jamk

Perheterapian kyselylomakkeen toteuttaminen verkkoympäristöön

Samu Tynkkynen

Opinnäytetyö, AMK
Toukokuu 2022
Tietojenkäsittely ja tietoliikenne
Insinööri (AMK), tieto- ja viestintätekniikka

jamk | Jyväskylän ammattikorkeakoulu
University of Applied Sciences

jamk

Kuvailulehti

Tynkkynen, Samu

Perheterapian kyselylomakkeen toteuttaminen verkkoympäristöön

Jyväskylä: Jyväskylän ammattikorkeakoulu. Toukokuu 2022, 31 sivua.

Tietojenkäsittely ja tietoliikenne. Tieto- ja viestintätekniikan tutkinto-ohjelma. Opinnäytetyö AMK.

Julkaisun kieli: Suomi

Verkkojulkaisulupa myönnetty: kyllä

Tiivistelmä

Psykoterapia ja konsultaatio Asta Suomen ja Terapiapalvelu Pirjo Lahtisen toiveena oli kehittää verkkopohjainen perheterapian kyselylomake, joka pisteyttäisi ja tuloksiin pohjautuen antaisi käyttäjästä arvion käyttäjälle. Tämän kaltaisille lomakkeille on tarvetta psykoterapian parissa, koska siellä vielä tehdään paljon käsin. Esimerkiksi pisteet lasketaan usein käsin olemassa olevien taulukoiden mukaan.

Kyselylomaketta alettiin kehittämään konstruktiivisella tutkimusmenetelmällä. Konstruktiivisessa tutkimusmenetelmässä yleensä on tarkoitus kehittää konkreettinen asia kuten esimerkiksi tuote. Tutkimusmenetelmän käytännönläheisyyden ja tuotteen tarpeiden vuoksi läheinen yhteistyö oli tärkeää, joten toimeksiantajien kanssa pidettiin palavereita välillä jopa viikoittain.

Kyselylomakkeen toteutusvälineiksi valikoituivat HTML-merkintäkieli kyselylomakkeen runkoa ja ulkoasua varten sekä JavaScript toteuttamaan toiminnallisuudet. Kyselyn kysymysten tallentamista varten valikoitui XML-tietokanta yksinkertaisuutensa vuoksi. Kysymyksistä tallennettiin esimerkiksi kysymykset itse sekä niiden pisteet.

Kyselylomakkeessa käyttäjät vastaavat kyselyn monivalintakysymyksiin, jotka vaihtelevat normaalin oloisista janamaisiin rakenteisiin, kun taas joillakin kysymyksillä voi olla myös jatkokysymyksiäkin. Erilaisilla kysymyksillä on HTML-rakenteet ja niihin liittyvät JavaScript toiminnot. Kysymykset ladataan XML:stä samoin kuin mitä rakennetta käytetään kysymyksen esittämiseksi. Käyttäjän vastattua kaikkiin kysymyksiin saa hän nähdä pdf:llä pisteensä sekä arvion itsestään pisteisiin perustuen.

Tutkimuksen tuloksena kehitettiin kyselylomake, joka toimii verkkoympäristössä ja joka antaa arvion käyttäjästä vastauksiin perustuen. Näiltä osin tutkimuksen tavoitteet tuli täytettyä. Tuote on tarkoitus kaupallistaa, mutta sitä ennen se tarvitsee vielä testausta ja jatkokehitystä. Esimerkiksi maksuominaisuus olisi tarpeen tuotteen kaupallistamisen vuoksi. Kyseinen ominaisuus kuitenkin vaatii tuotteen olevan muilta osin täysin valmis, koska monet maksupalvelut ovat kuukausimaksullisia kuten esimerkiksi Paytrail.

Avainsanat (asiasanat)

HTML, JavaScript, WWW-sivut, CSS, XML, PDF

Muut tiedot (salassa pidettävät liitteet)

Ei ole

Tynkkynen, Samu

Implementation of a family therapy questionnaire in an online environment

Jyväskylä: JAMK University of Applied Sciences, May 2022, 31 pages.

Information and Communications. Degree Programme in Information and Communication Technology. Bachelor's thesis.

Permission for web publication: Yes

Language of publication: Finnish

Abstract

Psykoterapia ja konsultaatio Asta Suomi and Terapiapalvelu Pirjo Lahtinen wish was to develop a web-based family therapy questionnaire, which would score and give an evaluation of the user based on the results. Forms like this are needed in psychotherapy because lot of this is still done by hand there. For example, scores are often calculated manually from existing tables.

The development of the questionnaire was developed by using constructive research method. The constructive research method usually aims to develop a concrete thing such as a product. Due to the practicality of the research method and the needs of the product, close co-operation was required. Therefore, meetings with the clients were sometimes held even on a weekly basis.

The HTML markup language for the body and layout of the questionnaire and JavaScript to implement the functionalities were selected as the implementation tools for the questionnaire. For the sake of simplicity, and XML database was selected to store the questions in the questionnaire. For example, the questions themselves and their scores were stored into the XML database.

In the questionnaire, users answer multiple-choice questions in the questionnaire, which looks can vary while some questions may also have follow-up questions. Various questions have HTML structures and related JavaScript functionality. Questions are loaded from XML database as well as the chosen looks of the question. After the user has answered all the questions, he / she can see his / her score in pdf as well as the evaluation of himself / herself based on score.

As a result of the research, a questionnaire was developed that works in an online environment and provides an assessment of the user based on responses. In these respects, the objectives of the study were met. The product is due to be commercialized, but before that it still needs to be tested and further developed. For example, a payment feature would be necessary because of the commercialization of the product. However, this feature requires the product to be fully finished in other respects, as many payment services have a monthly fee, such as Paytrail.

Keywords/tags (subjects)

WWW-sivut, HTML, JavaScript, CSS, XML, PDF

Miscellaneous (Confidential information)

None

Sisältö

Kuviot

1 Johdanto

Nykyään ihmisillä on paljon kiireitä ja stressiä. Tämän kaiken keskellä on myös mielenterveys, joka voi häiriintyä. THL:n tilastot vuodelta 2020 osoittavat esimerkiksi, että mielenterveyden avohoidon tarve on ollut kasvussa verrattuna vuoteen 2019. Vuonna 2020 yli 200 tuhatta asiakasta tarvitsi psykiatrisen erikoissairaanhoidon palveluita. (Psykiatrinen erikoissairaanhoito 2020. 2021.) Joka vuosi 1,5 prosenttia suomalaisista sairastuu johonkin mielenterveyden häiriöön ja joka viides suomalainen sairastaa jotakin mielenterveyden häiriötä (Huttunen. 2017.).

1.1 Psykoterapia

Psykoterapian on havaittu olevan toimiva hoitomuoto useimmille mielenterveyden häiriöistä kärsiville potilaille. Psykoterapia määritellään osa-alueeksi, jossa hoidetaan psyykkisistä ongelmista ja vaikeuksista kärsiviä potilaita psykologisin menetelmin. Psykoterapiassa hankalia asioita voidaan käsitellä keskustelemalla ja tarvittaessa käyttäytymisen ongelmakohtia muuttaa erilaisten harjoitteiden avulla. Usein psykoterapian alussa asiakkaalle myös tehdään erilaisia testejä ongelmien selvittämiseksi. (Psykoterapia, n.d.)

Mielenterveyshäiriöitä voi olla monenlaisia, mutta lievissä ja keskivaikeissa tapauksissa psykoterapia on keskeinen hoitomuoto. Psykoterapia voi kestää muutamasta kerrasta pariin kymmeneen kertaan tai hoidon tarpeesta riippuen jatkua vuosiakin, jolloin pyritään laajempaan persoonallisuuden ja toimintatapojen muuttamiseen. Psykoterapiaa on useita eri muotoja. Yleisimpiä muotoja ovat kognitiivinen terapia, kognitiivinen käyttäytymisterapia, psykodynaaminen terapia, sekä ryhmä- ja perheterapiat. (Psykoterapia, n.d.)

1.2 Perheterapia

Yksi psykoterapian muoto on perheterapia, jossa otetaan huomioon koko perhe. Perheterapiassa käsitellään monenlaisia ongelmia. Yleisimpiä ongelmia ovat esimerkiksi lasten tai vanhempien väliset riidat, kasvatusongelmat sekä kommunikaatio-ongelmat. Toisinaan vanhemmat huomaavat, että vaikeista asioista puhuminen on mahdotonta ja se johtaa puhumattomuuteen tai vihanpur-

kauksiin. Myös psyykkiset sairaudet tai vakavat fyysiset sairaudet tai vammat voivat luoda ongelmia. Uusperheillä voi olla myös joskus tarvetta perheterapialle. Joskus myös vakava kriisi, kuten perheen jäsenen sairastuminen tai kuolema voi luoda tarpeen terapialle. (Perheterapia n.d.)

Perheterapian tavoitteena on muuttaa vuorovaikutusmalleja siten, että huonot vuorovaikutusmallit muuttuisivat toimiviksi malleiksi ja kommunikaatio paranisi. Perheterapiassa pyritään myös löytämään perheen hyviä puolia ja vapauttamaan voimavaroja perheelle tällä tavalla. (Perheterapia n.d.) Perheterapiassa terapeutti tai useiden terapeuttien ryhmä tapaa istuntojen aikana useampia perheen jäseniä yhtä aikaa ja joskus myös laajempaa tukiverkostoa. Tällöin voidaan puhua perheen yhteisterapiasta, jossa käsitellään koko perheen asioita tai vain pariterapiasta, jossa käsitellään puolisoiden ongelmia. Eri perheterapiamuotojen väliset rajat voivat olla liukuvia käsitteitä. (Aaltonen. 2006.)

1.3 Pariterapia

Parisuhde voi olla yksi elämän tärkeimmistä ihmissuhteista. Hyvä parisuhde edistää parin ja lasten hyvinvointia sekä terveyttä. Joskus parisuhteet kuitenkin kuluttavat pariskunnan voivavaroja sillä myös parisuhteissa kuten muissakin ihmissuhteissa, esiintyy erilaisia vaikeuksia ja kriisejä. Riidat ovat yksi yleisin parisuhteen ongelma. Arkiset asiat kuten raha, kotityöt ja joskus jopa toisen muut ihmissuhteet voivat luoda yllättävänkin vaikeita riitoja. Erilaiset odotukset tai kommunikaation puute luovat monesti ongelmia esim. lapsiperhearjen ongelmat. (Parisuhteen vaikeudet 2021.) Parisuhteen kriiseihin voi hakea apua ja monesti ulkopuolinen apu voi auttaa. Ulkopuolista ammatillista apua voi hakea kuka tahansa. Moni pari saa apua parin ongelmiin pariterapiasta. (Kriisi parisuhteessa n.d.)

1.4 Toimeksiantajat ja tehtävä

Opinnäytetyön toimeksiantajina toimivat Terapiapalvelu Pirjo Lahtinen ja Psykoterapia ja konsultaatio Asta Suomi. Tässä opinnäytetyössä keskitytään tekemään testiversio kiintymyssuhdetyön teemaan osana perheterapiaa. Kiintymysvanhemmuus on John Bowlbyn kiintymyssuhdeteoriaan perustuva, läheisyyttä ja lapsen tarpeiden kunnioittamista korostava vanhemmuuden tyyli erityisesti vauvaikäisten vanhemmille. (Sinkkonen 2004). Kiintymyssuhdeteorian mukaan varhaislapsuu-

dessa kehittynyt vahva tunneside hoivanantajiin muodostaa pohjan aikuisiän läheisille ihmissuhteille, esimerkiksi parisuhteille ja vanhemmuudelle. Tämän testin tarkoituksena on antaa testin tekijälle tietoa omasta kiintymyssuhteen laadusta ja lisävinkkejä, millä omaa kiintymyssuhteen ilmenemistapaa voi kehittää. Testi ei ole standardoitu psykologinen testi vaan siinä on osia persoonallisuutta, kiintymyssuhdetta ja tunnetaitoja koskevista osuuksista. Testin tekijä voi itse testin tulokset saatuaan pohtia, mitä teemaa hän haluaa ja haluaako kehittää. Testi toimii psykoterapeuteille apuna asiakkaiden parisuhteen laatua ja asiakkaiden identiteettiä koskevassa terapiatyössä.

2 Tutkimusasetelma

Tutkimusongelmana oli selvittää, pystytäänkö tekemään sellainen verkkosivukyselylomake, joka voisi hakea kysymykset tietokannasta, näyttää ne ja laskea pisteet vastauksista sekä antaisi tulostettavan pdf tiedoston tuloksista. Alun perin oli tarkoitus lisätä kyselylomakkeelle maksuominaisuus, mutta se jätettiin pois, koska palveluntarjoajien tuotteet maksuominaisuuksille ovat maksullisia ja vain valmiiseen tuotteeseen tällainen ominaisuus olisi kannattanut lisätä.

Tutkimusongelman perusteella seuraavat tutkimuskysymykset valittiin ratkaistavaksi:

1) Kuinka tallennetaan erilaiset kysymykset tietokantaan ja miten ne näytetään satunnaisessa järjestyksessä?

2) Kuinka pisteet lasketaan?

3) Kuinka luodaan PDF-tiedosto, joka näyttää tulokset ja niihin perustuvan arvioinnin.

Tutkimuskysymysten ratkaisemiseksi sopiva tutkimusmenetelmä on konstruktiivinen tutkimus, koska opinnäytetyö on tuote ja tutkimuskysymykset käsittelevät konkreettisia kysymyksiä. Konstruktiivinen tutkimus on osa case-tutkimusmenetelmää, jota käytetään yhteiskuntatieteissä. Case-tutkimustyyppejä on useita ja konstruktiivinen on yksi niistä. Konstruktiivinen tutkimustapa on yleinen, kun haetaan tietotekniikan ratkaisuja. Konstruktiivisessa tutkimuksessa tulokset voivat

olla hyvin monenlaisia. Ne voivat olla esimerkiksi ohjeita tai tuotteita. Ne voivat olla myös matemaattisia ratkaisuja ongelmiin tai suunnitelmia jonkin asian toteuttamiseksi. (Lukka. 2001)

3 Tietoperusta

Nykyaikaisessa kehittyneessä yhteiskunnassa tiedon saaminen on helpompaa kuin koskaan aiemmin. Erilaiset verkkosivut ja sovellukset helpottavat tiedon hankintaa ja ihmisten arkea. Se mitä verkkosivuilla käyttäjät näkevät ovat käyttäjäpuolen (eng. front-end tai client-side) käyttöliittymien kehittäjien tekemiä visualisointeja kaikelle sille tiedolle, jota internet on täynnä. Käyttäjäpään kehittäjiä on kutsuttu monella nimellä kuten html-koodari tai web-kehittäjä. Käyttäjäpään kehitys on paljon muutakin kuin valmiin verkkosivun toimiva ulkoasu. Verkkosivun tekeminen vaatii monenlaisia eri tekniikoita, joiden käyttäminen sujuvasti on toimivan verkkosivun tekemisen perusta. (Codesido. 2009)

3.1 HTML ja CSS

HTML (Hyper Text Markup Language) on monen verkkosivun perusta. Sillä luodaan visuaaliselle ja funktionaaliselle esitykselle tukiranka. HTML muodostuu elementeistä ja tunnisteista (eng. tag). Usein HTML tiedosto alkaa <!DCOTYPE html> rivillä kuten Kuvio 1. esimerkissä. Tämä kertoo, että dokumentti on HTML5 dokumentti. Tämän jälkeen tulee <html> </html> -tunniste, joka on koko HTML sivun juuri. Tämän tunnisteen sisälle tulee <head></head> -tunniste, jonka sisälle tulee yleensä sivun metatiedot sekä sivun nimi. <html> tunnisteen sisälle tulee myös <body></body> -tunniste, jonka sisällä on yleensä kaikki verkkosivun rakenne ja toiminnot. esimerkiksi siellä voisi olla tunniste, joka tarkoittaa paragrafia. Tämän sisällä voisi olla jokin teksti.

```
<!Doctype html>
<html>
    <head>
        <title>Esimerkki</title>
    </head>
    <body>
        <h1>Otsikko</h1>
        <p>tekstiä</p>
    </body>
</html>
```

Kuvio 1. HTML esimerkki

CSS (Cascading Style Sheets) on pieni mutta tärkeä osa HTML verkkosivua. yleensä sitä varten luodaan oma tiedosto esimerkiksi style.css. Kyseinen tiedosto usein linkitetään html tiedostoon <head> -tunnisteen sisällä. Se määrittää kuinka sivu näkyy selaimella. Sillä säädellään verkkosivun tyyliä esimerkiksi fonttia, väriä ja sijoittumista verkkosivulla. (Thorndyke. 2021)

3.2 XML

XML (extensible Markup Language) on kielenä samankaltainen kuin HTML. Se sopii esimerkiksi tietojen tallentamiseen eräänlaisena tietokantana tai dokumenttien tallentamiseen tiedostona. Toisin kuin HTML:ssä, XML:ssä ei ole nimettyjä elementtejä tai tunnisteita, vaan käyttäjä itse nimeää ne haluamallaan nimellä ja muodostaa niistä haluamiaan rakenteita. Kuvio 2:ssa on esimerkki, kuinka kirjakokoelma voisi rakentua XML:ssä. Siinä <kirjasto> -tunnisteen sisälle on luotu <kirja> -tunnisteita, jotka sisältävät tunnisteet nimestä ja kirjoittajasta. <nimi> ja <kirjoittaja> tunnisteiden välissä on itse tiedot kirjan nimestä ja kirjoittajasta.

```
<kirjasto>
    <kirja>
        <nimi>kirja 1</nimi>
        <kirjoittaja>kirjoittaja 1</kirjoittaja>
    </kirja>
    <kirja>
        <nimi>kirja 2</nimi>
        <kirjoittaja>kirjoittaja 2</kirjoittaja>
    </kirja>
</kirjasto>
```

Kuvio 2. XML esimerkki

3.3 JavaScript

JavaScript on maailman käytetyin ohjelmointikieli. Se on helppo oppia ja sillä voi luoda monen laisia ohjelmia ja toimintoja vaikkapa verkkosivuihin. JavaScriptiä ei tule sekoittaa kuitenkaan Java-ohjelmointikieleen. Ne ovat nimeltään samankaltaisia mutta niiden syntaksi ja semantiikka ovat erilaisia. JavaScriptiä voi käyttää verkkosivujen yhteydessä kahdella tapaa. Se voidaan liittää suoraan HTML:ään <script></script> -tunnisteen väliin. Useimmiten se on sijoitettu body-elementin sisälle. Toinen tapa on luoda JavaScript-tiedosto ja liittää se body-elementtiin <script

type="text/javascript" src="script.js"></script> sisällytettynä <script> -tunnisteeseen. Javascriptissä on muuttujia kuten var, let ja const. Muuttujiin voi tallentaa tietoja kuten esimerkiksi numeroita tai tekstijonoja.

Usein JavaScriptiä käytetään funktioina, joita kutsutaan tarvittaessa kuten esimerkiksi nappia painettaessa. Tällöin funktio palauttaa yleensä jonkin halutun arvon. Esimerkiksi se voisi palauttaa (return) kellon ajan tai jonkin laskun lopputuloksen. Funktiot alkavat avainsanalla function ja tämän jälkeen tulee sen nimi ja lopuksi sulkeet (). Esimerkiksi function laskeLasku() (Kuvio 3). Sulkeet voivat sisältää funktion parametreja tai ne voivat olla tyhjät. Koodi, jonka funktion tahdotaan suorittavan, tulee kaarisulkeiden {} sisälle.

```
function myFunction(a, b) {
  return a * b;
}
```

Kuvio 3. Funktio esimerkki

3.4 HTML:n ja JavaScriptin välinen toiminta

JavaScript voi vaikuttaa HTML-elementteihin monella tavalla. Se voi lisätä ja poistaa sekä muuttaa elementtejä. Se voi myös lisätä ja poistaa tai muuttaa niiden attribuutteja. JavaScript voi myös muuttaa verkkosivun CSS-ominaisuuksia. Siinä missä se voi muuttaa CSS:n kautta ulkonäköä tai muokata HTML rakennetta, se voi myös reagoida olemassa oleviin HTML:n tapahtumiin kuten napin painallukseen tai se voi jopa luoda uusia tapahtumia.

DOM (Document Object Model) on tapa, jolla selain järjestelee verkkosivun niin että siihen voi päästä käsiksi JavaScriptillä. Toisin sanoen DOM tarjoaa funktiot, joita koodaaja voi kutsua JavaScript koodissaan luodakseen toiminnallisia verkkosivuja. Esimerkiksi vaikka verkkokaupat tai selain pelit. Yksi yleisimmistä DOM funktioista on document.getElementById(), joka palauttaa elementin jonka id on annettu parametrinä funktiolle. Esimerkiksi document.getElementById(tekstikenttä).innerHTML = ”tekstipätkä123” hakisi sellaisen elementin jonka id olisi tekstikenttä. Tämän lisäksi innerHTML muuttaisi kyseisen elementin sisällön halutuksi tekstiksi. Tässä tapauksessa se

olisi tekstipätkä123. Myös CSS ominaisuuksia voi muutella tällä tavalla muuttamalla style-ominaisuutta. Esimerkiksi style.color muuttaisi elementin väriä. Elementtien lisäys ja poistaminen onnistuisi createElement()- ja element.remove()-funktioilla. Edellä mainittujen esimerkkien tavoin JavaScript voi siis DOM:n kautta muuttaa verkkosivun ulkoasua ja toimintoja.

4 Toimeksiantajan toiveet ja suunnittelu

Kun opinnäytetyö otettiin vastaan, oli opinnäytetyöehdotelmassa toiveena persoonallisuustesti, joka pitäisi toimia verkkoympäristössä. Sen pitäisi esittää käyttäjälle kysymykset ja laskea pisteet sekä muodostaa persoonallisuutta koskevat profiilit ja niille tulkinnat. Lisäksi toiveina oli vastaajien identiteetin suojaus, tulosten tyhjentyminen tietyn ajan kuluessa ja kopioinnin suojaus sekä maksuominaisuus, joka oikeuttaisi yhteen käyttökertaan.

Varsinaisessa suunnitteluvaiheessa käytiin läpi monia yksityiskohtia, joita tehtävän antajat ajattelivat olevan hyödyllistä lisätä vielä tai muuttaa. Suunnitelmia tehtäessä kävi selväksi, että käyttäjien identiteetin suojaus ei olisi tarpeen, koska ohjelmassa ei missään vaiheessa kerätä sellaisia yksilöiviä tietoja, joiden perusteella voitaisiin tunnistaa henkilö. Myöskään kopioinnin suojaus ei ole mahdollista, koska mikään ei estä käyttäjää kuvakaappaamaan tai muuten kopioimaan sitä minkä hän näkee. Animaatiot olivat myös yksi mahdollinen ominaisuus. Ideana oli sisällyttää animaatioita kysymysten joukkoon kannustamaan asiakasta täyttämään kyselyä. Tästä kuitenkin luovuttiin, sillä animaatiot todennäköisesti eivät toisi haluttua lisäarvoa ja olisivat todennäköisesti enemmän häiriötekijä.

Lopulta opinnäytetyön suunnitelmaan kuului kysymysten tallentaminen tietokantaan. Aluksi oli vielä ajatus siitä, että asiakkaiden vastaukset tallennettaisiin jonnekin, mutta myöhemmin tarkentui, että tällaista ominaisuutta ei tarvita. Tehtävänantajat toimittivat kysymykset kaikkine vastauksineen ja pisteineen sekä kommentteineen. Pisteiden osalta he kertoivat, millaisia tuloksia he toivoivat tulevan ja kuinka pääpiirteittäin ne pitäisi laskea. Tässä siis tehtäväksi jäi kehitellä laskuohjelma kokonaisuudessaan, jotta halutut tulokset tulisivat ulos. Ulkoasua koskevia toiveita olivat luontoaiheinen taustakuva sekä erilaiset laatikot erilaisille kysymyksille. Kysymyslaatikoissa olisi voinut esimerkiksi olla normaalit allekkain olevat vastausvaihtoehdot tai ne olisi voitu asettaa jana-

maisesti peräkkäin tai mahdollisesti luoda niistä aukeavat jatkokysymykset. Tässä vaiheessa ulkoasusta ei tullut juurikaan tämän tarkempaa ohjeistusta. Tulokset ja niiden arviointi piti myös näyttää pdf-muodossa asiakkaalle, jonka hän voisi tallentaa itselleen.

Maksuominaisuus oli liitetty toteutettavaksi suunnitteluvaiheessa. Mahdollisena maksukanavana oli Paytrailin tai Paypalin kaltainen palvelu. Tämä ominaisuus kuitenkin karsiutui opinnäytetyöstä kehittämisen aikana, kun kävi selväksi, että kyseinen palvelu olisi kuukausimaksullinen ja sen liittäminen olisi kannattavaa vasta kun tuote olisi kokonaan valmis myyntiin. Tämän opinnäytetyön sisällä tämä tuote ei tulisi täysin myyntivalmiiksi. Tämän tuotteen testaus ammattilaisilla ei myöskään kuulu opinnäytetyön sisältöön, koska se vaatisi lähes valmiin tuotteen.

Tehtävän antajien toiveen mukaisesti opinnäytetyö toteutettiin salassapitosopimuksen alaisuudessa sekä muiden opinnäytetyöhön liittyvien sopimusten mukaisesti. Tämän lisäksi myös opinnäytetyön tekstiosa liitettiin salassapitosopimuksen piiriin ja lisäksi siirrettiin tehtävän antajien omistukseen. Tämä sopimuksen muutostyö viivästytti sopimusten allekirjoittamista, mutta sillä ei ollut vaikutusta itse projektiin.

5 Toteutus

Tämän opinnäytetyön tarkoitus oli luoda kyselylomake, joka kävisi verkkosivujen yhteyteen linkkinä tai aivan itsenäisenä sivuna. Opinnäytetyön puitteissa tähän kehitettiin sivun perusrakenne HTML:llä ja toiminnallisuudet JavaScriptiä hyödyntäen. Kehitystyötä varten XAMPP:lla luotiin virtuaalinen palvelin. XAMPP on ilmainen avoimen lähdekoodin virtuaalipalvelin ohjelmisto, joka toimii myös Windowsilla. Sen kehittäjä on Apache Friends. Siitä löytyy esimerkiksi Apache http-palvelin.

5.1 5.1 Kyselylomakkeen toimintaprosessi

Liitteessä 2 on kuvattu kuinka kyselylomakkeen toiminnot etenevät. Aluksi kun käyttäjä avaa sivun tulee vastaan aloituslaatikko. Tämä sisältää lyhyen kuvauksen ja mainoksen kyselylomakkeesta. Käyttäjällä on vaihtoehtona painaa Seuraava-nappia. Seuraavaksi käyttäjälle näytetään käyttöehdot ja annetaan kaksi vaihtoehtoa. Hylkäämällä käyttöehdot käyttäjä palautetaan takaisin aloituslaatikkoon. Hyväksymällä käyttöehdot käyttäjä pääsee maksulaatikkoon. Tässä tulisi olemaan jatkokehityksen jälkeen maksamistoiminnot. Tässä vaiheessa käyttäjä voi painaa joko Maksa-nappia

tai Peruuta-nappia. Peruuttamalla käyttäjä päätyy aloituslaatikkoon. Maksamalla käyttäjä pääsee ohjeisiin, jotka luettuaan hän voi painaa Seuraava-nappia.

Tästä alkaa kyselyn kysymysten esittäminen. ensimmäisenä käyttäjältä kysytään taustatietoja. Annettuaan ne ja painettuaan Seuraava-nappia ohjelma tallentaa ne ja antaa ensimmäisen kysymyspatteriston kysymyksen. Kun käyttäjä on vastannut kysymykseen ja painanut Seuraava-nappia laskee ohjelma pisteet ja tarkastaa onko kysymyksellä jatkokysymystä. Jos on jatkokysymys, niin käyttäjä saa sen eteensä seuraavaksi. Jos kysymyksellä ei ole jatkokysymystä, nostattaa ohjelma seuraavaksi kysymysnumeroa. Tämän jälkeen se tarkistaa onko kysymysnumero yli 60 vai ei. Jos kysymysnumero ei ole yli 60, antaa ohjelma uuden kysymyksen. Jos kysymysnumero on yli 60, antaa kysely kiitosviestin, josta pääsee Tulokset-nappia painamalla katsomaan pisteitä ja lopuksi vielä pääsee katsomaan arviota itsestään PDF:ssä. PDF:n voi tallentaa halutessaan.

5.2 Kysymysten tallentaminen XML:ään

Kyselyssä on 60 kysymystä, jotka on tallennettu XML:ään. XML valittiin tietokannaksi, koska se on yksinkertainen ja helppo muokata ilman erityisiä käyttöliittymiä tai vaatimuksia. Se ei myöskään tarvitse mitään erityisiä kirjastoja tai palvelimia toimiakseen. Tämä auttaa myös tehtävän antajia, jos he tahtovat muokata kysymyksiä itse. XML ei välttämättä ole yhtä hyvä tietokanta kuin MYSQL tai muut vastaavat, mutta tähän tehtävään se tuli valituksi edellä mainitusta syystä. Aluksi rakenteessa yritettiin käyttää jokaiselle kysymykselle tehtyä omaa rakennetta, mutta tämä ei tuottanut haluttua tulosta. Ongelmana oli tiedonhakutapa. JavaScriptissä käytettiin getElementByTagName-hakua. Muita mahdollisia XML:n hakumenetelmiä oli esimerkiksi XPath ja XQuery. Luultavasti myös nämä olisivat olleet hyviä ja jopa joissain oloissa parempiakin hakumenetelmiä. Niistä ei kuitenkaan ollut löydettävissä käyttöön sopivia ohjeita. GetElementByTagName hakee järjestyksessä kaikki XML-elementit, joiden nimi halutaan hakea. Kun haetaan juuri tietyn kohdan arvo, käytetään indeksiä. Ongelmaksi tuli, jos jollain kysymyksellä ei ollut kyseistä tunniste-nimeä. Silloin haku meni seuraavaan kysymykseen, jolta oikea tunniste-nimi löytyi ja siten vääristi hakutulosta näyttämällä väärän kohdan arvon. Jos esimerkiksi haettaisiin kysymyksen 23 vastausta 5, mutta kysymyksellä 20 ei sellaista riviä olekaan. Tällöin indeksillä 23 haettu vastaus viisi olisikin oikeasti kysymyksen 24 vastaus 5.

Tämän jälkeen kehitettiin kaikille kysymyksille standardi tallennusrakenne (Liite 1), jossa käyttämättömät kohdat täytettiin vain kirjoittamalla sana tyhjä. Aluksi tähän rakenteeseen tallennettiin vain kysymysnumero, kysymys itsessään ja mahdolliset apukysymykset. Tämän lisäksi siihen tallennettiin 10 kysymystä ja niille vastausvaihtoehdot, pistetyypit, pisteet ja kommentit sekä 15 lisäkysymystä ja niille samat ominaisuudet kuin tavallisille kysymyksillekin. Myöhemmin tähän lisättiin myös kysymystyyppi sekä ketjukysymysnumero, koska eri kysymykset saattoivat näyttää ja toimia eri tavoin ja osa kysymyksistä oli ketjutettuja eli edeltävä kysymys määräsi seuraavan kysymyksen.

5.3 HTML runko ja ulkonäkö

Kyselyn runko ja perusulkonäkö toteutettiin HTML:llä. Kysely tapahtuu yhdellä sivulla vaihtamalla kulloinkin näkyvissä olevaa tekstilaatikkoa. Eli kaikki kyselyn osat ovat omissa HTML-laatikoissa, joita näytetään halutussa järjestyksessä tarpeen mukaan. Tekstilaatikko vaihtuu käyttäjän painaessa nappia. Alkutilanteessa on ainoastaan aloituslaatikko näkyvissä. Kaikki muu on piilotettuna css:llä tehdyllä luokalla (class) hide (Kuvio 4).

```
.hide {
    display: none;
}
```

Kuvio 4. CSS hide class

Taustakuvana on tässä vaiheessa käytetty verkosta löytynyttä viidakon kuvaa. CSS:ssä on määritelty mitä kuvatiedostoa siinä käytetään sekä kuvan venyttäminen koko taustan kokoiseksi (Kuvio 5).

```
:root {
    background-image: url("jungle.jpg");
    background-repeat: no-repeat;
    background-attachment: fixed;
    background-size: cover;
}
```

Kuvio 5. CSS taustakuvan asetukset

Käyttäjän avatessa verkkokyselyn hänelle ensin näytetään aloituslaatikko. Kaikkien laatikoiden perusulkoasu on määritelty myös CSS:llä taustaväristä aina sijaintiin ja marginaaleihin (Kuvio 6).

```
.container {
    top: 200px;
    width: 800px;
    max-width: 80%;
    background-color: white;
    border-radius: 5px;
    padding: 10px;
    box-shadow: 0 0 10px 2px;
    margin-left: auto;
    margin-right: auto;
    position: relative;
}
```

Kuvio 6. Containerin asetukset

Aloituksessa on lyhyt teksti houkuttelemaan käyttäjää sekä pieni esittely kyselystä. Tämän jälkeen käyttäjä pääsee katsomaan käyttöehdot painamalla Käyttöehdot-nappia (Kuvio 7).

Kuvio 7. Esittelysivu

Käyttöehtojen kohdalla käyttäjälle tarjotaan joko hyväksy tai hylkää vaihtoehdot. Hylkää vaihtoehto johtaisi takaisin aloitukseen. Hyväksy vaihtoehto vie käyttäjän maksulaatikkoon, jossa tällä

hetkellä ei ole kuin Maksa- ja Peruuta-napit. Tämän jälkeen käyttäjälle vielä annetaan ohjeet ja sitten käyttäjä pääsee täyttämään itse kyselyä.

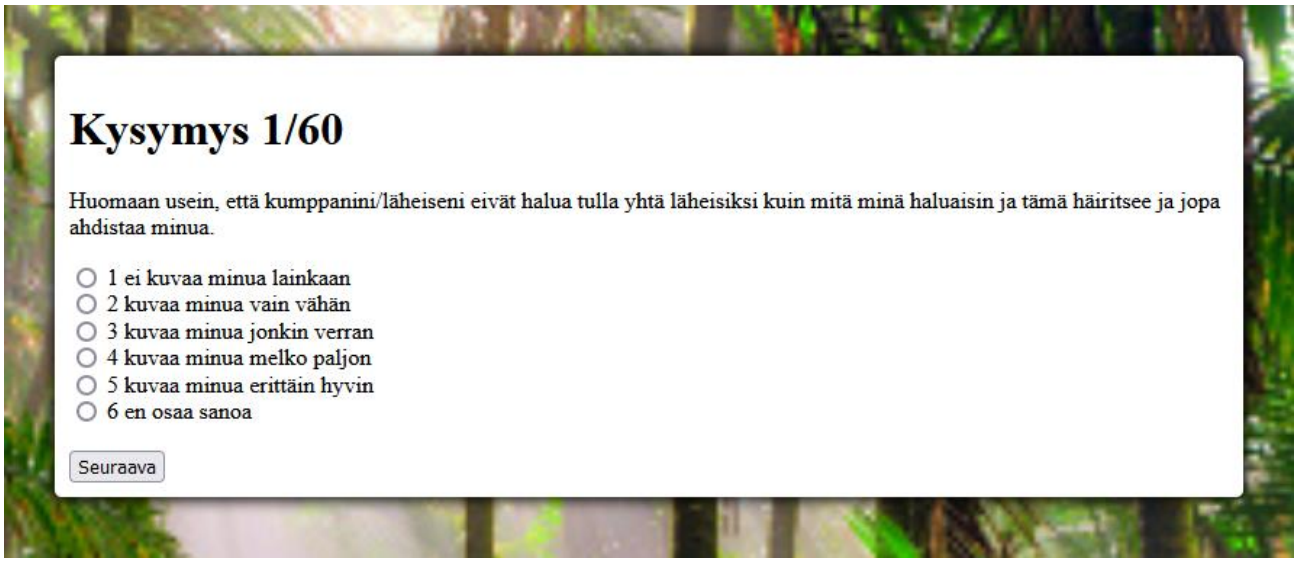
Kuvio 8. Taustatietokysymykset

Ensin kysytään käyttäjän taustoista yleisluontoisia asioita kuten ikää, sukupuolta ja etnisyyttä (Kuvio 8). Vastattuaan niihin käyttäjälle esitetään 60 sattumanvaraisessa järjestyksessä esitettyä kysymystä, joilla on 15 erilaista rakennetta. Kullekin erilaiselle kysymystyypille on tehty oma kysymyslaatikko (Kuvio 9).

```
<div id="questionTypeBox1" class="hide">
    <h1 id="questionNumber1"></h1>
    <p id="demo1"></p>
    <input type="radio" id="question11" name="questionRadio1" value="1">
    <label id="label11" for="question11"></label><br>
    <input type="radio" id="question12" name="questionRadio1" value="2">
    <label id="label12" for="question12"></label><br>
    <input type="radio" id="question13" name="questionRadio1" value="3">
    <label id="label13" for="question13"></label><br>
    <input type="radio" id="question14" name="questionRadio1" value="4">
    <label id="label14" for="question14"></label><br>
    <input type="radio" id="question15" name="questionRadio1" value="5">
    <label id="label15" for="question15"></label><br>
    <input type="radio" id="question16" name="questionRadio1" value="6">
    <label id="label16" for="question16"></label><br>
    <br>
    <button id="nextButton1" onclick="nextFunction1()">Seuraava</button>
</div>
```

Kuvio 9 Esimerkki kysymyslaatikon HTML rakenteesta

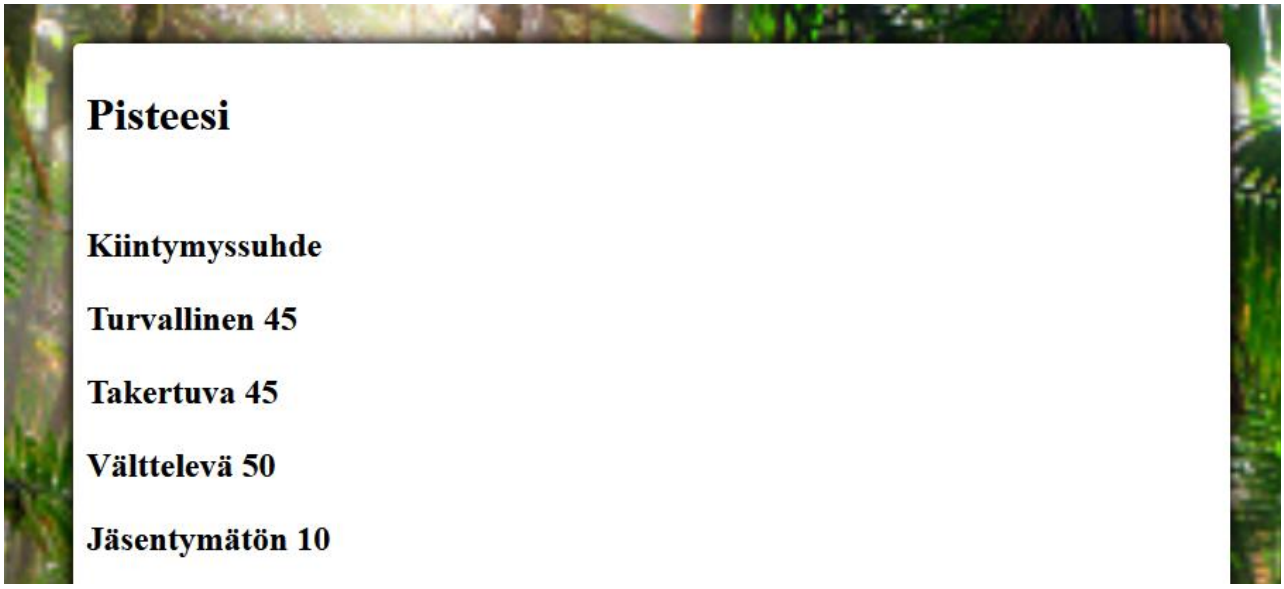
Aina kun siirrytään uuteen kysymykseen, valitaan kyseiselle kysymykselle sopiva laatikko ja täytetään se uuden kysymyksen tiedoilla. Samalla pyyhitään vanhan kysymyksen vastaukset ja jos kysymys sattuisi olemaan samanlainen kuin edeltävä silloin vain uuden kysymyksen tiedot korvaavat vanhan. Näin toteutettiin toimeksiantajan toivetta, että kysymyksissä ei pääsisi taaksepäin muuttelemaan vastauksia. Vanhoihin kysymyksiin ei pääse edes käyttämällä selaimen taaksepäin nappia. Taaksepäin napin käytön jälkeenkin eteenpäin nappi veisi alkuun kyselyn esittelysivulle. Alkutilanteessa kaikki laatikot ovat piilotettuja. Kysymysten esitys tapa voi vaihdella aina normaalista monivalinnasta (Kuvio 10) janamaiseen monivalintaan (Kuvio 11) ja jatkokysymyksiin. Huomaa, että kysymysnumerolla itsellään ei ole yhteyttä siihen mikä kysymys näytetään.



Kuvio 10. Esimerkki normaalista monivalinnasta

Kuvio 11. Esimerkki janamaisesta monivalinnasta

Kysymykset esitetään yksi kerrallaan ja uuden saa vasta kun edelliseen on vastattu. Kun kaikkiin kysymyksiin on vastattu, saa käyttäjä kiitosviestin ja mahdollisuuden mennä katsomaan tuloksiaan (Kuvio 12). Tuloksien yhteydessä annetaan mahdollisuus vielä katsoa tuloksien yhteenveto PDF-muodossa ja halutessaan hän voi ladata tämän PDF:n.



Kuvio 12. Osa tulosnäkymästä

5.4 Taustatoiminnot

Siinä missä HTML:llä luodaan verkkosivun runko ja ulkonäkö, on JavaScriptillä luotu kullekin napille toiminnot, joista käyttäjä näkee vain lopputuloksen. Kyselyn alussa mainitut esittelylaatikko, käyttöehdot, maksulaatikko sekä ohjeet ovat yksinkertaisia siirry seuraavaan ruutuun toimintoja pääosin, mutta jo näiden vaiheiden aikana alustetaan kyselylomaketta. Siinä vaiheessa, kun käyttäjä on maksanut eli tässä tapauksessa mennyt maksulaatikon ohi ja painanut Ohjeisiin-nappia ja on lukemassa ohjeita, ladataan jo XML valmiiksi muuttujaan xmlDoc. Kuviossa 13 on JavaScript funktio, joka tapahtuu Ohjeisiin-nappia painettaessa. Se piilottaa edellisen maksulaatikon (paymentBox) ja näyttää ohjeet (instructionsBox). Samalla se ajaa shuffleArray (numberArray)- ja loadXMLDoc()-funktiot. loadXMLDoc () on functio joka lataa XML:n tiedot xmlDoc-muuttujaan (Kuvio 14).

```
function instructionsFunction() {
    var a = document.getElementById("paymentBox");
    a.style.display = "none";
    var b = document.getElementById("instructionsBox");
    b.style.display = "block";
    shuffleArray (numberArray);
    loadXMLDoc();
}
```

Kuvio 13. Ohjeisiin meno funktio

```
function loadXMLDoc() {
    var xmlhttp = new XMLHttpRequest();
    xmlhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      innerLoadXMLDoc(this);
    }
    };
    xmlhttp.open("GET", "questions_fi.xml", true);
    xmlhttp.send();
}

function innerLoadXMLDoc(xml) {
xmlDoc = xml.responseXML;
}
```

Kuvio 14. XML:n lataus funktio(t)

Tämä tehdään jo tässä vaiheessa siksi, että aikaisemmissa versioissa, joissa oli yritetty ladata kysymykset XML:stä tarpeen mukaan, kun kysymystä vaihdettiin, johti erilaisiin tilanteisiin, joissa kysymykset saattoivat laahata jäljessä tai pisteitten lasku ei onnistunut, kun ulosanti oli myöhässä. Se johtuu siitä, että xml:stä ladattaessa tietoja niiden siirto palvelimelta takaisin ei tapahdu sillä hetkellä kuin koodi itsessään käskee tekemään vaan se vie hetken aikaa koska tämä funktio on asynkroninen. Samassa vaiheessa myös sekoitetaan numberArray-taulukko (array) käyttäen Fisher-Yates menetelmää. Tämän toteuttaa shuffleArray()-funktio (Kuvio 15). numberArray sisältää kaikki kysymysnumerot paitsi numeron 39. Sekoitettua taulukkoa käytetään menetelmänä, jolla kysymykset saadaan sattumanvaraiseen järjestykseen niiden ID:n perusteella. Kysymysnumero 39 on varattu ketjukysymystyypille.

```
function shuffleArray (array) {
    for (let i = array.length - 1; i > 0; i--) {
    let j = Math.floor(Math.random() * (i + 1));
    [array[i], array[j]] = [array[j], array[i]];
  }
}
```

Kuvio 15. Kysymysten sekoittaminen

Ensimmäisenä kyselyssä kysytään käyttäjän ikää, sukupuolta ja etnisyyttä. Nämä tiedot tallennetaan muuttujiin myöhempää käyttöä varten. Tämän jälkeen alkaa varsinainen kysely. Kysymysten esittämisen aloittaa firstQuestionFunction()-funktio joka tapahtuu Seuraava-nappia painettaessa kun taustatiedot on täytetty. Se tarkistaa, että taustakysymyksiin on vastattu. Tämän jälkeen se tallentaa ne. Seuraavaksi kasvatetaan jokaisessa kysymyslaatikossa esitettävää kysymysnumeroa questionNumberHeaderFunction() -funktiolla. Kyseinen funktio kasvattaa muuttujan questionNumberHeaderVar arvoa yhdellä ja asettaa sen kunkin kysymyslaatikon otsikkoon. Seuraavaksi ladataan muuttujaan questionNumberVar numberArray[numberVar]:sta muuttujan numberVar vastaava arvo miinus yksi. Tämän jälkeen kutsutaan funktiota loadFirstQuestionFunction(), jonka tehtävänä on ladata ensimmäinen kysymys.

loadFirstQuestionFunction()-funktion (Kuvio 16) ainoa tehtävä on ladata ensimmäinen kysymys ja tämän jälkeen aina, kun siirrytään uuteen kysymykseen, käytetään funktiota loadNextQuestionFunction() (Kuvio 17). Uutta kysymystä ladatessa ne tarkastavat mikä kysymystyyppi seuraava kysymys on ja täyttävät sitä vastaavan kysymyslaatikon kysymyksen sekä vastausvaihtoehdot ja näyttää ne. Näiden funktioiden ero on se, että jälkimmäinen ottaa myös huomioon kysymysketjun. Ensimmäisessä kysymyksessä tätä ei tarvitse huomioida, koska kysymysketju otetaan edellisestä kysymyksestä ja sitä ei vielä ole. Kysymysketju, jos sen arvo on muuta kuin tyhjä, määrää seuraavan kysymyksen. Nämä funktiot toteuttavat kysymyksen valinnan ja näyttämisen. loadFirstQuestionFunction()-funktiossa ei ole muuttujan numberVar arvon kasvattamista, koska tässä vaiheessa sitä ei vielä tarvitse tehdä eikä siinä ole myöskään ole tarkastusta siitä, että joko päästiin kyselyn loppuun, koska kyseessä on ensimmäinen kysymys.

```
function loadFirstQuestionFunction() {
    console.log("id: " + xmlDoc.getElementsByTagName("id")[questionNumberVar].childNodes[0].nodeValue);
    questionTypeVar = xmlDoc.getElementsByTagName("kysymystyyppi")[questionNumberVar].childNodes[0].nodeValue;
    if (questionTypeVar == 1){
        textVar = xmlDoc.getElementsByTagName("teksti")[questionNumberVar].childNodes[0].nodeValue;
        document.getElementById("demo1").innerHTML = textVar;
        choiceVar1 = xmlDoc.getElementsByTagName("vaihtoehto1")[questionNumberVar].childNodes[0].nodeValue;
        document.getElementById("label11").innerHTML = choiceVar1;
        choiceVar2 = xmlDoc.getElementsByTagName("vaihtoehto2")[questionNumberVar].childNodes[0].nodeValue;
        document.getElementById("label12").innerHTML = choiceVar2;
        choiceVar3 = xmlDoc.getElementsByTagName("vaihtoehto3")[questionNumberVar].childNodes[0].nodeValue;
        document.getElementById("label13").innerHTML = choiceVar3;
        choiceVar4 = xmlDoc.getElementsByTagName("vaihtoehto4")[questionNumberVar].childNodes[0].nodeValue;
        document.getElementById("label14").innerHTML = choiceVar4;
        choiceVar5 = xmlDoc.getElementsByTagName("vaihtoehto5")[questionNumberVar].childNodes[0].nodeValue;
        document.getElementById("label15").innerHTML = choiceVar5;
        choiceVar6 = xmlDoc.getElementsByTagName("vaihtoehto6")[questionNumberVar].childNodes[0].nodeValue;
        document.getElementById("label16").innerHTML = choiceVar6;
        var a = document.getElementById("questionTypeBox1");
        a.style.display = "block";
    }
```

Kuvio 16. Osa ensimmäisen kysymyksen lataamis funktiosta

```
function loadNextQuestionFunction() {
    questionChainVar = xmlDoc.getElementsByTagName("kysymysketju")[questionNumberVar].childNodes[0].nodeValue;
    if (questionNumberHeaderVar == 61) {
        var a = document.getElementById("endBox");
            a.style.display = "block";
    }
    else {
        if (questionChainVar == "tyhjä"){
            numberVar++;
            questionNumberVar = numberArray[numberVar]-1;
            questionTypeVar = xmlDoc.getElementsByTagName("kysymystyyppi")[questionNumberVar].childNodes[0].nodeValue;
            if (questionTypeVar == 1){
                textVar = xmlDoc.getElementsByTagName("teksti")[questionNumberVar].childNodes[0].nodeValue;
                document.getElementById("demo1").innerHTML = textVar;
                choiceVar1 = xmlDoc.getElementsByTagName("vaihtoehto1")[questionNumberVar].childNodes[0].nodeValue;
                document.getElementById("label11").innerHTML = choiceVar1;
                choiceVar2 = xmlDoc.getElementsByTagName("vaihtoehto2")[questionNumberVar].childNodes[0].nodeValue;
                document.getElementById("label12").innerHTML = choiceVar2;
                choiceVar3 = xmlDoc.getElementsByTagName("vaihtoehto3")[questionNumberVar].childNodes[0].nodeValue;
                document.getElementById("label13").innerHTML = choiceVar3;
                choiceVar4 = xmlDoc.getElementsByTagName("vaihtoehto4")[questionNumberVar].childNodes[0].nodeValue;
                document.getElementById("label14").innerHTML = choiceVar4;
                choiceVar5 = xmlDoc.getElementsByTagName("vaihtoehto5")[questionNumberVar].childNodes[0].nodeValue;
                document.getElementById("label15").innerHTML = choiceVar5;
                choiceVar6 = xmlDoc.getElementsByTagName("vaihtoehto6")[questionNumberVar].childNodes[0].nodeValue;
                document.getElementById("label16").innerHTML = choiceVar6;
                var a = document.getElementById("questionTypeBox1");
                a.style.display = "block";
            }
```

Kuvio 17. Osa loppujen kysymysten lataamis funktiosta

Seuraavissa kysymyksissä loadNextQuestionFunction() ensin tarkastaa joko päästiin loppuun ja jos päästiin, niin näyttää viestin jossa kiitellään vastauksista ja josta pääsee katsomaan tuloksia. Jos taas kysymysnumero on alle 61 katsotaan, onko ketjukysymystyyppi tyhjä vai onko sillä joku numeerinen arvo. Jos sillä on numeerinen arvo, niin valitaan kyseistä numeroa vastaava kysymys. Jos taas ketjukysymystyyppi on tyhjä, kasvatetaan muuttujan numberVar arvoa yhdellä ja käydään sen arvoa vastaava kysymysnumero hakemassa numberArraysta ja tallennetaan tämä questionNumberVar-muuttujaan. Huomaa että numberVar:n arvo viittaa taulukon indeksiin eikä esitä itsessään

valittua kysymystä, vaan questionNumberVar on tämän valitun kysymyksen numero. Sen jälkeen käydään hakemassa questionNumberVar vastaavan kysymyksen kysymystyyppi ja tallennetaan se muuttujaan. Sitten valitaan kysymystyyppiä vastaava kysymyslaatikko ja täytetään se kysymyksellä. Samalla myös muutetaan kyseisen kysymyslaatikon elementit näkyviksi.

Kaikilla kysymystyypeillä on oma funktio Seuraava-napille. Esimerkiksi seuraavaFunction1() (Kuvio 18). Näiden funktioiden sisällä tapahtuu pisteiden laskenta funktio CountPointsFunction1(), radionappien tyhjennys funktio radioClear(), kysymystyyppien resetointi questiontypeClear() ja kysymyksen otsikossa olevan kysymysnumeron kasvattaminen questionNumberHeaderFunction() sekä loadNextQuestionFunction().

```
function nextFunction1() {
    if(document.querySelector('input[name="questionRadio1"]:checked').value != null){
        countPointsFunction1();
        writePDF1();
        radioClear();
        questiontypeClear();
        questionNumberHeaderFunction();
        loadNextQuestionFunction();
    }
    else {

    }
}
```

Kuvio 18. Esimerkki seuraavaan kysymykseen siirtymis funktiosta

Lisäksi siellä on kysymysten tallennus funktio writePDF1(), jonka tehtävänä on tallentaa kysymyksistä tiedot siitä mikä kysymys oli, mikä oli sen vastaus, mahdollinen jatkokysymys ja vastaus sekä kommentit pdfArray-taulukkoon myöhempää käyttöä varten. Radionappien tyhjennys tapahtuu radioClear() (Kuvio 19) funktion sisällä for-silmukoilla. For-silmukat käy läpi kunkin radionappiryhmän ja asettaa kaikki radionappien checked asetuksen falseksi.

```
function radioClear() {
    var elel = document.getElementsByName("ikäradio");
    for(var i=0;i<elel.length;i++){
        elel[i].checked = false;
    }
```

Kuvio 19. Radiobuttonien valintojen poisto funktio

Kysymystyyppien resetoinnin tekee questiontypeClear()-funktio joka piilottaa kaikki kysymyslaatikot jotta seuraava kysymys näkyisi oikeassa laatikossa ja väärät laatikot eivät näkyisi sen alla. Joillakin kysymyksillä voi olla vastauksesta riippuen jatkokysymys. Näille kysymyksille on tehty kullekin kysymystyypille oma jatkofunktio, joka avaa kysymyksen jatkokysymys-osion. Tämä toimii siten, että käyttäjä on vastannut ensimmäiseen osioon ja painanut Seuraava-nappia, jolloin on tarkastettu if-lauseella, tuleeko vastauksen jälkeen jatkokysymys vai ei. Jos tulee jatkokysymys, niin piilotetaan alkuperäisen kysymyksen elementit ja paljastetaan jatkokysymys-osion elementit. Kun tähänkin on vastattu, piilotetaan jatkokysymys-elementit myöskin.

Pisteiden lasku tapahtuu pisteidenlasku funktioilla countPointsFunction1() (Kuvio 20). Kullekin kysymystyypille on tehty sille sopiva oma laskuri. Jos kysymyksellä on myös jatkokysymys, ottaa tämän kysymystyypin jatkokysymyslaskuri hoitaakseen jatkokysymyksen pisteiden laskennan. Ensimmäisenä laskuri käy hakemassa xmlDoc:sta kysymyksen kysymysryhmä arvon ja tallentaa sen muuttujaan questionGroupVar. Tämän jälkeen katsotaan mihin kysymysryhmään kysymys kuului if-lauseilla.

```
function countPointsFunction1() {
    questionGroupVar = xmlDoc.getElementsByTagName("kysymysryhmä")[questionNumberVar].childNodes[0].nodeValue;
    if (questionGroupVar == "kiintymyssuhdetyylit"){
        if (parseInt(document.querySelector('input[name="questionRadio1"]:checked').value) == "1") {
            pointsTypeVar1 = xmlDoc.getElementsByTagName("pistetyyppi1")[questionNumberVar].childNodes[0].nodeValue;
            pointsVar1 = xmlDoc.getElementsByTagName("pisteet1")[questionNumberVar].childNodes[0].nodeValue;
            if (pointsTypeVar1 == "turvallinen"){
                (affectionSecurityPoinstVar) = parseInt(affectionSecurityPoinstVar) + parseInt(pointsVar1);
            }
            if (pointsTypeVar1 == "takertuva"){
                (affectionClingPointsVar) = parseInt(affectionClingPointsVar) + parseInt(pointsVar1);
            }
            if (pointsTypeVar1 == "välttelevä"){
                (affectionAvoidancePointsVar) = parseInt(affectionAvoidancePointsVar) + parseInt(pointsVar1);
            }
            if (pointsTypeVar1 == "jäsentymätön"){
                (affectionUnstructuredPointsVar) = parseInt(affectionUnstructuredPointsVar) + parseInt(pointsVar1);
            }
        }
```

Kuvio 20. Esimerkki pisteidenlasku funktiosta

Kysymysryhmällä voi olla 3 erilaista arvoa. Sitten haetaan document.querySelector:lla kyseisen kysymyksen questionRadio1-valintanappiryhmän valittu arvo ja if-lauseilla verrataan sitten mikä vastaus se oli. Sitten haetaan taas xmlDoc:sta getElementsByTagName-toiminnolla vastauksen pistetyyppi ja pisteet. Ne tallennetaan vastaavan nimisiin muuttujiin. Pistetyyppejä (pointsTypeVar1) voi olla 4 erilaista. Tämän jälkeen if-lauseella katsotaan mikä pistetyyppi se on ja valitaan oikea muuttuja, johon lisätään nämä pisteet. Näitä muuttujia voi olla 12 erilaista, mutta edellä mainitut

if-lauseet johtavat oikeaan muuttujaan. Tämä ratkaisu tehtiin, koska halutiin seurata 4:ää erilaista pistetyyppiä kolmessa erilaisessa kysymysryhmässä kussakin.

Kun kaikki kysymykset on käyty läpi, saa käyttäjä eteensä laatikon, jossa kiitetään vastauksista ja annetaan nappi päästä eteenpäin. Seuraavana käyttäjälle näytetään yhteenveto pisteistä showPointsFunction()-funktiolla (Kuvio 21), joka siis piilottaa edellisen laatikon ja asettaa yhteen lasketut pisteet paikoilleen ja näyttää ne.

```
function showPointsFunction(){
    var a = document.getElementById("endBox");
    a.style.display = "none";
    var b = document.getElementById("pointsBox");
    b.style.display = "block";
    document.getElementById("affectionSecurityP").innerHTML = "Turvallinen " + affectionSecurityPoinstVar;
    document.getElementById("affectionClingP").innerHTML = "Takertuva " + affectionClingPointsVar;
    document.getElementById("affectionAvoidanceP").innerHTML = "Välttelevä " + affectionAvoidancePointsVar;
    document.getElementById("affectionUnstructuredP").innerHTML = "Jäsentymätön " + affectionUnstructuredPointsVar;

    document.getElementById("parentSecurityP").innerHTML = "Turvallinen " + parentSecurityPoinstVar;
    document.getElementById("parentClingP").innerHTML = "Takertuva " + parentClingPoinstVar;
    document.getElementById("parentAvoidanceP").innerHTML = "Välttelevä " + parentAvoidancePoinstVar;
    document.getElementById("parentUnstructuredP").innerHTML = "Jäsentymätön " + parentUnstructuredPoinstVar;

    document.getElementById("feelSecurityP").innerHTML = "Turvallinen " + feelingSecurityPoinstVar;
    document.getElementById("feelClingP").innerHTML = "Takertuva " + feelingClingPoinstVar;
    document.getElementById("feelAvoidanceP").innerHTML = "Välttelevä " + feelingAvoidancePoinstVar;
    document.getElementById("feelUnstructuredP").innerHTML = "Jäsentymätön " + feelingUnstructuredPoinstVar;
}
```

Kuvio 21. Pisteiden näyttämis funktio

Tässä yhteydessä annetaan käyttäjälle myös mahdollisuus katsoa kattavampi tulosarvio PDF-muodossa. PDF tiedoston luomiseen on netissä muutama erilainen vaihtoehto kuten pdfmake, react-PDF ja jsPDF. Tähän projektiin tuli valituksi jsPDF, koska se oli vaihtoehdoista yksinkertaisin ja helpoin käyttää. PDF:n ensimmäisellä sivulla näytetään kuva, joka valitaan korkeimman arvon saaneen pistetyypin mukaan. Tätä varten kunkin kysymysryhmän pistetyypit lasketaan tyypeittäin yhteen. Kuvan näyttäminen ei ollut aivan yksinkertaista, koska kuva piti kääntää Base64 tekstiformaattiin. Tätä varten netissä on monia valmiita kääntäjiä, mutta koska tämä projekti tehtiin salassapitosopimuksen alaisuudessa, tehtiin tätä varten oma kääntäjä netistä löytyneiden ohjeiden mukaisesti (Kuvio 22). Tämän jälkeen base64 tekstijono tallennettiin muuttujaan. Tämä tehtiin kaikille neljälle kuvalle.

```
document.getElementById('button').addEventListener('click', function() {
  var files = document.getElementById('file').files;
  if (files.length > 0) {
    getBase64(files[0]);
  }
});

function getBase64(file) {
   var reader = new FileReader();
   reader.readAsDataURL(file);
   reader.onload = function () {
     document.getElementById("as").innerHTML = reader.result;
   };
   reader.onerror = function (error) {
     console.log('Error: ', error);
   };
}
```

Kuvio 22. JavaScript kuvan muuttamiseksi base64 muotoon

Kuvan liittäminen PDF:n sivulle tapahtui jsPDF:n addImage()-toiminolla johon syötettiin valittu base64-tekstijono, kuvan tyyppi, esimerkiksi ’JPEG’, sekä kuvan vasen ylänurkka ja haluttu kuvan koko millimetreinä. tässä tapauksessa kuva asetettiin A4 kokoisena. Seuraavalle sivulle tuli kuvan mukainen kuvaus käyttäjästä kertomaan millainen hän on. Tekstin liittäminen PDF:n tapahtui jsPDF:n text()-toiminnolla (Kuvio 23). Tähän syötetään marginaali vasemmalta ja aloituskohta sivun yläreunasta millimetreinä sekä haluttu teksti. Uusi sivu luotiin taas addPage()-toiminnolla. Kolmannella sivulla käyttäjälle annettiin pisteet. Seuraaville 12 sivulle tulostettiin kysymykset sekä käyttäjän vastaukset ja mahdolliset kommentit perustuen vastauksiin.

```
if(sortedpdfArray[item].questionType == 1){
    var quest = sortedpdfArray[item].id + ". " + sortedpdfArray[item].question;
    pageTopVar = pageTopVar + 20;
    pdfCustomer.text(10, pageTopVar, quest);
    var answa = "Vastasit " + sortedpdfArray[item].answer;
    pageTopVar = pageTopVar + 20;
    pdfCustomer.text(10, pageTopVar, answa);
    var comm = sortedpdfArray[item].comment;
    if(comm != "tyhjä"){
        pageTopVar = pageTopVar + 20;
        pdfCustomer.text(10, pageTopVar, comm);
    }
    pageTopVar = pageTopVar + 20;
    pdfCustomer.text(10, pageTopVar, '---------------------------------------
}
```

Kuvio 23. Osa PDF:n luomis funktiota

Nämä kysymykset tulostettiin XML:ssä olevan id arvon mukaiseen järjestykseen ja samalla ryhmiteltiin kolmeen eri kysymysryhmään. Varsinainen PDF generointi tapahtui vasta save() toiminnolla. Lopuksi Käyttäjä voi tallentaa halutessaan PDF:n samalla tavalla kuin minkä tahansa muunkin internetissä selaimella näytetyn PDF:n koneelleen.

6 Työn tulokset ja jatkokehitys mahdollisuudet

6.1 Tulokset

Työn tuloksena asiakkaalle on kehitetty verkkokyselyn toiminnallinen runko, joka sisältää ohjeet sekä kaikki 60 kysymystä, jotka näytetään sattumanvaraisessa järjestyksessä yksi kerrallaan sekä pelkistetyn tulosten näyttämisen. Tämän lisäksi siihen on tehty tietokanta johon kysymykset ovat tallennettu sekä pelkistetty PDF:n luontiominaisuus, joka tulostaa kuvauksen, pisteet ja kysymykset ryhmiteltyinä kysymysryhmittäin sekä mitä niihin on vastattu ja vastausten mukaiset kommentit, jos niillä sellainen oli.

Samalla ratkaistiin tuotteeseen liittyvät tutkimuskysymykset. Ensimmäisessä kohdassa piti ratkaista, kuinka kysymykset tallennetaan tietokantaan sekä kuinka ne näytetään käyttäjälle sattumanvaraisessa järjestyksessä. Tämä ratkaistiin luomalla XML:ään standardi kysymysrakenne ja kaikki kysymykset tallennettiin siihen. Kysymysten sotkemiseksi luotiin numerolista, joka sekoitettiin ja sitten aina näytettiin sen hetkistä numeroa vastaava kysymys. Toisena kysymyksenä oli, kuinka laskea pisteet. Tätä varten ryhmiteltiin pisteet 12 ryhmään ja laskettiin yhteen riippuen mihin ryhmään kyseinen piste kuului. Tätä varten kullekin kysymykselle määriteltiin kysymysryhmä ja kullekin pisteelle pistetyyppi tietokantaan. Kolmantena piti ratkaista kuinka luoda vastauksista ja niiden arvioinnista PDF. Tätä varten käytettiin valmista jsPDF-kirjastoa, joka oli helpoin käyttää niistä vaihtoehdoista, joita oli suositeltu internetissä.

Työllä on myös ei aineellisia tuloksia. Tämän projektin aikana minä opin suunnitelman tärkeyden hyvinkin selvästi, kun projekti eteni. Mitä tarkempi suunnitelma sitä vähemmän tulee muutostarpeita sekä yllätyksiä. Myöskin tässä tuli paljon kerrattua ja opittua verkkosivujen tekemisestä kuten esimerkiksi asynkronisen xml haun vaikutuksista tai vaikkapa tällaisen laajemman dokumentin kirjottamisesta.

6.2 6.2 Matkalla kohdattuja ja ratkaistuja ongelmia

Ohjelmaa kehitettäessä vastaan tuli huomattava määrä erilaisia ongelmia. Osa niistä oli pieniä kirjoitusvirheitä, kun taas toiset saivat aika isojakin mittasuhteita. Suurin osa kohdatuista ongelmista oli puuttuvia pilkkuja ja puolipisteitä, joiden kerrannaisvaikutus saattoi välillä kasvaa huomattavaksi, kun paljon koodi pätkiä vain kopioitiin ja liitettiin ajan säästämiseksi. Myös muut kirjoitusvirheet kasvoivat arvaamattomiksi ongelmiksi samalla tavalla. Esimerkiksi yksi ongelma oli näennäisesti sattumanvaraisesti ilmaantuva undefined-arvo kysymysten seassa. Ongelman tutkiminen ja paikantaminen vei usean tunninverran aikaa, mutta lopulta kyse oli vain pienestä kirjoitusvirheestä ja puolipisteestä väärässä paikassa. Toinen huomattava ongelma oli kysymysten lataaminen kysymyslaatikoihin, kun aluksi ei ollut käytössä standardisoitua kysymysmallia XML:ssä. Tämä ongelma ratkesi, kun kysymyksille loi XML:ssä standardin kysymysrakenteen. Kolmas iso ongelma oli myös XML:n kysymysten lataamiseen liittyvä. Aluksi tarkoitus oli vain ladata xml:stä kysymyslaatikoihin tiedot kysymys kerrallaan. Tämä ei kuitenkaan toiminut, koska kysymysten lataamisessa käytettiin asynkronista tiedon siirtoa. Ongelma näkyi yleensä siten että kysymykset näyttivät olevan aina yhdellä jäljessä, kun laskettiin pisteitä. Näin ollen pisteiden lasku ei onnistunut lainkaan. Ongelma ratkaistiin lataamalla jo hyvissä avoin koko XML muuttujaan ja sitten muuttujasta haettiin kysymykset tarpeen mukaan.

6.3 Jatkokehitysmahdollisuudet

Ohjelmalla on vielä huomattava määrä jatkokehityskohteita ja tarvetta ennen kuin se on valmis tuote. Se tarvitsee vielä PDF:n sisällön hiomista, koska tehtävän antajien mielestä yli 15 sivua pitkä PDF ei vastaa heidän asiakkaitten tarpeita ja on liian pitkä. Tämän lisäksi kysely pitää vielä käyttää asiantuntijoiden testauksessa. Testauksen jälkeen on mahdollista, että kyselyä on korjattava tai muokattava sellaiseen suuntaan, jotta se paremmin kattaisi tarpeen. Myös maksuominaisuus on lisättävä. Maksuominaisuuden lisääminen ei kuitenkaan ole mahdollista ennen kuin asiakas on tyytyväinen muihin osiin, koska kyseinen ominaisuus on kuukausimaksullinen palvelu. Tällaisia palveluita tarjoaa muun muassa PayPal ja Paytrail. Englanninkielinen versio on myös suositeltavaa toteuttaa, jotta tuotetta voisi käyttää myös muut kuin suomenkieliset. Tämä tarvinnee jonkin verran lisäyksiä koodiin sekä mahdollisesti jonkin kielipalvelun osaamista, jotta käännökset olisivat oikein tehtyjä. Koska tämä kysely tulee yhdeksi tuotteeksi linkin taakse isompaan verkkosivukokonaisuu-

teen, tulisi myös taustakuva ja kyselyn värimaailma tehdä tämän sivuston kanssa sopivaksi. Samalla kuitenkin on tarkoitus, että tämä kysely toimisi itsenäisenäkin. Myös on harkittava tämän kyselyn sekä isomman verkkosivukokonaisuuden palvelimen palveluiden tarjoajaa.

Lähteet

Aaltonen J. 2006. Perheterapia psykoterapian muotona. Lääketieteellinen Aikakauskirja Duodecim, 122, 6, 724–731, Suomalainen Lääkäriseura Duodecim. Viitattu 13.02.2022. https://www.duodecimlehti.fi/duo95607

Codesido I. 2009, What is front-end development?. The Guardian. Viitattu 13.02.2022. https://www.theguardian.com/help/insideguardian/2009/sep/28/blogpost

Huttunen M. O. 2017. Mielenterveyden häiriöt. Lääkkeet mielen hoidossa. Duodecim Terveyskirjasto. Kustannus Oy Duodecim. Viitattu 14.03/2022. https://www.terveyskirjasto.fi/lam00002#s1

Sinkkonen J. Kiintymysihdeteoria – tutkimus löydöksistä käytännön sovelluksiin. Duodecim, 120, 1866-1873. Viitattu 04.05.2022. https://www.terveyskirjasto.fi/xmedia/duo/duo94437.pdf

Kriisi parisuhteessa, N.d. Lounais-Suomen mielenterveys ry. Viitattu 13.02.2022. https://www.mielenterveysseurat.fi/turku/materiaalit/tietoa-kriiseista/kriisi-parisuhteessa/

Lukka K. 2001. Konstruktiivinen tutkimusote. Metodix Oy. Viitattu 02.05.2022. https://metodix.fi/2014/05/19/lukka-konstruktiivinen-tutkimusote/

Parisuhteen vaikeudet. 2021. Mannerheimin Lastensuojeluliitto. Viitattu 30.01.2022. https://www.mll.fi/vanhemmille/tietoa-lapsiperheen-elamasta/vanhemman-hyvinvointi/parisuhde/parisuhteen-vaikeudet/

Perheterapia N.d. Väestöliiton Hyvinvointi Oy. Viitattu 30.01.2022 https://vaestoliitonterapia.fi/terapiat/perheterapia/?gclid=CjwKCAiA_omPBhBBEiwAcg7smW6Yaj7qAhdrnYjgJg_rSD_OEYwCSAglPJM2vVB5W34-defZsvwBjxoCNTQQAvD_BwE

Psykoterapia. N.d. Helsingin ja Uudenmaan Sairaanhoitopiiri. Viitattu 30.01.2022. https://www.mielenterveystalo.fi/aikuiset/Tietopankki/Hoitomuotoja/Pages/Psykoterapia.aspx.

Psykiatrinen erikoissairaanhoito 2020. 2021. Terveyden ja Hyvinvoinnin laitos. Viitattu 14.03.2022. https://www.slideshare.net/THLfi/psykiatrinen-erikoissairaanhoito-2020

Thorndyke K, 2021. What is CSS Used For?. codeacademy. Viitattu 13.02.2022. https://www.codecademy.com/resources/blog/what-is-css-used-for/

Liitteet

Liite 1. Liitteen otsikko

```
<kysymys>
    <id>01</id>
    <kysymystyyppi>1</kysymystyyppi>
    <kysymysketju>tyhjä</kysymysketju>
    <teksti>Pelkään menettäväni kumppanini, läheisen tai lapseni rakkauden.</teksti>
    <aputekstil>tyhjä</aputekstil>
    <aputeksti2>tyhjä</aputeksti2>
    <vaihtoehto1>1 ei kuvaa minua lainkaan</vaihtoehto1>
    <vaihtoehto2>2 kuvaa minua vain vähän</vaihtoehto2>
    <vaihtoehto3>3 kuvaa minua jonkin verran</vaihtoehto3>
    <vaihtoehto4>4 kuvaa minua melko paljon</vaihtoehto4>
    <vaihtoehto5>5 kuvaa minua erittäin hyvin</vaihtoehto5>
    <vaihtoehto6>6 en osaa sanoa</vaihtoehto6>
    <vaihtoehto7>tyhjä</vaihtoehto7>
    <vaihtoehto8>tyhjä</vaihtoehto8>
    <vaihtoehto9>tyhjä</vaihtoehto9>
    <vaihtoehto10>tyhjä</vaihtoehto10>
    <pistetyyppi1>turvallinen</pistetyyppi1>
    <pisteet1>5</pisteet1>
    <pistetyyppi2>turvallinen</pistetyyppi2>
    <pisteet2>10</pisteet2>
    <pistetyyppi3>turvallinen</pistetyyppi3>
    <pisteet3>5</pisteet3>
    <pistetyyppi4>takertuva</pistetyyppi4>
    <pisteet4>5</pisteet4>
    <pistetyyppi5>takertuva</pistetyyppi5>
    <pisteet5>5</pisteet5>
    <pistetyyppi6>jäsentymätön</pistetyyppi6>
    <pisteet6>5</pisteet6>
    <pistetyyppi7>tyhjä</pistetyyppi7>
    <pisteet7>tyhjä</pisteet7>
    <pistetyyppi8>tyhjä</pistetyyppi8>
    <pisteet8>tyhjä</pisteet8>
    <pistetyyppi9>tyhjä</pistetyyppi9>
    <pisteet9>tyhjä</pisteet9>
    <pistetyyppi10>tyhjä</pistetyyppi10>
    <pisteet10>tyhjä</pisteet10>
    <kommenttil>tyhjä</kommenttil>
    <kommentti2>tyhjä</kommentti2>
    <kommentti3>tyhjä</kommentti3>
    <kommentti4>tyhjä</kommentti4>
    <kommentti5>tyhjä</kommentti5>
    <kommentti6>tyhjä</kommentti6>
    <kommentti7>tyhjä</kommentti7>
    <kommentti8>tyhjä</kommentti8>
    <kommentti9>tyhjä</kommentti9>
    <kommenttil0>tyhjä</kommenttil0>
```

Liite 2. Liitteen otsikko