YRITYSTEN TIETO- JA KONTAKTIHAKU

HAMK
HÄMEEN AMMATTIKORKEAKOULU

Ammattikorkeakoulun opinnäytetyö

Tietotekniikan koulutusohjelma

Riihimäki, 16.10.2013

Tapani Kuronen

HAMK
HÄMEEN AMMATTIKORKEAKOULU

TIIVISTELMÄ

Riihimäki
Tietotekniikan koulutusohjelma

Tekijä Tapani Kuronen **Vuosi** 2013

Työn nimi Yritysten tieto- ja kontaktihaku

TIIVISTELMÄ

Opinnäytetyön toimeksiantajana toimi Zoined Oy. Työn tarkoituksena oli tutkia ja toteuttaa yritysten tieto- ja kontaktihaku käyttäen mahdollisuuksien mukaan ilmaisia tai halpoja ratkaisuja. Tietohaun tarkoituksena oli etsiä haettavan yrityksen perustiedot ja mahdollisuuksien mukaan myös taloustiedot. Kontaktihaun tarkoituksena oli puolestaan saada yrityksiin liittyvistä henkilöistä kuva ja perustiedot. Toimeksianto oli osa isompaa projektia, jonka tarkoituksena oli luoda eräänlainen monitoimihakukone.

Opinnäytetyön teoriaosuudessa käydään läpi yritystieto- ja kontaktihaun toteutukseen liittyvät vaihtoehdot sekä vertaillaan eri vaihtoehtoja keskenään. Teoriaosuudessa käydään läpi myös ohjelman toteutuksen kannalta mahdollisia ohjelmointikieliä ja -tekniikoita sekä vertaillaan niitä. Lisäksi luvussa esitellään opinnäytetyön toteutusosan ohjelmointivaiheessa käytetyt ohjelmointityökalut. Opinnäytetyön teorian tutkimuksessa käytettiin internetmateriaalia sekä kirjallisuuslähteitä.

Kontaktihaku toteutettiin käyttäen Linkedin yritysverkoston ohjelmistorajapintoja. Yritystietohaku toteutettiin puolestaan käyttäen OpenCorporates:in, YTJ:n ja Eniron palveluita ja ohjelmistorajapintoja. Yritystieto- ja kontaktihaku on toteutettu PHP-ohjelmointikielellä siten, että osaa PHP-koodimoduuleista on mahdollista käyttää mahdollisimman hyvin myös muihin käyttötarkoituksiin tulevaisuudessa.

Opinnäytetyön suunnittelu ja toteutusosa onnistui hyvin ottaen huomioon ilmaisten palveluiden käyttämisen. Tavoitteet saavutettiin lukuun ottamatta yritystietohaun taloustietoja, joita ei olisi ollut mahdollista saada ilman merkittäviä kustannuksia. Hakutulosten kattavuudeksi Suomen osalta saatiin noin 70 prosenttia ja Ruotsin ja Norjan osalta hieman enemmän. Opinnäytetyön pohjalta on mahdollista jatkokehittää melko helposti laajemman alueen käsittävä palvelu.

Avainsanat ohjelmistorajapinta, kontaktihaku, yritystietohaku

Sivut 49 s. + liitteet 19 s.

HAMK
UNIVERSITY OF APPLIED SCIENCES

ABSTRACT

Riihimäki
Degree Programme in Information Technology

Author Tapani Kuronen **Year** 2013

Subject of Bachelor's thesis Company information and contact search

ABSTRACT

This thesis was commissioned by Zoined Oy. The subject of the thesis was to research and implement a company information and contact search with low or no cost. The purpose of the company information search was to find basic company information together with basic financial information when possible. The contact search's purpose was to find basic information and pictures on employees from other companies. This thesis was part of a bigger project to create a multi-purpose search engine.

The theoretical section of the thesis includes implementation options for the company and contact search modules and compares these options to each other. Possible programming languages and programming techniques including their comparison are introduced in the theoretical section. Programming tools used in the implementation section of the thesis are also included in the theoretical section. Both literary and internet materials are used as sources for the theoretical.

The contact search was created using Linkedin's application programming interfaces. The company information search was implemented using OpenCorporates', YTJ's and Eniro's services or application programming interfaces. The implementation of the company information and the contact search was created using the PHP programming language. PHP files were made to work as independent as possible to be easy to use as part of later projects.

The planning and implementation sections of this Bachelor's thesis were successful considering the use of free services. The goals of the project were achieved with the exception of the financial information which was not possible to get without payment. The coverage of the search results in Finland is about 70 percent; coverage in Sweden and Norway is slightly larger. It is possible to continue development of larger service based on this thesis.

Keywords application programming interface, contact search, company information search

Pages 49 p. + appendices 19 p.

SANASTO

Automaattinen roskienkeruu (tietotekniikassa)	Garbage collection	Automaattinen roskienkeruu on ohjelmoinnissa käytetty muistinhallintamekanismi, joka tuhoaa tarpeettomat toiminnallisuudet ja tiedot vapauttaen muistia muuhun käyttöön.
ASCII	American Standard Code for Information Interchange (ASCII)	AsCII on 7-bittinen tietotekniikassa käytössä oleva merkistö, joka sisältää erityisesti amerikanenglannissa tarvittavat merkit ja ohjauskoodit.
Formaatti (tietotekniikassa)	Format	Formaatilla tarkoitetaan julkista määritelmää, jossa kuvataan, miten tietoa tulee koodata ja käsitellä.
Funktio (ohjelmoinnissa)	Function	Funktio on aliohjelma, joka suorittaa tietyn edeltä määritellyn asian ja on paketoitu funktion muotoon. Ohjelmakirjastot koostuvat usein funktioista.
Hyperteksti	Hypertext	Hyperteksti on tietokonelaitteella esitetty teksti, jossa käyttäjä voi siirtyä dokumentin eri osiin tai kokonaan eri dokumentteja valitsemalla hyperlinkin eli viittauksen dokumenttien välillä.
ID	Identifier	ID on tietojenkäsittelyssä käytettävä tunniste, jonka avulla esimerkiksi tietokannasta voidaan tunnistaa haluttu tieto.

Luokkakirjasto	Class library	Luokkakirjasto on kokoelma valmiiksi kirjoitettuja luokkia tai luokkakaavoja (class template), joita toiset ohjelmoijat voivat hyödyntää ohjelmoinnissa.
Merkintäkieli (kuvauskieli)	Markup language	Merkintäkieli (kuvauskieli) on kieli, jolla kuvataan tekstin esitystapaa tai rakennetta metainformaatiolla. Merkintäkielellä pyritään yleensä erottamaan tekstin looginen rakenne sisällöstä.
Ohjelmistomoduuli	Software module	Ohjelmistomoduulit ovat itsenäisiä ohjelmisto-osia, joista voidaan koota toimivia kokonaisuuksia.
Ohjelmointirajapinta, ohjelmistorajapinta	Application programming interface (API)	Ohjelmointirajapinta on rajapinta, jota käyttäen ohjelmat voivat tehdä pyyntöjä ja vaihtaa tietoa keskenään.
Ohjelmointi	Programming	Ohjelmoinnilla tarkoitetaan esim. tietokoneelle annettavia toimintaohjeita, joita käyttäen tietokone tekee halutut asiat.
Olio	Object	Olio on olio-ohjelmoinnin perusyksikkö, joka voi sisältää tietoa ja toiminnallisuutta. Oliot mahdollistavat helpommin toimivien kokonaisuuksien toteuttamisen.
Palvelin	Server	Palvelin on tietokone, jonka tehtävänä on tarjota toimintoja ja resursseja ohjelmille sekä paikallisesti että internetin yli.

Parametri (Tietotekniikassa)	Parameter	Parametri on erityinen muuttuja, joka määrittää ohjelman alitoiminnallisuutta.
Protokolla	Protocol	Protokolla on säännöstö, joka määrittelee esim. HTML:n käyttämän POST-lähetteen toiminnallisuuden.
Skripti	Script	Komentosarjoilla eli skripteillä tarkoitetaan lyhyttä koodiosaa, jolla toteutetaan haluttu toiminnallisuus.
Syntaksi (ohjelmointikielissä)	Syntax	Syntaksi kuvaa ennalta sovittua rakennetta, jolla muun muassa sanat ja lauseet esitetään siten, että esim. ohjelmointikielet voivat tunnistaa ne.
Tavukoodi	Bytecode	Tavukoodi on koodi, joka käännetään ohjelmointikielen lähdekoodista esim. koodin ajamista varten.
Tietokanta	Database	Tietokanta on tietotekniikassa käytetty termi tietovarastolle. Tietokannassa tieto on yhteydessä toisiinsa ja sitä käytetään hyödyksi sovellusten tekemisessä.
URL	Uniform resource locator (URL)	URL on merkkijono, joka sisältää viittauksen resurssiin. (http://microsoft.com/finland)

SISÄLLYS

1 JOHDANTO

Opinnäytetyöni on osa isompaa Zoined Oy:n projektia. Projektin tarkoituksena on luoda yrityshakukone, jonka kautta on mahdollista saada mahdollisimman paljon tietoa haettavasta yrityksestä yhdellä hakusanalla. Oma osuuteni projektista koostuu kahdesta ohjelmistomoduulista, jotka ovat yritystietohaku ja yrityskontaktihaku. Yritystietohaun tavoitteena on saada mahdollisimman hyvät perustiedot haettavasta yrityksestä. Kontaktihaun tavoitteena on puolestaan tuoda käyttäjän nähtäväksi haettavaan yritykseen liittyvien henkilöiden kuvat ja perustiedot. Näitä hakutoimintoja ja muita projektiin liittyviä hakutoimintoja käyttäen on tarkoitus muodostaa käyttäjälle suppea kuvaus yrityksestä ja osasta sen työntekijöitä. Tätä palvelua voidaan muun muassa hyödyntää tutkittaessa mahdollisia yhteistyökumppaneita.

Opinnäytetyön luvussa kaksi esitellään tutkitut tekniikat, toteutuksen kannalta mahdolliset ohjelmointikielet ja toteutuksessa käytettävät työkalut. Luvussa vertaillaan myös tekniikoiden ja ohjelmointikielien keskinäisiä heikkouksia ja vahvuuksia ottaen huomioon opinnäytetyön tavoitteet.

Työn kolmannessa luvussa tutkitaan kontaktihaun toteuttamisen mahdollisuuksia ja esitellään opinnäytetyön kannalta käytettävät ohjelmistorajapinnat. Luvussa pohditaan myös kontaktihakupalvelulle valittujen ohjelmistorajapintojen käyttömahdollisuuksia.

Opinnäytetyön neljännessä luvussa tutkitaan yritystietohaun toteuttamiselle vaihtoehtoisia palveluita ja selvitetään mitä vaihtoehtoisia tutkittuja palveluita voidaan käyttää mahdollisimman kattavien yritysten perustietojen hankkimiseen. Luvussa esitetään myös eri vaihtoehtojen vertailuja ottaen huomioon, miten ne toimisivat esimerkiksi Suomessa ja muualla maailmalla.

Ohjelman toteutusosa on dokumentoitu luvussa viisi. Toteutusosa alkaa ohjelman määrityksellä, minkä jälkeen esitellään toteutettavan ohjelman suunnitelma. Tämän jälkeen käydään yksitellen läpi ohjelman eri osien toteuttamiset ja arvioidaan toteutuksen onnistuminen. Toteutusosaan kuuluu lisäksi testausvaihe, jossa kerrotaan ohjelman testauksesta toteutettavan ohjelman tekemisen aikana.

Opinnäytetyöni viimeisessä luvussa käydään läpi omia ajatuksiani opinnäytetyöstä ja opinnäytetyön onnistumisesta ottaen huomioon alkuperäiset lähtökohdat. Luvussa käsitellään myös opinnäytetyön aikaista oppimisprosessia muun muassa käytettyjen tekniikkojen ja ohjelmointikielten osalta.

2 OHJELMOINTI JA TEKNIIKAT

Ohjelmointi ja tekniikat -osiossa esitellään opinnäytetyön kannalta mahdollisia tekniikoita ja ohjelmointikieliä sekä vertaillaan niiden ominaisuuksia. Osiossa myös esitellään toteutuksen tekemisessä käytettävät ohjelmistokehitystyökalut.

2.1 Tekniikat

Luvussa esitellään opinnäytetyön kannalta mahdollisia tekniikoita, joita voidaan hyödyntää ohjelmoinnin puolella.

2.1.1 HTML

HTML (Hypertext Markup Language) on hypertekstinen merkintäkieli. HTML-merkintäkielen kehitys alkoi vuonna 1989 ja sen ensimmäinen versio HTML 1.0 ilmestyi vuonna 1991. HTML:n alkuperäisenä kehittäjänä toimi Tim Berners-Lee. HTML 1.0 rakenne perustui isolta osin SGML:n syntaksiin. HTML:n uusin versio on HTML5, jota ei ole tosin vielä virallisesti standardoitu. HTML:n standardointia hoitaa tällä hetkellä W3C eli World Wide Web Consortium. (Wikipedia 2013a.)

HTML-merkintäkielen asema maailmalla on nykyään hallitseva eikä sillä ole tällä hetkellä varteenotettavia kilpailijoita. Muun muassa Flash-tekniikan asema verkkosivujen tekemisessä on minimaalinen. HTML:n tarkoitus on pääasiassa määrittää sivun rakenne ja sisältö eikä esimerkiksi sivun tyylejä ja toiminnallisuuksia. Tyylit ja toiminnallisuudet hoidetaan esimerkiksi skriptien ja CSS-tyylikielen avulla.

HTML kertoo WEB-selaimelle kuinka sen tulee näyttää sivun sisältöä. Se muun muassa erottelee esimerkiksi tekstin, kuvat, äänet ja videot. Tähän erottelemiseen HTML käyttää ennalta määritettyjen elementtien joukkoa, joiden avulla se määrittää sisällön tyypit. HTML-elementit rakentuvat aloitus- ja lopetusmerkinnöistä sekä niiden välissä olevasta sisällöstä. Tästä esimerkkinä <HTML>-aloitusmerkintä, joka päätetään </HTML>-lopetusmerkinnällä. Kuvassa 1 on esitelty minimaalinen oikeaoppinen HTML5-sivu. (Sheppy 2013.)

```
<!DOCTYPE html>
<html>
<head>
<title>Minimaalinen</title>
</head>

<body>
Sivun sisältö tähän
</body>

</html>
```

Kuva 1. Kuvassa esitetään minimaalinen HTML-sivu.

Kuvasta 1 nähdään myös että HTML-sivun rakenne toimii hyvin hierarkkisesti. Minimaalinen HTML-sivu tarvitsee aina vähintään <!DOCTYPE>-, html-, head-, title- ja body-elementit. Sivu toimii tosin selaimissa pienemmilläkin määrityksillä, kuten ilman head-elementtiä. Näissä tapauksissa kyse on HTML:n virheellisestä käytöstä, joka ei ole suositeltavaa. Pakollisten elementtien puuttuminen voi aiheuttaa esimerkiksi sen, että sivu lakkaa toimimasta osittain tai kokonaan.

2.1.2 JSON

JSON eli JavaScript Object Notation on suorituskyvyltään tehokas tekstidata formaatti, jonka kehitys alkoi vuonna 2001. Tiedettävästi ensimmäinen iso toimia, joka otti JSON:in käyttöön laajamittaisesti, oli Yahoo vuonna 2005. JSON pohjautuu JavaScript ohjelmointikieleen, mutta toimii nykyään lähes kaikkien ohjelmointikielien kanssa. Tosin esimerkiksi PHP-ohjelmointikielen kanssa käytettäessä vaaditaan JSON-koodin purkaminen ohjelmointikielen käyttämään muotoon siihen tehdyn ohjelmistotyökalukirjaston avulla. (Wikipedia 2013b.)

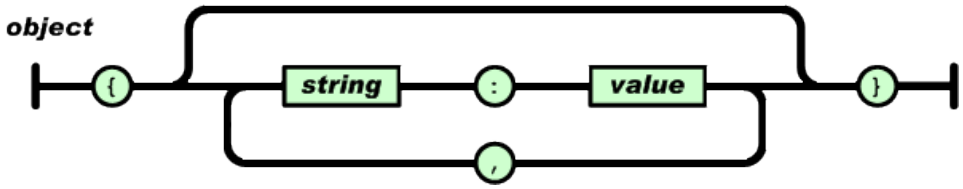
JSON-formaatin etuina ovat rakenteen minimaalinen yksinkertaisuus, jonka ansiosta formaattia on helppo ymmärtää, sekä useimpia muita vastaavia formaatteja suurempi koodinajotehokkuus. Kuvassa 2 esitetään yksinkertainen esimerkki JSON-muotoisesta testistä.

```
{"people": {
  "_total": 1,
  "values": [
    {"apiStandardProfileRequest": {
      "headers": {
        "_total": 1,
        "values": [{
          "name": "x-li-auth-token",
          "value": "OUT_OF_NETWORK:2sXb"
        }]
      },
      "url": "http://api.linkedin.com/v1/people/vazvW029-x"
    }}
  ]
}}
```

Kuva 2. Kuvassa esitetään yksinkertainen JSON-esimerkki.

Rakenteellisesti JSON:in toimintaa voi verrata käyttäjätunnus salasanapareihin, joissa käyttäjätunnus on tietoa kuvaava merkintä kuten ”_total”, jonka jälkeen tulee tieto-osa kaksoispisteellä erotettuna edellisen kuvan mukaisesti. Kaksoispisteellä erotettu tieto-osuus voi sisältää myös uusia kuvaus-tieto-pareja, mikä mahdollistaa JSON:in hierarkkisen rakenteen. Kuvausosa voi myös sisältää pitkän listan tieto-osia kuten edellisen kuvan ”values”- kohdassa tapahtuu. JSON toimii luettelomaisesti, jossa samalla hieakisella tasolla olevat osiot erotetaan toisistaan pilkun välityksellä edellisen kuvan mukaisesti. Tosin yksinkertaisimmillaan esimerkiksi pelkkä luku ”100” voi jo yksinään olla JSON:a, mutta näin yksinkertainen

JSON ei ole yleisesti käytettyä. Kuvassa 3 esitetään oliotyyppinen JSON-rakenne. (JSON.org.)



Kuva 3. Kuvassa on oliotyyppinen JSON-rakenne.

2.1.3 XML

XML on nykyään yleisesti käytetty kuvauskieli, jonka kehitys alkoi vuonna 1996. XML:n kehitykseen johti tarve helpompaan tekstityyppisen tiedon käsittelyyn internetin käyttäjämäärän alkaessa kasvaa nopealla vauhdilla 1990-luvun puolivälin vaiheilla. XML:n on kehittänyt World Wide Web Consortium (W3C). Sen ensimmäinen versio julkaistiin virallisesti vuonna 1998 ja nykyinen versio eli XML:n 1.0-versio (viides versio) vuonna 2008. XML on alun perin SGML:stä tehty yksinkertaistettu versio, jossa tiedon jäsentelystä ja käytettävyydestä on tehty helpompaa. (History of XML 2013.)

Xml on yksinkertainen ja erittäin joustava kuvauskieli, jota käytetään yleensä tiedon siirtämisessä internetin halki. Se mahdollistaa esimerkiksi tekstitiedostojen helpon siirtämisen internetin läpi palvelimelta asiakas-käyttäjälle. Tästä syystä sitä käytetään tavallisten verkkosivujen sisällön näyttämisen lisäksi myös monien ohjelmistorajapintojen palautussyötteen kanssa. (XML 2012.)

XML on standardisoitu formaatti, jota käytetään muun muassa tiedonsiirrossa järjestelmien välillä ja dokumenttien tallentamisessa. XML:ssä tieto on ympäröity sen tarkoitusta kuvaavilla merkinnöillä (eng. tag) kuvan 4 mukaisesti.

```
▼<people-search>
  ▼<people total="1">
    ▼<person>
      ▼<api-standard-profile-request>
          <url>http://api.linkedin.com/v1/people/vazvW029-x</url>
        ▼<headers total="1">
          ▼<http-header>
              <name>x-li-auth-token</name>
              <value>OUT_OF_NETWORK:2sXb</value>
            </http-header>
          </headers>
        </api-standard-profile-request>
      </person>
    </people>
  </people-search>
```

Kuva 4. Kuvassa esitellään yksinkertainen XML-esimerkki.

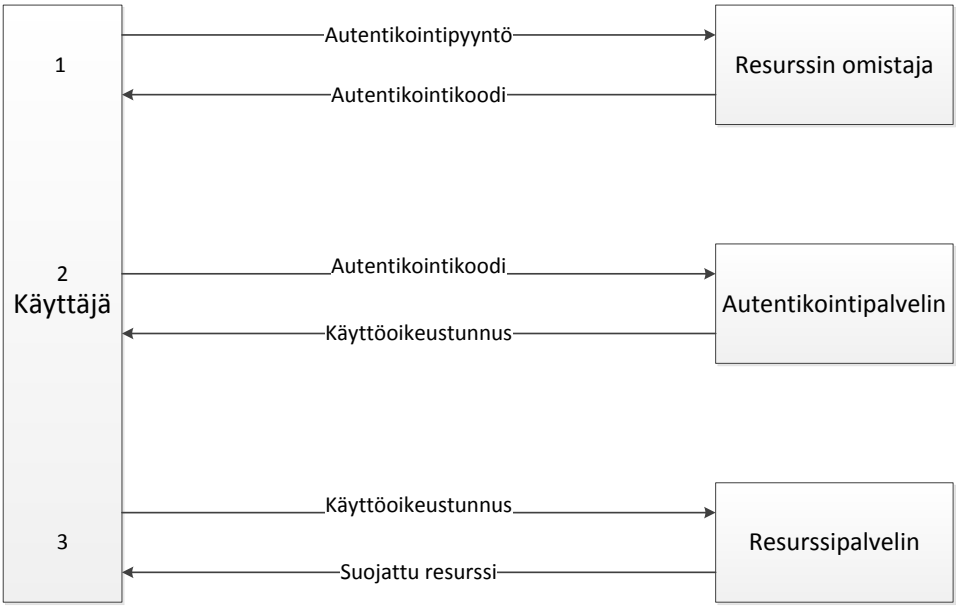
XML:n rakenteessa aloitettu merkintä esimerkiksi ”<people-search>” aloittaa hierarkkisen osion, joka loppuu aina merkintää vastaavaan lopetusmerkintään kuten tässä tapauksessa ”</people-search>”. Nämä merkintöjen sisältämät osiot voivat sisältää uusia merkinnöillä eli ”tageilla” erotettuja osioita tai suoraa varastoitua tietoa kuten esimerkiksi esimerkin ”<name>” - merkinnän sisältämä tieto.

Tekniikkana XML on nykyään yleisesti tuettu ja käytetty useimmissa järjestelmissä. Ohjelmistokehityksen kannalta XML on käytännössä riippumaton käytetystä ohjelmointikielestä, koska kaikissa nykyään käytetyissä ohjelmointikielissä kuten Java:ssa, php:ssa tai C#:ssa on joko sisäänrakennettu tuki tai ulkoinen kirjasto sen käyttämistä varten.

2.1.4 OAuth

OAuth on avoin ja laajasti käytetty protokolla, joka tarjoaa salatun käyttäjän tunnistamisen eli autentikoinnin yksinkertaisella ja standardoidulla menetelmällä työpöytä-, verkko- ja mobiilisovelluksia ajatellen. OAuth:n ensimmäinen versio julkaistiin vuoden 2007 lopussa eli noin vuoden projektin aloittamisen jälkeen. Nykyinen versio OAuth 2.0 julkaistiin lokakuussa 2012. (Hammer-Lahav 2007.)

OAuth 2.0 toiminta on jakautunut kolmeen osaan kuvan 5 mukaisesti.



Kuva 5. Kuvassa OAuth 2.0 protokollan toiminta pääpiirteittäin.

Ensimmäisessä osassa käyttäjä pyytää resurssin omistajalta autentikointikoodia. Autentikonti koodia pyydettäessä käyttäjä lähettää resurssin omistajalle esimerkiksi käyttäjätunnuksen ja palautusosoitteen. Tämän jälkeen käyttäjälle aukeaa yleensä ikkuna, jossa kysytään käyttäjätunnusta ja salasanaa. Tämän ikkunan tarkoituksena on käyttäjän tunnistautuminen. Lo-

puksi autentikointikoodi palautetaan käyttäjälle. Autentikointi voidaan toteuttaa myös muilla tavoilla ja toteutusmuoto on pitkälti resurssin omistajan päätettävissä. Toisessa vaiheessa pyydetään käyttöoikeustunnusta (Access Token) autentikointipalvelimelta käyttäen esimerkiksi autentikointikoodia, käyttäjätunnusta ja salasanaa. Tarvittavien parametrien määrä määräytyy kyseisen autentikoitipalvelimen tarpeiden mukaan. Kolmannessa vaiheessa voidaan käyttöoikeustunnusta käyttäen pyytää suojattua dataa resurssipalvelimelta. Autentikoinnin kaksi ensimmäistä vaihetta voidaan välttää tallentamalla käyttöoikeustunnus muistiin myöhempää käyttöä varten. (Hardt 2012.)

2.1.5 URL–koodaus

URL voidaan lähettää internetin yli ainoastaan ASCII-merkistön sallimassa muodossa. Tästä johtuen merkit, jotka eivät kuulu ASCII määritykseen, täytyy muokata sen mukaisiksi. Tämä tehdään URL-koodauksen avulla.
Kaikille merkeille on olemassa oma ”%”-merkillä alkava koodinsa, joka sisältää aina myös kaksi heksadesimaalilukua. URL-osoite ei voi esimerkiksi sisältää välilyöntejä, joten välilyönnit korvataan yleensä joko ”%20”- tai ”+”-merkeillä. (URL Encoding.)

Kuvassa 6 on esitettynä joitakin yleisiä URL-muunnoksia.

Space	%20	(	%28
&	%26	)	%29
/	%2F	*	%2A
ö	%F6	+	%2B
ä	%E4	,	%2C
%	%25	-	%2D
?	%3F	.	%2E

Kuva 6. Kuvassa esitetään merkkien URL – koodeja

URL–koodauksessa on huomioitava, että esimerkiksi tavallisia ASCII-merkistön mukaisia pieniä ja isoja kirjaimia ei yleensä URL-koodata, vaikka näillekin on olemassa omat koodinsa. Esimerkiksi iso A-kirjain olisi mahdollista koodata muotoon ”%41”. Tavallisia numeroita ei myöskään yleensä URL–koodata. Myös merkit ”.”, ”–”, ”~” ja ”_” jätetään koodaamatta. Kuvassa 7 on esimerkki URL-syötteen ja URL-koodatun version eroista. (Wikipedia 2013c.)

```
https://localhost/testapi?search=Yön ärsyttävät ötökät

https%3A%2F%2Flocalhost%2Ftestapi%3Fsearch%3DY%C3%B6n
+%C3%A4rsytt%C3%A4v%C3%A4t+%C3%B6t%C3%B6k%C3%A4t
```

Kuva 7. Kuvassa on esiteltynä alkuperäinen URL–syöte ja sen URL-koodattu muoto

Kuvan 7 perusteella voidaan todeta, että URL-koodatut URL-osoitteet voivat olla hyvinkin pitkiä, koska yksi koodaamaton merkki vastaa aina kolmea URL-koodattua merkkiä.

URL-rivin vastaanottajalla tapahtuu vastaanottotilanteen yhteydessä URL-koodin takaisin koodaaminen, mikä johtaa siihen, että käyttäjä ei huomaa koko URL-koodaus prosessia.

2.2 Ohjelmointikielet

Luvussa esitellään opinnäytetyön toteutuksen kannalta mahdolliset ohjelmointikielet ja vertaillaan niitä toisiinsa.

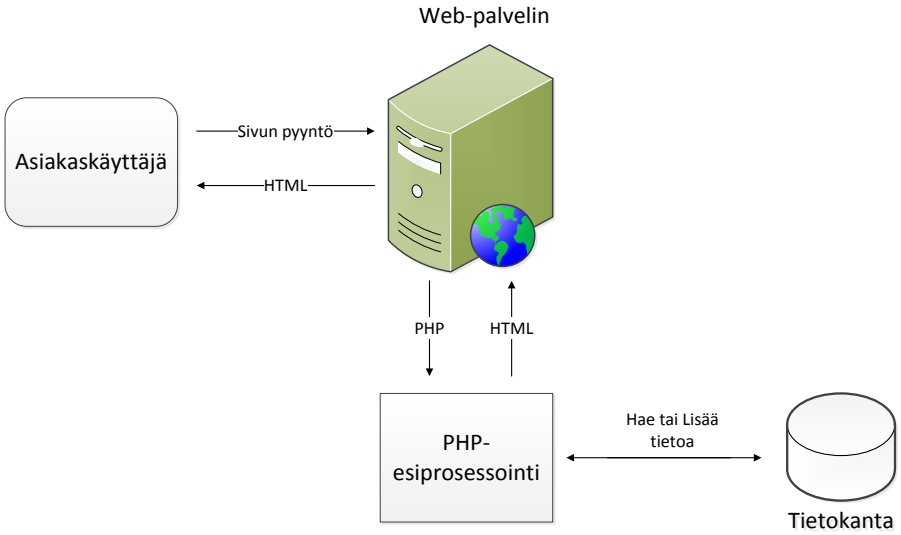
2.2.1 PHP

PHP on Rasmus Lerdorfin alun perin vuonna 1994 luoma ohjelmointikieli. Vuoden 1994 versiossa keskeistä oli, että muuttujat muistuttivat hyvin paljon Pearl-ohjelmointikieltä ja se sisälsi HTML sulautetun syntaksin. Kokonaan uudestaan kirjoitettu versio PHP:sta julkaistiin kuitenkin jo syksyllä 1995 korjaamaan ensimmäisen version suuria epäkohtia. Tätä vuoden 1995 versiota pidetään nykyisen PHP-kielen ensimmäisenä versiona. Tässä versiossa ohjelmointikielen syntaksi oli tehty C-ohjelmointikielen kaltaisesti helpottamaan siirtymää ja kynnystä sen käyttämiseen. PHP:n viimeisin versio 5.4.15 julkaistiin huhtikuussa 2013. (php.net 2013.)

PHP on yksinkertainen ja tehokas kieli HTML sisällön luomiseen. PHP-ohjelmointikieltä käyttävien sivustojen osuus internetissä on tällä hetkellä noin 80 %. PHP-ohjelmontikieli sisältää monia olio-ohjelmointikielen ominaisuuksia, mutta siitä puuttuu kokonaan säikeiden käytön mahdollisuus. Tämä estää sillä tehtyjen ohjelmien eri osien samanaikaisen suorittamisen, joka pakottaa suunnittelemaan ohjelman siten, että sen toiminnallisuudet voidaan ajaa jonossa.

PHP-ohjelmointikieltä voidaan käyttää kolmella eri tavalla. Näitä käyttötapoja ovat palvelinpuolen ohjelmointi, komentoriviohjelmointi ja asiakaspuolen työpöytäsovellukset. PHP-ohjelmointikielellä tehtävistä komentorivisovelluksia kutsutaan yleisesti skripteiksi. (Tatroe, MacIntyre & Lerdorf 2013, 1.)

PHP-ohjelmointikieli kehitettiin alun perin dynaamisten verkkosivujen tekemistä varten. Tämä alkuperäinen toiminnallisuus on vielä nykyäänkin sen tärkein ja sille parhaiten sopiva käyttötarkoitus. PHP-ohjelmointikielen ajaminen palvelinpuolella vaatii palvelimen ja PHP tulkin. Kuvassa 8 esitetään esimerkki PHP:n toiminnallisuudesta. (Tatroe, MacIntyre & Lerdorf 2013, 1.)



Kuva 8. Kuvassa PHP verkkosovelluksen toiminta malli

PHP-ohjelmointikielellä luodussa verkkosovelluksessa asiakaskäyttäjä pyytää palvelimelta haluamaansa verkkosivua, minkä jälkeen palvelin lähettää kyseiseen verkkosivuun liittyvän PHP-koodin PHP-tulkin käsiteltäväksi. PHP-ohjelmakoodi on koodin suorittamisen aikana mahdollisesti yhteydessä palvelimella oleviin tietokantoihin ja tiedostoihin, joita se käyttää asiakaskäyttäjälle lähetettävän HTML-sivun tekemiseen. Lopuksi palvelin palauttaa prosessoidun HTML-sivun asiakaskäyttäjälle.

Kuvassa 9 esitetään yksinkertainen esimerkki PHP-ohjelmointikielellä toimivasta verkkosovelluksesta, joka tulostaa käyttäjälle sanat ”HELLO WORLD”.

```php
<?php

class testiLuokka
{
    function hello()
    {
        echo  "HELLO WORLD";
    }
}

$ajo = new testiluokka();
$ajo->hello();

?>
```

Kuva 9. Kuvassa PHP-verkkosovellus

Toinen käyttötarkoitus PHP:lle on tekstipohjaiset komentoriviltä ajettavat ohjelmat. Näitä käytetään pääasiassa järjestelmänhallintatyökalujen tekemisessä. Esimerkiksi jotkut varmuuskopiointi- ja lokiohjelmat on tehty tällä tekniikalla. (Tatroe, MacIntyre & Lerdorf 2013, 1.)

Kolmas mahdollinen käyttötarkoitus on PHP-ohjelmointikielellä tehty visuaalinen työpöytäsovellus. Tässä hyödynnetään PHP:n GTK-tekniikkaa, jolla saadaan aikaan visuaalinen käyttöjärjestelmä riippumaton työpöytäsovellus. Kuvassa 10 esitetään yksinkertaisen visuaalisen työpöytä sovelluksen tekemiseen käytetty koodi. (Tatroe, MacIntyre & Lerdorf 2013, 1.)

```
<?php
$wnd = new GtkWindow();
$wnd->set_title('Hello world');
$wnd->connect_simple('destroy', array('gtk', 'main_quit'));

$lblHello = new GtkLabel("HELLo WORLD");
$wnd->add($lblHello);

$wnd->show_all();
Gtk::main();
?>
```

Kuva 10. Kuvassa on ”HELLO WORLD” PHP-GTK-sovelluksena.

PHP-ohjelmointikieli on turvallinen oikein käytettynä. Yleensä tietoturvauhan aiheuttaja on huolimaton ohjelmoija, joka ei esimerkiksi varmista käyttäjän antamaa syötettä ollenkaan tai tarpeeksi hyvin ennen syötteen prosessointia. PHP:n tunnetuin haavoittuvuus on SQL-injektiohyökkäys. Siinä hyökkääjä onnistuu ajamaan omia SQL-komentojaan, esimerkiksi huonosti tarkistetun lomakkeen kentän kautta, päästen käsiksi tietokannan tietoihin, joihin kyseisellä käyttäjälle ei pitäisi olla pääsyä. Muita PHP:n uhkia, jotka tulisi ottaa huomioon PHP-ohjelmoinnissa, ovat XSS-, CSRF- ja RFI-hyökkäykset. (Powers 2010, 6.)

2.2.2 C#

C# on Microsoftin yleiskäyttöön suunnattu olio-ohjelmointikieli, joka luotiin yhtiön .NET konseptia varten. Ohjelmointikielen ensimmäinen versio julkaistiin vuonna 2000. C#:n kehityksen tavoitteena on ollut erilaisiin ympäristöihin soveltuvan ja helppokäyttöinen olio-ohjelmointikielen luominen. Kehitystä on alusta alkaen johtanut Anders Hejlsberg, joka on tunnettu ja kokenut ohjelmointikielien kehittäjä. Nykyinen versio 5.0 on julkaistu vuonna 2012. (Wikipedia 2013d.)

C# on olio-ohjelmointikieli, joka sisältää automaattisen roskienkeruun (Garbage Collector), joka poistaa automaattiset vanhat käyttämättömät ohjelmankappaleet vapauttaen muistia. Tämä järjestely antaa anteeksi muun muassa koodaajan ohjelman muistinkäytönkannalta tekemiä virheitä ja

unohduksia, jotka aiheuttavat muistivuodoksi kutsuttua ilmiötä. Muistivuodossa ohjelma käyttää koko ajan enemmän muistia luodessaan esimerkiksi samaa asiaan käytettävää oliota jatkuvasti uudelleen niin että edelliset vanhat oliot jäävät käyttämättömiksi ohjelmaan. Osana .NET kieliä C# pystyy keskustelemaan muiden .NET kielten kanssa. Tietyin ehdoin on mahdollista esimerkiksi periyttää kokonaisia koodikirjastoja toisista .NET kielistä. C#-ohjelmointikielesä on myös mahdollista käyttää säikeistystä, joka mahdollistaa useampien koodiosien ajamisen yhtä aikaa. Sitä käyttäen voidaan esimerkiksi asettaa jatkuva kuuntelija kuuntelemaan jotakin tietokoneen liitäntää siten, että kuuntelija ei vaikuta muun koodin suorittamiseen. (Solis 2012, 5-6.)

C#-ohjelmointikieltä käytetään pääasiassa sovelluksien tekemiseen Windows-käyttöjärjestelmälle. Vaikka Microsoft on standardisoinut kielen ECMA:n kautta, sen ajamiseen tarkoitetut välineet muilla yleisillä käyttöliittymillä ovat hyvin rajalliset. Tämä vaikeuttaa C#-kielen kilpailua muiden olio-ohjelmointikielien kuten Javan kanssa, minkä hyviä puolia ovat juuri helppo ajettavuus kaikissa suuremmanluokan käyttöjärjestelmissä. C#-ohjelmointikielellä tehtyä koodia on kuitenkin mahdollista ajaa melkein kaikissa käyttöjärjestelmissä. Esimerkiksi palvelinpuolella ajettava C#:n avulla luotu ASP.NET-verkkosovellus näkyy asiakaskäyttäjälle pelkästään HTML-hypertekstinä, jota voi ajaa kaikilla selaimilla. (Albahari & Albahari 2012, 3.)

Toinen keino C#-koodin ajamiseen Windowsissa on Microsoft Silverlight sovelluksien tekeminen. Silverlight on Adoben Flash alustaa vastaava ajoympäristö, joka on tuettuna Windowsissa ja Mac OS X:ssä.

Kolmas keino C#:lla tehdyn koodin ajamiseen on Microsoftin ulkopuolisen ajorajapinnan käyttäminen. Tästä käytetyin esimerkki on Mono project. Mono toimii Windowsin lisäksi myös esimerkiksi Mac OS X-, Linuxi- ja Solaris-käyttöjärjestelmissä. Mono:n avulla on mahdollista käyttää suurinta osaa C#-ohjelmointikielen ominaisuuksista lukuun ottamatta esimerkiksi WPF-sovelluksia, joita ei tällä hetkellä tueta. (Albahari & Albahari 2012, 3.)

Kuvassa 11 on esitettynä yksinkertainen konsolisovellus. Kuvan esimerkissä tulostetaan ”Hello World” teksti käyttäen hyväksi luokkaoliosta saatua merkkijonoa x.

```
namespace ConsoleApplication
{
    class Program
    {
        static void Main(string[] args)
        {
            testi olio = new testi();

            Console.WriteLine(olio.x);
        }
    }
    class testi
    {
        public string x = "Hello World";
    }
}
```

Kuva 11. Kuvassa esitetään C#-konsolisovellus.

C#-ohjelmointikielen syntaksi on hyvin samankaltainen muiden C-kielien ja esimerkiksi Javan kanssa, koska C#-ohjelmointikielen syntaksi perustuu pitkälti C- ja C++-ohjelmointikielten syntaksiin. Myös esimerkiksi PHP-ohjelmointikieli muistuttaa syntaksiltaan hyvinkin paljon C#-ohjelmointikieltä, mikä puolestaan helpottaa samankaltaisen syntaksin omaavien ohjelmointikielten opettelemista ja hallitsemista.

2.2.3 Java

Java on Sun Microsystemsin luoma ohjelmointikieli, joka kehitettiin vuoden 1990-luvun alussa. Sen ensimmäinen kehitysalusta JDK 1.0 julkaistiin syksyllä 1995 ja uusin versio elokuussa 2012. Java-ohjelmointikielen kehitykseen ovat vaikuttaneet muun muassa C++, Eiffel ja Objective-C. (Wikipedia 2013e.)

Ohjelmoinnin näkökulmasta Java on puhdasverinen olio-ohjelmointikieli, jossa käytetään alusta alkaen olioita ja luokkia. Ohjelmoinnissa voidaan käyttää hyväksi Javassa valmiina olevia luokkakirjastoja. Yksi Javan tärkeimmistä ominaisuuksista on sen käyttöjärjestelmäriippumattomuus. Tämän takia Java-ohjelmat käännetään lähellä konekieltä olevaksi tavukoodiksi. Tavukoodin ajamiseen tarvitaan tulkki (virtuaalikone), jonka käyttö hidastaa ohjelman ajamista selvästi verrattuna esimerkiksi C++-ohjelmointikieleen. Tästä johtuen jokaiselle käyttöjärjestelmälle tehdään oma optimoitu virtuaalikone, jossa käyttöjärjestelmäriippumaton tavukoodi tulkataan. Javassa on vahva tyypitys verrattuna esimerkiksi C- ja C++-ohjelmointikieliin, joissa tietotyyppejä ei ole määritetty yhtä tarkasti. (Vesterholm & Kyppö 2006, 23-24.)

Javassa on C#-ohjelmointikielen tapaan automaattinen roskienkeruu, jonka ansiosta esimerkiksi olioiden varaama muisti vapautetaan automaattisesti kun olio esimerkiksi poistetaan. Java käyttää Unicode-merkistöä, joka mahdollistaa useimpien kielten merkkien näyttämisen mukaan lukien Suomen ä- ja ö-merkit. Poikkeustilanteiden käsittely on myös yksi Javan

vahvoista alueista. Java-ympäristö käytännössä pakottaa ohjelmoijan ottamaan kantaa mahdollisten virhetilojen käsittelyyn. Java-ohjelmointikieli käsittää myös säikeiden käytön, joka mahdollistaa moniajon eli useiden ohjelman ominaisuuksien suorittamisen yhtäaikaisesti.
(Vesterholm & Kyppö 2006, 23-24.)

Javalla voidaan luoda sekä työpöytä että verkkosovelluksia. Verkkosovellusten puolella Javaa käytetään etenkin Java Applettien eli käyttäjien selaimessa ajettavien Java-ohjelmien tekemiseen. Tässä tapauksessa Java-ohjelma on sisällytetty verkkosivuun. Javaa käytetään myös jonkin verran serveripuolen toiminnallisuuksissa, mutta se ei ole saanut serveripuolen ohjelmissa isompaa jalansijaa. Toisin kuin Appletit Javan työpöytäsovellukset toimivat itsenäisesti. Javalla voidaan työpöytä puolella tehdä melkein millaisia tahansa sovelluksia. Java-sovelluksia on yleisesti käytössä muun muassa hieman edistyneemmissä kahvinkeittimissä, mikroissa ja televisioissa. (Boese 2010, 6.)

Kuvassa 12 on esitettynä Esimerkki Java-koodista. Kun verrataan kuvan 12 ja kuvan 11 rakennetta voidaan huomata, että samaa asiaa tekevät koodit ovat samanlaiset ottamatta huomioon joitakin ohjelman ajoympäristön ominaisuuksia kuten tulostuskäskystä.

```
public class HelloWorld {

        public static void main(String[] args) {

                testi olio = new testi();

                System.out.println(olio.x);
        }
}

class testi {
        String x = "Hello World";
}
```

Kuva 12. Kuvassa Java-konsolisovellus

2.2.4 JavaScript

Netscapen Brendan Eich kehitti JavaScriptin vuonna 1995 Netscape selainta varten. Sen ensimmäinen versio julkaistiin vuonna 1996 nimellä LiveScript. Viimeisin vakaa versio on 1.8.5, joka julkaistiin joulukuussa 2009. JavaScript on nykyään yleisesti käytetty ohjelmointikieli, jota käytetään yleensä sivustojen toiminnallisuuksien tekemisessä. Vaikka JavaScript on olio-ohjelmointikieli, niin sitä kutsutaan siitä huolimatta alkeelli-

seksi skriptikieleksi. JavaScriptin syntaksi pohjautuu C ja Java ohjelmointikieliin, mikä tekee siitä helposti ymmärrettävää. (Willison 2006.)

JavaScriptiä käytetään pääasiassa clientin eli asiakasohjelman puolella tuomaan verkkosivuille tarvittavia interaktiivisia ominaisuuksia. JavaScriptillä toteutetaan muun muassa web-sivujen lomakkeiden, nappien ym. palikoiden toiminta. JavaScriptiä käyttäen voidaan siis luoda lennossa muuttuvia sivuja. JavaScript ei kuitenkaan ole pelkästään asiakaskäyttäjäpuolen ohjelmointikieli vaan sillä on myös mahdollista tehdä palvelinpuolen toiminnallisuutta. Sen käyttö palvelinpuolella ei kuitenkaan ole yleistä. JavaScriptiä on käytetty hyödyksi esimerkiksi Adobe Acrobat- ja Photoshop-ohjelmien tekemisessä. (Negrino & Smith 2007, 5.)

JavaScriptillä on kuitenkin omat rajoituksensa, jotka estävät sitä tekemästä muutoksia käyttäjän tietojärjestelmään. Nämä JavaScriptiin rajoitukset ovat olemassa pelkästään tietoturvasyistä. JavaScriptiä käyttäen ei voida esimerkiksi lukea käyttäjän henkilökohtaisia tiedostoja eikä myöskään tallentaa käyttäjän koneelle dataa. Tästä poikkeuksena selaimien välimuisti, johon on mahdollista tehdä tietojen tallennuksia rajoitetuin ehdoin. JavaScript ei voi myöskään sulkea sellaisia selaimen ikkunoita, joita se ei ole itse avannut. Myös niin sanottuja ”Cross domain” eli verkkosivustojen välisiä tietoja ei voida yleensä lukea turvallisuussyistä käyttäen JavasCript:ä. (Negrino & Smith 2007,6.)

Kuvassa 13 on esitelty yksinkertainen JavaScript-toiminnallisuus osana HTML-sivua.

```
<!DOCTYPE html>
<html>
<head>
<script>
function tulostus()
{
alert("Toimiiko?");
}
</script>
</head>

<body>
<button onclick="tulostus()">Paina</button>
</body>
</html>
```

Kuva 13. Kuvassa yksinkertainen esimerkki JavaScriptin käytöstä.

JavaScript erotetaan HTML:stä <script>-merkinnän ja sen </script>-lopetusmerkinnän avulla. Kuvassa 13 vihreällä merkitty osa on varsinaista JavaScript koodia, joka ajetaan HTML:n <button> eli nappielementistä onclick-nimisen funktion avulla. Kuvan esimerkin mukainen funktio tulostaa siis nappia painamalla ”Toimiiko?”-tekstin.

2.2.5 Ohjelmointikielten vertailu ja johtopäätökset

Ohjelmointikielten vertailussa on käsitelty toteutuksen kannalta mahdollisia palvelinpuolenohjelmointikieliä. Toteutuksen valintaan vaikuttaa muun muassa ohjelmointikielten käyttömahdollisuudet Linux-käyttöjärjestelmässä, jossa opinnäytetyön tilaajan palveluita ajetaan. Myös asiakkaan halukkuus ja oma ohjelmointikielen osaaminen vaikuttavat päätökseen. Taulukossa 1. esitetään pienimuotoinen vertailu PHP-, Java- ja C#-ohjelmointikielistä.

Taulukko 1. Taulukossa vertaillaan palvelinpuolen ohjelmointikieliä.

	PHP	Java	C#
Suorituskyky			
Koodausnopeus			
Työkalut			
Koodikirjastoja			
Säikeistys	Ei	Kyllä	Kyllä
Olio-ohjelmointi	Kyllä	Kyllä	Kyllä
Alustariippumattomuus			
Oma kielen tuntemus			
Asiakkaan halukkuus			
WEB-ohjelmointi			

Taulukossa tumma väri kuvaa positiivista vaihtoehtoa.

Kaikki taulukossa 1. mainitut ohjelmointikielet kelpaavat ohjelman toteuttamiseen. Oman ohjelmointikielten osaamiseni perusteella olisin toteuttanut sovelluksen C#-ohjelmointikielellä. Linux-alustalle olisi C#-kielen takia täytynyt asentaa Mono-ympäritö, joka mahdollistaisi C#-ohjelmointikielen suorittamisen Linux-ympäristössä. Mono-ympäristöä ei kuitenkaan lopulta päätetty asentaa pelkästään yhden projektin takia vaan pitkälti asiakkaan pyynnöstä siirryttiin käyttämään PHP-ohjelmointikieltä opinnäytetyön toteutusosan tekemisessä.

2.3 Sovelluskehityksessä käytettävät työkalut

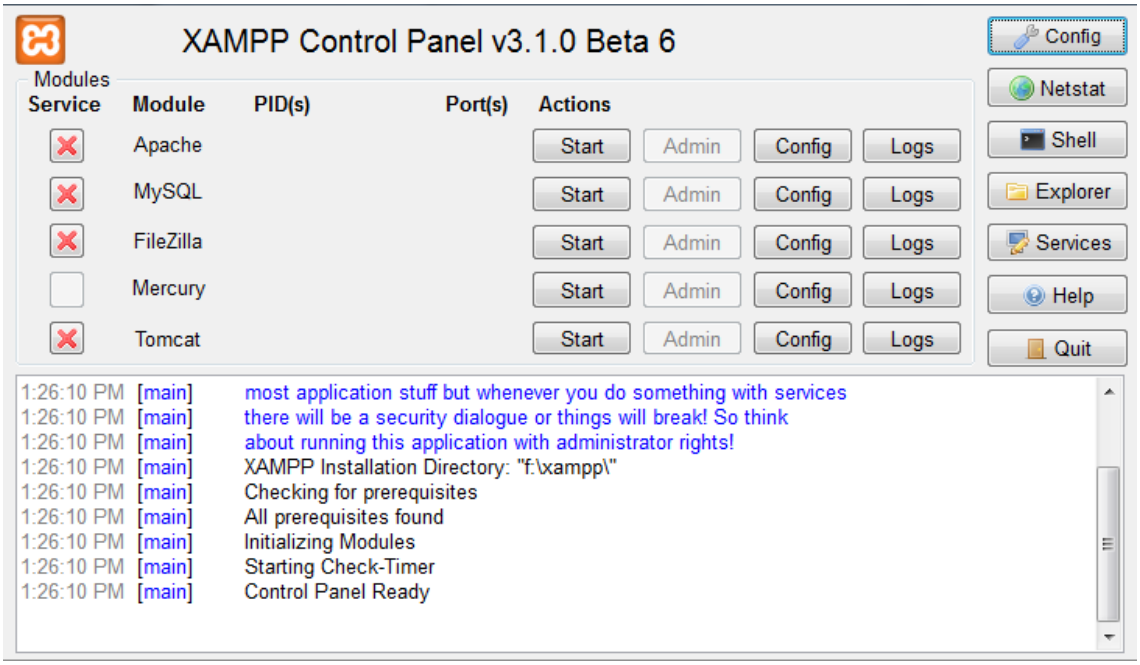
Luvussa kerrotaan toteutuksen yhteydessä käytetyistä sovelluskehitystyökaluista, joiden avulla toteutusosan sovellus kehitettiin.

2.3.1 XAMPP

XAMPP on ilmainen avoimen lähdekoodin verkkopalvelin, joka on saatavana Windowsille, Linuxille, Mac OS X:lle ja Solarikselle. XAMPP-palvelin on tarkoitettu työkaluksi sovelluskehittäjille helpottamaan kehitettävien ohjelmien ajamista paikallisesti. Tästä huolimatta XAMPP-palvelinohjelmaa käytetään nykyään jonkin verran myös esimerkiksi verkkosivujen ylläpitämispuolella. (Seidler 2013.)

XAMPP-palvelin tukee muun muassa Apache-, MySQL-, PHP-, PEAR-, PERL-, OpenSSL-, FileZilla FTP Server -, Mercury Mail - ja phpMyAdmin-nimisiä palvelinohjelmia, ohjelmointikieliä ja palveluja. Osa ominaisuuksista on käyttöjärjestelmäkohtaisia. (Seidler 2013.)

XAMPP on helppo asentaa ja siitä on saatavilla muun muassa muistitikuilla toimiva mukana pidettävä versio. Käytännössä asentamiseen riittää paketin purkaminen, jonka jälkeen XAMPP-palvelin on käyttövalmis. Eli mitään hankalaa asennusprosessia ei tarvitse tehdä, mikä puolestaan helpottaa XAMPP-palvelimen käyttöönottoa. Kuvassa 14 on esitettynä yksinkertainen XAMPP-ohjauspaneeli. (Seidler 2013.)



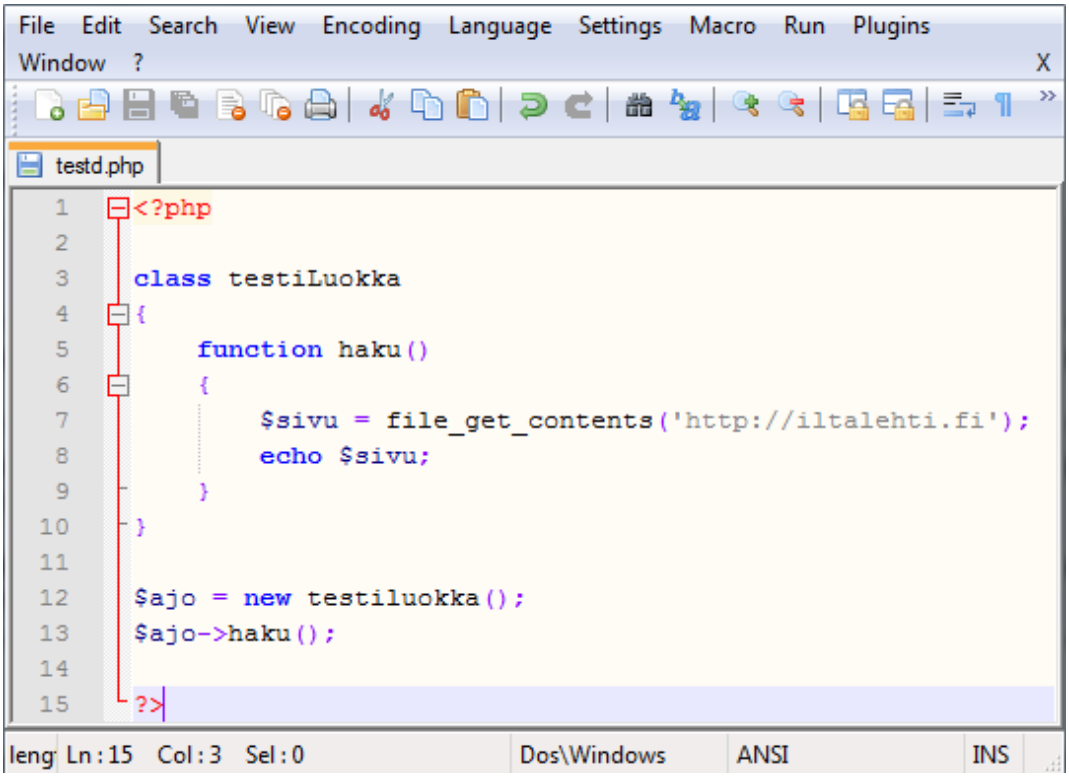
Kuva 14. Kuvassa on esitettynä XAMP-palvelimen ohjauspaneeli.

2.3.2 Notepad++

Notepad++ on suorituskyvyltään kevyt avoimenlähdekoodin teksti- ja lähdekoodieditori Windows käyttöjärjestelmälle. Sen julkaistiin alun perin vuonna 2003 ja viimeisin versio on vuoden 2013 toukokuulta. Ohjelma on kirjoitettu C++-ohjelmointikielellä ja se perustuu isolta osin Scintillaan, joka on ilmainen lähdekoodin editointikomponentti. (Wikipedia 2013f.)

Notepad++ on helppokäyttöinen tekstin ja koodin tarkoitettu ohjelma, joka sisältää yleisten tekstieditorien ominaisuuksien lisäksi muun muassa koodin värikoodauksen ja automaattisen jäsentelyn. Ohjelman ominaisuudet sisältävät myös automaattisen sanan ja parametrin täydennyksen sekä vihjelistat esimerkiksi olemassa olevista ohjelman tuntemista koodiparametreista. Liitteessä 1. on listattu ohjelman tukemat ohjelmointikielet ja tekniikat. (Ho 2011.)

Notepad++ ei sisällä koodinajamiseen tarvittavia välineitä. Sen ”Run”-ominaisuuden avulla voi tosin valita käyttöjärjestelmästä koodin ajamiseen kykenevän ohjelman. Tällöin esimerkiksi HTML-hypertekstin ja JavaScriptin ajamista varten voidaan valita haluttu selain, joka avautuu valinnan jälkeen editorin koodin kanssa. Kuvassa 15 on esiteltynä Notepad++-ohjelman käyttöliittymä.



Kuva 15. Kuvassa on Notepad++ editori, jossa käsitellään PHP-koodia.

3 KONTAKTIHAKU

Kontaktihaussa tavoitteena on luoda ohjelmistomoduuli, joka antaa käyttäjälle yrityksen nimen perusteella yrityksen työntekijöiden kuvat ja perustiedot. Linkedin on opinnäytetyön tilaajan määrittelemä ehdoton vaihtoehto kontaktihakupalvelu moduulin toteuttamiselle, joten moduuli toteutetaan sitä käyttäen. Teoreettisesti ajateltuna muita vaihtoehtoja olisivat myös Google+, Facebook ja Twitter, mutta mikään näistä ei pikaisen tutkimisen perusteella tarjoa tarpeeksi laajaa ja laadukasta kontaktitiehenkilötietoa maailman yrityksistä.

3.1 Linkedin yleisesti

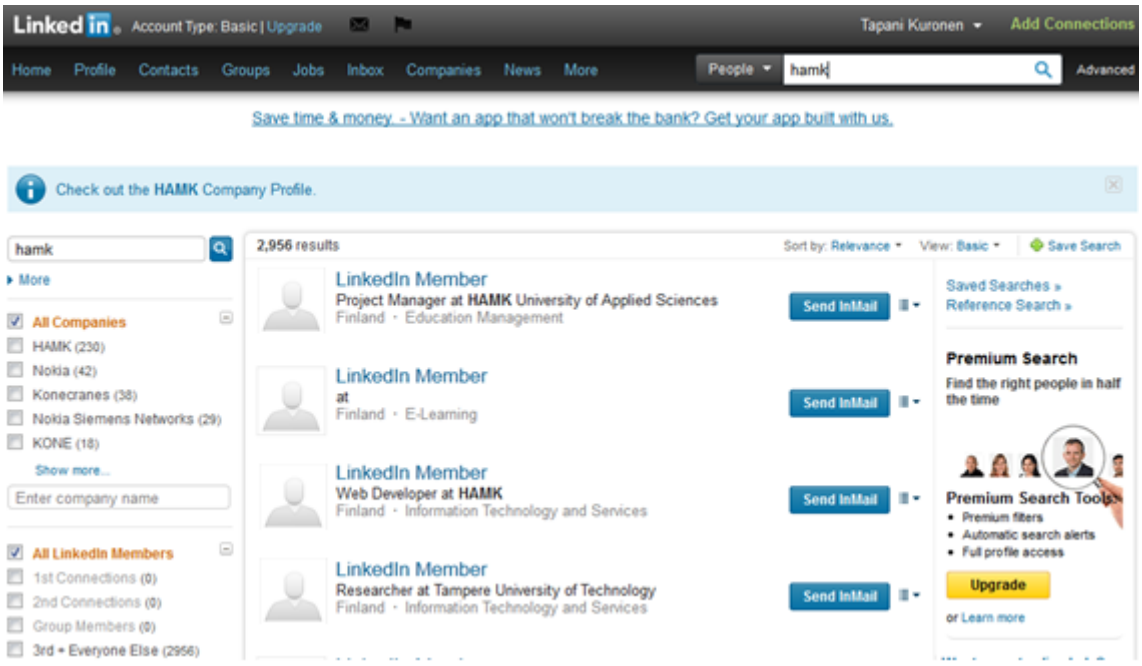
Linkedin on yritysten ja yksittäisten ihmisten käyttämä kansainvälinen verkkoyhteisöpalvelu, joka on perustettu vuonna 2003. Linkedin palvelua ylläpitävä Linkedin Corporation on yhdysvaltalainen pörssiyhtiö, joka oli ensimmäinen merkittävä sosiaalisen median verkkopalvelu New Yorkin pörssissä. Sillä on tällä hetkellä maailmanlaajuisesti yli 200 miljoonaa käyttäjää. Linkedinin liikevaihto koostuu pääasiassa Linkedinin sivuilla olevien mainosten, maksullisten käyttäjätilien maksuista sekä rekrytointipalveluista. (Wikipedia 2013g.) (Wikipedia 2013h.)

Yritysten kannalta Linkedin tarjoaa mahdollisuuden päästä esille ja saada sitä kautta tavallaan ilmaista mainosta. Yritykset ja heidän työntekijänsä voivat myös luoda Linkedinin kautta yhteyksiä muihin yrityksiin ja muihin käyttäjiin. Tätä kautta yritykset voivat hyötyä palvelusta esimerkiksi markkinoinnin ja asiakkaiden osalta.

Yksityiset käyttäjät voivat käyttää Linkediniä yhteyksien luomiseen muihin henkilökäyttäjiin ja yrityksiin. Käyttäjät voivat laittaa esille myös kiinnostuksen kohteensa ja esimerkiksi ansioluettelonsa. Tätä kautta Linkedinistä voi olla hyötyä myös työnhaussa, vaikka palvelua ei alun perin ole tähän käyttötarkoitukseen suunniteltu.

Linkedinin perustili on ilmainen ja hyvä vaihtoehto peruskäyttäjälle. Perustilissä Linkedin-hakujen tekemistä on rajoitettu tulosmäärien ja niiden tarjoaman sisällön mukaan. Myös suorat Linkedin sisäiset viestin omien yhteyksien ulkopuolelle on estetty. Maksulliset ”Business”-tilit mahdollistavat kattavamman hakemisen, viestinnän, teknisen tuen ym. hienot ja hyödylliset ominaisuudet, joita yrityskäytössä tarvitaan. (Linkedin a.)

Linkedin tarjoaa sovelluskehittäjille useita ohjelmistorajapintoja esimerkiksi henkilöiden, yritysten ja yhteyksien hakemiseen. Näitä ohjelmistorajapintoja käyttäen sovelluskehittäjät voivat toteuttaa omia sovelluksiaan esimerkiksi yritystensä verkkosivuille. Sovelluskehittäjät saavat luoda viisi käyttäjätiliä. Näistä ohjelmistorajapinnoista esitellään People-Search API luvussa 3.2 ja Profile API luvussa 3.3. Kuvassa 16 esitellään Linkedinin henkilöhaku. (Linkedin b.)



Kuva 16. Linkedinin henkilöhaku

3.2 People-Search API

People-Search API on Linkedinissä olevien käyttäjien hakemista varten tehty ohjelmistorajapinta. Se on tarkoitettu käytettäväksi kaikilla muilla ohjelmointikielillä paitsi JavaScriptillä, jolle on Linkedin on tehnyt omat ohjelmistorajapintansa.

Linkedinin People-Search-ohjelmistorajapinta käyttää käyttäjien tunnistamiseen OAuth 2.0 -protokollaa. OAuth 2.0 vaatii, että käyttäjän täytyy hyväksyä kehitetyn palvelun käyttö kirjautumalla sisään henkilökohtaisella Linkedinin käyttäjätunnuksella ja salasanalla, jotka saadaan hankittua ”http://developer.linkedin.com”-sivuston kautta. (Linkedin c.)

Autentikoinnin eli käyttäjän tunnistamisen jälkeen tehdään haku käyttäen hyväksi laajaa valikoimaa hakutermejä. Seuraavassa esimerkissä tehdään yksinkertainen haku käyttäen hakuparametrina avainsanoja (keywords):

```
api.linkedin.com/v1/people-
search?keywords=Nokia&oauth2_access_token=AVAIN
```

Kyseinen haku toimii omiin ensimmäisen ja toisen asteen kontakteihin, mutta ei lainkaan oman Linkedin verkoston ulkopuolisiin ihmisiin. Avainsanoja voi olla useampia. Käyttöoikeustunnus (oauth2_access_token=AVAIN) määritellään viimeisenä kyselyn lopussa erotettuna aina kysymys – tai & -merkillä. (Linkedin d.)

Haettaessa esimerkiksi avainsanoilla ihmisiä oman Linkedin-verkon ulkopuolelta täytyy käyttää lisäparametreja seuraavan esimerkin mukaisesti.

https://api.linkedin.com/v1/people-search:(people:(api-standard-profile-request))?first-name=&last-name=&keywords=Nokia

Api-standard-profile-request on parametri, joka kertoo, että haku tehdään oman Linkedin-verkon ulkopuolisista henkilöistä. Tällöin Linkedin voi seurata, mitä tai kuinka monta annettujen hakutulosten linkeistä käytetään esimerkiksi profiilihaun yhteydessä. Api-standard-profile-request palauttaa siis yksilöidyn id:n haun kohteille. Myös etunimi ja sukunimi täytyy määrittää tehtäessä oman Linkedin verkon ulkopuolisia hakuja. Sekä etunimen että sukunimen voi tosin jättää tyhjäksi, jolloin haku toteutuu avainsanan tai vastaavan hakuparametrin mukaisesti. (Linkedin d.)

Tarkemmat haussa käytettävät parametrit löytyvät osoitteesta http://developer.linkedin.com/documents/profile-fields.

Hakutulokset saadaan oletuksena XML-muodossa. Hakutulokset on myös mahdollista saada JSON-muodossa lisäämällä formaattiparametrin (format=json) seuraavan esimerkin mukaisesti.

api.linkedin.com/v1/people-search?keywords=Nokia&format=json
&oauth2_access_token=AVAIN

People-Search API vaatii sovellukselle r_network-luokan käyttöoikeudet, jotka käyttäjä hyväksyy Linkedin-tunnuksillaan. Päivittäinen hakumäärä on maksimissaan 100 000 hakua. Sovellus voi pyytää päivittäin 500 autentikointikoodia ja 500 000 käyttöoikeustunnusta. Tosin esimerkiksi autentikointikoodin voi tallentaa muistiin myöhempää käyttöä varten, joten niitä ei tarvitse pyytää koko ajan uudestaan.

Linkedinin käyttöehtojen mukaan hakutuloksia ei saa tallentaa tai liittää yhdistettyyn hakuun. Kaikki haut on myös tehtävä henkilökohtaisilla tunnuksilla aktiivisen käyttösession aikana eli omia tunnuksia ei saa käyttää muiden tekemiin hakuihin vaan jokaisen sovelluksen käyttäjän täytyy autentikoitua Linkedinille henkilökohtaisesti. (Linkedin e.)

Kun ohjelmistorajapintaa käytetään esimerkiksi ihmisten palkkaamiseen, myyntiin, markkinointiin tai ihmisten profiilien järjestelmälliseen tarkasteluun, täytyy liittyä Linkedinin Partner Programs -ohjelmaan, joka antaa lisää oikeuksia ohjelmistorajapinnan käyttöä varten. Kuvassa 17 tarkemmin Partner Program -ohjelmaan liittymisen vaatimukset. (Linkedin e.)

If your application falls into one or more of the following categories, you are required to be part of one of our Partner Programs and have a signed agreement with LinkedIn that allows you to use our APIs in your application:

- Applications used for hiring, marketing, or sales. For example, employee hiring, company marketing or monitoring, lead generation, relationship management, systematic matching of people with their LinkedIn profile, or anything with a similar feature or purpose. Developers interested in these categories often apply to one of the following Partner Programs: Social Media Management, Ads, ATS, Job Postings, CRM, or our Certified Developer Program.
- Applications that currently have or expect to have more than 250,000 lifetime members using our APIs, make more than 500,000 daily API calls, make more than 500,000 lifetime people search API calls, or serve greater than 1 million daily plugin impressions.
- Applications that store or export any data from LinkedIn other than the Profile Data as explicitly allowed in Section III.B. This includes data for anyone other than the member, such as a member's connections, people found via search, and information about companies and job postings.

Kuva 17. Ote Linkedinin käyttöehdoista

3.3 Profile API

Profile API on tarkoitettu Linkedin henkilöprofiilitietojen hakemiseen. Sovellusrajapinta soveltuu melko lailla kaikkien ohjelmointikielien käytettäväksi. Poikkeuksena JavaScript-ohjelmointikieli, jolle on olemassa oma sovellusrajapintansa. Linkedinin Profile API ohjelmistorajapinta käyttää käyttäjien autentikointiin OAuth 2.0 protokollaa People-Search API -ohjelmistorajapinnan mukaisesti.

Profile Api:ssa haut tehdään käyttämällä URL-osoiteriviä seuraavan esimerkin mukaisesti.

http://api.linkedin.com/v1/people/id=abcdefg&oauth2_access_token=AVAIN

Edellinen yksinkertainen haku esimerkki muodostuu kolmesta osasta. Ensimmäisessä vihreällä merkityssä osassa määritellään käytettävä ohjelmistorajapinta, joka esimerkin tapauksessa on ”people”-versio 1. Punaisella merkitty osa sisältää id-parametrin, jossa määritellään haettavan profiilin id-tunnus. Kolmas sinisellä merkattu osa käsittää käyttöoikeustunnuksen. (Linkedin f.)

Profile API -ohjelmistorajapintaa voi käyttää myös useamman käyttäjän profiilitietojen hakemiseen seuraavan esimerkin mukaisesti.

https://api.linkedin.com/v1/people::(HENKILÖ1,HENKILÖ2,):(first-name,last-name,headline,picture-url,location:(name))?format=json&oauth2_access_token AVAIN

Profiilista voi myös pyytää vain tietyt halutut tiedot kuten edellisessä esimerkissä on punaisella merkattuna. Haun tulokset on mahdollista saada joko XML - tai JSON formaateilla (”format=json”).

Profile API:n käyttäminen vaatii toimiakseen sovellukselle joko r_basicprofile - tai r_fullprofile –luokan oikeudet. r_basicprofile antaa oikeudet pyytää perustietoja kuten etunimi, sukunimi, asemayrityksessä ja profiilikuva. Parametri r_fullprofile palauttaa puolestaan lisätietoja kuten profiilin omistajan iän, kiinnostuksen kohteet ja kyvyt. (Linkedin f.)

Profile API:n käyttöoikeudet ovat käytännössä identtiset People-Search API:n kanssa.

3.4 Johtopäätökset

Kontaktihaun toteuttaminen on tehtävä käyttäen Linkedinin ohjelmistorajapintoja, koska muita vastaavia yrityskontaktihakupalveluita ei Linkedinin ohella ole olemassa. Myös opinnäytetyön asiakkaan selvä halukkuus tutkia Linkedinin ohjelmistorajapintoja vaikutti Linkedinin ehdottomaan valintaan opinnäytetyön sovelluksen toteutusosaan. Toteutuksessa tarvitaan molempia esitellyistä Linkedinin ohjelmistorajapinnoista siten, että People-Sears API -ohjelmistorajapinnan kautta saatuja tuloksia käytetään Profile API -ohjelmistorajapinnassa hakutermeinä, jonka avulla saadaan kontaktitiedot henkilöistä.

4 YRITYSHAKUPALVELUT

Luku käsittelee olemassa olevia yritystietohaun toteuttamiseen soveltuvia ohjelmistorajapintoja ja tietolähteitä. Tavoitteena on löytää tietolähteitä, joiden avulla saisi kohtuullisin käyttöehdoin ja ilmaiseksi tai halvalla käytettäväksi yritysten perustietoja ja mahdollisuuksien mukaan myös taloustietoja.

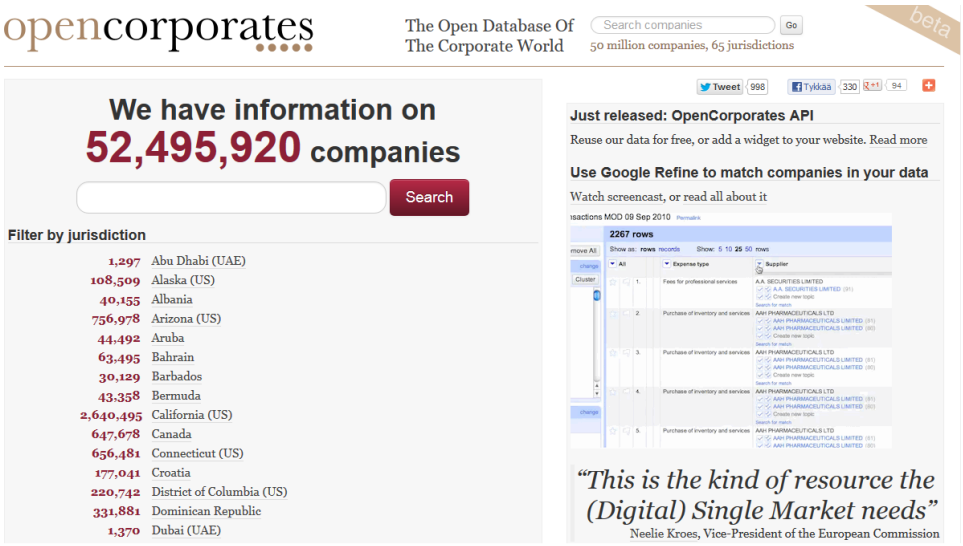
4.1 OpenCorporates

4.1.1 Yleisesti

OpenCorporates on verkkopalvelu, jonka englantilainen Chrinon Ltd perusti 20 joulukuuta 2010. OpenCorporates:in päätavoitteena on luoda URL-osoite kaikille maailman yrityksille ja tarjota keräämänsä tieto kaikille halukkaille ilmaiseksi. (Wikipedia 2013i.)

OpenCorporates hankkii yritystietoja julkisista lähteistä sitä mukaan kun sitä on tarjolla ilmaiseksi. Esimerkiksi Tanskan yritystiedot on päätetty julkistaa ja OpenCorporates on valmis tarjoamaan ne käyttäjille välittömästi, kun ne on saatu tietokantaan. Yksittäiset käyttäjät voivat ilmoittaa esimerkiksi omaan tai tuntemaansa yrityksiin liittyvää lisätietoa kuten postiosoitteita ja yrityksen verkkosivujen osoitteita palveluun. Yksittäiset asiasta kiinnostuneet OpenCorporates:in käyttäjät voivat myös etsiä uusia yritystiedon lähteitä palvelun käyttöön auttaen palvelun ylläpitoa laajentamaan yritys- ja maavalikoimaa. (OpenCorporates a.)

OpenCorporates:in tuntemien yritysten ja niiden hallinnollisten alueiden määrä on kasvanut nopeasti viimeisten vuosien aikana. Tällä hetkellä OpenCorporates tuntee noin 55,788,763yritystä yli 65 hallinnollisella alueella. Kuvassa 18 on esitetty OpenCorporates-palvelun etusivu. (OpenCorporates b.)



Kuva 18. OpenCorporates:n etusivu vuodelta 2013.

Sovelluskehitystä varten OpenCorporates tarjoaa OpenCorporates REST API:n ja Google Refine Reconciliation API:n, joista ensimmäistä käsitellään luku 4.1.2.

4.1.2 OpenCorporates REST API

OpenCorporates REST API on yksinkertainen ohjelmistorajapinta tiedon hakemiseen OpenCorporates:in tietokannasta. OpenCorporates REST API -ohjelmistorajapinta on tällä hetkellä beeta tilassa. Nykyinen versio on 0.2.

OpenCorporates tarjoaa yritystiedot avoimen tietokantalisenssin (Open Database Licence) ehdoilla. OpenCorporates kuitenkin edellyttää, että hakutulosten yhteyteen laitetaan linkki ”from OpenCorporates”, joka johtaa esimerkiksi OpenCorporates:in etusivulle. (OpenCorporates c.)

Sovellusrajapinta palauttaa hakutulokset oletuksena JSON-formaatissa, mutta tarvittaessa tulokset on myös mahdollista saada XML-formaatissa käyttäen lisäparametriä ”format=xml” seuraavan esimerkin mukaisesti. Esimerkissä sinisellä on merkattu haettava maa ja vihreällä yrityksen yritystunnus. (OpenCorporates d.)

http://api.opencorporates.com/v0.2/companies/gb/00102498?format=xml

Hakuja voi tehdä myös esimerkiksi käyttäen avainsanoja eli parametria ”search?q=”, jonka perään tulee haettava sana. Koska esimerkiksi Nokia-tyyppinen hakusana palauttaa satoja hakutuloksia, tuloksia on mahdollista rajoittaa käyttäen suodattimena esimerkiksi yritysmuotoa, posti –tai verkko-osoitteen saatavuutta tai maa tai aluekoodia kuten seuraavassa esimerkissä on määritetty punaisella.

http://api.opencorporates.com/v0.2/companies/search?q=nokia&jurisdiction_code=fi

Sovellusrajapinta mahdollistaa myös hakujen tekemisen yritysten johtajistoon kuuluvien henkilöiden nimillä seuraavan esimerkin mukaisesti.

http://api.opencorporates.com/v0.2/officers/search?q=stephen+elop

Kyseinen esimerkkihaku tosin löysi vain yhden tuloksen, joka ei aivan vastannut nykyistä työpaikkaa, joten ”officers” haussa näyttäisi olevan vielä parannettavaa verrattuna hyvin toimivaan yrityshakuun. (OpenCorporates d.)

4.2 Eniro

4.2.1 Yleisesti

Eniro on vuonna 2000 perustettu ruotsalainen osakeyhtiö. Se on tällä hetkellä pohjoismaiden suurin yhteystietoyritys, jolla on toimintaa Pohjoismaiden lisäksi myös Puolassa. Eniro lopetti vuonna 2010 pääasiallisen toimintansa Suomessa. Suomeen jäi jäljelle vain numeropalvelutoimintaa hoitava Eniro Sentraali.

Eniro tarjoaa ilmaiseksi monia yhteystietoihin liittyviä palveluita Pohjoismaiden ja Puolan alueella. Suurin osa palveluista on tällä hetkellä keskittynyt Ruotsiin, Norjaan ja Tanskaan. Esimerkiksi Eniron yrityshakupalvelut on pikkuhiljaa ajettu alas Suomessa ja Puolassa. Eniro tarjoaa myös karttapalvelun, joka sisältää Pohjoismaiden ja Puolan kartat kattaen sekä satelliittikuvat että tiekartat. Eniro tarjoaa tietenkin myös maksullisia palveluita, jotka sisältävät esimerkiksi tarkempaa tietoa liittyen yrityksiin ja puhelinnumeropalveluihin. Kuvassa 19 esitetään Eniron etusivu.
(Wikipedia 2013j.)

Kuva 19. Eniron Ruotsin aloitussivu

Sovelluskehittäjille Eniro tarjoaa neljä ohjelmistorajapintaa, jotka on esitetty seuraavassa luettelossa:

- Company proximity search - basic
- Company search - basic
- Maps - jsAPI
- Review company search - basic

Näistä ohjelmistorajapinnoista toista eli Company search - basic -nimistä ohjelmistorajapintaa kuvataan tarkemmin luvussa 4.2.2.

4.2.2 Company search – basic API

Company search – basic API on Eniron luoma ohjelmistorajapinta, joka mahdollistaa yritysten perustiedon hakemisen Eniron tietokannasta. Ohjelmistorajapinnan ensimmäinen 0.9.99-versio ilmestyi vuoden 2011 kesäkuussa ja nykyinen versio 1.1.3 ilmestyi vuoden 2012 maaliskuussa.

Ohjelmistorajapinnan käyttö tapahtuu HTTP - protokolan GET - pyynnön kautta. Ohjelmistorajapinta käyttää UTF-8 - koodausta URL - kyselyjen suorittamisessa.

URL-kyselylauseen alkuosa on muotoa ”http://api.eniro.com/cs/search/basic” ja se on pohjana jokaiselle Company search - basic API:lla tehtävälle kyselylle.

Eniro API:n vaatimat pakolliset kyselyparametrit ovat

- profile
- key
- country
- version

Yksinkertaisin mahdollinen kysely on muotoa:

http://api.eniro.com/cs/search/basic?profile=testaaja&key=xxxxx9784970xxx0&country=no&version=1.1.3

Kyselyssä sinisellä ja punaisella merkatuissa parametreissa ovat esitettyinä profiilin omistajan profiilinimi ja käyttäjätunnus. Käyttäjänimen ja käyttäjätunnuksen saa osoitteesta http://api.eniro.com. Myös oranssilla merkattu maakoodiparametri ja ohjelmistorajapinnan versio on sisällytettävä mukaan kyselyyn. Hyväksyttävät maakoodit ovat se, no ja dk, jotka tarkoittavat Ruotsia, Norjaa ja Tanskaa. Ohjelmistorajapinnalla ei siis voi hakea suomen yrityksiä.

Yleisesti käytettyjä valinnaisia hakuparametreja ovat

- `search_word`
- `from_list`
- `to_list`
- `geo_area`

Seuraavassa esimerkissä on esitelty kolme esitetyistä valinnaisista parametreista.

http://api.eniro.com/cs/search/basic?to_list=2&from_list=1&profile=testi&key=4860113978497038200&country=se&version=1.1.3&search_word=Telia

Parametrejä to_list ja from_list käytetään yleensä yhdessä. Kyseisessä esimerkissä tämä tarkoittaa sitä että esimerkiksi kymmenestä tuloksesta kaksi ensimmäistä laitetaan listaan, josta sitten poimitaan yksi tulos kyselyn vastaukseksi. Parametri search_word on valinnaisuudestaan huolimatta kyselyn tärkein parametri, koska se on ainoa keino määrittää haluttu yritys. Parametrilla search_word voidaan hakea yrityksen nimen lisäksi myös yritystunnuksella.

Tulokset palautetaan `JSON – formaatissa. Onnistunut kysely palautuu koodin 200 kanssa.`

Kyselyssä tapahtuva virhe johtaa JSON – muotoisen virhekoodin ilmestymiseen vastauksen tilalle. Virhekoodit on esitetty liitteessä 1.

Ilmaiskäytössä ohjelmistorajapinnan käyttöehdot sallivat 10 000 hakua kuukaudessa. Ohjelmistorajapinta on tällöin käytettävissä vain kaikille avoimissa järjestelmissä. Ohjelmistorajapinnan kautta saatuja hakutuloksia ei saa varastoida edes välimuistiin vaan, jokainen haku täytyy tehdä suoraan Company search – basic -ohjelmistorajapintaa käyttäen. Tuloksia ei saa myöskään uudelleen järjestellä tai muokata millään tavalla.

Kun ohjelmistorajapintaa käytetään suljetussa järjestelmässä, täytyy Eniron kanssa tehdä erikseen sopimus käyttömaksusta. Hinnat sovitaan tällöin neuvottelemalla ottaen huomioon tietenkin asiakkaan tarpeet. Samalla on tietenkin mahdollista nostaa kuukausikohtaista hakurajaa.

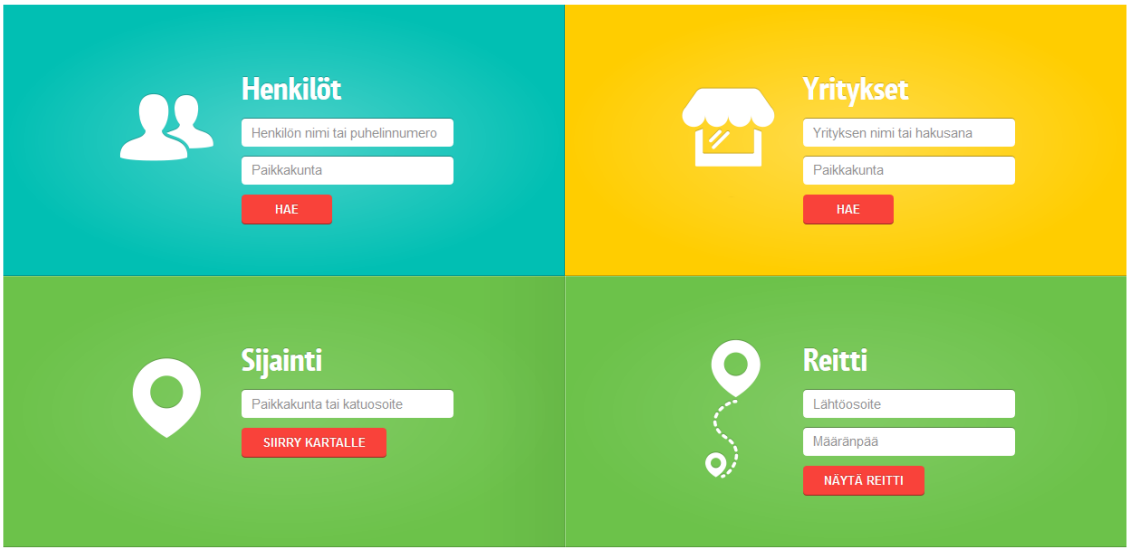
Ohjelmistorajapinnan lähteenä toimii Eniron dokumentaatio, joka löytyy liitteistä 15–19.

4.3 Fonecta

4.3.1 Yleisesti

Fonecta on European Directories nimiseen konserniin kuuluva yhteys- ja mediatietoa välittävä yhtiö, joka on perustettu vuonna 2003. Fonecta on hankkinut nykyisen yhteys – ja mediatietoihin perustuvan markkina-asemansa Suomessa ostamalla pahimmat kilpailijansa ulos markkinoilta. Se osti muun muassa yritystieto alalla toimivan Inoan vuonna 2005 ja pääosan Eniro Finland yhtiön liiketoiminnasta vuonna 2010.

Fonecta tarjoaa Suomessa yhteys- ja yrityshakupalveluiden lisäksi kartta- ja reittipalveluita. Muun muassa suosittu numeropalvelu 020202 on osa Fonecta yhtiötä. Kuvassa 20 esitetään Fonecta-haun etusivu.
(Wikipedia 2013k.)



Kuva 20. Fonecta.fi etusivu

Fonecta tarjoaa ohjelmistokehityksen näkökulmasta neljä ohjelmistorajapintaa, jotka ovat

- Kartat API
- Finder API
- Totaali API
- Osuma API

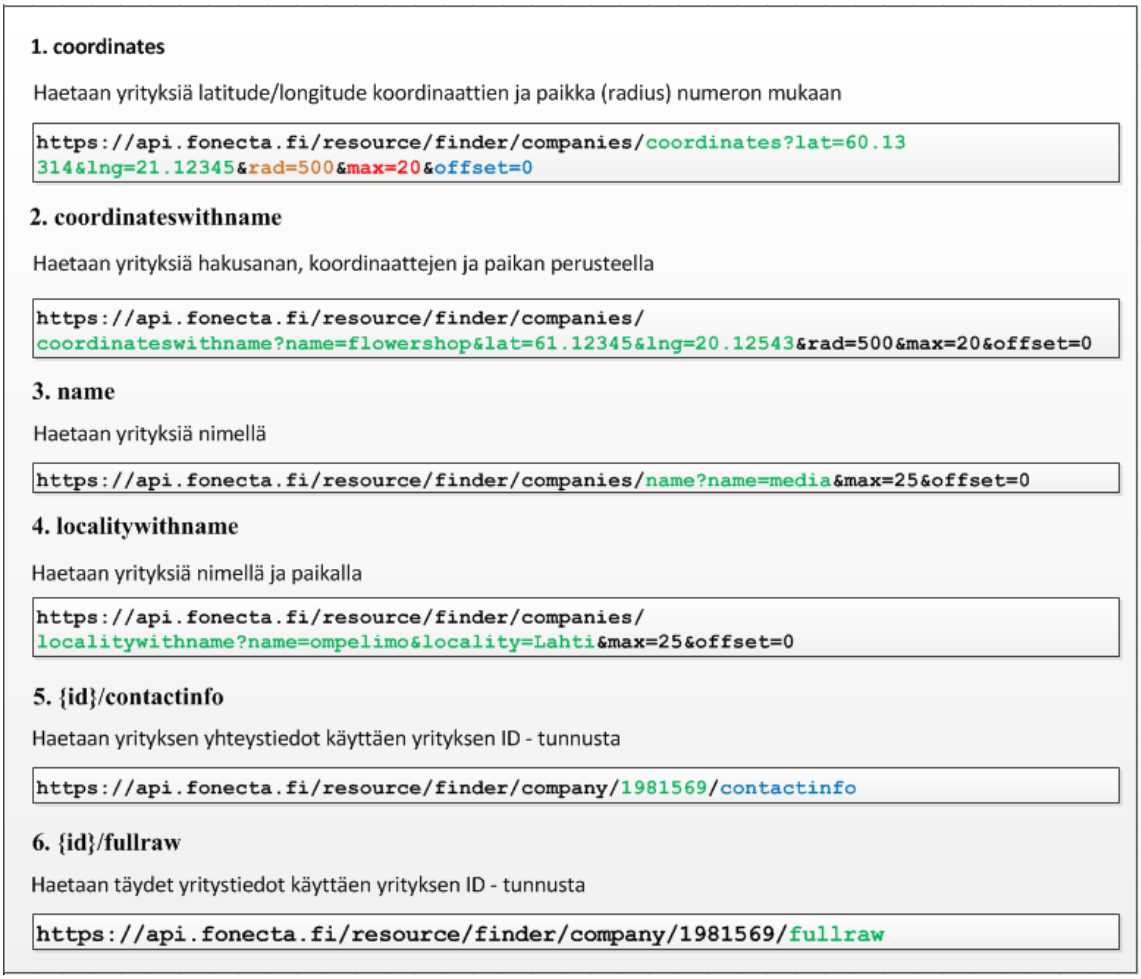
Näistä toista eli Finder API:a käsitellään tarkemmin luvussa 4.3.2.

4.3.2 Finder API

Finder API on Fonecta:n luoma yritys- ja henkilöhakujen suorittamiseen tarkoitettu ohjelmistorajapinta. Ohjelmistorajapinnalla on mahdollista tehdä kolmentyyppisiä kyselyitä, joiden URL-kyselypohjat on esitetty seuraavassa luettelossa.

1. https://api.fonecta.fi/resource/finder/companies/
2. https://api.fonecta.fi/resource/finder/persons/
3. https://api.fonecta.fi/resource/finder/

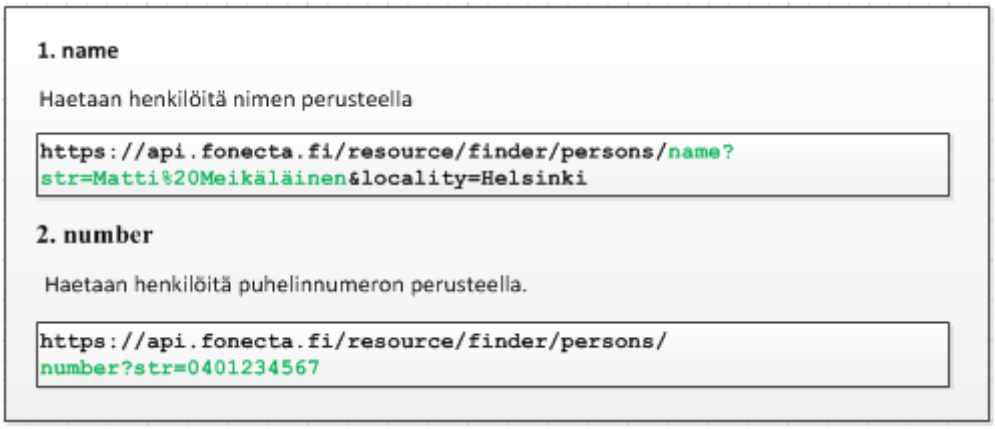
Ensimmäiseksi tarkastellaan ensimmäistä hakupohjaa, jolla haetaan pelkästään yrityksiä. Yrityshaussa voidaan käyttää kuutta eri hakuparametria, jotka on esitelty kuvassa 21. (developer.fonecta.net 2012.)



Kuva 21. Finder API:n yrityshaun hakutyypit

Kuvan kohdassa 1. käsitellään yritysten hakemista koordinaattien perusteella. Hakuosa alkaa ”/coordinates?”-parametrilla, minkä jälkeen kerrotaan koordinaatit kuvan esimerkin mukaisesti. Koordinaateilla haettaessa on myös määritettävä koordinaatteihin liittyvä aluekoodi (esim. rad=500). Hakutulosten määrää on mahdollista rajoittaa käyttämällä ”max” - parametria kuvan mukaisesti. Offset – parametrilla ilmoitetaan hakutermin sallittu poikkeama. Kohdassa kaksi otetaan yrityksen nimi käyttöön koordinaattien lisäksi. Tämä parantaa hakutuluksia kun samoilla koordinaateilla on olemassa useita yrityksiä (esim. monikerroksiset kauppakeskukset). Kohdassa kolme haetaan yrityksiä käyttäen pelkkää nimeä. Tällöin ei tosin ole mahdillista rajata hakutuloksia alueellisesti. Neljännessä kohdassa hakutermin lisäksi on annettu locality – parametri ilmoittamaan halutun hakualueen. Hakualueen rajoittaminen on hyödyksi esimerkiksi haettaessa yrityksiä epämääräisellä hakutermillä. Kohdissa viisi ja kuusi haetaan yritysten tietoja yrityksen ID-tunnuksen perusteella pyytäen joko pelkät yhteystiedot (contactinfo) tai yrityksen täydet tiedot (fullraw).

Ohjelmistorajapintaa käyttäen on mahdollista hakea henkilöitä joko nimen tai puhelinnumeron perusteella kuvan 22 mukaisesti. Rakenteellisesti täytyy mustaa että aiemman URL-osoitteen ”/company/”-kohdan sijasta käyttää ”/persons/” - kohtaa, jolla kerrotaan ohjelmistorajapinnalle hakutulosten tyyppi. (developer.fonecta.net 2012.)



Kuva 22. Finder API:n henkilöhaku

On myös mahdollista hakea yhtä aikaa sekä yrityksiä että henkilöitä. Tätä vaihtoehtoa ei tosin kannata käyttää kaikissa tilanteissa, koska esimerkiksi kuvan 23 mukaisella ”Hiltunen” – hakusanalla voi löytää Helsingistä melko paljon yrityksiä ja henkilöitä. (developer.fonecta.net 2012.)

Kuva 23. Finder API: yhdistelmähaku

Käyttäjien tunnistamisessa Finder API:ssa käytetään OAuth 2.0 protokollaa.

4.4 YTJ

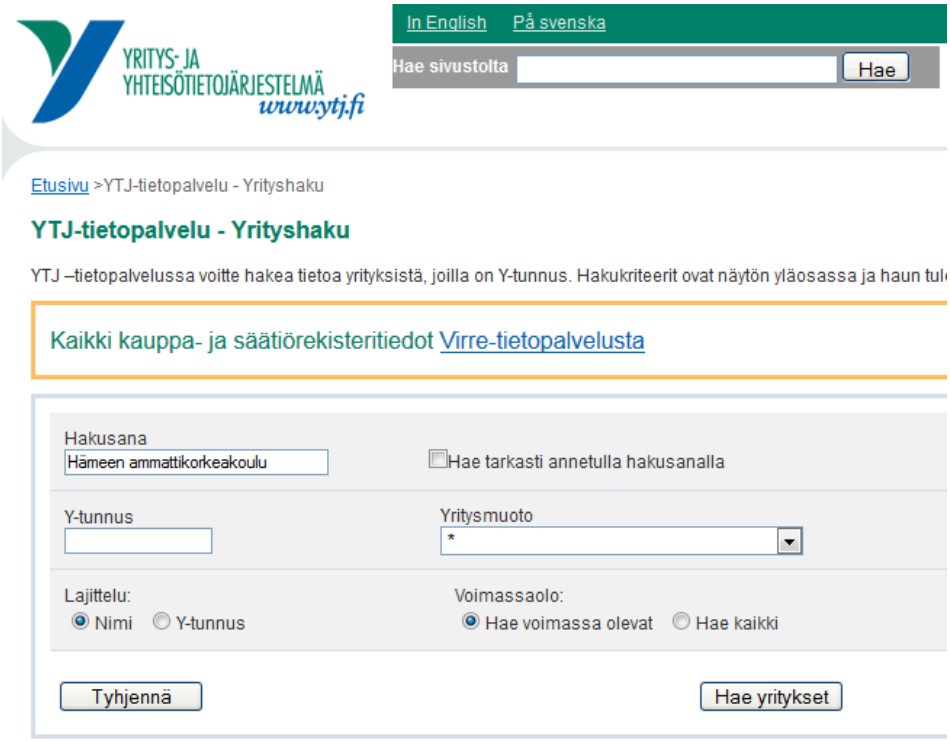
YTJ on rekisteri- ja patenttihallituksen ja verohallinnon yhteisesti ylläpitämä yritystietojärjestelmä. Yritys- ja yhteisötietojärjestelmä sisältää yritykset ja yhteisöt, jotka löytyvät seuraavan listan rekistereistä.

- säätiörekisteri
- kaupparekisteri
- arvonlisäverovelvollisten rekisteri
- ennakkoperintärekisteri
- verohallinnon asiakasrekisteri
- työnantajarekisteri

Myös yritykset ja yhteisöt, joista on tehty perustamisilmoitus, sisältyvät tietojärjestelmään, vaikka näitä yrityksiä tai yhteisöjä ei vielä olisi lisätty rekistereihin.

YTJ-sivusto tarjoaa useita palveluita. Se tarjoaa ilmaisen hakupalvelun yritysten perustietojen etsimiseen. YTJ:n asiointipalvelun kautta on myös

mahdollista perustaa esimerkiksi osakeyhtiö verkon kautta. Verkkosivuston kautta on myös mahdollista päivittää olemassa olevan yrityksen rekisteritietoja. Näiden lisäksi YTJ tarjoaa maksullisia poimintapalveluita sen yhteystyökumppaneiden kautta, joita ovat esimerkiksi Kauppalehti ja Fonecta. Kuvassa 24 esitetään YTJ:n yrityshaku. (YTJ.)



Kuva 24. Kuvassa on esitettynä YTJ-tietopalvelun yrityshaku.

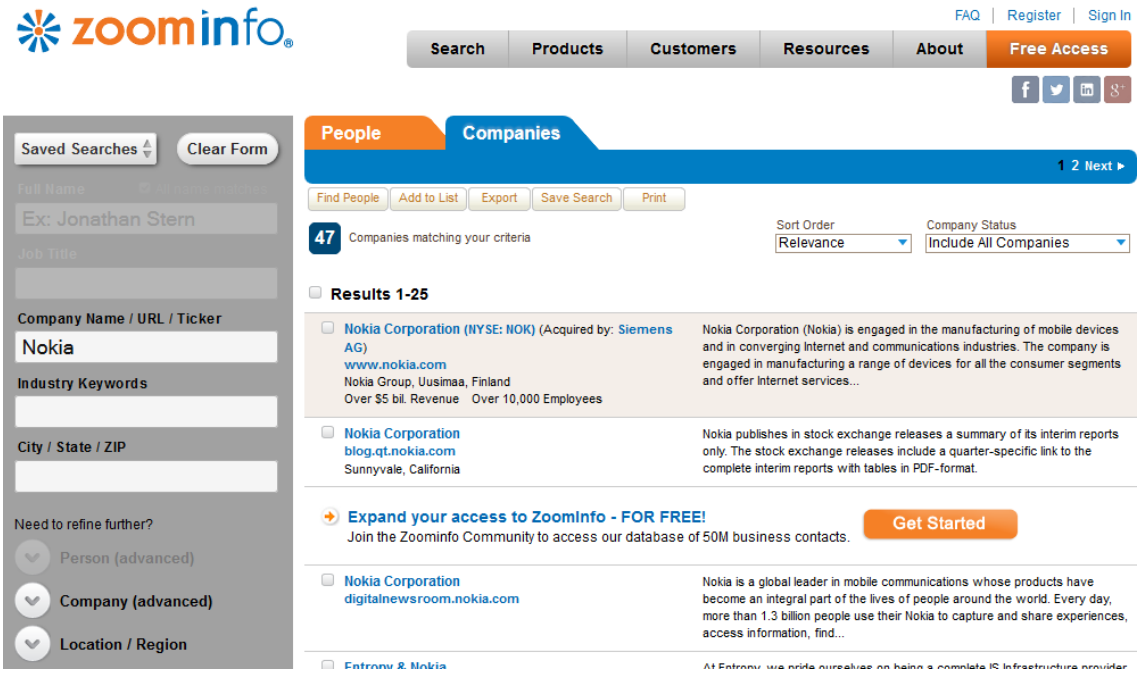
YTJ ei tarjoa ohjelmistorajapintoja tiedon hakemiselle sen palveluiden kautta, mutta sen yrityksien perustietojen hakemiseen käytettävä ilmainen yrityshaku tarjoaa tuhat hakua päivässä yhtä IP-osoitetta kohti. Palvelun käyttöehdoissa määritellään myös että sitä ei saa käyttää hakurobottien kautta. Nämä ehdot kuitenkin mahdollistavat palvelun käyttämisen tapauksessa, jossa saadaan suora URL-osoite haluttuun tietoon jonkin palvelun kautta. Tässä tapauksessa koko html-sivu voidaan ladata käyttäen ohjelmointikieltä. Ladatusta sivusta poimitaan tämän jälkeen html-elementtien otsikoiden perusteella halutut tiedot. YTJ:n kautta on myös mahdollista saada pientä maksua vastaan yritysten perustietoja sisältäviä tietokantoja.

4.5 ZoomInfo

4.5.1 Yleisesti

Zoominfo on yhdysvaltalainen yritys, joka on perustettu vuonna 2000. Se tarjoaa yhteys-, yritys- ja henkilötietoja sekä ilmaiseksi että maksullisesti. Zooinfo:n verkkosivujen hakupalvelun avulla on mahdollista tehdä ilmaisia hakuja esimerkiksi yrityksistä ja henkilöistä olematta millään tavalla osallisena Zoominfo:n toiminnassa. Tällöin kaikki hakutulokset eivät tosin

ole luettavissa. Kuvassa 25 on esitettynä Zoominfo:n ilmainen hakuosuus. (ZoomInfo a)



Kuva 25. Kuvassa on Zoominfo:n hakusivu.

Ilmaisen yhteyden dataan saa myös Community edition:in kautta. Community edition:issa käyttäjät luovuttavat omat kontaktinsa yhteisölle ja ZoomInfo:lle ja saavat vastineeksi oikeuden tehdä hakuja ZoomInfo:n tietokannasta. Kontaktit kerätään yleensä ZoomInfo:n ohjelmalla esimerkiksi Outlook - tai Gmail – käyttäjätileiltä. Kontaktien keräämiseen Outlook – sähköpostipalvelusta voidaan käyttää esimerkiksi ZoomInfo Contact Contributor nimistä ohjelmaa. (ZoomInfo a.)

Maksulliselta puolelta löytyy data- ja ZoomInfo Pro -palvelut. ZoomInfo Pro mahdollistaa Community edition:ia tarkempien ja monipuolisempien hakutulosten saamisen ilman tarvetta jakaa omia kontaktitietojaan. Datapalveluiden puolella ZoomInfo tarjoaa esimerkiksi jotain kyseistä tarkoitusta varten koottua tietoa, jolla voidaan päivittää esimerkiksi yrityksen kontaktitietokanta. (ZoomInfo b.) (ZoomInfo c.)

ZoomInfo:n tietokannat kattavat parhaiten Yhdysvaltojen, Kanadan, Australian ja Iso-Britannian yritykset ja kontaktit. Muista maista tuloksia löytyy vaihtelevasti. Esimerkiksi Suomesta tunnetaan yleisesti vain isot yritykset kuten Ruokakesto tai Nokia.

ZoomInfo:lla on The ZoomInfo Partner API niminen ohjelmistorajapinta, jota käsitellään luvussa 4.5.2.

4.5.2 The ZoomInfo Partner API

The ZoomInfo Partner API mahdollistaa kolmansille osapuolille mahdollisuuden käyttää ZoomInfo:n haku- ja vertailuominaisuuksia ja muuta ZoomInfo:n dataa riippuen ZoomInfo:n kanssa sovituista kaupallisista oikeuksista. Yleisimmät hakutyypit ovat luonnollisesti henkilö- ja yrityshaku, jotka haetaan GET - pyyntöjen avulla.

Ohjelmistorajapinnan kanssa on käytössä kolmentyyppisiä hakumäärärajoja. Ensimmäisenä on yleinen hakuraja, jolla määritellään ZoomInfo:n hakumäärät jonkin tietyn ajanjakson aikana. Toinen hakuraja on hakutyyppikohtainen. Sillä määritellään, että kuinka monta tietyntyyppistä hakua on mahdollista tehdä ajanjakson aikana. Kolmas hakuraja puolestaan määrittelee, kuinka paljon hakutuloksia on mahdollista saada vastauksena hakupyyntöön. Tämä rajoitus ei koske yritys- ja henkilöhaku pyyntöjä. (ZoomInfo d.)

Käyttäjän tunnistamiseksi jokainen yhteistyökumppani saa tunnistuskoodin ja salasanan, jotka esitetään aina osana jokaista hakukyselyä. Salasana koodataan MD5 salausalgoritmilla käyttäen UTF-8 formaattia kyselyn tietojen ja päivämäärän kanssa enenen kyselyn lähettämistä ZoomInfo:lle.

Kaikki ZoomInfo:lta saadut tulokset tulevat XML – formaatissa, jota on mahdillista käsitellä ohjelmointikielten avulla. The ZoomInfo Partner API:n hakukyselyt voidaan jakaa kahteen ryhmään ja kuuteen osaan:

- Henkilöhaku
    - People Search Query
    - Person Detail Query
    - Person Match Request

- Yrityshaku
    - Company Search Query
    - Company Detail Query
    - Company Match Request

People Search Query eli henkilöhakukysely on tarkoitettu henkilöiden etsimiseen. Henkilöhakukysely palauttaa vastauksessa käyttäjälle hakutermejä vastaavien henkilön tulokset, jotka eivät sisällä tarkkoja tietoja henkilöistä. Esimerkki henkilöhakukyselystä on esitetty seuraavassa esimerkissä.

```
http://partnerapi.zoominfo.com/partnerapi/xmloutput.aspx?query_type=people_search_query&pc=&personTitle=VP+Marketing&IndustryClassification=200714&State=Massachusetts&key=
```

Esimerkissä ensimmäinen varsinainen parametri on quary_type, jolla määritellään kyselyn tyyppi, joka on esimerkin tapauksessa people_search_query eli henkilöhakukysely. Company Search Query toimii samalla tavalla vaihtaen tietenkin kyselyntyypin. Toinen parametri (pc) on yhteistyökumppanikoodi, jota tarvitaan käyttäjän tunnistamisessa. Kolmas parametri on personTitle eli henkilön asema yrityksessä. Parametri on kä-

tevä rajaamaan hakua isommasta henkilöjoukosta. Viimeinen esimerkin merkittävistä parametreista on key, johon tulee käyttäjän salasana. (ZoomInfo d.)

Henkilötietokyselyn (Person Detail Query) tarkoituksena on saada henkilön täydet tiedot käyttäen apuna hakuparametreja esimerkiksi henkilöhakukyselyn kautta saatua henkilötunnistetta (PersonID) tai henkilön sähköpostiosoitetta EmailAddres. PersonID on siis tunnistenumero, joka saadaan yleensä henkilöhaun tuloksena. EmailAddres eli sähköpostiosoite on PersonID parametrin tapaisesti valinnainen hakuparametri, jolla pyritään määrittämään haluttu henkilö. Seuraavassa esimerkissä esitetään henkilötietokysely, jossa käytetään hakuterminä PersonID-hakuparametria.

http://partnerapi.zoominfo.com/partnerapi/xmloutput.aspx?query_type=person_detail&pc=&PersonID=457877&key=

Yritystietokysely (Company Detail Query) toimii henkilöhakukyselyn kanssa vastaavasti pyrkien palauttamaan hakijalle haettavan yrityksen tiedot. Yritystietokyselyssä on uutena hakuparametrina CompanyDomain ja CompanyID. CompanyID on yrityksen henkilökohtainen tunniste tunnus ZoomInfon palvelussa, joka saadaan yrityshakukyselyn perusteella. ComapanyDomain on yrityksen verkkosivujen osoitteen perusosa, joka saadaan joko yrityshakukyselystä tai kyseisen yrityksen yhteystiedoista. Seuraavassa esimerkissä on esitettynä yritystietokysely, jossa esitettynä CompanyDomain hakuparametri. (ZoomInfo d.)

http://partnerapi.zoominfo.com/partnerapi/xmloutput.aspx?query_type=company_detail&pc=&CompanyDomain=www.nokia.com&key=g576ethsgfhdfgdt6

Henkilövastaavuuspyyntöjä (Person Match Request) ja yritysvastaavuuspyyntöjä (Company Match Request) käytetään, kun halutaan mahdollisimman tarkkoja hakutuloksia henkilöistä tai yrityksistä. Seuraavassa esimerkissä on esitetty yritysvastaavuuspyyntö.

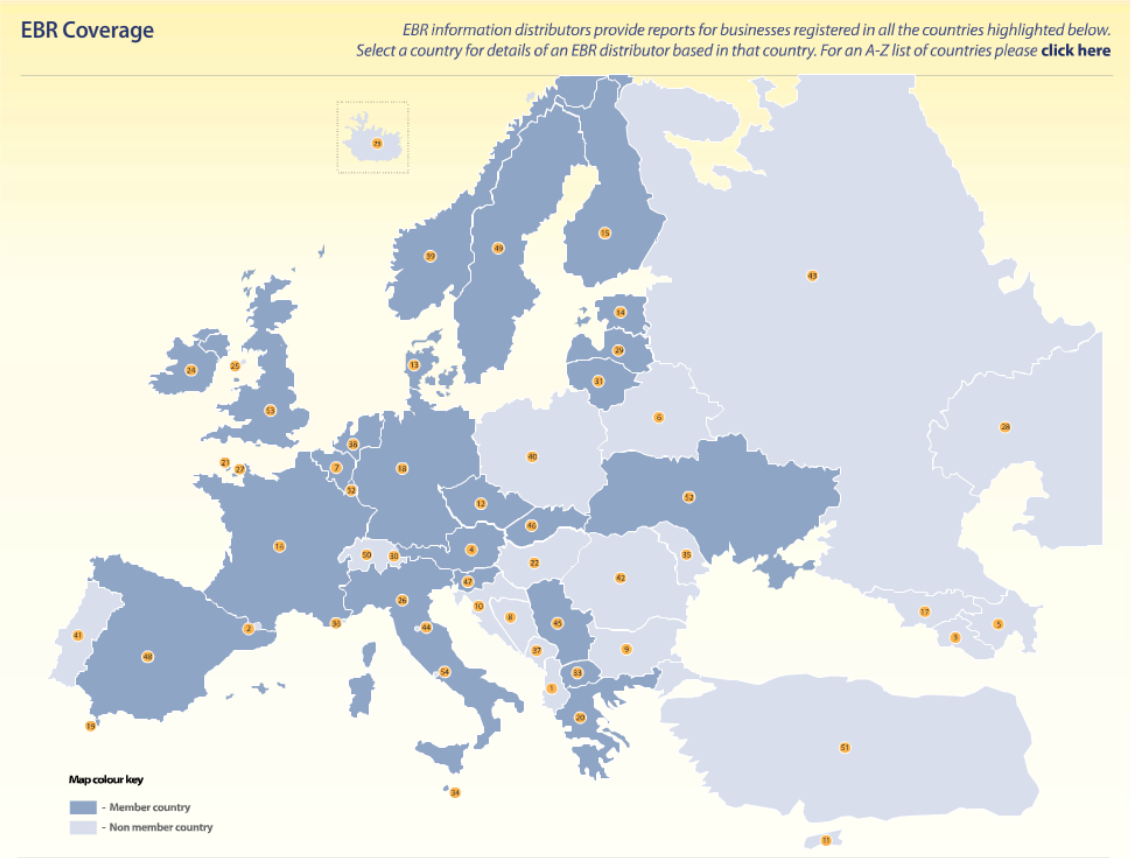
http://partnerapi.zoominfo.com/partnerapi/match/company?pc=&Name=dell&ticker=nasdaq:dell&phone1=123384400 &street1=One Dell Way&city1=Round Rock&state1=texas&street2=20 Park Plaza&city2=boston&state2=Massachusetts

Esimerkissä yritysvastaavuuspyynnön avulla pyritään saamaan tarkalleen yksi yritys hakutulokseksi, josta yritysvastaavuuspyyntö puolestaan palauttaa käyttäjälle yrityksen tiedot. Esimerkissä yrityksen nimen tukena on käytetty myös muun muassa puhelinnumeroa, katuosoitetta ja kaupunkia karsimaan halutun yrityksen mahdolliset samaa Name-parametria eli yrityksen nimeä vastaavat väärät vaihtoehdot hakutuloksista. Voidaan siis oikeastaan todeta että henkilövastaavuuspyyntö ja yritysvastaavuuspyyntö ovat ZoomInfo:n ohjelmistorajapinnassa absoluuttisen tuloksen saamista varten. (ZoomInfo d.)

Ohjelmistorajapinnan käyttäminen yritystoiminnan osana ei ole ilmaista missään tilanteessa. Mahdollisuus pienimuotoiseen ilmaiseen yksityiskäyttöön on olemassa, mutta ZoomInfolla on oikeus muuttaa palvelu maksulliseksi oman tahtonsa mukaisesti.

4.6 EBR (European Business Register)

Euroopan kaupparekisteri on rekisteri, joka koostuu kansallisten kauppa- tai yritysrekisterien verkostosta. Rekisteri perustettiin alun perin vuonna 1992, jolloin siihen kuului 12 maata käsittäen myös Suomen. Nykyään rekisteri käsittää 26 valtiota, joista 20 on Euroopan Unionin jäseniä. Rekisterin tarkoituksena on tarjota helppo pääsy yritysten tietoihin suoraan verkkolähteestä. Euroopan kaupparekisteri käsittää suuren osan Euroopasta lukuun ottamatta monia Itä-Euroopan ja Balkanin maita. Kuvassa 26 on esitetty Euroopan kaupparekisterin kattavuusalue. (EBR 2010a.)



Kuva 26. Kuvassa esitetään Euroopan kaupparekisterin kattavuusalue.

Palvelu toimii käytännössä siten, että käyttäjä ohjataan Euroopan kaupparekisterin tiedon jakelijoille, joiden kautta on mahdollista saada kattavia yritys- ja henkilötietoja maksua vastaan. Suomessa virallisina jakelijoina toimii tällä hetkellä Kauppalehti ja Asiakastieto. Palvelun hyvinä puolina voidaan pitää luetettavaa, ajan tasalla olevaa ja kattavaa tietoa. Hyvää on myös eri jakelijoiden kautta saatava kielivalikoima. Huonona puolena on puolestaan tiedon maksullisuus. (EBR 2010b.)

4.7 Vertailu ja johtopäätökset

Yrityshakupalveluiden osalta on tutkittu kuusi yritystietopalvelua, joiden kautta on mahdollista saada yritysten tietoja. Kuten aiemmissa luvuissa on todettu, niin kaikilla vaihtoehdoilla on sekä hyviä että huonoja puolia. Taulukossa 2 on esitettynä yrityshakupalveluiden ominaisuuksien vertailua.

Taulukko 2. Taulukossa on vertailtu yrityshakupalveluiden ominaisuuksia.

	OpenCorporates	Eniro	Fonecta	YTJ	ZoomInfo	EBR
Ilmainen	Kyllä	Osittain	Ei	Kyllä	Ei	Ei
Ohjelmistorajapinta	Kyllä	Kyllä	Kyllä	Ei	Kyllä	Ei
Kattavuus (Suomi)						
Kattavuus (Muut Pohjoismaat)						
Perustietojen laatu						
Taloustiedot						
Soveltuvuus toteutukseen						

Taulukossa tumma väri kuvaa hyvää ja vaalea huonoa.

Vertailun perusteella voidaan todeta, että esimerkiksi taloustietoja ei ole mahdollista saada ilmaiseksi, joten pyritään toimimaan ilman niitä. Ilmaisista palveluista yritysten perustietoja Suomen osalta tarjoavat YTJ ja OpenCorporates. YTJ kattaa koko Suomen yrityskannan, mutta sieltä ei voida suoraan hakea tuloksia oman hakupalvelun käyttöön YTJ:n käyttöehtojen takia. OpenCorporates kattaa suurimman osan Suomen yrityksistä ja tarjoaa tietonsa ilmaiseksi ohjelmistorajapintansa kautta. Ongelmana on kuitenkin se, että OpenCorporatesin omat perustiedot Suomen yrityksistä ovat riittämättömät. Luvussa 4.1 todettiin että OpenCorporates antaa kuitenkin haettavasta yrityksestä suoran linkin YTJ:n tietosivulle, mikä mahdollistaa sivuston tietojen käyttämisen rikkomatta käyttöehtoja. Tästä syystä Suomen yritystiedot kannattaa hakea käyttäen sekä OpenCorporates:ia että YTJ:n tietopalvelua.

Tutkitut vaihtoehdot mahdollistavat myös yleisemmin Pohjoismaita käsittelevän hakupalvelun toteuttamisen. OpenCorporates:in perustiedot Ruotsista ja Norjasta ovat hieman kattavammat kuin Suomesta, joten sitä kautta olisi mahdollista saada ilmaiseksi Pohjoismaiden perustietoja. Tanskan tietojen pitäisi tulla saataville OpenCorporates:iin lähiaikoina. Eniro kuitenkin tarjoaa OpenCorporetes:ia kattavamman perustiedot Ruotsin, Norjan ja Tanskan osalta, mutta sen palvelut ovat maksullisia suljetussa ympäristössä kuten kirjautumisen takana. Kattavuus syistä Eniroa kannattaisi ehdottomasti käyttää ensisijaisena palveluna yrityshauen toteuttamiselle Ruotsin, Norjan ja Tanskan osalta.

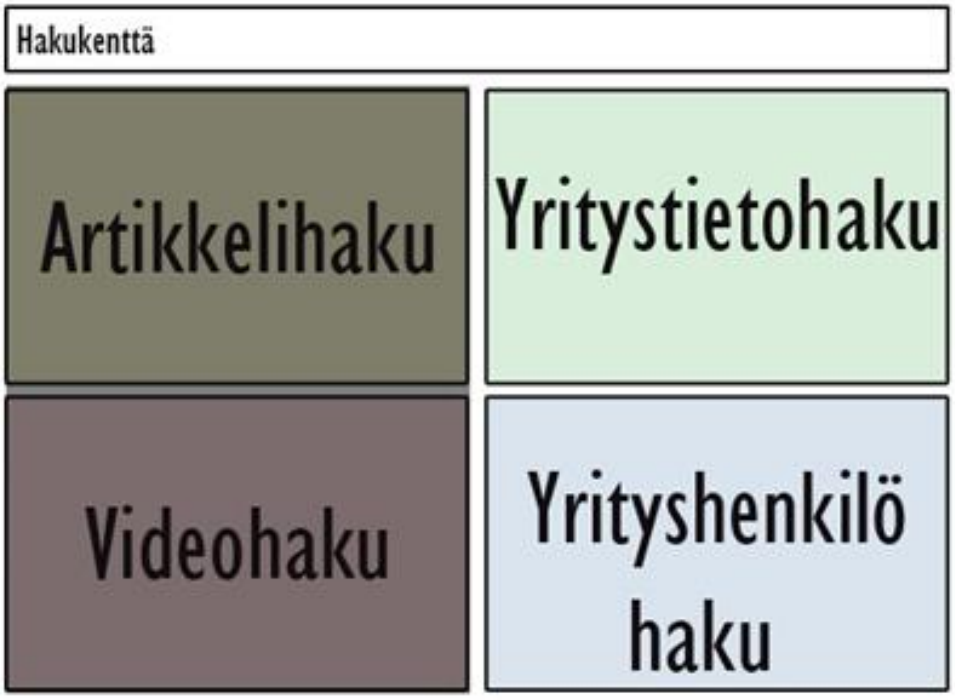
Tehtäessä koko Pohjoismaita kattava yrityshakupalvelu kannattaa ehdottomasti käyttää nopeasti kehittyvää OpenCorporates:in tietoja pohjana, jonka päälle lähdetään rakentamaan kattavanmaa palvelua käyttäen esimerkiksi YTJ:tä ja Eniroa apuna.

5 OHJELMAN TOTEUTUS

Ohjelman toteutusosassa käsitellään ensiksi ohjelman tavoitteita ja rajoitteita. Tämän jälkeen esitellään teoriaosuuden vertailuihin ja johtopäätöksiin pohjautuen toteutettavan ohjelman suunnitelma. Suunnitelmaa seuraa toteuttamisvaihe, jossa kerrotaan vaihevaiheelta koodintoiminta ja pohditaan lopputulosta suhteessa tavoitteisiin. Toteutusosan lopussa käsitellään myös lyhyesti koodin testausta.

5.1 Ohjelman määrittely

Määrittelyosassa esitellään ohjelman osalta tavoitteet ja rajoitteet. Tarkoituksena on luoda yritystieto- ja kontaktihaku toiminnallisuudet osana isompaa kokonaisuutta, joka on esitettynä kuvassa 27. Kokonaisuuden tarkoituksena on siis saada yhdellä hakusanalla mahdollisimman monipuoliset tiedot haettavasta yrityksestä. Omaan osuuteeni kuuluu siis kuvan oikean puolen suunnittelu ja toteuttaminen. Toteutus on tarkoitus tehdä mielellään ilmaisia tai vähän maksavia palveluita käyttäen. Toteutuksen kannalta vaihtoehdot ohjelmointikielille ovat PHP, Java ja C#.



Kuva 27. Kuvassa on esitettynä projektin alkuperäinen suunnitelma.

Yritystietohaussa tavoitteena on suunnitella ja toteuttaa hakutoiminto, jonka avulla on mahdollista saada haettavan yrityksen perustiedot ja mahdollisuuksien mukaan myös taloustiedot. Työnantajan puolelta ei kuitenkaan ole määritelty, mitä maita kyseisen palvelun tulisi sisältää, joten tavoitteena on kattaa ainakin Suomen alue. Kontaktihaun tavoitteena on puolestaan luoda ohjelmakoodi, jonka avulla on mahdollista saada haettavaa yritystä vastaavien henkilöiden perustiedot, yhteystiedot ja kuvat. Yhtenä tavoitteena on, että hakumoduulit voisivat toimia mahdillisuuksien mukaan

myös erikseen. Tämä lisäisi mahdollisuutta käyttää hakumoduuleja muissakin kuin niiden alkuperäisissä tarkoituksissa.

5.2 Suunnitelma

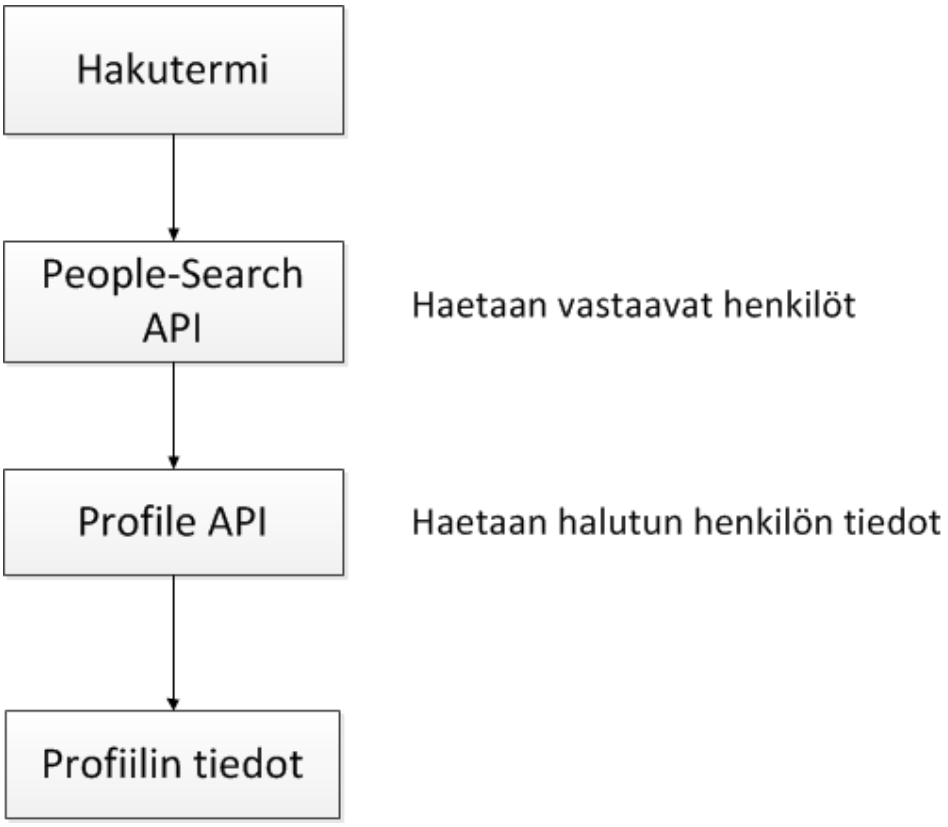
Tavoitteena on luoda verkkosovelluksen osa, joka koostuu neljästä osasta. Nämä osat ovat kirjautuminen, hakusanaa vastaavien tulosten hakeminen, kontaktihaku ja yritystietohaku. Sovellus on tarkoitus jakaa ohjelmamoduuleihin, joiden pitäisi toimia mahdollisemman paljon itsenäisesti. Tämä helpottaisi esimerkiksi koodin osien käyttämistä myös muihin kuin sen alkuperäiseen tarkoitukseen.

Sovelluksen palvelinohjelmointikielenä käytetään PHP-ohjelmointikieltä luvun 2.2.5 vertailun perusteella. Sovelluksessa käytetään myös HTML-hypertekstikieltä, jonka yhteydessä käytetään hieman myös asiakaspuolella käytettävää JavaScpipt-ohjelmointikieltä.

Sovelluksen ohjelmointirajapinnoista saadut palautusarvot on tarkoitus pyytää JSON-formaatissa, koska se on ohjelman ajamisen kannalta XML-formaattia tehokkaampaa ja soveltuu kaikkien toteutuksessa vaihtoehtoina käytettävien ohjelmistorajapintojen tietojen tuomiseen internetistä.

Sovelluksen kaikkien ohjelmistomoduulien käyttöön on tarkoitus tehdä OpenCorporates-ohjelmistorajapintaa hyödyntävä moduuli, jossa käyttöliittymän hakutermillä haetaan hakutermiä vastaavat yritykset ja esitetään ne asiakaskäyttäjälle, joka tämän jälkeen valitsee niistä yhden. Tämän jälkeen muut projektin moduulit hyödyntävät valitun yrityksen nimiä.

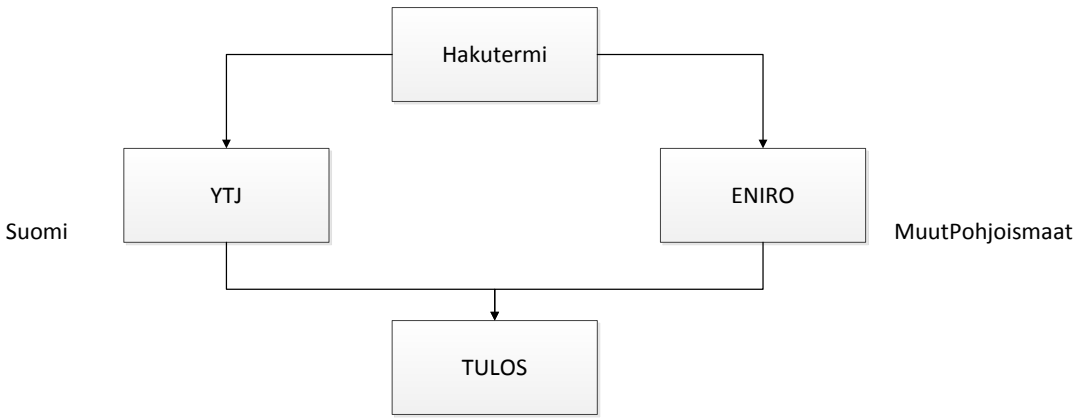
Kohdan 3.4 johtopäätösten mukaisesti kontaktihaun toteutuksessa käytetään Linkedinin ohjelmistorajapintoja, joiden avulla hankitaan tarvittavat tiedot yrityksiin liittyvistä henkilöistä. Kontaktihaun toteutuksessa käytetään Linkedinin Profile API ja People-Search API nimisiä ohjelmistorajapintoja. Kontaktihaku on suunniteltu toimimaan vaiheittain. Ensin käyttäjän valitsema hakutermi lähetetään Linkedinin People-Search API -ohjelmistorajapinnalle, joka palauttaa hakutermiä vastaavien henkilöiden ID-tunnukset. Näitä tunnuksia käyttäen on sitten tarkoitus kysyä varsinaiset profiilitiedot Linkedinin Profile API -ohjelmistorajapinnalta, minkä jälkeen tulos tai tulokset olisi tarkoitus tulostaa käyttöliittymään. Prosessi on esitettynä kuvassa 28.



Kuva 28. Kontaktihaun toteutus

Kontaktihaku vaatii OAuth-tunnistautumisen, jota on tarkoitus kysyä käyttäjältä heti sivustolle tullessa. Tämän jälkeen tunnistautuminen pysyy voimassa joko selaimen sulkemiseen tai sivustolta poistumiseen asti riippuen käytettävästä verkkoselaimesta ja sen asetuksista.

Yritystietohaun toteuttaminen on tarkoitus toteuttaa luvun 4.7 johtopäätöksien mukaisesti käyttäen hyödyksi Eniro- ja YTJ-palveluita, joiden kautta hankitaan yritystietohaun tarvitsemat yritysten perustiedot. Yritystietohaku on suunniteltu toimimaan siten, että OpenCorporates:ilta saadusta listasta valitulla yrityksen nimen avulla haetaan yrityksen perustiedot joko YTJ:stä tai Enirosta riippuen maasta. Yritystietohaun suunnitelma on esitettynä kuvassa 29.



Kuva 29. Kuvassa on esitettynä yritystietohaun alkuperäinen suunnitelma.

5.3 Toteuttamisvaihe

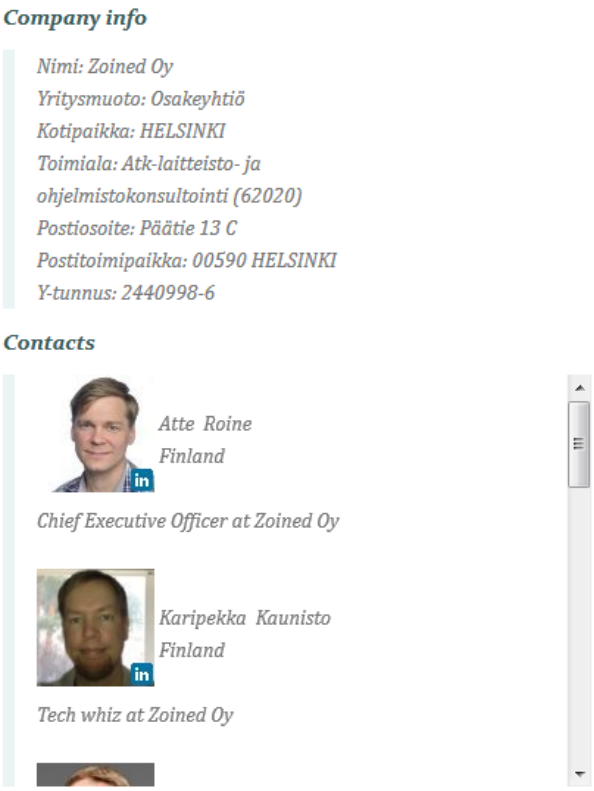
5.3.1 Yleisesti

Toteutus onnistui suunnitelmaan nähden melko hyvin. Yritystietohakuosan toteutus onnistui hyvin ottaen huomioon käytössä olevat tietolähteet. Tulosten kattavuuteen vaikutti hieman negatiivisesti se, että OpenCorporates ei sisällä kaikkia Suomen ja muiden Pohjoismaiden yrityksiä, joten sen kautta ei ole tällä hetkellä mahdollista saada täysin kattavaa yrityskarttaa. OpenCorporates ja Eniro eivät myöskään toimineet täysin keskenään. Niillä saattoi olla esimerkiksi erilaiset nimet joillekin yrityksille. Eniro myös summasi yrityksen tytäryhtiöitä yhden nimen alle, kun vastaavasti OpenCorporates tunsi ne kaikki erikseen. Tämä johti pieniin yhteensopivuusongelmiin ohjelmistorajapintojen tietokantojen välillä. Suomen osalta tutkittuja tietolähteitä käyttäen päästiin noin 70 %:n kattavuuteen. Muiden Pohjoismaiden osalta mitään varmaa prosenttilukua ei voida antaa. Pohjoismaista Tanskan tuloksia ei vielä ollut mahdollista saada toteutuksen tekemisen aikana, koska OpenCorporates:lla ei vielä ole Tanskan tietoja. Tanska on kuitenkin otettu huomioon koodin puolella ja sen tulokset tulevat näkyviin, kun tiedot kohtapuoliin saadaan OpenCorporates:n tietokantaa. Linkedin kontaktihaku onnistui myös melko hyvin. Toteutuksessa yllätti Linkedinin tulosten kattavuus myös pienien yritysten keskuudessa, joissa hakutulokset olivat hyvinkin tarkkoja. Haettaessa isompia yrityksiä hakutuloksia heikensi välillä se, että hakutuloksessa hyvinkin korkealla saattoi päästä henkilö, joka ei ollut missään tekemisessä kyseisen yrityksen kanssa. Tämä saattoi johtua esimerkiksi Linkedinissä esillä olevasta CV:stä, jossa oli mainittuna kyseinen yritys edellisenä työnantajana. Eri maiden yrityksiä haettaessa ei ilmennyt suurempia eroja. Kuvassa cx on esitettynä esimerkkitulostus yritys- ja kontaktihausta.

Suunnitelmaan nähden muutoksia tuli vähän. Merkittävin muutos tuli yritysten perustietojen hakemiseen, johon Eniron ja OpenCorporates:n pienten yhteensopivuusongelmien takia täytyi kehittää varasuunnitelma tilanteelle, jossa vastaavuutta ei löydy. Tämä on esitetty kohdassa 5.3.4.

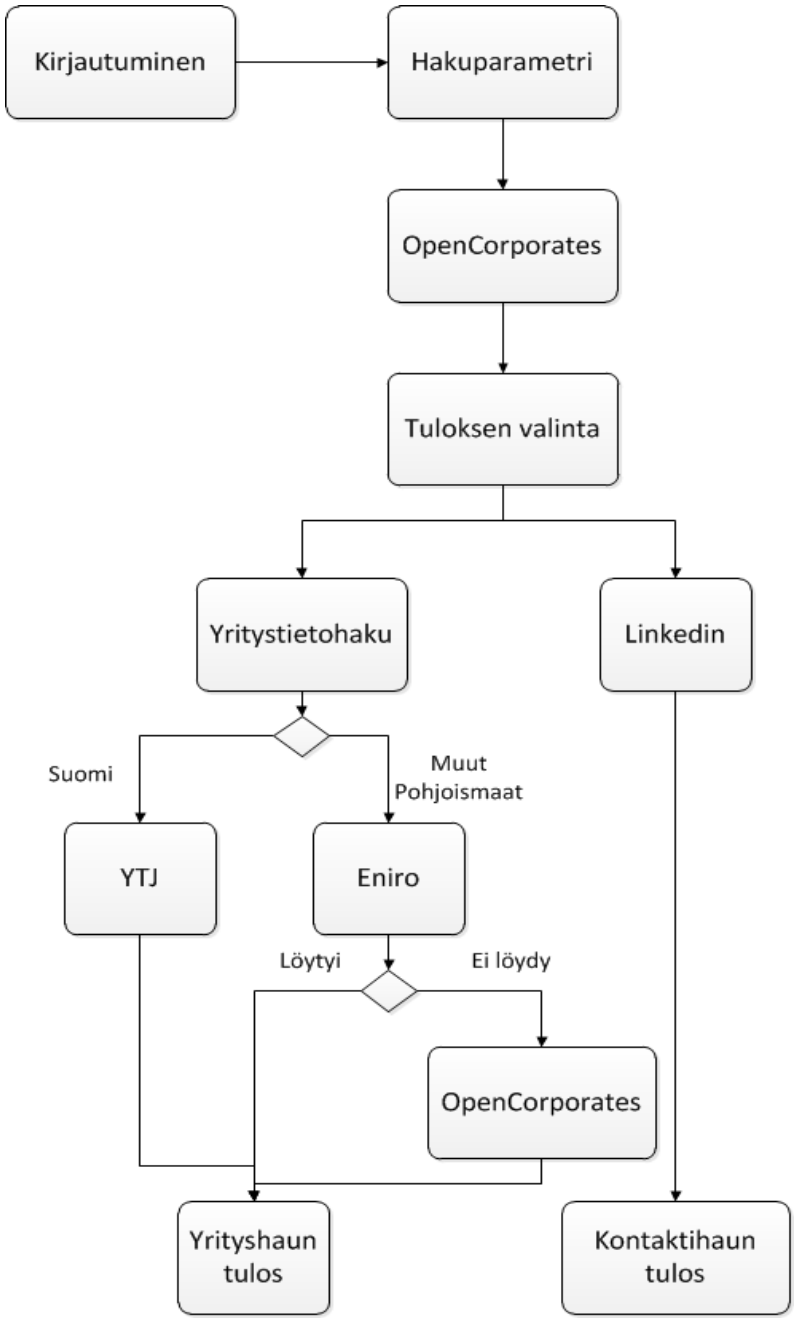
Toteutuksessani käytettiin palvelinpuolella PHP-ohjelmointikieltä. Aiakkaan puolella käytössä oli pääasiassa html-hypertekstiä. Toteutus koostuu kirjautumisesta, hakuparametrin käsittelystä, yritystietohausta ja kontaktihausta. Ohjelman tekemisessä käytettiin hyväksi XAMPP-palvelinta, jota käytettiin testiympäristönä PHP-ohjelmistokielen ajamista varten. Koodin tekemiseen käytettiin vapaan lähdekoodin Notepad++-koodieditoria.

Toteutusosan ohjelmakoodi osuuteni koostuu login.php, linkedin.php, openCorporates.php ja search.php nimisistä PHP-moduuleista. Ohjelmakoodi esitellään kokonaisuudessaan liitteissä 4-14. Toteutusosan search-moduuli vaatii openCorporates.php-moduulin toimiakseen. login.php ja linkedin.php moduuleja voi käyttää myös erikseen antaen oikean POST-muuttujan syötteellä. Moduulit sisällytetään Marko Ylitalon tekemään käyttöliittymään PHP-ohjelmointikielen include-komentoa käyttäen. Toteutusosuuteni on esitettynä kokonaisuutena kuvassa 30.



Kuva 30. Kuvassa esitetään oman toteutusosan tuloste.

Kuvassa 31 on esitetty myös toiminnallinen kaavio opinnäytetyön toteutuksesta.



Kuva 31. Kuvassa esitetään kokonaisuutena oma osuuteni projektin toteutuksesta.

Projektikokonaisuus, jossa opinnäytetyöni on osana, on myös esitetty liitteessä 3. Projektikokonaisuuden osalta opinnäytetyöhöni kuuluu kuvan oikea puoli.

5.3.2 Kirjautuminen

Linkedinin vaatima OAuth 2.0-kirjautuminen tehtiin, käyttäen Linkedinistä saatua ohjelmistorajapinta-avainta (API_KEY) ja ohjelmistorajapinta tunnusta (API_SECRET), jotka saa rekisteröimällä sovelluksen Linkedinin ohjelmistokehityssivustolla. Sovelluksen tunnisteiden ja käyttäjän kirjautumisesta saadun tunnistekoodin avulla muodostetaan lopullinen tunniste (AccessToken), jota käytetään Linkedinin tunnistautumisessa. Linkedinin kirjautuminen on tehty login.php-nimiseen tiedostoon. Kirjautu-

minen asetettiin kysyttäväksi heti sivulle tullessa. Katkelma kirjautumisen koodista on esitelty kuvassa 32.

```
function getAuthorizationCode() {
    $params = array('response_type' => 'code',
                    'client_id' => API_KEY,
                    'scope' => SCOPE,
                    'state' => uniqid('', true),
                    'redirect_uri' => REDIRECT_URI,
                );

    // Autentikointi pyyntö
    $url = 'https://www.linkedin.com/uas/oauth2/authorization?' . http_build_query($params);

    // Tallennetaan sessioon, koska arvo tarvitaan vastauksen tunnistamiseen
    $_SESSION['state'] = $params['state'];

    // Annetaan käyttäjälle Linkedinin tunnistus formi
    header("Location: $url");
    exit;
}
```

Kuva 32. Kuvassa esitetään osa käyttäjän kirjautumiskyselyn koodista.

5.3.3 Hakuparametrin tarkistus ja valinta

Kirjautumisen jälkeen käyttöliittymän hakuriviltä lähetetty hakusana menee käsiteltäväksi openCorporates.php nimiselle PHP- tiedostolle, jossa hakusanaa käyttäen muodostetaan URL- kysely verkossa toimivalle OpenCorporates ohjelmistorajapinnalle. Kyselyn perusteella OpenCorporates:in ohjelmistorajapinta käsittelee sen ja lähettää JSON-muotoisen vastauksen hakutermin antamista vaihtoehdoista. Tämän jälkeen openCorporates.php moduuli suodattaa tuloksista ylimääräiset maat jättäen jäljelle Suomen, Ruotsin, Norjan ja Tanskan tulokset. Tulokset pyritään tämän jälkeen järjestelemään siten, että osakeyhtiöt olisivat ensimmäisenä järjestyksessä. Lopuksi openCorporates.php moduuli tulostaa käyttöliittymään valintalomakkeen, josta käyttäjä voi valita halutun tuloksen. Haluttu tulos siirtyy HTML:n GET-viestillä neljään ohjelmistomoduuliin, joista kaksi eli search.php ja linkedin.php kuuluu omaan toteutukseeni. Jos OpenCorporates:in verkkorajapinta palauttaa vain yhden tuloksen, joka aiheutuu yleensä tarkasta yrityksen nimestä, niin käyttäjälle ei esitetä vaihtoehtoja vaan openCorporates.php moduuli lähettää GET-viestin suoraan ohjelmistomoduuleille.

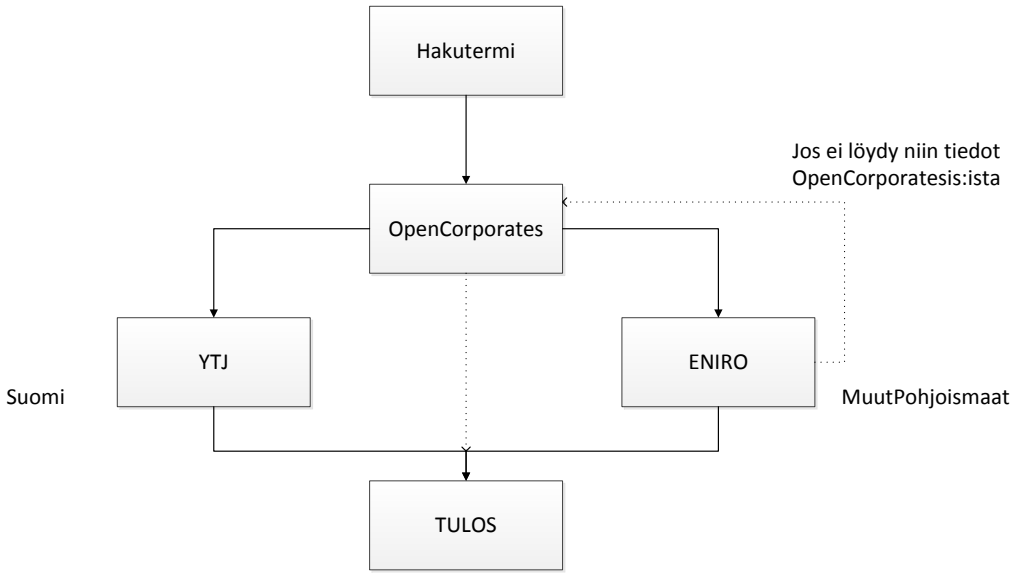
Kuvassa 33 on esitettynä pieni osa openCorporates.php-tiedoston koodista, jossa pyydetään hakusanaa vastaavat yritykset. Tulosten pyytämiseen OpenCorporates:in tietokannasta käytetään URL–muotoista kyselylausetta. OpenCorporates:lta saatu JSON–tiedosto puretaan PHP-ohjelmointikielen käyttämään muotoon. Tämän jälkeen käsitellään virhetila, jossa OpenCorporates:in palvelimeen ei saada yhteyttä.

```
// Pyydetään OpenCorporatesilta hakutuloksia
$response = @file_get_contents('http://api.opencorporates.com/v0.2/companies/search?q='
.$GET.'&per_page=100&format=json&page=1');
$responseDecoded =  json_decode($response,true);

// Jos yhteys OpenCorporatesiin ei toimi niin ilmitetaan siitä käyttäjälle.
if($response===false)
{
    echo utf8_encode("Yhteyden muodostaminen OpenCorporates tietokantaan ei onnistunut."
    ."Yritä hetken päästä uudestaan.");
    $error = 1;
}
```

Kuva 33. Kuvassa esitetään OpenCorporates API- kysely ja sen virhetarkastelu.

5.3.4 Yritystietohaku

Yritystietohaussa käytetään hyväksi openCorporates.php-tiedoston antamia session arvoja ja käyttäjän openCorporates.php-moduulin tulosteesta valitsemaa yritystä. Yritysten tietohaku on toteutettu kokonaisuudessaan search.php-moduulissa. openCorporates.php-ohjelmistomoduulista saatujen tietojen ja käyttäjän valitseman yrityksen perusteella valittu yritys luokitellaan maan perusteella. Yrityshaun toiminnallisuus on esitetty kuvassa 34.



Kuva 34. Kuvassa esitetään päivitetty yritystietohaku

Suomen rekisterissä olevat yritykset jatkavat koodissa maatunnuksen perusteella YTJ-hakuun, jossa haetaan PHP-komennolla ”file_get_content” OpenCorporates-ohjelmistorajapinnasta valitulle yritykselle saadun URL-rekisteriosoitteen perusteella koko verkkosivun HTML-hyperteksti. HTML-hypertekstistä otetaan tämän jälkeen halutut tiedot PHP:n DOM-tekniikkaa käyttäen. Lopuksi tiedot järjestetään ja tulostetaan käyttöliittymään. Suomen yritystietohaun toiminnallisuus DOM-tekniikan käyttöön asti on esitettynä kuvassa 35.

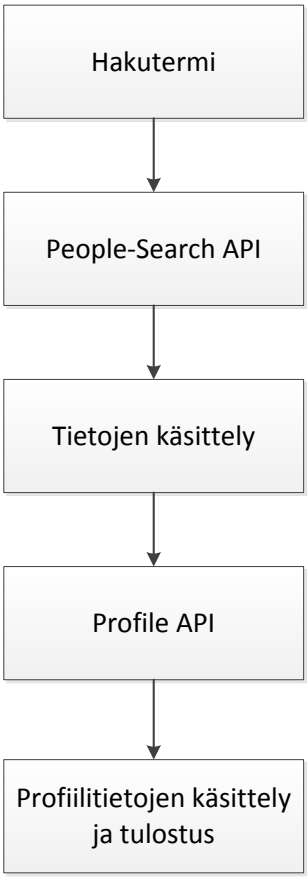
```php
// Haetaan YTJ:ltä haettavaa yritystä vastaava sivu
$result = file_get_contents($url);

// Luodaan DOM-olio html sivun käsittelyä varten
$dom = new DOMDocument();
@$dom->loadHTML($result);
$xpath = new DOMXPath($dom);

// Haetaan tuodusta sivusta tarvittavat materiaalit
$nameO = $xpath->evaluate("//span[@id='ContentPlaceHolder_lblToiminimi']");
$companyFormO = $xpath->evaluate("//span[@id='ContentPlaceHolder_lblYritysmuoto']");
$homeTownO = $xpath->evaluate("//span[@id='ContentPlaceHolder_lblYrityksenKotipaikka']");
$businessSectorO = $xpath->evaluate("//span[@id='ContentPlaceHolder_lblYrityksenToimiala']");
$streetAddressO = $xpath->evaluate("//span[@id='ContentPlaceHolder_lblPostiKatuOsoite']");
$localityO = $xpath->evaluate("//span[@id='ContentPlaceHolder_lblPostiToimipaikka']");
$yA = $xpath->evaluate("//span[@id='ContentPlaceHolder_lblytunnus']");
```

Kuva 35. Kuvassa on esitettynä katkelma Suomen yritystietohaun koodista.

Ruotsin, Norjan ja Tanskan tiedot puolestaan käyttävät Eniroa tietolähteenään. Eniro:lle lähetetään kysely, joka sisältää OpenCorporatesilta saadun halutun yrityksen yritystunnuksen ja ohjelmapuolen tunnistus tiedot. Eniro:n ohjelmistorajapinta palauttaa tämän jälkeen yhden JSON-tuloksen, josta otetaan PHP-ohjelmointikielen avulla tarvittavat tiedot. Tilanteessa, jossa tulosta ei löydy, pyydetään suppeammat perustiedot OpenCorporates:in ohjelmistorajapinnasta. Tämän jälkeen yrityksen perustiedot tulostetaan käyttöliittymään. Liitteessä 2. esitellään esimerkki maakohtaisista tuloksista.

5.3.5 Kontaktihaku

Kontaktihaku on tehty linkedin.php-nimiseen PHP-ohjelmistomoduuliin. Kontaktihaussa käyttäjän valitseman yrityksen perusteella tehdään kysely Linkedinin People-Search API -ohjelmistorajapinnalle, joka palauttaa ID-tunnukset hakutermiä vastaaville henkilöille. Näitä ID-tunnisteita käyttäen kysytään sitten enintään yhdeksän tulosta Linkedinin Profile API – ohjelmistorajapinnalta. Tämän jälkeen tiedot sisältävä JSON-tiedosto puretaan osiin ja tulokset tulostetaan käyttöliittymään PHP-ohjelmointikielen ”echo”-komennolla. Kontaktihaku kokonaisuus on esitettynä kuvassa 36 ja osa People-Search kyselyn koodista kuvassa 37.



Kuva 36. Kontaktihaun toiminta

```
// Haetaan yritykseen liittyviä henkilöitä
$url = 'https://api.linkedin.com/v1/people-search:'
.'(people:(api-standard-profile-request))?first-name=&last-name=&company-name='
.$keyword.'&format=json&oauth2_access_token='.$_SESSION['access_token'];

$response = json_decode(file_get_contents($url),true);
```

Kuva 37. Kuvassa on esitettynä Linkedinin ihmishaku.

6 LOPPUSANAT

Opinnäytetyö sujui melko hyvin. Sekä kontakti- että yritystietohaku onnistuivat mielestäni kohtalaisen hyvin huolimatta ilmaisvaihtoehtojen käytöstä opinnäytetyön toteutuksessa. Saavutin opinnäytetyön tavoitteet lukuun ottamatta yritystietohaun taloustietoja. Taloustietojen puute jäi vähän harmittamaan, koska se olisi antanut hienon lisäarvon yrityshaulle. Sen toteuttaminen ei kuitenkaan ollut mahdollista kohtuulliseen hintaan.

Linkedinin avulla tehty kontaktihaku onnistui aikalailla tavoitteiden mukaan. Omana tavoitteenani oli alun perin saada hakutuloksiin henkilön perustietojen lisäksi jonkinlaisia kontaktien ottamista helpottavia tietoja kuten puhelinnumeroita tai sähköpostiosoitteita. Tämä ei kuitenkaan ollut mahdollista suurimmassa osassa hakuja, koska muun muassa puhelinnumeron ja sähköpostiosoitteen saaminen olisi vaatinut, että käyttäjä olisi ollut omissa kontakteissaan enintään toisen asteen päässä omassa Linkedin verkossaan hakutuloksessa olevan henkilön kanssa. Eli Linkedinin osalta täytyi tyytyä lopulta kohteen kannalta pelkästään kuvaan, etu ja sukunimeen, sijaintiin ja kohdehenkilön asemaan yrityksessä. Lopputulokseen täytyy kuitenkin olla tyytyväinen Linkedinin rajoitusten ja hyvin vaativien käyttöehtojen takia.

Yritystietohaku onnistui myös ihan hyvin, vaikka sen toteuttaminen ei aina tuntunut edes mahdolliselta. Opinnäytetyön tutkimisvaihe keskittyy pääosin yritystietohaun toteutuksen suunnitteluun. Kun alkuperäiset vaihtoehdot Eniro ja Fonecta todettiin mahdottomiksi Suomen yritystietojen hankkimiseen, niin yritystietohaun tutkiminen meni viikoiksi jumiin. Lopulta OpenCorporates tuli vastaan ja sen ohjelmistorajapinta mahdollisti YTJ-tietohaun sisältämien yritystietojen hankkimisen. Ilman OpenCorporates:in ohjelmistorajapintaa yritysten perustietoja ei olisi ollut mahdollista saada ilmaiseksi tai edes kohtuulliseen hintaan. Myöhemmässä vaiheessa saimme tietoon, että YTJ:ltä olisi voinut saada myös kohtuullista maksua vastaan valmiita tietokantoja. Nykyisen automaattisesti päivittyvä OpenCorporates:n kautta toimiva järjestelmä on kuitenkin mielestäni parempi kuin oman tietokannan pitäminen ainakin taloudelliselta kannalta.

Suomen haun onnistumisen jälkeen päätin yrittää myös muiden pohjoismaiden yritysten tietojen saamista Eniron ja OpenCorporates:n avulla. Sain Ruotsin ja Norjan haun toimimaan kohtalaisesti, mutta Eniron heikot tiedot esimerkiksi monista Ruotsin yrityksistä ja epätäydellinen yhteensopivuus OpenCorporetes:n kanssa saivat välillä katumaan koko Eniron käyttöönottoa. Loppujen lopuksi sain myös Ruotsin ja Norjan yritysten perustiedot palautettua ohjelman kautta melko kelvollisina.

Työn aikana opin hyvin aiemmin vähän käyttämääni PHP-ohjelmointikieltä, XML- ja JSON-tiedostojen käsittelemistä ja muita ohjelmoinnissa käytettäviä tekniikoita. Myös laaja tutkimustyö oli ihan mielenkiintoinen kokemus, jossa tuli ilmi, että hyviä vaihtoehtoja ei aina ole montaakaan.

LÄHTEET

Albahari, J. & Albahari, B. : C# 5.0 IN A NUTSHELL
O'Reilly Media 2012, ISBN: 978-1-449-32010-2

Boese, E. : An Introduction to Programming with Java Applets.
Jones and Barlett Publishers 2010, ISBN: 978-0-7637-5460-0

developer.fonecta.net 2012. Finder API. Viitattu 29.4.2013.
http://developer.fonecta.net/Finder-API-doc

EBR 2010a. About EBR. Viitattu 30.7.2013.
http://www.ebr.org/section/2/index.html

EBR 2010b. Start using EBR. Viitattu 30.7.2013.
http://www.ebr.org/section/3/index.html

Hammer-Lahav, E. 2007. Introduction. Viitattu 23.3.2013.
https:// http://oauth.net/about/

Hardt, D. 2012. The OAuth 2.0 Authorization Framework. Viitattu 23.3.2013.
http://tools.ietf.org/html/rfc6749

History of XML, 2013. History of XML. Viitattu 15.3.2013.
http://www.totalxml.net/history-xml.php

Ho, D. 2011. Features. Viitattu 10.6.2013.
http://notepad-plus-plus.org/features.html

JSON.org. Introducing JSON. Viitattu 17.3.2013.
http://json.org/

Negrino, T. & Smith, D. : JavaScript–Tehokas hallinta
Gummerus Kirjapaino Oy 2007, ISBN: 978-952-5655-11-7

Linkedin a. Compare Account Types. Viitattu 13.3.2013.
http://help.linkedin.com/ci/fattach/get/1899813/0/filename/Compare_Account_Types.pdf

Linkedin b. Rest. Viitattu 13.3.2013.
http://developer.linkedin.com/rest

Linkedin c. Authentication. Viitattu 10.4.2013
http://developer.linkedin.com/documents/authentication

Linkedin d. People Search API. Viitattu 02.4.2013.
http://developer.linkedin.com/documents/people-search-api

Linkedin e. LinkedIn APIs Terms of Use. Viitattu 02.4.2013.
http://developer.linkedin.com/documents/linkedin-apis-terms-use

Linkedin f. Profile API. Viitattu 02.4.2013.
http://developer.linkedin.com/documents/profile-api

OpenCorporates a. Viitattu 14.4.2013.
http://opencorporates.com/info/about

OpenCorporates b. Viitattu 22.7.2013.
http://opencorporates.com/

OpenCorporates c. Legal/Licence info. Viitattu 02.4.2013.
http://opencorporates.com/info/licence

OpenCorporates d. API Reference :: OpenCorporates API. Viitattu 02.4.2013.
http://api.opencorporates.com/documentation/API-Reference

php.net 2013. History of PHP. Viitattu 4.6.2013.
http://www.php.net/manual/en/history.php.php

Powers, D. : PHP Solutions: Dynamic Web Design Made Easy.
2010, ISBN: 978-1-4302-3250-6

Seidler, K. 2013. XAMPP. Viitattu 11.6.2013.
http://www.apachefriends.org/en/xampp.html

Sheppy, 2013. Introduction to HTML. Viitattu 23.5.2013.
https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Introduction

Solis, D. : Illustrated C# 2012.
2012, ISBN: 978-1-4302-4279-6

Tatroe, K. , MacIntyre, P. & Lerdorf, R. : Programming PHP
O'Reilly Media 2013, ISBN: 978-1-4493-9277-2

URL Encoding. HTML URL Encoding Reference. Viitattu 21.5.2013.
http://w3schools.com/tags/ref_urlencode.asp

Vesterholm, M. & Kyppö, J. : Java-ohjelmointi
Talentum 2006, ISBN: 952-14-1025-6

Wikipedia 2013a. HTML. Viitattu 23.5.2013.
https://en.wikipedia.org/wiki/HTML

Wikipedia 2013b. JSON. Viitattu 17.3.2013.
http://en.wikipedia.org/wiki/JSON

Wikipedia 2013c. Query string. Viitattu 21.5.2013.
http://en.wikipedia.org/wiki/Query_string

Wikipedia 2013d. C Sharp (programming language). Viitattu 30.5.2013.
http://en.wikipedia.org/wiki/C_Sharp_%28programming_language%29

Wikipedia 2013e. Java. Viitattu 3.6.2013
http://fi.wikipedia.org/wiki/Java

Wikipedia 2013f. Notepad++. Viitattu 10.6.2013.
http://en.wikipedia.org/wiki/Notepad%2B%2B

Wikipedia 2013g. Linkedin. Viitattu 13.3.2013.
http://en.wikipedia.org/wiki/LinkedIn

Wikipedia 2013h. Linkedin. Viitattu 13.3.2013.
http://fi.wikipedia.org/wiki/LinkedIn

Wikipedia 2013i. OpenCorporates. Viitattu 14.4.2013.
http://en.wikipedia.org/wiki/OpenCorporates

Wikipedia 2013j. Eniro. Viitattu 27.4.2013.
http://fi.wikipedia.org/wiki/Eniro

Wikipedia 2013k. Fonecta. Viitattu 28.4.2013.
http://fi.wikipedia.org/wiki/Fonecta

Willison, S. 2006. A re-introduction to JavaScript. Viitattu 24.5.2013.
https://developer.mozilla.org/en-US/docs/JavaScript/A_re-introduction_to_JavaScript

XML 2012. Extensible Markup Language (XML). Viitattu 15.3.2013.
http://w3.org/XML/

YTJ, Mikä on YTJ. Viitattu 22.7.2013
http://ytj.fi/mika-on-ytj

ZoomInfo a. ZoomInfo Community Edition FAQ. Viitattu 6.5.2013.
http://www.zoominfo.com/business/products/zoominfo-ce/ce-faq

ZoomInfo b. ZoomInfo Pro. Viitattu 6.5.2013.
http://www.zoominfo.com/business/products/zoominfo-pro

ZoomInfo c. ZoomInfo Data Services. Viitattu 6.5.2013.
http://www.zoominfo.com/business/products/data-services

ZoomInfo d. ZoomInfo New Partner API Documentation. Viitattu 7.5.2013.
http://www.zoominfo.com/business/zoominfo-new-api-documentation

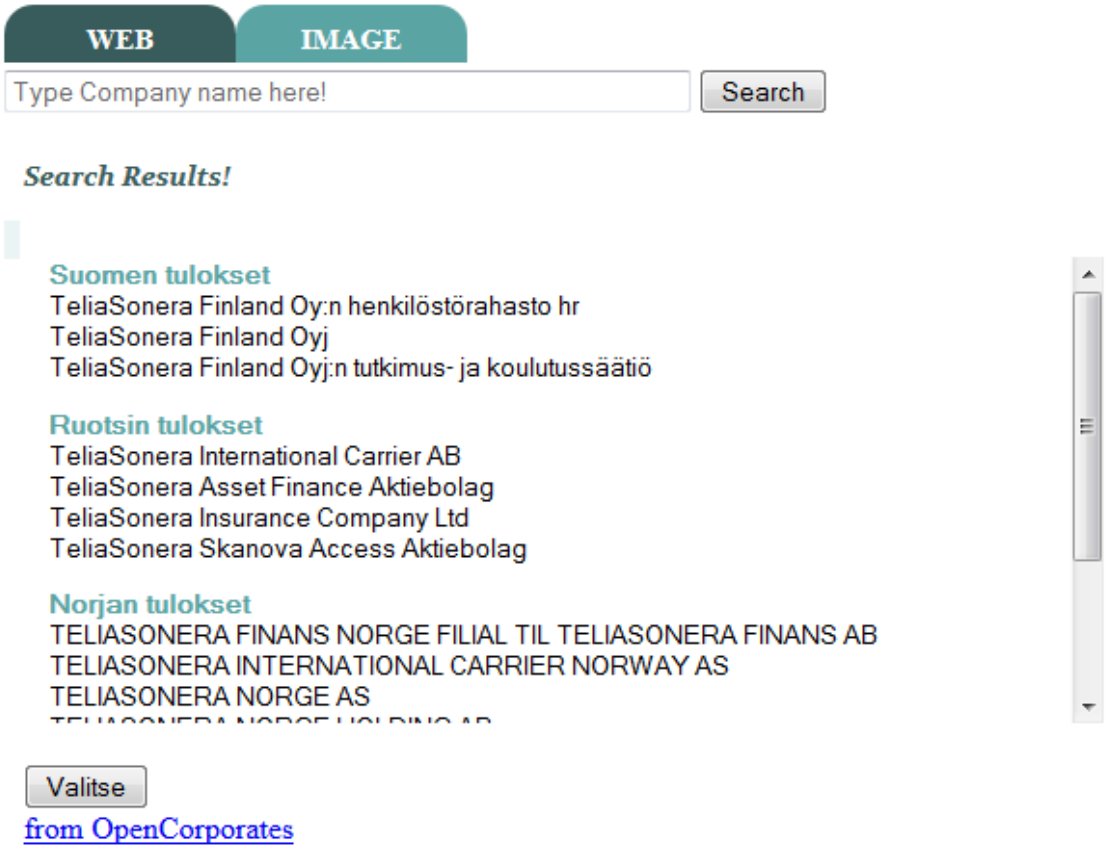
Liite 1

TUETUT OHJELMOINTIKIELET JA TEKNIIKAT NOTEPAD++-OHJELMASSA

Ada, asp, Assembly, autoIt
Batch
C, C++, C#, Caml, Cmake, COBOL, CSS
D, Diff
Flash ActionScript, Fortran
Gui4CLI , Go
Haskell, HTML
InnoSetup
Java, Javascript, JSP
KiXtart
LISP, Lua
Makefile, Matlab, MS-DOS, INI file
NSIS, Normal Text File
Objective-C
Pascal, Perl, PHP, PostScript, PowerShell, Properties file, Python
R, Resource file, Ruby
Shell, Scheme, Smalltalk, SQL
TCL, TeX
Visual Basic, VHDL, Verilog
XML
YAML

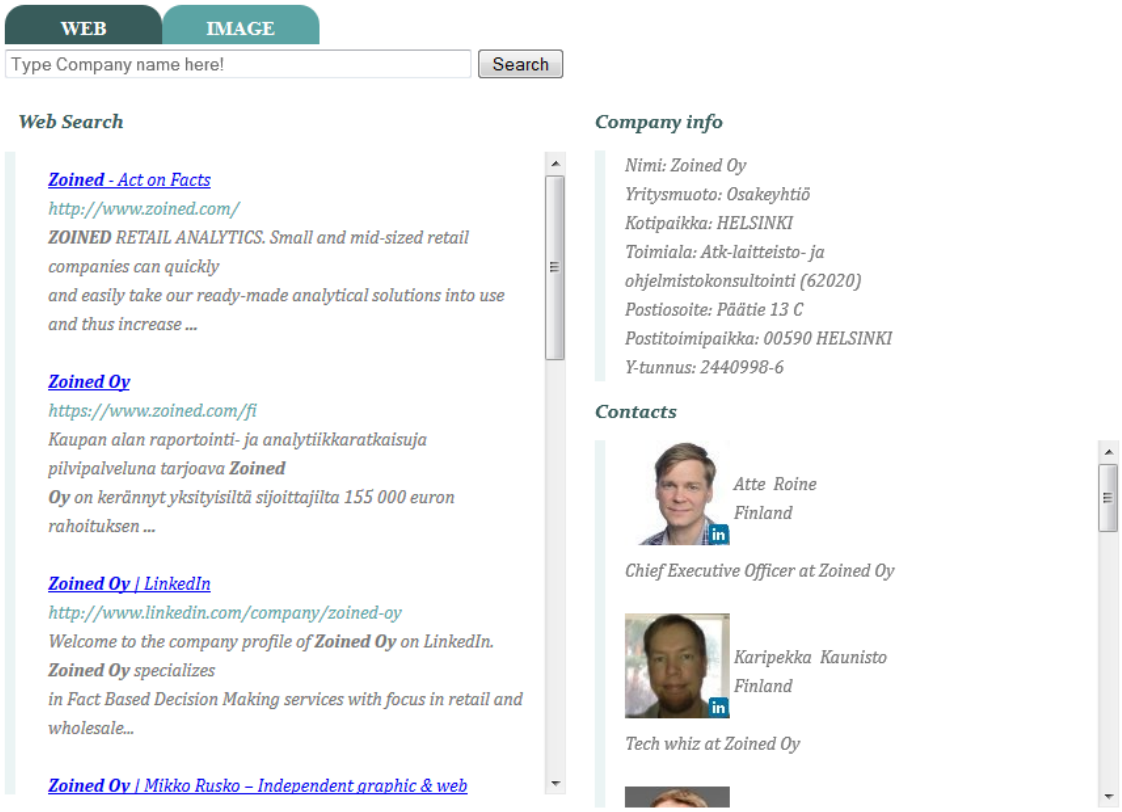
Liite 2

YRITYSTIETOHAUN MAAKOHTAISET TULOKSET



Liite 3

PROJEKTIKOKONAISUUS



Liite 4

LOGIN.PHP OSA1

```php
<?php
// Tässä tiedostossa hoidetaan Linkedinin vaatima kirjautumisrutiini

// Asetetaan Linkedinin ohjelmistorajapinta avain, tunnistesalaisuus,
// palautusosoite ja pyydettävien tietojen laajuus.
define('API_KEY',      'xxxxxxxxxxxx');
define('API_SECRET',   'xxxxxxxxxxxx');
define('REDIRECT_URI', 'http://localhost:8080/versio2/indexnewcss.php');
define('SCOPE',        'r_network r_basicprofile r_emailaddress');



// Linkedinin tunnistautuminen

// OAuth 2 Control Flow
if (isset($_GET['error'])) {
    // LinkedIn returned an error
    print $_GET['error'] . ': ' . $_GET['error_description'];
    exit;
} elseif (isset($_GET['code'])) {
    // Käyttäjä hyväksyy sovelluksen
    if ($_SESSION['state'] == $_GET['state']) {
        // Pyydetään AccessToken rajapintapyyntöjä varten
        getAccessToken();
    } else {
        exit;
    }
} else {
    if ((empty($_SESSION['expires_at'])) || (time() > $_SESSION['expires_at'])) {
        // Tunniste on vanhentunut, joten se poistetaan
        $_SESSION = array();
    }
    if (empty($_SESSION['access_token'])) {
        // Aloitetaan tunnistautumisprosessi
        getAuthorizationCode();
    }


}
function getAuthorizationCode() {
    $params = array('response_type' => 'code',
                    'client_id' => API_KEY,
                    'scope' => SCOPE,
                    'state' => uniqid('', true),
                    'redirect_uri' => REDIRECT_URI,
              );
```

Liite 5

LOGIN.PHP OSA2

```
    // Autentikointi pyyntö
    $url = 'https://www.linkedin.com/uas/oauth2/authorization?' . http_build_query($params);

    // Tallennetaan sessioon, koska arvo tarvitaan vastauksen tunnistamiseen
    $_SESSION['state'] = $params['state'];

    // Annetaan käyttäjälle Linkedinin tunnistus formi
    header("Location: $url");
    exit;
}

function getAccessToken() {
    $params = array('grant_type' => 'authorization_code',
                    'client_id' => API_KEY,
                    'client_secret' => API_SECRET,
                    'code' => $_GET['code'],
                    'redirect_uri' => REDIRECT_URI,
                );

    // Luodaan url-kysely tunnistus avaimen (Access Token) pyytämistä varten
    $url = 'https://www.linkedin.com/uas/oauth2/accessToken?' . http_build_query($params);

    // Kerrotaan seuraavalle file_get_contents-metodille että käytetään POST-metodia
    $context = stream_context_create(
                    array('http' =>
                        array('method' => 'POST',
                        )
                    )
                );

    // Pyydetään tunnistus avainta (Access Token)
    $token = json_decode(file_get_contents($url, false, $context));



    // Varastoidaan tunnistuavain ja sen vanhenemisarvot sessioon
    $_SESSION['access_token'] = $token->access_token;
    $_SESSION['expires_in']   = $token->expires_in;
    $_SESSION['expires_at']   = time() + $_SESSION['expires_in'];

    return true;
}

?>
```

Liite 6

OPENCORPORATES.PHP OSA1

```php
<?php

// Tässä tiedostossa käsitellään käyttäjän kirjoittama hakusana, jonka perusteella palautetaan käyttäjäll
// lista hakusanaa vastaavista yrityksistä.

// Tuodaan käyttäjän hakutermi
// Hakutermin oikeellisuus varmentuu OpenCorporates:in kautta, joten sitä ei tarkisteta tässä koodiosassa
$GET = $_GET['search-term'];
$error = 0;

if($GET != null)
{
$count = 0;
$responsePages = array();
$openData = array();
$openData2 = array();
$fiData = array();
$seData = array();
$noData = array();
$dkData = array();
$fiCount = 0;
$seCount = 0;
$noCount = 0;
$dkCount = 0;
$arrCount = 0;

// Pyydetään OpenCorporatesilta hakutuloksia
$response = @file_get_contents('http://api.opencorporates.com/v0.2/companies/search?q='.
$GET.'&per_page=100&format=json&page=1');
$responseDecoded =  json_decode($response,true);

if($response===false) // Jos yhteys OpenCorporatesiin ei toimi niin ilmitetaan siitä käyttäjälle.
{
    echo utf8_encode("Yhteyden muodostaminen OpenCorporates tietokantaan ei onnistunut.".
    "Yritä hetken päästä uudestaan.");
    $error = 1;
}

array_push($responsePages,$responseDecoded);

$totalPages = $responseDecoded['results']['total_pages'];
$totalCount = $responseDecoded['results']['total_count'];
$page = 1;
```

Liite 7

OPENCORPORATES.PHP OSA2

```
// Jos tuloksia on yli sata niin täytyy lähettää uusia pyyntöjä.
for($i=2;$i<=$totalPages;$i++)
{
array_push($responsePages,json_decode(file_get_contents('http://api.opencorporates.com/v0.2/companies/search?q='.
$GET.'&per_page=100&format=json&page='.$i),true));
}

// Valitaan listalle Suomen, Ruotsin, Norjan ja Tanskan tulokset
for($j=0;$j<$totalPages;$j++)
{
    if($totalCount>=100)
    {
        $resultInPage=100;
    }
    else
    {
        $resultInPage = $totalCount;
    }

    for($i=0;$i<$resultInPage;$i++)
    {

        if($responsePages[$j]['results']['companies'][$i]['company']['jurisdiction_code']=='fi'||$responseDecoded['results'].
        ['companies'][$i]['company']['jurisdiction_code']=='se'||$responseDecoded['results']['companies'][$i]['company'].
        ['jurisdiction_code']=='no'||$responseDecoded['results']['companies'][$i]['company']['jurisdiction_code']=='dk'  )
        {

            $obj = new stdClass;
            $obj->companyName = $responsePages[$j]['results']['companies'][$i]['company']['name'];
            $obj->jurisdiction_code = $responsePages[$j]['results']['companies'][$i]['company']['jurisdiction_code'];
            $obj->registry_url = $responsePages[$j]['results']['companies'][$i]['company']['registry_url'];
            $obj->company_number = $responsePages[$j]['results']['companies'][$i]['company']['company_number'];
            $count++;

            // Asetetaan osakeyhtiöt etusialle tuloksissa
            if(strpos($obj->companyName,'Oy')==true||strpos($obj->companyName,'AB')==true||strpos($obj->companyName,'AS').
            ==true||strpos($obj->companyName,'Oyj')==true||strpos($obj->companyName,'ASA')==true)
            {
                array_push($openData,$obj);
            }
            else
            {
                array_push($openData2,$obj);
                $arrCount++;
            }
```

Liite 8

OPENCORPORATES.PHP OSA3

```
        }
        $totalCount--;
    }

}


for($i=0;$i<$arrCount;$i++)
{
    array_push($openData,$openData2[$i]);
}



// Lajittelee tulokset maakohtaisesti

    for($i=0;$i<$count;$i++)
    {
        if($openData[$i]->jurisdiction_code=='fi')
        {
            array_push($fiData,$openData[$i]);
            $fiCount++;
        }

        if($openData[$i]->jurisdiction_code=='se')
        {
            array_push($seData,$openData[$i]);
            $seCount++;
        }

        if($openData[$i]->jurisdiction_code=='no')
        {
            array_push($noData ,$openData[$i]);
            $noCount++;
        }

        if($openData[$i]->jurisdiction_code=='dk')
        {
            array_push($dkData,$openData[$i]);
            $dkCount++;
        }
    }
```

Liite 9

OPENCORPORATES.PHP OSA 4

```
// Tallennetaan data search.php moduulia varten
$_SESSION['data'] = $openData;
$_SESSION['count'] = $count;

// Tarkistetaan onko tuloksia vähintään yksi vai enemmän
if($count!=0)
{
if($count!=1)
{

// Tulostetaan tulokset sisältävä lista käyttöliittymään
echo '<form name="listForm" action="indexnewcss.php" method="get"><br><select ondblclick="document.listForm.submit();"'.
'style="margin-left:10px; border:none; width:550px;" name="listbox" size="15">';

    if($fiCount!=0)
    {

        echo '<option value="" disabled style="color: #5ba4a4; font-weight:bold;">Suomen tulokset</option>';

        for($i=0;$i<$fiCount;$i++)
        {
            echo "<option value='".$fiData[$i]->companyName."'>".$fiData[$i]->companyName."</option>";
        }
    }

    if($seCount!=0)
    {
        echo '<option value="" disabled ></option>';
        echo '<option value="" disabled style="color: #5ba4a4; font-weight:bold;">Ruotsin tulokset</option>';

        for($i=0;$i<$seCount;$i++)
        {
            echo "<option value='".$seData[$i]->companyName."'>".$seData[$i]->companyName."</option>";
        }
    }
    if($noCount!=0)
    {
        echo '<option value="" disabled></option>';
        echo '<option value="" disabled style="color: #5ba4a4; font-weight:bold;">Norjan tulokset</option>';

        for($i=0;$i<$noCount;$i++)
        {
            echo "<option value='".$noData[$i]->companyName."'>".$noData[$i]->companyName."</option>";
        }
    }

    if($dkCount!=0)
    {
        echo '<option value="" disabled></option>';
        echo '<option value="" disabled style="color: #5ba4a4; font-weight:bold;">Tanskan tulokset</option>';

        for($i=0;$i<$dkCount;$i++)
        {
            echo "<option value='".$dkData[$i]->companyName."'>".$dkData[$i]->companyName."</option>";
        }
    }

echo "</select><br><br><input type='submit' value='Valitse' /></form>";

// Tulostetaan OpenCorporatesin logo
echo '<a href="http://www.opencorporates.com/">from OpenCorporates</a>';
}
else    // Jos tuloksia on vain yksi niin ohitetaan valinta lista. Tulokset lähetetään suoraan eteenpäin.
{
    $_GET['listbox'] = $openData[0]->companyName;
    header('Location: indexnewcss.php?listbox='.$_GET['listbox']);
}

}
else
{
if($error==0) echo "Ei hakutuloksia";
}
}
?>
```

Liite 10

SEARCH.PHP OSA 1

```php
<?php
// Tässä tiedostossa haetaan yrityksiin liittyvät perustiedot, jotka tulostetaan käyttöliittymään.

$GET  = $_GET['listbox'];

// Asetetan Eniro Api:n Käyttäjänimi ja avain
$profilename = "xxxxxxxx";
$profilekey = "xxxxxxxxxxxxxxxx";

if($GET!="")
{
$runFix = 0;
$openData = $_SESSION['data'];
$count = $_SESSION['count'];

// Käytetään kun haetaan tuloksia Suomen alueelta
function searchFinland ($url)
{
// Haetaan YTJ:ltä haettavaa yritystä vastaava sivu
$result = file_get_contents($url);

// Luodaan DOM-olio html sivun käsittelyä varten
$dom = new DOMDocument();
@$dom->loadHTML($result);
$xpath = new DOMXPath($dom);

// Haetaan tuodusta sivusta tarvittavat materiaalit
$nameO = $xpath->evaluate("//span[@id='ContentPlaceHolder_lblToiminimi']");
$companyFormO = $xpath->evaluate("//span[@id='ContentPlaceHolder_lblYritysmuoto']");
$homeTownO = $xpath->evaluate("//span[@id='ContentPlaceHolder_lblYrityksenKotipaikka']");
$businessSectorO = $xpath->evaluate("//span[@id='ContentPlaceHolder_lblYrityksenToimiala']");
$streetAddressO = $xpath->evaluate("//span[@id='ContentPlaceHolder_lblPostiKatuOsoite']");
$localityO = $xpath->evaluate("//span[@id='ContentPlaceHolder_lblPostiToimipaikka']");
$yA = $xpath->evaluate("//span[@id='ContentPlaceHolder_lblytunnus']");

// Käsitellään tulokset -- O = original, M = modified
foreach ($nameO as $nameM) { $name = $nameM ->nodeValue;}
foreach ($companyFormO as $companyFormM) { $companyForm = $companyFormM->nodeValue;}
foreach ($homeTownO as $homeTownM) { $homeTown = $homeTownM->nodeValue;}
foreach ($businessSectorO as $businessSectorM) { $businessSector = $businessSectorM->nodeValue;}
foreach ($streetAddressO as $streetAddressM) { $streetAddress= $streetAddressM->nodeValue;}
foreach ($localityO as $localityM) { $locality = $localityM->nodeValue;}
foreach ($yA as $yK) { $y = $yK->nodeValue;} // Y-tunnus
```

Liite 11

SEARCH.PHP OSA 2

```
// Tulostetaan tiedot käyttöliittymään
echo "Nimi: ".$name."<br> Yritysmuoto: ".$companyForm."<br> Kotipaikka: ".$homeTown."<br>Toimiala: ".
$businessSector."<br> Postiosoite: ".$streetAddress."<br> Postitoimipaikka: ".$locality."<br> Y-tunnus: ".$y;
}

// Käytetään Ruotsin, Norjan ja Tanskan tulosten kanssa. (Tanskan tuloksia ei tekohetkellä ollut vielä saatavilla)
function searchSeNoDk($company_number,$country,$company_name)
{
    // Kutsutaan globaaleja muuttujia
    GLOBAL $profilename;
    GLOBAL $profilekey;
    // Haetaan Enirolta tulokset käyttäen yritystunnusta ja puretaan json-koodaus
    $url = 'http://api.eniro.com/cs/search/basic?to_list=1&from_list=1&profile='.$profilename.'&key='.
    $profilekey.'&country='.$country.'&version=1.1.3&search_word='.$company_number;
    //var_dump($url);
    $response = json_decode(file_get_contents($url), true);

    // Käsitellään saaty JSON-tiedosto
    if($response['totalCount']!=0)
    {

    $companyName = $response['adverts'][0]['companyInfo']['companyName'];
    $streetName = $response['adverts'][0]['address']['streetName'];
    $postCode = $response['adverts'][0]['address']['postCode'];
    $postArea = $response['adverts'][0]['address']['postArea'];

    $phoneNumber = $response['adverts'][0]['phoneNumbers'][0]['phoneNumber'];

    // Tostetaan yrityksen perustiedot ja Eniron logo käyttöliittymään.
    echo utf8_encode("Yrityksen nimi: ".$companyName ."<br>Katuosoite: ".$streetName."<br>Postikoodi: ".
    $postCode."<br>Paikkakunta: ".$postArea."<br>Puhelinnumero: ".$phoneNumber);
    echo '<br><img src="http://api.eniro.com/static/img/eniro.org.logo.png" alt="Eniro logo" height="15"';

    }
    else
    {
        // Jos yritystunnusta ei löydy Eniron tietokannasta niin haetaan OpenCorporatesin tiedot yrityksestä

        $check = json_decode(file_get_contents('http://api.opencorporates.com/v0.2/companies/'.$country.'/'.
        $company_number));

        $companyType = $check->results->company->company_type;
        $address = $check->results->company->registered_address_in_full;
        $regURL = $check->results->company->registry_url;
```

Liite 12

SEARCH.PHP OSA 3

```
        $regURL = $check->results->company->registry_url;

        if($regURL!='')
        {
            $stringBuilder = '<br>Lisätietoa osoitteesta:<br><a href="'.$regURL.'">'.$regURL.'</a><br><br>from OpenCorporates';
        }
        else
        {
            $stringBuilder = '<br><br>from OpenCorporates';
        }
        // Tulostetaan tiedot käyttöliittymään
        if($address!='')
        {
            echo utf8_encode('Nimi: '.$company_name.' <br>Yritysnumero: '.$company_number.' <br>Yritysmuoto: '.
            $companyType.' <br>Postiosoite: '.$address.$stringBuilder);
        }
        else
        {
            echo utf8_encode('Nimi: '.$company_name.' <br>Yritysnumero: '.$company_number.' <br>Yritysmuoto: '.
            $companyType.$stringBuilder);
        }
    }
}

// Tarkistetaan onko haluttu yritys Suomesta vai muualta
for($i=0;$i<$count;$i++)
{

    if($openData[$i]->companyName==$GET && $openData[$i]->jurisdiction_code=="fi" && $runFix==0)
    {
        $url = $openData[$i]->registry_url;
        searchFinland($url);
        $runFix = 1;
    }
    if($openData[$i]->companyName==$GET && $openData[$i]->jurisdiction_code!="fi"&& $runFix==0)
    {
        $company_number = $openData[$i]->company_number;
        $country = $openData[$i]->jurisdiction_code;
        $company_name = $openData[$i]->companyName;
        searchSeNoDk($company_number,$country,$company_name);
        $runFix = 1;
    }
}
}
?>
```

Liite 13

LINKEDIN.PHP OSA1

```php
<?php
// Tässä tidostossa haetaan yritykseen liittyvien henkilöiden tiedot Linkediniltä.
// Tämän jälken tiedot tulostetaan käyttäliittymään.
$GET  = $_GET['listbox'];

if($GET != null)
{
    searchPeople($GET);
}

function noResult()
{
    echo "Ei tuloksia Linkedinin tietokannasta.";
}
function searchPeople($GET) {

$keyword = urlencode($GET);
$information = array();
$numberCount = 0;

// Haetaan yritykseen liittyviä henkilöitä
$url = 'https://api.linkedin.com/v1/people-search:(people:(api-standard-profile-request))?first-name=&last-name=&company-name='.
$keyword.'&format=json&oauth2_access_token='.$_SESSION['access_token'];

$response = json_decode(file_get_contents($url),true);


$count = $response['people']['_total'];


if($count==0||$count==null) // Ilmoitus käyttäjälle jos Linkedinistä ei löydy tuloksia kyseiselle yritykselle
{
noResult();
}
else
{

$personList = array();

$countIndex = -1;

$persons = "";
```

Liite 14

LINKEDIN.PHP OSA 2

```
// Haetaan henkiöprofiillit
for($i=0;$i<$count;$i++)
{
    $numberCount++;
    if($numberCount <10) // Määritellään tulosten lukumäärä yhdeksään
    {
        $personList[$i] = str_replace('http://api.linkedin.com/v1/people/',"",$response['people']['values'][$i]['apiStandardProfileRequest']['url']);
        $countIndex++;
    }
}

// Tarkistetaan tulosten määrä ja laatu (onko tulos tyhjä)
$dataResultCount = 0;
for($i=0;$i<$countIndex+1;$i++)
{
    if($personList[$i]!=null){
    $persons = $persons.$personList[$i].",";
    $dataResultCount++;
    }
}
// korjaus, jos tuloksisssa on yksi tyhjä solu välissä
if($dataResultCount<3&&str_replace('http://api.linkedin.com/v1/people/',"",$response['people']['values'][3]['apiStandardProfileRequest']['url'])!="")
{
    $persons = $persons.str_replace('http://api.linkedin.com/v1/people/',"",$response['people']['values'][3]['apiStandardProfileRequest']['url']).",";
}

    $persons = substr_replace($persons ,"",-1);

// Haetaan henkilöiden tiedot
$url ='https://api.linkedin.com/v1/people::('.$persons.'):(first-name,last-name,headline,picture-url,location:(name))?format=json&oauth2_access_token='
.$_SESSION['access_token'];

// Käsitellään henkilöiden profiilitiedot
$response = json_decode(file_get_contents($url),true);

for($i=0;$i<$dataResultCount;$i++)
{
    if(isset($response['values'][$i]['pictureUrl']) && $dataResultCount>$i)
    {
        array_push($information,array($response['values'][$i]['firstName'],$response['values'][$i]['lastName'],$response['values'][$i]['headline'],
        $response['values'][$i]['location']['name'],$response['values'][$i]['pictureUrl']));
    }

    else if($dataResultCount>$i)
    {
        array_push($information,$information1 = array($response['values'][$i]['firstName'],$response['values'][$i]['lastName'],
        $response['values'][$i]['headline'],$response['values'][$i]['location']['name'],'http://static.licdn.com/scds/common/u/img/icon/icon_no_photo_60x60.png'));
    }
}



// Tulsotetaan kuva ja tiedot käyttöliittymään
for($i=0;$i<$dataResultCount;$i++)
{
    if($dataResultCount>$i){echo '<img align="left" src='.$information[$i][4].'> <br> '.$information[$i][0].'  '.
    $information[$i][1].'<br> '.$information[$i][3].'<br><br>'.$information[$i][2].'<br><br>'; }
}


}
}

?>
```

Liite 15

ENIRO COMPANY SEARCH - BASIC API OSA 1

General information

The API is accessed via standard HTTP GET requests. Query-string parameters are passed along with the appropriate URL in order to specify search parameters. To guarantee consistent encoding of the result set the URL should be escaped with standard URL-escaping techniques using UTF-8 as the encoding. For example, if searching for the keyword `räksmörgås`, the encoded parameter should be `r%C3%A4ksm%C3%B6rg%C3%A5s`.

The search service returns JSON responses, served as `application/json` encoded as UTF-8. A successful JSON response returns a `200` response code.

URL

`http://api.eniro.com/cs/search/basic`

Request parameters

profile
required

Specifies the name of the profile which you have registered to use for the Eniro API. You can find your profile and key in your account information.

Example values: `website-search.myprofile`, `mobile-search.myprofile`

key
required

Specifies the key for the profile which you have registered to use for the Eniro API. You can find your `profile` and `key` in your account information.

Example value: `31415926535897932384`6

country
required

Specifies the country you would like to search in. You may search in only one country at a time.

Acceptable values:

`se`	Sweden (eniro.se)
`dk`	Denmark (krak.dk)
`no`	Norway (gulesider.no)

version
required

Specifies the version of the API you are using.

Example value: `1.0.0`

seed
optional

When receiving search results with the same relevance level, their sort order will be based on the value of the seed parameter.

Example value: `1337`

search_word
optional

Specifies the search criteria, the keyword or keywords you would like to search for.

Example values: `advokat`, `hoteller`, `vvs`

geo_area
optional

Specifies the general location where you would like to search, usually a city or area.

Example values: `stockholm`, `københavn`, `oslo`

from_list
optional

Specifies the pagination start location, i.e. where in the result set you want to the results to start. The start of the result set is at location `1`. The default result set size is `25`. Note: to be used in conjuction with `to_list` to get expected results.

Example value: `1`

to_list
optional

Specifies the pagination end location, i.e. where in the result set you want to the results to end. The default result set size is `25`. Note that you can maximum get a response with 100 results. I.e. `to_list - from_list < 100`. Also note `to_list` is to be used in conjuction with `from_list` to get expected results.

Example value: `20`

eniro_id
optional

Specifies the Eniro ID of a particular company you want to search for.

Example value: `14921547`

callback
optional

Specifies the JSONP callback name. Example: `&callback=parseResponse` will result in JSONP code: `parseResponse( {...} );` where *{...}* is the JSON result object.
Note! If an error occurs the response will be something like: `parseResponse( {"error": "010"} );`

Example values: `parseResponse`, `object.displayList`, `obj%5B%22function-name%22%5D = obj["function-name"]`

Liite 16

ENIRO COMPANY SEARCH - BASIC API OSA 2

Response

The JSON response contains the following fields:


```
{
  "title": "Gula sidorna API",
  "query": "http://api.eniro.com/cs/search/basic?to_list=25&from_list=1&search_word=hotell&geo_area=&country=se&version=1.1.2",
  "totalHits": 4170,
  "totalCount": 4170,
  "startIndex": 0,
  "itemsPerPage": 25,
  "adverts": [
    {
      "eniroId": "14380005",
      "companyInfo": {},
      "address": {},
      "location": {},
      "phoneNumbers": [],
      "homepage": null,
      "facebook": null,
      "companyReview": null,
      "infoPageLink" : "http://www.gulesider.no/firma/eniro-norge-as:p10002859083?search_word=eniro"
    },

    ...

    {}
  ]
}
```


title string	The search result title **Example value:** "Gula sidorna API"
query string	The query URI for the current response **Example value:** "http://api.eniro.com/cs/search/basic?search_word=eniro&version=1.1.2&country=se"
totalHits number	The total number of results for this search **Example value:** 4711
totalCount number	The total number of resuts for this search **Example value:** 4711
startIndex number	The starting index for this search (pagination) **Example value:** 0

Liite 17

ENIRO COMPANY SEARCH - BASIC API OSA 3

itemsPerPage
number

The total number of results per response (page size for pagination)

Example value: 25

adverts
array

The search result set, an array of objects

```
"adverts": [
  {
    "eniroId": "14380005",
    "companyInfo": {},
    "address": {},
    "location": {},
    "phoneNumbers": []
  },
  ..
  {}
]
```

eniroId
string

Internal Id for the customer/local business unit in Eniro system(s).

Example value: "14380005"

companyInfo
object

Object containing information about the company

```
"companyInfo": {
  "companyName": "Clarion Hotel Sign",
  "orgNumber": "5564660107",
  "companyText": "Stockholms största hotell med 558 dubbelrum, konferenslokaler för upp till 700 personer,
spa,..."
}
```

companyName
string

The company name

Example value: "Clarion Hotel Sign"

orgNumber
string

The company's organization number

Example value: "5564660107"

companyText
string

Text about the company

Example value: "Stockholms största hotell med 558 dubbelrum, konferenslokaler för upp till 700 personer,

ENIRO COMPANY SEARCH - BASIC API OSA 4

address
object

Object containing information about the address to this company/office. E.g. the office visiting address.

```
"address": {
  "coName": "...",
  "streetName": "Ö. Järnvägsg. 35",
  "postCode": "101 26",
  "postArea": "STOCKHOLM",
  "postBox": "Box 310"
}
```

coName
string

The care of name. If this address object contains a `co_name`.

Example value: `"Eniro"`

streetName
string

The street address

Example value: `"Ö. Järnvägsg. 35"`

postCode
string

The post code

Example value: `"101 26"`

postArea
string

The post area

Example value: `"STOCKHOLM"`

postBox
string

The post box

Example value: `"Box 310"`

location
object

Object containing geolocation information

```
"location": {
  "coordinates": []
}
```

coordinates
array

Array containing objects with geolocation coordinates

```
"coordinates": [
  {
     "longitude": 18.054142487983395,
     "latitude": 59.33432236299209
  },
  {
     "use": "route",
     "longitude": 18.054142487983395,
     "latitude": 59.33432236299209
  }
]
```

longitude
number

The WGS-84 longitude coordinate value

Example value: `18.054142487983395`

latitude
number

The WGS-84 latitude coordinate value

Example value: `59.33432236299209`

use
string

Example value: `"route"`

ENIRO COMPANY SEARCH - BASIC API OSA 5

phoneNumbers
array

Array of objects containing phone numbers

```
"phoneNumbers": [
  {
    "type": "std",
    "phoneNumber": "08 - 676 98 00",
    "label": null
  }
]
```

type
string

The type of phone number this object represents

Values:

std	Standard
mob	Mobile
fax	Fax

phoneNumber
string

The phone number

Example value: "08 - 676 98 00"

label
string

The label for this phone number

Example value:

homepage
string

This field contains a URL which redirects to the given company's homepage. This URL goes through a proxy which enables us to track statistics

Example value: "http://api.eniro.com/proxy/homepage/{generated hash}"

facebook
string

This field contains a URL which redirects to the given company's facebook page. This URL goes through a proxy which enables us to track statistics

Example value: "http://api.eniro.com/proxy/facebook/{generated hash}"

companyReviews
object

A link to the company's review page (rejta.se, Anbefalt.no, detHitter.dk).

Example value: http://www.rejta.se/omdome/270601

infoPageLink
string

Link to infopage on eniro.se, krak.dk or gulesider.no

Example value: "http://www.gulesider.no/firma/eniro-norge-as:p10002859083?search_word=eniro"