НПОУ «ЯКУТСКИЙ КОЛЛЕДЖ ИННОВАЦИОННЫХ ТЕХНОЛОГИЙ» (НПОУ «ЯКИТ»)

Отделение информационных технологий и туризма

Специальность 09.02.07 «Информационные системы и программирования»

КУРСОВАЯ РАБОТА

по дисциплине: Языки программирования

по теме:

Создание десктопного приложения для семантического анализа русского языка с использованием библиотеки Hugging Face Transformers на Python

Исполнитель студент гр. КИСП-23(1)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Л.П. Лебедева

подпись, дата

Руководитель,

преподаватель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д.А.Фёдоров

подпись, дата

Якутск, 2025

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc194747538)

[1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 5](#_Toc194747539)

[1.1 Терминология 5](#_Toc194747540)

[1.2 Распределение ролей и работы 6](#_Toc194747541)

[1.3 Стек технологий 7](#_Toc194747542)

[2.ПРАКТИЧЕСКАЯ ЧАСТЬ 8](#_Toc194747543)

[2.1. Архитектура приложения 8](#_Toc194747544)

[2.2.Разработка проекта по ролям 9](#_Toc194747545)

[2.3.Контроль выполнения плана 10](#_Toc194747546)

[ЗАКЛЮЧЕНИЕ 11](#_Toc194747547)

# ВВЕДЕНИЕ

Актуальность создания десктопного приложения для семантического анализа текста обусловлена несколькими факторами:

1. Рост объемов текстовой информации: В условиях цифровизации и доступности информации на русском языке возрастает потребность в инструментах, позволяющих эффективно обрабатывать и анализировать текстовые данные.
2. Необходимость в автоматизированных решениях: Бизнес и научные исследования требуют автоматизации процессов анализа текстов, что позволяет сократить время и ресурсы, затрачиваемые на ручной анализ.
3. Развитие технологий обработки естественного языка: Современные модели машинного обучения, такие как трансформеры, значительно увеличили точность и скорость семантического анализа, что делает их применение актуальным и необходимым.

Основная цель проекта заключается в разработке десктопного приложения для семантического анализа русского языка, использующего библиотеку Hugging Face Transformers. Приложение должно обеспечивать пользователям возможность быстро и эффективно анализировать текстовые данные, выявляя их смысл и настроение.

Задачи проекта

1. Исследование теоретических основ семантического анализа: Изучить ключевые аспекты семантического анализа, включая методы и подходы, применяемые в обработке естественного языка.
2. Выбор технологий и инструментов: Определить оптимальный стек технологий для реализации приложения, включая язык программирования, библиотеки и инструменты для создания интерфейса.
3. Разработка интерфейса пользователя: Создать интуитивно понятный графический интерфейс, который позволит пользователям вводить текст и получать результаты анализа.
4. Имплементация модуля обработки текста: Реализовать функциональность семантического анализа с использованием предобученных моделей из библиотеки Hugging Face Transformers.
5. Тестирование и отладка приложения: Провести тестирование приложения, выявить и исправить возможные ошибки, а также убедиться в корректности работы всех функций.
6. Документация и представление результатов: Подготовить документацию по использованию приложения и представить результаты работы проекта заинтересованным сторонам.

Объектом исследования является процесс семантического анализа текстов на русском языке, а также применяемые в этом процессе алгоритмы и технологии обработки естественного языка.

Предметом исследования выступает десктопное приложение, реализующее семантический анализ текста с использованием библиотеки Hugging Face Transformers.

1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ
   1. Терминология

В данной подглаве будут подробно рассмотрены ключевые термины и концепции, связанные с семантическим анализом и обработкой естественного языка (NLP), которые будут полезны для понимания основ нашего проекта.

Семантический анализ — это процесс извлечения значений и смыслов из текстовых данных. Он включает в себя понимание контекста, смысловых связей между словами и предложениями. Семантический анализ может быть использован для решения различных задач, таких как:

Анализ тональности: Определение эмоциональной окраски текста (положительная, отрицательная или нейтральная).

Извлечение сущностей: Определение имён, организаций, мест и других значимых элементов в тексте.

Понимание контекста: Учет контекста, в котором используются слова, для более точного анализа значений.

Обработка естественного языка (NLP) — это область искусственного интеллекта, занимающаяся взаимодействием между компьютерами и человеческим языком. Цели NLP включают:

Понимание текста: Создание моделей, способных интерпретировать и анализировать текст на естественном языке.

Генерация текста: Создание новых текстов на основе заданных параметров или примеров.

Перевод текста: Автоматический перевод текста с одного языка на другой.

NLP охватывает множество методов и подходов, включая машинное обучение и глубокое обучение.

Токенизация — это процесс разделения текста на отдельные элементы, называемые токенами. Эти токены могут быть словами, фразами или символами. Токенизация является одним из первых шагов в обработке текста, поскольку она позволяет модели работать с отдельными частями текста.

Модели трансформеров — это современные архитектуры нейронных сетей, которые обеспечивают высокую точность в задачах NLP. Они основаны на механизме внимания, который позволяет моделям учитывать контекст слов в предложении, независимо от их расположения. Основные модели трансформеров:

BERT (Bidirectional Encoder Representations from Transformers): Модель, разработанная Google, которая позволяет учитывать контекст слов слева и справа от целевого слова, что значительно улучшает точность семантического анализа.

GPT (Generative Pre-trained Transformer): Модель от OpenAI, которая может генерировать текст и выполнять задачи, такие как ответ на вопросы и создание описаний, на основе заданного контекста.

Эти модели были предобучены на больших объемах текстовых данных и могут быть дообучены на специфичных для задачи данных, что делает их очень гибкими и мощными инструментами для семантического анализа.

Контекстуальные представления — это способ представления слов в виде векторов, где каждое слово имеет свое значение в зависимости от контекста, в котором оно используется. Это позволяет моделям более точно понимать смысл слов и их взаимосвязи. В отличие от статических векторных представлений (например, Word2Vec), контекстуальные представления учитывают порядок слов и их окружение в предложении.

Семантический анализ находит применение в различных областях, включая:

Маркетинг: Анализ отзывов и комментариев клиентов для определения их мнения о продукте или услуге.

Социальные сети: Мониторинг и анализ упоминаний брендов или событий.

Образование: Автоматическая оценка эссе и тестов на основе анализа содержания и структуры текстов.

* 1. Распределение ролей и работы

Лебедева Лира Петровна занималась написанием отчета, отвечала за исправление и написание отчета.

План по выполнению работы по части написания отчета:

1. Написание структуры отчета.
2. Написание отчета.
3. Просмотр отчета на наличие ошибок.
4. Исправление ошибок.

Динганорбоев Эрдэм Булатович занимался поиском информации, отвечал за теорию и сбор материала.

План по выполнению работы по части поиска информации:

1. Написание плана работы.
2. Поиск информации.
3. Выделение важных терминов и определений.

Иннокентьев Влад Александрович занимался программированием, отвечал за репозиторий.

План по выполнению работы по части программирования:

1. Создание основы.
2. Создание UI.
3. Добавление Hugging Face библиотеки.
4. Написание кода.

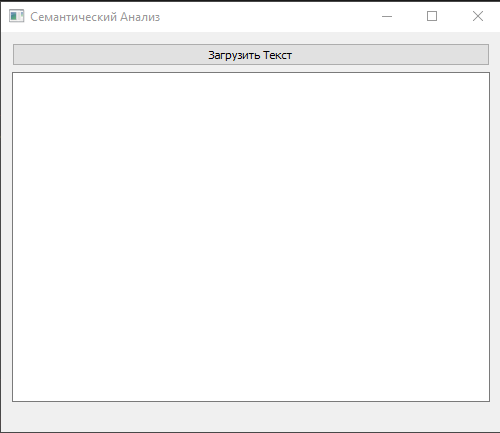
Николаев Аслан Анатольевич занимался программированием, отвечал за корректность кода.

План по выполнению работы по части исправления кода:

1. Просмотр кода.
2. Debug кода.
3. Push кода в GitHub.
   1. Стек технологий

Для реализации десктопного приложения были выбраны следующие технологии:

1. Языки программирования:
   * Python: Основной язык для разработки приложения, обладающий богатой экосистемой библиотек для обработки текста.
2. Библиотеки:
   * Hugging Face Transformers: Библиотека, предоставляющая доступ к предобученным моделям для семантического анализа.
   * Tkinter: Библиотека для создания графического пользовательского интерфейса (GUI).
3. API:



2.ПРАКТИЧЕСКАЯ ЧАСТЬ

2.1. Архитектура ПО

Фрагмент кода представляет собой пример простого графического интерфейса (GUI) на основе библиотеки **PyQt5** для выполнения семантического анализа текста с использованием предобученной модели из библиотеки **transformers**. Давайте разберем основные части кода и рассмотрим их взаимосвязи, а также общую архитектуру программы.

**Основные компоненты и взаимосвязи**

1. **Импорт необходимых библиотек**:

import sys

from PyQt5.QtWidgets import QApplication, QWidget, QPushButton, QLabel, QVBoxLayout, QFileDialog, QTextEdit

from transformers import pipeline

 Здесь импортируются необходимые классы и функции из библиотеки **PyQt5**, такие как QApplication, QWidget и другие виджеты, которые будут использоваться для построения GUI. Из библиотеки **transformers** импортируется функция pipeline, позволяющая легко загружать и использовать предобученные модели для различных NLP-задач.

 **Класс SemanticAnalysisApp**:

Этот класс наследуется от класса QWidget, который является базовым классом для всех виджетов в Qt. Внутри конструктора (\_\_init\_\_) происходит инициализация основных компонентов приложения:

* Загрузка предобученной модели DeepPavlov/rubert-base-cased-conversential через функцию pipeline. Эта модель будет использоваться для классификации текста на категории "Позитивный", "Негативный" и "Нейтральный".

self.model = pipeline("zero-shot-classification", model="DeepPavlov/rubert-base-cased-conversational")

Создание и настройка основного окна приложения:

self.setWindowTitle('Семантический Анализ')

self.resize(500, 400)

*  Эти строки задают заголовок окна и его размер.

 **Виджеты**:

В приложении используются следующие виджеты:

* Кнопка загрузки файла (QPushButton), которая вызывает метод open\_file\_dialog.
* Поле ввода текста (QTextEdit), куда загружается содержимое файла.
* Метка результата (QLabel), отображающая результат анализа текста.

self.load\_button = QPushButton('Загрузить Текст', self)

self.load\_button.clicked.connect(self.open\_file\_dialog)

self.text\_edit = QTextEdit()

self.result\_label = QLabel('')

**Макетирование виджетов**:

Все виджеты добавляются в вертикальную компоновку (QVBoxLayout), что позволяет расположить их друг под другом в окне приложения.

layout = QVBoxLayout()

layout.addWidget(self.load\_button)

layout.addWidget(self.text\_edit)

layout.addWidget(self.result\_label)

self.setLayout(layout)

**Метод open\_file\_dialog**:

Этот метод вызывается при нажатии кнопки "Загрузить Текст". Он открывает диалоговое окно выбора файла и считывает содержимое выбранного текстового файла.

def open\_file\_dialog(self):

file\_name, \_ = QFileDialog.getOpenFileName(self, 'Открыть Файл', '', 'Текстовые файлы (\*.txt)')

if file\_name:

with open(file\_name, 'r', encoding='utf-8') as f:

text = f.read()

self.analyze\_text(text)

 Если файл выбран успешно, его содержимое передается в метод analyze\_text.

 **Метод analyze\_text**:

Этот метод получает текст и проводит его семантический анализ с помощью предварительно загруженной модели. Результат выводится в метке self.result\_label.

def analyze\_text(self, text):

# Проведение семантического анализа

result = self.model(text, ["Позитивный", "Негативный", "Нейтральный"])

self.result\_label.setText(f'Результат: {result}')

**Запуск приложения**:

Основной блок кода создает экземпляр приложения QApplication, экземпляра класса SemanticAnalysisApp и запускает цикл обработки событий.

if \_\_name\_\_ == '\_\_main\_\_':

app = QApplication(sys.argv)

window = SemanticAnalysisApp()

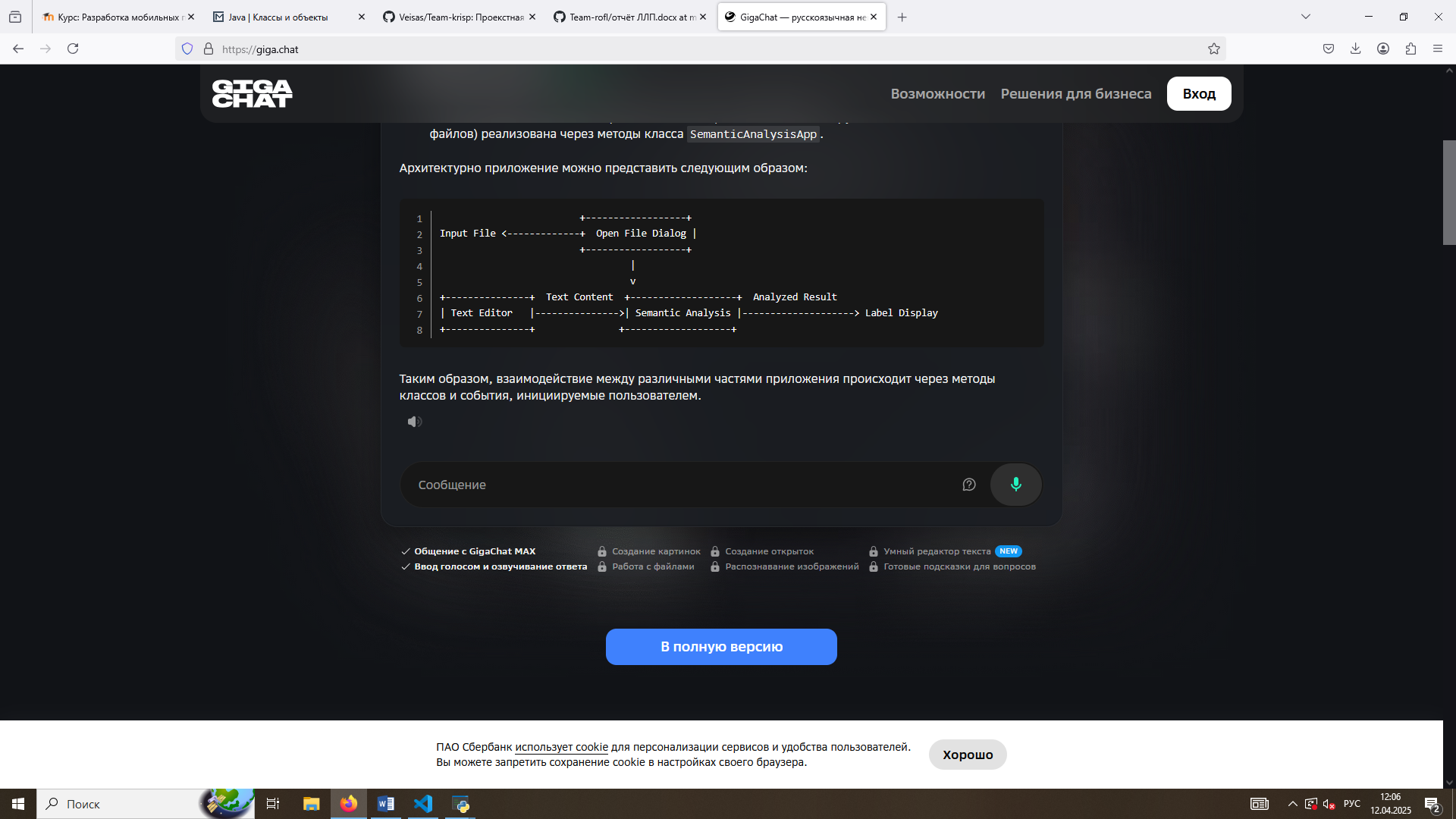
window.show()

sys.exit(app.exec\_())

**Архитектура ПО**

Приложение построено на принципах модульности и повторного использования компонентов. Основные компоненты включают:

1. **Модель NLP**: Предобученная модель DeepPavlov/rubert-base-cased-conversational, используемая для семантической классификации текста.
2. **Графический интерфейс**: Создан на основе библиотеки **PyQt5**, включает в себя основные элементы управления (кнопки, поля ввода, метки).
3. **Логика взаимодействия**: Обработка событий (нажатие кнопок, загрузка файлов) реализована через методы класса SemanticAnalysisApp.



Таким образом, взаимодействие между различными частями приложения происходит через методы классов и события, инициируемые пользователем.

2.2.Разработка проекта по ролям

Процесс разработки проекта можно разделить на несколько этапов, где каждый участник берет на себя конкретные задачи. Рассмотрим этапы разработки и распределение ролей по каждому этапу.

**Этап 1: Подготовка и планирование**

**Участники:**

* **Динганорбоев Эрдэм Булатович** (ответственный за теорию и сбор материала)
* **Лебедева Лира Петровна** (ответственная за отчетность)

**Задачи:**

1. **Динганорбоев Эрдэм Булатович**:
   * Определяет ключевые термины и концепции, важные для проекта.
   * Исследует доступные источники информации, собирает материалы и составляет библиографию.
   * Формирует общий план работы над проектом, включая цели, задачи и сроки выполнения.
2. **Лебедева Лира Петровна**:
   * Создает структуру отчета, разбивая его на главы и разделы.
   * Заполняет черновой вариант отчета основными идеями и концепциями, полученными от Динанорбоева Эрдэма Булатова.

**Этап 2: Проектирование и архитектура**

**Участники:**

* **Иннокентьев Влад Александрович** (программист, ответственный за основу и UI)
* **Николаев Аслан Анатольевич** (программист, ответственный за качество кода)

**Задачи:**

1. **Иннокентьев Влад Александрович**:
   * Работает над созданием общей архитектуры проекта.
   * Проектирует основные модули системы, определяет интерфейсы взаимодействия между ними.
   * Разрабатывает макет пользовательского интерфейса (UI).
2. **Николаев Аслан Анатольевич**:
   * Проверяет предложенную архитектуру на соответствие стандартам проектирования и возможностям реализации.
   * Оценивает возможные риски и предлагает улучшения.

**Этап 3: Реализация**

**Участники:**

* **Иннокентьев Влад Александрович** (основное программирование)
* **Николаев Аслан Анатольевич** (отладка и поддержка кода)

**Задачи:**

1. **Иннокентьев Влад Александрович**:
   * Начинает реализацию основной логики программы согласно спроектированной архитектуре.
   * Интегрирует библиотеку Hugging Face для обработки естественного языка.
   * Реализует функциональность пользовательского интерфейса.
2. **Николаев Аслан Анатольевич**:
   * Проводит регулярные проверки кода на наличие ошибок и несоответствий.
   * Занимается отладкой кода, устраняет баги и оптимизирует производительность

**Этап 4: Тестирование и интеграция**

**Участники:**

* **Николаев Аслан Анатольевич** (тестирование и деплой)
* **Все участники** (интеграция результатов)

**Задачи:**

1. **Николаев Аслан Анатольевич**:
   * Выполняет комплексные тесты программы на разных этапах развития.
   * Готовит проект к финальной интеграции и деплою.
   * Отправляет готовый код в репозиторий GitHub.
2. **Все участники**:
   * Совместно интегрируют результаты своих работ: теория, отчетность, программная часть.
   * Проводят общее тестирование интегрированного решения.

**Этап 5: Документация и завершение**

**Участники:**

* **Лебедева Лира Петровна** (документирование)
* **Все участники** (подведение итогов)

**Задачи:**

1. **Лебедева Лира Петровна**:
   * Завершает оформление отчета, учитывая все внесенные изменения и дополнения.
   * Добавляет в отчет документацию по использованию разработанного ПО.
2. **Все участники**:
   * Защищают проект перед руководством или заказчиками.
   * Оформляют заключительные отчеты и документы.

2.3.Контроль выполнения плана

Проект был направлен на разработку программного обеспечения для семантического анализа текста с использованием предобученной модели из библиотеки *Hugging Face*. Команда состояла из четырех человек, каждый из которых имел свою зону ответственности. Были определены четкие роли и этапы выполнения работы.

**Этапы выполнения**

1. **Подготовка и планирование**
   * Участниками была проведена работа по определению целей и задач проекта, составлению планов работы и созданию структуры отчета.
   * Динганорбоев Эрдэм Булатович разработал план работы и начал поиск необходимой информации.
   * Лебедева Лира Петровна создала структуру отчета и начала его написание.
2. **Проектирование и архитектура**
   * Иннокентьев Влад Александрович приступил к разработке основы проекта и созданию пользовательского интерфейса.
   * Николаев Аслан Анатольевич провел проверку предлагаемой архитектуры и предложил улучшения.
3. **Реализация**
   * Иннокентьев Влад Александрович продолжил работу над реализацией основной логики программы и интеграцией библиотеки *Hugging Face*.
   * Николаев Аслан Анатольевич активно участвовал в процессе отладки и устранения ошибок.
4. **Тестирование и интеграция**
   * Николаев Аслан Анатольевич выполнил тестирование программы и подготовил её к финальному деплою.
   * Вся команда участвовала в интеграции результатов и проведении общего тестирования.
5. **Документация и завершение**
   * Лебедева Лира Петровна завершила оформление отчета, добавив в него документацию по использованию разработанного ПО.
   * Все участники защитили проект перед руководством и подготовили заключительную документацию.

**Анализ текущего состояния**

На данный момент проект находится на стадии завершения. Большинство запланированных задач выполнено, однако остаются некоторые нерешенные проблемы и задачи, которые требуют дополнительного внимания.

* **Проблемы и препятствия**
  + Задержки в процессе разработки возникли из-за сложностей с интеграцией библиотеки *Hugging Face* и необходимостью внесения изменений в архитектуру проекта.
  + Некоторое время было потрачено на устранение ошибок, связанных с обработкой больших объемов текста.
* **Успешность выполнения задач**
  + Динганорбоев Эрдэм Булатович успешно завершил сбор и обработку необходимой информации.
  + Лебедева Лира Петровна выполнила значительную часть работы по подготовке отчета, хотя остается необходимость внести последние правки.
  + Иннокентьев Влад Александрович справился с разработкой основы проекта и интерфейса, но требует дополнительной помощи в завершении некоторых модулей.
  + Николаев Аслан Анатольевич продемонстрировал высокий уровень профессионализма в отладке и тестировании кода, но предстоит еще ряд тестов для полной уверенности в стабильности программы.

**Рекомендации и дальнейшие шаги**

* Необходимо завершить интеграцию оставшихся модулей и провести дополнительные тесты.
* Следует уделить внимание документированию кода и инструкций по эксплуатации программы.
* Рекомендуется пересмотреть процессы коммуникации внутри команды для предотвращения задержек в будущем.

**Заключение**

Проект достиг значительных успехов благодаря совместным усилиям всей команды. Несмотря на возникающие трудности, большинство задач было выполнено в срок. Остаются незначительные доработки, которые планируется завершить в ближайшее время.

ЗАКЛЮЧЕНИЕ

Создание десктопного приложения для семантического анализа текста на русском языке с использованием библиотеки Hugging Face Transformers представляет собой эффективный способ автоматизации анализа текстовой информации. Использование современных методов обработки естественного языка позволяет значительно улучшить качество анализа и расширить возможности взаимодействия с пользователями. Разработка такого приложения может быть полезной в различных областях, включая маркетинг, социологические исследования и автоматизацию клиентского обслуживания.