Бэкенд разработка на python

Лекция 11

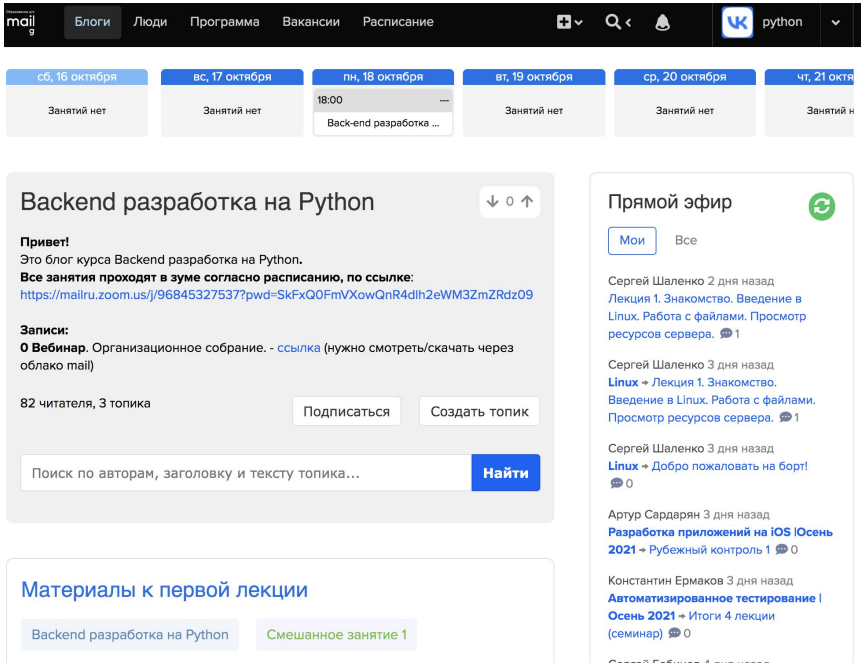
Поисковые движки, Elasticsearch

Кандауров Геннадий

образование

Напоминание отметиться на портале

+ оставить отзыв после лекции

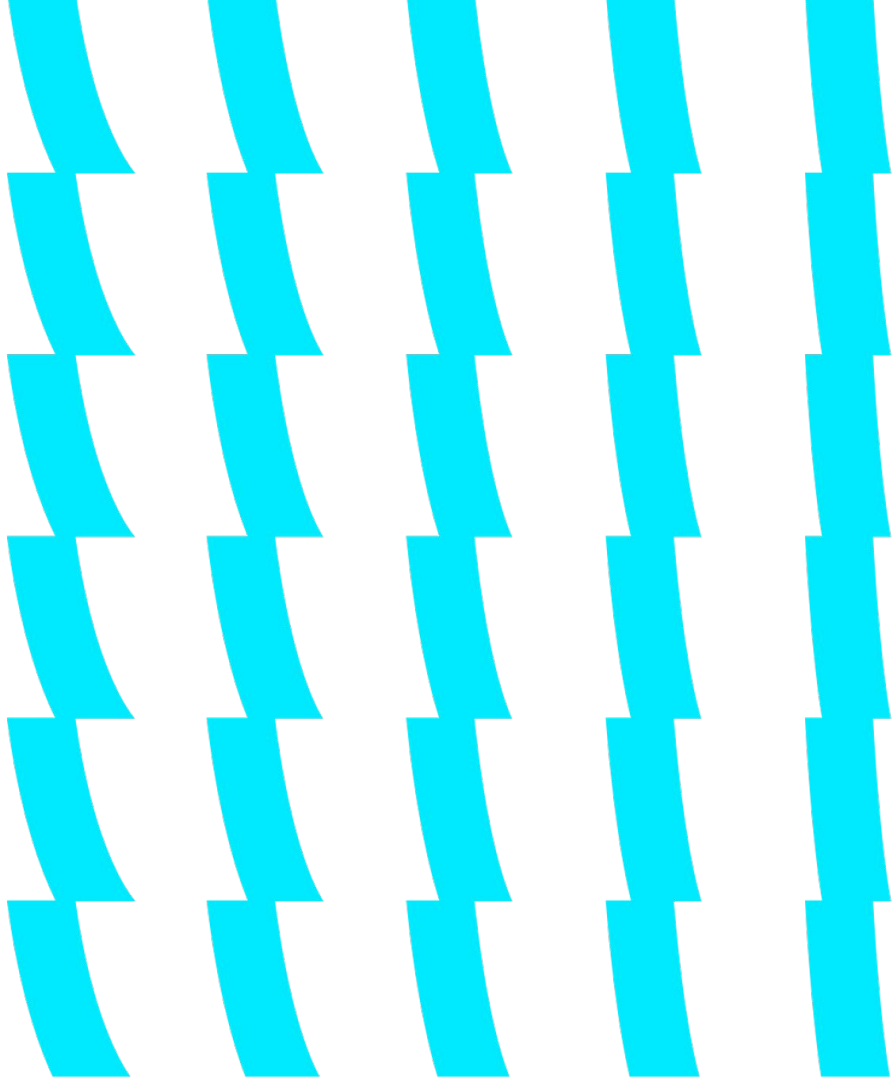


Квиз по прошлой лекции

Содержание занятия

- Поисковые платформы
- Elasticsearch

Поисковые платформы



Поисковые платформы

Основные термины поисковых систем

- **Морфорлогия**
  Раздел грамматики, который оперирует формами слов.
- **Стемминг**
  Приближённый эвристический процесс, в ходе которого от слов отбрасываются окончания в расчёте на то, что в большинстве случаев это себя оправдывает (running -> run).
- **Нечеткий поиск**
  По заданному слову найти в тексте или словаре размера n все слова, совпадающие с этим словом (или начинающиеся с этого слова) с учетом k возможных различий.

Основные термины поисковых систем

- **Лемматизация**
  Точный процесс с использованием лексикона и морфологического анализа слов, в результате которого удаляются только флективные окончания и возвращается основная, или словарная, форма слова, называемая леммой (ran -> run).
- **N-грамма**
  n каких-то элементов. Это более абстрактное понятие.
- **Стоп-слова**

Расстояние Левенштейна

Минимальное количество операций вставки одного символа, удаления одного символа и замены одного символа на другой, необходимых для превращения одной строки в другую.
Цены операций могут зависеть от вида операции

- w(a, b) — цена замены символа a на символ b
- w(ε, b) — цена вставки символа b
- w(a, ε) — цена удаления символа a

Частный случай задачи - **Расстояние Левенштейна**

- w(a, a) = 0
- w(a, b) = 1 при a≠b w(ε, b) = 1
- w(a, ε) = 1

Elasticsearch

- Open source (1690 контрибьюторов на 6 декабря 2021 г.)
- Масштабируемость и отказоустойчивость
- Удобный API (Restfull API)
- Гибкие настройки
- Динамический маппинг
- Геопоиск
- CJK

Где используется Elasticsearch?

- GitHub (поиск репозиториев)
- Uber
- Microsoft (хранилище для MSN)
- stackoverflow
- ebay
- docker (поиск репозиториев)

Elasticsearch: из коробки

- Огромные возможности для поиска документа;
- Около 50 видов агрегаций на все случаи жизни (максимальное, минимальное, среднее);
- Гео-поиск;
- Подсказки (suggester);
- Гибкая работа и настройка всего, что есть в Elasticsearch;
- И ещё много чего!

Elasticsearch: концепты сверху

- Нода
- Кластер
- Шард
- Реплика

Elasticsearch: концепты внутри

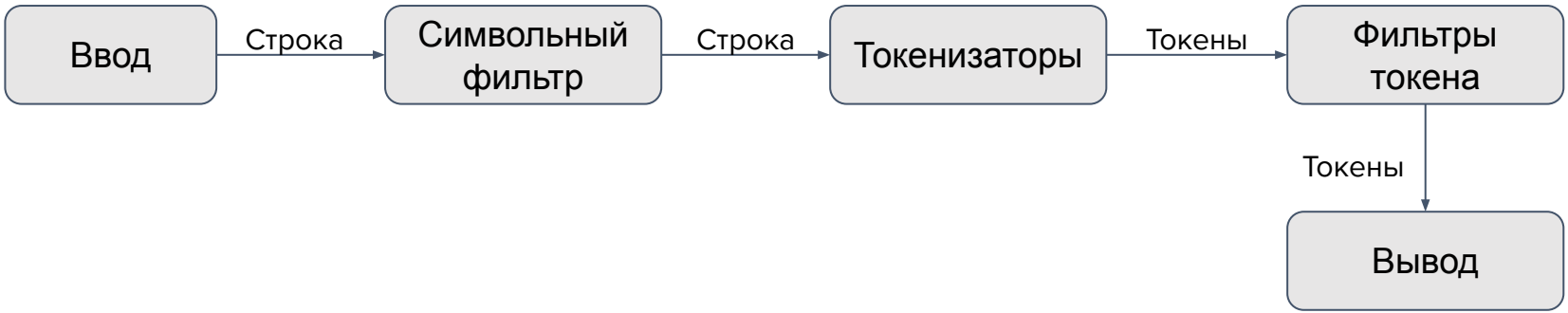
- Индекс
- Тип
- Документ
- Поле
- Отображение (mapping)
- Query DSL

Elasticsearch: концепты внутри

Мир реляционных БД	**Elasticsearch**
База данных (Database)	Индекс (Index)
Таблица (Table)	Тип (Type)
Запись (Row)	Документ (Document)
Колонка (Column)	Поле (Field)
Схема (Schema)	Отображение (Mapping)
SQL	Query DSL

Анализаторы

Цель - из входной фразы получить список токенов, которые максимально отражают её суть.



Пример анализатора

```
PUT /your-index/_settings
{
    "index": {
        "analysis": {
            "analyzer": {
                "customHTMLSnowball": {
                    "type": "custom",
                    "char_filter": ["html_strip"],
                    "tokenizer": "standard",
                    "filter": ["lowercase", "stop", "snowball"]
                }
            }
        }
    }
}
```

Elasticsearch: установка

https://www.elastic.co/downloads/elasticsearch

Ubuntu

Нужно установить Java >= version 7

```
apt install elasticsearch
sudo -i service elasticsearch start
```

http://localhost:9200/

MacOS

```
brew tap elastic/tap
brew install elastic/tap/elasticsearch-full
brew services start elasticsearch
```

Elasticsearch: mappings

```
GET my_index
{
    "mappings": {
        "_doc": {
            "properties": {
                "title": {"type": "text"},
                "name": {"type": "text"},
                "age": {"type": "integer"},
                "created": {
                    "type": "date",
                    "Format": "strict_date_optional_time||epoch_millis"
                }
            }
        }
    }
}
```

Elasticsearch: создание индекса

```
# Создание индекса
PUT http://localhost:9200/blogs
{
    "settings": {
        "index": {
            "number_of_shards" : 5,
            "number_of_replicas" : 3
        }
    }
}
```

Elasticsearch: создание и заполнение индекса

```
# Заполнение индекса пачкой
POST http://localhost:9200/blogs/_bulk
{ "index":{"_index":"blogs", "_type":"posts", "_id":"10"} }
{ "title":"Test1", "description":"First test description" }
{ "index":{"_index":"blogs", "_type":"posts", "_id":"11"} }
{ "title":"Test2", "description":"Second test description" }

# или
POST http://localhost:9200/blogs/posts/
{ "title":"Test3", "description":"Third test description" }
```

Elasticsearch: получение результатов

```
# Получение по id
GET http://localhost:9200/blogs/posts/1

# Поиск по индексам index1,index2,index3 и по полю
GET http://localhost:9200/index1,index2,index3/_search
{
    "query" : {
        "match" : { "title": "test" }
    }
}

# Поиск по определённому полю
GET http://localhost:9200/_search?q=name:central
```

Elasticsearch: синтаксис запросов

+ signifies **AND** operation

| signifies **OR** operation

- negates a single token

" wraps a number of tokens to signify a phrase for searching

***** at the end of a term signifies a prefix query

(and **)** signify precedence

~N after a word signifies edit distance (fuzziness)

~N after a phrase signifies slop amount

Внедряем в приложение: вариант 1

```
from elasticsearch import Elasticsearch
es = Elasticsearch()
es.indices.create(index='my-index', ignore=400)
es.index(index="my-index", id=42, body={"any": "data", "timestamp":
datetime.now()})
{'_index': 'my-index',
 '_type': '_doc',
 '_id': '42',
 '_version': 1,
 'result': 'created',
 '_shards': {'total': 2, 'successful': 1, 'failed': 0},
 '_seq_no': 0,
 '_primary_term': 1}
es.get(index="my-index", id=42)['_source']
```

Внедряем в приложение: вариант 2

```
from rest_framework_elasticsearch import es_views, es_pagination, es_filters
class BlogView(es_views.ListElasticAPIView):
    es_client = es_client
    es_model = BlogIndex
    es_pagination_class = es_pagination.ElasticLimitOffsetPagination
    es_filter_backends = (
        es_filters.ElasticFieldsFilter,
        es_filters.ElasticFieldsRangeFilter,
        es_filters.ElasticSearchFilter,
        es_filters.ElasticOrderingFilter,
        es_filters.ElasticGeoBoundingBoxFilter
    )
```

Внедряем в приложение: вариант 2

```
class BlogView(es_views.ListElasticAPIView):
    ...
    es_ordering = 'created_at'
    es_filter_fields = (es_filters.ESFieldFilter('tag', 'tags'),)
    es_range_filter_fields = (es_filters.ESFieldFilter('created_at'),)
    es_search_fields = ( 'tags', 'title', )
    es_geo_location_field = es_filters.ESFieldFilter('location')
    es_geo_location_field_name = 'location'
```

Внедряем в приложение: вариант 3

```
# documents.py
from django_elasticsearch_dsl import Document
from django_elasticsearch_dsl.registries import registry
from .models import Car

@registry.register_document
class CarDocument(Document):
    class Index:
        name = 'cars'
        settings = {'number_of_shards': 1,
                    'number_of_replicas': 0}
    ...
```

Внедряем в приложение: вариант 3

```
# ... продолжение
class CarDocument(Document):
    ...
    class Django:
        model = Car # The model associated with this Document
        # The fields of the model you want to be indexed in Elasticsearch
        fields = [
            'name',
            'color',
            'description',
            'type',
         ]
```

Внедряем в приложение: вариант 3

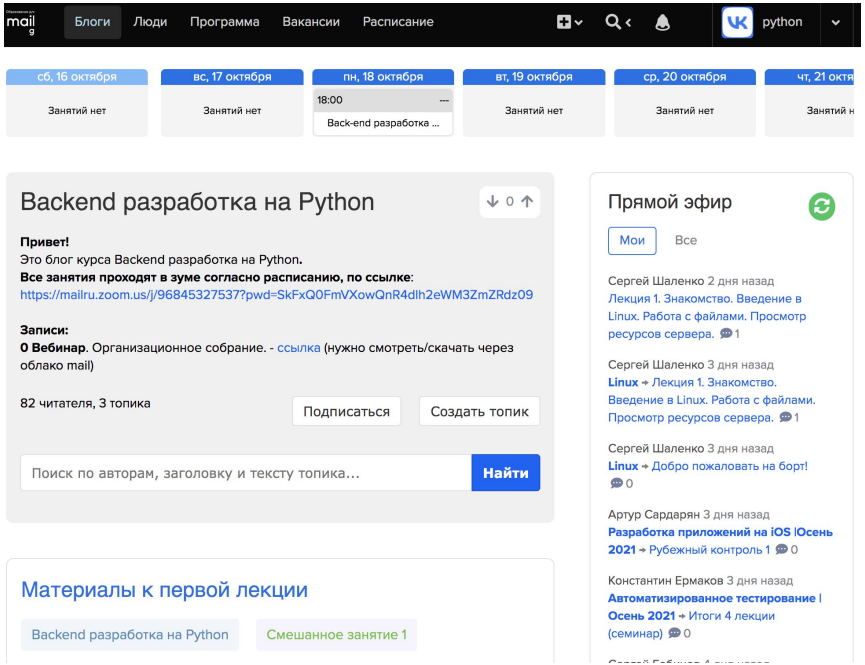
```
./manage.py search_index --rebuild

s = CarDocument.search().filter("term", color="blue")[:30]

qs = s.to_queryset()
```

Домашнее задание по лекции #11

- Написать функцию, которая считает расстояние Левенштейна между двумя словами;
- Развернуть и наполнить тестовыми данными Elasticsearch;
- Реализовать поиск по пользователям, продуктам (сущностям);
- Реализовать метод API для поиска по указанным сущностям и отображения результатов.

Напоминание отметиться на портале Vol 2

+ оставить отзыв



Спасибо за внимание

образование