**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 производственной практики, преддипломной практики**

(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 18.03.2025г. по 07.05.2025 г.

**Руководитель практики от НГУ** Галактионова Юлия Юрьевна, специалист УМОВОИИР /

(Ф.И.О. полностью, должность) (подпись)

**Руководитель ВКР**   Оглезнев Никита Сергеевич, сотрудник КафИСТИИР, 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 2025 г.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc196477919)

[ЗАКЛЮЧЕНИЕ 5](#_Toc196477920)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 6](#_Toc196477921)

ВВЕДЕНИЕ

В рамках производственной практики была поставлена задача: реализовать полноценную RAG-систему (Retrieval-Augmented Generation), предоставляющую пользователю возможность взаимодействовать с ИИ через веб-интерфейс. Основное внимание было уделено созданию клиент-серверной архитектуры, обеспечивающей стриминговую генерацию ответов.

1. АРХИТЕКТУРА ПРИЛОЖЕНИЯ

Приложение включает следующие основные компоненты:

1. Серверная часть
2. Клиентская часть
3. База данных
4. RAG-система
   1. Серверная часть

Серверная часть реализована на языке Python с использованием асинхронного фреймворка FastAPI. Для взаимодействия с базой данных используется библиотека SQLAlchemy[ссылка на sa]. Основные задачи серверной части:

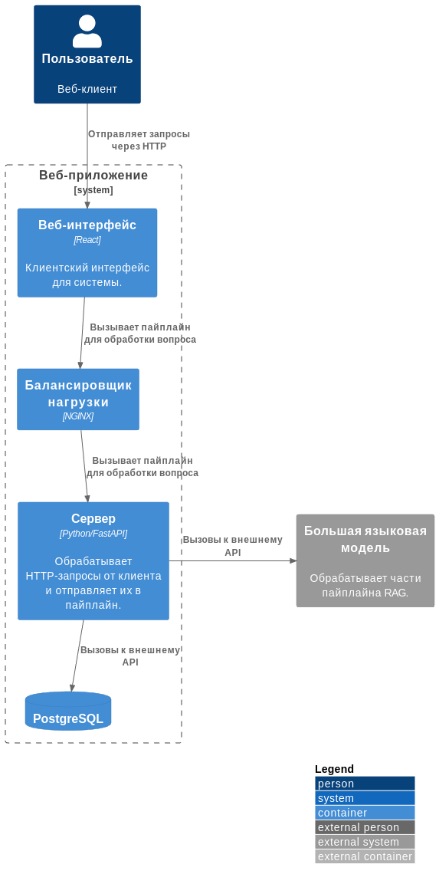
1. Приём и обработка HTTP- и SSE-запросов от клиента.
2. Взаимодействие с RAG-системой для генерации ответов.
3. Аутентификация и авторизация пользователей.
4. Обеспечение доступа к хранимым файлам и метаданным.
5. Поддержка многопользовательского режима.

Для реализации стриминговых ответов используется однонаправленный протокол Server-Sent Events (SSE). Благодаря асинхронной архитектуре, сервер способен обрабатывать несколько одновременных запросов без блокировок, что особенно важно при работе с языковыми моделями и хранилищами, где ответ может формироваться в течении длительного времени.

Чтобы стандартизировать подход к визуализации архитектуры была выбрана нотация моделирования C4. Она позволяет спроектировать систему на разных уровнях абстракции. Название “C4” расшифровывается как:

1. Context – Контекст – показывает, с какими внешними системами взаимодействует документируемая система, а также какие пользователи будут ее использовать.
2. Containers – Контейнеры – показывает архитектуру приложения описывается без глубокого погружения в техническую часть. На этом этапе описываются используемые технологии, основные логические элементы и их взаимодействие между собой, с внешними системами, и с пользователем.
3. Conponents – Компоненты – позволяет глубже погрузиться в архитектуру отдельных контейнеров.
4. Code – Код – самый низкий уровень абстракции в рамках нотации. Позволяет спроектировать взаимодействие отдельных классов.

Для проектирования своей системы я решил использовать 2 и 3 уровни абстракции, так как они помогают предоставляют удобный уровень абстракции для проектирования компонентов системы без избыточных представлений о взаимодействии классов в кодовой базе. На диаграмме второго уровня я реализовал верхнеуровневое представление о клиент-серверном приложении (см. на Рисунке 1).



* 1. Клиентская часть

Клиентская часть реализована с использованием библиотеки React и представляет собой одностраничное приложение (SPA), обеспечивающее удобный и интуитивно понятный пользовательский интерфейс. Основной функционал включает:

1. Интерфейс чата с возможностью ввода и отображения истории сообщений.
2. Стриминг частичных ответов от сервера.
3. Система авторизации и регистрации пользователей.
4. Меню для загрузки файлов.
5. Инструмент для просмотра PDF документов.
   1. Хранение данных

Для хранения данных используется PostgreSQL[ссылка на PG]. Там хранятся история сообщений, информация о пользователях, а также связанные с файлами метаданные. Загруженные PDF-файлы хранятся в локальном хранилище, доступ к ним осуществляется в асинхронном режиме.

* 1. RAG-система

Ключевой частью архитектуры является модуль Retrieval Augmented Generation. Взаимодействие с этой системой происходит по следующей схеме:

1. Поиск релевантных фрагментов в базе данных с помощью векторного поиска на основе косинусной близости векторов.
2. Формирование промпта в модель на основе найденных фрагментов, запроса пользователя и генерация финального ответа.
3. ТЕСТИРОВАНИЕ

Для проверки корректности и надёжности реализованного функционала были разработаны модульные и интеграционные тесты с использованием библиотеки pytest[ссылка на Pytest] и асинхронного клиента из библиотеки httpx[ссылка на httpx]. Тестами покрыты основные компоненты:

1. Авторизация и регистрация пользователей.
2. Получение и хранение истории сообщений
3. Работа с SSE, и получение ответов от системы.
4. Загрузка документов и парсинг PDF-файлов
5. Интеграция с RAG-системой

ЗАКЛЮЧЕНИЕ

Разработано полнофункциональное клиент-серверное приложение, демонстрирующее принципы современной веб-разработки, стриминга с использованием LLM и взаимодействия с RAG-системами. Архитектура масштабируема и может быть расширена дополнительными модулями.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ML System Design Doc – Reliable ML [Электронный ресурс]. — 2025. — Режим доступа: <https://github.com/IrinaGoloshchapova/ml_system_design_doc_ru/tree/main> (дата обращения: 15.03.2025).
2. HuggingFace [Электронный ресурс]. — 2025. — Режим доступа: https://huggingface.co/deepvk/USER-bge-m3 (дата обращения: 15.03.2025).