Task 2

(Flask-elasticsearch demo)

DEVELOPER’S

GUIDE

Contributor(s):

Sandino C. Cardinoza

Table of Contents

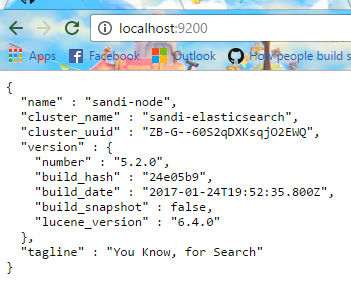
Prerequisites ………………………………………………….. 1

Functions …………………………………………………………. 2 - 3

1 . Prerequisites

Back-end:

* Download Python 2.7.13 (<https://www.python.org/downloads/>)
* For more details go to <https://docs.python.org/2/tutorial/index.html>
* Download pip (<https://bootstrap.pypa.io/get-pip.py>)
* run in cmd: *python get-pip.py*
* Install Flask
* run in cmd: *pip install flask*
* Install Elasticsearch (<https://www.elastic.co/downloads>)
* After downloading elastic search run *bin\elasticsearch.bat* (for windows)
* Run <http://localhost:9200> on your browser and you be able to see something like the image below:



* More details on: <https://www.elastic.co/guide/en/elasticsearch/reference/current/getting-started.html>

Front-end

* JQuery : <http://jquery.com/download/>
* Bootstrap : <http://getbootstrap.com/getting-started/#download>

IDE (depends on your preference)

* PyCharm Community Edition
* Sublime
* Notepad++
* Or Terminal for Linux users (Hardcore!)

2. Functions

Below are found in Views.py file.

Instantiate Elasticsearch (line: 13)

es = Elasticsearch("http://localhost:9200")

Route to home page index.html in the front-end is a form that you will fill out and submit in order to create index (line: 17-19)

@task2.route('/')  
**def index**():  
 **return** render\_template('index.html')

*es.search* collects all Documents in Elasticsearch and display the result in index.html (line: 23-28)

@task2.route('/documentlist')  
**def documentlist**():  
 # get all documents result = es.search(index='', doc\_type='', body={"query": {"match\_all": {}}})  
 **return** render\_template('index.html', result=result, list=list)

Receives POST request from and create (using es.index) a new document in elasticsearch. (line: 47-55)

@task2.route('/save', methods=['POST'])  
**def save**():  
 doc = {'id': request.form['id'],  
 'name': request.form['name'],  
 'power': request.form['power'],  
 'weakness': request.form['weakness']}  
 res = es.index(index=request.form['es\_index'], doc\_type=request.form['es\_type'], body=doc)  
 # return to index.html.  
 **return** render\_template('index.html', result=res['created'])

Receives POST request coming from route *documentlist* displays when Delete button is submitted. This removes each checked document in Elasticsearch and redirect to *documentlist* after. (line: 59-70)

@task2.route('/remove', methods=['GET', 'POST'])  
**def remove**():  
 # delete by index from earlier research  
 # res = es.indices.delete(index='test-index', ignore=[400, 404])  
 # delete by document data\_shard[0] = index , data\_shard[1] = doc\_type, data\_shard[2] = id  
 data = request.form.getlist('data[]')  
 **for** each\_data **in** data:  
 data\_shard = each\_data.split('~')  
 es.delete(index=data\_shard[1], doc\_type=data\_shard[2], id=data\_shard[0], ignore=[400, 404])  
  
 # return to index.html.  
 **return** redirect(url\_for('documentlist'))

3.

Jquery that is executed where delete button is click. Collects checked data (document) and push it into an array called *comicslist* and then do a POST request together with the *comicslist*

*Note: this is found in sripts.js under static task2/static folder (line:7-26)*

$('#remove').click(function (){  
 //bucket for checked ids  
 var strconfirm = confirm("Are you sure you?");  
  
 if(strconfirm == true){  
 var comicslist = [];  
 $('.checkboxes:checkbox:checked').each(function () {  
 if(this.checked && $(this).val() != 'on'){  
 comicslist.push($(this).val());  
 }  
 });  
 //post for the removal of the document  
 $.post('/remove',{'data[]': comicslist}, function(result){  
 console.log(result)  
 });  
 }  
  
 //so id doesn't redirect  
 return false;  
});

Receives POST request coming from route *documentlist* displays when Update button is submitted (can be found inside a dialog modal). This Update all checked document in Elasticsearch base on the input values in the form and redirect to *documentlist*. (line: 74-85)

@task2.route('/update', methods=['GET', 'POST'])  
**def edit**():  
 # from dialog request fields  
 body = {"doc": {'id': request.form['id'], 'name': request.form['name'], 'power': request.form['power'], 'weakness': request.form['weakness']}}  
 # update by document data\_shard[0] = index , data\_shard[1] = doc\_type, data\_shard[2] = id  
 data = request.form.getlist('checkboxes')  
 **for** each\_data **in** data:  
 data\_shard = each\_data.split('~')  
 es.update(index=data\_shard[1], doc\_type=data\_shard[2], id=data\_shard[0], body=body)  
  
 # return to index.html.  
 **return** redirect(url\_for('documentlist'))