Janne Vihermaa

BLOGIJÄRJESTELMÄ

Tietotekniikan koulutusohjelma
Ohjelmistotekniikan suuntautumisvaihtoehto
2010

samk
Satakunnan ammattikorkeakoulu
Satakunta University of Applied Sciences

BLOGIJÄRJESTELMÄ

Vihermaa Janne
Satakunnan ammattikorkeakoulu
Tietotekniikan koulutusohjelma
Toukokuu 2010
Ohjaaja: Karri Kivi
Sivumäärä: 27
Liitteitä: 1

Asiasanat: PHP, SQL, J2ME, WWW-sivustot

Tässä opinnäytetyössä toteutettiin kenen tahansa käyttöön tarkoitettu blogijärjestelmä kaikkine osineen. Järjestelmään kuuluu blogimerkinnät sisältävä tietokanta, WWW-sivusto blogien lukemiseen ja muokkaamiseen, sekä matkapuhelimelle toteutettu merkintöjen lisäys- ja muokkaustyökalu.

Tietokantana käytettiin MySQL-tietokantaa. Itse sivuston toteutuksessa käytettiin PHP:tä ja matkapuhelimelle tehdyssä lisäystyökalussa Java ME -alustaa.

Lisäksi työssä otettiin huomioon tämänkaltaisten järjestelmien tärkeimmät tietoturvariskit. Niihin varauduttiin asianmukaisesti ja pyrittiin estämään niiden hyödyntäminen.

BLOG SERVICE

Vihermaa Janne
Satakunta University of Applied Sciences
Degree Programme in Information Technology
May 2010
Supervisor: Karri Kivi
Number of pages: 27
Appendices: 1

Keywords: PHP, SQL, J2ME, websites

The purpose of this Bachelor's thesis was to create a blog service including all of its components. The system consists of a database containing the blog entries, a website for reading and maintaining the blogs, and a mobile phone application which is used to edit the blog entries.

MySQL was used as the database. The page itself was built using PHP, and the mobile phone application was made with the Java ME -platform.

In addition, the most important security risks in these kinds of services were taken into account. They were addressed appropriately and their exploitation was prevented.

SISÄLLYS

1 JOHDANTO

Tässä opinnäytetyössä rakennettiin kokonaisvaltainen blogijärjestelmä, johon kuuluu MySQL-tietokanta, joka sisältää kaikki käyttäjätiedot, blogimerkinnät ja niille annetut kommentit. Blogia käytetään oman WWW-sivuston välityksellä. Sillä voi ylläpitää omaa blogiaan ja lukea muiden blogeja. Lisäksi työssä tehtiin matkapuhelimelle oma merkintöjen lisäys- ja muokkaustyökalu, jonka avulla omia merkintöjään voi muokata missä tahansa, vaikka tietokonetta ei olisikaan käytettävissä.

Tein työn omatoimisesti pääasiassa sen takia, koska sopivaa opinnäytetyön aihetta ei ollut tiedossa mistään muualta. Lisäksi aiheen etsimiseen ei ollut enää paljoa aikaa, koska tavoitteenani oli saada työ ajoissa valmiiksi ja valmistua alun perin suunnittelemani aikataulun mukaisesti.

Idea työhön sai alkunsa ensinnäkin omasta kiinnostuksestani web-palveluja ja niiden toteuttamista kohtaan. Toisena motivoivana tekijänä oli aikaisempina vuosina saamani tiedustelut muutamalta tutulta, voisinko tehdä heille ja heidän yhdistykselleen kotisivut ja että tätä työtä tehdessäni saisin lisää kokemusta ja tietoa vastaavien tilanteiden varalle.

Tavoitteena oli rakentaa ulkoasultaan ja toiminnaltaan mahdollisimman selkeä ja toimiva kokonaisuus. Tarkoitus oli myös varautua tavallisimpiin tietoturvariskeihin ja sudenkuoppiin asianmukaisella tavalla.

2 BLOGIT

2.1 Taustaa

Blogit ovat tietyntyyppisiä verkkosivustoja, joihin tavallisesti yksi henkilö lisää säännöllisesti omia tarinoitaan, kuvia ja videoita tai kommentteja uutisista ja tapahtumista. Monille blogeille tärkeä ominaisuus on lukijoiden mahdollisuus kommentoida kirjoitettuja tekstejä. Valtaosa blogeista koostuu pelkästä tekstistä satunnaisilla kuvilla höystettynä, mutta osa koostuu lähes pelkästään valokuvista, videoista tai äänitiedostoista. Kaikkien blogien ja bloggaajien yhdessä muodostamaa yhteisöä kutsutaan blogosfääriksi, jonka keskuudessa yleisesti viitataan toisten kirjoituksiin ja kommentoidaan niitä.

Todella suosittujen blogien ylläpitäminen voi olla jopa tuottoisa ammatti, jos niillä on kymmeniä tuhansia lukijoita. Tulot tulevat pääasiassa sivulle upotetuista mainoksista, joita osa lukijoista väistämättä klikkaa.

2.2 Blogityypit

2.2.1 Henkilökohtainen blogi

Perinteisin ja yleisin blogityyppi on yksityishenkilön ylläpitämä ja se muistuttaa tyyliltään pitkälti päiväkirjaa. Kirjoittaja kertoo esimerkiksi omasta elämästään, mielipiteistään ja ajatuksistaan. Tämän tyyliset blogit harvemmin saavuttavat monia lukijoita, vaan ne ovat lähinnä kirjoittajaa itseään varten.

Henkilökohtaisten blogien eräs tyylisuunta on mikroblogi, esimerkiksi Twitter. Niissä kirjoitusten pituudet ovat hyvin lyhyitä, suunnilleen tekstiviestin mittaisia, ja ne käsittelevät pääasiassa kirjoittajan juuri sen hetkisiä tuntemuksia ja tekemisiä. Ne ovat yksi tapa kertoa omista tekemisistään ja kuulumisistaan esimerkiksi omalle lähipiirilleen.

2.2.2 Yritysblogit

Yritysten blogit voivat olla vain yrityksen omaan sisäiseen käyttöön ja viestintään tarkoitettuja yksityisiä blogeja, tai julkisia markkinointi- ym. tarkoituksiin. Yhdistyksillä ja kerhoilla vastaavia blogeja käytetään pääasiassa yhdistyksen toiminnasta ja tapahtumista tiedottamiseen omille jäsenille ja muille asiasta kiinnostuneille.

2.2.3 Tietyn tyylisuunnan mukaiset blogit

Keskittyvät vain yhteen tiettyyn aiheeseen, kuten politiikkaan, matkustamiseen, muotiin tai urheiluun. Näistä yleisimmin käsitellyt aiheet ovat taide ja musiikki.

2.2.4 Käytetyn median mukaiset blogit

Videoblogit koostuvat nimensä mukaisesti käyttäjien sinne lisäämistä videoista. Esimerkiksi YouTube mahdollistaa omien videoblogien pitämisen.
Valokuvablogit koostuvat valokuvista, ja linkkiblogit linkeistä muille sivustoille. Linkitettyjen sivustojen sisältöä on blogissa kuvailtu vain lyhyesti. /1/

3 KÄYTETYT TEKNIIKAT

Järjestelmän kehitysalustana käytettiin Ubuntu Server Edition 9.10 -käyttöjärjestelmää. Päädyin kyseiseen valintaan aikaisempien kokemusteni ja sen helpon ja nopean käyttöönoton vuoksi. Itse palvelinohjelmisto ja muut tarvittavat järjestelmän osat saatiin käyttöön asentamalla ns. LAMP ohjelmistopaketti. LAMP on lyhenne sanoista Linux, Apache, PHP ja MySQL. Näin käyttöjärjestelmään saatiin kerralla kaikki sivuston toiminnan kannalta tarvittavat komponentit ilman jokaisen erillistä asennusta jälkikäteen. Kyseisten komponenttien valinnan perusteena olivat niiden yleisyys, sekä aikaisempi kokemukseni niiden käytöstä, joten ei ollut mitään syytä harkita muiden minulle oudompien vaihtoehtojen käyttämistä. Jokainen näistä tekniikoista on seuraavaksi kuvattuna tarkemmin.

3.1 Apache

Apache on avoimeen lähdekoodiin perustuva HTTP-palvelinohjelma. Sen ensimmäinen versio julkaistiin vuonna 1995. Nykyinen Apachen versio 2 julkaistiin vuonna 2000, ja se on kirjoitettu kokonaan uudelleen edelliseen versioon nähden. Vuodesta 1996 lähtien Apache on ollut suosituin HTTP-palvelin ja 2008 sen on tutkittu olevan käytössä joka toisessa palvelimessa. Apache on saatavilla kaikille merkittäville käyttöjärjestelmille.

Pelkkä Apache yksinään tukee ainoastaan tiedostojen staattista siirtoa HTTP-protokollan välityksellä, mutta sen toimintaa voidaan laajentaa useilla erillisillä moduuleilla. Esimerkiksi tuki PHP:lle saadaan omalla moduulilla. /2/

3.2 PHP

PHP (PHP: Hypertext Preprocessor) on vuonna 1995 alkunsa saanut alustariippumaton ja todella laajasti käytetty skriptikieli, jossa ohjelmakoodi tulkataan vasta ohjelman suoritusvaiheessa. PHP:tä käytetään pääasiassa dynaamisten web-sivujen rakentamiseen. PHP on palvelinpuolen tekniikkaa, eli kaikki ohjelmakoodin suoritus tehdään sivustoa pyörittävällä WWW-palvelimella ja tulokset näytetään loppukäyttäjälle. Tulosteet saadaan web-sivulle näkyville tavallisimmin upottamalla PHP-koodia

HTML-elementtien sisään, tai tulostamalla HTML-elementit kokonaisuudessaan suoraan PHP:lla. /3/

3.3 MySQL

MySQL on relaatiotietokantojen hallintajärjestelmä. Myös se julkaistiin ensimmäisen kerran vuonna 1995. MySQL-tietokanta on todella suosittu erilaisten web-palveluiden perustana. Sitä käyttää mm. Wikipedia, YouTube ja Facebook. Aikaisempina vuosina MySQL:ää on hieman vierastettu ammattikäytössä sen puutteellisten ominaisuuksien takia, mutta uusimpien versioiden myötä ominaisuuksia on tullut lisää ja välimatkaa muihin tietokantajärjestelmiin on kurottu kiinni. MySQL:n suosio web-sovellusten pohjana seuraa melko tarkasti PHP:n suosiota, koska niitä käytetään hyvin usein yhdessä. /4/

3.4 J2ME

Java Platform, Micro Edition on Java-teknologian kevyehkö sovellusympäristö, joka on tarkoitettu sulautettujen ja ominaisuuksiltaan rajoitettujen laitteiden ohjelmointiin. Java SE-alustaan verrattuna merkittävin eroavaisuus on J2ME:n rajoittuneemmat ominaisuudet, joita on karsittu kohdelaitteiden rajallisten resurssien vuoksi. Käytännössä kaikki nykyaikaiset matkapuhelimet tukevat Java-ohjelmia. /5/

4 TIETOTURVA

4.1 Yleistä

Tietoturvan merkitys web-palveluissa kasvaa jatkuvasti. Palvelun ylläpitäjän kannalta on olennaista varmistaa, ettei kukaan ilkeämielinen käyttäjä pysty omilla toimillaan rikkomaan järjestelmää tai varastamaan sieltä tietoja. Tietoturvan toimivuudessa on silti aina kaksi osapuolta. Vaikka järjestelmän turvallisuusominaisuudet olisivat kuinka hyvät, voi tietämätön tai välinpitämätön käyttäjä aina heikentää omaa tilannettaan esimerkiksi valitsemalla salasanakseen hyvin tavallisen ja ennalta arvattavan merkkijonon.
Seuraavaksi on kerrottu mitä turvallisuusseikkoja tätä järjestelmää tehdessä on otettu huomioon ja miten niihin on varauduttu.

4.2 SQL-injektiot

SQL-injektioilla tarkoitetaan sitä, kun SQL-lauseeseen syötetään tietyllä tavalla muotoiltuja merkkejä, ja näin SQL-lause saadaan suorittamaan mahdollisesti vahingollisia toimintoja, kuten esimerkiksi poistamaan tietokantatauluja tai näyttämään käyttäjätietoja sivustolla. Tämän tyyppiset haavoittuvuudet ovat suhteellisen tavallisia eri web-palveluissa, varsinkin hieman kokemattomampien kehittäjien tapauksissa.

Otetaan esimerkiksi SQL-lause *SELECT * FROM users WHERE name = '" + userName + "';*. Tämä lause hakisi tietokannan users-taulusta kaikki käyttäjän userName tiedot. Kysely toimii hyvin niin kauan kuin userName-muuttujalle annetaan normaaleja kirjaimia ja numeroita sisältävä arvo. Mutta jos sille annetaankin arvoksi vaikka *a' or 't'='t*, SQL-lause muuttuu muotoon *SELECT * FROM users WHERE name = 'a' OR 't'='t';*. Tällä lauseella haetaan tietokannasta käyttäjänimeä, joka täyttää ehdon *'t'='t'*, ja koska tämä ehto on aina tosi, voisi se järjestelmän toteutuksesta riippuen mahdollistaa esimerkiksi sivustolle kirjautumisen ilman oikeita käyttäjätunnuksia.

SQL-injektioilta voidaan suojautua yksinkertaisesti lisäämällä SQL-kielen tunnistamien erikoismerkkien eteen ns. escape-merkki eli kenoviiva. Näin saadaan poistettua erikoismerkkien erikoismerkitys. Edellisen esimerkin SQL-lause korjattuna olisi muodossa *SELECT * FROM users WHERE name = 'a\' OR \'t\'=\'t';*. Nyt '-merkkien erikoismerkitys katoaa, ja ne käsitellään tavallisena tekstinä, joten tietokannasta kyseltävä merkkijono olisi käyttäjänimeksi erikoinen *a' OR 't'='t*.

Erikoismerkitysten poistamisessa käytetään PHP:n Magic Quotes GPC-ominaisuutta, joka lisää "\"-merkin automaattisesti kaikkien erikoismerkkien eteen, joita esiintyy Get, Post tai Cookie -tiedoissa. Tämä tekee käyttäjien antamista tiedoista turvallisia, ja ne voidaan lisätä suoraan SQL-lauseeseen.

Toinen tapa saada aikaan sama asia, olisi käyttää PHP:n mysql_real_escape_string-funktiota erikseen jokaiselle käyttäjän antamalle syötteelle ennen niiden liittämistä SQL-lauseeseen. Tämä tapa olisi tietyllä tavalla parempi ja suositeltavampi, mutta Magic Quotes oli nopeampi ja helpompi ottaa käyttöön, joten siksi tässä järjestelmässä päädyttiin käyttämään sitä. /6,7/

4.3 Verkon salakuuntelu

Asia, joka ei välttämättä tule aina kaikilla mieleen, on verkkoliikenteen salakuuntelu. Jos yhteys palvelimen ja oman tietokoneen välillä on täysin suojaamaton, sekä tiedot lähetetään selväkielisessä muodossa, voi tätä liikennettä helposti lukea käyttämällä asiaan soveltuvaa ohjelmistoa, kuten Wireshark. Etenkin avoimia WLAN-verkkoja käytettäessä kannattaa miettiä tarkkaan mille sivustoille uskaltaa kirjautua. Kuka tahansa saman WLAN-verkon käyttäjä voi Wireshark-ohjelmalla vakoilla kaikkea verkon liikennettä ja mahdollisesti löytää muiden käyttäjien salasanoja. Varsinkin moni pienemmän mittakaavan sivusto lähettää käyttäjän kirjautumistiedot salaamattomana verkon yli, jolloin ne ovat helposti luettavissa.

Paras ratkaisu salaamiseen on käyttää salattua HTTPS-protokollaa kaikkeen liikenteeseen. Tämä suojaus otetaan käyttöön web-palvelimen asetuksissa. Lisäksi vaaditaan kolmannen osapuolen myöntämä maksullinen sertifikaatti, joka vahvistaa sivus-

ton luotettavuuden. Jos tämä sertifikaatti puuttuu, selain antaa siitä varoituksen, joka saattaa aiheuttaa hämmennystä käyttäjissä.

Tässä järjestelmässä ongelma on pyritty ratkaisemaan käyttämällä JavaScriptiä ja MD5:n kanssa samantapaista SHA1-salausalgoritmia. Käyttäjän kirjautuessa palveluun, salasanaa ei lähetetä selväkielisenä palvelimelle, vaan se kryptataan ennen lähetystä. Nyt jos liikennettä salakuunnellaan, salasanan tilalla näkyy hyvin kryptinen merkkijono, joka täytyy murtaa oikean salasanan selvittämiseksi. Tämä menetelmä ei kuitenkaan ole täysin aukoton, vaan tarjoaa lähinnä vain ylimääräistä suojaa. Kryptatun salasanan voi lähettää suoraan palvelimelle ja onnistua kirjautumisessa tätä kautta. Se ei tosin onnistu keneltä tahansa, vaan vaaditaan myös hieman tietoa ja osaamista aiheeseen liittyen.

4.4 Salasanat

Erittäin paha virhe, jonka sivuston ylläpitäjä voi tehdä, on säilyttää salasanat selväkielisessä muodossa. Tällöin salasanojen jouduttua vääriin käsiin kaikki käyttäjätunnukset ovat vaarassa. Salasanojen paljastuminen ei välttämättä vaikuta ainoastaan murrettuun palveluun, sillä jos käyttäjällä on sama salasana kuin jossain toisessa käyttämässään palvelussa, hänen käyttäjätunnuksensa on vaarassa myös tässä toisessa palvelussa. Jokaisen palvelun ylläpitäjän tehtävä on taata riittävä tietoturva käyttäjilleen, joiden tulisi tosin itsekin kiinnittää huomiota salasanojensa laatuun ja monipuolisuuteen jokaisella eri sivustolla.

Käyttäjien salasanat ovat tietokannassa MD5-salausalgoritmilla kryptatussa muodossa. Oletetaan, että käyttäjä on antanut salasanakseen sanan ”kissa”, jonka MD5-tiiviste olisi ”1ad99cbe9e425d4f19c53a29d4f12597”. Sana ”kissa” on niin yleinen sana, että suurella todennäköisyydellä sen MD5-tiiviste on laskettu aikaisemmin ja otettu tiivistetietokantaan talteen salasanojen murtamistarkoituksessa. Siinä tapauksessa, että tämän blogijärjestelmän tai muun palvelun tietokanta joutuisi vääriin käsiin, voisi siellä olevia tiivisteitä verrata lukemattomaan määrään jo ennalta laskettuja tiivisteitä. Jos sama tiiviste sitten löytyy molemmista tietokannoista, käyttäjän salasana saadaan selville.

Kyseisen menetelmän estämiseksi tässä järjestelmässä käyttäjän antamiin salasanoihin on lisätty ns. suolaa. Suolalla tarkoitetaan salasanaan lisättyä merkkijonoa. Jos suolaksi valitaan esimerkiksi merkkijono ”1234”, niin se lisättynä varsinaiseen salasanaan voisi olla ”1234kissa”, näin ollen salasanan MD5-tiiviste muuttuu, eikä tätä uutta tiivistettä todennäköisesti löydy ennalta lasketusta tiivistetietokannasta. Jos tässä tapauksessa järjestelmän salasanatietokanta joutuisikin vääriin käsiin, salasanojen murtaminen tällä menetelmällä on lähes mahdotonta.

5 TOTEUTUS JA KÄYTTÖ

5.1 Tietokanta

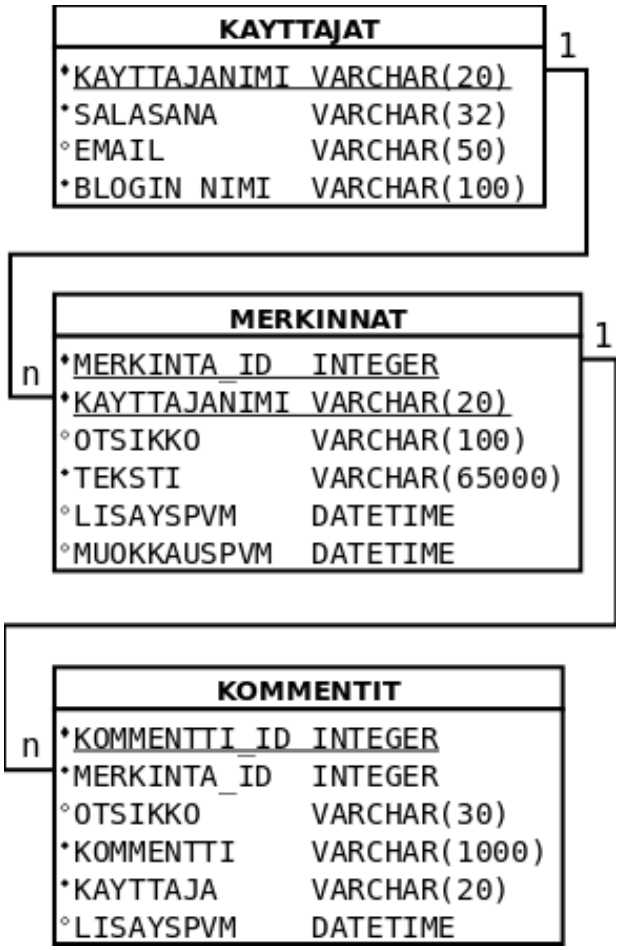
Tietokanta suunniteltiin kohtuullisen yksinkertaiseksi, ja sinne säilötään vain toiminnan kannalta oleellista tietoa. Tietokannassa on kolme erillistä taulua. Rekisteröityneiden käyttäjien käyttäjätiedot sijaitsevat yhdessä taulussa, blogimerkkinnät toisessa ja lukijoiden merkinnöille antamat kommentit kolmannessa taulussa.

MERKINNAT-taulussa oleva KAYTTAJANIMI-kenttä viittaa KAYTTAJAT-taulun samannimiseen kenttään. Viite-eheyden johdannaispoistolla saadaan aikaan se, että jos käyttäjän tiedot poistetaan tietokannasta, poistuu samalla myös kaikki tämän käyttäjän lisäämät merkinnät. KOMMENTIT-taulun MERKINTA_ID-kenttä puolestaan viittaa MERKINNAT-taulun vastaavaan kenttään, eli käyttäjän poistaessa merkintänsä tietokannasta, poistuu myös kyseiselle merkinnälle lisätyt kommentit.

Koska merkintöjen ja kommenttien pääavaimia ei voi mitenkään antaa erikseen jokaiselle uudelle lisäykselle, on näiden yksilöimiseen käytetyille ID-kentille määritelty auto_increment. Tämä tarkoittaa sitä, että kun lisätään uusi merkintä tai kommentti, se saa tunnisteluvukseen aina yhden isomman järjestysnumeron, kuin minkä tietokantaan viimeksi lisätty merkintä tai kommentti on saanut.

Tietokannan merkistökoodaukseksi valittiin UTF8, koska sama valittiin myös blogisivuston merkistökoodaukseksi. Niiden ollessa samat kaikissa järjestelmän osissa, vältytään ajoittain vaivalloiselta ja sekavalta merkkien uudelleenkoodaamiselta merkistöstä toiseen.

Koska tietokannassa käytetään viiteavaimia, tietokantamoottoriksi täytyi määrittää InnoDB, koska MySQL:n oletuksena toimiva MyISAM ei tue viiteavaimia, eikä näin ollen sovellu tähän käyttöön. Tietokannan luomiseen käytetyt SQL-lauseet ovat liitteessä 1.
Tietokannan käsitemalli on esitetty alla kuvassa 1.



Kuva 1. Tietokannan käsitemalli.

Koska tietokannan asetusten ja tietojen muokkaaminen on huomattavasti helpompaa graafisella työkalulla verrattuna komentorivipohjaiseen käyttöliittymään, asennettiin palvelimelle phpMyAdmin-hallintatyökalu. Sitä käytetään selaimen välityksellä, joten tämä mahdollistaa tietokannan hallinnan miltä tietokoneelta tahansa ilman erillisen graafisen hallintatyökalun asennusta kyseiselle tietokoneelle.

Asennuksen jälkeen avattiin hallintatyökalu selaimella ja lisättiin tietokantaan uusi käyttäjä, joka tulee jatkossa hoitamaan kaikki blogijärjestelmän tietokannan muokkaamiset.

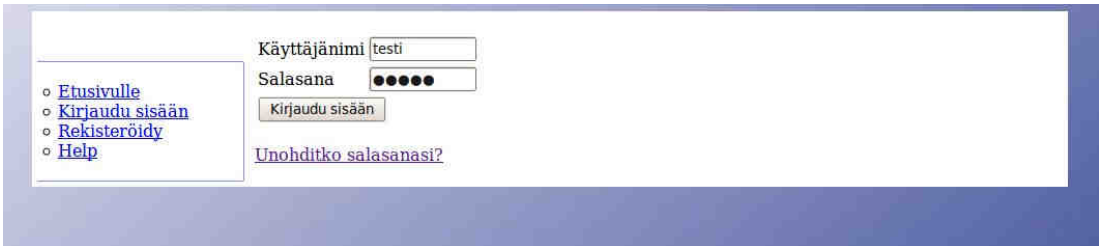
5.2 Käyttöliittymä

Ulkoasua ja käyttöliittymää suunnitellessa pääpaino oli selkeydessä. Kaikkien valikoiden tuli olla selkeillä paikoilla ja värivalinnat sellaisia, että sivustoa on helppo

katsella. Sivustolla navigointi tapahtuu pääasiassa vasemmassa reunassa olevan sivuvalikon avulla.

5.2.1 Kirjautuminen

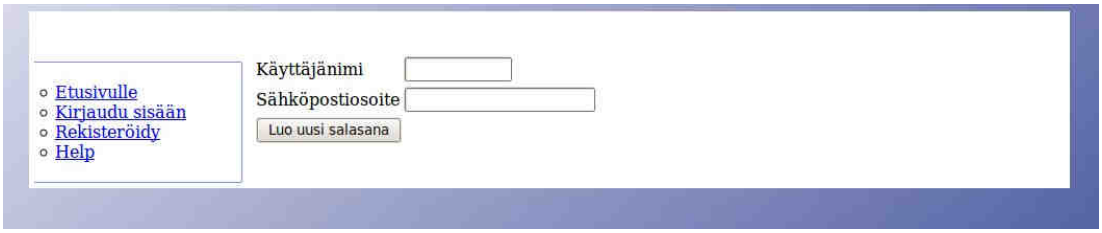
Kirjautumisikkuna on hyvin selkeä ja yksinkertainen. Annetaan vain käyttäjätiedot ja kirjaudutaan sisään. Mikäli oma salasana on päässyt unohtumaan, sen voi palauttaa kuvan 3 lomakkeen avulla.



Kuva 2. Palveluun kirjautuminen

5.2.2 Salasanan palautus

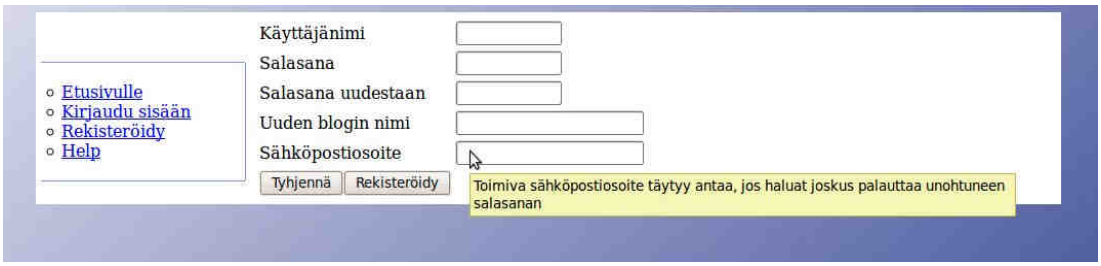
Jos salasana unohtuu, niin tällä lomakkeella voi luoda uuden satunnaisista merkeistä koostuvan salasanan, joka lähetetään omaan sähköpostiosoitteeseen. Lomakkeeseen annetun käyttäjänimen on pakko täsmätä sille ennalta määriteltyyn sähköpostiosoitteeseen. Jos käyttäjä ei ole määritellyt itselleen sähköpostiosoitetta, ei uutta salasanaa voida luoda turvallisuussyistä. Vanhaa salasanaa ei voi palauttaa, koska se on tietokannassa salatussa muodossa eikä sitä saa sen perusteella mitenkään selville.



Kuva 3. Unohtuneen salasanan palautuslomake

5.2.3 Rekisteröinti

Uuden käyttäjätunnuksen ja blogin luonti tapahtuu kuvan 4 mukaisella lomakkeella. Kaikki muut tiedot on annettava, paitsi sähköpostiosoite. Toimivan osoitteen antaminen on silti suositeltavaa, koska ilman sitä unohtunutta salasanaa ei voi palauttaa. Käyttäjätietoja voi muokata vielä rekisteröitymisen jälkeen, joten sähköpostiosoitteen voi antaa myös silloin. Sivusto suorittaa tarkistuksen, että kaikkiin kenttiin on annettu tarvittavat tiedot. Jos tietoja puuttuu tai salasanat eivät täsmää, aukeaa huomautusikkuna ja tiedot on korjattava, jotta rekisteröityminen on mahdollista. Onnistuneen rekisteröinnin jälkeen käyttäjä ohjataan automaattisesti oman bloginsa pääsivulle.



Kuva 4. Käyttäjätunnuksen luonti

5.2.4 Pääsivu

Jokaisen blogin perusnäkymä on kuvan 5 kaltainen. Oletuksena näkyvillä on kymmenen uusinta blogimerkintää. Jos kaikki merkinnät näytettäisiin heti, olisi pääsivun näkymä hyvin nopeasti monen uuden merkinnän lisäämisen seurauksena kohtuuttoman pitkä.

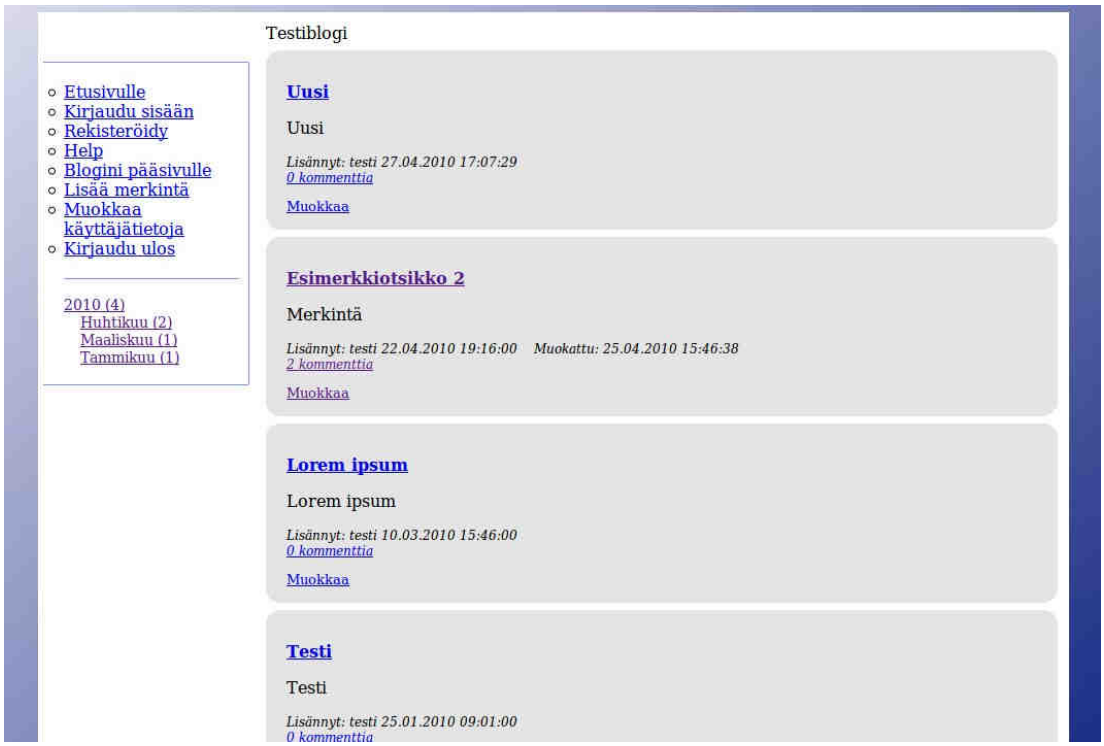
Jos käyttäjä ei ole kirjautuneena, vasemmassa reunassa olevasta navigaatiovalikosta puuttuvat oman blogin ja käyttäjätunnuksen hallinnoimiseen käytetyt linkit.

Navigaatiovalikon alapuolella on näkyvillä lista, jossa lukee kaikki kuukaudet ja vuodet, joina uusia merkintöjä on lisätty, sekä merkintöjen lukumäärä. Valitsemalla vuosiluvun tai kuukauden, näkyviin tulee kaikki kyseisen ajanjakson merkinnät.

Jokaisen blogimerkinnän laatikossa näkyy merkinnän otsikko, itse merkintäteksti, merkinnän lisääjän nimimerkki, lisäys- ja muokkauspäivämäärät sekä merkinnälle annettujen kommenttien lukumäärä. Lisäksi jos käyttäjä on kirjautuneena, näkyvissä on myös Muokkaa-linkki, jota painamalla valittu merkintä aukeaa omaan lomakkee-

seensa, jossa sitä voi muokata tai sen voi poistaa kokonaan. Muokkausnäkymä on esitettynä kuvassa 6.

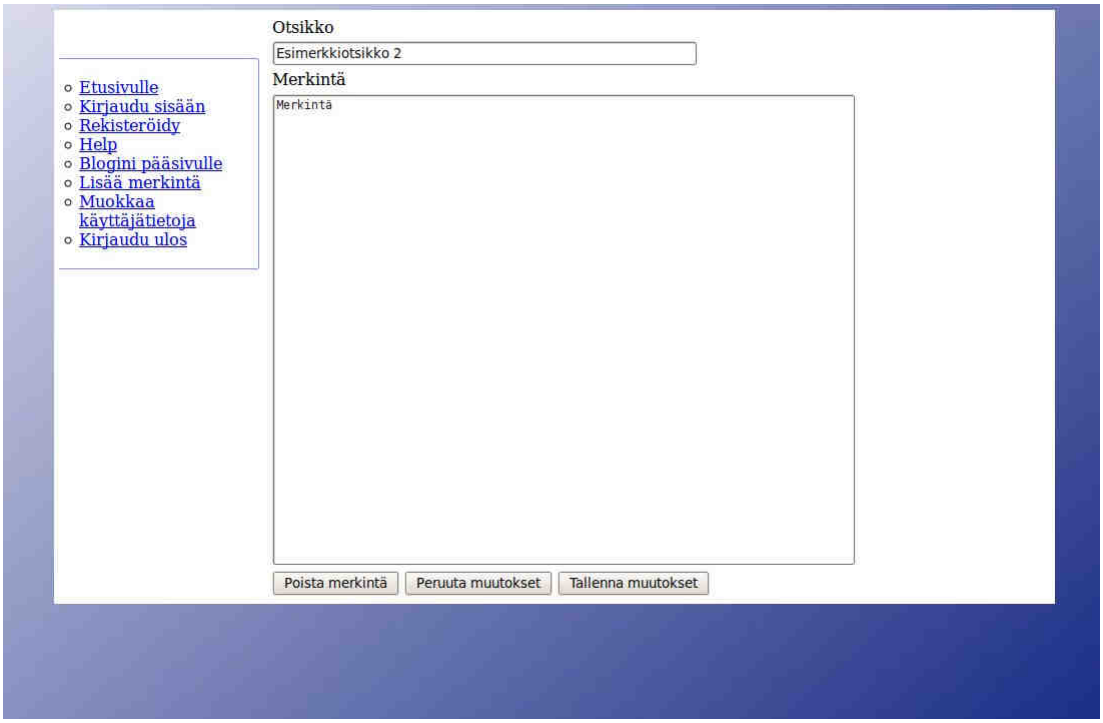
Blogimerkinnän otsikkoa klikkaamalla kyseinen merkintä aukeaa näkyviin yksinään. Sama näkymä aukeaa klikkaamalla kommenttilinkkiä, jossa ainoa ero on se, että kommenttilinkki kohdistaa näkymän kommenttilistan alkuun eikä merkinnän alkuun.



Kuva 5. Blogin pääsivu

5.2.5 Merkintöjen lisäys ja muokkaus

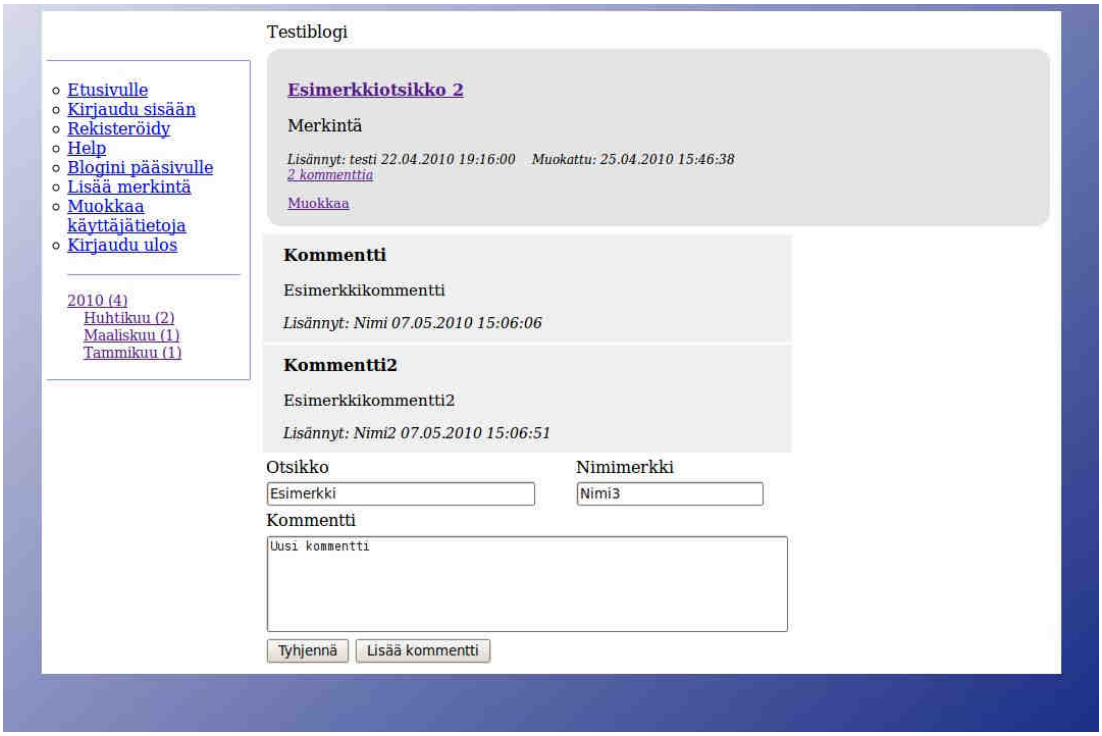
Uusien merkintöjen lisäysnäkymä on alla olevan muokkausnäkymän kanssa muuten identtinen, mutta silloin aukeaa vain tyhjä lomake, johon uuden merkinnän tiedot voi syöttää. Lomake on hyvin suoraviivainen eikä vaadi suurempia selityksiä. Ennen merkinnän poistamista kysytään käyttäjältä vahvistus, halutaanko merkintä varmasti poistaa.



Kuva 6. Merkinnän lisäys, muokkaus ja poisto.

5.2.6 Kommentointi

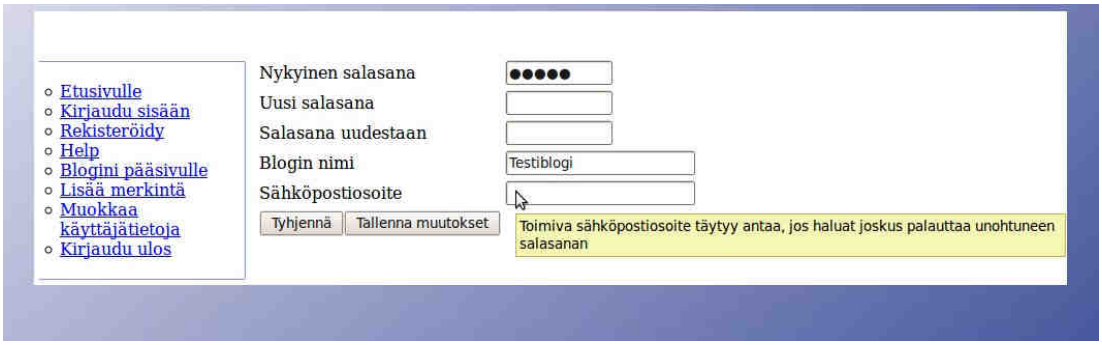
Kommenttien lisäysnäkymä aukeaa klikkaamalla pääsivulla merkinnän otsikkoa, tai kommenttilinkkiä. Ne näkyvät sivulla omissa laatikoissaan itse merkinnän jälkeen, vanhin kommentti ensin ja uusin viimeisenä. Kommentointi on myös hyvin helppoa. Lomakkeen tekstikenttiin annetaan merkinnän tiedot ja oma nimimerkki, jonka jälkeen painetaan Lisää kommentti -painiketta. Mikäli kaikki tiedot oli annettu, kommentti lisätään tietokantaan ja päivitetty näkymä juuri lisättyine kommentteineen tulee näkyviin.



Kuva 7. Kommentointi

5.2.7 Käyttäjätietojen muokkaus

Käyttäjän ollessa kirjautuneena, vasemman reunan navigaatiopalkissa on yhtenä vaihtoehtona käyttäjätietojen muokkauslinkki, jota painamalla näkyviin aukeaa kuvan 8 kaltainen lomake. Lomakkeella voi vaihtaa salasanansa uudeksi, bloginsa otsikkoa tai salasanaa. Kaikkien muutosten tekemiseksi täytyy antaa tämän hetkinen salasana. Tällä varmistutaan siitä, ettei kukaan sivullinen pääse muuttamaan käyttäjätietoja, jos sivusto esimerkiksi unohtuu auki julkiselle tietokoneelle.



Kuva 8. Käyttäjätietojen muokkausikkuna

5.3 J2ME-lisäystyökalu

Toinen tapa lisätä ja muokata oman blogin merkintöjä on matkapuhelimille tehty oma lisäystyökalu. Ohjelmaa voi käyttää merkintöjen lisäämiseen esimerkiksi silloin, kun tietokonetta ei ole käytettävissä. Ohjelma vaatii toimiakseen matkapuhelimelta tuen Java-ohjelmille. Tämä tuki löytyy lähes jokaisesta nykyaikaisesta puhelimesta.

5.3.1 Toteutus

Ohjelman kehityksessä käytettiin Eclipse-kehitysympäristöä ja sen EclipseME-lisäosaa. J2ME on karsittu versio normaalista Javasta, jotta se on saatu pienemmäksi ja kevyemmäksi matkapuhelimia varten. Monien hyödyllistenkin ominaisuuksien karsimisen vuoksi ohjelman kehityksessä täytyi käyttää kiertoteitä tiettyjen asioiden toteuttamiseksi. Selkein rajoite oli tietokantatuen puuttuminen, joten palvelimen tietokantaan ja merkintöihin ei päässyt suoraan käsiksi. Tämä ongelma kierrettiin käyttämällä HTTP-protokollaa ja POST-metodia. Palvelimelle luotiin oma j2me.php-niminen tiedosto, joka toimii välikappaleena tämän ohjelman ja tietokannan välillä. Esimerkiksi uusien merkintöjen lisäys toimii niin, että ohjelma muodostaa yhteyden palvelimeen käyttäen HTTP-protokollaa, kutsuu siellä olevaa j2me.php-tiedostoa ja lähettää sille uuden merkinnän tiedot POST-metodilla. Sen jälkeen j2me.php-tiedosto käsittelee saamansa merkinnän tiedot ja lisää ne palvelimen tietokantaan.

Olemassa olevien merkintöjen lukeminen toimii samalla periaatteella. Ohjelma kutsuu j2me.php:tä, joka palauttaa vastaukseksi tietokannasta lukemansa tiedot muotoiltuna tietyllä tavalla. J2ME-lisäystyökalu lukee tämän vastauksen, poimii siitä merkintöjen tiedot ja tulostaa ne käyttäjän nähtäväksi.

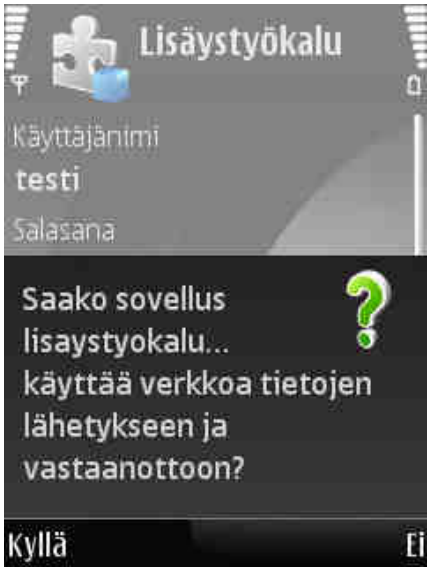
Lisäystyökalulla pystyy ainoastaan lisäämään, muokkaamaan ja poistamaan omia merkintöjään, joten muiden käyttäjien blogien lukeminen ei ole sillä mahdollista.
Lisäystyökalun toteutus onnistui suhteellisen sujuvasti. J2ME oli jo ennestään tuttua tekniikkaa, eivätkä sen rajoitteet olleet lopullisena esteenä ohjelmaan haluttujen ominaisuuksien toteuttamisessa.

Huomion arvoista on, että ohjelma kannattaa asentaa puhelimen omalle sisäiselle muistille eikä erilliselle muistikortille. Ohjelmaa testatessa Nokia 6120 classic -puhelimella muistikortille asennettuna, se käynnistyi heti asennuksen jälkeen hyvin, mutta jos puhelimen sammutti, niin sen uudelleenkäynnistyksen jälkeen ohjelma ei enää suostunut käynnistymään ilman uudelleenasennusta. Tämä ongelma katosi puhelimen sisäiselle muistille asennuksen myötä. Tätä ongelmaa ei ilmennyt muita J2ME-ohjelmia testatessa.

5.3.2 Käyttö

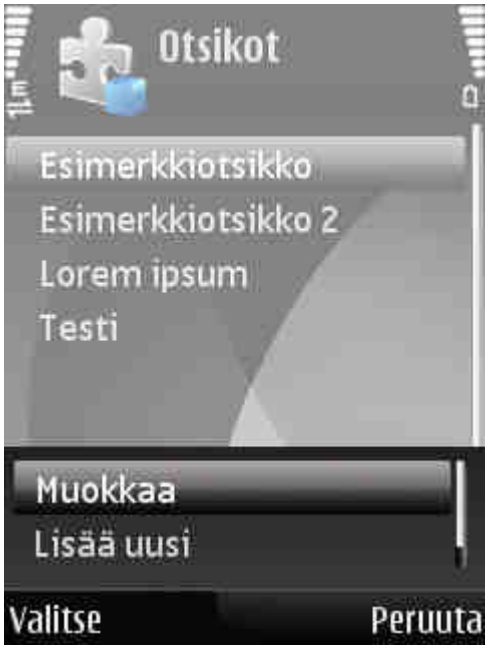
Seuraavaksi käsitellään lisäystyökalun käyttöliittymää ja ominaisuuksia tarkemmin. Ohjelmaa on testattu ja kuvakaappaukset otettu Nokia 6120 classic -puhelimella, joten näyttöjen ulkoasu saattaa vaihdella eri puhelinten välillä.

Aluksi näkyy kirjautumisikkuna, johon syötetään käyttäjätunnus ja salasana. Mikäli lisäystyökalua on käytetty jo aikaisemmin ja on kirjauduttu onnistuneesti, ohjelma lukee matkapuhelimen RMS:sta (Record Management System) käyttäjänimen ja lisää sen Käyttäjänimi-kenttään. Tämä nopeuttaa kirjautumista, koska käyttäjänimeä ei tarvitse aina kirjoittaa uudelleen, ja koska matkapuhelin on yleensä henkilökohtainen, niin lisäystyökalua eivät käytä muut henkilöt ja kirjautumisessa käytetty käyttäjänimi pysyy aina samana. Käyttäjätunnusten antamisen jälkeen puhelin kysyy annetaanko ohjelmalle lupa käyttää verkkoyhteyttä.



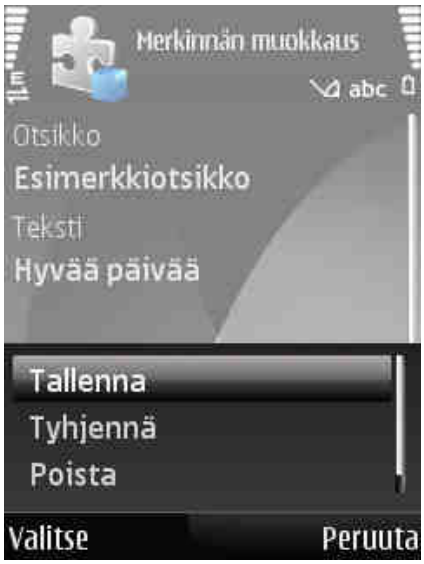
Kuva 9. Kirjautumisikkuna.

Onnistuneen kirjautumisen jälkeen aukeaa näkyviin lista kaikista lisättyjen merkintöjen otsikoista. Listasta voidaan valita merkintä, jota halutaan muokata, tai voidaan lisätä kokonaan uusi merkintä.



Kuva 10. Merkintöjen otsikkolista.

Jos vanha merkintä valitaan muokattavaksi, näkyviin aukeaa uusi ikkuna, jossa näkyy kyseisen merkinnän tämän hetkinen otsikko ja itse merkintäteksti. Näitä voi vapaasti muokata, ja valikon kautta muutokset voi tallentaa ja ne päivittyvät palvelimen tietokantaan. Vanhojen merkintöjen poisto onnistuu myös tämän saman ikkunan valikon kautta. Ohjelma kysyy käyttäjältä varmistuksen ennen merkinnän lopullista poistoa.



Kuva 11. Merkinnän muokkausnäkymä.

Jos otsikkolistan valikosta valitaan ”Lisää uusi”, niin näkyviin aukeaa muokkausikkunan kanssa ulkoasultaan identtinen tyhjä uuden merkinnän lisäysikkuna. Tähän voi kirjoittaa uuden merkinnän otsikon ja tekstin. Valikosta merkinnän voi joko tallentaa tietokantaan, tai tyhjentää siihen kirjoitetut tekstit.

Kuva 12. Uuden merkinnän lisäysnäkymä.

6 YHTEENVETO

Tämän opinnäytetyön tekeminen oli opettavainen kokemus. Opin aikaisempaa syvällisempiä asioita web-sivustojen ja palvelujen tekemisestä, muun muassa mitä turvallisuusnäkökulmia täytyy ottaa huomioon ja mitkä ovat suositeltavia käytäntöjä sivustojen suunnittelussa ja toteutuksessa.

Järjestelmän lopputulos oli sen kaltainen, kuin työn alussa kuvittelinkin sen olevan. Kaikki suunnittelemani ominaisuudet olivat toteuttamiskelpoisia, eikä missään vaiheessa tullut eteen tilannetta, jossa alkuperäiset suunnitelmat olisivat olleet liian puutteelliset, ja tämän takia jonkun ominaisuuden toteutus olisi ollut mahdotonta tai epäkäytännöllistä.
Eräs lisäominaisuus, jonka voisi tulevaisuudessa toteuttaa, on mahdollisuus vaihtaa oman bloginsa ulkoasua ja värimaailmaa. Tavallinen ulkoasu tuskin miellyttää kaikkia käyttäjiä, joten mahdollisuus sen vaihtamiseen voisi olla mieluista.
SQL-injektioiden estämiseen käytetty Magic quotes menetelmä tulee poistumaan tulevaisuudessa uuden PHP 6 version myötä, joten se täytyy jossain vaiheessa korvata jo aikaisemmin mainitulla PHP:n mysql_real_escape_string-funktiolla.

Alun suunnitteluvaihe olisi voinut olla perusteellisempi ja kattaa useampia eri vaihtoehtoja. Se olisi helpottanut ja nopeuttanut järjestelmän varsinaista rakennusvaihetta. Etenkin sivuston ulkoasua olisi kannattanut suunnitella alussa tarkemmin, koska sen muuttaminen on jälkikäteen työlästä. Vaikka ylitsepääsemättömiä esteitä ei matkan varrella tullutkaan vastaan, turvauduin joissain tilanteissa vain paikkaamaan järjestelmän virheitä. Nämä pikaiset paikkaukset saattavat muodostua hidastavaksi ja hankaloittavaksi tekijäksi järjestelmän mahdollisen jatkokehityksen kannalta. Perusteellisempi suunnittelu olisi auttanut ennaltaehkäisemään juuri näitä aukkoja, joita nyt täytyi jälkikäteen paikata.

Jos minun täytyisi kehittää vastaavanlainen järjestelmä uudestaan, niin panostaisin enemmän nimenomaan perinpohjaiseen suunnitteluun. Vanhaa sanontaa lainaten, ohjelmistokehityksessä tuntuu välillä siltä, että hyvin suunniteltu on jo yli puoliksi

tehty. Kehitysvaiheessa ilmenevien sudenkuoppien selvittämiseen ja ratkaisemiseen kuluu helposti enemmän aikaa, kuin mitä vastaavan asian hyvään suunnitteluun.

Lisäksi käytettävän tekniikan puolesta AJAX voisi olla mielenkiintoinen vaihtoehto sivuston toteutustavaksi. Olisin saattanut hyödyntää sitä nytkin jollain tavalla, mutta tämän hetkisen osaamiseni vuoksi katsoin paremmaksi luottaa vanhoihin tuttuihin tekniikoihin, jotka hallitsin jo paremmin.

LÄHTEET

1. Wikipedia, Blog [Verkkodokumentti, viitattu 3.5.2010]
http://en.wikipedia.org/wiki/Blog

2. Wikipedia, Apache [Verkkodokumentti, viitattu 5.5.2010]
http://fi.wikipedia.org/wiki/Apache_%28palvelinohjelma%29

3. Wikipedia, PHP [Verkkodokumentti, viitattu 5.5.2010]
http://en.wikipedia.org/wiki/Php

4. Wikipedia, MySQL [Verkkodokumentti, viitattu 5.5.2010]
http://fi.wikipedia.org/wiki/Mysql

5. Wikipedia, Java ME [Verkkodokumentti, viitattu 6.5.2010]
http://fi.wikipedia.org/wiki/Java_ME

6. Wikipedia, SQL-injection [Verkkodokumentti, viitattu 24.4.2010]
http://en.wikipedia.org/wiki/SQL_injection

7. Wikipedia, Magic quotes [Verkkodokumentti, viitattu 7.5.2010]
http://en.wikipedia.org/wiki/Magic_quotes

LIITE 1

```
CREATE TABLE KAYTTAJAT (
KAYTTAJANIMI        VARCHAR(20) PRIMARY KEY,
SALASANA            VARCHAR(32) NOT NULL,
EMAIL               VARCHAR(50),
BLOGIN_NIMI         VARCHAR(100) NOT NULL
) TYPE=INNODB, CHARACTER SET utf8;

CREATE TABLE MERKINNAT (
MERKINTA_ID         INTEGER PRIMARY KEY AUTO_INCREMENT,
KAYTTAJANIMI        VARCHAR(20) NOT NULL,
OTSIKKO             VARCHAR(100),
TEKSTI              VARCHAR(65000) NOT NULL,
LISAYSPVM           DATETIME,
MUOKKAUSPVM         DATETIME,
INDEX(KAYTTAJANIMI),
CONSTRAINT Kayttajanimi_viite
        FOREIGN KEY (KAYTTAJANIMI)
        REFERENCES KAYTTAJAT (KAYTTAJANIMI)
        ON DELETE CASCADE
) TYPE=INNODB, CHARACTER SET utf8;

CREATE TABLE KOMMENTIT (
KOMMENTTI_ID        INTEGER PRIMARY KEY AUTO_INCREMENT,
MERKINTA_ID         INTEGER NOT NULL,
OTSIKKO             VARCHAR(30),
KOMMENTTI           VARCHAR(1000) NOT NULL,
KAYTTAJA            VARCHAR(20) NOT NULL,
LISAYSPVM           DATETIME,
INDEX(MERKINTA_ID),
CONSTRAINT Kommentti_viite
        FOREIGN KEY (MERKINTA_ID)
        REFERENCES MERKINNAT (MERKINTA_ID)
        ON DELETE CASCADE
) TYPE=INNODB, CHARACTER SET utf8;
```