Архитектура апликације и конфигурација ElasticSearch-а

Управљање дигиталним документима Вукашин Јовић Е2 59/2019

Архитектура апликације

Апликација која ће се развијати ће бити реализована као *web* апликација. Како је она базирана на клијент-сервер архитектури, њу ће чинити три слоја: презентациони, апликативни и слој података. Презентациони слој биће представљен у виду *front-end* апликације и та апликација ће бити развијана коришћењем Angular *framework*-а. Апликативни слој биће представљен у виду *back-end* апликације која садржи сву бизнис логику и она ће бити реализована путем Spring *framework*-а. Слој података представљаће базу података и за потребе пројекта биће коришћена MySQL база. Поред тога, апликација ће користити ElasticSearch за индексирање и претрагу одређених података те ће се сходно томе у пројекту користити ElasticSearch Template и ElasticSearch Repository. Сва комуникација ће се обављати преко REST API-ја.

Дигитална библиотека прихваћених радова биће складиштена унутар ElasticSearch-а како би се могло вршити њено детаљније претраживање. Са друге стране, у бази података биће сачувани подаци о корисницима система, часописима као и радовима. За радове се евидентирају следећа обележја: наслов, подаци о коауторима, кључни појмови, апстракт, примарна научна област и pdf документ за текст рада.

Конфигурација ElasticSearch-а

Како би се користио ElasticSearch, потребно га је претходно покренути. Покретање се врши преко batch фајла након чега он слуша на портовима 9200 и 9300. За потребе пројекта користиће се порт 9200 који служи за комуникацију путем REST-а.

Да би се остварила комуникација Spring апликације и ElasticSearch-а, користиће се Spring Data ElasticSearch модул који нуди Java API и за то је потребно у пројекат убацити одговарајући maven *dependency.* Направи се Java класа која садржи конфигурацију за размену података односно комуникацију са ElasticSearch сервером. Такође прави се и репозиторијум како би се лакше обављале CRUD и остале операције. Како ElasticSearch складишти JSON документе, на класе модела се додаје одговарајућа анотација којом се означава мапирање POJO класе на документе. То уједно говори се како ће они изгледати, односно које атрибуте ће имати, као и начин на који ће се индексирати.

SerbianAnalyzer plugin

С обзиром да ElasticSearch сам по себи нема могућност обраде српских текстова, потребно је у њега укључити plugin који то омогућава. Сходно томе користи се SerbianAnalyzer plugin. Он се користи за претпроцесирање текстова чиме их припрема за индексирање. Писан је за Javu 13, Gradle 6.0 i ElasticSearch 7.4.

Како би се он интегрисао у ElasticSearch, након преузимања потребно га је распаковати и потиционирати се у терминалу у коренски директоријум. Потом је потребно извршити команду „`./gradlew clean build`“. Тиме се креира distribution директоријум на локацији build/distribution унутар кога се налази фајл под називом „`serbian-analyzer-1.0-SNAPSHOT.zip`“. Након тога потребно је позиционирати се у bin директоријум elasticsearch коренског директоријума и извршити команду „`./elasticsearch-plugin install file:<absolute path of distribution archive>`“ где се на месту за путању наводи путања до претходно креираног .zip фајла. Потом се покреће ElasticSearch сервер како би се извршила провера успешности претходних корака. За проверу може се користити следећи пример:

```
curl -H 'Content-Type: application/json' -X PUT -D
'{"mappings":{"properties":{"content":{"type":"text","fields":{"sr":{"type":"text","a
nalyzer":"serbian"},"en":{"type":"text","analyzer":"english"}}}}}}'
http://localhost:9200/tweet
```

Тиме се креира нови „tweet“ документ чије се поље са садржајем анализира подразумеваним анализирањем на енглеском језику али и на српком помоћу уграђеног plugin-a.

Овај plugin је могуће и обрисати тако што се позиционирамо у bin директоријум elasticsearch коренског директоријума и извршимо команду „`./elasticsearch-plugin remove serbian-analyzer`“.

MoreLikeThis функционалност

More like this функционалност проналази документе који су слични са датим сетом докумената. Након избора репрезентативних термина улазних докумената, формира се упит користећи те термине и извршава се. Резултат представља сличне документе по претраженим терминима. Могуће је користити ову функционалност на више начина: тражење докумената који су слични по делу задатог текста, тражење докумената комбиновањем текста и докумената који су већ индексирани и тражење докумената комбиновањем текста, индексираних докумената али и докумената који нису нужно индексирани. Како је у склопу нашег пројекта потребно пронаћи документе на основу унетог текста новог документа, користиће се први начин примене ове функционалности.

Пример једноставног More like this упита:

```
GET /_search
{
    "query": {
        "more_like_this" : {
            "fields" : ["title", "text"],
            "like" : "računari u svetu"
        }
    }
}
```

Геопросторна претрага

ElasticSearch подржава геопросторну претрагу на два начина, а то су geo_point који подржава парове лонгитуде и латитуде и geo_shape поља, која подржавају тачке, линије, кругове, полигоне и слично. Како се у пројекту тражи да се пронађу рецензенти чији је град удаљен више од 100km од градова свих аутора поднетог рада, довољно је користити geo_point односно координате градова аутора, како би се добио град удаљен више од жељене дистанце.

Након што аутори унесу своје адресе вршиће се конверзија стринг адреса у lat/lon парове (латитуде и лонгитуде) како би се могла извршити претрага и те координате се бележе приликом индексирања рада. Како би добили рецензенте из града удаљеног **више** од 100км, можемо узети оне који су у 100км па од свих рецензената научне области одстранити оне унутар дистанце али и можемо промењенити bool оператор да уместо „must“, који враћа све унутар 100km, користимо „must_not“ који враћа све изван 100km.

Пример једне претраге за све изван 100km удаљености и случајно изабраним координатама:

```
GET /rad/_search
{
    "query": {
        "bool" : {
            "must_not" : {
                "geo_distance" : {
                    "distance" : "100km",
                    "autor.lokacija" : { "lat" : 40, "lon" : -70 }
                }
```

```
            }
        }
    }
}
```

Изглед indexing unit-a

Као што је већ речено, индексираће се радови, али и рецензенти како би се могла вршити геопросторна претрага.

Пример рада који се индексира:

```
{
	"id" : 1,
	"casopis" : {
		"id" : 2,
		"naziv" : "Casopis"
	},
	"naslov_rada" : "Novi rad",
	"autor" : {
		"id" : 1,
		"ime" : "Vukasin",
		"prezime" : "Jovic",
		"lokacija" : {
			"lat" : 70,
			"lon" : -40
		}
	},
	"koautori" : [
		{
			"id" : 5,
			"ime" : "Petar",
			"prezime" : "Petrovic",
			"lokacija" : {
```

```
                        "lat" : 90.3,
                        "lon" : 10.432
                  }
            }
      ]
      "kljucni_pojmovi": "internet",
      "naucna_oblast": "informatika i racunarsrvo",
      "sadrzaj": "Tekst iz rada"
}
```


Пример рецензента који се индексира:


```
{
      "id_recenzenta" : 1,
      "lokacija" : {
            "lat" : 47.356,
            "lon" : 4.0135
      }
}
```