



**Московский авиационный институт
(национальный исследовательский университет)**

Выпускная квалификационная работа на тему:

«Поиск алгоритмов»

Студент: Шичко Алексей Игоревич
Группа: М8о-4076-18
Научный руководитель: Морозов Александр Юрьевич

Введение.

Алгоритмы в наше время решают судьбу многих компаний. 1 миллисекунда улучшения отклика сервиса может значительно повысить вовлеченность пользователей или клиентов. От вовлеченности в свою очередь, зависит прибыль и отношение самих пользователей.

Представив сервис, в котором программисты не знают и не умеют пользоваться алгоритмами, мы фактически представим себе большие накладные расходы на амортизацию вычислительных мощностей (т.к. их потребуется гораздо больше, чем на самом деле надо).

Проблема

- Требуется возможность поиска по типу алгоритма;
- Необходима возможность сохранять результаты выдачи сервиса;
- Пользователь должен иметь возможность выбрать короткий ответ из списка предложенных.

Актуальность проблемы

- Количество вещей, которое должен запоминать средний программист, с каждым днем растет;
- Программист не всегда знает алгоритм конкретно по имени, но знает что алгоритм должен делать (работать со строками, работать с числами)
- Существующие системы сложны в использовании и не предоставляют необходимого функционала.

Цель

Разработать приложение для поиска алгоритмов по типу алгоритма (например: на графах, на строках) или по имени алгоритма. Должна иметься возможность отслеживать состояние системы и каждой компоненты.

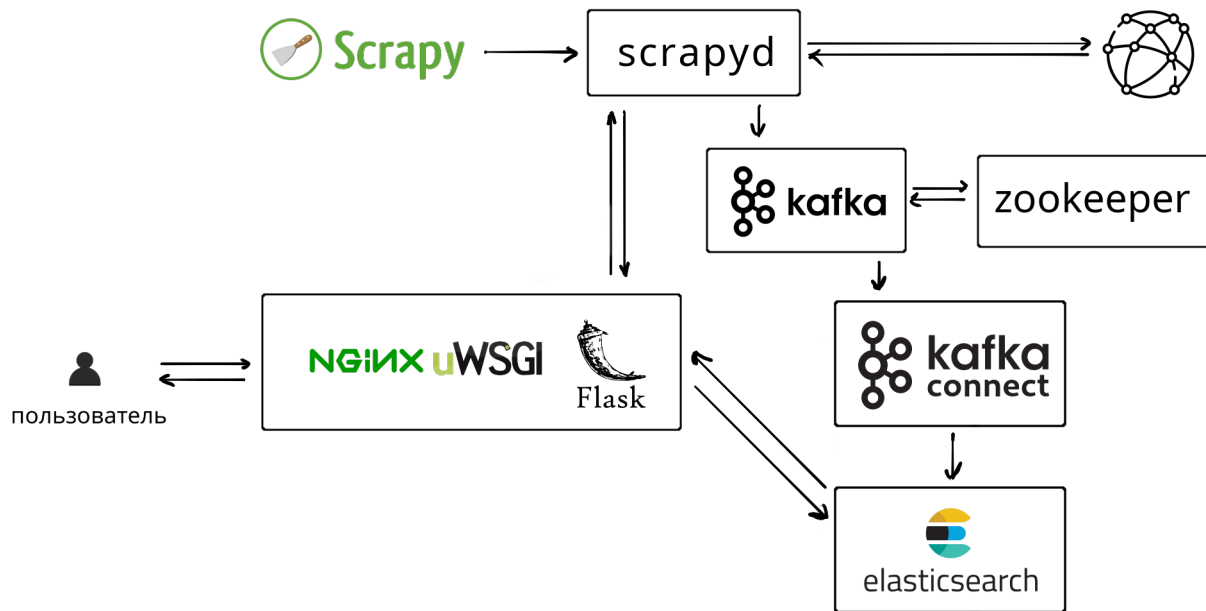
Задачи

- Проанализировать предметную область.
- Проанализировать актуальность поставленной задачи.
- Проанализировать существующие на рынке решения.
- Проанализировать существующие инструменты для написания решения.
- Составить макет решения.
- Протестировать его на подвыборке данных.
- Разработать итоговое решение.

Постановка задачи

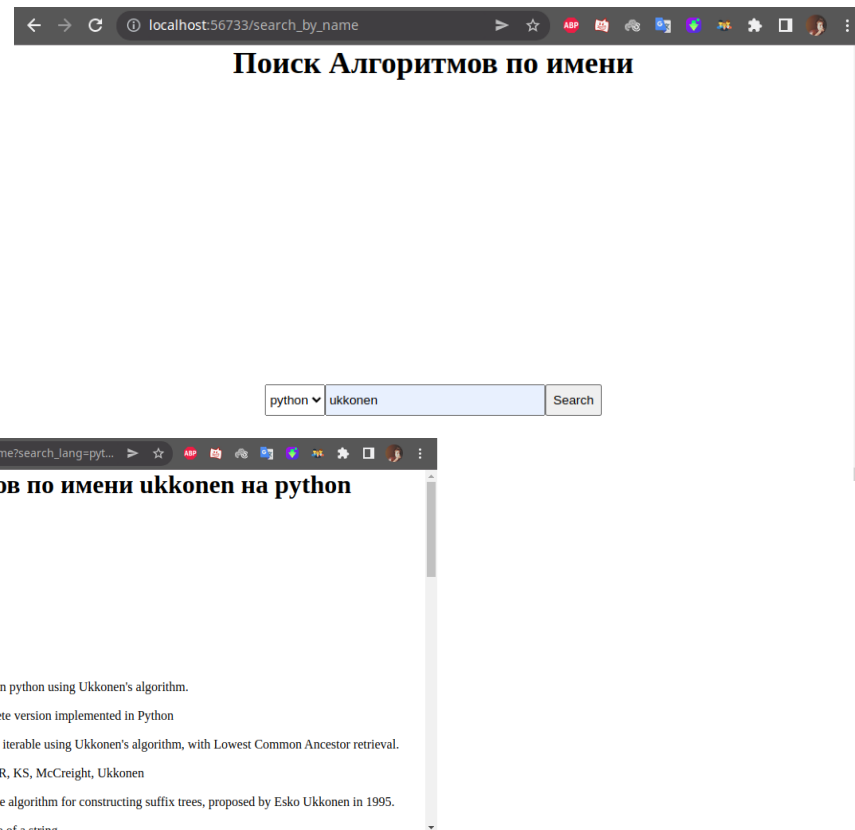
Найти оптимальные с точки зрения масштабирования и эксплуатации компоненты системы. Настроить их коммуникацию с дальнейшей виртуализацией. На основе построенных сервисов написать сервис для поиска алгоритмов на интернет ресурсах без использования API последних.

Результаты: архитектура проекта



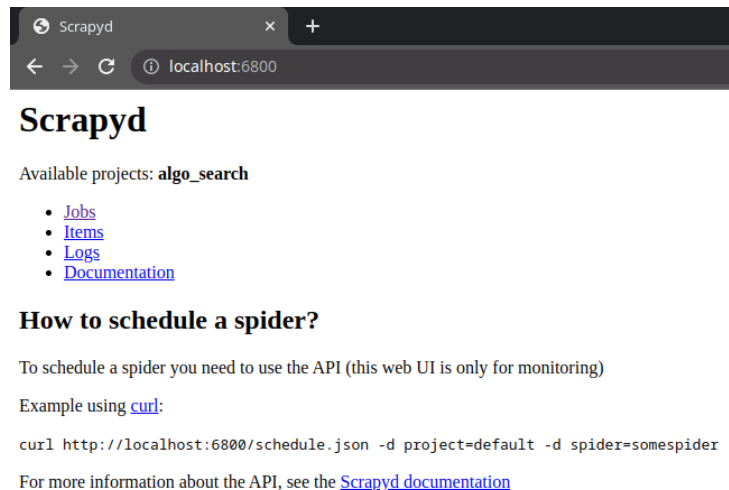
Результаты: компоненты системы: Flask

- Flask: позволяет писать сервера на python, использует jinja как систему шаблонов html. В общей структуре выполняет роль интерфейса пользователя к системе.



Результаты: компоненты системы: Scrapy + Scrapyd

- Scrapy: позволяет в формате “веб пауков” описать что нужно достать из страницы и куда положить. Остальное ложится на плечи самой библиотеки;
- Scrapyd: сервис для разветки “пауков”.



Результаты: компоненты системы: Kafka + Zookeeper

- Kafka: распределенный менеджер сообщений
- Zookeeper: центр синхронизации распределенной системы, необходим для kafka.

```
~/files/vus/diplom/solution(master*) » docker exec -it kafka kafka-topics --describe --topic algorithms-events --bootstrap-server localhost:9092
Topic: algorithms-events      TopicId: tuM0Ai20R_y2jmSuxHTS0Q PartitionCount: 1      ReplicationFactor: 1      Configs:
Topic: algorithms-events      Partition: 0      Leader: 1      Replicas: 1      Isr: 1
```

Результаты: компоненты системы:

Kafka Connect

- Сервис для интеграций с базами данных. Есть возможность написать свою интеграцию.

```
localhost:8083/connectors/sink-elastic-01
1 // 20220511110835
2 // http://localhost:8083/connectors/sink-elastic-01
3
4 {
5   "name": "sink-elastic-01",
6   "config": {
7     "connector.class": "io.confluent.connect.elasticsearch.ElasticsearchSinkConnector",
8     "type.name": "testtypename",
9     "topics": "algorithms-events",
10    "name": "sink-elastic-01",
11    "connection.url": "http://elasticsearch:9200",
12    "key.ignore": "true",
13    "schema.ignore": "true"
14  },
15  "tasks": [
16    {
17      "connector": "sink-elastic-01",
18      "task": 0
19    }
20  ]
21 }
```

Результаты: КОМПОНЕНТЫ СИСТЕМЫ: Elastic Search

- Распределенная документоориентированная СУБД.

```
localhost:9200/algorithms-events/_search
1 // 20220511110925
2 // http://localhost:9200/algorithms-events/_search
3
4 {
5   "took": 4,
6   "timed_out": false,
7   "_shards": {
8     "total": 5,
9     "successful": 5,
10    "skipped": 0,
11    "failed": 0
12  },
13  "hits": {
14    "total": 35,
15    "max_score": 1.0,
16    "hits": [
17      {
18        "_index": "algorithms-events",
19        "_type": "testtypename",
20        "_id": "algorithms-events+0+83",
21        "_score": 1.0,
22        "_source": {
23          "repo_url": "https://github.com/hpatterton/Ukkonen-suffix-tree",
24          "spider_name": "github",
25          "popularity": "0",
26          "about": "\n      Python 3 program to calculate and construct suffix trees using Ukkonen's algorithm\n    ",
27          "topic": "ukkonen",
28          "language": "python",
29          "start_urls": [
30            "https://github.com/search?q=ukkonen&l=Python"
31          ]
32        }
33      }
34    ]
35  }
36 }
```

Заключение

- Инфраструктура написанна полностью в микросервисной архитектуре. Работу каждой отдельной компоненты можно проследить войдя в необходимый сервис либо с помощью команды верхнего уровня системы виртуализации. Результаты инфраструктуры легко достижимы либо из командной строки, либо путем перехода в браузере.
- С помощью написанных микросервисов программисты смогут облегчить себе жизнь и ускорят цикл написания ПО.