МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ ИНТЕЛЛЕКТУАЛЬНОЙ РОБОТОТЕХНИКИ

Кафедра Интеллектуальных систем теплофизики ИИР

Направление подготовки 15.03.06 Мехатроника и робототехника

Направленность (профиль) Мехатроника и робототехника

ОТЧЕТ

о прохождении учебной практики, научно-исследовательской работы (получение первичных навыков научно-исследовательской работы)
(указывается наименование практики)

Обучающегося Сыренного Ильи Игоревича группы № 21930 курса 4

Тема задания: Разработка интерактивного учебного пособия с ответами на естественном языке на основе Retrieval Augmented Generation

Место прохождения пратики: Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Новосибирский национальный исследовательский государственный университет». 630090, Новосибирская область, г. Новосибирск, ул. Пирогова, д. 1

Сроки прохождения практики: с 30.09.2024 г. по 23.12.2024 г.

Руководитель практики от НГУ Галактионова Юлия Юрьевна, специалист УМОВОИИР /_____
(Ф.И.О. полностью, должность) (подпись)

Руководитель ВКР Оглезнев Никита Сергеевич, сотрудник ИИР НГУ, ассистент /_____
(Ф.И.О. полностью, должность) (подпись)

Оценка по итогам защиты отчета: _____
(неудовлетворительно, удовлетворительно, хорошо, отлично)

Отчет заслушан на заседании кафедры КафИСТИИР
(наименование кафедры)

протокол _____от «_____» _____ 20_____г.

Новосибирск 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В условиях стремительно развивающихся технологий и множества новых научных исследований в области ИТ и других дисциплин, необходимость в эффективном освоении научной литературы становится все более актуальной. Однако одним из существенных препятствий для многих специалистов, студентов и исследователей является языковой барьер, а также сложности с пониманием специализированных терминов, особенно в новых и быстро развивающихся областях знаний. Научные статьи часто пишутся на иностранных языках, в частности на английском, что затрудняет их восприятие для тех, кто не владеет языком на должном уровне. Это приводит к увеличению времени на изучение материалов, снижению качества усвоения информации.

К тому же, процесс работы с большими объемами научных данных и статей, поиск и извлечение релевантной информации остаются трудоемкими и зачастую неэффективными, особенно когда необходимо справляться с большими потоками информации. В этих условиях актуальной задачей является разработка систем, которые могут облегчить и ускорить процесс исследования, а также снизить языковые и информационные барьеры.

Цель выпускной квалификационной работы — разработка системы, использующей методы Retrieval-Augmented Generation (RAG) для облегчения процесса изучения научных статей, обеспечивая поиск и объяснение терминов, автоматический перевод, а также предоставление ссылок на оригинальные источники. Функционал системы позволит улучшить качество и скорость освоения материала, а также повысить эффективность научной работы для специалистов в области ИТ и науки.

Для достижения цели были поставлены следующие задачи:

1 проанализировать существующие решения, обозначить их особенности;

2 изучить современные подходы к построению систем на основе Retrieval Augmented Generation;

3 разработать интуитивно понятный интерфейс взаимодействия пользователя с системой;

4 создать сервис для интеграции Retrieval Augmented Generation, обеспечивающий взаимодействие пользователя с системой через вышеупомянутый интерфейс;

5 разработать алгоритм оценки качества системы;

1 ТЕОРЕТИЧЕСКИЕ ОСНОВЫ RETRIEVAL AUGMENTED GENERATION

Большие языковые модели доказали свою способность усваивать значительный объем знаний из данных. Они способны делать это без доступа к внешней памяти, выступая в роли неявной базы данных. Однако они склонны к генерации устаревшей информации или галлюцинациям. Различные подходы к построению RAG решают эти проблемы, объединяя преимущества больших языковых моделей и внешней базы данных.

Фундаментальный принцип RAG заключается в поиске релевантной информации для дополнения запроса, передаваемого в большую языковую модель. На Рисунке 1 представлена общая архитектура RAG, которая включает в себя поиск документов (retrieve), соответствующих запросу, и передачу большой языковой модели для генерации правильного ответа (generate).

RAG обеспечивает генерацию ответов, основанных на внешних данных, что позволяет системе успешно справляться с запросами, которые требуют знаний из новых, специфичных или динамически обновляемых источников, выходящих за рамки обучающей выборки модели.

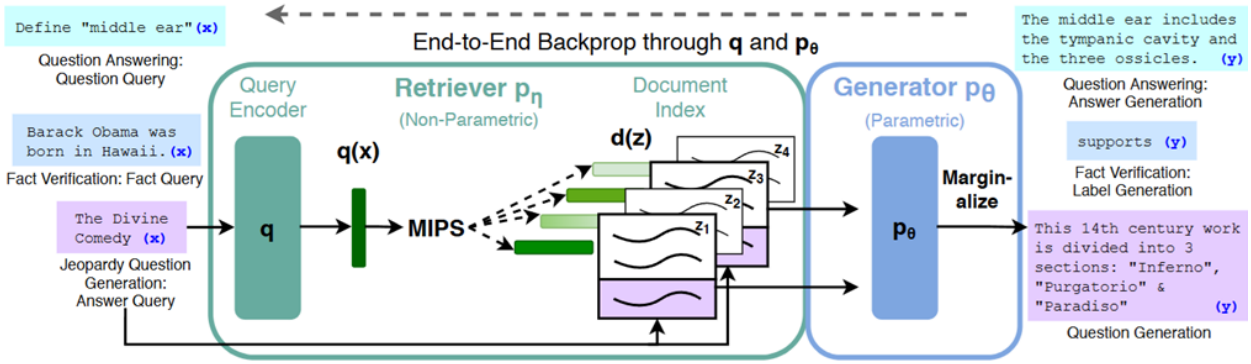


Рисунок 1 - Устройство пайплайна Retrieval Augmented Generation

1.1 Ключевые этапы RAG пайплайна

1.1.1 Chunking

Первый этап RAG-пайплайна – Chunking. Этот процесс заключается в разбиении текста на отдельные фрагменты, которые могут быть проиндексированы и впоследствии использованы для поиска релевантной информации. Основная задача Chunking — создать фрагменты, которые одновременно содержат достаточно информации для понимания, но не являются слишком объемными, чтобы усложнять обработку. Для реализации этого этапа используются два основных подхода:

1 эвристический подход. Он основан на явных признаках структуры текста, таких как знаки пунктуации, границы абзацев, заголовки или иерархия документа.

2 подход на основе семантического сходства фрагментов текста. Этот подход направлен на раделение текста на фрагменты, которые максимально сохраняют смысловую целостность.

1.1.2 Retrieval

После разделения текста на фрагменты необходимо отобрать среди них наиболее релевантные к запросу. Цель Retrieval-этапа — сузить множество доступных текстовых фрагментов до небольшого набора наиболее релевантных, которые затем могут быть использованы для послеюдующей обработки. Для этого применяются разные алгоритмы информационного поиска, например BM25 и TF-IDF, которые оценивают релевантность на основе текстовых характеристик, включая частотность слов и их значимость в контексте всего корпуса.

Помимо традиционных алгоритмов, в Retrieval часто используются подходы с использованием эмбеддинг-моделей. Эти модели преобразуют запросы и текстовые фрагменты в векторные представления, что позволяет вычислять их сходство в векторном пространстве.

2 ПРОДВИНУТЫЕ ПОДХОДЫ

Среди прочего существует множество подходов для улучшения оригинальной архитектуры RAG. Начнем рассмотрение подходов по их порядку следования в пайплайне.

2.1 Chunking

Сложность данного этапа в том, чтобы выдержать баланс между размером и смысловой наполненностью фрагментов. Это соотношение сильно варьируется в зависимости от доменной спецификации текстов, с которыми работает система.

Далее рассмотрим критерии качества чанков:

1 **Семантическая целостность.** Фрагменты текста должны содержать логически завершенный фрагмент текста, который не теряет смысла при извлечении из контекста. Так, при разрыве чанка на середине предложения или смыслового блока, это может привести к потере важной информации. Для соблюдения данного критерия необходимо не только сохранять целостность отдельных предложений, но и в идеале объединять предложения в смысловые блоки.

2 **Полнота информации.** Чанк должен содержать достаточно информации для ответа на типичный запрос, связанный с этим фрагментом текста.

2.1.1 Семантический Chunking

1 **Similarity Chunking.** Данный метод использует эмбеддинг-модели для группировки связанных по смыслу предложений. Основан на косинусной близости между эмбеддингами предложений. Чанки разделяются по пороговому значению косинусной близости между соседними предложениями.

2 **Perplexity Chunking.** На начальном этапе текст разделяется на набор предложений (x1, x2, ..., xn). Цель данного метода - сформировать набор чанков (X1, X2, ..., Xk), где каждый чанк представляет собой логически связанное объединение исходных предложений. Для объединения исходных предложений в чанки модель вычисляет перплексию (PPL) для каждого предложения x_i на основе предшествующих предложений:

$$\mathrm{PPL}_M(x_i) = \frac{\sum_{k=1}^{K} \mathrm{PPL}_M(t_k^i | t_{<k}^i, t_{<i})}{K} \quad (1)$$

где K — общее количество токенов в x_i, t_i^k — k-й токен в x_i, а $t^{<i}$ обозначает все токены, предшествующие x_i. Для определения границ чанков алгоритм

анализирует последовательность:

$$\mathrm{PPL}_{seq} = (\mathrm{PPL}_M(x_1), \mathrm{PPL}_M(x_2), ..., \mathrm{PPL}_M(x_n)) \tag{2}$$

В поисках минимальных значений:

$$\mathrm{Minima}_{index}(\mathrm{PPL}_{seq}) = \Big\{ i \;\Big|\; \min(\mathrm{PPL}_M(x_{i-1}), \mathrm{PPL}_M(x_{i+1})) - \mathrm{PPL}_M(x_i) > \theta, \\ or\ \mathrm{PPL}_M(x_{i-1}) - \mathrm{PPL}_M(x_i) > \theta\ and\ \mathrm{PPL}_M(x_{i+1}) = \mathrm{PPL}_M(x_i) \Big\} \tag{3}$$

Смысл данной формулы заключается в следующем: если значения PPL по обе стороны точки выше, чем в самой точке, и разница хотя бы с одной стороны превышает заданный порог θ; либо разница между левой точкой и текущей больше θ, а значение справа равно значению в текущей точке. Эти минимумы рассматриваются как потенциальные границы фрагментов.

2.2 Query Rewriting

Данный этап направлен на улучшение соответствия между пользовательским запросом и информацией, содержащейся в базе данных. Этот процесс позволяет уточнить исходный запрос и адаптировать его под структуру данных системы.

2.2.1 HyDE

Авторы данного метода предлагают с помощью большой языковой модели генерировать гипотетический документ, который мог бы послужить ответом на запрос пользователя. Запрос подается в модель с инструкцией “напишите документ, который отвечает на вопрос”, таким образом создается гипотетический документ. Созданный документ не является реальным, и может содержать фактические ошибки, так как его задача лишь имитировать релевантный текст. Ожидается, что в результате кодирования документа с помощью эмбеддинг-модели удалятся лишние (вымышленные) детали и в результате полученный вектор станет семантически ближе к релевантным фрагментам, чем вектор оригинального запроса пользователя.

2.3 Reranking (Two-Stage Retrieval)

Извлеченные на этапе Retrieval документы затем проходят этап переранжирования. Цель данного этапа — улучшить порядок документов, где наиболее релевантные документы располагаются выше. Рассмотрим два подхода для получения меры близости текстовых сущностей:

1 Bi-Encoder. Запрос и документ проходят через отдельные энкодеры, которые создают эмбеддинги. Затем эмбеддинги отображаются в одном векторном пространстве, и рассчитывается их косинусное сходство. На Рисунке 2 показано, как тексты преобразуются в векторы, объединяются в итоговые вектора (Pooling), а затем вычисляется их сходство.

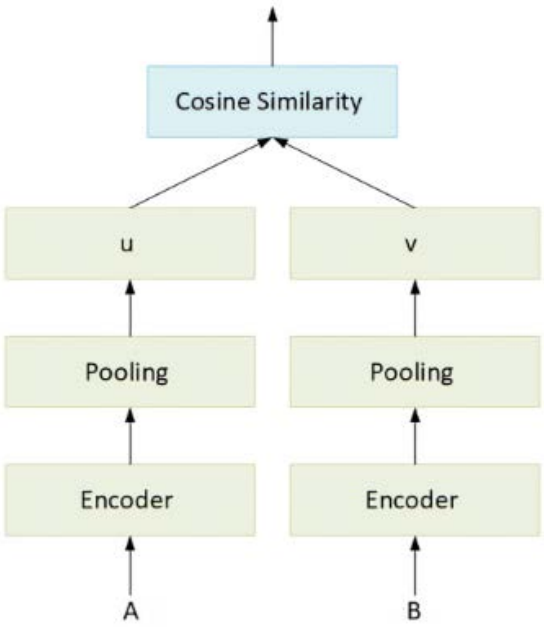


Рисунок 2 — Bi-Encoder

2 Cross-Encoder. Запрос и документ комбинируются в одном входе для модели. Модель затем генерирует один эмбеддинг для пары запрос-документ. На Рисунке 3 изображена последовательность преобразований. Similarity Score – выходной слой (например, полносвязный слой с сигмоидной функцией активации), который выдает оценку сходства – число в диапазоне от 0 до 1.

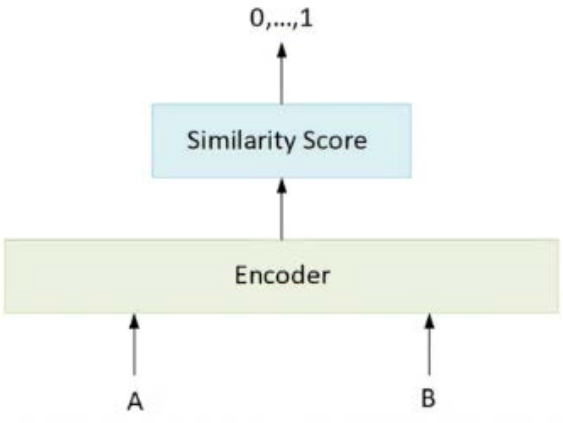


Рисунок 3 — Cross-Encoder

3 ТОЧКИ ОТКАЗА СИСТЕМ С RETRIEVAL AUGMENTED GENERATION

В статье “Seven Failure Points When Engineering a Retrieval Augmented Generation System” рассматриваются семь основных проблем, с которыми сталкиваются при разработке систем с RAG:

1. Отсутствие необходимого контента. Система может не находить релевантную информацию, если она отсутствует в базе данных или была неправильно индексирована.
2. Пропуск высокоранжированных результатов. Даже если релевантный контент был найден, то система может не выбрать его из-за неточностей в алгоритмах поиска или ранжирования.
3. Неправильный контент. Предоставленные документы могут не соответствовать запросу пользователя, что приводит к генерации нерелевантных или ошибочных ответов.
4. Неподходящий формат: Информация может быть представлена в формате, который затрудняет ее обработку или интеграцию в ответ, например, в виде изображений или таблиц.
5. Неправильная степень специфичности: Ответы могут быть слишком общими или, наоборот, чрезмерно детализированными, не соответствуя ожиданиям пользователя.
6. Неполные ответы: Система может предоставлять ответы, содержащие лишь часть необходимой информации, даже если полный ответ доступен в предоставленных документах.
7. Необходимость постоянной калибровки: Компоненты RAG-системы требуют регулярной настройки и обновления для поддержания согласованности и точности ответов, особенно при изменении данных или предпочтений пользователей.

4 ОЦЕНКА СИСТЕМ С RETRIEVAL AUGMENTED GENERATION

RAG — это система, тесно связанная с конкретными требованиями к системе, а также языковыми моделями, что приводит к использованию различных методов оценки. При анализе существующих решений можно выявить несколько ключевых способов оценки систем с RAG:

1 Оценка человеком.

2 Датасеты для оценки.

3 Фреймворки для оценки.

В обзорной статье Evaluation of Retrieval-Augmented Generation: A Survey представлена сводная таблица фреймворков и инструментов для оценки RAG-систем. Помимо RAGAS и MultiHop-RAG, о которых пойдет речь дальше, можно выделить датасет QASPER, как решение, подходящее по области вопросов.

4.1 QASPER

QASPER [14] — это датасет для задач вопросно-ответного поиска (QA) на научных статьях в области обработки естественного языка (NLP). Он включает 5,049 вопросов, относящихся к 1,585 научным статьям по NLP. Каждый вопрос был составлен специалистом в области NLP, который ознакомился только с заголовком и аннотацией соответствующей статьи. Задача этих специалистов — сформулировать вопросы, основанные на информации, которая должна быть найдена в полном тексте статьи. После этого на вопросы отвечает другая группа исследователей, их задача не только ответить на вопрос, но и предоставить релевантные фрагменты исходного текста, которые подтверждают ответ. Особенность данного датасета состоит в том, что для ответа на вопросы используется только один документ. Таким образом, для корректной оценки разрабатываемой системы необходим еще и датасет с вопросами, на которые придется отвечать, основываясь сразу на нескольких документах. Для оценки качества ответа на этом датасете используется f1-score метрика, которая считается по

4.2 RAGAS

RAGAS [6] — это фреймворк с открытым исходным кодом, разработанный для оценки качества работы RAG-систем. RAGAS включает в себя две основные части: генерацию синтетического тестового датасета и метрики качества. Метрики для отдельных частей RAG позволяют декомпозировать оценку пайплайна на оценку его составляющих. С помощью таких метрик проще отслеживать уязвимые места системы, а также оценивать реакцию на изменение отдельных компонентов.

Как генерация датасета, так и метрики основаны на принципе “LLM as judge”. С помощью специальных промптов делаются запросы в большую языковую модель с целью оценить качество или сгенерировать вопрос для теста. Такой подход позволяет максимально автоматизировать процесс оценки RAG-систем. С другой стороны, статичные промпты не

адаптируются под конкретные задачи, что может затруднить, например, оценку системы, которая должна отвечать на русском языке. Эту проблему отчасти решает расширение фреймворка RAGAS, разработанное в России — GigaRAGAS. Особенность расширения состоит в том, что промпты для метрик и генерации датасета переведены и адаптированы под оценку RAG на русском языке.

5 РАЗРАБОТКА СОБСТВЕННОЙ СИСТЕМЫ

5.1 Проектирование

Для проектирования системы были использованы UML-диаграммы (См. приложение Б), которые позволили структурировать требования, визуализировать архитектуру и глубже понять функциональность системы. В процессе проектирования также был создан прототип интерфейса, что позволило предположить, как пользователь будет взаимодействовать с системой и какие шаги будут необходимы для выполнения ключевых действий.

5.1.1 Диаграмма вариантов использования (Use Case)

Проектирование началось с создания диаграммы вариантов использования, которая отображает ключевые сценарии взаимодействия пользователей с системой. Основными акторами являются Пользователь и Администратор, каждый из которых взаимодействует с системой в рамках своих сценариев:

1. Пользователь: задает вопросы, просматривает найденные системой фрагменты документов.
2. Администратору: имеет доступ к функционалу пользователя, а также дополнительным функциям, таким как настройка параметров системы, загрузка, удаление и просмотр документов.

Диаграмма вариантов использования помогла выделить основные функции, которые система должна предоставлять пользователям в разных ролях.

5.1.2 Диаграмма последовательностей (Main Sequence)

Следующим шагом была разработка диаграммы последовательностей, которая иллюстрирует, как объекты системы взаимодействуют друг с другом для выполнения основного сценария работы. Эта диаграмма показывает последовательность шагов, включая отправку запроса, обработку его на сервере и возврат результатов пользователю. На основе диаграммы можно проследить, как данные перемещаются через систему и как различные компоненты обмениваются сообщениями. Этот этап помог уточнить логику взаимодействий между элементами сервиса.

5.1.3 Диаграмма компонентов сервиса

Для проектирования архитектуры системы была создана диаграмма компонентов. Она

представляет ключевые модули, такие как RAG-модули, базы данных, API и их взаимодействие. Эта диаграмма стала важным инструментом для понимания структуры системы, позволяя выделить основные компоненты и определить их связи. Кроме того, диаграмма помогла выявить протоколы взаимодействия компонентов, а также их зависимости друг от друга, что существенно облегчило разработку системы.

5.1.4 Дополнение: Диаграмма активностей (User Activity)

В дополнение к созданным диаграммам также была составлена диаграмма активностей. Это анализ активностей пользователей, который помог понять, как именно пользователи взаимодействуют с системой в рамках различных сценариев.

5.2 Прототип

На этапе разработки прототипа были реализованы ключевые компоненты системы, включая интерфейс, RAG-модули, API, взаимодействие с базами данных, а также документация. Основной целью было создать работающий минимальный вариант системы, который демонстрирует ее основные функции и позволяет протестировать концепцию на практике.

5.2.1 Интерфейс

Первым шагом в разработке был интерфейс чата, который обеспечивал основу для взаимодействия пользователя с системой. Интерфейс был спроектирован с учетом опыта аналогичных проектов, ориентируясь на удобство использования и интуитивную понятность. Пользователь может задавать вопросы, а система отвечает на них, используя RAG пайплайн. Для обеспечения асинхронности и стримингового обмена данными, интерфейс был интегрирован с серверной частью через WebSockets, что позволило реализовать динамическую генерацию ответов в реальном времени.

5.2.2 Серверная часть

В серверной части были реализованы основные части RAG-пайплайна, включающие:

1. Chunking: был использован эвристический метод разделения документов на фрагменты
2. Rewriter: запрос пользователя переписывается с использованием метода HyDE, что позволяет улучшить точность поиска релевантных данных
3. Retriever: реализация на основе алгоритма BM25
4. Reranker: реализованный с применением подхода Cross-Encoder и модели ru-bert2
5. Generator: на финальном этапе генерируется окончательный ответ с помощью API YandexGPT

Реализованный пайплайн использует функционал больших языковых моделей через

API, что позволило значительно упростить требования к вычислительным ресурсам машины, на которой система развертывается. Это решение не только не перегружает локальное оборудование, но и позволяет использовать более мощные языковые модели.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks // arXiv preprint arXiv:2005.11401. — 2020.
2. Seven Failure Points When Engineering a Retrieval Augmented Generation System // arXiv preprint arXiv:2401.05856. — 2024.
3. Query Rewriting for Retrieval-Augmented Large Language Models // arXiv preprint arXiv:2305.14283. — 2023.
4. Evaluation of Retrieval-Augmented Generation: A Survey // arXiv preprint arXiv:2405.07437. — 2024.
5. Re2G: Retrieve, Rerank, Generate // arXiv preprint arXiv:2207.06300. — 2022.
6. RAGAS: Automated Evaluation of Retrieval Augmented Generation // arXiv preprint arXiv:2309.15217. — 2023.
7. Searching for Best Practices in Retrieval-Augmented Generation // arXiv preprint arXiv:2407.01219. — 2024.
8. RAG and RAU: A Survey on Retrieval-Augmented Language Model in Natural Language Processing // arXiv preprint arXiv:2404.19543. — 2024.
9. Corrective Retrieval Augmented Generation // arXiv preprint arXiv:2401.15884. — 2024.
10. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models // arXiv preprint arXiv:2201.11903. — 2022.
11. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection // arXiv preprint arXiv:2310.11511. — 2023.
12. 5 Levels Of Text Splitting [Электронный ресурс]. — 2024. — Режим доступа: https://github.com/FullStackRetrieval-com/RetrievalTutorials/blob/main/tutorials/LevelsOfTextSplitting/5_Levels_Of_Text_Splitting.ipynb (дата обращения: 09.12.2024).
13. RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval // arXiv preprint arXiv:2401.18059. — 2024
14. A Dataset of Information-Seeking Questions and Answers Anchored in Research Papers // arXiv preprint arXiv:2105.03011. — 2021

ПРИЛОЖЕНИЕ А

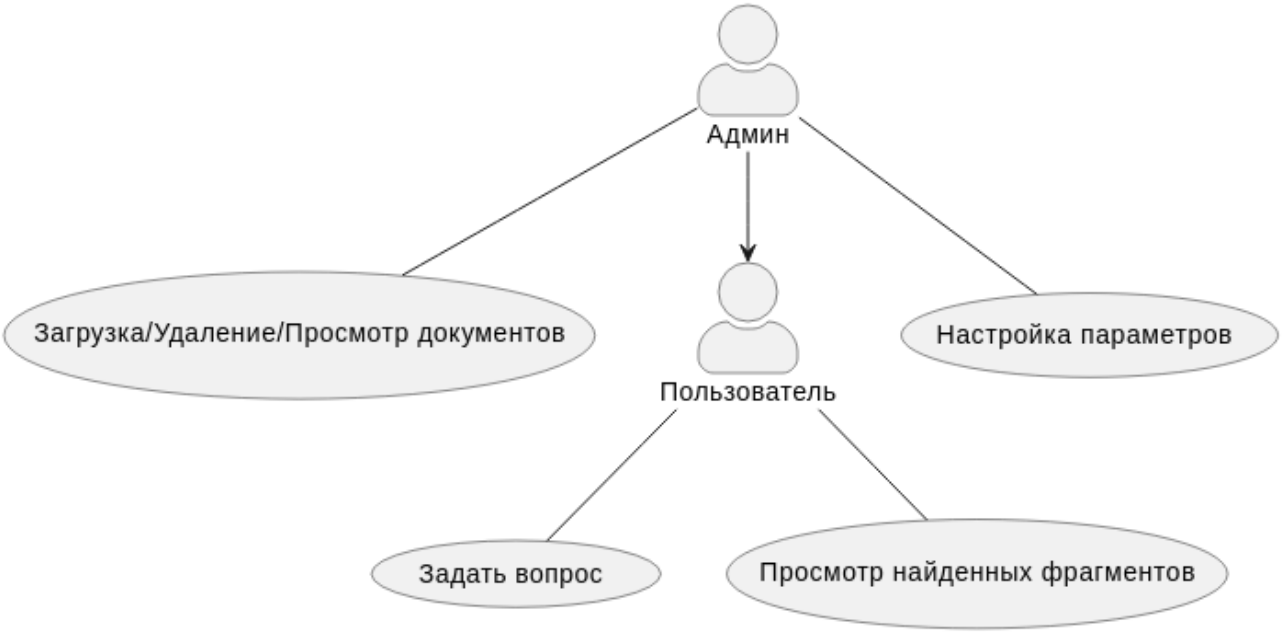
Листинг программного кода

```
import os
from gc import callbacks
from typing import Iterator
import inspect
# Environment
from environment import credentials, project_root
# Base
import torch
from langchain_core.messages import HumanMessage, SystemMessage, BaseMessage
from langchain.chains import SequentialChain
from langchain_core.runnables.config import RunnableConfig, Output, Input
from langchain.chains.base import Chain
from langchain_core.language_models.chat_models import BaseChatModel
from unstructured.documents.elements import Element
# Авторизация GigaChat
def init_gigachat(model="GigaChat"):
    return GigaChat(
        credentials=credentials.gigachat_authorization_key,
        scope=credentials.gigachat_scope,
        model=model,
        # Отключает проверку наличия сертификатов НУЦ Минцифры
        verify_ssl_certs=False,
        streaming=False
    )
```

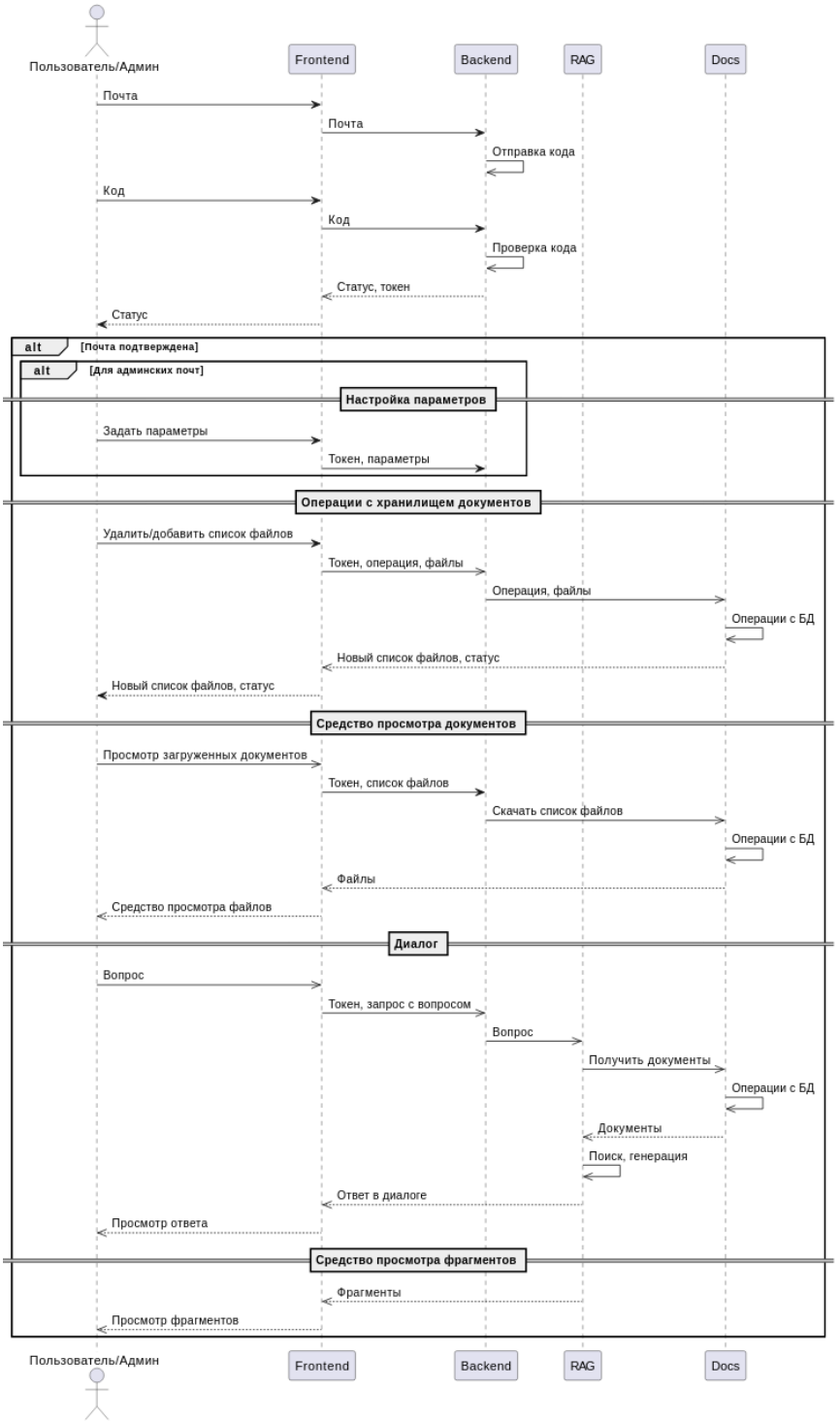
Пример скрипта для инициализации модели

ПРИЛОЖЕНИЕ Б

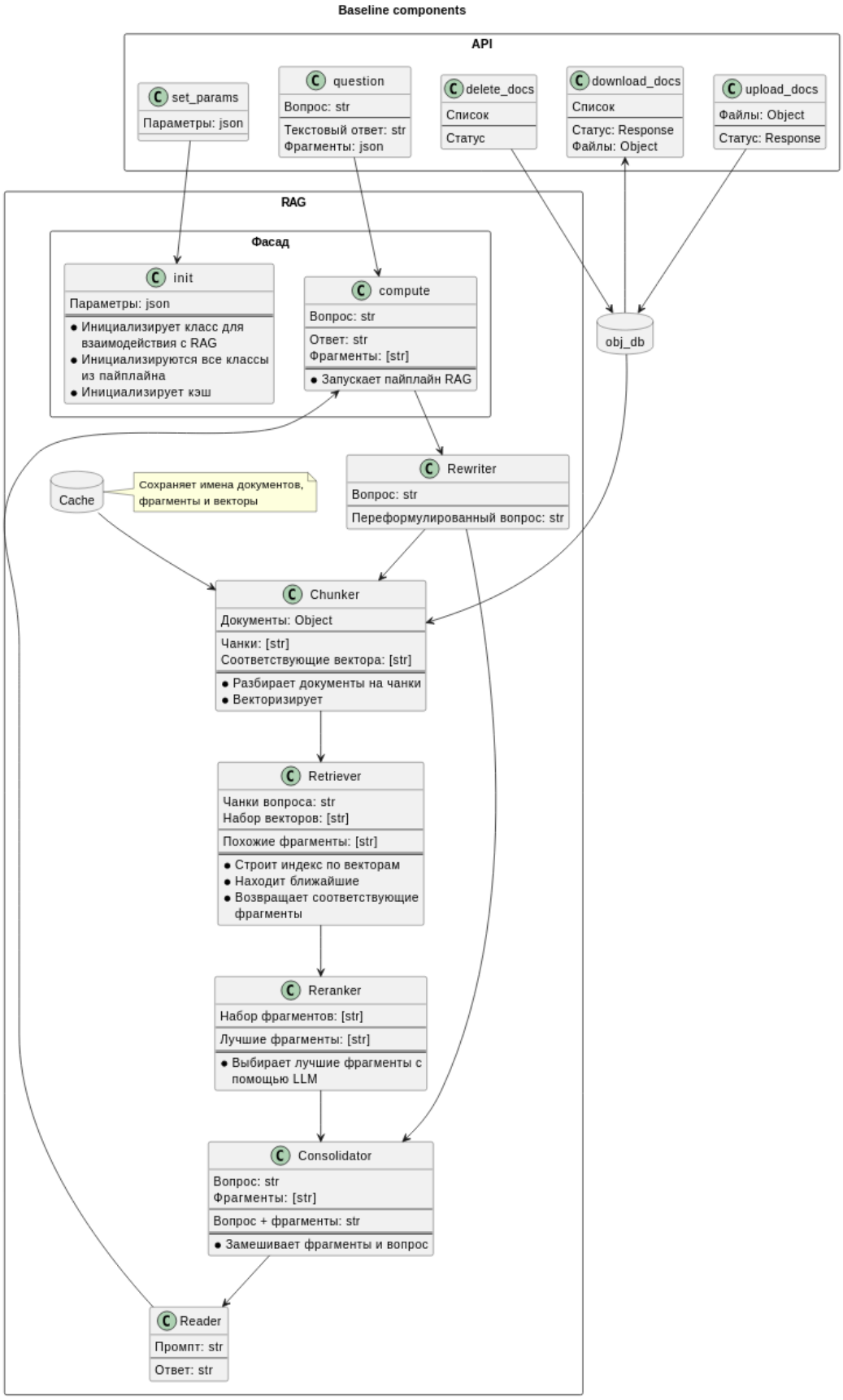
UML диаграммы



Use Case диаграмма



Main Sequence диаграмма



Baseline Components диаграмма

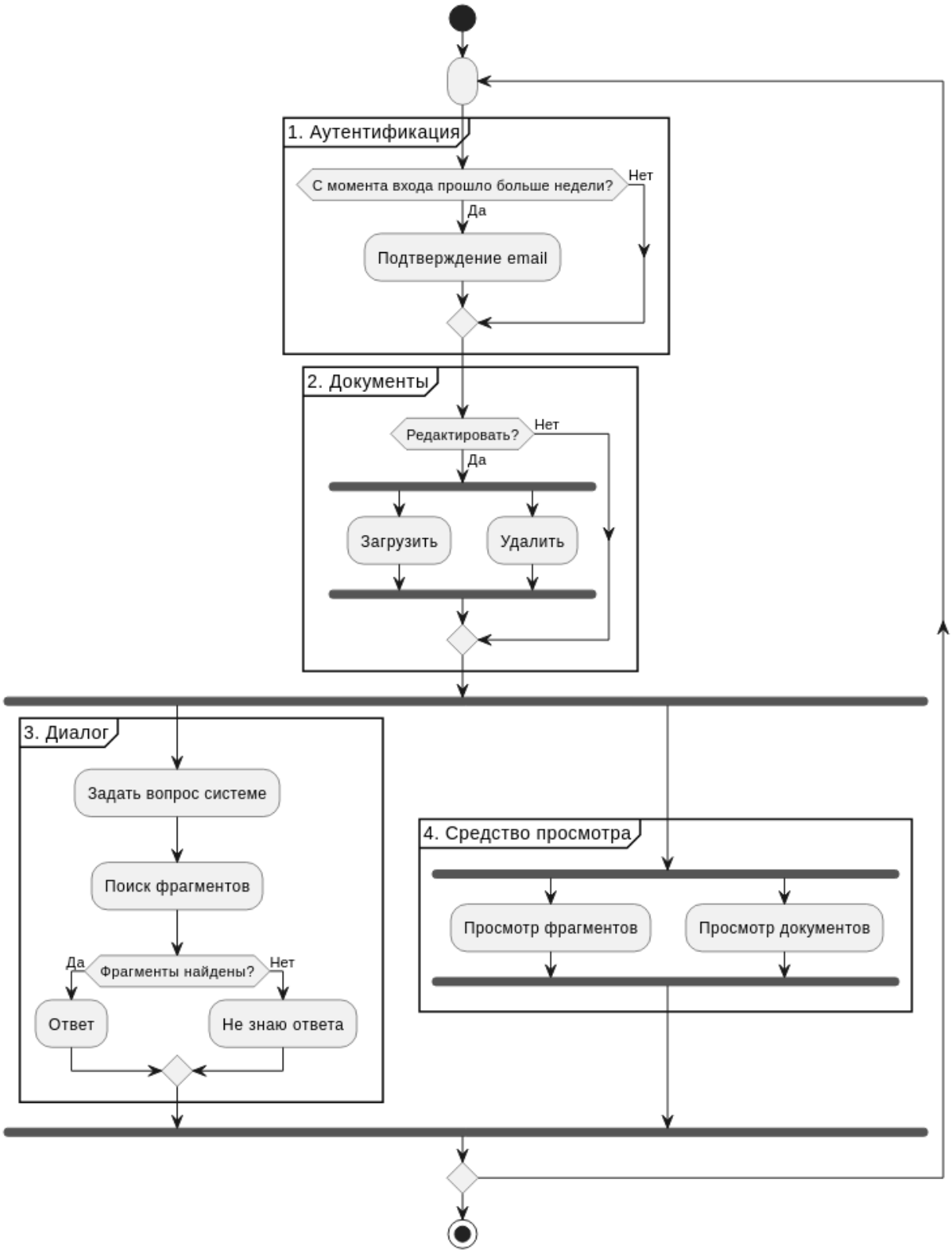


Диаграмма активностей пользователя

Выпускная квалификационная работа выполнена мной самостоятельно и с соблюдением правил профессиональной этики. Все использованные в работе материалы и заимствованные принципиальные положения (концепции) из опубликованной научной литературы и других источников имеют ссылки на них. Я несу ответственность за

приведенные данные и сделанные выводы.

Я ознакомлен с программой государственной итоговой аттестации, согласно которой обнаружение плагиата, фальсификации данных и ложного цитирования является основанием для не допуска к защите выпускной квалификационной работы и выставления оценки «неудовлетворительно».

_____	_____
ФИО студента	*Подпись студента*
« » 20 г.	
(заполняется от руки)	