**Web scrapping Linkedin‘s Job postings.**

**Project Logic**:

The nifi will fist fetch html file of your targeted webpage as nifi flow file. Then, a group of processors are used to extract and format only useful information and replace the content of flow file with such. Lastly, content can be loaded into HDFS for further analysis.

**If you want skip all steps,** **you can find complete nifi template from** [**https://github.com/jayjayjohn/nifiExercise**](https://github.com/jayjayjohn/nifiExercise)

Note: linkedin will only show maximum of 25 job postings per page. You will need to go to next page to fetch complete data set. While it can be done by program, it’s not in scope of this exercise.

**Information about target web page**:

To build this project, you must understand the structure of source page of your targeted website and how the information that your want are formatted.

The url we will be targeting is :

<https://fr.linkedin.com/jobs/search?f_TP=1%2C2%2C3%2C4&f_C=1353&locationId=us%3A0&keywords=Tata%20Consultancy%20Services&orig=FCTD&sortBy=DD>

This link will provide us with jobs posted by Tata Consultancy Serviced from past month.

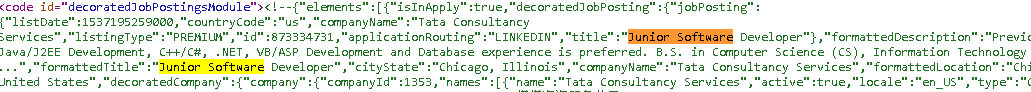
Input the link in your browser to see what the page looks like. Make sure you are not logged in your Linkedin, or page will be slightly different.

Click around to see how the url will change.

You may use ‘start=0 & count=50’ parameter to list more jobs per page. (50 is the max.)

Press Ctrl+U to open the source html page.

Press Ctrl+F to open finder tool. Now pick a job title that you see on the webpage and find the match on the html. 



The purpose of this is to locate the element where the job postings are stored, you can use many other tool to achieve this.

As you can see, job postings are stored in the element <code> where its id = ‘decoratedJobPostingsModule’.

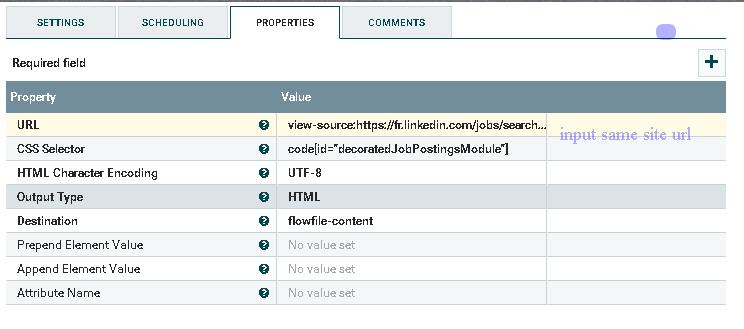
If you carefully study this element you will see the job posting module is formatted in JSON and is enclosed by <!-- and -->.

**Nifi**

1. Add **InvokeHTTP** processor.

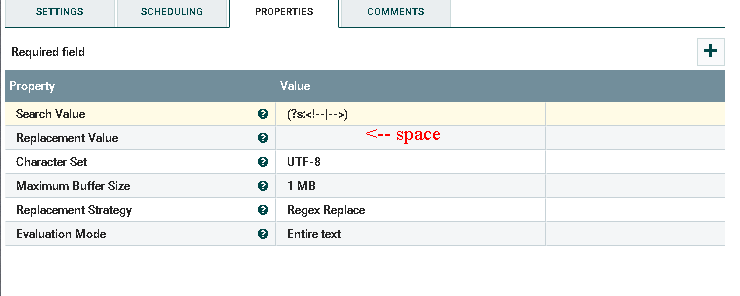
In the properties, for remoteURL input -> <https://fr.linkedin.com/jobs/search?f_TP=1%2C2%2C3%2C4&f_C=1353&locationId=us%3A0&keywords=Tata%20Consultancy%20Services&orig=FCTD&sortBy=DD>

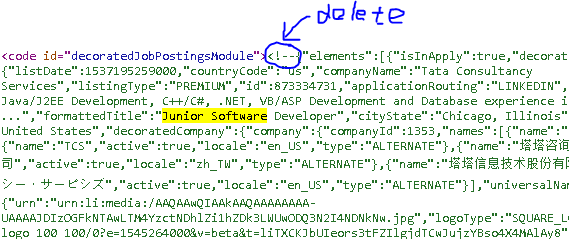
Set the Run Schedule to 10 sec.

1. Add a processor group and name it transformHTML, we will do all the transformation and formation in this processor group. Go into the processor group by double clicking the transformHTML.
2. Add a **InputPort**, name it whatever you want.
3. Add **GetHTMLElement** processor, configure it as follow; 

With above properties, processor is get HTML element based on the CSS Selector and replace entire content with the CSS selected element.

Connect from output port to this processor.

1. Add **ReplaceText** processor. 



The purpose of this is to delete the ‘<!--’ and ‘-->’ which enclose the json module. (we actually replace them with space).

Make connection from **getHTMLelement** to this processor using success relation and auto terminate all others.

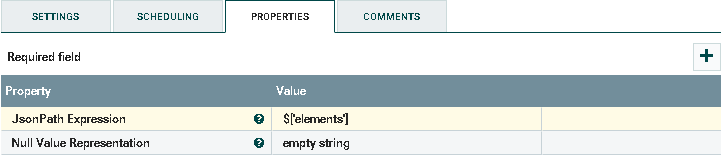
1. From this point, you will be dealing with full json file that contain the job posting.

The structure of json file is rather difficult to see in html page. What you can do is use an online json formatter to make your json file more readable. Go to jsonformatter.curiousconcept.com, Carefully copy and paste all json content in the decoratedJobPostingModule from source html page and format it using the tool in the link. You will see the json is nicely formatted;



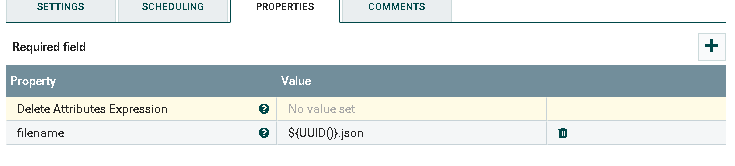
Play around with this tool to understand the hierarchy.

Now, add a **SplitJson** processor.



Note: elements is an array contains all the job object as you can see from the JSON.

1. Add **UpdateAttribute** to assign a unique name to each spitted file.

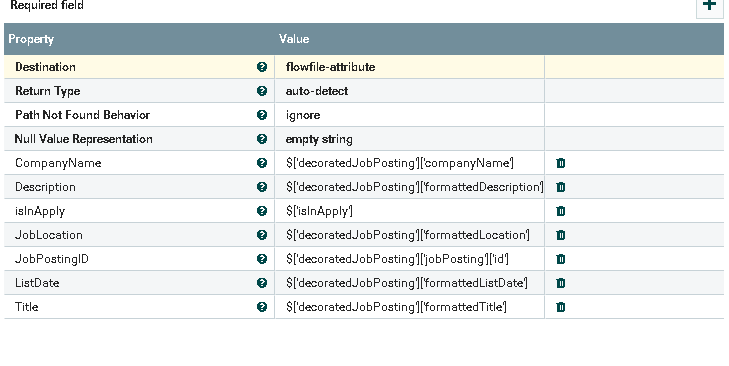


1. Now, we have multiple flowfiles each contain one job listing information formatted in JSON.



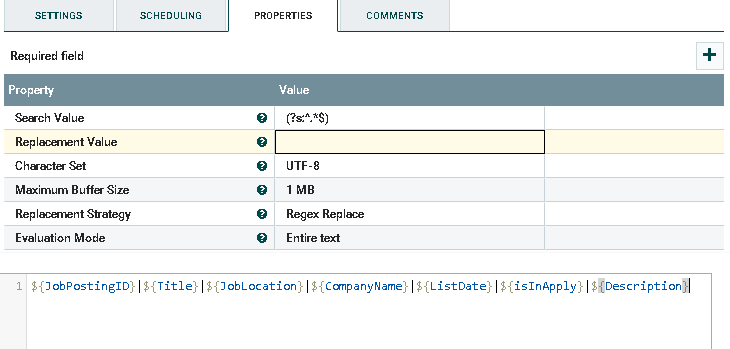
For our purpose, we will only extract few useful information.

Add an **EvaluateJsonPath** processor, configure it as follows;



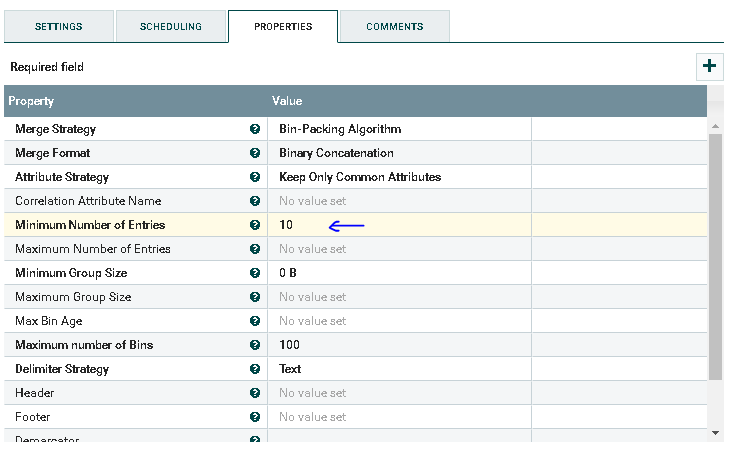
This will extract based on jsonpath, and assign result to the flowfile-attribute.

1. Add **ReplaceText**

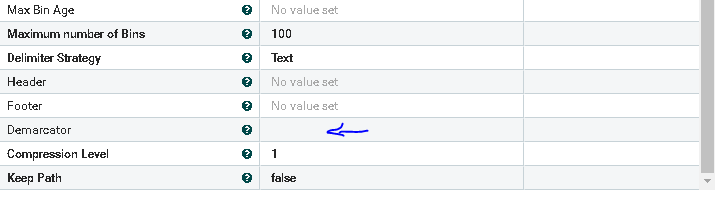


In here, we create a “|” delimited file.

1. Add **MergeContent**

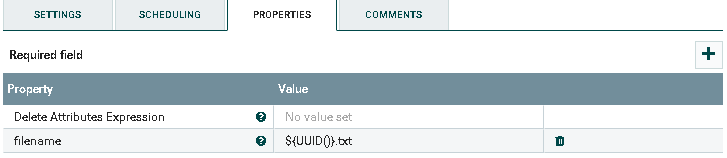


You can specify minimum number of rows in one file in above pointed property.

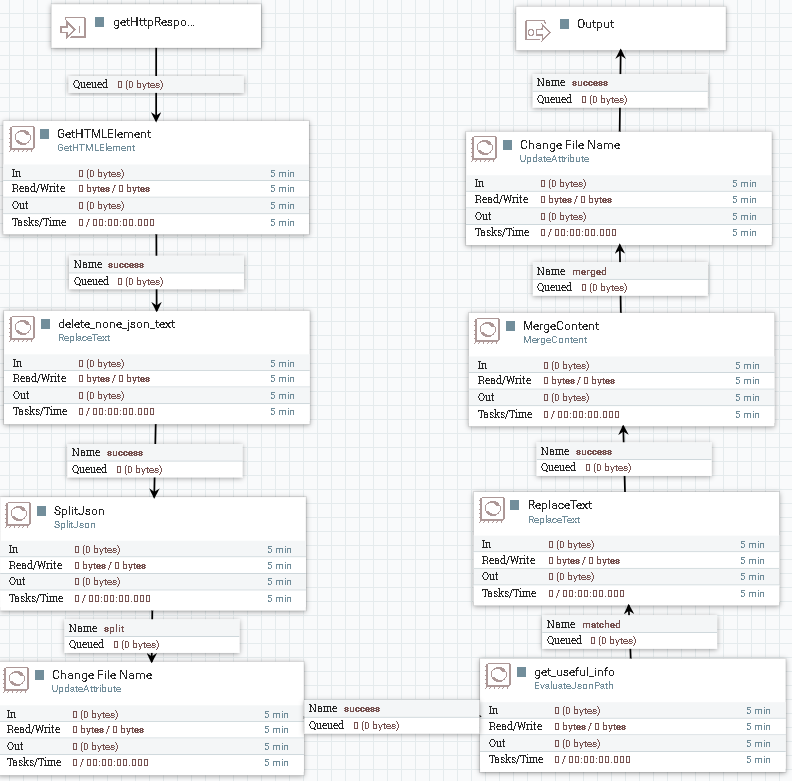


Press ctrl+enter to input next line to Demarcator property.

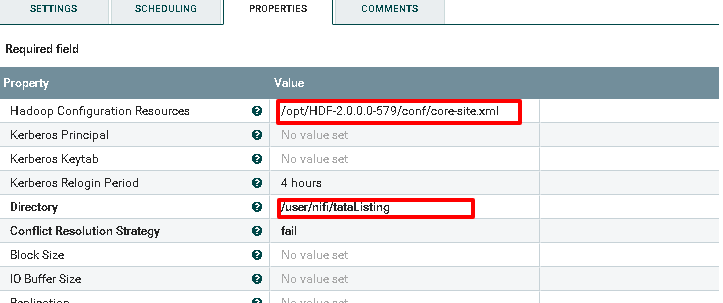
1. Add **UpdateAttribute** to assign a unique filename to the merged file.



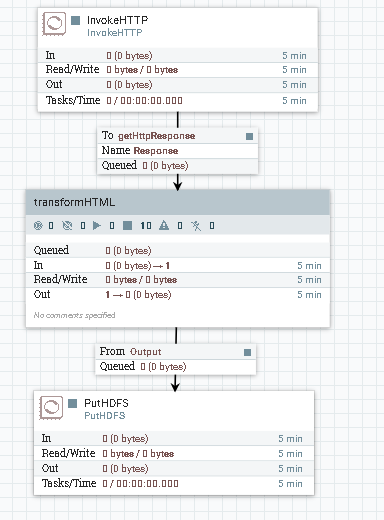
1. Add an **output** port.
2. Make connection as shown in following figure. Ensure to auto terminate all other relationships.



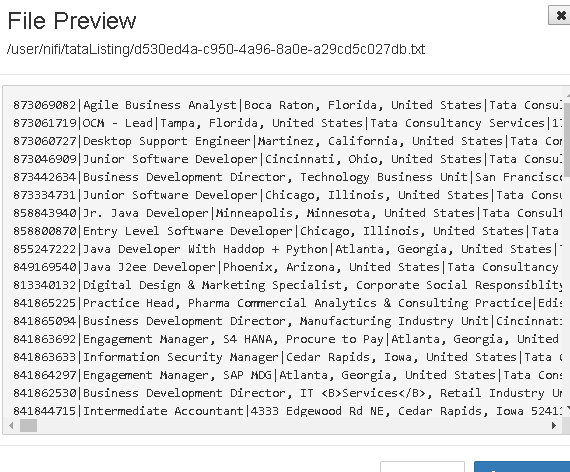
1. Exit the processor group be click on the path at lower left corner
2. Add **putHDFS** processor, follow the following step to configure it.
   * 1. Find core-site.xml in your local machine. If you are using hdp sandbox, it is located at /etc/hadoop/conf/. make a copy to /opt/HDF-2.0.0.0-579/conf/ or any nifi owned path.
     2. Open the copied xml, find properties by the name of io.compression.codec.lzo.class and io.compression.codecs. Delete these two properties.
     3. Input the full path of your modified core-site.xml to the processor and give it a output directory



Here is how your completed template should look like;



Here is how your final output look like;

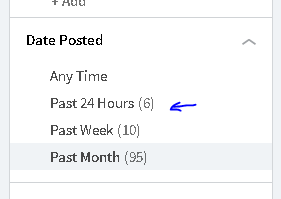


If you want more fun, you can create hive table and run query against the file to answer the following question.

Count the job positing at certain state/city

Find all jobs with title that contain ‘data’

In addition, you can filter the result to past 24 hour on Linkedin

 The url paramenter will changes slightly. You can run the nifi work flow every 24 hr to fetch daily job posting using that url.