

ПОСТРОЕНИЕ ВОПРОСНО-ОТВЕТНОЙ СИСТЕМЫ С ИСПОЛЬЗОВАНИЕМ RAG (RETRIEVAL-AUGMENTED GENERATION)

команда 55

Куратор:

Георгий Панчук tg: [@jmzzomg](#) github: [@joein](#)

Список участников:

Евгений Яковенко – tg: [@Yakovenko_evgenii](#), github: [@yakovenko96](#)

Людмила Теплова – tg: [@lteplova](#), github: [@lteplova](#)

Альберт Тайчинов – tg: [@tayar902](#), github: [@tayar902](#)

Алексей Ятковский – tg: [@BlackR_original](#), github [@Aleksei-la](#)

ПОСТАНОВКА ЦЕЛИ И ЗАДАЧ НА ГОД

Цели:

реализовать составляющие RAG системы

- Принятие вопроса пользователя
- Поиск тематически похожих документов, извлечение из векторной БД
- Передача документа генератору
- Процесс генерации ответа с помощью LLM
- Предоставление ответа



Задачи (сделано):

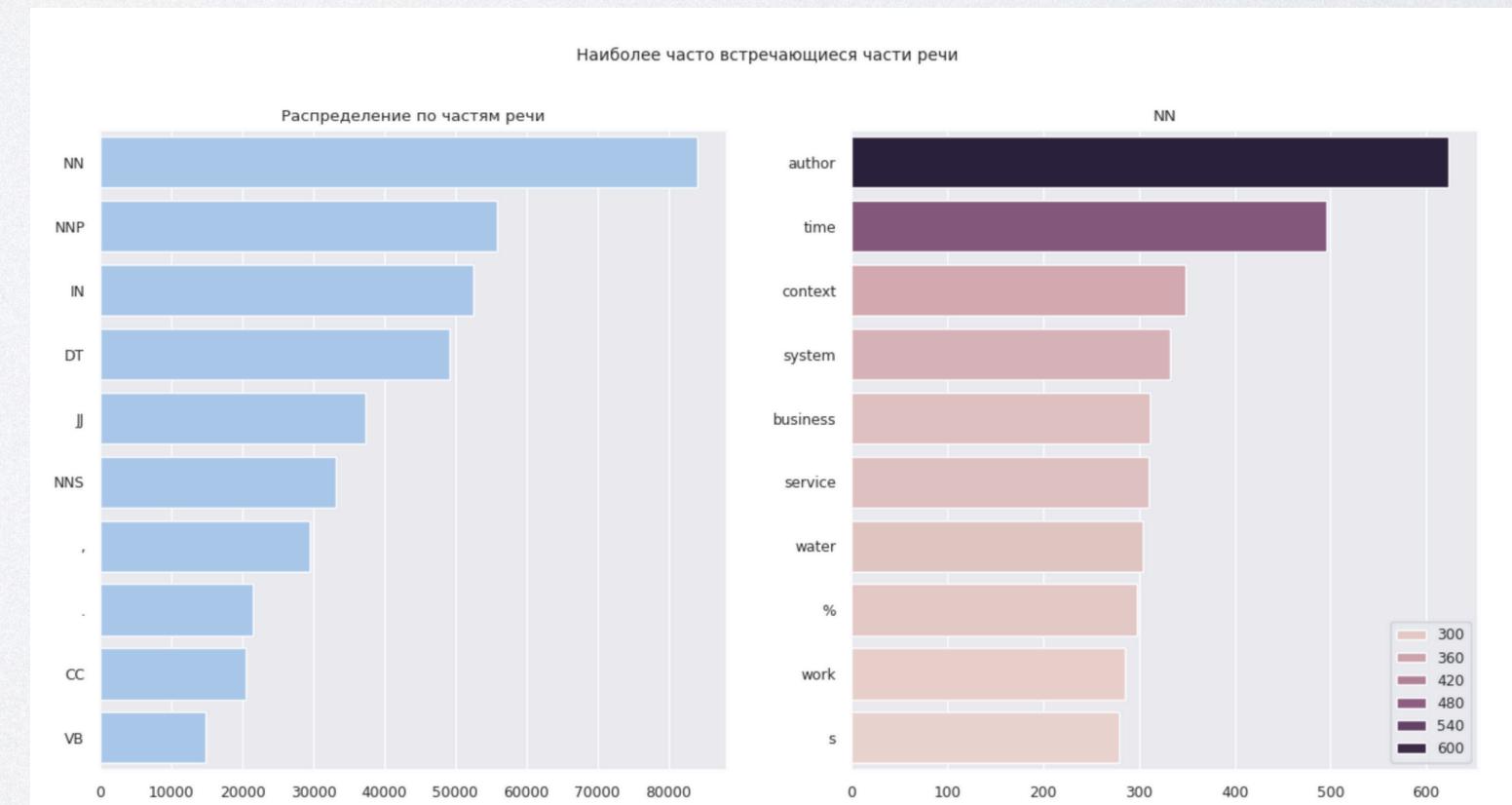
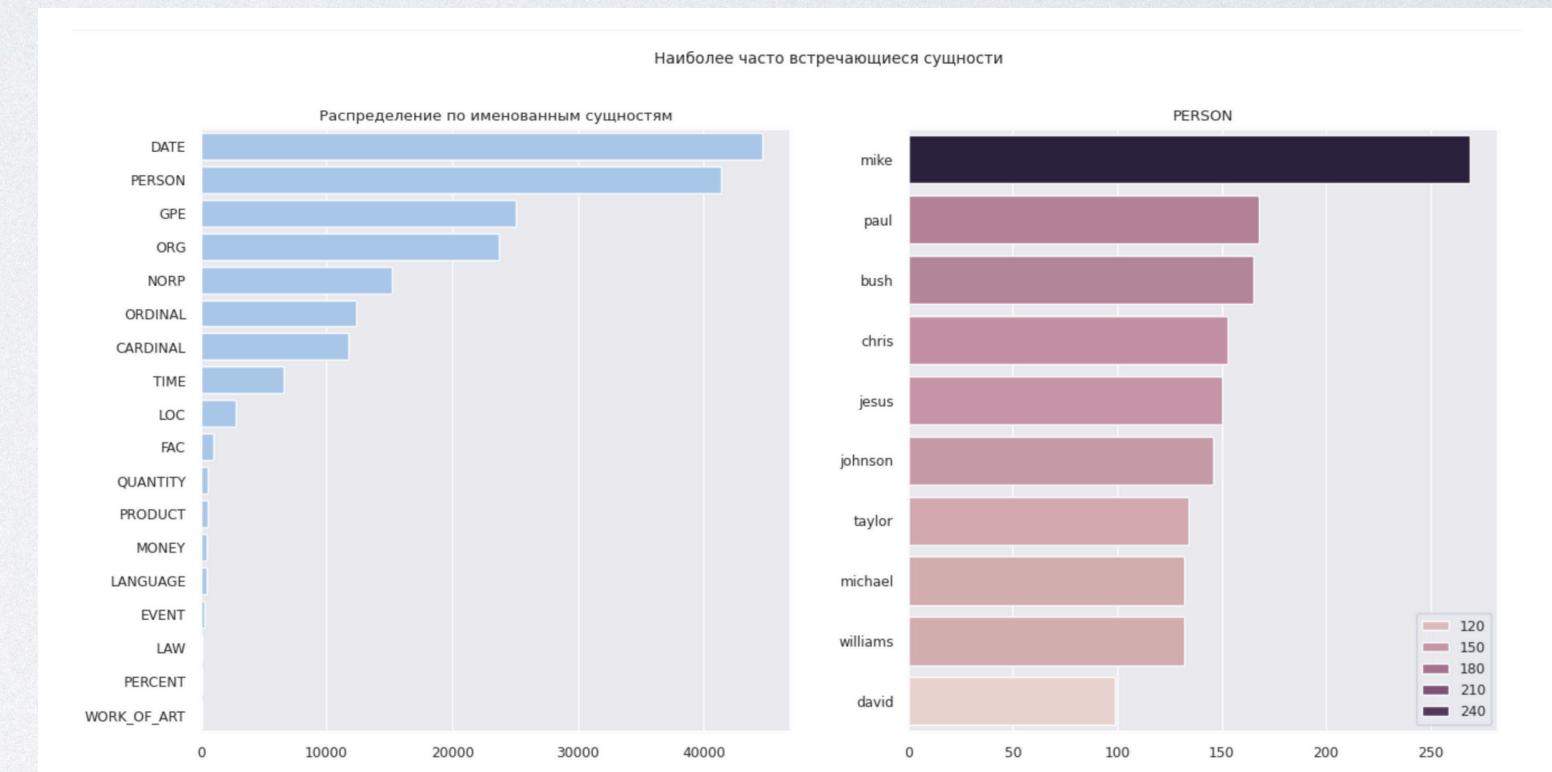
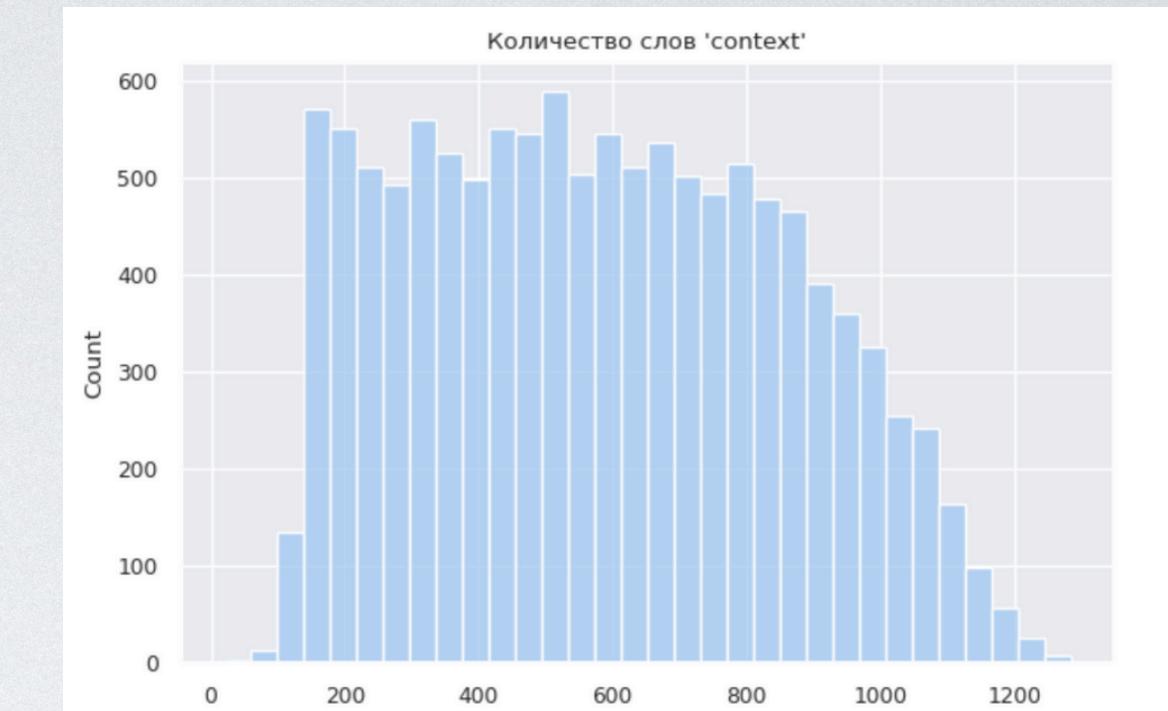
- выбрать и изучить данные
 - выявить аномалии в данных
- выбрать стратегию препроцессинга
- разработать **retrieval** часть приложения
 - выбор эмбеддингов (sparse / dense / hybrid / multivector)
 - выбор метрики
 - выбор алгоритмов поиска
 - выбор хранилища эмбеддингов
 - настройка параметров поиска и извлечения ответов

Задачи (будущая работа):

- разработать **generation** часть приложения
 - выбор метрик для оценки генерации
 - выбор модели для генерации
 - подбор параметров модели
 - оценка результатов и дообучение (опционально)
- объединение общей цепочки RAG и тестирование
- обернуть созданную модель в интерфейсный сервис в виде приложения
 - выбор платформы (telegram|streamlit)
 - разработка приложения и тестирование

ДАННЫЕ

- Датасет для проекта из базы huggingface - [neural-bridge/rag-dataset-12000](#), это контексты из набора - [Falcon RefinedWeb](#) - веб-данных на английском языке, созданный TII и выпущенный по лицензии ODC-BY 1.0. question и answer - вопросы и ответы, сгенерированные GPT-4
- Состоит из 3-х колонок (контекст, вопрос, ответ), 12 000 строк
- Длины текстов от 19 до 1285 слов,
- Обнаружено несколько пропусков и дубликатов по колонке с вопросами, а также несколько строк с текстом на другом языке, в текстах содержится знак перевода строки (\n)
- Было проведено тематическое моделирование, что позволило определить темы текстов: отдых/путешествия/образование, книги/фильмы, рецепты, события в мире искусства, премии награждения, компьютерная индустрия и компьютерные игры, популярные личности, популярные места и локации, болезни, способы лечения
- В текстах большинство составляют существительные и сущности - это имена собственные и даты



МЕТРИКИ

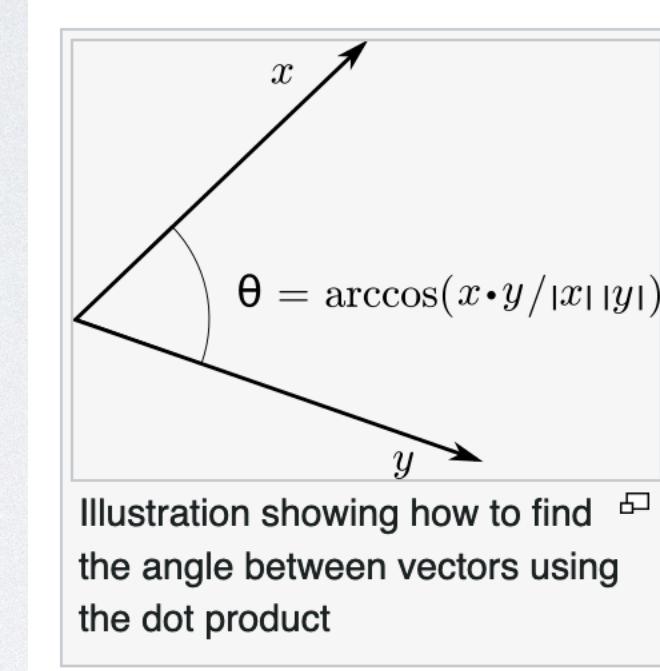
В процессе выбора baseline алгоритма рассматривались метрики:

- Cosine similarity
- Euclidean distance
- Dot product

В результате того, что в качестве baseline решения выбрано преобразование данных в sparse векторы, то по архитектуре извлечения sparse векторов из выбранной БД всегда используется **Dot Product**, которая и стала основной метрикой.

Если в дальнейшем будут использованы другие виды векторов, то будут использоваться и другие метрики.

Геометрическая интерпретация



Dot product (скалярное произведение) - измеряет сходство между двумя векторами путем их поэлементного перемножения и последующего суммирования результатов.

Чем больше значение dot product, тем более похожи векторы

Преимущества:

- простота вычисления
- интуитивная интерпретация
- эффективность реализации

Недостатки:

- зависимость от длины векторов
- чувствительность к масштабу
- не нормализованная метрика

RETRIEVAL ЧАСТЬ BASELINE И МЕТРИКИ

Модель	Точность
Word2Vec	28.5
Fasttext	36.6
glove-twitter-100	13
all-MiniLM-L6-v2	75
tf-idf	85.9

- Выбор способа векторизации - протестированы разные эмбеддинги и замерена точность (брутфорсом для каждого вопроса выполняется поиск по близости и сравнение с оригинальным ответом по индексу)
- Выбор хранилища данных сравнение Pandas и Qdrant (датасет был обогащен до 100 тыс строк и произведены замеры извлечения ответов), лучшее время (0.007с) с использованием Qdrant
- Выбор метрики - Dot product / Cosine similarity / Euclidean



Лучшая Базовая - Tf-Idf, БД Qdrant, Dot product

Precision - 85.9%

MVP НА ОСНОВЕ BASELINE

```
app/
└── fastapi_back/
    ├── .dockerignore
    ├── Dockerfile # создание образа из наших файлов приложения
    ├── LICENSE
    └── README.md
    └── requirements.txt # список зависимостей окружения для fastapi
        └── src/
            ├── __init__.py
            ├── api/
            │   └── v1/
            │       ├── api_route.py # основные эндпоинты
            │       ├── schemas.py # схемы вопросов и ответов
            │       └── tfidf_pretrained.joblib # предобученная модель tf-idf
            ├── logger.py # конфигурация инструмента для логирования
            ├── main.py # основной файл для запуска приложения
            └── qdrant/
                └── load_qdrant.py # интеграции с векторной БД qdrant
            └── tests/
                ├── conftest.py # конфигурация тестов
                ├── full_dataset.csv # датасет для тестов
                └── test_model.py # тесты
    └── grafana/provisioning/datasources/graf_loki.yaml - файл настройки
        мониторинга приложения и
        логов
    └── streamlit/
        ├── config.toml # конфигурация темы приложения
        ├── .dockerignore
        ├── Dockerfile # создание образа из наших файлов приложения
        ├── eda.py # функции для отрисовки графиков, препроцессинг текстов
        ├── project_logger.py # логирование и его установки, функция-обертка
        ├── st_app.py - основной код приложений
        ├── validate_df.py - модуль, отвечающий за валидацию файла
        └── requirements.txt - список зависимостей окружения для streamlit
    └── docker-compose.yml
    └── promtail.yaml - файл настройки сборщика логов
    └── loki.yaml - файл настройки сборщика логов
```

Запуск приложения на локальной машине:

- установить docker в зависимости от ОС - <https://docs.docker.com/compose/install/>
- в папке app прописать команду для сборки и запуска приложения:

docker-compose -p app_rag up

```
PS C:\Users\Asus\data\HSE\project\check5\team_55_rag_qa\app> docker-compose -p app_rag up -d
[+] Running 6/6
  ✓ Container promtail      Started
  ✓ Container grafana        Started
  ✓ Container qdrant         Started
  ✓ Container loki           Started
  ✓ Container backend        Started
  ✓ Container streamlit      Started
```

- открыть <http://localhost:8501/> для входа в приложение
- Для мониторинга приложения и сбора логов:
 - открыть <http://localhost:3000/>
 - подключить datasources loki, по адресу <http://loki:3100/>
 - в разделе explore/loki - ведем мониторинг приложения и логов

АВТОМАТИЧЕСКАЯ СИСТЕМА СБОРА ЛОГОВ GRAFANA

конфигурирование сервиса

The screenshot shows the 'Connections > Data sources' section of the Grafana interface. A new connection named 'loki' is being configured. The 'Connection' tab is active, showing the URL field with 'http://loki:3100/' entered. A red error message 'Please enter a URL' is displayed below the field. The 'Authentication' tab is visible below, showing 'No Authentication' selected. The 'TLS settings' tab is also present at the bottom.

фильтрация для
вывода логов

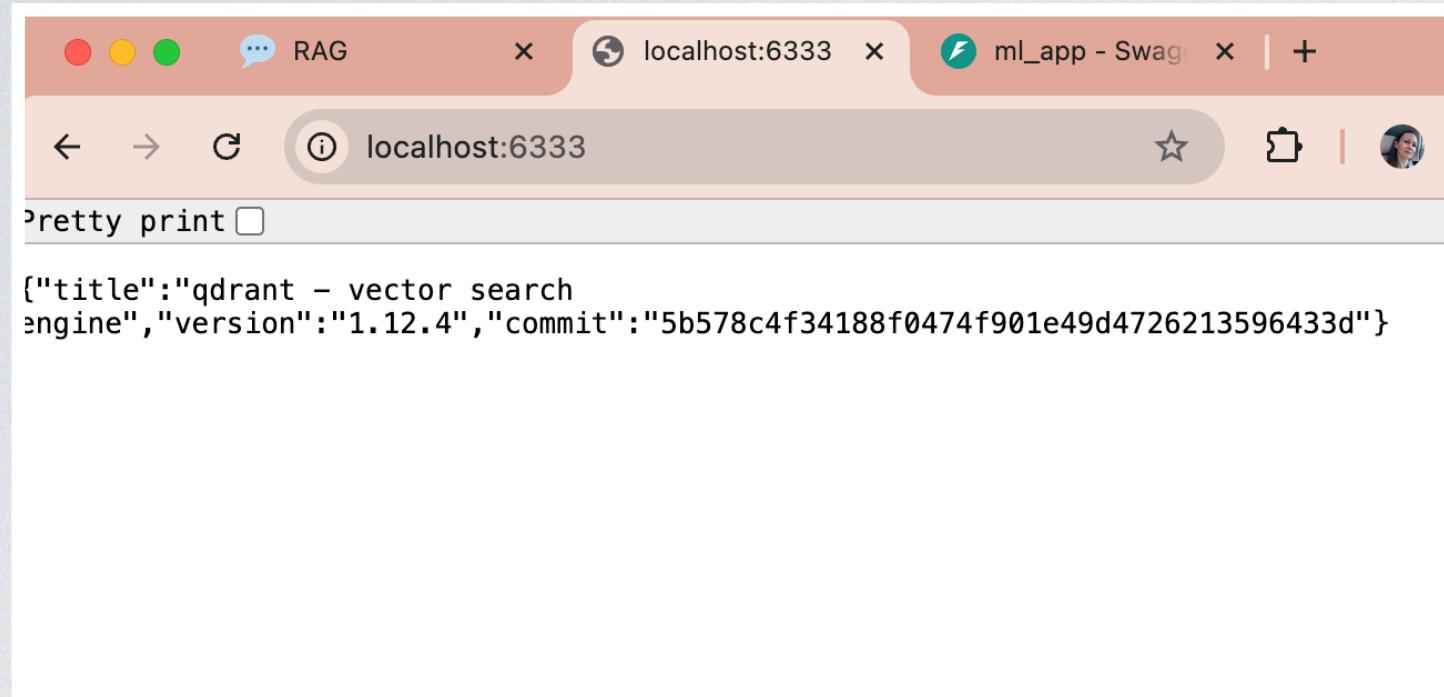
The screenshot shows the 'Queries' panel in Grafana. A query is being built with the following filter: 'job = streamlit'. The 'Label filters' section includes 'backend' and 'streamlit'. A warning message 'parse error : syntax error: unexpected \$end' is shown. The 'Logs' tab is selected at the bottom.

содержимое логов
согласно установкам

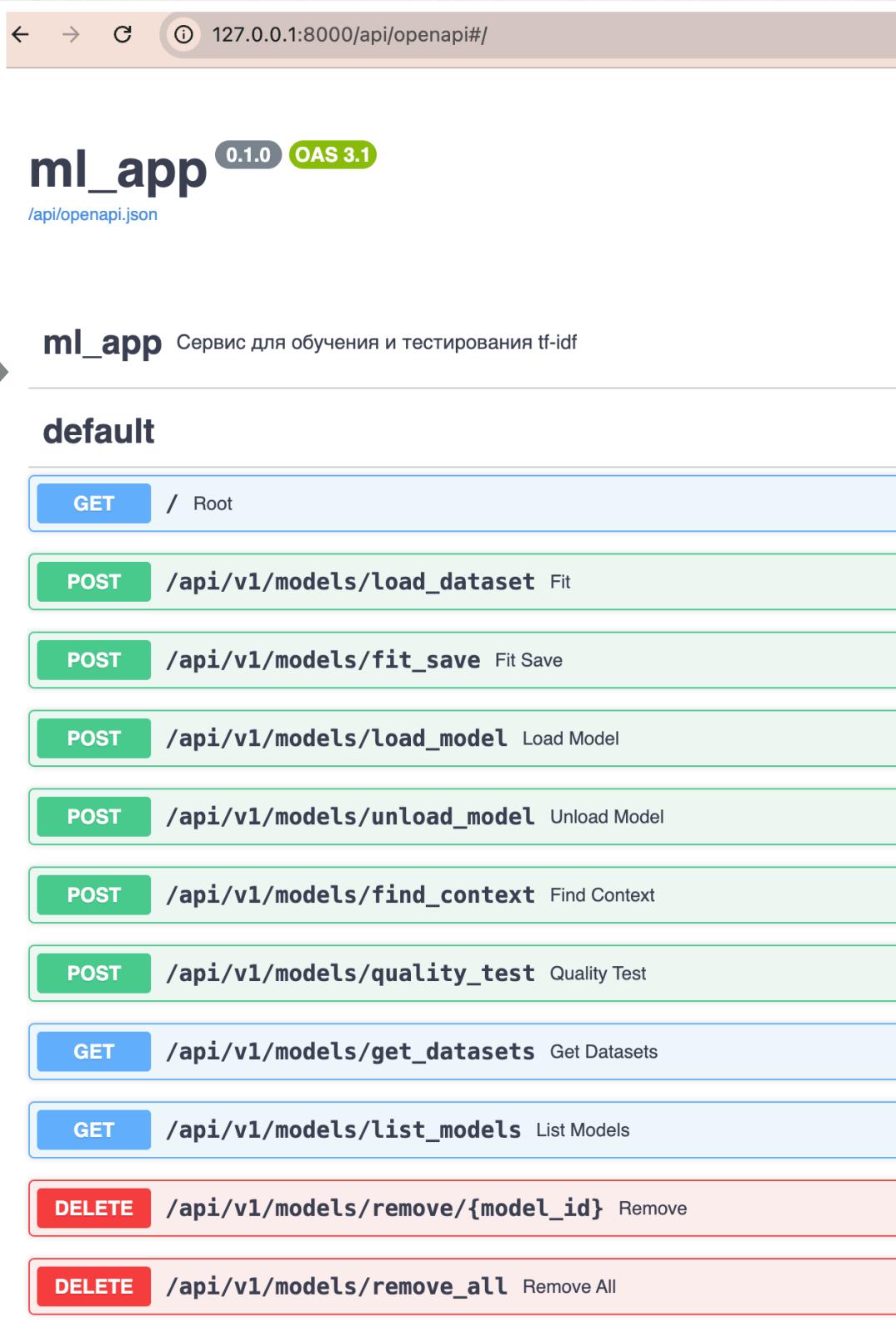
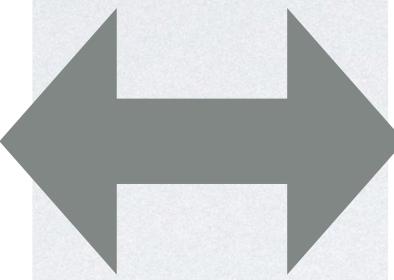
The screenshot shows the 'Logs' panel. At the top, a histogram titled 'Logs volume' displays log entries from 17:00 to 17:55. Below it, the 'Logs' table shows log entries for 'job=streamlit'. The table includes columns for Time, Unique labels, Wrap lines, Prettify JSON, Deduplication, and sorting options (Newest first, Oldest first). The logs listed include:

```
> 2025-01-02 17:54:25.019 2025-01-02 14:54:24.989 - INFO - SUCCESS: удалена модель test1
> 2025-01-02 17:54:25.019 2025-01-02 14:54:24.977 - INFO - SUCCESS: Предикт выполнен успешно
> 2025-01-02 17:54:25.019 2025-01-02 14:54:24.976 - INFO - SUCCESS: Score: 0.20268862, Идентификатор 54
> 2025-01-02 17:54:25.019 2025-01-02 14:54:24.975 - INFO - SUCCESS: овер: research design occupational education university except materials supplied d
ifferent departments university thesis handbook references used permission critique example graduate study bibliographical en
try hamby assessment applied biology chemistry curriculum selected oklahoma schools unpublished doctoral dissertation oklahom
```

MVP НА ОСНОВЕ BASELINE

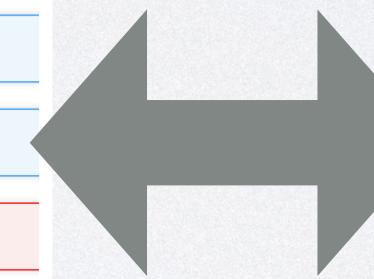


База данных Qdrant



Backend
FastAPI Service

Frontend
Streamlit Service



The screenshot shows the RAG application interface. The top navigation bar includes tabs for "RAG", "localhost:6333", "ml_app - Swagger", and "localhost:8000". The main content area has sections for "Обучение модели для чат-бота на основе RAG" and "Возможности приложения". The "Возможности приложения" section lists:

- загрузка датасета и анализ данных
- препроцессинг данных
- конфигурирование и обучение модели
- получение инференса

The "Загрузка данных" section includes instructions about the CSV format and a file upload input field.

Формат датасета - 3 колонки с текстами
контекст(context), вопрос(question), ответ(answer)

ссылка на [пример датасета](#)

Выберите CSV-файл, максимальный размер файла 200 МБ

Drag and drop file here
Limit 200MB per file • CSV

Browse files

MVP НА ОСНОВЕ BASELINE

Обучение модели для чат-бота на основе RAG

Возможности приложения

- загрузка датасета и анализ данных
- препроцессинг данных
- конфигурирование и обучение модели
- получение инференса

Загрузка данных

Формат датасета - 3 колонки с текстами
контекст(context) , вопрос(question) , ответ(answer)

ссылка на [пример датасета](#)

Выберите CSV-файл, максимальный размер файла 200 Мб

Drag and drop file here
Limit 200MB per file • CSV

Browse files

555.csv 110.0B

Следующие столбцы не являются текстовыми: ['Length1']

Выберите CSV-файл, максимальный размер файла 200 Мб

Drag and drop file here
Limit 200MB per file • CSV

123.csv 0.0B

проверка
загружаемого
файла на
соответствие
требованиям на
стороне frontend

Загрузка данных

Формат датасета - 3 колонки с текстами

контекст(context) , вопрос(question) , ответ(answer)

ссылка на [пример датасета](#)

Выберите CSV-файл, максимальный размер файла 200 Мб

Drag and drop file here
Limit 200MB per file • CSV

Browse files

full_dataset.csv 45.4MB

X

Файл успешно загружен и проверен!

Превью - первые 5 строк:

	context	question
0	Caption: Tasmanian berry grower Nic Hansen showing Macau chef Antimo Merone ar	What is the Berry Export
1	RWSN Collaborations Southern Africa Self-supply Study Review of Self-supply and its	What are some of the b
2	All Android applications categories Description Coolands for Twitter is a revolutionar	What are the unique fe
3	How unequal is India? The question is simple, the answer is not. For some 60 years, th	What is the main differ
4	Gunnar Nelson took his time on the feet against Zak Cummings tonight in the co-mai	How did Gunnar Nelson

Выберите CSV-файл, максимальный размер файла 200 Мб

Drag and drop file here
Limit 200MB per file • CSV

dataset1.csv 270.0B

Требуется ровно 3 столбца. Текущее количество: 2

MVP НА ОСНОВЕ BASELINE

Информация

- Показать информацию о данных
- Проверить есть ли дубликаты
- Проверить есть ли пропуски в данных

строки с пропусками

Всего 12000 строк

Всего 3 столбца

В датасете есть дубликаты в колонке question

В датасете есть пропущенные значения.

Список строк с пропущенными значениями:

Строка 340

Строка 2818

Строка 11746

Строки с пропущенными значениями:

	context	question	answer
340	. The Give people. Home Office. (2002). A guide to anti-social behaviour orders and a	None	None
2,818	come back ! for Bud come back ! for Bud the stooges are downmost This page hosted	None	None
11,746	Flood, Definition of Flood: - Arrive in large quantities or quantities. - Cover or wet with	None	None

ВЫВОД
ИНФОРМАЦИИ О
ДАННЫХ: ОБЪЕМ
ДАННЫХ, ПРОПУСКИ,
ДУБЛИКАТЫ,
ВЫПОЛНЯЕТСЯ НА
СТОРОНЕ frontend

MVP HA OCHOBE BASELINE

Графики

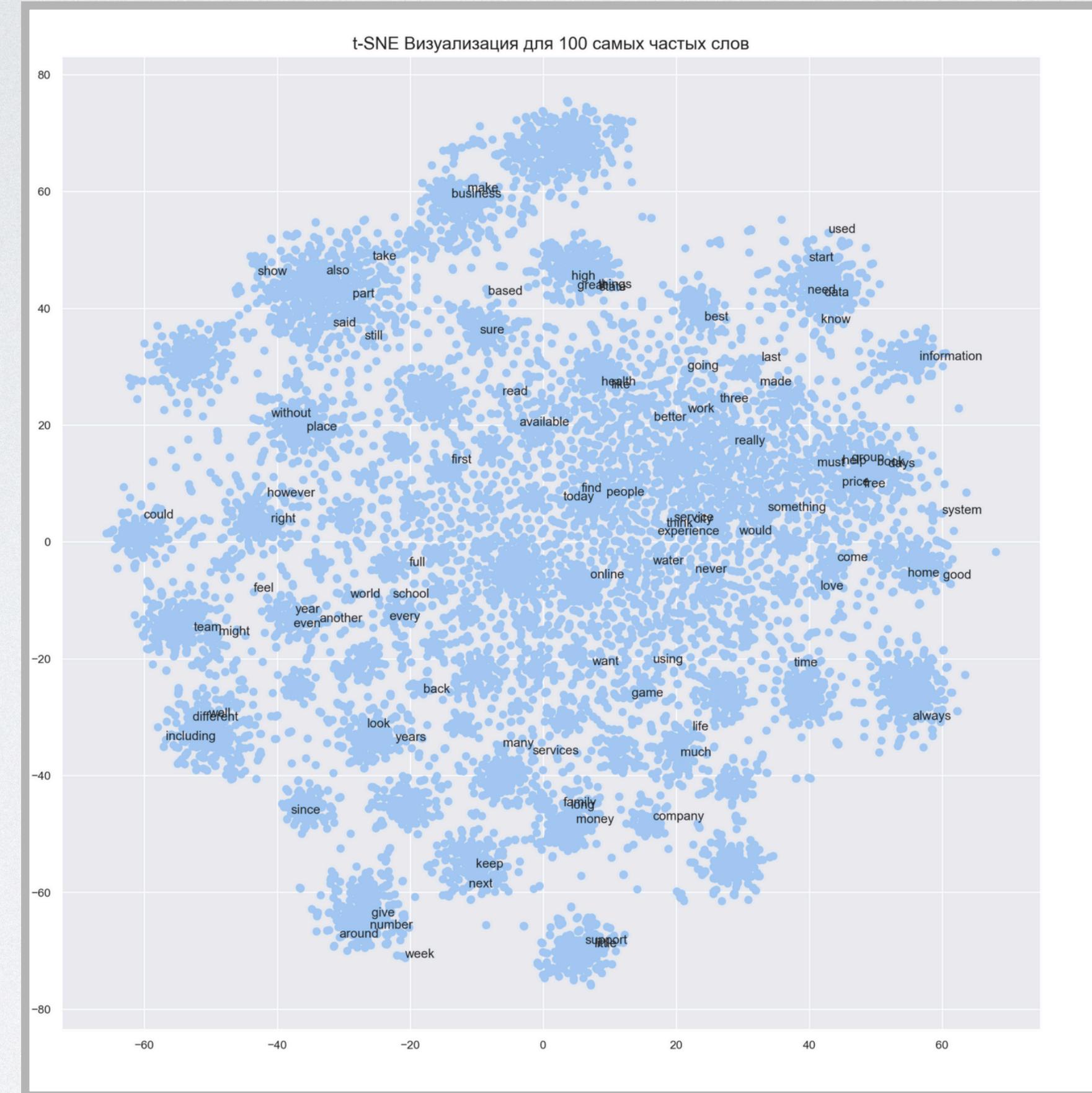
- Длины текстов
 - Частотность слов
 - Облако слов
 - t-SNE для топ-200 слов 

Препроцессинг

- удаление дубликатов и пропусков
 - предобработка текста
 - отправка данных на сервер

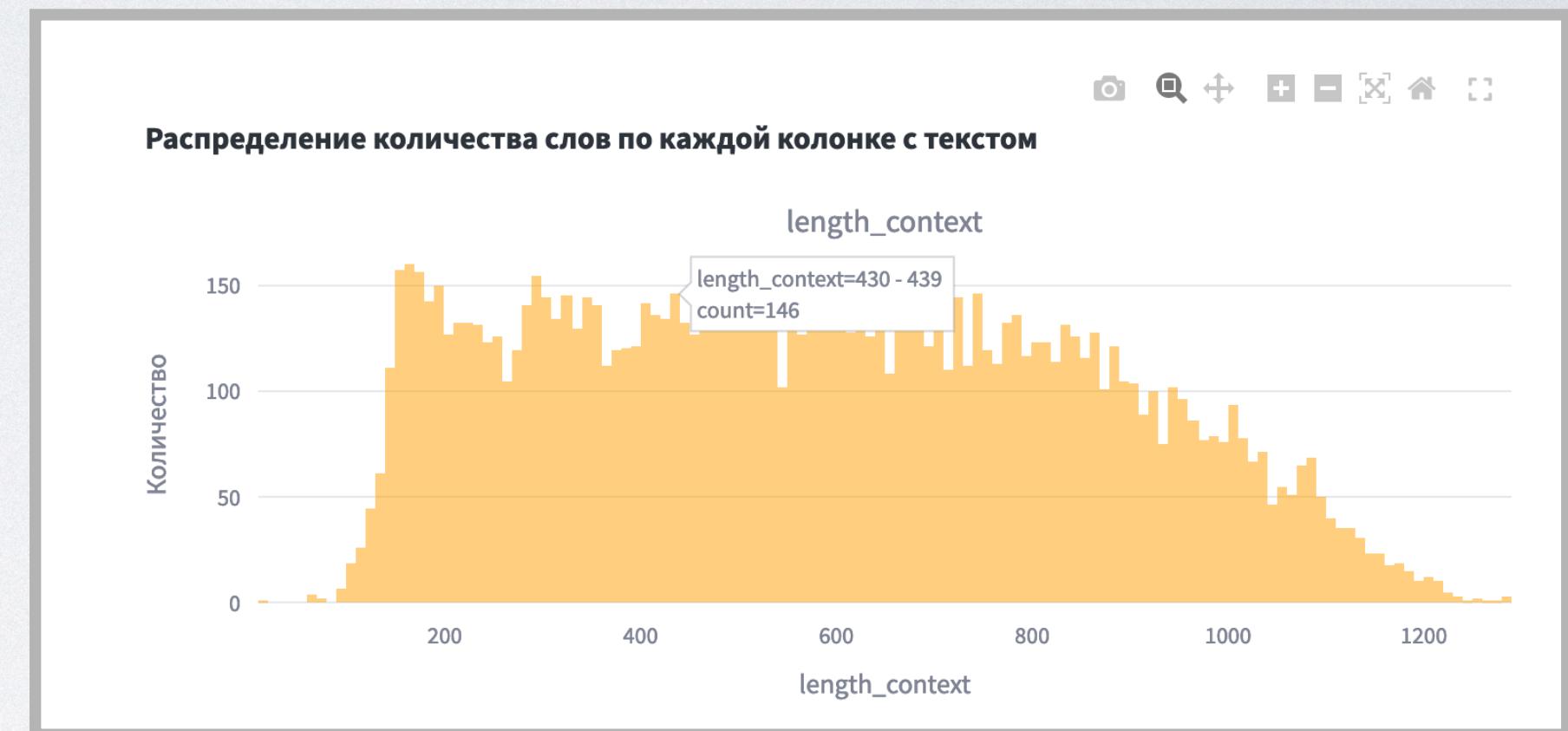
Преимущества предпроцессинга включают:

- удаление стоп-слов, дубликатов, пропусков
 - удаление символов не являющимися буквами и цифрами
 - приведение текста к нижнему регистру,



[очистить и отправить] вызывает метод POST/api/v1/models/load_dataset
очищенные данные загружаются на сервер

приложение строит аналитические графики, в том числе интерактивные



MVP НА ОСНОВЕ BASELINE

Модель

Настроить

Параметры векторизации

max_df
0.85

0.00 1.00

min_df
3

1 10

max_features
50000

20000 100000

sublinear_tf
 True
 False

ngram_range
 (1, 1)
 (1, 2)
 (1, 3)

model_id
test_01

7/20

Сохранить параметры

Модель

Настроить

Параметры векторизации

max_df
0.85

0.00 1.00

min_df
5

1 10

max_features
88000

20000 100000

sublinear_tf
 True
 False

ngram_range
 (1, 1)
 (1, 2)
 (1, 3)

model_id
test_02

Сохранить параметры

приложение позволяет настроить гиперпараметры и обучить несколько моделей, для дальнейшего сравнения и использования, у каждой модели есть свой идентификатор

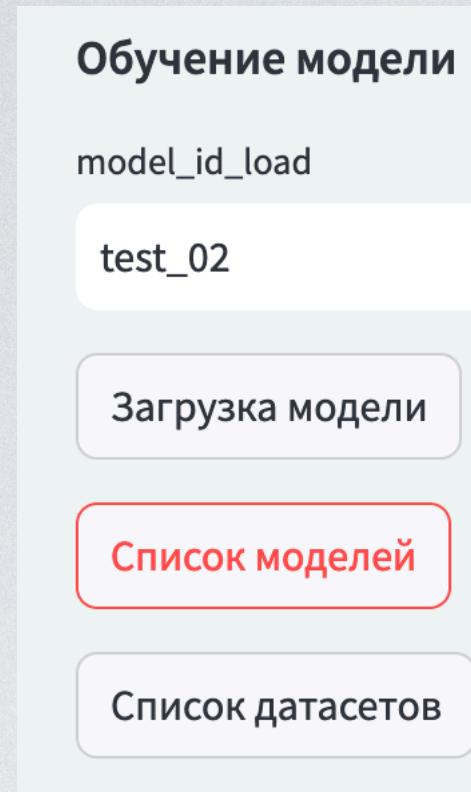
Данные преобразованы и загружены в qdrant с помощью модели 'test_01'

Данные преобразованы и загружены в qdrant с помощью модели 'test_02'

[сохранить параметры] вызывает метод

POST/api/v1/models/fit_save данные размещены в БД и готовы к использованию

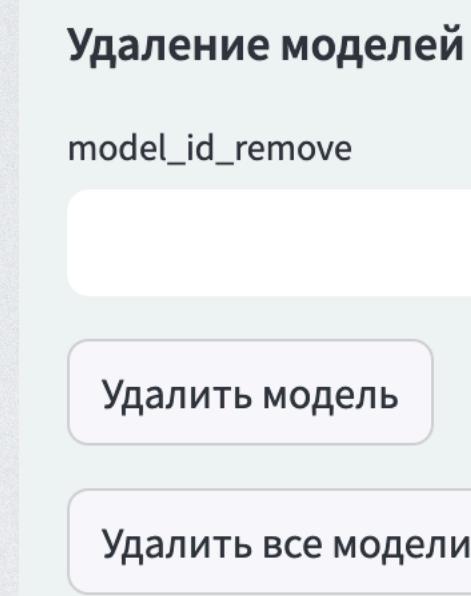
MVP НА ОСНОВЕ BASELINE



вывод информации о загруженных моделях

[список моделей] вызывает метод **GET/api/v1/models/load_model**

показывает список загруженных и обученных моделей



[список датасетов] вызывает метод **GET/api/v1/models/get_datasets**

получить список загруженных наборов данных

Возможно удаление модели по идентификатору и удаление всех моделей

[удалить модель] вызывает метод **DELETE /api/v1/models/remove/{model_id}**

[удалить все модели] вызывает метод **DELETE /api/v1/models/remove_all**

MVP НА ОСНОВЕ BASELINE

Тестирование моделей

- укажите id модели для теста

model_id_test1

test_01

model_id_test2

test_02

Бенчмарк

Точность

Точность для модели test_01: 69.12561473701759 %

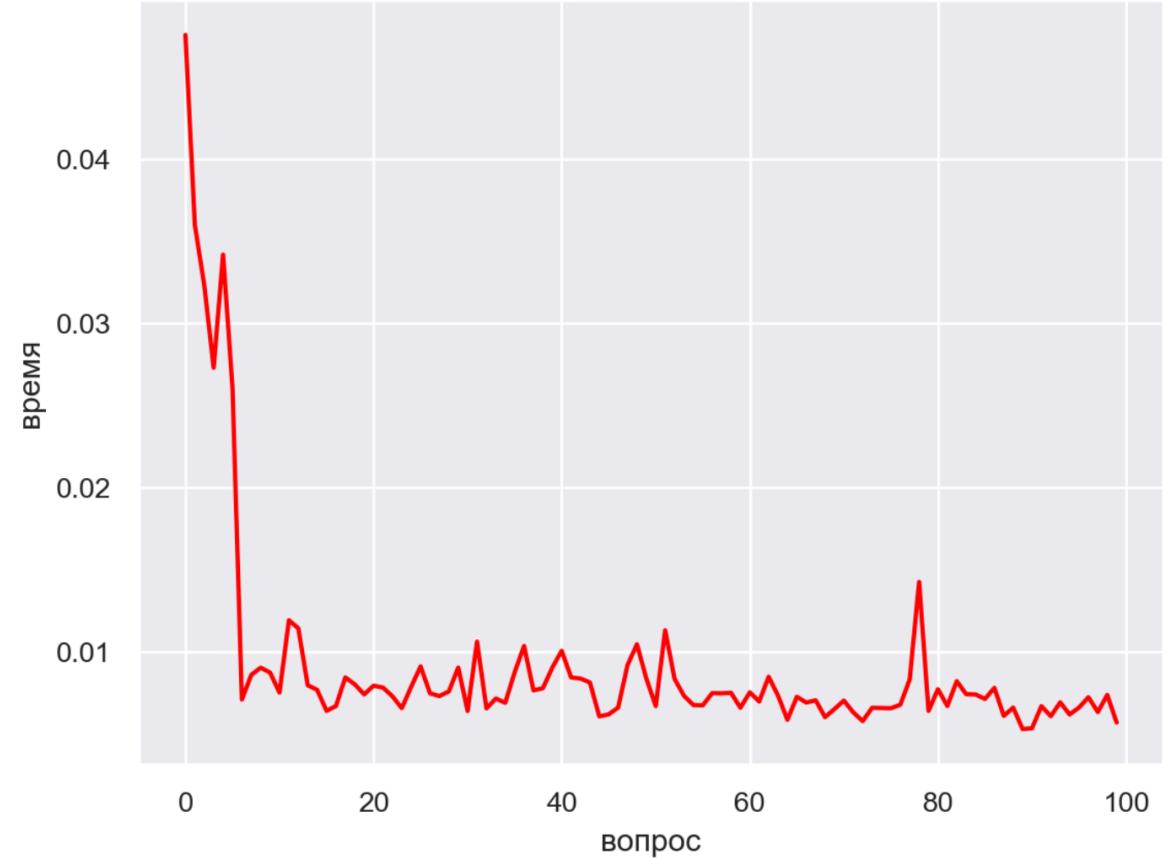
Точность для модели test_02: 68.77552721513712 %

“точность” вызывает метод **POST/api/v1/models/quality_test**

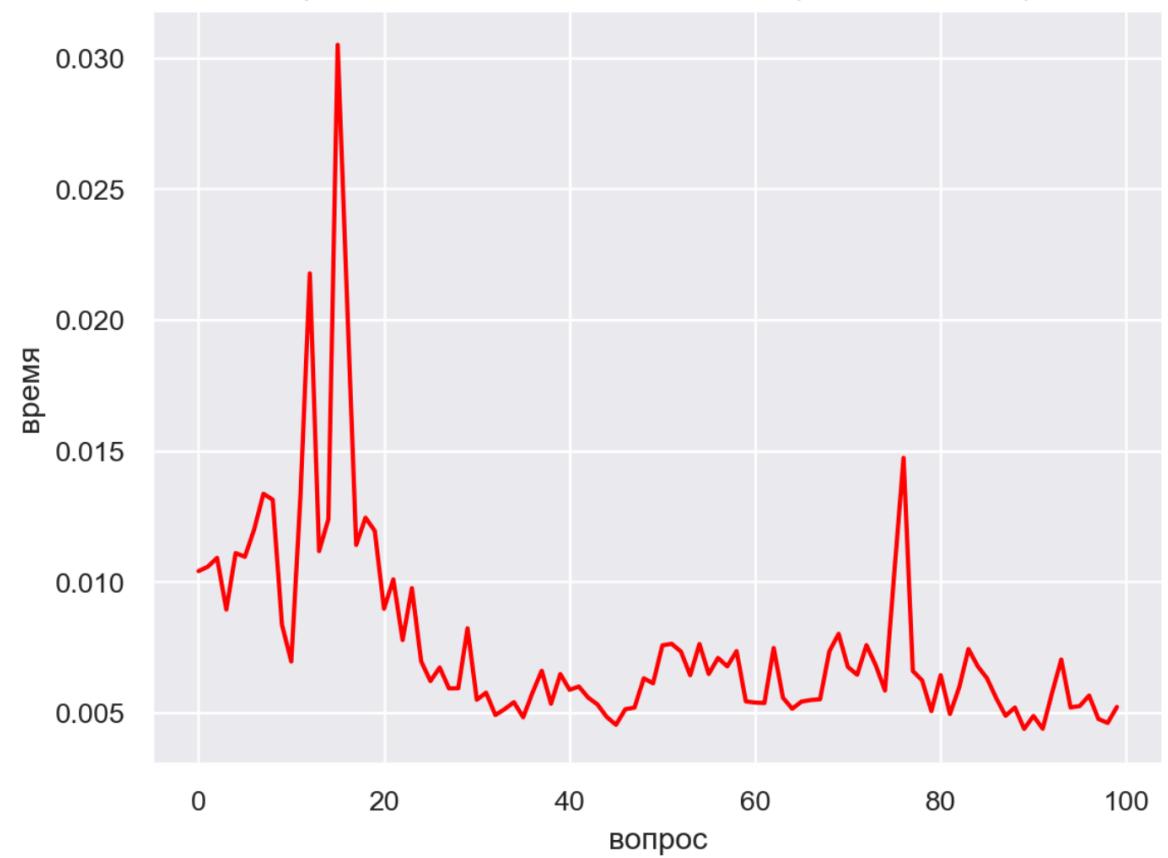
выполняется извлечение для каждого вопроса для датасета и сверка его с правильным

Бенчмарк - графики, показывающие время извлечения ответов из БД для двух разных моделей, тестирование проводится на одних и тех же 100 случайных примерах вопросов

Время извлечения ответов из базы (на 100 сэмплах)



Время извлечения ответов из базы (на 100 сэмплах)



MVP НА ОСНОВЕ BASELINE

Модель

Настроить

Инференс

[**отправить текст**] вызывает метод
POST/api/v1/models/find_context

выполняется извлечение:

- ответа из БД,
- идентификатора найденного контекста
- оценки

Введите текст для проверки работы модели

How did Gunnar Nelson win the fight against Zak Cummings at UFC Fight Night 46?

model_id
test_02

Отправить текст

Текст отправлен в модель

How did Gunnar Nelson win the fight against Zak Cummings at UFC Fight Night 46?

Ответ: gunnar nelson took time feet cummings tonight main event fight night finally opportunity switch gears second round pounced taking opponent seizing back locking fight ending choke almost effortlessly round nelson usual karate style stance keeps range cummings early strikes presses clinch cummings matches jockey position break apart nelson lands good knee nelson lands right hook misses overhand nelson relaxed alert continually makes cumming miss backing enough right hand counter lands nelson cummings moves back clinches nelson turns though breaks away clinch head kick blocked cummings pace slowed little final seconds cummings leaps nelson counters punch body missed shots cummings moves clinch nothing back however cummings comes back finally land nice punches nelson back cage long though nelson circles final action round round nelson biding time opening minute second cummings continually tries land strikes generally comes short however cummings back cage unleash flurry forces nelson clinch stay clinch trying gain advantage stalemate disengage much volume nelson striking round land speedy right hand clearly hits mark certainly better accuracy little offense astray finds home clinical right hand starts close distance remains patient snaps another quick right clinch range nelson reaches guillotine choke goes fluidly switches cummings back lands punches swiftly locks rear naked choke impressive stuff nelson able transition slickly dangerous ground game forces cummings twelve seconds second round remaining

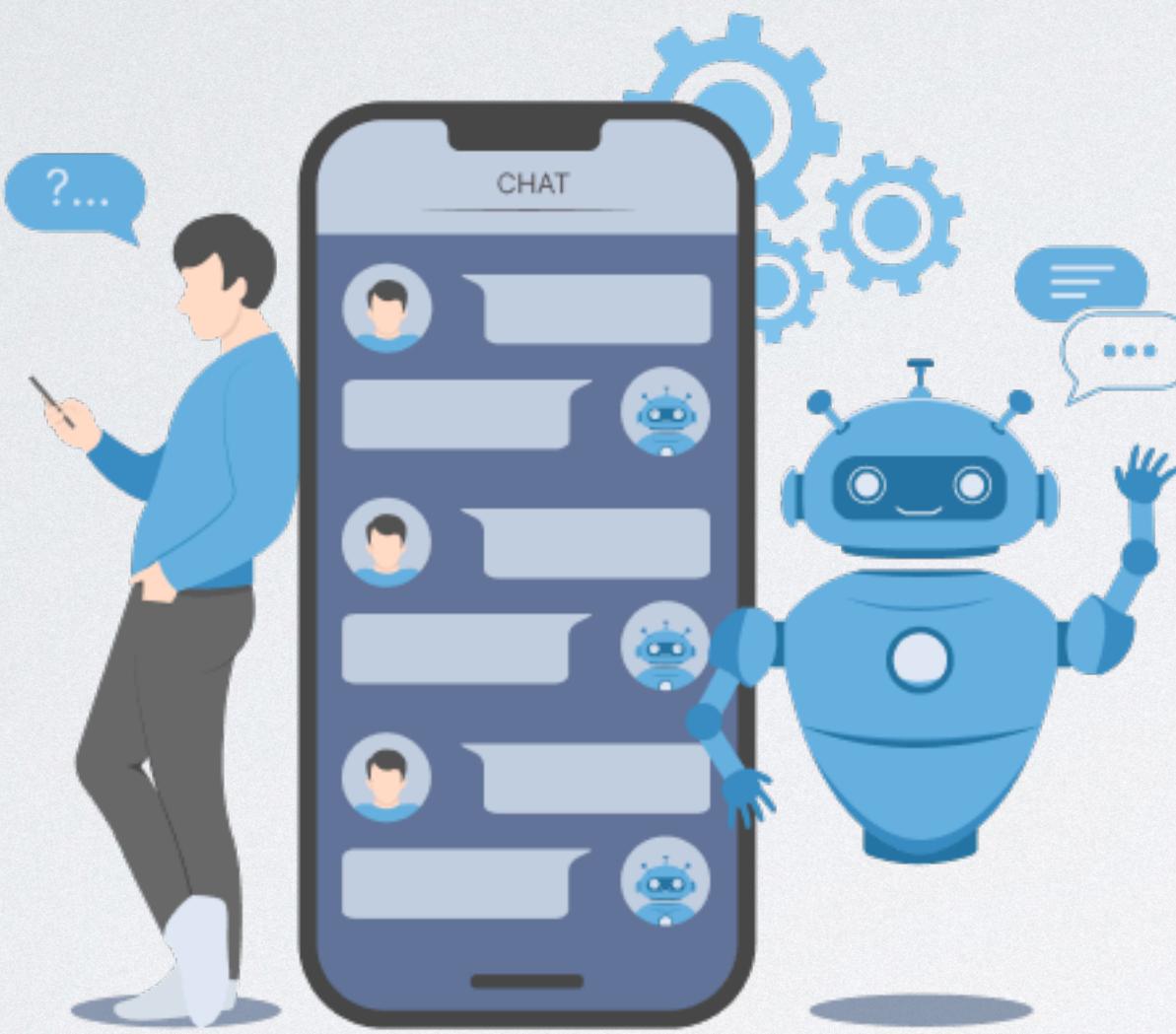
Score: 0.40582517, Идентификатор 4

PRE-COMMIT СЕРВИС

```
11 ./client.py:28:1: E302 expected 2 blank lines, found 1
12 ./client.py:43:1: E302 expected 2 blank lines, found 1
13 ./client.py:50:1: E302 expected 2 blank lines, found 1
14 ./client.py:57:1: E302 expected 2 blank lines, found 1
15 ./client.py:61:1: E302 expected 2 blank lines, found 1
16 ./client.py:65:1: E302 expected 2 blank lines, found 1
17 ./client.py:67:1: E115 expected an indented block (comment)
18 ./client.py:87:24: W292 no newline at end of file
19 **** Module client
20 client.py:87:0: C0304: Final newline missing (missing-final-newline)
21 client.py:11:0: E0401: Unable to import 'httpx' (import-error)
22 client.py:28:0: C0116: Missing function or method docstring (missing-function-docstring)
23 client.py:43:0: C0116: Missing function or method docstring (missing-function-docstring)
24 client.py:50:0: C0116: Missing function or method docstring (missing-function-docstring)
25 client.py:57:0: C0116: Missing function or method docstring (missing-function-docstring)
26 client.py:61:0: C0116: Missing function or method docstring (missing-function-docstring)
27 client.py:65:0: C0116: Missing function or method docstring (missing-function-docstring)
28 client.py:12:0: C0411: standard import "random" should be placed before third party import "httpx"
(wrong-import-order)
29 client.py:13:0: C0411: standard import "time" should be placed before third party import "httpx"
(wrong-import-order)
30
31 -----
32 Your code has been rated at 6.11/10
```

Автоматический pre-commit сервис при выполнении Pull Request
в git репозиторий реализован в файле: [**ci-lint.yml**](#)

ЦЕЛИ ПО ПРОЕКТУ НА ВТОРОЕ ПОЛУГОДИЕ



Разработать generation часть приложения

1. Работа над генерацией ответов:
 - Выбор метрик для оценки работы модели.
 - Подбор моделей для генерации ответа (Llama/Mistral/GigaChat/etc).
 - Настройка параметров моделей.
 - Оценка результатов и выбор модели.
 - Дообучение модели (Опционально).
2. Разработка и тестирование цепочки RAG.
3. Применение техник преобразования запросов (перефразирование/подзапросы и т.д.).
4. Оптимизация параметров извлечения информации (кол-во релевантных контекстов).
5. Применение алгоритмов для ускорения работы (Опционально).
6. Применение алгоритмов для улучшения генерации (Опционально).

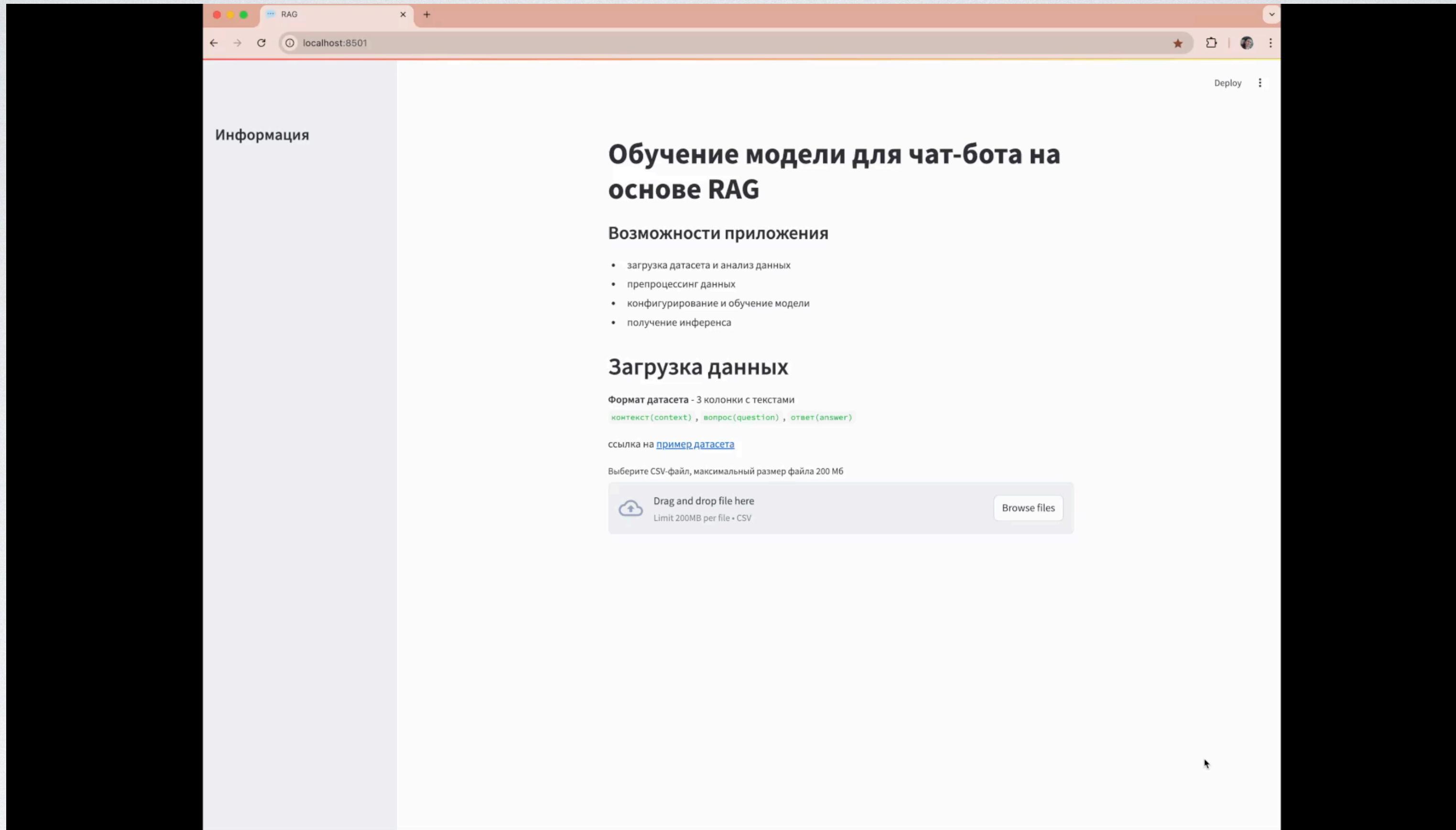
Обернуть созданную модель в интерфейсный сервис в виде приложения

1. Проектирование архитектуры приложения.
2. Выбор платформы telegram bot/streamlit.
3. Разработка интерфейса приложения.
4. Тестирование.
5. Разворачивание приложения на облачном сервере.

РАСПРЕДЕЛЕНИЕ РАБОТЫ В КОМАНДЕ

- EDA (каждый участник проводил исследования для того, чтобы познакомиться с данными)
- Baseline:
 - Fasttext (Евгений Яковенко)
 - Word2vec/Sentence Transformers/ замеры времени извлечения ответов в разных условиях (Людмила Теплова)
 - Tf-idf (Альберт Тайчинов)
 - BM25 with ANN/all-MiniLM-L6-v2/Hybrid Search (BM25 + Sentence Transformers) (Алексей Яковский)
- Backend (FastAPI) - (Альберт Тайчинов)
- Frontend (Streamlit) - (Людмила Теплова)
- Docker + VPS + Grafana (Евгений Яковенко)
- Pre-committer (Алексей Яковский)





Короткий скринкаст, демонстрирующий работу сервиса:

https://drive.google.com/file/d/1AiNuvZpioWQn_IMkA-I7Zoli_eNVIj3b/view?usp=sharing