KEMI-TORNION AMMATTIKORKEAKOULU

Web-puhelinluettelo

Web-sovellus Kolarin kunnan yhteystietojen löytämiseen

Tapani Joki & Jussi Ruokamo & Jari Suunta

Tietojenkäsittelyn koulutusohjelman opinnäytetyö
Web-asiantuntijan suuntautumisvaihtoehto
Tradenomi

TORNIO 2009

TIIVISTELMÄ

Joki, Tapani & Ruokamo, Jussi & Suunta, Jari. 2009. Web-sovellus Kolarin kunnan yhteystietojen löytämiseen. Opinnäytetyö. Kemi-Tornion ammattikorkeakoulu. Tietojenkäsittelyn ala. Tornio. 29 sivua. 22 liitettä.

Opinnäytetyömme tavoitteena on suunnitella ja toteuttaa Web-puhelinluettelo toimeksiantajallemme, Kolarin kunnalle. Toimeksiantajalla ilmeni tarve kyseisen sovelluksen kehittämiseen, koska yhteystietojen etsiminen kunnan Internet-sivustolta oli melko hankalaa sekä hidasta. Sovelluksen tarkoitus on siis sekä helpottaa, että nopeuttaa yhteystietojen etsimistä Kolarin kunnan Internet-sivustolta. Sovellusta suunniteltaessa pyrimme ensisijaisesti helppokäyttöisyyteen ja yksinkertaiseen sovellukseen.

Tutkimusmetodina käytimme konstruktiivista tutkimusmenetelmää. Opinnäytetyön aineistona toimi sekä ohjelmointiin liittyvä kirjallisuus, että Internet-lähteet. Työ aloitettiin sopimalla toimeksiantajan kanssa sovellukseen vaadittavista ominaisuuksista. Ulkoasun sekä käyttöliittymän suunnittelussa saimme käyttää omaa luovuutta. Sovelluksen tietokanta toteutettiin MySQL-Tietokannan hallintajärjestelmällä, minkä jälkeen itse sovellus PHP-kielellä.

Valmis sovellus toteuttiin asetettujen tavotteiden mukaisesti ja se tyydytti toimeksinantajamme tarpeita. Toiminnoiltaan sovellus on tarpeeksi yksinkertainen, jotta sitä pystyy käyttämään myös kokemattomampikin käyttäjä. Yhteistietojen etsiminen on nopeaa ja vaivatonta, mikä olikin yksi tärkeimmistä tavotteista sovellusta suunniteltaessa.

Kehittämisehdotuksemme sovelluksen parantamiseksi liittyy mahdollisiin lisätoimintoihin. Sovellus olisi entistä monipuolisempi, jos siihen liitettäisiin Flash-pohjainen karttapalvelu. Karttapalvelun avulla käyttäjät voisivat löytää esimerkiksi etsimänsä työntekijöiden työhuoneet. Tämä ratkaisu helpottaisi työntekijöiden paikantamista ja tekisi hakupalvelusta monipuolisemman.

Asiasanat: Web-sovellus, MySQL, PHP, Internet

Abstract

Joki, Tapani & Ruokamo, Jussi & Suunta, Jari. 2009. Web Application For Finding Contact Information Of Kolari. Bachelor's Thesis. Kemi-Tornio University of Applied Sciences. Business and Culture. ICT Study Program. Tornio. 29 pages. 22 appendices.

The objective of this thesis was to design and implement a Web phone book for our commissioner, the municipality of Kolari. The need to develop this application arose because the search for contact information from the municipality's web page was difficult and slow. Therefore the purpose of this application was to facilitate and quicken the search for contact information from the web page of Kolari municipality. During the designing of the application we pursued primarily for a user-friendly and simple application.

As a research method we used a constructive research method. The material for this thesis was programming-related literature and Internet-resources. We started the work by agreeing upon the specifications of the application with the commissioner. In designing of exterior and user interface we could use our own creativity. The database of the application was made by MySQL database management system. The application itself was made by PHP programming language.

The completed application was compatible with objectives which we set and our commissioner was satisfied with it. The application is simple enough, that even inexperienced users can use it. The search for contact information is quick and simple, which was also one of the most important objectives during the designing.

Our developing proposal for improving the application associates with eventual add-on functions. The application would be more versatile if a Flash-based map service is attached to it. With the map service users could find for example the office of the employee they are looking for. This solution would make it easier to find employees and also make application more versatile.

Keywords: Web application, MySQL, PHP, Internet

SISÄLTÖ

TIIVISTELMÄ
ABSTRACT

1 JOHDANTO.....5

1.1 Työn lähtökohta, tavoite ja rajaus.....5

1.2 Tutkimusongelma.....6

1.3 Keskeiset termit.....7

1.4 Projektiorganisaatio.....8

1.5 Tutkimusmetodi.....9

2 SUUNNITTELU JA TOTEUTUS.....11

2.1 Tietokannan suunnittelu.....11

2.2 Käyttöliittymän suunnittelu.....18

3 TOTEUTUS.....19

3.1 Sovellus.....19

3.2 Toiminnot.....20

3.3 Haku –ja ylläpitosovelluksen testaus.....21

4 KÄYTTÖÖNOTTO JA YLLÄPITO.....23

4.1Sovelluksen käyttöönotto.....23

4.2Sovelluksen ylläpito.....24

5 TULOKSET JA JOHTOPÄÄTÖKSET.....25

5.1 Tulos ja sen arviointi.....25

5.2 Jatkotutkimussuositukset.....25

5.3 Pohdinta.....26

LÄHTEET

LIITTEET

1. JOHDANTO

Opinnäytetyömme tavoitteena on toteuttaa Internet-selaimen kautta toimiva yhteystietojen hakusovellus toimeksiantajallemme, Kolarin kunnalle. Kolarin kunta on muun muassa siellä sijaitsevan Ylläksen laskettelukeskuksen myötä suosittu matkailukohde, jonka vuoksi myös kunnan Internet-sivuston käyttö on vilkasta.

Toimeksiantajalle ilmeni tarve kyseisen sovelluksen kehittämiseen, koska yhteistietojen etsiminen kunnan sivustolta on melko hankalaa ja hidasta. Sovelluksen tarkoitus on siis sekä helpottaa, että nopeuttaa yhteystietojen etsimistä kunnan Internet-sivustolta. Lisäksi sovelluksen tulisi olla mahdollisimman helppokäyttöinen. Koska kyseessä on hakusovellus, sen tärkein ominaisuus on hakutoiminto. Sovelluksessa on viisi vaihtoehtoista hakukenttää: etunimi, sukunimi, työnimike, toimipaikka sekä yleiset numerot. Näiden hakukenttien avulla käyttäjä voi siis etsiä haluamansa yhteistiedon. Opinnäytetyömme toteutettiin projektityönä ja ohjaavana opettajana toimi Yrjö Koskenniemi.

1.1 Työn lähtökohta tavoite ja rajaus

Opinnäytetyömme sai alkunsa, siitä että ollessani (Tapani Joki) työharjoittelussa Kolarin kunnan Atk-tuessa, kyselin Atk-vastaava Esa Nordbergilta mahdollista opinnäytetyön aihetta. Asiaa hetken mietittyään Esa Nordberg ehdotti aiheeksi Kolarin kunnan Internet sivuilla toimivan Web-puhelinluettelon tekoa, josta kunnanhallinnon ja terveyskeskuksen työntekijöiden yhteystiedot olisivat helposti löydettävissä. Kysyin luokkatovereiltani Jari Suunnalta ja Jussi Ruokamolta, olisivatko he kiinnostuneita lähtemään mukaan kyseiseen projektiin ja kiinnostusta oli, joten päätimme ottaa Web-puhelinluettelon suunnittelun ja toteutuksen opinnäytetyömme aiheeksi.

Tutustuessamme Kolarin kunnan Internetsivuihin huomasimme, että kunnan hallinto-osaston ja sivitysosaston yhteystiedot löytyivät melko helposti, mutta osastojen, kuten sosiaali-ja terveysosaston, terveyskeskuksen ja muiden kuntalaista kiinnostavien

yhteystietojen löytäminen sivustolta oli melko hankalaa. Sovimme toimeksiantajan kanssa, että tekisimme tietokantaan perustuvan sovelluksen, josta yhteystiedot löytyisivät nimeä, työtehtävää ja sijaintia hakukriteerinä käyttäen. Sivujuonena haulle suunnittelimme myös vektori-grafiikkaa käyttävän kartan lisäämistä osaksi yhteystietoja. Flash-animoitu kartta voisi näyttää, missä kunkin henkilön toimipiste sijaitsee.

Opinnäytetyöhömme kuuluvat Web-puhelinluettelo-sovelluksen suunnittelu ja toteutus sekä tietokantaratkaisun suunnittelu ja toteutus. Flash-animoinnin käyttäminen on myös opinnäytetyön osa. Työ sovittiin toteutettavaksi HTML ja PHP-ohjelmakoodia ja MySQL-tietokantaratkaisua hyväksikäyttäen. Myös Adobe Flash-kehitysympäristöä käytetään hyväksi työn teossa.

1.2 Tutkimusongelma

Tutkimusongelmaa voidaan pitää tutkimuksen lähtökohtana. Sen lisäksi, että me voimme perustella, että jostain aiheesta kannattaa hankkia tietoa, meidän on kyettävä perustelemaan aiheen tutkiminen tieteellistä lähestymistapaa ja tieteellisiä menetelmiä hyväksi käyttämällä. Tutkimusongelma sisältää siten paitsi aiheen tutkimisen mielekkyyden perustelun, myös täsmällisemmällä tasolla tutkimuskysymyksen tai kysymyksiä, joihin tutkimuksessa haetaan vastauksia. Tällä tavoin tutkimusongelman muotoilemisella ja sen vaatimuksella pyritään myös välttämään informaatioarvoltaan mitättömän nollatutkimuksen tuottaminen.

Tutkimusongelmassa olisi hyvin tärkeää selittää mahdollisimman selkeästi ja tarkkarajaisesti, mitä tutkimuksessa on tutkittu: millaisiin ongelmiin tai kysymyksiin on pyritty löytämään vastauksia.

Tutkimusongelmassa olemme pyrkineet saamaan vastaukset seuraaviin seikkoihin:

- Miten saamme suunniteltua ja toteutettua tilaajan laatukriteerit huomioon ottaen hyvät, toimivat ja selkeät PHP-sivut, sekä toimivan tietokannan?
- Miten saisimme tehtyä sovelluksestamme mahdollisimman helposti ylläpidettävän?

- Minkälainen tietokantaratkaisu olisi riittävän haastava toteuttaa, ja siitä olisi mahdollisimman paljon hyötyä asiakkaalle?
- Miten opastaa työn tilaajaa ylläpidon suhteen?

1.3 Keskeiset termit

MySQL on suosittu ja tehokas SQL-tietokannan hallintajärjestelmä, joka on hyvin laajalti levinnyt ihmisten käyttöön ympäri maailmaa. MySQL poikkeaa monista kaupallisista tietokantajärjestelmistä siten,että hallinnointi tapahtuu komentoriviltä tai tekstipohjaisella asiakasohjelmalla. MySQL on myöskin ilmainen tietokanta, joka tarkoittaa tässä tapauksessa GPL- lisenssiä, jonka mukaan sitä saa käyttää vapaasti, mutta sitä ei saa levittää kaupallisesti. (MySQL 2008)

PHP (lyhenne sanoista *PHP: Hypertext Preprocessor*) on Perlin kaltainen ohjelmointikieli, jota käytetään erityisesti Web-palvelinympäristöissä dynaamisten web-sivujen luonnissa. PHP on kehittäjiensä mukaan yleiskäyttöinen skriptikieli, joka soveltuu erityisesti web-sovellusten kehitykseen. Kieli on siis suunnattu dynaamisten web-sivustojen toteuttamiseen eli se sopii hyvin sivustojen toiminnallisuuden rakentamiseen. (PHP 2008)

Internet on eri puolille maailmaa ulottuva digitaalinen tiedonsiirtoverkko, joka koostuu tuhansista pienemmistä kaupallisista, akateemisista tai valtiollisista verkoista, jotka on teknisesti liitetty toisiinsa Internet-protokollaperheen avulla. Internetiä kutsutaankin usein verkkojen verkoksi. Käyttäjilleen Internet näkyy eri sovellutusten kautta, joista tutuimpia ovat WWW, sähköposti ja chat. (Internet 2008)

HTML lyhenne tulee sanoista *Hypertext Markup Language* on avoimesti standardoitu kuvauskieli, jolla voidaan kuvata hyperlinkkejä sisältävää tekstiä eli hypertekstiä. HTML tunnetaan erityisesti sellaisena kielenä, josta webbisivut rakentuvat. (HTML 2008)

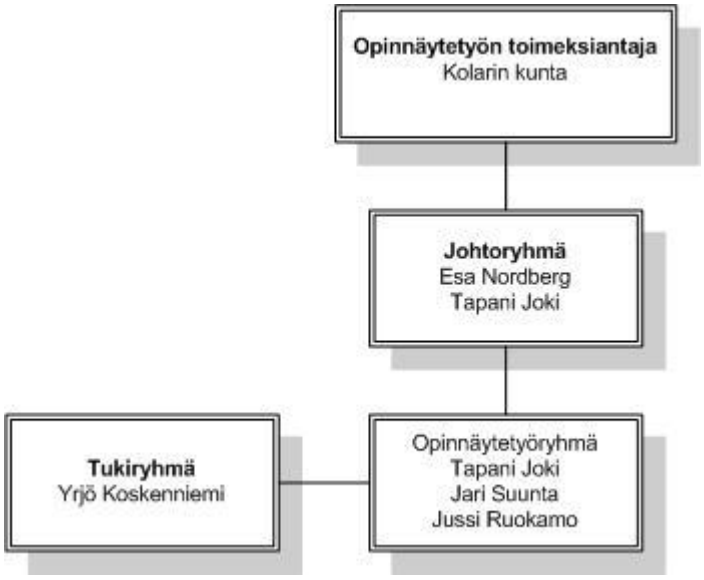
Käytettävyydellä tarkoitetaan sitä, että millainen sovellus on asiakkaalle käyttää ja miten luontevasti hän pystyy sovelluksen kanssa toimimaan. Käytettävyydeltään hyvän

sovelluksen käyttöliittymä on helppokäyttöinen, tehokas ja miellyttävä. (Käytettävyys 2008)

UML-mallinnus (*Unified Modeling Language*) on Object Management Groupin (OMG) vuonna 1997 standardoima graafinen mallinnuskieli, joka sisältää 13 erilaista kaaviota. Kaavioista kuudella kuvataan rakennetta, kolmella käyttäytymistä ja neljällä vuorovaikutusta. (UML 2008)

1.4 Projektiorganisaatio

Opinnäytetyön toimeksiantajana toimii Kolarin kunnan Atk-vastaava Esa Nordberg. Opinnäytetyön johtoryhmään kuuluu Atk-vastaava Esa nordberg ja yhteyshenkilönä Tapani Joki. Opinnäytetyöryhmään kuuluvat Tapani Joki, Jari Suunta ja Jussi Ruokamo. Opinnäytetyön ohjaavana opettajana toimii Yrjö Koskenniemi.



Kuva 1. Projektiorganisaatio

1.5 Tutkimusmetodi

Tutkimusmetodina opinnäytetyössämme olemme käyttäneet konstruktiivista tutkimustapaa. Tämä sen vuoksi, että koimme konstruktiivisen tutkimuksen olevan lähimpänä tapaa, jolla pyrimme löytämään ratkaisut ongelmiimme, ja pyrimme myöskin saamaan aikaan tutkimuksellamme konkreettisen ja toimivan sovelluksen.

Konstruktiivinen tutkimus voidaan nähdä soveltavan tutkimuksen muotona, jolle on ominaista sellaisen uuden tiedon tuottaminen, joka tähtää johonkin sovellutukseen tai tavoitteeseen.
Konstruktiivinen tutkimus tuottaa konstruktioita, jotka antavat ratkaisun joihinkin eksplisiittisiin ongelmiin. Rajanvetoa tutkimusote on tehnyt analyyttiseen mallinrakennukseen, tieteelliseen ongelmanratkaisutoimintaan ja myös konsultoinnin suuntaan. Olennaisena osana konstruktiiviseen tutkimukseen kuuluu ongelman sitominen aiempaan tietämykseen sekä ratkaisun uutuuden ja toimivuuden osoittaminen työssämme. ”Konstruktiivisen tutkimuksen keskeiset osat ovat ongelman käytännöllinen relevanssi ja kytkentä teoriaan, konstruktio ongelman ratkaisuna sekä ratkaisun käytännön toimivuus ja ratkaisun teoreettinen uutuusarvo”. Tutkimuksessa lähtökohdat ovat käytännössä ongelmalliseksi koetussa tilanteessa ja lopputulosta tutkimuksesta tulee voida käyttää ongelmien ratkaisemisessa. Tutkimustyö jaetaan vaiheisiin, jotka ovat seuraavanlaiset:

1. Relevantin ja tutkimuksellisesti mielenkiintoisen ongelman etsiminen.
2. Esiymmärryksen hankinta tutkimuskohteesta.
3. Innovaatiovaihe, ratkaisumallin konstruoiminen.
4. Ratkaisun toimivuuden testaus eli konstruktion oikeellisuuden osoittaminen.
5. Ratkaisussa käytettyjen teoriakytkentöjen näyttäminen ja ratkaisun tieteellisen uutuusarvon osoittaminen.
6. Ratkaisun soveltamisalueen laajuuden tarkastelu.

Konstruktiivinen tutkimus on luonnehdittavissa normatiiviseksi case-tutkimukseksi. Sille läheisiä tutkimusotteita ovat toisaalta teoreettisluonteisen päättelyn suhteen päätöksentekometodologinen ja muutoksen aikaansaamisen pyrkimyksen suhteen toimintatutkimus. Konstruktiivinen tutkimusote korostaa kuitenkin nimenomaan

konstruktioiden luomista ja niiden toimivuuden todentamista käytännössä. Konstruktioiden toimivuus on tutkimusotteen puitteissa ideoiden totuudellisuuden tunnusmerkki ja ratkaisujen yleistettävyyden tarkastelu on erityistapauksen yleisten piirteiden pohdintaa. Nimenomaan tässä tarkastelun kiinnittyminen teoreettiseen taustaan on tarpeen.

Olemme hyödyntäneet opinnäytetyössämme konstruktiivista tutkimusotetta siten, että olemme pyrkineet suunnittelemaan kaiken tekemämme mahdollisimman hyvin kuvin ja taulukoin, jonka avulla olemme pyrkineet saamaan aikaan mahdollisimman hyvin toimivan sovelluksen ja pyrkineet pääsemään tavoitteeseemme.

(Konstruktiivinen tutkimus 2008)

2. Suunnittelu ja toteutus

2.1Tietokannan suunnittelu

Web-puhelinluettelo-sovelluksen perustan muodostaa tietokanta. Tietokannan on oltava hyvin suunniteltu, vaikka käyttäjä ei sovelluksesta näekään muuta kuin käyttöliittymän ja hakujen tulokset. Sovellus voidaan mieltää vaikka tukevaksi tiilitaloksi. Jos tiilitalon perustana on vain hiekkaa, luhistuu talokin sen päällä. Jos taas perusta on suunniteltu ja toteutettu asianmukaisesti, seisoo tiilitalo jykevänä vielä vuosien päästäkin. Toimivan sovelluksen lähtökohtana on siis tässä tapauksessa looginen ja hyvin toteutettu tietokanta.

Tietokannan suunnittelu on kuitenkin monimutkaisempi asia kuin pelkkä tietokannan mallinnus. Mallinuksessa tietokannan rakennetta kuvataan jollakin kuvaustekniikalla, kuten esimerkiksi ER-Mallinnus eli käsitemallinnus ja UML:n mukainen luokkalaavio. Tietokannan suunnitteluun kuuluu useita työvaiheita vaatimusmäärittelyn teosta tietokannan mallinnuksen kautta fyysiseen suunnitteluun. Jotta tietokannan suunnittelu tuottaisi toivotun tuloksen, on järkevää opetella tulkitsemaan erilaisia kuvaustapoja ja menetelmiä.

> Ennen tietokannan suunnitteluun ryhtymistä, on hyvä pohtia, mikä on suunniteltavan tietokannan rakenne, eli mihin pyritään. Hyvän tietokannan keskeisiä ominaisuuksia ovat:
> - Kattavuus: Sisältää kaikki järjestelmissä ja kyselyissä tarvittavat tiedot ja yhteydet.
> - Selkeys ja ymmärrettävyys: Yksinkertainen rakenne, ilmaisuvoima; helppo tehdä kyselyjä.
> - Yleiskäyttöisyys: Soveltuvuus erilaisiin ympäristöihin ja asiakkaille tarvitsematta muuttaa tietokannan rakennetta.
> - Eheys: toisteisuuden välttäminen; oikeellisuus; sisäinen ristiriidattomuus.
> - Ohjelmointimukavuus; selkeät tietorakenteet, sarakkeilla kiinteä merkitys
> - Suorituskyky eli tehokkuus: Riittävä vastausaika tapahtumille ja riittävän tehokkaat eräajot
> (Hovi & Huotari & Lahdenmäki 2005, 21).

Ennen tietokannan varsinaiseen suunniteluun ryhtymistä, kartoitimme hiukan tarvitsemamme tietokannan rakennetta karkeasti yllä esiintyvän teorian pohjalta. Keskustelimme kunkin kohdan tarpeellisuudesta tarvitsemamme tietokannan näkökulmasta ja tulimme siihen tulokseen, että tietokannan yleiskäyttöisyys eli sen soveltuvuus muihin kuin käyttämäämme Web-ympäristöön, ei ole välttämätöntä. Tietokannan turvallisuus on myös yksi tavoitteista, eli tietokannassa olevia tietoja voi muokata vain ylläpitäjän myöntämien käyttöoikeuksien haltija. Koska tietokanta sisältää yhteystietoja, on tietokannan päivittämisenkin oltava mutkatonta. Siksi rakenteeseen ja tietokannan taulujen ja sarakkeiden nimet pyritään pitämään mahdollisimman tuttuina ja helppotajuisina. Ei ole järkeä tehdä tietokannasta esimerkiksi englanninkielistä, koska sen ylläpitäjä on suomalainen.

> CASE-työkalujen (Computer Aided Software Engineering) keskeinen tehtävä on auttaa tietojärjestelmien suunnitelmissa ja toteutuksessa samalla tavalla kuin CAD (Computer Aided Design) auttavat esim. arkkitehteja rakennusten suunnittelussa. CASE-välineillä voidaan piirtää, kuvata ja dokumentoida käsitemalleja. Käsitemallista voidaan näiden tuotteiden avulla muodostaa relaatiotietokannan taulujen perustamiskäskyt, mikä nopeuttaa tietokannan perustamistyötä
> (Hovi ym. 2005, 26).

Tietokannan suunnittelun apuna käytämme sitä varten suunniteltuja työkaluja. kuten Microsoft Visiota, joka käyttää vektori-grafiikkaa UML-taulukoiden muodostamiseen. Visiolla piirsimme tietokannasta malleja, joista tietokantaa ryhdytään muodostamaan. Tietokannan sunnittelussa ja toteutuksessa käytettiin myös Server2Go-ohjelmistokokonaisuudessa mukana olevaa PHPMyAdmin-työkalua. Suunnittelussa käytettiin myös Microsoftin Acces-tietokantatyökalua.

Puhelinluettelosovelluksen käyttämää MySQL-tietokantaa ryhdytään suunnittelemaan vaiheittain. Käyttämäämme eri vaiheiden muodostamaa kokonaisuutta kutsutaan suunnitteluputkeksi.
(Hovi ym. 2005, 26).

Totesimme suunniteluputken paremmin soveltuvaksi omaan työhömme, sillä suunnitteluputkessa on helpompi palata taaksepäin kuin toisessa,

vesiputous-suunnittelumallissa, jossa suunnittelu kuvataan vain eteenpäin menevänä prosessina jossa askel taaksepäin on hankalaa. Vähäisen tietokantasuunnittelu-kokemuksemme takia halusimme suunnitella tietokantaa mallin mukaan, joka on joustavampi virheiden korjaamisen kannalta.

Suunnittelu aloitettiinn tekemällä vaatimusmäärittely toimeksiantajan kanssa. Vaatimusmäärittelyssä selvittettiin mitä toimeksiantaja tietokannasta ja sovelluksesta haluaa.

> Seuraava askel putkimallissa on käsiteanalyysin teko. Käsiteanalyysin tavoitteena on määrittää ja kuvata havainnollisella kaaviolla tietokantaan talletettavia tietoja, jotta lopulta voidaan perustaa tarpeita hyvin palveleva tietokanta
> (Hovi, ym. 2005, 32).

> Käsiteanalyysiin kuuluu myös tallennettavien tietojen välisten riippuvuuksien ja ominaisuuksien analysointi. Kuitenkaan käsiteanalyysissä ei mennä vielä varsinaisen teknisen toteutuksen tarkempaan analyysiin.
> (Käsiteanalyysi 2008).

Tarkka käsiteanalyysi helpottaa varsinaisen tietokannan toteutusta valtavasti. Alunperin yksinkertaiselta tuntunut tietokanta alkoi tuntua varsin monimutkaiselta sitä analyyttisesti tutkaillessamme. Kuitenkin analyysin tuloksena oli käsitemalli, josta tietokannan todellinen rakenne on nähtävissä.. Käsiteanalyysi teki tietokannan suunnittelusta kuitenkin varsin stressaavaa, sillä jokaisen edistysaskeleen perästä jouduimme uudelleen analysoimaan tehdyistä muutoksista koituneet seuraukset. Käsitemallia siis päivitettiin aina tilanteen mukaan. Analyysin teossa työkaluina käytimme Microsoft Visio-ohjelmistokokonaisuuden Database-kategorian database model diagram-pohjaa, kuten myös Microsoft excel-taulukkojakin.

Käsiteanalyysissä pyritään kuvaamaan käsitteitä, tietoja eli attribuutteja ja näiden yhteyksiä. Käsite (entity) kuvaa asiaa, esinettä tai henkilöä, josta halutaan säilyttää tietoa tietokannassa myöhempää käyttöä varten
(Hovi, Ym. 2005, 35).

Käsitteitä käsitemallissamme ovat esimerkiksi Työntekijät, työtehtävät ja osasto. Käsitteille määritellään yksilöivä tieto eli perusavain (primary key), kuten henkilön henkilötunnus tai tilin tilinumero.
(Hovi,Ym 2005, 35).

Perusavain vastaa relaatiotietokannan taulun perusavainta, eli arvoa joka ei saa puuttua. Esimerkiksi Työntekijät-käsitteeseen perusavaimeksi määritetään tieto eli attribuutti TyöntekijäID. Attribuutit kuvaavat käsitettä. Käsitteeseen kuuluu joukko tietoja jotka ovat käsitteen ominaisuuksia. Muita tietoja Työntekijät-käsitteessä TyöntekijäID:n lisäksi ovat esimerkiksi, etunimi, sukunimi, puhelinnumero ja sähköposti-osoite.

Perusavaimiksi valittiin kaikkiin käsitteisiin itse keksitty avain, joka ei perustu mihinkään tiettyyn tietoon tai ominaisuuteen. Tälläisten Surrogaatti-, eli keinoavainten käyttö helpottaa tietokannan ylläpitoa, koska se ei sisällä muuttuvaa tietoa, kuten esimerkiksi sosiaaliturvatunnus, jonka käyttö perusavaimena saattaisi vaatia muokkausta. Esimerkiksi Työntekijät-käsitteenTyöntekijäID-avain on nelilukuinen numerosarja joka alkaa sarjasta 0001 ja päättyy sarjaan 9999. Jokaista työntekijää vastaa siis tälläinen numerosarja, joka ei juonnu mistään. Osasto-käsitteessä OsastoID-attribuutti on vastaavasti kaksilukuinen numerosarja, joka alkaa sarjasta 01 ja päättyy sarjaan 99. Avain-attribuutti on käsitteissä ja tietokannassa pakollinen, eli se yksilöi aina tietyn käsitteen. Tietokantaan tietoja tallennettaessa, eli esimerkiksi uuden työntekijän tietoja lisättäessä, TyöntekijäID ei saa jäädä tyhjäksi.
Tietokantaa suunniteltaessa on huomioon otettava myös käsitteiden väliset yhteydet eli suhteet. Yhteyksiä ovat yksi-yhteen, yksi-moneen ja moni-moneen suhteet. Yksi-moneen-yhteys on tietokantasuunnittelussa yleisin yhteys ja sitä kutsutaankin helppotajuisemmin isä-lapsi-yhteydeksi, jossa isällä on monta lasta, mutta lapsella vain yksi isä. Käsitemallissamme on ylivoimaisesti eniten moni-moneen yhteyksiä eri käsitteiden välillä, kuten esimerkiksi siten, että osastolla on useampi työntekijä ja myös sama työntekijä voi työskennellä useammalla osastolla. Yhteyttä voidaan myös mallin selkeyttämiseksi kuvata verbillä, kuten että työntekijä kuuluu osastoon tai osasto jakaantuu toimipaikkoihin.

Käsiteanalyysin teko helpottaa tietokannan rakenteen hahmottamista huomattavasti. Analyysin ensi vaiheissa käytössämme ei ollut vielä

varsinaisia, toimeksiantajan toimittamia tietoja, mutta vaatimusmäärittelyn pohjalta rakentamamme analyysi on varsin yksinkertainen, sillä siinä käsitteitä on vain kolme: Osasto, työntekijä ja aliosasto. Lopullisessa, tarkennetussa käsitemallissa käsitteitä ovat Osasto, työntekijä, työtehtävät, toimipaikat ja yleiset numerot. käsitteet liittyvät toisiinsa vielä moni-moneen yhteyksillä, joka vaikeuttaa mallin muuttamista suoraan tietokannaksi. Seuraava vaihe suunnitteluputkimallissa on tarveanalyysi, jossa käsitemalli täydennetään tarkalle tasolle.

Tarveanalyysi tehdään koska käsitemalli on vielä varsin suurpiirteinen kuva tietokannan rakenteesta. Käsitemallissa tehtyihin käsitteisiin lisätään tietoja sen mukaan, mikä on käsitteen tietotarve. Toisin sanoen tarveanalyysin teko merkitsee sitä, että mietimme tekemämme käsitemallin pohjalta, millaisia tietoja tarvitsimme. Esimerkiksi miettiessämme hakuja, joita hakukoneella pitäisi pystyä tekemään. huomasimme hankaluuden numeroissa, jotka eivät olleet puhelinumeroita ihmisille, vaan esimerkiksi terveyskeskuksen päivystykseen tai yleinen numero kirjastonhoitajille. Käsitemallissamme ei ollut käsitettä, johon tälläiset yleisnumerot voitaisiin liittää, joten teimme niille kokonaan uuden käsitteen nimeltä Yleisetnumerot, johon tiedoiksi tuli toimipaikkaID avaimeksi ja muiksi tiedoiksi nimike, puhelinnumero ja lisätieto-kentät. Seuraava askel Suunnitteluputkessa on Normalisointi.

> Normalisointi on menetelmä, jonka avulla tietorakenteita voidaan jalostaa ”parempaan” tallennusmuotoon. Parempi tarkoittaa tässä yhteydessä rakennetta, jossa Tietojen toistaminen on minimoitu, on tehokas päivitysten kannalta, on helpompi pitää yhdenmukaisena, sillä tiedot tarvitsee päivittää vain yhteen paikkaan.”
> (Hovi, Ym., 2005, 86).

> Normalisoinnissa käytettyjä sääntöjä kutsutaan normaalimuodoiksi. Jos tietokannan suunnittelu noudattaa ensimmäistä sääntöjoukkoa, kyseessä on ensimmäinen normaalimuoto. Jos seurataan kolmea ensimmäistä normalisoinnin sääntöjoukkoa, tietokannan sanotaan olevan kolmatta normaalimuotoa
> (Meloni, 2003, 32).

Normalisointisääntöjä on olemassa useampia, mutta käytännössä kolme ensimmäistä sääntöä ovat tärkeimpiä tietokannan rakenteen kannalta.
syötämme tiedot tietokantaan PHPMyAdmin-tietokannan hallinta ohjelmalla tekemiemme mallien pohjalta. Normalisointi vaiheet tehdään yhtäaikaa tietokannan teon kanssa.
Jotta tietokanta saadaan ensimmäiseen normaalimuotoon, tietokannasta poistetaan toistuvat ryhmät ja moniarvoiset sarakkeet. Toisessa normaalimuodossa kyse on jo yksittäisistä tiedoista, eli tietokannan tauluissa esiintyvistä attribuuteista ja niiden välisistä suhteista. Toisen normaalimuodon säännön mukaan kaikkien taulun sarakkeiden on oltava riippuvaisia taulun koko perusavaimesta, eikä vain osa-avaimesta. Tämä on oltava näin siinä tapauksessa, jos taulun avain on moni osainen. Tietokantamme taulut täyttävät toisen normaalimuodon suoraan, koska taulujen perusavaimet koostuvat vain yhdestä attribuutista, numerosarjasta joka ei ole yhteydessä mihinkään muuhun attribuuttiin.
Tietokanta on kolmannessa normaalimuodossa silloin, kun taulussa olevat kentät ovat riippuvaisia vain avainkentästä, eikä mistään muusta attribuutista. Normalisointi helpottaa tietojen päivittämistä tietokantaan, koska samat tiedot eivät toistu moneen kertaan.

Valmiissa tietokannassa on yhteensä seitsemän taulua, joissa yhteensä 541 riviä tietoa. Tietokannan koko on yhteensä 124 kilotavua.
Tietokanta toteutettiin MySQL-tietokantaohjelmistolla ensinnäkin sen matalien kustannuksien takia ja toiseksi helpon siirrettävyyden takia. Varsinainen rakennustyö tehtiin selaimen kautta käytettävällä PHPMyAdmin-tietokannan hallintatyökalulla, joka on mukana Server2Go-ohjelmistopaketissa.

Vaikka tietokantapalvelimen tarjoaja tarjoaakin tietokannan maksullista varmuuskopiointia, ei varmuuskopiointia kannata jättää pelkästään sen varaan. Puhelinluettelon tietokannasta voidaan tehdä PHPMyAdmin-hallintaohjelmalla varmuuskopio SQL-tiedostomuotoon erittäin kätevästi, josta se voidaan tarpeen tullen myös palauttaa yhtä helposti. Tietokannan voi varmuuskopioida myös kopioimalla tietokannan hakemisto ja sen alla olevat

.FRM, .MYD ja .MYI –tiedostot esimerkiksi USB-muistitikulle. Tietokannan varmuuskopiointi olisi järkevää tehdä 3-4 kuukauden välein, riippuen tietysti lisätyn ja päivitetyn tiedon määrästä. Automatisoidulle varmuuskopioinnille emme näe syytä, sillä päivityksiä tietokantaan tehdään melko harvoin ja silloin kanta voidaan varmuuskopioida käsin.

Koska tietokanta sisältää kunnan hallinnon yhteystietoja, on tietokannan tietoturva otettava tosissaan. Olisi valtava vahinko, jos joku pääsisi sotkemaan yhteystietoja vain siksi, koska tietoturva on jäänyt retuperälle. Koska MySQL-tietokanta on asennettu muualle kuin Kolarin kunnan palvelimelle, suurin huolenaihe on yhteyden turvallisuus. Internetin yli liikkuva tieto voidaan kaapata ja olisi erittäin ikävää jos tiedot kuten MySQL-tietokannan hallintaan liittyvät tiedot joutuisivat vääriin käsiin. Varsinkin Tietokannan käyttöönotossa, Nebula Oy:stä suositeltiin yhteydeksi salattua SSH-Yhteyttä tietokannan käyttöön (Secure Shell), mutta totesimme kuitenkin UNIX-taitojemme olevan riittämättömät yhteyden käyttöön. Toinen ja valitsemamme turvallinen tapa siirtää tietokanta Nebulan palvelimelle on käyttää selainpohjaista phpMyAdminia, joka käyttää HTTP-yhteyttä. Samaa yhteyttä käyttää myös sovellusta varten tehty päivitysohjelma.

Vakava uhka tietokannan turvallisuudelle on myös se, että tietokannan päivittäjä jättää koneensa valvomatta silloin kun on kirjautuneena päivityssovellukseen. Tälläiseen fyysiseen tietoturvaan voidaan kyllä vaikuttaa helposti, mutta unohduksiakin tapahtuu. Paras tapa välttää tälläisiä vahinkoja on päivittää tietokantaa silloin kun kukaan ei katsele päivittäjän olkapään yli hänen tekemisiään. Viisainta olisi tehdä tarvittavat päivitykset mahdollisimman nopeasti ja huolellisesti ja kirjautua ulos päivityssovelluksesta. Päivityssovelluksen salasana olisi myös järkevää vaihtaa säännöllisin väliajoin, tai viimeistään silloin kun epäilee salasanan selvinneen jollekin muulle.

Oikoteitä tietoturvaan ei ole, sellaisiin asioihin kuten salasanojen kirjoittaminen muistilapuille ja niiden liimaaminen tietokoneen näyttöön, ei voi vaikuttaa muut kuin käyttäjä itse. Salasanan merkitys katoaa, jos se on kaikkien nähtävissä.

2.2 Käyttöliittymän suunnittelu

Web-sovelluksen käyttöliittymä rakentuu minimissään sekä HTML-elementeistä, erityisesti lomakkeet, että Web-selaimen ominaisuuksista. Lisäksi voidaan käyttää erilaisia elävöittämistekniikoita kuten CSS, JavaScript ja Java-appletit. DHTML eli Dynamic HTML on lähinnä yleisnimitys tekniikoille, jotka koostuvat W3C:n DOM-määrittelystä Document Object Model, jossa HTML-dokumentti kuvataan puumaisena oliorakenteena.Tällä hetkelllä se tarkoittaa käytännössä HTML:n, CSS:n ja JavaScriptin integroimista yhden käsitteen alle. On erittäin tärkeää ymmärtää, että vaikka asiakastekniikoita käyttämällä voidaan saavuttaa tiettyä dynaamisuutta Web-sovellukseen, dynaamisuus rajoittuu pääsääntöisesti käyttöliittymän hallintaan. (Rantala vuosi 2002 9).

Web ohjelmointiympäristönä saattaa hämmentää aluksi kokenuttakin ohjelmoijaa. Hallittavaa on paljon Web-selaimessa muodostettavasta käyttöliittymästä palvelimella suoritetattavaan sovelluslogiikkaan. Vaikeinta on kuitenkin omaksua käyttöliittymän ja sovelluslogiikan yhdistävän HTTP-protokollan (HyperText Transfer Protocol) toimintamekanismeja. HTTP on se yhteyskäytäntö, jolla Web-sovelluksen käyttöliittymä ja palvelimella sijaitseva sovelluslogiikka viestivät keskenään. Etenkin HTTP-protokollan tilattomuus pakottaa tutustumaan HTTP-viestien rakenteeseen otsakkeineen, joita hyödyntällä voidaan käyttää mm. evästeitä, autentikointimenetelmiä ja istunnon hallintaa. Lisäksi viestien välityksessä käytetty URL:n koodaustapa vaati perehtymistä. (Rantala 2002, 11).

3. TOTEUTUS

3.1 Sovellus

Suunnittelun tavoitteena meillä on saada aikaan niin toimiva suunnitelma ja kokonaisuus, että joutuisimme normalisoimaan sovellusta mahdollisimman vähän. Tämä tarkoittaa sitä, että tarkoituksena on luoda kuvin ja kaavioin niin selkeä pohja työllemme kuin mahdollista. Pyrimme siis piirtämään ennen tulevaa ohjelmointivaihetta kaikki mahdolliset kaaviot, mitkä näemme tarpeellisiksi ja tulemme kysymään ohjaavilta opettajiltamme neuvoa, että mitä kannattaa piirtää ja hyväksytämme ne heillä vielä ennen kuin sovellusta aletaan tehdä.

Työkaluina PHP-sovelluksen suunnittelussa käytämme Microsoft Office Visioa ja Prosaa. Microsoft Office Visiolla teemme vuokaavion, jolla kuvataan sovelluksen toiminta ja pystymme käyttämään sitä sitten hyödyksi, kun tulee aika ohjelmoida varsinainen sovellus. Prosalla piirrämme UML luokkakaavion, jonka tarkoituksena on myöskin kuvata sitä, että miten sovellus tulee toimimaan, mutta luokkakaavio tulee kuvaamaan sovelluksen toimintoja enemmänkin käyttäjän kannalta kuin yleisen toimivuuden kannalta.

Php-sovelluksen teon aloitimme tekemällä hyvän suunnitelman siitä, että mitä tulemme tekemään ja millaisia ominaisuuksia ohjelma tulee pitämään sisällään. Tässäkin tapauksessa ajattelimme, että hyvin suunniteltu on melkein kuin puoliksi tehty. Suunnitelmaa tehdessämme otimme aluksi selvää toimeksiantajaltamme, että millaisia ominaisuuksia ohjelmassa tulisi olla. Kun meillä oli selvillä, millainen sovelluksesta tulee tehdä niin aloimme piirtää erilaisia kaavioita, joiden avulla sovellus olisi helppo toteuttaa ja tietäisimme koko ajan tavoitteemme sovelluksen teon etenemisestä. Kuvat ja kaaviot tulevat myöskin jatkossa helpottamaan sovelluksen toiminnan selittämistä, kun täytyy esimerkiksi selittää sovelluksen toimintaa ja ominaisuuksia ihmisille, jotka eivät ennestään tiedä kyseisetä asiasta tai sovelluksesta mitään.

Tehtyämme kattavan ja riittävän hyvän suunnitelman sovelluksesta aloimme kirjoittaa koodia, jolla saisimme aikaan toimivan ja helppokäyttöisen sovelluksen Kolarin kunnan käyttöön.
Aloitimme sovelluksen teon tekemällä sivulle tietysti aluksi pohjan, johon lisäisimme sen jälkeen toiminnot joita toimeksiantajamme halusi ohjelmalta.
Sovelluksen etusivun valmistuttua loimme sovellukseen tietokantayhteyden, jonka avulla puhelinluettelosta löytää kaikki tietokannan tiedot helposti ja vaivattomasti.
Saatuamme kaikki hakuehdot ja ominaisuudet toimimaan sovelluksesta meidän tarvitsi enää muokata sovellus Kolarin kunnan nettisivuille sopivan näköiseksi.

Ongelmakohtia sovellusta tehdessämme ilmeni useita, mutta varsinkin MySQL- kyselyt osoittautuivat vaikeaksi haasteeksi meille. Siinä vaikeaa oli se, että pienikin virhe taulujen yhteyksien kanssa tai vaikkapa vain yksi pieni näppäily virhe, niin ohjelma ei tulostanutkaan enää haluamiamme tietoja. Suurimpiin ongelmakohtiin sovelluksen teossa saimme apua onneksi ohjaavilta opettajiltamme, jotka olivat käytettävissämme lähes aina kun apua tarvittiin ja he olivat meille suunnattoman suuri apu. Itse sovellukseen meillä oli tarkoitus aluksi animoida myöskin karttapalvelu Flashilla, mutta kun tutustuimme hiukan kyseiseen aiheeseen kurssilla niin huomasimme, että haaste olisi ollut meille liian suuri ajan ja osaamisen kannalta. Tämän vuoksi karttapalvelua ei ainakaan vielä tässä vaiheessa Kolarin kunnan verkkosivuilla tulla näkemään, mutta sitä voisi harkita jatkossa toteutettavaksi haasteeksi.

3.2 Toiminnot

Sovellus, jonka olemme luoneet, on hakuohjelma, jolla voidaan etsiä Kolarin kunnan työntekijöiden, virastojen ja koulujen yhteystietoja. Halusimme sovellukseen monta hakuvaihtoehtoa, jotta yhteystietojen etsiminen olisi käyttäjälle mahdollisimman vaivatonta. Hakuohjelmassa on siis useita eri hakuvaihtoehtoja, joista osa on hakukenttiä ja osa alasvetovalikkoja. Hakukenttiin käyttäjä voi kirjoittaa itse haluamansa hakuehdon, kun taas alasvetovalikoissa käyttäjä valitsee haluamansa hakuvaihtoehdon. Valitun haun tulokset ilmestyvät itse sovelluksen alapuolella olevaan tilaan. Ensimmäisenä hakukenttänä on etu- ja sukunimiin perustuva haku, jossa sovellus etsii työntekijöitä

nimen perusteella. Toisena hakukenttänä on taas työtehtävään perustuva haku . Edellä mainittussa hakutoiminnossa käyttäjä täyttää itse haluamansa hakuehdon. Kolmas hakukenttä on osastoihin perustuva haku, jossa käyttäjä valitsee alasvetovalikosta haluamansa osaston. Hakutulokseksi käyttäjä saa valitsemansa osaston kaikki työntekijät. Neljäs hakukenttä on toimipaikkoihin perustuva haku ja se toimii samalla periaatteella kuin osastohaku. Viides hakuvaihtoehto pitää sisällään yleiset numerot, jossa käyttäjä voi etsiä esimerkiksi virastojen ja koulujen puhelinnumeroita. Kuten osasto- ja toimipaikkahaku myös yleisien numeroiden haku on toiminnoltaan alasvetovalikko. Sovelluksesta haluttiin helppokäyttöinen ja käyttäjäystävällinen. Hakutoiminnot ovat yksinkertaisia ja halutun yhteystiedon etsiminen on helppoa.

3.3 Haku- ja ylläpitosovelluksen testaus

Aloitimme haku- ja ylläpitosovellusten testauksen heti ohjelmointivaiheen alettua. Testauksen perustana käytimme niin sanottua V-mallia, joka alkaa itse koodauksesta ja päättyy koko järjestelmän testaukseen. Testauksen tavoitteita olivat sovelluksen toimivuuden osoittaminen, laadun varmistaminen, virheiden löytäminen ja ylläpidon helpottaminen. Tarkoituksemme oli siis testata sovelluksia, koko ohjelmointiprosessin ajan. Meillä oli ohjelmoinnin alettua tarkoin tiedossa mitä asiakas sovelluksilta vaatii, jonka perusteella loimme myös itse ohjelmat ja suoritimme testauksen. Käytimme sekä itse hakusovelluksessa että ylläpitosovelluksessa samanlaista kaavaa testauksen suhteen. Ohjelmointivaiheen alussa testauksen kohteita olivat pääasiassa sovelluksen yksittäiset osat ja niiden toimivuus sekä tietokanta. Tähän kuului muun muassa koodikielen oikeellisuuden huolellinen tarkistaminen ja tietokannassa esiintyvien tietojen tarkistaminen. Halusimme näin ollen varmistaa, että sovellus on vakaalla pohjalla ohjelmoinnin suhteen. Seuraava vaihe oli testata sovellusten eri osien toimivuutta keskenään. Tässä vaiheessa esiintyi ehkä eniten ongelmia ja jouduimme muuttamaan sovellusta aika ajoin radikaalistikin. Tämä johtui yksinkertaisesti siitä, että kellään meistä ei ollut kovinkaan paljon kokemusta sovellusten vaatimista ohjelmointikielistä. Sen vuoksi yksittäisten osien koodikielissä esiintyi monesti puutteita, joita jouduimme korjaamaan ja täydentämään jälkeenpäin. Mielestämme ohjelmoinnin vaativin vaihe olikin saada ohjelmointiosat toimimaan moitteetta keskenään. Kun sovellus oli tältä osin

testattu ja valmis, aloitimme koko järjestelmän testauksen. Ohjelmoinnin aikana tekemämme testaukset auttoivat tässä vaiheessa merkittävästi. Tämän vuoksi emme joutuneet lopputestauksen aikana tekemään kovinkaan suuria muutoksia sovellukseen. Koska sekä haku- että ylläpitosovellus olivat toiminnoiltaan melko yksinkertaisia, ei niiden testaus ollut kovinkaan vaativaa. Lopputestauksessa keskityimme pääasiallisesti siihen, että sovellukset toimivat asiakkaan antamien vaatimusten mukaisesti. Jouduimme tekemään muutamia pieniä muutoksia järjestelmään, mutta pääasiassa toimivuuden suhteen ei ollut lopulta ongelmia. Tämä johtui pitkälti siitä, että olimme korjanneet isoimmat ongelmat jo testauksen toisessa vaiheessa. Lopputestaukseen kuului myös ohjelmakoodin tarkistaminen, jossa pyrimme etsimään mahdollisia virheitä, jotka voivat heikentää sovelluksen toimivuutta. Kun testaus oli mielestämme suoritettu, toimitimme valmiit sovellukset asiakkaan hyväksyttävästi. Asiakas testasi sovellukset ja ilmoitti haluamistaan muutoksista. Teimme asiakkaan toiveesta muutaman muutoksen sovelluksiin ja pienimuotoisen uudelleen testauksen jälkeen ohjelmat olivat valmiita käyttöön.

4. KÄYTTÖÖNOTTO JA YLLÄPITO

4.1 Sovelluksen käyttöönotto

Web-Puhelinluettelon käyttöönotto aloitettiin tietokannan siirrolla. Kolarin kunnalla on käytössään Nebula Oy:n Web-hotellipalvelu, jossa kunta pitää sivujaan. Hotellipalveluun on mahdollista liittää MySQL-tietokantoja ja se olikin välttämätöntä, sovelluksemme kannalta. Uusi tietokanta avattiin ja sen hinnaksi tuli 9 eur./kk, eli yhteensä 108 euroa vuodessa 175 euron hintaisen Webhotellipalvelun lisäksi. Hämmästelimme hintaa,mutta kunnan atk-vastaava vakuutti, että kunnan talous ei kaadu tämän takia. Kun uusi tietokanta oli Nebula Oy:n toimesta perustettu, kyse oli vain tekemämme tietokannan kopion siirtämisestä Nebula Oy:n tietokantapalvelimelle. Tietokannan siirtoon suositeltiin tietokannanhallintaohjelmaa aiemmin käyttämäämme phpMyAdminia, jonka Tuo/Vie-toiminolla siirron tulisi olla lastenleikkiä. Tietokannan siirron kanssa vietettiin kuitenkin monta epätoivoista tuntia, koska uuden tietokantapalvelimen lisääminen phpMyAdminin konfigurointitiedostoon ei ollut tuttua ja Nebula Oy:stä ei ohjeita tietokannan siirtoon juuri herunut. Viimein apu kuitenkin löytyi kirjoja ja Internetiä selaamalla ja yrityksen ja erehdyksen kautta. Tekemämme tietokannan tauluista tehtiin Vie-toiminolla SQL-tiedosto, joka kopiotiin Tuo-toiminolla Nebula Oy:n tietokantapalvelimelle. Käyttöönoton ensimmäinen vaihe oli siis valmis.

Seuraava vaihe käyttöönotossa oli haku-ja ylläpitosovellusten siirtäminen Nebula Oy:n palvelimelle, jossa Kolarin kunnan verkkosivut sijaitsevat. Siirto suoritettiin FTP-ohjelmalla(File Transfer Protocol), jolla kunnan sivuja päivitetään muutenkin. Hakusovellus toimi moitteettomasti, kun vaadittavat muutokset tietokantayhteyden muodostamiseen oli tehty, eli mysql_connect()-funktion arvot oli muutettu Nebula Oy:n toimittamien tietokannan käyttäjänimien ja salasanojen mukaisiksi. Ylläpitosovelluksen käyttöönotto oli hiukan mutkikkaampaa, sillä Nebula Oy:n Web-hotellipalvelun PHP-tuki on versiota PHP5, jolloin konfiguraatiodirektiivi REGISTER_GLOBALS on oletusarvoisesti poissa päältä turvallisuussyistä. Vaikka käyttämässämme Server2Go-työkalussa on myös PHP versiota PHP5, on siinä em. direktiivi oletusarvoisesti päällä. Käytännössä REGISTER_GLOBALS-direktiivi vaikuttaa siten, että ylläpitosovelluksen muuttujia ei ole alustettu automaattisesti, joten ne täytyy alustaa erikseen jotta sovellus

toimisi odotetulla tavalla. Muuttujat alustettiin POST-metodilla ja ylläpitosovellus toimi kuten pitikin.

Enää jäljellä oli hakusovelluksen linkitys Kolarin kunnan verkkosivuston etusivulle ja ylläpitäjien opastus. Kaikeen kaikkiaan käyttöönotto onnistui yllättävän kivuttomasti, laajalti ohjaavan opettajan neuvojen ansiosta.

4.2 Sovelluksen ylläpito

Olemme luoneet hakuohjelmalle myös ylläpitosovelluksen, jolla voidaan päivittää tarvittaessa tietokantaa. Sovelluksen toimintoja suunniteltaessa käytimme samoja periaatteita, kuin itse hakuohjelman kanssa. Mahdollisimman helppokäyttöinen sovellus takaa tässäkin tapauksessa parhaan lopputuloksen. Ylläpitosovellus on toiminnoiltaan seuraavanlainen. Käyttäjä avaa sovelluksen, jonka ensimmäinen vaihe on sisäänkirjautuminen. Tässä vaiheessa täytetään käyttäjätunnus- ja salasanakenttä, jonka jälkeen päästään ylläpitosovelluksen etusivulle. Etusivulla on mahdollisuus kirjautua ulos ylläpitosovelluksesta tai valita tietokannan taulu, jota halutaan päivittää. Näitä tauluja ovat työntekijät-, työtehtävät, osastot-, toimipaikat-, yleiset numerot-, pikanumerot ja users-taulu. Ylläpitäjä pääsee taulujen päivitystilaan valitsemalla etusivulla olevista linkeistä haluamansa. Kun ylläpitäjä on valinnut haluamansa päivityskohteen, sovelluksessa avautuu itse päivitystila. Päivitystilassa näkyy kaikki taulun tiedot. Jos kyseessä on esimerkiksi työntekijät-taulu, ylläpitäjä voi muuttaa tietokannassa jo valmiiksi esiintyvien työntekijöiden tietoja tai lisätä uuden työntekijän. Lisäksi ylläpitäjällä on mahdollisuus poistaa työntekijöitä. Sovellus ilmoittaa mahdollisista virheistä päivitettäessä tietoja. Samalla periaatteella toimivat myös muihin tauluihin perustuvat päivitystilat. Olemme lisänneet joidenkin taulujen päivitystiloihin niin sanottuja passiivisia kenttiä, joita ei voi päivittää. Esimerkiksi työntekijät-taulussa näkyy työtehtävä-taulusta työntekijän työtehtävänimi. Tämä helpottaa joissain määrin päivitystä, koska tiedot ovat sidoksissa eri taulujen kesken toisiinsa. Ylläpitosovellukseen on lisätty joka sivulle paluu-mahdollisuus, jotta siirtyminen eri sivujen välillä olisi ylläpitäjälle mahdollisimman helppoa.

5. TULOKSET JA JOHTOPÄÄTÖKSET

5.1Tulos ja sen arviointi

Opinnäytetyönä teimme siis Kolarin kunnalle tietokannan kunnan työntekijöistä ja heidän tiedoistaan ja sijainnistaan kunnassa. Teimme lisäksi web-sovelluksen, joka toimii niin sanottuna nettipuhelinluettelona. Näiden avulla ihmisillä on helppo löytää kunkin kunnan työntekijän yhteystiedot vaivatta Internetistä, eikä heidän tarvitse soitella pitkin kuntaa saadakseen kiinni jotain tiettyä henkilöä.
Onnistuimme mielestämme saamaan aikaan toimivan kokonaisuuden, josta tulee varmasti olemaan jatkossa paljon hyötyä ihmisille, jotka Kolarin kunnan nettisivuilla käyvät. Olemme siis tyytyväisiä työmme tulokseen sen kannalta, että miten se toimii ja on lisäksi helppo ja selkeä kokonaisuus käyttäjälle.
Tuloksen kannalta huonoa on se, että emme ehtineet tehdä työtä sillä aikataululla jossa oli tarkoitus. Tämä johtuu siitä, että meillä tuli vastaan paljon ongelmia joiden kanssa aikaa kului huomattavasti odotettua pidempään.

5.2 Jatkotutkimussuositukset

Jatkotutkimukseksi opinnäytetyöllemme mietimme karttasovellusta, jolla Kolarin kunnan verkkosivuilta voisi löytää kaikkien työntekijöiden sijainnin kartalta.
Sovelluksen avulla käyttäjä löytäisi siis kaikkien työntekijöiden työhuoneet tai työtilat, mistä tavoittaa etsimänsä henkilö muullakin tavalla kuin puhelimitse tai sähköpostilla.
Parhaina vaihtoehtoina sovelluksen toteuttamiseen pidämme Flashia ja MapLibiä.
MapLib Integrator-sovelluksella voidaan tehdä kartta miltei mistä kohteesta tahansa ja siihen voidaan lisätä toiminnallisuutta PHP- ja Javascript-tiedostoilla, kuten esimerkiksi aktiivinen merkki joka vilkkuu kartalla jos työntekijä on paikalla toimistossaan.
MapLib-sovellus on käyttöjärjestelmäriippumaton, mutta se vaatii toimiakseen PHP4-ympäristön ja MySQL-tietokannan.
Adobe Flash-kehitysympäristöllä olisi myös mahdollista tehdä vastaavanlainen karttasovellus, mutta se vaatisi erittäin paljon Actioncript-ohjelmointikielen osaamista, jota meiltä ei löytynyt. Adobe Flash-kehitysympäristö on myös maksullinen, mutta monipuolisempi kuin MapLib.

5.3 Pohdinta

Tehdessämme opinnäytetyötä koimme työn vaiheet haastavina ja mielenkiintoisina. Opimme työn edetessä paljon uusia asioita tietokantojen suunnittelusta, teosta ja lisäksi tutustuimme meille uuteen tietokannantekoympäristöön nimeltään Server2Go. Opimme myöskin paljon uusia asioita PHP- ohjelmoinnista ja ennen kaikkea lisäsimme valmiuksiamme tehdä jatkossa töitä tiiminä, koska opinnäytetyön ajan teimme tiiviisti töitä ryhmänä aina, kun oli mahdollista.

Aloitimme opinnäytetyön tekemisen Syyskuussa.2008. Työssämme pyrimme aluksi tekemään mahdollisimman hyvät suunnitelmat, joiden pohjalta tehdä tietokanta ja PHP-sovellus. Kun piirsimme suunnitelmia kävimme näyttämässä niitä aina ohjaaville opettajillemme ennen kuin aloimme tekemään varsinaista työtä. Hyvien suunnittelujen tekeminen osoittautuikin yllättävän työlääksi ja aikaa vieväksi työksi. Tämän vuoksi aikataulun kanssa olikin ongelmia, koska suunnitteluun meni niin paljon aikaa, ja toimivan tietokannan tekeminen vei ryhmältämme enemmän aikaa, kun olimme odottaneet, koska tietoja tietokannassa on paljon ja ennen PHP-sovelluksen tekoa meillä piti olla moitteetta toimiva tietokanta valmiina.

Kolmen hengen ryhmänä opinnäytetyötä oli mielestämme hyvä tehdä, koska kohdatessamme ongelmia niitä oli ratkaisemassa kolme henkilöä, eikä kenenkään tarvinnut ratkaista ongelmia yksinään.

Opinnäytetyötä tehdessämme meillä oli hyötyä aiemmin opitusta muun muassa projektityönteosta, tietokannoista ja PHP:stä. Eniten aikaa ja tutkintaa vaati kuitenkin suunnittelu, vaikka osa malleista on käytä läpi koulutuksemme aikana, niin niitä ei joko muistanut, tai asiat tulivat kokonaan uutena asiana ryhmällemme.

Opinnäytetyön tekeminen antaa meille mielestämme valmiudet siirtyä työelämän haasteisiin valmistumisemme jälkeen. Tämä sen vuoksi, että tämä oli ensimmäinen näin laaja ja haastava projekti opiskelumme aikana ja olimme lisäksi projektin aikana tiimi

ryhmänä, jonka lisäksi todella paljon apua antoivat ohjaavat opettajamme Yrjö Koskenniemi ja Markku Henriksson.

LÄHTEET

Painetut

Gilmore, Jason 2005. PHP & MySQL – Tehokas hallinta. Gummerus kirjapaino Oy, Jyväskylä.

Heinisuo, Rami & Rauta, Ilkka 2007. PHP ja MySQL - Tietokantapohjaiset verkkopalvelut. Gummerus Kirjapaino Oy, Helsinki

Hovi, Ari & Huotari, Jouni & Lahdenmäki, Tapio 2005. Tietokantojen suunnittelu ja indeksointi. Docendo Finland Oy, Jyväskylä

Lahtonen, Tommi 2002. SQL. Docendo Finland Oy, Jyväskylä.

Meloni, Julie 2003. MySQL. Edita Prima Oy, Helsinki.

Rantala, Ari 2002. PHP – Web-ohjelmoinnin peruskirja. Docendo Finland Oy, Jyväskylä.

Painamattomat

HTML 2008. Luettu 20.10.2008
<http://fi.wikipedia.org/wiki/HTML>

Internet 2008. Luettu 20.10.2008
<http://fi.wikipedia.org/wiki/Internet>

Kollanius, Sami 2008. Johdanto. Luettu 28.10.2008
<http://users.jyu.fi/~kolli/ITK215_05/php/>

Kollanius, Sami 2008. PHP ja tietokanta. Luettu 28.10.2008
<http://users.jyu.fi/~kolli/ITK215_05/php/?sivu=tietokanta>

Laaksonen, Antti 2003. Käytännön PHP-opas. Luettu 28.10.2008
<http://www.ohjelmointiputka.net/opas.php?tunnus=phpj>

MySQL 2008. Luettu 15.9.2008
<http://fi.wikipedia.org/wiki/MySQL>

Orasaari, Marko 2009. Malli käyttöohjeen sisällöstä. Luettu 7.1.2009
<http://www.cs.helsinki.fi/group/alabra/orasaari/kayttoohje.html>

PHP 2008. Luettu 28.10.2008
<http://fi.wikipedia.org/wiki/PHP>

UML-mallinnus 2008. Luettu 15.9.2008
<http://fi.wikipedia.org/wiki/UML-mallinnus>

Vanhala-Nurmi, Vuokko 2008. Mikä Internet on. Luettu 24.11.2008
<http://myy.helia.fi/~vanvu/tietoliikenne/internetkehys.html>

Vaatimusmäärittely **Liite 1**
1(1)

1. Johdanto

Ohjelmisto tehdään yhteystietojen etsimiseen verkossa. Tietokanta tulee olemaan kaikkien käytettävissä, sitä käytetään Kolarin kunnan Internet-sivujen kautta. Tilaajana tietokannalle on Kolarin kunta.

Järjestelmän tavoitteena on helpottaa yhteystietojen löytämistä kunnan Internet-sivuilta.

2. Toiminnot

Järjestelmältä vaaditaan sitä, että tietokanta on nopea ja yksinkertainen käyttää, ja tietokannasta tulee löytyä kaikki tilaajan vaatimat tiedot. Toimintoja ovat: Tietokannan tietojen mutkaton ylläpidettävyys

3. Tiedot

Järjestelmällä hallittavat tiedot ovat Kolarin kunnan työtekijöiden yhteystietoja, mm. puhelinnumero, sähköpostiosoite ja ammattinimike.

4. Liittymät

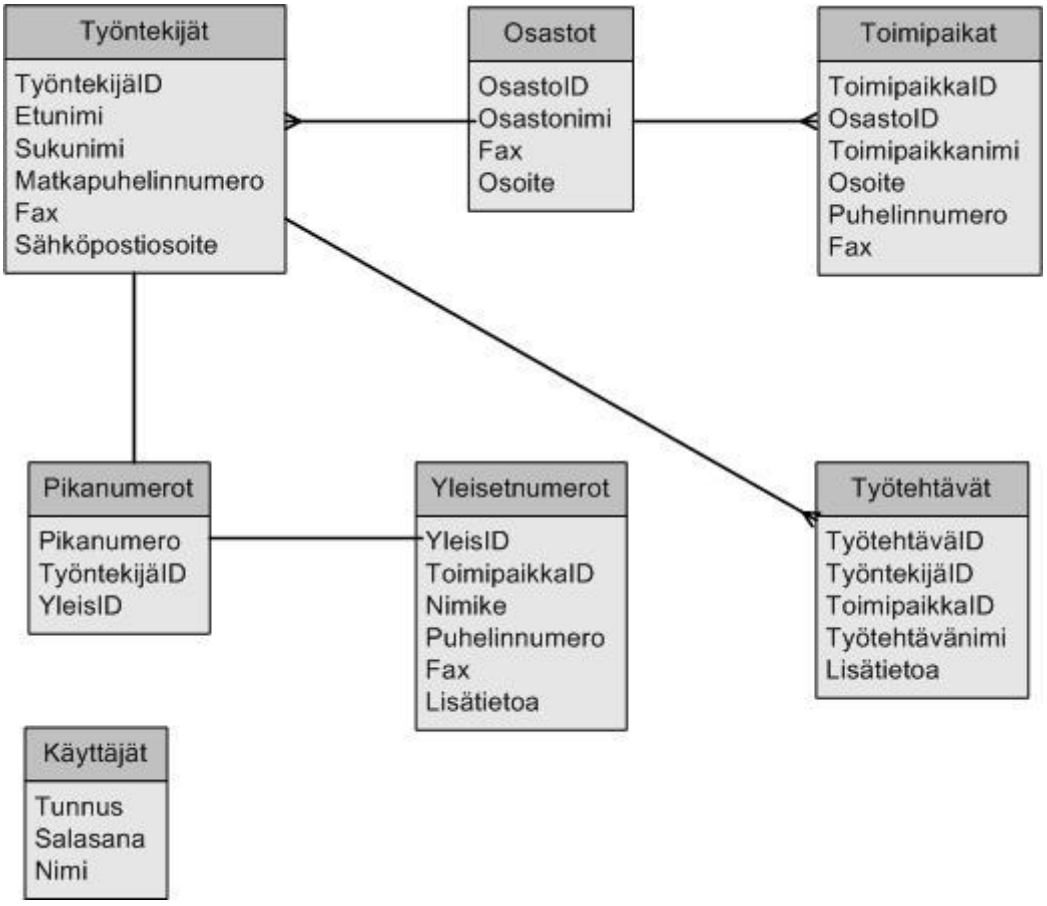
Tietokantaa käytetään Web-sovelluksella, joka on ohjelmoitu PHP:llä. Käyttäjinä toimivat kaikki Kolarin kunnan työtekijöiden yhteystiedoista kiinnostuneet. Käyttöympäristönä sovellukselle on Kolarin Web-sivut.

5. Muut ominaisuudet

Suorituskyky pyritään saamaan mahdollisimman hyväksi ja käytettävyys helpoksi.

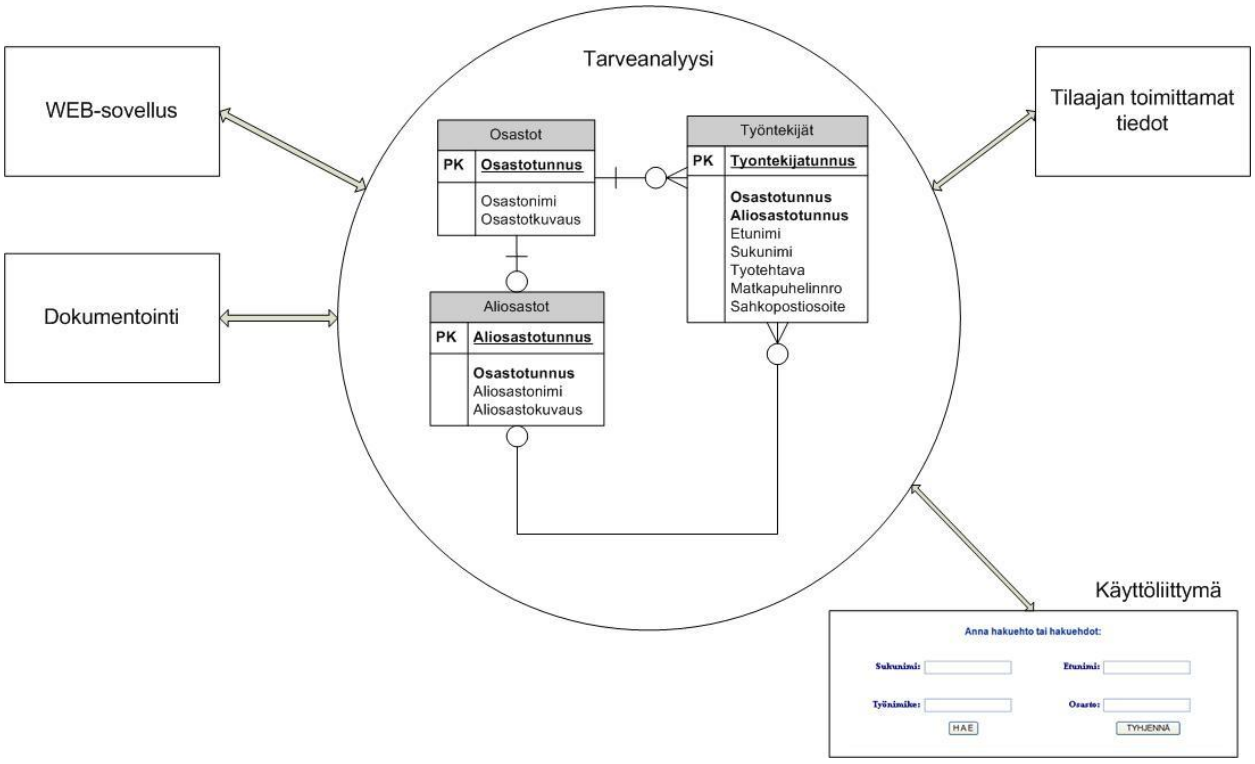
Käsitemalli **Liite 2**

1(1)



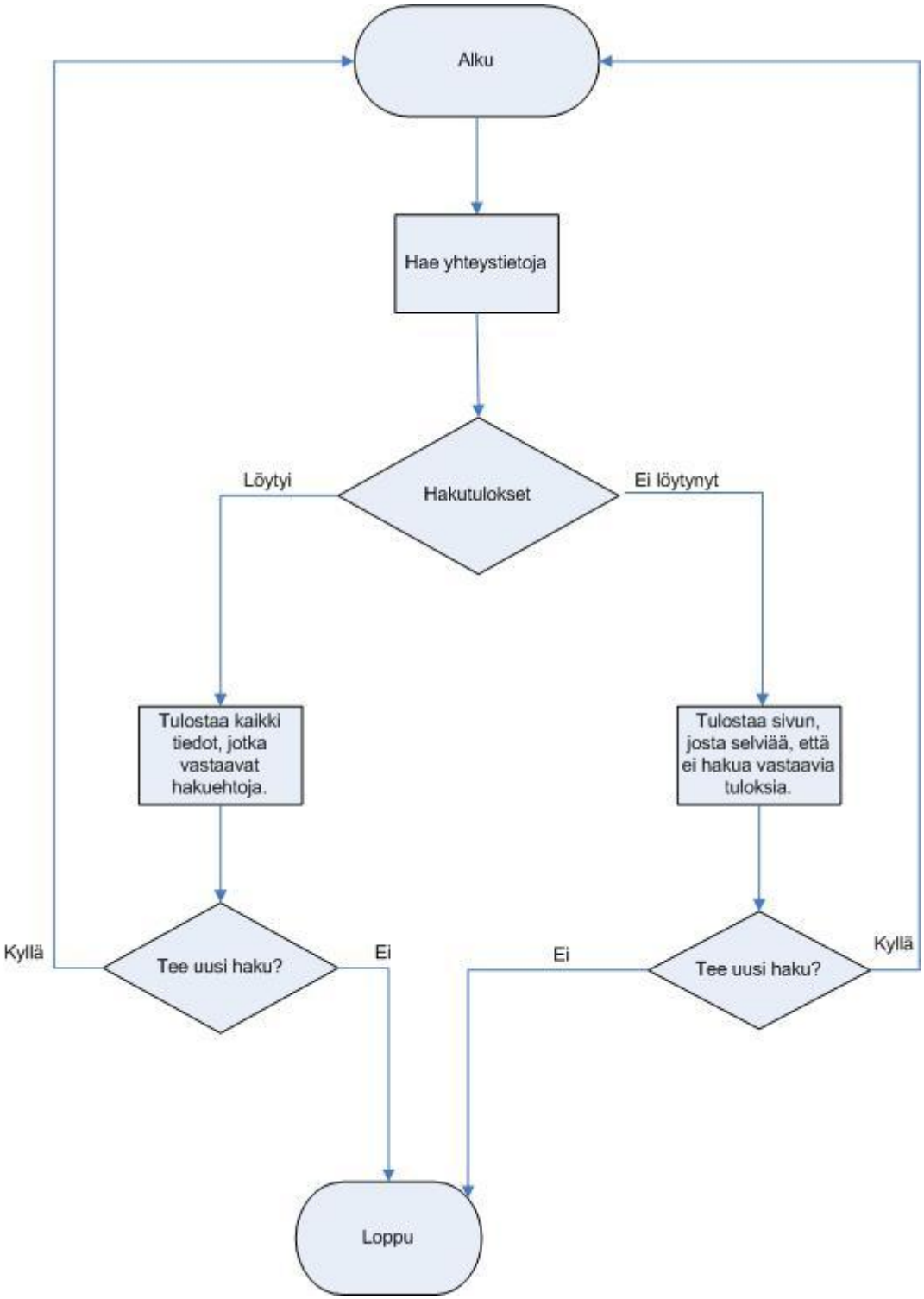
Tarveanalyysi **Liite 3**

1(1)



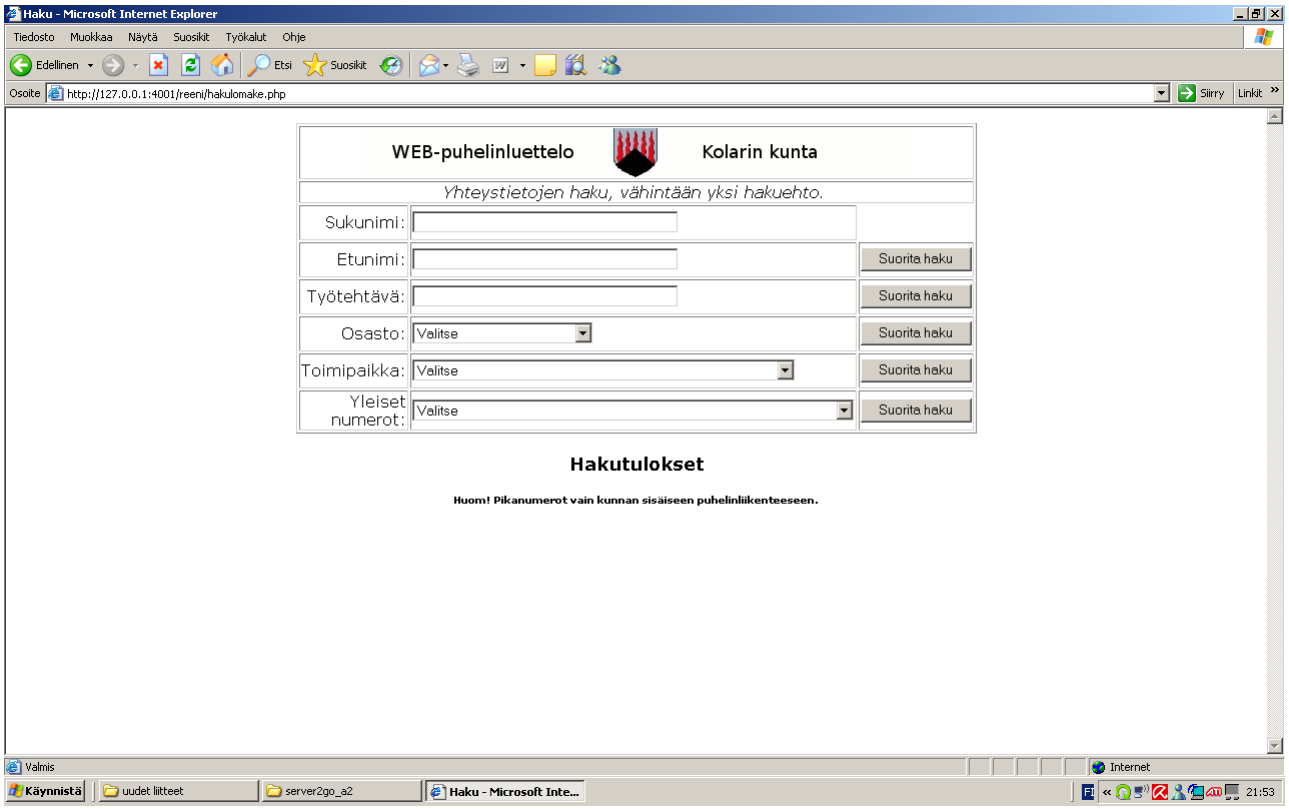
Vuokaavio sovelluksesta **Liite 4**

1(1)



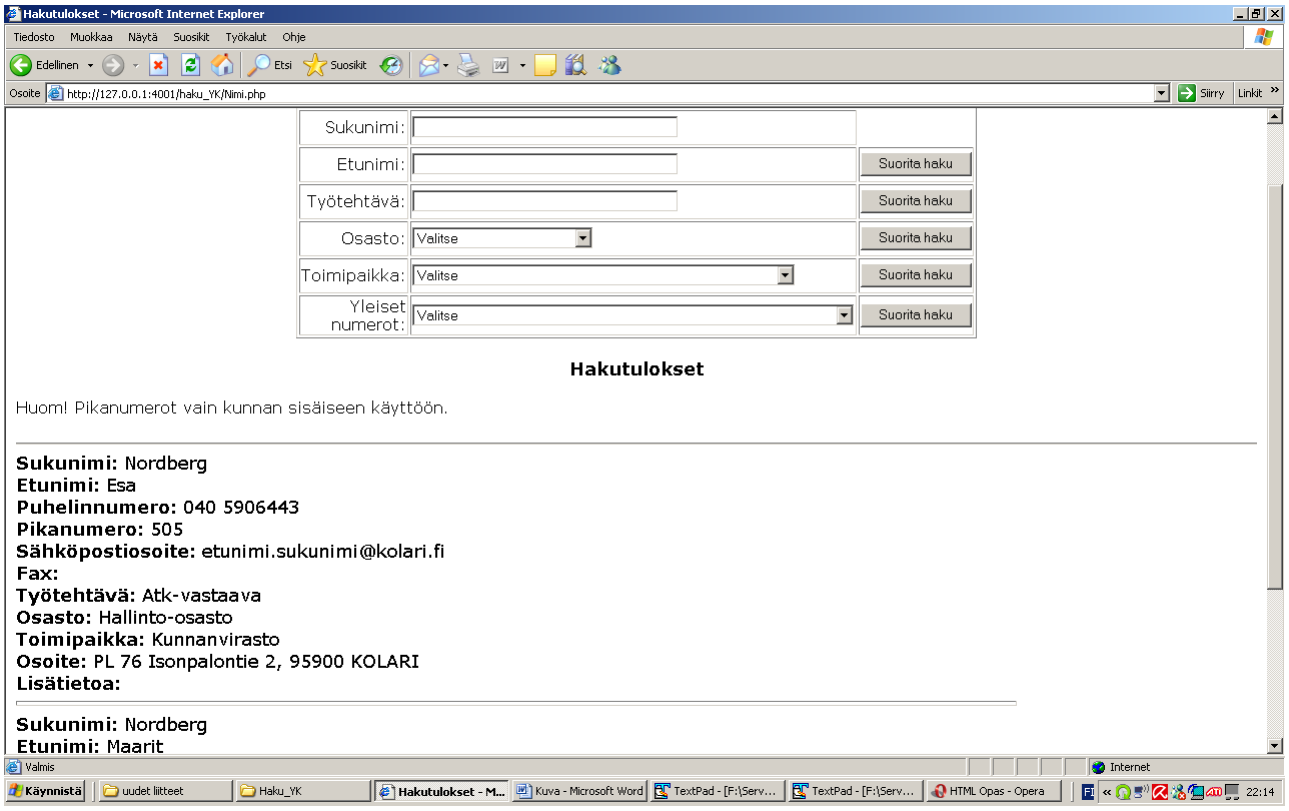
Hakusovellus selainikkunassa **Liite 5**

1(1)



Hakutulokset selainikkunassa **Liite 6**

1(1)



Hakulomake Koodiesimerkki **Liite 7**

1(4)

```
<html><head><title>Haku</title>
<body bgcolor ="#FFFFFF">
<font face = "Verdana, Helv, sans serif" size="4">
<meta http-equiv="Content-Type" content="text/html; charset=ISO8859-1" />
</head>
<body>

<?php
mysql_connect ("localhost", "root")  or die (mysql_error());
mysql_select_db ("puhelinluettelo_1");

$sql = mysql_query("SELECT toimipaikat.ToimipaikkaID, toimipaikat.Toimipaikkanimi,
osastot.Osastonimi FROM `toimipaikat` INNER JOIN  `osastot` ON
toimipaikat.OsastoID=osastot.OsastoID order by Toimipaikkanimi ");
$sql1 = mysql_query("select * from osastot order by Osastonimi");
$sql2 = mysql_query("SELECT yleisetnumerot.YleisID, yleisetnumerot.Nimike,
toimipaikat.Toimipaikkanimi FROM `yleisetnumerot` INNER JOIN  `toimipaikat` ON
yleisetnumerot.ToimipaikkaID=toimipaikat.ToimipaikkaID order by Toimipaikkanimi ");

?>
<tr>
<p>


<table width="550" border="1" align="center">

<form action="Nimi.php" method="post">
<td  colspan="5" align="center"><img src="Vaakunap.jpg"></td>
</tr>
<tr>
<td height="20"colspan="5"  align="center">Yhteystietojen haku</td>
</tr>
 <tr>
 <td  height="35"  align="right">Sukunimi:</td>
 <td> <input type="text" name="snimi" size="40" tabindex="1">
 </tr>
 <tr>
<td height="35" align="right">Etunimi:</td>
<td height="35"><input type="text" name="enimi" size="40" tabindex="1">
                                        <td align="right"><input type="submit" name="Hae"
value="Suorita haku"/></td>
                                </tr>

</form>

<form action="Ttehtava.php" method="post">
```

Hakulomake koodiesimerkki **Liite 7**
2(4)

```
                <tr>
                                        <td height="35" align="right" >Työtehtävä:</td>
                                        <td><input type="text" name="ttehtava" size="40"
tabindex="1">
                                        <td align="right"><input type="submit" name="Hae"
value="Suorita haku"/></td>
                                </tr>

                </form>

</p>

<p>

<tr>
<td height="35" align="right">
<form action="osasto.php" method="get">
                        Osasto:</td>
                        <td><select name="osasto" onchange='this.form.submit()'>
                        <option>Valitse</option>
                        <?

                        while ($row = mysql_fetch_array($sql1))

                         {
                                $ID=$row["OsastoID"];
                                //settype($ID,"integer");
                                echo "<option value=";
                                echo $ID;
                                if($ID==$term) echo " selected ";
                                echo ">";
                                echo $row["Osastonimi"];
                                echo "</option>";
                                }
                                ?>
                                </select>
                      <td align="right"><input type="submit" value="Suorita haku" />
  </td>
 </tr>
</form>

<tr>
<td height="35" align="right">
<form action="toimipaikka.php" method="get">
                        Toimipaikka:</td>
                        <td><select name="toimipaikka" onchange='this.form.submit()'>
                        <option>Valitse</option>
                        <?
```

Hakulomake koodiesimerkki **Liite 7**
3(4)

```
                while ($row = mysql_fetch_array($sql))

                 {
                        $ID=$row["ToimipaikkaID"];
                        //settype($ID,"integer");
                        echo "<option value=";
                        echo $ID;
                        if($ID==$term) echo " selected ";
                        echo ">";
                        echo $row["Toimipaikkanimi"].", ".$row["Osastonimi"];
                        echo "</option>";
                        }
                        ?>
                        </select>
                <td align="right"><input type="submit" value="Suorita haku" />
  </td>
 </tr>
</form>


<tr>
<td height="35" align="right">
<form action="ynumerot.php" method="get">
                Yleiset numerot:</td>
                <td><select name="ynumerot" onchange='this.form.submit()'>
                <option>Valitse</option>
                <?

                while ($row = mysql_fetch_array($sql2))

                 {
                        $ID=$row["YleisID"];
                        //settype($ID,"integer");
                        echo "<option value=";
                        echo $ID;
                        if($ID==$term) echo " selected ";
                        echo ">";
                                echo $row["Toimipaikkanimi"].", ".$row["Nimike"];
                        echo "</option>";
                        }
                        ?>
                        </select>
                <td align="right"><input type="submit" value="Suorita haku" />
  </td>
 </tr>
</form>
</table>
```

Hakulomake koodiesimerkki **Liite 7**
4(4)

```
<h3 align ="center">Hakutulokset</h3>
<small align ="center"> Huom! Pikanumerot vain kunnan sisäiseen käyttöön. </small>

   </p>

   </body>
</html>
```

SISÄLLYS

1 Johdanto
1.1 Ohjelman tarkoitus
1.2 Ohjelman käyttöympäristö

2 Toiminnot

3 Ongelmatilanteet
3.1 Virheilmoitukset

4 Käyttöohje
4.1 Uuden työntekijän lisääminen
4.2 Työntekijätietojen päivitys ja poisto

5 Työtehtävät
5.1 Uuden työtehtävän lisääminen
5.2 Työtehtävätietojen päivitys ja poisto

6 Osastot
6.1 Uuden osaston lisääminen
6.2 Osastotietojen päivitys ja poisto

7 Yleiset numerot
7.1 Uuden yleisnumeron lisääminen
7.2 Yleisnumerotietojen päivitys ja poisto

8 Käyttäjät
8.1 Uuden käyttäjän lisääminen
8.2 Käyttäjätietojen päivitys ja poisto

Päivityssovelluksen käyttöohje **Liite 8**

2(16)

1. Johdanto

Tässä ohjeessa kerrotaan miten Kolarin kunnan yhteystietohakusovellusta päivitetään ja ylläpidetään selainpohjaisella ylläpitosovelluksella.

1.1 Ohjelman tarkoitus

Ohjelman tarkoitus on helpottaa Kolarin kunnan käytössä olevan yhteystietohakusovelluksen tietokannan ylläpitoa.

1.2 Käyttöympäristö

Varsinainen tietokanta sijaitsee Nebula Oy:n palvelimella ja päivitys-sovellusta käytetään Internet-selaimella, esim Internet Explorer tai Mozilla Firefox. Jotta päivitys onnistuisi on päivittäjällä oltava toimiva Internet-yhteys.

2. Toiminnot

Päivitysohjelmalla voidaan lisätä tietokantaan uusien työntekijöiden yhteystietoja, työtehtävätietoja, toimipaikkatietoja, osastotietoja ja kunnan yleistennumeroiden tietoja tietokantaan. Kaikkia edellämainittuja voidaan myös poistaa ja päivittää.

3. Ongelmatilanteet

Virheilmoitukset

Virhe	Selitys	Toimenpide	
”TyöntekijäID tulee olla täytettynä”	Uuden työntekijän tunnisteID-kentässä ei ole numeroa.	Jos kenttä on tyhjä, kirjoita siihen seuraava vapaa ID-numero.	
”TyöntekijäID on jo olemassa”	TyöntekijäID-kentässä oleva numero on jo käytössä.	kirjoita TyöntekijäID-kenttään seuraava vapaa ID-numero.	

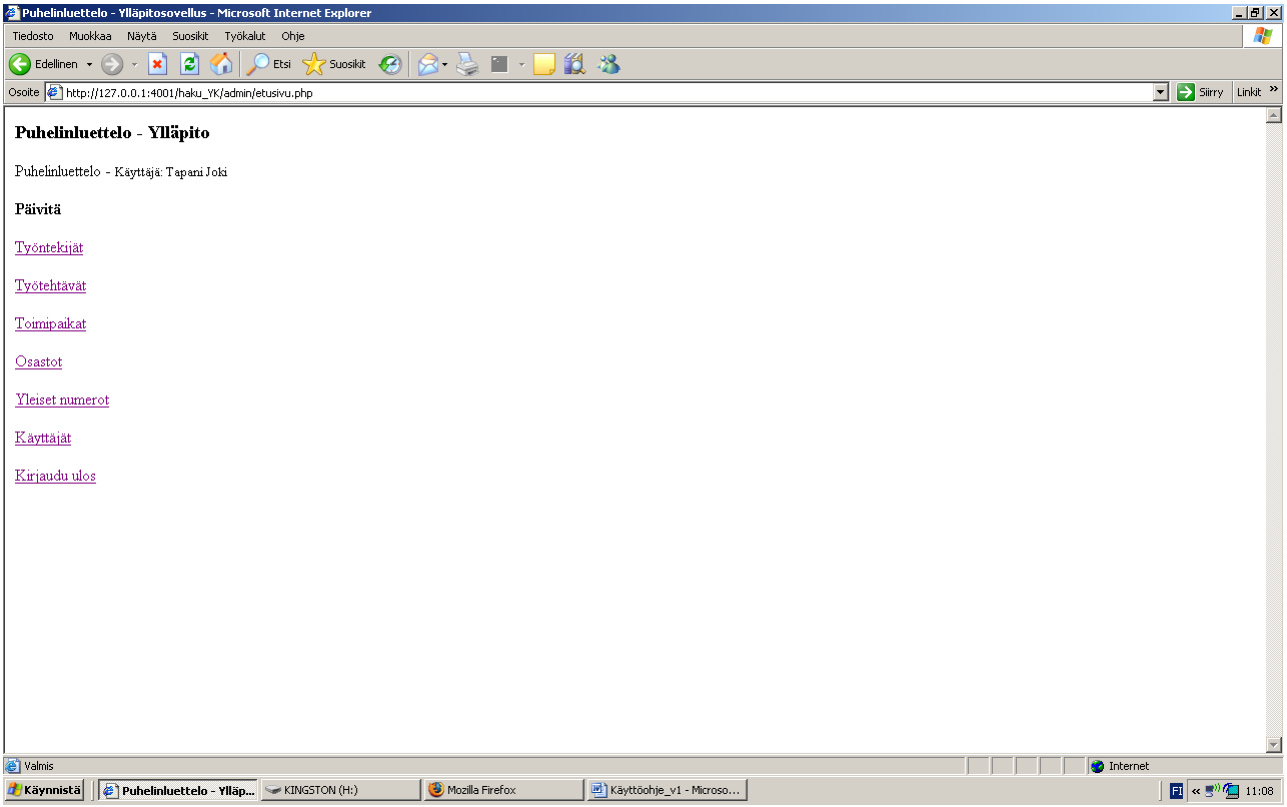
Tyypillisimmät ongelmatilanteet ja niistä selviytyminen.

K: Jos internet-yhteys katkeaa päivityksen aikana?
V: Jos tietoja ei ole ehditty tallentaa, katoat joten tallennuksia on viisainta tehdä pienellä välillä.

Käyttöohje

Kirjautuminen ylläpito-sovellukseen:

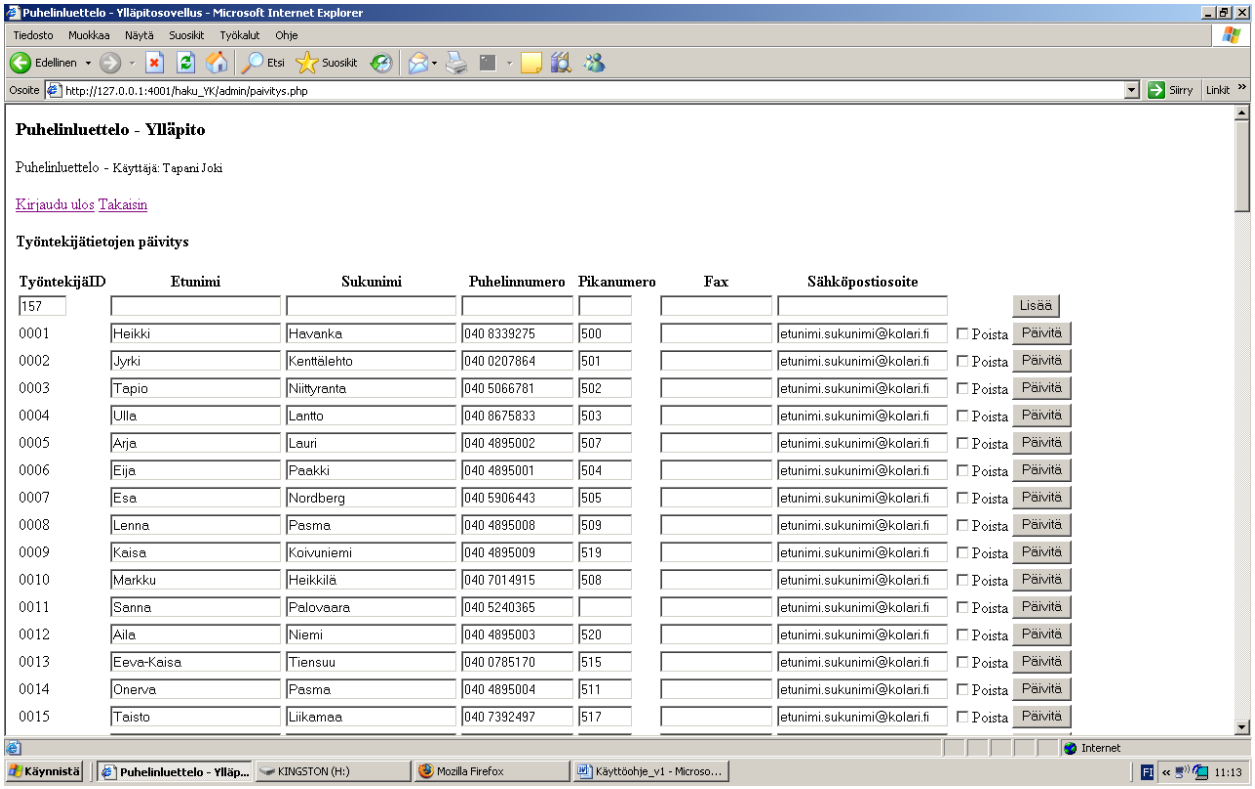
Ylläpito-sovellukseen pääsee kirjautumaan hakusovellus-sivulla sijaitsevan ”ylläpito”-linkin kautta. Avautuvalla kirjautumissivulla kirjautujan on syötettävä Tunnus ja salasana niille osoitettuihin kenttiin ja painettava ”kirjaudu”-Painiketta. Jos tunnus ja salasana ovat oikein, käyttäjä saa eteensä seuraavan näkymän:



Avautuvalta sivulta käyttäjä voi valita, mitä osaa haluaa päivittää. Ensimmäiseksi käsitellään uuden työntekijän lisääminen ja päivittäminen.

4.Työntekijät

Kun käyttäjä painaa etusivulta ”Työntekijät”-linkkiä, avautuu seuraava näkymä. Tällä sivulla käyttäjä voi lisätä uden työntekijän tietokantaan tai muokata jo tietokannassa olevien työntekijöiden yhteystietoja.



4.1 Uuden työntekijän lisääminen:

Avautuvalla sivulla ensimmäisenä näkyy TyöntekijäID-numerokenttä, jonka ohjelma on automaattisesti täyttänyt. Numero on järjestyksessä seuraava numero listassa. TyöntekijäID-numeron voi muuttaa, mutta suositeltavaa on antaa sen olla ehdotettu numero. Seuraavaksi käyttäjä voi täyttää uuden työntekijän varsinaiset tiedot, kuten Etu- ja sukunimen, puhelinnumeron, pikanumeron, fax-numeron ja sähköpostiosoitteen. Kun tiedot on täytetty, käyttäjä painaa rivin päässä olevaa ”Lisää”-painiketta ja tiedot tallentuvat tietokantaan. varmistuksena onnistuneelle

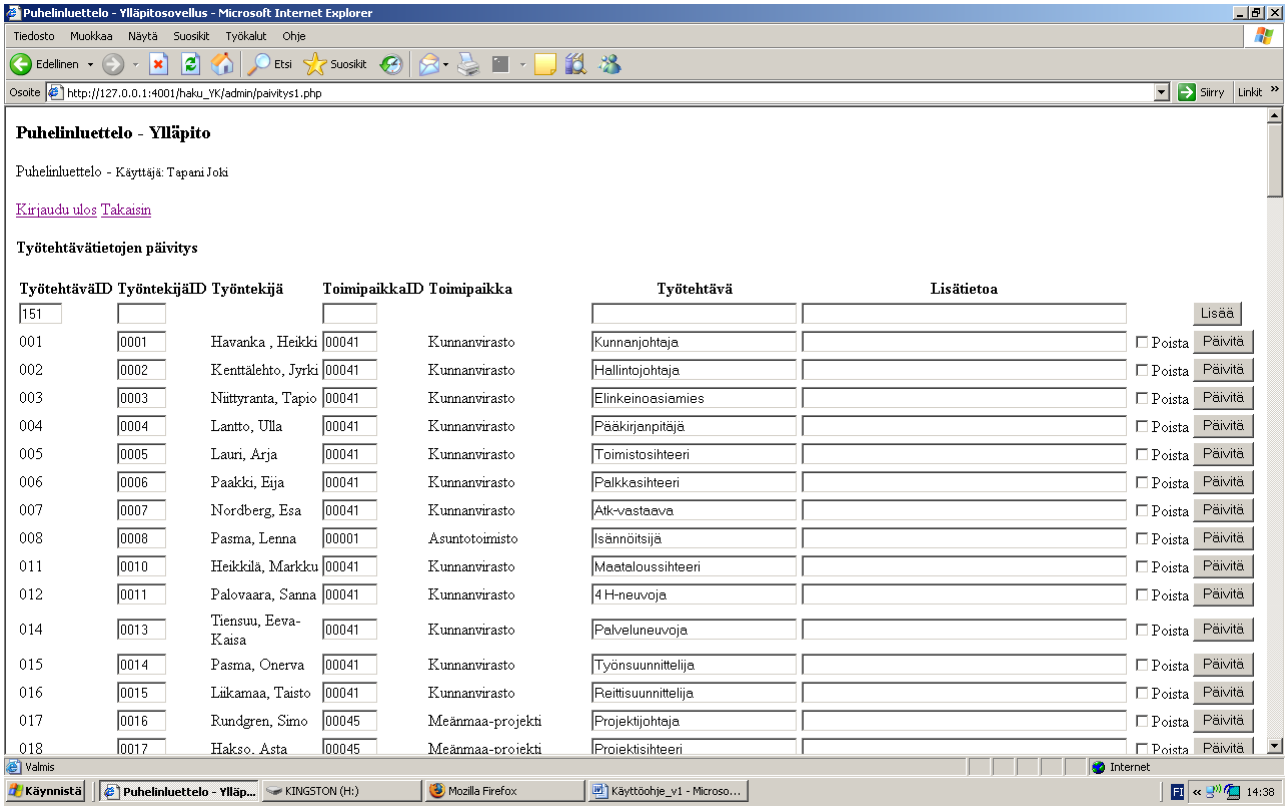
tallennukselle ohjelma antaa vastauksen ”Työntekijä on lisätty tietokantaan” ja linkin takaisin edelliselle sivulle. Jos muita lisättäviä työntekijöitä ei ole, ”Takaisin”-linkistä pääsee aloitussivulle tai ”Kirjaudu ulos”-linkistä voi kirjautua ulos koko ylläpitosovelluksesta.

4.2 Työntekijätietojen päivitys ja poisto:

Työntekijätietojen päivitys ja poisto tapahtuu samalta sivulta kuin uuden työntekijän lisääminenkin. kaikkia muita tietoja voidaan muuttaa, paitsi TyöntekijäID-numeroa. Kun tarvittavat muutokset tietoihin on tehty, käyttäjä painaa päivitettävän rivin päässä olevaa ”Päivitä”-painiketta. Varmistuksena onnistuneelle päivitykselle ohjelma antaa vastauksen ”Työntekijätiedot on päivitetty” ja linkin takaisin edelliselle sivulle. Työntekijätietojen poisto tapahtuu siten, että poistettavan rivin perässä oleva ”poista”-kenttä merkitään ja painetaan ”päivitä”-painiketta. Varmistuksena onnistuneesta poistosta ohjelma antaa vastauksen ”Työntekijä on poistettu”ja linkin takaisin edelliselle sivulle.

5. Työtehtävät

Kun käyttäjä painaa etusivulta ”Työtehtävät”-linkkiä, avautuu seuraava näkymä. Tällä sivulla käyttäjä voi lisätä uuden työtehtävän tietokantaan, tai muokata jo tietokannassa olevia työtehtävätietoja.



5.1 Uuden työtehtävän lisääminen

Avautuvalla sivulla ensimmäisenä näkyy TyötehtäväID-numerokenttä, jonka ohjelma automaattisesti täyttänyt. Seuraavana näkyy tyhjä TyöntekijäID-numerokenttä, johon käyttäjä lisää haluamansa työntekijän TyöntekijäID-numeron. Seuraavaksi käyttäjä voi lisätä ToimipaikkaID-numeron sille varattuun kenttään ID-kentän oikealla puolella näkyy numeroon yhdistyvä toimipaikan nimi. Seuraavaan kenttään voidaan lisätä työtehtävän nimi ja viimeiseksi työtehtävään liittyvää mahdollista lisätietoa. Kun tarvittavat tiedot lisätty, käyttäjä painaa rivin päässä olevaa ”Lisää”-painiketta ja tiedot tallentuvat tietokantaan. Varmistuksena onnistuneelle tallennukselle ohjelma antaa vastauksen”Työtehtävä lisätty tietokantaan” ja linkin takaisin edelliselle sivulle. Jos muita lisättäviä työthetäviä ei ole, ”Takaisin”-linkistä

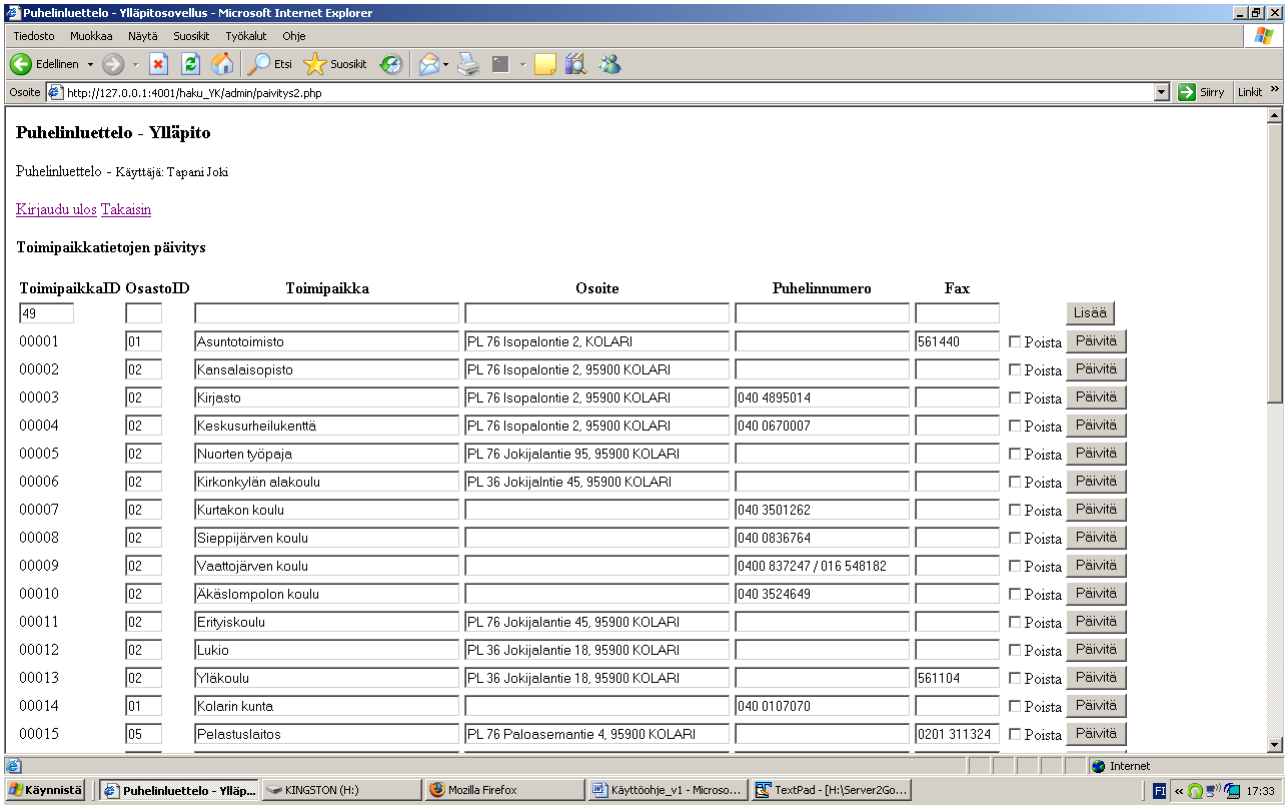
pääsee aloitussivulle, tai ”Kirjaudu ulos”-linkistä voi kirjautua ulos koko ylläpitosovelluksesta.

5.2 Työtehtävätietojen päivitys ja poisto

Työtehtävätietojen päivitys ja poisto tapahtuu samalta sivulta kuin uuden työtehtävän lisääminenkin. Kaikkia muita tietoja voidaan muuttaa paitsi TyötehtäväID-numeroa. Kun tarvittavat muutokset tietoihin ovat tehty, käyttäjä painaa päivitettävän rivin päässä olevaa ”Päivitä”-painiketta. Varmistuksena onnistuneelle päivitykselle ohjelma antaa vastauksen ”Työtehtävä tiedot on päivitetty” ja linkin takaisin edelliselle sivulle. Työtehtävien poisto tapahtuu siten, että poistettavan rivin päässä oleva ”Poista”-kenttä merkitään ja käyttäjän painaessa ”päivitä”-painiketta kyseinen työtehtävä poistetaan tietokannasta. Varmistuksena onnistuneesta poistosta ohjelam antaa vastauksen ”Työtehtävä on poistettu” ja linkin takaisin edelliselle sivulle.

6. Toimipaikat

Kun käyttäjä painaa etusivulta ”Toimipaikat”-linkkiä, avautuu seuraava näkymä. Tällä sivulla käyttäjä voi lisätä uuden toimipaikan tietokantaan, tai muokata jo tietokannassa olevia toimipaikkatietoja.



6.1 Uuden toimipaikan lisääminen

Avautuvalla sivulla ensimmäisenä näkyy ToimipaikkaID-numerokenttä, jonka ohjelma automaattisesti täyttänyt. Seuraavana näkyy tyhjä ToimipaikkaID-numerokenttä, johon käyttäjä lisää haluamansa työntekijän ToimipaikkaID-numeron. Seuraavaksi käyttäjä voi lisätä OsastoID-numeron sille varattuun kenttään. OsastoID-kentän oikealla puolella on kenttä toimipaikan nimeä varten. Seuraavaan kenttään voidaan lisätä toimipaikan osoite. Seuraavaan kenttään voidaan lisätä toimipaikan puhelinnumero ja viimeiseksi mahdollinen Fax-numero.. Kun tarvittavat tiedot lisätty, käyttäjä painaa rivin päässä olevaa ”Lisää”-painiketta ja tiedot tallentuvat tietokantaan. Varmistuksena onnistuneelle tallennukselle ohjelma antaa vastauksen”Toimipaikka lisätty tietokantaan” ja linkin takaisin edelliselle sivulle. Jos muita

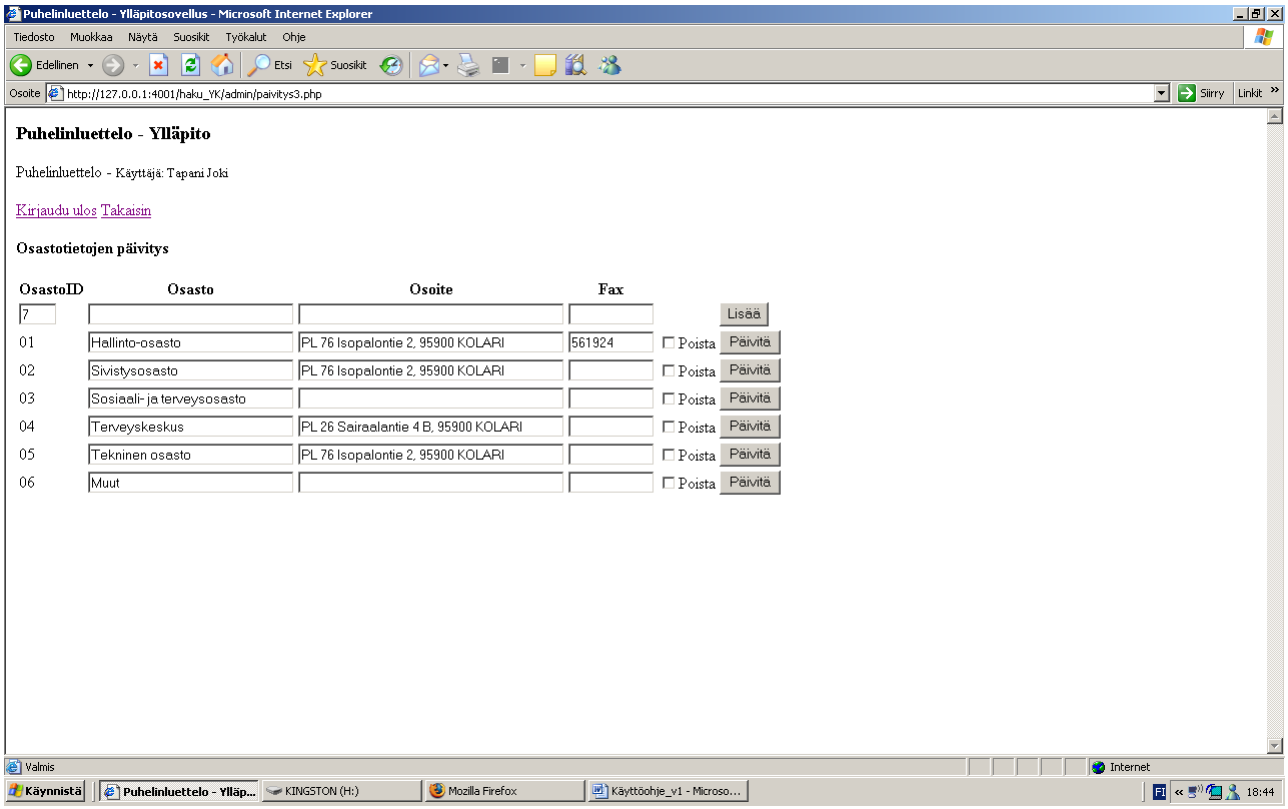
lisättäviä toimipaikkoja ei ole, ”Takaisin”-linkistä pääsee aloitussivulle, tai ”Kirjaudu ulos”-linkistä voi kirjautua ulos koko ylläpitosovelluksesta.

6.2 Toimipaikkatietojen päivitys ja poisto

Toimipaikkatietojen päivitys ja poisto tapahtuu samalta sivulta kuin uuden toimipaikan lisääminenkin. Kaikkia muita tietoja voidaan muuttaa paitsi ToimipaikkaID-numeroa. Kun tarvittavat muutokset tietoihin ovat tehty, käyttäjä painaa päivitettävän rivin päässä olevaa ”Päivitä”-painiketta. Varmistuksena onnistuneelle päivitykselle ohjelma antaa vastauksen ”Toimipaikka tiedot on päivitetty” ja linkin takaisin edelliselle sivulle. Toimipaikkojen poisto tapahtuu siten, että poistettavan rivin päässä oleva ”Poista”-kenttä merkitään ja käyttäjän painaessa ”päivitä”-painiketta kyseinen Toimipaikka poistetaan tietokannasta. Varmistuksena onnistuneesta poistosta ohjelam antaa vastauksen ”Toimipaikka on poistettu” ja linkin takaisin edelliselle sivulle.

7. Osastot

Kun käyttäjä painaa etusivulta ”Osastot”-linkkiä, avautuu seuraava näkymä. Tällä sivulla käyttäjä voi lisätä uuden Osaston tietokantaan, tai muokata jo tietokannassa olevia osastotietoja.



7.1 Uuden osaston lisääminen

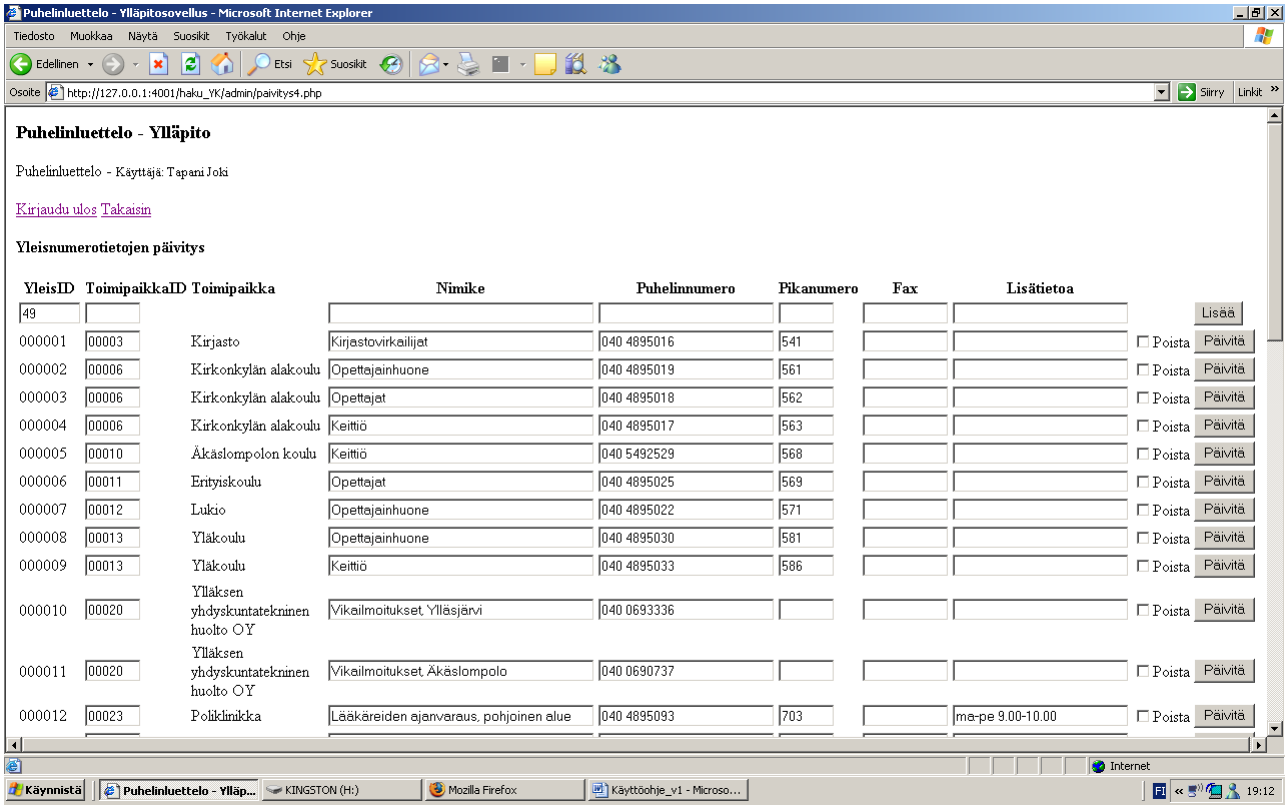
Avautuvalla sivulla ensimmäisenä näkyy OsastoID-numerokenttä, jonka ohjelma on automaattisesti täyttänyt. Seuraavana näkyy tyhjä Osasto-kenttä, johon käyttäjä lisää uuden osaston nimen. Seuraavaksi käyttäjä voi lisätä uuden osaston mahdollisen osoitteen ja Fax-numeron niille osoitettuihin kenttiin. Kun tarvittavat tiedot on lisätty, käyttäjä painaa rivin päässä olevaa ”Lisää”-painiketta ja tiedot tallentuvat tietokantaan. Varmistuksena onnistuneelle tallennukselle ohjelma antaa vastauksen”Osasto lisätty tietokantaan” ja linkin takaisin edelliselle sivulle. Jos muita lisättäviä osastoja ei ole, ”Takaisin”-linkistä pääsee aloitussivulle, tai ”Kirjaudu ulos”-linkistä voi kirjautua ulos koko ylläpitosovelluksesta.

7.2 Osastotietojen päivitys ja poisto

Osastotietojen päivitys ja poisto tapahtuu samalta sivulta kuin uuden osaston lisääminenkin. Kaikkia muita tietoja voidaan muuttaa paitsi OsastoID-numeroa. Kun tarvittavat muutokset tietoihin ovat tehty, käyttäjä painaa päivitettävän rivin päässä olevaa ”Päivitä”-painiketta. Varmistuksena onnistuneelle päivitykselle ohjelma antaa vastauksen ”Osastotiedot on päivitetty” ja linkin takaisin edelliselle sivulle. Osastojen poisto tapahtuu siten, että poistettavan rivin päässä oleva ”Poista”-kenttä merkitään ja käyttäjän painaessa ”päivitä”-painiketta kyseinen osasto poistetaan tietokannasta. Varmistuksena onnistuneesta poistosta ohjelma antaa vastauksen ”Osasto on poistettu” ja linkin takaisin edelliselle sivulle.

8. Yleiset numerot

Kun käyttäjä painaa etusivulta ”Yleiset numerot”-linkkiä, avautuu seuraava näkymä. Tällä sivulla käyttäjä voi lisätä uuden Yleisen numeron, eli numeron joka ei ole osoitettu varsinaiselle henkilölle, tietokantaan tai muokata jo tietokannassa olevia yleisnumerotietoja.



8.1 Uuden yleisnumeron lisääminen

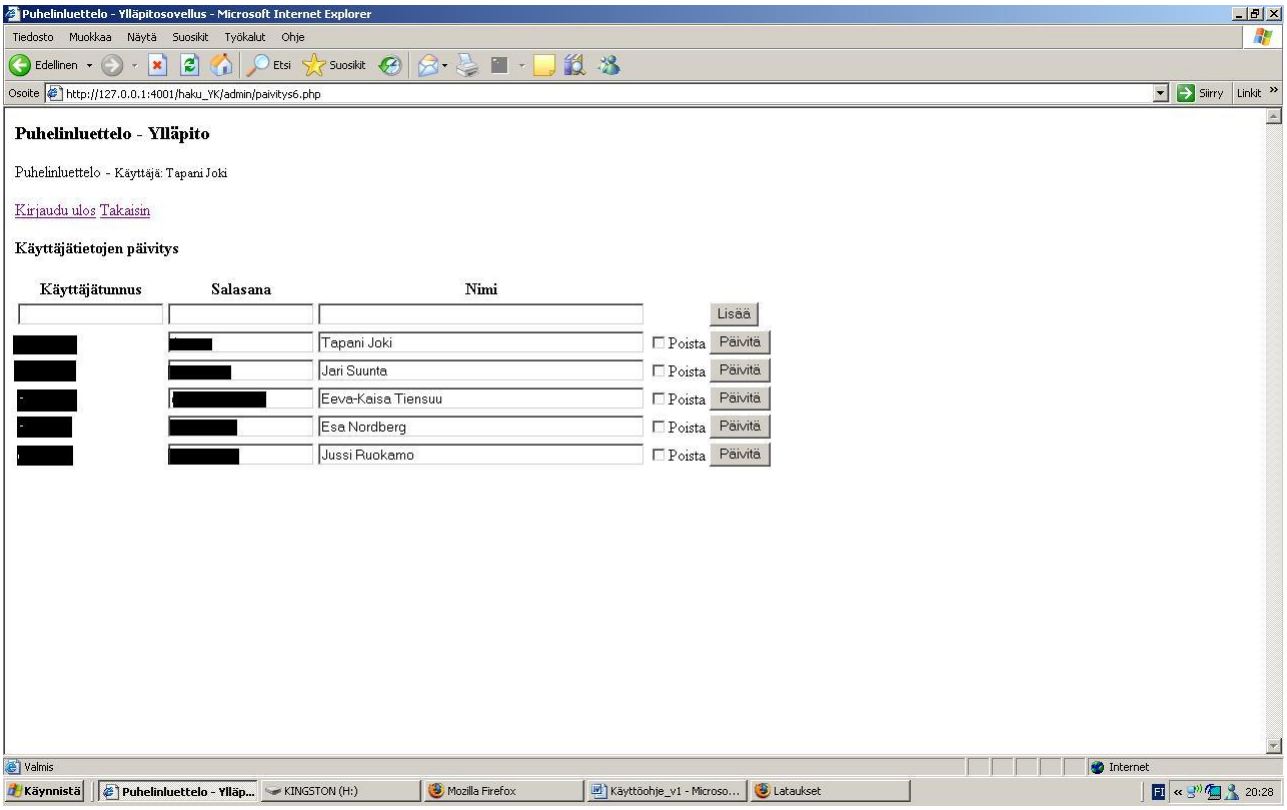
Avautuvalla sivulla ensimmäisenä näkyy YleisID-numerokenttä, jonka ohjelma on täyttänyt automaattisesti. Seuraavaksi käyttäjä lisää ToimipaikkaID-numeron sille varattuun kenttään. Numeron oikealta puolella näkyy numeroihin liittyvät toimipaikkanimet. Seuraavaksi käyttäjä lisää uuden yleisnumeron nimikkeen sille varattuun kenttään, kuten myös puhelin-, ja pikanumeron sekä mahdollisen Fax-numeron. Viimeiseen vapaaseen kenttään käyttäjä voi lisätä mahdollista lisätietoa uudesta yleisnumerosta. Kun tiedot on lisätty, käyttäjä painaa rivin päässä olevaa ”Lisää”-painiketta ja tiedot tallentuvat tietokantaan.

8.2 Yleisnumerotietojen päivitys ja poisto

Yleisnumerotietojen päivitys ja poisto tapahtuu samalta sivulta kuin uuden Yleisnumeron lisääminenkin. Kaikkia muita tietoja voidaan muuttaa paitsi YleisID-numeroa. Kun tarvittavat muutokset tietoihin ovat tehty, käyttäjä painaa päivitettävän rivin päässä olevaa ”Päivitä”-painiketta. Varmistuksena onnistuneelle päivitykselle ohjelma antaa vastauksen ”Yleisnumerot on päivitetty” ja linkin takaisin edelliselle sivulle. Yleisnumerojen poisto tapahtuu siten, että poistettavan rivin päässä oleva ”Poista”-kenttä merkitään ja käyttäjän painaessa ”päivitä”-painiketta kyseinen osasto poistetaan tietokannasta. Varmistuksena onnistuneesta poistosta ohjelma antaa vastauksen ”Yleisnumero on poistettu” ja linkin takaisin edelliselle sivulle.

9. Käyttäjät

Kun käyttäjä painaa etusivulta ”Käyttäjät”-linkkiä, avautuu seuraava näkymä. Tällä sivulla käyttäjä voi lisätä uuden päivityssovelluksen käyttäjän tai muokata jo tietokannassa olevia käyttäjätietoja.Seuraavassa kuvassa käyttäjätunnukset ja salasanat on piilotettu mustalla suorakaiteella.



9.1 Uuden käyttäjän lisääminen.

Ensimmäisenä lisätään Uuden käyttäjän kayttäjätunnus sille varattuun kenttään. Seuraavana vuorossa on salasana ja mahdollisesti käyttäjän nimi. käyttäjätunnuksella ja salasanalla ei ole vähimmäispituutta.

9.2 Käyttäjätietojen päivitys ja poisto.

Käyttäjätietojen päivitys ja poisto tapahtuu samalta sivulta kuin uuden käyttäjätiedon lisääminenkin. Tietojen muuttamisen jälkeen käyttäjä painaa rivin päässä olevaa ”päivitä”-painiketta ja varmistuksena onnistuneelle päivitykselle ohjelma antaa vastauksen ”Käyttäjätiedot on päivitetty” ja linkin takaisin edelliselle sivulle. Käyttäjätiedon poisto tapahtuu siten, että poistettavan rivin päässä oleva ”Poista”-kenttä merkitään ja käyttäjän

painaessan ”Päivitä”-painiketta kyseinen käyttäjätieto poistetaan tietokannasta. Varmistuksena onnistuneesta poistosta ohjelma vastaa ”käyttäjä on poistettu” ja näyttää linkin takaisin edelliselle sivulle.

LÄHTEET

Painetut

Gilmore, Jason 2005. PHP & MySQL – Tehokas hallinta. Gummerus kirjapaino Oy, Jyväskylä.

Heinisuo, Rami & Rauta, Ilkka 2007. PHP ja MySQL - Tietokantapohjaiset verkkopalvelut. Gummerus Kirjapaino Oy, Helsinki

Hovi, Ari & Huotari, Jouni & Lahdenmäki, Tapio 2005. Tietokantojen suunnittelu ja indeksointi. Docendo Finland Oy, Jyväskylä

Lahtonen, Tommi 2002. SQL. Docendo Finland Oy, Jyväskylä.

Meloni, Julie 2003. MySQL. Edita Prima Oy, Helsinki.

Rantala, Ari 2002. PHP – Web-ohjelmoinnin peruskirja. Docendo Finland Oy, Jyväskylä.

Painamattomat

HTML 2009. Luettu
<http://fi.wikipedia.org/wiki/HTML>

Internet 2009. Luettu
<http://fi.wikipedia.org/wiki/Internet>

Kollanius, Sami 2005. Johdanto. Luettu
<http://users.jyu.fi/~kolli/ITK215_05/php/>

Kollanius, Sami 2005. PHP ja tietokanta. Luettu
<http://users.jyu.fi/~kolli/ITK215_05/php/?sivu=tietokanta>

Laaksonen, Antti 2003. Käytännön PHP-opas. Luettu
<http://www.ohjelmointiputka.net/opas.php?tunnus=phpj>

MySQL 2009. Luettu
<http://fi.wikipedia.org/wiki/MySQL>

Orasaari, Marko. Malli käyttöohjeen sisällöstä. Luettu
<http://www.cs.helsinki.fi/group/alabra/orasaari/kayttoohje.html>
PHP 2009. Luettu
<http://fi.wikipedia.org/wiki/PHP>
UML-mallinnus 2009. Luettu
<http://fi.wikipedia.org/wiki/UML-mallinnus>
Vanhala-Nurmi, Vuokko. Mikä Internet on. Luettu
<http://myy.helia.fi/~vanvu/tietoliikenne/internetkehys.html>