Elasticsearch_solution

December 18, 2020

1 Playing with Elasticsearch via REST

In those examples you will see how clients interacts with Elasticsearch unsing REST APIs.

An index is defined using the Create Index API, which can be accomplished with a simple **PUT** command:

[1]:
```
import requests
import json
import pprint

req = requests.put('http://elasticsearch:9200/my_test')
print(req.json())
```

```
{'acknowledged': True, 'shards_acknowledged': True, 'index': 'my_test'}
```

To store a document on elasticsearch you should use **PUT** again, notice that in this example you are specifying a **unique id** of the document.

[2]:
```
document = {'username':'Alice',
            'comment':'I love to see the stars at night'}

requestResponse = requests.put('http://elasticsearch:9200/my_test/doc/
 ↪1',json=document)
pprint.pprint(requestResponse.json())
```

```
{'_id': '1',
 '_index': 'my_test',
 '_primary_term': 1,
 '_seq_no': 0,
 '_shards': {'failed': 0, 'successful': 1, 'total': 2},
 '_type': 'doc',
 '_version': 1,
 'result': 'created'}
```

You can avoid specifying the document id but using **POST** method. In this case elasticsearch create one

[3]:
```
document = {'username':'Maria',
            'comment':'My favorite painting is Starry Night'}
```

```
requestResponse = requests.post('http://elasticsearch:9200/my_test/doc/
↪',json=document)
pprint.pprint(requestResponse.json())
```

```
{'_id': 'FyZld3YBm1RE92eu1Dpj',
 '_index': 'my_test',
 '_primary_term': 1,
 '_seq_no': 1,
 '_shards': {'failed': 0, 'successful': 1, 'total': 2},
 '_type': 'doc',
 '_version': 1,
 'result': 'created'}
```

To obtain a document you need to use **GET** method.

[4]:
```
storedDocument = requests.get('http://elasticsearch:9200/my_test/doc/1')
pprint.pprint(storedDocument.json())
```

```
{'_id': '1',
 '_index': 'my_test',
 '_primary_term': 1,
 '_seq_no': 0,
 '_source': {'comment': 'I love to see the stars at night',
             'username': 'Alice'},
 '_type': 'doc',
 '_version': 1,
 'found': True}
```

But you can also launch queries over elasticsearch, using **GET** method and with a specific json that contain **query** and **match** clauses. Here you can see how you can obtain all documents from an index.

[5]:
```
document = { 'query': {
                'match_all': {}
              }
           }

storedDocuments = requests.get('http://elasticsearch:9200/my_test/
↪_search',json=document)
pprint.pprint(storedDocuments.json())
```

```
{'_shards': {'failed': 0, 'skipped': 0, 'successful': 1, 'total': 1},
 'hits': {'hits': [{'_id': '1',
                    '_index': 'my_test',
                    '_score': 1.0,
                    '_source': {'comment': 'I love to see the stars at night',
                                'username': 'Alice'},
                    '_type': 'doc'},
                   {'_id': 'FyZld3YBm1RE92eu1Dpj',
                    '_index': 'my_test',
                    '_score': 1.0,
                    '_source': {'comment': 'My favorite painting is Starry '
                                           'Night',
                                'username': 'Maria'},
                    '_type': 'doc'}],
          'max_score': 1.0,
          'total': {'relation': 'eq', 'value': 2}},
 'timed_out': False,
 'took': 7}
```

To store a document you can use **Create** or **Index** methods. Create will try to store a document if this id is not already present.

```
[6]: document = {'username':'Alice',
                 'comment':'I love to see the stars and the moon at night'}

     requestResponse = requests.put('http://elasticsearch:9200/my_test/doc/1/
      ↪_create',json=document)
     pprint.pprint(requestResponse.json())
```

```
{'error': {'index': 'my_test',
           'index_uuid': 'WXzbF5xgRpqYiaZyc8CggQ',
           'reason': '[1]: version conflict, document already exists (current '
                     'version [1])',
           'root_cause': [{'index': 'my_test',
                           'index_uuid': 'WXzbF5xgRpqYiaZyc8CggQ',
                           'reason': '[1]: version conflict, document already '
                                     'exists (current version [1])',
                           'shard': '0',
                           'type': 'version_conflict_engine_exception'}],
           'shard': '0',
           'type': 'version_conflict_engine_exception'},
 'status': 409}
```

However, if you use index it directly, the document will be overriten (without _create sufix).

```
[7]: document = {'username':'Alice',
                 'comment':'I love to see the stars and the moon at night'}

     requestResponse = requests.put('http://elasticsearch:9200/my_test/doc/
      ↪1',json=document)
     pprint.pprint(requestResponse.json())
```

```
{'_id': '1',
 '_index': 'my_test',
 '_primary_term': 1,
 '_seq_no': 2,
 '_shards': {'failed': 0, 'successful': 1, 'total': 2},
 '_type': 'doc',
 '_version': 2,
 'result': 'updated'}
```

[8]:
```
storedDocument = requests.get('http://elasticsearch:9200/my_test/doc/1')
pprint.pprint(storedDocument.json())
```

```
{'_id': '1',
 '_index': 'my_test',
 '_primary_term': 1,
 '_seq_no': 2,
 '_source': {'comment': 'I love to see the stars and the moon at night',
             'username': 'Alice'},
 '_type': 'doc',
 '_version': 2,
 'found': True}
```