INSTALLATION MANUAL

VJF version 6.0

Abstract

This document includes a detailed guide about how to setup and install the Virtual Job Fair system as of version 6.0. It will guide and prepare you for any problems you may encounter. It should be read before starting to work on the project.

Table of Contents

Installation

This chapter will explain how to install the project and setup the project before you begin working. Please follow this guide carefully.

Software Needed

You would need the following software no matter what is your OS:

1. An application server
    a. Apache (Recommended)
    b. Apache Tomcat
2. An IDE supporting PHP
    a. NetBeans (Recommended)
    b. PHPStorm
3. MySQL server
4. A subversion software
    a. SmartGit (Recommended)
    b. GitHub
5. Selenium IDE
    a. Plug in for Mozilla (Recommended)
    b. Plug in for Chrome

A faster and easier option is to download WAMP for Windows or XAMP for Mac and Linux versions. This will install the application server, the database server and the PHP configuration files.

GitHub

First clone the project from GitHub. This operation can be performed in Github.com whenever you create your repository for the semester.

Version 6.0 can be found at the following link:
https://github.com/FIU-SCIS-Senior-Projects/Virtual-Job-Fair.git
And can be cloned with the following link:
https://github.com/FIU-SCIS-Senior-Projects/Virtual-Job-Fair.git

You need to set up your local environment. This means that you will have a copy of your Git repository locally and changes would have to be synchronized. Start your Sub-version system (I strongly recommend SmartGit) and clone your project from the Git repository you clone it in the first place. That is, where you want to keep the files.

IDE

To install your IDE: if you chose WAMP/XAMP then you need to remember where you install it because you will need to create a **symbolic link (very important)** from the **www** folder inside WAMP (not really sure what is the folder for XAMP, but I believe is the same). The **symbolic link** needs to be named **JobFair** and point to the Website folder inside the **Code/Website** folder in the local Git repo folder.

Database and Website Setup

For the database: if you installed WAMP/XAMP, this was already made for you. Create a database with name **jobfairdb**. To get the latest data, I would suggest to go to the Production Server and make a dump of the tables and data at:

http://vjf.cis.fiu.edu/phpmyadmin/
or development:
http://vjf-dev.cis.fiu.edu/phpmyadmin/

Professor Masoud Sadjadi should give the credential to access both environments and the database server password. If you just want to get started there is an SQL file in the project Code folder.
Once have create the database and the application server (WAMP/XAMP) is running. Open your IDE and import the project from the Website folder inside your local Git repository. Go to the file: **JobFair/protected/config/main-back.php** and rename it to **main.php**. This file is one of the configuration files for the Yii framework and has the username and password to access your local database. Since all of the GitHub repositories are public and this file contain sensitive information, you may want to add it to your local Git-ignore file so SmartGit don't upload it to GitHub and I the long run will save you trouble.

Now, you can navigate to:
http://localhost/JobFair/index.php/site/login
and access the system.

I suggest to fully read all the documentation as they might be some functionalities not running in your environment and, that does not mean they are broken.

Selenium IDE

In this section we will cover how to install Selenium IDE plugin for Mozilla Firefox browser.

1. Open a browser and navigate to http://www.seleniumhq.org/download/
2. Download the Selenium IDE (currently version 2.9.0) which is the *.xpi and install it as a Firefox plugin.
3. Under Options->Options you can activate the developer tools.
4. An in the same window under Selenium IDE extensions, we can browse to were our JavaScript file is containing the data for the data driven tests. (Usually in the project folder there is a compressed file under ..\Code\Test\Student Actor\datasource.js)

Note that our GitHub repository us public and any sensitive data inside this file can be seen. What we are currently been doing is compressing the datasource.js file and putting a password on it. The password can be asked to your product owner.

Solr Server

Solr is a search server that provides a rich REST API and runs under Apache Lucene project to provide a fast way to search by indexing large amount of content.
Below are the necessary steps to get Solr Server 5.0 up and running, including what is necessary to configure and run it in a linux or windows environment.

Installation

Make sure your OS has a Java version 1.7 or higher. Independently from your OS, Solr can be downloaded from any of the links at:
http://www.apache.org/dyn/closer.cgi/lucene/solr/5.1.0
After downloading Solr, you will have to unzip the file and copy the folder to your project (Note that this is already done for you and have been uploaded to GitHub). This means, that whenever you clone your project the Solr server is already included.
This completes the installation of Solr.

Configuration

Solr consist of Cores. Each core has a different configuration and indexes the data depending of this configuration. In order to create a Core in Solr, 4 files must be manually created as followed:

1. Navigate to your project folder. This is the folder where you copied the solr folder extracted on previous topic.
2. Notice that in this folder you have a file named: solr.xml. This file in the main entry to all the cores in your server. It also specifies the port to access your Solr server request via HTTP. By default Solr uses port 8983, and it needs to be changed, this is the place to do it. See appendix for all the lines.
3. Now, navigate to ..\solr-5.0.0\server\solr and create a folder with the name of the core you want to create.
4. Inside the above created folder we must create a file to specify the name and properties of the core we are creating, named core.properties with the following lines.
    - #Written by Rogelio Alonso
    - #Wed Apr 17 2015 at 11:34 AM EDT
    - name= “the name of Core”

- config=solrconfig.xml
- schema=schema.xml
- dataDir=data

5. As you can see, the Solr Core will depend of the configuration file (solrconfig.xml), the schema file (schema.xml) and the data directory.
    - The solrconfig.xml is a configuration file and will tell the core what are the dependencies needed for the data handlers and the location of the respective jar files.
    - The schema.xml is another configuration file and will tell the core the types of data that will be indexed as well as what will be the indexing key, among much other stuff.
    - The data (which must be also manually created), is a directory where Solr will construct and maintain the search index.
6. So, now we must create three folders on the directory where you created the core.properties file.
    - conf
        i. This is where schema.xml and solrconfig.xml will be created.
    - data
        i. This is where Solr will build its index.
    - lib
        i. This is where any required library for the core will be copied.
7. Navigate to the conf folder (created above) and create the files named schema.xml and solrconfig.xml (a copy of these files can be made of the basic_config folder provided by Solr)
8. Now we need to update the schema.xml
    - Every filed you want to index with solr should be declared here with the name, type, if you want it index or stored. i.e. <field name="title" type="string" indexed="true" stored="true"/>
9. Congratulations you have just configured your Solr Core instance.

Starting and Stopping the Solr

Now, this depends on what OS you are using, so I will discuss about Linux and Windows, which are the ones I consider the most important ones.

Windows

1. Make sure you have added the "java" as an environmental variable. (If not please read this tutorial in how to do so: https://docs.oracle.com/javase/tutorial/essential/environment/paths.html)

2. Open a Command Prompt window and navigate to ..\solr-5.0.0\server
3. Type java –jar start.jar and press Enter
4. Open a browser and navigate to localhost:8983/solr to make sure your server is running. You can also select your core name from the core drop down menu and make sure is running with no problems.
5. To stop the server simply press Ctrl+C or close the Command Prompt.

Linux

Since most of linux is done from SSH terminal, you want to make sure that whenever you exit your terminal your server is still running. We will accomplish this with the help of "nohup". Moreover, we will need an alias for the process in case we want to stop it, it will be easier for us to identify it.

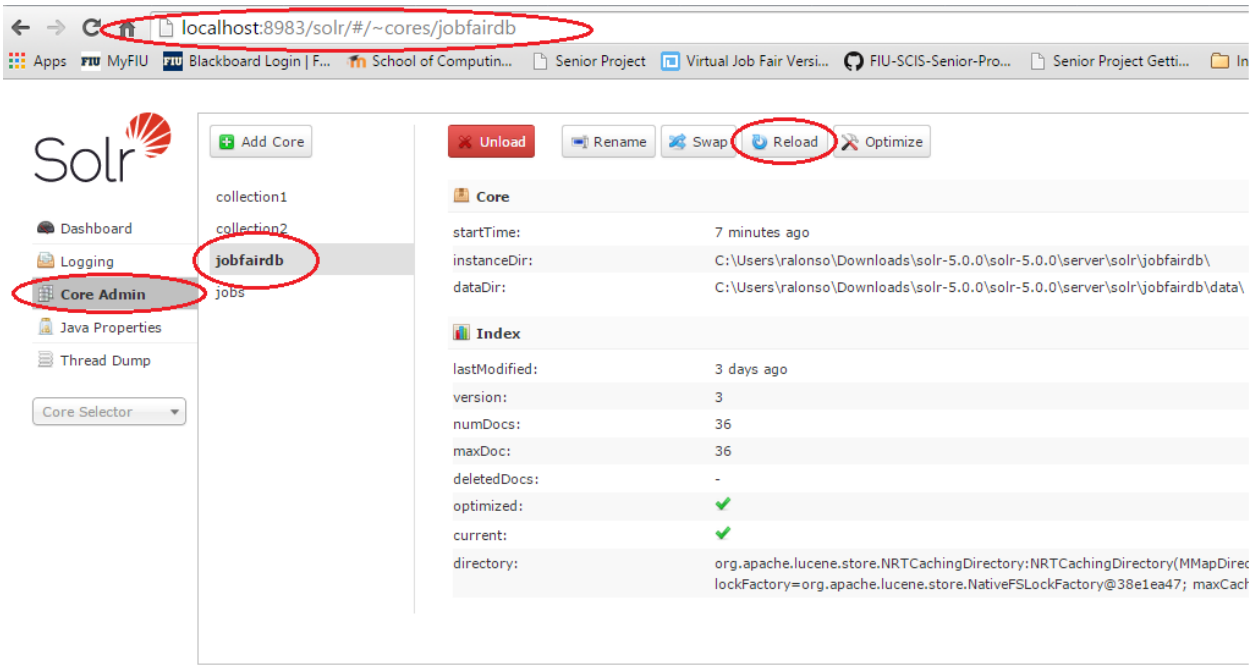
1. Navigate to ..\solr-5.0.0\server
2. Type sudo nohup java –Dsolr –jar start.jar & and press Enter
3. To stop the server you will need to look for the process and kill it with any of following commands:
    - sudo pkill -f Dsolr
    - sudo pkill -9 -f Dsolr
    - Or you can list the processes by jps –l and then kill $PID

Reloading Configurations for the Solr Server

Note that you do not need to start and stop the server every time you make an update in the schema or the configuration file. Solr provides a graphic way to manage your cores as follows:

1. Open a browser and navigate to http://localhost:8983/solr or your server i.e. http://vjf-dev.cis.fiu.edu:8983/solr
2. Click on Core Admin
3. Select your Core
4. Click on Reload

This will reload all the files needed and will restart the Core.



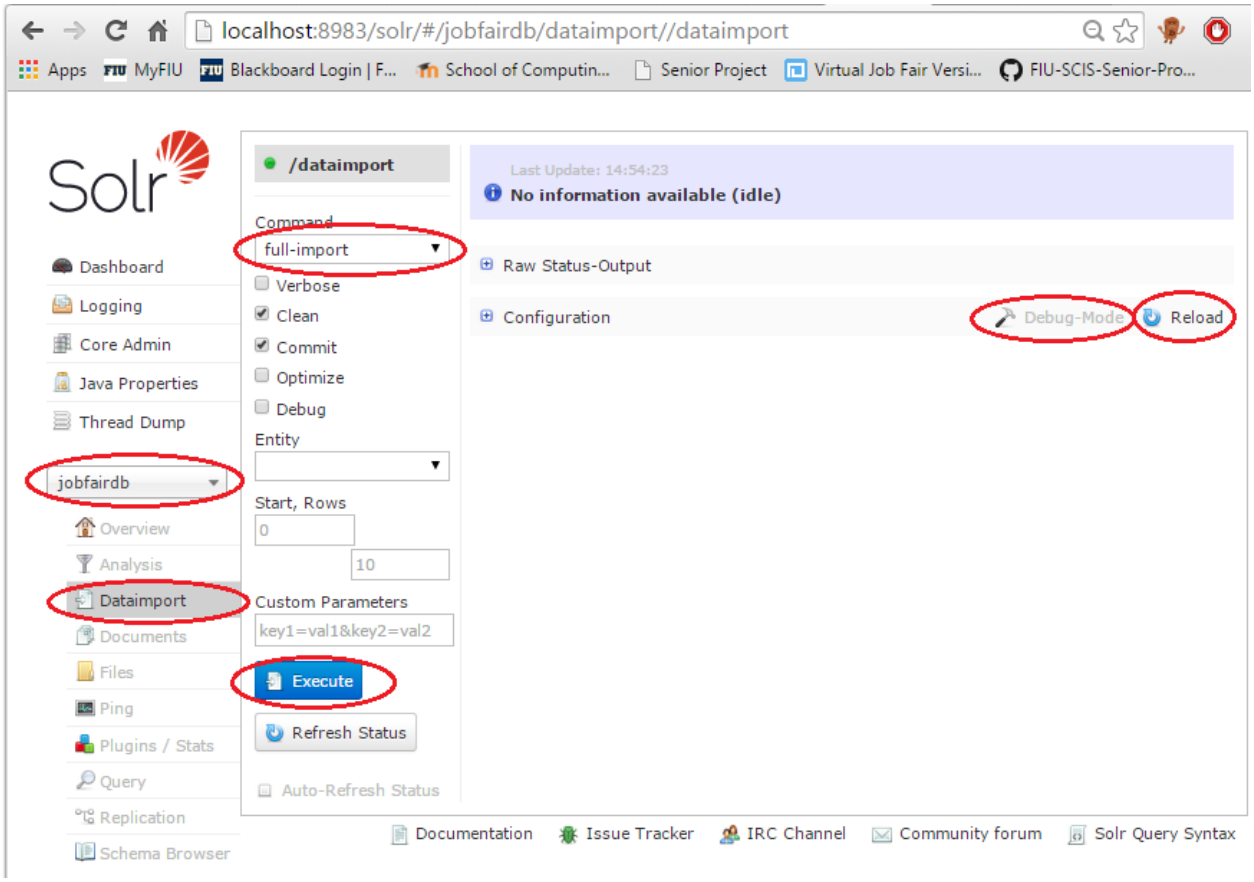
Importing Data from MySQL Database

Initially it is necessary to import data to be indexed. This data is the desired tables on the jobfairdb database in MySQL. I will now document how to integrate and import data from the MySQL database into Solr and how to query the data.

1. Download mysql-connector-java-*-bin.jar (version 5 and up) from Oracle web site: http://dev.mysql.com/downloads/connector/j/
2. Copy the appropriate mysql-connector-java-5.1.35-bin.jar in the lib folder created in the configuration process.
3. Copy the solr-dataimporthandler-5.0.0.jar and solr-dataimporthandler-extras-5.0.0.jar from the ..\solr-5.0.0\dist folder to the lib folder above mentioned. This two jar files will use the MySQL connector jar to access and retrieve rows from your MySQL database.
4. Add the following lines to the solrconfig.xml file:
    - Before the <dataDir></dataDir> tag add:
        - <!-- Dependency Libraries-->
        - <lib dir="**lib**" regex="**.*\.jar**" />
        - <lib dir="**lib**" regex="**solr-dataimporthandler-\d.*\.jar**" />
        - <lib dir="**lib**" regex="**mysql-connector-java-\d.*\.jar**" />
    - At the end of the file add:
        - <requestHandler name="**/dataimport**" class="**org.apache.solr.handler.dataimport.DataImportHandler**">

- <lst name="**defaults**">
- <str name="**config**">**data-config.xml**</str>
- </lst>
- </requestHandler>

5. Create a file name data-config.xml to save the queries to be run to MySQL Database for full and incremental (Delta) indexing as example in the Appendix.
    - To touch up upon this file. Here is where you want to map your MySQL Database fields to those defined on your schema.xml. This file will contain not only the information for the type of query you want run to your database, either Delta/Incremental or Full Import; but also, the link between data types and field names.
6. Open a browser and navigate to http://localhost:8983/solr or your server i.e. http://vjf-dev.cis.fiu.edu:8983/solr
7. Select your Core
8. Click on Dataimport
9. Select the full-import Command
10. Click Execute
11. After execution you can see how many documents or rows where imported into the index.



If you want to make changes to your schema.xml file you can do so on the Debug-Mode button and then reload the schema.xml file on the Reload button.

Indexing with Solr

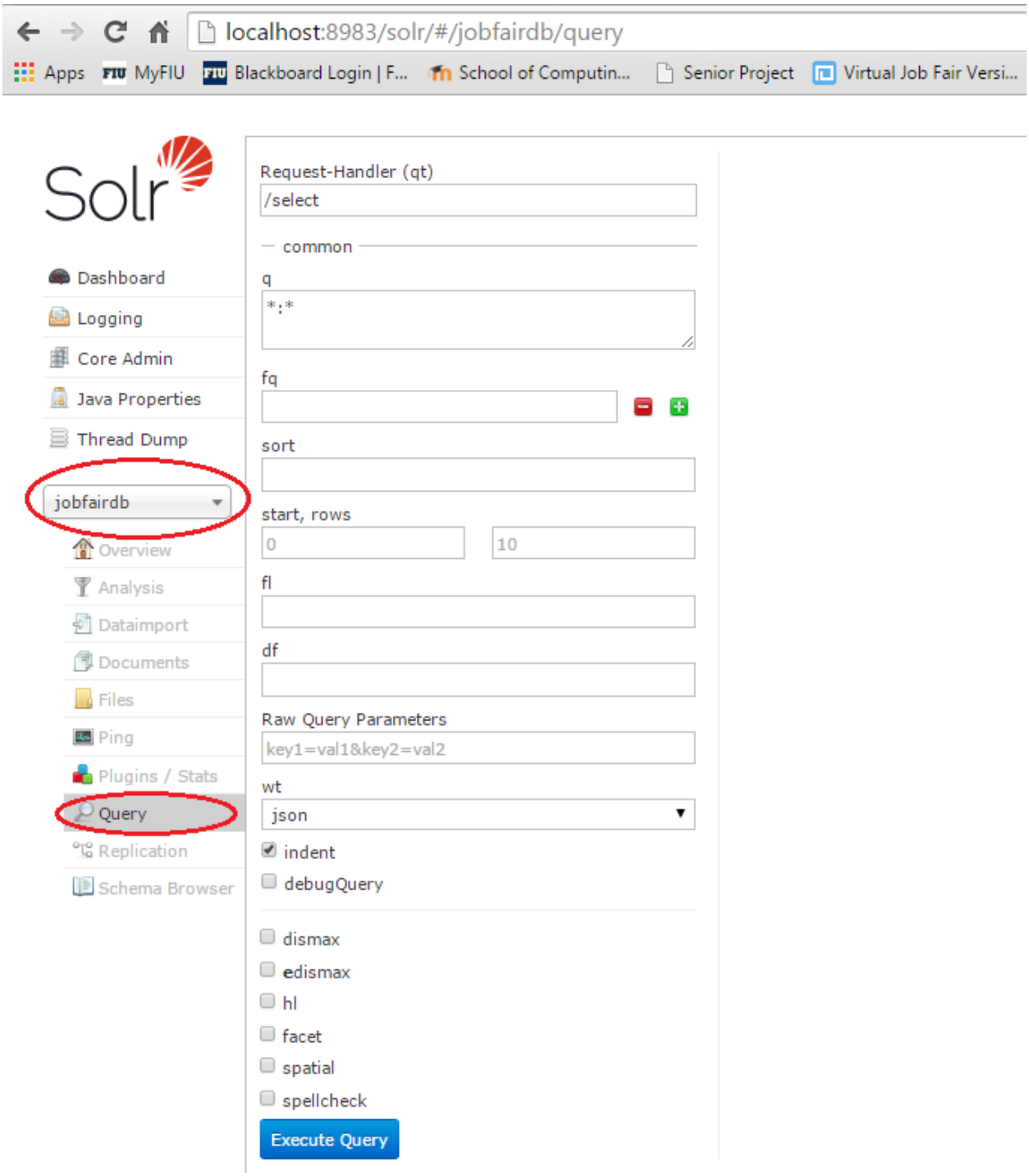
Solr REST API accepts document in the JSON, XML, PDF and a number of different formats to be indexed via POST HTTP request. I strongly recommend to read this page for information on how to index new documents.

http://lucene.apache.org/solr/quickstart.html

Querying Solr Index

To test your imported data or newly indexed data you can navigate in a browser window to:

1. http://localhost:8983/solr or your server i.e. http://vjf-dev.cis.fiu.edu:8983/solr depending where you installed the server.
2. Then select your core
3. Click on Query
4. Click Execute Query



There is a variety of features here in this page such as the format you want solr to return your data or what fields you actually want or the typo of query you want to run (SELECT/UPDATE/etc), or ever if you want to add facets to your results and so forth. For more information about how to query solr please download this pdf:

https://www.apache.org/dyn/closer.cgi/lucene/solr/ref-guide/

Appendix

solr.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!--
 Licensed to the Apache Software Foundation (ASF) under one or more
 contributor license agreements.  See the NOTICE file distributed with
 this work for additional information regarding copyright ownership.
 The ASF licenses this file to You under the Apache License, Version 2.0
 (the "License"); you may not use this file except in compliance with
 the License.  You may obtain a copy of the License at

     http://www.apache.org/licenses/LICENSE-2.0

 Unless required by applicable law or agreed to in writing, software
 distributed under the License is distributed on an "AS IS" BASIS,
 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 See the License for the specific language governing permissions and
 limitations under the License.
-->

<!--
   This is an example of a simple "solr.xml" file for configuring one or
   more Solr Cores, as well as allowing Cores to be added, removed, and
   reloaded via HTTP requests.

   More information about options available in this configuration file,
   and Solr Core administration can be found online:
   http://wiki.apache.org/solr/CoreAdmin
-->

<solr>

  <solrcloud>
```

```
    <str name="host">${host:}</str>
    <int name="hostPort">${jetty.port:8983}</int>
    <str name="hostContext">${hostContext:solr}</str>

    <bool name="genericCoreNodeNames">${genericCoreNodeNames:true}</bool>

    <int name="zkClientTimeout">${zkClientTimeout:30000}</int>
    <int name="distribUpdateSoTimeout">${distribUpdateSoTimeout:600000}</int>
    <int name="distribUpdateConnTimeout">${distribUpdateConnTimeout:60000}</int>

  </solrcloud>

  <shardHandlerFactory name="shardHandlerFactory"
    class="HttpShardHandlerFactory">
    <int name="socketTimeout">${socketTimeout:600000}</int>
    <int name="connTimeout">${connTimeout:60000}</int>
  </shardHandlerFactory>

</solr>
```

data-config.xml

```
<?xml version = "1.0" encoding = "UTF-8" ?>
<dataConfig>
 <dataSource type="JdbcDataSource" driver="com.mysql.jdbc.Driver"
url="jdbc:mysql://localhost:3306/jobfairdb" user="root" password="" batchSize="1" />

 <document name="jodfairdb">
  <entity name="solr"
                        query="SELECT * FROM solr"
                        deltaImportQuery= "SELECT * FROM solr WHERE id = '${dih.delta.id}'"
                        deltaQuery = "SELECT id FROM solr WHERE modified >
'${dih.last_index_time}'">
```

```
                <field column="id" name="id"/>
                <field column="username" name="username"/>
                <field column="email" name="email"/>
                <field column="registration_date" name="registration_date"/>
                <field column="first_name" name="first_name"/>
                <field column="last_name" name="last_name"/>
                <field column="image_url" name="image_url"/>
            <field column="type" name="type"/>
            <field column="title" name="title"/>
            <field column="post_date" name="post_date"/>
            <field column="deadline" name="deadline"/>
            <field column="description" name="description"/>
            <field column="compensation" name="compensation"/>
            <field column="other_requirements" name="other_requirements"/>
            <field column="matches_found" name="matches_found"/>
            <field column="posting_url" name="posting_url"/>
            <field column="comp_name" name="comp_name"/>
            <field column="poster_email" name="poster_email"/>
                        <entity name = "student_skills" query= "SELECT skillid FROM
student_skill_map WHERE student_skill_map.userid = '${solr.id}'">
                                <entity name = "skills" query= "SELECT name FROM skillset WHERE
skillset.id = '${student_skills.skillid}'">
                                        <field column="name" name="skillname" />
                                </entity>
                        </entity>
  </entity>
 </document>
</dataConfig>
```