KARELIA-AMMATTIKORKEAKOULU
Tietojenkäsittelyn koulutusohjelma

Eveliina Hirvonen

NEUROPSYKOLOGIN TEHTÄVÄPANKKI

Opinnäytetyö
Joulukuu 2017

Karelia AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Joulukuu 2017
Tietojenkäsittelyn koulutusohjelma

Karjalankatu 3
80200 JOENSUU
p. (013) 260 600

Tekijä(t)
Eveliina Hirvonen

Nimeke
Neuropsykologin tehtäväpankki

Toimeksiantaja
Neuropsykologi Irmeli Keränen

Tiivistelmä

Tässä toiminnallisessa opinnäytetyössä käsitellään neuropsykologi Irmeli Keräselle luotua verkkosivua, joka on laadittu arkistointityökaluksi. Tavoitteena oli luoda toimeksiantajalle selkeä, toimiva ja helppokäyttöinen tiedostojen tallennuspaikka ja kirjoittaa sen kehitykseen ja suunnitteluun liittyvistä vaiheista.

Opinnäytetyössä keskitytään arkistointisivun ohella myös verkkosivuihin, niiden kehitykseen ja siihen liittyviin menetelmiin. Pääpainona ovat erilaiset verkkosivupohjat sekä verkkosivuihin liittyvät ohjelmointikielet kuten HTML ja CSS.

Toimeksiantaja suunnitteli ja antoi arkistoitaville tiedoille kategoriat, joiden ympärille sivu rakennettiin. Tiedostojen lisäys ja poistaminen ovat verkkosivun tärkeimpiä toiminnallisuuksia. Tässä työssä on esitelty projektin suunnittelun ja luomisen vaiheet sekä pohditaan siihen liittyviä valintoja.

Kieli
suomi

Sivuja 43
Liitteet 0

Asiasanat
arkistointi, verkkosivukehitys, dropzone, html, css

Karelia AMMATTIKORKEAKOULU

THESIS
December 2017
Degree Programme in
Business Information Technology

Karjalankatu 3
80200 JOENSUU
p. (013) 260 600

Author(s)
Eveliina Hirvonen

Title
Neuropsychologists assignment bank

Commissioned by
Neuropsychologist Irmeli Keränen

Abstract

This functional thesis addresses a website made for neuropsychologist Irmeli Keränen which needed to work as an archiving tool for the client. The aim of the thesis was to create a distinct, working and easy to use file saving site and write about the steps of the development.

Besides the archiving website, this thesis also focuses on the methods and development related to website building. The main focus being on the different website templates and the related programming languages like HTML and CSS.

The client planned and provided categories for the archived files, around which the website was built. Adding and deleting files are the key functionalities of the website. This document covers the planning and creating phases of the project and the related deliberations and decisions taken.

Language	Pages 43
Finnish	Appendices 0

Keywords

archiving, website development, dropzone, html, css

Sisältö

1 Johdanto

Tässä dokumentissa käsitellään toiminnallista opinnäytetyötä, jossa luotiin neuropsykologi Irmeli Keräselle mahdollisuus arkistoida työhönsä liittyviä tehtävätiedostoja verkkoon. Tehtävätiedostoilla tarkoitetaan erilaisia oppimistehtäviä, jotka toimeksiantaja on itse luonut. Tehtävät on tehty erilaisille kuntoutuskävijöille ottaen huomioon heidän mahdolliset erilaiset oppimishäiriönsä tai sairautensa. Toimeksiannossa toivottiin verkkosivua, jonne voisi tallentaa kategorioittain luotuja tehtäviä eri oppimishäiriöille ja näin vähentää papereiden ja kansioiden määrää Psykologipalvelu Aapelin fyysisessä toimipisteessä sekä helpottamaan tietojen löytämistä ja tehtävätiedostojen hallinnointia.

Psykologipalvelu Aapeli on toiminimi, jonka toimipaikka on Joensuun keskustassa. Aapeli tarjoaa neuropsykologisia palveluita nuorille ja lapsille, sisältäen mm. neuropsykologiset tutkimukset, kuntoutuksen ja konsultaation. Keränen on Psykologipalvelu Aapelin ainoa työntekijä.

Kuntoutuskävijöille on luotu aiheeseen liittyviä tehtäviä ja varmistetaan, että tehtävä tukee oikeaa kuntoutettavaa oikealla tavalla. Oppimistyylejä ja -häiriöitä on paljon erilaisia, niin hyllyihin on kertynyt tehtäviä ja kansioita, jotka vievät toimistolta paljon tilaa. Toimeksiantaja lähestyi minua toivoen, että voisin luoda hänelle toimivan ratkaisun paperin ja kansioiden vähentämiseksi niin, että hän pääsisi tietoihin helposti ja turvallisesti käsiksi myös kotoaan.

Opinnäytetyön tuloksena toteutettiin arkistointisivu tehtävätiedostojen käyttöön. Tässä työssä käsitellään verkkosivujen luomista yleisellä tasolla sekä Psykologipalvelu Aapelin arkistointisivujen näkökulmasta. Kyseessä oleva verkkosivu on luotu toimeksiantajan käyttöön, ja se on räätälöity vastaamaan hänen tarpeitaan. Opinnäytetyön pohjalta ideaa voidaan kuitenkin myös soveltaa toiselle yritykselle tai henkilölle.

Sivustolla on käytetty hyväksi olemassa olevia työkaluja, jotka esitellään tässä opinnäytetyössä myöhemmin tarkemmin. Sivustolla olevat kategoriat ja tiedostot ovat opinnäytetyön toimeksiantajan luomia.

2 Verkkosivukehitys

Tässä luvussa käydään läpi verkkosivujen kehittämiseen liittyviä tapoja, menetelmiä ja tekniikoita. Tämä sisältää useamman verkkosivujen rakentamiseen käytetyn ohjelmointikielen sekä erilaiset valmiit verkkosivualustat.

2.1 HTML

”HTML on ollut web-sivujen merkkauskielenä alusta alkaen. Alun perin se suunniteltiin ensisijassa tieteellisten dokumenttien esittämiseen, mutta sen yleinen rakenne ja myöhempi kehitys ovat mahdollistaneet sen paljon laajemman käytön.” (Korpela 2014, 26.)

HTML eli Hypertext Markup Language luotiin alun perin vuonna 1990. Aluksi sitä kehitettiin CERNissä (Conseil Européen pour la Recherche Nucléaire), joka tunnetaan paremmin Euroopan hiukkasfysiikan tutkimuskeskuksena. CERNissä työskentelee myös Tim Berners-Lee, joka tunnetaan myös nimellä TimBL. Tim keksi vuonna 1990 tiedostonsiirtojärjestelmän, jonka avulla pystyttäisiin jakamaan tutkimustuloksia ympäri maailmaa. Myöhemmin tämä keksintö siirtyi maailmanlaajuiseen käyttöön, ja se tunnetaan nimellä WWW eli World Wide Web. HTML näki maailmanlaajuisen päivänvalon kuitenkin vasta vuonna 1995, jolloin laadittiin ensimmäinen kielen julkinen määritelmä, IETF:n HTML 2.0. (W3C 1998.)

HTML 3.2 -versiosta lähtien sen määrittelyä hoiti World Wide Web Consortium, joka tunnetaan lyhyemmin W3C:nä. Myöhemmin kuitenkin mukaan on tullut myös WHATWG, joka on Applen, Mozillan sekä Operan työntekijöiden perus-

tama yhteisö, jolla oli alussa eri linjaukset ja halut HTML-kielelle kuin W3C:llä. Myöhemmin vuonna 2006 W3C kuitenkin hylkäsi aiemman linjansa, jolloin Tim Berners-Lee ilmoitti, että W3C ja WHATWG tulisi työskentelemään yhdessä. Ensimmäinen HTML5-luonnos julkistettiin W3C:n toimesta vuonna 2008. Molemmat työryhmät toimivat erilaisilla periaatteilla. WHATWG on pääasiassa kaikille avoin yhteisö, joka toimii editorikeskeisesti. Kaikilla on vapaus sanoa mielipiteensä ja keskustella, mutta lopulliset päätökset tekee lopulta editori. W3C taas toimii yritysten ja laitosten muodostamana organisaationa omien ohjeiden ja sääntöjensä mukaan. (Korpela 2014, 28–30.)

”Joskus sanotaan, että HTML5 valmistuu vasta vuonna 2020 tai jopa myöhemmin” (Korpela 2014, 34). HTML-kielestä puhuttaessa tulee usein esille myös XHTML (Extensible HyperText Markup Language). Se on melkein täysin identtinen HTML-kielelle mutta on paljon tarkempi kuin HTML. ”XHTML-syntaksista ei yleensä ole verkkosivuilla mitään etua, mutta se on sallittu vaihtoehto” (Korpela 2014, 41).

Mitä HTML-kielellä sitten tehdään? HTML on kieli, jolla kuvataan verkkosivujen rakennetta. Sillä voi luoda verkkosivuja/dokumentteja, jotka sisältävät esimerkiksi tekstiä, kuvia, taulukoita ja listoja. Kirjoittaessa HTML-kieltä käytetään siinä erikseen määriteltyjä tageja, kuten esimerkiksi , joka aloittaa kappaleen. Tagi suljetaan halutun kappaleen lopussa kirjoittamalla . HTML-kielessä tagit tulee aina sulkea. Näitä tageja voi kuvitella rakennuspalikoiksi. Rakennuspalikoita käyttäen voidaan luoda persoonallisia ja uniikkeja dokumentteja, jotka näyttävät loppujen lopuksi selaimessa verkkosivulta vaikka tekstieditorissa se näyttäisi vain puhtaalta tekstiltä ilman muotoiluja. (Sarja 2012.)

Helpoin tapa näyttää, kuinka HTML- kieli ja verkkosivujen luonti toimii, on luoda yksinkertainen HTML-tiedosto ja avata se selaimessa. HTML-dokumentti aloitetaan ilmoittamalla käytetty kieli. <!DOCTYPE html> dokumentin alussa kertoo, että dokumentti on kirjoitettu HTML5-kielellä. Tätä tagia ei tarvitse sulkea myöhemmin. Tyypin määrittämisen jälkeen aloitetaan itse dokumentin kirjoittaminen avaamalla <html> tagi. Tämä tagi täytyy sulkea dokumentin lopussa ja se tulee

lisätä HTML-sivuun aina. Tähän mennessä tekstieditorissa sivusto näyttäisi tältä:

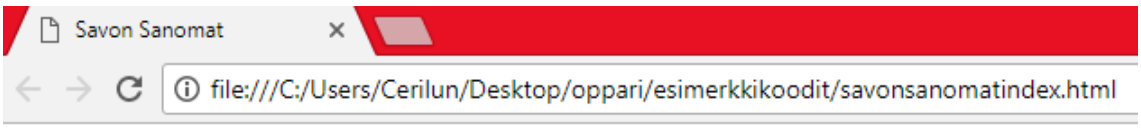
<!DOCTYPE html>
<html>
</html>

Tämä on yleinen HTML-dokumentin aloitus. Se ei vielä tällä hetkellä tee tai näytä miltään, sillä sivulla ei ole vielä ollenkaan sisältöä mutta tiedoston avatessasi selain tunnistaa sen nyt HTML-pohjaiseksi verkkosivuksi. HTML-tagien sisään kirjoitetaan kaikki muu, mitä sivulle halutaan lisätä. Yleensä melko alussa dokumentissa on tagi <head> </head>. Tämän elementin sisään kirjoitetaan dokumentin metatietoja, esimerkiksi sivun otsikko. Sivun otsikointi tapahtuu käyttämällä <title> </title>-tagia. Kuvitellaan vaikka, että olisimme luomassa Savon Sanomien verkkosivua. Tällöin kirjoitettu koodi näyttäisi kuvan 1 mukaiselta.

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <title>Savon Sanomat</title>
5      </head>
6  </html>
```

Kuva 1. <head>-tagien sisään lisätty otsikko.

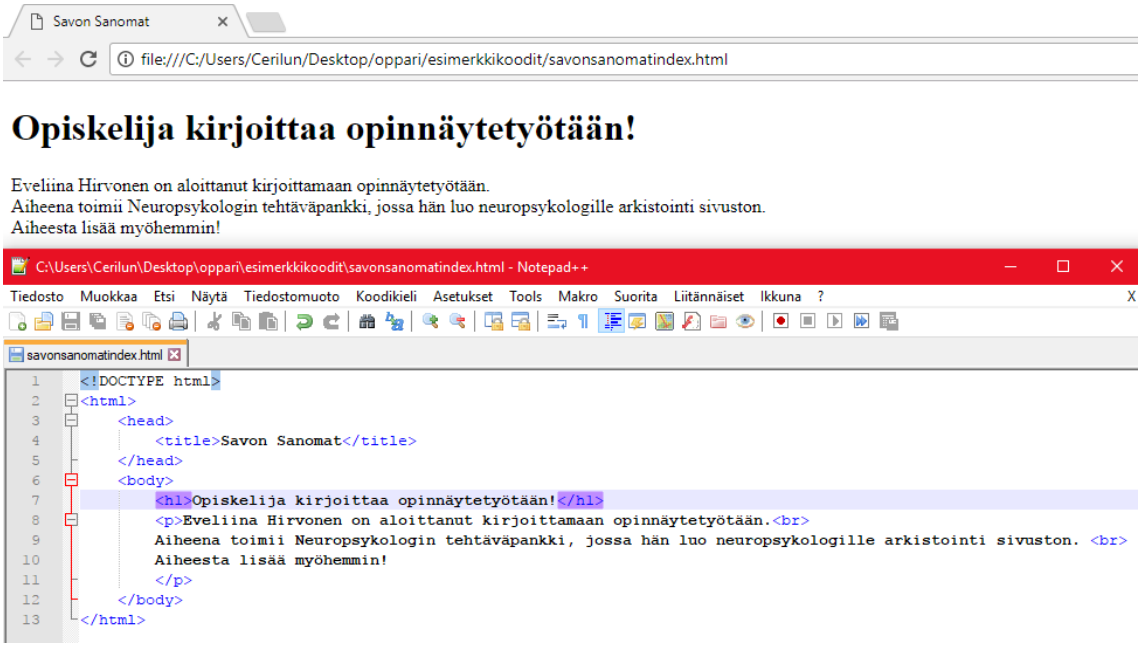
Selaimessa avatessa dokumentti näyttäisi kuvan 2 mukaiselta.



Kuva 2. HTML-dokumentti avattuna selaimessa.

Huomataan, että sivu on saanut otsikon. Osoite rivillä näkyy tallennetun HTML-tiedoston polku, sillä sivua ei ole siirretty verkkoon. Sen voi kuitenkin ajaa selaimessa, jotta voidaan nähdä, miltä sivu oikeastaan näyttää. <Head>-tagien sisään voidaan laittaa paljon muitakin tietoja ja tageja. Head tagien sisään laitetaan myös linkki käytettyyn tyylitiedostoon, eli CSS-tiedostoon. CSS-kieltä käydään läpi tässä dokumentissa hieman myöhemmin.

Suljetun <head>-tagin jälkeen voidaan aloittaa itse sisällön lisääminen. Sisältö kirjoitetaan <body> </body>-tagien sisälle. Tämä määrittää sivun ”vartalon”, joka sisältää kaiken HTML-dokumentin sisällön, esimerkiksi kuvat ja tekstin. Olen kirjoittanut pienen uutismaisen tekstin, jonka olen lisännyt esimerkkinä käyttämääni HTML-dokumenttiin. Kun HTML:n vartalossa on vihdoin sisältöä, näyttää se selaimessa ajettuna kuvan 3 mukaiselta.



Kuva 3. Verkkosivun ulkonäkö, kun vartaloon on lisätty tietoa.

Näin on luotu yksinkertainen HTML-dokumentti, joka avautuu selaimessa verkkosivun lailla. Vartalon sisään olen kirjoittanut lyhyen teksti kappaleen. Ensimmäisen tason otsikko määritellään dokumentissa tagilla <h1> </h1>. Tekstin olen kirjoittanut   -tagien sisään. Kappale merkitään HTML-koodiin -tagilla. Lauseiden lopuissa olevat
-tagit ovat pakotettuja rivinvaihtoja.

Sivua voi jatkossa kehittää monella eri tapaa käyttäen HTML-kieltä. Sen avulla voidaan lisätä ja muokata sivua haluamakseen monella eri tapaa. Verkkosivua tehtäessä kuitenkin usein halutaan tehdä sivusta näyttävämpi, persoonallisempi ja lisätä siihen toiminnallisuuksia. Tällöin tarvitaan apuun muutakin kuin pelkästään HTML-koodia.

2.2 CSS

HTML-kielen perusteiden oppimisen jälkeen kannattaa ensimmäisenä opetella kirjoittamaan CSS-kieltä. ”CSS-tyyliohje eli lyhyesti tyyliohje sisältää selaimelle annettavia ohjeita HTML-sivun tai XML-dokumentin ulkoasusta” (Korpela 2014, 71).

CSS on lyhenne sanoista Cascading Style Sheets, ja sitä kehittää W3C samalla tavalla kuin HTML-kieltä. CSS 1 julkaistiin alun perin vuonna 1996. Sitä seurasi uudelleenjulkaisu vuonna 1999. CSS 2 rakennettiin ensimmäisen tason päälle ja se näki päivänvalon vuonna 1998. CSS 3-versiota kehitetään edelleen. CSS-kielen alkuperäinen pohjaidea on sama kuin sen käyttö nykyäänkin: sen käyttö säästää verkkosivujen tekijältä paljon aikaa, sillä se voi määrittää useamman sivun ulkoasun yhdestä dokumentista. (W3C 2016.)

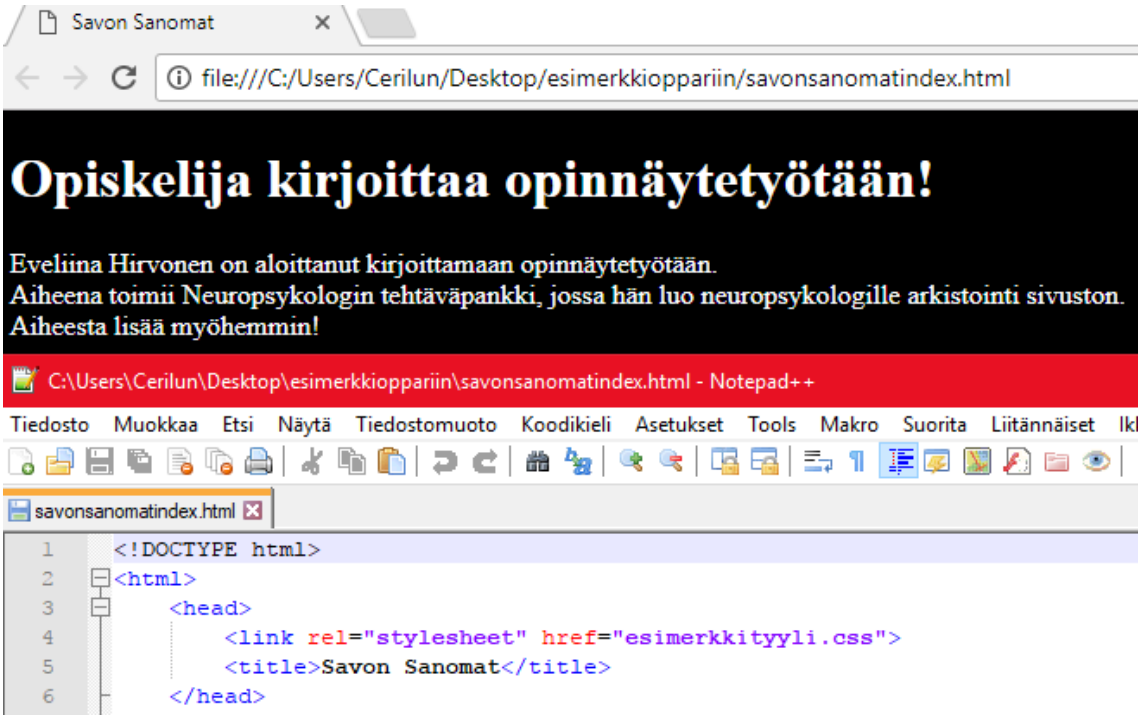
Kuvitellaan esimerkiksi, että on luotu verkkosivu, jossa on useampi sivu ja kaikki sivut käyttävät samaa tyylitiedostoa. Verkkosivun tekijä haluaisi vaihtaa tason 2 otsikot punaisiksi ja vaihtaa fontin Arialista Helveticaksi. Ilman CSS-dokumenttia joutuisi verkkosivu kehittäjä käymään jokaisen sivun erikseen läpi, etsimään tiedostoista ne kohdat, joissa käytetään toisen tason otsikkoa ja muokkaamaan niiden fonttia sieltä. Kun kaikki sivut käyttävät samaa tyylitiedostoa, voi kehittäjä yksinkertaisesti avata tyylitiedoston ja vaihtaa toisen tason otsikon määrittelyn sieltä käsin, jolloin kaikkien sivujen otsikot vaihtuvat yhdellä pienellä muutoksella. (Hoffman 2017.)

CSS-kieltä voidaan käyttää kolmella eri tavalla: Inline, jolloin tyyli attribuutit kirjoitetaan HTML-elementteihin, Internal, jolloin <style>-elementit kirjoitetaan <head>-tagien sisään HTML-koodissa, tai External, jolloin käytetään ulkoista CSS-tiedostoa. Ulkoista CSS-tiedostoa käyttäessä CSS-koodi kirjoitetaan erilliseen tiedostoon, joka linkitetään sitten HTML-koodiin, kuten HTML-osiossa aiemmin mainitsin. CSS on selittävä ohjelmointikieli; kun sitä kirjoitetaan, se ei kerro selaimelle, kuinka sivu näytetään, vaan se kertoo säännöt HTML-dokumentille ja selain seuraa sitä tehden lopun työn. (W3Schools 2017.)

Aiemmin luotuun HTML-dokumenttiin voidaan lisätä CSS-tiedosto tyyliohjeeksi todella helposti. Loin uuden dokumentin, jonka tallensin CSS-muotoon. Halusin pitää esimerkin yksinkertaisena, joten päätin muuttaa sivun taustavärin ja fontin. Määritin tätä varten CSS-tiedostossa bodylle taustavärin sekä värin. Käytännössä CSS-dokumentissa se näyttäisi tältä:

```
body {
background-color:black;
color:white;
}
```

Tallensin tiedoston samaan kansioon kuin sivun HTML-tiedosto, jotta sen linkittäminen olisi helpompaa. HTML-dokumenttiin tulee <head> </head>-tagien sisään kirjoittaa <link rel=”stylesheet” href=”tiedoston_nimi_tähän.css”>. Stylesheet kertoo, että kyseessä on tyylitiedosto ja href viittaa itse linkkiin. Tiedoston ollessa samassa kansiossa on polku tiedostolle lyhyt, jos tiedosto olisi toisessa kansiossa, kirjoitettaisiin sen polku ennen tiedostonimeä normaalisti. HTML-tiedoston voi nyt avata ja voidaan huomata, että taustan ja fontin värit ovat kääntyneet (kuva 4). CSS-koodissa voidaan käyttää värin nimen sijaan myös värin heksadesimaalia.



Kuva 4. HTML-sivu CSS-tiedoston kanssa.

2.3 JavaScript

JavaScript on yksi kolmesta tärkeästä ohjelmointikielestä, joka verkkosivukehittäjän tulee osata. Verkkosivuja tehtäessä HTML-kielellä määritellään sisältöä ja CSS-kielellä kuvataan ulkoasua. JavaScriptillä on kielistä se, jolla luodaan sivulle toiminnallisuutta. ”JavaScript on HTML-kielen yhteydessä pääasiassa selainscriptien tekemiseen käytetty kieli. Selainskripti eli lyhyesti skripti tarkoittaa ohjelmakoodia, jonka selain suorittaa HTML-sivun näyttämisen yhteydessä” (Korpela 2014, 55).

JavaScript kielen loi alun perin Brendan Eich vuonna 1995, ja siitä tuli ECMA standardi vuonna 1997. Virallinen nimi JavaScriptille on ECMAScript. Viimeisin versio kielestä on ECMAScript 7, joka julkaistiin vuonna 2006. (W3Schools 2017.) Alun perin kieli luotiin, jotta verkkosivut voitaisiin ”tuoda eloon”.

Selaimessa JavaScript voi esimerkiksi reagoida käyttäjän toimintoihin, kysyä käyttäjältä kysymyksiä, näyttää viestejä ja lähettää verkon yli pyyntöjä toiselle palvelimelle. JavaScriptiin ja sen kirjoittamisen helpottamiseen on olemassa

paljon työkaluja, kuten esimerkiksi CoffeeScript. CoffeeScriptillä kirjoitettu koodi on huomattavasti lyhyempi ja yksinkertaisempi kuin normaali ohjelmointikielellä kirjoitettu koodi. Koodi, joka muuten saattaisi näyttää tältä:

```
var, cube, square;
square = function(x){
return x*x;
};
cube = function(*){
return square(x)*x;
}
```

Voidaan kirjoittaa CoffeeScriptillä näin:

```
square = (x) -> x*x
cube = (x) -> square(x)*x
```

CoffeeScriptin komentorivi versio tarvitsee kuitenkin Node.js:n. Node.js on avoimen lähdekoodin sovelluskehys, joka mahdollistaa JavaScriptin suorittamisen palvelimen puolella. Node.js poistaa paljon odotusaikaa, sillä se mahdollistaa useiden asioiden käsittelyn samanaikaisesti kun taas esimerkiksi PHP odottaa ensin tiedoston avaamisen, sitten lukemisen ja lopuksi tiedon lähettämisen ja kykenee vasta sen jälkeen aloittamaan seuraavan pyynnön. (W3Schools 2017.)

JavaScript-kieltä voidaan kirjoittaa HTML-koodin sisään suoraan. JavaScript ohjelma kutsutaan dokumentissa <script>-tagillä. Toinen mahdollisuus on laittaa JavaScript-koodi omaan tiedostoonsa, samalla tavalla kuin CSS-koodi. Tällöin JavaScript-tiedosto linkitetään HTML-dokumenttiin, kuten CSS-tiedostokin. JavaScript-tiedosto linkitetään näin: <script src=”/aseta/tiedostopolku/tähän.js”></script>. Tiedoston voi myös linkittää verkosta, jolloin käytetään tiedostopolun sijasta URL-osoitetta. Hyvänä sääntönä on se, että jos JavaScript-koodia on paljon, tulee se lisätä omaan tiedostoonsa selkeyden vuoksi.

JavaScript-kielellä kirjoitettu ”Hello World” näyttäisi HTML-koodin sisällä tältä:

```
<!DOCTYPE HTML>
<html>
<head>
<title>Otsikko tähän</title>
</head>
<body>
<script>
alert(‘Hello World’);
</script>
</body>
</html>
```

Yllä olevan koodin avulla aukeaisi HTML-tiedostosta verkkosivu, joka ajettaessa selaimessa aukeaisi ja antaisi viesti ikkunan, jossa sanotaan “Hello World”. JavaScript tarvitsee siis periaatteessa vain <script>-tagin toimiakseen. JavaScript-kielen kirjoittamiseen löytyy Internetistä paljon erilaisia tutoriaaleja sekä oppaita. Kielen ymmärtäminen ja kirjoittaminen on HTML-kielen ja CSS-kielen perustason osaamisen jälkeen helpompaa.

2.4 PHP

PHP on akronyymi sanoista ”Hypertext Preprocessor”. PHP on avoimen lähdekoodin skriptaus kieli, jonka opetteleminen kannattaa aloittaa kun HTML-, CSS- ja JavaScript-kielten perusteet ovat hallinnassa. On arvioitu, että PHP on käytössä kymmenillä tai jopa sadoilla miljoonilla domaineilla ympäri maailmaa. (The PHP Group 2017.) Se on käytössä monella suuren käyttäjämäärän sivustolla, kuten esimerkiksi Facebookilla sekä WordPressillä.

Ensimmäinen PHP-kielen muoto luotiin vuonna 1994 Rasmus Lerdorfin toimesta. Se oli silloin yksinkertainen paketti CGI binäärejä kirjoitettuna C-ohjelmointikielellä. Rasmus käytti sitä seuratakseen verkko ansioluettelonsa kä-

vijämääriä. Silloin hän käytti PHP:stä nimeä ”Personal Home Page Tools”. (The PHP Group 2017.)

Rasmusta pyydettiin jatkokehittämään luomiaan PHP Toolseja, joten hän lisäsi siihen paljon erilaisia toiminnallisuuksia. Vuonna 1995 Rasmus jakoi PHP Toolsin lähdekoodin käyttöön kaikille. Lähdekoodin jakaminen mahdollisti sen, että kehittäjät pystyivät käyttämään PHP Toolseja heidän haluamallaan tavalla. Sen käyttäjät pystyivät myös tarjoamaan korjauksia ja parannuksia koodille. PHP5 on viimeisin PHP:n versio, joka on julkaistu vuonna 2004. (The PHP Group 2017.)

PHP-tiedosto, vaikka se sisältäisi myös HTML-kieltä, on tiedostomuodoltaan aina aseta_tiedoston_nimi.php. PHP-tiedosto voi sisältää HTML-, CSS-, JavaScript- ja PHP-kieltä.

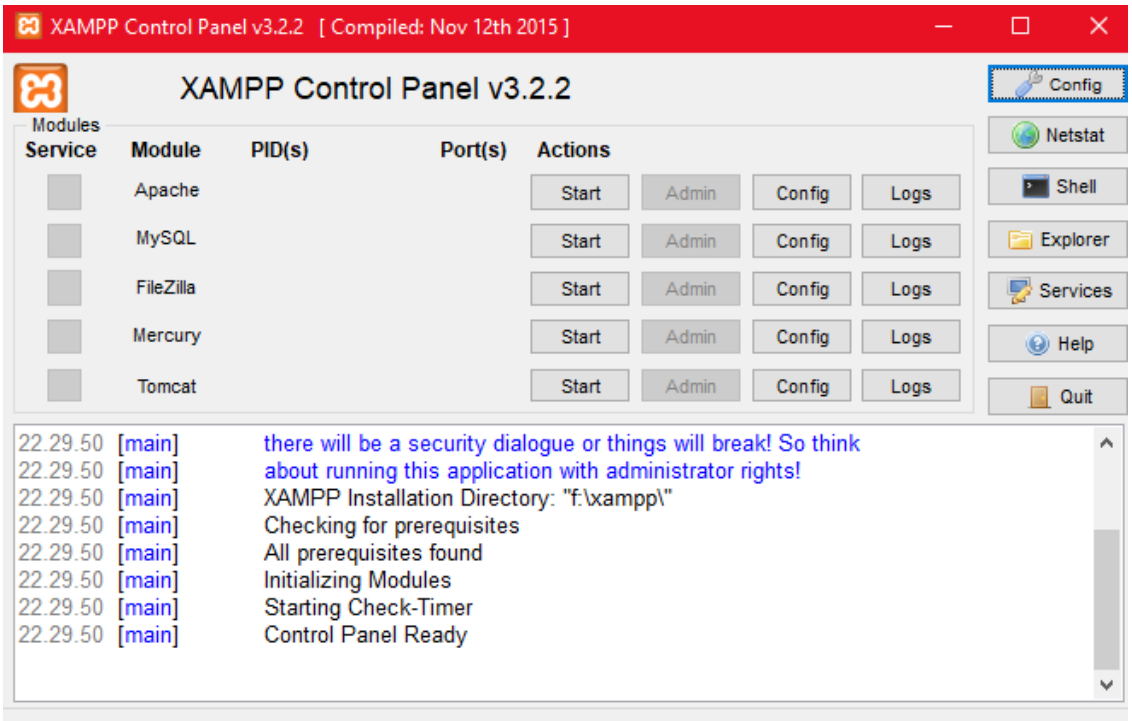
Lyhyenä esimerkkinä tästä voimme katsoa, miltä ”Hello World” näyttäisi HTML-koodin sisällä PHP-kieltä apuna käyttäen:

```
<!DOCTYPE html>
<html>
<head>
<title>Otsikko tähän</title>
</head>
<body>
<?php
echo ”Hello World”;
?>
</body>
</html>
```

Tällä koodilla aukeaisi verkkosivu, jolla lukisi pelkästään Hello World. PHP-ohjelmaa kutsutaan koodin sisällä käyttämällä <?php ja ?>-tageja. Toinen aloittaa ja toinen sulkee PHP-koodin.

PHP:n käyttämistä varten tarvitaan palvelin, joka pystyy suorittamaan PHP-koodia. Usealla edullisella webhotellilla on PHP-kielen tuki. Ennen kuin sivu on valmis laitettavaksi verkkoon, halutaan sitä kuitenkin usein tarkastella ensin omalla koneella. Tähän on olemassa monia ratkaisuja, joista yksi on XAMPP-ohjelman asentaminen. Sen voi ladata ilmaiseksi verkosta.

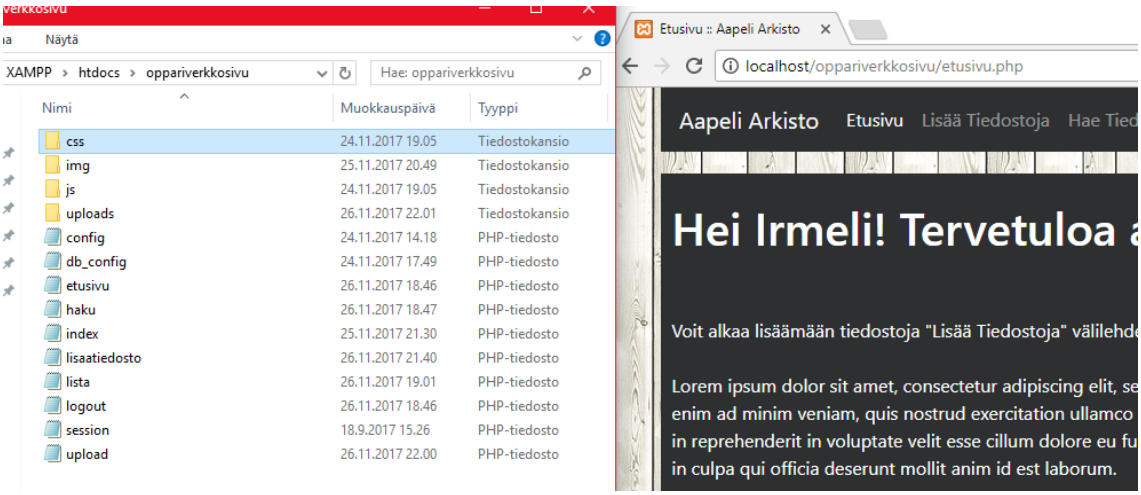
XAMPP on avoimen lähdekoodin verkkopalvelin ratkaisu. Sen käyttö mahdollistaa sen, että verkkosivuja voidaan katsella aivan kuin se olisi jo verkossa. Ohjelman avulla käyttäjä voi tehdä omasta koneestaan palvelimen ohjelman sisältämien ominaisuuksien avulla. XAMPP pitää sisällään Apachen sekä MySQL:n, jotka ovat ominaisuuksista tärkeimmät. MySQL on relaatiotietokantaohjelmisto, joka on yleisessä käytössä verkkosivuilla. Tietokantaan pystytään tallentamaan tietosisältöä, jota sivu käyttää. Tämä tieto voi olla esimerkiksi käyttäjätietoja. Apache on verkkopalvelinohjelma. Se mahdollistaa sen, että voit käyttää konettasi palvelimena ja käyttää sitä esimerkiksi verkkosivun ylläpitämiseen. Apache tukee PHP:tä, joten sen avulla näkyvät myös PHP-kieltä ja ominaisuuksia käyttävät tiedostot. (Apache Friends, 2017.)



Kuva 5. XAMPP Control Panel.

XAMPP voi kuulostaa monimutkaiselta käyttää mutta sen ohjauspaneelia on helppo käyttää (kuva 5). Apachen sekä MySQL:n käynnistäminen vaatii vain Start-napin painamisen.

Verkkosivu tulee siirtää omassa kansiossaan XAMPP-kansion sisällä olevaan htdocs-kansioon, jotta sitä voidaan tarkastella.



Kuva 6. Verkkosivun esikatselu XAMPP-ohjelman avulla.

Kuvassa 6 on esimerkki siitä, kuinka XAMPP-ohjelman avulla voidaan tarkastella verkkosivua. XAMPP-kansion sisällä on alikansio, nimeltään oppariverkkosivu, jonka sisälle on tallennettu kaikki verkkosivun sivut. CSS- ja JavaScript-koodit ovat omissa kansioissaan, sekä verkkosivulla olevat kuvatiedostot.

Sivua voidaan tarkastella kirjoittamalla selaimeen localhost/kansionnimi/sivunnimi.php. Esimerkki tapauksessa etusivun saa auki kuvassa näkyvällä tavalla. Haku.php sivulle pääsisi suoraan selaimessa osoitteella localhost/oppariverkkosivu/haku.php. Täytyy kuitenkin muistaa, että tällä tavalla kukaan ulkopuolinen ei pääse sivustoon käsiksi. Se on kuitenkin hyvä työkalu kirjoitusvaiheessa.

Ohjelman avulla käyttäjä voi myös luoda tietokantoja, joita käyttää verkkosivullaan. Se onnistuu menemällä selaimella osoitteeseen localhost/phpmyadmin. Sivun voi myös avata painamalla Admin nappia XAMPP-ohjelman kontrolli pa-

neelissa. PhpMyAdmin-sivun käyttäminen voi vaatia hieman opettelua mutta siihen on verkossa paljon erilaisia ohjeistuksia ja tutoriaaleja.

2.5 Valmiit alustat

Valmiita verkkosivun pohjia on olemassa paljon ja ne ovat kaikki erilaisia. On olemassa myös täysin valmiita sivuja, jonne kirjoitetaan itse vain sisältöä. Nämä sivut ovat yleensä todella yksinkertaisia, eivätkä ne ole laajasti muokattavissa tai hallittavissa.

Julkaisualustoja, joissa on paljon muokattavuutta ja hallittavuutta, on käytössä monilla suurilla yrityksillä. Näiden alustojen virallisempi nimitys on CMS, joka on lyhenne sanoista Content Management System. CMS tarkoittaa sisällönhallinta-järjestelmää, jonka avulla verkkosivujen tekeminen onnistuu ilman varsinaista ohjelmointitaitoa. Erilaisista CMS sivuista olen valinnut lähempään tarkasteluun WordPress-alustan. Valinta on tehty sen perusteella, että sen käyttöönotto ei vaadi käyttäjältä aikaa. WordPress toimii suoraan selaimesta eikä sitä tarvitse asentaa. Muut suositut alustat, kuten Drupal ja Joomla!, tarvitsevat käyttöönsä MySQL-tietokannan, verkkopalvelimen sekä tuen PHP-kielelle. WordPress on alustoista aloittelijaystävällisin vaihtoehto. Tunnettuja WordPress-sivuston käyttäjiä on monia, esimerkiksi Angry Birds pelin verkkosivut toimivat WordPress-alustalla. (WordPress, 2017.)

WordPress-sivustolla on tällä hetkellä käytössä neljä erilaista tilausmallia. Ensimmäinen on ”free” malli, joka on nimensä mukaan ilmainen. Seuraava vaihtoehto on ”personal”, jonka hinta on neljä euroa kuukaudessa. Kolmas vaihtoehto on ”premium”, joka maksaa kahdeksan euroa kuukaudessa. Neljäs ja kallein vaihtoehto on ”business”, jonka hinnaksi tulee 24,92 euroa kuukautta kohden. Kaikki vaihtoehdot laskutetaan kerran vuodessa, ei kuukausittain. (WordPress, 2017.)

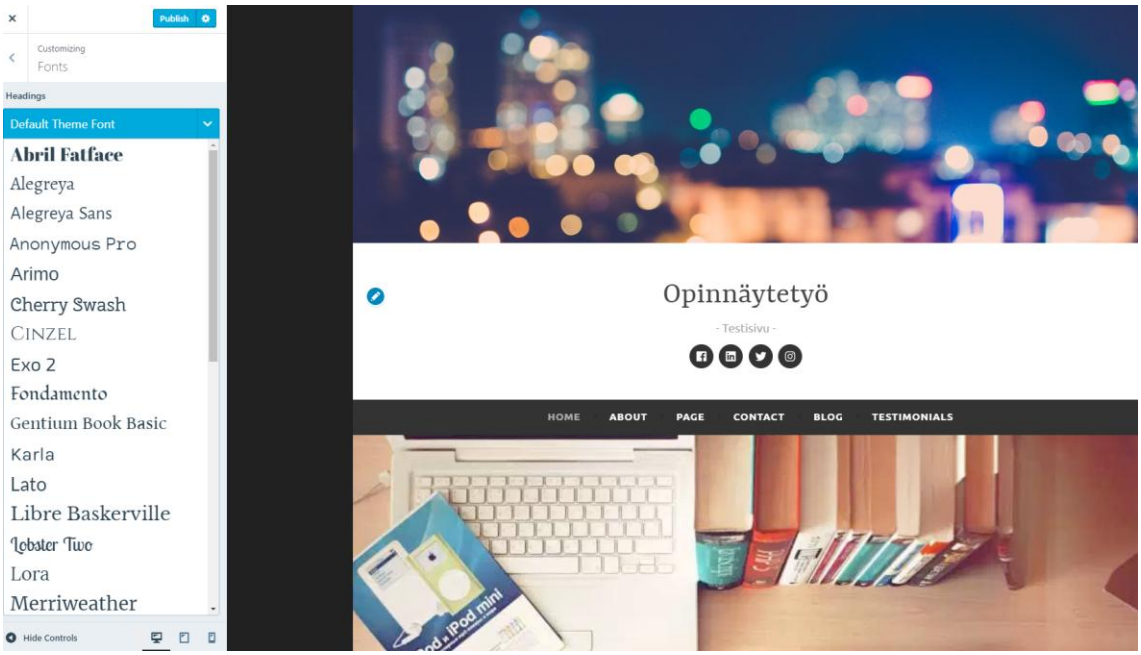
Kaikilla eri malleilla on tietyt ominaisuudet, joita tulee lisää hinnan noustessa. Halvimmassa vaihtoehdossa, eli ilmaisessa, on tallennustilaa 3GB kun kalleimmassa vaihtoehdossa tallennustila on rajaton. Teemoja ja muokattavuutta on

maksullisissa myös enemmän verrattuna ilmaiseen vaihtoehtoon. Kalleimpaan versioon saa ominaisuudeksi myös Google Analytics integroinnin sekä WordPress.com brändäyksen poiston. Kaksi kalleinta vaihtoehtoa saavat myös mahdollisuuden tienata rahaa verkkosivullaan. Niihin voi asettaa mainoksia, joista kertyy mainostuloja sen perusteella, kuinka monta ihmistä mainoksen näkee. (WordPress, 2017.)

WordPress-alustaa voi käyttää selaimen lisäksi myös suoraan tietokoneelle asennettuna, joka vaatii jonkin verran tietotaitoa ja osaamista tai sen voi ladata mobiililaitteelle. Sivun mobiilisovellus on tällä hetkellä saatavilla laitteisiin, joiden käyttöjärjestelmä on iOS tai Android. WordPressin sivuilla voi tarkastella muita sivustoja, jotka on luotu WordPress-alustaa käyttäen. Listalla on useita suuria tunnistettavia nimiä, kuten esimerkiksi New York Times Company sekä Angry Birds. (WordPress, 2017.)

Oman sivun saa WordPress palveluun luotua nopeasti. Rekisteröitymisen jälkeen valitaan sivun tyyli ja domain. Valintojen tekemisen jälkeen sivu on valmis käytettäväksi ja muokattavaksi. Ilmaisessa mallissa WordPress tarjoaa tällä hetkellä 117 teemaa, joista voi valita mieluisensa. Teeman asentamisesta voidaan siirtyä sen ominaisuuksien muokkaamiseen.

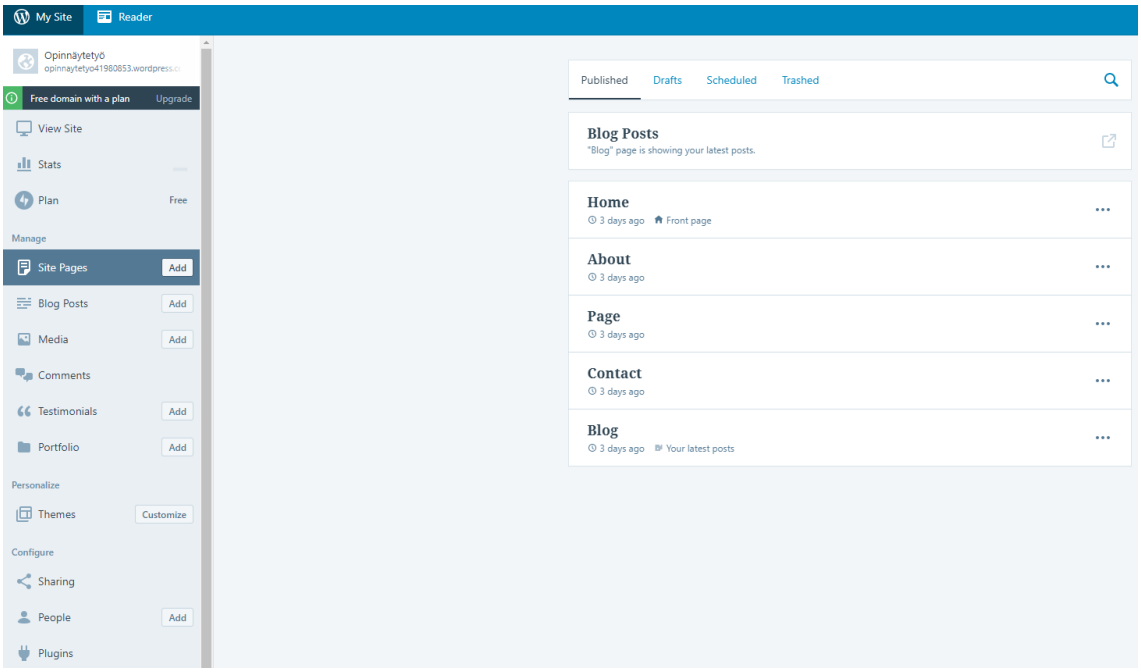
Sivun taustavärejä ja esimerkiksi otsikkokuvaa voi vaihtaa mielensä mukaan. Aivan kaikkea ulkoasussa ei voi muokata. Osa sisällöstä voidaan muokata valitsemalla valmiista vaihtoehdoista, esimerkiksi valittavien fonttien määrä on rajallinen (kuva 7). Sivun sisältö palikoita ei voi siirtää paikasta toiseen vapaasti, joten se poistaa muokattavuutta hieman.



Kuva 7. WordPress-ulkoasun muokkaus.

Ilmaisen version domainin eli verkkotunnuksen saa valita osittain itse. Esimerkkitapauksessa valitsin verkkotunnuksekseni ”opinnäytetyö”, jonka perään WordPress lisäsi kahdeksan satunnaista numeroa. Tämä poistaa ilmaisen version viehättävyyttä paljon. Maksullisessa versiossa (premium ja business) tätä ongelmaa ei ole. Käyttäjä saa valita domaininsa kokonaan itse.

Omaa WordPress-sivua voi muokata ja tarkastella sille suunnatun kojelaudan eli dashboardin kautta. Dashboardin eri tiloja voi avata sivun vasemmasta reunasta. Tätä kautta voidaan lisätä esimerkiksi uusia sivuja omaan WordPress-alustaan (kuva 8).



Kuva 8. WordPress-sivun "dashboard".

Sivun muu hallinnointi onnistuu myös tätä kautta ”Manage”-otsikon alta. Täältä sivulle voidaan lisätä esimerkiksi kuvia, luoda uusia päivityksiä tai lukea sivustolle tulleita kommentteja. Ylempänä listassa ovat valinnat oman sivun tarkasteluun sekä statistiikalle. Statistiikan alta voidaan nähdä erilaista dataa, jota sivusta on muodostunut. Statistiikka seuraa muun muassa kävijämääriä ja kommentteja päivittäin, viikoittain, kuukausittain ja vuosittain. Voidaan myös nähdä, mistä maista kävijöitä on tullut tai kuinka monta klikkausta sivulla on tehty.

WordPress-sivustolle voidaan myös ottaa käyttöön erilaisia plugineita eli liitännäisiä. Yksi käytetyimmistä plugineista on WooCommerce. WooCommercen avulla verkkosivustaan voi tehdä verkkokaupan. Oman kuvauksensa mukaan 28 % kaikista verkkokaupoista pyörii WooCommercella. (WordPress, 2017.) Plugineiden asentaminen vaatii kuitenkin business-tilausmallin valitsemisen.

3 Verkkosivun tavoitteet

Opinnäytetyön ja arkistosivun tarkoitus on tukea toimeksiantajan työntekoa ja toimia arkea helpottavana työkaluna. Tavoitteena oli luoda käyttöönotettava verkkosivu, jonne voidaan tallettaa Irmelin luomia oppimistehtäviä kategorioittain, jotta ne on helppo löytää ja niitä on helppo hallinnoida. Niihin täytyy päästä käsiksi niin kotoa kuin työpaikaltakin. Arkistosta tulee voida myös poistaa tehtävätiedostoja, mikäli tarve vaatii. Kategoriat, joiden alle tiedostoja lisätään, ovat Irmeli Keräsen itse määrittämiä. Verkkosivua tehtäessä luodaan valmiiksi kategoriat ja niiden alle lisätään esimerkiksi muutamia tehtävätiedostoja. Myöhemmin toimeksiantaja voi siirtää ja poistaa tiedostoja itsenäisesti. Kaikki tällä hetkellä paperimuodossa olevat tiedostot Irmeli Keränen skannaa itse ja siirtää ne sivustolle myöhemmässä vaiheessa.

Sivuston tulee olla selkeä ja helppo käyttää, joten se pidetään mahdollisimman yksinkertaisena. Verkkosivun koodin tulee myös olla selkeää ja kommentoitua siinä tapauksessa, että sivustoa tulee myöhemmin muokata tai päivittää. Sivuston tulee olla suojattu kirjautumisella, jotta väärinkäyttöä voidaan välttää. Irmeli Keräselle luodaan verkkosivulle oma käyttäjätunnus sekä hashattu, eli tiiviste suojattu salasana. Kustannukset pyritään pitämään mahdollisimman alhaisina vähäisen käyttäjämäärän vuoksi. Verkkosivujen valmistuttua toimeksiantajaa opastetaan käyttämään arkistointisivua.

4 Arkiston rakentaminen

Tässä luvussa käydään läpi itse arkistosivun rakentamista, mistä se lähti liikkeelle ja kuinka projekti eteni.

4.1 Toteutuksen määrittely

Ennen verkkosivun rakentamisen aloittamista kävimme toimeksiantajan kanssa keskusteluja siitä, kuinka verkkosivu otettaisiin lopulta käyttöön, millainen sivuston pitäisi olla ja kuinka tieto kategorisoitaisiin parhaiten.

Toimeksiantajan kanssa sovittiin, että hän siirtäisi muutamia eri kategorian tehtäviä ulkoiselle kovalevylle, jotta voisin testata sivustolle tiedostojen lisäämistä ja kategorisointia. Näin tiedostoja ei tarvitse lähettää liitetiedostoina sähköpostin kautta, mikä helpottaa ja nopeuttaa työn alkamista eikä lisää toimeksiantajan työtä. Sovimme myös, että toimeksiantaja lisäisi kovalevylle tiedoston, joka sisältäisi kaikki sivuille halutut kategoriat. Näin varmistettaisiin se, että kategoriat on räätälöity ja nimetty toimeksiantajan haluamalla tavalla.

Ennen varsinaista toteutuksen aloittamista suunnittelin ja hahmottelin valmiin verkkosivun paperille, jotta voisin suunnitella sivujen toimintaa ja ennalta miettiä, mitä kaikkea verkkosivuilla tulisi olla, kuinka se sinne lisättäisiin ja kuinka se onnistuisi parhaiten.

Ulkonäöllisesti halusin, että verkkosivu näyttäisi miellyttävältä ja värien käyttö olisi hillittyä. Kuvioita tai räikeitä värejä ei tulisi käyttää, sillä niiden käyttö on yleisesti ottaen verkkosivuilla häiritsevää eikä se näytä hyvältä.

Verkkosivujen suunnittelu vaihe oli helppo, sillä vaatimuksia ei ollut paljoa eikä sivusto tulisi muiden kuin Irmeli Keräsen käyttöön, eikä hän halunnut sivustolle paljoa toiminnallisuutta. Vähäisen käyttäjä määrän takia sivua ei tarvitse räätälöidä usean henkilön käyttöä varten. Pyrin siihen, että sivu olisi mahdollisimman yksinkertainen, selkeä ja helppo käyttää, sillä Irmeli Keräsellä ei ole paljoa tietoteknistä osaamista, ja mitä vähemmän tilaa sivu vaatisi, sitä pienemmät tulisivat webhotellin kustannukset olemaan, kun sivu lopulta siirrettäisiin verkkoon.

Henkilökohtaisesti valmistauduin toteutukseen opiskelemalla PHP- ja JavaScript-kielten käyttöä verkkosivu projekteissa. Olen aiemmin luonut verkkosivuja

pelkästään HTML- ja CSS-kieliä käyttäen, joten minulla ei ollut ennen tätä projektia osaamista muista verkkosivu kehitykseen liittyvistä ohjelmointikielistä. Palautin mieleeni myös aiemmin käyttämäni kielten perusteet ja kävin läpi viimeisimmät päivitykset esimerkiksi Bootstrap komponentti kirjastoon, jota käytin verkkosivua tehdessä paljon.

Suunnittelin myös, kuinka tiedostojen lisäämisen verkkosivulle voisi parhaiten toteuttaa. Sain ystävältäni ehdotuksen Dropzone.js-nimisen avoimen lähdekoodin kirjaston käytöstä. Dropzone.js:n voi vapaasti ladata sille tehdyltä verkkosivulta ilmaiseksi ilman minkäänlaisia käyttörajoituksia. Dropzone sisältää valmiit JavaScript- ja CSS-tiedostot, joiden avulla verkkosivulle voidaan luoda drag’n’drop periaatteella toimiva tiedostojen lataus toiminnallisuus.

4.2 Suunnittelu ja toteutus

Arkistosivun luominen aloitettiin luomalla ensin tyhjät tiedostot jokaiselle sivulle, mikä verkkosivun sisälle haluttiin lisätä. Aluksi sivuja oli neljä. Ensimmäisenä ”Etusivu”, jonne käyttäjä ohjattaisiin kirjautumisen jälkeen. Toisena ”Lisää Tiedostoja”, jonka alle luotaisiin toiminnallisuus tiedostojen lisäämiselle. Kolmantena ”Hae Tiedostoja”, jonne tulisi toiminnallisuus tiedostojen hakemiselle. Neljäntenä sivuna luotiin ”Selaa Tiedostoja”, jossa tiedostoja pystyisi selaamaan listassa vapaasti.

Sivuille lisättiin perinteinen HTML-kielen pohja ensimmäisenä, sisältäen <!DOCTYPE html>-, <html>-, <head>-, ja <body>-tagit. Näiden ympärille sivuja on helpompi alkaa luomaan. Tiedostot olivat vielä HTML-muodossa tässä vaiheessa, joten pystyin tekemään työn alun ilman XAMPP-ohjelman käyttöä. Kieli on myös minulle itselleni tutumpi kuin PHP-kieli, joten sillä oli helpompi alkaa ensin hahmottamaan sivua.

Loin tässä välissä myös verkkosivun oman CSS-tiedoston, jonka linkitin sivuille. CSS-tiedostoon en tehnyt alussa suuria muutoksia mutta lisäsin sinne body lohkon valmiiksi, sillä siihen tultaisiin ainakin lisäämään määrittelyjä ennemmin

tai myöhemmin. Oman CSS-tiedoston linkittämisen lisäksi lisäsin dokumentteihin linkin myös Bootstrapin CSS-tiedostoon. Muita lisättyjä oli muun muassa Bootstrapin JavaScript-koodit sekä meta-tagin tietoja, esimerkiksi utf-8-määrityksen sekä viewportin skaalausta varten.

Ensimmäisenä halusin toimeksiantajan toimittamat kategoriat itselleni sivulle talteen, jotta niitä ei tarvitsisi etsiä tiedostona aina, kun niitä toteutuksessa tarvittaisiin. Loin selaa tiedostoja välilehdelle Bootstrapillä accordion listan, johon sisällytin kaikki toimeksiantajan kategoriat. Yläkategorioita oli yhteensä seitsemän ja niiden alle sisällytettyjä alakategorioita oli yhteensä 36. Tästä syystä accordion listan tekemiseen meni hieman aikaa mutta olin lopulta tyytyväinen, että kaikki kategoriat oli selkeästi jaoteltu näkyviin jo heti projektin alussa. Tästä oli hyötyä myöhemmässä vaiheessa projektia.

Sivulle luotiin Bootstrapin avulla navigointipalkki. Käytin Bootstrapin sivuilla olevaa valmista koodia, jota muokkasin sellaiseksi kun sen halusin. Päädyin lisäämään brand merkin etusivun viereen Aapelille antaakseni palkille hieman ilmettä.

Seuraava vaihe oli luoda kirjautuminen verkkosivulle, joka tarkoitti sitä, että vaihdoin kaikki sivut PHP-muotoon ja otin työkaluksi käyttöön XAMPP-ohjelman. Kirjautumiselle luotiin index.php sivu. Sivulle asetettiin kaksi label kenttää, toinen käyttäjätunnukselle ja toinen salasanalle. Lisäsin myös ”muista minut” valinnan siltä varalta, että Irmeli Keränen haluaisi selaimen muistavan hänen kirjautumistietonsa esimerkiksi kotikoneella. Kaikkien näiden alle lisättiin kirjaudu-painike.

Käyttäjätunnuksen luontia varten tein erillisen rekisteröityminen.php-tiedoston, jossa oli samanlaisia label-kenttiä. Sivulla pyydettiin antamaan haluttu käyttäjätunnus sekä salasana. Ennen käyttäjän luontia olin tehnyt ”users” -taulukon phpMyAdminin kautta ja asettanut koodin toimimaan niin, että rekisteröityminen siirtäisi käyttäjätunnuksen ja salasanan suoraan tietokantaan (kuva 9). Käyttäjä tietoja ei lisätä tietokantaan käsin vaan nimenomaan rekisteröitymislomakkeella. Lomakkeen koodi muokkaa rekisteröitymisen yhteydessä käyttäjän salasanan

hashattyyn muotoon. Käyttäjän salasana saadaan tällä tavoin sotkettua niin, että se on vaikeampi murtaa. Salasana tallentuu tietokantaan hashatyssä muodossa eikä oikeaa salasanaa voida nähdä tietokannassa. Näin Irmeli Keräsen kirjautuminen on turvallisempaa ja tietokannassa olevat käyttäjätiedot on suojattu. Tällä lisätään myös sivuston turvallisuutta yleisesti. Verkkosivulle ei pääse ilman kirjautumista eikä kukaan voi luoda sivustolle lisää käyttäjiä, sillä rekisteröitymistä ei ole lisätty sivustolle.

```
// Lähetään käyttäjätunnus ja kryptattu salasana tietokantaan
$sql="INSERT INTO users (username,password) VALUES('$username','$encrypt_password')";
if ($conn->query($sql) === TRUE) {
    echo "Uusi käyttäjä luotu";
} else {
    echo "Virhe käyttäjätunnuksen luomisessa: " . $sql . "<br>" . $conn->error;
}
```

Kuva 9. Rekisteröinti lomakkeen käyttäjätietojen siirto tietokantaan.

Etusivulla ei ole tällä hetkellä muuta kuin ”Tervetuloa arkistoon” -viesti, joka on asetettu erillisen sisältö laatikon sisään, jotta se pysyy sille tarkoitetulla paikallaan. Myöhemmin mikäli Irmeli Keränen haluaa sisältöä lisätä tai muuttaa etusivulla, tämä onnistuu helposti, kun sisällölle on määritetty nimenomainen laatikko.

”Lisää Tiedostoja” -sivulle lisättiin tiedostojen talletukseen luotu ikkuna sisältö laatikon sisälle. Laatikkoon asetettiin kentät, joihin käyttäjä antaa lisättävän tiedoston tiedot, joiden mukaan tiedoston tiedot tallentuvat tietokantaan. Näitä tietoja käytetään tiedostojen hakemiseen ja listaamiseen. Tiedostolle pyydetään antamaan nimi, kuvaus, kategoria sekä alakategoria. Kategoriat valitaan alasvetovalikon kautta, jonne on tallennettu kaikki Irmeli Keräsen ennalta suunnittelemat kategoriat. Alakategoria valikossa ei alussa ole mitään, vaan valinnat riippuvat siitä, mitä kategoriaan on valittu. Näin kategorian alle ei voida tallentaa tiedostoa väärällä alakategorialla.

Kategoria-valikoiden toiminnan luominen oli yllättävän vaikeaa. Aluksi suunnitelma koodia varten oli todella erilainen, mutta en saanut sitä toimimaan. Pyysin tästä syystä mielipidettä vanhalta luokkalaiseltani, sillä hän on käyttänyt JavaScript-kieltä enemmän kuin minä. Olin tutkinut vaihtoehtoja ennen tätä itse,

joten tiesin, että toiminnallisuus luotaisiin JavaScript-kielellä. Suurin ongelma oli siinä, että kategorioita ja niiden alakategorioita oli paljon.

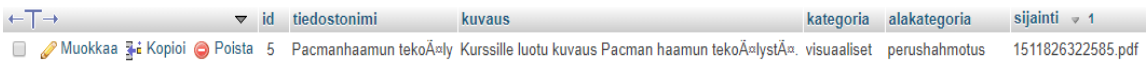
Ystäväni avulla päädyimme ratkaisuun luoda skriptin, jonka sisällä tarkistetaan pää kategorian arvo. Arvon perusteella annetaan valittavat vaihtoehdot alasvetovalikkoon. Tarkistaminen on luotu if- ja else if -lausekkeilla, joissa käydään jokainen kategoria läpi erikseen (kuva 10).

```
    <script>
function changeCategory(s1,s2){
    var s1 = document.getElementById(s1);
    var s2 = document.getElementById(s2);
    s2.innerHTML = "";
    if(s1.value == "kielelliset"){
        var optionArray = ["|Valitse kategoria","sensorinenaannenanalyysi|Sensorinen äär
        "motorinentuottaminen|Dynaaminen/Efferentti motorinen (kineettinen) tuottaminen"
        "kasitteidenhallinta|Käsitteiden hallinta","kielellistenmerkitysten|Kielellister

    } else if(s1.value == "visuaaliset"){


        var optionArray = ["|Valitse kategoria","perushahmotus|Perushahmotus","visuospat
        "muisti|Muisti"];

    } else if(s1.value == "lukivaikeus"){
        var optionArray = ["|Valitse kategoria","fonologinenprosessointi|Fonologinen pro
    } else if(s1.value == "lukivaikeusjakirjoitus"){

        var optionArray = ["|Valitse kategoria","visuaalinen|Visuaalinen","afferentti|Af
    } else if(s1.value == "matematiikka"){

    var optionArray = ["|Valitse kategoria","visuaalinenhahmotus|Visuaalinen hahmotus/ta
        "kasitteidenhallinta|Käsitteiden hallinta/muisti","semanttinenvaikeus|Semanttine
        "perakkaisenprosessoinnin|Peräkkäisen prosessoinnin hallinta","rinnakkaisenprose
    } else if(s1.value == "aidinkieli"){

    var optionArray = ["|Valitse kategoria","alkavalukutaito|Alkava lukutaito (kirjain-ä
    "lausetaso|Lausetaso", "tarinataso|Tarinataso","kielellinenoma|Kielellinen oma tuott
    "kielellinenpaattely|Kielellinen päättely","luetunymmartaminen|Luetun ymmärtäminen",
    "kasitteet|Käsitteet/Sanavarasto","kielioppi|Kielioppi (sanaluokat, sijamuodot, vuor
    }else if(s1.value == "vieraatkielet"){

        var optionArray = ["|Valitse kategoria","lauserakenteet|Lauserakenteet","sanasto
    }

    for(var option in optionArray){
        var pair = optionArray[option].split("|");
        var newOption = document.createElement("option");
        newOption.value = pair[0];
        newOption.innerHTML = pair[1];
        s2.options.add(newOption);
    }
}
</script>
```

Kuva 10. Kategorian tarkastus silmukka.

Dropzone.js-kirjastolla luotu drag'n'drop alue on lisää tiedostoja sivun viimeinen vaihe. Käyttäjän annettua lisättävälle tiedostolle halutut kuvaukset voi hän siirtyä valitsemaan lisättävää tiedostoa. Se onnistuu joko raahaamalla ja tiputtamalla tiedosto sivulle tai klikkaamalla sille tarkoitettua tilaa.

Tiedoston lisäämisen jälkeen sille annetut tiedot siirtyvät tiedostolistaus tietokantaan, joka on tehty phpMyAdmin puolella. Tietokantaan tallentuu tiedoston tietoihin myös sille yksilöity id sekä sijainti. Sijainti on tiedostolle satunnaisesti luotu uusi nimi koostuen pelkistä numeroista. Tiedosto tallentuu sillä nimellä uploads kansioon, josta se löytyy sen uudella nimellä. Käyttäjän ei tarvitse tietää tiedoston uutta nimeä. Tämä varmistaa sen, että ladatuissa tiedostoissa ei ole missään vaiheessa duplikaatti tiedostoja, eikä tiedostoja ole koskaan nimetty samalla tavalla vaikka niille antaisi lisäys vaiheessa saman nimen. Näin vältytään sotkuilta ja ylikirjoittamiselta. Tietokanta yhdistää tiedoston uuden sekä vanhan nimen, jolloin verkkosivun on helppo yhdistää oikea tiedosto oikeaan nimeen (kuva 11).



Kuva 11. Tiedosto arkiston tietokannassa.

”Hae tiedostoja” -sivulle on asetettu yksi label-kenttä sekä kaksi alasvetovalikkoa, joiden avulla tiedostoja voidaan hakea listattavaksi. Label-kentässä käyttäjä voi hakea tiedostoja tiedostonimen perusteella. Tiedosto löytyy antamalla kokonainen tiedostonimi tai tiedostonimen osa. Tällä haulla on oma erillinen PHP-tiedosto, joka käsittelee haun ja siirtää haun tulokset taulukkoon, jonne käyttäjä ohjataan hakemisen jälkeen. Ensimmäinen alasvetovalikko käsittää kaikki pääkategoriat. Tällä haulla on myös erillinen PHP-tiedosto, joka tekee saman kuin tiedostonimellä hakemiseen liitetty tiedosto. Toinen alasvetovalikko on vaihtoehto, jolla käyttäjä voi hakea tiedostoja alakategorian mukaan. Alakategorian haulla on myös oma erillinen PHP-tiedosto.

Jokainen haku siirtää käyttäjän uudelle sivulle, joka listaa hakutulokset. Taulukossa ilmoitetaan tiedoston kategoria, alakategoria sekä tiedoston nimi. Taulukossa on myös nappi, jota painamalla tiedosto voidaan ladata. Word-tiedostot

latautuvat käyttäjälle koneelle mutta esimerkiksi PDF-muodossa olevat dokumentit voivat avautua käyttäjälle selaimeen suoraan. Viimeisessä taulukon sarakkeessa on linkki tiedoston poistamiselle, kun poista linkkiä painetaan, siirretään käyttäjä poistamisen varmistamiseen, jotta voidaan välttyä tiedostojen vahinko poistamiselta. Jos käyttäjä valitsee varmistuksessa tietueen poistamisen, poistuu tiedosto uploads-kansiosta sekä tietokannasta. Näin käyttäjä voi poistaa tiedostot helposti verkkosivun kansioista ja tiedoista ilman tietoteknistä osaamista. Tällä tavalla vältytään myös siltä, että kansio täyttyy turhista tiedostoista, joita käyttäjä ei enää tarvitse.

Selaa tiedostoja oli etusivun jälkeen yksinkertaisin sivu tehdä. Sivu pitää sisällään taulukon, johon listataan kaikki arkistoon talletetut tiedostot. Tiedostot näkyvät samalla tavalla kuin tiedostojen yksittäisen hakemisen jälkeen. Taulukossa on esillä samat tiedot kuin haku taulussa.

Koko sivustolla on näiden kaikkien lisäksi muita koodin pätkiä ja tiedostoja, joiden avulla sivu toimii ja rakentuu. Selkeimmät niistä ovat ulkoasuun ja sijoitteluun liittyviä. Navigointipalkki, jonka avulla sivulla siirrytään paikasta toiseen, on luotu Bootstrapin sekä HTML-kielen avulla. Navigointipalkin koodi on lisätty erikseen jokaiselle käyttäjälle näytettävälle sivulle. Navigoinnin koodin olisi voinut tehdä myös erilliseen tiedostoon mutta alasivuja oli sen verran vähän, että päädyin pitämään sen jokaisessa koodissa erikseen. Riippuen siitä millä osalla sivustoa käyttäjä on, vaihtuvat navigointipalkin linkit aktiivisiksi. Irmeli Keräsen yrityksen nimi on lisätty navigointipalkin vasempaan reunaan tuomaan sille hieman ilmettä. Kuvassa 12 on esimerkiksi etusivu.php tiedoston navigointipalkin koodi. Koska kyseessä on etusivu, on sen nav-item-tagin määritykseksi laitettu active.

```
<!--Navigaatio alku-->
<nav class="navbar navbar-toggleable-md navbar-inverse bg-inverse">
    <button class="navbar-toggler navbar-toggler-right" type="button" data-toggle="collapse" data-target="#navbarNav"
    aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
    </button>
    <a class="navbar-brand" href="etusivu.php">Aapeli Arkisto</a>
    <div class="collapse navbar-collapse" id="navbarNav">
        <ul class="navbar-nav">
            <li class="nav-item active">
                <a class="nav-link" href="etusivu.php">Etusivu<span class="sr-only">(current)</span></a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="lisaatiedosto.php">Lisää Tiedostoja</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="haku.php">Hae Tiedostoja</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="lista.php">Selaa Tiedostoja</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="logout.php">Kirjaudu Ulos</a>
            </li>
        </ul>
    </div>
</nav>
<!--Navigaatio loppu-->
```

Kuva 12. Navigaation HTML-koodi.

Navigaation viimeinen linkki on ”Kirjaudu Ulos”. Se on sisällytetty koodiin samalla tavalla kuin muutkin eri sivulle ohjaavat linkit. Uloskirjautumisen toiminnallisuus on tehty erikseen logout.php tiedostossa, johon linkki viittaa. Tämä koodin pätkä on todella lyhyt mutta tärkeä, sillä se katkaisee käyttäjän session (kuva 13). Avonaiseen sessioon voisi joku yrittää päästä sisälle väkisin mutta session sulkeminen varmistaa sen, että se ei jää turhaan auki eikä se avaa sivua tietoturvarikkomuksille. Uloskirjautumisen jälkeen käyttäjä ohjataan takaisin index.php tiedostosta toimivaan sivuun, joka on tässä tapauksessa arkiston kirjautumissivu.

```
<?php
session_start();

if(session_destroy()) {
    header("Location: index.php");
}
?>
```

Kuva 13. PHP-koodi istunnon lopettamiselle.

Sivustoon kuuluu myös muutama muu PHP-tiedosto, joilla lisätään sivun toiminnallisuutta. Nämä tiedostot eivät myöskään ole käyttäjälle suoraan näkyvissä vaan ne toimivat sivun taustalla pyörittämässä toiminnallisuuksia ja tietokantayhteyksiä. Tiedostoja ovat esimerkiksi config.php ja db_config.php. Näissä

tiedostoissa määritellään tietokannat ja niihin liittyvät tiedot kuten tietokannan nimi sekä palvelimen localhost osoite.

Tiedostojen poistamista ajaa tiedosto nimeltä poistatiedosto.php. Tämän tiedoston sisällä on luotu toiminnallisuus tiedoston poistamiselle sekä uploads-kansiosta, että tietokannasta.

Tiedostojen lataamista tukemaan on luotu tiedosto upload.php. Tiedostossa siirretään käyttäjän antamat tiedot sivuston tietokantaan ja kerrotaan sivulle, että tiedosto lisätään uudella nimellään uploads-kansioon.

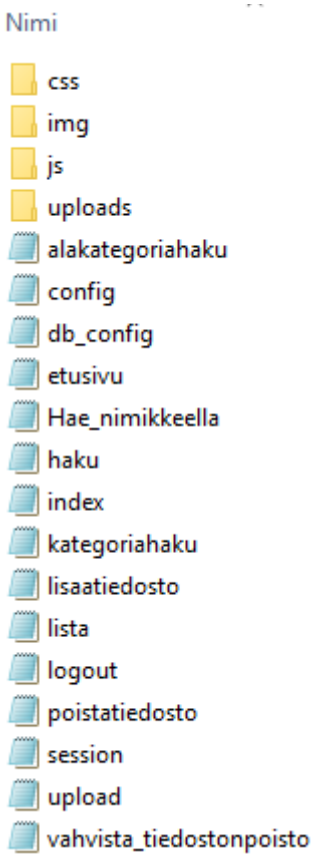
Sivustoon kuuluu myös tiedosto nimeltä session.php. Session määrittää istunnon aloittamisen, kun käyttäjä kirjautuu sisään. Sessio pysyy aktiivisena siihen asti kunnes käyttäjä kirjautuu ulos sivustosta. Sessio tiedosto on lisätty jokaisen sivun PHP-koodiin, jotta istunto voidaan pitää aktiivisena.

Sivuihin kuuluu olennaisena osana myös CSS-tiedostot, joita käyn läpi tarkemmin seuraavassa luvussa, jossa käsitellään sivuston ulkoasua. CSS-tiedostoja on sivuilla käytössä kolme kappaletta kaiken kaikkiaan. Kaksi näistä tiedostoista on sivun CSS-kansiossa ja kolmas on lisätty vain linkkinä, sillä se viittaa Bootstrapin tyylitiedostoon. CSS-kansiossa olevat tiedostot ovat Dropzonen mukana tullut CSS-tiedosto ja toinen on minun itse kirjoittamani, joka on luotu tätä arkistointi sivua varten.

Viimeinen kansio ja tiedosto, jota en ole tässä dokumentissa vielä maininnut on JavaScript-tiedosto, jolla Dropzone toimii. Tähän tiedostoon en ole tehnyt itse muutoksia ja se on saatu Dropzonen sivuilta. Se ajaa tiedostojen lisäämiseen luotua drag’n’drop ikkunaa.

Tiedostoja kertyi koko projektin aikana useampia (kuva 14), sillä oli selkeämpää ja helpompaa luoda osalle toiminnallisuuksille oma tiedosto sen sijaan, että se olisi sisällytetty varsinaiseen sivun PHP-koodiin. Tällä tavalla pelkkää toiminnallisuuden koodia oli nopeampi ja selkeämpi muokata. Toiminnallisuuden luomisessa minulle tuli usein ongelmia, koska en ollut aiemmin näitä ohjelmointikieliä

käyttänyt. Näin pystyin siirtymään suoraan toiminnallisuuden omaan tiedostoon muokkaamaan koodia, jolloin vältyin siltä, että sotkisin tai muokkaisin vahingossa sivujen muuta koodia.



Kuva 14. Verkkosivun tiedostot.

4.3 Ulkoasu

Verkkosivun ulkoasu on yritetty pitää yksinkertaisena ja selkeänä mutta samalla tyylikkäänä. Sivun väreinä toimivat pääosin musta, valkoinen ja harmaa, joiden kontrastiksi on koko sivun taustakuvaksi valittu toistuva vaalea puukuvio. Liiallinen värin käyttö on jätetty pois, jotta sivusto pysyisi rauhoittavana. Sisällön lukemisen puolesta oli tärkeää myös se, että tekstin väri on valittu niin, että se erottuu taustastaan selkeästi. Tämä on syy sille, miksi sisältö osion tausta on vaihdettu tumman harmaaksi. Sivuilla oleva teksti ja muu sisältö eivät erottuisi pohjimmaisena olevasta puukuviosta ilman tätä kerrosta.

Verkkosivun taustakuva on ladattu ja otettu käyttöön Pattern Library -sivulta. Pattern Libraryyn on koottu useita eri henkilöiden luomia kuvioita, joita saa ottaa vapaasti omaan käyttöön (kuva 15).

Kuva 15. Pattern Library.

Arkiston navigointipalkki on pidetty yksinkertaisena ja pienenä, jolloin se ei vie huomiota muun sivun käytöltä. Sama syy pätee sivustolla oleviin muihin ulkoasuun liittyviin ominaisuuksiin, kuten taulukoihin ja lomakkeisiin.

Sivustosta voisi helposti tehdä näyttävämmän, mutta tässä projektissa tärkeintä oli toiminnallisuus, joten ulkoasun luomiseen ei varattu paljoa aikaa. Muutoksia voidaan tehdä myöhemmin, jos sille on tarvetta.

Arkistolle luotu oma tyylitiedosto ei ole pitkä ja sisältää kaiken tähän mennessä tarpeellisen (kuva 16). Tiedoston avulla on pääosin aseteltu sivulla olevia lokeroita, joissa sisältöä tai toiminnallisuutta on. Bodyyn on määritelty taustakuvan ominaisuuksia, joka on aiemmin mainittu vaalea puukuvio. Sisältö on lokero, jonne kaikki sivustolla oleva sisältö asetetaan. CSS-koodissa sille on määritelty taustaväri, joka on sama tumman harmaa kuin navigointipalkilla. Muita määriteltyjä ominaisuuksia on laatikon ympärillä oleva tyhjä tila ja laatikon vähimmäispituus. Vähimmäispituus aiheuttaa sivulla sen, että tumma sisältölaatikko ei lopu kesken, mikäli sivu kasvaa vieritettäväksi. Dropdiv sisältää määrityksiä Dropzone alueen asettelulle ja sen ulkonäölle. Muut CSS-tiedostossa määritellyt asiat ovat kutakuinkin samanlaisia. Niissä on määritelty asetteluja esimerkiksi tiedoston latauksen luotuun lomakkeeseen, taulukon sarakkeen fonttiväriä ja rekisteröitymislomakkeen asetteluja.

```css
html, body {
  height: 100%;
}

body {
  background-image: url("../img/white-wood.jpg");
  background-position: center center;
  background-repeat: repeat;
  background-attachment: fixed;
  display: flex;
}

.sisalto {
    margin-top: 2%;
    background-color: #292B2C;
    color: white;
    padding-left: 1%;
    padding-right: 1%;
    padding-top: 1%;
    padding-bottom: 2%;
    min-height: 100%;
}


.dropdiv {
  position: fixed;
  top: 60%;
  left: 50%;

  transform: translate(-50%, -50%);
  width: 50%;
  border: 10px dashed black;
  padding: 10px;
  border-radius: 15px;
  opacity: 0.5;
}

.flexbox {
  position: fixed;
  top: 10%;
  left: 50%;

  transform: translate(-50%, -50%);

  display: flex;
  align-items: center;
  justify-content: center;
  width: 400px;
  height: 60px;
  background-color: black;
  opacity: 0.5;
  border-radius: 15px;
}

.flex-item {
  padding: 5px;

  line-height: 20px;
  color: white;
  font-weight: bold;
  font-size: 2em;
  text-align: center;
}

.col-2 {
    color: black;
}

.insert {
  position: fixed;
  top: 55%;
  left: 50%;
  /* bring your own prefixes */
  transform: translate(-50%, -50%);
  width: 50%;

  padding: 10px;
  border-radius: 15px;

}
.uploadForm{
  padding-top: 50px;
  background-color: LightGray;
  padding-left: 50px;
  border-radius: 15px;
  padding-bottom: 50px;
  color: black;
}

.regform {
  position: fixed;
  top: 60%;
  left: 50%;

  transform: translate(-50%, -50%);
  width: 40%;
  border: 10px solid black;
  padding: 10px;
  border-radius: 15px;
  opacity: 0.9;
}
.reginputs
{
  margin-left: 15px;
}
```

Kuva 16. Arkiston CSS-koodi.

Toinen sivustolle olennaisesti tärkeä CSS-tiedosto on Dropzonen oma CSS-koodi. Tähän koodiin en ole itse tehnyt muokkauksia tai lisäyksiä. Sen lisääminen koodiin on kuitenkin tärkeää Dropzone-laatikon ulkonäölle ja toimivuudelle. Tiedosto on saatu Dropzonen JavaScript-tiedoston mukana (kuva 17).

```
/*
 * The MIT License
 * Copyright (c) 2012 Matias Meno <m@tias.me>
 */
@-webkit-keyframes passing-through {
  0% {
    opacity: 0;
    -webkit-transform: translateY(40px);
    -moz-transform: translateY(40px);
    -ms-transform: translateY(40px);
    -o-transform: translateY(40px);
    transform: translateY(40px); }
  30%, 70% {
    opacity: 1;
    -webkit-transform: translateY(0px);
    -moz-transform: translateY(0px);
    -ms-transform: translateY(0px);
    -o-transform: translateY(0px);
    transform: translateY(0px); }
  100% {
    opacity: 0;
    -webkit-transform: translateY(-40px);
    -moz-transform: translateY(-40px);
    -ms-transform: translateY(-40px);
    -o-transform: translateY(-40px);
    transform: translateY(-40px); } }
@-moz-keyframes passing-through {
  0% {
    opacity: 0;
    -webkit-transform: translateY(40px);
    -moz-transform: translateY(40px);
    -ms-transform: translateY(40px);
    -o-transform: translateY(40px);
    transform: translateY(40px); }
  30%, 70% {
    opacity: 1;
    -webkit-transform: translateY(0px);
    -moz-transform: translateY(0px);
    -ms-transform: translateY(0px);
    -o-transform: translateY(0px);
    transform: translateY(0px); }
  100% {
    opacity: 0;
```

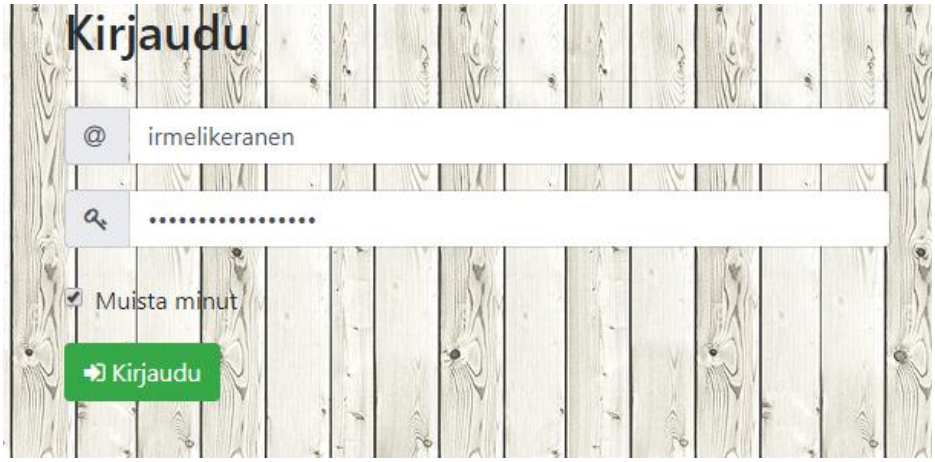
Kuva 17. Dropzone-tyylitiedosto.

5 Tutkimustulos

Tässä luvussa esittelen kuvin lopullisen arkistointisivun. Miltä se näyttää ja kuinka se toimii. Esittelyssä käydään läpi myös sivun kaikki toiminnallisuudet. Toimeksiantajan käyttöönotossa puhutaan tarkemmin siitä, kuinka sivu lopulta otetaan käyttöön.

5.1 Arkiston esittely

Arkiston ensimmäinen osa on kirjautumissivu, jossa käyttäjän tulee antaa käyttäjätunnus sekä siihen liitetty salasana (kuva 18). Halutessaan käyttäjä voi valita muista minut, jolloin käyttäjätunnus sekä salasana tallennetaan käyttäjän selaimeen. Kenttien täytön jälkeen käyttäjä painaa ”kirjaudu”-painiketta.



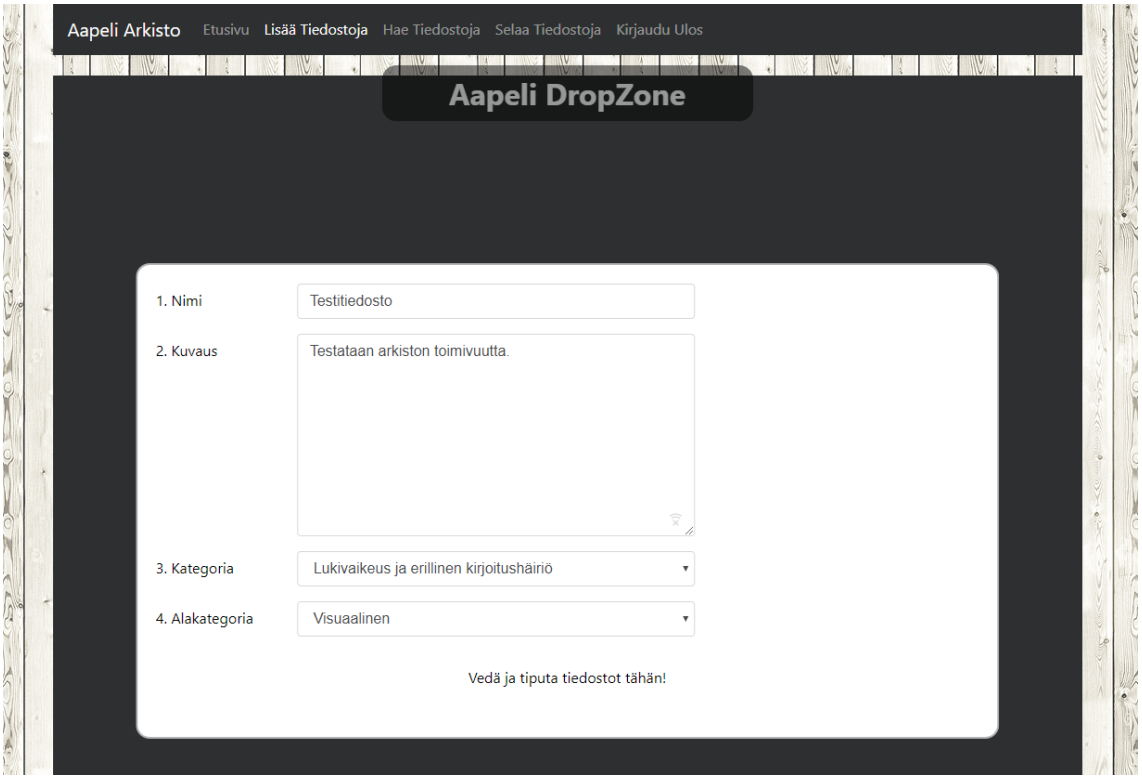
Kuva 18. Arkistoon kirjautuminen.

Oikein annettu käyttäjätunnus ja salasana yhdistelmä ohjaa käyttäjän arkiston etusivulle (kuva 19). Etusivulla ei ole tässä vaiheessa vielä paljoa tekstiä tai ominaisuuksia. Toimeksiantaja voi lisätä sivulle esimerkiksi tekstiä myöhemmin halutessaan. Sille on luotu valmiiksi laatikko, jonka sisään sisältöä voidaan tuottaa.

Kuva 19. Arkiston etusivu.

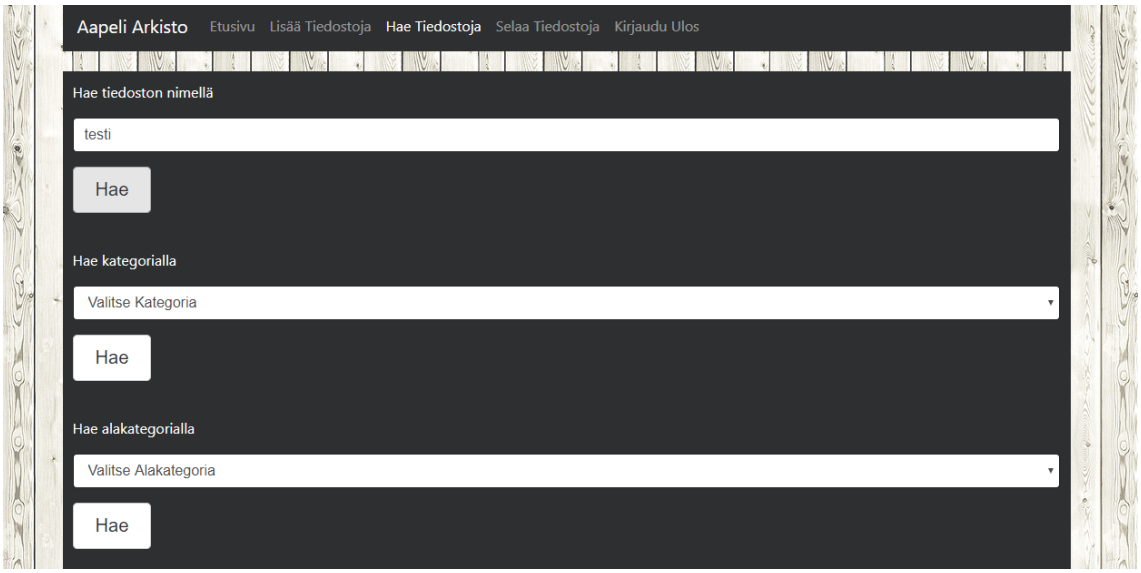
Sivun ylälaidassa näkyy sivuston navigointipalkki, jonka avulla sivustolla liikutaan. Se on näkyvissä jokaisella sivulla samalla tavalla.

”Lisää Tiedostoja”-linkin alta päästään sivulle, jossa käyttäjä voi ladata tiedostojaan arkistoon (kuva 20). Tiedostolle annetaan nimi ja kuvaus. Kategoria valitaan valmiista valikossa olevista vaihtoehdoista. Alakategorian vaihtoehdot määrittyvät annetun yläkategorian mukaan. Tietojen täyttämisen jälkeen voi käyttäjä joko klikata pohjalla olevaa aluetta, jossa lukee ”Vedä ja tiputa tiedostot tähän!” tai vaihtoehtoisesti tiedosto voidaan raahata tekstin päälle. Molemmat tavat toimivat.



Kuva 20. Tiedostojen lisääminen arkistoon.

Seuraavalla välilehdellä voidaan hakea arkistoon tallennettuja tiedostoja. Hakea voidaan joko tiedoston nimellä, kategorialla tai alakategorialla (kuva 21). Tiedostonimi löytää tiedoston koko tiedostonimellä tai tiedostonimen osalla.



Kuva 21. Tiedostojen hakeminen arkistosta.

Tiedoston hakemisen jälkeen käyttäjä ohjautuu uudelle sivulle, jossa hakutulokset näytetään. Hakutulokset listataan taulukkoon, jossa tiedostosta näytetään kategoria, alakategoria ja tiedoston nimi (kuva 22). Taulukosta käyttäjä voi myös ladata tiedoston valitsemalla ”Avaa”.

Kuva 22. Hakutulokset ja niiden näyttäminen.

Viimeinen vaihtoehto taulussa on ”Poista”. Tätä klikkaamalla käyttäjä ohjataan poistamisen varmistukseen, jossa käyttäjä voi vielä peruuttaa tiedoston poiston, jos painiketta on klikattu vahingossa (kuva 23). Poiston peruuttaminen siirtää käyttäjän ”Selaa Tiedostoja”-sivulle. Poistamisen vahvistaminen ohjaa käyttäjän samalle sivustolle, mutta tiedosto poistuu tietokannasta sekä verkkosivulta. Sitä ei löydy tämän jälkeen hakemalla eikä tiedostojen selauksesta.

Kuva 23. Tiedoston poistamisen vahvistus.

”Selaa Tiedostoja”-osio on luotu tiedostojen vapaata selaamista varten. Sivulla on samanlainen taulukko kuin hakutulosten annossa, mutta täällä kaikki arkistosta löytyvät tiedostot on listattu taulukkoon samaan aikaan (kuva 24). Taulukossa on vaihtoehdot avaamiselle sekä poistamiselle.

Kuva 24. Tiedostojen selaaminen.

Viimeinen navigaatiopalkin osio on ”Kirjaudu Ulos”, jota painamalla käyttäjä kirjautuu ulos istunnosta ja palautuu kirjautumissivulle.

5.2 Sivuston käyttöönotto

Verkkosivu esitellään ja otetaan käyttöön toimeksiantajalle vuoden 2018 alussa. Toimeksiantajalle opastetaan sivun käyttöä samassa tapaamisessa. Erillistä käyttöohjetta sivuston käyttöön ei luoda, sillä Irmeli Keränen on sivun ainoa käyttäjä. Tästä syystä on siis helpompaa ohjeistaa hänelle sivun käyttöä yhden tapaamisen aikana.

Arkistointisivu siirretään uuteen webhotelliin samaan aikaan yrityksen omien verkkosivujen kanssa, jotta ylimääräistä siirtelyä ei tarvitse tehdä. Sivun tekijänä olen luvannut hoitaa verkkosivujen siirtämisen webhotelliin.

Olen luvannut toimia tukena sivuston käytössä projektin valmistumisen jälkeen. Sivustolla voidaan huomata ongelmia vasta käyttöönoton jälkeen, sivua voidaan haluta parannella tai päivittää, joten halusin varmistua siitä, että toimeksiantaja ei jää sivun ongelmien kanssa yksin eikä joudu pyytämään ulkopuolista henkilöä korjaamaan sivun vikoja.

Haluaisin myöhemmin kehittää verkkoarkistoa lisää, mikäli sille on aikaa. Ulkoasua voisi hioa vielä ja erilaisia toiminnallisuuksia muokata tai lisätä. Yhtenä esimerkkinä tiedoston hakeminen voisi toimia niin, että siellä voidaan samaan aikaan hakea tiedostoa nimellä sekä kategorialla.

Mikäli tämä sivusto halutaan ottaa käyttöön jollain toisella henkilöllä tai työyhteisöllä tulisi rekisteröintilomake lisätä johonkin yhteyteen, jolloin uusia käyttäjiä voitaisiin luoda.

6 Yhteenveto

Kaiken kaikkiaan olen henkilökohtaisesti nauttinut projektista paljon. Sivujen tekeminen on antanut minulle päivitettyä osaamista verkkosivujen luontiin sekä mahdollisuuden luoda työkalun, jota oikeasti tarvitaan. Olen projektin alkuvaiheilla tutustunut erilaisiin oppimishäiriöihin, joka on saanut minut arvostamaan työtäni enemmän projektin edetessä. Verkkosivuja tekeviä yrityksiä on paljon mutta pienemmälle yrittäjälle se voi tulla todella kalliiksi ja henkilölle, jolla ei ole suurta tietoteknistä tietämystä tai osaamista ei välttämättä tulisi mieleenkään edes pyytää tämän kaltaista verkkosivua tehtäväksi. Tunnen siis tehneeni tärkeää työtä opinnäytetyöni toiminnallisen puolen osalta.

Syy miksi valitsin tähän dokumenttiin kirjoittaa verkkosivukehitykseen liittyvistä ohjelmointikielistä sekä valmiista verkkosivu alustoista liittyy paljon siihen, mitä kirjoitin edellisessä kappaleessa. On paljon ihmisiä, joilla ei ole tietämystä näistä asioista mutta haluaisivat tietää niistä. Kuinka tämänlaisia ohjelmointikieliä tulisi lähestyä, mitä ne ovat, mistä ne ovat tulleet ja mihin niitä käytetään. Kysymykset voivat tuntua suurilta silloin kun asiasta on joko vähän tai ei yhtään tietämystä. Kysymyksiä voi olla vaikea muotoilla tai niitä ei osata kysyä ollenkaan. Halusin antaa hyvää perustason tietoa kaikille, joita tämä kiinnostaa. Halusin myös kirjoittaa sen sellaisella tavalla, että siitä saisi jotain irti ilman minkäänlaista pohjatietoa.

Verkkosivu alustoja on olemassa paljon ja niitä tulee koko ajan lisää. Verkkosivuja itsessään tulee myös koko ajan lisää. Verkkosivujen tekeminen tuntuu mahdottomalta silloin, jos ei tiedä kuinka niitä tehdään. Voi olla myös pelkoa siitä, että ei omista osaamista ohjelmointikieliin. Tällöin ensimmäinen verkkosivu on helppo tehdä valmiilla toimivalla alustalla, jonka voi ottaa käyttöön vaikka ilmaiseksi. Sivu ei silloin sido käyttäjää maksamaan siitä vaikka sivua ei koskaan myöhemmin käyttäisikään. Tämän kaltainen toiminta rohkaisee ihmisiä kokeilemaan sivujen luomista ja se auttaa heitä hahmottamaan verkkosivujen toimivuutta.

Valitsin esiteltäväksi alustaksi WordPress-palvelun nimenomaan siksi, että halusin antaa ihmisille vaihtoehdon aloittaa jostain vaikka aiempaa tietoa ei olisi olemassa. Monet muut suositut alustat ovat hyviä mutta valitettavasti niiden käyttöönotto vaatii jo jonkin tasoista osaamista ja yksinomaan se voi pelottaa ihmisiä. WordPress on helppo paikka aloittaa.

Koen, että kaikesta tästä voi olla hyötyä niille, jotka haluavat aloittaa verkkosivukehityksen oppimisen. Toivon, että ne, jotka sen jo osaavat, saavat tästä jotain tietoa myös itselleen esimerkiksi CoffeeScript oli minulle uusi asia vaikka tiedän verkkosivukehityksestä asioita ja olen tehnyt verkkosivuja pidemmän aikaa. Verkkosivusta itsestään toivon, että joku saa idean luoda vastaavanlaisia apuvälineitä muille ja jopa kehittää ideaa eteenpäin. Työkalut ovat tärkeitä, varsinkin kiireisille ihmisille, joten niiden luonti ja kehittäminen ovat tärkeitä asioita.

Verkkosivuun olen tyytyväinen, sillä se sisältää kaikki halutut ominaisuudet ja ne toimivat niin kuin niiden haluttiin toimivan. On tietenkin asioita, joihin olisi varmasti ollut parempia menetelmiä tai ratkaisuja mutta nämä ovat ne, jotka valitsin. En ollut ennen tätä projektia kirjoittanut PHP- tai JavaScript-koodia. Opiskelin ne tätä projektia varten ja kirjoitin niitä nyt ensimmäistä kertaa. Tästä syystä olen ylpeä aikaansaannoksestani.

Sivun luonti ei ollut helppoa ja siihen kului paljon aikaa. Uudella ohjelmointikielellä kirjoittaminen on aina vaikeaa. Silloin täytyy selvittää paljon perustason asioita ja oppia soveltamaan tietoa siihen, mitä on itse tekemässä.

Ulkonäöllisesti sivu voisi varmasti olla näyttävämpi ja kauniimpi, mutta se ei ollut tämän projektin päätarkoitus, joten en ole siitä huolissani. Sivun ulkonäköä voidaan muokata myöhemmin, vaikka sivu olisi jo otettu käyttöön. Sivu toimii ilman ulkoasua, mutta se ei toimisi ilman haluttuja toiminnallisuuksia. Aika kävi verkkosivujen toteutuksen osalta vähiin nopeasti uusien asioiden opettelun takia, joten ulkoasu kärsi siinä paljon.

Myöhemmin verkkosivulle voitaisiin luoda uuden ulkoasun lisäksi mahdollisuus esimerkiksi tekstitiedostojen kirjoittamiselle, tai sivulle voitaisiin lisätä työkalenteri, jossa voisi seurata vastaanottoaikoja ja omaa aikataulutustaan. Mahdollisuudet ovat rajattomat. Arkisto voitaisiin ottaa käyttöön myös usean henkilön yritykseen, jolloin tiedostojen jakaminen yrityksen sisällä onnistuisi tämänlaisella verkkosivulla.

Olen oppinut tästä toimeksiantoprojektista ja opinnäytetyöraportin kirjoittamisesta paljon. Osaan käyttää XAMPP-ohjelmaa paremmin kuin aiemmin ja ymmärrän sen käyttötarkoituksia enemmän. Tiedän paljon asioita verkkosivukehityksen historiasta ja siitä, minne verkkosivukehitys on menossa. Voisin listata pienempiä asioita paljon, mutta tärkeimpänä pidän sitä, että opin vihdoin kirjoittamaan PHP- ja JavaScript-kieliä ja olen luonut niillä jotain, joka toimii.

Lähteet

Apache Friends. 2017. Apache Friends. https://www.apachefriends.org/index.html. 20.11.2017.

Apache HTTP Server Project. 2017. The Apache Software Foundation. https://httpd.apache.org/. 20.11.2017.

Arkistolaki 831/1994.

CoffeeScript. 2017. CoffeeScript. http://coffeescript.org/. 20.11.2017.

Heinisuo, R. 2004. PHP ja MySQL. Helsinki: Talentum.

Hoffman, J. 2017. A look back at the History of CSS. CSS-TRICKS. https://css-tricks.com/look-back-history-css/. 18.11.2017.

Javascript.info. 2017. Javascript.info. https://javascript.info/. 20.11.2017.

Korpela, J.K. 2014. HTML5-käsikirja. Jyväskylä: Docendo.

Laki potilaan asemasta ja oikeuksista 785/1992.

Nikkilä, T. & Malmirae, P. 1999. Internet. Jyväskylä: Teknolit.

Node.js. 2017. Node.js Foundation. https://nodejs.org/en/. 20.11.2017.

PHP. 2017. The PHP Group. http://php.net/. 20.11.2017.

Sarja, J. 2012. HTML:n perusteet. Otavan Opisto. http://opinnot.internetix.fi/fi/muikku2materiaalit/muut/ammatillinen/web/html/html_perusteet.pdf. 18.11.2017

W3C. 2017. W3C. https://www.w3.org/. 18.11.2017.

W3C. 2016. A brief history of CSS until 2016. W3C. https://www.w3.org/Style/CSS20/history.html. 28.11.2017.

W3Schools. 2017. W3Schools. https://www.w3schools.com. 18.11.2017.

WordPress. 2017. WordPress.org. https://wordpress.org/. 20.11.2017.