemmin graafinen ja animoitu esittelymalli Adobe XDlla. Näiden molempien esittelymallien graafisena ohjeena toimi PSOP:lla jo valmiiksi oleva graafinen ohje, jossa on määritetty värimaailmat ja fontit sekä muita ulkonäöllisiä seikkoja. Jotta työn eteneminen ja kerronta avautuisi hieman lukijalle, on alkuun kerrottu hieman aiemmasta graafisesta esittelymallista ja siitä mitä eroja näillä kahdella esittelymallilla on niin saavutettavuudessa, käytettävyydessä, kuin ulkoasullisestikin. Myös joissain kappaleen kohdissa esittelymallia vertaillaan tällä hetkellä käytössä olevaan PSOP-järjestelmään.

Työn toteutuksen alussa esittelymallia lähdettiin rakentamaan front-end puolelta, kunnes työhön asennettiin Node.js. Tällöin koodista luodaan sovellus. Jotta sovellus toimisi hyvin, niin sovellukseen tuli yhdistää myös tietokanta. Sovelluksen luonnin jälkeen työtä alettiin rakentamaan uudelleen, forEach looppien ja regexin kanssa. Html-tiedostot muutettiin .ejs -tiedostoiksi. Ejs on yksinkertainen malli, jonka avulla voi luoda HTML-merkintöjä JavaScriptillä. Myös kansiorakenne muuttui hieman, sillä tiedostot piti rakenteellistaa Node sovellukselle sopivaksi, jotta ohjelma toimisi oikein. Työn versionhallintaan ja työtehtävien hallintaan on käytetty Atlassianin työkaluja: BitBucketia sekä Jiraa ja Trelloa. Kun työn runko oli rakennettu, alettiin työstämään sisältöä, vertailua, hakutuloksia, lisätietosivua ja muita sivuja, joita sovelluksen luonnin prosessissa esitellään.

4.1 Muutokset graafiseen esittelymalliin

Aikaisemmin keväällä 2019 luodussa graafisessa esittelymallissa tehtiin uusi malli koko PSOPin palveluntuottajanhausta ja tuloksista. Aikaisempaan graafiseen esittelymalliin verrattuna saavutettavuutta ja käytettävyyttä on paranneltu yksinkertaistamalla ja lisäämällä ohjelmistokoodiin aria-label koodirivejä, jotka auttavat näytönlukulaitteiden lukemista.

Sivuston yläpalkki ja sisältö on koko näytön laajuudella. Myös skaalaus on realistisempi, kuin graafisessa esittelymallissa. Myös yläpalkin linkit on pidetty samoina, mutta hieman lyhyemmiksi ja ytimekkäämmiksi muokattuina. Esittelymallien väreinä on käytetty pääsääntöisesti tummanharmaata, jossa on valkoista tai vihreää tekstiä. Huomiovärinä on

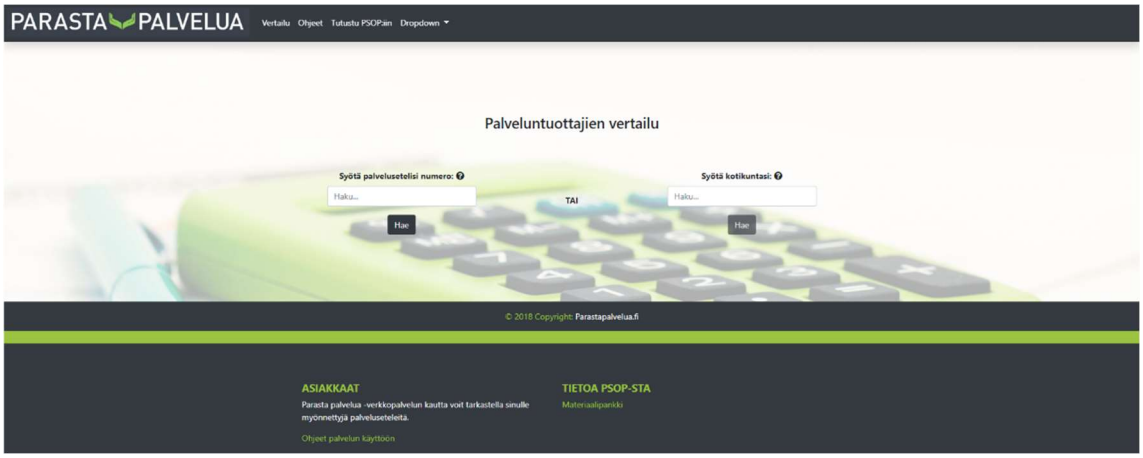
käytetty oranssinpunaista alleviivattuna. Tällöin tekstit erottuvat selkeästi tummasta taustasta, eikä punainen ja vihreäkään sekoitu niin helposti keskenään. Graafisessa esittelymallissa palkin väri on hieman vaaleampaa harmaata ja teksti vihreää, jolloin teksti erottuu hieman heikommin taustastaan. Uudemmassa esittelymallissa on selkeämmät kontrastit, jotka on testattu Wave -ohjelmalla.

Graafisen esittelymallin yläpalkki on välilehtilinkeiltään samankaltaisempi nykyisen PSOP-järjestelmässä olevan yläpalkin kanssa. Linkkien tekstit ovat pitkiä ja vihreällä, jolloin ne eivät erotu kontrastiltaan riittävän hyvin taustastaan nähden, ja jolloin ne voivat olla hieman raskaita lukea (Kuva 6). Nykyinen PSOP -sivusto on pelkistetty ja melko vaalea. Esimerkiksi yläpalkin tausta on vaalea ja teksti vihreää. Waven saavutettavuuden tarkistamisohjelmalla kontrastit eivät saavuta WCAG-kriteeri tasoja AA eikä AAA. Myöskään graafisen esittelymallin yläpalkki ei saavuttanut kumpaakaan kontrasti tasoa.

Kuva 6 Graafisen esittelymallin ulkoasu

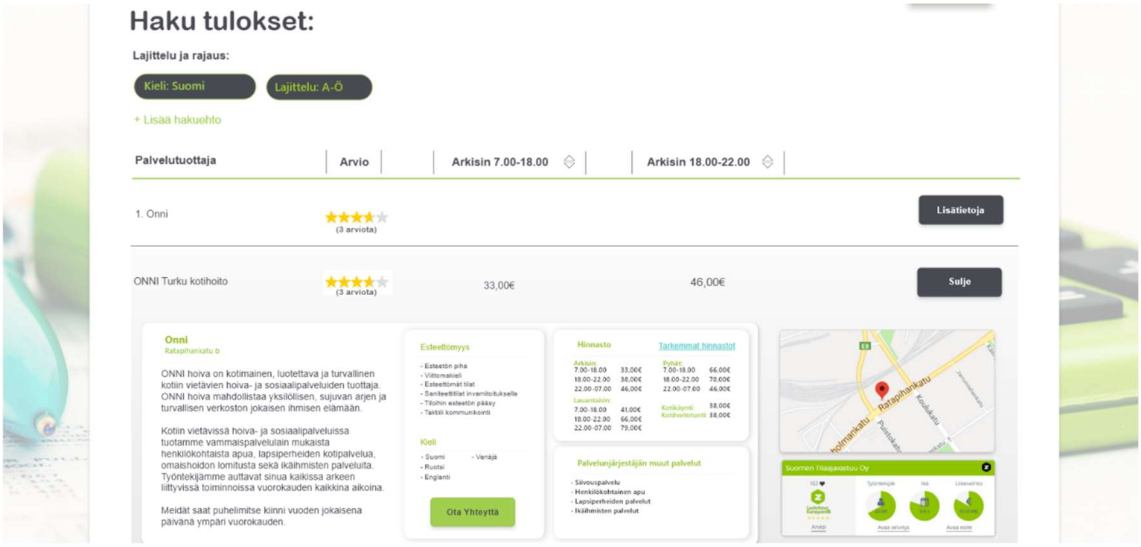
Uudessa esittelymallissa ei ole taustaa hakupalkkien alla, mikä tuo sivustolle modernin ilmeen (Kuva 7). Koodatussa esittelymallissa kunnalla hakemisen toiminto on otettu pois käytöstä, koska tämän esittelymallin päätehtävä on demonstroida palvelusetelin nume-

rolla hakua. Alatunniste (engl. footer) on tehty graafisen esittelymallin pohjalta samanlaiseksi. Sovelluksessa, eli esittelymallissa, yläpalkki ja painikkeet saavuttavat kontrasti kriteerien molemmat tasot AA ja AAA.



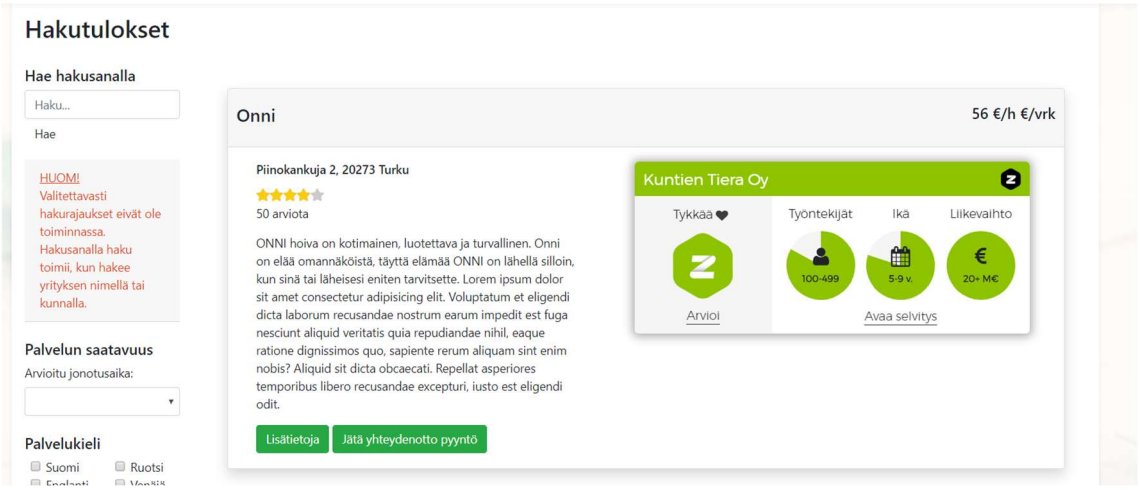
Kuva 7 Koodatun esittelymallin ulkoasu

Kun hakukenttään kirjoittaa palvelusetelin numeron, ohjelma hakee tietokannasta seteliä vastaavat tuottajat ja listaa ne hakutuloksiin. Graafisen ja koodatun esittelymallin visuaaliset ilmeet eroavat toisistaan huomattavasti. Graafisessa versiossa on taulukkomuotoisena hakutulokset, joista saa avattua lisätietoja palveluntuottajakortin ”Lisätietoja” painikkeella (Kuva 8). Kyseisen palveluntuottajan taulukon rivi laajenee niin, ja näkyviin tulee palveluntuottajan kortti. Kortissa näkyy yrityksen esittely, yhteyshenkilö, hinnastoa ja muita tietoja kuten kieli ja esteettömyystiedot. Adobe XD:llä tehty graafinen esittelymalli ei tosin ole täysin realistisesti skaalautuva ja tekstit olisivat liian pienellä. Tämä on osasyy, minkä takia hakutuloksien mallia muutettiin sen lisäksi, että hakutuloksista haluttiin mallintaa muunkinlainen vaihtoehto, kuin mitä tähän mennessä on esitetty. Myös saavutettavuuden näkökulmasta ohjelmoituun esittelymalliin luotiin erillinen lisätietosivu.



Kuva 8 Graafisen esittelymallin hakutulokset

Uuden, ohjelmoidun esittelymallin hakutulokset ovat yksinkertaiset, eikä tuloksiin ole ahdettu suuria määriä tietoja tuottajasta, vaan kaikki tarvittava lisätieto löytyy palveluntuottajan lisätietosivulta (Kuva 9).



Kuva 9 Ohjelmoidun esittelymallin hakutulokset

Uudessa esittelymallissa on mallinnettu uudenlainen idea, jossa tuottajat näytetään korteittain. Tässä mallissa palveluntuottajan lisätiedot ja tarkemmat hinnastot saataisiin palveluntuottajan infosivulta, johon pääsee hakutuloksien kortissa olevasta ”Lisätietoja” -

painikkeesta. Kortissa on myös ”Jätä yhteydenottopyyntö” -painike, jonka avulla asiakas voi jättää yhteydenottopyynnön palveluntuottajalle.

4.2 Kansiorakenne

On tärkeää, että kansiorakenne on oikeanlainen, jotta Node.js toimisi oikein. Hyvä kansiorakenne myös helpottaa tiedostojen löytämistä. Node.js:n asennuksessa ohjelmisto luo tiedoston app.js, joka pidetään juurikansiossa. App.js -tiedosto on käynnistystiedosto, joka tarvitaan, jotta saadaan node.js -palvelin käyntiin. Tähän app.js -tiedostoon on myös määritelty tietokantayhteys, jolloin data saadaan tietokannasta, eikä kovakoodattuna.

Juurikansiossa on hyvä olla joitakin tiettyjä tiedostoja käynnistys tiedoston lisäksi, kuten package.json -tiedosto, joka luodaan juurikansioon Node.jsn asennuksen yhteydessä. Package.json tiedostoon merkitään kirjastoriippuvuudet, joita projekti tarvitsee. Node.jsn mukana asennetaan myös paketinhallintatyökalu NPM, joka lataa riippuvuudet pilvestä npm install -komennolla.

Juurikansiossa on kansioita, joissa varsinainen palvelinlogiikka sijaitsee. Nämä kansiot ovat alakansiota, joista yleisiä ovat seuraavat:

- models

  Sisältää tietokantamalleja

- public

  css, kuvat, js

  Public -kansioon luodaan muutama alakansio, CSS:lle, kuville ja JavaScripteille. Ohjelma osaa hakea nämä tiedot suoraan public kansiosta.

- routes

  Tässä kansiossa on ohjelman reitityksen tiedot, ja tiedot siitä miten ja mihin sivut linkittyvät, ja mitä välisivuja on olemassa.

- node_modules

  Liitännäistenhallintasovellus npm:n kautta yleensä ylläpidettävä kansio. Sisältää sovellukselle tarpeellisten liitännäisten lähdekoodin.

- views

  Views kansiossa on sivuston eri näkymät, esimerkiksi index.ejs, infi.ejs. Niiden lisäksi tässä kansiossa on vielä yksi alikansio partials -kansio, johon tulee alatunniste ja ylätunniste tiedostot, josta eri sivut saavat haettua ne sivuilleen koodirivien avulla.

4.3 Palvelin ja tietokanta

Tietokantana on käytetty MongoDBtä, joka esiteltiin aiemmassa luvussa 3.6.2. Kun tietokanta on luotu, se voidaan yhdistää koodiin monella tapaa:

1. Voi yhdistää suoraan ohjelmaan MongoDBn luoman yhdistämistä varten luodun merkkijonon kanssa (engl. connection string). Tämä merkkijono on tässä tapauksessa seuravanlainen:

   mongodb+srv://<username>:<password>@mockup-db-kkkjc.azure.mongodb.net/test?retryWrites=true&w=majority

   Kohdat <username> ja <password> korvataan olemassa olevilla tunnuksilla.

2. MongoDB Compass on työpöytäsovellus tietokannan selailuun, hallitsemiseen ja muokkaamiseen. Sen voi asentaa koneelle ja sitä kautta yhdistää ohjelman ja tietokantaan myös samankaltaisen merkkijonnon avulla.

   mongodb+srv://<username>:<password>@mockup-db-kkkjc.azure.mongodb.net/test

3. Mongo Shell on interaktiivinen JavaScript rajapinta, jota voi käyttää datan tiedusteluun ja päivittämiseen. Mongo Shell asentuu koneelle MongoServerin asennuksen mukana. Tämänkin voi yhdistää koodiin oman merkkijonon avulla. Mongo Shellia hallitaan terminaalin kautta komentojen avulla.

Työssä käytettään suoraa yhdistämistä ohjelmasta tietokantaan. Aluksi pitää määritellä yhteys applikaation ja tietokannan välille. Yhteys määritetään seuraavasti:

```
var mongoose = require("mongoose");

mongoose.connect("mongodb+srv://<username>:<password>@mockup-db-
kkkjc.azure.mongodb.net/PSOP?retryWrites=true&w=majority")
```

Tässäkin kohdat <username> ja <password> korvataan olemassa olevalla ja tietokantaan kirjatulla käyttäjätunnuksella ja salasanalla.

Yhteyden määrittämisen jälkeen alettiin lisäämään tietokantaan palveluntuottajien tietoja mahdollisemman kattavasti. Joskus tietokantaan piti palata takaisin lisäämään tai muokkaan tietoja sitä mukaan, kun kehitettäviä ja muokattavia asioita tuli mieleen. Tärkeimpinä tietoina on yrityksen nimi, palvelut, hintatiedot, osoitetiedot ja yhteystiedot. Tietokannasta on tehty hyvin yksinkertainen, missä on yhden tuottajan tiedot samassa kokoelmassa, eikä hajautettuna. Esimerkiksi hinnastotiedot ovat samassa kokoelmassa oman palveluntuottajan tietojen kohdalla, eikä erikseen hinnastot -kokoelmassa. Tämä on myös yksi MongoDBn vahvuuksista.

4.4 Bootstrapin käyttöönotto

Jotta saataisiin Bootstrapin modernit ja responsiiviset ominaisuudet ja elementit käyttöön, koodiin pitää lisätä muutama rivi, jotka saa Bootstrapin Getting started/introduction sivulta. Koodirivit on helppo lisätä erillisinä jo olemassa olevaan skeletoniin, eli runkoon. Työn voi myös aloittaa kopioimalla Bootstrapin aloituspohjan uuteen tiedostoon. Tästä on hyvä aloittaa uusi työ, koska siinä on valmiiksi määritetty vaaditut metat, linkitykset tyyleihin ja JavaScripteihin. Tässä opinnäytetyössä on käytetty erikseen kopioitavia koodirivejä, koska työssä luodaan erilliset ylä- ja alatunniste tiedostot.

Tarvittavat koodirivit ovat CSS- ja JavaScript-kieltä. CSS-koodirivi määrittää ohjelmalle mistä haetaan Bootstrapin CSS-määritykset ja JavaScriptit, jotka sisältävät Bootstrapin komponenttien JavaScript funktiot, kuten jQuery, Popper.js ja niiden JavaScript lisäosansa.

Nämä kyseiset koodirivit lisätään sivun alkuun ja loppuun, mistä alkuun tulee CSS-koodirivit ja loppuun tulee JS-koodirivit. Jos tehdään erikseen ylätunniste- ja alatunnistetiedostot, niin nämä kyseiset koodirivit voidaan laittaa niihin, jolloin ei tarvitse kaikkia scriptejä eikä css linkityksiä lisätä jokaiseen tiedostoon erikseen. Koska tässä tapauksessa ohjelmassa käytetään .ejs tiedostoja, niin ylätunniste ja alatunniste -tiedostot linkitetään muihin tiedostoihin koodiriveillä (Kuva 10).

```
1
2   <% include partials/header %>
3
4                   ja
5
6   <% include partials/footer %>
7
```

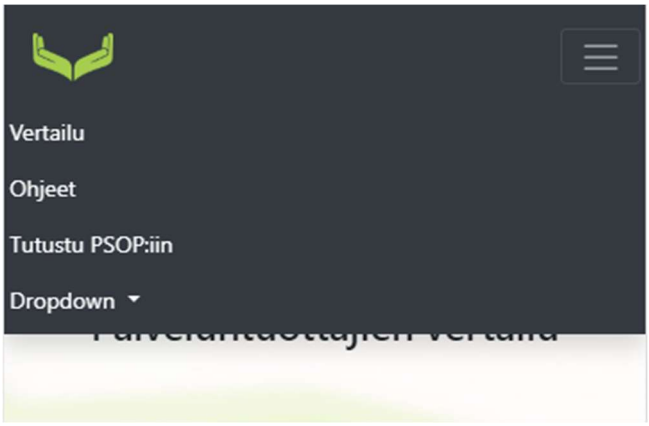
Kuva 10 Ylä- ja alaviite

Ylätunniste-tiedostoon lisätään aiemmin mainitut CSS-linkityksen koodirivit. Yllä mainittu .ejs koodirivi tulee sen tiedoston koodin yläreunaan, johon ylätunniste halutaan. Alatunnisteessa sen sijaan on JS-koodirivit ja .ejs koodirivi sijoitetaan aivan sivun alareunaan. Tällöin selain osaa lukea esimerkiksi results.ejs tiedoston kautta footer.ejs tiedostosta vaadittavat ja tarvittavat JavaScriptit. Ylätunniste (engl. header)- ja alatunniste (engl. footer) -tiedostot ovat /views-kansion alikansiossa /partials. Jotta ohjelma toimisi oikein, html-pohjan tulee olla HTML5 -tiedostotyyppiä. Siinä täytyy olla <head> elementin sisällä vaadittu ”viewport” meta tagi, joka määrittää responsiivisia käytöksiä.

4.5 Ylätunniste ja alatunniste

Työssä on tehty ylätunnisteeseen yläpalkki, jossa on yksinkertaisin sanoin nimetyt linkit, sekä Parasta Palvelua logo. Yläpalkki ja logo on rakennettu Bootstrapilla responsiivisesti. Ikkunakoon pienentyessä yläpalkki muuttaa tietyssä pysäytyspisteessä (engl.

breakpoint) muotonsa hampurilaisvalikoksi, eli linkit ja painikkeet listautuvat yhden painikkeen alle, jolloin niitä on helppo hallinnoida älylaitteellakin (Kuva 11). Myös logo muuttuu yksinkertaisemmaksi PSOPin logoksi. Myös alatunniste on responsiivinen, mikä tarkoittaa sitä, että sen sisältö ryhmittyy pienemmälle näyttökoolle sopivammaksi ja loogiseksi.



Kuva 11 Yläpalkin mobiilikoossa oleva hampurilais- valikko

Ylätunniste ja alatunniste on tehty erikseen omiksi tiedostoikseen, jotka on linkitetty muihin sivuihin. Niihin on liitetty CSS-, ja JS-linkitykset tarvittaviin tiedostoihin ja teknologioihin, kuten Bootstrapiin.

4.6 Palveluntuottajien vertailu

Tietokannan, ohjelman rungon ja etusivun rakentamisen jälkeen aloitettiin työstämään sovelluksen sydäntä, eli palveluntuottajien vertailua ja siihen kuuluvaa hakutoimintoa. Palveluseteli- ja ostopalvelujärjestelmän tarkoituksena on tarjota käyttäjilleen mahdollisuus vertailla tähän palveluun kuuluvia palveluntuottajia, joiden palveluita käyttäjät voivat sittemmin käyttää. Vertailussa vertaillaan palveluntuottajien tietoja, hintoja ja muita palveluita. Palvelupolku alkaa etusivun hakutoiminnosta valitsemalla hakutapa, hakemalla, ja seuraavaksi vertailemalla palveluntuottajia. Vertailusta voi valita palveluntuottajan ja katsoa heidän lisätietojaan, lähettää yhteydenottopyynnön tai vaikka tulostaa tiedot paperille. Kustakin vaiheesta kerrotaan, mikä vaihe on kyseessä ja miten se toimii. Luvussa kerrotaan myös hieman teknistä taustaa, kuten miten tai millä teknologioilla työ on toteutettu.

4.6.1 Haku ja hakutulokset

Etusivulla on kaksi hakupalkkia, joista voi hakea palveluntuottajia. Hakupalkeista vain palvelusetelin numerolla haku on käytössä. Haku on rakennettu käyttäen Regexiä eli Regula Expression. Regular Expressionit, eli säännölliset lausekkeet, ovat kuvioita, joita käytetään merkkien yhdistelmien yhdistämiseen merkkijonoissa. JavaScript-muodossa säännölliset lausekkeet ovat myös olioita. [35.] RegExp-rakentaja luo säännöllisen lausekeobjektin tekstin sovittamiseksi kuvioon [36].

Regular Expressionia käytetään RegExp metodeilla test ja exec sekä String metodeilla match, search ja split. Tässä työssä on käytetty String metodeista search metodia. Search metodi on siis String metodi, joka testaa osuvuuksia merkkijonossa. Se palauttaa osuvuustestin tuloksen, tai -1:n jos haku epäonnistuu. [35,36.]

Koska koodi on tehty Node.jsllä ja expressillä, joudutaan koodissa käyttämään ejs -tiedostomuotoisia tiedostoja. Ejs-tiedostojen sisältö on html koodia, mutta niihin voi lisätä .ejs komentoja, kuten: `<% include partials/header %>`, joka yhdistää kyseisen ylätunnisteen -tiedoston koodiin partials kansiosta. Tässä tapauksessa ylätunniste toimii yläpalkkina, joka halutaan toistuvan joka sivulla.

Ejs:llä voidaan tuoda JavaScriptillä tehtyä dataa html koodiin. Hakutulokset on tulostettu results.ejs sivulle forEach loopilla. ForEach() kutsuu annetun callback-funktion kerran jokaista taulukossa (engl. array) olevaa elementtiä kohden nousevassa järjestyksessä. ForEach:n avulla työssä siis tulostetaan palveluntuottajankortti niin useasti, kuin on dataa tietokannassa. Jotta saadaan tietyt datat koodiin, niin että tieto tulee tietokannasta, eikä kovakoodattuna, täytyy palveluntuottajankortin koodirivejä ennen ja niiden jälkeen olla tietynlaiset koodirivit (Kuva 12).

```
1
2   <% producers.forEach(function(producer){ %>
3       TÄHÄN TULEE TOISTETTAVA SISÄLTÖ
4   <% }); %>
5
```

Kuva 12.ejs rivit, jolla tuodaan sisältöä tietokannasta

Edellä olevien ejs, JavaScript-rivien sisälle tulee palveluntuottajan kortin koodi, elementit ja tiedot jne. Jotta saadaan tuotua kortille esimerkiksi palveluntuottajan nimi, pitää koodiin lisätä `<%=` - merkeillä alkava rivi (Kuva 13).

```
1
2   <% producers.forEach(function(producer){ %>
3         <div class="col-lg-8 p-2">
4         <h4><%= producer.name %></h4>
5         </div>
6   <% }); %>
7
```

Kuva 13 Lisätty html-sisältöä ja palveluntuottajan nimen nouto

Koodin producer tarkoittaa skeemasta haettavaa tietoa ja name tarkoittaa itse tietoa, jota haetaan, eli tässä tapauksessa palveluntuottajan nimeä. Tähän tyyliin lisätään muutkin palveluntuottajien halutut tiedot.

Skeema on aiemmin Models-kansioon luotu tiedosto, joka määrittää mongooselle mallin, jolla kukin data haetaan tietokannasta. Esimerkiksi jos haetaan palveluntuottajan nimeä, niin silloin halutaan ohjelman hakevan kyseisen tiedon stringinä, eli kirjaimina. Tai postinumeroa haettaessa määritämme postalcode: Number; koska tiedon halutaan tulevan numeroina. Jos muualla koodissa on esimerkiksi ”konttori”, mutta sitä ei ole skeemassa, niin ohjelma ei osaa tuoda kyseistä tietoa tietokannasta.

Hakutuloksiin on lisätty myös painikkeet kahdelle erityyppiselle tulostukselle. Ne on sijoitettu sivun oikeaan ylälaitaan. Painikkeet ovat tummanharmaita valkoisella tekstillä, jolloin kontrastikin on selkeä ja Wave -työkalulla tarkastettuna saavuttaa AA ja AAA kontrasti kriteerit. Tulostuspainikkeista olisi tarkoitus saada tulostettua hakutulokset kattavilla tiedoilla, kun taas ”Tulosta yhteystiedot” -painikkeesta pystyisi tulostamaan kaikkien hakutuloksien yhteystiedot listattuna. Nämä samaiset painikkeet ovat myös palveluntuottajan lisätietosivulla, jolloin käyttäjä saa tulostettu tietyn palveluntuottajan kattavammat tiedot, mm. hinnastot, palvelukielet ja muut palvelut.

Front-end-ohjelmointiosioissa on käytetty hyvin paljon Bootstrapin elementtejä ja ominaisuuksia. Muun muassa palveluntuottajan haussa tulokset ovat Bootstrapin kortteja, jotka ovat joustavia sisällön sisällytys elementtejä. Niitä on helppo käyttää ja ne ovat hyvin muokattavissa tarpeen mukaan.

Koska nykyisin olemassa olevassa PSOP-järjestelmässä on malliltaan toimivaksi todettu hakurajaus, toivottiin tähänkin versioon samanlaista rakennetta. Hakurajaukset sijaitsevat hakutuloksien vasemmalla puolella. Esittelymallissa hakurajaukset eivät ole toiminnallisia, mutta antavat ulkoasullista ohjeistusta, siitä miltä ne voisivat näyttää ja mitä kaikkia tietoja voisi olla tarpeellista rajata. Ainoastaan rajauksien yläpuolella oleva hakupalkki on toiminnallinen. Hakupalkilla voi hakea esimerkiksi tiettyä kuntaa, paikkakuntaa tai palveluntuottajaa. Teknologialtaan hakurajaus toimii samalla tavalla kuin itse palvelusetelin numerolla haku, eli Regexilla ja String metodilla.

Hakutuloksiin on tehty esimerkin vuoksi kuhunkin palveluntuottajan korttiin asiakasarvio, joka muodostuu palveluntuottajan asiakkailtaan saamista arvioista. Arviossa on tähtiä 1-5, jotka ovat maalattuina keltaisiksi arvion mukaan. Alapuolella näkyy arvioiden määrä, jolloin käyttäjät näkisivät tilastollisesti, kuinka luotettavia arviot ovat. Tässä esimerkissä palveluntuottaja on saanut arvioksi neljä tähteä viidestä. Tähtimäärä perustuu niihin 50 arvioon, joita palveluntuottaja on saanut. Arviot olisi hyvä myös saada avattua, ja nähty mitä palautetta palveluntuottaja on saanut, jolloin käyttäjä voisi valita juuri sellaista palvelua kuin hän tarvitsee. Korteissa on myös painikkeet, jotka ohjaavat PSOPissa olevaan palveluntuottajan omaan lisätietosivuun ja yhteydenottopyyntölomakkeeseen.

Palveluntuottajan korttiin toivottiin myös Zeckit-widget (Kuva 14), joka kertoo valittuja asioita palveluita etsiville ihmisille. Zeckit-widgetistä voi myös avata selvityksen, jossa näkyy lisätietoja yrityksestä. Zeckit-widget on saatavilla ilmaiseksi Zeckitin verkkosivuilta, josta voi kopioida widgetin koodin ja liittää sen oman verkkosivustonsa ohjelmistokoodiin. Näyttääkseen halutun yrityksen tiedot, pitää widgetin koodista muuttaa vihreällä näkyvä y-tunnus oman tai muun halutun yrityksen y-tunnukseksi.

Kuva 14 Zeckit widget toivottiin esittelymalliin antamaan lisätietoja yrityksestä [37]

Jotta vertailun yksi tärkeä ominaisuus, hintavertailu, ei menisi hukkaan, on tämän mallin palveluntuottajakorttien oikeisiin yläkulmiin lisätty näkyviin ensisijainen hinta palvelusta, jota haetaan. Lisähintatietoja saa palveluntuottajan Lisätietoja -sivulta, tai heidän omilta kotisivuiltaan. Koska palveluiden hintatiedot ja palvelut eri aikoina voivat olla niin eroavia toisistaan, päätettiin ohjelmoidun esittelymallin hinnastosta tehdä erilainen verrattuna aikaisempaan graafiseen malliin ja nykyiseen verkkopalveluun.

4.6.2 Palveluntuottajan lisätietosivu

Kun haku sekä vertailu oli rakennettu ja tietokantayhteys saatu varmistettua, seuraavaksi alettiin suunnittelemaan ja rakentamaan palveluntuottajien lisätieto-osiota, joka tulisi olemaan merkittävän erilainen kuin nykyisessä PSOP-järjestelmässä on. Nykyiset PSOP-järjestelmän palveluntuottajien lisätiedot tulevat esiin painamalla palveluntuottajan palkkia hakutuloksissa. Kun palkkia on painettu, siirtyy ikkuna sivun alalaitaan aivan palveluntuottajien listauksen alapuolelle, jonne lisätiedot avautuvat. Lisätietoja on vaivalloista tarkkailla useilta eri tuottajilta, koska käyttäjä joutuu rullaamaan ikkunansa itse takaisin ylös siihen mihin hän jäi. Palveluntuottajien lisätietojen näyttämiseen toivottiin parannuksia, jotka parantaisivat lisätietojen tarkkailun käytettävyyttä ja saavutettavuutta. PSOP-järjestelmän saavutettavuus ja käytettävyys gradututkielman kirjoittajan, mukaan näytönlukulaitteet eivät pysty lukemaan nykyisen järjestelmän lisätietoja riittävän hyvin [3]. Toimeksiantajan ja gradututkielman kirjoittajan kanssa keskusteltiin siitä, olisiko graafiseen esittelymalliin suunniteltu painikkeesta laajeneva lisätieto-osio toimiva. Keskustelussa todettiin, että paras vaihtoehto olisi tehdä erillinen lisätietosivu, joka aukeaa uuteen välilehteen. Myös se, että laajeneva lisätieto-osio olisi hyvin pieni ja siihen saisi vain rajallisesti sisältöä, vaikutti päätökseen.

Esittelymallin palveluntuottajakortin lisätietoja -painikkeesta pääsee kyseiselle palveluntuottajalle luodulle sivulle, jossa on pidempi esittely, hinnasto ja muita lisätietoja. Sivu on suunniteltu yksinkertaiseksi, ja siellä on tarpeellinen tieto helposti löydettävissä. Ohjelma ohjautuu palveluntuottajakohtaiseen infosivulle Mongoosen findById metodilla, joka löytää kyseisen palveluntuottajan tiedot sen _id kentän perusteella. Tällä tavoin tälle sivulle saadaan tulostettua vain kyseisen palveluntuottajan tietoja. Graafisessa esittelymallissa on heti hakutuloksien listauksessa laajentuva palveluntuottajan kortti. Painiketta painattaessa kortti laajenee. Laajentuvan lisätieto-osion tila on rajallista, jolloin kaikki toivottavat tiedot eivät välttämättä mahdu kortille. Lisätietoja -sivun otsikko-osio on Bootstrapin

Jumbotron elementti, joka korostaa otsikon ja alaotsikon suuremmalla fontilla, sekä taustalla. Jumbotronin sisällä on Palveluntuottajan nimi, toimipaikan tiedot ja pidempi esittely. Alareunassa on myös kolme painiketta: ”yrityksen sivut”, ”jätä yhteydenottopyyntö” ja ”ajanvaraus”. Otsikkoalueen vasemmalle yläpuolelle on laitettu ”Takaisin” -painike, jotta sivuston käyttäminen olisi käyttäjälle helpompaa.

Tässä versiossa hinnastot ovat lisätietosivulla taulukkomuotoisena (Kuva 15). Taulukkoon tulee näkyviin palveluntuottajan ne hintatiedot, joita on asiakkaan setelissä. Eli esimerkiksi tässä tapauksessa palveluseteli, jolla on tarkoitus hakea Turusta siivouspalveluita näyttää palveluntuottajan sivulle taulukon, jossa on määritetty palvelut ja hinnat eri päivinä ja kellonaikoina:

Hinnasto:

Palvelu	Arkisin 7-16	Arkisin 16-22	Arkisin 23-7	Pyhä 7-16	Pyhä 16-22	Pyhä 23-7
Siivous	56 €/tunti	60 €/tunti	65 €/tunti	56 €/tunti	65 €/tunti	65 €/tunti
Kodinhoito	45 €/h	45 €/h	45 €/h	45 €/h	45 €/h	45 €/h

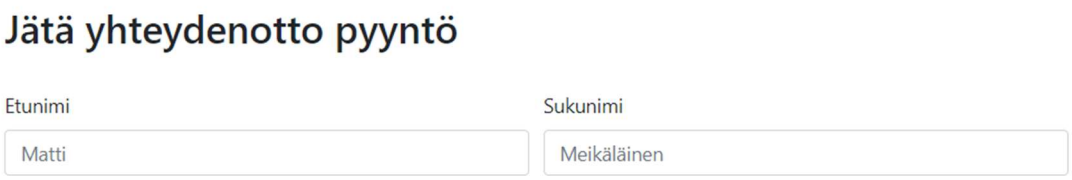
Kuva 15 Palveluntuottajan lisätietosivujen hinnasto taulukko.

Palveluntuottaja voi määrittää oman hinnaston PSOPin kautta, jossa hän voi määrittää eri hinnat eri ajoille. Tässä esimerkissä (Kuva 11) palveluntuottajalle on asetettu ilta- ja yövuoro hinnat jonkin verran suuremmiksi kuin päivisin. Hintatiedot tulevat tietokannasta, josta ne saadaan noudettua samalla tavalla, kuin aiemmin muutkin tiedot on noudettu.

Lisätietojen järjestys on yritetty sijoittaa loogisessa järjestyksessä. Ensin on palveluntuottajan esittely, jonka jälkeen hinnasto. Jos käyttäjä haluaa vielä lisätietoja, on hinnaston alapuolella korteissa vielä muut lisätiedot, kuten palveluntuottajan esteettömyystiedot, viittomakieli, esteetön piha tai muita vastaavia. Sivulla kerrotaan myös saatavat palvelukielet ja muut palvelut, joita palveluntuottaja tarjoaa. Näiden tietojen lisäksi on vielä yhteyshenkilön tiedot ja kartta. Nämä lisätiedot ovat sivustolla Bootstrapin Kortti-elementeissä. Karttaa painamalla aukeaa uuteen välilehteen Google Maps, jossa on kyseisen palveluntuottajan sijainti.

4.6.3 Yhteydenottopyyntö

Keväällä 2019 pidetyn työpajan sekä asiakaspäivien yhtenä tärkeänä kehitystoiveena on mm. mahdollisuus jättää yhteydenottopyyntö. Tämä lisäisi palveluntuottajien lähestyttävyyttä ja laskisi ihmisten kynnystä ottaa yhteyttä. Tällöin myös palveluntuottajat voisivat mahdollisesti rauhassa katsastaa pyynnön lähettäjän tarpeen ja viestin mukaan, ja vastata hänelle mahdollisella palvelun antamisen ajankohdalla. Koska ”Jätä yhteydenottopyyntö” -toiminnon tarve kuului tärkeimpien joukkoon, on hakutuloksien palveluntuottajan korttiin ja lisätietosivulle lisätty painikkeet yhteydenottopyynnön jättämiseksi. Painikkeesta aukeaa uusi välilehti, jossa on isohkot kentät ja joihin kirjoitetaan pyydetyt tiedot. Ideana olisi, että lomakkeita olisi erilaisia, eri palveluille riippuen palveluntuottajasta tai vaihtoehtoisesti olisi yksi yleispätevä vakio lomakepohja. Kentät on tehty <input> elementillä, joissa on paikkamerkki teksti (engl. placeholder) vaalean harmaalla tekstillä esimerkkinä mitä kyseiseen kenttään tulee kirjoittaa. Kenttien yläpuolella on myös <label> eli nimike (Kuva 16).



Kuva 16 Yhteydenottopyyntö lomakkeen nimikentät nimikkeillä ja paikkamerkeillä.

Lomakkeessa on myös viestikenttä ja valintanapit yhteydenottotavan valintaan, jossa kerrotaan vastaanottajalle, miten ottaa yhteyttä pyynnön lähettäjään. Lomakkeen pystyy lähettämään palveluntuottajalle tai peruuttamaan lomakkeen täytön ja palaamaan takaisin hakutuloksiin.

4.7 Ohjeet-sivu

Lopuksi on toteutettu sovelluksen Ohjeet-sivu, jotta esittelymallin käyttö onnistuisi sujuvasti ja käyttäjä voisi testata esittelymallia itsenäisestikin. Ohjeet luotiin erityisesti tulevia asiakaspäiviä varten, jossa osallistujat ja asiakkaat pääsisivät testaamaan sovellusta. Ohjeet-sivulla on kerrottu ohjeita ja taustaa ohjelman käyttöön. Itse käyttöohjeessa on

neljä vaihetta, joita käyttämällä esittelymallin testaus onnistuu. Ohjeeseen on listattu kolme palveluseteliä kolmelle eri palvelunjärjestäjälle: Turku, Helsinki ja PHHYKY. Jotta esittelymallin toiminnallisuuksien ymmärtäminen olisi helpompaa, on näiden järjestäjien setelinnumeroiden perään kirjoitettu myös se, mitä palveluita seteliin kuuluu. Ohjeet-sivun otsikko on myös Bootstrapin Jumbotronin päällä, jolloin se on yhteneväinen muun sivuston asetteluiden kanssa. Jumbotronin sisään on laitettu painike, jota painamalla aukeaa uusi välilehti YouTuben Parasta Palvelua -kanavalle, jossa on ohjevideoita PSOP-järjestelmän käyttöön. Ohjeet itsessään on Jumbotronin alla omassa kontissaan (engl. container). Tällä ohjeet-sivulla voisi tulevaisuudessa olla ohjeita PSOP-järjestelmän käyttöön ja järjestelmän taustaan liittyvää tietoa.

4.8 Esittelymallin jako

Valmis esittelymalli tallennetaan esimerkiksi GitHubiin, josta ohjelmakoodi linkitetään Herokuun. Heroku on konttipohjainen pilvialustan palvelu (PaaS). Kehittäjät käyttävät Herokua esimerkiksi sovellusten koostamiseen (engl. build), suorittamiseen (engl. run) ja hallintaan. Sovelluksesta luodaan oma linkki, josta käyttäjät pääsevät käyttämään sovellusta. [38.]

5 KÄYTTÄJÄPALAUTE

Parasta Palvelua järjestelmän Palveluseteli- ja ostopalvelujärjestelmästä tehty esittelymalli on tarkoitettu testaamiseen, markkinointiin ja kehitykseen. Kohderyhminä on toimeksiantaja, tuottaja ja asiakkaat. Toimeksiantaja voi mahdollisesti testata esittelymallin toimivuutta näkövammaisten liiton kanssa, jolloin esittelymalli pääsisi lukulaitteilla ja pistekirjoituslaitteilla testaukseen. Tällöin voitaisiin ottaa saavutettavuudesta palautetta. Toimeksiantaja tulee hyödyntämään esittelymallia PSOP-esittelyissä, kehityksessä, tai järjestelmän kehitysideoiden kartoittamisessa. Toimeksiantaja voisi mahdollisesti myös käyttää ohjelmistokoodia apunaan. Asiakkaat voisivat testata esittelymallia ja antaa palautetta, siitä mitä muuta PSOP-järjestelmässä voisi kehittä ja mitkä asiat ovat tärkeimpiä keitys kohteita. He osaisivat myös kertoa mitä ominaisuuksia ja toimintoja palveluun tarvittaisiin, jotta se vastaisi loppukäyttäjien tarpeita. Helpottaakseen testausta on linkkeihin ja tärkeisiin huomioitaviin kohtiin laitettu työkaluvihjeitä (engl. tooltip). Niiden tarkoitus on selostaa ja opastaa testaajille mitä tapahtuu mistäkin silloinkin, kun testaaja ei ole saanut selostusta tai suurempaa esittelyä esittelijältä esittelymallista.

Prototyyppiä testattiin asiakaspäivillä 10.10.2019, jolloin asiakkaat ja läsnäolijat pääsivät testaamaan esittelymallia. Testauksen jälkeen vierailijoita pyydettiin vastaamaan muutamaan Mentimeterillä luotuun kyselyyn (Kuva 17), jossa käsiteltiin ajankohtaisia asioita, joista yksi kysely oli esittelymalliin liittyvä. Kyselyyn vastasi 19 henkilöä. Kyselyssä pyydettiin vastaajia laittamaan seitsemän valmiiksi valittua asiaa tärkeysjärjestykseen. Näiden vastauksien perusteella, on mahdollisesti helpompi lähteä kehittämään oikeita asioita. Kolme eninten ääntä saanutta kiteyttää järjestelmän tärkeimmät asiat, joita halutaankin kehittää eteenpäin.

Kuva 17 Asiakaspäivien Menti -kyselyn vastaukset

Asiakaspäivien toiminnallisuuksien tärkeysjärjestys -kyselyssä haluttiin kartoittaa, mitkä ovat asiakkaiden mielestä tärkeimpiä kehitettäviä palveluita tai toiminnallisuuksia. Palveluntuottajan esittelysivua pidettiin hyvänä ja tärkeimpänä. Siinä palveluntuottajille luotaisiin oma lisätietosivu, jota tässä esittelymallissakin esiteltiin kehitys ehdotuksena. Toisella sijalla oleva ”Palveluntuottajan hintatiedot” tarkoittaa sitä, miten hinnastot näytetään ja miten niitä olisi helppo ja vaivaton tulkita. Hinnastot ovat muutenkin järjestelmässä tärkeässä roolissa, koska käyttäjälle voi olla nimenomaan tärkeää se, minkä hintaista palvelua hän saa. Kolmannen sijan ”palvelusetelin numerolla vertailu” tarkoittaa sitä, että järjestelmästä pystyisi hakemaan palveluita syöttämällä palvelusetelin numero järjestelmässä olevaan hakukenttään. Tällöin järjestelmä osaisi hakea seteliin merkityt palvelut ja tiedot, ja antaa sitä vastaavat tulokset käyttäjälle nähtäväksi. Tällöin käyttäjän ei tarvitsisi erikseen hakea palveluita niin kuin nykyisessä PSOP-järjestelmässä. Yhteydenottopyynnön jättämistoiminto tarkoittaisi sitä, että asiakas pystyisi lähettämään palveluntuottajalle pyynnön, ottaa yhteyttä asiakkaaseen. Palveluntuottajan muut palvelut tarkoittavat sitä, että palveluntuottajan tiedoissa kerrottaisiin, mitä kaikkia muita palveluita kyseinen palveluntuottaja tarjoaa. ”Kartta ja paikkatiedot” ja ”Tilaajavastuu, Zeckit-tiedot” toisivat palveluntuottajan korttiin tai lisätietoihin kartan, joka mahdollisesti näyttäisi tuottajan toimialueen ja Zeckit pienoisohjelman tai merkin sitä kuuluuko tuottaja Tilaajavastuuseen.

Toimeksiantajan, kollegoiden ja asiakaspalautteiden toiveena ja kehitysehdotuksena on myös saada palveluntuottajan lisätietoihin statistiikkaa, esimerkiksi hoitokodeista ja muista palveluista, joissa on tärkeää nähdä mitkä ovat asukkaiden keskimääräiset aktiviteetit. Näitä voivat olla mm. ulkoilu, ruokailu, vapaa-aika, aktiiviaika tai muut, jotka kuuluvat asiakkaan hyvinvointiin. Tällöin omainen voisi katso statistiikan perusteella, mikä sopisi heille parhaiten niin palveluiltaan kuin luotettavuudeltaankin.

6 YHTEENVETO

Työn tavoitteena oli luoda toimeksiantajalle Parasta Palvelua Palveluseteli- ja ostopalvelujärjestelmästä ohjelmoitu esittelymalli, jota voitaisiin käyttää saavutettavuuden ja käyttäjätiedon parantamiseen. Tarkoituksena oli tehdä graafisen esittelymallin pohjalta uusi, ohjelmoitu versio, jossa on toiminnallisuuksia, kuten hakutoiminnot kunnalla ja palvelusetelin numerolla, sekä hakutulokset haettuna tietokannasta. Tavoitteena oli myös, että esittelymallia testattaisiin näkövammaisten liitolla näytönlukulaitteilla ja pistekirjoituslaitteella. Tätä testausta ei kuitenkaan päästy toteuttamaan ajanpuutteen vuoksi.

Opinnäytetyön ajallisena tavoitteena oli luoda esittelymalli kesän-heinäkuun aikana. Ajallinen tavoite onnistui kohtalaisen hyvin. Työn etenemisessä oli joitain haasteita, joiden takia työn etenemisessä piti mennä joitain askeleita taaksepäin, jotta saataisiin Node.js -ohjelman luotua ja tietokannan yhdistettyä. Tämän takaiskun olisi voinut välttää kunnollisella ja huolellisella perehtymisellä sekä suunnittelulla. Takaisku myös aiheutti haasteita ajan kanssa, koska uudet aiheet olivat haastavia, minkä takia niihin piti perehtyä kattavasti. Tästä syystä työn eteneminen oli hidasta ja työlästä. Toimeksiantajan kanssa käytiin keskustelua työn vaatimuksista, jonka seurauksena päätettiin rajata toiminnallisuudet palvelusetelin numerolla hakuun, datan hakuun tietokannasta ja ulkoasuun. Myöskään joitain ominaisuuksia ei tehty toiminnallisiksi haasteellisuuden ja ajanpuutteen vuoksi. Tässä vaiheessa jätettiin myös tutkimus osion tekemisen pois aihepiiristä.

Haasteista huolimatta toteutuksen valmistuminen venyi vain joillain viikoilla eteenpäin. Heinä-elokuun aikana varattiin aikaa pelkästään opinnäytetyölle. Tämä oli kannattavaa, koska sinä aikana sain mietittyä, suunniteltua ja perehdyttyä haasteellisiin aiheisiin. Esittelymalli onnistui ulkoasullisesti ja toiminnallisesti paremmin kuin odotettiin. Joitain toiminnallisuuksia jouduttiin jättämään pois, kuten kunnalla haku ja hakurajaukset. Tyytyväisyys esittelymalliin on toimeksiantajan ja asiakkaidenkin kannalta hyvä. Harmillista on, että joitain haluttuja toimintoja, kuten toiminnallisuuksia ja kattavammat tietokannat ei pystytty toteuttamaan.

Työtä tehdessä on opittu todella paljon uusia asioita niin, että ne ovat jääneet mieleen. Vielä olisi ollut paljon opittavaa ja työtä olisi voinut jatkaa loputtomiin niin työmäärän kuin perfektionisminkin takia. Tämän takia pitikin vain päättää, missä kohtaa työ on valmis. Koska esittelymalli on vaillinainen versio käytettävästä järjestelmästä, oli työn rajaami-

nen helpompaa, kun sen ei tarvitse eikä kuulukaan tavoitella täydellisyyttä. Kun kyseessä on esittelymalli sitä ei kuulu myöskään ottaa suoraan käyttöön järjestelmäksi. Esittelymalli on sovellus, joka on tarkoitettu esittelyyn, kehitykseen ja testaukseen. Työ oli kiinnostava ja siihen sai paljon aikaa uppoamaan. Erityisesti on opittu back-end puolen ohjelmoinnista. Yksi tärkeimmistä asioista, joka tehtäisiin toisin, olisi heti alkuun kunnollisempi perehtymien aihealueeseen ja kunnollinen suunnittelu. Näiden avulla olisi voitu välttää suurimmat takaiskut ja saanut enemmän aikaa kehitykseen. Myös kattavan vaatimusmäärittelyn tekeminen olisi selventänyt mitkä ovat työn vaatimukset ja tavoitteet.

Esittelymalliin voitaisiin jatkokehityksessä lisätä molempien hakujen sekä hakurajauksien toiminnallisuudet. Tällöin esittelymalli olisi kattavampi ja kunnalla hakutoimintoakin voitaisiin paremmin luonnostella ja testata ennen toteutusta. Myös saavutettavuusominaisuuksia pitäisi parantaa, jolloin sitä olisi helppo käyttää myös näytönlukulaitteella ja pistekirjoituslaitteella. Palveluntuottajan lisätietosivun taulukko voitaisiin rakentaa paremmaksi, ja se voisi mahdollisesti vastata realistisempaa taulukkoa, jota tultaisiin käyttämään. Myös tulosten listauksessa näkyvien korttien hintatietoja voisi miettiä tarkemmin ja tarpeen tullen kehittää. Jotta ohjelma pysyisi miellyttävänä ja saavutettavana käyttää, ei suositella pitkiä ohjetekstejä hankalilla sanoilla. Myös yhteydenottopyyntölomakkeen kenttiä voisi miettiä tarkemmin ja kattavammin. Esimerkiksi voisi miettiä mitä kaikkia kenttiä tarvitaan ja halutaanko kaikille palveluille soveltuva peruslomake, vai palvelukohtaiset lomakkeet.

Saavutettavuuteen on jatkossa panostettava entistä enemmän erityisesti EU-saavutettavuusdirektiivi huomioiden. Saavutettavalla, helppokäyttöisellä ja ulkonäöllisesti hyvin suunnitellulla järjestelmällä yhä useampi kuntalainen pystyisi käyttämään järjestelmää itsenäisesti. Itse PSOP-järjestelmän osalta olisi jatkossa syytä huomioida joitain merkittäviä bugikorjauksia ja käytettävyyteen liittyviä korjauksia käyttöliittymässä. Käyttäjäkokemusta voitaisiin parantaa huomattavasti jo pienilläkin korjauksilla tai muutoksilla.

LÄHTEET

[1] Tiera.fi. Yritys. [Viitattu: 8.10.2019] [www-sivu] Saatavilla: https://www.tiera.fi/yritys/

[2] Parasta Palvelua. Etusivu. 2018. [Viitattu: 3.8.2019] [www-sivu] Saatavilla: http://www.parastapalvelua.fi/etusivu/

[3] Oscar Ranta. Analyzing the accessibility and usability of the PSOP system through user testing with visually impaired users. Pro gradu -tutkielma. Åbo Akademi, Turku. 2019. s. 117. [www-sivu] Saatavilla: https://www.doria.fi/bitstream/handle/10024/169391/ranta_oscar.pdf?sequence=2

[4] Markku Pekurinen. Valinnanvapauden nykytilanne Suomessa - Sote-uudistus - THL. [Viitattu: 8.10.2019] [www-sivu] Saatavilla: http://thl.fi/fi/web/sote-uudistus/valinnanvapaus/valinnanvapauden-nykytilanne-suomessa

[5] Saavutettavuusdirektiivi.fi. Saavutettavuus on arvovalinta. [Viitattu: 1.12.2019] [www-sivu] Saatavilla: https://saavutettavuusdirektiivi.fi/

[6] Ylänne Kirsi. Mikä ihmeen saavutettavuusdirektiivi? 2017. [Viitattu: 1.12.2019] [www-sivu] Saatavilla: https://www.celia.fi/Blogi/mika-ihmeen-saavutettavuusdirektiivi/

[7] Laura Luomalehto. Mobiililaitteiden käytettävyys ja käytettävyystestaus. Insinöörityö. Metropolia Ammattikorkeakoulu. 2018. s. 31. [Viitattu: 20.10.2019] [www-sivu] Saatavilla: http://urn.fi/URN:NBN:fi:amk-2018052910893

[8] ISO.org. ISO 9241-11:2018(en), Ergonomics of human-system interaction — Part 11: Usability: Definitions and concepts. [Viitattu: 5.11.2019] [www-sivu] Saatavilla: https://www.iso.org/obp/ui/fr/#iso:std:iso:9241:-11:ed-2:v1:en

[9] Suomen virallinen tilasto (SVT): Väestön tieto- ja viestintätekniikan käyttö [verkkojulkaisu]. ISSN=2341-8699. 2017, 2. Internetin käyttö mobiililaitteilla. Helsinki: Tilastokeskus [Viitattu: 20.10.2019] [www-sivu] Saatavilla: https://www.stat.fi/til/sutivi/2017/13/sutivi_2017_13_2017-11-22_kat_002_fi.html

[10] Cisco.com. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2017–2022 White Paper. [Viitattu: 20.10.2019] [www-sivu] Saatavilla:

https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-738429.html

[11] Minna Karukka ja Tommi Inkilä. Responsiivinen verkkosivujen suunnittelu mukauttaa sisällön eri päätelaitteille. 2013. [Viitattu: 10.10.2019] [www-sivu] Saatavilla: http://www.oamk.fi/epooki/2013/responsiivinen-verkkosivujen-suunnittelu/

[12] WEBYDO. Why Responsive Design Support is the Most Important Feature You Can Add To Your Website. 15.7. [Viitattu: 8.11.2019] [www-sivu] Saatavilla: https://www.awwwards.com/why-responsive-design-support-is-the-most-important-feature-you-can-add-to-your-website.html

[13] Matthew Carver. The Responsive Web2014[Viitattu: 24.10.2019 2019][www-sivu] Saatavilla: https://www.manning.com/books/the-responsive-web#toc

[14] Saavutettavasti.fi. Tietoa saavutettavuudesta. [Viitattu: 20.10.2019] [www-sivu] Saatavilla: https://www.saavutettavasti.fi/tietoa-saavutettavuudesta/

[15] Saavutettavuusvaatimukset.fi. Saavutettavuus. [Viitattu: 28.8.2019] [www-sivu] Saatavilla: https://www.saavutettavuusvaatimukset.fi/saavutettavuus/

[16] Laak Timo. Saavutettavaa typografiaa – Osa 1. 2006. [Viitattu: 5.11.2019] [www-sivu] Saatavilla: https://saavutettava.fi/2006/03/24/saavutettavaa-typografiaa-osa-1/

[17] Saavutettavasti.fi. Saavutettavat asiakirjat. [Viitattu: 20.10.2019] [www-sivu] Saatavilla: https://www.saavutettavasti.fi/saavutettavat-tiedostot/word/

[18] Celia. Värit ja kontrastit. [Viitattu: 30.10.2019] [www-sivu] Saatavilla: https://www.celia.fi/saavutettavuus/verkkopalvelujen-saavutettavuus/sivuston-paatoimittajan-ohjeet/varit-ja-kontrastit/

[19] W3C, MIT, ERCIM, Keio ja Beihang. Use of Color. 2016. [Viitattu: 30.11.2019] [www-sivu] Saatavilla: https://www.w3.org/TR/UNDERSTANDING-WCAG20/visual-audio-contrast-without-color.html

[20] W3C. The web standards model - HTML CSS and JavaScript. 2014. [Viitattu: 1.9.2019] [www-sivu] Saatavilla: https://www.w3.org/wiki/The_web_standards_model_-_HTML_CSS_and_JavaScript#CSS_.E2.80.94_let.E2.80.99s_add_some_style

[21] MDN. CSS. [Viitattu: 30.10.2019] [www-sivu] Saatavilla: https://developer.mozilla.org/en-US/docs/Glossary/CSS

[22] Mark Otto, Jacob Thornton ja Bootstrap. Bootstrap. [Viitattu: 26.8.2019] [www-sivu] Saatavilla: https://getbootstrap.com/

[23] Mikael Myhrberg. Web-sivujen kehitys ja mobiilioptimointi käytettäessä Bootstrap-alustaa. Opinnäytetyö. Hämeen ammattikorkeakoulu, Riihimäki. 2015. s. 41. [Viitattu: 1.12.2019] [www-sivu] Saatavilla: http://urn.fi/URN:NBN:fi:amk-2015090214269

[24] Metropolia. Responsiiviset sivustot, Bootsrap. [Viitattu: 25.10.2019] [www-sivu] Saatavilla: https://users.metropolia.fi/~kuivi/bfish/bootstrap.php

[25] Mark Otto, Jacob Thornton ja Bootstrap. About. [Viitattu: 27.8.2019] [www-sivu] Saatavilla: https://getbootstrap.com/docs/4.3/about/overview/

[26] BitBucket. Versionhallintaohjelmisto: yleiskatsaus. [Viitattu: 21.10.2019] [www-sivu] Saatavilla: https://bitbucket.org/product/fi/version-control-software

[27] SourceTree. Sourcetree | Free Git GUI for Mac and Windows. [Viitattu: 21.10.2019] [www-sivu] Saatavilla: https://www.sourcetreeapp.com

[28] W3schools. Node.js Introduction. [Viitattu: 1.9.2019] [www-sivu] Saatavilla: https://www.w3schools.com/nodejs/nodejs_intro.asp

[29] TutorialsPoint. Node.js - Introduction. [Viitattu: 15.10.2019] [www-sivu] Saatavilla: https://www.tutorialspoint.com/nodejs/nodejs_introduction.htm

[30] Nodejs.org. About. [Viitattu: 1.9.2019] [www-sivu] Saatavilla: https://nodejs.org/en/about/

[31] Tutorialspoint. MongoDB - Overview. 2019. [Viitattu: 15.10.2019] [www-sivu] Saatavilla: https://www.tutorialspoint.com/mongodb/mongodb_overview.htm

[32] Karnik Nick. Introduction to Mongoose for MongoDB. 2018. [Viitattu: 21.10.2019] [www-sivu] Saatavilla: https://www.freecodecamp.org/news/introduction-to-mongoose-for-mongodb-d2a7aa593c57/

[33] Tutorialspoint. MongoDB - ObjectId. 2019. [Viitattu: 15.10.2019] [www-sivu] Saatavilla: https://www.tutorialspoint.com/mongodb/mongodb_objectid.htm

[34] Mongoosejs.com. Mongoose. [Viitattu: 05.11.2019] [www-sivu] Saatavilla: https://mongoosejs.com/

[35] MDN contributors. Regular Expressions. 2019. [Viitattu: 11.10.2019] [www-sivu] Saatavilla: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions

[36] MDN contributors. RegExp. 2019. [Viitattu: 11.10.2019] [www-sivu] Saatavilla: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/RegExp

[37] Zeckit. Näin liität Zeckit-widgetin yrityksesi nettisivuille. [Viitattu: 1.10.2019] [www-sivu] Saatavilla: https://zeckit.com/fi/zeckit-logo-arviot-yrityksen-kotisivuille/

[38] Heroku. About Heroku | Heroku. [Viitattu: 30.11.2019] [www-sivu] Saatavilla: https://www.heroku.com/about