УДК 004.9

РАЗРАБОТКА ЧАСТОТНОГО СЛОВАРЯ И КОРПУСА ДОКУМЕНТОВ РУССКОГО ДОРЕФОРМЕННОГО ЯЗЫКА

Трошина Анна Валерьевна[1], Ершов Николай Михайлович[2]

[1]Студент магистратуры;
ГБОУ ВО МО «Университет «Дубна»,
Институт системного анализа и управления;
141980, Московская область, г. Дубна, ул. Университетская, 19;
e-mail: annet_troshina@mail.ru.

[2]Доцент;
ГБОУ ВО МО «Университет «Дубна»,
Институт системного анализа и управления;
141980, Московская область, г. Дубна, ул. Университетская, 19;
e-mail: ershovnm@gmail.com.

Работа посвящена вопросам создания корпуса русского языка в дореформенной орфографии и разработке основанного на этом корпусе частотного словаря русского языка 18-го – начала 20-го веков. Рассматриваются и анализируются существующие подходы к решению поставленной задачи, в том числе приводится обзор ряда наиболее популярных электронных национальных корпусов – русского, британского и чешского. Формулируется модель внутренней организации электронного частотного словаря и его функционал. Описывается программная реализация корпуса русского дореформенного языка и основанного на нем частотного словаря с использованием языков программирования Python и Javascript и базы данных Mongo DB. Рассматриваются вопросы реализации веб-приложения для доступа к разработанному электронному словарю.

Ключевые слова: частотный словарь, лингвистический корпус, распознавание текстов.

Для цитирования:
Трошина, А. В., Ершов, Н. М. Разработка частотного словаря и корпуса документов русского дореформенного языка / // Системный анализ в науке и образовании: сетевое научное издание. – 2020. – № 3. – С. 31–42. – URL: http://sanse.ru/download/404.

DEVELOPMENT OF A WORD FREQUENCY LISTS AND TEXTS CORPUS OF RUSSIAN PRE-REFORM LANGUAGE

Troshina Anna[1], Ershov Nikolay[2]

[1]Graduate student;
Dubna State University,
Institute of the system analysis and management;
141980, Russia, Moscow Region, Dubna, Universitetskaya str., 19;
e-mail: annet_troshina@mail.ru

[2]Assistant professor;
Dubna State University,
Institute of the system analysis and management;
141980, Russia, Moscow Region, Dubna, Universitetskaya str., 19;
e-mail: ershovnm@gmail.com

The paper is devoted to the development of the Russian language corpus in pre-reform spelling and the development of a frequency word list based on this corpus of the Russian language of the 18th - early 20th centuries. Existing approaches to solving this problem are considered and analyzed, including an overview of a number of the most popular electronic national corpuses – Russian, British and Czech. The model of the internal organization of the electronic frequency word list and its functionality are formulated. The software implementation of the Russian pre-reform language corpus and the frequency word list based on it is de-

scribed using the programming languages Python and Javascript and the Mongo DB database. The issues of web application implementation for access to the developed electronic dictionary are considered.

Keywords: word frequency list, linguistic corpus, texts recognition.

For citation:
Troshina, A., Ershov, N. Development of a word frequency lists and texts corpus of russian pre-reform language = Разработка частотного словаря и корпуса документов русского дореформенного языка // System Analysis in Science and Education. – № 3. – Pp. 31–42. – URL: http://sanse.ru/download/404.

Введение

Частотные словари являются необходимым и полезным вспомогательным средством при решении многих практических задач, например, в преподавании, при создании новых словарей (например, орфографических или толковых), для разработки приложений компьютерной лингвистики. Частотный словарь русского дореформенного языка предполагается использовать прежде всего в качестве составной части системы по распознаванию печатных текстов того времени. С одной стороны такой словарь должен пополняться на основе распознаваемых текстов, с другой стороны, хранимая в нем информация может и должна использоваться для улучшения работы самой системы распознавания, например, для разрешения спорных случаев или для автоматического выделения проблемных фрагментов распознаваемых текстов на основе анализа контекста (исторического, тематического или внутритекстового).

Основой построения частотных словарей являются *корпусы* – собрания текстовых документов, привязанные к конкретному языку, заданному времени, определенной тематике и т.д. Корпусы современных языков активно разрабатываются и для большинства из них имеются соответствующие веб-приложения, позволяющие пользователю получать различную статистическую информацию о том или ином слове. К сожалению, сами наборы текстовых документов и привязанные к ним частотные словари либо не доступны пользователям, либо доступны частично и только на платной основе. Все вышесказанное касается и весьма немногочисленных дореволюционных корпусов русского языка. Это делает актуальным задачу составления своего собственного корпуса на основе имеющихся открытых и свободных текстов.

Обзор национальных корпусов

Одним из ведущих направлений прикладной лингвистики является корпусная лингвистика [1], занимающаяся разработкой общих принципов построения и применения корпусов с использованием информационных технологий. Текстовый корпус – большой, представленный в машиночитаемом формате, унифицированный, структурированный, размеченный, филологически компетентный массив языковых данных, предназначенный для конкретных лингвистических задач. Основными чертами современного корпуса являются машиночитаемый формат, репрезентативность, наличие металингвистической информации. В качестве базы для корпуса в основном используются тексты, которые представляют язык во всех его проявлениях. Классифицировать корпусы можно по разным признакам: цель создания корпуса, тип языковых данных, динамичность, тип разметки, объём текстов, параллельность, аннотирование.

Из определения корпуса можно выделить главные свойства текстового корпуса:

- электронный – корпус должен быть в электронном виде и использоваться на электронных устройствах;
- репрезентативный – должен хорошо «представлять» объект, который моделирует;
- размеченный – главное отличие корпуса от коллекции текстов;
- прагматически ориентированный – должен быть создан под определенные цели;

Целесообразность создания текстовых корпусов объясняется следующими факторами:

- демонстрация лингвистических данных в реальном контексте;

- возможность многократного использования корпуса для решения различных лингвистических задач, таких, как например, реализация лексико-грамматического анализа текста;
- репрезентативность данных.

В качестве популярных примеров электронных корпусов, доступных в открытом доступе в сети Интернет, можно привести национальный корпус русского языка, британский национальный корпус и чешский национальный корпус.

В национальный корпус русского языка входят письменные тексты – художественная литература, мемуары, публицистика, научные тексты, религиозная литература, повседневная печатная продукция, а также записи устных текстов – публичной речи и частных бесед [2]. Также в корпус входят подкорпусы поэтических и диалектных текстов, корпусы параллельных текстов, отдельный газетный корпус (материалы СМИ начала XXI века), церковнославянский корпус (богослужебные тексты, современные (XIX—XX век) и более ранних периодов), исторический (древнерусский, старорусский, берестяных грамот), синтаксический, мультимедийный и обучающий подкорпусы. Общий объём корпусов насчитывает более 600 млн. словоупотреблений. На сегодняшний день в свободном доступе возможно осуществлять только поиск по корпусу. Сайт корпуса и поиск по нему (рис. 1) поддерживается кампанией «Яндекс», сотрудники которой принимали участие также в разработке программного обеспечения корпуса. Доступ ко всему корпусу невозможен в связи с законом об авторских правах. Для получения доступа к 1/6 размеченной части подкорпуса необходимо зарегистрироваться и принять лицензионное соглашение.

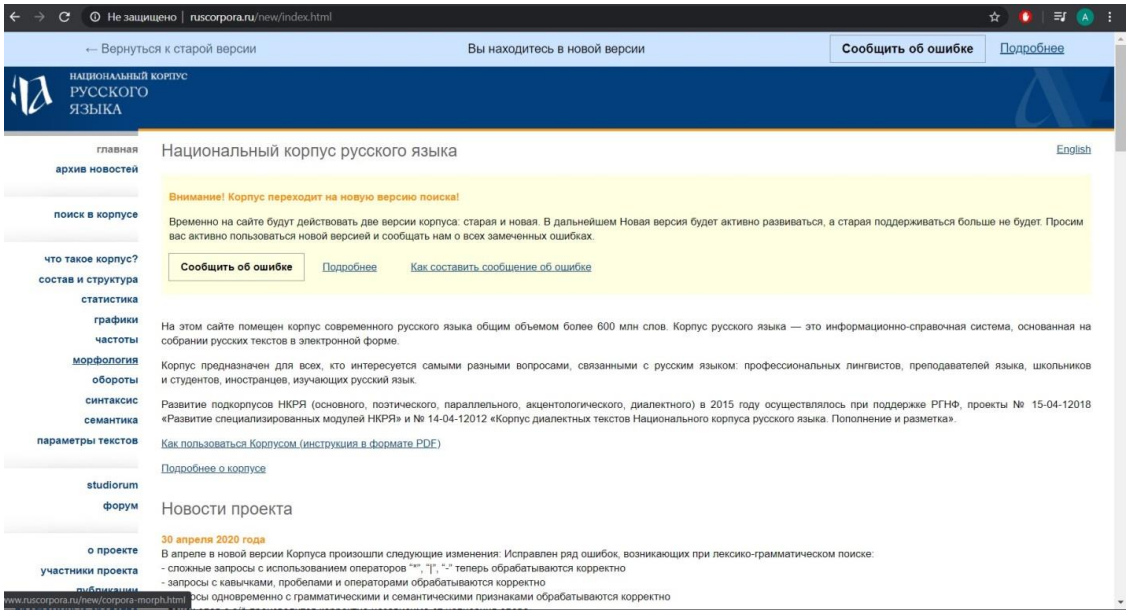

Рис. 1. Главная страница национального корпуса русского языка

Британский национальный корпус (англ. *British National Corpus, BNC*) – корпус, состоящий из более 100 миллионов слов, который содержит образцы как письменного, так и разговорного британского английского [3]. Корпус охватывает британский английский конца XXв., и задуман, как образец типичного разговорного и письменного британского английского языка того времени. Структура корпуса состоит из 2 частей (рис. 2): основную часть составляют образцы употребления письменного языка (90%), в которую включены тексты различных типов – газет, научных журналов, периодических изданий, художественной литературы оставшееся место (10%) занимают образцы применения разговорного языка, которые были представлены и записаны с помощью практической транскрипции. Корпус *BNC* содержит частеречную разметку – результат автоматической обработки текста, задачей которого является определение к какой части речи относится слово, а также его грамматических характеристик. Для этого при создании корпуса использовалась система разметки *CLAWS*. На сегодняшний день система модернизирована до версии *CLAWS*4, которая и была использована в корпусе. Издание корпуса доступно в формате *XML* и поставляется с программным обеспечением поисковой системы *Xaira*. Заказать корпус возможно через веб-сайт Британского национального корпуса.

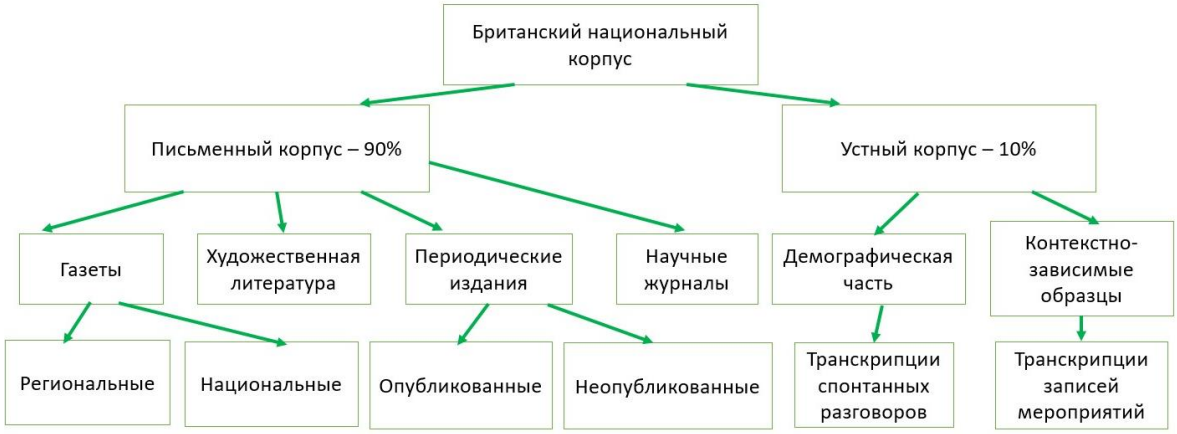

Рис. 2. Структура британского национального корпуса

Чешский национальный корпус (*Český národní korpus*, *ČNK*) – доступная для открытого поиска база письменных текстов в электронной форме на чешском языке. Сайт доступен на чешском и английском языках. Общий объем корпуса составляет свыше 9 млрд. словоупотреблений из которых размечено морфологическими тегами 8895 млн. Основным содержимым Чешского национального корпуса являются: тексты различных издательских домов, полученные в электронном виде; тексты из газет; тексты словарей. На сайте корпуса существует два вида доступа: публичный и полный. Полный доступ даёт возможность использовать базу данных института Чешского Национального Корпуса, а также специальному менеджеру корпуса *Bonito*. *Bonito* (*A Modular Corpus Manager Bonito*) – графический пользовательский интерфейс. Публичный доступ позволяет воспользоваться только корпусом *SYN2010,* объём которого 100 млн. слов, что составляет одну девяностую всей базы Чешского национального корпуса. Также публичный доступ позволяет увидеть количество вхождений в *SYN2010* и первые 50 примеров употреблений искомого слова в контексте. Слова выдаются в формате *concordance lines,* когда каждая строка – часть текста, в которой содержится искомое слово (рис. 3).

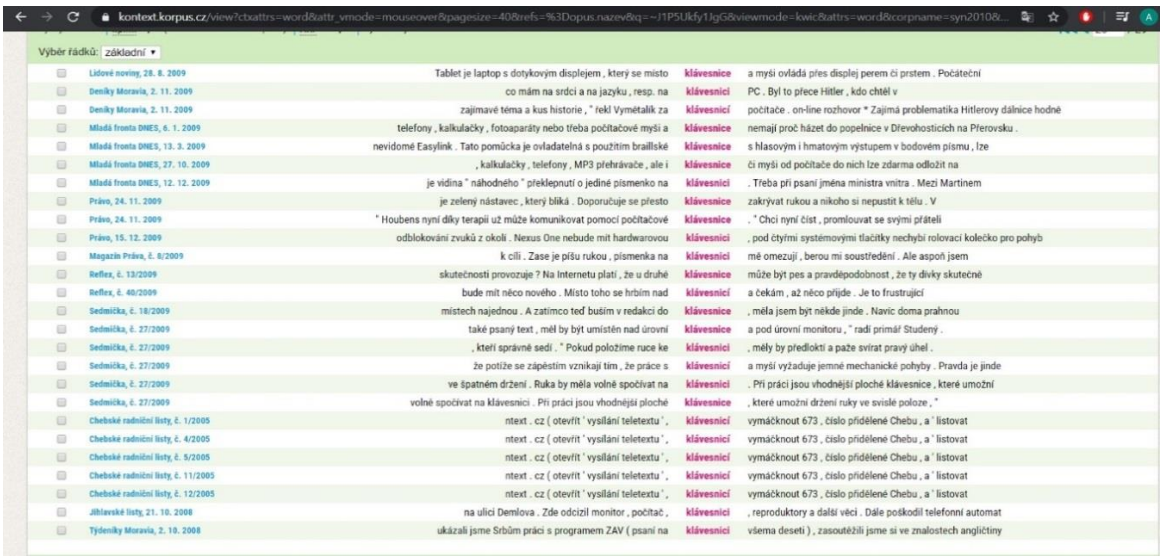

Рис. 3. Поиск слов в чешском национальном корпусе

Модель организации частотного словаря

На основании проведенного анализа электронных словарей и обзора существующих текстовых корпусов были сформулированы требования, которым должен удовлетворять моделируемый словарь:

- словарь должен быть открытым – вся находящаяся в нем информация будет доступна пользователю;

- словарь должен быть расширяемым – база данных имеет возможность дополняться;
- словарь должен иметь простой и понятный пользователю интерфейс;
- возможность использования в режиме *online*;
- возможность вывода статистики по типу произведения;
- возможность вывода статистики по году употребления;
- поиск по слову, вывод найденных данных или сообщение об их отсутствии;
- интеграция с программой распознавания.

Учитывая сформулированные требования, для словаря была выбрана архитектура – клиент-серверное приложение. Приложение клиент-сервер – вычислительная архитектура, в которой задания или сетевая нагрузка распределены между поставщиками услуг, называемыми серверами, и заказчиками услуг, называемыми клиентами. Можно выделить существенные преимущества такого приложения:

- нет необходимости установки программного обеспечения на персональный компьютер, необходим лишь доступ в интернет;
- имеет возможность запуска на любой операционной системе;
- не имеет конкретных требований к технике, на которой будет работать приложение.

Внутреннее устройство частотного словаря представляет из себя базу данных с текстами. Для хранения текстовых данных был выбран формат *JSON*. Преимущества этого формата:

- удобные и быстрые в работе методы, предназначенные для конвертации строки *JSON* в объект и обратно;
- простая и понятная структура данных;
- данные формата *JSON* загружаются быстрее, нежели данные других форматов (например *XML*), за счет того, что формат *JSON* содержит минимальное возможное форматирование, т.к. при его написании используется всего несколько специальных знаков.

Программная реализация

Разработанный программный комплекс представляет собой клиент-серверное приложение, созданное при помощи программной платформы *NodeJS* [5], а также подключенной к ней библиотеке *Express* [6]. Построение происходило с использованием двух языков программирования *JavaScript* и *Python*, а также различных библиотек которые написаны на языке программирования *JavaScript*: *React* – библиотека с открытым исходным кодом, предназначенная для создания пользовательского интерфейса [7]; *React-dom* – библиотека, предназначенная для внесения изменений в пользовательский интерфейс; *React-dom-router* – библиотека с внутренним компонентом *Route*, предназначенная для определения маршрутов приложения. Для хранения данных была выбрана база данных *Mongo DB* – документно-ориентированная база данных, с открытым исходным кодом, не требующая описания схемы таблиц, данные в которой хранятся в виде *JSON* документов [8].

Программный комплекс состоит из трёх модулей:

- главная страница – страница выгрузки данных, содержащихся в базе;
- страница поиска – страница, на которой осуществляется поиск слова и его атрибутов;
- страница добавления данных – страница, на которой осуществляется пополнение базы данных.

Рассмотрим более подробно каждый модуль. На рис. 4 изображена схема, иллюстрирующая работу первого модуля – «Главная страница».

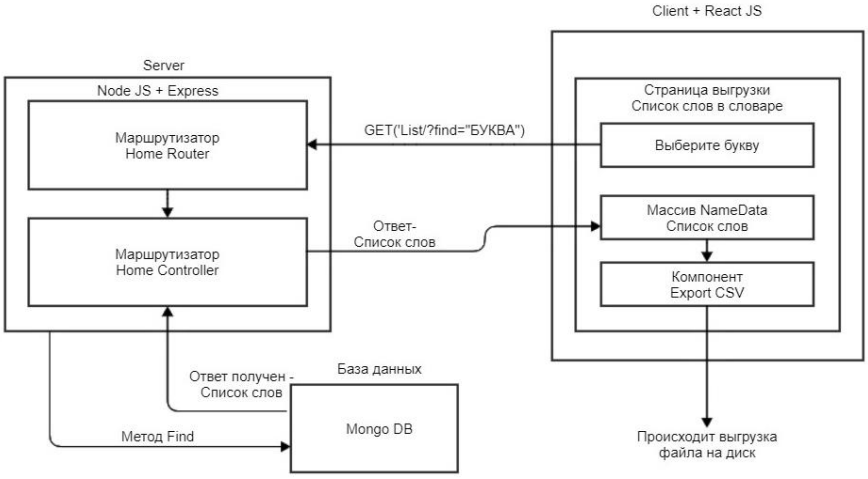

Рис. 4. Схема функционирования модуля «Главная страница»

При помощи переменной *SelectList* создаем форму для выбора буквы, далее используем *handleChange* – функция-обработчик, в которую поступает переменная *event* от элемента *select*, в переменной event содержится параметр *target.value* в котором хранится выбранная буква, которая в последствии присваивается имени файла. На следующем этапе, используя функцию *axios*, происходит вызов маршрутизатора *Home Router*, который, в свою очередь, отправляет запрос контроллеру *Home Controller*, контроллер вызывает метод *Find*, выполняющий поиск в базе данных. Далее база данных отправляет ответ контроллеру *Home Controller*, который перенаправляет данные клиенту, т.е. массиву *NameData*.

Этот массив направляет данные в компонент *ExportCSV* и осуществляет выгрузку данных на диск.

Далее рассмотрим структуру модуля «Поиск данных» (рис. 5). Создается три массива: *Data* – массив, в котором содержатся все найденные слова; *TimeData* – массив, в котором содержатся данные для отображения графика времени; *TypeData* – массив, в котором содержатся данные для отображения графика жанров. Далее вызывается компонент *SearchWord*, который отвечает за поиск слова. В компоненте присутствует *inputbox* – элемент для ввода слова, и кнопка, при нажатии на которую данные из *inputbox* поступают на сервер для последующей обработки, а именно слова поиска в БД.

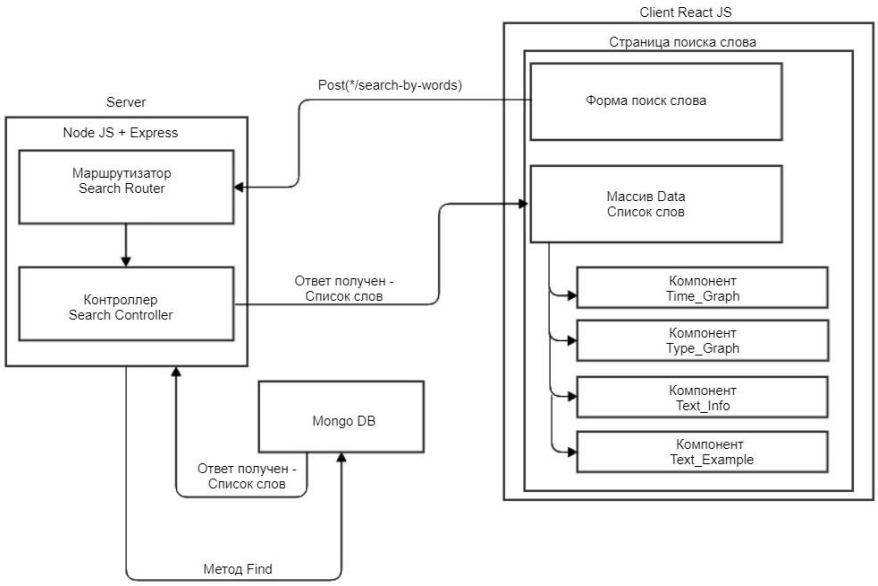

Рис. 5 Схема функционирования модуля «Поиск данных»

При помощи *axios* отправляет данные к серверу. Маршрутизатор Search Router принимает данные от клиента и отсылает их контроллеру *Search Controller*, которые отправляет запрос к БД на

извлечение из БД. В случае, если слово найдено, ответ через контроллер отправляется на массив *Data*. В случае если слово не найдено, то контроллер получает сигнал о том, что искомого слова в базе не существует. Далее полученные данные записываются и сохраняются, после чего идёт обновление системы и производится ранжирование данных при помощи следующих компонентов: *TimeGraph* – отображает график времени; *TypeGraph* – отображает график жанра; *TextExample* – отображает список примеров употребления заданного слова; *TextInfo* – отображает найденное слово. Далее вызывается контроллер *Search Controller*, который в свою очередь вызывает метод *Find* библиотеки *Mongo DB* для выборки данных. После чего клиенту возвращается полученный результат.

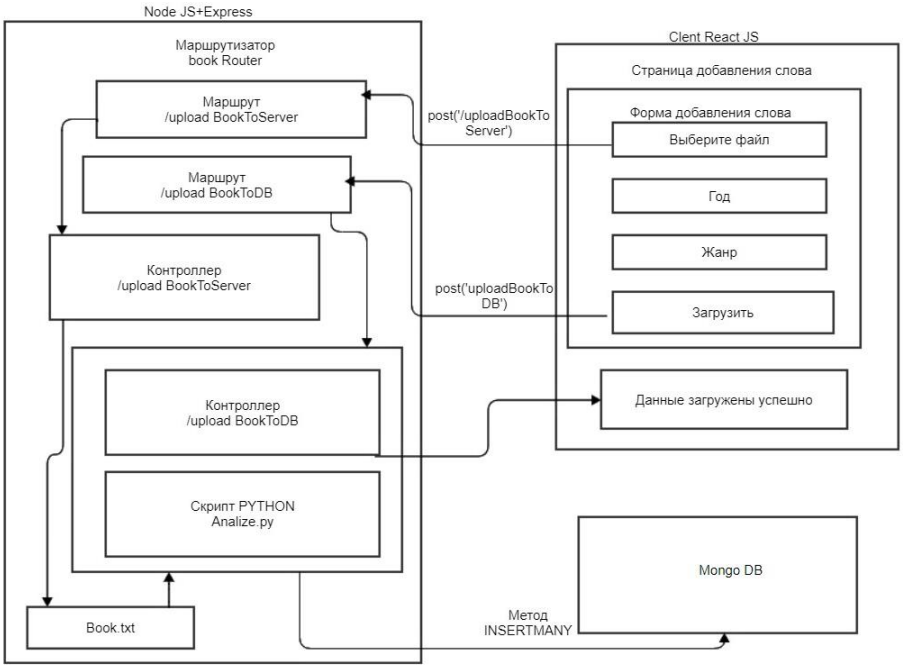

Рис. 6. Схема функционирования модуля «Добавление данных»

Схема, иллюстрирующая работу третьего модуля «Добавление данных» показана на рис. 6. В этом модуле создается компонент для добавления книг *Add_Book*. Далее создаются следующие переменные: *Type* – данные о жанре произведения; *Year* – данные о годе издания книги; *filePath* – переменная, в которой хранится путь до файла, где сохранена книга на сервере. После этого создается форма для сохранения книг и вызывается обработчик *sendBookInfo*, через который данные отправляются на сервер. Далее формируются данные для отправки на сервер при помощи команды *formData.append*, эти данные поступают в контроллер *uploadBookToDB*, где вызывается скрипт *AnalyzeText*, написанный на языке *Python* для обработки загруженного текста.

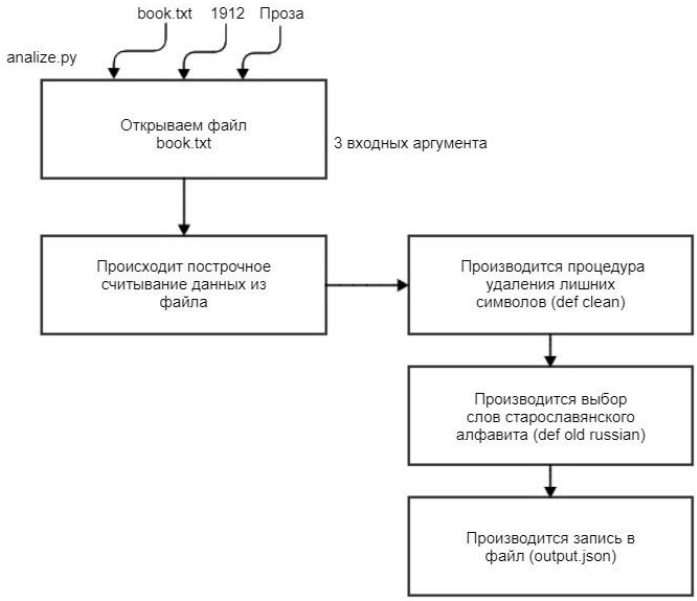

Рис. 7. Схема функционирования скрипта AnalyzeText

Скрипт *AnalyzeText* работает следующим образом (рис. 7): сначала идёт считывание данных из формы загрузки: текст, год, жанр произведения. Далее происходит построчное считывание данных из файла, после чего производится процедура удаления лишних символов, используя функцию *Clean*, и производится выбор слов старославянского алфавита, используя функцию *OldRussian*. Результатом работы скрипта является файл в формате *JSON*, в котором сохранены найденные слова, жанр, год издания книги. Этот файл сохраняется на сервере, который считывается для последующего анализа и добавления данных из него в БД.

Веб-приложение

Одной из задач создания электронного словаря является разработка его веб-приложения. Основной задачей данного приложения является наглядное предоставление пользователю всей необходимой ему информации. Приложение «Дореформенный словарь» включает в себя три страницы: «Главная», «Поиск», «Добавление данных» (рис. 8).

Рис. 8. Главное меню приложения «Дореформенный словарь»

Схема взаимодействия клиента и сервера на странице «Главная» показана на рис. 9.

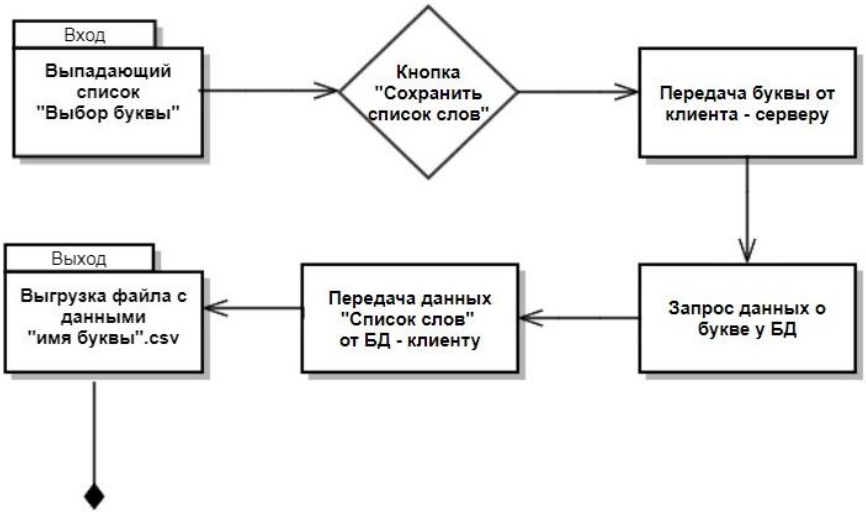

Рис. 9. Схема взаимодействия клиента и сервера на странице «Главная»

«Главная» страница включает в себя выпадающий список с буквами русского дореформенного алфавита, а также кнопку выгрузки – «Сохранить список слов». При помощи выпадающего списка пользователь имеет возможность выбрать интересующую его букву, после чего следует нажать кнопку «Сохранить список слов» и пользователю автоматически выгружается файл формата *csv* всех имеющихся слов в базе начинающихся на выбранную им букву (рис. 10).

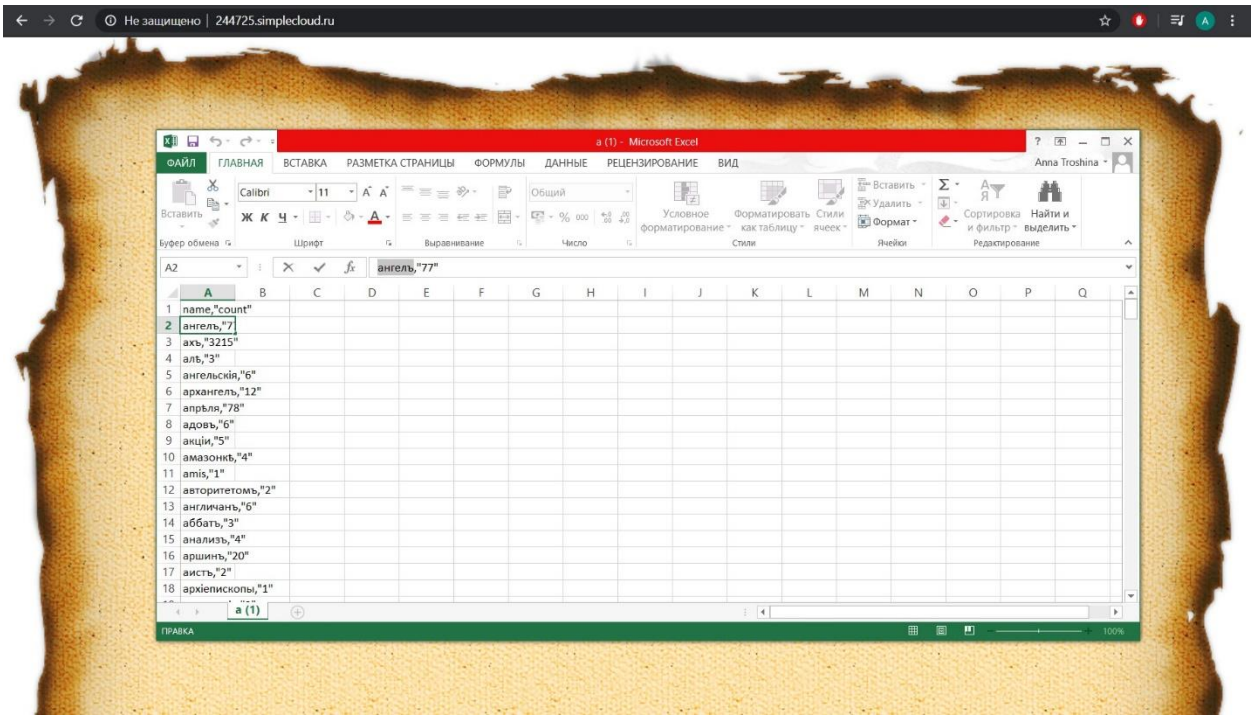

Рис. 10. Загрузка списка слов

На странице «Поиск» пользователь может осуществлять поиск по слову и его атрибутам. В поле ввода необходимо ввести интересующее слово и нажать кнопку «Найти». Схема взаимодействия клиента и сервера на странице «Поиск» показана на рис. 11. В случае, если слово содержится в базе данных, пользователь получает следующую информацию: в левом верхнем углу – искомое слово; в правом верхнем углу – статистика использования искомого слова в год; в левом нижнем углу – использование искомого слова в различных контекстах; в правом нижнем углу – статистика использования искомого слова в различных жанрах литературы (рис. 12).

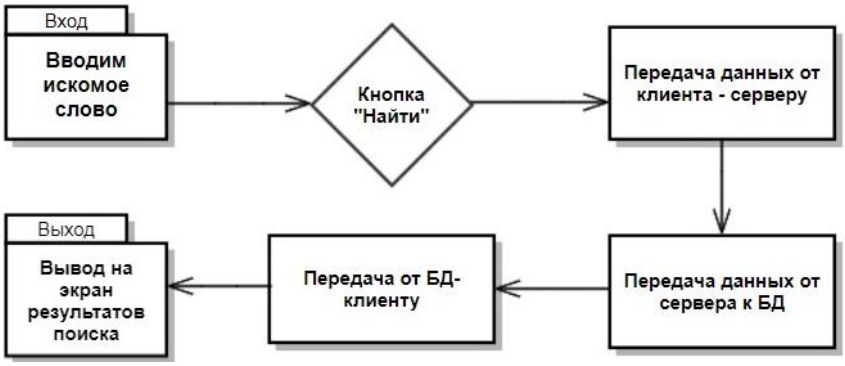

Рис. 11. Схема взаимодействия клиента и сервера на странице «Поиск»

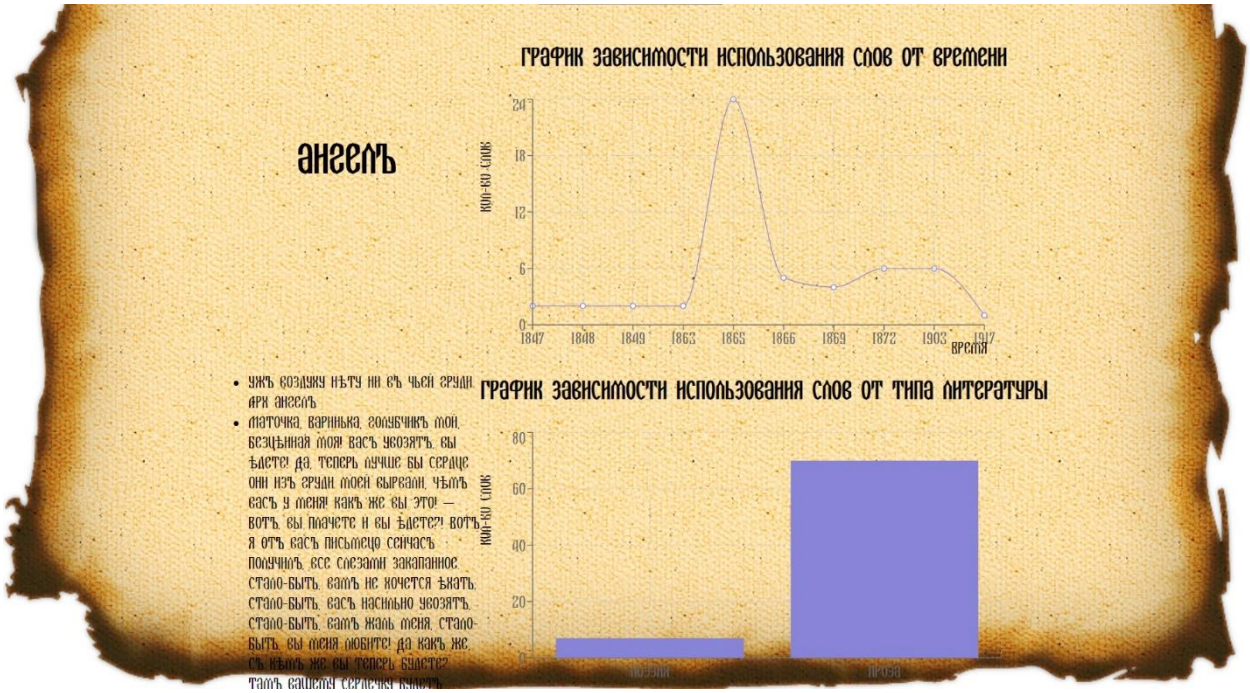

Рис. 12. Результат работы модуля «Поиск»

Страница «Добавление данных» позволяет расширять имеющуюся базу данных новыми словами и текстами. Схема ее функционирования представлена на рис. 13.

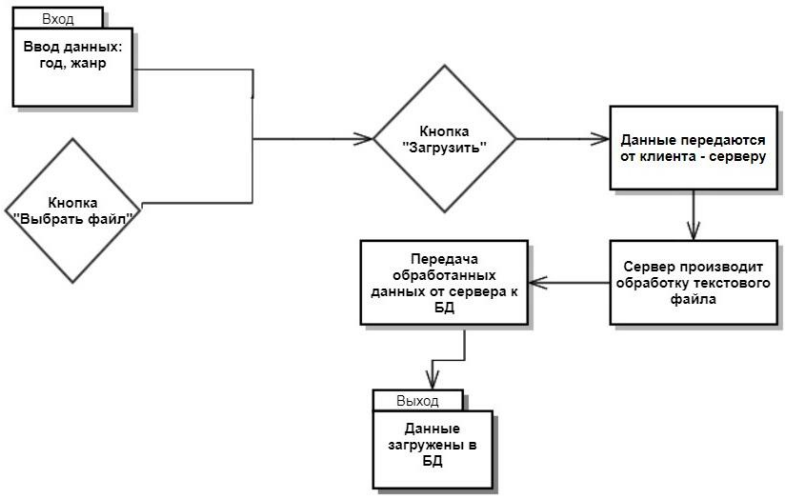

Рис. 13. Схема взаимодействия клиента и сервера на странице «Добавление данных»

На данной странице имеется два поля для ввода данных о загружаемом тексте – года издания и типа источника, а также две кнопки – «Выберите файл» и «Загрузить». Для того, чтобы загрузить новый текст в базу данных необходимо нажать кнопку «Выберите файл», после чего выбрать необходимый файл в формате *txt*. После загрузки файла необходимо указать год и тип в соответствующих полях ввода, затем нажать кнопку «Загрузить». После того, как данные загрузились, пользователь получает оповещение о том, что «Данные добавлены в БД» (рис. 14).

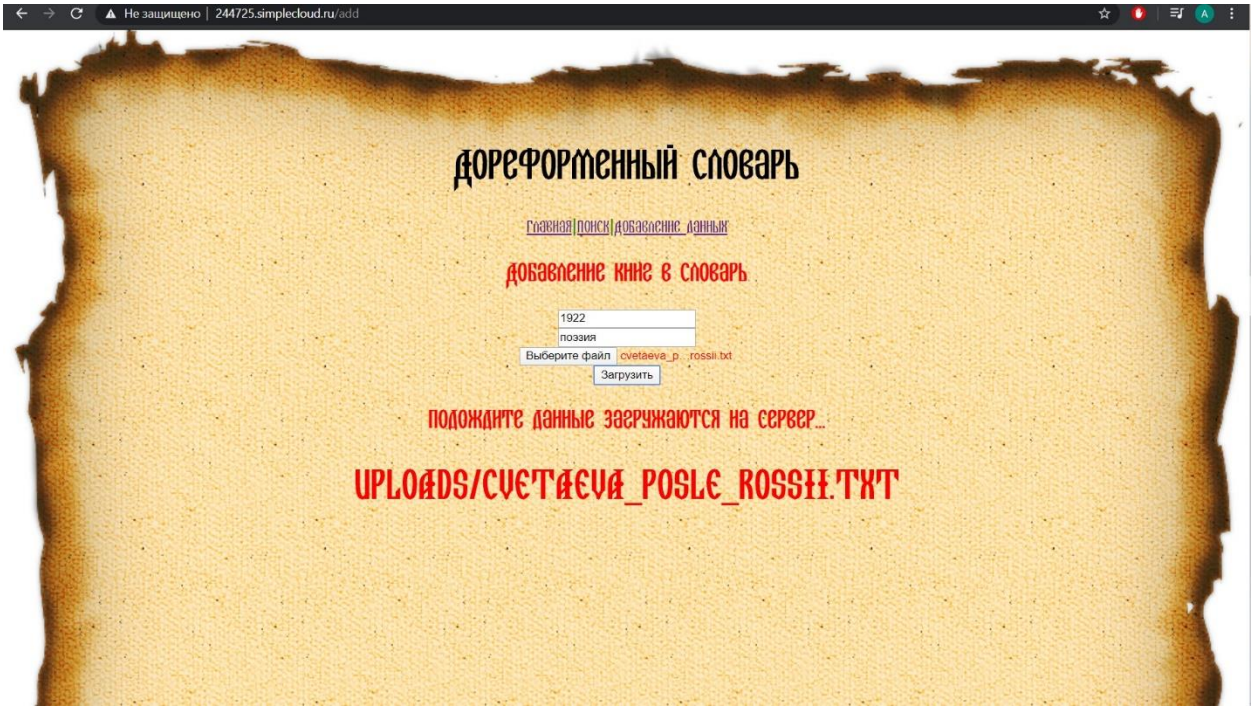

Рис. 14. Загрузка данных на сервер

Заключение

В результате выполненной работы получены следующие результаты.

- Проведен обзор и анализ существующих национальных текстовых корпусов.
- Разработана модель веб-приложения «Дореформенный словарь» с использованием двух языков программирования JavaScript и Python, а также библиотек React, React-dom и React-dom-router. Для хранения данных в приложении используется база данных MongoDB.
- Реализованы программная и интерфейсная часть веб-приложения. Пользователь приложения имеет возможность доступа ко всему перечню имеющихся слов и их атрибутов, таких как: статистика употребления искомого слова в различных типах произведения (проза, поэзия, научная литература, публицистика), частота употребления искомого слова в различные годы, примеры употребления искомого слова в различных контекстах. Кроме того, приложение даёт возможность пользователю осуществлять пополнение базы данных новыми текстами.
- Выполнено начальное наполнение базы данных художественными и поэтическим текстами 18-19 вв. в исходной дореформенной орфографии, найденными в открытых источниках.

Список литературы

1. Захаров, В. П., Богданова, С. Ю. Корпусная лингвистика: Учебник для студентов направления «Лингвистика». – СПб. : СПбГУ. РИО. Филологический факультет, 2013. – 148 с.
2. Национальный корпус русского языка : сайт. – URL : http://www.ruscorpora.ru (дата обращения: 3.07.2020).
3. British National Corpus : сайт. – URL : http://www.natcorp.ox.ac.uk (дата обращения: 3.07.2020).
4. Чешский национальный корпус : сайт. – URL : http://ucnk.ff.cuni.cz (дата обращения: 3.07.2020).
5. Руководство по использованию *Node.js*. – URL : https://metanit.com/web/nodejs/1.2.php (дата обращения: 3.07.2020).
6. Руководство по использованию *React.js*. – URL : https://metanit.com/web/nodejs/1.2.php (дата обращения: 3.07.2020).
7. Руководство по использованию *Express.js*. – URL : https://expressjs.com/ru/guide/routing.html (дата обращения: 3.07.2020).
8. Руководство по использованию *MongoDB*. – URL : https://metanit.com/nosql/mongodb/1.1.php (дата обращения: 3.07.2020).