# Wuzzuf Task – Wuzzuf Indexer:

## 1-Introduction:

- The search engine that I installed and used is ElasticSearch; because it was the first one suggested in the document, and it worked well.
- I used the Python API under Ubuntu Linux environment to index and search for the data.
- I didn't know anything about ElasticSearch, Curl, Docker, indexing and all the python libraries that I used.
- You can find the code that I made to index and search in this repository: https://github.com/AbdelrahmanRadwan/Wuzzuf-Indexer
- The task took me 4 days around 5 hours per day to finish. 2 days for research, 1 day for implementation, 1 day for documentation, I needed quiet long time to figure things out because these concepts are new for me, and the Elasticsearch document is not good enough at least for me.
- The task runs correctly but the "MoreLikeThis Handler" didn't work well with me, I think it's a simple error, I will try to resolve it later, and if I managed to do this, I will email you.
- The search now is done using the " match" query only, I will extend it also later, the indexing is done correctly, but I'm doing it on only part of the dataset not all of it to shorten the time needed for running and debugging.

## 2- The Problem & Its solution:

### A- Problem Overview:

The problem is that, you have many users, each of them has "passion" towards specific type of jobs, which somehow satisfy his/her skills.

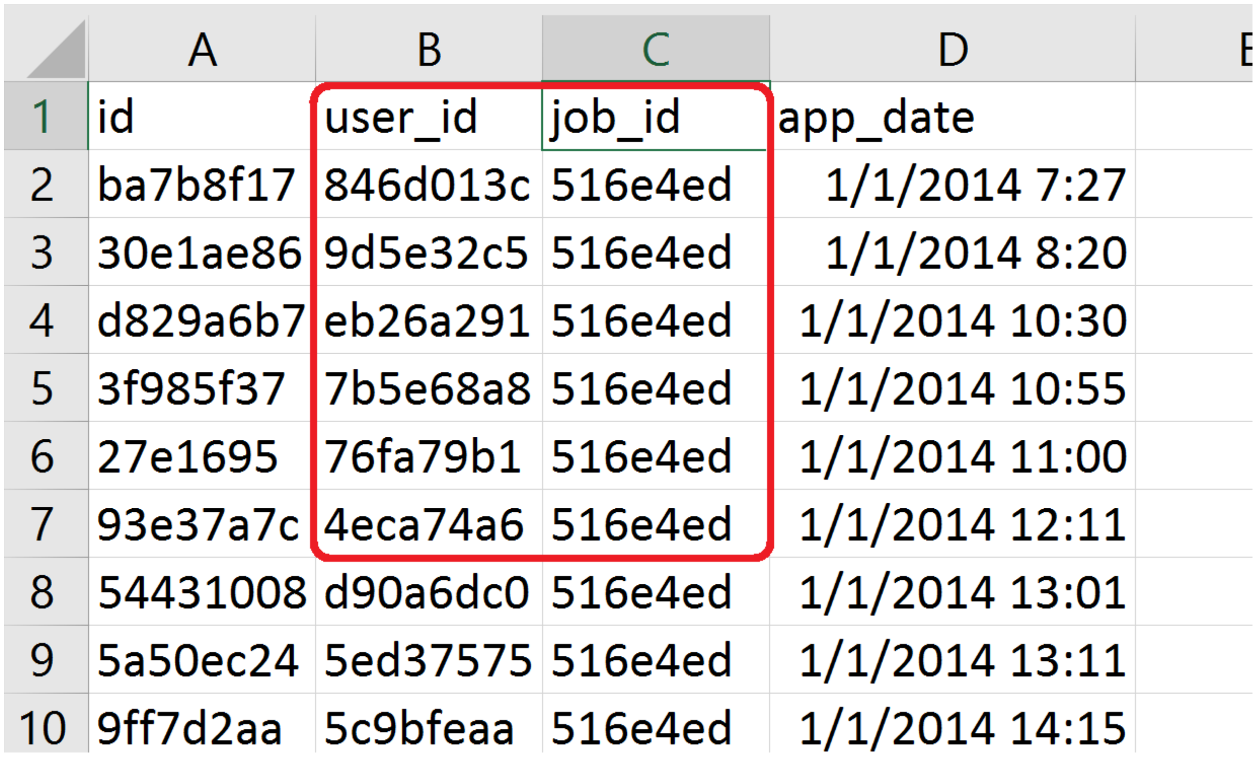
On the other side, you have list of jobs with different features, features like (place, salary, job title, job rule, …etc).

You want to match the jobs with the candidates' passion/interests of jobs(in other words, you want to choose some of the jobs that you have to show them for the candidate based on his/her passion).

## B- The implemented solution:

Here is how the solution here(https://github.com/AbdelrahmanRadwan/Wuzzuf-Indexer) works:

1. In the dataset that called (Wuzzuf_Applications_Sample):

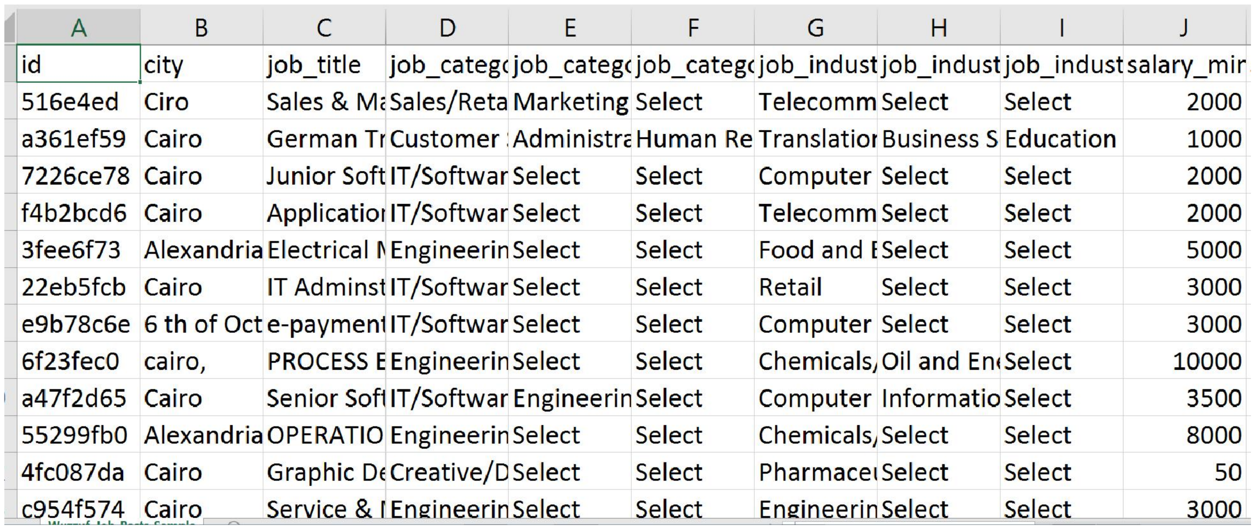


| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | id | user_id | job_id | app_date | |
| 2 | ba7b8f17 | 846d013c | 516e4ed | 1/1/2014 7:27 | |
| 3 | 30e1ae86 | 9d5e32c5 | 516e4ed | 1/1/2014 8:20 | |
| 4 | d829a6b7 | eb26a291 | 516e4ed | 1/1/2014 10:30 | |
| 5 | 3f985f37 | 7b5e68a8 | 516e4ed | 1/1/2014 10:55 | |
| 6 | 27e1695 | 76fa79b1 | 516e4ed | 1/1/2014 11:00 | |
| 7 | 93e37a7c | 4eca74a6 | 516e4ed | 1/1/2014 12:11 | |
| 8 | 54431008 | d90a6dc0 | 516e4ed | 1/1/2014 13:01 | |
| 9 | 5a50ec24 | 5ed37575 | 516e4ed | 1/1/2014 13:11 | |
| 10 | 9ff7d2aa | 5c9bfeaa | 516e4ed | 1/1/2014 14:15 | |

Wuzzuf_Applications_Sample

These two columns can help us get the passion of the user, or his interests, I mean the user whose ID is X who applied for job Y with ID Y_ID will be interested in any jobs like this one, so this is a good way to get the user passion or interest.

2. In the dataset that called (Wuzzuf_Job_Posts_Sample):



| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| | id | city | job_title | job_categ | job_categ | job_categ | job_indust | job_indust | job_indust | salary_mir |
| | 516e4ed | Ciro | Sales & Ma | Sales/Reta | Marketing | Select | Telecomm | Select | Select | 2000 |
| | a361ef59 | Cairo | German Tr | Customer : | Administra | Human Re | Translatio | Business S | Education | 1000 |
| | 7226ce78 | Cairo | Junior Soft | IT/Softwar | Select | Select | Computer | Select | Select | 2000 |
| | f4b2bcd6 | Cairo | Applicatio | IT/Softwar | Select | Select | Telecomm | Select | Select | 2000 |
| | 3fee6f73 | Alexandria | Electrical M | Engineerin | Select | Select | Food and E | Select | Select | 5000 |
| | 22eb5fcb | Cairo | IT Adminst | IT/Softwar | Select | Select | Retail | Select | Select | 3000 |
| | e9b78c6e | 6 th of Oct | e-payment | IT/Softwar | Select | Select | Computer | Select | Select | 3000 |
| | 6f23fec0 | cairo, | PROCESS E | Engineerin | Select | Select | Chemicals, | Oil and En | Select | 10000 |
| | a47f2d65 | Cairo | Senior Soft | IT/Softwar | Engineerin | Select | Computer | Informatio | Select | 3500 |
| | 55299fb0 | Alexandria | OPERATIO | Engineerin | Select | Select | Chemicals, | Select | Select | 8000 |
| | 4fc087da | Cairo | Graphic De | Creative/D | Select | Select | Pharmace | Select | Select | 50 |
| | c954f574 | Cairo | Service & I | Engineerin | Select | Select | Engineerin | Select | Select | 3000 |

Wuzzuf_Job_Posts_Sample

We can get the job tune, I mean by tune the collection of features.

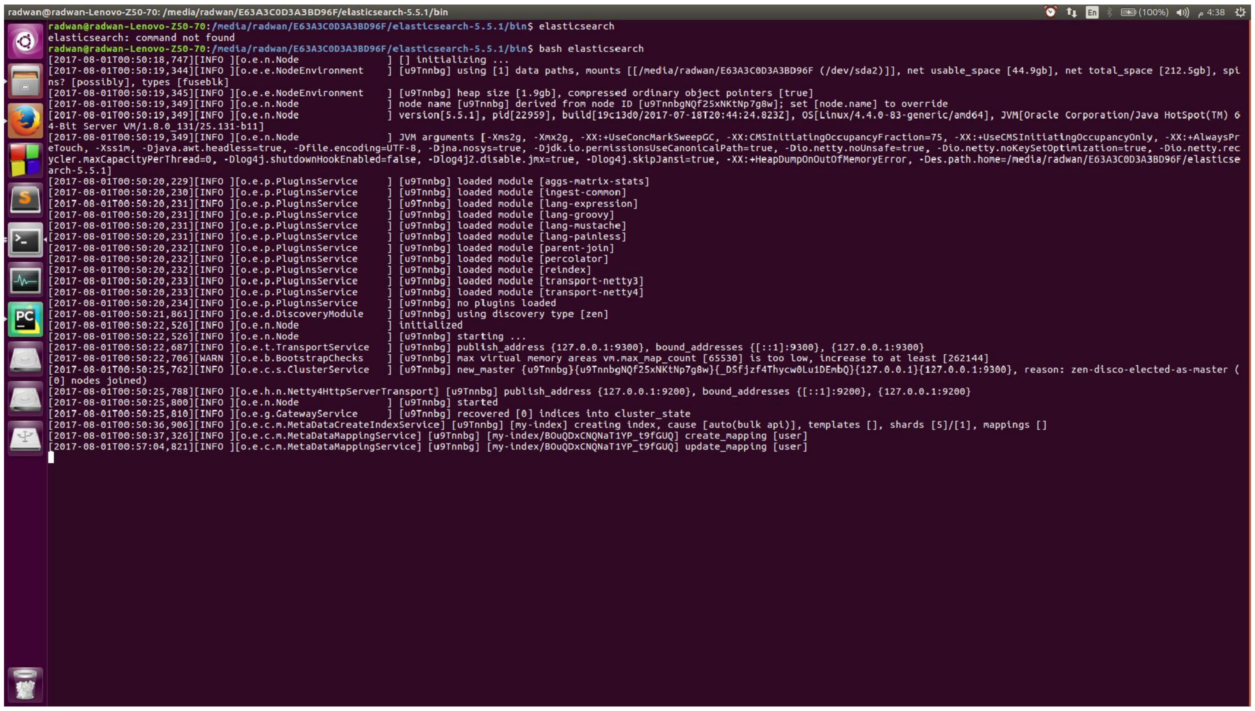
3. After indexing all the jobs in elasticsearch, and mapping the job id to the whole features, then we can search for the similar tunes of the jobs using elasticsearch search.
4. It's easy to get the user passion, simply it's any job id which he/she was interested in, and by using any query like "match" or "more like this" in Curl, we can get more jobs that this user maybe interested in.

## Code, Setup and workflow Overview:

### A- Setup the environment:

- I tried to run the Elasticsearch on windows 8.1, but I figured out later that I need to write Curl queries, and after searching about Curl, I figured out that I need to setup it on windows, so I moved to Ubuntu to check if it's already setup or not, and I found it works well ( I had the two OS already, I preferred Windows because it will be easier for me to document my work on it, but I could handle all that stuff).

- Running the Elasticsearch and lunching it was not so hard, I opened the local address(http://localhost:9200/) and investigated it.

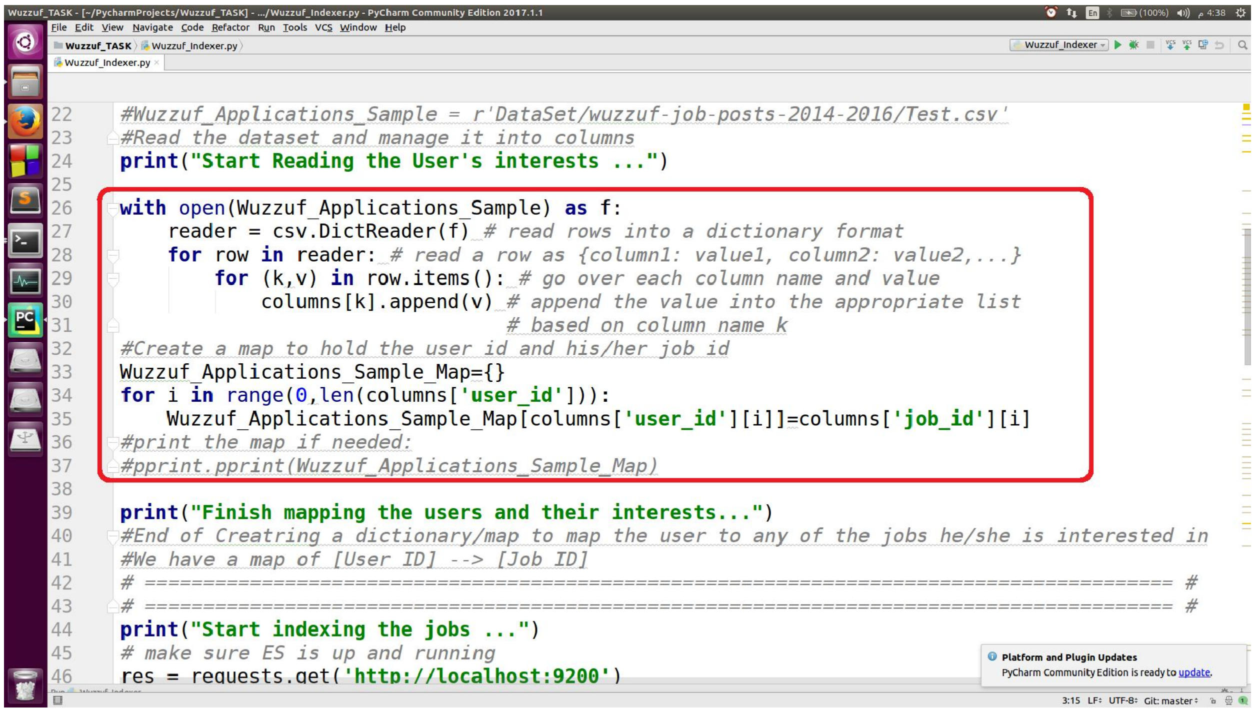


```
radwan@radwan-Lenovo-Z50-70: /media/radwan/E63A3C0D3A3BD96F/elasticsearch-5.5.1/bin
radwan@radwan-Lenovo-Z50-70:/media/radwan/E63A3C0D3A3BD96F/elasticsearch-5.5.1/bin$ elasticsearch
elasticsearch: command not found
radwan@radwan-Lenovo-Z50-70:/media/radwan/E63A3C0D3A3BD96F/elasticsearch-5.5.1/bin$ bash elasticsearch
[2017-08-01T00:50:18,747][INFO ][o.e.n.Node               ] [] initializing ...
[2017-08-01T00:50:19,344][INFO ][o.e.e.NodeEnvironment    ] [u9Tnnbg] using [1] data paths, mounts [[/media/radwan/E63A3C0D3A3BD96F (/dev/sda2)]], net usable_space [44.9gb], net total_space [212.5gb], spi
ns? [possibly], types [fuseblk]
[2017-08-01T00:50:19,345][INFO ][o.e.e.NodeEnvironment    ] [u9Tnnbg] heap size [1.9gb], compressed ordinary object pointers [true]
[2017-08-01T00:50:19,349][INFO ][o.e.n.Node               ] node name [u9Tnnbg] derived from node ID [u9TnnbgNQf25xNKtNp7g8w]; set [node.name] to override
[2017-08-01T00:50:19,349][INFO ][o.e.n.Node               ] version[5.5.1], pid[22959], build[19c13d0/2017-07-18T20:44:24.823Z], OS[Linux/4.4.0-83-generic/amd64], JVM[Oracle Corporation/Java HotSpot(TM) 6
4-Bit Server VM/1.8.0_131/25.131-b11]
[2017-08-01T00:50:19,349][INFO ][o.e.n.Node               ] JVM arguments [-Xms2g, -Xmx2g, -XX:+UseConcMarkSweepGC, -XX:CMSInitiatingOccupancyFraction=75, -XX:+UseCMSInitiatingOccupancyOnly, -XX:+AlwaysPr
eTouch, -Xss1m, -Djava.awt.headless=true, -Dfile.encoding=UTF-8, -Djna.nosys=true, -Djdk.io.permissionsUseCanonicalPath=true, -Dio.netty.noUnsafe=true, -Dio.netty.noKeySetOptimization=true, -Dio.netty.rec
ycler.maxCapacityPerThread=0, -Dlog4j.shutdownHookEnabled=false, -Dlog4j2.disable.jmx=true, -Dlog4j.skipJansi=true, -XX:+HeapDumpOnOutOfMemoryError, -Des.path.home=/media/radwan/E63A3C0D3A3BD96F/elasticse
arch-5.5.1]
[2017-08-01T00:50:20,229][INFO ][o.e.p.PluginsService     ] [u9Tnnbg] loaded module [aggs-matrix-stats]
[2017-08-01T00:50:20,230][INFO ][o.e.p.PluginsService     ] [u9Tnnbg] loaded module [ingest-common]
[2017-08-01T00:50:20,231][INFO ][o.e.p.PluginsService     ] [u9Tnnbg] loaded module [lang-expression]
[2017-08-01T00:50:20,231][INFO ][o.e.p.PluginsService     ] [u9Tnnbg] loaded module [lang-groovy]
[2017-08-01T00:50:20,231][INFO ][o.e.p.PluginsService     ] [u9Tnnbg] loaded module [lang-mustache]
[2017-08-01T00:50:20,231][INFO ][o.e.p.PluginsService     ] [u9Tnnbg] loaded module [lang-painless]
[2017-08-01T00:50:20,232][INFO ][o.e.p.PluginsService     ] [u9Tnnbg] loaded module [parent-join]
[2017-08-01T00:50:20,232][INFO ][o.e.p.PluginsService     ] [u9Tnnbg] loaded module [percolator]
[2017-08-01T00:50:20,232][INFO ][o.e.p.PluginsService     ] [u9Tnnbg] loaded module [reindex]
[2017-08-01T00:50:20,233][INFO ][o.e.p.PluginsService     ] [u9Tnnbg] loaded module [transport-netty3]
[2017-08-01T00:50:20,233][INFO ][o.e.p.PluginsService     ] [u9Tnnbg] loaded module [transport-netty4]
[2017-08-01T00:50:20,234][INFO ][o.e.p.PluginsService     ] [u9Tnnbg] no plugins loaded
[2017-08-01T00:50:21,861][INFO ][o.e.d.DiscoveryModule    ] [u9Tnnbg] using discovery type [zen]
[2017-08-01T00:50:22,526][INFO ][o.e.n.Node               ] initialized
[2017-08-01T00:50:22,526][INFO ][o.e.n.Node               ] [u9Tnnbg] starting ...
[2017-08-01T00:50:22,687][INFO ][o.e.t.TransportService   ] [u9Tnnbg] publish_address {127.0.0.1:9300}, bound_addresses {[::1]:9300}, {127.0.0.1:9300}
[2017-08-01T00:50:22,706][WARN ][o.e.b.BootstrapChecks    ] [u9Tnnbg] max virtual memory areas vm.max_map_count [65530] is too low, increase to at least [262144]
[2017-08-01T00:50:25,762][INFO ][o.e.c.s.ClusterService   ] [u9Tnnbg] new_master {u9Tnnbg}{u9TnnbgNQf25xNKtNp7g8w}{_DSfjzf4Thycw0Lu1DEmbQ}{127.0.0.1}{127.0.0.1:9300}, reason: zen-disco-elected-as-master (
[0] nodes joined)
[2017-08-01T00:50:25,788][INFO ][o.e.h.n.Netty4HttpServerTransport] [u9Tnnbg] publish_address {127.0.0.1:9200}, bound_addresses {[::1]:9200}, {127.0.0.1:9200}
[2017-08-01T00:50:25,800][INFO ][o.e.n.Node               ] [u9Tnnbg] started
[2017-08-01T00:50:25,810][INFO ][o.e.g.GatewayService     ] [u9Tnnbg] recovered [0] indices into cluster_state
[2017-08-01T00:50:36,906][INFO ][o.e.c.m.MetaDataCreateIndexService] [u9Tnnbg] [my-index] creating index, cause [auto(bulk api)], templates [], shards [5]/[1], mappings []
[2017-08-01T00:50:37,326][INFO ][o.e.c.m.MetaDataMappingService] [u9Tnnbg] [my-index/BOuQDxCNQNaT1YP_t9fGUQ] create_mapping [user]
[2017-08-01T00:57:04,821][INFO ][o.e.c.m.MetaDataMappingService] [u9Tnnbg] [my-index/BOuQDxCNQNaT1YP_t9fGUQ] update_mapping [user]
```

## B- Extracting user's passion:

- As I mentioned before, using the User ID, you can pick any of his/her interesting job ID, this is done and saved in simple map.

- I know that the one user can be interested in many different categories and jobs, but I think getting only one job id per user is much simpler, faster and easier, maybe extending the work will make better results.


Wuzzuf_TASK - [~/PycharmProjects/Wuzzuf_TASK] - .../Wuzzuf_Indexer.py - PyCharm Community Edition 2017.1.1
File Edit View Navigate Code Refactor Run Tools VCS Window Help
Wuzzuf_TASK
Wuzzuf_Indexer.py
Wuzzuf_Indexer
Platform and Plugin Updates
PyCharm Community Edition is ready to update.
3:15 LF÷ UTF-8÷ Git: master÷

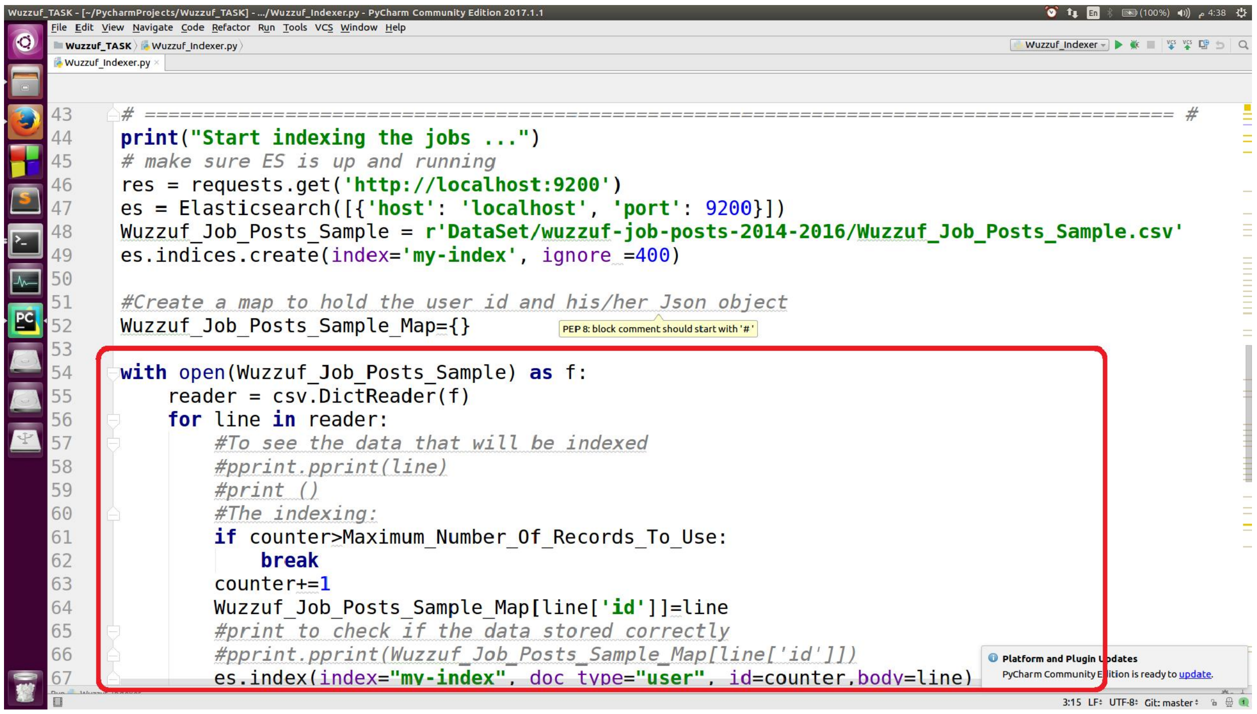
```python
#Wuzzuf_Applications_Sample = r'DataSet/wuzzuf-job-posts-2014-2016/Test.csv'
#Read the dataset and manage it into columns
print("Start Reading the User's interests ...")

with open(Wuzzuf_Applications_Sample) as f:
    reader = csv.DictReader(f) # read rows into a dictionary format
    for row in reader: # read a row as {column1: value1, column2: value2,...}
        for (k,v) in row.items(): # go over each column name and value
            columns[k].append(v) # append the value into the appropriate list
                                 # based on column name k
#Create a map to hold the user id and his/her job id
Wuzzuf_Applications_Sample_Map={}
for i in range(0,len(columns['user_id'])):
    Wuzzuf_Applications_Sample_Map[columns['user_id'][i]]=columns['job_id'][i]
#print the map if needed:
#pprint.pprint(Wuzzuf_Applications_Sample_Map)

print("Finish mapping the users and their interests...")
#End of Creatring a dictionary/map to map the user to any of the jobs he/she is interested in
#We have a map of [User ID] --> [Job ID]
# ========================================================================================== #
# ========================================================================================== #
print("Start indexing the jobs ...")
# make sure ES is up and running
res = requests.get('http://localhost:9200')
```

## C- Jobs Indexing & Mapping:

- The indexing is done by the elasticsearch.index finction, it takes the body of what you want to index.
- After the indexing, we map the id to the json object which contains the information about the job.


Wuzzuf_TASK - [~/PycharmProjects/Wuzzuf_TASK] - .../Wuzzuf_Indexer.py - PyCharm Community Edition 2017.1.1
File Edit View Navigate Code Refactor Run Tools VCS Window Help
Wuzzuf_TASK Wuzzuf_Indexer.py
Wuzzuf_Indexer.py
Wuzzuf_Indexer
PEP 8: block comment should start with '# '
Platform and Plugin Updates
PyCharm Community Edition is ready to update.
3:15 LF: UTF-8: Git: master:

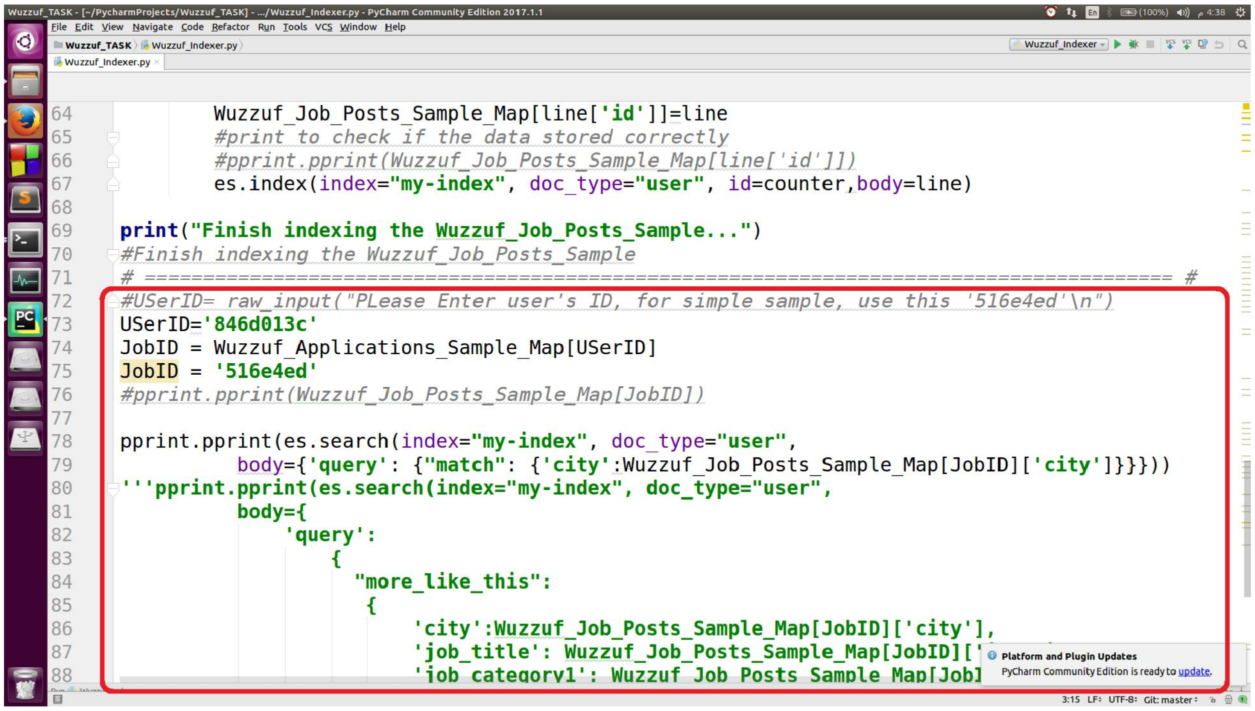
```
# ========================================================================================= #
print("Start indexing the jobs ...")
# make sure ES is up and running
res = requests.get('http://localhost:9200')
es = Elasticsearch([{'host': 'localhost', 'port': 9200}])
Wuzzuf_Job_Posts_Sample = r'DataSet/wuzzuf-job-posts-2014-2016/Wuzzuf_Job_Posts_Sample.csv'
es.indices.create(index='my-index', ignore =400)

#Create a map to hold the user id and his/her Json object
Wuzzuf_Job_Posts_Sample_Map={}

with open(Wuzzuf_Job_Posts_Sample) as f:
    reader = csv.DictReader(f)
    for line in reader:
        #To see the data that will be indexed
        #pprint.pprint(line)
        #print ()
        #The indexing:
        if counter>Maximum_Number_Of_Records_To_Use:
            break
        counter+=1
        Wuzzuf_Job_Posts_Sample_Map[line['id']]=line
        #print to check if the data stored correctly
        #pprint.pprint(Wuzzuf_Job_Posts_Sample_Map[line['id']])
        es.index(index="my-index", doc_type="user", id=counter,body=line)
```

## D- The Query part:

- As we are only creating a prototype, we can take the user ID as input from the console of as an argument, and then we will use the first map to get a job ID for which that user was interested before, and we then get the Json object of that job which have this ID (Using the second map) [I tried to do this using search query to search for the id inside the json object, but it returned more than element, I don't know why, but as a simple and clean solution (but needs more memory) I used the map of ID and Json object.
- After getting the job's Json object, we shall use some of these information to search using the more like this, we will not use all the features, something like how many views and how many vacations shouldn't affect the search results.



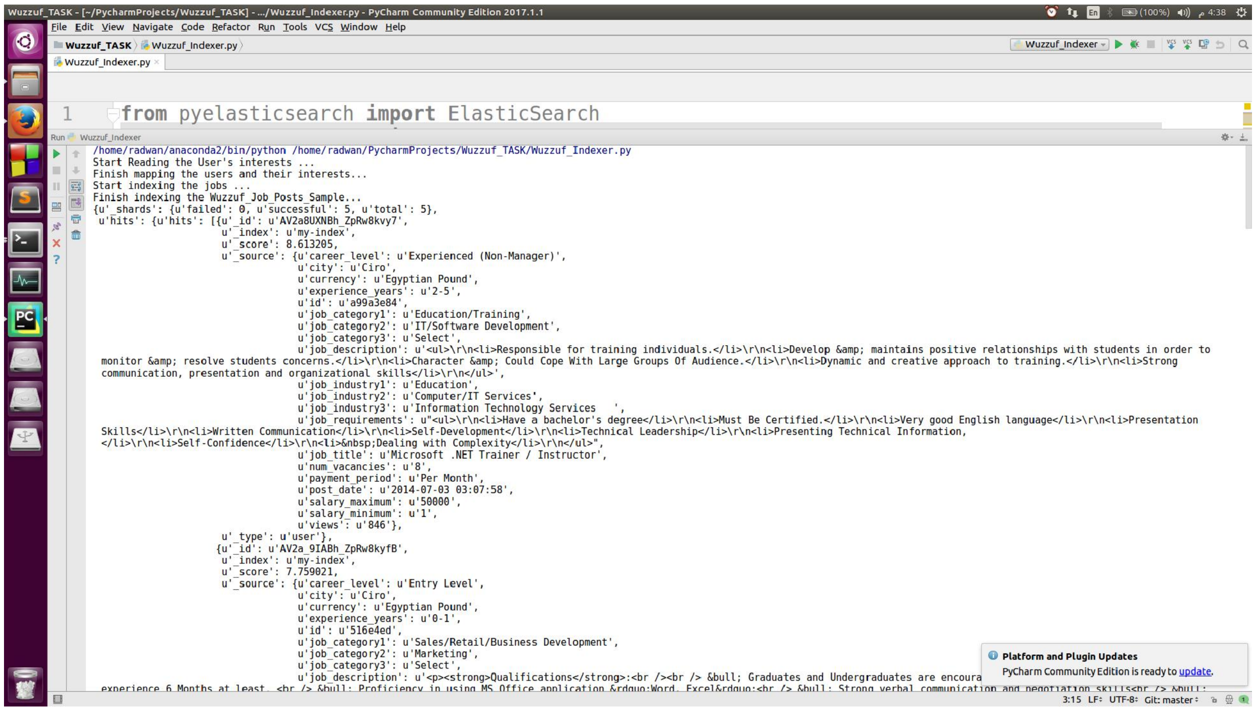
```
            Wuzzuf_Job_Posts_Sample_Map[line['id']]=line
            #print to check if the data stored correctly
            #pprint.pprint(Wuzzuf_Job_Posts_Sample_Map[line['id']])
            es.index(index="my-index", doc_type="user", id=counter,body=line)

print("Finish indexing the Wuzzuf_Job_Posts_Sample...")
#Finish indexing the Wuzzuf_Job_Posts_Sample
# ================================================================================ #
#USerID= raw_input("PLease Enter user's ID, for simple sample, use this '516e4ed'\n")
USerID='846d013c'
JobID = Wuzzuf_Applications_Sample_Map[USerID]
JobID = '516e4ed'
#pprint.pprint(Wuzzuf_Job_Posts_Sample_Map[JobID])

pprint.pprint(es.search(index="my-index", doc_type="user",
          body={'query': {"match": {'city':Wuzzuf_Job_Posts_Sample_Map[JobID]['city']}}}))
'''pprint.pprint(es.search(index="my-index", doc_type="user",
          body={
              'query':
                  {
                    "more_like_this":
                     {
                         'city':Wuzzuf_Job_Posts_Sample_Map[JobID]['city'],
                         'job_title': Wuzzuf_Job_Posts_Sample_Map[JobID]['
                         'job_category1': Wuzzuf_Job_Posts_Sample_Map[JobI
```

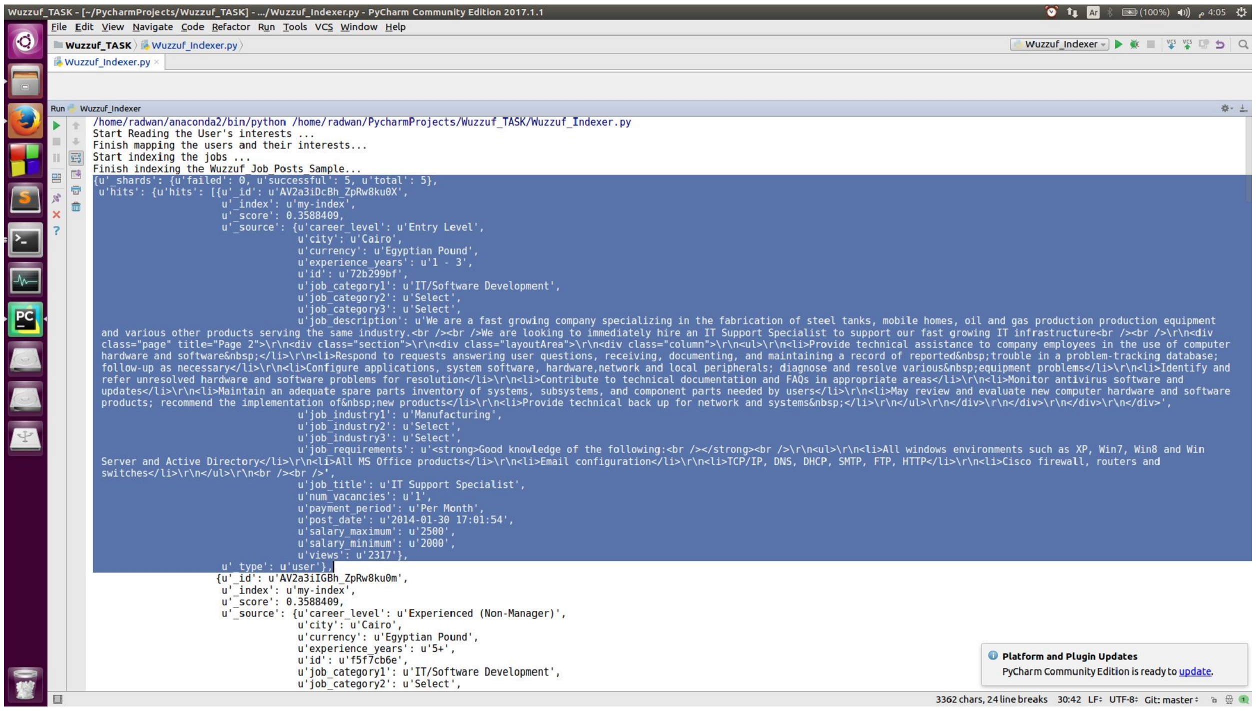
## E- The results:

- After executing the query, we get the Json objects that matches the query printed in the console.



```
from pyelasticsearch import ElasticSearch
```

```
/home/radwan/anaconda2/bin/python /home/radwan/PycharmProjects/Wuzzuf_TASK/Wuzzuf_Indexer.py
Start Reading the User's interests ...
Finish mapping the users and their interests...
Start indexing the jobs ...
Finish indexing the Wuzzuf_Job_Posts_Sample...
{u'_shards': {u'failed': 0, u'successful': 5, u'total': 5},
 u'hits': {u'hits': [{u'_id': u'AV2a8UXNBh_ZpRw8kvy7',
                      u'_index': u'my-index',
                      u'_score': 8.613205,
                      u'_source': {u'career_level': u'Experienced (Non-Manager)',
                                   u'city': u'Ciro',
                                   u'currency': u'Egyptian Pound',
                                   u'experience_years': u'2-5',
                                   u'id': u'a99a3e84',
                                   u'job_category1': u'Education/Training',
                                   u'job_category2': u'IT/Software Development',
                                   u'job_category3': u'Select',
                                   u'job_description': u'<ul>\r\n<li>Responsible for training individuals.</li>\r\n<li>Develop &amp; maintains positive relationships with students in order to
 monitor &amp; resolve students concerns.</li>\r\n<li>Character &amp; Could Cope With Large Groups Of Audience.</li>\r\n<li>Dynamic and creative approach to training.</li>\r\n<li>Strong
 communication, presentation and organizational skills</li>\r\n</ul>',
                                   u'job_industry1': u'Education',
                                   u'job_industry2': u'Computer/IT Services',
                                   u'job_industry3': u'Information Technology Services   ',
                                   u'job_requirements': u"<ul>\r\n<li>Have a bachelor's degree</li>\r\n<li>Must Be Certified.</li>\r\n<li>Very good English language</li>\r\n<li>Presentation
 Skills</li>\r\n<li>Written Communication</li>\r\n<li>Self-Development</li>\r\n<li>Technical Leadership</li>\r\n<li>Presenting Technical Information,
 </li>\r\n<li>Self-Confidence</li>\r\n<li>&nbsp;Dealing with Complexity</li>\r\n</ul>",
                                   u'job_title': u'Microsoft .NET Trainer / Instructor',
                                   u'num_vacancies': u'8',
                                   u'payment_period': u'Per Month',
                                   u'post_date': u'2014-07-03 03:07:58',
                                   u'salary_maximum': u'50000',
                                   u'salary_minimum': u'1',
                                   u'views': u'846'},
                      u'_type': u'user'},
                     {u'_id': u'AV2a_9IABh_ZpRw8kyfB',
                      u'_index': u'my-index',
                      u'_score': 7.759021,
                      u'_source': {u'career_level': u'Entry Level',
                                   u'city': u'Ciro',
                                   u'currency': u'Egyptian Pound',
                                   u'experience_years': u'0-1',
                                   u'id': u'516e4ed',
                                   u'job_category1': u'Sales/Retail/Business Development',
                                   u'job_category2': u'Marketing',
                                   u'job_category3': u'Select',
                                   u'job_description': u'<p><strong>Qualifications</strong>:<br /><br /> &bull; Graduates and Undergraduates are encoura
```


Wuzzuf_TASK - [~/PycharmProjects/Wuzzuf_TASK] - .../Wuzzuf_Indexer.py - PyCharm Community Edition 2017.1.1
File Edit View Navigate Code Refactor Run Tools VCS Window Help
Wuzzuf_TASK Wuzzuf_Indexer.py
Wuzzuf_Indexer.py
Run Wuzzuf_Indexer

```
/home/radwan/anaconda2/bin/python /home/radwan/PycharmProjects/Wuzzuf_TASK/Wuzzuf_Indexer.py
Start Reading the User's interests ...
Finish mapping the users and their interests...
Start indexing the jobs ...
Finish indexing the Wuzzuf_Job_Posts_Sample...
{u'_shards': {u'failed': 0, u'successful': 5, u'total': 5},
 u'hits': {u'hits': [{u'_id': u'AV2a3iDcBh_ZpRw8ku0X',
                      u'_index': u'my-index',
                      u'_score': 0.3588409,
                      u'_source': {u'career_level': u'Entry Level',
                                   u'city': u'Cairo',
                                   u'currency': u'Egyptian Pound',
                                   u'experience_years': u'1 - 3',
                                   u'id': u'72b299bf',
                                   u'job_category1': u'IT/Software Development',
                                   u'job_category2': u'Select',
                                   u'job_category3': u'Select',
                                   u'job_description': u'We are a fast growing company specializing in the fabrication of steel tanks, mobile homes, oil and gas production production equipment
 and various other products serving the same industry.<br /><br />We are looking to immediately hire an IT Support Specialist to support our fast growing IT infrastructure<br /><br />\r\n<div
 class="page" title="Page 2">\r\n<div class="section">\r\n<div class="layoutArea">\r\n<div class="column">\r\n<ul>\r\n<li>Provide technical assistance to company employees in the use of computer
 hardware and software&nbsp;</li>\r\n<li>Respond to requests answering user questions, receiving, documenting, and maintaining a record of reported&nbsp;trouble in a problem-tracking database;
 follow-up as necessary</li>\r\n<li>Configure applications, system software, hardware,network and local peripherals; diagnose and resolve various&nbsp;equipment problems</li>\r\n<li>Identify and
 refer unresolved hardware and software problems for resolution</li>\r\n<li>Contribute to technical documentation and FAQs in appropriate areas</li>\r\n<li>Monitor antivirus software and
 updates</li>\r\n<li>Maintain an adequate spare parts inventory of systems, subsystems, and component parts needed by users</li>\r\n<li>May review and evaluate new computer hardware and software
 products; recommend the implementation of&nbsp;new products</li>\r\n<li>Provide technical back up for network and systems&nbsp;</li>\r\n</ul>\r\n</div>\r\n</div>\r\n</div>\r\n</div>',
                                   u'job_industry1': u'Manufacturing',
                                   u'job_industry2': u'Select',
                                   u'job_industry3': u'Select',
                                   u'job_requirements': u'<strong>Good knowledge of the following:<br /></strong><br />\r\n<ul>\r\n<li>All windows environments such as XP, Win7, Win8 and Win
 Server and Active Directory</li>\r\n<li>All MS Office products</li>\r\n<li>Email configuration</li>\r\n<li>TCP/IP, DNS, DHCP, SMTP, FTP, HTTP</li>\r\n<li>Cisco firewall, routers and
 switches</li>\r\n</ul>\r\n<br /><br />',
                                   u'job_title': u'IT Support Specialist',
                                   u'num_vacancies': u'1',
                                   u'payment_period': u'Per Month',
                                   u'post_date': u'2014-01-30 17:01:54',
                                   u'salary_maximum': u'2500',
                                   u'salary_minimum': u'2000',
                                   u'views': u'2317'},
                      u'_type': u'user'},
                     {u'_id': u'AV2a3iIGBh_ZpRw8ku0m',
                      u'_index': u'my-index',
                      u'_score': 0.3588409,
                      u'_source': {u'career_level': u'Experienced (Non-Manager)',
                                   u'city': u'Cairo',
                                   u'currency': u'Egyptian Pound',
                                   u'experience_years': u'5+',
                                   u'id': u'f5f7cb6e',
                                   u'job_category1': u'IT/Software Development',
                                   u'job_category2': u'Select',
```

**Platform and Plugin Updates**
PyCharm Community Edition is ready to update.

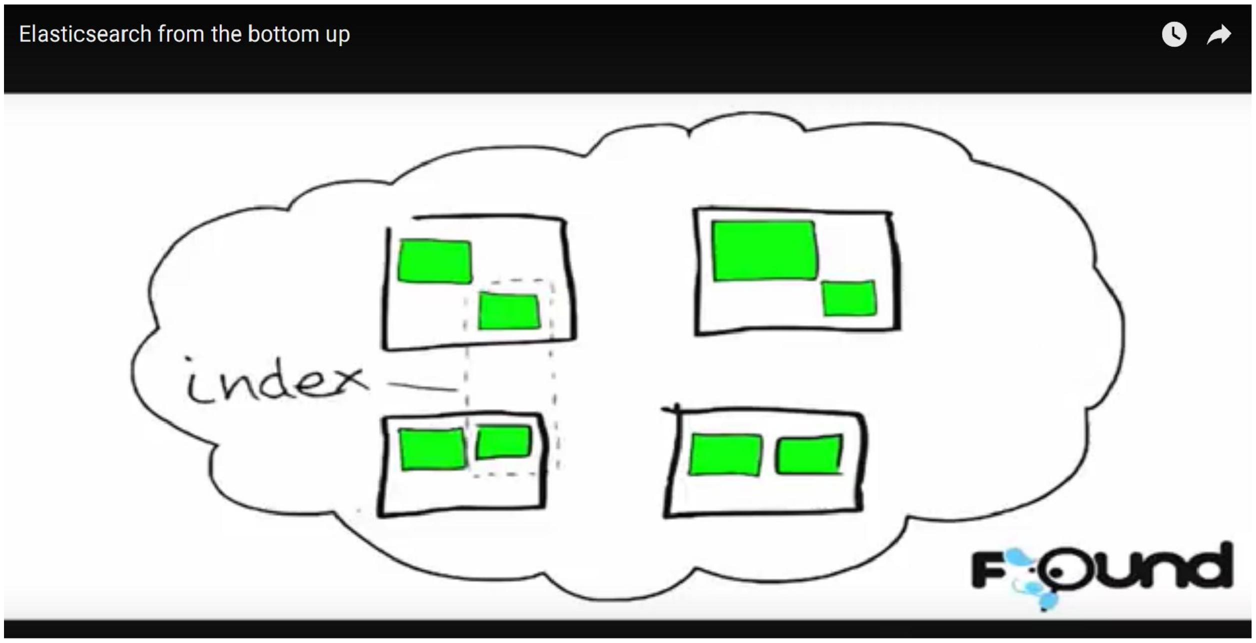
3362 chars, 24 line breaks 30:42 LF UTF-8 Git: master

## How the system is expected to be used as a product:

- Each user will have a name and some other staff (actually a full profile), and an ID.
- When the user is asking for more jobs that maybe interesting for him/her, we will call the “Wuzzuf indexer” with this ID.
- That indexer will search for more like this jobs, and return list of the jobs’ IDs.
- The UI will present the full profile of these jobs.

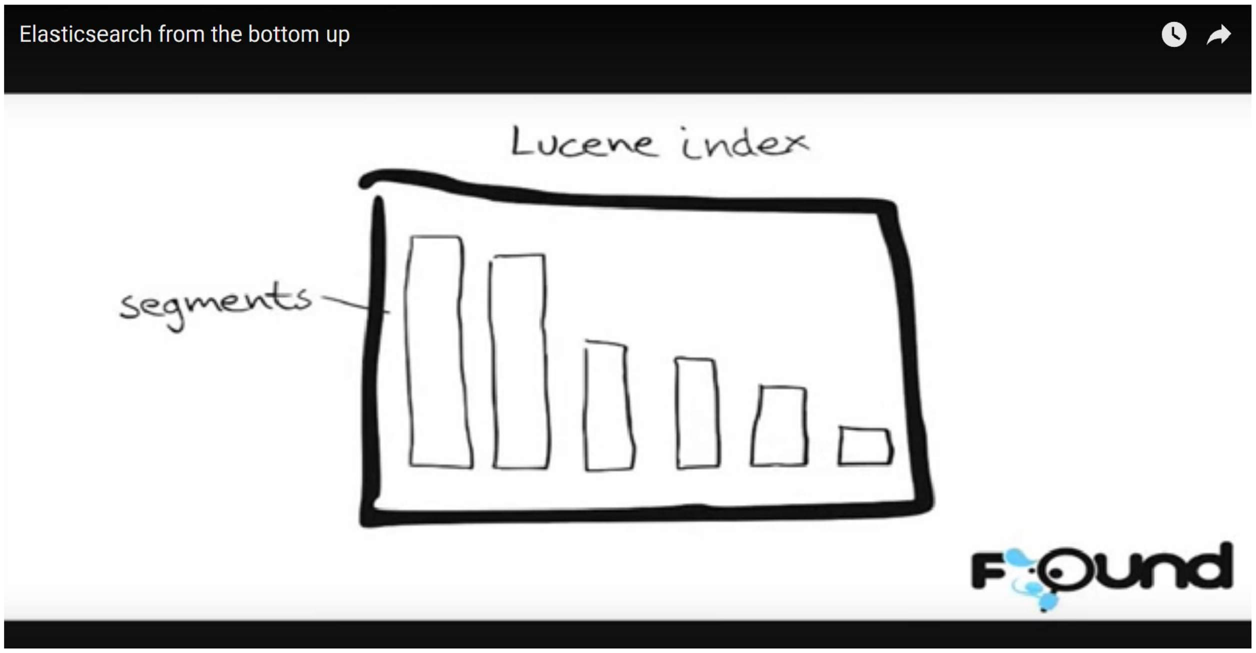
## More About ElasticSearch:

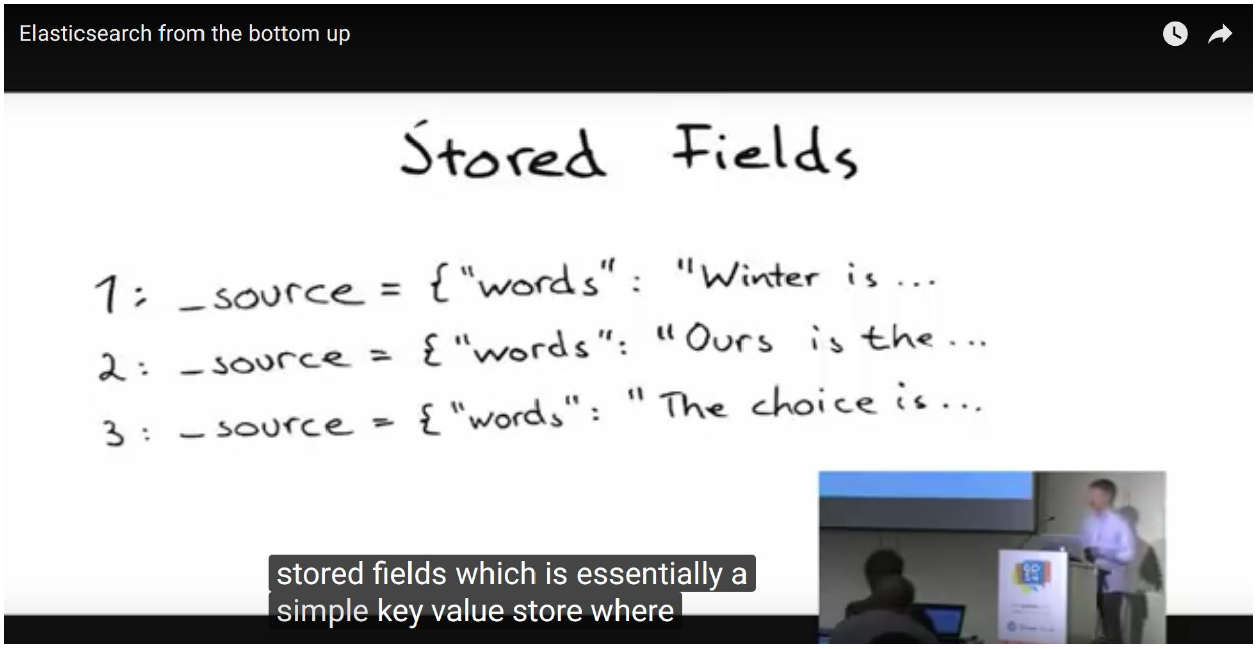
### 1. How does Elasticsearch work:



- We can think of Elasticsearch as blocks of Lucence index search blocks (shards), so the way elasticsearch works is the same that lucence index works but it's more organized.

- The lucence search works using the inverted index and the stored field and document values, and wrap all these up in something called segment, then when you give it a query, it gets the result from all the segments and merge them together.





Elasticsearch from the bottom up

Stored Fields

1: _source = {"words": "Winter is ...
2: _source = {"words": "Ours is the ...
3: _source = {"words": "The choice is ...

stored fields which is essentially a
simple key value store where

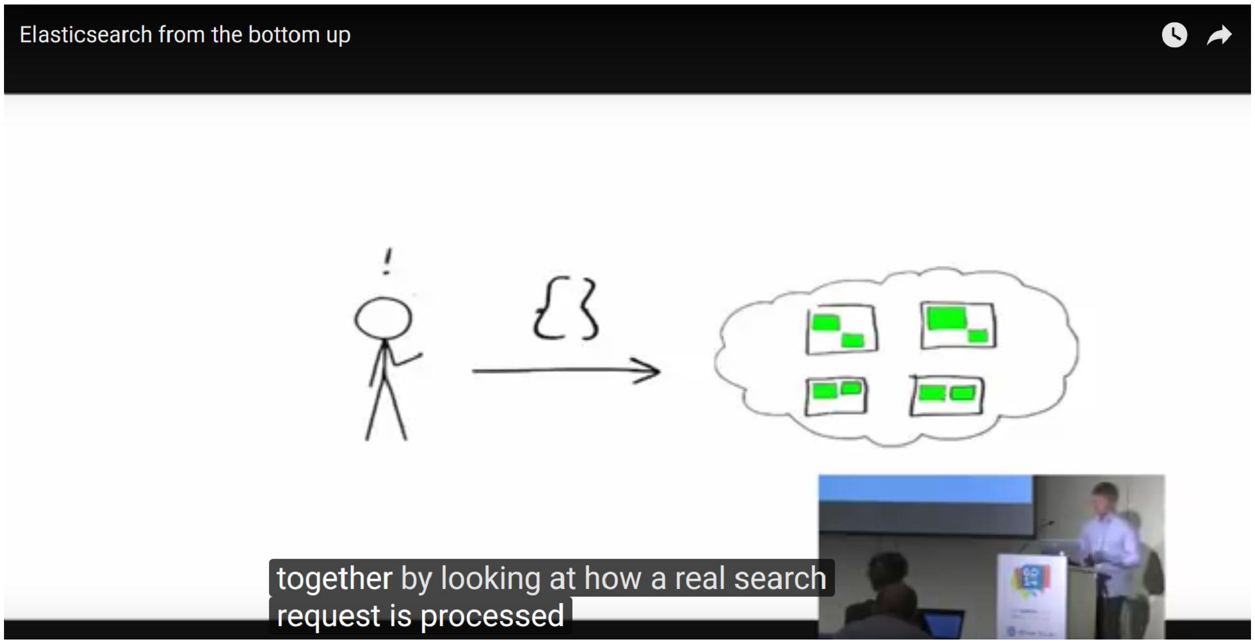
Pictures from (https://www.youtube.com/watch?v=PpX7J-G2PEo).

## 2. How does Elasticsearch's indexing work:

- The term "index" is used in a lot of contexts, with different meanings. You *index* documents, to an Elasticsearch *index*. The Elasticsearch index has shards, which are *Lucene* indexes. And those have inverted indexes. This can get confusing.
- An important insight is that, conceptually, an Elasticsearch index with two shards is exactly the same as two Elasticsearch indexes with one shard each. Ultimately, they are two Lucene indexes. The difference is largely the convenience Elasticsearch provides via its routing feature. It is possible to achieve the same "manually" having just single shard indexes. (Not that you should!)
- The important part is realizing that an Elasticsearch index is an abstraction on top of a collection of Lucene indexes, through the concept of shards.

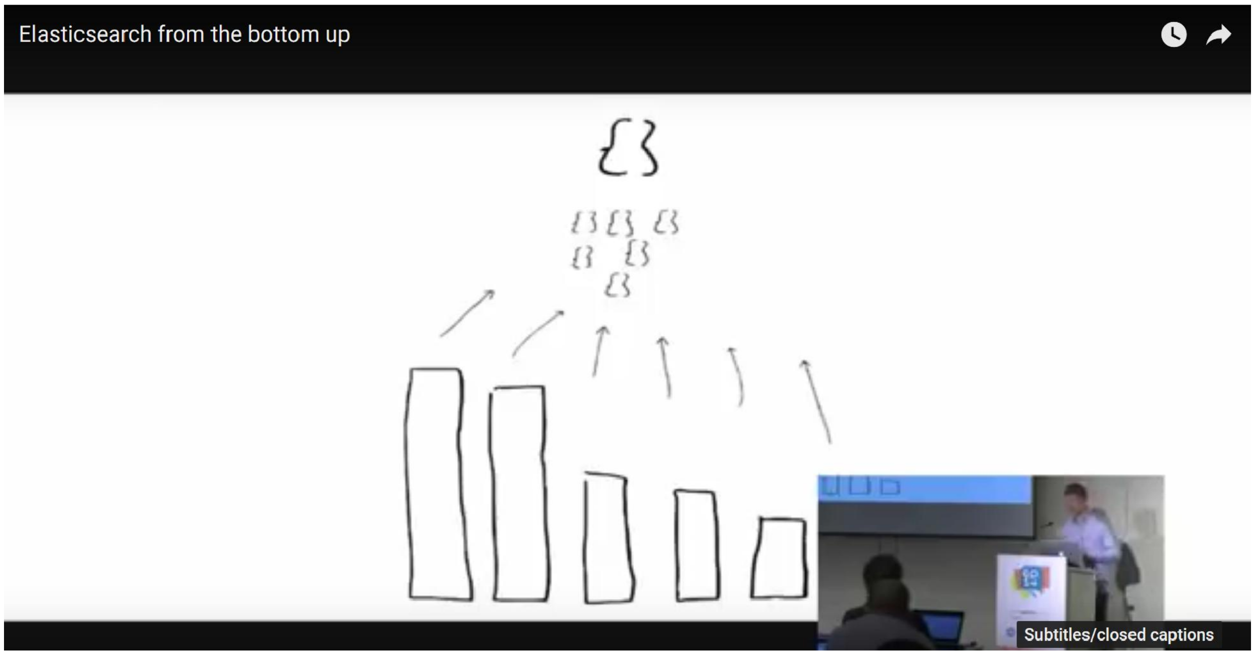
Source(https://www.elastic.co/blog/found-elasticsearch-top-down)

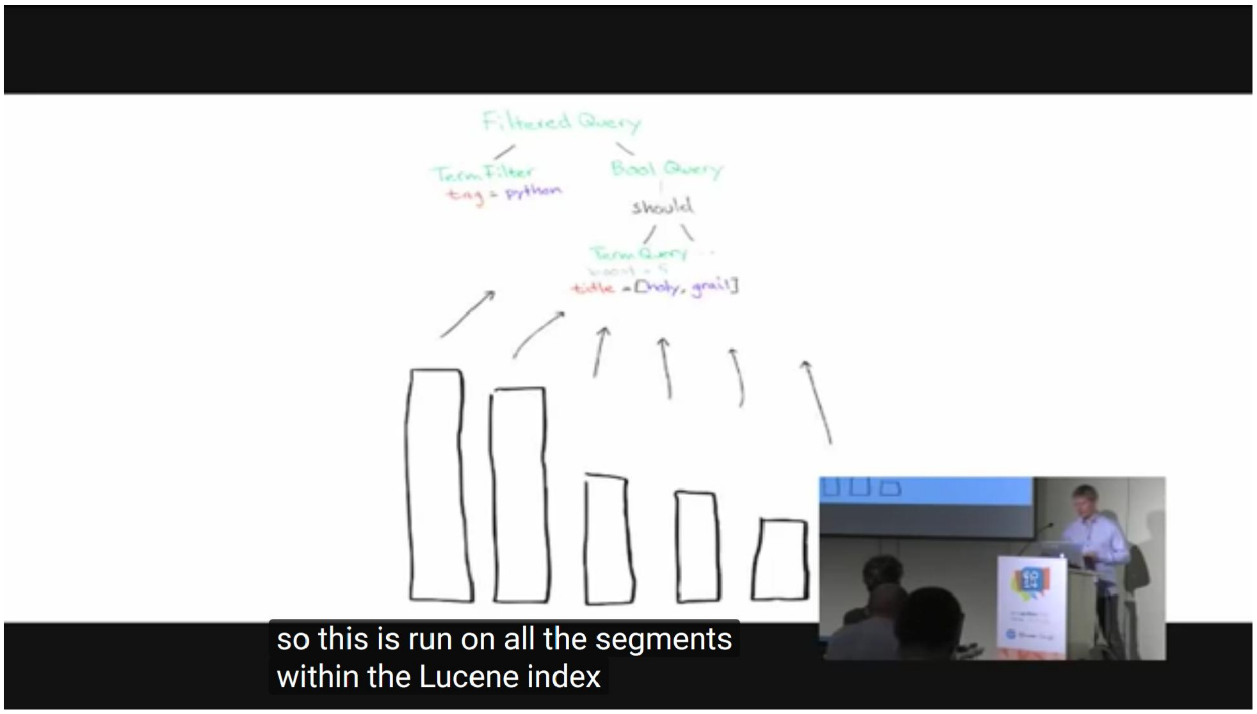
## 3. How does the search requests work:



together by looking at how a real search request is processed

- The request is targeted to any node, that node becomes the coordinator for that search query and it decides which node to work on this query.
- The query is targeted to the right node, then the result is retrieved from the segments in that node, and merged back.





so this is run on all the segments within the Lucene index

## 4. How the engine retrieves the similar documents:

After every shard has provided its contribution to the results, the coordinator will merge them. Specifically, there are two things it needs to find out:

- The true top 10 hits. The shards will provide the IDs of up to 10 documents, and their scores.
- An approximation of the top 10 authors. The shards have provided the counts for up to 100 authors each. If we did not do this, and only requested the top 10 per shard, what would happen if one of the true top 10 authors were actually the 11th in one of the shards? It would not have been submitted as a candidate, and caused that particular outer to have fallen out. This is still possible, if one of the true top 10 authors is actually the 101st on one of the shards, but it should be less likely. To have complete accuracy, Elasticsearch needs to gather the counts for *all* authors for *every* shard. This can be prohibitively expensive to do, so trading accuracy for speed is common in this case.

The merge process will determine the true top 10 hits, then reach out to the shards that host the documents and ask for the entire document. Whether this extra step is helpful or an optimization that ends up adding to the overall latency depends on your use case. If you have a big number of shards and rather big documents, it's probably worthwhile to do it in two rounds. If you have tiny documents and few shards, you can consider the `query_and_fetch` search type. As always, test and verify.

Source(https://www.elastic.co/blog/found-elasticsearch-top-down)

## 5. How does “match” query work:

- It searches for the exact word in the inverted index.

## 6. How does “more like this” work:

- Suppose we wanted to find all documents similar to a given input document. Obviously, the input document itself should be its best match for that type of query. And the reason would be mostly, according to Lucene scoring formula, due to the terms with the highest tf-idf. Therefore, the terms of the input document that have the highest tf-idf are good representatives of that document, and could be used within a disjunctive query (or OR) to retrieve similar documents. The MLT query simply extracts the text from the input document, analyzes it, usually using the same analyzer as the field, then selects the top K terms with highest tf-idf to form a disjunctive query of these terms.
  (source: https://www.elastic.co/guide/en/elasticsearch/reference/1.6/query-dsl-mlt-query.html)

## 7. ElasticSearch VS Solr:

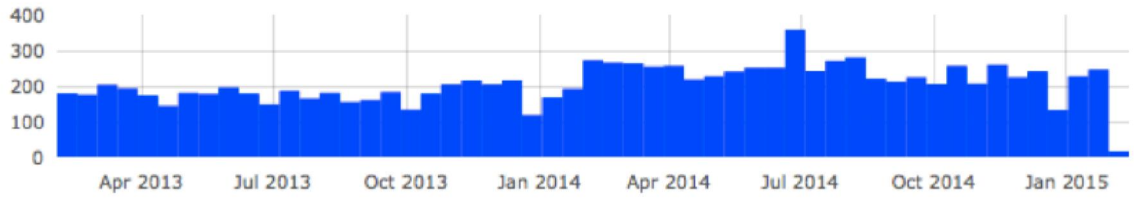
- Elasticsearch is the most popular enterprise search engine followed by Apache Solr, also based on Lucene.
  source(https://en.wikipedia.org/wiki/Elasticsearch).

- Elasticsearch, traditionally being easier to get started with, made it possible for anyone to start using it out of the box, without too much understanding of how things work. That’s great to get started, but dangerous when data/cluster grows.
- Elasticsearch, lending itself to easier scaling, attracts use cases demanding larger clusters with more data and more nodes.
- Elasticsearch is more dynamic – data can easily move around the cluster as its nodes come and go, and this can impact stability and performance of the cluster.
- While Solr has traditionally been more geared toward text search, Elasticsearch is aiming to handle analytical types of queries, too, and such queries come at a price.

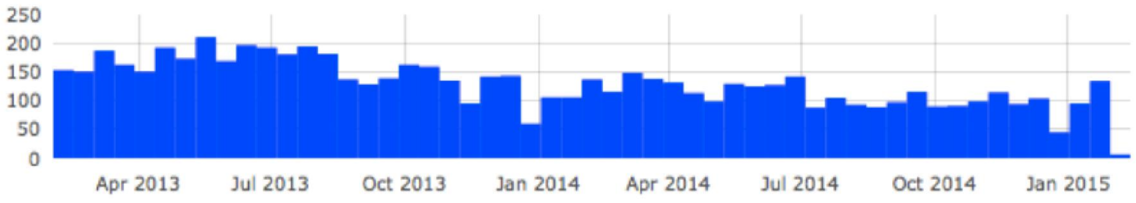
source(https://sematext.com/blog/2015/01/30/solr-elasticsearch-comparison/)

- I think Elasticsearch is the most suitable for Wuzzuf website as it’s better in query search.
- Elasticsearch is more popular:

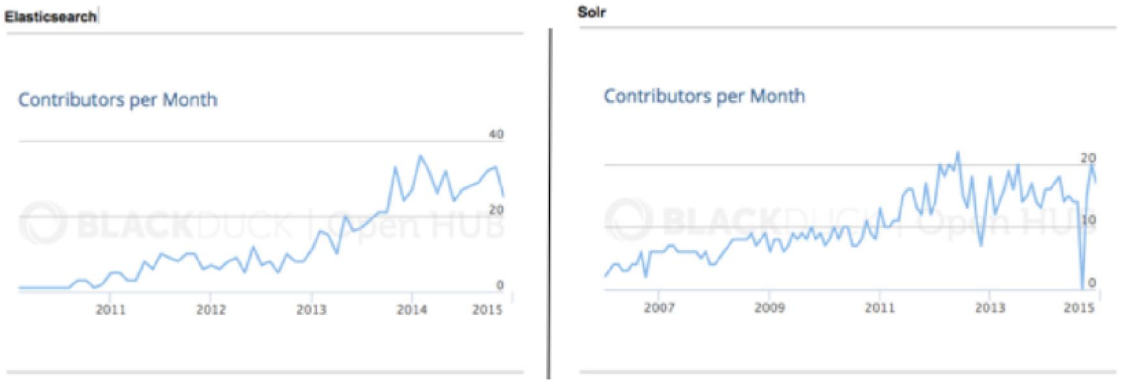
**Elasticsearch user mailing list traffic: 36,127** (source: Search-Lucene)



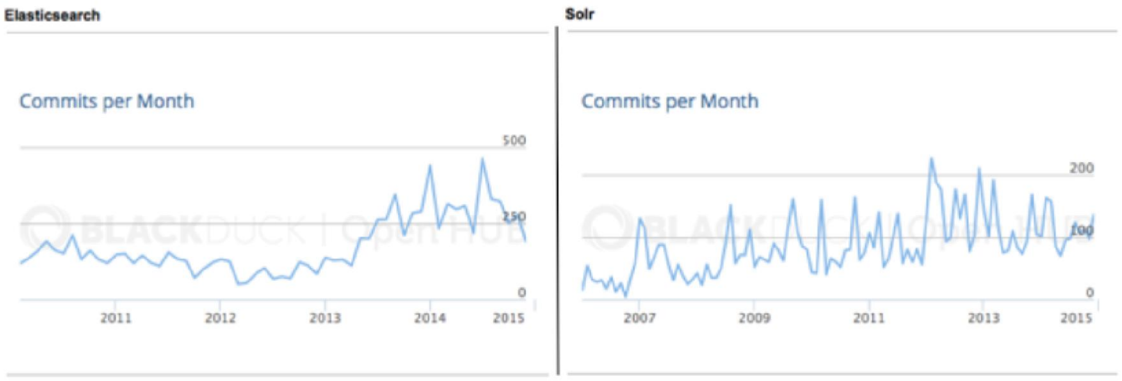
**Solr user mailing list traffic: 24,288** (source: Search-Lucene)



**Elasticsearch vs. Solr Contributors** (source: Open Hub) *click to enlarge*



**Elasticsearch vs. Solr Commits** (source: Open Hub) *click to enlarge*



## 8. My opinion in ElasticSearch VS my opinion in Solr:

- Well, I don’t see or feel much difference, but I loved that elasticsearch is easier to use and easier to be scaled, and much popular, so if you faced a problem, there is much probability that you will get the solution.

## Future work and how can we enhance the search engine:

- The code is not OOP, it even doesn’t have functions, it’s a simple one script, but this can be modified easily, I didn’t bother myself doing this as I was concentrating on investigating the new concepts more than the code maintainability.

- We can use all the jobs that the specific user is interested in and get all the “more like that” jobs, then we can merge them, or rank them based on the number of mentioning.

- Instead of having the Json objects printed in the console, we can have only the indices of them printed or saved in a specific file, and if the user wanted to get them, we shall retrieve the indices from the file and print the corresponding jobs.

- Don’t use the whole features of the job, and don’t store them all too, just use the important features.

- We can run some ranking system on the returned recommended jobs, to sort them in relevance to similarity to the user’s interest.

- We can use more than one cluster or more than one node to make it faster in indexing the data or mapping it.

## References:

**Task link(s):**

- https://docs.google.com/document/d/1nzOYOfC4Min7D2aTvaR6fnVDwVrQ1t2lH8altGV7F6A/edit
- https://www.kaggle.com/WUZZUF/wuzzuf-job-posts

**Elasticsearch install/get familiar/basic knowledge/curl:**

- https://en.wikipedia.org/wiki/Elasticsearch
- https://www.elastic.co/
- https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html
- https://www.elastic.co/webinars/get-started-with-elasticsearch
- https://www.elastic.co/guide/en/elasticsearch/reference/current/_cluster_health.html
- https://dzone.com/articles/elasticsearch-getting-started
- http://elasticsearch-py.readthedocs.io/en/master/api.html
- https://logz.io/blog/elasticsearch-tutorial/

- https://tests4geeks.com/elasticsearch-tutorial/
- http://www.tutorialspoint.com/elasticsearch/
- http://joelabrahamsson.com/elasticsearch-101/
- http://www.elasticsearchtutorial.com/elasticsearch-in-5-minutes.html
- http://www.elasticsearchtutorial.com/basic-elasticsearch-concepts.html
- http://www.elasticsearchtutorial.com/
- https://www.youtube.com/watch?v=60UsHHsKyN4&t=224s
- https://www.youtube.com/watch?v=PpX7J-G2PEo
- https://www.youtube.com/watch?v=ksTTlXNLick
- http://www.linkedkeeper.com/detail/blog.action?bid=97
- https://www.slideshare.net/foundsearch/elasticsearch-from-the-bottom-up
- https://www.elastic.co/blog/found-elasticsearch-from-the-bottom-up

**Python API:**

- https://docs.objectrocket.com/elastic_python_examples.html
- https://www.elastic.co/guide/en/elasticsearch/reference/current/api-conventions.html
- https://bitquabit.com/post/having-fun-python-and-elasticsearch-part-1/
- https://qbox.io/blog/python-scripts-interact-elasticsearch-examples
- https://elasticsearch-py.readthedocs.io/en/master/
- https://tryolabs.com/blog/2015/02/17/python-elasticsearch-first-steps/

**Indexing:**

- https://www.elastic.co/guide/en/elasticsearch/reference/current/index-modules.html
- https://marcobonzanini.com/2015/02/02/how-to-query-elasticsearch-with-python/

**The more like this handler:**

- https://www.elastic.co/guide/en/elasticsearch/reference/1.6/query-dsl-mlt-query.html
- http://elasticutils.readthedocs.io/en/latest/mlt.html

**Dealing with the CSV files:**

- https://stackoverflow.com/questions/41573616/index-csv-to-elasticsearch-in-python
- https://www.elastic.co/blog/indexing-csv-elasticsearch-ingest-node

**ElasticSearch VS Solr:**

- https://www.quora.com/How-do-Lucene-ElasticSearch-and-Solr-compare
- https://sematext.com/blog/2015/01/30/solr-elasticsearch-comparison/
- http://solr-vs-elasticsearch.com/
- https://sematext.com/blog/2015/01/30/solr-elasticsearch-comparison/
- https://www.searchtechnologies.com/blog/solr-vs-elasticsearch-top-open-source-search
- https://findwise.com/blog/solr-or-elasticsearch/