Научная статья
УДК 004.853
doi:10.37614/2949.1215.2024.15.3.001

ИСПОЛЬЗОВАНИЕ RAG-ТЕХНОЛОГИИ ДЛЯ СОЗДАНИЯ ИНТЕЛЛЕКТУАЛЬНОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ ПОДДЕРЖКИ ИССЛЕДОВАТЕЛЬСКОГО ПОИСКА

Андрей Григорьевич Олейник[1], Андрей Михайлович Федоров[2]✉, Игорь Олегович Датьев[3], Александр Анатольевич Зуенко[4], Алексей Владимирович Шестаков[5], Иван Геннадьевич Вишняков[6]
[1–6]Институт информатики и математического моделирования имени В. А. Путилова Кольского научного центра Российской академии наук, Апатиты, Россия
[1]a.oleynik@ksc.ru, https://orcid.org/0000-0002-7612-5999
[2]a.fedorov@ksc.ru, https://orcid.org/0000-0002-2862-7994
[3]i.datyev@ksc.ru, https://orcid.org/0000-0002-8372-8704
[4]a.zuenko@ksc.ru, https://orcid.org/ 0000-0002-7165-6651
[5]a.shestakov@ksc.ru, https://orcid.org/0000-0002-9052-2579
[6]i.vishnyakov@ksc.ru, https://orcid.org/0009-0003-4938-5693

Аннотация

Работа посвящена практике интеграции большой языковой модели в систему интеллектуальной информационно-аналитической поддержки исследовательского поиска информации в заданных массивах данных. Проведен обзор возможностей современных больших языковых моделей. Особое внимание уделено технологии RAG, используемой для оперативного формирования и управления контекстом поиска. Описаны особенности проектирования и разработки функций бизнес-логики и элементов пользовательских интерфейсов системы. Представлены результаты экспериментов применения разработанной системы к текстам с заданной спецификой. Сделаны выводы о применимости системы для автоматизации процесса заселения онтологий, а также наполнения других хранилищ знаний, построенных с использованием автоматизированного мониторинга больших открытых данных и интеллектуальных средств их обработки.

Ключевые слова:

онтология, RAG-технология, нейронная сеть, большая языковая модель, LLM, информационно-аналитическая система, интеллектуальный анализ данных, проектирование программного обеспечения, разработка программного обеспечения, веб-сервис, веб-приложение

Финансирование:

работа выполнена в рамках НИР «Разработка теоретических и организационно-технических основ информационной поддержки управления жизнеспособностью региональных критических инфраструктур Арктической зоны Российской Федерации» (регистрационный номер 122022800547-3) и НИР «Методология создания информационно-аналитических систем поддержки управления региональным развитием, основанных на формирующем искусственном интеллекте и больших данных» (регистрационный номер 122022800551-0).

Для цитирования:

Олейник А. Г., Федоров А. М., Датьев И. О., Зуенко А. А., Шестаков А. В., Вишняков И. Г. Использование RAG-технологии для создания интеллектуальной информационной системы поддержки исследовательского поиска // Труды Кольского научного центра РАН. Серия: Технические науки. 2024 Т. 15, № 3. С. 5–26. doi:10.37614/2949.1215.2024.15.3.001.

Original article

USING RAG TECHNOLOGY TO DESIGN AN INTELLIGENT INFORMATION SYSTEM FOR SUPPORT EXPLORATORY SEARCH

Andrey G. Oleynik[1], Andrey. M. Fedorov[2]✉, Igor O. Datyev[3], Alexander A. Zuenko[4], Aleksey V. Shestakov[5], Ivan G. Vishnyakov[6]
[1–6]Putilov Institute for Informatics and Mathematical Modeling of the Kola Science Centre of the Russian Academy of Sciences, Apatity, Russia
[1]a.oleynik@ksc.ru, https://orcid.org/0000-0002-7612-5999
[2]a.fedorov@ksc.ru, https://orcid.org/0000-0002-2862-7994
[3]i.datyev@ksc.ru, https://orcid.org/0000-0002-8372-8704
[4]a.zuenko@ksc.ru, https://orcid.org/ 0000-0002-7165-6651
[5]a.shestakov@ksc.ru, https://orcid.org/0000-0002-9052-2579
[6]i.vishnyakov@ksc.ru, https://orcid.org/0009-0003-4938-5693

Abstract

This paper presents the integration of a large language model (LLM) into an intelligent information and analytical system for exploratory search support within specific texts. An overview of the capabilities of modern large language models is provided. Special attention is paid to the Retrieval Augmented Generation (RAG) technology used for efficient formation and control of search contexts. The design and development of the system's business logic functions and user interface elements are detailed. Experimental results applying the developed system to texts with specific characteristics are presented. Conclusions are drawn regarding the system's applicability for automating ontology population. Conclusions are drawn regarding the system's applicability for automating the process of populating ontologies, as well as filling other knowledge repositories built using automated monitoring of open big data and intelligent data processing.

Keywords:

ontology, retrieval augmented generation, RAG, neural network, large language model, LLM, information and analytical system, data mining, software design, software development, web service, web application

Acknowledgments:

the study was carried out within the framework of the Putilov Institute for Informatics and Mathematical Modeling of the Kola Science Centre of the Russian Academy of Sciences state assignment of the Ministry of Science and Higher Education of the Russian Federation, research topic “Development of theoretical and organizational and technical foundations of information support for managing the viability of regional critical infrastructures of the Arctic zone of the Russian Federation” (registration number of the research topic 122022800547-3) and research topic “Methodology for creating information and analytical systems to support regional development management based on formative artificial intelligence and big data” (registration number of the research topic 122022800551-0).

For citation:

Andrey G. Oleynik, Igor O. Datyev, Alexander A. Zuenko, Andrey. M. Fedorov, Aleksey V. Shestakov, Ivan G. Vishnyakov Using RAG technology to design an intelligent information system for support exploratory search // Transactions of the Kola Science Centre of RAS. Series: Engineering Sciences. 2023. Vol. 15, No. 3. P. 5–26. doi:10.37614/2949.1215.2024.15.3.001.

Введение

Плановые исследования Института информатики и математического моделирования им. В. А. Путилова (далее — ИИММ) включают разработку решений ряда актуальных научных задач, связанных с созданием эффективных методов формирования баз знаний и интеллектуальных информационных систем в условиях сложности, неопределенности, разнородности и динамичности предметной области, а также методов аналитической обработки данных и знаний, ориентированных на проблематику регионального развития и поддержку соответствующего ему управления. Одним из важных направлений является разработка теоретических и организационно-технических основ информационной поддержки управления жизнеспособностью региональных критических инфраструктур Арктической зоны Российской Федерации. Эффективное решение задач управления требует повышения уровня согласованности как нормативно-правовой базы, регламентирующей различные виды деятельности, так и «мультидисциплинарного» понятийного аппарата, используемого в процессах принятия решений в различных сферах и на различных уровнях управления. Поэтому в рамках тематики ИИММ предполагается развить методологию создания интеллектуальных проблемно-ориентированных информационных систем на базе концепции «формирующего искусственного интеллекта».

Так, в ходе разработки основ информационной поддержки управления жизнеспособностью региональных критических инфраструктур Арктической зоны Российской Федерации требует решения задача четкой аргументированной идентификации и классификации объектов, которые могут быть «слабым звеном» включающих их или связанных с ними инфраструктур. К этим объектам прежде всего относятся критически важные объекты (КВО) и потенциально опасные объекты (ПОО) [1]. В качестве инструмента решения данной задачи могут быть использованы формальные онтологии, позволяющие сформировать согласованный понятийный базис для специалистов различных предметных областей [2]. В работе [3] была представлена технология согласования нормативно-правовых документов на основе онтологического подхода, использование которой могло бы способствовать формированию целостной системы норм и правил функционирования различных акторов как на региональном уровне, так и их взаимодействия с уровнем федеральным.

Подобные задачи также возникают при создании информационно-аналитических систем, нацеленных на обеспечение информационной поддержки задач управления региональным развитием. Региональная социально-экономическая система рассматривается как взаимосвязанная триада «бизнес-сообщество-власть» [4]. Элементам такой системы и ее критической инфраструктуре свойственны

неопределенность и спонтанная динамика в виду наличия факторов риска. Анализировать и управлять объектами и процессами с подобной спецификой позволяют системы информационной поддержки на основе интеллектуализированных компонент в виде семантических моделей, онтологий и других хранилищ знаний, построенных с использованием автоматизированного мониторинга больших открытых данных (например, контента социальных сетей) [5] и интеллектуальных средств их обработки [6].

Несмотря на развитие методов, технологий и инструментальных средств онтологического проектирования, построение конкретных проблемно-ориентированных онтологий остается трудоемкой задачей. Одним из путей технической поддержки создания и наполнения/пополнения онтологий стало использование инструментов искусственного интеллекта. В первую очередь это инструменты работы с текстами на естественном языке. Одной из подзадач обучения онтологии — ее «заселение» (Ontology Population — OP), которая подразумевает расширение существующей онтологии новыми экземплярами без изменения структуры онтологии. Варианту решения этой подзадачи посвящена, в частности, работа [7]. В свою очередь, в исследовании [8], являющемся логическим продолжением работы [7], предлагается вариант решения задачи извлечения отношений из текстов предметной области для их возможного добавления в существующую онтологию. Вопросам извлечения сущностей и отношений посвящена и статья [9], представляющая технологию генеративных многошаговых вопрос-ответов (generative multi-turn question answering). В ней также кратко представлен обзор ряда существующих подходов к решению этой задачи. Отмечено, что недостатком конвейерного режима распознавания сущностей является распространение ошибок. Даются ссылки на публикации по различным стратегиям реализации методов совместного обучения с использованием взаимосвязи задач, подходам к совместному извлечению сущностей и отношений как проблеме заполнения таблиц, а также ряду других технологий.

Как указанные, так и ряд других публикаций представляют технологии использования нейросетевых моделей обработки естественноязыковых текстов, предполагающие предварительное обучение, или дообучение этих моделей на объемных текстовых корпусах. Данная процедура сама по себе является довольно ресурсоемкой. Кроме того, такой подход может оказаться неэффективным в случае, когда контекст, с которым работает модель, существенно изменяется со временем. В этом случае начальная база знаний, сформированная на основе контекста, на котором проходило обучение модели, становится неактуальной и полученные ранее «знания» могут повлечь ошибочные для изменившейся ситуации ответы, формируемые моделью. Подобная ситуация может возникнуть, если «актуальный» контекст представляет собой, например, документацию развивающегося программного продукта или изменяющиеся нормативно-правовые документы. Постоянное дообучение модели с целью актуализации ее базы знаний дорого и не эффективно. Более эффективным решением в этом случае представляется использование технологии RAG (Retrieval Augmented Generation) [10]. Данная технология позволяет исключить ресурсоемкий этап дообучения языковой модели при создании проблемно-ориентированных приложений на основе хорошо обученных и уже готовых к использованию языковых моделей таких, как YandexGPT [11], GigaChat [12], OpenAI [13] и др.

Настоящая статья представляет начальный этап исследований коллектива авторов по применению RAG-технологий в работе со справочными и нормативными текстами для создания как автономных интеллектуализированных информационно-справочных систем, так подсистем аналитической поддержки, интегрируемых в сторонние информационные комплексы.

Используемые в работе технологии интеллектуализированной обработки данных

RAG-технология является одним из новых этапов развития технологий работы с текстами на естественном языке, использующих большие языковые модели (Large Language Models, LLM) [14]. LLM представляют собой нейронные сети, основанные на архитектуре трансформер [15] и обученные на больших объемах данных обрабатывать и генерировать текст на естественном языке. Среди задач, для решения которых применяются LLM, наибольший интерес в рамках проводимого исследования представляют обработка запросов и генерация ответов на вопросы пользователя в режиме диалога, а также генерация контекстуализированного текста на основе предоставленных вводных данных. Это позволяет создавать с использованием LLM «интеллектуальные» информационные системы,

взаимодействующие с пользователем на естественном языке, что, в свою очередь, снижает требования к технической подготовке пользователя. Однако априорно обученные LLM обладают существенными ограничениями при обработке запросов, требующих использования «новой» (выходящих за рамки данных, на которых они обучены) информации. В таких ситуациях LLM нередко генерируют «галлюцинации» — ответы, не имеющие отношения к действительности [16].

Во многом преодолеть проблемы, обусловленные ограничениями непосредственного использования LLM, позволяет RAG-технология и ее вариации. В рамках RAG-технологии реализуются механизмы поиска в базе данных информационной системы фрагментов текста, вектора которых наиболее релевантны векторизованному вопросу пользователя. На основе найденных фрагментов с использованием «знаний» LLM генерируется текст ответа. Это позволяет улучшить качество ответа системы на вопрос за счет учета конкретного контекста. Возможность оперативного изменения базы данных системы обуславливает ее адаптивность к изменению данных без необходимости полного переобучения всей модели или ее дообучения, для чего в общем случае требуются значительные вычислительные и временные ресурсы [17, 18].

При этом очевидно, что качество генерируемых системой ответов будет существенно зависеть от полноты и адекватности имеющейся базы данных. Поэтому оценка влияния подготовки данных на качество ответов системы является одной из основных задач начального этапа проводимого исследования.

В статье [10] представлен обзор эволюции RAG-технологии и довольно подробно рассматриваются методы работы трех основных компонентов этих технологий — поиска (Retrieval), дополнения (Augment) и генерации (Generation). Авторы этого обзора выделяют три этапа развития RAG, отличающиеся развитием инструментов тонкой настройки каждого из базовых компонентов.

Первый этап определен как простой, или «наивный» (Naive RAG). В Naive RAG [19] реализуются процедуры индексации исходных данных, их сегментация и векторизация. Векторизованные фрагменты сохраняются в базе данных информационной системы. Аналогичным образом происходит обработка и векторизация поступившего в систему вопроса пользователя. Процедуры извлечения из базы фрагментов, наиболее релевантных поступившему вопросу, основаны на вычислении оценки сходства между вектором вопроса и векторами хранимых фрагментов. Наиболее релевантные фрагменты, извлеченные из базы, совместно с текстом вопроса в дальнейшем включаются в расширенный контекст подсказки для ответа на вопрос, который должна сгенерировать используемая в системе предобученная LLM. Более того, уже и Naive RAG поддерживает и режим «диалога», когда в подсказку могут быть интегрированы вопросы и ответы, предшествующие текущему вопросу.

На втором этапе развития, названном Advanced RAG, вводятся дополнительные процедуры, повышающие качество поиска за счет усовершенствованных механизмов индексации и включения метаданных, позволяющие более точно соотнести извлекаемые данные с вопросом [20].

Третий этап характеризуется вводом в RAG-систему дополнительных компонентов, повышающих ее адаптивность и улучшающих возможности извлечения и обработки целевого контента [10]. Данный этап назван «Модульным» (Modular RAG).

Для реализации инструментальной среды проблемно-ориентированного использования RAG-технологии был применен фреймворк LangChain [21]. Он обеспечивает возможность интеграции и комбинирования различных компонентов обработки естественного языка и возможность создания адаптивных решений, которые могут взаимодействовать с внешними данными и контекстом пользователя. Поддержка LangChain работы с различными предобученными языковыми моделями, в том числе и с локальными, дает возможность исследовать и сравнивать эффективность разных LLM при решении различных задач. К основным компонентам LangChain относятся: API-интерфейсы; шаблоны подсказок (prompt patterns) различных типов; инструменты работы с индексами и инструменты создания и использования цепочек автоматизированного выполнения операций; функции запоминания как текущего «диалога», так и предшествующих. Также LangChain обладаем инструментами преобразования, хранения, поиска и извлечения информации, что позволяет создавать RAG-системы.

Организационные, методологические и практические особенности проектирования и разработки системы

Представленная разработка опирается на идею по созданию интеллектуализированной справочной системы, способной выдавать пользователю ответы на основе предварительно заданного

набора документов. На этапе проектирования классический механизм формирования ответов на основе полнотекстового синтаксического поиска был определен как недостаточный. Ответы разрабатываемой системы должны учитывать неявные семантические взаимосвязи между различными фрагментами загружаемого набора документов. С учетом указанных требований для построения системы выбраны технологии, используемые при разработке чат-ботов. В частности, основным рабочим механизмом для реализации системы выбраны большие языковые модели и технология RAG для оперативного доступа к ним.

Указанный набор инструментов и технологий позволил реализовать требуемую базовую функциональность системы. Дополнительно возникла необходимость расширить область применения разработки и использовать ее не только в качестве интеллектуальной справки, но и как исследовательский инструмент для изучения особенностей языковых моделей и эффективных способов взаимодействия с ними.

На рис. 1 в виде UML-диаграммы использования представлены формальные требования к функциональным возможностям информационной системы интеллектуальной справки.

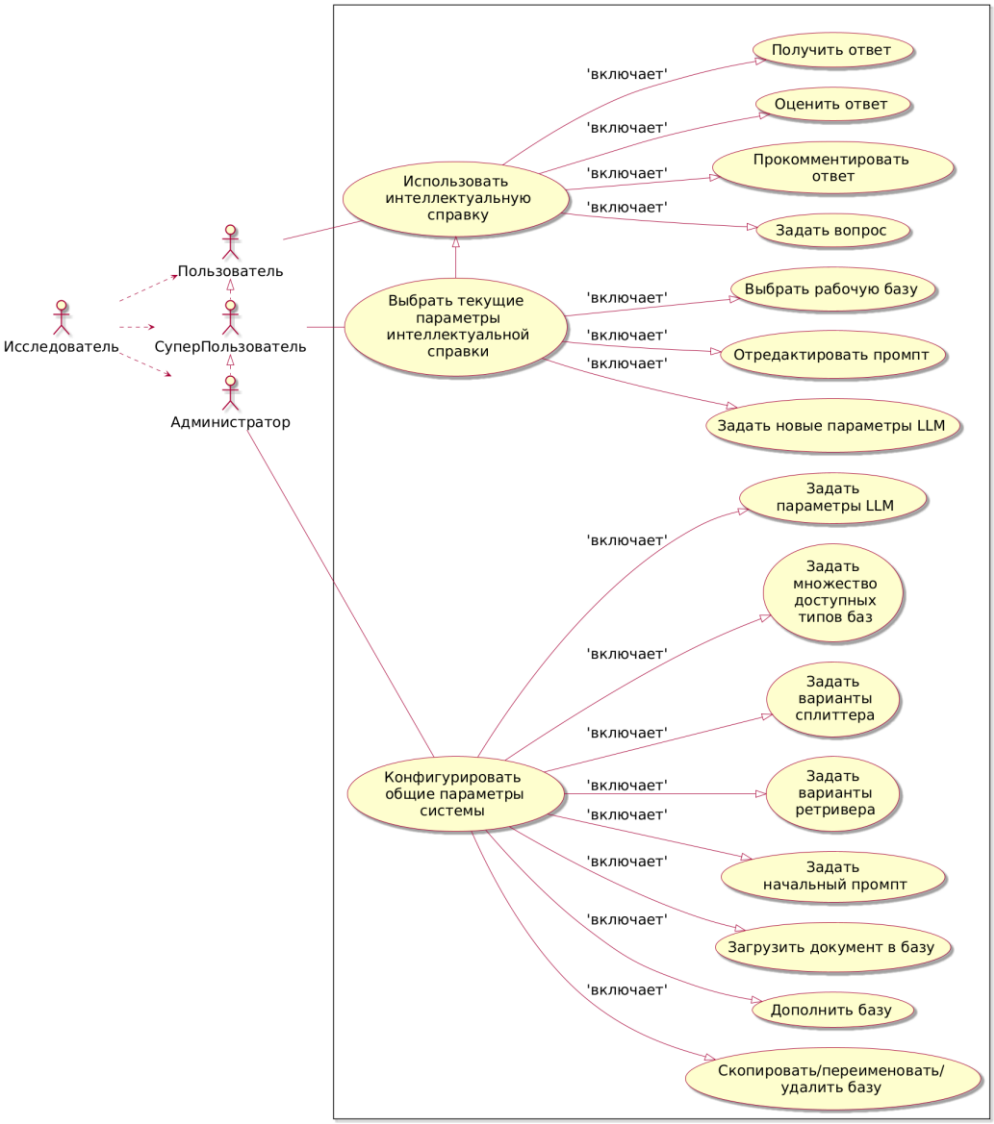

Рис. 1. UML-диаграмма использования — требования к функциональным возможностям информационной системы интеллектуальной справки

Для удобства использования и администрирования в системе реализована классическая ролевая пользовательская модель. Обобщенная роль «Исследователь» специализирована конкретными ролями «Пользователь», «СуперПользователь», «Администратор». Специализация задает уровни доступа пользователей к параметрам и функциями системы. «Администраторам» доступны все инструменты для использования и конфигурирования всех параметров системы. «СуперПользователи», помимо использования, могут изменять текущие параметры интеллектуальной справки. Для «Пользователей» предоставляется доступ к преднастроенным основным функциями справки.

Все функции в системе отнесены к следующим группам: 1) конфигурирование общих параметров системы; 2) настройка текущих параметров конфигурации; 3) непосредственное использование справочной системы.

Первая группа функций предназначена для администраторов системы. В нее входят функции загрузки наборов документов и управления соответствующими базами данных, а также функции для настройки необходимых операций по подготовке документов и их размещению в базах.

Вторую группу функций используют привилегированные пользователи («СуперПользователи») для предварительной настройки системы непосредственно в процессе ее работы. В частности, здесь реализованы действия по выбору рабочей базы из списка ранее загруженных баз данных, а также выбор опций для формирования запросов (промптов) к языковой модели.

Основная группа функций используется обыкновенными пользователями для непосредственной работы со справочной системой. Именно этот набор интерфейсных вызовов был основным требованием к разрабатываемой системе. Здесь сгруппированы действия: сформировать вопрос, отправить его на вход языковой модели, получить и оценить ответ.

В выборе инструментария для разработки предпочтение отдавалось отечественным и платформонезависимым технологиям, программам и системам. В рамках данного проекта для координации и совместной работы команды разработчиков и исследователей использованы:

— GitLab — развернутая на серверной инфраструктуре ИИММ система управления проектами и репозиториями программного кода;

— Яндекс.Диск, Яндекс.Документы — облачная инфраструктура для создания и совместного редактирования документов;

— «Телеграм» — приложение для обмена мгновенными сообщениями.

Для непосредственной разработки активно использовались:

— PyCharm Community — кроссплатформенная интегрированная среда разработки для языка программирования Python;

— VS Codium — свободно лицензируемый дистрибутив кроссплатформенного редактора для разработки различных приложений VS Code (Microsoft);

— Python 3.11.7 — объектно-ориентированный интерактивный язык программирования;

— ALT Linux — отечественный дистрибутив ОС Linux, основанный на ядре Linux и репозитории пакетов Sisyphus. ОС на рабочей станции разработчиков;

— Debian 12 (Bookworm) — операционная система с открытым исходным кодом GNU/Linux. ОС на облачном сервере с конфигурацией 1×3.3 ГГц, CPU 1 Гб, RAM 15 Гб NVMe для развертывания компонентов приложения.

Все используемые в данной работе программные системы, информационные технологии и подходы к коллективной разработке приложений ранее эффективно опробованы авторами как командой разработчиков и успешно зарекомендовали себя на практических этапах [22, 23] в научно-исследовательских работах по поддержке управления региональным развитием [24] и поддержке управления жизнеспособностью региональных критических инфраструктур [25], выполненных в рамках плановых НИР в ИИММ.

Общая архитектура и принципы построения разработанной системы

Архитектура разработанной системы представлена в виде UML-диаграммы развертывания (deployment diagram) на рис. 2. В разработке применена технология докеризации. В настоящее время

в системе используется один докер-компонент, который построен на основе последней версии образа контейнера “langchain/langchain” из репозитория dockerhub. Докеризация позволяет эффективно управлять ресурсами в процессе разработки, гибко разворачивать компоненты проекта в различные рабочие конфигурации, а также масштабировать и планировать дальнейшее развитие полученного решения.

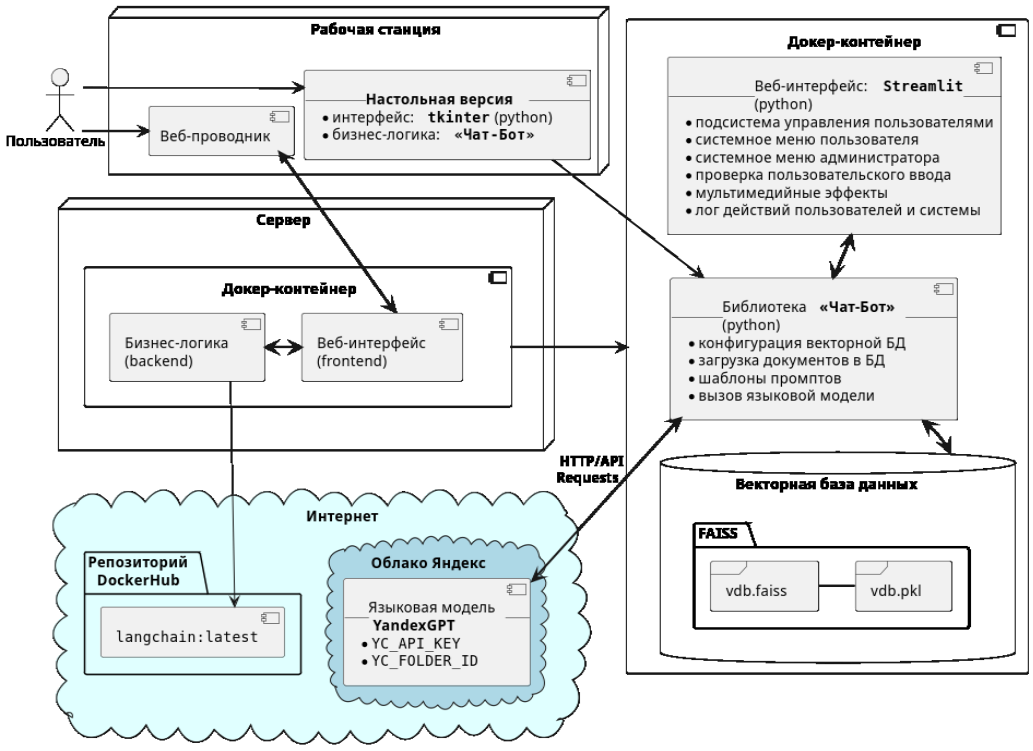

Рис. 2. Архитектура разработанной системы (UML-диаграмма развертывания)

Логически система разделена на два компонента. Реализующий бизнес-логику компонент (серверная часть, бэкенд) выполнен в виде библиотеки на языке Python 3. С помощью функций этой библиотеки производятся подготовка всех необходимых структур данных и обращения к языковой модели YandexGPT. Для обеспечения более универсального обращения к выбранной языковой модели и к ее аналогам используется библиотека LangChain, все необходимые компоненты которой предустановлены в используемом образе контейнера. Технология RAG реализуется через использование локальной векторной базы данных FAISS, компоненты которой размещаются в файловой системе докера.

Вторая составляющая системы — это пользовательский веб-интерфейс (фронтенд), также выполненный на языке Python 3 на основе фреймворка Streamlit [26]. Использование языка Python и для бизнес-логики, и для веб-интерфейса позволило естественным образом соединить два этих компонента. Библиотека для работы с языковой моделью импортирована в основной программный модуль, в котором в виде веб-интерфейсов реализована общая логика работы системы.

Дополнительно в разделе «Разработка пользовательского интерфейса» дано описание альтернативной версии интерфейса, разработанной для специального режима запуска системы в формате настольного приложения.

Принципы реализации основных функциональных элементов системы

К основным функциям серверной части системы относятся: 1) загрузка документов в базу данных и 2) обращение к языковой модели для получения ответа на вопрос.

Все функции реализованной библиотеки являются атомарными и работают независимо друг от друга. Каждая функция получает на вход все необходимые для работы данные в виде непосредственных значений из передаваемых параметров и в виде инициализирующих значений из конфигурационных файлов. Полученные в ходе выполнения функции данные либо возвращаются как результат функции, либо сохраняются в заранее определенных структурах — конфигурационных файлах, базах данных. Возникающие в процессе работы функций ошибки и нештатные ситуации обрабатываются и отображаются для пользователя в виде информативных сервисных сообщений. В программном коде широко используется стандартный для языка Python механизм обработки исключительных ситуаций (try...catch). Это позволяет сохранять работоспособность системы на приемлемом для решения поставленных задач уровне.

Сообщения о возникающих в системе событиях, ошибках, штатных и нештатных ситуациях записываются в файл протокола (лог-файл) и доступны администраторам системы для проведения последующих процедур аудита и анализа.

Загрузка документов в базу данных

На рис. 3 в виде UML-диаграммы последовательности (sequence diagram) представлен процесс загрузки документов в векторную базу данных.

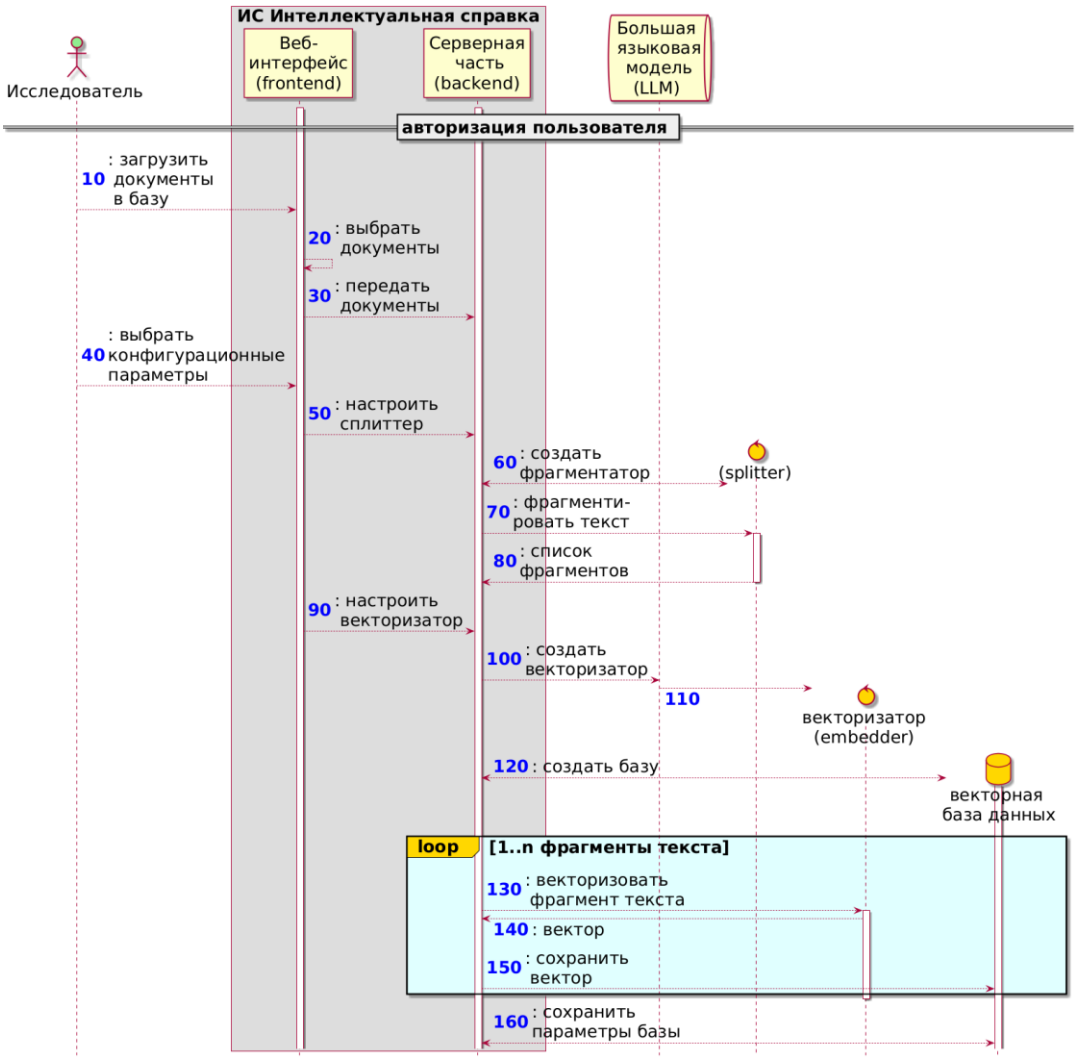

Рис. 3. UML-диаграмма последовательности: загрузка документов в векторную базу данных

В соответствии особенностями технологии RAG неотъемлемой частью разработанной системы является векторная база данных, которая используется для формирования контекстной части запроса (промпта) к языковой модели. Необходимо отметить, что используемые для работы вектора (эмбеддинги) можно хранить в альтернативных структурах, например, в файлах на диске или в списках в оперативной памяти. В данной работе для удобства реализации технологии RAG, в том числе для обеспечения возможностей адаптивной масштабируемости, гибкой развертываемости и оперативной интеграции, используется векторная база данных FAISS.

Загрузка документов в базы данных производится привилегированным пользователем в рамках предварительной настройки системы. Дизайн пользовательского интерфейса выполнен в классическом стиле, с использованием стандартных интерактивных веб-элементов из фреймворка Streamlit (рис. 4, 5).

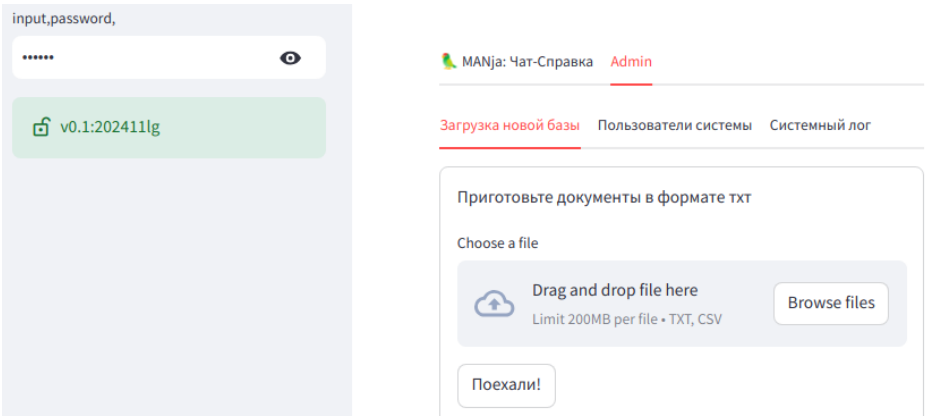

Рис. 4. Интерфейс пользователя: загрузка документов в базу данных

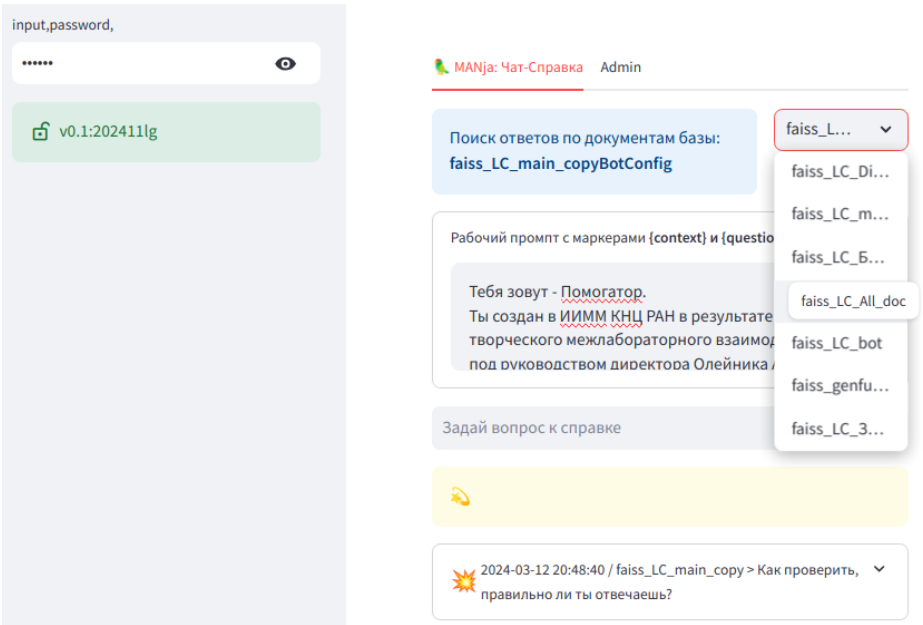

Рис. 5. Интерфейс пользователя: выбор рабочей базы данных

Большая часть представленных на UML-диаграмме операций скрыта от пользователя. После указания загружаемых документов пользователь получает сообщение о статусе загрузки и может продолжать работу с системой. Однако некоторые параметры этих операций представляют собой

исследовательский интерес. Поэтому в системе есть возможность предварительно настраивать эти параметры и сохранять настройки в конфигурационном файле, который и будет в дальнейшем использоваться для загрузки документов.

Подготовка запросов

Процесс работы пользователя с системой представлен UML-диаграммой последовательности (рис. 6).

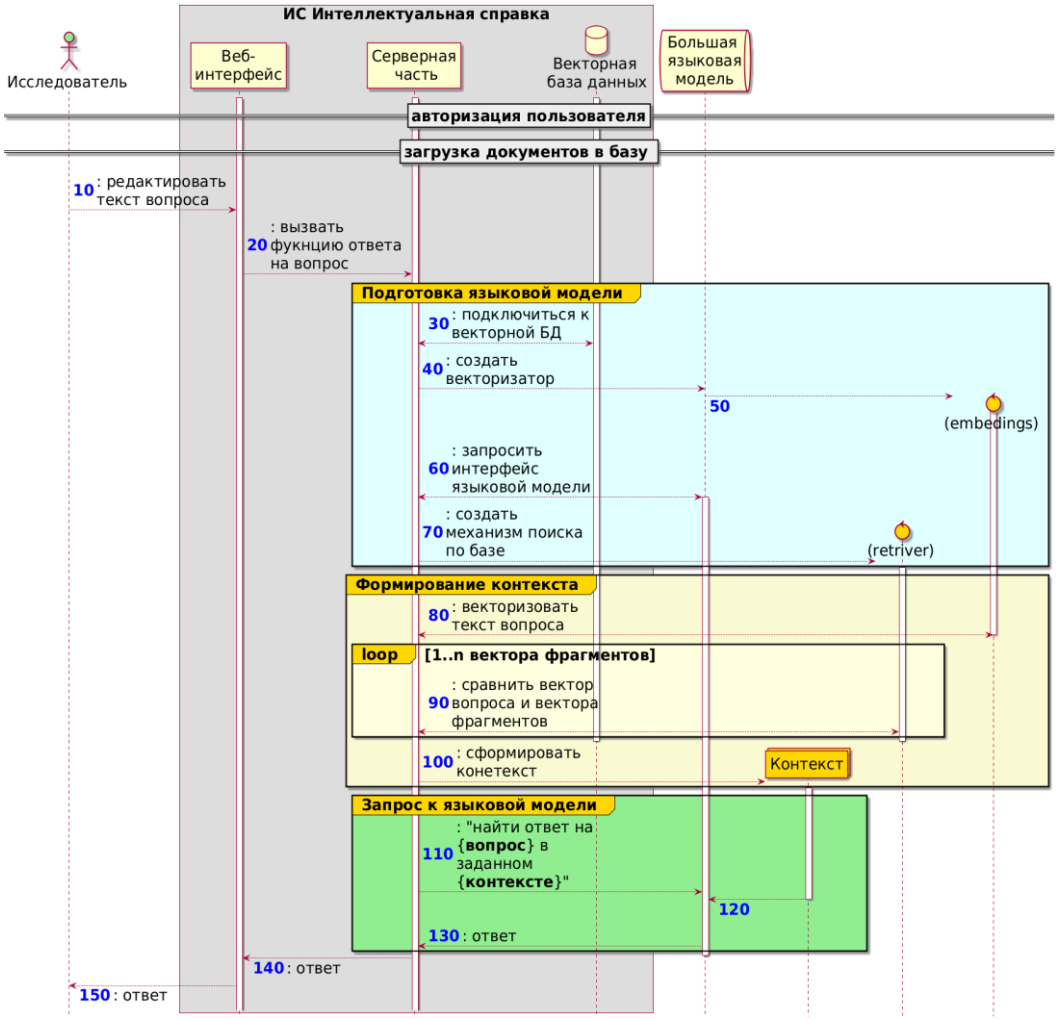

Рис. 6. UML-диаграмма последовательности: запросы пользователя к языковой модели

Для всех пользователей системы рабочим режимом по умолчанию является режим «вопрос-ответ». На первом шаге пользователь формирует в свободной форме вопрос и после подтверждения ввода получает ответ. Список заданных вопросов и полученных на них ответов сохраняется и доступен пользователю в течение активной сессии. Также пользователю предоставляется возможность оценить полученный ответ. Протокол сессии с вопросами, ответами и оценками в дальнейшем можно выгрузить и использовать для оценки качества работы системы.

Для обычных пользователей применяется преднастроенный администратором системы промпт запроса. Пользователям с более высоким уровнем привилегий доступна возможность корректировки промпта. Такая корректировка предполагает формулировку инструкции с использованием в тексте двух обязательных маркеров {context} и {question} (рис. 7). На следующем шаге эти маркеры автоматически заменяются соответственно: а) на контекст, т. е. извлеченный из векторной базы знаний

набор фрагментов; б) и на текст самого вопроса. Объем сформированного таким образом текста промпта не должен превышать допустимого моделью значения. Пример компоновки и наполнения интерфейса на этапе предоставления пользователю результатов запроса представлен на рис. 8.

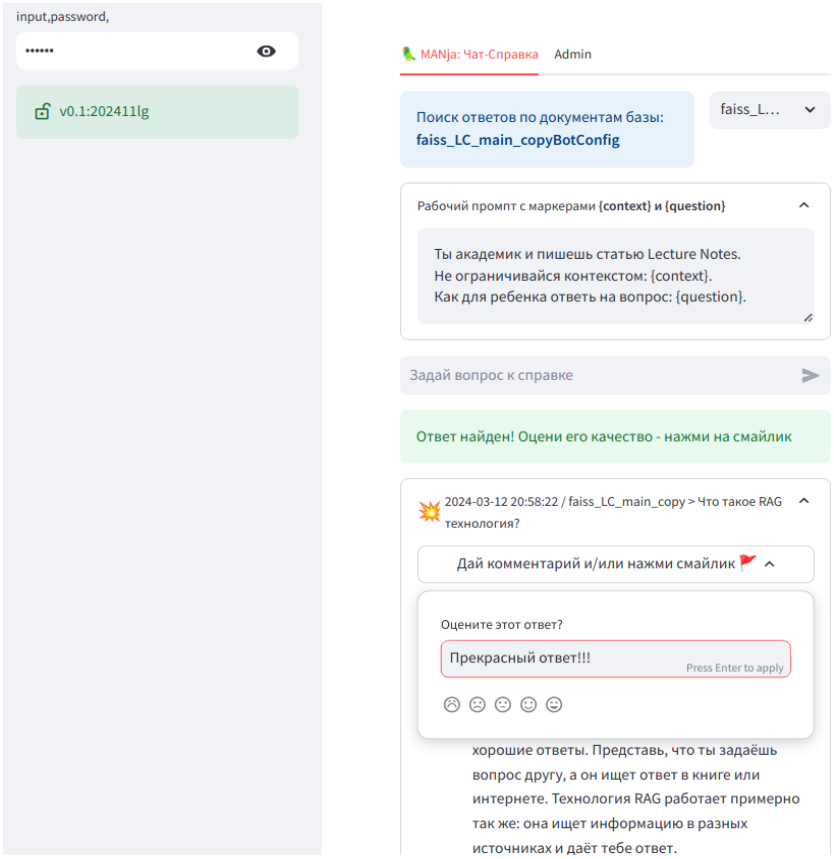

Рис. 7. Интерфейс пользователя: формирование промпта и оценка полученного ответа

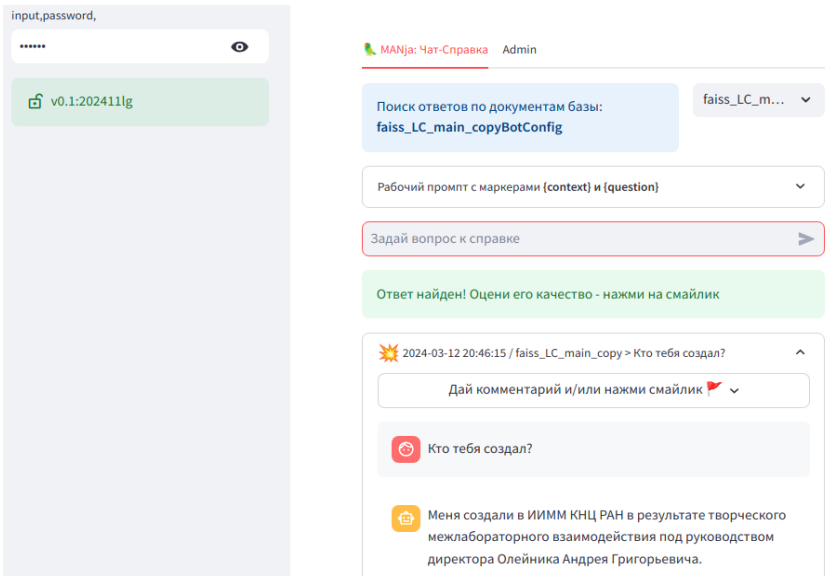

Рис. 8. Интерфейс пользователя: получение ответа на заданный вопрос

Разработка пользовательского интерфейса

Разработанная система построена в соответствии с классической веб-архитектурой и разделена на серверную (бэкенд) и пользовательскую (фронтенд) части. Это позволило эффективно управлять ограниченными ресурсами команды разработчиков, а также заложить в систему возможности по ее оперативному развертыванию, гибкой интеграции и адаптивной масштабируемости.

Использование веб-интерфейса позволяет системе работать в режиме онлайн и обеспечивает обращение к ней одновременно нескольких пользователей. Этот формат работы с системой необходим для решения целевых практических задач, например, при интеграции данной интеллектуальной справочной системы в сторонние предметно-ориентированные информационные системы. Примеры компоновки интерфейса представлены в предыдущем разделе и ниже на рис. 9–12.

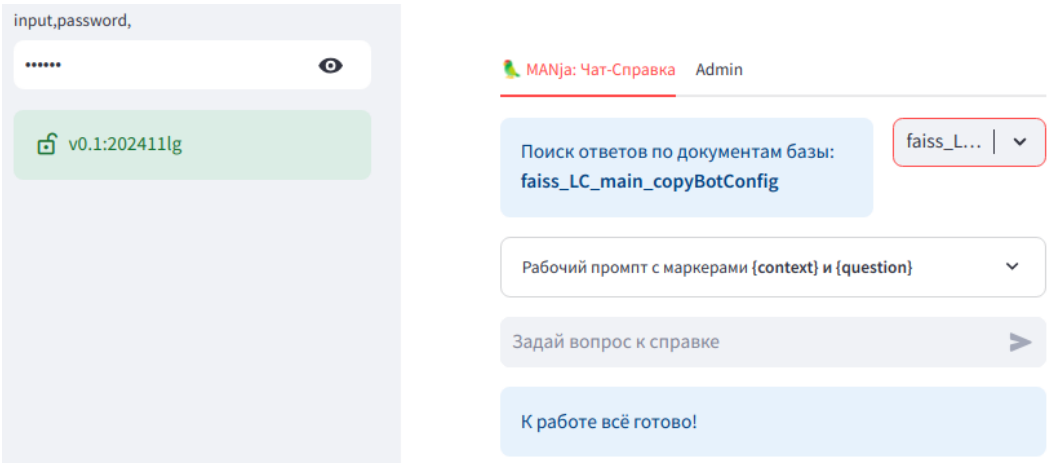

Рис. 9. Пользовательский интерфейс: начало работы

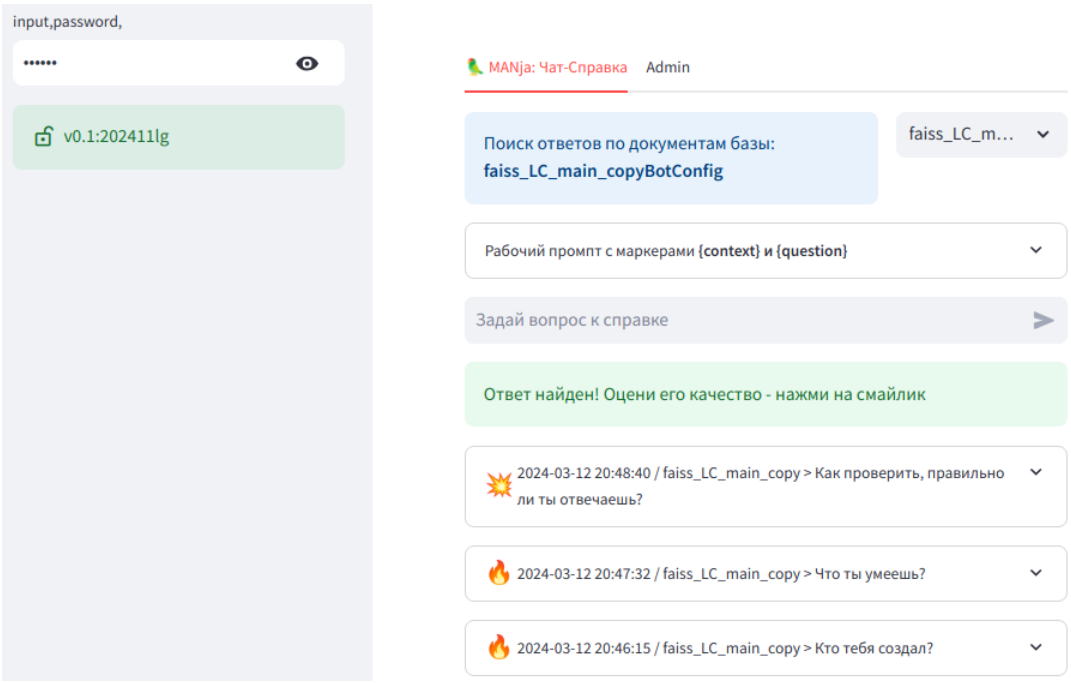

Рис. 10. Пользовательский интерфейс: режим работы «вопрос-ответ»

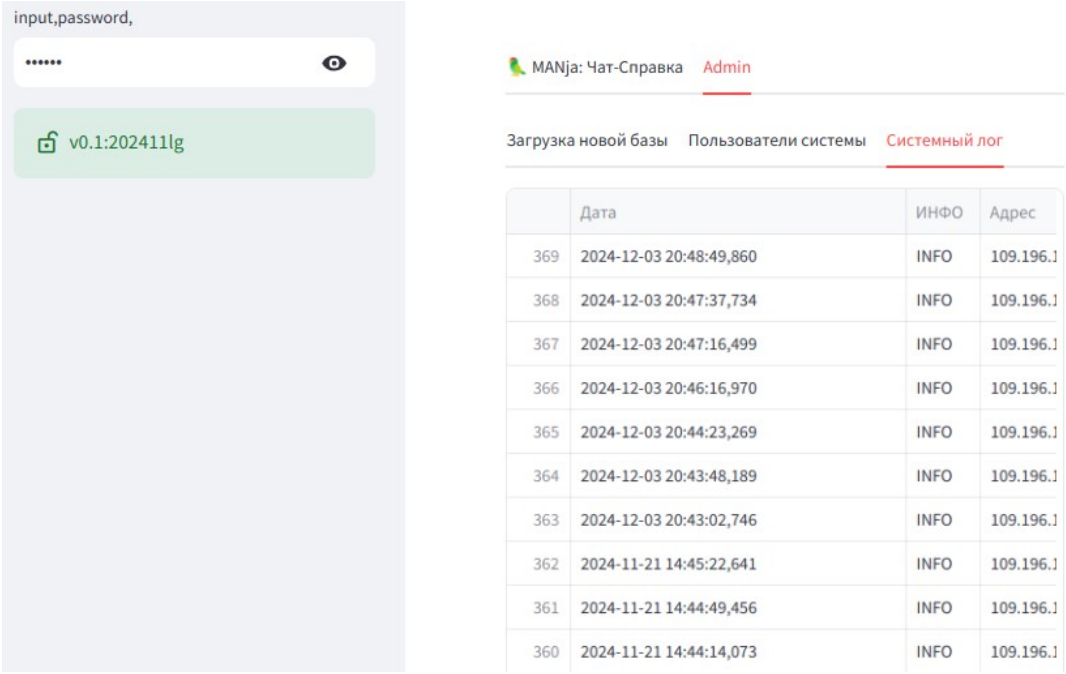

Рис. 11. Пользовательский интерфейс: панель администратора: системный лог

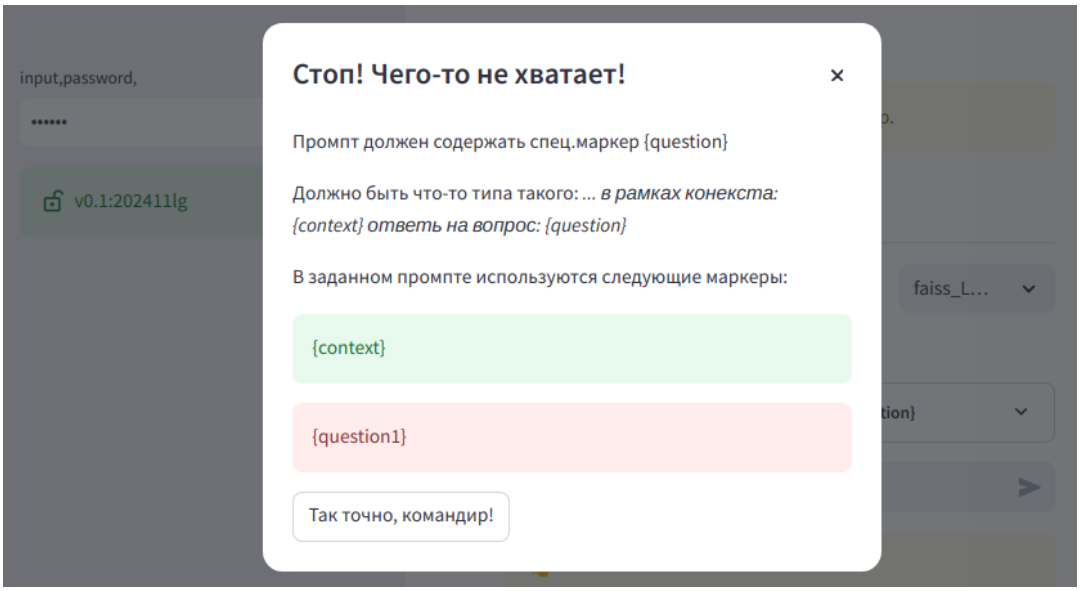

Рис. 12. Пользовательский интерфейс: пример сообщения об ошибке

Необходимо отметить, что большинство современных профессиональных веб-приложений строятся на основе архитектурного стиля REST API [27]. В них, в зависимости от концептуальных и технических требований, для реализации пользовательской части используют фреймворки преимущественно на основе языка JavaScript такие, как Angular, React.js и Vue.js [28]. В данной работе в первую очередь сделана ставка на модульность, возможность гибкой трансформации и адаптивной интеграции. Поэтому на начальных этапах разработки большее внимание уделено базовым функциям бизнес-логики, а пользовательский интерфейс реализован в виде прототипа на основе фреймворка Streamlit. Этот фреймворк, в отличие

от указанных выше и, по сути, признанных в веб-разработке стандартом де-факто, построен на языке Python и имеет открытый исходный код. Streamlit позволяет создавать быстрые прототипы динамических приложений, в том числе для работы с большими данными. На текущем этапе разработки применение данного фреймворка более чем оправдано, т. к. он широко используется исследователями, специалистами по обработке данных и инженерами в области искусственного интеллекта и машинного обучения. Это связано с тем, что самым популярным инструментом решения подобных задач является язык Python. Однако дальнейшее развитие данного проекта предусматривает плановый переход от прототипной реализации фронтенда к более профессиональному варианту на основе фреймворка Vue.js, успешный опыт применения которого был представлен авторами в работе [23].

Другим направлением является использование данной системы в качестве экспериментального стенда для проведения разнопланового тестирования и изучения эффективности применения больших языковых моделей на практике. Для удобства работы в этом режиме в системе дополнительно реализован альтернативный пользовательский интерфейс. Этот вариант интерфейса предоставляет пользователю возможность работать с системой как настольным приложением.

Пользовательский интерфейс настольного приложения реализован на основе кроссплатформенной библиотеки Tkinter. Этот вид интерфейса, по сравнению с веб-вариантом, использует менее выразительные средства (см. рисунки в разделе «Эксперименты»: рис. 13–15). Однако его преимуществом является возможность оперативного и непосредственного взаимодействия пользователя-оператора с программным кодом компонентов системы. Такой подход дает возможность максимально задействовать потенциал хорошо зарекомендовавшего себя в научных исследованиях языка программирования Python, находить новые варианты использования системы, оперативно тестировать экспериментальные решения. Полученные здесь результаты в дальнейшем используются для оптимизации, улучшения и расширения возможностей веб-версии системы.

Эксперименты

Для подготовки вариантов проблемно-ориентированного контекста, с которым должна работать система, реализован многоступенчатый процесс, состоящий из следующих этапов.

Загрузка и считывание документа. Для загрузки содержимого файла используется модуль *UnstructuredWordDocumentLoader* [29] из библиотеки *langchain_community.document_loaders*. Данный модуль имеет несколько режимов настройки загрузки документов.

Разбиение документа. Метод *langchain load_and_split* разделяет документ на части. Метод применяется для упрощения обработки, разбивая его на более управляемые блоки. В данном случае используется разделитель *RecursiveCharacterTextSplitter* [30] для разбиения документов. Разбиение документа дает возможность более простого анализа больших документов и особенно полезен для выделения содержимого таблиц и других структурированных данных.

Создание векторного представления. На этом этапе происходит работа методов класса YandexGPTEmbeddings, который отвечает за эмбеддинг, т. е. преобразование текста в векторные представления.

Создание векторного хранилища. При использовании библиотеки FAISS происходит создание векторного хранилища. На этом этапе обрабатываются фрагментированные документы и их эмбеддингов, а также создается индекс для облегчения дальнейшего поиска по тексту.

Варьирование загружаемого контекста позволяет исследовать особенности генерации системой ответов в зависимости как от структуры контекста, так и формулировки запросов.

Анализ качества генерации ответов на пользовательские запросы

Для исследования работы чат-бота на основе модели YandexGPT 3 Lite был проведен эксперимент, в ходе которого пользователи производили запросы и ставили оценки «отлично», «средне», «плохо» полученным ответам. В качестве контекста было использовано руководство пользователя многофункциональной программной системы инженерного моделирования.

В рамках эксперимента был задан 81 запрос. Из всех запросов 31 % (25 запросов) получил оценку «отлично», 44 % (36 запросов) — «средне» и 25 % (20 запросов) — «плохо». Таким образом, в ходе исследования было получено 75 % положительных оценок, что свидетельствует о приемлемом уровне ответов в большинстве запросов к чат-боту. Разберем результаты более детально.

Оценка «отлично» и оценка «средне» для запросов к чат-боту наиболее часто ставилась для запросов с формулировкой вопроса с использованием конструкций: «Где…», «Что такое…», «Какие…», «Чем…».

Также было выявлено, что чат-бот хорошо работает с вопросами на общие темы, для ответа на которые достаточно «знаний» используемой LLM.

Однако в случае вопросов, которые требуют более детального анализа, например, «Что умеет эта программная система?» и «Что нужно для запуска расчета качества?», запросы чаще всего получали оценку «средне». Слабой стороной чат-бота оказались запросы, касающиеся инструкций, т. е. вопросы типа «Как добавить/удалить/изменить…», «Как работает/функционирует…», и «Как правильно…». Такие запросы чаще всего получали пользовательскую оценку «плохо». Разработчики чат-бота предполагают, что данный результат во многом обусловлен тем, что необходимая информация представлена в руководстве в виде изображений.

При оценке работы чат-бота анализировались комментарии пользователей относительно качества ответов на запросы. Среди замечаний к ответам на разные вопросы отмечался как избыток информации, не относящейся к делу, так и недостаточная информативность ответов. Также была выявлена проблема, обусловленная тем, что в различных разделах справки одинаковые по написанию термины могут нести разную смысловую нагрузку.

Проведенный анализ показал, что для повышения качества ответов чат-бота необходимо обеспечить его фокусировку на контексте запроса для более релевантных ответов, а также добавление подробностей в ответы на сложные вопросы для полного понимания пользователем информации по интересуемой процедуре или функции.

Эксперименты с табличными данными

При работе с нормативными и справочными документами для генерации ответа в качестве контекста нередко требуется использовать данные, размещенные во включенных в документы таблицах. Корректная обработка данных, представленных в таблицах, дает возможность более точно выбирать информацию, необходимую для генерации ответов, касающихся как структуры, так и «внутреннего содержания» ячеек таблиц.

Рассмотрим результаты работы чат-бота с таблицей на примере, представляющем фрагмент учебного плана (табл.).

На рис. 13 представлен результат тестового «диалога» пользователя с чат-ботом.

Таблица

Таблица для тестирования чат-бота

Раздел дисциплины	Семестр	Используемые активные/интерактивные образовательные технологии	Количество часов
Теория технологии и комплексной механизации	10	Дискуссионные методы. Проблемное обучение	2 л 4 пр
Технология и комплексная механизация при сплошных системах разработки		Дискуссионные методы. Проблемное обучение	6 пр
Технология и комплексная механизация при углубочных системах разработки		Проблемное обучение	4 л
Перспективное и текущее планирование горных работ. Качество продукции		Дискуссионные методы. Проблемное обучение	2 пр
Итого:			6 л 12 пр

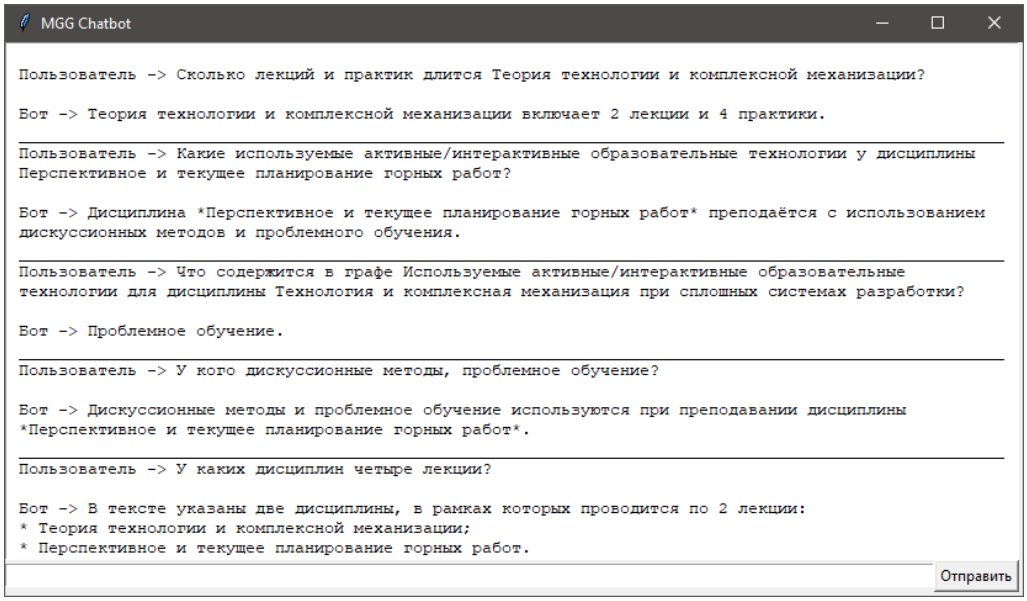

Рис. 13. Примеры запросов по табличным данным

Из представленного на рис. 13 примера видно, что в целом ответы чат-бота с использованием данных из таблицы являются корректными. Тем не менее имеются и неправильные ответы — в случае вопроса о дисциплинах с четырьмя лекциями бот указал на дисциплины, имеющие по две лекции. Выяснение причин появления некорректных ответов требует дополнительного исследования, в том числе и с точки зрения доработки методов интерпретации вопросов.

Эксперименты с анализом частей речи

Одним из направлений использования разрабатываемой системы планируется поддержка построения онтологий предметной области. Онтология, по сути, является концептуальной схемой, представляющей множество сущностей (понятий, объектов) предметной области и связей (отношений) между ними. В текстах на естественном языке сущности, как правило, представляются существительными, а вот для «описания» связи между ними могут использоваться различные части речи. Проведена серия экспериментов, связанных с оценкой возможностей системы идентифицировать и извлекать из текстов определенные части речи, в первую очередь существительных. Ниже представлены некоторые результаты этих экспериментов.

Рис. 14 демонстрирует, что чат-бот успешно идентифицирует различные части речи. Помимо существительных, прилагательных и глаголов, он также определяет более сложные грамматические категории такие, как причастия и деепричастия.

При ответе на запрос по идентификации частей речи (рис. 15) чат-бот приводит список слов, это означает, что система умеет выделять ключевые элементы текста. Тем не менее бот не упомянул такие слова, как «OPMPS» и «Pit», «Mine», «Scheduling», это указывает на наличие проблем в идентификации существительных на латинице и, возможно, аббревиатур.

Пример, представленный на рис. 16, показывает, что чат-бот может провести детальный разбор и выдать характеристику предложения, включая его тип, структуру и синтаксические компоненты. Он правильно определяет подлежащее, сказуемое и выделяет определения и дополнения.

По результатам проведенных экспериментов можно сделать вывод, что чат-бот выдает в большинстве случаев корректные структурированные ответы, демонстрирует способность работать с технической и специализированной лексикой, а также проводить комплексный анализ текста на естественном языке. Анализ текстов выполняется чат-ботом с достаточной степенью точности и детализации.

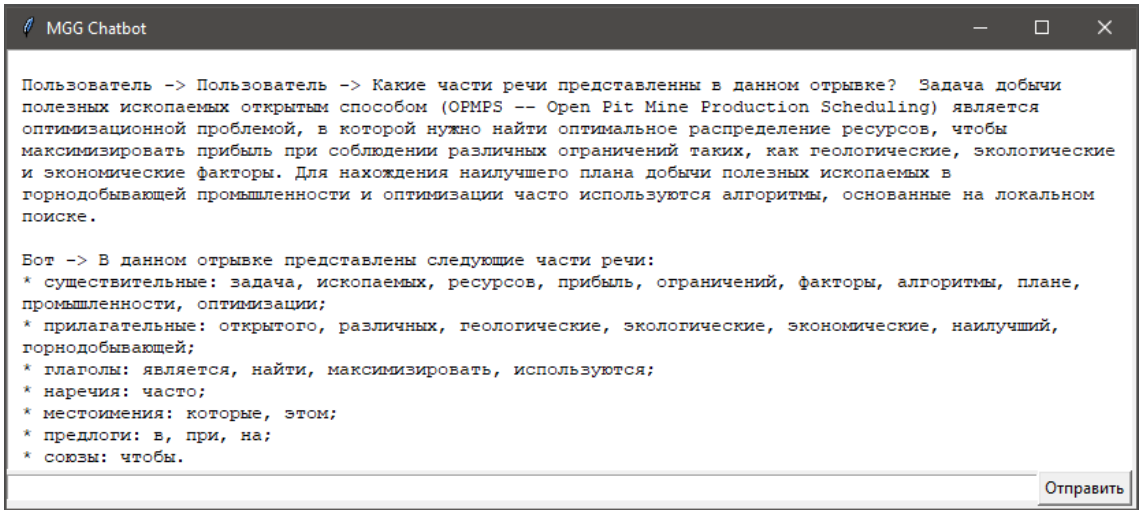

Рис. 14. Пример запроса по определению частей речи

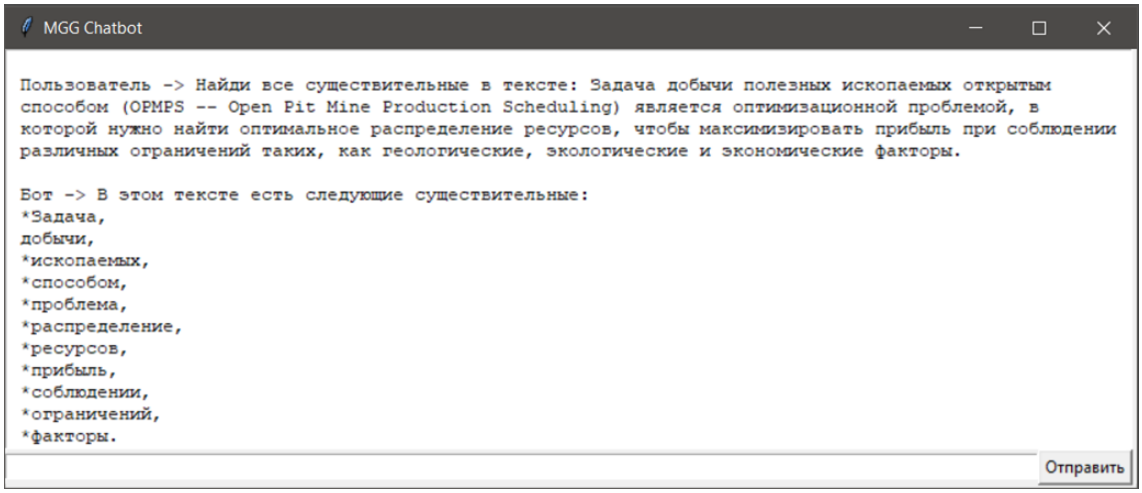

Рис. 15. Пример запроса по идентификации частей речи

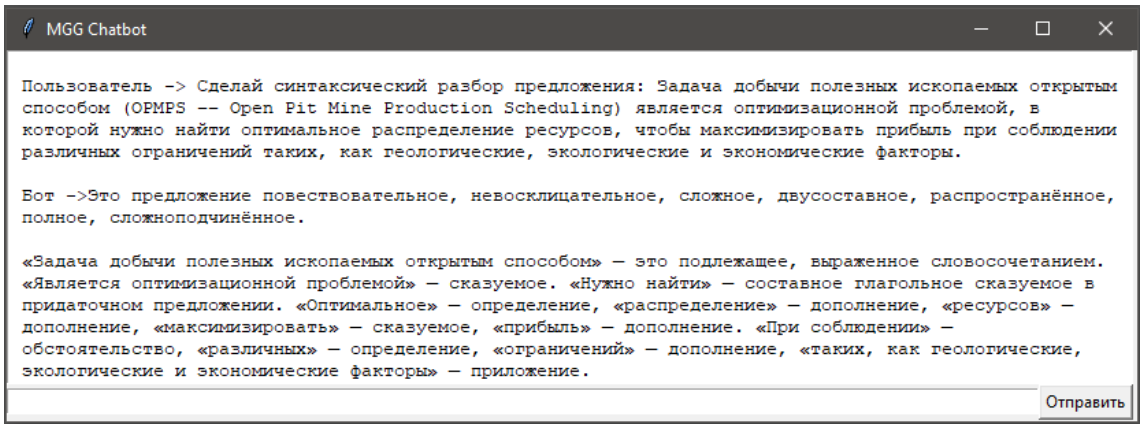

Рис. 16. Пример запроса по синтаксическому разбору предложения

Влияние на релевантность ответа на запрос дробления контекста на разделы

При использовании чат-бота возможны разные подходы к формированию контекста запроса, который напрямую влияет на ответ. Варьирование подготовки контекста позволяет исследовать влияние таких аспектов, как многозначность терминов, контекстуальная зависимость.

На рис. 17 представлены ответы на один и тот же вопрос с использованием разных разделов документации.

Различия в ответах на заданный вопрос обусловлены тем, что в разных разделах документа слово «Модель» интерпретируется различным образом.

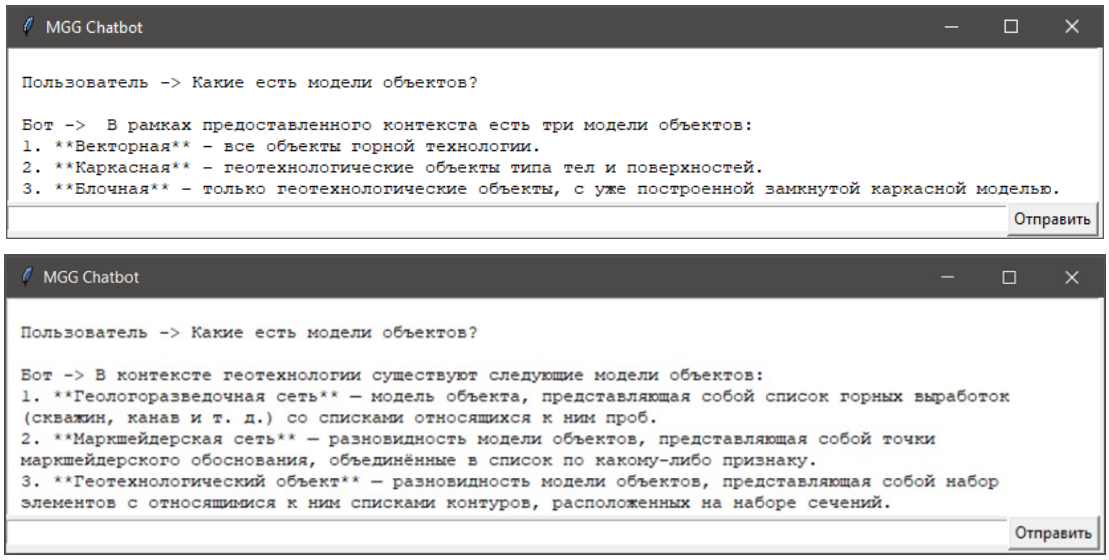

Рис. 17. Варианты ответов при использовании в качестве контекста разных разделов одного документа

Заключение

В публикациях, посвященных использованию больших языковых моделей, отмечается, что, наряду с достижениями LLM в области работы с текстами на естественном языке, имеет место и ряд проблем, не позволяющих полностью «доверять» результатам их работы. Более корректные результаты получаются в случае ограничения (конкретизации) контекста, с которым должна работать LLM в рамках определенной задачи. Обеспечить такую конкретизацию позволяет использование технологии RAG.

В данной статье представлены основные этапы проектирования и реализации интеллектуализированной системы работы с массивами документов. Созданный инструмент ориентирован на интеллектуальную поддержку решения практических задач как в автономном режиме, так и в режиме интеграции в сторонние предметно-ориентированные информационные системы.

Проведенные эксперименты по работе с системой носят предварительный характер. Но и они показали, что разрабатываемая система может не только выполнять функции «подсказчика» для пользователя документации, общаясь с ним на естественном языке, но и существенно упростить работу системных аналитиков и разработчиков онтологий. В рамках этих экспериментов были выявлены подходящие конфигурационные параметры, позволяющие использовать большие языковые модели для выделения в тексте целевых концептов, используемых для заселения онтологии.

Список источников

1. Постановление Правительства РФ от 14 августа 2020 г. № 1225 «Об утверждении Правил разработки критериев отнесения объектов всех форм собственности к критически важным объектам». [Электронный ресурс]. URL: http://government.ru/docs/all/129416/ (дата обращения: 01.11.2024).

2. Олейник А. Г., Ломов П. А. Разработка онтологии интегрированного пространства знаний // Онтология проектирования. 2016. Т. 6, № 4(22). С. 465–474.
3. Ломов П. А., Олейник А. Г. Разработка технологии проверки и согласования нормативно-правовой базы на основе онтологий // Труды Института системного анализа Российской академии наук. 2013. Т. 63, № 2. С. 62–69.
4. Bystrov, V. V., Khaliullina, D. N., Malygina, S. N. (2024). Conceptual Modeling of the Resilience of Regional Socio-Economic Systems “Business-Society-Government”. In: Silhavy, R., Silhavy, P. (eds) Software Engineering Methods in Systems and Network Systems. CoMeSySo 2023. Lecture Notes in Networks and Systems, vol 934., pp. 179–191.
5. Датьев И. О., Федоров А. М. Аддитивная регуляризация при тематическом моделировании текстов сообществ онлайновых социальных сетей // Онтология проектирования. 2022. Т. 12, № 2(44). С. 186–199.
6. Пимешков В. К., Никонорова М. Л., Шишаев М. Г. Комбинированный метод извлечения терминов для задачи мониторинга тематических обсуждений в социальных медиа // Информатика и автоматизация. 2024. № 4(23). С. 1110–1138.
7. Lomov P., Malozemova M., Shishaev M. Training and application of neural-network language model for ontology population // Software engineering perspectives in intelligent systems / ed. Silhavy R., Silhavy P., Prokopova Z. Cham: Springer International Publishing, 2020. P. 919–926.
8. Lomov P., Malozemova M., Shishaev M. Extracting relations from NER-tagged sentences for ontology learning // Artificial Intelligence Trends in Systems: Proceedings of 11th Computer Science On-line Conference 2022. Vol. 2. Cham: Springer, 2022. P. 337–344.
9. Wang L. et al. Genre: Generative Multi-Turn Question Answering with Contrastive Learning for Entity–Relation Extraction // Complex & Intelligent Systems. 2024. Vol. 10. P. 3429–3443.
10. Gao Y. et al. Retrieval-Augmented Generation for Large Language Models: A Survey // arXiv:2312.10997 [cs]. 2023.
11. Yandex Foundation Models [Электронный ресурс]. URL: https://yandex.cloud/ru/docs/foundation-models/ (дата обращения: 20.11.2024).
12. GigaChat API [Электронный ресурс]. URL: https://developers.sber.ru/docs/ru/gigachat/api/overview (дата обращения: 20.10.2024).
13. OpenAI developer platform [Электронный ресурс]. URL: https://platform.openai.com/docs/overview (дата обращения: 07.03.2024).
14. Manning C. D. Human Language Understanding & Reasoning // Daedalus. 2022. Vol. 151, № 2. P. 127–138.
15. Vaswani A. et al. Attention Is All You Need // arXiv:1706.03762 [cs]. 2017.
16. Zhang Y. et al. Siren’s Song in the AI Ocean: A Survey on Hallucination in Large Language Models // arXiv:2309.01219 [cs]. 2023.
17. Экономика больших языковых моделей: сколько стоит обучить и сколько — эксплуатировать. [Электронный ресурс]. URL: https://telegra.ph/EHkonomika-bolshih-yazykovyh-modelej-skolko-stoit-obuchit-i-skolko--ehkspluatirovat-04-07 (дата обращения: 01.11.2024).
18. 30 миллиардов параметров: реально ли обучить русский GPT-3 в «домашних» условиях? [Электронный ресурс]. URL: https://habr.com/ru/articles/565564/ (дата обращения: 01.11.2024).
19. Ma X. et al. Query Rewriting for Retrieval-Augmented Large Language Models // arXiv:2305.14283 [cs]. 2023
20. Advanced RAG Techniques: an Illustrated Overview [Электронный ресурс]. URL: https://pub.towardsai.net/advanced-rag-techniques-an-illustrated-overview-04d193d8fec6 (дата обращения: 19.10.2024).
21. Langchain introduction [Электронный ресурс]. URL: https://python.langchain.com/docs/introduction/
22. Федоров А. М., Датьев И. О., Вишняков И. Г. Проектирование информационной системы комплексного тематического анализа больших данных социальных медиа // Онтология проектирования. 2024. Т. 14, № 1(51). С. 55–70. doi:10.18287/2223-9537-2024-14-1-55-70.
23. Датьев И. О., Федоров А. М., Ревякин А. А. Фокусированный сбор и обработка открытых данных социальных медиа. Онтология проектирования. 2024. Т. 14, № 4(54). С. 569–581.

24. Информационная карта научно-исследовательской работы (НИР) «Методология создания информационно-аналитических систем поддержки управления региональным развитием, основанных на формирующем искусственном интеллекте и больших данных» (регистрационный номер: 122022800551-0) [Электронный ресурс]. URL: https://gisnauka.ru/nioktr/detail/SB2430UAGEM4J2MSBKAK1Q3C (дата обращения: 01.11.2024).
25. Информационная карта научно-исследовательской работы (НИР) «Разработка теоретических и организационно-технических основ информационной поддержки управления жизнеспособностью региональных критических инфраструктур Арктической зоны Российской Федерации» (регистрационный номер: 122022800547-3) [Электронный ресурс]. URL: https://gisnauka.ru/nioktr/detail/IFE0D83R25SX5MYIYNDZQJDL (дата обращения: 01.11.2024).
26. Streamlit: наиболее быстрый способ создания приложений для обработки данных и обмена ими [Электронный ресурс]. URL: https://streamlit.io/ (дата обращения: 31.10.2024).
27. Prayogi A. A., Niswar M., Indrabayu and Rijal M. Design and Implementation of REST API for Academic Information System. A A Prayogi et al 2020 IOP Conf. Ser.: Mater. Sci. Eng. 875 012047 doi:10.1088/1757-899X/875/1/012047
28. Vue.js: прогрессивный JavaScript-фреймворк [Электронный ресурс]. URL: https://ru.vuejs.org/ (дата обращения: 01.11.2024).
29. UnstructuredWordDocumentLoader [Электронный ресурс]. URL: https://python.langchain.com/api_reference/community/document_loaders/langchain_community.document_loaders.word_document. Unstructured WordDocumentLoader.html (дата обращения: 31.10.2024).
30. RecursiveCharacterTextSplitter [Электронный ресурс]. URL: https://python.langchain.com/api_reference/community/document_loaders/langchain_community.document_loaders.word_document.UnstructuredWordDocumentLoader.html (дата обращения: 01.11.2024).

References

1. Postanovlenie Pravitel'stva RF ot 14 avgusta 2020 g. № 1225 "Ob utverzhdenii Pravil razrabotki kriteriev otneseniya ob"ektov vsekh form sobstvennosti k kriticheski vazhnym ob"ektam". [Resolution of the Government of the Russian Federation of August 14, 2020 no. 1225 “On Approval of the Rules for the Development of Criteria for Attributing Facilities of All Forms of Ownership to Critically Important Facilities”.]. Available at: http://government.ru/docs/all/129416/ (accessed 01.11.2024) (In Russ.).
2. Oleynik A. G., Lomov P. A. Razrabotka ontologii integrirovannogo prostranstva znanij [Development of the ontology of integrated knowledge space]. Ontologiya proektirovaniya [Ontology of designing], 2016, Vol. 6, no. 4, pp. 465–474. (In Russ.).
3. Lomov P. A., Oleynik A. G. Razrabotka tekhnologii proverki i soglasovaniya normativno-pravovoy bazy na osnove ontologiy [Development of the technology for verifying and coordinating the regulatory framework based on ontologies], Trudy Instituta sistemnogo analiza Rossiyskoy akademii nauk [Proceedings of the Institute of System Analysis of the Russian Academy of Sciences], 2013, Vol. 63, no. 2, pp. 62–69. (In Russ.).
4. Bystrov V. V., Khaliullina D. N., Malygina S. N. Conceptual Modeling of the Resilience of Regional Socio-Economic Systems “Business-Society-Government”. In: Silhavy, R., Silhavy, P. (eds) Software Engineering Methods in Systems and Network Systems. CoMeSySo. Lecture Notes in Networks and Systems, 2023, Vol. 934, pp. 179–191.
5. Datyev I. O., Fedorov A. M. Additivnaya regulyarizaciya pri tematicheskom modelirovanii tekstov soobshchestv onlajnovyh social'nyh setej [Additive regularization for topic modeling of social media communities]. Ontologiya proektirovaniya [Ontology of designing], 2022, Vol. 12 no. 2, pp. 186–199. (In Russ.)
6. Pimeshkov V. K, Nikonorova M. L, Shishaev M. G. A Kombinirovannyj metod izvlecheniya terminov dlya zadachi monitoringa tematicheskih obsuzhdenij v social'nyh media [Combined Term Extraction Method for the Problem of Monitoring Thematic Discussions in Social Media]. Informatika i avtomatizaciya [Informatics and Automation], 2024, Vol. 23, no. 4, pp. 1110-1138. (In Russ.).

7. Lomov P., Malozemova M., Shishaev M. Training and application of neural-network language model for ontology population. Software engineering perspectives in intelligent systems. In: Silhavy R., Silhavy P., Prokopova Z. (eds). Software Engineering Perspectives in Intelligent Systems. CoMeSySo 2020 Advances in Intelligent Systems and Computing, 2020, Vol. 1295, Springer, Cham, pp. 919–926.
8. Lomov P., Malozemova M., Shishaev M. Extracting relations from NER-tagged sentences for ontology learning. Proceedings of 11th Computer Science Online Conference 2022, 2022, Vol. 2, Springer, Cham, pp. 337–344.
9. Wang L. et al. Genre: Generative Multi-Turn Question Answering with Contrastive Learning for Entity–Relation Extraction. Complex & Intelligent Systems, 2024, Vol. 10, pp. 3429–3443.
10. Gao Y. et al. Retrieval-Augmented Generation for Large Language Models: A Survey. arXiv:2312.10997 [cs], 2023.
11. Yandex Foundation Models. Available at: https://yandex.cloud/ru/docs/foundation-models/ (accessed 20.11.2024).
12. GigaChat API. Available at: https://developers.sber.ru/docs/ru/gigachat/api/overview (accessed 20.10.2024).
13. OpenAI developer platform. Available at: https://platform.openai.com/docs/overview (accessed 07.10.2024).
14. Manning C. D. Human Language Understanding & Reasoning. Daedalus, 2022, vol. 151, no. 2, pp. 127–138.
15. Vaswani A. et al. Attention Is All You Need. arXiv:1706.03762 [cs], 2017.
16. Zhang Y. et al. Siren's Song in the AI Ocean: A Survey on Hallucination in Large Language Models. arXiv:2309.01219 [cs], 2023.
17. Ekonomika bol'shih yazykovyh modelej: skol'ko stoit obuchit', i skol'ko — ekspluatirovat' [The economics of large language models: how much it costs to teach, and how much to exploit]. (In Russ.). Available at: https://telegra.ph/EHkonomika-bolshih-yazykovyh-modelej-skolko-stoit-obuchit-i-skolko--ehkspluatirovat-04-07 (accessed 01.11.2024)
18. 30 milliardov parametrov: real'no li obuchit' russkij GPT-3 v «domashnih» usloviyah? [30 billion parameters: is it really possible to teach Russian GPT-3 at home?] (In Russ.). Available at: https://habr.com/ru/articles/565564/ (accessed 01.11.2024)
19. Ma X. et al. Query Rewriting for Retrieval-Augmented Large Language Models. arXiv: 2305.14283 [cs], 2023.
20. Advanced RAG Techniques: an Illustrated Overview. Available at: https://pub.towardsai.net/advanced-rag-techniques-an-illustrated-overview-04d193d8fec6 (accessed 19.10.2024).
21. Langchain introduction. Available at: https://python.langchain.com/docs/introduction/ (accessed 19.10.2024).
22. Fedorov A. M., Datyev I. O., Vishnyakov I. G. Proektirovanie informacionnoj sistemy kompleksnogo tematicheskogo analiza bol'shih dannyh social'nyh media [Designing an information system for integrated topic analysis of social media big data]. Ontologiya proektirovaniya [Ontology of designing], 2024, Vol. 14, no. 1, pp. 55–70. (In Russ.).
23. Datyev I. O., Fedorov A. M., Reviakin A. A. Fokusirovannyj sbor i obrabotka otkrytyh dannyh social'nyh media [Focused collection and processing of open social media data]. Ontologiya proektirovaniya [Ontology of designing], 2024, Vol. 14, no. 4, pp. 569–581. (In Russ.).
24. Informacionnaya karta nauchno-issledovatel'skoj raboty (NIR) «Metodologiya sozdaniya informacionno-analiticheskih sistem podderzhki upravleniya regional'nym razvitiem, osnovannyh na formiruyushchem iskusstvennom intellekte i bol'shih dannyh» (Registracionnyj nomer: 122022800551-0) [Information card of scientific research "Methodology for creating information and analytical systems to support regional development management based on formative artificial intelligence and big data" (Registration number: 122022800551-0)]. (In Russ.). Available at: https://gisnauka.ru/nioktr/detail/SB2430UAGEM4J2MSBKAK1Q3C (accessed 01.11.2024).
25. Informacionnaya karta nauchno-issledovatel'skoj raboty (NIR) «Razrabotka teoreticheskih i organizacionno-tekhnicheskih osnov informacionnoj podderzhki upravleniya zhiznesposobnost'yu regional'nyh kriticheskih infrastruktur Arkticheskoj zony Rossijskoj Federacii» (Registracionnyj nomer: 122022800547-3) [Information card of scientific research "Development of theoretical and organizational and technical foundations of information support for the management of the viability of regional critical infrastructures of the Arctic zone of the Russian Federation" (Registration number: 122022800547-3)]. (In Russ.). Available at: https://gisnauka.ru/nioktr/detail/ IFE0D83R25SX5MYIYNDZQJDL (accessed 01.11.2024).

26. Streamlit: naibolee bystryj sposob sozdaniya prilozhenij dlya obrabotki dannyh i obmena imi [Streamlit: a faster way to build and share data apps]. (In Russ.). Available at: https://streamlit.io/ (accessed 31.10.2024).
27. Prayogi A. A., Niswar M., Indrabayu, and Rijal M., Design and Implementation of REST API for Academic Information System, IOP Conf Ser Mater Sci Eng, 2020, Vol. 875, no. 1, P. 012047.
28. Vue.js: The Progressive JavaScript Framework. Available at: https://vuejs.org/ (accessed 01.11.2024).
29. UnstructuredWordDocumentLoader. Available at: https://python.langchain.com/apireference/community/documentloaders/langchaincommunity.documentloaders.worddocument.UnstructuredWordDocumentLoader.html (accessed 31.10.2024).
30. RecursiveCharacterTextSplitter. Available at: https://python.langchain.com/apireference/community/documentloaders/langchaincommunity.documentloaders.worddocument.UnstructuredWordDocumentLoader.html (accessed 01.11.2024).

Информация об авторах

А. Г. Олейник — доктор технических наук, директор, главный научный сотрудник;
А. М. Федоров — кандидат технических наук, ведущий научный сотрудник;
И. О. Датьев — кандидат технических наук, старший научный сотрудник;
А. А. Зуенко — кандидат технических наук, ведущий научный сотрудник;
А. В. Шестаков — аспирант, стажер-исследователь;
И. Г. Вишняков — аспирант, системный администратор.

Information about the authors

A. G. Oleynik — Doctor of Science (Tech.), Director, Chief Scientific Officer;
A. M. Fedorov — Candidate of Science (Tech.), Leading Scientific Officer;
I. O. Datyev — Candidate of Science (Tech.), Senior Scientific Officer;
A. A. Zuenko — Candidate of Science (Tech.), Leading Scientific Officer;
A. V. Shestakov — Graduate Student, Intern Researcher;
I. G. Vishnyakov — Graduate Student, System Administrator.

Статья поступила в редакцию 16.10.2024; одобрена после рецензирования 01.11.2024; принята к публикации 07.11.2024.
The article was submitted 16.10.2024; approved after reviewing 01.11.2024; accepted for publication 07.11.2024