**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](#_Toc197015497)

[1 АРХИТЕКТУРА ПРИЛОЖЕНИЯ 4](#_Toc197015498)

[1.1 Серверная часть 4](#_Toc197015499)

[1.2 Моделирование архитектуры: подход C4 4](#_Toc197015500)

[1.3 Клиентская часть 7](#_Toc197015501)

[1.4 Хранение данных 8](#_Toc197015502)

[1.5 Модуль Retrieval-Augmented-Generation 8](#_Toc197015503)

[2 ТЕСТИРОВАНИЕ 9](#_Toc197015504)

[ЗАКЛЮЧЕНИЕ 10](#_Toc197015505)

ВВЕДЕНИЕ

В рамках производственной практики передо мной была поставлена задача — разработать полнофункциональную RAG-систему, предоставляющую пользователю возможность взаимодействовать с большой языковой моделью через удобный веб-интерфейс. Ключевыми целями проекта стали: обеспечение потоковой генерации ответов на основе пользовательских запросов, поддержка многопользовательского режима, а также реализация полной клиент-серверной архитектуры с безопасной авторизацией и хранением данных.

Особое внимание было уделено созданию надёжной архитектуры, тестируемости компонентов, а также визуализации системы с помощью современной нотации моделирования C4, позволяющей рассматривать архитектуру на различных уровнях абстракции.

1. АРХИТЕКТУРА ПРИЛОЖЕНИЯ
   1. Серверная часть

Сервер реализован на языке Python с использованием асинхронного фреймворка FastAPI. Для работы с базой данных используется библиотека SQLAlchemy. Основные функции серверной части включают:

1. Обработку HTTP- и SSE-запросов от клиента.
2. Взаимодействие с RAG-системой для генерации ответов.
3. Аутентификация и авторизация пользователей (с использованием JWT-токенов).
4. Асинхронный доступ к хранимым файлам и метаданным.
5. Поддержка многопользовательского режима.

Для реализации потоковой генерации ответов используется однонаправленный протокол Server-Sent Events (SSE). Благодаря асинхронной архитектуре, сервер способен эффективно обрабатывать множество одновременных клиентов.

* 1. Моделирование архитектуры: подход C4

Для формализации архитектуры системы была выбрана C4-нотация. Она предлагает гибкий набор инструментов для проектирования программных систем, и включает в себя несколько уровней детализации:

1. Контекст (Context) – показывает внешние системы и пользователей.
2. Контейнеры (Containers) – архитектура приложения без глубокого погружения в техническую часть. Отображает основные логические блоки и используемые технологии.
3. Компоненты (Components) – раскрывает архитектуру отдельных контейнеров.
4. Code – Код – самый низкий уровень абстракции, чтобы показать классы и их связи.

Для проекта были использованы второй и третий уровни – “Контейнеры” и “Компоненты”, они позволили описать систему в достаточной детализации без избыточных подробностей о кодовой базе проекта.

На рисунке 1 представлена диаграмма контейнеров, показывающая общее взаимодействие клиента, сервера, базы данных и RAG-пайплайна.



Рисунок 1 – диаграмма C4 (уровень контейнеров).

Следующий уровень детализации – компоненты серверной части (Рисунок 2). На нем я выделил ключевые модули:

1. API-сервис – точка входа, обрабатывает запросы от клиента, авторизует пользователей, управляет загрузкой файлов, и инициирует взаимодействие с RAG-пайплайном.
2. Сервис индексации RAG – обрабатывает загруженные документы, извлекает текст, и разделяет его на фрагменты для последующего поиска.
3. Сервис генерации RAG – принимает пользовательский запрос, извлекает релевантные фрагменты, генерирует финальный ответ с помощью большой языковой модели.



Рисунок 2 – диаграмма C4 третьего уровня (Components)

Наконец, я визуализировал более детализированную схему взаимодействия внутри RAG-пайплайна, куда включил этапы индексации и генерации вместе. (Рисунок 3).



Рисунок 3 - диаграмма C4 (уровень компонентов, RAG-пайплайн)

* 1. Клиентская часть

Клиент (Рисунок 4) реализован в виде одностраничного веб-приложения (SPA) с использованием библиотеки React. Основной функционал включает:

1. Интерактивный чат с возможностью ввода и отображения истории сообщений.
2. Поддержка потоковых ответов от сервера.
3. Регистрация и авторизация пользователей.
4. Загрузка и просмотр PDF-документов.
5. Навигация по загруженным файлам.



Рисунок 4 – Пользовательский интерфейс веб-приложения

* 1. Хранение данных

В качестве основной СУБД используется PostgreSQL. В базе хранятся:

1. Информация о пользователях
2. История запросов и ответов
3. Метаданные, связанные с загруженными файлами
4. Чанки и их векторные представления

Сами PDF-файлы сохраняются в локальном хранилище. Доступ к ним происходит асинхронно по запросу клиента.

* 1. Модуль Retrieval-Augmented-Generation

Модуль RAG – ключевая часть архитектуры моего приложения. Его работа включает два этапа: Индексация и Генерация (Рисунок 3). Было принято решение отказаться от классического метода с использованием поиска на основе BM25. Несмотря на его простоту и эффективность в ряде задач, он имеет ограничение, связанное с работой с мультиязычными данными. В моей системе как запросы, так и чанки в базе данных могут быть представлены как на русском, так и на английском языках. BM25 оперирует на уровне отдельных слов, а значит не способен учитывать семантику.

Вместо этого я реализовал семантический поиск с использованием векторных представлений как для запросов, так и для чанков из базы данных. Для вычисления схожести используется косинусная мера. Это позволяет системе находить релевантные фрагменты даже с учетом различия в формулировках и языках.

1. ТЕСТИРОВАНИЕ

Для проверки корректности и надёжности функционала были разработаны модульные и интеграционные тесты с использованием библиотеки Pytest и асинхронного клиента из библиотеки HTTPX. Тестами покрыты основные компоненты:

1. Регистрация и авторизация пользователей.
2. Хранение и получение истории сообщений.
3. Работа с SSE, и получение потоковых ответов от системы.
4. Загрузка и парсинг PDF-файлов.
5. Интеграция с RAG-системой и корректная обработка запросов.

Тестирование проводится автоматически, что позволяет оперативно выявлять ошибки при изменении кода и вносить улучшения без снижения стабильности системы.

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.