Juha-Tapio Anttonen

WEB-SIVUSTO SISÄLLÖNHALLINTAJÄRJESTELMÄLLÄ YRITYKSELLE

Tietojenkäsittelyn koulutusohjelma
Ohjelmistotuotannon suuntautumisvaihtoehto
2011

samk
Satakunnan ammattikorkeakoulu
Satakunta University of Applied Sciences

WEB-SIVUSTO SISÄLLÖNHALLINTAJÄRJESTELMÄLLÄ YRITYKSELLE

Anttonen, Juha-Tapio
Satakunnan ammattikorkeakoulu
Tietojenkäsittelyn koulutusohjelma
Elokuu 2011
Ohjaaja: Stenfors, Juha
Sivumäärä: 30
Liitteitä: -

Asiasanat: web-sivut, www, cms, sisällönhallinta, hallintapaneeli, hallinta, administrator

---

Hiusstudio Bläk. on vuonna 2009 perustettu yritys. Yrityksen toimintaan sisältyy nykyisin hiusstudion lisäksi myös kauneusstudio ja rakennekynsitukku. Yrityksessä työskentelevät viisi tyttöä itsenäisinä yrittäjinä. Tämän opinnäytetyön aiheeksi muodostui yritykselle toteutettava tietokantapohjainen web-sivusto sisällönhallintajärjestelmällä. Yrityksen tarpeena oli yksinkertainen web-sivusto, johon sisältyi palvelut, hinnat, galleria sekä yhteystiedot. Sivuston päivittämisen helpottamiseksi halusin toteuttaa sivustolle sisällönhallintajärjestelmän. Valmiiden järjestelmien sijasta halusin toteuttaa järjestelmän itse. Lähinnä siksi, että halusin opetella syvemmin tietokannan käyttöä yhdessä ohjelmoinnissa. Aikaisempaa kokemusta tietokannan ja ohjelmointikielen yhteiskäytöstä minulla ei juuri ole.

Sivusto on saanut hyvän vastaanoton ja tytöt ovat olleet tyytyväisiä sivuston ulkoasuun ja käytettävyyteen. Yritys on saanut myös uusia asiakkaita sivuston johdosta ja tytöillä on melkoinen kiire. Se on asia joka miellyttää, koska yritysten web-sivujen päätarkoitus on tuoda yritystä julki kuluttajille ja saada uusia asiakkaita. Toisin sanoen yritysten web-sivut ovat nykyaikainen käyntikortti, josta ilmenee muutakin kuin puhelinnumero. Tunnen onnistuneeni projektissa ja olen myös itse tyytyväinen sivuston ulkoasuun ja käytettävyyteen.

WEB-SITE FOR A COMPANY INCLUDING CONTENT MANAGEMENT SYSTEM

Anttonen, Juha-Tapio
Satakunnan ammattikorkeakoulu, Satakunta University of Applied Sciences
Degree Programme in Information Technologies
August 2011
Supervisor: Stenfors, Juha
Number of pages: 30
Appendices: -

Keywords: web-site, www, cms, content managing, managing panel, managing, administrator

---

Hiusstudio Bläk. is a company founded in 2009. The company's activities include nowadays a hair studio, beauty studio and structurenail wholesale. The company includes five girls self-employed. This thesis topic became the database-based website including CMS. The company had a need in a simple web site, including services, prices, photo-gallery and contact information. In addition to relieve updating the site, I wanted to implement a content management system there. Instead of ready-made systems, I wanted to implement the system myself. That’s because I wanted to learn how to use a deeper base with programming. Previous experience in database and programming together, I do not exist too much.

The site has been well received, and customers have been pleased with the site layout and usability. The company has also gained new customers because of the site and the girls have quite a rush. It is something that appeals, because a company's websites main purpose is to bring the company public and consumers to get new customers. In other words, a company’s web pages are a modern business card, which shows more than just a phone number. I’ve had a successful project and I am also satisfied with the site layout and usability as well.

SISÄLLYS

1 JOHDANTO

Tämä opinnäytetyö käsittelee yrityksen nimeltä Hiusstudio Bläk. web-sivuja. Ystäväni Jenna Korpela kertoi tarvitsevansa yrityksellensä web-sivut, jotka lupasin tehdä. Olin etsinyt opinnäytetyöaihetta jo kuukausia ja lupasin toteuttaa sivut. Otin yhteyttä SAMKiin ja tiedustelujen jälkeen alun perin yksinkertaisista web-sivuista tulikin laajahko web-sivusto hallintapaneeleineen ja samalla tämän opinnäytetyöni aihe.

Toteutin sivuston tietokantaan pohjautuvaksi. Kaikki teksti ja kuvien polut on tietokannassa, josta haetaan tieto sivulle. Sivustossa käytetään tietokantana MySQL:ää, joka on avoimeen lähdekoodiin perustuva tietokannanhallintajärjestelmä. Tietokannan kanssa yhteistyössä käytin PHP-ohjelmointikieltä, joka on helppo ja tehokas palvelinpuolen ohjelmointikieli, sekä työväline dynaamisten web-sivustojen luomiseen.

Työn tavoitteena oli suunnitella ja tehdä Hiusstudio Bläk.:n web-sivut. Sivut tuli olla helppokäyttöiset, kevyet, mutta kuitenkin tyylikkäät ja asialliset käyttää. Sivuston tuli sisältää etusivun, kuvagallerian, palvelut, hinnaston, henkilökunnan tiedot ja yhteydenottolomakkeen. Lisäksi, koska projekti on opinnäytetyöni, eikä pelkästään asiakkaan tilaama projekti, piti suunnitella sivustoon vähän lisämausteita ja ominaisuuksia, jotta aihe saisi SAMKilta hyväksynnän. Näitä ominaisuuksia on muun muassa tietokantapohjaisuus ja CMS- eli sisällönhallintajärjestelmä. Henkilökohtainen tavoitteeni oli viimeistään tässä vaiheessa oppia tietokantojen käyttö web-sivujen tekemisessä ja suunnittelussa, sekä myös PHP:n vuorovaikuttteisuutta tietokannan kanssa. Tästä syystä päätin toteuttaa yksinkertaisen sisällönhallintajärjestelmän itse, vaikka mahdollisuus olisi ollut käyttää monipuolistakin valmista CMS:ää ja upottaa se sivuihin tai rakentaa sivusto valmiin CMS:n pohjalle. Näin opinnäytetyöstä tuli kehittävä ja antaa minulle mahdollisuuden oppia uutta.

2 WWW-SIVUSTON TUOTTAMINEN

2.1 Sivuston rakenteen tuottaminen

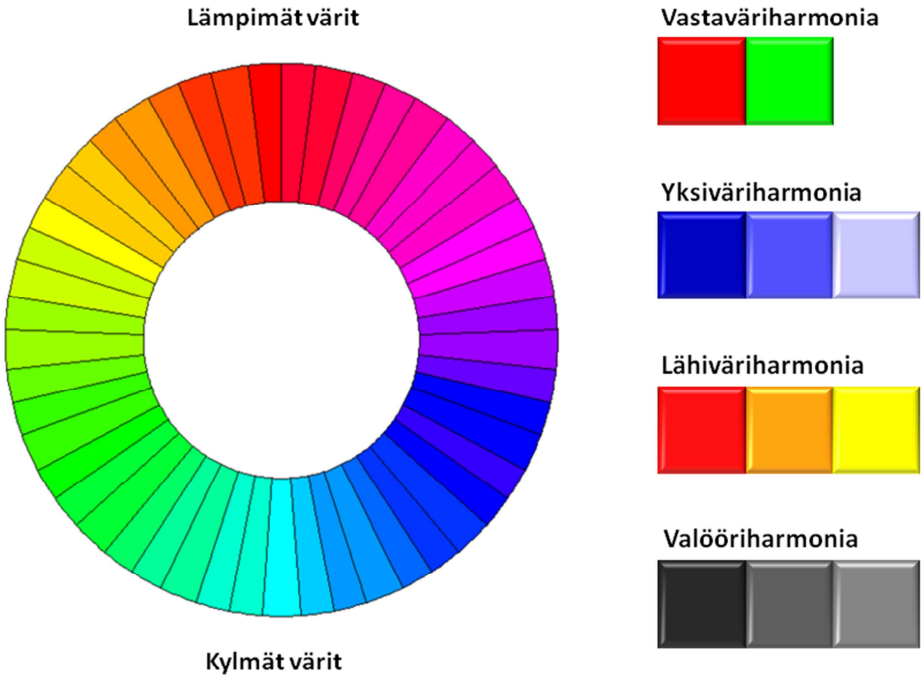
Sivuston rakenne on suoraan verrannollinen tietokantaan. Tästä johtuen helpoin ja tähän tilanteeseen sopivin rakenne on looginen div-elementtien kanssa toteutettu taukkomainen rakenne. Sivuston rungossa on taulua vastaava div-elementti, jonka sisällä on muut div-elementit vastaamassa rivejä ja soluja. Tällöin div-elementtien sisältö on helppo PHP:lla koodaten ladata tulevasta tietokannasta. Rakenteen valmistuttua, on kyseinen sivu index.php. Suunnittelemani rakenne sisältää vakioelementtejä, jotka pysyvät joka sivulla tarvitsematta ladata aina elementtejä uudestaan ja uudestaan. Näitä vakioelementtejä ovat otsikko, navigointi- eli valikkopalkki, vasen ja oikea palsta sekä footer eli ns. alatunniste. Vakioelementtien keskellä on muuttuva div-elementti nimeltä mainContent, jonka sisältö muuttuu käyttäjän valitessa navigointipalkista sivu, jota haluaa tarkastella. Yleisesti sivuston rakenteen tuottamisessa on tiettyjä huomioitavia avainkohtia (Taulukko 1)

Taulukko 1 (Oliver 2004, 191)

Pohdittava asia	**Suositus**
Tekstin sisältö	100-500 sanaa sivua kohti
Tauot tekstissä	Otsikko, viiva tai kuva aina 40-100 sanan välein (paitsi pitkissä artikkeleissa ja tarinoissa)
Sivun pituus	Kahdesta neljään ruutua (640 x 480px resoluutiolla)
Tiedoston koko	Enintään 50kt sivua kohti, kuvat mukaan lukien; animoituja GIF-kuvia voi olla enintään 100kt
Nopeus	Ensimmäinen ruudullinen tekstiä ja tärkeimmät kuvat ilmestyvät alle kolmessa sekunnissa 28.8kbps modeemilla
Värit	Kahdesta neljään hallitsevaa väriä
Fontit	Enintään kolme erilaista fonttia (kuvissa ja teksteissä)
Tyhjä tila	Taustan pitäisi näkyä vähintään puolella sivusta
Kontrasti	Taustaväri ei saa olla liian lähellä tekstin väriä
Sävy ja tyyli	Kaikki teksti ja kuvat sopivat sävy sävyyn ja aiheeseen
Yleisvaikutelma	Sivun pitäisi olla kokonaisuutena tasapainoinen ja puoleensavetävä

2.2 Värien käyttö

Värien käytössä ei ole yhdentekevää mitä värejä käyttää ja miksi. Yleisesti ottaen sivustolle valitaan päävärit joiden kautta suunnitellaan sivuston värimaailma (Kuva 1). Päävärit vaikuttavat eri ihmisiin eri tavalla ja eri kulttuureissa eri väreillä on eri merkityksiä. Värien käytön oppaita ja jopa valmiita selainpohjaisia värimaailman suunnitteluohjelmia löytyy useita verkosta. (Oulun seudun ammattiopisto 2006)



Kuva 1: Väriympyrä sekä erilaisia väriharmonioita (Hypermedia 2008)

Lähiväriharmonialla tarkoitetaan harmoniaa, joka perustuu väriympyrän lähivärien, niiden erilaisten kylläisyys- ja valööriasteiden yhdistelyyn. Tällaisia yhdistelmiä löytyy usein luonnosta ja ne ovat usein miellyttäviä silmälle. (Hypermedia 2008)

Yksiväriharmonialla eli monokromaattisella harmonialla tarkoitetaan yhden värin valintaa (esimerkiksi punainen) ja vaihdellaan sen puhtautta ja valööriä eli mustan ja valkoisen määrää värissä. (Hypermedia 2008)

Valööriharmonian perustana on yhden värin tummuuserojen sopusointuinen yhdistäminen. Suurilla lähekkäisillä valöörieroilla voi kiinnittää katsojan huomion; esimerkiksi musta, valkoinen ja kaikki siltä väliltä. (Hypermedia 2008)

Vastäväriharmonialla tarkoitetaan väriympyrässä vastapäätä sijaitsevien värien yhdistelmää – kuten esimerkiksi yhdistämällä punainen ja vihreä tai sininen ja oranssi – tätä kutsutaan *kaksisoinnuksi*. (Hypermedia 2008)

Kolmisointuharmonialla tarkoitetaan harmoniaa, joka syntyy, kun valitaan väriympyrältä kolme väriä, jotka ovat tasasivuisen kolmion kärkien kohdalla yhtä kaukana toisistaan – esimerkiksi päävärit punainen, sininen ja keltainen tai välivärit oranssi, vihreä ja violetti. (Hypermedia 2008)

Nelisointuharmonialla tarkoitetaan kahta vastäväriparia, jotka poimitaan neliön tai suorakaiteen avulla väriympyrältä. Esimerkiksi punainen, sinivioletti, vihreä ja keltaoranssi muodostavat yhden nelisoinnun. (Hypermedia 2008)

Tässä opinnäytetyössä päävärit musta ja valkoinen tulivat käyntikortissa olevasta logosta. Näistä väreistä päädyin käyttämään sivuston ulkoasussa valööriharmoniaa eli musta, valkoinen ja kaikki harmaan sävyt näiden väliltä. Värimaailman elävyys tulee sivuilla olevista kuvista.

2.3 Grafiikka ja taustakuvat

Grafiikan ja taustakuvat toteutin edellä mainitusta käyntikortin logosta musta-valkoharmaita värejä ja sävyjä yhdistellen. Sivuston taustakuvan ja otsikkoelementin eli headerin taustan toteutin yhteneväisellä tyylillä käyttäen logoa epäkeskosommittelulla. Lisäksi headerin taustaan tein GIF-animoinnin tuomaan elävyyttä sivulle. Vaihtoehto GIF-animaatiolle oli flash-animaatiolla toteutettu header. Tämän karsin toteutuslistalta, koska flash ei ole sataprosenttisesti tuettu joka koneella, kuten GIF.

2.4 Kuvista yleisesti

Yleisesti web-sivulla olevien kuvien muodot kannattaa miettiä käyttötarkoituksen mukaan. Valokuvat esimerkiksi suositellaan tallennettavan JPG-tiedostona, jolloin kaikki värit ovat tuettuina, lisäksi yleensä suuret kuvat on mahdollista pakata pienemmiksi, jolloin sivuista ei tule raskaat. Kevyet, pienet, tasaväriset pinnat tai kuvat, joissa tarvitaan läpinäkyvyyttä, vaihtoehdot ovat joko häviöttömät 256 värin GIF tai JPG:n ja GIF:n yhdistelmä PNG. (Verkko-opas 2010). Käytän sivustolla PNG- ja JPG-tiedostomuotoja, pois lukien header –eli otsikkotaso, johon käytin GIF-kuvaa animaation vuoksi.

3 ADOBE DREAMWEAVER WEB-SIVUJEN TOTEUTUKSESSA

3.1 Yleistä Dreamweaverista

Adobe Dreamweaver on suosittu, mutta kotikäyttöön kohtalaisen kallis HTML-editori, jota käytetään www-sivujen suunnitteluun, toteutukseen ja ylläpitoon. Dreamweaver sisältää tuen seuraaville kielille: HTML *(Hyper Text Markup Language)*, CFML *(Cold Fusion Markup Language)*, PHP *(PHP: Hypertext Preprocessor)*, ASP VBScript, XSLT *(Extensible Stylesheet Language Transformations)*, CSS *(Cascading Style Sheets)*, JavaScript ja XML *(eXtensible Markup Language)*.

Dreamweaver mahdollistaa www-sivujen tekemisen jopa ilman koodaamistaitoja graafisesti ohjelman työkaluilla (Lyytikäinen, Mäkitalo 2004, 2). Dreamweaver helpottaa ja yksinkertaistaa kuitenkin jopa web-ohjelmointitaitoisen koodaajan sivujen luomista ja ylläpitoa kuten tässäkin opinnäytetyössä voidaan todeta.

3.2 Dreamweaverin hyödyntäminen sivuston toteutuksessa

Projektisivuston toteutin Adobe Dreamweaver-ohjelmalla. Olen hyödyntänyt ohjelmasta muun muassa automaattista ftp-yhteys- eli tiedostonsiirto-ominaisuutta (File Transfer Protocol), joka helpottaa sivujen tekemistä huomattavasti. Myös ulkoasun suunnittelu on sujunut ohjelmalla kätevästi hyödyntämällä niin sanottua split-käyttöliittymää, jossa muokkausikkuna on jaettu puoliksi koodinäkymään ja puoliksi designer- eli ulkoasunäkymään.

3.3 Vaihtoehtoiset toteutustavat

Vaihtoehtoisesti mahdollisuus olisi ollut käyttää Adobe Dreamweaverin WYSIWYG eli "What You See Is What You Get" -työskentelytilaa, joka mahdollistaisi sivujen tekemisen ilman koodin käsittelemistä. Olen kuitenkin tottunut kirjoittamaan sivuja tehdessä koodin alusta loppuun itse, joten WYSIWYG ei tullut kyseeseen.

Adobe Dreamweaver ei myöskään ole ainoa vaihtoehto. On muitakin maksullisia, sekä myös ilmaisia ja kevyempiä vaihtoehtoja karsitummilla ominaisuuksilla kuten muutamia mainitakseni; EditPad Lite, HTML-Kit, EasyHtml ja Notepad++. Olen tottunut käyttämään jo vuodesta 2003 Porin ammattiopistosta Dreamweaver-ohjelmaa, joka silloin tunnettiin nimellä Macromedia Dreamweaver MX. Päädyin siis käyttämään tuttua ja turvallista työkalua.

4 TIETOKANTA OSANA WEB-SIVUSTOA

Tietokanta käsitteenä voi olla mikä tahansa loogisesti yhteenkuuluva ja tallennettava tietokokoelma. Myös tekstitiedostoihin tallennettavat tietokokoelmatkin ovat käytännössä tietokantoja, mutta yleisesti tietokannalta vaadittavia ominaisuuksia ovat seuraavat:

- Tiedon rakenne ei riipu sitä käsittelevistä välineistä. Tällöin on mahdollista tehdä vaikkapa kymmenen vuotta käytössä olleelle relaatiotietokannalle web-käyttöliittymä ilman, että varsinaiseen tietokantaan tarvitsee tehdä muutoksia
- Suurten tietomäärien käsittelyyn tehokkaat välineet, joilla haluttua tietoa voidaan osoittaa suoraan (tiedoston kaikkien rivien läpikäynti, kunnes törmätään oikeaan)
- Hallittu yhteiskäyttö; Saman tietokannan tietoja voi käsitellä usea samanaikainen käyttäjä ja sovellus siten, että ne eivät vahingoita tietoja (tiedostojen lukitus)
- Tietojen suojaus. Tiedon saantia voidaan rajoittaa käyttäjäkohtaisesti monipuolisemmin, kuin tiedostojärjestelmän rajallisilla (rwxr-wr--) mahdollisuuksilla.
- Varmistus. Laajakin tietokanta voidaan varmistaa luotettavasti, vaikka sitä käytetään koko ajan.
- Välineet tietojen oikeellisuuden ja eheyden varmistamiseen; Näin esimerkiksi syötteillä ja käsittelytoimenpiteille voidaan määrittää monenlaisia rajoituksia ja tietojen päivitykset toteutuvat "kaikki tai ei mitään"-periaatteella.
- Jokainen tietoalkio pyritään tallentamaan vain kerran, jolloin ristiriitaisen tiedon tallentumisen riski minimoituu.
  (Rantala 2005, 252)

4.1 Apache HTTP Server

Apache HTTP Server on avoimeen lähdekoodiin perustuva HTTP-palvelinohjelma. Vuoden 2010 aikana Apache nousi 46.6 prosentin osuudesta 59.4 prosentin osuuteen HTTP-palvelinohjelmistomittauksessa ja nyt yli 170 000 000 sivustoa verkossa pyörii Apache:lla (Netcraft 2011).

Apachen vahvuuksia ovat sen nopeus, varmuus ja stabiili eli vakaa palvelinohjelmisto. Historia on myös pitkä verrattuna muihin keskeisiin palvelinohjelmiin. Apachen edeltäjä NCSA:n HTTPd on yksi vanhimmista palvelimista ja vuodesta 1995 lähtien Apachestakin on ollut tarjolla käyttökelpoisia versioita.

4.2 PHP (Hypertext Preprocessor)

PHP on C:n, Javan ja Perlin kaltainen palvelinpuolen ohjelmointikieli, jota käytetään dynaamisten web-sivujen tekemiseen (PHP-Group 2009). PHP:ssä ohjelmakoodi tulkitaan vasta ohjelman suoritusvaiheessa. Muun muassa Perlistä PHP eroaa siinä, että se on kehitetty yksinomaan web-käyttöön.

PHP-koodi upotetaan HTML-koodiin ja erotetaan HTML:stä alkutagilla <?php ja lopetetaan tagiin ?>. PHP:n käyttäminen web-ohjelmoinnissa on turvallista. Koodia ei esimerkiksi näytetä tutkittaessa selaimessa sivuston lähdekoodia. Käyttötapoja PHP:lle on useita. PHP:n avulla voidaan muun muassa käyttää monia tietokantoja, esimerkiksi MySQL:ää jota tässäkin projektissa on käytetty.

4.3 MySQL

MySQL on avoimeen lähdekoodiin perustuva tietokannanhallintajärjestelmä, jolla on useita miljoonia käyttäjiä. MySQL:n käyttäjiä ovat yksityiset henkilöt, jotka käyttävät henkilökohtaisilla sivuilla tietokantaa, sekä isot yritykset, joiden sivustot ovat erittäin tiheässä käytössä.

MySQL:n tärkeimpiä piirteitä ovat nopeus, siirrettävyys, hinta sekä yhteensopivuus minkä tahansa ohjelmointikielen kanssa. Koska MySQL on monisäikeinen palvelin, aloitetaan joka kerta yhteyden muodostuessa uusi palvelinprosessi. Jos prosessi päättyy tai palvelin ylikuormittuu, yksi prosessi sulkeutuu eikä koko palvelin kaadu. Tämä lisää MySQL:n nopeutta. (MySQL 2011).

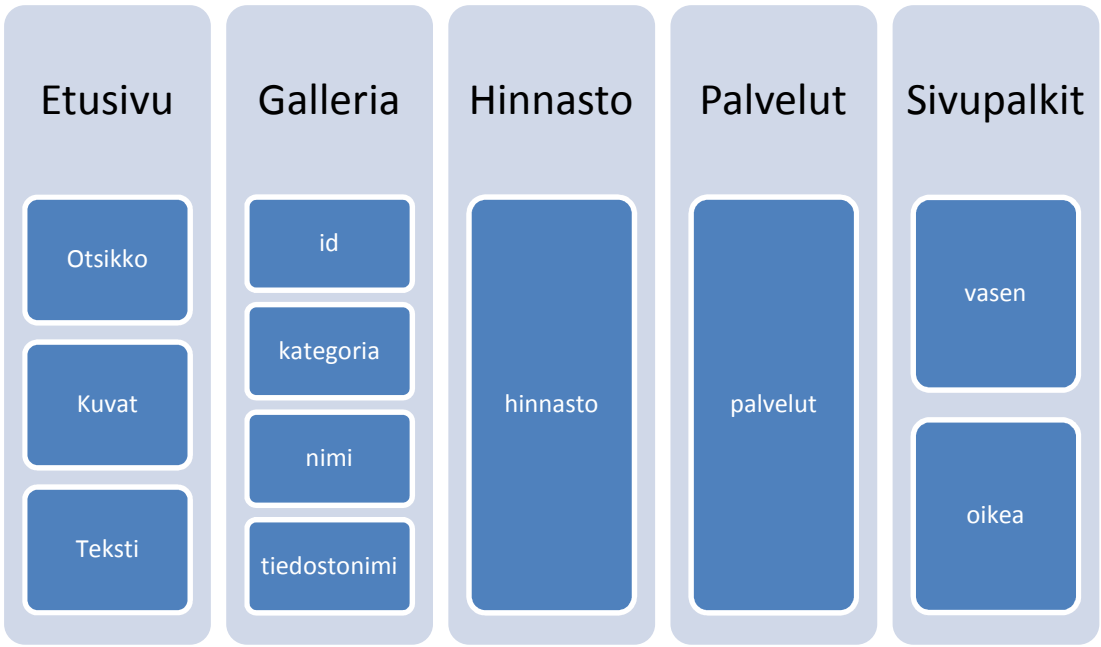
5 HIUSSTUDIO BLÄK. -WWW-SIVUT

Tämän opinnäytetyön aiheena oli web-sivusto yritykselle. Ystäväni Jenna Korpelan yritykseltä Hiusstudio Bläk.:ltä puuttui web-sivut ja minulta kysyttiin halukkuutta toteuttaa ne. Suostuin projektiin ja sain projektia laajentamalla itselleni opinnäytetyöaiheen. Sivusto on tietokantapohjainen ja päivitys tapahtuu pääsääntöisesti sisällönhallintajärjestelmällä, joka on toteutettu ja räätälöity tätä projektia ajatellen. Sivustoa varten tilasin webhotellipalvelun yritykselle. Vaihtoehtoja webhotellille oli useita kymmeniä. Tarvittava palvelun laajuus piti ensin kartoittaa, ja päädyin webhotelli.fi –palveluun josta valittiin Silver-paketti lähinnä tietokannan ja sähköpostiosoitteiden määrän vuoksi. Sivusto oli tarkoitus ensin toteuttaa omalla palvelintilallani, jota vuokraan ystävältäni Linux- ja palvelinasiantuntija Antti Ojalta. Sivuston valmistuttua tarkoituksena oli julkaista ja siirtää valmis sivusto kokonaisuudessaan tietokantoineen web-hotellin palvelimelle.

5.1 Tietokannan suunnittelu

Projektin toteutus lähti tietokannan suunnittelusta. Hyvä pohjatyö takaa toimivan lopputuloksen ja vähentää matkan varrella syntyviä komplikaatioita. Ensin piti suunnitella tietokannan rakenne, taulujen määrä, taulujen sisällöt ja tyypit. Relaatioita ei taulujen välille tässä tietokannassa synny, koska ajatuksena on yksikertaisesti tietokantaan sijoittaa tietoa ja hakea PHP:n avulla tietoa tietokannasta sivulle.

Tauluja tietokantaan tarvitaan jokaiselle hallintapaneelissa muokattavalle sivulle yksi eli tässä tapauksessa viisi: Etusivu, galleria, hinnasto, palvelut ja sivupalkit. Etusivulle tarvitaan sarakkeet otsikko, kuva ja teksti, joita täytyy päästä muokkaamaan. Galleria-tauluun tarvitaan sarakkeet id, kategoria, nimi ja tiedostonimi. Hinnastoon ja palveluihin riittää yksi sarake: hinnasto ja palvelut. Sidebars -taulussa pitää löytyä vasen ja oikea, jotka kuvaavat osuvasti vasenta ja oikeaa sivupalkkia sivustolla.



Kuvio 1: Tietokannan rakenne

5.2 Tietokannan toteutus

Tietokannan toteutukseen käytin yksinkertaista palvelimelta löytyvää phpMyAdmin-sovellusta, joka on ilmainen avoimen lähdekoodin MySQL-tietokannan selainpohjainen hallintatyökalu. phpMyAdmin ei ollut itselleni kovinkaan tuttu ja ensimmäiseksi tuli opetella työkalun käyttöä ja tutustua hieman käyttöliittymään ja mitä mistäkin löytyy. Apunani ja tukihenkilönäni toimi jo aiemminkin mainittu Linux- ja palvelin-asiantuntija Antti Oja, tuttavallisemmin Tera. Sain tarvittavan avun ja tuen tietokantoihin ja tietokannan hallintaan häneltä. Kyseinen aihealue on kuitenkin ollut heikoin osa-alueeni, vaikkakin yksi tärkeimmistä ja edellytettävistä asioista web-ohjelmoinnissa ja yleensä ohjelmoinnissa.

Ensin piti luoda taulut ja tauluihin sarakkeet, joista tärkeintä oli määritellä sarakkeiden tyypit. Pääsääntöisesti tietokannassa on käytetty varchar-tyyppiä ja pituus määritelty 500 merkiksi. Poikkeuksena id-sarakkeet, joissa tyyppinä integer. Hinnastoon ja palveluihin on käytetty mediumtext-tyyppiä johtuen tekstin epämääräisestä määrästä. Tekstin määrä saattaa lisääntyä huomattavasti tai vähentyä huomattavasti. Rajaa ei siis voi määrittää. Lisäksi sivupalkkien vasen-sarake on muista eroten tyypiltään 5000 merkin pituinen varchar. Tämä siksi, koska siinä palkissa ei ole tarkoitus olla koskaan kymmeniä tuhansia merkkejä. Vain muutama yksittäinen tarjous, johon 5000 merkkiä riittää mainiosti. Näillä asetuksilla tietokannan koko pysyy todella pienenä ja käyttö on nopeaa. Varsinaisesti tämän kokoluokan tietokannassa suurtakaan merkitystä ei ole tyyppien koossa, mutta suuremman luokan tietokannassa kokojen määritykset ovat välttämättömiä.

5.3 Sivuston ulkoasun toteutus

Sivuston ulkoasut ja visuaalinen puoli on yleisesti ollut aina vahvin osa-alueeni. Olen koulun projekteissa ja tiimitöissä ollut aina tiimin graafikko ja visualisti. Visuaalinen toteutus lähti yksinkertaisesta etusivun luomisesta, joka sisälsi yksinkertaisesti elementit, joita sivulla tulisi löytyä. Kun etusivun rakenne oli kunnossa, muotoilin sivun CSS-tyylikielellä. Tällä tarkoitetaan muotoilusääntöjä, joilla voi kontrolloida sivun ulkoasua mielensä mukaan. CSS-tiedostoilla saa ulkoasusta tarvittaessa hyvinkin näyttävän. Tyylitiedostoihin voi määritellä Class -tyylejä, joilla voidaan määritellä hyvin yksilöllisesti elementtien luokkia. CSS-tiedostossa nämä määritykset alkavat pisteellä esimerkiksi *.etusivukuva*. Tällöin esimerkiksi kuvan attribuuteissa tulee olla **. Myös Tag-määreitä voi lisätä, jolloin tietyt muotoilut koskevat esimerkiksi otsikkoa H1. Tällöin ei H1-tagissa tarvitse erikseen määritellä class-attribuuttia, vaan CSS-tiedostossa määritellään H1-tagiin aina tietyt muotoilut (Kuva 3). (Lyytikäinen Mäkitalo 2004, 173-178)

Kuten aiemmin mainittu kohdassa 2.2, värimaailman päävärit ovat puhtaasti musta (#000000) ja valkoinen (#FFFFFF) asiakkaan käyntikorttien värityksen mukaisesti. Eloisuutta sivustolle saatiin lisäämällä sivustoon harmaan eri sävyjä ja lisäksi eloa tuovat värilliset kuvat. Sävyt ovat jokainen web-yhteensopivia eli #000000 ja #FFFFFF ovat päävärit, lisäksi harmaat #666666, #333333 ja #c0c0c0 tuovat sävyt eli valöörit koko sivustolle.

5.3.1 Css ja tasot

Tasot ovat (X)HTML-sivun elementtejä, joita voidaan sijoittaa mihin tahansa web-sivulle. Tasot voivat sisältää mitä tahansa sisältöä, joka voidaan sijoittaa HTML-dokumentin Body-osioon. Tasot voit liittää päällekkäin, allekkain, tai syvyysjärjestykseen. Myös tasojen muuntaminen taulukoksi onnistuu. Tasoja on kahdenlaisia; div- ja span-tasoja. (Lyytikäinen Mäkitalo 2004, 178-182)

Div-tasoille olen antanut id-attribuutit, jotka CSS-tyylitiedostossa määritellään alkavaksi merkillä #. Sivulla olen lisännyt esimerkiksi otsikkotason * *, joka taas CSS-tiedostossa määritellään #header -attribuutilla (Kuva 2).

```
#header {
        background-image: url(img/header.gif);
        height: 120px;
        padding: 0 10px 0 20px;
        }
```

Kuva 2: Otsikkotason muotoilu.

Näin on CSS-tyylitiedostoon tehty tyylitaso, jonka sisällä olevia elementtejä pystyy muokkaamaan esimerkiksi seuraavalla tavalla (Kuva 3).

```
#header h1 {
        margin: 0;
        padding: 10px 0;
        }
```

Kuva 3: Otsikkotason alla olevan tagin muotoilu.

Edellä mainitussa esimerkissä (Kuva 3) on tason, id:llä header sisällä olevan H1-tagin tyyliin määritelty marginaalit ja välistykset. Tämä muotoilu ei päde muihin H1-otsikkotageihin, vaan se muokkaa jo aikaisemmin tyylitiedostossa H1:lle asetettua muotoilua (Kuva 4).

```
h1 {
        font: 16px Verdana, Arial, Helvetica, sans-serif;
        padding: 5px;
}
```

Kuva 4: Otsikkotagin H1 –muotoilu.

5.4 Sivuston rakenteen toteutus

Sivuston rakenne piti toteuttaa tietokannan taulujen ja tietokantasuunnitelman mukaisesti (Kuvio 1). Sivuston rakenteen ensiaskel oli luoda runko *index.php*, joka sisältää edellä mainittuja tasoja. Sivun eli BODY-tagin sisällä on taso *container*, jonka tarkoitus on kuvastaa web-sivun kehystä. Container-tason sisällä on tasot *header, menu, sidebar1, sidebar2* sekä *mainContent*.

Halusin index.php-sivun olevan ainoa sivu, joka vierailijalle fyysisesti näkyy. Sivun sisällä PHP-ohjelman avulla muuttujan *$id* arvo määrittelee sen, millaisen osoitteen sivuston vierailija näkee osoiterivillä. Vierailijalle näkyy osoiterivillä *http://www.hiusstudioblak.fi/index.php?id=1* avoinna olevan sivun ollessa *main.php*. Muuttuja *$id* saa arvon aina navigointipalkin linkissä seuraavalla tavalla: *<a href="index.php?id=1" />*. Sivulla index.php on PHP-ohjelma, joka tarkistaa muuttujan arvon ja sisällytettävä sivu määrittyy arvon $id perusteella (Kuva 5).

```
<?php
if (!$id){
	include("main.php"); //Oletuksena sisällytetään sivulle main.php..
}
else //muussa tapauksessa..
	{
	switch ($id) //käyttäjän valinnan mukaisesti..
		{
		case 1: //esimerkiksi $id:n arvon ollessa 1..
			include("main.php");
			break;
		case 2: //tai 2..
			include("gallery.php");
			break;
		case 3: //ja niin edelleen..
			include("services.php");
			break;
		case 4:
			include("prices.php");
			break;
		case 5:
			include("staff.php");
			break;
		case 6:
			include("contact.php");
			break;
		case 7:
			include("thx.php");
			break;
		case 8:
			include("sry.php"); //määritellään sisällytettävä sivu
		}
	}
?>
```

Kuva 5: $id –muuttujan arvon tarkistus ja sivun sisällyttäminen index.php-sivuun.

5.4.1 Rakenne tietokannan pohjalta

Tietokanta määrittelee sivuston sivujen rakenteen ja tasot. Tietokannan taulu kuvastaa sivua, taulussa oleva sarake taas sivussa olevaa tasoa. Tästä johtuen aina avoinna oleva sivu sisältää PHP-ohjelmakoodin, joka hakee tietokannasta kyseisestä taulusta sarakkeiden tiedot ja sijoittaa ne HTML-tasojen sisälle. Näin sivujen muokkaaminen tapahtuu tietokannan kautta ja itse sivuston koodiin ei tarvitse kajota millään tavalla. Sivuston lähdekoodin tarkasteleminen näyttää sivuston vieraalle siis tietokannan taulun sarakkeissa olevat sisällöt koodin sisällä.

5.5 Admin-hallintapaneelin (CMS) toteutus

Hallintapaneelilla tarkoitetaan sisällönhallintajärjestelmää. Sisällönhallintajärjestelmän tarkoituksena voi olla sivuston ylläpito ja luominen hyvinkin laajasti, aina ulkoasuista ja väreistä teksteihin ja koko sisältöön (Tolvanen 2008). Tässä opinnäytetyössä oli tarpeena yksinkertaisesti sisällön hallinta eli kaiken mahdollisen päivittyvän tiedon päivittämisen helpottaminen. Toisin sanoen ettei pienten päivitysten takia tarvitse käydä itse koodiin käsiksi, vaan kirjautua sisään hallintapaneeliin, josta päivittäminen käy käden käänteessä.

Sisällönhallintajärjestelmä olisi ollut mahdollista toteuttaa hyväksikäyttämällä valmiita sisällönhallintajärjestelmiä muokkaamalla sivusto valmiin hallintajärjestelmän päälle tai käyttää hallintaan valmista järjestelmää. Kävin läpi avoimen lähdekoodin vaihtoehdot. Tutkiessani vaihtoehtoja, karsiutui kymmeniä vaihtoehtoja pois, lähinnä monimutkaisuuden ja epämääräisyyksien vuoksi. Karsimisen jälkeen ensimmäinen vaihtoehtoni oli Drupal, joka vaikutti mielenkiintoisimmalta ratkaisulta. Muita vaihtoehtoja oli WordPress, Joomla! ja TextPattern. WordPress järjestelmänä oli enemmän blogi-tyyppinen ratkaisu, joka ei sopinut suunnittelemani sivuston kokonaisuuteen. Joomla!:n ongelma oli taas maksullinen työkalu, jolla ulkoasun muotoilu oli mahdollista. Tästä syystä Joomla! karsiutui pois. TextPattern ja Drupal olivat siis soveltuvat vaihtoehdot.

Halusin kuitenkin itse oppia PHP:n ja tietokannan yhteiskäyttöä ja jätin valmiit järjestelmät käyttämättä. Tietokantoihin olen perehtynyt kursseilla, joissa on lähinnä käsitelty tietokantojen perusteita, kyselyitä ja lähinnä itse SQL-kieltä (Structure Query Language eli Rakenteellinen Kyselykieli). Tietokantaa en ole käyttänyt ikinä käytännössä eli yhdessä jonkin ohjelmointikielen kanssa. Tämä asia oli siis selkeä puute ammattitaidossani ja tiedoissani. Tämän opinnäytetyön yhteydessä sain mahdollisuuden opetella ja täyttämään aukko asiantuntemuksessani.

Toteuttamani hallintapaneeli on hyvin yksinkertainen erillinen sivusto varsinaisen sivuston päällä. Paneelin toteutuksessa korvaamaton apu oli PHP-Manual-sivusto, joka sisältää PHP-käsikirjan englannin kielellä. Sivustolla on eriteltynä PHP-kielen kaikki kirjastot ja funktiot, sekä esimerkit toimivista toteutuksista ja yleisimmistä virheistä. Toteutus alkoi suunnittelusta, jossa määrittelin kynällä paperille ranskalaisin viivoin mitä hallintapaneeli tulisi sisältää.

Sivusto sisältää otsikkotason, navigointipalkin, josta valitaan sivu, jota halutaan muokata sekä sisältötason, jonka sisältö riippuu tietokannan sisällä olevasta tiedosta. Jokainen sivu on itse asiassa erillinen lomake, joka lähetetään PHP-ohjelmalle. PHP-ohjelma muuttaa tietokannan sisältämää dataa käyttäjän valitsemalla tavalla.

Ensin haetaan tietokannan sisällä oleva data ja sijoitetaan jokaisen sarakkeen sisältö omiin muuttujiinsa (Kuva 6). Tämän jälkeen toteutetaan sivu, joka sisältää lomake-objektit tarpeen mukaan. Esimerkiksi textbox- tai textarea-objektiin ladataan kyseisen muuttujan sisältö, jota muokataan tarpeen mukaan. Lisäksi mahdollisesti tiedostoselausobjektilla valitaan tiedosto tietokoneen kiintolevyltä jos halutaan päivittää/lisätä kuva ja päivityksen hoitaa submit- eli kuittausnappiin ohjelmoitu koodirivi joka suorittaa UPDATE-SET SQL-lauseen ja päivittää sarakkeen/sarakkeiden tiedot tietokantaan. Näin sivulla oleva tieto päivittyy automaattisesti, kun tiedot haetaan automaattisesti tietokannasta (Kuva 6).

Tietokantaan yhteydenottaminen oli ensimmäinen ongelma, josta koko toteutus lähti. Piti heti turvautua PHP-käsikirjaan, josta vastaus löytyikin hyvin nopeasti. Päätin tehdä oman tiedoston mysql.php, jossa otetaan mysql_connect()-funktiolla yhteys tietokantaan. Tämän jälkeen ei tarvitse joka sivulla ottaa yhteyttä erikseen, vaan voi include()-funktiolla sisällyttää mysql.php-sivun sisällön sivulle.

```
<?php
include("../mysql.php"); //luodaan tietokantaan yhteys

	$sql = mysql_query("SELECT * FROM `etusivu`"); //luodaan SQL-lause ja asetetaan se muuttujaan $sql

	if (!$sql) {
    die('Invalid query: ' . mysql_error()); //Virheen sattuessa lopetetaan toiminto ja tulostetaan SQL-virheilmoitus
	}

$otsikko = mysql_result($sql, 0, "otsikko");
$kuva = mysql_result($sql, 0, "kuva");
$teksti = mysql_result($sql, 0, "teksti"); //Asetetaan kyselyn tulokset omiin muuttujiinsa sarake sarakkeelta.

?>
```

Kuva 6: Haetaan tiedot tietokannasta ja asetetaan tiedot muuttujiin.

Toinen ongelma tuli vastaan heti päivityksen yhteydessä. Piti osata toteuttaa tietokantakysely PHP-koodissa. Jälleen kerran korvaamaton apu PHP-käsikirja pelasti tilanteen nopeasti ja varmasti. Muokkaussivulla piti submit-nappulaan asettaa tekstien päivitystoiminto oikeilla argumenteilla (kuva 7).

```
<?php
	if (isset($_POST[commit])) { //Jos käyttäjä painaa päivitä-nappulaa..
		$header = $_POST[otsikko]; //Asetetaan inputin sisältö muuttujaan $header..
		$text = $_POST[teksti]; //Ja $teksti..

		$result1 = mysql_query("UPDATE etusivu SET otsikko='$header'")
		or die(mysql_error()); //Tämän jälkeen toteutetaan SQL-lause UPDATE-SET otsikko-sarakkeeseen..

		$result2 = mysql_query("UPDATE etusivu SET teksti='$text'")
		or die(mysql_error()); //ja teksti-sarakkeeseen..
		header('Location: index.php'); //Ja edelleenohjaus CMS-paneelin etusivulle
	}
?
```

Kuva 7: Submit-nappulan tekstien päivitystoiminto.

Ensin yritin saada kuvaa päivittymään tietokantaan samalla koodilla ja samalla napilla, tässä kuitenkaan onnistumatta. Tarpeeksi tutkiessani ongelmaa syvemmältä, totesin, ettei kuvan päivitys tietokantaan onnistu samalla koodilla vaan pitää olla oma upload-nappi kuvaa varten (Kuva 8).

```
<?php
	include("upload.php"); //Lisätään tähän kohtaan sivua upload sovellus
		if (isset($_POST[kuvasubmit])) {
			$result1 = mysql_query("UPDATE etusivu SET ku-
va='$tiedostospeksit[name]'") //Jos käyttäjä on painanut päivitä nappulaa,
toteuta SQL-lause UPDATE-SET
			or die(mysql_error()); //Tai kuole ☺
			echo("<pre>");
			print_r($tiedostospeksit); //Informaatiotulostusta
			echo("</pre>");
		}
?>
```

Kuva 8: Submit-nappulan kuvan päivitystoiminto.

Kuvan päivittämiseen tuli luoda oma ohjelmansa – upload.php), joka siirtää kuvan palvelimelle (Kuva 9). Mielestäni on turhaa keksiä pyörää uudestaan, joten hain verkosta valmiin avoimen koodinpätkän, jota muutin ja sovelsin tähän sivustoon sopivaksi. Tämä kävi varsin helposti PHP-käsikirjan ja PHP-keskustelufoorumeiden avulla.

```
<?php
//ob_start(); //Aloitetaan puskurin täyttö

define("MAX_COUNT", 1);
define("UPLOAD_DIRECTORY", "../img/"); //Määritellään tiedostojen uusi koti
define("MAX_SIZE", 9999999); //Yksittäisen tiedoston maksimikoko
define("INDEX_PAGE", "index.php"); //Määritetään sivu johon edelleenohjataan

if(!is_dir(UPLOAD_DIRECTORY)) {
    mkdir(UPLOAD_DIRECTORY, 0777); //Jos kansiota ei ole olemassa, luodaan
sellainen
}

if(file_exists($_FILES['file1']['tmp_name'])) { //Tarkistetaan että kuva on
tallentunut temp-hakemistoon, aloitetaan lataaminen
    for($i=1; $i<=MAX_COUNT; $i++) {

        if($_FILES['file'.$i]['size'] > MAX_SIZE) { echo "Liian iso tiedos-
to!<br>".MAX_SIZE." on raja"; break; } //Tiedoston ylittäessä maksimikoko,
annetaan virheilmoitus

        if(file_exists(UPLOAD_DIRECTORY.$_FILES['file'.$i]['name'])) {
exit($_FILES['file'.$i]['name']." niminen tiedosto on jo serverillÃ¤"); }
//Jos samanniminen tiedosto on jo palvelimella, annetaan virheilmoitus

        move_uploaded_file($_FILES['file'.$i]['tmp_name'],
UPLOAD_DIRECTORY.$_FILES['file'.$i]['name']); //Tallennetaan tiedosto
serverille
             $tiedostospeksit = $_FILES['file'.$i];
        header("Location: ".INDEX_PAGE); //Redirect eli edelleenohjaus
    }
}
?>
    <form action="<?php echo $_SERVER['PHP_SELF'];?>" method="post"
ENCTYPE="multipart/form-data">
    <INPUT TYPE="hidden" name="MAX_FILE_SIZE" value="102400000">
<?php
for($i=1;$i<=MAX_COUNT;$i++) {
    echo "    <input type=\"file\" name=\"file".$i."\" size=\"35\"><br>";
//Sivulla näkyvä tiedostonhakuobjekti
}

?>    <input type="submit" name="kuvasubmit" value="upload"></form><?php

//ob_end_flush(); //Lopetetaan puskurin täyttö
//exit; //Poistutaan sovelluksesta eli suljetaan funktio
?>
```

Kuva 9: Sovellettu upload-sovellus.

Galleria-sivulle piti luoda oma PHP upload-sovellus, joka lisää kuvan tietokantaan samalla periaatteella, kuin edellä mainitussa funktiossa (Kuva 9). Galleriasivun tapauksessa kuva on tarkoitus lisätä sivulle, ei päivittää. Lisäksi sivulle on tarkoitus lisätä enemmän kuin yksi kuva, jolloin erillinen upload_gallery-sovellus on tarpeen. Sovelluksessa suoritetaan lisäys submit-nappiin ohjelmoidulla INSERT INTO SQL-lauseella (Kuva 10)

```
<?php
	include("upload_gallery.php"); //Sisällytetään upload_gallery
		if (isset($_POST[kuvasubmit])) { //Jos käyttäjä painaa update

			$result1 = mysql_query("INSERT INTO galleria (katego-
ria, nimi, tiedostonimi, kuvaus, isokuva) VALUES ('$kategoria', '$nimi',
'$tiedostospeksit1[name]', '$kuvaus', '$tiedostospeksit2[name]')")
			or die(mysql_error()); //Toteutetaan INSERT INTO-lause
			echo("<pre>");
			print_r($tiedostospeksit1);
			print_r($tiedostospeksit2); //Informaatiotulostusta
			echo("</pre>");		}
?>
```

Kuva 10: Galleria-sivun kuvan lisäys-funktio.

Palvelut- ja hinnasto-sivut ovat hyvin samankaltaiset sivut. Hinnasto-sivulla erona palveluihin on hinnat palveluiden perässä. Tämä kuitenkin haluttiin toimeksiantajan toimesta siellä olevan näin. Palvelut- ja hinnasto-sivujen muokkaus siis ovat käytännössä samat (kuva 11, kuva 12).

```
<?php
	if (isset($_POST[commit])) { //Jos käyttäjä on painanut Päivitä..
		$services = $_POST[palvelut]; //..asetetaan $services-
muuttujan arvoksi palvelut-nimisestä kentästä sisältö..

		$result1 = mysql_query("UPDATE palvelut SET palve-
lut='$services'") //..toteutetaan MySQL-kysely UPDATE-SET
		or die(mysql_error());

		header('Location: services.php'); //Päivitetään sivu eli ohja-
taan uudestaan sivulle services.php, jotta muutokset näkyvät kentissä
		}
	?>
<table align="center">
	<tr>
		<td valign="top">Palvelut:</td>
    <form action="services.php" method="post">
	    <td><textarea name="palvelut" rows="20" cols="50" ><?php echo
"$palvelut"; ?></textarea></td>
	  </tr>
	<tr>
		<td><input type="submit" name="commit" value="PÃ¤ivitÃ¤" on-
click="window.alert('Tiedot pÃ¤ivitetty..!')" /></td>
    </form>
	  </tr>
    </table>
```

Kuva 11: Palvelut-sivun muokkaus.

```
<?php
include("../mysql.php"); //Luodaan tietokantaan yhteys

       $sql = mysql_query("SELECT * FROM `hinnasto`"); //Asetetaan kyselyn
tulos muuttujaan $sql

       if (!$sql) { //Jos kyselyssä tapahtui virhe..
    die('Invalid query: ' . mysql_error()); //Keskeytä ja anna virheilmoitus
       }

$hinnasto = mysql_result($sql, 0, "hinnasto"); //Asetetaan $hinnasto-
muuttujan arvoksi kyselyn tulos.

?>

<?php
       if (isset($_POST[commit])) { //Jos käyttäjä on painanut Päivitä
              $prices = $_POST[hinnasto]; //Asetetaan $prices-muuttujan ar-
voksi hinnasto-nimisen kentän sisältö

              $result1 = mysql_query("UPDATE hinnasto SET hin-
nasto='$prices'") //Toteutetaan kysely UPDATE-SET
              or die(mysql_error());

              header('Location: prices.php'); //Päivitetään sivu lataamalla
uudestaan sivu prices.php
              }
       ?>
<table align="center">
       <tr>
              <td valign="top">Hinnasto:</td>
              <form action="prices.php" method="post">
              <td><textarea name="hinnasto" rows="20" cols="50" ><?php echo
"$hinnasto"; ?></textarea></td>
        </tr>
       <tr>
              <td><input type="submit" name="commit" value="PÃ¤ivitÃ¤" on-
click="window.alert('Tiedot pÃ¤ivitetty..!')" /></td>
    </form>
        </tr>
    </table>
```

Kuva 12: Hinnasto-sivun muokkaus

5.6 Hallintapaneelin käyttäjien hallinta ja suojaus

Hallintapaneelin valmistuttua piti hallinta suojata jollain tavalla. Ensimmäinen vaihtoehtoni oli luoda PHP-sessio, joka alkaa oikean käyttäjätunnuksen sekä salasanan antamisesta ja päättyy kirjautumalla ulos napista painamalla. Toinen vaihtoehtoni olisi ollut sisään- ja uloskirjautuminen evästeiden (Cookies, eli keksit) avulla. Kolmas ja yksinkertaisin tapa oli suojata hakemisto htaccess-salasanasuojauksella. Tutkiessani vaihtoehtoja totesin htaccessin olevan käytännössä tietoturvallisin ja yksinkertaisin ratkaisu salaamiseen ja lisäksi se on vielä erittäin suosittu pienten ja keskisuurien yritysten sivustojen salaamisessa.

Salasanatiedosto sisältää tunnukset ja salasanat muodossa ”käyttäjätunnus:salasana”, joilla on pääsy suojattuun kohteeseen. Salasanatiedostossa olevat salasanat täytyy kryptata. Tähän löytyy useita eri kryptausgeneraattoreita internetistä hakemalla esimerkiksi hakusanoilla ”htaccess password generator”. Kun salasanatiedosto on tehty, täytyy www-palvelimelle kertoa hakemisto, jonka kanssa salasanakyselyä halutaan käyttää. Tämä tapahtuu luomalla tiedosto ".htaccess" suojattavaan hakemistoon (Kuva 13). Kohdan /home/meikalainen/ -kohtaan tulee vaihtaa sivuston kotihakemisto, jossa edellä mainittu salasanatiedosto sijaitsee.

```
AuthType Basic
AuthName 'Yksityinen hakemisto'
AuthUserFile /home/meikalainen/.htpasswd
Require valid-user
```

Kuva 13: Yksinkertainen .htaccess-tiedosto näyttää tältä.

6 AIKATAULU JA ONGELMAT

6.1 Aikataulu

Itse projektin aikataulu oli tiukka. Aikaa projektin valmistumiseen oli noin kaksi kuukautta. Projektin valmistuminen olisi saattanut olla helpompaa, jos olisin pystynyt työskentelemään lähes tauotta – edes päivittäin. Tein töitä päivisin yrittäessäni samalla tehdä iltaisin klo. 19.00 – 20.00 jälkeen projektia. Aina ei ehtinyt eikä pystynyt päivittäin projektia tekemään, joka toi oman haasteensa koko opinnäytetyöhön. Aina jokainen työskentelyhetki käytännössä tarkoitti projektin alusta aloittamista. Jos olisi ollut mahdollista työskennellä ilman pitkiä taukoja, olisi projekti ollut huomattavasti kivuttomampi. Jokainen projektin tekohetki tarkoitti lähes alusta aloittamista ja aina piti asennoitua uudestaan ja tutkia, mihin jäätiin ja muistella mistä tulisi jatkaa.

Vaikeuksista huolimatta aikataulussa pysyttiin ja alle kahdessa kuukaudessa oli projekti käytännössä valmis, eli asiakas sai web-sivustonsa verkkoon. Opinnäytetyön eli tämän opinnäytetyöraportin takarajaksi asetin vaaditun kolme viikkoa ennen kesäkuun valmistumispäivää. Tämä aikataulu kuitenkin koki pakollisen muutaman kuukauden lykkäyksen johtuen Kansaneläkelaitoksen vuositulorajoista ja maksetuista opintorahoista.

6.2 Ongelmat

Projektin alussa kohtasin ongelmia PHP:n ja tietokannan kanssa. Ajattelin ongelmien olevan ensin koodissa, mutta tutkiessani ongelmaa syvemmältä, selvisi kotihakemistossa (public_html) olevan väärät oikeudet. Oikeudet hakemistoon oli 700, jossa ensimmäinen numero tarkoittaa tiedoston omistajan oikeuksia, toinen tiedoston ryhmän oikeuksia ja kolmas muiden oikeuksia. 700 tarkoittaa siis, että tiedoston omistajan oikeudet ovat sekä luku-, kirjoitus ja suoritusoikeus, tiedoston ryhmällä ja muilla käyttäjillä taas ei ole oikeuksia. Oikeudet piti muuttaa 711:ksi, jolloin ryhmällä ja muilla käyttäjillä on suoritusoikeus hakemistoon. Muut mahdolliset arvot ovat 2 kirjoitusoikeus, 3 kirjoitus- ja suoritusoikeus (1+2), 4 lukuoikeus, 5 luku- ja suoritusoikeus (1+4), 6 luku- ja kirjoitusoikeus (2+4) ja lopuksi 7 joka antaa sekä luku-, kirjoitus että suoritusoikeudet. Tässä tapauksessa 711 riittää, koska sivuston käyttäjälle riittää suoritusoikeus PHP-ohjelmiin.

Seuraavat ongelmat projektissa tulivat eteen hallintapaneelin ohjelmoinnissa, jossa sain kerta toisensa jälkeen virheviestin ”headers already sent”. Hain vastausta verkosta ja löysinkin vastauksen ohjelmointiaiheisilta keskustelufoorumeilta. PHP:ssa on funktio, jolla voi muokata headereita. Header taas on erikoisominaisuus http-protokollassa, jolla voidaan lähettää erikoisinformaatiota ennen ensimmäistäkään bittiä PHP-koodia. Virheviesti siis tarkoittaa suunnilleen sitä, että jokin on generoinut headerit aikaisemmin ja nyt PHP yrittää muokata headereita uudestaan rivillä x. Ratkaisin ongelman poistamalla tarpeettomat header()-funktiot koodista tai siirtämällä tarpeelliset funktiot aivan sivun alkuun.

Seuraava suuri ongelma tuli eteen etusivun kuvan lisäysfunktiossa (Kuva 9). Kuvan lisääminen kaikkien merkkien mukaan onnistui, mutta kuvaa ei palvelimelle asti saatu. Virheilmoituksia en saanut esille. Jälleen hain ongelmaan ratkaisua verkosta ja samaisilta ohjelmointiaiheisilta keskustelufoorumeilta tajusin tarkistaa väliaikaiskansion – johon kuva lisätään ennen määriteltyä kansiota – oikeudet, johon palvelimella ei tosiaan ollut oikeuksia. Oikeudet piti muuttaa, jonka jälkeen kuvan palvelimelle siirtyminen onnistui.

7 POHDINTA

Web-sivut nykyisin ovat enemmänkin sääntö kuin poikkeus. Ennen lähdettiin hakemaan yhteystietoja ja yrityksiä lähinnä puhelinluetteloista keltaisilta sivuilta tai soitettiin 118. Tänään lähes poikkeuksetta kuluttajan hakiessa yritystä/yrityksestä tietoa tai yhteystietoja, niin apuväline on Google tai jokin muu hakukone. Siksi on hyvin tärkeää yrityksellä olla omat verkkosivut. Lisäksi sivujen ulkoasu ja käytettävyys tulee olla kunnossa. Käyttöliittymän ergonomia ja toimivuus tulee huomioida sivuja tehdessä. Web-sivujen ergonomiasta ja toimivuudesta puhuttaessa tarkoitetaan käytön helppoutta ja sivujen loogisuutta ja miten, eli kuinka nopeasti sivut toimivat. Liian raskaat sivut eivät ole ergonomiset.

Tulevaisuutta ajatellen voisi ajatella www.hiusstudioblak.fi -web-sivustolla jossain vaiheessa olevan esimerkiksi varausjärjestelmä, josta voisi kalenterista katsoa vapaat ajat ja varata ajan itsellensä. Myöskään jokin verkkokauppasovellus ei välttämättä olisi huono ajatus. Verkkokaupasta voisi ostaa yrityksen tarjoamia oheistarvikkeita, kuten hiusten-, kauneuden- ja kynsienhoitotuotteita.

Projekti kokonaisuudessaan oli kasvattava ja miellyttävä. Ongelmia oli ja niihin piti löytyä ratkaisu verkosta. Opin paljon uutta tätä projektia tehdessäni ja jatkossakin vain ja ainoastaan tekemällä pystyn ylläpitämään taitojani ja tietojani, joita tätä opinnäytetyötä tehdessäni olen oppinut. Sivustoja ja web-sivuja tekemällä oppii joka päivä uutta ja kaikkea ei voi ulkoa toki opetella, ainakaan heti. Web-sivujen tekemisessä ja varsinkin web-ohjelmoinnissa tärkeintä ei ole ulkoa osaaminen vaan tiedon hakemisen taito. Sama toki koskee yleensä ICT-alan töitä ja ohjelmointia. Kaiken kaikkiaan olen tyytyväinen projektiin, sen laatuun ja onnistumiseen. Olen saanut positiivista palautetta sivustosta, sekä asiakkaalta itseltään, että muilta sivustolla käyviltä ihmisiltä. Tulevaisuudessa toimin yrityksen webmasterina, eli sivuston ja sähköpostien ylläpitäjänä.

LÄHTEET

The PHP Group [verkkodokumentti] 2009: "PHP - General Information". Haettu 25.3.2010. Saatavissa http://fi2.php.net/manual/en/faq.general.php.

Lyytikäinen Miikka, Mäkitalo Mauri 2004: Dreamweaver MX 2004. Docendo. Porvoo.

Oliver Dick, 2004: Kotisivut. Edita Prima Oy. Helsinki.

Oulun seudun ammattiopisto [verkkodokumentti] 2006: " Www-sivujen suunnittelu". Haettu 21.4.2011. Saatavissa http://www.okol.org/verkkokurssit/vapaastivalittavat/www_sivujen_suunnittelu/luku5/suunnittelun_teoriaa.htm.

Verkko-opas [verkkodokumentti] 2010: "Kuvat ja grafiikka verkkosivuilla". Haettu 21.4.2011. Saatavissa http://verkko-opas.fi/?id=112.

Rantala Ari, 2005: Web-ohjelmointi. Docendo. Jyväskylä.

Netcraft [verkkodokumentti] 2011: "April 2011 Web Server Survey". Haettu 25.4.2011. Saatavissa http://news.netcraft.com/archives/category/web-server-survey/

MySQL [verkkodokumentti] 2011: "MySQL 5.5 Reference Manual". Haettu 25.4.2011. Saatavissa http://dev.mysql.com/doc/refman/5.5/en.

Hypermedia [verkkodokumentti] 2008: ”Verkkopalvelun sisällöntuotanto”. Haettu 23.8.2011. Saatavissa http://hlab.ee.tut.fi/hmopetus/vpsist.

Perttu Tolvanen [verkkodokumentti] 2008: ”Web-sisällönhallintajärjestelmä”. Haettu 23.8.2011. Saatavissa http://vierityspalkki.wordpress.com/2008/03/31/julkaisujarjestelmat-suomessa-markkinakatsaus-2008/.