LAPIN AMK
Lapland University of Applied Sciences

HELPPOKÄYTTÖISEN CMS-SOVELLUKSEN SUUNNITTELU JA TOTEUTUS

Vallineva Tuomo

Opinnäytetyö
Tietojenkäsittely ja tietoliikenne (ICT)
Tietojenkäsittelyn koulutusohjelma
Tradenomi (AMK)

2018

LAPIN AMK
Lapland University of Applied Sciences

Opinnäytetyön tiivistelmä

Tietojenkäsittelyn koulutusohjelma
Tietojenkäsittely ja tietoliikenne(ICT)
Tradenomi (AMK)

Tekijä Tuomo Vallineva Vuosi 2018
Ohjaaja(t) Johanna Vuokila
Toimeksiantaja Kotipalvelu
Työn nimi Helppokäyttöisen CMS-sovelluksen suunnittelu ja toteutus
Sivu- ja liitesivumäärä 38 + 3

Tämän opinnäytetyön aiheena oli helppokäyttöisen CMS-sovelluksen suunnittelu ja toteutus. Toimeksiantaja kotipalvelu-alan yritys tarvitsi nettisivujen ylläpitoon helposti käytettävän julkaisujärjestelmän. Yrityksellä ei ole ennestään olemassa kotisivuja, joten liikkeelle lähdettiin puhtaalta pöydältä. Julkaisujärjestelmiä on tarjolla runsaasti, mutta useimmat ovat raskaita ja monimutkaisia käyttää. Toimeksiantaja toivoi helppokäyttöistä julkaisujärjestelmää. Tästä lähtökohdasta lähti idea rakentaa kevyt ja helppokäyttöinen CMS-sovellus. Sovellus toimii kaikilla päätelaitteilla responsiivisesti. Opinnäytetyö esittelee teorian ja käytännön kautta, miten rakennetaan helppokäyttöinen sovellus.

Käsittelen tässä opinnäytetyössä käytettävyyden perusteita, käyttämiäni teknologioita ja työn eri vaiheita. Sovelluksesta rakennettiin ensimmäinen versio, jolle tehtiin käytettävyystesti. Käytettävyystestin tulokset arvioitiin ja näiden perusteella sovellukselle tehtiin jatkokehittämistä. Esittelen työssäni myös työni tuloksia. Lopussa on pohdintaa siitä miten koin työni onnistuneen.

Sovellus toimii PHP-kiellellä ja ulkoasuun on käytetty HTML-kieltä ja Bootstrapia, joka on avoimen lähdekoodin front-end kirjasto. Sovelluksessa käsitelty tieto tallennetaan MySQL-relaatiotietokantaan.

Avainsanat Web-sovellus, PHP, MySQL, Bootstrap, käytettävyys
Muita tietoja CMS-sovellus

LAPIN AMK
Lapland University of Applied Sciences

Abstract of Thesis

School of Business and Culture
Degree Programme in Business Information Technology
Bachelor of Business Administration

Author	Vallineva Tuomo	Year	2018
Supervisor	Vuokila Johanna		
Commissioned by	Kotipalvelu		
Subject of thesis	Design and implementation of user-friendly CMS application		
Number of pages	38 + 3		

The subject of this thesis is the design and implementation of an easy-to-use CMS application. The home help service company, the commissioner of this thesis, needed an easy-to-use publishing system for maintaining the website. The company did not have any existing home pages. There were numerous publishing systems, but most were cumbersome and complicated to use. From this point of view, the idea was to build an agile and easy-to-use CMS application. The commissioner wanted an easy-to-use publishing system. The application created worked on all devices such as laptop, smartphone and tablet, because the application was responsive. The thesis presents theory and practice on how to build an easy-to-use application. I explain what I did and I justify with reasoning. Acquired the resources to maintain websites for the commissioner, but if the pages required updating evenly and periodically, there may be problems with time management. It would therefore be best for the commissioner to update the website personally. The website is the easiest to update with the CMS application. Using a publishing system does not require much technical knowledge.

In this thesis, I handled the basics of usability, the technologies that I used and the different stages of my work. When I finished my first version, I made a usability test for this version. The results of the usability test were evaluated and, on this basis, the application was further developed. I present the results of my work in this thesis. At the end of this thesis, I reflected on how well or poorly did this work succeed.

The application runs under PHP, HTML language and Bootstrap which is an open-source front-end library. The information processed in the application was stored in the MySQL relational database.

Key words	Web application, PHP, MySQL, Bootstrap, usability
Special remarks	CMS-application

SISÄLLYS

1 JOHDANTO

Opinnäytetyöni aihe on kehittää helppo ja kevyt ratkaisu ylläpitää nettisivuja. Nettisivut tulevat eräälle kotipalvelua tarjoavalle yritykselle. Kyseisen yrityksen yrittäjä on pyytänyt minua tekemään yritykselleen nettisivut. Tämä pyyntö tehtiin alkusyksystä 2017. Alkuvuodesta 2018 tuli ajankohtaiseksi opinnäytetyön tekeminen. Ajattelin että nettisivujen toteuttamisen voisi yhdistää opinnäytetyöhön. Opinnäyteyönä suunnittelin CMS-sovelluksen eli toiselta nimeltään Web-julkaisujärjestelmän jolla nettisivuja ylläpidetään.

Yrityksellä, jolle teen CMS-sovelluksen ei ole omaa nettisivua olemassa. Yrittäjällä ei ole kovin vahvaa teknistä osaamista. Minulla on nyt resursseja tehdä kyseiselle firmalle nettisivut, mutta jos sivuja pitää päivittää tasaisin määräajoin voi tulla ajankäytön kanssa ongelmia. Siksi on parasta, että yrittäjä itse päivittää oman yrityksensä nettisivua. Nettisivuja on kaikista helpointa päivittää WEB-julkaisujärjestelmällä eli CMS-sovelluksella. Julkaisujärjestelmän käyttäminen ei vaadi paljoa teknistä osaamista. Siksi päädyin, että nettisivut toteutetaan julkaisujärjestelmän avulla. Sivujen ylläpitäjällä ei tarvitse olla osaamista nettisivujen koodaamisesta. Jos osaa käyttää esimerkiksi verkkopankkia ja tekstinkäsittelyohjelmaa, niin silloin pitäisi myös olla valmiudet käyttää WEB-julkaisujärjestelmää.

1.1 Toimeksiannon vaatimukset

Ennen sovelluksen suunnittelua määriteltiin yhdessä toimeksiantajan kanssa, millainen sovelluksen tulisi olla ja mitä sillä pitäisi pystyä tekemään. Määrittely tapahtui puhelimitse ja WhatsApp-viestisovelluksella. Keskustelujen pohjalta pystyin järjestelmästä määrittelemään seuraavaa:

- Asiakas voi luoda uuden sivun sovelluksella ja muokata sitä myöhemmin.
- Sivuja pitää olla mahdollisuus poistaa.
- Sovelluksen termien pitää olla yksinkertaisia ja helposti ymmärrettäviä.

- Sovellusta pitää voida käyttää niin läppärillä kuin tablettitietokoneella.
- Sovellus pitää olla suomenkielinen.
- Sivuille voi lisätä valokuvia.

1.2 Raskaat CMS-sovellukset

Vapaasti käytettävät julkaisujärjestelmät ovat yleensä raskaita kooltaan ja niissä on paljon toiminallisuuksia mitä ei tule koskaan hyödynnettyä. Etenkin jos julkaisujärjestelmään lisätään useita lisäosia, niin lopputulos on välillä sellainen, että sivujen lataus WEB-selaimella on hidasta. Raskaat olemassa olevat julkaisujärjestelmät puoltavat sitä, että luon oman julkaisujärjestelmän, jossa on vain ne ominaisuudet mitä tarvitaan. Täten julkaisujärjestelmä pysyy kevyenä.

WordPress on kaikkein suosituin CMS-järjestelmä, joka on käytössä 30,7 % maailman kaikista nettisivuista (W3techs 2018). CMS-järjestelmien välisessä kilpailussa WordPressillä eli WP:llä on yli 60 %:n markkinaosuus (Sucuri.net 2016). WordPress-julkaisujärjestelmä on hyvin helppokäyttöinen ja helppo asentaa. Useat WEB-hotellien palveluntarjoajat mainostavat WordPressin asentamista ja ovat helpottaneet WP:n asentamista mahdollistamalla sen muutamalla klikkauksella. Helppouden varjoon jää helposti se, että se on raskas pyörittää palvelimelle. Tätä ongelmaa helpottamaan on tarjolla WordPress-lisäosia jotka pyrkivät vähentämään kuormitusta kuten W3 Total Cache (Wordpress.org 2018). Wordpress.org (2018) omat ohjesivut kehottavat käyttämään cache-lisäosia. Cache-lisäosat hyödyntävät käyttäjien selainten omaa välimuistia ja suuriin palvelimiin kertynyttä välimuistia. Sen sijaan, että sivut ladattaisiin aina palvelimilta, hyödynnetään välimuisteihin jäänyttä dataa. (Wordpress.org 2018.) Näin ollen kevennetään sen palvelimen kuormaa, jolla WordPress pyörii.

Suuri suosio on tuonut WordPressille myös tietoturvaongelman. Sucuri.netin (2016) mukaan saastuneista Websivu alustoista 78% on WordPress-sivuja Q1-kvartaalilla vuonna 2016. WordPress itsessään on varsin turvallinen, mutta lisäosilla on merkittävä vaikutus WordPressin turvallisuuteen. Hannes Trunden ja Edgar Weipplin (2015, 3) analyysissä tuodaan esille, että turvallisuusongelma

johtuu suurimmaksi osaksi lisäosista. Lisäosien tekijät eivät aina huolehdi tietoturvallisuudesta ja lisäosia ei päivitetä tarpeeksi usein.

WordPressin kanssa työskentelevien ammattilaisten, mukaan on hyvin merkityksellistä minkälaiselle palvelimelle WP:n asentaa (Laukkarinen 2018). Uusimmat ajantasaiset ohjelmistot sisältävä palvelin toimii huomattavasti nopeammin WordPressin kanssa (wordpress.org. 2018). Jeff Geerlin (jeffgeerling.com 2015) testin mukaan PHP 7 on huomattavasti nopeampi WordPressin kanssa kuin PHP 5.6 -version kanssa. Tämä on toki luonnollista, sillä kehitys ja innovaatiot parantavat tekniikkaa, mutta samalla se kertoo WP:n raskaudesta. Kevyet järjestelmät toimivat hyvin myös palvelimilla, missä on vanhempaa teknologiaa.

Toinen syy on helppokäyttöisyys. Isot julkaisujärjestelmät saattavat sekoittaa käyttäjänsä siitä syystä, että järjestelmässä on paljon erilaisia ominaisuuksia. Tekemällä kevyen järjestelmän tehdään samalla järjestelmä, missä ei ole liikaa kaikkea. Palvelussa pitää olla vain sellaiset palvelut mitä käyttäjä tarvitsee, ei yhtään sen enempää. (Sinkkonen, Nuutinen & Törmä 2009, 36.) Siten se on myös helppokäyttöinen järjestelmä.

1.3 Kehitystyön menetelmät

Opinnäytetyöni tutkimusote on toiminnallinen. Kanasen (2012, 27) Kehittämistutkimus opinnäytetyönä -kirjan mukaan toiminallisessa tutkimusta tehdään tutkimussuuntana käytännöstä teoriaan päin. Se sopii minun opinnäytetyöhön, sillä aion käytännön kokeiluilla tutkia, minkälainen on helppokäyttöinen CMS-sovellus. Ensin toteutan sellaisen kokeiluversion CMS-sovelluksesta, joka on omasta mielestäni helppokäyttöinen. Seuraavassa vaiheessa testaan sovelluksen muilla ihmisillä. Jos käyttäjätutkimus osoittaa, että tekemäni sovellus on vaikea käyttää, aion muuttaa käyttösovellusta. Korjattu versio testataan taas uudestaan käyttäjätutkimuksella. Tämä testaus ja korjaus tehdään niin kauan, että sovelluksen käyttäminen on riittävän helppokäyttöistä. Testaus ja havaittujen erojen korjaus tekee siitä toimintatutkimuksen. Toimintatutkimuksen ero kehittämistutkimukseen on se, että tutkija on myös mukana kehittämiskohteen toiminnassa, kun taasen kehittämistutkimuksessa tutkija ei osallistu ollenkaan toimintaan (Kananen 2012, 41).

Hankin taustatietoa käytettävyyttä käsittelevästä kirjallisuudesta. Käytettävyydestä löytyy kirjallisuutta niin suomen kielellä kuin englanniksi. Paras kirjallisuus käsittelee nettisivujen käytettävyyttä, mutta yleinen käytettävyyttä ja käyttökokemusta käsittelevä kirjallisuus on myös käyttökelpoista materiaalia. Käyttökokemus on laaja ilmiö ja ei koske vain nettisivuja.

Minulla on omia kokemuksia eri verkkosivujen käytettävyydestä. Huonot käyttökokemukset kannattaa analysoida sillä tuloksista voi päätellä millaisia ratkaisuja ei kannata toteuttaa.

Sitten kun sovellus on kehitetty toimivaksi, niin sovelluksella tehdään käytettävyystesti koehenkilöillä. Käytännön kokeen sovelluksella voi tehdä muutamalla henkilöllä. Jos vähäisen teknisen taidon omaava henkilö onnistuu lisäämään nettisivuille sisältöä järkevästi, johtopäätös on se, että sovellus on helppokäyttöinen.

2 HELPPOKÄYTTÖISTEN SOVELLUSTEN PIIRTEET

Helppokäyttöisyys on sitä, että henkilö kokee käyttäjäkokemuksenaan, että palvelua on hänen mielestään helppo käyttää. Käyttäjäkokemuksella tarkoitetaan käyttäjän tuntemuksia tämän käyttäessään palvelua. (Sinkkonen, Nuutila & Törmä 2009, 23.) Käyttäjäkokemus on siis jokaisen oma henkilökohtainen kokemus. Koska ihmiset ovat erilaisia, ovat kokemukset myös erilaisia. Käyttäjäkokemuksen tutkimisella yritetään saada selville mikä on keskimääräinen kokemus asiassa. Tutkin käyttäjäkokemusta jotta osaisin tehdä helppokäyttöisen sovelluksen.

Verkkosivuston käyttökokemus koostuu sivun omaksuttavuudesta, sisällöstä ja sisällön merkityksellisyydestä. Käyttäjäkokemukseen liittyy myös visuaalinen ilme ja miten löydettäviä asiat ovat sivuilla. (Sinkkonen, Nuutila & Törmä 2009, 23.)

Käytettävyys on käyttäjä- ja tehtäväriippuvainen, samoin kuin kuinka hyvin käyttäjä pystyy saavuttamaan sen, mitä on asetettu tavoitteeksi, kuinka tehokkaasti tämän voi tehdä ja kuinka tyytyväinen käyttäjä on prosessin aikana ja sen jälkeen. (Whitehead 2006, 788).

Internetissä tärkeintä on käytettävyys. Tämä tarkoittaa yksinkertaisesti sitä, että jos käyttäjä ei löydä jotakin tuotetta, hän ei myöskään osta sitä. Internetissä asiakas on kuningas. Jos palvelu ei tyydytä, asiakkaan on helppo siirtyä muualle, koska myös kaikki kilpailijat ovat vain hiiren liikautuksen päässä. Netissä on paljon sivuja, joten valinnanvaraa on tarjolla runsaasti. Internetin käyttäjät ovat hyvin kärsimättömiä. Jos internetin käyttäjä ei opi käyttämään sivua yhdessä minuutissa, hän lähtee sivulta pois hyvin todennäköisesti. (Nielsen 2000, 9, 10.) Tämä pätee hyvin myös CMS-sovellukseen. Jos käyttökokemus on huono ja asioiden tekeminen on haastavaa, alkaa helposti käyttäjä pohtimaan toisen CMS-sovelluksen hankkimista. Jos kokemukset ovat huonoja, voi sivuston uusimisvaiheessa olla houkutus suuri vaihtaa ulkoasun mukana myös CMS-järjestelmä.

Whitehead (2006, 1) mainitsee McLaughin ja Skinnerin jakavan kirjoituksessaan käytettävyyden kuuteen erilliseen komponenttiin:

1. Tarkastus: Varmistetaan sähköisellä tunnustautumisella kuka pääsee tarkastelemaan tietoja.

2. Luottamus: Käyttäjillä on luottamus omaan kykyyn käyttää järjestelmää ja luottamus myös itse järjestelmään.

3. Kontrolli: Käyttäjät hallitsevat järjestelmän toimintaa, etenkin kun tiedot syötetään järjestelmään ja on mahdollisuus poistaa tiedot.

4. Helppokäyttöinen: Järjestelmä on helppokäyttöinen.

5. Nopeus: Järjestelmää voidaan käyttää nopeasti.

6. Ymmärtäminen: Järjestelmä on ymmärrettävä.

Näissä teeseissä on hyvän käytettävyyden ydin: käyttäjä tietää aina mitä on tekemässä. Käyttäjällä on turvallinen olo turvallisuuden suhteen. Syötettävä tieto nettipalvelussa pysyy salassa niiltä joille asia ei kuulu. Tämä tarkoittaa sähköistä käyttäjätunnistusta. Käyttäjä voi olla myös turvallisella mielellä käyttäessään palvelua, sillä hän tietää, että palvelu tekee vain sen mitä pitääkin tehdä: poistaessaan yhden nettisivun käyttäjä tietää, ettei samalla häviä kaikki muut sivut. Nielsenin (2000, 46) mukaan WEB-sivujen tärkein ominaisuus on nopea latautumisaika. Nielsen on tehnyt tutkimusta, että suositummat nettisivut ovat myös nopeinten latautuvia sivuja. Nopeutta arvostetaan käyttäjien keskuudessa. Hitaasti toimivat sivut karkottavat osan käyttäjistä muualle, koska eivät jaksa odottaa.

Asioiden ja toimintojen muistettavuus on haasteellista ihmisille. Kognitiiviset psykologit ehdottavat, että vuorovaikutuksessa ympäristön kanssa mielemme rakentaa henkistä malleja siitä, miten asiat toimivat (Johnson-Laird 1983, 10). Vain testaamalla sovellusta, voi oppia käyttämään sovellusta.

Jos toteutus on yhdenmukaisia niin muistijälki tukee kykyä tulkita ja antaa siten kyvyn olla vuorovaikutuksessa järjestelmän kanssa. Käyttäjä luo itselleen henkisen mallin (mielikuvan) minkä mukaisesti hän uskoo voivansa käyttää järjestelmää. Järjestelmän suunnittelija on pyrittävä tuomaan oma henkisen mallista käyttäjien käytettäväksi. Ongelmana on, ettei käyttäjän henkinen malli vastaa suun-

nittelijan mallia, koska suunnittelija ei puhu suoraan käyttäjän kanssa. Suunnittelija voi vain puhua käyttäjälle suunnittelevansa järjestelmän kautta. (Norman 1988, 16.) Siksi järjestelmän on kyettävä ohjaamaan käyttäjää.

Tunnetuin heurististen sääntöjen kokoelma on Jacob Nielsenin (1995) 10 heuristisen ohjeen lista:

1. Vuorovaikutuksen käyttäjän kanssa tulee olla yksinkertaista ja luonnollista. Järjestelmän tulisi aina pitää käyttäjät ajan tasalla siitä, mitä tapahtuu, asianmukaisen palautteen ansiosta kohtuullisen ajan kuluessa.

2. Järjestelmän tulisi puhua käyttäjien kieltä, käyttäjille tuttuja sanoja, lauseita ja käsitteitä eikä järjestelmäperusteisia termejä.

3. Käyttäjien pitää hallita tilanne täysin itse. Käyttäjällä pitää olla mahdollisuus palata takaisin (hätäuloskäynti). Kumota tilanne ja yrittää uudelleen. Poistumistiet merkittävä selkeästi.

4. Käyttöliittymän tulee olla yhdenmukainen. Käyttäjien ei pitäisi joutua miettimään tarkoittaako erilaiset sanat ja käyttötilanteet samaa asiaa. Samaa asiaa ei saa sanoa eri tavalla samassa sovelluksessa.

5. Järjestelmän tulee antaa käyttäjälle kunnon palautetta reaaliajassa.

6. Minimoi käyttäjän muistikuormaa. Käyttäjän ei tarvitse muistaa asioita sovelluksen eri vaiheissa. Järjestelmän käyttöohjeiden on oltava näkyviä tai niitä on helppo hakea aina kun se on tarkoituksenmukaista.

7. Oikopolkuja ja tehokasta työskentelyä tulisi tukea.

8. Virheilmoitusten tulee olla selkeitä ja ymmärrettäviä.

9. Hyvin tehtyjä virheilmoituksia parempi on huolellinen suunnittelu, joka estää ongelman syntymisen. Poista ongelmat tai tarkista virheet ja anna käyttäjille vahvistusvaihtoehto ennen kuin he sitoutuvat toimintaan.

10. Käyttöliittymässä tulee olla kunnolliset avustustoiminnot ja dokumentaatio.

2.1 Käytettävyys tietojärjestelmissä

Yksi osa käytettävyydeltään hyvää sovellusta/WEB-sivua on se, että sovellus toimii usealla eri laitteella. Sivun katseluun käytetyn näytön kokoa on mahdoton arvioida, joten suunnittelussa pitäisi ottaa huomioon kaikki vaihtoehdot (Nielsen 2000, 29). Aion tehdä responsiiviset nettisivut. Sovelluksen pitää toimia niin tietokoneella kuin myös älypuhelimella. Mobiililaitteiden ja tablettien käyttö on ylittänyt perinteisten tietokoneiden käytön netin katselussa (Gibbs 2016). Karen H Jinin (2017, 115) mukaan nykyaikaisten verkkosivujen on oltava mobiiliystävällisiä, jotta ne voivat tuottaa optimaalisen käyttökokemuksen niin työpöytäkäytössä kuin mobiililaitteilla. Responsiivinen suunnittelu tarkoittaa esisijaisesti sitä, että sisällön rakenne ja asettelu suunnitellaan niin, että se sopii hyvin kullekin laitteelle ja sen normaalille käyttötavalle (Lehdonvirta & Korpela 2013, 103). Nettisivu joka on suunniteltu näyttämään hyvältä läppärin ja pöytätietokoneen näytöltä, ei välttämättä toimi hyvin pieneltä älypuhelimen näytöltä. Isolle näytölle sovitettu nettisivu näyttää pienellä näytöltä katsottuna hyvin pieneltä. Pienelle näytölle ei mahdu niin paljon kuin isommalle näytölle. Siksi pienelle näytölle on hyvä sovittaa olennaisin sisältö ja siksi se näyttää isommalta näytöltä pelkistetyltä.

Menneinä vuosina oli yleinen tapa tehdä mobiililaitteille omat sivut, jotka sopivat mobiililaitteiden näytölle. Nykyään on yleistynyt tehdä yksi nettisivu, joka mukautuu sen mukaan, millaisella näyttöpäätteellä sitä käytetään. Sivun koodiin on koodattu koodinpätkä, joka tunnistaa millainen näyttö sivulla vierailevalla on. Media queries eli mediakysely on W3C:n hyväksymä standardi jolla tunnistetaan käytetty media (W3C 2012). Tunnistuksen perusteella vierailijan näytölle tulostetaan sellainen ulkoasuversio, joka on paras sen kokoiselle näytölle.

Nyrkkisääntö on, että itse sisällön pitäisi viedä vähintään 50% mieluummin jopa 80% näytöstä. Navigointiapuvälineille vastaava luku on alle 20% tavallisilla sivuilla. Yksinkertainen on aina monimutkaista parempi vaihtoehto. Perusperiaate kaikkien käyttöliittymien suunnittelussa on se, että kaikki elementit poistetaan yksi kerrallaan. Mikäli tuote toimii ilman elementtiä, se voidaan poistaa. (Nielsen 2000, 22.)

Käyttäjän täytyy aina tietää, missä on, mitä voi tehdä ja mihin siirtyä (Sinkkonen, Nuutila & Törmä 2009, 288). Käyttöliittymä pitää olla selkeä ja ymmärrettävä. Käyttäjän pitää pystyä ymmärtämään millä sivulla hän juuri on, joka näkyy ruudussa. ”Uuden sivun luonti”-sivulla pitää siis pystyä ymmärtämään, että tällä sivulla ollaan luomassa uutta sivua.

Tuotteen täytyy olla visuaalisesti miellyttävä ja vastata taloudellisia ja teknisiä vaatimuksia (Sinkkonen, Nuutila & Törmä 2009, 289). Visuaalinen, mutta samalla taloudelliset ja tekniset vaatimukset täyttävä tuote on haaste, mutta ei mahdoton tavoite. Sovelluksen kohdalla kevyt on myös taloudellinen, sillä kevyt käyttää tietoliikennettä paljon vähemmän kuin raskas sovellus. Visuaalisesti näyttävää nettisivua luodessa on houkutus käyttää paljon kuvia. Kuvat kuluttavat verkkoyhteyttä ja serveriä. Kuvat hidastavat myös sivujen latausnopeutta (Nielsen 2000, 46). Palvelun visuaalisen suunnittelun pitää tukea tärkeiden asioiden havaitsemisessa oikeassa järjestyksessä ja auttaa tulkitsemaan oikein (Sinkkonen, Nuutila & Törmä 2009, 36). Tyhjä tila sivuilla ei ole tilan ”haaskausta” jos tarkoituksena on selkiyttää sisällön jaottelua. Tyhjän tilan avulla voidaan ohjata katsetta ja selkiyttää sisällön jaottelua. (Nielsen 2000, 18.)

2.2 Käyttäjäkeskeinen suunnittelu

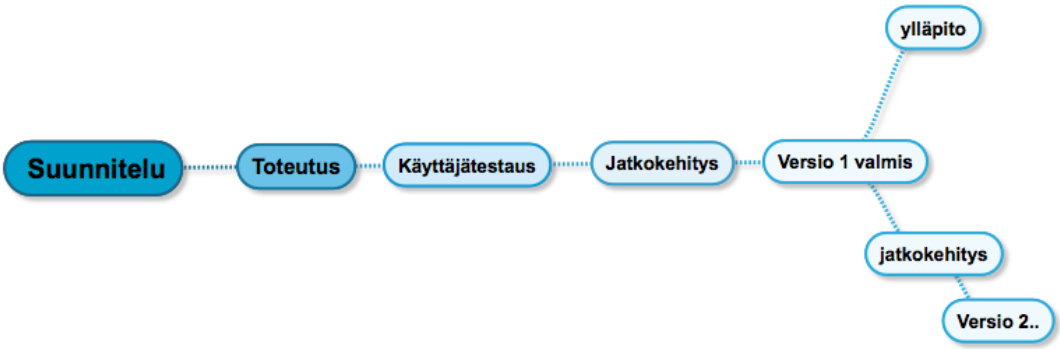
Henkilö, jolla on heikko tekninen osaaminen, on haastava kohderyhmä, jolle suunnittelee sovellusta. Sovelluksen pitää olla selkeä hahmottaa. Termien pitää olla selkeästi ymmärrettäviä (Sinkkonen, Nuutila & Törmä 2009, 36). Painikkeessa josta pääsee lisäämään uuden WEB-sivun, pitää lukea selkeästi ”Lisää uusi sivu”. Jos painikkeessa lukee vain ”Lisää”, voi käyttäjälle jäädä epäselväksi mitä lisätään kyseisellä painikkeella.

Monimutkaisuus sekoittaa usein etenkin uusia käyttäjiä. Jos jonkun asian toimittamista varten täytyy tehdä lukuisia klikkauksia ja edetään monimutkaisten valikkojen kautta, saattaa asian loppuunsaattaminen jäädä kesken. (Nielsen 2000, 9, 10.) Toiminallisuuksia kannattaa hioa siten että polku maaliin olisi mahdollisimman lyhyt ja helppo.

Minulla itselläni on huonoja kokemuksia eräästä toiminnanohjausjärjestelmästä. Järjestelmää ei voi mitenkään sanoa helppokäyttöiseksi. Järjestelmää pitää käyttää pitkän aikaa ja lukuisia kertoja ennen kuin jonkun tehtävän hoitaminen on sujuvaa. Jonkun asian suorittamista varten voi joutua käyttämään useita eri transaktioita ja yhdessä transaktiossa täytyy painaa montaa eri nappia ja täyttää monta eri kohtaa. Voin vain miettiä kuinka paljon työaikaa säästyisi, jos asiat olisi suunniteltu käyttäjäkeskeisestä näkökulmasta. Tämän tuotteen kohdalla ei voida kritiikkiä tosin kohdistaa vain tuotteen valmistajaan, koska jokainen käyttäjäyritys suunnittelee ja toteuttaa sisällön itse ja sen miten se toimii.

3 SOVELLUKSEN SUUNNITTELU JA TOTEUTUS

Ensimmäinen vaihe sovelluksen tekemisessä on suunnittelu. Olin jo aikaisemmin päättänyt, että sovellus tehdään PHP-kielellä ja MySQL-relaatiotietokantaa käyttäen. Olen muutama vuosi sitten koodannut oman sovelluksen omaan henkilökohtaiseen ja puolisoni käyttöön. Sovelluksen pohja ja käyttöliittymä olivat mielestäni toimivia, joten otin sen suunnittelutyön pohjaksi. Matkan varrella sovellus on muuttunut huomattavasti.



Kuvio 1. Käytännön toteutuksen vaiheet

Tässä opinnäytetyössä edetään suunnittelusta vaihe vaiheelta siihen asti, kun versio 1 on valmis (kuvio 1). Jatkokehitys ja ylläpito ovat opinnäytetyön jälkeistä aikaa.

Kuten yleensäkin CMS-sovelluksissa, sovelluksella on kaksi osiota. Toinen osio on kotisivu eli jotain asiaa varten tehty nettisivu. Se on yleensä julkinen sivu. Toinen osio on ”hallintapaneeli” minkä kautta hallitaan tätä toista puolta. Tämä hallintapaneeli on lähes poikkeuksetta salattu, johon pääsee sisään kirjautumalla sovellukseen sisään.

Tässä opinnäytetyössä tein CMS-sovelluksen jossa on hallintapaneeli ja nettisivu, jonka sisältöä hallitaan tämän hallintapaneelin kautta.

Olen suunnitellut tekeväni hyvin kevyen ja riisutun CMS-sovelluksen. Sovelluksen avulla voi lisätä sivun ja muokata sivua myöhempänä ajankohtana. Sivun poistaminen kokonaan onnistuu myös. Sovelluksen käyttäjä voi profiilin asetuksissa muokata salasanaa.

Jos sovellukseen haluaa lisätä muita toimintoja, niin sekin on periaatteessa helposti tehtävissä. Sovellus on laajennettavissa, mutta samanlaista lisäosien lisäys- ja hallintasysteemiä en ala rakentamaan, mitä on esim. Joomlassa ja WordPressissä. Jos jotain halutaan lisätä, se tehdään käsityönä. Toinen asia on se, kannattaako laajennusta edes tehdä. Tämä CMS-sovellus soveltuu parhaiten pienelle nettisivulle. Jos sivuille halutaan vaikka yksi palautelomake, ei sen takia kannata rakentaa sovellukseen palautelomakkeen hallintasivua. Helpointa on tehdä PHP-kielellä tehty palautelomake-sivu ja käsittelyä varten toinen tiedosto joka lähettää palautteen haluttuun sähköpostiin. Palautelomakkeen sivu lisätään sitten julkisen sivun PHP-koodiin ja se toimii osana sivua, vaikka se on itsenäinen lomakesivu. Ideologiana tässä on pitää sovelluksesta pois kaikki mikä ei ole välttämätöntä. Kun pienetkin asiat pidetään kevyinä, ei kokonaisuus ryöpsähdä hallitsemattoman suureksi, jos kaikesta ollaan tarkkana.

Nettisivu jota tällä sovelluksella hallitaan voi ulkoasultaan olla lähes minkä näköinen tahansa. Ulkoasuun lisätään tarvittavat koodit. Valikon koodipätkä lisätään siihen kohtaan, johon valikko on suunniteltu laitettavaksi. Sisällön koodit laitetaan siihen kohtaan, sisällön on tarkoitus sijaita. Sama sivupohja on kaikilla sivuilla, jotka lisätään tällä sovelluksella.

Ensimmäisenä suunnittelussa sovelluksen nettiisivu jaetaan eri osiin. Ylimmässä osassa on navigointivalikko. Valikon avulla sivulla suunnistetaan eteenpäin. Keskiosassa sivua näytetään sisältö. Tässä osiossa tapahtuu sivujen varsinainen toiminnallisuus eli esim. tehdään uusia sivuja ja muokataan ja poistetaan sivuja. Sivun alaosassa on copyright-merkintä.

Seuraavaksi alkoi koodin kirjoittaminen. Ensimmäiseksi rakennettiin kirjautumisjärjestelmä. Sovellukseen pääsee sisään vain sähköisellä tunnistautumisella. Kirjautumiseen käytetyt tiedot on tallennettu relaatiotietokantaan. Relaatiotietokantana käytän MySQL-tietokantaa. Käyttäjätiedot on tallennettu Users-tauluun.

Kun kirjautumisjärjestelmä on valmis, alkoi seuraava osio. Tietoturvasyistä sovelluksen sivuston rakenteen halusin häivyttää. Siksi kirjoitin koodin niin että osoite on muotoa index.php?sivu=. On-merkin jälkeen tulee termi esim. sivu ja muokkaa. Koodiin on kirjoitettu, mille sivulle milläkin termillä pääsee.

Uuden sivun lisäämiselle on oma sivunsa, kuten myös muokkaukselle ja poistamiselle. Ennen kuin nämä sivut voidaan toteuttaa, pitää perustaa relaatiotietokantaan taulu (table). Lisää tietoa tietokannasta kerrotaan luvussa, jossa käsitellään SQL-kieltä ja relaatiotietokantoja.

Sisältö-kenttään lisätään uuden sivun sisältö. Tähän kenttään on upotettu WYSIWYG-editori. Se on lyhenne sanoista: What You See Is What You Get (Markoff 2007). Suomeksi se on ”mitä näet, sen myös saat”. WYSIWYG -editori on html-editori. Editorin nimi on NicEdit. Editorilla voi muokata sisältöä sen näköiseksi kuin haluaa sen olla nettisivulla. Editori muuttaa sisällön html-koodiksi ja tallentaa sen tietokantaan. Jos ei olisi WYSIWYG-editoria käytössä, pitäisi sovelluksen käyttäjän itse kirjoittaa html-koodia tekstikenttään. Tämä editori on lisätty helppokäyttöisyyttä lisäämään (Flynn 2014, 40).

Jaoin sivut kahteen kategoriaan: pääsivut ja alasivut. Sivut joiden statukseksi merkitään pääsivu, lisätään valikkoon. Sivut joiden status on alasivu, ei näy missään listauksessa. Jokaiselle sivulle luodaan oma id-numero. Alasivuille voi tehdä hyperlinkin jolloin kyseiselle sivulle pääsee sisään.

3.1 Käyttöliittymän suunnittelu

Nielsenin (2000, 188) mukaan käyttöliittymän avulla käyttäjän on kyettävä vastaamaan kolmeen peruskysymykseen:

- Missä minä olen?
- Mistä minä tulin?
- Minne täältä voin mennä?

Jos käyttäjä ei tiedä sijaintiaan, ei hän ymmärrä sivun merkitystä ja käyttötarkoitusta. Siksi sivulla on hyvä olla selkeä otsikko, joka kertoo sivun tarkoituksen (Nielsen 2000, 191). Jos ei tiedä missä on, harvemmin tietää myös mistä on tullut. Selaimen paluu -painikkeella pääsee onneksi takaisin edelliselle sivulle. Kysymykseen ”minne voi mennä seuraavaksi?” voi tarjoilla vastauksen selkeillä linkki -vaihtoehdoilla. Termien merkitys on tässä siis tärkeää.

Keskeisimmät toiminnallisuudet ja niille vievät linkit ja painikkeet on hyvä sijoittaa mahdollisimmat keskeisille paikoille. Tämä tarkoittaa näytön yläosaa ja keskiosaa näyttöä. Näkökenttä hakeutuu yleensä niihin helpoiten.

Suunnittelin navigointivalikon sivun yläosaan. Ylimmäisenä on ”Tervetuloa hallintapaneeliin” -tervehdysteksti. Näin käyttäjä ymmärtää, että on saapunut sisään hallintapaneeliin. Luvussa kaksi kirjoitin termien tärkeydestä ja tämän sovelluksen suunnittelussa ja toteutuksessa on tärkeää, että termit ovat hyvin ymmärrettäviä. Tervehdystekstin alla on navigointivalikko. Sovellus on pieni ja yksinkertainen. Siksi valikkoon ei tarvitse laittaa montaa nimikettä.

Sovelluksen ulkoasu on responsiivinen, joka mukautuu käyttäjän näytön mukaan. Käytin Bootstrapin elementtejä jotka sopivat hyvin responsiivisille sivuille. Täten se sopii hyvin myös mobiiliin käyttöön.

Käytin koodia jolla ulkoasu mukautuu käytettävän näytön mukaan. Koodi käskee leveyden olemaan näytön leveyden mukaisesti:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">  (1)
```

3.2 Tietokannan suunnittelu toteutus ja SQL-kieli

SQL on standardi relaatiotietokannan vuorovaikutteinen ohjelmointikieli (Williams & Lane 2002, 11). Olen suunnitellut, että sovelluksen tietokantana toimii MySQL-relaatiotietokanta. Se on hyvin suosittu tietokanta ja sitä käyttää monet suuret yrityksen verkkosivuillan kuten Facebook, Netflix, Ebay, LinkedIn, Twitter, PayPal, Airbnb, Uber, Youtube ja moni muu firma ja sivusto (MySQL.com 2018). Minulla on kokemusta MySQL-tietokannoista ja olen käyttänyt sitä omissa nettisivuissani.

Sovelluksen olen suunnitellut käyttävän kahta eri taulua relaatiotietokannassa. User-taulu käsittelee käyttäjätunnuksia (kuvio 2). Sovellus-tauluun puolestaan tallennetaan sivujen sisältö (kuvio 3).

user

Column	Type
userID	int(11)
nimi	varchar(50)
username	varchar(50)
password	varchar(120)
email	text

Kuvio 2: user taulu

Sovellus

Column	Type
SivuID	bigint(20)
Otsikko	text
Sisalto	varchar(3000)
status	text
aika	timestamp

Kuvio 3. sovellus-taulu

3.3 Tietoturvallisuus

Tietoturvallisuus on tärkeä osa sovelluksen kehittämistä. Sovellukseen sisään ei pidä antaa kenenkään asiattoman päästä sisään. Siksi käyttäjätunnistus on hyvin tärkeää. Tutkimusten mukaan 80% ihmisten käyttämistä salasanoista voidaan selvittää 30 sekunnissa. (Gilhoolyn 2005 mukaan Fowler 2005.) Parhaat salasanat ovat sekoitus kirjaimia, numeroita ja erikoismerkkejä. Tällaiset salasanat jotka sisältävät kirjaimia, numeroita ja erikoismerkkejä, on vaikeita muistaa, josta johtuen johtuen ihmiset käyttävät helposti muistettavia salasanoja. Mukavat ja helposti muistettavat salasanat ovat yleensä sanoja ja nimiä ja päivämääriä. Helppojen salasanojen käyttäminen on tietoturvariski.

Käytän sovelluksessa salasanan salausta Hash-salauksella joka salakirjoittaa salasanan. Näin ollen salasanan joutuminen ulkopuolisten käsiin on hyvin vaikeaa. Perinteinen käyttäjätunnus ja salasana yhdistelmä ei ole paras mahdollinen ratkaisu ainoana tunnistautumisena. Aion paneutua asiaan jatkokehityksessä tulevaisuuden ajankohtana ja se ei kuulu tähän opinnäytetyöhön.

3.4 Käytetyt tekniikat ja niiden valinta

3.4.1 PHP

Sovelluksessa käytän PHP-ohjelmointikieltä. PHP on lyhenne sanoista Hypertext Preprocessor. PHP:tä käytetään WEB-palvelimella dynaamisten nettisivujen luomiseen. Se on Open Source –lisenssillä, eli se on ilmaiseksi vapaasti käytettävissä. Yksi tai useampi PHP-skripti voidaan upottaa staattisiin HTML-tiedostoihin ja tämä tekee eri toiminallisuuksien yhdistämisestä helppoa yhdelle samalle html-sivulle (Williams & Lane 2002, 1). Kun käyttäjä on nettisivulla, PHP-koodi suoritetaan palvelimella, joka sen jälkeen generoi sen HTML:ksi, joka lähetetään sitten asiakkaan selaimelle takaisin. (PHP.net 2018.) PHP:ssä tiedot tallennetaan erilliseen tiedostoon tai tietokantaan. Tietokantaan tallentaminen on tietoturvallisempaa, sillä tietokanta vahtii aina käyttäjän tunnustautumisen. Tietokannassa pitää aina tietää mitä tietokantataulua käytetään ja millä hakusanoilla haetaan. Ulkopuolisen toimesta tehty tietojenkalastelu on siis hakuammuntaa.

```
<?php
print "Hello world!<br>";
?>
```
(2)

Esimerkki tulostaa sivulle tekstin: Hello wold!

```
<?php

echo date("l jS \of F Y h:i:s A") .

?>
```
(3)

Esimerkin koodi tulostaa päivämäärän ja kellonajan. Päivämäärä ja kellonaika on koodin tulostushetken palvelimen aika.

3.4.2 HTML

HTML on lyhenne sanoista Hypertext Markup Language. Se on avoimesti standardoitu kuvauskieli, jota käytetään internet-sivujen koodikielenä. Tällä kuvauskielellä voidaan määrittää esimerkiksi minkälainen fontti tekstillä on tietyssä kohtaa ja mihin kohtaan valokuva tulee. Nimensä mukaisesti HTML-kielellä voidaan tehdä myös linkkejä toiseen dokumenttiin. HTML-koodilla määritellään minkälainen nettisivun ulkoasu on. HTML-koodissa käytetään elementtejä, joilla määritellään esimerkiksi mikä tekstin osa lihavoidaan. Elementit ovat koodissa sisäkkäin ja peräkkäin. Elementit merkitään kulmasulkein (Sarja 2012).

Lihavoidun tekstin saa komennolla testi Tällä koodilla tulostuu: testi

Hyperlinkin saa tehtyä näin:

Lapin Ammattikorkeakoulu (4)

Tämä tulostaa linkin Lapin ammattikorkeakoulun sivuille.

CMS-sovellus toimii PHP-tiedostojen avulla. Sovelluksen ulkoasun määrittelyssä käytetään html-koodia, joka on sisällytetty PHP-tiedostoon PHP-koodiin väliin.

3.4.3 CSS

CSS tarkoittaa Cascading Style sheets. CSS on WEB-sivuille suunniteltu tyyliohjeiden laji. Sillä voidaan määritellä ehdot, miten WEB-sivut näytetään. CSS:ssä voidaan määritellä esim. tekstin muotoilu, taustaväri ja sivusto rakenne (esim. leveys, korkeus, asemointi). CSS-koodia voidaan lisätä html-koodin sekaan tai se voi olla omana .css-tiedostonaan.

Lähes samat asiat voidaan tehdä suoraan html-kielellä kuin CSS-tyyliohjeena. HTML-kielellä pitää jokainen elementti kirjoittaa joka kerta, kun sen haluaa toteuttaa. CSS:ssä voidaan määrittely tehdä keskitetysti. Kun HTML-koodissa H1-tason otsikon väri määritellään jokaisella kerralla erikseen, voi CSS-koodissa määritellä, että H1-tason otsikot tulostetaan aina tietyllä värillä.

Määritellään H1-tason otsikot sinisiksi:

```
h1 {
  color:blue;
}
```
(5)

3.4.4 Bootstrap

Bootstrap on avoimen lähdekoodin front-end-kirjasto jolla saa WEB-sivujen toiminallisuuksien kuten painikkeiden, lomakkeiden ja typografian ulkoasua muutettua valmiilla vaihtoehdoilla (Getbootstrap.com 2018). Bootstrap-toiminnallisuus lisätään kirjoittamalla komennot sisään html-koodiin. Valmiin Bootstrap-kirjaston käyttäminen helpottaa käyttöliittymän kehittämistä koska sillä voidaan käyttää valmiita malleja. Esimerkiksi painikkeiden ulkoasun saa muutettua erilaiseksi verrattuna perinteiseen painikkeeseen. Bootstrapin valmiiden vaihtoehtojen käyttö säästää aikaa, kun ei tarvitse määritellä CSS-tiedostojen asetuksia.

Bootstrapin käyttö aloitetaan lisäämällä Bootstrapin CSS-tiedostojen linkit html-tiedostoon:

```
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/css/bootstrap.min.css">
```
(6)

Painikkeen ulkoasua voi muuttaa lisäämällä <button> sisään määritteitä:

```
<button type="button" class="btn btn-primary active">Active Primary</button>
<button type="button" class="btn btn-primary disabled">Disabled Primary</button>
```
(7)

Tällä komennolla saadaan kuvion 4 näköisiä painikkeita

Kuvio 4. Painikkeet

3.4.5 SQL

SQL on standardi relaatiotietokannan vuorovaikutteinen ohjelmointikieli (Williams & Lane 2002, 11). SQL on lyhenne sanoista Structured Query Language. Sillä voidaan luoda, muokata ja poistaa tietokantoja. SQL-kielellä haetaan myös tietoa tietokannoista. PHP-koodiin voidaan upottaa SQL-komentoja, joiden avulla tietokantaa voidaan käyttää sovelluksella. SQL on yleinen käytössä oleva kyselykieli relaatiotietokannoissa, joita on määrällisesti lukuisia.

```
SELECT  `Otsikko`, `Sisalto` FROM `Sovellus` WHERE `SivuID`= 36        (8)
```

Tällä komennolla haetaan sivun 36 otsikko ja sisältö sovellus-taulusta.

```
SELECT `Otsikko` FROM `Sovellus`                                       (9)
```

Tällä komennolla haetaan kaikki otsikot sovellus-taulusta.

```
SELECT * FROM Sovellus WHERE NOT status='Alasivu'                      (10)
```

Tällä komennolla haetaan sovellus-taulusta kaikki tiedot, paitsi rivit joiden status on alasivu. Tällä samaisella komennolla määritin haun, jonka avulla nettisivujen valikossa näytetään vain pääsivut. NOT on poissulkeva haku. Hakea voi myös useammalla hakusanalla (AND) ja vaihtoehtoisella (OR) haulla.

3.4.6 Relaatiotiekanta – MySQL

Käytän MySQL-relaatiotietokantaa sovelluksessa. Se on hyvin suosittu tietokanta ja sitä käyttää monet suuret yrityksen verkkosivuillan kuten Facebook, Netflix, Ebay, ja moni muu firma ja sivusto (MySQL.com 2018). MySQL-tietokantoja voi käyttää usean eri ohjelmointikielen kanssa. Näitä on esimerkiksi PHP, Perl, ASP ja Ruby. (Cabral & Murphy 2009, 3.) Sen käyttö on ilmaista niin yksityisille kuin kaupalliseen käyttöön. Useat WEB-hotellien tarjoajat tarjoavat paketeissaan oletuksena MySQL-tietokannan. Täten MySQL-tietokanta on hyvä vaihtoehto relaatiotietokannaksi, koska sovellus sopii liitettäväksi moniin eri WEB-hotelleihin. MySQL-relaatiotietokanta toimii SQL-kielen komennoilla.

3.4.7 TextWrangler

Koodia kirjoitanTextWrangler-ohjelmalla. Se on koodin kirjoittamiseen suunniteltu ohjelma. (Textwrangler 2018.) Ohjelmalla voidaan ottaa FTP-yhteyden avulla yhteyttä palvelimelle ja siten päivittää tiedoston välittömästi. Se toimii myös erillisten FTP-ohjelmien kanssa yhteystyössä. Ohjelma värittää koodin eri osia ja siten visuaalisesti helpottaa koodin lukemista.

3.4.8 NicEdit

Käytän sovelluksessa NicEdit-nimistä WYSIWYG-editoria. Se on helposti asennettavissa sivuille. NicEdit on Brian Kirchoffin kehittämä (NicEdit.com, 2018). Editorilla voi muokata sisältöä sen näköiseksi kuin haluaa sen olevan nettisivulla. Editori muuttaa sisällön html-koodiksi ja tallentaa sen tietokantaan. WYSIWYG-editorin valinnassa oli yhtenä osalueena tärkeää, että lisenssi sallii sen käytön tässä sovelluksessa.

4 KÄYTETTÄVYYDEN TESTAUS TESTIVERSIOLLA

4.1 Tutkimuksen suunnittelu

CMS-sovellus testataan ennen, kuin se voidaan ottaa tuotantokäyttöön. Testauksella varmistetaan, että kaikki toimii eikä sovelluksesta löydy mitään bugeja. Testillä selvitetään myös, onko projektin yksi tavoite sovelluksen helppokäyttöisyydestä saavutettu. Jos testissä ilmenee teknisisiä vikoja, pyritään ne korjaamaan välittömästi. Jos testissä huomataan, että sovellus on vaikeakäyttöinen tai joku osa sovelluksessa on vaikeakäyttöinen, pyritään tilanteeseen etsimään ratkaisu.

Käytettävyystestaus on luotettavampi ja objektiivisempi arviointimenetelmä kuin arviointi asiantuntijoita käyttäen. Hyvin tehty asiaintuntija-arviointi ja käytettävyystestaus tukevat toisiaan. (Sinkkonen, Nuutila & Törmä 2006, 285.) Kun on käyttänyt paljon aikaa toteuttaakseen työnsä, on se vaara, että omaan työnjälkeensä sokeutuu. Omia virheitään ei helposti löydä ja parempia ratkaisuja ei keksitä. Täten on hyvä, että työtä tarkastellaan uusin silmin. Toisen ihmisen näkökanta avaa näkemään mikä käyttöjärjestelmässä on ongelmallista. Järjestelmän kehittäjät tuntevat koko järjestelmän ja ajattelevat asiaa teknisestä näkökulmasta. Se ei ole sama näkökanta kuin järjestelmän käyttäjällä. Käyttäjä näkee vain sen mikä on näkyvää. Uusi näkökanta voi avata luovuuden kehittää entistä parempaa käyttöjärjestelmää.

Käytettävyystestaus on taloudellisesti kannattavaa, vaikka se aiheuttaakin kuluja. Huomaamatta jääneet virheet tulevat huomattavasti kalliimmiksi, mikäli ne ovat päässeet itse julkaisuun.

Liiketoiminnalle kriittisen WEB-sovelluksen alas ajaminen, korjaaminen ja uudelleenkäynnistys voi tapauksesta riippuen viedä paljon aikaa ja näin ollen maksaa suuria rahamääriä (Matbouli & Gao 2012, 5).

Käytettävyystestaus jaetaan yleensä kolmeen eri vaiheeseen: testin suunnitteluun, testin suorittamiseen ja myös testin tulosten analysointiin (Sinkkonen, Nuutila & Törmä 2006, 280–281).

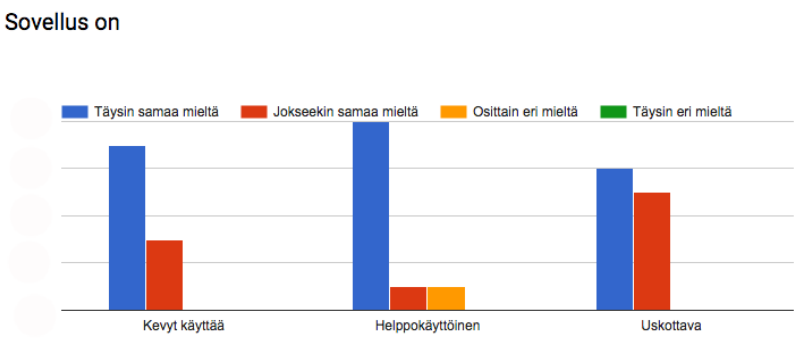
4.2 Toteutus

Käytettävyystestauksen suoritin siten, että otin yhteyttä ihmisiin ja pyysin heitä osallistumaan testiin. Testaushenkilöiksi pyrin saamaan mahdollisimman homogeenisen porukan. Osalla heistä on hyvät tietotekniset valmiudet ja osa oli henkilöitä, joilla oli vähäisemmät tietotekniset valmiudet. Yhteyttä otin Facebookin Messengerin avulla ja omalle Facebook-seinälle laitoin linkin jaon ja pyysin ihmisiä osallistumaan testiin. Testattaville annetaan linkki kyselylomakkeelle, joka on tehty Google Formsilla. Kyselylomakkeessa on ohjeet, miten testi tehdään. Kyselylomakkeen pitää olla yksiselitteisiä, etukäteen mietittyjä ja myös motivoivia. Kysymyksistä pyrin tekemään mahdollisimman selkeitä ja yksinkertaisia (Liite 1).

4.3 Tulokset

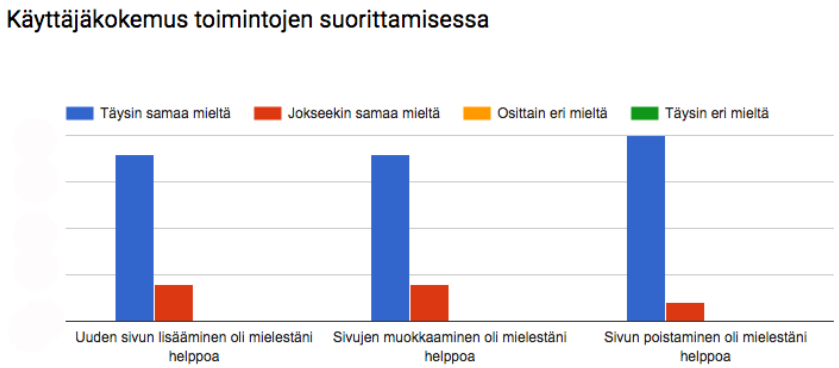
Testit onnistuivat varsin hyvin. Nettisivun lisäys omatoimisesti onnistui kaikilta paitsi yhdeltä testaajalta. Testattava henkilö joka ei onnistunut lisäämään sivua, oli vanhempi henkilö, joka käytti älypuhelinta. Testissä oli nähtävissä, että älypuhelimilla testiin osallistuneet antoivat heikoimmat arvosanat. Pienellä näytöllä sivujen lisäys ja muokkaaminen kyllä onnistuu, mutta se on hankalampaa kuin isommalla näytöllä.

Keskiarvosana sovelluksen yleisestä käytettävyydestä on käytettävyystestin mukaan 8,2 jota voidaan pitää hyvänä. Testaajat pitävät sovelluksen käytettävyyttä siis varsin hyvänä.



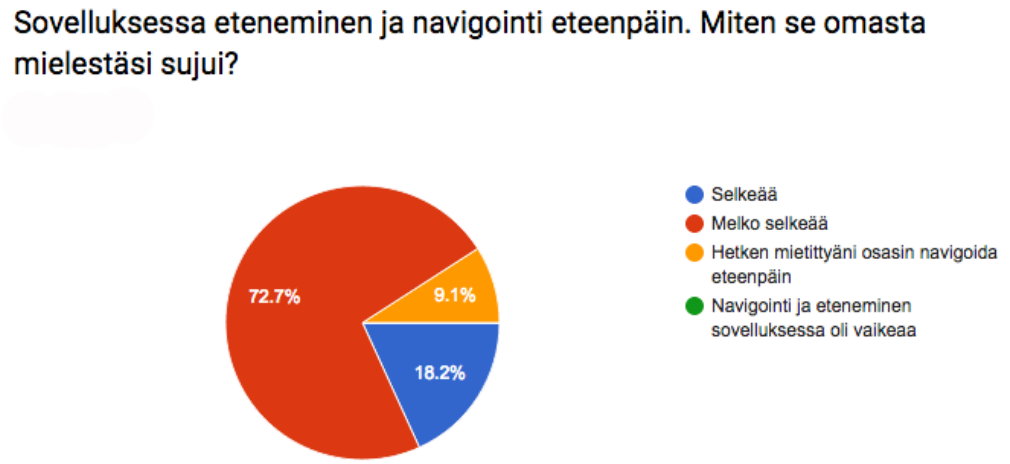
Kuvio 5. Testaajien arvio sovelluksen käytettävyydestä

Pääosa testaajista on sitä mieltä, että sovellus on helppokäyttöinen (kuvio 5). Suurin osa on myös sitä mieltä, että sovellus on kevyt ja uskottava.



Kuvio 6. Käytettävyystestin tuloksia

Testattavien käyttäjäkokemus oli testin mukaan varsin positiivinen (kuvio 6).



Kuvio 7. Käytettävyystestin tuloksia

Suurin osa testaajista osasi edetä omasta mielestään sovelluksessa melko hyvin (kuvio 7). Tämä on kehittäjänä mieluisaa huomata. Toki olisin toivonut, että suurin siivu olisi ollut ”selkeää” -vaihtoehto. Tämä laittaa miettimään, miten valikkoa voisi vielä viilata entistä paremmaksi.

Vapaassa kommentoinnissa tuli tällaista viestiä:

- Painikkeiden termit paremmaksi

- ”Valokuvan lisäys” -ikonia vaikea löytää

4.4 Jatkokehitys

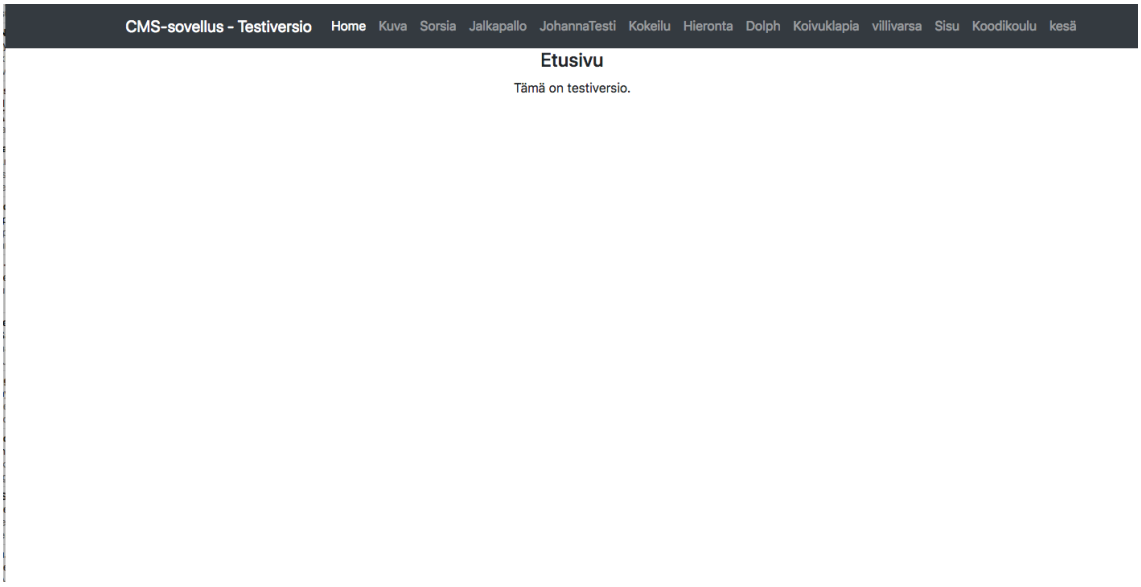
Painikkeiden termeistä tuli palautetta, että ne voisivat olla parempia. "Lisää sivun sisältö" -termistä tuli palautetta, että se ei ole kovin intuitiivinen otsikko, joten vaihdoin termiin ”Lisää uusi sivu”. Tämä kuvaa paremmin sivua millä voi lisätä uuden sivun. Yläosan valikossa oli painike ”sisällönhallinta” josta pääsi katsomaan mitä sivuja on olemassa. Tässä näkymässä on myös ”lisää uusi sivu”-painike. Vaihdoin ”Sisällönhallinta” -termin ”Lisää tai muokkaa sivuja” termiin sillä se kuvaa mielestäni selkeämmin sivun tarkoitusta.

Uuden sivun lisäyssivulla ja myös olemassa olevan sivun muokkaussivulla olevan sisältölaatikon koosta johon sivun sisältö laitetaan, tuli palautetta, että se on pieni tietokoneen näytölle. Laatikko oli pituudeltaan vajaa kymmenen riviä pitkä, joka soveltui hyvin mobiilikäyttöön. Tietokoneen näytöllä se näyttää pieneltä. Kompromissina suurensin laatikon kokoa, mutta en liikaa, ettei se ole liian suuri mobiilikäyttöön.

5 SOVELLUKSEN ESITTELYÄ

5.1 Sovelluksesta yleistä

Sovellus sisältää itse varsinaisen sovelluksen, jolla nettisivua hallitaan. Lisäksi on olemassa julkinen nettisivu (kuva 1), jonka sisältöä tällä sovelluksella hallitaan. Sovelluksella lisätyt ja muokatut nettisivujen sisällöt tallennetaan relaatiotietokantaan ja valokuvat palvelimelle. Julkinen nettisivu hakee tiedot tietokannasta ja valokuvat palvelimelta ja tulostaa määritellyn ulkoasupohjan mukaisesti nettisivuna. Kuva on testiversion etusivulta eli pääsivulta. Yläosassa on valikko, johon PHP-koodi hakee relaatiotietokannasta sivut, joiden status on pääsivu. PHP-koodi tekee linkin kyseisille sivuille.



Kuva 1. Julkisen sivun etusivu

5.2 Uuden sivun luonti ja sivujen muokkaaminen sisällönhallinta-osiossa

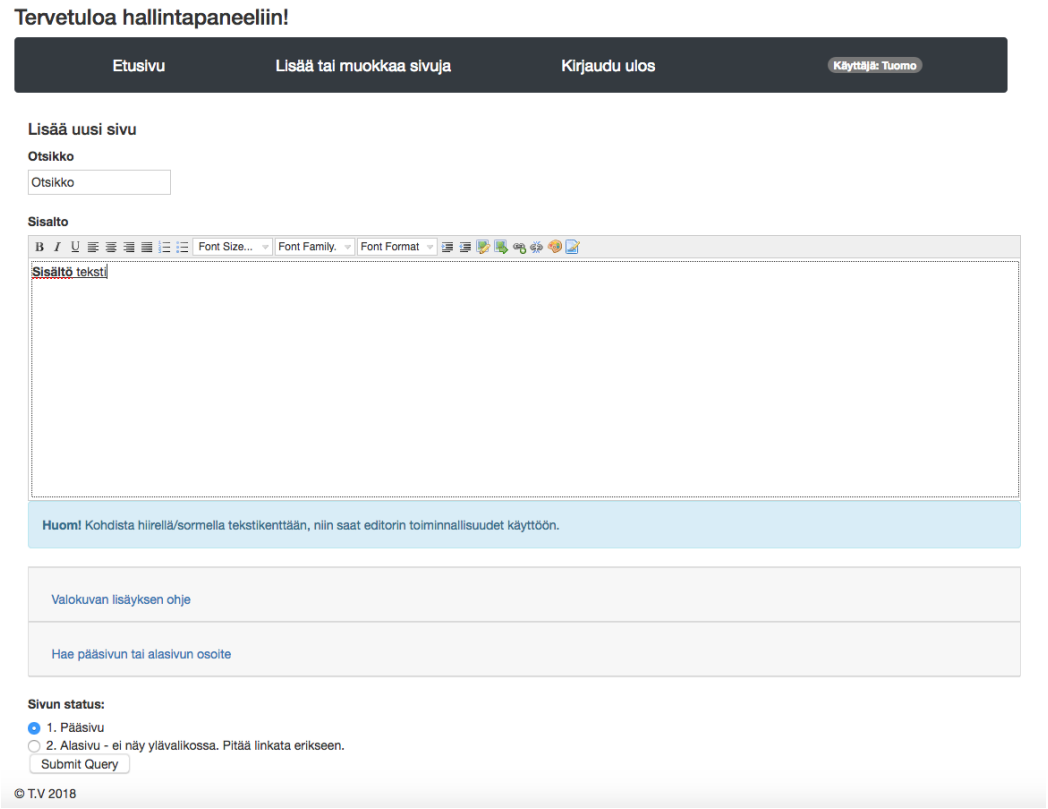
Sovelluksen ylläpito-osioon pääsee sisään kirjautuminen-sivun kautta.

Etusivu on sisääntulosivu, jonne saavutaan, kun ensin ollaan kirjauduttu sovellukseen sisään (kuva 2). Etusivulta jatketaan eteenpäin. Etusivu ei oikeastaan ole mielestäni välttämätön navigointivalikossa. Ajattelen, että se on kuitenkin psykologisesti turvallinen valinta pitää etusivu valikossa mukana, että käyttäjän on

turvallista palata ensimmäiselle sivulle, jos hän jostain syystä ”eksyy” tai tulee ”turvaton olo”. Näin hän pystyy hallitsemaan tilannetta (Nielsen, J. 1995).

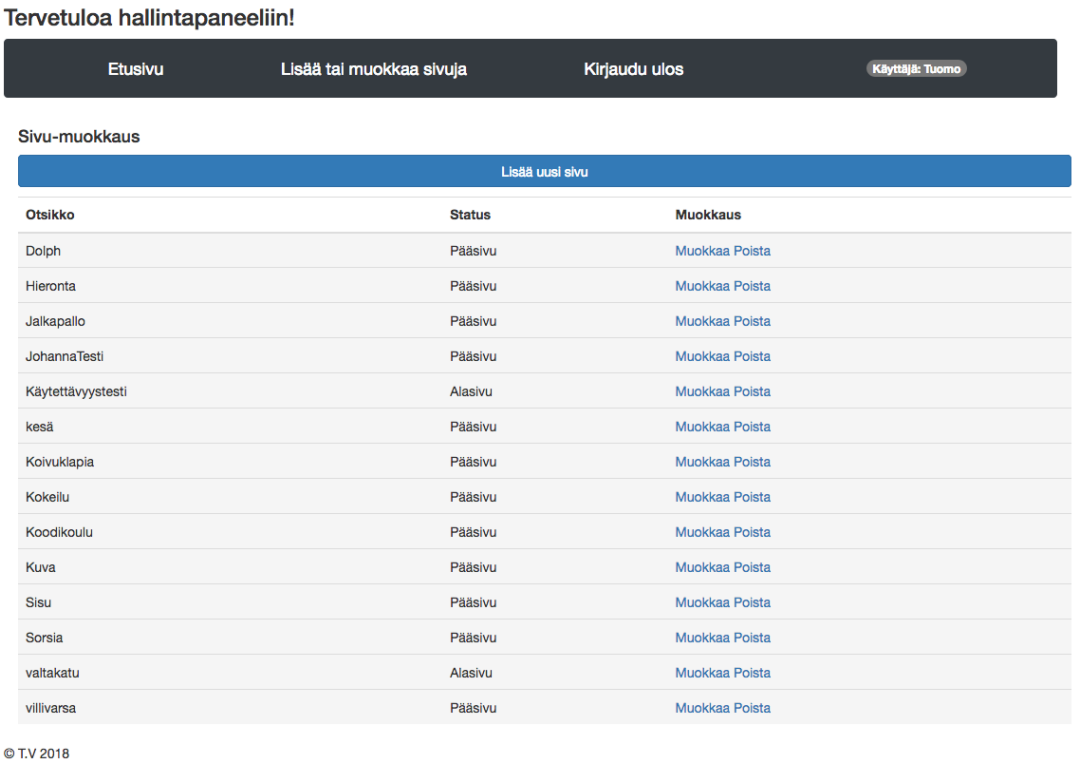
Kuva 2. Sovelluksen etusivu

Uuden sivun tekemiselle on oma sivu (kuva 3). Sivulla on lomake, johon syötetään tiedot. Lomakkeessa on neljä kohtaa: otsikko, sisältö ja sivun status. Lisäksi on lähetä (Submit) -painike. Kun painikkeesta painaa, lähettää sivu tiedot tietokantaan, jonne ne tallennetaan. Otsikko-tekstikenttään kirjoitetaan uuden nettisivun otsikko. Se on tavallinen tekstikenttä.



Kuva 3. Lisää uusi sivu

”Lisää tai muokkaa sivuja” on sovelluksen tärkein sivu ja siksi on tärkeää, että se on valikossa. Sitä kautta pääsee lisäämää uuden sivun. Sivun myöhempi muokkaus ja poistaminen onnistuu myös tätä kautta (kuva 4).



Kuva 4. Sivut listattuna.

Pohdiskelin paljon, kannattaako minun laittaa ”Lisää uusi sivu”-sivu omaksi nimikkeeksi valikkoon. Tämä olisi helppokäyttöisyyden ajatuksesta varsin perusteltua, mutta vaakakupissa painaa myös se, että se olisi yksi lisä valikkoon ja pienemmillä näytöillä se vie lisää tilaa. Tämä sivu on valittavissa yhden klikkauksen päästä, joten se ei kovin paha asia ole mielestäni. Sisällönhallinnassa saa hyvän kuvauksen sivujen määrästä ja niiden laadusta, joten se laittaa väkisin miettimään mitä seuraavaksi kannattaa tehdä. Täten voi yhden turhan sivun tekeminen jäädä väliin.

Tietoturvallisuuden takia on hyvä aina kirjautua ulos sovelluksesta. Siksi on hyvä, että uloskirjaus-mahdollisuus on näkyvällä paikalla valikossa

Navigoinnin oikealla laidalla lukee käyttäjä ja käyttäjän käyttämä tunnus. Esimerkiksi ”Käyttäjä: Tuomo”. Klikkaamalla tätä voi käydä vaihtamassa salasanan.

Kun käyttäjä kirjautuu sovellukseen sisään, on hyvä kertoa käyttäjälle, että hän on kirjautuneena järjestelmässä sisällä. Käyttäjän nimen kertominen on hyvä tapa edesauttaa ymmärrystä, että käyttäjä on todella kirjautuneena sisään sovellukseen. Kuten luvussa kaksi käsiteltiin, käyttäjän on hallittava tilanne täysin.

6 POHDINTA

CMS-sovelluksen rakentaminen on ollut opettavainen kokemus. Olen oppinut uusia asioita ja ohjelmointitaidot ovat parantuneet huomattavasti. Sovelluksen kehittämisessä otin mallia useasta eri lähteestä ja mukautin sitä juuri tälle sovelluksen sopivaksi. Tässä opinnäytetyössä sovelluksen toteuttamisessa kiinnitettiin erityisesti huomiota siihen, että sovellus on mahdollisimman helppokäyttöinen. Tässä työssäni kerroin myös teknisistä ratkaisuista.

Olen oppinut käytettävyyden suunnittelua ja toteutusta. Kirjallisuudesta olen saanut viisautta käytettävyyden toteuttamiseen. Olennainen huomio mitä olen oppinut on se, ettei itse osaa ottaa aivan kaikkea huomioon helppokäyttöisen sovelluksen teossa. Käytettävyystestaus auttaa näkemään asiat myös muiden ihmisten silmin. Muut ihmiset huomaavat sellaista, mitä itse ei huomaa. Testaus tuo arvokasta näkökulmaa ja auttaa jatkokehittämään sovellusta entistä paremmaksi ja käyttäjäystävällisemmäksi. Olen huomannut, että tekniikoilla on väliä käytettävyyden katsontakannalta katsottuna. Siksi esimerkiksi päädyin Bootstrapiin koska se mukautuu hyvin näytön koon mukaan. Kokonaisuudessa kaikella on oma merkityksensä.

Olen esitellyt sovelluksen toimeksiantajalle ja hän on hyvin tyytyväinen sovellukseen. Hän on testannut sovellusta ja osaa käyttää sitä. Sovellus on suunniteltu tavallisille ihmisille jotka eivät ole kovin edistyneitä tietotekniikan käyttäjiä ja toimeksiantaja on sellainen, joten katson että tämä projekti on onnistunut tavoitteessaan.

Sovelluksen ykkösversio on nyt valmis, mutta se tuskin tulee olemaan viimeinen sovelluksen versio. Tulen jatkokehittämään sovellusta myös jatkossa. Sovellus on tarkoituksella tehty yksinkertaiseksi ja vaatimattomaksi. Jos tarvetta on, voidaan sovellusta myös laajentaa. Koska toteutus on tehty yksinkertaisesti, ei laajennusosien lisääminen ole kovin vaikeaa. Ykkösversio soveltuu mielestäni hyvin toimeksiantajalle hänen käyttötarpeeseensa. Silti minulla jäi vähän hampaankoloon muutaman ominaisuuden kanssa. Sovelluksen pääsynhallintaan on olemassa tietoturvallisimpia vaihtoehtoja ja aion tutkia ja kokeilla miten se olisi hyvä

toteuttaa. Valokuvien lisäyksen ja linkkien luomiseen olen ajatellut kokeilla parempia vaihtoehtoja.

Tällä hetkellä CMS-sovellus sopii hyvin pienelle ja yksinkertaiselle nettisivulle. Jos tulee tarvetta isomman nettisivun CMS-sovellukselle, näkisin että sekin on mahdollista toteuttaa tälle sovelluksella, jos tehdään muutamia muutoksia. Esimerkiksi sivun status -luokittelua voidaan muokata ja lisätä eri nimikkeitä. Näin saadaan tehtyä erilaisia alasivutyyppejä, joita usein on isommilla sivulla.

LÄHTEET

Cabral, S. & Murphy, K. 2009, MySQL Administrator's Bible, John Wiley & Sons Incorporated.

Flynn,P. 2014. Human interfaces to Structured documents, University College Cork, Viitattu 02.05.2018 https://cora.ucc.ie/bitstream/handle/10468/1690/Human-Interfaces-to-Structured-Documents.pdf#page=69.

Geerlin, J. 2015. Viitattu. 25.04.2018 https://www.jeffgeerling.com/blogs/jeff-geerling/benchmarking-drupal-8-php-7-vs-hhvm.

Getbootstrap.com 2018. About. Viitattu: 28.04.2018 https://getbootstrap.com/docs/3.3/about/.

Gibbs, S. 2016. Mobile web browsing over-takes desktop for the first time.

(2016). The Guardian. Viitattu 28.04.2018 https://www.theguardian.com/technology/2016/nov/02/.

Gilhooly, K. 2005. Biometrics: Getting Back to Business. Computerworld. Viitattu 27.04.2018 https://www.computerworld.com/article/2556096/security0/biometrics--getting-back-to-business.html.

Johnson-Laird, P. 1983. Mental Models: Towards a Cognitive Science of Language, Inference, and Consciousness. Cambridge, Iso-Britannia: Cambridge University Press.

Jin, K. Teaching Responsive Web Design to Novice Learners. Sigite, K. 2017. 17, Proceedings of the 18th Annual Conference on Information Technology Education, New York, Viitattu 28.04.2018 https://luc.finna.fi/lapinamk/ . ACM.

Kananen J. 2012. Kehittämistutkimus opinnäytetyönä. Jyväskylän ammattikorkeakoulun julkaisuja 134.

Laukkarinen, R. 2018. Älä hanki mitä tahansa webhotellia WordPress-sivuillesi. Jyväskylä: Dude.fi. 21.3.2018. viitattu 28.04.2018 https://www.dude.fi/ala-hanki-mita-tahansa-webhotellia-wordpress-sivuillesi.

Lehdonvirta P & Korpela J. 2013. HTML5 sovellusalustana, Helsinki: RPS-yhtiöt.

Markoff, J. 2007. The Real History of WYSIWYG. New York Times 18.10.2007. Viitattu 28.04.2018 https://bits.blogs.nytimes.com/2007/10/18/the-real-history-of-wysiwyg/?_r=0.

Matbouli, H. & Gao, Q. 2012. An Overview on Web Security Threats and Impact to E-Commerce Success. IEEE. Viitattu 29.04.2018 http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6216645.

MySQL. 2018. Viitattu 16.04.2018 https://www.mysql.com/.

NicEdit.com. Viitattu 17.04.2018 http://nicedit.com/.

Nielsen, J. 1995. 10 Usability Heuristics for User Interface Design. Nielsen Norman Group. Viitattu 03.05.2018 https://www.nngroup.com/articles/ten-usability-heuristics/ .

Nielsen, J. 2000. WWW suunnittelu. Helsinki: Oy Edita Ab.

Norman, D. 1988, The Design of Everyday Things. New York, Yhdysvallat: Basic Books.

PHP.net. 2018. What is PHP?. Viitattu 24.04.2018 https://secure.php.net/manual/en/intro-whatis.php.

Sarja J, 2012. HTMLn perusteet. Otavan Opisto. Mikkeli. Viitattu 02.05.2018 http://opinnot.internetix.fi/fi/muikku2materiaalit/muut/ammatillinen/web/html/html_perusteet.pdf .

Sinkkonen, I. Nuutila, E. & Törmä, S. 2009. Helppokäyttöisen verkkopalvelun suunnittelu. Helsinki: Tietosanoma Oy.

Sucuri.net. 2016. Viitattu 27.04.2018 http://sucuri.net/.

Textwrangler 2018. Viitattu 30.04.2018 https://www.barebones.com/products/textwrangler/ .

Trunde, H. & Weippl, E. 2015. WordPress security: an analysis based on publicly available exploits. iiWAS `15 Proceedings of the 17th International Conference on Information Integration and Web-based Applications & Services. Article No. 81. Viitattu 27.04.2018 https://luc.finna.fi/lapinamk/ ACM Digital Library e-journals.

Whitehead C. 2006. Evaluating Web Page and Web Site Usability. Columbus State University. Viitattu 18.02.2018 https://luc.finna.fi/lapinamk/ ACM Digital Library e-journals.

Williams H. Lane D. 2002. Web Database Applications with PHP and MySQL. Sebastopol. Ca. Usa: O´Reilly & Associates, Inc.

Wordpress.org. 2018. WordPress Optimization/Caching. Viitattu 27.04.2018 https://codex.wordpress.org/WordPress_Optimization/Caching.

W3C 2012. Media Queries - W3C Recommendation 19 June 2012. Viitattu 30.04.2018 https://www.w3.org/TR/css3-mediaqueries/.

W3techs 2018. Historical yearly trends in the usage of content management systems for websites. Viitattu 27.04.2018 https://w3techs.com/technologies/history_overview/content_management/all/y.

LIITTEET

Liite 1 Käytettävyystestin kysymykset

Liite 1. Käytettävyystestin kysymykset 1(3)

Käytettävyystestin kysymykset:

Kuinka vanha olet?

- 0-20
- 21-35
- 36-50
- 51-69
- Yli 50v

Määrittele tietotekninen osaamisesi:

- Olen IT-alan ammattilainen
- Erinomainen tietotekninen osaaminen: Perus tietokoneen käytön lisäksi onnistuu jonkunlainen edistyneempi säätäminen
- Kohtalainen tietotekninen osaaminen: Osaan käyttää sähköpostia, sosiaalinen media, verkkopankkia. Pelien pelaaminen onnistuu
- Hyvä tietotekninen osaaminen: Sähköposti, sosiaalinen media, verkkopankki ja muut verkkopalvelut. Ohjelmien asennus onnistuu.

Osasin lisätä uuden sivun

- Kyllä
- En

Sovellus on

Väittämät:

- Kevyt käyttää
- Helppokäyttöinen

Liite 1. Käytettävyystestin kysymykset 2(3)

- Uskottava

Vastausvaihtoehdot:

- Täysin samaa mieltä
- Jokseekin samaa mieltä
- Osittain eri mieltä
- Täysin eri mieltä

Anna arvosana sovelluksen yleisestä käytettävyydestä

- 1 - 10

Anna arvosana sovelluksen ulkoasusta

- 1-10

Käyttäjäkokemus toimintojen suorittamisessa

Väittämät:

- Uuden sivun lisääminen oli mielestäni helppoa
- Sivujen muokkaaminen oli mielestäni helppoa
- Sivun poistaminen oli mielestäni helppoa

Vastausvaihtoehdot:

- Täysin samaa mieltä
- Jokseekin samaa mieltä
- Osittain eri mieltä
- Täysin eri mieltä

Käytettävyystestin kysymykset 3(3)

Tein testin seuraavalla laitteella?

- Älypuhelin
- Tablettitietokone
- Läppäri/pöytätietokone

Sovelluksessa eteneminen ja navigointi eteenpäin. Miten se omasta mielestäsi sujui?

- Selkeää
- Melko selkeää
- Hetken mietittyäni osasin navigoida eteenpäin
- Navigointi ja eteneminen sovelluksessa oli vaikeaa
- Other:

Sivun otsikko jonka loin ja jota en poistanut

- Tekstikenttä

Vapaa sana: Terveisiä sovelluksen kehittäjälle. Mikä oli sovelluksessa vaikeaa käyttää? Kehitysideoita otetaan myös vastaan.

- Tekstikenttä