МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева»

(Самарский университет)

Институт информатики и кибернетики Кафедра программных систем

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

**«ВЕБ-ПРИЛОЖЕНИЕ ПРОГНОЗИРОВАНИЯ СТОИМОСТИ ЛЕГКОВОГО АВТОМОБИЛЯ»**

по направлению подготовки 02.03.02 Фундаментальная информатика и информационные технологии

Обучающийся Д.О. Колбанов

(подпись, дата)

Руководитель ВКР

доцент кафедры программных систем,

к.т.н., доцент О.А. Гордеева

(подпись, дата)

Нормоконтролер О.А. Гордеева

(подпись, дата)

Самара 2025

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«Самарский национальный исследовательский университет имени академика С.П. Королева»

Кафедра программных систем

УТВЕРЖДАЮ

Заведующий кафедрой

С.В. Востокин

« » 2024 г.

**задание на выпускную квалификационную работу (ВКР)**

обучающемуся Колбанову Дмитрию Олеговичу группы 6401-020302D

1. Тема работы: Веб-приложение прогнозирования стоимости легкового автомобиля Исходные данные
2. Перечень вопросов, подлежащих разработке в ВКР:
   1. Провести анализ предметной области
   2. Сделать обзор систем-аналогов в области прогнозирования стоимости автомобилей
   3. Собрать данные для обучения моделей машинного обучения
   4. Провести сравнительный анализ примененных алгоритмов прогнозирования стоимости легкового автомобиля
   5. Разработать и реализовать информационное и программное обеспечение
   6. Провести тестирование и отладку разработанного веб-приложения
3. Дата выдачи задания: «29» января 2024г.
4. Срок представления на кафедру законченной ВКР: «01» июня 2024г.

Руководитель ВКР

доцент кафедры программных систем**,**

к.т.н., доцент О.А. Гордеева

« 29 » 01 2024 г.

Задание принял к исполнению Д.О. Колбанов

« 29 » 01 2024 г.

РЕФЕРАТ

Пояснительная записка 64 с, 28 рисунков, 13 таблиц, 37 источников,

2 приложения.

Графическая часть: 22 слайда презентации PowerPoint.

СЕМАНТИЧЕСКИЙ АНАЛИЗ ТЕКСТА, КОМПЬЮТЕРНАЯ ЛИНГВИСТИКА, ЧАСТОТНО-СЕМАНТИЧЕСКИЙ АНАЛИЗ, ЛАТЕНТНО- СЕМАНТИЧЕСКИЙ АНАЛИЗ, СИНТАКСИКО-СЕМАНТИЧЕСКИЙ АНАЛИЗ, СЕМАНТИЧЕСКОЕ ЯДРО

Цель работы – разработать веб-приложение для прогнозирования стоимости легкового автомобиля с использованием нескольких алгоритмов машинного обучения и провести сравнительный анализ их эффективности.

В процессе работы были разработаны алгоритмы и соответствующая программа, позволяющая пользователю получить прогноз рыночной стоимости легкового автомобиля на основе входных параметров. Система выполняет предобработку данных, рассчитывает стоимость, предоставляет результаты в текстовом виде.

Клиентская часть системы разработана на языке JavaScript с использованием библиотеки Vue.js. Серверная часть системы разработана на языке Python с использованием фреймворка Flask. Доступ к данным осуществляется с помощью СУБД SQLite.

СОДЕРЖАНИЕ

[Введение 6](#_bookmark0)

1. [Описание и анализ предметной области 8](#_bookmark1)
   1. [Основные понятия и определения 8](#_bookmark2)
   2. [Методы семантического анализа текста 8](#_bookmark3)
      1. [Метод семантических шаблонов 9](#_bookmark4)
      2. [Частотно-семантический алгоритм 10](#_bookmark5)
      3. [Стемминг 10](#_bookmark6)
      4. [Латентно-семантический анализ 12](#_bookmark7)
   3. [Актуальность задачи 14](#_bookmark8)
   4. [Описание систем-аналогов 14](#_bookmark9)
      1. [Сервис ADVEGO 14](#_bookmark10)
      2. [Сервис miratext 15](#_bookmark11)
      3. [Сервис istio 17](#_bookmark12)
      4. [Конкурентный анализ систем-аналогов 18](#_bookmark13)
   5. [Описание автоматизируемого процесса 20](#_bookmark14)
   6. [Постановка задачи 21](#_bookmark15)
   7. [Выводы по главе 22](#_bookmark16)
2. [Проектирование системы 23](#_bookmark17)
   1. [Выбор и обоснование архитектуры системы 23](#_bookmark18)
   2. [Проект системы 24](#_bookmark19)
      1. [Структурная схема системы 24](#_bookmark20)
      2. [Диаграмма вариантов использования 26](#_bookmark21)
      3. [Диаграмма деятельности 28](#_bookmark22)
      4. [Диаграмма последовательности 29](#_bookmark23)
      5. [Диаграмма классов 30](#_bookmark24)
      6. [Логическая модель данных системы 32](#_bookmark25)
   3. [Выбор и обоснование средств реализации 34](#_bookmark26)
      1. [Выбор языка программирования 34](#_bookmark27)
      2. [Выбор операционной системы 35](#_bookmark28)
      3. [Выбор среды разработки 35](#_bookmark29)
      4. [Выбор СУБД 36](#_bookmark30)
   4. [Выводы по главе 36](#_bookmark31)
3. [Реализация системы 37](#_bookmark32)
   1. [Физическая модель данных 37](#_bookmark33)
   2. [Диаграммы реализации 38](#_bookmark34)
      1. [Диаграмма развертывания 38](#_bookmark35)
   3. [Описание интерфейса пользователя 39](#_bookmark36)
      1. [Авторизация пользователя в системе 39](#_bookmark37)
      2. [Ввод анализируемого текста 40](#_bookmark38)
      3. [Результаты анализа 41](#_bookmark39)
      4. [Интерфейс администратора 42](#_bookmark40)
   4. [Описание проведенных исследований 43](#_bookmark41)
      1. [Предмет исследования 43](#_bookmark42)
      2. [Набор данных 43](#_bookmark43)
      3. [Результаты экспериментальных исследований 43](#_bookmark44)
      4. [Выводы по главе 48](#_bookmark45)

[Заключение 49](#_bookmark46)

[Список использованных источников 50](#_bookmark47)

[Приложение А Руководство пользователя 54](#_bookmark48)

[А.1 Назначение системы 54](#_bookmark49)

[А.2 Условия работы системы 54](#_bookmark50)

[А.3 Установка системы 54](#_bookmark51)

[А.4 Работа с системой 54](#_bookmark52)

[Приложение Б Код программы 57](#_bookmark53)

ВВЕДЕНИЕ

В настоящее время рынок подержанных и новых легковых автомобилей характеризуется высокой динамичностью и большим разнообразием моделей, что усложняет процесс объективной оценки их стоимости [1]. Одновременно с этим широкое распространение современных веб‑технологий и методов машинного обучения открывает новые возможности для автоматизации процесса прогнозирования цен, повышая прозрачность рынка и помогая как продавцам, так и покупателям принимать более обоснованные решения.

Прогнозирование стоимости автомобиля представляет собой задачу регрессионного анализа, где в качестве входных признаков используются технические и эксплуатационные характеристики (марка, модель, год выпуска, пробег, тип двигателя и др.). Традиционные методы ручной оценки требуют глубоких экспертных знаний и существенно зависят от субъективного опыта оценщика. Автоматизированная система на основе машинного обучения позволяет построить модель, учитывающую многомерность данных и выявляющую сложные взаимосвязи между признаками, что повышает точность прогноза и скорость обработки запросов.

Практическая значимость работы заключается в создании доступного и наглядного инструмента, который может быть использован автодилерами, сервисами частных объявлений и конечными пользователями для быстрой оценки рыночной стоимости автомобиля. Полученные результаты могут быть внедрены в информационно‑аналитические системы автокомпаний и онлайн‑площадок для повышения конкурентоспособности и доверия клиентов.

В процессе работы необходимо разработать алгоритмы и соответствующую программу, позволяющую пользователю получить прогноз рыночной стоимости легкового автомобиля на основе входных параметров. Система должна выполнять предобработку данных, рассчитывать стоимость, предоставлять результаты в текстовом виде.

1. Описание и анализ предметной области
   1. Основные понятия и определения

Прогнозирование рыночной стоимости автомобиля — процесс определения будущей или текущей стоимости транспортного средства на основе анализа исторических данных, рыночных тенденций и технических характеристик [2]. Прогноз строится с использованием математических моделей и алгоритмов машинного обучения.

Регрессионный анализ — это статистический метод, позволяющий исследовать связь переменных. Основная цель регрессионного анализа — предсказать значение одной переменной на основе одной или нескольких других переменных. При этом регрессионный анализ помогает выявить, насколько тесно эти переменные связаны и как одна из них влияет на другую. [3]. В задачах прогнозирования стоимости автомобилей регрессионный анализ позволяет установить связь между характеристиками автомобиля (например, пробег, год выпуска, марка) и его ценой.

Машинное обучение — это область знаний и исследований в области искусственного интеллекта, которая занимается разработкой алгоритмов и статистических моделей, которые могут аппроксимировать данные, обучаться на них, обобщать их на невидимые зависимости и, таким образом, выполнять задачи без явных инструкций [4].

Модель машинного обучения — это объект (хранящийся локально в файле), который был обучен для распознавания определенных типов шаблонов. Модель обучается на основе набора данных по предоставленному ей алгоритму, который она может использовать для анализа и обучения на основе этих данных [5].

Регрессия в машинном обучении относится к задаче обучения с учителем, при котором целью является прогнозирование непрерывного числового значения на основе одной или нескольких независимых характеристик [6]. Прогнозирование стоимости автомобиля является задачей регрессии.

* 1. Актуальность задачи

Задача семантического анализа является одной из самых сложных задач в компьютерной лингвистике, так как формализировать естественный язык непросто. При семантическом анализе важно учитывать такие аспекты, как культурные и социальные особенности носителя языка, структуру естественного языка. Семантический анализ текста имеет практическое применение в различных областях, таких как контекстная реклама, определение тональности текста, вопросно-ответные системы, информационный поиск.

Как было сказано выше, существует большое количество методов и алгоритмов, решающих данную задачу. В связи с этим, разработка автоматизированной системы семантического анализа и проведение исследований алгоритмов, с точки зрения точности и скорости работы является актуальной темой для исследования.

* 1. Описание систем-аналогов

В настоящее время существуют различные сервисы для проведения семантического анализа текста, предоставляющие различный функционал. Каждая система имеет свои достоинства и недостатки. Рассмотрим некоторые из них.

* + 1. Сервис ADVEGO

Сервис ADVEGO [12] позволяет провести семантический анализ текста бесплатно, используется для оптимизации статей и поиска семантических ядер в тексте. Анализ проводится на различных естественных языках, в том числе и на русском. Для анализа необходимо вводить текст вручную, считывание текстовой информации с файла отсутствует.

При семантическом анализе вычисляются следующие характеристики текста:

* + - * плотность ключевых слов, процент ключевых фраз;
      * частотность слов;
      * количество стоп-слов;
      * объем текста: количество символов с пробелами и без пробелов;
      * количество слов: уникальных, значимых, всего;
      * водность, процент воды;
      * тошноту текста, классическую и академическую;
      * количество грамматических ошибок.

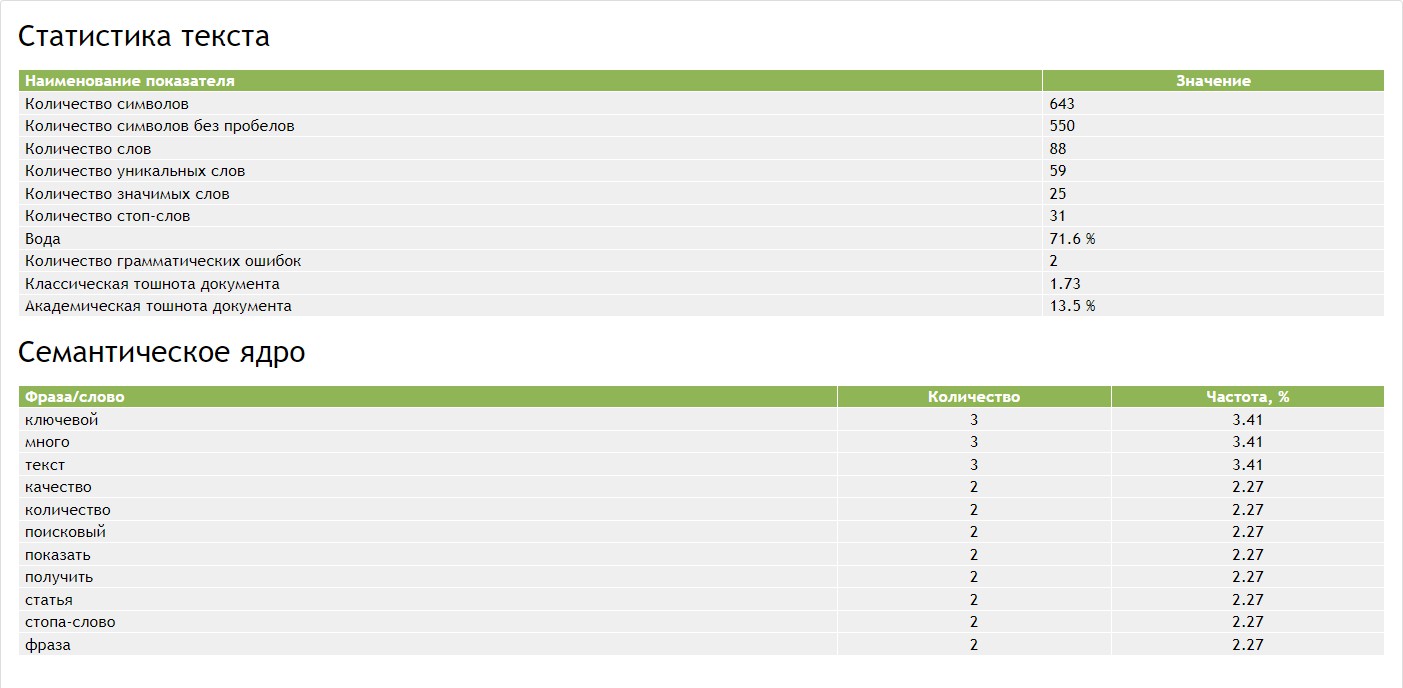
Результат семантического анализа, полученный данным сервисом представлен на рисунке 2.

Рисунок 2 – Пример результата семантического анализа сервисом ADVEGO

* + 1. Сервис miratext

Веб-сайт miratext [13] проводит полный семантический анализ текста, с некоторыми ограничениями. При бесплатном анализе пользователь добавляется в очередь, семантический анализ происходит по мере продвижения пользователей в ней.

Преимуществом данного сервиса можно выделить возможность анализа текста по ссылкам, а также парсинг текста с карточек-товаров интернет- магазинов. Помимо этого, miratext позволяет проводить анализ с сравнением двух текстов или ссылок. При использовании сервиса можно указать дополнительные параметры: не учитывать в анализе стоп-слова, числа, атрибуты заголовков.

Основным достоинством данного сервиса является визуализация результатов анализа в виде графиков и облаков слов.

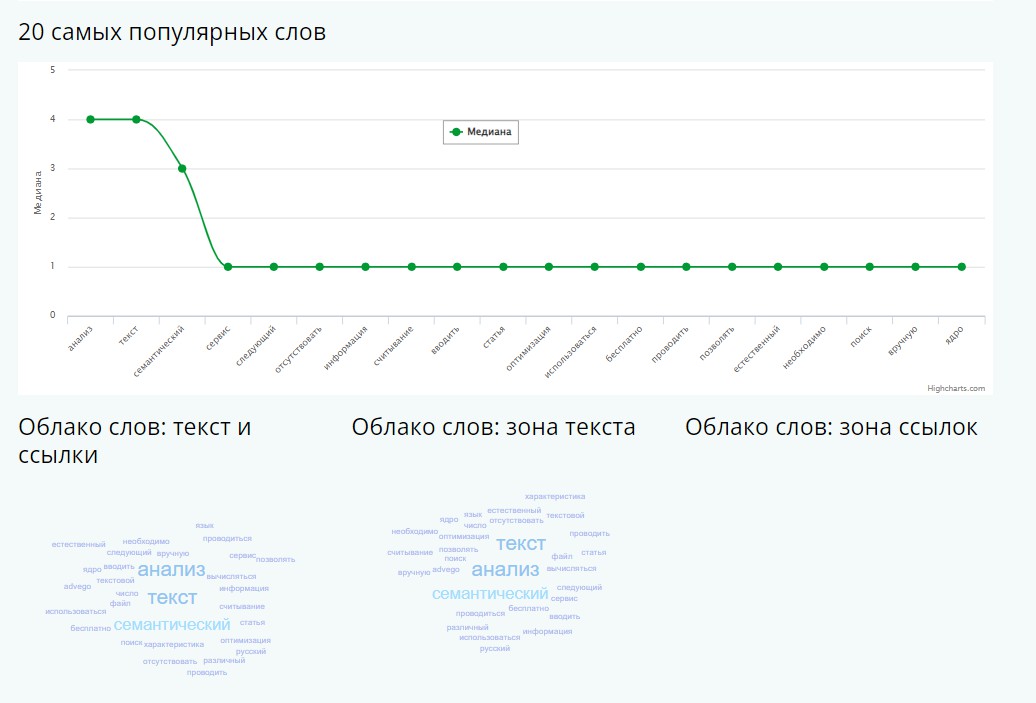
На рисунке 3 представлен пример семантического анализа данным веб- сайтом.

Рисунок 3 – Пример результата семантического анализа сервисом miratext При анализе помимо выделения ядер текста подсчитывается общая

статистика: количество символов с пробелами и без, водянистость и тошнота текста, общее и уникальное количество слов, а также качество текста по закону Ципфа. Закон Ципфа означает, что если все слова в тексте отсортировать по убыванию частоты встречаемости слова в тексте, то частота слова n будет являться обратно пропорциональной его порядковому номеру. Иначе говоря, данная характеристика отражает естественность текста. Предполагается, что естественный язык подчиняется данному закону и при высокой частоте встречаемости одних и тех же слов в тексте, текст характеризуется как неестественный.

Еще одним преимуществом данного сервиса является возможность загрузки подробного отчета в виде XLS – файла.

* + 1. Сервис istio

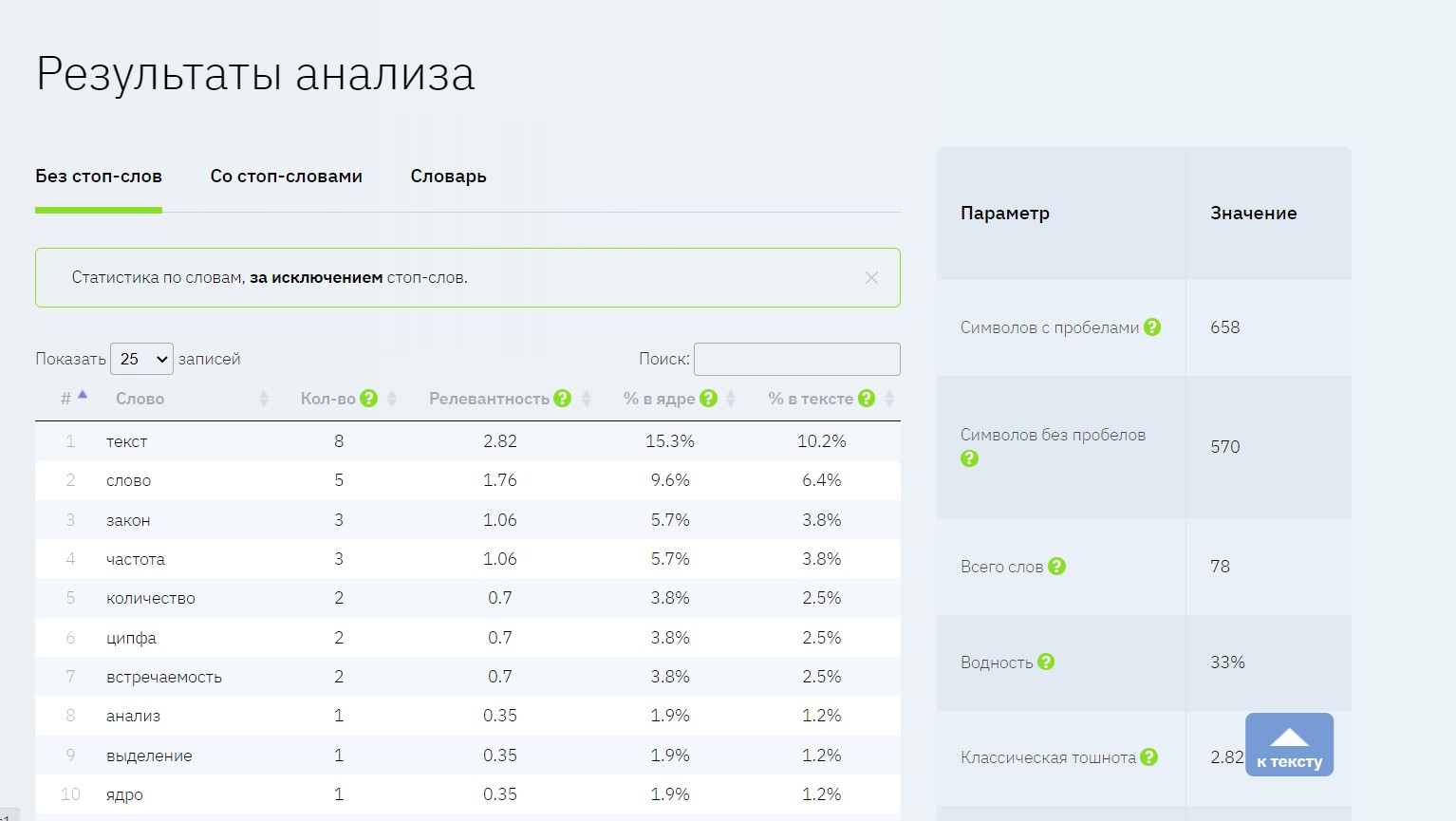
Сервис istio [14] позволяет полностью бесплатно провести семантический анализ текста без каких-либо ограничений. Поддерживает как ручной ввод текста, так и парсинг с веб-страниц по URL. При вводе текста присутствует возможность ручного ввода ключевых слов. Пример семантического анализа данным сервисом представлен на рисунке 4.

Рисунок 4 – Пример результата семантического анализа сервисом istio Сайт предоставляет подробный семантический анализ текста.

Вычисляются аналогичные характеристики текста сравнительно предыдущих сервисом. Однако, данный сервис обладает своими отличительными функциями.

При семантическом анализе istio формирует таблицы, в которых отражается слово и его подсчитанные характеристики. Просмотреть статистику можно как без стоп-слов, так и с ними. Доступен поиск по таблице, что является полезным функционалом при анализе больших текстов.

Главной отличительной чертой данного сервиса является формирования словаря, отсортированного по частоте встречаемости слов. Помимо этого, анализатор явно определяет тематику текста по выделенным ключевым словам.

* + 1. Конкурентный анализ систем-аналогов

Проведем сравнительный анализ для вышеописанных систем семантического анализа текста, выделив их основные отличительные особенности, см. таблицу 1.

Таблица 1 – Сравнительная характеристика систем-аналогов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Показатель | Система | | | Требования к  разрабатываемой системе |
| ADVEGO | miratext | istio |
| Бесплатное использование | + | − | + | + |
| Выделение семантического ядра | + | + | + | + |
| Определение частоты встречаемости слов | + | + | + | + |
| Визуализация статистики | − | + | + | + |
| Выгрузка отчета по анализу текста | − | + | − | + |
| Выбор метода семантического анализа | − | − | − | + |

18

* 1. Описание автоматизируемого процесса

В рамках выпускной квалификационной работы необходимо разработать автоматизируемую систему семантического анализа текста предоставляющую возможность выбора алгоритма семантического анализа. Данная возможность позволит оценить эффективность алгоритмов по таким параметрам как скорость анализа и его точность.

Наиболее подходящим подходом для реализации данной задачи является разработка web-приложения, предоставляющего удобный интерфейс и возможность визуализации результатов анализа.

К основным этапам разработки web-приложений относятся следующие: проектирование баз данных, разработка клиентской части, разработка серверной части.

Разработка клиентской части приложения включает в себя такие задачи как разработка интерфейса пользователя и его прототипов. Необходимо учитывать, что интерфейс пользователя является важной частью системы, так как именно через работу с интерфейсом пользователь может взаимодействовать с системой.

Разработка серверной части приложения включает в себя следующие подзадачи: реализация основной бизнес-логики системы, разработка способов взаимодействия сервера с клиентом и базой данных. В рамках автоматизируемого процесса семантического анализа текста, необходимо реализовать предварительную обработку данных, исключение стоп-слов (союзы, предлоги, служебные части речи), алгоритм семантического анализа, расчет точности и времени работы алгоритмов.

Для решения задачи семантического анализа текста и последующего исследования были выбраны частотно-семантический, латентно- семантический и синтаксико-семантический алгоритмы. Этап исследования включает в себя вычисление метрик, отражающих точность выделения тематики текста – precision, recall, а также вычисление времени работы

алгоритмов в зависимости от размера анализируемого текста.

20

* 1. Постановка задачи

Цель работы: во время выпускной квалификационной работы необходимо разработать автоматизированную систему семантического анализа текста.

Задачи:

* изучить основные понятия предметной области: семантический анализ текста, существующие алгоритмы семантического анализа;
* выполнить обзор систем-аналогов в области семантического анализа;
* разработать проект автоматизированной системы с использованием методологии UML;
* разработать программное обеспечение системы, произвести его тестирование и отладку;
* провести сравнительный анализ, разработанных алгоритмов семантического анализа текста.

Разрабатываемая автоматизированная система должна выполнять следующие функции:

1. серверная часть системы;
   * аутентификация и авторизация в системе;
   * регистрация пользователей;
   * разделение ролей (пользователь, администратор);
   * семантический анализ текста;
   * расчет статистики пользования системой;
2. клиентская часть системы;
   * выбор алгоритма семантического анализа;
   * просмотр результатов анализа;
   * визуализация данных анализа, в виде графиков;
   * выгрузка результатов анализа в виде файла;
   * просмотр статистики.
   1. Выводы по главе.

В данной главе был произведен анализ предметной области: изучены и описаны основные определения в области семантического анализа текста, актуальность исследования, рассмотрены существующие алгоритмы решения задачи семантического анализа текста, выделены достоинства и недостатки.

Произведен обзор существующих систем-аналогов, сформулирована постановка задачи ВКР и определены основные функции разрабатываемой системы.

1. Проектирование системы
   1. Выбор и обоснование архитектуры системы

Для программной реализации автоматизированной системы семантического анализа текста выбрана клиент-серверная архитектура, позволяющая передавать и обрабатывать только необходимые данные, тем самым уменьшая нагрузку на сеть.

Клиент-серверная архитектура – способ организации программного обеспечения, в которой функционал системы распределяется между клиентской и серверной частью [15]. Основными компонентами данной архитектуры является клиент, сервер и база данных.

Взаимодействие между клиентом и сервером осуществляется с помощью двухзвенной архитектуры. Клиент отправляет запросы напрямую серверу, сервер в свою очередь, при необходимости отправляет запрос в базу данных, получает обработанный результат и передает данные обратно клиенту. Двухзвенная архитектура подразумевает, что сервер обрабатывает, полученные запросы в полной мере, не прибегая к сторонним серверам и ресурсам для получения каких-либо данных. На рисунке 5 представлена схема двухзвенной клиент-серверной архитектуры.

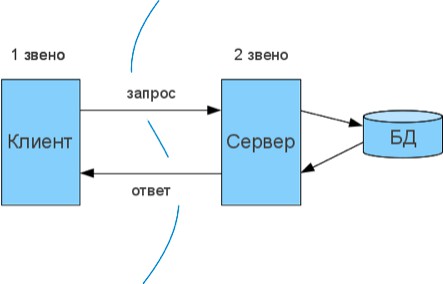


Рисунок 5 – Схема двухзвенной клиент-серверной архитектуры

Web-приложение – это клиент-серверное приложение, в котором взаимодействие с сервером происходит с помощью браузера [16]. Клиентская

часть отвечает за отрисовку пользовательского интерфейса, визуализацию данных, формы взаимодействия пользователя с системой, сервер отвечает за бизнес-логику системы, обработку данных, полученных с сервера и взаимодействие с базой данных. При таком распределении задач между клиентом и сервером выделяют понятие тонкого клиента, означающее, что вся основная логика работы системы реализована на сервере [17]. Реализация системы в виде web-приложения обоснована удобством использования таких систем, так как web-приложения могут быть доступны с любого устройства, имеющего доступ к сети, а также являются кроссплатформенными, в связи с тем, что выполняются в браузере.

Взаимодействие между клиентом и сервером осуществляется с помощью протокола HTTP (HyperText Transfer Protocol), чаще всего использующегося в веб-приложениях при клиент-серверной архитектуре. Протокол HTTP определяет способ взаимодействия компонентов программы и состоит из запросов и ответов [18]. Данный протокол не накладывает ограничений на формат передаваемых и получаемых данных, и является универсальным.

* 1. Проект системы
     1. Структурная схема системы

Структурный подход при разработке системы подразумевает разбиение системы на функциональные подсистемы, тем самым система представляет из себя совокупность отдельных компонентов, взаимодействующих между собой [19]. Таким образом, сохраняется целостность представления о системе, в которой все выделенные подсистемы связаны.

Структурная схема системы представляет из себя диаграмму, отражающую взаимосвязь компонентов системы для понимания принципов работы разрабатываемой системы.

На рисунке 6 приведена структурная схема разрабатываемой системы, в ее состав входят клиентская и серверная части, которые взаимодействуют между собой с помощью протокола HTTP.

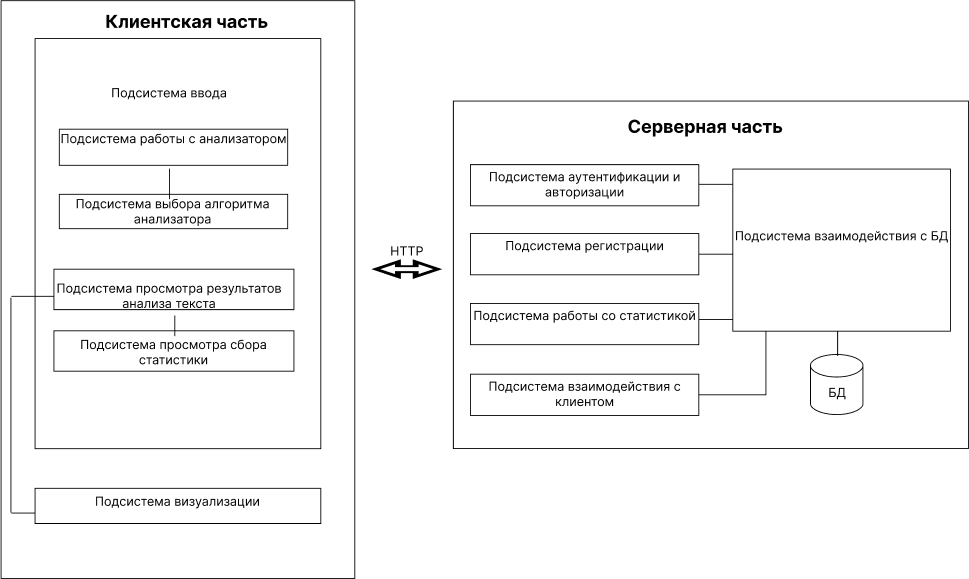


Рисунок 6 – Структурная схема системы

В состав клиентской части входят следующие подсистемы:

1. подсистема ввода;

− подсистема работы с анализатором, позволяющей вводить анализируемый текст и редактировать его;

− подсистема выбора алгоритма семантического анализа;

− подсистема сбора статистики, которая агрегирует статистику пользователей;

− подсистема просмотра результатов анализа текста, которая позволяет пользователю просмотреть подробный семантический отчет по тексту.

1. подсистема визуализации, которая отвечает за отображение данных, визуализирует результаты анализа и статистику работы с сервисом для администратора.

В состав серверной части входят следующие подсистемы:

1. подсистема регистрации, которая отвечает за регистрацию пользователей в системе;
2. подсистема аутентификации и авторизации, которая отвечает за аутентификацию и авторизацию каждого зарегистрированного пользователя;
3. подсистема работы со статистикой, которая отвечает за сбор статистики по количеству использования системы и определения популярных тематик текстов;
4. подсистема взаимодействия с БД, которая отвечает за добавление, изменение или удаление данных о пользователях, статистике в базе данных.
   * 1. Диаграмма вариантов использования

Диаграмма вариантов использования – диаграмма, отражающая взаимодействие между актерами и вариантами использования, позволяющая описать концептуальную модель системы [20].

Данная диаграмма используется для описания функционала и поведения системы, не погружаясь в особенности реализации функционала.

В качестве актеров выступает некоторое множество ролей, взаимодействующих с системой различными вариантами использования.

Вариант использования представляет из себя спецификацию последовательности действий, которые система выполняет при взаимодействии с актерами, таким образом выделяется набор действий, которые совершает систем, взаимодействуя с актером.

На рисунке 7 приведена диаграмма вариантов использования для пользователя в разрабатываемой системе.

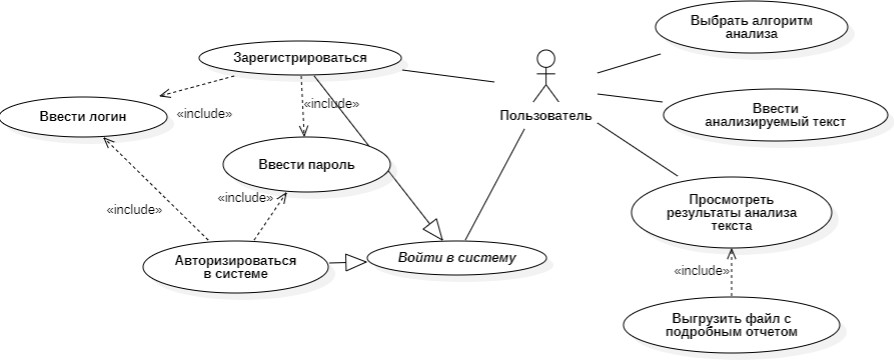


Рисунок 7 – Диаграмма вариантов использования для пользователя

Любой пользователь (включая администратора) для работы с системой должен или зарегистрироваться, или войти в систему, указав логин и пароль.

Для проведения семантического анализа текста пользователь должен ввести текст в специальную форму, при необходимости отредактировать его, а также выбрать один из представленных алгоритмов анализа. После обработки текста пользователь может просмотреть и оценить работу анализатора, просмотрев вычисленные характеристики текста (встречаемость слов, общее количество слов/символов, водность, тематику текста). Также пользователю доступен визуализированный просмотр работы анализатора, отображающий данные в виде диаграмм и графиков. Помимо этого, пользователь может выгрузить результат обработки в виде текстового файла.

На рисунке 8 представлена диаграмма вариантов использования для администратора системы.

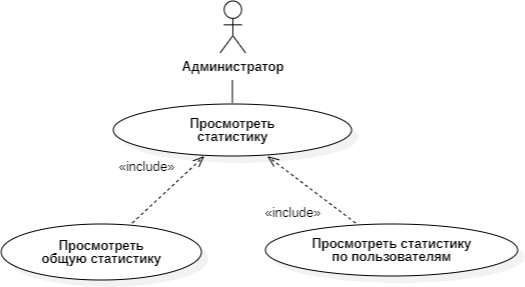


Рисунок 8 – Диаграмма вариантов использования для администратора При входе в систему администратор может просматривать статистику,

отражающую работу системы. Администратор может просмотреть общую статистику по использованию системы, показывающую количество обращений, тематики текстов, по которым производились запросы. Также

администратору доступна статистика по отдельным пользователям. Доступна визуализация статистических данных в виде графиков.

* + 1. Диаграмма деятельности

Диаграмма деятельности – это диаграмма, разработанная в виде блок- схем, отображающая последовательность действий использования системы [21]. На диаграмме деятельности изображаются действия и состояния, которые связаны потоком управления. На диаграмме отображаются узлы начала действий и окончание всего процесса. Диаграмма деятельности позволяет графически отобразить порядок действий работы со системой, что дает возможность выявить проблемные места системы.

На рисунке 9 представлена диаграмма деятельности системы.

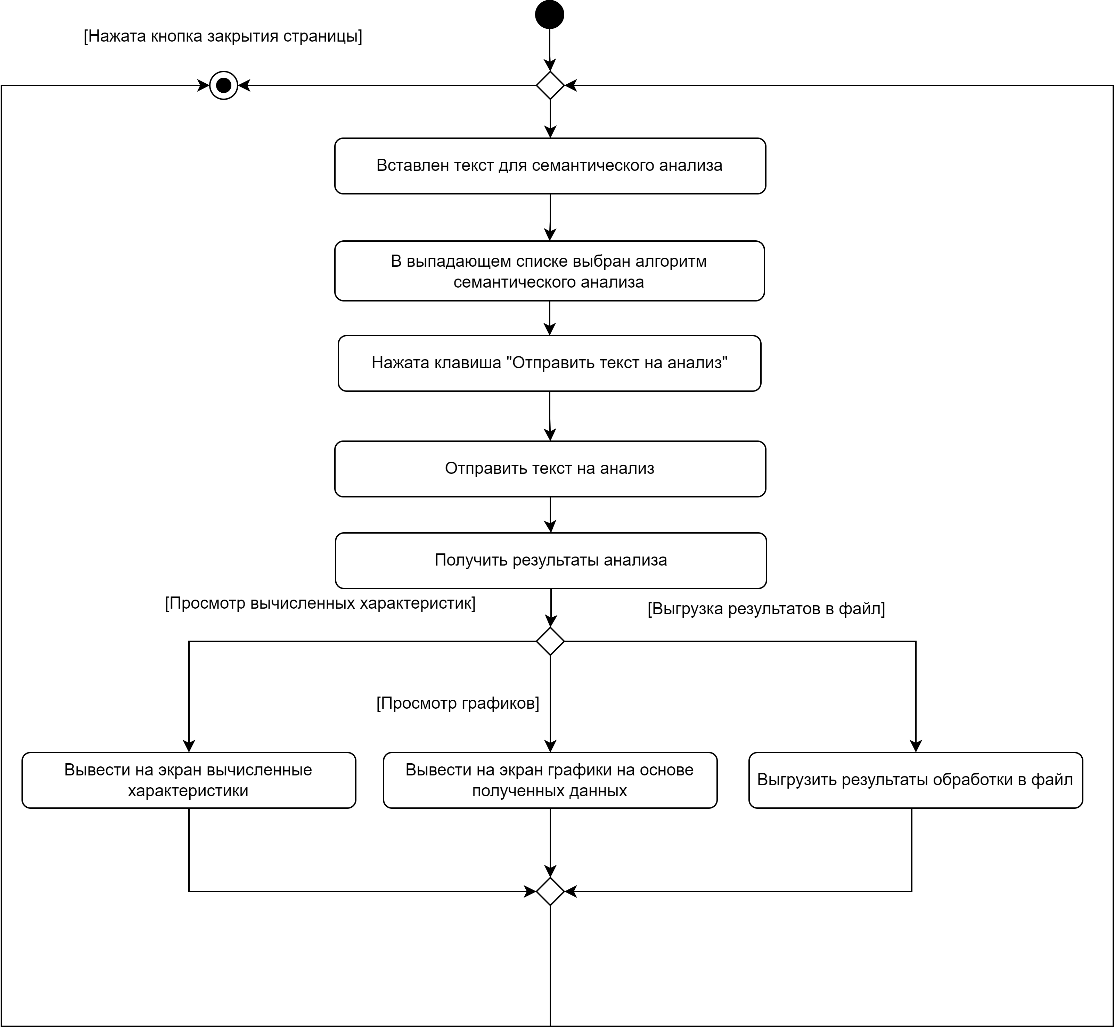


Рисунок 9 – Диаграмма деятельности системы

После ввода анализируемого текста и выбора алгоритма семантического анализа, клиентская часть системы отправляет текст на анализ серверной части системы, та в свою очередь предоставляет клиентской части полученные

результаты анализа текста. Система отображает результат анализа на странице, а также визуализирует их в виде графиков. При выгрузке результатов в виде файла, система создает файл формата docx, в который записывает анализируемый текст, а также полученные результаты семантического анализа.

* + 1. Диаграмма последовательности

Диаграмма последовательности – UML-диаграмма, отображающая жизненный цикл взаимодействия объектов и акторов на временной оси.

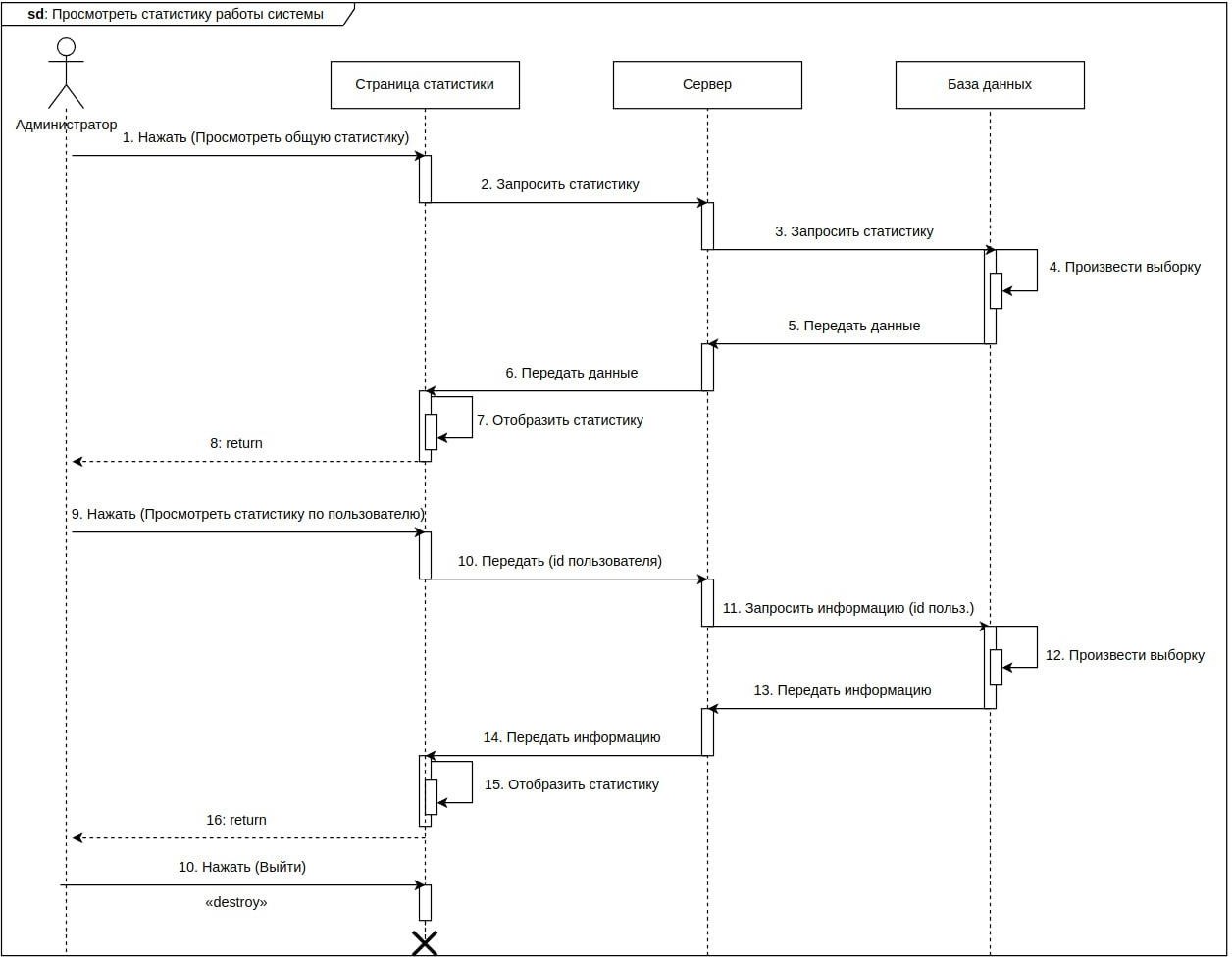
На рисунке 10 представлена диаграмма последовательности для варианта использования – просмотр статистики системы.

Рисунок 10 – Диаграмма последовательности для варианта использования

«Просмотр статистики системы»

Диаграмма последовательности состоит из следующих элементов: объектов, вертикальных «линий жизни», отображающих временной отрезок, прямоугольников, которые отображают деятельность объекта или исполнение им определенной функции (прямоугольники на пунктирной «линии жизни»), и стрелок, показывающих взаимодействия между объектами [22].

Диаграмма последовательности облегчает разработку автоматизированной системы, так как предоставляет информацию о взаимодействии элементов системы во времени, отражая последовательность проводимых операций.

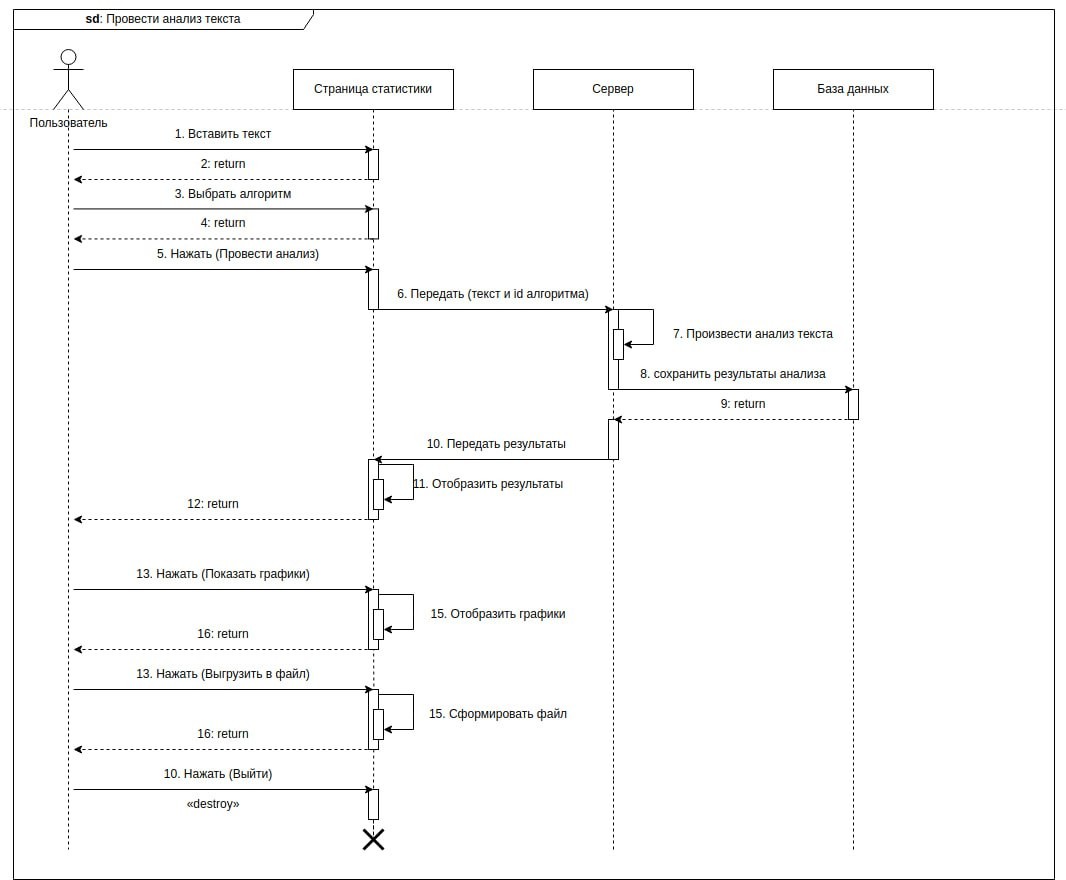
На рисунке 11 отображена диаграмма последовательности для варианта использования – анализ текста.

Рисунок 11 – Диаграмма последовательности для варианта использования

«Анализ текста»

* + 1. Диаграмма классов

Диаграмма классов – UML диаграмма, описывающая структуру системы, классы и их атрибуты, а также зависимости между классами. Целью разработки диаграммы классов является наглядное отображение компонентов системы и ее сложности [23].

На рисунке 12 представлена диаграмма классов.



Рисунок 12 – Диаграмма классов

В таблице 2 представлено описание классов системы.

Таблица 2 – Описание классов системы

|  |  |
| --- | --- |
| Название класса | Назначение |
| Модель роли | Предоставляет данные объекта «Роль» |
| Модель пользователя | Предоставляет данные объекта «Пользователь» и  реагирует на команды контроллера, изменяя своё состояние |
| Модель результатов анализа | Предоставляет данные объекта «Результат анализа» и реагирует на команды контроллера, изменяя своё  состояние |
| Модель статистики | Предоставляет данные объекта «Статистика» и  реагирует на команды контроллера, изменяя своё состояние |
| Сервис маршрутизации | Сопоставление внешнего запроса с соответствующим  контроллером |
| Контроллер роли | Предоставляет доступ к модели роли |
| Контроллер пользователя | Предоставляет доступ к модели пользователя |
| Контроллер результатов  анализа | Предоставляет доступ к модели результат анализа |
| Контроллер статистики | Предоставляет доступ к модели статистики |

* + 1. Логическая модель данных системы

Логическая информационная модель – это модель данных, которая отражает способ хранения данных в памяти ЭВМ [24]. При этом в модели названия её объектов указаны в удобном для чтения формате и могут отличаться от названий в физической модели. При построении данное модели используются следующие понятия.

Сущность – объект предметной области, обособленный от других подобных объектов каким-либо набором признаков. У каждого такого объекта имеются свойства, называемые атрибутами.

Атрибут – это определённое свойство сущности, которое выделяет сущность на фоне прочих сущностей. Каждый атрибут сущности имеет уникальное имя в рамках этой сущности.

Ключ сущности – это атрибут или несколько атрибутов, которые используются для идентификации экземпляра сущности. Ключи бывают простыми, когда для идентификации используется всего один атрибут или же сложные, когда используется несколько атрибутов.

Для разрабатываемой системы была создана логическая модель данных, которая приведена на рисунке 13. Данная модель создана с использованием языка UML.

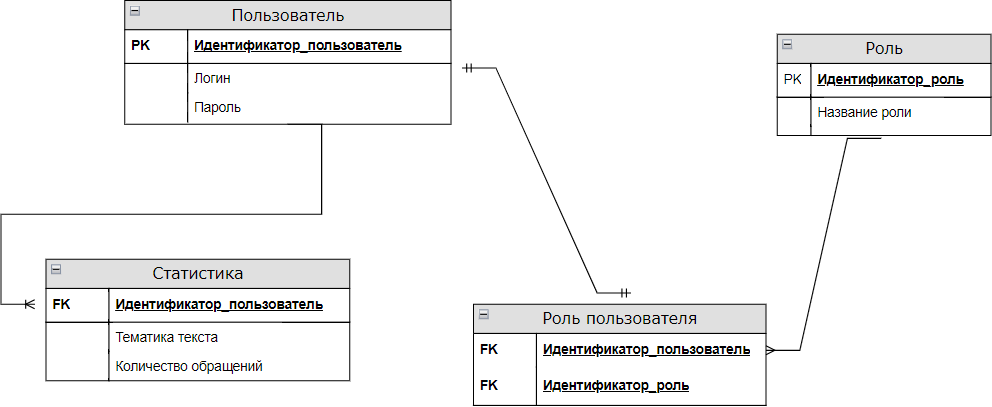


Рисунок 13 – Логическая модель системы

Описание объектов логической модели данных приведено в таблицах 3-6. Таблица 3 – Сущность «Пользователь»

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| Идентификатор\_пользователь | Целочисленный | Уникальный идентификатор роли |
| Логин | Символьный [50] | Имя, используемое при  идентификации пользователя |
| Пароль | Символьный [100] | Пароль пользователя, преобразованный в закодированную строку |

Таблица 4 – Сущность «Роль»

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| Идентификатор\_роль | Целочисленный | Уникальный  идентификатор роли |
| Название роли | Символьный [20] | Имя роли, используемое для определения  возможностей пользователя |

Таблица 5 – Сущность «Роль пользователя»

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| Идентификатор\_роль | Целочисленный | Внешний ключ, указывающий на сущность  «Роль» |
| Идентификатор\_пользователь | Целочисленный | Внешний ключ, указывающий на сущность  «Пользователь» |

Таблица 6 – Сущность «Статистика»

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| 1 | 2 | 3 |
| Идентификатор\_пользователь | Целочисленный | Внешний ключ, указывающий на сущность  «Пользователь» |

Продолжение таблицы 6

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Тематика текста | Символьный [20] | Тема текста, полученная  при анализе пользователем |
| Количество обращений | Целочисленный | Количество запросов пользования на данную  тему |

* 1. Выбор и обоснование средств реализации

Выбор программных средств для реализации программного обеспечения является важным этапом разработки системы. Выбор верных и подходящих технологий позволит ускорить и повысить качество разработки автоматизированной системы.

* + 1. Выбор языка программирования

Язык программирования – это набор определенных правил и действий, которые преобразуются в машинный код при исполнении [25].

Для разработки серверной части системы выбран объектно- ориентированный язык программирования Java, являющийся популярным решением для разработки серверов. Язык Java является кроссплатформенным и может быть исполнен на любой операционной системе, на которой установлена виртуальная машина JVM [26]. Для упрощения разработки серверной части выбран Spring Framework – фреймворк, предоставляющий огромный набор инструментов для описания сервисов и взаимодействия с базами данных [27]. Для разработки латентно-семантического и синтаксико- семантического алгоритма выбрана библиотека Stanford CoreNLP [28]. Данная библиотека предоставляет множество инструментов для обработки текста на естественном языке, разделение текста на части, выделение слов, определение частей речи, анализ структуры предложений, понимание семантики и другие возможности. Stanford CoreNLP поддерживает обработку текста на различных языках, в том числе и на русском языке.

Для разработки клиентской части приложения выбран язык программирования TypeScript – объектно-ориентированный язык

программирования, продолжение языка JavaScript, предоставляющий возможность статической типизации переменных, что делает код более читаемым и стабильным [29]. Выбрана библиотека React, предоставляющая обширный функционал для создания пользовательских интерфейсов [30]. С помощью React клиентская часть системы разбивается на небольшие компоненты, объединяющиеся в один пользовательский интерфейс, что позволяет создавать SPA – одностраничные приложения, в которых необходимый контент отображается на одной странице при определенных действиях пользователя.

* + 1. Выбор операционной системы

В качестве операционной системы выбрана ОС Windows, являющаяся самой популярной операционной системой в мире. ОС Windows используются как для домашнего использования, так и корпоративного. Windows имеет удобный пользовательский интерфейс, панель управления, а также встроенную систему защиты от вирусов [31]. За счёт своей популярности Windows поддерживает множество различных программ и инструментов, что делает данную систему удобной для разработки программного обеспечения.

* + 1. Выбор среды разработки

Интегрированная среда разработки (IDE) – комплекс программных средств, предоставляющий набор различных инструментов и возможностей для разработки и отладки кода [32]. IDE чаще всего состоит из редактора кода, средств отладки, встроенных компиляторов и плагинов, что позволяет ускорить и облегчить разработку кода.

Для разработки серверной части системы была выбрана среда разработки IntelliJ IDEA, позволяющая работать с языком программирования Java и его фреймворками. IntelliJ IDEA предоставляет широкий функционал для написания кода, среди его главных достоинств авто подсказки кода, интеграция с базами данных, поддержка системы контроля версий.

В качестве среды разработки для клиентской части приложения выбрана IDE WebStorm, ориентированная на разработку веб-приложений. WebStorm

поддерживает множество технологий для веб-разработки и не требует установки дополнительного программного обеспечения.

* + 1. Выбор СУБД

В качестве системы управления базами данных выбрана PostgreSQL.

PostgreSQL – объектно-реляционная система для управления базами данных, поддерживающая стандарты SQL. PostgreSQL обладает функционалом создания новых типов данных и различных функций, что делает ее расширяемой и масштабируемой СУБД [33]. Помимо стандартных SQL операций, PostgreSQL дает возможность работать с триггерами, транзакциями, представлениями и хранимыми процедурами. Для удобства сопровождения и управления базами данными PostgreSQL предоставляет платформу администрирования pgAdmin.

* 1. Выводы по главе

В данной главе было выполнено проектирование разрабатываемой автоматизируемой системы семантического анализа текста с использованием методологий структурного проектирования и UML. Была описана и выбрана архитектура системы, приведена структурная схема, а также создан UML- проект, включающий необходимые диаграммы. Представлена логическая модель данных системы и описан комплекс программных средств для реализации.

1. Реализация системы
   1. Физическая модель данных

Физическая модель данных представляет структуру хранения данных, в рамках конкретной СУБД. Физический уровень представления является последним уровнем, на котором определяются типы данных, размер и доступ к этим данным. Разработка физической модели данных происходит на основе логической модели данных.

На рисунке 14 представлена физическая модель данных системы.

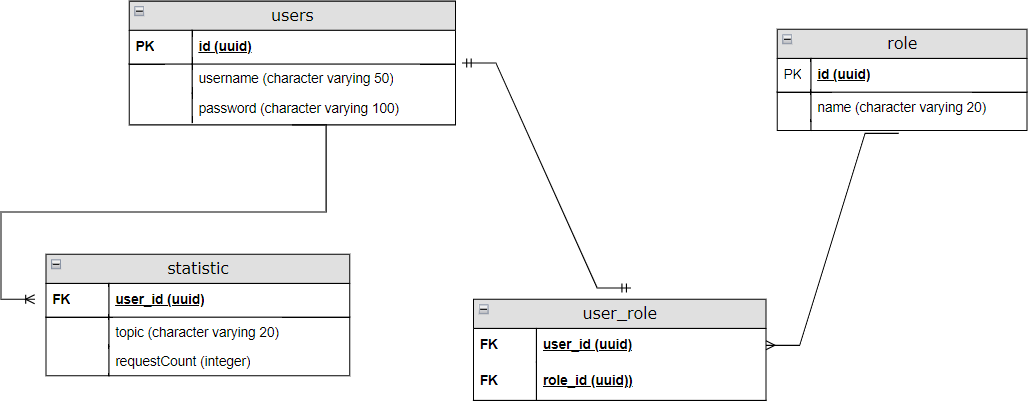


Рисунок 14 – Физическая модель данных

В таблицах 7-10 представлены описания сущностей физической модели данных.

Таблица 7 – Сущность «users»

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| id | uuid | Уникальный идентификатор  роли |
| username | character varying [50] | Имя, используемое при идентификации пользователя и его  взаимодействии с системой |
| password | character varying [100] | Пароль пользователя,  преобразованный в закодированную строку |

Таблица 8 – Сущность «role»

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| id | uuid | Уникальный  идентификатор роли |
| name | character varying [20] | Имя роли, используемое для определения  возможностей пользователя |

Таблица 9 – Сущность «user\_role»

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| role\_id | uuid | Внешний ключ, указывающий на сущность  «Роль» |
| user\_id | uuid | Внешний ключ, указывающий на сущность  «Пользователь» |

Таблица 10 – Сущность «statistic»

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| user\_id | uuid | Внешний ключ, указывающий на сущность  «Пользователь» |
| topic | character varying [20] | Тема текста, полученная  при анализе пользователем |
| requestCount | integer | Количество запросов пользования на данную  тему |

* 1. Диаграммы реализации

Диаграммы реализации используется для описания компонентов системы и их структуры, а также обозначения связей между ними.

* + 1. Диаграмма развертывания

Диаграмма развертывания представляет из себя диаграмму, отображающую узлы, компоненты системы и связь между ними [34].

Каждый узел может представлять из себя множество из нескольких узлов системы.

На сервере должна быть установлена ОС, а также приложения Spring и Node.js, которые взаимодействуют друг с другом. Также на сервере развертывается база данных с соответствующей СУБД (PostgreSQL). Помимо этого, на сервере хранятся файлы ресурсов и справки.

Для пользования системой, пользователю на своем ПК необходимо иметь ОС и браузер.

На рисунке 15 представлена диаграмма развертывания системы.

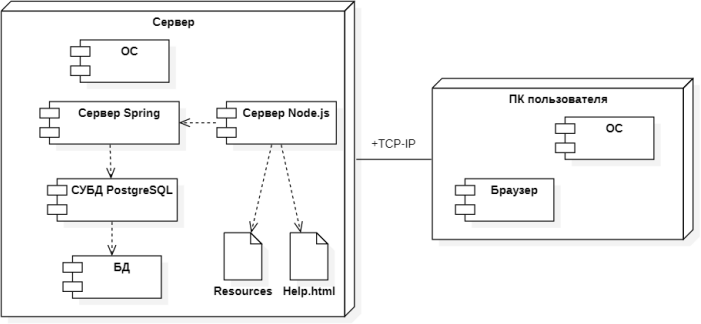


Рисунок 15 – Диаграмма развертывания

* 1. Описание интерфейса пользователя

Интерфейс пользователя важная часть программного обеспечения, связывающая пользователя с другими компонентами системы.

Для эффективной работы с системой пользовательский интерфейс должен быть удобным и интуитивно понятным.

Разработанная система является web-приложением, рассмотрим клиентские экранные формы взаимодействия с системой.

* + 1. Авторизация пользователя в системе

Для работы с системой пользователь должен пройти регистрацию. На рисунке 16 представлена форма регистрации в системе.

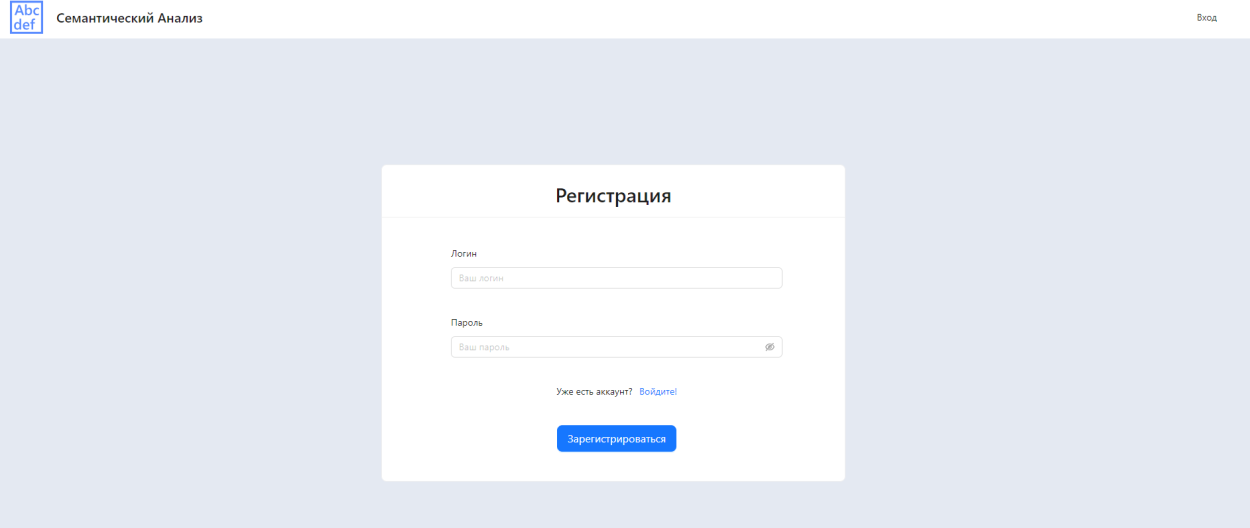


Рисунок 16 – Экранная форма регистрации

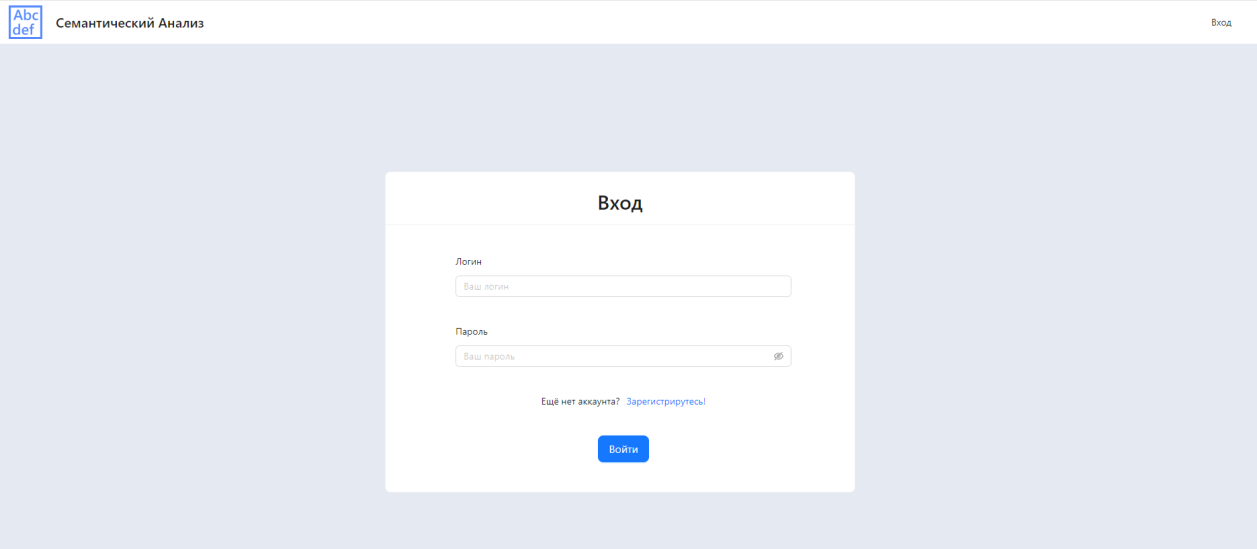
После регистрации в системе пользователя перенаправит на страницу входа в систему, представленную на рисунке 17.

Рисунок 17 – Экранная форма входа

Для входа в систему пользователю необходимо ввести свой логин и пароль, затем нажать на кнопку «Войти», если пользователь не зарегистрирован, есть возможно перейти на страницу регистрации по ссылке

«Зарегистрируйтесь».

* + 1. Ввод анализируемого текста

После входа в систему пользователь переходит на главную страницу, на которой представлена форма ввода текста для проведения семантического анализа. В форму ввода пользователю необходимо ввести анализируемый

текст, а также выбрать алгоритм семантического анализа в выпадающем списке.

Экранная форма представлена на рисунке 18.

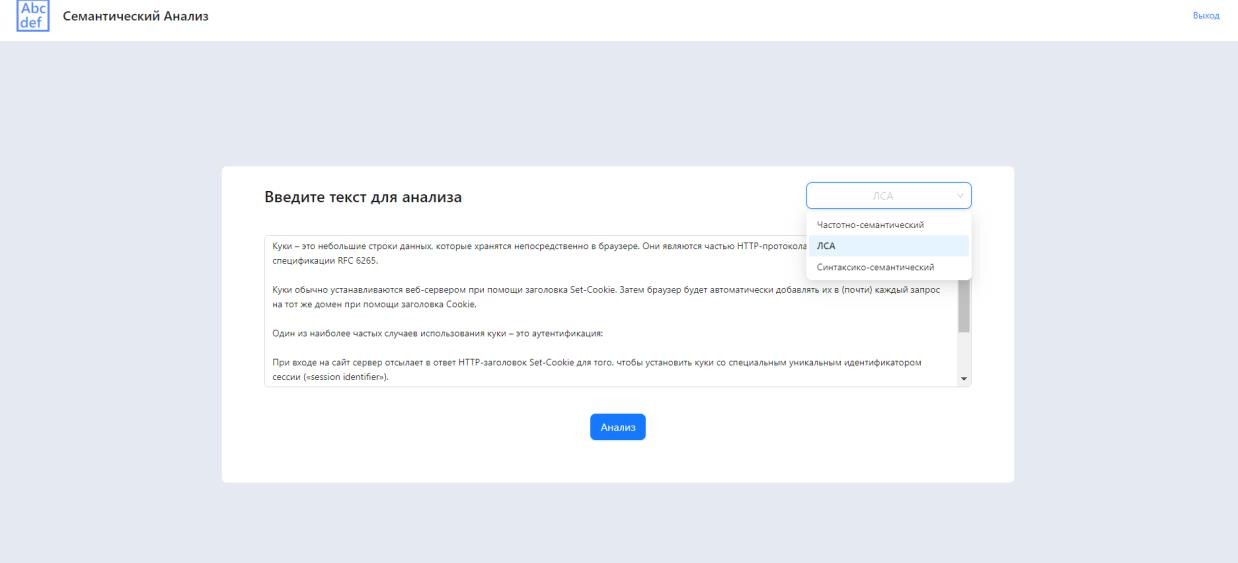


Рисунок 18 – Экранная форма анализатора

В шапке страницы находится кнопка «Выход» для выхода из системы. После ввода текста и выбора алгоритма анализа, необходимо нажать на кнопку

«Анализ».

* + 1. Результаты анализа

После нажатия на кнопку «Анализ» системой открывается страница с представленными результатами семантического анализа. На рисунке 19 представлена экранная форма результатов анализа.

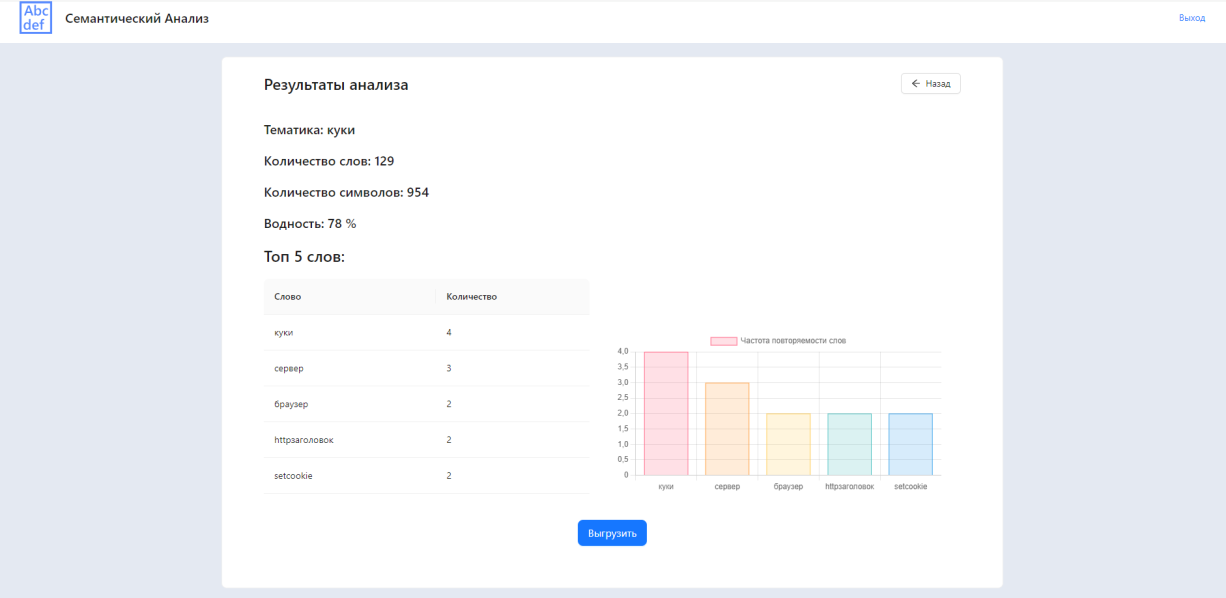


Рисунок 19 – Экранная форма результатов анализа текста

В таблице и графике отображаются пять самых повторяемых слов и частота их встречаемости. Для того чтобы получить результаты анализа в виде текстового файла формата docx, пользователю необходимо нажать на кнопку

«Выгрузить». Для возврата на страницу ввода текста нажать на кнопку

«Назад».

На рисунке 20 представлена структура файла с результатами анализа.

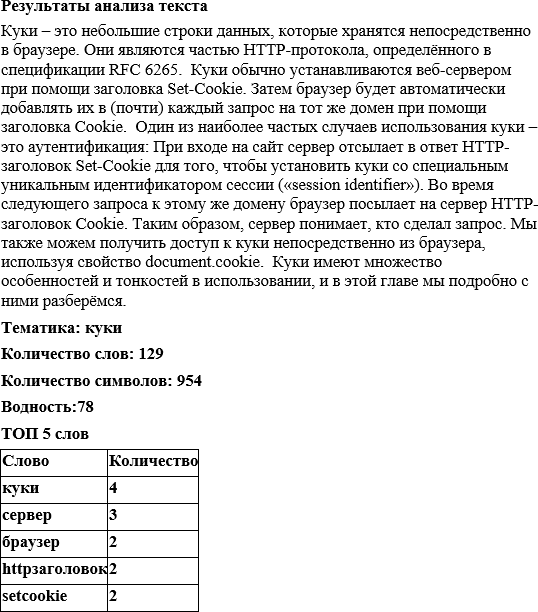


Рисунок 20 – Выгрузка результатов анализа

В файл сохраняется введенный анализируемый текст, а также вычисленные характеристики.

* + 1. Интерфейс администратора

После входа в систему под ролью «Администратор» открывается страница со статистикой.

На данной странице администратор может посмотреть количество запросов к системе, а также тематики текстов, на которые производилось больше всего запросов. Статистика доступна как в общем виде, так и по выбранным пользователям, рисунок 21.

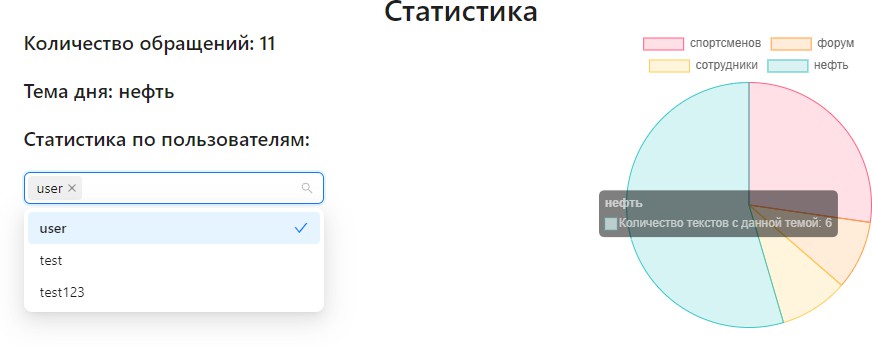


Рисунок 21 – Экранная форма администратора

* 1. Описание проведенных исследований
     1. Предмет исследования

В рамках выпускной квалификационной работы предметом исследования является проведение сравнительного анализа разработанных алгоритмов семантического анализа. Для исследования были разработаны частотно-семантический, латентно-семантический и синтаксико- семантический алгоритмы. Для сравнения алгоритмов замерено время работы алгоритмов в зависимости от количества символов в тексте, а также точность определения тематики текстов.

* + 1. Набор данных

Для сравнения точности алгоритмов использован набор данных Datasets for evaluation of keyword extraction in Russian [35], предоставляющий данные для оценки выделения тематики текстов на русском языке. Для проведения экспериментов выбран набор данных, содержащий новостные статьи с портала Russia Today [36]. Данные в наборе представлены в виде файла формата jsonline, где каждая строка описана в json-формате. В данных содержится ссылка на источник, заголовок статьи, текст и выделенные ключевые слова.

* + 1. Результаты экспериментальных исследований

Для оценки скорости работы алгоритмов было измерено время работы каждого из алгоритмов на текстах с различным количеством символов.

В таблице 11 представлены результаты времени работы алгоритмов в зависимости от количества символов.

Таблица 11 – Время работы алгоритмов

|  |  |  |  |
| --- | --- | --- | --- |
| Количество символов | Время работы ЧСА, с | Время работы ЛСА, с | Время работы синтаксико- семантического  алгоритма, с |
| 1000 | 2 | 59 | 187 |
| 3000 | 3 | 81 | 262 |
| 5000 | 2 | 101 | 438 |
| 10000 | 4 | 152 | 591 |

На рисунке 22 представлен график времени работы алгоритмов в зависимости от количества символов.

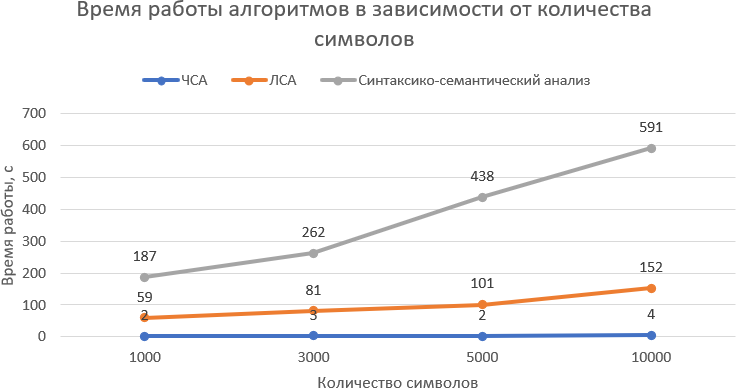


Рисунок 22 – Время работы алгоритмов в зависимости от количества символов

Частотно-семантический алгоритм показал наилучшие результаты по скорости работы, при увеличении размера анализируемого текста время работы изменяется незначительно. Дольше всего при анализе работает синтаксико-семантический алгоритм, в связи со сложностью данного алгоритма, время работы приблизительно в 4 раза дольше, чем у латентно- семантического алгоритма, показывающего средние результаты.

Для оценки точности работы на исследуемом наборе данных были рассчитаны статистические метрики precision и recall, часто использующиеся в задачах классификации [37].

Данные метрики рассчитываются на основе матрицы неточностей, представленной на рисунке 23.

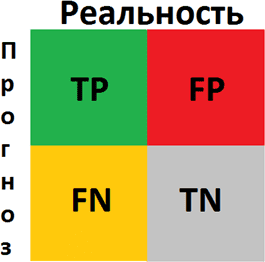


Рисунок 23 – Матрица неточностей

TP (True Positive) - означает, что положительный прогноз классификатора совпадает с реальностью. FP (False Positive) – положительный прогнозе совпадает с реальностью. FN (False Negative) – выдан ошибочный отрицательный результат. TN (True Negative) – отрицательный результат верен.

Метрика recall (полнота) в общем виде вычисляется по следующей формуле:

𝑟𝑒𝑐𝑎𝑙𝑙 =

𝑇𝑃

.

𝑇𝑃 + 𝐹𝑁

Данная метрика отражает количество верно классифицированных объектов к общему количеству данных этого класса.

В рамках задачи семантического анализа данную метрику можно охарактеризовать как отношение количества верно выделенных ключевых слов к количеству эталонных ключевых слов, представленных в анализируемом наборе данных.

Метрика precision (точность) отражает долю правильно классифицированных объектов среди всех классифицированных данных и вычисляется по формуле:

𝑝𝑟𝑒𝑐𝑖𝑠𝑖𝑜𝑛 =

𝑇𝑃

.

𝑇𝑃 + 𝐹𝑃

Таким образом, точность в задаче семантического анализа точность можно рассчитать, как отношение количества верно выделенных ключевых слов к общему количеству ключевых слов.

На рисунке 24 представлен пример данных, взятый из исследуемого корпуса.

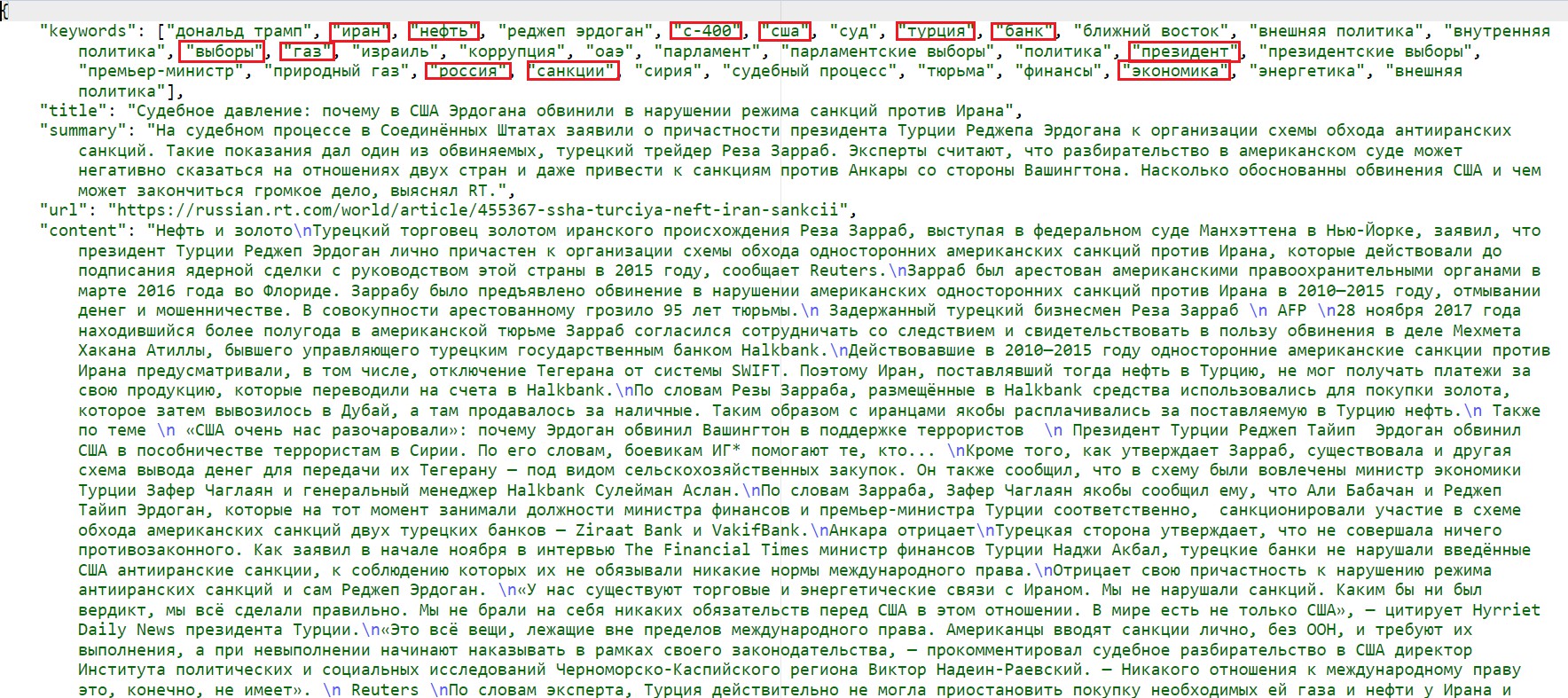


Рисунок 24 – Пример исследуемых данных

Для данной новостной статьи выделен обширный список ключевых слов, однако в данном списке присутствуют ключевые слова в виде выражений, в которых более чем одно слово, и не учитывающиеся при расчете метрик. Помимо этого, в данном наборе встречаются ключевые слова явно не указанные в самом тексте статьи, что влияет на точность анализа, в связи с тем, что исследуемые алгоритмы основываются на определении взаимосвязей слов, частоте встречаемости внутри текста. Красным цветом на рисунке выделены ключевые слова, встречающиеся в исследуемом тексте не менее одного раза. В таблице 12 представлены результаты выделения тематики на представленном примере данных с помощью реализованных алгоритмов.

Таблица 12 – Результаты анализа текста

|  |  |
| --- | --- |
| Алгоритм | Выделенные ключевые слова |
| Частотно-семантический  алгоритм | турция, санкции, сша, слова, отношения, президент,  года, сейчас, американцы, общество |
| Латентно-семантический  алгоритм | сша, отношения, президент, санкции, турция, нефть,  обвинение, эрдоган, банк, дело |
| Синтаксико-семантический  алгоритм | сша, санкции, иран, позиция, обвинение, нефть,  президент, виктор, турция, отношения |

Как можно заметить, выделенные ключевые слова частично совпадают с ключевыми слова исследуемого текста, однако не охватывают набор данных полностью, в связи с отсутствием некоторых ключевых слов в исходном тексте.

Для расчета метрик точности и полноты во всех алгоритмах использованы десять выделенных ключевых слов, обладающих наибольшим весом, в случае частотно-семантического анализа десять самых часто встречающихся слов в тексте. Из эталонных ключевых слов, представленных в наборе данных были исключены выражения, состоящие из нескольких слов.

В таблице 13 представлены характеристики, рассчитанные для алгоритмов.

Таблица 13 – Расчет метрик precision, recall

|  |  |  |  |
| --- | --- | --- | --- |
| Метрика | Частотно-  семантический алгоритм | Латентно-  семантический алгоритм | Синтаксико-  семантический алгоритм |
| precision | 57 % | 69 % | 63 % |
| recall | 44 % | 61 % | 56 % |

На основе вычисленных метрик, можно сделать следующие выводы, исследуемые алгоритмы показали среднюю точность определения тематики текста, в связи со сложностью анализируемого набора данных. Наиболее высокий результат получен с помощью латентно-семантического алгоритма, точность, которого составила приблизительно 69%, наименьший результат показал частотно-семантический алгоритм. Полученные результаты

напрямую связаны со сложностью текста, исходный набор данных содержит новостные статьи с различными тематиками, с наличием в тексте прямой речи, собственных частей речи, что усложняет анализ и приводит к ложным результатам. Помимо этого, алгоритмы семантического анализа выделяют семантику и семантические связи, основываясь на словах внутри текста. Представленные ключевые слова в наборе данных, выделены экспертами и часто являются более общими, такие как, политика и экономика, при этом в самом тексте данные ключевые слова могут быть и не отражены.

* + 1. Выводы по главе

В результате проведенных экспериментов по точности выделения семантики и скорости работы показал себя латентно-семантический алгоритм. Было выяснено, что исследование точности и полноты семантического анализа является нетривиальной задачей, результаты анализа напрямую зависят от сложности анализируемого текста и области применения. Так, в рамках задачи исследования поисковых запросов и интернет-маркетинга данные алгоритмы более уместны, в связи с тем, что алгоритмы основаны на выявлении семантических связей, частоте использования слов внутри текста. Для определения тематики новостных статей и художественных текстов необходимо использовать более сложные алгоритмы, например, нейросетевые.

ЗАКЛЮЧЕНИЕ

В процессе выполнения выпускной работы была разработана автоматизированная система семантического анализа текста, позволяющая определять тематику анализируемого текста, а также рассчитывать статистические характеристики, такие как количество слов, символов, водность, частота повторения слов. Система визуализирует полученные результаты в виде графиков и позволят выгружать результаты в текстовом формате. Помимо этого, в системе реализован администраторский интерфейс, отражающий статистику использования сервиса и отслеживающий наиболее популярные темы анализируемых текстов.

В первом разделе были приведены основные понятия и определения предметной области семантического анализа, рассмотрены существующие методы и алгоритмы задачи семантического анализа текста, приведены характеристики систем-аналогов, на основании этого была сформулирована постановка задачи и основные требования к системе.

Во втором разделе была разработана структура системы, разработан информационно-логический проект системы по методологии UML, логическая модель данных, а также был выбран и обоснован комплекс программных средств.

В третьем разделе описан интерфейс пользователя, физическая модель данных, а также приведены результаты проведенных исследований, проведен сравнительный анализ разработанных алгоритмов, решающих задачу семантического анализа текста.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. В России бум продаж подержанных машин. [Электронный ресурс] // Газета.Ru – URL: https://www.gazeta.ru/auto/2025/02/20/20585066.shtml (дата обращения: 22.04.2025).
2. 1 Обзор методов прогнозирования. [Электронный ресурс] // Ivan Shamaev – URL: https://ivan-shamaev.ru/overview-forecast-methods/ (дата обращения: 24.04.2025).
3. Что такое регрессионный анализ [Электронный ресурс] // Яндекс Практикум – URL: https://practicum.yandex.ru/blog/chto-takoe-regressionnyj-analiz/ (дата обращения: 22.04.2025).
4. Машинное обучение: общие принципы и концепции [Электронный ресурс] // Хабр: [сайт]. – URL: https://habr.com/ru/articles/862704/ (дата обращения: 22.04.2025).
5. Что такое модель машинного обучения? [Электронный ресурс]. // Microsoft Learn: [сайт]. – URL: https://learn.microsoft.com/ru-ru/windows/ai/windows-ml/what-is-a-machine-learning-model (дата обращения: 22.04.2025).
6. Regression in machine learning [Электронный ресурс]. // GeeksforGeeks: [сайт]. – URL: https://www.geeksforgeeks.org/regression-in-machine-learning/ (дата обращения: 22.04.2025).
7. Мочалова, А.В. Алгоритм сематического анализа текста, основанный на базовых семантических шаблонах с удалением // Научно- технический вестник информационных технологий, механики и оптики. – 2014. – Вып. 5.
8. Черненко О.А., Гордеева О.А. Разработка автоматизированной системы семантического анализа текстовой информации // «Информационные технологии и нанотехнологии» (ИТНТ-2017). — 2017. — С. 1807-1811.
9. Russian stemming algorithm [Электронный ресурс]. – URL: <http://snowball.tartarus.org/algorithms/russian/stemmer.html> (дата обращения: 18.03.2024).
10. Стемминг [Электронный ресурс]. – URL: <http://xlench.bget.ru/doku.php/p/common/stemming> (дата обращения: 18.03.2024).
11. Воронин В. М. Латентно-семантический и пропозициональный анализ связного текста / В. М. Воронин. С. В. Курицин // Психологический вестник Уральского государственного университета. Вып. 6. — Екатеринбург: Изд-во Урал. ун-та, 2008. — С. 198-201.
12. ADVEGO [Электронный ресурс]. – URL: https://advego.com/text/seo/ (дата обращения: 20.03.2024).
13. Miratext биржа копирайтеров [Электронный ресурс]. – URL: https://miratext.ru/seo\_analiz\_text (дата обращения: 20.03.2024).
14. ISTIO [Электронный ресурс]. – URL: https://istio.com/ (дата обращения: 20.03.2024).
15. Архитектура «клиент-сервер» [Электронный ресурс]. – URL: https://sergeygavaga.gitbooks.io/kurs-lektsii-testirovanie-programnogo- obespecheni/content/lektsiya-6-ch1-arhitektura-klient-server.html (дата обращения: 21.03.2024).
16. Как работают веб-приложения [Электронный ресурс] // ИНТУИТ Национальный открытый университет. – URL: https://intuit.ru/studies/courses/ 4455/712/lecture/21291 (дата обращения: 21.03.2024).
17. ПОНЯТИЕ ТОНКОГО И ТОЛСТОГО КЛИЕНТА [Электронный

ресурс] // TestMatick [сайт]. – URL: https://testmatick.com/ru/ponyatie-tonkogo-i- tolstogo-klienta/ (дата обращения: 21.03.2024).

1. HTTP [Электронный ресурс] // Википедия. Свободная энциклопедия: [сайт]. – URL: https://ru.wikipedia.org/wiki/HTTP (дата обращения: 21.03.2024).
2. Определение понятия «система» [Электронный ресурс]. – URL: <http://e-educ.ru/tsisa3.html> (дата обращения: 21.03.2024).
3. Диаграмма вариантов использования [Электронный ресурс] // ИНТУИТ Национальный открытый университет. – URL:

https://[www.intuit.ru/studies/courses/32/32/lecture/1004](http://www.intuit.ru/studies/courses/32/32/lecture/1004) (дата обращения: 21.03.2024).

1. Буч, Г. Язык UML. Руководство пользователя [Текст] /Г. Буч, Д. Рамбо, А. Якобсон. 2-е изд.: Пер. с англ. Мухина Н. М.: ДМК Пресс, 2006. 496 с.: ил.
2. Диаграмма последовательности [Электронный ресурс] //

Википедия. Свободная энциклопедия: [сайт]. – URL: https://ru.wikipedia.org/wiki/Диаграмма\_последовательности (дата обращения: 21.03.2024).

1. Создание диаграммы классов [Электронный ресурс] // Студопедия: [сайт]. – URL: https://studopedia.ru/19\_181519\_sozdanie- diagrammi-klassov.html (дата обращения: 23.03. 2024).
2. Основные понятия баз данных [Электронный ресурс]. – URL: <http://inf.susu.ac.ru/Klinachev/lc_sga_26.htm> (дата обращения: 23.03.2024).
3. ГОСТ 28397-89. Языки программирования. Термины и определения М., 1989. 8 с. (Издательство стандартов).
4. Руководство по языку программирования Java [Электронный ресурс] // Metanit: [сайт]. – URL: https://metanit.com/java/tutorial/ (дата обращения: 25.03.2024).
5. Spring Framework [Электронный ресурс]. – URL: https://spring.io/projects/spring-framework (дата обращения: 25.03.2024).
6. Stanford CoreNLP [Электронный ресурс]. – URL: https://stanfordnlp.github.io/CoreNLP/ (дата обращения: 25.03.2024).
7. TypeScript is JavaScript with syntax for types [Электронный ресурс]. – URL: https://[www.typescriptlang.org/](http://www.typescriptlang.org/) (дата обращения: 25.03.2024).
8. React JavaScript-библиотека для создания пользовательских интерфейсов [Электронный ресурс]. – URL: https://ru.legacy.reactjs.org/ (дата обращения: 25.03.2024).
9. Windows — что это такое? [Электронный ресурс] // internet-lab.ru: [cайт]. – URL: https://internet-lab.ru/windows\_os (дата обращения: 25.03.2024).
10. Интегрированная среда разработки [Электронный ресурс] // Википедия. Свободная энциклопедия: [сайт]. – URL: https://ru.wikipedia.org/ wiki/Интегрированная\_среда\_разработки (дата обращения: 25.03.2024).
11. СУБД PostgreSQL. Особенности и архитектура Postgres [Электронный ресурс] // otus: [сайт]. – URL: https://otus.ru/nest/post/1584/ (дата обращения: 25.03.2024).
12. Узел на диаграмме развертывания [Электронный ресурс] // Dev&Type: [сайт]. – URL: https://devtype.blogspot.com/2015/08/Uzel-na- diagramme-razvertyvaniya.html (дата обращения: 01.04.2024).
13. Datasets for evaluation of keyword extraction in Russian [Электронный ресурс]. – URL: https://geeksrepos.com/mannefedov/ ru\_kw\_eval\_datasets (дата обращения: 01.04.2024).
14. Russia Today [Электронный ресурс]. – URL: https://russian.rt.com/ (дата обращения: 01.04.2024).
15. Метрики качества моделей бинарной классификации [Электронный ресурс] // Loginom: [Сайт]. – URL: https://loginom.ru/blog/classification-quality (дата обращения: 01.04.2024).

ПРИЛОЖЕНИЕ А

Руководство пользователя

А.1 Назначение системы

Данная система предназначена для проведения семантического анализа текста и расчета его статистических характеристик.

А.2 Условия работы системы

Для корректной работы системы необходимо наличие соответствующих программных и аппаратных средств.

1. Требования к техническому обеспечению:
   * ЭВМ типа IBM PC;
   * процессор типа x86 или x64 тактовой частоты 1400 МГц и выше;
   * клавиатура или иное устройство ввода;
   * мышь или иное манипулирующее ввода;
   * дисплей с разрешением не менее 1280 × 768 пикселей;
   * широкополосное подключение к сети Интернет, не менее 1 Мб/сек.
2. Требования к программному обеспечению:
   * операционная система Windows 10 и выше;
   * виртуальная машина JVM 8.

А.3 Установка системы

Система поставляется в виде zip-архива. Данный файл необходимо распаковать в любую директорию на жестком диске. Запускаемым файлом системы является файл app.jar.

А.4 Работа с системой

При начале работы с системой пользователю открывается страница авторизации. При наличии существующей учетной записи, пользователю необходимо ввести свой логин и пароль, если учетная запись не создана, пользователь должен пройти регистрацию в системе, перейдя на соответствующую страницу. После ввода учетных данных пользователю

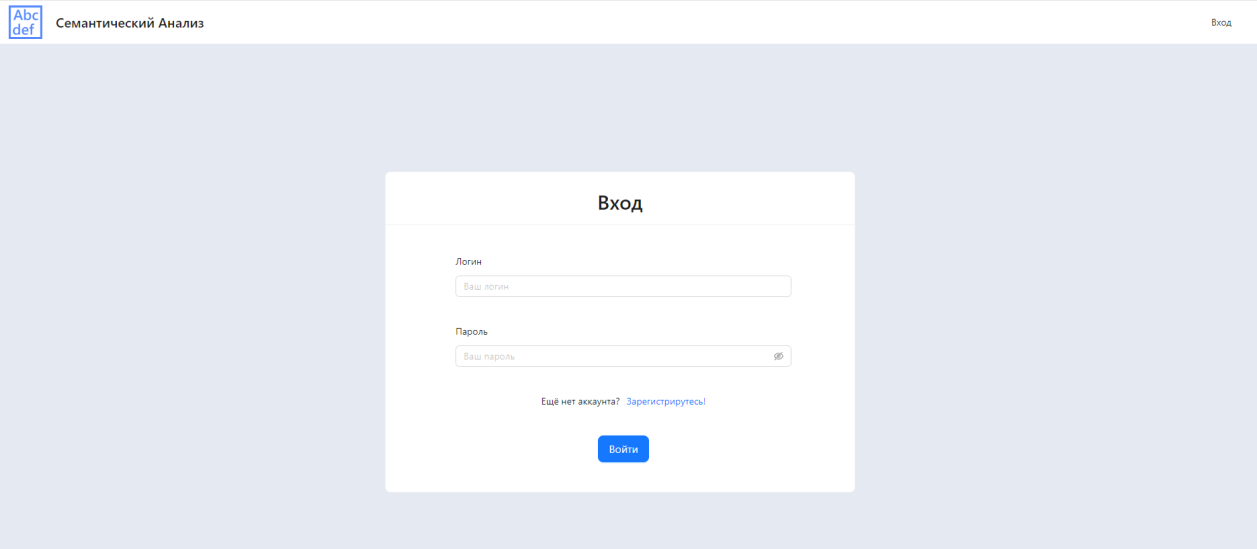
необходимо нажать на кнопку «Войти». На рисунке А.1 представлена страница авторизации в системе.

Рисунок А.1 – Страница авторизации

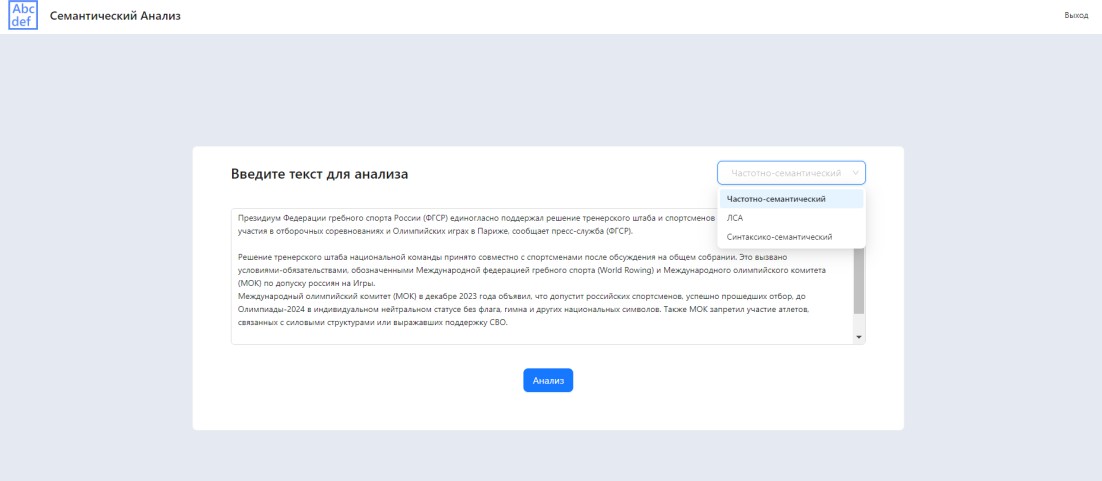
После входа в систему пользователь переходит на страницу анализатора, см. рисунок А.2.

Рисунок А.2 – Страница анализатора

На данной странице представлена форма ввода для анализируемого текста, а также выпадающий список с предложенными алгоритмами семантического анализа. После ввода текста и выбора алгоритма, пользователю необходимо нажать на кнопку «Анализ».

Далее система открывает страницу с представленными результатами анализа, на данной странице отображены вычисленные характеристики и выделена тема анализируемого текста, представлен график с частотой встречаемости слов. Для того, чтобы получить результаты анализа в виде docx файла, пользователь должен нажать на кнопку «Выгрузить», после чего

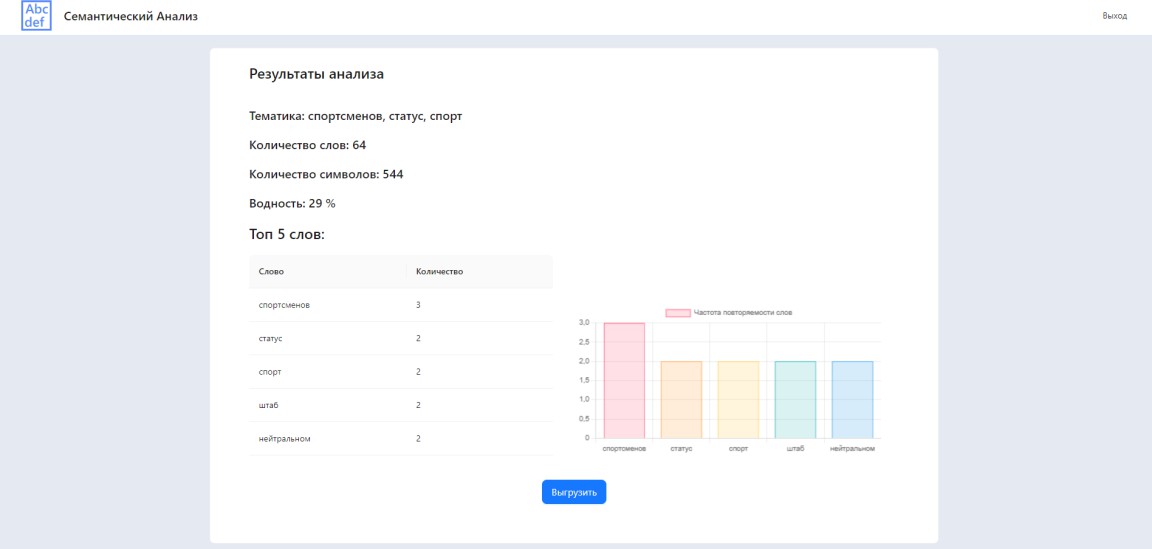
откроется системный проводник для выбора места загрузки файла. На рисунке А.3 представлена страница результатов анализа.

Рисунок А.3 – Результаты анализа текста

При входе в систему под встроенной ролью администратор, администратору открывается страница статистики, см. рисунок А.4.

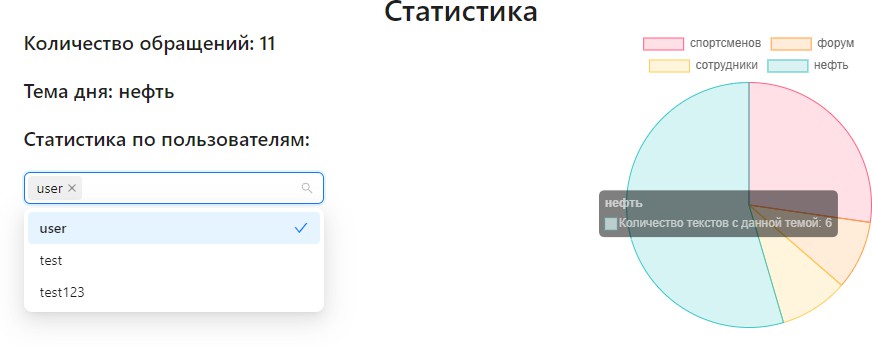


Рисунок А.4 – Статистика администратора

На данной странице отображается информация по количеству обращений к сервису, тема дня, а также график, отражающий темы текстов, по котором проводился анализ. Для того чтобы просмотреть статистику по определенным пользователям, администратору необходимо выбрать пользователя в выпадающем списки, предоставлена возможность поиска пользователя.

ПРИЛОЖЕНИЕ Б

Код программы type ResultPanelFactoryOptions = {

domain: Domain;

};

export const resultPanelFactory = modelFactory(({domain}: ResultPanelFactoryOptions) => {

const d = domain.createDomain('TextPanel'); const gate = createGate({domain: d});

const formSubmitted = d.createEvent();

const textFieldUpdated = d.createEvent<string>(); const analysisTypeChanged = d.createEvent<string>();

const $text = d.createStore<string>(''); const setText = d.createEvent<string>();

const $analysisType = d.createStore<string>(DEFAULT\_ALGORITHM\_OPTION\_VALUE);

const $analysisRequestPending = d.createStore(false); const $analysisRequestDone = d.createStore(false);

const $wordStatItems = domain.createStore<WordsStatsItem[]>([]); const setWordStatItems = domain.createEvent<WordsStatsItem[]>();

const $chartLabels = $wordStatItems.map(state => state.map(item => item.word).slice(0,5));

const $chartData = $wordStatItems.map(state => state.map(item => item.count).slice(0,5))

const $theme = domain.createStore('');

sample({

clock: $wordStatItems, fn: (state) => {

return state[0]?.word ?? ''

},

target: $theme,

})

const resetResults = domain.createEvent();

const semanticAnalyzeSecondFx = domain.createEffect(async (value: Record<string, string>) => {

return await requestSemanticAnalyseSecondMethod(value);

});

//const $semanticAnalyzeSecondData = domain.createStore<WordsStatsItem[]>([]);

sample({

clock: setWordStatItems,

fn: (items) => items.sort((a,b) => a.count < b.count ? 1 : -1), target: $wordStatItems,

})

sample({

clock: setText, target: $text,

})

const $isSelectedSecondAlgorithm = $analysisType.map(state => state

=== SelectAlgorithmOptions[1].value);

const $isSelectedThirdAlgorithm = $analysisType.map(state => state === SelectAlgorithmOptions[2].value);

const triggerSecondFx = domain.createEvent<string>(); const triggerThirdFx = domain.createEvent<string>();

const targetTriggerFxEvent = domain.createEvent<{ a: string; b: WordsStatsItem[];}>();

const targetTriggerThirdFxEvent = domain.createEvent<{ a: string; b: WordsStatsItem[];}>();

const debouncedTargetTriggerFxEvent = debounce(targetTriggerThirdFxEvent, 1000);

condition({ source: $text,

if: $isSelectedSecondAlgorithm, then: triggerSecondFx,

});

condition({ source: $text,

if: $isSelectedThirdAlgorithm, then: triggerThirdFx,

});

combineEvents({

events: {a: triggerThirdFx, b:setWordStatItems}, reset: gate.close,

target: targetTriggerThirdFxEvent,

});

combineEvents({

events: {a: triggerSecondFx, b:setWordStatItems}, reset: gate.close,

target: targetTriggerFxEvent,

});

sample({

clock: debouncedTargetTriggerFxEvent, source: $text,

fn: (source, clock) => ({text: source}), target: semanticAnalyzeSecondFx,

})

sample({

clock: targetTriggerFxEvent, source: $text,

fn: (source, clock) => ({text: source}), target: semanticAnalyzeSecondFx,

})

sample({

clock: semanticAnalyzeSecondFx.doneData,

fn: (data) => {

let theme = '', maxWeight = 0;

for (const key in data.data.result) {

if (data.data.result[key] > maxWeight && key.length > 3) { maxWeight = data.data.result[key];

theme = key;

}

}

console.log(data.data.result, theme);

return theme

},

target: $theme

})

debug($analysisType, $isSelectedSecondAlgorithm, triggerSecondFx);

return {

gate,

//$semanticAnalyzeSecondData,

$analysisRequestPending,

$analysisRequestDone,

$text, setText,

$theme,

$wordStatItems, setWordStatItems,

$chartData,

$chartLabels,

resetResults,

$analysisType

};

});

export type ResultPanelModel = Model<typeof resultPanelFactory>; @Service

@Slf4j

public record AuthService( AuthenticationManager authenticationManager, JwtTokenProvider jwtTokenProvider, PasswordEncoder passwordEncoder, UserRepository userRepository, RoleRepository roleRepository) {

public JWTAuthResponse login(LoginRequest loginRequest) { Authentication authentication = authenticationManager.authenticate(

new UsernamePasswordAuthenticationToken(loginRequest.getUsername(),

loginRequest.getPassword()));

if (authentication.isAuthenticated()) {

return new

JWTAuthResponse(jwtTokenProvider.generateToken(loginRequest.getUsername()

));

} else {

throw new UsernameNotFoundException(

"Не найден пользователь с именем " + loginRequest.getUsername());

}

}

public void signup(SignupRequest signupRequest) {

if (!userRepository.existsByUsername(signupRequest.getUsername())) { var roles = roleRepository.findByNameIn(signupRequest.getRoles()); userRepository.save(User.builder()

.username(signupRequest.getUsername())

.password(passwordEncoder().encode(signupRequest.getPassword()))

.roles(roles)

.build());

} else {

throw new UserAlreadyExistsException(String.format("Пользователь с логином %s уже существует", signupRequest.getUsername()));

}

}

}

@Component @FieldDefaults(level = PRIVATE) @RequiredArgsConstructor

public class JwtAuthenticationFilter extends OncePerRequestFilter {

JwtTokenProvider jwtService; UserDetailsServiceImpl userDetailsService;

@Override

protected void doFilterInternal(HttpServletRequest request, HttpServletResponse response,

FilterChain filterChain) throws ServletException, IOException { String authHeader = request.getHeader("Authorization");

String token = null; String username = null;

if (authHeader != null && authHeader.startsWith("Bearer ")) {

token = authHeader.split(" ")[1].trim(); username = jwtService.extractUsername(token);

}

if (username != null && SecurityContextHolder.getContext().getAuthentication() == null) {

UserDetails userDetails = userDetailsService.loadUserByUsername(username);

if (jwtService.validateToken(token, userDetails)) { UsernamePasswordAuthenticationToken authToken = new

UsernamePasswordAuthenticationToken(

userDetails, null, userDetails.getAuthorities()); authToken.setDetails(new

WebAuthenticationDetailsSource().buildDetails(request)); SecurityContextHolder.getContext().setAuthentication(authToken);

}

}

filterChain.doFilter(request, response);

}

@Autowired

public void setJwtService(JwtTokenProvider jwtService) { this.jwtService = jwtService;

}

@Autowired

public void setUserDetailsService( UserDetailsServiceImpl userDetailsService) { this.userDetailsService = userDetailsService;

}}