Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение
высшего образования
«Уральский федеральный университет
имени первого Президента России Б.Н. Ельцина»

Институт радиоэлектроники и информационных технологий – РТФ

Школа бакалавриата

ОТЧЕТ

По проекту
«Разработка чат-бота GPT для анализа и сравнения документации по объектам»

по дисциплине «Проектный практикум»

Заказчик: Чекалова Э.Р.

Куратор: Кузнецов Д.С.

Студенты команды Legion

Беликов Д.В.

Гук Е.М.

Денисова Е.И.

Зайкова С.А.

Кузнецов. В.П.

Екатеринбург, 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 3
1 Основная часть 5
1.1 Разбор требований заказчика и пользователей к продукту. Составление плана действий 5
1.1.1 Требования заказчика, составленное им техническое задание 5
1.1.2 Составленный план действий на семестр 6
1.2 Информация о работе каждого участника 6
1.3 Информация о планировании деятельности в ходе разработки и распределении задач между участниками команды разработчиков 6
1.3.1 Анализ аналогов и выявление лучшего для работы над проектом 7
1.3.2 Анализ целевой аудитории 16
1.4 Обзор архитектуры программного продукта, описание основных компонентов и связей между ними, обоснование выбора архитектурного решения 19
1.5 Описание методологии разработки, информация о процессе разработки, отчет о результатах тестирования на промежуточных этапах, разбор выявленных ошибок 21
1.5.1 Методология и процесс разработки. 21
1.5.2 Кодовые решения 22
1.5.3 Сложности при выполнении проекта 24
заключение 26
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 28

ВВЕДЕНИЕ

Цель:

Создание чат бота GPT, в который пользователь сможет выгружать документацию по объекту, техническое задание объекта. Бот будет сравнивать и анализировать документацию между собой и находить расхождения между предоставляемыми документами по определенным параметрам.

Задачи:

1) Реализовать возможность бота сравнивать Архитектурный раздел с готовым Техническим Заданием. Сравнение производится по всем материалам в стенах и полах на соответствие итоговому Техническому Заданию.

2) Задать возможность боту после проведенного анализа «складывать» полученный результат в файл, в котором будет отображена вся информация о том, что не совпало с итоговым Техническим Заданием.

3) Производить сравнение Конструктивного раздела с готовым Техническим Заданием, а именно – сравнение материала стен.

4) Выполнить сравнение Инженерного раздела с готовым Техническим заданием, «научить» бот сопоставлять между собой производителей оборудования отопления, вентиляции, кондиционирования.

Актуальность проекта заключается в непосредственном облегчении и ускорении работы пользователя с документацией. В обычной жизни без посторонней помощи в виде технологий на ручной анализ полной документации об объекте, которая включает в себя сразу три огромных раздела – архитектурный, конструктивный, инженерный, уходит огромное количество времени. При этом необходимо учитывать, что огромное количество проектной документации увеличивает риск человеческих ошибок. Использование искусственного интеллекта позволит повысить прозрачность и качество сравнения, он автоматизирует процесс и поможет выявить несоответствия на ранних этапах, предотвращая финансовые и временные потери в будущем. Этот бот будет упрощать работу с огромным количеством данных, сведений и информации, так как производит самостоятельно полную

работу по сравнению документов. А также сможет выгрузить сравнительный отчет в любом разрешении (.pdf, .txt, .xls), что будет облегчать процесс отчетности и удобство передачи информации между всеми участниками проекта. Человеку останется только проанализировать полученный сравнительный отчет и провести исправительные работы по объекту. Таким образом, разработка такого чат-бота отвечает современным методам цифровизации строительной отрасли, позволяет повысить качество, скорость и прозрачность процессов согласования проектной документации и существенно снижает риски на этапах проектирования и строительства.

Область применения бота: архитектурные, инженерные и строительные компании, отделы технического контроля и документационного обеспечения, заказчики, участвующие в крупных строительных и реконструкционных проектах, проектные бюро и консалтинговые фирмы, работающие с тендерной документацией.

В конечном итоге должен получиться бот GPT, в который пользователь может выгружать документацию по объекту и техническое задание объекта. Бот будет проводить анализ и сравнивать документы между собой, находить расхождения и добавлять их в итоговый сравнительный отчет. Результатом будет быстрый поиск информации по объекту и решений, которые на данный момент не реализованы по техническому заданию в проекте.

1 Основная часть

1.1 Разбор требований заказчика и пользователей к продукту. Составление плана действий

1.1.1 Требования заказчика, составленное им техническое задание

Для работы с проектом было представлено следующее техническое задание.

Чат-бот предполагает сравнение Технического задания от заказчика, где есть информация по проектируемому объекту и Проектную документацию. Анализ проводится с помощью ИИ. ИИ должна изучить и проанализировать всю информацию по объекту и выполнить сравнение ТЗ от заказчика и Проектной документации.

Перед загрузкой файлов, бот должен предложить список того, что именно мы сравниваем:

– Архитектурный раздел,

– Конструктивный раздел,

– Инженерный раздел,

После того как был выбран нужный раздел, нужно подгрузить необходимый документ. Бот производит анализ при помощи ИИ, все сравнивает и сопоставляет. После чего получаем результат сравнения, который можно выгрузить в форматах: .pdf ; .txt ; .xls.

Цель на семестр для команды - полностью создать бота, который может выполнять данный функционал.

Задача №1. Сравнение Архитектурного раздела с готовым ТЗ. Проводим сравнение всех материалов в стенах и полах, все ли соответствует итоговому ТЗ? Кроме материала, так же сравниваем толщину стен.

Итог завершения задачи – получение сравнительного файла со всеми результатами, в котором отображена информация, что именно не совпадает и на какой странице.

Задача №2. Сравнение Конструктивного раздела с готовым ТЗ. Сравниваем материал стен.

Задача №3. Сравнение Инженерного раздела с готовым ТЗ. Сравниваем производителей оборудования отопления, вентиляции, кондиционирования

1.1.2 Составленный план действий на семестр

1.2 Информация о работе каждого участника

Аналитики выполняли сбор и анализ требований, создание отчета, презентаций.

Разработчик выполнял написание кода, интеграцию GPT-модели.

Второй разработчик выполнял проведение тестов, выявление ошибок.

Тимлид следил за контролем сроков и вел координацию команды.

1.3 Информация о планировании деятельности в ходе разработки и распределении задач между участниками команды разработчиков

Для составления более слаженной работы и выполнения ее в срок необходимо составить план действий по достижению целей. Он был составлен Тимлидом и состоял из следующих пунктов:

Этап 1: Анализ требований и сбор данных.

Аналитики: Рассмотрение требований, сбор информации об нейросетях, которые можно использовать при разработке, определение целевой аудитории. Анализ и сопоставление аналогов разрабатываемого продукта, работа с целевой аудиторией.

Тимлид: Организация стартовой встречи, постановка задач, создание плана разработки.

Этап 2: Проектирование архитектуры чат-бота.

Разработчики: На базе проанализированных требований разрабатывается архитектура системы. Определяется, какие модули будут обеспечивать загрузку документов, их предварительную обработку. Проектируется логика хранения и поиска документов, механизмы сравнения документов по заданным параметрам и система уведомлений о найденных расхождениях.

Этап 3: Реализация функционала сравнения документов.

Разработчики: На этом этапе реализуются основные модули системы. Разрабатывается функционал загрузки и хранения документов, реализуются механизмы предварительной обработки текста, а также интеграция с ИИ для анализа и выявления различий между документами. Особое внимание уделяется настройке функции сравнения по заданным параметрам (по разделам, техническим характеристикам). Реализуется интерфейс взаимодействия: пользователь может загружать, просматривать, сравнивать документы через чат-бота. Обеспечивается возможность поиска информации и формирования отчётов по результатам анализа.

Этап 4: Тестирование и отладка.

Разработчики: Проводится комплексное тестирование всех компонентов: загрузка документов, корректность их обработки, стабильность работы бэкенда, функциональность бота. Проверяется точность работы алгоритмов сравнения и полнота выявления расхождений. В тестовой среде прогоняются различные кейсы (наборы документов с разными форматами и типами ошибок). Исправляются найденные недочёты, проводится оптимизация работы для увеличения скорости поиска и анализа.

Этап 5: Внедрение и документирование.

Разработчики: Готовый чат-бот разворачивается на целевой платформе.

Аналитики: Производится написание отчета по итогам проделанной работы.

1.3.1 Анализ аналогов и выявление лучшего для работы над проектом

В начале проекта мы провели анализ ИИ, которые мы можем применить в проекте [1, 2].

Анализ возможностей Chat GPT в сравнении документации.

Chat GPT — мощная модель для обработки и анализа текстовой информации. В задаче сравнения архитектурного, конструктивного и инженерного разделов с готовым ТЗ он может использоваться для автоматизации рутинных проверок и анализа документов.

Плюсы Chat GPT в выполнении функций анализа и сравнения документации.

– продвинутое понимание контекста, может интерпретировать сложные текстовые структуры и анализировать большое количество документации;

– генерация отчетов, модель способна создавать сравнительные файлы с результатами анализа, включая информацию о несоответствиях;

– гибкость в работе с различными форматами текста, chat GPT может обрабатывать текст в разнообразных форматах, что делает его применимым к сложной документации;

– поддержка длинных контекстов;

– гибкость промптов дает возможность легко адаптировать запросы на любой вид анализа или сравнения;

– не требует self-hosted развертывания, chatGPT работает на облачных серверах, что избавляет пользователей от необходимости собственноручно развертывать систему на своих серверах.

Минусы Chat GPT в выполнении функций анализа и сравнения документации.

– ограниченные возможности в анализе числовой информации;

– использование Chat GPT для масштабного анализа может потребовать значительных вычислительных мощностей и времени на обработку данных;

– ограниченные возможности в выводах без конкретного контекста;

– в некоторых случаях без дополнительных данных или механизма проверки модель может сделать неверные выводы, особенно если информация неоднозначна или противоречива;

Анализ возможностей Deep Seek в сравнении документации.

Deep Seek — это специализированный инструмент искусственного интеллекта, разработанный для глубокого анализа и понимания текстовой и числовой информации. В контексте сравнения архитектурного, конструктивного и инженерного разделов с готовым техническим заданием (ТЗ), он может предложить ряд преимуществ и недостатков.

Плюсы Deep Seek в выполнении функций анализа и сравнения документации.

– высокая точность в анализе текстовых и числовых данных;

– система способна автоматически выявлять несоответствия между разными разделами документов и ТЗ, deep seek создает сравнительные файлы, которые четко указывают на различия и на какой странице они находятся;

– интеграция с различными форматами данных, deep seek поддерживает множество форматов данных, что упрощает процесс анализа и сравнения сложной документации, включая чертежи, спецификации и технические условия;

– система способна обрабатывать большие объемы данных, ускоряя процесс анализа и сравнения документации;

– deep Seek предоставляет возможности для настройки алгоритмов под конкретные задачи и требования.

Минусы Deep Seek в выполнении функций анализа и сравнения документации.

– сложность настройки и интеграции, для эффективного использования Deep Seek может потребоваться значительное время на настройку систем и интеграцию с существующими рабочими процессами и документами;

– необходимость обучения сотрудников, поскольку система может быть сложной в использовании;

– высокая стоимость;

– потенциальные проблемы с интерпретацией неоднозначных данных;

– для поддержания высокой точности и актуальности анализа документации может потребоваться регулярное обновление системы и её адаптация под новые требования и стандарты;

Анализ возможностей Giga Chat в сравнении документации.

Giga Chat — это современный инструмент искусственного интеллекта, который предназначен для обработки больших массивов данных и выполнения сложных аналитических задач, включая сравнение и анализ документации.

Плюсы Giga Chat в выполнении функций анализа и сравнения документации.

Продвинутый обработчик естественного языка. Giga Chat может эффективно анализировать текст, выделять ключевые моменты и искать несоответствия.

– интерактивное взаимодействие и ответы в реальном времени;

– система предоставляет возможности для настройки и дополнения функционала путем добавления модулей, что позволяет адаптировать Giga Chat для конкретных задач, таких как сравнение толщины стен или анализ оборудования;

– способность к обучению и адаптации;

– гибкость в работе с неоднозначными данными.

Минусы Giga Chat в выполнении функций анализа и сравнения документации.

– Giga Chat может потребовать высокое качество и структуру входных данных для максимально точного анализа и сравнения, что может обусловливать дополнительные усилия по подготовке документации.

– ограниченная способность к обработке сложных числовых данных;

– при обработке очень больших объемов данных эффективность работы Giga Chat может снизиться, что может потребовать дополнительных ресурсов для оптимизации;

– необходимая поддержка со стороны технических специалистов;

– Зависимость от интернета.

Анализ возможностей Copilot в сравнении документации.

Copilot, разработанный совместно GitHub и OpenAI, позиционируется как помощник для программистов, однако его способности анализа и обработки текста позволяют рассмотреть его потенциал и в сравнении документации. В контексте задачи по сравнению архитектурного, конструктивного и инженерного разделов с готовым техническим заданием (ТЗ), Copilot имеет свои особенности.

Плюсы Copilot в выполнении функций анализа и сравнения документации.

– Copilot использует мощные алгоритмы для анализа текста, что дает возможность искать несоответствия в документации, особенно в текстовых описаниях материалов стен и полов;

– глубокая интеграция с инструментами разработки;

– способность Copilot предлагать продолжения текста может быть использована для автоматической генерации отчетов о несоответствиях;

– Copilot может эффективно работать с таблицами и структурированными данными;

– эффективность для шаблонных задач.

Минусы Copilot в выполнении функций анализа и сравнения документации.

– ограниченная ориентация на анализ текста;

– требуется высокая степень структурированности данных;

– проблемы с развернутым анализом сложных документов;

– зависимость от среды разработки, Copilot работает преимущественно в IDE;

– ограниченные возможности для глубокого анализа графических данных.

Анализ возможностей OpenAI Embeddings в сравнении документации.

OpenAI Embeddings — это инструмент для представления текстовой информации в виде векторов, что используется для поиска смысловой близости, анализа и сравнения документов. Рассмотрим его потенциал в задаче сравнения архитектурного, конструктивного и инженерного разделов с готовым техническим заданием (ТЗ) и специфику применения.

Плюсы OpenAI Embeddings в выполнении функций анализа и сравнения документации.

– OpenAI Embeddings преобразует текст в векторные представления, что позволяет искать информацию по смыслу, а не только по ключевым словам.

– может обрабатывать любые текстовые данные (PDF, DOCX, TXT и др.), если они преобразованы в читаемый текст;

– легко комбинируется с такими инструментами, как DeepSeek, LangChain, векторные БД (FAISS, Pinecone);

– нет необходимости в fine-tuning, готовая модель, не требующая дообучения под конкретные задачи;

– эффективность при больших объемах данных, оптимизирован для быстрого поиска в больших документах.

Минусы OpenAI Embeddings в выполнении функций анализа и сравнения документации.

– не анализирует данные напрямую, OpenAI Embeddings только преобразует текст в векторы, но не умеет анализировать, сравнивать или генерировать выводы;

– не работает с графикой и таблицами;

– если данные в PDF представлены в виде изображений или сложных таблиц, их нужно предварительно извлекать;

– зависит от качества текста, требуется предварительная очистка и нормализация данных;

– нет встроенной генерации отчетов, OpenAI Embeddings не формирует отчеты — для этого нужен дополнительный слой;

– ограниченная глубина контекста, хорошо работает на уровне абзацев, но может терять глобальный контекст документа.

Так как ни один искусственный интеллект не отвечал в полной мере всем необходимым запросам для реализации проекта, было принято решение задействовать две нейросети, а именно DeepSeek в связке с OpenAI Embeddings.

Преимущества связки DeepSeek и OpenAI Embeddings для проекта [3].

а) Семантический поиск вместо буквального сравнения.

Вместо простого сопоставления ключевых слов, связка моделей обеспечивает поиск по смыслу. Это особенно важно при работе с архитектурной документацией, где одно и то же требование может быть сформулировано разными словами.

б) Автоматизация и ускорение анализа.

Весь процесс—from преобразования текста до формирования структурированного отчета—максимально автоматизирован. Не требуется ручная обработка каждого документа.

в) Сокращение ошибок при извлечении данных.

DeepSeek анализирует всю базу векторов, что позволяет находить даже те фрагменты, которые сложно было бы отследить вручную или с помощью простых алгоритмов поиска.

г) Гибкость по формату получаемых данных.

После извлечения информации отчёт автоматически формируется в нужном формате (Excel, PDF, TXT).

д) Масштабируемость и универсальность.

Система одинаково эффективна при работе как с маленькими, так и с большими объёмами документов. Подход подходит для разных типов инженерных, архитектурных и технических файлов.

е) Контекстуальность и структурированность результатов.

Системный промпт автоматически структурирует найденную информацию, приводя её к удобному для анализа виду.

Связка DeepSeek и OpenAI Embeddings подходит для задачи сравнения и анализа инженерной и архитектурной документации. Она обеспечивает большую точность, значительно ускоряет процесс поиска и сбора данных, а также позволяет масштабировать решение под нужды проекта без больших трудозатрат на ручную обработку информации. Такая связка позволит компенсировать недостатки и получить максимальную эффективность.

Рассмотрит плюсы и минусы анализируемых ИИ в таблице «Таблица сравнения нейросетей».

Таблица 1 – Таблица сравнения нейросетей

Критерий	Chat GPT	DeepSeek	Giga Chat	Copilot	OpenAI Embeddings
Анализ текста	Высокое качество, понимание контекста	Высокая точность, работа с текстом и числами	Продвинутая обработка естественного языка	Контекстуальный анализ, предиктивные возможности	Преобразование текста в векторы для семантического поиска
Анализ чисел	Ограниченный	Отличный	Ограниченный	Умеренный (для структурированных данных)	Не применяется напрямую

Продолжение таблицы 1

Генерация отчетов	Да (автоматизированная)	Да (автоматизированная)	Да (интерактивная)	Да (предиктивная)	Нет (используется для поиска)
Форматы данных	Гибкость (разные текстовые форматы)	Поддержка множества форматов, включая чертежи	Модульная архитектура для адаптации	Требует структурированных данных	Работает с текстом из PDF
Интеграция	LangChain (RAG), облачные серверы	Требует настройки	Модульная архитектура	Глубокая интеграция с IDE	Интеграция с DeepSeek
Обучение/настройка	Не требуется fine-tuning	Требуется обучение сотрудников	Способность к обучению	Не требуется	Не требуется
Ресурсы	Высокие вычислительные затраты	Высокая стоимость внедрения	Зависит от объема данных	Зависит от среды разработки	Низкие (для преобразования)
Плюсы	Гибкость, простота использования	Точность, автоматизация проверок	Интерактивность, адаптивность	Автоматизация рутинных задач	Семантический поиск
Минусы	Слабая работа с числами	Сложность настройки	Требует качественных входных данных	Ограничен для глубокого анализа	Не анализирует данные напрямую

1.3.2 Анализ целевой аудитории

Целевая аудитория чат-бота — это специалисты, вовлечённые в процесс проектирования, согласования и экспертизы проектной документации. Основная задача чат-бота — автоматизировать и упростить проверку соответствия между исходным техническим заданием и готовой проектной документацией, снижая вероятность ошибок и повышая скорость анализа.

Основные категории пользователей.

5) Заказчики проектов.

Цель: Проверка соответствия проекта изначально поставленным требованиям.

Потребности: Убедиться, что ПД соответствует утверждённому ТЗ, без необходимости вручную перечитывать оба документа.

Ожидания: Простота использования, надёжность результатов, возможность получить выводы и комментарии.

6) Проектировщики и архитекторы.

Цель: Самопроверка проекта на соответствие ТЗ до передачи заказчику или в экспертизу.

Потребности: Быстро находить несоответствия и устранить их до сдачи проекта.

Ожидания: Гибкость анализа, возможность уточнения параметров проверки, детализация отчётов.

7) Эксперты и специалисты по технадзору.

Цель: Формализованный и объективный подход к анализу проектной документации.

Потребности: Инструмент для первичного анализа, ускоряющий работу эксперта.

Ожидания: Точность, соответствие нормативной базе, возможность экспорта отчётов.

8) Менеджеры проектов и BIM-координаторы.

Цель: Контроль за соблюдением требований на всех стадиях проектирования.

Потребности: Интеграция с другими системами, отслеживание истории изменений.

Ожидания: Интуитивно понятный интерфейс, адаптация под корпоративные стандарты.

Уровень технической подготовки.

Пользователи, как правило, обладают:

– средним или высоким уровнем владения технической терминологией;

– знанием проектных нормативов и ГОСТов;

– определённым опытом работы с проектной документацией (PDF, DWG, Word, Excel и др.).

Болевые точки и потребности.

– ручная проверка соответствия ТЗ и ПД отнимает много времени;

– высок риск человеческой ошибки при анализе большого объёма документов;

– отсутствие унифицированного подхода к верификации ТЗ и ПД;

– сложность коммуникации между участниками проектного процесса из-за расхождений.

Как ИИ-чат-бот закрывает потребности.

– автоматически находит и подсвечивает расхождения между ТЗ и ПД;

– генерирует структурированные отчёты;

– работает с различными форматами файлов;

– может обучаться на данных компании (тонкая настройка под внутренние стандарты);

– быстро адаптируется под новые проекты и изменения;

– дополненный анализ целевой аудитории чат-бота для сравнения ТЗ и проектной документации с помощью GPT.

Уточнённая характеристика аудитории.

Целевая аудитория чат-бота включает профессионалов строительной и проектной сферы, которые нуждаются в автоматизированном инструменте для сравнения и анализа технической документации. Основная ценность бота — сокращение времени проверки, минимизация ошибок и стандартизация процесса сверки документов.

Методы сбора данных

Для анализа были использованы следующие инструменты:

– изучение форумов и обсуждений на профессиональных платформах (vc.ru, habr, Telegram-чаты);

– анализ отзывов на существующие инструменты для работы с документацией.

Ключевые выводы анализа.

Основная проблема: ручное сравнение ТЗ и ПД занимает много времени и часто приводит к ошибкам из-за человеческого фактора.

Интерес к автоматизации: большинство опрошенных выразили заинтересованность в инструменте, который позволит находить расхождения автоматически.

Форматы документов: наиболее распространенные — .docx, .pdf, .xlsx. Возможность загружать и обрабатывать эти форматы — критически важна.

Желаемый функционал:

– подсветка отличий и несоответствий;

– возможность фильтрации по типам изменений (текстовые, числовые, структурные);

– генерация отчета по итогам сравнения;

– простота интерфейса и отсутствие необходимости в обучении.

Проведённый анализ помог четко определить целевую аудиторию и их ключевые запросы. Это дало возможность сформировать обоснованные требования к функционалу чат-бота, его интерфейсу и алгоритмам ИИ-сравнения.

1.4 Обзор архитектуры программного продукта, описание основных компонентов и связей между ними, обоснование выбора архитектурного решения

Основные компоненты:

Пользовательский интерфейс: Telegram Бот.

Telegram Бот выступает в роли основного канала взаимодействия пользователей с системой. Создается удобный и интуитивно понятный интерфейс, через который пользователи могут отправлять документы, формулировать запросы, получать результаты анализа и отчёты о найденных расхождениях. Такой подход обеспечивает кроссплатформенность, отсутствие необходимости устанавливать дополнительное ПО, а также простоту внедрения и настройки в корпоративной среде.

Обработчик документов: Модуль для парсинга и анализа ТЗ и проектной документации.

Обработчик документов выполняет ключевые задачи по приёму, предварительной обработке и подготовке документов к анализу:

– поддержка различных форматов документов (PDF, DOCX и др.).

– извлечение текста и структурирование информации.

– первичная фильтрация и нормализация данных (удаление артефактов, разделение по пунктам, разбивка по параметрам).

– формирование унифицированного представления содержимого для передачи в GPT-модель.

Данный модуль строится с учетом расширяемости: при необходимости можно добавлять новые форматы документов, дорабатывать логику обработки под специфику входных данных.

GPT-модель: ИИ для сравнения текстов и выявления расхождений (DeepSeek, OpenAI Embeddings).

Связи: Пользователь → Интерфейс → Обработчик → GPT-модель → Отчет.

Пошаговое описание процесса.

1) Преобразование PDF-документов.

Каждый исходный PDF-документ проходит этап преобразования текста в векторные представления с помощью модели OpenAI Embeddings. Это необходимо для того, чтобы сохранить семантическую и контекстуальную информацию из исходного текста, а также обеспечить возможность быстрого поиска данных по смыслу, а не только по ключевым словам.

2) Поиск информации.

После преобразования все полученные векторы вместе с исходными фрагментами текста загружаются в систему DeepSeek. Далее, когда возникает задача найти определенную информацию (например, только значения толщины стен или названия используемых материалов), соответствующий текстовый запрос также преобразуется в вектор.

DeepSeek анализирует всю базу данных векторов и находит те фрагменты, которые максимально соответствуют запросу по смыслу, а не просто совпадают по словам. Это делает поиск гораздо эффективнее и качественнее.

3) Сбор информации.

После получения релевантных фрагментов от DeepSeek собирается вся необходимая информация из выбранных документов. Для этого формируется специальный системный промпт, который структурирует данные и подготавливает их к дальнейшему использованию.

4) Формирование отчета.

На заключительном этапе пользователь получает компактный и информативный отчет, составленный на основе извлеченной информации. DeepSeek возвращает ответ, который оформляется в различных вариантах формата на выбор пользователя:

- Excel;
- PDF;
- TXT;

Обоснование архитектуры:

1) Использование GPT-модели обеспечивает высокую точность анализа.

Применение современных языковых моделей позволяет эффективно решать задачи поиска различий даже в сложных и объёмных текстах технической документации, где традиционные методы сопоставления малоприменимы.

2) Модульность позволяет легко масштабировать систему.

Архитектура построена по модульному принципу: каждый компонент отвечает за свою зону ответственности и может дорабатываться или заменяться независимо от других. Это позволяет расширять поддержку новых форматов документов, подключать альтернативные ИИ-модели, интегрироваться с внешними сервисами (например, корпоративными порталами или системами хранения документации) без риска для устойчивости платформы.

1.5 Описание методологии разработки, информация о процессе разработки, отчет о результатах тестирования на промежуточных этапах, разбор выявленных ошибок

1.5.1 Методология и процесс разработки.

1) Методология: Agile с итерациями.

Для разработки программного продукта была выбрана гибкая методология Agile. Такой подход основан на работе короткими итерациями (спринтами), постоянной обратной связью и гибкой доработкой требований в процессе создания решения. Благодаря Agile команда имела возможность быстро реагировать на изменения, корректировать приоритеты задач и вносить улучшения.

2) Процесс разработки.

Регулярные демонстрации функционала.

3) Тестирование.

Для каждого отдельного модуля реализовывались unit-тесты, чтобы убедиться в корректности работы ключевых функций. Unit-тестирование позволяло оперативно отлавливать ошибки в логике на ранних этапах, не дожидаясь сборки всех компонентов системы.

При помощи интеграционных тестов проверялась совместная работа всех компонентов: корректность передачи данных между интерфейсом, обработчиком документов и GPT-моделью; отсутствие утечек и сбоев на уровне коммуникаций между сервисами.

4) Выявленные ошибки.

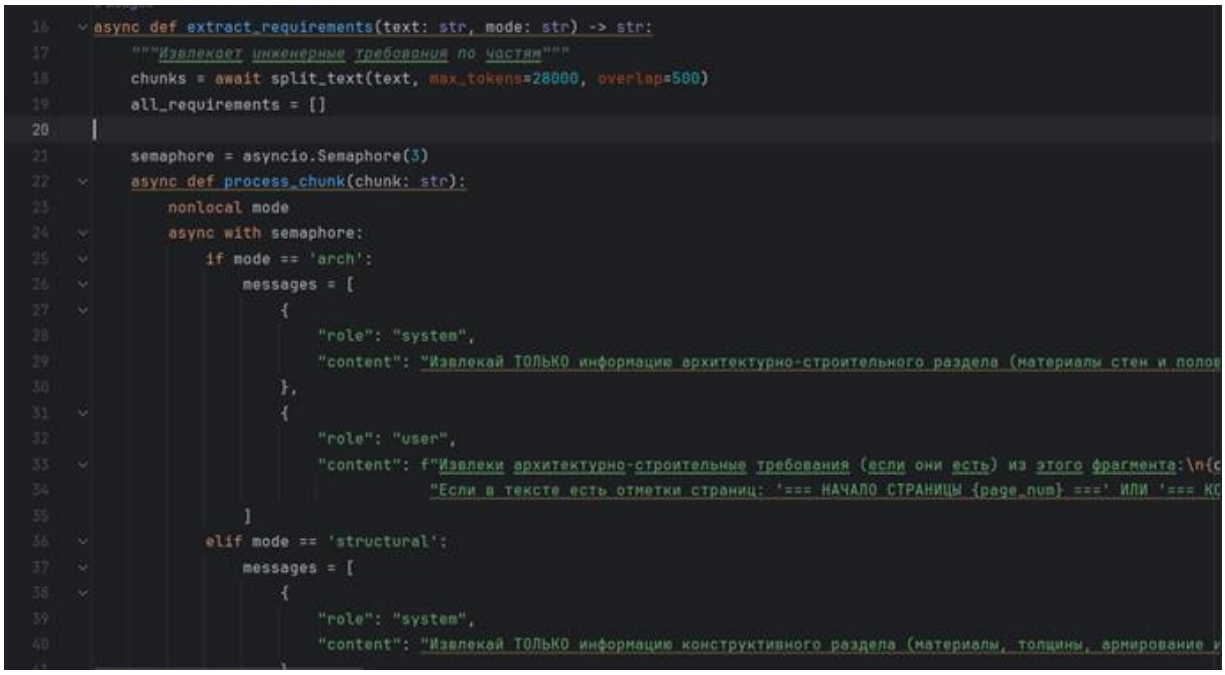
В процессе тестирования были замечены случаи некорректной интерпретации проектной документации, когда модель давала слишком общее или неструктурированное сравнение.

Проведен разбор результатов, после чего был детализирован и уточнён системный промпт для GPT-модели, что позволило добиться структурированных и релевантных сравнений документации.

Также при тестировании на больших документах наблюдалась высокая задержка в получении ответа от ИИ. Анализ показал, что задержки связаны с избыточными данными и неэффективной подготовкой запросов к модели. Проведена оптимизация этапа подготовки данных: внедрена предварительная фильтрация информации, ограничено избыточное дублирование текста и оптимизированы логика упаковки данных для отправки в модель. В результате среднее время ожидания ответа существенно сократилось.

1.5.2 Кодовые решения

На рисунках представлены кодовые решения проекта для извлечения требований из документов (рисунок 1), сравнения документов с помощью ИИ (рисунок 2), структурирования отчета (рисунок 3).

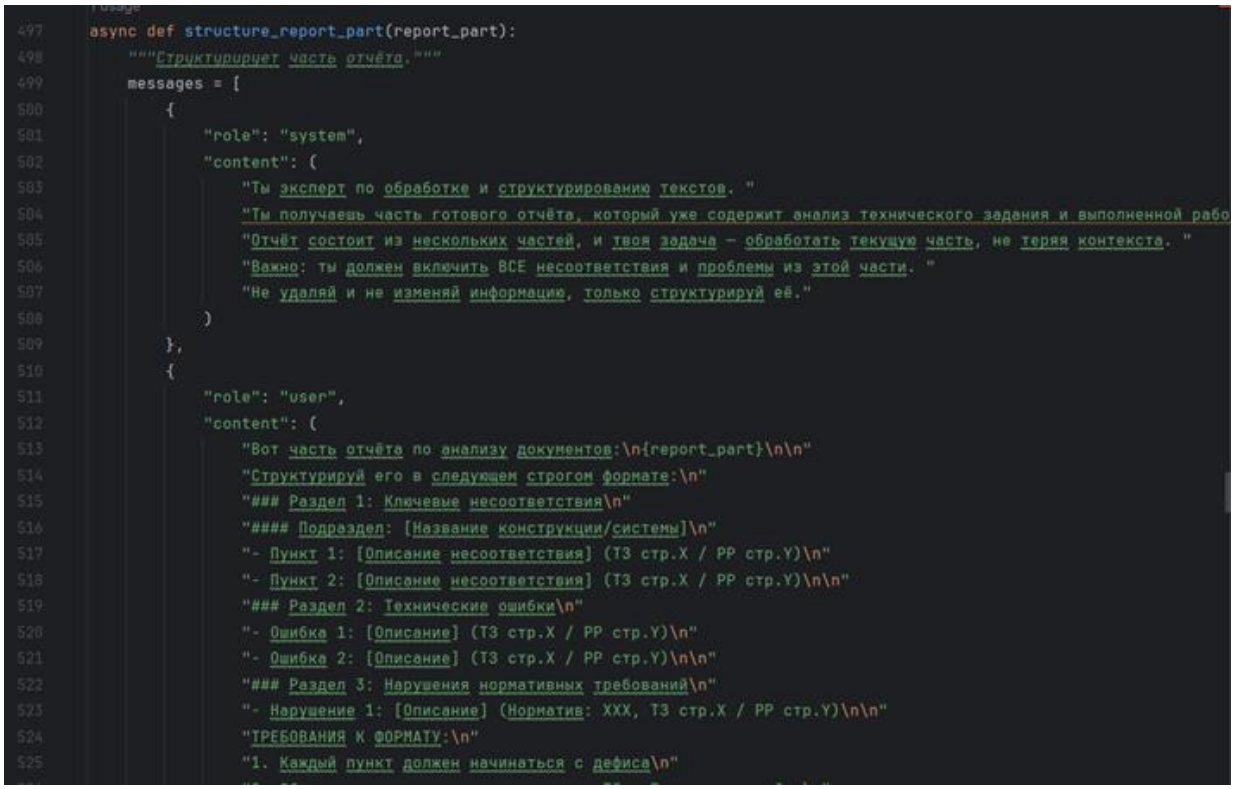


```python
async def extract_requirements(text: str, mode: str) -> str:
    """Извлекает инженерные требования по частям"""
    chunks = await split_text(text, max_tokens=28000, overlap=500)
    all_requirements = []

    semaphore = asyncio.Semaphore(3)
    async def process_chunk(chunk: str):
        nonlocal mode
        async with semaphore:
            if mode == 'arch':
                messages = [
                    {
                        "role": "system",
                        "content": "Извлекай ТОЛЬКО информацию архитектурно-строительного раздела (материалы стен и полов
                    },
                    {
                        "role": "user",
                        "content": f"Извлеки архитектурно-строительные требования (если они есть) из этого фрагмента:\n{c
                                   "Если в тексте есть отметки страниц: '=== НАЧАЛО СТРАНИЦЫ {page_num} ===' ИЛИ '=== КО
                ]
            elif mode == 'structural':
                messages = [
                    {
                        "role": "system",
                        "content": "Извлекай ТОЛЬКО информацию конструктивного раздела (материалы, толщины, армирование и
```

Рисунок 1 – Извлечение требований из документов

```python
async def compare_documents(technical_spec, result_doc, mode: str = "arch"):
    """Сравнивает разделы ТЗ и результата с учетом типа анализа."""
    mode_prompts = {
        "arch": {
            "system": "Анализ архитектурных решений."
                      "Если производитель или материал совпадает - несоответствия нет. "
                      "Если производитель или материал, но в ТЗ есть пометка о том, что аналог из результата можно испол
                      "Если производитель или материал и аналог не допускается - укажи как несоответствие.",
            "user": "Проверь соответствие архитектурных требований и реализации. Особое внимание удели всем материалам с
        },
        "structural": {
            "system": "Анализ конструктивных решений."
                      "Если производитель или материал совпадает - несоответствия нет. "
                      "Если производитель или материал отличается, но в ТЗ есть пометка о том, что аналог из результата
                      "Если производитель или материал отличается и аналог не допускается - укажи как несоответствие.",
            "user": "Проверь соответствие конструктивных требований и реализации. Особое внимание удели соответствию нес
        },
        "engineer": {
            "system": "Анализ инженерных систем."
                      "Если производитель или материал совпадает - несоответствия нет. "
                     "Если производитель или материал отличается, но в ТЗ есть пометка о том, что аналог из результата мо
                     "Если производитель или материал отличается и аналог не допускается - укажи как несоответствие.",
            "user": "Проверь соответствие инженерных требований и реализации. Особое внимание удели соответствию указанн
        }
    }

    prompt = mode_prompts.get(mode, mode_prompts["engineer"])
```

Рисунок 2 – Сравнение документов с помощью ИИ



```
async def structure_report_part(report_part):
    """Структурирует часть отчёта."""
    messages = [
        {
            "role": "system",
            "content": (
                "Ты эксперт по обработке и структурированию текстов. "
                "Ты получаешь часть готового отчёта, который уже содержит анализ технического задания и выполненной рабо
                "Отчёт состоит из нескольких частей, и твоя задача – обработать текущую часть, не теряя контекста. "
                "Важно: ты должен включить ВСЕ несоответствия и проблемы из этой части. "
                "Не удаляй и не изменяй информацию, только структурируй её."
            )
        },
        {
            "role": "user",
            "content": (
                "Вот часть отчёта по анализу документов:\n{report_part}\n\n"
                "Структурируй его в следующем строгом формате:\n"
                "### Раздел 1: Ключевые несоответствия\n"
                "#### Подраздел: [Название конструкции/системы]\n"
                "- Пункт 1: [Описание несоответствия] (ТЗ стр.X / РР стр.Y)\n"
                "- Пункт 2: [Описание несоответствия] (ТЗ стр.X / РР стр.Y)\n\n"
                "### Раздел 2: Технические ошибки\n"
                "- Ошибка 1: [Описание] (ТЗ стр.X / РР стр.Y)\n"
                "- Ошибка 2: [Описание] (ТЗ стр.X / РР стр.Y)\n\n"
                "### Раздел 3: Нарушения нормативных требований\n"
                "- Нарушение 1: [Описание] (Норматив: XXX, ТЗ стр.X / РР стр.Y)\n\n"
                "ТРЕБОВАНИЯ К ФОРМАТУ:\n"
                "1. Каждый пункт должен начинаться с дефиса\n"
```

Рисунок 3 – Структурирование отчета

1.5.3 Сложности при выполнении проекта

При внедрении сторонних нейросетей для анализа строительной и проектной документации возникло несколько существенных проблем.

1) Большой объём проектных документов.

Основное затруднение связано с тем, что строительная документация зачастую состоит из больших файлов — это могут быть многостраничные PDF-документы, схемы, таблицы, чертежи, спецификации. Стандартные нейросети, такие как ChatGPT, Copilot, DeepSeek, имеют ограничение на размер обрабатываемого текста.

Из-за этого крупные документы либо не загружаются полностью, либо приходится делить файлы на небольшие части вручную, что значительно усложняет рабочий процесс и увеличивает риск потери взаимосвязей между разделами документации.

2) Ограниченные возможности работы с вложенными и сложными структурами.

В проектных документах часто встречаются вложенные таблицы, спецификации и рисунки, которые плохо интерпретируются большинством универсальных языковых моделей. Многие полезные аспекты (например, ссылки между разделами, поясняющие примечания на схемах) могут теряться после конвертации файлов или при попытке преобразовать их в пригодный для нейросети вид.

3) Ограничения при использовании OpenAI.

Одной из наиболее популярных платформ с мощными языковыми моделями является OpenAI (включая ChatGPT, GPT-4, OpenAI Embeddings). Однако для работы с OpenAI из России и ряда других стран требуется подключение к VPN, что вызывает ряд трудностей:

– не все корпоративные или рабочие компьютеры поддерживают стабильную работу через VPN;

– возникают задержки в работе из-за снижения скорости соединения;

– могут возникнуть юридические вопросы или ограничения по политике информационной безопасности в компании;

– использование VPN иногда приводит к сбоям доступа или некорректной работе сервисов.

ЗАКЛЮЧЕНИЕ

Разработанный программный продукт — бот для анализа проектной документации — в полной мере соответствует основным требованиям, заявленным заказчиком. Инструмент предоставляет возможность автоматизированного анализа и сравнения документов, относящихся к различным объектам строительства.

Бот позволяет пользователям быстро и наглядно выявлять ключевые отличия между техническими заданиями (ТЗ) и другими связанными документами по одному и тому же объекту. Благодаря этому устраняется необходимость ручной проверки больших объёмов текстов, сокращается время анализа, минимизируется влияние человеческого фактора и повышается прозрачность процесса принятия решений.

По итогам проведённых тестов можно сделать вывод, что бот корректно обрабатывает текстовые данные из предоставленных документов, осуществляет поиск различий и формирует релевантные отчёты по результатам сравнения.

Однако, несмотря на достигнутые результаты, в процессе тестирования были выявлены аспекты, требующие дальнейшего улучшения. В частности, при работе с PDF-файлами с низким качеством сканирования или сложной структурой у бота иногда возникают затруднения с корректным распознаванием текста. Это может привести к неполной или ошибочной выдаче различий между документами. Данный вопрос требует доработки механизма распознавания и оптимизации обработки PDF-файлов, чтобы минимизировать вероятность подобных ситуаций в будущем.

Предложения по улучшению и развитию продукта

1) Реализация поддержки облачных хранилищ.

Для повышения удобства использования и расширения возможностей обмена информацией целесообразно интегрировать бот с популярными облачными хранилищами (такими как Google Drive, Dropbox, Яндекс.Диск и другие). Это позволит пользователям загружать документы непосредственно

из облака, работать с ними совместно, а также повысит скорость обработки и обмена файлами между участниками проектных команд.

2) Добавление гибких настроек параметров анализа

Для различной специфики проектных команд и объектов целесообразно реализовать расширенные настройки параметров анализа. Пользователи смогут самостоятельно определять критерии сопоставления, акцентировать внимание на определённых разделах или типах данных, выбирать уровень детализации результатов сравнения.

Такой подход позволит адаптировать работу бота под индивидуальные потребности пользователей и обеспечить более высокую точность и ценность анализа.

В целом, дальнейшее развитие и совершенствование инструментов обработки документов повысит ценность продукта для конечных пользователей, упростит рабочие процессы и позволит заказчику достичь ещё более эффективных результатов при реализации строительных и инженерных проектов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. DeepSeek vs ChatGPT: Comparison of Best AI Titans in 2025: [Электронный ресурс] – URL: https://www.geeksforgeeks.org/deepseek-vs-chatgpt/ (дата обращения 07.04.25).

2. ChatGPT против GigaChat против Microsoft Copilot Сравнительная таблица: [Электронный ресурс] – URL: https://sourceforge.net/software/compare/ChatGPT-vs-GigaChat-vs-Microsoft-Copilot/ (дата обращения 02.04.25).

3. Carter, S. DeepSeek Embedding Model: A Comprehensive Guide [Электронный ресурс] / Carter Steve. – URL: https://irnpost.com/deepseek-embedding-model-a-comprehensive-guide/ (дата обращения 05.04.25).