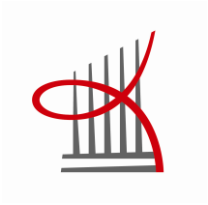


OPPIMISPOLKU VERKKO-OPPIMATERIAALEISSA

Tomi Mäenpää

Opinnäytetyö
Maaliskuu 2014
Tietojenkäsittelyn koulutus-ohjelma
Terveysalan tietohallinta

TAMPEREEN AMMATTIKORKEAKOULU
Tampere University of Applied Sciences

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietojenkäsittelyn koulutusohjelma
Terveysalan tietohallinta

MÄENPÄÄ, TOMI:
Oppimispolku verkko-oppimateriaaleissa

Opinnäytetyö 34 sivua
Maaliskuu 2014

Opinnäytetyö on osa Metsäteho Oy:n sekä Tampereen ammattikorkeakoulun metsätalouden koulutusohjelman yhteistä hanketta, jossa rakennetaan verkkopalvelu puuhuollon oppimateriaaleille. Tämän opinnäytetyön tavoitteena oli kehittää internetissä olevien oppimateriaalien esitystapaa. Opinnäytetyössä kerrotaan, miten oppimispolun on ajateltu toimivan ja kuinka sitä varten luotu sivusto suunniteltiin ja toteutettiin.

Oppimispolkua varten luotu sivusto jakautui kahteen suurempaan, toisiinsa liittyvään kokonaisuuteen: oppimispolun esittämiseen sekä sen luomiseen. Näitä varten suunniteltiin luonnokset, joiden pohjalta sivusto rakennettiin yleisimpiä WWW-tekniikoita käyttäen.

Loppuvaiheessa kävi ilmi, että jatkokehitystä varten oppimispolku tulisi siirtää jonkin sisällönhallintajärjestelmän päälle. Vaikka kaikkia haluttuja ominaisuuksia ei ehditty toteuttamaan, oli lopullinen tuotos onnistunut esimerkki siitä, kuinka oppimispolku voisi toimia.

Asiasanat: oppimispolku, www

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in Business Information Systems
Option of Data Management in the Field of Health Care

MÄENPÄÄ, TOMI:
Learning Path in Web-Based Learning Materials

Bachelor's thesis 34 pages
March 2014

This thesis is a part of a joint project between Metsäteho Ltd. and Degree Programme of Forestry at Tampere University of Applied Sciences, which aimed at creating a web service for forestry learning materials. The objective of this thesis was to improve the presentation of the web-based learning materials. The purpose was to describe how the learning path works and how the web page for it was designed and created.

The web page for the learning path was divided into two larger parts: presentation and creation of the learning path. After these two parts were sketched, the web page was created using common web technologies.

It became apparent in the end that before continuing development, the learning path should be incorporated onto some content management system. Although there was not enough time to implement all planned features, the end result was a successful example as to how the learning path could work.

Key words: learning path, www

SISÄLLYS

1 JOHDANTO 5
2 SIVUSTON SUUNNITTELU 6
2.1 Mikä on oppimispolku? 6
2.2 Käyttöliittymän suunnittelu 6
2.3 Luonnosten tekeminen 8
2.3.1 Polun esittäminen 8
2.3.2 Polun luominen 9
3 KÄYTETYT TEKNIIKAT 11
3.1 HTML 11
3.2 CSS 12
3.3 PHP 12
3.4 MySQL 13
3.5 jQuery 13
4 SIVUSTON TOTEUTUS 15
4.1 Rakenne 15
4.1.1 Tietokanta 15
4.2 Polun esittäminen 16
4.2.1 Polun hakeminen 16
4.2.2 Same-origin policy 19
4.2.3 Materiaalin lataaminen 19
4.2.4 Käyttöliittymän päivittäminen 21
4.2.5 Polussa liikkuminen 23
4.3 Kokoelma 24
4.4 Polun luominen 25
4.4.1 Polun muokkaaminen 27
4.4.2 Polun hallinta 28
4.5 Testaaminen 30
5 POHDINTA 31
LÄHTEET 33

1 JOHDANTO

Opinnäytetyöni taustalla on ajatus siitä, että internetissä olevan verkko-oppimateriaalin esitystapaa tulisi kehittää. Kehitettävää löytyisi myös oppimateriaaleissa, mutta tässä opinnäytetyössä keskitytään vain siihen, kuinka verkossa olevien oppimateriaalien käyttöä voitaisiin tehostaa.

Opinnäytetyöni tavoitteena on kehittää internetissä olevien oppimateriaalien esitystapaa. Opinnäytetyössäni kerron, kuinka suunnittelin sekä toteutin oppimispolun.

Perinteisen oppikirjassa olevan hierarkkisen rakenteen esittäminen verkossa ei ole välttämättä oppimisen kannalta tehokkain ja mielenkiintoisin tapa. Tästä syystä olen tarttunut ajatukseen oppimispolusta, jonka lähestymistapa poikkeaa normaalista oppikirjan rakenteesta melko radikaalisesti.

Olen ollut aina kiinnostunut kaikesta WWW-sivuihin liittyvästä, ja toiveenani onkin, että tulevaisuudessa löytäisin itseni tekemässä WWW-sivustoja työkseni. Tästä syystä valitsin opinnäytetyöni aiheeksi oppimispolun, koska sen tarkoituksena on tarjota käyttäjälle jotain uutta, ja minä saan mahdollisuuden kehittää sitä.

Toimeksiantaja opinnäytetyölleni on Metsäteho Oy:n sekä Tampereen ammattikorkeakoulun metsätalouden koulutusohjelman yhteinen hanke, jossa tarkoituksena on rakentaa verkkopalvelu puuhuollon yhtenäisille oppimateriaaleille. Metsäteho Oy on useiden metsäalan yritysten sekä organisaatioiden omistama yritys, jonka tavoitteena on tutkia ja kehittää erilaisia metsäalan toimenpiteitä.

Ajatus opinnäytetyössäni esiintyvästä oppimispolun toiminnasta on niin yleisluontoinen, että sen voidaan olettaa toimivan yhtä hyvin sekä puuhuollon että muidenkin alojen oppimateriaaleilla. Tavoitteena on, että opinnäytetyönä tehtävää sivustoa voidaan hyödyntää myös muissa vastaavissa yhteyksissä.

2 SIVUSTON SUUNNITTELU

2.1 Mikä on oppimispolku?

Oppimispolusta ei löydy sanakirjamääritelmää, mutta se koostuu sanoista ”oppia” (*omaksua opiskelemalla, opettamalla, harjoittelemalla jokin tieto tai taito, tietoja tai taitoja, hankkia oppia*) sekä ”polku” (*ihmisten tai eläinten polkema kapea kulku-ura*) (MOT 2011). Haluaisinkin aluksi avata sitä, mitä tarkoitan oppimispolulla (myöhemmin pelkkä polku), ja mitä se tarkoittaa tämän opinnäytetyön kannalta.

Polun tavoitteena tässä opinnäytetyössä on opettaa kokonaisuus jostain tietystä asiasta. Mikäli oppimisen kohteena olisi esimerkiksi perunoiden keittäminen, opettaja valitsisi siihen kuuluvat eri vaiheet (oppimateriaalit) ja lisäisi ne polulle haluamaansa järjestykseen. Opiskelija voisi sitten seurata polkua ja oppia perunoiden keittämisen käymällä vaihe vaiheelta läpi siihen liittyvät oppimateriaalit.

Yksi tämän opinnäytetyön polun rajoitteista on kuitenkin se, että vain yksi materiaali voi olla avoinna kerrallaan. Opettaja tulee asettamaan vaiheet tiettyyn järjestykseen, vaikka ne voitaisiinkin toteuttaa samaan aikaan tai satunnaisessa järjestyksessä. Esimerkiksi perunoita keittäessä opettajan tulisi järjestää vaiheet siten, että perunat täytyy ensin kuoria ja tämän jälkeen laittaa vesi kiehumaan, kun nämä vaiheet voitaisiin oikeasti toteuttaa samanaikaisesti.

2.2 Käyttöliittymän suunnittelu

Ensivaikutelma

Sivuston ulkoasu on tärkeässä roolissa ensivaikutelman luomisessa, sillä WWW-sivulle saavuttaessa tekee käyttäjä päätöksen sivustolle jäämisestä kuuden ja kahdeksan sekunnin välisenä aikana. Ulkoasun kaksi tehtävää ovatkin viestittää käyttäjälle sivuston tunnelma sekä sen tarkoitus, ja jos nämä vastaavat käyttäjän odotuksiin, sivustolle jäädään. (Plumley 2010, 52)

Kokematon internetin käyttäjä viettää sivuston etusivulla aikaa keskimäärin 35 sekuntia, ja kokenut käyttäjä 25 sekuntia. Käytettävissä on siis noin puoli minuuttia aikaa tehdä käyttäjään vaikutus, ja tästä puolesta minuutistakin suurin osa kuluu siihen, että käyttäjä selvittää, minne hän haluaa mennä seuraavaksi. (Nielsen & Loranger 2006, 30)

Mikäli sivusto selviää ensimmäiset kymmenen sekuntia, selaavat käyttäjät sitä, mutta ovat kuitenkin valmiita lähtemään seuraavien 20 sekunnin kuluttua. Jos käyttäjä viettää sivustolla 30 sekuntia, kasvavat mahdollisuudet siihen, että he ovat siellä myös pidempään, usein jopa muutamien minuuttien ajan. (Nielsen 2011)

Sivuston asettelu

Sivuston asettelu voidaan jakaa kahteen osaan: sisältöalueiden asetteluun sekä sisällön asettelemiseen näihin alueisiin. Asettelulla pyritään pitämään käyttäjä keskittyneenä sivuston olennaiseen sisältöön. Koska ensivaikutuksen tekemiseen ei ole paljoa aikaa, on parempi keskittyä esimerkiksi helppokäyttöiseen navigointiin sekä tunteita herättäviin kuviin, kuin uudenlaisen rakenteen luomiseen. Sivuston asettelun sekä elementtien sijoittelun tulisikin pysyä samanlaisena läpi koko sivuston (Plumley 2010, 55–57)

Sivuston päänavigaation tulisi erottua ulkoasusta, muttei kuitenkaan häiritsevästi. Kuvien käyttöä navigaatiossa ei suositella, vaan kannattaa käyttää värejä, jotka sopivat sivuston ulkoasuun. Navigaation linkkien tulee olla helppolukuisia, jonka varmistaa selkeä ja suurikokoinen kirjasintyyppi sekä hyvä kontrastiero. Käyttäjät myös odottavat, että sivun alatunnisteesta (*footer*) löytyy linkkejä. Nämä linkit keskittyvät yleensä muun muassa yhteystietoihin sekä rekisteriselosteisiin, mutta pienemmillä sivustoilla voidaan alatunnisteeseen kuitenkin laittaa kaikkien alasivujen linkit. (Plumley 2010, 89–91)

Valkoinen tila

Valkoisella tilalla (*white space*) viitataan ulkoasussa alueisiin, joissa ei ole sisältöä. Valkoisen tilan avulla voidaan helpottaa käyttäjää havainnoimaan sivuilla olevaa sisältöä antamalla sisältöalueille tilaa. Jos WWW-sivulla ei ole ollenkaan valkoista tilaa, se saattaa tuntua ahtaalta ja sivulta voi olla vaikea löytää etsimäänsä. Valkoisen tilan lisääminen sivulle saattaa kuitenkin johtaa siihen, että joitain elementtejä tulee korostaa vähemmän tai jopa poistaa kokonaan. (Nielsen & Loranger 2006, 347–348)

Responsiivisuus

Responsiivisuus WWW-sivujen ulkoasussa tarkoittaa sitä, että sivustolle voidaan tehdä yksi ulkoasu, joka mukautuu näyttämään hyvältä käytettävästä laitteesta, kuten tietokoneesta tai älypuhelimesta riippuen. Tällöin sivustosta ei tarvitse tehdä erikseen mobiililaitteille sopivaa versiota. (Frain 2012, 1)

On ennustettu, että tabletit ylittävät tietokoneiden myynnin vuonna 2014 (Canalys 2013). Voidaan siis olettaa, että sivuston polkuja tullaan käyttämään myös tablet-laitteilla jo pelkästään mukavuussyistä, koska ne eivät sido käyttäjää yhteen paikkaan. Tällöin on siis otettava huomioon se, että sivusto tulee suunnitella responsiiviseksi.

2.3 Luonnosten tekeminen

Erinomainen tapa luoda ja jakaa ideoita nopeasti, on tehdä niistä luonnoksia. Luonnosten avulla voidaan testata erilaisia suunnitelmia ilman, että tarvitsee käyttää kohtuuttomasti aikaa yksityiskohtiin. Luonnosten tekemistä varten tarvitsee vain kynän ja paperin. (Chudley & Allen 2012, 269, 274)

Luonnosten yksinkertaisuus voi olla erittäin hyödyllistä, vaikka ne eivät aluksi siltä vaikuttaisikaan. Yksinkertaisuus luonnoksissa tuo esille suunnitelman idean ilman, että tartutaan epäolennaisiin yksityiskohtiin. (Chudley & Allen 2012, 271)

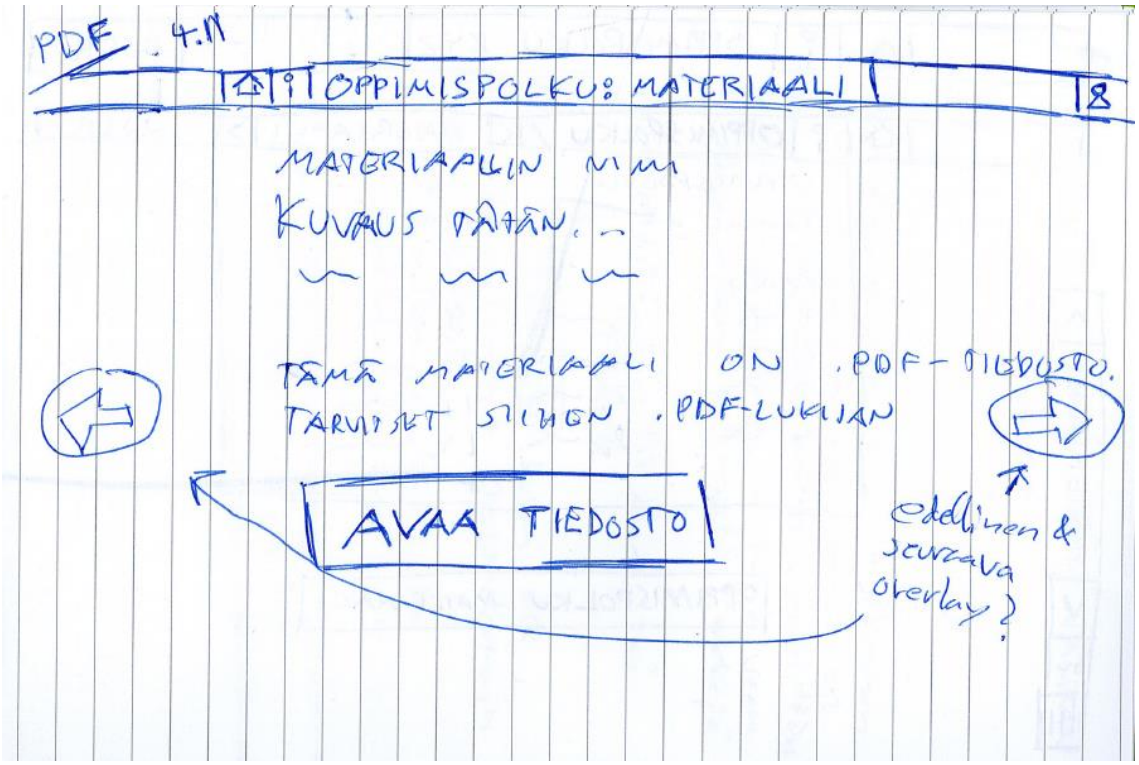
Opinnäytetyön toteutusosassa tehdyn sivuston suunnittelun tavoitteena on ollut tehdä siitä mahdollisimman yksinkertainen käyttää, koska sillä on selkeät käyttötarpeet. Yksinkertaisuus on merkittävä tekijä siksi, että käyttäjillä olisi mahdollisimman pieni kynnys lähteä käyttämään sivustoa.

2.3.1 Polun esittäminen

Kuvassa 1 on polun esittämiseen tehty suunnitelma. Käyttöliittymän yläosa sisältää vasemmalta katsottuna ensimmäiseksi painikkeen sivuston etusivulle, jota kuvaa talo. Tämän jälkeen tulee lisätietoa kuvaava i-painike, jonka tarkoituksena on sisältää polun sekä materiaalin kuvaus. Jotta käyttäjä näkee vaivattomasti, missä polussa sekä materi-

aalissa hän on, on näiden nimet keskitettynä yläosassa. Oikeaan reunaan on varattu painike kirjautuneelle käyttäjälle, jota kuvastaa ihmisen kuva.

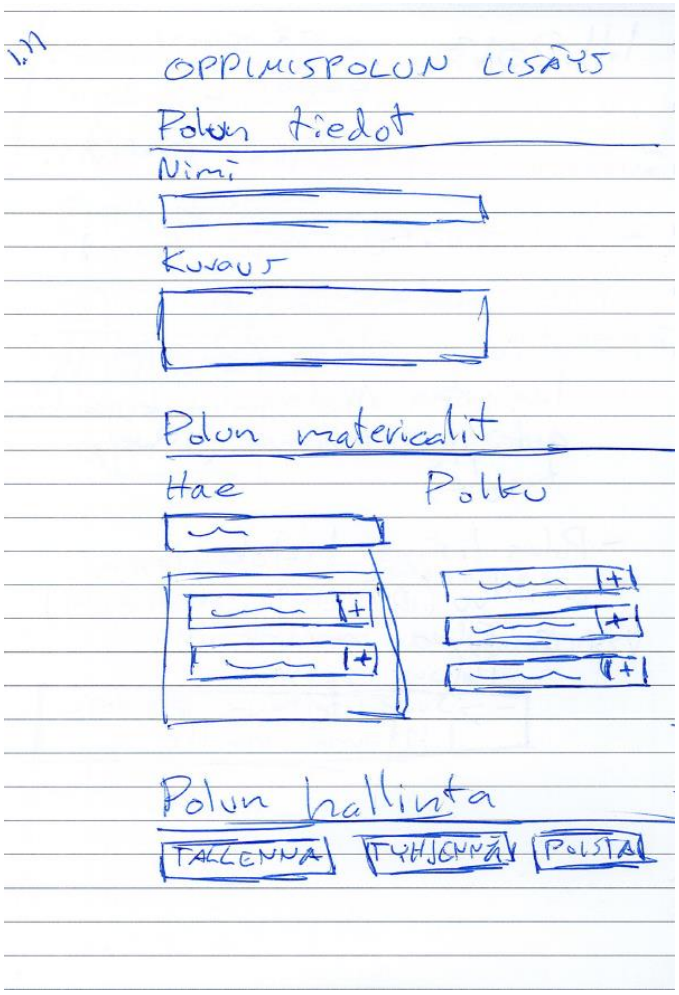
Polussa liikutaan kuvassa näkyvien nuolten avulla. Nuolet on tasattu pystysuunnassa keskelle sivustoa. Näiden nuolten avulla päästään edelliseen tai seuraavaan materiaaliin polussa. Materiaali ladataan luettavaksi käyttöliittymän yläosan alapuolelle.



Kuva 1 Polun esittämisen suunnitelma

2.3.2 Polun luominen

Kuvassa 2 on polun lisäämistä varten tehty suunnitelma. Tarkoituksena on jakaa sivu muutamaan suurempaan alueeseen, josta ensimmäisenä pyydetään polun tiedot ja toisessa luodaan itse polku. Materiaalit haetaan tietokannasta ja ne esitetään haun alapuolella. Haun tuloksia voidaan sitten siirtää polun puolelle ja niitä on mahdollista järjestellä. Polun hallintaan liittyvät painikkeet näytetään lopuksi.



Kuva 2 Polun luomisen suunnitelma

3 KÄYTETYT TEKNIIKAT

3.1 HTML

HTML (the Hypertext Markup Language) on internetsivujen perusta. Se on merkkauskieli, jonka avulla kuvataan internetsivujen rakennetta. Sen avulla voidaan luoda internetsivu, joka sisältää muun muassa otsikoita, tekstiä, taulukoita, listoja sekä kuvia. (W3C 2013)

HTML5

HTML-merkkauskielestä on tällä hetkellä kehitteillä uusi, viides versio (HTML5). Syyskuussa 2012 on esitetty, että HTML5 julkaistaan vuoden 2014 loppuun mennessä ja HTML5.1 vuoden 2016 lopussa. (W3C 2012)

Vaikka HTML5 ei ole vielä valmis, voidaan internetsivut kuitenkin tänä päivänä kirjoittaa sillä. Kaikki nykyaikaiset selaimet ymmärtävät HTML5-kielen yleisimpiä uusia ominaisuuksia, kuten sen rakennetta kuvaavia, sekä audio- ja video elementtejä. (Frain 2012, 98)

Rakenne ja media ovatkin kaksi pääkategoriaa, jolla HTML5 eroaa edellisistä versioista. Aikaisemmin sivun rakenne on jaoteltu yleisellä <*div*>-elementillä esimerkiksi yläosaan, sisältöön sekä alaosaan. HTML5 tarjoaa sivun rakenteelle erityiset <*header*>- ja <*footer*>-elementit sekä sisältöä varten <*article*>- sekä <*summary*>-elementit. HTML5 sisältää myös lukuisia muita rakennetta kuvaavia elementtejä muun muassa lomakkeille sekä navigaatiolle. (Lowery & Fletcher 2011, 38–39)

HTML5 sisältää sisäänrakennetun tuen videon ja äänen toistamiseen <*video*>- sekä <*audio*>-elementtien avulla ilman erillisiä selainlaajennoksia. Staattista sekä liikkuvaa vektorigrafiikkaa voidaan esittää <*canvas*>-elementin avulla. (Lowery & Fletcher 2011, 39)

Opinnäytetyön käytännön toteutusosuudessa sivusto on toteutettu HTML-kielellä. Selaimille ilmoitetaan sivuston dokumenttien tyypiksi HTML5, mutta HTML5:n tarjoamia uusia ominaisuuksia ei käytetä ollenkaan.

3.2 CSS

CSS (Cascading Style Sheets) on HTML-merkkauskielen rinnalla toinen perusta internetsivujen luontiin. HTML on vastuussa sivun rakenteen kuvaamisesta, kun taas CSS kuvaa tämän rakenteen esitystavan, kuten sen värit, kirjasinlajit sekä asettelun. (W3C 2013)

CSS3

CSS3 on uusin versio, jonka tarkoituksena on helpottaa HTML-elementtien muotoilua ilman tarvetta käyttää joko JavaScriptiä, kuvia tai Flashia. Tästä huolimatta CSS3-muotoilut eivät näytä samoilta kaikilla selaimilla. (Avola & Raasch 2012, 46)

CSS2 vaati yhdeksän vuotta päästäkseen suositelluksi standardiksi, koska muutamat toissijaiset ominaisuudet viivästyttivät sitä. Tästä syystä CSS3 on jaettu pienempiin osiin, moduuleihin, jotka standardisoidaan itsenäisesti. CSS3 tarjoaakin muun muassa pyöristettyjä reunuksia, varjostuksia, liukuvärejä, siirtymiä sekä animaatioita. (Mozilla Developer Network 2013)

Opinnäytetyön sivuston HTML-elementtien muotoilu on toteutettu käyttäen CSS-määrittelyjä. CSS3-ominaisuuksia ei koettu tarpeellisiksi suunniteltua ulkoasua varten.

3.3 PHP

PHP (PHP: Hypertext Preprocessor) on erityisesti internetsovellusten kehittämiseen soveltuva avointa lähdekoodia oleva ohjelmointikieli. PHP eroaa asiakaspuolen ohjelmointikielistä siinä, että sen koodi ajetaan ilman käännettävää konekielistä versiota palvelimen puolella. Tämä mahdollistaa sen, että taustalla ajettavaa koodia ei voida selvittää. (The PHP Group 2013)

Internetsivuista 81,5 % käyttää palvelinpuolen ohjelmointikielenään PHP:ta. Näihin palveluihin lukeutuvat esimerkiksi Facebook, Wikipedia, Twitter sekä WordPress. (W3Techs 2013)

Opinnäytetyön sivustolla PHP on vastuussa kaikista kutsuista tietokantaan. Kun esimerkiksi polkua haetaan, ottaa PHP yhteyden tietokantaan, tekee haun ja palauttaa tulokset.

3.4 MySQL

MySQL on suosittu ja helppokäyttöinen tietokannan hallintajärjestelmä. Se on toteutettu avoimella lähdekoodilla ja on erittäin nopea, luotettava sekä skaalautuva. (MySQL 2013)

MySQL tietokannan tuettuja käyttöjärjestelmiä on Unix, Linux, Windows, OS/2, Solaris sekä MacOS. MySQL hallitsee suuriakin tietokantoja ja sitä käyttääkin muun muassa Yahoo!, Google, Cisco sekä HP. MySQL vastaa tietokantakyselyihin jopa nopeammin kuin monet kaupalliset relaatiotietokantajärjestelmät, ja se on helppo ja nopea ottaa käyttöön muutamien minuuttien sisällä. MySQL tietokannassa on myös tuki useille eri ohjelmointikielille, kuten C, C++, Perl, PHP, Java sekä Python. (Sheldon & Moes 2005, 8)

MySQL oli luonnollinen valinta opinnäytetyön toteutusosan sivuston tietokannaksi, koska se oli valmiiksi asennettuna kehitysympäristöön. Kaikki polkujen sekä materiaalien tiedot sijaitsevat MySQL-tietokannassa, josta ne haetaan PHP:n avulla.

3.5 jQuery

jQuery on JavaScript-kirjasto, joka on luotu helpottamaan JavaScriptin ominaisuuksien käyttöä. Se ei tarjoa uusia ominaisuuksia, vaan muuttaa vaikeasti käytettävät JavaScript API:t (application programming interfaces) helpommin kirjoitettavaan muotoon. Tämä auttaa WWW-kehittäjiä hyödyntämään JavaScriptia tehokkaammin. (Rutter 2010, 5)

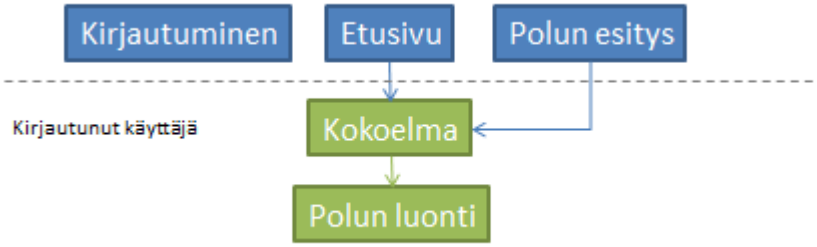
jQueryn yksi vahvuuksista onkin se, että sillä kirjoitettu koodi toimii selainriippumattomasti. Tällöin eri selaimille ei tarvitse kirjoittaa omaa JavaScript-koodia. jQueryllä on aktiivinen käyttäjäyhteisö, ja uusia virallisia sekä kolmannen osapuolen laajennuksia tuleekin päivittäin lisää. (Otero & Larsen 2012, 25–26)

jQueryä käytetään opinnäytetyön käytännön toteutusosan sivustolla käytännössä kaikki-alla, missä se vain on mahdollista. Se tuo sivustolle dynaamisuuden, joka muuten ei onnistuisi, ja on vastuussa muun muassa Ajax-kutsuista tietokantaan sekä materiaalin vaihtamisesta polussa.

4 SIVUSTON TOTEUTUS

4.1 Rakenne

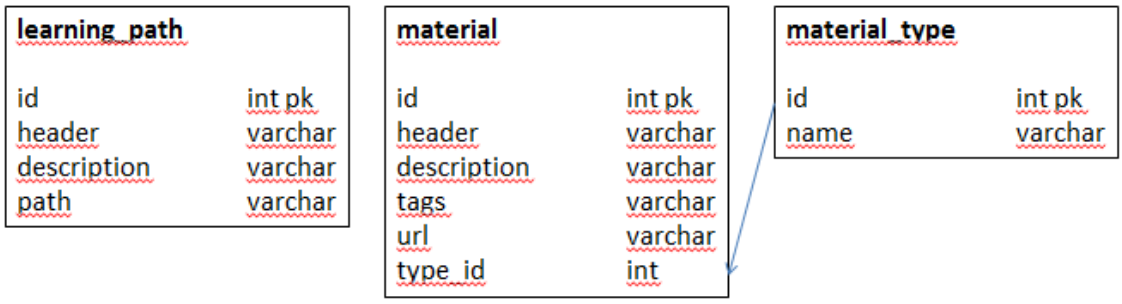
Sivuston rakenne jakautui kuvassa 3 oleviin kahteen selkeään osioon, julkiseen ja yksityiseen puoleen. Julkinen puoli palvelee kaikkia sivustolle saapuvia käyttäjiä. Heillä on pääsy etusivulle ja polkuihin. Yksityinen puoli on tarkoitettu ylläpitoon kuuluville henkilöille, joille annetaan pääsy kokoelmaan. Kokoelmassa sijaitsee sivustolle luodut polut ja mahdollisuus niiden hallintaan, kuten luomiseen, muokkaamiseen ja poistamiseen.



Kuva 3 Sivuston rakenne

4.1.1 Tietokanta

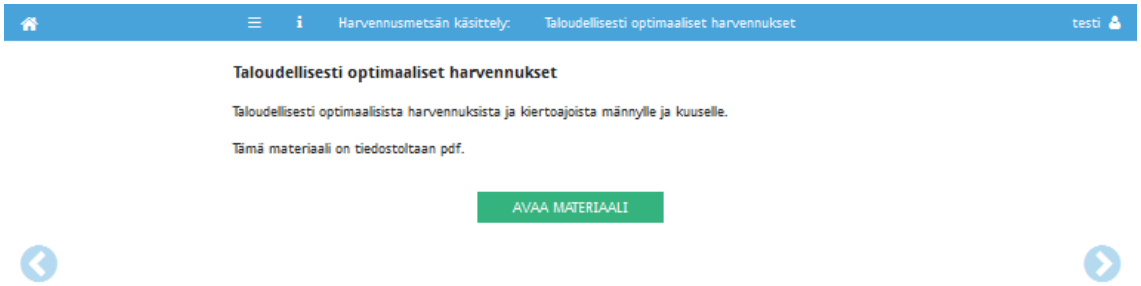
Kuva 4 havainnollistaa tietokantaa, joka koostuu kolmesta taulusta: learning_path (oppimispolku), material (materiaali) sekä material_type (materiaalin tyyppi). Oppimispolku sisältää polkujen nimen, kuvauksen sekä polkuun kuuluvien materiaalien id:t pilkulla eroteltuina. Materiaali sisältää nimen, kuvauksen, avainsanat, URL-osoitteen materiaaliin sekä materiaalin tyypin id:n. Tämä id vastaa tiettyä tekstijonoa materiaalin tyyppitaulussa.



Kuva 4 Tietokannan rakenne

4.2 Polun esittäminen

Polun visuaalinen esittäminen oli ensimmäinen kahdesta toimeksiannon tavoitteesta. Polun hakemisesta ja esittämisestä vastaa `path.php`-niminen ohjelma. Polun näkymä on kuvassa 5.



Kuva 5 Polun käyttöliittymä

4.2.1 Polun hakeminen

Polun hakeminen tapahtuu antamalla ohjelmalle parametri `show` ja haettavan polun id. Annettu parametri voisi siis olla seuraavanlainen: `path.php?show=10`. Tällöin tietokannasta haettaisiin polku, jonka id on kymmenen.

Saavuttaessa ohjelmaan, tehdään ensimmäisenä tarkastus sille, onko parametri `show` määritetty. Mikäli parametria ei ole määritetty, ilmoitetaan sen puuttumisesta käyttäjälle JavaScriptin `alert()`-ponnahdusikkunalla, jonka jälkeen käyttäjän selain ohjataan sivuston etusivulle (`index.php`). Parametri tarkistetaan selaimen internetosoitteesta funktiolla `$.url().param()`, joka on osa Purl (A JavaScript URL parser) JavaScript-kirjastoa. Tämä tarkistus esitetään koodiesimerkissä 1.

```
if (typeof $.url().param("show") === "undefined") {
    alert("Polkua ei löytynyt!");
    window.location = "index.php";
}
```

Koodiesimerkki 1 Parametrin tarkistus

Mikäli koodiesimerkin 1 if-lauseke ei toteudu, tiedetään, että parametri on määritetty ja ohjelma voi jatkaa eteenpäin. Koodiesimerkissä 2 otetaan kyseisen parametrin arvo internetosoitteesta muistiin ja tehdään Ajax-kutsu tietokantaan. Kutsu palauttaa parametrin arvoa vastaavan polun tiedot ja sen materiaalit, ja laittaa ne talteen `results`-muuttujaan.

```
jQuery.extend({
    getPath: function(path) {
    var result = null;
        $.ajax({
            url: 'database/database.php',
            type: 'get',
            data: path,
            dataType: 'json',
            async: false,
            success: function(data) {
                result = data;
            }
        });
    return result;
    }
});
var currentPathId = $.url().param('show');
var results = $.getPath("show=" + currentPathId);
```

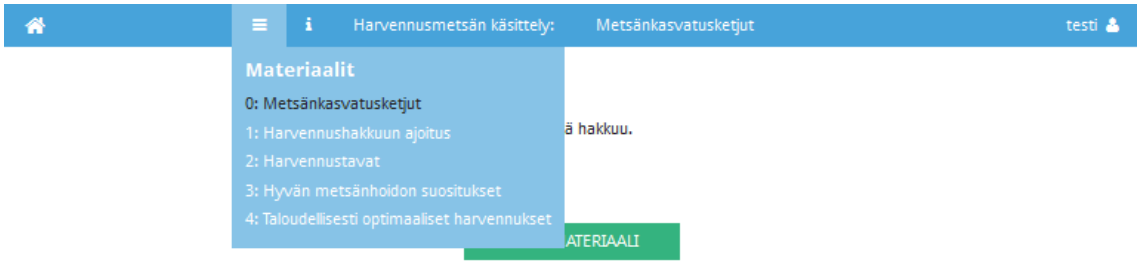
Koodiesimerkki 2 Ajax-kutsu tietokantaan

Tämän jälkeen käydään läpi polun jokainen materiaali erikseen koodiesimerkissä 3 esiteltävällä tavalla ja lisätään ne kuvassa 6 näkyvään listaan. Jokainen materiaali saa myös `step`-ankkurin. joka kertoo materiaalin sijainnin `results`-muuttujan taulukossa ja samalla myös antaa polulle oikean järjestyksen. Tämän ankkurin arvon tehtävänä on

myös samalla kertoa materiaalin askel polussa. Taulukko aloittaa laskemisen nollasta, ja täten myös ensimmäinen materiaali saa askeleekseen kyseisen arvon. Askeleen sisältävä polku voisi olla esimerkiksi `path.php?show=10#step=2`. Tällöin esitettäisiin polun kolmas materiaali.

```
$(document).ready(function() {
    $.each(results.pathMaterial,function(path_num, obj) {
        var addLink = '<li id="' + path_num + '"><a href="#step=' +
                      path_num + '">' + path_num + ': ' + obj.header
                      + '</a></li>';
        $(".show-path .menu-list").append($(addLink));
    });
});
```

Koodiesimerkki 3 Materiaalien lisäys listaan



Kuva 6 Lista polun materiaaleista

Kun ohjelmalle on määritetty `show`-parametri, suoritetaan myös koodiesimerkissä 4 esitettävä tarkistus sille, onko polulle annettu askel. Muuttujalle `currentStep` annetaan arvoksi nolla, jos `step`-ankkuria ei ole määritetty internetosoitteeseen. Tällöin osataan ladata polun ensimmäinen materiaali. Mikäli askel on määritetty, tiedetään, että käyttäjä haluaa päästä polun tiettyyn materiaaliin käsiksi ja asetetaan ankkurin arvo muuttujalle. Lopuksi annetaan materiaalin lataamisesta vastuussa olevalle `loadMaterial()`-funktiolle parametriksi muuttuja `currentStep`.

```
if (typeof $.url().fparam("step") === "undefined") {
    var currentStep = 0;
} else {
    var currentStep = $.url().fparam("step")
}
loadMaterial(currentStep);
```

Koodiesimerkki 4 Askeleen tarkistus internetosoitteesta

4.2.2 Same-origin policy

Same-origin policy on yksi tärkeimmistä selainten turvallisuuteen vaikuttavista ominaisuuksista. Sen tarkoituksena on estää verkkosivua lataamasta dokumenttia internetistä, mikäli se ei sijaitse saman verkkotunnuksen alla. Tällä estetään muun muassa haitallisten JavaScript-koodien ajaminen. (Zalewski 2011)

Tarkoituksenani oli alun perin suorittaa materiaalien sisällön lataaminen sivulle Ajax-kutsun avulla. Ajax-kutsu, jota suunnittelin käytettäväksi, on kuitenkin Same-origin policyn alla, joten materiaalien haku ei sillä onnistunut. Same-origin policy on mahdollista kiertää, mutta turvallisuussyistä jätin sen tekemättä ja siirryin käyttämään `iframe`-elementtiä materiaalin esittämistä varten.

Käytännössä tämä tarkoitti sitä, että jouduin luopumaan materiaalin loppuun sijoitettavasta osiosta, johon olisi tuotu avoinna olevaan materiaaliin liittyviä muita materiaaleja. Sivun käyttöliittymää lukuun ottamatta kaikki muu vapaa tila selainikkunasta varattaisiin täysin materiaalin esitystä varten.

4.2.3 Materiaalin lataaminen

Materiaali ladataan, kun kutsutaan funktiota `loadMaterial()`. Tälle funktiolle annetaan parametriksi muuttujan `currentStep` arvo, joka pitää sisällään nykyisen askeleen. Kun funktiota on kutsuttu, suoritetaan koodiesimerkissä 5 oleva koodi.

```
var loadMaterial = function(currentStep) {
    var currentMaterial = results.pathMaterial[currentStep].url;
    if (results.pathMaterial[currentStep].type === "html") {
        $("iframe").attr("src", currentMaterial);
    } else {
        var materialType = results.pathMaterial[currentStep].type;
        $("iframe").attr("src", "template/template.html");
        $("iframe").on("load", function() {
            loadInfo(currentStep);
        });
    }
    hideShows();
}
```

Koodiesimerkki 5 Materiaalin lataaminen

Muuttujaan `currentMaterial` asetetaan nykyisen askeleen materiaalin internetosoite. Tämän jälkeen tehdään tarkistus if-lausekkeella sille, onko materiaalin tyyppi `html` vai jokin muu. Mikäli tyyppi on `html`, tiedetään, että kyseinen materiaali voidaan ladata suoraan iframeen näytettäväksi. Tällöin iframen src-attribuutin arvoksi asetetaan muuttujan `currentMaterial` sisältämä internetosoite materiaalin sijainnista.

Jos materiaali ei ole tyypiltään `html`, oletetaan, että kyseessä on sellainen materiaali, jota ei voida jostain syystä tuoda suoraan iframeen esitettäväksi. Silloin iframeen ladataan `template.html`-niminen tiedosto. Tämä tiedosto on mallipohja, joka sisältää yksinkertaiset raamit materiaalin otsikolle, kuvaukselle sekä painikkeelle, josta materiaali avautuu uuteen ikkunaan. Koodiesimerkissä 6 ajetaan funktio `loadInfo()`, joka saa parametrikseen muuttujan `currentStep` arvon ja mallipohja täytetään materiaalin tiedoilla.

```
var loadInfo = function(currentStep) {
    $("iframe").contents().find("div.material-info
      h3").text(results.pathMaterial[currentStep].header);
    $("iframe").contents().find("div.material-info
      p").text(results.pathMaterial[currentStep].description);
    $("iframe").contents().find("div.material-open a").attr("href",
      results.pathMaterial[currentStep].url);
}
```

Koodiesimerkki 6 Materiaalin tiedot mallipohjaan

Materiaalin lataaminen suoritetaan aina, kun polussa liikutaan. Funktion `hideShows()` toiminta esitetään koodiesimerkissä 7. Se piilottaa kaikki mahdollisesti avoinna olevat valikot, kuten listan polun materiaaleista, info-ikkunan sekä kirjautuneen käyttäjän valikon poistamalla `show`- ja `menu-li-active`-luokat niiden elementeistä.

```
var hideShows = function() {
    $(".show-path").removeClass("show");
    $("li.material").removeClass("menu-li-active");
    $(".show-info").removeClass("show");
    $("li.info").removeClass("menu-li-active");
    $(".show-login").removeClass("show");
    $("li.user").removeClass("menu-li-active");
}
```

Koodiesimerkki 7 Ylimääräisten ikkunoiden piilotus

4.2.4 Käyttöliittymän päivittäminen

Iframeen on kiinnitetty JavaScript-tapahtuma `onload()`, joka ajaa funktion `generateNavigation()`, kun materiaalin sisältö on ladattu. Tälle funktiolle annetaan patametriksi nykyinen askel. Jos käyttäjä on saapunut polulle internetosoitteen kautta, missä on määritettynä `step`-ankkuri, tulee muuttuja `currentStep`-merkkijonona. Laskutoimituksia varten se muunnetaan kokonaisluvuksi `parseInt()` funktiolla, joka esitetään koodiesimerkissä 8.

```
var generateNavigation = function(currentStep) {
    var currentStep = parseInt(currentStep, 10);
    updateArrows(currentStep);
    updateTexts(currentStep);
    activePathLink(currentStep);
};
```

Koodiesimerkki 8 Käyttöliittymän päivitys

Koodiesimerkissä 8 oleva `generateNavigation()` kutsuu vielä kolme funktiota, joiden jälkeen polku on ladattu ja valmis käytettäväksi. Ensimmäinen funktioista on `updateArrows()`, joka päivittää sovelluksessa esiintyvät nuolet, joilla siirrytään edelliseen ja seuraavaan materiaaliin polussa. Tämän funktion laskutoimituksissa tulee ottaa huomioon se, että polun materiaalien laskeminen alkaa nollasta.

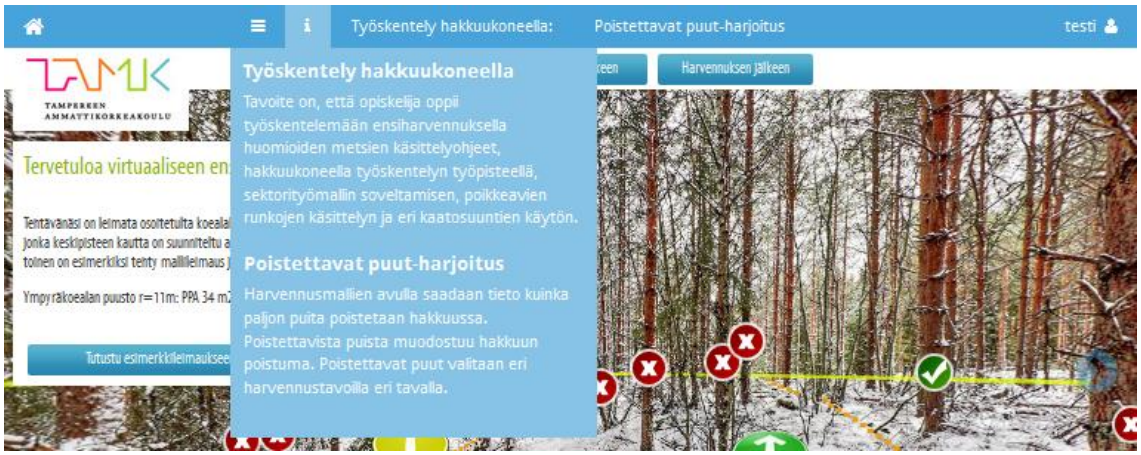
Koodiesimerkissä 9 oleva funktio saa parametrikseen muuttujan `currentStep`. Se sisältää kaksi if-lauseketta, kummallekin nuolelle omansa. Ensimmäisessä if-lausekkeessa tehdään tarkistus sille, mikä on muuttujan `currentStep` arvo, kun siitä vähennetään yksi. Jos tulos on miinus yksi, tiedetään, että ollaan polun ensimmäisessä materiaalissa ja nuoli voidaan piilottaa pois näkyviltä. Muussa tapauksessa tiedetään, että voidaan antaa edellinen-nuolelle nykyistä materiaalia edeltävä askel ja asettaa nuoli näkyville poistamalla elementin piilottava hide-tyyliluokka. Toisessa if-lausekkeessa tehdään vastaavanlainen tarkistus seuraava-nuolelle, mutta tällä kertaa vertaillaan sitä, kun `currentStep` muuttujaan lisätään yksi, että onko tämän laskutoimituksen tulos yhtä suuri kuin polun materiaalien yhteenlaskettu tulos. Jos tulos on yhtä suuri, ollaan viimeisessä materiaalissa ja nuoli piilotetaan näkyvistä, muussa tapauksessa voidaan antaa seuraava-nuolelle polun seuraavan materiaalin askel.

```
var updateArrows = function(currentStep) {
    if ((currentStep - 1) != -1) {
        $("a.previous").attr("href", "#step=" + (currentStep - 1));
        $(".prev-middle").removeClass("hide");
    } else {
        $(".prev-middle").addClass("hide");
    }

    if ((currentStep + 1) === results.pathMaterial.length) {
        $(".next-middle").addClass("hide");
    } else {
        $("a.next").attr("href", "#step=" + (currentStep + 1));
        $(".next-middle").removeClass("hide");
    }
}
```

Koodiesimerkki 9 Nuolten päivittäminen

Kuvassa 7 näkyvän polun käyttöliittymän tekstit ja linkit päivitetään `updateTexts()`-funktion avulla. Koodiesimerkissä 10 näkyy, kuinka polun ja materiaalin nimi sekä niiden linkki päivitetään käyttöliittymään. Info-ikkunaan päivitetään polun ja materiaalin nimen lisäksi myös niiden kuvaukset.



Kuva 7 Polun info

```
var updateTexts = function(currentStep) {
    $("a.path").attr("href", "path.php?show=" + re
      sults.currentPath.id);
    $("a.path").text(results.currentPath.header + ":");

    $("a.current").attr("href", "#step=" + currentStep);
    $("a.current").text(results.pathMaterial[currentStep].header);

    $(".nav-top-center-secondary #insert-path-
      name").text(results.currentPath.header);
    $(".nav-top-center-secondary #insert-path-
      description").text(results.currentPath.description);
```

```
    $(".nav-top-center-secondary #insert-material-
      name").text(results.pathMaterial[currentStep].header);
    $(".nav-top-center-secondary #insert-material-
      description").text(results.pathMaterial[currentStep].desc
      ription);
}
```

Koodiesimerkki 10 Käyttöliittymän tekstien päivitys

Käyttöliittymän päivityksen viimeinen funktio on `activePathLink()` jonka sisältö näkyy koodiesimerkissä 11, ja joka käy läpi listan polun materiaaleista. If-lauseke tehdään jokaiselle listan `<li>`-elementille, jossa tehdään vertailu siitä, onko kyseisen elementin id sama kuin muuttujan `currentStep` arvo. Tämä vertailu toteutuu yhden materiaalin kohdalla ja sen `<a>`-elementti (linkki) saa tyyliluokan `menu-list-active`, joka vaihtaa linkin värin valkoisesta mustaksi, jolloin se erottuu listasta aktiivisena. Kaikista muista listan materiaaleista poistetaan kyseinen tyyliluokka.

```
var activePathLink = function(currentStep) {
    $(".menu-list li").each(function() {
        if ($(this).attr("id") != currentStep) {
            $(this).children("a").removeClass("menu-list-active");
        } else {
            $(this).children("a").addClass("menu-list-active");
        }
    });
}
```

Koodiesimerkki 11 Aktiivinen materiaali

4.2.5 Polussa liikkuminen

Polussa liikkuminen edelliseen ja seuraavaan materiaalin tapahtuu käyttämällä joko polulta löytyviä nuolia tai listan kautta, josta löytyy kaikki polun materiaalit. Kun nuolta tai listassa olevaa materiaalia klikataan, suoritetaan koodiesimerkissä 12 esiintyvä funktio, joka ottaa painetun linkin `step`-ankkurin arvon muistiin `currentStep`-muuttujaan. Tämä arvo muutetaan merkkijonosta kokonaisluvuksi ja kutsutaan funktiota `loadMaterial()` tällä muuttujalla.

```
$(document).ready(function() {
    $(".link").on("click", "a", function(obj) {
        var str = $(this).attr("href");
        currentStep = parseInt(str.split('=')[1], 10);
```

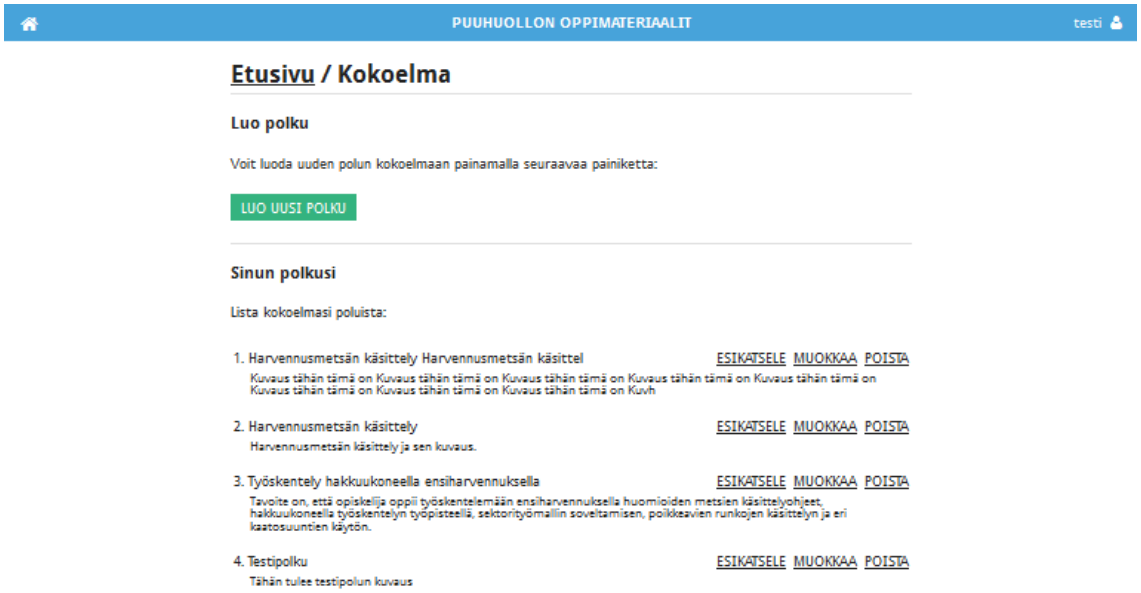
```
        loadMaterial(currentStep);
    });
    $(".menu-list").on("click", "a", function(obj) {
        var str = $(this).attr("href");
        currentStep = parseInt(str.split('=')[1], 10);
        loadMaterial(currentStep);
    });
});
```

Koodiesimerkki 12 Polussa liikkuminen

Koodiesimerkissä 12 esitetty funktio suorittaa tapahtumaketjun, joka on kuvattu alaluvuissa 4.2.3 sekä 4.2.4. Alaluvussa 4.2.1 esitettyä polun hakua ei tarvitse enää tehdä, koska polussa liikuttaessa vain `step`-ankkurin arvoa muutetaan ja materiaali valitaan dynaamisesti jo olemassa olevasta `results`-muuttujasta, joka sisältää kaiken polkuun liittyvän tiedon. Tämä vähentää tietokantahaun vain yhteen kertaan, kun polku ladataan ensimmäisen kerran, mikä nopeuttaa sivuston latautumista sekä polussa liikkumista.

4.3 Kokoelma

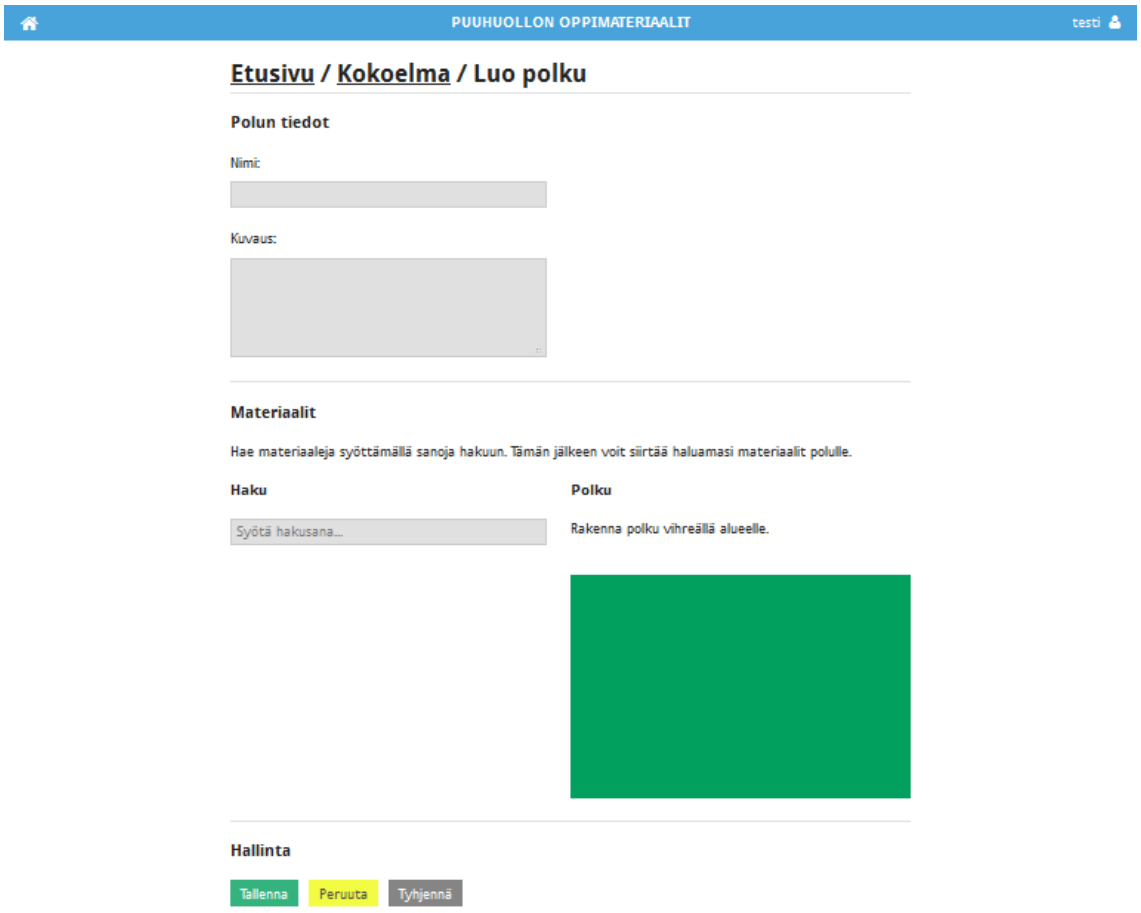
Kokoelma näkyy kuvassa 8, ja se sisältää listan käyttäjän luomista poluista. Kokoelman kautta päästään hallinnoimaan polkuja. Sen kautta voidaan luoda täysin uusi polku tai muokata jotain olemassa olevaa polkua. Polkuja voi myös esikatsella sekä poistaa tämän näkymän kautta.



Kuva 8 Kokoelma

4.4 Polun luominen

Polkujen luominen oli toinen toimeksiannon tavoite. Kuvassa 9 oleva polun luontia koskeva sivu on jaettu kahteen osaan sekä polun hallintaan liittyviin painikkeisiin. Polun luonnista vastaa `create_path.php`-ohjelma.



Kuva 9 Polun luonti

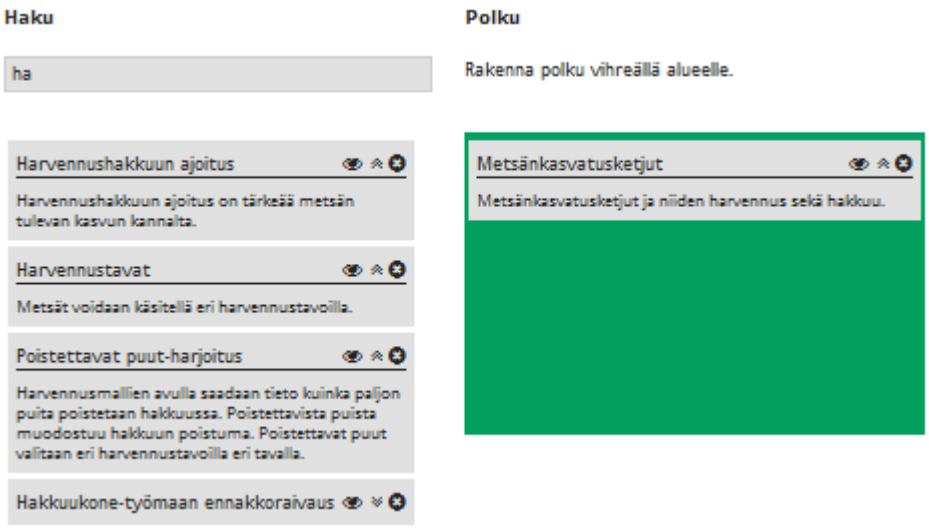
Kuvassa 9 esiintyvä ensimmäinen alue sisältää tekstikentät polun nimeä sekä kuvausta varten. Toinen osa keskittyy polun sisältämiin materiaaleihin ja se on jaettu kahteen osaan, materiaalien hakuun ja itse polkuun.

Materiaalien haku tapahtuu koodiesimerkin 13 Ajax-kutsun kautta välittömästi, kun hakukenttään syötetään tai poistetaan merkkejä. Hakulaatikkoon on kiinnitetty jQueryn `keyup()`-tapahtuma, joka toteutuu silloin, kun käyttäjä vapauttaa näppäimen painalluksen näppäimistöltä. If-lausekkeella tarkistetaan, onko hakulaatikossa merkkejä. Mikäli merkkejä on, tehdään Ajax-kutsu tietokantaan, jolla haetaan materiaalit joiden nimi,

kuvaus tai avainsanat sisältävät hakuun syötettyjä merkkejä. Kutsu palauttaa kuvassa 10 esiintyvällä tavalla materiaalit laatikkoina. Ne sisältävät materiaalin nimen, kuvauksen sekä pikanäppäimet materiaalin esikatseluun, kuvauksen piilottamiseen sekä materiaalin poistamiseen hakutuloksista.

```
$(document).ready(function() {
    $("#keywords").keyup(function() {
        var kw = $("#keywords").val();
        if (kw != '') {
            $.ajax ({
                type: "post",
                url: "database/database.php",
                data: "kw=" + kw,
                success: function(option) {
                    $("#search").html(option);
                }
            });
        }
        return false;
    });
});
```

Koodiesimerkki 13 Haun Ajax-kutsu



Kuva 10 Hakutulos

Polku tehdään kuvassa 10 näkyvälle vihreälle alueelle. Materiaalien siirtäminen polulle tapahtuu vedä ja pudota -tekniikalla ottamalla hakutulosten materiaalista kiinni ja vetämällä se vihreälle alueelle. Kun hiiren osoittimen siirtää materiaalin kohdalle, muuttuu se nelipäiseksi nuoleksi, mikä viestii käyttäjälle, että kyseessä on siirrettävissä oleva elementti. Materiaalien järjestystä voidaan polussa muuttaa samalla tavalla ottamalla

materiaalista kiinni ja siirtämällä se haluttuun kohtaan. Kun materiaalia liikutetaan, kertoo näyttöön ilmestyvä keltainen alue, mihin materiaali siirtyy, kun siitä päästää irti. Materiaalin voi poistaa polusta klikkaamalla siinä olevaa `x`-painiketta.

Haussa ja polussa on listat materiaaleja varten. Nämä listat on identifioitu seuraavanlaisesti: haun lista on `#search` ja polun lista `#createPath`. Koodiesimerkissä 13 esiintyvä Ajax-kutsu palauttaa materiaalit hakutulosten listaan. Haun ja polun listat on yhdistetty jQuery UI-kirjaston `Sortable`-laajennoksen avulla. Tämän laajennoksen avulla voidaan antaa listalle ominaisuus, jolla siinä olevia asioita voidaan järjestellä.

Koodiesimerkissä 14 esiintyvän `sortable()`-funktion avulla yhdistetään haun ja polun listat siten, että niissä olevia asioita voidaan siirtää myös listojen välillä. Tälle funktiolle annetaan asetus `connectWith`, jossa kerrotaan, mitkä listat ovat yhdistettyjä. Jotta haun tuloksia voidaan siirtää vain polulle (eikä polulta takaisin hakutuloksiin), on tälle asetukselle määritetty vain polun `#createPath` lista, jolloin siirtäminen onnistuu yhteen suuntaan.

```
$(function() {
    $( "#search, #createPath" ).sortable({
        connectWith: "#createPath",
        placeholder: "placeholder",
        forcePlaceholderSize: true
    }).disableSelection();
});
```

Koodiesimerkki 14 Listojen yhdistäminen

4.4.1 Polun muokkaaminen

Ohjelman kautta päästään muokkaamaan olemassa olevaa polkua antamalla sille parametriksi `edit` ja muokattavan polun id. Esimerkki ohjelman osoitteesta: `create_path.php?edit=10`. Tällöin muokataan polkua, jonka id on kymmenen.

Ajettaessa ohjelmaa, luodaan muuttuja `pathEdit`, joka pitää yllä tiedon siitä, ollaanko luomassa uutta polkua, vai muokkaamassa jotain jo olemassa olevaa polkua. Oletusarvoisesti tämä muuttuja on nolla, jolloin kyseessä on uusi polku. Muuttujan luonnin jälkeen tehdään tarkistus sille, onko `edit`-parametri määritetty. Mikäli parametri löytyy,

ajetaan koodiesimerkissä 15 oleva funktio. Koska kyseessä on muokattava polku, saa `pathEdit` arvokseen yksi. Tämän jälkeen polun tiedot haetaan tietokannasta ja polun tiedot täyttävät ohjelman tekstikentät sekä asettaa polun materiaalit sille varattuun paikkaan.

```
var pathEdit = 0;

if (typeof $.url().param("edit") != "undefined") {

    var currentPathId = $.url().param('edit');

    pathEdit = 1;

    var results = $.getPath("show=" + currentPathId);

    $("#create-path-name").val(results.currentPath.header);
    $("#create-path-description").val(results.currentPath.desc
      ription);

    $(".content-header h1").html('<a href="index.php">Etusivu</a> / <a
      href="collection.php">Kokoelma</a> / Muokkaa polkua');

    $.each(results.pathMaterial,function(path_num, obj) {
        var addLink = '<li id="' + obj.id + '"><div class="item-
wrapper"><div class="item-header">' + obj.header + '<span class="item-
remove"><i class="fa fa-times-circle"></i></span><span class="item-
more-info"><i class="fa fa-angle-double-down"></i></span><span
class="item-preview"><a href="' + obj.url + '" target="_blank"><i
class="fa fa-eye"></i></a></span></div><div class="item-info"><div
class="item-description">' + obj.description +
'</div></div></div></li>';
        $("#createPath").append($(addLink));
    });
}
```

Koodiesimerkki 15 Muokattava polku

4.4.2 Polun hallinta

Tallenna-painike

Painettaessa Tallenna-painiketta, ajetaan koodiesimerkissä 16 näkyvä funktio. Siinä asetetaan vihreälle alueelle luotu polku muuttujaan `createdPathArray` kutsumalla `sortable()`-funktion `toArray`-metodia, joka lisää listan materiaalien id:t taulukkoon. Tietokantaan syöttöä varten luodaan uusi muuttuja `parsedPath` ja muuttujassa `createdPathArray` sijaitseva taulukko käydään läpi ja sen jokainen arvo lisätään muuttujaan `parsedPath` perään pilkulla eroteltuna. Tämän jälkeen otetaan vielä polun nimi ja kuvaus ylös omiin muuttujiin. Tallenna-painiketta käytetään sekä uuden po-

lun tallentamiseen sekä olemassa olevan päivittämiseen. Tämän vuoksi tehdään tarkistus if-lausekkeella sille, kumpi arvo muuttujassa `pathEdit` on, ja ajetaan joko tietokantaan syöttö `$.addPath()` tai päivitys `$.updatePath()`.

```
$(function() {
    $("#save").on("click", function() {
        var createdPathArray = $("#createPath").sortable("toArray");

        var parsedPath = new String();


        $(createdPathArray).each(function(index, value) {
                parsedPath += value + ",";
        });

        var insertPathName = $("#create-path-name").val();
        var insertPathDescription = $("#create-path-
                                    description").val();

        if (pathEdit === 1) {
            var updatePath = new String();

            updatePath += 'updatePath=true';
            updatePath += '&pathId=' + currentPathId;
            updatePath += '&pathName=' + insertPathName;
            updatePath += '&pathDescription=' + insertPathDescription;
            updatePath += '&path=' + parsedPath;

            $.updatePath(updatePath);
        } else {
            var insertPath = new String();

            insertPath += 'createPath=true';
            insertPath += '&pathName=' + insertPathName;
            insertPath += '&pathDescription=' + insertPathDescription;
            insertPath += '&path=' + parsedPath;

            $.addPath(insertPath);
        }
    });
});
```

Koodiesimerkki 16 Tallenna-painikkeen tarkistus

Muut painikkeet

Muut materiaalin hallintaan liittyvät painikkeet ovat Peruuta, Tyhjennä sekä Poista. Peruuta mitätöi polkuun tehdyt muutokset ja palauttaa käyttäjän kokoelmaan. Tyhjennä tyhjentää polun tiedot sekä materiaalit. Poista-painike poistaa polun tietokannasta ja se näkyy käyttäjälle vain, kun kyseessä olemassa oleva polku jota muokataan.

4.5 Testaaminen

Sivuston testaamista on suoritettu koko prosessin ajan. Heti alussa kävi ilmi, että jQuery ei toiminut Internet Explorerilla. Sivuston kehityksessä täytyi kuitenkin päästä eteenpäin, joten tuki Internet Explorerille päätettiin jättää myöhempää varten.

Lopullinen sivusto on todettu toimivaksi seuraavilla selaimilla, joista kaikista on ollut käytössä uusin versio: Mozilla Firefox, Google Chrome, Safari, sekä Opera. Tableteilla sivusto on testattu toimivaksi Safarilla (iOS) sekä Chromella (Android).

5 POHDINTA

Olemme toimeksiantajan kanssa tyytyväisiä sivuston toiminnallisuuteen. Koen, että lopputuotos täytti oman näkemykseni siitä, kuinka oppimispolku voitaisiin luoda ja esittää. Olen erityisesti tyytyväinen siihen, kuinka onnistuin tavoitteessani tehdä kaikesta mahdollisimman yksinkertaista.

Vaikken kaikkea haluamaani saanutkaan tehtyä valmiiksi, toteutui opinnäytetyön tavoitteena ollut internetissä olevien oppimateriaalien esitystavan kehittäminen siltä osin kuin sen oli tarkoitus. Opinnäytetyöstä ei käy ilmi, kuinka oppimispolku kehittyi parempaan suuntaan projektin aikana, mutta lopullinen tuotos on esitetty hyvinkin yksityiskohtaisesti.

Tiesin toimeksiannon hyväksyessäni, että tulisin tarvitsemaan toteutusta varten JavaScript-osaamista. En ollut ennen tätä projektia tehnyt JavaScriptillä käytännössä yhtään mitään, mutta nyt voin sanoa hallitsevani sen. Se on suurin yksittäinen asia, jonka opin opinnäytetyötä tehdessä.

Yllätyksiä sivuston toteuttamisessa ei tullut alakappaleessa 4.2.2 esitetyn Same-origin policyn lisäksi. Sivustoon jäi kuitenkin muutamia asioita, jotka jäivät häiritsemään. Polun esittäminen sekä muokkaaminen tehdään polun id:llä, joka ei kerro yhtään mitään. Ohjelmoijille on tuttua, että taulukko aloittaa laskemisen nollasta, mutta voin vain kuvitella, kuinka tavallinen käyttäjä ihmettelee, miksi ihmeessä nolla kuvastaa ensimmäistä materiaalia. Myös materiaalien haku polun luontia varten saattaa vaatia käyttäjältä laajaa tuntemusta ennalta siitä, mitä materiaalia on tarjolla, jotta löytää haluamansa.

Olisin kuitenkin halunnut vielä toteuttaa muutamia ominaisuuksia, joita en ajan puutteen vuoksi ehtinyt tekemään. Erityisesti materiaalin lisäystä varten olisin tehnyt koodin, joka tutkii WWW-sivun sisältöä ja tarjoaa automaattisesti materiaalille nimen, kuvauksen sekä avainsanoja. Materiaaleille sekä poluille pitäisi myös olla mahdollisuus asettaa kuva, joka helpottaisi materiaalin tai polun sisällön muistelemista.

Jatkokehitystä ajatellen suurin virhe tapahtui kuitenkin heti alussa, kun päätin sivuuttaa kaikki olemassa olevat sisällönhallintajärjestelmät ja luoda sivustoon kaiken alusta asti

itse. Vaikka tavoitteena ei ollutkaan luoda täysimittaista sisällönhallintajärjestelmää, on siitä hyvin vaikea lähteä kehittämään tuotetta. Oppimispolku olisi siis tullut tehdä jonkin olemassa olevan, avoimen lähdekoodin sisällönhallintajärjestelmän päälle ominaisuutena, joka laajentaisi valitun alustan toiminnallisuutta tuomalla siihen mahdollisuuden oppimispolkujen luontiin. Valmiin sisällönhallintajärjestelmän valinta alustaksi olisi tehnyt siitä helpommin jatkokehitettävän, ja se olisi ratkaissut monta asiaa, jotka siihen tulisi muutenkin lisätä, erityisesti käyttäjien hallinnan ja materiaalien luomisen.

En missään nimessä kuitenkaan pidä demoa turhana. Se tarjoaa lähtökohdan idealle oppimispoluista ja siitä, kuinka se voisi toimia. Tätä kannattaakin pitää ensimmäisenä määrittelynä oppimispolulle ja hyödyntää sen koodi, logiikka ja idea tulevissa projekteissa.

LÄHTEET

Avola, G. & Raasch, J. 2012 Smashing Magazine Book Series. Smashing Mobile Web Development. New Jersey: Wiley.

Canalys. 2013. Tablets to make up 50% of PC market in 2014. Luettu 28.11.2013. http://www.canalys.com/newsroom/tablets-make-50-pc-market-2014

Chudley, J & Allen, J. 2012. Smashing UX Design. Foundations for Designing Online User Experiences. New Jersey: Wiley.

Frain, B. 2012. Responsive Web Design with HTML5 and CSS3. Birmingham: Packt Publishing Ltd.

Lowery, J. & Fletcher, M. 2011. HTML5 24-Hour Trainer. New Jersey: Wrox.

MOT. 2011. MOT Kielitoimiston sanakirja. Luettu 4.12.2013. http://mot.kielikone.fi/mot/tamk/netmot.exe

Mozilla Developer Network. 2013. CSS3. Luettu 9.12.2013. https://developer.mozilla.org/en-US/docs/Web/CSS/CSS3

MySQL. 2013. What is MySQL?. Luettu 29.11.2013. http://dev.mysql.com/doc/refman/5.7/en/what-is-mysql.html

Nielsen, J. 2011. How Long do Users Stay on Web Pages? Luettu 4.12.2013. http://www.nngroup.com/articles/how-long-do-users-stay-on-web-pages/

Nielsen, J & Loranger, H. 2006. Prioritizing Web Usability. Berkeley: New Riders.

Otero, C. & Larsen, R. 2012. Professional jQuery. New Jersey: Wrox.

Plumley, G. 2010. Website Design and Development. 100 Questions to Ask Before Building a Website. New Jersey: Wiley.

Rutter, J. 2010. Smashing jQuery. Professional Techniques with Ajax and JQuery. New Jersey: Wiley.

Sheldon, R & Moes, G. 2005. Beginning MySQL. New Jersey: Wiley.

The PHP Group. 2013. What is PHP? Luettu 29.11.2013. http://php.net/manual/en/intro-whatis.php

W3C. 2012. Plan 2014. Luettu 29.11.2013 http://dev.w3.org/html5/decision-policy/html5-2014-plan.html

W3C. 2013. HTML & CSS. Luettu 29.11.2013 http://www.w3.org/standards/webdesign/htmlcss

W3Techs. 2013. Usage statistics and market share of PHP for websites. Luettu 9.12.2013.

http://w3techs.com/technologies/details/pl-php/all/all

Zalewski, M. 2011. Browser Security Handbook, part 2. Luettu 14.1.2014. http://code.google.com/p/browsersec/wiki/Part2