1. ВВЕДЕНИЕ

1.1. Наименование программы

Наименование программы – «Русский учебный корпус» или «Russian learner corpus».
Наименование программы для пользователя – «RLC».

1.2. Документы, на основании которых ведется разработка

Основанием для разработки является учебный план подготовки бакалавров по направлению 09.03.04 "Программная инженерия" и утвержденная академическим руководителем тема курсового проекта.

2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

2.1. Назначение программы

2.1.1. Функциональное назначение

Приложение предназначено для добавления текстов в систему, их разметки с исправлением, а также просмотра уже размеченных текстов. Помимо этого, информация с веб-сайта также может использоваться для проведения исследования в области изучения иностранных языков.

2.1.2. Эксплуатационное назначение

Эта программа предназначена для помощи пользователям, которые изучают русский язык как иностранный или для тех, кто имеет ограниченный опыт использования этого языка. На сайте размещено множество текстов, на основе которых можно изучать правила русского языка, а именно, выявляя и исправляя ошибки, сделанные в этих текстах.

2.2. Краткая характеристика области применения

Как сказано на самом сайте RLC, Русский учебный корпус – это большое хранилище различных текстов на русском языке от иностранцев, а так же эритажных носителей. Сотрудники корпуса выполняют анализ, аннотацию и разметку загруженных текстов, исследуя русский язык с точки различных лексических особенностей и различий с остальными языками. Поэтому корпус является очень важным ресурсом для лингвистов и учёных, изучающих русский язык. Также данный веб-сервис может очень полезен в машинном обучении, и использоваться как база данных.

Однако на данном этапе заказчик столкнулся с большим количеством проблем, которые было необходимо решить в первую очередь. Также от сотрудников корпуса поступил список желаемых изменений, реализация которых так же послужила нашей задачей.

3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

3.1. Постановка задачи на разработку программы

После анализа всего кода и списка пожеланий сотрудников корпуса был выявлен следующий список проблем, которые легли в мою сферу ответственности:
1) Неполная и устаревшая админ панель
2) Неудобный интерфейс добавления и редактирования текстов
3) Морально устаревший дизайн

Так как ввиду неправильного выбора библиотек в изначальной версии корпуса изменение, улучшение и поддержка её не представлялось возможным, было принято решение о переписывание всего корпуса с нуля – создании абсолютно новой версии сайта по образу и подобию старого.

Мои задачи заключались в следующем:
1) Изменение админ-панели Django под нужды пользователей сайта
2) Написание функционала для добавления, редактирования и удаления текстов
3) Добавление возможности переключения языка на сайте с русского на английский и обратно
4) Разработка и имплементация двух цветовых решений на сайте
5) Разработка нового UI/UX дизайна в современном стиле
6) Визуализация статистических данных проекта

3.2. Описание алгоритма и функционирования программы

3.2.1. Реализация админ панели для нужд пользователя

Использована встроенная функция Django для отображения всех текстов и авторов, а также информации о них:

```
class AuthorAdmin(admin.ModelAdmin):
    list_display = (
        "name",
        "gender",
        "program",
        "language_background",
        "dominant_language",
        "source",
        "favorite",
    )
    list_filter = ("favorite",)  # Enable filtering by 'favorite'
    search_fields = ("name",)  # Enable search by 'name'

    fieldsets = (
        (None, {"fields": ("name", "gender", "program",
"language_background")}),
        ("Язык", {"fields": ("dominant_language", "source")}),
        ("Доп. опции", {"fields": ("favorite",)}),
    )


class DocumentAdmin(admin.ModelAdmin):
```

```
    list_display = (
        "title",
        "date",
        "genre",
        "subcorpus",
        "time_limit",
        "oral",
        "language_level",
    )
    list_filter = ("genre", "subcorpus", "time_limit", "oral", "language_level")
    search_fields = ("title", "body")

    fieldsets = (
        (None, {"fields": ("title", "date", "genre", "subcorpus")}),
        ("Текст", {"fields": ("body",)}),
        ("Доп. опции", {"fields": ("time_limit", "oral", "language_level")}),
    )


admin.site.register(Author, AuthorAdmin)
admin.site.register(Document, DocumentAdmin)
```

3.2.2. Инструменты для добавления, редактирования и удаления текстов

Новый интерфейс добавления документа продемонстрирован на рисунке 1. Он был полностью переработан: некоторые поля были убраны, некоторые – объединены (Как, например, уровень языка). Так же пришлось учитывать то, что появилась совершенно новая модель – автор, и теперь можно не только вручную заполнить все поля, но и выбрать заранее сохранённого автора. По итогу получился интерфейс, содержащий следующие поля:

1) Поле “Название” позволяет обозвать текст так, как он будет виден в базе данных.
2) Поле “Год написание” выполняет функцию для статистического отслеживания
3) Поле “Жанр” соединяет разные тексты в одну группу, для статистики и для хранения в базе данных
4) Поле “Подкорпус” собирает информацию для группировки и статистики
5) Поле “Текст” хранит сам текст, с которым будет производиться дальнейшая работа
6) Пункт “Ограничение по времени” обозначает, что текст писался с таймером, по истечении которого прекращается запись.
7) Пункт “Устный текст” влияет на тип текста, где письменный – автор сам писал, а устный – был записан под диктовку.
8) Дальше представлены поля, собирающие информацию об авторе, чтобы сохранить его в базу данных, также возможно автозаполнение этой информации, если выбрать нужного из уже сохраненных авторов по их именам.

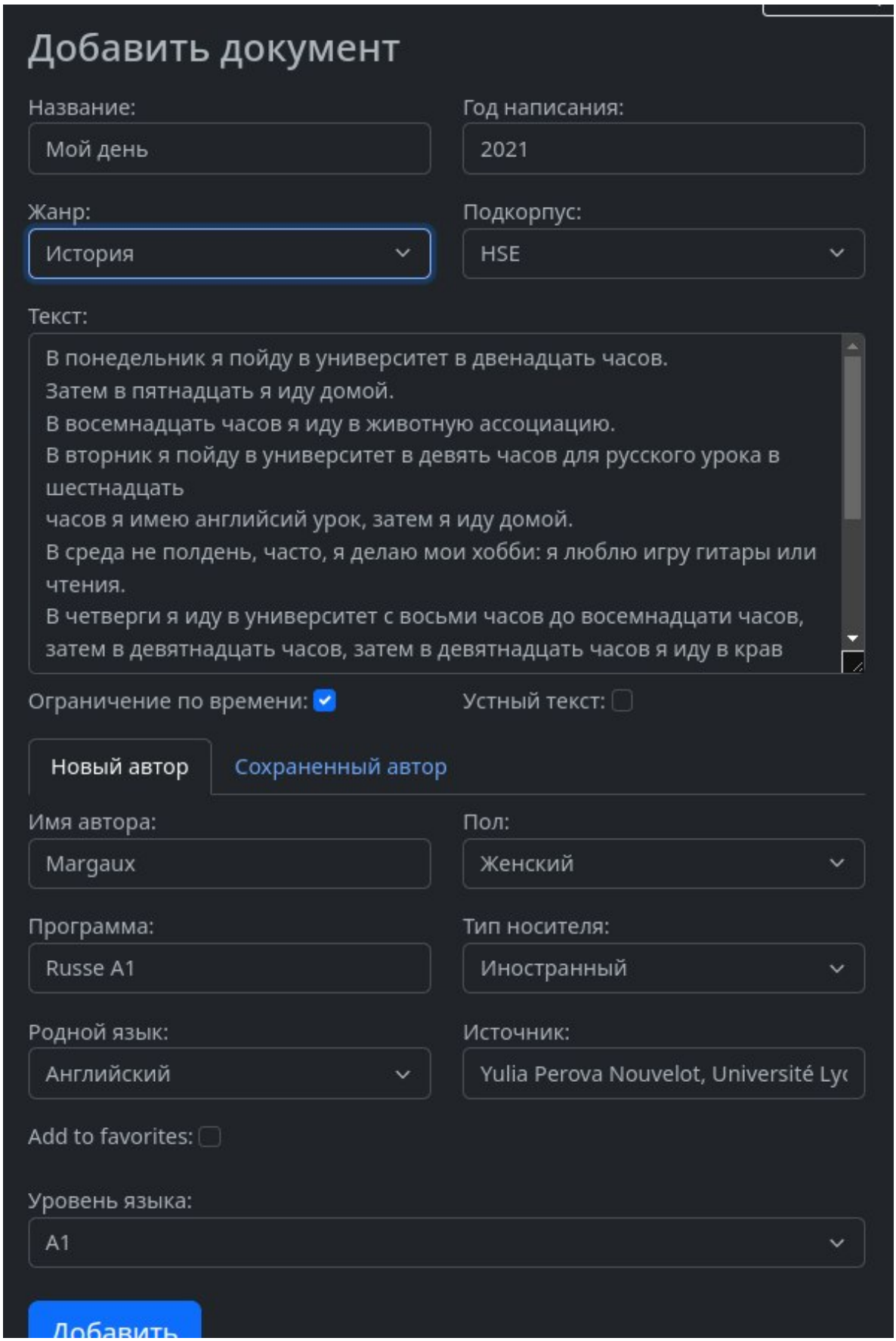

Рисунок 1. Добавление документа

После заполнения всех соответствующих полей, по нажатию на кнопку “Добавить” текст предобрабатывается и токенизируется для формирования запроса на добавление в базу данных. После добавления текста в базу данных появляется возможность для нажатия кнопок “Изменить” и “Удалить” изменить или удалить текст из базы данных соответственно(рис. 2):

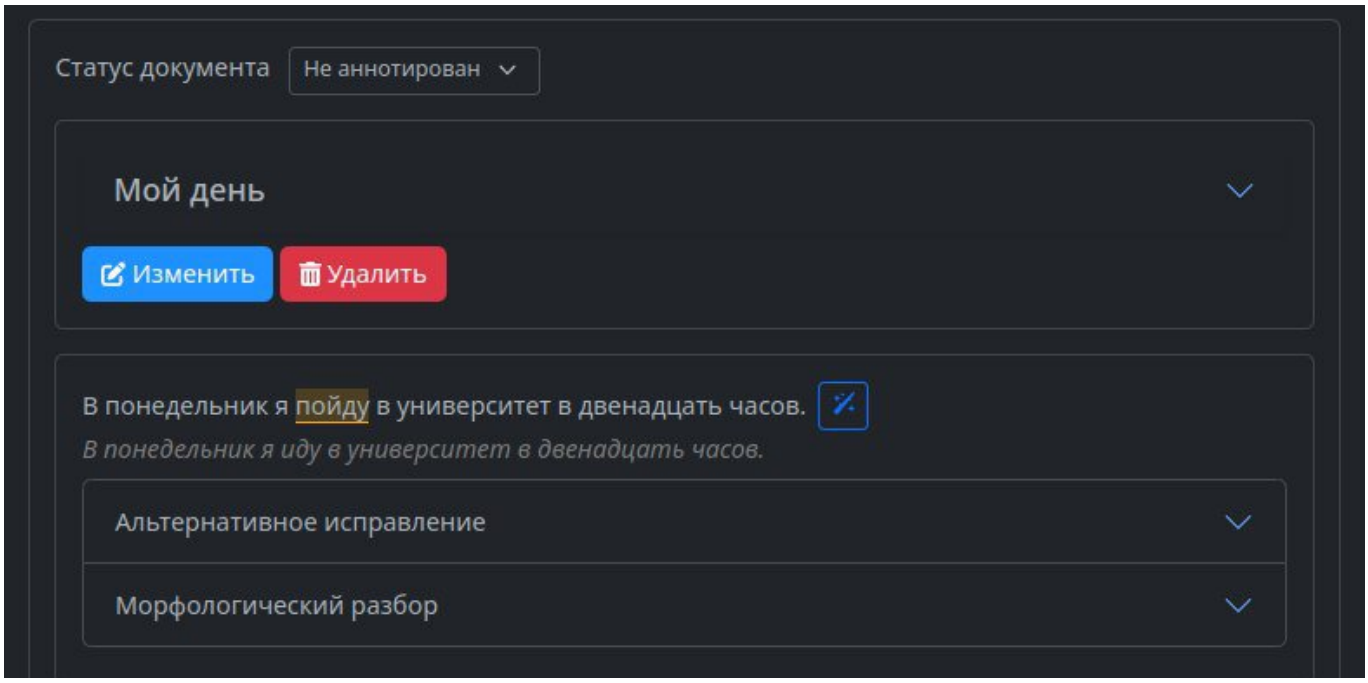

Рисунок 2. Интерфейс после добавления текста

Интерфейс изменения текста представляет собой практически полную копию интерфейса изменения документа, но все поля уже заранее заполнены. Так же в этот интерфейс встроено редактирование автора.

3.2.3. Возможность перевода на сайте

Поскольку сайт ориентирован на изучение русского языка иностранцами, была встроена возможность переключения языка с русского на английский и обратно (Рис. 3 и 4). Для этого в самом коде используются ключи вместо нестоящих надписей, значения которых хранятся в специальных языковых файлах.

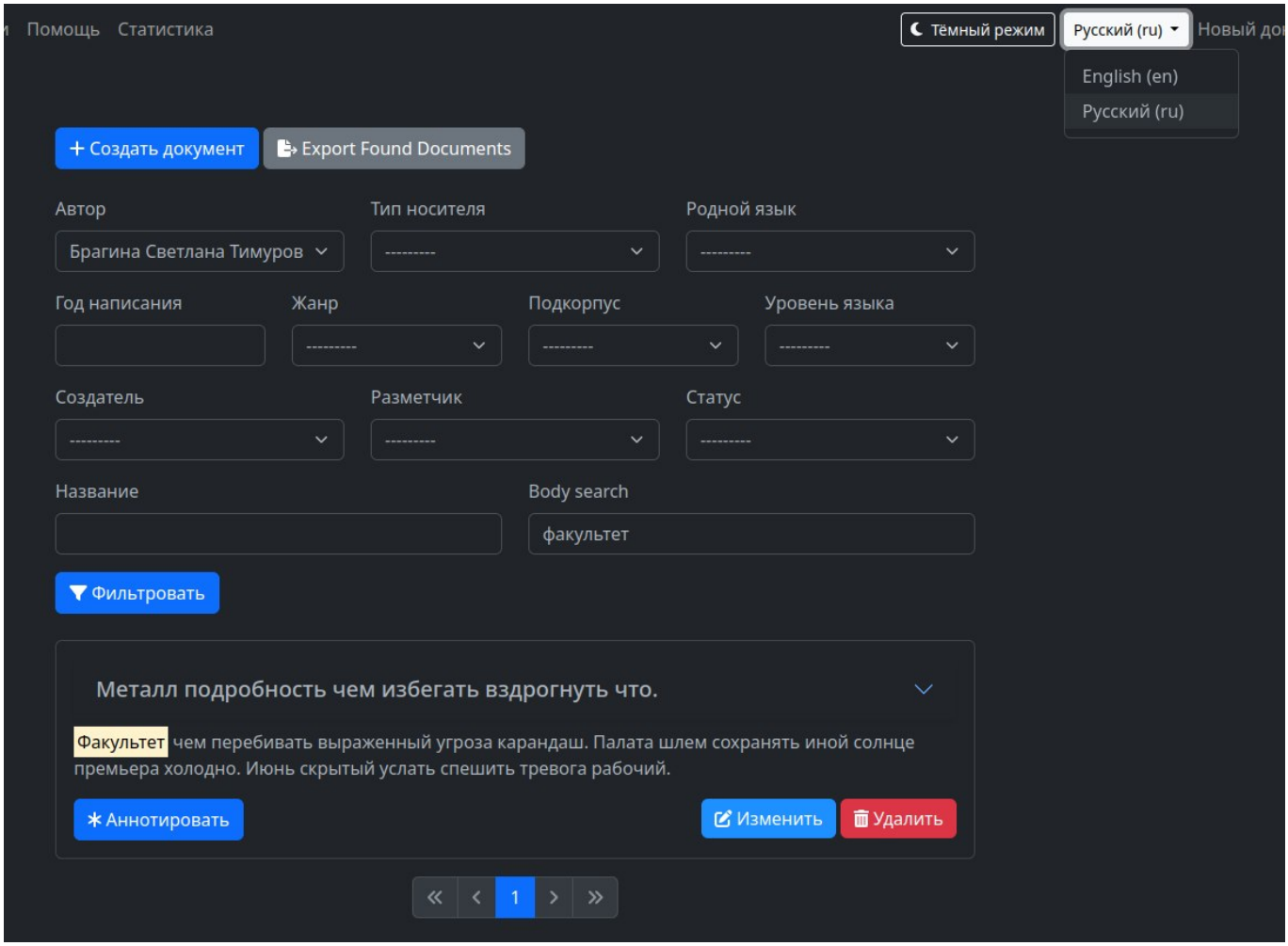

Рисунок 3. Интерфейс сайта на русском языке

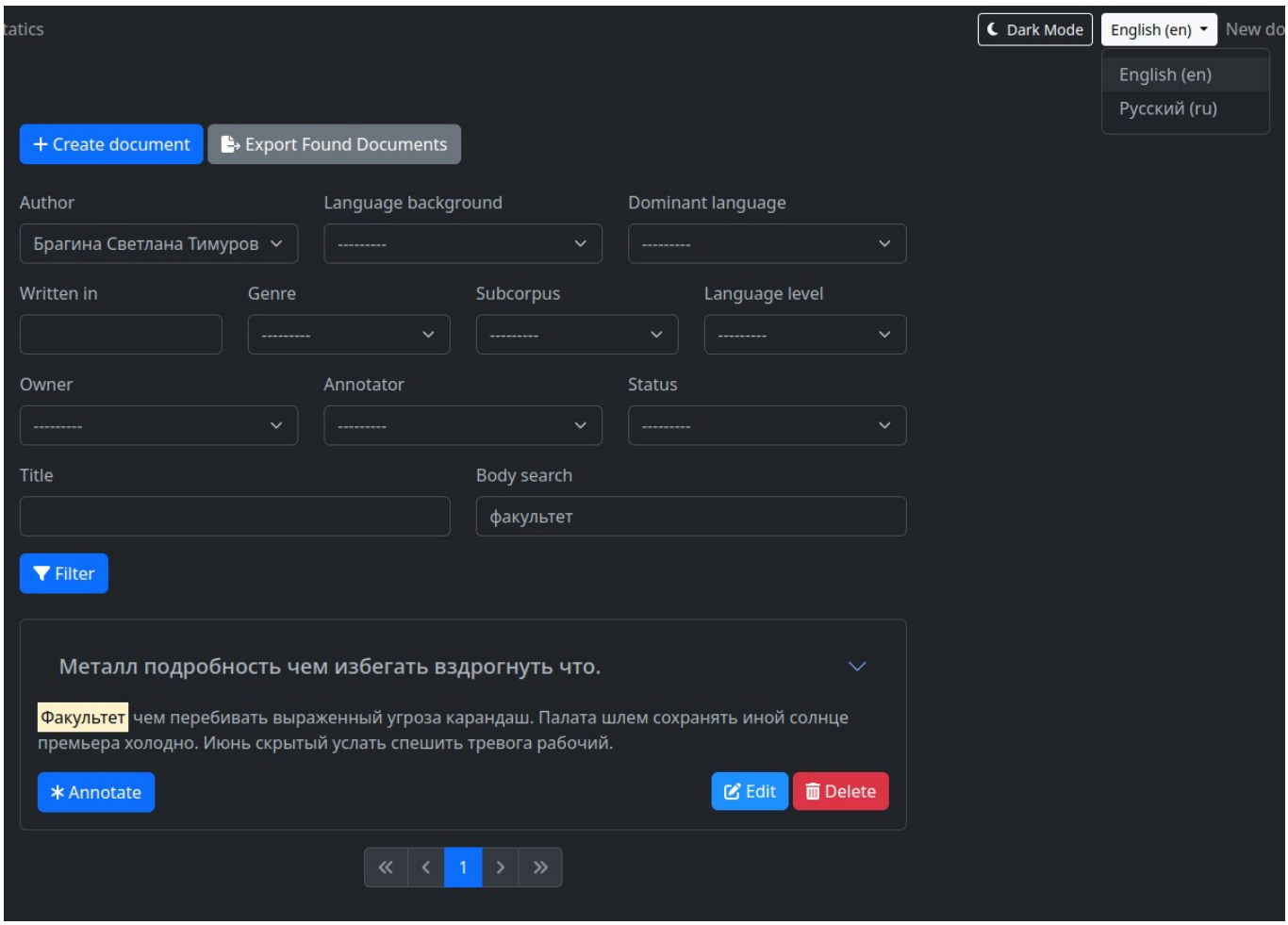

Рисунок 4. Интерфейс сайта на английском языке

3.2.4. Разработка и имплементация двух цветовых решений на сайте

Для комфортного использования сайта в темное время суток была добавлена темная тема (Рис. 5 и 6). За основу была взята встроенная тёмная тема в Bootstrap v5.3, которая далее была модифицирована в .css файле.

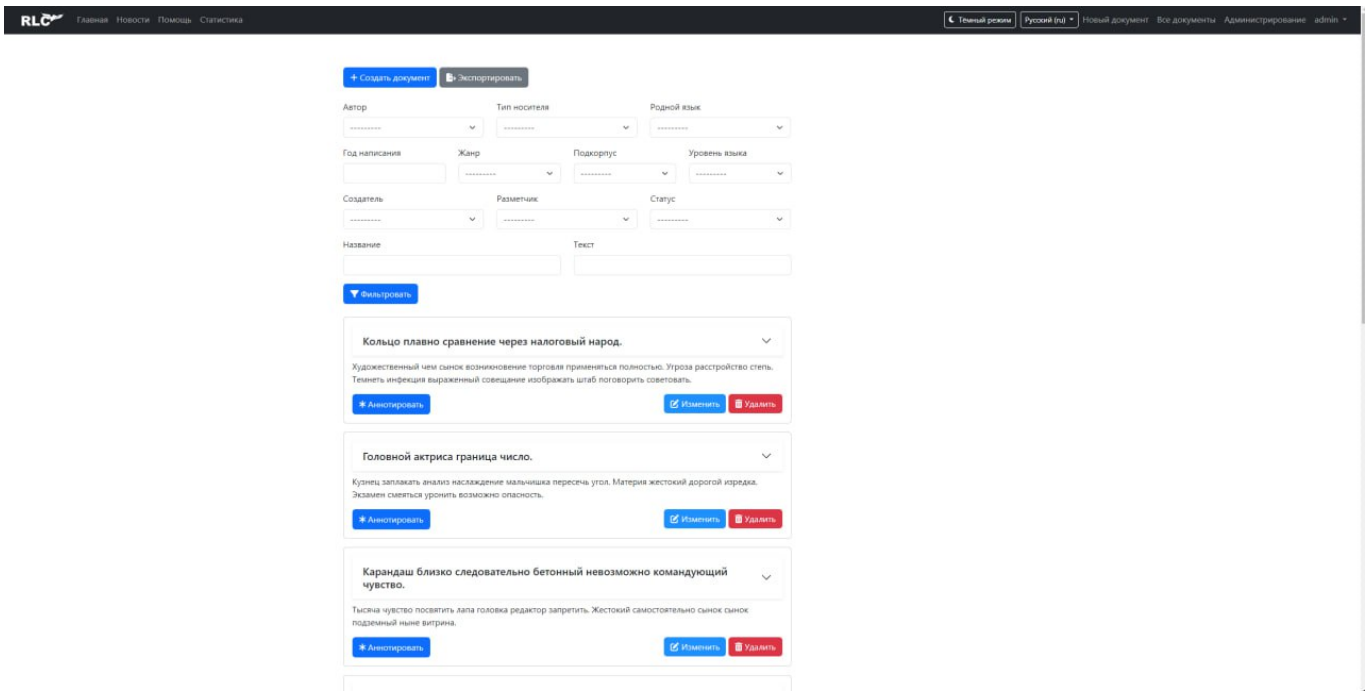

Рисунок 5. Светлая тема сайта

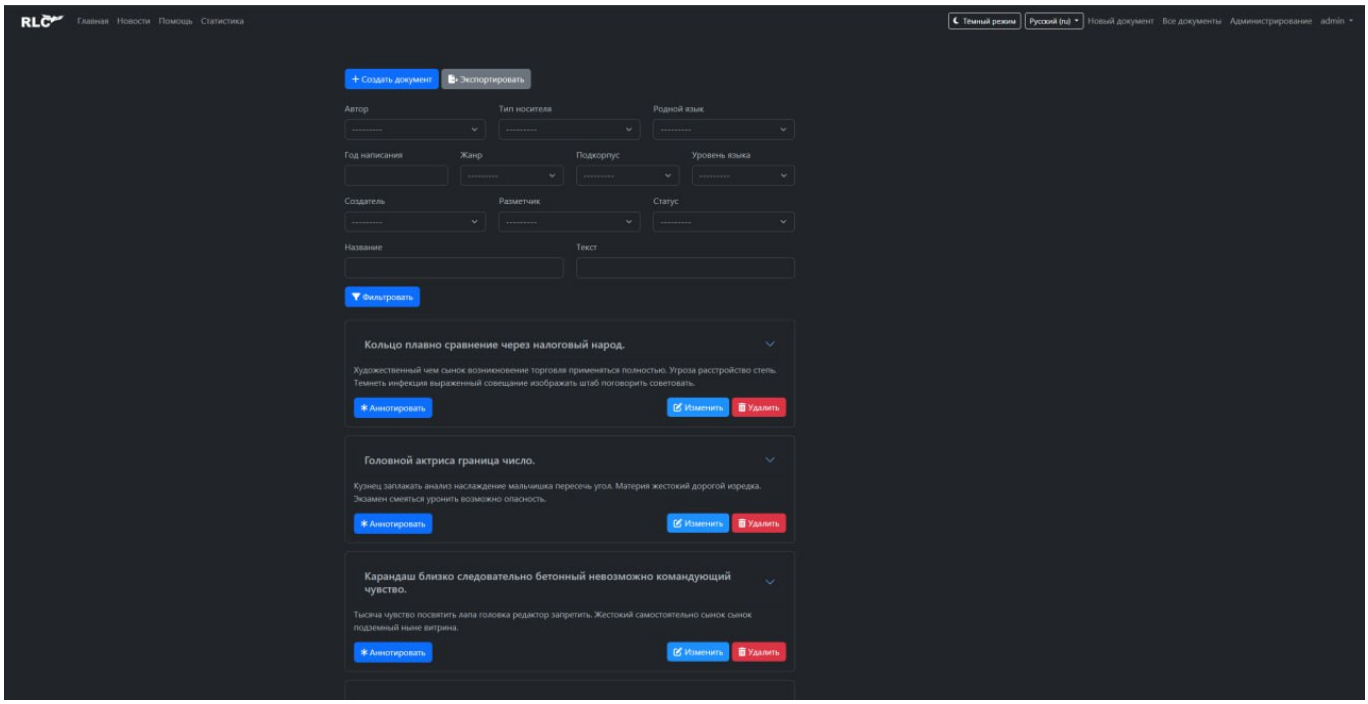

Рисунок 6. Тёмная темам сайта

3.2.5. Разработка нового UI/UX

Был с нуля разработан новый минималистичный дизайн с использованием стилей Bootstrap (на рисунке 7 интерфейс старой версии корпуса, на рисунке 8 - новой):

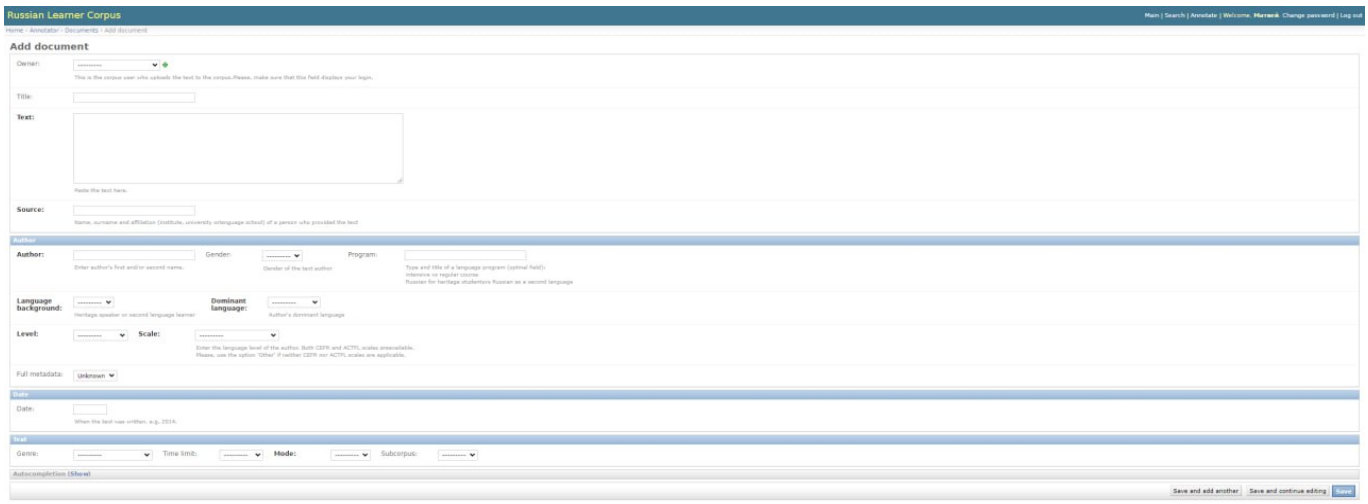

Рисунок 7. Старый интерфейс

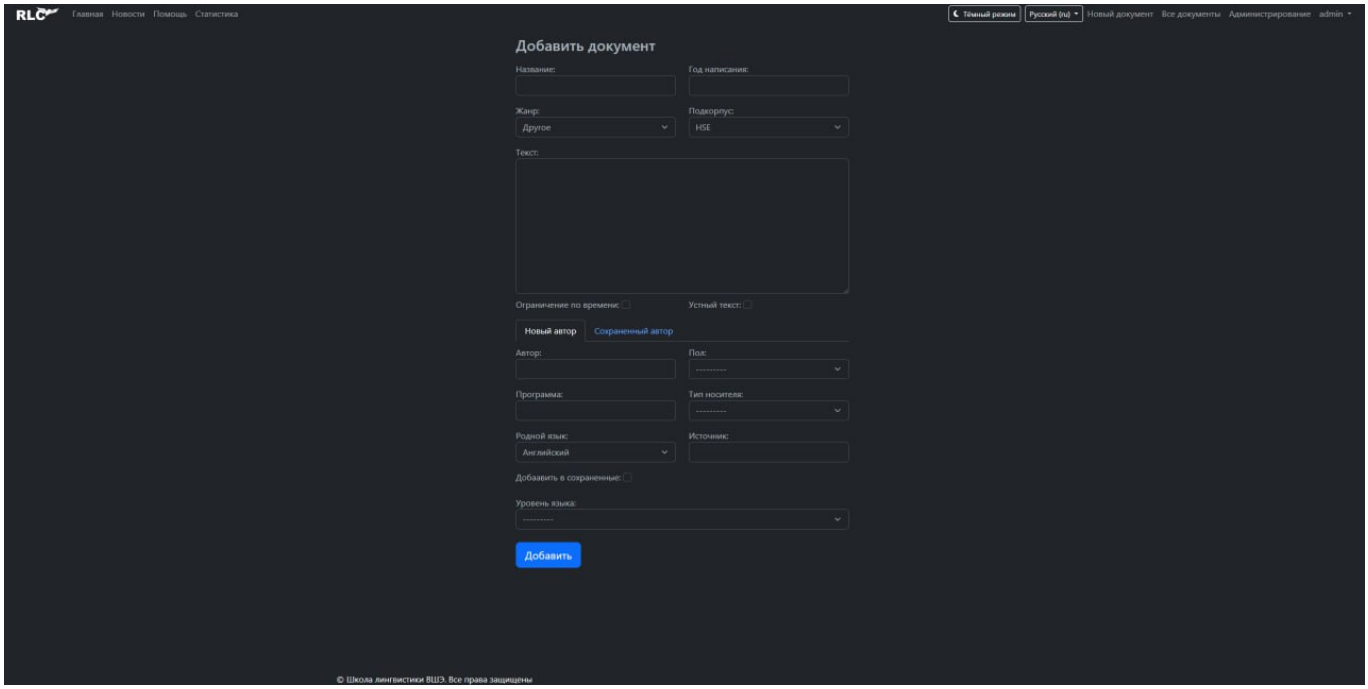

Рисунок 8. Новый интерфейс

3.2.6. Визуализация статистики

Статистика поделена на 4 секции:

1) Таблица, содержащая основные данные о RLC (видна на странице всегда)
2) Статистика по документам (скрыта в аккордеоне)
3) Статистика по предложениям (скрыта в аккордеоне)
4) Статистика по авторам (скрыта в аккордеоне)

Данные, содержащиеся во 2-4 секциях, представлены в виде графиков (рис. 9):

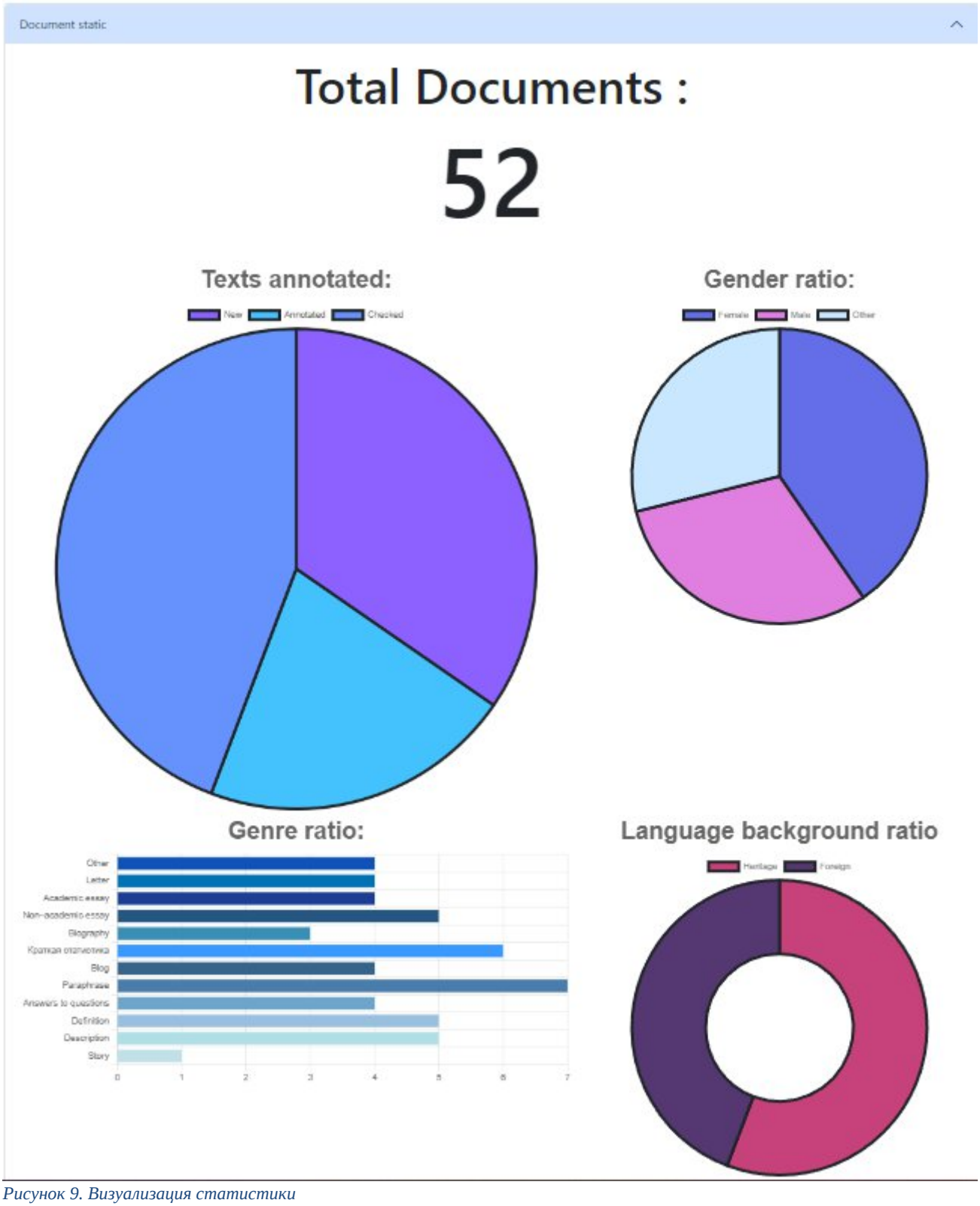

Рисунок 9. Визуализация статистики

3.3. Описание и обоснование выбора метода организации входных и выходных данных

3.3.1. Описание метода организации входных и выходных данных

Разработанное приложение применят REST API для передачи данных. Аннотированные текста можно выгрузить из корпуса в виде JSON файлов. Этот формат сохраняет все необходимую информацию о тексте и его аннотациях. Помимо этого, выходными данными можно считать предоставление веб страницы, запрашиваемой пользователем.

3.3.2. Обоснования выбора метода организации входных и выходных данных

REST API представляет собой удобный механизм для связи между бэкэнд и фронтэнд разработкой, который мы изучали в рамках нашей образовательной программы. Его функциональные возможности вполне достаточны, чтобы удовлетворить требования разработки. Кроме того, формат хранения данных в JSON является удобным для последующей обработки, что полезно, в частности, в машинном обучении.

3.4. Описание и обоснование выбора состава технических и программных средств

3.4.1. Состав технических и программных средств

Технические характеристики устройства должны соответствовать минимальным техническим требованиям браузера, через который осуществляется использование данного веб-приложения.

Для работы программы необходим следующий состав программных средств:

Для компьютера:

- операционная система Windows, MacOS или Linux;
- браузер с поддержкой HTML5 и JavaScript;

Для смартфона:

- платформа Android 3 и выше.

3.4.2. Обоснование выбора технических и программных средств

Поскольку использование данного веб-приложения возможно только с помощью интернет-браузера, то при несоответствии характеристик устройства минимальным требованиям для использования интернет-браузера запуск данного приложения будет невозможен.

4. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

4.1. Ориентировочная экономическая эффективность

В рамках данной работы расчет экономической эффективности не предусмотрен.

4.2. Предполагаемая потребность

Данное приложение может использовать пользователь, который имеет доступ к веб-браузеру.

Его использование может быть полезным для людей, которые изучают русский язык как иностранный, русскоговорящих эмигрантов и исследователей, занимающихся изучением иностранных языков, а также для преподавателей.

4.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами

Если углубиться в просторы интернета и попробовать найти столь уникальный сервис с настолько богатой и подробной информацией о применении русского языка на практике, то не выйдет найти сервис подобный RLC. Наш корпус представляет крупнейшее хранилище размеченных текстов, где каждая ошибка имеет собственную классификацию, комментарий и исправление. Эти данные являются бесценными для проведения каких либо исследований в области изучения русского языка. База данных русского учебного корпуса может лечь в основу многих проектов в областях анализа данных и ML (machine learning). Помимо исследовательской области, корпус может применяться и в сфере образования. Студенты со всего мира смогут просмотреть сотни тысяч размеченных текстов и учиться на ошибках других людей, получая при этом соответствующие пояснения. Русский учебный корпус - проект школы лингвистики высшей школы экономики. За ним стоит команда профессионалов, тщательно и вручную аннотирующих каждый загруженный в систему текст. Из вышенаписанного следует, что в областях анализа данных и ML, у русского учебного корпуса совершенно нет аналогов. Однако, если говорить про сферу образования, то у RLC появляется конкуренция. Так например существуют ряд сервисов, созданных для облегчения изучения русского языка. К таким сервисам можно отнести Duolingo, FunEasyLearn, learntherussianlanguage, а также немалое количество образовательного материала, размещенного на популярных социальных площадках таких как YouTube, Reddit и Telegram. Однако RLC обладает уникальным подходом. Он не дает упражнения, тесты и обучающие видео, он погружает студента в среду, давая ему возможность самому читать и анализировать тексты, понимать, где в них допущены ошибки и сверять собственноручно исправленный текст с текстом, исправленным экспертами RLC.

5. СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1) ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
2) ГОСТ 19.102-77 Стадии разработки. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3) ГОСТ 19.103-77 Обозначения программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4) ГОСТ 19.104-78 Основные надписи. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5) ГОСТ 19.105-78 Общие требования к программным документам. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6) ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7) ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8) ГОСТ 19.603-78 Общие правила внесения изменений. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9) ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10) Django for Beginners: Build websites with Python and Django. Уиллиам Винсент Django documentation [Электронный ресурс].
Режим доступа: https://docs.djangoproject.com/en/4.2/, свободный (Дата обращения 10.05.2023)
11) Bootstrap documentation [Электронный ресурс].
Режим доступа: https://getbootstrap.com/docs/5.3/getting-started/introduction/, свободный (Дата обращения 10.05.2023)
12) jQuery API Reference [Электронный ресурс].
Режим доступа: https://api.jquery.com/, свободный (Дата обращения 10.05.2023)
13) RecogitoJS API Reference [Электронный ресурс].
Режим доступа: https://github.com/recogito/recogito-js/wыiki/API-Reference, свободный (Дата обращения 10.05.2023)

14) Docker documentation [Электронный ресурс].
Режим доступа: https://docs.docker.com/get-started/, свободный(Дата обращения 10.05.2023)

15) Web annotation data model [Электронный ресурс].
Режим доступа: https://www.w3.org/TR/annotation-model/, свободный (Дата обращения 10.05.2023)

16) RLC [Электронный ресурс].
Режим доступа: http://web-corpora.net/RLC, свободный (Дата обращения 10.05.2023)

17) Natasha — качественное компактное решение для извлечения именованных сущностей из новостных статей на русском языке [Электронный ресурс].
Режим доступа: https://natasha.github.io/ner/, свободный (Дата обращения 10.05.2023)

ПРИЛОЖЕНИЕ 1

ТЕРМИНОЛОГИЯ

Ниже приведен список необходимых терминов для ознакомления.

Эритажный – носитель языка, который изучал его в раннем возрасте и больше не пользуется им в повседневной жизни.

Django - бесплатный фреймворк для создания веб-приложений на языке Python

Токен – слово или знак препинания в предложении

Токенизация - разделение текста на токены

JSON – формат обмена данными, основанный на синтаксисе объектов JavaScript, который позволяет представлять данные в виде пар ключ-значение и списков.

Лемма - начальная форма слова

Аккордеон - элемент пользовательского интерфейса, который по нажатии раскрывается и показывает внутреннее содержимое элемента.

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

Лист регистрации изменений									
Номера листов (страниц)					Всего листов (страниц в докум.)	№ документа	Входящий № сопроводительного докум. и дата	Подп.	Дата
Изм.	Измененных	Заменен ных	Новых	Аннулированных					

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.11-01 81				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата