VAASAN AMMATTIKORKEAKOULU
VASA YRKESHÖGSKOLA
UNIVERSITY OF APPLIED SCIENCES

Samu Lehtonen

# B2B-KONTAKTIPANKKI

Tekniikka ja liikenne
2014

VAASAN AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma

# TIIVISTELMÄ

| | |
|---|---|
| Tekijä | Samu Lehtonen |
| Opinnäytetyön nimi | B2B-kontaktipankki |
| Vuosi | 2014 |
| Kieli | suomi |
| Sivumäärä | 24 |
| Ohjaaja | Pirjo Prosi |

Tämän työn tavoitteena oli luoda LeadDesk Oy:lle B2B kontaktien hakutyökalu, jonka kautta asiakkaat voivat tehdä suodatuksia ja ladata suodatuksen mukaisten kontaktien tiedot.

Sovellus toteutettiin pilvipohjaiseksi palveluksi, jota voi käyttää selaimella. Toteutukseen käytettiin hyväksi suurimmaksi osaksi HTML:ää ja PHP:tä. Sovelluksen taustalla on myös MySQL tietokanta.

Sovellukseen saatiin toteutettua kaikki halutut ominaisuudet ja ensimmäinen versio pystyttiin laittamaan tuotantoon.

Avainsanat B2B, pilvipohjainen, HTML, PHP, MySQL

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Tietotekniikan koulutusohjelma

# ABSTRACT

| | |
|---|---|
| Author | Samu Lehtonen |
| Title | B2B-contactbank |
| Year | 2014 |
| Language | Finnish |
| Pages | 24 |
| Name of Supervisor | Pirjo Prosi |

The goal of this thesis was to create a B2B contact tool for LeadDesk Oy, with which the clients could filter contacts and download contact information based on the chosen filters.

The application is cloud-based and it can be accessed via a browser. Most of the application was build using HTML and PHP. The application also uses a MySQL database.

All of the set requirements were met and the first version of the application could be put into production.

Keywords B2B, cloud-based, HTML, PHP, MySQL

# LYHENTEET

| | |
|---|---|
| B2B | Business-to-business, yritysten välinen kanssakäynti. |
| PHP | PHP: Hypertext Preprocessor, ohjelmointikieli. |
| SQL | Structured Query Language, kyselykieli. |
| HTML | HyperText Markup Language, merkkauskieli. |
| CSS | Cascading Style Sheets, tyylitiedostokieli. |
| URL | Uniform Recource Locator, verkkosivun osoite. |
| CSV | Comma-Separated Values, tiedostomuoto. |

# KUVIOLUETTELO

# SISÄLLYS

# 1 JOHDANTO

Tämä opinnäytetyö tehtiin LeadDesk Oy:lle. Työn tarkoituksena oli kehittää hakutyökalu yrityskontaktien hakemiseen.

Suomessa on ennestään olemassa muutamia palveluntarjoajia, jotka tarjoavat palveluita yrityskontaktien hakemiseen. Tämän työn päämääränä oli kuitenkin kehittää LeadDeskille oma vastaava palvelu, jotta työkalun ominaisuudet ja käyttömahdollisuudet voidaan muokata paremmin yrityksen tarpeita vastaaviksi. LeadDeskillä on jo ennestään olemassa vastaavia palveluita, esimerkiksi kuluttajakontaktien hakemiseen, joten yrityksen kannalta on luonnollista täydentää palveluita mahdollisimman kattaviksi.

LeadDesk on pohjoismaissa sekä Keski-Euroopassa toimiva yritys, jonka päätuotteena on pilvipohjainen soittojärjestelmä. Yritys tarjoaa myös erilaisia oheispalveluita, kuten esimerkiksi haku- ja päivitystyökaluja kontaktien hakemista ja päivittämistä varten sekä mahdollisuutta pitää soitossa olleita kontakteja sekä aiempia soittotietoja tallessa pilvipalvelussa. /3/

Yritys on perustettu vuonna 2009, jolloin se tunnettiin nimellä Comdashboard Oy. LeadDesk Oy:ksi yritys vaihtoi nimensä vuonna 2013. Yrityksellä on toimipaikkoja Suomessa, Ruotsissa, Norjassa, Saksassa ja Puolassa, mutta soittojärjestelmä on käytössä Euroopanlaajuisesti. /3/

# 2 KÄYTETYT TEKNIIKAT

Sovelluksen tarkoituksena oli luoda pilvipohjainen palvelu, jota käytetään selaimen kautta. Sovellus kuitenkin vaatii toimiakseen toimintoja, joita on mahdoton toteuttaa pelkän HTML:n avulla, joten tämän lisäksi sovelluksen rakentamisessa käytettiin hyväksi PHP:tä ja JavaScriptiä. Sovelluksen taustalla on myös tietokanta, josta kontaktien tiedot haetaan, ja jossa olevien tietojen perusteella kontaktijoukkoa voidaan rajata. Tämä tietokanta on toteutettu MySQL:llä.

Kehitysympäristönä tässä työssä käytettiin Eclipseä, johon on ladattavissa ilmainen PHP-lisäosa, ja sovellusta ajettiin Apache web serverillä. Tietokannan hallinnassa apuna käytettiin MySQL Workbenchiä.

## 2.1 PHP

PHP on avoimen lähdekoodin skriptikieli, jota voidaan upottaa HTML:n sisään. Tämä mahdollistaa sen että sivuille voidaan luoda dynaamista sisältöä, joka ei olisi mahdollista pelkästään HTML:n avulla. PHP:n juuret ja syntaksi periytyvät suurelta osin C-, Java- ja Perl-ohjelmointikielistä. /8/

Yksinkertainen sivulle tulostus toteutetaan PHP:llä seuraavasti:

```
<?php

echo 'Hello world';

?>
```

Kuten esimerkistäkin nähdään, PHP:tä täytyy kirjoittaa PHP-tagien sisään. Tagien ulkopuolella oleva sisältö käyttäytyy normaalin HTML-sisällön tavoin. PHP vaatii myös että sitä kirjoitetaan PHP-tiedostoihin, joiden tiedostopääte on .php. Tämän

johdosta PHP-koodi ei toimi normaaleissa HTML-tiedostoissa, mutta HTML-sisältö toimii PHP-tiedostoissa samoin kuin normaalissa HTML-tiedostossa.

Toisin kuin HTML, joka suoritetaan käyttäjän päässä selaimen välityksellä, PHP-koodi suoritetaan palvelimella. Tämän johdosta PHP tarvitsee palvelinohjelman, jonka avulla koodia voidaan suorittaa. Tässä työssä PHP:n ajamiseen käytettiin Apache web serveriä, joka on ilmainen ja avoimen lähdekoodin palvelinohjelma. /8/

## 2.2 MySQL

MySQL on yksi maailman suosituimmista tietokannan hallintajärjestelmistä. Se on lisäksi ilmainen ja sen lähdekoodi on avointa. Tässä työssä sovelluksen taustalla oleva tietomäärä on hyvin suuri, joten tietokannan käyttäminen kaiken kontaktidatan säilyttämiseen ja sen tietojen hyväksikäyttämiseen oli lähes pakollista. MySQL-tietokannan ylläpitoon on myös tarjolla MySQL Workbench-ohjelmisto, jota tässä työssä käytettiin apuna. /7/

MySQL-tietokannat ovat relaatiotietokantoja, joissa tiedot on jaettu tauluihin. Relaatiotietokantamalli perustuu siihen, että tietokannan taulut voivat olla yhteydessä toisiinsa yhteisten tietojoukkojen perusteella. Esimerkkinä tästä voidaan ottaa tilanne, jossa tietokantaan halutaan lisätä yritysten ja työntekijöiden tietoja. Samoja yrityksiä voi olla vain yksi, mutta yrityksellä voi olla useita työntekijöitä. Tällöin tietokantarakenne voidaan muodostaa niin että yritykset ja niiden tiedot ovat omassa taulussaan ja työntekijöiden tiedot ovat omassa taulussaan. Tällöin työntekijätaulu tarvitsee avaimen, jolla oikea työntekijä voidaan yhdistää oikeaan yritykseen.

Tällainen avain voisi olla esimerkiksi yrityksen ID numero. Tämä johtaa siihen että yritysten tietoja ei tarvitse merkitä jokaisen työntekijän kohdalle, vaan yrityksen tiedot voidaan tallentaa omaan tauluunsa vain kerran ja työntekijätaulun avaimen

avulla oikeat työntekijät voidaan yhdistää oikeaan yritykseen. Tällainen tilanne helpottaa myös tietokannan ylläpitoa, koska mahdolliset muutokset yrityksen tietoihin täytyy tehdä vain yhteen paikkaan. Hyvin luodun tietokantarakenteen onkin tarkoitus johtaa siihen, ettei tieto olisi epäjohdonmukaista, irrallaan olevaa, vanhentunutta, puutteellista tai että se sisältäisi duplikaatteja.

Tässä työssä relaatiorakennetta on käytetty hyväksi asiakkaiden ja heidän tilaustensa tunnistamisessa. /7/

### 2.3 Käyttöliittymä

Kaikissa Web-sovelluksissa tärkeänä osana on sivujen ulkoasu ja se millaisena käyttäjä sovelluksen käyttöliittymän näkee, kuinka hän voi sitä käyttää ja kuinka sovellus reagoi käyttäjän haluamalla tavalla. Aiemmin esitelty PHP-kieli ei itsessään ole tekemisissä sovelluksen käyttöliittymän kanssa, vaan sen tarkoituksena on tarjota laajat mahdollisuudet käyttäjän antamien komentojen ja muiden taustalla tapahtuvien prosessien hallintaan.

Aiemmin mainittiin, että PHP-skriptejä voidaan upottaa HTML:n sisään ja HTML onkin nykyisin tärkein merkintäkieli web-dokumenttien kuvaamiseen. HTML-dokumentit koostuvat tageista, jotka määrittelevät kuinka niiden sisällä olevaa sisältöä kuuluisi käsitellä. Web-selaimet tulkitsevat nämä dokumentit ja näyttävät käyttäjälle sivun, joka on muotoiltu käytettyjen tagien ja niissä olevan sisällön mukaisesti. HTML-dokumenteilla on muutamia pakollisia kohtia, jotka niistä pitäisi löytyä, jotta selaimet osaavat käsitellä niitä oikein. Ensimmäisenä on julistus, joka määrittelee dokumentin tyypin, HTML5:n mukainen julistus olisi:

```
<!DOCTYPE html>
```

Web-dokumentin sisältö taas määritellään html- tagien sisään.

```
<html></html>
```

Html- tagien sisällä voidaan body- tageilla määrittää, mikä osa dokumentin sisällöstä halutaan näyttää käyttäjälle:

<body></body>

Nämä esimerkkikohdat muodostavat rungon, jonka varaan HMTL-dokumentit pitäisi rakentaa. /5/

HTML- kieli itsessään määrittelee vain sen, mitä käyttäjälle näytetään, eikä sen avulla voida kovin laajasti määritellä miten sisältö näytetään. Jotta Web- sivujen ulkoasua voidaan laajamittaisesti muokata, pitää avuksi ottaa CSS. CSS:n avulla luodaan Web- sivuille tyyli. HTML:n avulla määritellään mitä komponentteja sivulla näytetään ja CSS:n avulla näiden komponenttien sijoittelua ja ulkoasua voidaan muokata halutunlaiseksi. Tässä työssä käytettiin Bootstrap CSS:ää, joka on ilmainen ja tarjoaa laajan kirjaston yhteneväisen näköisiä tyylejä eri HTML- elementeille. Parhaiten Bootstrap CSS:n käytön huomaa sovelluksessa olevista painikkeista ja talukoista. /1/ /4/

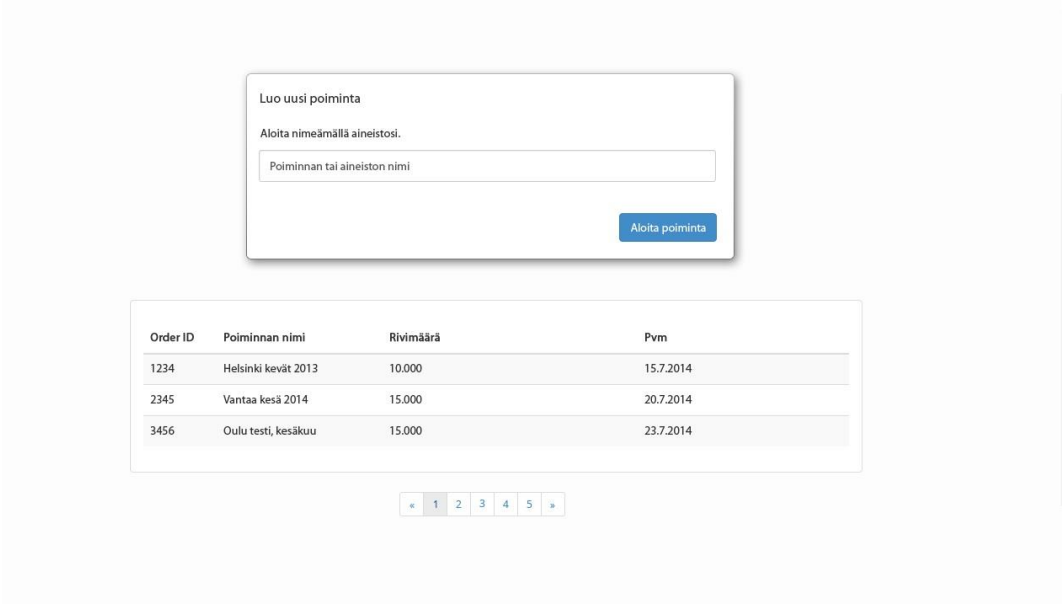
Web- sovelluksissa tulee usein myös tarve pystyä muuttamaan sivujen sisältöä dynaamisesti ilman että, sivua pitäisi ladata uudelleen. PHP on tähän tarkoitukseen riittämätön, koska PHP- skriptit ajetaan aina sivun latauksen yhteydessä, eikä niitä voi enää suorittaa kun sivu on kokonaan ladattu. Tällöin voidaan ottaa käyttöön JavaScript- kieli, joka mahdollistaa HTML-elementtien dynaamisen muokkaamisen. JavaScript on laajalti käytössä, koska toisin kuin PHP, JavaScript ei tarvitse palvelinta, jotta koodia voidaan suorittaa, vaan JavaScript koodi suoritetaan käyttäjän koneella. JavaScriptillä onkin helppo täydentää HTML- sivuja, jotta sisällöstä saadaan helposti tehtyä dynaamista. Tässä työssä JavaScriptiä täydennettiin jQuery-kirjastolla, jonka avulla esimerkiksi etusivulla olevan painikkeen tilaa voitiin muuttaa käyttäjän tekemisen mukaan. /2/ /6/

# 3 TYÖN KUVAUS

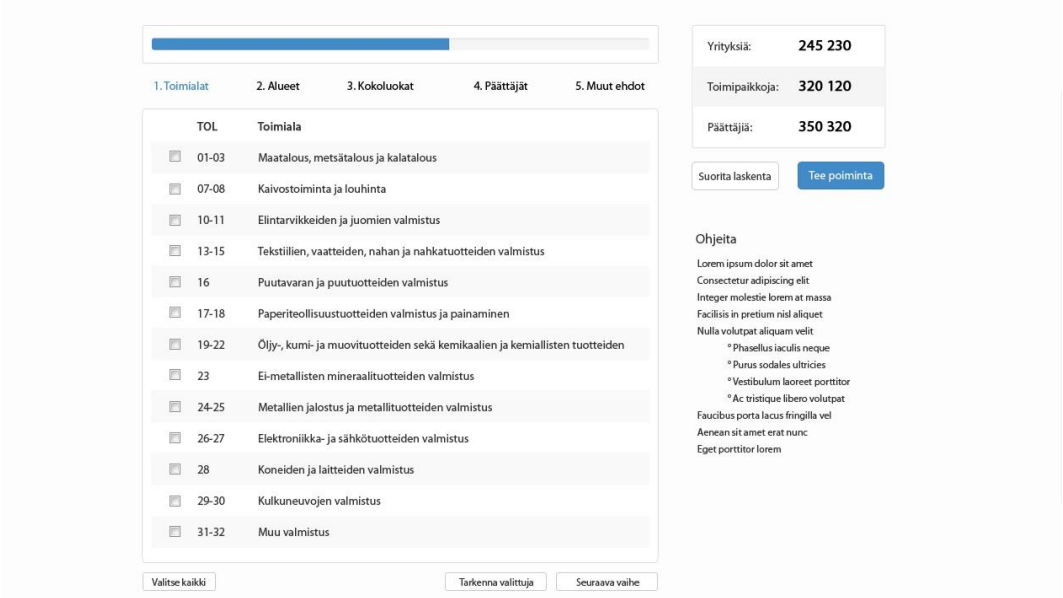
Työn aloituksesta sovittiin LeadDeskin sisäisessä palaverissa, jossa käytiin pääpiirteittään läpi halutut minimiominaisuudet, jotta sovelluksen ensimmäinen versio saataisiin aikataulun mukaisesti tuotantoon ja asiakkaan käyttöön. Samalla sovittiin tulevista jatkokehityksistä. Tässä työssä esitelty sovellus on ensimmäisen tuotantoversion mukainen. Yleisesti sovelluksella olisi tarkoitus pystyä luomaan poimintoja, jotka koostuvat käyttäjän tekemistä valinnoista, joilla yritykset jaetaan halutunlaisiin kohderyhmiin. Kun rajaukset on valittu, voi käyttäjä hakea niitä vastaavien yritysten tiedot tietokannasta ja sovellus tulostaa tiedot CSV- tiedostoon.

## 3.1 Ulkoasu

Tekniseltä näkökannalta LeadDesk antoi sovelluksen toteuttamiseen hyvin vapaat kädet, eikä sovelluksesta sovittu kuin yleinen toimintaperiaate sekä ulkoasu, joka sovittiin toteutettavaksi jo aiemmin mainitun Bootstrap CSS:n mukaisesti. Ulkoasun luomista varten sain myös esimerkkikuvat sovelluksessa käytettävistä kahdesta sivusta, joiden pohjalta ulkoasun luominen oli helppoa. Alhaalla olevissa kuvissa on esitelty sovelluksen sivujen alkuperäiset luonnokset.




**Kuvio 1.** Luonnos etusivusta




**Kuvio 2.** Luonnos poimintasivusta

Annetut esimerkkikuvat olivat kuitenkin enemmänkin suuntaa antavia ja niiden tarkoitus oli tarjota pohja, jonka avulla ulkoasua olisi mahdollisimman helppoa toteuttaa.

### 3.2 Poimintojen rakenne

Sovelluksen pohjaksi annettiin myös tiedot eri valinnoista, joita käyttäjän tulisi pystyä sovelluksessa tekemään ja joiden mukaan poiminnan kohderyhmä muodostetaan. Nämä valinnat on jaettu viiteen eri kohtaan: toimialoihin, alueisiin, kokoluokkiin, päättäjiin ja koontiin. Osa kohdista on vielä jaettu alavalintoihin.

Toimialat- kohta pitää sisällään listauksen toimialoista, joihin yritykset on jaoteltu sovelluksen käyttämässä tietokannassa. Tämän valintakohdan pääsivulla on karkea jaottelu erilaisista toimialoista, joihin yritykset voidaan jaotella. Tässä vaiheessa käyttäjälle tulisi olla mahdollista tarkentaa valitsemiaan toimialoja, jotta hakukriteerit voidaan kohdentaa tarkemmin. Esimerkiksi kaivostoiminta ja louhinta- valinnan alla tarkentavina kohtina ovat metallimalmien louhinta ja muu kaivostoiminta ja louhinta. Näiden kahden kohdan alla on edelleen lisää niiden alle kuuluvia toimialoja.

Alueet- kohdassa käyttäjän tulisi voida rajata yrityksiä niiden maantieteellisen sijainnin perusteella. Myös tässä kohdassa käyttäjän tulisi voida edelleen tarkentaa pääsivulla tekemiään valintoja. Tämän vaiheen valinnat on karkeasti jaoteltu niin, että etusivulla käyttäjä valitsee maakunnan ja tarkennusvalintoina poimintaa on edelleen mahdollista tarkentaa kuntatasolla.

Kokoluokat- vaiheen rajaukset on jaoteltu neljään alatasoon. Yrityksen henkilökuntaluokka- rajaukset antavat käyttäjän rajata poimintaa yrityksen henkilökunnan määrän mukaan. Toimipaikan henkilökuntaluokka- rajaus antaa mahdollisuuden rajata poimintaa toimipaikan henkilökunnan määrän mukaan. Yrityksen liikevaihtoluokka- rajauksessa käyttäjä voi rajata poimintaa yrityksen liikevaihdon mukaan. Toimipaikkaluokka- rajauksessa taas on mahdollista rajata poimintaa sen mukaan minkä tyyppiset toimipaikat hakuun halutaan sisällyttää. Toimipaikat jaotellaan sen mukaan, ovatko ne päätoimipaikkoja, sivutoimipaikkoja vai yksitoimipaikkoja. Kokoluokat- rajauksissa ei ole tarkentavia valintoja, toisin kuin aiemmissa vaiheissa.

Päättäjät- vaiheessa käyttäjän tulisi olla mahdollista rajata poimintaa edelleen sen mukaan minkä tyyppisiä kontakteja yrityksestä halutaan ottaa hakuun mukaan. Tässä työssä tämä vaihe on kuitenkin poistettu käytöstä, koska asiakkaalle tarjotaan sovelluksesta vain versiota, josta on mahdollisuus hakea yrityksen yleisiä yhteystietoja, ilman tietoa päättäjistä. Tämä vaihe on kuitenkin lisätty sovellukseen koska sovellusta tullaan jatkokehittämään niin, että lopulta poimintaa olisi tarkoitus pystyä rajaamaan myös näiden tietojen mukaan.

Koonti-vaiheessa käyttäjän ei ole enää mahdollista rajata poimintaa, vaan nimensä mukaisesti tässä vaiheessa käyttäjän on mahdollista tarkistaa muissa vaiheissa tehdyt valinnat, joiden pohjalta poiminnan kohderyhmä muodostetaan. Koska käyttäjän on mahdollista tehdä useita valintoja, tulisi koonti tulostaa mahdollisimman selkeästi, jotta käyttäjän olisi helppoa tarkistaa valitsemansa suodattimet. Koska useissa poimintavaiheissa on mahdollista tehdä tarkentavia valintoja, tulisi tämä ottaa huomioon kun koonti tulostetaan, jotta pää- ja alavalinnat olisi listauksessa helppo erottaa toisistaan.

Kun käyttäjä on edennyt kaikista vaiheista läpi, tulisi hänelle tarjota mahdollisuus aloittaa poiminta valittujen suodattimien kanssa. Käytännössä tämä tarkoittaa sitä, että käyttäjän olisi mahdollista suorittaa laskenta, joka laskee käytössä olevien suodattimien mukaiset yritykset ja tämän laskennan tulos tulisi tulostaa käyttäjälle, jotta hän tietää kuinka suuren tulosjoukon valitut suodattimet muodostavat. Laskennan jälkeen käyttäjälle tulisi tarjota mahdollisuus ladata CSV- tiedosto, johon tulostetaan poiminnassa mukana olevat yritykset. Listan lataaminen on poiminnan viimeinen vaihe.

# 4 TOTEUTUS

Sovelluksen toiminta on hyvin suoraviivaista. Käyttäjän tulisi aloittaa sovelluksen käyttäminen etusivulta, jonka kautta uusi poiminta olisi mahdollista aloittaa antamalla uudelle poiminnalle nimi. Etusivulla näytettäisiin myös listaus viimeisimmistä poiminnoista, jotka käyttäjä on tehnyt. Kun käyttäjä on antanut uudelle poiminnalleen nimen ja jatkanut eteenpäin, ohjattaisiin hänet poimintasivulle, jossa hän kävisi edellä esitellyt poimintavaiheet läpi ja tekisi valinnat, joiden mukaan poiminnan tulosjoukko rajataan, suorittaisi tulosjoukon laskemisen ja lataisi tulostiedoston. Näiden määritelmien perusteella sovellusta lähdettiin rakentamaan.

## 4.1 Sovelluksen etusivu ja poiminnan aloittaminen

Sovelluksen etusivun rakenne on hyvin yksinkertainen, joten sitä lähdettiin toteuttamaan suoraan annetun luonnoksen pohjalta. Näkyvimpinä komponentteina sivulle luotiin tekstikenttä poiminnan nimeä varten, painike poiminnan aloittamista varten sekä taulukko, johon on koottu käyttäjän viimeisimmät poiminnat. Tämän taulukon sisältö luetaan tietokannasta, johon poiminnat tallennetaan aina listan latauksen yhteydessä. Tietokantahauissa on tässä sovelluksessa käytetty PHP:n mysqli- rajapintaa. Sivulla oleva sisältö on aseteltu yhden koko sivun sisällön kattavan divin sisään. Tämän divin sisältö on edelleen jaoteltu kahden pienemmän divin sisään. Ylempi näistä diveistä pitää sisällään sivulla olevan tekstin, tekstikentän ja painikkeen ja alemman divin sisältönä on käyttäjän aiemmat poiminnat näyttävä taulukko. Tekstikentän ja painikkeen sisällään pitämä div on lisäksi muotoiltu CSS:n avulla pyöreäkulmaiseksi ja sillä on taustavarjo. Tämän on tarkoitus antaa sivulle hieman lisäilmettä.

Käyttäjälle hieman näkymättömämpi ominaisuus etusivulla on käyttäjän tunnistaminen URL:ssa annetulla GET-parametrilla. Tämän parametrin nimi on id ja jotta käyttäjä lopulta voisi ladata sovelluksesta CSV- tiedoston, tulisi tämän parametrin arvon vastata jonkin sovelluksen tietokannassa määritellyn käyttäjän vastaavaa arvoa. Käyttäjien tunnistaminen toteutettiin ennen kaikkea sen takia, ettei listoja voi

ladata kuin tietokannassa määritetyt käyttäjät mutta myös sen takia että käyttäjien tekemiä latauksia voidaan seurata. Tämä mahdollistaa käyttäjäkohtaisten aiempien latausten tulostamisen etusivulle.

Ennen poiminnan aloittamista käyttäjää vaaditaan antamaan poiminnalleen nimi, sivulla olevaan tekstikenttään. Painike poiminnan aloittamiseen pysyy pois käytöstä, jos sovellus havaitsee että tekstikenttä on tyhjä. Tämä toiminnallisuus on toteutettu jQuery:n avulla, joka tarkkailee tekstikenttään tehtäviä muutoksia. Aina kun tekstikentässä painetaan jotain näppäimistönpainiketta sovellus tarkistaa onko tekstikenttä tyhjä. Jos kenttä havaitaan tyhjäksi, painike muutetaan pois käytöstä- tilaan. Vastaavasti jos tekstikentässä havaitaan olevan tekstiä, painike muutetaan käytössä- tilaan. Kun käyttäjä on antanut poiminnalleen nimen voi hän painiketta painamalla jatkaa poimintaan. Alhaalla olevassa kuviossa on esitelty jQuery-funktio, joka huolehtii painikkeen tilan muuttamisesta.

```
$(document).ready(function(){
    $('input[type="submit"]').attr('disabled','disabled');
    $('input[type="text"]').keyup(function(){
        if($('input[type="text"]').val() == ""){
            $('input[type="submit"]').attr('disabled','disabled');
        }
        else{
            $('input[type="submit"]').removeAttr('disabled');
        }
    })
});
```

**Kuvio 3.** Painikkeen tilan muuttaminen tekstikentän tilan mukaan

### 4.2 Poimintasivu ja poimintojen muodostaminen

Kun käyttäjä on antanut poiminnalleen nimen ja painanut etusivun painiketta, ohjataan hänet poimintasivulle. Tällä sivulla käyttäjän on mahdollista tehdä aiemmin esiteltyjen vaiheiden mukaisia suodatuksia, suorittaa laskenta ja tulostaa valittujen suodattimien mukainen tulosjoukko CSV- tiedostoon. Poimintasivulle tultaessa sovellus tekee vielä ylimääräisen varmennuksen, että poiminnalle on etusivulle

varmasti annettu nimi. Jos nimeä ei löydy, käyttäjä ohjataan takaisin etusivulle. Sen lisäksi, että tällä tarkistuksella varmistetaan se että etusivulla poiminnalle on varmasti annettu nimi, varmistetaan myös se, että sovelluksen käyttö aloitetaan aina pääsivun kautta.

Poimintasivun rakenne on hieman monimutkaisempi kuin pääsivun. Tälläkin sivulla, kuten etusivulla, on yksi kaiken sivun sisällön sisällään pitämä div. Tämä div on määritelty 900 pikseliä leveäksi ja sen marginaalit vasemmalle ja oikealle on asetettu automaattisiksi, jolloin sisältö pysyy aina sivun keskellä eikä divin sisällä olevien komponenttien paikka muutu, jos käyttäjä esimerkiksi pienentää selainikkunaa. Tämän divin sisältö on jaettu kahtia niin, että vasemmalla puolen sisällön sisällään pitämä div on leveydeltään 550 pikseliä ja oikean puolen sisällön sisällään pitämä div 250 pikseliä leveä. Nämä kaksi diviä on asetettu CSS:n avulla kellumaan, jolloin ne on mahdollista asettaa rinnakkain.

Sovelluksen komponentit on sijoitettu näiden kahden divin sisään niin, että vasemmalle puolelle tulostetaan käyttäjän etusivulla antama poiminnan nimi sekä kaksi painiketta jotka mahdollistavat suodattimien tyhjäämisen ja etusivulle palaamisen. Näiden alla on poimintavaiheen ilmaiseva palkki, jossa käyttäjän nykyinen poimintavaihe näytetään korostettuna. Palkin alle on sijoitettu div, jolle on annettu leveydeksi 550 pikseliä ja korkeudeksi 400 pikseliä. Tämän divin sisään tulostetaan kunkin poimintavaiheen mukaiset suodattimet ja valintalaatikot taulukkomuodossa. Div on määritelty kiinteänkokoiseksi, jotta sivun asettelu pysyy yhtenäisenä sovelluksen käytön ajan. Tämän divin alle on aseteltu painikkeet, joilla käyttäjä voi valita tai poistaa valinnat valintalaatikoista, tarkentaa valitsemiaan suodattimia tai siirtyä poiminnassa eteen- tai taaksepäin. Sivun oikeanpuoleiseen diviin on sijoitettu kentät, jotka ilmaisevat tietokannassa olevien tietojen määrää. Näiden tietojen alla on kaksi painiketta, joilla käyttäjä lopulta voi suorittaa laskennan valituilla suodattimilla ja lopulta ladata CSV-tiedoston. Painikkeiden alla on lisäksi ohjeita poiminnan helpottamiseksi.

Sovelluksen käyttöliittymä on käyttäjän kannalta sovelluksen näkyvin osa. Sovellukseen piti kuitenkin toteuttaa monia käyttäjälle vähemmän näkyviä ominaisuuksia, jotta poimintojen muodostaminen onnistuu. Sovelluksen pitää pystyä seuraamaan missä poimintavaiheessa tai poiminnan tarkennusvaiheessa käyttäjä milläkin hetkellä on, sekä poistaa mahdollisuus siirtyä vaiheissa taaksepäin jos käyttäjä on jo ensimmäisessä vaiheessa tai vastaavasti poistaa mahdollisuus siirtyä eteenpäin jos käyttäjä on viimeisessä vaiheessa. Sovelluksen on myös pystyttävä tallentamaan käyttäjän tekemät valinnat kun hän liikkuu poimintavaiheiden välillä. Tallennettuja valintoja käytetään koonnin luomiseen koonti-vaiheessa, sekä tietokantahakujen muodostamiseen.

Sovellus tarkkailee käyttäjän liikkumista sovelluksessa poimintavaihe- muuttujan sekä tarkennusvaihe- muuttujan avulla. Poimintasivulle tultaessa poimintavaihe asetetaan aina yhteen, mikä siis vastaa poiminnan ensimmäistä vaihetta. Tämä lukuarvo muuttuu sen mukaan, monennessako vaiheessa käyttäjä milläkin hetkellä on. Kun poimintavaihe on yksi, ei käyttäjä voi liikkua vaiheissa taaksepäin ja vastaavasti kun poimintavaihe on viisi, ei käyttäjä voi enää liikkua vaiheissa eteenpäin. Poimintavaihe- muuttujan lisäksi on käytössä tarkennusvaihe-muuttuja. Tämä muuttuja tarkkailee onko käyttäjä poimintavaiheen pää- vai tarkennussivulla. Jos käyttäjä on pääsivulla, tämän muuttujan arvo on nolla. Jos käyttäjä on tarkennussivulla, muuttujan arvo on yksi. Näiden kahden muuttujan käyttäminen mahdollistaa käyttäjän valintojen tallentamisen.

Poiminta- ja tarkennusvaiheiden tarkkailun avulla käyttäjän tekemät valinnat voidaan tallentaa. Sovelluksen käyttämät suodattimien nimet on tallennettu PHP-taulukoihin, joista ne luetaan käyttäjän näkyviin. Poimintavaiheen seuraaminen on tärkeää jotta voidaan yhdistää sivu, jolla käyttäjä on sen taulukon kanssa, josta sivun arvot on luettu. Sovelluksessa jokaisen poiminta- tai tarkennusvaiheen valintalaatikot numeroidaan juoksevalla numerolla, mikä vastaa kyseisen valintalaatikon viereen tulostetun suodattimen nimen paikkaa taulukossa, josta nimi on lu-

ettu. Kun käyttäjä liikkuu vaiheiden välillä, lähetetään edellisessä vaiheessa valittujen valintalaatikkojen arvot POSTilla sovelluksen käyttöön samalla kun uusi vaihe ladataan. Nämä arvot tallennetaan niiden vastaaman vaiheen mukaiseen tauluun. Jos sovellus havaitsee, että poimintavaiheessa on tehty jo aiemmin valintoja, mutta käyttäjä on palannut vaiheeseen takaisin, verrataan nykyisiä valintoja aiemmin tallennettuihin. Jos arvot eivät täsmää, sovellus ylikirjoittaa vanhat valinnat uusilla. Valintojen seuraaminen mahdollistaa lisäksi ominaisuuden joka automaattisesti asettaa aiemmin valitut valintalaatikot valittu- tilaan. Tällöin käyttäjä näkee helposti mitkä suodattimet hän on aiemmin valinnut jos hän liikkuu takaisin edellisiin vaiheisiin. Tämä toiminnallisuus on toteutettu niin, että kun valintalaatikkoja luodaan sivulle, sovellus tarkistaa löytyykö valintalaatikon saamaa numeroarvoa vastaava arvo taulukosta johon aiemmat valinnat on tallennettu. Jos arvo löytyy, valintalaatikko asetetaan valittu- tilaan. Alhaalla olevassa kuvassa on esitelty ote valintalaatikon tilan määrittelevästä koodista

```
if(in_array($_SESSION['counter'], $_SESSION['poimintaValinnat1'])) {
    echo '<td><input type="checkbox" checked="true" name="checkbox[]" value='.$_SESSION['counter'].'></td>';
}
else {
    echo '<td><input type="checkbox" name="checkbox[]" value='.$_SESSION['counter'].'></td>';
}
```

**Kuvio 4.** Esimerkki valintalaatikoiden tilan määrittämisestä

Kun käyttäjä on kulkenut poimintavaiheiden läpi ja valinnut haluamansa suodattimet, päätyy hän lopulta koonti-sivulle. Tällä sivulla näytetään listamuotoinen koonti kaikista valituista suodattimista. Lisäksi tälle sivulle tuleminen muuttaa suorita laskenta-painikkeen käytössä-tilaan, joka mahdollistaa valittuja suodattimia vastaavien rivien laskemisen tietokannasta. Kun käyttäjä painaa tätä painiketta, sovellus käy läpi kaikissa poimintavaiheissa muodostuneet taulut, joihin käyttäjän valinnat on tallennettu, ja muodostaa näiden perusteella SQL-lauseen. Jos sovellus havaitsee, ettei käyttäjä ole tehnyt valintoja, tietokannasta lasketaan kaikki rivit. Kyselyä muodostettaessa on otettava myös huomioon pää- ja tarkennusvalintojen välinen suhde. Sovellus käsittelee päävalintoja niin, että jos käyttäjä

on tehnyt vain päävalinnan, mukaan otetaan kaikki kyseisen päävalinnan alle kuuluvat tarkennusvalinnat. Jos tarkennusvalintoja on tehty, sovellus käyttää vain valittuja kohtia. Alla olevassa kuvassa on esitelty ote koodista, joka luo SQL-lasueen käyttäjän tekemien valintojen mukaan.

```
if(!isset($_SESSION['poimintaSubValinnat2'])) {
    $_SESSION['stubQuery']=$_SESSION['stubQuery'].' SLKPTP IN';
    foreach ($GLOBALS['alueetMain'] as $key=>$value) {
        if(in_array($_SESSION['counter'], $_SESSION['poimintaValinnat2'])) {
            foreach($GLOBALS['alueetSub'.$_SESSION['counter']] as $_key=>$_value) {
                    if($_SESSION['startCounter']!=0) {
                        $_SESSION['stubQuery']=$_SESSION['stubQuery']. ', ';
                    }
                    if($_SESSION['startCounter']==0) {
                        $_SESSION['stubQuery']=$_SESSION['stubQuery']. ' (';
                        $_SESSION['startCounter']=1;
                    }
                    $_SESSION['stubQuery']=$_SESSION['stubQuery'].'"'.$_value.'"';


                $_SESSION['subCounter']++;
            }
        }
        $_SESSION['counter']++;
    }
}
```

**Kuvio 5.** Otos SQL-lauseen muodostamisesta

Kun laskenta on suoritettu, käyttäjälle näytetään lukumäärä riveistä, jotka vastasivat valittuja suodattimia. Tällöin myös lataa lista- painike muutetaan käytössä- tilaan ja sitä painamalla käyttäjä voi hakea rivit, jotka sovellus aiemmin laski, CSV-tiedostoon. Tietojen hakeminen tietokannasta toimii samaan tapaan kuin niiden laskeminenkin. Kyselyä ei kuitenkaan muodosteta uudelleen vaan sovellus käyttää samaa kyselyä tietojen hakemiseen kuin se käytti rivien laskemiseen. Erona tietenkin vain, että tällä kertaa valitaan tietoja, eikä vain lasketa rivejä. Kun tiedot on haettu, ne luetaan taulukosta CSV- yhteensopivaksi merkkijonoksi. Kun merkkijono on muodostettu, se lähetetään HTML-otsikkokenttien mukana selaimelle. Tällöin tiedosto latautuu käyttäjälle automaattisesti. Alla olevassa kuvassa on esitelty tiedoston lähettäminen käyttäjän selaimeen.

```
$filename = $_SESSION['poiminnanNimi'].date("Y-m-d_H-i",time());
header("Content-type: application/vnd.ms-excel");
header("Content-disposition: csv" . date("Y-m-d") . ".csv");
header("Content-disposition: filename=".$filename.".csv");
print $csv_output;
```

**Kuvio 6.** Tiedoston automaattinen lataus

Tiedoston latauksen jälkeen poiminnasta tallennetaan vielä yleisiä tietoja tietokantaan. Näitä tietoja käytetään hyväksi etusivulla, jossa käyttäjän aiemmat poiminnat näytetään. Tietojen avulla voidaan myös seurata käyttäjien sovelluksen kautta tekemiä latauksia.

### 4.3 Sovelluksen testaus

Sovellusta testattiin sen rakentamisen yhteydessä tasaisin väliajoin, jotta virhetilanteiden korjaaminen olisi helpompaa. Haastavimpia toteutettavia oli käyttäjän valintojen tallentaminen ja näiden valintojen käyttäminen sekä tietokantakyselei-den muodostaminen tehtyjen valintojen pohjalta. Tärkeimmät ominaisuudet saatiin kuitenkin toimimaan halutulla tavalla.

Puutteiksi sovelluksessa jäi poimintasivulla näytettävä koonti tietokannan sisällöstä, jossa ei tällä hetkellä näytetä lukumääriä. Tietokanta pitää sisällään tiedot vain toimipaikoista, joten tämä ominaisuus aktivoidaan kun tietokannasta saadaan käyttöön päivitetty versio. Lisäksi sovellus hyppää päättäjät- poimintavaiheen yli, tämä johtuu samasta syystä, eli tietokanta ei sisällä päättäjätietoja, joiden mukaan hakuja voisi suodataa.

Kun sovellus laitettiin tuotantoon, löytyi siitä lisäksi puute joka toistui vain Firefox-selaimella. Poimintasivun diville, joka pitää sisällään taulukon suodatinvalinnoista, ei oltu määritelty CSS:ssä float-arvoa. Tämä johti siihen, että div asettui keskelle sivua, muiden elementtien alle. Virhe saatiin kuitenkin nopeasti korjattua.

# 5 YHTEENVETO

Sovelluksen toteutus sujui suurimmaksi osaksi hyvin. Suurimpana ongelmana oli ajan puute muiden töiden johdosta ja välillä sovelluksen rakentamisesta täytyi pitää usean päivän taukoja. Kaikki vaatimukset saatiin kuitenkin lopulta täytettyä.

Sovelluksen toteutukseen annettiin LeadDeskin puolesta hyvin vapaat kädet, joten sain lähestyä toteutusta parhaaksi katsomallani tavalla ja pystyin ratkaisemaan vastaan tulleet ongelmat itsenäisesti. Sain myös lisätä sovellukseen muutamia tärkeiksi katsomiani ominaisuuksia. Erityisesti oppimisen kannalta pidin tärkeinä sitä, että sain itsenäisesti suunnitella toteutuksen ja ratkaista vastaan tulleet ongelmat.

Tässä työssä esitelty sovellusversio on sovelluksen ensimmäinen tuotantoversio ja sovellusta on tarkoitus kehittää tulevaisuudessa entistä pidemmälle. Pääpiirteittäin sovelluksen toiminta pysyy jatkossakin hyvin samanlaisena, mutta jatkokehitystä riittää silti vielä pitkäksi aikaa.

# LÄHTEET

/1/ Bootstrap CSS. Viitattu 24.11.2014. http://getbootstrap.com/css/

/2/ jQuery. Viitattu 24.11.2014. http://jquery.com/

/3/ LeadDesk yritysesittely. Viitattu 9.11.2014. http://www.leaddesk.com/fi/

/4/ Mozilla Developer Network. CSS Developer Guide. Viitattu 17.11.2014. https://developer.mozilla.org/en-US/docs/Web/Guide/CSS

/5/ Mozilla Developer Network. Introduction to HTML. Viitattu 17.11.2014. https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Introduction

/6/ Mozilla Developer Network. JavaScript Guide. Viitattu 17.11.2014. https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide

/7/ MySQL Reference Manuals. Viitattu 9.11.2014. http://dev.mysql.com/doc/#manual

/8/ PHP Manual. Viitattu 9.11.2014. http://php.net/manual/en/